Karlsruher Institut für Technologie

Design, Implementierung und Evaluierung eines Web-Editors für Research Object Crates

Bachelorarbeit von

Christopher Raquet

An der KIT-Fakultät für Informatik
SCC - Scientific Computing Center

Erster Prüfer:	Prof. Dr. Achim Streit
Zweiter Prüfer:	Prof. Dr. Veit Hagenmeyer
Betreuerin:	M.Sc. Sabrine Chelbi

16. Mai 2024 – 16. September 2024

Karlsruher Institut für Technologie
Fakultät für Informatik
Postfach 6980
76128 Karlsruhe

Ich versichere wahrheitsgemäß, die Arbeit selbstständig verfasst, alle benutzten Quellen und Hilfsmittel vollständig und genau angegeben und alles kenntlich gemacht zu haben, was aus Arbeiten anderer unverändert oder mit Abänderungen entnommen wurde sowie die Satzung des KIT zur Sicherung guter wissenschaftlicher Praxis in der jeweils gültigen Fassung beachtet zu haben.

Karlsruhe, 16. September 2024

. . . . . .

(Christopher Raquet)

Zusammenfassung

Research Object Crates sind eine neue Methode, um den einfachen Austausch von Forschungsdaten sowie die Reproduzierbarkeit wissenschaftlicher Arbeiten zu unterstützen. Dabei werden Forschungsdaten zusammen mit ihren Metadaten paketiert und damit in einer einzelnen Archivdatei aggregiert. Das so entstehende RO-Crate ist in sich geschlossen und selbstbeschreibend. Um dieses Format für Forschende zugänglicher zu machen, werden Anwendungen zum Erstellen, Bearbeiten und Lesen von RO-Crates benötigt. Während bereits unzählige Software-Bibliotheken zu diesem Zweck existieren, fehlt es an intuitiven und benutzbaren Editoren. Im Rahmen dieser Arbeit wurde der Web-Editor „NovaCrate“ entwickelt, welcher die benutzerfreundliche Erstellung und Bearbeitung von RO-Crates ermöglicht. Dieser wurde mithilfe von modernen Web-Technologien wie Next.js und React implementiert. In der Arbeit wird, anhand klar definierter funktionaler und nichtfunktionaler Anforderungen, im Detail auf den Entwurfs- und Entwicklungsprozess eingegangen, mit einer ausführlichen Erläuterung besonders interessanter und innovativer Komponenten. Dabei wird nicht nur die technische Architektur und das Design betrachtet, sondern auch ein Einblick in die tatsächliche Implementierung gegeben. Mithilfe eines Nutzertests mit fünf Testpersonen wurde die Erfüllung der Anforderungen an den Editor evaluiert und diskutiert, wie der Editor in Zukunft eingesetzt werden kann. Der Großteil der Anforderungen konnte erfüllt werden, sodass nur granulare Anpassungen nötig sein werden, um einen produktiven Standard zu erreichen. NovaCrate könnte somit ein wichtiges neues Werkzeug für die Erstellung und Bearbeitung von RO-Crates werden.

Abstract

Research Object Crates are a new method for enabling easy exchange of research data and supporting the reproducibility of scientific works. RO-Crates are created by packaging research data and their metadata together, aggregating them in a single archive. They are self-enclosed and self-describing. To enable researchers to benefit from this format, an editor is required for creating, editing and reading RO-Crates. While dozens of software libraries with these capabilities already exist, no easy and usable editor has been created yet. As part of this thesis, the Web-Editor „NovaCrate“ was developed, with the goal of enabling user-friendly creating and editing of RO-Crates. The editor was developed based on modern technologies like Next.js and React. In this thesis, the design and implementation of the editor are described in detail, based on defined functional and non-functional requirements, with great attention to interesting and innovative mechanisms. Not only the technical architecture and design are covered, but also the implementation. Subsequently, a user test was conducted to examine and discuss the requirements as well as the possible use-cases of the editor in the future. The majority of the requirements has been met, so only slight adjustments will be necessary for the editor to reach a productive quality. As such, NovaCrate could become an important new tool for working with RO-Crates.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1. Einleitung

In dieser Einleitung wird eine Motivation zum Thema der Arbeit gegeben und anschließend dessen Ziele und Aufbau vorgestellt.

1.1. Motivation

Forschungsdaten werden heutzutage immer häufiger digital veröffentlicht. Die altbewährte Methode, Forschungsergebnisse in Form von wissenschaftlichen Papieren zu teilen, geriet im digitalen Umfeld jedoch an ihre Grenzen. Ein zentraler Aspekt beim Austausch von Forschungsarbeiten ist die Reproduzierbarkeit der Ergebnisse. Dafür ist es nötig, Konfigurationsdateien, Datensätze oder sonstige Artefakte zusammen mit der wissenschaftlichen Arbeit zu teilen. Mit klassischen Medien, wie Papieren oder Präsentationen, war dies bis jetzt nicht einfach möglich, selbst wenn diese digital bereitgestellt werden. [1, 45]

Das Ziel von Research Objects ist es, wiederverwendbare Artefakte der Forschung zusammen mit ihren wissenschaftlichen Erkenntnissen zu kapseln. Die dabei entstehenden Pakete sollen frei austauschbar sein und für Maschine als auch Mensch zugänglich sein. Eine Spezifikation im Rahmen der Research Objects sind die Research Object Crates (RO-Crate), welche Forschungsdaten in einem Archiv paketieren und mit Metadaten versehen. [48]

Research Objects, und damit auch Research Object Crates, stützen sich dabei auf einige zentrale Prinzipien. Die folgenden sind besonders relevant: Identität, Aggregation und Annotation. Völlig unabhängig von den letztendlich eingesetzten Mechanismen sollen diese Prinzipien die Alleinstellungsmerkmale von Research Objects definieren. [48, 1]

Identität Jedes Research Object soll eine Identität in Form eines eindeutigen Bezeichners haben. Dadurch wird das Research Object referenzierbar und lässt sich in klassische Publikationen integrieren. [48, 1]

Aggregation Das Research Object soll alle zugehörigen Artefakte an einem Ort aggregieren und so für jeden bereitstellen. Für jedes Werk können somit alle wichtigen Artefakte mitveröffentlicht werden, um jedem das Prüfen und Nachvollziehen der Ergebnisse zu ermöglichen. [48, 1]

Annotation Die im Research Object enthaltenen Artefakte sollen mit zusätzlichen Metadaten versehen werden, um sie vollumfänglich zu beschreiben. Dazu gehören nicht nur klassische Metadaten, wie Dateiart oder Dateigröße, sondern zum Beispiel auch woher das Artefakt kommt und mit welchen Mitteln es entstanden ist. [48, 1]

In dieser Arbeit soll ein einfach zu erlernender, intuitiver Web-Editor zur Bearbeitung und Erstellung von RO-Crates entwickelt werden. Während bereits unzählige Bibliotheken zur programmatischen Verwendung von RO-Crates existieren, gibt es nach aktuellem Stand nur zwei RO-Crate Editoren. Der in dieser Arbeit entwickelte Editor soll dabei auf die Stärken der existierenden Editoren aufbauen und neue, innovative Wege finden, um die Bearbeitung von RO-Crates durch den Menschen noch einfacher und intuitiver zu gestalten.

Die Spezifikation der RO-Crates wird in Kapitel 2 genauer vorgestellt.

1.2. Ziel der Arbeit

Das Ziel dieser Bachelorarbeit ist das Design, die Implementierung sowie die Evaluation eines Web-Editors für Research Object Crates. Der entwickelte Editor soll einfach zu erlernen und intuitiv in der Verwendung sein. Der Funktionsumfang soll dabei dem der zugrunde liegenden Java-Bibliothek ro-crate-java [47] entsprechen. Dabei soll der Editor in der Lage sein, existierende Crates frei zu bearbeiten, als auch neue Crates von Grund auf Aufzubauen. Des Weiteren soll es möglich sein, den Inhalt des Crates mittels eines Graphen zu visualisieren. Weitere Ziele sind eine gute Wartbarkeit und Erweiterbarkeit, welche in Kapitel 5 genauer betrachtet werden.

Die Arbeit soll darüber hinaus einen tiefen Einblick in den gesamten Entwicklungsprozess bieten. Da die Entwicklung des Editors die Hauptaufgabe dieser Arbeit darstellt, wird dem Design des Editors, sowie der Implementierung, besonders viel Aufmerksamkeit geschenkt. Hier sollen die gefällten Designentscheidungen wiedergegeben und begründet werden, teils auch in Abwägung mit alternativen Designmöglichkeiten.

Auch die Evaluation ist eines der Hauptziele dieser Arbeit. Um zu prüfen, ob der Editor die Anforderungen erfüllt, wird ein Nutzertest nach Krug [12] durchgeführt. Dafür folgt zunächst eine Beschreibung der Methodik des Nutzertests, sowie eine Aufarbeitung der entstandenen Transkripte. Die tatsächliche Evaluation des Editors auf Basis des Nutzertests rundet die Arbeit ab und stellt fest, ob die Anforderungen eingehalten werden konnten.

1.3. Struktur der Arbeit

Nach dieser Einleitung wird in Kapitel 2 zunächst die Research Object Crate Spezifikation eingeführt. Dieses enthält das Basiswissen zum Verständnis der folgenden Kapitel.

In Kapitel 3 wird anschließend ein Überblick über den Stand der Technik, also bereits existierende RO-Crate Editoren [13, 37], gegeben. Hier werden Probleme der Benutzbarkeit sowie technische Stärken der bestehenden Editoren hervorgehoben.

Die Einführung in die technischen Grundlagen erfolgt in Kapitel 4. Die RO-Crate Java-Bibliothek [47] und die darauf aufbauende REST-API [36] werden vorgestellt und deren Relevanz für den Editor herausgehoben. Darauf folgt ein Überblick über die im neuen

Editor verwendeten Frameworks und Bibliotheken. Um im vierten Kapitel genauer auf die Design- und Implementationsdetails eingehen zu können, wird hierbei auch teils im Detail die Aufgabe des Next.js Web-Frameworks [53] und der Frontend-Bibliothek React [19] erläutert.

Kapitel 5 befasst sich mit dem Design und der Implementierung des Editors, sowie der Erweiterung der zugrunde liegenden REST-API [36]. Zunächst wird dabei die technische Architektur vorgestellt und deren Wahl begründet. Anschließend werden einige wichtige Bestandteile der technischen Architektur im Detail vorgestellt. Diese bieten besondere Vorteile in der Wartbarkeit und Erweiterbarkeit. Daraufhin werden interessante Komponenten und nötige Hilfsmittel der Implementierung vorgestellt. Zusätzlich wird den nötigen React [19] Performanceoptimierungen ein Unterabschnitt gewürdigt.

Die Evaluation des Editors findet schließlich in Kapitel 6 statt. Hier wird mittels eines Nutzertests nach Krug [12] die Benutzbarkeit des Editors qualitativ ermittelt. Dabei wird der Begriff der Benutzbarkeit nach Nielsen [26] und Beval et al [2] definiert. Im Anschluss folgt ein Abschnitt über die Durchführung des Nutzertests. Darauf folgt die Auswertung des Nutzertests, wobei auf Grundlage der Transkripte evaluiert werden soll, ob der Editor die definierten Ziele erreicht hat. Dies stellt insbesondere eine Evaluation der in Abschnitt 1.2 genannten Anforderungen für diese Arbeit dar.

In Kapitel 7 wird schließlich über die Ergebnisse aus dem fünften Kapitel diskutiert. Außerdem werden einige weitere Ideen formuliert, die den Editor in Zukunft noch bereichern könnten.

Das Fazit sowie eine Aussicht auf weiteres Entwicklungspotenzial wird abschließend in Kapitel 8 gegeben.

2. Research Object Crate Spezifikation

Die Research Object Crate Spezifikation [51] definiert zum einen, wie ein Crate aufgebaut ist, also auch wie die Metadaten in der Metadaten-Datei zu strukturieren sind (vgl. Abbildung 2.1) [52, 49]. Ein Crate ist dabei ein Archiv, hier beispielsweise ein ZIP-Archiv, in dem beliebige Dateien und Ordner paketiert sind. Diese Daten stellen den Inhalt des Crates dar. Zusätzlich liegt noch eine Metadaten-Datei im Archiv welche die Metadaten der Dateien und Ordner des Crates enthält. Die Vorschaudateien des Crates sind hier nicht weiter relevant.

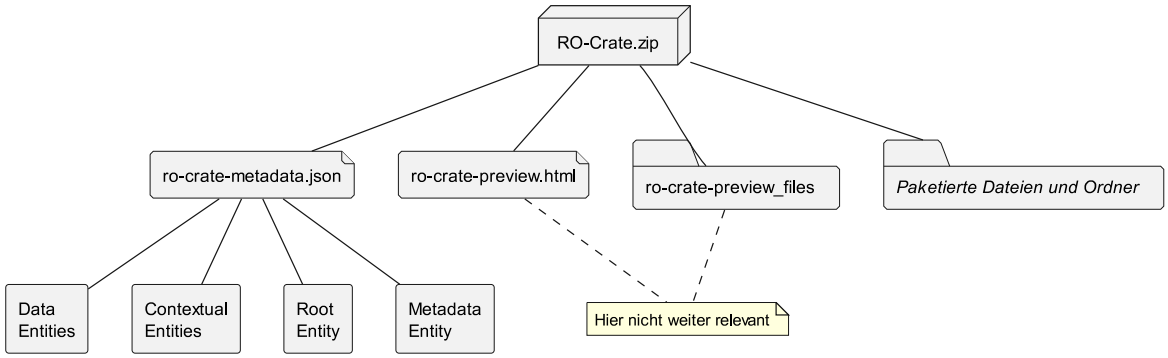

Abbildung 2.1.: Aufbau eines Research Object Crates

Die Metadaten-Datei `ro-crate-metadata.json` liegt auf der obersten Ebene des Archivs. Hierbei handelt es sich um eine flache JSON Datei nach dem Linked-Data Prinzip [55] (JSON-LD). Alle Entitäten (im Weiteren auch als „Entities“ oder „Entity“ bezeichnet) liegen dabei in einem flachen Array, genannt `@graph`. Jede Entität hat einen eindeutigen Bezeichner. Über diesen eindeutigen Bezeichner können Entitäten andere Entitäten referenzieren. Dabei können nicht nur lokale Bezeichner verwendet werden, sondern auch URLs für externe Ressourcen. Ein unvollständiges Beispiel einer Metadaten-Datei ist in Listing 2.1 dargestellt. [52, 49]

Grundlegend dienen die Entitäten in der Metadaten-Daten der Beschreibung der paketierten Daten. Jeder Datei bzw. jedem Ordner im Crate kann höchstens eine Entität zugeordnet sein. Eine Entity, die eine Datei oder einen Ordner beschreibt, ist eine Data Entity mit dem Typ „File“ oder „Dataset“. [51]

Zusätzlich können Kontextinformationen gegeben werden, zum Beispiel zu den involvierten Autoren, Organisationen oder Softwares. Eine Entity, welche Kontextinformationen

```
{   "@context" : [ "https://w3id.org/ro/crate/1.1/context", {
        "myCustomExtension" : "https://example.org/exampleProperty"
    } ],
    "@graph": [
        {
            "@id": "someFile.pdf",
            "@type": "File",
            "author": { "@id": "#beispiel-bernd" }
        },
        {
            "@id": "#beispiel-bernd",
            "@type": "Person",
            "givenName": "Bernd",
            "familyName": "Beispiel"
        }
] }
```

Listing 2.1: Beispielhafte, unvollständige Metadaten-Datei eines Crates

repräsentiert, ist eine Contextual Entity. Der Entity-Typ ist dabei beispielsweise „Person“, „Organisation“ oder auch „CreativeWork“. [51]

Außerdem gibt es die Root Entity. Diese assoziiert das Crate selbst mit Metadaten. Auch die Metadaten-Datei kann mit Metadaten versehen werden, über die sogenannte Metadaten Entität. Typischerweise bleibt die Metadaten Entität jedoch unberührt. [51]

Zur Definition der Entity-Typen nutzen Research Object Crates vor allem die „Schema.org“ [35] Schemata. Das RO-Crate Schema nimmt hierbei lediglich einige Umbenennungen vor. Im Normalfall entsprechen beispielsweise die Eigenschaften eine Entity mit dem Typ „Person“ genau der Definition des Schemas unter `https://schema.org/Person`. In der Metadaten-Datei ist dazu, ebenfalls nach dem Linked-Data Prinzip [55], der `@context`-Schlüssel gegeben. Dessen Wert gibt an, welchem Schema die Metadaten-Datei folgt. Außerdem kann das Schema hier lokal erweitert werden. Ein Beispiel hierzu ist in Listing 2.1 dargestellt. [51]

Darüber hinaus enthält die Spezifikation noch viele weitere Empfehlungen für den Umgang mit spezifischen Typen oder die korrekte Wahl der Bezeichner. Diese sollen hier jedoch nicht weiter ausgeführt werden.

3. Stand der Technik

Es existieren bereits zwei Web-Editoren, die das Erstellen und Bearbeiten von RO-Crates ermöglichen. Aktuell sind sie, abseits von manueller Erstellung, die einzigen an Endbenutzer gerichteten Möglichkeiten, um ein RO-Crate zu manipulieren. Da sie somit den aktuellen Stand der Technik in Bezug auf RO-Crate Editoren definieren, bedarf es eines kurzen Überblicks über ihre Stärken und Schwächen.

In den folgenden Abschnitten werden die bestehenden Editoren vorgestellt und ihre Vor- und Nachteile kurz beschrieben.

3.1. Describo

Describo ist ein intuitiver, intelligenter und erweiterbarer Metadaten-Editor [16]. Seit 2020 ist dieser in aktiver Entwicklung. Describo war lang der einzige interaktive Editor für Research Object Crates [41, S. 110f]. Der Editor kann entweder in Chromium-basierten Browsern wie Chrome oder Edge, aber auch als Desktop-App auf allen gängigen Betriebssystemen verwendet werden. Er ist in JavaScript mit dem Vue.js-Framework [58] geschrieben. Describo operiert direkt auf einem lokalen Ordner, in den auch die entsprechende RO-Crate Metadaten-Datei gespeichert wird. Für diesen Abschnitt wurde Describo Desktop in der Version 0.37.0 verwendet [13].

Der grundlegende Arbeitsablauf mit Describo ist relativ simpel. Zunächst wählt man einen Ordner aus dem eigenen Dateisystem aus. Anschließend werden alle Dateien in dem Ordner aufgelistet. Eine Bildschirmaufnahme von Describo mit einigen Dateien ist in Abbildung 3.1 zu sehen. Diese können dann ausgewählt werden, um die zugehörige Data Entity zu erstellen und mit Eigenschaften zu versehen. Hierfür werden direkt einige Metadaten von der Datei selbst übernommen, wie zum Beispiel die Größe oder Art der Datei. Falls möglich, wird eine Vorschau der Datei auf der rechten Seite angezeigt. Der Nutzer kann nun selbst Entities und Eigenschaften hinzufügen und diese untereinander verlinken. Texteigenschaften können über ein einfaches Textfeld eingegeben werden, wobei die Änderungen immer direkt im Anschluss gespeichert werden. Für Datentypen wie Datums- oder Zeitangaben stehen spezialisierte Eingabefelder zur Verfügung. Referenzen können durch ein Suchfeld hinzugefügt werden. Existiert eine Entity mit den gesuchten Namen, so kann diese direkt ausgewählt werden. Falls keine solche Entity existiert, wird eine neue mit dem eingegebenen Namen erstellt.

Über diesen grundlegenden Editor hinaus bietet Describo einige Features mit künstlicher Intelligenz. Dazu gehört das automatische Extrahieren von Metadaten aus Bildern über

optische Zeichenerkennung (OCR), aber auch ein künstlicher Assistent, der wichtige Inhalte und Zusammenhänge aus einem Text extrahieren soll. Für die Nutzung der KI-gestützten Funktionen ist jedoch eine Zahlung nötig, um die nötigen „AI-Credits“ zu erhalten [15].

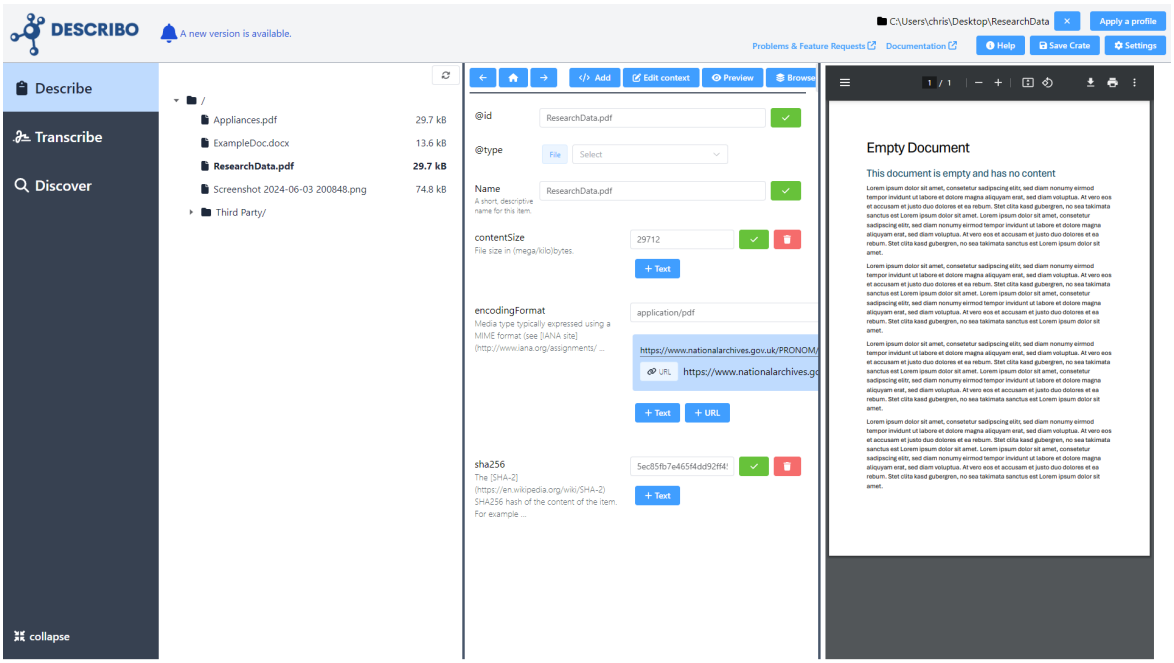

Abbildung 3.1.: Ein Ordner mit beispielhaften Daten, geöffnet mit Describo Desktop [13]

Außerdem unterstützt Describo sogenannte „Profiles“. Dadurch werden die Eigenschaften einer Entity in Gruppen unterteilt und spezifischere Anforderungen angewandt. Es erscheinen zudem Hinweise für erforderliche Eigenschaften (Siehe Abbildung 3.2).

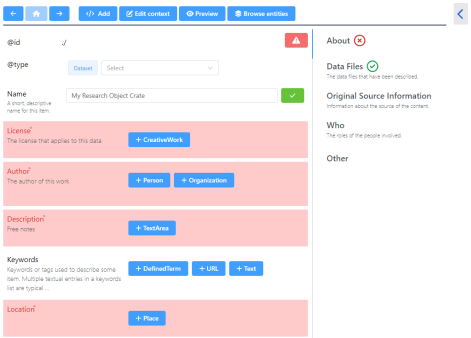

Abbildung 3.2.: Describo Desktop [13] mit einem angewendeten Profil

Describo Web ist nur relativ schwer über einen Link im Footer der Homepage zu finden. Der Web-Edition fehlen viele der Funktionen der Desktop-Variante. So ist hier kein Datei-Explorer mit Dateivorschau zu finden, auch gibt es keine KI-gestützten Features. Interessanter ist die Describo Crate Builder Komponente. Hierbei handelt es sich um eine

Vue-Komponente, die es erlaubt die grundlegenden Funktionen von Describo (entsprechen dem Funktionsumfang des Web-Editors) in andere Web-Services zu integrieren [42, Abschn. 2.6][14].

3.1.1. Vorteile

Describo bietet alle nötigen Funktionen, um ein Crate vollumfänglich erstellen und bearbeiten zu können. Durch die separat gekapselte Describo Engine, lässt sich Describo nicht nur als Desktop Anwendung, sondern auch im Web oder in Form von einzelnen Komponenten verwenden. Dadurch ist Describo sehr vielfältig einsetzbar und für verschiedenste Zwecke geeignet.

Es gibt außerdem einige Hilfsmittel, um den Nutzenden beispielsweise die Erstellung von „Place“ (dt. Ort) Entities zu vereinfachen. Hier wird dem Nutzer eine Landkarte angezeigt, von der direkt ein Ort ausgewählt werden kann.

Die KI-Funktionalitäten bieten außerdem eine spannende Möglichkeit, um den Anwender*innen lästiges Abschreiben abzunehmen. Hier wird zunächst per OCR der Text in einem Bild transkribiert, und kann anschließend im Gespräch mit einer KI genauer untersucht werden. Diese sind jedoch nur in der Desktop-Version und gegen Bezahlung verfügbar.

3.1.2. Nachteile

Neben seinen stärken hat Describo auch ein paar Schwächen, vor allem in der Benutzbarkeit. In diesem Abschnitt werden diese Probleme etwas genauer beschrieben.

3.1.2.1. Benutzerfreundlichkeit

Als Grundlage für die Beurteilung der Benutzerfreundlichkeit von Describo (als auch Crate-O, welches in der nächsten Sektion vorgestellt wird), dienen die einschlägigen Bücher von Steve Krug [11], Jenifer Tidwell et al [46] und Shneiderman et al [40].

Eines der zentralsten Probleme abseits des Designs ist die Navigation zwischen den Entities im Crate. Tidwell stellt einige Patterns für den Aufbau einer Navigation auf [46, S. 135f]. Describo verwendet eine „Tree-Navigation“, bei der einige Entities direkt über den Datei-Explorer erreichbar sind. Dies entspricht der obersten Ebene. Alle Contextual Entities, welche an die Data Entities angeknüpft sind, sind nur über diese aufzufinden. Hierbei ergibt sich eine Baumstruktur mit beliebig tiefer Verschachtelung. Das grundlegende Problem ist hierbei, dass der Nutzer sich merken muss, über welchen Pfad er eine bestimmte Entity erreicht. Alle Contextual Entities lassen sich schließlich nur über eine solche Verlinkung finden, ebenso wie Data Entities die keine zugehörige Datei im Crate-Ordner haben. Alternativ lässt sich eine vollständige aber unsortierte Liste aller Entities aufklappen.

Der Knopf der zur Liste aller Entities im Crate führt, ist in der zentralen Spalte angesiedelt. Hier wird die aktuelle Entity angezeigt. Dies leitet direkt zu einem nächsten möglichen Kritikpunkt über, nämlich der Platzierung und Beschriftung der Knöpfe. Die meisten von ihnen sind in der besagten zentralen Spalte untergebracht. Hier sind scheinbar ohne Unterteilung oder Abgrenzung alle Knöpfe untergebracht, die entweder die aktuelle Entity oder auch das aktuelle Crate beeinflussen (Siehe Abbildung 3.3). Darunter finden sich beispielsweise Knöpfe mit der Beschriftung „Add“ (dt. Hinzufügen) und „Preview“ (dt. Vorschau). Dabei dient der Hinzufügen-Knopf dem Hinzufügen einer Eigenschaft zur aktuellen Entity und der Vorschau-Knopf zeigt eine Vorschau der Metadaten-Datei des gesamten Crates an. Einen dedizierten Knopf zum Hinzufügen einer Entity gibt es nicht, dies ist nur über Referenzfelder möglich.

Abbildung 3.3.: Knöpfe, die sich am oberen Rand der mittleren Spalte von Describo Desktop [13] befinden

Ein weiterer Mangel ist der Spalten-Artige Aufbau von Describo. Im Normalfall werden 4 Spalten nebeneinander angezeigt (siehe Abbildung 3.1). Auf einem Full-HD Bildschirm kommt es hier bereits zu Schwierigkeiten mit der Anzeige. Wendet man schließlich noch ein Profil auf das Crate an, werden es sogar 5 Spalten. Dies könnte zu Problemen der Benutzbarkeit auf Bildschirmen mit Full-HD-Auflösung oder kleiner sorgen.

3.1.2.2. Technische Herausforderungen

An einigen Stellen treten noch technische Probleme auf. Diese reichen von recht banalen Problemen, wie der doppelten Auflistung mancher Entity-Typen, hin zu gravierenderen Problemen, wie dass Entities mit einem eigens erstellten Typ nicht gespeichert werden. Auch eigene Kontext-Erweiterungen werden derzeit von Describo nicht beachtet. Ein Kontextmenü zum Kopieren oder Einfügen von Text ist ebenfalls noch nicht vorhanden.

3.2. Crate-O

Crate-O wird derzeit an der Universität von Queensland in Australien entwickelt und ist eine Neuaufsetzung von Describo [38]. Bereits jetzt wird Crate-O in der Community als Nachfolger bzw. Alternative zu Describo bezeichnet [42, Abschn. 2.4]. Crate-O ist in der Verwendung als auch im Design recht ähnlich zu Describo. Für diesen Abschnitt wurde Crate-O im Edge-Browser in der Version 0.3.10 verwendet [37]. Auch Crate-O ist in JavaScript mit dem Vue-Framework [58] geschrieben, allerdings gibt es keine Desktop-Version. Genau wie Describo lässt sich dieser Editor aber nur in Chromium-basierten Browsern wie Edge oder Chrome verwenden. Einige bemerkenswerte Unterschiede gibt es dennoch.

Der grobe Aufbau von Crate-O ist aufgeräumter als der Describos (Siehe Abbildung 3.4). Am oberen Bildschirmrand werden einige allgemeine Funktionen angeboten. Der Großteil des Fensters wird von der aktuell ausgewählten Entität ausgefüllt. Auf der rechten Seite ist eine Liste aller Entitäten zu sehen.

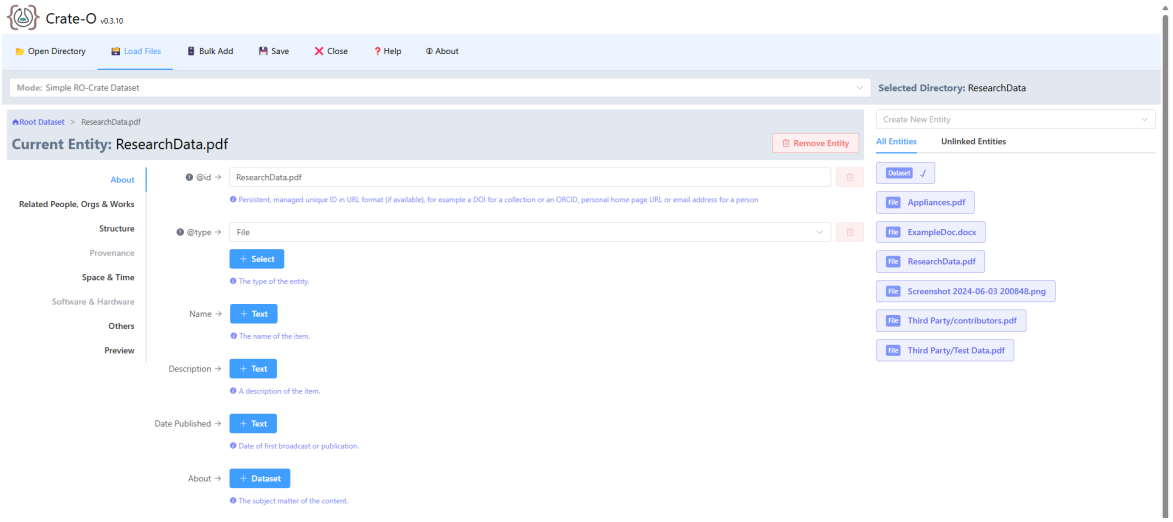

Abbildung 3.4.: Ein Ordner mit beispielhaften Daten, geöffnet mit Crate-O [37]

Crate-O gruppiert außerdem die Eigenschaften einer Entity nach bestimmten Überthemen (Siehe Abbildung 3.5). Anders als bei Describo sind hier insgesamt Themen mehr untergebracht, wobei auch Themen angezeigt werden, die aktuelle keinen Inhalt zeigen können (im Abbildung 3.5 ausgegraut). Die Root Entity hat unter anderem die Gruppierungen „About“, „Structure“ oder „Space & Time“. In diesen Gruppierungen wird eine feste Menge an Eigenschaften angezeigt. Je nachdem, in welchem Modus man den Editor betreibt, stehen andere Eigenschaften zur Verfügung. Hier gibt es neben der Möglichkeit, das vollständige RO-Crate Schema auszunutzen, auch zugeschnittene Modi für bestimmte Zwecke. Es werden jedoch immer alle verfügbaren Eigenschaften aufgelistet, ohne dass man Eigenschaften explizit entfernen oder hinzufügen kann. Die eigenen Änderungen werden erst durch Klicken des „Save“-Knopfes gespeichert.

Ein weiteres neues Feature ist die Funktion „Bulk Add“, die es erlaubt, viele Metadaten direkt aus einer Excel-Tabelle zu importieren. Im Gegensatz zu Describo verfügt Crate-O jedoch über keine KI-gestützten Funktionen und hat auch keinen Datei-Explorer oder eine Dateivorschau.

3.2.1. Vorteile

Genau wie Describo bietet auch Crate-O alles Nötige um ein Crate vollständig bearbeiten zu können. Vor allem ist aber der Aufbau der Benutzeroberfläche deutlich verbessert, sodass die App im Allgemeinen weniger unübersichtlich wirkt. Durch das ständige Anzeigen eines

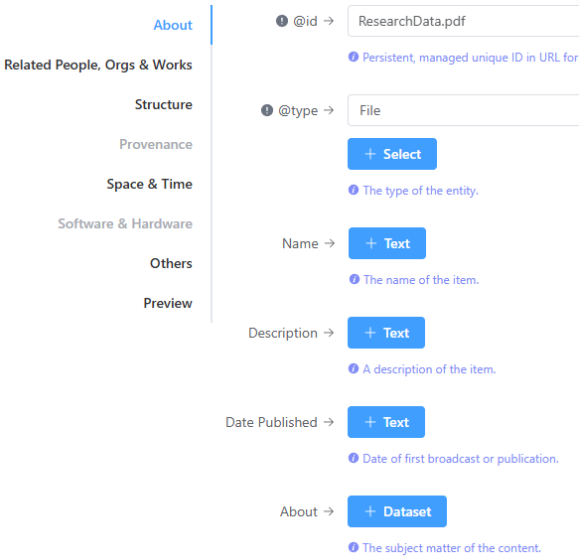

Abbildung 3.5.: Aufteilung der Eigenschaften einer Entity in Crate-O [37]

Entity-Browsers auf der rechten Bildschirmseite sind hier auch keine Probleme bei der Navigation zwischen Entities zu erwarten (vgl. Unterunterabschnitt 3.1.2.1).

3.2.2. Nachteile

Technische Probleme konnten bei Crate-O zunächst nicht identifiziert werden, zumindest sind keine der Probleme von Describo in Crate-O vertreten. Daher sollen hier nur kurz Probleme der Benutzbarkeit aufgezeigt werden, ähnlich zu Unterunterabschnitt 3.1.2.1.

Anders als in Describo zeigt Crate-O für eine Entity alle möglichen Eigenschaften an, soweit kein Profil ausgewählt ist. Eine Suche nach bestimmten Eigenschaften bzw. ein Überblick über Eigenschaften, die einen Wert besitzen, gibt es nicht. Es liegt Nahe, das Crate-O somit ausschließlich für die Verwendung mit bestimmten Profilen gedacht ist. Aber auch hier kann es zu Problemen kommen, wenn ein Profil beispielsweise sehr viele verschiedene Eigenschaften erlaubt.

Entities ohne eingehende Referenzen werden auch hier automatisch gelöscht. Es ist daher denkbar, dass unerfahrene Nutzer*innen recht schnell aus Versehen einen Datenverlust herbeiführen könnten.

Die Gruppierung der Eigenschaften in bestimmte Untergruppen ist zwar sinnvoll, wenn pro Gruppe immer alle Eigenschaften gleichzeitig angezeigt werden sollen, erschwert aber erneut die Suche nach bestimmten Eigenschaften. Beispielsweise kann die Suchfunktion des Browsers nicht herangezogen werden, wenn sich die gesuchte Eigenschaft in einer anderen (nicht ausgewählten) Gruppe befindet (vgl. Abbildung 3.5).

4. Basistechnologien

In diesem Kapitel werden die für die Entwicklung des neuen Editors verwendeten Basistechnologien vorgestellt. Dies sind zum einen Serverseitig die RO-Crate Java Bibliothek [47] und dessen REST-API [36] sowie Clientseitig das Next.js [53] Web-Framework und zugehörige Bibliotheken [19, 56].

4.1. RO-Crate Bibliothek

Die RO-Crate Java Bibliothek „ro-crate-java“ [47] wird seit 2022 am Scientific Computing Center des Karlsruher Instituts für Technologie entwickelt. Diese Bibliothek erlaubt das Erstellen, Bearbeiten und Lesen von Research Object Crates über die Programmiersprache Java. Sie stellt eine Vielzahl an Funktionen bereit, die den korrekten Umgang mit RO-Crates vereinfachen.

Die Bibliothek dient als technische Grundlage für den Web-Editor, der in dieser Arbeit entwickelt wird. Während die Benutzer*in das Crate über den Web-Editor manipuliert, führt die Bibliothek die tatsächlichen Änderungen auf dem Crate aus. Der Funktionsumfang des Web-Editors orientiert sich daher direkt am Funktionsumfang der Java Bibliothek. Die Interaktion zwischen der Java Bibliothek und dem Web-Editor wird durch eine REST-API (siehe Abschnitt 4.2) ermöglicht.

Auf der GitHub Seite [47] der RO-Crate Java Bibliothek ist eine kurze Dokumentation des Funktionsumfangs zu finden. Die Bibliothek unterstützt die aktuelle RO-Crate Spezifikation (v1.1) [51] vollständig.

4.2. REST-API der RO-Crate Bibliothek

Eine REST-API ist ein simpler Web-Server, der die Manipulation von verschiedenen Ressourcen ermöglicht. Dabei gibt es beispielsweise die `GET` Methode, um eine Ressource abzufragen, oder die `POST` Methode, um eine Aktion auf einer Ressource auszuführen.

Dadurch bildet die REST-API eine Brücke zwischen der RO-Crate Java Bibliothek und dem Web-Editor. Ihre Aufgabe ist es dabei, die Anfragen des Web-Editors zu verarbeiten und die entsprechende Funktion bei der Java Bibliothek aufzurufen. So kann der Web-Editor die RO-Crate Bibliothek Fernsteuern.

Die API wird ebenfalls am Scientific Computing Center des Karlsruher Instituts für Technologie entwickelt. Es handelt sich dabei um eine Java Spring Applikation [3], wodurch die RO-Crate Java Bibliothek direkt integriert werden kann.

Um allen technischen Anforderungen des Editors gerecht zu werden, musste die API an einigen Stellen angepasst werden. Auf die nötigen Änderungen und Verbesserungen wird in Abschnitt 5.5 eingegangen.

4.3. Web-Frameworks

Für die Entwicklung des Web-Editors wurde das Web-Framework Next.js [53] zusammen mit der Frontend-Bibliothek React [19] verwendet. Diese werden in diesen Abschnitt kurz vorgestellt, da einige der Begrifflichkeiten und Funktionen in Kapitel 5 erneut aufgegriffen werden.

4.3.1. Next.js

Für den Web-Editor wird das Next.js [53] Framework verwendet. Dieses setzt als Frontend-Bibliothek React [19] ein. Next.js bezeichnet sich als „Das React Framework für das Web“ und liefert viele Features, die React allein nicht zur Verfügung stellt. Dazu gehören viele automatische Optimierungen, zum Beispiel für große Bilddateien. Außerdem unterstützt Next.js „Dynamic HTML Streaming“. Dabei wird auf der aktuellen Seite nur der Code (sei es HTML, CSS oder JavaScript) geladen, der tatsächlich benötigt wird. Wechselt man auf eine andere Seite, so wird der nötige Code nachgeladen. Um dies zu ermöglichen, setzt Next.js von vorneherein auf die Aufteilung der App in mehrere Seiten. Dabei übernimmt das Framework direkt auch das Routing und servieren der zugehörigen Dateien. Klassische React Apps sind im Gegensatz dazu meist als Monolithen, bestehend aus nur einem großen Skript, konzipiert. [53, 31]

In Next.js werden Seiten definiert, indem eine `page.tsx`-Datei im App-Ordner ablegt wird. Durch Verschachtelung von Ordnern können Unterseiten erstellt werden. Der Pfad zu einer Seite ergibt sich dann direkt aus dem Dateisystem-Pfad. Falls sich mehrere Seiten ein gemeinsames Layout teilen, kann dieses in einer `layout.tsx`-Datei definiert werden. Dessen Layout wird automatisch auf alle Unterseiten angewendet. Dies wird beispielsweise für den in Abschnitt 5.3 eingeführten globalen Zustand verwendet. Indem dieser in einer hohen Layout-Datei definiert wird, ist er automatisch auf allen Unterseiten verfügbar. [54]

Verwandte Ressourcen wie das App-Icon oder globale CSS Dateien können ebenfalls direkt in diesem Ordner abgelegt werden. Next.js liefert somit nicht nur die JavaScript-Skripte aus, sondern auch alle weiteren verwendeten Ressourcen, wie Stylesheets und Bilder.

4.3.2. React

Als Frontend-Framework setzt Next.js auf React [19]. React ist 2013 erschienen und wird von Meta entwickelt [19]. Es ist schon länger das beliebteste und am meisten heruntergeladene JavaScript Framework [7, 9, 43]. Aufgrund seiner großen Beliebtheit gibt es unzählige Pakete, die React erweitern und nützliche Funktionalitäten nachliefern. Allein die Suche nach dem Stichwort „react“ liefert auf npmjs.org über 6000 Ergebnisse [27]. Somit ist das React-Ökosystem ein entscheidender Punkt für die Wahl von Next.js und React. [34]

Der Name des Frameworks ist zugleich Programm; React zeichnet sich dadurch aus, dass allein das Ändern des Zustands der App direkt zum Neu-Rendern führt. Mit entsprechenden Optimierungen werden am Ende nur jene Teile neu gerendert, die sich geändert haben.

React setzt auf einen Komponenten-basierten Aufbau. Eine Komponente wird dabei durch eine klassische JavaScript-Funktion definiert. Um als Komponente verwendet werden zu können, muss die Funktion dafür lediglich ein React Element (mittels JSX-Code [21], siehe Listing 4.1) oder `null` zurückgeben. Anschließend kann die Komponente frei in der App verwendet werden. JSX (JavaScript Syntax Extension) erlaubt die Definition von HTML direkt in einer JavaScript Datei, wobei neben den klassischen HTML-Tags auch eigene Komponenten verwendet werden können. Durch Schachtelung verschiedener Komponenten und HTML-Tags ergibt sich schließlich eine App.

```
1   function App(props) {
2       const [mySize, setMySize] = useState(props.defaultSize) // State Hook
3       const myCallback = useCallback(() => setMySize(i => i + 1), []) // Callback Hook
4
5       // Effect will be run when mySize changes
6       useEffect(() => console.log("My size is now " + mySize), [mySize])
7
8       return (
9           <MyComponent className="flex justify-center" size={mySize}>
10              <button onClick={myCallback}>Increase Size</button>
11          </MyComponent>
12      )
13  }
```

Listing 4.1: Definition einer React Komponente mit JSX-Code

Eine Komponente muss dabei nicht zwingend ein Element der Benutzeroberfläche sein. Gibt eine Komponente JSX-Code zurück, so wird diese im Browser als HTML gerendert. Gibt sie `null` zurück, so ist sie zwar kein Teil der UI, kann aber dennoch einen Zustand halten und beispielsweise Netzwerkanfragen durchführen.

Jede Komponente kann „React Hooks“ [17] aufrufen, die vielerlei nützliche Funktionalitäten bieten. Dazu gehört unter anderem die Hook `useState`, die das Speichern einer lokalen Zustandsvariable ermöglicht. Die `useEffect` Hook ermöglicht das Ausführen von Code, sobald sich eine bestimmte Abhängigkeit der Komponente geändert hat. Komponenten

können zusätzlich zu ihrem lokalen Zustand Parameter von außen entgegennehmen. Dadurch kann eine allgemeine Komponente mithilfe ihrer Parameter auf viele verschiedene Einsatzzwecke zugeschnitten werden. In Listing 4.1 ist eine Komponente dargestellt, die einige Hooks verwendet. [19] Somit können Komponenten einzelne intelligente Bausteine einer Benutzeroberfläche implementieren.

Ein wichtiger Nachteil von React ist jedoch die schlechte Performance [28]. Auf die nötigen Performanceoptimierungen wird später in Unterabschnitt 5.4.3 noch ausführlich eingegangen.

4.3.3. Radix UI

Viele interessante Bausteine einer Benutzeroberfläche sind weder in React enthalten, noch in HTML5 standardisiert, wie zum Beispiel Pop-ups oder Tooltips. Während diese sehr hilfreich für die Verbesserung der Benutzbarkeit der Oberfläche sind [11, S. 73][40, S. 279], müssen sie meist mühselig von Hand implementiert werden.

Die React-Bibliothek Radix UI [56] schließt genau diese Lücke. Sie liefert einige hilfreiche Komponenten, welche besagte Bausteine implementieren. Ein Beispiel für eine Pop-up Komponente ist in Listing 4.2 dargestellt. Der dargestellte Code erzeugt einen Button der durch Klicken ein Pop-up öffnet, welches über dem Button schwebt. Ähnlich simpel ist auch die Integration komplexer Bausteine wie beispielsweise eines verschachtelten Dropdown-Menüs. Dabei ist jedoch kein Styling inkludiert, sodass das äußerliche Design der Komponenten perfekt auf die eigene App zugeschnitten werden kann.

```
<Popover.Root>
    <Popover.Trigger>
        Click to Open
    </Popover.Trigger>
    <Popover.Portal>
        <Popover.Content>
            // Content of the pop-up
        </Popover.Content>
    </Popover.Portal>
</Popover.Root>
```

Listing 4.2: Beispielhafte Pop-up Komponente aus Radix UI

Auf Radix UI aufbauend sind einige Komponenten-Sammlungen verfügbar, die ein konsistentes äußerliches Design beisteuern. In Abbildung 4.1 ist das Design der „shadcn/ui“-Sammlung, welche für den Web-Editor verwendet wurde, anhand eines beispielhaften Mail-Programmes zu sehen. Im Gegensatz zu einer klassischen Bibliothek wird shadcn/ui nicht durch einen Paketmanager importiert, sondern alle benötigten Komponenten müssen manuell in den Quellcode des Projektes kopiert werden. So kann das Design ohne weiteren Aufwand personalisiert und die Funktionalität angepasst werden.

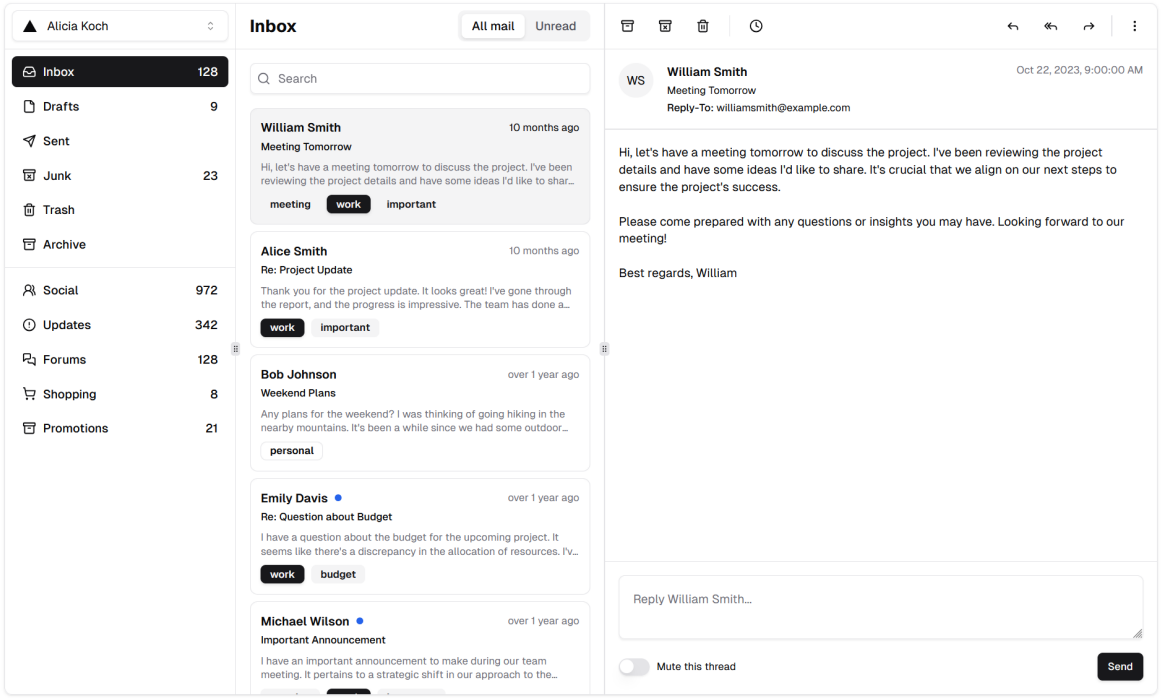

Abbildung 4.1.: Die Komponenten-Sammlung shadcn/ui [39] am Beispiel eines Mail-Programmes. Quelle: [39]

5. Web-Editor NovaCrate

Im Rahmen dieser Arbeit wurde der RO-Crate Web-Editor „NovaCrate“ entwickelt. Der Name setzt sich aus dem lateinischen Wort „Nova“ (dt. „Neues“) und „Crate“ zusammen. Dieses Kapitel befasst sich mit dem Design und der Implementierung von NovaCrate. Der Editor baut dabei auf die RO-Crate Java-Bibliothek [47], sowie dessen REST-API [36] (siehe Kapitel 4), auf.

Da die Entwicklung des Web-Editors ein zentrales Ziel dieser Arbeit ist, sollen in den folgenden Abschnitten wichtige Designentscheidungen begründet und interessante Komponenten der Implementierung vorgestellt werden. Zunächst werden die funktionalen und nichtfunktionalen Anforderungen des Editors definiert.

5.1. Anforderungen

Die Anforderungen an den Editor ergeben sich zum einen direkt aus dem Funktionsumfang der Java Bibliothek [47], welcher sich wiederum an der Spezifikation für RO-Crates (siehe Kapitel 2) orientiert, als auch aus den beobachteten Nachteilen der bestehenden Editoren (siehe Kapitel 3).

5.1.1. Funktionale Anforderungen

/F1/ Erstellen von neuen Crates

/F2/ Importieren von existierenden Crates im Zip-Format

/F3/ Export des entstandenen Crates im Zip-Format

/F4/ Löschen eines Crates aus dem lokalen Speicher

/F5/ Hinzufügen, Löschen und Ändern der Entities eines Crates

/F6/ Generieren von Data Entities aus lokalen Dateien und Ordnern

/F7/ Erstellen von Data Entities für Web- oder Remote-Ressourcen

/F8/ Importieren von Contextual Entities aus ORCID [30] und ROR [4]

/F9/ Hinzufügen, Löschen und Ändern der Eigenschaften einer Entity

/F10/ Anzeigen und Manipulieren des Crates durch einen interaktiven Graphen

/F11/ Dateivorschau für im Crate enthaltene Data Entities

/F12/ Direktes Bearbeiten der Metadaten durch einen integrierten JSON-Editor

5.1.2. Nichtfunktionale Anforderungen

/N1/ Einfachheit: Der Editor muss einfach zu erlernen sein

/N2/ Intuitivität: Der Editor muss intuitiv in der Verwendung sein

/N3/ Unterstützung: Der Editor muss den Nutzer bei der Bearbeitung des Crates unterstützen

/N4/ Erweiterbarkeit: Der Editor muss um neue Funktionen erweiterbar sein

/N5/ Wartbarkeit: Der Editor muss einfach zu warten und zu aktualisieren sein

/N6/ Portabilität: Der Editor muss alle gängigen Desktop-Browser unterstützen

5.2. Design

Das Design bestimmt in großen Maße die Qualität des Editors sowie dessen Akzeptanz durch die Benutzenden [46, S. 225]. Es spielt vor allem auch für die letztendliche Benutzbarkeit der Software eine große Rolle. Als maßgebende Werke für das Design der Benutzeroberfläche dienen die Bücher von Krug [11], Tidwell [46] und Shneiderman [40] et al. Während des Designs wurden die Anforderungen aus Abschnitt 5.1 eng einbezogen.

Das optische Design liefert die in Unterabschnitt 4.3.3 vorgestellte shadcn/ui [39] Komponentensammlung, basierend auf den Radix UI [56] Bausteinen. So kann die Entwicklung auf den RO-Crate Editor konzentriert werden, ohne selbst ein konsistentes Design oder komplexe interaktive Komponenten entwickeln zu müssen.

5.2.1. Startseite des Editors

Beim Start von NovaCrate öffnet sich zunächst eine Einstiegsseite (siehe Abbildung 5.1). Diese listet alle importierten Crates auf, und ermöglicht das Erstellen oder Importieren neuer Crates.

Auf der rechten Seite, die den Großteil des Bildschirms einnimmt, soll der Einstieg in den Editor vereinfacht werden. Große Schaltflächen erlauben das Erstellen eines neuen Crates. Eine Auflistung der in der Vergangenheit bearbeiteten Crates soll einen zügigen Wiedereinstieg ermöglichen. Auf der linken Seite sind erneut alle Funktionen der Startseite aufgelistet. So wird neuen Benutzern eine Hilfestellung zum Einstieg in das Programm gegeben und erfahrene Nutzer haben direkten Zugang zu allen Funktion. [46, S. 146]

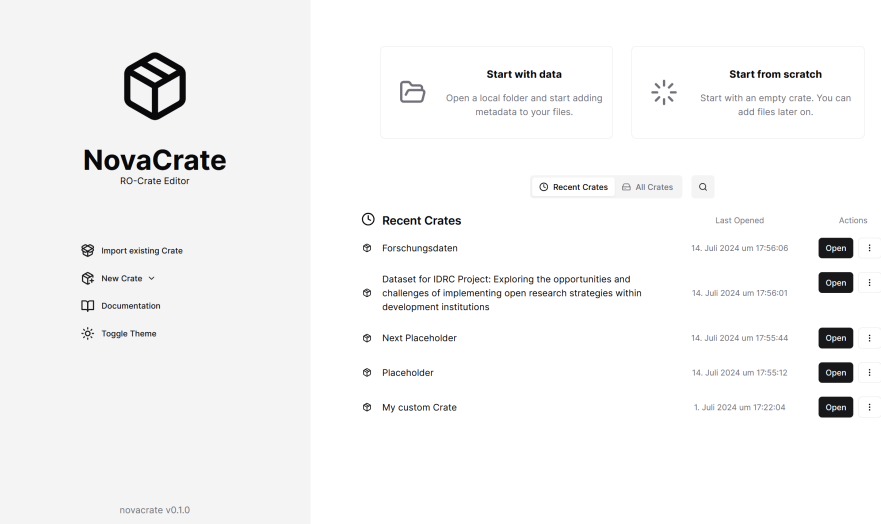

Abbildung 5.1.: Die Einstiegsseite von NovaCrate

Sobald hier ein Crate zum Bearbeiten gewählt wird, öffnet sich der Editor. Der Übergang wird mit einer Animation dargestellt, um den Benutzer nicht mit der plötzlichen Inhaltsänderung zu desorientieren [46, S. 202]. Im Editor kann immer genau ein Crate bearbeitet werden.

5.2.2. Globale Designelemente

Der grundlegende Aufbau des Editors besteht aus einer globalen Navigation auf der linken Seite sowie einem Funktionsmenü am oberen Bildschirmrand (siehe Abbildung 5.2). Dies ist eine klassische „inverted L navigation“ [46, S. 132] wie sie bei vielen Websites, als auch Code-Editoren, zu finden ist. Dies ermöglicht den Nutzer*innen zu jedem Zeitpunkt eine der fünf Hauptseiten aufzurufen und erleichtert die Navigation erheblich [11, S. 92f]. Dabei kann beliebig zwischen den Seiten gewechselt werden, ohne dass Ladezeiten entstehen oder Änderungen verloren gehen. So können neue Nutzer*innen sich schnell zurechtfinden und den Editor frei erkunden [46, S. 134]. Diese fünf Hauptseiten, abseits der Einstiegsseite, gliedern zusammengehörige Funktionalitäten an einem zentralen Punkt:

- **Entities**: Hier kann jede Entity im aktuellen Crate bearbeitet werden (siehe Abbildung 5.3). Auf der linken Seite ist ein Entity Browser angesiedelt, der alle Entities im Crate anzeigt. In der Mitte werden alle geöffneten Entities als Tabs angezeigt. Auf der rechten Seite lässt sich optional eine Dateivorschau zu einer Data Entity anzeigen.
- **File Explorer**: Hier können die Dateien im Crate inspiziert werden (siehe Abbildung 5.4). Links werden alle Dateien und Ordner aufgelistet, die im Crate enthalten sind. Die Dateien lassen sich direkt in einer Vorschau auf der rechten Seite ansehen. Hier werden auch Dateien angezeigt, für die noch keine Data Entity erstellt wurde. Außerdem können hier besonders einfach neue Dateien und Ordner hochgeladen werden.

- **Graph**: Der Graph stellt alle Entities im Crate als Knoten in einem Graphen dar (siehe Abbildung 5.5). Die Verbindungen zwischen den Knoten entsprechen den Referenzen zwischen den Entitäten und sind frei bearbeitbar. Zusätzlich können hier auch neue Eigenschaften oder neue Entities erstellt werden.
- **JSON Editor**: Diese Seite erlaubt das direkte Bearbeiten der zugrundeliegenden Metadaten-Datei über einen integrierten JSON Editor (siehe Abbildung 5.6). So können Einzelheiten in der Metadaten-Datei frei geändert werden. Da durch diese Funktion schnell Fehler in die Metadaten-Datei eingebaut werden können, ist diese Seite nur für Experten empfohlen.
- **Context**: Hier kann der Kontext des Crates angepasst und erweitert werden (siehe Kapitel 2 und Abbildung 5.7). Es wird die aktuelle Spezifikation angezeigt und es können eigene Schlüssel-Wert-Paare zum Kontext hinzugefügt werden.

Über das globale Funktionsmenü am oberen Bildschirmrand (siehe Abbildung 5.2) können auf jeder der Hauptseiten viele grundlegende Funktionalitäten des Editors ausgeführt werden. Sollte auf der aktuellen Seite eine bestimmte Entity gewählt sein, so wird auch ein Menü für die aktuelle Entity angezeigt. Die Nutzenden haben also auf jeder Seite Zugriff auf dieselbe Grundfunktionalität. Auch abseits des globalen Funktionsmenüs bieten viele der Hauptseiten sich überschneidende Funktionalitäten. So verbringen die Nutzenden weniger Zeit mit der Suche nach bestimmten Schaltflächen, sondern können alle anwendbaren Funktionen direkt verwenden. [46, S. 135]

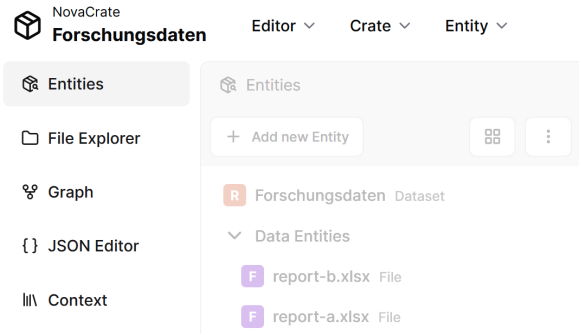

Abbildung 5.2.: Die globale Navigation sowie das Funktionsmenü in NovaCrate

Außerdem enthält jede Seite am rechten Bildschirmrand ein Suchfeld. Darüber kann nicht nur jede der Hauptseiten des Editors geöffnet und jede der grundlegenden Funktionen ausgeführt werden, sondern es kann auch nach allen Entities gesucht werden. So können Nutzer zu jeden Zeitpunkt direkt nach einer Funktionalität suchen, falls sie diese nicht direkt finden. [11, S. 98]

5.2.3. Entity Editor

Der Entity Editor bildet das Herzstück von NovaCrate. Dieser ist in die Entities-Seite eingebettet und bearbeitet je eine bestimmte Entity des Crates. Da die Benutzenden wohl die meiste Zeit auf dieser Seite verbringen werden, fand hier die meiste Designarbeit statt.

Die Entities-Seite hat ihre eigene Navigation zum Wählen der zu bearbeitenden Entity. Hierfür wird eine Auflistung aller Entities an der linken Seite angezeigt. Zwischen den zuletzt bearbeiteten Entities kann einfach über eine klassische Tab-Navigation gewechselt werden, die für viele Nutzende vertraut und einfach zu verwenden ist [11, S. 106f]. So kann schnell zwischen mehreren Entities hin-und-her gesprungen werden, ohne dass die Entities aufs neue herausgesucht werden müssen. Ein Beispielbild ist in Abbildung 5.3 zu sehen.

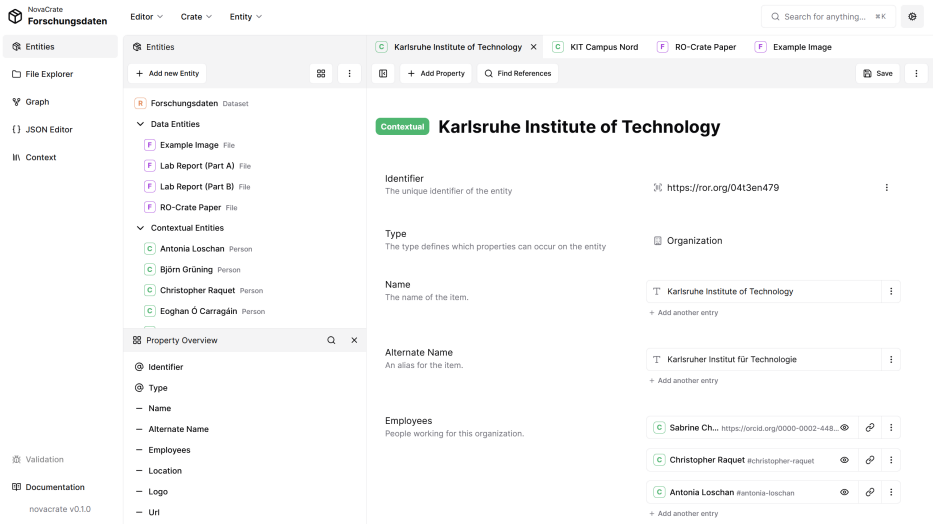

Abbildung 5.3.: Die Entity Editor-Seite in NovaCrate

Dank der Tabs kann frei zwischen den Entities gewechselt werden, ohne dass gespeichert werden muss. Falls eine Daten-Entity bearbeitet wird, lässt sich für einige gängige Dateitypen (wie zum Beispiel PDF) eine Vorschau anzeigen. Der Entity Editor selbst besteht hauptsächlich aus einer Auflistung aller Eigenschaften der Entity. Links wird der Name sowie eine Beschreibung der Eigenschaft angezeigt und rechts dessen Werte. Hier können auch direkt die Werte geändert oder neue hinzugefügt werden. Je nach Art der Eigenschaft läuft das editieren anders ab. Während Texteinträge einfach per Tastatur geschrieben werden, können Referenzen über einen Dialog geknüpft oder ausgetauscht werden. Um die Bearbeitung von Datums-, Uhrzeit-, Nummernfeldern oder Booleschen Feldern zu erleichtern, werden entsprechende Eingabefelder angezeigt.

5.2.4. File Explorer

Die File Explorer-Seite ist beispielhaft in Abbildung 5.4 dargestellt. Das Design dieser Seite ist recht simpel. Auf der linken Seite wird in einer klassischen Ordnerstruktur der Dateiinhalt

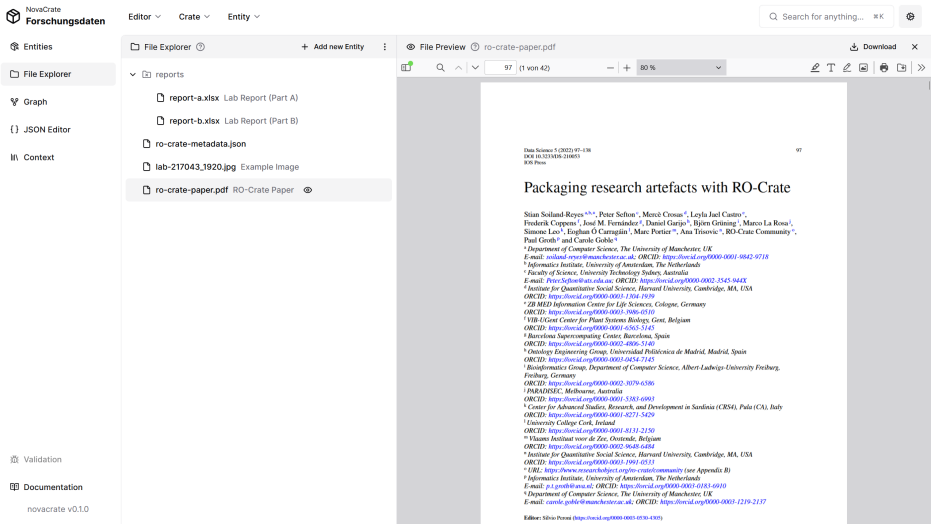

Abbildung 5.4.: Die File Explorer-Seite in NovaCrate

des Crates angezeigt. Auf der rechten Seite wird eine Vorschau für die aktuelle Datei angezeigt, falls vorhanden. Über einen Rechtsklick auf eine Datei kann zu der zugehörigen Entity auf der Entities-Seite navigiert werden. Falls noch keine zugehörige Entity existiert, kann diese direkt erstellt werden.

5.2.5. Graph

Die Graph-Seite bietet einen interaktiven Graphen, in dem alle Entities des Crates dargestellt sind. Ein Beispiel ist in Abbildung 5.5 zu sehen. Die Referenzen zwischen den Entities bilden letztendlich die Kanten, welche die Entities miteinander verbinden. Mit der Maus können ganz einfach neue Kanten gezogen werden, um Entities miteinander zu verknüpfen. Das globale Funktionsmenü erlaubt auch hier das Erstellen neuer Entities. Die Implementierung des Graphen wird in Unterabschnitt 5.4.1 erläutert.

5.2.6. JSON Editor

Auf der JSON Editor Seite ist, über die ganze Seite erstreckt, ein Texteditor integriert. Dies ist in Abbildung 5.6 dargestellt. Dieser erlaubt das Bearbeiten der ro-crate-metadata.json Datei, so wie es andere Texteditoren auch tun würden. Der Funktionsumfang ist hier jedoch stark begrenzt und besteht hauptsächlich aus einer Speichern-Funktion, und natürlich dem Ändern des Inhaltes. Das Design und die Funktionalität wird durch die Monaco [24] Bibliothek geliefert. Lediglich das Farbschema für den dunklen Modus wurde an das von NovaCrate angepasst.

Hierbei handelt es sich um eine Expertenfunktion, wie die Warnung am oberen rechten Bildschirmrand anzeigt. Durch das direkte Bearbeiten der Datei kann jedes Detail des

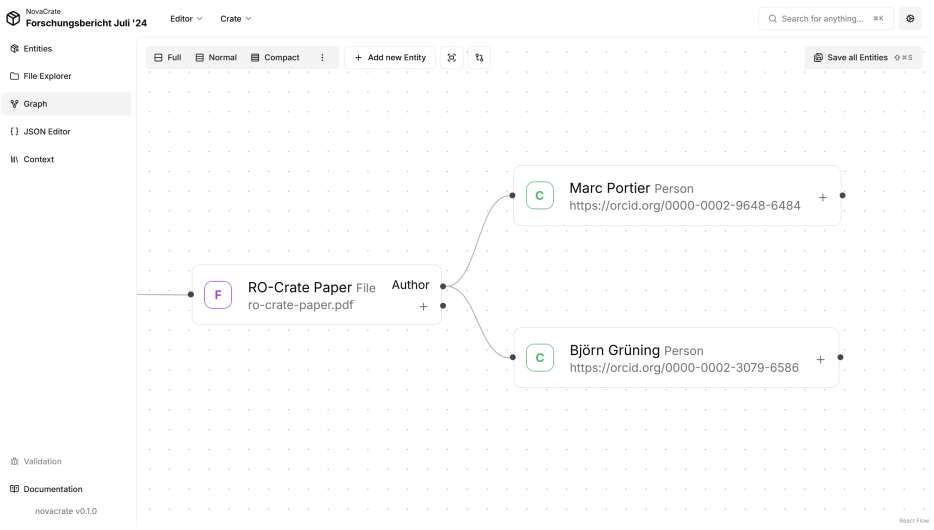

Abbildung 5.5.: Die Graph-Seite in NovaCrate

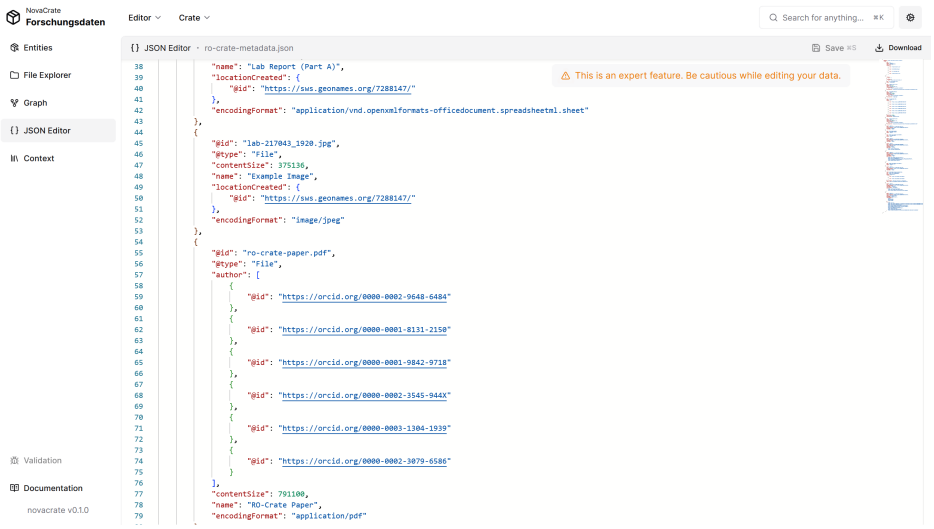

Abbildung 5.6.: Der JSON Editor in NovaCrate

Crates manipuliert werden. Ein unachtsamer Gebrauch würde zum Defekt des Crates führen. Syntaxfehler werden jedoch von Monaco erkannt und verhindern das Speichern der Datei.

Darüber hinaus kann die ro-crate-metadata.json Datei auf dieser Seite direkt heruntergeladen werden. Die gleiche Funktion findet sich aber auch im globalen Funktionsmenü unter dem Punkt „Crate“.

5.2.7. Context

Die Context Seite folgt einem simplen Aufbau, der in Abbildung 5.7 zu sehen ist. Sie ermöglicht es den Nutzenden den Context des Crates zu erweitern. Im aktuellen Entwicklungsstand wird hier lediglich die aktuell verwendete Spezifikation angezeigt. Zusätzlich können neue Schlüssel-Wert-Paare in den Kontext eingefügt werden. Für tiefergehende Anpassungen muss jedoch der JSON Editor verwendet werden.

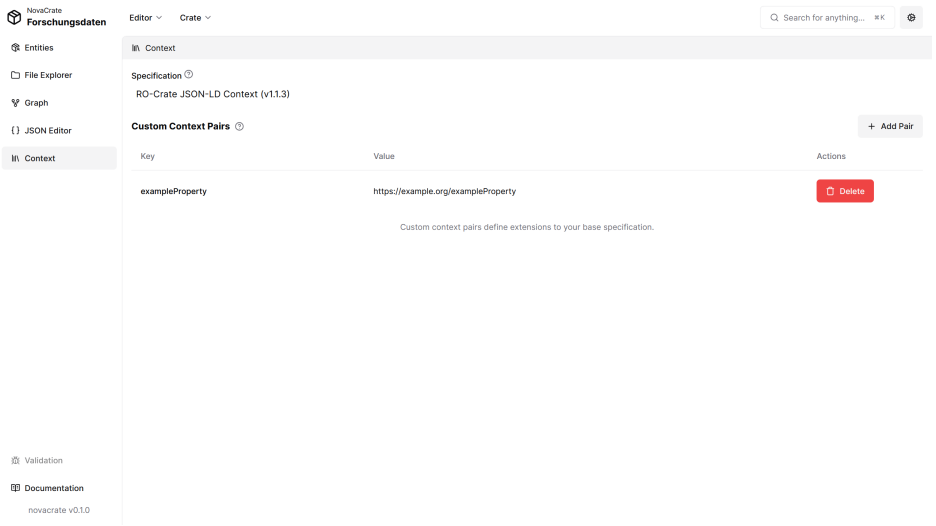

Abbildung 5.7.: Die Context-Seite in NovaCrate

5.3. Technische Architektur

In diesem Abschnitt wird die technische Architektur des Editors beleuchtet und einige interessante Komponenten genauer betrachtet. Besonders relevant für die Erweiterbarkeit und Wartbarkeit ist eine sorgfältige Planung der technischen Architektur und ihrer wichtigsten Komponenten. In diesem Abschnitt soll die gute Wartbarkeit, Portierbarkeit[1] und Erweiterbarkeit des Editors begründet werden, wie in Abschnitt 5.1 gefordert.

Die technische Architektur des Editors folgt dem bewährten Model-View-Controller [5] Modell. In React wird dies typischerweise über einen globalen Zustand realisiert, der über Aktionen verändert wird. Dieses Design wird auch für NovaCrate verwendet. Der große Vorteil eines solchen Aufbaus ist, dass das Model für die gesamte App zur Verfügung steht. Wenn mehrere Komponenten die gleichen Daten anzeigen, müssen Sie diese nicht jeweils einzeln Abrufen, sondern lesen aus dem globalen Zustand. In NovaCrate ist der globale Zustand in mehrere „Zustandsbehälter“ aufgeteilt, die größtenteils unabhängig voneinander arbeiten.

[1] Da NovaCrate eine Web-App ist, ist sie auch automatisch gut portierbar. Es muss lediglich sichergestellt werden, dass NovaCrate keine Features verwendet, die nur in spezifischen Browsern verfügbar sind.

Diese sind dann entweder, je nach Anforderungen des Zustandsbehälters, in Form von einem React Context oder einem Zustand Store [33] realisiert. React Context ist dabei ein in React eingebauter Mechanismus, der einer Komponente ermöglicht, ihre Methoden allen darunter liegenden Komponenten zugänglich zu machen. Die Referenz auf eine Methode muss somit nicht durch den gesamten Komponenten-Baum geschleift werden, sondern kann „aus der Luft“ gegriffen werden, sofern die bereitstellende Komponente weiter oben im Komponenten-Baum sitzt. Problematisch bei React Context ist jedoch die Performance, siehe dazu Unterabschnitt 5.4.3.

Die Zustand [33] Bibliothek bietet hingegen simplere Zustandsbehälter, die nicht durch eine React Komponente bereitgestellt werden. Ihre Verwendung ist dabei so simpel wie die von React Contexts. Da der Zustandsbehälter hier aber nicht in die Komponenten-Hierarchie integriert ist, führt eine Änderung des Zustandes in einem Zustand Store nur in den tatsächlich betroffenen Komponenten zum Neu-Rendern. Ein Zustand Store ist also keine React Komponente und kann entsprechend auch keine React Hooks verwenden. Im Folgenden werden einige Beispiele gezeigt, die den Unterschied zwischen den beiden Herangehensweisen für Zustandsbehälter verdeutlichen.

Eine Auflistung der globalen Stores und Context-Komponenten ist in Tabelle 5.1 zu sehen. Hier wird auch jeweils die Art des Zustandsbehälters kurz begründet. Die beiden wichtigsten sind der „Crate Data Context“ (React Context) und der „Editor State Store“ (Zustand Store). Der Crate Data Context führt die Kommunikation mit der REST-API durch, was in einem Zustand Store technisch nicht möglich wäre. Der Editor State Store hält den grundlegenden Zustand des Editors, also eine Arbeitskopie des aktuellen Crates. Jede Komponente im Editor kann aus diesem Store die aktuellen Änderungen abfragen oder neue Änderungen hinzufügen. Der Store stellt dafür direkt maßgeschneiderte Aktionen zur Verfügung, beispielsweise `setPropertyValue()` um den Wert einer Eigenschaft zu ändern.

Der Editor State Store, so wie viele der anderen Stores und Contexts auch, ist zugleich Model und Controller. Er hält nicht nur einen Zustand (Arbeitskopie des Crates), sondern stellt auch komplexere Methoden zur Änderung des Zustandes bereit. Dies hat den großen Vorteil, dass alle View-Komponenten über simple Methodenaufrufe das Crate vollumfänglich manipulieren können.

In den folgenden Unterabschnitten werden einige weitere Konzepte im Detail vorgestellt. Da diese zentral für die Funktionalität des Editors sind, werden sie als Teil der technischen Architektur angesehen.

5.3.1. Aufteilung in Arbeits- und Serverzustand

Eine der zentralen Herausforderungen dieser Arbeit ist das Bearbeiten eines RO-Crates über die REST-API (siehe Abschnitt 4.2 und [36]). Diese REST-API nutzt wiederum die RO-Crate Bibliothek [47]. Dabei handelt es sich um eine Java Bibliothek zum Erstellen oder Lesen von RO-Crates. Funktionalitäten zum Editieren eines existierenden Crates sind jedoch eher spärlich vorhanden. Diese Lücke schließt zwar zum Großteil bereits die REST API, jedoch

Name des Zustandes	Art des Zustandes	Begründung
Crate Data	React Context	Lädt selbstständig Crate Daten vom Server
Editor State	Zustand Store	Zustand aller Entities, wird sehr häufig aktualisiert. Als React Context zu schlechte Performance.
Crate Verify	React Context	Nutzt einen Web-Worker um Verifikation in einem zusätzlichen Thread durchzuführen
Entity Editor Tabs	Zustand Store	Siehe Unterabschnitt 5.4.3
File Explorer State	Zustand Store	Einfacher Zustand
Global Modal	React Context	Rendert die Modal-Komponenten
Graph State	Zustand Store	Einfacher Zustand
Graph Settings	Zustand Store	Wird persistent gespeichert
Actions	React Context	Führt Aktionen in Form von React Hooks aus
Entity Browser Settings	Zustand Store	Wird persistent gespeichert

Tabelle 5.1.: Globale Stores und Context-Komponenten in NovaCrate

verbleibt ein zentrales Problem: Wann sollen die Änderungen, die der Nutzer im Editor vornimmt, an die REST API geschickt und somit gespeichert werden?

Der einfachste Ansatz wäre hier, so wie es die bestehenden Editoren tun (siehe Kapitel 3), alle Änderungen so schnell wie möglich zu speichern, sobald der Nutzer ein Eingabefeld verlässt. Über diesen Ansatz ergeben sich jedoch Probleme bei der Benutzerfreundlichkeit. So können die Nutzenden am Ende nicht mehr nachvollziehen, welche Änderungen getätigt wurden. Die Änderungen können außerdem nicht rückgängig gemacht oder vor dem Speichern geprüft werden. Daher wurde für NovaCrate ein anderer, benutzerfreundlicher Ansatz gewählt.

NovaCrate hält eine vollständige Kopie des Crates als Arbeitszustand. Dieser wird kontinuierlich verändert, indem die Nutzenden im Editor Änderungen vornehmen. Zusätzlich kennt NovaCrate den Serverzustand, also das Crate ohne die aktuellen Änderungen. Da jede Entity einen eindeutigen Bezeichner führt, kann einfach bestimmt werden, welche Entitäten geändert, gelöscht oder hinzugefügt wurden. Selbiges gilt für die Eigenschaften besagter Entities.

Der Vorteil dieses Ansatzes ist, dass die Nutzenden alle ihre Änderungen prüfen können, bevor Sie diese speichern. Dabei ist der Arbeitszustand über alle Editor-Seiten synchronisiert. Nimmt man also Änderungen im Entity Editor vor, so kann man die Auswirkungen direkt im Graphen beobachten.

Das Problem dieses Ansatzes ist nun, dass die RO-Crate Bibliothek das Manipulieren des Crates übernimmt. Beispielsweise kümmert sie sich um die Pflege der `hasPart` Eigenschaft der Wurzel-Entität. Legt eine Benutzer*in eine neue Entität an, so muss dessen Bezeichner in die `hasPart`-Liste aufgenommen werden. Folgendes Szenario verdeutlicht das Problem: Die Benutzer*in ändert den Namen des Crates in der Wurzel-Entität. Anschließend erstellt sie eine neue Entität und speichert diese. Die RO-Crate Java Bibliothek fügt die neue Entität hinzu und passt die `hasPart` Eigenschaft der Wurzel-Entität an, sodass das Crate der Spezifikation entspricht. NovaCrate erhält den neuen Serverzustand und ersetzt den Arbeitszustand durch diesen. In diesem Szenario würden die Änderung des Crate-Namens verloren gehen, da die Benutzer*in die Änderungen an der Wurzel-Entität noch nicht gespeichert hat.

Um dieses Problem sinnvoll zu lösen, muss NovaCrate den Server- und Arbeitszustand intelligenter zusammenführen. Der Ablauf dieses Vorgangs ist in Abbildung 5.8 dargestellt. Letztendlich sorgt dieser Algorithmus dafür, dass nur jene Eigenschaften im Arbeitszustand durch den Serverzustand überschrieben werden, welche durch die RO-Crate Bibliothek bearbeitet wurden.

5.3.2. Editor Aktionen

Viele Aktionen, die der Nutzer im Editor ausführen kann, sind an mehreren Stellen dupliziert. So kann eine Entity sowohl im Entity-Explorer als auch im Graph gelöscht werden. Auch das Speichern aller geänderter Entities ist auf jeder Seite möglich. Zusätzlich sind viele dieser Aktionen mit spezifischen Icons und Tastaturkürzeln versehen. Um den Editor trotz der vielfachen Duplizierung dieser Aktionen simpel und konsistent zu halten, enthält NovaCrate ein eigenes Aktionssystem. Dieses basiert auf einem React Context (vgl. Tabelle 5.1) und ist global verfügbar.

Über die eigene React Hook `useAction` kann eine Aktion über ihren Namen abgerufen werden. Anschließend können direkt Icon, Tastaturkürzel sowie der Titel der Aktion ausgelesen und konsistent dargestellt werden. Natürlich kann die Aktion auf diesem Wege auch direkt ausgeführt werden. Aktionen können beliebig von Komponenten mithilfe einer weiteren React Hook während der Laufzeit hinzugefügt und entfernt werden, sodass auch dynamische Aktionen möglich sind. Da die Aktionen in Komponenten definiert werden, die hierarchisch unter den globalen Zuständen und Contexts liegen, können die Aktionen frei auf den globalen Zustand zugreifen und diesen manipulieren. Ein gutes Beispiel ist das in Abschnitt 5.2 bzw. Abbildung 5.2 gezeigte globale Aktionsmenü, welches aus Aktionen aus dem Action Context besteht.

Mit dem Aktionssystem kann der Großteil der Aktionen des Editors abgebildet werden. So können problemlos mehrere Schaltflächen oder Menüoptionen für eine gemeinsame Aktion existieren, denn die Aktion ist zentral definiert. Auch die Designkonsistenz wird garantiert, da das Icon ebenfalls an diesem zentralen Punkt definiert wird. Das Aktionssystem stellt also eine hohe Wartbarkeit und vor allem auch einfache und schnelle Erweiterbarkeit sicher.

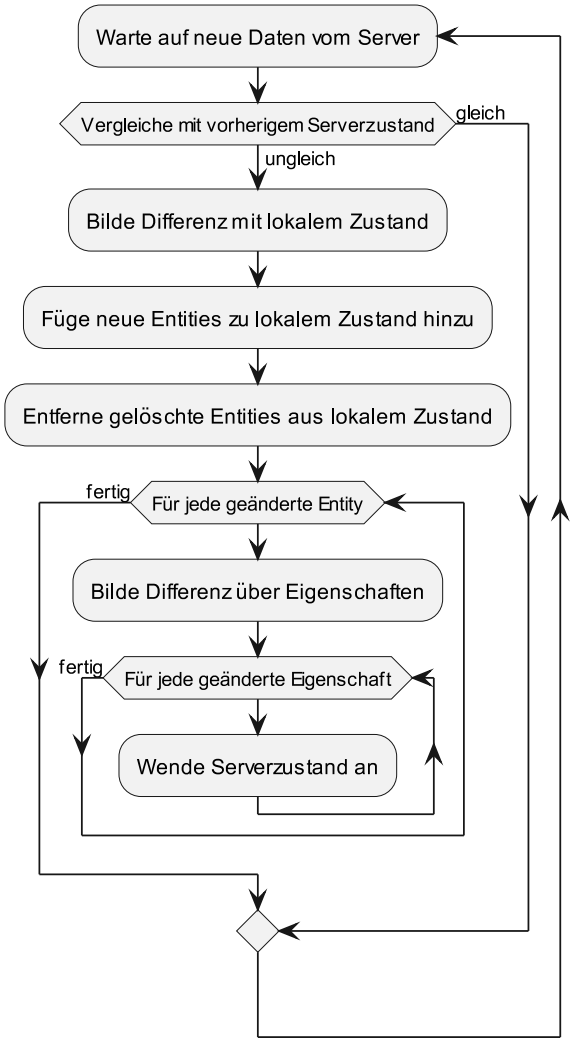

Abbildung 5.8.: Ablauf der Klärung von Abweichungen zwischen Server- und Arbeitszustand

5.3.3. Schema Web-Worker

Der Schema Web-Worker ist zentral für den Editor, da dieser dynamische Abfragen gegenüber dem RO-Crate Schema bzw. Crate Context (siehe Kapitel 2) ermöglicht. Im Entity Editor wird für jede Eigenschaft eine Beschreibung sowie ein zum Typ der Eigenschaft passendes Eingabefeld angezeigt. Diese sind in NovaCrate nicht fest definiert, sondern werden während der Laufzeit direkt aus dem RO-Crate Schema gelesen. Eine weitere Aufgabe des Schema Web-Workers ist das Filtern der Entities im Crate nach dem Typ. Sucht man im Crate beispielsweise nach einer Autor-Entität, so werden auch alle Autor-Unterklassen-Entitäten zurückgeliefert.

Wie der Name bereits vermuten lässt, werden diese Aufgaben in einem Web-Worker ausgeführt. Web-Worker sind ein JavaScript Standardfeature und erlauben das Ausführen von einem JavaScript-Skript abseits des Hauptfadens [25]. Sie sind verfügbar in allen gängigen

Browsern und schränken die Portierbarkeit von NovaCrate nicht ein. Somit können kostspielige Berechnungen im Hintergrund durchgeführt werden, ohne dass React im Hauptfaden beeinträchtigt wird. Ursprünglich wurde der Schema-Algorithmus ohne Web-Worker ausgeführt, hatte jedoch bei jeder Anfrage die App für etwa eine Sekunde blockiert. Durch den Web-Worker kann das Interface während dieser Zeit eine Ladeanimation anzeigen und bleibt responsiv. Auch Aufgaben auf die nicht zwingend gewartet werden muss, wie das Laden von Eigenschaftskommentaren, können im Hintergrund ausgeführt werden. Solange die Kommentare laden, kann die Nutzer*in den Editor weiterhin vollumfänglich nutzen, und merkt nichts von dem Rechenaufwand.

Der ursprüngliche Prototyp des Schema Web-Workers war sehr schwer zu warten und zu erweitern. Die Fallstricke die Web-Worker im Allgemeinen mit sich bringen, und wie sie in NovaCrate gelöst wurden, werden in Unterabschnitt 5.4.2 beschrieben.

5.4. Implementierung

NovaCrate ist in der Programmiersprache TypeScript [23] implementiert. TypeScript ist Open Source und wird von Microsoft entwickelt. Die Sprache erweitert JavaScript um ein dynamisches Typisierungssystem. TypeScript wird vor der Ausführung in klassisches JavaScript kompiliert, das in allen gängigen Browsern läuft. Die Prüfung der Typisierung findet dabei nur während des Kompilierens, und nicht während der Laufzeit statt. Damit bietet TypeScript viele Vorteile während der Entwicklung (wie zum Beispiel Typsicherheit) und keine Nachteile während der Laufzeit. [23]

In den folgenden Unterabschnitten soll ein tieferer Einblick in einige Implementierungsdetails des Editors gegeben werden. Zu guter Letzt wird den nötigen Performanceoptimierungen noch ein Unterabschnitt gewidmet.

5.4.1. RO-Crate Graph

Der RO-Crate Graph in NovaCrate ist ein innovativer Ansatz zur Darstellung der Beziehungen in einem Crate als Graph. Jede Entity ist dabei als Knoten vertreten. Referenzen von einer Entität A auf eine Entität B werden mit einer Kante angezeigt, die auf der rechten Seite von A ausgeht und in der linken Seite von B endet.

Es gibt bereits eine React-Bibliothek, genannt „React-Flow“ [57], die interaktive Graphen dieser Art ermöglicht. Hier muss lediglich die Knotenmenge sowie die Kantenmenge definiert werden, um einen simplen Graphen in React einzubinden. Außerdem müssen entsprechende Callbacks definiert werden, welche die Interaktivität des Graphen ermöglichen. Wenn also ein Knoten verschoben wird, wird für jede Koordinatenänderung eine Funktion aufgerufen, welche die Koordinaten des Knotens im internen Zustand anpasst.

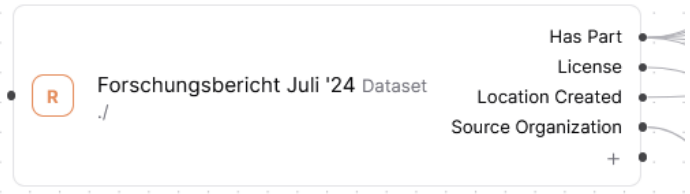

Abbildung 5.9.: Ein Knoten im RO-Crate Graph von NovaCrate

Ein großer Vorteil von React-Flow ist, dass jeder Knoten eine eigene React Komponente ist. Dadurch kann ein Knoten ganze Bedienelemente enthalten und Informationen beliebig anzeigen. In NovaCrate wurde dafür eine Komponente zum Anzeigen von Entities (siehe Abbildung 5.9), sowie eine Komponente zum Anzeigen von Externen URLs implementiert.

Die Hauptschwierigkeiten bei der Darstellung von Graphen im Allgemeinem ist das Routing der Kanten. Überkreuzen sich die Kanten zu oft, fällt es sehr schwer die Zusammenhänge

nachzuvollziehen. Zu diesem Zweck setzt NovaCrate auf die Routing-Bibliothek „Dagre“ [32], welche die Knoten so verteilt, dass möglichst wenige Kanten sich überschneiden.

Letztendlich war es jedoch recht schwierig den internen Zustand von NovaCrate und von React-Flow zusammen mit dem asynchronen Routing von Dagre in Einklang zu bringen. Während in React normalerweise eine Änderung des Zustandes auch zum neu Ausführen aller abhängigen Callbacks führt, würde dies in diesem Fall nicht den erwünschten Effekt haben. So würde Dagre jedes Mal die Knoten neu anordnen, wenn der Nutzer gerade versucht, einen Knoten zu verschieben. Auch wenn ein neuer Knoten hinzugefügt wurde, sollte sich das Layout der bestehenden Knoten nicht verändern, um den Nutzer nicht zu desorientieren.

Letztendlich wurde hier eine simple und doch effektive Faustregel angewendet: Das Routing wird nur ausgeführt, wenn alle Knoten sich an der Position $(0, 0)$ befinden. Dies ist genau dann der Fall, wenn der Graph zum ersten Mal initialisiert wird, also wenn die Graph-Seite zum ersten Mal geöffnet wurde. Alternativ können die Nutzer*innen das Routing jederzeit per Knopfdruck durchführen.

5.4.2. Schema Web-Worker

Der Editor muss häufig relativ rechenintensive Operationen auf dem Crate Schema durchführen. Aber auch bereits wenig intensive Vorgange wie das Nachschlagen von Kommentaren für eine Eigenschaft kann die Nutzererfahrung beeinträchtigen. Vor allem dann, wenn für eine Entity sehr viele Kommentare auf einmal geladen werden. Siehe Unterabschnitt 5.3.3 für weitere Details.

Moderne Webbrowser bieten hier eine simple Lösung: Web-Worker erlauben es, JavaScript abseits des Hauptfadens auszuführen. Somit können parallel zum Hauptthread, in dem React das Rendering durchführt, teure Berechnungen durchgeführt werden. Dazu muss lediglich ein entsprechendes JavaScript-Skript angegeben werden. Durch den Aufruf `new Worker("path/to/script.js")` wird der Web-Worker instanziiert. Diesem kann dann per `postMessage` eine Nachricht geschickt werden, bzw. über `addEventListener("message", msg => ...)` können Nachrichten empfangen werden. Gleichermaßen sendet und empfängt das Web-Worker Skript Nachrichten gegenüber dem Hauptthread.

Die einzige, und nicht zu unterschätzende, Schwierigkeit ergibt sich daraus, dass für den Web-Worker ein dediziertes Skript gegeben sein muss. Insbesondere können per `postMessage` keine Funktionen o.ä. an den Web-Worker übergeben werden. Alle Funktionen, die der Web-Worker beherrschen soll, müssen also von Anfang an in dessen Skript enthalten sein.

In NovaCrate wurde dafür der klassische Next.js-Bauprozess erweitert, um zusätzlich ein Skript für den Web-Worker zu produzieren. Dafür wurden alle Funktionen, die der Web-Worker unterstützen soll, in eine dedizierte Datei ausgelagert. Zusätzlich musste sichergestellt werden, dass diese Datei anderen Code nur aus Dateien importiert, die selbst keinen React-Code enthalten. Da der Web-Worker keinen Zugriff auf das DOM (Document Object Model, entspricht dem HTML und CSS der Website) hat, kann React nicht ausgeführt

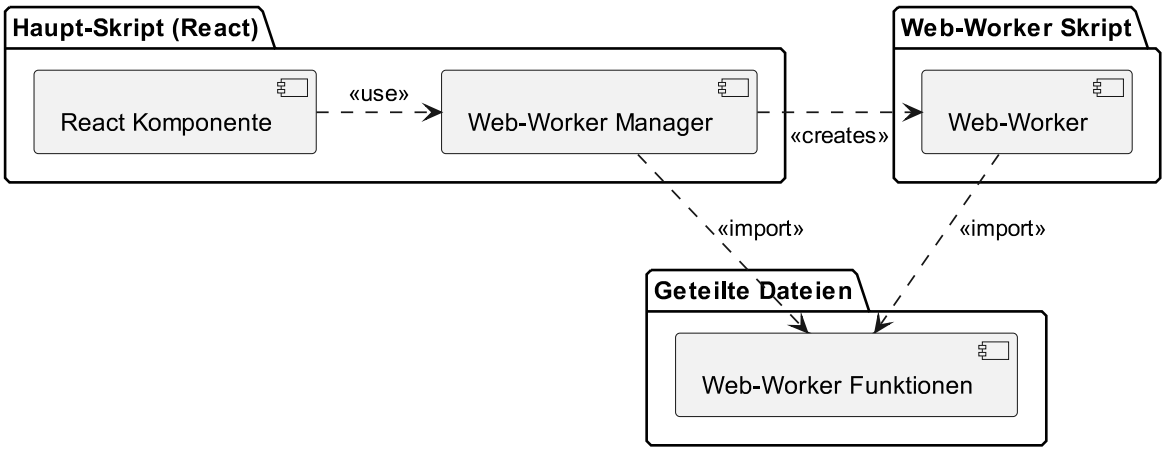

Abbildung 5.10.: Darstellung der Aufteilung in Haupt-Skript und Web-Worker Skript für den Schema Web Worker

werden und es würde zum Absturz des Web-Workers kommen. Diese Aufteilung ist in Abbildung 5.10 dargestellt.

Anschließend wurde eine Hilfsklasse erstellt, die das Interagieren mit dem Web-Worker deutlich vereinfacht. Die Anforderungen waren dabei, dass der Editor beliebig auf Funktionen des Web-Workers zugreifen können soll, ohne prüfen zu müssen ob dieser überhaupt läuft. Aufgerufene Funktionen sollten bei Ausfall des Web-Workers stattdessen lokal ausgeführt werden. Die Interaktion von beliebigen Komponenten mit dieser Hilfsklasse ist in Abbildung 5.11 dargestellt.

5.4.3. Performance Optimierungen

Im Gegensatz zu vielen anderen Frontend-Frameworks ist React sehr anfällig für Performance-Probleme. Wie in [34] bereits gezeigt, schneidet React im Vergleich zu Angular, Svelte oder Vue.js in Bezug auf die Performance am schlechtesten ab. Auch bei NovaCrate sind Performance-Probleme aufgetreten und haben zwischenzeitlich Zweifel an der Eignung von React geweckt. Dieser Abschnitt geht auf vier Optimierungsmechanismen ein, die diese Probleme eindämmen konnten.

Die Performance von React wird dabei in diesem Abschnitt einzig an der Zeit von der Änderung eines Zustandes bis zum fertigen Rendern der App bemessen. Wird ein Zustand geändert, so sorgt React automatisch dafür, das alle Komponenten neu rendern, die den Zustand verwenden. Alle Kinder von neu rendernden Komponenten werden ebenfalls neu gerendert. Das Rendern einer Komponente besteht dabei aus dem Aufruf der Funktion, die die Komponente definiert. Die Komponenten-Funktion gibt anschließend eine Komponente in HTML-JavaScript Mischform, genannt JSX (JavaScript Syntax Extension), zurück. Dieser Vorgang wird als rendern bezeichnet. Anschließend passt React das DOM (Document Object Model) der Website so an, das die intern gerenderten Komponenten durch den Browser

gezeichnet werden. Der Großteil der Arbeit liegt hier beim Rendern der Komponenten. Es liegt in der Hand der Programmierer, dafür zu sorgen, dass alle Komponenten zügig (bzw. selten genug) rendern. [20]

Während der Implementierung von NovaCrate war die allgemeine Performance der App zwischenzeitlich so schlecht, dass das Tippen in einem Textfeld des Entity Editors praktisch unmöglich war. Die Latenz von einem Tastenschlag bis zur Anzeige des getippten Buchstabens lag bei bis zu 200 Millisekunden. Entsprechend fühlte sich der Editor langsam und träge an. Um den Ansprüchen der Benutzbarkeit gerecht zu werden, musste die Performance optimiert werden.

Während der Implementierung wurden vier verschiedene Optimierungen abgewandt. Diese stammen zum Teil aus Arbeiten zu Performanceoptimierungen in React [8, 10] sowie aus der React Dokumentation [19].

1. React Context in Zustand Stores [33] umwandeln
2. Komponenten in der `memo`-Funktion [18] kapseln
3. Abhängigkeiten vom globalen Zustand reduzieren
4. Änderungsrate des globalen Zustands reduzieren

5.4.3.1. Umwandlung von React Context in Zustand Store

React Contexts führen schnell zu Performance-Problemen. Grund dafür ist, dass ein React Context immer durch eine klassische React Komponente (Provider-Komponente) bereitgestellt werden muss. Ändert sich der Zustand des Contexts, und damit der Zustand der Provider-Komponente, müssen alle Kind-Komponenten der Provider-Komponente neu gerendert werden. Außerdem können andere Komponenten nur auf einen Context zugreifen, wenn die zugehörige Provider-Komponente in der Komponenten-Hierarchie über ihnen liegt. Soll ein Context der gesamten App zugänglich gemacht werden, so muss er also sehr weit oben liegen. Entsprechend führt eine Aktualisierung des Contexts zum Neu-Rendern der gesamten App.

Besser geeignet als Zustandsbehälter sind Zustand Stores aus der Zustand [33] Bibliothek. Diese werden global definiert und sind in allen Komponenten verwendbar, ohne dass ein React Context oder eine Provider-Komponente nötig ist. Das eben genannte Problem kann hier also nicht auftreten.

Das Zustand Stores keine React Komponenten sind, ist zugleich auch ihr größter Nachteil. Sie können deshalb keine React Hooks verwenden (siehe Abschnitt 4.3 für eine Erläuterung von React Hooks). Dieses Problem lässt sich jedoch recht elegant umgehen, indem eine Singleton-React-Komponente erzeugt wird, die mit der Instandhaltung des Zustand Stores beauftragt wird. Diese Komponente kann frei React Hooks verwenden und kann bei Bedarf den Zustand Store anpassen. Sie kann außerdem beliebig tief in der Komponenten-Hierarchie sitzen.

Dieses Muster wurde für den Entity Editor Tabs Store verwendet. Dieser speichert die aktuell geöffneten Tabs, sowie welche Entität jeweils bearbeitet wird. Dabei soll sichergestellt werden, dass für jede Entität mit ungespeicherten Änderungen ein Tab geöffnet ist. Der Store muss also selbstständig bemerken, wenn eine Entität ungespeicherte Änderungen aufweist. Dies übernimmt die `EntityEditorTabsSupervisor`-Komponente. Mithilfe von React Hooks (in diesem Fall `useEffect`) kann die Komponenten einfach die Liste an geänderten Entitäten beobachten und fehlende Tabs öffnen lassen.

Ursprünglich war diese Funktionalität in einem einzelnen React Context implementiert. Durch die Umwandlung des Contexts in einen Zustand Store konnte ein erheblicher Performance-Gewinn erzielt werden. Bei einem großen Crate lag die Latenz bei einem Wechsel des aktiven Tabs bei deutlich spürbaren 300 Millisekunden. Nach der Umwandlung in den Zustand Store lag die Latenz nur noch bei 50 Millisekunden, was einer Beschleunigung um 600 % entspricht. Durch den Wechsel entstand kein Verlust an Funktionalität. Ursache war, dass bei Verwendung des React Context die gesamte App neu rendern musste, wenn der Tab gewechselt wurde. Nun rendern nur Komponenten neu, die tatsächlich von dem Tab-Wechsel betroffen sind.

5.4.3.2. Verwenden von React Memo

Die zweite, wohl trivialste Optimierung, ist das Schachteln einer Komponente in die React-Funktion `memo` [18]. Normalerweise führt das Rendern einer Komponente in React auch zum neu rendern aller Kind-Komponenten. Die `memo`-Funktion unterbindet dies jedoch, und lässt das Kind nur neu rendern, wenn tatsächlich nötig. Das ist genau dann, wenn eine direkte Abhängigkeit der Komponente sich geändert hat.

Besonders bei großen Listen ist dies eine wichtige Optimierung. Normalerweise würde React die gesamte Liste neu zeichnen, wenn sich der Zustand der Listenkomponente ändert. Jedes einzelne Listenelement würde also neu gezeichnet werden. Sind die Listenelemente wiederum teure Komponenten, kommt es schnell zu Performance-Problemen.

Indem alle Listenelemente in der `memo`-Funktion geschachtelt werden, rendern sie nur neu, wenn sich ihr eigener Inhalt geändert hat. So können auch sehr große Listen mit mehreren tausend Einträgen gut dargestellt werden. Während das initiale Rendern der Liste eine Weile dauern wird, können anschließende Änderungen die nur einzelne Elemente betreffen problemlos durchgeführt werden.

5.4.3.3. Reduzieren der Abhängigkeiten

Ein weiteres wichtiges Muster zur Verbesserung der Performance ist das Reduzieren der Abhängigkeiten einer Komponente. Es ist das grundlegende Konzept von React, dass eine Komponente neu rendert, wenn eine ihrer Abhängigkeiten sich ändert. Somit liegt es nahe, die Abhängigkeiten in kritischen Komponenten auf das absolut Notwendige zu reduzieren. Ein gutes Beispiel ist der Entity Browser, der eine vollständige Liste aller Entities im Crate

anzeigt. Zusätzlich zeigt jeder Eintrag im Entity Browser an, ob die entsprechende Entity seit dem letzten Speichern geändert wurde.

Ursprünglich hatte dafür jeder Eintrag eine vollständige Liste der geänderten Entities im Crate angefragt. Dies ist in Abbildung 5.12 zu sehen. Die Rendering-Zeit hing also quadratisch von der Menge an Entities im Crate ab. Die simple und dennoch sehr effektive Lösung war schließlich, in jedem Eintrag nur den Status der zugehörigen Entity abzufragen.

Durch das zusätzliche Verwenden eines guten Selektors für den Entity Zustand Store wurde die Änderungsanzeige nur neu gezeichnet, wenn sie sich tatsächlich ändern sollte. Dies ist in Abbildung 5.13 zu sehen. Beim Editieren einer Entity kam es vor dieser Änderung zu einer Latenz von etwa 50 Millisekunden für jeden Tastendruck. Nach der Anpassung lag die Latenz nur noch bei 8 Millisekunden (Beschleunigt um mehr als 500 %). Aber auch Abseits dieses Beispiels konnte das Reduzieren von Abhängigkeiten an vielen Stellen zu verbesserter Leistung beitragen.

5.4.3.4. Verzögerte Änderung des globalen Zustandes

Ursprünglich führte das Tippen in einem Eingabefeld direkt zur Änderung des Inhaltsgebenden globalen Zustandes. Diese akribische Aktualisierung des Zustandes ist zwar React-Standard, jedoch in diesem Fall nicht nötig. Die Eingabefeld-Komponente wurde nun so angepasst, dass sie ihre Änderungen erst nach einer Verzögerung von 300 Millisekunden an den globalen Zustand weitergibt. Bei kontinuierlichen Tippen kommt es also nur alle 300 Millisekunden zu einer Änderung des globalen Zustandes. Da die Eingabefeld-Komponente ihren internen Zustand selbst hält, wird auch nur sie selbst während des Tippens neu gezeichnet. Dies ist in Abbildung 5.14 zu sehen. Die zuvor geringste Latenz von etwa 8 Millisekunden pro Tastendruck konnte durch diese einfache Optimierung auf weniger als ein Zehntel einer Millisekunde pro Tastendruck verbessert werden. Einzig das alle 300 Millisekunden stattfindende aktualisieren des globalen Zustandes führt zu einer kaum bemerkbaren Verzögerung, die etwa 8 Millisekunden andauert.

Für den Nutzer entsteht durch diese Anpassung keinerlei Nachteil. Zumal diese Änderung sich sowieso nur bei der Bearbeitung der `name`-Eigenschaft bemerkbar macht. So wird der Name des aktuellen Tabs, der dem Namen der Entity entspricht, nicht mehr bei jedem Tastendruck aktualisiert, sondern in längeren regelmäßigen Abständen.

Die meisten Optimierungen dieses Abschnitts sind recht simpel und schränken die Funktionalität des Editors in keiner Weise ein. Umso bedauernswerter ist es, das React nicht in der Lage ist, diese Probleme selbst zu verhindern. Andere Frontend-Frameworks scheinen React an dieser Stelle etwas voraus zu sein. [34]

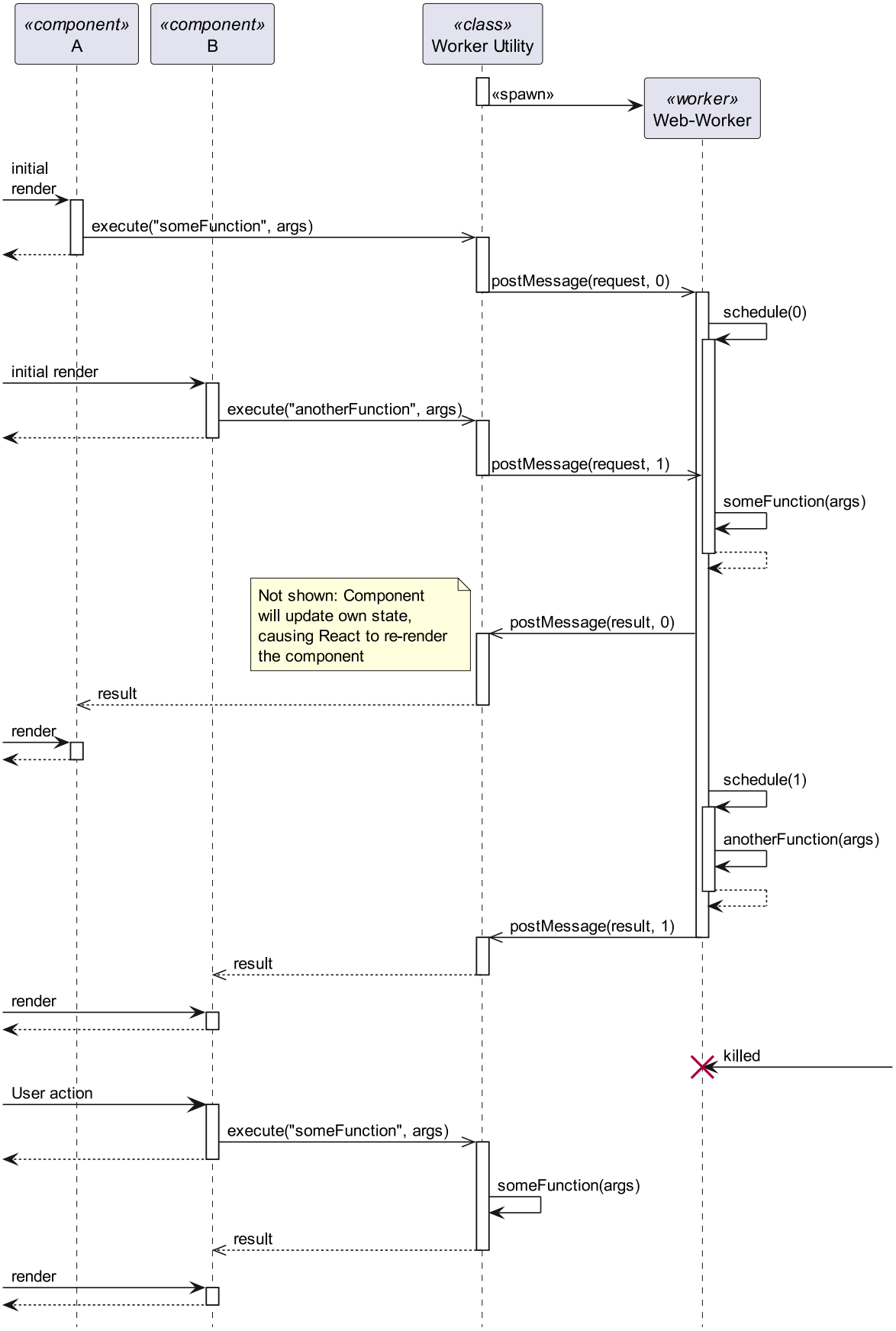

Abbildung 5.11.: Sequenzdiagramm zur Interaktion der React Komponenten mit dem Schema Web-Worker

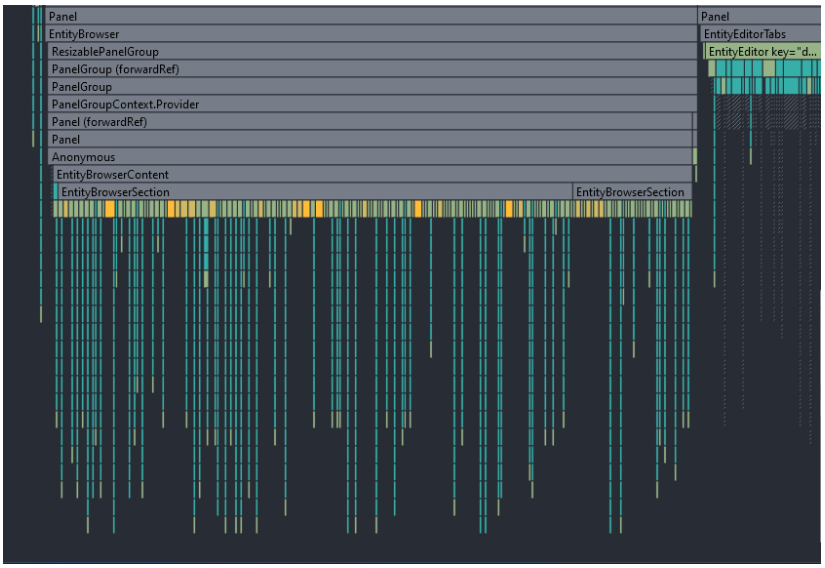

Abbildung 5.12.: Neu rendernde Komponenten bei Tastenschlag, ohne Optimierung. In Farbe sind Komponenten, die neu gerendert wurden, dargestellt (grün: kurze Verzögerung, gelb: große Verzögerung).

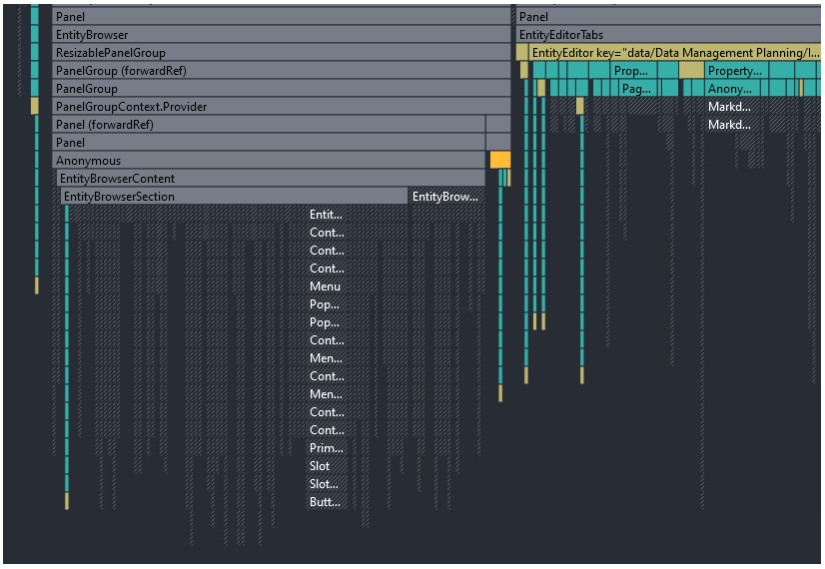

Abbildung 5.13.: Neu rendernde Komponenten bei Tastenschlag, mit optimiertem Entity Browser. In Farbe sind Komponenten, die neu gerendert wurden, dargestellt (grün: kurze Verzögerung, gelb: große Verzögerung). Dunkelgraue Komponenten wurden wegen der `memo`-Funktion nicht erneut gerendert.

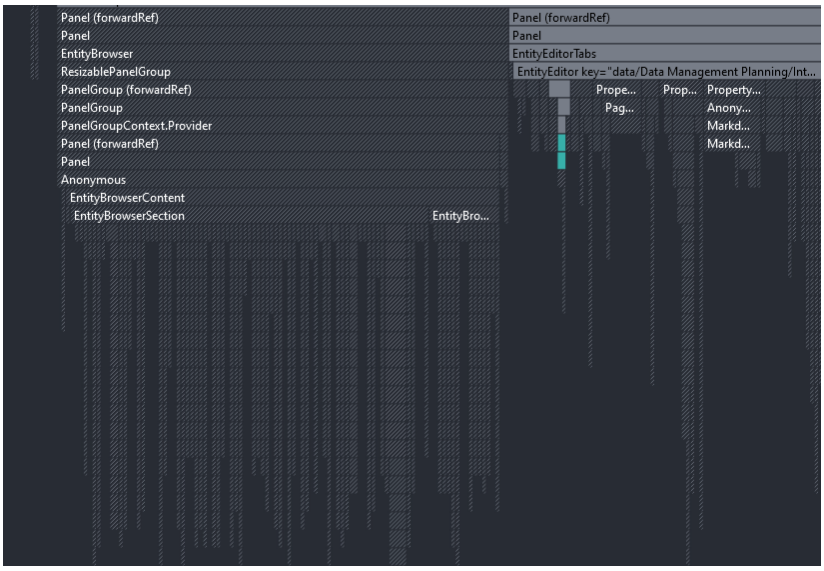

Abbildung 5.14.: Neu rendernde Komponenten bei Tastenschlag, mit verzögerten Änderungen und optimiertem Entity Browser. In Farbe sind Komponenten, die neu gerendert wurden, dargestellt (grün: kurze Verzögerung, gelb: große Verzögerung). Dunkelgraue Komponenten wurden wegen der `memo`-Funktion nicht erneut gerendert.

5.5. Erweiterung der REST-API der RO-Crate Bibliothek

Ebenfalls Teil dieser Arbeit war die Erweiterung der bereits bestehenden REST-API [36] für die RO-Crate Java Bibliothek [47]. Diese enthielt bereits den Großteil der benötigten Funktionalitäten.

Die wichtigste Änderung befasst sich mit dem Bearbeiten einer Entity. Ursprünglich wurden die Änderungen an einer Entity angewendet, indem diese gelöscht und neu hinzugefügt wurde. Dies hat jedoch zu unvorhergesehenen Problemen geführt, da ro-crate-java automatisch alle Referenzen zu Entities entfernt, die gelöscht wurden. Die Rest-API wurde also um entsprechende `PATCH`-Routen (dt. Korrektur) erweitert, welche die verschiedenen Entity-Arten ohne Löschen anpassen können.

Außerdem wurden zuvor nicht unterstützte Funktionen hinzugefügt. Dies umfasste den Import von Personen aus der ORCID [30] Datenbank, sowie von Unternehmen aus der ROR [4] Datenbank. Zudem wurden Optimierungen vorgenommen, um das schnelle Hochladen mehrerer Dateien, meist in Form eines Ordners, zu ermöglichen.

6. Evaluation

Zur Evaluation des Editors wurde im Rahmen dieser Arbeit ein Nutzertest durchgeführt. Aus den Ergebnissen des Nutzertestes soll abgeleitet werden, ob der Editor die in Abschnitt 5.1 formulierten Anforderungen erfüllen kann. Hierbei handelt es sich lediglich um eine qualitative Evaluation der Benutzbarkeit und Funktionalität, die nicht dem Zweck eines Vergleiches mit den anderen Editoren dienen soll.

Als Definition für Usability wurde hier die allgemein anerkannte Definition von Nielsen verwendet: „A system is usable if it is [...] easy to learn, [...] efficient, [...] easy to remember, [...] relatively error-free or error-forgiving [...] and pleasant to use [...]." [26, S. 33]. Diese fünf Kriterien werden hier durch die nichtfunktionalen Anforderungen /N1/ (Einfach), /N2/ (Intuitiv) und /N3/ (Unterstützung) abgedeckt.

Wie bereits in Abschnitt 5.3 erwähnt, sind die verbleibenden nichtfunktionalen Anforderungen durch den Entwurf begründet.

6.1. Methodik

Die Evaluation bestand aus einem simplen Nutzertest nach Steve Krug [12]. Ein Nutzertest dauert eine Stunde und wird mit jeweils einer Testperson durchgeführt. Ziel ist zunächst die Evaluation der Benutzbarkeit, jedoch werden durch die Aufgaben auch die meisten funktionalen Anforderungen geprüft.

Als Testpersonen wurden fünf Mitarbeitende der Arbeitsgruppe Data Exploitation Methods des Scientific Computing Center am KIT ausgewählt. Hierbei handelt es sich hauptsächlich um Forschende, die der Zielgruppe des Editors entsprechen. Die Hintergründe der Testpersonen reichen hier vom Fachgebiet der Digital Humanities, über Germanistik, bis in die Biologie, Mathematik und Informationstechnik (vgl. A.3.1 0:04, A.3.2 0:18, A.3.3 0:09, A.3.4 0:03, A.3.5 0:05). Somit wurde die Eignung des Editors für die tatsächliche Zielgruppe und ihren zu erwartenden Aufgaben getestet. Die Umgebung der Tests ist dabei ebenfalls nah an der Realität, denn der Nutzertest wurde im Büro der Arbeitsgruppe durchgeführt. Ein realitätsnaher Testaufbau ist wichtig, um die Ergebnisse nicht zu verfälschen. [2, S. 3].

Dass viele der Testpersonen noch keine Erfahrung mit Research Object Crates haben, geschweige denn mit RO-Crate Editoren, ist auch laut Nielsen kein Problem [26, S. 34]. So kann besonders gut bestimmt werden, wie einfach der Editor zu erlernen ist, bzw. wie leicht neuen Nutzern die Einarbeitung in den Editor fällt.

Der Nutzertest folgt dem Schema von Steve Krug [12]. Dieser beschreibt in seinem Buch „Rocket surgery made easy“ [12] eine einfache und doch sehr effiziente Möglichkeit, Nutzertests kostengünstig und mit wenig Zeitaufwand durchzuführen. Dabei werden die Testpersonen je für eine Stunde eingeladen. Der gesamte Test wird durch eine Bildschirmaufnahme und ein Mikrofon aufgezeichnet. So kann die Sitzung nachträglich evaluiert werden. Zu Beginn wird ein Skript vorgelesen, dass die Testperson in den Nutzertest einführt und die Rahmenbedingungen klärt (siehe Abschnitt A.1). Anschließend werden der Testperson einige Aufgaben gestellt. Diese sind jeweils als Szenarien ausgestaltet, um möglichst genau eine spezifische Situation zu simulieren [12, S. 48]. Die Bewältigung der Aufgaben soll etwa eine halbe Stunde dauern. [12]

6.1.1. Aufgabenstellungen

Die Aufgabenstellungen für den Nutzertest sind so gestellt, dass mit hoher Wahrscheinlichkeit alle wichtigen Funktionen des Editors durch die Testperson verwendet werden. Um auch die Benutzbarkeit und Intuitivität des Editors zu testen, ist es wichtig, den Testpersonen nicht vorzuschreiben, wo sie eine bestimmte Funktion finden können. Stattdessen wird von ihnen verlangt, eine bestimmte Änderung am aktuellen Crate durchzuführen, ohne ins Detail zu gehen, wie dies mit dem Editor idealerweise ablaufen würde. So wird getestet, wie gut die Testperson die entsprechende Funktion findet, oder ob intuitiv klar ist, welchen Effekt die verfügbaren Funktionen haben.

Dennoch kann nicht der gesamte Funktionsumfang des Editors getestet werden. Außerdem kann es passieren, dass manche Testpersonen bestimmte Seiten oder Funktionen überhaupt nicht benutzen, weil sie diese nicht benötigt haben. Ein vollumfänglicher Nutzertest, der alle Funktionen und Seiten im Detail prüft, wäre extrem aufwändig und würde wohl keinen großen Mehrwert bieten. So ist davon auszugehen, dass die wichtigsten Probleme der Benutzerfreundlichkeit auch schon bei einem Test mit geringerem Umfang gefunden werden. Probleme die in allen Nutzertests nicht gefunden werden, sind für die Grundfunktionalität des Editors schließlich nicht relevant und nicht ausschlaggebend für die Benutzbarkeit des Editors insgesamt. [12, S. 46ff][2, S. 5]

In Tabelle 6.1 sind die einzelnen Aufgaben aufgelistet, sowie eine kurze Beschreibung, welche Seiten und funktionalen Anforderungen, zusätzlich zu den nichtfunktionalen Anforderungen /NF1-3/, durch die Aufgaben abgedeckt werden sollen. Die Aufgaben wurden für den Nutzertest als Szenarien ausgestaltet, welche in Abschnitt A.2 angehängt sind.

Das in Aufgabe 1 zu importierende Crate wird vor dem Nutzertest vorbereitet und ist für alle Testpersonen zu Beginn gleich. Alle weiteren benötigten Ressourcen sind ebenfalls im Vorhinein vorbereitet und für alle Testpersonen gleich.

#	Aufgabe	Abdeckung
1	Importieren eines Crates	• Startseite des Editors • /F2/ (Crate import)
2	Ändern einer Eigenschaft	• Entity Editor • Evtl. File Explorer • /F9/ (Ändern von Eigenschaften) • /F11/ (Dateivorschau)
3	Hinzufügen einer Eigenschaft	• Entity Editor • /F9/ (Ändern von Eigenschaften)
4	Löschen einer Eigenschaft	• Entity Editor • /F9/ (Ändern von Eigenschaften)
5	Hinzufügen einer Data Entity	• Entity Editor • Evtl. File Explorer • /F5/ (Entity erstellen) • /F6/ (Data Entity erstellen)
6	Anpassen von Referenzen im Graph	• Graph • /F9/ (Ändern von Eigenschaften) • /F10/ (Interaktiver Graph)

Tabelle 6.1.: Aufgabenstellungen für den Nutzertest sowie die erwarteten abgedeckten Seiten bzw. funktionalen Anforderungen des Editors

6.2. Durchführung

Der Nutzertest wurde am 22. Juli in einem Raum am SCC Campus Nord (Geb. 449, KIT Campus Nord) durchgeführt. Die fünf Testpersonen (vgl. Abschnitt 6.1) wurden für je eine Stunde eingeladen.

Die Tests wurden mit der jeweils neusten Version von NovaCrate, der RO-Crate Java Bibliothek [47] und der REST-API [36] auf einem aktuellen iMac durchgeführt. Mit dem vorinstallierten QuickTime Player wurden die Nutzertests aufgenommen, wobei das integrierte Mikrofon für die Aufzeichnung der Stimme verwendet wurde.

Zu Beginn des Tests wurde, wie in Abschnitt 6.1 beschrieben, das Skript (siehe Abschnitt A.1) vorgelesen. Da der Nutzertest ausgezeichnet wurde, musste außerdem eine Einverständniserklärung unterzeichnet werden. Anschließend wurden die Szenarien (siehe Abschnitt A.2) vorgelesen nacheinander ausgeteilt.

Während für den Nutzertest jeweils eine Stunde eingeplant war, lag die tatsächlich benötigte Zeit zum Bewältigen der Aufgaben deutlich darunter. Alle Proband*innen waren nach etwa 20 Minuten fertig mit den Aufgaben. Anschließend blieb Zeit für eine allgemeine Unterhaltung zum Editor bzw. RO-Crates im Allgemeinen.

Zu guter Letzt wurde die Aufnahme beendet und gespeichert. Alle Aufnahmen wurden vor Ende des Tages von dem verwendeten iMac gelöscht.

6.3. Auswertung

In diesem Abschnitt wird kurz beschrieben, wie die Aufnahmen des Nutzertests ausgewertet wurden. Die Diskussion der Ergebnisse findet in Kapitel 7 statt.

6.3.1. Transkribieren der Aufzeichnungen

Für die Auswertung des Nutzertests wurden Transkripte der Aufzeichnungen angefertigt. Über diese kann schließlich ermittelt werden, welche Probleme der Benutzbarkeit aufgedeckt werden konnten. Außerdem lässt sich ein Gesamteindruck gewinnen, ob die Testpersonen den Editor einfach und intuitiv in der Benutzung empfanden.

Die Transkription der Tonspur wurde zunächst durch das KI-Sprachmodell „Whisper“ [29] von OpenAI durchgeführt. Dieses wurde lokal auf einer modernen Grafikkarte durchgeführt, sodass keine Daten an zweite oder dritte gelangt sind. Die Transkription eines 20-minütigen Gesprächs dauerte hier etwa 20 Sekunden.

Anschließend wurden die von der KI erstellten Transkripte mit der Aufzeichnung abgeglichen und, nach bestem Gewissen, alle Fehler korrigiert. Außerdem musste der aktuelle Sprecher für jeden Satz nachgetragen werden. Dies wurde über das kostenlose Online-Tool „HappyScribe“ [6], das normalerweise für die Erstellung von Untertiteln verwendet wird, durchgeführt. Auch hier wurden keine persönlichen Daten an zweite oder dritte weitergegeben.

Die fertigen Transkripte sind in Abschnitt A.3 angehängt.

6.3.2. Ergebnisse

Insgesamt konnten alle sechs Aufgaben von allen Teilnehmer*innen absolviert werden. Der Nutzertest konnte erfolgreich und problemlos durchgeführt werden. Die Evaluation in Bezug auf die Anforderungen aus Abschnitt 5.1 wird in Abschnitt 7.1 diskutiert.

7. Diskussion

In diesem Abschnitt werden die Ergebnisse der Evaluation sowie mögliche Lösungsansätze und Auswirkungen auf die Anforderungen an den Editor diskutiert. Außerdem werden Vorschläge zum produktiven Einsatz von NovaCrate vorgestellt.

7.1. Anforderungen und Evaluation

Alle Teilnehmer*innen des Nutzertests konnten alle sechs Aufgaben bewältigen. Dennoch sind in allen Aufgaben Probleme der Benutzbarkeit aufgetreten, die hier genauer betrachtet werden. Dabei wird jede getestete Funktion, wie in Tabelle 6.1 aufgelistet, einzeln unter Zunahme der Anforderungen aus Abschnitt 5.1 betrachtet.

7.1.1. Startseite des Editors und Import von Crates

In der ersten Aufgaben sollte ein vorbereitetes Crate importiert werden. Vier Testpersonen hatten den Aufbau der Startseite intuitiv verstanden und zügig (in 6 bis 25 Sekunden) die richtige Schaltfläche gefunden (vgl. A.3.1 03:51, A.3.2 05:24, A.3.4 05:40, A.3.5 05:52). Bei einer Testperson kam es zu einer Verwechslung der Importfunktion mit dem „Start with Data“-Button, diese wurde jedoch nach etwa einer Minute überwunden (vgl. A.3.3 03:16). Hier könnte die Benennung der Funktionen verbessert werden, um eindeutiger zwischen dem Importieren und Erstellen eines Crates zu unterscheiden.

Letztendlich empfanden alle Proband*innen den Aufbau der Startseite intuitiv und übersichtlich, bis auf die Duplizierung der Schaltflächen „Start with Data“ und „Start from Scratch“. Während diese Duplizierung von einigen Personen direkt als solche identifiziert wurde (vgl. A.3.4 05:05), sorgte sie bei anderen Testpersonen für kurze Verwirrung (vgl. A.3.3 04:13).

Der Importvorgang an sich war allen klar und sorgte nicht für Probleme.

7.1.2. Datei-Preview im Entity Editor

In der zweiten Aufgabe sollten zwei Eigenschaften einer Data Entity bearbeitet werden. Um die korrekten Werte der Eigenschaften zu finden, mussten die Testpersonen zunächst das zugehörige Bild der Entity anzeigen lassen. Konkret waren die Eigenschaften „Caption“ und

„Copyright Year“ gefragt. Eine Person suchte hier nach den EXIF-Metadaten des Bildes, die der Editor jedoch nicht bereitstellte (vgl. A.3.3 08:48).

In allen Tests hatte die Proband*in eine Funktion zum Anzeigen des Bildes zunächst an anderen Stellen vermutet (vgl. A.3.1 07:02, A.3.2 07:19, A.3.4 10:05, A.3.5 08:29). Die richtige Schaltfläche wurde von allen erst recht spät (durchschnittlich nach 37 Sekunden) gefunden. Zwei Proband*innen fanden die Schaltfläche gar nicht, und griffen auf den File Explorer zurück (vgl. A.3.1 07:25, A.3.3 07:08). Die effektivste Lösung wäre wohl das Einfügen einer zusätzlichen Schaltfläche, an der Stelle, an der die Proband*innen sie vermuteten.

7.1.3. Entity Editor

Das Bearbeiten der Eigenschaften über den Entity Editor stellte kein Problem dar. Alle Testpersonen hatten die Eigenschaften direkt gefunden (höchstens nach 5 Sekunden) und hatten keine Probleme bei der Bearbeitung (vgl. A.3.1 06:45, A.3.3 05:49, A.3.4 09:18, A.3.5 08:03). Lediglich die fehlende Möglichkeit zum Vergrößern der Textfelder wurde von zwei Testpersonen kritisiert (vgl. A.3.2 07:52, A.3.5 09:45).

Die dritte Aufgabe umfasste das Hinzufügen einer Eigenschaft zu zwei Entities. Dabei sollte jeweils dieselbe Eigenschaft zu zwei Entities mit dem gleichen Wert hinzugefügt werden. Es handelte sich um eine Referenz-Eigenschaft, die als Wert auf eine andere Entity verweist. Wichtig war hier, dass der Nutzer bei einem leeren Referenz-Feld die Wahl zwischen dem Erstellen einer neuen Entity und dem Verlinken einer existierenden hat. Die Testpersonen wurden informiert, dass die Ziel-Entity bereits existiert.

Das Hinzufügen der Eigenschaft an sich war für alle Testpersonen einfach und intuitiv. Die entsprechende Schaltfläche war nach höchstens 10 Sekunden gefunden (vgl. A.3.1 10:27, A.3.2 10:13, A.3.3 10:20, A.3.4 13:04, A.3.5 11:24). Lediglich das Verlinken der richtigen Entity sorgte bei drei Personen für Verwirrung. Ihnen war nicht klar, was der Unterschied zwischen den „Create“ und „Select“ Schaltflächen ist. Hier wurden dann zum einen die Symbole der Schaltflächen kritisiert, als auch das Fehlen einer Erklärung der Funktionen (vgl. A.3.3 10:47, A.3.4 14:19 und 16:30, A.3.5 11:58). Ein Tooltip mit einer knappen Erklärung sollte dieses Problem lösen können.

Die Aufgabe sah explizit vor, dass dieselbe Eigenschaft zweimal hinzugefügt werden soll. Dadurch sollte getestet werden, ob die Testpersonen den Vorgang beim zweiten Mal bereits ohne jegliche Probleme durchführen konnten. Dies war bei allen Proband*innen der Fall, denn der zweite Durchgang dauerte im Durchschnitt nur 15 Sekunden (vgl. A.3.1 11:38, A.3.2 10:37, A.3.3 13:47, A.3.4 15:29, A.3.5 12:37). Somit waren die Funktionen des Referenz-Feldes zunächst nicht eindeutig klar, aber dennoch einfach zu erlernen und zu verwenden.

Die wohl einfachste Aufgabe (Nr. 4) befasste sich mit dem Löschen einer Eigenschaft. Hier sollten die Testpersonen die richtige Eigenschaft finden, und anschließend über dessen 3-Punkte-Menü löschen. Fast alle Testpersonen hatten dieses intuitiv gefunden und die Aufgabe schnell (durchschnittlich nach 7 Sekunden) gelöst (vgl. A.3.1 12:23, A.3.2 12:39, A.3.3 15:01, A.3.4 17:06). Lediglich eine Proband*in hatte zunächst den „Clear“-Button versucht,

war aber kurz darauf auch auf dem richtigen Weg (vgl. A.3.5 14:06). Dennoch war die Funktion des 3-Punkte-Menüs klar und intuitiv.

Da es keine technischen Probleme gab, sind alle zutreffenden technischen Anforderungen erfüllt. Die Entity Editor-Seite ist einfach zu erlernen und unterstützt die Nutzer. Lediglich Probleme bei der Intuitivität wurden beobachtet. (Vgl. Anforderungen aus Abschnitt 5.1)

7.1.4. Modal zum Hinzufügen von Entities und Datei-Upload

In der fünften Aufgabe sollte eine neue Data Entity durch das Hochladen einer vorbereiteten Datei erstellt werden.

Hier wurde angemerkt, dass die Aufgabenstellung womöglich zu viele Hinweise gab (vgl. A.3.2 14:19). Diese wies darauf hin, dass der richtige Typ für die geforderte Entity „File“ ist. Das Wählen des richtigen Typs aus dem Modal war dementsprechend sehr einfach und dauerte etwa 5 Sekunden (vgl. A.3.1 14:23, A.3.2 13:39, A.3.3 18:27, A.3.4 17:50, A.3.5 15:45). Dennoch haben andere Testpersonen angebracht, dass die Wahl auch ohne diesen Hinweis nicht schwergefallen wäre (vgl. A.3.1 14:18, A.3.2 15:53). Die Auswahl des richtigen Entität-Typs ist also unproblematisch, also unterstützt der Editor die Nutzenden an dieser Stelle.

Lediglich die darauf folgende Ansicht hat bei allen Testpersonen für Verwirrung gesorgt. Am oberen Rand kann über Tabs ausgewählt werden, ob eine Datei hochgeladen werden soll. Diese wurden jedoch fälschlicherweise als Buttons für den Upload identifiziert (vgl. A.3.1 14:43, A.3.3 20:11, A.3.4 17:56, A.3.5 16:15). Diese Maske ist also nicht intuitiv. Nachdem die Proband*innen den richtigen Button versucht hatten, war die Funktionalität jedoch klar.

Eine Testperson, die bereits Erfahrung mit RO-Crates hatte, konnte ein funktionales Problem des Modals aufdecken (A.3.3 24:51). Dieses begünstigte das Erstellen inkorrekter Data Entities. Somit verfehlt das Modal an dieser Stelle die nichtfunktionalen Anforderungen /N3/ (Unterstützung für Bearbeitung des Crates) und /N2/ (Intuitivität) sowie die funktionalen Anforderungen /F6/ (Generieren von Data Entities) aus Abschnitt 5.1.

Abseits des technischen Problems und der Verwirrung über den Upload-Button gab es jedoch keine Probleme, damit sollten die entsprechenden Anforderungen größtenteils erfüllt sein. Da dieses Modal jedoch recht komplex ist, und der Nutzertest nur eine Schnittmenge der Funktionen betrachtet hat, braucht es einen ausführlicheren Nutzertest für eine vollständige Evaluation des Modals.

7.1.5. Graph

Der Graph ist in der Evaluation besonders interessant, da die Benutzerinteraktion sich hier völlig vom Rest des Editors unterscheidet. Eine Proband*in benötigten daher einen kurzen Augenblick (150 Sekunden), um die Bedienung des Graphen nachzuvollziehen (vgl. A.3.1 17:22, A.3.2 18:05, A.3.4 20:00).

Alle anderen Proband*innen haben bereits nach durchschnittlich 8 Sekunden herausgefunden, dass sich Knoten durch das Ziehen von Kanten verbinden lassen. Dies wurde als sehr intuitiv und einfach empfunden (vgl. A.3.1 21:01, A.3.2 17:42, A.3.3 26:06, A.3.4 20:20, A.3.5 18:40). Gleichzeitig stellte sich jedoch heraus, dass es hier relativ leicht ist, Kanten zu ziehen, die laut Spezifikation nicht erlaubt währen. Somit ist zwar die nichtfunktionale Anforderung /N3/ verfehlt, dennoch halten die /N1/ und /N2/ (vgl. Abschnitt 5.1).

7.1.6. Allgemeine Struktur

Der allgemeine Aufbau des Editors wurde nicht explizit evaluiert. Natürlich sind alle Funktionen des Editors in dessen Struktur eingebettet, wodurch die Struktur auch automatisch einem Test unterzogen wird. Die in Abschnitt 5.2 erwähnte „inverted L navigation“ [46, S. 132] war allen Proband*innen ohne weiteres zugänglich. Das globale Funktionsmenü am oberen Rand wurde jedoch nicht verwendet, da die Testpersonen alle Funktionen an einer anderen Stelle gefunden hatten. Die globale Navigation wurde sofort als solche erkannt und ohne Probleme verwendet (vgl. A.3.1 05:05, A.3.2 06:36 und 17:01, A.3.3 04:27, A.3.4 06:11 und 21:02).

Andere, allgemeinere Funktionen, wie das alle Änderungen in einem Zwischenzustand gehalten werden, kamen ebenfalls gut an. Diese Funktion wurde von allen ohne weiteres Verstanden und sogar gelobt (vgl. A.3.3 16:19). Auch die Tab-Navigation im Entity Editor, oder die Weiterleitung auf den Entity Editor vom File-Explorer oder Graph aus, hat keine Probleme gemacht. Den Proband*innen war direkt klar, dass der Entity Editor das Herzstück der Anwendung bildet (A.3.3 14:53).

Die allgemeine Struktur des Editors ist also intuitiv und einfach zu verstehen. Es kam zu keinen technischen Problemen, wodurch alle zutreffenden technischen Anforderungen, sowie die /N1/ bis /N3/ erfüllt sind (vgl. Abschnitt 5.1).

7.1.7. Fazit

Insgesamt haben alle getesteten Komponenten des Editors den Test gut überstanden. Lediglich ein technisches Problem wurde in Bezug auf /F7/ festgestellt (siehe Unterabschnitt 7.1.4). Probleme der Benutzbarkeit gab es an wenigen Stellen, besonders gravierend jedoch bei /N3/. Über das Modal zum Hinzufügen von Entities wird das Erstellen inkorrekter Entities begünstigt (siehe ebenfalls Unterabschnitt 7.1.4). Auch der Graph verhindert das Erstellen ungültiger Kanten nicht (vgl. Unterabschnitt 7.1.5). Die nichtfunktionalen Anforderungen der Erweiterbarkeit, Wartbarkeit und Portierbarkeit können nicht durch einen Nutzertest evaluiert werden und sind allein durch die technische Architektur in Abschnitt 5.3 begründet.

Insgesamt haben alle Proband*innen eine positive Meinung zum Editor geäußert und diesen insgesamt als einfach und intuitiv bezeichnet (vgl. A.3.1 21:41, A.3.2 11:57 und 14:11, A.3.3

28:56, A.3.4 21:02, A.3.5 20:30). Eine Zusammenfassung der Erfüllung der Anforderungen ist in Tabelle 7.1 dargestellt.

Anforderung	NovaCrate	Referenz
/F1/	Erfüllt	-
/F2/	Erfüllt	7.1.1
/F3/	Erfüllt	A.3.3 33:30
/F4/	Erfüllt	-
/F5/	Erfüllt	A.3.3 21:48, 7.1
/F6/	Erfüllt	7.1.4
/F7/	Nicht erfüllt	7.1.4
/F8/	Erfüllt	-
/F9/	Erfüllt	7.1.3
/F10/	Erfüllt	7.1.5
/F11/	Erfüllt	7.1.2
/F12/	Erfüllt	-
/N1/	Erfüllt	7.1
/N2/	Erfüllt	7.1
/N3/	Größtenteils erfüllt	7.1.4, 7.1.5
/N4/	Erfüllt	5.3
/N5/	Erfüllt	5.3
/N6/	Erfüllt	5.3

Tabelle 7.1.: Funktionale und nichtfunktionale Anforderungen an NovaCrate und ob diese im Nutzertest erfüllt wurden. Die Definition der Anforderungen ist in Abschnitt 5.1 zu finden. Als Referenz sind Abschnitte hinterlegt, welche die Anforderung behandeln.

7.2. Produktiver Einsatz von NovaCrate

NovaCrate ist momentan zwar bereits voll funktionsfähig, kann jedoch noch nicht produktiv eingesetzt werden. Die in Abschnitt 4.2 erwähnte REST-API muss manuell im Hintergrund gestartet werden und NovaCrate muss für jeden Einsatz aus dem Quellcode gebaut werden. In diesem Abschnitt sollen mögliche Einsatzgebiete von NovaCrate sowie deren voraussichtlicher Implementierungsaufwand diskutiert werden.

Eine Möglichkeit hierfür wäre der verteilter Einsatz von NovaCrate. Die REST-API sowie der NovaCrate Web-Server würden dann auf einem dedizierten Server aufgesetzt werden, sodass über den Webbrowser ganz unkompliziert eine Instanz aufgerufen werden kann. Problematisch ist hier, dass NovaCrate momentan nicht für simultane Nutzung durch mehrere Nutzer geeignet ist. Es fehlt an einer Nutzerverwaltung sowie einer sinnvollen Synchronisationsmethode für den Fall, dass zwei Nutzer*innen das gleiche Crate bearbeiten. Beides sollte mit geringem Aufwand in NovaCrate integriert werden können, wenn auch der Aufwand für ein entsprechendes Backend recht hoch sein könnte.

Auch weitere Einsatzgebiete, abseits einer dedizierten NovaCrate Instanz, sind denkbar. So könnte der Entity Editor als einzelne Komponente in ein existierendes Projekt im Bereich des Forschungsdatenmanagements integriert werden. Als losgelöste Komponente ist der Entity Editor zwar nicht ohne weiteres einsetzbar, das Entfernen der globalen Navigation würde aber schon genügen, um zumindest den Anschein zu erwecken. NovaCrate könnte so für viele verschiedene Projekte eingesetzt werden.

Besonders einfach wäre letztendlich die Ausführung als lokale Desktop-App, beispielsweise über Tauri [44]. Dabei handelt es sich um ein Framework, dass Webseiten als eigenständige Anwendung ausführt, und dabei auch zusätzlich benötigte Anwendungen, wie die REST-API, selbst im Hintergrund ausführen kann. Ähnliche Ansätze verfolgen zum Beispiel das neue Microsoft Teams (eine React-App die im Edge Web-View läuft) [22] oder auch Describo [13] (vgl. Abschnitt 3.1).

8. Fazit

Research Object Crates sind ein wichtiger neuer Mechanismus um digitale Forschungsdaten austauschbar und Forschung reproduzierbarer zu machen (vgl. Kapitel 1). Während bereits viele Bibliotheken zur programmatischen Erstellung von RO-Crates existieren, gab es bisher nur zwei interaktive Editoren. Das Ziel der Arbeit war die Entwicklung des benutzerfreundlichen RO-Crate Editors „NovaCrate“.

Die grundlegenden Konzepte von Research Objects sowie die Spezifikation von RO-Crates [51] wurde eingeführt. Anschließend wurde der aktuelle Stand der Technik in Bezug auf RO-Crate Editoren aufgegriffen. Die beiden bestehenden Editoren wurden vorgestellt und ihre Vor- und Nachteile beschrieben.

Als Grundlage für die Entwicklung von NovaCrate wurde zunächst die RO-Crate Java Bibliothek „ro-crate-java“ [47] vorgestellt, die mithilfe einer REST-API [36] durch den Editor zur Manipulation des RO-Crates verwendet wurde. Außerdem wurden Next.js [53], React [19] und Radix UI [56] gezeigt, welche die Web-Basistechnologien des Editors darstellen.

Es wurden funktionale und nichtfunktionale Anforderungen definiert, an denen sich der Entwurf und die Implementierung orientierten sollten. Das Design des Editors wurde im Detail vorgestellt und begründet, wobei vor allem die Vorteile des Designs für die Benutzbarkeit herausgehoben wurden. Auch die technische Architektur wurde im Detail betrachtet, um einige nichtfunktionale Anforderungen, wie die Wartbarkeit, Erweiterbarkeit und Portierbarkeit, zu begründen.

Die Implementierung einiger interessanter Komponenten wurde vorgestellt. So konnte ein Einblick in die Funktionsweise des Editors gewonnen werden, also auch wie jene Funktionen sich in eine React App integrieren lassen. Auch hier wurde wieder auf einige nichtfunktionale Anforderungen eingegangen.

Um die restlichen nichtfunktionalen Anforderungen sowie die funktionalen Anforderungen zu prüfen, wurde der Editor durch einen Nutzertest nach Krug [12] evaluiert. Fünf Testpersonen wurden eingeladen, um einige Aufgaben mit dem Editor zu bewältigen.

Die Diskussion befasste sich schließlich mit der Auswertung der Transkripte aus der Evaluation. Diese fiel vor allem positiv aus, da viele der Aufgaben aus dem Nutzertest ohne größere Probleme bewältigt werden konnten. Etwaige Probleme wurden genauer betrachtet und mögliche Lösungen vorgestellt. NovaCrate konnte fast alle Anforderungen erfüllen und kann mit nur geringen weiteren Aufwand auf einen produktiven Standard gebracht werden. Außerdem wurde diskutiert, wie NovaCrate in Zukunft produktiv eingesetzt werden könnte.

Das Ziel der Arbeit, die Entwicklung eines einfachen und intuitiven Editors für Research Object Crates, konnte erfüllt werden. NovaCrate ist ein benutzerfreundlicher Editor der dank seiner robusten und modularen Architektur zu einem wichtigen neuen Werkzeug für die Bearbeitung und Erstellung von RO-Crates werden könnte.

8.1. Ausblick

Wie bereits in der Diskussion erwähnt müssen die im Nutzertest gefundenen Probleme noch behoben werden. Außerdem muss eine produktive Version verfügbar gemacht werden, entweder durch zentrales Hosting der Seite oder einer ausführbaren Datei.

NovaCrate bietet an vielen Stellen noch Potenzial für weitere Funktionen, die es den Nutzer*innen möglichst einfach machen könnte ein gutes RO-Crate zu erstellen. Diese werden in den folgenden Unterabschnitten kurz erläutert.

Aber auch ein ausführlicher, eventuell auch vergleichender Nutzertest, könnte helfen die Benutzbarkeit und den Funktionsumfang von NovaCrate zu verbessern. Dieser könnte auch die bestehenden Editoren evaluieren.

8.1.1. Validierung von RO-Crates

Eine besonders interessante Funktion, die aktuell noch nicht in NovaCrate integriert ist, ist die Validierung des Crates. Inspiriert von klassischen Entwicklungsumgebungen könnte NovaCrate Fehler im Crate suchen und den Nutzer*innen dabei helfen, diese zu beheben.

Gleichzeitig stellt sich die Frage, ob ein solches Feature nicht auch für die RO-Crate Java Bibliothek [47] sinnvoll wäre. Somit könnten alle Verwender der Bibliothek von der Validierung profitieren. Außerdem die Validierung auf den Server ausgelagert werden, um im Browser wertvolle Ressourcen zu sparen.

8.1.2. RO-Crate Profile

Describo [13] und Crate-O [37] enthalten Funktionen zum Anwenden von sogenannten Profilen (vgl. Kapitel 3). Diese schneiden die Editoren so zu, dass Sie nur noch bestimmte Entitäten und Eigenschaften anzeigen. Dies soll die Arbeit mit den Editoren für bestimmte Zwecke beschleunigen. Eine Integration der Describo-Profile sollte gut möglich sein, und würde Nutzer*innen von Describo einen Umstieg auf NovaCrate deutlich erleichtern.

Im Entwurf der RO-Crate Spezifikation v1.2 ist ein ähnlicher Mechanismus für nicht-proprietäre Profile enthalten [50]. Eine entsprechende Funktion müsste ohnehin in NovaCrate integriert werden, sollte der Editor die Spezifikation v1.2 unterstützen wollen.

Literatur

[1] Sean Bechhofer u. a. „Research Objects: Towards Exchange and Reuse of Digital Knowledge“. Englisch. In: *Nature Precedings* (Juli 2010). ISSN: 1756-0357. DOI: 10.1038/npre.2010.4626.1. URL: https://www.nature.com/articles/npre.2010.4626.1 (besucht am 18. 07. 2024).

[2] Nigel Beval, Jurek Kirakowski und Jonathan Maissel. „What is usability?“ Englisch. In: *Proceedings of the 4th International Conference on HCI*. Stuttgart, Apr. 1994, S. 527–531. ISBN: 978-0-429-18054-5. DOI: 10.1201/9781482272574-76. URL: https://www.taylorfrancis.com/books/9781482272574/chapters/10.1201/9781482272574-76 (besucht am 12. 06. 2024).

[3] Broadcom. *Spring Homepage*. Englisch. 2005. URL: https://spring.io/ (besucht am 30. 07. 2024).

[4] California Digital Library, Crossref und DataCite. *Research Organization Registry (ROR)*. Englisch. 2024. URL: https://ror.org/ (besucht am 13. 07. 2024).

[5] John Deacon. „Model-View-Controller (MVC) Architecture“. Englisch. In: 4 (2009), S. 6.

[6] Happy Scribe Ltd. *Happy Scribe: Audio Transcription & Video Subtitles*. Englisch. 2024. URL: https://www.happyscribe.com/ (besucht am 31. 07. 2024).

[7] Jamstack. *Community Survey Results 2022*. Englisch. 2022. URL: https://jamstack.org/survey/2022/ (besucht am 04. 08. 2024).

[8] Arshad Javeed. „Performance Optimization Techniques for ReactJS“. Englisch. In: *2019 IEEE International Conference on Electrical, Computer and Communication Technologies (ICECCT)*. Feb. 2019, S. 1–5. DOI: 10.1109/ICECCT.2019.8869134. URL: https://ieeexplore.ieee.org/document/8869134/?arnumber=8869134 (besucht am 12. 07. 2024).

[9] JetBrains. *JavaScript Programming - The State of Developer Ecosystem in 2023 Infographic*. Englisch. 2023. URL: https://www.jetbrains.com/lp/devecosystem-2023 (besucht am 04. 08. 2024).

[10] Iida Kainu. „Optimization in React.js“. Englisch. Bachelorarbeit. Tampere, Finnland: Tampere University, Feb. 2022. URL: https://trepo.tuni.fi/bitstream/handle/10024/140258/KainuIida.pdf?sequence=2 (besucht am 12. 07. 2024).

[11] Steve Krug. *Don't make me think, revisited: a common sense approach to Web usability*. Englisch. 3. Aufl. New Riders, Juli 2014. ISBN: 978-0-13-359725-7. URL: https://www.pearson.de/dont-make-me-think-revisited-a-common-sense-approach-to-web-usability-9780133597257 (besucht am 09. 06. 2024).

[12] Steve Krug. *Rocket surgery made easy: the do-it-yourself guide to finding and fixing usability problems*. Englisch. Voices that matter. Berkeley, Calif: New Riders, 2010. ISBN: 978-0-321-65729-9.

[13] Marco La Rosa. *Describo*. Juni 2024. URL: https://describo.github.io/desktop.html (besucht am 01. 06. 2024).

[14] Marco La Rosa. *Describo Crate Builder Component*. Englisch. Juni 2024. URL: https://describo.github.io/docs/component/introduction.html (besucht am 10. 06. 2024).

[15] Marco La Rosa et al. *Describo Pricing*. Englisch. 2024. URL: https://describo.github.io/pricing.html (besucht am 03. 06. 2024).

[16] Marco La Rosa et al. *Describo Website*. Englisch. 2024. URL: https://describo.github.io/ (besucht am 03. 06. 2024).

[17] Meta Open Source. *Built-in React Hooks – React*. Englisch. 2024. URL: https://react.dev/reference/react/hooks (besucht am 13. 07. 2024).

[18] Meta Open Source. *Documentation for memo – React*. Englisch. URL: https://react.dev/reference/react/memo (besucht am 12. 07. 2024).

[19] Meta Open Source. *React Homepage*. Englisch. URL: https://react.dev/ (besucht am 26. 06. 2024).

[20] Meta Open Source. *Render and Commit – React*. Englisch. URL: https://react.dev/learn/render-and-commit (besucht am 12. 07. 2024).

[21] Meta Open Source. *Writing Markup with JSX in React*. Englisch. 2024. URL: https://react.dev/learn/writing-markup-with-jsx (besucht am 13. 07. 2024).

[22] Microsoft. *Der neue Microsoft Teams-Client*. Deutsch. Apr. 2024. URL: https://learn.microsoft.com/de-de/microsoftteams/platform/resources/teams-updates (besucht am 07. 08. 2024).

[23] Microsoft. *TypeScript Homepage*. Englisch. 2012. URL: https://www.typescriptlang.org/ (besucht am 19. 07. 2024).

[24] Microsoft Open Source. *Monaco Editor*. original-date: 2016-06-07T16:56:31Z. 0.51.0, Sep. 2024. URL: https://github.com/microsoft/monaco-editor (besucht am 03. 09. 2024).

[25] Mozilla. *Web Workers API Reference*. Englisch. Juni 2024. URL: https://developer.mozilla.org/en-US/docs/Web/API/Web_Workers_API (besucht am 05. 07. 2024).

[26] J. Nielsen. „Iterative user-interface design“. Englisch. In: *Computer* 26.11 (Nov. 1993), S. 32–41. ISSN: 0018-9162. DOI: 10.1109/2.241424. URL: http://ieeexplore.ieee.org/document/241424/ (besucht am 12. 06. 2024).

[27] npmjs.org. *keywords:react - npm search*. Englisch. URL: https://www.npmjs.com/search?ranking=optimal&q=keywords%3Areact&page=0&perPage=20 (besucht am 26. 06. 2024).

[28] Risto Ollila, Niko Mäkitalo und Tommi Mikkonen. „Modern Web Frameworks: A Comparison of Rendering Performance“. Englisch. In: *Journal of Web Engineering* (März 2022). ISSN: 1544-5976, 1540-9589. DOI: 10.13052/jwe1540-9589.21311. URL: https://journals.riverpublishers.com/index.php/JWE/article/view/7217 (besucht am 10. 06. 2024).

[29] OpenAI. *Whisper GitHub Repository*. original-date: 2022-09-16T20:02:54Z. Juli 2024. URL: https://github.com/openai/whisper (besucht am 31. 07. 2024).

[30] ORCID. *ORCID*. Englisch. 2024. URL: https://orcid.org/ (besucht am 13. 07. 2024).

[31] Vishal Patel. „Analyzing the Impact of Next.JS on Site Performance and SEO“. Englisch. In: *International Journal of Computer Applications Technology and Research* (Okt. 2023). ISSN: 23198656. DOI: 10.7753/IJCATR1210.1004. URL: https://ijcat.com/archieve/volume12/issue10/ijcatr12101004.pdf (besucht am 26. 06. 2024).

[32] Chris Pettitt. *dagre GitHub Repository*. original-date: 2012-09-27T04:39:56Z. Juli 2024. URL: https://github.com/dagrejs/dagre (besucht am 31. 07. 2024).

[33] pmndrs. *Zustand GitHub Repository*. original-date: 2019-04-09T09:10:06Z. Juli 2024. URL: https://github.com/pmndrs/zustand (besucht am 04. 07. 2024).

[34] Brandon Satrom. „Choosing the Right JavaScript Framework for Your Next Web Application“. Englisch. In: (2018).

[35] Schema.org. *Schema.org Website*. Englisch. Juli 2024. URL: https://schema.org/ (besucht am 31. 07. 2024).

[36] Jonas Scholz. *ro-crate-rest GitHub Repository*. 2024. URL: https://github.com/kit-data-manager/ro-crate-rest (besucht am 05. 07. 2024).

[37] Alvin Sebastian und Moises Sacal. *Crate-O*. Jan. 2024. URL: https://language-research-technology.github.io/crate-o/#/ (besucht am 09. 06. 2024).

[38] Alvin Sebastian und Moises Sacal. *GitHub Repository of Crate-O*. Englisch. Juni 2024. URL: https://github.com/Language-Research-Technology/crate-o (besucht am 09. 06. 2024).

[39] shadcn. *shadcn/ui*. Englisch. URL: https://ui.shadcn.com (besucht am 26. 06. 2024).

[40] Ben Shneiderman u. a. *Designing the user interface: strategies for effective human-computer interaction*. Englisch. Sixth Edition. Boston: Pearson, 2017. ISBN: 978-0-13-438038-4.

[41] Stian Soiland-Reyes u. a. „Packaging research artefacts with RO-Crate“. Englisch. In: *Data Science* 5.2 (Juli 2022). Hrsg. von Silvio Peroni, S. 97–138. ISSN: 24518492, 24518484. DOI: 10.3233/DS-210053. URL: https://www.medra.org/servlet/aliasResolver?alias=iospress&doi=10.3233/DS-210053 (besucht am 10. 06. 2024).

[42] Stian Soiland-Reyes u. a. „Practical webby FDOs with RO-Crate and FAIR Signposting“. Englisch. In: *Open Conference Proceedings: International FAIR Digital Objects Implementation Summit 2024* Vol. 4 (Apr. 2024). URL: https://pure.manchester.ac.uk/ws/portalfiles/portal/300255290/TIB_FDO2024_signposting_ro_crate-11.pdf.

[43] Stack Overflow. *Developer Survey 2023*. Englisch. 2023. URL: https://survey.stackoverflow.co/2023/#section-admired-and-desired-web-frameworks-and-technologies (besucht am 04. 08. 2024).

[44] Tauri Contributors. *Build smaller, faster, and more secure desktop applications with a web frontend*. Englisch. 2024. URL: https://tauri.app/ (besucht am 07. 08. 2024).

[45] Dai Hai Ton That u. a. *Sciunits: Reusable Research Objects*. Englisch. arXiv:1707.05731 [cs]. Sep. 2017. URL: http://arxiv.org/abs/1707.05731 (besucht am 18. 07. 2024).

[46] Jenifer Tidwell, Charles Brewer und Aynne Valencia-Brooks. *Designing Interfaces: Patterns for Effective Interaction Design*. Englisch. 3. Aufl. O'Reilly Media, Jan. 2020. ISBN: 978-1-4920-5196-1. URL: https://www.oreilly.com/library/view/designing-interfaces-3rd/9781492051954/ (besucht am 10. 02. 2024).

[47] Nikola Tzochev und Andreas Pfeil. *ro-crate-java GitHub Repository*. original-date: 2022-05-16T10:10:50Z. 2024. URL: https://github.com/kit-data-manager/ro-crate-java (besucht am 05. 07. 2024).

[48] University of Manchester. *Research Object Overview Website*. Englisch. 2024. URL: https://www.researchobject.org/overview/ (besucht am 18. 07. 2024).

[49] University of Technology Sydney, The University of Manchester UK and RO-Crate contributors. *RO-Crate Metadata | Specification 1.1*. Englisch. 2021. URL: https://www.researchobject.org/ro-crate/specification/1.1/metadata.html (besucht am 30. 07. 2024).

[50] University of Technology Sydney, The University of Manchester UK and RO-Crate contributors. *RO-Crate Profiles | Specification 1.2*. Englisch. 2024. URL: https://www.researchobject.org/ro-crate/specification/1.2-DRAFT/profiles.html (besucht am 07. 08. 2024).

[51] University of Technology Sydney, The University of Manchester UK and RO-Crate contributors. *RO-Crate Specification 1.1*. Englisch. 2021. URL: https://www.researchobject.org/ro-crate/specification/1.1/ (besucht am 30. 07. 2024).

[52] University of Technology Sydney, The University of Manchester UK and RO-Crate contributors. *RO-Crate Structure | Specification 1.1*. Englisch. 2021. URL: https://www.researchobject.org/ro-crate/specification/1.1/structure.html (besucht am 30. 07. 2024).

[53] Vercel. *Next.js by Vercel - The React Framework*. Englisch. 2024. URL: https://nextjs.org/ (besucht am 26. 06. 2024).

[54] Vercel. *Pages and Layouts in Next.js*. Englisch. 2024. URL: https://nextjs.org/docs/app/building-your-application/routing/pages-and-layouts (besucht am 13. 07. 2024).

[55] W3C JSON-LD Working Group. *JSON-LD - JSON for Linking Data*. Englisch. 2024. URL: https://json-ld.org/ (besucht am 31. 07. 2024).

[56] WorkOS. *Radix UI*. Englisch. URL: https://radix-ui.com/ (besucht am 26. 06. 2024).

[57] xyflow. *Node-Based UIs in React – React Flow*. Englisch. Juni 2024. URL: https://reactflow.dev/ (besucht am 26. 06. 2024).

[58] Evan You. *Vue.js*. Englisch. 2024. URL: https://vuejs.org/ (besucht am 13. 07. 2024).

A. Anhang

A.1. Skript des Nutzertests

Abgewandelt und übersetzt nach der Vorlage aus [12].

A.1.1. Einleitung

- ☐ Webserver für NovaCrate läuft.
- ☐ REST-API läuft.
- ☐ Ein Browser mit einer neutralen Startseite (Google) ist geöffnet.
- ☐ Beispiel-Crate und „Neuer Forschungsbericht.pdf“ liegen auf dem Schreibtisch.

Hallo, `Name`. Mein Name ist Christopher und ich werde dich heute durch diesen Usability Test leiten. Bevor wir Anfangen habe ich noch ein paar wichtige Informationen, die ich von meinem Zettel ablesen werde, um nichts zu vergessen.

Du weißt bestimmt schon ungefähr, worum es jetzt gehen soll, ich werde es trotzdem noch einmal erläutern. Im Rahmen meiner Bachelorarbeit habe ich einen Editor entwickelt, der heute durch einige Usability Tests evaluiert werden soll. Unsere Sitzung wird etwa eine Stunde dauern.

Als allererstes möchte ich dafür klarstellen, das heute nicht du, sondern der Editor getestet wird. Du kannst also keine Fehler machen.

Ich bitte dich während der Verwendung des Editors so laut wie möglich zu denken. Sag mir, was du siehst, was du denkst, was es tut, was du versuchst zu tun und was du denkst. Mach dir keine Sorgen meine Gefühle zu verletzen. Dein Feedback wird eine große Hilfe sein, um die Benutzbarkeit des Editors zu bewerten.

Falls du im Laufe der Aufgaben irgendwelche Fragen hast, kannst du mich einfach fragen. Ich kann dir vielleicht nicht direkt helfen, weil ich den Test nicht verfälschen will. Schließlich hat der letztendliche Nutzer auch nicht den Programmierer neben sich. Falls du am Ende noch Fragen hast, werde ich gerne noch weiter darauf eingehen. Falls du zwischendrin eine Pause brauchst, sag einfach Bescheid.

Das Mikrofon ist dir vielleicht auch schon aufgefallen. Mit deiner Erlaubnis werde ich den Bildschirm und unsere Konversation aufzeichnen. Das würde mir sehr bei der Evaluation

des Usability Tests helfen, und niemand außer mir wir die Aufnahmen sehen. So muss ich nicht alles mühselig mitschreiben.

Ich würde dich nun bitten dieses Formular zu unterschreiben, was mir erlaubt deine Stimme und den Bildschirm aufzuzeichnen, um die Evaluation meines Tools durchführen zu können.

☐ Genehmigungsformular aushändigen. ☐ Aufnahme des Bildschirms und der Stimme starten.

Gibt es bis hier hin schon Fragen?

A.1.2. Die Fragen

Bevor wir uns den Editor ansehen, habe ich noch ein paar kurze allgemeine Fragen an dich:

- Was hast du studiert oder gearbeitet, bevor du zu DEM gekommen bist?
- Bist du mit Research Object Crates vertraut? Hast du schonmal damit gearbeitet?

Falls die Probanden nicht mit Research Object Crates vertraut sind, folgt diese kurze Erklärung. Ansonsten geht es mit den Aufgaben weiter.

RO-Crates, oder auch Research Object Crates, sind eine Methode um beliebige Daten zusammen mit ihren Metadaten zu paketieren. Ein RO-Crate sieht dabei folgendermaßen aus (siehe Abbildung A.1).

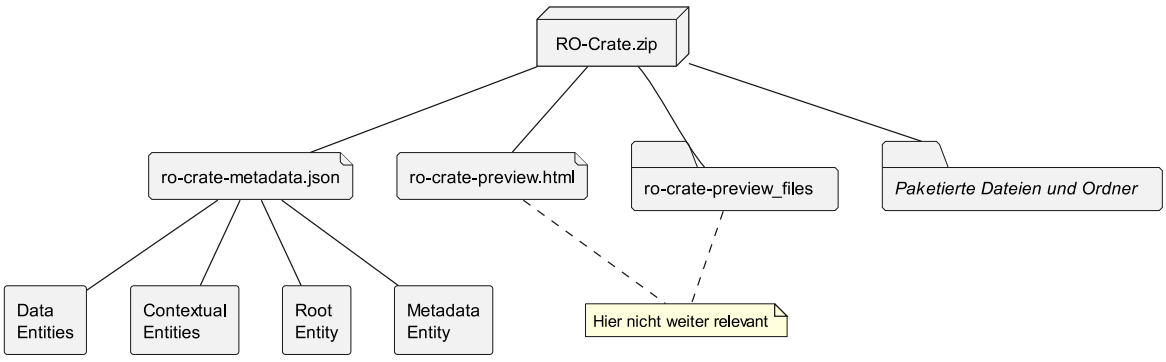

Abbildung A.1.: Aufbau eines Research Object Crates (Wiederholung von Abbildung 2.1)

Oben dargestellt ist das Crate selbst. Darin enthalten sind zu einem die Daten die durch Metadaten (wie Autor, Veröffentlichungsdatum, Kodierung …) genauer beschrieben werden sollen. Die Metadaten sind in einer JSON Datei gespeichert. Zusätzlich gibt es noch Preview-Dateien, die für uns nicht weiter relevant sind.

Zusammenfassend sind RO-Crates ein maschinenfreundliches Format zum Paketieren von Daten zusammen mit ihren Metadaten. Im heutigen Nutzertest befassen wir uns mit einem Editor, der RO-Crates erstellen und bearbeiten kann. In diesem Fall ist aber ausschließlich das Bearbeiten der Metadaten gemeint.

A.1.3. Die Aufgaben

Danke. Als Nächstes machen wir mit spezifischeren Aufgaben weiter. Ich werde dir jede Aufgabe vorlesen und auch als kleinen Zettel für dich hinlegen. Und nur als Erinnerung, sag mir bitte alles, was du denkst und was dir auffällt, negativ oder positiv. Danke fürs mitmachen!

- ☐ Erstes Szenario aushändigen und vorlesen.
- ☐ Nutzer die Aufgabe bewältigen lassen, bis er nicht mehr weiterkommt oder frustriert wird.
- ☐ Mit den anderen Aufgaben fortfahren bis die Zeit abgelaufen ist.

A.1.4. Schluss

Hast du noch irgendwelche Fragen?

- ☐ Aufzeichnung beenden und speichern.
- ☐ Danken und heraus begleiten.

A.2. Szenarien für den Nutzertest

Die folgenden Szenarien wurden den Testpersonen für jede Aufgabe auf einem separaten Zettel ausgehändigt und vorgelesen. Dieser dient der Orientierung mit Tabelle 6.1.

A.2.1. Importieren eines Crates

Du arbeitest als Teil einer Forschungsgruppe, die sich mit modernen Forschungsthemen befasst. Deine Aufgabe ist es, die Forschungsergebnisse deiner Gruppe zu veröffentlichen. Um die maschinelle Verarbeitung der Daten zu ermöglichen, entschied sich deine Forschungsgruppe, Research Object Crates zu verwenden. Die nächste Veröffentlichung von Forschungsdaten steht kurz bevor. Ein Kollege hat bereits den Großteil vorbereitet und ein Crate erstellt. Du sollst noch einige Details am Crate anpassen.

Um das Crate bearbeiten zu können, musst du es zunächst in den Editor für Research Object Crates „NovaCrate“ importieren. Verschaffe dir einen Überblick über die Startseite des Editors. Importiere anschließend das vorbereitete Crate. Du findest die ZIP-Datei auf dem Schreibtisch.

A.2.2. Ändern einer Eigenschaft

Das Crate enthält bereits den Großteil der nötigen Daten. Dazu gehören Forschungsberichte, die als Data Entities hinterlegt sind und Informationen zu Autoren und Organisationen, die als Contextual Entities hinterlegt sind. Verschaffe dir einen kurzen Überblick über den Editor. Du musst den Inhalt des Crates im Moment nicht weiter beachten.

Anschließend fällt dir auf, dass deinem Kollegen in der Data Entity `Example Image` einige Fehler unterlaufen sind. Korrigiere die Eigenschaften `Caption` und `Copyright Year`. Die richtigen Werte können aus dem zugehörigen Bild entnommen werden.

A.2.3. Hinzufügen einer Eigenschaft

Anschließend möchtest du zu den Data Entities „Lab Report (Part A)“ und „Lab Report (Part B)“ wichtige Informationen hinzufügen. Du sollst angeben, dass diese Dateien am KIT Campus Nord erstellt wurden. Eine passende Eigenschaft hierfür heißt `Location Created`.

Füge die Eigenschaft `Location Created` zu den Data Entities „Lab Report (Part A)“ und „Lab Report (Part B)“ hinzu. Setze sie auf KIT Campus Nord. Die Entity für KIT Campus Nord existiert bereits.

A.2.4. Löschen einer Eigenschaft

Dir fällt auf, dass in der Contextual Entity zu Björn Grüning private Daten hinterlegt sind. Vor der Veröffentlichung möchtest du diese natürlich löschen.

Entferne die Eigenschaft für das Geburtsdatum von der Entity Björn Grüning.

A.2.5. Hinzufügen einer Data Entity

Kurz bevor du dich in den wohl verdienten Feierabend verabschieden wolltest, kommt dein Vorgesetzter mit einem neuen Forschungsbericht. Er hat diesen in Form einer PDF-Datei auf deinem Schreibtisch abgelegt. Du sollst den neuen Forschungsbericht zum Crate hinzufügen.

Erstelle eine neue Entity für den Laborbericht. Die zugehörige Datei findest du als „Neuer Forschungsbericht.pdf“ auf dem Schreibtisch. Der richtige Typ für eine Entity, die eine Datei beschreibt, ist `File`.

A.2.6. Anpassen von Referenzen im Graph

Um die Korrektheit des Crates ein letztes Mal zu überprüfen, möchtest du dir im Graphen einen Überblick verschaffen. Dort fällt dir auf, dass du für den neu hochgeladenen Forschungsbericht noch keinen Autor hinterlegt hast.

Erstelle im Graphen eine neue Kante (entspricht einer Referenz), ausgehend vom neuen Forschungsbericht. Das Ende der Kante soll eine beliebige Person sein. Wähle `Author` als Eigenschaft für die Kante.

A.3. Transkripte der Nutzertests

Die Aufzeichnungen der Nutzertests wurden jeweils nach Unterzeichnung der Einverständniserklärung gestartet. Siehe Unterabschnitt A.1.2 für eine Zuordnung der getesteten Bestandteile des Editors.

A.3.1. Transkript eines Nutzertests (1)

00:00 **Christopher Raquet:** Erste Frage. Was hast du studiert oder gearbeitet, bevor du zu DEM gekommen bist?

00:04 **Proband*in:** Ich habe im Bachelor europäische Kultur und Ideengeschichte studiert und im Master Digital Humanities.

00:11 **Christopher Raquet:** Und Digital Humanities hat so ein Informatikteil drin?

00:14 **Proband*in:** Genau, da geht es im Prinzip darum, dass man computergestützte Methoden entwickelt und anwendet auf Probleme aus den Geisteswissenschaften.

00:25 **Christopher Raquet:** Okay, weil du hast ja auch in deiner Masterarbeit implementiert quasi?

00:29 **Proband*in:** Genau.

00:30 **Christopher Raquet:** Cool. Bist du mit Research Object Crates vertraut?

00:34 **Proband*in:** Nein.

00:35 **Christopher Raquet:** Dann hast du wahrscheinlich auch noch nicht damit gearbeitet.

00:37 **Proband*in:** Nein.

00:38 **Christopher Raquet:** Das erübrigt die nächste Frage. Gut, dann werde ich kurz einführen, was Research Object Creates sind. Und zwar ist das eine Methode, um beliebige Daten zusammen mit ihren Metadaten zu paketieren. Dafür habe ich ein Schaubild. Also das Paketieren heißt quasi eine ZIP-Datei draus machen. Sagt dir bestimmt was?

00:56 **Proband*in:** Ja.

00:57 **Christopher Raquet:** Und dann haben wir hier oben die Research Object Crate ZIP-Datei, die diese ganzen Dateien enthält. Das sind einmal die RO-Crate-Metadata.json. Die Datei enthält alle Metadaten. Und dann noch hier kursiv die paketierten Dateien und Ordner. Also alles, was man sonst noch in dieser ZIP-Datei hat. Und die ganzen Metadaten, die hier in dieser JSON-Datei drinstehen, die beschreiben dann die paketierten Daten.

01:21 **Proband*in:** Ja.

01:22 **Christopher Raquet:** Also geht es darum, Daten an einem Ort zu einem Paket zu schnüren und sie gleichzeitig noch mit allen Metadaten zu fassen.

01:28 **Proband*in:** Ja.

01:29 **Christopher Raquet:** Das sind dann zum einen die Data Entities. Quasi wie der Name schon sagt, eine Data Entity beschreibt eine Datei im Crate.

01:36 **Proband*in:** In dem, ja.

01:37 **Christopher Raquet:** Genau. Und die Contextual Entities sind dann quasi so weiterführende Informationen. Damit kann man dann einen Autor als eigenes Objekt beschreiben oder, was weiß ich, Veröffentlichungsdatum, Codierung von der Datei und so weiter.

01:50 **Proband*in:** Ja.

01:52 **Christopher Raquet:** Genau. Die Root Entity ist quasi die Data Entity für das ZIP selbst. Und die Metadata Entity ist eine Data Entity für die Metadaten-Datei selbst.

02:01 **Proband*in:** Ja.

02:02 **Christopher Raquet:** Und die Preview-File sind irrelevant für uns.

02:04 **Proband*in:** Okay.

02:05 **Christopher Raquet:** Genau. Also zusammenfassend sind RO-Crates ein Maschinenfreundliches Format zum Paketieren von Daten zusammen mit ihren Metadaten. Im heutigen Nutzer-Test befassen wir uns mit einem Editor, der Research-Object-Crates erstellen und bearbeiten kann. In diesem Fall ist aber ausschließlich das Bearbeiten der Metadaten gemeint.

02:22 **Proband*in:** Mhm.

02:23 **Christopher Raquet:** Gut. Als Nächstes machen wir mit spezifischen Aufgaben weiter. Ich werde dir jede Aufgabe vorlesen und auch als kleinen Zettel für dich hinlegen. Nur als Erinnerung sag mir bitte alles, was du denkst und was dir auffällt, positiv oder negativ. Danke fürs Mitmachen.

02:38 **Proband*in:** Gerne.

02:47 **Christopher Raquet:** Dann, allererste Aufgabe. Du arbeitest als Teil einer Forschungsgruppe, die sich mit modernen Forschungsthemen befasst. Deine Aufgabe ist es, die Forschungsergebnisse deiner Gruppe zu veröffentlichen. Um die maschinelle Verarbeitung der Daten zu ermöglichen, entschied sich deine Forschungsgruppe Research-Object-Crates zu verwenden. Die nächste Veröffentlichung von Forschungsdaten steht kurz bevor. Ein Kollege hat bereits den Großteil vorbereitet und ein Crate erstellt. Du solltest noch einige Details am Crate anpassen. Um das Crate bearbeiten zu können, musst du es zunächst in den Editor für Research-Object-Crates namens NovaCrate importieren. Verschaffe dir einen Überblick über die Startseite des Editors und importiere anschließend das vorbereitete Crate. Du findest die Zip-Datei auf dem Schreibtisch.

03:29 **Proband*in:** Okay.

03:31 **Christopher Raquet:** Genau, jetzt habe ich ... Das sind die falschen. Die sind nicht schön ausgeschnitten.

03:37 **Proband*in:** Okay.

03:39 **Christopher Raquet:** Ja.

03:41 **Proband*in:** Gut.

03:43 **Proband*in:** Also, ich sehe hier eine Startseite, NovaCrate. Das hast du ja schon gesagt, dass das ...

03:51 **Proband*in:** Ah, und hier steht, was ich machen kann, Start with Data oder Start from scratch.

03:58 **Proband*in:** Und ich nehme an, ich muss Start with Data machen.

04:01 **Proband*in:** Aber ich sehe hier auch auf der linken Seite Import existing Crate und ...

04:08 **Proband*in:** Ich muss ja was importieren. Also mache ich mal Import existing Crate.

04:15 **Proband*in:** So, auf dem Schreibtisch ist der Forschungsbericht.

04:20 **Christopher Raquet:** Genau.

04:21 **Proband*in:** Ist eine Zip-Datei. Dann würde ich die mal hochladen.

04:24 **Proband*in:** Ich weiß ja, dass RO-Crates irgendwie Zip-Dateien jetzt sind.

04:27 **Proband*in:** Dann lad ich das mal hoch. Und das ging sehr schnell. Das finde ich gut.

04:33 **Christopher Raquet:** Wunderbar, das war schon die erste Aufgabe.

04:35 **Proband*in:** Ohja, schön.

04:36 **Christopher Raquet:** Vielen Dank.

04:37 **Proband*in:** Gerne.

04:39 **Christopher Raquet:** Genau.

04:41 **Christopher Raquet:** Dann bevor wir ... wobei steht ja eigentlich auch drin. Ich lese einfach vor.

04:44 **Proband*in:** Ja.

04:45 **Christopher Raquet:** Gut. Das Crate enthält bereits den Großteil der nötigen Daten. Dazu gehören Forschungsberichte, die als Data Entities hinterlegt sind und Informationen zu Autoren und Organisationen, die als Contextual Entities hinterlegt sind. Verschaffe dir einen kurzen Überblick über den Editor. Du musst den Inhalt des Crates im Moment nicht weiter beachten. Vielleicht kannst du einfach kurz beschreiben, was du siehst.

05:05 **Proband*in:** Ja, also ich sehe hier einen recht aufgeschlüsselten und ausführlichen ... so eine Übersichtseite halt.

05:15 **Proband*in:** Und ...

05:18 **Proband*in:** ... kann das auch einklappen und so.

05:21 **Proband*in:** Das sieht mir ganz übersichtlich aus, eigentlich.

05:24 **Proband*in:** Es gibt hier den File Explorer.

05:27 **Proband*in:** Ah, das sind wahrscheinlich die Dateien, die in dieser ZIP Datei drin sind, nehme ich an.

05:33 **Proband*in:** Graph, oh, das ist cool.

05:35 **Proband*in:** Das zeigt das irgendwie ...

05:37 **Proband*in:** ... die Abhängigkeiten im Prinzip als Graphen, die Beziehungen.

05:41 **Proband*in:** Haben wir hier JSON Editor. Cool.

05:45 **Proband*in:** Okay, aber wichtig sind hier, glaube ich, jetzt erstmal diese Entities-Seite.

05:49 **Proband*in:** Property Overview.

05:52 **Proband*in:** Ah, okay. Also das hier ist im Prinzip die Struktur von ...

05:59 **Proband*in:** Ah ja, das ist die Struktur von der ausgewählten Datei im Prinzip immer.

06:04 **Christopher Raquet:** Mhm.

06:05 **Proband*in:** Genau. Und wir waren jetzt eben auf der obersten.

06:08 **Proband*in:** Das finde ich ein bisschen komisch, weil ich ...

06:11 **Proband*in:** ... ah aber hier steht ja, wo ich bin.

06:13 **Proband*in:** Ich habe zuerst gedacht, weil das hier nicht highlighted ist, aber hier steht es ja dann, das ist gut.

06:18 **Proband*in:** Hier oben sehe ich, was ich geöffnet habe.

06:20 **Proband*in:** Ja.

06:21 **Proband*in:** Und was soll ich jetzt nochmal machen?

06:23 **Christopher Raquet:** Habe ich nicht vorgelesen, aber das war schon mal sehr hilfreich. Vielen Dank.

06:26 **Christopher Raquet:** Anschließend fällt dir auf, also nachdem du den Überblick gemacht hast,

06:29 **Christopher Raquet:** Anschließend fällt dir auf, dass deinem Kollegen in der Data Entity Example Image einige Fehler unterlaufen sind. Korrigiere die Eigenschaften Caption

und Copyright Year. Die richtigen Werte können aus dem zugehörigen Bild entnommen werden.

06:44 **Proband*in:** Okay.

06:45 **Proband*in:** Die Caption habe ich hier schon gefunden, an der schönen Strukturübersichtssache.

06:51 **Proband*in:** „Caption for this Object, [...] Machine Format, Use Media Object, Indicator, Encoding."

06:57 **Proband*in:** Ah, okay. Und hier sehe ich schon, da ist was ganz Falsches geschrieben worden.

07:02 **Proband*in:** So.

07:03 **Proband*in:** Das zugehörige Bild.

07:08 **Proband*in:** Das zugehörige Bild.

07:11 **Proband*in:** Das ist das Example Image. Das wird ja hier irgendwo sein.

07:15 **Proband*in:** Da ist es.

07:18 **Proband*in:** File.

07:19 **Proband*in:** Nee.

07:20 **Proband*in:** Wo finde ich denn ein Bild?

07:25 **Proband*in:** Ja, dann gehe ich nochmal in den File Explorer. Vielleicht finde ich es hier.

07:32 **Proband*in:** Ja, da ist es. Okay.

07:35 **Proband*in:** Ich habe das Bild gefunden, im File Explorer. Das war jetzt gefühlt ein Zufallstreffer.

07:41 **Proband*in:** Aber hier: „the Caption for this Object".

07:46 **Proband*in:** Okay.

07:47 **Proband*in:** Die können dem zugehörigen Bild entnommen werden.

07:53 **Proband*in:** Da steht Labor Symbol Bild 2024 Imaginary.

07:58 **Proband*in:** Aber da steht ja schon Subtitle, Labor Symbol Bild.

08:03 **Proband*in:** Ich würde das ja einfach mal löschen.

08:08 **Proband*in:** Weil eigentlich...

08:11 **Christopher Raquet:** Das reicht schon. Das ist perfekt.

08:13 **Proband*in:** Okay.

08:14 **Proband*in:** Die richtigen Werte können... Okay, dann würde ich das löschen.

08:16 **Proband*in:** Und ich sehe hier oben, dass ich speichern kann.

08:19 **Proband*in:** Und ich bekomme ja auch eine Rückmeldung, dass gespeichert ist und damit würde ich sagen...

08:23 **Christopher Raquet:** Es fehlt noch der zweite Aufgabenteil quasi.

08:25 **Proband*in:** Ja, okay.

08:26 **Christopher Raquet:** Das Copyright Year.

08:27 **Proband*in:** Ah, das Copyright Year, stimmt.

08:29 **Proband*in:** Okay, das Copyright Year ist 2024, wenn ich das richtig sehe, aber ich gucke nochmal im Bild.

08:36 **Proband*in:** Ja, 2024.

08:38 **Proband*in:** Und das ist Data Entity Example Image. Also es geht auch wirklich um das Copyright Year von diesem Bild.

08:45 **Proband*in:** Also würde ich hier Copyright Year eintragen und speichern.

08:51 **Christopher Raquet:** Wunderbar. Das war's schon. Dankeschön.

08:55 **Christopher Raquet:** Genau, also du hattest jetzt Schwierigkeiten, das Bild zu finden.

08:59 **Proband*in:** Ja, das war mir... Weil ich habe mir gedacht, das Bild finde ich bestimmt auch hier.

09:03 **Proband*in:** Aber ich habe jetzt, also das habe ich nicht gefunden und deshalb habe ich mir gedacht, gucke ich beim File Explorer.

09:09 **Proband*in:** Ah, aber was ist hier?

09:11 **Proband*in:** Hint Data Entities. Oh, das ist cool mit den Hints.

09:15 **Proband*in:** Ja, okay.

09:17 **Christopher Raquet:** Es ist, glaube ich, in den künftigen Aufgaben nicht weiter relevant, deswegen zeige ich einfach, dass es hier versteckt gewesen ist.

09:25 **Proband*in:** Ah, Preview-File. Okay, aber das hätte ich sehen können.

09:28 **Christopher Raquet:** Aber es ist gut, dass du zeigst, dass es nicht offensichtlich zu sehen ist.

09:32 **Christopher Raquet:** Gut, machen wir weiter?

09:33 **Proband*in:** Mhm.

09:34 **Christopher Raquet:** Anschließend möchtest du zu den Data Entities „Lab Report Part A" und „Lab Report Part B" wichtige Informationen hinzufügen. Du sollst angeben, dass diese Dateien am KIT Campus Nord erstellt wurden. Eine passende Eigenschaft hierfür heißt Location Created.

09:50 **Christopher Raquet:** Füge die Eigenschaft Location Created zu den Data Entities „Lab Report Part A“ und „Lab Report Part B“ hinzu. Setze sie auf KIT Campus Nord. Die Entity für KIT Campus Nord existiert bereits.

10:03 **Proband*in:** Okay.

10:04 **Proband*in:** Ich habe hier oben schon gesehen, dass es „Add Property“ gibt. Also nehme ich an, das hilft mir jetzt.

10:09 **Proband*in:** Jetzt gucke ich mal hier.

10:11 **Proband*in:** Ah, und jetzt kann ich auch Preview-File.

10:13 **Proband*in:** Möchtest du downloaden? Und das breche ich mal ab.

10:17 **Christopher Raquet:** Mhm.

10:18 **Proband*in:** Ah, okay.

10:20 **Christopher Raquet:** Ja, Safari scheint die Dateien nicht zu previewen zu wollen, sondern direkt herunterladen zu wollen.

10:24 **Proband*in:** Ah, ja, gut. Ist ja nicht schlimm.

10:27 **Proband*in:** So, füge die Eigenschaft „Location Created“ hinzu.

10:31 **Proband*in:** Mach ich mal „Add Property“.

10:33 **Proband*in:** Und hier kann ich jetzt nämlich suchen und da habe ich schon „Location Created“.

10:37 **Proband*in:** So, das wurde jetzt hier hinzugefügt. Das sehe ich auch durch das Grüne. Das finde ich gut.

10:42 **Christopher Raquet:** Mhm.

10:46 **Proband*in:** Okay, „setze sie auf KIT Campus Nord“.

10:49 **Proband*in:** Jetzt muss ich hier nehme ich an, vielleicht Select, ja, Select KIT Campus Nord.

10:57 **Christopher Raquet:** Mhm.

10:58 **Proband*in:** Und damit ist das da gespeichert.

11:02 **Proband*in:** Na, und das stand ja auch in der Aufgabe, dass die Entity bereits existiert. Und jetzt speichere ich das.

11:08 **Proband*in:** Ich habe auch gesehen, da steht, dass man ungespeicherte Änderungen hat. Und das war irgendwie so hervorgehoben. Das hat man dann ja noch gesehen.

11:17 **Proband*in:** Das mache ich jetzt noch für Lab Report Part B.

11:21 **Proband*in:** Und dann mach ich wieder Add Property.

11:25 **Proband*in:** Location Created.

11:28 **Proband*in:** Select KIT Campus Nord.

11:32 **Proband*in:** Ja, und das ist wieder so schön.

11:34 **Proband*in:** So, saved.

11:36 **Christopher Raquet:** Wunderbar, vielen Dank.

11:38 **Christopher Raquet:** Also, als du das jetzt das zweite Mal gemacht hast, also bei Lab Report Part B, musstest du da wieder viel darüber nachdenken oder war es jetzt eigentlich klar?

11:45 **Proband*in:** Das war mir ganz klar eigentlich, dass das hat genauso funktioniert.

11:49 **Christopher Raquet:** Wunderbar.

11:51 **Christopher Raquet:** Perfekt.

11:52 **Christopher Raquet:** Dann, nächste. Zeit ist sehr gut.

11:56 **Christopher Raquet:** Dir fällt auf, dass du in der Contextual Entity zu Björn Grüning,

12:00 **Christopher Raquet:** Ah, ne, nicht du, aber dein Kollege, dir fällt auf, dass in der Contextual Entity zu Björn Grüning private Daten hinterlegt wurden. Vor der Veröffentlichung möchtest du diese natürlich löschen. Entferne die Eigenschaft für das Geburtsdatum von der Entity Björn Grüning.

12:14 **Proband*in:** Okay.

12:16 **Proband*in:** Gut, dann gehe ich mal auf Björn Grüning und ich sehe hier Birth Date.

12:21 **Proband*in:** Gut, das will ich weghaben.

12:23 **Proband*in:** Ah ja, ich habe jetzt hier auf die drei Punkte geklickt, weil ich das kennen als Optionen-Taste und hier steht dann direkt „delete entry“, das ist auch rot hervorgehoben, das erkenne ich direkt als Deleting und jetzt habe ich es Deleted.

12:39 **Proband*in:** So, jetzt steht hier aber noch Birth Date.

12:46 **Proband*in:** „Empty property“. Ich weiß jetzt nicht genau, ich speichere das erst mal.

12:53 **Proband*in:** Ah, und jetzt ist es weg.

12:55 **Proband*in:** Okay, ich wusste jetzt nicht genau, ob ich nochmal das Birth Date entfernen muss, nachdem ich den Wert entfernt habe, weil das nicht direkt weg war.

13:04 **Proband*in:** Aber als ich dann noch speichern geklickt habe, ist es direkt weggegangen und es war auch so rot markiert, vielleicht sollte das mir ein Hinweis gewesen sein, aber das war mir jetzt nicht ganz klar.

13:16 **Proband*in:** Wieso das nicht direkt weggegangen ist, aber jetzt ist es ja weg.

13:19 **Christopher Raquet:** Alles klar.

13:21 **Christopher Raquet:** Wunderbar.

13:22 **Christopher Raquet:** Dann, ich weiß gar nicht, wie ich jetzt noch auf mehr Zeit kommen soll.

13:26 **Proband*in:** Ich habe auch viel zu lange eingeplant bei meinem Test.

13:30 **Christopher Raquet:** Okay, gut.

13:32 **Christopher Raquet:** Dann die fünfte Aufgabe.

13:34 **Christopher Raquet:** Kurz bevor du dich in den wohlverdienten Feierabend verabschieden wolltest, kommt dein Vorgesetzter mit einem neuen Forschungsbericht. Er hat diesen in Form einer PDF-Datei auf deinem Schreibtisch abgelegt. Du sollst den neuen Forschungsbericht zum Crate hinzufügen. Erstelle eine neue Entity für den Laborbericht. Forschungsbericht meine ich. Die zugehörige Datei findest du als neuer Forschungsbericht.pdf auf dem Schreibtisch. Der richtige Typ für eine Entity, die einen Datei beschreibt, ist File.

14:01 **Proband*in:** Okay.

14:02 **Proband*in:** Also, ich sehe hier oben Add New Entity.

14:06 **Proband*in:** Und hier ist schon ein Forschungsbericht, aber ich will jetzt ja einen neuen ...

14:10 **Proband*in:** Erstelle eine neue Entity.

14:12 **Proband*in:** Okay, ich mache mal Add New Entity.

14:14 **Proband*in:** So, „select the type of the Entity you want to create."

14:18 **Proband*in:** Das ist „File". Das hätte ich jetzt mir auch so gedacht.

14:23 **Proband*in:** Dann gehe ich mal auf File.

14:26 **Proband*in:** Und jetzt bin ich schon verwirrt, weil okay, add a file to the Crate.

14:33 **Proband*in:** „Use the File-Explorer to upload the file to the specific folder."

14:38 **Proband*in:** Weil hier steht jetzt Upload File und hier steht Select File.

14:43 **Proband*in:** Und ich klicke jetzt mal auf Upload File.

14:47 **Proband*in:** Und da passiert nichts.

14:49 **Proband*in:** Dann gehe ich mal auf Select File und da passiert etwas, das ist schon mal gut.

14:54 **Proband*in:** So, neuer Forschungsbericht.

14:58 **Proband*in:** Jetzt habe ich das hochgeladen, ich gehe jetzt nochmal auf Upload File.

15:02 **Proband*in:** Vielleicht lädt der dann...

15:05 **Proband*in:** „Use the File-Explorer“, okay, das ist das.

15:08 **Proband*in:** Das heißt schon mal nicht File-Explorer, das ist irgendwie...

15:11 **Proband*in:** Wobei, das beschreibt ja das File und damit kommt man zum File-Explorer.

15:15 **Proband*in:** Okay, „this will import the file into the Crate“.

15:18 **Proband*in:** Aber wofür das jetzt ist?

15:21 **Proband*in:** Create.

15:23 **Proband*in:** Ah nee, jetzt ist es aber nicht aufgetaucht.

15:25 **Proband*in:** Weil das müsste ja hier oben... Ah, das ist jetzt hier.

15:28 **Proband*in:** Aber muss das nicht hier oben sein?

15:30 **Christopher Raquet:** Nee, das ist richtig.

15:32 **Proband*in:** Ah, weil das hat mich jetzt verwirrt, weil das beides Forschungsbericht heißt.

15:36 **Proband*in:** Aber, das hier ist das Data Set, jetzt sehe ich's.

15:38 **Proband*in:** Und das da ist im Prinzip eine Datei in diesem...

15:41 **Proband*in:** Okay, gut.

15:43 **Christopher Raquet:** Perfekt, danke schön.

15:45 **Christopher Raquet:** Ich find's gut, dass du auch über dieses „Upload File“ gestolpert bist. Ich habe das gestern schon mit meiner Freundin ausprobiert und sie hatte genau das gleiche Problem.

15:52 **Christopher Raquet:** Aber tatsächlich sind es einfach nur zwei Tabs. Aber das...

15:57 **Proband*in:** Ah, Okay, das sieht nämlich aus, als wäre das ein Button, auf den ich drauf klicke, mit dem ich jetzt das Ding hochladen kann.

16:03 **Proband*in:** Vielleicht, das ist ja von der Form her, ist das auch ein Button.

16:07 **Christopher Raquet:** Ja, das muss man auf jeden Fall noch anpassen.

16:09 **Proband*in:** Weil wenn das ein Tab ist, dann würde ich das glaube ich verstehen.

16:13 **Christopher Raquet:** Wir können jetzt ja mal kurz abseits der Aufgaben ...

16:16 **Christopher Raquet:** Hier habe ich ja richtige Tabs benutzt.

16:18 **Proband*in:** Genau, sowas wäre gut.

16:19 **Christopher Raquet:** Genau, ja.

16:21 **Christopher Raquet:** Gut. Wunderbar.

16:24 **Christopher Raquet:** Dann, um die Korrektheit des Crates ein letztes Mal zu überprüfen, möchtest du dir im Graphen einen Überblick verschaffen. Dort fällt dir auf, dass du für den neu hochgeladenen Forschungsbericht noch keinen Autor hinterlegt hast. Erstelle im Graphen eine neue Kante, was einer Referenz entspricht, ausgehend vom neuen Forschungsbericht. Das Ende der Kante soll eine beliebige Person sein. Wähle Autor, beziehungsweise Author, als Eigenschaft für die Kante.

16:53 **Proband*in:** Okay, Graph hatte ich ja schon gesehen.

16:56 **Proband*in:** Das ist hier an der Seite, gibts die Option Graph. Und hier sehe ich das direkt.

17:03 **Proband*in:** Jetzt brauche ich meinen neu hochgeladenen Forschungsbericht, da ist er.

17:09 **Proband*in:** So, und ich sehe hier jetzt, ah, ich kann zoomen, das ist cool.

17:13 **Proband*in:** Aber das geht nur bis zum gewissen Punkt, was ja wahrscheinlich auch klug ist. Dann kommt noch nichts zu daran, so ... „ausgehend".

17:22 **Proband*in:** Und ich nehme an, um was Neues zu erstellen, muss ich hier auf das Plus klicken. Das geht nicht.

17:27 **Proband*in:** Ah, vielleicht ist das zum Ausklappen von Sachen.

17:30 **Proband*in:** Ja, gut, das verstehe ich nicht, wofür das Plus ist.

17:33 **Proband*in:** Aber ich sehe hier oben, dass ich „Add new Entity" machen kann

17:38 **Proband*in:** Aber das ist jetzt natürlich eine neue Kante.

17:43 **Proband*in:** Aber das ist jetzt natürlich, entspricht einer Referenz.

17:46 **Proband*in:** Ich versuche jetzt mal „Add new Entity".

17:52 **Christopher Raquet:** Das ist nicht ganz der richtige Weg.

17:54 **Proband*in:** Nein, ich guck mal weiter, dann mal.

17:57 **Proband*in:** „Save all entities", vielleicht kann ich ja hier, ah, hier gibt es irgendwas.

18:05 **Proband*in:** „Add property".

18:07 **Proband*in:** „Find references".

18:09 **Proband*in:** Okay, da bin ich irgendwie schon mal auf der richtigen Spur.

18:12 **Proband*in:** Ich mache mal Add property.

18:17 **Proband*in:** „Author".

18:18 **Proband*in:** Okay.

18:20 **Proband*in:** Also was mir jetzt hier, wenn das klappen sollte, verwirren würde, wäre das hier Referenz steht, aber da irgendwie property.

18:26 **Proband*in:** So, jetzt sehe ich aber, das hat einen Autor.

18:30 **Proband*in:** Und jetzt brauche ich aber noch so eine Person.

18:39 **Proband*in:** „Entity“.

18:50 **Proband*in:** Okay, wie mache ich das denn?

18:52 **Proband*in:** Open Entity Editor.

18:55 **Proband*in:** Okay, das habe ich jetzt mal gesehen. Das finde ich auch gut, dass man da so schnell ’rüberwechseln kann.

19:01 **Proband*in:** Das Ende der Kante soll eine beliebige Person sein.

19:04 **Proband*in:** Aber ich glaube, ich weiß jetzt halt nicht genau, was eine Kante ist.

19:08 **Proband*in:** Ob eine Kante, das ist ja eigentlich ein Node.

19:10 **Proband*in:** Ob das eine Kante schon ist, oder ob eine Kante irgendwie halt so ein neuer, so ein neues Rechteck ist.

19:19 **Proband*in:** Ich speicher das jetzt erstmal.

19:22 **Proband*in:** Das sind ja auch die Punkte, von denen das ausgehen sollte. Und das, was mich total verwirrt, ist dieses Plus, von dem auch ein Punkt ausgeht.

19:30 **Proband*in:** Ah, so kann ich. Ah, okay, aber damit könnte ich was verbinden jetzt.

19:39 **Proband*in:** Okay, das heißt, aber ich kann jetzt eine New Entity machen, was eine Person ist. Und dann könnte ich es verbinden.

19:49 **Proband*in:** „Create manually“ ...

19:54 **Proband*in:** Ich mach jetzt mal... Bernd Beliebig.

19:59 **Proband*in:** Create.

20:01 **Proband*in:** Wo ist er jetzt?

20:05 **Proband*in:** „Bernd Beliebig“, so. Das ist jetzt die beliebige Person. Und jetzt kann ich das verbinden. Und jetzt ist das der Autor.

20:15 **Proband*in:** Genau, jetzt habe ich einen Autor hinterlegt.

20:18 **Proband*in:** Und das Ende der Kante...

20:20 **Proband*in:** Ich weiß jetzt halt nicht so richtig, was die Kante ist.

20:24 **Christopher Raquet:** Es ist schon perfekt die Aufgabe, das ist genau richtig.

20:26 **Proband*in:** Okay, gut, dann ist es gut.

20:28 **Christopher Raquet:** Und das ist wahrscheinlich...

20:30 **Christopher Raquet:** Also mit Kanten sind letztendlich einfach nur die Kanten gemeint.

20:33 **Proband*in:** Ah, okay, das heißt aber, das ist schon richtig, dass das Ende der Kante eine Person wird.

20:38 **Christopher Raquet:** Genau, richtig.

20:39 **Proband*in:** Gut.

20:40 **Proband*in:** Also was, was ich total verwirrend fand, war dieses Plus, weil ich gedacht habe, ich kann irgendwas mit dem Plus hinzufügen, aber jetzt, wo ich gesehen habe, dass man das hier machen kann, denke ich mir, dass das irgendwie einfach ein Weg ist, um eine neue Verbindung herzustellen.

20:56 **Proband*in:** Ah, und dann kann ich auswählen, was ich machen will.

21:00 **Christopher Raquet:** Genau.

21:01 **Proband*in:** Okay, jetzt habe ich das, dann ist es cool eigentlich.

21:04 **Christopher Raquet:** Sehr gut. Aber es war jetzt für dich nicht offensichtlich.

21:07 **Proband*in:** Nee, das fand ich relativ verwirrend. Und auch, dass …

21:12 **Proband*in:** Ich weiß schon wieder nicht mehr …

21:14 **Proband*in:** Ach so, man muss da drauf drücken und dann... dann Entity, okay. Das macht ja auch Sinn. Und dann „Add property“, gut.

21:24 **Proband*in:** Ja doch, jetzt speichere ich das nochmal und dann... Ja.

21:28 **Christopher Raquet:** Genau, wunderbar. Das wären schon alle Aufgaben gewesen.

21:36 **Christopher Raquet:** Ähm, genau. Hast du noch irgendwelche Fragen?

21:41 **Proband*in:** Nein, eigentlich nicht, aber ich finde es mega cool. Das finde ich sehr... Sieht sehr schlank aus so.

21:54 **Christopher Raquet:** Nice, freut mich.

A.3.2. Transkript eines Nutzertests (2)

00:10 **Christopher Raquet:** Bevor wir uns den Editor ansehen, habe ich noch ein paar kurz allgemeine Fragen an dich.

00:14 **Christopher Raquet:** Was hast du studiert oder gearbeitet, bevor du zu DEM gekommen bist?

00:18 **Proband*in:** Ich habe Mathe studiert.

00:20 **Christopher Raquet:** Und jetzt machst du bei DEM Mathethemen, oder?

00:23 **Proband*in:** Nein, ich bin in NFDI4Ing tätig.

00:27 **Christopher Raquet:** Was macht das so ungefähr?

00:28 **Proband*in:** Das ist die nationale Forschungs-Daten-Infrastruktur für die Ingenieurswissenschaften.

00:33 **Proband*in:** Da bin ich für die Base-Services Storage & Repositories verantwortlich. Ich koordiniere das.

00:41 **Proband*in:** Ich habe den Data-Collection Explorer aufgebaut. Das ist so ein Informationssystem für die Ingenieure.

00:49 **Christopher Raquet:** Also quasi implementiert und entwickelt und solche Sachen?

00:53 **Proband*in:** Ja, und da habe ich mich in der letzten Zeit vor allem mit Knowledge Graphen beschäftigt.

00:58 **Christopher Raquet:** Okay, cool.

01:00 **Christopher Raquet:** Bist du schon mit Research Object Crates vertraut?

01:04 **Proband*in:** Den Namen nach.

01:06 **Christopher Raquet:** Okay, also du hast wahrscheinlich noch nicht damit gearbeitet.

01:10 **Proband*in:** Nein

01:10 **Christopher Raquet:** Dann werde ich kurz einführen, was das ist. Dafür habe ich das Schaubild dabei. Also das hier oben ist das Research Object Crate. Das ist einfach nur ein ZIP-Archiv.

01:24 **Christopher Raquet:** Und die Idee hinter RO Crates ist es jetzt beliebige Daten, also meistens Daten, die bei Forschung entstehen, in einem ZIP-Archiv zu paketieren, also zusammenzufassen, und dann zusätzlich noch mit allen nötigen oder allen bekannten Metadaten zu versehen.

01:41 **Christopher Raquet:** Das ist zum einen hier kursiv geschrieben, sind die ganzen paketierten Dateien und Ordner in dieser ZIP-Datei drin, zusätzlich noch diese RO-Crate-Metadata.json-Datei und diese Metadaten beschreiben eben alle Dateien, die in dieser ZIP-Datei archiviert sind quasi.

02:00 **Christopher Raquet:** Genau, da sind zum einen Data Entities enthalten, die Entities beschreiben jeweils eine Datei, deswegen Data Entities.

02:08 **Christopher Raquet:** Die Contextual Entities können weiterführende Objekte beschreiben, seien es Autoren oder Locations oder Organisationen, kann man quasi für alles.

02:20 **Proband*in:** Andere RO-Crates, auf die man referenzieren will.

02:23 **Christopher Raquet:** Das würde wahrscheinlich auch gehen. Das wäre vermutlich eher eine Data Entity, auch wenn es nicht im Crate enthalten ist, würde man externe Sachen auch per Data Entity beschreiben. Genau, das Ganze wird dann untereinander verlinkt über das JSON-LD-Prinzip, falls ihr das was sagt.

02:40 **Proband*in:** Nicht wirklich, ja.

02:42 **Christopher Raquet:** Das heißt, die ganzen Entities liegen in einem flachen Array drin und verweisen auch einander mit ihren IDs. Also nichts Wildes.

02:49 **Christopher Raquet:** Genau, Abseits dessen gibt es noch zwei besondere Entities, das ist die Root Entity. Das ist einfach quasi eine Data Entity für das Crate selbst, um sich selbst zu beschreiben.

02:59 **Christopher Raquet:** Und dann gibt es noch eine Metadata Entity, um die Metadaten-Datei zu beschreiben. Ist aber eher eine Formalie.

03:06 **Christopher Raquet:** Genau.

03:08 **Christopher Raquet:** Dann gibt es noch Preview-Files, die sind aber nicht relevant.

03:13 **Christopher Raquet:** Genau, so viel dazu.

03:15 **Christopher Raquet:** Zusammenfassend sind RO-Crates ein Maschinen-freundliches Format zum Pakettieren von Daten zusammen mit ihren Metadaten. Im heutigen Nutzer-Test befassen wir uns mit einem Editor, der RO-Crates erstellen und bearbeiten kann.

03:27 **Christopher Raquet:** In diesem Fall geht es aber ausschließlich um das Bearbeiten der Metadaten.

03:32 **Christopher Raquet:** Genau. Dazu schon fragen?

03:35 **Proband*in:** Nö.

03:37 **Christopher Raquet:** Als Nächstes machen wir mit spezifischeren Aufgaben weiter.

03:40 **Christopher Raquet:** Ich werde dir jede Aufgabe vorlesen und auch als kleinen Zettel für dich hinlegen.

03:44 **Christopher Raquet:** Und nur als Erinnerung sag mir bitte alles, was du denkst und was dir auffällt, negativ oder positiv. Danke fürs Mitmachen.

03:58 **Christopher Raquet:** Meine Zettel sind natürlich nicht richtig vorsortiert.

04:02 **Proband*in:** Alles gut.

04:04 **Christopher Raquet:** Los geht's.

04:06 **Christopher Raquet:** Du arbeitest als Teil einer Forschungsgruppe, die sich mit modernen Forschungsthemen befasst. Deine Aufgabe ist es, die Forschungsergebnisse deiner Gruppe zu veröffentlichen. Um die maschinelle Verarbeitung der Daten zu ermöglichen, entschied sich deine Forschungsgruppe Research Object Crates zu verwenden. Die nächste Veröffentlichung von Forschungsdaten steht kurz bevor. Ein Kollege hat bereits den Großteil vorbereitet und ein Crate erstellt. Du sollst noch einige Details am Crate anpassen.

04:32 **Christopher Raquet:** Um das Crate bearbeiten zu können, musst du es zunächst in den Editor für Research Object Crates Nova Crate importieren. Verschaffe dir einen Überblick über die Startseite des Editors. Importiere anschließend das vorbereitete Crate. Du findest die ZIP-Datei auf dem Schreibtisch.

04:48 **Proband*in:** Also vermutlich auf dem digitalen Schreibtisch.

04:56 **Christopher Raquet:** Genau, du kannst einfach mal anfangen zu beschreiben, was du siehst und was du denkst, was die ganzen Sachen machen.

05:01 **Proband*in:** Viel Schwarz.

05:03 **Christopher Raquet:** Wenn es dir lieber ist, kannst du auch den weißen Modus anschalten.

05:06 **Proband*in:** Nee, das ist okay, das passt so. Ich habe jetzt den Toggle-Dingens hier oben erwartet.

05:17 **Proband*in:** Also das Menü ist links.

05:24 **Proband*in:** Was halt jetzt hier auffällig ist, das ist da dieses „Start with Data" und das „Start from Scratch".

05:29 **Proband*in:** Ich soll diesen schon existierenden Crate importieren.

05:33 **Proband*in:** Das mache ich da. Mir ist es nicht sofort aufgefallen, dass da der Importmöglichkeit besteht.

05:42 **Proband*in:** Das wird der Forschungsbericht sein, oder?

05:45 **Christopher Raquet:** Genau.

05:48 **Christopher Raquet:** Wunderbar. Das war die erste Aufgabe.

05:55 **Christopher Raquet:** Dann, nächste.

05:57 **Christopher Raquet:** Das Crate enthält bereits den Großteil der nötigen Daten. Dazu gehören Forschungsberichte, die als Data Entities hinterlegt sind und Informationen zu Autoren und Organisationen, die als Contextual Entities hinterlegt sind.

06:09 **Christopher Raquet:** Verschaffe dir einen kurzen Überblick über den Editor. Du musst den Inhalt des Crates im Moment nicht weiter beachten.

06:15 **Christopher Raquet:** Anschließend fällt dir auf, dass deinem Kollegen in der Data Entity Example Image einige Fehler unterlaufen sind. Korrigiere die Eigenschaften Caption und Copyright Year. Die richtigen Werte können aus dem zugehörigen Bild entnommen werden.

06:29 **Christopher Raquet:** Aber zum Überblick kannst du gerne wieder wie bei der Seite vorher kurz beschreiben, was du siehst und was du denkst.

06:36 **Proband*in:** Was auffällig ist, ist, wir haben jetzt oben quer über die ganze Breite noch ein Menü.

06:42 **Proband*in:** Der Rest ist wieder auf der linken Seite.

06:48 **Proband*in:** Und dann, der Großteil vom Bildschirm wird eingenommen von Übersicht über den Inhalt vom Crate und eine Übersicht über die Eigenschaften von der Entität, die wir gerade anschauen.

07:04 **Proband*in:** So, das heißt, wir dürfen das Example Image ändern, die Caption.

07:15 **Proband*in:** Ah, okay. Moment.

07:19 **Proband*in:** Aus dem zugehörigen Bild. Das zugehörige Bild. Irgendwie kann ich das aufrufen.

07:32 **Proband*in:** Ah, da oben hat es ein Preview.

07:52 **Proband*in:** Das Textfeld kann ich nicht größer machen, oder?

07:57 **Christopher Raquet:** Ja, aber du kannst insgesamt ran zoomen, wenn du möchtest.

08:01 **Proband*in:** Nein, mir ging es eigentlich eher darum, dass ich den, dass ich Übersicht habe. Weil der Text ja nicht auf die, der ist ja abgeschnitten, weil er nicht auf die Zeile passt.

08:13 **Proband*in:** Und ich wollte jetzt einfach den, es gibt hier die Möglichkeit, dass man in so Webinterfaces die Textfelder in der Größe anpasst.

08:23 **Proband*in:** Da war ich der Ansicht, dass das jetzt da geht, weil halt der Text zu lang ist.

08:29 **Proband*in:** „An Image showing some... Flasks in der Laboratory“. Mhm.

08:41 **Christopher Raquet:** Aber das genügt schon für den Teil.

08:44 **Proband*in:** So, und das Copyright hier ist 2024.

08:50 **Christopher Raquet:** Wunderbar. Perfekt.

08:53 **Proband*in:** So, und dann muss ich das Ganze noch speichern.

08:55 **Christopher Raquet:** Sehr gut. Dankeschön.

08:58 **Proband*in:** Das war jetzt ein bisschen unerwartet, dass das schwarz ist. Aber gut.

09:04 **Christopher Raquet:** Was hättest du jetzt erwartet?

09:06 **Proband*in:** Ich hätte jetzt eigentlich eher sowas wie rot oder orange erwartet.

09:09 **Christopher Raquet:** Mhm.

09:11 **Proband*in:** Was ich cool finde ist, dass sich da die Farbe geändert hat, weil man dann sieht, dass es abgeschlossen ist. Mhm.

09:18 **Christopher Raquet:** Cool. Gut, dann.

09:22 **Christopher Raquet:** Anschließend möchtest du zu den Data Entities Lab Report Part A und Lab Report Part B wichtige Informationen hinzufügen. Du solltest angeben, dass diese Dateien am KIT Campus Nord erstellt wurden. Eine passende Eigenschaft hierfür heißt Location Created.

09:38 **Christopher Raquet:** Füge die Eigenschaft Location Created zu den Data Entities Lab Report Part A und Lab Report Part B hinzu. Setze sie auf KIT Campus Nord. Die Entity für KIT Campus Nord existiert bereits.

09:51 **Proband*in:** Also da haben wir den KIT Campus Nord, das ist diese Contextual Entity.

09:58 **Proband*in:** So, Lab Report Part A. Links oben drüber hat es wieder das Menü, das Wichtig ist für uns, so und jetzt haben wir da die Location.

10:13 **Proband*in:** Dann mache ich ein Add Property. Suche nach Location Created. Und dann was?

10:23 **Proband*in:** Dann KIT Campus Nord ist schon angelegt, wähle ich aus.

10:31 **Proband*in:** Sag ich Speichern.

10:37 **Proband*in:** Und dann machen wir das gleiche nochmal. Location Created. Select.

10:50 **Christopher Raquet:** Wunderbar.

10:52 **Proband*in:** Und save.

10:54 **Christopher Raquet:** Du bist zu schnell. Aber sehr gut.

10:58 **Proband*in:** Wenn man sich daran gewöhnt hat, dass oben quer über die ganze Breite, das globale Menü für die Eigenschaft ist, oder für den Teil, an dem man arbeitet und links aufgelistet ist, also die Einzelheiten aufgelistet sind, das wiederholt sich dann in dem kleineren Teil, wo ich die Entitäten habe.

11:27 **Proband*in:** Und innerhalb dieses Entitäten-Previews ist es ja wieder ähnlich aufgebaut. Ich habe dann links die Sachen, mit denen ich arbeite und oben quer wieder das Menü.

11:41 **Proband*in:** Und wenn man das mal herausgekriegt hat, dann ist es eigentlich, was ein bisschen verwirrend ist, ist, dass man da unten das Add Property über die ganze Breite hat und oben auch.

11:53 **Christopher Raquet:** Ja, aber würdest du insgesamt sagen, dass du gut zurechtkommst mit der Struktur?

11:57 **Proband*in:** Bis jetzt schon, ja.

11:59 **Proband*in:** Also, ich habe jetzt nicht wirklich damit gearbeitet, aber dafür, dass ich noch gar keine Ahnung hatte, wie das Ding funktioniert, ist ja absolut kein Problem.

12:11 **Christopher Raquet:** Perfekt.

12:13 **Christopher Raquet:** Dir fällt auf, dass in der Contextual Entity zu Björn Grüning private Daten hinterlegt sind. Vor der Veröffentlichung möchte ich es zu dieser natürlich löschen. Entferne die Eigenschaft für das Geburtsdatum von der Entity Björn Grüning.

12:27 **Proband*in:** Also, hier sehe ich links, wähle ich aus und dann habe ich da das Geburtsdatum, das soll weg.

12:39 **Proband*in:** Das bekannte 3-Punkte-Menü. Ah, cool.

12:45 **Proband*in:** Das finde ich klasse, wenn ich damit der Maus nicht darauf bin und in dem Submenü, dass das Delete Entry rot hinterlegt ist.

12:53 **Proband*in:** Und das ist es.

12:56 **Christopher Raquet:** Wunderbar. Dankeschön.

13:00 **Christopher Raquet:** Kurz bevor du dich in den wohlverdienten Feierabend verabschieden wolltest, kommt dein Vorgesetzter mit einem neuen Forschungsbericht. Er hat diesen in Form einer PDF-Datei auf deinem Schreibtisch abgelegt. Du sollst den neuen Forschungsbericht zum Crate hinzufügen.

13:16 **Christopher Raquet:** Erstelle eine neue Entity für den Forschungsbericht. Die zugehörige Datei findest du als ein neuer Forschungsbericht vom PDF auf deinem Schreibtisch. Der richtige Typ für eine Entity, die eine Datei beschreibt, ist File.

13:30 **Proband*in:** Jetzt räumen wir da erst mal da oben auf, dass das alles ein bisschen übersichtlich ist.

13:39 **Proband*in:** Das heißt, ich mache hier ein Add New Entity und dann darf ich einen File anlegen. Dann gibt es den neuen Forschungsbericht.

14:02 **Christopher Raquet:** Perfekt. Das war's.

14:03 **Proband*in:** Okay, klar.

14:04 **Christopher Raquet:** Das ging wieder sehr schnell.

14:06 **Proband*in:** Okay.

14:08 **Christopher Raquet:** Wie fandest du den Prozess zum Erstellen?

14:11 **Proband*in:** Unkompliziert.

14:19 **Proband*in:** Wenn du mir nicht gesagt hast, dass die richtige Datei Entity oder die richtige Entity File ist, dann hätte ich mir erst mal da noch viel mehr Zeit genommen, das alles anzuschauen.

14:33 **Proband*in:** Was mich kurz verwirrt hat, ist dann dieses „Upload File“ or „Without File“ und dann noch dieses „Select File“.

14:45 **Proband*in:** Ja, ich gehe dann halt immer, also das hat mit dem Editor selber nichts zu tun. Ich gehe halt in solchen Fällen immer über die, ich sage jetzt mal, die sicherste Route.

15:00 **Christopher Raquet:** Ja, also da an dem Punkt ist auf jeden Fall schon jeder hängen geblieben, der es probiert hat. Das ist ein Punkt, den ich auf jeden Fall bei allen bemerkt habe.

15:08 **Christopher Raquet:** Genau, und das, das ich schon gesagt habe, das ist ein File, es ist eine schwierige Gratwanderung, weil ich habe mir gedacht, ich will ja nicht testen, wie gut RO-Crates ihre Sachen benannt haben.

15:20 **Christopher Raquet:** Deshalb habe ich es jetzt hingeschrieben.

15:23 **Proband*in:** Ja, ich meine, wenn du es einfach nur so hast und sagst, es gibt ein Forschungsbericht oder ein Bericht, der heißt neuer Forschungsbericht.pdf, dann hätte man sich halt da noch mal eine Gedenkminute nehmen müssen, um kurz sich einen Überblick zu verschaffen, was passt.

15:41 **Proband*in:** Und dadurch, dass du schon dazu gesagt hast, was passt, ist es eigentlich auch nur noch, such mal auf den Stichworten, das Richtige raus.

15:51 **Christopher Raquet:** Das ist vielleicht auch schon zu simpel, das stimmt.

15:53 **Proband*in:** Und dann hast du bei diesen, dadurch, dass du ja vorher, als du die RO-Crates eingeführt hast, dargelegt hast, dass diese Data Entities Dateien oder Verzeichnissen sein können, kommen eigentlich diese ganzen Contextual Entities gar nicht zum Tragen.

16:11 **Proband*in:** Das heißt, du musst ja eigentlich nur noch überlegen, ist diese Dateien ein Verzeichnis, oder ist das, was du hinzufügen willst, eine Datei oder ein Verzeichnis.

16:22 **Proband*in:** Und nachdem eh angegeben ist, ist eine PDF-Datei, fällt das mit dem Verzeichnis sofort raus, und bist du bei der Datei.

16:29 **Proband*in:** Das hätte man eigentlich auch gar nicht dazu schreiben müssen.

16:31 **Christopher Raquet:** Das ist voll wahr, da hast du recht.

16:34 **Christopher Raquet:** Gut, um die Korrektheit des Crates, ein letztes Mal zu überprüfen, möchtest du dir im Grafen einen Überblick verschaffen. Dort fällt dir auf, dass du für den neu hochgeladenen Forschungsbericht noch keinen Autor hinterlegt hast.

16:49 **Christopher Raquet:** Erstelle im Grafen, eine neue Kante, ausgehend vom neuen Forschungsbericht. Das Ende der Kante soll eine beliebige Person sein. Wähle Author als Eigenschaft für die Kante.

17:01 **Proband*in:** Also, das ist das ganz globale Menü. Da gibt es den Graf. Und jetzt habe ich da unten den neuen Forschungsbericht.

17:27 **Proband*in:** Ah, das ist überraschend.

17:29 **Christopher Raquet:** Das ist leider, das ist ein Safari-Problem, dass das nach oben herausgeht.

17:35 **Proband*in:** Okay, so, jetzt ist die Frage.

17:38 **Christopher Raquet:** Aber da würde stehen, Formatieren und Vollbild.

17:42 **Proband*in:** Okay, jetzt ist die Frage, wie kriege ich da ... ? Ah.

17:58 **Christopher Raquet:** Okay. Perfekt, das war es schon.

18:02 **Christopher Raquet:** Kannst du noch was zum Grafen sagen?

18:05 **Proband*in:** Was mich am Anfang stutzig gemacht hat, ich hätte da oben noch was erwartet, um etwas zu tun, von wegen Edge hinzufügen.

18:19 **Proband*in:** Weil es war halt bei den anderen Sachen, hast du immer irgendwie oben, das für alles im Kopf gehabt, und das ist da nicht so.

18:28 **Proband*in:** Da ist so bei dem Knoten, gibt es dieses Plus. Aber da muss man halt auch erstmal, also ich bin halt auch erst darauf gekommen, nachdem ich da irgendwie mit der Maus rumgefahren bin und dann gesehen habe, dass sich der Mauszeiger ändert

18:44 **Proband*in:** Und als ich dann da eine neue Kante wegziehen kann, das hat mich ein bisschen stutzig gemacht.

18:51 **Proband*in:** Also das ist das Einzige, was ich dir an der Stelle ankreiden würde, aber das ist, aus meiner Sicht, echt jammert auf hohem Niveau.

19:03 **Proband*in:** Ja, ansonsten, wenn man den Grafen irgendwie planar darstellen kann, wäre es schön.

19:10 **Christopher Raquet:** Das ist leider NP-vollständig.

19:12 **Proband*in:** Ja.

[...]

22:14 **Christopher Raquet:** Ich beende mal die Aufnahme.

A.3.3. Transkript eines Nutzertests (3)

00:00 **Christopher Raquet:** Dann, bevor wir uns den Editor ansehen, habe ich noch ein paar kurz allgemeine Fragen an dich.

00:05 **Christopher Raquet:** Was hast du studiert oder gearbeitet, bevor du zur DEM gekommen bist?

00:09 **Proband*in:** Ich hab Informationstechnik studiert, an der dualen Hochschule.

00:13 **Christopher Raquet:** In Karlsruhe?

00:15 **Proband*in:** Ja. Und da im Prinzip meinen Praxisteil gleich bei DEM, damals noch Software Methoden, an einem anderen Institut gemacht.

00:25 **Proband*in:** Ja, ich bin eigentlich schon seit immer hier in der Gruppe.

00:29 **Christopher Raquet:** Was machst du hier so?

00:31 **Proband*in:** Zum Teil Software-Entwicklung, wenn es die Zeit hergibt, aber halt auch viel administratives Arbeiten und ich koordiniere im Wesentlichen die Software-Entwicklung zum Forschungsdatenmanagement in unserer Gruppe und schaue da, dass die einzelnen Entwicklungen ein bisschen zusammen laufen und ein gemeinsames Bild ergeben.

00:52 **Christopher Raquet:** Okay, Forschungsdatenmanagement hört sich fast so an, wie als ob du weißt, was Research Object Crates sind?

00:57 **Proband*in:** Jawoll

00:57 **Christopher Raquet:** Hast du schon mal damit gearbeitet?

00:59 **Proband*in:** Ja, also nicht sehr tief, aber ich weiß, wie sie aufgebaut sind, was für Teile drin sind, wofür sie da sind.

01:08 **Christopher Raquet:** Sonst hätte ich zur Erklärung noch dieses Schaubild gehabt, aber wenn dir schon klar ist, was da ist, dann müssen wir nicht weiter drauf eingehen.

01:17 **Christopher Raquet:** Gut. Das lass ich einfach mal hier oben liegen.

01:22 **Christopher Raquet:** Dann machen wir als Nächstes mit spezifischeren Aufgaben weiter. Ich werde dir jede Aufgabe vorlesen und auch als kleinen Zettel für dich hinlegen. Und nur als Erinnerung: sag mir bitte alles, was du denkst und was dir auffällt, negativ oder positiv. Danke fürs Mitmachen.

01:38 **Christopher Raquet:** So, erstes Szenario. Du arbeitest als Teil einer Forschungsgruppe, die sich mit modernen Forschungsthemen befasst. Deine Aufgabe ist es, die Forschungsergebnisse deiner Gruppe zu veröffentlichen. Um die maschinelle Verarbeitung der Daten zu ermöglichen, entschied sich deine Forschungsgruppe Research Object Crates zu verwenden. Die nächste Veröffentlichung von Forschungsdaten steht kurz bevor. Ein Kollege hat bereits den Großteil vorbereitet und ein Crate erstellt. So soll es noch einige Details am Crate anpassen.

02:07 **Christopher Raquet:** Um das Crate bearbeiten zu können, muss es zunächst in dem Editor für Research Object Crates namens NovaCrate importieren. Verschaffe dir einen Überblick über die Startseite des Editors. Importiere anschließend das vorbereitete Crate. Du findest die ZIP-Datei auf dem digitalen Schreibtisch.

02:23 **Proband*in:** Okay. Gut.

02:29 **Proband*in:** Ich sehe die Startseite des Editors, habe einmal links die Navigation, mit wahrscheinlich schnell Zugriff für einzelne Aufgaben und rechts, müsste ich jetzt gucken, ob das Schaltflächen sind, oder ja, sind auch Schaltflächen, wo ich im Prinzip neue Crates aus einem existierenden Datensatz, wie auch immer erstellen kann, oder halt komplett neu anfangen.

03:01 **Proband*in:** Und ich habe eine gewisse Historie, wo ich dann wahrscheinlich die Crates sehe die ich zuletzt bearbeitet habe.

03:10 **Proband*in:** Genau, das heißt im Prinzip diese beiden Funktionen und diese sollten von meinem Verständnis identisch sein.

03:16 **Proband*in:** Was ich nun machen würde, ist importieren eines Crates, ich mache mal das hier, und habe dann die Auswahl eines Verzeichnisses.

03:35 **Proband*in:** Schaue ich mal rein, Schreibtisch.

03:37 **Christopher Raquet:** Wobei mein zustimmendes Ja war wahrscheinlich nicht ganz richtig an der Stelle.

03:41 **Proband*in:** Ja, ich bin auch gerade am überlegen, ob das wirklich das macht, was ich jetzt denke.

03:46 **Proband*in:** Und ich glaube, was ich mir jetzt gerade so erschlossen habe, „Start with Data“, ist glaube ich so ein Zwischending, dass ich schon Daten habe, aber ich will die Metadaten erstellen.

04:01 **Proband*in:** Und „Start from scratch“ ist, dass ich im Prinzip noch nichts habe, das heißt, ich will Metadaten erstellen und Daten erstellen.

04:13 **Proband*in:** Das heißt, hier erschließt sich der Unterschied mir gerade nicht.

04:15 **Proband*in:** Das heißt, ich würde jetzt nochmal hier auf die Sache gehen, da es ja wirklich um den Import geht.

04:20 **Proband*in:** Und dann habe ich eine komplette Crate als ZIP-File und die lade ich jetzt einfach mal hoch.

04:27 **Proband*in:** So, jetzt sehe ich die Navigation, wo ich die Entitäten, also im Prinzip die Metadaten, die in den Crate-Metadaten liegen, als Baumstruktur habe.

04:42 **Proband*in:** File Explorer sind wahrscheinlich die Daten zu der Crate. Der Graph, schau ich mal darauf.

04:49 **Proband*in:** Okay, das ist dann die grafische Darstellung. Mit den einzelnen Relationen.

04:53 **Proband*in:** JSON-Editor, dann kann ich das Ganze händisch editieren.

04:59 **Proband*in:** Context, gut, das sind dann halt hier nach dem Link-Data die verschiedene Kontexte, die hier verwendet werden.

05:06 **Christopher Raquet:** Perfekt, danke schön. Das war schon die erste Aufgabe.

05:10 **Proband*in:** Okay.

05:11 **Christopher Raquet:** So, das Crate enthält bereits den Großteil der nötigen Daten. Dazu gehören Forschungsberichte, die als Data Entities hinterlegt sind und Informationen zu Autoren und Organisationen, die als Contextual Entities hinterlegt sind. Überblick haben wir uns schon verschafft. Anschließend fällt dir auf, dass deinem Kollegen in der Data Entity „Example Image“ einige Fehler unterlaufen sind. Korrigiere die Eigenschaften Caption und Copyright Year. Die richtigen Werte können aus dem zugehörigen Bild entnommen werden.

05:38 **Proband*in:** Okay, gucken wir mal.

05:41 **Proband*in:** Ich wähle das „Example Image“ Entity aus.

05:49 **Proband*in:** Caption und was war Copyright Year, 2023.

05:53 **Proband*in:** Das heißt, ich würde jetzt schauen nach einem Image „lab-217043“.

06:03 **Proband*in:** Was ich jetzt gerne machen würde, ist, ich suche jetzt nach irgendeiner Möglichkeit, wie ich von dem Entity auf das File komme.

06:19 **Proband*in:** Das geht nicht über Klicken.

06:21 **Proband*in:** Copy haben wir gerade schon herausgefunden, geht auch nicht.

06:25 **Proband*in:** Hier oben ist eine Suche. Das heißt, ich würde einfach mal das rauskopieren und hier einfügen.

06:36 **Proband*in:** So, da passiert nichts.

06:44 **Proband*in:** „Find References“. Passiert auch nichts.

06:55 **Proband*in:** Okay, das heißt, ich würde mich jetzt händisch auf die Suche nach dem File machen. Weil ansonsten sehe ich jetzt nichts, was mir da irgendwie den Weg ebnen könnte.

07:08 **Proband*in:** „lab-2170...“, da habe ich das hier.

07:15 **Proband*in:** Copyright ist 2024, das kann ich mir schon mal merken. Die Frage ist, wo ich jetzt die Caption hernehmen sollte.

07:27 **Proband*in:** Aber ich gehe jetzt mal hier rüber zurück und finde es irritierend, dass hier unten das Ding wie ausgewählt aussieht.

07:39 **Proband*in:** Aber hier oben das nicht. Nur als kurze Anmerkung.

07:43 **Proband*in:** Korrigiere das, speicher das schon mal. So, „Caption for this Image“.

07:57 **Proband*in:** Ist jetzt Frage, was ich jetzt für eine Caption auswählen sollte?

08:01 **Christopher Raquet:** Das muss jetzt nichts ...

08:03 **Proband*in:** Im Prinzip würde ich jetzt Laborsymbolbild als Caption hernehmen.

08:07 **Christopher Raquet:** Genau.

08:09 **Proband*in:** Ich würde also den ganzen Kram hier der Außenrum ist wegschmeißen. Und das Ganze auch speichern.

08:22 **Christopher Raquet:** Perfekt. Gut, das war's.

08:26 **Christopher Raquet:** Nur um es gesagt zu haben, das haben auch schon deine Vorgänger nicht einfach gefunden, aber hier wäre auch ein Kopf gewesen.

08:35 **Christopher Raquet:** Der wäre aber wahrscheinlich an anderer Stelle besser platziert gewesen, um die Datei zu zeigen.

08:41 **Proband*in:** Okay, das Preview.

08:43 **Proband*in:** Die Frage für mich wäre dann halt, wenn ich ein Preview von einem File mache, also mir war ja klar, dass es ein Bild ist.

08:48 **Proband*in:** Wenn ich als User die Aufgabe hätte, einen Titel und ein Publication Year oder irgendwie ein Copyright Year anzugeben, dann würde ich instinktiv nicht auf das Preview von dem Bild klicken.

09:02 **Proband*in:** Also das ist schon untypisch, dass ein Copyright und Titel auf dem Bild direkt drauf sind. Ich würde jetzt eher irgendwie schauen, ob ich irgendwo Exif-Metadaten oder so was finde. Ob solche Informationen irgendwo sind. Von daher, ja.

09:17 **Christopher Raquet:** Hast du den Knopf gesehen?

09:19 **Proband*in:** Ich hatte den Knopf tatsächlich nicht gesehen, weil ich nicht danach gesucht habe.

09:25 **Proband*in:** Ich habe mich nach und nach am Bildpreview gesucht, um Metadaten zu kriegen.

09:29 **Christopher Raquet:** Das erschließt sich mir, das ergibt Sinn, ja.

09:32 **Christopher Raquet:** Gut, machen wir weiter.

09:34 **Christopher Raquet:** Anschließend möchtest du zu den Data-Entities „Lab Report Part A" und „Lab Report Part B" wichtige Informationen hinzufügen. Du solltest angeben, dass diese Dateien am KIT Campus Nord erstellt wurden. Eine passende Eigenschaft hierfür heißt Location Created.

09:51 **Christopher Raquet:** Füge die Eigenschaft Location Created zu den Data-Entities „Lab Report Part A" und „Lab Report Part B" hinzu. Setze sie auf KIT Campus Nord. Die Entity für KIT Campus Nord existiert bereits.

10:04 **Proband*in:** Okay.

10:06 **Proband*in:** Also „Lab Report Part A“, ist klar.

10:10 **Proband*in:** So, jetzt schaue ich.

10:17 **Proband*in:** Das sind im Prinzip die Eigenschaften, die das Ding schon hat.

10:20 **Proband*in:** Ich will jetzt ein Property hinzufügen.

10:23 **Proband*in:** Das nennt sich... Passende Eigenschaft Location Created.

10:33 **Proband*in:** Location Created ist ein Place, okay.

10:36 **Proband*in:** Das Ding füge ich jetzt hinzu. Und jetzt will ich...

10:47 **Proband*in:** „Create“, „Select“?

10:49 **Proband*in:** Also instinktiv würde ich jetzt Create machen, obwohl mir das ein bisschen komisch vorkommt von der Semantik.

10:56 **Proband*in:** Ich meine Created habe ich es ja schon.

11:01 **Proband*in:** Also sage ich jetzt Create. Das heißt, ich kriege jetzt eine Auswahl. Contextual Entities. Dadurch, dass es ein Place ist, habe ich das schon vorausgewählt.

11:15 **Proband*in:** Das erschließt sich mir auch. „Entity Name“. „Identifier for Place“.

11:28 **Proband*in:** Das heißt, hier oben habe ich die allgemeine Anleitung, was ich damit machen muss.

11:35 **Proband*in:** Und hier habe ich dann kontextabhängig noch die Beispiele, was wahrscheinlich aus dem Schema ausgelesen wird.

11:45 **Proband*in:** Hier würde ich jetzt eigentlich... KIT Campus Nord einfügen. Aber es steht ja geschrieben, dass es das schon existiert.

12:01 **Proband*in:** Das heißt, ich gehe mal hier zurück. Okay, hier auf Select.

12:10 **Christopher Raquet:** Ja, das war es schon.

12:12 **Proband*in:** Nee, war’s noch nicht, muss ich fürs Zweite auch noch machen.

12:15 **Proband*in:** Also Add Property, Location und Select.

12:26 **Proband*in:** Gut.

12:29 **Christopher Raquet:** War dir da irgendwas unklar, was du anders strukturiert hättest?

12:37 **Proband*in:** Mir war dieser Unterschied zwischen Create und Select war mir wieder nicht ganz klar. Wobei sich das vielleicht während der Arbeit erschließt.

12:52 **Proband*in:** Ich fange jetzt mit einer existierenden Create an. Und ich soll jetzt etwas erstellen. Das heißt, intuitiv würde ich jetzt zuerst ein Create machen.

13:02 **Proband*in:** Das heißt, die Information, dass sowas schon existiert in diesem Crate-Kontext. Die habe ich ja eigentlich gar nicht. Das weiß ich ja nicht.

13:10 **Proband*in:** Das heißt, dass ich jetzt Select drücke, ist eher unwahrscheinlich, wenn ich irgendwas hinzufügen will.

13:15 **Proband*in:** Das heißt, ich würde immer erst Create drücken. Und das Select würde ich halt machen, wenn ich weiß, okay, ich weiß, dass ich KIT Campus Nord angeben will.

13:26 **Proband*in:** Und ich weiß, dass ich vorher schon mal KIT Campus Nord in der Crate verwendet habe.

13:31 **Proband*in:** Aus dem Workflow, wenn ich es von vornherein mache, dann ja, für jemand, der damit anfängt, ist es eher schwierig, weil ich will dann auch keine unnötigen Clicks machen.

13:43 **Proband*in:** Und ich will jetzt nicht irgendwie erst „Select“ klicken, um dann zu sehen, okay, da ist nichts.

13:47 **Proband*in:** Aber so, wenn man es weiß, und wenn der Workflow passt bis dahin, finde ich es sehr intuitiv gelöst.

13:56 **Proband*in:** Das ist ja sehr angenehm.

13:58 **Proband*in:** Das du halt die Möglichkeit hast, externe Sachen zu verlinken und man sich da keine Gedanken mehr machen muss.

14:04 **Christopher Raquet:** Perfekt, Dankeschön.

14:07 **Christopher Raquet:** Dir fällt auf, dass in der Contextual Entity zu Björn Grüning private Daten hinterlegt sind. Vor der Veröffentlichung möchtest du diese natürlich löschen. Entferne die Eigenschaft für das Geburtsdatum von der Entity Björn Grüning.

14:20 **Proband*in:** Gut, das heißt, ich schau mir jetzt die Björn Grüning an.

14:25 **Proband*in:** Kann ich eigentlich auch über den Graphen die Sachen auswählen?

14:28 **Christopher Raquet:** Kannst du gerne alles probieren.

14:30 **Proband*in:** Kann was werden.

14:32 **Proband*in:** So, jetzt können wir über den Graphen arbeiten. So, jetzt suche ich mir den.

14:36 **Proband*in:** Kann ich da auch Suchen drin? Ah, das ist schön. Das gefällt mir, aber dann komme ich wieder [zum Entity Editor].

14:43 **Proband*in:** So, das heißt, ich kann jetzt über den Graphen nicht direkt... Rechtsklick...

14:53 **Proband*in:** Okay, das heißt, das ist mein Editor. Damit muss ich arbeiten. Das ist in Ordnung.

14:57 **Proband*in:** So, was haben wir gesagt? Geburtsdatum sollen wir entfernen.

15:01 **Proband*in:** Geburtsdatum so. Dann gehen wir mal, ich glaube hier.

15:08 **Proband*in:** Okay, das heißt, hier kann ich das Entity löschen.

15:16 **Proband*in:** Okay, das geht aber nur, weil es ein Entity...

15:18 **Proband*in:** Ich überlege gerade, ich habe das ja vorhin zufällig bei dem Bildnamen auch gehabt. Und da war im Kontextmenü bloß das Copy. Das heißt, ich kann das gesamte Entity nicht darüber löschen.

15:30 **Proband*in:** Das muss ich dann wahrscheinlich hier machen.

15:32 **Proband*in:** Ich gucke gerade mal, dass nichts mit der Aufgabe zu tun war. Okay, aber so einzeln Entities dadurch, dass sie mit dem verbunden sind, kann ich hier löschen.

15:48 **Proband*in:** Okay, jetzt ist die Frage. Ganz kurz, hat auch nichts damit zu tun.

15:58 **Proband*in:** Okay, Revert Changes. Sehr gut, genau, das wollte ich.

16:02 **Proband*in:** Perfekt, gefällt mir.

16:04 **Proband*in:** Speichern, gelöscht.

16:07 **Christopher Raquet:** Sehr gut. Dankeschön.

16:11 **Christopher Raquet:** Und das fandest du gut, dass die Eigenschaft quasi da noch war und angezeigt hat, dass...

16:19 **Proband*in:** Ich fand das mit dem Undo, fand ich extrem gut, weil normalerweise...

16:21 **Proband*in:** Ich hätte jetzt, zuerst hätte ich gesagt, okay, ich hätte irgendwie eine Bestätigung erwartet. So ein Confirm-Dialog, wo gefragt wird wirklich löschen, ja, nein.

16:29 **Proband*in:** Aber dadurch, dass du halt klarmachst, dass es das in dem Zwischenstand ist, der erst aktiviert wird, wenn es gespeichert ist, kann ich damit leben, was man jetzt hätte noch machen können, was es ein bisschen einfacher noch macht, ist...

16:42 **Proband*in:** Also für mich, wo mein Verständnis...

16:45 **Proband*in:** Wenn ich jetzt hier statt dem Safe ein Cancel habe, weil hier in das Kontext-Menü zu gehen, das macht vielleicht nicht jeder Instinktiv, aber die Funktionalität ist da, also für jemand, der sich auskennt und so ein bisschen rumfummelt, der findet das auf jeden Fall.

16:59 **Christopher Raquet:** Perfekt.

17:01 **Christopher Raquet:** Gut, kurz bevor du dich in den wohlverdienten Feierabend verabschieden wolltest, kommt dein Vorgesetzter mit einem neuen Forschungsbericht.

17:08 **Proband*in:** Typisch.

17:10 **Christopher Raquet:** Er hat diesen in Form einer PDF-Datei auf deinem Schreibtisch abgelegt. Du sollst den neuen Forschungsbericht zum Crate hinzufügen.

17:16 **Christopher Raquet:** Erstelle eine neue Entity für den Laborbericht. Die zugehörige Datei findest du als „neuer Forschungsbericht.pdf“ auf dem Schreibtisch. Der richtige Typ für eine Entity, die eine Datei beschreibt, ist File.

17:30 **Proband*in:** Okay.

17:32 **Proband*in:** Also für mich ist jetzt die Frage, lad ich erst das File hoch oder erstelle ich erst die Entity?

17:41 **Proband*in:** Ich gehe jetzt mal die Richtung, dass ich zuerst die Entity erstelle.

17:43 **Proband*in:** Also von meinem Wissen her weiß ich, dass du für eine Entity einen eindeutigen Identifier brauchst.

17:49 **Proband*in:** Der üblicherweise der Dateiname ist in der Crate. Das heißt, ich frage mich jetzt, was passiert, wenn ich jetzt eine Entity hinzufügen will?

17:59 **Proband*in:** Ich lese jetzt das hier oben nicht.

18:01 **Proband*in:** Ich schau hier, weil hier habe ich gesehen, dass hier ein Kontextmenü gibt.

18:04 **Proband*in:** So, hier gibt es kein Kontextmenü.

18:06 **Proband*in:** Das heißt, ich muss mir das New Entity irgendwo anders her saugen.

18:09 **Proband*in:** Ich finde jetzt den Button hier oben. So sage ich jetzt New Entity. Okay.

18:17 **Proband*in:** Ich habe jetzt hier die verschiedenen Entities.

18:20 **Proband*in:** Mir fällt auf, dass je nachdem, wo ich das hinzufüge, habe ich die Möglichkeiten aktiviert oder deaktiviert.

18:25 **Proband*in:** Das gefällt mir auch sehr gut.

18:27 **Proband*in:** Ich will jetzt einen File hinzufügen. Das heißt, ich gehe auf File.

18:32 **Proband*in:** Ich kann jetzt hier ein File auswählen. Mit einem Pfad, mit einem Namen.

18:41 **Proband*in:** Wie viele Aufgaben haben wir noch?

18:44 **Christopher Raquet:** Wir haben Zeit, genug Zeit.

18:46 **Proband*in:** So, jetzt, wie gesagt, vor dem Hintergrund, dass ich eigentlich weiß, dass man eine File braucht oder zumindest ein Identifier, würde ich jetzt das einfach mal lassen und frage mich jetzt gerade, was an diesem Formular mandatory ist.

19:01 **Proband*in:** Das heißt, ich kann hier nichts auswählen. Ich gebe hier einfach mal... Gibt es da irgendwas Forschungsbericht... „Forschungsbericht.pdf“.

19:13 **Proband*in:** So, Entity Name. Keine Ahnung. Auch mal...

19:20 **Proband*in:** Oh, siehst du, deswegen hasse ich Mac Tastaturen.

19:22 **Christopher Raquet:** Control ist ganz links.

19:24 **Proband*in:** Ja, ich tippe es einfach nochmal.

19:26 **Proband*in:** So, „Forschungsbericht“. Jetzt mache ich das mal.

19:34 **Proband*in:** Ja, okay. So, jetzt habe ich hier Forschungsbericht.

19:42 **Proband*in:** Hab da aber noch kein File dazu.

19:44 **Proband*in:** Was passiert bei Preview-File? Da passiert nichts, weil ist ja noch nicht da.

19:52 **Proband*in:** So, File-Explorer.

19:54 **Proband*in:** File-Explorer habe ich jetzt auch noch kein File. Muss ich noch etwas mit machen? „Erstelle Forschungsbericht... Neuer Forschungsbericht...“ Okay, das ist egal.

20:06 **Proband*in:** Was ist, wenn ich das jetzt mache?

20:11 **Proband*in:** Wieso geht das nicht zu klicken?

20:16 **Proband*in:** Ah, okay, „Upload File“ aktiviert das.

20:20 **Proband*in:** Gut.

20:26 **Proband*in:** Okay.

20:28 **Proband*in:** „Neuer Forschungsbericht“.

20:30 **Proband*in:** Das heißt sogar anders.

20:32 **Proband*in:** Okay.

20:34 **Proband*in:** So, wenn ich das jetzt mache, dann stellt ihr mir einen neuen Pfad mit einem neuen Namen. Das will ich aber eigentlich nicht.

20:40 **Proband*in:** Eigentlich würde ich hier gerne die Dateien zufügen.

20:50 **Proband*in:** File, da ist der Typ, da ist der Name.

20:54 **Christopher Raquet:** Bevor ich dich ewig Suchen lasse, würde ich sagen, dass es nicht geht.

20:58 **Proband*in:** Es geht nicht, okay.

21:00 **Christopher Raquet:** Also, diese Reihenfolge klappt nicht.

21:02 **Christopher Raquet:** Man kann nicht Entity erstellen und dann Datei. Andersrum wäre es möglich.

21:06 **Christopher Raquet:** Also, wenn das Crate und Dateien enthalten würde, ohne Entity, könnte man die trotzdem im File-Explorer sehen und dann Entity zufügen.

21:14 **Proband*in:** Ja, aber nur, wenn die zufällig die gleiche ID haben, den gleichen Namen.

21:18 **Christopher Raquet:** Gut, wenn beide schon existieren, dann sind sie ja sowieso über den Identifier verlinkt.

21:24 **Proband*in:** Wie mache ich das? Also, was ich jetzt...

21:28 **Proband*in:** Gut, jetzt unabhängig von der, von der Problematik jetzt.

21:30 **Proband*in:** Ich habe ja auch die Möglichkeit, Dateien zu referenzieren.

21:34 **Christopher Raquet:** Da müsste man den Identifier ändern.

21:38 **Proband*in:** Also, der Identifier muss dann im Prinzip die URL sein. Das heißt, in dem Fall jetzt habe ich jetzt im Prinzip eine ungültige Crate erstellt.

21:44 **Christopher Raquet:** Ja.

21:48 **Proband*in:** Gut, das heißt, ich lösche das mal wieder und gehe mal den offiziellen Weg.

21:56 **Proband*in:** So, „Upload File".

22:02 **Proband*in:** Okay, das heißt, bei „Without File" würde ich dann eine URL angeben müssen. Okay, gut. So, dann mache ich das nochmal richtig.

22:10 **Proband*in:** Forschungsbericht, Forschungsbericht.

22:14 **Proband*in:** Was passiert, wenn ich jetzt den Pfad ändere?

22:22 **Christopher Raquet:** Das kannst du gerne ausprobieren.

22:24 **Proband*in:** Ja. So, den Namen lassen wir mal so.

22:32 **Proband*in:** So, neuer Forschungsbericht, Preview.

22:38 **Proband*in:** Wieso ist da Kaffee drauf.

22:42 **Proband*in:** Okay, also das funktioniert. Das heißt, wahrscheinlich wird dann das File entsprechend umbenannt.

22:48 **Proband*in:** Okay. Gut.

22:52 **Proband*in:** Da ist echt Kaffee drauf.

22:54 **Christopher Raquet:** Sehen wir in der nächsten Aufgabe.

22:56 **Proband*in:** Okay.

22:58 **Proband*in:** Gut, aber das sollte es jetzt im Prinzip sein.

23:00 **Christopher Raquet:** Genau.

23:02 **Christopher Raquet:** Also, du hast jetzt das ohne Datei einfach mal probiert? Ja.

23:11 **Christopher Raquet:** Weil du... quasi technisch ausprobieren wollte, wie der Editor das macht. Datei hinzufügen, Entity hinzufügen.

23:17 **Proband*in:** Ja.

23:19 **Proband*in:** Ja, weil die Möglichkeit besteht.

23:21 **Proband*in:** Also, ich habe die Möglichkeit. Es sagt ja niemand, dass ich als Nutzer jetzt immer zwischen dieser File Perspektive und der Metadata Perspektive wechseln muss. Gedanklich.

23:34 **Proband*in:** Ich kann das ja im Prinzip so machen, dass ich mir erst meine Crate aufbaue. Schön, so wie ich es gerne möchte. Ja, und dann später die Dateien dazu packe.

23:43 **Christopher Raquet:** Okay.

23:45 **Proband*in:** Deswegen habe ich das jetzt so herumprobiert.

23:48 **Proband*in:** Es ist halt die Frage, wie man es handlen soll.

23:54 **Proband*in:** In dem Fall, wenn man File auswählt und Upload hat. Also, ich vermute mal, dass hier in dem Fall beide Mandatory sind.

24:06 **Christopher Raquet:** Ja.

24:08 **Proband*in:** Okay, dann hat der einen internen Identifier, was passiert in dem Fall? Worauf verweist das jetzt?

24:16 **Christopher Raquet:** Das verweist auf nichts.

24:18 **Proband*in:** Auf nichts. Und das ist legal entsprechend der Crate Spezifikation?

24:22 **Christopher Raquet:** Ich glaube schon, ja. Es ist nur die Frage, ob der Identifier mit dem Hashtag für ein Data Entity legal ist im spezifischen.

24:30 **Proband*in:** Ja.

24:31 **Christopher Raquet:** Aber man würde da im Idealfall natürlich stattdessen eine URL als Identifier angeben.

24:35 **Proband*in:** Genau, das wäre jetzt halt die Frage. Ich meine, jetzt bist du ja sehr benutzerfreundlich, dass du sagst, okay, man muss halt nur den Namen eingeben.

24:43 **Christopher Raquet:** Genau, das ist vor allem für die Contextual Entities dann sinnvoll.

24:47 **Christopher Raquet:** Wenn man lokale Contextual Entities ohne URL hat.

24:49 **Proband*in:** Ja, ja, ja.

24:51 **Proband*in:** Das macht es jetzt dem Nutzer extrem leicht, hier gerade an der Stelle mit den Files, Fehler zu machen.

24:55 **Christopher Raquet:** Das stimmt.

24:57 **Proband*in:** Und hier würde ich halt dann wirklich ganz strikt sagen, wenn das ausgewählt ist, muss halt ein File ausgewählt sein.

25:04 **Christopher Raquet:** Ja.

25:06 **Christopher Raquet:** Das stimmt.

25:08 **Christopher Raquet:** Gut. Wunderbar.

25:12 **Proband*in:** Gut.

25:14 **Christopher Raquet:** Um die Korrektheit des Crates ein letztes Mal zu überprüfen, möchtest du dir im Graphen einen Überblick verschaffen. Dort fällt dir auf, dass du für den neu hochgeladenen Forschungsbericht noch keinen Autor hinterlegt hast.

25:24 **Christopher Raquet:** Erstelle im Graphen eine neue Kante, ausgehend vom neuen Forschungsbericht. Das Ende der Kante soll eine beliebige Person sein. Wähle „Author“ als Eigenschaft für die Kante.

25:32 **Proband*in:** Okay.

25:34 **Proband*in:** Graph.

25:36 **Proband*in:** So, das heißt, ich schaue mal den Graphen an. Das habe ich ja vorhin schon mal.

25:44 **Proband*in:** So. Jetzt suche ich mir meinen neuen Forschungsbericht.

25:48 **Proband*in:** Der ist hier.

25:54 **Proband*in:** Und jetzt soll ich für den einen Author festlegen. Jetzt habe ich hier den Vergleich. Hier ist der Author einfach eine ausgehende Verbindung.

26:06 **Proband*in:** Das heißt, ich kann das wahrscheinlich so ziehen. Und ich nehme mal die Sabrine einfach hier.

26:18 **Proband*in:** Jetzt habe ich die Verbindung gezogen. Ich habe noch nicht gesagt, dass es ein Author ist. Das heißt, das würde ich an der Stelle machen.

26:24 **Proband*in:** Und dann habe ich meine Verbindung.

26:26 **Proband*in:** Was mir jetzt hier auffällt, man hat jetzt hier für die einzelnen Entities nur die cratespezifischen Markierungen.

26:38 **Proband*in:** Also entweder File oder Contextual entity.

26:44 **Proband*in:** Ich frage, für mich wäre jetzt, ob man hierbei bei solchen Sachen vielleicht noch spezifisch sein kann.

26:52 **Proband*in:** Und sagen kann, okay, man findet halt die Persons schneller.

26:56 **Proband*in:** Wenn ich mir jetzt vorstelle ich habe hier sehr, sehr viele von diesen contextual entities. Dann wird es halt für den Nutzer erstmal schwierig zu sehen.

27:06 **Proband*in:** Schwierig zu sehen, okay, was sind denn jetzt überhaupt potenzielle Autoren? Die Frage wäre jetzt zum Beispiel auch, kann ich jetzt Campus Nord als Author angeben?

27:18 **Proband*in:** Kann ich auch angeben, weil Person und Organisation, okay, das ist interessant, haben wir denn was anderes?

27:26 **Proband*in:** Nee, haben wir natürlich nicht.

27:30 **Proband*in:** Place haben wir. Place.

27:34 **Proband*in:** Also, auf was möchte ich denn Place draufziehen?

27:38 **Proband*in:** Oh.

27:42 **Christopher Raquet:** Ja, das ist möglich. Auch, wenn es natürlich eigentlich kein valides Crate ist.

27:50 **Proband*in:** Okay.

27:52 **Christopher Raquet:** Und es hängt dann im Endeffekt damit zusammen, dass ich eigentlich noch quasi eine umfassende Validierung für dieses Crate einbauen wollte.

28:00 **Christopher Raquet:** Also für den Editor meine ich. Deshalb gibt es auch hier diese Schaltfläche.

28:03 **Proband*in:** Okay.

28:04 **Christopher Raquet:** Hat aber letztendlich dann nicht in den Umfang gepasst.

28:07 **Proband*in:** Okay.

28:08 **Christopher Raquet:** Deshalb bleiben solche Sachen dann eben übrig.

28:11 **Proband*in:** Ja.

28:13 **Proband*in:** So, für mich jetzt aber die Frage, jetzt weiß ich, dass ich hier einen Fehler gemacht habe. Wie kriege ich das Ding wieder entfernt?

28:22 **Proband*in:** Ich schaue mal kurz, „Forschungsbericht“ ..., „Author“ ..., hier kann ich das wahrscheinlich entfernen, mache ich das mal.

28:32 **Proband*in:** Und dann speichere ich den Kram.

28:36 **Proband*in:** Mal gucken. Ja.

28:40 **Christopher Raquet:** Mhm. Ja. Perfekt.

28:44 **Christopher Raquet:** Das war’s dann mit der letzten Aufgabe. Okay.

28:46 **Proband*in:** Ja.

28:48 **Christopher Raquet:** Kommentare, Anmerkungen, Meinungen.

28:52 **Christopher Raquet:** Sagen wir einfach alles. Was in deinem Kopf herumschwirrt.

28:56 **Proband*in:** Ja. Also, rein optisch gefällt es mir sehr gut.

29:02 **Proband*in:** Von der Handhabung her auch. Die Sachen, die ich jetzt so ein bisschen umgetrieben habe, die habe ich schon erwähnt.

29:10 **Christopher Raquet:** Mhm.

29:12 **Proband*in:** Wie gesagt, mit der Validierung das ist ja, das macht mir halt ein bisschen Bauchweh.

29:22 **Proband*in:** Einfach, weil ich weiß, dass der Nikolai, der die RO-Crate Library initial angefangen hat, dass der da relativ viel Wert drauf gelegt hat und das auch so ein

bisschen als Alleinstellungsmerkmal hatte, dass diese Crate Library im Prinzip solche Fehler recht zuverlässig erkennt.

29:46 **Proband*in:** Und deswegen finde ich es schade, dass das nicht mehr, nicht mehr möglich ist.

29:50 **Christopher Raquet:** Also, um kurz zu unterbrechen, also was sie tatsächlich gut macht, ist den Typ der Entities richtig zu bestimmen.

29:58 **Proband*in:** Okay.

29:58 **Christopher Raquet:** Also, wenn man den jetzt über den JSON Editor händisch falsch machen würde, also zum Beispiel eine Data Entity als Dataset deklarieren würde, das würde der merken.

30:08 **Christopher Raquet:** Aber solche fehlerhaften Referenzen und so interessieren den leider nicht.

30:12 **Proband*in:** Okay, gut. Also geht das schon auf Library-Ebene nicht?

30:15 **Christopher Raquet:** Ja, genau.

30:18 **Proband*in:** Dann hier an der Stelle ist mir diese „Property Overview", die war mir nicht sofort klar, was die macht.

30:32 **Proband*in:** Habe ich dann später kapiert, es kommt dann so ein bisschen aus diesem IDE-Umfeld, wo du im Prinzip hier deinen Source Tree hast und hier dann im Prinzip, abhängig vom ausgewählten, die Elemente.

30:46 **Proband*in:** Hat sich mir am Anfang nicht so erschlossen.

30:52 **Proband*in:** Ich gucke aber gerade mal, wo das herkam.

31:02 **Proband*in:** Okay, wahrscheinlich hier von der Ebene noch, weil ich da noch gar nichts ausgewählt hatte, also aktiv noch nichts ausgewählt hatte, aber hier schon Sachen standen.

31:10 **Proband*in:** Deswegen habe ich das eigentlich die meiste Zeit ignoriert.

31:14 **Proband*in:** Und ja, das macht natürlich total Sinn so was zu haben.

31:18 **Proband*in:** Jetzt wissen wir, was da ist.

31:20 **Proband*in:** Okay, das wird dann auch ausgewählt, wunderbar.

31:24 **Proband*in:** Auch nützlich.

31:26 **Proband*in:** Das mit der Suche, das hatte ich am Anfang verwendet, um diesen Bildnamen zu finden, hat sich mir jetzt bis zum Schluss nicht erschlossen, wofür die Suche jetzt da ist oder wo ich jetzt da was finden kann.

[...]

Verabschiedung nach der Aufnahme

A.3.4. Transkript eines Nutzertests (4)

00:00 **Christopher Raquet:** Was hast du studiert oder gearbeitet, bevor du zu DEM gekommen bist?

00:03 **Proband*in:** Ich habe vorher Biological Sciences studiert, aber also eher naturwissenschaftlich im Hintergrund, genau.

00:09 **Proband*in:** Ich habe dann zwar über Bioinformatik natürlich wieder in den Bezug zu IT dann gefunden, aber ja, das ist mein Hintergrund.

00:18 **Christopher Raquet:** Was machst du bei DEM?

00:20 **Proband*in:** Bei DEM bin ich hauptsächlich in der, also ich mache meine Promotionen im Rahmen meiner wissenschaftlichen Arbeit, als wissenschaftlicher Mitarbeiter, so rum.

00:29 **Proband*in:** Das enthält auf der einen Seite viel konzeptioneller Arbeit, aber natürlich auch teilweise Implementierungsarbeit, ja, für Services.

00:39 **Christopher Raquet:** Okay, in welche Richtung gehen die Services ungefähr?

00:42 **Proband*in:** Ja, es ist Infrastruktur, würde ich sagen, also auch um technische, also aus den Konzepten stammende technische Details zu realisieren und zu implementieren.

00:53 **Christopher Raquet:** Okay. Bist du schon mit dem Konzept der Research Object Crates vertraut?

00:59 **Proband*in:** Ja, bin ich auf jeden Fall schon mit in die Berührung gekommen, aber wahrscheinlich nicht auf dem Experten-Level.

01:06 **Christopher Raquet:** Das ist auch gar nicht nötig.

01:07 **Christopher Raquet:** Hast du schon mal damit gearbeitet?

01:10 **Proband*in:** Ja, aber in dem Sinne nicht, also ich meine, ich habe mir natürlich schon mal so einen Container angeschaut, nur die Struktur dahinter.

01:16 **Proband*in:** Aber ich habe es jetzt nicht irgendwie in einem meiner Frameworks oder Arbeitsroutinen integriert.

01:23 **Christopher Raquet:** Okay, dann gehe ich nur noch mal ganz kurz darauf ein.

01:26 **Christopher Raquet:** Ich habe hier ein Schaubild, wie so ein Crate aussieht.

01:29 **Christopher Raquet:** Das ist einfach nur eine ZIP-Datei.

01:31 **Proband*in:** Ja.

01:32 **Christopher Raquet:** Da drinnen enthalten sind dann alle Dateien, die du eben in deinem Paket haben willst, so wie die Metadaten-Datei, die diese Dateien wieder beschreibt.

01:39 **Proband*in:** Genau, ja.

01:41 **Proband*in:** So ungefähr habe ich das dann auch mitbekommen.

01:44 **Christopher Raquet:** Genau, dann haben wir hier zum einen die Data-Entities, die beschreiben immer genau eine Datei oder einen Ordner.

01:50 **Proband*in:** Okay.

01:51 **Christopher Raquet:** Contextual Entities liefern dann Kontextinformationen.

01:54 **Christopher Raquet:** Zum Beispiel kannst du da eine Entity für einen Autor erstellen.

01:58 **Proband*in:** Okay.

01:59 **Christopher Raquet:** Und dann hast du nicht nur den Namen, sondern kannst zu dem Autor alles angeben, was du weißt.

02:03 **Christopher Raquet:** Und das eben für alle Kontext-Entities, die einem einfallen, also Ort, Firma, alles Mögliche.

02:12 **Christopher Raquet:** Die Root-Entity ist sonst noch wichtig, die beschreibt das Crate selbst, also ihre eigenen Metadaten, wer es herausgibt und wer daran gearbeitet hat.

02:21 **Christopher Raquet:** Die Metadaten-Entity kann sich auch noch selbst beschreiben, aber das macht man im Üblichen nicht [selbst].

02:27 **Proband*in:** Ach so, dass wir jetzt noch mal dieses separate JSON-File beschreiben.

02:31 **Christopher Raquet:** Genau.

02:32 **Christopher Raquet:** Die sind eigentlich, das ist keine Ahnung, so eine reine Formale, die ändern sich eigentlich nicht die Metadaten.

02:38 **Proband*in:** Ja.

02:40 **Christopher Raquet:** Genau.

02:45 **Christopher Raquet:** Passt.

02:47 **Christopher Raquet:** Genau, dann machen wir mit den Aufgaben weiter.

02:49 **Christopher Raquet:** Wie gesagt, sag alles, was du denkst, negativ oder positiv.

02:53 **Proband*in:** Okay.

02:54 **Christopher Raquet:** Genau.

02:55 **Christopher Raquet:** Ich werde jetzt die Aufgaben nacheinander, es sind sechs Stück, eins dann vorlesen und dir dann nochmal hinlegen.

03:00 **Proband*in:** Okay.

03:01 **Christopher Raquet:** Und dann kannst du dir beliebig Zeit lassen.

03:03 **Proband*in:** Okay.

03:04 **Christopher Raquet:** Du arbeitest als Teil einer Forschungsgruppe, die sich mit modernen Forschungsthemen befasst.

03:08 **Christopher Raquet:** Deiner Aufgabe ist es, die Forschungsergebnisse deiner Gruppe zu veröffentlichen.

03:12 **Christopher Raquet:** Und die maschinelle Verarbeitung der Daten zu ermöglichen, entschied sich deine Forschungsgruppe Research Object Grates zu verwenden.

03:18 **Christopher Raquet:** Die nächste Veröffentlichung von Forschungsdaten steht kurz bevor.

03:22 **Christopher Raquet:** Ein Kollege hat bereits den Großteil vorbereitet und ein Crate erstellt.

03:25 **Christopher Raquet:** Das heißt, noch einige Details am Crate anpassen.

03:28 **Christopher Raquet:** Um das Crate bearbeiten zu können, musst du es zunächst in den Editor für Research Object Grates, NovaCrate importieren.

03:34 **Christopher Raquet:** Verschaffe dir einen Überblick über die Startseite des Editors und importiere anschließend das vorbereitete Crate.

03:40 **Christopher Raquet:** Du findest die ZIP Datei auf dem Schreibtisch, auf dem digitalen Schreibtisch.

03:44 **Proband*in:** Genau.

03:45 **Proband*in:** Okay, ich soll dir jetzt erst mal importieren.

03:52 **Christopher Raquet:** Zunächst erst mal gerne beschreiben, was du siehst, der grobe Aufbau.

03:56 **Proband*in:** Ja, genau. Ich sehe hier die Startseite des Userinterface.

04:00 **Proband*in:** Hier mit der Bezeichnung, dann, ich würde sagen, Übersichtsliste.

04:06 **Proband*in:** Ich muss mal gucken, ob das hier praktisch auf weitere, wahrscheinlich Reiter führt, die hier aufgelistet sind.

04:13 **Proband*in:** „Import Existing Crate“ ist wahrscheinlich schon das hier. „Start with data“, „start from scratch“, wenn ich jetzt hier mal drauf gehe.

04:19 **Proband*in:** Ah, okay, dann öffnet sich das Pop-up. Was ist, wenn ich hier drauf gehe? „Start with data“. Ist es das gleiche, wie hier?

04:27 **Proband*in:** Aha.

04:30 **Proband*in:** Okay, dann würde ich das kriegen, kann ich hier irgendwas sagen? Ah, okay, das ist also quasi das Gleiche. Okay.

04:36 **Proband*in:** Ja, dann frage ich mich an der Stelle, also auf jeden Fall, warum wird das New Crate hier direkt angezeigt? Ist es einfach so der Standard-Einstieg, sozusagen, wie Landingpage?

04:47 **Proband*in:** Documentation würde mich dann, ah, okay, das ist als Pop-up. Und dann würde ich hier auf externe Webseiten weitergeleitet werden.

04:54 **Proband*in:** Genau, also ich habe mich jetzt als erstes Mal auf der linken Seite orientiert, irgendwie eine Übersicht gesucht. Ich habe mir dann schon gedacht, dass das halt irgendwie dann auf der rechten Seite spezifisch ist.

05:05 **Proband*in:** Ja, also ich denke, dass das jetzt hier speziell für den zweiten Eintrag ist, da kommt man dann relativ schnell drauf.

05:14 **Proband*in:** Toggle Theme, kann ich mir das gerade... Ja, okay, gut. Ja, dann bleiben wir bei Schwarz. Angenehmer.

05:22 **Proband*in:** Ne gut, also gefällt mir eigentlich vom Aufbau, es ist recht intuitiv.

05:27 **Proband*in:** Man fängt ja eigentlich grundsätzlich eher links dann an, fängt an sich zu orientieren und zu lesen. Und kann den Aufbau eigentlich gut nachvollziehen.

05:35 **Christopher Raquet:** Perfekt, danke schön.

05:37 **Proband*in:** Gut, jetzt habe ich ja eine Aufgabe, jetzt darf ich importieren.

05:40 **Proband*in:** Und, naja, das scheint ja recht straight forward zu sein. Dann nehme ich diesen Forschungsbericht, weil wenn ich mich nicht täusche, war das der hier, oder?

05:48 **Christopher Raquet:** Ja, das ist richtig.

05:50 **Proband*in:** Ja, muss ja nur so sein. Also, Hochladen.

05:53 **Christopher Raquet:** Okay, wunderbar.

05:55 **Proband*in:** Dann sind wir jetzt schon mal an dem Punkt.

05:57 **Proband*in:** Das war wahrscheinlich die erste Aufgabe.

05:59 **Christopher Raquet:** Genau, du darfst gerne wieder beschreiben, was du siehst, was du für was hältst, aber am besten noch nichts drücken.

06:05 **Christopher Raquet:** Okay, beziehungsweise links die Navigation, darfst du benutzt.

06:09 **Proband*in:** Die darf ich benutzen.

06:11 **Proband*in:** Genau, da gebe ich auch erstmal wieder hier Entities, die Kategorie. Dann sind hier Einträge dabei, da müsste ich mich jetzt gleich nochmal genauer mit Befassen.

06:21 **Proband*in:** Okay, schon so ein paar Schlagworte wie Root, einiges an Metadaten, die hier aufgeführt werden. Aber müsste man sich dann gleich mal angucken.

06:31 **Proband*in:** File Explorer, ich spiele jetzt mal ein bisschen rum. Okay, es scheint eine Übersicht zu sein über die Files, die in dem Crate sind. Das ist auch eigentlich recht intuitiv.

06:40 **Proband*in:** Eine Graphenstruktur. Ja, okay, das ist natürlich cool. Ja, da kriegt man wahrscheinlich gleich mal eben den großen Überblick, wenn man dann genauer gucken müsste, wie das jetzt hier aufgebaut ist.

06:51 **Proband*in:** Okay, da werden die Dateien referenziert, beziehungsweise verknüpft. Und dann eben auch noch entsprechende Entitäten, also Personen in dem Fall.

07:02 **Proband*in:** Das hast du dir wahrscheinlich nach einem bestimmten Schema überlegt, dass halt solche Entitäten, die auch vielleicht mit einer PID ausgestattet sind, hier noch weiter verlinkt werden.

07:15 **Proband*in:** Weil ich weiß es nicht, ob es noch andere Metadaten gibt, die man theoretisch hier verlinken könnte, die jetzt nicht irgendwie da sind.

07:21 **Christopher Raquet:** Das ist quasi einfach nur das Link-Data-Prinzip, das diese Metadata-Datei hat. Da sind alle Entities, die in diesem Crate in dieser Metadaten-Datei sind, die liegen quasi in einem flachen Array.

07:33 **Christopher Raquet:** Die werden dann einfach per ID untereinander referenziert. Und genau diese Referenzen zwischen den Entities sind dann hier als Kanten dargestellt.

07:40 **Proband*in:** Ja, okay, ja, macht absolut Sinn und ist auch übersichtlich, gefällt mir ziemlich gut.

07:47 **Proband*in:** Editor, genau, ist halt JSON-LD-Format.

07:49 **Proband*in:** Ja, ich denke für denjenigen, der sich damit auskennt, ist es auch sehr nützlich.

07:55 **Proband*in:** Und Context, okay, das ist dann einfach nur die Spezifikation, Example-Property.

08:06 **Proband*in:** Okay, ja, also auch hier wieder schön die Struktur dargestellt. Was kann ich machen? Sehr intuitiv.

08:15 **Proband*in:** Gut hier oben, das ändert sich dann auch ein bisschen. Da kriege ich hier die Möglichkeit, noch den Editor einzustellen oder das Crate.

08:23 **Proband*in:** Ja, also einige Funktionalitäten, die man sich halt dann im Detail anschauen muss.

08:28 **Proband*in:** Weiß jetzt nicht, wenn ich hier drauf gehe, fällt das eine weg, das hier oben bleibt immer gleich. Also das ist so eine Art globale Funktion oder sowas. Und hier kommt dann auch die Entität dazu.

08:39 **Proband*in:** Okay, ja, das ist super.

08:43 **Christopher Raquet:** Dankeschön.

08:44 **Christopher Raquet:** Dann machen wir direkt weiter.

08:46 **Christopher Raquet:** Das Crate enthält bereits den Großteil der nötigen Daten. Dazu gehören Forschungsberichte, die als Data Entities hinterlegt sind und Informationen zu Autoren und Organisationen, die als Contextual Entities hinterlegt sind.

08:57 **Christopher Raquet:** Ja, den Überblick haben wir gemacht.

08:59 **Christopher Raquet:** Anschließend fällt ihr auf, dass deinem Kollegen in der Data Entity Example-Image einige Fehler unterlaufen sind. Korrigiere die Eigenschaften Caption und Copyright Year. Die richtigen Werte können aus dem zugehörigen Bild entnommen werden.

09:12 **Proband*in:** Ah, okay, genau.

09:14 **Proband*in:** Dann gucke ich erstmal hier. Data Entity haben wir Example-Image gesagt.

09:18 **Proband*in:** Das habe ich dann auch relativ schnell jetzt gefunden. Und dann sehe ich hier schon, Caption scheint falsch zu sein.

09:25 **Proband*in:** Und was hatten wir noch, Copyright Year.

09:27 **Proband*in:** Ist hier unten.

09:28 **Proband*in:** Okay, das findet man alles relativ schnell.

09:30 **Proband*in:** Jetzt muss ich erstmal, okay, irgendwo muss ich jetzt natürlich die Bearbeitung finden, bzw. die Möglichkeit, das zu bearbeiten.

09:44 **Proband*in:** Vielleicht hier, „Show in File Explorer“, „Revert Changes“, oder kann ich das vielleicht sogar direkt reinschreiben? Hier irgendwie anklicken, ne kann ich nicht.

09:52 **Proband*in:** Jetzt gucke ich gerade nochmal.

09:54 **Proband*in:** Also hier sind ja irgendwelche Dateien. Okay, für Images, wo ich auch die Informationen herausziehen soll.

10:05 **Proband*in:** Aus dem dazugehörigen Bild. Kann ich das hier irgendwie anklicken? Ich kann es kopieren. Dann kriege ich aber nur an der Stelle wahrscheinlich den Namen oder so.

10:19 **Proband*in:** Jetzt müsste ich gucken, wie kann ich auf das Bild zugreifen. „Another Entry“.

10:41 **Proband*in:** Also ich frage mich halt gerade, ist das Bild, das ist jetzt schon nochmal so eine Art JPG oder so was. Das müsste ich jetzt anzeigen können irgendwo nehme ich mal an.

10:52 **Proband*in:** Preview File vielleicht.

10:54 **Proband*in:** Okay. Ah ja, Okay.

11:00 **Proband*in:** Das ist jetzt das Bild, von dem wir wahrscheinlich sprechen, oder?

11:03 **Christopher Raquet:** Richtig.

11:04 **Proband*in:** Gut.

11:05 **Proband*in:** Dann Caption und Copyright Year. Okay, das ist relativ eindeutig. Das ist das Copyright Year.

11:11 **Proband*in:** Und Caption ist dann „Caption for this Object, for downloadable machine formats, close Caption subtitles, media object".

11:23 **Proband*in:** Okay, „an image showing some hmm-hmm in a laboratory".

11:26 **Proband*in:** Ah, okay.

11:29 **Proband*in:** Laborsymbolbild, okay.

11:31 **Proband*in:** Na ja, gut.

11:32 **Proband*in:** Ah, jetzt kann ich hier auch gleich editieren. Jetzt kann ich mir hier irgendwas eintragen. Showing some...

11:38 **Proband*in:** Was würde man da jetzt machen?

11:43 **Proband*in:** Chemicals oder was? Also da gab es jetzt nichts Konkretes, was da tut.

11:49 **Proband*in:** So, Copyright hier haben wir gesagt, 2024.

11:52 **Proband*in:** Gut.

11:53 **Proband*in:** Und jetzt mache ich einfach save, nehme ich mal an. Okay.

11:57 **Christopher Raquet:** Gut. Wunderbar.

11:59 **Proband*in:** Ja. Also ich denke, das war zu einer Ordnung. Ich musste kurz irgendwie suchen, wo ich dieses Bild finde, aber nach etwas längerem Schauen, war das dann eigentlich relativ, also nicht mal nach längerem Schauen, man ist relativ schnell drauf gekommen, würde ich sagen.

12:15 **Christopher Raquet:** Ja, war perfekt.

12:17 **Christopher Raquet:** Anschließend möchtest du zu den Data Entities Lab Report (Part A) und Lab Report (Part B) wichtige Informationen hinzufügen. Du solltest angeben, dass diese Dateien am KIT Campus Nord erstellt wurden. Eine passende Eigenschaft hierfür heißt Location Created. Füge die Eigenschaft Location Created zu den Data Entities Lab Report (Part A) und Lab Report (Part B) hinzu. Setze sie auf KIT Campus Nord. Die Contextual Entity für KIT Campus Nord existiert bereits.

12:46 **Proband*in:** Okay, dann fange ich mal hier an.

12:49 **Proband*in:** Jetzt sehe ich auch genau, man kann hier das alles schön aufrufen. Es wird direkt hier reingemacht.

12:55 **Proband*in:** Jetzt nehme ich mal alles, was ich brauche. Das ist wahrscheinlich das hier. Place KIT Campus Nord.

13:04 **Proband*in:** Jetzt soll ich etwas hinzufügen, dann kann ich wahrscheinlich Add Property machen.

13:08 **Proband*in:** Aha, dann kann ich hier auch sogar schon aus einer kleinen Historie aussuchen.

13:13 **Proband*in:** Schauen wir mal, Location Created war es. Kann ich das wahrscheinlich hier suchen, perfekt, Location Created ist da. Okay, the Location, bla bla bla.

13:27 **Proband*in:** Ich muss schauen, was ich da brauche. Ja, ist das jetzt der Identifier, den ich hier eintragen soll in Location Created?

13:45 **Christopher Raquet:** Probier einfach mal weiter aus.

13:48 **Christopher Raquet:** Du musst auf jeden Fall nicht auf einem schlechten Weg.

13:51 **Proband*in:** Okay, Location, ich nehme jetzt einfach mal das hier.

13:54 **Proband*in:** Okay, ich mache jetzt mal Create.

13:58 **Proband*in:** Physical Location. Okay, Place anscheinend.

14:02 **Proband*in:** Entity Name. Kann ich wahrscheinlich selber auswählen. KIT CN mache jetzt einfach mal.

14:10 **Proband*in:** Oder ist das, ah ne Quatsch, das ist dann... Aha. Kann ich das damit machen?

14:19 **Proband*in:** Wenn ich jetzt Create mache, kriege ich da dann einen entsprechenden Eintrag. Okay, also das kann ich wahrscheinlich frei entscheiden, wie ich das benenne.

14:30 **Proband*in:** Wobei dann kriege ich jede neue... Ach so, nee Quatsch, ich verstehe, das hat jetzt eine neue gemacht.

14:35 **Proband*in:** Also ich soll es quasi wirklich auf den Namen machen. Okay, dann gucke ich nochmal kurz. Ich mache...

14:47 **Proband*in:** Wenn ich Unlink mache, kann ich es bearbeiten?

14:55 **Proband*in:** Was ist das hier?

14:57 **Proband*in:** Campus Nord Place.

14:59 **Proband*in:** Aha, jetzt habe ich es, glaube ich, auf den richtigen gesetzt. Okay, okay, verstehe.

15:04 **Proband*in:** Ich dachte irgendwie, die ID ist vielleicht einmal... ist vielleicht sozusagen das... einzigartige und den Namen kann man dann vielleicht noch irgendwie selbst wählen, aber dann ist das in Ordnung.

15:16 **Proband*in:** Da mache ich hier das Gleiche.

15:19 **Proband*in:** Okay, schon created. So create, place.

15:26 **Christopher Raquet:** Das ist jetzt leider wieder nicht ganz der richtige Weg.

15:29 **Proband*in:** Ach so, okay, dann würde man jetzt hier quasi... Kann ich Select machen?

15:34 **Proband*in:** Ah, ja, es ist einfach nur Select, so herum, genau.

15:37 **Christopher Raquet:** Genau, ja.

15:39 **Proband*in:** Jetzt hat das schon automatisch gemacht, oder?

15:41 **Proband*in:** Ja, okay.

15:43 **Christopher Raquet:** Das Create steht ja immer dafür eine neue Entity zu erstellen, die in dieses Feld soll.

15:46 **Proband*in:** Okay.

15:47 **Christopher Raquet:** Und Select, um andere zu wählen.

15:49 **Proband*in:** Die schon da ist, ja.

15:51 **Christopher Raquet:** War dieses mit dem Create und Select nicht ganz klar, oder hast du den Knopf einfach überlesen, übersehen?

15:56 **Proband*in:** Also, ich habe es... Anfang war es mir nicht so ganz klar. Ich dachte eigentlich, dass ich create mache und dann irgendwie...

16:04 **Proband*in:** Also, dass ich ein create für diesen Eintrag jetzt quasi mache und dass es create sich jetzt nur auf dieses Property quasi bezieht.

16:10 **Proband*in:** Aber es hat sich quasi darauf bezogen, einen kompletten neuen Eintrag anzulegen.

16:14 **Christopher Raquet:** Genau, der neuen Place.

16:16 **Proband*in:** Ja, okay. Haben vielleicht andere dann direkt verstanden, oder? Das kann ja sein.

16:22 **Christopher Raquet:** Das steht ja nicht eindeutig da, das steht ja einfach nur create.

16:24 **Proband*in:** Genau, genau. Also, das war jetzt irgendwie das erste, wo ich geklickt habe, das Select. Ja, also...

16:30 **Proband*in:** Vielleicht, wenn man irgendwie im Zweifel noch ein kleines Pop-up oder so hat, was dann sagt, hier leg eine neue Entity für irgendwas an, dann ist es klar, dass man eigentlich was heraussuchen will, was schon existiert, wenn man weiß, dass es existiert.

16:44 **Christopher Raquet:** Ja, genau. Perfekt.

16:48 **Christopher Raquet:** Dann vierte Aufgabe.

16:50 **Christopher Raquet:** Dir fällt auf, dass in der Contextual Entity zu Björn Grüning private Daten hinterlegt sind. Vor der Veröffentlichung möchtest du diese natürlich löschen. Entferne die Eigenschaft für das Geburtsdatum vor der Entity Björn Grüning.

17:06 **Proband*in:** Birth Date...

17:08 **Proband*in:** Also, Delete. Gut. Das war es wahrscheinlich schon, oder?

17:14 **Christopher Raquet:** Ja, genau.

17:16 **Proband*in:** Ja, also das ist natürlich sehr straight forward.

17:20 **Christopher Raquet:** Da haben Leute schon länger für gebraucht.

17:22 **Proband*in:** Ach so. *lacht*

17:24 **Christopher Raquet:** Wunderbar.

17:26 **Christopher Raquet:** Kurz, bevor du dich in den wohlverdienten Feierabend verabschieden wolltest, kommt dein Vorgesetzter mit einem neuen Forschungsbericht. Er hat diesen in Form einer PDF-Datei auf deinem Schreibtisch abgelegt. Du solltest den neuen Forschungsbericht zum Crate hinzufügen. Erstelle eine neue Entity für den Laborbericht. Die zugehörige Datei findest du als „neuer Forschungsbericht.pdf“ auf dem Schreibtisch. Der richtige Typ für eine Entity, die eine Datei beschreibt, ist File.

17:50 **Proband*in:** Da muss ich wahrscheinlich hier oben auf new Entity gehen.

17:52 **Proband*in:** File.

17:54 **Proband*in:** Perfekt.

17:56 **Proband*in:** Das ist dann... or without File. Upload File?

18:00 **Proband*in:** Ah, ich könnte jetzt auch ohne ein File nehmen. Okay, interessant.

18:04 **Proband*in:** Erst Select File, gut. Klar. Forschungsbericht.

18:10 **Proband*in:** So, jetzt haben wir einen Pfad. Den lasse ich mal so.

18:14 **Proband*in:** Dann mache ich einfach hier Create. Und dann haben wir das, glaube ich, auch schon.

18:18 **Christopher Raquet:** Wunderbar.

18:20 **Proband*in:** Super.

18:22 **Christopher Raquet:** Dankeschön. Okay, das war jetzt irgendwie zu schnell.

18:26 **Christopher Raquet:** *lachen*

18:26 **Christopher Raquet:** So. Um die Korrektheit des Crates ein letztes Mal zu überprüfen, möchtest du dir im Graphen einen Überblick verschaffen. Dort fällt dir auf, dass du für den neu hochgeladenen Forschungsbericht noch keinen Autor hinterlegt hast.

18:40 **Christopher Raquet:** Erstelle im Graphen eine neue Kante, ausgehend vom neuen Forschungsbericht. Das Ende der Kante soll eine beliebige Person sein. Wähle Author als Eigenschaft für die Kante.

18:48 **Proband*in:** Das haben wir schon gesehen, der Graph.

18:52 **Proband*in:** Und dann kommt der Forschungsbericht. So, wahrscheinlich.

18:56 **Proband*in:** Jetzt hier New Entity machen. Person, würde ich mal sagen. Irgendeine Bestehende.

19:02 **Proband*in:** Ne, jetzt frage ich mich, gerade ist das jetzt wieder das gleiche, was wir vorhin hatten.

19:10 **Proband*in:** Aber da steht schon Create a New Entity.

19:16 **Proband*in:** Eine neue Kante ausgehend vom neuen Forschungsbericht. Das Ende der Kante soll eine beliebige Person sein.

19:20 **Proband*in:** Okay. Es soll eine beliebige Person sein.

19:28 **Proband*in:** So, import from ORCID. Nur so Interessehalber. Ah, okay, du kannst das direkt so machen.

19:34 **Proband*in:** Ja, aber jetzt also jetzt habe ich ja an dem Punkt noch keinen.

19:40 **Proband*in:** Oder soll ich jetzt irgendein noch da reinmachen? Ich habe jetzt halt keine ORCID zur Hand.

19:46 **Proband*in:** Also... Test. Aber jetzt mache ich das einmal ohne eine richtige ORCID sozusagen.

19:52 **Proband*in:** Also es war schon der Gedanke, dass man jetzt einen neuen erstellt und nicht zu einem der Bestehenden verlinkt.

19:58 **Christopher Raquet:** Das ist völlig freigestellt.

20:00 **Proband*in:** Genau, aber jetzt ist die Verbindung hier irgendwie nicht direkt ersichtlich.

20:06 **Christopher Raquet:** Ich würde noch kurz eine Korrektur einwerfen. Es geht um den neuen Forschungsbericht.

20:12 **Proband*in:** Ach, das stimmt. Das ist ja gar nicht der neue Forschungsbericht. Ah, warte mal.

20:18 **Christopher Raquet:** Die sind sehr ähnlich benannt.

20:20 **Proband*in:** Ja. Das habe ich gelesen. Kann ich das so machen?

20:26 **Proband*in:** Das weiß ich jetzt gar nicht.

20:28 **Proband*in:** So. Select a property. Steht das hier drin? Welches ist das? Einfach Autor dann.

20:38 **Proband*in:** Wärs das? Also wärs das jetzt schon?

20:42 **Christopher Raquet:** Ja. Das ist die Aufgabe. Super. Wunderbar.

20:48 **Proband*in:** Ich speichere mal noch alles.

20:50 **Proband*in:** Ja.

20:52 **Christopher Raquet:** Damit warst du wahrscheinlich der schnellste heute.

20:54 **Proband*in:** Echt? Okay.

20:56 **Christopher Raquet:** Also gut, nenne mir gerne noch ein paar Gedanken, die dir im Kopf umschwirren.

21:00 **Proband*in:** Ja.

21:02 **Proband*in:** Also es hat mir sehr gut gefallen vom Handling her, sehr übersichtlich.

21:06 **Proband*in:** Da wo ich jetzt eben die Funktionalität erwartet hätte, waren sie in der Regel auch.

21:12 **Proband*in:** Ich weiß nicht, ob andere, das weiß ich nicht, da vielleicht andere Interfaces gewohnt sind oder so.

21:20 **Proband*in:** Ich fand es schon ziemlich intuitiv. Genau.

21:24 **Proband*in:** Vielleicht bei dem ein oder anderen könnte man, wenn man noch Lust hat, so kleinere Pop-ups als Beschreibung machen.

21:30 **Proband*in:** Also wenn man jetzt länger mit der Maus darauf bleibt, dass man sagt, okay man, kann da noch ein bisschen was draus ziehen.

21:36 **Proband*in:** Ansonsten. Gefällt mir.

[...]

27:32 **Proband*in:** Ja, gut. Dann würde ich sagen, danke ich für die Möglichkeit

27:38 **Proband*in:** Hat Spaß gemacht.

A.3.5. Transkript eines Nutzertests (5)

00:00 **Christopher Raquet:** Dann habe ich erst ein paar allgemeine Fragen zu deiner Person. Was hast du studiert oder gearbeitet, bevor du zu DEM gekommen bist?

00:05 **Proband*in:** Ich habe Germanistik und Informatik studiert.

00:08 **Christopher Raquet:** Parallel?

00:09 **Proband*in:** Nacheinander. Erst Germanistik, dann Informatik.

00:12 **Christopher Raquet:** Heißt, du hast jetzt einen Doppelbachelor oder ...?

00:15 **Proband*in:** Ich habe Master Germanistik und Bachelor Informatik.

00:18 **Christopher Raquet:** Interessant. Was machst du im DEM?

00:20 **Proband*in:** Ich habe hier promoviert und seither arbeite ich in meiner Post-Doc-Phase hier im Bereich Digital Humanities.

00:28 **Christopher Raquet:** Was macht man so im Bereich der Digital Humanities? Ist das dann eher Verwaltungsarbeit oder ...?

00:33 **Proband*in:** Nein, das ist Forschung und Entwicklung.

00:35 **Christopher Raquet:** Entwicklung von Tools zur Hilfe bei Digital Humanities?

00:40 **Proband*in:** Genau.

00:40 **Christopher Raquet:** Hast du denn in dem Zug schon mal mit Research Object Crates gearbeitet?

00:46 **Proband*in:** Tatsächlich noch nie, ne.

00:47 **Christopher Raquet:** Ok, dann werde ich auch noch kurz einführen, was Research Object Crates sind. Dafür habe ich diese nützliche Grafik.

00:56 **Christopher Raquet:** RO-Crates oder auch Research Object Crates genannt, sind eine Methode um beliebige Daten, meistens Forschungsdaten, zusammen mit ihren Metadaten zu paketieren, also zu einer ZIP-Datei zu machen.

01:07 **Christopher Raquet:** Ein RO-Crate sieht dabei folgendermaßen aus. Hier oben, da ist das Crate selbst dargestellt, das ist einfach die ZIP-Datei an sich. Da drin sind dann enthalten alle Dateien und Ordner, die man eben paketieren möchte. Und daneben liegt die RO-Crate Metadata-Datei, wo die Metadaten zu diesen Dateien drin liegen.

01:27 **Christopher Raquet:** Jede Data Entity beschreibt dann immer genau eine Datei, in diesen Crate oder einen Ordner, eben mit allen Metadaten, die zu dieser Datei einem Einfallen oder die man Angeben will.

01:39 **Christopher Raquet:** Dann gibt es noch Contextual Entities, die es erlauben, Kontextinformationen zu geben. Zum Beispiel kann man dann eine ganze Entität für einen Autor anlegen, um noch Metadaten zu dem Autor zu geben.

01:50 **Christopher Raquet:** Und dann kann man noch für den Autor angeben, wo er arbeitet und zu seinem Arbeitsplatz eine Contextual Entity anlegen, wo alle Metadaten drin stehen.

02:01 **Proband*in:** Aber gibt es da eine 1-zu-1 Beziehung zu allen Sachen, also ist es verpflichtend für alle Sachen, die paketiert sind, auch die Data Entities ...?

02:08 **Christopher Raquet:** Ne, also es kann höchstens eine Data Entity pro Datei geben, sonst ist es ja...

02:13 **Proband*in:** Das ist das obere Limit, aber es ist nicht verpflichtend.

02:15 **Christopher Raquet:** Genau, also Dateien, die hier drin sind, müssen nicht beschrieben sein und es kann Data Entities geben für URLs zum Beispiel, also für Daten auf einem Server oder so.

02:24 **Christopher Raquet:** Genau, dann gibt es noch die Root Entity, das ist eine besondere Data Entity, die

02:30 **Christopher Raquet:** einfach das Crate selbst beschreibt. Daher der Name. Und die Metadata Entity ist auch eine besondere Data Entity, die diese Metadaten-Datei betreibt.

02:38 **Christopher Raquet:** Die ist aber nicht weiter relevant.

02:39 **Proband*in:** Okay.

02:40 **Christopher Raquet:** Genau, zusammengefasst sind RO-Crates ein Maschinenfreundliches Format zum Paketieren von Daten zusammen mit ihren Metadaten.

02:47 **Christopher Raquet:** Im heutigen Nutzertest, befassen wir uns mit einem Editor, der RO-Crates erstellen und bearbeiten kann, in diesem Fall ist aber ausschließlich das Bearbeiten der Metadaten gemeint.

02:58 **Christopher Raquet:** Gut, bis hier hin Fragen zu RO-Crates?

03:00 **Proband*in:** Glaube nicht.

03:02 **Christopher Raquet:** Schauen wir mal, was sich ergibt.

03:04 **Christopher Raquet:** Als Nächstes machen wir mit spezifischeren Aufgaben weiter.

03:07 **Christopher Raquet:** Ich werde dir jede Aufgabe vorlesen und auch als kleinen Zettel für dich hinlegen.

03:11 **Christopher Raquet:** Nur als Erinnerung sagen wir bitte alles, was du denkst, was dir auffällt, negativ oder positiv.

03:16 **Christopher Raquet:** Danke fürs mitmachen!

03:23 **Christopher Raquet:** So, jetzt darfst du übernehmen.

03:24 **Proband*in:** Ja, erst du fragst, muss ich die Tastatur benutzen?

03:26 **Christopher Raquet:** Ja.

03:27 **Proband*in:** Werde ich die benutzen müssen? Eine Mac Tastatur und ich...

03:28 **Christopher Raquet:** Wirst du leider müssen, ja.

03:30 **Proband*in:** Dann liegt es nicht an deiner Usability, wenn ich irgendwas auf der Tastatur nicht hin kriege.

03:35 **Christopher Raquet:** Nur Buchstaben. Keine Sondertasten.

03:37 **Proband*in:** Sehr gut.

03:38 **Christopher Raquet:** Also dann, du arbeitest als Teil einer Forschungsgruppe, die sich mit modernen Forschungsthemen befasst. Deine Aufgabe ist es, die Forschungsergebnisse deiner Gruppe zu veröffentlichen.

03:47 **Christopher Raquet:** Um die maschinelle Verarbeitung der Daten zu ermöglichen, entschied sich deine Forschungsgruppe Research Object Crates zu verwenden. Die nächste Veröffentlichung von Forschungsdaten steht kurz bevor.

03:57 **Christopher Raquet:** Ein Kollege hat bereits den Großteil vorbereitet und ein Crate erstellt. Du solltest noch einige Details am Crate anpassen.

04:04 **Christopher Raquet:** Um das Crate bearbeiten zu können, musst du es zunächst in den Editor für Research Object Crates namens NovaCrate importieren. Verschaffe dir einen Überblick über die Startseite des Editors und importiere anschließend das vorbereitete Crate. Du findest die ZIP-Datei auf dem Schreibtisch, also dem Mac-Schreibtisch.

04:22 **Proband*in:** Okay.

04:23 **Christopher Raquet:** Bitteschön.

04:24 **Proband*in:** Ja.

04:25 **Christopher Raquet:** Du darfst jetzt gerne einfach mal anfangen zu beschreiben, was du siehst, was du wo für Funktionalitäten vermutest und dann einfach mit der Aufgabe loslegen.

04:32 **Proband*in:** Ja, genau.

04:33 **Proband*in:** Sehe ja zum ersten Mal den Editor.

04:36 **Proband*in:** Dark Mode ist das allererste, was mir auffällt.

04:38 **Christopher Raquet:** Gut oder schlecht?

04:40 **Proband*in:** Tatsächlich wäre es jetzt nicht meine erste Wahl gewesen, es wäre auch der erste Button, den ich benutzen würde, wer tatsächlich der, für "Toggle Theme".

04:46 **Christopher Raquet:** Ja, gerne. Mach das gerne.

04:48 **Proband*in:** Ja, schön. Kein Dark Mode.

04:50 **Proband*in:** Muss ein bisschen Abwechslung haben, es ist ja Tageslicht.

04:55 **Proband*in:** Gut, genau. Und jetzt muss ich mir erst mal orientieren.

04:57 **Proband*in:** Also der Button ist tatsächlich direkt aufgefallen und ich wusste, wo ich hinklicken muss, das fand ich schon mal gut, weil dann kann ich es schon mal auf mich direkt anpassen.

05:03 **Proband*in:** Und jetzt kann ich mir mal orientieren, was hier so ist.

05:06 **Proband*in:** Genau, so links irgendein Hauptmenü offensichtlich, was Funktionalitäten bereitstellt, wir haben offensichtlich rechts irgendeine Einteilung mit so Cards, die oben sind, wo ich mir gleich mal angucken muss, was die tun und unten irgendwie wahrscheinlich eine Art Arbeitsbereich, wäre so meine Vermutung.

05:21 **Proband*in:** Genau, sonst ist ja noch nicht so viel drin.

05:26 **Proband*in:** Es ist jetzt tatsächlich am ersten Blick nicht erkennbar, in welcher hierarchischen Abfolge sozusagen das hier links und das da oben steht.

05:35 **Proband*in:** Also ob das irgendwie übergeordnet ist gegenüber dem oben oder ob es vergleichbare Funktionalitäten beinhaltet.

05:43 **Proband*in:** Ich versuche mal ein bisschen durchzulesen.

05:45 **Christopher Raquet:** Und du kannst gerne auch schon mit der Aufgabe in dem Zuge dann anfangen.

05:51 **Proband*in:** Ja.

05:52 **Proband*in:** Okay, also ich meine, wenn Import ist, das Buzz Word Import, habe ich schon gefunden.

05:57 **Proband*in:** So, genau, Überblick verschafft.

06:02 **Proband*in:** Ich soll was importieren, klicken wir doch mal auf Import.

06:05 **Proband*in:** Das ist der Schreibtisch, das ist gut, dann entsteht schon mal keine Verwirrung.

06:11 **Proband*in:** Und ich soll einen vorbereiteten Crate importieren.

06:16 **Proband*in:** Was ist jetzt der vorbereitete Crate?

06:19 **Proband*in:** Da es nur eine ZIP-Datei hier gibt, gehe ich mal davon aus, was es die ist.

06:24 **Proband*in:** Ansonsten könnte ich es jetzt tatsächlich mir nicht erschließen.

06:27 **Proband*in:** Also es heißt jetzt ja nicht Crate oder irgendwas.

06:29 **Proband*in:** Gehen wir mal davon aus der Kollege meinte das hier.

06:32 **Christopher Raquet:** Wunderbar. Das war schon die erste Aufgabe. Mit Bravour bestanden!

06:36 **Proband*in:** Juhu! Ich werde nicht getestet, habe ich gehört?

06:39 **Christopher Raquet:** Ah stimmt, dann hat der Editor mit Bravour bestanden.

06:41 **Proband*in:** Okay, darauf können wir uns einigen.

06:44 **Christopher Raquet:** Das Crate enthält bereits den Großteil der nötigen Daten. Dazu gehören Forschungsberichte, die als Data Entities hinterlegt sind und Informationen zu Autoren und Organisationen, die als Contextual Entities hinterlegt sind.

06:55 **Christopher Raquet:** Verschaffen wir dir einen kurzen Überblick über den Editor. Du musst den Inhalt des Crates selbst im Moment nicht weiter beachten.

07:01 **Christopher Raquet:** Ich lese direkt auf den nächsten Teil vor.

07:03 **Christopher Raquet:** Anschließend fällt dir auf, dass deinem Kollegen in der Data Entity Ëxample Imageëinige Fehler unterlaufen sind.

07:08 **Christopher Raquet:** Korrigiere die Eigenschaften Caption und Copyright Year. Die richtigen Werte können aus dem zugehörigen Bild entnommen werden.

07:15 **Proband*in:** So, okay.

07:17 **Proband*in:** Lass mich die Aufgaben nochmal kurz einmal lesen.

07:20 **Christopher Raquet:** Mhm.

07:22 **Proband*in:** Okay.

07:33 **Proband*in:** Also, ich habe irgendetwas mit einem Ëxample Image"gehört, in dem mir offensichtlich irgendwas auffallen muss.

07:39 **Proband*in:** Also muss ich da offensichtlich mal hinklicken. Sonst wird es schwierig.

07:43 **Proband*in:** Scrollrichtung falsch herum, großartig, hat auch nichts mit deinem Editor zu tun.

07:48 **Christopher Raquet:** Wir merken, Mac-Usability ist ganz schlecht.

07:50 **Proband*in:** Ja, wenn man halt alles umgestellt hat an seinem eigenen Mac, dann ganz schlecht.

07:54 **Proband*in:** So, und jetzt fallen mir hier natürlich sofort Dinge auf, habe ich gehört. Und zwar bei den Eigenschaften Caption und Copyright Year.

08:03 **Proband*in:** Okay, also Caption ist offensichtlich hier.

08:06 **Proband*in:** Copyright Year ist offensichtlich hier.

08:09 **Proband*in:** So, dann gehen wir mal auf Fehlersuche. So, ich habe das. Und was ist das zugehörige Bild, das du hier erwähnst?

08:19 **Christopher Raquet:** Das zugehörige Bild gehört zu der Data Entity, die du momentan geöffnet hast.

08:24 **Proband*in:** Ah, okay, ich verstehe.

08:29 **Proband*in:** Okay, das ist offensichtlich nicht, wie ich zu dem Bild komme, aber ich muss mir offensichtlich irgendwie das Bild angucken, so wie ich die Aufgabe verstehe.

08:36 **Christopher Raquet:** Mhm.

08:40 **Proband*in:** Ich würde das nächste Mal mit dem Preview File probieren. Das ist ziemlich zielführend. Richtiges Rätsel hier.

08:49 **Proband*in:** Okay, also, dann müssen wir mal die Arbeit hier korrigieren.

08:58 **Proband*in:** Ne, was soll das sein? Ach so, das ist nicht das gemeint, sondern...

09:06 **Christopher Raquet:** Da muss jetzt nicht sinnvolles stehen.

09:10 **Proband*in:** Ich möchte schon gerne was Sinnvolles eintragen, wenn ich hier schon am Korrigieren bin. Oh, Englisch ist heute ganz toll.

09:24 **Proband*in:** Mir war es gerade rein geschwirrt, aber egal. Wie heißt denn das? Machen wir mal das, es muss ja nichts Schönes drinstehen, und da ist offensichtlich irgendetwas falsch, haben wir gesagt.

09:42 **Proband*in:** Vielleicht auch mal hier, nee, kann ich nicht.

09:45 **Proband*in:** Also, ich hatte jetzt irgendwie gehofft, dass ich mir das Ganze irgendwie quasi als sozusagen ausgeklappt angucken kann, dadurch, dass es ja relativ lang ist.

09:52 **Proband*in:** Das heißt, es ist jetzt für mich relativ unpraktisch, dass wenn ich jetzt irgendwas an dem Wert tun oder machen möchte, dass ich jetzt quasi hin und her scrollen muss.

10:00 **Proband*in:** Also, dass ich das irgendwie sozusagen ausklappen könnte, wenn das tatsächlich praktisch wäre.

10:04 **Proband*in:** So, aber gut, also ich sollte die Fragezeichen korrigieren. Habe ich das jetzt richtig verstanden?

10:08 **Christopher Raquet:** Genau, das war schon das Richtige.

10:10 **Proband*in:** Der Rest scheint ja auch zu stimmen, 2024 und so weiter.

10:13 **Proband*in:** So, und dann haben wir noch Copyright hier. Ja, das haben wir.

10:18 **Proband*in:** Ich war irgendwie gedanklich schon beim Nächsten.

10:23 **Christopher Raquet:** Wunderbar, das war schon die Aufgabe.

10:25 **Christopher Raquet:** Dann, weiter geht es.

10:29 **Christopher Raquet:** Anschließend möchtest du zu den Data Entities Lab Report Part A und Lab Report Part B wichtige Informationen hinzufügen.

10:36 **Christopher Raquet:** Du sollst angeben, dass diese Dateien am KIT Campus Nord erstellt wurden. Eine passende Eigenschaft hierfür heißt Location Created.

10:43 **Christopher Raquet:** Füge die Eigenschaft Location Created zu den Data Entities Lab Report Part A und Lab Report Part B hinzu. Setze sie auf KIT Campus Nord.

10:52 **Christopher Raquet:** Die Entity für KIT Campus Nord existiert bereits.

10:56 **Proband*in:** So, eine Frage, die ich mir jetzt gerade zwischendurch zwischen den Aufgaben stelle, ob ich speichern muss, bevor ich zu etwas anderem weiterkomme oder ob ich es irgendwann gesamt speichern kann.

11:06 **Proband*in:** Also jetzt, ich weiß es für mich, ist jetzt nicht wichtig, aber wenn ich jetzt quasi daran arbeiten würde, dann wäre jetzt halt die Frage, wann muss ich speichern.

11:13 **Proband*in:** Ich sehe ja hier schön, das ist ganz praktisch, dass ich offensichtlich, ich nehme an das soll, heißen das da umgespeicherte Dinge sind.

11:19 **Proband*in:** Das finde ich ganz gut.

11:20 **Proband*in:** Ich würde es jetzt einfach mal ohne Speichern probieren und gucken, ob ich später speichern kann.

11:24 **Proband*in:** So, ich sollte was hinzufügen.

11:31 **Proband*in:** Wird das wohl hier sein?

11:35 **Proband*in:** Dann werde ich danach wohl mal suchen müssen. So, ich habe eine Location created. Das klingt zielführend.

11:46 **Proband*in:** So, und jetzt muss ich die irgendwie noch auf das Setzen, was ich möchte.

11:55 **Proband*in:** Gut, das ist jetzt der Button.

11:58 **Proband*in:** Ich weiß nicht, ob ich tatsächlich mit diesem Symbol SSelect"verbinde. Möglicherweise ist es korrekt. Ich bin mir gerade nicht sicher. Mich irritiert es ein bisschen.

12:05 **Proband*in:** Ich hätte jetzt gedacht, das ist eine Verlinkung oder irgendwas. Also mir kommt das Symbol an der Stelle ein bisschen komisch vor.

12:11 **Proband*in:** Dann habe ich glaube, ich habe den Campus Nord ausgewählt. Das kann ich jetzt leider nicht mehr sehen. Mit drüber hovern kann ich es sehen.

12:18 **Proband*in:** Also gehe ich mal davon aus, das wird stimmen. Was passiert, wenn ich hier draufklicken, nur der Neugier halber.

12:24 **Proband*in:** Okay, dann kann ich die ganze Entity angucken.

12:28 **Proband*in:** Ich weiß jetzt nicht, ob ich, ah, das ist Tab organisiert.

12:31 **Proband*in:** Das heißt, um da wieder zurückzukommen, kann ich den Tab zu machen.

12:34 **Proband*in:** Das ist auch erstmal nicht intuitiv, aber dann sehr einleuchtend.

12:37 **Proband*in:** So, dann haben wir das für Lab Report A gemacht. Dann machen wir das mal noch für Lab Report B.

12:44 **Proband*in:** Ich glaube, ich sollte für beide das gleiche Tun richtig.

12:47 **Christopher Raquet:** Genau.

12:47 **Proband*in:** Zur Aufgabenstellung darf ich auch Fragen stellen, richtig?

12:49 **Christopher Raquet:** Auf jeden Fall.

12:51 **Proband*in:** So.

12:53 **Christopher Raquet:** Wunderbar

12:55 **Proband*in:** Moment, ich bin einmal kurz neugierig... Nee, da kriege ich das nicht raus. Entschuldigung, darfst gleich weiter machen. Ich wollte nur wissen, was passiert, wenn ich Matching Places ausmache.

13:04 **Proband*in:** Egal, lass das nicht machen.

13:06 **Christopher Raquet:** Das kannst du gerne auch.

13:12 **Christopher Raquet:** Okay, dann kannst du einfach alles auswählen.

13:14 **Proband*in:** Okay, dann sind das nicht mehr nur Places.

13:16 **Proband*in:** Ja, okay, alles klar.

13:18 **Proband*in:** Gut, Neugier befriedigt, danke.

13:20 **Christopher Raquet:** Vielleicht direkt noch zu diesem Symbol.

13:23 **Christopher Raquet:** Ich glaube, damit verfälsche ich nichts, wenn ich das sage. Das habe ich technisch motiviert genommen, weil es ja tatsächlich Verlinkungen sind.

13:29 **Christopher Raquet:** Also in diesen beiden hast du jetzt einfach nur eine Verlinkung auf dieses hier.

13:34 **Proband*in:** Ich verstehe, weil als Select-Button würde ich halt irgendwie ein Select-Symbol, wie auch immer ein Select-Symbol aussehen würde.

13:43 **Christopher Raquet:** Dir fällt auf, dass in der Contextual Entity zu Björn Grüning private Daten hinterlegt sind. Vor der Veröffentlichung möchtest du diese natürlich löschen.

13:51 **Christopher Raquet:** Entferne die Eigenschaft für das Geburtsdatum von der Entity für Björn Grüning.

14:00 **Proband*in:** Dann machen wir das.

14:02 **Proband*in:** So, Geburtsdatum haben wir gesagt.

14:04 **Proband*in:** Das geht natürlich gar nicht.

14:06 **Proband*in:** So, dann machen wir hier.

14:10 **Proband*in:** Das ist jetzt nicht das, was ich erwartet habe.

14:15 **Proband*in:** Ah, "Delete Entry"wäre schlauer gewesen. Aber warum hat Clear jetzt gar nichts gemacht?

14:20 **Christopher Raquet:** Clear hat was gemacht. Es war wohl nur nicht erkennbar.

14:24 **Proband*in:** Genau, also ein erkennbarer Effekt war es jetzt nicht. Aber ich nehme an, ich sollte die löschen, weil da stand auch löschen und nicht irgendwie leer machen oder so.

14:31 **Proband*in:** Okay, ja.

14:35 **Christopher Raquet:** Ja, das mit dem Clear ist tatsächlich...

14:39 **Christopher Raquet:** Also was es gemacht hat, nur falls es dich interessiert.

14:44 **Christopher Raquet:** Clear setzt es auf jetzt.

14:46 **Proband*in:** Ah, das konnte ich jetzt nicht sehen.

14:46 **Christopher Raquet:** Aber das ist, das sieht man natürlich nicht. Man liest das ja nicht nochmal.

14:51 **Proband*in:** Ja, okay.

14:53 **Proband*in:** Und das ist natürlich nicht intuitiv, dass Clear auf heutiges Datum setzt.

14:56 **Christopher Raquet:** Ja, das ist richtig.

14:58 **Christopher Raquet:** Das hat technische Gründe, die mir sehr viele Kopfschmerzen bereitet haben.

15:00 **Proband*in:** Immer mit Datumssachen, immer.

15:02 **Christopher Raquet:** Ja, wirklich so.

15:04 **Proband*in:** Black Magic.

15:06 **Christopher Raquet:** Sobald du dich in den wohlverdienten Feierabend verabschieden wolltest, kommt dein Vorgesetzter mit einem neuen Forschungsbericht.

15:12 **Christopher Raquet:** Er hat diesen in Form einer PDF-Datei auf deinem Schreibtisch abgelegt. Du sollst den neuen Forschungsbericht zum Crate hinzufügen.

15:18 **Christopher Raquet:** Erstelle einen neuen Entity für den Forschungsbericht. Die zugehörige Datei findest du als "neuer Forschungsbericht.pdfäuf dem Schreibtisch. Der richtige Typ für eine Entity, die eine Datei beschreibt, ist File.

15:30 **Proband*in:** Jetzt werden nur die Daumenschrauben hier angezogen bei den Aufgaben.

15:34 **Proband*in:** Okay, neue Entity erstellen.

15:39 **Proband*in:** Ich mache diesen schönen Button dafür.

15:45 **Proband*in:** Offensichtlich brauche ich einen File.

15:50 **Proband*in:** Ich gucke noch mal kurz... ich schau mal drüber, was da sonst noch so ist.

15:55 **Proband*in:** Ïf you want to add a file or folder to the Crate, choose the ...".

16:05 **Proband*in:** So, jetzt liegt er auf dem Schreibtisch.

16:15 **Proband*in:** Das war offensichtlich nicht der Button, den ich wollte.

16:17 **Proband*in:** Das fand ich jetzt sehr unerwartet.

16:21 **Proband*in:** Also, ich verstehe jetzt, beim noch mal hinschauen, dass es ein Toggle ist, aber ich hätte jetzt erwartet, dass ich da was hochladen kann, weil es so funktioniert, glaube ich, immer solche Masken, dass man upload file und dort kriegt man den Dialog fürs File.

16:34 **Proband*in:** So, ich habe das PDF dahin gepackt.

16:37 **Proband*in:** Sonst habe ich, glaube ich, keine weiteren Arbeitsaufträge irgendwas zu tun.

16:42 **Proband*in:** Also würde ich alles andere wahrscheinlich mal lassen. Also, ich weiß nicht, ob ich irgendetwas hier schöner angeben soll.

16:48 **Christopher Raquet:** Laut Aufgabenbestimmung nicht, aber du kannst machen, was du willst.

16:51 **Christopher Raquet:** Du kannst nichts kaputt machen.

16:53 **Proband*in:** Ach, schade eigentlich. Langweilig.

16:57 **Christopher Raquet:** Wunderbar.

16:59 **Proband*in:** Dass da jetzt kein Katzenbild rumlag, das ich stattdessen hätte hochladen können. Das ist auch sehr schade.

17:05 **Christopher Raquet:** Ich habe den Schreibtisch leider aufgeräumt.

17:07 **Christopher Raquet:** Hier wäre auch noch der Nutzertest von Nicolas gewesen, falls du den hochladen willst.

17:11 **Proband*in:** Ja, das hätte ich mal machen können, ja.

17:13 **Christopher Raquet:** Genau, noch zu den Tabs, da sind tatsächlich alle drüber gestolpert.

17:16 **Proband*in:** Ja, okay, ja.

17:18 **Christopher Raquet:** Das ist technisch nicht geglückt, dass das Tabs sein sollten.

17:20 **Proband*in:** Ja, genau. Also ich glaube, wenn dann wirklich auch so als Tabmarkierung würde es dann, glaube ich, wieder gehen, aber dadurch, dass es halt ein Button ist, in dem Moment ist es halt nicht glücklich.

17:30 **Christopher Raquet:** Ich würde mich mal zu deiner Meinung fragen, ob du die Tabs eindeutig findest?

17:32 **Proband*in:** Ja, da war es okay, ja. Also da fand ich es, glaube ich, eben ganz einleuchtend, dass es halt so eine Viewfilter quasi ist.

17:38 **Christopher Raquet:** Ja, okay.

17:40 **Proband*in:** Und dass man dann hin und her gehen kann.

17:42 **Christopher Raquet:** Das habe ich danach gemacht. Und dann rückwirkend aber nicht verbessert.

17:47 **Christopher Raquet:** Um die Korrektheit des Crates ein letztes Mal zu überprüfen, möchtest du dir im Grafen einen Überblick verschaffen.

17:52 **Christopher Raquet:** Dort fällt dir auf, dass du für den neu hochgeladenen Forschungsbericht noch keinen Autor hinterlegt hast.

17:58 **Christopher Raquet:** Erstelle im Grafen eine neue Kante, ausgehend vom neuen Forschungsbericht. Das Ende der Kante soll eine beliebige Person sein. Wähle Author als Eigenschaft für die Kante.

18:08 **Proband*in:** Cool. Graph klingt gut.

18:12 **Proband*in:** Ich wollte nicht doppelklicken übrigens.

18:14 **Proband*in:** So. Jetzt hat er mir die Aufgabe nicht gegeben.

18:18 **Christopher Raquet:** Sorry.

18:20 **Proband*in:** Dankeschön. Muss mich doch mal an irgendwas festhalten können.

18:24 **Proband*in:** So, ich möchte irgendwie zum neuen Forschungsbericht orientieren.

18:28 **Proband*in:** Der Forschungsbericht ist hier.

18:30 **Proband*in:** Der, kann ich zustimmen, dass der keinen Autor als Kante hat.

18:36 **Proband*in:** Ich darf mir jetzt einen ausdenken.

18:40 **Proband*in:** Sehr naheliegend.

18:44 **Proband*in:** Was soll das sein?

18:46 **Proband*in:** Überraschung.

18:48 **Proband*in:** Okay.

18:50 **Christopher Raquet:** Ja, das war's.

18:52 **Proband*in:** Das könnte ich jetzt wahrscheinlich auch noch mehrfach tun. Wäre ziemlich sinnlos wenn ich es nicht mehrfach tun könnte.

18:58 **Proband*in:** Müssen wir noch fair sein und das noch mehrfach verteilen. Okay.

19:04 **Proband*in:** Aber es gäbe bestimmt keine Beschränkung, ... Das dürfte jetzt.

19:07 **Proband*in:** Warum schlägt der mir nichts vor?

19:09 **Christopher Raquet:** Du bist einmal vom Plus und einmal vom Author ausgehend gestartet.

19:14 **Proband*in:** Ah ...

19:18 **Proband*in:** Okay. Verstehe. Wenn es von da ist, dann kann ich, ok. Ich habe beim zweiten Mal, das habe ich beim zweiten Mal gemerkt, ich habe beim zweiten Mal nicht Author genommen und musste das nochmal angeben und ansonsten kann man es immer direkt machen.

19:29 **Proband*in:** Okay. Das heißt aber, wenn man diesen Author benutzt, dann kann es einem sozusagen auch relativ leicht passieren, dass man eine Fehlkante zieht. Weil man halt irgendwas neues Anlegen wollte und dann nicht weiter drüber nachdenkt und dann ist die Kante halt wo sie ist.

19:44 **Christopher Raquet:** Das ist richtig. In Grafen gibt es keine sinnvolle Validierung, zumindest noch nicht.

19:50 **Christopher Raquet:** Also dieser Knopf hier unten, Validation, soll andeuten, dass ich das ursprünglich einbauen wollte.

19:56 **Christopher Raquet:** Das es quasi eine vollständige Validierung vom Crate hat, in der man einen schönen Log hat, was alles nicht so gut aussieht und so was.

20:02 **Christopher Raquet:** Aber das sprengt den Rahmen.

20:04 **Proband*in:** Kann ich mir vorstellen.

20:06 **Christopher Raquet:** Das muss ich leider auch akzeptieren.

20:08 **Christopher Raquet:** Und am Ende alles speichern ist auch kein Problem.

20:10 **Proband*in:** Ja, das habe ich mir auch gedacht.

20:12 **Proband*in:** Wie gesagt, das ist ja auch angezeigt, weil was ungespeichert ist, war dann auch klar, dass man es sicherlich später speichern kann.

20:18 **Proband*in:** Thinking out loud.

20:20 **Christopher Raquet:** Wunderbar.

20:22 **Christopher Raquet:** Anmerkungen, Ideen, Fragen, was dir aufgefallen ist, wie du die Usability bewerten würdest. Einfach alles.

20:30 **Proband*in:** Es ist jetzt sehr fokussiert auf einzelne Aufgaben gewesen.

20:34 **Proband*in:** Die hat man sehr gut gefunden.

20:36 **Proband*in:** Als ich kann jetzt nicht sagen, ob da jetzt drum herum noch viel ist, was ich nicht verstanden hätte.

20:40 **Proband*in:** Das habe ich jetzt sehr gut ausgeblendet, in den Beschreibungen der Aufgaben.

20:44 **Proband*in:** Da kriegt man Tunnelblick.

20:46 **Proband*in:** Gefühlt ist da schon noch sehr viel. Da ist links noch viel, oben noch viel. Aber ansonsten, wie du bemerkt hast, alles zielgerichtet gefunden.

20:54 **Proband*in:** Daher kein Problem in der Hinsicht.

20:56 **Proband*in:** Und das, was wir jetzt schon hatten, die suboptimalen Symbole oder Buttons oder so, sind ja klar geworden.

21:04 **Christopher Raquet:** Cool.

[...]

22:10 **Proband*in:** Ja, aber wie gesagt, war alles gut. Erkennt man gut, unterscheidbar, gut lesbar, Schriften und so weiter passt auch alles, denke ich.

22:18 **Proband*in:** So weit.

22:21 **Proband*in:** Sehr schön. Cool.

22:24 **Christopher Raquet:** Gut, dann sind wir fertig.