

LeoTour

DIPLOMARBEIT

verfasst im Rahmen der

Reife- und Diplomprüfung

an der

Höheren Abteilung für Informatik

Eingereicht von:

Oliver Sugic

Betreuer:

Thomas Stütz

Leonding, April 2023

Ich erkläre an Eides statt, dass ich die vorliegende Diplomarbeit selbstständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen und Hilfsmittel nicht benutzt bzw. die wörtlich oder sinngemäß entnommenen Stellen als solche kenntlich gemacht habe.

Die Arbeit wurde bisher in gleicher oder ähnlicher Weise keiner anderen Prüfungsbehörde vorgelegt und auch noch nicht veröffentlicht.

Die vorliegende Diplomarbeit ist mit dem elektronisch übermittelten Textdokument identisch.

Leonding, April 2023

Oliver Sugic

Abstract

In this thesis we present a new approach to help the teachers and the students to make their traveling experience better. It is done by an Web App that has the information over the excursion and guides the member of it from their location to the next Activity of the excursion. By having their location, the fear of losing people on the trip is reduced by a lot, because all members of the trip have the route to the Activity. The App is based on the Quarkus Framework and the Angular Framework which is secured by the Keycloak Framework.

Zusammenfassung

In dieser Arbeit stellen wir einen neuen Ansatz vor, der Lehrern und Schülern hilft, ihre Reiseerfahrung zu verbessern. Dies geschieht durch eine Web-App, die Informationen über die Exkursion enthält und die Teilnehmer von ihrem Standort zur nächsten Aktivität der Exkursion führt. Durch die Angabe ihres Standorts wird die Angst, Personen auf der Reise zu verlieren, stark reduziert, da alle Mitglieder der Reise den Weg zur aktuellen Aktivität in der Web-App finden. Die App basiert auf dem Quarkus Framework und dem Angular Framework, das durch das Keycloak Framework abgesichert ist.

Danksagung

Ich möchte mich herzlich bei meinem Diplomarbeitsbetreuer, Prof. Stütz, bedanken, der mich nicht nur mir beim realisieren dieser Arbeit geholfen hat, sondern mir auch viele gute Ideen und Tips geben konnte. Auch bei der Suche des Themas konnte er mir sehr gut helfen.

Ich bedanke mich auch bei meinem Mitschüler Herrn Pavelescu, der mir gute Tipps gegeben hat und mir auch bei Probleme helfen konnte. Durch Ihn konnte ich viele gute Ideen für die Arbeit sammeln.

Inhaltsverzeichnis

1 Ausgangssituation und Zielsetzung	1
1.1 Ausgangssituation	1
1.2 Ist-Zustand	1
1.3 Problemstellung	1
1.4 Aufgabenstellung	1
1.5 Marktanalyse	2
1.6 Gesamtkonzept	3
1.7 Ziele	3
2 Systemarchitektur	5
2.1 Quarkus Backend	5
2.2 Angular Frontend	9
2.3 Aktuelle Komponenten	12
2.4 Keycloak	22
2.5 Deployment	24
3 Schnittstellendefinition	29
3.1 Rest-Endpoints	29
4 Ausgewählte Aspekte	33
4.1 JsonManagedReference und JsonBackReference	33
5 Resümee	35
Literaturverzeichnis	VII
Abbildungsverzeichnis	VIII
Tabellenverzeichnis	IX
Quellcodeverzeichnis	X
	IV

1 Ausgangssituation und Zielsetzung

1.1 Ausgangssituation

An der HTBLA Leonding befinden etwa 1000 Schüler und 100 Lehrer. Die Schüler fahren auf Exkursionen wie z.B Sportwochen im Ausland oder auch im Inland.

1.2 Ist-Zustand

Es werden viele Exkursionen sowohl innerhalb Österreichs, aber auch ins europäische Ausland durchgeführt. Die Schüler und Schülerinnen werden dabei von den Lehrkräften betreut.

1.3 Problemstellung

Bei der Durchführung von Exkursionen, Projekt-, oder Sportwochen werden oft Treffpunkte vereinbart, an denen sich die Schülerinnen und Schüler zu einer gewissen Zeit mit den Lehrkräften treffen. Befindet man sich in einer fremden Stadt, so kann das Finden eines Treffpunkts durchaus herausfordernd sein. Auch ist es möglich das die Zeit oder der Ort verwechselt werden wird.

1.4 Aufgabenstellung

Es ist eine Applikation zu erstellen, in der die einzelnen Stationen und Treffpunkte der Reise aufgelistet sind. Die Schülerinnen und Schüler werden zwar über die einzelnen Zeile informiert, aber die Treffpunkte die Schülerinnen und Schüler aufrufbar, damit keine Konfusionen entstehen kann.

1.5 Marktanalyse

Um ein fertiges Produkt zu verwenden, muss eine Marktanalyse durchgeführt werden. Es wurden verschiedene Produkte verglichen und analysiert. Dabei wurden die folgenden Kriterien betrachtet:

- Funktionalität: Wie einfach können Reisen geplant werden?
- Benutzerfreundlichkeit: Wie einfach ist die Bedienung für die Teilnehmer der Reise?
- Preis: Wie viel kostet das Produkt?
- Datenschutz: Wie sicher sind die Daten der Teilnehmer der Reise? Werden Standortdaten gespeichert?

1.5.1 Wanderlog

Wanderlog ist eine mobile App des Unternehmens Travelchime Inc. Mit dieser App können Reisen geplant und durchgeführt werden. Die App ist kostenlos und kann für Android und iOS heruntergeladen werden. Die App bietet Funktionen wie Planung und Verbesserungen von Reisepläne. Mit der Pro-Version ist es möglich die Route der Reise zu optimieren um Geld zu sparen, als auch die Nutzung der App offline zu ermöglichen. Die Nutzung der Pro-Version kostet allerdings 49.99\$ jährlich.[1]

1.5.2 TripIt

TripIt ist eine mobile App des Unternehmens Concur Technologies. Durch die Möglichkeiten der App kann die Organisation und Verwaltung einer Reise dem Nutzenden sehr geholfen werden. Die jeweiligen Reisepläne können jederzeit in der App aufgerufen werden. Man kann sich seine Reisestatistiken anzeigen, als auch den CO² Fußabdruck der Reise berechnen lassen. Die App ist kostenlos und kann für Android und iOS heruntergeladen werden. Außerdem wird eine Pro-Version der App angeboten, bei der genauerer Informationen über die Flugreise bekannt gegeben werden. Die Nutzung der Pro-Version kostet allerdings 49\$ jährlich. [2]

1.6 Gesamtkonzept

Nach dem die Marktanalyse abgeschlossen war, wurde beschlossen, eine eigene Anwendung unter dem Name LeoTour zu entwickeln. Das System wurde in mehrere Bereiche unterteilt, die zu implementieren waren

1.6.1 Aufgabenbereiche der vorliegenden Arbeit

Die Bereiche werden folgendermaßen unterteilt:

- Backend: zur Verwaltung der Daten
- Frontend: Anzeigen der Daten
- Deployment: Bereitstellung der Anwendung

1.6.2 Funktionale Anforderungen

Es ist eine Anwendung zu entwickeln, die ...:

- Lehrkräfte das Erstellen von Reisen ermöglicht
- Schülerinnen und Schüler eine richtige Route angezeigt
- die Verwaltung von Reisen ermöglicht

1.6.3 Nicht funktionale Anforderungen

Für die jeweiligen Anforderungen ist es wichtig, ein übersichtliche graphische Oberfläche zu haben, die zu einfache Bedienung ermöglicht.

1.7 Ziele

Durch einen reibungslosen Ablaufs der Veranstaltung, sodass keine teilnehmenden Schüler gehen in einer fremden Stadt verloren. Somit wird die Sicherheit der Teilnehmer erhöht.

Um den Reisenden die Reise zu vereinfachen, soll die App die Koordinaten der jeweiligen Ziele anzeigen. Somit können bei Ausflügen von z.B Städte oder sonstigen Sehenswürdigkeiten Probleme wie z.B das Verlieren von Personen oder das Antreffen

bei einem falschen Ziel verhindert werden. Den benutzenden Personen soll der Ausflug erleichtert werden, sodass man ihn ohne Probleme genießen kann

2 Systemarchitektur

2.1 Quarkus Backend

2.1.1 Datenmodell

Bevor mit der Implementierung begonnen hatte, musste zuerst eine passendes Datenmodell entworfen werden, die für Anwendung passt. Ein erster Entwurf wurde daher konzipiert, um die Datenstruktur zu definieren.

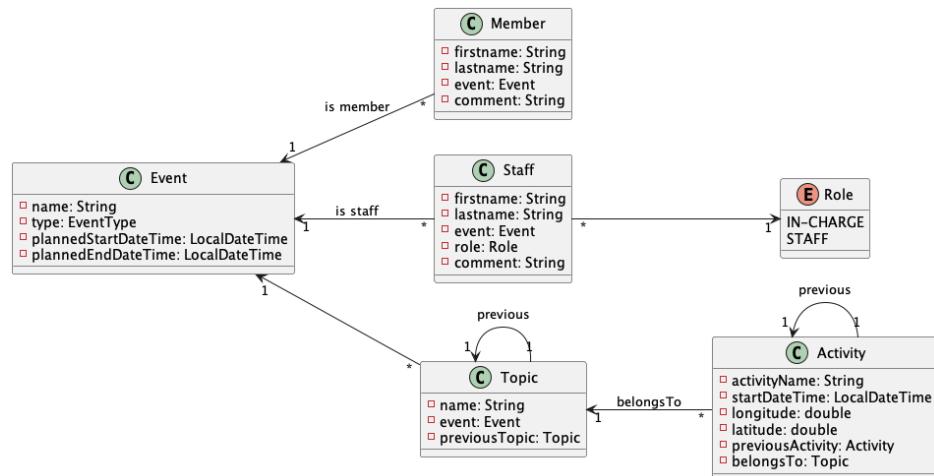


Abbildung 1: Erstes Entwurf des Datenmodells

Es gibt 5 Entitäten:

- Member: Ist eine teilnehmender Schüler der Reise
- Event: Ist die Reise selbst
- Topic: Ist ein Thema, das in der Reise behandelt wird beispielsweise die Stadt Rom
- Activity : Ist eine Aktivität, die in der Reise stattfindet beispielsweise ein Besuch im Colosseum
- Staff: Ist eine Begleitperson der Reise beispielsweise ein Lehrer

Doch nach dem Entwurf musste das Datenmodell nochmal neu überarbeitet werden, da die Entität „Staff“ überflüssig war.

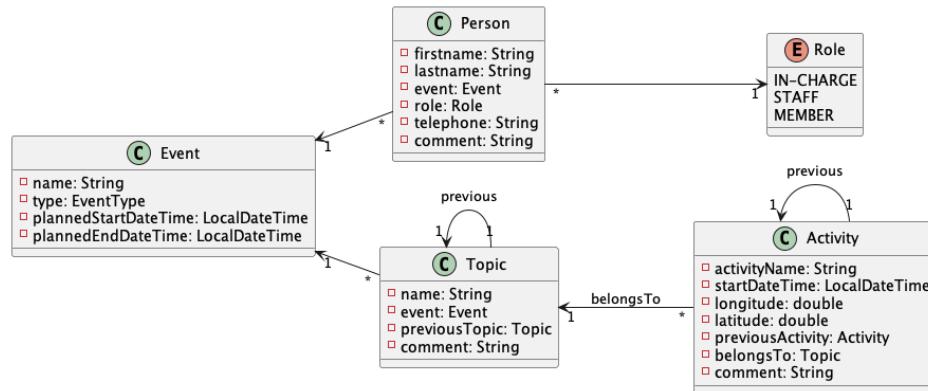


Abbildung 2: Überarbeiteter Entwurf des Datenmodell

Hier sieht man das überarbeitete Datenmodell. Es gibt nun nur noch 4 Entitäten:

- Person: Ist eine Teilnehmer Schüler der Reise
- Event: Ist die Reise selbst
- Topic: Ist ein Thema, das in der Reise behandelt wird beispielsweise die Stadt Rom
- Activity: Ist eine Aktivität, die in der Reise stattfindet beispielsweise ein Besuch im Colosseum

Die „Staff“-Entität wurde entfernt und zum Enum „Role“ hinzugefügt. Es gibt 3 verschiedene Rollen für eine teilnehmende Person:

- IN-Charge: die verantwortliche Person der Reisen
- Staff: die Personen, die die Reise begleiten und bei der Organisation helfen
- Member: die Schüler der Reise

Während der Implementierung wurde das Datenmodell nochmal überarbeitet, da man wenige Attribute hinzugefügt hat, die während der Implementierung benötigt wurden.

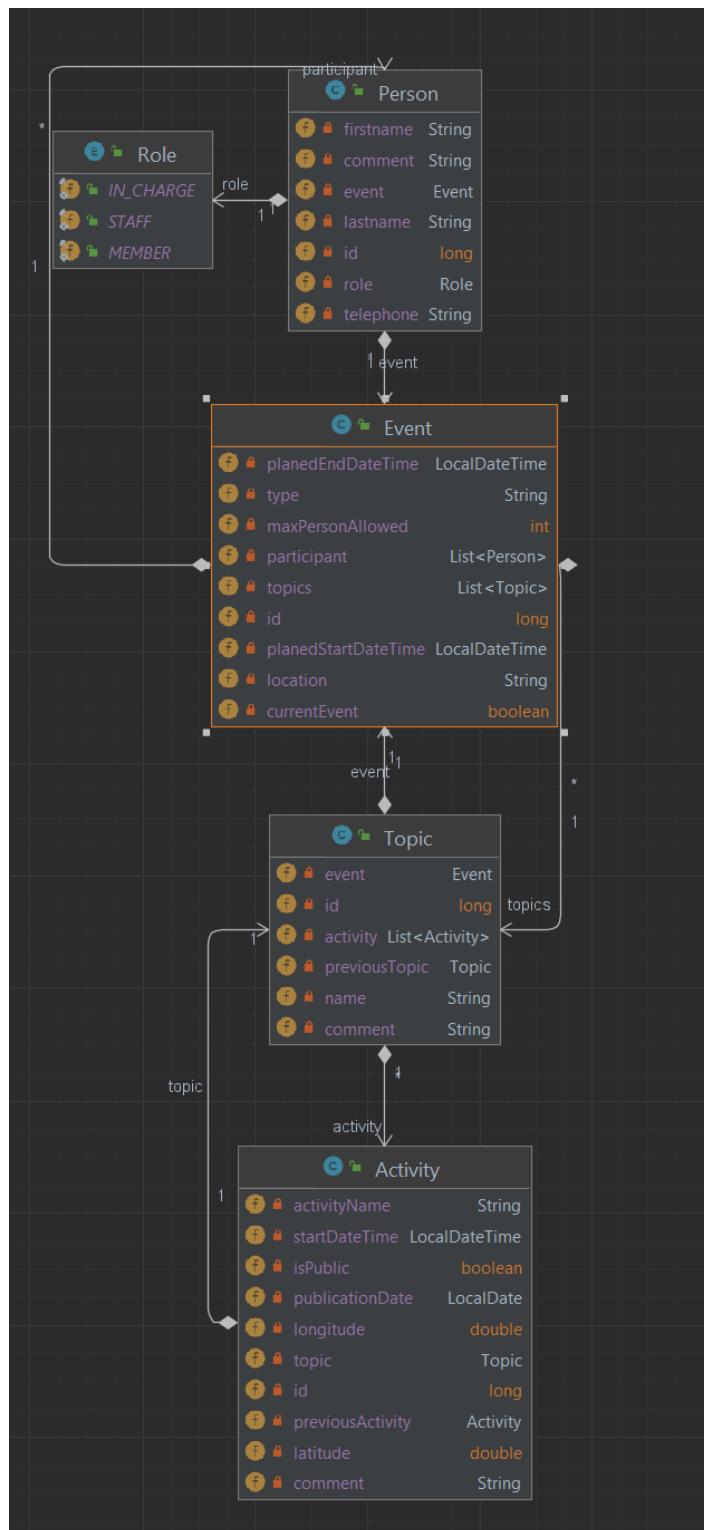


Abbildung 3: Endgültiger Entwurf des Datenmodell

Hier ist das endgültige Datenmodell zu sehen. Die wohl wichtigste Änderung ist das Attribut „currentEvent“, was die aktuelle Reise der Person angibt. Somit konnte in der neuen Anzeige14 die aktuelle Reise angezeigt werden. Auch für das Dashboard 20 wurde das Attribut benötigt, um die Reise zu wählen, welche in der Anzeige14 angezeigt werden soll.

2.1.2 Quarkus

Für die Verwaltung der Daten wurde ein Backend entwickelt und implementiert. Es wurde mit den Quarkus Framework geschrieben.

Quarkus ist ein Java-Framework, welches von der Firma RedHat entwickelt wurde. Es ermöglicht durch Verarbeitung der Anwendungscode, während des Kompilierens und der Ausführung, eine schnellere Startzeit und eine geringere Speicherauslastung. Dies wirkt sich positiv auf die Java Virtual Maschine aus. [3]

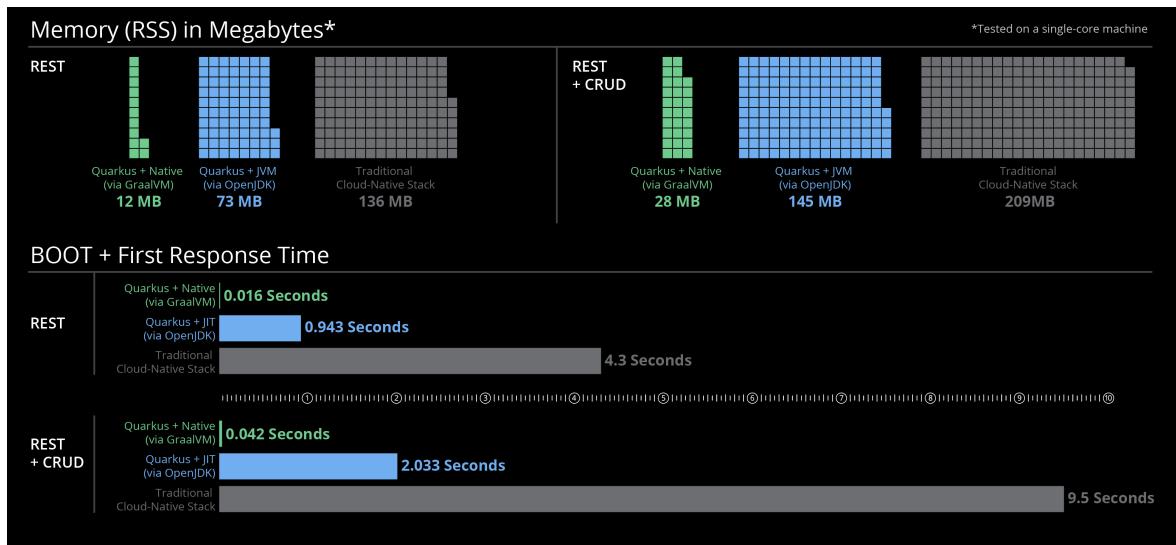


Abbildung 4: Quarkus Speicher Verbrauch [4]

Wie man in der überliegenden Abbildung 4 sehen kann, ist der gebrauchte Speicher von einer nativen kompilierten Quarkus Anwendung, egal ob REST mit CRUD Funktionalität oder ohne. Dies war einer der Gründe warum man für dieses Projekt Quarkus gewählt hat. Außerdem hat die Erfahrung mit dem Framework auch eine tragende Rolle gespielt.

2.1.3 Implementierung des Backend

Da nun das Datenmodell und das Framework festgelegt waren, konnte mit der Implementierung begonnen werden. Das backend sollte die Schnittstelle zum Frontend sein, aber auch die Datenbank verwalten.

2.2 Angular Frontend

Für die graphische Oberfläche wurden verschiedene Ideen und Konzepte ausprobiert, um die Benutzbarkeit für die Lernenden und Lehrenden zu optimieren. Im Laufe des Kapitels wird erläutert, welche Versionen der Anzeige es gab und welche Versionen sich als sinnvoll erwiesen haben.

2.2.1 Version 1: Visualisierung mit Karten

Am Anfang wurden Überlegungen angestellt, wie man die Lernenden als auch die Lehrenden am besten in nicht vertrautes Gebiet führen kann. Da die meisten Personen mit Kartenservices, wie beispielsweise Google Maps oder ähnlichen Dienstleistungen, vertraut sind, wurde eine Karte implementiert. Es gibt viele verschiedene Anbieter von Kartenservices, die in Betracht gezogen wurde. Da Google Maps eines der bekanntesten Anbieter, wurde auf die Google Maps Api gesetzt. Doch im Laufe der Recherche ist klar geworden, dass die Google Maps Api nicht die beste Lösung für dieses Projekt ist aus folgenden Gründen:

- Die Google Maps Api läuft über die Google Cloud und ist daher kostenpflichtig
- Google Maps Api ist nicht open-source
- Google Maps Api ist nicht einfach zu anzupassen an die Bedürfnisse des Projektes

[5]

Leaflet Karten

Auf der Suche nach weiteren Alternativen, verwies mich mein Klassenkollege Herr Pavelescu auf die Open-Source Kartenlösung Leaflet.

Leaflet ist eine JavaScript Library, die es ermöglicht, Karten in Webanwendungen zu integrieren. [6]

Listing 1: Implementierung einer Karte mit Leaflet

```

1   <style>
2     #map { height: 1000px; }
3   </style>
4   </head>
5   <body>
6     <link rel="stylesheet" href="https://unpkg.com/leaflet@1.9.3/dist/leaflet.css"
7       integrity="sha256-kLaT2GOSpHechhszzB+flnD+zUyjE2L1fWPgU04xyI="
8       crossorigin="" />
9
10    <script src="https://unpkg.com/leaflet@1.9.3/dist/leaflet.js"
11      integrity="sha256-WBkoX0wTeyKcl0HuWtc+i2uENFpDZ9YPdf5Hf+D7ewM="
12      crossorigin="" ></script>
13
14    <div id="map"></div>
15    <script>
16      var map = L.map('map').setView([48.2684159, 14.2517532], 20);
17      L.tileLayer('https://tile.openstreetmap.org/{z}/{x}/{y}.png', {
18        maxZoom: 19,
19        attribution: '&copy; <a href="http://www.openstreetmap.org/copyright">OpenStreetMap</a>',
20      }).addTo(map);
21    </script>

```

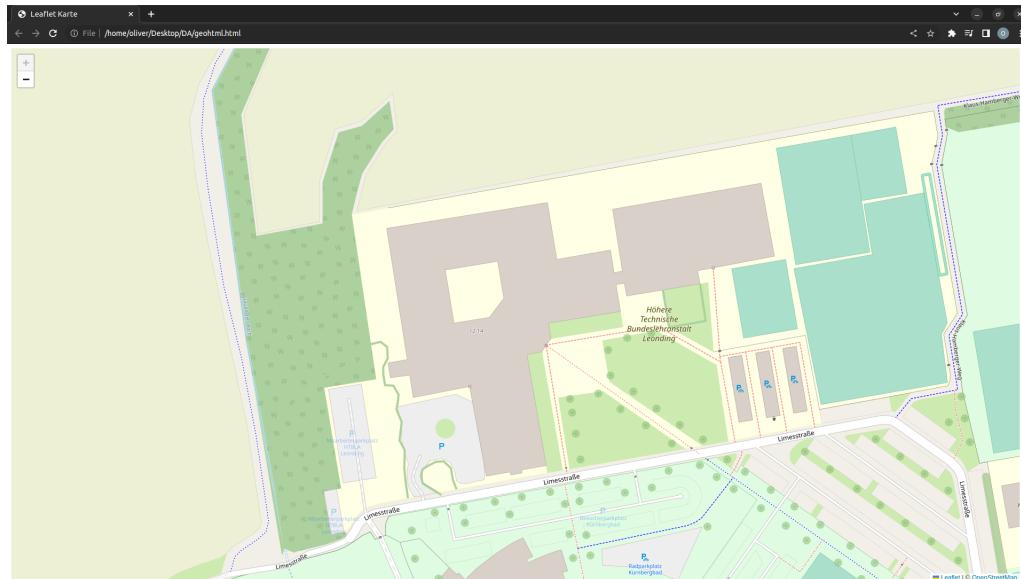


Abbildung 5: Ergebnis der Implementierung mit Leaflet

2.2.2 Version 2: Leaflet Karte mit Routing

Nach dem ersten Prototypen mit Leaflet, wurde die Idee weiter verfolgt, um die Benutzerfreundlichkeit für die Teilnehmenden zu verbessern. Eine Route vom eigenen Standort zur nächsten Aktivität wurde implementiert. Um die Routen anzeigen zu können, wurde die Leaflet Routing Machine verwendet. Hierfür wird das Plugin Leaflet Routing Machine verwendet, das ebenfalls vom Leaflet zur Verfügung gestellt wird. Mit dieser Erweiterung können Routen zwischen zwei Punkten auf der Karte berechnet werden und angezeigt werden. Ebenfalls können die Wegpunkte einfach geändert werden.[7]

Listing 2: Implementierung einer Karte mit Leaflet Routing Engine

```

1  <head>
2      <title>Leaflet Karte</title>
3      <style>
4          #map {
5              height: 1500px;
6          }
7      </style>
8  </head>
9
10 <body>
11     <link rel="stylesheet" href="https://unpkg.com/leaflet@1.9.3/dist/leaflet.css"
12         integrity="sha256-kLaT2GOSpHechhszzB+fInD+zUyjE2LlfWPgU04xyI="
13         crossorigin="" />
14
15     <script src="https://unpkg.com/leaflet@1.9.3/dist/leaflet.js"
16         integrity="sha256-WBkoX0wTeyKclOHuWtc+i2uENFpDZ9Ypdf5Hf+D7ewM="
17         crossorigin="">
18     </script>
19     <link rel="stylesheet" href="https://unpkg.com/leaflet-routing-machine@latest/dist/leaflet-routing-machine.css"
20         />
21     <script
22         src="https://unpkg.com/leaflet@1.2.0/dist/leaflet.js"></script>
23     <script
24         src="https://unpkg.com/leaflet-routing-machine@latest/dist/leaflet-routing-machine.js">
25     </script>
26     <div id="map"></div>
27     <script>
28         var map = L.map('map').setView([48.2684159, 14.2517532], 15);
29         L.tileLayer('https://tile.openstreetmap.org/{z}/{x}/{y}.png', {
30             maxZoom: 19,
31             attribution: '&copy; <a href="http://www.openstreetmap.org/copyright">OpenStreetMap</a>',
32         }).addTo(map);
33         L.Routing.control({
34             waypoints: [
35                 L.latLng(48.2684159, 14.2517532),
36                 L.latLng(48.2627373, 14.2589871)
37             ]
38         }).addTo(map);
39     </script>
40 </body>

```

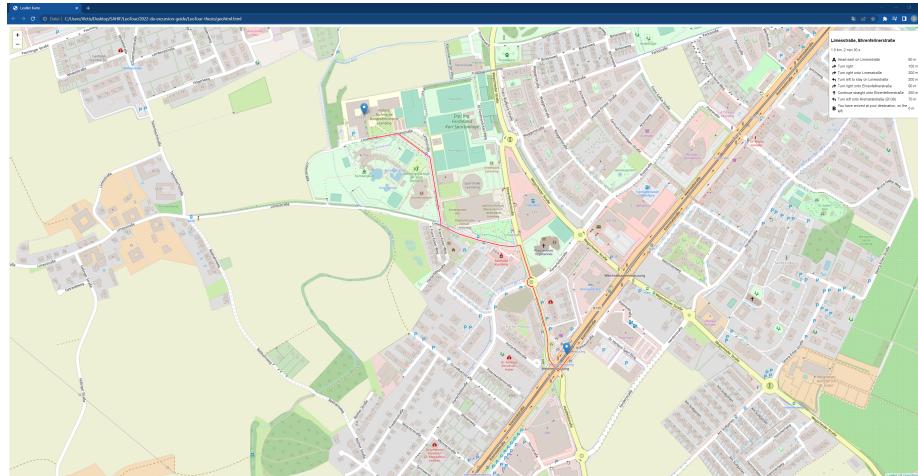


Abbildung 6: Ergebnis der Implementierung mit Leaflet Routing Machine

Allerdings traten bei der Implementierung Probleme auf. Das Plugin konnte nicht die Route darstellen, was auf einen Fehler in der Implementierung zurückzuführen war.

2.2.3 Version 3: Angular Geoloaction API

Da die Implementierung mit der Leaflet Routing Machine nicht funktioniert hat, wurde versucht, mittels des Standorts des Nutzenden zur nächsten Aktivität zu navigieren.

2.3 Aktuelle Komponenten

Derzeit besteht die Anwendung aus drei Komponenten. In diesem Kapitel werden der Aufbau und Zusammenhänge der einzelnen Komponenten erläutert.

2.3.1 Angular Frontend

Der schwierigste Teil der Arbeit liegt darin, eine Benutzeroberfläche zu erstellen, die einfach zu bedienen aber auch ansprechend für den Benutzer ist. Es wurde daher auf das Angular Framework genommen, da es sehr

Schüler Sicht

Nach dem Scheitern der Implementierung mit der Leaflet Routing Machine, wurde eine neues Design entworfen, um die Benutzerfreundlichkeit zu verbessern. Nach vielen Überlegung wurde ein Wireframe entworfen 7, welches sowohl für mobile Endgeräte geeignet ist, aber auch überschaubar für die nutzenden Personen ist.

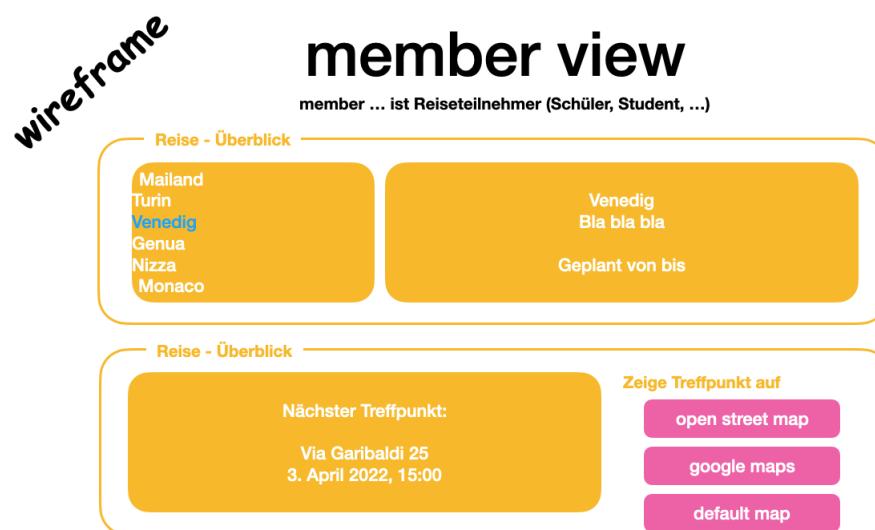


Abbildung 7: Entwurf der Schüleransicht

Der Nutzer soll in einen groben Überblick haben, welche Städte er besuchen wird. Durch eine kurze Beschreibung der Aktivität solle der Reisende einen kurzen Einblick haben, was Ihn erwarten wird. In der unteren Hälfte des Bildschirm erhält man genauere Informationen über den Treffpunkt, außerdem ist es möglich sich auf der Karte die Route anzeigen zu lassen. Es ist möglich, sich auf der standartmäßigen Karte des Endgeräts, über Google Maps oder über die OpenStreetMap zu navigieren. Der Nutzende kann für sich entscheiden, welche Karte Ihm am besten gefällt oder welche Karte Ihm am meisten vertraut ist wie in Abbildung 8 zu sehen ist.

Kulturwoche Norditalien

Mailand	Mailand ist bekannt als die Modemetropole schlechthin, doch die italienische Stadt weiß mit noch viel mehr zu beeindrucken. Hier befinden sich historische Sehenswürdigkeiten, die nicht nur wunderschön zu betrachten sind, sondern auch viel zu erzählen haben.
Turin	
Venedig	
Genua	

Nächster Treffpunkt

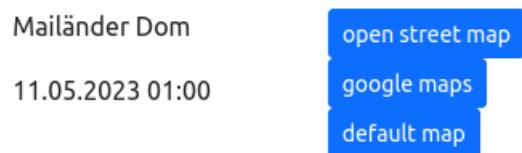


Abbildung 8: Schüler Ansicht

Da den Schüler und Schülerinnen die Mitführung von elektronischen Geräten, müssen die Schüler und Schülerinnen Ihre Handys benutzen, da die Mitführung eines Handys nicht verboten ist. Somit musste die Ansichten für mobil Geräte angepasst werden.

Aktivitäten freischalten

Um die Aktivitäten freizuschalten, kann der Lehrer über die Unlock-Activity Komponente die Aktivitäten freischalten. Es werden alle Aktivitäten angezeigt und kann durch bedienen des Buttons freigeschaltet werden oder auch wieder gesperrt werden.

Kulturwoche Norditalien

Mailand

Königlicher Palast	Aktivität freischalten
Mailänder Dom	Aktivität sperren

Turin

Palazzo Reale	Aktivität freischalten
Venaria Reale	Aktivität sperren

Venedig

Scuola Grande di San Rocco	Aktivität freischalten
Canal Grande	Aktivität sperren

Genua

Porto Antico	Aktivität freischalten
Das Aquarium von Genua	Aktivität sperren

Abbildung 9: Aktivitäten freischalten und sperren

Wieder musste auf die mobile Ansicht geachtet werden. Die Lehrer oder die Lehrerinnen haben zwar einen Laptop mit, aber da es sehr unpraktisch ist mit einem Laptop durch unbekanntes Gebiet herumzureisen, wurde die Ansicht für mobile Geräte angepasst.

Event erstellen

Für das erstellen eines Events wurde eine eigene Komponente entworfen und entwickelt. Da eine Lehrkraft in die Reise im voraus planen muss, ist sie für den Gebrauch von Laptops angepasst worden.

Die Komponente wurde in vier Abschnitte geteilt:

- Daten des Events
- Daten der Teilnehmer
- Daten des Themas
- Daten der Aktivitäten

The screenshot shows a mobile application interface for creating an event. It includes fields for Location, Max Person Allowed, Type, Planned Start Date & Time, and Planned End Date & Time. Each field has an input box and a date/time picker icon.

Abbildung 10: Abschnitt für die Daten des Events

The screenshot shows a mobile application interface for managing participant details. It includes fields for First Name, Last Name, Role (with a dropdown menu showing 'IN_CHARGE'), Telephone, and Comment. Each field has an input box.

Abbildung 11: Abschnitt für die Daten der Teilnehmer

The screenshot shows a form titled 'Topics' with a 'Create' button at the bottom. It contains five input fields: 'Name' (with placeholder 'Topic'), 'Activities' (with placeholder 'Topic'), 'Public' (with placeholder 'Public'), 'Previous Topic' (with placeholder 'Topic'), and 'Comment' (with placeholder 'Comment').

Abbildung 12: Abschnitt für die Daten des Themas

The screenshot shows a form titled 'Activities' with a 'Create' button at the bottom. It contains seven input fields: 'Activity Name' (with placeholder 'Activity Name'), 'Start Date & Time' (with placeholder 'mm/dd/yyyy, --:--'), 'Longitude' (with placeholder 'Longitude'), 'Latitude' (with placeholder 'Latitude'), 'Previous Activity' (with placeholder 'Activity'), 'Comment' (with placeholder 'Comment'), and 'Is Public' (with placeholder 'Is Public').

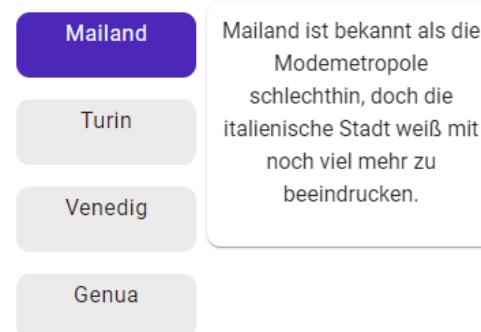
Abbildung 13: Abschnitt für die Daten der Aktivitäten

Überarbeitung mit Angular Material

Allerdings waren die Entwürfe teilweise unübersichtlich und farblich nicht ansprechend. Schlussendlich wurden alle Angular Komponenten mit Angular Material überarbeitet. Durch das hinzufügen des Angular Material Frameworks, wurde die Benutzeroberfläche ansprechender und übersichtlicher.

Mit den Möglichkeiten des Angular Material Framework wurde die Benutzeroberfläche überarbeitet.

Kulturwoche Norditalien



Nächster Treffpunkt

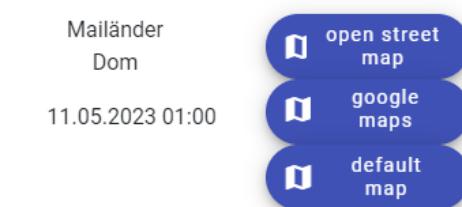


Abbildung 14: Neue Oberfläche für die Schüleransicht

Im Vergleich ist die Abbildung 14 im Vergleich mit der alten Version 8 viel ansprechender. Durch die Verwendung eines Material Cards hat der Text des ausgewählten Themas eine bessere Lesbarkeit und wird auch besser herausgehoben. Durch das Trennen der zwei Abschnitte mit Hilfe von Material Divider wird ein klare Trennung zwischen den beiden Abschnitten erreicht. Ebenso wurde die Buttons für die Navigation auf der Karte in Material Buttons umgewandelt.

Ebenso wurde das Design für das Freischalten der Aktivitäten überarbeitet.

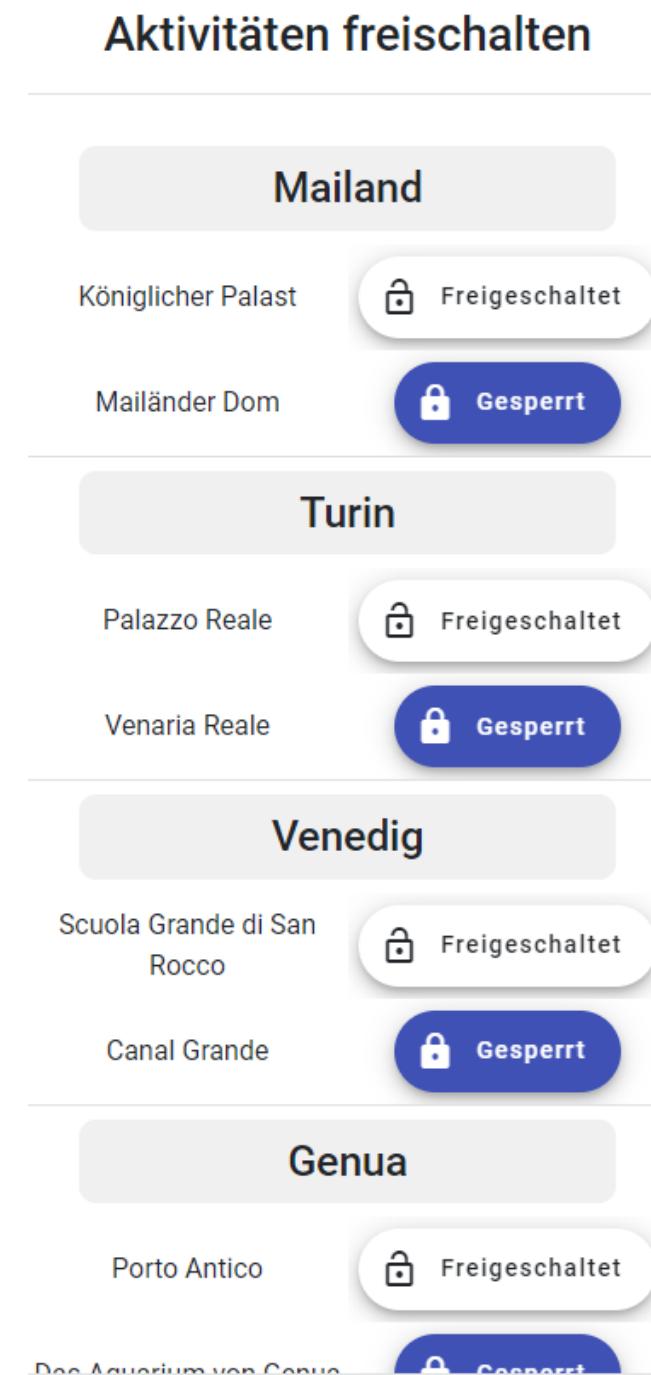


Abbildung 15: Neue Oberfläche für die Schüleransicht

Im Vergleich zur alten Version 9 ist die neue Version 15 verbessert worden. Die Buttons wurden in Material Buttons umgewandelt und so entworfen, dass die Lehrerinnen und Lehrer einen direkten Überblick haben, welche Aktivitäten freigeschaltet sind und welche nicht.

Neu ist auch die Ansicht zum erstellen eines Events. Da beim Testen der alten Version festgestellt wurde, dass diese nicht die Funktionalität erfüllt, musste die Komponente neu entworfen werden. Dazu entschloss man sich für die Umsetzung, den Angular Material Stepper zu verwenden. Dieser ermöglicht es, die einzelnen Abschnitte des Events in einzelne Schritte zu unterteilen. Praktisch ist auch, dass der Stepper die Möglichkeit bietet, Schritte zurückzugehen, um die Eingaben zu korrigieren. Mit Hilfe des Steppers ist das Erstellen eines Events einfacher und besser im Vergleich zur Vorgänger Version.

- Daten des Events

Create Event

1 Event Information 2 Add Participants 3 Add Topics and Activities

Event Information

Location: Belgien
Max Person Allowed: 36
Type: Kulturwoche
Planned Start Date & Time: 27.10.2025 23:09
Planned End Date & Time: 31.10.2025 23:09

Abbildung 16: Neue Oberfläche für die Eingabe der Daten des Events

- Daten der Personen

Create Event

1 Event Information 2 Add Participants 3 Add Topics and Activities

Participant

Firstname: Oliver
Lastname: Sugic
Role: Member
Telephone: 0123456789
Comment: Schüler

Firstname	Lastname	Role	Delete
Oliver	Sugic	In Charge	

+ Add Participant

Abbildung 17: Neue Oberfläche für die Eingabe der Daten der Personen

- Daten der Themen Die Oberfläche wurde so gelöst, dass die Lehrerinnen und

The screenshot shows the 'Create Event' interface. On the left, there's a sidebar with 'Event Information' (selected), 'Topics', 'Participants', and 'Topics and Activities'. Under 'Topics', there's a 'Name' field with 'Antwerpen' and a 'Comment' field. Below these are buttons for '+ Add Topic' and 'Create Event'. To the right, there are two tables: 'Activities for Brüssel' (with one entry: 'Europäische Kommission') and 'Activities for Antwerpen' (empty). A 'Previous Topic' dropdown shows 'Brüssel'.

Abbildung 18: Neue Oberfläche für die Eingabe der Daten der Themen

Lehrer bei der Eingabe der Daten ein direkte Übersicht haben, welche Themen angelegt wurden. Beim Anklicken eines Themas werden die bereits angelegten Aktivitäten angezeigt. Beim Anklicken des Buttons „Add Activity“ wird ein Dialog geöffnet, in dem die Daten für eine neue Aktivität eingegeben werden können.

- Daten der Aktivitäten

The screenshot shows the 'Create Event' interface with the 'Activity' section active. It includes fields for 'Activityname' (Europäische Kommission), 'Start Date & Time' (28.10.2024 23:12), 'Longitude' (14.789), 'Latitude' (5.456), 'Comment', 'Public' (checked), and 'Publication Date' (28.10.2024). There's also a '+ Add Activity' button at the bottom.

Abbildung 19: Neue Oberfläche für die Eingabe der Daten der Aktivitäten

Nach dem die Komponente für das Anlegen der Events nun fertig war, stellte man fest, das man keine Übersicht hat über die schon geplanten Reisen. Außerdem musste man einen Weg finden, dass die Lehrerinnen und Lehrer die richtige Reise auswählen können.

Nach mehreren Überlegungen, entschloss man sich für die Erstellung einer Komponente. Die sollte als Übersicht oder auch als „Dashboard“ dienen. Wie sie implementiert werden sollte, war nicht ganz klar. Hierbei gab es verschiedene Möglichkeiten, welche man in Betracht gezogen worden sind:

- Anzeige der Events an die zugewiesenen Schülerinnen und Schülern

Hier war die Überlegung, dass die Lehrerinnen und Lehrer eine Reise auswählen und diese ihrer Klasse zuweisen. Die Schülerinnen und Schüler der jeweiligen Klasse können sich dann die Reise sehen. Allerdings wurde die Idee verworfen, da im Laufe der Entwicklung mit dem Keycloak Probleme entstanden, die im Kapitel 2.4 beschrieben werden.

- Anzeige des Events an alle Personen

Nach dem gescheiterten Versuch mit dem Keycloak, wurde eine neue Idee umgesetzt. Lehrerinnen und Lehrer müssen die Schulexkursion oft mehrere Monate im voraus planen. Jeder kann sich die Reise ansehen, die gerade oder als nächstes stattfindet.

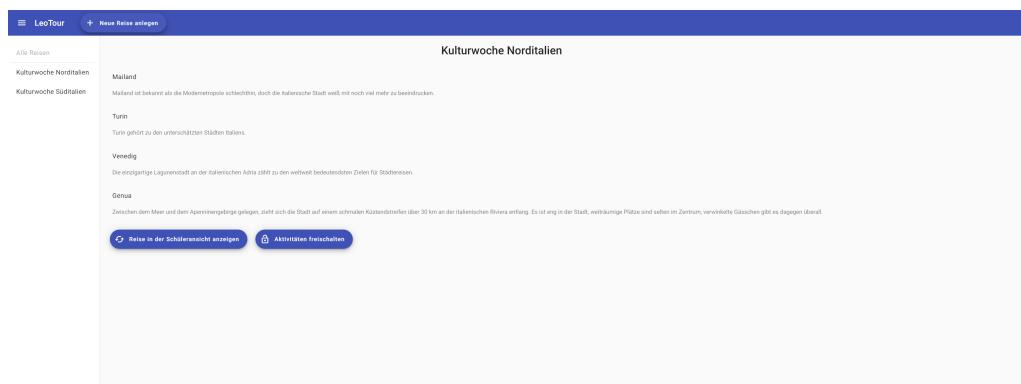


Abbildung 20: Dashboard für die Lehrerinnen und Lehrer

Hier werden die Reisen aufgelistet werden und die Lehrerinnen und Lehrer haben die Möglichkeit haben, ihre geplante Reise auszuwählen. Am oberen Rand befindet sich ein Button „Neu Reise anlegen“ mit dem man sich zum Erstellen von Reisen navigieren kann. Nachdem man eine Reise angeklickt hat, kann man sich Informationen über Reise einholen. Ebenso kann man zum Freischalten der Aktivitäten navigieren, hierfür gibt es auch einen Button „Aktivitäten freischalten“. Falls die Lehrerin oder der Lehrer nicht sicher sind, ob zu viele Informationen über die Reise vorhanden sind, können sie via den Button „Reise in der Schüleransicht anzeigen“ die Reise zu Überprüfung in der Schüleransicht nachsehen.

2.4 Keycloak

Um die Anwendung zu schützen, braucht es einen Authentifizierungsserver. Dieser ist für die Anwendung sehr wichtig, da die Schülerinnen und Schüler nur die Reise sehen können, die ihnen zugewiesen wurde. Außerdem können nur Lehrerinnen und Lehrer Reisen erstellen und bearbeiten. Um sich von großen Firmen loszulösen wie beispielsweise von Google. Sollte daher bei einem Daten Leak von Google keine Daten von Schülerinnen und Schülern betroffen sein. Wurde daher Keycloak als Authentifizierungsserver gewählt. Keycloak ist ein Open-Source-Software-Produkt, das als Authentifizierungsserver und Identityprovider dient. Außerdem bietet Keycloak den Vorteil, dass nicht nur die Autorisierung übernimmt sondern auch die Authentisierung der Benutzer. Ebenfalls kann er mit Active-Directory-Servern authentifizieren, was für die Anwendung sehr praktisch ist, da die HTBLA Leonding bereits so ein System besitzt. [8]

Zwar wurde Anfang auf ein eigene Keycloak gesetzt, doch schnell wurde klar, dass es einfacher wäre, den Keycloak zu nehmen der von der Schule bereit gestellt wird, um sich mit den Schüler Login Daten anzumelden.

Dieser wurde im Angular Frontend eingebunden. Somit wurde eine neue Komponente erstellt, welche den Zweck hat, den Login und Logout zu ermöglichen. Außerdem wurde überprüft ob der Benutzer ein als Schülerin oder Schüler angemeldet ist. Falls ja, wird die Komponente für die Schülerin oder den Schüler angezeigt. Falls nein, wird die Komponente für die Lehrerinnen und Lehrer angezeigt. Im folgenden Code sieht man die Implementierung des Keycloak in Angular.

Listing 3: Implementierung im Angular

```

1  ngOnInit() {
2      this.oidcSecurityService
3          .checkAuth()
4          .subscribe(({isAuthenticated, userData, accessToken, idToken}) => {
5              this.userData = userData;
6          });
7      if (this.userData.preferred_username.indexOf('..') > -1 || 
8          this.userData.preferred_username == "if180157") {
9          this.isTeacher = true
10     } else {
11         this.isStudent = true;
12     }
13 }
14 login() {
15     this.oidcSecurityService
16         .authorizeWithPopUp()
17         .subscribe(({isAuthenticated, userData, accessToken, errorMessage}) => {
18             this.userData = userData;
19         });
20     this.oidcSecurityService.authorize();
21 }
22 logout() {
23     this.oidcSecurityService
24         .logoff()
25         .subscribe((result) => console.log(result));
26 }
```

Nach dem allerdings diese Funktion beim Deployment getestet wurde, funktionierte der Redirect auf die eigentliche Applikation nicht mehr.



Abbildung 21: Fehlschlagender Redirect nach dem Login

Aufgrund dessen wurde der Keycloak aus dem Projekt entfernt und eine funktionierende Anwendung zu gewährleisten.

2.5 Deployment

Deployment ist die Bereitstellung von Software. Es umfasst dabei Prozesse, die automatisiert werden wie die Installation, Konfiguration und weiteren auf dem Server. [9]

Deployment war daher für das Projekt ein wichtiger Bestandteil. Die teilnehmenden Personen müssen auch auf die Website zugreifen können, um dann die Funktionalität der Anwendung nutzen zu können. Wo die Anwendung bereitgestellt wird, gab es zwei Optionen:

- LeoCloud
- Oracle VM

Nach nicht allzu langer Überlegungen nahmen wir die LeoCloud, welche von der Schule bereit gestellt wurde, wo die Schüler ihre Anwendung deployen können. Die Oracle VM schlossen wir aus, da sie viel zu wenig Leistung hat und die Anwendung nicht richtig laufen würde.

Mittels Gihtub-Actions konnten wir dann die sowohl das Backend, Frontend und die Datenbank in GitHub Packages speichern. Die Github Actions sehen folgendermaßen aus:

Listing 4: Github Actions für das Backend

```

1  name: build-image-backend
2  on:
3      push:
4          branches: [ main ]
5
6  env:
7      REGISTRY: ghcr.io
8
9  jobs:
10     build:
11         name: build
12         runs-on: ubuntu-latest
13         steps:
14             - uses: actions/checkout@v3
15
16             - name: convert github repository name to lowercase
17             run: echo "IMAGE_REPOSITORY=$(echo ${{ github.repository }} | tr
18                   '[:upper:]' '[:lower:])" >> $GITHUB_ENV
19
20             - name: convert github registry name to lowercase
21             run: echo "IMAGE_REGISTRY=$(echo ${{ env.REGISTRY }} | tr '[:upper:]'
22                   '[:lower:]')" >> $GITHUB_ENV
23
24             - name: Log in to the Container registry
25             uses: docker/login-action@v2
26             with:
27                 registry: ${{ env.REGISTRY }}
28                 username: ${{ github.actor }}
29                 password: ${{ github.token }}
30
31             - name: Set up Docker Buildx
32             uses: docker/setup-buildx-action@v2
33
34             - name: Build and push
35             uses: docker/build-push-action@v4
36             with:
37                 context: "${{ defaultContext }}:backend"
38                 file: ./src/main/docker/Dockerfile.jvm
39                 push: true
40                 tags: ${{ env.REGISTRY }}/${{ env.IMAGE_REPOSITORY }}-backend:latest
41                 cache-from: type=registry,ref=${{ env.REGISTRY }}/${{ env.IMAGE_REPOSITORY }}-backend:buildcache
42                 cache-to: type=registry,ref=${{ env.REGISTRY }}/${{ env.IMAGE_REPOSITORY }}-backend:buildcache,mode=max

```

Wie man bereits erkennen kann, wird diese Action immer dann ausgeführt, wenn ein Push auf den ‘main’ Branch gemacht wird. Außerdem wird ein Image gespeichert, welches dann zu einem Github Package hochgeladen wird. Allerdings braucht es ein Dockerfile, um überhaupt ein Image zu erstellen. Dieses sieht folgendermaßen aus:

Listing 5: Dockerfile für das Backend

```

1  FROM registry.access.redhat.com/ubi8/openjdk-17:1.13-1.1655306439 as build
2
3  USER root
4
5  COPY mvnw /code/mvnw
6  COPY .mvn /code/.mvn
7  COPY pom.xml /code/
8
9  WORKDIR /code
10 RUN ./mvnw -B org.apache.maven.plugins:maven-dependency-plugin:3.1.2:go-offline
11 COPY ./src /code/src
12 RUN ./mvnw package -Dmaven.test.skip
13
14 FROM registry.access.redhat.com/ubi8/openjdk-17:1.13-1.1655306439
15
16 ENV LANG='en_US.UTF-8' LANGUAGE='en_US:en'
17
18 # We make four distinct layers so if there are application changes the library
19 # layers can be re-used
20 COPY --from=build --chown=185 /code/target/quarkus-app/lib/ /deployments/lib/
21 COPY --from=build --chown=185 /code/target/quarkus-app/*.jar /deployments/
22 COPY --from=build --chown=185 /code/target/quarkus-app/app/ /deployments/app/
23 COPY --from=build --chown=185 /code/target/quarkus-app/quarkus/
24 /deployments/quarkus/
25
26 EXPOSE 8080
27 USER 185
28 ENV AB_JOLOKIA_OFF=""
29 ENV JAVA_OPTS="-Dquarkus.http.host=0.0.0.0
30           -Djava.util.logging.manager=org.jboss.logmanager.LogManager"
31 ENV JAVA_APP_JAR="/deployments/quarkus-run.jar"
```

Nach dem nun jeweils die Packages generiert waren, konnte man sich nun auf die LeoCloud konzentrieren. Dazu braucht man ein YAML File, welches das Deployment als auch den Service zu Verfügung stellt. Doch was ist die Aufgabe eines Services und einem Deployment? Mit einem Deployment kann ich mehrere Pods eines Kubernetes Cluster verwalten, hingegen der Service dafür zuständig ist, dass die Pods ein Netzwerkzugriff ermöglicht wird.[10]

Da dies nun geklärt ist, schauen wir uns nun das YAML File dazu an:

Listing 6: Dockerfile für das Backend

```
1 apiVersion: apps/v1
2 kind: Deployment
3 metadata:
4   name: frontend-deployment
5   namespace: student-o-sugic
6 spec:
7   replicas: 1
8   selector:
9     matchLabels:
10    app: frontend
11   template:
12     metadata:
13       labels:
14         app: frontend
15     spec:
16       containers:
17         - name: frontend
18           image:
19             ghcr.io/htl-leonding-project/2022-da-excursion-guide-frontend:latest
20           ports:
21             - containerPort: 80
22           imagePullPolicy: Always
23 ---
24 apiVersion: v1
25 kind: Service
26 metadata:
27   name: frontend-svc
28   namespace: student-o-sugic
29 spec:
30   ports:
31     - port: 80
32       targetPort: 80
33       protocol: TCP
34       name: http
35   selector:
36     app: frontend
```

Pods								
Name	Labels	Node	Status	Neustarts	CPU-Nutzung (Kerne)	Speicher Nutzung (Bytes)	Erstellungszeitpunkt	⋮
frontend-deployment-7ddcfdd8b-hrlfv	app: frontend pod-template-hash: 7ddcfdd8b	hulk	Running	0	-	-	13.days.ago	⋮
leotour-backend-77b59c8b5-r54bv	app: leotour-backend pod-template-hash: 7859c8b5	hulk	Running	0	-	-	13.days.ago	⋮
leotour-db-6fb2eb75khrik6	app: leotour-db pod-template-hash: 6fb2eb75	godzilla	Running	0	-	-	13.days.ago	⋮

Abbildung 22: Kubernetes Dashboard

Services						
Name	Labels	Cluster-IP	Interne Endpoints	Externe Endpoints	Erstellungszeitpunkt	⋮
leotour-db	-	10.104.173.209	leotour-db.student-o-sugic:5432 TCP leotour-db.student-o-sugic:5432 TCP	-	23.days.ago	⋮
frontend-svc	-	10.101.146.146	frontend-svc.student-o-sugic:80 TCP frontend-svc.student-o-sugic:80 TCP	-	23.days.ago	⋮
leotour-backend	-	10.99.136.205	leotour-backend.student-o-sugic:8090 TCP leotour-backend.student-o-sugic:8090 TCP	-	23.days.ago	⋮

Abbildung 23: Kubernetes Services

Zur Überprüfung wird sich das Kubernetes Dashboard kontrolliert ob keine Fehler aufgetreten sind.

2.5.1 Mögliche zukünftige Erweiterungen

Für die Zukunft könnten weiter mögliche Erweiterungen implementiert werden:

- Entwicklung einer Mobile App mit Android oder Swift Durch diese könnte man die Web-Ansicht ablösen und durch eine App ersetzen.
- Anmelden mit Schullogins Aufgrund des bereits beschriebenen Problems mit dem Keycloak ist eine Anmeldung mit den Logindaten der Schule nicht möglich. Sollte eine Lösung
- Hinzufügen einer Schnitzeljagd Man könnte die Funktion einer Schnitzeljagd implementieren, um die Schülerinnen und Schüler mehr zur Reise motivieren .

3 Schnittstellendefinition

3.1 Rest-Endpoints

Wichtig war es, die Schnittstellen so zu implementieren, das Sie sowohl GET Requests für das anzeigen von Reisen und Aktivitäten, als auch POST Requests für das Erstellen von Reisen. Jede Entität hat eine CRUD Funktionalität implementiert. Im Quarkus Swagger kann man sich die Schnittstellen anschauen:

The screenshot shows the 'Event Resource' section of the API documentation. It lists various HTTP methods and their corresponding URLs:

- POST** /api/event/addEvent
- POST** /api/event/addPerson/{eventname}
- POST** /api/event/addTopic/{eventname}
- DELETE** /api/event/deleteEvent/{id}
- PATCH** /api/event/editEvent/{id}
- GET** /api/event/getAll
- GET** /api/event/getCurrentEvent
- GET** /api/event/updateCurrentEventField/{id}
- GET** /api/event/{id}

Abbildung 24: Event Resource

The screenshot shows the 'Topic Resource' section of the API documentation. It lists various HTTP methods and their corresponding URLs:

- POST** /api/topic/addActivity/{topicname}
- POST** /api/topic/addTopic
- DELETE** /api/topic/deleteTopic/{id}
- PATCH** /api/topic/editTopic/{id}
- GET** /api/topic/getAll
- GET** /api/topic/{id}

Abbildung 25: Topic Resource

The screenshot shows the 'Activity Resource' section of the API documentation. It lists various HTTP methods and their corresponding URLs:

- POST** /api/activity/addActivity
- DELETE** /api/activity/deleteActivity/{id}
- PUT** /api/activity/editActivity/{id}
- GET** /api/activity/getAll
- GET** /api/activity/{id}

Abbildung 26: Activity Resource

Person Resource	
POST	/api/person/addPerson
DELETE	/api/person/deletePerson/{id}
PATCH	/api/person/editPerson/{id}
GET	/api/person/getAll
GET	/api/person/getByName/{name}
GET	/api/person/{id}

Abbildung 27: Person Resource

3.1.1 Create Event

Wenn die Lehrerin oder Lehrer die Reise richtig angelegt haben, muss die Struktur als JSON Objekt folgendermaßen aussehen:

```

1  {
2      "location": "string",
3      "maxPersonAllowed": 0,
4      "type": "string",
5      "participant": [
6          {
7              "firstname": "string",
8              "lastname": "string",
9              "role": "IN_CHARGE",
10             "telephone": "string",
11             "comment": "string",
12             "event": "string"
13         }
14     ],
15     "topics": [
16         {
17             "name": "string",
18             "activity": [
19                 {
20                     "activityName": "string",
21                     "startDateTime": "2023-03-30T17:59:13.428Z",
22                     "longitude": 0,
23                     "latitude": 0,
24                     "previousActivity": "string",
25                     "comment": "string",
26                     "isPublic": true,
27                     "publicationDate": "2023-03-30",
28                     "topic": "string",
29                     "public": true
30                 }
31             ],
32             "previousTopic": "string",
33             "event": "string",
34             "comment": "string"
35         }
36     ],
37     "plannedStartTime": "2023-03-30T17:59",
38     "plannedEndTime": "2023-03-30T17:59",
39     "currentEvent": true
40 }
```

3.1.2 Get Current Event

Diese Methode wurde implementiert, um die aktuelle Reise zu bekommen. Die Methode gibt ein JSON Objekt zurück, welches die aktuelle Reise enthält. So kann man im Dashboard als auch in der Anzeige die aktuelle Reise anzeigen. Sie sieht wie folgt aus:

```

1      {
2          "id": 1,
3          "location": " Norditalien",
4          "maxPersonAllowed": 100,
5          "type": "Kulturwoche",
6          "participant": [
7              {
8                  "id": 1,
9                  "firstname": "Oliver",
10                 "lastname": "Sugic",
11                 "role": "STAFF",
12                 "telephone": "0123456789",
13                 "comment": "No comment"
14             }
15         ],
16         "topics": [
17             {
18                 "id": 1,
19                 "name": "Mailand",
20                 "activity": [
21                     {
22                         "id": 2,
23                         "activityName": "Koeniglicher Palast",
24                         "startDateTime": "2023-05-11T16:00:00",
25                         "longitude": 9.1911,
26                         "latitude": 45.4632,
27                         "previousActivity": {
28                             "id": 1,
29                             "activityName": "Maeilaender Dom",
30                             "startDateTime": "2023-05-11T13:00:00",
31                             "longitude": 9.191926,
32                             "latitude": 45.464098,
33                             "previousActivity": null,
34                             "comment": "Die Liste ..",
35                             "publicationDate": "2023-03-30",
36                             "public": true
37                         },
38                         "comment": "Kunstinteressierte s..",
39                         "publicationDate": "2023-03-30",
40                         "public": false
41                     }
42                 ],
43                 "previousTopic": null,
44                 "comment": "Mailand ist .."
45             }
46         ],
47         "planedStartTime": "2022-05-19T09:30:00",
48         "planedEndTime": "2023-05-15T00:00:00",
49         "currentEvent": true
50     }

```

Die Kommentare zu den jeweiligen Entitäten wurden gekürzt, um die Übersichtlichkeit zu gewährleisten.

3.1.3 List All Events

Für die Liste aller Reisen nutze man **GET http://localhost:8080/api/event/getAll** Request. Die Methode gibt ein JSON Array zurück, welches alle Reisen enthält. Die Struktur sieht wie folgt aus:

```

1   [
2     {
3       "id": 1,
4       "location": " Norditalien",
5       "maxPersonAllowed": 100,
6       "type": "Kulturwoche",
7       "participant": [
8         {
9           "id": 1,
10          "firstname": "Oliver",
11          "lastname": "Sugic",
12          "role": "STAFF",
13          "telephone": "0123456789",
14          "comment": "No comment"
15        },
16        {
17          "topics": [
18            {
19              "plannedStartTime": "2022-05-19T09:30:00",
20              "plannedEndTime": "2023-05-15T00:00:00",
21              "currentEvent": true
22            },
23            {
24              "id": 2,
25              "location": " Norditalien",
26              "maxPersonAllowed": 100,
27              "type": "Kulturwoche",
28              "participant": [
29                {
30                  "id": 1,
31                  "firstname": "Max",
32                  "lastname": "Musterman",
33                  "role": "STAFF",
34                  "telephone": "0123456789",
35                  "comment": "No comment"
36                },
37                {
38                  "topics": [
39                    {
40                      "plannedStartTime": "2022-05-19T09:30:00",
41                      "plannedEndTime": "2023-05-15T00:00:00",
42                      "currentEvent": false
43                    }
44                  ]
45                }
46              ]
47            }
48          ]
49        }
50      ]
51    }
52  ]

```

Die Topic Entitäten sind hier herausgekürzt worden, da sie in dem Kontext nicht relevant sind.

Die Reisen werde hier im Dashboard sichtbar angezeigt.

The screenshot shows the LeoTour application interface. At the top, there's a blue header bar with the title 'LeoTour' and a 'Neue Reise anlegen' button. Below the header, there are three tabs: 'Alle Reisen' (which is active and highlighted in blue), 'Kulturwoche Norditalien', and 'Kulturwoche Süditalien'. The main content area displays two travel entries. The first entry is 'Kulturwoche Norditalien', which is described as 'Malland' and 'Turin' being known as the 'Metropole schlechthin, doch die italienische Stadt weiß, mit noch viel mehr zu beeindrucken.' The second entry is 'Kulturwoche Süditalien', described as 'Venedig' and 'Genoa'. At the bottom of each entry, there are two buttons: a blue one labeled 'Reise in der Schüleranreise anzeigen' and a grey one labeled 'Aktivitäten freischalten'.

Abbildung 28: Get All Example

4 Ausgewählte Aspekte

4.1 JsonManagedReference und JsonBackReference

Im Laufe der Entwicklung des Backend stößt man auf ein Problem, wo ich auf eine unendliche Rekursion zurückgab in meinen GET Requests. Das Problem bestand darin, dass mit bidirektionale Beziehungen das Datenmodell implementiert wurde. Das heißt, dass ich zwar eine Liste von Personen in der Event Klasse habe, ich aber auch ein Event Objekt in der Klasse Person habe. Somit entsteht bei der Serialisierung von einer Endlosschleife

Listing 7: Event Klasse

```
1  @Entity
2  public class Event extends PanacheEntityBase {
3
4      @Id
5      @GeneratedValue(strategy = GenerationType.IDENTITY)
6      private long id;
7      private String location;
8      private int maxPersonAllowed;
9      private String type;
10
11     @OneToMany(mappedBy = "event", cascade = CascadeType.ALL)
12     private List<Person> participant;
13
14     @OneToMany(mappedBy = "event", cascade = CascadeType.ALL)
15     private List<Topic> topics;
16
17     private LocalDateTime planedStartTime;
18     private LocalDateTime planedEndTime;
19
20     private boolean currentEvent;
21
22     public Event() {
23
24     }
25 }
```

Listing 8: Person Klasse

```

1  @Entity
2  public class Event extends PanacheEntityBase {
3
4      @Id
5      @GeneratedValue(strategy = GenerationType.IDENTITY)
6      private long id;
7      private String location;
8      private int maxPersonAllowed;
9      private String type;
10
11     @OneToMany(mappedBy = "event", cascade = CascadeType.ALL)
12     private List<Person> participant;
13
14     @OneToMany(mappedBy = "event", cascade = CascadeType.ALL)
15     private List<Topic> topics;
16
17     private LocalDateTime planedStartTime;
18     private LocalDateTime planedEndTime;
19
20     private boolean currentEvent;
21
22     public Event() {
23
24 }

```

Daher wurde nach einer Lösung gesucht und diese war schnell gefunden

- @JsonManagedReference
- @JsonBackReference

Diese ermöglichen es, dass die Serialisierung richtig stattfindet indem die Entität, welche @JsonManagedReference beinhaltet, serialisiert wird und die Entität, welche @JsonBackReference beinhaltet, ignoriert wird.[11]

5 Resümee

Während der Umsetzung der Diplomarbeit habe den durch Umgang mit neuen Technologien vieles neu dazu lernen können. Auch bei der Planung und Realisierung konnte ich viele Erfahrungen für mich mitnehmen. Blöderweise konnten das Projekt in ganzen unseren Vorstellung umgesetzt werden, das es seitens meiner Diplomarbeitspartner nicht mehr weiter verfolgt wurde. Das Projekt ist durchaus gut gelungen, für die harte Arbeit, die ich investiert habe.

Literaturverzeichnis

- [1] A. Sewell, „Wanderlog.” Online verfügbar: <https://wanderlog.com/blog/author/alicesewell/>
- [2] „TripIt,” 2006. Online verfügbar: <https://www.tripit.com/de/web/free>
- [3] N. Thomas, „Quarkus.” Online verfügbar: <https://blog.doubleslash.de/quarkus-einfach-erklaert>
- [4] „Quarkus.” Online verfügbar: <https://quarkus.io/>
- [5] A. Ashraf, „Google Maps Api.” Online verfügbar: <https://cloud.google.com/blog/products/maps-platform/build-faster-new-optimized-solutions-and-step-step-guidance?hl=en>
- [6] V. Agafonkin, „Leaflet.” Online verfügbar: <https://leafletjs.com/>
- [7] P. Liedman, „Leaflet Routing Machine,” 2015. Online verfügbar: <https://www.liedman.net/leaflet-routing-machine/>
- [8] D. Klassen, „Eine Identität für alles mit Keycloak.” Online verfügbar: <https://www.heise.de/ratgeber/Eine-Identitaet-fuer-alles-mit-Keycloak-3834525.html>
- [9] „Was ist Deployment?” Online verfügbar: <https://www.dev-insider.de/was-ist-deployment-a-0463e840e8b2b2372b3ba71865b875ca/>
- [10] M. Palmer, „Kubernetes Deployment Tutorial with Example YAML.” Online verfügbar: <https://matthewpalmer.net/kubernetes-app-developer/articles/kubernetes-deployment-tutorial-example-yaml.html#:~:text=What's%20the%20difference%20between%20a,using%20labels%20and%20label%20selectors.>
- [11] E. Paraschiv, „Jackson - Bidirectional Relationships.” Online verfügbar: <https://www.baeldung.com/jackson-bidirectional-relationships-and-infinite-recursion#infinite-recursion>

Abbildungsverzeichnis

1	Erstes Entwurf des Datenmodells	5
2	Überarbeiteter Entwurf des Datenmodell	6
3	Endgültiger Entwurf des Datenmodell	7
4	Quarkus Speicher Verbrauch [4]	8
5	Ergebnis der Implementierung mit Leaflet	10
6	Ergebnis der Implementierung mit Leaflet Routing Machine	11
7	Entwurf der Schüleransicht	12
8	Schüler Ansicht	13
9	Aktivitäten freischalten und sperren	14
10	Abschnitt für die Daten des Events	15
11	Abschnitt für die Daten der Teilnehmer	15
12	Abschnitt für die Daten des Themas	16
13	Abschnitt für die Daten der Aktivitäten	16
14	Neue Oberfläche für die Schüleransicht	17
15	Neue Oberfläche für die Schüleransicht	18
16	Neue Oberfläche für die Eingabe der Daten des Events	19
17	Neue Oberfläche für die Eingabe der Daten der Personen	19
18	Neue Oberfläche für die Eingabe der Daten der Themen	20
19	Neue Oberfläche für die Eingabe der Daten der Aktivitäten	20
20	Dashboard für die Lehrerinnen und Lehrer	21
21	Fehlschlagender Redirect nach dem Login	23
22	Kubernetes Dashboard	28
23	Kubernetes Services	28
24	Event Resource	29
25	Topic Resource	29
26	Activity Resource	29
27	Person Resource	30
28	Get All Example	32

Tabellenverzeichnis

Quellcodeverzeichnis

1	Implementierung einer Karte mit Leaflet	10
2	Implementierung einer Karte mit Leaflet Routing Engine	10
3	Implementierung im Angular	22
4	Github Actions für das Backend	25
5	Dockerfile für das Backend	26
6	Dockerfile für das Backend	27
7	Event Klasse	33
8	Person Klasse	34

Anhang

Besprechungsprotokoll 30.09.2022

Table 1. Teilnehmer

anwesend
Oliver Sugic
Prof. Thomas Stütz

Table 2. Ort und Zeit

Ort	Htl-Leonding
von-bis	Fr. 09.05.2022 12:40-13:00
Dauer	20 min

Besprochene Themen

- Erster Blick auf das backend
 - Kontrolle der Pom.xml
 - reactive dependencies entfernt
 - Flyway Migration besprochen
 - import.sql unnötig → V1.0.0_leotour-backend.sql macht alles
 - Endpoints haben keine richtig Namenskonventionen
 - Kann geändert werden, muss aber nicht erfolgen
 - Blick auf Tests
- Nächsten Schritte
 - Usecase diagramme erstellen
 - für die Testung mit Karate

Besprechungsprotokoll 14.10.2022

Table 1. Teilnehmer

anwesend
Oliver Sugic
Prof. Thomas Stütz

Table 2. Ort und Zeit

Ort	Htl-Leonding
von-bis	Fr. 14.10.2022 17:15-17:35
Dauer	20 min

Besprochene Themen

- Routing ist in der Webapplikation nicht notwendig
 - Die Zielkoordinaten (oder Adresse) werden an Google Maps übergeben
 - Das Routing erfolgt innerhalb Google Maps
 - TODO: Wie ruft man aus Angular die Google Maps auf mit Übergabe der Koordinaten auf
- Test Daten ändern
 - Entsprechend von realen Daten (z.B. Kroatienwoche)



Figure 2. Wireframe Beispiel

- Glosar: Begriffe mit Bsp anführen
- Karte wegen Platzprobleme entfernen