TravelPal

Itinerary planner for travel & everyday events

Benedict Chukwubuikem Benson-Obilom
20447376

Final Year Project – **2023**
B.Sc. Single/Double Honours in
Computer Science/Computer Science and Software Engineering

Department of Computer Science
Maynooth University
Maynooth, Co. Kildare
Ireland
A thesis submitted in partial fulfilment of the requirements for the B.Sc. Single/Double Honours in Computer Science/Computer Science and Software Engineering.
Supervisor: **Joseph Duffin**.

Contents

Declaration

I hereby certify that this material, which I now submit for assessment on the program of study as part of **Computer Science & Software Engineering** qualification, is *entirely* my own work and has not been taken from the work of others - save and to the extent that such work has been cited and acknowledged within the text of my work.

I hereby acknowledge and accept that this thesis may be distributed to future final year students, as an example of the standard expected of final year projects.

Signed: BenB Date: 09/11/2023

Acknowledgements

I want to take this opportunity to extend my deepest gratitude to my supervisor, Joseph Duffin, for his continuous support, invaluable advice, and the belief he placed in me. His expertise and insights have not only given me guidance in this report but also for skills I wish to apply in the future.

A special mention must also go to Vanush Misha Paturyan, the CS department server's admin, whose expertise was instrumental in deploying TravelPal. His patience and willingness to tackle the intricacies of hosting on the servers played a crucial role in the project's success.

I am also immensely thankful for the continuous encouragement and understanding from my family, friends, and girlfriend. Their belief in me provided the strength and motivation needed during the toughest periods of this project.

Abstract

TravelPal, developed using Spring Boot and React, targets simplifying travel planning and itinerary management amidst the complexity of existing applications. This project started on creating a user-friendly platform, focusing on simplifying the itinerary management process. It separates itself from the feature-heavy crowd of travel planning applications, focusing in on simplicity and efficiency. This thesis details TravelPal's development journey, discussing the motivation, the technical background, the problem identification, solution strategy, the conclusion, and achievements, including the deployment on the Maynooth Computer Science department servers. Through testing and applying user feedback, TravelPal refined its functionality and design, ensuring a balance between technical reliability and user experience. The evaluation chapter backs up the application's success in usability and effectiveness, through both technical verification and user satisfaction surveys. Future enhancements aim to expand features while maintaining the core focus on simplicity. This exploration contributes to the field by demonstrating the impact of a focused, simplistic approach in travel planning application development, setting an example for future innovations.

Keywords: TravelPal, travel planning, Spring Boot, React.

List of Figures

List of Tables

Chapter one: Introduction

Summary

Chapter one introduces TravelPal, an application designed with Spring Boot and React to make travel planning and itinerary management straightforward and user-friendly. This chapter talks about the initial inspiration and motivation behind the project, the problem statement it aims to address, the approach taken to develop the application, the metrics for success, and the project achievements.

1.1 Topic addressed in this project

TravelPal was inspired by the need for a simplified travel planning experience. By focusing solely on managing itineraries, the project separates itself from existing travel applications that are overly complex.

1.2 Motivation

When looking at some travel apps out there, they try to do too much. They're packed with features, buttons, and options that can become overwhelming, and even take the fun out of planning a trip.

TravelPal was initially ambitious and wanted to go down the same road to create an all-in-one app that could handle everything. It aimed to provide users with the tools to plan trips, create dynamic itineraries, customize routes with real-time information, access destination-specific weather forecasts, etc. But shared experiences of frustration with the "jack of all trades" approach show that this can make the app less efficient and overwhelming for users.

This realization sparked a change in direction. The final year project journal shows a shift from trying to include everything, to focusing on a more focused and simplistic approach (See Appendix 2B). The class diagrams became less cluttered, the features more focused, and the user interface vision became clearer.

1.3 Problem statement

In developing TravelPal, the technical problem is to create a focused travel planning application that provides a user-friendly experience. TravelPal aims to address the complexity and loss of focus that can be common in travel planning apps.

1.4 Approach

The development of TravelPal involved refining the application’s functionality and design through active testing and user feedback. Using API testing tool Insomnia and focusing on a responsive frontend design, the project achieved a nice balance between user experience and application functionality.

1.5 Metrics

The success of TravelPal is measured by its usability, user-friendliness, and how effective its key features are. These are evaluated through a user feedback survey which is discussed extensively in Chapter five. The application’s design and backend structure were developed with a focus on these user-focused metrics.

1.6 Project Achievements

Throughout the TravelPal journey several achievements were marked, each contributing to the final goal.

- **Simplified Travel Planning Process**: A simple, user-friendly interface that allows users to create and manage their travel itineraries was developed.

- **Efficient System Design**: A SpringBoot backend, supporting CRUD operations for managing user itineraries and profiles was developed. This backend, complimented by a responsive React.js frontend, ensures the efficiency and accessibility of TravelPal across a variety of devices (See Appendix 2C for the backend code documentation).

- **Deployment on the Maynooth CS dept Servers**: One of the best moments for TravelPal was achieving its deployment and hosting on the Maynooth CS dept servers, a process tackled through detailed collaboration with Vanush Misha Paturyan (CS dept servers admin). The initial discussions revolved around Docker and introducing containerization to simplify deployment. The transition required relocating the project repository from GitHub to the Maynooth GitLab. The project structure was tackled, optimizing the organization of frontend and backend components within a single repository for efficiency. A crucial step involved creating a text file with mock API requests, mirroring Insomnia's functionality to validate the server's responsiveness, the server's admin would use this to test TravelPal's API. Throughout these meetings, a lot of knowledge was gained in the importance of having dynamically adjustable URLs in an application through environmental variables to accommodate the hosting environment. The detailed records of these meetings, which document our approach and the technical solutions implemented, can be found in Appendix 2E. By the conclusion of these meetings saw the full TravelPal application successfully hosted and connected to the Gitlab repository (Find TravelPal's application link in Appendix 2E).

- **Feedback Integration**: A user feedback survey was integrated within the application in the form of a link at the bottom of the Dashboard, About page and Settings page. This allows for the collection of real-time user feedback which is pivotal in guiding ongoing improvements and feature additions to TravelPal. (See Appendix 2F for full results)

Chapter two: Technical Background

Summary

This chapter discusses the theoretical and technical foundations supporting TravelPal. It contains an evaluation of TravelPal’s position amongst existing travel planning applications. Additionally, it details the technical material that powers the application, discussing the rationale behind their selection.

2.1 Topic material

2.1.1 Existing Travel Planning Solutions

Travel planning applications such as TripAdvisor and Wanderlog have shaped user expectations within the travel planning applications, offering an abundance of features including itinerary planning, weather forecast integration, transport, booking services etc. Despite their diversity, these platforms often suffer from feature overload, complicating the user experience. In response, TravelPal emphasizes a focus on itinerary creation and simplicity, therefore addressing the confusion and frustration experienced by users navigating some of these over-saturated platforms.

2.2 Technical material

TravelPal is built with a combination of modern software development frameworks, programming languages, tools, and libraries. Below, I detail the important components of TravelPal's development and their roles in the project.

2.2.1 Spring Boot for Backend Development

Spring Boot serves as the backbone for TravelPal's backend, selected for its reliable ecosystem and simple development process. Features like dependency injection and auto-configuration support efficient RESTful API development for managing users and itineraries. My choice of Spring Boot was influenced by its widespread use in the industry and by foundational knowledge acquired through a comprehensive tutorial by (Amigoscode, 2021b) and practical experience from a previous internship.

2.2.2 React for Frontend Development

React's component-based architecture drives TravelPal's frontend, supporting the creation of a dynamic and responsive user interface. This decision aligns with the project's aim to deliver an intuitive travel planning experience, where React's capabilities allow for seamless updates and user interactions without the need for full page reloads. The selection of React, like SpringBoot was influenced by its industry popularity for building efficient web applications. My skills in React were built by engaging with a paid course by Udemy (Udemy & Schwarzmüller, 2023). This educational resource, coupled with practical exercises, equipped me with the necessary skills to start using React in developing TravelPal's frontend.

2.2.3 PostgreSQL for Data Management

PostgreSQL serves as the database backbone for TravelPal, selected for its reliability, performance, and advanced data management capabilities. It stores and manages all TravelPal’s data, including user and itinerary information. A notable aspect of employing PostgreSQL was the discovery that the term "user" is a reserved keyword within the database system. This led to a necessary adjustment, renaming the model and table from "user" to "client" to ensure compatibility and avoid potential conflicts within the database structure (Overflow, 2014).

2.2.4 Insomnia for Testing

Insomnia is used for testing TravelPal's backend, ensuring that the RESTful APIs function correctly across all CRUD operations. Selected due to previous internship experience, this tool supports comprehensive testing of the application's server-side logic, validating the integration between the frontend and backend.

2.2.5 Git, GitLab, and GitHub

Version control and collaborative development are managed through Git, with GitLab hosting the main repository and GitHub serving as a platform for documentation and planning resources, including the project journal and an early development roadmap. This setup ensures efficient, organized progression and documentation of the project.

Additional Tools and Libraries:

- **Figma:** Used for designing UI mock-ups of an early landing page and dashboard design. Figma's intuitive tools supported the iterative design process of TravelPal's frontend. (Flux Academy, 2022; Rino de Boer, 2023)
- **Axios:** A promise-based HTTP client used for making requests from the React frontend to the Spring Boot backend, enabling smooth data exchange between the server and client-side of the application. (Dave Gray, 2021)
- **React Router:** Used for routing in the React application, ensuring users can navigate through different parts of the application without reloading the page. (Ninja, 2023)
- **Spring Data JPA:** Simplifies data access operations in the backend, reducing the amount of boilerplate code required for data access layers. (Amigoscode, 2021b)
- **Doxygen:** Used for generating code documentation, Doxygen makes the project's structure and functions accessible for review and further development. See Appendix 2C for link to code documentation. (HurrayBanana, 2016)

Table 2.1 Technology Implementation Overview

Technology	Purpose in TravelPal
Spring Boot	Backend dev, API creation, server-side logic, security
React	Frontend dev, UI
PostgreSQL	Data storage and management
Insomnia	API testing and validation
Git/GitLab	Version control, repo hosting
GitHub	Documentation & project planning
Figma	UI design and prototyping tool
Doxygen	Code documentation generation
Axios	Facilitates HTTP requests for client-server communication
React Router	Enables smooth navigation within the frontend application
Spring Data JPA	Simplifies backend data access

Table 2.2 Data Sources

Source	Type	Purpose in TravelPal
PostgreSQL	Internal Database	Stores user accounts and itinerary details

Chapter three: The Problem

Summary

This chapter presents various UML diagrams, including Sequence Diagrams (Cognitive Programmer, 2020), a System Architecture Diagram (Random code, 2021b), and a Class Diagram (CS Hero, 2023), to fully illustrate the system architecture and workflows. All diagrams were developed with PlantUml. (Vaughan, 2013; Cognitive Programmer, 2020)

3.1 Project UML documentation

This section visually models the system architecture and the user's journey through the application using UML diagrams. These diagrams allow exploration of the functionality and communication pathways between system components, showing user interactions from arriving at the landing page to logging out. The TravelPal UML documentation includes:

3.1.1 Use Cases

Use cases are used for highlighting the core interactions between a user and the TravelPal application, creating the context for the sequence diagrams.

1. User navigates to the Landing Page
2. User is redirected to the Login/Register page from the Landing Page
3. User enters credentials and Logs in or Registers
4. User views the Dashboard and their Itineraries after Login
5. User views Settings/About pages from the Dashboard
6. User views and creates an Itinerary from the Dashboard
7. User Signs out

3.1.2 Sequence Diagrams

Sequence Diagrams are used in conjunction with the Use Cases to show the system communication pathways.

1. User Navigates to Landing Page
2. User is redirected to the Login/Register page from the Landing Page

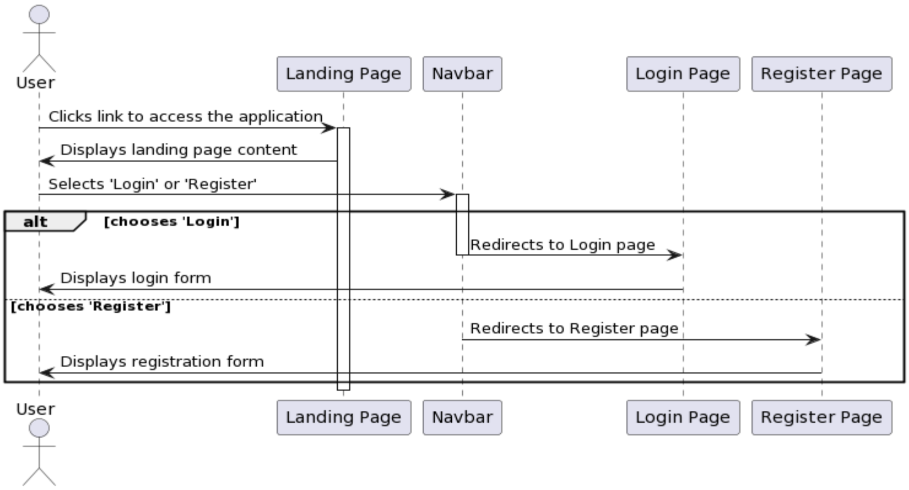


Figure 3.1 UML Sequence diagram for Use Case 1 & 2.

3. User enters credentials and Logs in or Registers
4. User views the Dashboard after Login
5. User views Settings/About pages from the Dashboard

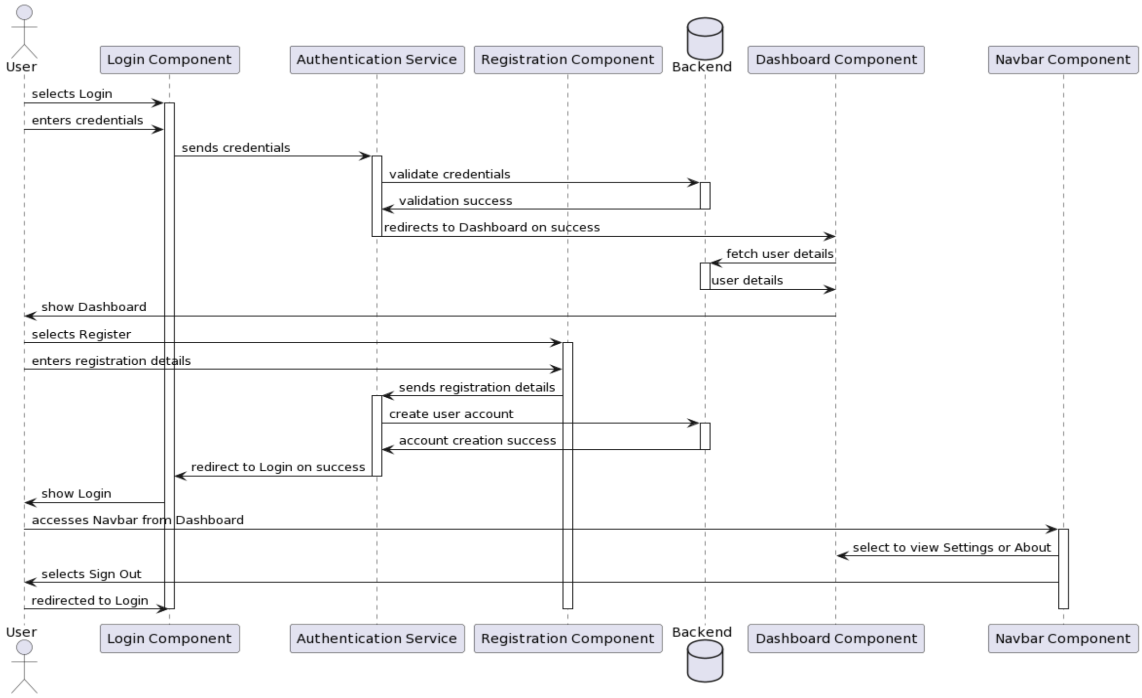


Figure 3.2 UML Sequence diagram for Use Case 3, 4 & 5.

6. User views and creates an Itinerary from the Dashboard

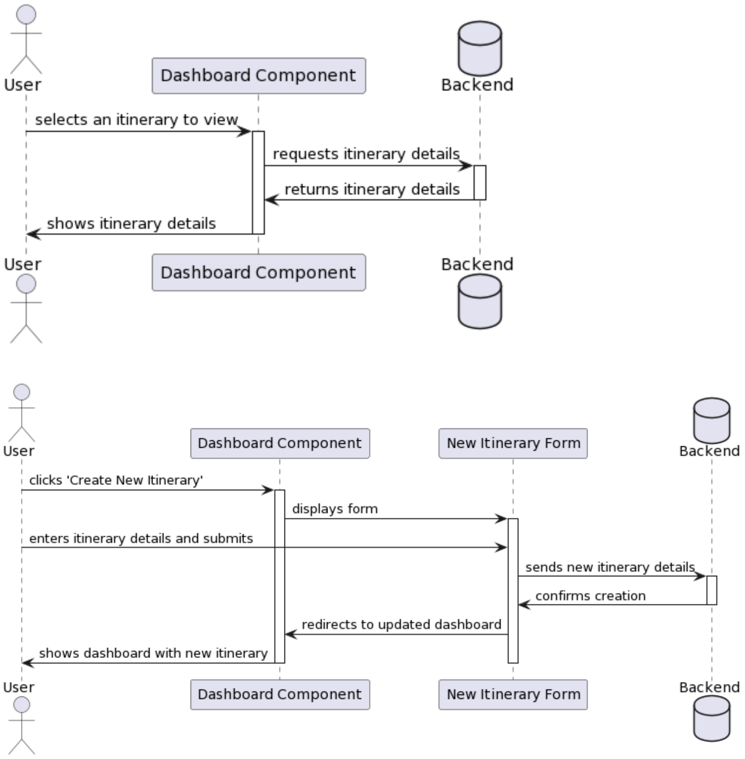

User
Dashboard Component
Backend
selects an itinerary to view
requests itinerary details
returns itinerary details
shows itinerary details
User
Dashboard Component
New Itinerary Form
Backend
clicks 'Create New Itinerary'
displays form
enters itinerary details and submits
sends new itinerary details
confirms creation
redirects to updated dashboard
shows dashboard with new itinerary

Figure 3.3 & 3.4 UML Sequence diagrams for Use Case 6.

7. User Signs out

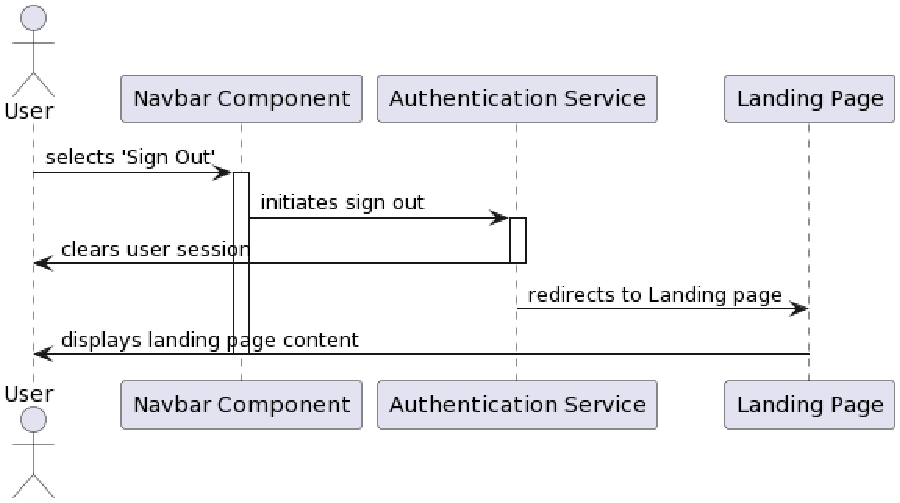


Figure 3.5 UML Sequence diagram for Use Case 7.

3.1.3 System Architecture Diagram

This diagram presents the architecture of the TravelPal system, including its deployment on the CS Department's servers.

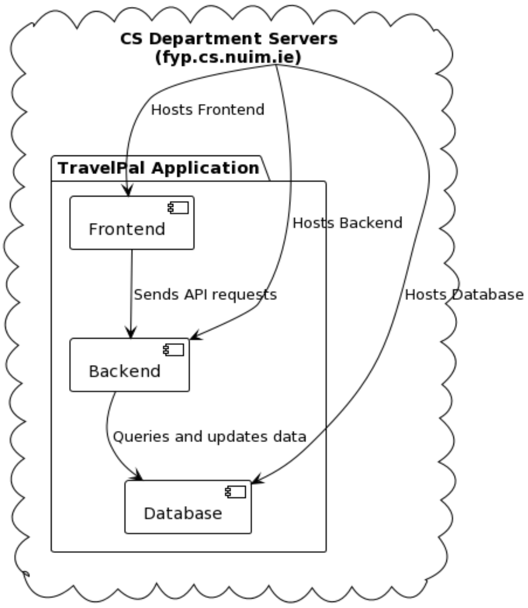


Figure 3.7 System diagram for TravelPal.

3.1.4 Class Diagram

This diagram shows the structure and relationships of the core entities/models in the application. Displaying the object-oriented design.

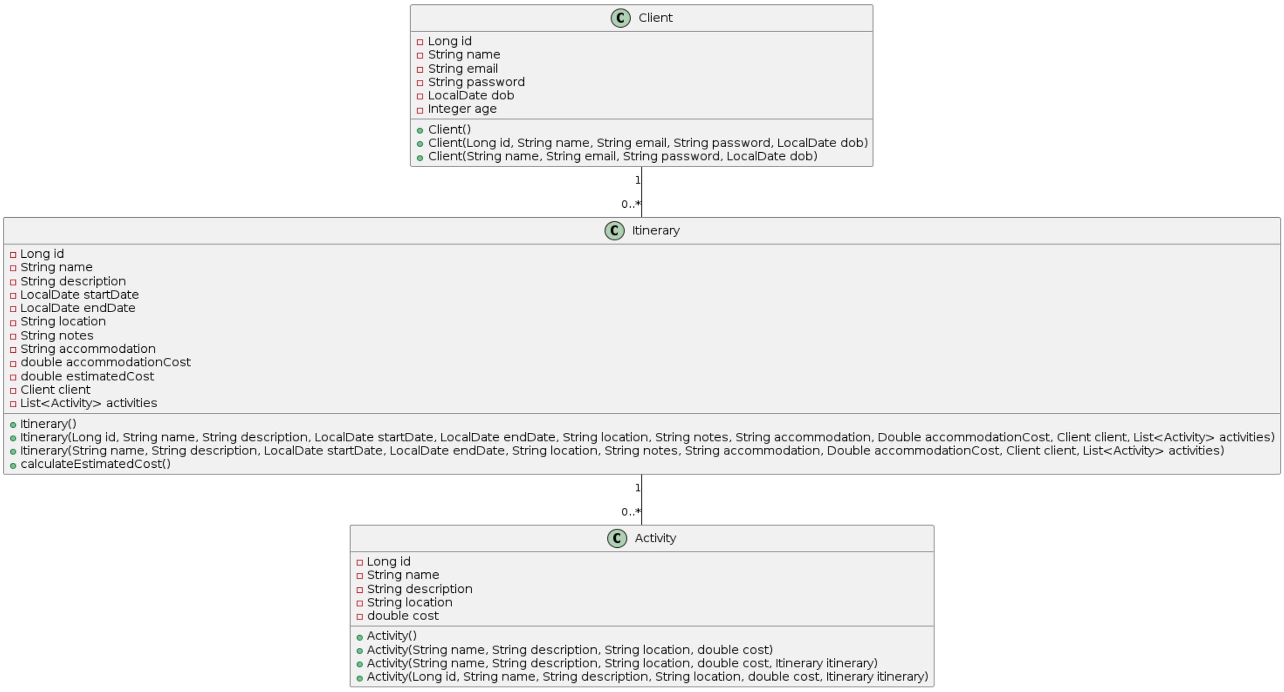

C Client
Long id
String name
String email
String password
LocalDate dob
Integer age
Client()
Client(Long id, String name, String email, String password, LocalDate dob)
Client(String name, String email, String password, LocalDate dob)
1
0..*
C Itinerary
Long id
String name
String description
LocalDate startDate
LocalDate endDate
String location
String notes
String accommodation
double accommodationCost
double estimatedCost
Client client
List<Activity> activities
Itinerary()
Itinerary(Long id, String name, String description, LocalDate startDate, LocalDate endDate, String location, String notes, String accommodation, Double accommodationCost, Client client, List<Activity> activities)
Itinerary(String name, String description, LocalDate startDate, LocalDate endDate, String location, String notes, String accommodation, Double accommodationCost, Client client, List<Activity> activities)
calculateEstimatedCost()
1
0..*
C Activity
Long id
String name
String description
String location
double cost
Activity()
Activity(String name, String description, String location, double cost)
Activity(String name, String description, String location, double cost, Itinerary itinerary)
Activity(Long id, String name, String description, String location, double cost, Itinerary itinerary)

Figure 3.8 UML class diagram for TravelPal.

Additional UML diagrams can be found in Appendix 3 & All User Stories in supporting documents on Moodle.

3.2 Problem analysis

TravelPal is a response to the complexity problem in travel planning. Existing apps often overwhelm users with an excess of features and complicated user interfaces, which can take the fun out of travel planning. This section analyses key issues in travel planning applications such as usability which is concerned with users not being able to fully engage with the application because of its complexities and feature overload, and the technical challenges that go into developing an efficient application.

Chapter four: The Solution

Summary

This chapter dives into the development journey of TravelPal. It breaks down the design approach, chosen technologies, application structure and features that shaped the development of the application.

4.1 Design Approach

The focus of TravelPal's design was to blend an aesthetically pleasing webpage with user-friendly functionality. The design process prioritized the user experience and user interface to ensure that users could navigate the app with ease and enjoyment. The webpage aspired for bright consistent colours as it uses a consistent dark and light blue theme throughout. It also uses clear text, and buttons, making sure that the overall design is inviting and intuitive. A custom logo icon and the TravelPal name was added to the head element of the application through the index.html file in the React configuration. This gives the webpage more of a unique and clean feel.

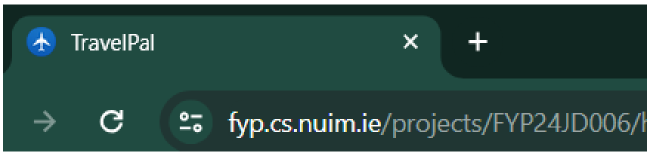
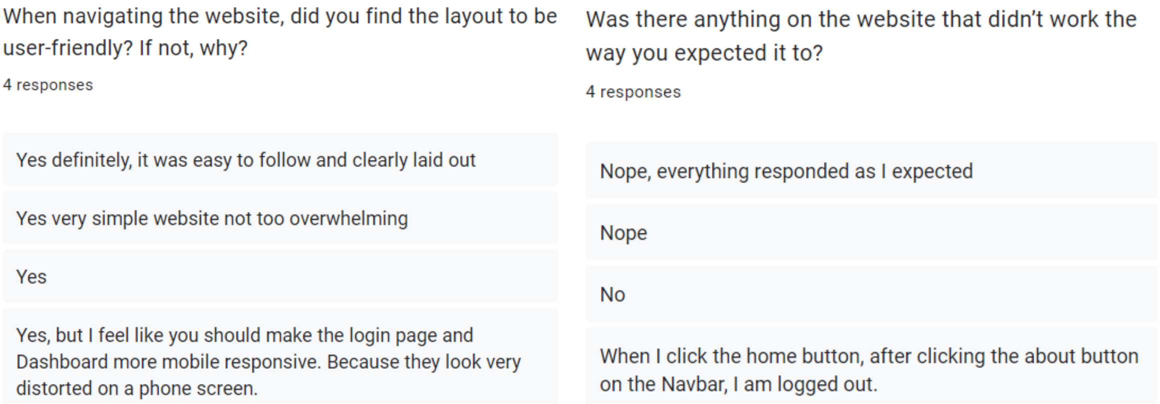


Figure 4.1 Snippet of custom webpage logo and name

User experience testing in the form of feedback surveys played a massive role in the early development of TravelPal and was instrumental in some of the key features that were added later in development. This is expanded upon in Chapter five: Evaluation.



When navigating the website, did you find the layout to be user-friendly? If not, why?

4 responses

Yes definitely, it was easy to follow and clearly laid out

Yes very simple website not too overwhelming

Yes

Yes, but I feel like you should make the login page and Dashboard more mobile responsive. Because they look very distorted on a phone screen.

Was there anything on the website that didn't work the way you expected it to?

4 responses

Nope, everything responded as I expected

Nope

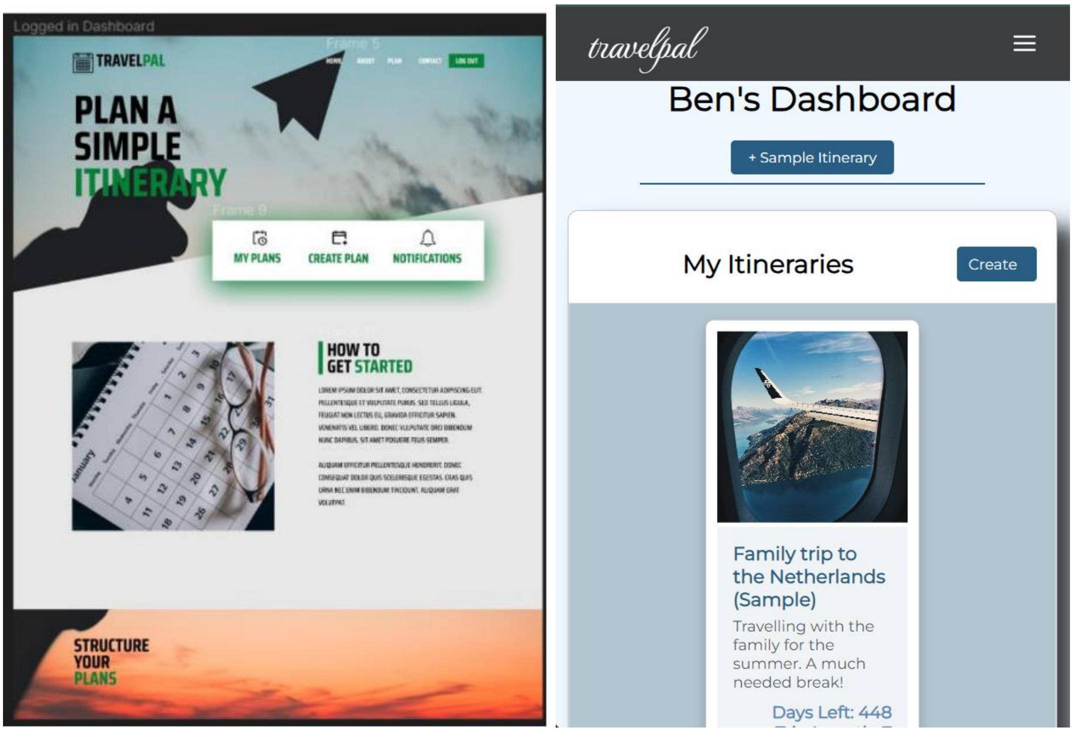
No

When I click the home button, after clicking the about button on the Navbar, I am logged out.

Figure 4.2 & 4.3 Snippets of early user feedback from the TravelPal survey

4.1.1 Figma Mock-up

The design phase began with mock-ups created using Figma. Using YouTube tutorials to help get a base understanding of Figma, pre and post login pages were created. This ended up being used as major inspiration for TravelPal's Landing page and Navbar. These early designs were instrumental in visualizing the app's layout, giving guidance to the pathway vison from landing page to itinerary creation.



PLAN A SIMPLE ITINERARY

HOW TO GET STARTED

LOREM IPSUM DOLOR SIT AMET, CONSECTETUR ADIPISCING ELIT. PELLENTESQUE ET VOLPUTATE PURUS. SED TELLUS LIGULA, FEUGIAT NON LECTUS EU, GRAVIDA EFFICITUR SAPIEN. VENENATIS VEL LIBERO. DONEC VULPUTATE ORCI BIBENDUM NUNC DAPIBUS, SIT AMET POSUERE FELIS SEMPER.

ALIQUAM EFFICITUR PELLENTESQUE HENDRERIT. DONEC CONSEQUAT DOLOR QUIS SCELERISQUE EGESTAS. CRAS QUIS URNA NEC ENIM BIBENDUM TINCIDUNT. ALIQUAM ERAT VOLUTPAT.

STRUCTURE YOUR PLANS

Ben's Dashboard

My Itineraries

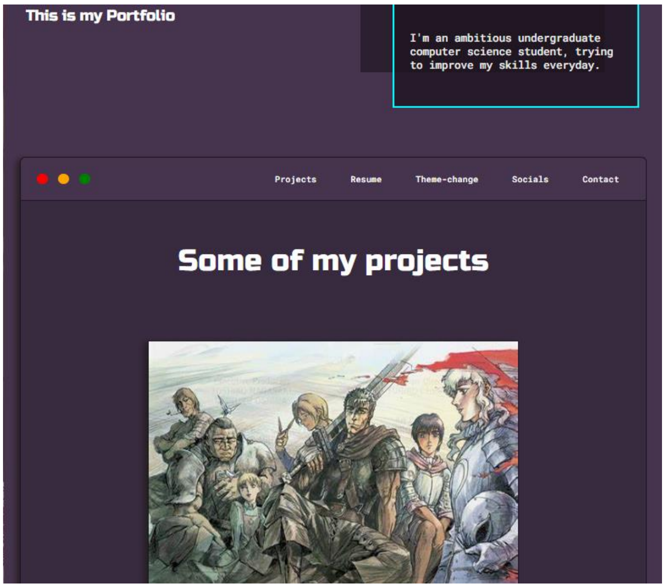
Family trip to the Netherlands (Sample)

Travelling with the family for the summer. A much needed break!

Figure 4.4 & 4.5 Early Figma mock-up vs Final Dashboard design (mobile mode)

4.1.2 Design Influences from Previous Projects

The structural and visual design of TravelPal's Dashboard was heavily influenced by a personal portfolio project completed in the summer of 2022 (Benson-Obilom, 2022). The portfolio, which was developed using vanilla HTML, CSS, and JavaScript, was hosted on GitHub pages (Benson-Obilom, 2022b). The main inspiration from the portfolio was taken from its Hero section and its floating mini browser that holds past projects. This element was not only aesthetically pleasing but also an effective way of managing an itinerary list for TravelPal.



This is my Portfolio

I'm an ambitious undergraduate computer science student, trying to improve my skills everyday.

Some of my projects

Figure 4.6 My personal portfolio Hero section

4.2 Development Tools and Environment

This section discusses the tools and environments used for developing TravelPal.

4.2.1 Programming Languages

JavaScript with its React library for enhancement, was chosen over libraries like Angular and Vue.js because of its massive community support in general and the support amongst other students doing their final year projects. Java, enhanced by SpringBoot, was chosen over other backend development options because of past internship experience with the framework. PostreSQL was chosen for data management because of its well-established data integrity and query functionality.

4.2.2 Integrated Development Environment (IDE)

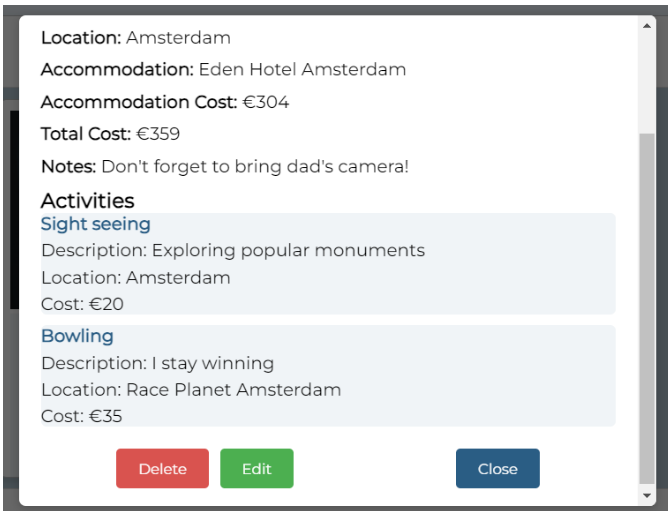
Visual Studio Code and Intellij were chosen for their reliability and abundance of features for supporting the programming languages used in TravelPal's development. Previous internship experience with Intellij and portfolio project development experience with Visual Studio Code helped speed up the development process.

4.3 Application Features

This section dives into some of the core features in the application and the decisions made during their implementation.

4.3.1 Itineraries

Itinerary creation and management are at the very core of TravelPal. Using a floating panel for displaying itineraries, the application displays a modal pop up when a user interacts with their itinerary. This modal displays the itinerary details along with bringing the applications crud functionality to life by allowing the user to delete or update their itinerary. Refer to Figure 4.5 & 4.7.



Location: Amsterdam

Accommodation: Eden Hotel Amsterdam

Accommodation Cost: €304

Total Cost: €359

Notes: Don't forget to bring dad's camera!

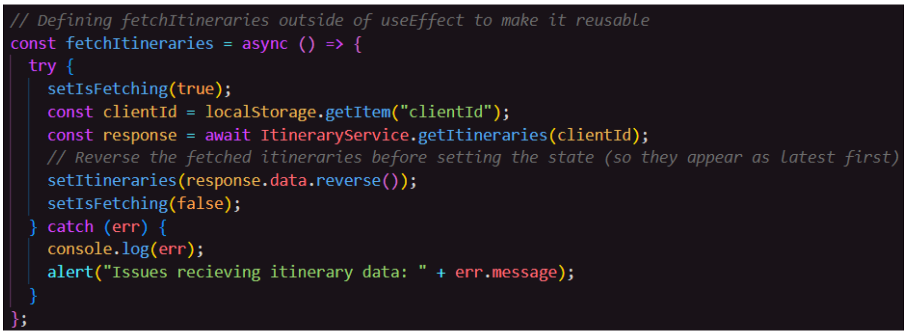
Activities

Sight seeing
Description: Exploring popular monuments
Location: Amsterdam
Cost: €20

Bowling
Description: I stay winning
Location: Race Planet Amsterdam
Cost: €35

Figure 4.7 Itinerary details modal interface.

In the React code, the ItineraryList component acts as a repository for the users' itinerary data. One of the unique aspects of the ItineraryList component, is the logic that prioritizes the display of newer itineraries. Refer to Figure 4.8.



```
// Defining fetchItineraries outside of useEffect to make it reusable
const fetchItineraries = async () => {
  try {
    setIsFetching(true);
    const clientId = localStorage.getItem("clientId");
    const response = await ItineraryService.getItineraries(clientId);
    // Reverse the fetched itineraries before setting the state (so they appear as latest first)
    setItineraries(response.data.reverse());
    setIsFetching(false);
  } catch (err) {
    console.log(err);
    alert("Issues recieving itinerary data: " + err.message);
  }
};
```

Figure 4.8 Code snippet of the ItineraryList component’s reverse list logic.

To create an itinerary, the user must click the create (itinerary) button which opens a modal dial interface (see button in Figure 4.5). Once the user fills and submits the form, a new itinerary is created uniquely for that user. Refer to Figure 4.9.

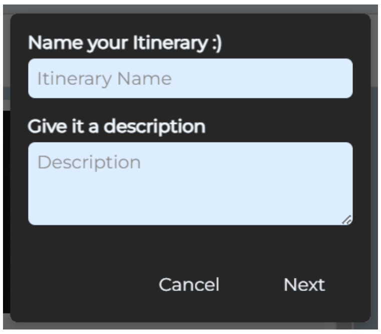
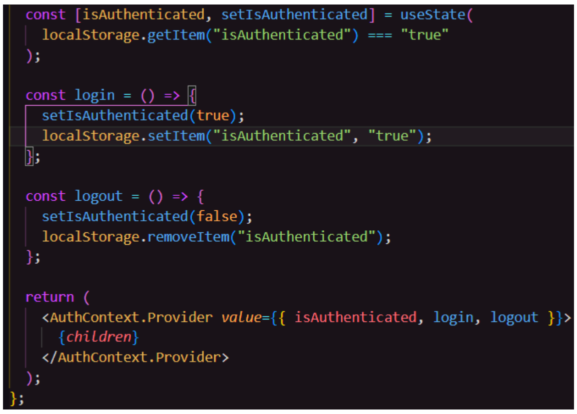


Figure 4.9 Modal dial used as the Itinerary creation interface.

4.3.2 Login/Sign-up

The late implementation of stable authentication in TravelPal was achieved through the introduction of the AuthContext component in React (Solutions, 2023, 2024). The core of the AuthContext component is the use of functions called login and logout to set the authentication status of a user (see Figure 4.10). These functions are used by the Login and Register components to set the user’s state. The ProtectedRoute component uses AuthContext to restrict access to certain application routes i.e., URL’s, to only authenticated users, redirecting unauthenticated users to the login page. See Figure 4.11.

The responsive design of the Login and Sign-up pages a long with the authentication method, enhances the user’s security and general navigation through the page.

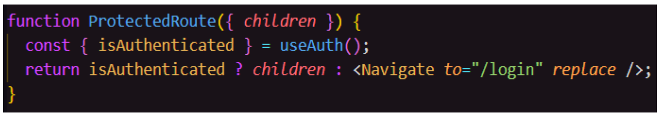


```
  const [isAuthenticated, setIsAuthenticated] = useState(
    localStorage.getItem("isAuthenticated") === "true"
  );

  const login = () => {
    setIsAuthenticated(true);
    localStorage.setItem("isAuthenticated", "true");
  };

  const logout = () => {
    setIsAuthenticated(false);
    localStorage.removeItem("isAuthenticated");
  };

  return (
    <AuthContext.Provider value={{ isAuthenticated, login, logout }}>
      {children}
    </AuthContext.Provider>
  );
};
```



```
function ProtectedRoute({ children }) {
  const { isAuthenticated } = useAuth();
  return isAuthenticated ? children : <Navigate to="/login" replace />;
}
```

Figure 4.10 & 4.11 AuthContext & ProtectedRoute code snippets.

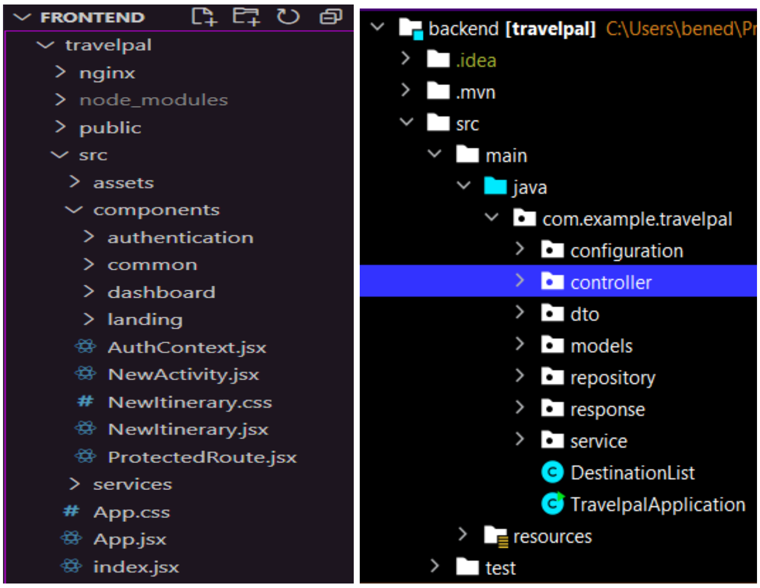
4.3.3 Responsive Design and Page Navigation

TravelPal’s navigation structure uses React Router, which enhances the single-page application’s user flow by avoiding server requests when unnecessary. The webpage’s responsive design shows in the Navbar changing to a hamburger menu, as well as the create itinerary button’s position being repositioned to the top of the itinerary manager on mobile devices. Refer to Figure 4.5.

See Figure 3.2 and 3.5 for sequence diagrams that showcase how the user can use the Navbar to navigate through the application.

4.4 Application structure

TravelPal's file and folder structure ensures the application's maintainability as it grows. Clearly structuring core features of the frontend and backend as seen in Figure 4.12.



```
FRONTEND
  travelpal
    > nginx
    > node_modules
    > public
    src
      > assets
      components
        > authentication
        > common
        > dashboard
        > landing
        AuthContext.jsx
        NewActivity.jsx
        NewItinerary.css
        NewItinerary.jsx
        ProtectedRoute.jsx
      > services
      App.css
      App.jsx
      index.jsx
```

```
backend [travelpal] C:\Users\bened\Pr
  > .idea
  > .mvn
  src
    main
      java
        com.example.travelpal
          > configuration
          > controller
          > dto
          > models
          > repository
          > response
          > service
          DestinationList
          TravelpalApplication
      > resources
    > test
```

Figures 4.12 & 4.13 React & SpringBoot File structure.

For a full view of the system architecture, refer to Figure 3.7.

4.5 Challenges and Solutions

The TravelPal journey had its fair share of challenges, each met with solutions that refined the final product. This section is a collection of the core problems and resolutions encountered throughout the project's journey.

4.5.1 Project and Supervision Search

The first two weeks of the project journey set the stage with seeking a project idea and a supervisor. An early setback had been encountered when the original project choice of a weather forecast application using SpringBoot had already been assigned. The resolution came in taking inspiration from the initial idea and proposing a new idea to a different lecturer, leading to the birth of TravelPal with Joe Duffin as supervisor.

4.5.2 Frontend Development with React

One of the major concerns before starting this project was the lack of experience with the React.js library. As a solution, Week 6 was devoted to strengthening foundational knowledge in web development through a course with freeCodeCamp (freeCodeCamp.org, 2021) and Scrimba (Scrimba & Borgen, 2021) for JavaScript and the Sololearn (Sololearn, 2014) course for HTML and CSS. This process was crucial for gaining a better understanding of React.

From Week 9 to 10, proactive learning of React through tutorials and a paid Udemy course (covering the core/essential features through a project), allowed for development of a modern and responsive landing page for TravelPal.

4.5.3 Deployment on the CS dept. servers

In the final weeks, collaboration with the department's server administrator, Vanush Misha Paturyan, to dockerize and fully host the project intensified at its last stages, marked by back-to-back meetings. The frontend was already being hosted, but the SpringBoot backend was proving to be more complex.

This phase was pivotal, offering a deep dive into the deployment nuances, particularly the pivotal role of environmental variables and dynamic URLs, as these were the keys to resolving the issues and successfully getting the whole application hosted.

See section 1.6 for a more comprehensive look at this journey.

4.5.4 Development of a User-Focused Dashboard and Itinerary Management

Throughout the development process, ensuring that the dashboard and itinerary management were user-focused was challenging, but pivotal. This was achieved by integrating CRUD operations, sample itinerary generation for inspiration, and linking Dashboard’s as well as the Itinerary list to individual users, therefore personalising the user experience. See Figure 4.5 for the final Dashboard design, showcasing these features.

Chapter five: Evaluation

Summary

Chapter five describes the evaluation of TravelPal, bringing focus to the validation process undertaken for the project. The dynamic between technical testing and user feedback comes together to make a comprehensive evaluation of the application's backend reliability and frontend user satisfaction.

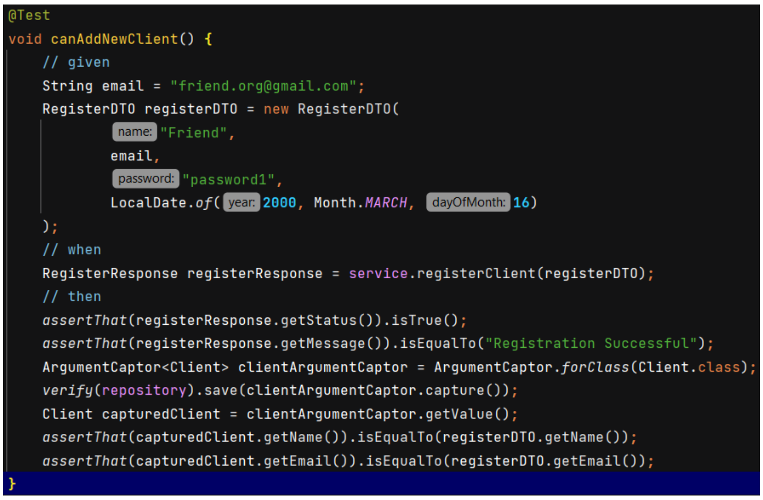
5.1 Technical Verification and Testing

5.1.1 Test Approach

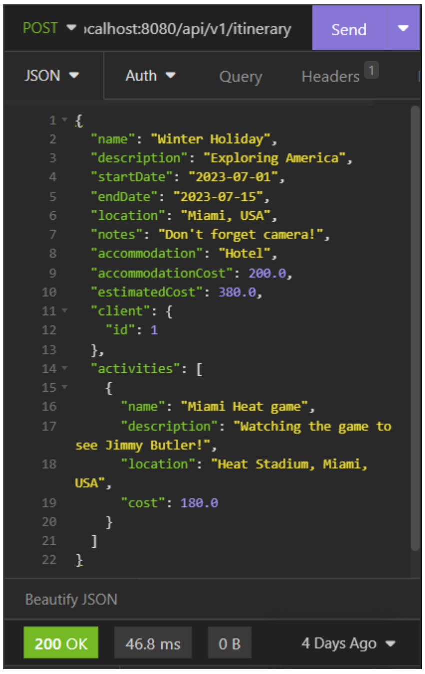
Focusing on a dual testing strategy, the project integrated unit tests to ensure the correctness of code in the backend and was complemented by API tests with Insomnia to verify the CRUD operations for the client-side functionalities. The unit tests consisted of tests for the logic of service and repository classes for the Client model.

5.1.2 Test Cases and Results

One of the unit tests verifies that the user can create a new account in the form of a client (see Figure 5.1). One of the API tests verifies that an itinerary can be created when the client makes the requests. The Insomnia API tests mimic typical user behaviours, this helps in verifying the crud operations for specific inputs. Note that the backend API will not run successfully unless all the unit tests are passed, and the Insomnia API will return a 200 OK if the request was successful (see Figure 5.2).



```java
@Test
void canAddNewClient() {
    // given
    String email = "friend.org@gmail.com";
    RegisterDTO registerDTO = new RegisterDTO(
            name: "Friend",
            email,
            password: "password1",
            LocalDate.of( year: 2000, Month.MARCH,  dayOfMonth: 16)
    );
    // when
    RegisterResponse registerResponse = service.registerClient(registerDTO);
    // then
    assertThat(registerResponse.getStatus()).isTrue();
    assertThat(registerResponse.getMessage()).isEqualTo("Registration Successful");
    ArgumentCaptor<Client> clientArgumentCaptor = ArgumentCaptor.forClass(Client.class);
    verify(repository).save(clientArgumentCaptor.capture());
    Client capturedClient = clientArgumentCaptor.getValue();
    assertThat(capturedClient.getName()).isEqualTo(registerDTO.getName());
    assertThat(capturedClient.getEmail()).isEqualTo(registerDTO.getEmail());
}
```



```json
{
  "name": "Winter Holiday",
  "description": "Exploring America",
  "startDate": "2023-07-01",
  "endDate": "2023-07-15",
  "location": "Miami, USA",
  "notes": "Don't forget camera!",
  "accommodation": "Hotel",
  "accommodationCost": 200.0,
  "estimatedCost": 380.0,
  "client": {
    "id": 1
  },
  "activities": [
    {
      "name": "Miami Heat game",
      "description": "Watching the game to see Jimmy Butler!",
      "location": "Heat Stadium, Miami, USA",
      "cost": 180.0
    }
  ]
}
```

Figure 5.1 & 5.2 canAddNewClient unit test & creating itinerary API test.

5.2 Solution Validation and User Feedback

This section dives into the user feedback from technical users and consumer (non-technical) users.

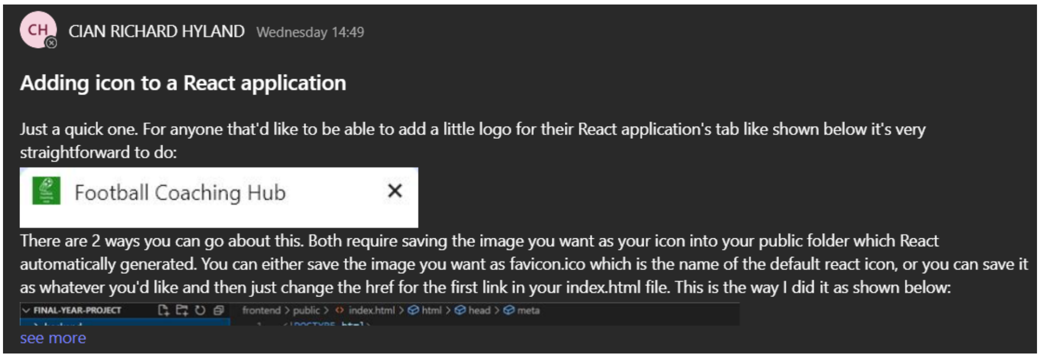
5.2.1 Consumer User Feedback

Feedback from the consumer user group, collected through a survey (sent to friends and family), helped to highlight the user satisfaction of TravelPal. The feedback offered not only validation of the user experience (discussed in section 5.2.3), but also constructive criticism that was used to further refine the UI/UX design. This constructive criticism can be seen in Figure 4.2 where a user says that the Login page and Dashboard can be made more mobile responsive, as the display was distorted. Quick refinements were made to these pages to make them more responsive, as can be seen in Figure 4.5.

More constructive criticism can be seen in Figure 4.3 where a user says that they are logged out upon clicking through the Navbar. This issue was patched by the inclusion of the AuthContext component (discussed in more detail in section 4.3.2) which helps keep the user logged in while they are authenticated, until they click the Log out button, making them unauthenticated.
See appendix documents list for link to the survey.

5.2.1 Technical User Feedback:

Feedback from the technical user group, which involved final year project teammates (under the same supervisor, Joe Duffin), provided constructive feedback from a different lens. The technical feedback was given in the form of technical suggestions for the application, found in the Joe Duffin FYP team Teams channel. One of the suggestions was to add a logo and name to the React application tabs instead of using the default. The application of this suggestion can be seen in Figure 4.1. Also refer to Figure 5.3.



CH CIAN RICHARD HYLAND Wednesday 14:49

Adding icon to a React application

Just a quick one. For anyone that'd like to be able to add a little logo for their React application's tab like shown below it's very straightforward to do:

Football Coaching Hub ✕

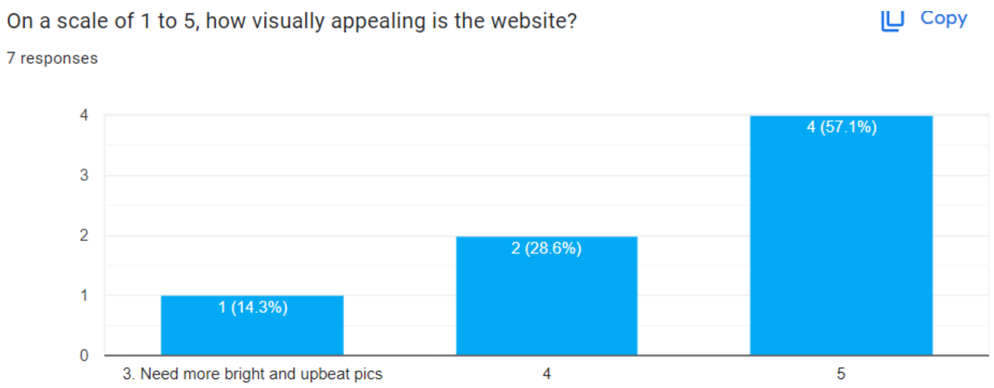
There are 2 ways you can go about this. Both require saving the image you want as your icon into your public folder which React automatically generated. You can either save the image you want as favicon.ico which is the name of the default react icon, or you can save it as whatever you'd like and then just change the href for the first link in your index.html file. This is the way I did it as shown below:

Figure 5.3 Icon suggestion from Joe Duffin FYP team Teams channel

Another suggestion was to integrate Route-Protection to the application in the form of a ProtectedRoute component in React. I discuss this in more detail in section 4.3.2. Instructions for this was given in the form of a YouTube link.

5.2.3 User Satisfaction and Positive Feedback

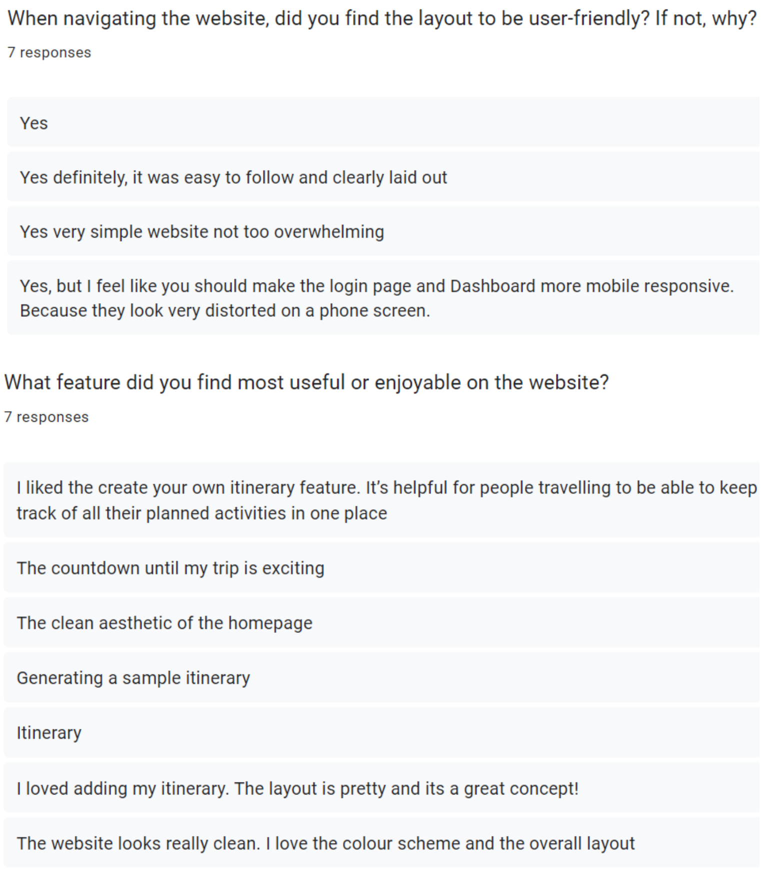
As we've discussed the constructive criticism and how it was acted upon, it is only appropriate that we also discuss the positive feedback from users towards TravelPal. Figures 5.4, 5.5 & 5.6 showcase the favourable survey responses from users, praising the websites visual appeal, user-friendliness, and enjoyable features. See Appendix 2F for user survey.



On a scale of 1 to 5, how visually appealing is the website?

7 responses

Figure 5.4 Visual appeal feedback



When navigating the website, did you find the layout to be user-friendly? If not, why?

7 responses

Yes

Yes definitely, it was easy to follow and clearly laid out

Yes very simple website not too overwhelming

Yes, but I feel like you should make the login page and Dashboard more mobile responsive. Because they look very distorted on a phone screen.

What feature did you find most useful or enjoyable on the website?

7 responses

I liked the create your own itinerary feature. It's helpful for people travelling to be able to keep track of all their planned activities in one place

The countdown until my trip is exciting

The clean aesthetic of the homepage

Generating a sample itinerary

Itinerary

I loved adding my itinerary. The layout is pretty and its a great concept!

The website looks really clean. I love the colour scheme and the overall layout

Figure 5.5 & 5.6 Enjoyable features and User-Friendliness feedback

Chapter six: Conclusion

Summary

This chapter concludes the exploration and development journey of TravelPal. It covers the project's contributions to simplifying travel planning, discusses the project's approach and results, and outlines future directions for further enhancement.

6.1 Contribution to the state-of-the-art

TravelPal stands out in the crowd of travel applications by taking the uncommon approach of prioritizing focus and simplicity. This straight-forward approach makes travel planning stress free and enjoyable. Its use of a responsive React.js frontend and a reliable Spring Boot backend contributes an alternative to the more feature heavy market.

6.2 Results discussion

The project's outcomes highlight the appeal of TravelPal for users. TravelPal's development approach, emphasizing user feedback and iterative design, has proven effective in creating a solution that resonates with the needs of travellers seeking simplicity in planning.

6.3 Project Approach

TravelPal's approach was user-focused and iterative. Starting with a broader ambition, the direction changed towards simplicity, a decision guided by user feedback and technical practicality. This process not only shaped TravelPal but also highlighted the importance of flexibility and user engagement in project development.

6.4 Future Work

Future enhancements for TravelPal include expanding the itinerary features by integrating more travel data in the field of itinerary management, like flight information and integrating personalized travel recommendations based on user and destination information. TravelPal will continue to make improvements based on user feedback, ranging from minor tweaks to major feature additions, while also reinforcing backend security as the user-base expands.

An important aspect of future development involves several models outlined in our initial documentation, specifically "Destination", and "Transport", which were not incorporated into the initial release. Future versions of TravelPal will likely see these models integrated, strengthening the user experience.

User Manual (Short)

A short segment of the user manual will be displayed in this section. See Appendix 1 or the user manual file in Supporting documents for the full User manual.

Step 1 (Landing page):

When in the landing page (after clicking the link), you can click any of the Navbar buttons at the top to go to their respective pages or scroll down to view the landing page sliders with more information.

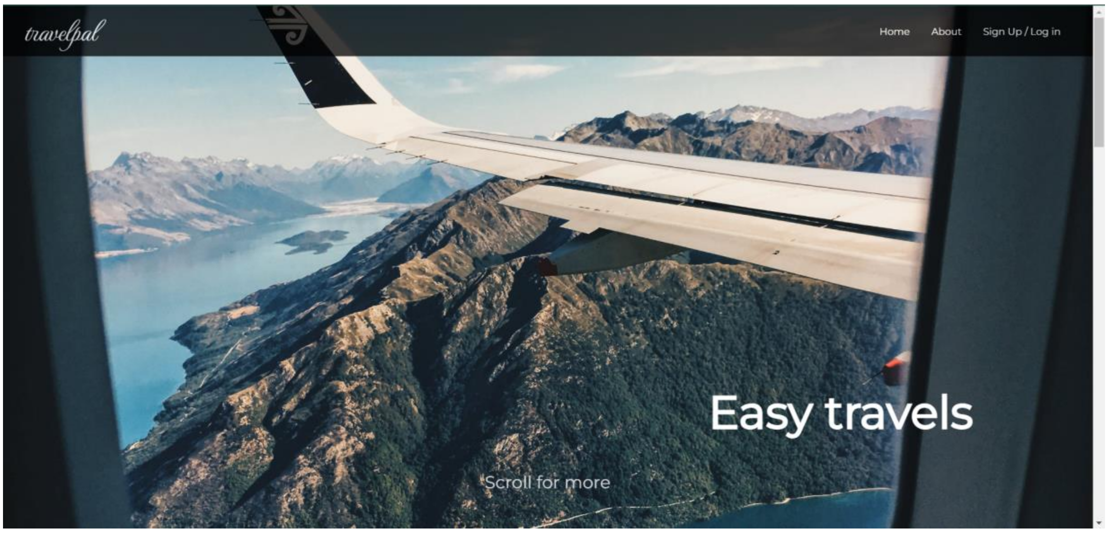


Figure 7.1 Landing Page

Step 2 (Landing page scroll):

As you scroll down, you will see images that fade in and out of the screen, prompting you to get started (see Figure 7.2). If you click the button, you will be re-directed to the Login page. You can observe that the webpage will not be refreshed as you are being re-directed throughout the application (a benefit of using React).

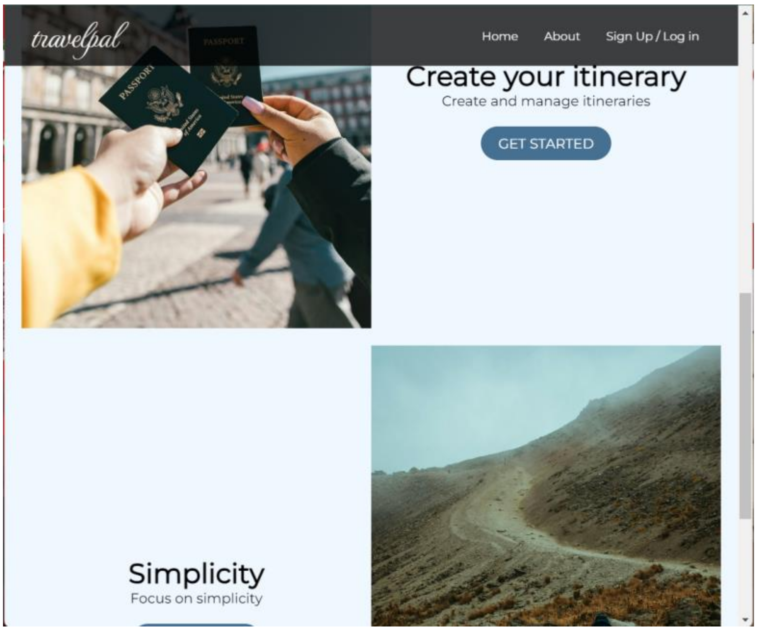


Figure 7.2 Landing Page Sliders

Step 3 (Log In/Sign Up):

In the Log In/Sign Up page, fill in your details and click the Log In or Sign Up button to Sign Up/Log In (see Figure 7.3). You will then be re-directed to your personal Dashboard.

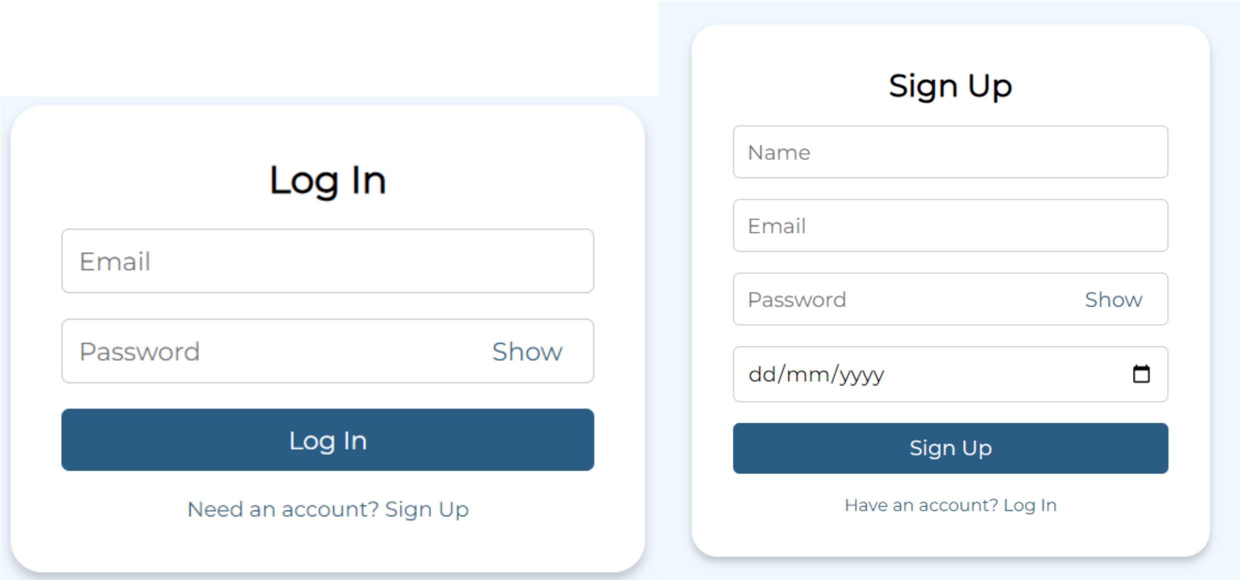


Figure 7.3 Log In/Sign Up Page

See Appendix 1 or the user manual file in Supporting documents for the full User manual.

References

Amigoscode. (2021a). *Software Testing Tutorial - Learn Unit Testing and Integration Testing*. https://www.youtube.com/watch?v=Geq60OVyBPg

Amigoscode. (2021b). *Spring Boot Tutorial | Full Course [2023] [NEW]*. https://www.youtube.com/watch?v=9SGDpanrc8U

Arjun, C. W. (2022). *Full Stack Spring Boot and React CRUD 1.5 hours Course | Full Stack Web App | MySQL | Hibernate*. https://www.youtube.com/watch?v=4LZKnegAm4g&list=PLwNA2fNOG0AAVwLhLw7gdLpDfKBe3Ve4t&index=18&t=2000s

Benson-Obilom, B. (2022a). *benb0201/myportfolio2: The definitive version of my portfolio. Still a work-in-progress.* https://github.com/benb0201/myportfolio2

Benson-Obilom, B. (2022b). *myportfolio*. https://benb0201.github.io/myportfolio2/

Coding, D. (2021). *Entity Mapping | @OneToOne, @OneToMany, @JoinColumn| Spring Data JPA | Spring Boot | Dev2Prod Coding*. https://www.youtube.com/watch?v=2fvUrpYG95w&list=PLwNA2fNOG0AAVwLhLw7gdLpDfKBe3Ve4t&index=18&t=582s

Cognitive Programmer. (2020, March 28). *How to auto generate Sequence Diagrams using plantuml - Show how your software works visually - YouTube*. https://www.youtube.com/watch?v=NIhxW-bIkDA

CS Hero. (2023, January 11). *PlantUML: Intro to Class Diagrams - YouTube*. https://www.youtube.com/watch?v=9CLE33qPIvI

Design, B. (2020). *React Website Tutorial - Beginner React JS Project Fully Responsive*. https://www.youtube.com/watch?v=I2UBjN5ER4s&list=PLwNA2fNOG0AAVwLhLw7gdLpDfKBe3Ve4t&index=14

etc, T. (2022). *Create Login and Registration Form In React JS (Beginner)*. https://www.youtube.com/watch?v=Y-XW9m8qOis&list=PLwNA2fNOG0AAVwLhLw7gdLpDfKBe3Ve4t&index=15

Fayock, C. (2022). *Trigger a Function when Scrolling to an Element in React with Intersection Observer*. https://www.youtube.com/watch?v=r1auJEf9ISo

Flux Academy. (2022, August 31). *Perfect Landing Page Design Explained (in 5 minutes) - YouTube*. https://www.youtube.com/watch?v=GTNgiTK-ic8

freeCodeCamp.org. (2021). *JavaScript Programming - Full Course*. https://www.youtube.com/watch?v=jS4aFq5-91M

Gray, D. (2021). *React Axios API Requests | Axios with React JS Tutorial*. https://www.youtube.com/watch?v=ZEKBDXGnD4s&list=PLwNA2fNOG0AAVwLhLw7gdLpDfKBe3Ve4t&index=16&t=1654s

GreatStack. (2023). *How To Make Sign In & Sign Up Form Using React JS | ReactJS Login & Registration Form*. https://www.youtube.com/watch?v=8QgQKRcAUvM&list=PLwNA2fNOG0AAVwLhLw7gdLpDfKBe3Ve4t&index=16

HurrayBanana. (2016). *Using Doxygen to document computing projects*. https://www.youtube.com/watch?v=-V_vHZPOZfY&list=PLwNA2fNOG0AAVwLhLw7gdLpDfKBe3Ve4t&index=2&t=319s

Ninja, N. (2023). *React Router in Depth - YouTube*. https://www.youtube.com/playlist?list=PL4cUxeGkcC9iVKmtNuCeIswnQ97in2GGf

Overflow, S. (2014). *Cannot create a database table named “user” in PostgreSQL*. https://stackoverflow.com/questions/22256124/cannot-create-a-database-table-named-user-in-postgresql

Overflow, S. (2016). *Using env variable in Spring Boot’s application.properties*. https://stackoverflow.com/questions/35531661/using-env-variable-in-spring-boots-application-properties

Random code. (2021a, February 24). *UML Basics (with PlantUML) #9: Activity Diagram - if, else and while - YouTube*. https://www.youtube.com/watch?v=hpVA0Qk2d1M

Random code. (2021b, March 3). *UML Basics (with PlantUML) #12: Component Diagram - Elements - YouTube*. https://www.youtube.com/watch?v=IlVRTPEHjHU

Rino de Boer. (2023, January 12). *Introducing Figma: A Beginners Tutorial (2023 UI UX Design) - YouTube*. https://www.youtube.com/watch?v=JGLfyTDgfDc

Scrimba, & Borgen, P. H. (2021). *Congrats on completing the Learn JavaScript course!* https://scrimba.com/learn/learnjavascript/congrats-on-completing-the-learn-javascript-course-c4ZL6Juv

Simplified, W. D. (2020). *Learn CSS Animation In 15 Minutes*. https://www.youtube.com/watch?v=YszONjKpgg4

Sololearn. (2014). *Web Development | Learn with Sololearn*. https://www.sololearn.com/en/learn/courses/web-development

Solutions, C. (2023, March). *Learn React Hooks: useContext - Simply Explained!* https://www.youtube.com/watch?v=HYKDUF8X3qI&list=PLwNA2fNOG0AAVwLhLw7gdLpDfKBe3Ve4t&index=6

Solutions, C. (2024, March). *Custom protected route component in react - YouTube*. https://www.youtube.com/@cosdensolutions. https://www.youtube.com/watch?v=eFPvXGZETiY&list=PLwNA2fNOG0AAVwLhLw7gdLpDfKBe3Ve4t&index=7&t=429s

Talk, R. D. (2021). *Javascript Promises vs Async Await EXPLAINED (in 5 minutes)*. https://www.youtube.com/watch?v=li7FzDHYZpc&list=PLwNA2fNOG0AAVwLhLw7gdLpDfKBe3Ve4t&index=19

Udemy, & Schwarzmüller, M. (2023). *React 16: The Complete Course (incl. React Router 4 & Redux)*. https://www.udemy.com/course/react-the-complete-guide-incl-redux/

Vaughan, A. (2013). *PlantText UML Editor*. https://www.planttext.com/

Appendices

Appendix 1 User Manual (Complete)

Step 1 (Landing page):

When in the landing page (after clicking the link), you can click any of the Navbar buttons at the top to go to their respective pages or scroll down to view the landing page sliders with more information. See Figure 7.1.

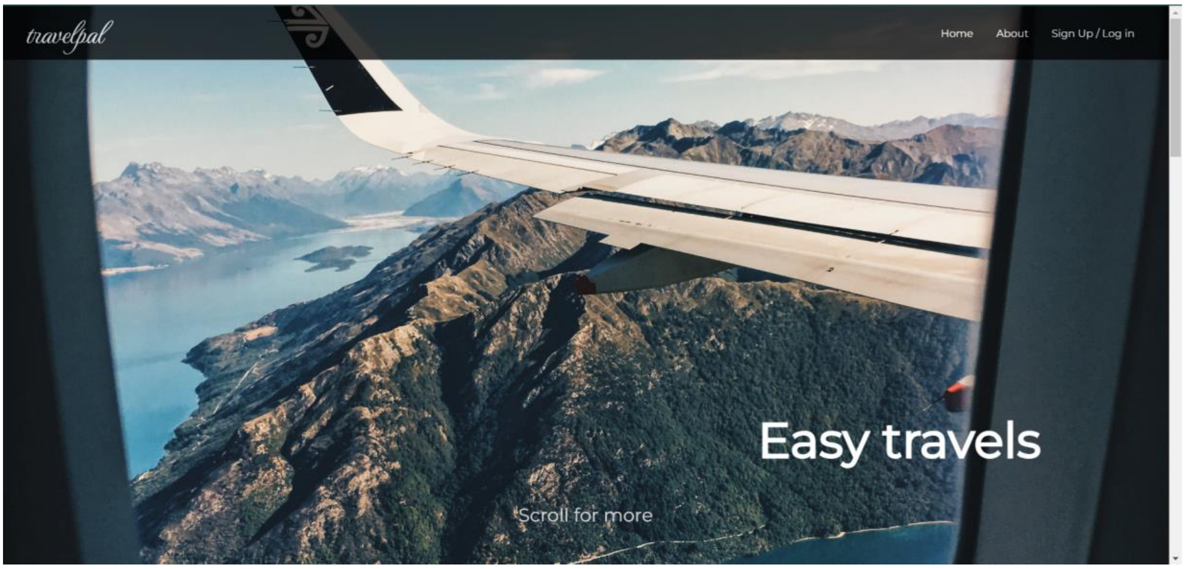


Figure 7.1 Landing Page

Step 2 (Landing page scroll):

As you scroll down, you will see images that fade in and out of the screen, prompting you to get started (see Figure 7.2). If you click the button, you will be re-directed to the Login page. You can observe that the webpage will not be refreshed as you are being re-directed throughout the application (a benefit of using React).

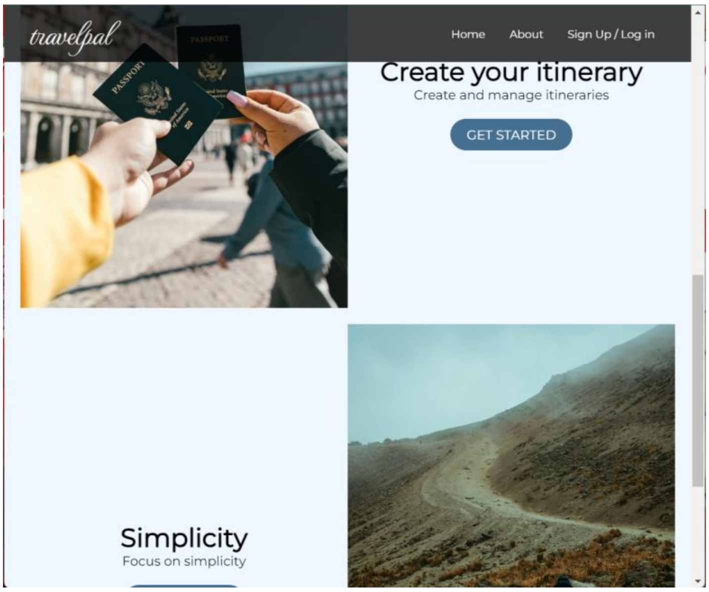


Figure 7.2 Landing Page Sliders

Step 3 (Log In/Sign Up):

In the Log In/Sign Up page, fill in your details and click the Log In or Sign Up button to Sign Up/Log In (see Figure 7.3). You will then be re-directed to your personal Dashboard.

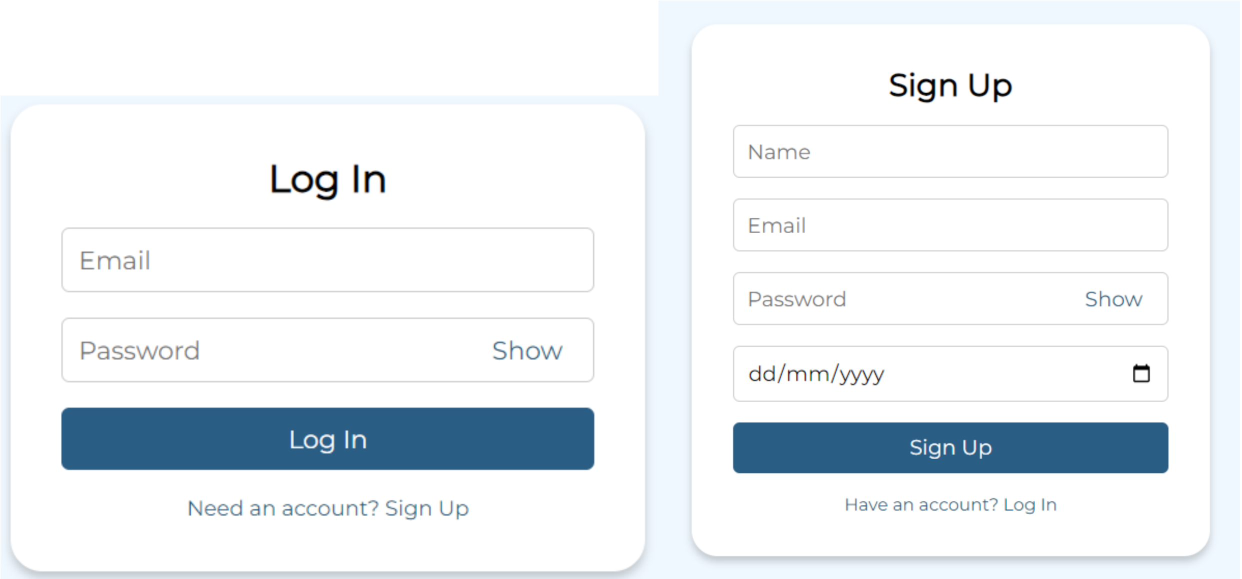
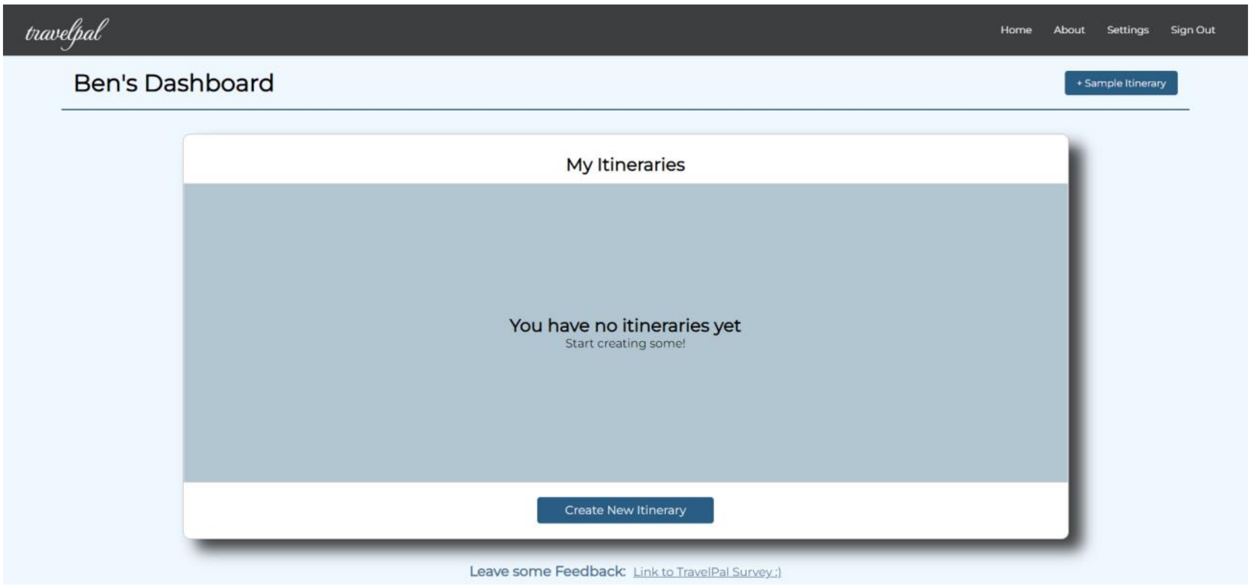


Figure 7.3 Log In/Sign Up Page

Step 4 (Dashboard):

When logged in, you will see a Dashboard with your name on it. You will start with no itineraries but will have two options to get one. (Note: Click the Sign Out button on the Navbar at any point to sign out)

Option 1: Generate a sample itinerary, by clicking the blue button **outside** the "My itineraries" box. It will say "+ Sample Itinerary".
Option 2: Create a new itinerary, by clicking the blue button **inside** the "My itineraries" box. It will say "Create New Itinerary" or "Create" if in mobile display.



You have no itineraries yet

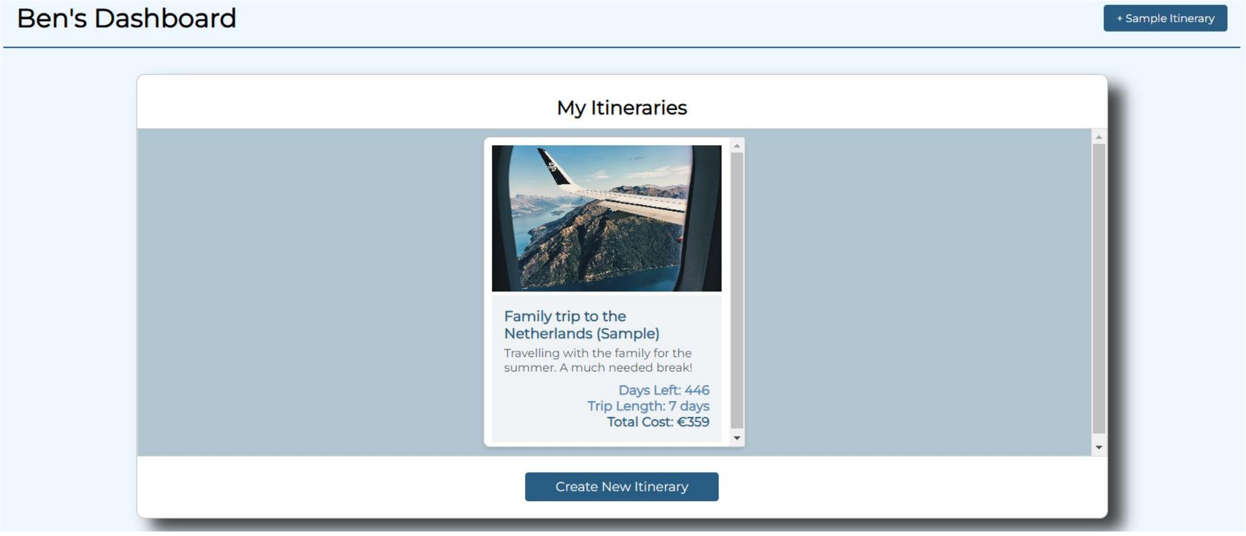
Start creating some!

Leave some Feedback: Link to TravelPal Survey :)

Figure 7.4 Dashboard (no itineraries)

Step 5 (Option 1: Sample Itinerary):

If you click the sample itinerary button, a sample itinerary will be generated for you. You can use it for some inspiration. Clicking the itinerary preview, will display all the itinerary details (see Figure 7.8).



Family trip to the Netherlands (Sample)

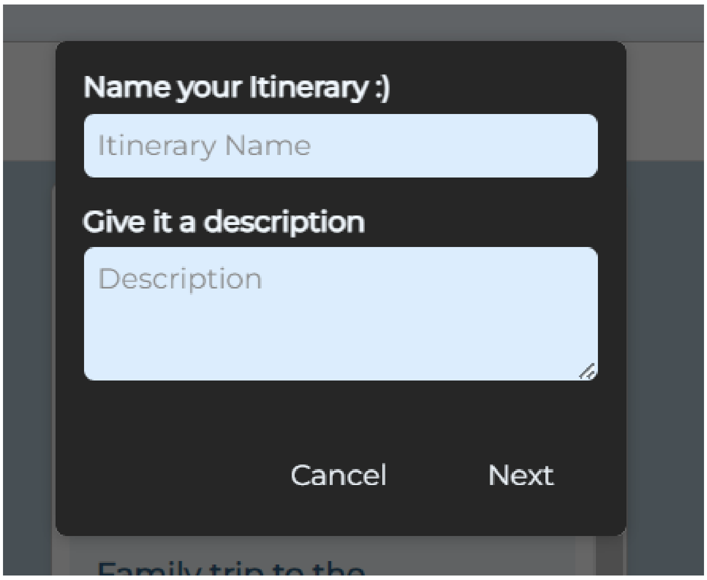
Travelling with the family for the summer. A much needed break!

Days Left: 446
Trip Length: 7 days
Total Cost: €359

Figure 7.5 Dashboard (sample itinerary)

Step 6 (Option 2: Create New Itinerary):

Now that you have some inspiration, if you click the create button, a display will appear over the Dashboard for you to fill out your itinerary information.



Name your Itinerary :)

Itinerary Name

Give it a description

Description

Figure 7.6 Dashboard (create itinerary)

Step 7 (Add Activities):
During the creation process, if you want to add an activity you will have to click the “Add Another Activity” button, then “expand activity”. From there, you can fill in the information for your Activity. Once all the itinerary is information is filled out and submitted, your new itinerary will be generated and stored in your “My Itineraries” list.

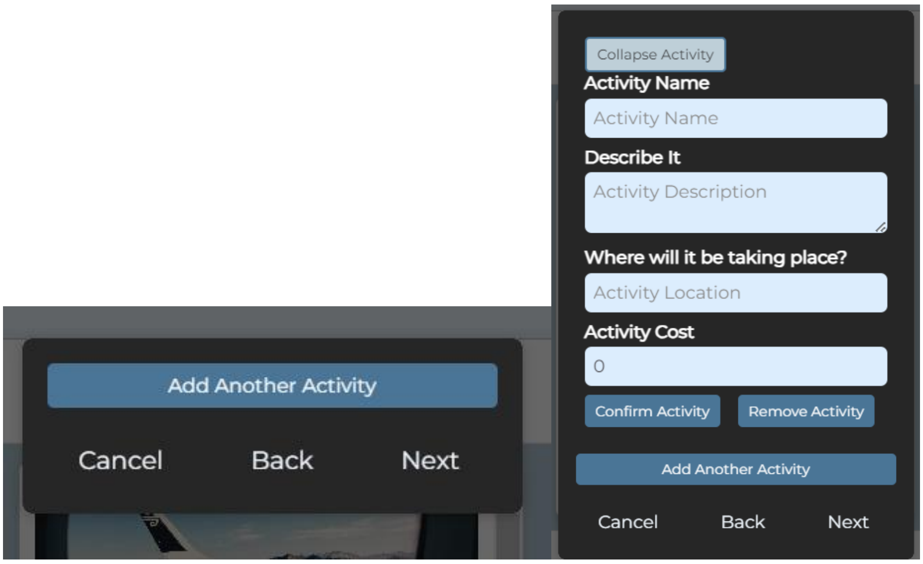
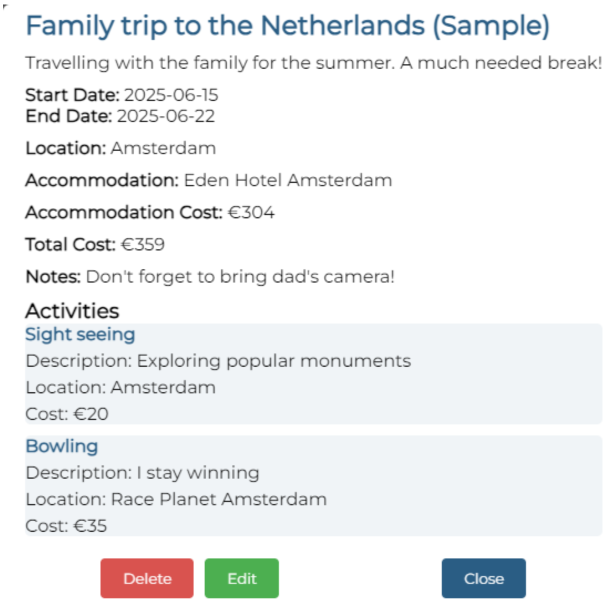


Figure 7.7 Dashboard (add activities)

Step 8 (Itinerary Details):
Upon clicking an Itinerary, you will be greeted with a display of all its details. Here you can close the itinerary or pick one of two options at the bottom of the details display.

Option 1: Click the red delete button to delete your itinerary.
Option 2: Click the green edit button to go through your itinerary and make changes of your choice.



Family trip to the Netherlands (Sample)

Travelling with the family for the summer. A much needed break!

Start Date: 2025-06-15
End Date: 2025-06-22

Location: Amsterdam

Accommodation: Eden Hotel Amsterdam

Accommodation Cost: €304

Total Cost: €359

Notes: Don't forget to bring dad's camera!

Activities

Sight seeing
Description: Exploring popular monuments
Location: Amsterdam
Cost: €20

Bowling
Description: I stay winning
Location: Race Planet Amsterdam
Cost: €35

Figure 7.8 Dashboard (add activities)

Appendix 2 Code and Project Resources

This appendix details the code developed for the TravelPal project and provides links to additional resources and materials created during the development process.

A. Code Repository

The complete source code for TravelPal, including both the frontend and backend components, is available on GitLab. Note: GitLab repo is private to MU CS students and staff. GitHub repo is public.

- GitLab Repository (directly connected to local branch): [https://gitlab.cs.nuim.ie/u210266/fyp]

- GitHub Repository (same source code): [https://github.com/benb0201/fyp/tree/master]

B. Final Year Project Journal

A journal detailing the development journey of TravelPal, including design decisions, challenges encountered, and solutions implemented. This journal offers insights into the project's evolution from conception to completion.

- Project Journal: [https://github.com/benb0201/fyp/wiki/FYP-Journal]

C. Code Documentation

Code documentation for the backend codebase, providing a detailed overview of the classes, methods, and structures used.

- Code Documentation: [https://benb0201.github.io/travelpal-code-documentation/]

Supporting Documents (Following resources can be found in Supporting documents file)

D. User Manual

A step-by-step guide on using TravelPal, including screenshots and descriptions of how to navigate the application and create itineraries.

- Can be found in **Appendix 1 or supporting documents file** on Moodle. [File name: User Manual]

E. Deployment and Hosting Information

- **TravalPal's official Webpage!**
  - TravelPal Website: [https://fyp.cs.nuim.ie/projects/FYP24JD006/]

- Meeting details about the process of dockerizing TravelPal and hosting it on the Maynooth University CS Department servers.
  - Can be found in **supporting documents file** on Moodle. [File name: CS Dept Hosting meetings with Misha 2023 2024]

F. Survey Feedback

Results of user feedback collected through surveys, which informed various improvements and feature additions to TravelPal.

- Can be found at: (https://docs.google.com/forms/d/1XcPEkv4oTa1Vir8wTbN5UiPXql4gvzRDea8Yt_P8_Sc/viewanalytics)

- Also found in **supporting documents file** on Moodle. [File name: TravelPal Feedback]

G. Accessibiliy and Usabilty report

Short Report done for the Accessibility and Usability of TravelPal.

- Can be found in **supporting documents file** on Moodle. [File name: Short Accessibility Report by Ben_Obilom]

H. User Stories

TravelPal user stories new and old.

- Can be found **in supporting documents file** on Moodle. [File name: TravelPal User Stories]

I. Wireframe Mockups

TravelPal wireframes (One seen in Figure 4.4)

- Can be found **in supporting documents file** on Moodle. [File name: TravelPal Wireframe mockups]

Appendix 3 UML Diagrams: Activity, ER and Class for this project.

Activity Diagram

Here, the overall process flow from the user's perspective is on show. This diagram details the decision-making process and the sequence of activities within the TravelPal application.

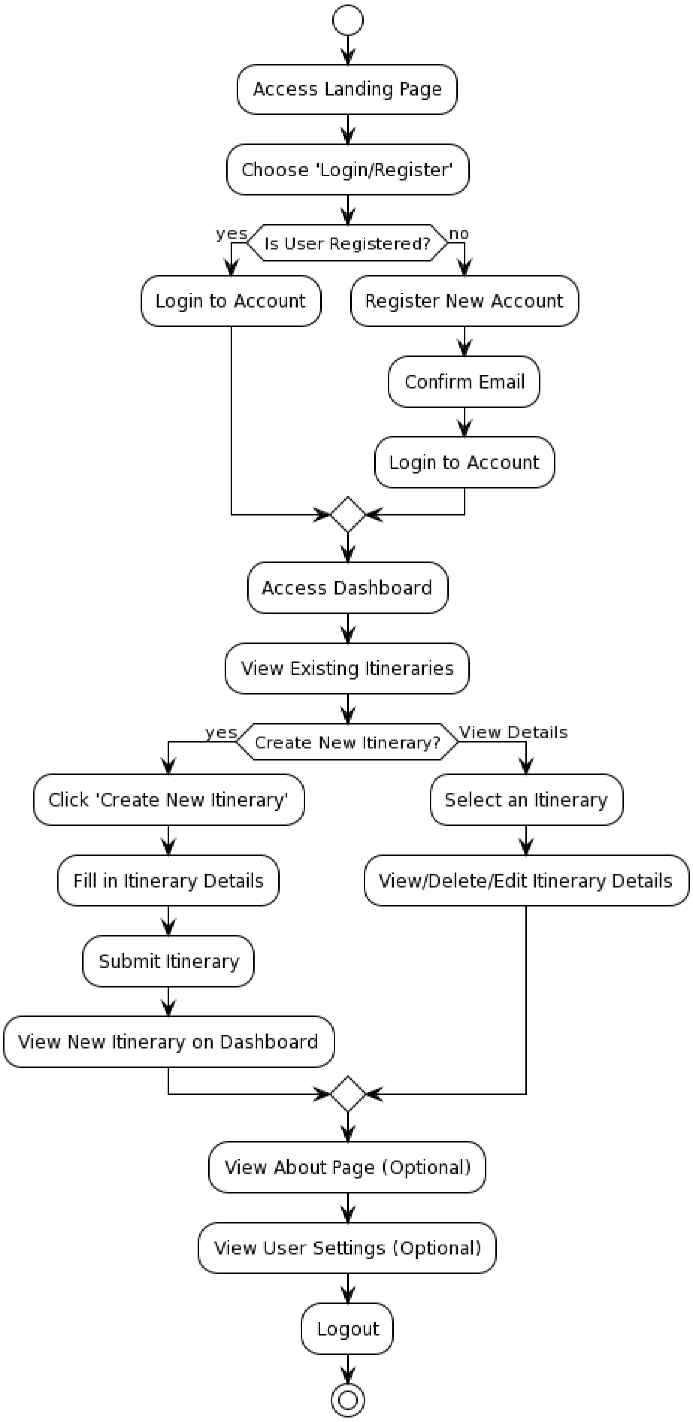


Figure A1 UML Activity diagram for TravelPal.

Entity-Relationship Diagram: This diagram illustrates the database schema and the relationships.

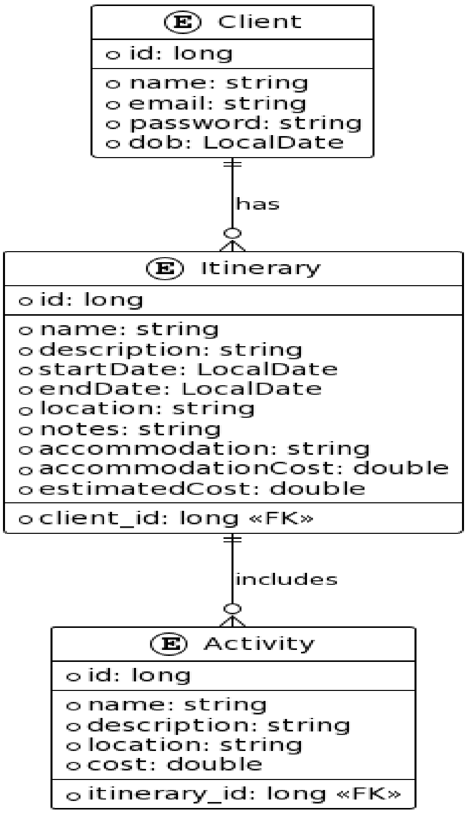


Figure A2 UML Entity-relationship diagram for TravelPal.

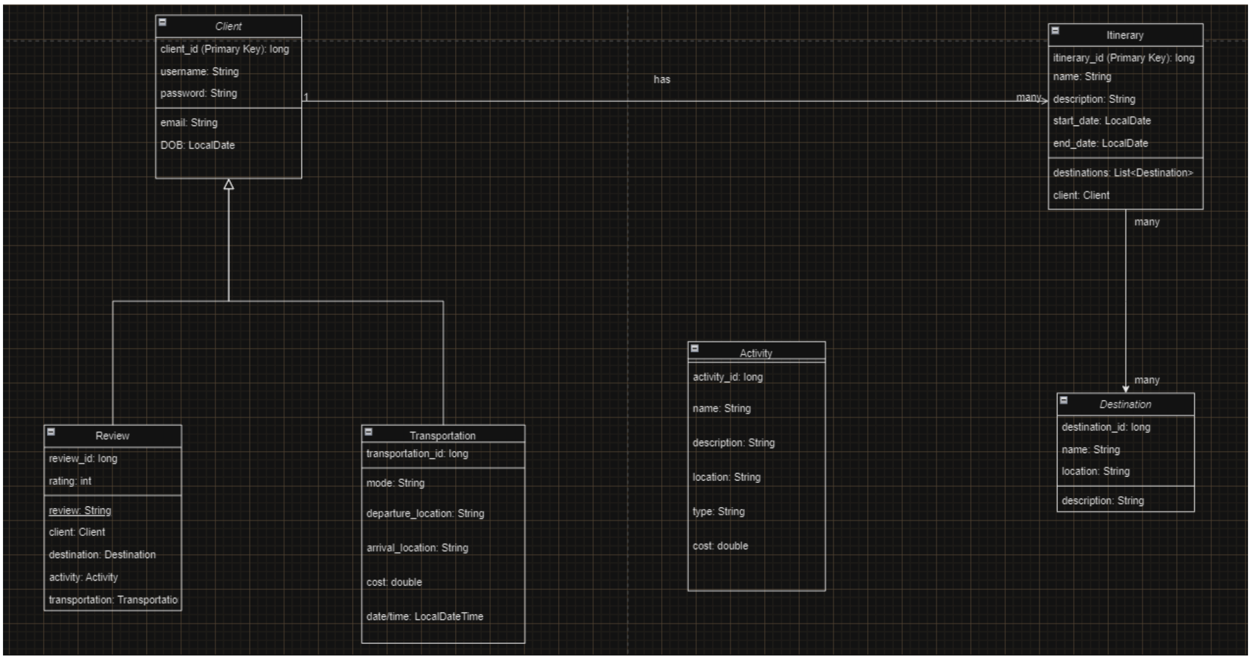


Figure A3 Original class diagram from the feature rich TravelPal plan.