Bilkent University
Department of Computer Engineering

Senior Design Project

T2307

Travela

Detailed Design Report

Group members

21903224, Yağmur Eryılmaz, yagmur.eryilmaz@ug.bilkent.edu.tr
21901972, Cenk Duran, cenk.duran@ug.bilkent.edu.tr
21902620, Efe Ertürk, efe.erturk@ug.bilkent.edu.tr
21902820, Çağla Ataoğlu, cagla.ataoglu@ug.bilkent.edu.tr
21903248, Efe Şaman, efe.saman@ug.bilkent.edu.tr

Supervisor

Cevdet Aykanat

13.3.2023

This report is submitted to the Department of Computer Engineering of Bilkent University in partial fulfilment of the requirements of the Senior Design Project course CS491/2.

Contents

Detailed Design Report

Travela

1 Introduction

Vacations are fun, but planning them, not so much. Especially if you are responsible for planning a holiday trip that suits many people with different interests, budget and available times. This is a serious task for everyone, and on average, a person spends 10 hours planning their holiday, as the survey of Independent Newspaper claims [1]. Moreover, the results of the poll that includes more than 7800 people's opinions from 26 different countries show that over 25 percent of people think holiday planning is one of life's 'biggest stressors'. Thus, our application, "Travela" focuses on solving this problem, while being a total travel companion.

Travela is an app that provides a unique way of planning your holiday trips with your friends. With the machine learning systems we use, the app suggests the best location or route of vacation for your trip group, taking all of the group members' holiday interests, budget and time constraints into consideration. While doing that, the app also suggests to you the popular landmarks or recommended activities in your route, to make your trip well-worth. Also, you can share these holiday plans and your holiday memories on the app, and highlight the places visited in your own unique Globe. By this way, you can see your previous holidays with your most fun memories, and also your friends' too. Thus, it is also a social media app for travel lovers!

In this report, analysis of the system will be explained thoroughly. Firsty, we will mention current systems, and our products' differences from the rest. Then we will explain our system, Travela, in detail, by mentioning functional and nonfunctional requirements of the project, as well as the pseudo requirements and the constraints. Then, in part 3.5 of the report, we will demonstrate our System Models, by first listing the possible Scenarios within the app, then showing our Use-Case Model, followed by the Object and

Class Model, Dynamic Models and lastly, the User Interface. Then, in part 4 of the report we will mention Other Analysis Elements such as Consideration of Various Factors in Engineering Design, Risks and Alternatives, Project Plan, Ensuring Proper Teamwork, Ethics and Professional Responsibilities and Planning for New Knowledge and Learning Strategies.

2 Current System

Current Systems like Airbnb or TripAdvisor are vacation recommender systems for individual users. You can set flexible dates or budget for your vacation on Airbnb, create “Curated Wishlists” that are basically your vacation wishlists [2]. On the other hand, Tripadvisor recommends you to visit popular landmarks near your location or your desired vacation location, while giving you new trip ideas. As you can see, if you are planning a trip for a group of people with different interests and available time or budgets, these systems do not help at all. Our proposed system is designed to solve this issue, while covering most of the functionality of the already existing systems like recommending landmarks in a particular place, like TripAdvisor does, or enable you to create trip wishlists like Airbnb does. Moreover, current systems and apps do not have a social media aspect, most of them just function as a travel planner system. But Travela has a social media aspect as well, you can share your memories all around the world, color your globe to show the places that you have visited, or share your trip ideas, plans or routes to your followers.

3 Proposed System

3.1 Overview

Travela is planned to be both a travel companion, and a new social media platform. As for the travel companion aspect, the most important features can be named as enabling users to form travel groups via sending an invitation on the app to arrange a trip together with friends, suggesting users the most suitable vacation by taking each group members holiday interests, budget and time constraints in mind, and suggesting suitable hotels, transportation opportunities, popular landmarks and activities in the vicinity of the travel. For the social media aspect, the most important features can be described as allowing users to see previous trips of both themselves and their friends on unique personalized Globes, create routes and plans according to their map and allowing users to post and attach photos of their trips to their world map which can be seen by the others. The features that are supported by Travela are illustrated in depth in part 3.2 “Functional Requirements" of this report.

To give the customers the best experience, it is vital to gather as much information about a user as possible, to suggest the most suitable vacations. For this reason, it is a good idea to ask the users to solve some sort of a questionnaire, involving questions like “Which of these vacations below do you see yourself in?”, and letting the user choose its own preference. Moreover, integrations with the third party applications that provide the data of nearby hotels, suitable flights or transportation opportunities, like Google hotels, Skyscanner, Amadeus, Travelport, Tripadvisor or Google map reviews will surely increase our service quality as a total travel companion.

Travela is planned to bring innovation of service and customer engagement, as the provided service and functionality of recommending vacations to a group of users by taking account their interests, is different from the already existing products on the market. Also, this app makes planning trips easier and more enjoyable, thus it can be also considered as an innovation of service that increases user experience. The innovation that is planned is also incremental, as it is not achieved by leveraging a radical technology, but it is something that fulfills a need in the market for many users. The machine learning recommender system that we plan on using is similar to the preexisting ones in terms of creating a feature matrix of a person's interests, but whenever a group is formed, these matrices will be taken into consideration together, which is a new technique, but not a radical one. Travela’s business strategy is Digital Business Optimization, which focuses on enhancing customer experiences. Also, it provides a new way of engaging the customers in the tourism / travel planning domain, thus, it can be said that it creates a new digital business model.

As Travela is an app whose success depends on the experience of users within the app, it is essential that the recommender systems are fast and accurate. It is important to mention that the target user group is anyone that plans trips online, especially young people that try to plan holidays through an app with their friend group. Baring that in mind, having an easy and usable user interface for the target customer base is a requirement, which adds to the user experience. One should easily search for trips, form travel groups, view possible vacations for the group and view the activities around the travel route, almost intuitively and without any issue. Our goal as the Travela team is that this app is used internationally, has lots of users in holiday seasons (as it is not realistic to expect people to browse through a vacation app every day), and recognized in the market as a new social media app.

It is important to note that Travela is an intermediary application that helps people to find suitable vacations for themselves, and also, helps hotels, transportation companies, and local landmarks to find their customers. Once a group decides on their travel route, involving where to stay and which transportation unit they will use, they will be directed to actual websites of these services. For example, they will be directed to the web site of the hotel they have chosen for reservation, and the web site of the airline company for buying the seats. There is no reservation or payment functionality on Travela. The advertisements on the app will help Travela to continue giving high quality service to its customers.

3.2 Functional Requirements

3.2.1 Sign up - Login

- Allows users to sign up and login to the system by using their Google/ Facebook account or via email and password that is determined in the sign up process.

- Enables users to renew their password when it is forgotten using their email.
- Collects data from the user during the sign up process to see his/her preferences like favorite location, preferred routes or activities via a quick questionnaire.

3.2.2 Holiday Arrangement

- Enables users to send a friendship request to other users.
- Enables users to form travel groups via sending an invitation on the app to friends in order to arrange a trip together.
- Enables users to integrate their calendar hence suitable dates to the system.
- Shows the common available dates of the group members via calendar usage.
- Allows users to search destinations that they want to visit.
- Shows suitable hotels and transportation opportunities for the given dates and place.
- Allows users to share destinations and related information (hotel, transportation) with their travel group via chat feature.
- Suggests travel routes according to groups' overall preferences (budget, date) by using ML algorithms.
- Suggests activities (concerts, festivals, etc.) between the given dates in selected locations.
- Allows users to see their upcoming trips that are finalized by the travel group.
- Shows reviews and ratings of the destinations and activities.
- Shows information on which season a place is generally visited like Ibiza is often visited in summer season and Switzerland is in winter season.
- Allows users to see their current travel groups in their account.
- Enables users to create polls among their group to determine possible routes, plans, itineraries, activities, etc.

3.2.3 Social Media Aspect

- Allows users to see previous trips of both themselves and their friends in a World map, create routes and plans according to their map.
- Allows users to post photos belonging to a specific trip to their world map which can be seen from the others.
- Enables users to vote the destinations that they see in the platform as dislike,like and love.
- Enables users to rate their previous trips out of 5 to be able to collect data for suggestion algorithms and also add review for the trip.
- Enables users to delete their account if preferred.
- Allows users to take a questionnaire to his/her traveler personality.

3.3 Non-functional Requirements

3.3.1 Usability

- User interface should be understandable and not complex, also catchy to impress users.
- Headings should be in larger font size than the other parts to draw attention.
- Font size for the body should be minimum 16 px.
- Interfaces should be responsive for both mobile and web.
- Design should be made with a mobile first approach as mobile app is thought to be used more commonly.
- To apply mobile first design, for iOS the Human Interface Design should be considered and for Android Material Design should be considered [3] [4].
- While fetching data, refreshing animations should be attractive not to distract users.
- It should address all kinds of users from various ages.
- The language should be English to be universal.
- The user shouldn’t have to re-login if he/she refreshes the page.

3.3.2 Security

- Users will not be able to enter the system without a password as the app also has a social media aspect.
- Passwords will be saved in the database in an encrypted version.
- Loss of data in a possible crash should be prevented.
- Users' all previous trip data should be removed if the user deletes his/her account.
- An authentication token should be implemented to determine the specifications of a session [5].
-

3.3.3 Maintainability

- Updating the system should not affect the end user's experience for a long time.
- Encapsulation should be used to facilitate detection of problems.
- Consistent commenting method will be used to follow the process of code and make it more understandable.

3.3.4 Performance

- While logging in and logging out, the processes should not take more than 5-10 seconds.
- Navigation between pages should happen fast enough to keep users in the system and should happen at the same speed on all platforms.

3.3.5 Extensibility

- Travela should be available on multiple platforms (web, mobile).
- Travela should work well with external APIs. (Skyscanner,Google Hotels , etc.)

3.3.6 **Scalability**

- It is aimed to reach many people so Travela should be scalable and handle data properly.

3.3.7 **Accessibility**

- Travela's mobile app should be downloadable for free from the App Store or Google Play Store.
- Travelable should be easily visited through its URL.

3.3.8 **Supportability**

- For mobile, Travela should support iOS and Android operating systems.
- For web Travela should support Firefox, Chrome, Opera, Microsoft Edge, and Safari browsers.

3.4 Pseudo Requirements

- Application will be available on multiple platforms which are Web, Android and iOS.
- GitHub will be used for version control to track changes.
- Google Meets will be used for group meetings.
- Google Docs will be used for further reports.

3.5 System Models

3.5.1 Scenarios

1. Name: Sign up

2. Participating Actor: User

3. Entry Conditions: Web page is browsed or Travela is downloaded

4. Exit Conditions: Registration is completed or failed

5. Flow of Events:

5.1 User clicks the “Sign Up” button

5.2 User fills the registration form.

5.3 System checks the provided information if data is missing, invalid or irrelevant it gives an error, else it creates a new account.

5.4 User fills a questionnaire to mention his/her favorite location, preferred routes or activities etc.

---

1. Name: Login

2. Participating Actor: User

3. Entry Conditions: User should have a valid password and id

4. Exit Conditions: User logins to the system or the login fails

5. Flow of Events:

5.1 User clicks the “Login” button

5.2 User fills login information

5.3 System checks the provided information if data is missing, invalid or the user does not exist. The system gives an error .

---

1. Name: Form travel group

2. Participating Actor: User

3. Entry Conditions:

3.1 User should be logged in

3.2 User should already have friends

4. Exit Conditions: Invitations are accepted by users who received requests

5. Flow of Events:

5.1 User navigates to the Profile Page

5.2 User clicks “Form Group” button

5.3 User selects Friends to send an invitation and presses the “Send Invitation” button

5.4 User clicks “Close” button when he/she is done sending invitations

5.5 Invitations are accepted by users who received requests

---

1. Name: Integrate calendar (important dates) to the system

2. Participating Actor: User

3. Entry Conditions: User should be logged in

4. Exit Conditions: User integrates his/her calendar

5. Flow of Events:

User:

5.1 Switches to the profile page.

5.2 Clicks “Integrate Calendar” button.

5.3 Add suitable dates through the year to the system.

5.4 Clicks the “Close” button when integration is done.

---

1. Name: Arrange trip in a travel group

2. Participating Actor: Users in the group

3. Entry Conditions:

3.1 Users should be logged in to the system.

3.2 Users should be in the same travel group.

3.3 Users should already have added the suitable dates to their calendars.

4. Exit Conditions: Users are navigated to the related sites for purchasing tickets or reserve the determined place.

5. Flow of Events:

5.1 Users navigate to the profile page.

5.2 Users select the travel group that they want to have a trip with.

5.3 Each user views the common available dates of the group members via calendar integration and prefers one of the dates.

5.4 Users select a place among ML suggestions that is given by the system or according to their own poll results that is created by the users.

5.5 After selection of the place each user views and votes for the suitable hotels and transportation opportunities for the given available dates and place.

5.8 Users are navigated to the related sites for purchasing tickets or reserve the determined place.

---

1. Name: Send chat to your travel group

2. Participating Actor: Users in the group

3. Entry Conditions:

3.1 Users should be logged in

3.2 Users should be in the same travel group

4. Exit Conditions: Users are navigated to the related sites for purchasing tickets or reserve the determined place.

5. Flow of Events:

5.1 Users navigate to the profile page

5.2 Users select the travel group that they want to have a trip with

5.3 Users send a chat to that group about the trip arrangement

---

1. Name: Search places to visit

2. Participating Actor: User

3. Entry Conditions:

3.1 User should be logged in

4. Exit Conditions: "Search" button is pressed with filled textfield

5. Flow of Events:

User:

5.1 Writes a place he/she wishes to search in the TextField at the top of the Home page.

5.2 Press the "Search" button.

---

1. Name: View selected destination info

2. Participating Actor: User

3. Entry Conditions:

3.1 User should be logged in to the system.

3.2 User should be on the selected Destination page.

4. Exit Conditions: User closes the selected Destination page.

5. Flow of Events:

User:

5.1 Searches and selects a place he/wishes to view.

5.2 View Preferred season information of a destination.

5.3 Scrolls down and views the review and rating section.

5.4 Clicks the “Back” button when he/she is done viewing.

---

1. Name: View route and destination suggestions

2. Participating Actor: User

3. Entry Conditions:

3.1 User should be logged in to the system.

4. Exit Conditions: User closes the Home page.

5. Flow of Events:

5.1 User switches to Home page.

5.2 User sees the most suitable destination options in the Home page.

---

1. Name: View activity suggestions

2. Participating Actor: User

3. Entry Conditions:

3.1 User should be logged in.

4. Exit Conditions: User closes the Home page.

5. Flow of Events:

5.1 User switches to Home page.

5.2 User selects a destination.

5.5 User sees the popular activities that are done in that destination.

---

1. Name: Rate previous trip location

2. Participating Actor: User

3. Entry Conditions:

:3.1 User should be logged into the system.

3.2 User should have a previous trip that is planned through Travela.

4. Exit Conditions: User rates the trip location.

5. Flow of Events:

User:

5.1 Switches to the Maps page.

5.2 Clicks to the trip that is seen in the World map.

5.3 Clicks to the “Rate” button.

5.4 Rates the trip location out of 5.

---

1. Name: Vote trip locations that is seen in Home page

2. Participating Actor: User

3. Entry Conditions: User should be logged into the system.

4. Exit Conditions: User rates the trip location.

5. Flow of Events:

User:

5.1 Switches to the Home page.

5.2 Clicks to the trip that is seen in the Home page.

5.3 Clicks to one of the “Dislike”, “Like” or “Love” buttons (like Netflix).

---

1. Name: See current travel groups

2. Participating Actor: User

3. Entry Conditions: User should be logged into the system.

4. Exit Conditions: User presses the “Logout” button or switches pages.

5. Flow of Events: User switches to the Profile page.

---

1. Name: Create polls among group

2. Participating Actor: User

3. Entry Conditions:

3.1 User should be logged in to the system.

3.2 User should be in a travel group.

4. Exit Conditions: Poll is created

5. Flow of Events:

User:

5.1 Switches to the Profile page.

5.2 Clicks to the travel group that he/she wants to create a poll.

5.3 Clicks to the “Create Poll” button.

5.4 Adds questions to the poll.

5.5 Clicks to the “Publish” button to send the poll to the group members.

---

1. Name: See upcoming trips

2. Participating Actor: User

3. Entry Conditions: User should be logged into the system.

4. Exit Conditions: User presses the “Logout” button or switch pages.

5. Flow of Events:

5.1 User switches to the Trips page.

5.2 User sees the upcoming trips that is planned by their travel groups.

---

1. Name: See previous trips of him/herself or his/her friends in a World map

2. Participating Actor: User

3. Entry Conditions:

3.1 User should be logged into the system.

3.2 User should be on the Maps page.

4. Exit Conditions: User presses the “Logout” button or switches pages.

5. Flow of Events:

5.1 User switches to the Maps page.

5.2 User views his/her map directly.

5.2 If the user wants to see their friends’ previous trips.

5.2.1 User views the friend list that is found in the Maps page.

5.2.2 User clicks to the friend name that has their own map.

5.3.3 User sees the related info (photo,review) of their friends.

---

1. Name: Send friend request

2. Participating Actor: User

3. Entry Conditions: User should be logged into the system.

4. Exit Conditions: Friend request is sent

5. Flow of Events:

5.1 User searches for a user name which the request will be sent.

5.2 User clicks the “Follow” button.

---

1. Name: Post photos belonging to the previous trips to the World map.

2. Participating Actor: User

3. Entry Conditions:

3.1 User should be logged into the system.

3.2 User should have a previous trip that is planned through Travela.

4. Exit Conditions: Trip (photos) is posted to the World map

5. Flow of Events:

User:

5.1 Switches to the Maps page.

5.2 Clicks to the trip that is seen in the World map.

5.3 Clicks to the “Add Photo” button.

5.4 Selects the photos to be posted.

5.5 Clicks “Post” button.

---

1. Name: Add a review of previous trips to the World map.

2. Participating Actor: User

3. Entry Conditions:

3.1 User should be logged into the system.

3.2 User should have a previous trip that is planned through Travela

4. Exit Conditions: Add review to trip

5. Flow of Events:

User:

5.1 Switches to the maps page

5.2 Clicks to the trip that is seen in the World map

5.3 Clicks to the “Add Review” button

5.4 Writes review

5.5 Clicks “Post” button

---

1. Name: Delete account

2. Participating Actor: User

3. Entry Conditions:

:3.1 User should be logged into the system.

3.2 User should have an active account.

4. Exit Conditions: User deletes the account.

5. Flow of Events:

User:

5.1 Switches to the profile page.

5.2 Clicks to the “Settings” button.

5.3 Clicks to the “Account” button.

5.4 Clicks to the “Delete Account” button.

5.5 Confirms to delete the account.

---

1. Name: Participate in questionnaire

2. Participating Actor: User

3. Entry Conditions: User should be logged into the system.

4. Exit Conditions: User completes the questionnaire.

5. Flow of Events:

User:

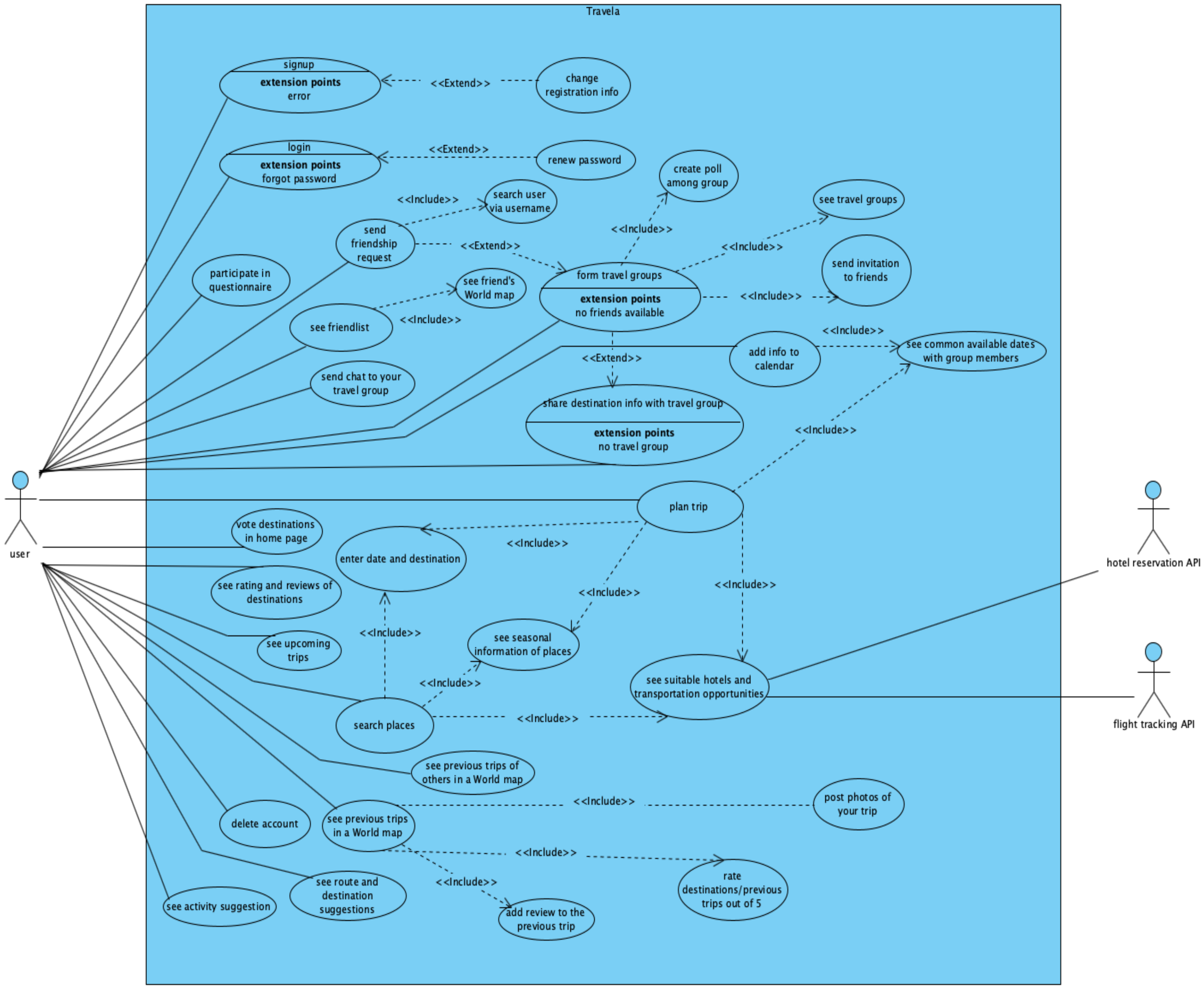
5.1 Switches to the profile page.

5.2 Clicks to the “Questionnaire” button.

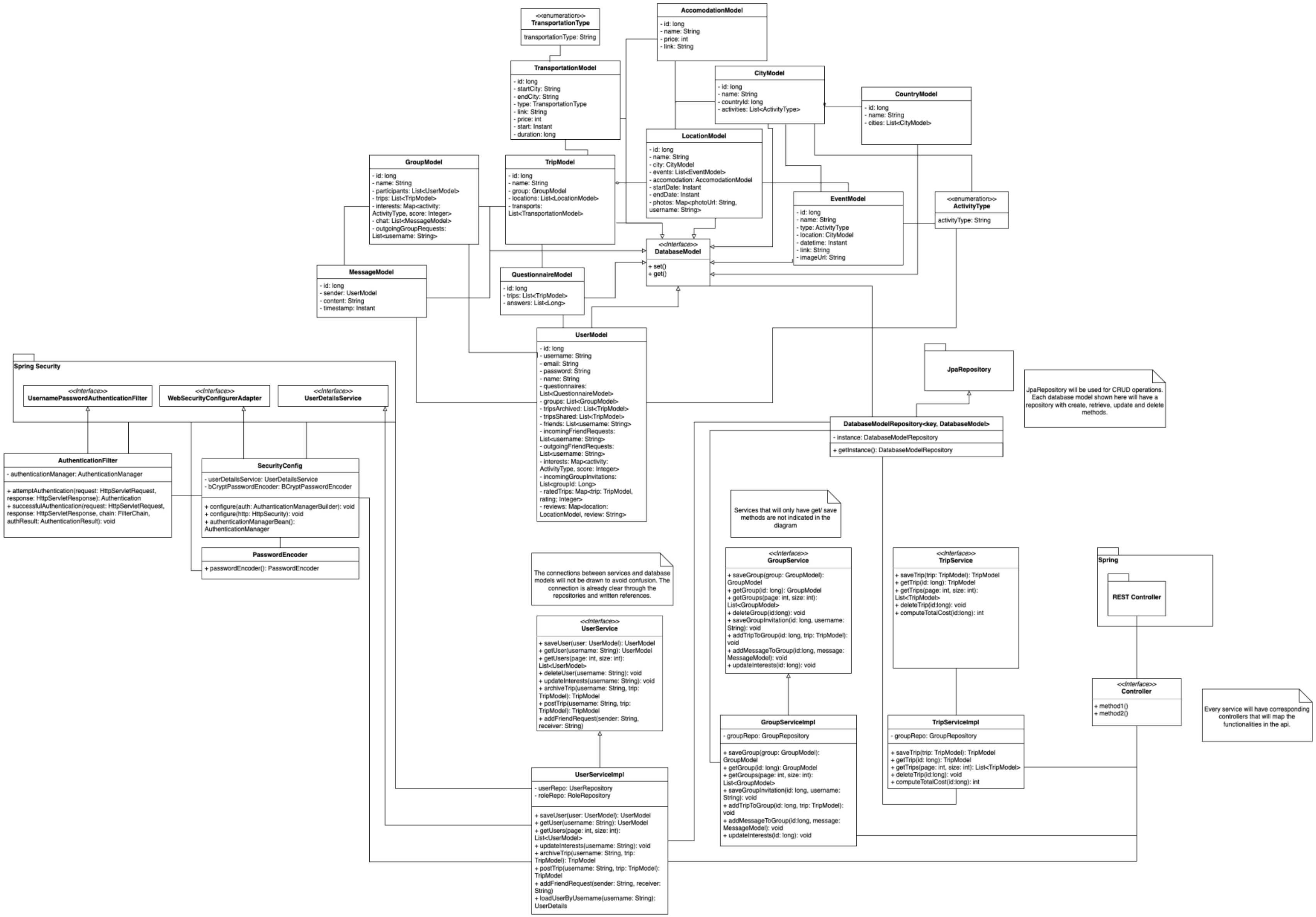
5.3 Completes the questionnaire.

5.4 Clicks to the “Send” button.

3.5.2 Use-Case Model



3.5.3 Object and Class Model


<<enumeration>>
TransportationType
transportationType: String
TransportationModel
- id: long
- startCity: String
- endCity: String
- type: TransportationType
- link: String
- price: int
- start: Instant
- duration: long
AccomodationModel
- id: long
- name: String
- price: int
- link: String
CityModel
- id: long
- name: String
- countryId: long
- activities: List<ActivityType>
CountryModel
- id: long
- name: String
- cities: List<CityModel>
LocationModel
- id: long
- name: String
- city: CityModel
- events: List<EventModel>
- accomodation: AccomodationModel
- startDate: Instant
- endDate: Instant
- photos: Map<photoUrl: String, username: String>
GroupModel
- id: long
- name: String
- participants: List<UserModel>
- trips: List<TripModel>
- interests: Map<activity: ActivityType, score: Integer>
- chat: List<MessageModel>
- outgoingGroupRequests: List<username: String>
TripModel
- id: long
- name: String
- group: GroupModel
- locations: List<LocationModel>
- transports: List<TransportationModel>
EventModel
- id: long
- name: String
- type: ActivityType
- location: CityModel
- datetime: Instant
- link: String
- imageUrl: String
<<enumeration>>
ActivityType
activityType: String
<<Interface>>
DatabaseModel
+ set()
+ get()
MessageModel
- id: long
- sender: UserModel
- content: String
- timestamp: Instant
QuestionnaireModel
- id: long
- trips: List<TripModel>
- answers: List<Long>
UserModel
- id: long
- username: String
- email: String
- password: String
- name: String
- questionnaires: List<QuestionnaireModel>
- groups: List<GroupModel>
- tripsArchived: List<TripModel>
- tripsShared: List<TripModel>
- friends: List<username: String>
- incomingFriendRequests: List<username: String>
- outgoingFriendRequests: List<username: String>
- interests: Map<activity: ActivityType, score: Integer>
- incomingGroupInvitations: List<groupId: Long>
- ratedTrips: Map<trip: TripModel, rating; Integer>
- reviews: Map<location: LocationModel, review: String>
JpaRepository
DatabaseModelRepository<key, DatabaseModel>
- instance: DatabaseModelRepository
+ getInstance(): DatabaseModelRepository
Spring Security
<<Interface>>
UsernamePasswordAuthenticationFilter
<<Interface>>
WebSecurityConfigurerAdapter
<<Interface>>
UserDetailsService
AuthenticationFilter
- authenticationManager: AuthenticationManager
+ attemptAuthentication(request: HttpServletRequest, response: HttpServletResponse): Authentication
+ successfulAuthentication(request: HttpServletRequest, response: HttpServletResponse, chain: FilterChain, authResult: AuthenticationResult): void
SecurityConfig
- userDetailsService: UserDetailsService
- bCryptPasswordEncoder: BCryptPasswordEncoder
+ configure(auth: AuthanticationManagerBuilder): void
+ configure(http: HttpSecurity): void
+ authenticationManagerBean(): AuthenticationManager
PasswordEncoder
+ passwordEncoder(): PasswordEncoder
<<Interface>>
UserService
+ saveUser(user: UserModel): UserModel
+ getUser(username: String): UserModel
+ getUsers(page: int, size: int): List<UserModel>
+ deleteUser(username: String): void
+ updateInterests(username: String): void
+ archiveTrip(username: String, trip: TripModel): TripModel
+ postTrip(username: String, trip: TripModel): TripModel
+ addFriendRequest(sender: String, receiver: String)
UserServiceImpl
- userRepo: UserRepository
- roleRepo: RoleRepository
+ saveUser(user: UserModel): UserModel
+ getUser(username: String): UserModel
+ getUsers(page: int, size: int): List<UserModel>
+ updateInterests(username: String): void
+ archiveTrip(username: String, trip: TripModel): TripModel
+ postTrip(username: String, trip: TripModel): TripModel
+ addFriendRequest(sender: String, receiver: String)
+ loadUserByUsername(username: String): UserDetails
<<Interface>>
GroupService
+ saveGroup(group: GroupModel): GroupModel
+ getGroup(id: long): GroupModel
+ getGroups(page: int, size: int): List<GroupModel>
+ deleteGroup(id:long): void
+ saveGroupInvitation(id: long, username: String): void
+ addTripToGroup(id: long, trip: TripModel): void
+ addMessageToGroup(id:long, message: MessageModel): void
+ updateInterests(id: long): void
GroupServiceImpl
- groupRepo: GroupRepository
+ saveGroup(group: GroupModel): GroupModel
+ getGroup(id: long): GroupModel
+ getGroups(page: int, size: int): List<GroupModel>
+ saveGroupInvitation(id: long, username: String): void
+ addTripToGroup(id: long, trip: TripModel): void
+ addMessageToGroup(id:long, message: MessageModel): void
+ updateInterests(id: long): void
<<Interface>>
TripService
+ saveTrip(trip: TripModel): TripModel
+ getTrip(id: long): TripModel
+ getTrips(page: int, size: int): List<TripModel>
+ deleteTrip(id:long): void
+ computeTotalCost(id:long): int
TripServiceImpl
- groupRepo: GroupRepository
+ saveTrip(trip: TripModel): TripModel
+ getTrip(id: long): TripModel
+ getTrips(page: int, size: int): List<TripModel>
+ deleteTrip(id:long): void
+ computeTotalCost(id:long): int
Spring
REST Controller
<<Interface>>
Controller
+ method1()
+ method2()

JpaRepository will be used for CRUD operations. Each database model shown here will have a repository with create, retrieve, update and delete methods.

Services that will only have get/ save methods are not indicated in the diagram

The connections between services and database models will not be drawn to avoid confusion. The connection is already clear through the repositories and written references.

Every service will have corresponding controllers that will map the functionalities in the api.

Table 1. Descriptions of classes.

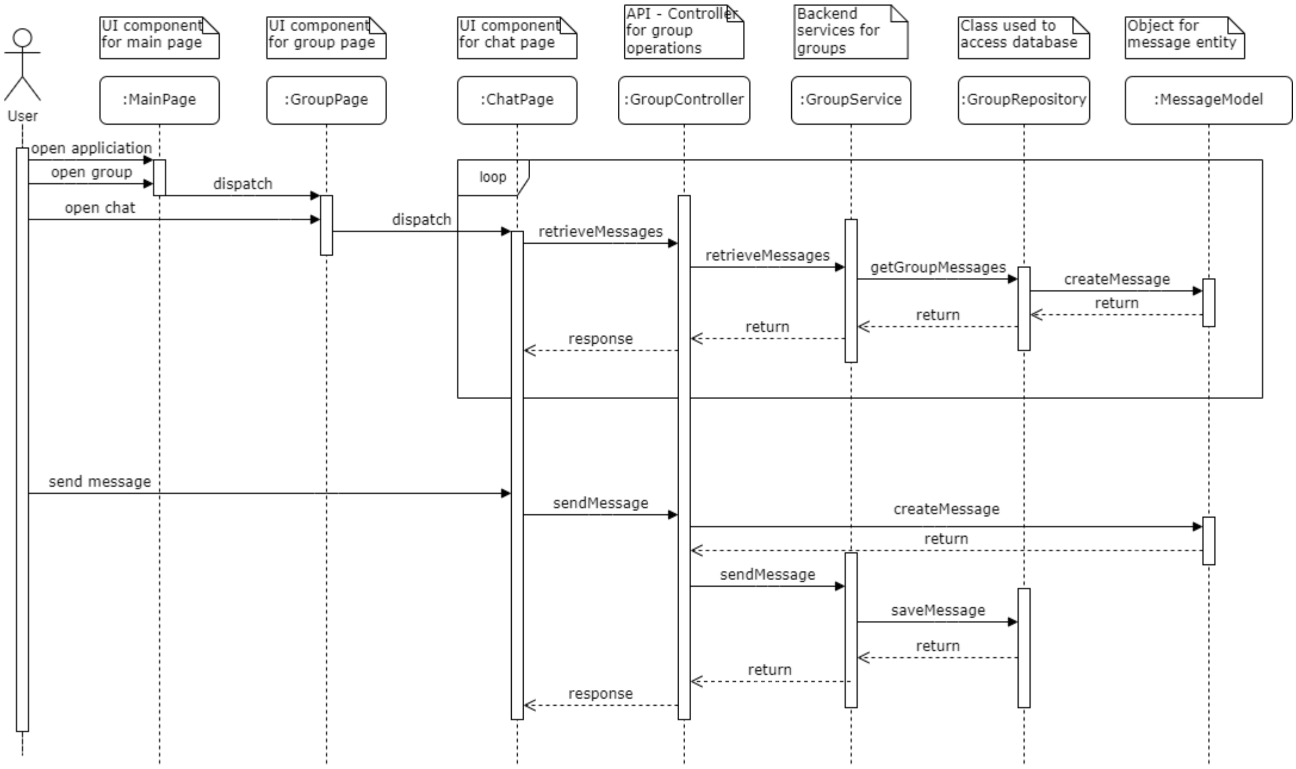
Class	Description
DatabaseModel	Interface for database entity models.
UserModel	Database model for user object. Stores data related to a user of the application.
GroupModel	Database model for group object. Stores data related to groups formed by users.
TripModel	Database model for trip object. Stores data related to trips added to a group (does not have to belong to a group, but mostly does)
QuestionnaireModel	Database model for questionnaire object. Stores data related to questionnaires taken by users.
LocationModel	Database model for location object. Stores data related to a location in a trip.
TransportationModel	Database model for transportation object. Stores data related to transportation between two locations.
TransportationType	Enumeration for different types of transportation.
AccomodationModel	Database model for accomodation object. Stores data related to accommodation in a location.
CityModel	Database model for city objects. Stores data related to a city that can be added as a location.
CountryModel	Database model for country object. Stores data related to a country that contains cities.
EventModel	Database model for event object. Stores data related to an event in a city.
ActivityType	Enumeration for different types of activities an event can be.
MessageModel	Database model for message object. Stores data related to a message sent to a group.
UserService	Interface for database operations related to user objects.
UserServiceImpl	Implementation of UserService.
GroupService	Interface for database operations related to group objects.
GroupServiceImpl	Implementation of GroupService.
TripService	Interface for database operations related to trip objects.
TripServiceImpl	Implementation of TripService.
JpaRepository	Third party package to be used for CRUD operations.
DatabaseModelRepository	Singleton representing the database model repository.
REST Controller	Third party package provided by Spring.
Controller	Controller for each service.

UserNamePassword AuthenticationFilter	Third party interface provided by Spring Security.
WebSecurityConfigurerAdapter	Third party interface provided by Spring Security.
UserDetailsService	Third party interface provided by Spring Security.
AuthenticationFilter	Customization of authentication filter manager from Spring Security.
SecurityConfig	Customization of security configurations from Spring Security.
PasswordEncoder	Customization of password encoder from Spring Security.

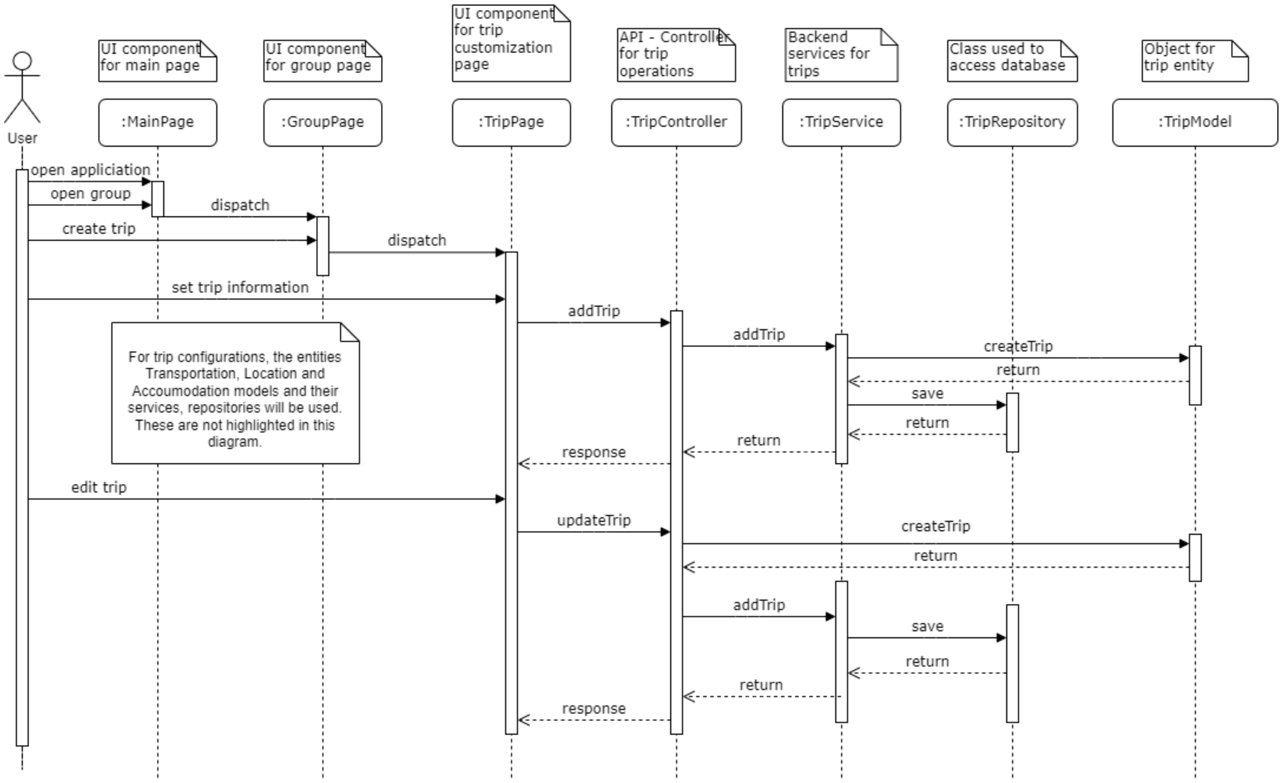
3.5.4 Dynamic Models

3.5.4.1 Sequence Diagrams

Open travel group chat and send messages

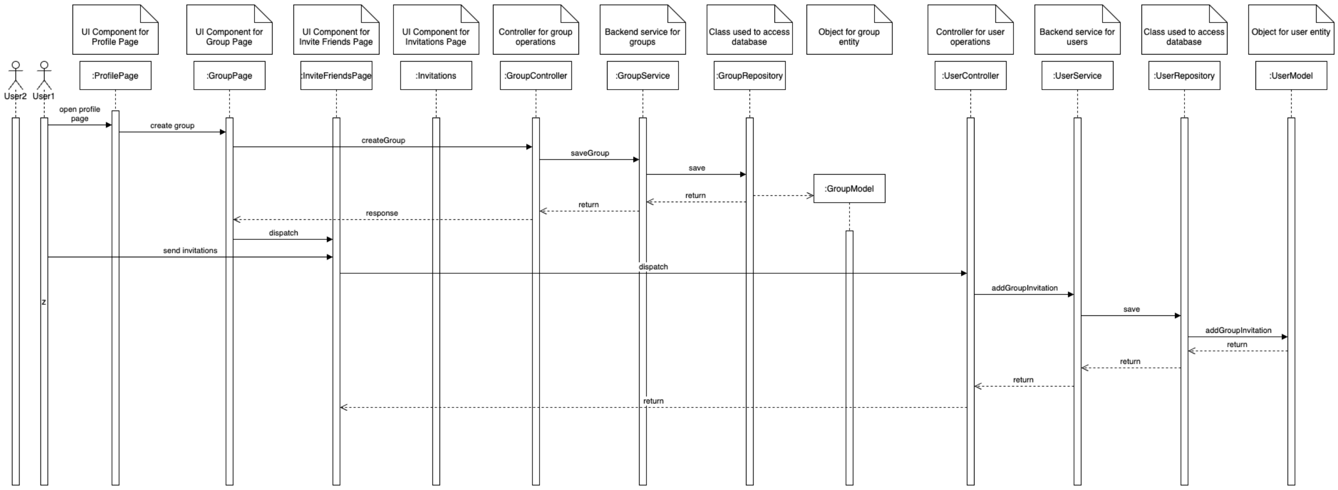


Arrange trip and edit trip

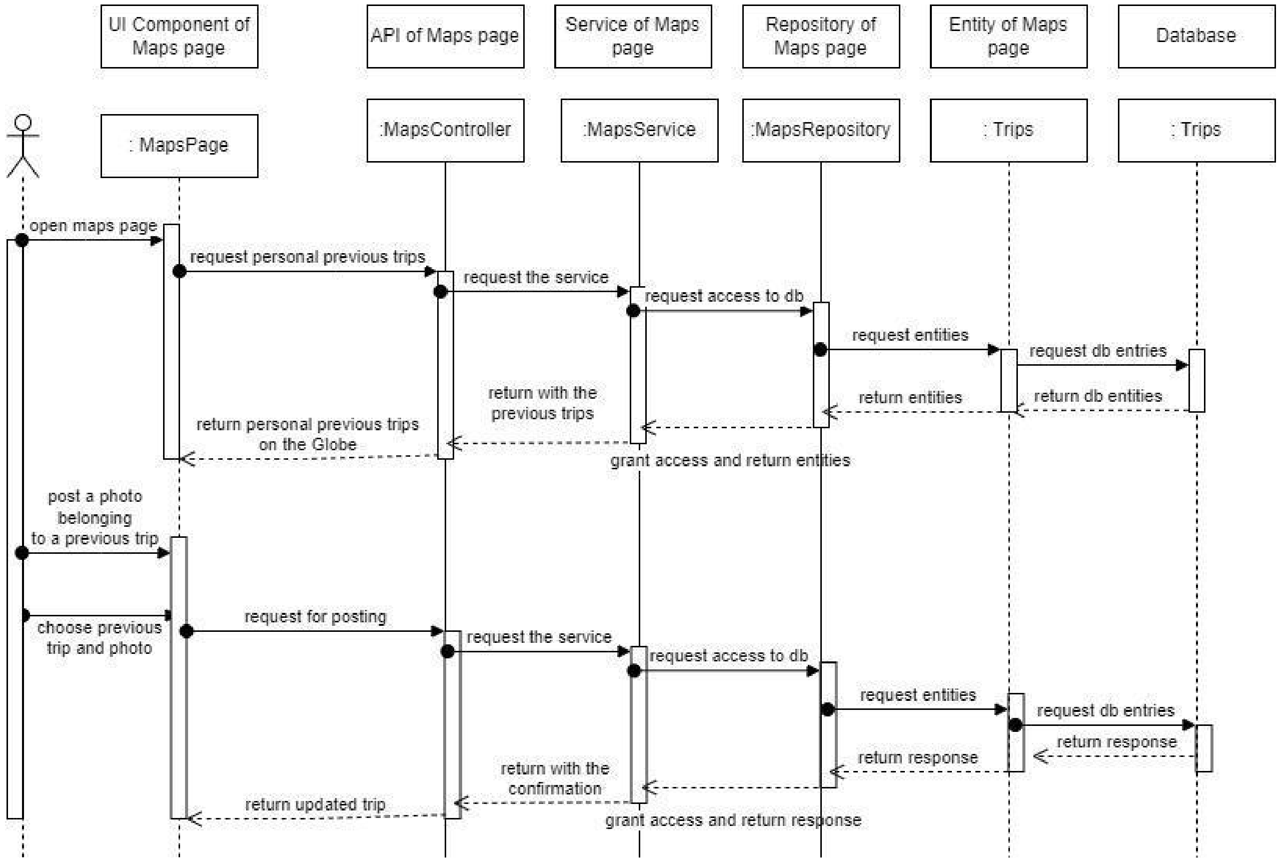


For trip configurations, the entities Transportation, Location and Accoumodation models and their services, repositories will be used. These are not highlighted in this diagram.

Create new group and invite friends

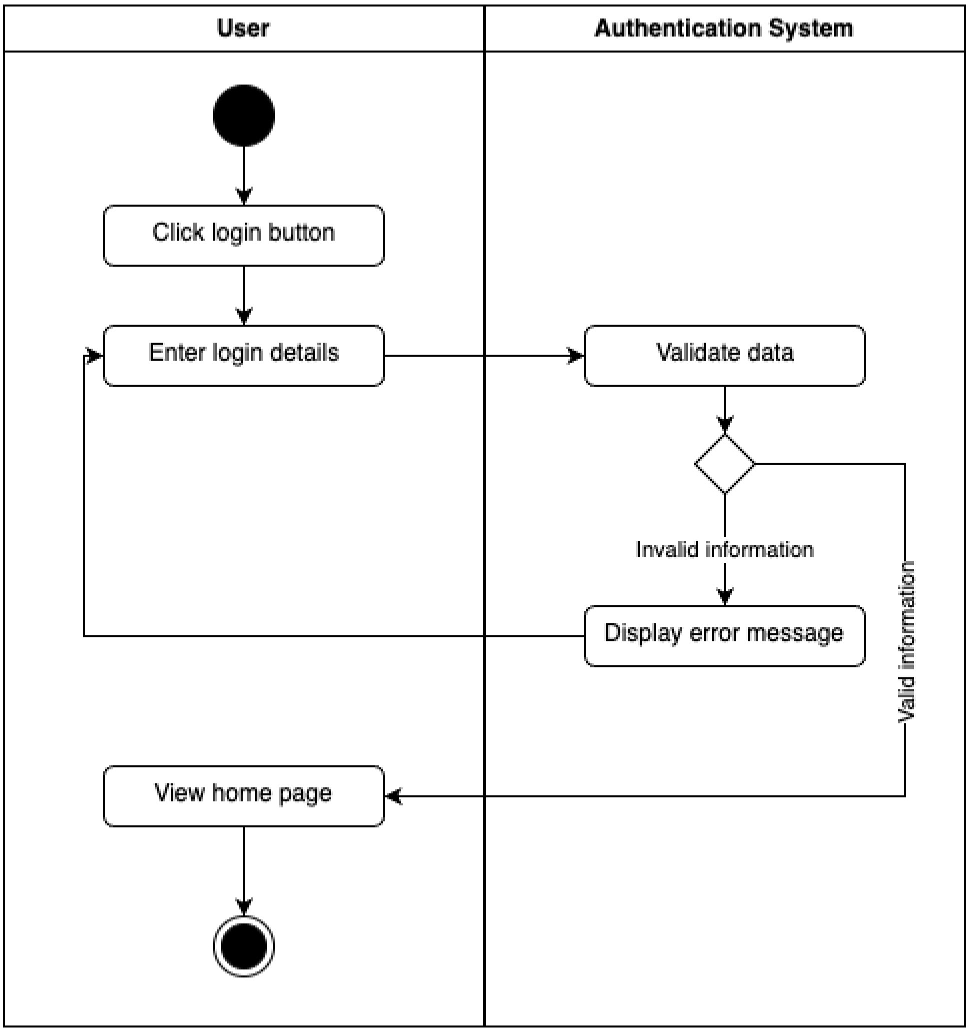


View the previous trips on globe and post photos belonging to previous trips

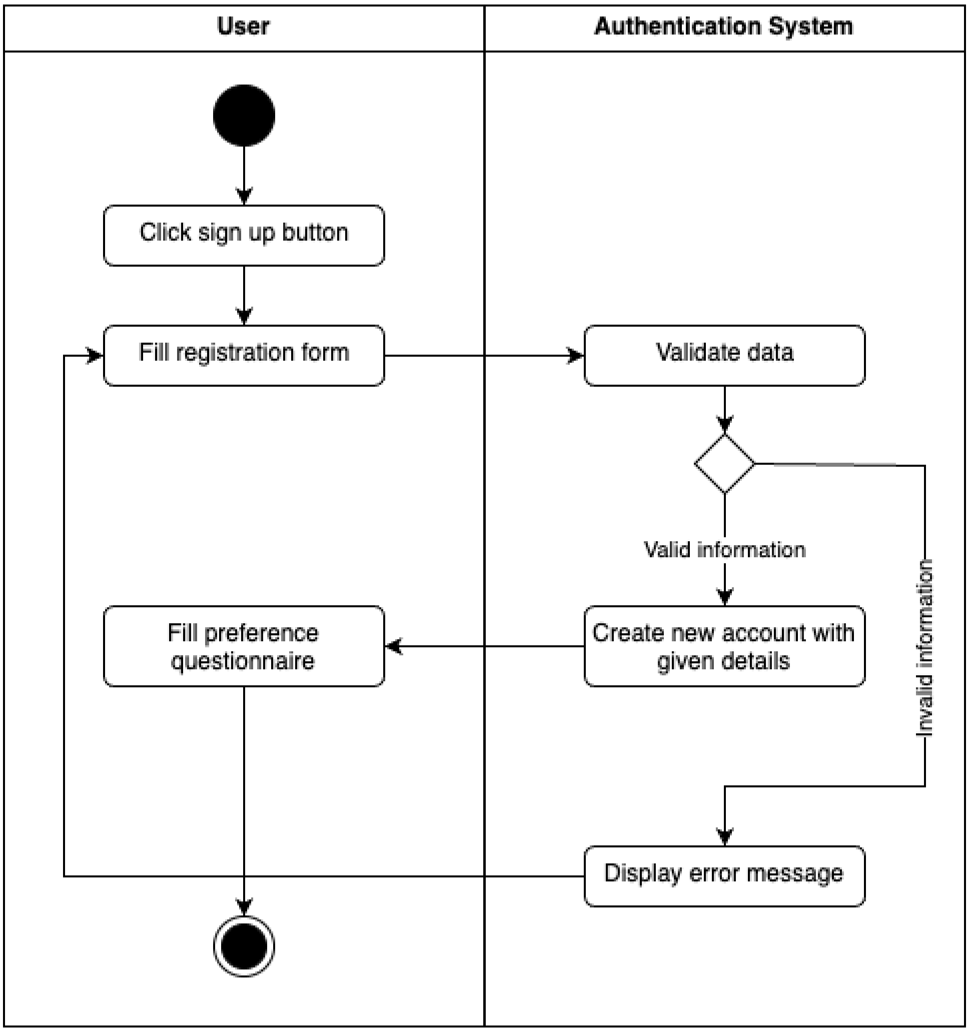


3.5.4.2 Activity Diagrams

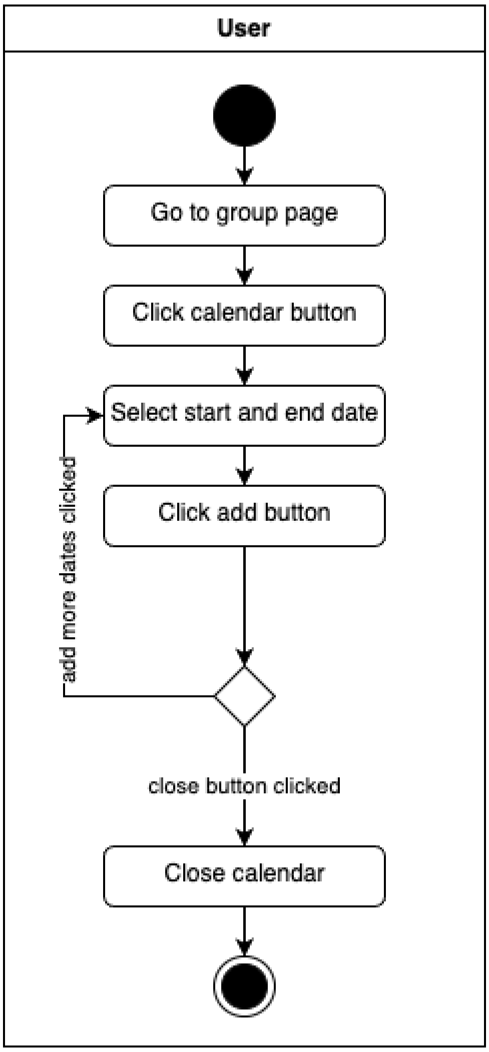
Login



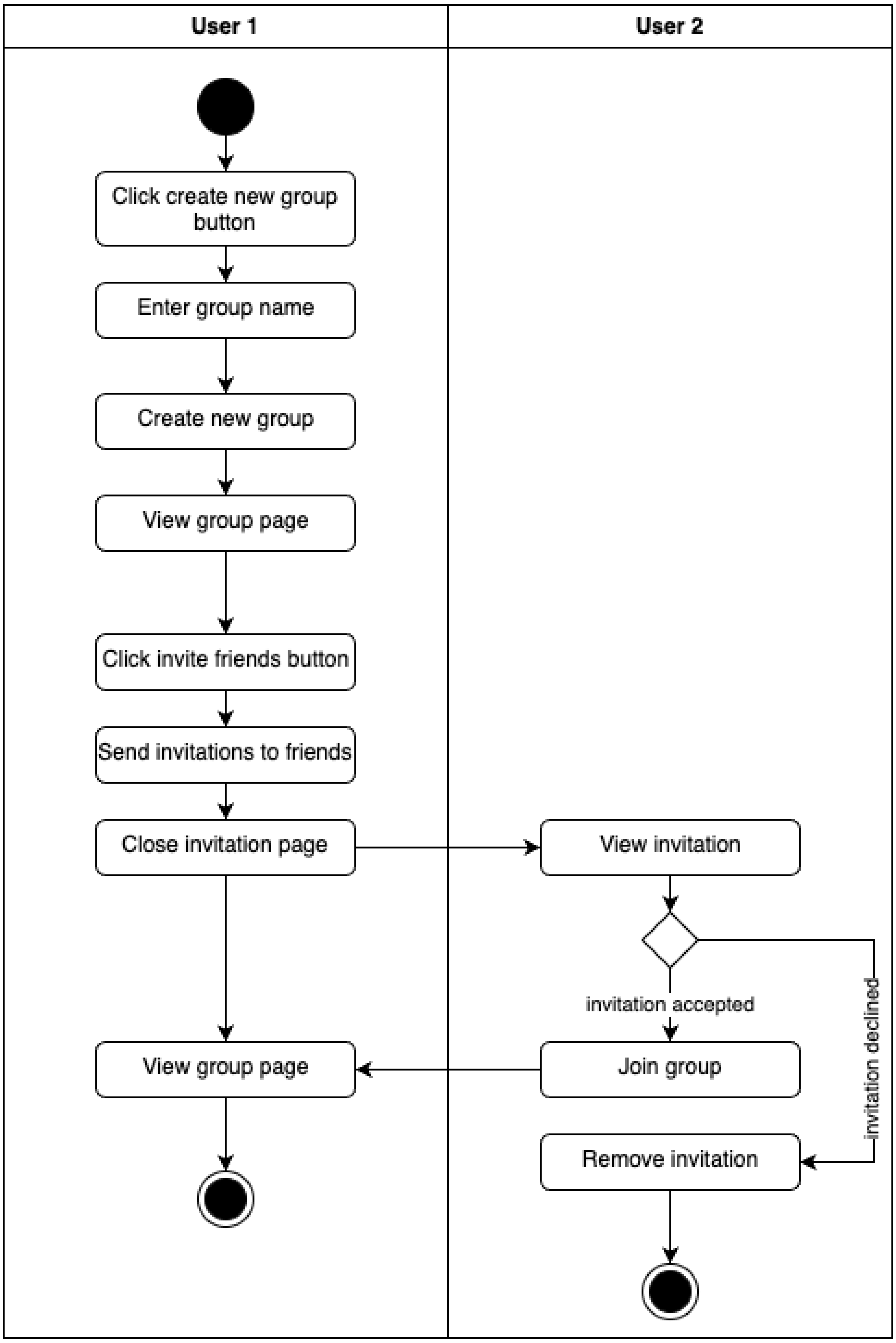
Signup


User
Authentication System
Click sign up button
Fill registration form
Validate data
Valid information
Invalid information
Create new account with given details
Fill preference questionnaire
Display error message

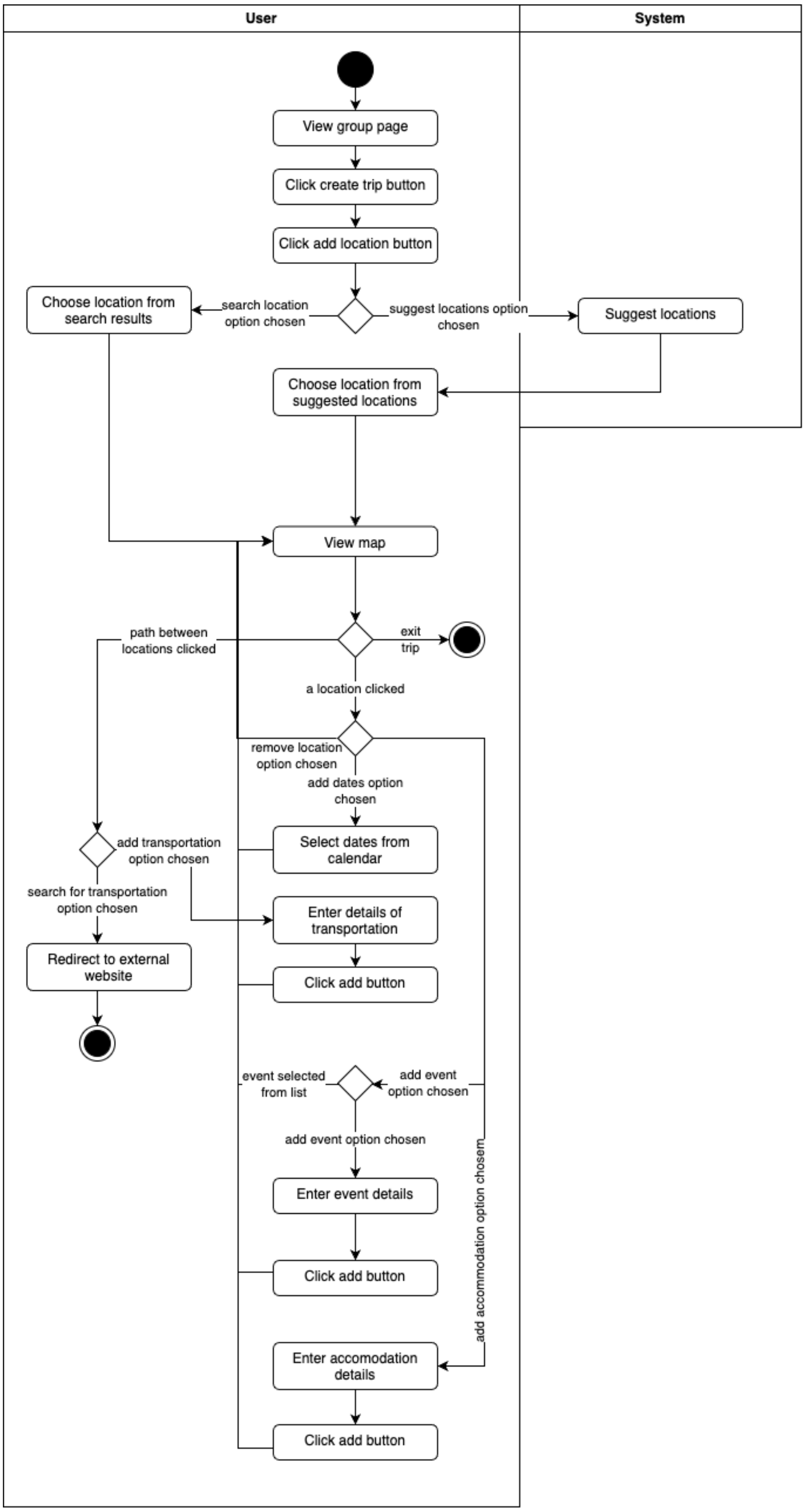
Add Available Dates



Create New Travel Group

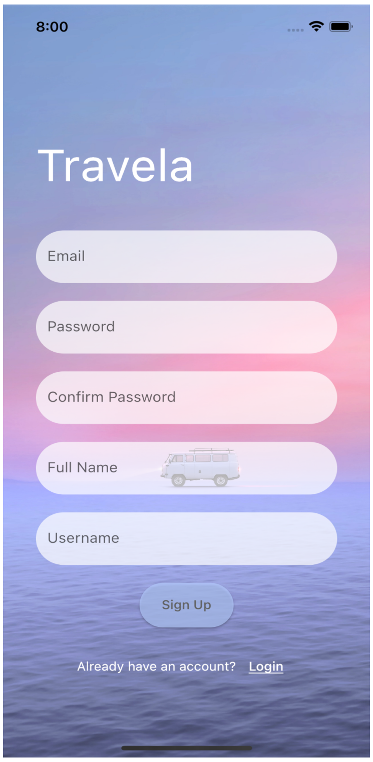


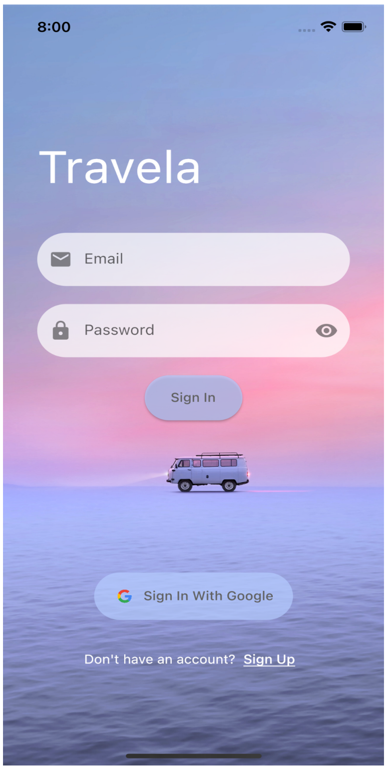
Create and Edit Trip


User
System
View group page
Click create trip button
Click add location button
Choose location from search results
search location option chosen
suggest locations option chosen
Suggest locations
Choose location from suggested locations
View map
path between locations clicked
exit trip
a location clicked
remove location option chosen
add dates option chosen
Select dates from calendar
add transportation option chosen
search for transportation option chosen
Enter details of transportation
Redirect to external website
Click add button
event selected from list
add event option chosen
add event option chosen
Enter event details
Click add button
add accommodation option chosem
Enter accomodation details
Click add button

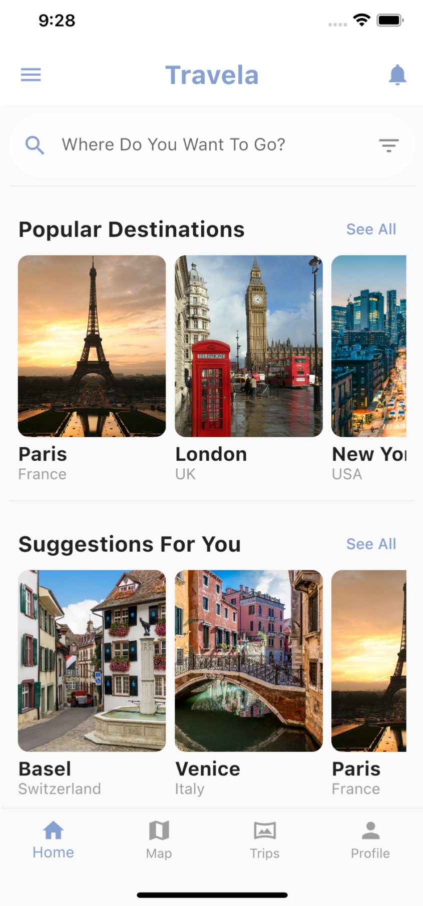
3.5.5 User Interface

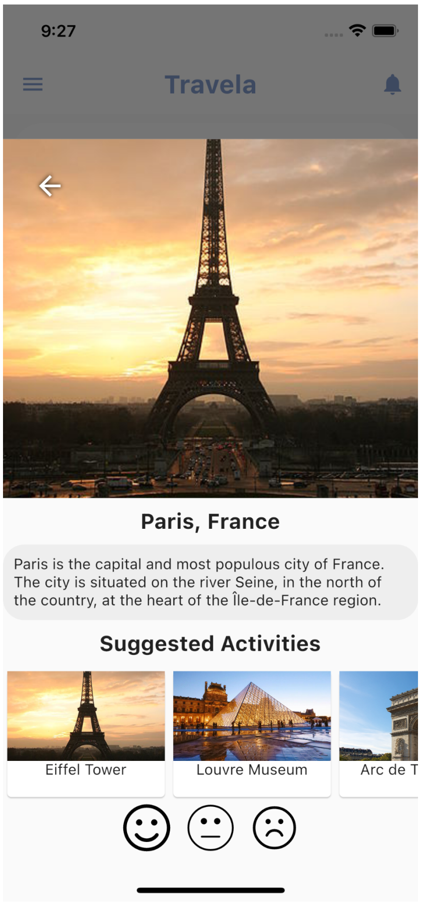
3.5.5.1 Sign Up - Login Page





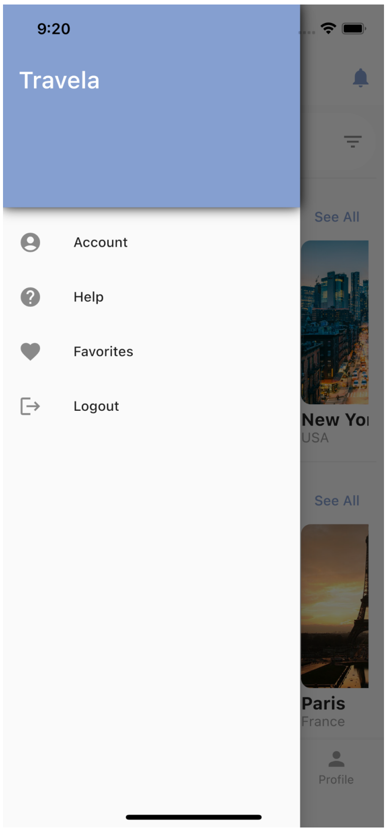
3.5.5.2 Home Page





Paris is the capital and most populous city of France. The city is situated on the river Seine, in the north of the country, at the heart of the Île-de-France region.

3.5.5.4 Drawer



9:20

Travela

Account

Help

Favorites

Logout

See All

New Yo

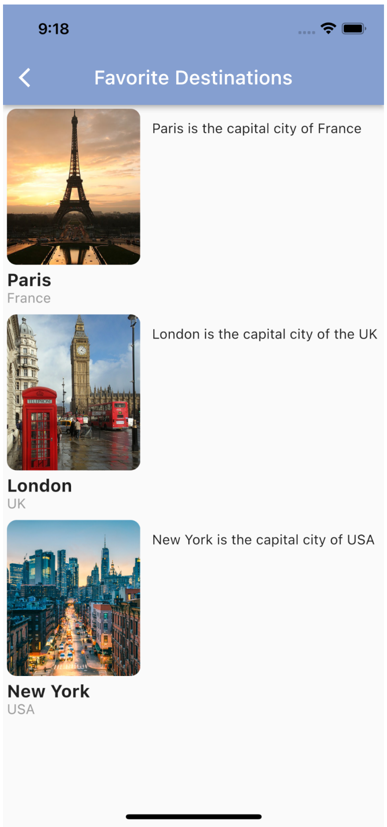
USA

See All

Paris

France

Profile



9:18

Favorite Destinations

Paris is the capital city of France

Paris

France

London is the capital city of the UK

London

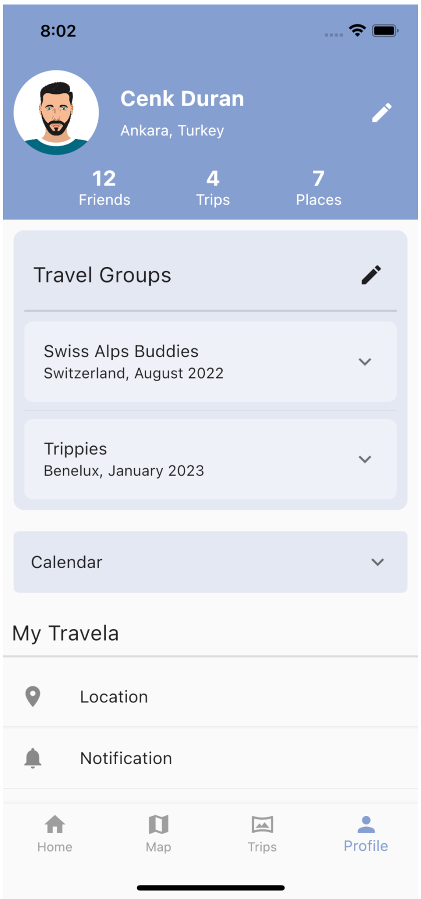
UK

New York is the capital city of USA

New York

USA

3.5.5.5 Profile Page



8:02

Cenk Duran

Ankara, Turkey

12 Friends

4 Trips

7 Places

Travel Groups

Swiss Alps Buddies

Switzerland, August 2022

Trippies

Benelux, January 2023

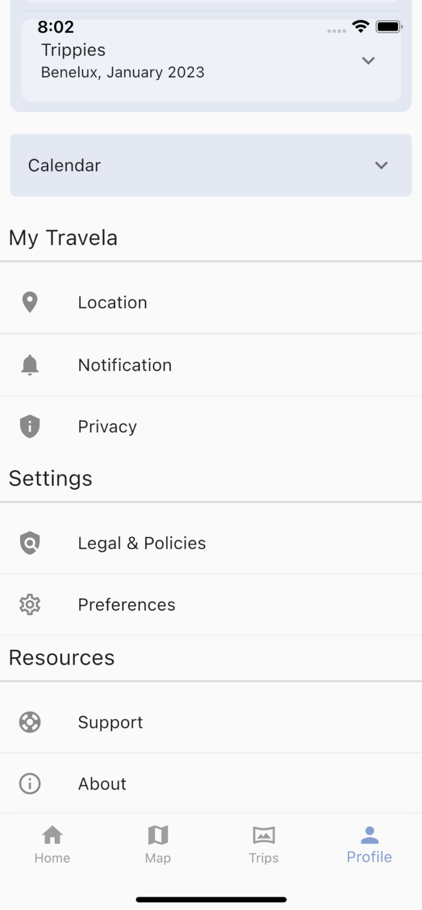
Calendar

My Travela

Location

Notification

Home Map Trips Profile



8:02

Trippies

Benelux, January 2023

Calendar

My Travela

Location

Notification

Privacy

Settings

Legal & Policies

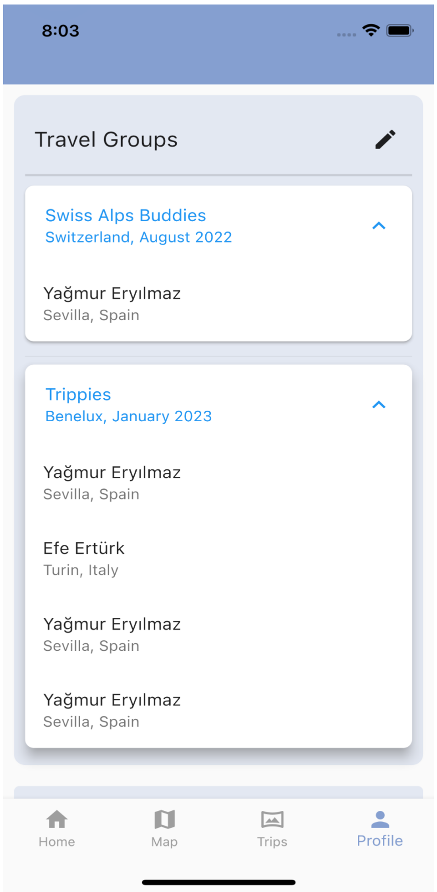
Preferences

Resources

Support

About

Home Map Trips Profile



8:03

Travel Groups

Swiss Alps Buddies

Switzerland, August 2022

Yağmur Eryılmaz

Sevilla, Spain

Trippies

Benelux, January 2023

Yağmur Eryılmaz

Sevilla, Spain

Efe Ertürk

Turin, Italy

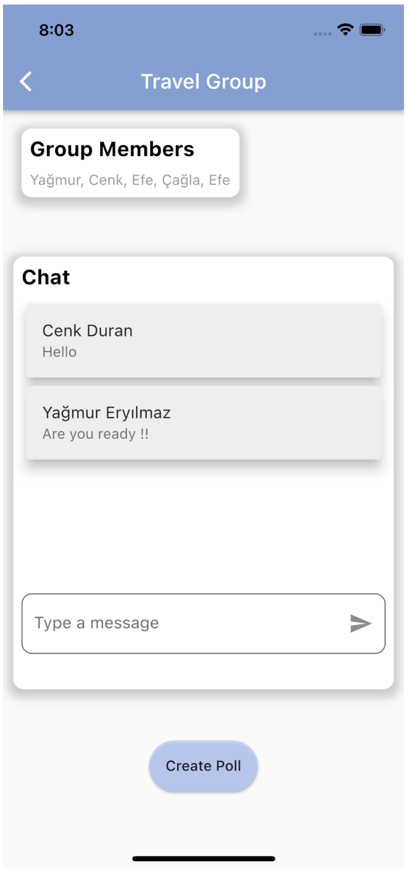
Yağmur Eryılmaz

Sevilla, Spain

Yağmur Eryılmaz

Sevilla, Spain

Home Map Trips Profile



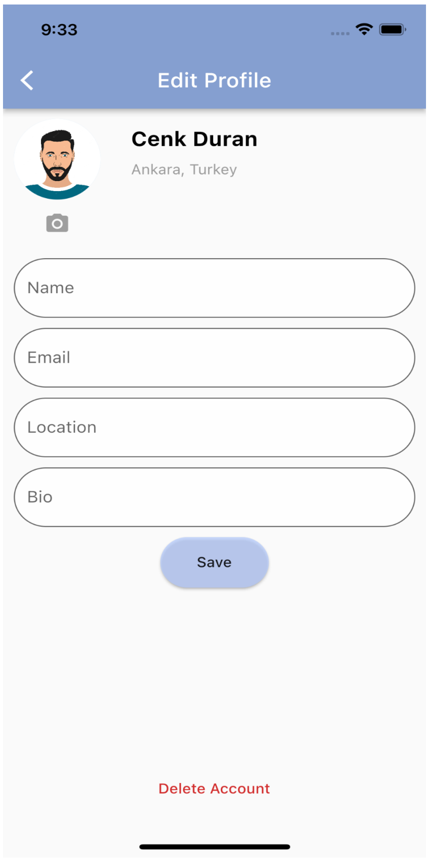
Group Members

Yağmur, Cenk, Efe, Çağla, Efe

Chat

Cenk Duran
Hello

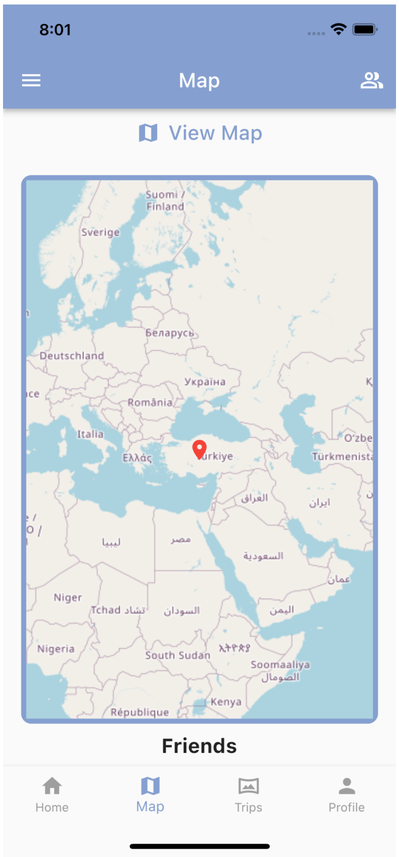
Yağmur Eryılmaz
Are you ready !!

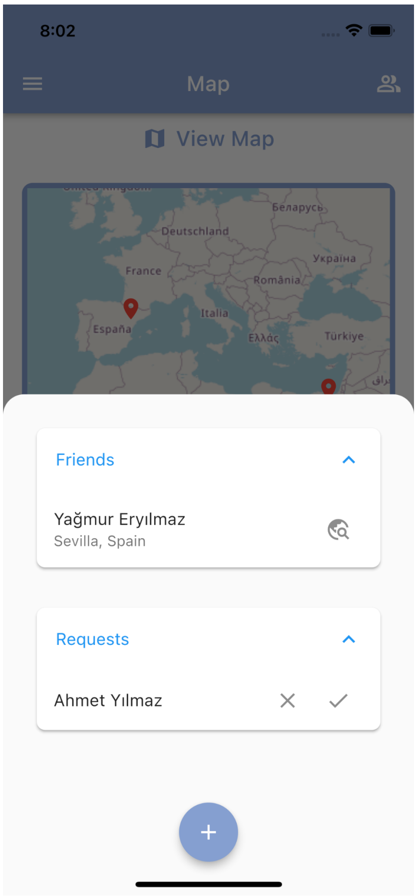


Cenk Duran

Ankara, Turkey

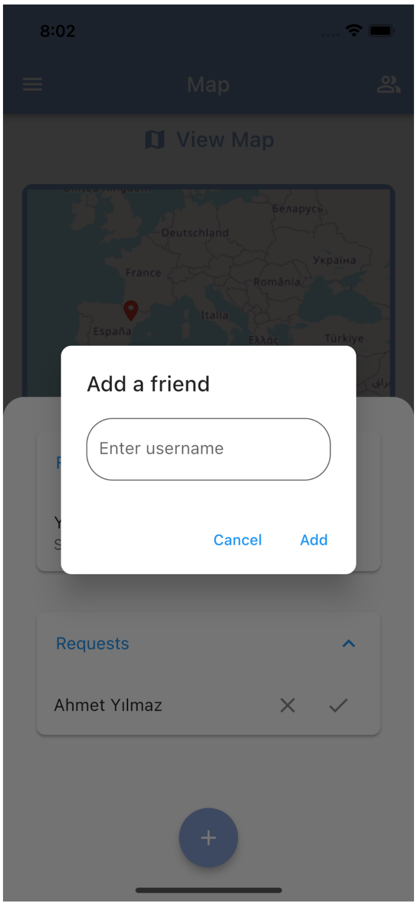
3.5.5.6 Maps Page





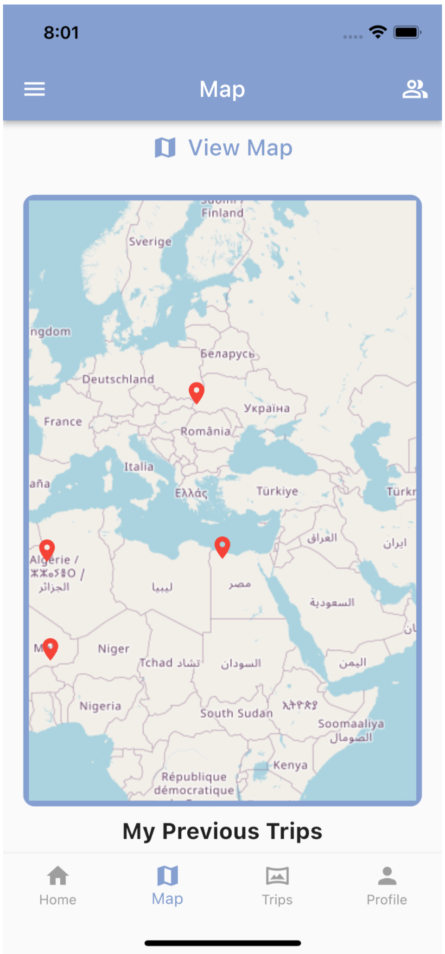
Yağmur Eryılmaz
Sevilla, Spain

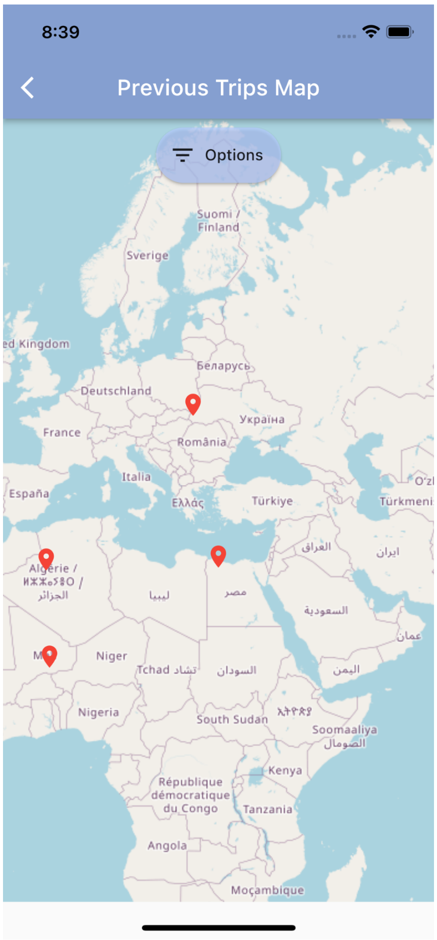
Ahmet Yılmaz

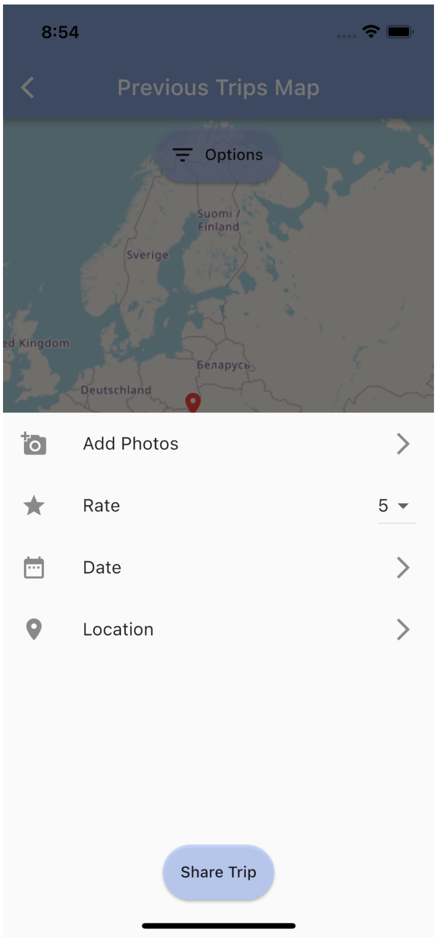


Ahmet Yılmaz

3.5.5.6.1 Seeing previous trips perform add photo and rate

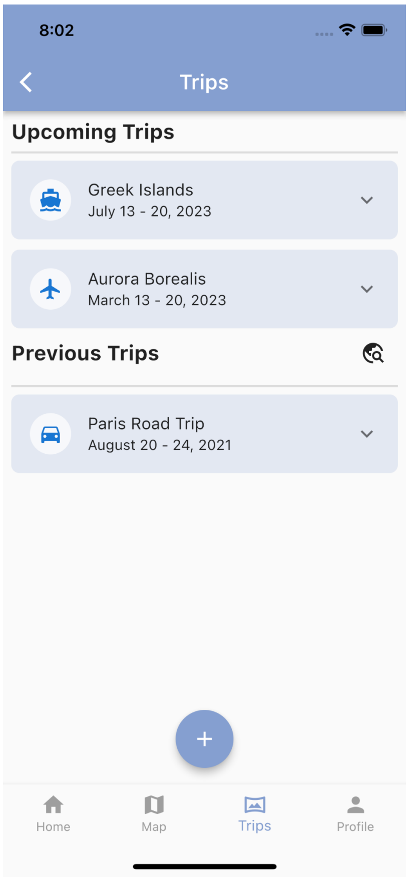






Rate 5

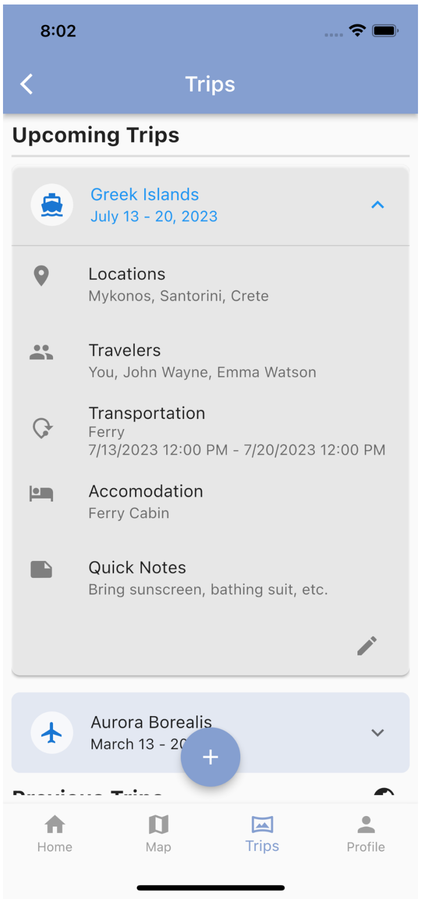
3.5.5.7 Trips Page



Greek Islands
July 13 - 20, 2023

Aurora Borealis
March 13 - 20, 2023

Paris Road Trip
August 20 - 24, 2021



Greek Islands
July 13 - 20, 2023

Locations
Mykonos, Santorini, Crete

Travelers
You, John Wayne, Emma Watson

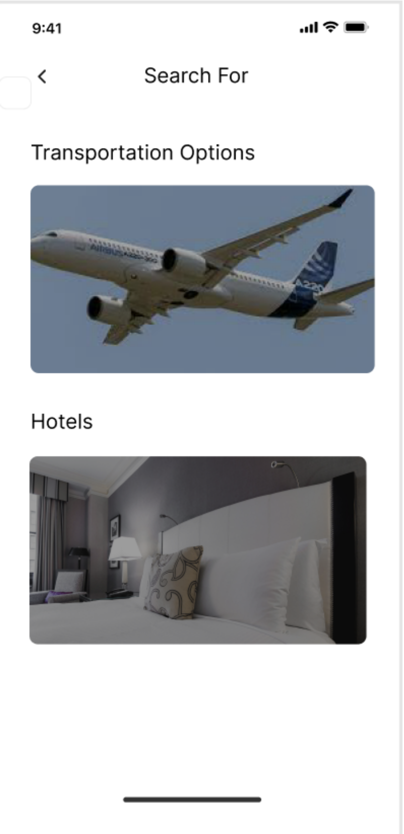
Transportation
Ferry
7/13/2023 12:00 PM - 7/20/2023 12:00 PM

Accomodation
Ferry Cabin

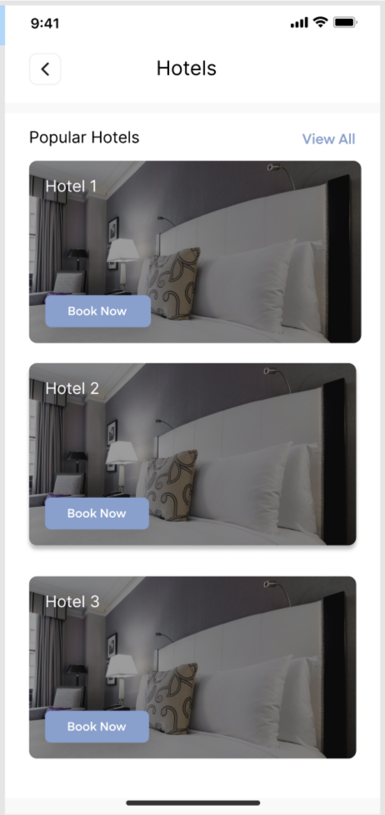
Quick Notes
Bring sunscreen, bathing suit, etc.

Aurora Borealis
March 13 - 20

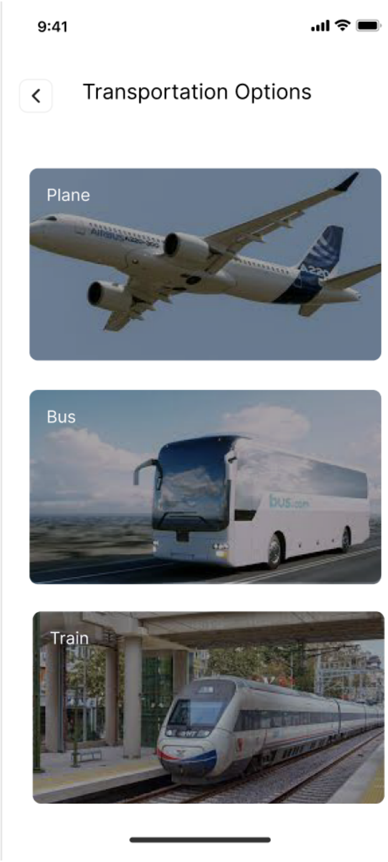
3.5.5.8 After entering date and destination to the search bar in home Page or editing an upcoming trip from trips page

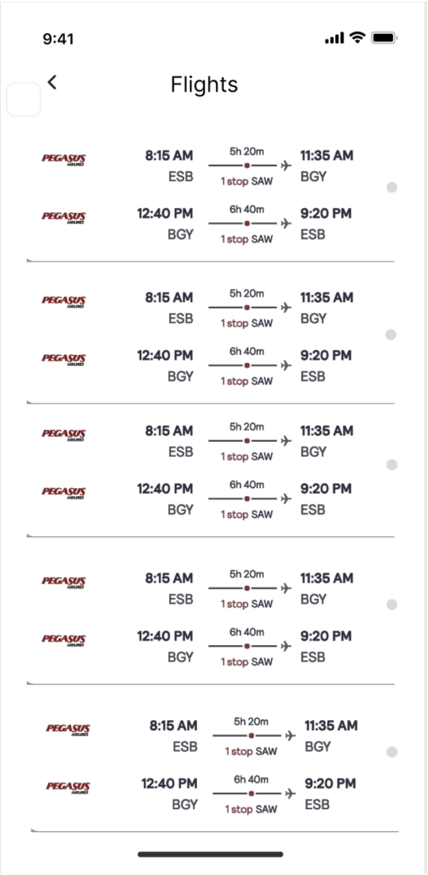


3.5.5.9 Hotel Info Page



3.5.5.10 Transportation Options





PEGASUS	8:15 AM ESB	5h 20m 1 stop SAW	11:35 AM BGY
PEGASUS	12:40 PM BGY	6h 40m 1 stop SAW	9:20 PM ESB
PEGASUS	8:15 AM ESB	5h 20m 1 stop SAW	11:35 AM BGY
PEGASUS	12:40 PM BGY	6h 40m 1 stop SAW	9:20 PM ESB
PEGASUS	8:15 AM ESB	5h 20m 1 stop SAW	11:35 AM BGY
PEGASUS	12:40 PM BGY	6h 40m 1 stop SAW	9:20 PM ESB
PEGASUS	8:15 AM ESB	5h 20m 1 stop SAW	11:35 AM BGY
PEGASUS	12:40 PM BGY	6h 40m 1 stop SAW	9:20 PM ESB
PEGASUS	8:15 AM ESB	5h 20m 1 stop SAW	11:35 AM BGY
PEGASUS	12:40 PM BGY	6h 40m 1 stop SAW	9:20 PM ESB

4 Test Cases

4.1 Functional Test Cases

1. Test Case ID: 1

2. Test Case Description: Check the response when invalid or missing information is given during sign-up.

3. Flow of Events:

3.1 Fill the registration form with an already existing account information, invalid email, or do not fill in the necessary fields.

3.2 Click the “Sign Up” button

4. Expected Result: Sign Up/ Registration fails and an error message appears.

5. Priority: Critical

---

1. Test Case ID: 2

2. Test Case Description: Check the database when the sign-up process succeeded.

3. Flow of Events:

3.1 Fill the registration form with valid information and fill in all necessary fields.

3.2 Click the “Sign Up” button

3.3 Check the database to see whether the new user has been added with the given information.

4. Expected Result: A new entry should be seen in the database with the given information.

5. Priority: Critical

---

1. Test Case ID: 3

2. Test Case Description: Check the response when Google sign-up functionality is used.

3. Flow of Events:

3.1 Click “Sign up with Google” button

3.2 Choose the Google account

3.3 Agree with the terms and conditions and authorize

4. Expected Result: A new entry should be seen in the database with the given sign up information and Login succeeds

5. Priority: Major

---

1. Test Case ID: 4

2. Test Case Description: Check the response when Google sign-in functionality is used.

3. Flow of Events:

3.1 Click “Sign in with Google” button

3.2 Choose the Google account

4. Expected Result: Login should succeed

5. Priority: Major

---

1. Test Case ID: 5

2. Test Case Description: Check the response when an invalid password or email is given during login

3. Flow of Events:

3.1 Enter the email

3.2 Enter the password

3.3 Click the “Sign In” button

4. Expected Result: Login fails and an error message appears.

5. Priority: Critical

---

1. Test Case ID: 6

2. Test Case Description: Check the response when one of the necessary fields is empty during login

3. Flow of Events:

3.1 Leave empty one of the necessary fields

3.2 Click the “Sign In” button

4. Expected Result: Login fails and an error message appears.

5. Priority: Critical

---

1. Test Case ID: 7

2. Test Case Description: Check the response during profile edit.

3. Flow of Events:

3.1 Go to the Profile Page.

3.2 Click “Edit ” icon in the top right corner.

3.3 Fill the fields that you want to change.

3.2 Click the “Save” button.

4. Expected Result: Profile information should be updated and a success message appears.

5. Priority: Major

---

1. Test Case ID: 8

2. Test Case Description: Check the response during profile edit when all fields are empty.

3. Flow of Events:

3.1 Go to the Profile Page.

3.2 Click “Edit ” icon in the top right corner.

3.3 Leave all the fields empty.

3.2 Click the “Save” button.

4. Expected Result: Profile information should not be updated and an error message appears.

5. Priority: Major

---

1. Test Case ID: 9

2. Test Case Description: Check the database when a travel group is formed.

3. Flow of Events:

3.1 Go to the Profile Page

3.2 Click the “Form Group” button

3.3 Select friends to send an invitation and press the “Send Invitation” button

3.4 Click “Close” button when he/she is done sending invitations

4. Expected Result: A new row should be seen in the database with the given information.

5. Priority: Critical

---

1. Test Case ID: 10

2. Test Case Description: Check the response after adding a friend to a travel group

3. Flow of Events:

3.1 Go to the Profile Page

3.2 Select the travel group that you want to add a friend

3.3 Click “+” button and chose the friend to be added and send invitation

4. Expected Result: The chosen friend should be added to the travel group, database should be updated and a travel group notification should be sent to the chosen friend

5. Priority: Critical

---

1. Test Case ID: 11

2. Test Case Description: Check the response when a friend request is sent

3. Flow of Events:

3.1 Go to the Map Page

3.2 Click the “Friends” icon in the upper right corner

3.3 Click the “Plus” button at the bottom of the page

3.4 Search for a user name to which the request will be sent.

3.4 Click the “Send” button

4. Expected Result: The request should be seen in the same section (mentioned above) of the user so that the request is sent.

5. Priority: Critical

---

1. Test Case ID: 12

2. Test Case Description: Check the response when a friend request is sent to the user itself or to an already friend.

3. Flow of Events:

3.1 Go to the Map Page

3.2 Click the “Friends” icon in the upper right corner

3.3 Click the “Plus” button at the bottom of the page

3.4 Search for the username of a friend or yourself

3.4 Click the “Send” button

4. Expected Result: Sending friend request fails and an error message appears.

5. Priority: Major

---

1. Test Case ID: 13

2. Test Case Description: Check the response after arranging a trip.

3. Flow of Events:

3.1 Go to the Profile Page.

3.2 Select the travel group that you want to have a trip with.

3.3 Select among available dates of group members.

3.4 Select a place among ML suggestions that are given by the system or according to the poll results that are created by the other users.

3.5 Vote for suitable hotels and transportation opportunities for the given available dates and places.

4. Expected Result: The user should be navigated to the related sites for purchasing tickets or to reserve the determined accommodation.

5. Priority: Major

---

1. Test Case ID: 14

2. Test Case Description: Check the order of trip suggestions, suggested by the recommendation system, and check the response on Amadeus API calls

3. Flow of Events:

3.1 Go to the Profile Page.

3.2 Select the travel group that you want to have a trip with.

3.3 Select among available dates of group members.

3.4 ML suggestions should be listed

4. Expected Result: The ML suggestions should be formed by requesting the Amadeus API. First, common locations that are visited by the travel group should be used as a parameter to the recommendation algorithm, then the rest of the cities. The API should return suggestions, and it should be displayed to the user by their order of relevance.

5. Priority: Critical

---

1. Test Case ID: 15

2. Test Case Description: Check the database after arranging a trip.

3. Flow of Events:

3.1 Go to the Profile Page

3.2 Select the travel group that you want to have a trip with.

3.3 Select friends to send an invitation and press the “Send Invitation” button.

3.4 Select a place among ML suggestions that are given by the system or according to the poll results that are created by the other users.

3.5 Vote for suitable hotels and transportation opportunities for the given available dates and places.

4. Expected Result: A new trip should be seen in the database with the given information.

5. Priority: Critical

---

1. Test Case ID: 16

2. Test Case Description: Check the response when a photo is posted for a previous trip.

3. Flow of Events:

3.1 Go to the Map Page

3.2 Go to the “My Previous Trips” page by sliding the pages to the right.

3.3 Click the “View Map” button in the top of the page.

3.4 Click the trip location on the map that you want to add a photo.

3.5 Click the “Add Photo” button in the top of the page .

3.6 Click the “Allow” button for access permission to your gallery.

3.7 Select the photos that you want to post from your gallery.

3.4 Click the “Upload” button.

4. Expected Result: The photo should appear on the “My Previous Trips” page on the specific trip.

5. Priority: Major

---

1. Test Case ID: 17

2. Test Case Description: Check the response when a previous trip is rated.

3. Flow of Events:

3.1 Go to the Map Page

3.2 Go to the “My Previous Trips” page by sliding the pages to the right.

3.3 Click the “View Map” button in the top of the page.

3.4 Click the trip location on the map that you want to rate.

3.5 Click the “Rate” dropdown button and select from 1 to 5.

4. Expected Result: The overall rate of that place should be recalculated according to the given rate and displayed on the location itself.

5. Priority: Critical

---

1. Test Case ID: 18

2. Test Case Description: Check the database when the account is deleted.

3. Flow of Events:

3.1 Go to the Profile Page.

3.2 Click the “Edit ” icon in the top right corner.

3.3 Click the “Delete Account ” button on the bottom of the page.

4. Expected Result: Account should be deleted from database, along with all the personal data of the user (previous trips etc.)

5. Priority: Critical

---

1. Test Case ID: 19

2. Test Case Description: Check the response when a location is added to favorites.

3. Flow of Events:

3.1 Go to the Home Page or search the wanted location from the Home Page.

3.2 Click one of the locations.

3.3 Click the “Heart” icon on the top right corner of the page.

4. Expected Result: The location should be seen in the favorites page that can be reached from the drawer.

5. Priority: Minor

---

1. Test Case ID: 20

2. Test Case Description: Check the response when a location is selected.

3. Flow of Events:

3.1 Go to the Home Page or search the wanted location from the Home Page.

3.2 Click one of the locations.

4. Expected Result: The suggested activities and the city information should be seen on the bottom sheet.

5. Priority: Major

---

1. Test Case ID: 21

2. Test Case Description: Check the response when available dates are entered.

3. Flow of Events:

3.1 Go to Profile Page.

3.2 Select the available dates of travel from the calendar.

4. Expected Result: Calendar is updated according to the given dates.

5. Priority: Critical

---

1. Test Case ID: 22

2. Test Case Description: Check the response when the “Show Common Dates” button is clicked.

3. Flow of Events:

3.1 Go to Profile Page.

3.2 Select the travel group that you want to have a trip with.

3.4 Click the “Show Common Dates” button.

4. Expected Result: User should only see available common dates of all group members in the calendar.

5. Priority: Critical

---

1. Test Case ID: 23

2. Test Case Description: Check the response when a message is sent in group chat.

3. Flow of Events:

3.1 Go to Profile Page.

3.2 Select the travel group that you want to have a trip with.

3.3 Write the message you want to send.

4. Expected Result: The message should be seen in the chat box of the group.

5. Priority: Critical

---

1. Test Case ID: 24

2. Test Case Description: Check the response when a poll is created.

3. Flow of Events:

3.1 Go to Profile Page.

3.2 Select the travel group that you want to have a trip with.

3.3 Click the “Create Poll” button at the bottom of the page.

3.4 Add the questions and the choices for forming the poll.

3.5 Click the “Create” button.

4. Expected Result: The poll should be seen in the chat box and can be reached by all members.

5. Priority: Critical

1. Test Case ID: 25

2. Test Case Description: Check the response when a location is voted more than once while arranging a trip.

3. Flow of Events:

3.1 Go to the Profile Page.

3.2 Select the travel group that you want to have a trip with.

3.3 Select among available dates of group members.

3.4 Select a place among ML suggestions that are given by the system or according to the poll results that are created by the other users.

3.5 Vote for suitable hotels and transportation opportunities for the given available dates and places by selecting more than one opportunity.

4. Expected Result: Voting fails and an error message appears.

5. Priority: Critical

1. Test Case ID: 26

2. Test Case Description: Check whether search bar for places to visit returns relevant results

3. Flow of Events:

3.1 Enters a place name he/she wishes to search in the TextField at the top of the Home page.

3.2 Press the “Search” button.

4. Expected Result: If the input is valid, it should show a number of places that are relevant to the input.

If not, it should display an error message

5. Priority: Critical

1. Test Case ID: 27

2. Test Case Description: Check whether the info (including review and rating) of selected destination can be viewed

3. Flow of Events:

3.1 Searches and selects a place he/wishes to view in the main menu.

3.2 View Preferred season information of a destination.

3.3 Scrolls down and views the review and rating section.

4. Expected Result: For every destination, reviews and its rating should be up to date and synchronously retrieved from the database

5. Priority: Critical

---

1. Test Case ID: 28

2. Test Case Description: Check the home page for user specific route and destination suggestions

3. Flow of Events:

3.1 Switches to Home page.

4. Expected Result: The most suitable destination options specifically recommended to the user should be seen in the Home page.

5. Priority: Major

---

1. Test Case ID: 29

2. Test Case Description: Check whether there are activity suggestions for a destination

5. Flow of Events:

3.1 Switch to Home page.

3.2 Select a destination.

3.2 Views the destination page.

4. Expected Result: The popular activities that can be done in that destination should be seen.

5. Priority: Minor

---

1. Test Case ID: 30

2. Test Case Description: Check if upcoming trips is visible

5. Flow of Events:

3.1 Switch to the Trips page, after creating a trip on the app.

4. Expected Result: Upcoming trips that are planned by their travel groups should be visible on the screen.

5. Priority: Critical

---

1. Test Case ID: 31

2. Test Case Description: Check whether the previous trips of yourself is visible or not

5. Flow of Events:

3.1 Switch to the Maps page.

4. Expected Result: A world map with the previous trips of the user should be seen directly.

5. Priority: Major

---

1. Test Case ID: 32

2. Test Case Description: Check whether the previous trips of your friends is visible or not in the World map

5. Flow of Events:

3.1 Switch to the Maps page.

3.2 View the friend list that is found in the Maps page.

3.3 Click to the friend's name that has their own map.

3.4 Sees the related info (photo,review) of their friends.

4. Expected Result: A world map with the previous trips of the user’s friends should be seen.

5. Priority: Major

---

1. Test Case ID: 33

2. Test Case Description: Get a bearer token for authorization

5. Flow of Events:

3.1 Make a post request to “/authentication/token” endpoint with the body {username, password}

3.2 Try accessing the list of trips of your own.

3.3 Try accessing the list of trips of another user.

4. Expected Result: Get a valid bearer token that allows the user to access user’s own data, but not others.

5. Priority: Critical

---

1. Test Case ID: 34

2. Test Case Description: Response to wrong authentication requests

5. Flow of Events:

3.1 Make a post request to “/authentication/token” endpoint with wrong information the body {username, password}

4. Expected Result: The endpoint should not give a bearer token for wrong credentials. There should be an error.

5. Priority: Critical

---

1. Test Case ID: 35

2. Test Case Description: Check response when a poll receives a vote

5. Flow of Events:

3.1 Navigate to a group chat.

3.2 Click on a poll.

3.3 Vote one of the options in the poll by clicking on it.

3.4 View the results of the poll.

4. Expected Result: The poll results should update according to the new vote received.

5. Priority: Critical

---

1. Test Case ID: 36

2. Test Case Description: Check response when a location is added to a trip

5. Flow of Events:

3.1 Navigate to a trip.

3.2 Click on edit button.

3.3 Search for a new location by name.

3.4 Choose the location to add to the trip.

4. Expected Result: The location can be seen among the list of locations in the trip and is marked on the map.

5. Priority: Critical

---

1. Test Case ID: 37

2. Test Case Description: Check whether password is encrypted when a user signs up.

5. Flow of Events:

3.1 Navigate to Sign Up page.

3.2 Choose the sign up with email option.

3.3 Fill in the account information.

3.4 Click sign up button.

4. Expected Result: The newly registered user’s password should be encrypted in the database.

5. Priority: Critical

---

1. Test Case ID: 38

2. Test Case Description: Check if the user gets a notification when a new message is sent to a group chat.

5. Flow of Events:

3.1 The user has the Travela app installed and is logged in.

3.2 A new message is sent to a group chat that the user is a member of.

4. Expected Result: A push notification is sent to the user's device.

5. Priority: Critical

---

1. Test Case ID: 39

2. Test Case Description: Check the response when a new transportation method is added to a trip.

5. Flow of Events:

3.1 Navigate to a trip.

3.2 Click on the edit button.

3.3 Click on add transportation button.

3.4 Select the departure and arrival locations.

3.5 Select type of transport.

3.6 Select the transportation method from available options.

4. Expected Result: The transportation method between selected locations should be visible among trip information and should be added to the map between the two locations.

5. Priority: Major

---

1. Test Case ID: 40

2. Test Case Description: Check the response when a new accommodation method is added to a trip location.

5. Flow of Events:

3.1 Navigate to a trip.

3.2 Click on edit button.

3.3 Select a location to add accommodation to.

3.6 Select the accommodation method from available options.

4. Expected Result: The accommodation method between locations should be visible among trip information.

5. Priority: Major

---

1. Test Case ID: 41

2. Test Case Description: Check whether old messages are still available after a user leaves a group.

5. Flow of Events:

3.1 Navigate to a group.

3.2 Click on the leave group button.

4. Expected Result: The past messages from the group should still be available, but the messages sent to the group after they leave should not be sent to the user.

5. Priority: Critical

---

1. Test Case ID: 42

2. Test Case Description: Check the total cost of the trip.

5. Flow of Events:

3.1 Navigate to a trip.

3.2 Click on the trip to view more information.

3.3 View the total cost of the trip.

4. Expected Result: The cost of the trip should be equal to the sum of all the costs of accommodation and transportation.

5. Priority: Minor

---

1. Test Case ID: 43

2. Test Case Description: Check whether previous trips are updated correctly.

5. Flow of Events:

3.1 Navigate to a trip.

3.2 Mark the trip as completed.

3.3 Navigate to the previous trips page.

4. Expected Result: The completed trip should be removed from the current trips page and be added to the previous trips page.

5. Priority: Critical

---

1. Test Case ID: 44

2. Test Case Description: Check response to password change

3. Flow of Events:

3.1 Go to the Profile Page.

3.2 Click change password.

3.3 Enter current and new password.

3.4 Confirm password change.

3.5 Log out.

3.6 Try to login with the old password.

3.7 Try to login with the new password.

4. Expected Result: New password should be encrypted in the database and replace the old password. When the old password is used, login should be unsuccessful. When the new password is used, the user should be able to login successfully.

5. Priority: Critical

---

1. Test Case ID: 45

2. Test Case Description: Check response to trip deletion

3. Flow of Events:

3.1 Navigate to a trip.

3.2 Click delete trip button.

3.3 Refresh trips page.

4. Expected Result: Trip should be removed from the group and/or the user's profile, along with all the locations, transportation methods and accommodation options. Finally, the trip itself should be removed from the database.

5. Priority: Critical

---

1. Test Case ID: 46

2. Test Case Description: Check response to group deletion

3. Flow of Events:

3.1 Navigate to a group.

3.2 Click delete group button.

3.3 Refresh groups page.

4. Expected Result: All members of the group should be removed from the group. All trips associated with that group should be deleted. All messages in the group chat should be deleted. Finally, the group itself should be removed from the database.

5. Priority: Critical

---

1. Test Case ID: 47

2. Test Case Description: Check the response of correct login credentials

3. Flow of Events:

3.1 Enter the email

3.2 Enter the password

3.3 Click the “Sign In” button

4. Expected Result: If the login credentials are correct, the user should be navigated to the home page.

5. Priority: Critical

4.2 Non-Functional Test Cases

1. Test Case ID: 48

2. Test Case Description: Check the response of the navigation bar.

3. Flow of Events:

3.1 Go to the Home Page.

3.2 Click one of the icons on the navigation bar.

4. Expected Result: User should be navigated to the related page.

5. Priority: Critical

---

1. Test Case ID: 49

2. Test Case Description: Check the positioning of the drawer widget.

3. Flow of Events:

3.1 Go to the Home Page or Map Page.

3.2 Click the “Drawer” icon in the upper left corner.

4. Expected Result: The navigation bar should not be seen with the drawer.

5. Priority: Major

---

1. Test Case ID: 50

2. Test Case Description: Check the map in trip view

3. Flow of Events:

3.1 Navigate to a trip.

3.2 View the trip in map format.

4. Expected Result: The trip should appear as a map, where the locations and transportations between them are shown and further information about a location can be viewed by clicking on it.

5. Priority: Major

---

1. Test Case ID: 51

2. Test Case Description: Check that the messages in group chats are in correct order.

3. Flow of Events:

3.1 Navigate to a group chat.

3.2 Send several messages to the group chat.

3.3 Receive several messages from other group members.

4. Expected Result: The messages should appear sorted based on time, with the most recent message appearing at the bottom.

5. Priority: Critical

4.3 Performance Related Test Cases

1. Test Case ID: 52

2. Test Case Description: Check whether the login operation takes less than 2 seconds

3. Flow of Events:

3.1 Enter the email

3.2 Enter the password

3.3 Click the “Sign In” button

4. Expected Result: The login operation which involves backend and database communication should take less than 2 seconds, and output the correct response.

5. Priority: Minor

---

1. Test Case ID: 53

2. Test Case Description: Check whether the suggest trip functionality takes less than 8 seconds

3. Flow of Events:

3.1 Go to the Profile Page.

3.2 Select the travel group that you want to have a trip with.

3.3 Select among available dates of group members.

3.4 ML suggestions should be listed

4. Expected Result: Recommending trip locations which involve backend and database communication as well as external API calls should take less than 8 seconds, and output the correct response.

5. Priority: Minor

---

1. Test Case ID: 54

2. Test Case Description: Check the response during fetching data.

3. Flow of Events:

3.1 Go to the Maps page.

4. Expected Result: See the loading icon on the page until the data is fetched from the server, which should not take more than 5 seconds.

5. Priority: Minor

---

5 Other Analysis Elements

5.1 Consideration of Various Factors in Engineering Design

- Aesthetics

Aesthetics of a product is one of the most important things as before looking at the specifications of a product, the first thing people notice is the way the app looks. To ensure a good aesthetic feel in the first look, we will try to follow the design trends in the industry and create a modern interface.

- Maintenance

To develop a sustainable and continuous product, responsive design is a crucial part as Travela will be available for many devices and platforms with different sizes. Maintaining different designs for all different platforms is not a good idea for effectiveness so our goal will be to create limited different designs when needed and responsive for all devices otherwise.

- Social Factors

Any person from any gender, age, race, etc can use Travela both individually or in groups. Thus, we can say that there are no social factors directly affecting our app.

5.2 Risks and Alternatives

Like every new business idea, Travela has some risks too. Firstly, it proposes a new social media, which may have a hard time finding its customer base amongst the giant media companies like Instagram or Meta (Facebook). And one important consideration is that it is vital to provide good trip recommendation service when users sign into the app for the first time, as most of the users would download the app to plan a trip immediately. Thus, it is a challenge to crawl as much data about the users holiday interests as possible in a short amount of time and suggest possible vacation alternatives. Also, the success of recommender systems depend on third party applications and their integration with Travela such as

- Google hotels

- Google map reviews
- Skyscanner
- Amadeus
- Travelport

Due to privacy concerns of these big companies, the integration to them can be challenging. But high risk comes with great reward, the more data that Travela crawls from third party applications, the better all around trip recommendation service it provides. Alternatively, integration with other third party applications like TripAdvisor, Airbnb can be established, if the expected success cannot be achieved with the third party applications that are listed above.

6 Teamwork Details

6.1 Ensuring Proper Teamwork

Teamwork is one of the most important factors that lead to success in such projects. Thus, it is important to utilize as much teamwork as possible, for the best outcomes. There are several ways of ensuring this, some examples can be seen below.

- Dividing work equally that matches the interests and specialities of the team members, in face to face or online meetings
- Scheduling and having meetings weekly
- Setting weekly/monthly goals about the project, foreseeing how much work has to be done by each individual and planning forward accordingly
- Using Google docs for enabling simultaneous work on reports
- Using Git and GitHub for the version control and enabling simultaneous work on the code
- Using a kanban board (maybe Jira) to keep track of the weekly goals and sprints

- Using communication channels such as Whatsapp efficiently, to daily update the work that has been done, or the progresses that have been made, or which tasks need more manpower to finish
- Communicating regularly

6.2 Contributing and functioning effectively on the team

Efe Ertürk contributed to the backend part of the project, as well as documentation. He also worked on ML solutions for the recommendation problem.

Efe Şaman worked on the backend of the project, mainly focusing on the authentication and authorization features. Took part in designing software architecture as well.

Çağla Ataoğlu is on the backend team as well. She mostly focused on part of the controller, service and entity code. She contributed to the UML diagrams during the design process.

Yağmur Eryılmaz is on the frontend team. She contributed to the UI design and implementation. She worked on the connection between backend and frontend. Also contributed to the documentation and UML.

Cenk Duran contributed to the frontend part of the project. He mainly focused on the implementation of the UI and also the API integrations of the project.

6.3 Helping creating a collaborative and inclusive environment

To help create a collaborative and inclusive environment, Efe Ertürk used communication channels like whatsapp regularly and efficiently to keep up with the overall condition of other teammates and also to be involved in the project decisions. He also helps schedule weekly meetings.

Efe Şaman played an active role on Whatsapp and other communication channels to create an inclusive and friendly environment. Worked on building a casual, collaborative relation with his peers. Focused on the relation with his peers as much as the code.

To help create an environment that values everyone's opinions and preferences, Çağla Ataoğlu took part in communicating with the group through their Whatsapp group, online meetings and in-person meetings. She listened and valued others' opinions and helped distribute workload and tasks.

Yağmur Eryılmaz had an active role on Whatsapp and other communication channels and also in person meetings to create an inclusive environment. Also she expressed her opinions clearly and contributed to discussions and helped creating a collaborative environment via focusing on the relation with her teammates.

Cenk Duran played an active role on our communication channels, in the group meetings both online and face to face. He described how he wanted to implement directly with the team and worked according to the group's decision all the time.

6.4 Taking the lead role and sharing leadership on the team

As a team, every individual had different roles for different tasks, and the lead role was shared amongst the team members and switched frequently.

Efe Ertürk took the leading role at backend artifact of the project, as well as the ML part of the project, due to his prior experiences in these domains.

Efe Şaman is responsible for the authorization and authentication part of the project, and took a lead role in building it on the backend. Moreover, at times he took a leading role in designing the backend.

Çağla Ataoğlu assumed a leadership role in the backend team in some instances. On some occasions, she led meetings and other collaborative work, such as writing documentation.

Yağmur Eryılmaz took a leading role in UI design and determination of features. She also communicated with the supervisor and informed him about the advancements.

Cenk Duran took the leading role at implementing the frontend artifact because of his previous experiences with flutter. He used his experience in both mobile and web platforms.

7 References

[1] Knight, Rob. “Average person spends 10 hours planning their holiday, survey claims”. Independent UK. https://www.independent.co.uk/travel/holiday-booking-planning-travel-survey-tourist-a8801211.html. (Accessed: 15 Oct 2022)

[2] Lamothe , Marta. “Airbnb’s 100+ new app features: Everything you need to know”. Your Rentals. https://your.rentals/blog/airbnb-2021-new-app-features-everything-you-need-to-know/. (Accessed: 23 Oct 2022)

[3] A. Inc., “Human interface guidelines,” *Human Interface Guidelines - Human Interface Guidelines - Design - Apple Developer*. [Online]. Available: https://developer.apple.com/design/human-interface-guidelines/guidelines/overview/. (Accessed: 17 Oct 2022).

[4] “Material design for Android : android developers,” *Android Developers*. [Online] Available:https://developer.android.com/develop/ui/views/theming/look-and-feel. (Accessed: 17 Oct 2022).

[5] “What is token-based authentication?,” *Okta*. [Online]. Available: https://www.okta.com/identity-101/what-is-token-based-authentication/. (Accessed: 17 Oct 2022).