Bilkent University

Department of Computer Engineering

Senior Design Project

T2312

Funravel

Analysis and Requirement Report

Fatma Sena Genç , 21901426 , sena.genc@ug.bilkent.edu.tr
Gülin Yılmaz , 21903057 , gulin.yilmaz@ug.bilkent.edu.tr
Hissam Mahmoud Elsayed Faramawy , 21901253, hissam.faramawy@ug.bilkent.edu.tr
Mustafa Efe Tamyapar , 21902856 , efe.tamyapar@ug.bilkent.edu.tr
Öykü Erhan , 21901541 , oyku.erhan@ug.bilkent.edu.tr

Supervisor: Fazlı Can

Course Instructors: Erhan Dolak, Tağmaç Topal

Final Reader: Öykü Erhan

November 13, 2022

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

Contents

1 Introduction 4

2 Current System 5

3 Proposed System 6

3.1 Overview 6

3.1.1 Suggesting a Travel Plan for Concerts 6

3.1.2 Creating Review Posts 7

3.1.3 Group chats and Invitations 7

3.4 Pseudo Requirements 9

3.5 System Models 10

3.5.1 Scenarios 10

3.5.2 Use-Case Model 18

3.5.3 Object and Class Model 19

3.5.3.1 Class Diagram 19

3.5.3.2 Explanation of the Class Diagram 21

3.5.4 Dynamic Models 22

3.5.4.1 Sequence Diagram 22

3.5.4.1.1 Sequence Diagram for Creating Invitations 22

3.5.4.1.2 Sequence Diagram for Creating Travel Plans 23

3.5.4.1 Activity Diagram 24

3.5.4.1 State Diagram 28

3.5.5 User Interface 29

3.5.5.1 Navigational Path 29

3.5.5.2 Login screen 30

3.5.5.3 Register screen 31

3.5.5.4 Home screen 32

3.5.5.5 Filter screen 33

3.5.5.6 Concert screen 35

3.5.5.7 Create Group Invitation screen 36

3.5.5.8 Suggest Travel Plan Screen 37

3.5.5.9 Suggested Plan Screen 38

3.5.5.10 Social screen 39

3.5.5.11 Groups Chats screen 40
3.5.5.12 Profile screen 43
3.5.5.13 Settings screen 44
3.5.5.14 Edit Profile screen 45
3.5.5.15 Your Past Reviews screen 46
3.5.5.16 Your Group Invitations screen 47
3.5.5.17 Create Review Post screen 48
3.5.5.18 Saved Plans screen 49
4 Other Analysis Elements 50
4.1 Consideration of Various Factors in Engineering Design 50
4.2 Risks and Alternatives 52
4.3 Project Plan 54
4.4 Ensuring Proper Teamwork 74
4.5 Ethics and Professional Responsibilities 74
4.6 Planning for New Knowledge and Learning Strategies 74
5 Glossary 75
6 References 76

Analysis and Requirement Report

Project Short-Name: Funravel

1 Introduction

Planning a trip for an event outside your city can be time-consuming and challenging. According to Ticketmaster's database, passionate fans, which is around 10% of an artist’s fans, travel 100+ miles for a concert and even some of them cross the Atlantic [1]. Target users of this application are the type of fans who are willing to travel far away and make a trip. The average person plans their trip in 10 hours as surveys claim [2]. Funravel wants to optimize this planning time spent, by suggesting to users a travel plan for an event and calculating the total cost of their travel, including accommodation and transportation. Essentially, Funravel is an event based travel optimization mobile application. Funravel also aims to connect users with each other who do not want to attend these events all by themselves. Overall, it wants to reduce time spent on the planning of far-away events as much as possible with the possibility of group planning.

The innovation that is planned to implement is an offering type of innovation. The main goal of this application is to assist fans who want to attend a concert far away. In this offering, combining multiple platforms to ease the planning for an event trip is intended. The innovation type of Funravel is incremental innovation because Funravel adds new functionalities to the services which are currently in the market.

In this report, current solutions for travel planning are discussed in the current system section. Description of the system, requirements, technical constraints and some system models are discussed under the proposed system section. Lastly, the project development plan, our strategy for proper teamwork, various ethical and professional responsibilities and our learning strategies are discussed under the other analysis section of the report.

2 Current System

Our domain is travel planning but specific to concerts that are far away from the location where the user lives. The problem is that planning for a concert that is far away is time-consuming and challenging. We aim to optimize the time spent on planning. There are competitors in the market with similar approaches, but their approach is not as specific as ours. For the travel planning aspect, there are Booking.com, Expedia, and Skyscanner. These applications help users in the aspect of planning their trips. In the concert aspect, Ticketmaster is an application that can be used as an example for finding concert tickets. Our approach combines both aspects and allows us to offer a user-friendly, gamified interaction that is more user-friendly and engaging while planning a trip. On booking.com or Expedia, users can manually select components of travel packages, such as transport and accommodations. But in our application, we create a package if the user wants to change a thing in the package using the change button. As a result, the packages are changed in a more gamified way. Also in this regard, there is Waynabox. This generates randomized travel plans for the users based just on their starting location and preferences, such as duration of stay and hotel rate. As there are similarities with the application but again it is not specific.

From the overall perspective, we are combining multiple applications that are currently available on the market and making them user-friendly and gamified for the user. In this way, the application optimizes the time spent on travel planning. Also as our system has group chat and review features, overall the experience of planning will be less stressful for the user. It is possible for users to find concert friends if they do not wish to participate alone. So that the whole experience will be more pleasant to the user.

3 Proposed System

Details of the results of analysis of the project are given in this section.

3.1 Overview

The main goal of Funravel is to be a travel assistant for fans who want to attend an event that can be distant. Concerts are the main focus of the Funravel app. Using the user's preferences Funravel finds the most optimal travel package for the user. Other functionalities also aim to help users with planning trips for events, creating posts, and finding groups to attend with. Users can connect with each other before the event and plan their trip together using group chats. Moreover, users can share posts that include reviews about their experience. In the application, each user has a profile page. On this page, posts and invitations generated by the user, profile picture, and the top 3 singers of the user are shown.

3.1.1 Suggesting a Travel Plan for Concerts

On the homepage of the application, there is a search bar where users can search for the event. While searching users can filter the search for a city and country. After finding the event, the user can directly go to the ticket-buying website using the redirection button. Also, the user can select a package. After selecting a package, the user selects the desired period of stay at the location, hotel preference, starting location, and desired transportation type. The hotel preference is the hotel rating and the desired transportation type includes bus and plane. After collecting all this information, Funravel finds the most optimized package for the user in terms of budget and distance between location of the event and hotel. The total calculated cost and redirection buttons for the ticket, and booking sites are shown to the user. After creating a package, if the user wishes to see the package later, the user can save the package using the save button. The saved packages with calculated costs are shown in the saved events page.

3.1.2 Creating Review Posts

Funravel helps fans who want to share their experience with other people after the concert. Users can create posts on the profile page, about the concert they attended. These posts will be shown on the page of each event. They include text and a rate from 10. Using these, users can share the kind of experience they had in the concert. This way, the users can express their feelings using the application and they will be encouraged to use the application more.

3.1.3 Group chats and Invitations

Users can share a group invitation for connecting with others who want to attend the same event. These invitations can be seen through a separate invitations page in the application. The explanation written by the user and information about the concert is shown in each of the group invitations. The user can click the join group chat button by using these group invitations. In the group chat, users can communicate and plan a concert event together. Each user can see the group chat they have joined on the group chats page.

3.2 Functional Requirements

- The system should be able to suggest generated travel plans for an event after receiving the preferred dates, star rate of hotel, start point and transportation type from the user.
- The system should be able to show the total cost and redirect the user to the related booking and means of transport reservation website (Booking.com, TripAdvisor etc.) for the travel plan of the concert event.
- The system should display the type of the currency according to the users' preference.
- The user should be able to create their accounts with their email and password.
- The user should be able to filter the accommodations in which they want to stay according to the star rating of that place.
- The user should be able to select the date range for the travel they want to go to.
- The user should be able to see the upcoming events on the home page.

- The user should be able to filter the city and country of the upcoming events in order to choose a specific location.
- The user should be able to see details and shared posts related to the events.
- The user should be able to create a review post to write their comment on a specific event.
- The user should be able to create group invites in order to connect with other users. The system will automatically create a group chat regarding that invite.
- The user should be able to see the group invitations other users created and join their chat rooms in order to find a group to attend an event with.
- The user should be able to see their own group invitations and review posts on their profile page.
- The user should be able to see the group invites they attended at their group chats page.
- The user should be able to leave the chat rooms when they wish to do so.
- The user should be able to change each of the suggested travel plan elements (hotel and transportation suggestion) by requesting the next best option.

3.3 Non-functional Requirements

User friendliness / Usability: The user interface is intended to be as convenient but uncomplicated as possible so that the users will be able to adapt to it easily. There will be use of components that the majority of internet users are accustomed to. Additionally, a user interface manual will be provided.

Maintainability: The system should be capable of being modified to fix the fault or improve the performance in a cost-effective way. Some technical approaches will be used such as high cohesion, loose coupling and using standard API formats and clear document interfaces. In that way, it will be easy to track the code and see where the errors might be occurring. Also, Flutter’s built-in tools will be used to catch errors and write test cases.

Flexibility: The system should be open to changes and adding new features since the market and demands of the customers rapidly change. The software should be easily modified to adapt to different environments and user expectations.

Scalability: The system should be able to handle expansion and adapt its resources to changing demands. Additionally, system architecture should be developed in such a way that it enables for easy change and component upgrade/downgrade.

Portability: Flutter is a cross-platform SDK for mobile applications. So, the system should be working on different mobile platforms such as IOS and Android simultaneously with small time and effort.

3.4 Pseudo Requirements

- The platforms of the project are planned to be both Android and IOS.
- Flutter will be used as a framework of the application.
- In order to work simultaneously, Git and Github will be used as a version control system.
- Firebase (NoSQL database) and MySQL databases will be used to store and handle the data.
- Dart will be used as a programming language.
- Ticketmaster API, Flight Data API, Flixbus API, Google Hotel Prices APIs and other similar travel APIs will be used to collect information, gathered from rapid API. [3]
- Both Android and IOS parts of the project will support English and Turkish.

3.5 System Models

3.5.1 Scenarios

Use Case #1

Use Case Name: Search for artist

Participating Actor: User

Flow of Events:

1. Users can search for the artist’s name they want to see the events of, on the “Concert page.

Entry Condition: User navigates to concert page.

Exit Condition: Concerts will be shown based on searched artist.

Use Case #2

Use Case Name: Apply filters on events

Participating Actor: User

Flow of Events:

1. Users can click the filter icon next to “all results” on the “Concert” page.
2. User chooses and/or a city and country and clicks on “apply filter”.

Entry Condition: User should be logged in to the system and navigated to the “Concert” page.

Exit Condition: Concerts will be shown based on the filtering.

Use Case #3

Use Case Name: See details of events

Participating Actor: User

Flow of Events:

1. User clicks on any concert on the “Concert” page
2. User sees the location and time of the concert

Entry Condition: User should be logged in to the system and navigated to the “Concert” page.

Exit Condition: User leaves the “Concert” page.

Use Case #4

Use Case Name: Go to event ticket purchase website

Participating Actor: User

Flow of Events:

1. User clicks on the “buy tickets” button on the concert detail page.
2. The system redirects the user to the official ticket purchase website.

Entry Condition: User should be logged in to the system and navigated to the “Concert” page.

Exit Condition: User leaves the “Concert” page.

Use Case #5

Use Case Name: Update profile information

Participating Actor: User

Flow of Events:

1. User clicks the “Profile” icon and navigates to the “Profile” page.
2. User clicks on the “edit” icon on the “Profile” page.
3. Users can change their displayed name, user name, and biography.

Entry Condition: User should be logged in to the system, navigated to “Edit Profile” page.

Exit Condition: User leaves the “Edit Profile” page.

Use Case #6

Use Case Name: Ask for a travel plan

Participating Actor: User

Flow of Events:

1. User goes to concert detail page for a specific concert
2. User should indicate their location.
3. User should indicate their preferred transportation type.
4. User should indicate their preferred star rating for accommodation.
5. User should indicate their preferred date range.
6. User can also add additional optional filters. (they are explained further in the next scenario.)

Entry Condition: User navigates to the “Concert” page and selects their preferred filters.

Exit Condition: User clicks on make package or leaves “Concert” page.

Use Case #7

Use Case Name: Select options

Participating Actor: User

Flow of Events:

1. User goes to “Concert” page for a specific concert.

2. User can choose their preferences among the options:

- apply minimum transfer,
- apply best price,
- apply shortest distance to event location,
- apply the shortest travel time.

Entry Condition: User navigates to the “Concert” page and selects their preferred optional filters.

Exit Condition: User clicks on “make package” or leaves concert detail page.

Use Case #8

Use Case Name: Request for another best

Participating Actor: User

Flow of Events:

1. User goes to “Concert” for a specific concert

2. User asks for a travel plan with their preferences.

3. User clicks the change button on the travel suggestions.

4. New suggestions for hotels or transportation are created and displayed.

Entry Condition: User should be logged in to the system and clicks on the “make package” button on the plan page.

Exit Condition: New suggestion is created and displayed.

Use Case #9

Use Case Name: See travel plan information

Participating Actor: User

Flow of Events:

1. User goes to suggest travel plan page, enters necessary information and clicks on “make package”.

2. User can view their travel information which includes:

- Hotel information
- Transportation information
- Total cost
- Reservation links

Entry Condition: User should be logged in to the system and clicks on the “make package” button on the plan page.

Exit Condition: User clicks back or saves the package with “save this package” button.

Use Case #10

Use Case Name: See upcoming events

Participating Actor: User

Flow of Events:

1. User clicks the “Home” icon and can see all upcoming events from the home page.

Entry Condition: User should be logged in to the system, navigated to the “Home” page.

Use Case #11

Use Case Name: Choose a type for currency

Participating Actor: User

Flow of Events:

1. Users can click the “Settings” icon within the “Profile” page.

2. Then, users can change the language to English or Turkish.

Entry Condition: User should be logged in to the system, navigate to “Settings” page.

Use Case #12

Use Case Name: Leave group chats

Participating Actor: User

Flow of Events:

1. User clicks the “Group Chat” icon.
2. Users click to the group chat they want to leave.
3. Then the user clicks the information icon and a pop up page for group chat information is seen.
4. Users can leave the group chat by clicking the “leave chat” button.

Entry Condition: User should be logged in to the system, navigated to “Group Chat” page.

Exit Condition: When a user is not part of any group chats.

Use Case #13

Use Case Name: Create group invitations

Participating Actor: User

Flow of Events:

1. User clicks the “Home” icon.
2. User selects the concert they wish to create a group chat and navigates to the “Concert” page.
3. User clicks the “Create Group Invitation” button and navigates to the “Create Group Invitation” page.
4. User sets a title and writes an explanation accordingly and clicks the “Create” button to generate a group chat for the concert they selected.

Entry Condition: User should be logged in to the system, navigated to the “Concert” page.

Exit Condition: Group chat is created automatically for that group invitation.

Use Case #14

Use Case Name: Filter group invitations

Participating Actor: User

Flow of Events:

1. User clicks the “Social” icon and navigates to the “Social” page.
2. Users can see all the group chat invitations and filter them by artist, city and country name.

Entry Condition: User can see all the group chat invitations, navigated to the “Social” page.

Exit Condition: Group invitations are seen according to the filters chosen.

Use Case #15

Use Case Name: Join group chats

Participating Actor: User

Flow of Events:

1. User clicks the “Social” icon and navigates to the “Social” page.
2. Users can see all the group chat invitations and join them by clicking the “Join Group” button.

Entry Condition: User should be logged in to the system, navigated to “Social” page.

Exit Condition: Group chat will appear in the user's group chats page.

Use Case #16

Use Case Name: Send message within the group chat

Participating Actor: User

Flow of Events:

1. User clicks the “Group Chat” icon.
2. Then, the user clicks to the group chat that they want to send a message.
3. After typing the message, the user clicks the send button to send the message.

Entry Condition: Users should be logged in and join any group chat in order to send messages.

Exit Condition: Message is sent to the group for every other member to see.

Use Case #17

Use Case Name: See own group invitations

Participating Actor: User

Flow of Events:

1. User clicks the Profile page icon.

2. User clicks to "Your Group Invitations" button and navigates to "Your Group Invitations" page.

Entry Condition: User should be logged in to the system, navigated to "Your Group Invitations" page.

Exit Condition: User clicks other pages icons.

Use Case #18

Use Case Name: See reviews for event

Participating Actor: User

Flow of Events:

1. User clicks the "Home" icon.

2. User selects the concert they like and navigates to the "Concert" page.

3. Under the concert information, users can see the reviews for that artist's past concerts.

Entry Condition: User should be logged in to the system, navigated to the "Concert" page.

Exit Condition: User clicks other pages icons.

Use Case #19

Use Case Name: Review events

Participating Actor: User

Flow of Events:

1. User clicks the "Profile" page icon.

2. User clicks to create a review button.

3. Users select artists, share and rate their experience and post the review by clicking share your review button.

Entry Condition: User should be logged in to the system, navigated to the “Concert” page.

Exit Condition: Review will appear in user’s past reviews and under the related artist’s concert page.

Use Case #20

Use Case Name: See own reviews

Participating Actor: User

Flow of Events:

1. User clicks the Profile page icon.
2. User clicks to “Your Posts” button.

Entry Condition: User should be logged in to the system, navigated to “Your Posts” page.

Exit Condition: User clicks back or other pages icons.

Use Case #21

Use Case Name: Delete review

Participating Actor: User

Flow of Events:

1. User navigates to the “Profile” page.
2. User clicks to “Your Posts” button and navigates to “Your Group Invitation” page.
3. User clicks the “Delete” button on the review they want to delete.

Entry Condition: User should be logged in to the system, navigated to “Profile” page.

Exit Condition: Review post will be deleted from the system and removed from user’s reviews.

3.5.2 Use-Case Model

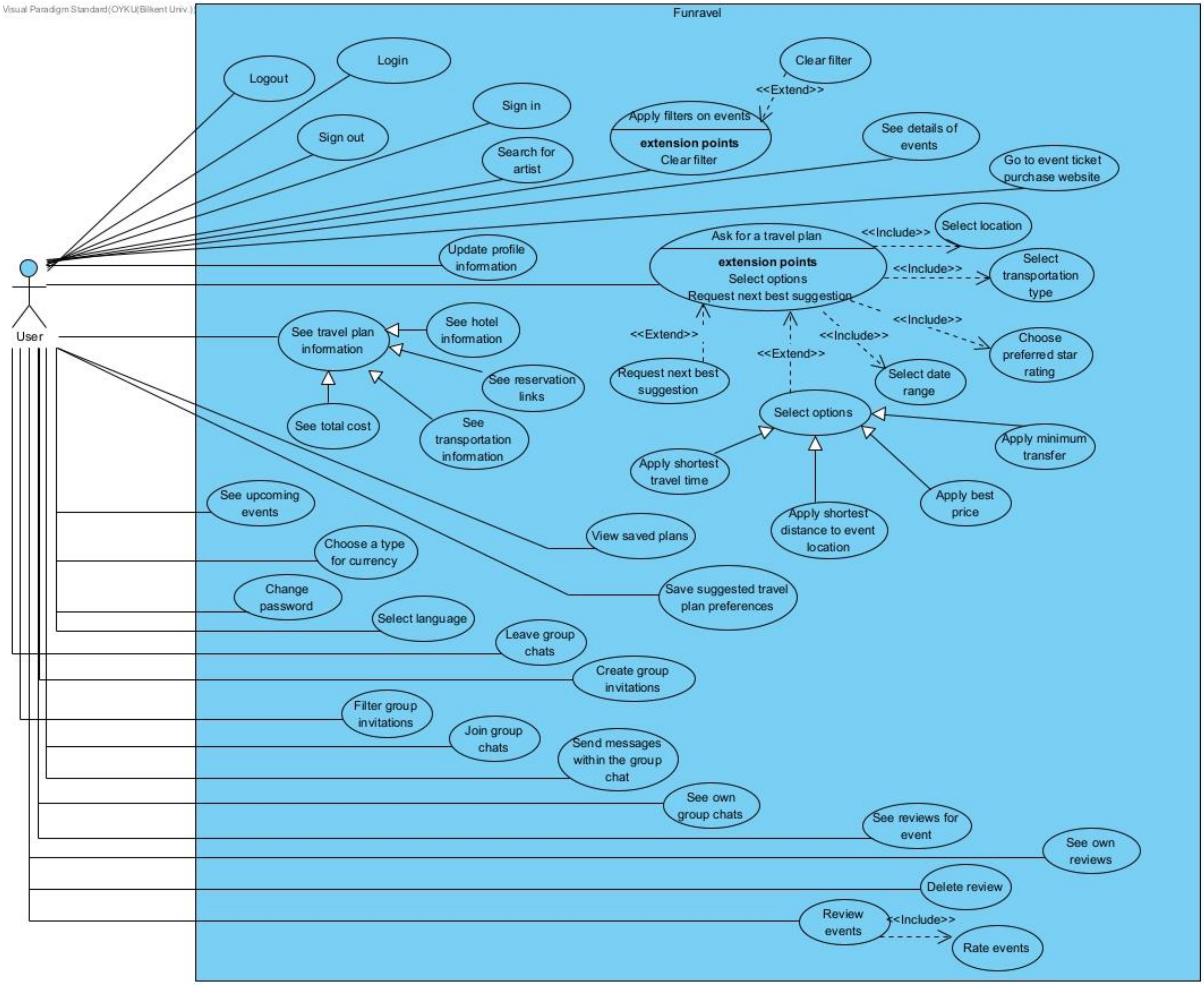
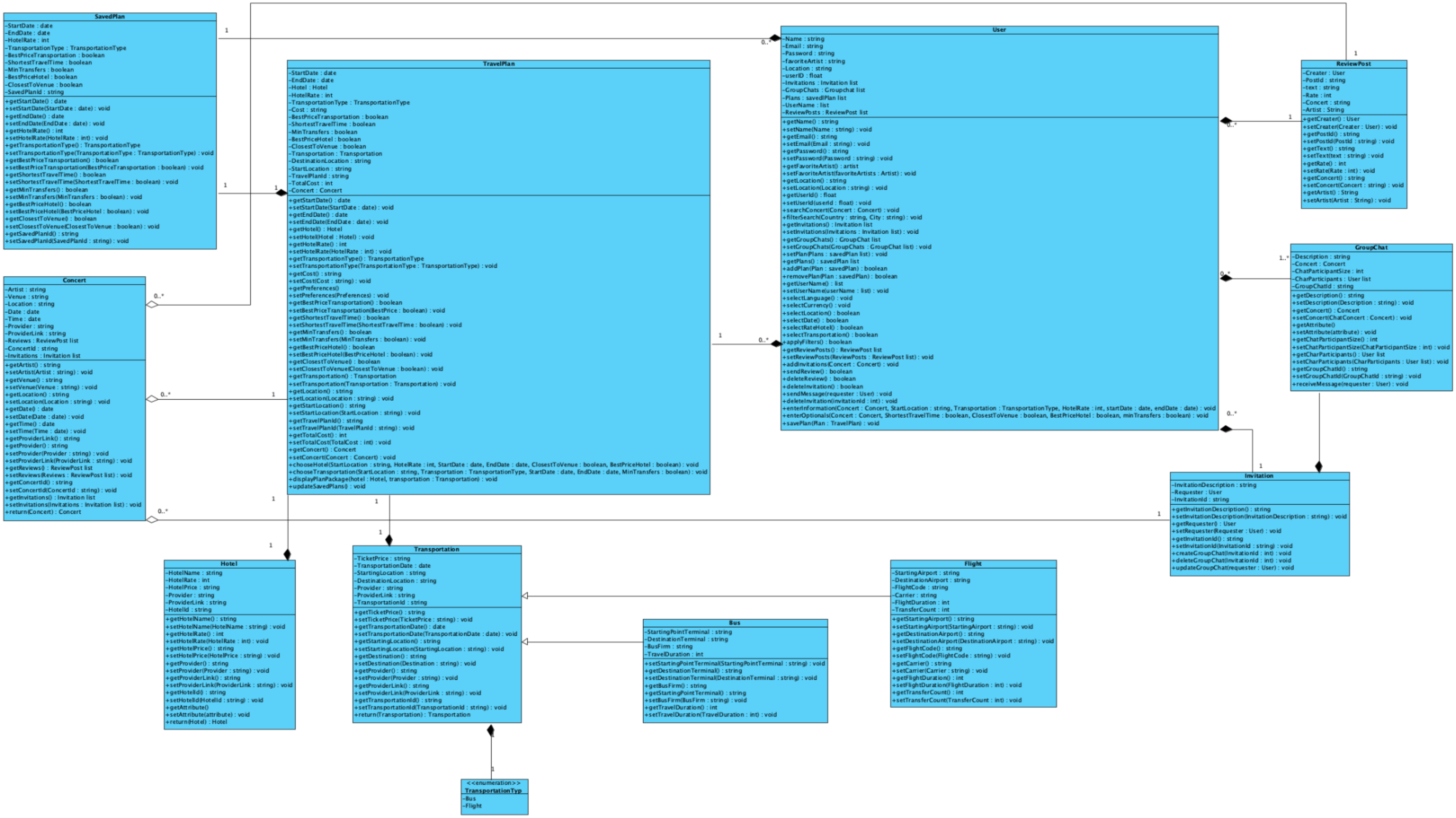


Figure 1. Use case model.

3.5.3 Object and Class Model

3.5.3.1 Class Diagram

Figure 2. Class Diagram


SavedPlan
-StartDate : date
-EndDate : date
-HotelRate : int
-TransportationType : TransportationType
-BestPriceTransportation : boolean
-ShortestTravelTime : boolean
-MinTransfers : boolean
-BestPriceHotel : boolean
-ClosestToVenue : boolean
-SavedPlanId : string
+getStartDate() : date
+setStartDate(StartDate : date) : void
+getEndDate() : date
+setEndDate(EndDate : date) : void
+getHotelRate() : int
+setHotelRate(HotelRate : int) : void
+getTransportationType() : TransportationType
+setTransportationType(TransportationType : TransportationType) : void
+getBestPriceTransportation() : boolean
+setBestPriceTransportation(BestPriceTransportation : boolean) : void
+getShortestTravelTime() : boolean
+setShortestTravelTime(ShortestTravelTime : boolean) : void
+getMinTransfers() : boolean
+setMinTransfers(MinTransfers : boolean) : void
+getBestPriceHotel() : boolean
+setBestPriceHotel(BestPriceHotel : boolean) : void
+getClosestToVenue() : boolean
+setClosestToVenue(ClosestToVenue : boolean) : void
+getSavedPlanId() : string
+setSavedPlanId(SavedPlanId : string) : void
1
1
Concert
-Artist : string
-Venue : string
-Location : string
-Date : date
-Time : date
-Provider : string
-ProviderLink : string
-Reviews : ReviewPost list
-ConcertId : string
-Invitations : Invitation list
+getArtist() : string
+setArtist(Artist : string) : void
+getVenue() : string
+setVenue(Venue : string) : void
+getLocation() : string
+setLocation(Location : string) : void
+getDate() : date
+setDate(Date : date) : void
+getTime() : date
+setTime(Time : date) : void
+getProviderLink() : string
+getProvider() : string
+setProvider(Provider : string) : void
+setProviderLink(ProviderLink : string) : void
+getReviews() : ReviewPost list
+setReviews(Reviews : ReviewPost list) : void
+getConcertId() : string
+setConcertId(ConcertId : string) : void
+getInvitations() : Invitation list
+setInvitations(Invitations : Invitation list) : void
+return(Concert) : Concert
0..*
0..*
0..*
TravelPlan
-StartDate : date
-EndDate : date
-Hotel : Hotel
-HotelRate : int
-TransportationType : TransportationType
-Cost : string
-BestPriceTransportation : boolean
-ShortestTravelTime : boolean
-MinTransfers : boolean
-BestPriceHotel : boolean
-ClosestToVenue : boolean
-Transportation : Transportation
-DestinationLocation : string
-StartLocation : string
-TravelPlanId : string
-TotalCost : int
-Concert : Concert
+getStartDate() : date
+setStartDate(StartDate : date) : void
+getEndDate() : date
+setEndDate(EndDate : date) : void
+getHotel() : Hotel
+setHotel(Hotel : Hotel) : void
+getHotelRate() : int
+setHotelRate(HotelRate : int) : void
+getTransportationType() : TransportationType
+setTransportationType(TransportationType : TransportationType) : void
+getCost() : string
+setCost(Cost : string) : void
+getPreferences()
+setPreferences(Preferences) : void
+getBestPriceTransportation() : boolean
+setBestPriceTransportation(BestPrice : boolean) : void
+getShortestTravelTime() : boolean
+setShortestTravelTime(ShortestTravelTime : boolean) : void
+getMinTransfers() : boolean
+setMinTransfers(MinTransfers : boolean) : void
+getBestPriceHotel() : boolean
+setBestPriceHotel(BestPriceHotel : boolean) : void
+getClosestToVenue() : boolean
+setClosestToVenue(ClosestToVenue : boolean) : void
+getTransportation() : Transportation
+setTransportation(Transportation : Transportation) : void
+getLocation() : string
+setLocation(Location : string) : void
+getStartLocation() : string
+setStartLocation(StartLocation : string) : void
+getTravelPlanId() : string
+setTravelPlanId(TravelPlanId : string) : void
+getTotalCost() : int
+setTotalCost(TotalCost : int) : void
+getConcert() : Concert
+setConcert(Concert : Concert) : void
+chooseHotel(StartLocation : string, HotelRate : int, StartDate : date, EndDate : date, ClosestToVenue : boolean, BestPriceHotel : boolean) : void
+chooseTransportation(StartLocation : string, Transportation : TransportationType, StartDate : date, EndDate : date, MinTransfers : boolean) : void
+displayPlanPackage(hotel : Hotel, transportation : Transportation) : void
+updateSavedPlans() : void
1
1
1
1
1
1
User
-Name : string
-Email : string
-Password : string
-favoriteArtist : string
-Location : string
-userID : float
-Invitations : Invitation list
-GroupChats : Groupchat list
-Plans : savedlPlan list
-UserName : list
-ReviewPosts : ReviewPost list
+getName() : string
+setName(Name : string) : void
+getEmail() : string
+setEmail(Email : string) : void
+getPassword() : string
+setPassword(Password : string) : void
+getFavoriteArtist() : artist
+setFavoriteArtist(favoriteArtists : Artist) : void
+getLocation() : string
+setLocation(Location : string) : void
+getUserId() : float
+setUserId(userId : float) : void
+searchConcert(Concert : Concert) : void
+filterSearch(Country : string, City : string) : void
+getInvitations() : Invitation list
+setInvitations(Invitations : Invitation list) : void
+getGroupChats() : GroupChat list
+setGroupChats(GroupChats : GroupChat list) : void
+setPlan(Plans : savedPlan list) : void
+getPlans() : savedPlan list
+addPlan(Plan : savedPlan) : boolean
+removePlan(Plan : savedPlan) : boolean
+getUserName() : list
+setUserName(userName : list) : void
+selectLanguage() : void
+selectCurrency() : void
+selectLocation() : boolean
+selectDate() : boolean
+selectRateHotel() : boolean
+selectTransportation() : boolean
+applyFilters() : boolean
+getReviewPosts() : ReviewPost list
+setReviewPosts(ReviewPosts : ReviewPost list) : void
+addInvitations(Concert : Concert) : void
+sendReview() : boolean
+deleteReview() : boolean
+deleteInvitation() : boolean
+sendMessage(requester : User) : void
+deleteInvitation(invitationId : int) : void
+enterInformation(Concert : Concert, StartLocation : string, Transportation : TransportationType, HotelRate : int, startDate : date, endDate : date) : void
+enterOptionals(Concert : Concert, ShortestTravelTime : boolean, ClosestToVenue : boolean, BestPriceHotel : boolean, minTransfers : boolean) : void
+savePlan(Plan : TravelPlan) : void
0..*
0..*
0..*
0..*
0..*
ReviewPost
-Creater : User
-PostId : string
-text : string
-Rate : int
-Concert : string
-Artist : String
+getCreater() : User
+setCreater(Creater : User) : void
+getPostId() : string
+setPostId(PostId : string) : void
+getText() : string
+setText(text : string) : void
+getRate() : int
+setRate(Rate : int) : void
+getConcert() : string
+setConcert(Concert : string) : void
+getArtist() : String
+setArtist(Artist : String) : void
1
1
GroupChat
-Description : string
-Concert : Concert
-ChatParticipantSize : int
-CharParticipants : User list
-GroupChatId : string
+getDescription() : string
+setDescription(Description : string) : void
+getConcert() : Concert
+setConcert(ChatConcert : Concert) : void
+getAttribute()
+setAttribute(attribute) : void
+getChatParticipantSize() : int
+setChatParticipantSize(ChatParticipantSize : int) : void
+getCharParticipants() : User list
+setCharParticipants(CharParticipants : User list) : void
+getGroupChatId() : string
+setGroupChatId(GroupChatId : string) : void
+receiveMessage(requester : User) : void
1..*
Invitation
-InvitationDescription : string
-Requester : User
-InvitationId : string
+getInvitationDescription() : string
+setInvitationDescription(InvitationDescription : string) : void
+getRequester() : User
+setRequester(Requester : User) : void
+getInvitationId() : string
+setInvitationId(InvitationId : string) : void
+createGroupChat(InvitationId : int) : void
+deleteGroupChat(InvitationId : int) : void
+updateGroupChat(requester : User) : void
1
1
Hotel
-HotelName : string
-HotelRate : int
-HotelPrice : string
-Provider : string
-ProviderLink : string
-HotelId : string
+getHotelName() : string
+setHotelName(HotelName : string) : void
+getHotelRate() : int
+setHotelRate(HotelRate : int) : void
+getHotelPrice() : string
+setHotelPrice(HotelPrice : string) : void
+getProvider() : string
+setProvider(Provider : string) : void
+getProviderLink() : string
+setProviderLink(ProviderLink : string) : void
+getHotelId() : string
+setHotelId(HotelId : string) : void
+getAttribute()
+setAttribute(attribute) : void
+return(Hotel) : Hotel
Transportation
-TicketPrice : string
-TransportationDate : date
-StartingLocation : string
-DestinationLocation : string
-Provider : string
-ProviderLink : string
-TransportationId : string
+getTicketPrice() : string
+setTicketPrice(TicketPrice : string) : void
+getTransportationDate() : date
+setTransportationDate(TransportationDate : date) : void
+getStartingLocation() : string
+setStartingLocation(StartingLocation : string) : void
+getDestination() : string
+setDestination(Destination : string) : void
+getProvider() : string
+setProvider(Provider : string) : void
+getProviderLink() : string
+setProviderLink(ProviderLink : string) : void
+getTransportationId() : string
+setTransportationId(TransportationId : string) : void
+return(Transportation) : Transportation
1
1
<<enumeration>>
TransportationTyp
-Bus
-Flight
Bus
-StartingPointTerminal : string
-DestinationTerminal : string
-BusFirm : string
-TravelDuration : int
+setStartingPointTerminal(StartingPointTerminal : string) : void
+getDestinationTerminal() : string
+setDestinationTerminal(DestinationTerminal : string) : void
+getBusFirm() : string
+getStartingPointTerminal() : string
+setBusFirm(BusFirm : string) : void
+getTravelDuration() : int
+setTravelDuration(TravelDuration : int) : void
Flight
-StartingAirport : string
-DestinationAirport : string
-FlightCode : string
-Carrier : string
-FlightDuration : int
-TransferCount : int
+getStartingAirport() : string
+setStartingAirport(StartingAirport : string) : void
+getDestinationAirport() : string
+setDestinationAirport(DestinationAirport : string) : void
+getFlightCode() : string
+setFlightCode(FlightCode : string) : void
+getCarrier() : string
+setCarrier(Carrier : string) : void
+getFlightDuration() : int
+setFlightDuration(FlightDuration : int) : void
+getTransferCount() : int
+setTransferCount(TransferCount : int) : void

For a better view of the class diagram, online link: https://imgur.com/a/6xH7CQK .

3.5.3.2 Explanation of the Class Diagram

User: This class contains all the information that is required to identify the user of the application. In addition, group chats, plans, and posts are stored in lists in this class.

TravelPlan: The TravelPlan class is used to represent the object that represents a travel plan. This class includes the information that is gathered from the user in order to create a travel plan.

SavedPlan: This class shows the saved travel plans of the user. The preferences that the user has set for later use of the travel plan can be accessed by the user through this object when they wish to access the plan.

Concert: This class represents the concert object, it is used for storing data which is fetched from the api.

Hotel: This class represents a hotel object, it is used for storing hotel related data which is fetched from the api.

Transportation: This class represents the transportation object, it includes flight and bus objects. It is used for storing transportation data which is fetched from the api.

Flight: This class represents the flight of the user, it includes flight related data.

Bus: The Bus class represents the bus trip taken by the user, along with all the data associated with the bus trip.

<<enumeration>> TransportationType: It has been determined that this enumeration is one of the most commonly used to indicate transportation types (flight, bus).

ReviewPost: This class represents the review post of the user, it includes the data about the user's review on concerts.

GroupChat: This class represents the group chats of the user, it includes the information related to the chats, participants.

Invitation: This class represents the invitation which user’s create when they invite other users to the group chats.

3.5.4 Dynamic Models

3.5.4.1 Sequence Diagram

3.5.4.1.1 Sequence Diagram for Creating Invitations

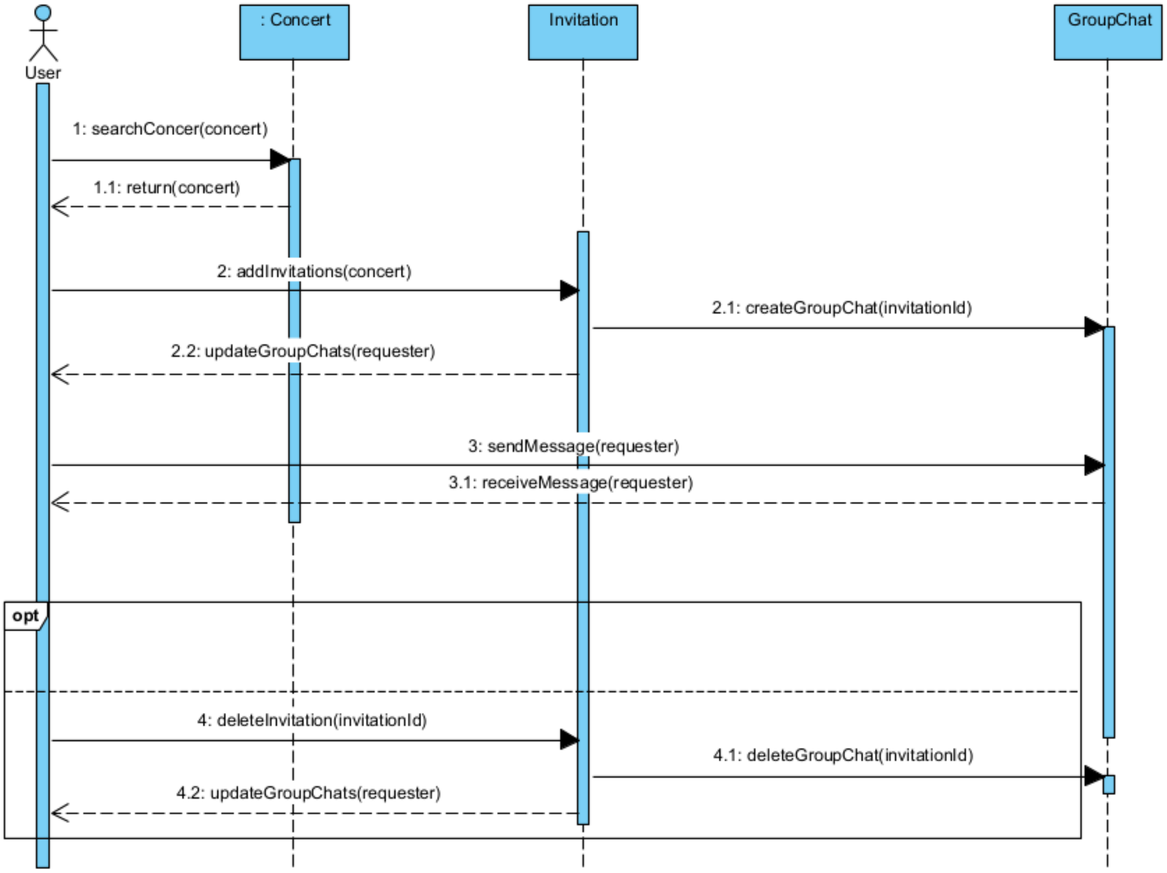

User
: Concert
Invitation
GroupChat
1: searchConcer(concert)
1.1: return(concert)
2: addInvitations(concert)
2.1: createGroupChat(invitationId)
2.2: updateGroupChats(requester)
3: sendMessage(requester)
3.1: receiveMessage(requester)
opt
4: deleteInvitation(invitationId)
4.1: deleteGroupChat(invitationId)
4.2: updateGroupChats(requester)

Figure 3. Sequence diagram for invitation creation

Users can search for a concert they want and add invitations for that concert. When a user adds an invitation, there is a group chat created for that invitation and createGroupChat is used for this purpose and it takes the invitationId as its parameter. Then this group chat is added to the group chats of that user. Users can send and receive messages via group chats. If they wish, they can delete an invitation and the group chat for this invitation will be also deleted. Invitation then updates the group chats page of the user.

3.5.4.1.2 Sequence Diagram for Creating Travel Plans

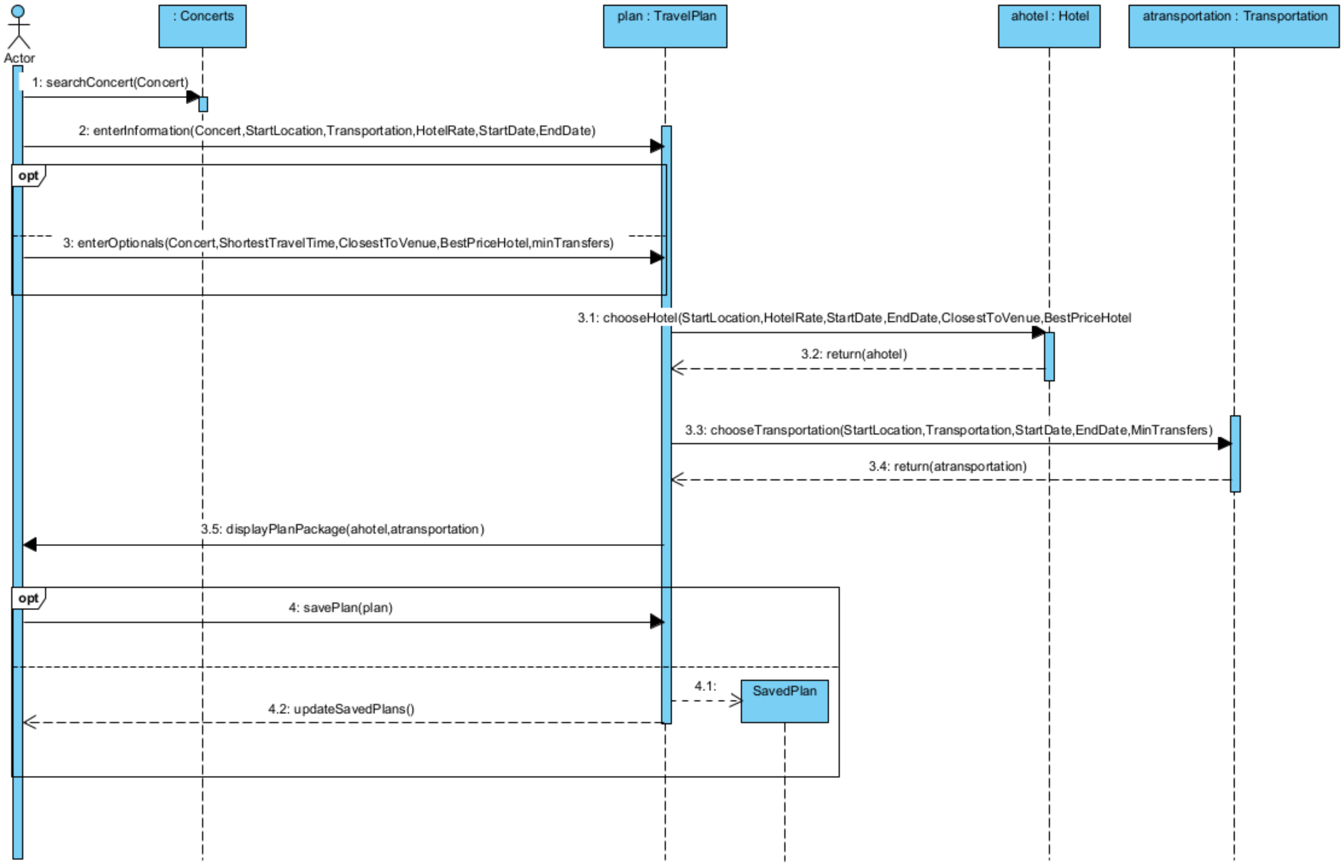

Actor
: Concerts
plan : TravelPlan
ahotel : Hotel
atransportation : Transportation
1: searchConcert(Concert)
2: enterInformation(Concert,StartLocation,Transportation,HotelRate,StartDate,EndDate)
opt
3: enterOptionals(Concert,ShortestTravelTime,ClosestToVenue,BestPriceHotel,minTransfers)
3.1: chooseHotel(StartLocation,HotelRate,StartDate,EndDate,ClosestToVenue,BestPriceHotel
3.2: return(ahotel)
3.3: chooseTransportation(StartLocation,Transportation,StartDate,EndDate,MinTransfers)
3.4: return(atransportation)
3.5: displayPlanPackage(ahotel,atransportation)
opt
4: savePlan(plan)
4.1:
SavedPlan
4.2: updateSavedPlans()

Figure 4. Sequence diagram for travel plan creation

Users can search for a concert and create a travel plan for this concert after entering their necessary and optional preferences. Travel plan finds the optimal hotel and transportation based on those preferences and returns the package to the user. If they wish, users can save the travel plans they want and that plan will be added to their saved plan list and will be shown on their saved plans page.

3.5.4.1 Activity Diagram

Log In & Sign Up

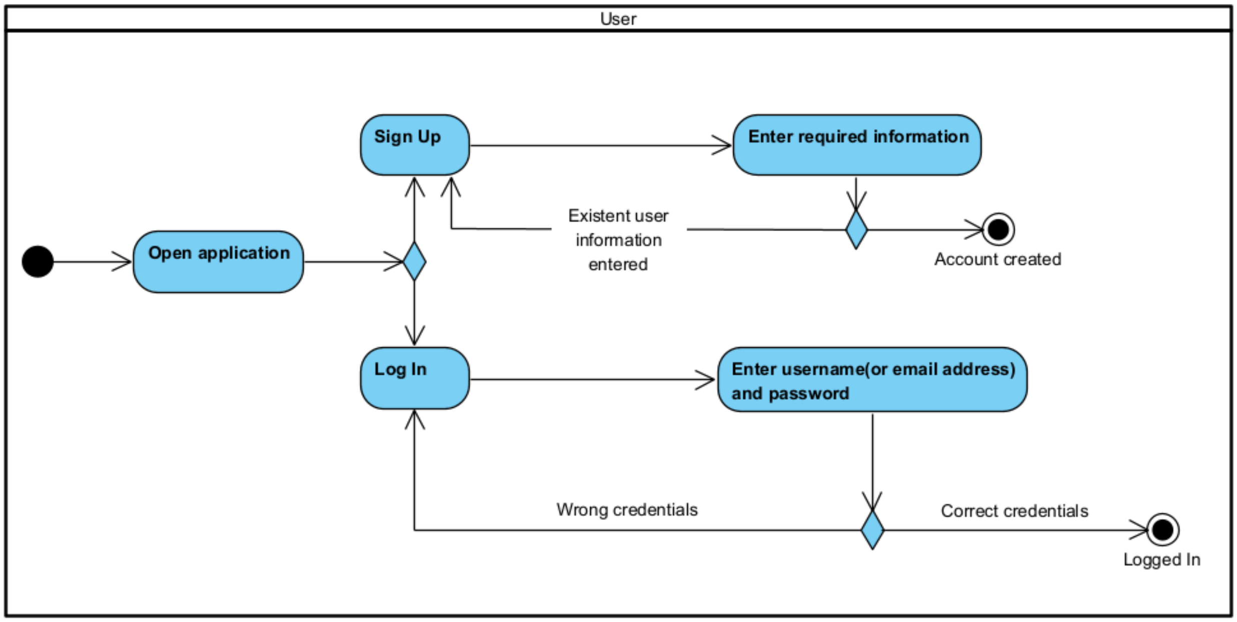

User
Sign Up
Enter required information
Existent user information entered
Account created
Open application
Log In
Enter username(or email address) and password
Wrong credentials
Correct credentials
Logged In

Figure 5. Log-in and sign-up activity diagram

New users are asked to enter their name and surname, username, email address and phone number when they are creating an account for the first time.

Find concert

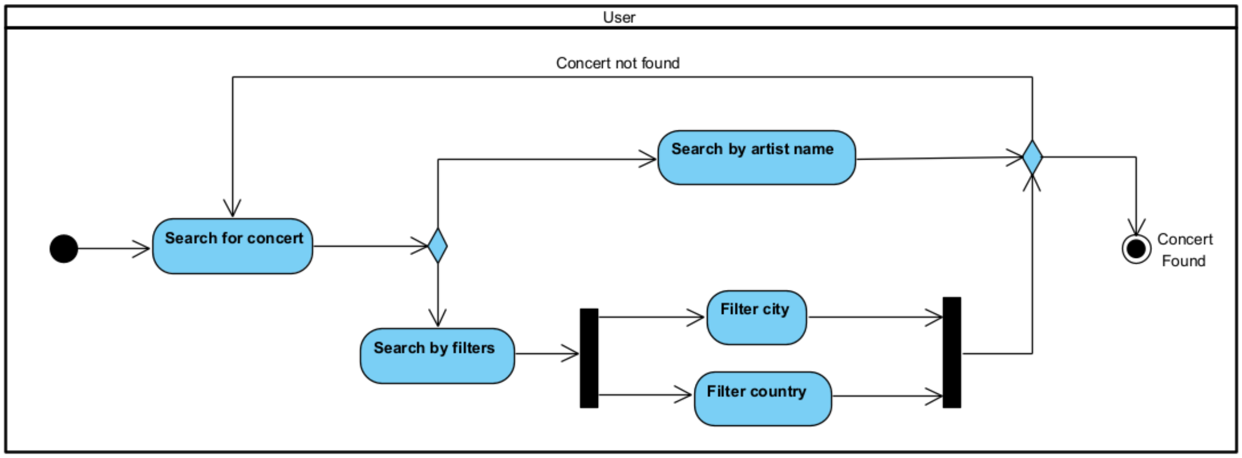


Figure 6. Find concert activity diagram

In Funravel, users can search for concerts by entering the artist name. They can also filter out certain cities or countries if they don't wish to see the concerts taking place in those places.

Planning

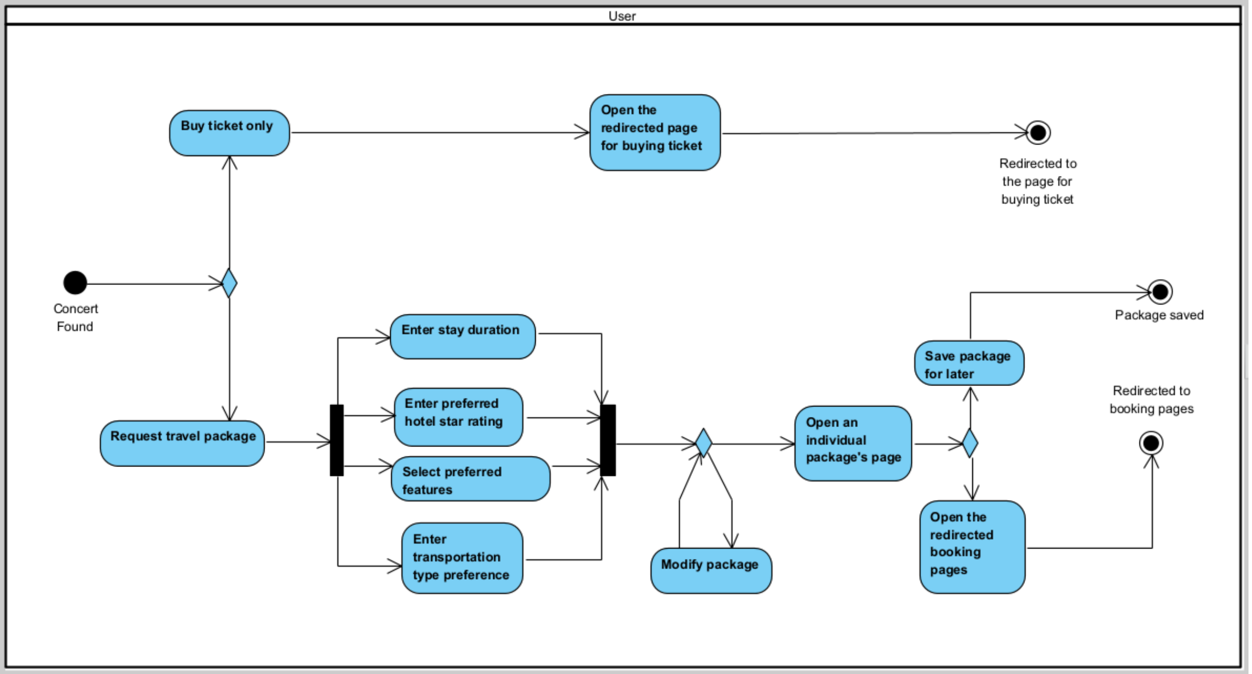

User
Buy ticket only
Open the redirected page for buying ticket
Redirected to the page for buying ticket
Concert Found
Package saved
Enter stay duration
Save package for later
Redirected to booking pages
Enter preferred hotel star rating
Open an individual package's page
Request travel package
Select preferred features
Open the redirected booking pages
Enter transportation type preference
Modify package

Figure 7. Planning activity diagram

After finding a concert in the system, users can choose to buy the ticket for the concert directly or they can ask for a travel plan. When they request a travel plan, they are asked to enter their stay duration, preferred hotel star rating, select preferred feature options (best price, closest to venue etc.) and select transportation type. When the package is created, users can modify the package. Modification here is changing the suggested transportation or hotel option one by one according to the options, or see all possibilities and find what they want specifically. For example, if a user has chosen the "best price" option when entering preferences, when they change the suggested hotel in a one-by-one approach, the next possible accommodation option displayed for the user is the one with the lowest price. Users can also share their preferences to generate plans later. They can proceed to payments as well, by clicking on the redirection buttons for transportation and hotel suggestions.

Invitations

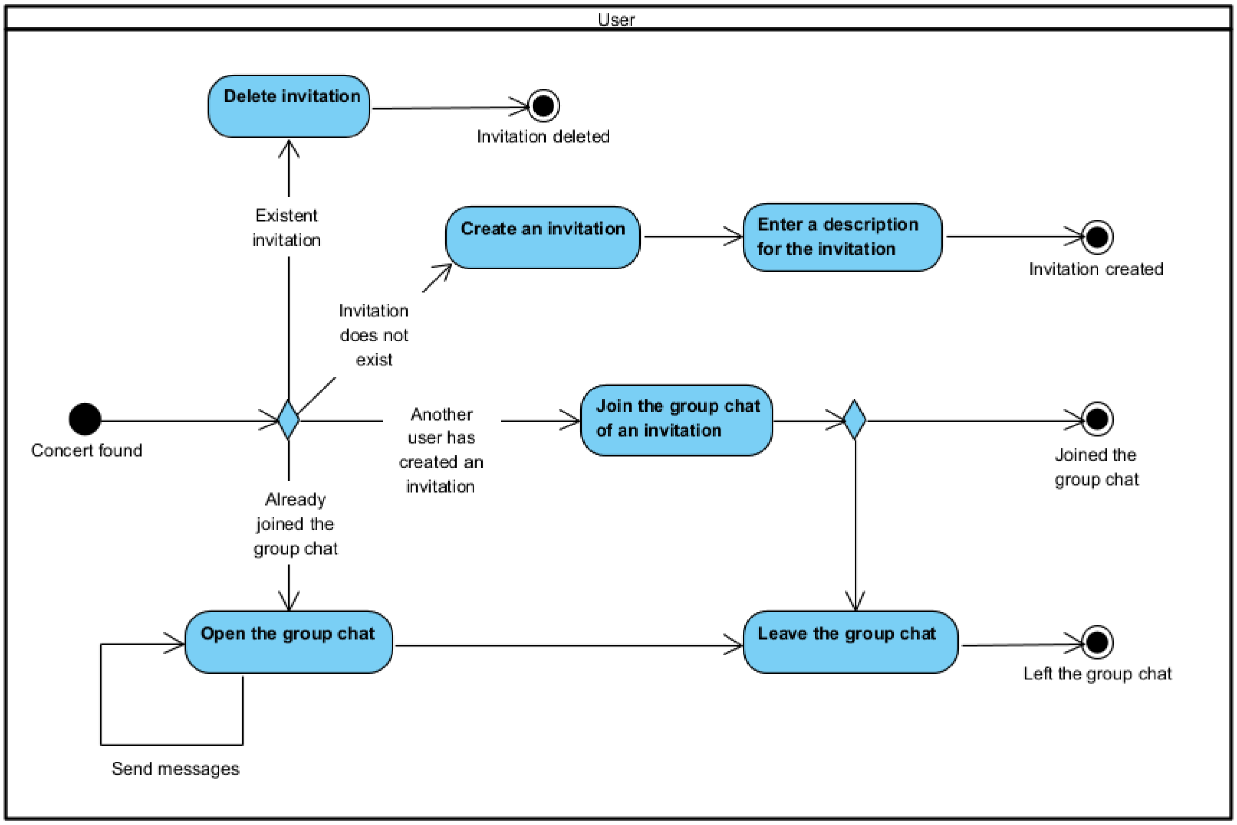


Figure 6. Invitations activity diagram

Users can create invitations, delete an invitation they created, join another invitation's group chat or leave them.

Reviews

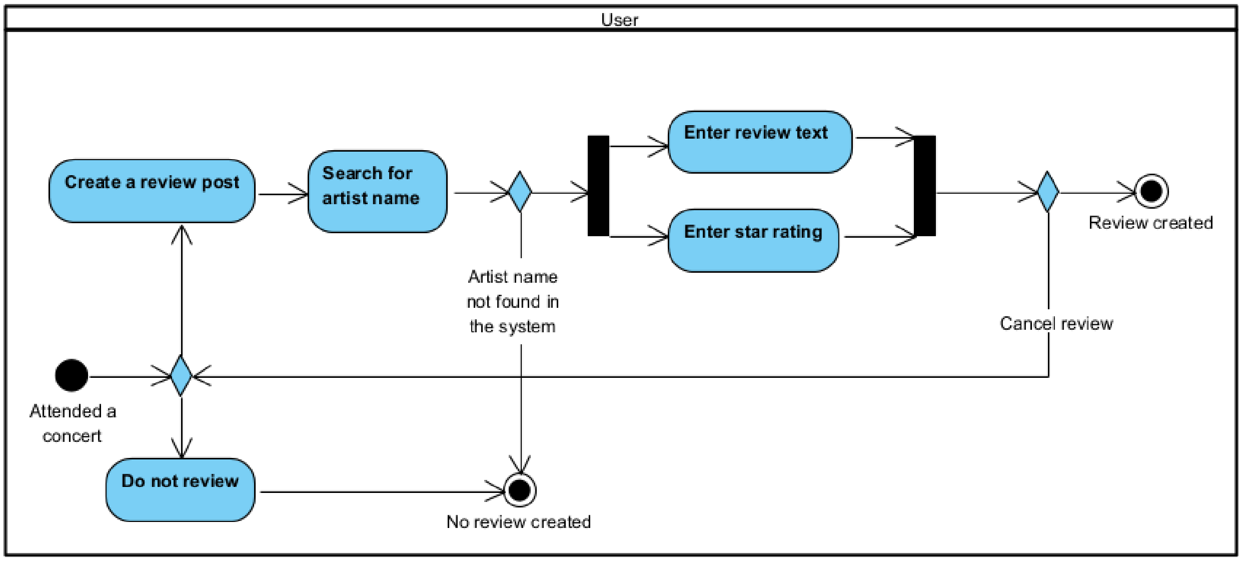


Figure 7. Reviews activity diagram

Users can create review posts to share their experiences. When they want to create a review, they enter their review text and a star rating. Also, in order for the review to be successfully created, the user must find the name of the artist they want to talk about the concert of. The name of the artist is used like a tag for the review post. If the artist's name is not in the system and the user cannot find it, the review cannot be created.

3.5.4.1 State Diagram

Suggesting a Travel Plan

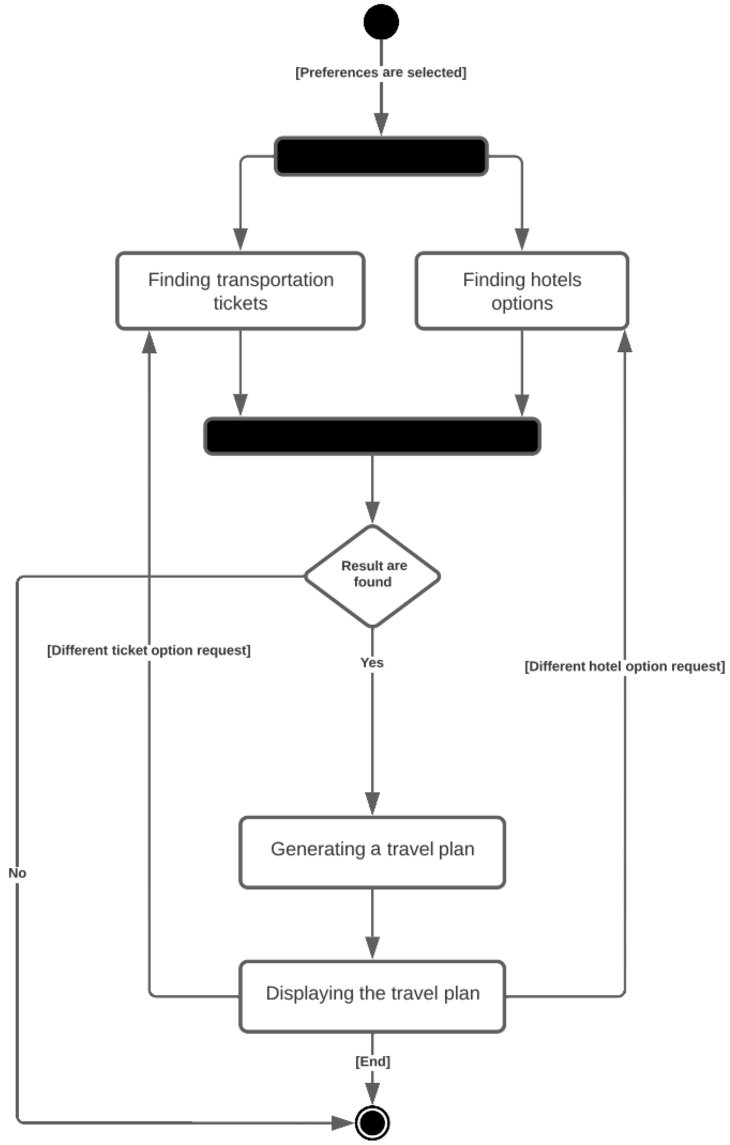

[Preferences are selected]
Finding transportation tickets
Finding hotels options
Result are found
[Different ticket option request]
Yes
[Different hotel option request]
No
Generating a travel plan
Displaying the travel plan
[End]

Figure 8. Travel Plan Suggestion state diagram

In this state diagram, it is shown that after obtaining the preferences of users the system finds the optimal transportation tickets and hotel options. If the results are found without an error, it generates a plan and displays it, else displays that no such no such options are found. Also after displaying the plan, the user can ask for a different ticket or hotel option and in this case the system goes back to the state where it searches for optimal tickets.

3.5.5 User Interface

3.5.5.1 Navigational Path

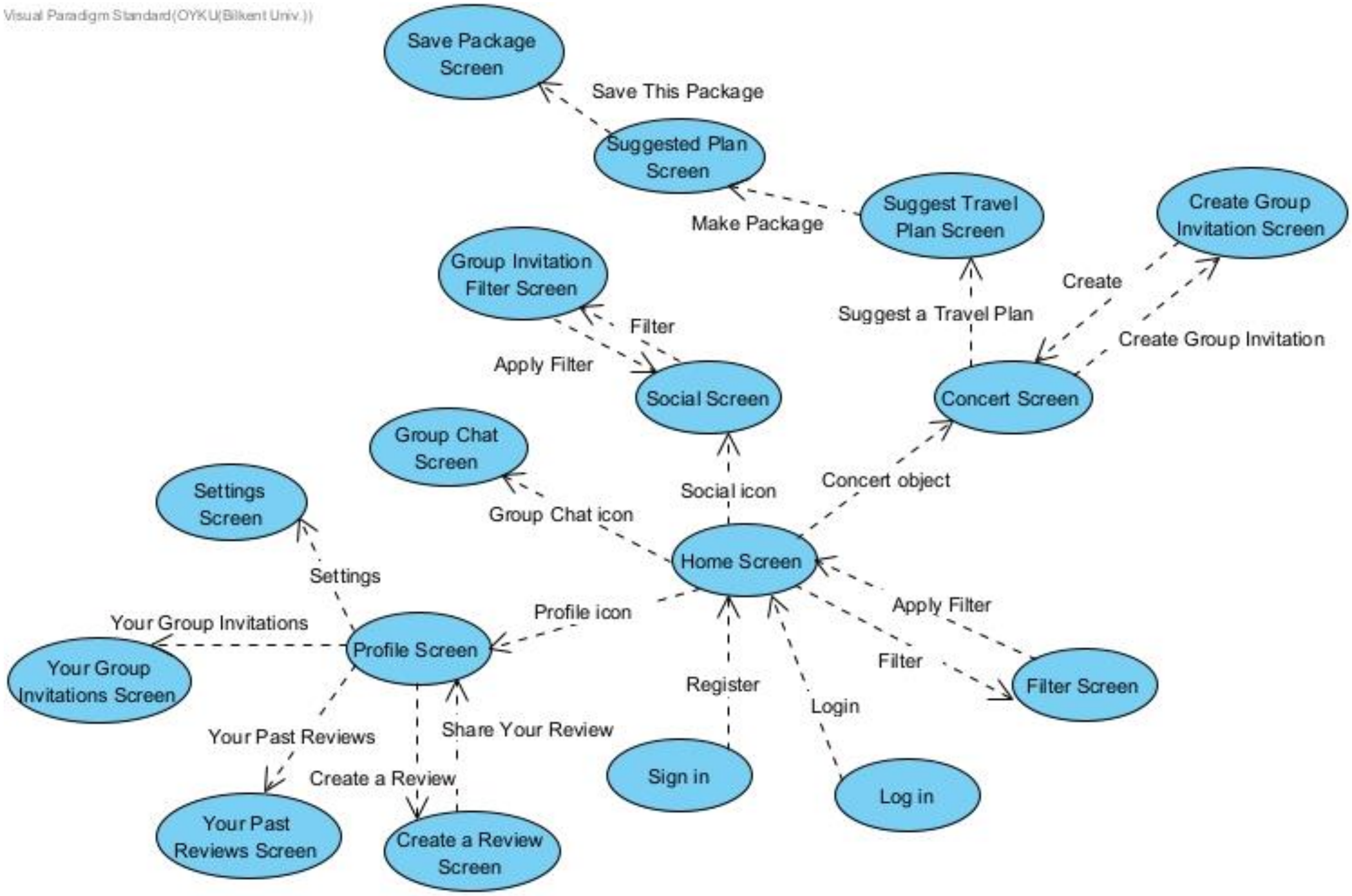


Figure 9. Navigational path

3.5.5.2 Login screen

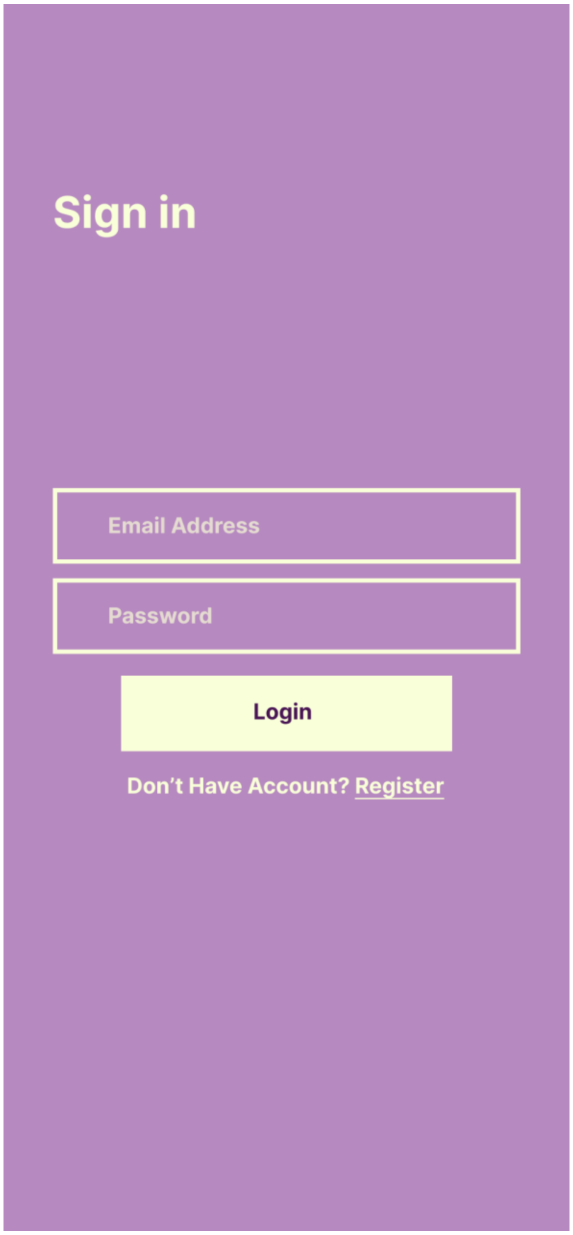


Figure 10. Login screen

When the user first opens the app, a login screen shows up, asking the user to enter their account information. If the user does not have an account yet, they can click Register to create a new account.

3.5.5.3 Register screen

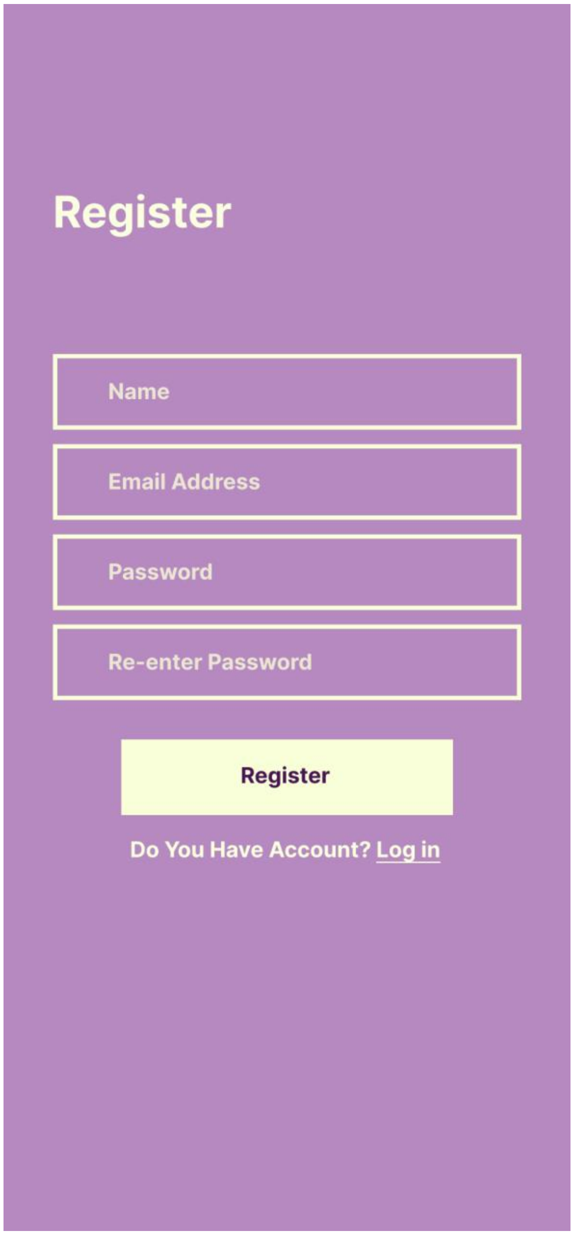


Figure 11. Register screen

In this screen, users can create a new account by entering their name, email, and password. When they click register, if the entered information is valid an account will be created and the user is redirected to the login screen to log in.

3.5.5.4 Home screen

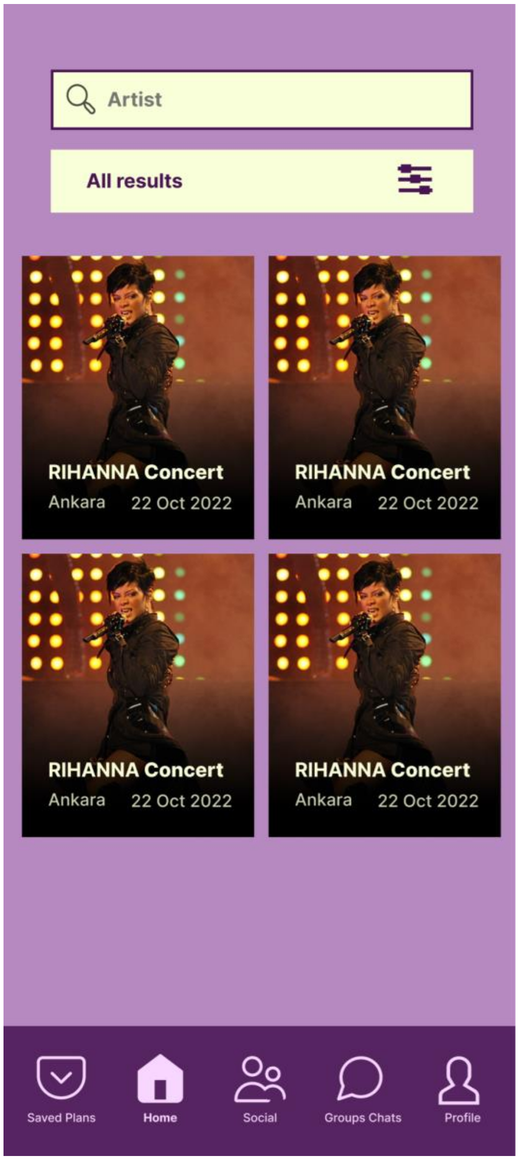


Figure 12. Home screen

After the user is successfully logged in, the app takes them to the home screen where they can see and scroll through all upcoming concerts. If they want to filter the concerts, they can search for a specific artis by tapping the search bar and entering their desired artist. If the artist has an upcoming concert, they will be shown on the screen. In addition, they can add extra filters by clicking on the filter icon.

3.5.5.5 Filter screen

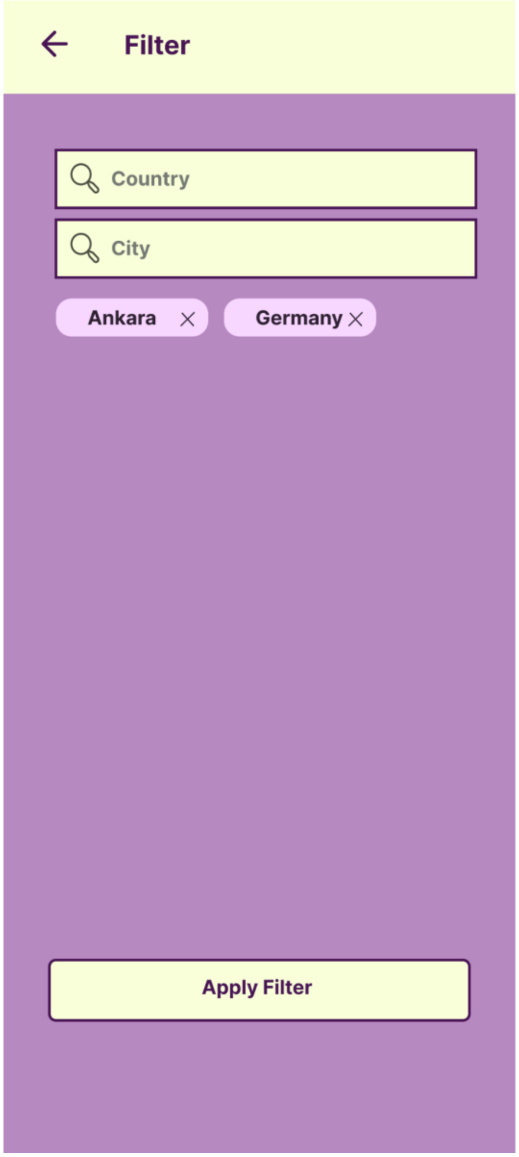


Figure 13. Filter screen (for concerts and group invitations)

In the filter screen, users can add a country and city filters. Users can add as many countries/cities as they want to the filter and the results are to be all the concerts that are held in the countries/cities selected.

After the user clicks "Apply Filter", they are directed to the home page where they can see the filtered results. See figure 14.

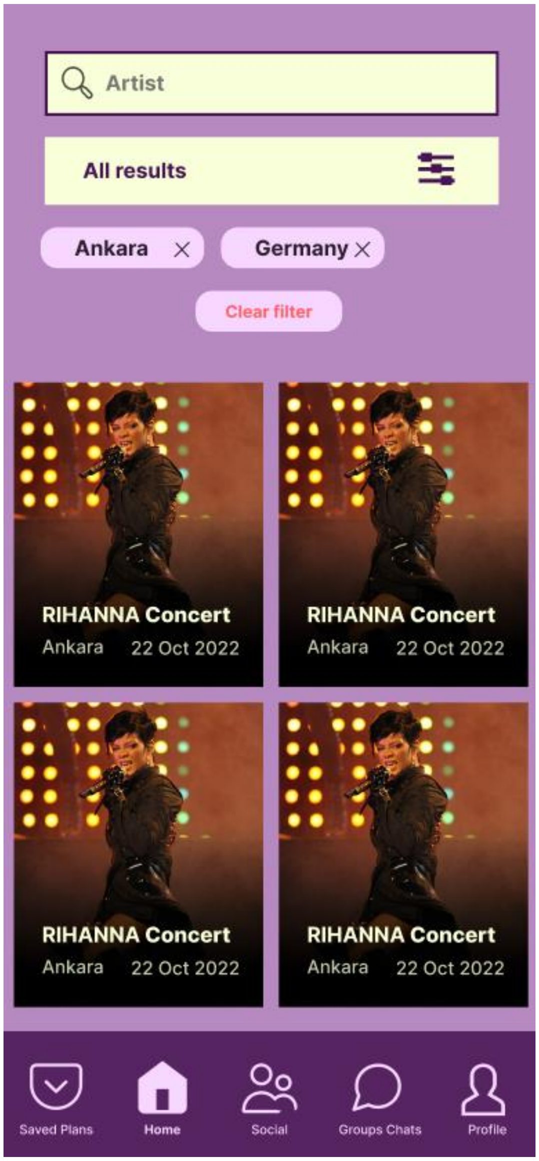
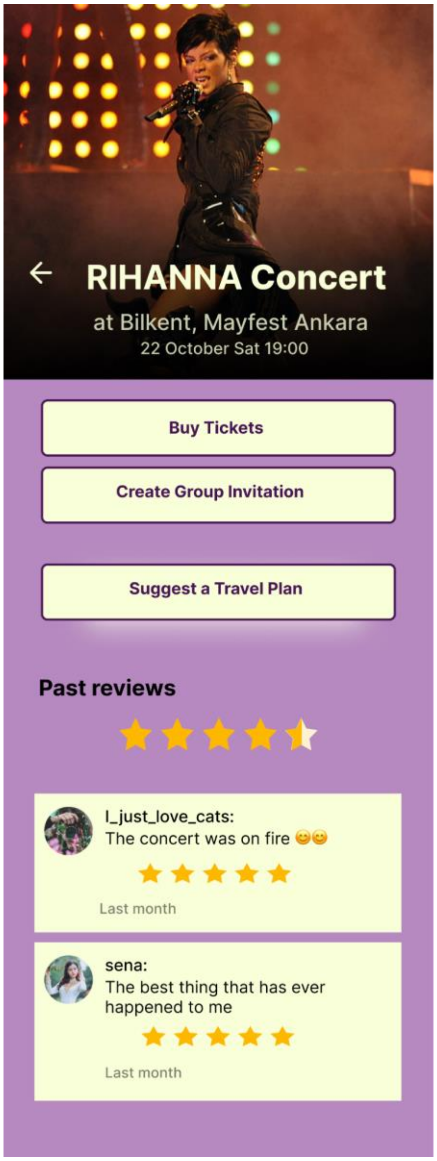


Figure 14. Home screen (after filters are applied)

Filtered results are shown in the home screen after the filter is applied. All filter selections can be seen from that screen too. They can delete one of the filters or clear all of them. Moreover, they can click the filter icon again which will direct them to the filter screen where they can apply more filters or edit the current ones.

3.5.5.6 Concert screen



RIHANNA Concert

at Bilkent, Mayfest Ankara

22 October Sat 19:00

l_just_love_cats:
The concert was on fire 😊😊

Last month

sena:
The best thing that has ever happened to me

Last month

Figure 15. Concert screen

When a user clicks on one concert widget, they are directed to the concert screen. At this screen they can see all the details of the concert. A user can take a number of actions from this screen. Firstly, they can tap “Buy Tickets” to buy the concert’s ticket. Secondly, they can tap “Create Group Invitation” to create a group invitation. Thirdly, they can request a travel plan suggestion by tapping the “Suggest a Travel Plan" button. Moreover, a user can see reviews’ posts previously left to the artist.

3.5.5.7 Create Group Invitation screen

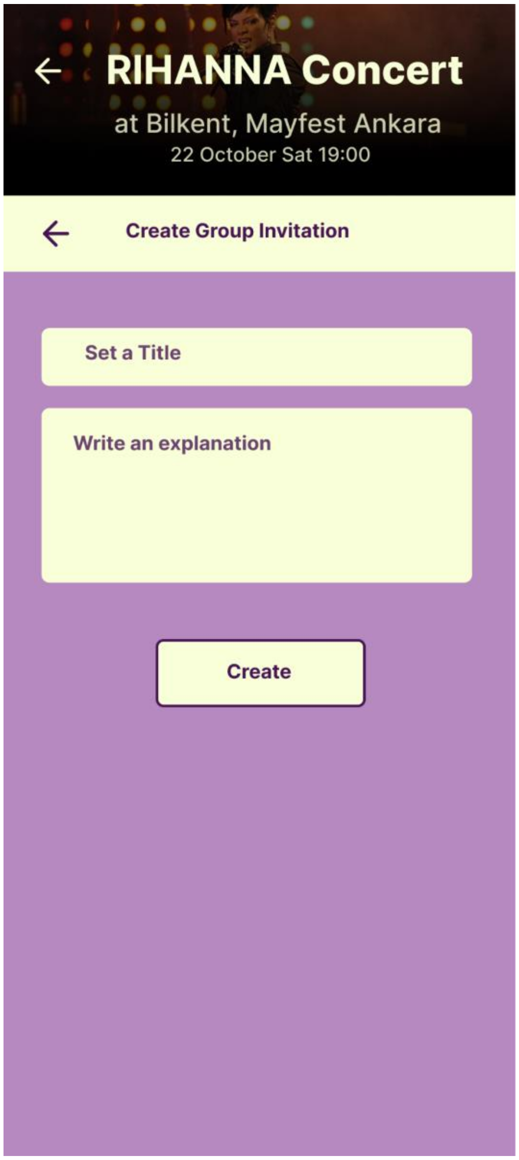


Figure 16. Create group invitation screen

This screen is accessed by clicking on the “Create Group Invitation” button at the concert screen. At this page a user can type a title as well as an invitation post explanation. They can then create the group invitation post. All group invitation posts can be seen in the social screen as well as from “Your Group Invitations” in the profile screen. See Figures 19 and 20.

Moreover, whenever a new group invitation is created, a group chat for this invitation is automatically created. The group chat can be accessed by the post owner as well as all users who joined this invitation group from the groups chats screen, Figure 20

3.5.5.8 Suggest Travel Plan screen

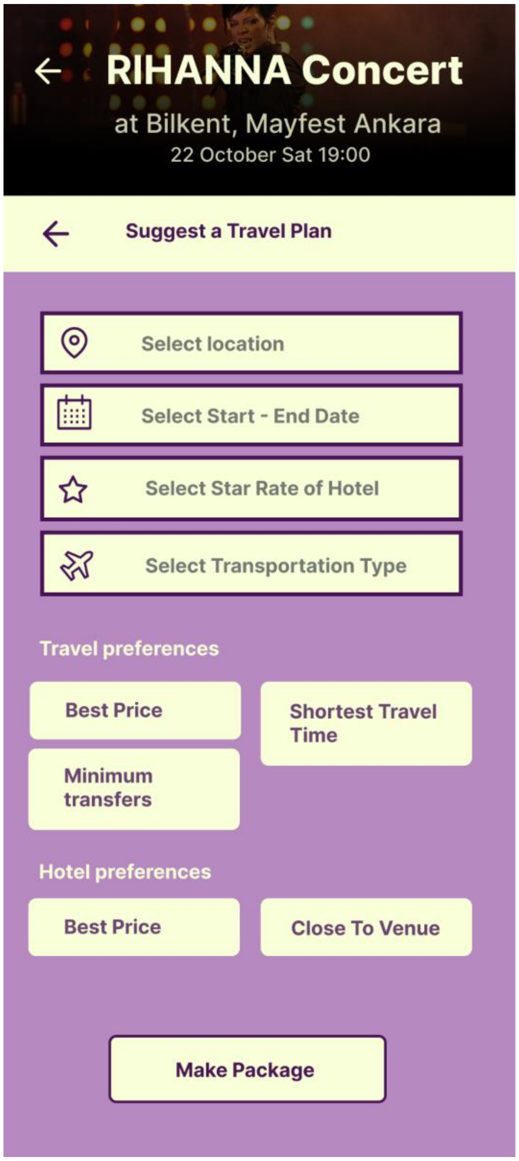


Figure 17. Suggest a travel plan screen (adding preferences)

This screen is accessed by tapping on the “Suggest a Travel Plan" button in the concert screen. At this screen a user can enter their travel plan information and transportation and hotel preferences. They should enter the start location, plan’s start and end dates, hotel star preference, and transportation type (plane, bus). In addition, they can select other preferences such as the best price transportation or a hotel that is close to the venue. They then can tap on “Make Package” to get a plan packet suggestion based on their preferences. Figure 18 shows the package suggestion screen.

3.5.5.9 Suggested Plan screen

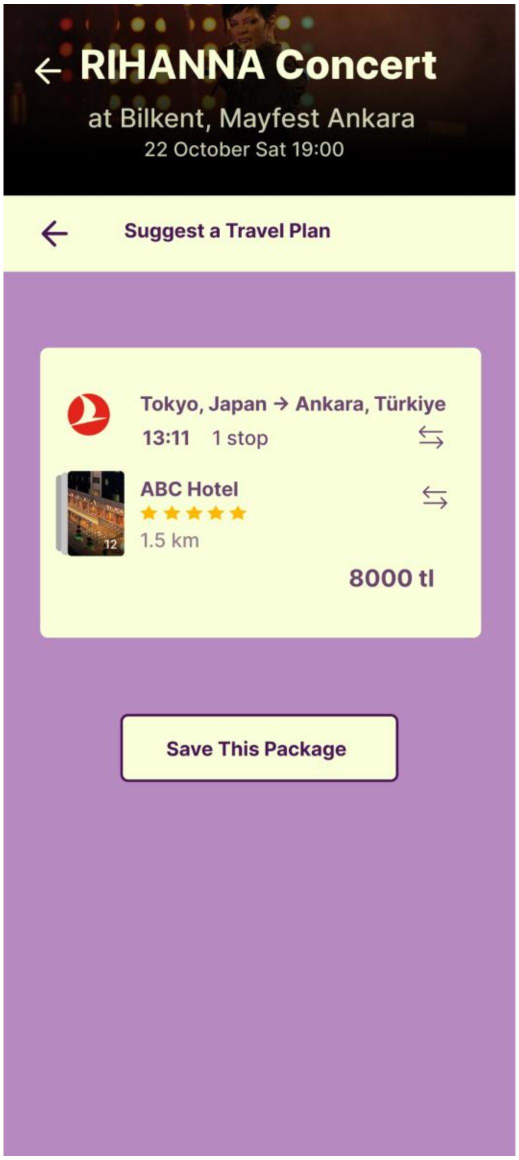
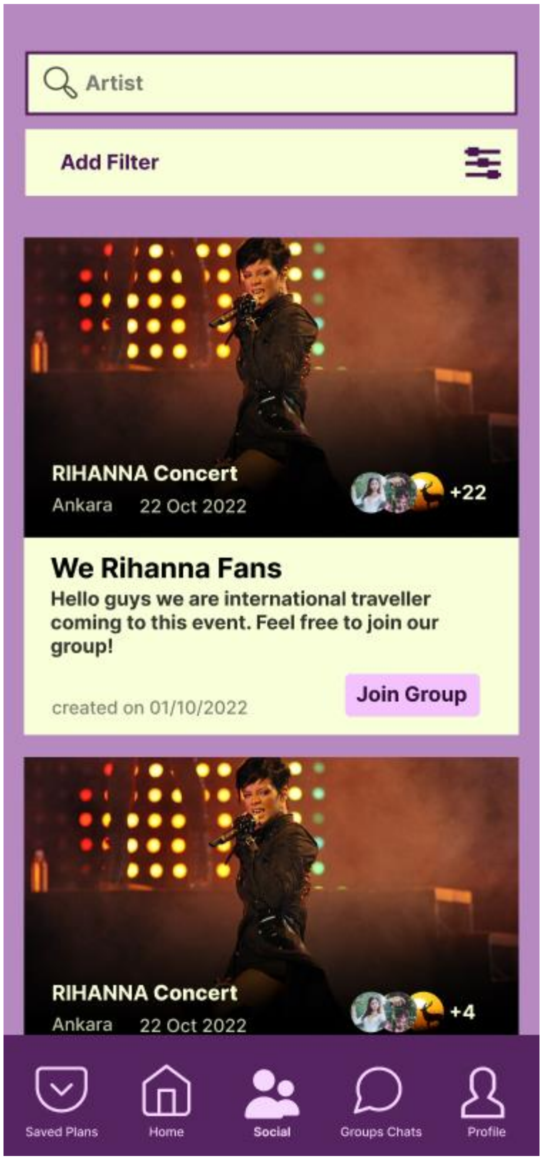


Figure 18. Suggested plan screen (package suggestion)

At this screen, the user can see the generated package with the necessary transportation and hotel information. They can see the transportation details, hotels name and photos, and price. Also, they can click on "Save This Package". Saved packages/plans can be viewed from the "Saved Plans" screen accessed from the navigation bar.

3.5.5.10 Social screen



RIHANNA Concert

Ankara 22 Oct 2022

We Rihanna Fans

Hello guys we are international traveller coming to this event. Feel free to join our group!

created on 01/10/2022

RIHANNA Concert

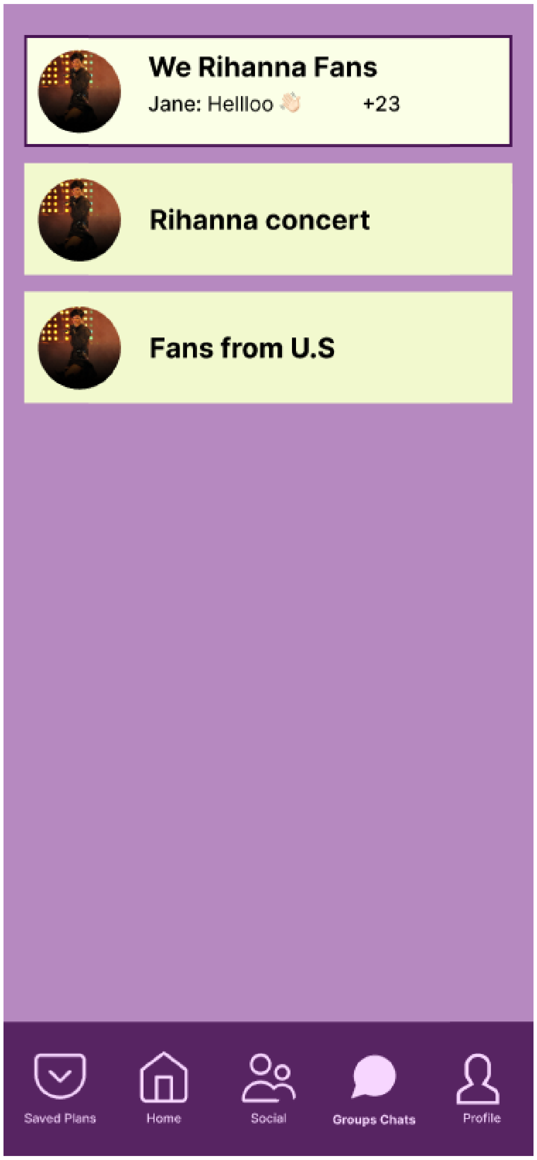
Ankara 22 Oct 2022

Figure 19. Social screen

At this page, users can see all group invitation posts created by other users. They can join a group by clicking “Join Group”. After they join a group, they can access the post’s group chat from the Groups Chats screen.

Similar to Home page, a user can search posts by artist name. They can also add extra filters by tapping “Add Filter”.

3.5.5.11 Groups Chats screen



We Rihanna Fans

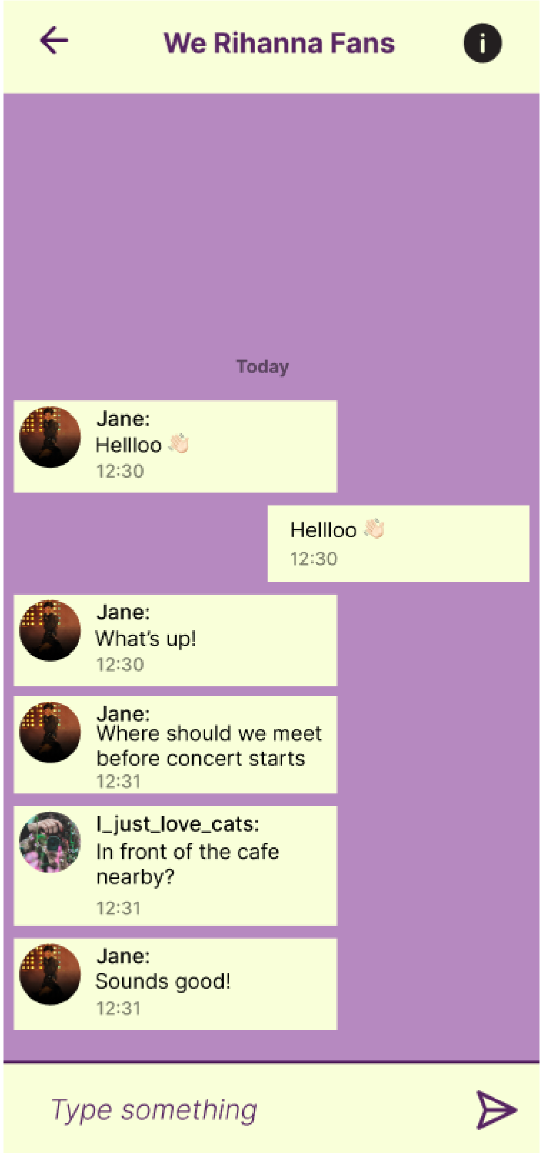
Jane: Hellloo 👋 +23

Rihanna concert

Fans from U.S

Figure 20. Groups Chats screen

In this page a user can see all the groups chats of the invitations they have joined. A group that has new messages has a brighter design color and stroke border as well as insights of the new message's content. When the user clicks on one group they are directed to the chat messages screen, see Figure 21.



Jane:
Hellloo 👋
12:30

Hellloo 👋
12:30

Jane:
What’s up!
12:30

Jane:
Where should we meet before concert starts
12:31

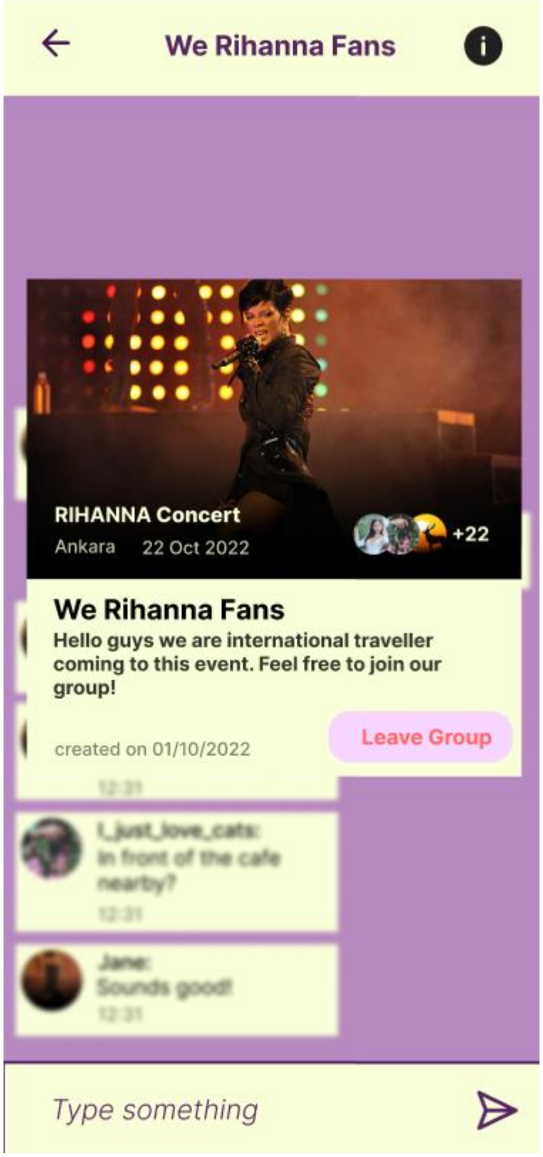
I_just_love_cats:
In front of the cafe nearby?
12:31

Jane:
Sounds good!
12:31

Figure 21. Groups Chats screen (chat messages screen)

Here a user can see a group members’ messages and reply to them. They can write a message in the “Type something” field and click on the arrow icon to send the message.

They can also click on the info icon on the top right corner to see details about the group, leave the group, or go to the concert screen by clicking on the concert title. See Figure 22.



RIHANNA Concert

Ankara 22 Oct 2022

We Rihanna Fans

Hello guys we are international traveller coming to this event. Feel free to join our group!

created on 01/10/2022

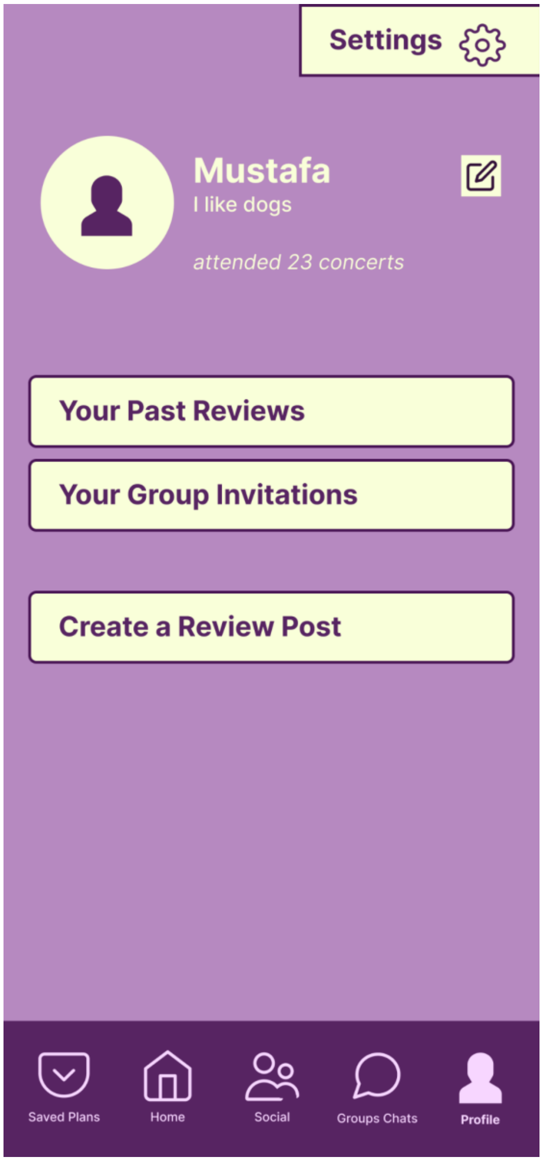
12:31

I_just_love_cats:
In front of the cafe nearby?
12:31

Jane:
Sounds good!
12:31

Figure 22. Groups Chats screen (group chat details)

3.5.5.12 Profile screen



Mustafa

I like dogs

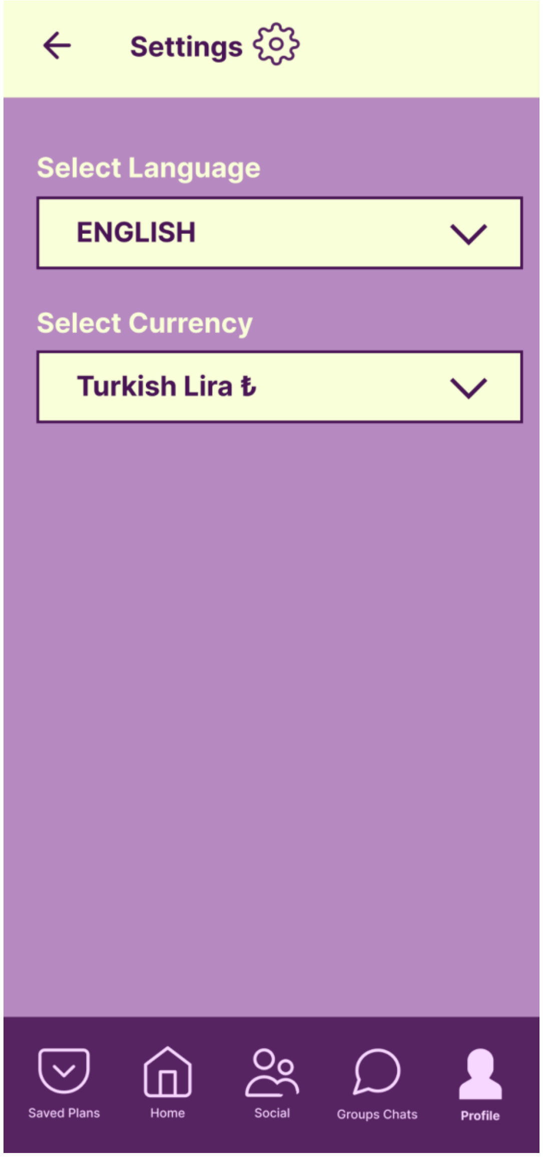
attended 23 concerts

Figure 23. Profile screen

At this screen, a user can see their profile details. They can see their profile photo, full name, username, and bio.

They can also access different features from this screen; They can click on settings icon to navigate to settings screen; they can click on the edit icon to edit their profile information; also, they can click on “Your Past Reviews” to see their past reviews on artists, if any; in addition, they can click on “Your Group Invitations” to see the groups they have created; and finally they can tap “Create a Review Post” to leave a review post to an artist, which is displayed to their concerts screens. See Figures 24-27.

3.5.5.13 Settings screen



Settings

Select Language

ENGLISH

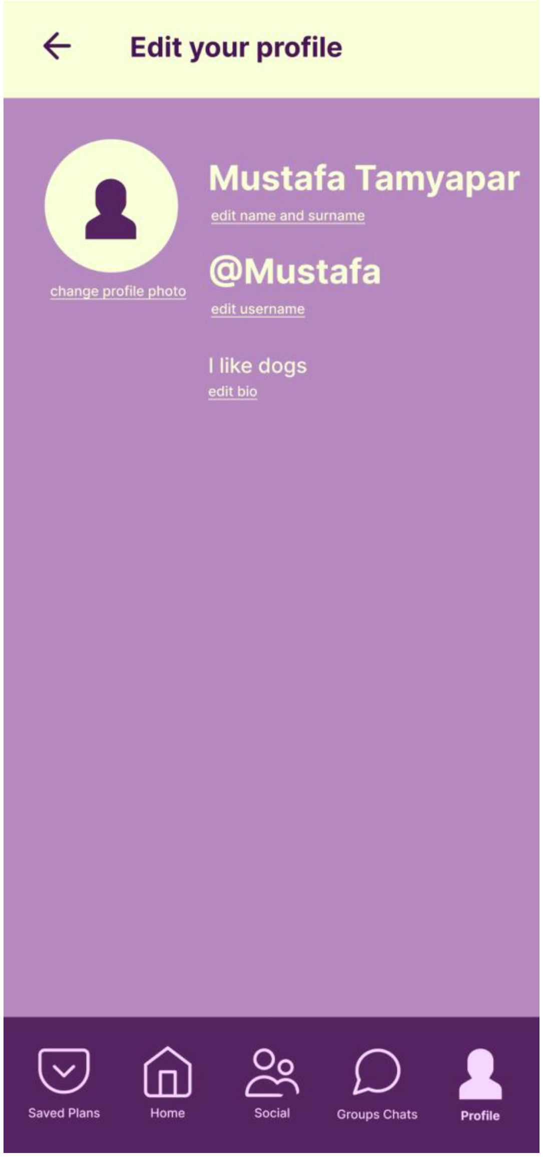
Select Currency

Turkish Lira ₺

Figure 24. Settings screen

In this settings screen, accessed by tapping the settings icon from the profile screen, users can change language and money currency.

3.5.5.14 Edit Profile screen



Edit your profile

Mustafa Tamyapar

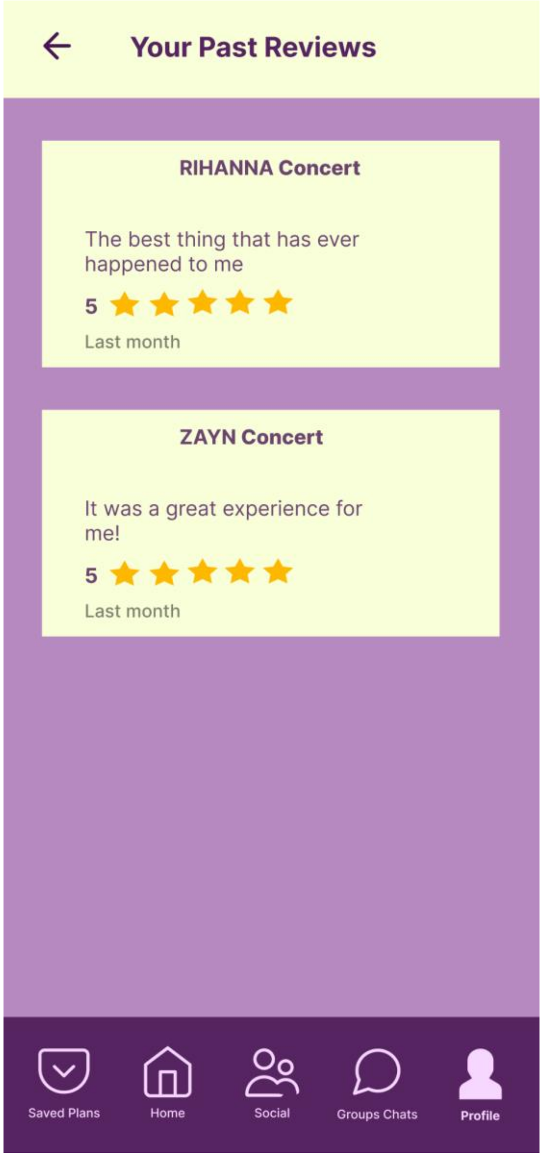
@Mustafa

I like dogs

Figure 25. Groups Invitation screen

This screen is accessed by clicking on the edit icon at the profile screen. At this screen, users can see all their profile information as well as edit them. They can change their profile photo, name and surname, username, and bio.

3.5.5.15 Your Past Reviews screen



RIHANNA Concert

The best thing that has ever happened to me

5 ★★★★★

Last month

ZAYN Concert

It was a great experience for me!

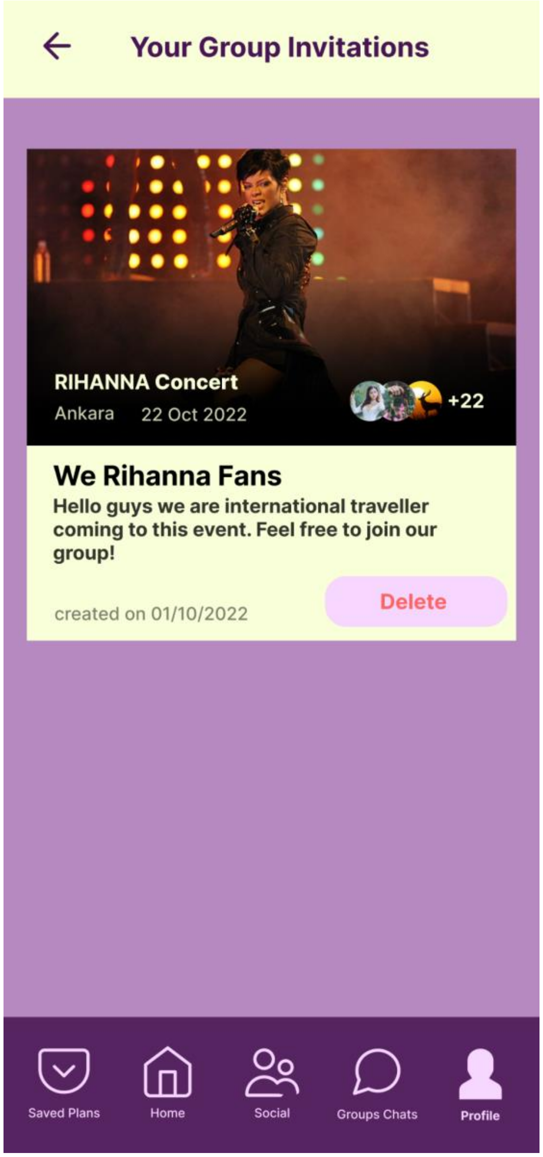
5 ★★★★★

Last month

Figure 26. Past Reviews screen

This screen is accessed by clicking on “Your Past Reviews” from the profile screen. In this screen users can see the reviews they have left to artists.

3.5.5.16 Your Group Invitations screen



Your Group Invitations

RIHANNA Concert

Ankara 22 Oct 2022

We Rihanna Fans

Hello guys we are international traveller coming to this event. Feel free to join our group!

created on 01/10/2022

Figure 27. Groups Invitation screen

This screen is accessed by clicking on “Your Group Invitations” from the profile screen. In this screen users can see the groups invitations they have created. They can also delete them if they desire.

3.5.5.17 Create Review Post screen

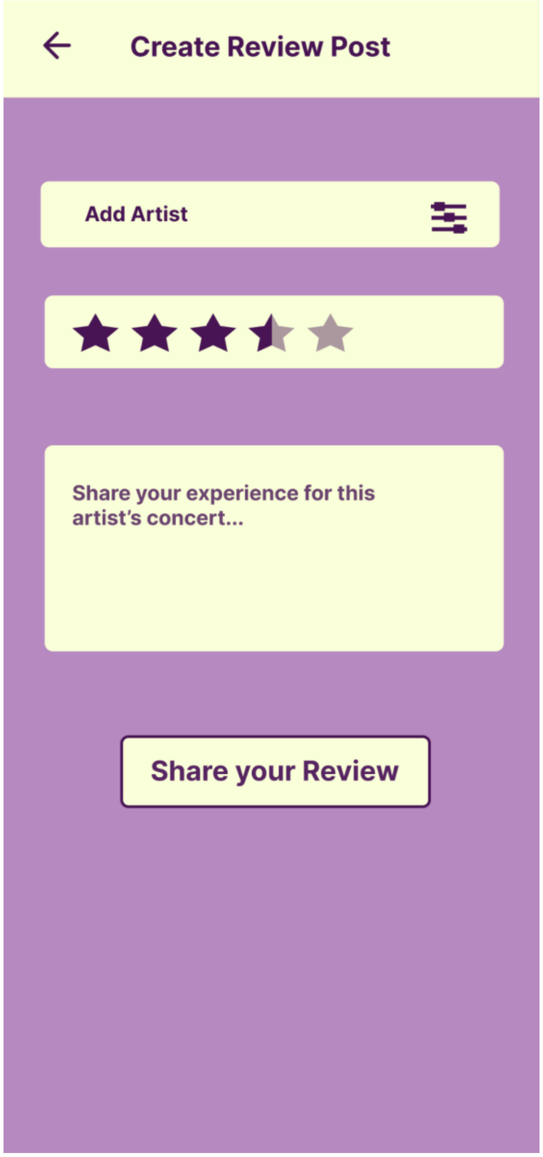
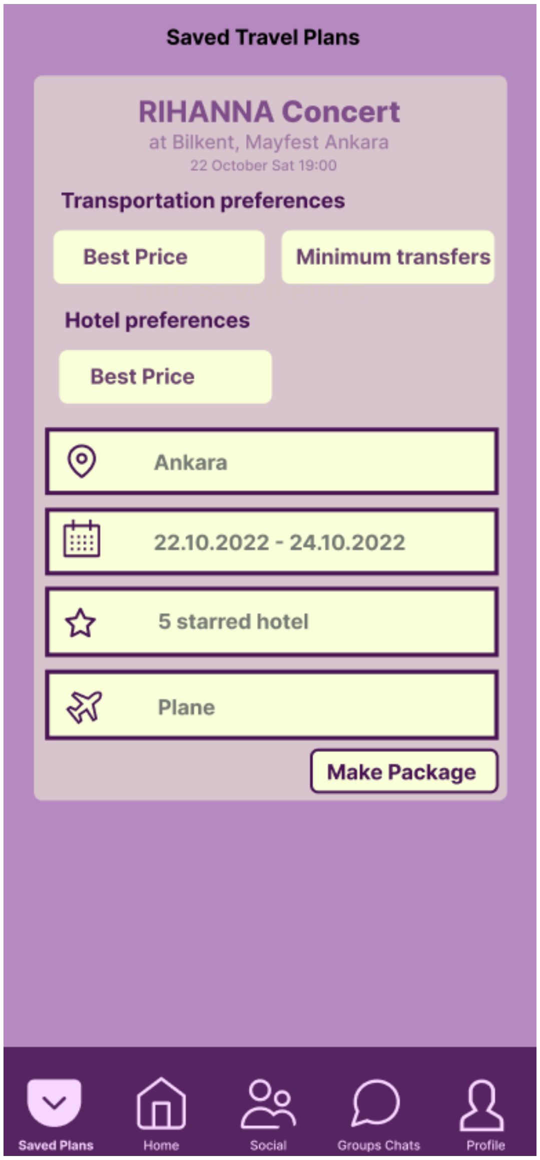


Figure 28. Create Review Post Screen

This screen is accessed by tapping on "Create Review Post" at the profile screen. Here, users can select an artist, select a star rate, type a review, and share it. This review post is displayed at the concert screen as well as at "Your Past Reviews" screen.

3.5.5.18 Saved Plans screen



Saved Travel Plans

RIHANNA Concert

at Bilkent, Mayfest Ankara

22 October Sat 19:00

Transportation preferences

Best Price

Minimum transfers

Hotel preferences

Best Price

Ankara

22.10.2022 - 24.10.2022

5 starred hotel

Plane

Figure 29. Saved Plans

A user can see all the plans they saved at any time from the saved plan screen. They can access this page by clicking the “Saved Plans” icon at the navigation bar. They can click on the plan and the system generates a new plane based on the saved preferences and the current available tickets and hotels. The user can ask for new suggestions based on saved preferences by clicking the make package button. Users will be redirected to the suggested travel plans page.

4 Other Analysis Elements

4.1 Consideration of Various Factors in Engineering Design

Funravel’s design decisions are shaped by various factors and some of those factors have aspects which could be affected by the application in turn. Those factors are public safety factors, global factors (specifically international entertainment industry), social factors, economical factors and environmental factors.

In terms of public safety, Funravel’s invitation and group chat features were evaluated more than the other functions since they create an environment for people to communicate. Considering this communication is almost always between strangers who want to get together at a concert event, it brought up topics of trust and genuinity. In the current design, there is a phone number authentication feature to somehow ensure the users are real people. However, it was decided that this issue cannot be handled completely without removing this communicative aspect completely, which would be a huge loss from the applications features. So, the current version is kept as it is with authentication but updated solutions could be added to the systems if any is developed.

In terms of global factors, the international entertainment industry seems to be a valid consideration since Funravel is aimed for users who are willing to travel abroad to attend concerts. It can be used as a tracker for international concerts and it is meant to motivate users to actually attend those concerts by providing them travel packages according to their needs. This way, sales could be increased.

In terms of social factors, the sense of community is important in Funravel. Its invitations and group chat features bring together music fans, and more specifically fans of individual artists. Users can meet new people and experience being a part of the community alongside other fellow members. In addition to this, user reviews are another aspect which strengthens the community aspect. Users share their experience and the artists they like (or possibly the artists they do not listen to) and these review posts are visible to other users. As music fans, they communicate through those posts as well.

In terms of economical factors, coming up with alternatives to sustain the application was a complicated process and the final decisions are not made since the application is still in development. The most likely approach seems to be commissions from the reservation websites which the users are redirected to. Another alternative is in-app advertisements in the form of promoted hotel and transportation suggestions. There are also possible costs for using some of the APIs needed to retrieve data and since they are not free to use. Regarding how the application affects economical factors, Funravel's option to create travel packages with the best possible price can be considered. This way, it was aimed to help users spend less when they attend these concerts.

In terms of environmental factors, transportation options (especially flights) are considered. Flights' carbon emission is high and it is expected that Funravel's users will mostly have to use planes for transportation. The other option currently offered in the application is transportation by bus. For this initial development stage, the two most common transportation alternatives are included but adding a variety of transportation options is decided as a priority in further development stages. In this current design, packages with minimum number of transfers in transportation and shortest travel duration could be generated by the application which would help keep the emission as low as possible.

Table 1: Factors that can affect analysis and design.

	Effect level	Effect
Public safety	3/10	Trust and genuinity issues when interacting with other users, possibly solved with authentication mechanism

Global factors	6/10	International entertainment industry (specifically music) is central to Funravel
Social factors	7/10	Funravel brings the sense of community through group chats and review posts which allow users to share experiences
Economical factors	6/10	There are several possibilities for funding the app and possible costs for using APIs. User-wise, having an option to have best price suggestions is helpful
Environmental factors	5/10	Transportations (especially flights) propose high levels of carbon emission. Currently, offering packages with the shortest travel duration and minimum number of transfers is the best practice.

4.2 Risks and Alternatives

The main purpose of Funravel is to optimize users' time by creating travel plans for concerts that they wish to attend. Creating true and efficient travel plans is therefore an essential part of the application. During that process, if a problem arises while getting data from the API's, it will cause an inconvenient travel plan for the user. To avoid that problem we should check the API request’s correctness and the correctness of the data fetched using double checking.

Secondly, the concert’s status can change, concerts can be canceled, the venue can change or the concert date can change. So the application should ensure that the concert, transportation, and hotel booking data is consistent and synchronized. We can use alternative tools to sync API to ensure that the data is accurate.

Furthermore, in order for the application to function properly, the application needs information about the users, such as the user's location, email, etc. It is possible that the personal data could be leaked, but we have to ensure that the possibility is as low as possible. If the personal data of the users are leaked the application’s reliability will decrease and user satisfaction will decrease as well. The plan is to minimize this risk by encrypting the personal data of users as much as possible.

Table 2: Risks

	Likelihood	Effect on the project	B Plan Summary
Fetching incorrect data from the APIs for event, accommodation and transportation.	4 /10	The Travel plans are not correctly created and redirecting users to wrong booking sites.	Ensuring the data is current and data is correct with checking requests.
Not being able to sync event status.	6 /10	The concert information and travel plans are not correctly displayed.	Usage of alternative tools to the sync of the APIs and data.
Possibility of leaking the personal data of users.	3 /10	The reliability of the app will decrease which will decrease the customer satisfaction of the application.	Encrypting the personal data of the users.

4.3 Project Plan

The Project plan of the project is explained with work packages, which are shown below. The work package represents a big part of the project which requires for the project to funcion. The packages are distributed along with the member’s interest and previous experience. The Gannt chart for better understanding is presented in the next section.

These are the project goals:

Project Goals:

1. Implementing a successful web crawler if the API’s for the necessary data is not enough.
2. Fetching and synchronizing necessary data for the travel plan, concerts , hotel data, transportation data.
3. Creating travel plans according to the user's needs and displaying the data needed.
4. Implementing group chats and invitations for the user’s communication. Also implementing necessary database for this feature.
5. Implementing review posts and necessary databases.
6. Developing a unique, enjoyable, user interface.

Below are the current work packages. The start and end dates for each package are provided as well, but those dates are tentative. There might be changes according to academic calendar and deadline specifications.

Work Packages:

Table 3: List of work packages

WP#	Work package title	Leader	Members involved
WP1	Set ups and User Interface	Hissam Faramawy	Sena Genç, Öykü Erhan, Mustafa Efe Tamyapar, Gülin Yılmaz
WP2	Web Crawlers and APIs	Hissam Faramawy	Öykü Erhan , Gülin Yılmaz
WP3	Concert Data and Synchronisation	Mustafa Efe Tamyapar	Gülin Yılmaz, Hissam Faramawy
WP4	Detailed Design Report	Sena Genç	Hissam Faramawy, Öykü Erhan, Mustafa Efe Tamyapar, Gülin Yılmaz
WP5	Transportation Data	Öykü Erhan	Gülin Yılmaz, Sena Genç
WP6	Hotel Data	Gülin Yılmaz	Öykü Erhan, Sena Genç
WP7	Generating Travel Plan	Mustafa Efe Tamyapar	Hissam Faramawy, Öykü Erhan, Sena Genç, Gülin Yılmaz
WP8	Saved Plans	Mustafa Efe Tamyapar	Hissam Faramawy, Sena Genç

WP9	Group Invitations	Öykü Erhan	Gülin Yılmaz
WP10	Group Chats	Öykü Erhan	Gülin Yılmaz
WP11	Review posts	Gülin Yılmaz	Öykü Erhan, Mustafa Efe Tamyapar
WP12	Final Report	Sena Genç	Hissam Faramawy, Öykü Erhan, Mustafa Efe Tamyapar, Gülin Yılmaz
WP13	Enhancements and Testing	Gülin Yılmaz	Öykü Erhan, Mustafa Efe Tamyapar

Table 4: Work Package 1 describing set ups and user interface

WP 1: Set ups and User Interface			
Start date: *14.11.2022* **End date:** *27.11.2022*			
Leader:	*Hissam Faramawy*	**Members involved:**	*Sena Genç, Öykü Erhan, Mustafa Efe Tamyapar, Gülin Yılmaz*

Objectives: *In this work package we aim to develop the user interface of the application, set up the database and set up the project in flutter.This UI is not the enhanced UI, it is for the first demo*

Tasks:

Task 1.1 Setting up project in the flutter**:** In this task the project is setted up in flutter in all of the members computers and necessary connections are made.

Task 1.5 Creating database in the firebase**:** *The firebase database is created in this task.*

Task 1.2 Development of the log-in/sign-up and home screens**:** The log-in, sign-up and home screen is developed in this task.

Task 1.3 Development of the travel package creation screen**:** *Travel package creation page is made in this task.*

Task 1.4 Development of the saved plans, group chats and invitations and profile screens**:** *Saved plans, group chats, profile and invitations screens are made in this task.*

Deliverables

D1.1: *Log-in/Sign-up screens*

D1.2: *Home screen*

D1.3: *Saved plans screen*

D1.4: *Group chats screen*

D1.5: *Invitations screen*

D1.6: *Travel package creation screen*

D1.7: *Profile screen*

D1.8: *The database and project setups*

Table 5: Work Package 2 describing Web Crawlers and APIs

WP 2: Web Crawlers and APIs			
Start date: *28.11.2022* **End date:** 25.12.2022			
Leader:	*Hissam Faramawy*	**Members involved:**	*Öykü Erhan , Gülin Yılmaz*
Objectives: *In this work package we aim to find useful APIs and Web Crawlers that we can use for fetching our accommodation and transportation data. If we cannot find free or usable ones then we will be implementing our own web crawler.*			
Tasks: ***Task 1.1 Finding useful APIs :*** *In this task, we will try to find useful APIs we can make use of in our application. There are already some APIs we found that we mentioned in the reports but we will do more detailed research.* ***Task 1.2 Fetching data from APIs*** **:** *We will try to fetch the data from the APIs we found in Task 1.1 and see if we are able to obtain the useful information we need.* ***Task 1.3 Finding Web Crawlers*** **:** *We will try to find web crawlers to be able to use the data in the reservation websites.* ***Task 1.3 Creating Web Crawlers*** **:** *We need to implement our own web crawlers if we cannot make use of the web crawlers we found online.*			

Deliverables

D1.1: *Reservation APIs*

D1.2: *Data obtained from APIs*

D1.3: *Web crawlers to fetch data from reservation websites*

D1.4: *Newly implemented web crawlers*

Table 6: Work Package 3 describing Concert Data and Synchronisation

WP 3: Concert Data and Synchronisation			
Start date: *28.11.2022* **End date:** *25.12.2022*			
Leader:	*Mustafa Efe Tamyapar*	**Members involved:**	*Gülin Yılmaz, Hissam Faramawy*
Objectives: *Fetching the concert data correctly and synchronizing with the database.*			
Tasks: ***Task 1.1 Creating necessary data tables in the database :*** *In this task, the data tables specific to concerts is created* ***Task 1.3 Using web crawlers or API for the database and connecting them*****:** *In this task, web crawler or UI connection with database is made.* ***Task 1.4 Connecting UI requests with the API or web crawlers*****:** *The request from the UI and the API or web crawlers are made in this task.*			

Task 1.5 Synchronizing the database with the API or web crawlers : *The updating of the data tables with api and web crawlers are made in this task.* ***Task 1.6 Displaying the Data in the UI:*** *Displaying the requests to the UI is made in this task.*
Deliverables ***D1.1:*** *Database tables for the concerts* ***D1.2:*** *Web crawler or API connections with the database* ***D1.3:*** *UI request connections with the API or web crawler* ***D1.3:*** *Synchronization of the database tables*

Table 7: Work Package 4 describing Detailed Design Report

WP 4: Detailed Design Report			
Start date: *26.12.2022* **End date:** *29.01.2023*			
Leader:	*Sena Genç*	**Members involved:**	*Hissam Faramawy, Öykü Erhan, Mustafa Efe Tamyapar, Gülin Yılmaz*
Objectives: *Making a detailed design report including high and low levels of the project such as classes, packages, system architecture.*			

Tasks:

Task 1.1 Introduction

Task 1.2 Current Software Architecture

Task 1.3 Proposed Software Architecture

Task 1.4 Subsystem Services

Task 1.5 Consideration of Various Factors in Engineering Design

Task 1.6 Packages

Task 1.7 Class Interfaces

Task 1.8 Teamwork Details

Task 1.9 Glossary

Task 1.10 References

Deliverables

D1.1: *Detailed Design Report*

Table 8: Work Package 5 describing Transportation Data

<table>
<tr><td colspan="4">**WP 5:** Transportation Data</td></tr>
<tr><td colspan="4">**Start date:** *09.01.2023* **End date:** *05.02.2023*</td></tr>
<tr><td>**Leader:**</td><td>*Öykü Erhan*</td><td>**Members involved:**</td><td>*Gülin Yılmaz, Sena Genç*</td></tr>
<tr><td colspan="4">**Objectives:** *Fetching the transportation data correctly and synchronizing with the database.*</td></tr>
<tr><td colspan="4">**Tasks:**
Task 1.1 Creating necessary data tables in the database : *In this task, the data tables specific to transportation is created*
Task 1.3 Using web crawlers or API for the database and connecting them: *In this task, web crawler or UI connection with database is made.*
Task 1.4 Connecting UI requests with the API or web crawlers: *The request from the ui and the api or web crawlers are made in this task.*
Task 1.5 Synchronizing the database with the API or web crawlers : *The updating of the transportation data tables with api and web crawlers are made in this task.*
Task 1.6 Displaying the Data in the UI: *Displaying the requests to the UI is made in this task.*</td></tr>
</table>

Deliverables

D1.1: *Database tables for the transportation*

D1.2: *Web crawler or API connections with the database*

D1.3: *UI request connections with the API or web crawler*

D1.3: *Synchronization of the database tables*

Table 9: Work Package 6 describing Hotel Data

WP 6: Hotel Data			
Start date: *09.01.2023* **End date:** *05.02.2023*			
Leader:	*Gülin Yılmaz*	**Members involved:**	*Öykü Erhan, Sena Genç*
Objectives: *Fetching the hotel data correctly and synchronizing with the database.*			
Tasks: ***Task 1.1 Creating necessary data tables in the database :*** *In this task, the data tables specific to hotels is created* ***Task 1.3 Using web crawlers or API for the database and connecting them:*** *In this task, web crawler or UI connection with database is made.* ***Task 1.4 Connecting UI requests with the API or web crawlers:*** *The request from the ui and the api or web crawlers are made in this task.*			

Task 1.5 Synchronizing the database with the API or web crawlers : *The updating of the data tables with api and web crawlers are made in this task.* ***Task 1.6 Displaying the Data in the UI:*** *Displaying the requests to the UI is made in this task.*
Deliverables ***D1.1:*** *Database tables for the hotels.* ***D1.2:*** *Web crawler or API connections with the database* ***D1.3:*** UI request connections with the API or web crawler ***D1.3:*** Synchronization of the database tables

Table 10: Work Package 7 describing Generating Travel Plan

WP 7: Generating Travel Plan			
Start date: *06.02.2023* **End date:** *05.03.2023*			
Leader:	*Mustafa Efe Tamyapar*	**Members involved:**	*Hissam Faramawy, Öykü Erhan, Sena Genç, Gülin Yılmaz*
Objectives: The objective for this work package is implementing the main functionality of Funravel which is creating travel plans for the concerts users want to attend. We aim to provide them plans based on their preferences			

Tasks:

Task 1.1 Obtaining necessary preferences of users**:** *We will ask the users some necessary information to be able to create a plan for them such as their starting location, preferred transportation type, preferred star rating of the hotels and the date range in which they want to stay in the city they will attend the concert in.*

Task 1.2 Obtaining optional preferences of users **:** *We also ask the users if they want to apply optional filters which are shortest travel time, shortest distance to event location, best price, minimum transfer and will generate a plan based on all the preferences we obtain.*

Task 1.3 Finding suitable hotel and transportation: We will find the optimal travel plan using the data we fetch and the preferences we have of the user.

Deliverables

D1.1: Necessary user preferences

D1.2: Optional user preferences

D1.3: Optimal travel plan

Table 11: Work Package 8 describing Saved Plans

WP 8: Saved Plans			
Start date: *06.02.2023* **End date:** *05.03.2023*			
Leader:	*Mustafa Efe Tamyapar*	**Members involved:**	*Hissam Faramawy, Sena Genç*
Objectives: *We aim to let the users save their past preferences for a specific concert. When they navigate to the saved plans page they will be able to see their past preferences and load a package based on these again.*			
Tasks: ***Task 1.1 Saving past preferences:*** We will save the preferences of the user for a specific plan and display it on the saved plans page. ***Task 1.2 Display plan based on past preferences:*** When the user navigates to save plans page they can click on the make package button and a plan based on the past preferences will be created.			
Deliverables ***D1.1:*** *Data for past preferences and saved plans page* ***D1.2:*** *A plan based on past preferences of the user for a specific concert.*			

Table 12: Work Package 9 describing Invitations

WP 9: Invitations			
Start date: *20.02.2023* **End date:** *19.03.2023*			
Leader:	*Öykü Erhan*	**Members involved:**	*Gülin Yılmaz*
Objectives: *Implementation of the invitations which are used for inviting other users to the group chats. Databases for the invitations are created as well.*			
Tasks: ***Task 1.1 Creating the invitation objects and connecting with invitations page:*** *In this task the invitation object is created and connected with the invitations page where these objects are shown.* ***Task 1.2 Creating the database for the invitations:*** *The data tables for the invitations are created in this task.* ***Task 1.3 Showing the invitations in the User interface:*** *Fetching data from the database and showing to users the invitations are made in this task.*			
Deliverables ***D1.1:*** *Invitation object* ***D1.2:*** *Invitations page connection* ***D1.3:*** *Data tables for the invitations*			

Table 13: Work Package 10 describing Group Chats

WP 10: Group Chats			
Start date: *20.02.2023* **End date:** *19.03.2023*			
Leader:	*Öykü Erhan*	**Members involved:**	*Gülin Yılmaz*
Objectives: *Implementation of the group chats which are used for communication between users. The group chat functionalities are made in this package. Databases for the invitations are created as well.*			
Tasks: ***Task 1.1 Creating group chat objects*****:** In this task group chat objects are created. ***Task 1.2 Creating chat functionalities of the group chats*****:** *In this task chat functionalities such as sending and receiving messages are created.* ***Task 1.3 Connecting invitations with the group chats****: In this task invitations of each group chat are connected with that group chat. Also adding and removing a participant is made in this part.* ***Task 1.4 Creating database for the group chat****: In this task, the database for the data that need to be stored for correctly working groupchat is created and connected with the group chat.* ***Task 1.5 Connecting all group chat data with database and synchronization of the data:*** *In this task the group chats are connected and synchronized with the database. The requests for messaging features and other user interface requests like user adding to the group chats are made in this task.*			

Deliverables

D1.1: *Group chat object*

D1.2: *Messaging features of the group chats*

D1.3: *Connection of group chats with their invitations*

D1.4: *Database tables for the group chats*

D1.5: *Synchronization of the data for the group chats*

Table 14: Work Package 11 describing Review Posts

WP 11: Review Posts			
Start date: *06.02.2023* **End date:** *05.03.2023*			
Leader:	*Gülin Yılmaz*	**Members involved:**	*Öykü Erhan, Mustafa Efe Tamyapar*
Objectives: *Implementation of users' ability to review and rate old concerts based on artist names. Database section will be created for keeping reviews and rates. These posts will be linked to users and artist names.*			
Tasks: ***Task 1.1 Data tables creation for reviews*** **:** *Database portion for reviews and rates is created within Firebase and has artist and user name attributes for start.*			

Task 1.2 Creating the review post objects and connecting with reviews page: *Review post object is created and connected with the invitations page where these objects are shown.* ***Task 1.3 Showing the review in the UI page:*** *Fetching data from the database and showing to users the reviews on profile and concert page.*
Deliverables ***D1.1:*** *Data tables for review & rates of users based on artist names* ***D1.2:*** *Review object*

Table 15: Work Package 12 describing Final Report

<table>
<tr><td colspan="4"><b>WP 12:</b> Final Report</td></tr>
<tr><td colspan="4"><b>Start date:</b> <i>06.03.2023</i> <b>End date:</b> <i>14.05.2023</i></td></tr>
<tr><td><b>Leader:</b></td><td><i>Sena Genç</i></td><td><b>Members involved:</b></td><td><i>Hissam Faramawy, Öykü Erhan, Mustafa Efe Tamyapar, Gülin Yılmaz</i></td></tr>
<tr><td colspan="4"><b>Objectives:</b> <i>Making a final report including final status and future works of the project.</i></td></tr>
</table>

Tasks:

Task 1.1 Introduction

Task 1.2 Requirements Details

Task 1.3 Final Architecture and Design Details

Task 1.4 Development/Implementation Details

Task 1.5 Testing Details

Task 1.6 Maintenance Plan and Details

Task 1.7 Other Project Elements

Task 1.8 Conclusion and Future Work

Task 1.9 Glossary

Task 1.10 References

Deliverables

D1.1: *Final Report*

Table 16: Work Package 13 describing Enhancements & Testing

WP 13: Enhancements & Testing			
Start date: *20.04.2023* **End date:** *14.05.2023*			
Leader:	*Gülin Yılmaz*	**Members involved:**	Öykü Erhan, Mustafa Efe Tamyapar
Objectives: *After the main functionalities are finished, bug fixing will start. Unit testing will be done according to the test cases. If errors occur, related classes will be looked at and errors are going to be fixed.*			
Tasks: ***Task 1.1 Creating test cases*** **:** *Test cases for each scenario will be created.* ***Task 1.2 Testing scenarios*** **:** *Created test cases will be tested with emulator and physical phones.*			
Deliverables ***D1.1:*** *Test scenarios* ***D1.2:*** *Bug fixing*			

Below is the Gantt chart (tentative) for the project plan.

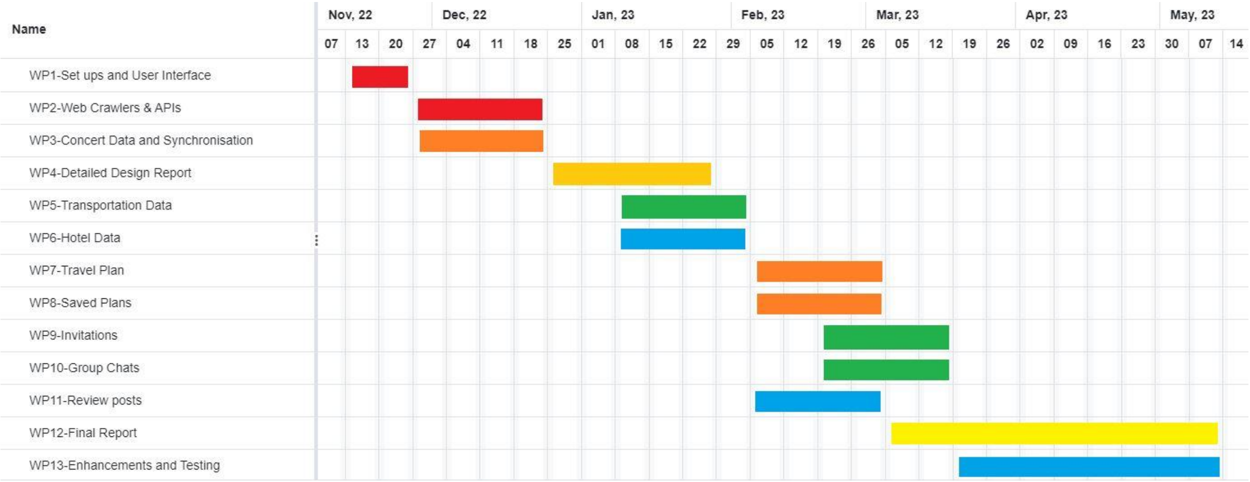


Figure 30. Gannt Chart

Each color corresponds to the leader of that package: Hissam Faramawy, Mustafa Efe Tamyapar, Fatma Sena Genç, Öykü Erhan, Gülin Yılmaz

4.4 Ensuring Proper Teamwork

Our team has a shared Trello board on which we are planning to put up and track tasks. Each task will be assigned to a member and as they progress, they can update the current status of the task. This way, we all are able to see the progress of each other and we can keep each other accountable. We will also have our weekly meetings which we are already doing, to discuss our progress and share outcomes. Our meetings are logged.

4.5 Ethics and Professional Responsibilities

Funravel utilizes users' information and preferences, integrates social environments, and provides a platform for those who want to find people to attend events with. These features might bring ethical issues that needed to be handled. Privacy should be protected and the safety of users should be managed in an efficient way with the available tools. These features also must follow the local Turkish laws as well as the General Data Protection Regulation, GDPR [4].

4.6 Planning for New Knowledge and Learning Strategies

For the purpose of planning the new knowledge that we have acquired, we have divided the learning areas into two parts. The first part is learning the flutter and dart language. Even though some group members have experience in android and ios mobile development using native languages only one of us uses Flutter. To learn mobile development with Dart and Flutter, we are taking online courses.

Second, APIs can create some issues, such as being a trial version or being expensive, when fetching necessary data from sources via APIs. We need to learn about web crawlers for fetching the necessary data for our application. We are planning to use online resources to learn about web crawlers and their usage.

Every group member has experience of mobile development and has different perspectives on the topics we plan to implement. Accordingly, we will follow different strategies. Members with more experience will assist others during the learning phase.

5 Glossary

API (Application Programming Interface): Mechanisms that allow two software components to communicate with one another using a set of definitions and protocols.

Dart: Open source general purpose programming language designed by Google.

Flutter: Open source user interface and software development kit designed by Google.

Firebase: Application development platform for developing Android, IOS and web applications, developed by Google.

SDK (Software Development Kit): A set of software-developing tools for a specific platform.

6 References

[1] Ticketmaster, “Concert Road Trips: How Far Do Fans Go to see live music?,” *Ticketmaster Blog*, 16-Apr-2020. [Online]. Available: https://blog.ticketmaster.com/concert-road-trips/. [Accessed: 12-Nov-2022].

[2] R. Knight, “Average person spends 10 hours planning their holiday, survey claims,” *The Independent*, 28-Feb-2019. [Online]. Available: https://www.independent.co.uk/travel/holiday-booking-planning-travel-survey-tourist-a8801211.html. [Accessed: 12-Nov-2022].

[3] Rapid, *The next generation API platform*. 10-November-2022. [Online]. Available: https://rapidapi.com/ . [Accessed: 12-Nov-2022].

[4] The European Parliament and the Council of the European Union, “General Data Protection Regulation,” 27-Sep-2022. [Online]. Available: https://gdpr-info.eu/. [Accessed: 12-Nov-2022].