Travlendar+
Requirement Analysis and Specification Document

Fumagalli Paolo, Grotti Pietro, Gullo Marco

October 28, 2017

Contents

1 Introduction

1.1 Purpose

Many endeavors require scheduling meetings at various locations all across a city, whether in support of a mobile job or a busy parent. The goal of this project is to create a calendar interface that automatically computes and accounts for travel time between appointments to make sure that you're never late for an appointment. The application will also suggest travel means by appointment (e.g., perhaps you drive to the office in the morning but the bus is a better choice between a pair of afternoon meetings) and by day (e.g. working days or weekends, traffic, public transport strikes, weather). The application will support an user interface where complete and automatically well-fitted schedules can be made. System will alert for any appointment overlap and non-doable consecutive appointments (e.g. two meetings really close in time but in locations too far from each other). Furthermore, different preferences, such as travel options and break time, can be expressed by the user.

Goals

The application has the following goals:

G1: Allow a guest user to register to Travlendar+ by filling the registration form with the data needed.

G2: Allow the user to select preferences and modify them whenever he wants.

G3: Allow the user to easily create an organized and customizable agenda based on his preferences.

G4: To help the user plan his movements in a clever and efficient way.

G5: To guarantee the user no to be late for his appointments.

- **G5.1:** The system will take into account the possibility of accidents and plan for the user to arrive at least 10 minutes before the beginning of the event.
- **G5.2:** The user will be notified if an appointment is not reachable in time.

G6: To let the users buy bus or train tickets (both single or seasonal).

G7: To let the user find vehicles from vehicle sharing systems.

G8: Allow the user to buy in advance tickets which can be used later on.

1.2 Scope

Travlendar+ is an application which will be able to manage daily appointments of the users and assist them by creating a specific course around the city which will identify the best mobility option to move from one appointment to the other. It will consider all public transportation options (train, metro, bus, etc.), car and bike sharing systems, the eventuality of a private vehicle and the weather conditions as well, to avoid having the user bike in harsh weather conditions or when it is too hot. It will also be possible to buy tickets and locate cars and bikes directly through the application. Users will have to create meetings specifying the location, the date and the time and the application will automatically calculate the suggested course and travel time, if it is impossible to reach a certain meeting in time the application will send a warning to the user. It will also feature the possibility of selecting a time span in which it will have to save some time for a break in which the user can have lunch (for example if the user wants to have lunch between 11:30 to 2:30 and it should be at least half an hour long, the application will reserve at least 30 minutes for lunch every day). Travlendar+ will also give the possibility to the users to buy tickets, this will make it easier to move with public transportation around the city. To allow these transactions the system will have to work with a bank which will support the possibility of either using a credit/debit card to buy tickets or creating a balance that will be connected to the account. The balance can be charged by credit/debit card or through selected shops which will allow the acquisition of cards containing unique codes, which will load money on the balance.

1.3 Definitions, Acronyms, Abbreviations

1.3.1 Definitions

1.3.2 Abbreviations

G* : Specific goal
R* : Specific functional requirements
D* : Specific domain assumption
App : Application

1.3.3 Acronyms

ETA : Estimated Time of Arrival
API : Application Programming Interface
PTS : Public Transportation System
CSS : Car-Sharing System
BSS : Bike-Sharing System

1.4 Revision history

Version 1.0.0

1.5 Reference Documents

1.6 Document Structure

2 Overall Description

2.1 Product perspective

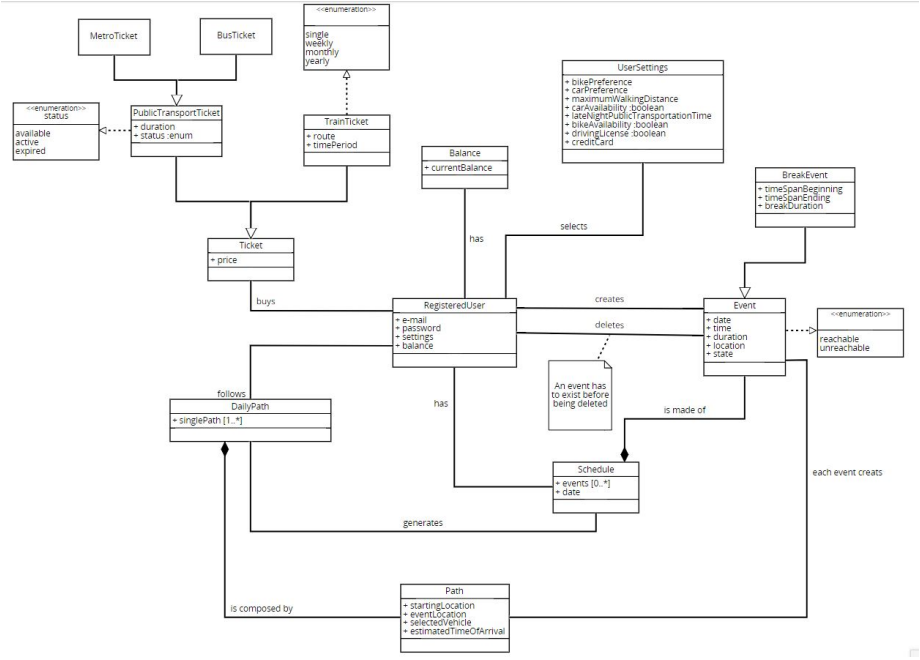

Figure 1: Class Diagram

The products main purpose is to fit in users everyday reality, dealing with a large number of external factors and environments, such as weather conditions, traffic, news, etc. In order to face up all these agents, the system will always cooperate with other existing applications and ask for external help. Travlendar+ will work as an agent-in-the-middle between user and different interfaces, putting together all data required for users planning and traveling needs. All the various interactions will be described in high-level details in UML Section (see).

2.2 Product functions

All main functions will be described in following sections (see) and are the ones that will cover all project goals provided above(see).

2.3 User characteristics

Actors:

- **Guest User:** unregistered customer, he has just downloaded the app or visited the website. He wants to try Travlendar+ and needs to find a user-friendly interface.
- **Registered User:** more familiar with the environment, can access all Travlendar+ functionalities and customize his preferences. Expects an efficient service.
- **Public Transportation System:** local public transportation business, is willing to cooperate with the app in order to increase tickets sales and reduce irregular PTS use. Online purchases may also decrease the costs of printed tickets.
- **Car-sharing System:** local private enterprise, is interested in Travlendar+ development to enhance their business reach and visibility.
- **Bike-sharing System:** same as car-sharing system.
- **Google Maps:** is a web mapping service developed by Google. It offers satellite imagery, street maps, 360 panoramic views of streets, real-time traffic conditions, and route planning for traveling by foot, car or public transportation. It is completely free-to-use (its API can be integrated easily in any system) for any system. Travlendar+ will benefit of this.
- **Bank:** it is the entity necessary for Travlendar+ to deal with money transactions (ticket and pass purchases). The bank surely has its own secure channels and count on integrity of data coming from the apps system.

2.4 Assumptions, dependencies and constraints

The following assumptions are to be considered true in the world that we will analyize:

D1: The city is covered by several public and private transport services.

D2: The services are almost always available, and when they arent theres an alternative which can be suggested by Travlendar+ to the user.

D3: A database containing all the correct information needed for transport services (location of the vehicles or the stationts, time tables, etc.) is connected to Travlendar+.

D4: An application able to calculate travel times and distances which can also access current traffic information is connected to Travlendar+.

D5: Some services allow external applications to interact with them to take advantage of certain functions (e.g.: to buy bus tickets from the PTS or to find a car of a BSS) in a transparent way from the user's point of view.

D6. An internet connection is available everywhere and at any given time in the area covered by Travlendar+.

D7: Users of the application have a working GPS that can accurately calculate their position.

D8: The public transportation companies agree to let the application developers sell tickets through Travlendar+.

D9: The transactions are handled through a bank that agrees to work with Travlendar+.

D10: Bikes and cars of sharing-systems are equipped with an accurate GPS that can be traced in the map.

D11: Vehicles sharing systems agree to work with Travlendar+.

D12: "Activated" tickets are accepted by public transportation companies.

3 Specific Requirements

3.1 External Interface Requirements

3.1.1 User Interfaces

In the following section we will show some mock-ups of how the application will look like from the user's point of view.

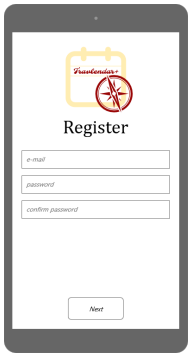

Figure 2: Account Creation Mock-Up

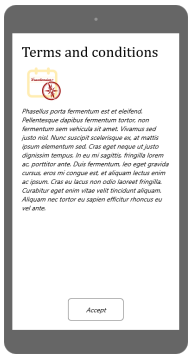

Figure 3: Terms And Conditions Mock-Up

Figure 4: Login Mock-Up

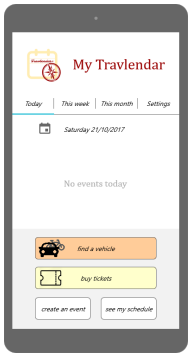

Figure 5: Home Page Mock-Up

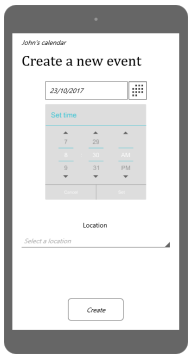

Figure 6: Create Event Mock-Up

Figure 7: Summary Mock-Up

Figure 8: Schedule Mock-Up

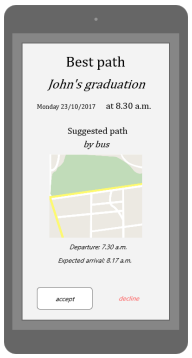

Figure 9: Path Mock-Up

Figure 10: Unreachable Path Mock-Up

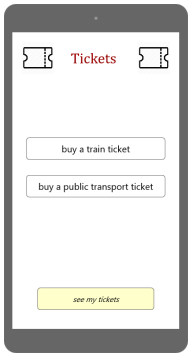

Figure 11: Ticket Store Mock-Up

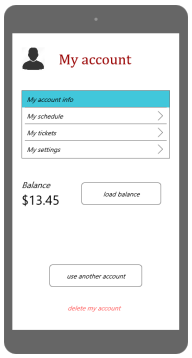

Figure 12: Personal Page Mock-Up

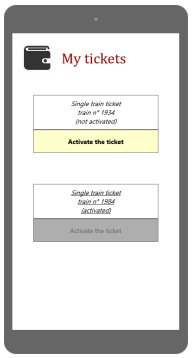

Figure 13: My Tickets Mock-Up

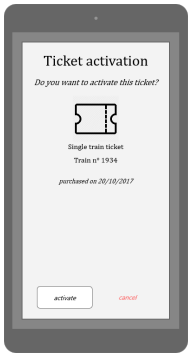

Figure 14: Ticket Activation Mock-Up

Figure 15: Find a Vehicle Mock-Up

Figure 16: Balance Mock-Up

Figure 17: Balance Loading Mock-Up

3.1.2 Hardware Interfaces

3.1.3 Software Interfaces

3.1.4 Communication Interfaces

3.2 Functional Requirements

In this section we will define the functional requirements combined with the domain assumptions that are needed to reach the goals set in the section 1.1.

G1: Allow a guest user to register to Travlendar+ by filling the registration form with the data needed.

- **R1:** A guest user can create a new account through the registration process by providing his credentials to the system. An e-mail will be necessary and it must be unique.
- **R2:** At the end of the registration process the system will send an e-mail to the user to verify the account.
- **R3:** The account must be verified with the link provided by the system before it becomes active.
- **R4:** A guest user can log in the application with his credentials.

G2: Allow the user to select preferences and modify them whenever he wants.

- **R5:** A registered user can modify his preferences by choosing the preferred movement system at any time.
- **R6:** A registered user can select a specific time during the day (or night) after which he doesn't want to take public transportation.
- **R7:** A registered user can choose a maximum distance that he is willing to walk to move from one place to the other.

G3: Allow the user to easily create an organized and customizable agenda based on his preferences.

- **R8:** A registered user can create an event by filling the time, date, duration and location of the event.
- **R9:** A registered user can select a time slot during the day in which he wants to take a break, he can also choose the length of the break (which doesn't have to necessarily be equal to the time slot).

G4: To help the user plan his movements in a clever and efficient way.

- **R10:** The user has to declare if he has a driving license.
- **R11:** The user can declare if he owns a personal vehicle and indicate which kind of vehicle (car, motorbike, bike).
- **R12:** The system has to calculate a route to move araound the city based on the user's preferences.

R13: The system will have to adjust accordingly to unexpected events (accidents, strikes, weather, natural disasters) changing the route of the day and notifying the user.

D1: The city is covered by several public and private transport services..

D2: The services are almost always available, and when they arent theres an alternative which can be suggested by Travlendar+ to the user.

D3: A database containing all the correct information needed for transport services (location of the vehicles or the stationts, time tables, etc.) is connected to Travlendar+.

D4: An application able to calculate travel times and distances which can also access current traffic information is connected to Travlendar+.

G5: To guarantee the user no to be late for his appointments.

R14: The system will always try to have the ETA to an event at least 10 minutes earlier than the beginning of the event.

R15: The system will notify the user if one appointment is not reachable, this feature has to update the user in real time, which means that if an event becomes unreachable it will notify the user immediately.

D4: An application able to calculate travel times and distances which can also access current traffic information is connected to Travlendar+.

D7: Users of the application have a working GPS that can accurately calculate their position.

G6: To let the users buy bus or train tickets (both single or seasonal).

R16: The system will support the acquisition of valid tickets for public transportation.

R17: The user can link h is account to a credit card to pay for tickets.

R18: The user can open a balance connected to his account that can be used to pay for tickets.

R19: The balance can be charged with credit cards or with cash, by going to shops that are certified by the application.

R20: After being bought a ticket will be flagged as "Available".

D8: The public transportation companies agree to let the application developers sell tickets through Travlendar+.

D9: The transactions are handled through a bank that agrees to work with Travlendar+.

G7: To let the user find vehicles from vehicle sharing systems.

R21: The system must be able to locate vehicles from vehicle sharing systems.

D10: Bikes and cars of sharing-systems are equipped with an accurate GPS that can be traced in the map.

D11: Vehicles sharing systems agree to work with Travlendar+.

G8: Allow the user to buy in advance tickets which can be used later on.

R21: An "Available" ticket can be activated.

R22: After a ticket is activated it will be flagged as "Active".

R23: Once the time expires on an "Active" ticket it will be flagged as "Expired".

R24: An "Expired" ticket can't be activated.

D12: "Activated" tickets are accepted by public transportation companies.

3.3 Design Constraints

3.3.1 Regulatory Policy

Any sensible data, such as personal and payment info, will be stored with the purpose of ensure the best service and will not be handed only to third parties. This agreement will be described in better details in the Terms and Conditions the user will accept in the registration process.

3.3.2 Hardware Limitations

Main limitations will come from the mobile version of Travlendar+, which means:

- No Windows Phone OS-supporting version
- 3G/4G/LTE connection
- GPS accuracy

3.3.3 Interfaces With Other Applications

As written above in Product Perspective (section 2.1), the system will be developed to cooperate and interact dynamically with different external applications, so it must be flexible and open to new collaborations with new services in future. Developers will make sure to choose the appropriate middleware for such aim.

3.4 Non-Functional Requirements

3.4.1 Extensibility

Since Travlendar+ will be developed from scratch, it will be first launched with only the essential characteristics, its core goals, but is meant to be open to future additional features and improvements. Therefore, the system shall

provide an API, a way for programmers to create own application which can interact with the systems data and functionalities. This choice implies a more complex security system, in order to avoid developers to access sensitive users data. More specific information will be given in the design document.

3.4.2 Performance

The aim of Travlendar+ is to become a large-scale used application. The system must be able to handle a great number of simultaneous requests and respond correctly in the best time possible.

3.4.3 Reliablity

System must work ideally 24/7, but a short period of time should be dedicated to daily maintenance.

3.4.4 Security

Users credentials, personal information, payment data will be stored and must be protected.

3.4.5 Accuracy

System must be accurate in computing routes and in estimating time necessary of any movement from a place to another. For this purposes very precise maps must be used, the system also must take account of GPS accuracy (GPS technology is theoretically able to provide locations with an error less or equal to 7.8 meters).

3.4.6 Interoperability

System must interact efficiently with other applications, such as vehicle sharing apps, to provide a complete service. Travlendar+ will be also mediator in purchases, so needs to be connected to one or more bank/payment system.

4 UML Diagrams, Use Cases, Statecharts And Sequence Diagrams

4.1 Use Case Diagram

We defined the Use Case Diagrams for the actors as Guest Users and Registered User.

4.1.1 Guest User

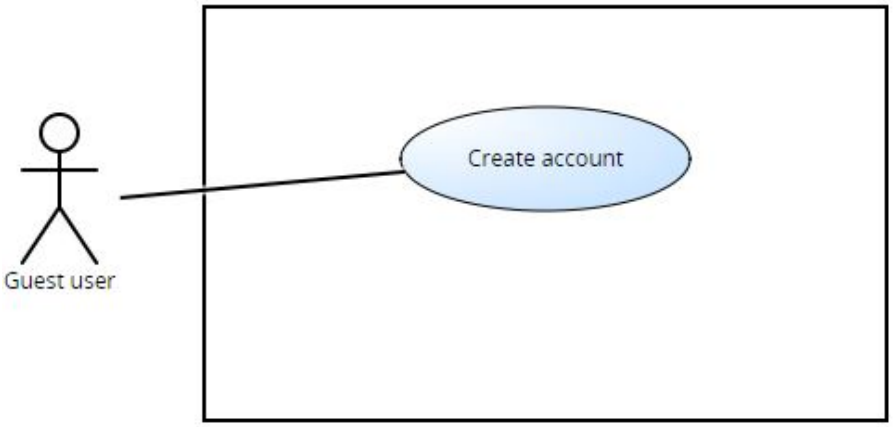

Figure 18: Guest user use case diagram

4.1.2 Registered User

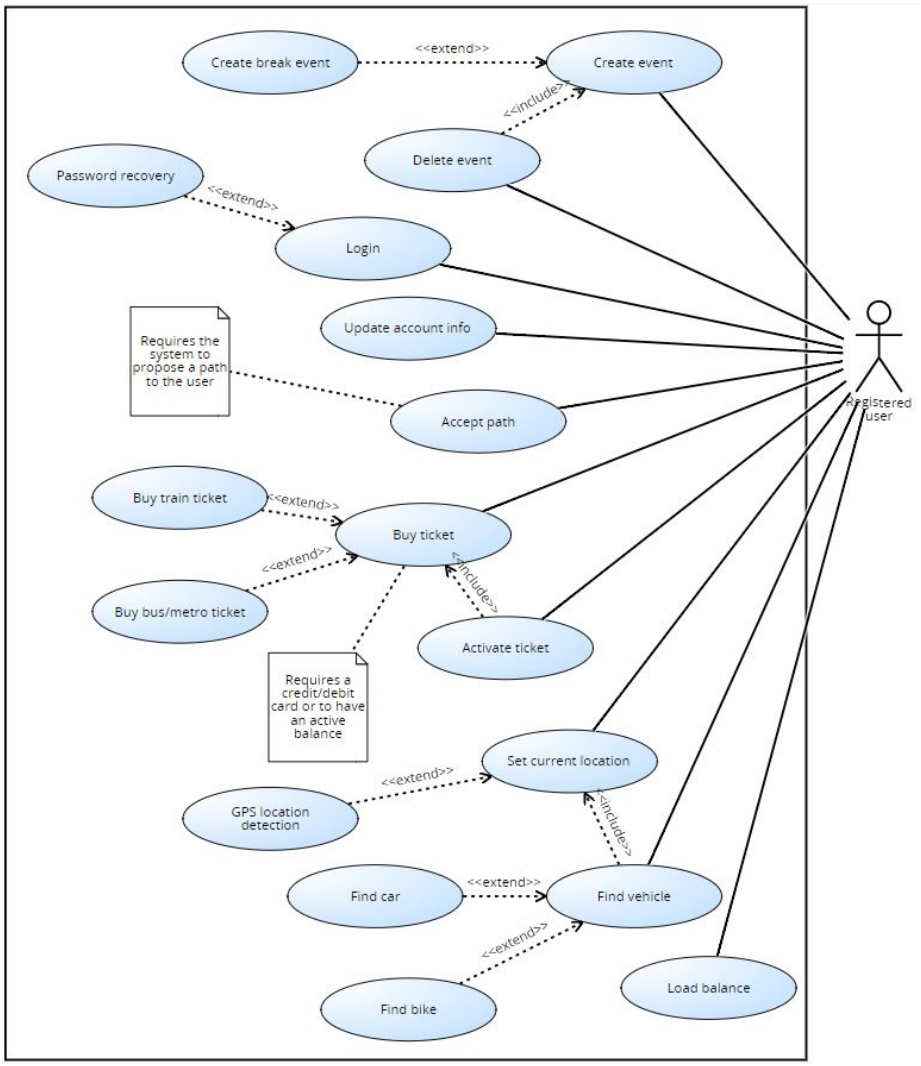

Figure 19: Registered user use case diagram

4.2 Use Cases

In the following section we include the use cases with their relative Sequence Diagrams and Statecharts.

4.2.1 Create Account

Name	Create Account
Actors	Guest User
Assumptions	The user has started a registration form. The user has a valid e-mail address.
Flow Of Events	The user types his e-mail in the form. The user reads the app's terms and conditions. The user accepts the app's terms and conditions. The user submits the information. The system sends an e-mail to the user. The user confirms the account through the link in the e-mail.
Exit Condition	The account is created
Exceptions	The e-mail is already linked to an account. The account is not created.

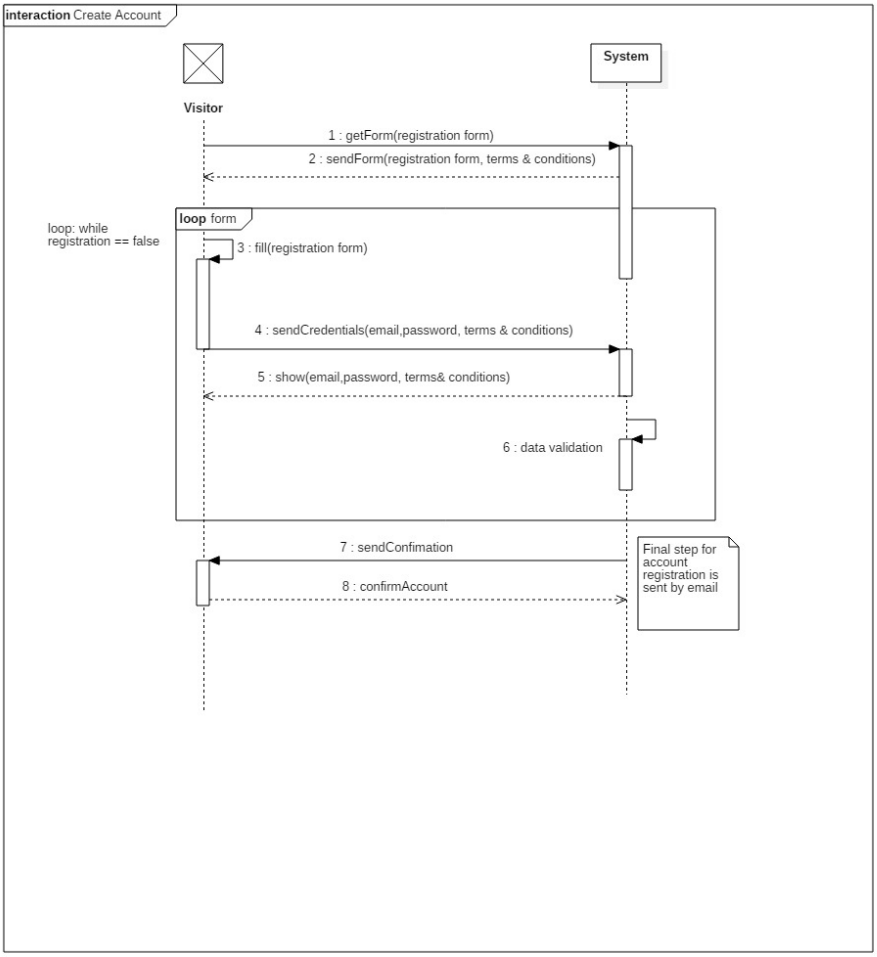

Figure 20: Create Account Sequence Diagram

4.2.2 Login

Name	Login
Actors	Registered User
Assumptions	The User has alreadty registered an account. The user hasn't logged in yet.
Flow Of Events	The user has to open the app homepage. The user has to input his information (e-mail and password). The system checks if the information sent by the user are valid.
Exit Condition	The user is logged in the system
Exceptions	The information provided by the user are wrong.The user is not logged in. The page will refresh with an error message.

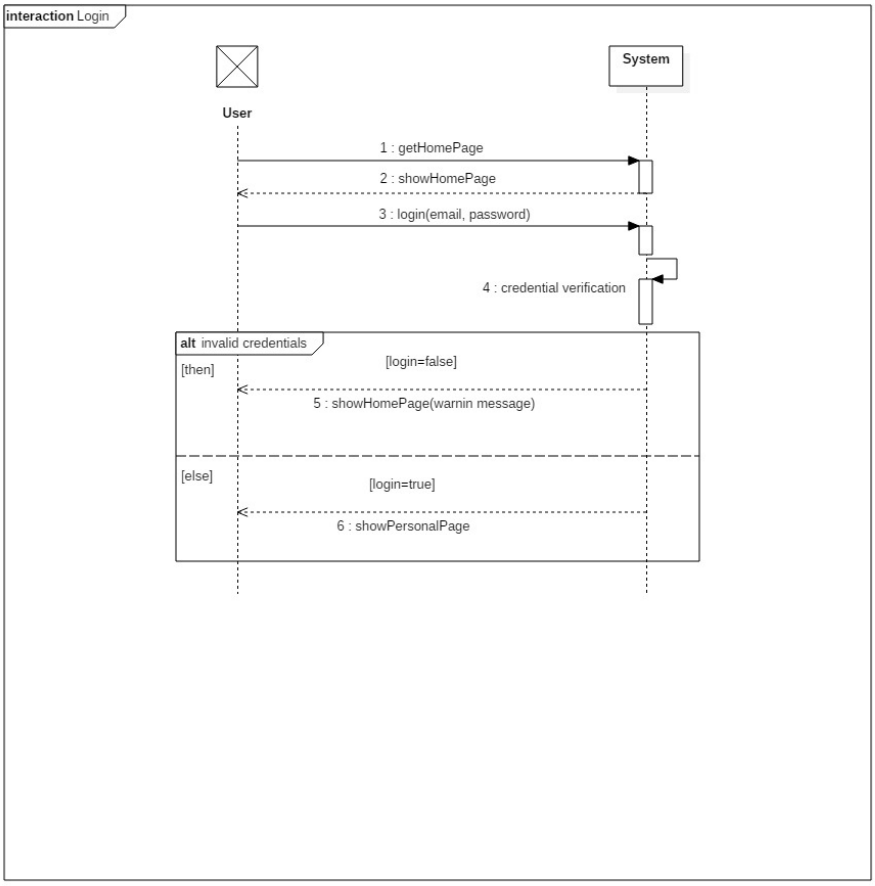

Figure 21: Login Sequence Diagram

4.2.3 Create Event

Name	Create Event
Actors	Registered User
Assumptions	The user has already logged in the account.
Flow Of Events	The user has to go to the homepage of the app The user has to open the "Create an event" page. The user has to fill the information needed for the event (time, date, duration and location). The user submits the information. The system sends a confirmation request under the form of a review of the event. The user confirms the information. The system loads the information on the user's schedule.
Exit Condition	The event is successfully created and added on the schedule.
Exceptions	The user does not confirm the information. The system will go back to the information submission page and ask the user to input the correct information. The connection is lost during the submission of the information. The event will not be created and the system will send an error message to the user who will be notified once the connection is restored. The location does not exist on the map. The system will send an error message and the user will have to update the location. The event is conflicting in time with another event, in this case the event will be created but the system will send the user a notification.

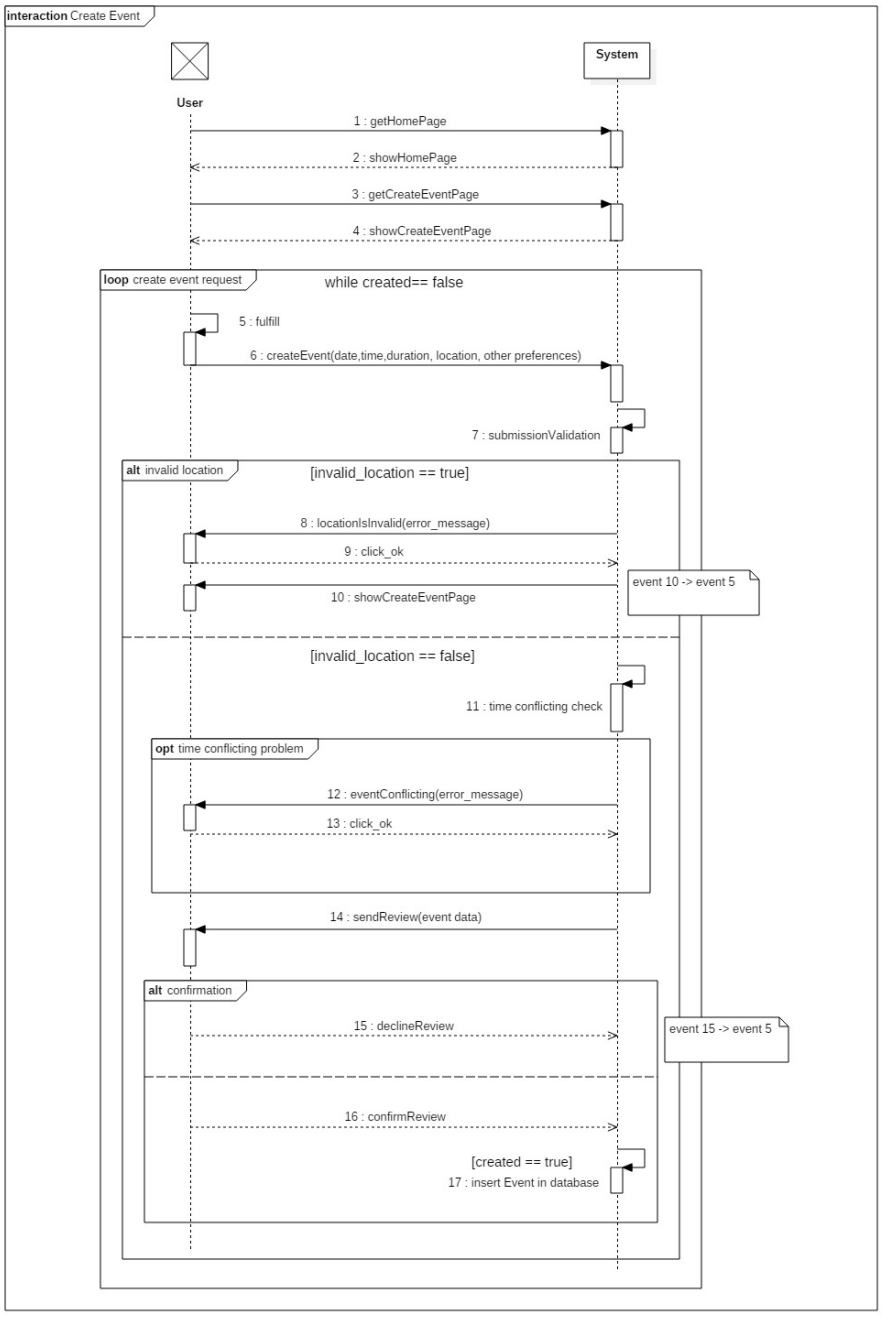

Figure 22: Create Event Sequence Diagram

4.2.4 Accept Path

Name	Accept Path
Actors	Registered User Google Maps
Assumptions	The user is logged in the system. The user has successfully created an event.
Flow Of Events	The system receives the new event The system checks the preferences of the user. The system will send the information of the event to Google Maps. Google Maps will calculate a path. Google Maps will send the path to the system. The system will send the path to the user. The user accepts the path. The system verifies its compatibility in the users' schedule.
Exit Condition	The path is created.
Exceptions	The user doesnt accept the path. The system will be notified and will have to start again by sending the information to Google Maps. The system, after learning from Google Maps, realises there are incompatible events and send a warning to the user. The user will have to select which event is more important. The system will send the new information to Google Maps and start again.

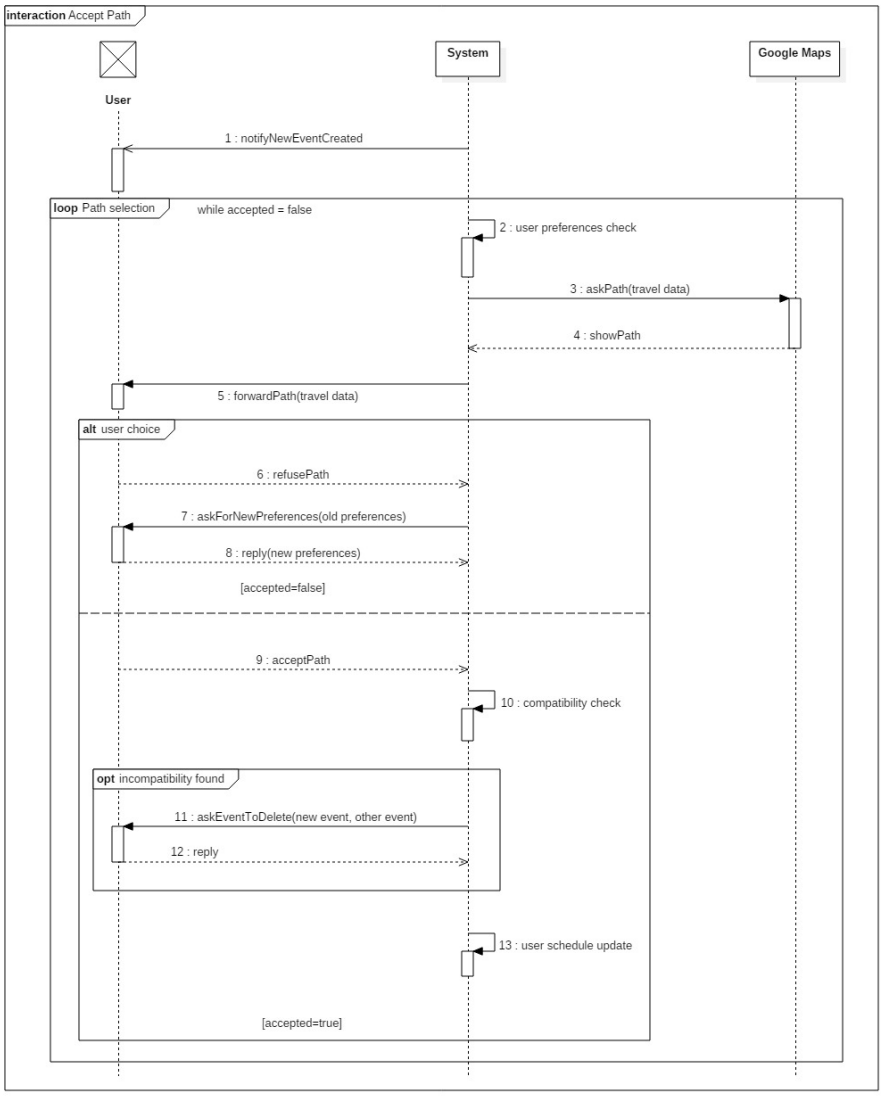

Figure 23: Accept Path Sequence Diagram

4.2.5 Buy Ticket

Name	Buy Ticket
Actors	Registered User Public Transportation System
Assumptions	The user is logged in the system. The user has opened a balance in the app or has a credit/debit card connected to it.
Flow Of Events	The user clicks on buy a ticket from the homepage of the app. The user selects the ticket he wants to buy. The system receives the request and forwards it to the correct public transportation system. The public transportation system sends the information to Travlendar+s system to complete the transaction. The system sends the information to the user. The user accepts the payment request. The user selects the payment option. The system sends the payment confirmation to the PTS. The PTS confirms the ticket. The PTS sends the ticket information to the system. The system saves the ticket in the users information.
Exit Condition	The ticket is available to be used by the user.
Exceptions	The balance does not have enough money to buy the ticket. The system will ask the user to select a different payment method. The credit card does not allow transaction. The system will ask the user to select a different payment method. The connection is lost between the system and the PTS. The system will have to ask to the user to start again from the selection of the ticket.

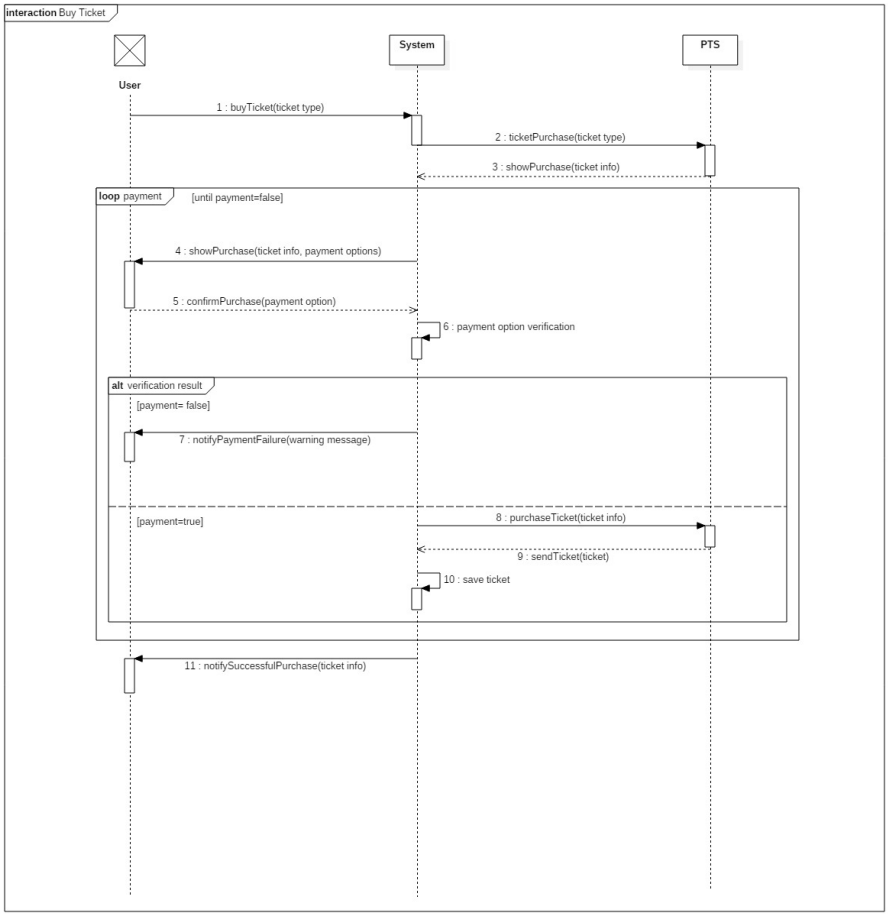

Figure 24: Buy Ticket Sequence Diagram

4.2.6 Activate Ticket

Name	Activate Ticket
Actors	Registered user PTS
Assumptions	The user has logged in the system. The user has bought at least one ticket. The ticket has not been activated yet.
Flow Of Events	The user has to click on My tickets in his personal page. The system will send to the user the information about his ticket. The user will select the ticket that he wants to activate. The user will press on Activate ticket. The system will send a confirmation message to the user with a summary about the ticket. The user confirms the ticket. The system sends the details about the activation to the PTS. The PTS will send a confirmation of the activation of the ticket. The system sends the confirmation to the user. The system will tag the ticket as activated.
Exit Condition	The ticket is activated and it becomes valid for the duration of the ticket.
Exceptions	The user doesnt confirm the ticket. The system will bring the user back to the selection of the ticket. There is a connection problem between the system and the PTS. The user will be notified of this and he will have to start the process again from the selection of the ticket.

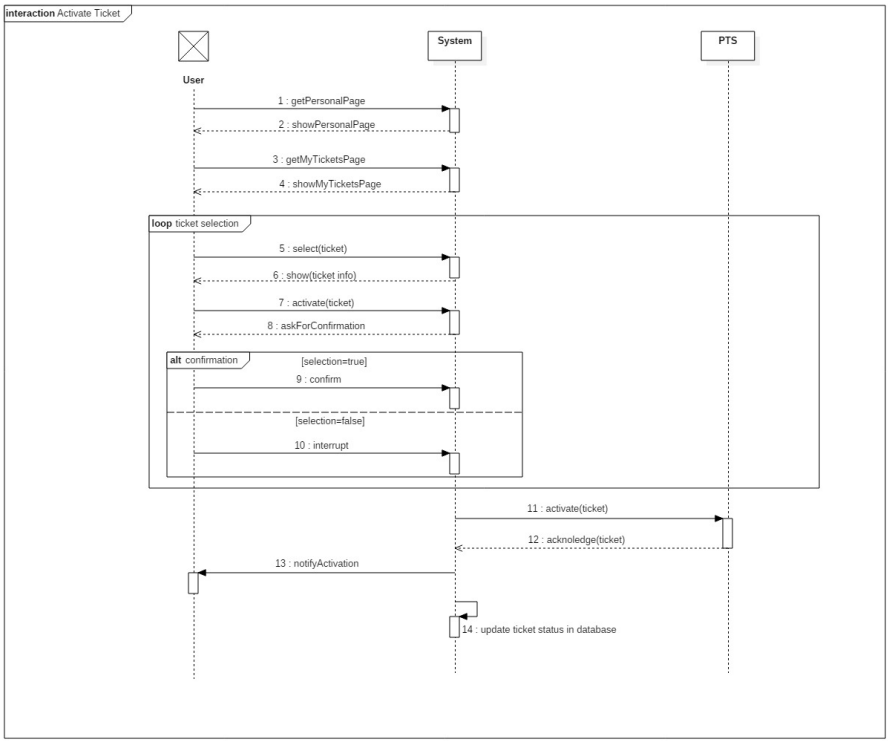

Figure 25: Activate Ticket Sequence Diagram

4.2.7 Find Vehicle And Access Service

Name	Find Car/Bike And Access Service
Actors	Registered user Car-Sharing system (CSS) or Bike-Sharing system (BSS)
Assumptions	The user is logged in the system. The user has a driving license. The information provided by the CSS are true. The users GPS is working and has the correct position.
Flow Of Events	The user has to click on Find a vehicle in the homepage of the app. The user has to select Car or Bike. The system sends a request for information to the CSS/ BSS. The CSS/ BSS sends the information needed to the system. The system will redirect the user to a map of his surroundings, with system. Once a vehicle is chosen, the system will redirect to that vehicles service app.
Exit Condition	The user will be able to see available cars from different car sharing systems near his location and pick one.
Exceptions	There is a connection problem between the CSS(BSS) and Travlendar+s system. The system will notify the user. The user will have to start again.

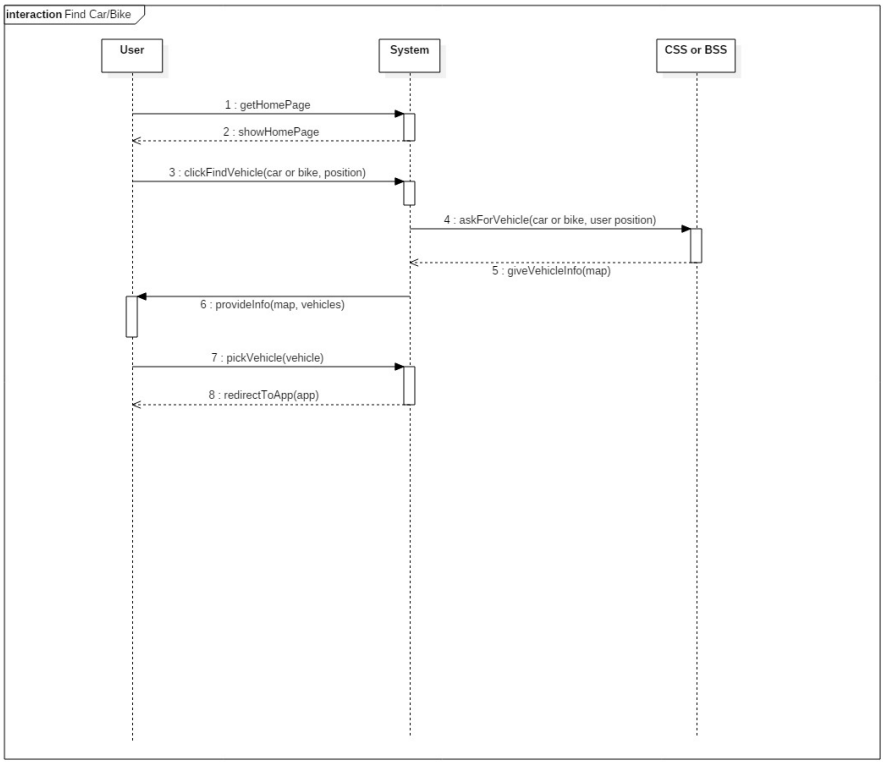

Figure 26: Find Vehicle And Acceess Service Sequence Diagram

4.2.8 Load Balance

Name	Load Balance
Actors	Registered User Bank
Assumptions	The user has a credit/debit card connected to the account or The user has bought a code to load the balance from a certified shop.
Flow Of Events	The user has to click on Load balance. The system will redirect him to the loading balance page. The user will have to select Credit/debit Card or Use a code. 1. Credit/Debit Card case: The user has to select the amount of money he wishes to load into the balance. The system sends the request to the bank. The bank confirms the transaction. 2. Use a code: The system asks the user to insert the code written on the card he bought. The user inserts the information requested and confirms. The system checks that the information is correct. The system sends a confirmation message to the user.
Exit Condition	The balance is updated.
Exceptions	The bank does not allow the transaction. The system requests the user to try again. The information sent by the user are incorrect. The system asks the user to check if the code is correct and to try again.

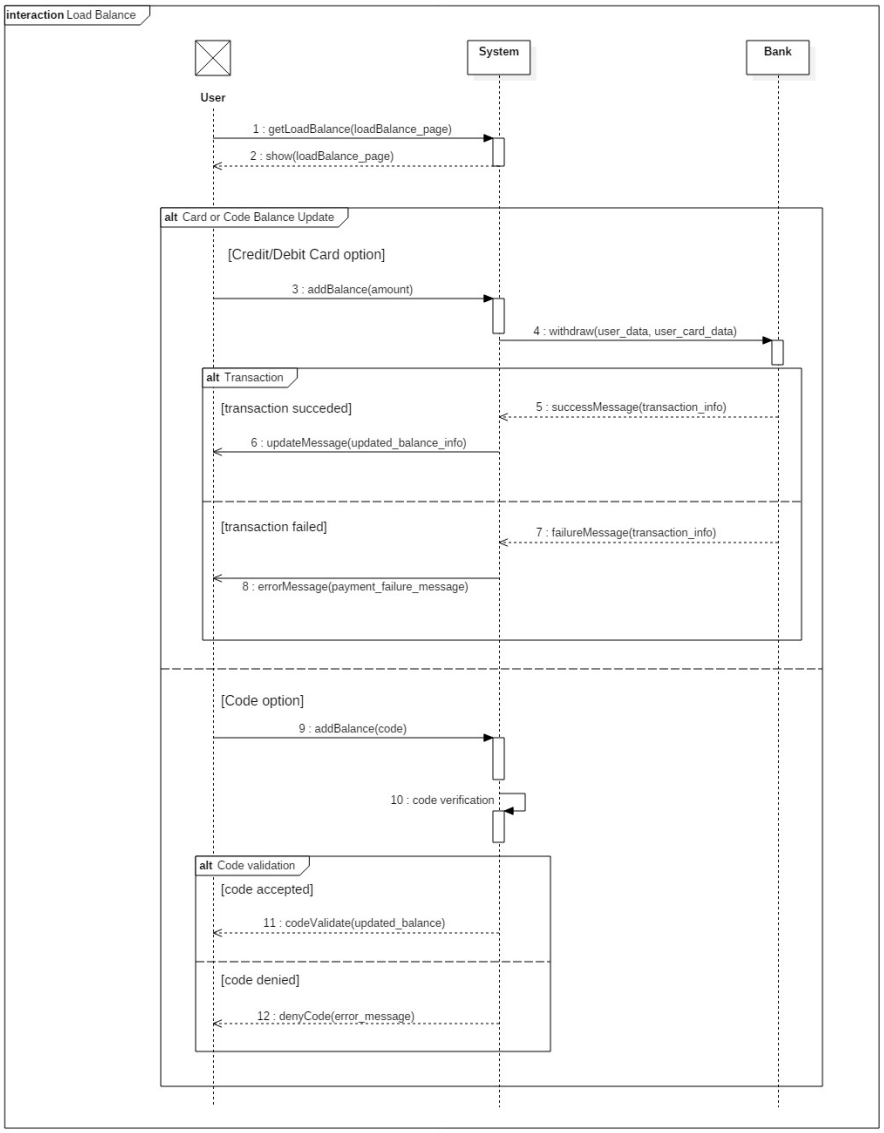

Figure 27: Load Balance Sequence Diagram