FMWay Işık

Requirements Specification and Analysis

V1.0

08.03.2020

Ersin ÇEBİ

Umutcan Neşet ÖLÇER

Çağatay DEMİRCAN

Emre Can RUA

Prepared for

SE301 Software Engineering



Table of Contents

[1. Introduction 1](#_Toc496873294)

[1.1. Purpose of the System 1](#_Toc496873295)

[1.2. Scope of the System 1](#_Toc496873296)

[1.3. Objectives and Success Criteria of the Project 1](#_Toc496873297)

[1.4. Definitions, Acronyms, and Abbreviations 1](#_Toc496873298)

[1.5. Overview 1](#_Toc496873299)

[2. Current System 1](#_Toc496873300)

[3. Proposed System 1](#_Toc496873301)

[3.1. Overview 1](#_Toc496873302)

[3.2. Functional Requirements 2](#_Toc496873303)

[3.3. Nonfunctional Requirements 2](#_Toc496873304)

[Usability 2](#_Toc496873305)

[Reliability 2](#_Toc496873306)

[Performance 2](#_Toc496873307)

[Supportability 2](#_Toc496873308)

[Implementation 2](#_Toc496873309)

[Interface 2](#_Toc496873310)

[Packaging 2](#_Toc496873311)

[Legal 2](#_Toc496873312)

[3.4. System Models 2](#_Toc496873313)

[Scenarios 2](#_Toc496873314)

[Use case model 2](#_Toc496873315)

[Object model 2](#_Toc496873316)

[Dynamic model 2](#_Toc496873317)

[User interface—navigational paths and screen mock-ups 3](#_Toc496873318)

[3.5. Project Schedule 3](#_Toc496873319)

[4. Glossary 3](#_Toc496873320)

[5. References 3](#_Toc496873321)

REQUIREMENTS ANALYSIS DOCUMENT[1]

# Introduction

## Purpose of the System

## Scope of the System

## Objectives and Success Criteria of the Project

* To provide reliable, efficient, lossless data.
* Well association between platform and database design.
* The general design of system in order to have fast, efficient application.
* The system has implementations that are understandable, clear and efficient.
* The system should be used by the people who are related with the application.
* The system should guarantee every user’s protection of information of their data.

## Definitions, Acronyms, and Abbreviations

- View is a visual representation of a model.

- GUI is graphical user interface.

- DB is short version of database term.

- Instructor is an actor an FMWay Işık and approved by admin

- Passenger, driver, visitor and registered user are an actor in a system.

## Overview

* Rest of the RAD contains non-functional (includes usability, reliability, performance, supportability, implementation, interface, operational, packaging, and legal requirements) and functional requirements (includes high-level functionality of the system).
* System models are given. Scenarios are inside of system model section. Scenarios are telling us about details of functional requirements. Use case models, object model, dynamic model and user interface view (mockup) are the parts of system model section.

# Current System

The application is providing travel opportunity exchange of money to registered users. In addition, the system should guarantee every user’s protection of information of their data.

# Proposed System

We are going to try developing an hitchhiking application, exchange of money between users. This application is a user friendly application and useful for everyone.

## Overview

The system will be including three actors which are admin, passenger, driver. Every driver is also can be passenger.

## Functional Requirements

## Nonfunctional Requirements

**Usability**

Application should be easily available for all users. The admin panel must also be available on phone. The home page must be the same for all users. Each page must have the same body structure. Actors’ pages should be understandable.

**Reliability**

All members must have secure access to the application. Visitors should not do anything. Logging to the application should be provided with unique e-mails and passwords that are appropriate for password criteria.

**Performance**

Application should be shown in mobile phones and tablets. Application should be running in more than one device simultaneously and access should be guaranteed. The application is going to be a dynamic content, so there should not be complicated queries in back end to not decrease performance.

**Supportability**

The system should be managed by admin. Developer will be responsible to provide continuance, compatibility and testability for the created program.

**Implementation**

**Interface**

The system should not interact with any existing system. The system should be able to be used by a user. The user should be connected to the network to use the features of the system.

**Packaging**

Admin should install the system. Also, the system is an application so the application should be uploaded on the server. There should be no time constraints on the installation. System’s all steps as a package are given within GitHub.

**Legal**

Project’s all contents are protected by the law of copyright.

## System Models

|  |
| --- |
| **Scenario name: Sign Up** |
| **Participant Actor Instances:** **Ersin: User** |
| **Flow of events:**   1. Ersin is at login page. He wants to login to the system but he has not any account for FMWay Işık. 2. He clicks register button. He enters information of his name, e-mail and password for this application. He clicks register button and; 3. If he enters wrong/used e-mail address, he sees an error message and he returns to the register page. 4. If he enters a used user name or selects bad user name, user sees error message and returns to the register page. 5. If he enters short password or does not enter any passwords, he sees an error message and returns to the register page. 6. He registers. |

|  |
| --- |
| **Scenario name: Login** |
| **Participating Actor : Ersin : User** |
| **Flow of Events:**   1. He is at login page screen. He enters e-mail and password and he clicks on the login button. 2. User can see 3 options; 3. If user leaves any blank area that is email/user name or password, user sees an error message that is saying “Username or password is wrong”. 4. If user enters wrong email/user name or password, user sees a message saying “Username or password is wrong”. 5. If user doesn’t make any mistakes, user logins to system. 6. He clicks log out button and exits his account and he returns to login page. |

|  |
| --- |
| **Scenario name: Edit Trip** |
| **Participant actor instances:** **Ersin: User** |
| **Flow of events:**   1. He logins at login page screen and He goes to home page. 2. He clicks the profile button. 3. He sees the current trips button and opens the trips page. He clicks the edit button on list, so he makes the desired changes. 4. He clicks on “Save” button and sees changes are saved, then he returns to profile page. 5. He clicks home page button, so he can click on log out button. She logs out of system. |

|  |
| --- |
| **Scenario name: Delete Trip** |
| **Participant actor instances:** **Ersin: User** |
| **Flow of events:**   1. He logins at login page screen and He goes to home page. 2. He clicks the profile button. 3. He sees the current trips button and opens the trips page. 4. He clicks on “Delete” button and sees dialog box clicks ‘confirm’, then he returns to profile page. 5. He clicks home page button, so he can click on log out button. She logs out of system. |

|  |
| --- |
| **Scenario Name: Search Trip** |
| **Participant Actor instances: Deniz(User)** |
| **Flow of events:**  1) She logins to the application and sees the homepage with some trips listed.  2) On the list, she doesn’t see the right trip, so she clicked on the search button.  3) She moves to the search page, enters the needed information such as starting point or destination.  4)She clicks “Show” button,  a) If the wanted trips exist, she sees the matching trips.  b) If the wanted trips don’t exist, the application says “ No match found!”  5)She chooses one and continues to approve that trip. |

|  |
| --- |
| **Scenario Name: Adding Driver Role** |
| **Participant Actor instances: İlkay(User)** |
| **Flow of events:**  1)He logins to the application and sees the homepage.  2)Because of every user sign up as passengers in the beginning, his profile is still a passenger. He wants to be a driver also.  3)He clicks to his profile, moves to the profile page. Then he clicks edit profile.  4)On edit page, he moves to “Driver” section, when he moves that, he sees the needed information for to be a driver.  5)He fills the information and upload driver-car licenses.  a)If he doesn’t fill the information, the application sends a feedback “This area cannot be empty!”  6)He clicks to “Send” button, then he waits for the approval.  7)After approval, he becomes a driver. |

|  |
| --- |
| **Scenario Name: Cancelling Trip** |
| **Participant Actor instances: Simay(User)** |
| **Flow of events:**  1)Simay enters the application and sees the homepage.  2)She selected and confirmed a trip earlier, so the trip appears on the homepage as “Upcoming trips”.  3)She clicks on the trip and moves to the “Trip Details” page.  4)She cannot joint to the trip, hence she wants to cancel her participation. Therefore she clicks “Cancel the trip” button.  5)There is a feedback says “Are you sure?”, she clicks “Yes” button.  6)Application cancels her participation to the trip and notifies the driver. |

|  |
| --- |
| **Scenario Name: User Feedback** |
| **Participant Actor instances: Hera(Passenger), Venüs(Driver)** |
| **Flow of events:**  1)Hera and Venüs completes the trip.  2)Hera enters the application and sees the specific trip that she attended to, on “Upcoming Trips” section.  3)She enters “Trip Details” page and then clicks “Trip completed” button.  4) The driver feedback page appears, she grades Venüs out of 5.  1)Venüs enters the application and sees the specific trip.  2) She enters “Trip Details” page and then clicks “Trip completed” button.  3)The passenger feedback page appears, she grades Hera with “Tick” or “Cross” shapes. |

|  |
| --- |
| **Scenario name: Edit Profile** |
| **Participant actor instances:** **Emre: User** |
| **Flow of events:**   1. He logins at login page screen and he goes to home page. 2. He clicks the profile button. 3. He sees the information of his own. He clicks the edit button, so he can change his user e-mail, password, first name, last name and gender. 4. He clicks on “Save” button and sees changes are saved (“It is edited successfully”), then he returns to profile page. 5. He clicks home page button, so he can click on log out button. He logs out of system. |

|  |
| --- |
| **Scenario name: Log Out** |
| **Participant actor instances:** **Emre: User** |
| **Flow of events:**   1. He is at login page screen and he enters email/username and password and clicks “Log In” button. 2. He clicks log out button, and he exits system in addition he returns to the login page. |

|  |
| --- |
| **Scenario name: Accepting Trip** |
| **Participant actor instances:** **Emre: Passenger** |
| **Flow of events:**   1. Emre is on the login page screen, he has entered the program by filling in the e-mail / username and password section. 2. Driver / trip search is completed from the trip search section. He matches the driver. He sees the information of the drive it matches. 3. If Emre wants to accept the trip, he presses the ‘Accept’ button. 4. Emre is directed to the homepage. // The passenger with the trip is directed to the trip tracking page containing the map on which he appears. |

|  |
| --- |
| **Scenario name:** Altering trip (EDITLEME OLAYI DEDİK GÜZERGAH VEYA SAAT) |
| **Participant actor instances:** **Emre: Passenger** |
| **Flow of events:**   1. The passenger has accepted the trip. 2. Emre can change the position of the trip if the vehicle is cruising. In addition, while waiting for the vehicle, it can change the location or time of the destination. 3. If Emre wants to accept the question ‘Are you sure’ about this, he approves the change by pressing the ‘Accept’ button. 4. Emre is directed to the homepage. // The passenger with the trip is directed to the trip tracking page containing the map on which he appears. |

|  |
| --- |
| **Use case name:** Sing up |
| **Participant actors: Potential User** |
| **Flow of events:**   1. Visitor opens application. 2. Visitor wants to registered in FMWay, so Registered User clicks Register button. 3. Visitor enters e-mail, first name, last name, password. 4. Visitor clicks Register button and returns login page. |
| **Entry Condition:** Visitor is not login. |
| **Exit Condition:** Visitor wants to exit in the system. |
| **Quality Requirements:**  If there is login, visiter dont acces Registered Page.  If This e-mail used before, no permission to register.  If user name used before, no permision to register. |

|  |
| --- |
| **Use case name:** Login |
| **Participant actors: User** |
| **Flow of events:**   1. User opens application. 2. System responds by displaying the login screen of the application. 3. User enters his/hers login info e-mail and password to the login fields. 4. User clicks "Login" button. 5. If the username and password match with the e-mail and password which is stored in database, users logins. 6. System fetches the dashboard of user. |
| **Entry Condition:** User visits the application. |
| **Exit Condition:** USer clicks "Login" button and entered information should be correct. |
| **Quality Requirements:**  If users leaves one or more fields empty, system displays a warning message, like "This area cannot be empty.”  If the information’s checked from database are not true, system displays a warning message, like "Wrong username or password, please retry.” |

|  |
| --- |
| **Use case name:** Edit Trip |
| **Participant actors:**User |
| **Flow of events:**   1. User clicks edit trip button. 2. User makes desired changes on a desired trip. |
| **Entry Condition: U**ser must be login**.** |
| **Quality Requirements:**  User cant leave break any area. |

|  |
| --- |
| **Use case name:** Delete Trip |
| **Participant actors:**User |
| **Flow of events:**   1. User clicks delete trip button on desired trip . 2. System respondes with a dialog box. 3. According to dialog box system deletes/keeps the trip. |
| **Entry Condition: U**ser must be login**.** |
| **Exit Condition:** User declines the dialog box. |
| **Quality Requirements:**  User cant leave break any area. |

|  |  |
| --- | --- |
| **Use Case Name** | **Search Trip** |
| **Participating Actor** | Registered User |
| **Entry Condition** | User must be logged in. |
| **Flow of Events** | 1)User clicks “Search Trip” Button.  2)User moves to “Search Trip” Page.  3)User fills the needed information about wanted trip.  4)User clicks “Show” button.  5)The application shows the filtered result. |
| **Exit Condition** | User sees the filtered result with related to his/her needs. |
| **Quality Requirements** | * If the user leaves one or more fields empty on needed areas, system displays a warning message such as "This area cannot be empty.” * If there isn’t any matched trip, the application displays a warning message such as “No match found!” |

|  |  |
| --- | --- |
| **Use Case Name** | **Adding Driver Role** |
| **Participating Actor** | Registered User |
| **Entry Condition** | User must be logged in. |
| **Flow of Events** | 1)User clicks “Profile” button.  2)User sees the details of his/her profile.  3)User clicks “ Edit Profile” button.  4)User moves to “Driver” section.  5)User fills the needed information ,uploads his/her licenses.  6)User clicks “Send” button.  7)User waits for the approval.  8)After approval, user gets logged out and gets log in again.  9)User has the option of “Add Trip” |
| **Exit Condition** | User becomes a driver. |
| **Quality Requirements** | * If the user leaves one or more fields empty on needed areas, system displays a warning message such as "This area cannot be empty.” * If the needed information is not accurate or true, the user cannot be a driver on the system. |

|  |  |
| --- | --- |
| **Use Case Name** | **Cancelling Trip** |
| **Participating Actor** | Registered User(as Passenger and as Driver) |
| **Entry Condition** | User must have chosen a trip earlier. |
| **Flow of Events** | 1)User clicks on the specific trip below the section “Upcoming Trips”.  2)User sees the details of the trip.  3)User clicks “Cancel the Trip” button.  4)User clicks “Yes” on the warning.  5)The driver gets notified. |
| **Exit Condition** | User cancels the trip. |

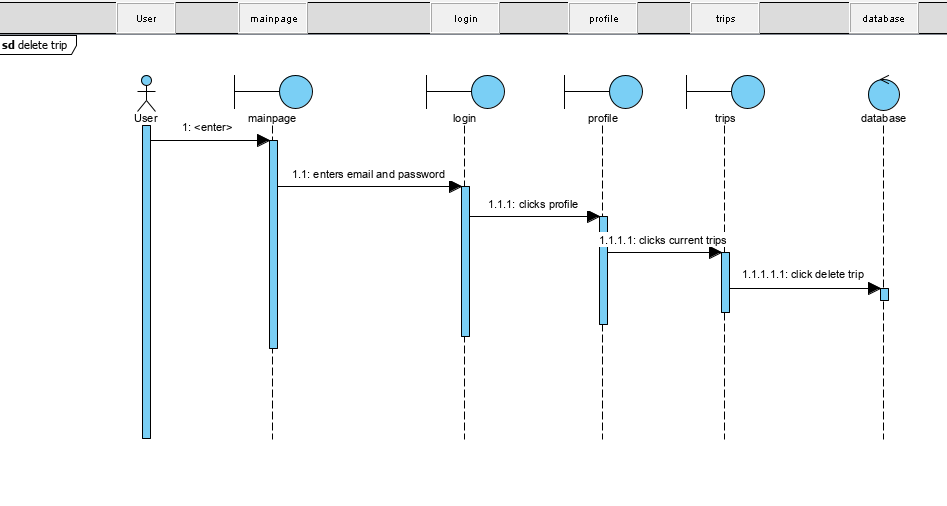
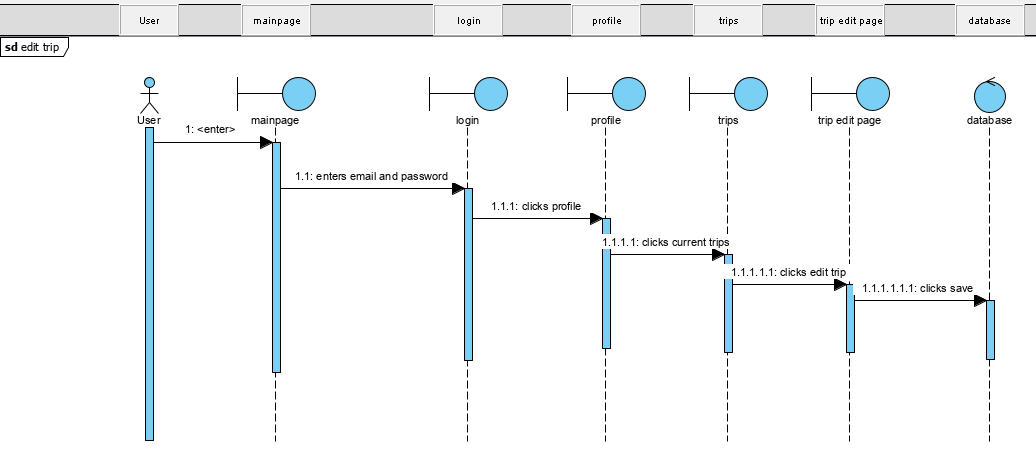
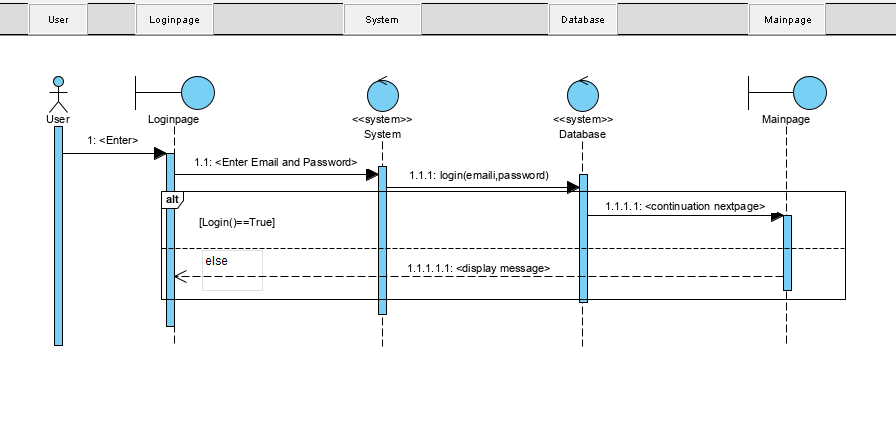
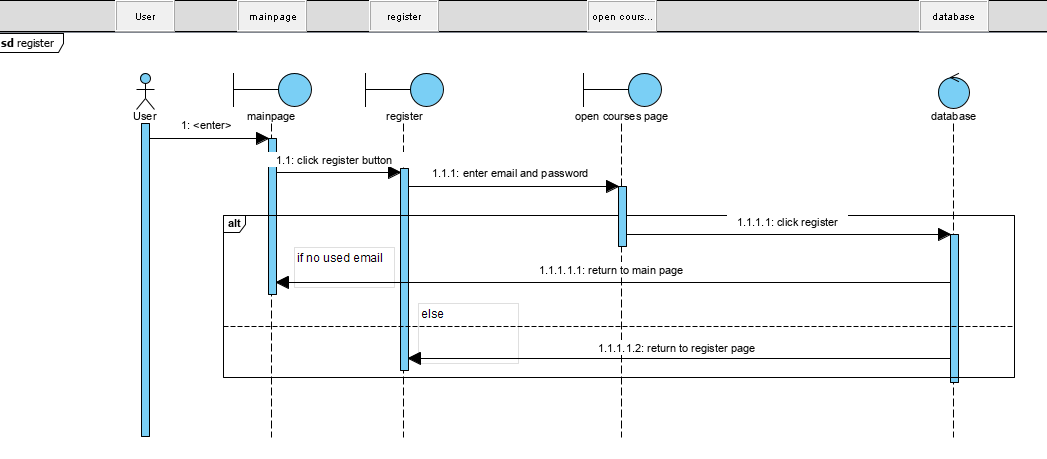
|  |  |
| --- | --- |
| **Use Case Name** | **User Feedback** |
| **Participating Actor** | Registered User(as Passenger and as Driver) |
| **Entry Condition** | The trip must be completed. |
| **Flow of Events** | 1)After clicking “Trip Completed” button on trip details, the feedback of driver appears for the passenger and the feedback of passenger for the driver .  2)Passenger rates the driver out of 5.  3)Driver rates the passenger with the signs “Tick” or “Cross”. |
| **Exit Condition** | The rates of both users reaches to the application database and the rates appear on both profiles. |
| **Quality Requirements** | * If the users don’t want to rate, they can skip the feedback option. |

|  |
| --- |
| **Use case name:** Edit Profile |
| **Participant actors:**Registered User |
| **Flow of events:**   1. User clicks Edit Profile button. 2. User wants to change first name, last name and e-mail. If user don’t enter anything any area, Edit profile button is not active. 3. User can change information and clicks Edit Profile button, so user information saved and redirect Profile Page. |
| **Entry Condition: U**ser must be login**.** |
| **Exit Condition:** User wants to exit system. |
| **Quality Requirements:**  User cannot leave break any area. |

|  |
| --- |
| **Use case name:** Log Out |
| **Participant actors:**Registered User |
| **Flow of events:**   1. Registered User wants to exit in the system. 2. Registered User clicks log out button. 3. Registered User returns Login Page. |
| **Entry Condition:** Registered User must be login in the system. |
| **Exit Condition:** Registered User wants to exit system. |
| **Quality Requirements:**  If there is not login, Registered User will does not do log out. |

|  |
| --- |
| **Use case name: Accepting Trip** |
| **Participant actors: Passenger** |
| **Flow of events:**   1. After the passenger has entered the required route and time information, he presses the trip search button. 2. A list of trip suitable for the passenger's screen is displayed. The passenger chooses the one that suits him. 3. Yolcu (öncekinin 4ü ile bağlantılı) |
| **Entry Condition:**Passengers must have completed the required travel hours and logged in. |
| **Exit Condition:** Passenger wants to find a suitable trip. |
| **Quality Requirements:**  After the passenger starts the journey, if he / she wants to search for the choice of the destination, the approval is taken. |

|  |
| --- |
| **Use case name:** Altering trip |
| **Participant actors: Passenger** |
| **Flow of events:**   1. The passenger has accepted the trip. 2. While the passenger waits for the car, he wants to change the location of his destination. 3. He changes his destination 4. He confirms the changes by pressing the passenger ‘Accept’ section. |
| **Entry Condition:**Passenger must have confirmed the trip |
| **Exit Condition:** The passenger wants to change the time or route of the trip. |
| **Quality Requirements:**  Passenger after making changes to the trip, he confirms his transactions by pressing ‘Accept’ under the question ‘Are you sure?’ |



SEARCH TRIP

ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

ADDING DRIVER ROLE

ekran görüntüsü, harita içeren bir resim

Açıklama otomatik olarak oluşturuldu

CANCELLING TRIP

ekran görüntüsü içeren bir resim

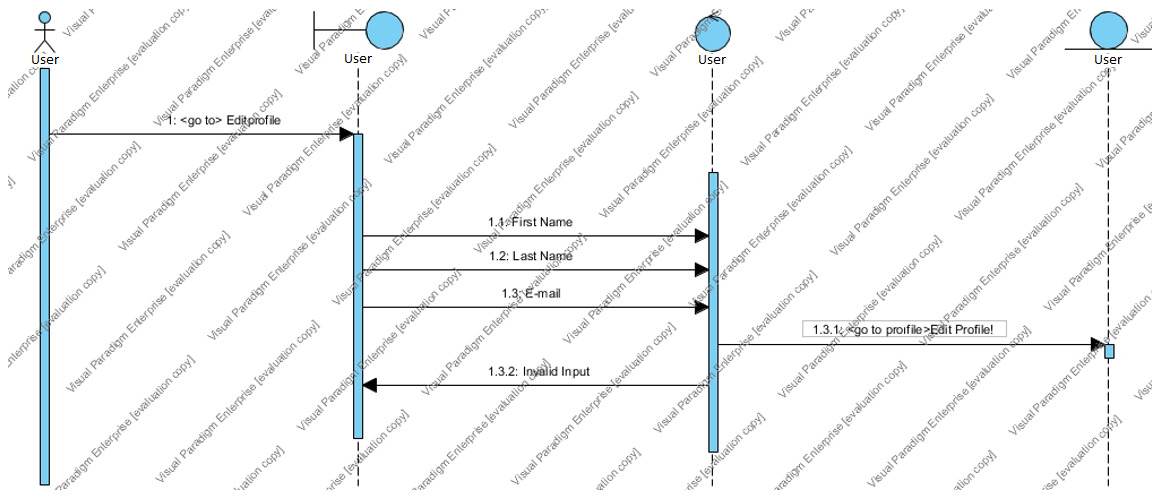
Açıklama otomatik olarak oluşturuldu

USER FEEDBACK

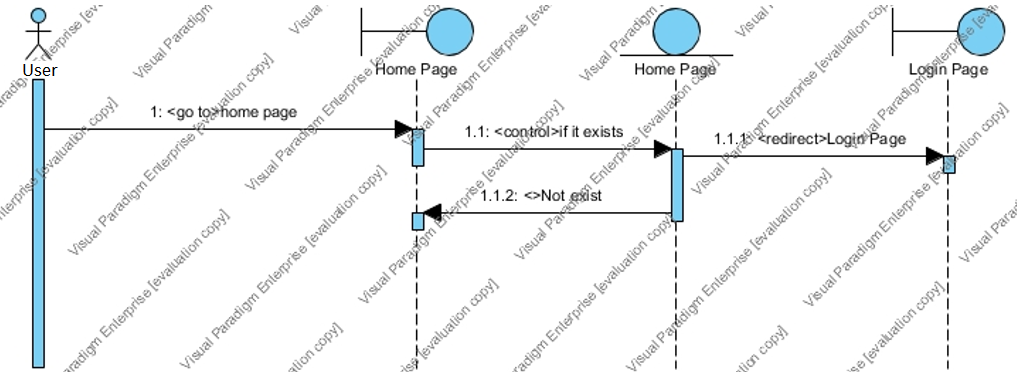
ekran görüntüsü, harita içeren bir resim

Açıklama otomatik olarak oluşturuldu

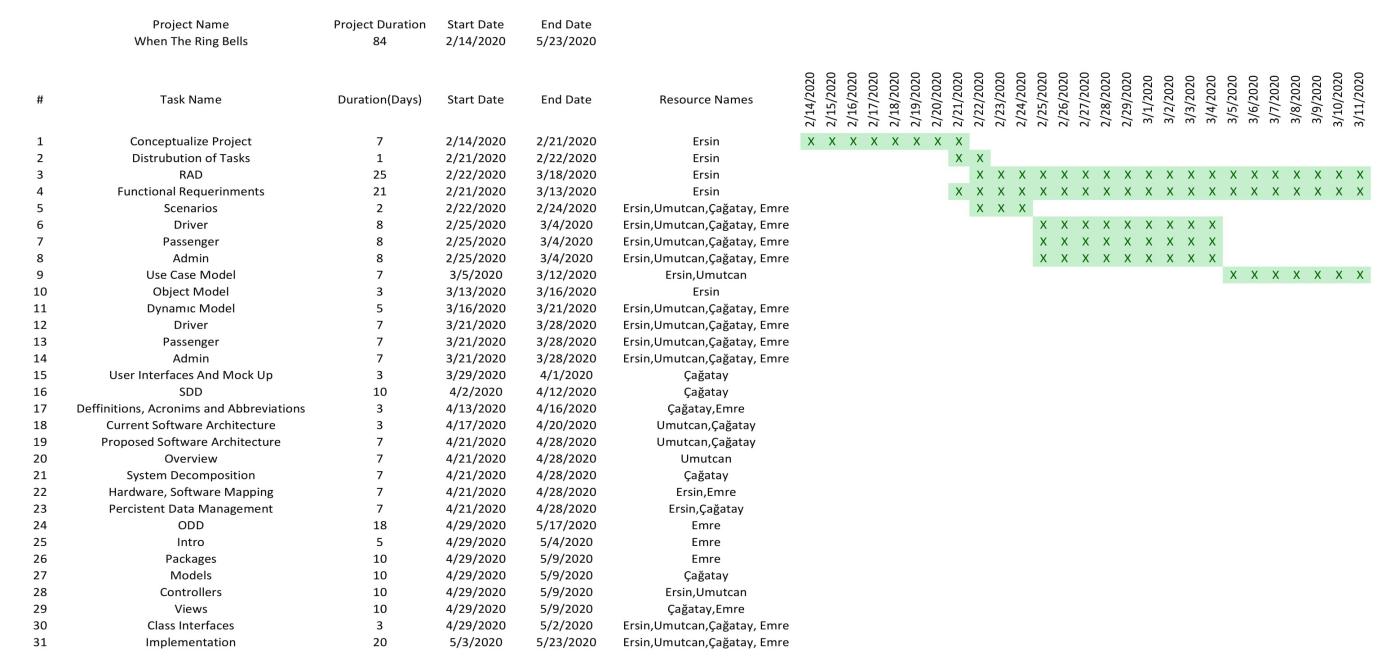
User Edit Profile

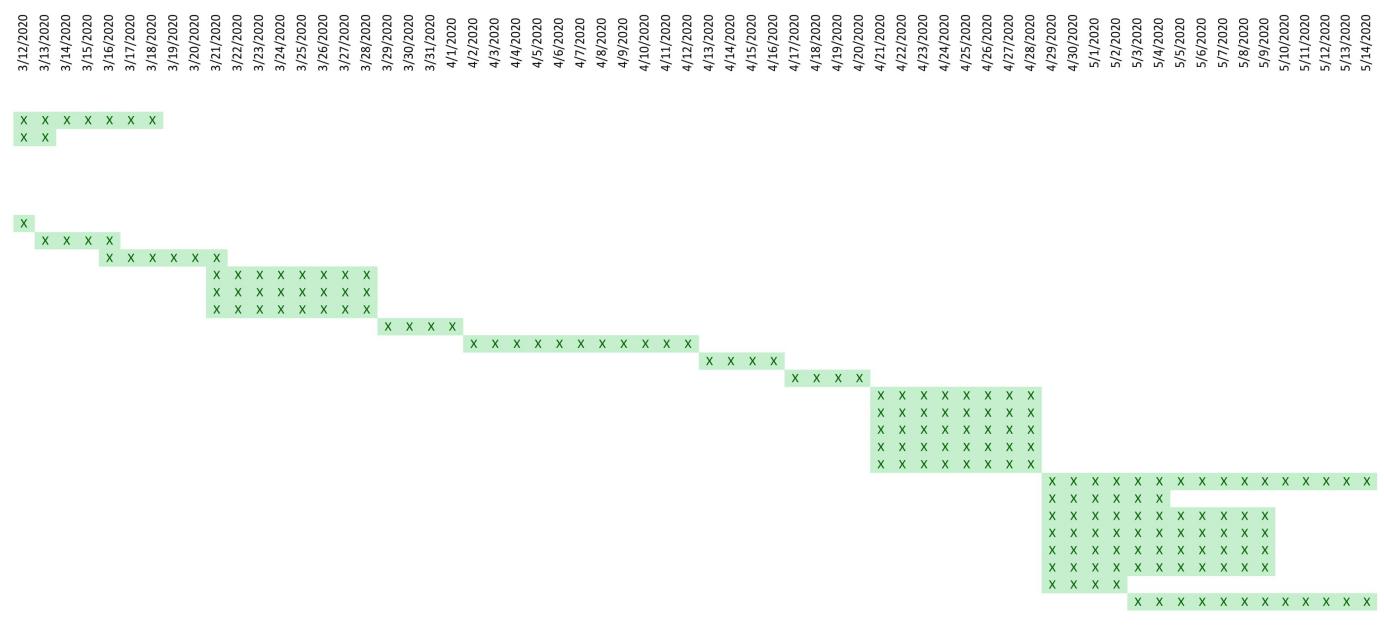


User Log Out



## Project Schedule







# Glossary

# References

1. Bruegge B. & Dutoit A.H.. (2010). *Object-Oriented Software Engineering Using UML, Patterns, and Java*, Prentice Hall, 3rd ed.