<THE BUS TICKET SYSTEM>

Requirements Specification and Analysis

<1.0>

<Date>

<DOĞAN SARIBAŞ

FEYZULLAH BERKAY DANIŞ

DENİZ ÇALIŞKAN

NURSENA KARAKULAH>

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

The main purpose of the bus ticket system is to buy or book the bus ticket/tickets as a registered user or unregistered user.

## Scope of the System

Users will be able to buy single or multiple tickets as an e-ticket and use the bus ticket reservation/sell system as a registered user or unregistered user. For the registered users, the system will offer the campaigns. For example, the registered user will get the ticket at a discounted price but the unregistered user will not profit by campaigns, they will buy at the fixed price. Users must enter their names, surnames, identification numbers and cell phone numbers to register to the system. Also, they must accept the 'Membership Agreement and Privacy Policy'. Registered users can edit their profile, they can renew their personal and contact information. Users will be able to reserve the ticket first, after that, if they wish they will buy the ticket or cancel the reservation. Admin will be able to add bus/busses or remove bus/busses, edit the campaigns and remove the ticket/tickets. If the user encounters a problem while using the system, they will be able to contact the admin.

## Objectives and Success Criteria of the Project

Through the system, users should be able to easily view the appropriate bus schedule

between the day and time they want to travel and select the appropriate one. The system should ensure a service that users can be able to reserve or buy single/multiple tickets and if they wish, they can cancel. To make easier to use, the system should have simple design, user should access easily services of the system. If the users encounter any problem, they should contact the admin and get a feedback as soon as possible.

## Definitions, Acronyms, and Abbreviations

Admin: Admin is an actor that has access to all functions and basicacally manages the system.

Registered User: Registered users are users that sign up to the system. They can buy and reserve multiple tickets at once, give feedbacks, view trips, check their current tickets and ticket history, and contact with admin.

Unregistered User: Unregistered users are users who do not sign up to system. They have access to system and buy ticket however they cannot buy multiple ticket or see their ticket history. Also they cannot give feedbacks.

## Overview

* Rest of the RAD contains information about the current system and how it handles tasks that the new system performs.
* An overview of the new system, functional requirements(high-level fuctionality of the system) which mentions all the actions that the users of the system will perform, non-functional requirements which are user related requirements such as usability, reliability, supportability, implementation, interface, packaging and legal requirements.
* System models contains scenarios and use cases which describes the functional requirements with details, use case model(UML diagram of use cases), object model (UML class diagram of the system), dynamic model which contains sequence diagrams, screen mockups of the system.
* Project schedule which contains Gannt Chart.
* Last two parts of the RAD are Glossary that contains defeinition of the objects in use cases and References that is the specification of the sources that we used.

# Current System

One of the current systems that provide users with the online ticket reservation/purchase service is “KamilKoç.com”. People can search for trips and buy tickets from them as registered or unregistered users, registered users can also reserve tickets and win parapuan from the tickets they buy. Registered users can see their tickets after logging in and unregistered users can see their ticket by using a ticket serial number, Users can see contact information in contact page. Our system is expected to achieve these tasks more efficiently.

# Proposed System

Documents the requirements elicitation and the analysis model of the new system

## Overview

Our website - Viatorem – is a website that can be used to buy bus tickets. As an unregistered user you can see bus trips, buy tickets for busses, see popular trip rotations and check or remove your already bought tickets. Registered users can simply login the system and they gain access to see their ticket history. They also can buy multiple tickets at once also they benefit from our campaigns when they buy tickets. Registered users also can give feedbacks about system. They also can contact admins through the viatorem. Admins arrange bus trips with adding or removing trips. They also decide campaigns and they are able to remove certain tickets if it is necessary.

## Functional Requirements

Admins specifies campaigns for special days or long time users.

Admins add adviced trip rotations of populer places for those who looks for trip rotations.

Admins can login the system with their special username and password

Admins add new busses to the system by specifying their starting and final destination, how many hours will the trip takes.

Admins can also remove added busses. In case of a removement, purchased tickets from that bus will be removed as well.

Admins can remove someone’s ticket in necessary situations.

Users who did not register before can easily register to Viatorem by simply clicking sign up button and entering asked informations.

Users who signed up before can login with their username and passwords

Unregistered users can check their tickets from view tickets section. On the other hand registered users can check their tickets from view tickets section and they can see their ticket history.

Registered users can contact with admins through viatorem and send messages to admins’ mailbox.

Registered users can give feedbacks about system.

Registered users can easily reserve a ticket for further purchase instead of directly purchasing it.

Registered users can easily change their passwrod by simply clicking forgot my passwword button.

Registered users can change their personal information such as their phone number in Edit Profile section. Also they can see their “moneypoints”.

Both registered and unregistered users can purchase tickets. However registered users can purchase multiple tickets at once. Also registered users are able to benefit from campaigns that are decided by admins.

Both registered and unregistered users can go to viatorem and see available bus trips.

Both registered users and unregistered who have bought tickets can remove their tickets from viatorem.

Both registered users and unregistered users have access to help section which explain how to use viatorem system.

## Nonfunctional Requirements

Usability : • Anyone who has already shopped on the internet should be able to register to this site and perform ticketing functions without any help.

Reliability : • Payment information should not be displayed anywhere except the purchase page and the user password should not be displayed anywhere.

Performance : • The user should be able to buy as many tickets as he wants.

Supportability : • Admin should be able to add new buses of different types and features to different cities.

•Admin should be able to configure campaigns at any time without problems.

Implementation : : • The system will be implemented with Django and the tools specified in the previous file.

Interface : • Other than the programs used to make the overall site will not be used and The interface will not be made to disrupt the overall structure

Packaging : • The system was built by Nursena K, Dogan S, F. Berkay D, and Deniz O by establishing approximately 20-30 programs.

Legal : • This software ins regulated under the DDNB(first letters of our names) general public license.

## System Models

Describes the scenarios, use cases, object model, and dynamic models for the system. This section contains the complete functional specification, including mock-ups illustrating the user interface of the system and navigational paths representing the sequence of screens.

### Scenarios

Berkay’s Scenarios

|  |  |
| --- | --- |
| *Scenario name* | **ViewTrip** |
| *Participating actors* | Mike as User(Admin/Registered User/Unregistered User) |
| *Flow of Events* | 1. Mike wants to see the trips from İstanbul to Ankara and opens the VİATOREM website. 2. Mike then chooses the city İstanbul as starting location and city Ankara as ending location of the trip also chooses the day of the trip and confirms these informations. 3. Mike sees the trips starting from İstanbul at the choosen date and ends at Ankara listed in front of him. |

|  |  |
| --- | --- |
| *Scenario name* | **MakeReservation** |
| *Participating actors* | Sarah as Registered User |
| *Flow of Events* | 1. Sarah thinks she might go from İstanbul to Ankara but she is not sure, she opens the VİATOREM website. 2. Sarah signs in to the website. 3. Sarah views the trips from İstanbul to Ankara on a specified date. 4. Sarah selects a trip and opens the trip page for reservation. 5. Sarah chooses an empty seat on the bus for reservation and confirms it. 6. Sarah submits a form that asks for her credit card number, name and surname and email. |

|  |  |
| --- | --- |
| *Scenario name* | **CancelTrip** |
| *Participating actors* | Jack as Admin |
| *Flow of Events* | 1. Jack wants to remove the trip that starts from İstanbul on November 10th and ends at Ankara from the website and he signs in to the website. 2. Jack opens the Cancel Trip page. 3. Jack selects the trip he wants to cancel and confirms it for cancelling. |

|  |  |
| --- | --- |
| *Scenario name* | **AddTrip** |
| *Participating actors* | Jack as Admin |
| *Flow of Events* | 1. Jack wants to add a new bus trip that starts at Bursa on July 13th and ends at Trabzon to the website so he opens VİATOREM website on his computer. 2. He signs in to the website. 3. Jack opens the AddBus page. 4. Jack fills a form by entering the trip ID, starting location, ending location, starting date and starting hour of the of the trip , price and adds the new trip. |

Deniz’s Scenarios

Nursena’s Scenarios

|  |  |
| --- | --- |
| ***Scenario name*** | loginToTheBusSystem |
| ***Participating actors*** | Mike: RegisteredUser |
| ***Flow of Events*** | 1. Mike decides to travel to his family that live at another city. He registered to cite viatorem that is bus ticket reservation/sell system. 2. Mike enters ‘Log in to Viatorem’ button and appears the screen of login. He writes his Email address and password that he determined during registration process. 3. Viatorem does not confirm his email address and he writes his id and password and viatorem confirms his informations and Mike logs to the Viatorem. |

|  |  |
| --- | --- |
| ***Scenario name*** | editYourProfile |
| ***Participating actors*** | Mike: RegisteredUser |
| ***Flow of Events*** | 1. Mike logs in to Viatorem by entering login button and with his email address and password. 2. Mike enters button of my profile and opens the screen of ‘my profile’. There are Mike’s details of money points earned from previous travels ,view tickets, profile information. 3. Mike enters to the profile information button to change his contact information because he changed his cellphone number. 4. Mike renews his cellphone number to get the information of the next trips to the his new cellphone number. 5. To record his new information, after the writing new one ,he pushes to the button of ‘save’ and he logs out from Viatorem. |

|  |  |
| --- | --- |
| ***Scenario name*** | forgotPassword |
| ***Participating actors*** | Mike: RegisteredUser |
| ***Flow of Events*** | 1. Mike decides to enter to Viatorem. 2. Mike opens the log in page and writes his email address and password but he sees a pop-up on that page about the wrong email address or password. 3. Mike tries again but he encounters the same situtation and he clicks on the ‘forgot password’ button. 4. Mike gets an email to the his email address that registered on the Viatorem about changing the password. 5. Mike opens the mail and clicks the change password button and a page on Viatorem are opened. 6. Mike writes a new password and under the new password part , he writes again same password to confirm the new one, lastly he clicks save button. 7. He sees a pop-up on the page about renewal of the password. 8. Mike changes successfully his password. |

|  |  |
| --- | --- |
| ***Scenario name*** | registerViatorem |
| ***Participating actors*** | George: RegisteredUser |
| ***Flow of Events*** | 1. George wants to travel around the his country so he decides to register a bus ticket system. He finds the Viatorem. 2. George opens the Viatorem and clicks the ‘register’ button. 3. George fills respectively name, surname, gender, birth date, email address, cellphone number and password parts. 4. After the writing the information, George clicks the ‘finish the registration button. 5. To finish the registration, George gets an email to his email address that registered to Viatorem and he opens mail. There is a ‘finish the registration’ button, he clicks the button and a page are opened on Viatorem about finishing the registration process. 6. George registeres to Viatorem successfully. |

|  |  |
| --- | --- |
| ***Scenario name*** | signInAsAdmin |
| ***Participating actors*** | Alice:Admin |
| ***Flow of Events*** | 1. Alice is admin of Viatorem cite and she opens Viatorem login page. 2. Alice enters button of login and writes her email address and password. 3. Alice logs in to Viatorem. |

Doğan’s Scenarios

|  |  |
| --- | --- |
| *Scenario name* | **GiveFeedback** |
| *Participating actors* | John : RegisteredUser |
| *Flow of Events* | 1. John would like to give his views on his travels, return to comment on the services and make suggestions. 2. To do so, John first enters the system with his account. 3. John then clicks the give feedbak button on the home page. 4. He writes his views in the empty section he sees, and after pressing the send button, he reads the message your message has been delivered. 5. So he knows it's being transmitted. |

|  |  |
| --- | --- |
| *Scenario name* | **UserRemovesTicket** |
| *Participating actors* | Joshua : User |
| *Flow of Events* | 1. Joshua is giving up his journey and wants to cancel his ticket. 2. He enters the system and clicks the view trips button. 3. By scrolling down he finds his ticket and press the remove ticket button in the ticket information section |

|  |  |
| --- | --- |
| *Scenario name* | **AdminRemovesTicket** |
| *Participating actors* | David : Admin |
| *Flow of Events* | 1. David enters the system as admin. 2. It is receiving information that there is a fault in the system and that a ticket must be removed. 3. He presses the edit ticket button that appears on his home page and searches for tickets from the list. 4. He finds the ticket and presses the remove ticket button on it. |

|  |  |
| --- | --- |
| *Scenario name* | **UserWantsHelp** |
| *Participating actors* | Thomas : User |
| *Flow of Events* | 1. Thomas is looking for help to learn how to cancel his ticket. 2. To do this, he presses the help button on the main page. 3. In the frequently asked questions section, he looks for a question that can help him and reads the answer when he clicks on it. So he understands how to do it. |

|  |  |
| --- | --- |
| *Scenario name* | **WantToSeeCampaign** |
| *Participating actors* | Daniel : User |
| *Flow of Events* | 1. Daniel wants to save money for travelling. First he opens the home page. 2. Then He's looking for a suitable campaign for himself. 3. He realizes the campaign that says 20% discount for tickets to Antalya within a week. 4. He clicks on the campaign and redirect to Antalya tickets page. |

|  |  |
| --- | --- |
| *Scenario name* | **AdminAddCampign** |
| *Participating actors* | Charles : Admin |
| *Flow of Events* | 1. Charles enters the system as admin. 2. He then clicks the add campaign button that appears on his home page. 3. After entering the information such as discount percentage, number of tickets and date of the campaign, he clicks the add campaign button. |

### Use case model

Deniz’s Use Case Descprition

|  |  |
| --- | --- |
| *Use case name* | **BuyTicket** |
| *Participating actors* | Initiated by UnregisteredUser |
| *Flow of events* | 1. UnregisteredUser clicks view trip to see available bus trips.  2. Viatorem show all available trips.  3. UnregisteredUser selects a trip.  4. Viatorem displays empty seats.  5. UnregisteredUser selects an empty seat from the bus.  6. Viatorem asks for credit card information, Name, Surname and e-mail address.  7. UnregisteredUser enters required information.  8. If the budget in the credit card or wrong credit card information is entered , The system shows a pop-up dialogue about it. If not Viatorem finishes the buy ticket process. |
| *Entry condition* | UnregisteredUser clicks the view trips button. |
| *Exit condition* | * UnregisteredUser completes purchase and see a confirmation message. * The Unregistered user has got an explanation about why the buy ticket process could not performed. |
| *Quality requirements* |  |

|  |  |
| --- | --- |
| *Use case name* | **BuyTicket** |
| *Participating actors* | Initiated by RegisteredUser |
| *Flow of events* | 1. RegisteredUser clicks view trip to see available bus trips.  2. Viatorem show all available trips.  3. RegisteredUser selects a trip.  4. Viatorem displays empty seats.  5. RegisteredUser selects two empty seats from the bus for her friend July and herself.  6. Viatorem asks for credit card information  7 RegisteredUser enters required information  8.If the budget in the credit card or wrong credit card information is entered , The system shows a pop-up dialogue about it. If not Viatorem finishes the buy ticket process |
| *Entry condition* | The registeredUser clicks the view trips button. |
| *Exit condition* | * RegisteredUser completes purchase and see a confirmation message. * The Unregistered user has got an explanation about why the buy ticket process could not performed. |
| *Quality requirements* |  |

|  |  |
| --- | --- |
| *Use case name* | **contactWithOfficer** |
| *Participating actors* | Initiated by RegisteredUser |
| *Flow of events* | 1 RegisteredUser opens viatorem  2. RegisteredUser goes to “contact us” section on viatorem.  3. Viatorem displays a page where user can send a mail to an officer’s mailbox.  4. RegisteredUser sends a mail to ask about campaign.  5. Viatorem informs the registered user with a pop-up about successfully sending the email. |
| *Entry condition* | Logging in the system. |
| *Exit condition* | * Getting mail sent confirmation. |
| *Exceptional*  *Cases* | Connection down |

|  |  |
| --- | --- |
| *Use case name* | **viewMyAllTickets** |
| *Participating actors* | Initiated by RegisteredUser |
| *Flow of events* | 1. RegisteredUser goes to view tickets section on viatorem.  2. Viatorem shows all tickets registeredUser had bought.  3. RegisteredUser checks his/her ticket history. |
| *Entry condition* | Logging in to system with mail and password |
| *Exit condition* | * Succesfully viewing ticket history. |
| *Exceptional*  *Cases* | Connection down |

|  |  |
| --- | --- |
| *Use case name* | viewTicketDetail |
| *Participating actors* | Initiated by UnregisteredUser |
| *Flow of events* | 1. UnregisteredUser visits viatorem to check his/her ticket.  2. UnregisteredUser opens the view ticket section.  3. Viatorem asks PNR number.  4. UnregisteredUser enters PNR number.  5.If the PNR is invalid, Viatorem shows a pop-up about invalid PNR number. If not Viatorem shows he detail of the ticket. |
| *Entry condition* | To have a purchased a ticket |
| *Exit condition* | Succesfully monitoring ticket information. |
| *Quality requirements* |  |

Berkay’s Use Case Descprition

|  |  |
| --- | --- |
| *Use case name* | viewTrip |
| *Participating actors* | Initiated by Registered/Unregistered User/Admin (User) |
| *Flow of events* | 1. User opens the VİATOREM website and opens the ViewTrip page by clicking on the “View Trip” button. 2. VIATOREM shows the View trip page that includes a search bar that asks for the informations of starting location, ending location and starting date of the trip. 3. User enters starting location, ending location and starting date of the trip and searches the trips by pressing the “View Trip” button, if one of the information boxes are left emtpy, system displays a message that says “You need to fill all the boxes”. 4. VİATOREM system shows a list of trip that fits the description, if there is not a trip that fits the description system displays a message that says “No trip found”. |
| *Entry condition* |  |
| *Exit condition* | User can see the list of trips. |
| *Quality requirements* |  |

|  |  |
| --- | --- |
| *Use case name* | AddTrip |
| *Participating actors* | Initiated by Admin |
| *Flow of events* | 1. Admin clicks on the “AddTrip” button. 2. VIATOREM opens the add trip page that displays a form that asks for trip ID, starting location, ending location and starting date and time of the trip and the price. 3. Admin enters trip ID, starting location, ending location, starting date and time of the trip and price and clicks on “Add Trip” button.if one of the information boxes is left empty system displays a message that says “You need to fill all the boxes”. 4. VIATOREM adds the new trip to system and displays a message saying “New trip is successfully added”. |
| *Entry condition* | Admin must be logged in. |
| *Exit condition* | The new trip is in the system. |
| *Quality requirements* |  |

|  |  |
| --- | --- |
| *Use case name* | CancelTrip |
| *Participating actors* | Initiated by Admin |
| *Flow of events* | 1. Admin opens the Cancel Trip page by clicking the “Cancel Trip” button on his homepage. 2. VIATOREM displays a page that asks for the trip ID. 3. Admin writes the PNR number of the trip that is wanted to cancel and selects the trip and clicks on “Cancel” button. If the trip does not exist in the system VIATOREM displays a message saying “trip could not be found”. 4. VİATOREM cancels that trip from system and displays a message that says “Trip is successfully canceled”. |
| *Entry condition* | Admin must be logged in. |
| *Exit condition* | The trip is cancelled. |
| *Quality requirements* |  |

|  |  |
| --- | --- |
| *Use case name* | MakeReservation |
| *Participating actors* | Initiated by Registered User |
| *Flow of events* | 1. Registered user views available bus trips. 2. VIATOREM shows a list of possible trips with the “Buy Ticket” and “Make Reservation” buttons next to them. 3. Registered User selects a trip and clicks on “Make Reservation” button. 4. VİATOREM system displays the trip page that shows trip information and bus seats. 5. Registered User selects a seat by clicking on it and clicks on “Done” button, if the seat selected is taken by someone else system displays a message that says “You need to choose an empty seat.” 6. VIATOREM displays a form that requires the name, surname and email of the Registered User. 7. Registered User enters his/her name, surname, email and submits the form. 8. System reserves the seat for that user. |
| *Entry condition* | Registered User must be logged in. |
| *Exit condition* | Registered User has got the seat reserved for him/her. |
| *Quality requirements* |  |

Nursena’s Use Case Descprition

|  |  |
| --- | --- |
| *Use case name* | **loginViatorem** |
| *Participating actors* | Iniated by Registered User |
| *Flow of events* | 1. The registered user opens the Viatorem login page . 2. The registered user clicks on the ‘Log in’ button. 3. Viatorem responds by opening the log in page. The log in page contains a form which includes email address and password. 4. The registered user writes their email address and password and clicks the ‘Log in’ button. 5. If the registered user does not fill the one of the   email address or password, Vitorem  shows a pop-up dialogue  about filling the all field or email  address/password is not saved  on the databse,Viatorem  shows a pop-up about “wrong  password or email address”.  If not Viatorem confirms logging process. |
| *Entry condition* | * The registered user opens the home page of Viatorem. |
| *Exit condition* | * The registered user succesfully logged into Viatorem. * The registered user has got an explanation about why the login process could not performed. |
| *Quality requirements* | * The registered user receives a response from the system in less than 4 second. |

|  |  |
| --- | --- |
| *Use case name* | Logout |
| *Participating actors* | Initiated by Registered User |
| *Flow of events* | 1. The Registered User clicks on the log out button that under the my profile field. 2. Viatorem responds by acting a visitor to the registered user. |
| *Entry condition* | The Registered User must be logged in. |
| *Exit condition* | * Registered User has looged out successfully from VIATOREM. |
| *Quality requirements* | The registered user should be logged out from the system in less than 4 seconds. |

|  |  |
| --- | --- |
| *Use case name* | **loginAdmin** |
| *Participating actors* | Iniated by Admin |
| *Flow of events* | 1. Admin clicks on the log in button on the home page of Viatorem. 2. Viatorem opens the log in page. 3. Admin writes their email address and password. 4. If the registered user does not fill the one of the   email address or password, Vitorem shows a pop-up dialogue  about filling the all field or email address/password is not saved  on the databse, Viatorem shows a pop-up about “wrong  password or email address”.  If not Viatorem confirms logging process. |
| *Entry condition* | * The registered user opens the home page of Viatorem. |
| *Exit condition* | * The registered user succesfully logged into Viatorem. * The registered user has got an explanation about why the login process could not performed. |
| *Quality requirements* | * The registered user receives a response from the system in less than 4 second. |

|  |  |
| --- | --- |
| *Use case name* | **registerToViatorem** |
| *Participating actors* | Iniated by User |
| *Flow of events* | 1. The user opens the home page of Viatorem. 2. The user clicks the register button. 3. Viatorem responds by opening the register page. The register page contains a form which includes the name, surname, gender, birth date, email address, cell phone number and password. 4. The user completes the form by writing its informations. 5. The user clicks the “finish the registration” button to complete the register process. 6. Viatorem checks whether the required fields such as   name, surname, email address, password and cell phone number.  If some are not filled, Viatorem shows a pop-up dialogue about  mandatory fields not completed or the email address is aldready  registered in the database , the System shows a pop-up dialogue  about registered email address problem. If not Viatorem sends an  email to finish the registration to the user.     1. User gets an email, email includes a button “finish the registration” ,user clicks the button . 2. Viatorem confirms the registration. |
| *Entry condition* | * The user opens the home page of Viatorem and clicks the registration button. |
| *Exit condition* | * The registered user successfully registered into Viatorem. * The registered user has got an explanation about why the register process could not performed. |
| *Quality requirements* | * The registered user receives a response from the system in less than 4 second. |

|  |  |
| --- | --- |
| *Use case name* | **editProfile** |
| *Participating actors* | Iniated by Registered User |
| *Flow of events* | 1. The registered user logs into Viatorem. 2. The registered user clicks the “edit profile” button that under the my profile field. 3. Viatorem responds by opening the edit   profile page that includes three main parts, personal  information, contact information, membership information.  The personal information field includes name, surname,  birth date and gender. The contact information field includes  email address and cell phone number. The membership field  includes the old password, new password and new password  repeat fields.   1. The registered user changes their mail address and cell phone number, also changes the password by writing the old one and new one and clicks the save button. 2. Viatorem responds by showing a pop-up   dialogue such as “we have successfully saved your  information” and directs the registered user to the edit  profile page. If old password is wrıng when changing the  password, Viatorem shows a pop-up dialogue about wrong  old password. |
| *Entry condition* | * The registered user is logged into Viatorem. |
| *Exit condition* | * The registered user changed information which they wants to change successfully. * The registered user got an explanation about the mismatch of the new password and its repetition. |
| *Quality requirements* |  |

Doğan’s Use Case Descprition

|  |  |
| --- | --- |
| *Use case name* | **GiveFeedBack** |
| *Participating actors* | Initiated by RegisteredUser |
| *Flow of events* | 1. RegisteredUser clicks the give feedback button on he home page.   2. Viatorem opens the give feedback page.  3. RegisteredUser writes his message to text box and clicks the send button.  4. Viatorem shows a page with the message Your action has been performed. |
| *Entry condition* | * RegisteredUser is logged into Viatorem. |
| *Exit condition* | * Viatorem displays the confirmation message. |
| *Exceptional*  *Cases* |  |

|  |  |
| --- | --- |
| *Use case name* | **UserCancelTicket** |
| *Participating actors* | Initiated by RegisteredUser and UnregisteredUser |
| *Flow of events* | 1.RegisteredUser clicks the view trips button.  2. Viatorem opens the trips page.  3. RegisteredUser clicks the cancel ticket button next to the ticket that he wants to cancel.  4. Viatorem shows a page with the message Your ticket has been canceled. |
| *Entry condition* | * RegisteredUser is logged into Viatorem. |
| *Exit condition* | * Viatorem displays the " Your ticket has been canceled”message. |
| *Exceptional*  *Cases* |  |

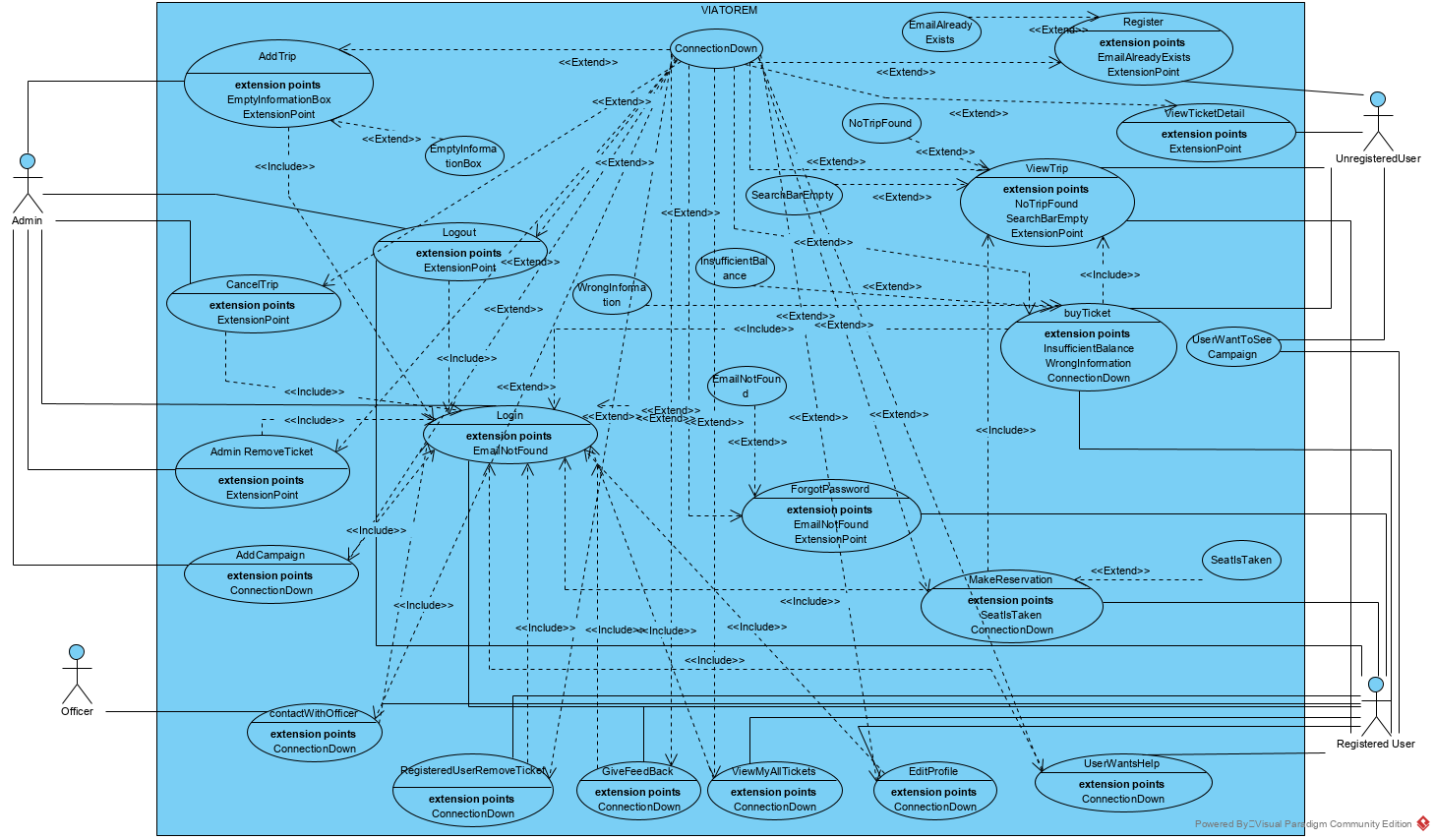
|  |  |
| --- | --- |
| *Use case name* | **AdminCancelTicket** |
| *Participating actors* | Initiated by Admin |
| *Flow of events* | 1.Admin clicks the edit ticket button on the home page.  2. Viatorem opens the search ticket page.  3. Admin writes the ticket id and clicks the cancel ticket button.  4. Viatorem shows a page with the message The ticket has been canceled. |
| *Entry condition* | * Admin is logged into Viatorem. |
| *Exit condition* | * Viatorem displays the " Ticket has been canceled” message. |
| *Exceptional*  *Cases* |  |

|  |  |
| --- | --- |
| *Use case name* | **UserWantHelp** |
| *Participating actors* | Initiated by RegisteredUser and UnregisteredUser |
| *Flow of events* | 1.User clicks the help button on the home page.  2. Viatorem opens the help page.  3. User clicks the topic that he search for.  4. Viatorem shows the topic page. |
| *Entry condition* | * User opens the homepage. |
| *Exit condition* | * Viatorem displays the topic page. |
| *Exceptional*  *Cases* |  |

|  |  |
| --- | --- |
| *Use case name* | **UserWantToSeeCampaign** |
| *Participating actors* | Initiated by RegisteredUser and UnregisteredUser |
| *Flow of events* | 1.User clicks the Campaigns button on the home page.  2. Viatorem opens the Campaigns page.  3. User clicks the Campaign that he is related to.  4. Viatorem shows the campaign information page. |
| *Entry condition* | * User opens the homepage. |
| *Exit condition* | * Viatorem displays the campaign information. |
| *Exceptional*  *Cases* |  |

|  |  |
| --- | --- |
| *Use case name* | **AdminAddCampaign** |
| *Participating actors* | Initiated by Admin |
| *Flow of events* | 1. Admin clicks the Add Campaign button on the home page. 2. Viatorem opens the Add Campaign page. 3. Admin enters the campaign information and clicks the add button.   4. Viatorem shows the campaign has been added message. |
| *Entry condition* | * Admin is logged into Viatorem. |
| *Exit condition* | * Viatorem displays the confirmation message. |
| *Exceptional*  *Cases* |  |

Use Case Diagram



### Object model

The analysis object model, depicted with UML class diagrams, includes classes, attributes, and operations. The analysis object model is a visual dictionary of the main concepts visible to the user.

### Dynamic model

The dynamic model is depicted with sequence diagrams and with state machines. Sequence diagrams represent the interactions among a set of objects during a single use case. State machines represent the behavior of a single object (or a group of very tightly coupled objects). The dynamic model serves to assign responsibilities to individual classes and, in the process, to identify new classes, associations, and attributes to be added to the analysis object model.

When working with either the analysis object model or the dynamic model, it is essential to remember that these models **represent user-level concepts, not actual software classes or components.**

### User interface—navigational paths and screen mock-ups

## Project Schedule

Prepare Gannt Chart, and add it to this section.

# Glossary

To establish a clear terminology, developers **identify the participating objects** for each use case. Developers should **identify, name, and describe them** unambiguously and collate them into a glossary.

# References

This subsection should:

* Provide a complete list of all documents referenced elsewhere in the RAD, or in a separate, specified document.
* Identify each document by title, report number - if applicable - date, and publishing organization.
* Specify the sources from which the references can be obtained.

The following is an example of listing a book in this section. Check the text to see how it is cross referenced (The whole document is based on [1]).

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