MEKELLE UNIVERSTIY

EITM-SCHOOL OF COMPUTING

DEPARTMENT OF SOFTWARE ENGNEERING

# Software engineering capstone project 1

Documentation submission

| Group members | ID |
|---|---|
| Yonas Bekele | 117633/09 |
| Tsion yegeremu | 117663/09 |
| Dawit Teka | 117681/09 |
| Kidus T/yohannes | 117776/09 |

Submitted date. December 25, 2019

Title

# Ethio Train Ticket management system

## Abstract

This proposal is a brief description of the system that we are going to design and implement. It includes highlight of the system functionalities and some features it consists. We believe by writing this proposal the intended users will have a brief knowledge about the system what we are trying to develop.

However this paper wont contain every detailed aspects of the system but the next papers will have a proper and detailed parts of the system.

This papers intention is to make sure our intended users like the idea and let us develop the system we are trying to do.

# Chapter 1

# Proposal

## 1.1 Introduction

As the title mentioned above this system is based on the newly emerging and expanding rail way system of our country. So, this rail way by itself has multiple benefits that gives to the society, but when we revolutionize this the users of the rail way will have access to the benefits anywhere any time.

As this century is turning to be a technological century, we believe every aspects of this world should be implemented in a technological aspect. And we are playing a part in that technological aspect by changing a manual and tiring work to a more advanced and easy system.

Some of the benefits of this system have been mentioned above like easiness and so but this system provides a lot more than that. For Instance: - It decreases energy for the customers, accessibility will increase, and there are a lot more.

So We believed you have been introduced to system a little bit so, some of the components and things we used and are going to use will be mentioned below.

## 1.2 Background of the system and its problem areas

Now a days the rail way’s ticket management working system is done in manually. The users have to go to ticketing offices to get information. If they found their information and they acquire some problem or challenge (EG: - doesn’t have enough cash) they have to go back home and come back again.

This working system is not available without working hours. So, some customers won’t be able to get their information and do want they intend to do.

The current implementation has very small amount of offices so there is the lack of availability to the entire customers.

If the implementation is continuing like this it will be very hard to manage and handle customers because this rail way system is developing and emerging quickly and widely increasing its destinations.

## 1.3 Motivation

Our motive is that there are a lot of problems in the current system so we as students and future working society believe there might be difficulties for us and the customers that use this system. So, our sake and also for the societies purpose we intend to create a great interaction between the customers and services provides.

There were some other motives. From those one of them is there is a lack of transportation in our country for people with low income. So, this rail way system helps a lot of people. So, we choose this title to work on because it solves some problems and we also solve the interactions between those customers of the rail way and the service provider.

## 1.4 General and specific objectives

Our general objective is to solve the problems that the current system produces. More or less the main general objective is to create an automated system.

What we want to accomplish mainly in this system as mentioned above is automation. When achieving this objective, we also will achieve some of the modules or specific objectives. some of them are:

- Online Ticketing
- Online information Management (information availability)
- Easy access.
- Data organization
- Data analysis

This are the specific objective that we want to achieve. While achieving this we also want to achieve the non-functional requirements of the system. And those are: -

- Accessibility
- Effectiveness
- Easy usage
- Decreasing queues and decreasing tiresomeness
- Report Generation
- Manageability
- Easy information access
- understandability

## 1.5 Scope

The scope of our system as the name indicates, it is in our country Ethiopia and specifically on the newly emerging long distance rail way transportation mechanism. And more specifically we are focusing on the ticketing aspect.

But when we say we focus on the ticketing aspect we aren't narrowing the scope on that aspect only there are additional features that are available. Like cites to visit, hotels to stay on, near by activities, and other activities.

As this rail way is developing and has a lot of upcoming destinations it needs update. So, it will incorporate a lot of city activities and a lot of destinations.

### 1.5.2 Scope of the system based on the technological aspect

This systems scope rotates around the automation of the Long distance Rail way system. This system now a days have one route but when it grows in the next few years it will have different routes. This now a day's route is from Addis Ababa to Dire Dewa to Djibouti and the vice versa.

This system will be developed using a web based and mobile based applications. There will be two interfaces to the web system one for the users or customers and the other for the service providers that will provide the features.

The web application for the users will have the ability to provide information about the rail way system and also it will have the capability to ticket or book our travels.

The mobile application will have less information compared to the web-based system but it provides necessary information and also provide booking and ticketing options. The mobile applications are developed because of their mobility.

The second interface of the web-based application that is used for the service providers will have multiple features like ticketing the booked event, ticketing an event where the customer didn't use the systems we provided. And provide information for the customers based on their requests.

## 1.6 Limitation of our system

There are a few limitations to system like

- We were planning to add an online payment system for the mobile apps, but for some security reasons banking systems won't provide the necessary API's to provide the requested features.
- The other limitation is we won't be able to provide the mobile application with the feature of being installed in an IOS systems. We will do the feature on android devices.

This are some of the limitations but we will try our best to solve these problems by solving the necessary requirements like the security aspects. And also, we will try our best to include other mobile operating systems.

## 1.7 Methodology and software tools

Lets start with the methodologies we are using: -

We are using Agile software developing mechanism for the sake of the system being able to grow from time to time.

And also we are want to use MVC architecture on the code implementation. While we talk about code implementation we are changing from structured coding style to object orientation.

Some of the software tools we are going to use are: -

- Visual studio code
- Sublime text editor
- Wamp or Xamp server
- Git
- Android studio
- Simulators for android development

Some of the programming languages are: -

- Java for mobile programming development
- HTML
- CSS
- Bootstrap
- PHP
- JavaScript
- MySQL DB
- R programming

# Chapter 2

# System requirement documentation

## 2.1 Introduction

As mentioned in the proposal we are going to develop a system that is going to be used to book, ticket and view available information's about the newly emerging long-distance train in Ethiopia.

And this documentation is going to include the description of the features the system is going to include. In addition, it is going to include the highlight description of how the system is going behave with a certain user and a certain criteria.

This documentation will also include the functional and non-functional requirements of the system. Like also mentioned in the above statements it will show the interaction of the user to these requirements. And also, it will tell or define how the system will work based on these requirements.

To conclude, this documentation is about the systems specific requirements. Which generally describes the interaction of the user and the system and how they behave statically without detailed data definition.

## 2.2 Background of the system

In this section we are going to discuss the previous working system and how the interaction was with the users with the given system.

We developed our system because the we believe there is a huge wastage of both the staff's resource and the user's resource as well. And our proposed system will solve some of the problems that the current working system will provide.

So, when we come back to how the current working system works is that, Most of the functionalities are done manually. And users must arrive at ticketing stations to do their bidding.

These requirements are gathered first hand from the working staff of the organizations. And some of these requirements are coming from some of the users perspectives who have done things in real life. Meaning the users have to come to the offices manually and get information and get their services.

Data management was done manually or done partially done with partially done with the help of computer programs like excel and other programs.

The user's access to the system is limited to the ticketing offices. There was no remote access. There were no mechanisms of accessing the services provided by the organization.

These all problems are making the life of the users uneasy and the organizations data management is very unorganized. So, the systems we are providing manages the problems that are provided above.

## 2.3 General and specific Objectives

The objectives of the system we are going to develop will solve the given problems of the current working environment. Those objectives can be classified in specific and general objectives.

The specific objectives: - are objectives that solve a specific problems that the organization and users provide. E.g.  book tickets.

The specific objectives provide the main functionalities that the system provide. This doesn't mean the that the general objectives aren't main functionalities, they just are view of the major perspective. Or General objectives provide with the bigger picture.

Here are few general objectives of our system.

- ⇨ Automation
- ⇨ Easy data management
- ⇨ Availability
- ⇨ Effectiveness and efficiency
- ⇨ Resource management

And some of the specific objectives are listed down below:-

- ⇨ Ticket management and reservation
- ⇨ Report generation
- ⇨ Security
- ⇨ Manage payment activities
- ⇨ Provide extra information on destination cities
- ⇨ Provide Information (history and new news)
- ⇨ Employee management

These are the basics but through time and during code implementation this objectives might get broader even there might be additional objectives that the system will uphold.

Other features will be discussed later in the functional and non-functional requirements are defined.

## 2.4 How the system works

In this part we will discuss briefly how the system is going to work and how many actors there are. And Also, we are going to see if our system will interact with external environment.

To start in general manner, the system will have three parts. The first part there is web application for the user. This web app consists of different features like book ticket, book and pay ticket, view travel information, manage seat information, choose class, and other ticket management issues for the long distance Ethiopian rail way.

This web app doesn’t require any login and sign up formats because any user can access it and use it for their needed purpose.

The second system is still again a web but this web is developed for the admin and staff workers of the organization. And this manages the user activity both on their remote activities and manual transactions in the ticketing offices.

The admins manage both the staff workers(employee management) and also manages the users interactions with the staffs. But the user is mainly interactive with staff members when they arrive to their respective offices.

The admins have privileges to see the individual staff members activity. Manages the accounts of the employees. And able to see reports generated automatically by the system.

The staff members are able to reserve tickets, convert booked tickets to reserved tickets, cancel reserved tickets, change reservation ticket information, view available information’s about future tickets, and other features that might be added through time. To do all these activities all staff members and admins must login.

We are implying that there might be additional features is that because we are using agile software development methodology. And this technique will help us add up new features to the system through the development process.

The third system is a mobile application for the users. It has mostly the same features as the web application which is implied for the users but with a little bit more added features.

This mobile application increases the mobility of the system in a big way. Because of this mobile app a lot of users will become more eligible for this system.

As the technology increases the want for big system to be in small device is increasing dramatically, our system will fulfill these wants. And also gains more attraction from the society.

There are interactions with external systems for the sake of few reasons. One of the reasons is, there might be some needed information about services that are available near our destination areas and there might be links that forward us to their pages and systems.

The other reason is we are asked to develop a payment mechanism that is going to be integrated with our system. so these payment system needs to be externally integrated with our system like an API. And we might use API's that are provided by online banking systems. Or a simulation web services that will act us our payment mechanism.

## 2.5 Functional and non-functional requirements

To identify the functional and non-functional requirement we used the following criteria's:

- The functional requirements are the requirements that construct the basis of the system
- Functional requirements are the requirements that we get directly from the stakeholders required functionalities.
- The non-functional requirements are the requirements that we gather indirectly from the stake holders.

If we say the following concepts lets get to the listing of those requirements.

Functional requirements

- Book a ticket
- Reserve ticket
- View available sites
- View available travels
- Pay for booked tickets
- Add new travel information's
- login
- cancel travels/reservations
- change reservation information
- view previous trips
- view reports
- add and remove employees

Most of the functional requirements are listed above. The non-functional requirements are:

- Availability
- Mobility
- Easy access
- Understandability
- Fast and reliable
- Security
- Maintainability
- Reusability
- Scalability
- Lower down time

## 2.6 Models

In this paper there are going to be three models that are used to represent the structure of the system.

The first model is use case model this model represents each and every interactions of the users to the functionalities they access.

This model also defines and describes about the use cases. Meaning it gives brief description about each use cases like who initiates them other actions they take, flow of events alternative events and others.

The other model is the class model. This defines the data structure of the system like, the classes it holds and attributes it consists of and the methods it holds.

This model is very necessary and mandatory for developing the code aspect and also to create the database of the system.

The last but not the least model used to represent our system in this paper is deployment model.

This model shows the placement of components or devices while the system is running. This model helps us to understand the working mechanism of the devices.

## 2.6.1 Use case model

Actors

- User(can be using both web and mobile app portals)
- Staff member
- Admin

Use cases for each user

Users

- View available travel info
- Book a ticket
- Reserve a ticket
- Pay for ticket
- View available extra features

Staff members

- Pay booked tickets for users
- Reserve tickets for users
- Insert user info
- View available travel information
- Cancel reserved tickets
- Change reserved ticket info
- Login
- Log out

Admin

- Inset news
- Login
- Logout
- View reports
- Add staff members
- Remove staff members

## 2.6.1.1 Use case diagrams

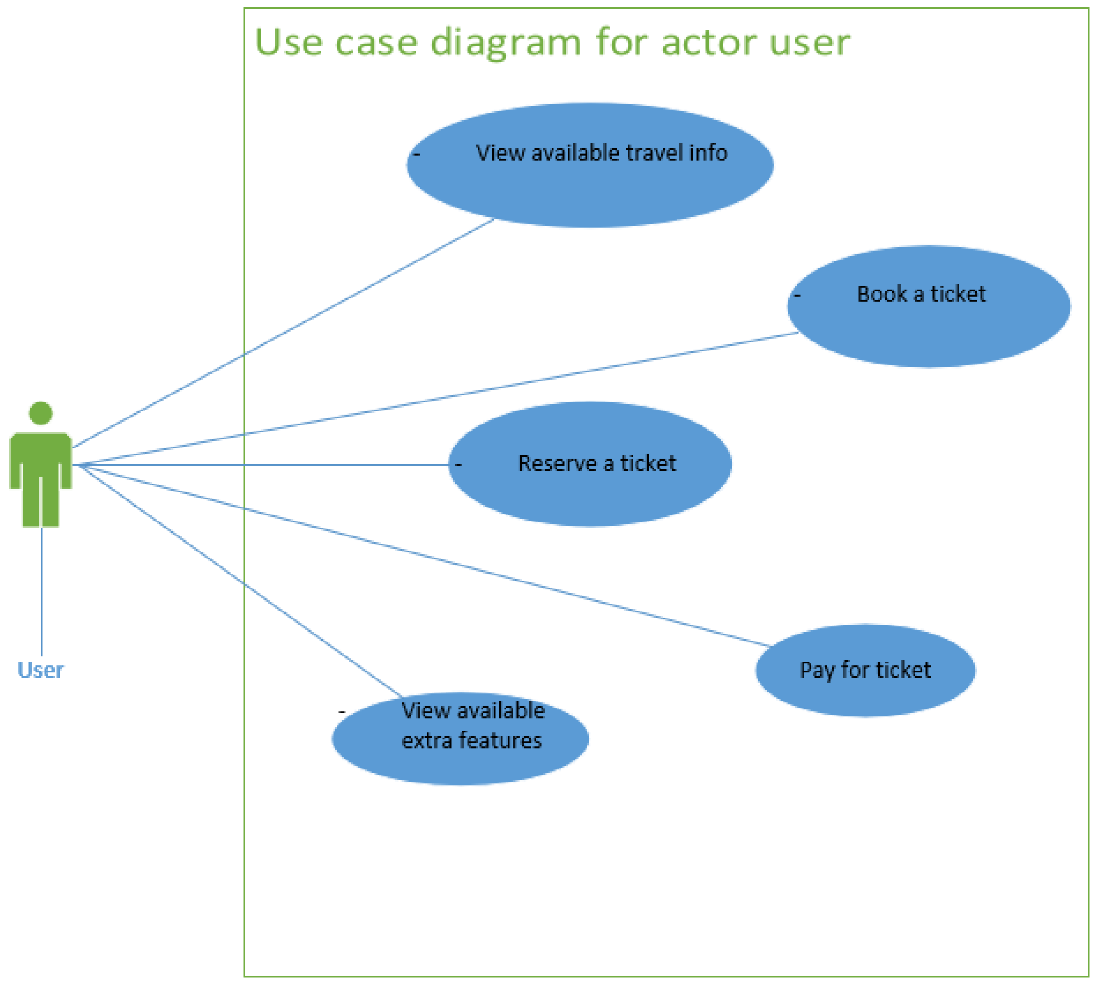
Use case diagram for actor user
View available travel info
Book a ticket
Reserve a ticket
Pay for ticket
View available extra features
User

Figure 2.6.1.1.1 use case diagram for user actor

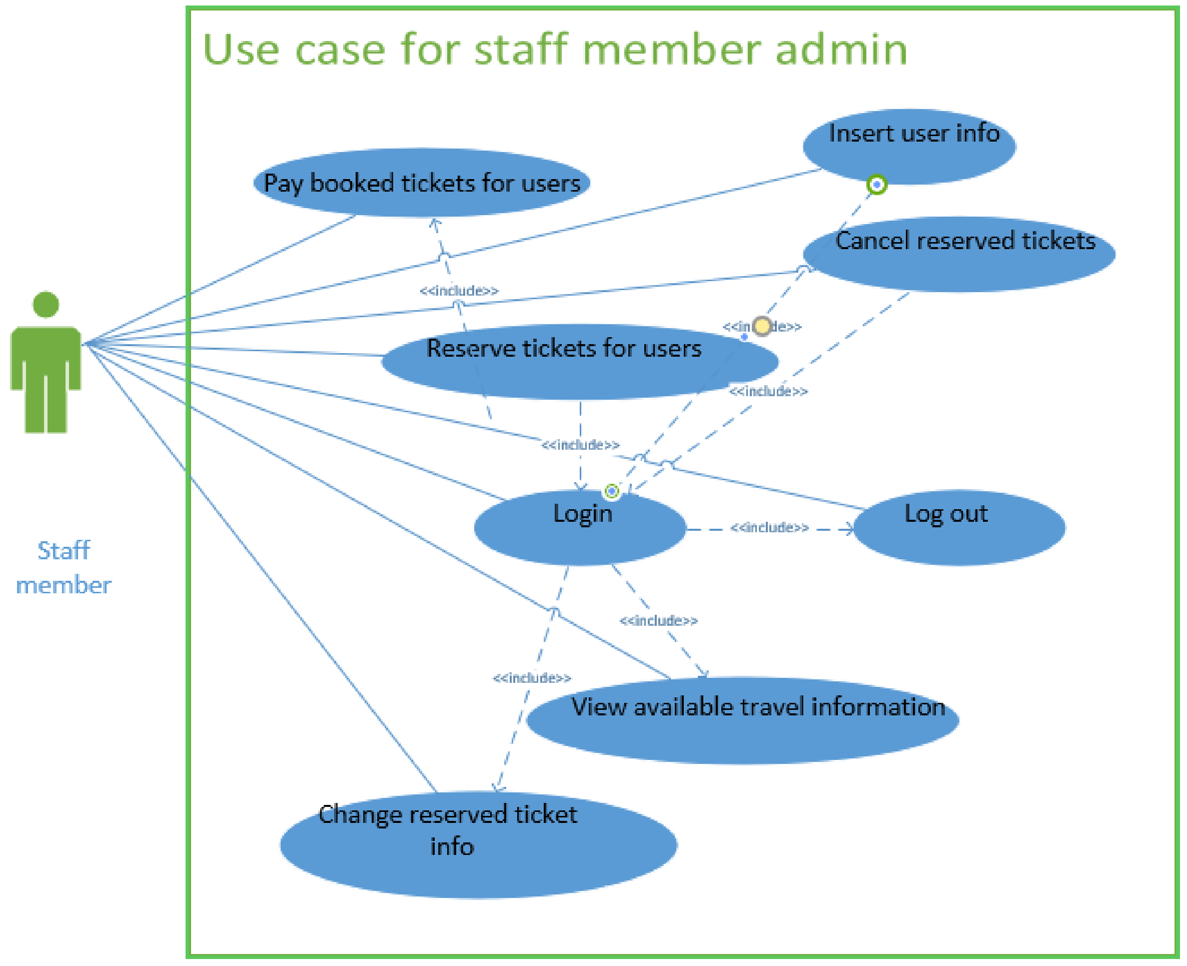
Use case for staff member admin
Insert user info
Pay booked tickets for users
Cancel reserved tickets
<<include>>
Staff member
Reserve tickets for users
<<include>>
<<include>>
<<include>>
Login
<<include>>
Log out
<<include>>
<<include>>
View available travel information
Change reserved ticket info

Figure 2.6.1.1.2 use case diagram for staff members

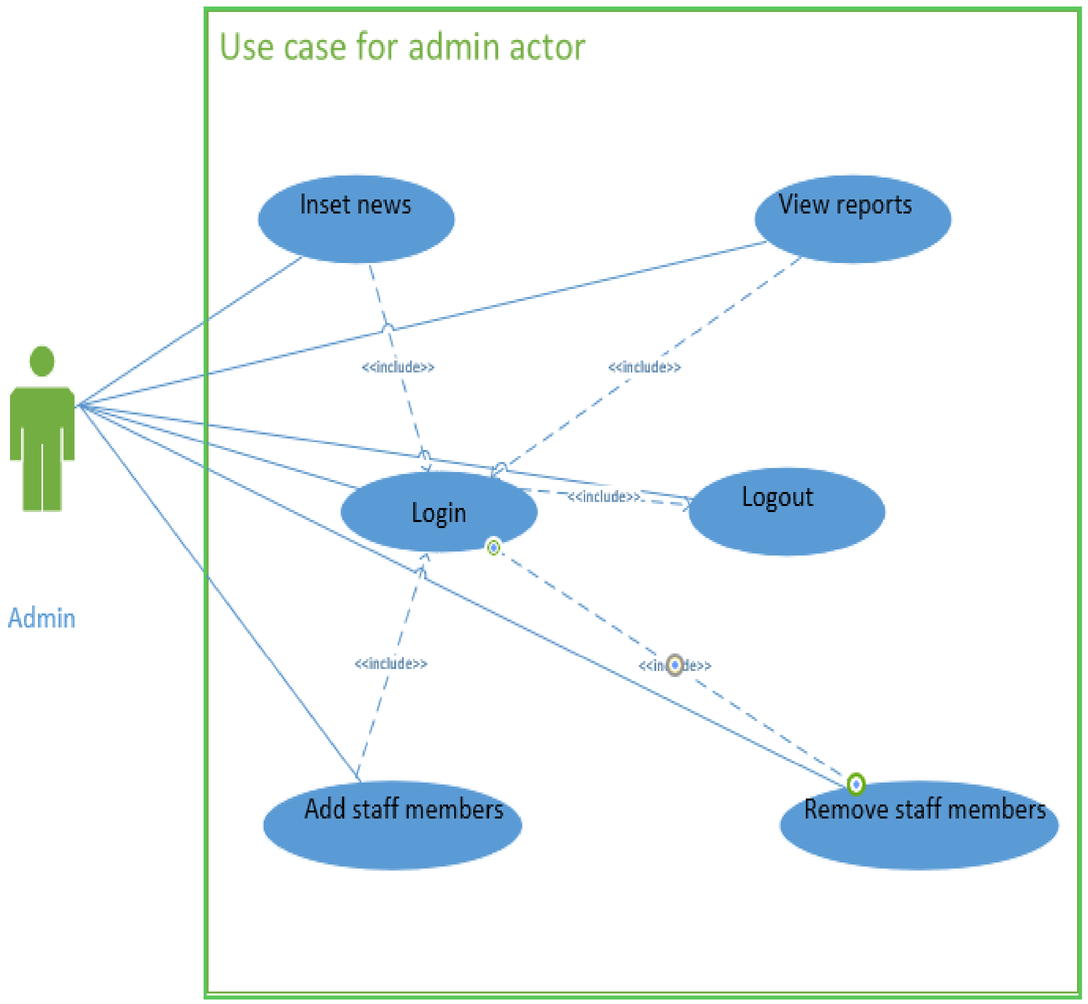

Figure 2.6.1.1.3 usecase diagram for admin actor

## 2.6.1.2 Use case descriptions

| | |
|---|---|
| Use case id | UC1 |
| Name | Book a ticket |
| Description | Books a ticket of a travel for later use |
| Actor | user |
| Pre-condition | Should select a single travel information from the available travel information |
| Post-condition | Booked ticket must be paid for the ticket to get justified |
| Main success scenario | 1. The user accesses a web or mobile app portal<br>2. The user fills the necessary form<br>3. The system validates the form and checks for available travels on that inserted information<br>4. The system displays booked information to user for validation.<br>5. The user accepts the displayed information.<br>6. The system saves the inserted data to the Database for later use. |
| Alternative scenario | 4.1 the system sends back no available travels on that specified information.<br>4.2 the user reinserts the form to check available travels. |

| | |
|---|---|
| Use case id | UC2 |
| Name | View available travel info |
| Description | Displays available travel information |
| Actor | user |
| Pre-condition | none |
| Post-condition | Make information available to book and reserve a ticket |
| Main success scenario | 1. The user accesses a web or mobile app portal<br>2. The user fills the necessary form<br>3. The system validates the form and checks for available travels on that inserted information<br>4. The system displays available information to the user.<br>5. The user selects the displayed information.<br>6. The system saves goes to booking or reserving pages. |
| Alternative scenario | 6.1 user can cancel and get back to home and other pages. |

| | |
|---|---|
| Use case id | UC3 |
| Name | Reserve a ticket |
| Description | To reserve new ticket for travel information |
| Actor | User and staff |
| Pre-condition | If staff must login. |
| Post-condition | Ticket number and another user information's are displayed and stored. |
| Main success scenario | 1. The actor requests to reserve a ticket<br>2. The system provides the necessary form to be filled.<br>3. Actor fills those forms then submits<br>4. The system verifies and send a confirmation message and requests user to go to payment mechanisms<br>5. The actor then go to payment mechanism. |
| Alternative scenario | 4. The system may send error messages if the filled information was incorrect. |

| | |
|---|---|
| Use case id | UC4 |
| Name | Pay for ticket |
| Description | Payment for booked or reserved ticket |
| Actor | User and staff |
| Pre-condition | For payment of booked ticket user must have booked a ticket using web or mobile app portal.<br>if staff must login. |
| Post-condition | Ticket number and another user information's are displayed and stored. |
| Main success scenario | 1. The necessary actor initiates the payment mechanism.<br>2. The Actor Chooses from booked ticket and reserved ticket.<br>3. If booked is chosen the system provides form to fill about booking information<br>3.1 The actor fills booking information and submits.<br>3. 2 The system checks the input values and validates and sends the confirmation message.<br>3.3 The actor gets the full travel information and gets print out.<br>4. The system accepts the reservation information and provides form to fill about payment information<br>4.1 the system verifies and send confirmation.<br>4.2 The Actor get the full travel information and gets print out. |
| Alternative scenario | 3.2 and 4.1 The system will send error message and get back to the page if any wrong information is inserted. |

| | |
|---|---|
| Use case id | UC5 |
| Name | Cancel reserved ticket |
| Description | When a user needs to cancel there travel plan and the staff member executes there wish |
| Actor | Staff |
| Pre-condition | staff must login and user must have a reserved ticket |
| Post-condition | Ticket will be cancelled. |
| Main success scenario | 1. The system provides for the insertion of ticket number.<br>2. The staff member inserts the users ticket number.<br>3. The system checks if the inserted information is correct<br>4. Then the system will calculate the punishment fee and will show that and the return fee.<br>5. The staff member accepts and proceeds. |
| Alternative scenario | 3. The system might return error information like the booking number not correct or the booking has expired. |

| | |
|---|---|
| Use case id | UC6 |
| Name | change reserved ticket info |
| Description | When a user needs to change there travel plan and the staff member executes there wish |
| Actor | Staff |
| Pre-condition | staff must login and user must have a reserved ticket |
| Post-condition | Ticket will be changed to new travel plan. |
| Main success scenario | 1. The system provides for the insertion of ticket number.<br>2. The staff member inserts the users ticket number.<br>3. The system checks if the inserted information is correct and provides a form for the new travel plan.<br>4. The staff member fills the form and proceeds<br>5. Then the system will calculate the punishment fee and will show that and the new fee.<br>6. The staff member accepts and proceeds. |
| Alternative scenario | 3. The system might return error information like the booking number not correct or the booking has expired. |

| | |
|---|---|
| Use case id | UC7 |
| Name | Pay booked tickets for users |
| Description | Pay and reserve the ticket of users for later use |
| Actor | staff |
| Pre-condition | staff must login and user must book a ticket |
| Post-condition | Ticket will be reserved and be available to use. |
| Main success scenario | 1. The system provides for the insertion of booking number<br>2. The staff member inserts the user booking information<br>3. The system checks if the inserted information is correct<br>4. Then the system will display the users information and request to proceed to payment.<br>5. The staff member selects h proceed button.<br>6. The system will provide a form for payment information.<br>7. The staff member will fill the payment information.<br>8. The system will display all information and prints out the ticket. |
| Alternative scenario | 3. The system might return error information like the booking number not correct or the booking has expired. |

## 2.6.2 The sequence and activity model

This model reprsents some what more dynamin adpet of the system. it will show how the system works in a given condition.

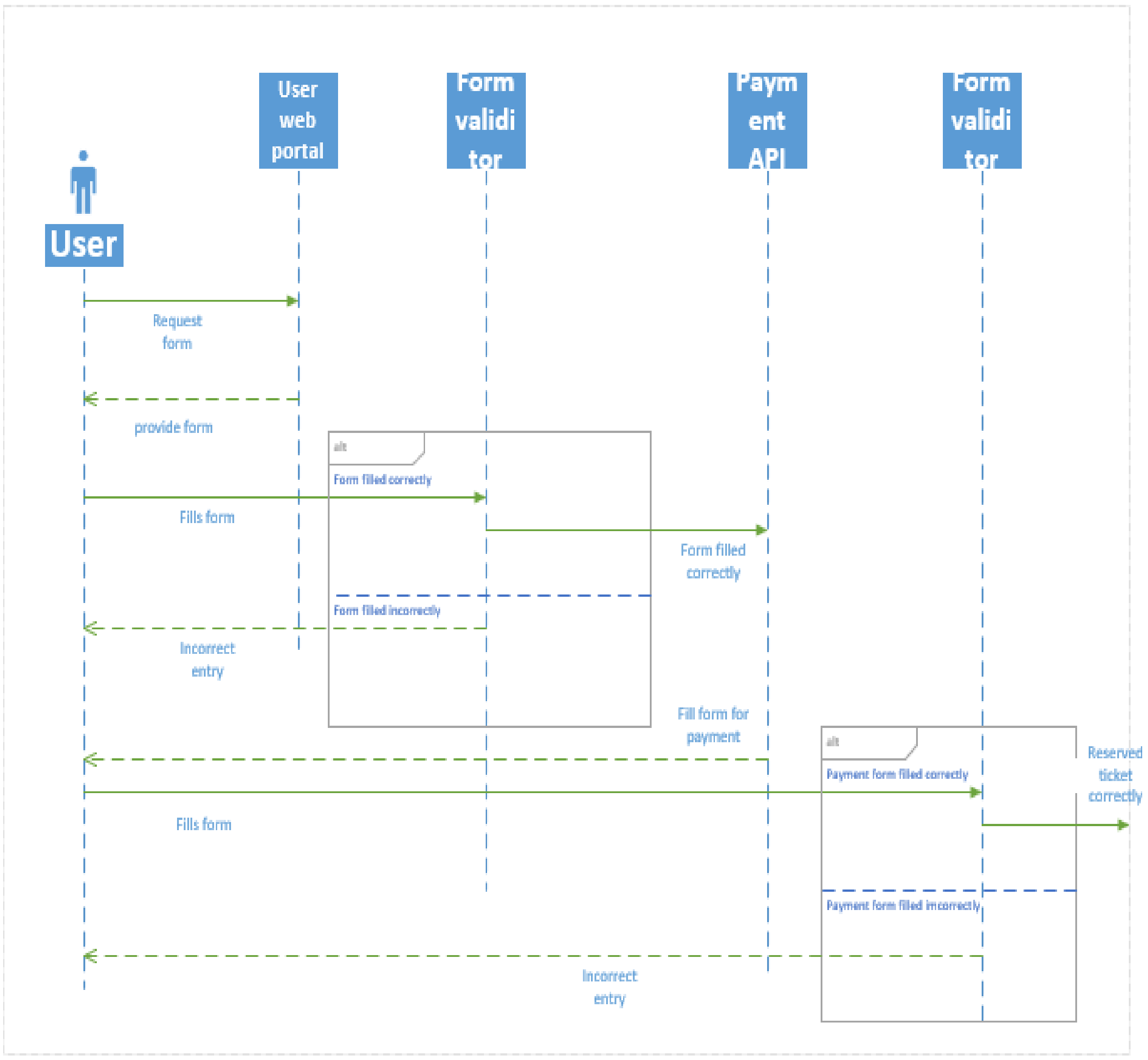

Figure 2.6.2.1 sequence diagram for Reserve a ticket

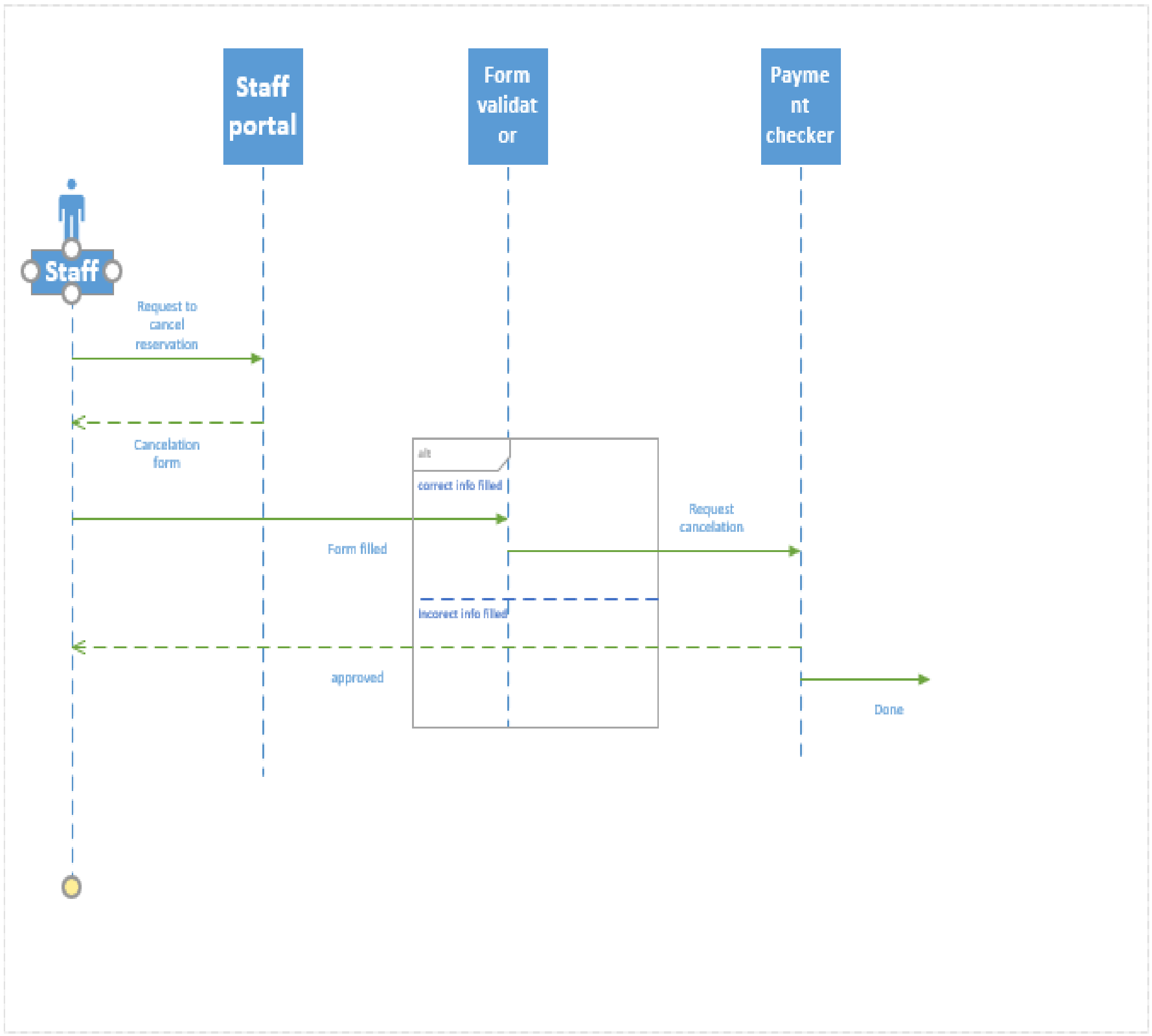

Figure 2.6.2.2 sequence diagram for cancling reservation

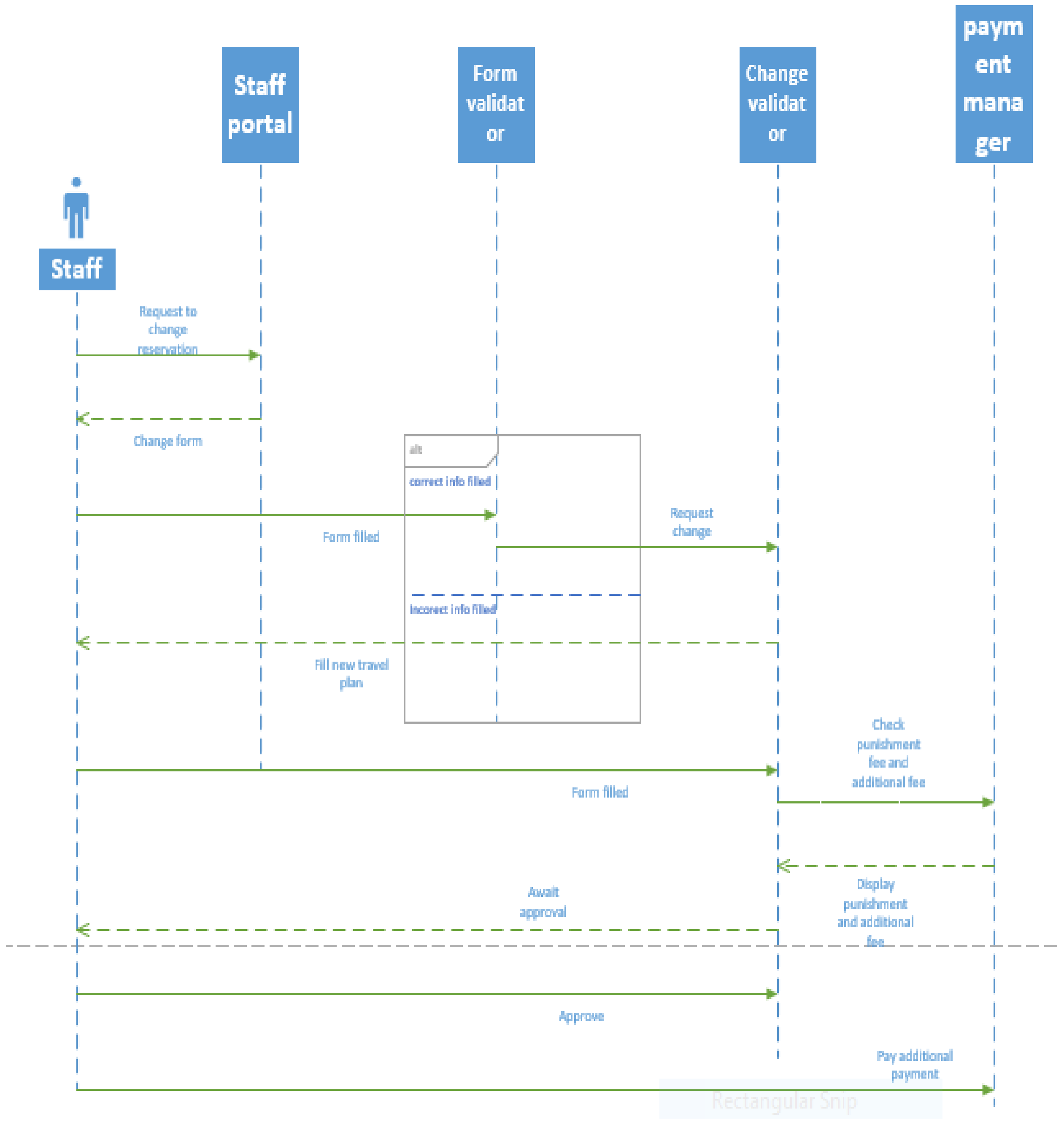

Figure 2.6.2.3 sequence diagram for change reservation

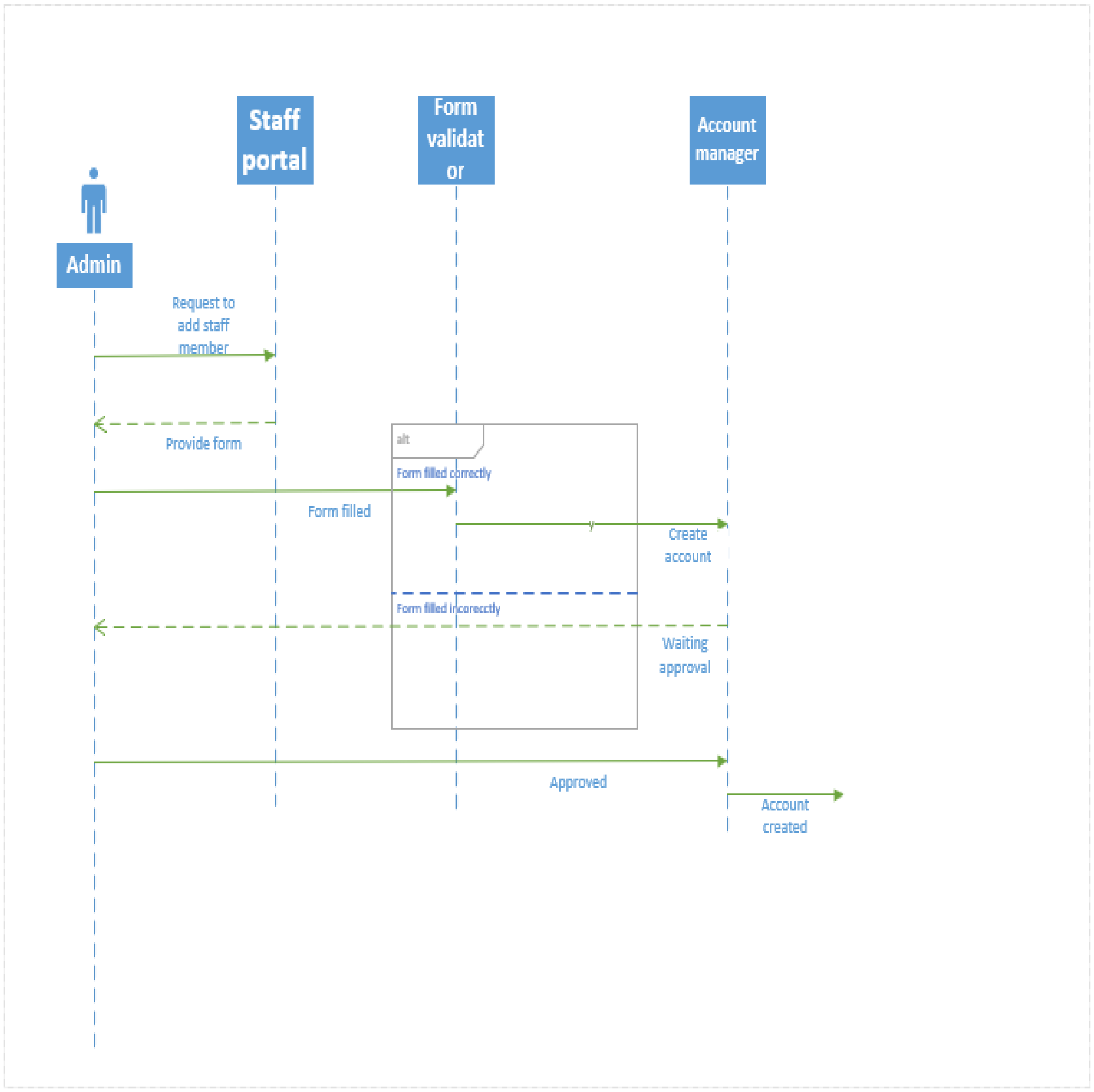

Figure 2.6.2.4 Sequence diagram for creating account

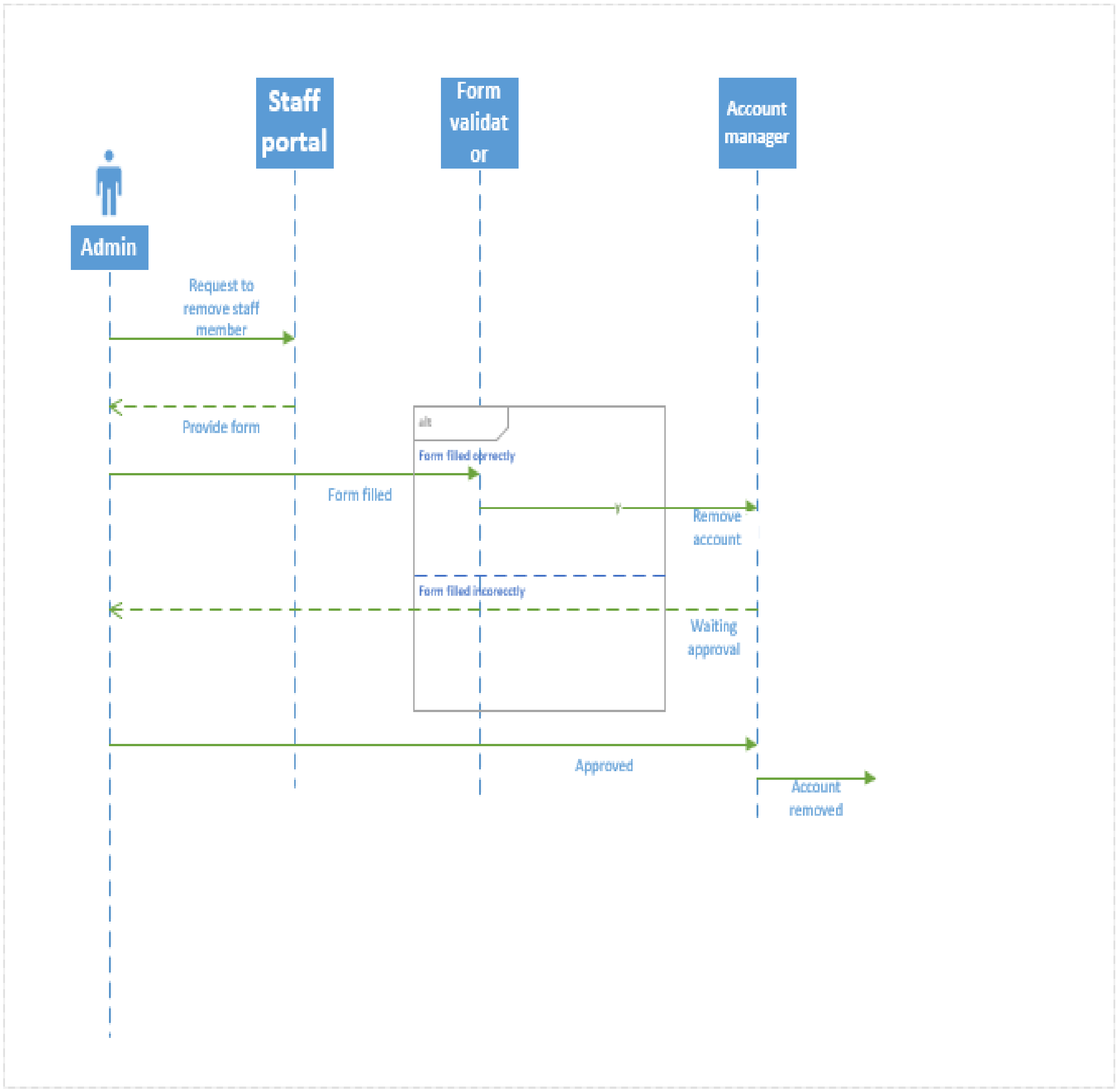

Figure 2.6.2.5 Sequence diagram for removing account

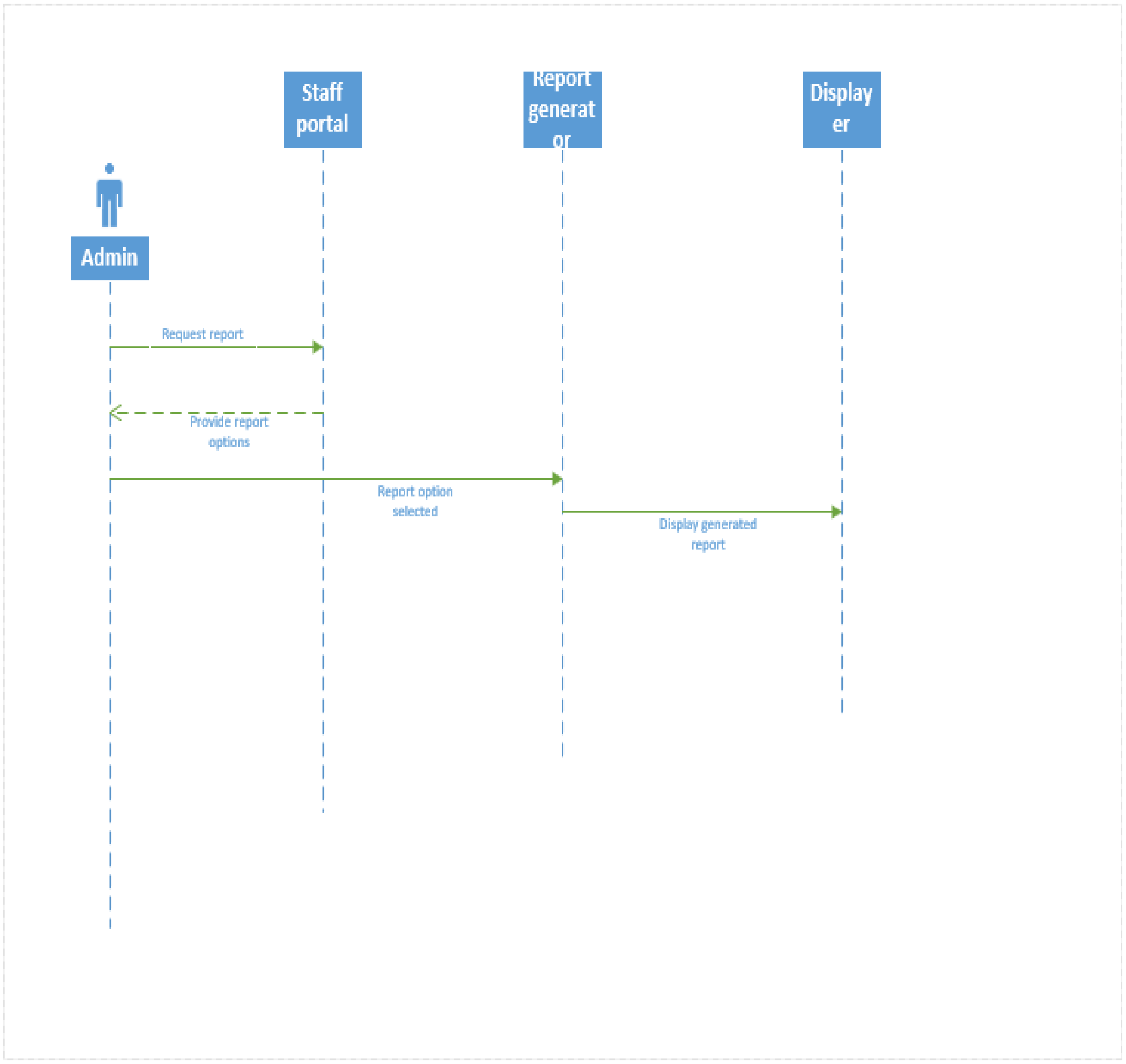

Figure 2.6.2.6 Sequence diagram for Viewing report

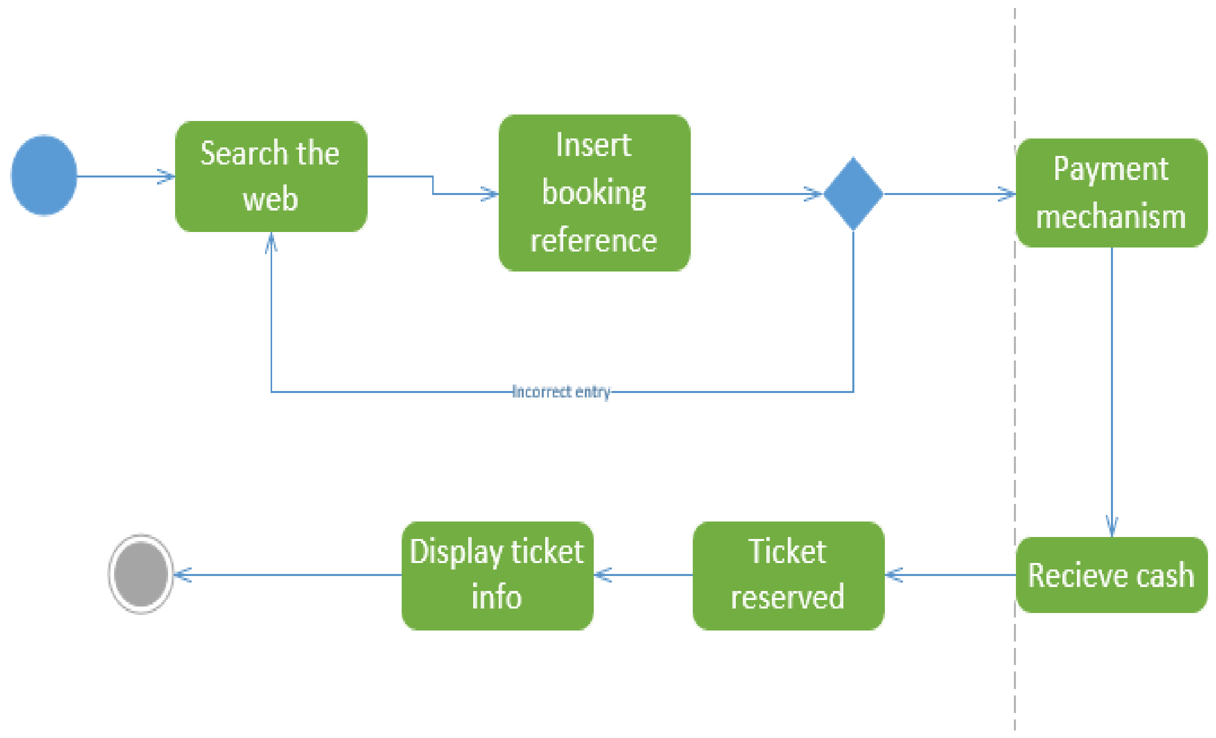

Figure 2.6.2.7 Activity diagram to pay for booked ticket

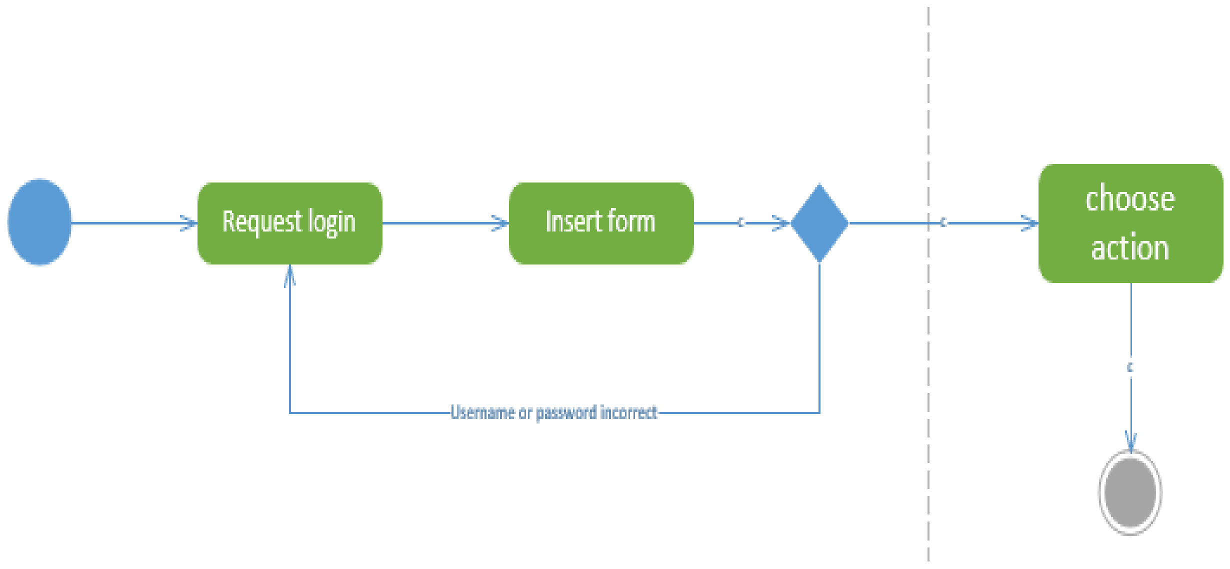

Figure 2.6.2.8 sequence diagram to login to the system

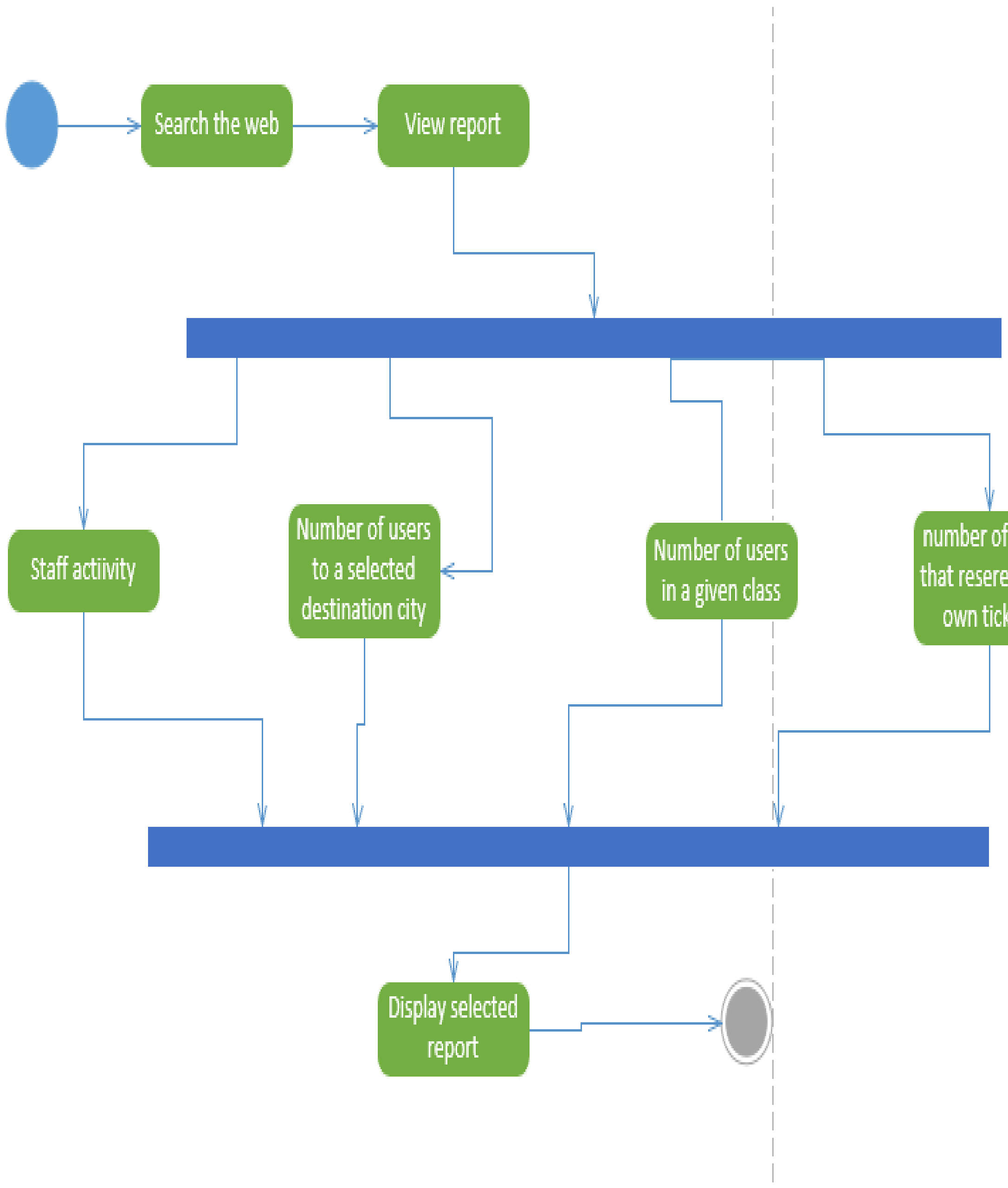
Search the web
View report
Staff actiivity
Number of users to a selected destination city
Number of users in a given class
number of users that resere there own tickets
Display selected report

Figure 2.6.2.9 activiy diagram to view reports

## 2.6.3 Class model

This model is used to reperesent the dta structure of the system and the below diagrams and modeling concepts reperesent our system data concept.

The classes of our systems are: -

- Staff: - this class stores and manages the information about staff employees and admins.
- Users: - this class is modeled to manage and handle user informations
- Reservation: - is the class that is created to manage the events that are related to reservations.
- Payment: - this class holds and manages information related to payment mechanisms
- Routes: - this like the above class manages its own attributes and methods which are related to there own features, which in this case is route infromations.

Relationships

- A reservation can only have one payment mechanism.
- A reservation has only one route information.
- One user can make multiple reservations.
- One staff member can make multiple reservations.

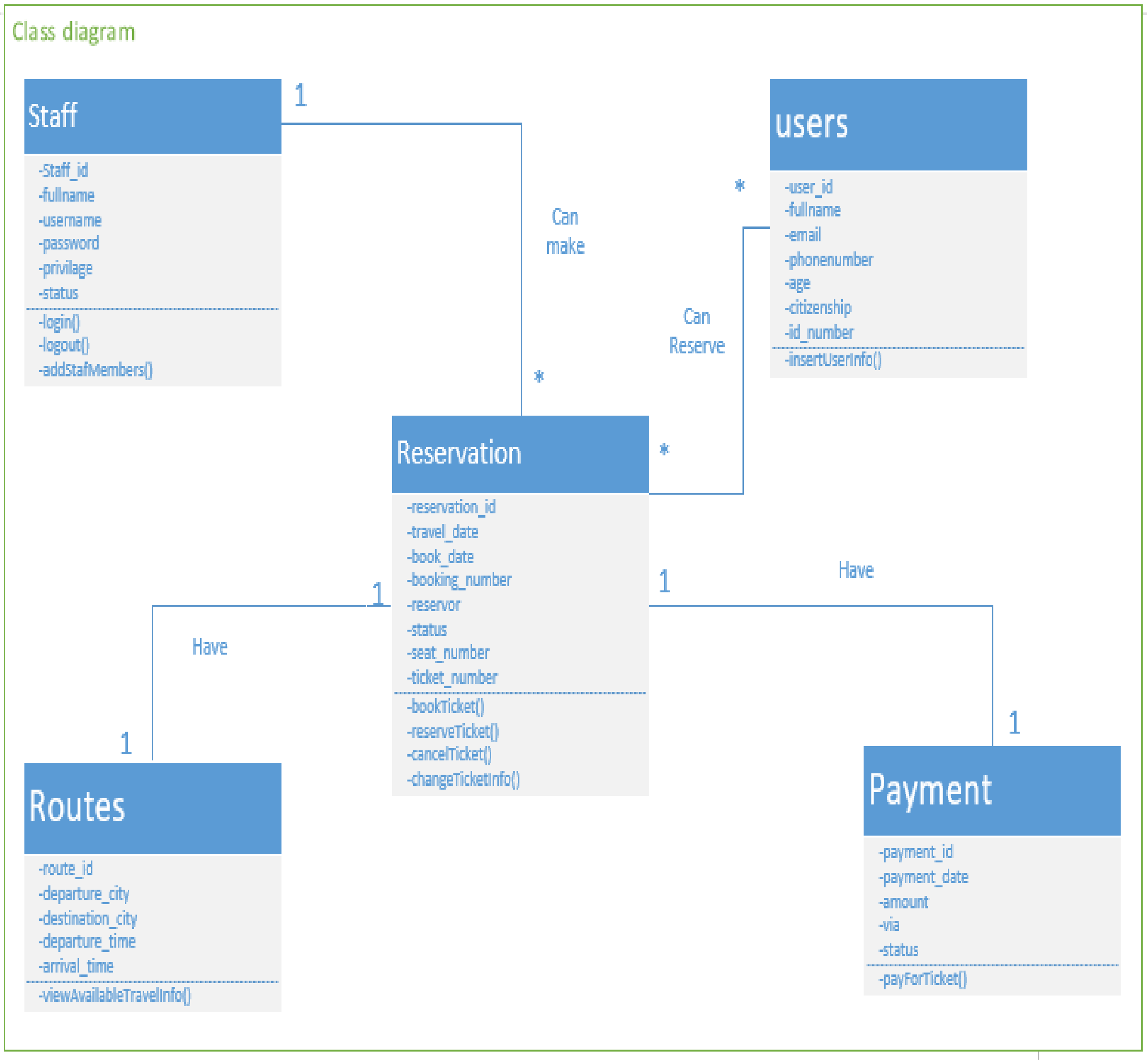

Figure 2.6.3.1 class diagram

# Chapter 3
# System design documentation

## 3.1 Introduction

This chapter is going to include the architectural description of the system. in addition to this it will have the description and decomposition of the modules.

This paper is all about the architectural and structural view of the system. This is very important to the software development because it will have a brief description of the system that is going to be built and also it will have the detailed knowledge on the modules that are going to be developed.

Some of the topics covered here are going to be directly converted to the implementation phase or the concepts are going to be converted to code.

The limitation of this documentation is, as said in the SRS documentation the concept of design we are using to develop this system is agile so, it is going to be very dynamic and changeable. In addition to this while development some changes may be done without the guide of this document so the listed configuration of the system down below my change through time.

To conclude, this documentation says to the developer how the system will work. It even tells how the modules are classified and how they work.

## 3.2 Architecture overview

The architectural style used here is going to be the most commonly used and most understandable way of implementing currently emerging systems, which is the MVC architecture.

Which this model is generally all about division of the work of classes. It has three components: -

- The view: It is the component that is responsible for displaying the information of the system. in our system, the web uses html and partly php and in the android system it uses XML. It is used for interaction between the user and system. or it is generally called us front end development.

- The Controller: This component is the component where the data interaction is available between the view(front end), and the database. This component accepts data from the users manages it and sends it to the model, the model analyses the data in to query and does the CRUD operation and then sends back to the controller and this controller sends the organized data to the view. In our web system this is done using PHP, in the android system it uses PHP and java.

- The Model: This component is the part where there is interaction between the data that came from controller and the database. It changes the data that came into a query for the sake of data manipulation from database. In our web system this is done using PHP, in the android system it uses PHP and java.

## 3.3 The Component models

This particular section deals with the component model which aspires to explain the subsystem decomposition in detail. And it describes the components of the system. These are the few components.

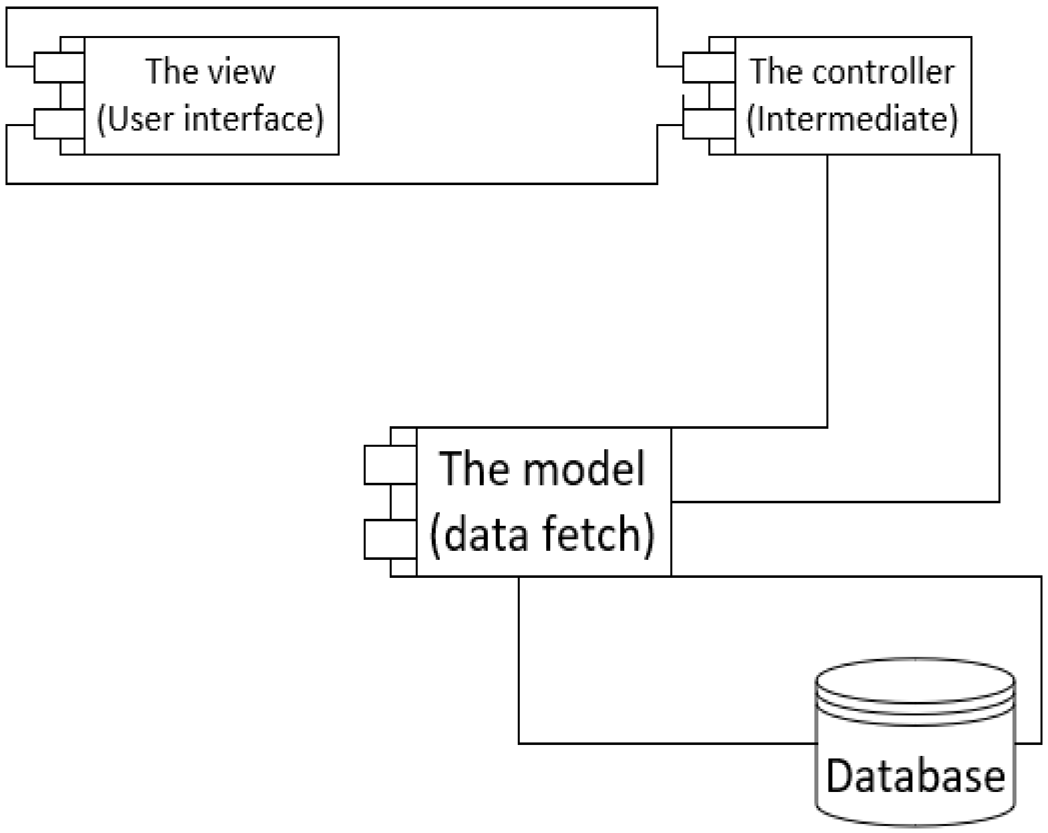

Figure 3.3.1 Component diagram for the MVC architecture

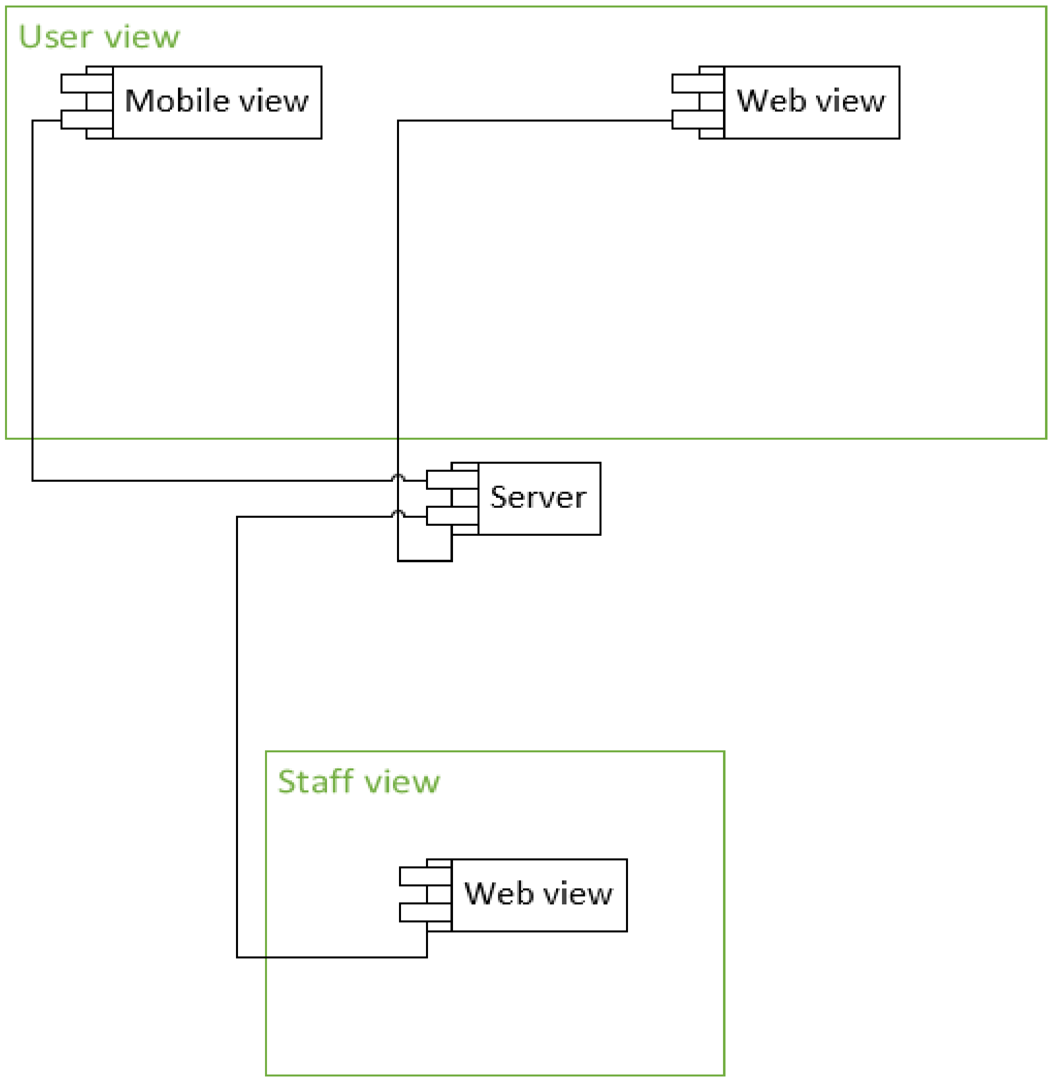

Figure 3.3.2 component diagram for user and staff members interaction with the system

## 3.4 The deployment models

In the deployment aspect we are thinking of using the 3-tier architecture. Because we need our system to get no bottle necked and crowded. Instead of describing it, the below diagram will tell more on how it works.

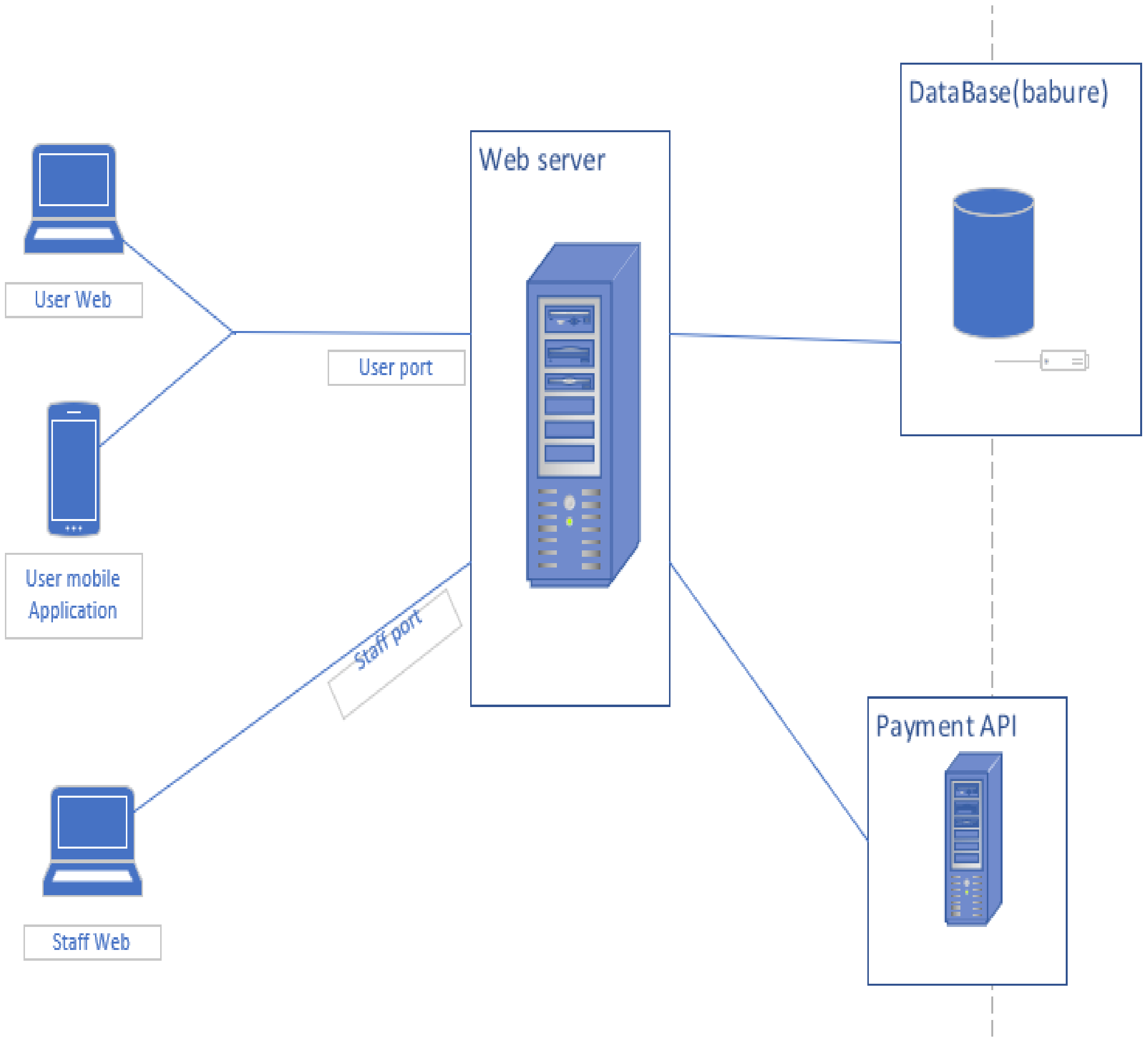

Figure 3.4.1 deployment diagram

## 3.5 Module Decomposition

This is a concept that deals with the division of labor or in other words the distribution of load in to sub-systems or components. This is very helpful to manage and work with, especially for testing and implementation. Because the decomposition makes it easier because a given component will only have one specific task, that makes it easily manageable.

In this part we will see each modules description and internal working of the decomposed systems. Like the data that will accept and the information that give or what it does with the given input data.

In our system there are a lot of modules, so we used a technique to decompose the system. and that system is we created components and put the features that are similar together for easy management.

Some of the components are listed below: -

- Login
- Reserve Ticket
- Book a ticket
- Create account
- View travel information
- Change reservation
- Cancel reservation
- Pay for booked tickets

These are the major components that we think are used to make the system full and work in a well-organized manner.

### 3.5.1 The Login component

This component as the name indicates is responsible for managing the activities of personals that access the system.

This component is used to manage staff members activities and give them privileges that are needed by them to do their work.

Also, this sub system is very important and must be well implemented because it our main line of defense from intruders. No user will have access to the staff portal if they don't have access given to them by the admin.

It uses normal username and password combination to identify the personal who is accessing the system. It will also provide id numbers for the next page because the system has to record that is done and what is being done.

There is the usage of session for two purposes. One, for the sake of security. Because no one is allowed to enter without the insertion of the username and password. And this session security helps us block any intruder with no privileges and there is no transition from one page to another without the session managing it especially from the home page of the staff portal to the login page and vice versa. Second, this session is used to identify who is accessing the system. it is mainly used in report generation and log reports.

The input parameters for this module or components are username (string datatype) and password (string datatype). The personal that access this component should insert this data's and get authenticated. These parameters go through the normal architecture style meaning the view is responsible to display the form, the controller fetch's the data from the user interface then sends it to the model, then the model uses the data and generates the query and fetch's data from the database. Whilst checking it also validates if the user is authenticated and sends the user id to pass to the session.

This module uses the query operation select from the "staff account" table with the where condition that checks if the username and password are equal to the inserted data above.

The testing mechanism we used are manual testing but if we get time and finish our projects on time, we might use php unit testing for this module. But our major testing mechanism is inserting data manually and checking if they will pass or not. The other testing mechanism is we insert the query directly to our MySQL DB and test it and verify the result.

### 3.5.2 Reserving a ticket component

This sub system deals with the major functionalities of the system. As the name indicates it is all about reserving a ticket.

This component can be both done by users and staff members. Where users reserve their own ticket using a user portal (web or mobile app). But while doing this some of the inserted data's and some of the mechanisms might differ.

This component will require the login component. Meaning the staff members must login before managing user's reservations.

As, this is the core component of the system its functionalities are implemented in a very serious and well-established manner. Other components will interact with it in different aspects and other components will use its output for their input mechanism.

To explain how it works. It starts by making the reservoir insert arrival date (date datatype), Arrival city (string datatype), departure city (string datatype). Then after insertion the system checks the seats left in each class levels (Hard sleeper, soft sleeper, normal) and displays the available seats to the user. Then after those are inserted, the user information is inserted by the reservoir. While that insertion of data is done it is only left with the payment mechanism. If the user is paying using online mechanisms, he /she will insert payment information in the API that provides the system will payment options. Or if the staff is managing the ticket and the user is paying using direct payment the staff accepts the cash and confirms payment mechanism.

When the staff members are managing the ticket for users, there id will be stored. Because when later use in report generation I will be needed. And if the user themselves were managing their own ticket, "user" will be stored in the reservoir attribute.

After each action are completed this module will produce ticket number also the ticket itself. This ticket will include the user's information, seat number, class, ticket number. This are helpful later when the users are boarding their train. And also, the ticket number is also necessary in other components described below.

This module is going to use three tables "reservation" table, "user info" table and payment table for inserting data. And also, it uses the "route" table to get route id that is going to be used to store the routing information. This table is predefined.

Testing mechanism for this module is the same as above. We will try to test it using manually by inserting different types of data's which can be illegible and false. And also, we will test the queries using our DB provider toc check the queries are well formed.

### 3.5.3 Create account component

This component deals with account creation of staff members. It gives staff members access to use the system.

This component is done by the actor admin. It is both initiated and managed by this actor only no other actors are allowed to get privileges. This is because if one actor gets this action's they can create accounts and forge different data. There is a security risk there.

So, when we directly go to how it works. The admin wants to give privilege to a new employee. The admin opens the system to create account. Then the admin accepts the user information like full name (string datatype), email (string datatype), phone number (string data type), then inserts it into the system to create the account. And the system creates username and password form those given inputs. These all inserted and created systems will be stored in the DB.

The username and password of the staff will be sent to the staff's email and will get its access to manipulate the system.

Testing for this component is also manual. We insert data into the system and the system will respond to those inserted data's and we verify the output of the system. And also, we check the email sender if it is working properly or not. And us the above components we test the queries to check their correctness and efficiency.

Whilst creating this module we might face challenges like the email sender module might not work, or it needs an internet connection.

### 3.5.4 Pay booked ticket

This component is done by the staff members for the user that couldn't reserve the ticket because don't have the online payment mechanisms.

This sub system needs a pre requirements like the user before coming to the staffs counter must book a ticket and the staff must login as well to do the needed action.

The staff members accept the booking reference of the user and checks if the reference number is correct or not. If it is correct, he/she will be forwarded to payment option. And the payment option will be cash only because it is direct payment to the staff's counter.

After this is done the booked status will turn in to reserved and payment status will be active. And the user can travel to his destination city.

Testing of this component can be done using the previous means like entering data to this component and checking its results.