EEC 521 – Software Engineering

SOFTWARE PROJECT FINAL REPORT

LOCAL TRANSPORTATION TICKETING SYSTEM

Group Members

Venkatesh Manthu (2826995)

Karthik Vasu (2835576)

Laasya Sri Vanka (2832460)

Date: 12-08-2022

Table of Contents

List of Figures

List of Tables

INTRODUCTION

1.1 Purpose and Scope

The Local Transportation Ticketing system allows the users to book local bus tickets and receive a confirmation receipt online. The system provides the login access for both the users and administrator. The user can login to the application to book the ticket online. This Ticketing system includes all the required information of the local buses in all cities and include inter and intra city networks. The user can use this application on their devices and book tickets for all the local buses they wish to travel while sitting at home.

The Current ticketing system works manually and is quite a tedious process involving queues and waiting for hours together. The main objective for this project is developing an application so that passengers can book the tickets online directly from their smart phones and receive a message to their phones that's enough for travelling until the destination. Thus, the process of standing in lines to book the tickets can be completely avoided.

The user needs to select the starting point and destination of the travel. The user can also opt whether it is a one -way journey or a round trip. The admin maintains the user account balance and shows the history of journey tickets booked by the user.

1.2 Product Overview

This is an e-ticketing app for local transport like buses. This software is for booking tickets and managing reservations. It allows the user to book a ticket to any local place and time of their liking.

The software takes e-mail id or phone number as input for a primary verification to create an account. Payment should be through online transactions like net banking, or through debit/ credit cards. So, it also takes input according to their payment choice like details of net banking, or debit/credit cards. The software produces an e-ticket as an output. The major software functions of this app are elaborated below.

1) Login/ Sign up: A customer needs to create an account first, in order to book tickets for their journey.
2) Bus Schedule: All the schedules of the trains or buses are displayed in this module. This module keeps getting updated frequently.
3) Booking Tickets: For booking of the tickets, a customer has to fill up certain details like number of persons, starting point and destination, one way or round trip etc.
4) Payment: In the payment section, tickets are paid for and a confirmation ID of the ticket is generated.
5) Admin: All the updates and issuing of the tickets are done.

1.3 Structure of the document

This is a report of the e-ticketing app. This document is going to describe the project management plan followed by project software requirements, design and architecture, test plan and finally concluded with results and further scope.

REQUIREMENT SPECIFICATIONS

2.1 Stakeholders for the system

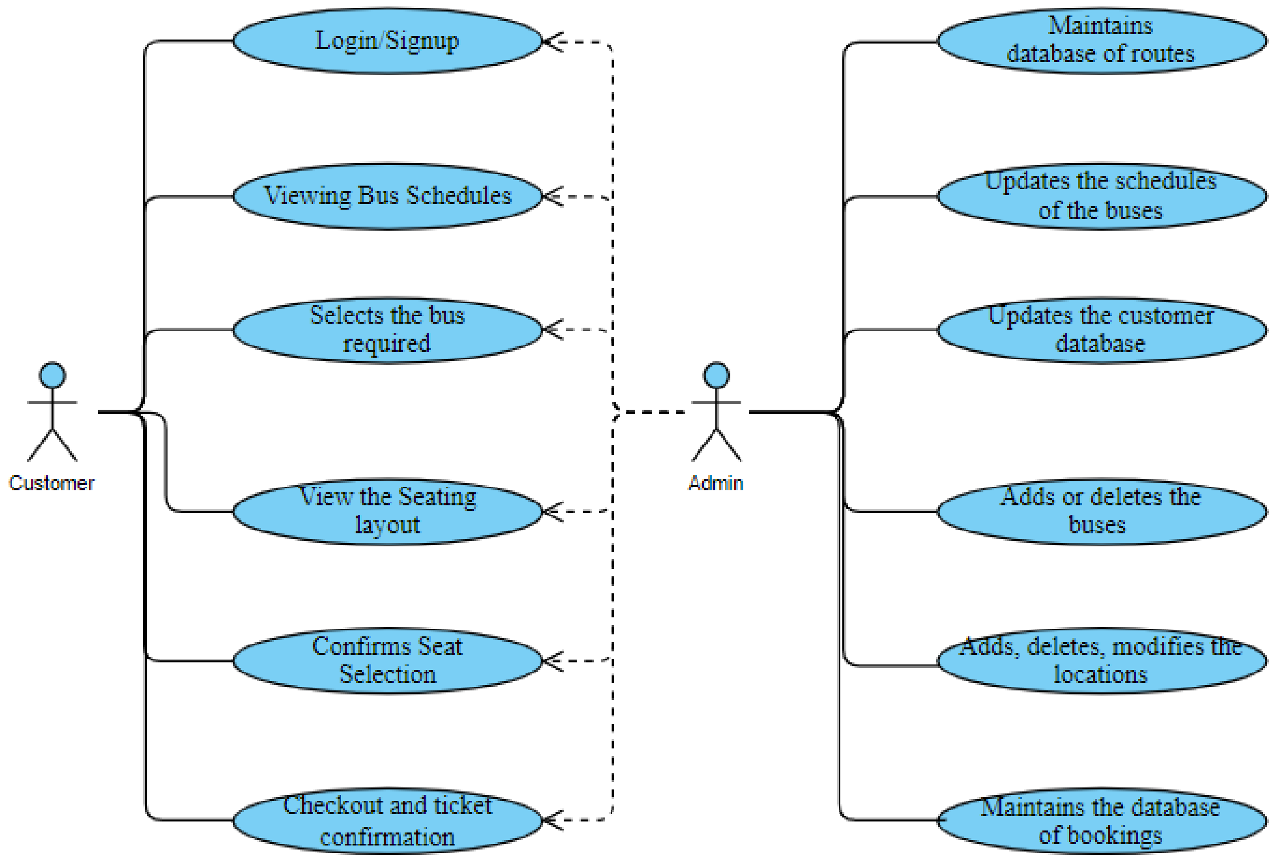
This is a local transportation ticketing system used to book tickets for local travel through buses. The stakeholders of this app are customers using the app, be it an individual or a group of people or corporate companies to book many numbers of tickets at a time or people from a society or a community, the travel agencies using this app for their bookings, investors and the admin that manages the app.

2.2 Use cases

The use cases of this app are described below:

- Customer login
- User viewing Bus Schedules
- User Selects the bus required
- User can view the Seating layout
- User Confirms Seat Selection
- User Proceeds to checkout
- Admin Sends eTicket to the passenger
- Admin updates the schedules of the buses
- Admin updates the customer database
- Admin adds or deletes the buses
- Admin maintains and modifies the locations
- Admin maintains the database of bookings
- Admin maintains database of routes

2.2.1 Graphic use case model


Login/Signup
Viewing Bus Schedules
Selects the bus required
View the Seating layout
Confirms Seat Selection
Checkout and ticket confirmation
Customer
Admin
Maintains database of routes
Updates the schedules of the buses
Updates the customer database
Adds or deletes the buses
Adds, deletes, modifies the locations
Maintains the database of bookings

2.2.2 Description for each use case

E-ticketing App	Login/Sign Up
Actors	Customer, Admin
Description	Customer has to create an account or register as new user if they never used the app before. Otherwise, customer has to login. This login information is stored at the admin and the credentials are verified from admin end
Data	Email ID or mobile number and Password
Stimulus	When the app just begins to run or is opened, immediately login page pops
Response	Enter the login credentials or register as new user
Comments	This is the login page function.
E-ticketing App	**Viewing Bus Schedules**
Actors	Customer, Admin
Description	Customer can view the bus schedules to select one they desire and book the ticket
Data	No data is collected here. But schedules of buses data is displayed from admin end
Stimulus	When the customer selects View Schedules from the menu
Response	Data of bus schedules is displayed
Comments	This is Viewing Bus Schedules
E-ticketing App	**Selects the bus required**
Actors	Customer, admin
Description	The customer selects a bus with a particular route and this selection is updated in the bookings database on admin end
Data	Selection of bus is updated on admin side
Stimulus	A touch stimulus on bus schedule display
Response	The bus and that route gets selected
Comments	This is a simple operation of Selecting the bus
E-ticketing App	**View the Seating layout**
Actors	Customer, admin
Description	A seating Layout is shown to the customer from which they select their desired seat for travel
Data	The seating layout is displayed from the admin end
Stimulus	On selecting a bus and a route
Response	The seating layout is displayed
Comments	This is the operation of Viewing the seating layout
E-ticketing App	**Confirms Seat Selection**
Actors	Customer, admin
Description	Customer selects and confirms a seat for their travel and it gets updated on the admin side database.
Data	On selecting the seat, the chosen seat gets updated in the admin side
Stimulus	On selecting a bus and route
Response	The seat is confirmed and is updated as one of the booking details on the admin side
Comments	This is confirmation of seat selection.

E-ticketing App	Checkout and Ticket Confirmation
Actors	Customer, Admin
Description	Here, the customer pays for the ticket they've selected and receives a ticket confirmation in return
Data	The customer's payment details are taken, payment is processed and ticket confirmation is sent
Stimulus	On hitting proceed to checkout
Response	Customer is directed to the payment page where the payment is done who then receives the ticket confirmation
Comments	This is the Checkout function
E-ticketing App	**Updates the schedules of the buses**
Actors	Admin
Description	The admin updates the schedules of the buses on a daily basis.
Data	External Data of bus schedules are taken and updated on the system
Stimulus	The change of date
Response	Schedules are refreshed and updated if necessary
Comments	This is updation of schedules of buses
E-ticketing App	**Updates the customer database**
Actors	Admin
Description	The customer database is updated each time there is a new registration with the details of the new customers
Data	Personal Details of the customer like name, age, gender, contact details like email ID or mobile number, etc.
Stimulus	When there is a new user registration or a new Sign Up
Response	The personal details are collected and updated to the customer database
Comments	This is Customer database updation
E-ticketing App	**Adds or deletes the buses**
Actors	Admin
Description	The admin is responsible for addition or deletion of buses for the day if and when there are any changes, like a certain bus failure or a certain contract closure with a travel agency, etc.
Data	Add or Delete the information of buses
Stimulus	There is no electronic stimulus to it, it's a non-functional stimulus
Response	Modifications are made to the bus information database
Comments	This is the function of addition or deletion of buses.
E-ticketing App	**Maintains and modifies the locations**
Actors	Admin
Description	The admin is responsible for maintaining the database of the set of locations this app is compatible to book tickets to. If there is any change, like, A certain location is closed for visiting, that information must be updated to the respective databases
Data	Information about working locations
Stimulus	No particular stimulus
Response	Changes are made according to requirement
Comments	This is the function for Maintaining and Modifying the locations
E-ticketing App	**Maintains the database of bookings**

Actors	Admin
Description	Admin maintains a record of all the bookings made from the beginning of the app.
Data	Information like Bus type, Route info, Customer name, Seat Selection, Date of journey, Source and Destination, etc are updated in that database
Stimulus	Whenever a customer makes a booking
Response	The bookings database is updated with relevant information
Comments	This is Maintenance of bookings database
E-ticketing App	Maintains database of routes
Actors	Admin
Description	Admin maintains the database of the different routes the buses travel
Data	Different routes travelled by the bus
Stimulus	Change of date
Response	Rutes are refreshed, rechecked and modified if required
Comments	This is the maintenance of Routes database
E-ticketing App	Sends eTicket to the passenger
Actors	Admin, Customer
Description	When a ticket is purchased by customer, the customer receives a confirmation from the admin side
Data	This eTicket contains information such as Name, Age, Starting point of journey, destination, Seat number, etc
Stimulus	As soon as the customer completes a payment transaction
Response	The eTicket is sent to them via mobile number and/or email id
Comments	This is the function of eTicket

2.3 Rationale for our use case model

We have chosen this use case model or these set of use cases as these are the basic features of any online ticket booking system. We intended to make a basic app, not too advanced, that helps even a common man, who might not know the first thing about operating an electronic device. For a person like that, the features should be as simple as possible. This system is designed to be working as a web application. But this can also be implemented in any bus stations or bus stops just like an ATM machine, where people can simply print the ticket before getting into their respective buses, saving time later. From the admin side, this app can reduce the tedious work of maintaining physical records of customers or routes or buses, etc. It can work as a basic solution with simple maintenance.

2.4 Non-functional Requirements

Non-functional requirements are those aspects of the system other than the specific functions it performs. These include system performance, costs, and such general system characteristics as reliability, security, and portability. The system shall provide attractive graphical interface for the user. The system shall allow developer access to installed environment. The application should be reliable and it should generate all updated information in correct order and in time. Application should be available and working properly at all times (24 hours). User information and Transport related information must be secure. The system must be responsive at all times. Any lag in response could lead to loss of interest from the customer end. Other requirements such as updating the schedule on time.

ARCHITECTURE

3.1 Architectural Style used

We have used the Layered Architecture Style for this Bus ticketing system. We have three layers interacting with each other. The three layers are user interface layer, application service layer and the database layer. This app can also be categorized under the transaction processing application architecture. Since it is a ticket booking app, the user enquires about the availability of tickets, the different routes and timings, etc and the system responds with corresponding information.

3.2 Architectural Model

The architectural model, as mentioned above, contains three layers as explained in the following part.

The first is the user interface layer. This is the program that runs on a remote user computer. It displays the provided services by the app to the user. The user selects options, such as ticket booking or schedule viewing etc. The user interface interacts with the application layer using the internet services.

The application service layer, is the most important layer, the system functions and business logic are handled in this layer. The system's logic is encapsulated in this layer. The application service interfaces are provided for the user interface layer and the system modules between the function calls. The application service layer also updates data in the database, according to the service request of the top layer.

The database layer, is used to hold data, including user registration information, ticket ordering information, ticket information and all of the other information.

3.3 Technology: Software and Hardware used

Hardware used in building this app is a PC with basic specifications like 8GB RAM, 512GB ROM and stable internet connection. The software used is as follows:

- Python
- Android Studio
- Django
- HTML
- CSS
- JavaScript
- jQuery
- Ajax
- Bootstrap v5
- Material Design Bootstrap Template
- Font-Awesome

Android studio, Django, and Python are used in the development of this project app, while PHP is used for the administration panel. In this case, the java programming language is utilised for the validation of the fields, and the XML programming language is used for the transmission of data. This project will continue to inquire for the latest plugin update in order to keep the internet connection active. Additionally, you will need to update the version of your SDK, and you will also need to update any instant run plugins that you are using.

3.4 Rationale for Architectural Style and Model

We’ve chosen the layered architecture with the transaction processing application architecture. It is a straight forward selection. We are designing an app that deals with information transactions like ticket availability, route enquiries, etc and this is happening in a layered fashion. From the user interface layer the query goes to the application layer and in turn the datatbase layer, fetches the information and comes back to the user interface layer. Here each layer is interacting with the one before and after it.

DESIGN

4.1 User Interface Design

Users of the system will find the new interface straightforward and easy to use. There will be a graphical user interface with menus at its core. It will be up to the users to make the appropriate choices or input the necessary information. The following is the user interface design:

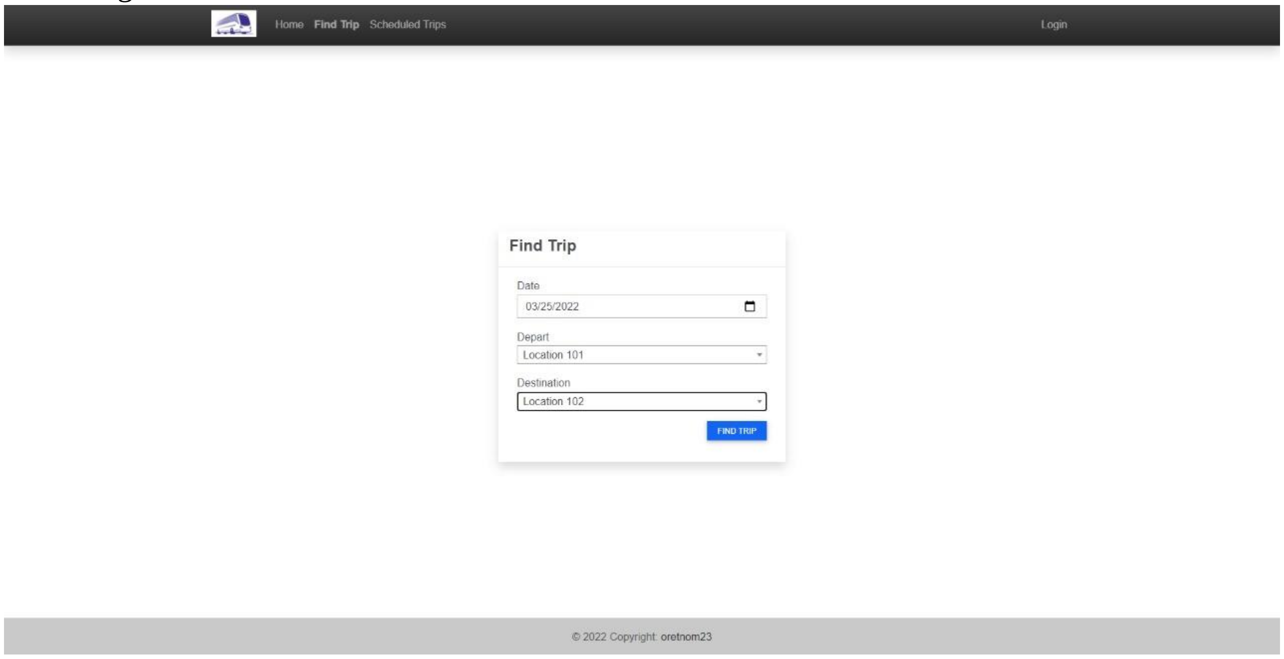


Figure 4.1: **Find Trip page**

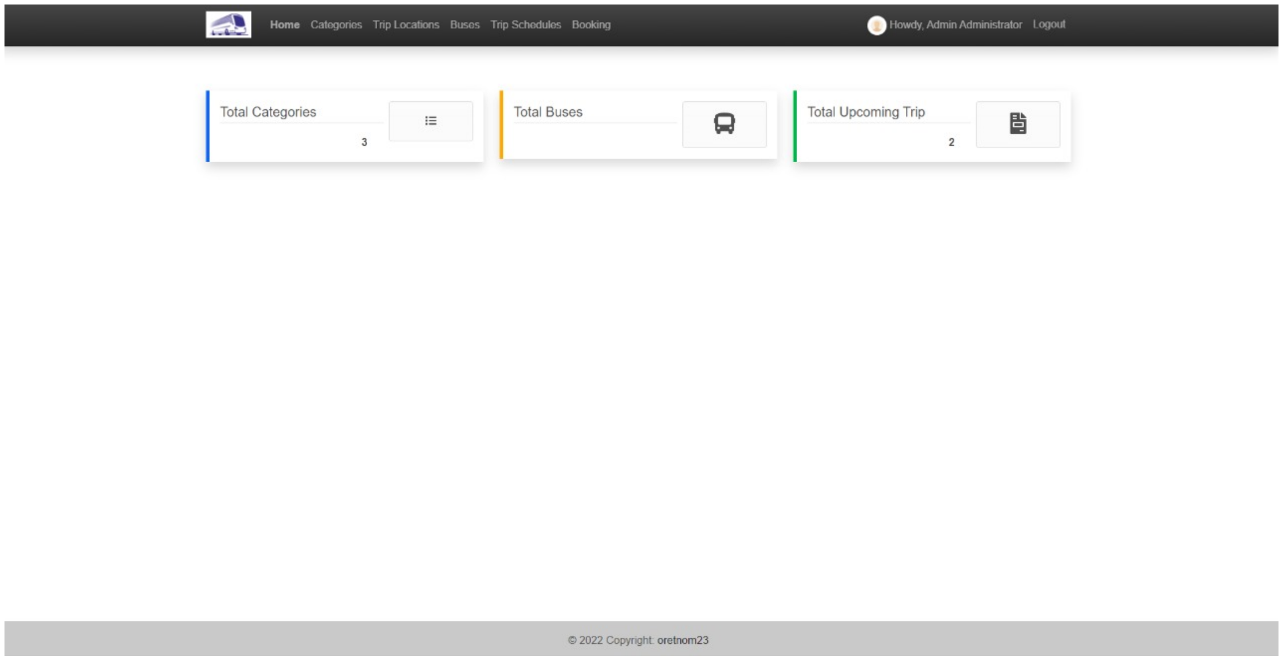
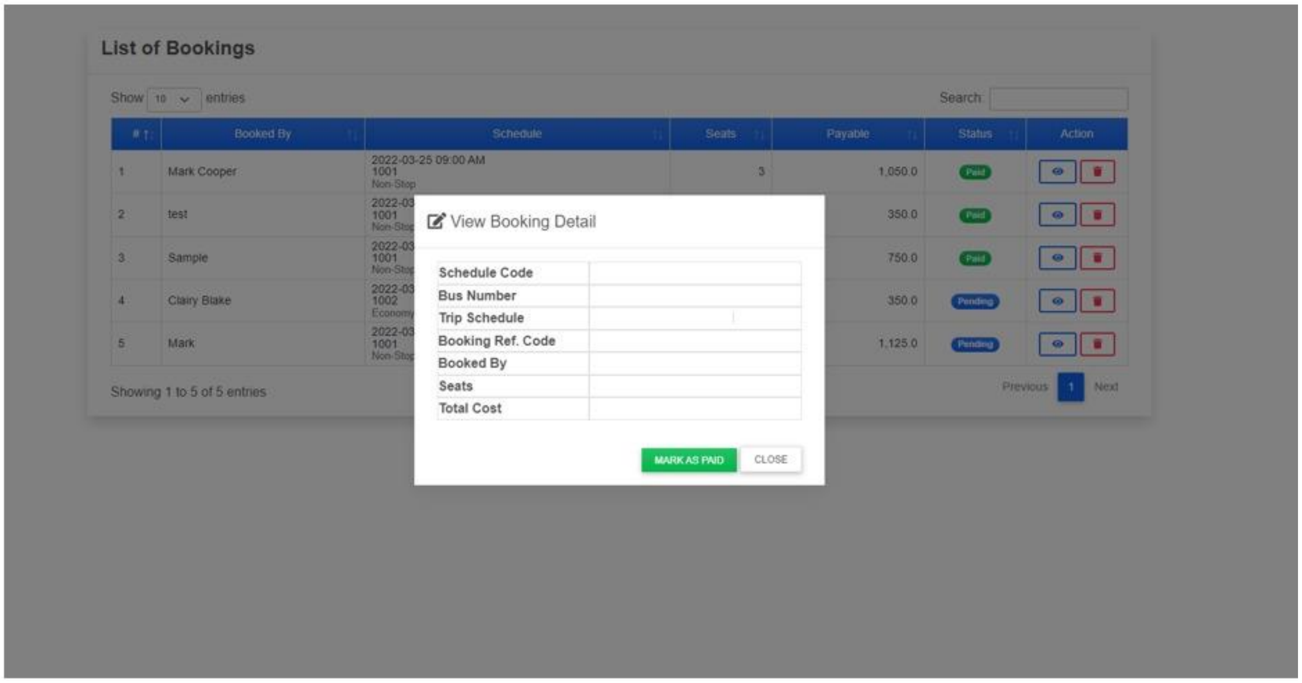


Figure 4.2: Menu



List of Bookings

Show 10 entries Search:

#	Booked By	Schedule	Seats	Payable	Status	Action
1	Mark Cooper	2022-03-25 09:00 AM 1001 Non-Stop	3	1,050.0	Paid	
2	test	2022-03 1001 Non-Stop		350.0	Paid	
3	Sample	2022-03 1001 Non-Stop		750.0	Paid	
4	Clairy Blake	2022-03 1002 Economy		350.0	Pending	
5	Mark	2022-03 1001 Non-Stop		1,125.0	Pending	

Showing 1 to 5 of 5 entries

Previous 1 Next

View Booking Detail

Schedule Code	
Bus Number	
Trip Schedule	
Booking Ref. Code	
Booked By	
Seats	
Total Cost	

MARK AS PAID CLOSE

Figure 4.3: Booking Detail Viewing Page

4.2 Components design

One of the dynamic models of the system, Sequence Diagram, is shown below:

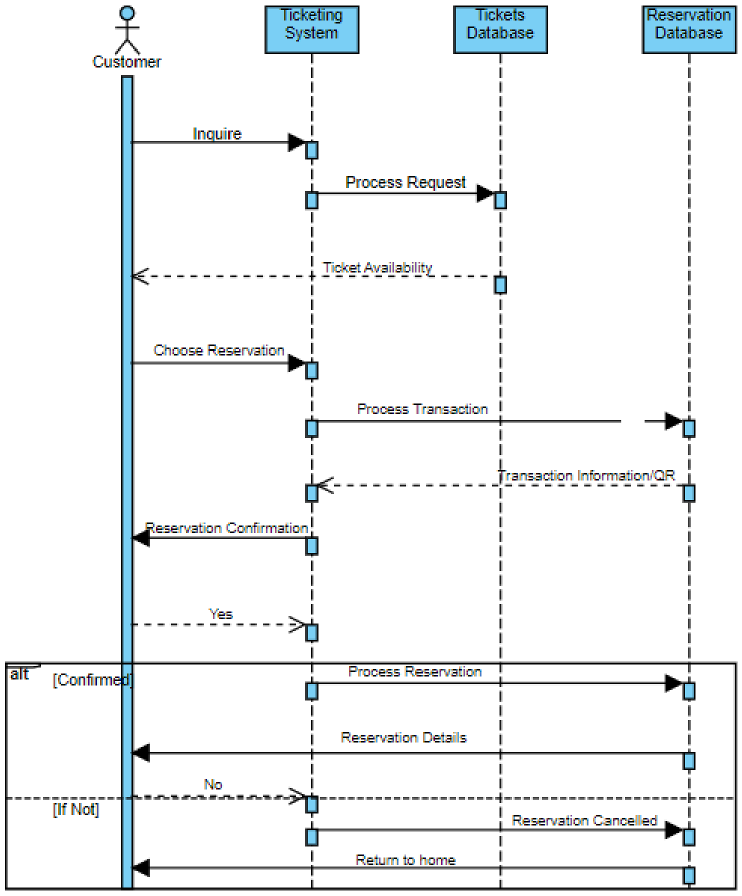


Figure 4.4: Sequence Diagram

The other dynamic model of the system, State Diagram, is shown below:

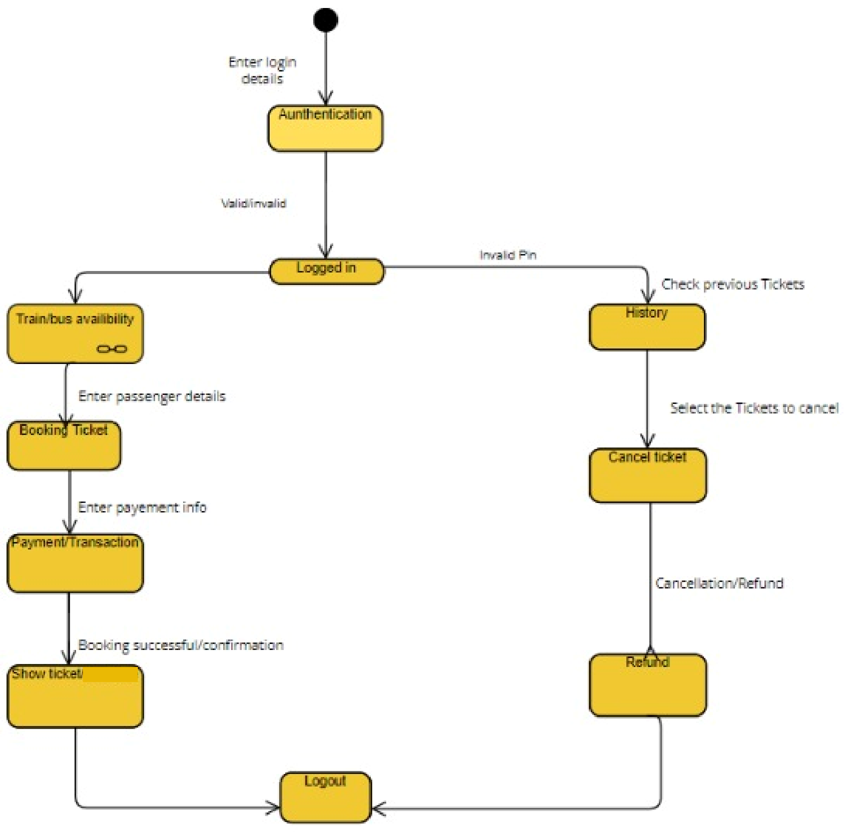


Figure 4.5: State Diagram

The static model of this system, Class diagram, is as shown below:

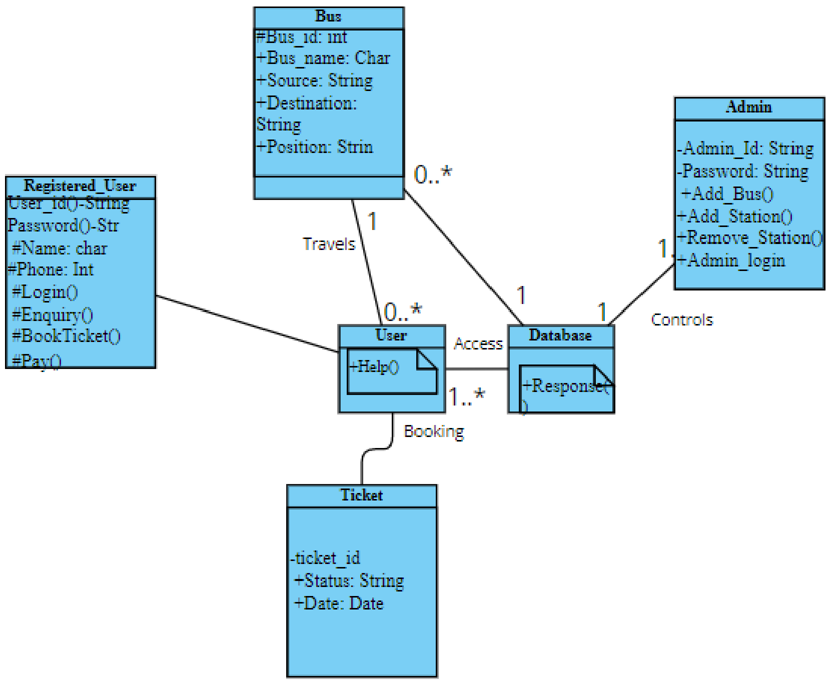
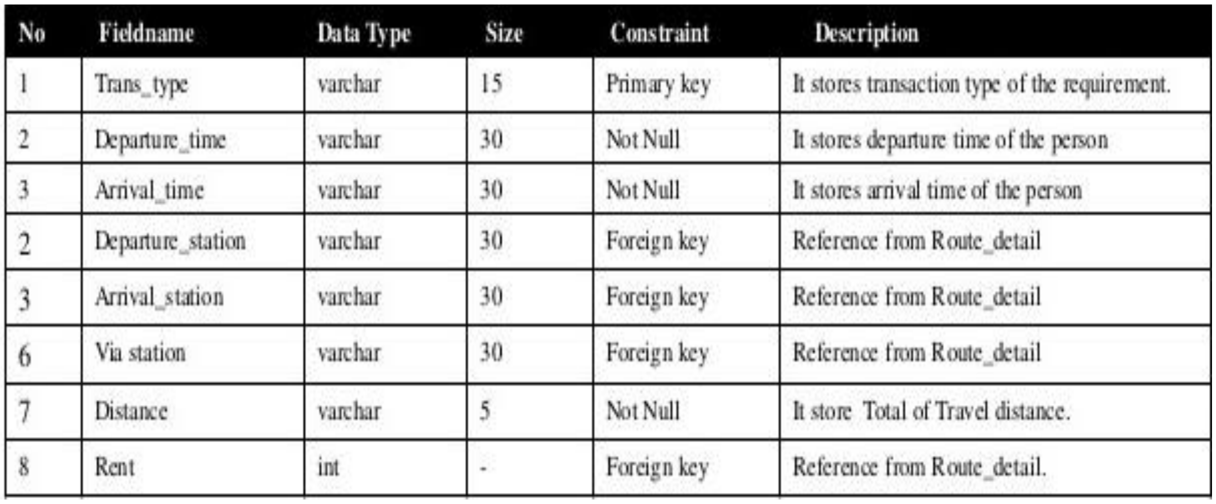


Figure 4.5: Class Diagram

4.3 Database Design

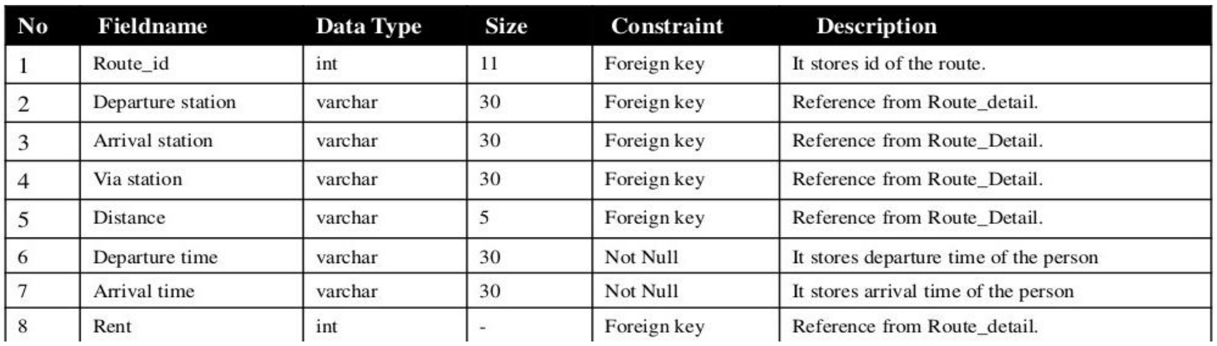
The SQL-based customer database includes the following tables:

- customers
- ticket
- bus
- destination
- route
- timetable



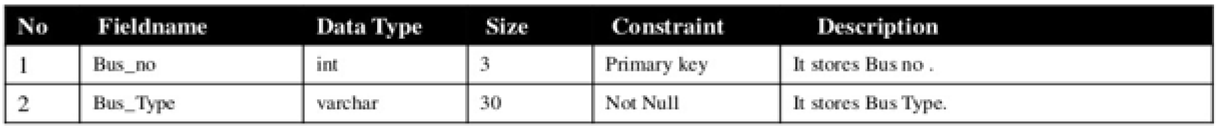
No	Fieldname	Data Type	Size	Constraint	Description
1	Trans_type	varchar	15	Primary key	It stores transaction type of the requirement.
2	Departure_time	varchar	30	Not Null	It stores departure time of the person
3	Arrival_time	varchar	30	Not Null	It stores arrival time of the person
2	Departure_station	varchar	30	Foreign key	Reference from Route_detail
3	Arrival_station	varchar	30	Foreign key	Reference from Route_detail
6	Via station	varchar	30	Foreign key	Reference from Route_detail
7	Distance	varchar	5	Not Null	It store Total of Travel distance.
8	Rent	int	-	Foreign key	Reference from Route_detail.

Table 4.1: Store the information of ticket



No	Fieldname	Data Type	Size	Constraint	Description
1	Route_id	int	11	Foreign key	It stores id of the route.
2	Departure station	varchar	30	Foreign key	Reference from Route_detail.
3	Arrival station	varchar	30	Foreign key	Reference from Route_Detail.
4	Via station	varchar	30	Foreign key	Reference from Route_Detail.
5	Distance	varchar	5	Foreign key	Reference from Route_Detail.
6	Departure time	varchar	30	Not Null	It stores departure time of the person
7	Arrival time	varchar	30	Not Null	It stores arrival time of the person
8	Rent	int	-	Foreign key	Reference from Route_detail.

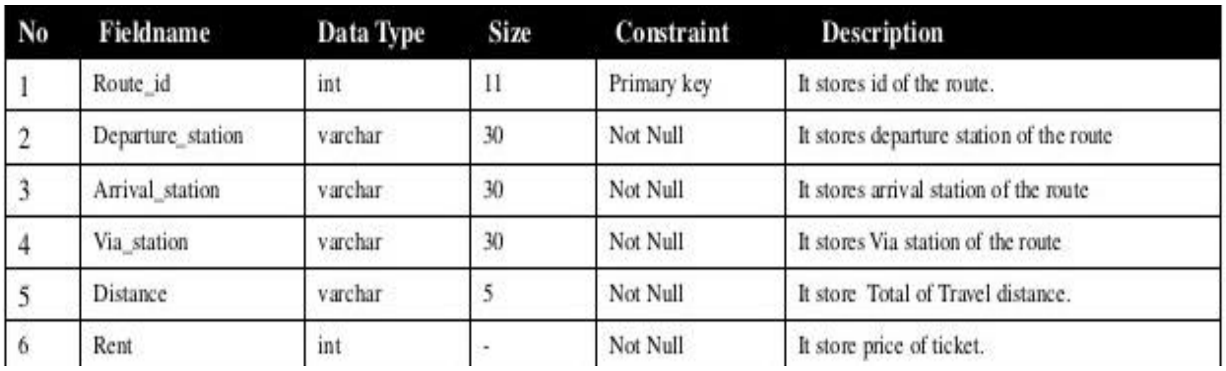
Table 4.2: Store the route information of the ticket



No	Fieldname	Data Type	Size	Constraint	Description
1	Bus_no	int	3	Primary key	It stores Bus no .
2	Bus_Type	varchar	30	Not Null	It stores Bus Type.

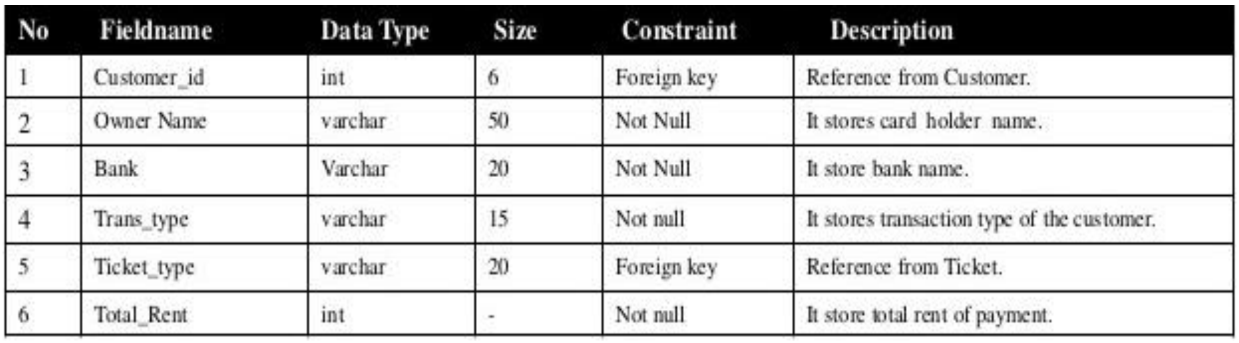
Table 4.3: Store the information of the bus booking detail

This table show the activities both a customer and admin can perform once logged in



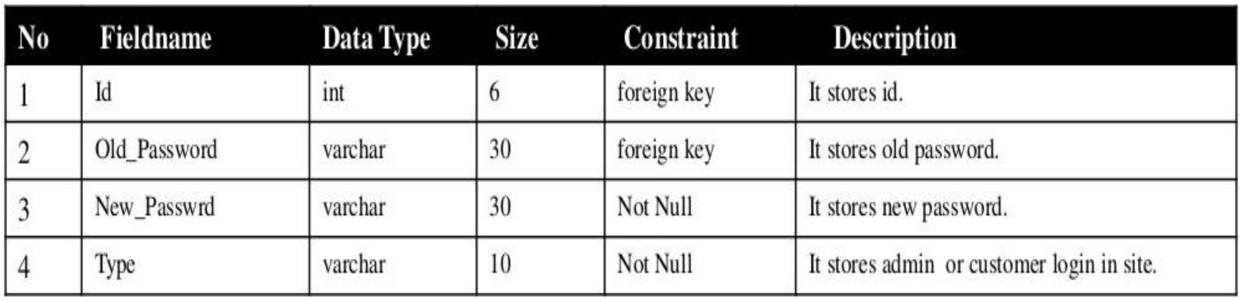
No	Fieldname	Data Type	Size	Constraint	Description
1	Route_id	int	11	Primary key	It stores id of the route.
2	Departure_station	varchar	30	Not Null	It stores departure station of the route
3	Arrival_station	varchar	30	Not Null	It stores arrival station of the route
4	Via_station	varchar	30	Not Null	It stores Via station of the route
5	Distance	varchar	5	Not Null	It store Total of Travel distance.
6	Rent	int	-	Not Null	It store price of ticket.

Table 4.4: Store the route information of the ticket



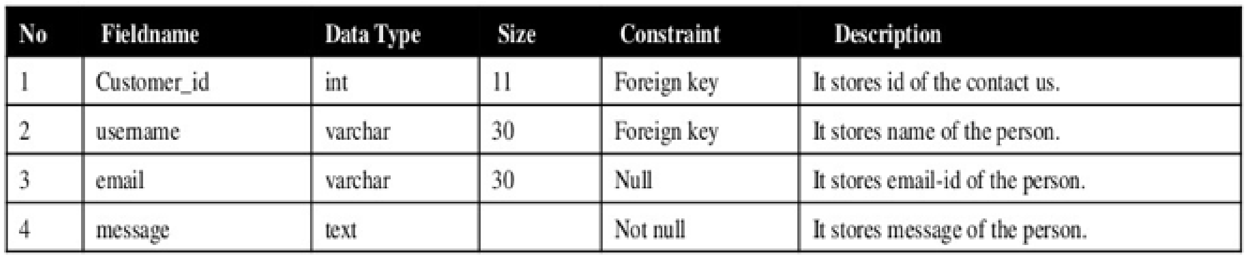
No	Fieldname	Data Type	Size	Constraint	Description
1	Customer_id	int	6	Foreign key	Reference from Customer.
2	Owner Name	varchar	50	Not Null	It stores card holder name.
3	Bank	Varchar	20	Not Null	It store bank name.
4	Trans_type	varchar	15	Not null	It stores transaction type of the customer.
5	Ticket_type	varchar	20	Foreign key	Reference from Ticket.
6	Total_Rent	int	-	Not null	It store total rent of payment.

Table 4.5: Store the information of the payment



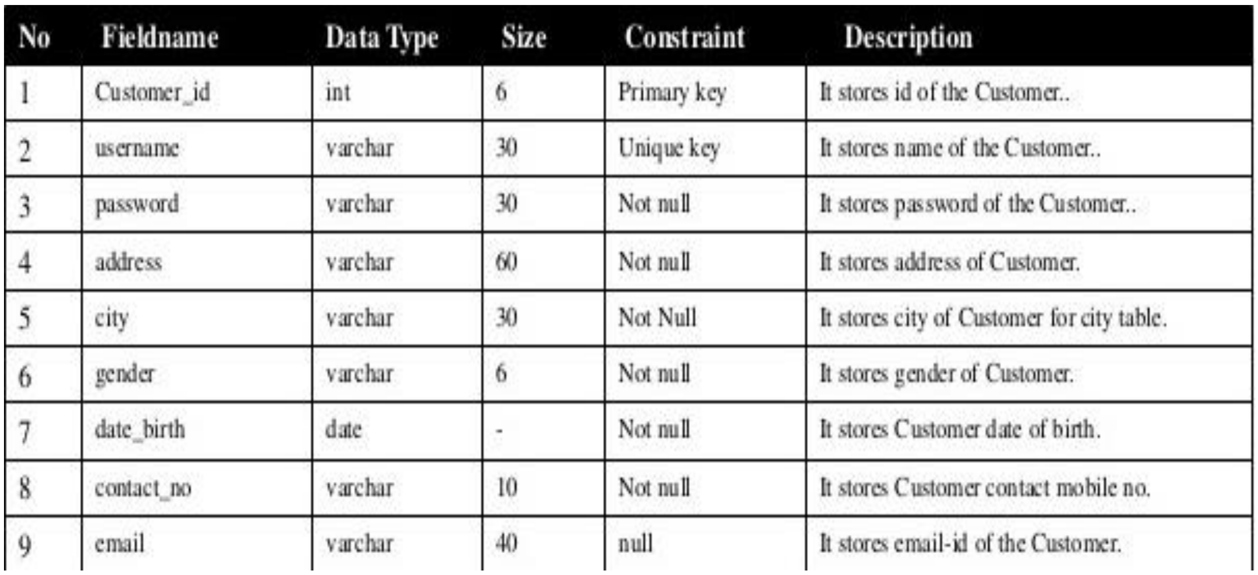
No	Fieldname	Data Type	Size	Constraint	Description
1	Id	int	6	foreign key	It stores id.
2	Old_Password	varchar	30	foreign key	It stores old password.
3	New_Passwrd	varchar	30	Not Null	It stores new password.
4	Type	varchar	10	Not Null	It stores admin or customer login in site.

Table 4.6: Store the information of the change password



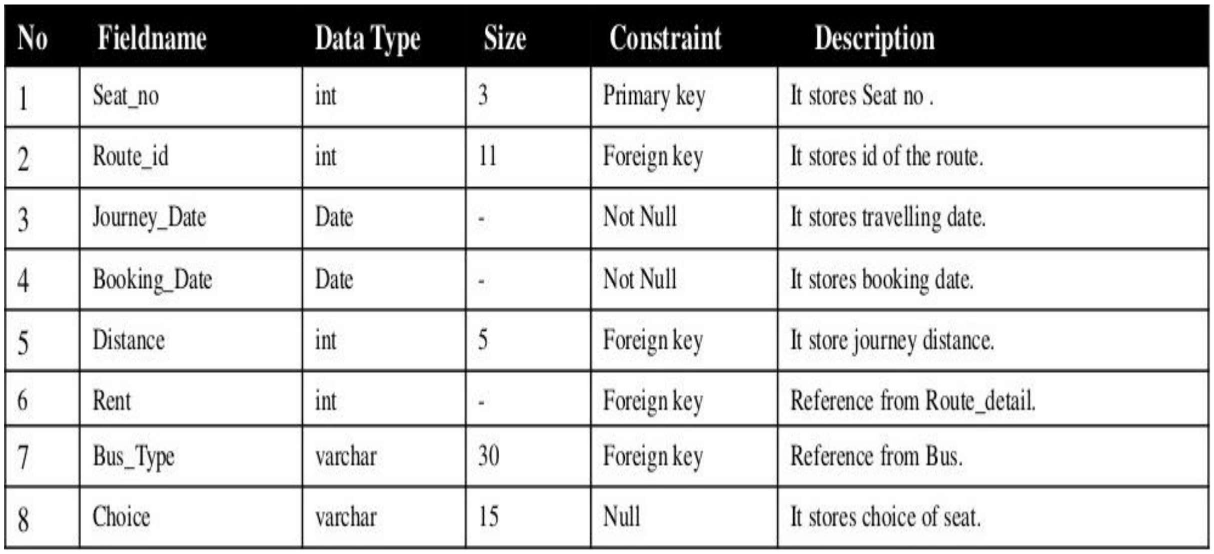
No	Fieldname	Data Type	Size	Constraint	Description
1	Customer_id	int	11	Foreign key	It stores id of the contact us.
2	username	varchar	30	Foreign key	It stores name of the person.
3	email	varchar	30	Null	It stores email-id of the person.
4	message	text		Not null	It stores message of the person.

Table 4.7: Stores all the contact person detail



No	Fieldname	Data Type	Size	Constraint	Description
1	Customer_id	int	6	Primary key	It stores id of the Customer..
2	username	varchar	30	Unique key	It stores name of the Customer..
3	password	varchar	30	Not null	It stores password of the Customer..
4	address	varchar	60	Not null	It stores address of Customer.
5	city	varchar	30	Not Null	It stores city of Customer for city table.
6	gender	varchar	6	Not null	It stores gender of Customer.
7	date_birth	date	-	Not null	It stores Customer date of birth.
8	contact_no	varchar	10	Not null	It stores Customer contact mobile no.
9	email	varchar	40	null	It stores email-id of the Customer.

Table 4.8: Store the customer information



No	Fieldname	Data Type	Size	Constraint	Description
1	Seat_no	int	3	Primary key	It stores Seat no .
2	Route_id	int	11	Foreign key	It stores id of the route.
3	Journey_Date	Date	-	Not Null	It stores travelling date.
4	Booking_Date	Date	-	Not Null	It stores booking date.
5	Distance	int	5	Foreign key	It store journey distance.
6	Rent	int	-	Foreign key	Reference from Route_detail.
7	Bus_Type	varchar	30	Foreign key	Reference from Bus.
8	Choice	varchar	15	Null	It stores choice of seat.

Table 4.9: Store the information of the book detail

4.4 Traceability from requirements to detailed design models

According to the requirements, our main goal was to create a bus ticket booking app with a sustainable database to make the work of the user as well as the administrator easier. In the same way, we tried to bring about as many features as possible into existence with a well-defined database design.

TEST MANAGEMENT

5.1 A Complete list of System Test cases

Home page

The system's homepage is the page that loads when a user enters the app URL in a browser. The homepage provides summary data, such as the company's headline, welcome words, core values, mission statement, and a few representative photos. Additional pages may be accessed using the buttons labelled "log in," "register," "admin login," "services," "about us," and "contacts."

About us page

This page provides specifics regarding the services offered by the Online Bus Ticketing System, as well as the terms and conditions associated with working with them. For the convenience of the user, the page also has buttons to login and create an account.

Contacts us and addresses

On the page labelled "contacts," users may find all of the pertinent contact information on the location of the bus business, including the telephone numbers and postal addresses of its many locations. Again, the link to register and log in is provided on this page so that the user may be sent to the correct page.

Customer registration

On this page, the user will be asked to register for an account with the Online Bus Ticketing System by filling out a form that is made available. The following fields are available for user input on this form:

- **First name -** It is necessary for the user to input the first name of his or her choosing.
- **Last name -** When asked about their last name, the user supplies a different name from the one they used for their first name.
- **Username -** The account holder reveals the name that will be used for future sign-in.
- **Email Address -** Users must have a working email address for communication purposes.
- **Contact -** Users may leave their contact information here in case they need to get in touch with the bus service.
- **Password -** For authentication, the user must reveal this secret string of values, which may include letters, numbers, and other symbols.
- **Confirm password -** Users are often asked to retype their passwords as a security measure to ensure they are familiar with the password they have just entered and to check for any errors.
- **Register -** This is a submit input for users to enter their registration information and send it off to the server.

To facilitate speedy exploration, the page has a login button.

User log in
Users who claim to have an account with the Online Bus Ticketing System may verify their status by entering their credentials on this page. It provides a short form with only two fields:

- **Username -** The username must be typed precisely as it appeared on the account creation form the user completed. The error rate increases with every deviation.
- **Password -**The user must enter the password that was previously entered and verified during the account creation process. If the user enters an invalid password, an error message will be shown.
- **Login -** When the user is ready to send their credentials to the database server, they may do so by clicking this button. An invalid login is the consequence of database changes.

Customer booking

Only those who have signed up for the Online Bus Ticketing System will be able to view this website, and only then will they be able to reserve a "spot" for the desired time frame. There's a booking form there that has to be filled out with specifics. Therefore, the fields that need input are:

- **Category -** The user may choose between different kinds of buses like Air-conditioned, Non-Air-Conditioned, etc.
- **Location -** The user is obligated to indicate their current abode in the location form.
- **Dates of booking -** Actual booking dates should be selected by the user.
- **Time-From-To -** The time frame in which the reservation will be held on the scheduled day. The time range, such as 10:00 a.m. to 11:00 a.m., is given.
- **Nature of bus company -** It is up to the user to choose whether or not the Online Bus Ticketing System is stationary or mobile.
- **Book -** In order to confirm a reservation, the user must fill out certain information and then click the book button, which sends the information to the database server.

Administrator log in

Only the system administrator will be able to access this page. A form containing the following fields will be filled out by him or her:

- **Username -** The administrator was obligated to use the exact same username that appeared in the administrators’ database. An erroneous result is guaranteed with any deviation.
- **Password -** The administrator must type in the password exactly as it appears in the database record corresponding to administrators. If the password is changed, an error will be generated when submitted to the administrators.
- **Login -** There is a submit button here, which the administrator must use to save the login information to the server's database. An invalid login is the consequence of database changes.

Administrator update the booking

Upon logging in, the administrator is granted access to all of the booking and registration information for each individual client. Once a legitimate reservation has been made with a bus company, the administrator will activate it and update the user's progress view page. The administrator also removes any information that is deemed to be incorrect.

Administrator update the payment

The administrator checks the transactions by accepting or declining the payments. Any payments that are invalid are removed.

Physical process design

Everything that happens behind the scenes whenever a user interacts with the system is detailed here.

User registration, user login, user booking, user posting of payments, user checking of progress and payments, user logging out, administrator login, administrator update of progress and payments, administrator print available bookings, administrator posting and changing of bus company prices, administrator database manipulation, administrator logging out.

Storage needs might range from:

- ✓ User details of registration,
- ✓ User booking details and
- ✓ Admins authentication details.

5.2 Traceability of test cases to use cases

The use cases mentioned before match up to 90% with the test cases. We have designed and tested all the use cases right from customer login to customer booking and administrator log in to updating the databases. The user interface is smooth enough for a person with no prior experience with any app. On the other hand, the administrator has all the minimum required databases for his/her work to be done without much struggle.

5.3 Techniques used for test case generation

The following resources were invaluable over the course of the system's coding.

Coding tools

Editing: During the coding process, we used the IDLE (Python 3.10 64-bit) software as the tool for editing the code using the various languages as discussed below.

- **SQL**: The full meaning of this acronym is "Structured Query Language." To make the Python code work with the database and to run the different queries, we utilized the SQL language.

Testing tools performance test

This evaluation determines whether or not the produced system effectively addresses the specified issue. This system will undergo the following performance reviews.

Unit testing: This necessitates checking out the parts that make up the whole system. Since each component was tested separately, this method aided in the detection of flaws.

Stress testing: This kind of test is designed to verify how well a system handles out-of-the-ordinary scenarios. In our tests, the system was unable to proceed with execution when presented with erroneous input data like blank form fields.

Actual system testing: The whole system undergoes this procedure to ensure that all of its components are functioning as intended when development is complete. In order to determine whether the aforementioned goals have been met, this system will be put through a test.

Functional testing: Providing input data and evaluating the program's output is a crucial part of testing its functionality. This will be done to ensure the proper operation of the program's features and to detect and fix any unforeseen issues.

System test plan

Inputs of different formats, including integers (INT), variable characters (VARCHAR), dates and times, and others, were utilized to put the system through its paces.

User acceptance testing

Any incorrect information entered during testing would cause unexpected outcomes, which might be flagged by the system's validation features.

Potential users were also given the system for testing and input, whereupon they agreed that it was a viable alternative to the time-consuming and laborious manual processes now used to create the Online Bus Ticketing System. After the system had been developed to completion and handed to the users for their feedback, an acceptance test was conducted to ensure that the system really matched their needs. This system's user acceptability testing was performed after the majority of its features were completed so that interested parties could provide feedback.

Proposed change-over techniques

There are typically four ways the system may be introduced to a company. One may choose from a direct switch, a phased or gradual switch, a pilot program, or a parallel program. After much consideration, I decided on a staged approach to launching the system.

Phased changeover

A phased operation is one that is carried out in phases. Module-based system deployment is a common method for introducing a new system. In addition, it combines elements of both the direct transition and the parallel strategy. Due to the novelty of the system and the lack of clarity on its anticipated user base, we plan to execute it in this manner, bringing in new sections of the system one at a time until everything is up and running as planned. The potential for mistakes or failures in this system may have also encouraged me to employ the same, but such dangers will not affect the whole system, just the module or modules that have been implemented so far. Similarly, the direct method, which requires implementing the whole system at once, may have a higher initial outlay but a lower total cost of ownership for its usage.

A phased operation is one that is carried out in phases. Phased operation refers to the method of introducing a new system by breaking it up into smaller chunks. This method combines the advantages of both direct handoff and parallel running, much like a pilot switch. In contrast to other methods, where just a subset of the system is made available to consumers, this one gives everyone the whole works.

`As well as being cheaper than complete parallel operation, phased operation also reduces the risk of mistakes and failures to only the implemented module. When there are several distinct stages to the system, however, phased operation might be more expensive than a pilot method.

After the development process is complete and the system is presented to the users for their feedback, if the users are satisfied with the system, it is said to have fulfilled the user requirement. The system's eventual end users and clients will be consulted throughout the user acceptability testing phase.

5.4 Test Results and Assessments

TC_ID	Test Case Description	Prerequisite	Steps	Data	Expected Result	Actual Result
TC_ID _01	Test Login functionality by entering registered mobile number and password	1)App should be installed 2)User should have registered mobile number	1)Install app 2)Launch App 3)Navigate to login screen 4)Enter mobile number 5)Enter password 6)Click on login	Mobile number:1234567890 password:123456	1)User should be successfully logged in and the user should be redirected to home page of app	User successfully logged in and the user redirected to home page of app
TC_ID _02	Test by entering special characters in mobile	App should be installed	1)Install app 2)Launch App	Mobile number= 2345678909	User should not be able to enter special characters in	User is not able to enter special characters in

	number field and try to login		3)Navigate to login screen 4)Enter special character in mobile number field 5)Enter password 6)Click on login		mobile number field	mobile number field
TC_ID _03	Test by entering characters in mobile number field and try to login	App should be installed	1)Install app 2)Launch App 3)Navigate to login screen 4)Enter character in mobile number field 5)Enter password 6)Click on login	Mobile number=ab cdef 123456	User should not be allowed to enter characters inmobile number field	User is not allowed to enter characters inmobile number field

TC_ID _04	Test by leaving Mobile number and password field blank and try to login	App should be installed	1)Install app 2)Launch App 3)Navigate to login screen 4)Leave mobile number field blank 5)Leave password field blank 6)Click on login		Enter mobile number and password error message should be displayed	Enter mobile number and password error message is displayed
TC_ID _05	Test the login functionality by login with mobile number which is not registered	App should be installed	1)Install app 2)Launch App 3)Navigate to login screen 4)Enter mobile number which is not registered 5)Enter password		User not found error message should be displayed	User not found error message is displayed

			6)Click on login			
TC_ID _06	Test login to app by entering invalid password	App should be installed	1)Install app 2)Launch App 3)Navigate to login screen 4)Enter mobile number 5)Enter wrong password 6)Click on login		Invalid Password error message should be displayed	Invalid Password error message is displayed

CONCLUSIONS

6.1 Outcomes of the project

The project Local Transportation Ticketing System is an app used to book local bus tickets to travel in and around the city or in between two cities. This app was mainly developed to reduce the tediousness of a manual ticket reservation process. It contains all the basic features to make the ticket reservation process as smooth and easy as possible. We could achieve maximum of the goals that we set in our use cases and requirements. The user interface is as shown below:

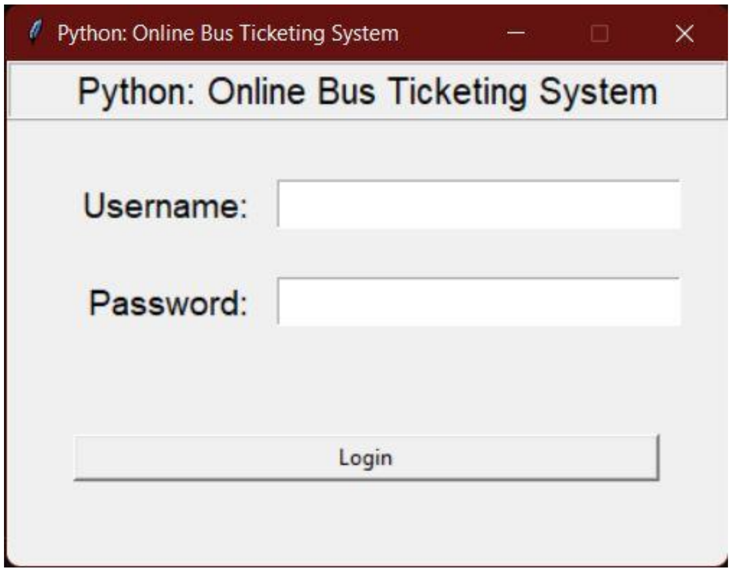

Python: Online Bus Ticketing System
Python: Online Bus Ticketing System
Username:
Password:
Login

Figure 6.1: Login Page

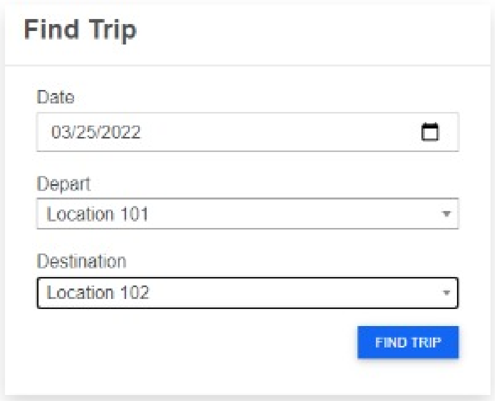
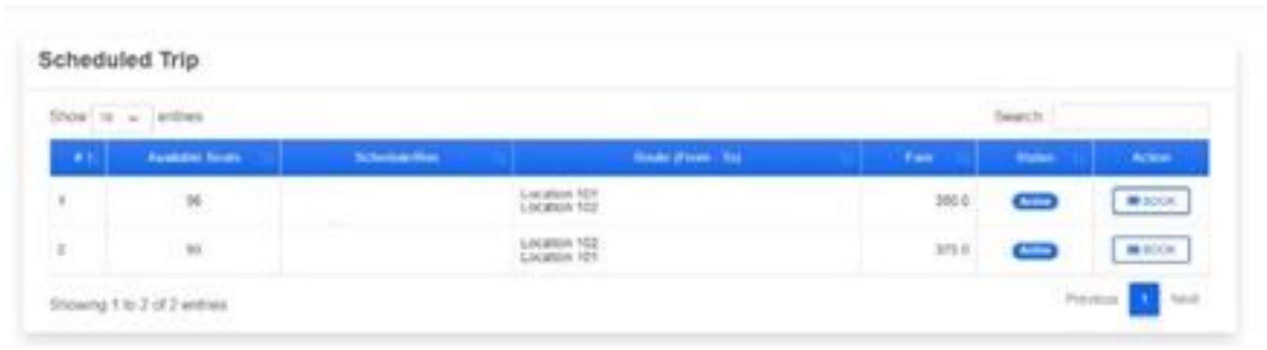


Figure 6.2: Find Trips Page



Scheduled Trip

Show 10 entries Search:

#	Available Seats	Schedule/Bus	Route (From - To)	Fare	Status	Action
1	96		Location 101 Location 102	350.0	Active	BOOK
2	90		Location 102 Location 101	375.0	Active	BOOK

Showing 1 to 2 of 2 entries

Previous 1 Next

Figure 6.3: Scheduled Trips page

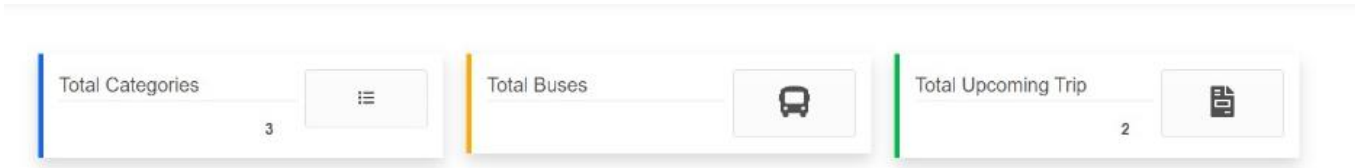
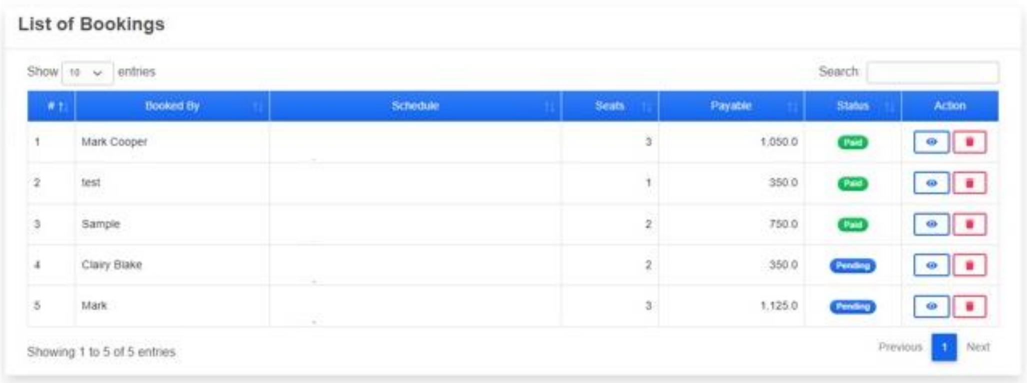


Figure 6.4: Menu



List of Bookings

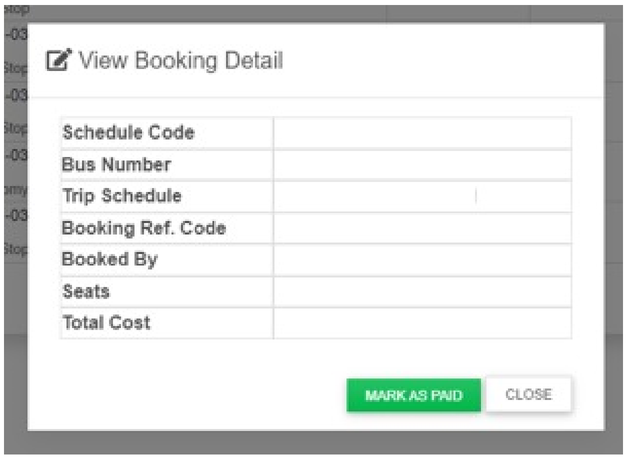
Show 10 entries Search:

#	Booked By	Schedule	Seats	Payable	Status	Action
1	Mark Cooper		3	1,050.0	Paid	
2	test		1	350.0	Paid	
3	Sample		2	750.0	Paid	
4	Clairy Blake		2	350.0	Pending	
5	Mark		3	1,125.0	Pending	

Showing 1 to 5 of 5 entries

Previous 1 Next

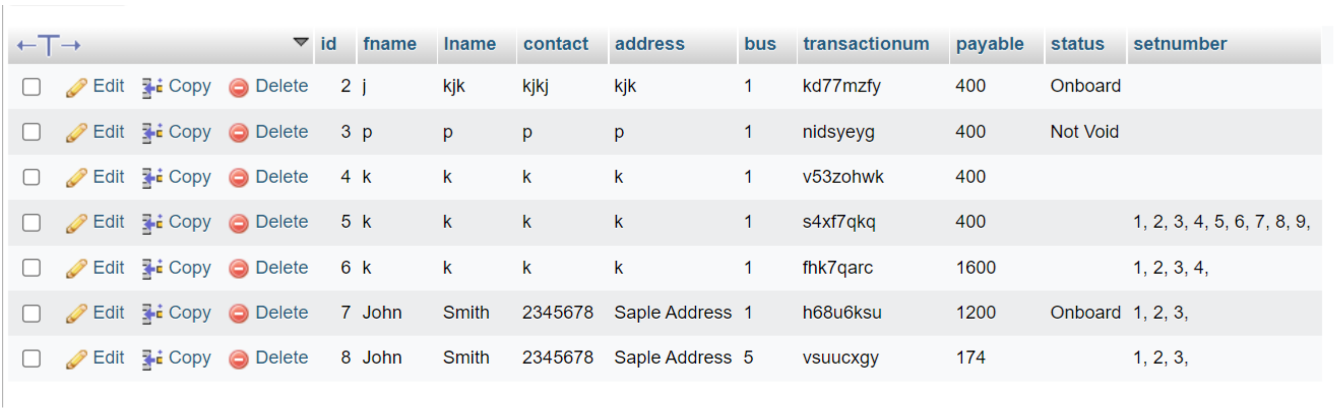
Figure 6.5: List of All Bookings



View Booking Detail

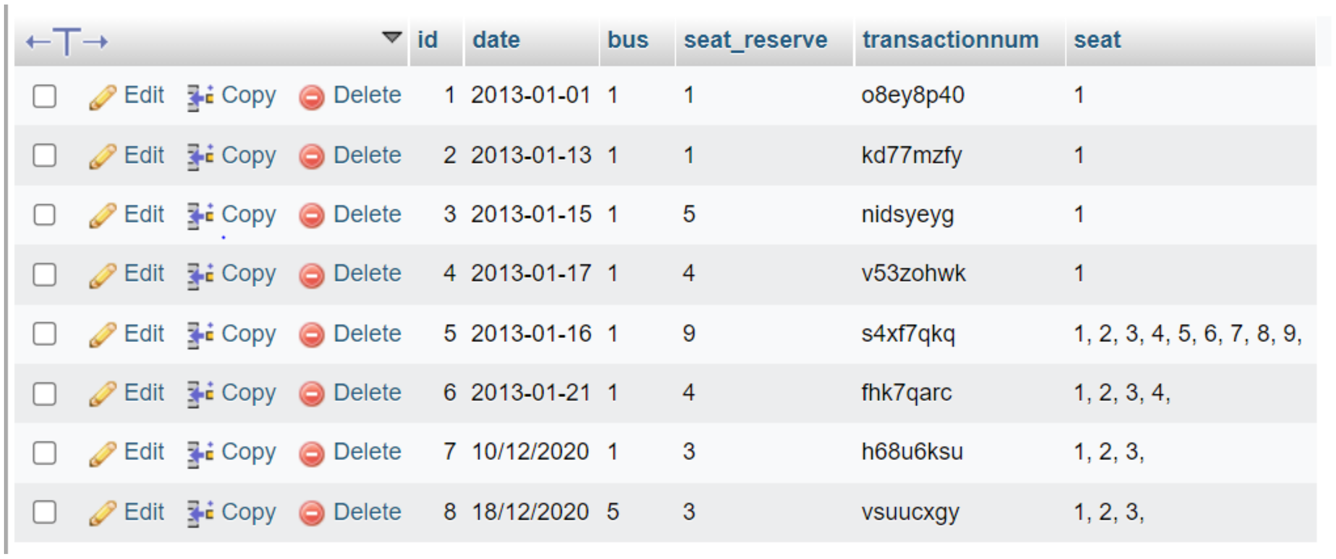
Schedule Code	
Bus Number	
Trip Schedule	
Booking Ref. Code	
Booked By	
Seats	
Total Cost	

Figure 6.6: Booking Details Viewing



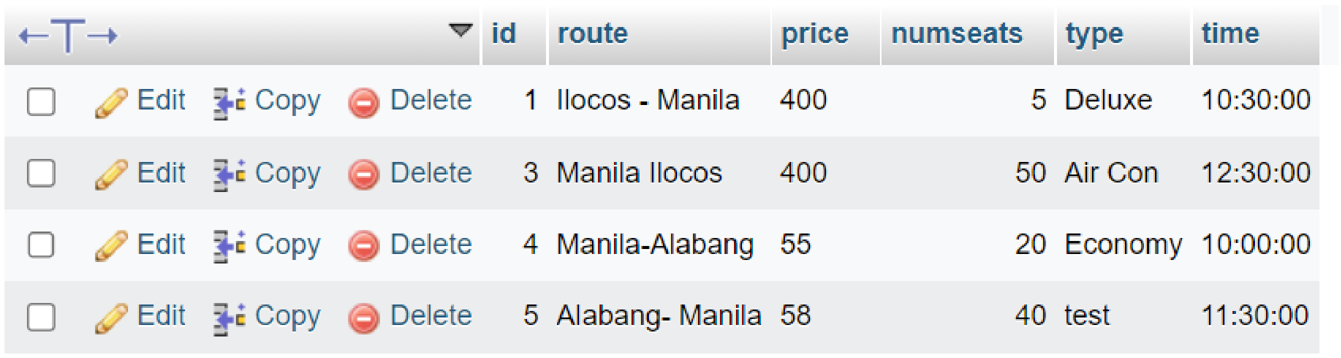
←T→ ▼	id	fname	lname	contact	address	bus	transactionum	payable	status	setnumber
☐ Edit Copy Delete	2	j	kjk	kjkj	kjk	1	kd77mzfy	400	Onboard	
☐ Edit Copy Delete	3	p	p	p	p	1	nidsyeyg	400	Not Void	
☐ Edit Copy Delete	4	k	k	k	k	1	v53zohwk	400		
☐ Edit Copy Delete	5	k	k	k	k	1	s4xf7qkq	400		1, 2, 3, 4, 5, 6, 7, 8, 9,
☐ Edit Copy Delete	6	k	k	k	k	1	fhk7qarc	1600		1, 2, 3, 4,
☐ Edit Copy Delete	7	John	Smith	2345678	Saple Address	1	h68u6ksu	1200	Onboard	1, 2, 3,
☐ Edit Copy Delete	8	John	Smith	2345678	Saple Address	5	vsuucxgy	174		1, 2, 3,

Figure 6.7: Customer database



←T→ ▼	id	date	bus	seat_reserve	transactionnum	seat
☐ Edit Copy Delete	1	2013-01-01	1	1	o8ey8p40	1
☐ Edit Copy Delete	2	2013-01-13	1	1	kd77mzfy	1
☐ Edit Copy Delete	3	2013-01-15	1	5	nidsyeyg	1
☐ Edit Copy Delete	4	2013-01-17	1	4	v53zohwk	1
☐ Edit Copy Delete	5	2013-01-16	1	9	s4xf7qkq	1, 2, 3, 4, 5, 6, 7, 8, 9,
☐ Edit Copy Delete	6	2013-01-21	1	4	fhk7qarc	1, 2, 3, 4,
☐ Edit Copy Delete	7	10/12/2020	1	3	h68u6ksu	1, 2, 3,
☐ Edit Copy Delete	8	18/12/2020	5	3	vsuucxgy	1, 2, 3,

Figure 6.8: Reservations Database



←T→ ▼	id	route	price	numseats	type	time
☐ Edit Copy Delete	1	Ilocos - Manila	400	5	Deluxe	10:30:00
☐ Edit Copy Delete	3	Manila Ilocos	400	50	Air Con	12:30:00
☐ Edit Copy Delete	4	Manila-Alabang	55	20	Economy	10:00:00
☐ Edit Copy Delete	5	Alabang- Manila	58	40	test	11:30:00

Figure 6.9: Routes database

6.2 Lessons Learned

In this whole project experience, we have learned about software engineering. Different concepts of a software, its architecture and types of architectural models, design, the different representations of a single software and most of all the way a software is built and steps in building a software. It was a very valuable experience that has given some insight into the software engineer’s world.

6.3 Future Development

This app can be further developed by adding other means of transport like trains or cabs. This app can be turned into a whole experience for travelers coming from different places around the world to feel like a local person. To roam is local means of transport, reserve accommodation in the best places of the city or town, add reservations to restaurants or movies or theatres, etc. It can be made into an overall Local Experience App.

REFERENCES

[1] https://www.interviewbit.com/blog/android-projects/ - For project idea
[2] https://www.python.org/doc/ - Coding
[3] https://docs.djangoproject.com/en/4.1/
[4] https://www.geeksforgeeks.org/
[5] https://developer.android.com/docs
[6] https://www.javascript.com/learn
[7] https://learn.jquery.com/