

Darshan University

A Project Report on

**Taxi management System**

Under the subject

**Software Engineering (2101CS503)**

B. Tech, Semester – V

Computer Science & Engineering Department

|  |  |
| --- | --- |
| Submitted By | |
| Student Name:Bhavesh Kadachha | Enrollment No.:23010101625 |
| Academic Year  (2023-2024) | |
| Internal Guide Mr. Rajkumar B Gondaliya Darshan University | Dean-DIET  Dr. Gopi Sanghani  Darshan University |

|  |  |
| --- | --- |
|  | **Computer Science & Engineering Department**  **Darshan University** |

**DECLARATION**

We hereby declare that the SRS, submitted along with the **Software Engineering** **(2101CS503)** for entitled **Taxi management System** submitted in partial fulfilment for the Semester-5 of **Bachelor Technology (B. Tech)** in **Computer Science and Engineering (CSE)** Departmentto Darshan University, Rajkot, is a record of the work carried out at **Darshan University, Rajkot** under the supervision of Prof. Rajkumar B Gondaliya and that no part of any of report has been directly copied from any students’ reports, without providing due reference.

Bhavesh kadachha

Student’s Signature

Date: \_\_\_\_\_\_\_\_\_\_

|  |  |
| --- | --- |
|  | **Computer Science & Engineering Department**  **Darshan University** |

**CERTIFICATE**

This is to certify that the SRS on **Taxi management System** has been satisfactorily prepared by Bhavesh Kadachha(23010101625) under my guidance in the fulfillment of the course **Software Engineering (2101CS503)** work during the academic year 2023-2024.

|  |  |  |
| --- | --- | --- |
| Internal Guide  Prof. Rajkumar B Gondaliya  Darshan University |  | Dean-DIET  Dr. Gopi Sanghani  Darshan University |

**Acknowledgement**

I wish to express my sincere gratitude to my project guide Prof. Rajkumar B Gondaliya and all the faculty members for helping me through my project by giving me the necessary suggestions and advices along with their valuable co- ordination in completing this work.

I also thank my parents, friends and all the members of the family for their precious support and encouragement which they had provided in completion of my work. In addition to that, I would also like to mention the Darshan University personals who gave me the permission to use and experience the valuable resources required for the project from the University premises.

Thus, in conclusion to the above said, I once again thank the faculties and members of **Darshan University** for their valuable support in completion of the project.

Thanking You

**Bhavesh kadachha**

**ABSTRACT**

The **Taxi Management System** (TMS) is designed to simplify and improve taxi services using a computerized system. This system makes it easy for passengers to book rides, view ride details, and track their taxi in real-time. Passengers can also manage their accounts and see their ride history.

For drivers, TMS provides tools to manage their work, accept or reject ride requests, and see their earnings. The system helps dispatch taxis efficiently, reducing waiting time and finding the best routes.

Administrators have a special panel to manage everything, including user accounts and system performance. They can also generate reports on ride details, driver performance, and earnings.

The main goal of TMS is to make the process of booking and managing taxi rides easier for everyone, reducing the need for manual work and improving overall efficiency.

**Table of Contents**

[List of Figures I](#_Toc146706805)

[List of Tables II](#_Toc146706806)

[1 Introduction 1](#_Toc146706807)

[1.1 Product perspective 1](#_Toc146706808)

[1.2 Product features 1](#_Toc146706809)

[1.2.1 There are three different users who will be using this product: 1](#_Toc146706810)

[1.2.2 The features that are required for the Admin are: 1](#_Toc146706811)

[1.2.3 The features that are required for the Manager are: 1](#_Toc146706812)

1.2. 4 The features that are required for the Driver are: 1

[1.2.5 The features that are required for the Customer are: 2](#_Toc146706810)

[1.3 Functional Requirement 2](#_Toc146706813)

[1.3.1 Admin 2](#_Toc146706814)

[1.3.2 Manager](#_Toc146706815) 2

[1.3.3 Driver](#_Toc146706815) 2

[1.3.4 Customer](#_Toc146706815) 3

[1.4 Non-Functional Requirement 3](#_Toc146706816)

[1.4.1 Usability: 3](#_Toc146706817)

[1.4.2 Accuracy: 3](#_Toc146706818)

[1.4.3 Availability: 3](#_Toc146706819)

[1.4.4 Maintainability: 3](#_Toc146706820)

[1.4.5 Response Time: .........................................................................................................................3](#_Toc146706820)

[1.4.6 Data Protection: .......................................................................................................................3](#_Toc146706820)

[2 Design and Implementation Constraints 4](#_Toc146706821)

[2.1 Use case diagram 4](#_Toc146706822)

[2.2 Activity diagram and Swimlane diagram](#_Toc146706823) 6

[2.3 Sequence diagram](#_Toc146706824) 10

[2.4 State diagram](#_Toc146706825) 12

[2.5 Class diagram](#_Toc146706826) 13

[2.6 Data flow diagram](#_Toc146706827) 14

[2.6.1 Context diagram (level-0)](#_Toc146706828) 14

[2.6.2 DFD Level-1 1](#_Toc146706829)4

[3 External interface requirement (Screens) 15](#_Toc146706831)

[3.1 Screen-1: Registration Form](#_Toc146706832) 15

[3.2 Screen-2: Taxi booking 17](#_Toc146706833)7

[3.3 Screen-3: Profile 19](#_Toc146706834)9

[4 Database design](#_Toc146706835) 21

[4.1 List of Tables](#_Toc146706836) 21

[5 Stories and Scenario](#_Toc146706837) 23

[5.1 Story-1: Add New Taxi](#_Toc146706838) 23

[5.1.1 Scenario# S1.1](#_Toc146706839) 23

[5.1.2 Scenario# S1.2](#_Toc146706840) 23

[5.1.3 Scenario# S1.3](#_Toc146706841) 24

[5.2 Story-2: Taxi booking](#_Toc146706842) 24

[5.1.1 Scenario# S2.1](#_Toc146706839) 24

[5.1.2 Scenario# S2.2](#_Toc146706840) 24

[6 Test cases](#_Toc146706846) 25

[7 References](#_Toc146706847) 29

# List of Figures

[Figure 2.1‑1 Use case diagram for Taxi management System](#_Toc146706848) 4

[Figure 2.1‑2 Use case diagram for Taxi management System](#_Toc146706848) 5

[Figure 2.2‑1 Activity diagram for taxi booking](#_Toc146706849) 6

[Figure 2.2‑2 Activity diagram for profile edit](#_Toc146706849) 7

[Figure 2.2‑2 Swimlane diagram for taxi booking](#_Toc146706850) 8

[Figure 2.2‑2 Swimlane diagram for profile edit](#_Toc146706850) 9

[Figure 2.3‑1 Sequence diagram for taxi bookin](#_Toc146706851) 10

[Figure 2.3‑1 Sequence diagram for profile edit](#_Toc146706851) 11

[Figure 2.4‑1 State diagram of Ride](#_Toc146706852) 12

[Figure 2.4‑2 State diagram for Driver 1](#_Toc146706853)2

[Figure 2.5‑1 Class diagram for Taximanagement system](#_Toc146706854) 13

[Figure 2.6‑1 Context diagram for Library management system](#_Toc146706855) 14

[Figure 2.6‑2 DFD level-1 for taxi management system](#_Toc146706856) 14

[Figure 3.1‑1 Screen-1: Registration Form 1](#_Toc146706858)5

[Figure 3.2‑1 Screen-2: Taxi bookin](#_Toc146706859) 17

[Figure 3.3‑1 Screen-3: Profile](#_Toc146706860) 18

# List of Tables

[Table 3.1‑1 Screen element of Registration form](#_Toc146706861) 15

[Table 3.2‑1 Screen element of taxi booking](#_Toc146706862) 17

[Table 3.3‑1 Screen element of Profile](#_Toc146706863) 19

[Table 4.1‑1 Table: Book](#_Toc146706864) 21

[Table 4.1‑2 Table: Borrower](#_Toc146706865) 21

[Table 4.1‑3 Table: Staff](#_Toc146706866) 21

[Table 4.1‑4 Table: Student](#_Toc146706867) 22

[Table 4.1‑4 Table: Student](#_Toc146706867) 22

# Introduction

## Product perspective

This project updates traditional taxi services into an internet-based system, making it easier for both drivers and passengers to use. It allows passengers to book taxis, track their rides, and manage their accounts online. For drivers, it provides tools to manage their schedules, accept ride requests, and track their earnings.

## Product features

### There are four different users who will be using this product:

* **Admin**
* **Manager**
* **Driver**
* **Customer**

### The features that are required for the **Admin** are:

* **User Management**: Add, edit, and delete users; assign roles (admin, manager, driver, customer).
* **System Configuration**: Set up system preferences including pricing, service areas, and vehicle options.
* **Financial Oversight**: Process payments, handle refunds, and monitor financial transactions.
* **Monitoring and Reporting**: Generate reports, track vehicle status, and oversee the system dashboard.
* **Complaint Resolution**: Address and resolve customer complaints and issues.
* **Promotions**: Create and manage promotional offers and discounts.
* **Compliance**: Ensure adherence to local regulations and data protection rules.
* **Data Management**: Perform system backups and restore the system when needed

### The features that are required for the **Manager** are:

* **Driver Scheduling**: Assign shifts and plan work schedules for drivers.
* **Trip Monitoring**: Track the status of ongoing trips and assist with issues.
* **Routing**: Provide drivers with optimized routes considering traffic conditions.
* **Performance Tracking:** Monitor and evaluate driver performance based on trip completion and ratings.
* **Customer Assistance**: Help customers with issues during their trips.
* **Vehicle Dispatch**: Assign vehicles to drivers as per requirements.
* **Revenue Analysis**: Review revenue data and assess profitability.
* **Feedback Review:** Look at customer feedback for service quality improvements.

### The features that are required for the **Driver** are:

* **Trip Management**: Accept or decline ride requests and navigate using maps.
* **Status Updates:** Change availability status (available, on trip, offline).
* **Payment Collection:** Collect payments through various methods from customers.
* **Customer Communication**: Contact customers for trip details and updates.
* **Vehicle Inspection**: Check the vehicle's condition before starting trips.
* **Emergency Handling**: Access emergency contacts and instructions.
* **Performance Tracking:** View personal performance metrics and customer ratings.

### The features that are required for the Customer are:

* **Ride Booking**: Request and book taxis via app, website, or phone.
* **Tracking**: Track the taxi's location and estimated arrival time.
* **Fare and Payment**: Get fare estimates and pay securely using various methods.
* **Profile Management**: Update personal details and preferences in their profile.
* **Trip History**: View details of past trips and payments.
* **Feedback and Ratings**: Rate drivers and provide feedback on trips.
* **Discounts and Offers:** Apply promo codes and discounts for rides.

## Functional Requirement

### **Admin**

* **Add Users**: Add new users like drivers, managers, and customers.
* **Edit Users**: Edit details of existing users.
* **Delete Users**: Remove users from the system.
* **Set Roles**: Assign roles (admin, manager, driver, customer) to users.
* **View Dashboard**: Access a main dashboard with key metrics and system status.
* **Configure Settings**: Set up system preferences like pricing, service areas, and vehicle options.
* **Generate Reports:** Create reports on bookings, revenue, and user activity.
* **Track Vehicles**: Monitor vehicle status, including availability and condition.
* **Handle Payments:** Oversee financial transactions, including processing payments and refunds.
* **Respond to Complaints**: Address and resolve customer complaints and issues.
* **Create Promotions**: Develop and activate promotional offers and discounts.
* **Ensure Compliance**: Verify adherence to local regulations and data protection rules.
* **Backup Data**: Perform regular system backups to prevent data loss.
* **Restore System**: Use backups to restore system functionality if needed.

### **Manager**

* **Assign Shifts**: Schedule drivers for specific shifts.
* **Monitor Trips**: Keep an eye on ongoing trips and their statuses.
* **Provide Routes**: Give drivers optimized routes considering traffic conditions.
* **Track Performance**: Check driver performance, like how well they complete trips and their ratings.
* **Help Customers**: Assist customers with issues during their trips.
* **Dispatch Vehicles**: Assign vehicles to drivers based on needs.
* **Plan Schedules**: Create and adjust driver work schedules.
* **View Feedback**: Look at customer feedback and ratings for quality improvements.
* **Resolve Issues**: Address complaints from customers regarding service or billing.
* **Check Revenues:** Review revenue data and profitability for different areas.

1.3.3 **Driver**

* **Accept Trips**: Accept or decline ride requests.
* **Use Navigation**: Access maps and directions for efficient travel.
* **Update Status**: Change status to show availability (available, on trip, offline).
* **Collect Payments**: Take payments from customers through various methods.
* **Communicate with Customers**: Contact customers for trip details and updates.
* **Inspect Vehicle**: Check vehicle before starting trips for safety.
* **Set Availability**: Update availability for taking rides.
* **Handle Emergencies:** Access emergency contacts and instructions.
* **View Feedback**: See customer ratings and comments.
* **Track Own Performance**: View personal performance data like trip completion and ratings.

1.3.4 **Customer**

* **Request Rides:** Book a taxi using the app, website, or phone.
* **Track Taxi:** See the taxi’s location and estimated arrival time.
* **Get Fare Estimate**: See an estimated fare before booking.
* **Pay for Rides:** Pay securely using various methods like cards or mobile wallets.
* **Cancel Rides**: Cancel a booking within a given time frame without charges.
* **Rate Drivers**: Provide ratings and feedback on drivers and trips.
* **Edit Profile**: Update personal details and preferences in their profile.
* **Receive Notifications:** Get alerts for booking confirmation, taxi arrival, and trip updates.
* **Use Accessibility Features:** Access options for special needs like wheelchair access.
* **View Trip History**: Check details of past trips and payments.
* **Apply Discounts:** Use promo codes and discounts when booking rides.
* **Schedule Rides:** Schedule rides for future dates and times.
* **Contact Driver:** Communicate with the driver directly through the app.
* **Choose Vehicle Type**: Select preferred vehicle type if available.
* **Save Locations:** Save frequently used locations like home or work.
* **View Driver Details:** See driver’s name, photo, and vehicle information before the ride.

### 

## Non-Functional Requirement

### **Usability**:

* The UI should be simple enough for everyone to understand and get the relevant information without any special training. Different languages can be provided based on the requirements.

### **Accuracy**:

* The data stored about the books and the fines calculated should be correct, consistent, and reliable.

### **Availability**:

* The System should be available for the duration when the library operates and must be recovered within an hour or less if it fails. The system should respond to the requests within two seconds or less.

### **Maintainability**:

* The software should be easily maintainable and adding new features and making changes to the software must be as simple as possible. In addition to this, the software must also be portable.

### **Response Time**:

### The system should respond to user queries and actions (e.g., booking a taxi, viewing available taxis) within 2 seconds under normal load conditions.

### **Data Protection**:

### All user data, including payment information and personal details, must be encrypted in transit (using TLS) and at rest (using AES-256 encryption).

# Design and Implementation Constraints

## Use case diagram

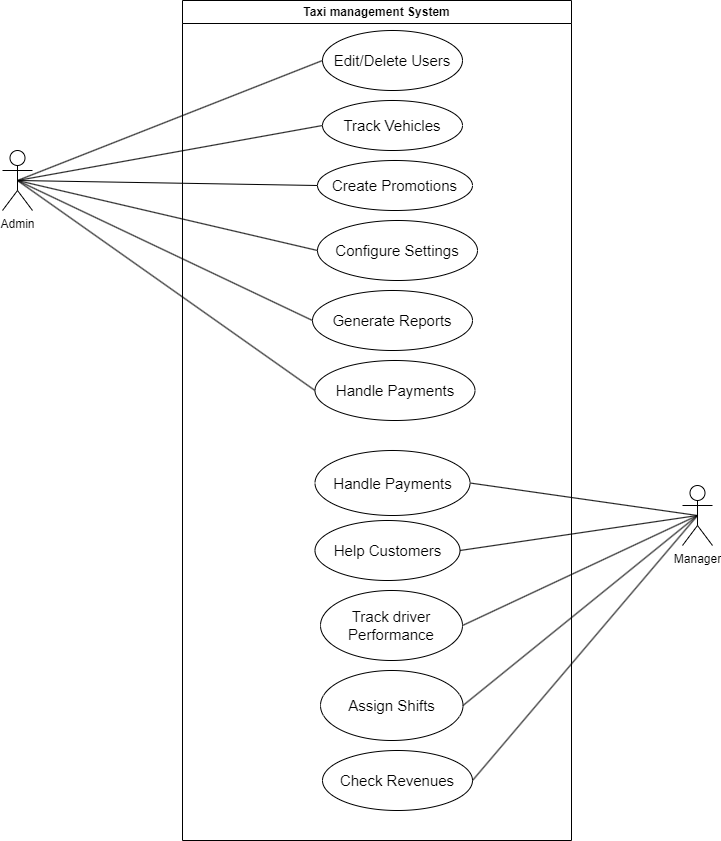


Figure 2.1‑1 Use case diagram for Taxi management System

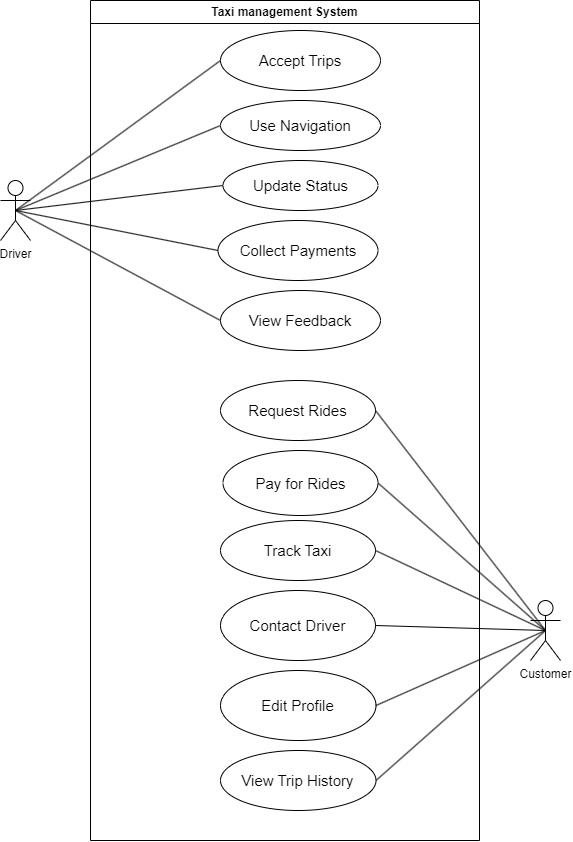


Figure 2.1‑1 Use case diagram for Taxi management System

## Activity diagram and Swimlane diagram

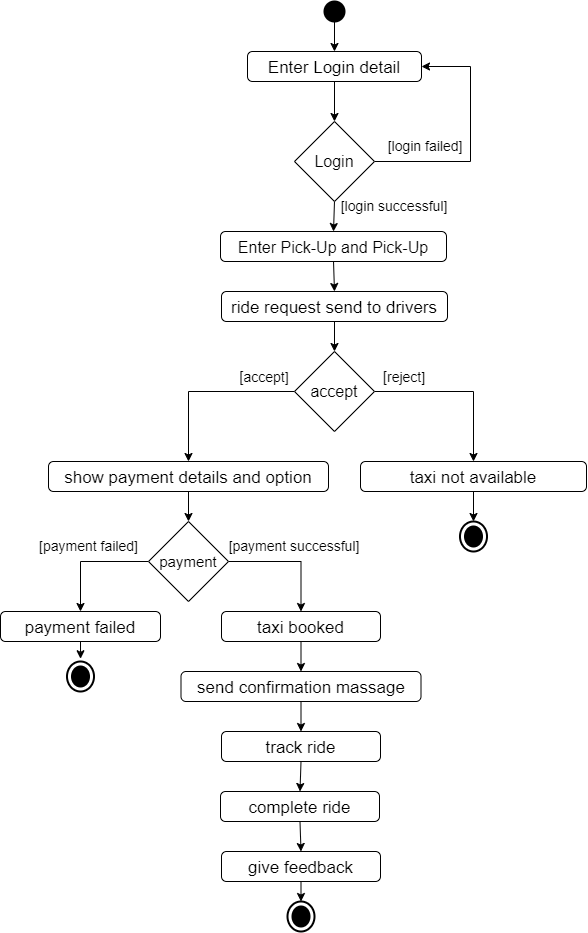


Figure 2.2‑1 Activity diagram for taxi booking

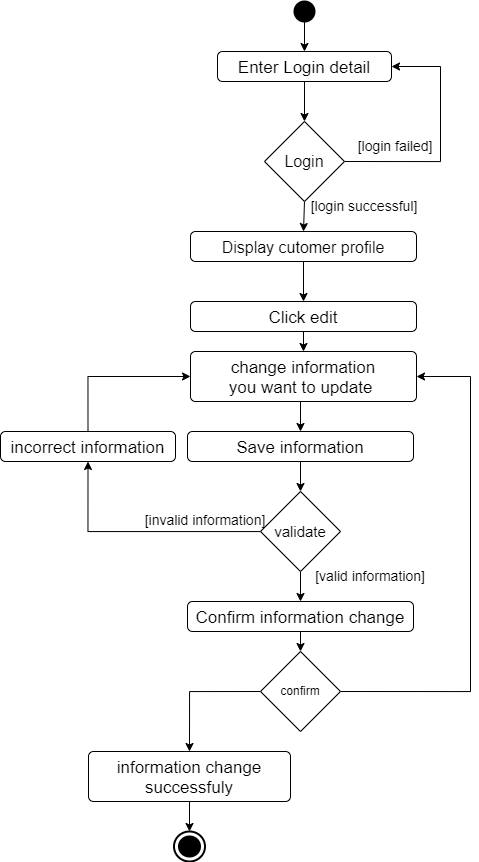


Figure 2.2‑1 Activity diagram for profile edit

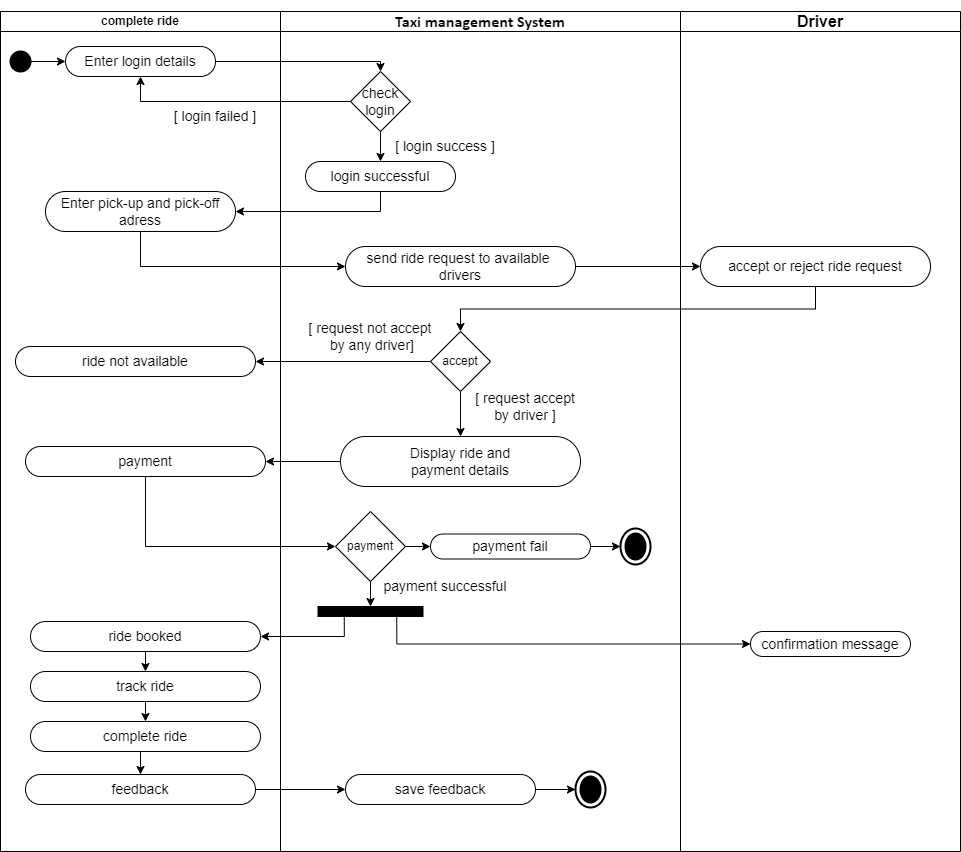


Figure 2.2‑3 Swimlane diagram for taxi booking

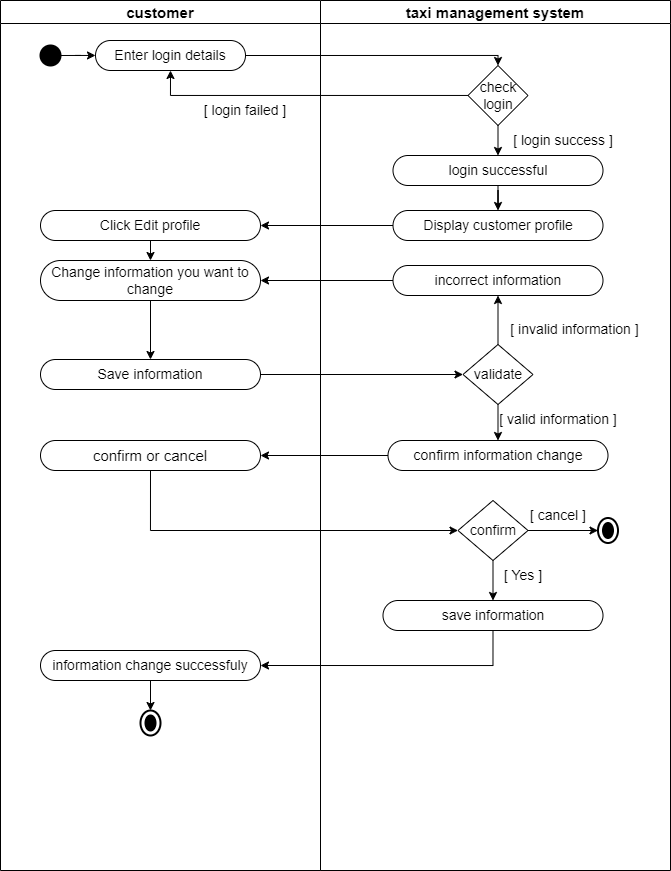


Figure 2.2‑4 Swimlane diagram for profile edit

## Sequence diagram

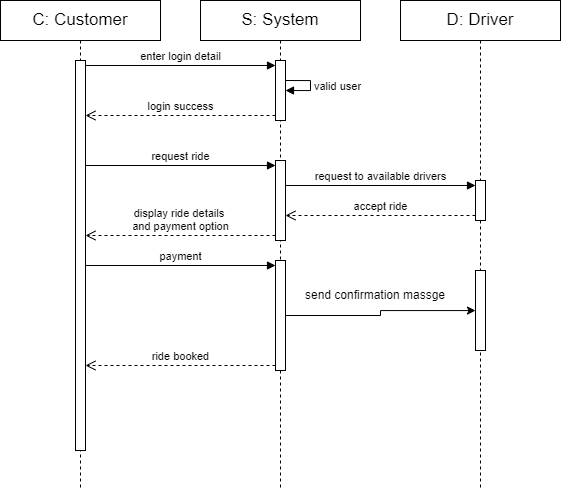
Untitled Diagram.drawio (6)Untitled Diagram.drawio (6)Untitled Diagram.drawio (6)

Figure 2.3‑1 Sequence diagram for taxi booking

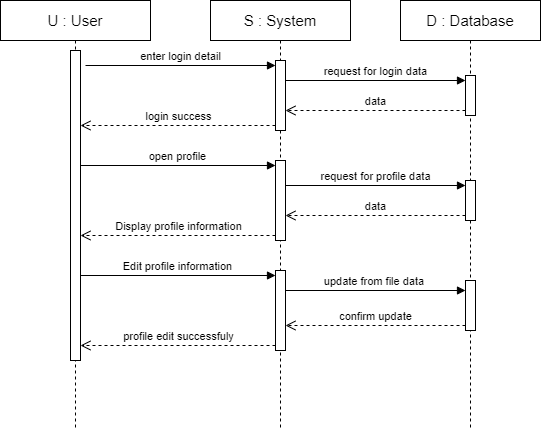
Untitled Diagram.drawio (6)Untitled Diagram.drawio (6)Untitled Diagram.drawio (6)

Figure 2.3‑2 Sequence diagram for profile edit

## State diagram

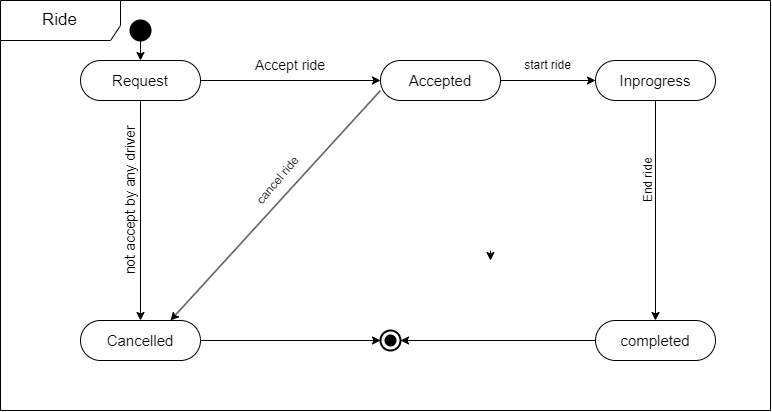


Figure 2.4‑1 State diagram for ride

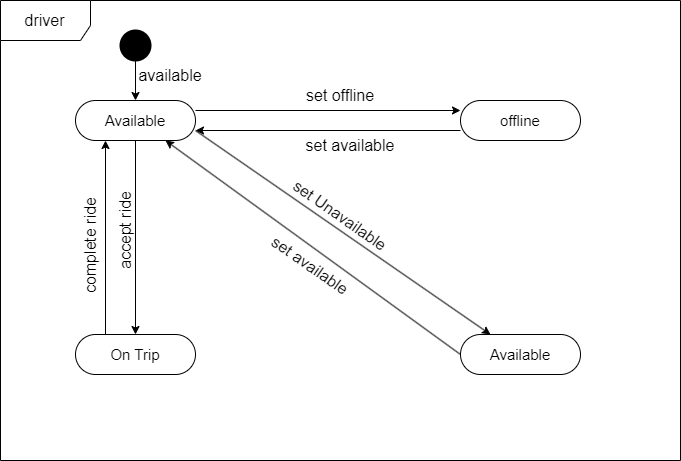


Figure 2.4‑2 State diagram for driver

## Class diagram

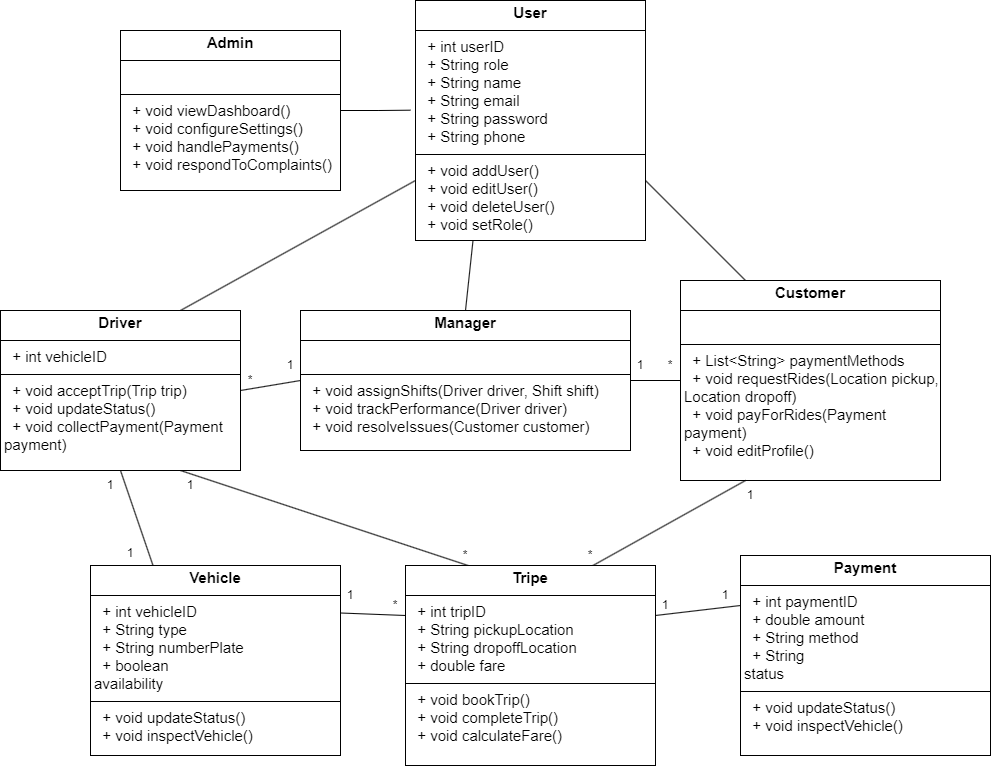


Figure 2.5‑1 Class diagram for Library management system

## Data flow diagram

### Context diagram (level-0)

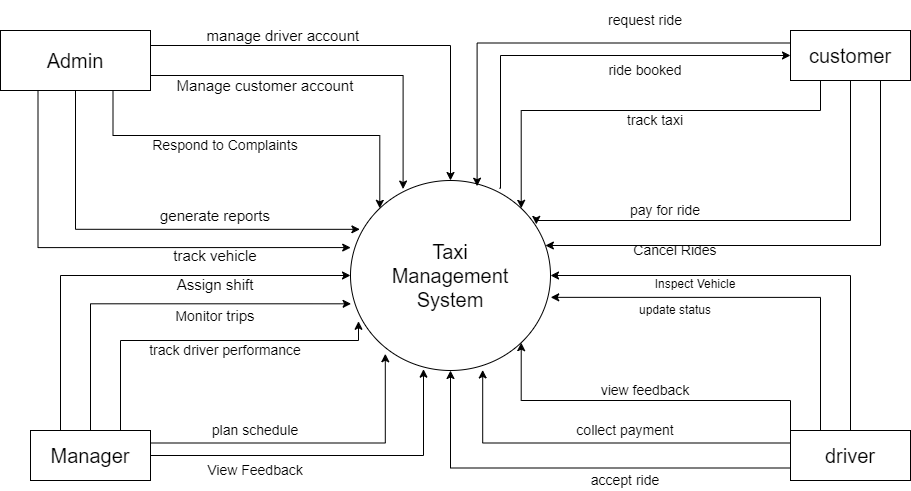


Figure 2.6‑1 Context diagram for taxi management system

### DFD Level-1

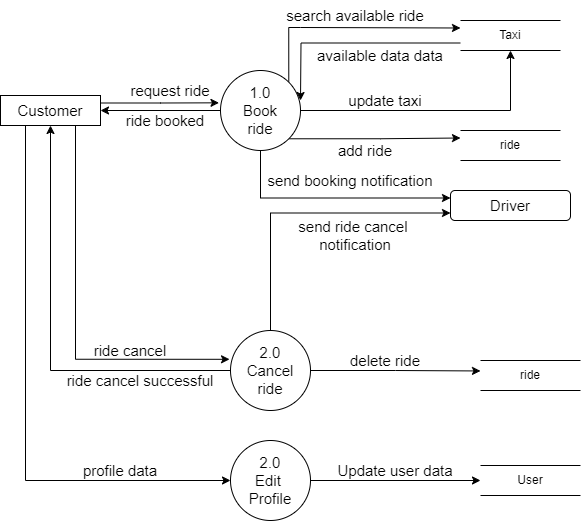


Figure 2.6‑2 DFD level-1 for taxi booking

# External interface requirement (Screens)

## Screen-1: Registration Form

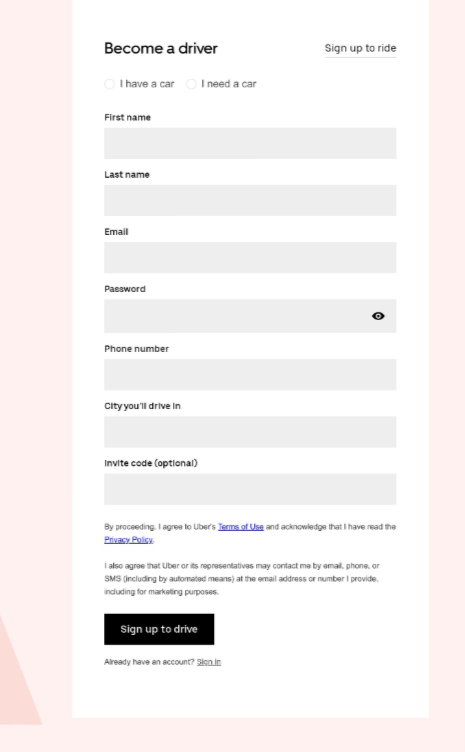


Figure 3.1‑1 Screen-1: driver Registration Form

**Purpose:** This form will allow the target end-users to register in the system. To register , the following information will be encoded in the system.

Table 3.1‑1 Screen element of Registration form

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Sr.** | **Screen Element** | **Input Type** | **O/M** | **1/N** | **Description** |
| **1** | I have a car / I need a car | Radio Button | M | 1 | select whether the user already has a car or needs one. The user must select one to proceed. |
| **2** | First Name | Textbox | M | 1 | Editable field for entering the user's first name. Should validate that the field is not empty. |
| **3** | Last Name | Textbox | M | 1 | Editable field for entering the user's last name. Should validate that the field is not empty. |
| **4** | Email | Email | M | 1 | Editable field for entering a valid email address. Must validate the format and ensure it is not empty. |
| **5** | Password | Password | M | 1 | Editable field for entering the user's password. Should mask input for privacy. Includes toggle visibility option. |
| **6** | Phone Number | Textbox | M | 1 | Editable field for entering a valid phone number. Must validate the format and ensure it is not empty. |
| **7** | City You'll Drive In | Textbox | M | 1 | Editable field for entering the city where the user will be driving. Must validate that it is not empty. |
| **8** | Invite Code (Optional) | Textbox | M | 1 | Editable field for entering an optional invite code. This field is not mandatory. |
| **9** | erms of Use & Privacy Policy Agreement | Link | M | 1 | Checkbox to confirm the user agrees to the Terms of Use and Privacy Policy. Must be checked to proceed |
| **10** | Sign Up to Drive | Button | M | 1 | Button to submit the registration form and proceed with account creation. |
| **11** | Already have an account? | Link | M | 1 | Link to navigate to the login page for users who already have an account. |

## Screen-2: Taxi Booking

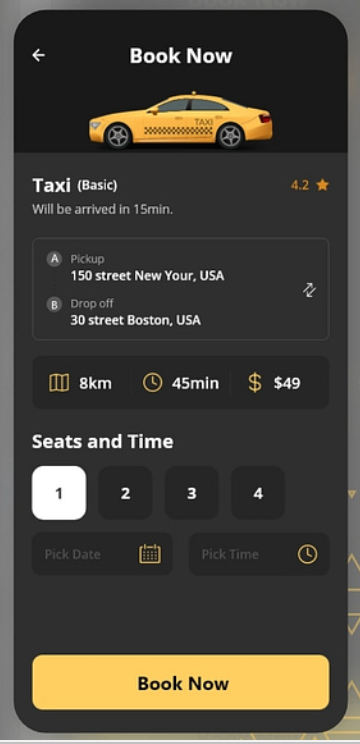


Figure 3.2‑1 Screen-2: taxi booking

**Purpose:** This form will be used by the system’s users to access records and features of the system. The users will input the correct combination of their username and password to be able to login to the system.

Table 3.2‑1 Screen element of taxi booking

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Sr.** | **Screen Element** | **Input Type** | **O/M** | **1/N** | **Description** |
| **1** | Taxi (Basic) | Label | M | 1 | Displays the type of taxi selected with a rating (4.2 stars) and estimated arrival time (15 min). |
| **2** | Pickup Location | Drop down | M | 1 | Displays the user's selected pickup location (e.g., "150 street New York, USA"). A refresh icon allows editing the location. |
| **3** | Drop-off Location | Drop down | M | 1 | Displays the user's selected drop-off location (e.g., "30 street Boston, USA"). A refresh icon allows editing the location. |
| **4** | Distance | Label | M | 1 | Displays the distance between the pickup and drop-off locations (e.g., 8 km). |
| **5** | Estimated time | Label | M | 1 | Displays the estimated travel time for the trip (e.g., 45 minutes).. |
| **6** | Estimated fare | Label | M | 1 | Displays the estimated fare for the trip (e.g., $49). |
| **7** | Seat Selection | Button Group | M | 1 | Allows the user to select the number of seats (1 to 4) required for the booking. |
| **8** | Pick Date | Date Picker | M | 1 | A field for selecting the date for the trip. Must be completed to proceed with the booking. |
| **9** | Pick Time | Date Picker | M | 1 | A field for selecting the time for the trip. Must be completed to proceed with the booking. |
| **10** | Book Now | Button | M | 1 | A button to confirm and finalize the booking process based on the selected options. |

## Screen-3: Profile

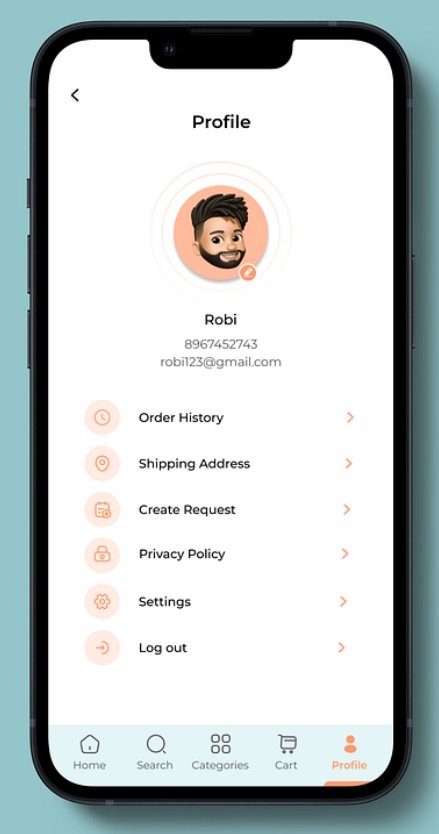


Figure 3.3‑1 Screen-3:profile view

**Purpose:** This module will allow the system administrator to add, edit, update or delete borrowers of book. The admin can add borrower information and manage it.

Table 3.3‑1 Screen element of Add borrower book

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Sr.** | **Screen Element** | **Input Type** | **O/M** | **1/N** | **Description** |
| **1** | Profile Picture | Image | M | 1 | Displays the user's profile picture. The image is editable, allowing the user to change their profile picture by clicking on the camera icon. |
| **2** | User Name | Label | M | 1 | Displays the user's name. |
| **3** | Contact Information | List Item | M | 1 | Displays the user's phone number and email address |
| **4** | Order History | List Item | M | 1 | Navigates to a screen displaying the user's order history when clicked. |
| **5** | Shipping Address | List Item | M | 1 | Navigates to a screen where the user can view or edit their shipping address. |
| **6** | Create Request | List Item | M | 1 | Navigates to a screen where the user can create a new request or inquiry. |
| **7** | Privacy Policy | List Item | M | 1 | Navigates to a screen displaying the app's privacy policy. |
| **8** | Settings | List Item | M | 1 | Navigates to the settings screen, allowing the user to configure various app settings. |
| **9** | Log out | List Item | M | 1 | Logs the user out of the app when clicked. |

# Database design

## List of Tables

* Users
* Vehicle
* Payments
* Trips
* Feedback

Table 4.1‑1 Table: Users

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Column** | **Data Type** | **Null** | **Keys & Constrains** | **Default Value & Description** |
| **user\_id** | int | Not Null | PK (Auto Increment) |  |
| **first\_name** | varchar(100) | Not Null |  |  |
| **last\_name** | varchar(100) | Not Null |  |  |
| **email** | varchar(100) | Not Null |  |  |
| **phone** | varchar(50) | Not Null |  |  |
| **password** | varchar(100) |  |  |  |
| **role\_id** | int |  | FK |  |
| **status** | ENUM |  |  |  |
| **profile\_picture** | BLOB |  |  |  |

Table 4.1‑2 Table: Vehicle

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Column** | **Data Type** | **Null** | **Keys & Constrains** | **Default Value & Description** |
| **vehicle\_id** | int | Not Null | PK (Auto Increment) |  |
| **driver\_id** | int | Not Null | FK | Reference of Driver Table |
| **vehicle\_type** | Varchar(50) | Not Null |  |  |
| **number\_plate** | Varchar(50) | Not Null |  |  |
| **status** | ENUM | Not Null |  |  |
| **location** | Varchar(50) | Not Null | FK |  |

Table 4.1‑3 Table: Payments

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Column** | **Data Type** | **Null** | **Keys & Constrains** | **Default Value & Description** |
| **payment\_id** | int | Not Null | PK (Auto Increment) |  |
| **trip\_id** | int | Not Null | FK | Reference of Trip Table |
| **amount** | Decimal(10,2) | Not Null |  |  |
| **payment\_method** | ENUM | Not Null |  |  |
| **status** | ENUM | Not Null |  |  |
| **created\_at** | datetime | Not Null |  |  |

Table 4.1‑4 Table: Trips

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Column** | **Data Type** | **Null** | **Keys & Constrains** | **Default Value & Description** |
| **trip\_id** | int | Not Null | PK (Auto Increment) |  |
| **customer\_id** | int | Not Null | FK | Reference of customer Table |
| **driver\_id** | int | Not Null | FK | Reference of driver Table |
| **pickup\_location** | Varchar(100) | Not Null |  |  |
| **dropoff\_location** | Varchar(100) | Not Null |  |  |
| **distance** | decimal | Not Null |  |  |
| **duration** | int | Not Null |  |  |
| **fare** | decimal | Not Null |  |  |
| **status** | ENUM | Not Null |  |  |
| **created\_at** | datetime | Not Null |  |  |

Table 4.1‑5 Table: Feedback

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Column** | **Data Type** | **Null** | **Keys & Constrains** | **Default Value & Description** |
| **feedback\_id** | int | Not Null | PK (Auto Increment) |  |
| **user\_id** | int | Not Null | FK | Reference of user Table |
| **trip\_id** | int | Not Null | FK | Reference of trip Table |
| **rating** | Int | Not Null |  |  |
| **comments** | text | Not Null |  |  |
| **created\_at** | datetime | Not Null |  |  |

# Stories and Scenario

## Story-1: Add New Taxi

|  |  |  |
| --- | --- | --- |
| *Story #* ***S1*** | **:** | **As a** Admin,  **I want to** add a new taxi to the system **So that** everyone can easily book a taxi. |
| **Priority** | **:** | High |
| **Estimate** | **:** | XL |
| **Reason** | **:** | The addition of a new taxi is crucial for ensuring that the fleet is up-to-date and accessible to customers. |

### Scenario# S1.1

|  |  |  |
| --- | --- | --- |
| *Scenario#* ***S1.1*** | **:** | Adding a New Taxi with Valid Information |
| **Prerequisite** | **:** | Admin is logged into the Taxi Management System. |
| **Acceptance Criteria** | **:** | **Given**: The Admin is navigated to the taxi management page. Valid taxi information, including license plate number, model, capacity, and other relevant details, is added.  **When**:  The Admin selects the "Add New Taxi" option  And the Admin enters valid taxi details  The Admin clicks the "Save" button to add the taxi to the fleet.  **Then**: The system successfully adds the taxi to the fleet, and the Admin receives a confirmation message with the taxi's identification number |

### Scenario# S1.2

|  |  |  |
| --- | --- | --- |
| *Scenario#* ***S1.2*** | **:** | Adding a New Taxi with Invalid Information |
| **Prerequisite** | **:** | Admin is logged into the Taxi Management System. |
| **Acceptance Criteria** | **:** | **Given**: The Admin is on the taxi management page **When**: The Admin selects the "Add New Taxi" option and enters incomplete or incorrect taxi details, and the Admin clicks the "Save" button to add the taxi to the fleet. **Then**: The system displays error messages for the incorrect or missing information, and the taxi is not added to the fleet. |

### Scenario# S1.3

|  |  |  |
| --- | --- | --- |
| *Scenario#* ***S1.3*** | **:** | Attempting to Add a Duplicate Taxi |
| **Prerequisite** | **:** | Admin is logged into the Taxi Management System and is on the taxi management page. |
| **Acceptance Criteria** | **:** | **Given**: The taxi information, including license plate number, model, capacity, and other relevant details, is available, and the taxi with the same license plate number is already in the fleet. **When**: The Admin clicks on the “Add Taxi” button and enters duplicate taxi details. **Then**: The system alerts the Admin about the duplicate taxi and does not add it to the fleet. |

## Story-2: Booking Taxi

|  |  |  |
| --- | --- | --- |
| *Story #* ***S2*** | : | **As a** Customer,  **I want to** book a taxi,  **So that** I can travel to my destination easily. |
| **Priority** | **:** | High |
| **Estimate** | **:** | M |
| **Reason** | **:** | Booking a taxi is the core functionality of the system and is crucial for user satisfaction. |

### Scenario# S2.1

|  |  |  |
| --- | --- | --- |
| *Scenario#* ***S2.1*** | **:** | Booking a Taxi with Valid Information |
| **Prerequisite** | **:** | Customer is logged into the Taxi Management System. |
| **Acceptance Criteria** | **:** | **Given**: The Customer is on the taxi booking page.  **When**:  The Customer selects pickup and drop-off locations  The Customer chooses a vehicle type and enters valid booking details  The Customer confirms the booking.  **Then**: The system confirms the booking, provides the estimated fare, and displays the driver's details and the estimated arrival time. |

### 5.2.2 Scenario# S2.2

|  |  |  |
| --- | --- | --- |
| *Scenario#* ***S2.2*** | **:** | Booking a Taxi with Invalid Information |
| **Prerequisite** | **:** | Customer is logged into the Taxi Management System. |
| **Acceptance Criteria** | **:** | **Given**: The Customer is on the taxi booking page. **When**: The Customer enters incomplete or incorrect booking information and confirms the booking. **Then**: The system displays error messages for missing or invalid details, and the booking is not confirmed. |

# Test cases

|  |  |  |  |
| --- | --- | --- | --- |
| **Project Name:** | Taxi Management System | **Test Designed by:** | Bhavesh Kadachha |
| **Module Name:** | Register | **Test Designed date:** | 01-08-2024 |
| **Release Version:** | 1.0 | **Test Executed by:** | Bhavesh Kaachha |
|  |  | **Test Execution date:** | 15-08-2024 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Pre-condition**: **Web application should be accessible** | | | | |
| **Test Case ID** | **Test Title** | **Test Type** | **Description** | **Test Case ID** |
| **TC\_001** | Register to web application with valid credential | Functional | Register to Taxi management system web application through valid details | TC\_001 |
| **TC\_002** | Register to web application with invalid credential | Functional | Register to Taxi management system web application through invalid details | TC\_002 |
| **TC\_003** | Varify register page elements | GUI | varify that all elements are availabe on register page | TC\_003 |

|  |  |
| --- | --- |
| **Test Case Title** | Register to web application with valid details |
| **Test Type** | Functional |
| **Test Priority** | High |
| **Pre-condition** | Web application should be accessible |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Step** | **Test Case Description** | **Expected Result** | **Actual Result** | **Status** | **Comment** | **Data** | **BUG ID** |
| 1 | Access the driver sign-up form | The form should load properly when accessing the URL | Form loads successfuly | Pass |  |  |  |
| 2 | Select the "I have a car" option | The radio button should be selectable | "I have a car" option selected | Pass |  |  |  |
| 3 | Select the "I need a car" option | The radio button should be selectable and deselect "I have a car" | "I need a car" option selected, "I have a car" deselected | pass |  |  |  |
| 4 | Enter valid data in "First name" field | The first name field should be editable and accept valid input | Input accepted in "First name" field | Pass |  | First Name: John |  |
| 5 | Enter valid data in "Last name" field | The last name field should be editable and accept valid input | Input accepted in "Last name" field | pass |  | Last Name: Doe |  |
| 6 | Enter valid email in "Email" field | The email field should be editable, accept a valid email, and show validation | Valid email input accepted | pass |  | Email: johndoe@example.com |  |
| 7 | Enter valid password in "Password" field | The password field should be editable and display the password as masked | Password input accepted and masked | pass |  | Password: \*\*\*\*\*\* |  |
| 8 | Enter valid phone number in "Phone number" | The phone number field should be editable and accept valid input | Phone number input accepted | pass |  | Phone: 987654321 |  |
| 9 | Enter valid city in "City you'll drive in" field | The city field should be editable and accept valid input | City input accepted | pass |  | City: New York |  |
| 10 | (Optional) Enter invite code in "Invite code" field | The invite code field should accept alphanumeric input | Invite code input accepted | pass |  | Invite Code: ABC123 |  |
| 11 | Click the "Sign up to drive" buttonThe form should load properly when accessing the URL | Form should be submitted, and user should be directed to the next step | Form submitted and user directed to next step | pass |  |  |  |

|  |  |
| --- | --- |
| **Test Case Title** | Login to web application with invalid credential |
| **Test Type** | Functional |
| **Test Priority** | Medium |
| **Pre-condition** | Web application should be accessible |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Step** | **Test Case Description** | **Expected Result** | **Actual Result** | **Status** | **Comment** | **Data** | **Bug ID** |
| 1 | Verify that User is not able to Login with invalid Username and invalid Password | Should be display an error message enter wrong username or password | Display an error of wrong username and password | Pass |  |  |  |
| 2 | Verify that User is not able to Login with Valid Username and invalid Password | Should be display an error message enter wrong password | Display an error of wrong password | Pass |  |  |  |
| 3 | Verify that User is not able to Login with invalid Username and Valid Password | Should be display an error message User not found | Display an error Username not found | Pass |  |  |  |
| 4 | Verify that User is not able to Login with blank Username or Password | Set required field validation message for Username and Password | Display an error of wrong username and password | Fail | Not performa a validation function fix it |  |  |

|  |  |
| --- | --- |
| **Test Case Title** | Varify login page elements |
| **Test Type** | GUI |
| **Test Priority** | Medium |
| **Pre-condition** | Web application should be accessible |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Step** | **Test Case Description** | **Expected Result** | **Actual Result** | **Status** | **Comment** | **Data** | **Bug ID** |
| 1 | Launch application with the given url | The site launched properly | Site launched successfully | Pass |  |  |  |
| 2 | Verify that the login screen contains elements such as Username, Password, Sign in button, Remember password check box, Forgot password link, and Create an account link. | All listed control displayed properly on the page | Login page loaded successfully | Pass |  |  |  |
| 3 | Verify that cursor is focused on “Username” text box on the page load | Cursor is focused in Username textbox | Cursor focus in Username textbox | Pass |  |  |  |
| 4 | Verify that tab functionality is working properly or not | When tab pressed cursor move in next control | Cursor moving in next control | Pass |  |  |  |
| 5 | Verify that all the fields such as Username, Password has a valid placeholder | All text fields have proper placeholder | All text fields have proper placeholder | Pass |  |  |  |
| 6 | Verify that the labels float upward when the text field is in focus or filled (In case of floating label) | When field is focused or filled, label display on top of the filled | When field is focus or filled, label display on top of the filled | Pass | step required when fields with floating label |  |  |
| 7 | verify that forgot password link working properly | when click on forgot password load forgot passworg page | forgot password link not working | Fail |  |  |  |

# References

* http://www.w3schools.com/html/html\_intro.asp
* https://www.w3schools.com/php/default.asp
* https://www.javatpoint.com/uml