# Design Document

for

# CAB SHARING

Version <1.1>

Prepared by

Group 18: Group Name: Code Closed

| | | |
|---|---|---|
| AAKASH LAWA | 190006 | aaklawa@iitk.ac.in |
| ABHISHEK GURJAR | 190037 | gurjara96@gmail.com |
| AKASH KUMAR BHOI | 190082 | akashbhoi525@gmail.com |
| RISHABH MUKATI | 190704 | mukatirishabh02@gmail.com |
| PRINCE KUMAR AHIRWAR | 190646 | princeprinceahirwar@gmail.com |
| HARIOM SHAKYAWAL | 190354 | hariomkoli306@gmail.com |
| GOPAL AGGARWAL | 200390 | gopalaggarwal5858@gmail.com |
| UJJAWAL GOYAL | 201058 | ujjawalgo247@gmail.com |
| SOURABH MINA | 200996 | sourabh2002wow@gmail.com |
| SOURAV ANAND | 200997 | souravanand1982@gmail.com |

Course: CS253

Mentor TA: Pinaki Chakraborty

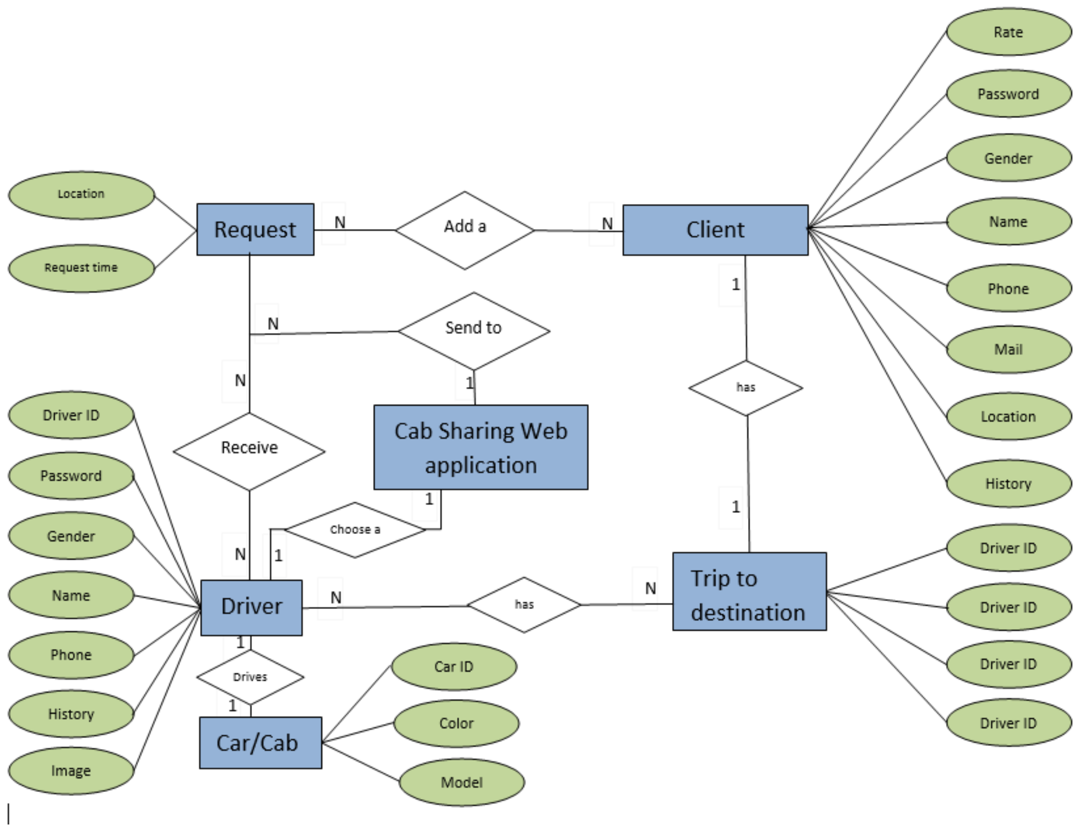
Date: 15 Feb, 2022

# Revisions

| Version | Primary Author(s) | Description of Version | Date Completed |
|---|---|---|---|
| Draft Type and Number | Full Name | Information about the revision. This table does not need to be filled in whenever a document is touched, only when the version is being upgraded. | 00/00/00 |

# 1 Context Design

## 1.1 Context Model



## 1.2 Human Interface Design

### Login/ Registration

On this page, users can register themselves. If the user is already registered, they can log in using
their mobile no. or email id and a generated password. After the registration, the user goes to their
respective interface.

### Campus Community Interface

Other than the menu bar, news and advertisements appear on the dashboard.
The things on the dashboard menu of the campus community are

● **Profile:** It contains all the user's details. They can edit their profile.
● T**ravel detail:** In these sections, you can apply for cab sharing and fill in the information like
date, destination, luggage info, time, etc. The user can also update this info.
● **Match:** These buttons match the info which you give on the Travel detail page with other users and provide all the matches (if available).
● Search: In this, users can search for others by their initial and final destination, dates, etc.
● **Notification:** If someone sends a request or accepts/rejects the request, the notification comes in these sections.
● **Requests:** There are two sections in this
  ○ **Sent:** All the requests the user sends appear in this section.
  ○ **Received:** In this section, all the requests which are received appear.
● **Chat:** In this section, the user can chat with other users.
● **Driver Detail:** This section will show all the details of cab drivers.

## CAB Driver Interface

At the dashboard, news and advertisements appear other than the menu bar.
The things that are on the dashboard menu of the cab driver are

● **Profile:** It contains all the details of the user. They can edit their profile and update their status
here.
● **Notification:**The notification appears on these sections if someone sends a request.
● **Request:** In these sections, all the requests come which are sent by other users.

# 2 Architecture Design

## MVC (Model-View-Controller)

Model-View-Controller is a software design pattern used to implement user interfaces, data, and controlling logic. Our software is a user-interacted web application; hence, this design pattern is chosen.

The three parts of the MVC software design pattern can be described as follows:

1. Model: Manages data and business logic.
2. View: Handles layout and display.
3. Controller: Routes commands to the model and view parts.

**Models**

The model defines the type of data the database must contain. If the state of this data changes, then the model will usually notify the view (so the display can change as needed) and sometimes the controller (if different logic is required to control the updated view).

Our web application has:-

- User Database: It stores all data about a system, for example, user's personal information, journey time, and destination, saved login information.

**View**

The view defines how the data should be presented to the user (layout and display on the application).
We design to have different views as:-

- Login and registration
- User Profile View
- User bookings View
- Journey summary View
- About the developers View
- Notification View

**Controller**

The controller contains logic that updates the model and views in response to users' input. We have input forms and buttons to book or cancel rides. These actions require the model to be updated, so the input is sent to the controller, which then manipulates the model as appropriate and sends updated data to the view.

We have:-

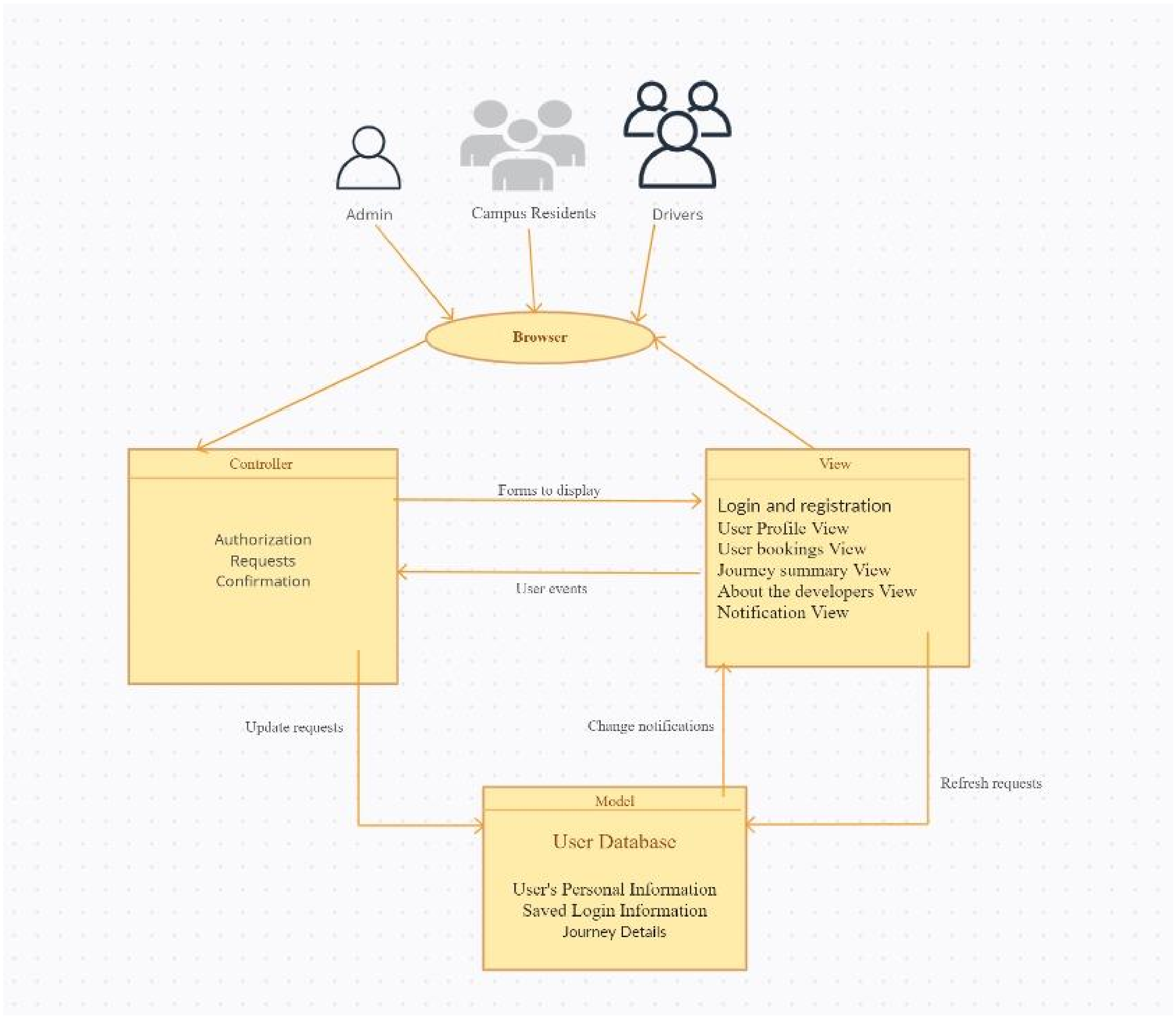
- Authorisation
- Requests
- Confirmation

The *authorisation* comprises the Login and Registration.

The *requests* control undertakes:-

- Booking/cancelling a ride
- Updating user information
- Adding a match to your journey
- Chats

Confirmation updates the journey details after enquiring with the matched people and the driver.

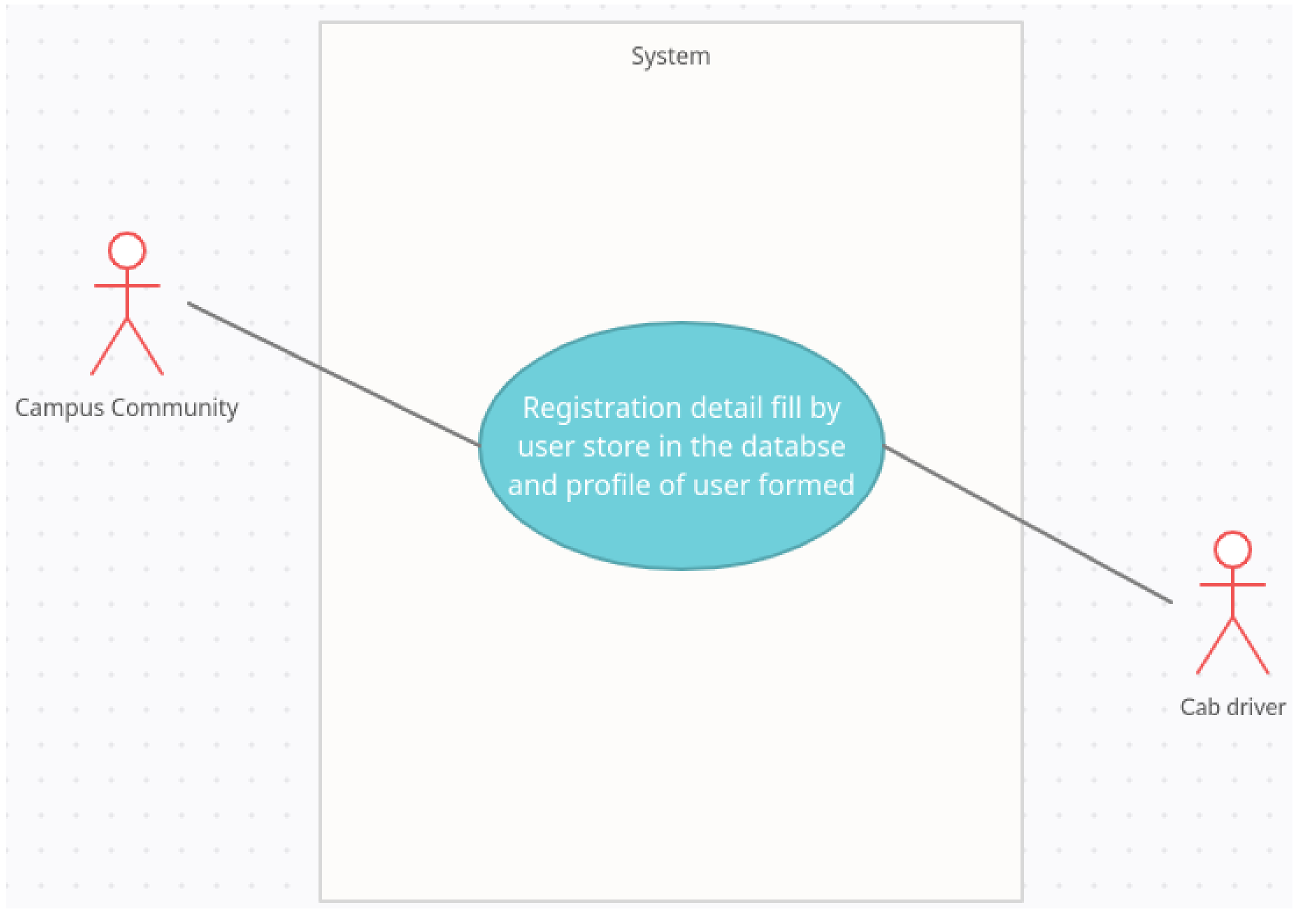

Admin
Campus Residents
Drivers
Browser
Controller
Authorization
Requests
Confirmation
Forms to display
User events
View
Login and registration
User Profile View
User bookings View
Journey summary View
About the developers View
Notification View
Update requests
Change notifications
Refresh requests
Model
User Database
User's Personal Information
Saved Login Information
Journey Details

# 3 Object Oriented Design

## 3.1 Use Case Diagrams

### 3.1.1 Registration

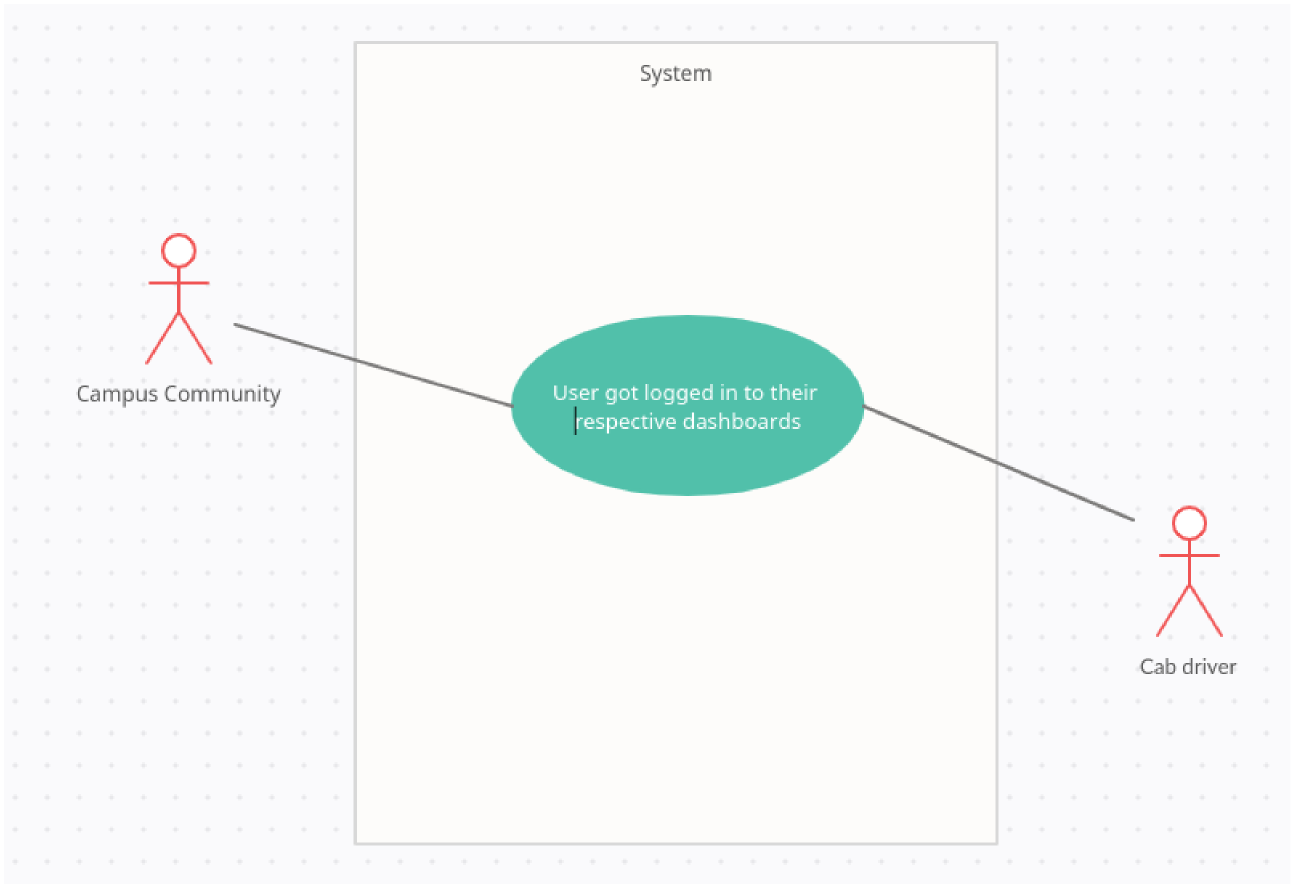
| | |
|---|---|
| Use case | U1 |
| Author | Hariom Shakyawal |
| Purpose | User registration and profile making |
| Requirement Traceability | The User has access to the registration page. |
| Priority | High |
| Preconditions | None |
| Postconditions | Users can sign in at any time and can use the system. |
| Actor(s) | User |
| Exception | If a given username is not available, users have to refill details. |


System
Campus Community
Registration detail fill by user store in the databse and profile of user formed
Cab driver

### 3.1.2 Sign In/ Log In

| | |
|---|---|
| Use case | U2 |
| Author | Hariom Shakyawal |
| Purpose | For checking that user is registered and get his page |
| Requirement Traceability | The User has access to the login page and has login details |
| Priority | High |
| Preconditions | User must be registered. |
| Postconditions | User will be able to use the system. |
| Actor(s) | User |

| | |
|---|---|
| Exception | If the username or password is found invalid user can not enter the system. |
| Includes | U1 |



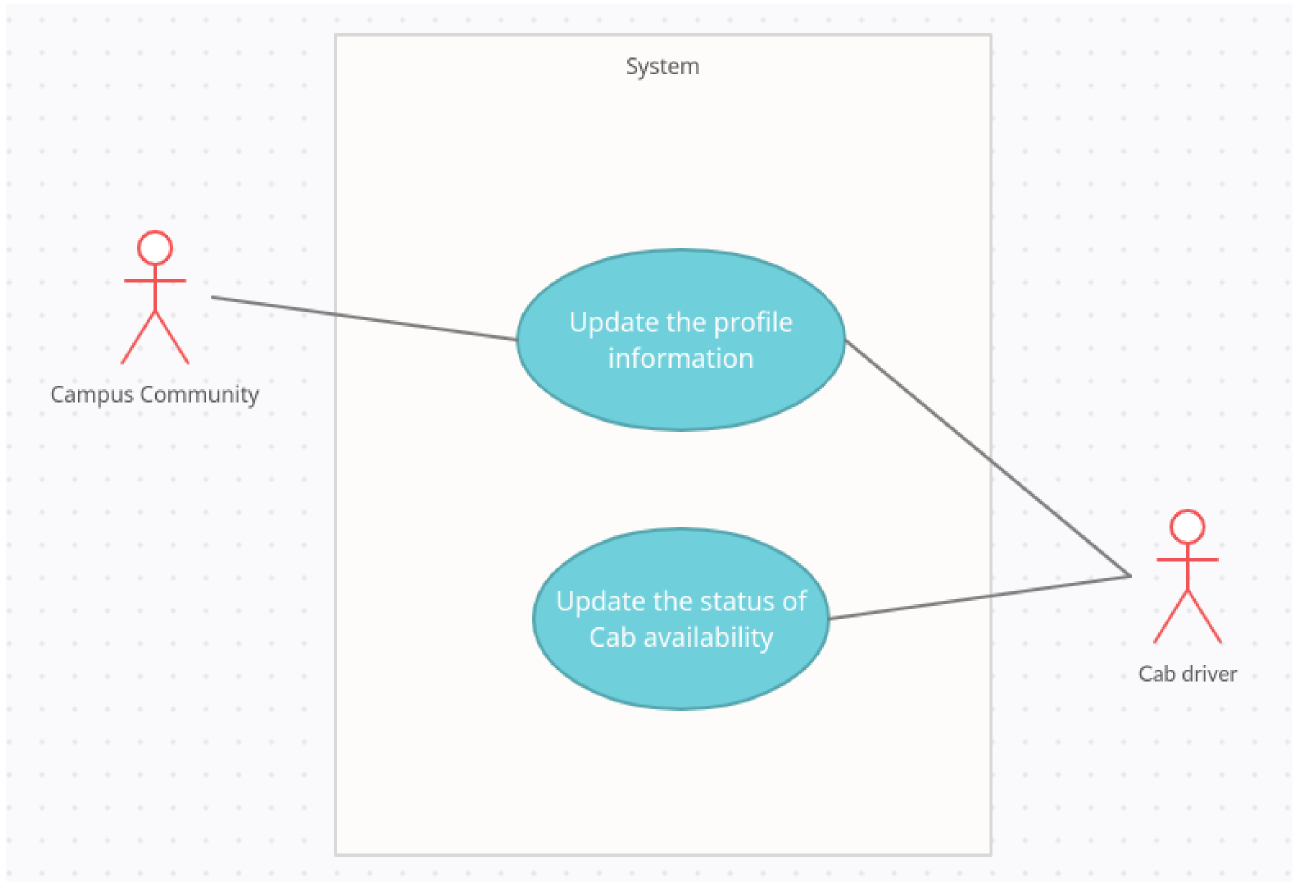
### 3.1.3 Forgot Password

| | |
|---|---|
| Use case | U3 |
| Author | Gopal Aggarwal |
| Purpose | For resetting users password |
| Requirement Traceability | The User has access to the login page |
| Priority | Medium |
| Preconditions | User should be registered |

| | |
|---|---|
| Postconditions | User will be able to login with a new password |
| Actor(s) | User |
| Exceptions | If the user is found unregistered, the user will be prompted to register on the page. |
| Includes | U1 |

### 3.1.4 Update Information

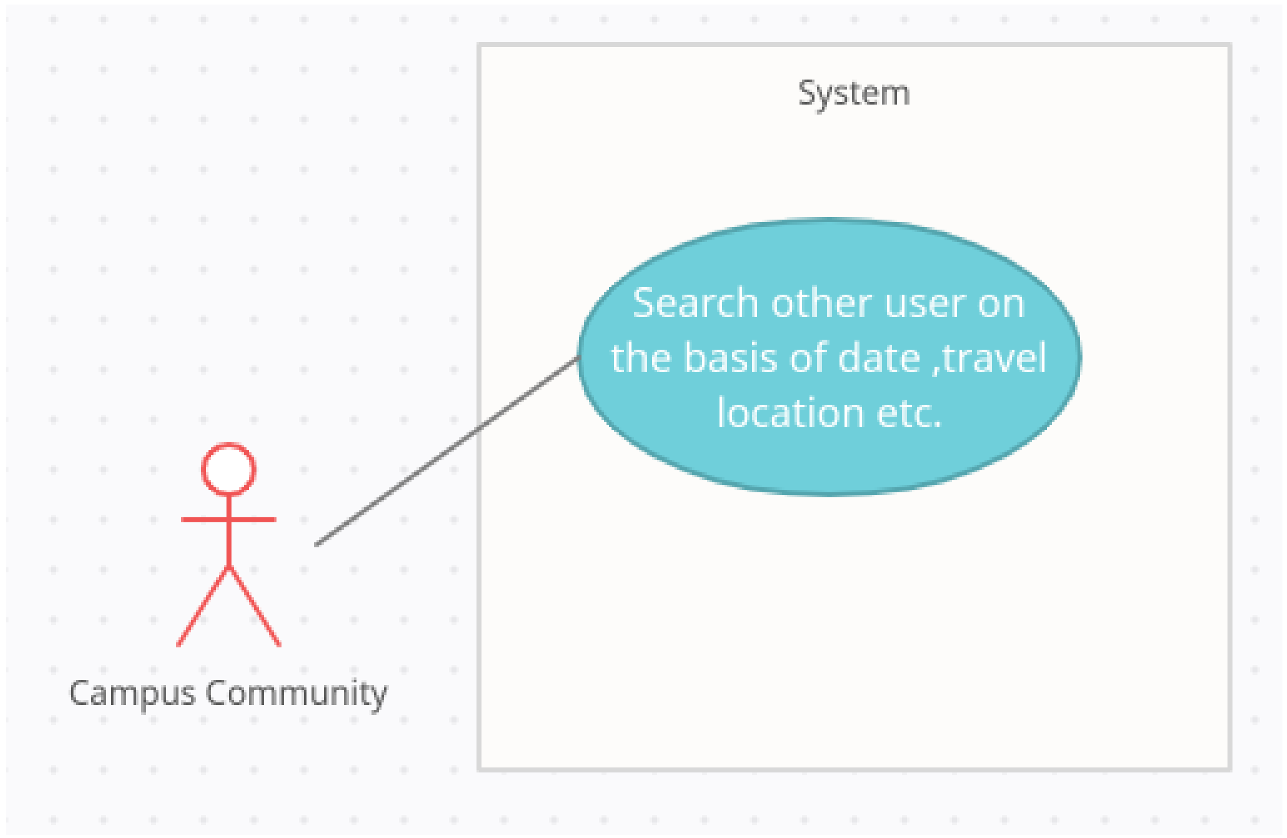
| | |
|---|---|
| Use case | U4 |
| Author | Rishabh Mukati |
| Purpose | For updating the profile information of the user and cab driver can also update that they are Available/Not Available |
| Requirement Traceability | The User has access to the login page and has login details and access to the profile section. |
| Priority | High |
| Preconditions | User should be a registered member and logged in to the system |
| Postconditions | User profile gets updated, and cab driver status gets updated |
| Actor(s) | Cab Availability status- Cab driver, Update profile - Users |
| Exception | None |
| Includes | U2 |


System
Campus Community
Update the profile information
Update the status of Cab availability
Cab driver

### 3.1.5 Search

| | |
|---|---|
| Use case | U5 |
| Author | Sourav Anand |
| Purpose | To help users searching for vacant seats in cabs |
| Requirement Traceability | The User has access to the login page and has login details and also has access to the search section. |
| Priority | High |
| Preconditions | User should be signed in |
| Postconditions | User will be able to see a list of available seats |
| Actor(s) | Campus Community |
| Exception | none |

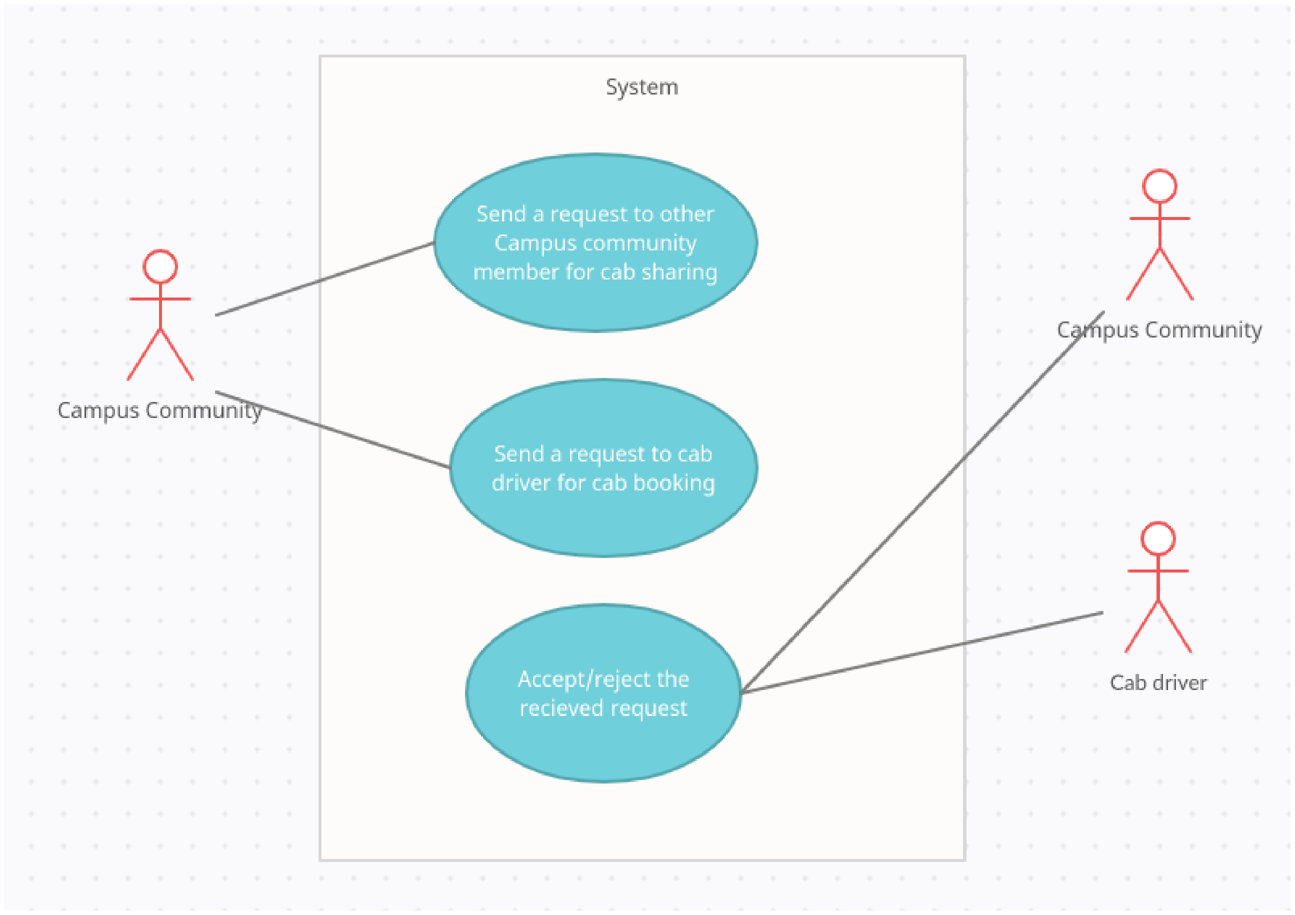
| | |
|---|---|
| Includes | U2 |


System
Search other user on the basis of date ,travel location etc.
Campus Community

### 3.3.6 Sending and Accepting/Rejecting Request

| | |
|---|---|
| Use case | U6 |
| Author | Rishabh Mukati |
| Purpose | The Campus community can send a request to other Campus community members for cab sharing or to the cab driver for booking cabs, and they can accept or reject the request. |
| Requirement Traceability | The User has access to Search, or the user has access to Travel detail and matches pages and Cab details page to send requests to other users.<br>The campus community has access to the request page to see the request and accept that. |
| Priority | High |
| Preconditions | User should be a registered member and logged in to the system |

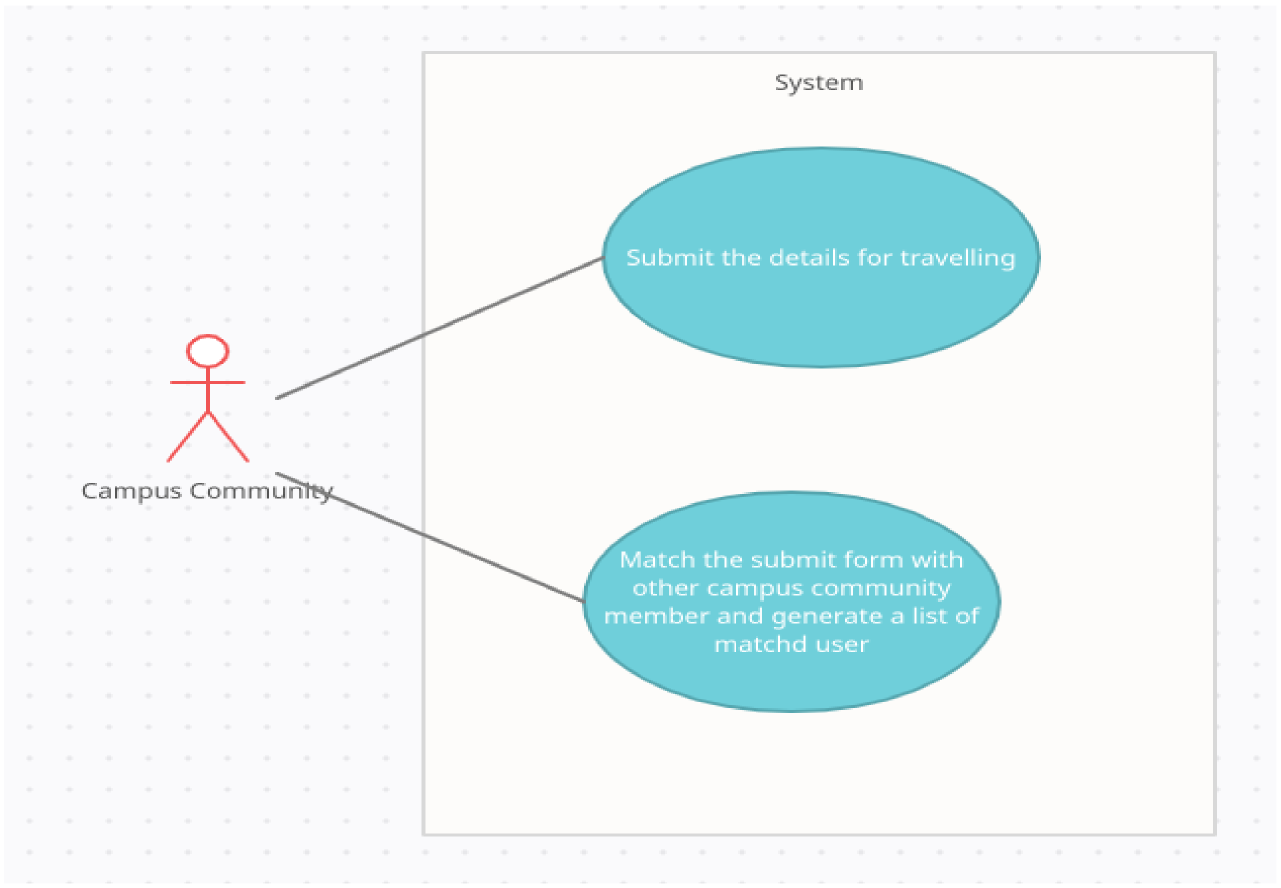
| | |
|---|---|
| Postconditions | The user who receives a request will be able to see the request in the request section and can accept or reject the request. |
| Actor(s) | Sending request - Campus Community Accepting/Rejecting request - Users |
| Exception | None |
| Includes | U2, U5 |


System
Send a request to other Campus community member for cab sharing
Send a request to cab driver for cab booking
Accept/reject the recieved request
Campus Community
Campus Community
Cab driver

### 3.1.7 Travel detail and Match

| | |
|---|---|
| Use case | U7 |
| Author | Rishabh Mukati |
| Purpose | To fill the details about travel and match with other users |
| Requirement | The User has access to the login page and has login details and also has |

| | |
|---|---|
| Traceability | access to the search section. |
| Priority | High |
| Preconditions | Users should be registered members and logged into the system and have access to the Travel details and matches . |
| Postconditions | The travel details will be stored in the database.<br>The user will be matched with another user and the list of match users will appear. |
| Actor(s) | Campus Community |
| Exception | None |
| Includes | U2 |


System
Submit the details for travelling
Campus Community
Match the submit form with other campus community member and generate a list of matchd user

## 3.2 Class Diagrams

### 3.2.1 User Class

| Key | Data Type | Description |
|---|---|---|
| my_name | string | Name of the user |
| user_email | string | Email Id of the user |
| mobile_number | long int | Mobile number of the user |
| | | |
| user_address | string[ ] | Address |
| user_name | string | Username |
| user_password | string | Password |
| user_role_id | int | Specify the role of the user |
| addUser() | module | Add a new user |
| deleteUser() | module | Delete an existing user |
| serachUser() | module | Search an user |

### 3.2.2 Role Class

| Key | Data Type | Description |
|---|---|---|
| role_id | int | Specify the role of the user |
| role_title | string | Type of role of user |
| role_discription | string | Describe role of user |
| addRole() | module | Add a role of user |
| deleteRole() | module | Delete an role |

| | | |
|---|---|---|
| searchRole() | module | Search user's role |

### 3.2.3 Verification Class

| Key | Data Type | Description |
|---|---|---|
| is_valid_username( ) | module | Check is username valid |
| is_valid_password( ) | module | check is password is valid |
| is_email_taken( ) | module | check user's email |
| is_user_found( ) | module | Check is user id registered |
| is_passord_matched( ) | module | check password validity |

### 3.2.4 Student Class

| Key | Data Type | Description |
|---|---|---|
| student_id | int | Assign student ID |
| student_name | string | name of student |
| student_email | string | Email Id of student |
| student_mobile_number | long int | took student's mobile number |
| student_address | string[ ] | Took student's address |
| update_details() | module | for update above informations |
| search_for_trip() | module | search for suitable trip partner |
| book_trip() | module | for confirmation of trip |
| send_message() | module | open chat box |

### 3.2.5 Driver Class

| Key | Data Type | Description |
|---|---|---|
| driver_id | int | identify drivers |
| driver_name | string | name of driver |
| driver_email | string | email of driver |
| driver_mobile_number | long int | mobile no. of driver |
| driver_user_name | string | login user name of driver |
| driver_pasword | string | login password of driver |
| update_details() | module | for update above informations |
| confirm_avalibilty() | module | for availability of driver |
| add_vechile( ) | module | vehicle of driver |
| accept_request() | module | accept or decline requests |
| update_price( ) | module | update price for ride |

### 3.2.6 Admin Class

| Key | Data Type | Description |
|---|---|---|
| admin_id | int | Identify of admin |
| admin_name | string | took admin's name |
| admin_user_name | string | login user name of admin |
| admin_password | string | login password of admin |

| | | |
|---|---|---|
| delete_user( ) | module | delete user as admin |
| add_user( ) | module | add user as admin |
| update_details( ) | module | update details of admin |

### 3.2.7 Vehicle Class

| **Key** | **Data Type** | **Description** |
|---|---|---|
| vehicle_id | int | Id for driver's vehicles |
| vehicle_size | int | seat capacity of vehicle |
| vehicle_model | string | type of vehicle |
| driver_id | int | Id of driver |
| delete_vehicle( ) | module | delete vehicle of driver |

### 3.2.8 Chat class

| **Key** | **Data Type** | **Description** |
|---|---|---|
| chat_window_id | int | Id of the chat window |
| students_id | int [ ] | Ids of the students in the window |
| messages | string [ ] | Set of message in that window |
| send_meassage( ) | module | to send a message |
| delete_message( ) | module | delete a message |

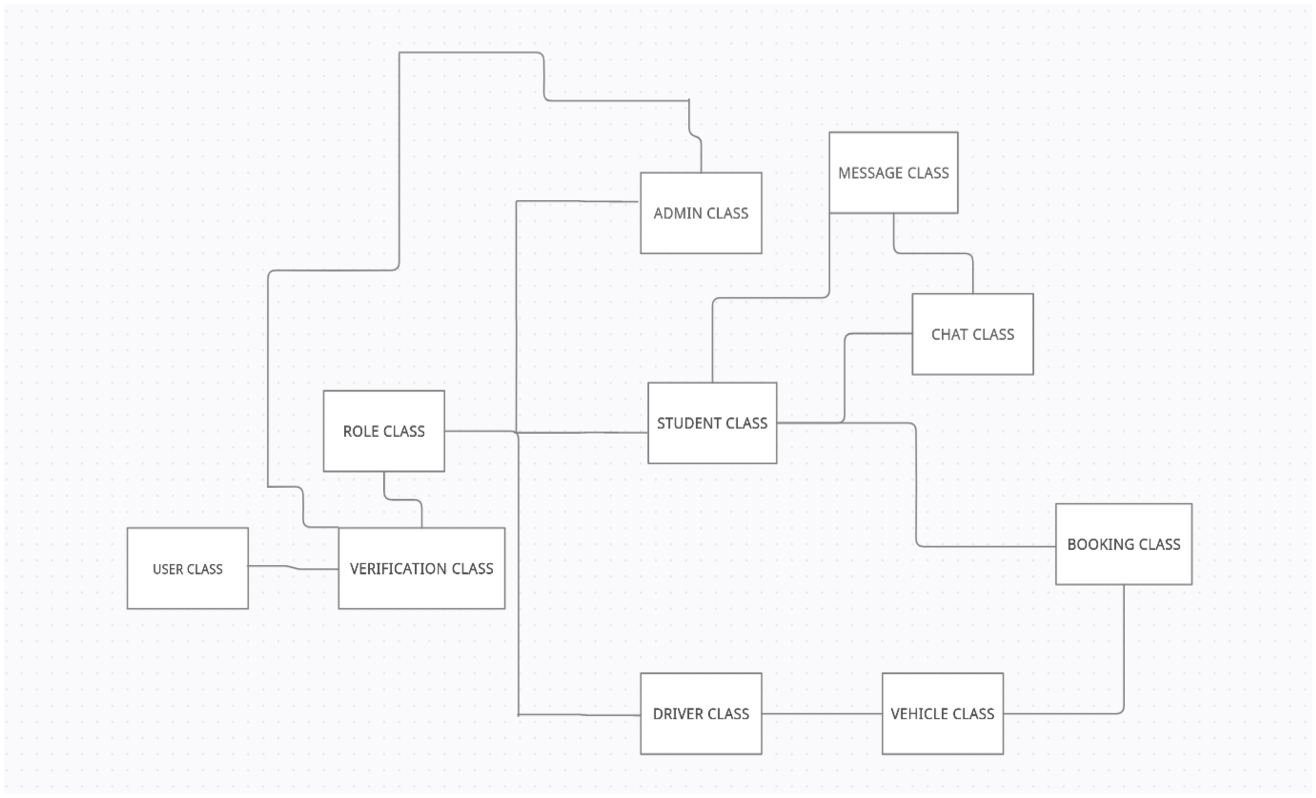
### 3.2.9 Message Class

| **Key** | **Data Type** | **Description** |
|---|---|---|
| message_id | int | Id of specific massage |
| sender_id | int | Id of sender |

| | | |
|---|---|---|
| message_content | string | massage content |

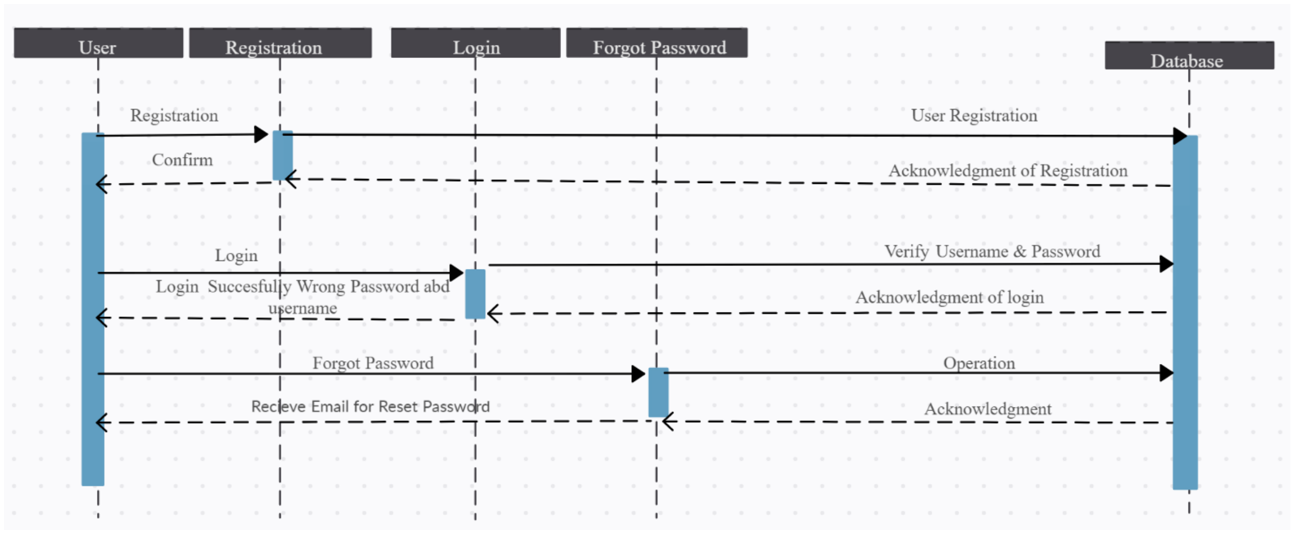
3.2.10 Booking  Class

| **Key** | **Data Type** | **Description** |
|---|---|---|
| booking_id | int | Specify a booking |
| booking_title | string | type of booking |
| booking_date | date | Date of booking |
| booking_date_work | date | Date of cab |
| booking_start_point | string | start location of journey |
| booking_end_point | string | end location of journey |
| booking_time | int [ ] | Time to book |
| add_booking( ) | module | add a booking |
| delete_booking( ) | module | delete a booking |
| search_booking( ) | module | search a booking |

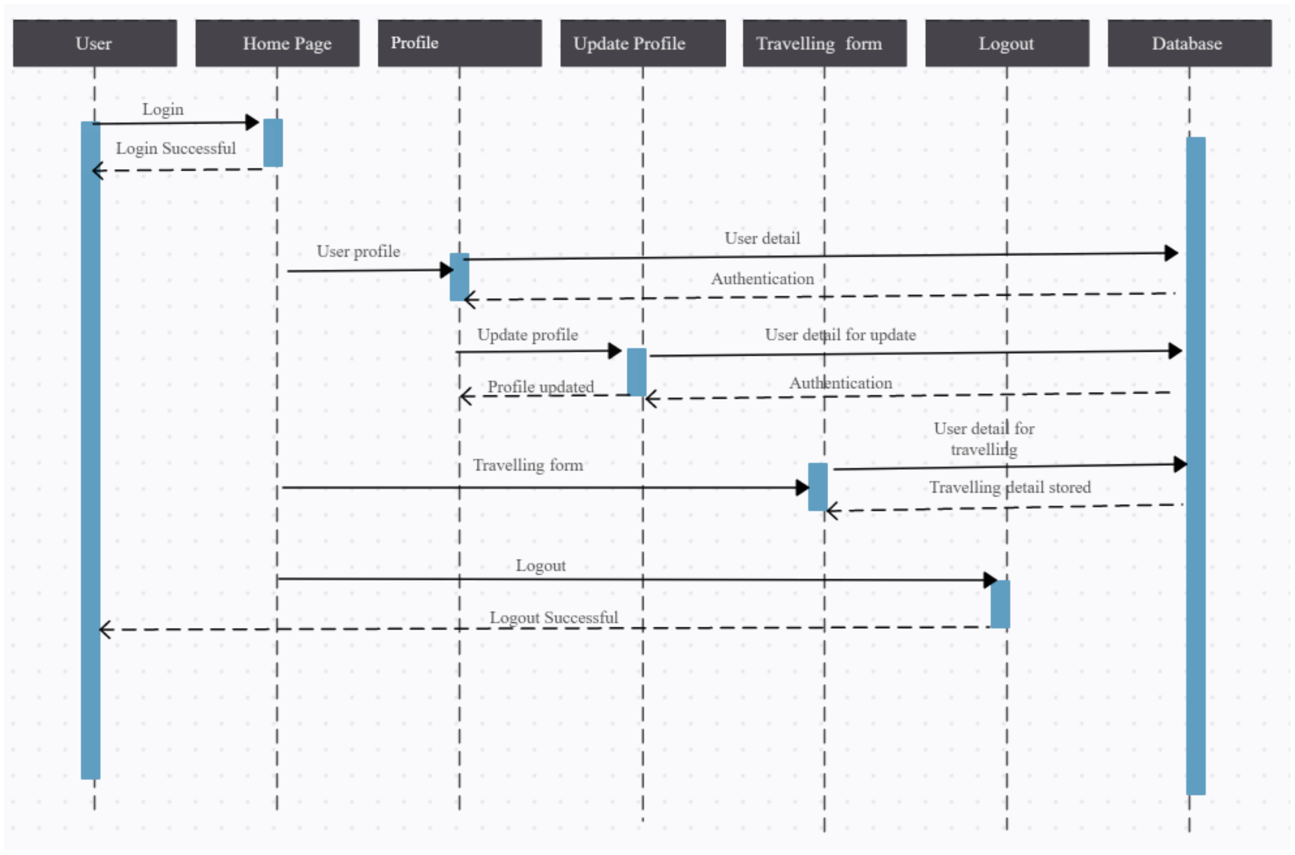


## 3.3 Sequence Diagrams

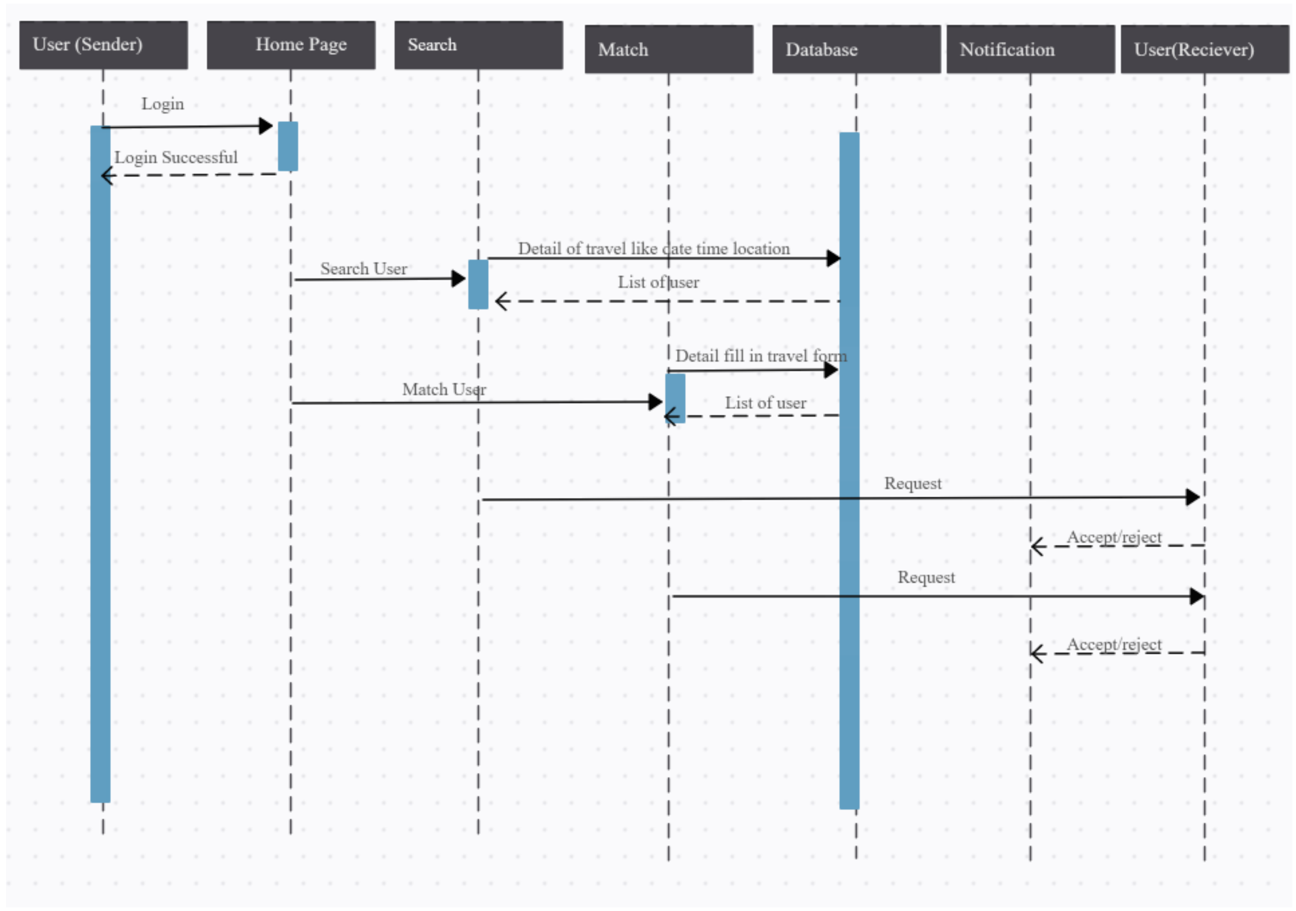
### User Login



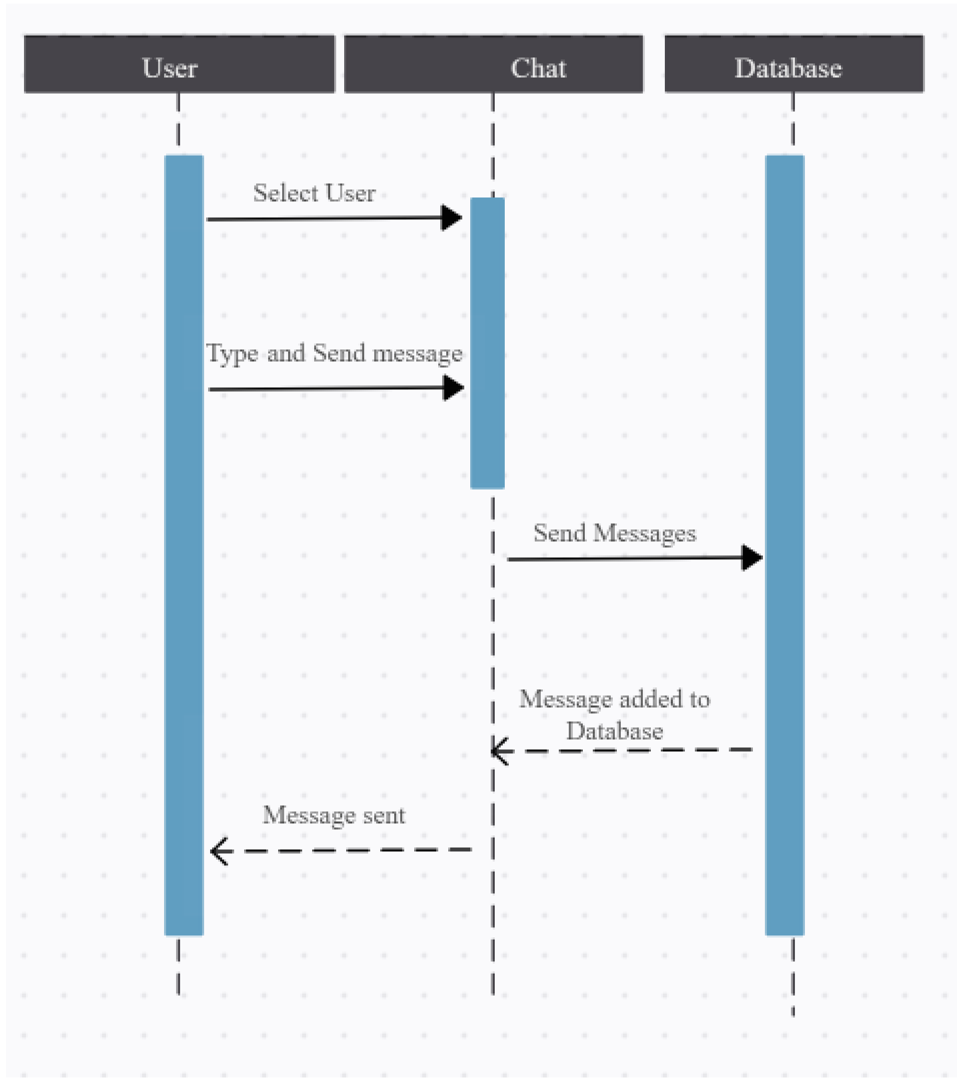
## Filling Detail and Form Submission



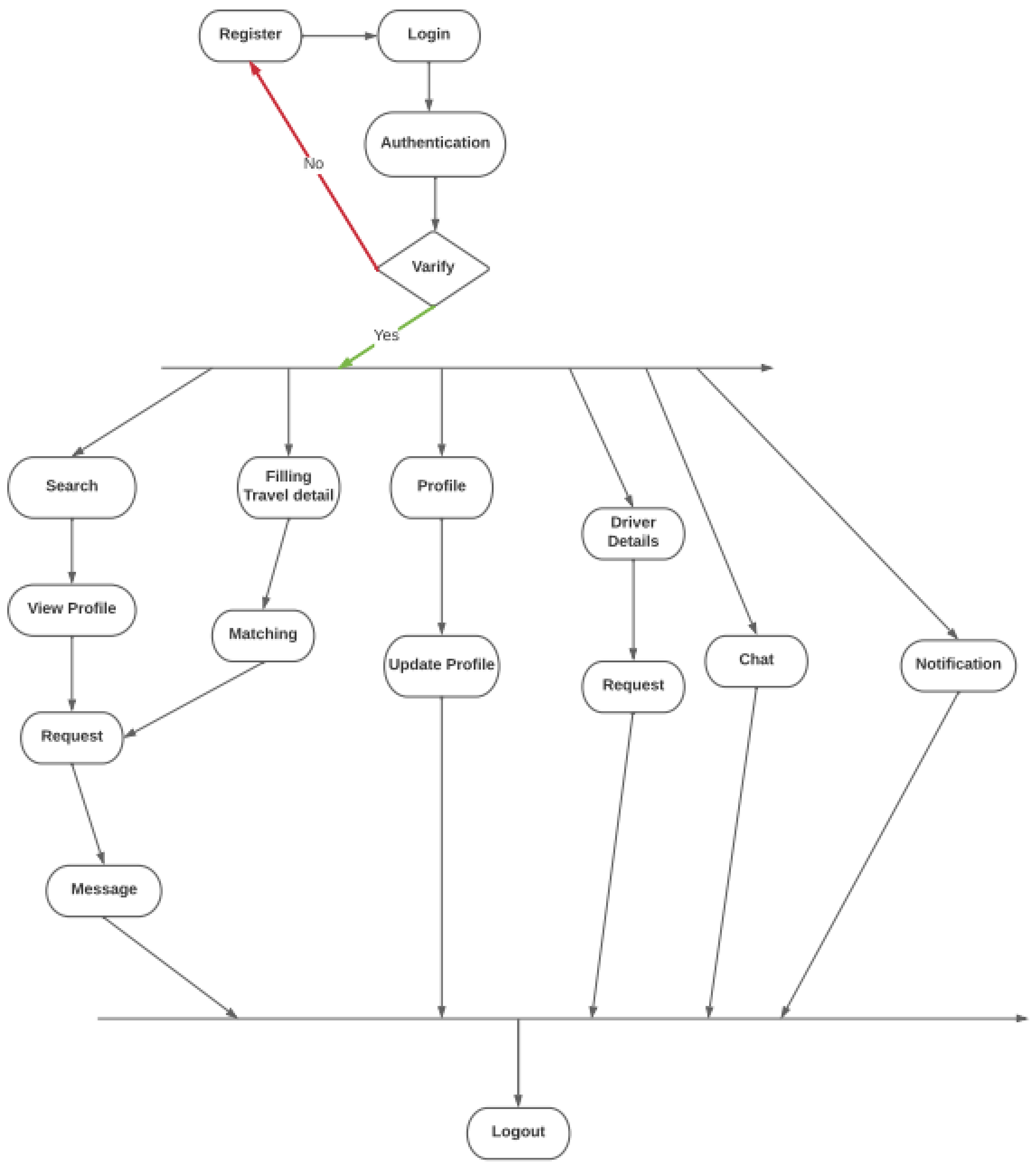
## Search Match and Request


User (Sender)
Home Page
Search
Match
Database
Notification
User(Reciever)
Login
Login Successful
Detail of travel like date time location
Search User
List of user
Detail fill in travel form
Match User
List of user
Request
Accept/reject
Request
Accept/reject

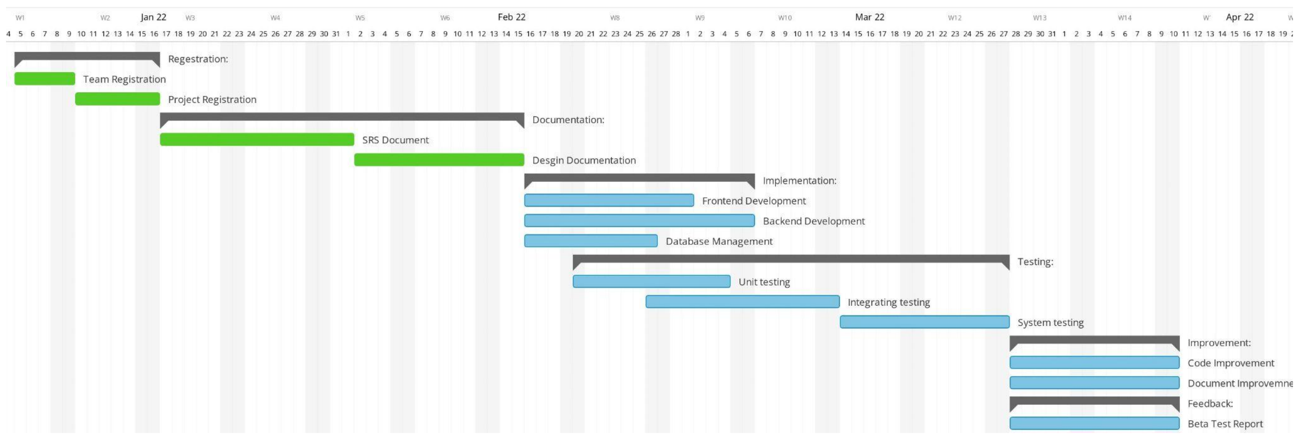
## Chats


User
Chat
Database
Select User
Type and Send message
Send Messages
Message added to Database
Message sent

## 3.4 State Diagrams



# 4 Project Plan



Frontend Development

| | Assigned to |
|---|---|
| Login | Akash Kumar Bhoi, Ujjawal Goyal, Abhishek Gurjar |
| Registration | Gopal Aggarwal, Hariom Shakyawal |
| Dashboard(Campus community) | Hariom Shakyawal, Gopal Aggarwal,Sourav Anand |
| Dashboard(Driver) | Rishabh Mukati, Aakash Lawa |
| Search Interface | Rishabh Muakti, Gopal Aggarwal |
| Notification Interface | Gopal Aggarwal, Hariom Shakyawal |
| Driver Detail Interface | Rishabh Mukati, Sourav Anand, Sourabh Meena |
| Chat Interface | Gopal Aggarwal, Ujjawal Goyal, Sourabh Meena |
| Travelling Detail Interface | Abhishek Gurjar, Hariom Shakyawal |

| | |
|---|---|
| Profile Interface | Hariom Shakyawal, Gopal Aggarwal |
| Match Interface | Rishabh Mukati,Sourabh Meena |

Backend Development

| | Assigned to |
|---|---|
| User Class | Akash Kumar Bhoi, Aakash Lawa, Sourav Anand |
| Role Class | Abhishek Gurjar, Prince Ahirwar |
| Verification Class | Hariom Shakyawal, Rishab Mukati, Sourabh Meena |
| Student Class | Ujjawal Goyal, Abhishek Gurjar, Hariom Shakyawal |
| Driver Class | Prince Ahirwar, Sourav Anand, Abhishek Gurjar |
| Admin Class | Rishab Mukati, Aakash Lawa, Akash Bhoi |
| Vehicle Class | Gopal Aggarwal, Hariom Shakyawal, Sourabh Meena |
| Chat Class | Gopal Aggarwal,Rishabh Mukati, Akash Bhoi |
| Message Class | Rishabh Mukati.Abhishek Gurjar |
| Booking Class | Ujjawal Goyal,Sourabh Meena, Sourav Anand |
| Other | Akash Kumar Bhoi, Prince Ahirwar, Abhishek Gurjar |

Database management

| | Assigned to |
|---|---|

| | |
|---|---|
| Database for User | Hariom Shakyawal, Ujjawal Goyal, Sourav Anand |
| Database For chat | Ujjawal Goyal, Rishab Mukati, Sourabh Meena |
| Database for Travelling detail | Akash Bhoi, Aakash Lawa, Abhishek Gurjar |
| Other database | Abhishek Gurjar, Prince Ahirwar |

Testing

| | Assigned to |
|---|---|
| Unit testing | Hariom Shakyawal, Sourabh Mina, Abhishek Gurjar |
| Integrated Testing | Rishabh Mukati, Sourav Anand, Aakash Lawa |
| Beta testing | Entire Team |
| Non - Functional Testing | |

# 5 Other Requirements

<*This section is Optional. Please provide any other details that are suitable for being included in the design document .*>

## Appendix A - Group Log

<*Please include here all the minutes from your group meetings, your group activities, and any other relevant information that will assist in determining the effort put forth to produce this document*>

| DATE | TIME | Discussion |
|---|---|---|
| 10th February | 4.00 pm - 4.20 pm | Tell every member to read the Design document and decide to meet on 11th Feb at 4.00 pm. |
| 11th February | 6.00 pm - 6.40 pm | In zoom meet, discuss Design among us, clear some doubts, and discuss unclear doubts with TA on discord. Also divide work between us. |
| 12th February | 1.00 pm - 1.45 pm | Evaluate work among us and help each other where anybody is stuck. |
| 13th February | 4.00 pm - 5.10 pm | Discussion on zoom meetings and getting feedback from each other and discussing class diagrams. |
| 14th February | 1.00 pm - 1.30 pm | Evaluate work among us and help each other where anybody is stuck. |
| 15th February | 1.00 pm - 1.45 pm | Finalise document and make log entry, fix minor error |
| 15th February | 8.00 pm -8.30 pm | Document reviewed by TA and make changes as he suggests |