By: Kevin Dawson

Supervisor: Joseph Kehoe



Design Document

Abstract

The purpose of this document is to provide the design specification for the project, this includes the system architecture, database design, detailed use cases and UI design.

Table of Contents

[System Architecture 4](#_Toc468133951)

[Database Design 5](#_Toc468133952)

[Table design 5](#_Toc468133953)

[User 5](#_Toc468133954)

[SQL command 6](#_Toc468133955)

[UserRating 6](#_Toc468133956)

[SQL command 6](#_Toc468133957)

[Driver 7](#_Toc468133958)

[SQL command 7](#_Toc468133959)

[Passenger 7](#_Toc468133960)

[SQL command 7](#_Toc468133961)

[Experience 8](#_Toc468133962)

[SQL command 8](#_Toc468133963)

[CarDetails 8](#_Toc468133964)

[SQL command 9](#_Toc468133965)

[Lift 9](#_Toc468133966)

[SQL command 9](#_Toc468133967)

[CompletedLifts 10](#_Toc468133968)

[SQL command 10](#_Toc468133969)

[Use Cases 11](#_Toc468133970)

[Use Case Diagram 11](#_Toc468133971)

[Detailed Use Cases 11](#_Toc468133972)

[Register Use Case 11](#_Toc468133973)

[System Sequence Diagram 13](#_Toc468133974)

[Login Use Case 14](#_Toc468133975)

[System Sequence Diagram 15](#_Toc468133976)

[Delete Account Use Case 15](#_Toc468133977)

[System Sequence Diagram 16](#_Toc468133978)

[Offer Lift Use Case 16](#_Toc468133979)

[System Sequence Diagram 17](#_Toc468133980)

[Accept/Deny Lift Request Use Case 17](#_Toc468133981)

[System Sequence Diagram 18](#_Toc468133982)

[Search for Lift Use Case 18](#_Toc468133983)

[System Sequence Diagram 19](#_Toc468133984)

[Request Lift Use Case 19](#_Toc468133985)

[System Sequence Diagram 20](#_Toc468133986)

[Completed Lift Use Case 20](#_Toc468133987)

[System Sequence Diagram 21](#_Toc468133988)

[Rate User Use Case 21](#_Toc468133989)

[System Sequence Diagram 22](#_Toc468133990)

[Design 22](#_Toc468133991)

[UI Screenshots 23](#_Toc468133992)

[Landing Page 23](#_Toc468133993)

[Registration Page 23](#_Toc468133994)

[Car Details Page 24](#_Toc468133995)

[Main Page 25](#_Toc468133996)

[Available Lifts Page 25](#_Toc468133997)

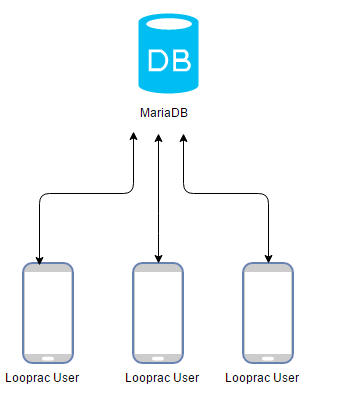
[Request Lift Page 26](#_Toc468133998)

[Driver Details Page 27](#_Toc468133999)

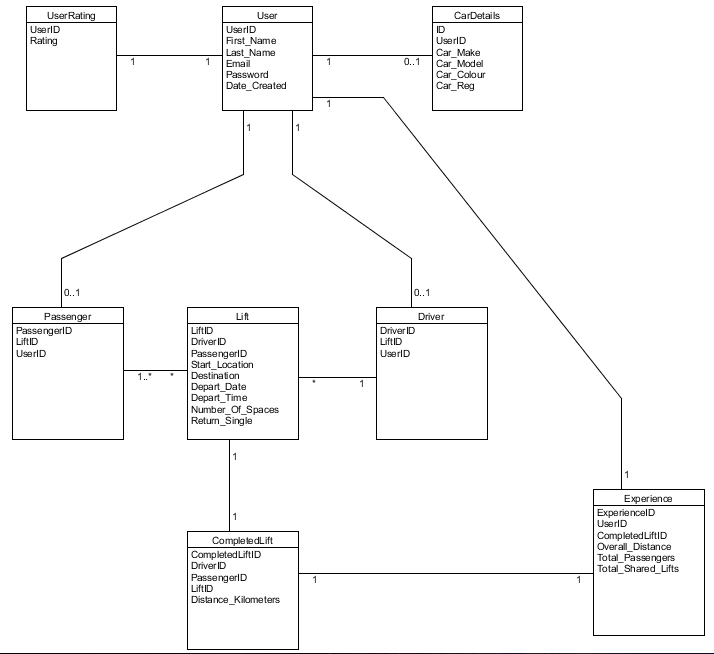
[Offer Lift Page 28](#_Toc468134000)

[UI Flow 28](#_Toc468134001)

# System Architecture



# Database Design



The database chosen for Looprac is MySQL. Below are the tables that will be created in the database along with the SQL commands to add data to them.

## Table design

### User

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| UserID | First\_Name | Last\_Name | Email | Password | Date\_Created |
| 100 | John | Sample | John.sample@email.com | samplePassword | 2016-11-15 |

#### SQL command

Table creation:

CREATE TABLE User

(

UserID INT NOT NULL AUTO\_INCREMENT,

First\_Name VARCHAR(40) NOT NULL,

Last\_Name VARCHAR(40) NOT NULL,

Email VARCHAR(20) NOT NULL,

Password VARCHAR(15),

Date\_Created DATE,

PRIMARY KEY (UserID)

);

Adding data to the table

INSERT into User(First\_Name, Last\_Name, Email, Password, Date\_Created)

VALUES

(‘John’, ‘Sample’, ‘john.sample@email.com’, ‘samplePassword’, ‘2016-04-07’);

### UserRating

|  |  |  |
| --- | --- | --- |
| ID | UserID | Rating |
| 250 | 100 | 1.2 |

#### SQL command

Table creation:

CREATE TABLE UserRating

(

ID INT NOT NULL AUTO\_INCREMENT,

UserID INT NOT NULL,

CONSTRAINT fk\_User

FOREIGN KEY(UserID)

REFERENCES User(UserID),

Rating DECIMAL(3,2) DEFAULT 0.00,

PRIMARY KEY (ID)

);

Adding data to the table

INSERT into UserRating(UserID, Rating)

VALUES

(‘100’,’1.2’);

### Driver

|  |  |  |
| --- | --- | --- |
| DriverID | UserID | LiftID |
| 550 | 100 | 200 |

#### SQL command

Table creation:

CREATE TABLE Driver

(

DriverID INT NOT NULL AUTO\_INCREMENT,

UserID INT NOT NULL,

CONSTRAINT fk\_User\_Driver

FOREIGN KEY(UserID)

REFERENCES User(UserID),

LiftID INT,

CONSTRAINT fk\_Lift\_Driver

FOREIGN KEY(LiftID) REFERENCES Lift(LiftID),

PRIMARY KEY (DriverID)

);

Adding data to the table

INSERT into Driver(UserID)

VALUES

(‘100’);

### Passenger

|  |  |  |
| --- | --- | --- |
| PassengerID | UserID | LiftID |
| 600 | 100 | 200 |

#### SQL command

Table creation:

CREATE TABLE Passenger

(

PassengerID INT NOT NULL AUTO\_INCREMENT,

UserID INT NOT NULL,

CONSTRAINT fk\_User\_Passenger

FOREIGN KEY(UserID)

REFERENCES User(UserID),

LiftID INT,

CONSTRAINT fk\_Lift\_Passenger

FOREIGN KEY(LiftID)

REFERENCES Lift(LiftID),

PRIMARY KEY (PassengerID)

);

Adding data to the table

INSERT into Passenger(UserID)

VALUES

(‘100’);

### Experience

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ExperienceID | UserID | CompletedLiftID | Overall\_Distance\_kilo | Total\_Passengers | Total\_Shared\_Lifts |
| 325 | 100 | 300 | 240.5 | 15 | 5 |

#### SQL command

Table creation:

CREATE TABLE Experience

(

ExperienceID INT NOT NULL AUTO\_INCREMENT,

UserID INT NOT NULL,

CONSTRAINT fk\_User\_Experience

FOREIGN KEY(UserID)

REFERENCES User(UserID),

CompletedLiftID INT NOT NULL,

CONSTRAINT fk\_CompletedLift\_Experience

FOREIGN KEY(CompletedLiftID)

REFERENCES CompletedLifts(CompletedLiftID),

Overall\_Distance\_kilo DECIMAL(5,1),

Total\_Passengers INT,

Total\_Shared\_Lifts INT,

PRIMARY KEY (ExperienceID)

);

Adding data to the table

INSERT into Experience(UserID,CompletedLiftID, Overall\_Distance\_kilo, Total\_Passengers, Total\_Shared\_Lifts)

VALUES

(‘100’, ‘300’, ’240.5’, ’15’, ’5’);

### CarDetails

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ID | UserID | Car\_Make | Car\_Model | Car\_Colour | Car\_Reg |
| 400 | 100 | Ford | Mondeo | Black | 01-MH-12345 |

#### SQL command

Table creation:

CREATE TABLE CarDetails

(

ID INT NOT NULL AUTO\_INCREMENT,

UserID INT NOT NULL,

CONSTRAINT fk\_User\_CarDetails

FOREIGN KEY(UserID)

REFERENCES User(UserID),

Car\_Make VARCHAR(20) NOT NULL,

Car\_Model VARCHAR(20) NOT NULL,

Car\_Colour VARCHAR(10),

Car\_Reg VARCHAR(20) NOT NULL,

PRIMARY KEY (ID)

);

Adding data to the table

INSERT into CarDetails(UserID,Car\_Make, Car\_Model, Car\_Colour, Car\_Reg)

VALUES

(‘100’, ‘Ford’, ‘Mondeo’, ‘Black’, ’01-MH-12345’);

### Lift

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| LiftID | UserID | Start\_Location | Destination | Depart\_Date | Depart\_Time | Available\_Spaces | Return\_Single | Created\_At |
| 200 | 100 | Carlow | Dublin | 2016-11-25 | 09:00 | 3 | return | 2016-11-25 |

#### SQL command

Table creation:

CREATE TABLE Lift

(

LiftID INT NOT NULL AUTO\_INCREMENT,

UserID INT NOT NULL,

CONSTRAINT fk\_User\_Lift

FOREIGN KEY(UserID)

REFERENCES User(UserID),

Start\_Location VARCHAR(50) NOT NULL,

Destination VARCHAR(50) NOT NULL,

Depart\_Date DATE NOT NULL,

Depart\_Time TIME NOT NULL,

Available\_Spaces INT,

Return\_Single VARCHAR(6),

Created\_At DATETIME,

PRIMARY KEY (LiftID)

);

Adding data to the table

INSERT into Lift(UserID, Start\_Location, Destination, Depart\_Date, Depart\_Time, Available\_Spaces, Return\_Single, Created\_At)

VALUES

(‘100’, ‘Carlow’, ‘Dublin’, ‘2016-11-25’, ’09:00’, ‘3’, ‘return’, ‘2016-11-25’);

### CompletedLifts

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CompletedLiftID | LiftID | DriverID | PassengerID | Distance\_kilometers |
| 300 | 200 | 100 | 400 | 73.1 |
| 300 | 200 | 100 | 500 | 73.1 |
| 300 | 200 | 100 | 600 | 73.1 |

#### SQL command

Table creation:

CREATE TABLE CompletedLifts

(

CompletedLiftID INT NOT NULL AUTO\_INCREMENT,

LiftID INT NOT NULL,

CONSTRAINT fk\_Lift\_CompletedLifts

FOREIGN KEY(LiftID)

REFERENCES Lift(LiftID),

DriverID INT NOT NULL,

CONSTRAINT fk\_Driver\_CompletedLifts

FOREIGN KEY(DriverID)

REFERENCES Driver(DriverID),

PassengerID INT NOT NULL,

CONSTRAINT fk\_Passenger\_CompletedLifts

FOREIGN KEY(PassengerID)

REFERENCES Passenger(PassengerID),

Distance\_Kilometers DECIMAL(5,1),

PRIMARY KEY (CompletedLiftID)

);

Adding data to the table

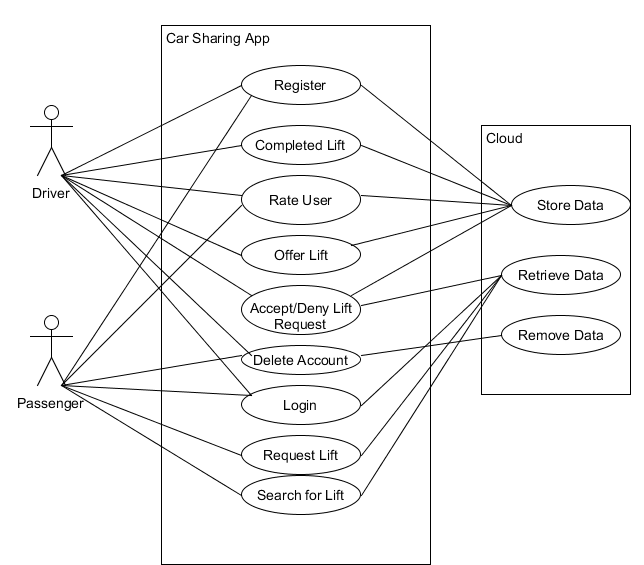
INSERT into CompletedLifts( LiftID, DriverID, PassengerID, Distance\_Kilometers)

VALUES

(‘200’, ‘100’, ‘400’, ’73.1’);

# Use Cases

## Use Case Diagram



## Detailed Use Cases

This section will look at the brief use cases from the functional specification document and turn them into detailed use cases.

### Register Use Case

**Actors**: user, cloud

**Brief description**: This use case begins when a user wishes to register with the app. The user enters their details (name, email, car details (if they want to be drivers)). The app validates the entered information and stores it to the cloud. The use case ends when the user’s details are registered with the app.

**Main success scenario:**

1. The user starts up the app
2. The app offers them choice to login or register
3. The user selects register
4. The app displays register page with detail inputs
5. The user enters details – name, email address, password
6. The user skips car details
7. The app validates the inputted details
8. The app sends the data to be stored in the cloud
9. The app presents user with acknowledgment that they are registered

**Alternatives**

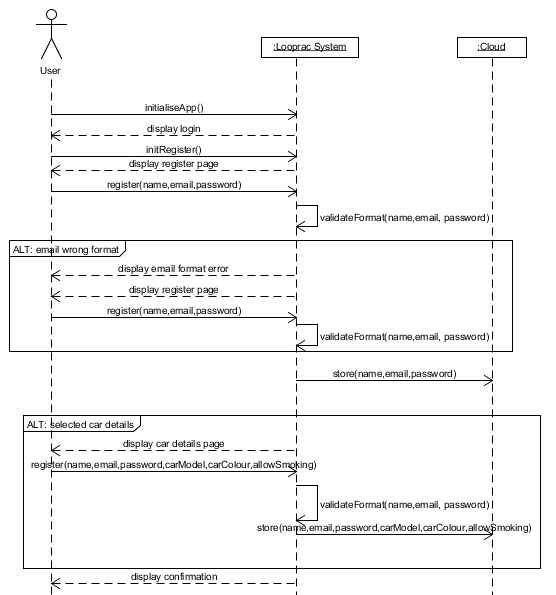
5A. The user selects car details

1. User enters car details – car model, car colour, allow smokers or not

6A. Details entered are of incorrect format

1. App alerts user that inputted details were of incorrect format
2. The app displays register page with detail inputs

#### System Sequence Diagram



### Login Use Case

**Actors:** user, cloud

**Brief description:** This use case begins when a user wants to login to the app. The user enters their email address and password. The app validates the information and queries the cloud. This use case ends when the information is validated and grants the user access to the app.

**Main success scenario:**

1. The user starts up the app
2. The app displays login page
3. The user submits their email address and password
4. The app validates the email format is correct
5. The app queries the cloud with the entered email address
6. It verifies that the right password was entered
7. The cloud returns a session token
8. The app grants the user access to the app

**Alternatives**

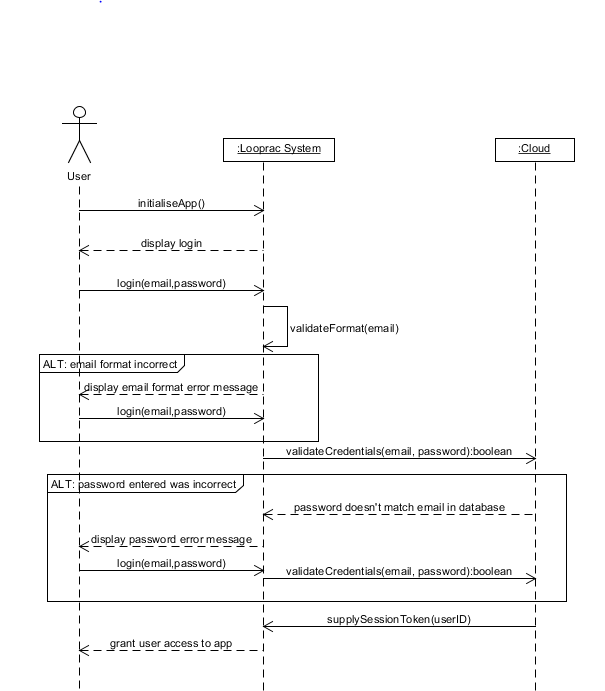
5A. The email format entered is incorrect

1. The app alerts the user that the email was incorrect
2. The app prompts the user to recheck their email

7A. The password entered was incorrect

1. The app alerts the user that the password was incorrect
2. The app prompts the user to enter password

#### System Sequence Diagram



### Delete Account Use Case

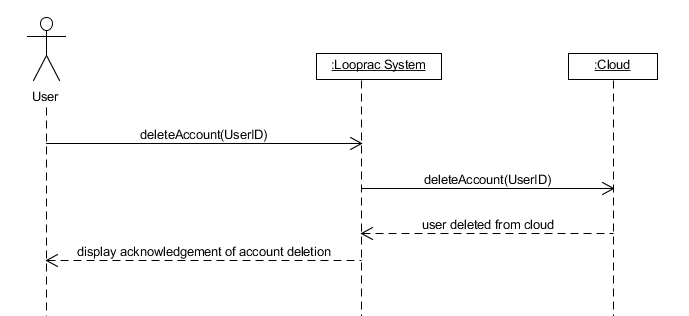
**Actors:** User, cloud

**Brief description:**  This use case begins when a user wishes to delete their account. The user will select ‘delete account’ option. The app will verify that they want to continue. The user selects ‘yes’. The app updates the cloud by removing the user’s data from the cloud. This use case ends when the app provides the user with an acknowledgement that the task is complete.

**Main success scenario:**

1. The user selects delete account option
2. The app alerts the user of the choice they are making and asks them if they are sure
3. The user selects yes
4. The app queries the cloud with the user’s unique ID
5. The cloud updates itself by removing the account associated with the unique ID
6. The app displays an acknowledgement to the user that their account was deleted

#### System Sequence Diagram



### Offer Lift Use Case

**Actors:** driver, cloud

**Brief description:** This use case begins when a driver wishes to offer a lift in their vehicle. The app will ask the driver for their location, destination, the depart date, depart time, how many spaces are available in the car and is it a single or return trip. The app validates this information and sends it to the cloud. This use case ends when the details are validated and an acknowledgment is shown to the driver.

**Main success scenario:**

1. The driver selects to offer a lift
2. The app prompts the driver for details regarding the lift – location, destination, depart date, depart time, how many spaces are available and whether it is a single or return trip
3. The app validates that the entered information is valid format
4. The app sends the information to the cloud
5. The app presents the driver with acknowledgement that the lift offer was accepted

**Alternatives:**

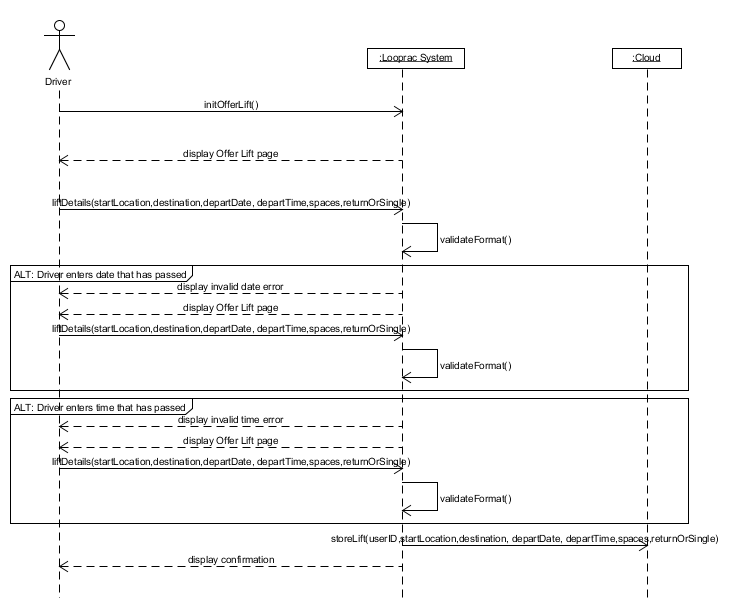
3A. Driver enters date that has passed

1. App alerts driver that the entered date is not valid
2. App prompts driver to re-enter a valid date

3B. Driver enters time that has passed

1. App alerts driver that the entered time is not valid
2. App prompts driver to re-enter a valid time

#### System Sequence Diagram



### Accept/Deny Lift Request Use Case

**Actors:** driver, passenger, cloud

**Brief description:** this use case begins when a driver gets a notification that a lift request is waiting for their approval or denial. The driver will select the lift request. The app will display details to which lift that the request is for that the driver has previously setup. The app will display the potential passengers name and rating to the driver. The driver can accept the request or deny the request. This use case ends when the driver accepts or denies the request.

**Main success scenario:**

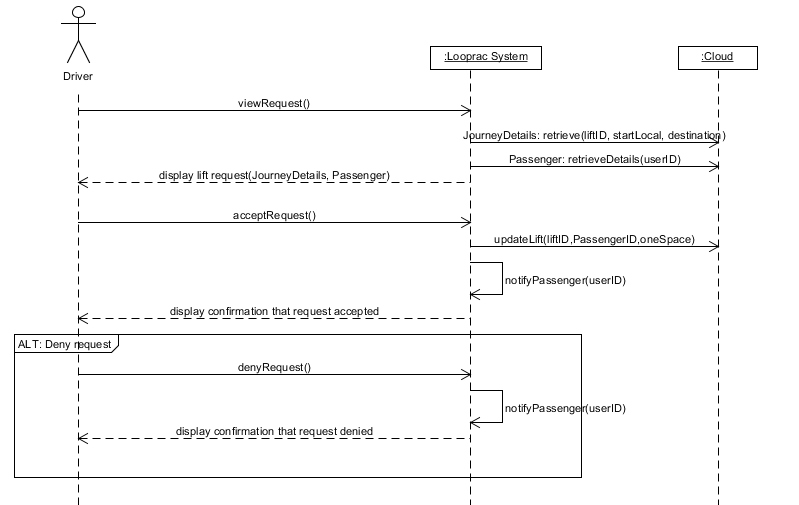
1. The app notifies the driver that a passenger has requested a space in their vehicle
2. The driver selects the request
3. The app displays information about the request – what lift offer it is connected to (location, destination, time) and the requested passengers name and rating
4. The driver selects to accept the request
5. The app displays an acknowledgement to the driver of their choice
6. The app updates the lift record by adding the passenger and updating spaces available
7. The app sends a notification to the passenger that their request was approved with details about the lift and the car including make, model, colour and registration

**Alternatives:**

4A. The driver denies the request

1. The app displays an acknowledgement to the driver of their choice
2. The app notifies the passenger that their request was denied

#### System Sequence Diagram



### Search for Lift Use Case

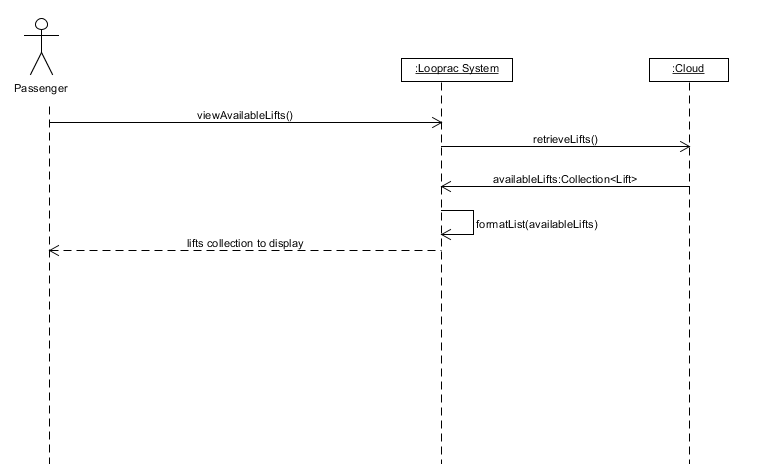
**Actors:** passenger, cloud

**Brief description:** this use case begins when a passenger wants to search for available lifts. The passenger selects the option ‘available lifts’. The app queries the cloud and retrieves and populates the list with available lifts. The app displays a short description of the lift in the list. This use case ends when the app displays the available lifts to the passenger.

**Main success scenario:**

1. The passenger selects the available lifts option
2. The app queries the cloud and updates the list with the available lifts
3. The app displays the list of lifts with the location, destination and time added

#### System Sequence Diagram



### Request Lift Use Case

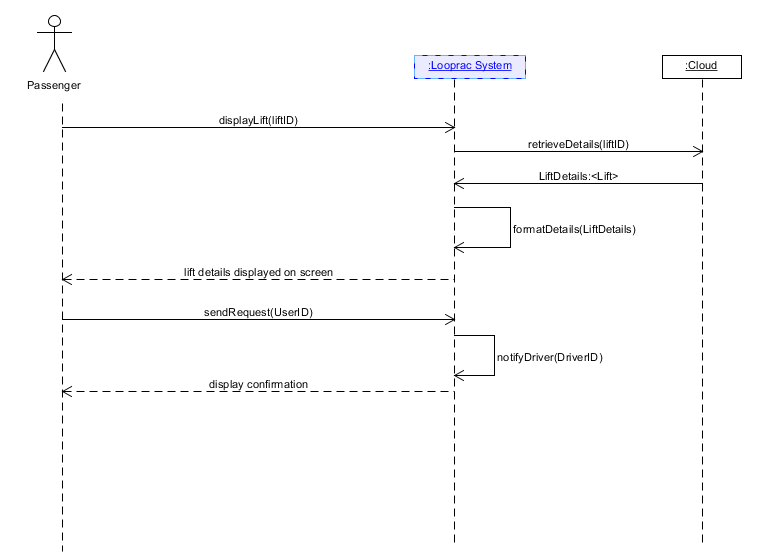
**Actors:** passenger, cloud

**Brief description:** this use case begins when a passenger has chosen a lift and they wish to request a space in the vehicle. The passenger will select the lift from the list. The app will query the cloud and display details of the lift and the driver. The passenger will select the request lift option. This use case ends when the app gives the passenger acknowledgement that the request has been sent.

**Main success scenario:**

1. The passenger selects the lift from the list of available lifts
2. The app queries the cloud for information on the lift and driver
3. The app displays to the passenger – location of lift and destination on a map, the time the lift will be leaving at, spaces available in the vehicle, whether it’s a single or return trip and the drivers rating
4. The passenger selects the request lift option
5. The app provides the passenger with acknowledgment that the request has been sent

#### System Sequence Diagram



### Completed Lift Use Case

**Actors:** user, cloud

**Brief description:** this use case begins when the user selects lift complete option when they reach the destination. The app will display a completed option for the user throughout the journey. The user will select the completed option. The app will store the completed journey to the cloud. This use case ends when the app informs the user with a message showing them they’re new statistics.

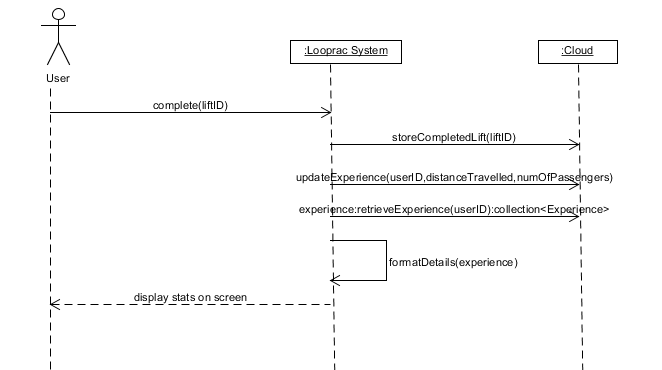
**Main success scenario:**

1. The app displays a ‘complete’ option
2. The user selects the ‘complete’ option when they arrive at the destination
3. The app stores the completed journey to the cloud along with distance the user travelled
4. The app retrieves from the cloud the total distance that the user has from completed journeys
5. The app displays a message to the user informing them of the distance driven on this journey and their overall distance driven from other journeys.

**Alternatives**

5A. If the user is the driver a message informs them of the distance driven on this journey, their overall distance driven, and number of passengers altogether.

#### System Sequence Diagram



### Rate User Use Case

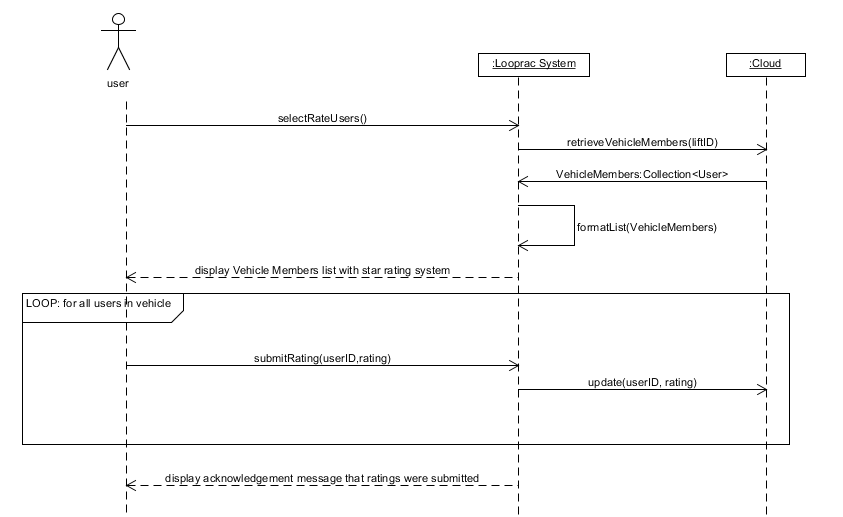
**Actors:** user, cloud

**Brief description:** this use case begins when the user is prompted to rate the driver/passenger(s) when the lift has been completed. The app will display a five-star system and allow the user to select between zero stars to five stars. The user submits their rating. The app updates the driver and passenger’s rating in the cloud. This use case ends when the app displays an acknowledgement that the rating was submitted.

**Main success scenario:**

1. The app prompts the user to rate other users in the vehicle
2. The app retrieves other users sharing the lift
3. The app displays a screen with five empty stars and informs the user to select a rating out of five for each user
4. The user selects a rating and selects submit
5. The app updates the driver and passenger(s) ratings in the cloud
6. The app displays an acknowledgement to the user that the rating was successfully submitted.

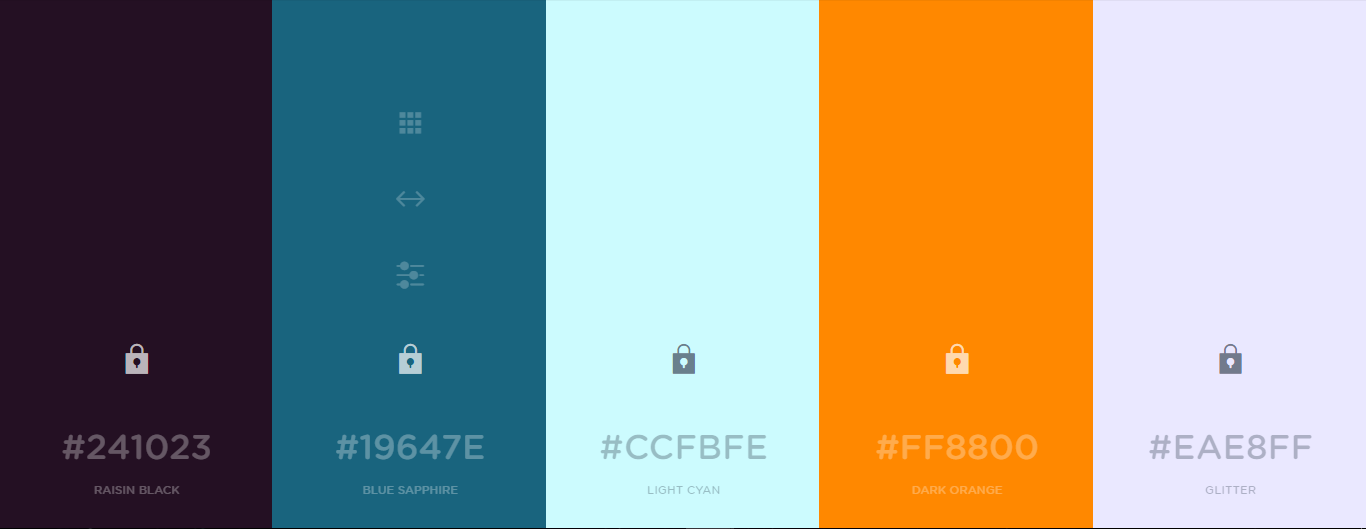
#### System Sequence Diagram



# Design

The design of the app will be a simplistic and minimalistic interface aiming at a positive user experience. This involves an easy and fluid flow of navigation through the app. Below are some potential UI mock ups for certain pages. These few screenshots are to give an idea of the UI design and colour palette ideas.

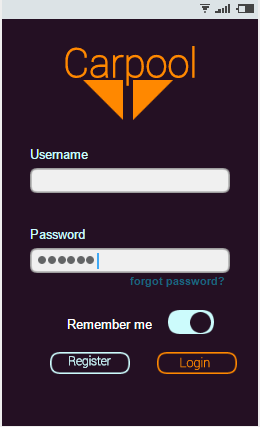
Below is the primary colour palette chosen for the UI of Looprac.



## UI Screenshots

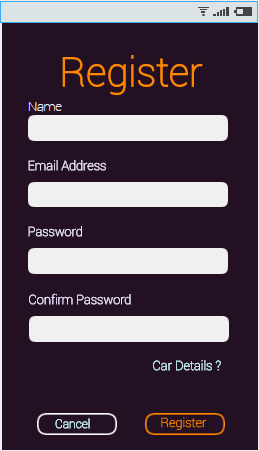
Below are mock ups for what a potential look that the app may incorporate. These mock ups may be accurate in some cases and only guidelines in others.

### Landing Page

The landing page is the very first page that the users will see. Below are mock ups.

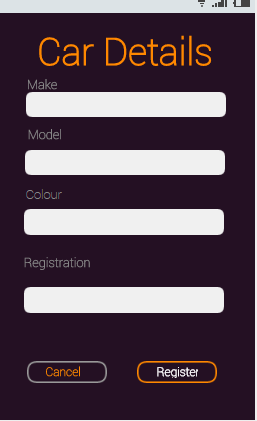
### Registration Page

This is the page where users will register for the app. It requires little information so that the process is quick. There is an optional link to register a car or this can be done at another time within the app.



### Car Details Page

This page can be accessed when the user first registers with Looprac or can be found in the options if they wish to complete it another time or update the details.

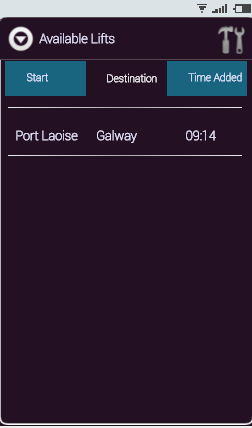


### Main Page

This is the main page of the app. It will be the page that the users will be brought to when they log in for the first time, and thereafter. Potentially, the map will be populated with available lifts in their area. Alternatively, they can select to find lift which will bring them to a list of available lifts. They can also offer a lift from this page.

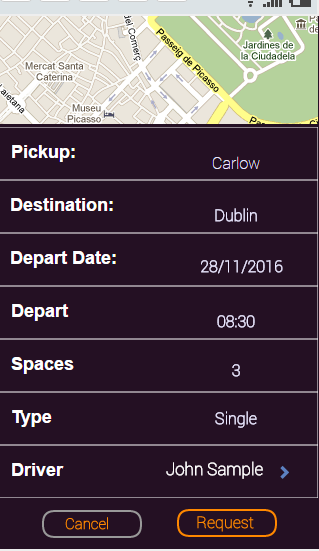
### Available Lifts Page

This page presents the user with a list of available. It displays available lifts start point, destination and time it was added. The user can also select different ways to sort the list.



### Request Lift Page

This page presents the user with details about the lift that they are interested in. It displays the route at the top which they can expand by selecting it. It also provides them with details about the lift and the drivers name.



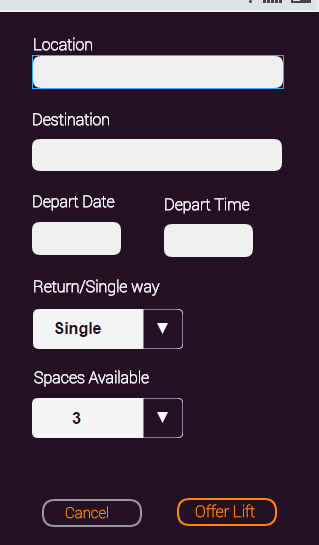
### Driver Details Page

This page presents the user with some details about the driver that is offering the lift. It includes their name, rating out of five and any badges that they may have earned

.

### Offer Lift Page

This page is presented to the user when they want to become a driver and offer a lift. They are presented with a page that asks for details regarding the lift which includes start location, destination, depart date, depart time, whether it is single or return and number of spaces they are offering in the car.



## UI Flow

Below is a picture of the flow of the main activities of the app.

