Parking Garage System

Group No. 26

Group ID: 6604

Team Members

Mohamed Nouh ID: 6355

Mohamed Bahaa ID: 6586

Nour Yasser ID: 6604

Table of Contents

Page title	Page No.
Cover Page…………………………………………………………………………	1
Table of Contents…………………………………………………………………	2
Requirements………………………………………………………………………	3
Software Process…………………………………………………………………	7
Architectural Design……………………………………………………………	8
System Modeling ………………………………………………………………..	11
Design Description…………………………………………………………….	15
Testing………………………………………………………………………………	16

1. REQUIREMENTS

Customer Requirements

Time is a person’s most valuable asset which has called for the digitalizing of many human activities whether daily or irregular. This project introduces a new regular activity into the digital world that has never been applied in Egypt.

The plan is to allow people to make a garage reservation before reaching their destination, this will be of great value especially due to our overcrowded streets and the fact that people are always in a hurry and would rather not deal with the hassle of finding a parking spot.

Moreover, the garage will offer services that you would not find in your regular garage, which people usually do in separate specialized places. For example, car washing whether interior or exterior, car polish and tire gauge. And of course they would be performed at the best qualities.

There would be several garage locations well spread all over Alexandria, as a starting phase, there would be 4 garages located in smouha, gleem, roushdy and Fouad Street.

The system end-users are the garage’s employees and customers. Customers should be allowed to choose between the garage locations, check availability then login if there are available spots and he plans to make a reservation. The website should request him to enter an entry time & an expected exit time to be able to identify the number of available spots at different times of the day. Any customer that makes an online reservation, must make an online payment to be sure the system would be secured from bots and intruders.

The garage employees that have access to the system are the managers and the cashiers. The cashier should be responsible of logging cars in and out. At log in the cashier should be able to confirm a reserved car’s entrance and register its

entrance time. And if an unreserved car arrives he should be able to log it in if there're available unreserved spots. At checkout, the cashier inputs the license number and the services provided. The system would be capable of calculating the total fees and amount to be paid or returned if there was a mismatch between the customer requests and the actual services provided.
In both the customer and cashier sides, appropriate validations should be applied to help customers and protect cashiers from making mistakes. For example, a customer cannot choose both overnight and temporary parking and he can't input an expected exit time that is before his entry time. And, a cashier cannot log a car in as registered when it is not.

The manager should have access to everything and act as the system supervisor. He would also be able to identify if a reserved customer doesn't make it so he can be able to return his money anytime.

System Requirements

Functional Requirements

1. The system shall be split into 3 different views – customer, cashier, manager – where each view has different screens.
2. The customer's interface shall allow the user to reserve a spot for his/her car at one of the available locations through a drop-down menu.
3. The customer's interface shall allow the user to select the services needed to be applied to the car while staying at the garage.
4. The customer's interface shall allow the user to choose between temporary or overnight parking while reserving.
5. The customer's interface shall have a login screen where frequent users log in using their credentials and new users have the option to sign up.

6. The system shall calculate at checkout if customer exceeded the expected exit time and subsequently charge extra fees on the credit card payment in case of an online reservation.
7. The cashier’s interface shall display cashier name as well as parking location.
8. The cashier’s interface shall allow the cashier to log a customer in and check if that customer already reserved before arriving.
9. The cashier’s interface shall allow the cashier to log out a customer.
10. The cashier’s interface shall check if services at logout align with the picked services at login and charge fees accordingly.
11. The system shall allow payment to be in cash in case of a non-reserving customer.
12. The manager’s interface shall allow him/her to have an overview of the databases of the system, by giving him/her access to view all reservations, cashiers, customers and garages.
13. Available spots in each garage shall be updated in runtime whenever a new customer logs in or out.

Non-functional Requirements

1. The system shall be easy to navigate by using large buttons, drop-down menus and large text fonts.
2. The system shall be highly responsive by making it lightweight to further speed up the loading and execution of functions of the system.
3. The system shall be secure by only allowing users access to their accounts after correct entry of username and password.

4. The system shall be viewable in different device sizes: mobiles, tablets and laptops.

<u>FURPS</u>

Functionality

- Features online payment for reservations which is another step towards the automation of the whole process.
- Features entry and expected exit time at reservation to ensure no location exceeds its capacity.

Usability

- Input boxes as well as drop-down menus are clearly labeled to further facilitate usage of website.
- The order of webpages from the customer's view ensures a swift reservation.
- All the pages for each of the 3 views have a similar interface so as not to confuse the user.

Reliability

- Checks are done at logout from the cashier's view to validate if services have been done as per reservation. Any extra charges by either ends will be calculated and paid then.

Performance

- The webpages are not highly sophisticated in an attempt to prevent too much processing which hinders the performance.

Supportability

- Website runs on many supported browsers.
- Website is split into multiple views which could be separately updated later on.

2. SOFTWARE PROCESS

Incremental Development of Reservation Function

The main function on the customer's side is reserving a spot for his/her car at one of the available locations. The customer passes over multiple screens of the web application until the reservation, with all its details, is confirmed. The screens were developed one after the other according to the order of tasks to be completed:

1. Branch Screen
    a. Includes a drop-down menu to allow the user to pick their desired location.
2. Services Screen
    a. Includes 2 types of parking either temporary or overnight.
    b. Includes 4 types of services to choose from, we initially started with 3 but decided to add a 4th service which is the Tire Gauge.
    c. Another feature that wasn't initially included but was added later on in the development phase was a display of the available spots within the next hour at the chosen branch.
3. Login Screen
    a. Includes 2 input text fields one for the username and another for the password.
    b. A signup link was also added to allow a new user to register into the system if not already a customer.

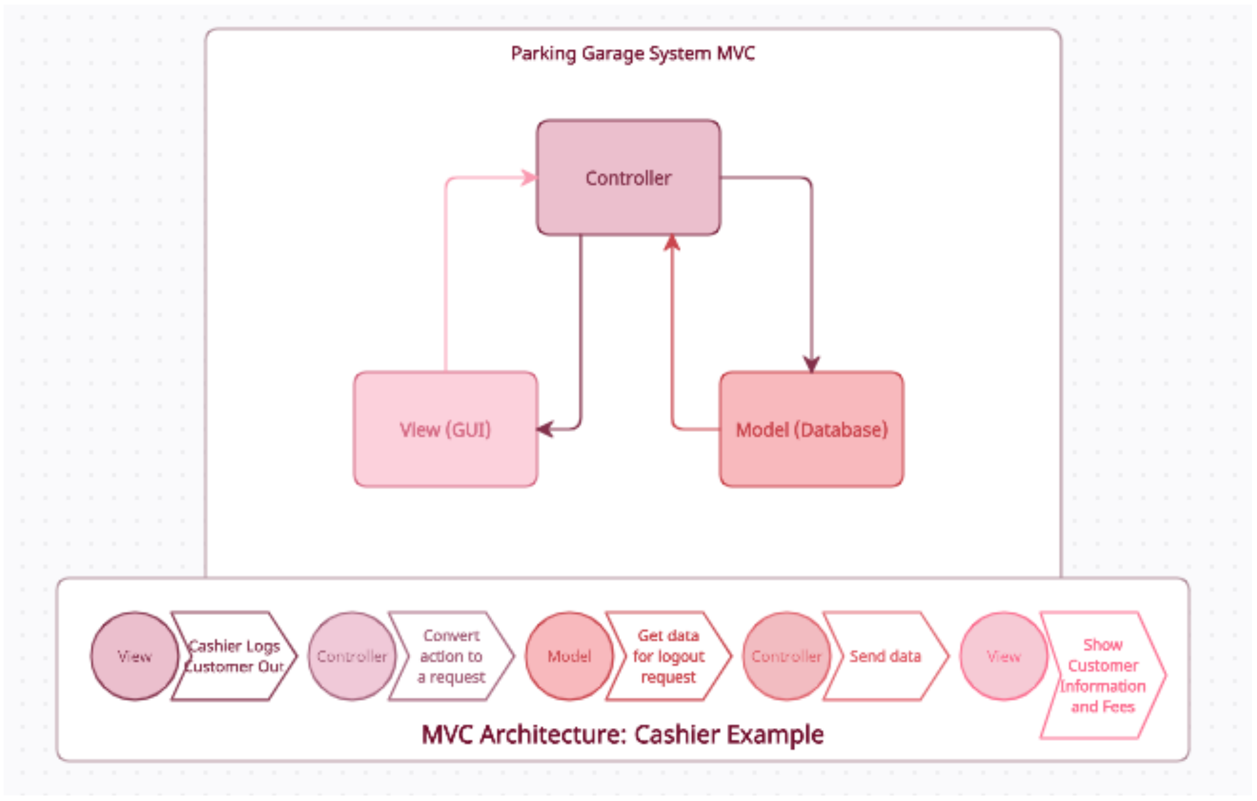
4. Time Frame Screen
    a. Includes 2 time fields where one is the expected entry time and the other field is for the expected exit time.
5. Payment Screen
    a. Includes an input field for the credit card number, another field for the expiration date of that card and a third field for the CVV.
    b. A note that wasn't initially planned to appear on the page but was later added is the note found at the top of the screen stating that in case of more time spent in the garage than originally planned, then extra fees shall be charged.

3. ARCHITECTURAL DESIGN

Software Architecture

In our project, we decided to go with the model-view-controller (MVC) architectural style. The user, whether that's the customer, cashier or manager, interacts with the view. In our case the view is the graphical user interface of our application, where each type of user has a different interface according to the functions needed by that person. After the interaction of the user with the GUI, that interaction is passed over to the controller which decides how to move forward according to the action applied by the user. The controller then proceeds to either gather or send data to the model.

In the case where the user is for example a customer who wishes to reserve a spot at one of the available locations, first the customer interacts with the GUI (view) and picks the desired time frame as well as services needed. Once the customer decides these inputs to the view, the view alerts the controller of that particular event. Next step would be done by the controller which moves forward and updates the model with the reservation.


Parking Garage System MVC
Controller
View (GUI)
Model (Database)
View
Cashier Logs Customer Out
Controller
Convert action to a request
Model
Get data for logout request
Controller
Send data
View
Show Customer Information and Fees
MVC Architecture: Cashier Example

System Architecture

For our system architecture, we will first go into the web front end. The front end was split into 3 different divisions, one for the customer, one for the cashier and finally one for the manager. Each one of those divisions has different pages as each one of the users need a different set of functions.

Firstly we'll go into the customer's view. The customer has a webpage for reserving a spot at the desired location, then he/she will be directed to a login page where the credentials of the person should be inserted to move forward to the next webpage. After successfully logging in, there will be a webpage to pick the exact time frame of the customer's stay at the garage and finally there will be a webpage to pay using a credit card.

Secondly we'll take a look at the cashier's view. Once the cashier logs in to the system, he/she shall be greeted with a welcome message and will have the option to either log a customer in or out depending on the situation at hand.

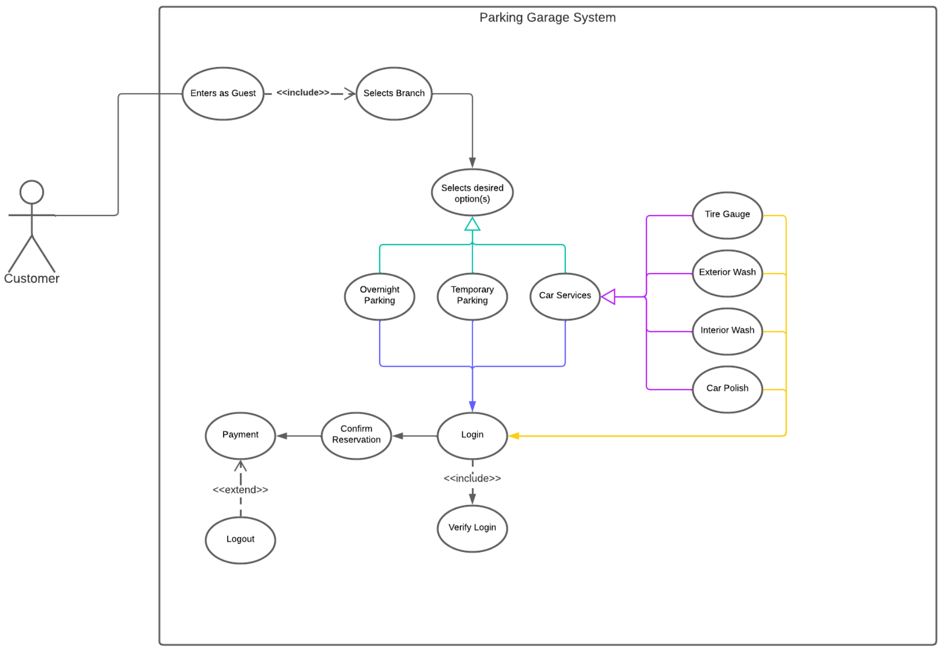
The third view that we'll revise is the manager's view. The manager has a single centralized webpage that allows him/her to have an overview over the whole database. From viewing all customers registered in the system, to current customers staying at all locations and even to future reservations with the possibility to check if those reservations were confirmed or not.

Another part of the system architecture we will proceed to talk about is the database itself. The database is split into multiple tables, some of those tables are concerned with the customers and their reservations while others are concerned with the garages and the services offered at each of the locations available. The tables include: Customer, Current_Customers, Current_Reservation, Reservation_Confirmation, Garage, Provided_Services, Requested_Services and Cashier.

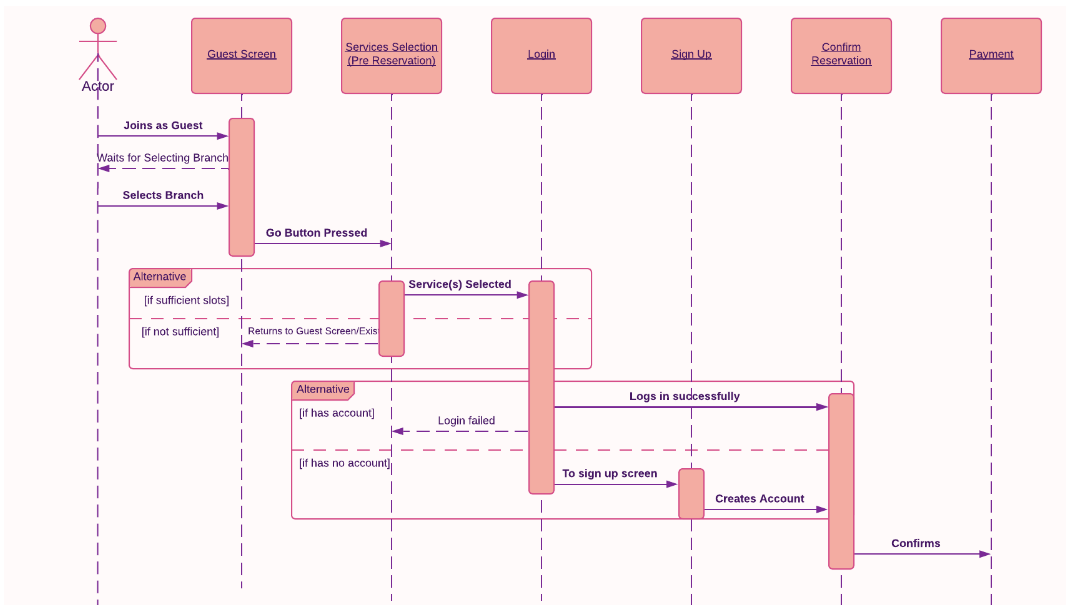
4. SYSTEM MODELING

Customer Diagrams

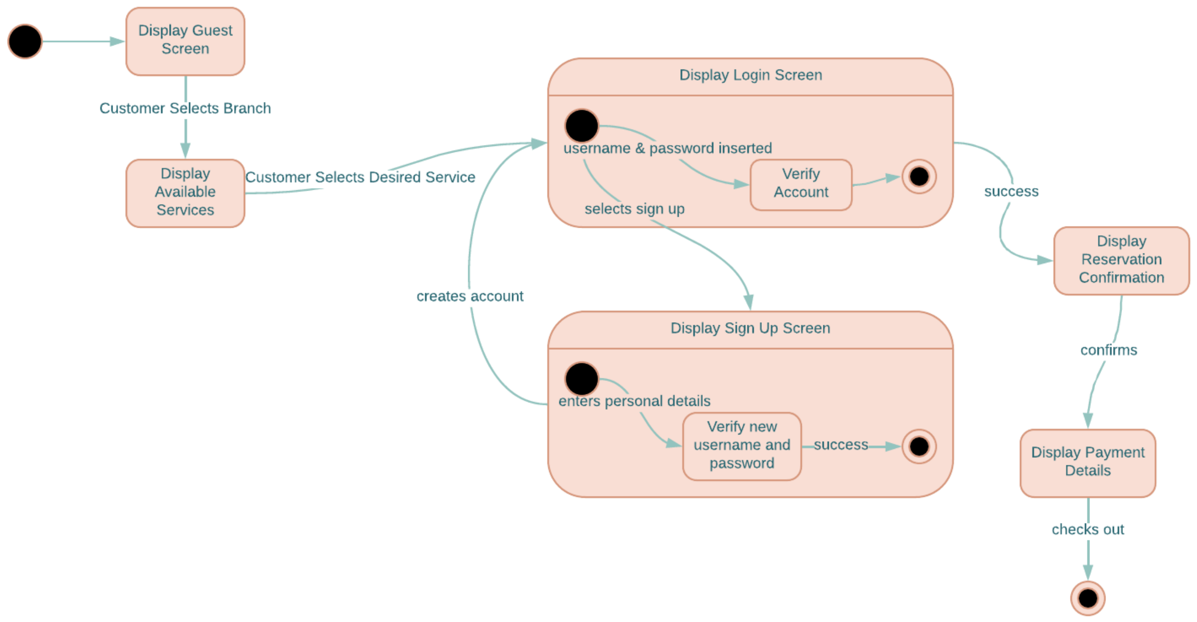
- ***Use Case Diagram***



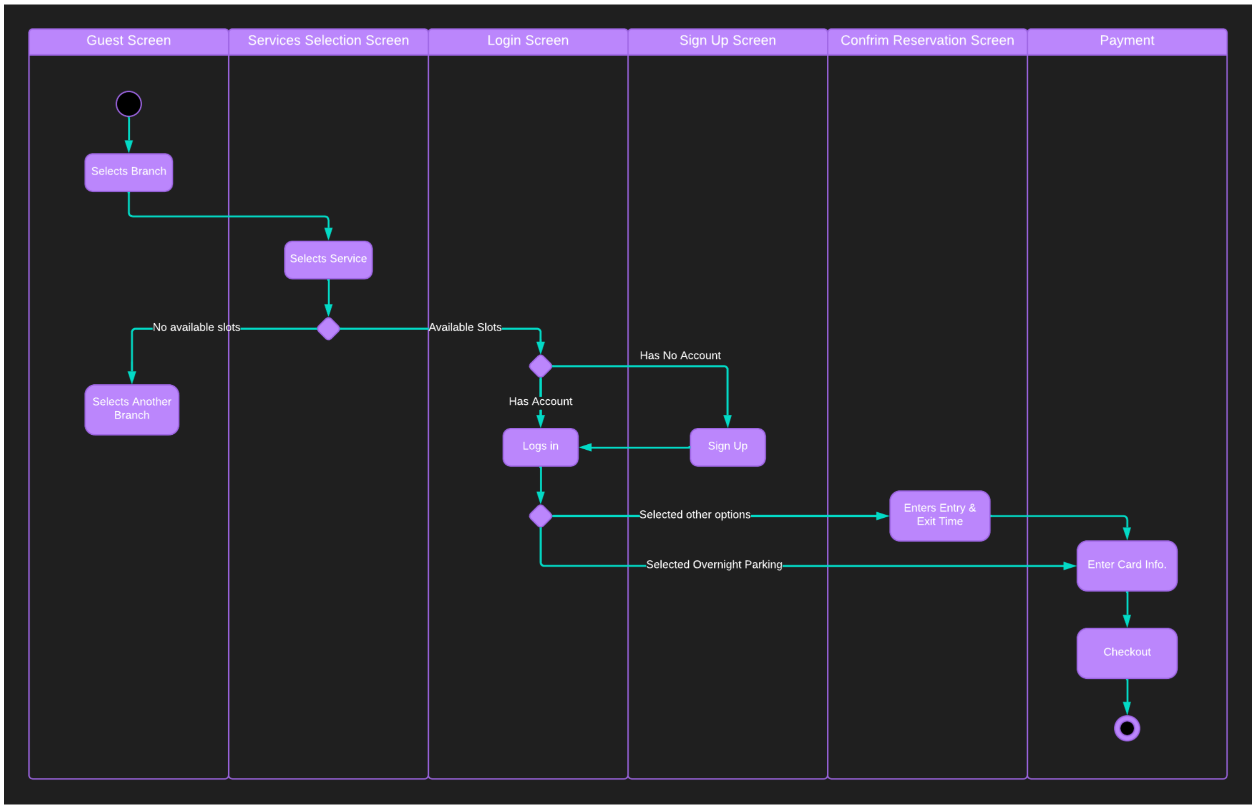
- ***Sequence Diagram***



- ***State Machine Diagram***

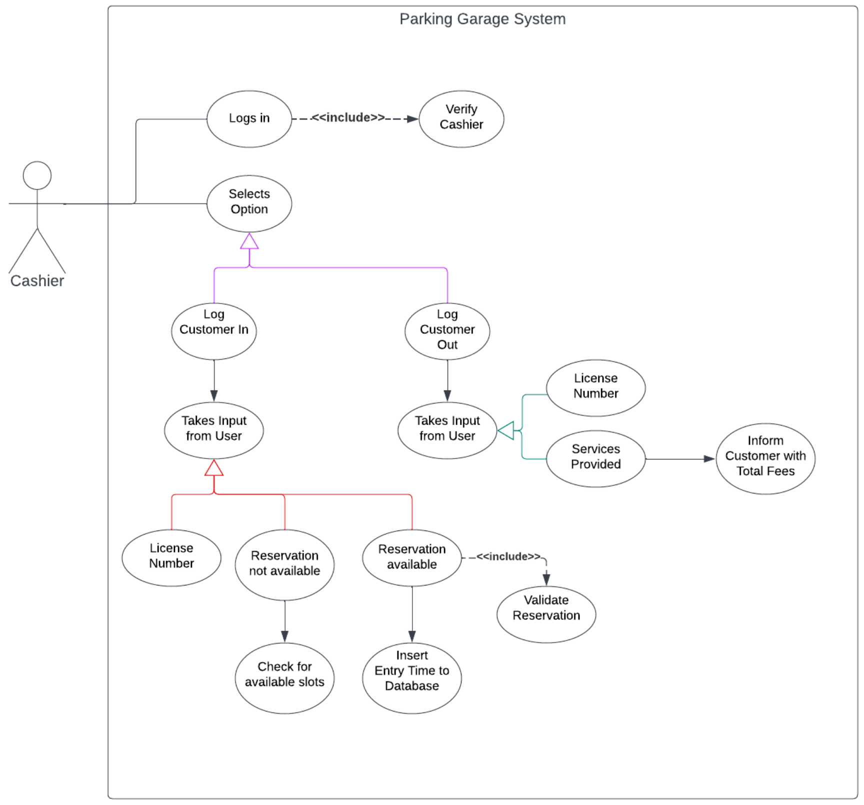


- ***Activity Diagram***

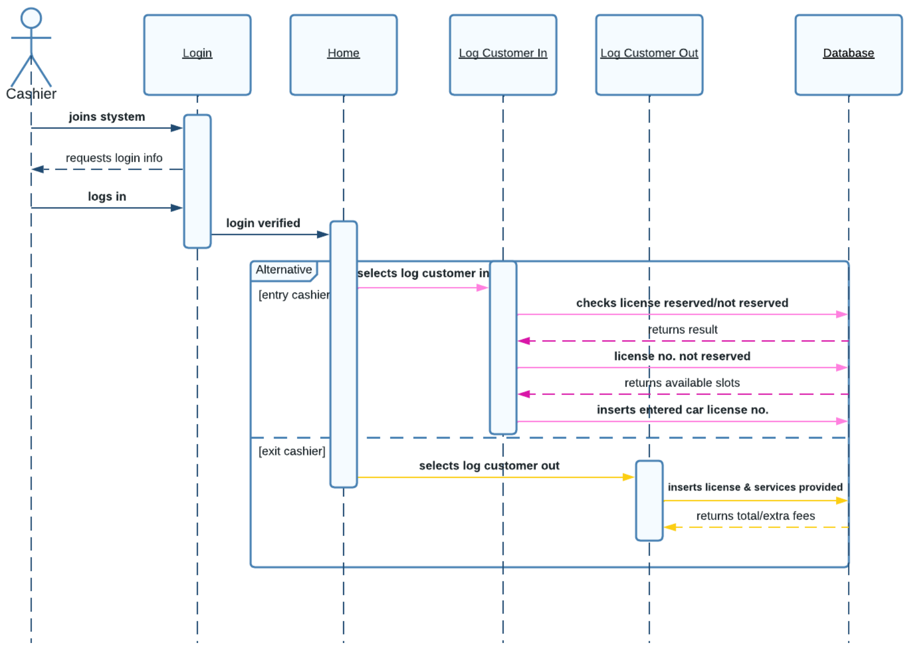


Cashier Diagrams

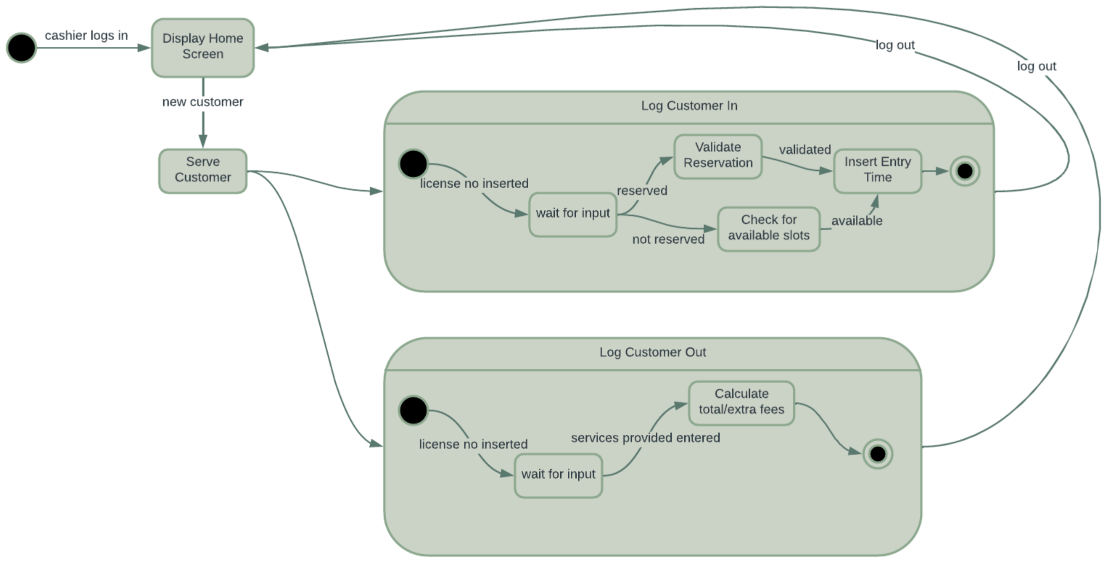
- ***Use Case Diagram***



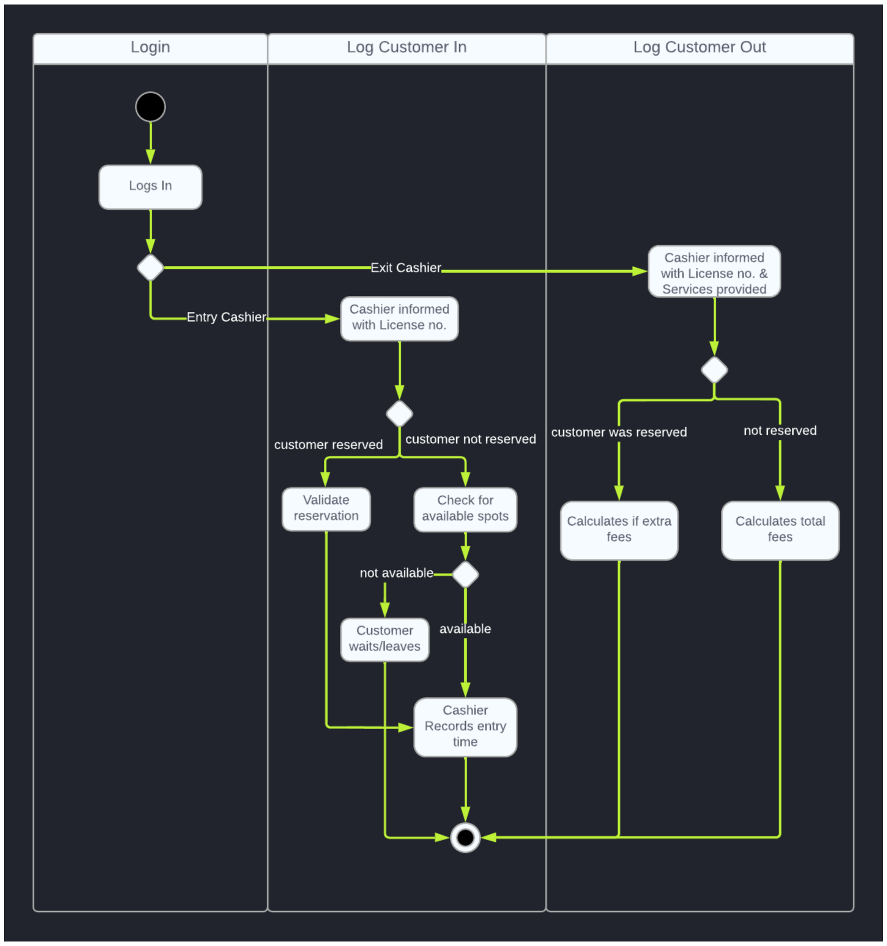
- ***Sequence Diagram***



- ***State Machine Diagram***

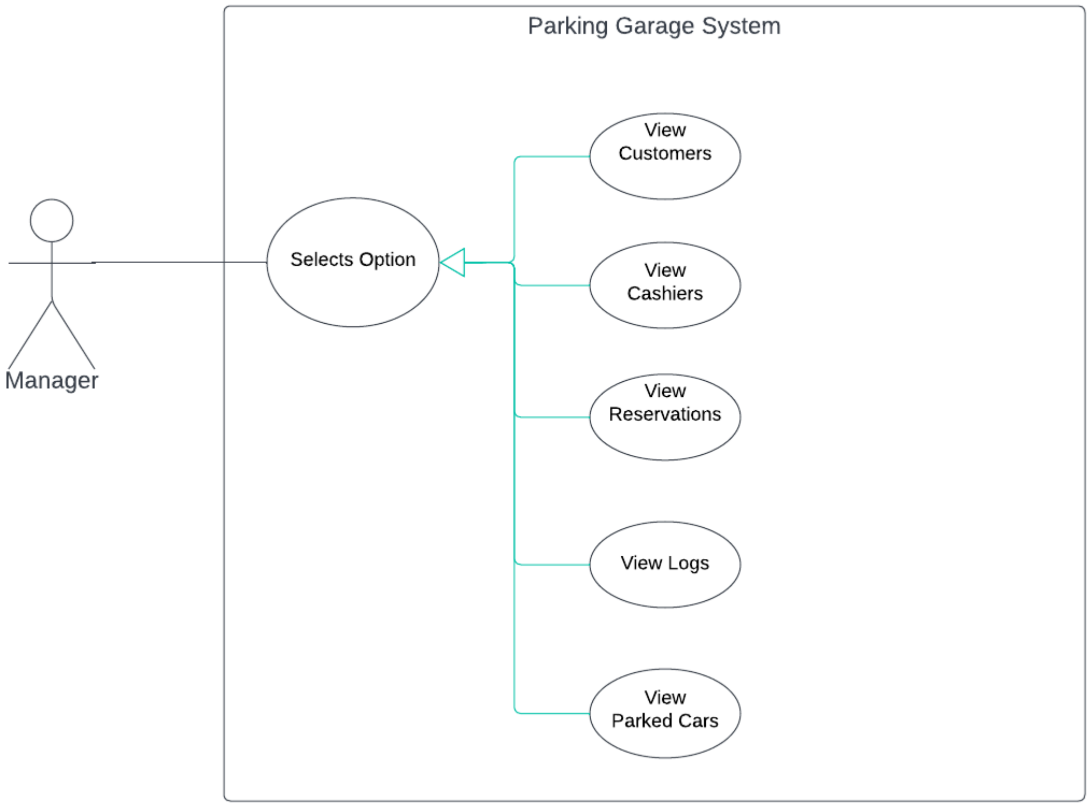


- ***Activity Diagram***



Manager Diagram

- ***Use Case Diagram***



5. DESIGN DESCRIPTION

From the customer's point of view, he would have the option to navigate through the available branches and pick whichever closest to his destination, where the availability of parking spots would appear to him. Then he could choose from several options: Overnight Parking, Car Care Service, Reserve a Spot. Details of every option would be available including price, expected fees in case of cancellation or delay.

From the parking cashier's point of view, he would have to login then have the option to: search for a certain reservation, record entrance of a certain car, reserve a spot if not already reserved, and record number of available spots.

From the parking manager's point of view (admin), he has all the authority to access any branch's details and surveil it. He can also check profits (or deficits) until a certain date, check for complaints and contact customers if required.

6. TESTING

Customer Testing

The user will input the intended branch, the desired services, their username and password, and in the case of temporary parking or any of the car care services, user inputs entry and exit times to/from the. The output will transfer the customer to the payment screen where reservation is completed. The test cases will test the code's output with an expected result, and check to see if they match. The test cases should also be aware of empty or null data entries.

(Welcome Page)

- **Test Case 1**:

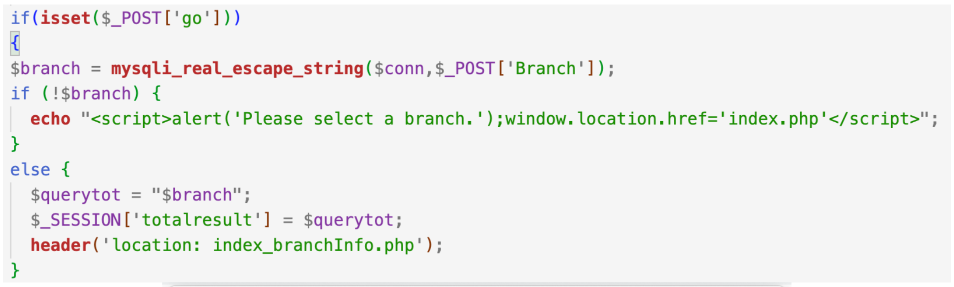
User selects "Gleem" Branch.

Expected: transfer to Branch Information page.

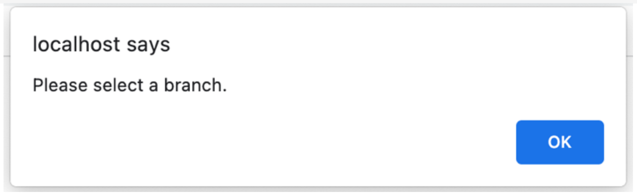
- **Test Case 2**:

User does not select any branch.

Expected: error message and stay in same page until selection.



```
if(isset($_POST['go']))
{
$branch = mysqli_real_escape_string($conn,$_POST['Branch']);
if (!$branch) {
  echo "<script>alert('Please select a branch.');window.location.href='index.php'</script>";
}
else {
  $querytot = "$branch";
  $_SESSION['totalresult'] = $querytot;
  header('location: index_branchInfo.php');
}
```



localhost says

Please select a branch.

(Services Selection Page)

- **Test Case 3**:

User selects one service (temporary parking)

Expected: transfer to Login page.

- **Test Case 4**:

User selects more than one service (temporary parking, interior wash, exterior wash)

Expected: transfer to Login page.

- **Test Case 5:**

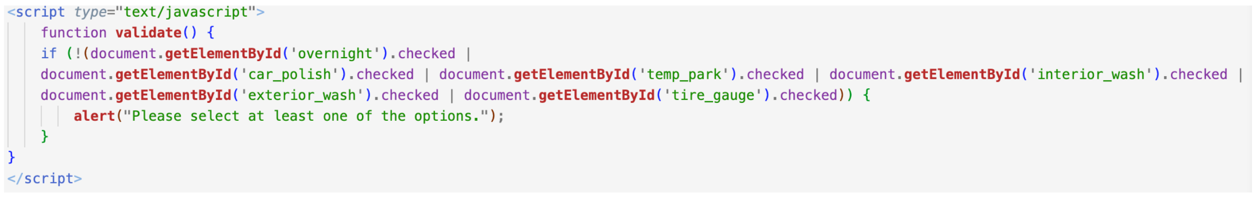
User selects only Overnight Parking service.

Expected: transfer to login page and confirming (entering entry and exit times) is skipped and user is directed directly to Payment Page.

- **Test Case 6**:

User does not select any service

Expected: error message and stay in same page until selection.



```
<script type="text/javascript">
    function validate() {
    if (!(document.getElementById('overnight').checked |
    document.getElementById('car_polish').checked | document.getElementById('temp_park').checked | document.getElementById('interior_wash').checked |
    document.getElementById('exterior_wash').checked | document.getElementById('tire_gauge').checked)) {
        alert("Please select at least one of the options.");
    }
}
</script>
```

(Login Page)

- **Test Case 7**:

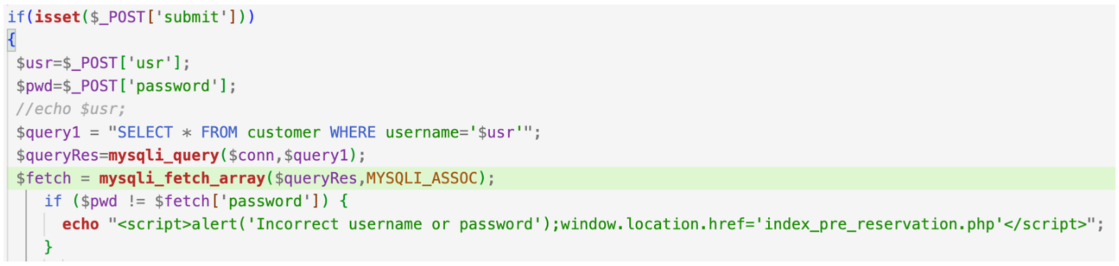
User enters correct username and password.

Expected: transfer to confirming reservation page.

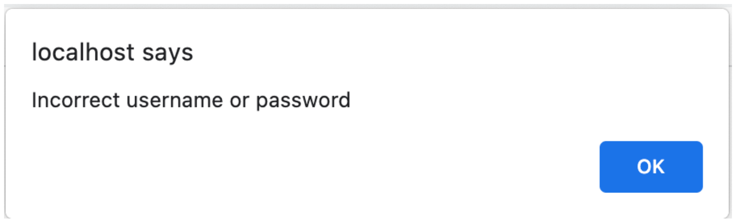
- **Test Case 8**:

User enters incorrect username/password.

Expected: error message and stay in same page until correct entry.



```
if(isset($_POST['submit']))
{
 $usr=$_POST['usr'];
 $pwd=$_POST['password'];
 //echo $usr;
 $query1 = "SELECT * FROM customer WHERE username='$usr'";
 $queryRes=mysqli_query($conn,$query1);
 $fetch = mysqli_fetch_array($queryRes,MYSQLI_ASSOC);
    if ($pwd != $fetch['password']) {
      echo "<script>alert('Incorrect username or password');window.location.href='index_pre_reservation.php'</script>";
    }
```



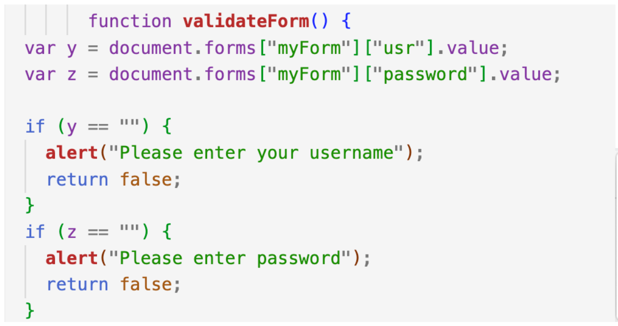
localhost says

Incorrect username or password

- **Test Case 9**:

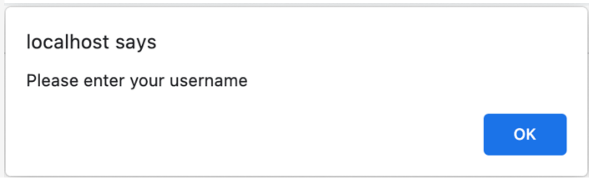
User does not input username/password.

Expected: error message and stay in page until entry.



```
function validateForm() {
var y = document.forms["myForm"]["usr"].value;
var z = document.forms["myForm"]["password"].value;

if (y == "") {
  alert("Please enter your username");
  return false;
}
if (z == "") {
  alert("Please enter password");
  return false;
}
```



```
localhost says
Please enter your username
```

- **Test Case 10:**

User chooses to sign up.

Expected: transfer to sign up page.

(Confirming Reservation Page)

- **Test Case 11**:

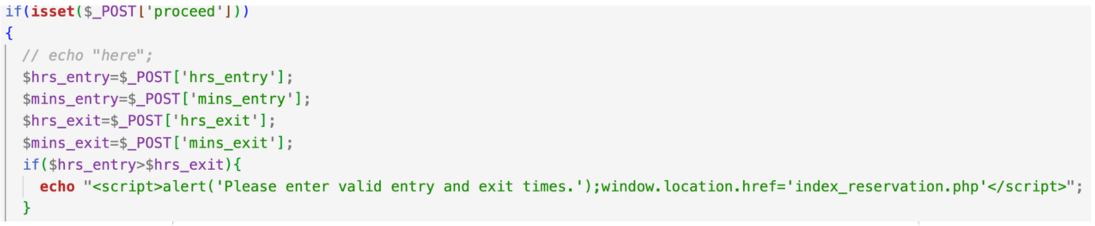
User enters valid entry and exit times.

Expected: transfer to payment page.

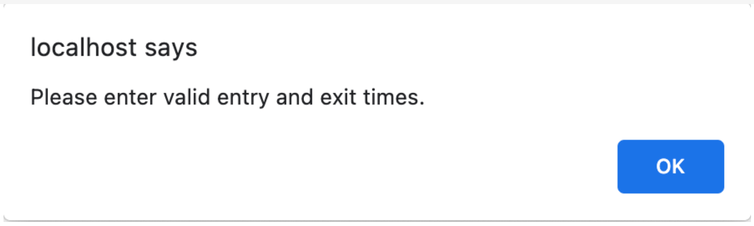
- **Test Case 12**:

User enters valid entry and exit times (entry time > exit time).

Expected: error message and stay in page until valid entry.



```
if(isset($_POST['proceed']))
{
  // echo "here";
  $hrs_entry=$_POST['hrs_entry'];
  $mins_entry=$_POST['mins_entry'];
  $hrs_exit=$_POST['hrs_exit'];
  $mins_exit=$_POST['mins_exit'];
  if($hrs_entry>$hrs_exit){
    echo "<script>alert('Please enter valid entry and exit times.');window.location.href='index_reservation.php'</script>";
  }
```



```
localhost says
Please enter valid entry and exit times.
```