



Internet Programming (TLM2008)

<<Online Car Park Booking>>

<u>Done by</u>	<u>Name:</u>	<u>Student Number:</u>
1.	Ahmad Amir Faris Bin Mazlan	1902020
2.	Alex Goei Jing En	1902005
3.	Choo Xian Zhi	1902026
4.	Haziq Isyraq Shah Bin Bismillah Shah	1901990
5.	Pham Do Minh Quang	1902021
6.	Wilson Lee Kian Yi	1902022

Contents

Project Summary/Abstract.....	3
Project Description	4
1. Why is the application selected?	4
2. Who are the target viewers?	4
3. What does the application service?	4
4. What is/are the problems that you want to solve?	4
5. What is the design concept?	5
6. What are the technologies used in the application?	5
7. Were the design and technologies used able to address the problem?	5
Project Implementation Plan	6
Functionalities of Our Website Explained.....	7
1) Registration and Login	7
2) Homepage.....	9
3) Search Engine Bar	10
4) Display Car Park Details	11
5) Shopping Cart	12
6) Manage User's Booking	14
7) Location Map.....	15
8) Contact Us	18
Problem Faced with Technology	20
HTML.....	20
CSS	20
PHP/JavaScript.....	20
Roles & Responsibilities.....	Error! Bookmark not defined.
Conclusion	21
Individual Reflection	Error! Bookmark not defined.

Project Summary/Abstract

In the modern days of Singapore, personal transportation vehicles such as motorcycles and cars are used and required places to be parked. By considering this factor, the Land Transport Authority of Singapore adopted parking systems where the parking lot occupancy data are collected for usage for applications and processes.

Since data provided comprises a wide range of information ranging from the number of cars spending a specific period of time, to the number of transactions by type from motorcycle to cars. Some factors from the data will help in the planning of how the car is supposed to be parked or whether the parking lot is occupied throughout the year. Some of the areas could be busier during certain days or timing thus affecting the occupancy.

Thus, for our project, we will be building a web application to facilitate the management and booking of car park lots in Singapore.

The web server has two main objectives for the users:

1. Vehicle count and space allocation for the drivers to plan or to decide whether to make a booking.
2. Provide information for the driver/user point of view to check on their booking.

The information collected from the webpage can be used to optimize the parking cost, through information such as when the car park is full or the least amount of parking or even what type of vehicles are expected at a certain timing for future planning.

Things that the web page can have:

- A. Information such as how many slots are available
- B. Information on booking from customers
- C. The amount that has to be paid depending on the duration
- D. Nearest car park in the area

Project Description

1. Why is the application selected?

In the digital world, people have begun to move most of their transactions to online payment. It is because online payment is more secure and convenient. Thus, our website captures this opportunity to levitate users' parking experience with our online booking system.

2. Who are the target viewers?

Our target audiences are car and truck drivers, riders who often need to park their vehicles.

3. What does the application service?

The application provides drivers a platform to locate available car parks in the area using maps. We allow the user to estimate their parking's price at ease. Furthermore, users can identify the crowd level of different car parks.

4. What is/are the problems that you want to solve?

Car parking has always been a common issue especially in common areas such as shopping malls and tourist attractions. Apart from that, it is a hassle to be dealing with parking coupons when parking in certain areas. We are here to provide convenience to common road users by implementing features in our booking site ranging from booking parking lots in advance to comparing or checking prices during specific timing. Users can simply book his/her desired parking lot at specific timing and avoid redundant waiting time during peak hours. This can be done easily within seconds.

5. What is the design concept?

Our design concept is about an online car park booking system. It consists of a search engine, navigation via Google maps, a shopping cart system for checking out, storing, and managing bookings. Users will have to login or sign up at our login page. Once they have logged in, they can have the full experience of navigating around our website. On the home and about pages, they can expect to see the search bar, the current number of bookings, and the total number of members we have. They can also click on the “quick booking” to select which location they want to book at and they can also view the members behind the creation of the website. On the booking page, they can choose whether they want to browse or manage their bookings. Upon clicking on “browse”, users can search for the location they want, and if they find something they want, users can add the location to their cart. Upon clicking on “manage booking”, they can delete their bookings if they want to. The data will be updated in our database. Upon clicking on “location”, users can check the locations where our services are provided. Clicking on the pins/markers, they can view the specific location. Finally, if the users have any questions or complaints, they can email us at “parkinglots@gmail.com” or contact us at “999” on the contact page.

6. What are the technologies used in the application?

- ❖ HTML
- ❖ CSS, JavaScript
- ❖ Ajax,
- ❖ PHP
- ❖ MySQL

The environments that were used to build our applications were ATOM and XAMPP.

7. Were the design and technologies used able to address the problem?

We are able to address the issue with the chosen technologies. For data management, PHP and MySQL are used. The XAMPP environment is used as the server to run our website while at the same time is used to store the database of the website. and for styling and website development, HTML/CSS is used. For enhancement of our website, JavaScript/Ajax is used, displaying appealing effects with the implementation of functions within the web. That being said, these technologies allow smooth integration with each other, providing minimum inefficiency and complications on both frontend and backend development.

Project Implementation Plan

<u>Week</u>	<u>Date</u>	<u>Tasks to be completed</u>
7	22/02 to 26/02	<ol style="list-style-type: none">1. Outline of the project2. Roles & Responsibilities of each member
8-11	27/02 to 22/03	<ol style="list-style-type: none">1. Design Concepts2. Server<ol style="list-style-type: none">a. Host Web Pagesb. PHP to Databasec. User Login Sessions3. Database<ol style="list-style-type: none">a. Carparks & Lotsb. View Lotsc. Reserve Lotsd. Cancel Bookinge. Update Bookingf. View Bookings4. Finalize Webpages<ol style="list-style-type: none">i. Homepageii. View Car Parks with Google Mapiii. View Available Lotsiv. Make Lot Bookingv. View Bookingsvi. Update Booking
12 - 13	23/03 to 01/04	<p>Testings:</p> <ol style="list-style-type: none">1. Build/Test Server2. Design Web Pages3. Connect Database <p>Submission:</p> <ol style="list-style-type: none">4. Finalize Report & Powerpoint Slides ** (26th March)5. Script Writing for Demo6. Group Presentation Video

Functionalities of Our Website Explained

1) Registration and Login

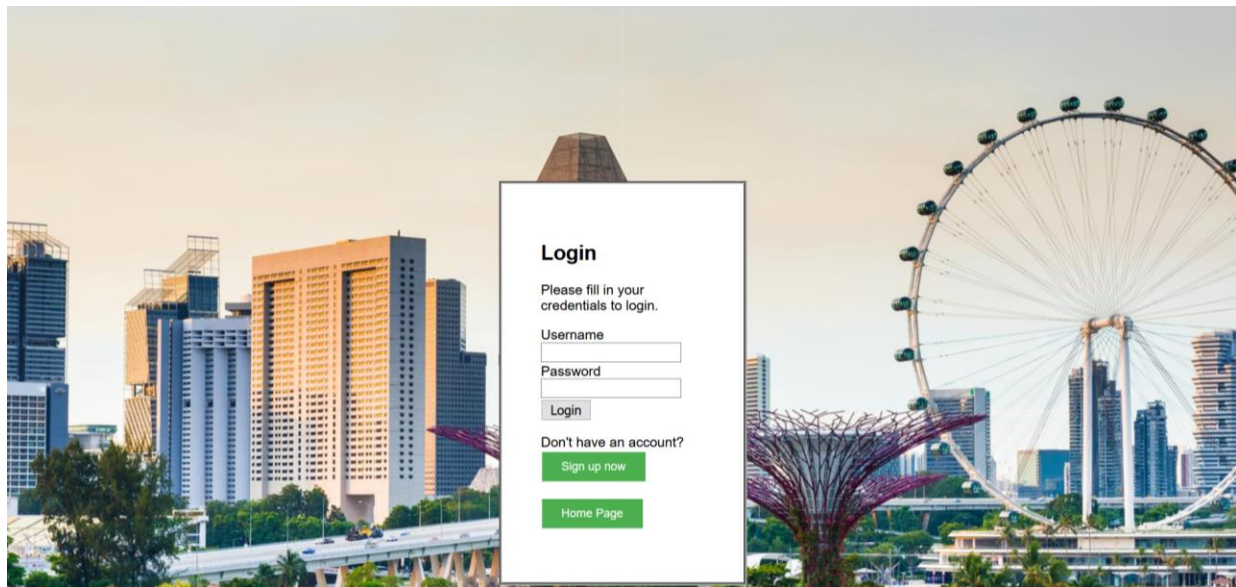


Figure 1

Our log-in portal consists of various functions, login, sign up, and resetting of password. These features will be explained further in this report. In figure 1, this is the display panel of the login and the registration page where the user can enter into the web application after the registration first and login afterward.

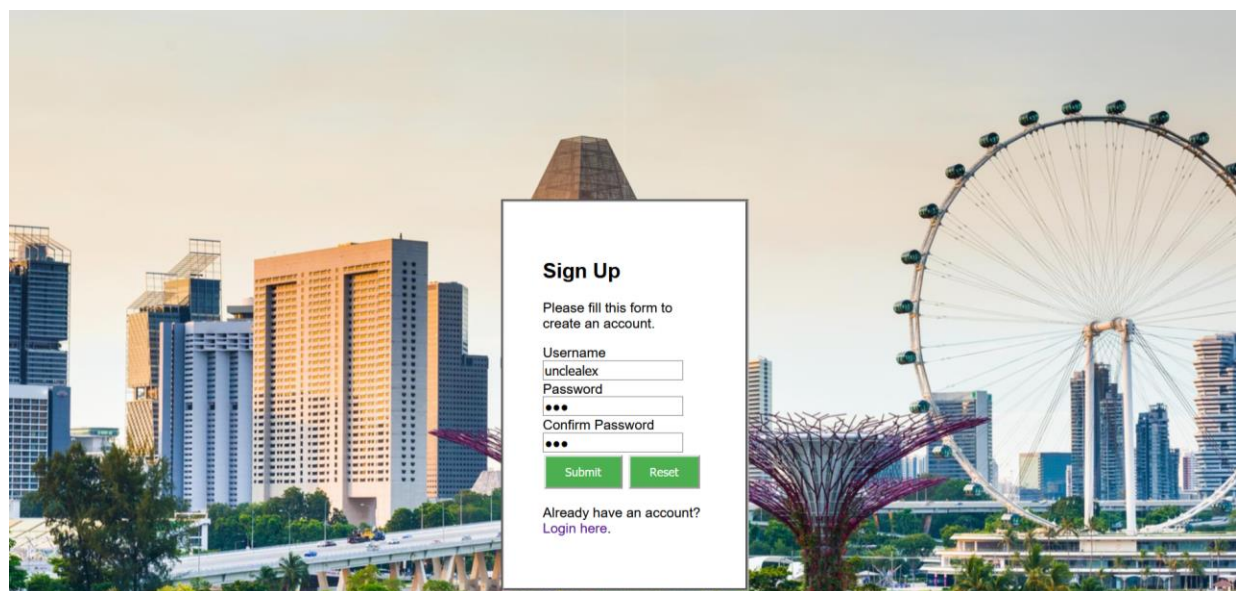


Figure 2

In figure 2, similar to figure 1, shows that the user can sign up for our web application as a user, and the data input from the sign-up will be directed into our database server and stored inside after the submit button was pressed. The data will be stored in the format of two entities which are, the username and the password.

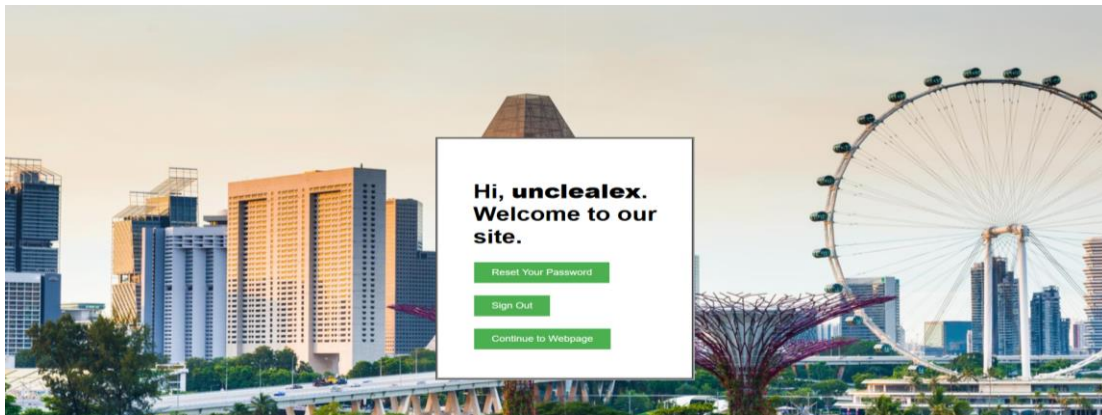


Figure 3

Validations are applied on both our sign-up and login systems. Attempting to login with a wrong password or non-existing username or creating a new user with an existing username, such actions will lead to the prompting of an error message. This is to prevent any duplications on our database for better data management which is vital when dealing with a substantial number of datasets in the near future.

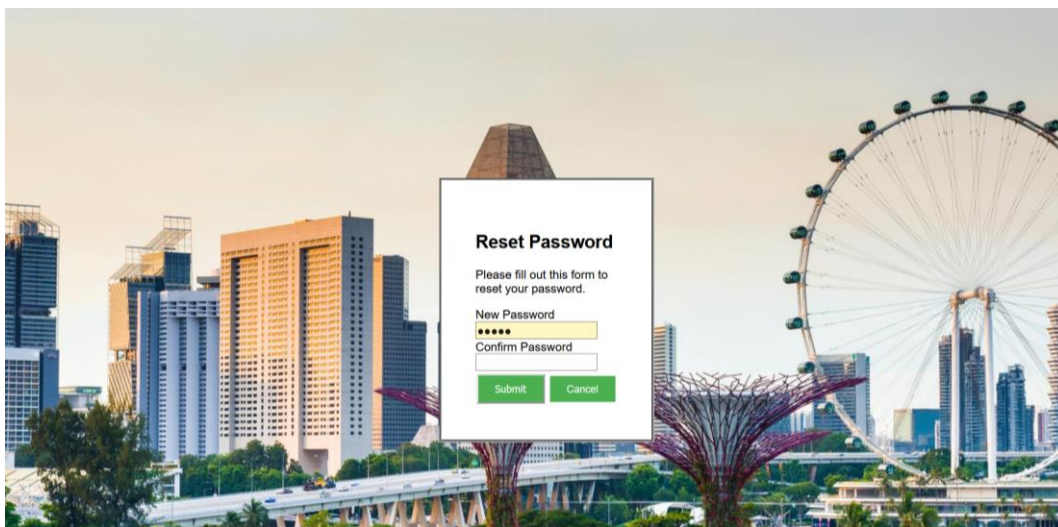


Figure 4

In figure 4, this is to support the user that if he/she forgets the password, there is an option for him/her to enter a new password so that the user can reenter into our web application. This function is important as many users would tend to forget their password after a certain period of time.

2) Homepage

Our web application consists of several main tabs, home, about, booking, location, contact, user identification and shopping cart. In the figure 5 below, the home or the main page consists of the overview of our information and explains how each user in our application will benefit from. In the bottom half of the page, it provides the motto of the web application and provides the details of our team. This ensures that interactions between the users and us would be sustainable and if the users need any form of help, they will have a platform to know who to look for as well.

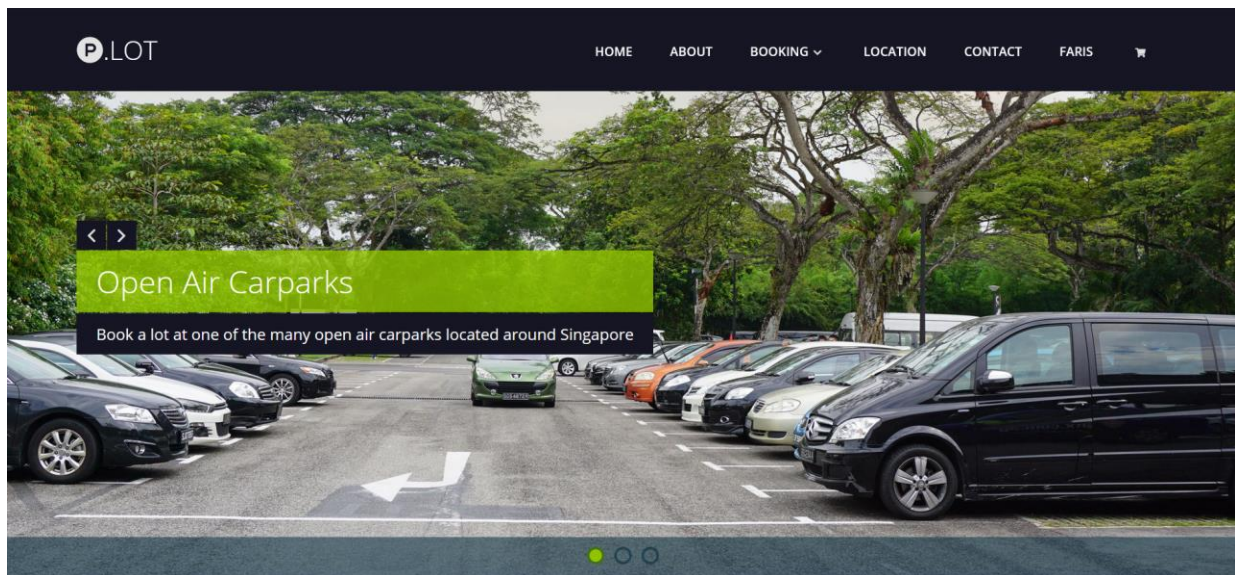
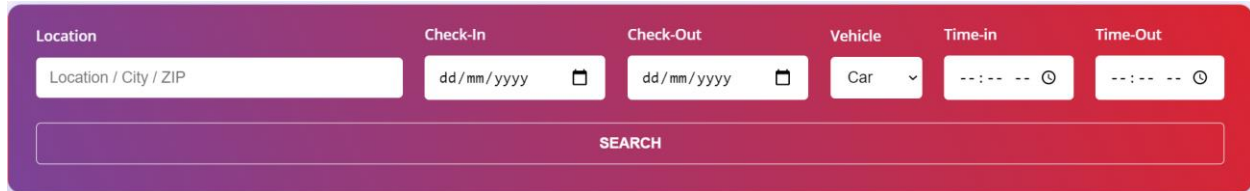


Figure 5

3) Search Engine Bar

The search engine allows users to locate the destination for their desired car park. Besides location, the user is required to fill in his/her check-in and check-out date, vehicle type and time, for the insert statement later as shown in figure 6 below.

The image shows a search engine bar with a purple-to-red gradient background. It contains several input fields: a text box for 'Location / City / ZIP', two date pickers for 'Check-In' and 'Check-Out' (both showing 'dd/mm/yyyy'), a dropdown menu for 'Vehicle' (set to 'Car'), and two time pickers for 'Time-in' and 'Time-Out' (both showing '--:-- --'). A 'SEARCH' button is located at the bottom center of the bar.

Location	Check-In	Check-Out	Vehicle	Time-in	Time-Out
Location / City / ZIP	dd/mm/yyyy	dd/mm/yyyy	Car	--:-- --	--:-- --
SEARCH					

Figure 6

We used a 'POST' method to collect form data once a user has submitted the required fields on our search engine bar. Basically, the data are fetched from our database with regular expression in place to search for all corresponding car parks located in the searched area. Afterwards, users can browse through all the car parks and add the lot to our shopping cart that they wish to book. In short, it provides convenience to users, by allowing them to locate the desired car park in the shortest and effective manner instead of having to check through the whole to find out if the located car park is available or to even book for the lot itself.

4) Display Car Park Details

SEARCH.

All Available Parking Lots in Yishun

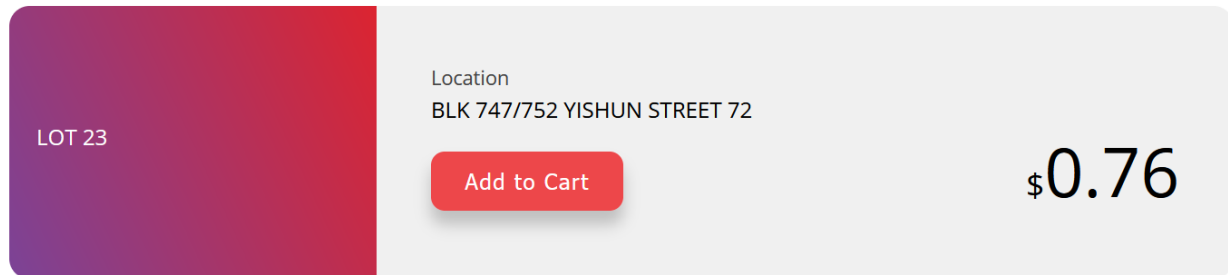


Figure 7

This section is where the user is able to browse his/her desired booking location. Once a user fills all required fields under the search engine, upon clicking “Search”, all available parking lots will be displayed below the header based on the requirements entered by the user. The header will then display the user’s input location and available parking lots will indicate the available lot, location, and price.

```
if (count($_POST) > 0) {  
    $address = $_POST["address"];  
    $desired_checkin = $_POST["enter_date"] . " " . $_POST["enter_time"];  
    $desired_checkout = $_POST["exit_date"] . " " . $_POST["exit_time"];  
    $search_list_carparkIDs_query = "select * from user_reservation where address like '%{$address}%' AND check_in='';  
  
    $result = $db->query($search_list_carparkIDs_query);  
    $rows = $db->fetch($result);  
}
```

Figure 8

The code [Figure 8] is to ensure the data that is desired to be used goes through the server and display out the desired data required from the database server. The information shown via the webpage is to display the no. of lot, location of the lot and the price of the lot.

5) Shopping Cart

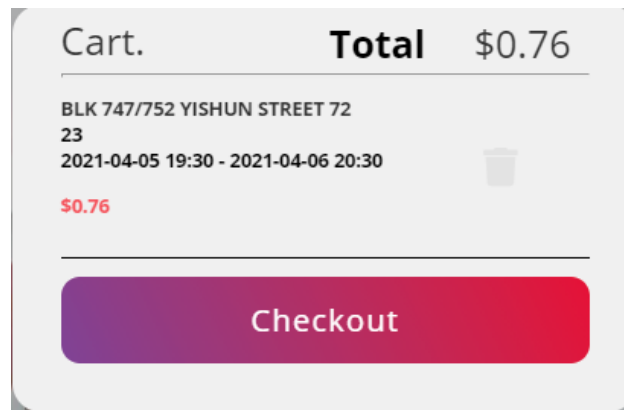


Figure 9

This is our shopping cart section. Basically, all information that the user entered previously in our search engine, will be displayed in our shopping cart once “Add to Cart”. For further simplicity, total cost will be tallied on the cart. Additionally, the user can make any necessary amendments to their cart by clicking the trash icon.

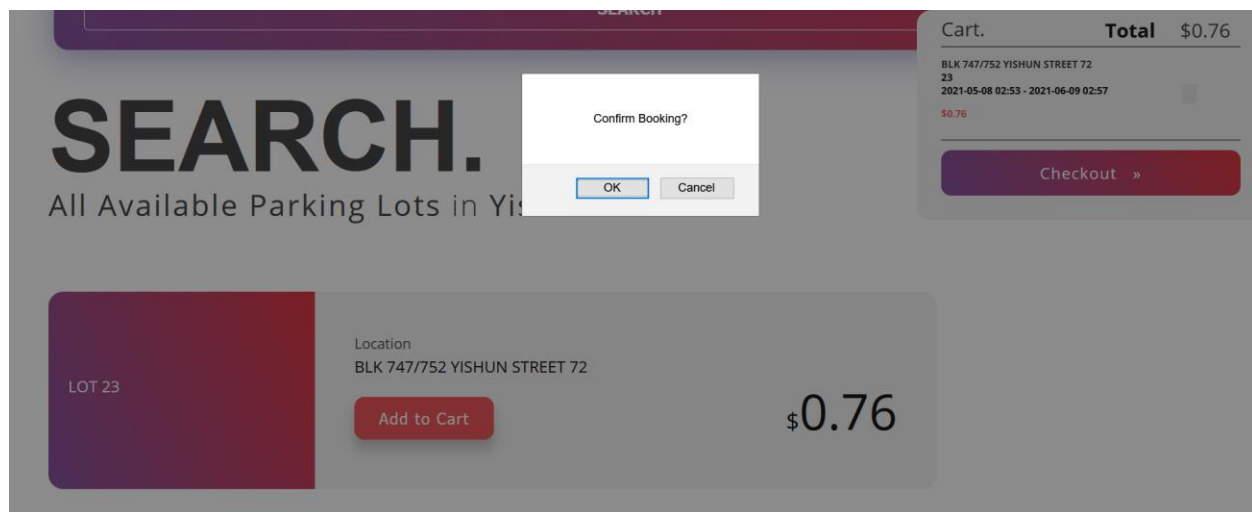


Figure 10

Upon checkout, our site will prompt a message for confirmation of booking via JavaScript. Once confirmed, the user has successfully made booking under our system and booking data will be stored in our database. All key information on the cart will be sent to the PHP backend via JavaScript Ajax after the checkout button is clicked. Data are sent, containing all the booking information that existed in the cart. Ajax allows the front-end to make a POST request to the PHP URL asynchronously (without reloading the page).

```

function confirmBooking() {
    var myWindow;
    $.ajax({
        url: "data/get-cart.php",
        type: "POST",
        data: {'cart': item_carts},
        success: function(data, textStatus, jqXHR) {
            if(confirm("Confirm Booking?")){
                window.open("loading.php");
            }
            document.getElementById("btn-purchase").innerHTML = myWindow;
        },
        error: function(jqXHR, textStatus, errorThrown) {
            alert('Error occurred!');
        }
    });
}

```

Figure 11

After receiving the data using \$_POST, we iterate through all the items in the cart and insert them into our database with the codes in figure 11 and 12 respectively.

```

for ($i=0; $i < count($cart_items); $i++) {
    $userId = $_SESSION['id'];
    $carparkId = $cart_items[$i]['carparkId'];
    $lotId = $cart_items[$i]['lotId'];
    $address = $cart_items[$i]['address'];
    $price = $cart_items[$i]['price'];
    $check_in = $cart_items[$i]['check_in'];
    $check_out = $cart_items[$i]['check_out'];

    $insert_booking_query = "INSERT INTO `user_reservation` (`userId`, `carparkId`, `lotId`, `address`, `price`, `check_in`, `check_out`)
    VALUES ('{$userId}', '{$carparkId}', '{$lotId}', '{$address}', '{$price}', '{$check_in}', '{$check_out}')";
    $result = $db->query($insert_booking_query);
}

```

Figure 12

6) Manage User's Booking

Manage Booking

Booking ID	Customer ID	Address of carpark	Lot ID	Check-In	Check-Out	Action
39	21	BLK 747/752 YISHUN STREET 72	23	2021-04-05 19:30	2021-04-06 20:30	Delete

Figure 13

This section, the user is able to track all the details of all his/her bookings. This can be easily accessed from “Manage Booking”, under the dropdown of the navigation “Booking”. All new and existing bookings are displayed over here and the data corresponds with whatever the user has input.

```
foreach ($rows as $value) {
    ?>
    <tr>
        <td><?php echo $value["bookId"]; ?></td>
        <td><?php echo $value["userId"]; ?></td>
        <td><?php echo $value["address"]; ?></td>
        <td><?php echo $value["lotId"]; ?></td>
        <td><?php echo $value["check_in"]; ?></td>
        <td><?php echo $value["check_out"]; ?></td>
        <td><a class="delete" href="data/delete-booking.php?bookId=<?php echo $value["bookId"]; ?>">Delete</a></td>
    </tr>
    <?php
    }
    ?>
</table>
```

Figure 14

Additionally, an anchor element has been created for deletion purposes for bookings. The user can delete their existing booking and our database will be updated simultaneously.

7) Location Map



Figure 15

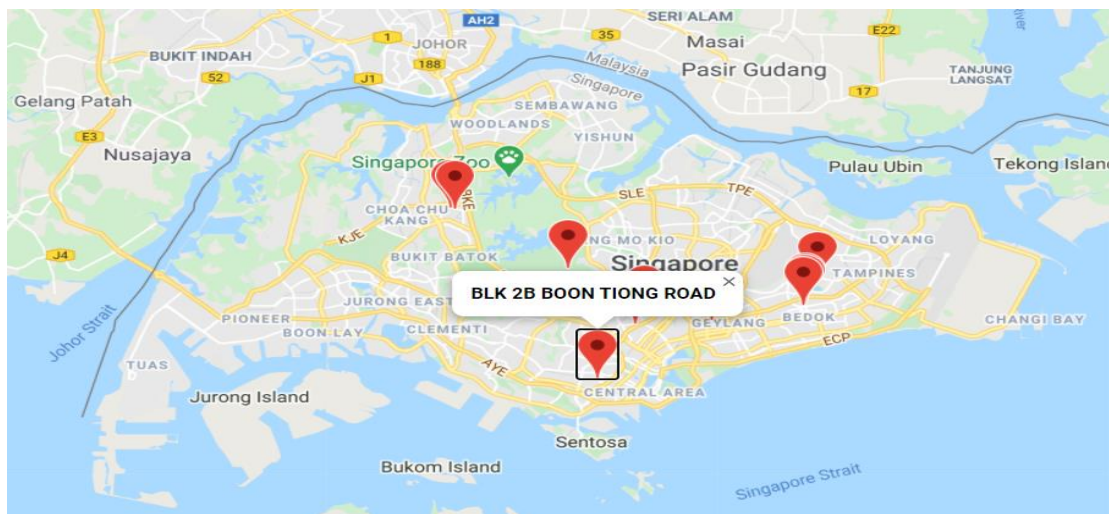


Figure 16

This section is where users are able to view the location of all the parking lots that are available by our service with the help of Google JavaScript API and Google Geocode API. Each marker represents the location of a parking lot. When the marker is clicked on, an info window would open, and the address of the parking lot would be shown.

```
<script async src = "https://maps.googleapis.com/maps/api/js?key=AIzaSyAFJgy-VzJ2Mb67aqbRGkhmWe6k7V0fYwk&callback=initMap"></script>
```

Figure 17

First to be able to view the map, a google API key is needed. We head over to the google maps platform to create and enable our API key. For our website, we set a restriction to only use Google Maps JavaScript API and Google Geocode API as we do not require the other services that were available. The API key is then added to the URL within a script tag as shown above [Figure 17].

After setting up the google map API, we needed to ensure that when the map is loaded in our website, the map would be located in Singapore. To ensure that the map is set in Singapore, a variable with the longitude and latitude of Singapore is declared. With the new google Maps. Map(), a google maps object is created where the center property informs the API where the center of the map is. The code shown below is used to set the overall map to be centered in Singapore.

```
function initMap() {  
  var singapore = {lat: 1.3521, lng: 103.8198};  
  map = new google.maps.Map(document.getElementById('map'), {  
    zoom: 11,  
    center: singapore  
  });  
}
```

Figure 18

Next, to show all the locations of the parking lots that are stored in a database, we made a get function specifically to retrieve all data related to locations. This can be seen in the code shown below [Figure 19]

```
public function getAllLocation() {  
  $sql = "SELECT * FROM $this->tableName";  
  $stmt = $this->conn->prepare($sql);  
  $stmt->execute();  
  return $stmt->fetchAll(PDO::FETCH_ASSOC);  
}
```

Figure 19

Next, to ensure that the Google Geocode API is able to convert addresses in the database into geographical coordinates, the Geocoder constructor is used. With the use of this constructor, every new instance would send a geocode request to Google Servers. This can be seen in the code shown below.

```
geocoder = new google.maps.Geocoder();
```

Figure 20

With the Geocoder constructor, a function would be needed so as to parse the parking lot address and return a geographical coordinate. The set of codes seen below is used to achieve this.


```

function codeAddress(cdata){
    Array.prototype.forEach.call(cdata, function(data){
        var address = data.address + ' ' + data.id;
        geocoder.geocode( { 'address': address}, function(results, status){
            if (status == 'OK') {
                map.setCenter(results[0].geometry.location);
                var points = {};
                points.id = data.id;
                points.lat = map.getCenter().lat();
                points.lng = map.getCenter().lng();
                //alert(map.getCenter().lat())
                updateLocationWithLatLng(points);
            } else {
                alert('Geocode was not successful for the following reason: ' + status)
            }
        });
    });
}

```

Figure 21

Next, to set the markers on the map according to the coordinates of the parking lot, a function is made. With the use of the `Array.prototype.forEach.call()` function, all the data is set in an array and a function is declared where each array would have a marker allocated to it. The code for this function is as shown below.

```

function showAllLocation(allData){
    var infoWind = new google.maps.InfoWindow;
    Array.prototype.forEach.call(allData, function(data){
        var content = document.createElement('div');
        var strong = document.createElement('strong');
        strong.textContent = data.address;
        content.appendChild(strong);

        var marker = new google.maps.Marker({
            position: new google.maps.LatLng(data.lat, data.lng),
            map: map
        });
    });
}

```

Figure 22

Finally, an `addListener` method was used where the moment the user clicks on a marker, a function would be called. This function is tasked with opening the information window. The code for this method is shown in Figure 23.

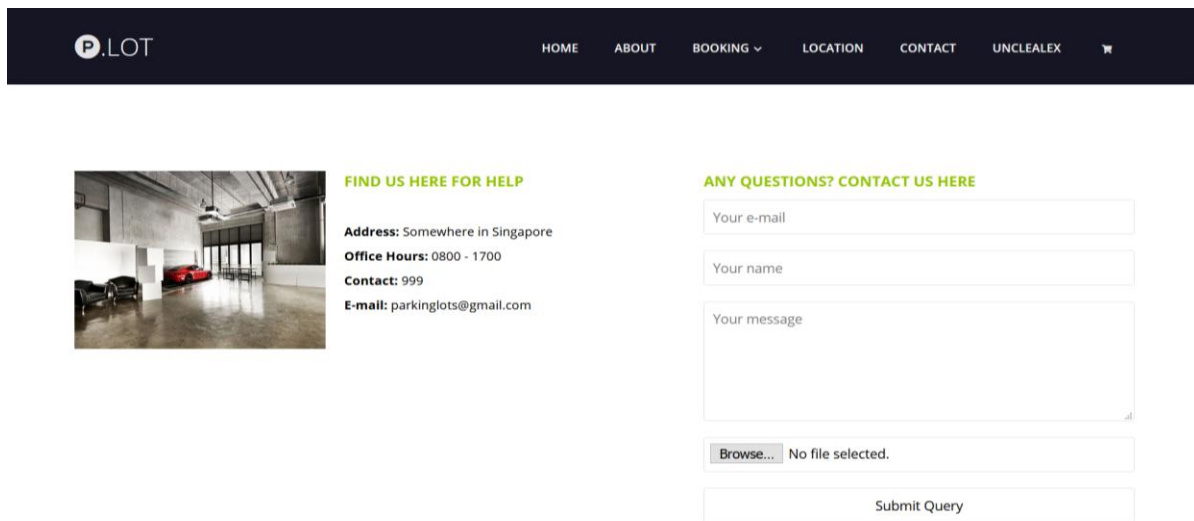
```

marker.addListener('click', function(){
    infoWind.setContent(content);
    infoWind.open(map, marker);
});

```

Figure 23

8) Contact Us

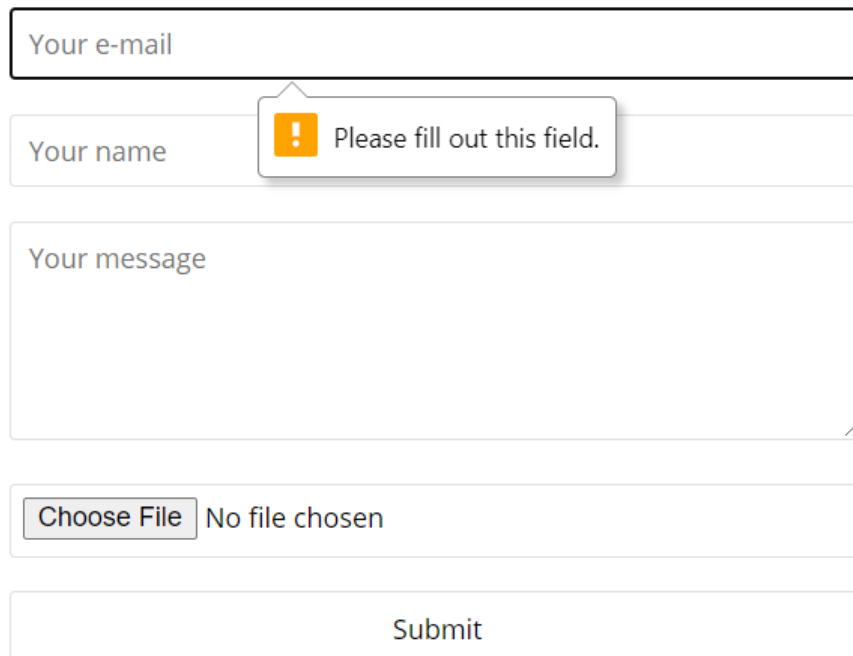


The screenshot shows the 'Contact Us' page of P.LOT. At the top is a dark navigation bar with the P.LOT logo and links for HOME, ABOUT, BOOKING, LOCATION, CONTACT, and UNCLEALEX. The main content area is divided into two columns. The left column features a photo of a parking garage and contact information: 'FIND US HERE FOR HELP', 'Address: Somewhere in Singapore', 'Office Hours: 0800 - 1700', 'Contact: 999', and 'E-mail: parkinglots@gmail.com'. The right column is titled 'ANY QUESTIONS? CONTACT US HERE' and contains a form with fields for 'Your e-mail', 'Your name', and 'Your message'. Below these is a file upload section with a 'Browse...' button and the text 'No file selected.' At the bottom of the form is a 'Submit Query' button.

Figure 24

In this section, we have our “contact us” page as shown in Figure 24. Users can contact us through an e-mail. All of the fields must be filled before the users can submit the form. All of the information that is submitted will be stored in our database. In short, it provides the platform for the company and the users to have interactions and understand how our company collaborated car parks are doing and ensuring that the services provided to the users are still in tip-top condition.

ANY QUESTIONS? CONTACT US HERE



This figure provides a detailed view of the contact form. It includes the following elements: a 'Your e-mail' text input field; a 'Your name' text input field with a validation error message 'Please fill out this field.' displayed in a callout box; a 'Your message' text area; a file upload section with a 'Choose File' button and the text 'No file chosen'; and a 'Submit' button at the bottom.

Figure 25

```

<form class="customform" method="post" enctype="multipart/form-data">
  <div><input name="email" placeholder="Your e-mail" title="e-mail" type="text" required /></div>
  <div><input name="username" placeholder="Your name" title="username" type="text" required /></div>
  <div><textarea placeholder="Your message" name="message" rows="5" required></textarea></div>
  <input type="File" name="file">
  <input type="submit" name="submit">
</form>

```

Figure 26

```

#upload directory path
$uploads_dir = 'files';
#TO move the uploaded file to specific location
move_uploaded_file($name, $uploads_dir.'/'.$pname);

#sql query to insert into database
$sql = "INSERT into upload3(email,username,message,file) VALUES('$email','$username','$message','$pname')";

if(mysqli_query($conn,$sql)){
  $alert = "<script>alert('Thank You for your Feedback!');</script>";
  echo $alert;
}
else{
  $Error = "<script>alert('There seems to be an error! Please try ');</script>";
  echo $Error;
}

```

Figure 27

These codes shown in Figure 26 and Figure 27 are to create the form template out and provide the validation check for each of the sections to be filled and not leaving them blank. The use of type="text" required makes it so that the users must put in their information before they can submit the form.

The lower half of the code is to support the web page to have the function of the users to insert any files that the users want to the company such as images of the car park condition or complaint letters etc. Then, it will be uploaded to our database and the information will be stored there until it is deleted.

Problem Faced with Technology

HTML

For the HTML, when merging the footer and data from different sources and people, it causes lots of conflicts. First, we faced the issue of improper usage of form tags error because when creating the HTML page, the team did not assign the number for the headers which causes a block-level tag error. It took everyone some time to standardize the values for everyone so that eventually everyone's effort is not gone to waste or keep having errors when compiling the HTML codes. One way that we took is to have better communication within the team and eventually came out with the table so that codes of each other would not conflict.

CSS

For the CSS side, we wanted to make use of just one CSS file as universal template for all web pages with header, footer, and buttons size etc. to reduce the time and effort of needing to change the element in every page. However, when the codes get more complex, the standardization is not able to fulfill all of our requirements and the only two solutions that we could think of, are either to not touch CSS at all or to include different CSS file to each page to ensure the aesthetic of the webpage is able to keep up with what we want.

PHP/JavaScript

In this project, we believe that this is one of the most important parts of our project where both of them work differently for the web page application. For the PHP side, our initial thinking of PHP is to use it as an all-time features language to develop the complete applications. But this thinking was wrong as PHP is actually a server-side script that is interpreted on the server and the use of JavaScript is the platform for interpretation on the client-side script that will be embedded into HTML pages.

With more interaction with Mr. Howey and with the lab session for Internet Programming, we manage to enhance our learning about PHP, and we understand that PHP has its own built support for working hand in hand with MySQL where knowledge were taught in our module of TLM2004 with the SQL server side.

Conclusion

To conclude, a web application is developed to provide convenience or to spread awareness to the market about certain problems in the internet where in our case, the application is able to support the car owners looking for an available car park lot. In the modern days of Singapore, this application would then be beneficial for users as the built-in functions would be able to provide the users the needs that they are looking for, be it on the searching for a lot to the price range of the car park lots for a certain amount of time. With the use of the technology used for this application, it also enhance the learning for us, as students, to understand how each of the languages or the software used can be synchronized to each other with their usage and activities and provide us the platform to create a channel for drivers to have their car park lot booked safely as an user and in the long run, it might help the government bodies in managing the traffic in car park with the application too.