PAY AS YOU PARK SMART PARKING SOLUTION: SUGGESTING OPTIMAL PARKING YARD TO PARK

Mohamed Aadil

(IT18013092)

B.Sc. (Hons) Information Technology – Software Engineering

Department of Computer Science and Software Engineering

Sri Lanka Institute of Information Technology

Sri Lanka

September 2021

PAY AS YOU PARK SMART PARKING SOLUTION: SUGGESTING OPTIMAL PARKING YARD TO PARK

Aadil M.R.M

(IT18013092)

Dissertation submitted in partial fulfillment of the requirement for the Bachelor of Science in Information Technology specialization in Software Engineering

Department of Computer Science and Software Engineering

Sri Lanka Institute of Information Technology

Sri Lanka

September 2021

Declaration

I declare that this is our own work and this proposal does not incorporate without acknowledgement any material previously submitted for a degree or diploma in any other university or Institute of higher learning and to the best of our knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgement is made in the text.

Also, I hereby grant to Sri Lanka Institute of Information Technology, the nonexclusive right to reproduce and distribute my dissertation, in whole or in part in print, electronic or other medium. I retain the right to use this content in whole or part in future works (such as articles or books).

Name	Student ID	Signature
Aadil M.R.M	IT18013092	

The above candidates are carrying out research for the undergraduate Dissertation under my supervision.

Signature of the Supervisor: Date

DEDICATION

This research is dedicated to all the people who daily travel in the road who facing many difficulties in finding a better parking yard to park. In urban cities parking is an essential task. People face to this problem in their day-to-day life.

My solution is predicting an optimal parking yard for a vehicle considering availability of free spaces, physical characteristic of vehicle, distance and time taken to reach the destination. Considering the problems faced by drivers my effort is to introduce a smart optimal parking for a better parking.

ACKNOWLEDGEMENT

I would like to present my heartfelt thanks for Ms. Nadeesha Pemadasa(Assistant Lecturer – Faculty of Computing SLIIT) for the support, guidance and motivation throughout the research as our supervisor for make it successful and effective research.

Also would like to thank Ms. Thamali Dissanayake (Lecturer – Faculty of Computing SLIIT) for the support, guidance and motivation throughout the research as our co supervisor for make it successful and effective research.

In addition to that I would like thank Dr. Dharshana Kasturiyatna for the special guidance given in the beginning of the research. He mainly helped us with the research components in the earlier stages.

ABSTRACT

Parking in urban cities is the most difficult task for drivers. As the result of high population; number of vehicles in the road also gradually increased. As a result, finding a correct parking place is a difficult task for most of the drivers as the result of unavailability of spaces. Proposed system provides an optimal and accurate suggestion for drivers to find the correct parking yard for their vehicles based on distance, physical characteristic of vehicle and availability of free spaces in that parking yard. Hence the system predicts the availability and distance drivers will receive the most optimal solution for their parking. This system also provides a service to direct users for the selected parking yards. Control of fuel wastage and emission of Carbon Dioxide are environmentally impacting of the proposed system. And this proposed system will save the users time wastage on streets.

Table of Contents

List of Figures

List of Tables

List of Abbreviation

SARIMA	Seasonal Autoregressive Integrated Moving Average
ARIMA	Autoregressive Integrated Moving Average
API	Application Programming Interface
UI	User Interface
UX	User Experience

1. INTRODUCTION

1.1 Background and Literature Survey

1.1.1 Background

The concept of smart cities and smart technology is rapidly growing with growth IoT. As a result, smart parking systems also introduced to the urban areas. But still, most of the users find it is hard to find the optimal and accurate parking yard to park their vehicle. Parking is an essential task need to be performed by almost everyone who travel away from their residence. As a result, smart parking concept was introduced. Finding an optimal parking yard to park the vehicle is still a hard task for most of the drivers. Day by day the number of vehicles among the street increases and relatively finding an optimal parking yard to park also get increased rapidly. In the survey conducted in the earlier stages of research it was revealed that most users prefer to park in the parking yard as much they concern about the security of their vehicle. And parking yard can be considered as a solution for the limitations with street parking. As the number of vehicles increases, drivers rarely able to find free spot to park in the streets as a result of those drivers use parking yards to fulfill their parking needs. In the survey conducted in early stages of this research, in figure1.1 it was revealed that majority of drivers are using parking yards to park. And only very few expressed that they have not been using parking yards to park their vehicles.

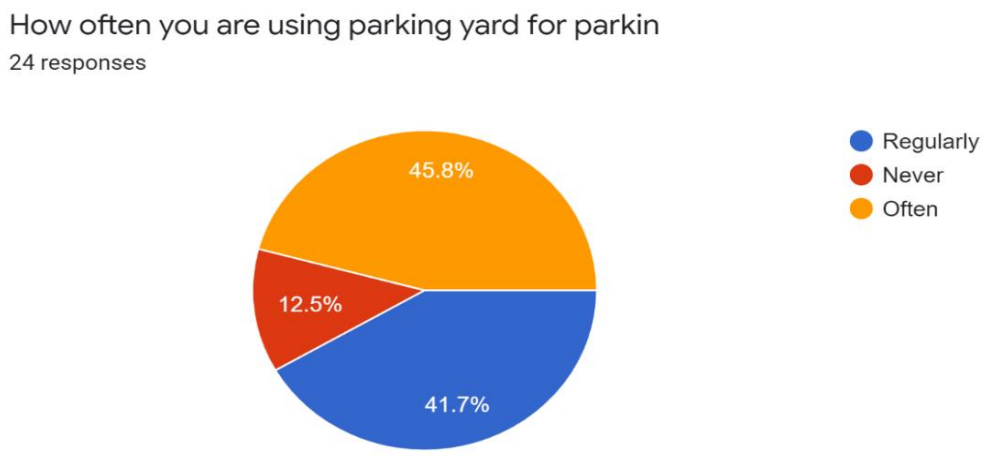


Figure 1.1 - How frequent people use parkin yards

Although majority of drivers are using parking yards to fulfill their parking needs it most of the users are facing difficulties in finding the correct parking yard to park their vehicle. As per the figure 1.2 most of the drivers are facing issue with finding an optimal parking yard to park. Also, it depict that although there are parking solutions available in the society with some reasons they are failed to provide optimal parking suggestions and users are facing issues with parking in their day to day life.

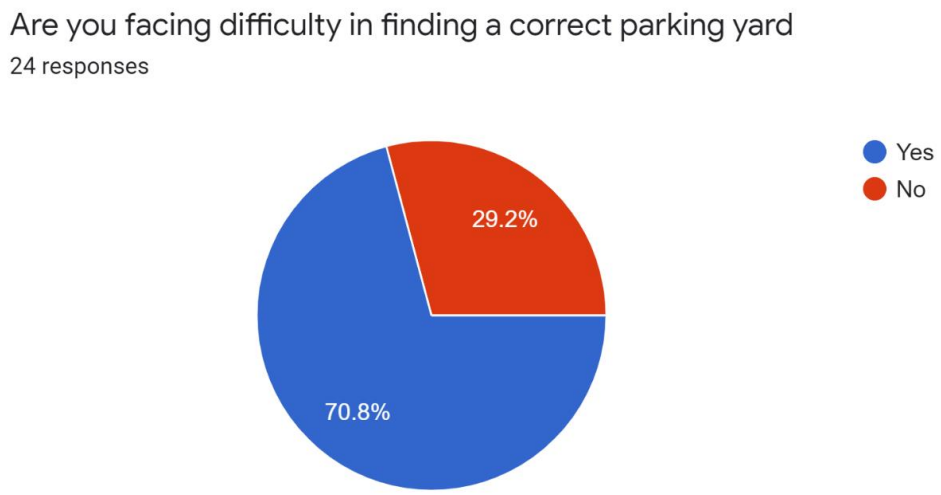


Figure 1.2 - Difficulty faced to find a parking yard to park

Though the smart city concepts developed drivers are still facing the problem of finding an optimal parking yard to park their vehicle. This would affect the society and environment in a negative way. When drivers are unable to find the parking yard to park, they may become stressful, and this will lead to road accidents and unusual traffic situation in urban cities. Also, when drivers travel in the roads reasonlessly to find a parking yard it would waste the fuel and cause air pollution. Drivers may use to park their vehicle unethically and illegally when they could not find a proper parking yard to park.

There are many smart parking solutions available but none of them are addressing the key issue faced by the drivers. As parking is more essential drivers would be provided with most optimal parking yard suggestion to overcome the above mentioned social and environmental issues. Though there are newer technologies

available today the present parking system fails to overcome by providing optimal results [1]. The research conducted by Bhavani, D. S., & Ghalib, M. R. have illustrated that although there are smart parking solutions available, they are failed to address the optimality of the solution. In a smart parking solution parking yard suggestion takes the most prominent functionality and it is the main component of a parking system. Optimization of parking yard suggestion must consider the all the key factors that affect the suggestion. Most of the research conducted by the researchers are addressing only the distance and the availability of a particular parking yard. For a effective parking only those facts would not be enough and it has to be consider some other key features and facts which will be discussed in this paper later.

Effective parking can solve the most problems faced by the people and the environment in busy cities. With the rapid growing rate of population, it is predicted that problems with parking also get increased. A fine solution for these problems would be a smart parking solution using the smart concepts. For a better livelihood of residence in urban cities this smart parking will be affect in a positive way. With the presence of optimal parking suggestion in a smart parking solution it is aimed to solve the social issues present in the urban cities.

Wastage of fuel, environment pollution, sound pollution and stressful environment in urban areas could be minimalized with use of the optimal parking suggestion in urban cities. Most importantly illegal parking and then number of road accident occur annually can be dragged down with the optimal parking suggestion. Prominently it would impact the traffic conditions in urban cities in a positive way. Providing an optimal parking suggestion would impact the society and the environment in a positive way.

1.1.2 Literature Survey

Many research have been conducted to locate the nearest location of user. In research conducted to locate the nearest police station, they have used Global Positioning System (GPS) device to identify the location of user. It works in any weather if the device has a clear line of sight to the satellites [2]. It indicates the GPS technology got a high availability in global. According to many authors GPS is a widely used technology to locate the current position of user, as in current smart world each one got a smartphone on their hands, it will be able to reach many users. A-GPS, Assisted Global Positioning System, works on the same principles as the GPS [2]. The main difference between these is A-GPS obtain the user coordinates via assistant servers while GPS coordinates with satellites directly.

Google Map is a service provide by Google; many researchers use this tool to digitally visualize the locations to the users. Just after the closest location is found, the coordinate of this location is sent on Google Maps [1]. And Google Map API provides different service to integrate with the map. Direction to the location service API is used by many authors to direct user to the destination. Google provides several APIs related to google maps. For this scenario Distance Matrix API which provides the distance and the time to reach data can be used the algorithm. Also google provide the Google Map SDK for both android and IOS mobile operating systems.

Authors have been many Machine Learning algorithms to find the nearest parking location to the user. They have used the user current location coordinates which retrieve by using GPS and A-GPS. In research conducted to identify the nearest medical service provider, they have used algorithm called TOPSIS to get information about nearest medical service provider. TOPSIS (Technique for Order of Preference by Similarity to Ideal Solution) algorithm is a multiple-criteria decision-making algorithm [3].

The Haversine theorem is used to calculate the lengths of two points on the surface of the earth based on latitude and longitude [4]. This research was conducted to find the nearest mosque for a user. Author have used Haversine theorem to identify which is the nearest to the user. Haversine algorithm mostly used to predict the distance between two coordinates, computational cost remail minimum for this algorithm compared to other algorithms used for same purpose. Also, in haversine algorithms it assumes that Earth is spherical but in reality, it is elliptical. Although this assumption is with a minimum error regarding the distance in order to find the shortest distance Haversine still remain as one of the best algorithms with low computational cost.

Ahuja-Dijkstra's Algorithm is to calculate for routing analysis to find the shortest route to reach the nearest facility [5]. In the reading it is noticeable that Ahuja-Dijkstra's Algorithm is mentioned by majority of authors in order to find the shortest path from current location to destination.

In order to predict the occupancy of a parking yard at a given time since parking yard occupancy data possess a seasonality SARIMA can be used as the prediction model since it is widely used for seasonal time series forecasting. Authors of research [6] have compared the timeseries forecasting models and revealed that SARIMA performs better for datasets with seasonality. ARIMA is a model which perform with autoregressive and moving average processes, and SARIMA is advanced with the seasonal package.

As per the above mentioned reading main component of a smart parking system is identifying the most appropriate parking yard to park. By suggesting optimal solution to park, it will be benefit for both users and the environment.

1.2 Research Gap

When predicting the optimal parking yard for a vehicle it is needed to be consider many facts. In the present parking systems researchers have only considered about the distance. As it mentioned in the research A [3] author has built up the parking suggestion using K-Nearest Algorithm. However, in terms of accuracy of the result considering only the distance to the parking yard does not solve drivers' problems with finding a parking yard to park their vehicle. It must be considered the other facts also.

As a result of the research suggesting an optimal parking yard for the drivers it is discovered that following facts are impacting on the optimality and the accuracy of parking yard suggestions.

- Distance
- Availability
- Physical characteristics of vehicle
- Time taken to reach the destination

In the research B [1], research conducted to identify the nearest medical service provider, author have mentioned about the availability term. But in the scenario of smart parking, it has to consider about a new algorithm to find out the availability of free parking spaces in each and every nearest parking yard and also it has be considering the other impacting results on the optimality.

Research C [7], is consider about individual parking slots. In that proposed system they are considering the availability of free slots, but they are not considering about the other main facts. In this scenario also authors are missing the other key facts that need to be consider. It is verdict that most of the authors are mainly considering about the parking yard availability and the distance but in the practical scenario it has

to be consider on the other impacting facts too when it comes to parking yard suggestions.

Also, when we evaluate availability when users arrive at the parking lot without pre-booking, the spaces may be already filled during peak hours. To avoid such an unfavorable situation, we decided to enhance the suggestion by using future prediction as well. In this case, it will make recommendations based on the number of vehicles that may arrive at the parking lot at that time of day. We will present our users with the most accurate and best parking options possible.

Table 1.1 shows the summarized details in tabular format.

Table 1.1 - Comparison of Former Research

Product	Distance	Availability	Physical Characters	Time taken to reach the destination	Optimize suggestions
Research A	✓	✗	✗	✗	✓
Research B	✓	✓	✗	✗	✗
Research C	✗	✓	✗	✗	✓

Pay as You Park Smart Solution	✓	✓	✓	✗	✓

With the comparison in table 1.1 the authors of the previous research have considered mostly on distance and availability but for a better parking the parking yard suggestions need to be accurate and optimal. In order to provide most optimal and accurate parking yard suggestions it must be consider all those facts that impact on parking yard suggestions. In ‘Pay as you Park’ it considered all the facts including distance to the parking yard, availability of the parking yard, physical characteristic of vehicle and the time taken to reach a parking yard. By using the suggestion in ‘Pay as you Park’ for parking yard drivers would be able to gain the accurate and optimal parking suggestions for a better parking.

1.3 Research Problem

In the daily routine of human life people used to go out and they must park their vehicle in a safe environment until they can perform their task. Most users in the urban cities are in habit of using external parking yards. But when we consider the availability of free slots to park their vehicle, users are unaware if the parking yard got free slots. Also, they may not be aware of the parking yards around them or their destination. And also, in some parking yards there may be barriers which prevent the vehicle to perform the parking. If we consider the vehicle height as a character for

some parking yard that vehicle might not be able to enter because of some physical barriers.

There are several parking applications used in the society which suggest a parking yard for users to park. Since urban cities have a huge traffic, when we consider a peek time although it suggested the parking yard but, parking yard may get unavailable when user reach the destinated parking yard. The fact time taken to reach the destination also must be consider when it comes to parking yard suggestions.

Users are facing these issues in their day-to-day life when they going to perform parking. As a result, they must repeat the searching for a parking yard repeatedly until they find a suitable yard to park. It will wastage their fuel and time. Also, it may result in unusual traffic conditions, environment and air pollution. Also, it impacts in the social aspect in negative way. People will used to park their vehicles in unethical and illegal manner. Also, mental wellbeing of the people in those urban cities would affect in a negative way with stress.

So, I have implemented a solution for the above issues considering all the facts to suggest parking yard for a user as my research solution. With the optimal parking suggestions that provided the above-mentioned problems may minimized.

2 OBJECTIVES

2.1 Main Objectives

The main objective of Pay as You Park Smart Parking solution is to provide optimal and accurate parking yard suggestions to users to securely park their vehicle effectively. With this proposed system users can save their valuable time and fuel. And also, with the introduction of optimal parking suggestions; unusual traffic conditions in urban areas will get reduced.

Since smart solution plays a major role in society, when users get used to this system, difficulty faced to park their vehicle will be reduced. This fact clearly demonstrates that drivers can avoid illegal parking in streets, and this will help to reduce the street traffic in urban areas.

2.2 Specific Objectives

- Implement a user-friendly mobile application which support cross platforms to demonstrate the maps and directions. Introduce rich User Experience with the application

- Identify the nearest and available parking yard to the user's destination. As a initial process, have to retrieve the nearest parking yard for user's destination or location as the preference.

- Implement an algorithm to identify the vehicle physical characteristics and filter the option which the vehicle can be used. Assume that vehicle height over than the height of entrance to the parking yard system will remove that yard from the suggestions.

- Implement an algorithm to identify the availability of free spaces in parking yard. This is the mandatory objective which will filter out the available parking yards and suggest to user. Finally, user will receive most accurate and appropriate suggestions.

3 METHODOLOGY

In the very beginning of our research, we found that parking systems in Sri Lanka does not provide optimal parking suggestions for the users. And most of the users are facing hard to find a correct parking yard to park their vehicle. So, we decided to implement a solution for this problem with modern existing technology.

First, we wanted to find out the users' thoughts on the current existing parking suggestions in Sri Lanka, so we done a survey on that, and it proved us that most of the users are unhappy on the existing parking suggestions. In that decided to implement an algorithm with high accuracy and optimality.

Since the solution need a portability and availability, we decided to provide a mobile application to users with the functionality of parking yard suggestions. Flutter is used for mobile application development as it supports cross platform mobile development, and it is currently in the trend of mobile application technology.

- **Get the availability**

  In our research the parking yard availability is captured by the cameras in the parking yard in another component and it provide the live availability of a parking yard. Using the integrated API with node.js it can gain the availability of that time. For get the availability the API will be used.

- **Considering the physical characteristic of vehicle**

  As discussed earlier, sometime physical characteristics of a vehicle may restrict the vehicle from entering or parking in some parking yards. It is identified that length, width and height of a vehicle are characteristic that cause the restriction. As a solution to the problem, we collected those data

from parking yards and store them in the database. In the process of parking yard suggestion through the API services used in algorithm we fetch only the valid parking yards for vehicle. It make sure only the valid parking yards are suggested to vehicle which can perform the parking.

- **Identifying the nearest parking yard:**

  To find the nearest parking yard to user it is needed to be calculated the distance to parking yards and order them to process further. Haversine is a widely used algorithm to calculate the distance of two location points was firstly published by James Andrew. Given the longitudes and latitudes of two points on a sphere, it calculates the great-circle distance between them. It is a specific example of the law of haversines, a more general formula in spherical trigonometry that connects the sides and angles of spherical triangles. Haversine provide us the accurate result with least computational cost.

- **Predict occupancy when user reach parking yard**

  Predicting parking yard occupancy when user reach the destination is a main section of this research. In order to get the prediction, we need to get the time taken to reach the destinated parking yard. For that we will be using the Google map API platform provided by the google. We have discovered that the Distance Matrix API for this functionality which provide us the duration, distance and other matrix. With the use of Distance Matric API then the time is processed and taken the reach time for the destinated parking yard.

It is needed to be predict the occupancy of the parking yard within the time taken to reach the destinated parking yard. Since the parking data consist of a seasonality with it. SARIMA a time series forecasting model is used to predict the occupancy of a parking yard when user reach the destinated parking yard. SARIMA is an extension of ARIMA which support seasonality. SARIMA is widely used in the industry to forecast the univariate dataset with seasonal component. SARIMA fit well for the model since the parking dataset possess a seasonality with it. In order use the SARIMA fist, it needed to be identified the autocorrelation and partial autocorrelation of dataset. This can be achieved by plotting the correlation between the occupancy and that the same measures of X period before. Autocorrelation process a single lag while partial correlation processes a moving average.

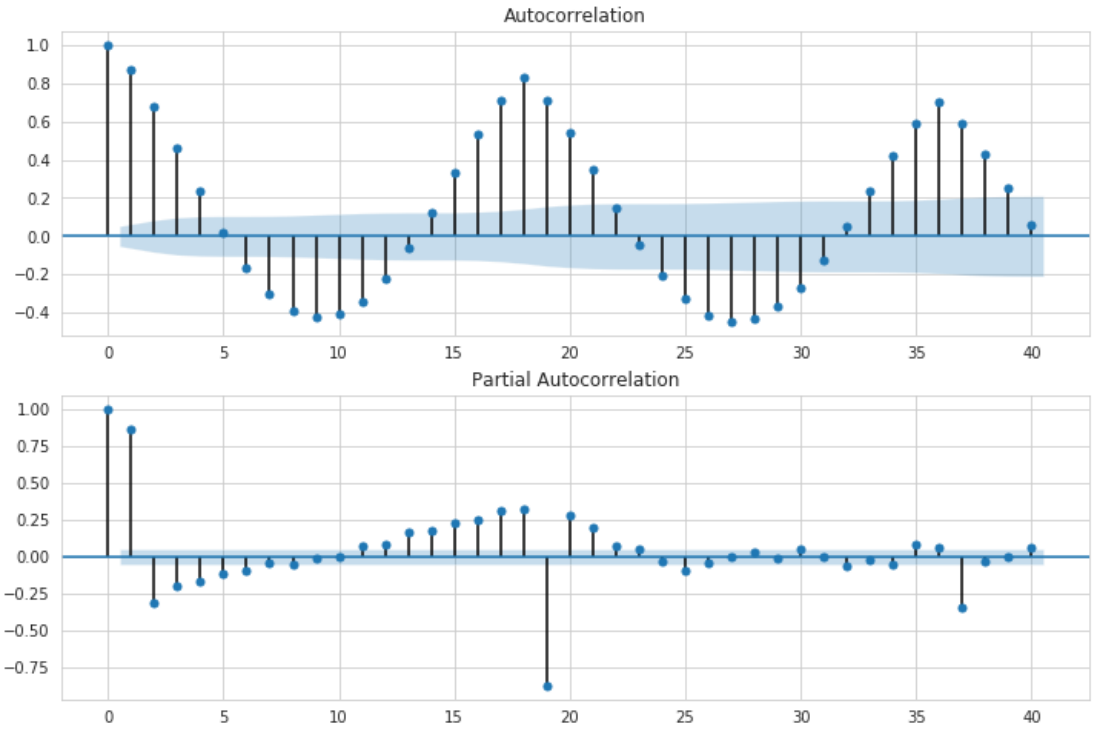


Figure 3.1 ACF and PCF plot

The autocorrelation and partial correlation graphs in Figure 3.1 show that there is a pattern in the data set of occupancy. SARIMA can be applied to this scenario and predict the accuracy by forecasting.

- **Workflow of the component**

As a solution for the issues identified in parking yard suggestions, we come up with an algorithm considering all the key factors that impact on parking yard suggestions. The component flow diagram *figure 3.2* given below demonstrate how the parking yard suggestion algorithm works. With the use of that algorithm, we provide the users optimal parking yard suggestions.

The respective algorithm is deployed with fast API which is a widely using API in the industry. Users' location coordinates and vehicle's physical characters are needed to provide as input parameters and parking yard and location matrix are return as return types. Those return matrix are used in mobile application level along with Google maps route API and GPS services the results are visualized in the mobile.

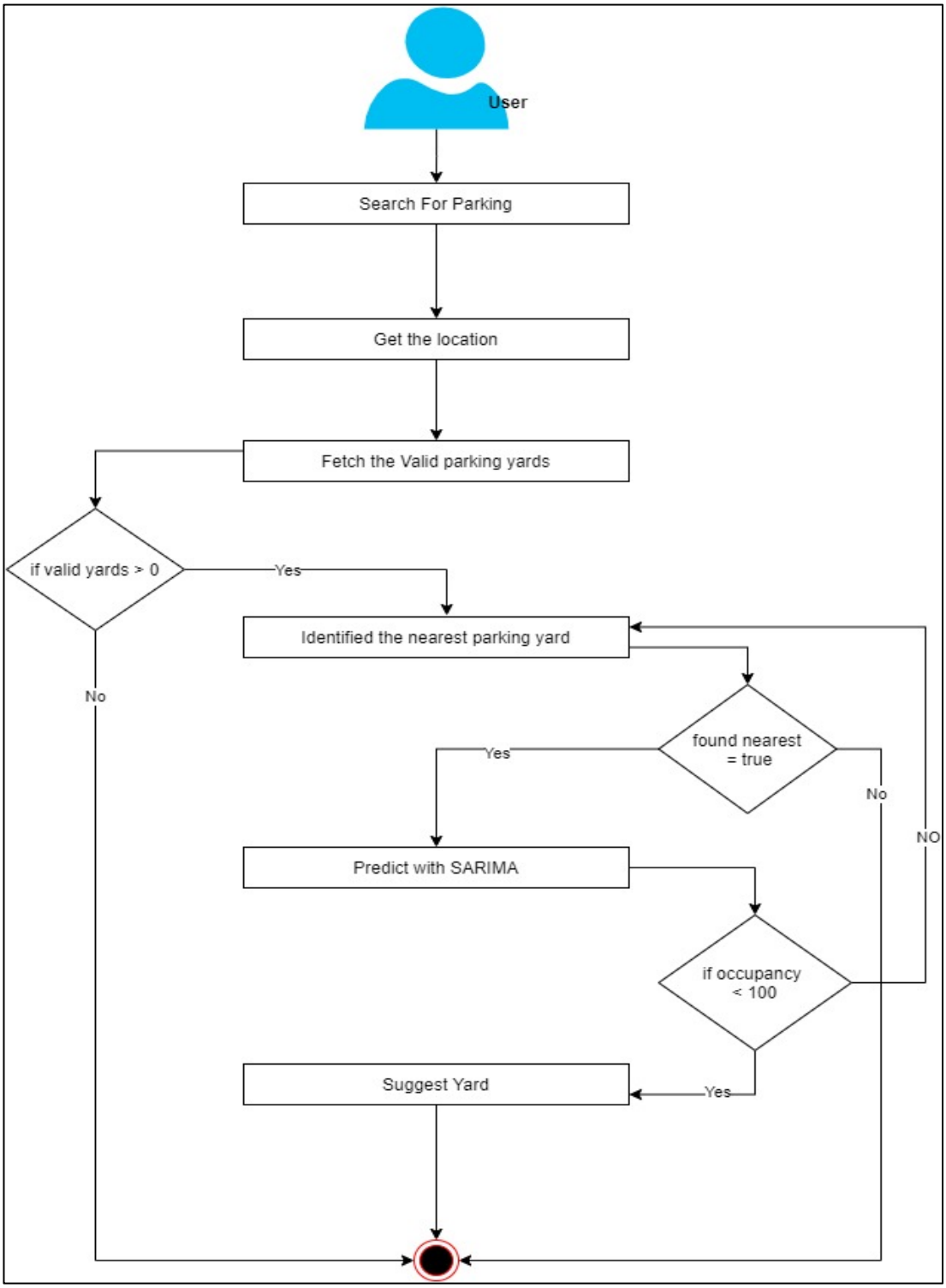


Figure 3.2 Component Flow

- **Component Architecture**

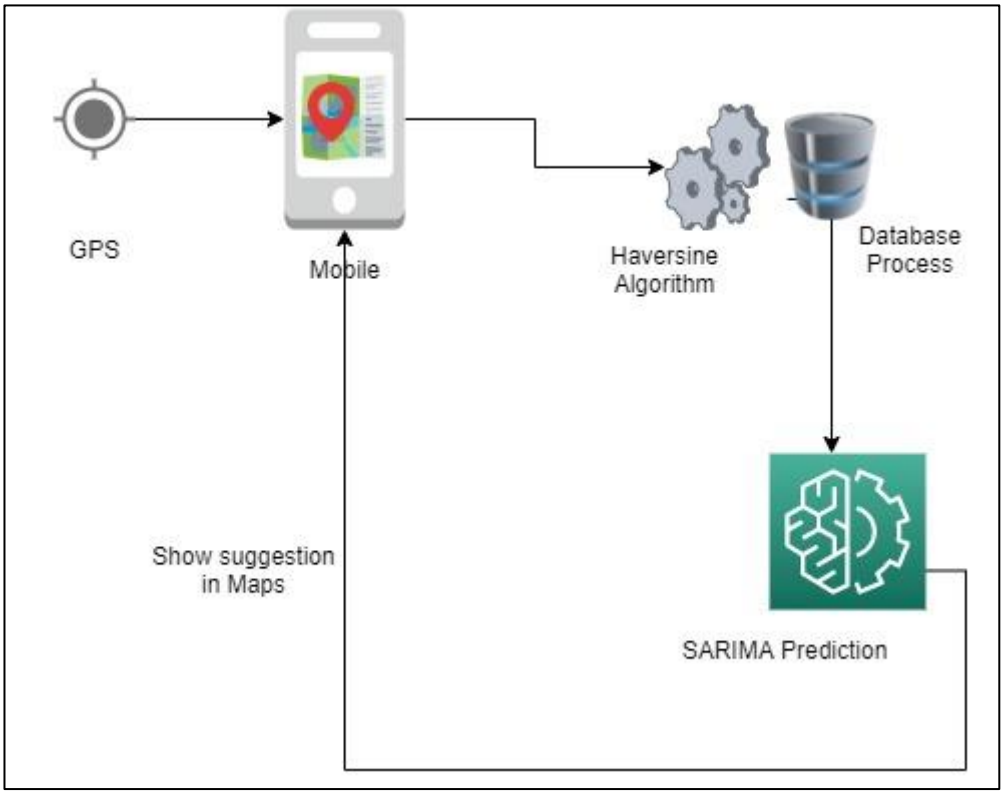


Figure 3.3 Component Architecture

In the figure 3.3 it demonstrates the architecture of the component which describes the communication flow between mobile application and servers.

- **Commercialization of the Product**

Parking is a highly demanded sector in urban cities and most of the users are facing issues with parking in their day-to-day life. With the introduction of optimal parking suggestions, the accuracy of the parking will improve, and it led users for a better parking experience. The mobile application can gain a high demand with the parking yards and application market will improved with efficiency of the product.

4 TESTING AND IMPLEMENTATION

4.1 Implementation

Pay as You Park smart parking solution system have two applications. One for parking yard owners and it is a web application. For the end users we implemented a mobile application with flutter. It supports both android and iOS operating systems. For my research component I have also used mobile application to visualize the suggestions.

Users need a mobile device with active internet connection in order to access the mobile application.

4.1.1 Mobile Application Implementation

Mobile application is developed with flutter which is a widely used cross mobile application development. And it supports widget-based development which is reusable as components. For the ease of development flutter has the feature called hot reload which saves the development time. We do not have to build the whole application again and again after small changes. Flutter will reload the app quickly with function of hot reload.

For the database of whole application Mongo DB is used since it got a high availability and accuracy. Also, it provides object-based database which is useful for the mobile platform. Compared to other cloud providers mongo support flexibility than other.

We have used node.js application for API level development to do operations with Mongo DB. Database operations are executed through node.js using the API.

Express.js used as a framework for the API development. Further, the API project is deployed in Heroku and get the Heroku endpoint.

In order to visualize the location in mobile app I have used the google map API service with flutter and implemented the map widget to demonstrate the map in mobile application. Also, to locate the current position of user Location service is used which is provided by dart. Once the algorithm suggests the parking yard to visualize the directions in map, I have used the Google route API which returns the route matrix and other related data.

Figure 4.1 Initial Map Load

Initial map load in flutter application with Google map API services. Here it has implemented the Google Maps widget provided by google.

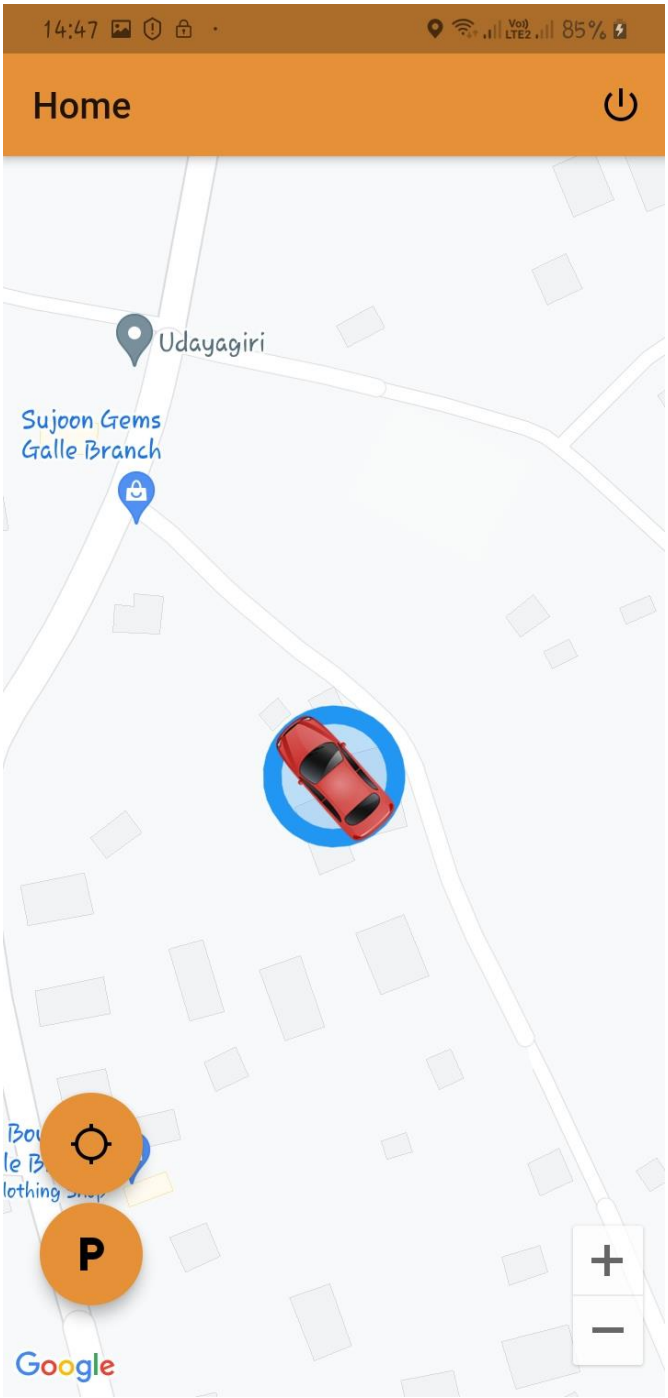


Figure 4.2 Tracking User Location

In figure 4.2 it visualizes the user's current location with the red colored car icon. For this functionality I have used Location service provide by Location Platforms in dart. Also, it tracks user location and update the map when uses change the position. This target is achieved through location tracker service provided by dart. User must grant permission to track the location of the mobile. This function initially prompt to ask the access and then user must grant the access.

```
67
68  void getCurrentLocation() async {
69    try {
70
71      Uint8List imageData = await getMarker();
72      var location = await _locationTracker.getLocation();
73
74      updateMarkerAndCircle(location, imageData);
75
76      marker_active = true;
77
78      /*if (_locationSubscription != null) {
79        _locationSubscription.cancel();
80      }*/
81
82
83      _locationSubscription = _locationTracker.onLocationChanged.listen((newLocalData) {
84        if (_controller != null) {
85          _controller.animateCamera(CameraUpdate.newCameraPosition(new CameraPosition(
86              bearing: 192.8334901395799,
87              target: LatLng(newLocalData.latitude!.toDouble(), newLocalData.longitude!.toDouble()),
88              tilt: 0,
89              zoom: 18.00)));  // CameraPosition
90          updateMarkerAndCircle(newLocalData, imageData);
91        }
92      });
93
94    } on PlatformException catch (e) {
95      if (e.code == 'PERMISSION_DENIED') {
96        debugPrint("Permission Denied");
97      }
98    }
99  }
```

Figure 4.3 Implementation of location tracker 1

```
void updateMarkerAndCircle(LocationData newLocalData, Uint8List imageData) {
  LatLng latlng = LatLng(newLocalData.latitude!.toDouble(), newLocalData.longitude!.toDouble());
  this.setState(() {
    marker = Marker(
        markerId: MarkerId("home"),
        position: latlng,
        rotation: newLocalData.heading!.toDouble(),
        draggable: false,
        zIndex: 2,
        flat: true,
        anchor: Offset(0.5, 0.5),
        icon: BitmapDescriptor.fromBytes(imageData));  // Marker
    circle = Circle(
        circleId: CircleId("car"),
        radius: newLocalData.accuracy!.toDouble(),
        zIndex: 1,
        strokeColor: Colors.blue,
        center: latlng,
        fillColor: Colors.blue.withAlpha(70));  // Circle
  });
}
```

Figure 4.4 Implementation of Location tracker 2

Above figure 4.3 and figure 4.4 are the implementations for location tracking using Location service by dart. Also, it updates the map time to time when use changes the position. I have used custom marker icons and widgets in order provide a better user experience for the users.

And when it comes to visualize the direction, I have used Poly Lines services which lay over the map and visualize the path which is gained by the Google Route API. And it starts the location tracker service along with directions. So user can navigate to the destinated parkin yard with the help of navigations provide by the application.

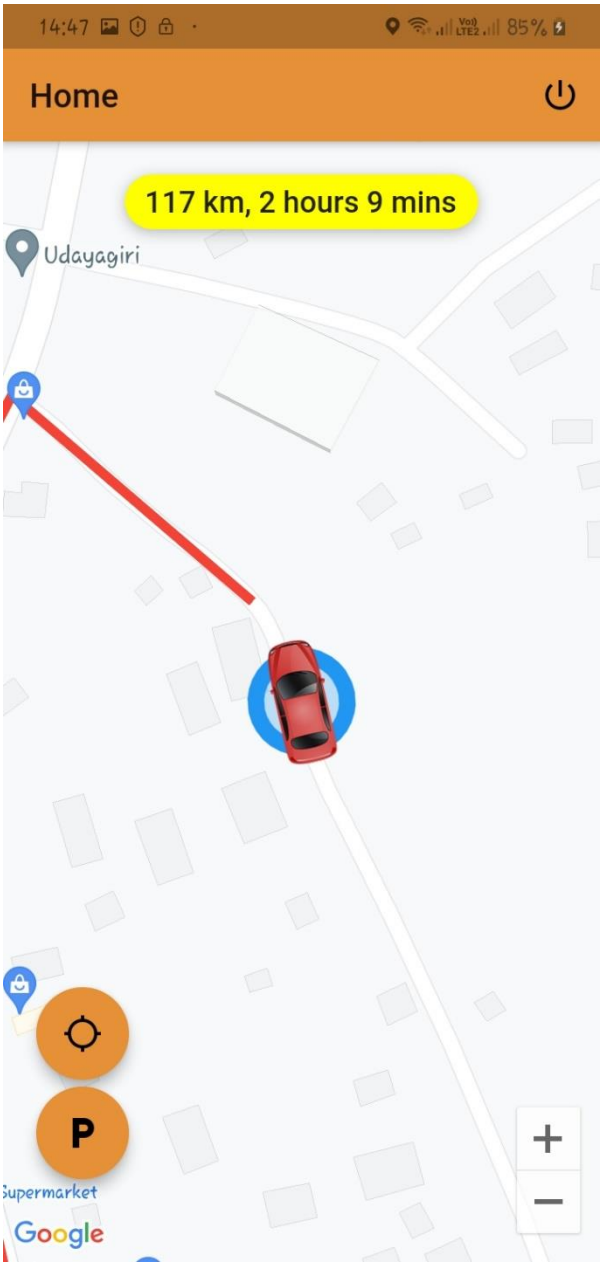
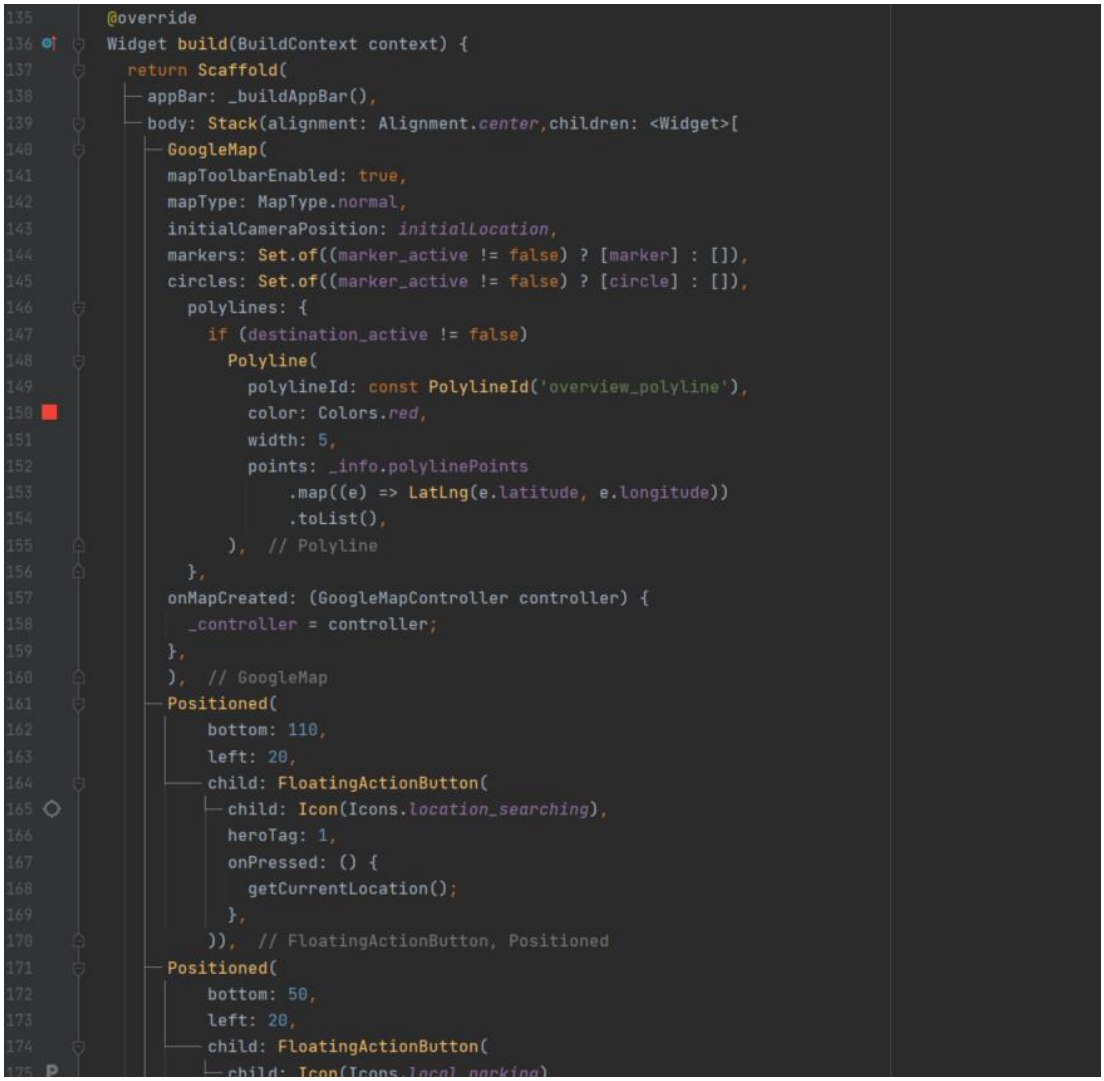


Figure 4.5 Directions Interface

Above figure 4.5 demonstrate the visualization of directions in mobile application with Google Maps and Poly Lines.



```
@override
Widget build(BuildContext context) {
  return Scaffold(
    appBar: _buildAppBar(),
    body: Stack(alignment: Alignment.center,children: <Widget>[
      GoogleMap(
      mapToolbarEnabled: true,
      mapType: MapType.normal,
      initialCameraPosition: initialLocation,
      markers: Set.of((marker_active != false) ? [marker] : []),
      circles: Set.of((marker_active != false) ? [circle] : []),
        polylines: {
          if (destination_active != false)
            Polyline(
              polylineId: const PolylineId('overview_polyline'),
              color: Colors.red,
              width: 5,
              points: _info.polylinePoints
                  .map((e) => LatLng(e.latitude, e.longitude))
                  .toList(),
            ),  // Polyline
        },
      onMapCreated: (GoogleMapController controller) {
        _controller = controller;
      },
      ),  // GoogleMap
      Positioned(
          bottom: 110,
          left: 20,
          child: FloatingActionButton(
            child: Icon(Icons.location_searching),
            heroTag: 1,
            onPressed: () {
              getCurrentLocation();
            },
          )),  // FloatingActionButton, Positioned
      Positioned(
          bottom: 50,
          left: 20,
          child: FloatingActionButton(
            child: Icon(Icons.local_parking)
```

Figure 4.6 Implementation Google Map Widget

Above figure 4.6 demonstrate the implementation of Google Map with the Locations and Poly Lines.

4.1.2 API Server Implementation

To achieve the functionality of communicate with the Mono DB with mobile application we have used a node.js server application project using Express.js. Using

this provide we provide endpoints for each collection in Mongo cluster to perform several operations. And the application is deployed in Heroku and using the endpoint provide by Heroku communication is done between server and client mobile application.

4.1.3 Parking Yard Suggestion Implementation

To achieve the target of achieving suggest the optimal parking yard for a user to park the vehicle I have implemented an algorithm to suggest a parking yard considering the distance, availability, physical characteristics of vehicle and time taken to reach the destinated parking yard.

- **Haversine Implementation**

  Haversine is used for identifying the nearest parking yard for a user. It takes the users location coordinates as inputs and get the nearest for that location. I have implemented the haversine algorithm with python which is widely used for dataset handling. As a first step I fetch all the parking yard in the Mongo DB into a dataset. Which is used in the next stages to process. Among the dataset fetched it will fetch a set of valid nearest parking yards to the user location with the use of haversine algorithm. Initially it will pick up the initial parking yard with shortest distance to the user.

	ID	lat	lon	Name	address	no of slots
0	P1	79.882664	6.914951	LOLC Car Park	21, 25 Chandraleka Mawatha, Colombo 00800	120
1	P2	79.876301	6.883533	Best Western Elyon Colombo	Baseline Road, 102A Kirulapone Ave, Colombo 00500	75
2	P3	79.850030	6.933701	Fort	Fort, Colombo	100
3	P4	79.864816	6.928458	Maradana Railway Station	Jayantha Weerasekara Mawatha, Colombo 01000	225
4	P5	79.870644	6.879944	Tenaga Carparks (Pvt)Ltd	Level 4, 124 Maya Ave, Colombo 00500	300

Haversine Formula

```
from math import radians, cos, sin, asin, sqrt
def dist(lat1, long1, lat2, long2):
    # convert decimal degrees to radians
    lat1, long1, lat2, long2 = map(radians, [lat1, long1, lat2, long2])
    # haversine formula
    dlon = long2 - long1
    dlat = lat2 - lat1
    a = sin(dlat/2)**2 + cos(lat1) * cos(lat2) * sin(dlon/2)**2
    c = 2 * asin(sqrt(a))
    # Radius of earth in kilometers is 6371
    km = 6371* c
    return km

def find_nearest(lat, long):
    distances = parkin_yards.apply(
        lambda row: dist(lat, long, row['lat'], row['lon']),
        axis=1)
    return parkin_yards.loc[distances.idxmin(), 'ID']
```

Figure 4.7 Haversine Implementation

- **SARIMA model Implementation**

SARIMA is used the forecasting model to predict the occupancy at a give time of a parking yard. I have implemented this model using python with Jupiter Notebooks and then build a pickle dataset for the predictions. However, built pickle file is used in Algorithm in order to predict the occupancy when the time is given.

First and initial phase of the model implementation is to analyze the parking dataset and clean the data set as there may be unwanted and improper data which would affect the final predictions. In the analyzation process it includes autocorrelation and partial correlation functions. Since SARIMA is

compatible with dataset with seasonal component those functional tests need to be perform in order to use the SARIMA model. Also, there is a process for test stationary before fitting to the model. In this analysis phase we have make sure that the means and means of standard deviation are stable in the endogenous variable are stable through time. Usually, the Dickey-Filler test is used to test is used to test for stationary.

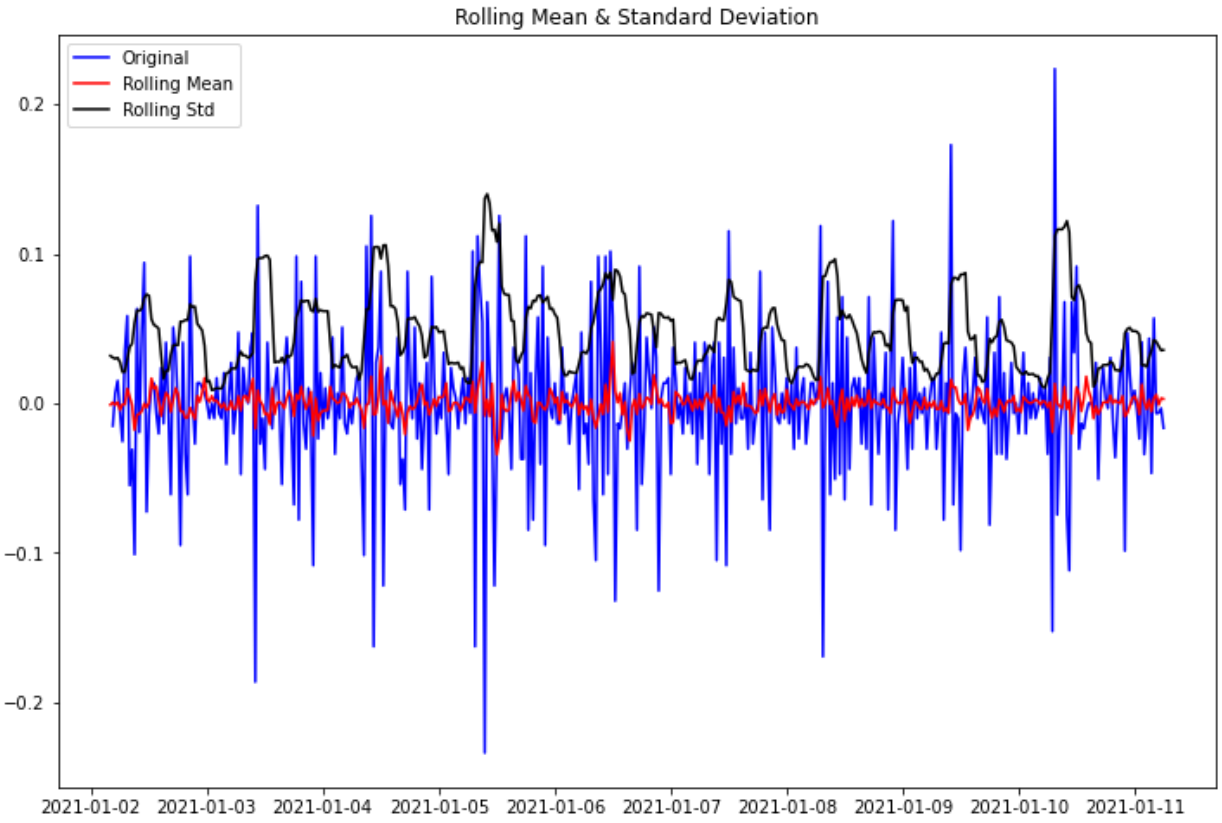


Figure 4.8 Stationary Test

Above figure 4.8 shows the stationary test used for 48 periods of the day. Which means I have split an hour for two half hour sections. In the SARIMA model I have done implementations to fit the model to 48 periods.

```
%%time
# Define and fit SARIMA model
my_seasonal_order = (1, 1, 1, 48)
sarima_model = SARIMAX(train, order=(1, 0, 1), seasonal_order=my_seasonal_order)
results_SAR = sarima_model.fit(disp=-1)
```

```
C:\Users\moham\anaconda3\lib\site-packages\statsmodels\tsa\base\tsa_model.py:524: ValueWarning: No frequency information was pr
ovided, so inferred frequency 30T will be used.
  warnings.warn('No frequency information was'
C:\Users\moham\anaconda3\lib\site-packages\statsmodels\tsa\base\tsa_model.py:524: ValueWarning: No frequency information was pr
ovided, so inferred frequency 30T will be used.
  warnings.warn('No frequency information was'
```

```
Wall time: 9.44 s
```

```
plt.figure(figsize=(16,6))
plt.title('SARIMA Model on Aggregate Data')
plt.plot(train, label='Training Actual Occupancy Rate')
plt.xlabel('Date')
plt.ylabel('Percent Occupied')
y_pred_sar = pd.Series(results_SAR.forecast(steps=len(test)).values, index=test.index)
plt.plot(test, label='Testing Actual Occupancy Rate')
plt.plot(y_pred_sar, color='red', label='SARIMA Predicted Occupancy Rate')
plt.legend()

plt.show()


print('-'*77)
print('SARIMA Model Metrics on Test Data')
print('='*77)
report_metrics(test.squeeze(), y_pred_sar.squeeze())
```

Figure 4.9 SARIMA implementation

After the analyzation and dataset processes using the above code that I have implemented the dataset can be fit to the SARIMA.

As shown in the figure 3.2 the implementations of final algorithm have done using python by integrating all the key facts that need to be consider when suggest parking yard to users. PyMongo is used in the algorithm to fetch the data from mongo DB. Mongo Client is a API provided by pymongo which is used to do operations in collection python. Also, distance matrix API is used to get the distance and time taken to reach the destinated parking yard. Python provides a package called google maps which provide access to the google APIs. Initially we must create a project in google developer console and must enable the APIs we need to use. Using the generated API key, we can use the subscribed API within the project.

```
    dbname = get_database()

    # Create a new collection
    collection_name = dbname["parking_yard"]
    query = {"Capacity" > "Occupancy"}

    parking_yards = collection_name.find()

    # convert the dictionary objects to dataframe
    items_df = DataFrame(parking_yards)

    # see the magic
    #print(items_df)

    items_df['Capacity'] = items_df.Capacity.astype(int)
    items_df['Occupancy'] = items_df.Occupancy.astype(int)
    items_df['MaxWidth'] = items_df.MaxWidth.astype(int)
    items_df['MaxLength'] = items_df.MaxLength.astype(int)
    items_df['MaxHeight'] = items_df.MaxHeight.astype(int)
    items_df['Latitude'] = items_df.Latitude.astype(str).astype(float)
    items_df['Longitude'] = items_df.Longitude.astype(str).astype(float)

    available_yards_df = items_df.loc[(items_df.Capacity > items_df.Occupancy) & (items_df.MaxHeight > user_max_height) & (items_df.MaxWidth > user_max_width)

    # print(available_yards_df)

    found_yard = False
    sugest_yard_id = ''
    yard_id = ''
    latitude = ''
    longitude = ''
```

Figure 4.10 Component Implementation

```
60
61      while found_yard == False:
62          yard_id = find_nearest(user_lat,user_lon, available_yards_df)
63          latitude = available_yards_df.loc[available_yards_df['PID'] == yard_id].Latitude
64          longitude = available_yards_df.loc[available_yards_df['PID'] == yard_id].Longitude
65          # print(yard_id)
66          # print(list(latitude))
67
68          origins = (user_lat,user_lon)
69          destination = (longitude,latitude)
70
71          #print(latitude)
72          # print(longitude)
73
74          result = gmaps.distance_matrix(origins, destination, mode='driving')["rows"][0]["elements"][0]["duration"]["value"]
75          # result = 1331
76
77          # datetime object containing current date and time
78          now = datetime.datetime.now()
79          reach_time = now + datetime.timedelta(seconds=result)
80          print(reach_time)
81
82          occupancy = model.predict(start=reach_time, end=reach_time)
83
84          # print(list(occupancy)[0])
85          # print(type(list(occupancy)[0]))
86
87          if list(occupancy)[0] < 100:
88              found_yard = True
89
90          # found_yard = True
91
92
93      return{
94          "occupancy" : list(occupancy)[0],
95          "yard_id" : yard_id,
96          "latitude" : list(latitude)[0],
97          "longitude" : list(longitude)[0],
98          'found_yard': found_yard
99      }
```

Figure 4.11 Model Implementation 2

4.2 Testing

Testing is a compulsory phase in software development. Since parking is an essential task accuracy and the efficiency of the application must test. I have performed the individual unit testing as well as integrated testing in the application. Since we are considering the privacy of the research, we provided our application with only a limited number of users who used this application to test the location tracking and parking yard suggestions.

Following table 2 demonstrate summary of Unit testing done by our team.

Table 2 Test Results Summary

Function Process	Issues Yes/No
Haversine testing	No
API testing	No
Physical Characteristics Validation	No
SARIMA testing	No
Location Tracking	No
Direction Suggestion	No
Integrated Testing	No

5. RESULT AND DISCUSSION

5.1 Results

My research component is to suggest optimal parking yards to the users. To achieve that I have found the factors that make impact on parking yard suggestions. As per the research conducted, I was able to gain UI/UX and data accuracy.

1. Users don’t need to consider about availability of parking yard
2. Simple UI to improve use experience

As a result of the research, I was able to come up with an algorithm which can derive the suggestions for a user to park the vehicle with:

1. Simple mobile app
2. Less screens
3. Accurate suggestions

Following graph shows the predicted result and trained data in the dataset. The predicted result got a high accuracy with data set and only a minor error.

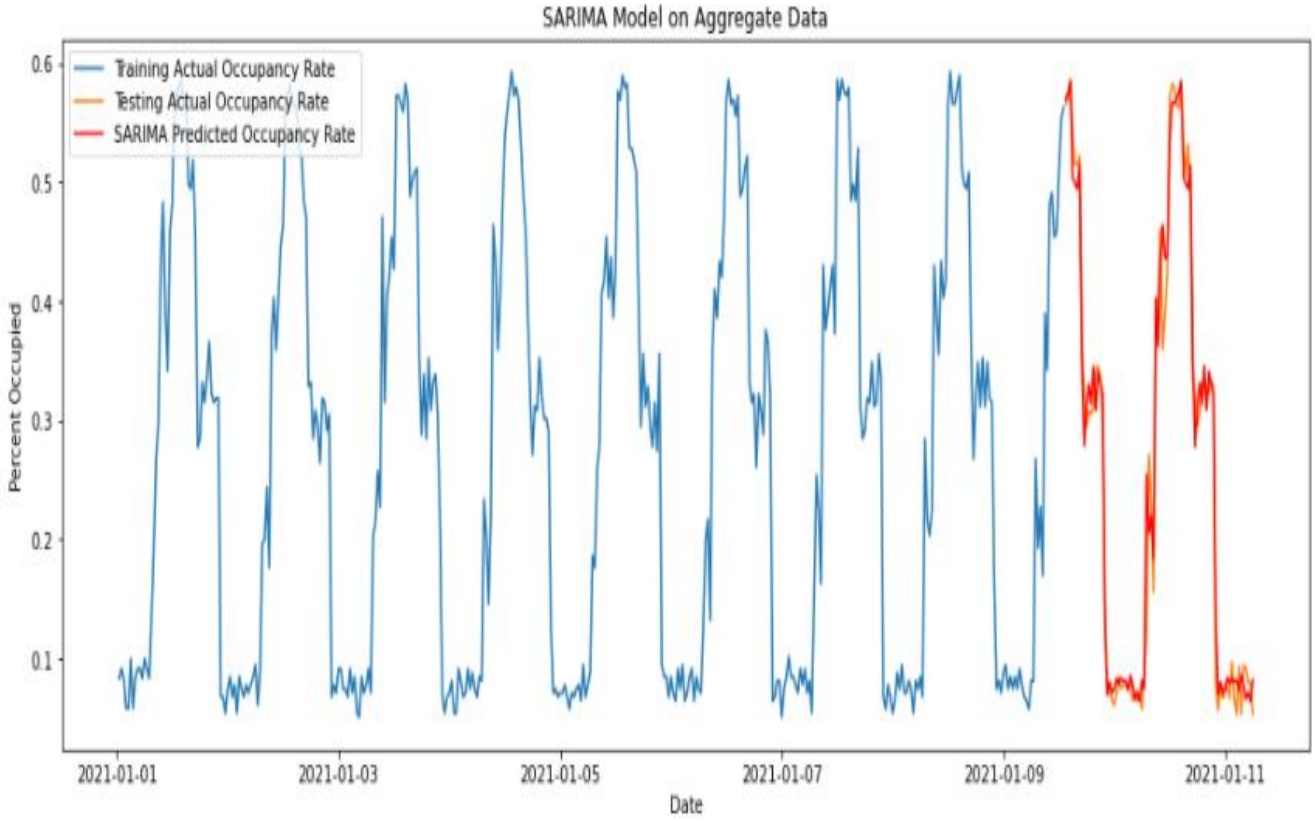


Figure 5.1 SARIMA Graph

As shown in the figure 5.1 the predicted results are mostly like the trained results. Its verdict that the model predicts the occupancy rate mostly like the actual occupancy of parking yard.

```
--------------------------------------------------------------------------------
SARIMA Model Metrics on Test Data
================================================================================
Explained Variance:
        0.9771461256899351
MAE:
        0.01961308780436944
```

Figure 5.2 SARIMA Results

Figure 5.2 demonstrate the model explained variance and mean absolute error. The model got an explained variance above 97% and error rate lower than 0.02. These results verdict the model got a high accuracy.

5.2 Research Findings

Our research is based on a mobile application, and we implemented using flutter so both the android and iOS users can access the application. My aim is to provide optimal parking suggestions for the users to park their vehicles in a safe and secure way.

With research conducted along with experts and end users it helps us to derived below things,

- Better UI/UX experienced theme for the application
- Facts impacting the parking yard suggestions
- Main problems in parking suggestions
- Effect of unethical parking

- Areas where the parking can be improved

With the above listed things and through the research are used in research solution to improve better real time mobile application.

5.3 Discussion

Throughout my research my primary goal was to provide a better parking for the users who faced different kind of problems in their daily routine. There are factors that affect on the parking yard suggestions with my research I was considering deriving a solution which will minimize of those impacts on those parking yard suggestions. Also, I was focused on the real time mobile application development which provide the end users a better UI and UX. When it comes to mobile level application our team focused on using a cross platform mobile application development where we can ignore the native development. It was added as a big advantage to the research. And to evaluate the key factors impact on parking I have used haversine and SARIMA model as it gives the accurate result with least computational cost. We focused on using latest tech stack evolving in the industry which would give advantage to the application level. When it comes to parking yard suggestions, I got attention on what the most users want and what are the problems they are faced I their parking activity. As per the response of users gained by the users, it clearly demonstrated that parking suggestions are not accurate and optimal in the existing parking yard suggestions. So, I focused more on the problems and fact cause the problems and implemented algorithm including the model derived through haversine and SARIMA. Also, we decided to node.js as server application where we gained the endpoints to our mobile application. Node.js API application implemented in Express.js framework which connect with Mongo DB and perform database operations. Mongo DB plays key role in this application since it provides high

availability, accuracy and efficiency. Through the solution that I have implemented for parking yard suggestions; we can use this component in final product to provide users a better parking experience which may result in positive way to environment and society.

6. SUMMARY OF THE STUDENT CONTRIBUTION

Software Implementation and Testing

- Identifying suitable algorithms and correct way to achieve the prediction

  When we consider about machine learning we must find an appropriate data set analyze the dataset carefully before try to fit in to a model. Here I have gone through the analysis phase of the dataset and derived it got seasonal component and decided to fit to the SARIMA model. With that I can easily forecast the occupancy of the parking yard at a given time in the application level with the generated pickle file.

  Also, when it comes to find the nearest yard to the user there were several algorithms including machine learning models. Since I was focused on minimizing the computational cost for that operation Haversine suits the best for that functionality. Its verdict the result with accuracy and minimum computational cost.

- Identifying the key problems faced when it comes to parking

  In the earlier phases of research, I was more focused on finding the issues faced by users with parking. Through those surveys and research, it was able to identify the factors impacting those problems. Through the research solution my primary task was to address those issues.

- Design and implement attractive user interfaces

Most of the users can be attracted to the application with the better UI and UX we provided. So, I have implemented the mobile application section for my component considering the simplicity and performance of the functionality to provide uses a better experience.

- Validate the result and testing

  It is more important to test an application before getting it to the production level. Here I have performed unit testing and integrated testing in my application level and functionalities.

7. CONCLUSION

Using the mobile application to perform parking users can get the optimal and accurate parking yard suggestions to perform their parking in effective and efficient way. They can access the application with a mobile device which runs on android or iOS and get use of the application. And the availability of the application will be high, and users can save their time by using this application. Most of the users are facing issues since they have several issues with the existing parking prediction, and I am addressing those issues in my solution. Through the suggestions predict through 'Pay as you Park' smart parking solutions it could be minimize the issues faced by users in parking yard suggestions. It provides a better user experience in the application it could be able get the trust of users with the product. Also, literally confidence of the parking yard suggestions among the users would be improved.

In order to provide the users a better experience; we have used latest tech stack and focused on the performance of our application. I used express.js framework along with node.js and MongoDB to achieve the API service and it got high availability and efficiency. Through the solution I was considering the distance, availability, physical characteristics of vehicle and time taken to reach the destinated parking yard which are the most impacting factors on parking yard suggestions. In my solution I am addressing all the above factors and provide the suggestions for the end users. Since the application considering on those factors the issues which derived from the impact of those factors can be minimized. With the use of 'Pay as you Park' smart parking solution it can be reduce the issues faced by users with finding a yard to park and parking process can be improved which result the environment and society in positive way.

7. REFERENCES

[1] Bhavani, D. S., & Ghalib, M. R. “Internet of Things Based Smart Car Parking System Using K-Nearest Neighbor Algorithm to Find the Nearest Slot. Journal of Computational and Theoretical Nanoscience,”2018, 15(6), 2040–2045. doi:10.1166/jctn.2018.7403

[2] M. A. P. Chamikara, Y. P. R. D. Yapa, S. R. Kodituwakku and J. Gunathilake, “An Efficient Algorithm to Detect The Nearest Location Of A Map For A Given Theme,” 2013 International Journal of Scientific & Technology Research.

[3] Y. Dian Harja and R. Sarno, “Determine the best option for nearest medical services using Google maps API, Haversine and TOPSIS algorithm,” 2018 International Conference on Information and Communications Technology (ICOIACT), Yogyakarta, Indonesia, 2018, pp. 814-819, doi: 10.1109/ICOIACT.2018.8350709.

[4] Siahaan, Andysah P. U. 2017. “Haversine Method in Looking for the Nearest Masjid.” INA-Rxiv. September 22. doi:10.31227/osf.io/eb3ja.

[5] M. Abinaya and R. Ganesan, "Effective search mechanism for finding nearest healthcare facilities," 2015 Global Conference on Communication Technologies (GCCT), Thuckalay, India, 2015, pp. 534-538, doi: 10.1109/GCCT.2015.7342719.

[6] 1. Fokker ES, Koch T, van Leeuwen M, Dugundji ER. Short-Term Forecasting of Off-Street Parking Occupancy. Transportation Research Record. September 2021. doi:10.1177/03611981211036373

[7] S. Shinde, A. Patil, S. Chavan, S. Deshmukh and S. Ingleshwar, "IoT based parking system using Google," 2017 International Conference on I-SMAC (IoT in

Social, Mobile, Analytics and Cloud) (I-SMAC), Palladam, 2017, pp. 634-636, doi: 10.1109/I-SMAC.2017.8058256.