A PROJECT REPORT ON

PARKING MANAGEMENT SYSTEM USING IMAGE PROCESSING AND DISTRIBUTED APPROACH

SUBMITTED TO THE SAVITRIBAI PHULE PUNE UNIVERSITY, PUNE
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE AWARD OF THE DEGREE

BACHELOR OF ENGINEERING

In

COMPUTER ENGINEERING

Of

SAVITRIBAI PHULE PUNE UNIVERSITY

By

AJINKYA LAKADE	**B120234315**
SHANTANU KOTAMBKAR	**B120234308**
KUNAL MANDE	**B120234320**
TANMAY DESHMUKH	**B120234250**

Under the guidance of

Prof. S. P. BHOLANE

DEPARTMENT OF COMPUTER ENGINEERING
SINHGAD COLLEGE OF ENGINEERING, PUNE-41
Accredited by NBA

YEAR 2016-17

Sinhgad Technical Education Society,
Sinhgad College of Engineering, Pune - 41
Department of Computer Engineering

Sinhgad Institutes

Date:

CERTIFICATE

This is to certify that the project report entitled

PARKING MANAGEMENT SYSTEM USING IMAGE PROCESSING AND DISTRIBUTED APPROACH

Submitted by

AJINKYA LAKADE	**B120234315**
SHANTANU KOTAMBKAR	**B120234308**
KUNAL MANDE	**B120234320**
TANMAY DESHMUKH	**B120234250**

is a bonafide work carried out by him under the supervision of Prof S.P. Bholane and it is approved for the partial fulfillment of the requirements of Savitribai Phule Pune University, Pune for the award of the degree of Bachelor of Engineering (Computer Engineering) during the year 2016-17.

Prof. S. P. BHOLANE
Internal Guide
Department of Computer Engineering

Prof. M. P. Wankhade
Head of Department
Department of Computer Engineering

Dr. S. D. Lokhande
Principal
SCOE, Pune

Acknowledgement

We are extremely thankful to Prof. S.P. Bholane for his expert guidance and continuous encouragement throughout to see that this project rights its target since its commencement to its completion.Collectively, we would also like to thank our project committee members Prof. A. R. Joshi and Prof. S. S. Pawar for their time, suggestions, and for graciously agreeing to be on our committee, and always making themselves available. We would like to express deepest appreciation towards Prof. M. P. Wankhade, Head of Department of Computer Engineering, Dr. S. D. Lokhande, Principal, Sinhgad College Of Engineering and Prof. S. P. Bholane whose extremely valuable guidance supported us in this project.

Abstract

Vehicle parking in today's date has become a major issue in urban areas, due to lack of parking facilities and poor management. Problems arising from lack of parking facilities and poor management include traffic congestion, increased pollution, increased use of fuel .To mitigate these problems, a prototype has been developed with android application, distributed systems, image processing.

The prototype with the help of an android application gives the information regarding availability of car parking to the user directly. Delivering User the information regarding the parking space availability will, thus reduce the time, fuel and efforts invested in search of a parking space. This will also lead to reduction in traffic congestion,ultimately leading to less pollution. A database is maintained in which the availability of parking space is managed .For detection of car parking spaces, image processing is done which will determine the parking availability. This prototype hence,will reduce man power needs and increase flexibility and security. The image processing is done at local server level and only the required information is provided to the central server,thus optimizing the process followed by this prototype. Thus,a distributed parking prototype is developed to mitigate the problems faced by a common man and in the process conserving fuel and helping reduce global warming,pollution as well.

List of Figures

List of Tables

Abbreviations

1. REST - Representational state transfer
2. JSON - JavaScript Object Notation
3. KLOC - Kilo Line of Code
4. RFID - Radio Frequency identification.
5. GPS - Global Positioning System.

Contents

Chapter 1

Introduction

1.1 Background and Basics

- The project " Parking Management System using Image Processing and Distributed Approach " is made with an outlook to solve a civic issue of free parking space management and employ them in urban areas.
- The project considers an distributed approach to solve the issues ,thus minimizing manpower needs and maximizing the optimal efficacy of the system.
- Parking Management has been a issue of concern in today's date due to the increasing ratio of personal vehicles to parking spaces.
- The project works on the concepts of distributed processor, image processing, android application development and Near field communication technology.
- Optimal scheduling of free parking slots is aimed to increase the productivity of parking system.
- The parking slots are fulfilled according to the user requirement, thus ensuring optimal use of parking space and user's ease of access.

1.2 Literature Survey

A literature review is a text of a scholarly paper, which includes the current knowledge including substantive findings, as well as theoretical and methodological contributions to a particular topic. Literature reviews are secondary sources, and do not report new or original experimental work. Most often associated with academic-oriented literature, such reviews are found in academic journals, and are not to be confused with book reviews that may also appear in the same publication. Literature reviews are a basis for research in nearly every academic field. A narrow-scope literature review may be included as part of a peer-reviewed journal article presenting new research, serving to situate the current study within the body of the relevant literature and to provide context for the reader. In such a case, the review usually precedes the

methodology and results sections of the work.

- In the paper, “Trajectory analysis for parking lot vacancy detection system”, the authors provide a vision-based system for real-time management of parking spaces in the case of outdoor parking. Different real-world challenges may face these systems such as weather conditions, luminance variation, perspective distortion and inter-spaces occlusion. In this study, they propose a decisional module based on a tracking approach that determines in real time the state of the parking lots and localise’s the vacant parking spaces according to several extracted attributes. There is a growing interest in the use of vision-based approaches as the surveillance cameras are already installed in most of the parking stations. In addition, a single installed camera can handle an important number of parking spaces simultaneously without extra costs, while providing reliable results.

- In the paper,” Integration of RFID and WSN Technologies in a Smart Parking System ”, a Smart Parking System (SPS) based on the integration of Ultra-High Frequency (UHF) Radio Frequency Identification (RFID) and Wireless Sensor Network (WSN) technologies is presented. The system is able to collect information about the occupancy state of parking spaces, and to direct drivers to the nearest vacant parking spot by using a customized software application. Such application also leverages an NFC-based e-wallet system to allow users to pay for the parking fee .A technologies has been installed on a Central Server in order to manage alert events (e.g. improper use of a reserved space or expiration of the purchased time).

- The paper, ” Scalable real-time parking lot classification: An evaluation of image features and supervised learning algorithms ”, presents a video-based system for cost-effective detection of vacant parking lots, and an extensive evaluation with respect to the system’s transferability to unseen environments. Therefore, different image features and learning algorithms were examined on three independent datasets for an unbiased validation. A feature/ classifier combination which solved the given task against the background of a robustly scalable system, which does not require re-training on new parking areas, was found. In addition, the best feature provides high performance on gray value surveillance cameras.

- The paper,” Sensor Fusion-Based Vacant Parking Slot Detection and Tracking ”,proposes a vacant parking slot detection and tracking system that fuses the sensors of an Around

View Monitor (AVM) system and an ultrasonic sensor-based automatic parking system. The proposed system consists of three stages: parking slot marking detection, parking slot occupancy classification, and parking slot marking tracking. The parking slot marking detection stage recognizes various types of parking slot markings using AVM image sequences. It detects parking slots in individual AVM images by exploiting a hierarchical tree structure of parking slot markings and combines sequential detection results. The parking slot occupancy classification stage identifies vacancies of detected parking slots using ultrasonic sensor data. Parking slot occupancy is probabilistically calculated by treating each parking slot region as a single cell of the occupancy grid.

1.3 Project Undertaken

1.3.1 Problem Statement

To help improve parking management in urban areas this prototype is proposed which will enable proper utilization of parking spaces and will lead to lesser traffic congestion and lesser pollution due to congestion. This will decrease the problems which are caused due to increase in the number of vehicles, and due to lack of management of parking facilities.

1.3.2 Need Statement

- There is a need to promote the commercial parking by private as well as public sector. . Government is taking efforts to reduce the traffic congestion .
- There has been the construction of two or more storey parkings at various places in the city .
- Also the expansion of roads are done to avoid traffic congestion. There has been promotion of bus services in city. Yet, still there are problems related to parking. To solve this parking problem, government needs to develop parking zone rather than restrain policies.
- There is a need of an application that could let the people know where to park their vehicles before hand to avoid the traffic congestion. In addition, the app would also help them discover the nearest parking area during their stop.

1.3.3 Impact Statement

- Since there are many cases when vehicles are parked on roads, so if there are proper parking places then traffic problem would be less. Even if there are less parking slots, if people are aware of the free space for the parking then they would go and park in that free space.
- Thus, an application would guide the vehicle owners in urban areas towards the free spaced parking slots. This would avoid the traffic congestions in the roads.
- Also, due to proper parking system, the government could charge the vehicle owner and it could generate the income that could further be used to build another parking area and maintain the existing ones.

1.3.4 Scope Statement

The project scope extends to:

- Providing and facilitating increase in efficacy to parking management system.
- Decreasing traffic congestion in urban areas
- Help reduce global warming
- Save time by reducing search time of free parking slot.

Chapter 2

Project Planning and Management

2.1 Detail System Requirement Specification (SRS)

Software Requirements Specification for Parking Management System using distributed approach

2.1.1 Introduction

Purpose

Purpose of this system is to efficiently allot parking spot to the user. With the help of this system, global warming and pollution levels can be decreased. Also it will help to efficiently locate a parking slot.

Conventions

This document contains all the requirements for the implementation of this model. These requirements are denoted for a small simulation of a larger model that can be deployed on a large area. Larger model and larger implementation of this model will require more number of requirements.

Intended Audience and Reading Suggestions

This requirement document contains general information about the project functions,features and special technologies. It describes in detail all the needs to work properly and with safety.
This document is intended for:

- Developers: in order to be sure they are developing the project that fulfills requirements provided in this document.
- Testers: in order to have an exact list of features and functions that have to respond according to the requirements and provided diagrams.
- Users: in order to get familiar with the idea of the project and suggest other features that would make it even more functional .

Product Scope

This software is developed to give information of parking to a user via an android application. Information will contain specific availability of parking slots at the specified destination. This will increase the parking space productivity and reduce the unnecessary traffic congestion at the parking spot.

2.1.2 Overall Description

Product Perspective

This product is a new idea to ease parking management. This product with proper amount of hardware can be employed to a bigger system. This bigger system can be employed to entire city for managing parking systems.

Product Functions

These are following functions used in the proposed model.

- Login Authentication.
- Display of result according to user.
- Establishing connection to central server.
- Detection and evaluating empty parking slots.
- Independent processing on distributed server.
- Scheduling is done according to user's time preference.
- Establishing network of central and distributed servers.

User Classes and Characteristics

The following are the User Classes.

- Any type of user who require parking availability of a specific destination can use this model.
- Government or private institution can use this system to analyze and maintain parking system.

Operating Environment

Platforms on which the software will operate are Android, Ubuntu, Linux-Fedora. Android Application is to be created on android platform and the image processing part is to be done on Fedora.Libraries like OpenCV is used for image processing. Languages such as Java, MySQL are used for database and connection. GPS is used for user location identification for project to be implemented.

Design and Implementation Constraints

- User should have mobile with android operating system.
- User's mobile device should be equipped with Gps facility.
- Processing time.

User Documentation

- User manual.

Assumptions and Dependencies

The project is dependent on Image Processing and the video feed taken from a camera, some faults or discrepancies which could lead to false computing of data. An easy tap payment system has been added through NFC in future scope which would enable user for easy payment just by a tap of his/her NFC enabled handset.

2.1.3 External Interface Requirements

User Interfaces

1.User Login Page

- Username
- Password
- Login(Button)

2. User Query Page

- Destination requirement
- Time specification
- Search(Button)

3. Display result Page
4. Connection to the central server(button).
5. Display result page

Hardware Interfaces

Hardware interfaces for the android application will run only on android operating system. Other applications can be created on other operating system like Apple's IOS, Windows Linux, etc. For creation and connection of local servers to the main servers will require network. For a larger model, the network and servers will increase. Android application and central server will have database which will be constantly updated.

Software Interfaces

Android application will be connected to the database and this database will be connected to central server. Central server is connected to local servers, which will update database regarding the parking availability. Android supported version will comply with MYSQL. Distributed programming concept is used to do the image processing on distributed machines and only the data is to be sent to the central database.

Communications Interfaces

For communication of the mobile devices and other servers Ethernet is to be used. NFC connection is used for storing user information in the database and then this connection determines user's car's parking slot. Other communication includes connection of central server to local server.

2.1.4 System Features

This system is grouped according to the mode of operation as follows.

Image processing car detection

- Description and Priority:
  Image processing is an easy way to detect cars and update data in the central database. This is the higher priority feature. The benefit of this feature is much larger and the cost of this feature is reasonably low. Only risk of this feature is that it can sometimes may not work as intended.
- Stimulus/Response Sequences:
  This feature will be triggered when a user parks a car and is therefore used to detect the user's car. In response it will generate an entry in the central database for the parking spot.
- Functional Requirements:
  This feature will require image processing algorithm specifically for the car detection. This will include Java and Open CV for detection of car in the parking space.
    - Open CV

Android Application

- Description and Priority:
  With the help of android application, user will be able to determine the destination of the place where he intend to park his vehicle. This application will give proper number of the available parking spaces in that area. When the user arrives to its destination, user will

connect its phone to the parking space. This will store the car's parking spot in the user's phone.

- Stimulus/Response Sequences:
  This feature will be take user's destination address and the time for which he want the parking space. Application will give parking availability space of the specified destination. User will connect to the network and will get in response the information of the parking space.

- Functional Requirements:
  This will require Android smart phone which can run specified version of android application and this android devices has gps facility in it.

    - Android

2.1.5 Other Nonfunctional Requirements

Performance Requirements

Performance requirements for this model include the time user can connect to database via NFC. The android application should connect the database or it will not be efficient and feasible. Image processing should comply with timings of the user's parking vehicle at the specific spot or it will lead to discrepancies.

Safety Requirements

This model will decrease the amount of toxic pollutants which are emitted by vehicles due to unnecessary congestion at the traffic place.

Security Requirements

User need to login to the android application and therefore the user's data should be kept secured. User's car information and other information need to be kept secured.

Software Quality Attributes

This software will increase the efficiency of usage of the parking system. Proper implementation will increase adaptability to the any user and the any parking space. This model will decrease traffic congestion and therefore will decrease the pollution in the air due to vehicle emission.

2.1.6 Other Requirements

Apart from the requirements stated above, Permission from parking institution and other government bodies is required to implement this proposed model.

2.2 Project Process Modelling

Project process modelling for the parking system is given as below.

2.2.1 Project Scheduling

Time Line Chart

Table 2.1: Time Line Chart

No.	Milestone	Description	Timeline	Remarks
1	Literature survey	Study of existing system	August 2016	Detailed document of IEEE research papers to be ready for each requirement.
2	Planning	Complete specification of the system	September 2016	Complete planning of the exact content of project to be decided.
3	Design and Feasibility of Design	Identify each modules	October 2016	The designing of UML and interface designs must be ready.
4	Platform for Development	Selection of tools for implementing code and language	November 2016	Proper software tools are selected and development environment is created.
5	Development	Code for the system	December 2016	Develop the different software modules and connect them.
6	Testing	Testing of the Developed modules	March 2017	Independent and group testing of the software modules.
7	Report writing	Writing the final report of the project	April 2016	Documentation and report of the project to be written.
8	Final Review	Final demo of the project	April 2017	All the requirements are fulfilled and the final demo is delivered.

2.3 Cost and Efforts Estimates

2.3.1 Constructive Cost Model (COCOMO)

COCOMO model is a single-valued, static model that computes software development effort (and cost) as a function of program size expressed in estimated lines of code (LOC).Like all estimation models , the COCOMO model requires sizing information.The information can be specified in the form of :

- Object Points
- Function Points(FP)
- Lines of source code(KLOC)

For our project the sizing information in the form of lines of source code (KLOC) is used.

Type of Component	Complexity of Components			
	Low	Average	High	Total
External Input=2	3	4	6	8
External Output=4	4	5	7	20
External Inquiries=2	3	4	6	8
Internal Logic Files=11	7	10	15	110
External Interface Files=8	5	7	10	56

Figure 2.1: Account Management Operational Diagram

Development Mode - Semidetached

A development project can be considered of semidetached type, if the development consists of a mixture of experienced and inexperienced staff. Team members may have limited experience on related systems but may be unfamiliar with some aspects of the system being developed. The coefficient values in semidetached mode are as, $a = 3.0$, $b = 1.12$, $c = 2.5$ and $d = 0.35$

Calculations

1. **Development Effort**

   Source Lines of Codes (approx.) := 6000

   Kilo Delivered Source Instruction ($KDSI$) = $SLOC/1000 = 6000/1000 = 6$ $E = a * KDSI^b$

   $E = 3.0 * 6^{1.12}$

   $E = 22.31$

2. **Efforts and Development Time (TDEV)**

   $TDEV = cMM^d$

   $TDEV = 2.5 * 22.31.52^{0.35}$

   $TDEV = 7.4$

3. **Staffing Size (N)**

   $N = Effort/Duration$

   $N = E/TDEV$

   $N = 22.31/7.4 = 3.01 = 3(approx.)$

2.3.2 FP Based Estimation(Function Point)

Factor .	Value .
Backup and Recovery Data Comunication	2 5
Distributed processing function	4
Performance Critical	1
Operational environment	3
On-Line Data Entry	1
On-line update	1
Master files updated on-line	4
Input,Output and files complex	3
Internal processing complex	3
Code designed for reuse	4
Application designed for change	4
Complexity adjustment factor	4
Multiple installation	3

2.3.3 Cost Estimate

As we are working on an open source software, and partly hardware Independent software there is hardly any finance needed.The entire system was implemented by a team of four people working for 8 months.

2.4 Risk Management

Risk management is the identification, assessment, and prioritization of risks.

2.4.1 Risk Analysis

There are quite different types of risk analysis that can be used. Basically, risk analysis is used to identify the high-risk elements of a project in software engineering. Also, it provides ways of detailing the impact of risk mitigation strategies. Risk analysis has also been found to be most important in the software design phase to evaluate criticality of the system, where risks are analyzed and necessary counter measures are introduced. The main purpose of risk analysis is to understand risks in better ways and to verify and correct attributes. A successful risk analysis

includes important elements like problem definition, problem formulation, data collection. Risk analysis is useful in many situations:

- When you're planning projects, to help you anticipate and neutralize possible problems.
- When you're deciding whether or not to move forward with a project.
- When you're improving safety, and managing potential risks in the workplace.
- When you're preparing for events such as equipment or technology failure, theft, staff sickness, or natural disasters
- When you're planning for changes in your environment, such as new competitors coming into the market, or changes to government policy.

2.4.2 Risk Management Analysis

If there is a possibility that the achievement of a goal is harmed, prevented from occurring or suffers negatively due to the occurrence of uncertain events, we call it the risk.These so-called uncertain events can be caused by different factors. An efficient risk management analysis should be able to attend to every one of them to be able to identify them promptly in each of the listed cases:

Personel Risks:

Caused by a lack of Knowledge about technology and training to perform functions. There is a possibility that errors are intentional, this is the result of the dubious conduct. The main risks from personal issues are:

- Unintentional; resulting in omission or negligence.
- Cannot perform task because lack of ability.
- Lack of time management.

Process Risks:

The occurrence of internal process deficiencies like inadequate performance indicators, inefficient controls, modeling failures and an inability to abide by the current laws.

Systems risks:

Arising from inadequate, poorly structured or defective IT systems. Some examples:

- Intermittent networks
- Server crash
- Physical damage to data storage components
- System obsolescence
- Improper maintenance
- Power outage from internal causes
- System slowdown
- Security holes

2.4.3 Risk Management w.r.t NP Hard Analysis:

- In rural area most of the time Internet will not be available so our system may not work.
- If reviews not available and false review are there then systems results will fails.
- If provide wrong input then system will show wrong output or it may fail.

2.4.4 Risk Identification:

- System may fail during review database.
- Results may fail.

2.4.5 Risk Analysis Table

The risks for the Project can be analyzed within the constraints of time and quality.

Table 2.2: Table 1.1

ID	Risk Description	Probablity	Schedule	Quality	Overall
1	Internet connection not available	Low	Low	High	High
2	False review	Low	Low	High	High
3	Incorrect input	Low	Low	High	High

Table 2.3: Table 1.2

No.	Probablity	Value	Description
1	High	Probability of occurrence is	>75%
2	Medium	Medium Probability of occurrence is	26-75%
3	Low	Low Probability of occurrence is	<25%

2.4.6 Overview of Risk Mitigation, Monitoring, Management

Following are the details for each risk.

Table 2.4: 1.3

Risk ID	1
Risk Description	Change of Requirements
Category	Software requirement risk
Probability	Low
Impact	High
Response	Mitigate
Risk Status	Occurred

Table 2.5: 1.4

Risk ID	**2**
Risk Description	Human Errors
Category	Software scheduling risk
Probability	Low
Impact	High
Response	Mitigate
Risk Status	Identified

Chapter 3

Analysis and Design

3.1 IDEA matrix

Idea Matrix		
Increase	• Providing flexibility for users for usage of various no. of parking. • Providing ease of use for accessing parking facilities by android app.	• Flexibility • Usability
Improve	• Improving the efficiency of parking system by scheduling parking slots. • Improving precision for use of whether parking is available or not.	• Effectiveness • Precision
Decrease	• Using local server system for decreasing latency for updating parking slot details. • Decreasing Traffic congestion . • Decrease global warming by reducing carbon emission. • Decrease sound pollution.	• Latency • Congestion • Global warming • Pollution
Develop	• Developing an android application for handy use. •Developing a network between local and central servers. •Developing a database for parking information.	• Application • Network • Database
Discovery	•Using image processingfor sensing free parking slots	•Detection
Evaluate	• Evaluating and estimating filling rate of a parking system. • Calculating and stating final cost for customer.	• Fulfillment rate • Cost
Eliminate	• Eliminate user problems for search of parking space usually in peak hours. • Use of technology to eliminate man-power needs.	• Inconvenience • Human Resource
Exploration	• Exploring parking spaces using image processing • User can explore various parking systems according to his destination needs.	• Availability • Enquiry
Analysis	• Analysis of usage pattern and storing information in database. • Analysis of frequency of user	• Usage pattern • Frequency
Allocation	• Specific Parking slot is allocated to the user. •Due to allocation of parking spaces, the parking spaces are used efficiently.	• Categorization • Efficiency
Adaptation	• Using this adaptive approach i.e. image processing this system can be implemented on various parking system.	• Compatibility.

3.2 Mathematical Model

Let S be the solution set for the given problem statement.

S={Input,Function,Output,DD,NDD,Success,Failure}.

Where,

Input =Input to the System.

Function =Functions of the system.

Output =Output of the system.

DD =Deterministic Data.

NDD =Non-Deterministic Data.

Success =Success cases for the System.

Failure =Failure cases for the system.

1. Input ={Username,Password,User_query}

    - Username = user_id.
    - Password = user_password.
    - User_query = {Set_destination,Set_time}
        a. Set_destination = value of destination selected by user .
        b. Set_time = value of time t for which user needs destination information .

2. Function = {Login_auth, user_InputResultDisplay, Network_connection , emptySlot_detection , Distributed_processing , Scheduling_slot, network_ofServers}

    a. Login_auth =Authentication of user account.

    b. User_InputResultDisplay =Accepting user input and displaying results accordingly

    c. Network_connection =Connection establishment to central server

    d. emptySlot_detection =Detecting and Evaluating empty parking slots.

    e. Distributed_processing =Independent processing on distributed servers.

    f. Scheduling_slot =Scheduling is done according to users time preference

g. network_ofServers =Establishing network of central and distributed servers

3. Output = { display_availability, display_errormsg }

    a. Display_availability =display no. of parking slots available.

    b. display_errormsg =display error message if any error occurs.

4. DD={Username, Password, User_query, slot_availability}

5. NDD={default_Destinationvalue, default_Timevalue}

    a. default_Destinationvalue -> Default value set for destination if destination_value is not defined.

    b. Default_Timevalue->Default value set for time if time_value is not defined.

6. Success

    a. Successful user login .

    b. Displaying result according to user query.

    c. Successful connection establishment of user to central server .

    d. Successful detection and evaluation of free parking slots.

    e. Successful network establishment between central and distributed servers.

    f. Appropriate error messages in case of invalid input.

7. Failure

    a. Android app crashes.

    b. Hardware faults.

    c. Network establishment failure.

    d. Faulty scheduling of parking slots .

    e. Not displaying required results for user query.

3.3 Feasibility Analysis (NP Completeness Analysis)

The problem is to detect a vehicle as an object and then determine availability of parking spaces. The main objective of our project is to provide a real-time solution for vacant parking lots detection in crowded cities. The proposed solution should be easily adapted with different parking. Providing that our approach should be adapted for different parking dispositions, this system is introduced as real-time approach for detecting the vacancies of the parking lots independently of a specific architecture of the parking. Since camera provides fixed scene videos, this model is proposed to detect and track the objects in motion. This allows to instantly analyze its trajectory, to detect its actions in the parking based on a temporal integration and to calculate its degree of occupancy once it is inside a parking space. Besides the objects in motion detection and tracking, we locally explore the state of a parking space based on the speeded up robust features (SURFs) speeded up robust features extraction in order to strengthen the approach of event detection.

3.3.1 Trajectory Analysis

Once the objects that are in motion are extracted, it is important to follow each one separately in order to analyze its trajectory over time. This analysis is important for the understanding of each object trajectory through the parking. Other than the actual position, a trajectory T at an instant t can be characterized by a descriptor vector presented with T=(Pt , Dt , Vt) where Pt is the actual position of the tracked object at the instant t and Dt represents the direction of motion of the object. Whether it is passing from the street to the parking spaces or the inverse; Vt is the measure of velocity of the object at t.

3.3.2 Parking Events Detection

A car's position pt at an instant t is defined according to the center of the bounding box surrounding it. We can verify if it is overlapping with our extracted parking model or not according to its position. At an instant t, a vehicle state st can be 'Out' of all parking spaces or 'In' a specific parking lot. To calculate this state, we present the on-street surface of each parking space as a set of four segments (L1, L2, L3, L4). The position of a given vehicle at the coordinates (x, y) to the line L1(p11, p12) is calculated as

d2, d3 and d4 can be calculated similarly. The vehicle state st is 'In' and the tracked vehicle is considered crossing a parking on-surface only if these four measurements are positive. Since

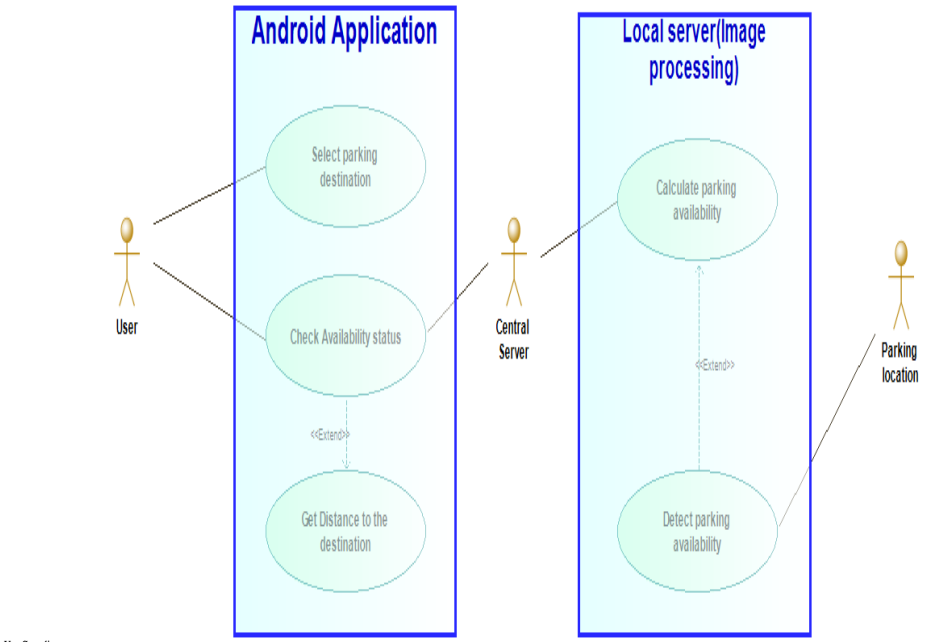
the loop for pattern matching runs n times, the complexity of the algorithm is O(n). Hence, the problem runs in polynomial time complexity. The algorithm falls in P. Hence, the given problem is NP.

3.3.3 Use-Case Diagrams

A Use Case diagram is a type of behavioral diagram defined by the UML created from a use case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goal represented as use case and any dependencies between those use cases. It is a type of diagram that shows a set of use cases, actors and their relationships. It should have a distinct name. It commonly contains:

- Use Cases
- Actors (Primary and Secondary)
- Dependency, Generalization and association relationships

Use Case



Use Case diagram.png

Figure 3.1: Parking Management Use case

3.3.4 Activity Diagram

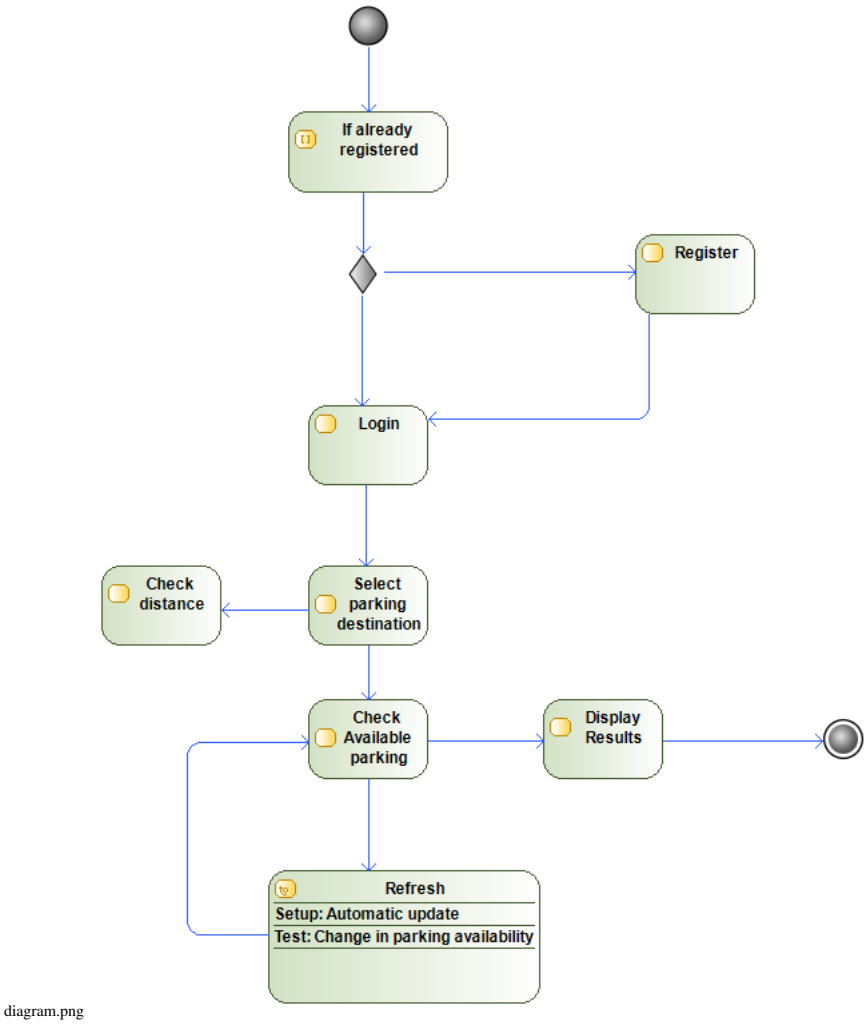
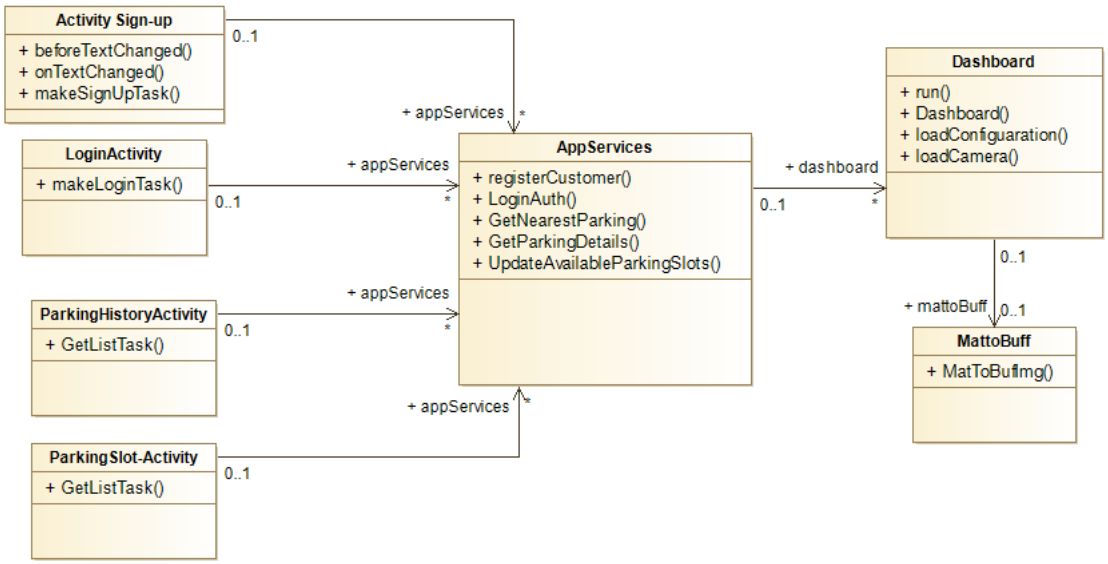


diagram.png

Figure 3.2: Parking System Management Activity Diagram

3.3.5 Class Diagrams



Class diagram.png

Figure 3.3: Class Diagram

3.3.6 Sequence Diagrams

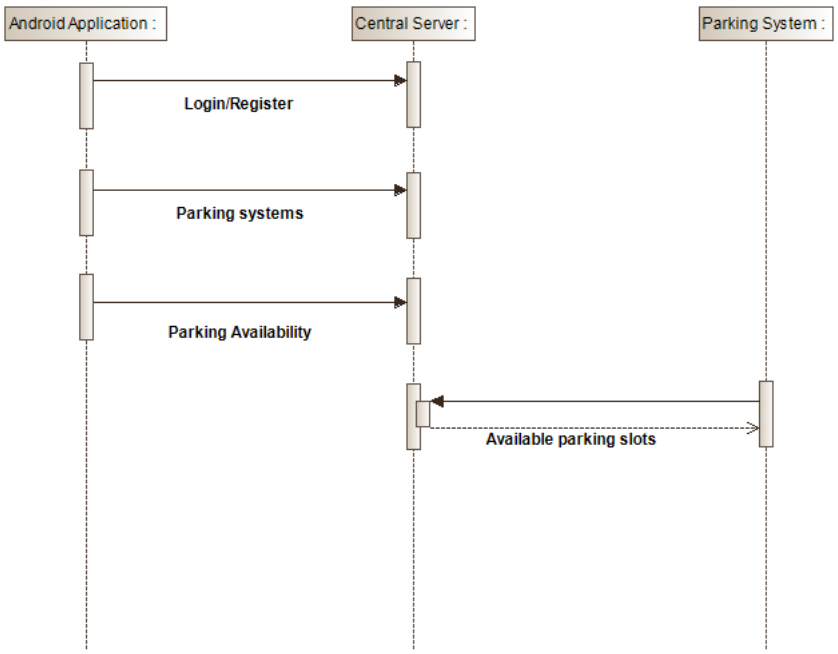
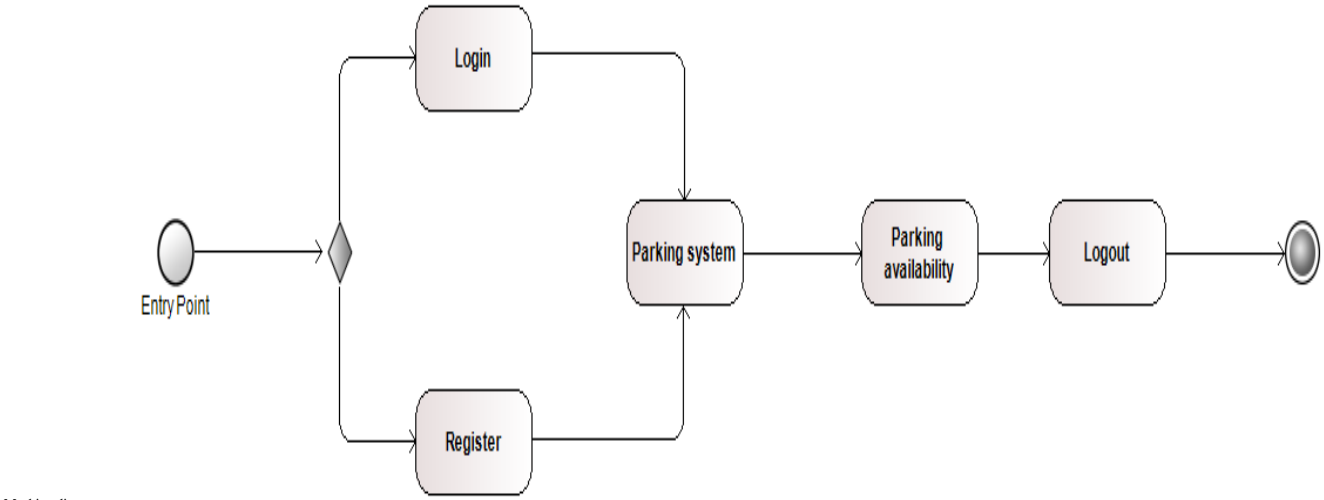


diagram.png

Figure 3.4: Sequence Diagram

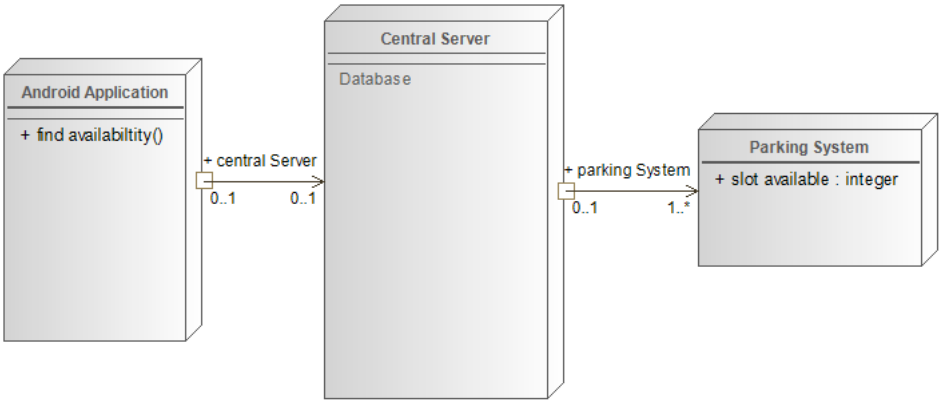
3.3.7 State Transition Diagram



Machine diagram.png

Figure 3.5: State Transition Diagram

3.3.8 Deployment Diagrams



Deployment diagram.png

Figure 3.6: Deployment Diagram

Chapter 4

Implementation and Coding

4.1 Introduction

During the implementation, the project team creates the actual product. Project developers begin building and coding the software. This section covers the listing of the code for major functionalities. It also covers the classes along with responsibilities.

4.2 Implementation Details

This section describes each module of the project. The modules are described for their functionalities and implementation details.

4.2.1 Operational Details

The project consists of three modules viz., Android Application, Image Processing and a Distributed Network.

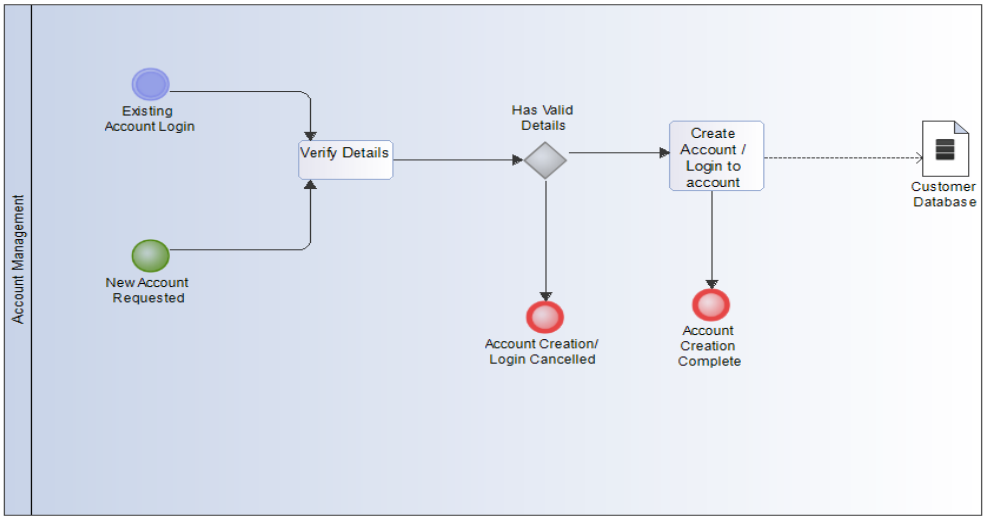
1. **Android Application**

   The android application is built to assist the user keep track of the parking systems available in his area and also the no. of vacant parking slots in the parking system. Th functionalities like Login, Signup, GPS Location detection and display of parking system and display of available parking slots in the chosen parking system. For this it maintains the customer registry table and performs validation of user. The operational flow of Android module as shown in Figure 4.1 and Figure 4.2 is discussed as follows:

   (a) A user requests for a new account or to login from an existing account login

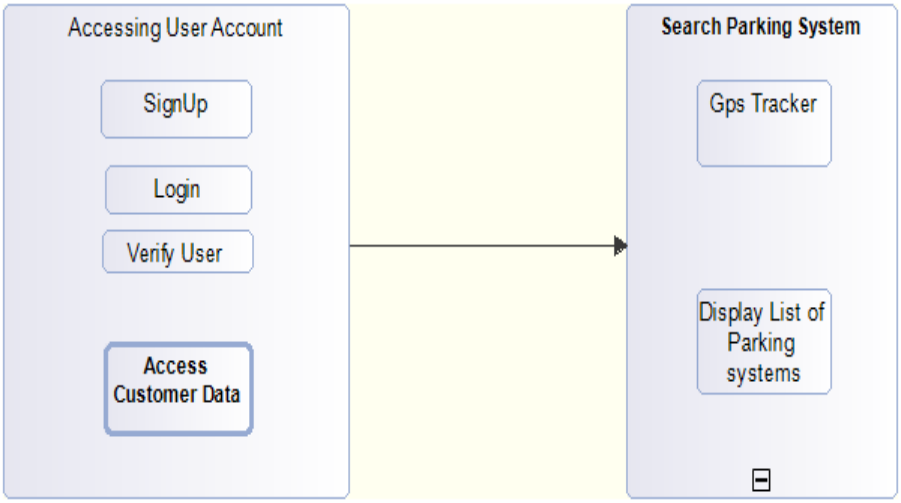
   (b) The user request is validated, if the details are valid, an account is created or existing user is successfully logged in.

   (c) The logged in user can now view the available parking systems, arranged according to user location and distance from the parking system.This module uses GPS Tracker to identify user location and calculate its distance from the parking system.



Login Bpmn Process diagram.png

Figure 4.1: Account Management Operational Diagram



complete Bpmn Process diagram.png

Figure 4.2: Android Module Operational Diagram

(d) The user must select the parking system facility they prefer.

(e) The android Application then performs dynamically to provide the no. of parking slots available in the selected parking system. The count of parking slots is dynamically updated as soon as an update request is sent from the central database to the android application.

(f) The application provides a logout option to switch accounts for safe usage.

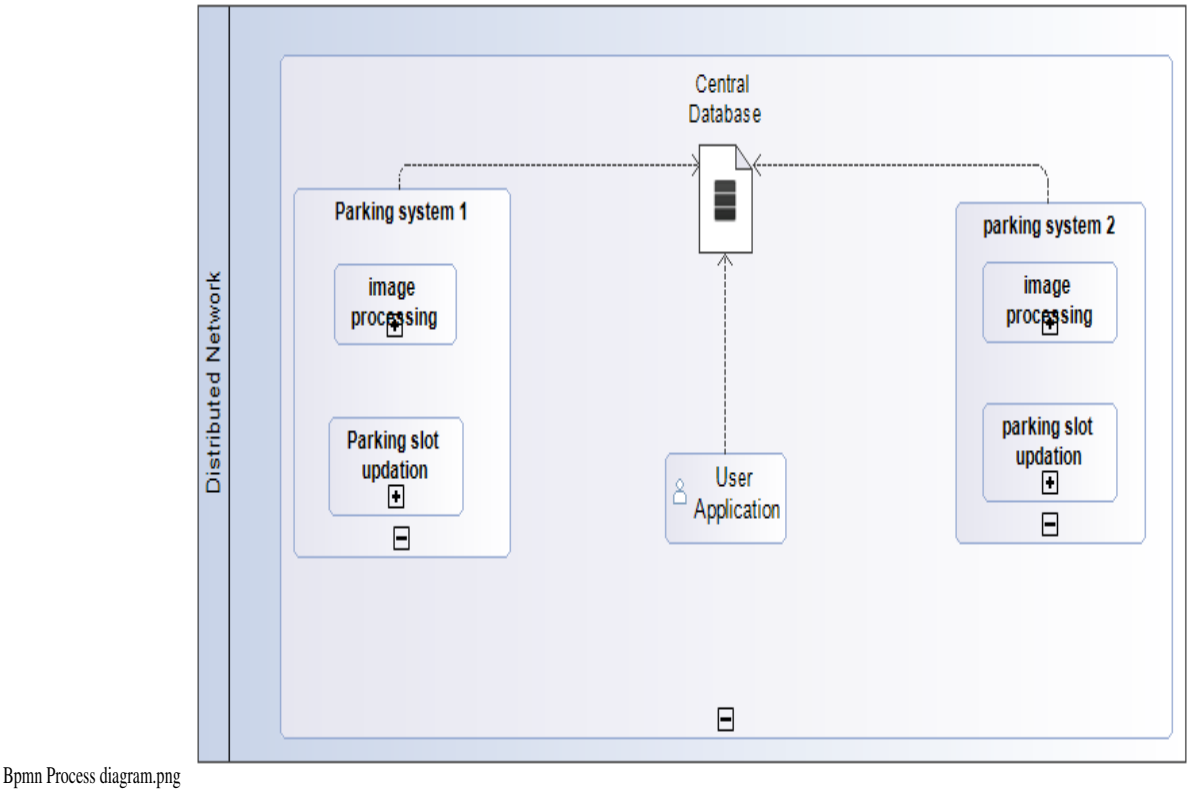
2. **Distributed Network**

The Distributed Network is used to implement multitenancy and operation of multiple Parking system using a centralized database possible. A single centralized database is used to provide efficient access to multiple parking systems to provide intel regarding the available parking slots. The distribited Network uses the concept of process local and request central. The Distributed Network can be seen as in Figure 4.3 as follows:

(a) The administrator provides host IPs to the parking systems that are to be connected the central database.

(b) All the devices used including mobile devices, databases, local processors are needed to be connected to this Distributed Network.

(c) Local processing is used to avoid latencies in distributed network.



Bpmn Process diagram.png

Figure 4.3: Distributed Network

(d) The central database requests only the processed and refined data from local databases.

(e) The request from user is entertained by the central server, which in turn requests the local database where the processed results are updated dynamically.

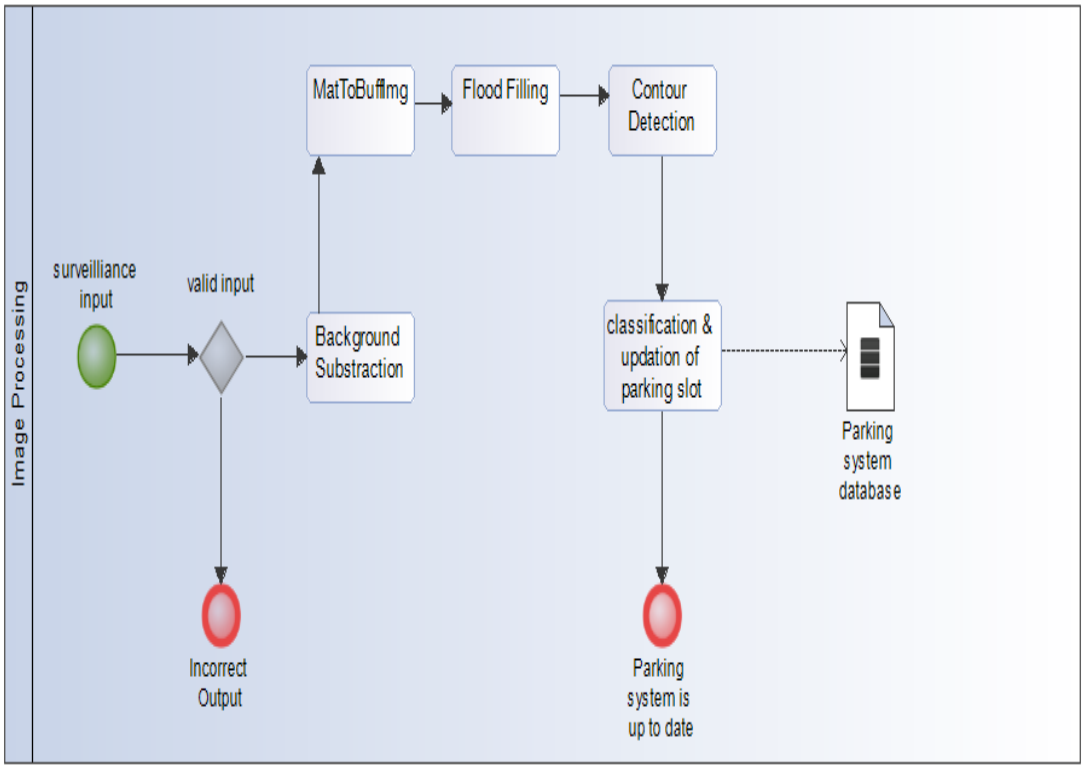
3. **Image Processing**

The image processing is the most vital module of the project. It is built using OpenCv libraries, Java, Image processing input, Background subtraction algorithms, Flood Fill Algorithm, Contour Detection and MatrixToBuffer image conversion techniques. The organizational flow of image processing is shown in Figure 4.4 as follows:

(a) The image processing module works on input to detect free parking slots.

(b) The input are subjected to background subtraction and to obtain a distinct and clear object detection by eliminating the background environment.

(c) Flood fill, also called seed fill, is an algorithm that determines the area connected to a given node in a multi-dimensional array. It is used to detect the total area of the car.



processing Bpmn Process diagram.png

Figure 4.4: Image Processing flow diagram

(d) core.matrix supports multiple implemntations of vector/matrix maths conforming to the same API.

(e) Contour tracing is a technique that is applied to digital images in order to extract their boundary. The Boundary of the car is extracted by using contour detection technique.

4.3 Major Classes

(a) **Main class**

This class deals with calling all the other functions and reorder the sequence of execution of code. The main class used in the project to provide a flow of project which is described as in Figure 4.2

(b) **matToBufferedImage**

It consists of all the functionality of getting the matrix to buffered conversion of the image on which processing needs to be done.

(c) **AppService**

AppService is a public class used for Linking apache Tomcat server and Json objects. It has app:service element which is the container for service information associated with one or more workspaces. An app:service element must contain one or ore app:workspace elements.

4.4 Code Listing

This section lists the sample code for the major functional requirements.

4.4.1 Code for Image processing main class

```
boolean flagg=video.isOpened();
System.out.println(flagg);
  if (video.isOpened()) {
      /**
       * Grabs, decodes and returns the next video frame.
       */
      System.out.println("in video opened");
      video.read(imgX);
      Mat mgray = new Mat();
      Imgproc.cvtColor(imgX, mgray, Imgproc.COLOR_BGR2GRAY);
      setImage(imgX);
       Mat fgMask=new Mat();
    backgroundSubtractorMOG.apply(imgX, fgMask,0.1);

    Mat output=new Mat();
    imgX.copyTo(output,fgMask);
    try{
    BufferedImage img1 = matToBufferedImage(output);
    Image imageScaled2 = img1.getScaledInstance(
   180,
    150, Image.SCALE_SMOOTH);
ImageIcon iic2 = new ImageIcon(imageScaled2);
jLabel2.setIcon(iic2);

BufferedImage img2 = matToBufferedImage(fgMask);
    Image imageScaled3 = img2.getScaledInstance(
   180,
    150, Image.SCALE_SMOOTH);
```

Figure 4.5: Code for Image processing main class

4.4.2 Code for mattToBuff Image Conversion

```
public static BufferedImage matToBufferedImage(Mat matrix) {
    if (matrix.channels() == 1) {
        int cols = matrix.cols();
        int rows = matrix.rows();
        int elemSize = (int) matrix.elemSize();
        byte[] data = new byte[cols * rows * elemSize];
        int type;
        matrix.get(0, 0, data);
        switch (matrix.channels()) {
            case 1:
                type = BufferedImage.TYPE_BYTE_GRAY;
                break;
            case 3:
                type = BufferedImage.TYPE_3BYTE_BGR;
                // bgr to rgb
                byte b;
                for (int i = 0; i < data.length; i = i + 3) {
                    b = data[i];
                    data[i] = data[i + 2];
                    data[i + 2] = b;
                }
                break;
            default:
                return null;
        }

        BufferedImage image2 = new BufferedImage(cols, rows, type);
        image2.getRaster().setDataElements(0, 0, cols, rows, data);
        return image2;
```

Figure 4.6: Code for mattToBuff Image Conversion

4.4.3 Code for Hibernate

```
?xml version="1.0" encoding="UTF-8"?>
!DOCTYPE hibernate-configuration SYSTEM
"http://hibernate.sourceforge.net/hibernate-configuration-3.0.dtd">

hibernate-configuration>
	<session-factory>
		<property name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</prop
		<property name="hibernate.connection.driver_class">com.mysql.jdbc.Driver</p
		<property name="hibernate.connection.url">jdbc:mysql://localhost:3306/parki
		<property name="hibernate.connection.username">root</property>
		<property name="hibernate.connection.password">123456</property>
		<property name="hibernate.hbm2ddl.auto">update</property>

		<mapping class="com.model.dao.User" />
		<mapping class="com.model.dao.Customer" />
		<mapping class="com.model.dao.Parking" />

	</session-factory>
/hibernate-configuration>
```

Figure 4.7: Code for Hibernate

4.4.4 Code for Login Authorization

```
@POST
@Path("/LoginAuth")
@Produces({ MediaType.APPLICATION_JSON })
public String LoginAuth(@FormParam("username") String userName, @FormParam("password") String password,
                @Context HttpServletResponse servletResponse) {
        try {
                JSONObject object = new JSONObject();
                HashMap<String, String> map = new HashMap<String, String>();
                map.put("username", userName);
                map.put("password", password);

                User user = (User) abstractDao.authenticate(map, User.class);
                if (user != null && user.getCustomer() != null) {
                        object.put("id", user.getUserid());
                        object.put("customerName", user.getCustomer().getCustomerName());
                        object.put("mobileNumber", user.getCustomer().getMobileNo());
                        object.put("address", user.getCustomer().getAddress());
                        object.put("email", user.getCustomer().getEmail());
                        object.put("result", "success");
                } else
                        object.put("result", "invalid");
                return object.toString();

        } catch (

        Exception e) {
                e.printStackTrace();
                return "{\"result\": \"error\"}";
        }
}
```

Figure 4.8: Code for Login Authorization

4.4.5 Code for User Registeration

```
// TODO Register Customer
@POST
@Path("/RegisterCustomer")
@Produces({ MediaType.APPLICATION_JSON })
public String registerCustomer(@FormParam("name") String name, @FormParam("email") String email,
                @FormParam("address") String address, @FormParam("mobileNumber") String mobileNumber,
                @FormParam("username") String username, @FormParam("password") String password) {
        try {
                JSONObject object = new JSONObject();

                User user = new User();
                user.setUsername(username);
                user.setPassword(password);
                Boolean res = abstractDao.add(user);
                if (res) {
                        Customer customer = new Customer();
                        customer.setCustomerName(name);
                        customer.setMobileNo(mobileNumber);
                        customer.setAddress(address);
                        customer.setEmail(email);
                        abstractDao.add(customer);
                        user.setCustomer(customer);
                        abstractDao.update(user);
                        object.put("result", "success");
                        return object.toString();
                } else {
                        object.put("result", "already");
                        return object.toString();
                }
        } catch (Exception e) {
                e.printStackTrace();
                return "{\"result\": \"error\"}";
        }
}
```

register.png

Figure 4.9: Code for User Register

4.4.6 Code for Parking System Detection

```
// TODO Get Nearest Parking
@POST
@Path("/GetNearestParking")
@Produces({ MediaType.APPLICATION_JSON })
public String GetNearestParking(@FormParam("latitude") Double latitude, @FormParam("longitude") Double longitude)
                throws URISyntaxException {
        JSONArray array = new JSONArray();
        try {
                TreeMap<Double, Long> tr = new TreeMap<Double, Long>();
                List<DataInterface> list = abstractDao.listByQuery("from Parking");
                for (DataInterface dataInterface : list) {
                        Parking parking = (Parking) dataInterface;
                        Double slat = parking.getLatitude();
                        Double slong = parking.getLongitude();
                        Double dis = abstractDao.getDistance(latitude, longitude, slat, slong);
                        tr.put(dis, parking.getParkingId());
                }
                DecimalFormat decimalFormat = new DecimalFormat("0.##");
                for (Entry<Double, Long> e : tr.entrySet()) {
                        Double distance = e.getKey();
                        JSONObject object = new JSONObject();
                        Parking parking = (Parking) abstractDao.getById("parkingId", e.getValue(), Parking.class);
                        object.put("parkingId", parking.getParkingId());
                        object.put("parkingLocationName", parking.getParkingLocationName());
                        object.put("distance", decimalFormat.format(distance) + " KM");
                        array.put(object);
                }
                return array.toString();
        } catch (Exception e) {
                e.printStackTrace();
                return array.toString();
        }
}
```

Figure 4.10: Code for Parking System Detection

4.4.7 Code for Retrieving Parking System Details

```
,

// TODO Get Nearest Parking
@POST
@Path("/GetParkingDetails")
@Produces({ MediaType.APPLICATION_JSON })
public String GetParkingDetails(@FormParam("parkingId") Long parkingId) {
	JSONObject object = new JSONObject();
	try {
		Parking parking = (Parking) abstractDao.getById("parkingId", parkingId, Parking.class);
		object.put("parkingLocationName", parking.getParkingLocationName());
		object.put("address", parking.getAddress());
		object.put("slotAvailable", parking.getSlotAvailable());
		object.put("result", "success");
		return object.toString();
	} catch (Exception e) {
		e.printStackTrace();
		object.put("result", "error");
		return object.toString();
	}
}
```

Figure 4.11: Code for Retrieving Parking System Details

4.4.8 Code for Updating Parking Slots dynamically

```
        // TODO Update Available Parking Slots
        @GET
        @Path("/UpdateAvailableParkingSlots")
        @Produces({ MediaType.APPLICATION_JSON })
        public String UpdateAvailableParkingSlots() {
                try {
                        Long parkingId = Long.parseLong(request.getParameter("parkingId"));
                        Integer slotAvailable = Integer.parseInt(request.getParameter("slotAvailable"));
                        Parking parking = (Parking) abstractDao.getById("parkingId", parkingId, Parking.class);
                        if (parking != null) {
                                parking.setSlotAvailable(slotAvailable);
                                abstractDao.update(parking);
                                return "OK";
                        } else
                                return "INVALID";
                } catch (Exception e) {
                        e.printStackTrace();
                        return "ERROR";
                }
        }

}
```

park.png

Figure 4.12: Code for Updating Parking Slots dynamically

4.5 Screenshots

This section shows screen shots for important/major functionalities.

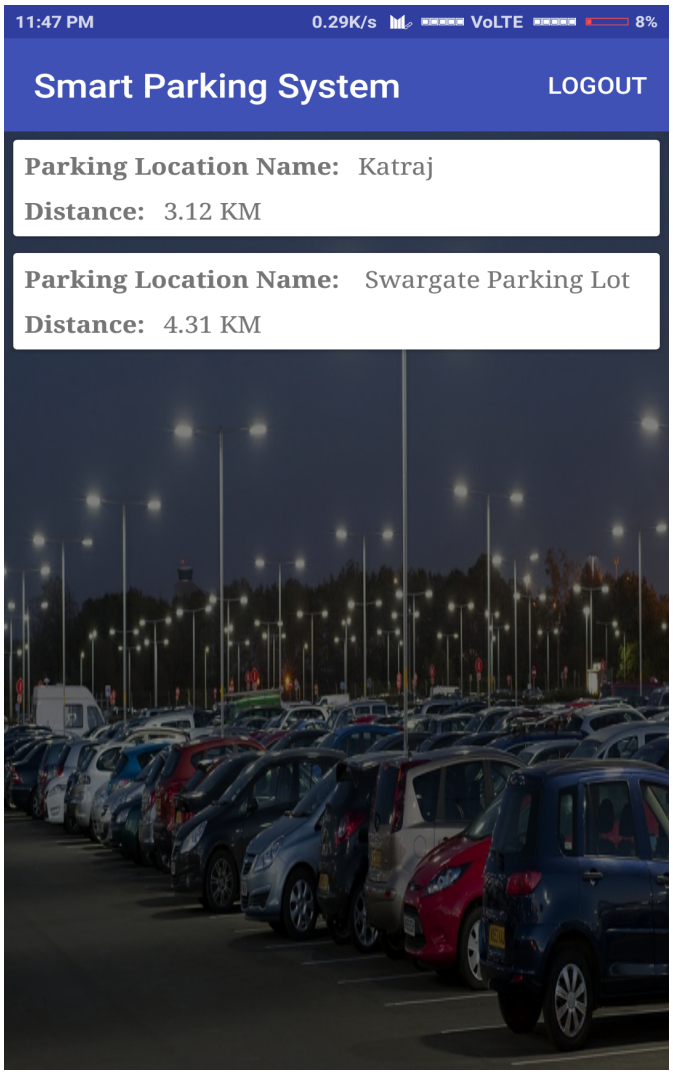


Figure 4.13: List of available parking system

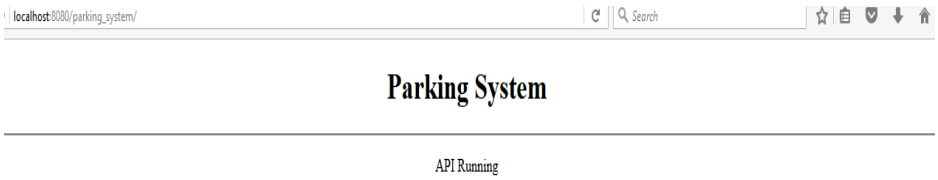


Figure 4.14: MySql Workbench api run screen

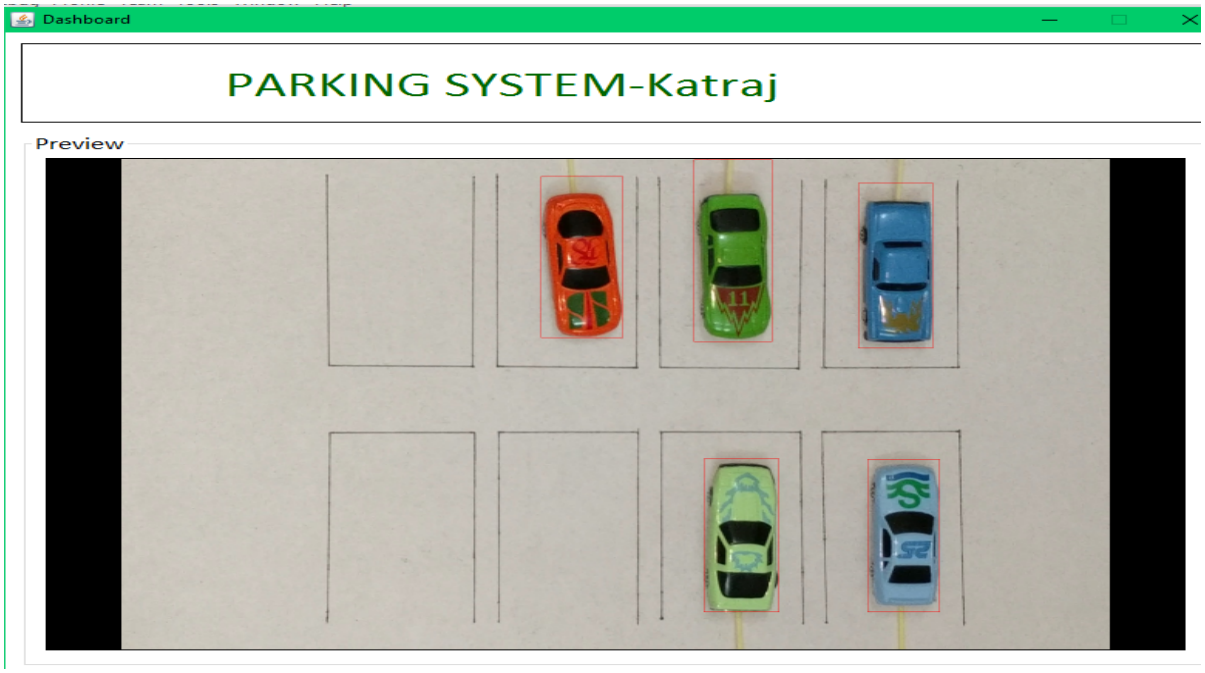


Figure 4.15: Image processing image 4

Chapter 5

Testing

5.1 Introduction

Software testing is the process of evaluation a software item to detect difference between given input and expected output. Also to assess the feature of a software item. Testing assesses the quality of the product. Software testing is a process that should be done during the development process. In other words software testing is a verification and validation process. A test approach is the test strategy implementation of a project, defines how testing would be carried out.

5.2 Unit Testing

Unit testing, also known as component testing, refers to tests that verify the functionality of a specific section of code, usually at the function level. In an object-oriented environment, this is usually at the class level, and the minimal unit tests include the constructors and destructor. It is often done by the programmer to test that the unit he/she has implemented is producing expected output against given input. In unit testing the smallest testable parts of an application, called units, are individually and independently scrutinized for proper operation.

Table 5.1: Unit Test - 1

Title	Login and Registration Check
Test Item	UserID, Password, Address, Mob. No., etc
Input Specification	Test Item containing alphanumeric characters.
Description	Specified details of Login/Register are entered and stored in database.
Expected Results	Entry for Test Item is created in Database.
Result	Pass

Table 5.2: Unit Test - 2

Title	Parking System check
Test Item	Parking System, Distance, etc
Input Specification	Test Item containing list with various parking system.
Description	List of parking system stored in the database is displayed to choose any parking system from the list.
Expected Results	Selected item from list displays parking availability information.
Result	Pass

Table 5.3: Unit Test - 3

Title	Parking Availability Check
Test Item	Parking System list
Input Specification	Select Test Item from list containing various parking system.
Description	Various Parking System are displayed in list so that parking availability can be displayed.
Expected Results	Selected parking system displays equivalent number of available parking slots from the specified parking system.
Result	Pass

Table 5.4: Unit Test - 4

Title	Parking System processing check
Test Item	Video based data of specified parking system.
Input Specification	Test Item containing various ongoing parking surveillance activities.
Description	Parking surveillance data is used to detect various vehicles and to determine the availability of parking slots. This data is stored in central database.
Expected Results	Entry for Test Item is created in Database.
Result	Pass

5.3 Integration Testing

Integration testing is any type of software testing that seeks to verify the interfaces between components against a software design. Software components may be integrated in an iterative way or all together. Normally the former is considered a better practice since it allows interface issues to be located more quickly and fixed. Integration testing works to expose defects in the interfaces and interaction between integrated components (modules). Progressively larger groups of tested software components corresponding to elements of the architectural design are integrated and tested until the software works as a system.

The purpose of integration testing is to verify functional, performance, and reliability requirements placed on major design items. These design items, i.e., assemblages (or groups of units), are exercised through their interfaces using black box testing, success and error cases being simulated via appropriate parameter and data inputs. Simulated usage of shared data areas and inter-process communication is tested and individual subsystems are exercised through their input interface. Test cases are constructed to test whether all the components within assemblages interact correctly, for example across procedure calls or process activations, and this is done after testing individual modules, i.e., unit testing. The overall idea is a building block approach, in which verified assemblages are added to a verified base which is then used to support the integration testing of further assemblages.

Software Integration Testing is performed according to the Software Development Life Cycle (SDLC) after module and functional tests. The cross-dependencies for software integration testing are: schedule for integration testing, strategy and selection of the tools used for integration, define the cyclomatical complexity of the software and software architecture, re-usability of modules and life-cycle / versioning management. Objective for this model is to take unit tested components and build a program structure that has been dictated by design. All modules that are unit tested will be taken as a whole and testing will be carried out. Integration strategy used will be “bottom up integration”.

Table 5.5: Integration Test - 1

Title	Android Application.
Description	Android application for parking availability detection is used to determine available number of parkings.
Test Steps	1. Install the Android application 2. On successful installation, functionality of all pages are checked 3. Check available parking slots from the specified parking system
Expected Results	On successful installation, all the functionality are working properly and appropriate availability of parking slots is displayed.
Result	Pass

Table 5.6: Integration Test - 2

Title	Web Application.
Description	Web Application is used to provide a interface between android application and parking system.
Test Steps	Run the application program
Expected Results	The application is able to interface the android application and parking system on web.
Result	Pass

Table 5.7: Integration Test - 3

Title	Parking System.
Description	The parking system processes live surveillance video of a specific parking system and determines appropriate available parking system.
Test Steps	1. Provide live surveillance video 2. Run the program for image processing of parking system
Expected Results	Image processing of the parking system determines available parking slots and send the avalability data to web application.
Result	Pass

5.4 Acceptance Testing

In engineering and its various sub disciplines, acceptance testing is a test conducted to determine if the requirements of a specification or contract are met. It may involve chemical tests, physical tests, or performance tests.

In systems engineering it may involve black-box testing performed on a system (for example: a piece of software, lots of manufactured mechanical parts, or batches of chemical products) prior to its delivery. In software testing the ISTQB defines acceptance as: formal testing with respect to user needs, requirements, and business processes conducted to determine whether a system satisfies the acceptance criteria and to enable the user, customers or other authorized entity to determine whether or not to accept the system. Acceptance testing is also known as user acceptance testing (UAT), end-user testing, operational acceptance testing (OAT) or field (acceptance) testing.

Table 5.8: Acceptance Test - 1

Title	User information
Description	User information is stored in database along with all the credentials.
Expected Output	Database stores all the attributes of the user in the user table.
Result	Pass

Table 5.9: Acceptance Test - 2

Title	Parking Availability data
Description	Image processing of live surveillance of parking system is processed to gather parking availability data in the database.
Expected Output	Parking availability data is displayed in android application of a specified parking system.
Result	Pass

Table 5.10: Acceptance Test - 3

Title	Performance
Description	The delay in running of the application determines reliability of the application.
Expected Output	The delay in determining of parking availability of the system is very less and very feasible and it produces reliable data.
Result	Pass

Chapter 6

Results and Discussion

6.1 Discussion

The proposed system is fully implemented and is ready to be deployed. Android application gives user the ease to login and register from the stored data in the database. These android application connects to central server. It sends object by creating a JSON object. The central server contains database which manages all the information regarding the user credentials and the parking availability details of various parking system. Parking system are equipped with image processing module which on the basis of live surveillance video of the parking system, determines the number of parking spaces available. As a whole, this system works fine with very low latency.

6.2 Main GUI Snapshots

This section contains the main GUI snapshots of the project.

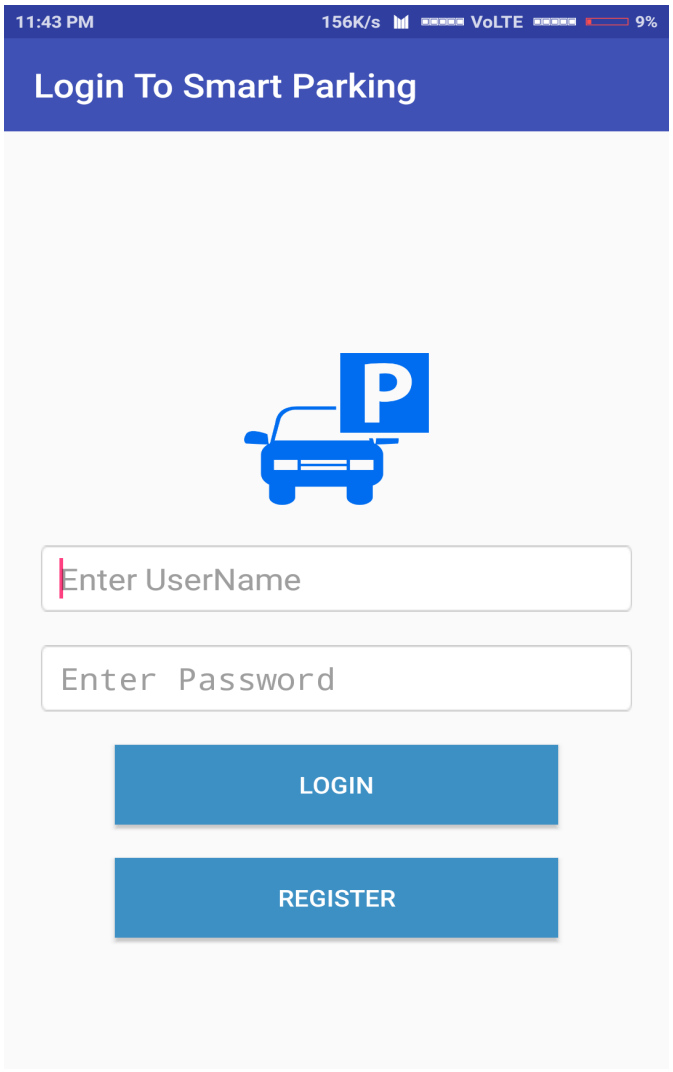


Figure 6.1: Android Application Start Screen

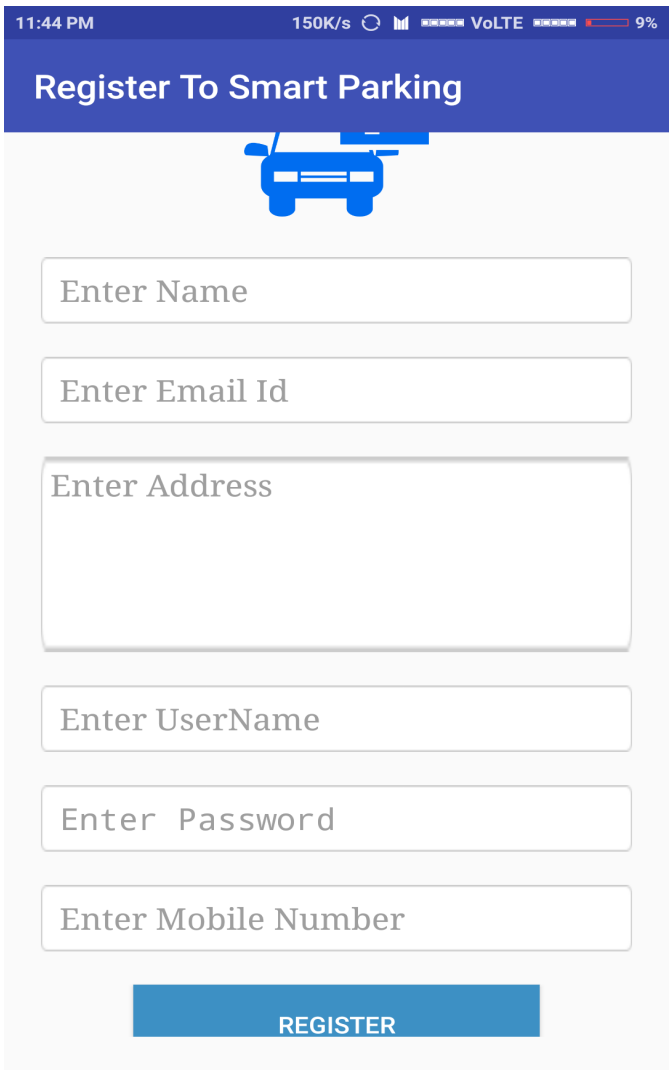


Figure 6.2: User Registration Screen

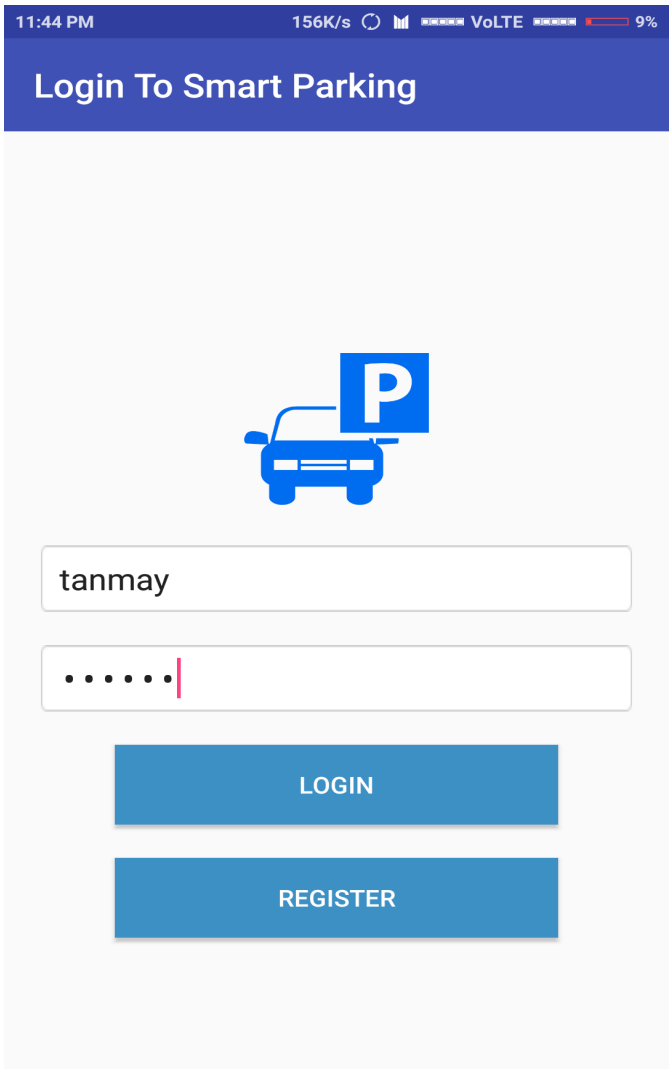


Figure 6.3: User Login Screen

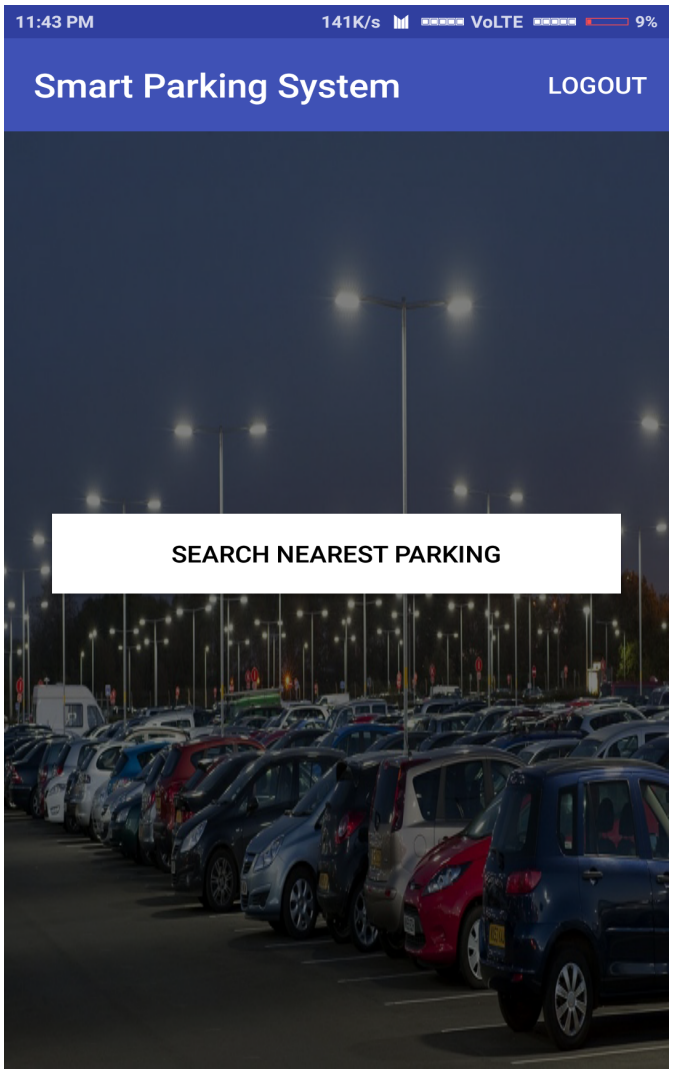


Figure 6.4: Search Page

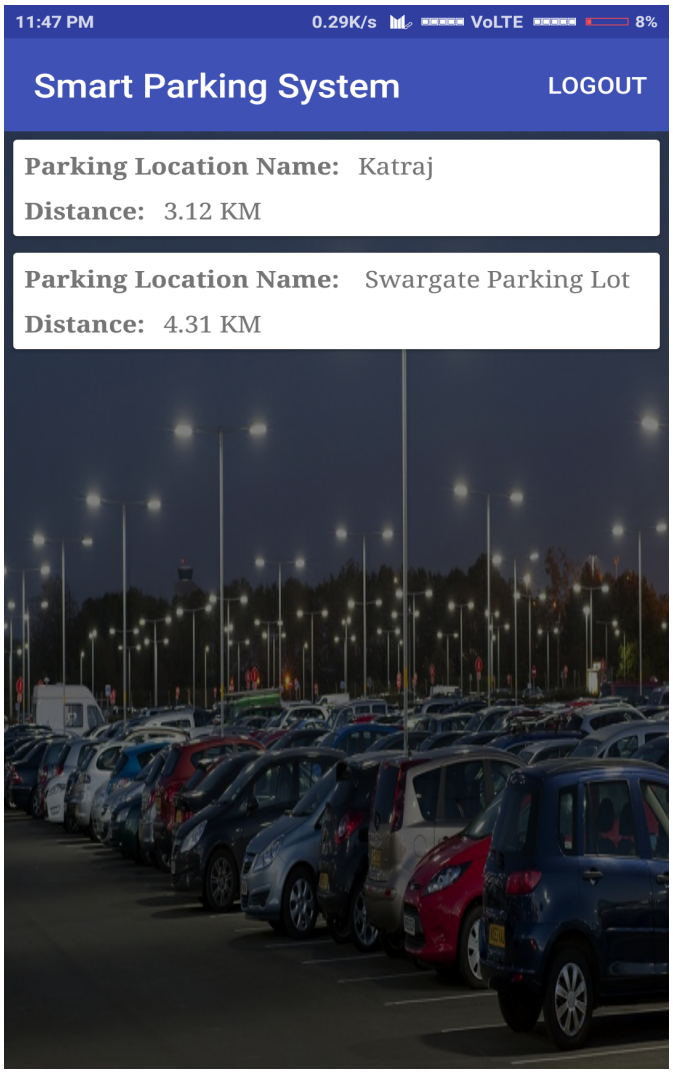


Figure 6.5: List of available parking system

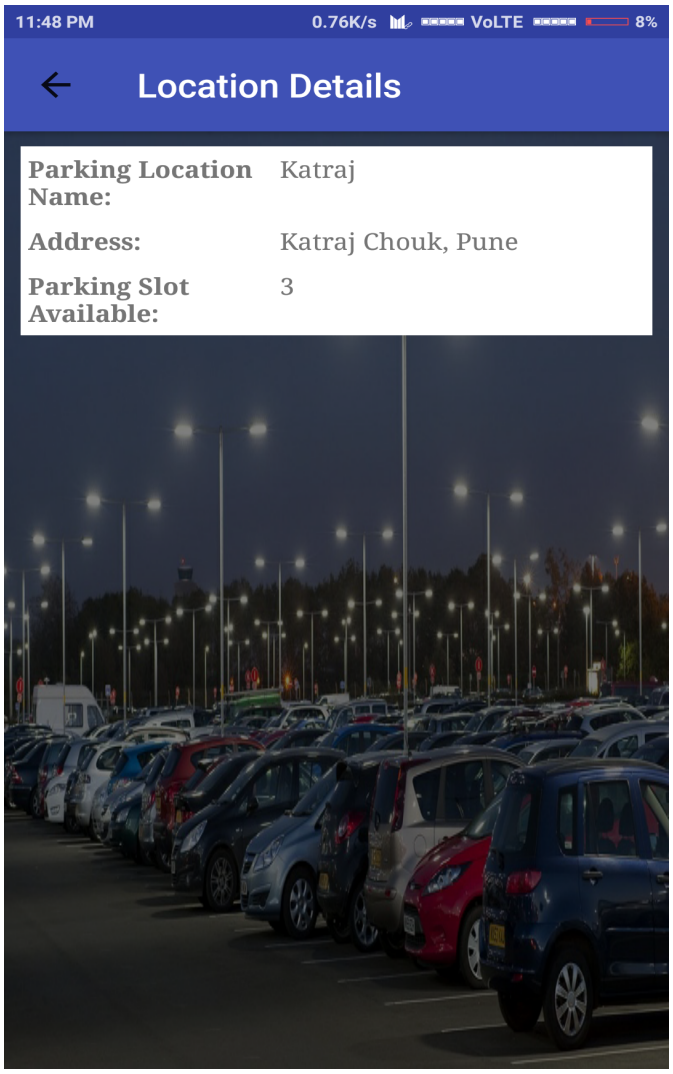


Figure 6.6: Display information regarding selected Parking system

Figure 6.7: Display information regarding selected Parking system

Figure 6.8: MySql Workbench api run screen

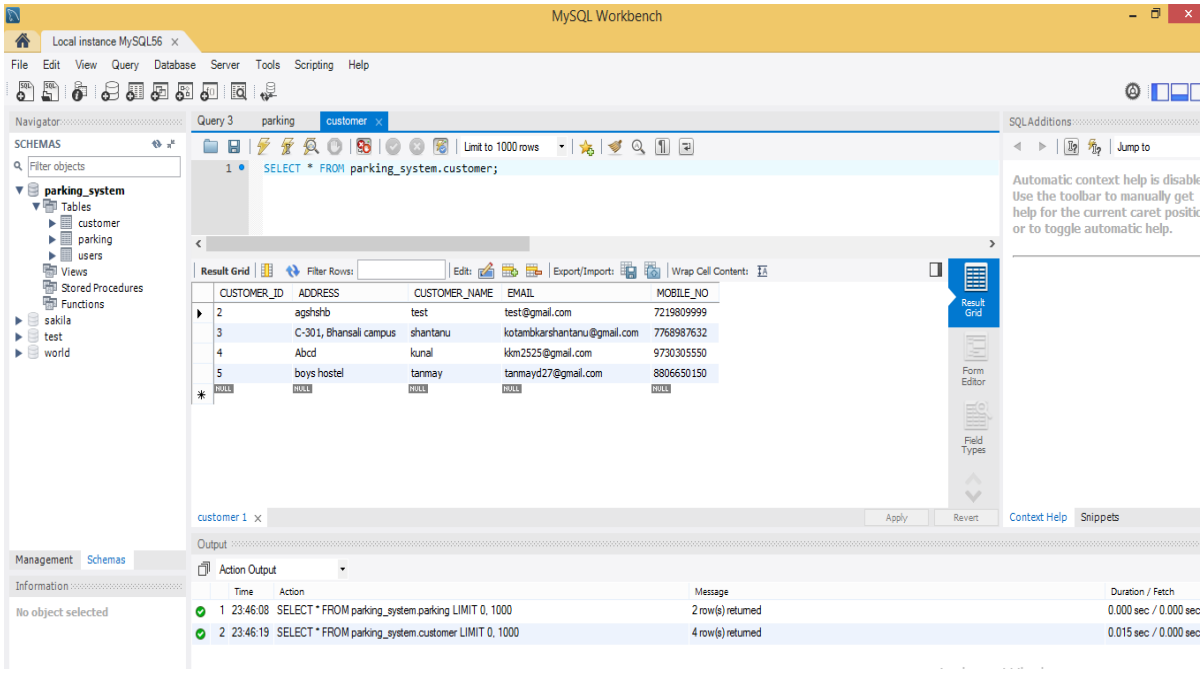


Figure 6.9: Customer Database

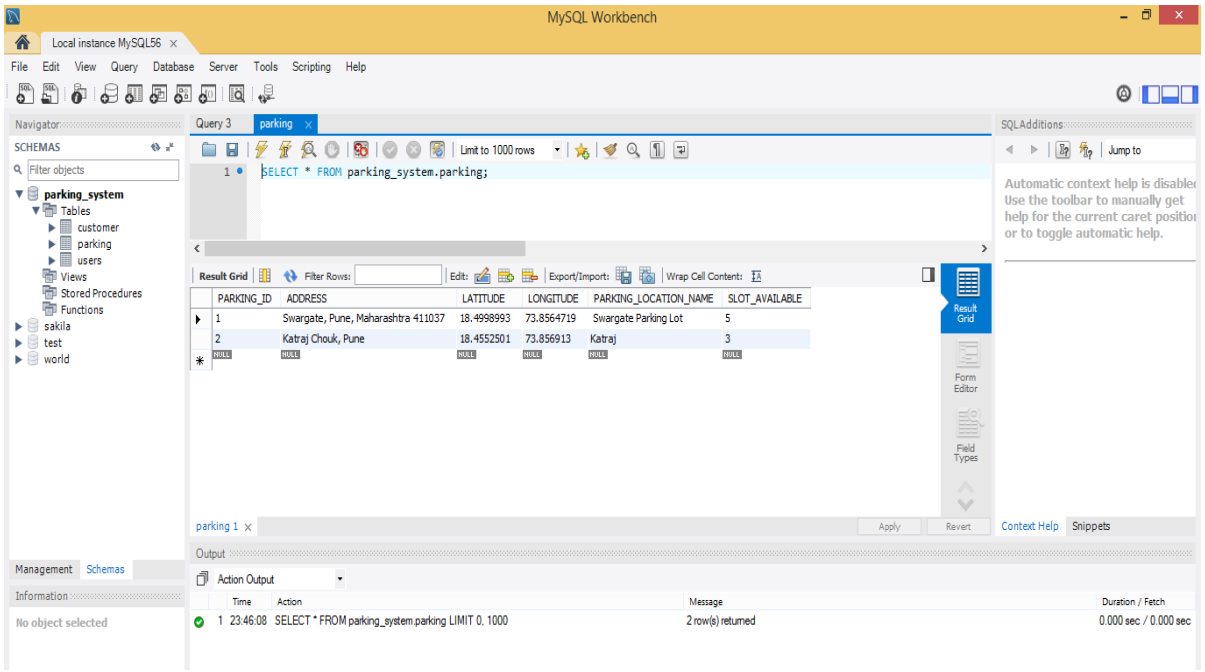


Figure 6.10: Parking system Database

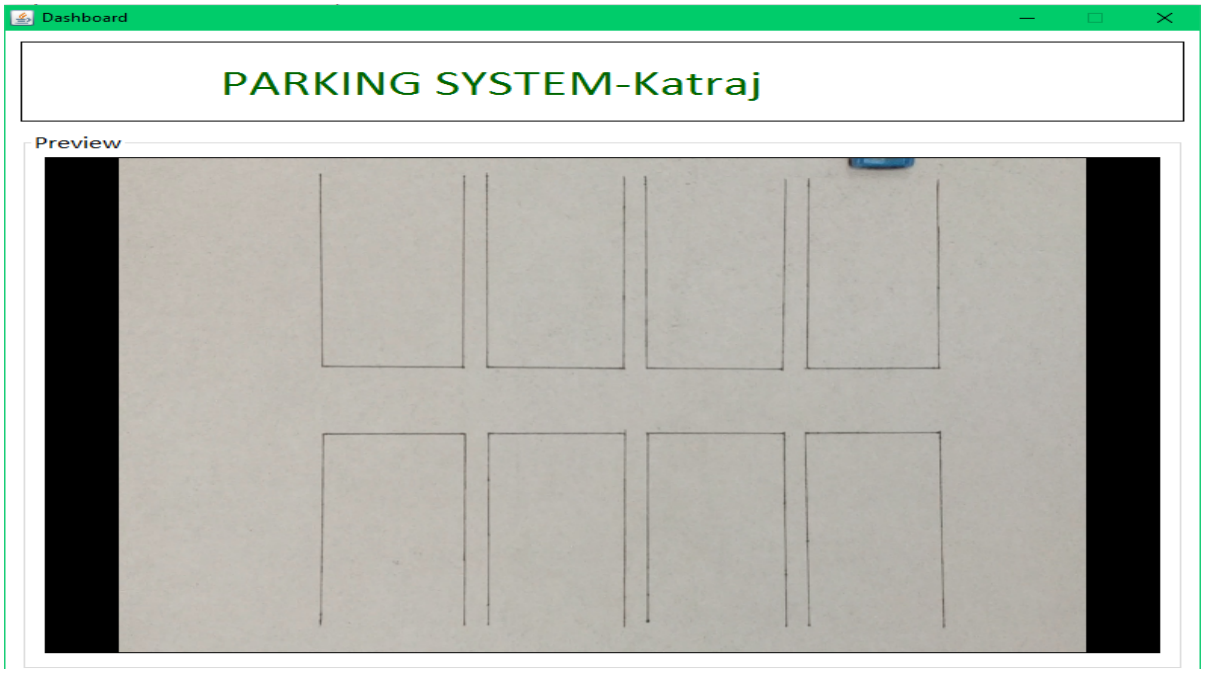


Figure 6.11: Image processing image 1

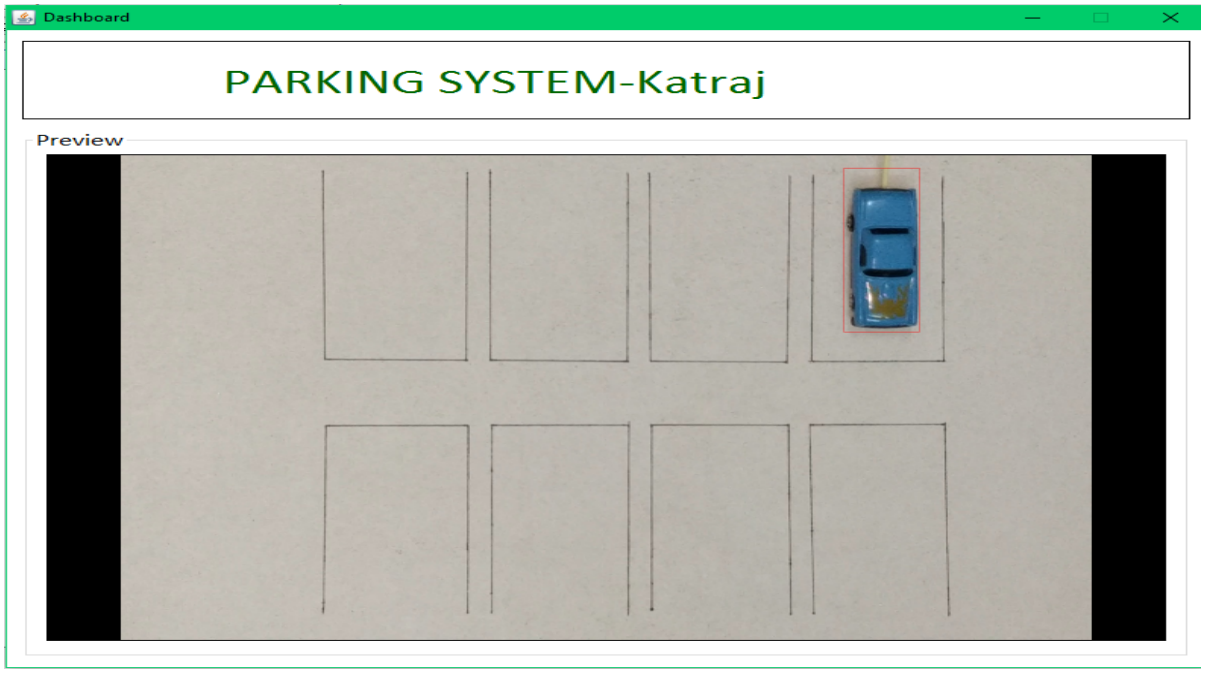


Figure 6.12: Image processing image 2

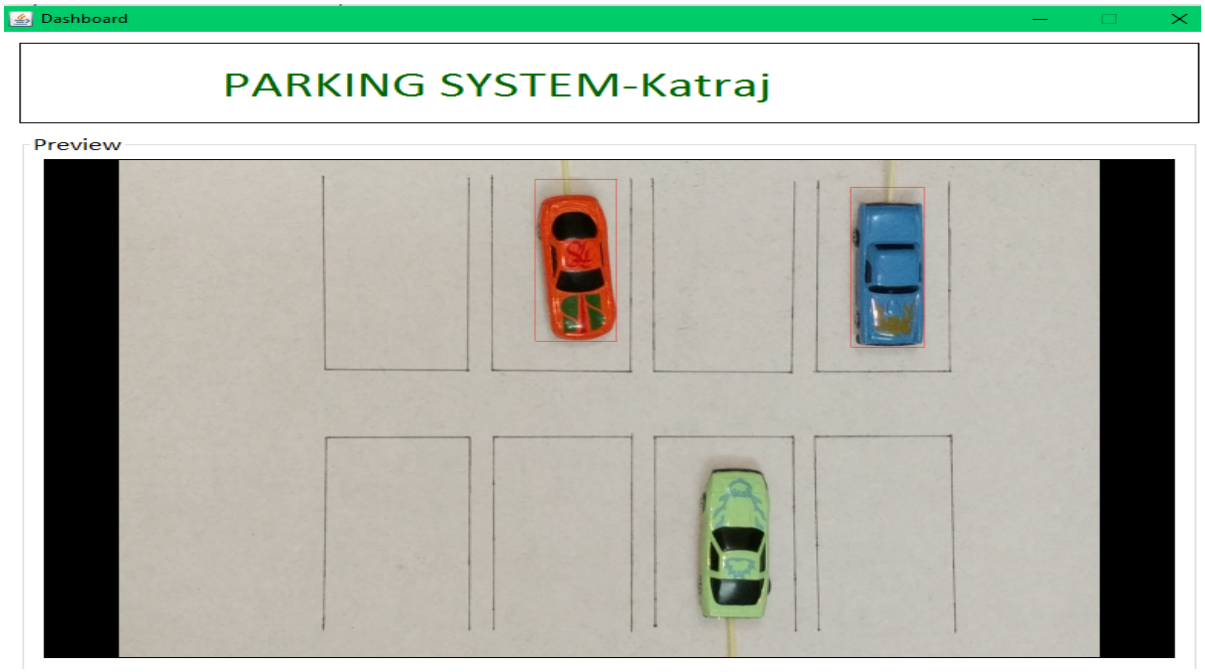


Figure 6.13: Image processing image 3

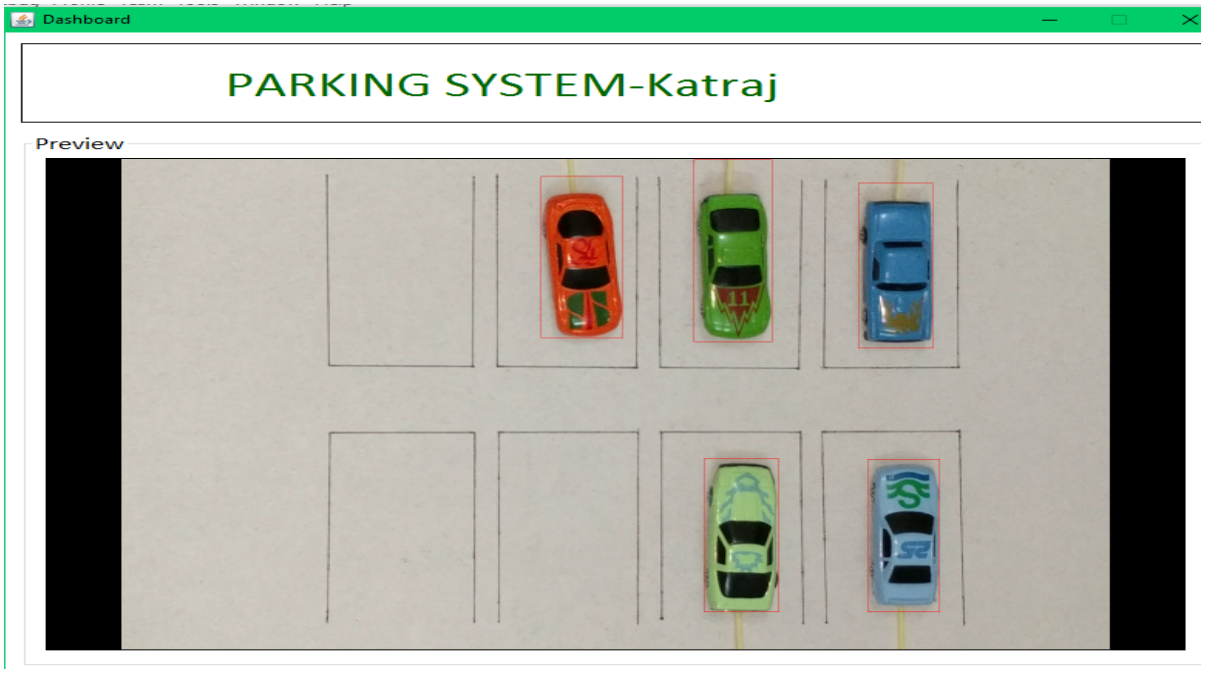


Figure 6.14: Image processing image 4

Chapter 7

Conclusion

We acknowledge Parking System as an issue that affects everyone in the community, from traffic management and law enforcement to residents of nearby neighborhood. A number of systems are needed that provide parking availability at the regional and lot-, floor-, aisle- or space-specific level to high-tech parking reservation and navigation systems.

This system helps addressing the parking problem upto an extent by providing the usage information about nearby parking spots.This system proves to be beneficial for the user as it is aimed at customer satisfaction as it helps in reducing the time required for finding parking spaces.An advanced parking management system with a user-friendly GUI helps in reducing the parking problems.

References

[1] "Onf sdn evolution - a white paper," 2016.

[2] D. Kreutz, F. M. Ramos, P. E. Verissimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-defined networking: A comprehensive survey," *Proceedings of the IEEE*, vol. 103, no. 1, pp. 14–76, 2015.

[3] "The opendaylight platform | opendaylight." `https://www.opendaylight.org/`.

[4] "Mininet: An instant virtual network on your laptop (or other pc) - mininet." `http://mininet.org/`.

[5] "Maven – welcome to apache maven." `https://maven.apache.org/`.

[6] I. F. Akyildiz, A. Lee, P. Wang, M. Luo, and W. Chou, "Research challenges for traffic engineering in software defined networks," *IEEE Network*, vol. 30, no. 3, pp. 52–58, 2016.

[7] S. Kaur, K. Kumar, J. Singh, and N. S. Ghumman, "Round-robin based load balancing in software defined networking," in *Computing for Sustainable Global Development (INDIACom), 2015 2nd International Conference on*, pp. 2136–2139, IEEE, 2015.

[8] R. Khondoker, A. Zaalouk, R. Marx, and K. Bayarou, "Feature-based comparison and selection of software defined networking (sdn) controllers," in *Computer Applications and Information Systems (WCCAIS), 2014 World Congress on*, pp. 1–7, IEEE, 2014.

[9] S.-Y. Wang, "Comparison of sdn openflow network simulator and emulators: Estinet vs. mininet," in *Computers and Communication (ISCC), 2014 IEEE Symposium on*, pp. 1–6, IEEE, 2014.

[10] "Estinet technologies inc.." `http://www.estinet.com/`.

[11] T. D. Nadeau and K. Gray, *SDN: Software Defined Networks: An Authoritative Review of Network Programmability Technologies*. " O'Reilly Media, Inc.", 2013.

Appendices

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 6, Issue 3, March 2017

Parking Management System using Image Processing and Distributed Approach

Prof. S.P. Bholane[1], Shantanu Kotambkar[2], Ajinkya Lakade[2], Kunal Mande[2], Tanmay Deshmukh[2]

Assistant Professor, Department of Computer Engineering, Sinhgad College of Engineering, Pune, India [1]

Student, Department of Computer Engineering, Sinhgad College of Engineering, Pune, India [2]

Abstract: Vehicle parking in today's date has turned into a noteworthy issue in urban territories, because of the absence of parking facilities and poor management. Problems emerging from the absence of parking facilities and poor administration incorporate movement clog, expanded contamination, expanded utilization of fuel.To relieve these issues, a model has been produced with the android application, appropriated frameworks, image processing. The prototype with the help of an android application gives the information regarding the availability of car parking to the user directly. Delivering User the information regarding the parking space availability will thus reduce the time, fuel and efforts invested in search of a parking space. This will also lead to a reduction in traffic congestion, ultimately leading to less pollution. A database maintained for the availability of parking space can be used for better management of parking spaces. For detection of car parking spaces, image processing is helpful which will determine the parking availability. This prototype hence, will reduce manpower needs and increase the flexibility and security. The image processing at the local server level and only providing the required information to the central server thus optimizes the process followed by this prototype. Thus, a distributed parking prototype is developed to mitigate the problems faced by a common man and in the process conserving fuel and helping reduce global warming, pollution as well.

Keywords: Image processing, car park occupancy, object detection, information system.

I. INTRODUCTION

The model with the assistance of an android application gives the data in regards to accessibility of available parking details to the client directly. The user knows the parking spot accessibility, in this way lessening the time, fuel and endeavors put resources into the search for a parking space. This will likewise prompt the user to look for parking spaces easily. A database in which the accessibility of parking spot is overseen helps maintain real-time availability data. For recognition of vehicle parking spots, real-time processing of surveillance video will help decide the parking accessibility. This model hence, will decrease labor needs and increment adaptability and security. The image processing is done at the local server level and just the required data is given to the central server, thus diminishing the enhancing the procedure and expounding the disseminated approach took after by this model. Thus, a appropriated parking model is created to moderate the issues confronted by a typical man and in the process monitoring fuel and decreasing global warming, air pollution.

The project "Parking Management System using Image Processing and Distributed Approach " is made with an outlook to solve a civic issue of free parking space management and employ them in urban areas. The project considers a distributed approach to solving the issues, thus minimizing manpower needs and maximizing the optimal efficacy of the system.

Parking Management has been an issue of concern in today's date due to the increasing ratio of personal vehicles to parking spaces. The project works on the concepts of distributed processing, image processing, android application development and web application development is aimed to increase the productivity of parking system. The parking slots are fulfilled according to the user requirement, thus ensuring optimal use of parking space and user's ease of access.

II. RELATED WORK

This section provides a brief overview of relevant literature on parking availability detection, including supporting information on related work in Real-time Data Extraction, Decision Module, Driver's guidance.

2.1 Real-time Data Extraction.

Provided that the purpose of this approach is to ensure that system should be adapted for multiple parking dispositions, the aim is to introduce a real-time approach for data extraction and moving car detection. Since cameras are used to provide fixed scene video frames, it is proposed to detect the car in its motion. This allows to allocate the car a suitable parking slot and update the parking system count of parking slots.

The first phase is to interpret a base frame for image processing. Whenever a car enters the frame, we extract the motion of the cars in the fixed scene. Providing the base frame as the initial condition, we can opt to the implementation of adaptive background subtraction

DOI10.17148/IJARCCE.2017.63136

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319 5940
International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 6, Issue 3, March 2017

algorithm that adaptively eliminates the background of the parking video frame and consequently, detects and extracts the car in its motion. The first approach involves extraction. Secondly, the number of slots correspond to the detected cars. So as to update the slot vacancy data in our central server and also notify the user regarding available vacant slots.

2.2 Parking Event Detection and Update in Database

Previous work that has been carried out in the area of car motion detection. This section focuses on the back-end event detection and updates to the central database. As soon as the car is detected in its motion using background subtraction algorithm, an event is triggered and an update to the database is sent. Another such event is triggered after every 15 frames are captured and processed to assist and ensure timely update of the database. Along with this, a user triggers an event as soon as an update is requested by the user. Thus, the timely and well-maintained database is updated.

2.3 Integrated Approach in the Design of Car Park Occupancy

In large parking areas such as those at mega shopping malls or stadiums, drivers always have difficulty to find vacant car park lots especially during peak periods or when the parking lots are almost full. A solution to reduce the drivers' searching time for vacant car-park lots will greatly save time, reduce cost and improve the traffic flow in the car park areas. In this approach, a research project which was developed to acquire car-park occupancy information using an integrated approach of image processing algorithms is used. The motivation for developing this system came from the fact that minimum cost is involved because image processing technique is used rather than sensor-based techniques. Security surveillance cameras which are readily available in most car parks can be used to acquire the images of the car park. This solution is much cost effective than installing the sensor on each parking lot.

A wide range of stopping direction frameworks have been created and intended to abbreviate the looking time for empty parking areas. This is particularly essential for drivers who need to scan for accessible auto parks amid pinnacle hours or when the auto parks are full. Searching for parking space inhabitance data, for the most part, fall into four classifications – counter-based, wired-based, remote based and picture based frameworks.

III.SYSTEM OVERVIEW

The main objective of this proposed system is to provide a real-time solution for the ongoing parking problems especially in the metropolitan cities where searching for parking space is a major problem. This proposed solution will easily adapt to various parking system.
There are several approaches to solving such problems but those require heavy manpower duty or heavy sensor equipment which proves to be rather costly. This system will not only provide an efficient solution to such problems but also will help reduce the pollution and keeping the environment safe.

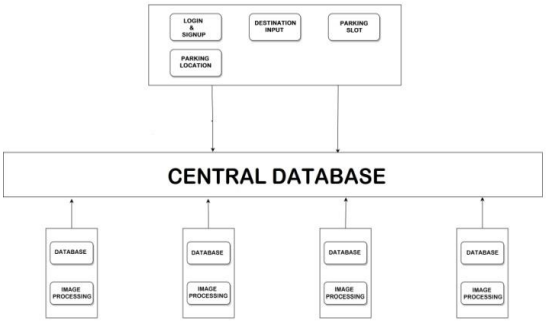


Fig.01. Architectural Overview

There are three modules to implement this system. Firstly, the image processing module where at various parking system the number of parking slots available will be calculated with the help of image processing. Secondly, the central server where the actual data would reside. The data from image processing is sent to the central server. The third module consists of android application where the individual user can access the data at the central server. The proposed system consisting of these modules will help the user at any location to search for available parking slot via various parking options in the android application. The user will get to locate available parking spaces nearest to their location.

3.1 Real-time data extraction using Image processing

In this module, some set of algorithms is applied to real-time parking system through a surveillance camera. This surveillance camera will enable this system to perform image processing on live video tracking of a specific parking system. Initial frames from the video are recorded with all empty parking spaces empty as a reference point. This method is called offline configuration. After this method, image frames with vehicles parked at the specific parking slot are recorded and using background subtraction method, these frames are subtracted to obtain the desired result. A contour is drawn which is nothing but a rectangular boundary alongside the vehicle. This contour helps to determine whether a parking spot is occupied or vacant. The number of parking slots available is calculated and these data is stored in the database at the central server.

3.2 Distributed Approach

This section discusses the distributed approach used in the Parking Management System. The distributed approach consists of a central Database and several local Database servers, each handling an individual parking system. The central database, handle multiple local servers and thus, serves as the main information provider to the user. The

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 6, Issue 3, March 2017

user posts a query regarding a particular parking system to the central server, which in turn retrieves the information from the appropriate local database and then sends the data to the user. This enables lesser congestion of network, provides better security of data and optimizes the Parking System management using Distributed approach.

3.3 Android Applications

The Android application enables the user to search for parking spaces near their specified location. The user will get the number of parking systems on the server and will be able to access the available parking slot in the specified parking system. Android application will also keep Login/Sign-Up info to specify parking spots for the specified user. Android Application gives a real-time solution to search for parking in the crowded cities among various options.

As the name suggests, "Parking Management System using Image processing and Distributed approach", the system uses Image processing, Distributed Network consisting of a central and local database, along with an android application for user convenience. The image processing module performs the major task of detecting and assessing the count of vacant parking slots. This count is updated in a local database at regular intervals and whenever is an event trigger database access requirement. This local data is fetched from the central server when the user requests for information about the corresponding parking system. The android application provides functionality such as User login and sign-up. The android application displays all nearby parking locations along with their distance from the user. The user can select the parking location and view the available parking slot.

IV. ALGORITHM

Image processing Algorithm

The image processing algorithm is used to detect vehicles at the parking slot and to determine whether the parking spot is empty. The following algorithm is implemented using OpenCV libraries.

1) Initial setup of parking system requires and image frame which consists of all empty parking slots in the parking system. This frame is stored in a matrix as a reference to the parking system. This frame is referred as initial frame.

2) When a vehicle arrives, the image frames with the vehicle at a specific parking slot is gathered. These frames denote that a vehicle arrives in the frame boundary.

3) Subtract the initial frame from the frame where the vehicle is in parking space. Store this difference in a matrix.

4) Use all the frames of the vehicle arriving to draw a contour alongside the boundary of the vehicle. Store this contour in a list. Predefine a threshold value for the vehicle size according to the image frame.

5) If the contour size is in the range of threshold value then increment the count for a number of vehicles parked.

6) Calculate the number of parking available and store this data in the database at the central server.

This algorithm runs at the every parking system and collects data from all parking system and store it in the database at the central server.

V. CONCLUSION

The proposed framework gives an alternative way to traditional parking management system, to deal with identification of opportunities of parking spots. This approach comprises of distinguishing empty parking spots by means of a surveillance camera. The framework likewise gives drivers' information assistance. This module gives a continuous refresh of parking slots to drivers through cellphone application with a specific end goal to guide them and encourage the errand of finding an accessible parking slot.

The proposed system provides a vision-based approach for detecting vacancies of parking spaces for a given context. This approach of the module provides real-time updates of parking spaces to drivers via mobile phone application in order to guide them and facilitate the task of locating an available parking space

ACKNOWLEDGMENT

We want to express our earnest appreciation towards our guide **Prof. S. P. Bholane** for his significant direction and supervision that helped us in our exploration. He has constantly motivated us to investigate new ideas and seek after more up to date examine issues. We credit our venture commitment to him. By and large, we likewise want to thank our review committee members **Prof. A. R. Joshi and Prof. S. S. Pawar** for their time, proposals, and for thoughtfully consenting to be on our review committee, and continually making themselves accessible. We can't express gratitude toward them enough.

REFERENCES

[1] Masmoudi, Imen, et al. "Trajectory Analysis for Parking Lot Vacancy Detection System." IET Intelligent Transport Systems (2016).

[2] Masmoudi, Imen, et al. "Architecture of Parking Lots Management System for Drivers' Guidance." Systems, Man, and Cybernetics (SMC), 2015 IEEE International Conference on. IEEE, 2015

[3] Bong, D.B.L., K.C. Ting , and K.C.Lai." Integrated approach in the design of car park occupancy information system(COINS)."IAENG International Journal of Computer Science 35.1 (2008): 7-14.

[4] Suhr, Jae Kyu, and Ho Gi Jung. "Sensor fusion-based vacant parking slot detection and tracking." IEEE Transactions on Intelligent Transportation Systems 15.1 (2014): 21-36.

[5] Tschentscher, Marc, et al. "Scalable real-time parking lot classification: An evaluation of image features and supervised learning algorithms." 2015 International Joint Conference on Neural Networks (IJCNN). IEEE, 2015.

[6] Momin, B. F., and S. M. Kumbhare. "Vehicle detection in video surveillance system using Symmetrical SURF." Electrical, Computer and Communication Technologies (ICECCT), 2015 IEEE International Conference on. IEEE, 2015.

ISSN (Online) 2278–1021
ISSN (Print) 2319–5940

IJARCCE

INTERNATIONAL JOURNAL OF ADVANCED RESEARCH IN COMPUTER AND COMMUNICATION ENGINEERING

ISO 3297:2007 Certified

Impact Factor 5.947

CERTIFICATE OF PUBLICATION

PROF. S.P. BHOLANE

Assistant Professor, Department of Computer Engineering, Sinhgad College of Engineering, Pune, India

Published a research paper entitled

Parking Management System using Image Processing and Distributed Approach

in IJARCCE, Volume 6, Issue 3, March 2017

DOI 10.17148/IJARCCE.2017.63136

Certificate No: 10.17148-1/5-A
www.ijarcce.com

Tejass Publisheers
ORGANIZATION

Editor-in-Chief
IJARCCE

ISSN (Online) 2278–1021
ISSN (Print) 2319–5940

IJARCCE

INTERNATIONAL JOURNAL OF ADVANCED RESEARCH IN COMPUTER AND COMMUNICATION ENGINEERING

ISO 3297:2007 Certified

Impact Factor 5.947

CERTIFICATE OF PUBLICATION

SHANTANU KOTAMBKAR

Student, Department of Computer Engineering, Sinhgad College of Engineering, Pune, India

Published a research paper entitled

Parking Management System using Image Processing and Distributed Approach

in IJARCCE, Volume 6, Issue 3, March 2017

DOI 10.17148/IJARCCE.2017.63136

Certificate No: 10.17148-2/5-A
www.ijarcce.com

Tejass Publisheers
ORGANIZATION

Editor-in-Chief
IJARCCE

ISSN (Online) 2278–1021
ISSN (Print) 2319–5940

IJARCCE

INTERNATIONAL JOURNAL OF ADVANCED RESEARCH IN COMPUTER AND COMMUNICATION ENGINEERING

ISO 3297:2007 Certified

Impact Factor 5.947

CERTIFICATE OF PUBLICATION

AJINKYA LAKADE

Student, Department of Computer Engineering, Sinhgad College of Engineering, Pune, India

Published a research paper entitled

Parking Management System using Image Processing and Distributed Approach

in IJARCCE, Volume 6, Issue 3, March 2017

DOI 10.17148/IJARCCE.2017.63136

Certificate No: 10.17148-3/5-A
www.ijarcce.com

Tejass Publisheers
ORGANIZATION

Editor-in-Chief
IJARCCE

ISSN (Online) 2278–1021
ISSN (Print) 2319–5940

IJARCCE

INTERNATIONAL JOURNAL OF ADVANCED RESEARCH IN COMPUTER AND COMMUNICATION ENGINEERING

ISO 3297:2007 Certified

Impact Factor 5.947

CERTIFICATE OF PUBLICATION

KUNAL MANDE

Student, Department of Computer Engineering, Sinhgad College of Engineering, Pune, India

Published a research paper entitled

Parking Management System using Image Processing and Distributed Approach

in IJARCCE, Volume 6, Issue 3, March 2017

DOI 10.17148/IJARCCE.2017.63136

Certificate No: 10.17148-4/5-A
www.ijarcce.com

Tejass Publisheers
ORGANIZATION

Editor-in-Chief
IJARCCE

ISSN (Online) 2278–1021
ISSN (Print) 2319–5940

IJARCCE

INTERNATIONAL JOURNAL OF ADVANCED RESEARCH IN COMPUTER AND COMMUNICATION ENGINEERING

ISO 3297:2007 Certified

Impact Factor 5.947

CERTIFICATE OF PUBLICATION

TANMAY DESHMUKH

Student, Department of Computer Engineering, Sinhgad College of Engineering, Pune, India

Published a research paper entitled

Parking Management System using Image Processing and Distributed Approach

in IJARCCE, Volume 6, Issue 3, March 2017

DOI 10.17148/IJARCCE.2017.63136

Certificate No: 10.17148-5/5-A
www.ijarcce.com

Tejass Publisheers
ORGANIZATION

Editor-in-Chief
IJARCCE