**A Project Report**

**On**

**Automated Vehicle Parking System**

**Submitted in partial fulfillment for award of the**

**Degree In**

Bachelor of Technology

**INFORMATION TECHNOLOGY**

****

**H. R. Institute of Technology**

**GHAZIABAD**

**20xx-20xx**

**Guided by : Submitted by :**

**jhvhvvj NAME (ROLL NO)**

**NAME (ROLL NO)**

**Table of Contents**

1. **PROBLEM STATEMENT…………………………………………...5**
2. **ABSTRACT…………………………………………………………...6**
3. **INTRODUCTION………………………………………………….....8**
4. **OBJECTIVES………………………………………………………..10**
5. **TECHNICAL ARCHITECTURE**

* **FLOWCHART………………………………………………… 11**
* **DATA FLOW DIAGRAM……………………………………………………..13**
* **ENTITY-RELATIONSHIP DIAGRAM……………………. 14**

1. **METHODOLOGY…………………………………………………....16**
2. **HARDWARE & SOFTWARE REQUIREMENTS………………...18**
3. **IMPLICATIONS FOR FUTURE RESEARCH……………………. 19**
4. **REFERENCES………………………………………………………..20**

**PROBLEM STATEMENT**

Everyone who owns or drives a vehicle in India or abroad, would be all too familiar with the hassles of finding parking spaces, misbehaving parking attendants, inconsistent or monopolized rates and other problems associated with it.

The Automated Vehicle Parking System had been conceived with the view to automate the manual workflows involved in the management of Vehicle Parking Lots.

**abstract**

The Automated Vehicle Parking System had been conceived with the view to automate the manual workflows involved in the management of Vehicle Parking Lots. It drastically reduces the effort, inaccuracies, error-prone tendencies, delays and overheads involved in performing the same tasks by hand.

It was aimed to provide a fully automated system that was capable of checking in and out of vehicles entering and exiting the designated parking lot, and recording relevant information about them. Revenue calculation and data entry is automated to the largest possible extent. Only minimum intervention from the manual user is required, with a possibility of eliminating it altogether with further advancement in the technology encompassed by the project, and supporting hardware.

A rich and easy-to-use GUI aids the user in navigating the system easily and comprehensively. Features such as multiple searching and viewing options further add to the capabilities of the system and thereby also help in reducing the entry time. Transactions concurrency and their unambiguous nature have been carefully balanced and user sessions are purposefully managed. Direct implementation of the printing code helps the entry clerks as well as managers/administrators, to print the parking slips, reports and user information as and when required.

With a marked difference in the types of users, and user privileges, the system provides a clear hierarchy and shields private data and administration details from those users, who are not permitted or concerned with them.

A lot of data mining options have also been included, in the form of managerial reports. These provide an insight into the daily working subroutines of the system, such as the total revenue collection, employee sign in times, total vehicles parked in the lot, etc. on a Daily, Monthly, or Yearly basis.

The most fundamental factor, that sets this system apart from others, is the fact that it is a real life application, designed with a specific target audience of India, taken into consideration. It makes no assumptions about the perfectionist implementation of the system, and also factors in human tendency.

For instance, it was considered that the system would automatically allocate a parking slot to each vehicle. However, this was ruled out in view of the fact that human error or misdemeanor may lead to a vehicle owner parking elsewhere, rather than the designated spot, which would lead to utter chaos.Similarly, the revenue rates have been formulated keeping the standard manual parking rates in India in mind. However, these can be changed as per requirement as well.

**Introduction**

Everyone who owns or drives a vehicle in India or abroad would be all too familiar with the hassles of finding parking spaces, misbehaving parking attendants, inconsistent or monopolized rates and other problems associated with it.

What is proposed here, is not just another automation of a manual workflow system, it can also be viewed as a solution to the aforementioned problems of the everyday consumer. Rise to the occasion, an Automated Vehicle Parking System.

It not only rids the vehicle owner from the hassles of finding parking spots, it ensures that there is never over or under accommodation of vehicles beyond the lot’s capacity. The system completely eliminated even the possibility of embezzlements. The rates are fixed and predefined. No tampering can be done with the automated calculation of the revenue based on the time taken directly from the console.

It has been built using the Java Swings Framework, in lieu of its rich GUI capabilities, robustness and ease of use. Data collected about vehicle entries is stored in an MS Access Database. It is a console based desktop application that can be configured to run on virtually any PC with a Java Virtual Machine (JVM).

In this report, the complete structure of the system, its design and planning, working and conception, will be outlined in distinct sections. A Waterfall Model approach to Software Engineering was undertaken, and hence the steps are detailed in a similar fashion. First we shall deal with the analysis and planning part of the Software Development Life Cycle (SDLC). Then we go on to explain the Technical Architecture and Methodology adopted in building the system. Next there is an insight to the working and functional requirements of the system. A comprehensive view has been presented n the form of screenshots to give the reader an idea about the GUI and working of the system.

**Objectives**

**Automated Vehicle Parking System** is an innovation that can help solve the ever-challenging problem of parking space limitations in India. It was built to be optimized for use in multilevel parking buildings where a huge number of parking spaces are available, but difficult to keep track of. It was planned keeping the following specific objectives in mind:

1. To provide an efficient, user-friendly, hi-performance, reliable system for implementing the workflow involved in a Multi-Level Parking Lot.
2. To provide vehicle owners a fast, hassle-free experience while saving the time wasted in searching the entire lot for a single parking space.
3. To provide multiple login authorizations and user account types based on functionality.
4. To enable separation of entry and exit terminals and allow addition of supplementary terminals in the future.
5. To provide differentiated vehicle based services at the entry and exit terminals.
6. To accommodate multiple user logins at different terminals simultaneously.
7. To electronically calculate revenue based on pre-defined standard parking rates.
8. To automate the manual workflow, and add speed, efficiency and performance to it.
9. To enable easy report generation and viewing features for the administrator by using data mining techniques.

**Technical ArchitectureFlowcharts**

Barrier is down.

Vehicle Enters.

Enter Vehicle Type.

Check Slot Availability.

Slot Available?

Display Unavailable Message.

**Quit.**

NO

**Begin.**

Enter Vehicle Number.

Generate Parking Slip.

Print Parking Slip.

Raise Barrier.

**Success.**

YES

**Flowchart 1: IN Terminal. New Vehicle Entry**

Barrier is down.

Vehicle Exits.

Submit Slip or Enter Slip Missing in Status.

**Begin.**

Search vehicle By Slip No. or Vehicle No.

Display and Collect Revenue.

**Success.**

Raise Barrier.

Found?

NO

YES

**Flowchart 2: OUT Terminal. Vehicle ExitData Flow Diagrams**

**Clerks**

Vehicle Type

Vehicle No.

**Manager**

**Reports**

**Parking Slips**

Report/User

Details

Slip/Vehicle No.

**0-Level DFD: Context Level Diagram**

**Clerk**

**Manager**

**Parking slips**

Vehicle Type

Vehicle No.

**DATABASE (AVPS)**

Report/User

Details

**Clerk**

Slip/Vehicle No.

**Reports**

**Level 1 DFD**

**Entity-Relationship Diagrams**

**Vehicle**

**Enters**

**At**

**IN Terminal**

**1..\***

**1..\***

**ER Diagram 1**

**Vehicle**

**Leaves From**

**OUT Terminal**

**1..\***

**1..\***

**ER Diagram 2**

**ER Diagram 3**

**PARKING AREA**

VehicleType

Password

UserID

EmpType

EmpName

Status

TotalSlots

VacantSlots

FloorNo

VehicleType

**EMPLOYEE**

**VEHICLE**

VehicleNo

FloorNo

Revenue

SlipNo

TimeOut

TimeIn

SlipStatus

**Enters Details of**

**Enters In/Exits From**

**1**

**1..\***

**1..\***

**1..\***

**Logs Into/ Out From**

**LOGIN**

**1..\***

**1**

*UserID*

*Password*

TerminalNo

EmpInTime

EmpOutTime

**Methodology**

The general approach to automate a manual system is to make the system flexible enough to accommodate future changes in the environment. The methodology should also be incremental progress in this case. Starting with whatever maximum understanding can be gained over a fixed period of time, we start developing the system. The very essence of the process is splitting the system at hand into manageable, fairly understandable and sufficiently complete modules.

Bearing this fact in mind, the system was divided into fairly comprehensive and distinct modules, as follows:

**LOGIN**

**ADMIN**

**CLERK**

**IN TERMINAL**

**OUT TERMINAL**

**PRINT RECIEPT**

**COLLECT REVENUE**

**LOGOUT**

**USERS**

**VIEWS**

**REPORTS**

**Add**

**Edit**

**Delete**

**Available Slots**

**Employee**

**Parking**

**Slips**

**Revenue**

**Vehicles**

**Total Slots**

**Figure: Modules in AVPS**

While carrying out the development of this project, we implemented the WATERFALL MODEL for software development life cycle. It was chosen because of the following reasons:

* It is optimal for use in small projects with pre-defined requirements.
* It has a streamlined and linear approach to development which can help in the project being carried out effectively in the time available.
* It is simple and easy to follow.
* It is relatively cost-saving on inexpensive as compared to other SDLC models.

**Hardware Requirements**

**Minimum Configurations**

* **Processor:**800MHz Intel Pentium III or equivalent
* **Memory:**128 MB
* **Disk space:**20 MB of free disk space

**Recommended Configurations**

* **Processor:**2.6 GHz Intel Pentium IV or equivalent
* **Memory:**256 MB
* **Disk space:**40 MB of free disk space

**Peripheral & Connectivity Devices**

* Network Interface Card *(for LAN Access)*
* Monitor *(for Display)*
* Keyboard *(for Data Entry)*
* Mouse *(for Navigation)*
* Printer *(for Receipts/Reports)*

**software requirements**

**Minimum Software Requirements**

* Operating System: Windows 95/ME/NT/2000 or equivalent
* Java2 Platform Standard Edition v1.4.2 or higher
* Oracle DB or My SQL

**Implications for**

**Future Research**

This application can be easily implemented under various situations. We can add new features as and when we require. Reusability is possible as and when require in this application. There is flexibility in all the modules.

• Extensibility

This software is extendable in ways that one may not have expected at the time of planning. The following principles enhance extensibility like hide data structure, distinguish public and private operations, and further additions like including more terminals, enhancing vehicle types, linkage with other software applications like CRM, Payroll etc.

• Reusability

We can update the software in the next version. Reusable software reduces design, coding and testing cost by amortizing effortover several designs. Reducing the amount of code alsosimplifies understanding, which increases the likelihood that the code is correct. We follow up both types of reusability: Sharing of newly written code within a project and reuse of previously written code on new projects.

•Cost-effectiveness

Its cost is under the budget to begin with, and made within the given time period. It is desirable to aim for a system with a minimum cost subject to the condition that it must satisfy the entire requirement. Scope of this document is to put down the requirements, clearlyidentifying the information needed by the user, the source of the information and outputs expected from the system.

• Understandability

A method is understandable if someone other than the creator of the method can understand the code (as well as the creator after a time lapse). We use the method which is small and coherent, and helps in accomplishing this.

References

**Books:**

1. Anderson, Virginia (2007), The Complete Reference, Microsoft Office Access 2007, 1st Edition (McGraw Hill/Osborne Media)
2. Schildt, Herbert (2002), The Complete Reference, JAVA 2,5thEdition (Tata McGraw Hill/Osborne Media).
3. Sierra, Kathy and Bates, Bert (2003), Head First Java, 2nd Edition (O’Reilly Media)
4. Sommerville, Ian (2010), Software Engineering, 9th Edition (Addison Wesley Publications)

**Websites:**

1. E-books download *- http://scribd.com*
2. Google Search Engine *– http://www.google.com*
3. NetBeans Docs and Support –*http://netbeans.org/kb/*
4. Oracle Sun Developer Network - *http://java.sun.com/developer/*
5. RoseIndia Online Java Tutorials – *www.roseindia.net/*
6. W3Schools Online Web Tutorials - *http://www.w3schools.com/*
7. Wikipedia Online Encyclopedia *- http://en.wikipedia.org*