**PURBANCHAL UNIVERSITY**

**Biratnagar Nepal**

****

A Project report on

**“PARKING MANAGEMENT SYSTEM”**

In the partial fulfillment for the requirement of the 1ST Semester Project-II (subject code- BIT 179CO) in the completion of **Bachelor of Information Technology (BIT)** degree at **KIST college** **of Information Technology**, under **Purbanchal University.**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Submitted By:-** | | |  | **Submitted To:-** | |
| **1)** |  | **Romiya Dangol** |  |  |  |  |
| **2)** |  | **Samira Shahi** | | **Purbanchal University** | | |
|  |  |  | |  | | |

**Under The Guidance of**

**Mr. Prawesh Dhungana**

**Lecturer, BIT**

**KIST COLLEGE OF INFORMATION AND TECHNOLOGY KAMALPOKHARI, KATHMANDU NEPAL**

**KIST COLLEGE OF INFORMATION AND TECHNOLOGY KAMALPOKHARI, KATHMANDU NEPAL**

****

**CERTIFICATE**

This is to certify that the project work entitled **“PARKING MANAGEMENT SYTEM”** is carried out by **ROMIYA DANGOL (5402), SAMIRA SHAHI (5434)** bona fide students of **KIST COLLEGE OF INFORMATION AND TECHNOLOGY** in partial fulfillment for the award of **BACHELOR IN INFORMATION AND TECHNOLOGY** of the **PURBANCHAL UNIVERSITY, BIRATNAGAR NEPAL**, during the year **2021-2022**. It is certified that all corrections indicatedfor internal assessment have been incorporated in the report submitted in the department library. The project report has been approved, as it satisfied the academic requirements in respect of the project work prescribed for the said degree.

The details of the students are as follows:-

|  |  |  |  |
| --- | --- | --- | --- |
| **NAME** | **REGISTRATION NO.** | | **SYMBOL NO.** |
| Romiya Dangol | 058-3-2-04735-2020 | | 313436 |
| Samira Shahi | 058-3-2-04737-2020 | | 313438 |
|  | |
|  | |

Course Semester: - 1ST Semester

Subject: - Project-I

Subject Code: - BIT (106CO)

Mr. Deepak Khadka

Program Coordinator, BIT



**KIST COLLEGE OF INFORMATION AND TECHNOLOGY KAMALPOKHARI, KATHMANDU**

**Examiner’s Certification**

The Project Report

On

**“PARKING MANAGEMENT SYSTEM”**

**Developed by**

**Romiya Dangol**

**Samira Shahi**

Is approved and is acceptable in qualify form.

**Internal Examiner**

Name:

Designation:

**External Examiner**

Name:

Designation:

**ACKNOWLEDGEMENT**

It is with greatest satisfaction and euphoria that we are submitting our project report entitled **“PARKING MANAGEMENT SYSTEM”.** We have completed it as a part of the curriculum of **PURBANCHAL UNIVERSITY.**

We also take this opportunity to express a deep sense of gratefulness to our **BIT Coordinator Mr.** **Deepak Khadka** and **BIT Lecturer Mr. Prawesh Dhungana** for their amiable support, valuableinformation and guidance which helped us in completing this task throughout its various stages. We are indebted to all members of **KIST College,** for the valuable support and suggestion provided by them using their specific fields’ knowledge. We are grateful for their cooperation during the period of our project.

Finally we would also like to express our gratefulness towards **Purbanchal University** for designing such a wonderful course structure. It will help us to get more knowledge in the field of Information Technology & help us to have a bright future in the field of technology.

We hope our university will accept this attempt as a successful project.

Last but not the least, our sincere thanks to our parents, teaching and non-teaching staffs of our college and also my friends.

**ROMIYA DANGOL (313436)**

**SAMIRA SHAHI (313438)**

**STUDENT’S DECLARATION**

We hereby declare that the project report entitled “**PARKING MANAGEMENT SYTEM**” is a result of our own work. If we are found guilty of copying any other report or published information and showing as our original work, we understand that we shall be liable and punishable by **Purbanchal University**.

We further certify that this Project submitted in partial fulfillment of the requirement for the award of Bachelor in Information Technology (**BIT**) of the **Purbanchal University** is our original work and has not been submitted for award of any other degree or other similar title or prize.

|  |  |  |  |
| --- | --- | --- | --- |
| S.N. | Name | Registration No. | Symbol No. |
| 1 | Romiya Dangol | 058-3-2-04735-2020 | 313436 |
| 2 | Samira Shahi | 058-3-2-04737-2020 | 313438 |

**TO WHOM IT MAY CONCERN**

This is to certify that Ms. Romiya Dangol and Ms. Samira Shahi of Bachelor in Information Technology (BIT) has studied as per the curriculum of BIT 1st Semester and completed the project entitled “**Parking Management System”**.This project is the original work of Ms. Romiya Dangol and Ms. Samira Shahi. and was carried out under the supervision of Mr. Prawesh Dhungana as per the guidelines provided by Purbanchal University and certified as per the student’s declaration that project “**Parking Management System**” has not been presented anywhere as a part of any other academic work.

The detail of the student is as follows:

Name of Students : Romiya Dangol

Samira Shahi

Semester : 1st

Subject Code : BIT 178C0

Project Title : **Parking Management System**

…………………………….

Mr. Deepak Khadka  
Program Coordinator, BIT  
KIST College of Information Technology

**TABLE OF CONTENTS**

[ABSTRACT 1](#_Toc94116003)

[Chapter 1 1](#_Toc94116004)

[INTRODUCTION 1](#_Toc94116005)

[1.1 Introduction 1](#_Toc94116006)

[1.2 Problem Statement 1](#_Toc94116007)

[1.3 Objectives 2](#_Toc94116008)

[1.4 Scope 2](#_Toc94116009)

[1.5 Advantages 2](#_Toc94116010)

[Chapter 2 3](#_Toc94116011)

[System Design 3](#_Toc94116012)

[2.1 Algorithm 3](#_Toc94116013)

[2.2 Flowchart 6](#_Toc94116014)

[Chapter 3 19](#_Toc94116015)

[Requirement Analysis and implementation System 19](#_Toc94116016)

[3.1 Requirements 19](#_Toc94116017)

[3.2 System Methodology 20](#_Toc94116018)

[3.2.1. REQUIREMENT ANALYSIS 21](#_Toc94116019)

[3.2.2. SYSTEM DESIGN 22](#_Toc94116020)

[3.2.3. IMPLEMENTATION 25](#_Toc94116021)

[3.2.4. INTEGRATION AND TESTING 25](#_Toc94116022)

[3.2.5. DEPLOYMENT AND MAINTENANCE 25](#_Toc94116023)

[Chapter 4 26](#_Toc94116024)

[Conclusion and future scope 26](#_Toc94116025)

[4.1 Conclusion 26](#_Toc94116026)

[4.2 Future Scope 27](#_Toc94116027)

[REFERENCES 28](#_Toc94116028)

[Chapter 5 29](#_Toc94116029)

[APPENDIX 29](#_Toc94116030)

[5.1 SNAPSHOTS 29](#_Toc94116031)

[*SOURCE CODE* 38](#_Toc94116032)

# ABSTRACT

**“Parking Management System”** is a web based application which is used to keep the records of vehicle which is going to be parked in the parking area

 This application is developed by using C programming language as its backend. It is an easy and time-efficient way of storing and retrieving recorded data. It is based on GUI (Graphical User Interface) so it is easy to use. The objective of our project is to build the better parking system that enables time management and control of vehicles to be parked. The system will track entry and exit of vehicle, maintain the list and will determine the cost. The project is totally built at administrative end and thus only the administrator is guaranteed the access.

The aim is to automate its existing manual system by the help of computerized equipment and full-fledged computer software, fulfilling their requirements, so that it is convenient to the users. Basically the project describes how to manage for good performance and better services for the clients.

# Chapter 1

# INTRODUCTION

### 1.1 Introduction

Parking management system has been developed for managing the bookings and the records of the incoming and outgoing vehicles in a parking house. Its an easy way for generating parking fees.

Nowadays in many public places such as malls, multiplex system, hospitals, offices, market areas there is a crucial problem of vehicle parking. Moreover this involves a lot of manual labor.

The application is reduced as much as possible to avoid errors while entering the data. It also provides error message while entering invalid data. No formal knowledge is needed for the user to use this system. Thus by this all it proves it is user friendly.

### 1.2 Problem Statement

Without too many distinctions, the **“parking war” is a dramatic scene of everyday life.**

The problem faced by the people during parking process is inevitable.Some of the problems are:

* People had no idea about the available slots.
* No backup if the receipt is lost
* Time consuming
* More manual work
* Overcharged parking fees

**“Parking management system** is ideal for overcoming these complexities

### 1.3 Objectives

* To manage the details about vehicles, parking slots, customers, parking fees, duration,bookings.
* To reduce the manual labor and energy.
* To reduce time and increase efficiency of the current Parking Management System.
* To makes parking experience unique and attractive.
* To provides user access to check parking slots available and book as per their convenience
* To generates the receipt that includes parking details ,user details, parking fees etc.

### 1.4 Scope

In the modern age. Vehicle is now a basic need. Every place is under the process of urbanization. There are many corporate offices and shopping centers etc. There are many recreational places where people go for refreshment. So all the places need a parking space where they can park their vehicles easily. Every parking area needs a system that records the detail of vehicles to give the facility. These systems might be computerized or non-computerized. With the help of computerized system, we can deliver a good service to a customer who wants to park their vehicle into the any organization’s premises.

### 1.5 Advantages

The advantages of Parking management system are as follows:

* The main important benefit of parking management system is its advancement and a reliable parking system.
* For parking authorities and vehicle owner, parking management system resources are flexible enough to operate and manage.
* The design and implementation parking management is very easy to supervise and manage. This system can be easily handled by the staff members because of its well organized structure.
* Another unique advantage of a reliable **parking management system** varies in its cost. It is a cost-effective technique of monitoring and manage parking areas. With less manpower, large sum of money is saved. Moreover, it saves manual labor and energy. [1]

# Chapter 2

## System Design

### 2.1 Algorithm

Algorithm is a process or set of rules to be followed in calculation or other problem solving operations, especially by a computer.

Algorithm of the program of parking management system project according to program done in c programing. [2]

STEP 1: START

STEP 2: Press any key to continue,

STEP 3: Enter the username, password,

STEP 4: Compare password with password saved in a file.

STEP 5: If login is successful then press any key to continue,

Else goto step 3,

STEP 6: Show menu,

STEP 7: Enter the choice from menu,

STEP 8:If the choice is a digit from 1 to 11 go to step 9

Else

STEP 8.1: Display "Invalid choice",

STEP 8.2: Goto step 6,

STEP 9: If choice is 1,

STEP 10: Again we go inside the menu of 2 ,

STEP 10.1: If the choice is 1(twowheeler),

STEP 10.1.1: Enter the vehicle plate number,

STEP 10.1.2: Enter the number of customer,

STEP 10.1.3: Enter the phone number of customer,

STEP10.1.4: Enter the date of booking (dd//mm//yy),

STEP 10.1.5: Enter the number of slots do you need,

STEP 10.1.6: Press enter two view available parking,

STEP 10.1.7: Menu of the available parking which

Is available form the time being

STEP 10.1.8: After that enter the parking section,

STEP 10.1.9:Display total bill amount,

STEP 10.1.10: Press enter two view booking slots,

STEP 10.1.11: Enter the seat number,

STEP 10.1.12:Confirm reservation,

If yes then say reservation done,

Else

reservation is not done,

STEP 10.1.13: If the process of choice is done

press any key to go back to main menu,

STEP 10.2: If choice is 2(fourwheeler),

STEP 10.2.1: Enter the vehicle plate number,

STEP 10.2.2: Enter the number of customer,

STEP 10.2.3: Enter the phone number of customer,

STEP10.2.4: Enter the date of booking (dd//mm//yy),

STEP 10.2.5: Enter the number of slots do you need,

STEP 10.2.6:Press enter to view available parking ,

STEP 10.2.7: Menu of the available parking which

is available form the time being,

STEP 10.2.8: After that enter the parking section,

STEP 10.2.9:Display total bill amount,

STEP 10.2.10: Enter the slot number,

STEP 10.2.11:Confirm reservation,

If yes then say reservation ,

Else

Reservation is not done,

STEP 10.2.12: if the process of choice is done press

any key to go back to main menu,

STEP 11: If choice is 2,

STEP 12: Display the total bill amount of the customer ,

STEP 13: If the choice is 3,

STEP 14: Asking the parking section of customer,

If

the data is matched with the saved data it

cancel booking ,

Else

It show that booking is not done to cancel ,

STEP 15: if choice is 4,

STEP 15.1: Add the type and vehicle number

for arrival of vehicle ,

STEP 16: If the choice is 5,

STEP 16.1: Display the total no of vehicle

in the parking ,

STEP 17: If the choice is 6 ,

STEP 17.1: Fill the information that was entered

Before where it shows the total no of two

wheelers parked in parking

STEP 18: If the choice is 7

STEP 18.1: It shows the total number of four

wheelers parked in parking

STEP 19: If the choice is 8

STEP 19.1: It display order in which vechicles

are parked in parking

STEP 20: If the choice is 9

STEP 20.1: Add the type and vehicle number

for departure of vechile ,

STEP 21: If choice is 10 then

STEP 21.1: Check history from the above data to check

STEP 21.2: Again we go inside the menu of 2 ,

STEP 21.2.1:If choice is 1,

STEP 21.2.2:Display the arrival history,

STEP 21.2.3:If choice is 2,

STEP 21.2.4:Display the departure history,

STEP 21.2.5:If choice is 3,

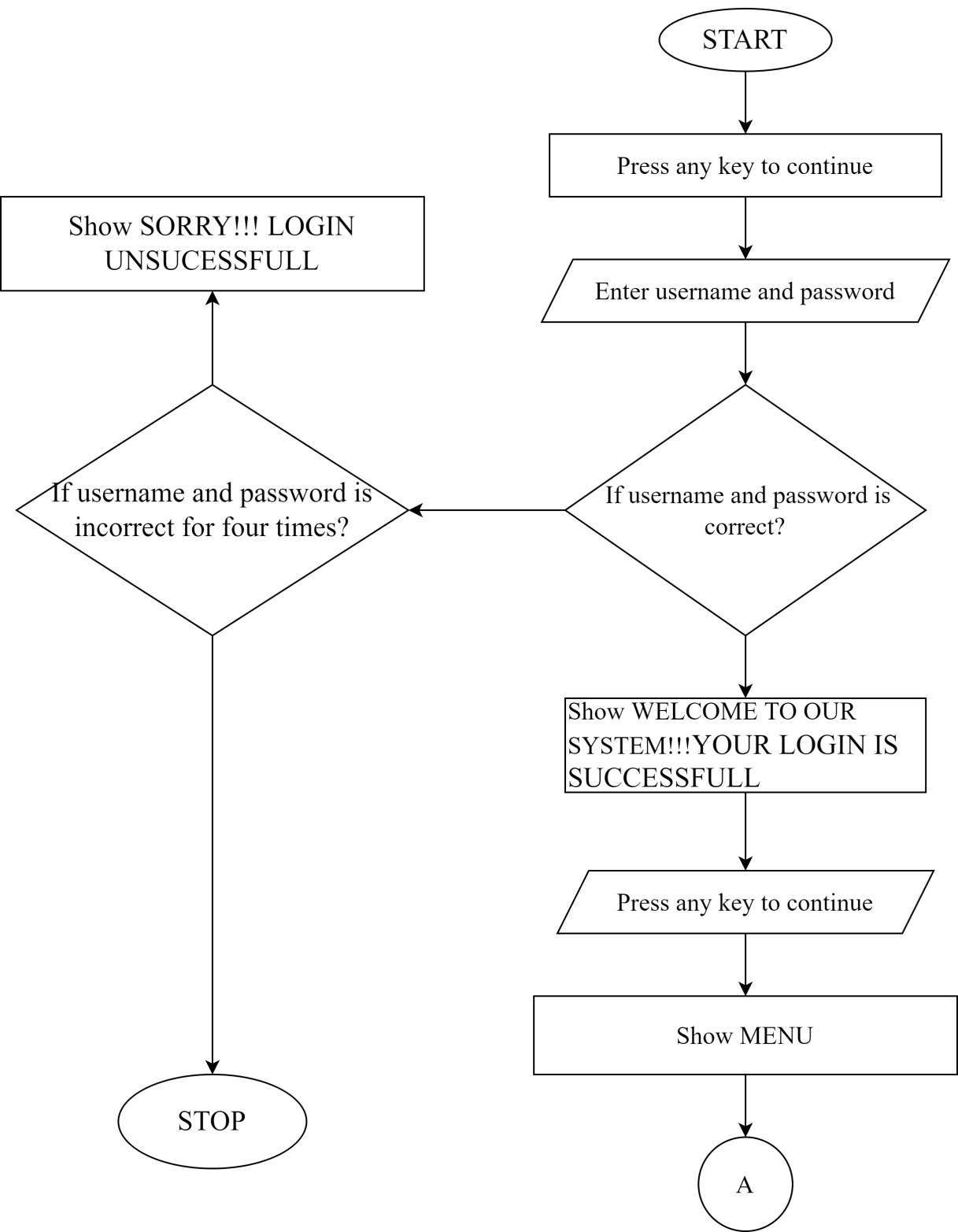
STEP 21.2.6:Display the fine sheet,

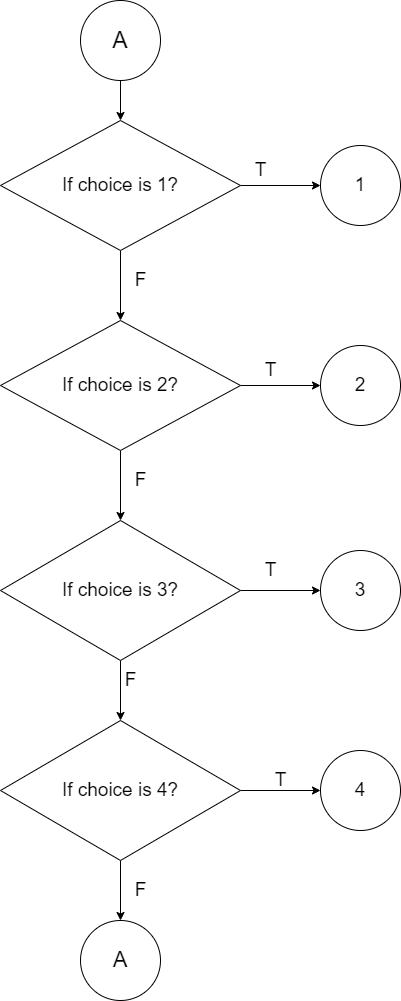
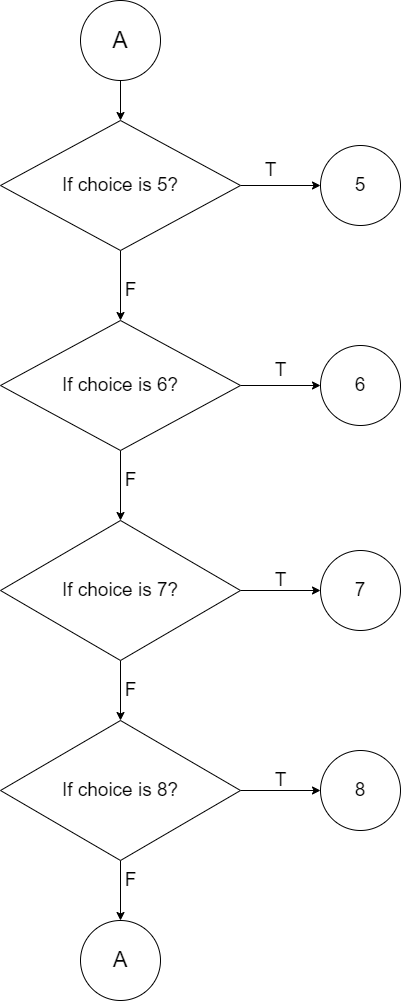
STEP 28: If the choice is 11 then

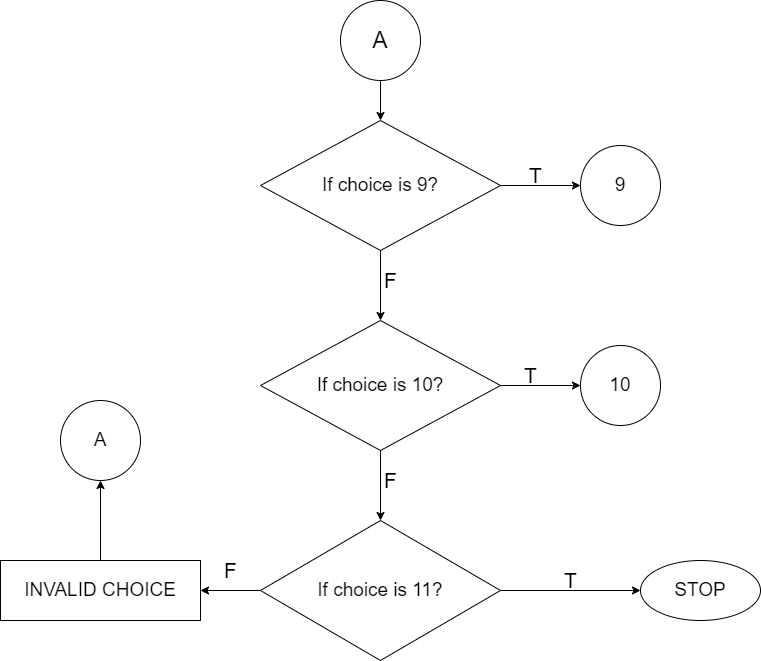
STEP 28.1: then program got exit

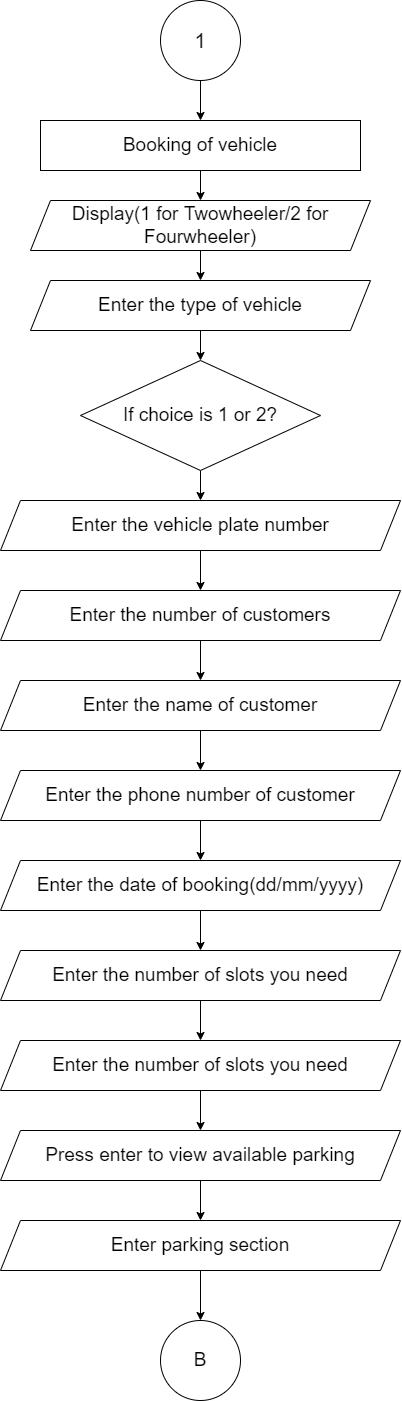
### 2.2 Flowchart

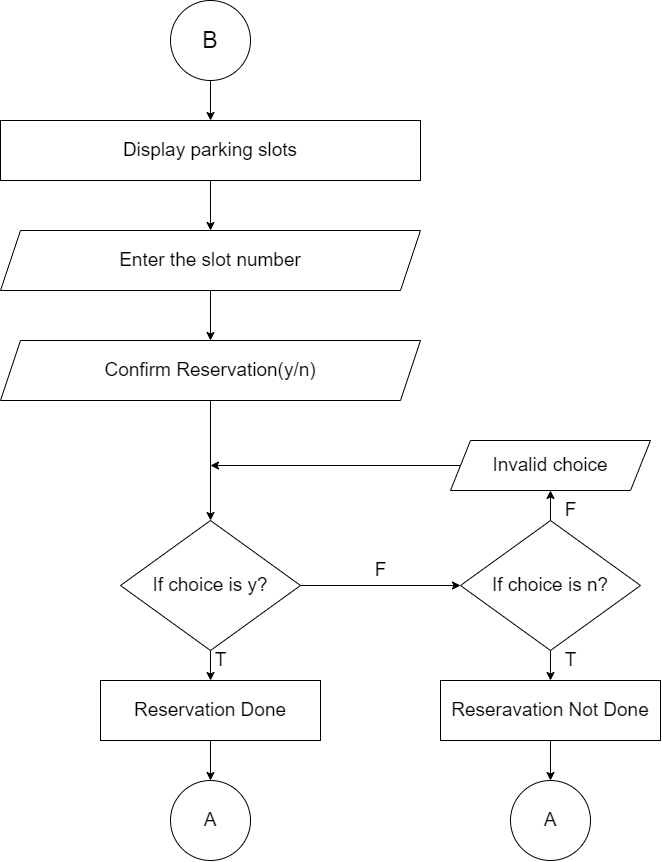
Flow chart is a diagram that represent workflow or process. [2]

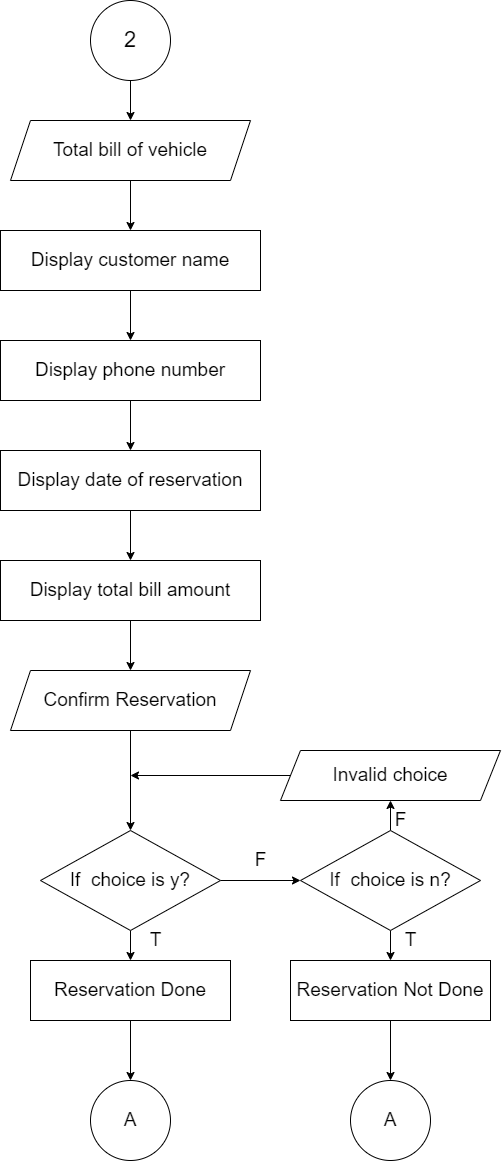
****

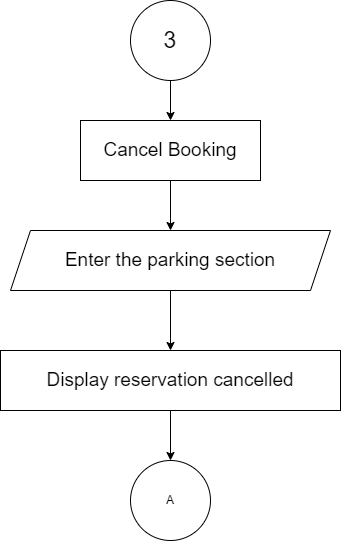




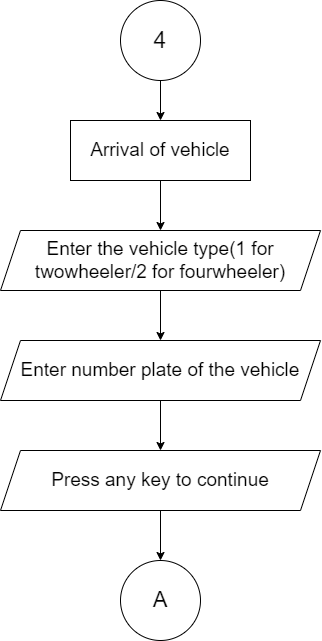




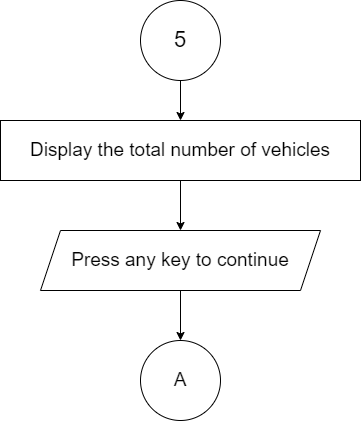


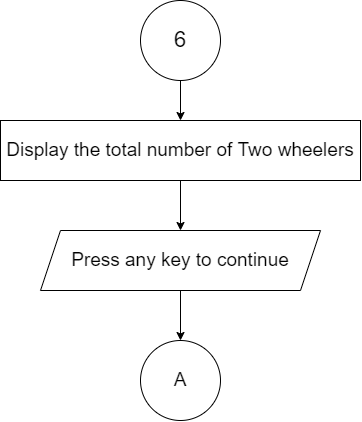


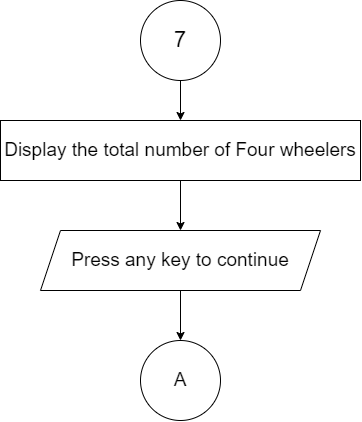
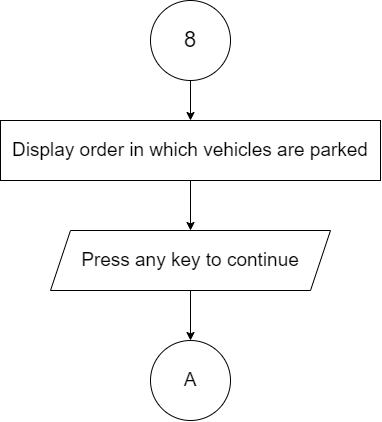
# 

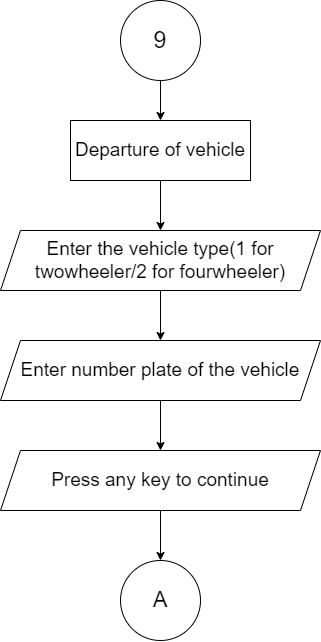


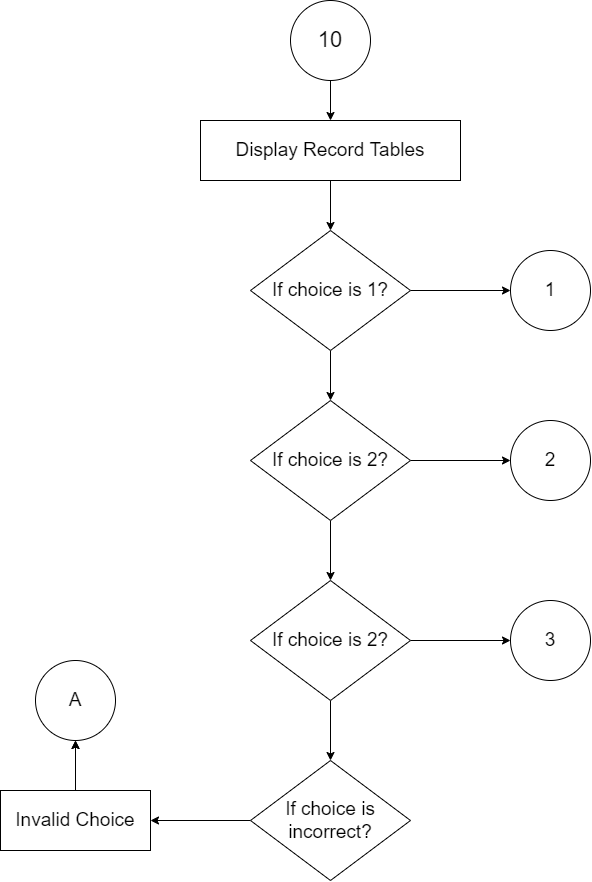
6

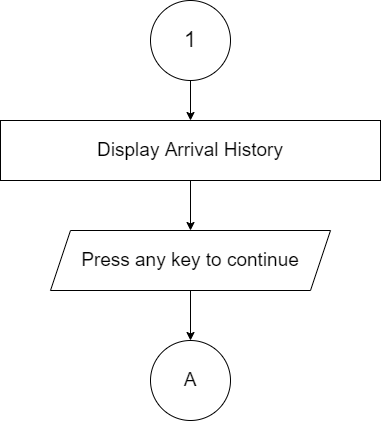


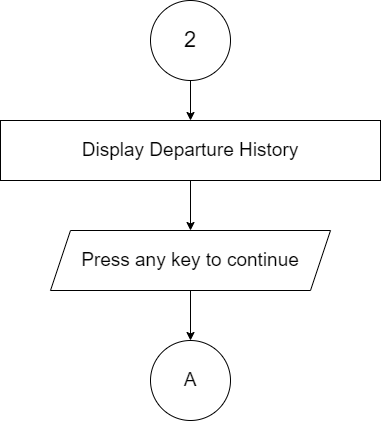


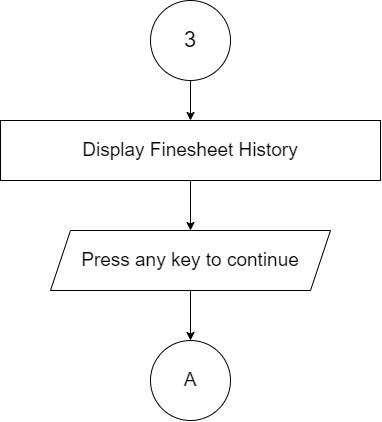












**Fig 1: FLOWCHART**

# Chapter 3

## Requirement Analysis and implementation System

### 3.1 Requirements

Following hardware and software requirement should be met for flawless running

of this system:

**Hardware:**

Hardware is the collection of physical parts of a computer system. This includes the computer case, monitor, keyboard, and mouse. It also includes all the parts inside the computer case, such as the hard disk drive, motherboard, video card, and many others. Computer hardware is what you can physically touch.

RAM: 256 MB

Hard Disk: 200MB or above free space available

Processor: Intel Pentium III or higher

**Software:**

Software is a set of instructions, data or programs used to operate computers and execute specific tasks. It is the opposite of hardware, which describes the physical aspects of a Computer.

OS: Windows 7 or Higher

Applications: Dev C++ or other C++ compiler.

### 3.2 System Methodology

**WATERFALL MODEL**

The Waterfall methodology—also known as the Waterfall model—is a sequential development process that flows like a waterfall through all phases of a project (analysis, design, development, and testing, for example), with each phase completely wrapping up before the next phase begins. [4]

**The sequential phases described in the Waterfall model are:**

Deployment and Maintenance

Integration and Testing

Implementation

System Design

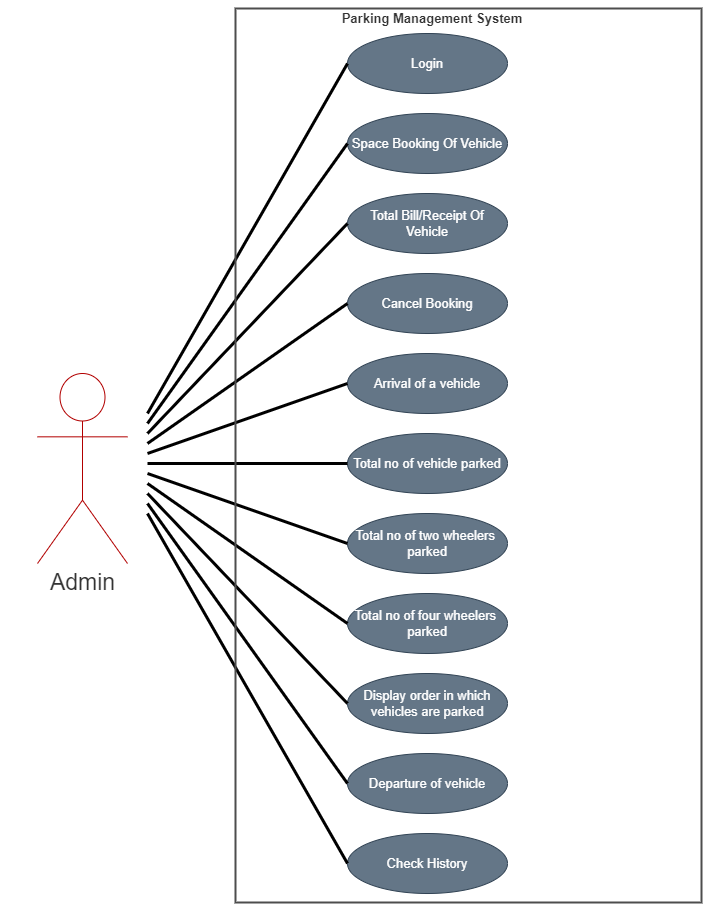
Requirement Analysis

### 3.2.1. REQUIREMENT ANALYSIS

**FUNCTIONAL REQUIREMENT**

In software and system engineering, a functional requirement defines a function of a system or its component, where a function is described as a specification of behavior between input and outputs. [3]

**Fig 2: Use-Case Diagram**



### 3.2.2. SYSTEM DESIGN

This implementation is followed by the next stage in the waterfall model, which is the system design phase. In this section, the requirements that have been analyzed will be translated into detailed design and flowchart of the software code is being created. System design is the important stage that depending on the previous stage to make the great implementation and can be executed properly. When have anything requirements to be insert in designing the code, it will be add up in the requirement analysis phase and the design phase is carried out based on the new set of resources.

* **Importance** :
* If any pre-existing code needs to be understood, organized, and pieced together.
* It is common for the project team to have to write some code and produce original programs that support the application logic of the system.

There are many strategies or techniques for performing system design.

* **Top-down approach:** Top-down integration testing is an integration testing technique used in order to simulate the behavior of the lower-level modules that are not yet integrated. Each system is divided into several subsystems and components. Each of the subsystems is further divided into a set of subsystems and components.
* **Advantages of top-down approach:**
* The main advantage of the top-down approach is that its strong focus on requirements helps to make a design responsive according to its requirements. [4]

### 3.2.2.1 FUNCTIONAL ANALYSIS

|  |  |  |
| --- | --- | --- |
| No. | **Function Module** | Function Description |
| 1. | Login() | This function is for security purpose so that person other than admin cannot manipulate the system or program. |
| 2. | Book() | This function is for space booking for vehicles. |
| 3. | Slot() | This function shows the details of slot in slot matrix for booking. |
| 4. | Bill() | This function is for the printing thereceipt /bill of the parking vehicles. |
| 5. | Viewdetails() | This function is to show the details of available slots with time. |
| 6. | Cancel() | This function allows admin to cancel the booking. |
| 7. | Display() | This function is to display the list of vehicles parked. |
| 8. | **Changecol()** | This function decrements the col. number by one. This function is called, when the data is shifted one place to left |
| 9. | **Randint()** | This function generates random numbers in the range [0, RAND\_MAX). |
| 10. | **Del()** | This function deletes the data of the specified vehicle from the array, if found. |

|  |  |  |
| --- | --- | --- |
| No. | Function Module | Function Description |
| 11. | get\_arrival\_time() | This function is for geting time for arrival of vehicle. |
| 12. | backupw()backupr() | These functions are used for placing of vehicles. |
| 13. | \* add() \* **addonstart()** | These functions are used for adding a data of vehicle. |
| 14. | Getfreerowcol() | This function is to get the row-col position for the vehicle to be parked. |
| 15. | Getrcbyinfo() | This function is to get the row-col position for the vehicle with specified number. |
| 16. | Historyrec() | This function is to record the history of vehicles. |

### 3.2.3. IMPLEMENTATION

With inputs from the system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality, which is referred to as Unit Testing.

* We used C programming to implement our project.
* File Handling was used for the data and records.
* Functions for sub modules.
* The system is first developed in small programs called units, which are integrated in the next phase. The testing of each developed unit individually is referred as unit testing.

### 3.2.4. INTEGRATION AND TESTING

The complete coding will follow by the testing department where it testing based on the functional and non-functional requirements. It checks if there is any problem in the designed software and if it follows the specifications. At this stage, testing activity will include the involvement of computer technician and client. Here, the good flow of the process in designing the software will ensure satisfaction from the client. If there is any problem with the design, it must be reverted back to the system design. Coding and testing are repeated again. [4]

### 3.2.5. DEPLOYMENT AND MAINTENANCE

Once the functional and nonfunctional testing is done, the product  
is deployed in the customer environment or released into the market.  
There are some issues which come up in the client environment. To fix those  
issues patches are released. Also to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment. [4]

# Chapter 4

## Conclusion and future scope

### 4.1 Conclusion

The proposed parking management system takes into account all possible attributes that is expected from it. Our project is only a humble venture to satisfy the needs to manage the project work. Several user friendly coding have also adopted. This package shall prove to be a powerful package in satisfying all the requirements of the parking system.

The developed application is tested with sample inputs and outputs obtained in according to the requirement.The efficiency of the developed system can be enhanced with some minor modifications. Future development can be made in proposed system by integration more services like:

* It can be implemented through web pages.
* New effectives modules can be added time to time

**At the end it is concluded that we have made effort on following points**

* A description of the background and context of the project and its relation to work already done in the area.
* The description of Purpose, Scope, and applicability.
* We define the problem on which we are working in the project.
* We describe the requirement Specifications of the system and the actions that can be done on these things.
* We included features and operations in detail, including screen layouts.
* Finally the system is implemented and tested according to test cases.

### 4.2 Future Scope

It may help collecting perfect management in details. In a very short time, the collection will be obvious, simple and sensible. It also helps in current all works relative to Parking System. It will enhance user experience. Parking is a growing need of the time. Development of this system is very useful in this area of field. We can sell this system to any organization. By using our system, they can maintain records and generate report easily. Our system will be helpful to the excessive need of parking management system in the future.

With the help of computerized system, we can deliver a good service to a customer who wants to park their vehicle into the any organization’s premises..

* + - * + In computer system, it is not necessary to create the manifest but we can directly print it, which saves our time.
        + To utilize resources in an efficient manner by increasing their productivity through automation.
        + It satisfy the user requirement.
        + Be easy to understand by the user and operator.
        + Be easy to operate.
        + Have a good user interface
        + Be expandable
        + Delivered on schedule within the budget. [5]

# References

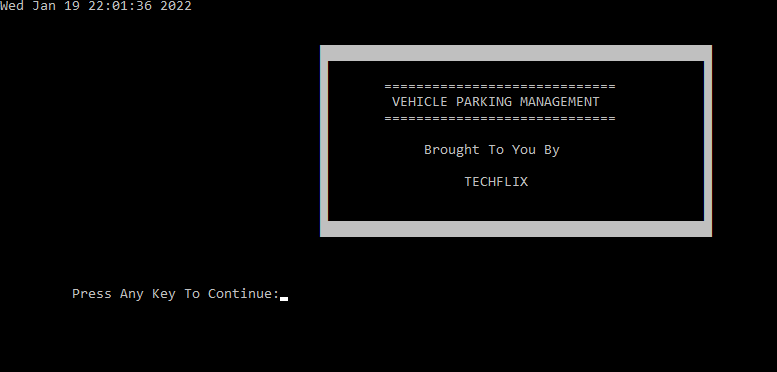
|  |  |
| --- | --- |
| [1] | O. Group, "Omnitec," omnitec Group, 07 december 2015. [Online]. Available: www.omnitecgroup.com/blog/parking-management-system-provides-efficiency-and-advantages-4742. [Accessed 12 1 2022]. |
| [2] | E. Balagurusamy, Programming in ANSI C, ‎ MHE; 6th edition, January 1, 2000. |
| [3] | J. F. G. J. S. V. Jeffrey A. Hoffer, Modern Systems Analysis and Design, ‎ Benjamin-Cummings Pub Co, (May 1, 1996). |
| [4] | R. S.PRESSMAN, SOFTWARE ENGINEERING A PRACTITIONER'S APPROACH FORTH EDITION, Singapore: MC Graw-Hill Book Co-Singapore, 1997. |
| [5] | g. kamaraj, "Academia.edu- Share reasearch," 2016. [Online]. Available: www.academia.edu/36410792/Parking\_Management\_System\_Parking\_Management\_System. [Accessed 12 1 2020]. |

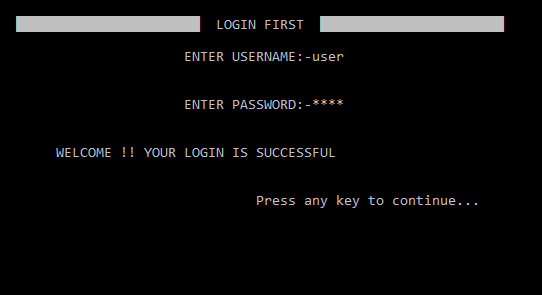
# Chapter 5

## APPENDIX

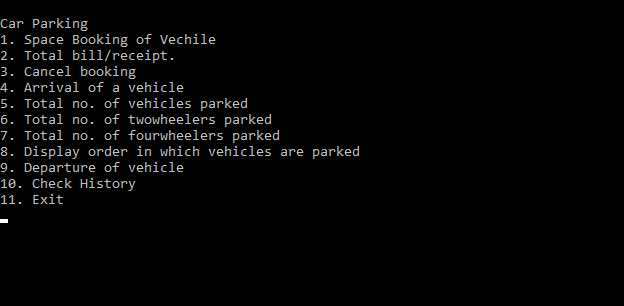
### 5.1 SNAPSHOTS

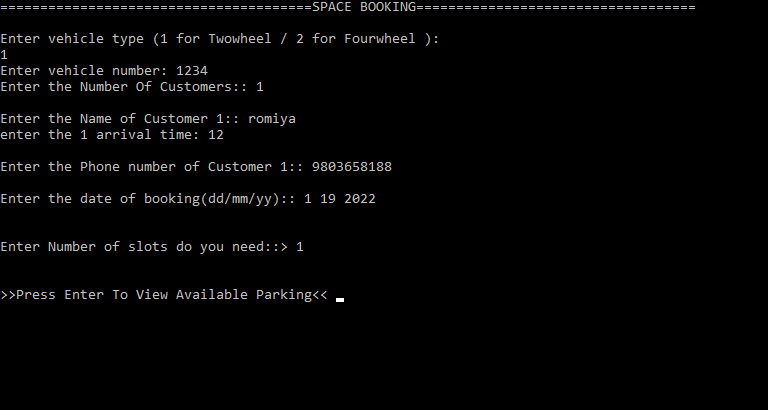
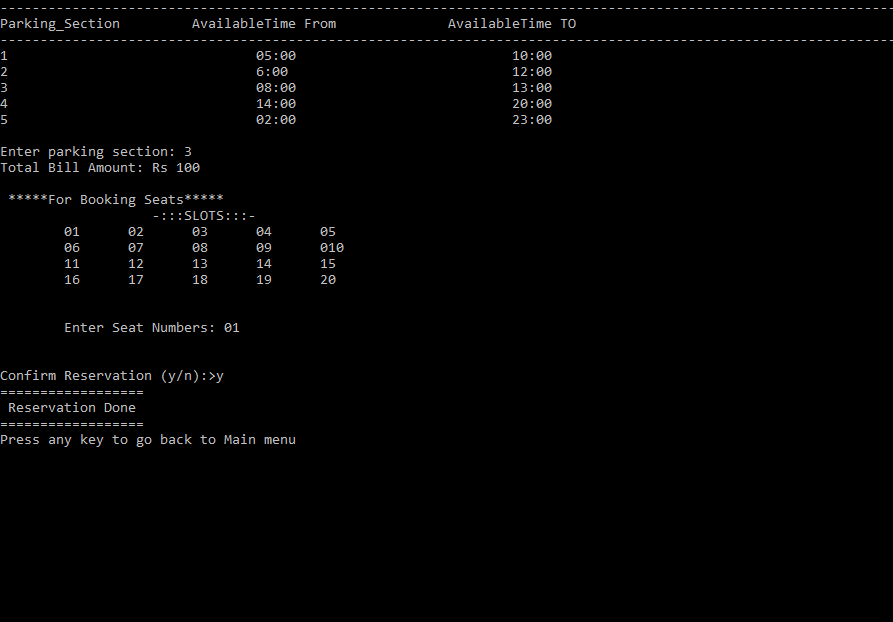
**WELCOME PAGE**



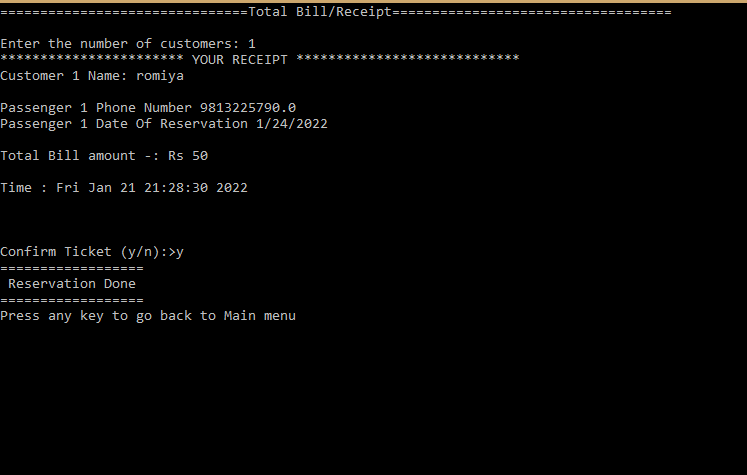
**LOGIN PAGE**

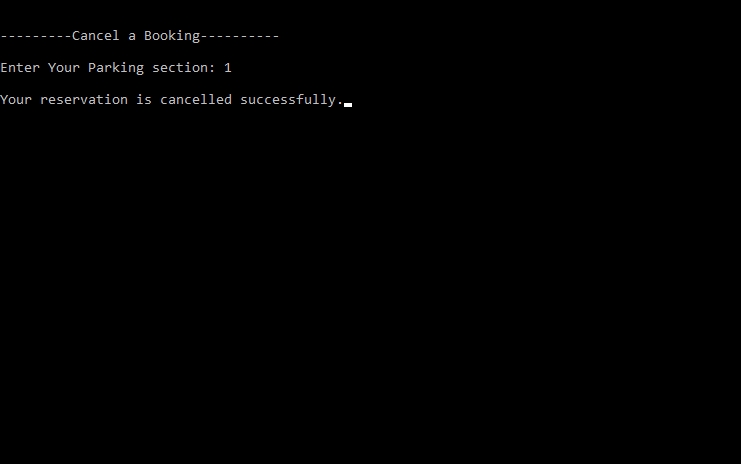
**MAIN MENU**



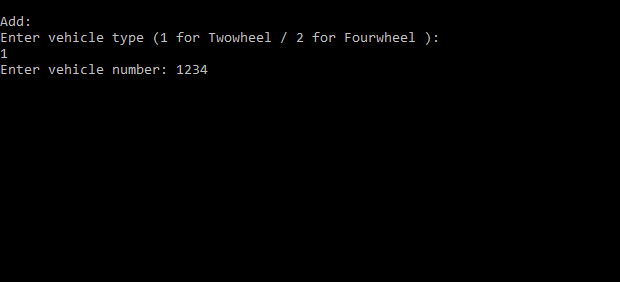
**SPACE BOOKING**

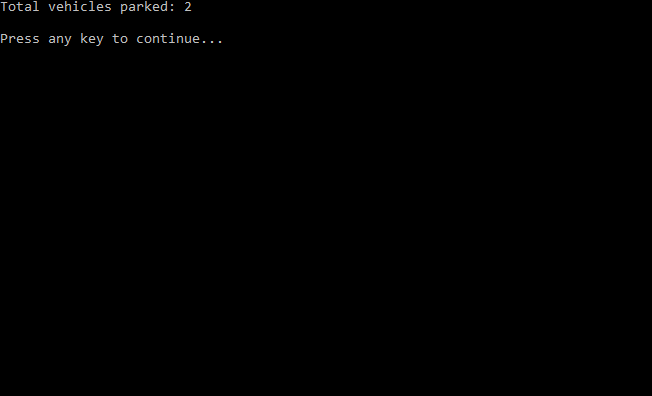
**TOTAL BILL/RECEIPT**

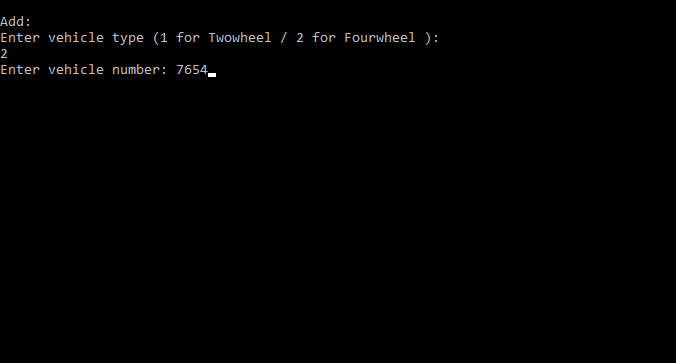


**CANCEL BOOKING**

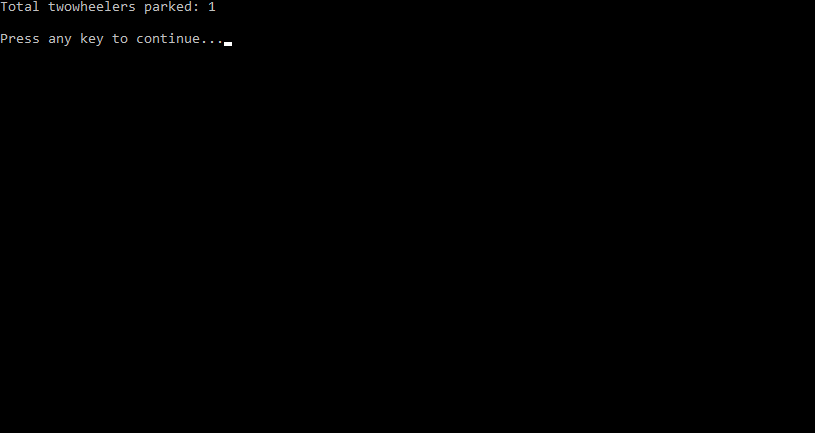
**ARRIVAL OF VEHICLE**



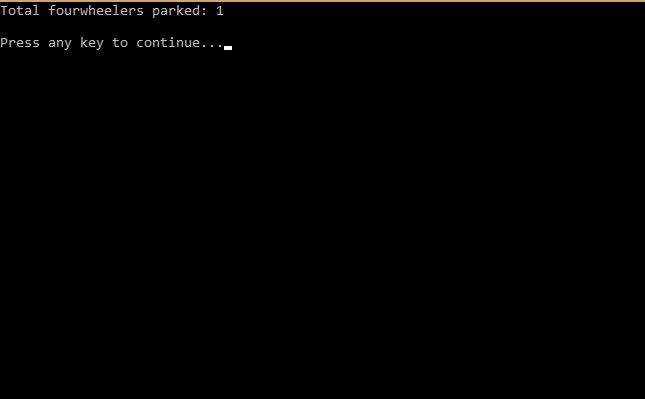
**TOTAL NO. OF VEHICLE**

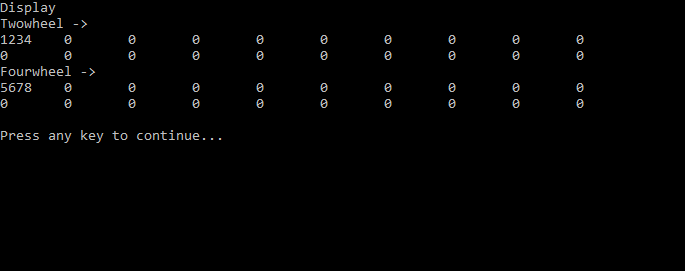


**TOTAL NO. OF TWO WHEELERS**

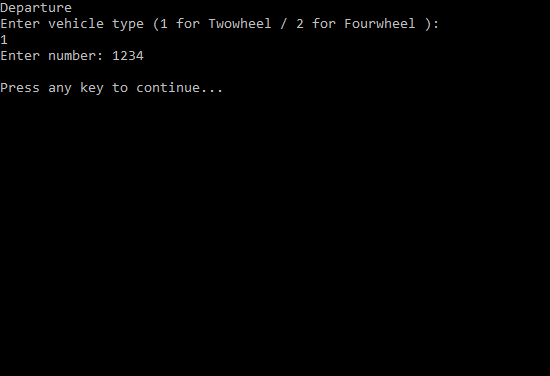


**TOTAL NO. OF FOURWHEELERS**



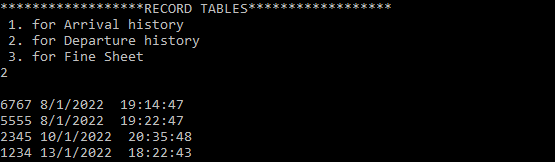
**DISPLAY ORDER IN WHICH VEHICLES ARE PARKED**

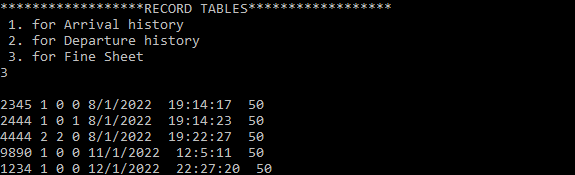
**DEPARTURE OF VEHICLES**



**CHECK HISTORY**

**1.ARRIVAL HISTORY**

**2. DEPARTURE HISTORY**

**3. FINESHEET**

## *SOURCE CODE*

//PARKING MANAGEMENT SYSTEM

#include <stdio.h>

#include <conio.h>

#include <time.h>

#define TWOWHEEL 1

#define FOURWHEEL 2

#include <stdlib.h>

int time1, time2, time3,time4,a[55];

int j;

struct pd

{

char name[20];

int age,d,m,year;

double Phno;

int Park\_num;

float arr\_time;

int num\_of\_slots;

}parkdetails[10],pd2;

struct billpay

{

int bill\_amount;//

}bp;

struct vehicle

{

int num ;

int row ;

int col ;

int type ;

struct tm \*at;

} ;

struct vehicle \*twowheel[2][10] ;

struct vehicle \*fourwheel[2][10] ;

int parkinfo[4][10] ;

int vehcount ; /

int twowheelcount ;

int fourwheelcount ;

void details(void);

int slot(int);

void book(void);

void bill();

void viewdetails(void);

void login();

void cancel(void);

void display( ) ;

void changecol ( struct vehicle \* ) ;

void get\_arrival\_time(int);

int randint();

int backupw();

int backupr();

struct vehicle \* add ( int, int, int, int ) ;

struct vehicle \* addonstart ( int, int, int, int ) ;

void login();

void del ( struct vehicle \* ) ;

void getfreerowcol ( int, int \* ) ;

void getrcbyinfo ( int, int, int \* ) ;

void show() ;

void total\_bill(int ,struct tm \*ct1,struct tm \*ct2);

void changecol ( struct vehicle \*v )

{

v -> col--;

}

struct tm\* datetime()//at=datetime();

{

time\_t ts;

ts = time(NULL);

ct = localtime(&ts);

return ct ;

}

int record(int veh,int type,int row,int col, struct tm \*ct )

{

FILE \*fp;

fp = fopen("D://arival.dat", "a");

if (fp == NULL)

{

printf("Error!File does not exists \n");

return 0 ;

}

fprintf(fp,"\n");

fprintf(fp,"%d ",veh);

fprintf(fp,"%d ",type);

fprintf(fp,"%d ",row);

fprintf(fp,"%d ",col);

fprintf(fp,"%d/%d/%d\n", ct->tm\_mday,

ct->tm\_mon + 1, ct->tm\_year + 1900); fprintf(fp,"%d:%d:%d\n", ct->tm\_hour,

ct->tm\_min, ct->tm\_sec);

fclose(fp);

}

int record2(int veh, struct tm \*ct )

{

FILE \*fp;

fp = fopen("D://depart.dat", "a");

if (fp == NULL)

{

printf("Error!File does not exists \n");

return 0 ;

}

fprintf(fp,"\n");

fprintf(fp,"%d ",veh);

fprintf(fp,"\n%d/%d/%d", ct->tm\_mday, ct->tm\_mon + 1, ct->tm\_year + 1900);

fprintf(fp,"%d:%d:%d", ct->tm\_hour, ct->tm\_min, ct->tm\_sec);

fclose(fp);

}

void get\_arrival\_time(int num)

{

int veh ,type,row,col ;

int mon,day,year,hour,min,sec;

int mon2,day2,year2,hour2,min2,sec2;

int mon1,day1,year1,hour1,min1,sec1;

int mon3,day3,year3,hour3,min3,sec3;

FILE \*fp;

fp= fopen("D://arival.dat", "r");

if (fp == NULL)

{

printf("Error!File does not exists \n");

getch();

}

while(!feof(fp))//(fgetc(fp))!=EOF)

{

fscanf(fp,"\n");

fscanf(fp,"%d ",&veh);

fscanf(fp,"%d ",&type);

fscanf(fp,"%d ",&row);

fscanf(fp,"%d ",&col);

fscanf(fp,"%d/%d/%d ", &day, &mon, &year);

fscanf(fp,"%d:%d:%d ", &hour, &min, &sec);

if(veh == num)

{

day2 = day;

mon2 = mon;

year2 = year;

hour2 = hour;

min2 = min;

sec2 = sec ;

}

}

fclose(fp);

FILE \*fp1;

fp1 = fopen("D://depart.dat", "r");

if (fp1 == NULL)

{

printf("Error!File does not exists \n");

getch() ;

}

while(!feof(fp1))//(fgetc(fp1))!=EOF)

{

fscanf(fp1,"\n");

fscanf(fp1,"%d ",&veh);

fscanf(fp1,"%d/%d/%d ",&day1, &mon1, &year1);

fscanf(fp1,"%d:%d:%d ", &hour1, &min1, &sec1);

if(veh == num)

{

day3 = day1;

mon3 = mon1;

year3 = year1;

hour3 = hour1;

min3 = min1;

sec3 = sec1 ;

}

}

fclose(fp1);

int tsec1,tsec2,tsec3;

tsec1 = sec3;

tsec1 += min3\*60;

tsec1 += (hour3\*60)\*60;

tsec2 = sec2;

tsec2 += min2\*60;

tsec2 += (hour2\*60)\*60;

tsec3 = tsec1 - tsec2 ;

int second , minute ,hours;

int secc , temp1,temp2,temp3,temp4 ;

second = tsec3 % 60 ;

temp1 = tsec3 - second ;

temp2 = temp1/60 ;

minute = temp2 % 60 ;

temp4 = temp2 - minute ;

hours = temp4 /60 ;

printf("you have parked your vehicle for %d/%d/%d",hours,minute,second);

}

int randint()//generate random numbers in the range [0, RAND\_MAX).

{

int r ;

srand(time(NULL));//The srand() function sets the starting point for producing a series of pseudo-random integers.

r = rand() % 20;

return r ;

}

void finesheet(int veh,int type,int row,int col, struct tm \*ct)

{

FILE \*fp;

fp= fopen("D://finesheet.dat", "a");

if (fp == NULL)

{

printf("Error!File does not exists \n");

getch();

}

fprintf(fp,"\n");

fprintf(fp,"%d \n",veh);

fprintf(fp,"%d ",type);

fprintf(fp,"%d ",row);

fprintf(fp,"%d ",col);

fprintf(fp,"%d/%d/%d ",ct->tm\_mday, ct->tm\_mon + 1, ct->tm\_year + 1900);

fprintf(fp,"%d:%d:%d ", ct->tm\_hour, ct->tm\_min, ct->tm\_sec);

fprintf(fp,"%d ",50);

fclose(fp);

}

int historyrec(int value)

{

FILE \*fp;

if(value==1)

{

fp = fopen("D://arival.dat", "r");

}

else if(value ==2)

{

fp = fopen("D://depart.dat", "r");

}

else if(value==3)

{

fp = fopen("D://finesheet.dat", "r");

}

else

{

printf("invalid input");

return 0 ;

}

if (fp == NULL)

{

printf("File does not exists \n");

return 0 ;

}

char ch;

ch = getc(fp);

while(ch!=EOF)

{

printf("%c",ch);

ch = getc(fp);

}

return 0 ;

}

int backupw()

{

int r,c ;

FILE \*fp;

fp = fopen("backupw.dat", "w");

if (fp == NULL)

{

printf("Error!File does not exists \n");

return 0 ;

}

for(r=0;r<4;r++)

for(c=0;c<10;c++)

{

fprintf(fp,"\n");

fprintf(fp,"%d ",parkinfo[r][c]);

fprintf(fp,"%d ",r);

fprintf(fp,"%d ",c);

fprintf(fp,"%d ",vehcount);

fprintf(fp,"%d ",twowheelcount);

fprintf(fp,"%d ",fourwheelcount);

}

fclose(fp);

}

int backupr()

{

int r,c ;

int park[4][10];

FILE \*fp;

fp = fopen("backupr.dat", "r");

if (fp == NULL)

{

printf("Error!File does not exists \n");

return 0 ;

}

int rr ,cc ,veh,twowheeler,fourwheeler,numb;

for(r=0;r<4;r++)

{

for(c=0;c<10;c++)

{

fscanf(fp,"\n");

fscanf(fp,"%d ",&park[r][c]);

numb= park[r][c];

fscanf(fp,"%d ",&rr);

fscanf(fp,"%d ",&cc);

fscanf(fp,"%d ",&veh);

fscanf(fp,"%d ",&twowheeler);

fscanf(fp,"%d ",&fourwheeler);

if(numb!=0)

{

if ( r == 0 || r == 1 )

twowheel[r][c] = addonstart(1,numb,r,c);

else

fourwheel[r][c] = addonstart(2,numb,r,c);

}

}

fclose(fp);

}

}

struct vehicle \* addonstart ( int t, int num, int row, int col )

{

struct vehicle \*v ;

int r ;

v = ( struct vehicle \* ) malloc ( sizeof ( struct vehicle ) ) ;

/\*malloc: used to dynamically allocate a single large block of memory with the specified size.

It returns a pointer of type void which can be cast into a pointer of any form.\*/

v -> type = t ;

v -> row = row ;

v -> col = col ;

if ( t == TWOWHEEL )

twowheelcount++ ;

else

fourwheelcount++ ;

vehcount++ ;

parkinfo[row][col] = num ;

v->at = datetime();

record(num,t,row,col,datetime());

backupw(t);

r = randint();

if(r < 5 )

{

printf("\a");

printf("\a");

finesheet(num,t,row,col,datetime());

}

return v ;

}

void login()

{

int a=0,i=0;

char uname[10],c=' ';

char pword[10],code[10];

char user[10]="user";

char pass[10]="pass";

do

{

printf("\n \xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb LOGIN FIRST \xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb ");

printf(" \n\n ENTER USERNAME:-");

scanf("%s", &uname);

printf(" \n\n ENTER PASSWORD:-");

while(i<10)

{

pword[i]=getch();

c=pword[i];

if(c==13) break;

else printf("\*");

i++;

}

pword[i]='\0';

//char code=pword;

i=0;

//scanf("%s",&pword);

if(strcmp(uname,"user")==0 && strcmp(pword,"pass")==0)

{

printf(" \n\n\n WELCOME !! YOUR LOGIN IS SUCCESSFUL");

printf("\n\n\n\t\t\t\tPress any key to continue...");

getch();//holds the screen

break;

}

else

{

printf("\n SORRY !!!! LOGIN IS UNSUCESSFUL");

a++;

getch();//holds the screen

}

}

while(a<=2);

if (a>2)

{

printf("\nSorry you have entered the wrong username and password for four times!!!");

getch();

}

system("cls");

}

void del ( struct vehicle \*v )

{

int c ;

for ( c = v -> col ; c < 9 ; c++ )

{

parkinfo[v -> row][c] = parkinfo[v -> row][c+1] ;

}

parkinfo[v -> row][v -> col] = 0 ;

if ( v -> type == TWOWHEEL )

twowheelcount-- ;

else

fourwheelcount-- ;

vehcount-- ;

backupw();

}

void getrowcol ( int type, int \*arr )

{

int r, c, fromrow = 0, torow = 2 ;

if ( type == FOURWHEEL )

{

fromrow += 2 ;

torow += 2 ;

}

for ( r = fromrow ; r < torow ; r++ )

{

for ( c = 0 ; c < 10 ; c++ )

{

if ( parkinfo[r][c] == 0 )

{

arr[0] = r ;

arr[1] = c ;

return ;

}

}

}

if ( r == 2 || r == 4 )

{

arr[0] = -1 ;

arr[1] = -1 ;

}

}

void getrcbyinfo ( int type, int num, int \*arr )

{

int r, c, fromrow = 0, torow = 2 ;

if ( type == FOURWHEEL )

{

fromrow += 2 ;

torow += 2 ;

}

for ( r = fromrow ; r < torow ; r++ )

{

for ( c = 0 ; c < 10 ; c++ )

{

if ( parkinfo[r][c] == num )

{

arr[0] = r ;

arr[1] = c ;

return ;

}

}

}

if ( r == 2 || r == 4 )

{

arr[0] = -1 ;

arr[1] = -1 ;

}

}

void display( )

{

int r, c ;

printf ( "Twowheel ->\n" ) ;

for ( r = 0 ; r < 4 ; r++ )

{

if ( r == 2 )

printf ( "Fourwheel ->\n" ) ;

for ( c = 0 ; c < 10 ; c++ )

printf ( "%d\t", parkinfo[r][c] ) ;

printf ( "\n" ) ;

}

}

int main( )

{

int j,n;

time\_t currentTime;

time(&currentTime);

printf("%s\n",ctime(&currentTime));

printf("\n\t\t\t\t\t\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\n");

printf("\t\t\t\t\t\xdb \xdb\n");

printf("\t\t\t\t\t\xdb ======================= \xdb\n");

printf("\t\t\t\t\t\xdb VEHICLE PARKING MANAGEMENT \xdb\n");

printf("\t\t\t\t\t\xdb ============================= \xdb\n");

printf("\t\t\t\t\t\xdb \xdb\n");

printf("\t\t\t\t\t\xdb Brought To You By \xdb\n");

printf("\t\t\t\t\t\xdb \xdb\n");

printf("\t\t\t\t\t\xdb TECHFLIX \xdb\n");

printf("\t\t\t\t\t\xdb \xdb\n");

printf("\t\t\t\t\t\xdb \xdb\n");

printf("\t\t\t\t\t\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\xdb\n\n\n");

printf(" \n\t Press Any Key To Continue:");

getch();

system("cls");

login();

int choice, type, number, row = 0, col = 0 ;

int i, tarr[2] ;

int finish = 1 ;

backupr();

system ( "cls" ) ;

/\* displays menu and calls corresponding functions \*/

while ( finish )

{

system ( "cls" ) ;

printf ( "\nCar Parking\n" ) ;

printf ( "1. Space Booking of Vehicle\n" ) ;

printf ( "2. Total bill/receipt.\n" ) ;

printf ( "3. Cancel booking\n" ) ;

printf ( "4. Arrival of a vehicle\n" ) ;

printf ( "5. Total no. of vehicles parked\n" ) ;

printf ( "6. Total no. of twowheelers parked\n" ) ;

printf ( "7. Total no. of fourwheelers parked\n" ) ;

printf ( "8. Display order in which vehicles are parked\n" ) ;

printf ( "9. Departure of vehicle\n" ) ;

printf ( "10. Check History\n" ) ;

printf ( "11. Exit\n" ) ;

scanf ( "%d", &choice ) ;

switch ( choice )

{

case 1 :

system("cls");

book();

break;

case 2 :

system("cls");

bill();

break;

case 3 :

system("cls");

cancel();

break;

case 4 :

system ( "cls" ) ;

printf ( "\nAdd: \n" ) ;

type = 0 ;

while ( type != TWOWHEEL && type != FOURWHEEL )

{

printf ( "Enter vehicle type (1 for Twowheel / 2 for Fourwheel ): \n" ) ;

scanf ( "%d", &type ) ;

if ( type != TWOWHEEL && type != FOURWHEEL )

printf ( "\nInvalid vehicle type.\n" ) ;

}

printf ( "Enter vehicle number: " ) ;

scanf ( "%d", &number ) ;

system ( "cls" ) ;

if ( type == TWOWHEEL || type == FOURWHEEL )

{

getrowcol ( type, tarr ) ;

if ( tarr[0] != -1 && tarr[1] != -1 )

{

row = tarr[0] ;

col = tarr[1] ;

if ( type == TWOWHEEL )

twowheel[row][col] = addonstart ( type, number, row, col ) ;

else

fourwheel[row - 2][col] = addonstart ( type, number, row, col ) ; ;

}

else

{

if ( type == TWOWHEEL )

printf ( "\nNo parking slot free to park a twowheel\n" ) ;

else

printf ( "\nNo parking slot free to park a fourwheel\n" ) ;

}

}

else

{

printf ( "Invalid type\n" ) ;

break ;

}

printf ( "\nPress any key to continue..." ) ;

getch( ) ;

break ;

case 5 :

system ( "cls" ) ;

printf ( "Total vehicles parked: %d\n", vehcount ) ;

printf ( "\nPress any key to continue..." ) ;

getch( ) ;

break ;

case 6 :

system ( "cls" ) ;

printf ( "Total twowheelers parked: %d\n", twowheelcount ) ;

printf ( "\nPress any key to continue..." ) ;

getch( ) ;

break ;

case 7 :

system ( "cls" ) ;

printf ( "Total fourwheelers parked: %d\n", fourwheelcount ) ;

printf ( "\nPress any key to continue..." ) ;

getch( ) ;

break ;

case 8 :

system ( "cls" ) ;

printf ( "Display\n" ) ;

display( ) ;

printf ( "\nPress any key to continue..." ) ;

getch( ) ;

break ;

case 9 :

system ( "cls" ) ;

printf ( "Departure\n" ) ;

type = 0 ;

/\* check for vehicle type \*/

while ( type != TWOWHEEL && type != FOURWHEEL )

{

printf ( "Enter vehicle type (1 for Twowheel / 2 for Fourwheel ): \n" ) ;

scanf ( "%d", &type ) ;

if ( type != TWOWHEEL && type != FOURWHEEL )

printf ( "\nInvalid vehicle type.\n" ) ;

}

printf ( "Enter number: " ) ;

scanf ( "%d", &number ) ;

if ( type == TWOWHEEL || type == FOURWHEEL )

{

getrcbyinfo ( type, number, tarr ) ;

if ( tarr[0] != -1 && tarr[1] != -1 )

{

col = tarr [1] ;

/\* if the vehicle is Twowheel \*/

if ( type == TWOWHEEL )

{

row = tarr [0] ;

del ( twowheel [row][col] ) ;

for ( i = col ; i < 9 ; i++ )

{

twowheel[row][i] = twowheel[row][i + 1] ;

// changecol ( twowheel[row][col] ) ;

}

free ( twowheel[row][i] ) ;

twowheel[row][i] = NULL ;

}

/\* if a vehicle is Fourwheel \*/

else

{

row = tarr[0] - 2 ;

if ( ! ( row < 0 ) )

{

del ( fourwheel[row][col] ) ;

for ( i = col ; i < 9 ; i++ )

{

fourwheel[row][i] = fourwheel[row][i + 1] ;

//changecol ( fourwheel[row][col] ) ;

}

fourwheel[row][i] = NULL ;

}

}

}

else

{

if ( type == TWOWHEEL )

printf ( "\nInvalid Twowheel number, or a twowheel with such number has not been parked here.\n" ) ;

else

printf ( "\nInvalid Fourwheel number, or a fourwheel with such number has not been parked here.\n" ) ;

}

}

printf ( "\nPress any key to continue..." ) ;

getch( ) ;

break ;

case 11 :

system ( "cls" ) ;

for ( row = 0 ; row < 2 ; row++ )

{

for ( col = 0 ; col < 10 ; col++ )

{

if ( twowheel[row][col] -> num != 0 )

free ( twowheel[row][col] ) ;

if ( fourwheel[row][col] -> num != 0 )

free ( fourwheel[row+2][col] ) ;

}

}

finish = 0 ;

getch();

break ;

case 10 :

system("cls");

int option ;

printf ( "RECORD TABLES\n" ) ;

printf ( " 1. for Arrival history\n" ) ;

printf ( " 2. for Departure history\n" ) ;

printf ( "3. for Fine Sheet\n" ) ;

scanf ( "%d", &option ) ;

historyrec(option);

printf ( "\nPress any key to continue..." ) ;

getch( ) ;

break ;

}

}

return 0;

}

void book()

{

int i,age,m,year,d;

int num\_of\_seats;

int park\_section;

char confirm,y,n;

int c;

int x=0;

char name[20];

FILE \*fp,\*fq;

fp=fopen("D:\\Parking\_Booking.txt","r+");

if (fp==NULL)

{

fp=fopen("D:\\Parking\_Booking.txt","w+");

if (fp==NULL)

{

puts("File cannot be opened ");

exit(0);

}

}

fq=fopen("Parking\_Booking1.txt","r+");

if (fq==NULL)

{

fq=fopen("Parking\_Booking1.txt","w+");

if (fq==NULL)

{

puts("File cannot be opened ");

exit(0);

}

}

system("cls");

int type,number;

int num\_of\_slots;

printf("=========SPACEBOOKING=======\n\n");

while ( type != TWOWHEEL && type != FOURWHEEL)

{

printf ( "Enter vehicle type (1 for Twowheel / 2 for Fourwheel ): \n" ) ;

scanf ( "%d", &type ) ;

if ( type != TWOWHEEL && type != FOURWHEEL )

printf ( "\nInvalid vehicle type.\n" ) ;

}

printf ( "Enter vehicle number: " ) ;

scanf ( "%d", &number ) ;

printf("Enter the Number Of Customers:: ");

fflush(stdin);

scanf("%d", &j);

for (i = 0; i < j; i++)

{

printf("\nEnter the Name of Customer %d:: ", i+1);

fflush(stdin);

gets(parkdetails[i].name);

printf("enter the %d arrival time: ",i+1);

fflush(stdin);

scanf ( "%f", &parkdetails[i].arr\_time ) ;

printf("\nEnter the Phone number of Customer %d:: ", i+1);

fflush(stdin);

scanf("%lf",&parkdetails[i].Phno);

printf("\nEnter the date of booking(dd/mm/yy):: ");

fflush(stdin);

scanf("%d/%d/%d",&parkdetails[i].d,&parkdetails[i].m,&parkdetails[i].year);

}

fwrite(&parkdetails,sizeof(parkdetails),1,fp);

printf("\n\nEnter Number of slots do you need::> ");

fflush(stdin);

scanf("%d",&num\_of\_slots);

printf("\n\n>>Press Enter To View Available Parking<< ");

getch();

system("cls");

viewdetails();

label:

printf("\n\nEnter parking section: ");

scanf("%d",&park\_section);

do

{

switch(park\_section)

{

case 1:

{

time1 = 5;

time2 = 00;

time3 = 10;

time4 = 00;

printf("Total Bill Amount: Rs 50 \n",d);

}break;

case 2:

{

time1 = 6;

time2 = 00;

time3 = 12;

time4 = 00;

printf("Total Bill Amount: Rs 50 \n",d);

} break;

case 3:

{

time1 = 8;

time2 = 00;

time3 = 13;

time4 = 00;

printf("Total Bill Amount: Rs 50 \n",d);

} break;

case 4:

{

time1 = 14;

time2 = 10;

time3 = 20;

time4 = 00;

printf("Total Bill Amount: Rs 50 \n",d);

} break;

case 5:

{

time1 = 2;

time2 = 00;

time4 = 23;

time4 = 00;

printf("Total Bill Amount: Rs 50 \n",d);

} break;

default:

printf("Enter Correct choice.....\n");

goto label;

}

}

while(x);

{

bp.bill\_amount=50;

fwrite(&bp,sizeof(bp),1,fq);

printf("\n \*\*\*\*\*For Booking Seats\*\*\*\*\*\n");

// Calling slot() function with number of passenger

slot(j);

printf("\n\nConfirm Reservation (y/n):>");

start:

scanf(" %c",&confirm);

if(confirm == 'y')

{

printf("==================");

printf("\n Reservation Done\n");

printf("==================");

printf("\nPress any key to go back to Main menu");

}

else

{

if(confirm=='n'){

printf("\nReservation Not Done!\nPress any key to go back to Main menu!");

}

else

{

printf("\nInvalid choice entered! Enter again-----> ");

goto start;

}

}

}

fclose(fp);

fclose(fq);

getch();

}

void bill()

{

char confirm;

time\_t currentTime;

time(&currentTime);

int i,c;

FILE \*fp,\*fq;

fp=fopen("Parking\_Booking.txt","r+");

if (fp==NULL)

{

fp=fopen("Parking\_Booking.txt","w+");

if (fp==NULL)

{

puts("File cannot be opened ");

exit(0);

}

}

fq=fopen("Parking\_Booking1.txt","r+");

if (fq==NULL)

{

fq=fopen("Parking\_Booking2.txt","w+");

if (fq==NULL)

{

puts("File cannot be open2ed ");

exit(0);

}

}

printf("===========TotalBill/Receipt==\n\n");

printf("Enter the number of customers: ");

scanf("%d",&j);

fread(&parkdetails,sizeof(parkdetails),1,fp);

fread(&bp,sizeof(bp),1,fq);

printf("\*\*\*\*\*\*\*\*\* YOUR RECEIPT\*\*\*\*\*\*\*\*\*\*\*\*\*");

for (i = 0; i < j; i++)

{

printf("\t\t\nCustomer %d Name: ", i+1);

puts(parkdetails[i].name);

printf("\t\t\nPassenger %d Phone Number %.1lf",i+1,parkdetails[i].Phno);

printf("\t\t\nPassenger %d Date Of Reservation %d/%d/%d",i+1,parkdetails[i].d,parkdetails[i].m,parkdetails[i].year);

}

printf("\t\t\n\nTotal Bill amount -: Rs %d",bp.bill\_amount);

printf("\t\t\n\nTime : %s",ctime(&currentTime));

printf("\n");

printf("\n\nConfirm Ticket (y/n):>");

start:

scanf(" %c",&confirm);

if(confirm == 'y')

{

printf("==================");

printf("\n Reservation Done\n");

printf("==================");

printf("\nPress any key to go back to Main menu");

}

else

{

if(confirm=='n'){

printf("\nReservation Not Done!\nPress any key to go back to Main menu!");

}

else

{

printf("\nInvalid choice entered! Enter again-----> ");

goto start;

}

}

fclose(fp);

fclose(fq);

getch();

}

void viewdetails(void)

{

system("cls");

printf("----------------------------------------------------------------------------------------------------------------");

printf("\nParking\_Section\t\tAvailableTime From\t\tAvailableTime TO\n");

printf("----------------------------------------------------------------------------------------------------------------");

printf("\n1\t\t\t\t05:00\t\t\t\t10:00");

printf("\n2\t\t\t\t6:00\t\t\t\t12:00");

printf("\n3\t\t\t\t08:00\t\t\t\t13:00");

printf("\n4\t\t\t\t14:00\t\t\t\t20:00");

printf("\n5\t\t\t\t02:00\t\t\t\t23:00");

}

int slot(int p)

{

int i;

printf("\t -:::SLOTS:::- \n");

printf("\t01\t02\t03\t04\t05\n");

printf("\t06\t07\t08\t09\t010\n");

printf("\t11\t12\t13\t14\t15\n");

printf("\t16\t17\t18\t19\t20\n");

printf("\n\n\tEnter slot Numbers: ");

for (i = 0; i < p; i++)

scanf("%d", &a[i]);

}

void cancel(void)

{

int sec;

system("cls");

printf("\n\n---------Cancel a Booking----------\n");

label:

printf("\nEnter Your Parking section: ");

scanf("%d",&sec);

if(sec>=1 && sec<=5)

{

printf("\nYour reservation is cancelled successfully.");

}

else

{

printf("\nPlease enter a valid booking number to cancel reservation!");

goto label;

}

getch();

}