

Automated Parking System

MLB\_09.01\_07

Malabe

14/05/2022





1 | P a g e

We declare that this is our own work, and this Assignment does not incorporate without acknowledgment any material previously submitted by anyone else in SLIIT or any other university/Institute. And we declare that each one of us equally contributed to the completion of this Assignment.

|  |  |  |
| --- | --- | --- |
| **Registration No** | **Name** | **Contact Number** |
|  |  |  |
| IT21175398 | Godage D.N.P | 0715589299 |
|  |  |  |
| IT21176210 | Asmitha Thiraviyarasa | 0777189479 |
|  |  |  |
| IT21176142 | Abiramy Kumaresan | 0765767994 |
|  |  |  |
| IT21173936 | Kodithuwakku K.G.K.M.J | 0775959865 |
|  |  |  |
| IT21174612 | G.G.P.N. De Davin | 0714044749 |
|  |  |  |





2 | P a g e

**Introduction**

An automated parking system enables people to acquire a parking space in a much more convenient way. Every day people spend a lot of time searching for a place to park their vehicles.

Therefore, they need to search for available places to park their vehicles and it is only a waste of time and fuel and it also increases the traffic congestion. This automated parking system is dedicated for customers to check available parking area and book them for a limited time. We have enhanced the service to provide customers with the solution to lack of parking space in modern cities.

The automated parking system is dedicated for customers to check available parking and reserve for a certain period of time. There are two types of users who can access this system: registered and unregistered. The system allows users to both view and use the system for reservations. Users who have registered can gain access to the system by logging in. If unregistered users wish to register, they can do so by signing up for the system.

Users can choose and reserve a preferred parking space ahead of time. The parking fee is calculated based on the locations and time slots chosen by the users. Users can pay using any online transaction method they prefer. To complete the payment, a QR code is generated. Users must contact support service agents to update reservations



3 | P a g e

**User Requirements**

1. The width of viewing and accessing the system is different according to the user.
2. Unregistered users can view the details about the site, which are parking availability, cost, about us, and ratings.
3. Registered users can gain access to the availability of the parking slots, schedule, reservation, cost, ratings, customer support, feedback, and complaints.
4. Admin of the system can view, edit, update, and delete the system information. And also contact customers and responded to customers.
5. When registering to the system as a new user, if the user is a customer, the customer should register as a customer to the site by providing the required details. (Name, contact number, E-mail address, Date of Birth, address)
6. Users can log in to the site by using their email/username and password.
7. Once logged in to the system as a user, the customer can get the service of the parking spot by choosing the date, time, and location, and making the payment done.
8. Payment can be done online or on any other preferred method. (Credit, debit, cash)
9. Once the customer completes the process, the system will provide a receipt with a QR Code.
10. Customers can give their complaints, feedback, and suggestions using the feedback page.
11. Customers can get online help using the system.
12. System admin can generate reports such as daily/monthly financial reports.



4 | P a g e

**Noun/Verb Analysis**

1. The width of viewing and accessing the system is different according to the user.
2. Unregistered users can view the details about the site, which are parking availability, cost, about us, and ratings.
3. Registered users can gain access to the availability of the parking slots, schedule, reservation, cost, ratings, customer support, feedback, and complaints.
4. Admin of the system can view, edit, update and delete the system information. And also contact customers, and responded to customers.
5. When registering to the system as a new user, if the user is a customer, the customer should register as a customer to the site by providing the required details. (Name, contact number, E-mail address, Date of Birth, address)
6. Users can log in to the site by using their email/username and password.
7. Once logged in to the system as a user, the customer can get the service of the parking spot by choosing the date, time, and location, and making the payment done.
8. Payment can be done online or on any other preferred method. (Credit, debit, cash)
9. Once the customer completes the process, the system will provide a receipt with a QR Code.
10. Customers can give their complaints, feedback, and suggestions using the feedback page.
11. Customers can get online help using the system.
12. System admin can generate reports such as daily/monthly financial reports.



5 | P a g e

**Noun Analysis**

Identified classes:

* + - **CLASSES**
      * **Unregistered User**
      * **Registered user**
      * **Parking slots**
      * **Reservation**
      * **Feedback**
      * **Admin**
      * **Payment**

**Reasons for rejecting other nouns**

* + - **OUT OF SCOPE**
      * **System**
      * **Customers**
      * **New user**
      * **Online help**
      * **location**
    - **REDUNDANT** 
      * **Width**
      * **User**
      * **Site**
      * **Parking Availability**
      * **Access**
      * **Availability**
      * **Schedule**
      * **Cost**
      * **Customers support**
      * **Complaints**
      * **System Information**
      * **Service**
      * **Parking spot**
      * **Online**
      * **Preferred method**
      * **Process**
      * **Suggestions**
      * **Feedback page**



6 | P a g e

* + - **ATTRIBUTE**
      * **Details**
      * **Cost**
      * **Rating**
      * **Name**
      * **Contact number**
      * **Email address**
      * **Date of birth**
      * **Address**
      * **Email/username**
      * **Password**
      * **Date**
      * **Time**
      * **Credit**
      * **Debit**
      * **Cash**
      * **Receipt**
      * **QR Code**



7 | P a g e

**Verb Analysis**

**Unregistered Customer**

* Viewing
* Accessing
* Registering
* Providing
* Get
* Choosing
* Making
* Done
* Completes
* Give

**Registered Customer**

* Gain
* Log in
* Using
* Get
* Choosing
* Making
* Done
* Completes
* Give

**Admin**

* View
* Edit
* Update
* Delete
* Contact
* Responded
* Provide
* Generate



8 | P a g e

**CRC Cards**

|  |  |
| --- | --- |
| **Class: Admin** |  |
| **Responsibility** | **Collaborations** |
| Login |  |
| Search customer details | Customer |
| View parking details |  |
| Register Person Details |  |
| Update person Details |  |
| View person Details |  |
| Delete Person Details |  |

|  |  |
| --- | --- |
| **Class: Payment** |  |
| **Responsibility** | **Collaborations** |
| Store the payment details |  |
| Put the reservation through online transaction |  |
| Registered customers can do their month-to-month payments at the end of the month |  |
| Will get discounts |  |
| Show Payment details |  |



9 | P a g e

|  |  |
| --- | --- |
| **Class: Parking Locations** |  |
| **Responsibility** | **Collaborations** |
| Input parking location |  |
| Expel parking location |  |
| Update parking location |  |

|  |  |
| --- | --- |
| **Class: Inquires** |  |
| **Responsibility** | **Collaborations** |
| Input User Details |  |
| Add Inquiry |  |
| Update Inquiry |  |
| Show Inquiry |  |

|  |  |
| --- | --- |
| **Class Name: Reservations** |  |
| **Responsibility** | **Collaborations** |
| Input reservations details | Payment, location |
| Update reservation details |  |
| Show reservation details |  |



10 | P a g e

|  |  |
| --- | --- |
| **Class: Feedback** |  |
| **Responsibility** | **Collaborations** |
| Input User Details |  |
| Add Feedback |  |
| Show Feedback |  |

|  |  |
| --- | --- |
| **Class: Reports** |  |
| **Responsibility** | **Collaborations** |
| Create financial report | Customer |
| Show reports |  |

|  |  |
| --- | --- |
| **Class: Unregistered Customer** |  |
| **Responsibility** | **Collaborations** |
| Register Details |  |
| Search parking Location |  |
| View parking Location | Parking Location |
| Reserve parking Location | Reservation |
| Pay the payment for parking | Payment |
| Create Inquiry | Inquiry |
| Provide Feedback | Feedback |



11 | P a g e

|  |  |
| --- | --- |
| **Class: Registered Customer** |  |
| **Responsibility** | **Collaborations** |
| Login |  |
| Search parking Location |  |
| View parking Location | Parking Location |
| Reserve parking Location | Reservation |
| Pay the payment for parking | Payment |
| Get Discount |  |
| Create Inquiry | Inquiry |
| Provide Feedback | Feedback |



12 | P a g e

**Class Diagram**



**Diagram

Description automatically generated**

13 | P a g e

**Codes**

**Customer.h**

class Customer {

protected: char name[20];

char email[30];

char num[11];

char code[5];

public: void setDetails(char const pname, char const pemail,char const pnum, char const pcode);

char getDetails();

Customer() {}

Customer(char const pname[], char const pemail[], char const pnum[], char const pcode[]);

void displayDetails();

};

class Unregistered\_customer: public Customer{

private: char userID[6];

public:Unregistered\_customer( char const pname[], char const pemail[],char const pnum[],char const pcode[]);

void displayunregDetails();

char getDetails();

};

class Registered\_customer: public Customer{

private: char userID[6];

char password[8];

float discount;

public:Registered\_customer(char const ID[],char const pemail[], char const ppw[], float disc);

void displayregDetails();

char getDetails();

};





14 | P a g e

**Customer.cpp**

#include <iostream>

#include<cstring>

#include "Customer.h"

using namespace std;

Customer::Customer(char const pname[], char const pemail[], char const pnum[], char const pcode[]){

strcpy( name,pname);

strcpy( email, pemail);

strcpy ( num, pnum);

strcpy (code, pcode);

}

void Customer::displayDetails(){

cout<< name <<endl;

cout<< email <<endl;

cout<< num <<endl;

cout<< code <<endl;

cout<<endl;

}

Unregistered\_customer::Unregistered\_customer(char const pname[],char const pemail[],char const pnum[],char const pcode[]){

strcpy( name,pname);

strcpy( email, pemail);

strcpy ( num, pnum);

strcpy (code, pcode);

};

void Unregistered\_customer::displayunregDetails(){

cout<<"Name: "<< name <<endl;

cout<<"Email Address" <<email <<endl;

cout<<"Phone Number: "<< num <<endl;

cout<<"Postal Code: "<< code <<endl;

cout<<endl;

}

Registered\_customer::Registered\_customer(char const ID[], char const pemail[],char const ppw[], float disc){

strcpy( userID,ID);

strcpy( email, pemail);

strcpy ( password, ppw);

discount= disc;

};



15 | P a g e

void Registered\_customer::displayregDetails(){

cout<<"User ID: "<< userID <<endl;

cout<<"Email Address: "<< email <<endl;

cout<<"Password: "<< password <<endl;

cout<<"Discount Amount: Rs."<< discount <<endl;

cout<<endl;

}

**Main.cpp**

//Customer codes

#include <iostream>

#include<cstring>

#include "Customer.h"

using namespace std;

int main()

{

cout<<"Unregistered Customers " <<endl<<endl;

Unregistered\_customer UC1("Kavin", "kavin@yahoo.com", "0717777770", "11111"); UC1.displayunregDetails();

Unregistered\_customer UC2("Prasa", "prasa@gmail.com", "0717777770", "22222");

UC2.displayunregDetails();

Unregistered\_customer UC3("Asmi", "asmi@yahoo.com", "0776543210", "33333");

UC3.displayunregDetails();

Unregistered\_customer UC4("Nandun", "Nandun@hotmail.com", "0767865432", "44444");

UC4.displayunregDetails();

cout<<"Registered Customers " <<endl<<endl;

Registered\_customer RC1("RG001", "kamal@yahoo.com", "zxcv8n7", 100.00); RC1.displayregDetails();

Registered\_customer RC2("RG002", "nimal@gmail.com", "as3h6gf", 150.00 ); RC2.displayregDetails();

Registered\_customer RC3("RG003", "sunil@gmail.com", "qwe6rty", 100.00); RC3.displayregDetails();

Registered\_customer RC4("RG004", "gayan@hotmail.com", "wsde5rf", 125.00);

RC4.displayregDetails();

}

16 | P a g e

**Feedback.h**

class Feedback

{

private:

int feedbackNo;

char name[25];

char email[50];

char feedbackMsg[100];

public:

void feedback(int fbNo, char userName[], char userEmail[], char

addFeedback[]);

void displayFeedback();

};

**Feedback.cpp**

#include "Feedback.h"

#include <iostream>

#include <cstring>

using namespace std;

// Assign User Details and Feedback Message

void Feedback::feedback(int fbNo, char userName[], char userEmail[], char addFeedback[]) {

int feedbackNo = fbNo;

strcpy(name, userName);

strcpy(email, userEmail);

strcpy(feedbackMsg, addFeedback);

}

// Display User Details and Feedback Message

void Feedback::displayFeedback() {

cout << "Feedback No : " << feedbackNo << endl;

cout << "Customer Name : " << name << endl;

cout << "Email : " << email << endl;

cout << "Feedback : " << feedbackMsg << endl;

}



17 | P a g e

**Inquiry.h**

class Inquiry

{

private:

char uName[25];

int contactNo;

char uEmail[50];

char addEnquire[100];

public:

void enquire(char username[], int cNo, char useremail[], char addEnq[]);

void displayEnquire();

};

**Inquiry.cpp**

#include "Inquiry.h"

#include <iostream>

#include <cstring>

using namespace std;

// Assign User Details and Inquiry Details

void Inquiry::enquire(char username[], int cNo, char useremail[], char addEnq[]){

strcpy(uName, username);

int contactNo = cNo;

strcpy(uEmail, useremail);

strcpy(addEnquire, addEnq);

}

// Display User Details and Inquiry Details

void Inquiry::displayEnquire(){

cout << "Customer Name : " << uName << endl;

cout << "Contact Number : " << contactNo << endl;

cout << "Email : " << uEmail << endl;

cout << "Inquiry : " << addEnquire << endl;

}



18 | P a g e

**Main.cpp**

//Feedback codes

#include <iostream>

#include "Feedback.h"

#include "Inquiry.h"

using namespace std;

int main()

{

Feedback fb;

fb.feedback(17, (char\*)"Saman Perera", (char\*)"samanPerera@gmail.com", (char\*)"Great Solution"); //Create Feedback object

fb.displayFeedback(); //Display Feedback

cout << "\n" << endl;

Inquiry inq;

inq.enquire((char\*)"Dasun Nethsara", 717128328, (char\*)"Nethsara@gmail.com", (char\*)"Change Password"); //Create Inquiry object

inq.displayEnquire(); //Display Inquiry

return 0;

}

**Payment.h**

class Payment{

private:

int p\_ID;

char p\_Type[25];

char p\_Status[20];

public:

void payment(int pP\_ID, char pP\_Type[], char pP\_Status[]);

void displayPaymentDetails();

void addDetails();

void getUpdatePaymentDetails();

char removeDetails();

};

**Payment.cpp**

#include<iostream>

#include<cstring>

#include"payment.h"

using namespace std;

void Payment::payment(int pP\_ID, char pP\_Type[], char pP\_Status[]){

p\_ID = pP\_ID;



19 | P a g e

strcpy (p\_Type ,pP\_Type);

strcpy(p\_Status ,pP\_Status);

}

void Payment::displayPaymentDetails(){ cout<<"Payment ID: "<<p\_ID<<endl; cout<<"Payment Type: "<<p\_Type<<endl; cout<<"Payment Status: "<<p\_Status<<endl;

}

void Payment::getUpdatePaymentDetails()

{

}

**Park Location.h**

class ParkLocation {

private:

int Location\_ID;

char Location\_Name[20];

char Details[25];

double Price;

public:

void parkLocation(int pLocation\_ID, char pLocation\_Name[], char pDetails[], double pPrice);

void addLocation();

void displayLocationDetails();

void deleteLocation();

void searchLocation();

};

**Park Location.cpp**

#include<iostream>

#include<cstring>

#include"ParkLocation.h"

using namespace std;

void ParkLocation::parkLocation(int pLocation\_ID, char pLocation\_Name[], char

pDetails[], double pPrice)

{

Location\_ID = pLocation\_ID;

strcpy(Location\_Name, pLocation\_Name);

strcpy(Details, pDetails);

Price = pPrice;

}

void ParkLocation::addLocation()

{

}

20 | P a g e

void ParkLocation::displayLocationDetails()

{

cout<<"Location ID: "<<Location\_ID<<endl;

cout<<"Location Name: "<<Location\_Name<<endl;

cout<<"Details: "<<Details<<endl;

cout<<"Price: "<<Price<<endl;

}

**Main.cpp**

//Pay & Location codes

#include<iostream>

#include<cstring>

#include"payment.h"

#include"ParkLocation.h"

using namespace std;

int main()

{

Payment pay;

pay.payment(12,(char\*)"PayPal",(char\*)"Done"); pay.displayPaymentDetails();

cout<<"--------------------------------"<<endl;

ParkLocation parkL;

parkL.parkLocation(0001,(char\*)"NugegodaPark", (char\*)"Nugegoda Highlevel Rd, Nugegoda", 500.00);

parkL.displayLocationDetails();

return 0;

}



21 | P a g e

**Admin.h**

class Admin

{

private:

int AdminID;

char name[25];

public:

void Admins(int ADNo, char ADName[]);

void displayAdmin();

};

**Admin.cpp**

#include "Admin.h"

#include <iostream>

#include <cstring>

using namespace std;

void Admin::Admins(int ADNo, char ADName[])

{

int AdminID = ADNo;

strcpy(name, ADName);

}

void Admin::displayAdmin()

{

cout << "ADMIN ID : " << AdminID << endl;

cout << "ADMIN Name : " << name << endl;

}



22 | P a g e

**Report.h**

class Report

{

private:

int ReportNo;

char name[25];

public:

void Reports(int rpNo, char reportName[]);

void displayReport();

};

**Report.cpp**

#include "Report.h"

#include <iostream>

#include <cstring>

using namespace std;

void Report::Reports(int rpNo, char reportName[])

{

int ReportNo = rpNo;

strcpy(name, reportName);

}

void Report::displayReport()

{

cout << "Report No : " << ReportNo << endl;

cout << "Repoet Name : " << name << endl;

}



23 | P a g e

**Main.cpp**

//Admin & Report codes

#include <iostream>

#include <cstring>

#include "Admin.h"

#include "Report.h"

using namespace std;

int main() {

Report rpNo;

rpNo.Reports(4197392, (char\*)"CUSTOMR REPORTS"); rpNo.displayReport();

cout << "\n" << endl;

cout << "--------------------------------" << endl;

Admin ADNo;

ADNo.Admins(4, (char\*)"Nimal");

ADNo.displayAdmin();

cout << "\n" << endl;

cout << "--------------------------------" << endl;

}



24 | P a g e

**Sample Outputs**

**Graphical user interface, text, application

Description automatically generated**

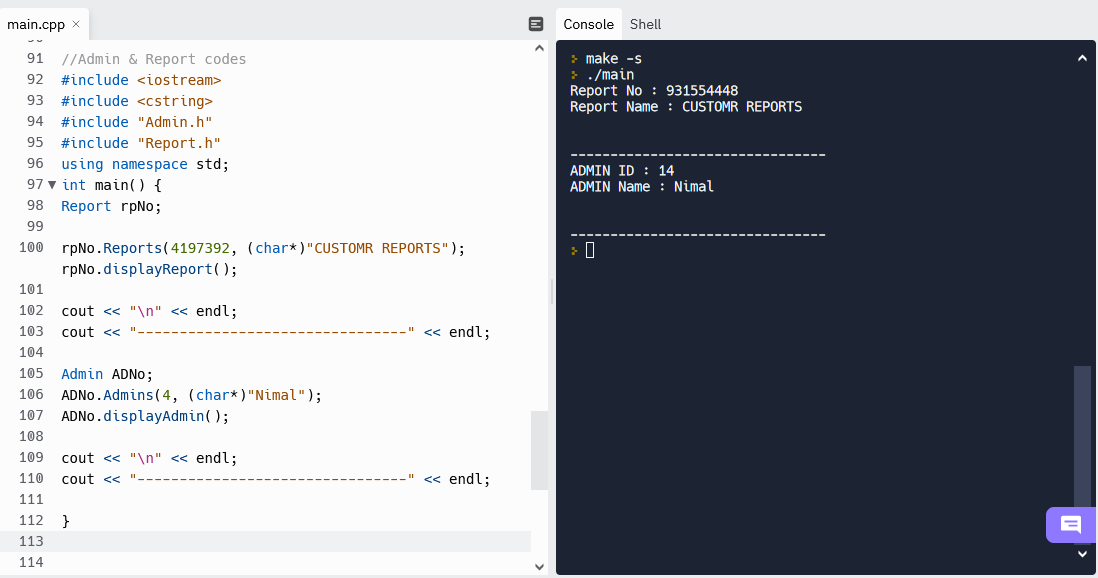
**Graphical user interface, text

Description automatically generated**



25 | P a g e





26 | P a g e

**Individual Contributions**

**IT21175398 – Godage D.N.P**

* Wrote the introduction.
* Created and documented the user requirements.
* Created the CRC cards for unregistered user , location.
* Implemented the coding for admin, customer, feedback, inquiry, park location and report class.

**IT21176210 - Asmitha Thiraviyarasa**

* Created and documented noun verb analysis.
* I have drawn the admin , reports class diagram .
* Created the CRC cards for admin , reports.
* Assisted in implementing the coding for admin , reports.

**IT21176142 - Abiramy Kumaresan**

* Created and documented noun verb analysis.
* I have drawn the registered user ,reservation class diagram .
* Created the CRC cards for registered user ,reservation.
* Assisted in implementing the coding for registered user ,reservation.

**IT21173936 - Kodithuwakku K.G.K.M.J**

* I have drawn the payment ,feedback class diagram .
* Assisted in implementing the coding for payment ,feedback.
* Created the CRC cards for payment ,feedback.

**IT21174612 - G.G.P.N. De Davin**

* Created and documented the user requirements.
* I have drawn the inquiry class diagram.
* Created the CRC cards for inquiry.
* Assisted in implementing the coding for inquiry.



27 | P a g e