Topic : **Identity Issuing Service System** (Driving license issuing service)

Group no : MLB_05.02_02

Campus : Malabe / ~~Metro~~ ~~/ Matara~~ / ~~Kandy~~ / ~~Kurunegala~~ / ~~Kandy~~ / ~~Jaffna~~

Submission Date : 20/05/2022

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**
IT21295706	K.R.K Vidyaratna	0772256676
IT21292736	K. M. N Ayeshani	076 9711706
IT21293894	Siriwardana K.D.S. P	0710888133
IT21294402	Perera.G.A.T.D	0778994878
IT21295638	Vishvanath M.S.M. V	0713464893

Requirements

1) Guest users can register to the system by filling the registration form the guest name, email, DOB and contact number with a strong password.

2) A registered user can login to the system by entering the Username, User ID and the password.

3) If a registered user has forgotten the password, it can be resented by using the user's email address.

4) Both registered / guest users have only front-end access.

5) Guest users have access to any page, except for form submissions and making payments

6) Every form submission requires a specific payment.

7) Payments can be made using both card and cash methods.

8) Forms can be added to the cart for both reference and submission.

9) System will display the form ID and quantity and charge per form once forms are added to the cart.

10) Necessary payments should be made prior to the submission.

11) Both registered and guest users can view/download any form.

12) Registered user should choose a payment type and enter correct payment details and must confirm the payment once it is done system will generate a payment ID.

13) Once the payment type is selected as either cash or card, user must provide the card type, card number and expiry date if card payment method is selected else payment method is selected as cash, system will generate the payment amount and balance automatically.

14) Both registered users and guest users can contact customer support.

15) Registered users can manage/update his/her personal profiles.

16) Registered users can post questions and answers in the FAQ's section.

17) Guest users can add/remove/edit the user account.

18) Both registered and guest users can send feedbacks through the system to clarify doubts.

19) System will generate a feedback ID once the user email and the feedback message is submitted.

Identified Classes

- Guest user
- Registered user
- Feedbacks
- Form
- cart
- Payments
- Card
- Cash
- Submission

Attrributes

- User ID
- Guest name
- User email
- User DOB
- Username
- Password
- Contact number
- Form ID
- Quantity
- Form name
- Charges of forms

- Feedback no
- User email
- Feedback msg
- Payment id
- Amount
- Payment type
- Card type
- Card number
- Expire date
- Pay amount
- Balance
- Submission id
- User id
- Guest name
- Submission date

CRC cards

Guest User	
Responsibilities:	**Collaborations:**
Register to the system	
View forms	Forms
Download forms	Forms
Contact customer support	

Registered User	
Responsibilities:	**Collaborations:**
Log in to the system	
Search for online services	
Download forms	Forms
View forms	Forms
Do a payment	Payment
Manage personal profiles	
Post questions	
Contact customer support	

Forms	
Responsibilities:	**Collaborations:**
View forms	Registered user, guest user
Download forms	Registered user, guest user
Add forms	Administrator

Payment	
Responsibilities:	**Collaborations:**
Choose payment method	Registered user
Enter payment details	Registered user
Confirm payment	Registered user

Feedback	
Responsibilities:	**Collaborations:**
Add feedback	Registered user
Check feedback	Administrator

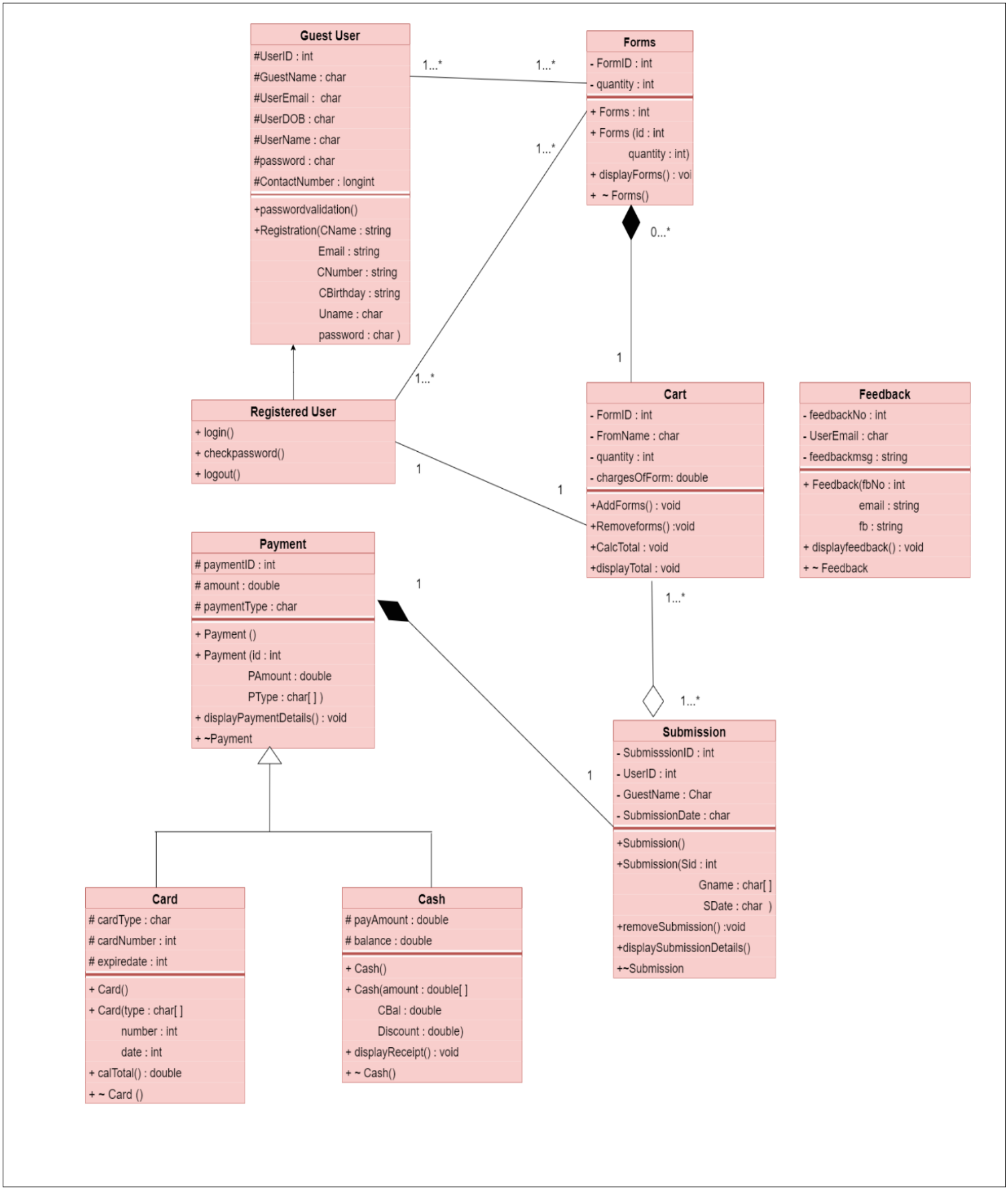
Card	
Responsibilities:	**Collaborations:**
Keep card details	Payment
Calculate the discount	Payment

Cash	
Responsibilities:	**Collaborations:**
Calculate balance	Payment
Display receipt	

Submission	
Responsibilities:	**Collaborations:**
Add submission	
Remove submission	
Confirm submission	Payment

Cart	
Responsibilities:	**Collaborations:**
Add forms	
Remove forms	
Update cart	

Class Diagram



C++ Coding

Header Files

GuestUser.h

```
#pragma once
#include "Form.h"

#define SIZE 2

class GuestUser
{
private:
	// class relationship
	Form * form[SIZE] ;

protected:
	int UserID ;
	char GuestName[30] ;
	char GuestEmail[50] ;
	char GuestDOB[10] ;
	char UserName[20] ;
	char Password[15] ;
	int ContactNumber ;

public:
```

```
        GuestUser();
        GuestUser(int uid,char const gname[], char const email[], char const dob[],
        char const uname[], char const pwd[], int number);
        void Registration();
        int passwordValidation();
        ~GuestUser();

};
```

RegisterUser.h

```
#pragma once

#include "GuestUser.h"
#include "Form.h"
#include "Cart.h"

#define SIZE 2

class RegisteredUser:

public GuestUser
{

private:

        //class relationship
```

```
        Form * form[SIZE];
Cart * cart ;

public:

        RegisteredUser();
        RegisteredUser(char const runame[25], char const rpwd[8]);
        void login();
         void checkPassword();
        void logout();
        ~RegisteredUser();

};
```

Forms.h

```
#pragma once
#include "Cart.h"
#include "GuestUser.h"
#include "Registereduser.h"

class Registered; class

GuestUser;

Class Forms
```

```
{
private:
        int FormID;
        quantity;
         double unitprice;

        //class relationship
           GuestUser *un[2];
           RegisteredUser *r[2];
            Cart *cart[2];

public:
         Forms();
           Forms(int id, int q, int fid, char const fName[], char const pDes[], int qa,double up);
         void display();
           ~Forms();
}
```

Payment.h

```
#pragma once
#include "Submission.h"
#define SIZE 2
class Payment
```

```
{

protected:

        int paymentID;

        double amount;

        char paymentType[20];


private:
        Submission *Sub[SIZE];

public:
        Payment();

        Payment(int id, double PAmount, const char PType[]);

        void displayPaymentDetails();

        ~Payment();
};
```

Feedback.h

```
class Feedback
{
private:
        int feedbackNo;

        string email;

        string feedbackmsg;

public:
        Feedback(int fbNo, string mail, string fb);

        void displayFeedback();

        ~Feedback();
```

```
};
```

Card.h

```
# pragma once

# include "payment.h"

# include <cstring>

class Card : public Payment

{

protected:

        char cardType[10];

        int cardNumber;

         int expiredate ;


public:

        card();

        card(const char cType[],int cNo,int Date);

        double calcTotal();

        ~Card();

}
```

Cash.h

```
#pragma once

#include "Payment.h"

class Cash :public Payment
```

```
{
protected:
        double payAmount;
        double balance;
public:
        Cash();
        Cash(double amount, double CBal);
        void displayReceipt();
        ~Cash();
};
```

Submission.h

```
# pragma once
# include "Cart.h"
#include "Payment.h"
# define SIZE 2

class Submission
{
private:
        int SubmissionId ;
        int UserId ;
        char GuestName[30] ;
        char SubmissionDate[20] ;
```

```
        // class relationship


        Cart*crt[SIZE] ;


public:
        Submission();
        Submission(int Sid,const char Gname[],const char SDate[]) ;
         void removeSubmission ;
         void displaySubmissionDetails() ;
         ~Submission() ;
}
```

Cart.h

```
#pragma once
Class Cart
{
private:
        int FormID;
        char FormName;
        int quantity;
        double ChargesOfForm;


public:
```

```
        double total=0;
        Cart ( int FID, char const FName, int q, double c);
        void addForms ();
        void removeForms ();
        double calcTotal ();
        void displayTotal ();
        Cart ();

};
```

CPP Files

GuestUser.CPP

```
#include "GuestUser.h"
#include <cstring>
#include <iostream>

using namespace std;

GuestUser::GuestUser()
{

        UserID = 0;
        strcpy(GuestName, "");
        strcpy(GuestEmail, "");
        strcpy(GuestDOB, "");
```

```
        strcpy(UserName, "");
        strcpy(Password, "");
        contactNumber = 0;
}


GuestUser::GuestUser(int uid,char const gname[], char const email[], char const dob[],
 char const uname[], char const pwd[], int number)
{

        UserID = uid;
        strcpy(GuestName, gname);
        strcpy(GuestEmail, email);
        strcpy(GuestDOB, dob);
        strcpy(UserName, uname);
        strcpy(Password, pwd);
        contactNumber = number;
}
void GuestUser::Registration()
{

}

int GuestUser::passwordValidation()
```

```
{
	char rav[30];
	 strcpy(rav,"guest");

	if((strcmp (userName,rav)== 0) && (strcmp (Password,"1234")== 0))
	 {

		return 1;
	 }

	else
	 {

		return 0;
	}

}

GuestUser::~GuestUser()
{
	cout << "Destructor runs" << endl;
}
```

RegisterUser.CPP

```
#include "RegisteredUser.h"

#include <cstring>

#include <iostream>


using namespace std;


RegisteredUser::RegisteredUser()


{

	strcpy(GuestName, "");

	strcpy(Password, "");

}


RegisteredUser::RegisteredUser(char const gname[], char const pwd[])

{


	strcpy(userName, gname);

	strcpy(Password, pwd);

}


void RegisteredUser::login()

{

}
```

```
void RegisteredUser::checkPassword()
{

        char rav[30];
        strcpy(rav,"guest");

        if((strcmp (userName,rav)==0) && (strcmp(Password,"1234") ==0))
        {

                cout << "******************" << endl << endl << "------ Welcome ------" << endl
        << endl;
        }

        else
        {
                cout << endl << endl << "Invalid Username or Password" << endl << endl;
        }

}

void RegisteredUser::logout()
{
}
```

```
RegisteredUser::~RegisteredUser()
{
        cout << "Destructor runs" << endl;
}
```

Forms.CPP

```
#include "Forms.h"
#include <cstring>
#include <iostream>
#include <iomanip> using
namespace std;
Forms::Forms()

{
}
Forms::Forms(int id, int q, int fid, char const fName[], char const pDes[], int qa,double up)
{
         FormsID = id;
        quantity = q;
         unitprice = up;
        //creating objects from Cart class
        cart[0] = new Cart(fid, fName,
pDes,qa,up);
}
```

```
void Forms::Forms ()
{

        cout << "ID : " << formsID << endl;

       cout << "Quantity : " << quantity
<<endl;

}


Forms::Forms()
{
        cout << "Form is deleted" << endl;
        for (int i = 0; i < 2; i++)
        {
                delete cart[i];
        }
}
```

Payment.CPP

```
#include "Payment.h"
#include <iostream>
#include <cstring>
using namespace std;
```

```
Payment::Payment()

{
	paymentID = 0;

	amount = 0;

	strcpy(paymentType, "");

	sub = new Submission[SIZE];

	cout << "Payment default constructor called" << endl;

}

Payment::Payment(int id, double PAmount, const char PType[])

{
	paymentID = id;

	amount = PAmount;

	strcpy(paymentType, PType);

}

void Payment::displayPaymentDetails()

{

}

Payment::~Payment()

{
	Delete sub[SIZE];

	cout << "Payment destructor called" << endl;

}
```

Feedback.CPP

```
Feedback::Feedback(int fbNo, string mail, string fb)
{
        feedbackNo = fbNo;    email = mail;
                        feedbackmsg = fb;}

void Feedback::displayFeedback()
{
}

Feedback::~Feedback()
{
}
```

Card.CPP

```
#include "Card.h"

#include <iostream>

#include <cstring>

using namespace std;


Card::Card()

{

        Strcpy(cardType , " ");

        cardNumber = 0;

        expiredate = 0;

};


Card::Card (const char cType[],int cNo,int Date[] , int id , double Pamount , const char
PType[]):Payment(id , PAmount , PType[])
```

```
{
	strcpy(cardType, cType);
	cardNumber = cNo;
	expiredate = Date;
}

double Card::calcTotal()
{

Card::~Card()
{
}
```

Cash.CPP

```
#include "Cash.h"
#include <iostream>

Cash::Cash()
{
	payAmount = 0;
	balance = 0;
}

Cash::Cash(double amount, double CBal, int id, double PAmount, const chrPType[]):Payment(id
, PAmount , PType[])
{
	payAmount = amount;
```

```
        balance = CBal;
}
void Cash::displayReceipt()
{
}
Cash::~Cash()
{
}
```

Submission.CPP

```
# include "Submission.h"
# include <cstring>
# include <iostream>
using namespace std ;

Submission::Submission()
{
        SubmissionId = 0 ;
        UserId = 0 ;
        strcpy(GuestName,"") ;
        strcpy(SubmissionDate,"") ;
}

Submission::Submission(int SId,const char Gname[],const char SDate[]);
```

```
{
        SubmissionId = SId ;
        strcpy(GuestName,Gname);
        strcpy(SubmissionDate,SDate) ;
}

void Submission :: removeSubmission ()
{

}

void Submission :: displaySubmissionDetails ()
{

}

Submission::~Submission()
{

}
```

Cart.CPP

```
#include <iostream>

#include <cstring>

using namespace std;


Cart:: Cart (int FID, char const FName[], int q, double c)

{

	FormID = FID;

	strcpy ( FormName, FName);

	quantity =q;

	ChargesOfForm= c;

}


void Cart:: addForms ()

{

	cout << "Form is added" << endl;

}


void Cart:: removeForms ()

{

	cout << " Form is removed" << endl;

}


double Cart:: calcTotal ()
```

```
{
        total+=ChargesOfForm;
        return total;
}

void Cart :: displayTotal ()
{
        cout<<" Total" << calcTotal () << endl;
}

void Cart::Cart()
{
 cout << "deleted" << endl;
}
```

Main Program

```
#include <iostream>
#include “GuestUser.h”
#include “RegisteredUser.h”
#include “Payment.h”
#include “Cart.h”
#include “Card.h”
#include “Cash.h”
#include “Form.h”
```

```
#include “Feedback.h”

#include “Submission.h”

int main ()

{

//Guest User Class Object

GuestUser* guestuser;


//Registered User Class Object

RegisteredUser* registereduser;


//Payment Class Object

Payment* payment;


//Cart Class Object

Cart* cart;


//Card Class Object

Card* card;


//Cash Class Object

Cash* cash;


//Form Class Object

Form* form;
```

```
//Feedback Class Object
Feedback* feedback;

//Submission Class Object
Submission* submission;

//----------Method Calling------------------
guestuser->registration();
guestuser->cancelRegistration();

registereduser->login();
registereduser->logout();

payment->checkDetails();
payment->confirmDetails();

cart->addItems();
cart->removeItems();

card->addDetails();
card->removeDetails();

cash->displayCash();
```

```
cash->remove();


form->addForms();

form->removeForms();


feedback->addFeedback();

feedback->removeFeedback();


submission->addSubmission();

submission->removeSubmission()

}
```

C++ Coding part individual contribution	
K.R.K Vidyaratna (IT21295706)	header and cpp files of Guest user and registered user classes
Siriwardana K.D.S. P (IT21293894)	header and cpp files of Payment parent class and cash sub class
Vishvanath M.S.M. V (IT21295638)	header and cpp files of submission class and card sub class
K. M. N Ayeshani (IT21292736)	-header and cpp file of cart class -main program
Perera.G.A.T.D (IT21294402)	header and cpp files of Form and feedback classes
UML class diagram individual contribution	
K.R.K Vidyaratna (IT21295706)	- Found out the relationships between classes -Found out the methods of each class and included them in the diagram with the correct UML notations of both attributes and methods - Found the multiplicity between class relationships -Wrote The requirements of the system.
Siriwardana K.D.S. P (IT21293894)	- Found all the classes of the system by doing the noun verb analysis of the requirements - Helped finding private (-) and protected (#) attributes of the diagram
Vishvanath M.S.M. V (IT21295638)	Found the attributes of the classes from the noun verb analysis. - Helped finding relationships between classes

K. M. N Ayeshani (IT21292736)	Helped rechecking the classes found from the noun verb analysis by creating CRC cards for all the classes in reference to the requirements of the system
Perera.G.A.T.D (IT21294402)	-Created the UML class diagram on an online platform - Helped solving issues arise while writing the UML notations