

Topic : Online Recipe Management System

Group no : MLB 10.2\_06

Campus : Malabe

Submission Date : 2022/05/17

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** |
| IT21388934 | Siriwardhana S.M.R.R. | 0703961965 |
| IT21388002 | Hasamal M.A.P. | 0782498019 |
| IT21388552 | Karunasena D.M.P.S.M | 77 886 9302 |
| IT21386268 | Jayarathna P.G.L.N | 77 035 1657 |
| IT21387012 | P.Yasith Hansaka | 76 666 0760 |

1. **System Requirements**

* The system should function 24/7/365.
* Visitor should enter the email and name before entering the system.
* Visitors can explore and study the system and they can search for recipes by recipe name, ingredient name, and mealtime.
* For further use of the system, guests must provide the relevant information and register to the system.
* There are three main characters in the system registered customers, publishers, and administrators, who can enter the system by entering their username and password.
* Visitors can register to the system as a publisher or as normal users.
* Normal user can become a publisher
* Registered users can view any recipe but if it is a pro recipe visitors should do the relevant payment before viewing the recipe page.
* The customer must enter their payment information systems such as payment type, and card details.
* After the payment is made, the customer and the system receive the relevant payment notification.
* If the registered user wants a legal recipe document with the license. They can download the full document file through the recipe page.
* Registered users can check whether there is any allergen from the ingredient list that they have already prepared or listed.
* Publishers can publish the recipe and also they can delete and edit recipes, that they published.
* Publishers can make income from the pro recipe that they listed.
* Publishers can withdraw their income if the total income is more than 30USD.
* Publisher will get the notification after withdrawal is completed.
* Administrators control the system and create revenue reports, feedback reports, and traffic reports.

1. **Noun & Verbs Analysis**
2. **Nouns**

* The system should function 24/7/365.
* Visitor should enter the email and name before entering the system.
* Visitors can explore and study the system and also, and they can search for recipes by recipe name, ingredient name, and mealtime.
* For further use of the system, guests must provide the relevant information and register to the system.
* There are three main characters in the system normal users, publishers, and administrators, who can enter the system by entering their username and password.
* Visitors can register to the system as a publisher or as normal users.
* Normal user can become a publisher
* Registered users can view any recipe but if it is a pro recipe visitors should do the relevant payment before viewing the recipe page.
* The customer must enter their payment information systems such as payment method, and card details.
* After the payment is made, the customer and the system receive the relevant payment notification.
* If the registered user wants a legal recipe document with the license. They can download the full document file through the recipe page.
* Registered users can check whether there is any allergen from the ingredient list that they have already prepared or listed.
* Publishers can publish the recipe and also they can delete and edit recipes, that they published.
* Publishers can make income from the pro recipe that they listed.
* Publishers can withdraw their income if the total income is more than 30USD.
* Publisher will get the notification after withdrawal is completed.
* Administrators control the system and create reports.

**Identify classes**

visitor

normal user

publisher

administrator

recipe

Legal recipe document

payment

withdrawal

allergen

income

report

**Reasons for rejecting other nouns**

1. **Redundant**

* Guests
* Character
* Register user
* Recipe page
* Document file
* Total income

1. **An event or an operation**

* Explore and study the system

1. **Outside scope of the system**

* System
* License
* Notification

1. **Meta-language**

* They

1. **An attributes**

* Recipe name
* Mealtime
* Ingredients name
* ingredient
* Username
* Password
* Payment method
* Card details
* Pro recipe
* Name
* email

1. **Verbs**

* The system should function 24/7/365
* Visitor should enter the email and name before entering the system..
* Visitors can explore and study the system and also, and they can search for recipes by recipe name, ingredient name, and mealtime.
* For further use of the system, guests must provide the relevant information and register to the system.
* There are three main characters in the system registered customers, publishers, and administrators, who can enter the system by entering their username and password.
* Visitors can register to the system as a publisher or as normal users.
* Normal user can become a publisher
* Registered users can view any recipe but if it is a pro recipe visitors should do the relevant payment before viewing the recipe page.
* The customer must enter their payment information systems such as payment type, and card details.
* After the payment is made, the customer and the system receive the relevant payment notification.
* If the registered user wants a legal recipe document with the license. They can download the full document file through the recipe page.
* Registered users can check whether there is any allergen from the ingredient list that they have already prepared or listed.
* Publishers can publish the recipe and also they can delete and edit recipes, that they published.
* Publishers can make income from the pro recipe that they listed.
* Publishers can withdraw their income if the total income is more than 30USD.
* Publisher will get the notification after withdrawal is completed.
* Administrators control the system and create revenue reports, feedback reports, and traffic reports.

1. **CRC Card**

|  |  |
| --- | --- |
| **Visitor** | |
| **Responsibilities** | **Collaborations** |
| Search | Recipe |
| Register |  |

|  |  |
| --- | --- |
| **Normal user** | |
| **Responsibilities** | **Collaborations** |
| Log in |  |
| Become a publisher |  |
| Enter ingredient for allergen check | Allergen |
| Add payment | Payment |

|  |  |
| --- | --- |
| **Publisher** | |
| **Responsibilities** | **Collaborations** |
| Log in |  |
| Make withdrawal | Withdraw |

|  |  |
| --- | --- |
| **Administrator** | |
| **Responsibilities** | **Collaborations** |
| Log in |  |
| Control the system |  |
| Display Admin details |  |

|  |  |
| --- | --- |
| **Payment** | |
| **Responsibilities** | **Collaborations** |
| Store payment details |  |
| Validate |  |

|  |  |
| --- | --- |
| **recipe** | |
| **Responsibilities** | **Collaborations** |
| Store recipe details |  |
| Store recipe document | Recipe document |
| Generate recipe ID |  |
| Edit recipe |  |
| Delete recipe |  |
| Publish Recipe |  |

|  |  |
| --- | --- |
| **Legal recipe document** | |
| **Responsibilities** | **Collaborations** |
| Generate document ID |  |
| Download recipe | Recipe |

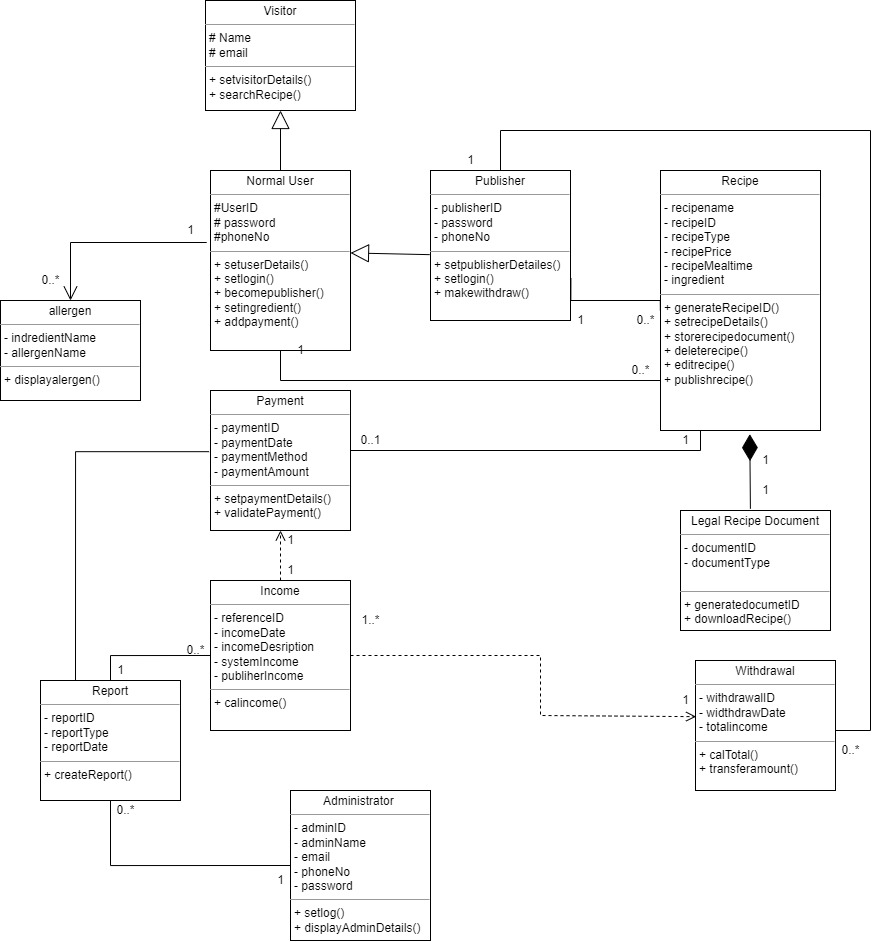
|  |  |
| --- | --- |
| **Withdrawal** | |
| **Responsibilities** | **Collaborations** |
| Calculate total income | Income |
| Transfer amount |  |

|  |  |
| --- | --- |
| **Allergen** | |
| **Responsibilities** | **Collaborations** |
| Identify allergenicity |  |

|  |  |
| --- | --- |
| **Income** | |
| **Responsibilities** | **Collaborations** |
| Calculate income | Payment |

|  |  |
| --- | --- |
| **Report** | |
| **Responsibilities** | **Collaborations** |
| Create report | Payment, Income |

**Class Diagram**

****

**Hader files**

Visitor.h

class visitor{

protected :

char name[20];

char email[30];

public :

visitor();

visitor ( const char cName[20], const char cemail[30]);

void setvisitordetailes();

void searchrecipe();

~ visitor();

}

NormalUser.h

#include "visitor.h"

#include "Recipe.h"

#include "allergen.h"

#define SIZE 100

class NormalUser : public visitor{

protected :

char userID[10];

char userPassword[12];

int phoneNo;

Recipe\*recipe[SIZE];

allergen\*allergen[12];

public:

NormalUser();

NormalUser( const char cuserID[10], const char cpassword[12], int cphoneNo, const char cName[], const char cemail[]);

void normaluserdetails();

void login();

void becomePubliser();

~ NormalUser();

}

Publisher.h

#include "NormalUser.h"

class Publisher: public NormalUser{

protected:

char pID[10];

char pPassword[12];

public :

Publisher();

Publisher(const char publisherID[],const char publisherpass[], const char cuserID[], const char cpassword[], int cphoneNo, const char cName[], const char cemail[]);

void setpublisherdetailes();

void login();

void makewidthdraw();

~ Publisher();

};

Administrator.h

#include "Report.h"

class Administrator{

private:

char adminID[10];

char adminName[20];

char adminemail[30];

int adminphoneNo;

char adminpassword[12];

Report\*report[SIZE];

public:

Administrator();

Administrator(int aID, char aName[], char aEmail[], int aPhoneNo, char aPassword[] );

void setadmindetails();

void setlogin();

void displayAdminDetails();

~ Administrator();

}

Recipe.h

#include "NormalUser.h"

#include "Payment.h"

#include "LegalRecipeDocument.h"

#include <cstring>

class Recipe{

private:

char recipeName[50];

int recipeID;

char recipeType[5];

double recipePrice;

char mealTime[10];

Payment\*payment;

LegalRecipeDocument\*legalrecipedocument;

public:

Recipe();

Recipe(const char rName[],const char rType[],double rPrice,const char mTime);

void setrecipe();

void generateRecipeID();

void publishrecipe();

void editrecipe();

void deleterecipe();

void storeRecipeDocumnet(){

legalrecipedocument = new LegalRecipeDocument();

};

~Recipe();

};

LegalRecipeDocument.h

#include "Recipe.h"

class LegalRecipeDocument{

private:

char documentType[10];

int documentID;

public:

LegalRecipeDocument();

LegalRecipeDocument(const char dType[10]);

void setDoumentDetails();

void genarateDocumentID();

void downloadrecipe();

~LegalRecipeDocument();

}

Allergen.h

#include "NormalUser.h"

class allergen{

private:

char ingredientName[20];

char allergenName[20];

public :

allergen();

allergen(const char iName[], const char aName[]);

void setIngredient(const char iName[]);

void setAllergen(const char aName[]);

void displayAllergenicity();

~allergen();

};

Income.h

#include "Payment.h"

#include "Report.h"

class Income{

private:

int referenceID;

int incomeDate;

int systemIncome;

int publisherIncome;

char incomeDiscription[50];

Report\*report;

public:

Income();

Income(int rId, int iDate,const char iDiscrip);

void calIncome(Payment\*p);

void displayIncome();

double getpublishertotal();

~Income();

}

Payment.h

#include "Recipe.h"

#include "Report.h"

#include "Income.h"

#include "Withdrawal.h"

class Payment

{

private:

int paymentID;

int paymentDate;

double paymentAmount;

char paymentMethod;

Report\*report;

withdrawal\*withdarwal[50];

public:

Payment();

Payment (int pId, int pDate, char pMethod[],double pAmount);

double getpaymentAmount();

void validatePayment();

~Payment();

}

Report.h

#include "Payment.h"

#include "Administrator.h"

#include "Income.h"

class Report{

private:

int reportID;

char reportType;

int reportDate;

Payment\*payment[SIZE];

Income\*income[SIZE];

Administrator\*admin;

public:

Report();

Report(int reID, const char reType[], int reDate);

void setreport();

void createReport();

~Report();

}

Withdrawal.h

#include "Income.h"

#include "Publisher.h"

class withdrawal{

private :

int withdrawalID;

int withdrawalDate;

int totalIncome;

Publisher\*withdraw;

public :

withdrawal();

withdrawal(int wID,int wDate);

void setwithdrawdetails();

void calTotal(Income\*i);

void transferAmount();

~withdrawal();

};

**CPP FILES.**

Visitor.cpp

#include <iostream>

#include "visitor.h"

#include <cstring>

visitor:: visitor(){

strcpy(name[20],"");

strcpy (email[30],"");

}

visitor::visitor ( const char cName[], const char cemail[]){

strcpy(name,cName);

strcpy(email,cemail);

}

void visitor:: searchrecipe(){

}

visitor::~visitor(){

//destructor

}

NormalUser.cpp

#include <iostream>

#include "NormalUser.h"

#include <cstring>

NormalUser::NormalUser(){

strcpy (userID[10],"");

strcpy (userPassword[12],"");

phoneNo=0;

}

NormalUser::( const char cuserID[], const char cpassword[],int cphoneNo,const char cName[], const char cemail[]):visitor ( const char cName[], const char cemail[])

{

strcpy (userID,cuserID);

strcpy (userPassword,cpassword);

phoneNo=cphoneNo;

}

void login(){

};

void becomePubliser(){

};

NormalUser::~NormalUser(){

//destructor

}

Publisher.cpp

#include <iostream>

#include "Publisher.h"

#include <cstring>

Publisher::Publisher(){

strcpy(pID[10],"");

strcpy(pPassword[12],"");

}

Publisher::Publisher(const char publisherID[],const char publisherpass[],const char cuserID[], const char cpassword[], int cphoneNo, const char cName[], const char cemail[]):NormalUser(cuserID[],cpassword[],cphoneNo,cName[],cemail[]){

strcpy(pID,publisherID);

strcpy(pPassword,publisherpass);

}

void login(){

};

void makewidthdraw(){

};

Publisher::~Publisher(){

//destructor

}

Recipe.cpp

#include "Recipe.h"

#include <iostream>

#include <cstring>

Recipe::Recipe(){

strcpy(recipeName,"");

strcpy(recipeID,"");

strcpy(recipeType,"");

recipePrice=0;

strcpy(mealTime,"");

}

Recipe::Recipe(const char rName[],const char rType[],double rPrice,const char mTime){

strcpy(recipeName,rName);

strcpy(recipeID,rName);

strcpy(recipeType,rType);

recipePrice=rPrice;

strcpy(mealTime,mTime);

}

void Recipe::generateRecipeID(){

}

void Recipe::publishrecipe(){

};

void Recipe::editrecipe(){

};

void Recipe::deleterecipe(){

};

Recipe::~Recipe(){

//destructor

}

Adminitrator.cpp

#include "Administrator.h"

#include <iostream>

#include <cstring>

Administrator::Administrator(){

strcpy(adminID,"");

strcpy(adminName,"");

strcpy(adminemail,"");

adminphoneNo=0;

strcpy(adminpassword,"");

};

Administrator::Administrator(int aID, char aName[], char aEmail[], int aPhoneNo, char aPassword[] ){

strcpy(adminID,aID);

strcpy(adminName,aName);

strcpy(adminemail,aEmail);

adminphoneNo=aPhoneNo;

strcpy(adminpassword,aPassword);

}

void Administrator::setlogin(){

}

void Administrator::displayAdminDetails(){

}

Administrator::~Administrator(){

//destructor

}

Allergen.cpp

#include "allergen.h"

#include <iostream>

#include <cstring>

using namespace std;

allergen::allergen(){

strcpy(ingredientName,"");

strcpy(allergenName,"");

}

allergen::allergen(const char iName[], const char aName[]){

strcpy(ingredientName,iName);

strcpy(allergenName,aName);

}

void allergen::setIngredient(const char iName[]){

strcpy(ingredientName,iName);

}

void allergen::setAllergen(const char aName[]){

strcpy(allergenName,aName);

}

void allergen::displayAllergenicity(){

strcmp(ingredientName,allergenName);

if (ingredientName[]==allergenName[]){

cout<<ingredientName[]<<"is a allergen"<<endl;

}

}

allergen::~allergen(){

//destructor

}

Income.cpp

#include "Income.h"

#include <iostream>

#include <cstring>

using namespace std;

Income::Income(){

referenceID=0;

incomeDate=0;

strcpy(incomeDiscription,"");

}

Income::Income(int rId, int iDate,const char iDiscrip){

referenceID=rId;

incomeDate=iDate;

strcpy(incomeDiscription,iDiscrip);

}

void Income::calIncome(Payment\*p){

systemIncome=(20/100)\*p->getpaymentAmount();

publisherIncome=(80/100)\*p->getpaymentAmount();

};

double Income:: getpublishertotal(){

};

void Income::displayIncome(){

cout << "Income Details"<<endl;

cout << "Reference : "<< referenceID<<endl;

cout << "Income Date : "<< incomeDate<<endl;

cout << "Income Discriotion"<< incomeDiscription<<endl;

cout << "System Income"<< systemIncome<<endl;

cout << " Publisher Income"<< publisherIncome <<endl;

};

Income::~Income(){

//destructor

}

LeagalRecipeDocument.cpp

#include "LegalRecipeDocument.h"

#include <iostream>

#include <cstring>

LegalRecipeDocument::LegalRecipeDocument(){

strcpy(documentType,"None");

documentID=0;

}

LegalRecipeDocument::LegalRecipeDocument(const char dType[10]){

strcpy(documentType,dType);

}

void LegalRecipeDocument::genarateDocumentID(){

};

void LegalRecipeDocument::downloadrecipe(){

};

LegalRecipeDocument::~LegalRecipeDocument(){

//destructor

}

Payment.cpp

#include <iostream>

#include "Payment.h"

#include <cstring>

Payment::Payment(){

paymentID = 0;

paymentDate = 0;

strcpy(paymentMethod,"");

paymentAmount=0;

}

Payment::Payment(int pId, int pDate, char pMethod[],double pAmount){

paymentID = pId;

paymentDate = pDate;

strcpy(paymentMethod, pMethod);

paymentAmount=pAmount;

};

void Payment::validatePayment(){

}

double Payment::getpaymentAmount(){

return paymentAmount;

};

Payment::~Payment(){

//destructor

}

Report.cpp

#include "Report.h"

#include <iostream>

#include <cstring>

Report::Report(){

reportID=0;

strcpy(reportType,"");

reportDate=reDate;

}Report::Report(int reID, const char reType[], int reDate){

reportID=reID;

strcpy(reportType,reType);

reportDate=reDate;

}

void Report::createReport(){

};

Report::~Report(){

//destructor

}

Withdrawal.cpp

#include <iostream>

#include "Withdrawal.h"

#include <cstring>

withdrawal::withdrawal(){

withdrawalID=0;

withdrawalDate=0;

}

withdrawal::withdrawal(int wID,int wDate){

withdrawalID=wID;

withdrawalDate=wDate;

}

void withdrawal:: calTotal(Income\*i){

totalIncome=i->getpublishertotal();

}

void transferAmount(){

};

withdrawal::~withdrawal(){

//destructor

}

**Main cpp**

#include "Administrator.h"

#include "allergen.h"

#include "Income.h"

#include "LegalRecipeDocument.h"

#include "NoramlUser.h"

#include "Payment.h"

#include "Publiher.h"

#include "Recipe.h"

#include "Report.h"

#include "visitor.h"

#include "Withdrawal.h"

#include <iostream>

using namespace std;

int main(){

Administrator\*adm = new Administrator();

allergen\*all = new allergen();

Income\*in= new Income();

LegalRecipeDocument\*lrd= new LegalRecipeDocument();

NormalUser\*nu = new NormalUser();

Payment\*pay = new Payment();

Publisher\*pub = new Publisher();

Recipe\*rcp = new Recipe();

Report\*rpt = new Report();

visitor\*vis = new visitor();

withdrawal\*with =new withdrawal();

adm -> setadmindetails();

adm -> setlogin();

adm -> displayAdminDetails();

all -> setIngredient();

all -> setAllergen();

all -> displayAllergenicity();

in -> displayIncome();

in -> calIncome;

in -> getpublishertotal();

lrd -> setDoumentDetails();

lrd -> genarateDocumentID();

lrd -> downloadrecipe();

nu -> normaluserdetails();

nu -> login();

nu -> becomePubliser();

pay ->validatePayment();

pay ->getpaymentAmount();

pub -> setpublisherdetailes();

pub -> login();

pub -> makewidthdraw();

rcp -> setrecipe();

rcp -> generateRecipeID();

rcp -> publishrecipe ();

rcp -> editrecipe ();

rcp -> deleterecipe ();

rcp -> storeRecipeDocumnet ();

rpt -> setreport();

rpt -> createReport();

vis -> setvisitordetailes();

vis -> searchrecipe();

with -> setwithdrawdetails();

with -> calTotal();

with -> transferAmount();

delete adm;

delete all;

delete in;

delete nu;

delete pay;

delete pub;

delete rcp;

delete rpt;

delete vis;

delete with;

delete lrd;

return 0;

}

**Contribution on Class diagram and Cording.**

|  |  |  |
| --- | --- | --- |
| **IT Number** | **Student Name** | **Contributed Class** |
| IT21388934 | S.M.R.R.Siriwardhana | Visitor , NormalUser, Publisher classes, Main program |
| IT21388002 | M.A.P. Hasamal | Recipe, LegalRecipeDocument class and Main program |
| IT21388522 | Karunasena D.M.P.S.M | Payment , Income Classes |
| IT21386268 | Jayarathna P.G.L.N. | Allergen and withdrawal classes |
| IT21387012 | P.Yasith Hansaka | Report and Administrator classes. |