Topic :Diet Plan and Health Check-up Systems

Group no :MLB_WE_01.01_09

Campus : Malabe

Submission Date : 16/05/2022

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**
IT21152078	Rathnayake R.M.M.D	0719744878
IT21219702	De Silva G.H.G.T.S.	0710137323
IT21248030	Harischandra D.P.N.B	0715235925
IT21072888	Samarakoon S.M.S.C	0763435633
IT21221682	Thomas W.W.R.A	0718986262

Requirements

1. Users have the ability to access the website without registering. But for Advance options, users must register to the system.

2. Guests can search and view diet plans. They can search the diet plans by standard Plans, vegetarian plans, and nonvegetarian plans.

3. The system allows registered users to view their registration details, and update and edit them.

4. The dietician and doctor are employees of the company; it is compulsory to register with the system, if not dieticians and doctors cannot view diet plans.

5. Dieticians can post new diet plans and edit them.

6. After selecting his preferred option, the user is directed to the payment page to proceed with the payment.

7. A registered user has the ability to do the payments using different payment methods such as credit/debit cards, PayPal, or bank deposits

8. A registered user can contact a doctor or a dietician to get more information about his or her own diet plans.

9. After purchasing the diet plan, registered users can view the purchase history and give their feedback on those purchased diet plans.

10. All users (Registered users, Dieticians, Doctors, and Administrators) can view feedback.

11. The system allows administrators to handle feedback

12. Database administrator can manage the database by updating or deleting existing user accounts. The database administrator can create a backup of the database.

13. System administrator can generate user reports.

Noun-Verb Analysis

1. Users have the ability to access the website without registering. But for Advance options, users must register to the system.

2. Guests can search and view diet plans. They can search the diet plans by standard Plans, vegetarian plans, and nonvegetarian plans.

3. The system allows registered users to view their registration details, and update and edit them.

4. The dietician and doctor are employees of the company; it is compulsory to register with the system, if not dieticians and doctors cannot view diet plans.

5. Dieticians can post new diet plans and edit them.

6. After selecting his preferred option, the user is directed to the payment page to proceed with the payment.

7. A registered user has the ability to do the payments using credit card, debit cards, PayPal, or bank deposits.

8. A registered user can contact a doctor or a dietician to get more information about his/her own diet plans.

9. After purchasing the diet plan, registered users can view the purchase history and give their feedback on those purchased diet plans.

10. All users (Registered users, Dieticians, Doctors, and Administrators) can view feedback.

11. The system allows administrators to handle feedback.

12. Database administrator can manage the database by updating or deleting existing user accounts. The database administrator can create a backup of the database.

13. System administrator can generate user reports.

Rejecting Nouns

Users - Class

Website – Outside scope

Advanced options – Outside scope

System – Outside scope

Guests - Redundant

Diet plans - Class

Standard plans – Attributes

Vegetarian plans - Attributes

Non-vegetarian plans - Attributes

Registered users - Class

Registration details - Class

Dietician - Class

Doctor - class

Employee – Class

Company – outside scope

Payment page – Meta-language

Payment - Class

Credit card - Redundant

Debit card – redundant

PayPal - Redundant

Bank deposits - Redundant

Information – Meta language

History – outside scope

Feedback - Class

Administrators – Outside scope

Database administrator – outside scope

Database – outside scope

User accounts - Class

Backup – outside scope

System administrator – outside scope

Reports - Class

.

Classes

1. Users
2. Registered users
3. Employee
4. Dietician
5. Doctor
6. Diet plans
7. User accounts
8. Payment
9. Reports
10. Feedback

CRC Cards

Class name: User	
Responsibilities	**Collaborations**
Register to the system	

Class name: Registered User	
Responsibilities	**Collaborations**
Login to the system	
View registration details	
Update registration details	

Class name: Employee	
Responsibilities	**Collaborations**
Register to the system	

Class name: Dietician	
Responsibilities	**Collaborations**
Post diet plans	Diet plans
Edit diet plan details	Diet plans
View feedbacks	Feedback

Class name: Doctor	
Responsibilities	**Collaborations**
View user reports	Report
Contact registered user	Registered user
Give feedback	Feedback

Class name: Diet Plan	
Responsibilities	**Collaborations**
Store diet plan details	

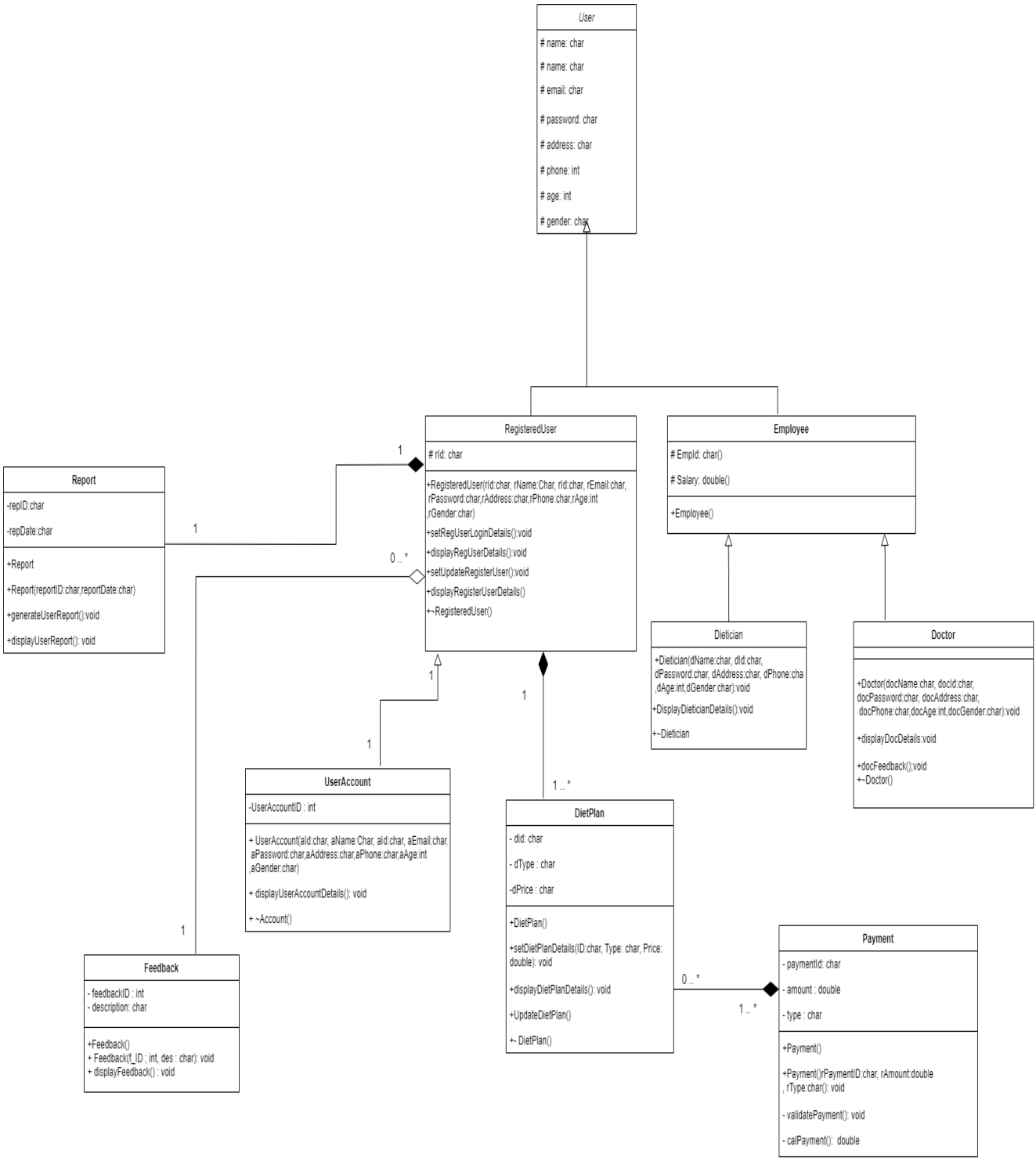
Class name: User account	
Responsibilities	**Collaborations**
Store user details	
Validate user accounts	

Class name: Payment	
Responsibilities	**Collaborations**
Validate payment details	
Store payment details	
Calculate payment	

Class name: Reports	
Responsibilities	**Collaborations**
Generates user reports	User

Class name: Feedback	
Responsibilities	**Collaborations**
Store feedback	

Class Diagram



```
#include<iostream>
#include<cstring>
using namespace std;
//IT21219702-De Silva G.H.G.T.S.
//MLB_WE_01.01_09

//User class
class User{
protected:
char name[20];
char email[20];
char password[8];
char address[30];
long int contactNo;
int age;
char gender[8];

public:
User();
};

//IT21219702-De Silva G.H.G.T.S.
//MLB_WE_01.01_09

//Registered User class
class RegisteredUser : public User {
protected:
char rId[5];
double hight;
double weight;

public:
RegisteredUser();
RegisteredUser(char regID[],double rHight,double rWeight, char rName[], char
rEmail[],
char rPassword[], char rAddress[], long int rContactNo, int rAge, char
rGender[]);
void setRegUserLoginDetails();
void displayRegUserDetails();
void setupdateUserDetails();
~RegisteredUser();
```

```
};

//IT21219702-De Silva G.H.G.T.S.
//MLB_WE_01.01_09

//Employee class

class Employee : public User {
protected:
char empId[5];
double salary;

public:
Employee();
};

//IT21219702-De Silva G.H.G.T.S.
//MLB_WE_01.01_09

//Doctor class

class Doctor : public Employee{

public:
Doctor();
Doctor(char docId[], double docSalary, char docName[], char docEmail[],
char docPassword[], char docAddress[], long int docContactNo, int docAge,
char docGender[]);
void displayDocDetails();
void DoctorFeedback();
~Doctor();
};

//IT21219702-De Silva G.H.G.T.S.
//MLB_WE_01.01_09

//Dietician class

class Dietician : public Employee{

public:
Dietician(char dId[], double dSalary, char dName[], char dEmail[],
```

```
char dPassword[], char dAddress[], long int dContactNo, int dAge, char
dGender[]);
void displayDieticianDetails();
~Dietician();

};

//IT21248030 D.P.N.B.Harischandra
//MLB_WE_01.01_09


//UserAccount class
class UserAccount : public RegisteredUser {

protected :
   char UserAccountId[5];

public:
UserAccount();
UserAccount(char aId[], char aName[], char aEmail[],
char aPassword[], char aAddress[], long int aContactNo, int aAge, char
aGender[]);
void displayUserAccountDetails();
void setupdateUserAccountDetails();

~UserAccount();

};

//IT21248030 D.P.N.B.Harischandra
//MLB_WE_01.01_09

//Report class
class Report{
private:
char repId[5];
char repDate[10];
RegisteredUser *pReport;

public:
Report()
{
strcpy(repId,"");
```

```
strcpy(repDate, "");
}
Report(char reportId[], char reportDate[])
{
strcpy(repId,reportId);
strcpy(repDate,reportDate);
}
void generateUserReport(RegisteredUser *R1 ){
pReport = R1;
}
void displayUserReport()
{
}

};

//IT21072888-Samarakoon S.M.S.C
//MLB_WE_01.01_09

//Feedback class
class Feedback
{
private :
int feedbackID;
char description[50];


public :
Feedback();
Feedback(int f_ID,char des[]);
void displayFeedback();
~Feedback();

};

//IT21072888-Samarakoon S.M.S.C
//MLB_WE_01.01_09

//Payment class
class Payment
{
private:
char paymentID[10];
```

```
double amount;
char type[20];

public:
Payment();
Payment(char rPaymentID[], double rAmount, char rType[]);
void setPaymentDetails(char RPaymentID[], double RAmount, char RType[]);
void validatePayment();
double calPayment();

~Payment();
};

//IT21072888-Samarakoon S.M.S.C
//MLB_WE_01.01_09

class DietPlan //Creating a DietPlan class
{
//Declaring private propeties
private:
   char did[5];
   char dType[20];
   double dPrice;

//Declare public method
public:
   DietPlan(); //Default constructor
   DietPlan(char ID[], char Type[], double Price);
   void setDietPlanDetails(char ID[], char Type[], double Price);
   //Display diet plan details
   void displayDietPlanDetails();
   void updateDietPlanDetails();

   ~DietPlan(); //Destructor
};

//IT21219702-De Silva G.H.G.T.S.
//MLB_WE_01.01_09

//function implementation

User::User(){
strcpy(name, "");
```

```
strcpy(email, "");
strcpy(password,"");
strcpy(address, "");
contactNo = 0;
age = 0;
strcpy(gender, "");
}

RegisteredUser::RegisteredUser(){
strcpy(rId,"");
hight = 0.0;
weight = 0.0;
}

RegisteredUser::RegisteredUser(char regId[],double rHight,double rWeight,
char rName[], char rEmail[],
char rPassword[], char rAddress[], long int rContactNo, int rAge, char
rGender[]){

strcpy(rId, regId);
hight = rHight;
weight = rWeight;
strcpy(name, rName);
strcpy(email,rEmail);
strcpy(password,rPassword);
strcpy(address,rAddress);
contactNo = rContactNo;
age = rAge;
strcpy(gender, rGender);
}

void RegisteredUser::setRegUserLoginDetails(){

cout << "-----------Regitered User login Details-----------" << endl;

cout << "Registered User Id       : ";
cin >> rId;
cout << "Registered User Email    : ";
cin >> email;
cout << "Registered User Password  : ";
}

void RegisteredUser::displayRegUserDetails(){
```

```
cout << "-------Display Registerd User Details-------" << endl ;

    cout << "Name of the registered User is : "<< name << endl;
    cout << "ID of the registered User is : "<< rId << endl;
cout << "Email of the registered User is : "<< email << endl;
  cout << "Password of the registered User is : "<< password << endl;
  cout << "Address of the registered User is : "<< address << endl;
  cout << "Phone of the registered User is : "<< contactNo << endl;
  cout << "Age of the registered User is : "<< age << endl;
  cout << "Gender of the registered User is : "<< gender << endl;

}

void RegisteredUser::setupdateUserDetails(){

cout << "Enter new email               : ";
cin >> email;
cout << "Creat new password            : ";
cin >> password;
cout << "Enter new Address             : ";
cin >> address;
cout << "Enter new contact No          : ";
cin >> contactNo;
}

Employee::Employee(){
strcpy(empId,"");
salary = 0.0;
strcpy(name, "");
strcpy(email, "");
strcpy(password,"");
strcpy(address, "");
contactNo = 0;
age = 0;
strcpy(gender, "");
}

Doctor::Doctor(char docId[], double docSalary,char docName[], char
docEmail[],
char docPassword[], char docAddress[], long int docContactNo, int docAge,
char docGender[]){
```

```
strcpy(empId,docId);
salary = docSalary;
strcpy(name, docName);
strcpy(email,docEmail);
strcpy(password,docPassword);
strcpy(address,docAddress);
contactNo = docContactNo;
age = docAge;
strcpy(gender, docGender);

}

void Doctor::displayDocDetails(){

cout << "-----Doctor Details-------" << endl;
  cout << "Name of the doctor : "<< name << endl;
  cout << "Email of the doctor : "<< email << endl;
  cout << "Address of the doctor : "<< address << endl;
  cout << "Phone of the doctor : "<< contactNo << endl;

}

Dietician::Dietician(char dId[], double dSalary, char dName[], char dEmail[],
char dPassword[], char dAddress[], long int dContactNo, int dAge, char
dGender[]){

strcpy(empId,dId);
salary = dSalary;
strcpy(name, dName);
strcpy(email,dEmail);
strcpy(password,dPassword);
strcpy(address,dAddress);
contactNo = dContactNo;
age = dAge;
strcpy(gender, dGender);

}

void Dietician::displayDieticianDetails(){

  cout << "-----Dietician Details-------" << endl;
  cout << "Name of the dietician : "<< name << endl;
  cout << "Email of the dieticina : "<< email << endl;
```

```
  cout << "Address of the dietician : "<< address << endl;
  cout << "Phone of the dietician : "<< contactNo << endl;

}
//IT21248030 D.P.N.B.Harischandra
//MLB_WE_01.01_09

UserAccount::UserAccount()
{
strcpy(UserAccountId, "");
}
UserAccount :: UserAccount(char aId[], char aName[], char aEmail[],
char aPassword[], char aAddress[], long int aContactNo, int aAge, char
aGender[]){
strcpy(UserAccountId,aId);
strcpy(name,aName);
strcpy(email,aEmail);
strcpy(password,aPassword);
strcpy(address,aAddress);
contactNo = aContactNo;
age = aAge;
strcpy(gender, aGender);
}



void UserAccount::displayUserAccountDetails(){

cout << "-------Display Customer Details-------" << endl ;

    cout << "Name of the customer is : "<< name << endl;
    cout << "ID of the customer is : "<< rId << endl;
cout << "Email of the customer is : "<< email << endl;
  cout << "Password of the customer is : "<< password << endl;
  cout << "Address of the customer is : "<< address << endl;
  cout << "Phone of the customer is : "<< contactNo << endl;
  cout << "Age of the customer is : "<< age << endl;
  cout << "Gender of the customer is : "<< gender << endl;

}

void UserAccount::setupdateUserAccountDetails(){
```

```
cout << "Enter new email               : ";
cin >> email;
cout << "Creat new password            : ";
cin >> password;
cout << "Enter new Address             : ";
cin >> address;
cout << "Enter new contact No          : ";
cin >> contactNo;
}

//IT21072888-Samarakoon S.M.S.C
//MLB_WE_01.01_09

Payment::Payment()
{
strcpy(paymentID, "");
amount = 0;
strcpy(type, "");
}

Payment::Payment(char rPaymentID[], double rAmount, char rType[])
{
strcpy(paymentID, rPaymentID);
amount = rAmount;
strcpy(type, rType);
}

void Payment::setPaymentDetails(char RPaymentID[], double RAmount, char
RType[])
{
strcpy(paymentID, RPaymentID);
amount = RAmount;
strcpy(type, RType);
}

void Payment::validatePayment()
{
int length;
length = strlen(paymentID);
if (length > 1 && length <= 10)
{
cout << "Valid Payment ID" << endl;
}
```

```
else
{
cout << "Sorry! Can't Validate details" << endl;
}
}

double Payment::calPayment()
{

}

DietPlan::DietPlan()
{
   strcpy(did, "");
   strcpy(dType, "");
   dPrice = 0;
}

DietPlan::DietPlan(char ID[], char Type[], double Price)
{
   strcpy(did,ID);
   strcpy(dType, Type);
   dPrice = Price;
}

void DietPlan::setDietPlanDetails(char ID[], char Type[], double Price)
{
   cout << "-----------Diet Plan Details-----------" << endl;
   cout << "Diet Plan Id : ";
   cin >> did;
   cout << "Diet Plan Type : ";
   cin >> dType;
   cout << "Diet Plan Price : ";
   cin >> dPrice;
}

void DietPlan::displayDietPlanDetails()
{
   cout <<  "ID of the Diet Plan    : " << did << endl;
   cout << "Type of the Diet Plan   : " << dType << endl;
   cout << "Price of the Diet Plan  : " << dPrice << endl;
}
```

```
void DietPlan::updateDietPlanDetails()
{
   cout << "Enter New Diet Plan Id    : ";
   cin >> did;
   cout << "Enter New Diet Plan Type  : ";
   cin >> dType;
   cout << "Enter New Diet Plan Price : ";
   cin >> dPrice;
}

Feedback::Feedback(){
feedbackID = 0;
strcpy(description,"");
}
Feedback::Feedback(int f_ID,char des[])
{
feedbackID = f_ID;
strcpy(description, des);

}

void Feedback::displayFeedback()
{
cout << "Feedback ID  : " << feedbackID << endl;
cout << "Description  : " << description << endl;
}

int main(){
//Registered User

RegisteredUser *reg1 = new RegisteredUser("RU001",176.5,48.0,"Dewni",
"Dew@gmail.com",
"dew@1234", "Colombo", 9412435679, 21, "Female");

reg1 -> displayRegUserDetails();

cout << endl << endl;

reg1 -> setupdateUserDetails();

//Doctor
```

```
Doctor *doc1 = new Doctor("DOC01", 200000.00,"Nilmini",
"nilmini@gmailcom",
"nrjnahf", "Dehiwala", 94712782420, 50, "Female");
cout << endl << endl;

doc1 -> displayDocDetails();

//Dietician

Dietician *d1 = new Dietician("DI001", 300000.00,"Arosha",
"Aroshanp@gmailcom",
"NilAhf12", "Negombo", 94712782420, 50, "Male");

cout << endl << endl;
d1 -> displayDieticianDetails();

//payment
Payment *P1 = new Payment("PAY01",20000.00, "Credit card");



//Diet Plan
DietPlan* diet1 = new DietPlan("d0001", "The Vegan Diet plans", 3000.00);
cout << endl << endl;

diet1->displayDietPlanDetails();

//Feedback
Feedback *FD1 = new Feedback(0011,"Great healthy diet plan");
cout << endl << endl;

FD1->displayFeedback();

//Report

Report *prep = new Report("REP01", "2022/04/14");


prep -> displayUserReport();

//UserAccount
```

```
UserAccount *Acc1 = new
UserAccount("ACC01","Malith","malith@gmail.com","jeffkihfi","Colombo",9
41245783,24,"Male");

Acc1->displayUserAccountDetails();

Acc1->setupdateUserAccountDetails();



return 0;

}
```