Discover Your Future

Topic : Diet Planning & Health Check-up. System

Group no : MLB_15.01_07

Campus : Malabe

Submission Date: 07.06.2023

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	Email
IT22575562	WIJEKOON W.M.MG.K.P	0714985025	It22575562@my.sliit.lk
IT22567642	THILAKARATHNE P.L.B.H	0776597112	It22567642@my.sliit.lk
IT22591098	JAYALATH J.P.R.J	0767221436	It22591098@my.sliit.lk
IT22566034	PERERA K.C.D	0765814575	It22566034@my.sliit.lk
IT22217622	HETTIARACHCHI A.N.M	0713611854	It22217622@my.sliit.lk

Exercise 1

System Requirements for Diet Planning & Health Check-up System

1) Any users who has the need can get the relevant services

2) Any user can register as a patient

3) All users can login to the website by providing usernames and passwords

4) Patient Details Doctors details will be displayed on the website

5) Once the user register he/she can change and edit his/her account inquiry.

6) The system will direct the user through text messages or e-mails regarding the appointment details of the patient.

7) According to the test reports, the patient can refer to a relevant health care or food planning department and get relevant solutions.

8) The patient can request a report to receive the relevant treatment after the relevant tests

9) Based on the reports of the first doctor's tests, the health or food planning departments may be referred.

10) The patient can check his records and the doctor's details in the system and the doctor can also access the patient's details through the system.

11) Payments should be made for the services received by the patient

12) The user or patient can make payments online through our website through credit cards and online money transfers.

13) Administrator must also be loaded into the system before accessing the relevant services and activities.

14) Admin will check the system and send notification email messages to the users.

Noun & Verb Analysis

- Nouns in Red color
- Verbs in Blue color

- Any users(patient) who has the need can get the relevant services.
- Any user can register as a patient(users)
- All patients(users) can login to the website by providing usernames &passwords
- Patient Details Doctors details will be displayed on the website
- Once the user registers he/she can change and edit his/her account inquiry.
- The system will direct the user through text messages or e-mails regarding the appointment details of the patient.
- According to the test reports, the patient can refer to a relevant health checkup or diet planning department and get relevant solutions.
- The patient(users) can request a report to receive the relevant treatment after the relevant tests
- Based on the reports of the first doctor's tests, the health or food planning departments may be referred.
- The patient can check his records and the doctor's details in the system and the doctor can also access the patient's details through the system.
- Payments should be made for the services received by the patient
- The user or patient can make payments online through our website through credit cards and online money transfers
- Administrator must also be loaded into the system before accessing the relevant services and activities.
- Admin will check the system and send notification email messages to the users.

Identified Classes using Noun Verb Analysis

Identified Classes:

- Patient
- Health Checkup
- Diet Plan
- Admin
- Manage
- Appointment
- Report
- Payment

Nouns

- User
- Username
- Hospital
- Patient
- Website
- E-mail
- Password
- Credit Card
- Online Payment
- Health Check up
- Diet Plan
- Doctor
- Report
- Appointment
- Date & Time

Reasons for Rejecting Other Nouns

1. Redundant:

- System user, member of diet plan & Health checkup system refers to the same person.

2. Outside scope of system:

- Website, Health checkup system & Diet plan system user requirement request & inquiries are out side the scope of the diet plan & health checkup system.

3. Meta - Language:

- In this all the type of medical tests can only as “Check-up”
- In this all the type of medical reports can only as “Report”.

4. An attribute:

- Appointment
- User name
- Password
- Payment Type

CRC Cards for Diet Planning & Health Check-up System

User (Patient)	
Responsibilities:	**Collaborations:**
Login to the system	
Search Doctors	Admin
View profile	
Delete profile	
Edit profile details	
Request for a Services	Admin
providing usernames &passwords	Admin
Make payment	Payment
Request to add patient (Users)	Admin
Request to remove patient (Users)	Apartment

Admin	
Responsibilities:	**Collaborations:**
Login to the system	
Accept or request	Patient
Add or remove users(patient) in system	Admin
Update appointment schedule	Appointment
Generate report	Report

Report	
Responsibilities:	**Collaborations:**
Display health checkup details	
Display diet plan details	
Update appointment details	Admin
Cash flow	Payment

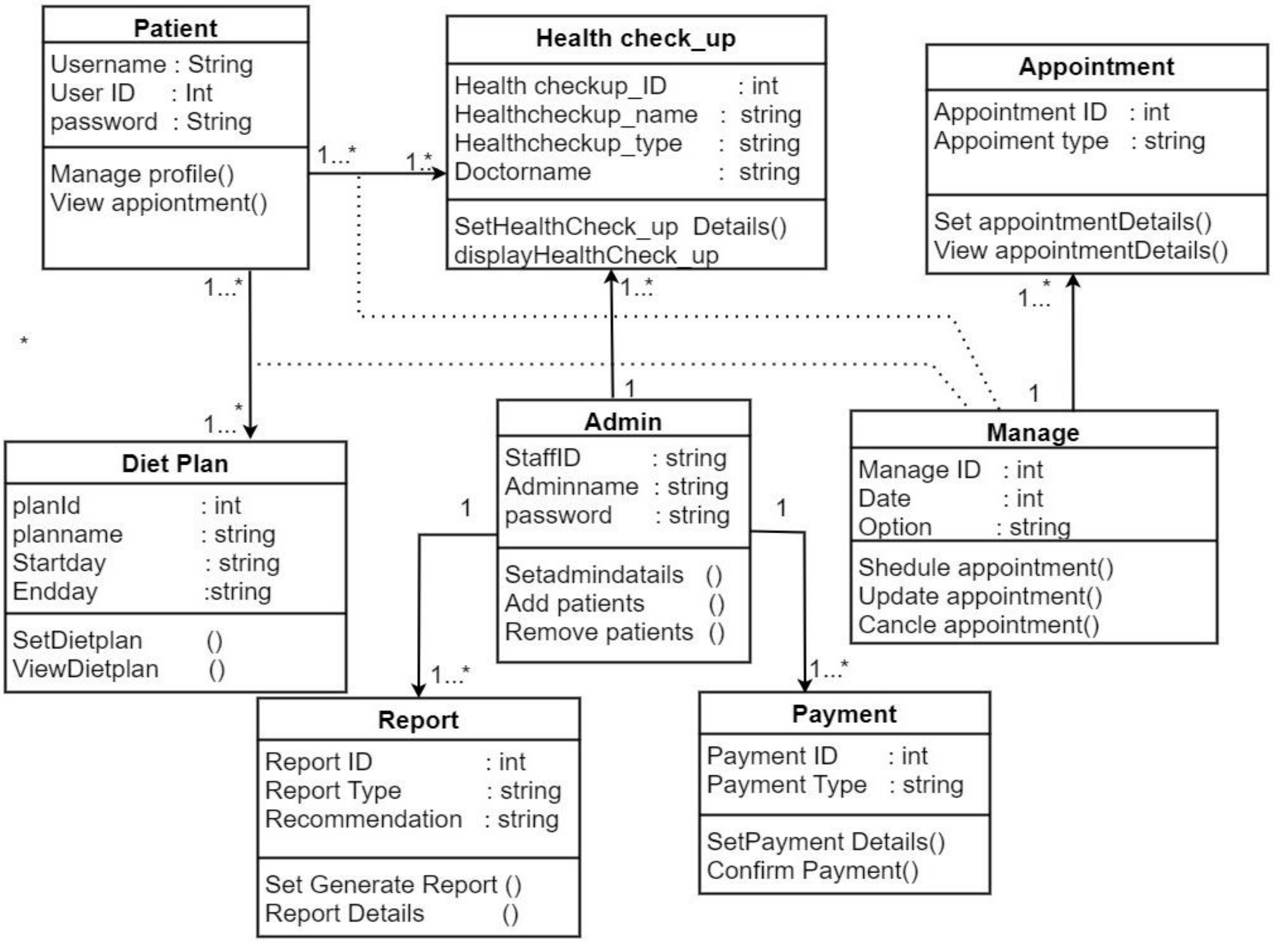
Payment	
Responsibilities:	**Collaborations:**
View payment details	Patient(users)
Validate the payment	Admin
Save payment details	Patient(users)

Health checkup	
Responsibilities:	**Collaborations:**
Obtaining details of a patient	Patient
Store patient details about who requested for test	Admin
Procedure for appointment a doctor	Appointment

Diet plan	
Responsibilities:	**Collaborations:**
Obtaining details of a patient	Patient
List of sold out of Diet plan report	Report
User request	Patient
Procedure for appointment a doctor	Appointment

Appointment	
Responsibilities:	**Collaborations:**
List of available patient appointment details	
List of appointment details of the patient concerned	patient
List of appointment details of the doctor concerned	Admin

Class Diagram for the Diet Planning & Health Check-up System



Exercise 2-

Coding for the Classes in Class Diagram

```
// patient class
// part class of diet plan & health checkup
//The operation and control of the system is done by admin and manage.
//On demand, a user (patient) can connect to the system through any
  of the diet plan or health check-up classes.
#pragma once
#include <string>
using namespace std;

class Patient
{
    private:

        string username;
        int userId;
        string password;

    public:

        int manageProfile(string newUsername, int newUserId, string newPassword);
        void viewAppointment();
};
// Patient .cpp

#include <iostream>
#include "Patient.h"

using namespace std;

int Patient::manageProfile(string newUsername, int newUserId, string newPassword)
{
    cout << "Managing profile for user: " << username << endl;
    // Implement the code to manage the patient's profile here
    // You can provide options to update username, userId, and password
    // For example:

    cout << "Enter new username: ";
    cin >> username;

    cout << "Enter new user ID: ";
    cin >> username;

    cout << "New password: ";
    cin >> password;

    cout << "Profile updated successfully!\n\n";
    return 0;
}

void Patient::viewAppointment()
{
    cout << "Viewing appointments for user: " << username << endl;
    // Implement the code to view the patient's appointments here
    // For example:
    cout << "Here are your upcoming appointments:\n";
    cout << "1. Appointment for Health check-up 1\n";
}
```

```cpp
// Health check_up  class

#pragma once
#include <string>
using namespace std;

class Health_checkup
{
    private:

        int checkup_id;
        string checkup_name;
        string checkup_type;
        string doctorname;
        string date;
        string results;

    public:

        void setHealthCheckup(int id, const string& name, const string& type,
const string& doctor, string checkupDate, const string& res);

        void displayHealthCheckup();
};
```

```cpp
// Health_checkup .cpp

#include <iostream>
#include "Health_checkup.h"
using namespace std;

void Health_checkup::setHealthCheckup(int id, const string& name, const string&
type, const string& doctor, string checkupDate, const string& res)
{
    checkup_id = id;
    checkup_name = name;
    checkup_type = type;
    doctorname = doctor;
    date = checkupDate;
    results = res;
}

void Health_checkup::displayHealthCheckup()
{
    // Accessing member variables
    cout << "Checkup ID: " << checkup_id << endl;
    cout << "Checkup Name: " << checkup_name << endl;
    cout << "Checkup Type: " << checkup_type << endl;
    cout << "Doctor: " << doctorname << endl;
    cout << "Date: " << date << endl;
    cout << "Results: " << results << endl;
}
```

```cpp
// Diet plan class

#pragma once
#include <string>
using namespace std;

class Diet_plan {

    private:

        int planId;
        string planName;
        string startDay;
        string endDay;

    public:

        void setDietPlan(int id, string name, string start, string end);
        void viewDietPlan();
};
```

```cpp
// Diet_plan .cpp


#include <iostream>
#include "Diet_plan.h"


void Diet_plan::setDietPlan(int id, string name, string start, string end)
{
    planId = id;
    planName = name;
    startDay = start;
    endDay = end;
}

void Diet_plan::viewDietPlan()
{
    cout << "Diet Plan ID: " << planId << endl;
    cout << "Diet Plan Name: " << planName << endl;
    cout << "Start Day: " << startDay << endl;
    cout << "End Day: " << endDay << endl;
}
```

```
// appointment class



#pragma once
#include <string>
using namespace std;

class Appointment
{
    private:

        int Appointmentid;
        string Appointmenttype;

    public:

        void setappointment(int appid, string Apptype);
        void viewappointmentDetails();
};
```

```
// Appointment.cpp



#include <iostream>
#include "Appointment.h"
using namespace std;

void Appointment::setappointment(int appid, string Apptype)
{
    Appointmentid = appid;
    Appointmenttype = Apptype;
}

void Appointment::viewappointmentDetails()
{
    cout << "Appointment ID: " << Appointmentid << endl;
    cout << "Appointment Type: " << Appointmenttype << endl;
}
```

```
// Report class


//Giveing Details diet plan or health class to check & Generate Report
#pragma once
#include <string>
using namespace std;

class Report
{
    private:

        int Report_id;
        string Report_type;
        string Recommendation;

    public:

        void setgeneratereport(int repid, string Retype, string Recm);
        void reportDetails();
};
```

```
// Report.cpp

#include <iostream>
#include "Report.h"

using namespace std;

void Report::setgeneratereport(int repid, string Retype, string Recm)
{
    Report_id = repid;
    Report_type = Retype;
    Recommendation = Recm;
}

void Report::reportDetails()
{
    cout << "Report ID: " << Report_id << endl;
    cout << "Report Type: " << Report_type << endl;
    cout << "Recommendation: " << Recommendation << endl;
}
```

```
// Payment class
//Payments including report class related to the service received

#pragma once
#include<string>


class Payment
{
      private:

            int payment_id;
            string Payment_type;

      public:
            // Set the details of the payment
            void setpayment_Details(int payid, string paytype);
            // Confirm the payment
            void Confirm_payment();
};
```

```
// Payment.cpp

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



void Payment::setpayment_Details(int payid, string paytype)
{
    payment_id = payid;
    Payment_type = paytype;
}

void Payment::Confirm_payment()
{
    cout << "Payment Successfully" << endl;
    cout << "Payment ID: " << payment_id << endl;
    cout << "Type: " << Payment_type << endl << endl;
}
```

```
//  Admin class
//Testing and preparation based on user data through class diet plan
  and class health check-up

#pragma once
#include <string>
using namespace std;

class Admin {

    private:

        string staffID;
        string admin_name;
        string password;


    public:

        void setadmindetails(string sid, string adname, string pwd);
        void addPatient();
        void removePatient();

};



// Admin .cpp

#include <iostream>
#include "Admin.h"



void Admin::setadmindetails(string sid, string adname, string pwd)
{
    staffID = sid;
    admin_name = adname;
    password = pwd;
}

void Admin::addPatient()
{
    // Implement logic to add a patient
    cout << "Patient added successfully!" << endl;
}

void Admin::removePatient()
{
    // Implement logic to remove a patient
    //cout << "Patient removed successfully!" << endl;
}
```

```
// Manage class
//Class administration and examination

#pragma once
#include <string>
using namespace std;

class Manage {

    private:

        int ManageId;
        string Date;
        string option;

    public:

        void scheduleAppointment(int MngId, string D, string opt);
        void updateAppointment();
        void cancelAppointment();
};

// Manage.cpp

#include <iostream>
#include <string>
#include "Manage.h"


void Manage::scheduleAppointment(int MngId, string D, string opt)
{
    // Implement logic to schedule an appointment
    ManageId = MngId;
    Date = D;
    option = opt;
    cout << "Appointment scheduled successfully!" << endl;
}

void Manage::updateAppointment()
{
    // Implement logic to update an appointment
    //cout << "Appointment updated successfully!"<< endl;
}

void Manage::cancelAppointment()
{
    // Implement logic to cancel an appointment
    //cout << "Appointment canceled successfully!" << endl;
}
```

```
// main programm .cpp
// Diet plan & Health checkup

#include <iostream>
#include <string>

#include "Patient.h"
#include "Appointment.h"
#include "Health_checkup.h"
#include "Diet_plan.h"
#include "Manage.h"
#include "Admin.h"
#include "Report.h"
#include "Payment.h"

using namespace std;

int main()
{
    // Create objects of the classes
    Patient patient;
    patient.manageProfile("John", 12345, "password");
    patient.viewAppointment();



 cout << endl;

    Appointment appointment;
    appointment.setappointment(1, "Checkup");
    appointment.viewappointmentDetails();

    cout << endl;

    Health_checkup checkup;
    checkup.setHealthCheckup(1, "Routine Checkup", "General",
      "Dr. Ruwan", "2023/05/30", "Normal");

    checkup.displayHealthCheckup();

    cout << endl;

    Diet_plan dietPlan;
    dietPlan.setDietPlan(1, "Weight Loss Plan", "2023/06/01", "2023/06/30");
    dietPlan.viewDietPlan();

    cout << endl;

    Manage appointmentManager;
    appointmentManager.scheduleAppointment(1, "2023-06-01", "Option 1");
    appointmentManager.updateAppointment();
    appointmentManager.cancelAppointment();

    cout << endl;

    Admin admin;
    admin.setadmindetails("st123", "jahan", "RJ3165");
    admin.addPatient();
    admin.removePatient();

    cout << endl;
    Report dpReport;
    dpReport.setgeneratereport(1, "Diet Plan", "Follow the recommended
                                diet plan strictly.");
    dpReport.reportDetails();

    cout << endl;

    Report hCReport;
    hCReport.setgeneratereport(2, "Health Check-up", "You are in good
                              health. Continue to follow a healthy lifestyle.");
    hCReport.reportDetails();

    cout << endl;

    Payment Pay;
    Pay.setpayment_Details(1, "online payment");
    Pay.Confirm_payment();


    return 0;
}
```

Individual Contribution -

IT Number	Name	Contribution
IT22575562	WIJEKOON W.M.MG.K.P	Class • Manage • Report
IT22567642	THILAKARATHNA P.L.B.H	Class • Health check up • Diet Plan
IT22591098	JAYALATH J.P.R.J	Class • Patient • Payment
IT22566034	PERERA K.C.D	Class • Appointment
IT22217622	HETTIARACHCHI A.N.M	Class • Admin