



TITLE: STUDENT REGISTRATION SYSTEM (SGiS)

OBJECT-ORIENTED PROGRAMMING

APPLICATION PLAN

SEMESTER JANUARY 2020

PREPARED BY	HANISHA BINTI ASNUL
STUDENT ID	23231
PROGRAMME	BUSINESS INFORMATION SYSTEM
SUBMITTED TO	DR. MOHAMED NORDIN BIN ZAKARIA

Table of Contents

1. INTRODUCTION	3
1.1 PROJECT OBJECTIVE.....	3
1.2 PROPOSED SYSTEM.....	3
2. ARCHITECTURAL DESIGN.....	4
2.1 COMPONENTS AND INTERFACES.....	4
2.1.1 Components	4
2.1.2 Interfaces	4
3. UML DIAGRAM.....	6
4. FIRST DUMMY CODE IMPLEMENTATION	7
5. ADVANCEMENT FROM FIRST DUMMY CODE.....	9
5.1 SERVER INTERFACE.....	9
5.2 DUMMY CODE FOR SERVER INTERFACE	9
6. CLIENT MAIN INTERFACE.....	11

1. INTRODUCTION

1.1 PROJECT OBJECTIVE

The propose of this Student Registration System (SGiS) project is to build a student registration system that will completely automate the process of new student registration in the university. The system will handle the document submission, testing process and registration of the new students. The system will be based and will have two implementations (i.e. client side [student] and server side [university]). Implementation on the server side can only be reached through the university intranet, while the client side can be reached by internet.

The cycle begins as prospective students decide to enroll in class. They would be supposed to visit the university if they show interest in some course and will establish a user account at the administration door. It is the only aspect of the program where each user needs to provide human contact, which is meant as a protection mechanism to avoid fake user IDs from being generated. SGiS will assign the student a unique roll number and register them as a student of the university. This SGiS project will transfers the latest student information directly into the database of the university students.

1.2 PROPOSED SYSTEM

The proposed SGiS system is to remove the paper trail required to complete the numerous formalities. This system will manage the whole cycle of prospective student enrolment before new student enrolment. The implementation of this system is to removes the confusion that prospective students may misunderstand.

2. ARCHITECTURAL DESIGN

2.1 COMPONENTS AND INTERFACES

2.1.1 Components

- Student Component

This is one of the Student Registration System's main components. It is where the collection method for the student subject is applied. It contains the summary of grades, the overview of the semester and the features for profiles.

- User Management Component and Authentication

This is the main sub-system which is responsible for the Student Registration system security. It authenticates users and also performs user management tasks such as developing new user accounts, deleting device accounts, etc. This portion also implements the "privileged access control matrix."

- Subject Component

This is the main component that performs the functions related to the administrator's subject operations such as introducing a new subject, changing credits to an existing particular subject and removing subjects etc. It is also the duty of the subject component to view the available subject list for each semester.

2.1.2 Interfaces

Student Component

studentSelectSubjects: This framework includes the list of subjects available for the semester. This helps the student to pick the subjects. This interface is the bridge between the part of the student and the component of the subject.

studentProfile: This interface is used to access and edit details about the student's information.

User Management Component and Authentication

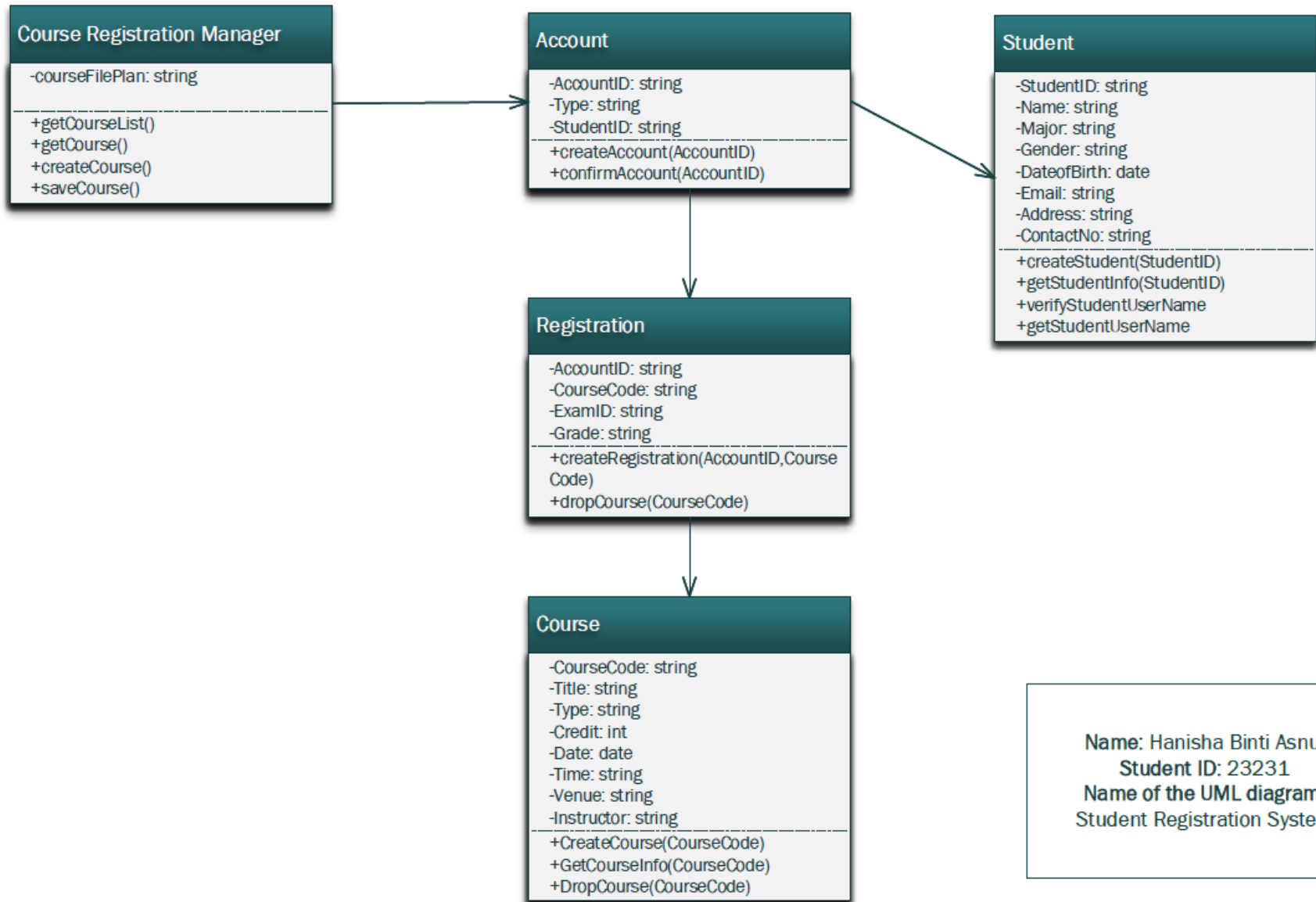
authenticateUser: This is the framework where users can sign in to the device.
This will direct users to their respective home page.

Subject Component

newSubject: This interface is where administrator can add new subjects to the offered courses. The components of Subject and Student are related.

editSubject: The Administrator changes current subjects in this interface. The components of Subject and Student are related.

3. UML DIAGRAM



4. FIRST DUMMY CODE IMPLEMENTATION

```
public class StudentRegistrationSystem
{
    private File courseFilePlan;
    private Students[] students;

    public int getNumberOfStudents()
    {
        return validStudents;
    }

    private double getStudentID(int s)
    {
        return Double.stuDouble(students[s].getStudentID());
    }

    private void getCourseList(File f)
    {
        currentFile = f;
    }

    private void getCourse()
    {
    }

    private void createCourse()
    {
    }

    private void saveCourse()
    {
    }

    private void beginAddStudent()
    {
    }

    public static void main(String[] args)
    {
        RegistrationSystem system = new RegistrationSystem();
    }
}
```

```
public void actionPerformed(ActionEvent e)
{

}

}
```


5. ADVANCEMENT FROM FIRST DUMMY CODE

5.1 SERVER INTERFACE

```
public interface ServerIntf extends Remote
{
    public boolean validUser(String user, String pass) throws RemoteException;
    public Vector retNotDoneSubList(String sid) throws RemoteException;
    public Vector retTakenSubList(String sid) throws RemoteException;
    public void regSub(Vector v, String sid) throws RemoteException;
}
```

5.2 DUMMY CODE FOR SERVER INTERFACE

```
class ServerImpl extends UnicastRemoteObject implements ServerIntf
{
    public ServerImpl() throws RemoteException
    {
    }

    public void regSub(Vector v, String sid)
    {
        System.out.println

        try
        {

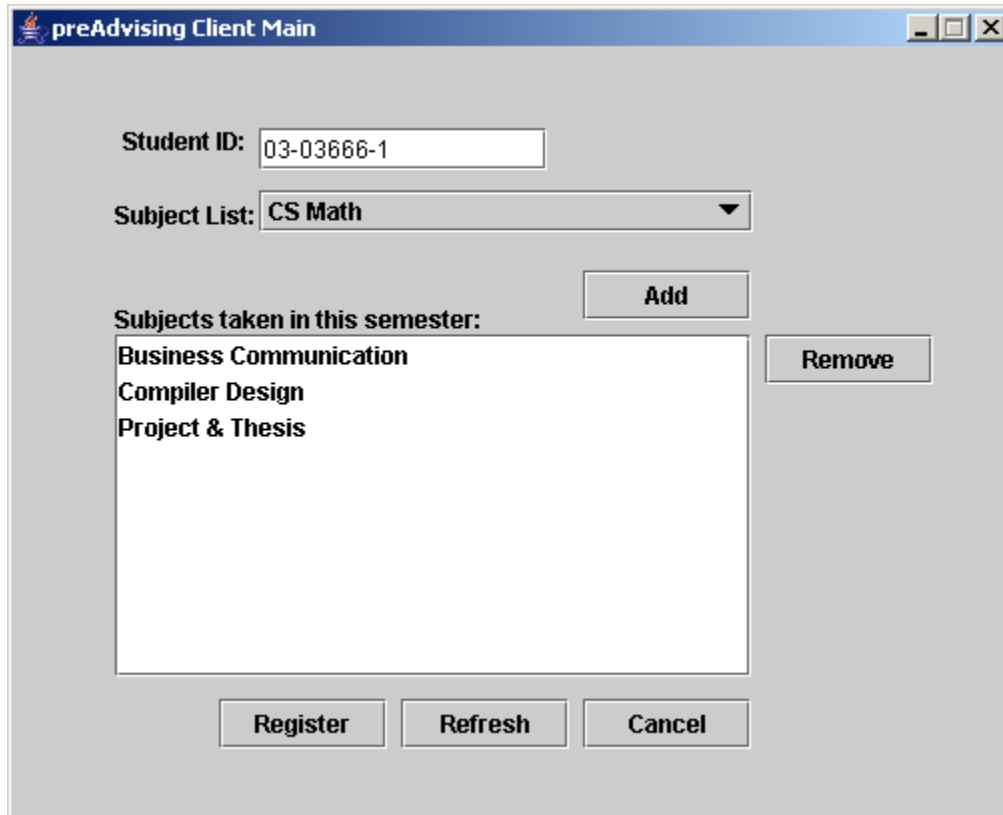
        }
        catch(Exception e)
        {
            System.out.println(e);
        }
    }

    public Vector retNotDoneSubList(String sid)
    {
    }
    catch(Exception e)
    {
        System.out.println(e);
    }
    return v;
}

    public Vector retTakenSubList(String sid)
    {
        try
        {
        }
        catch(Exception e)
        {
            System.out.println(e);
        }
    }
}
```

```
    }  
    return v;  
}  
  
public boolean validUser(String user, String pass)  
{  
    catch(Exception e)  
    {  
        System.out.println(e);  
    }  
    return res;  
}  
}  
  
public class Server  
{  
    public static void main(String args[])  
    {  
        try  
        {  
        }  
        catch(Exception e)  
        {  
            System.out.println(e);  
        }  
    }  
}
```

6. CLIENT MAIN INTERFACE



The image shows a software window titled "preAdvising Client Main". It contains a "Student ID" text box with the value "03-03666-1", a "Subject List" dropdown menu currently showing "CS Math", and a list of subjects taken in the semester: "Business Communication", "Compiler Design", and "Project & Thesis". There are "Add" and "Remove" buttons next to the subject list, and "Register", "Refresh", and "Cancel" buttons at the bottom.

preAdvising Client Main

Student ID: 03-03666-1

Subject List: CS Math ▼

Subjects taken in this semester:

- Business Communication
- Compiler Design
- Project & Thesis

Add Remove

Register Refresh Cancel