

Course Enrollment System for NSBM University

Introduction

This is a stand-alone System for the NSBM University. The System implemented for Administrator and Students. Java is the only language used to implement the system. In this system Object oriented concept, Exception handling and GUI java Swing are used.

In this system admin can only register Students. Students can't create an account if they not registered.

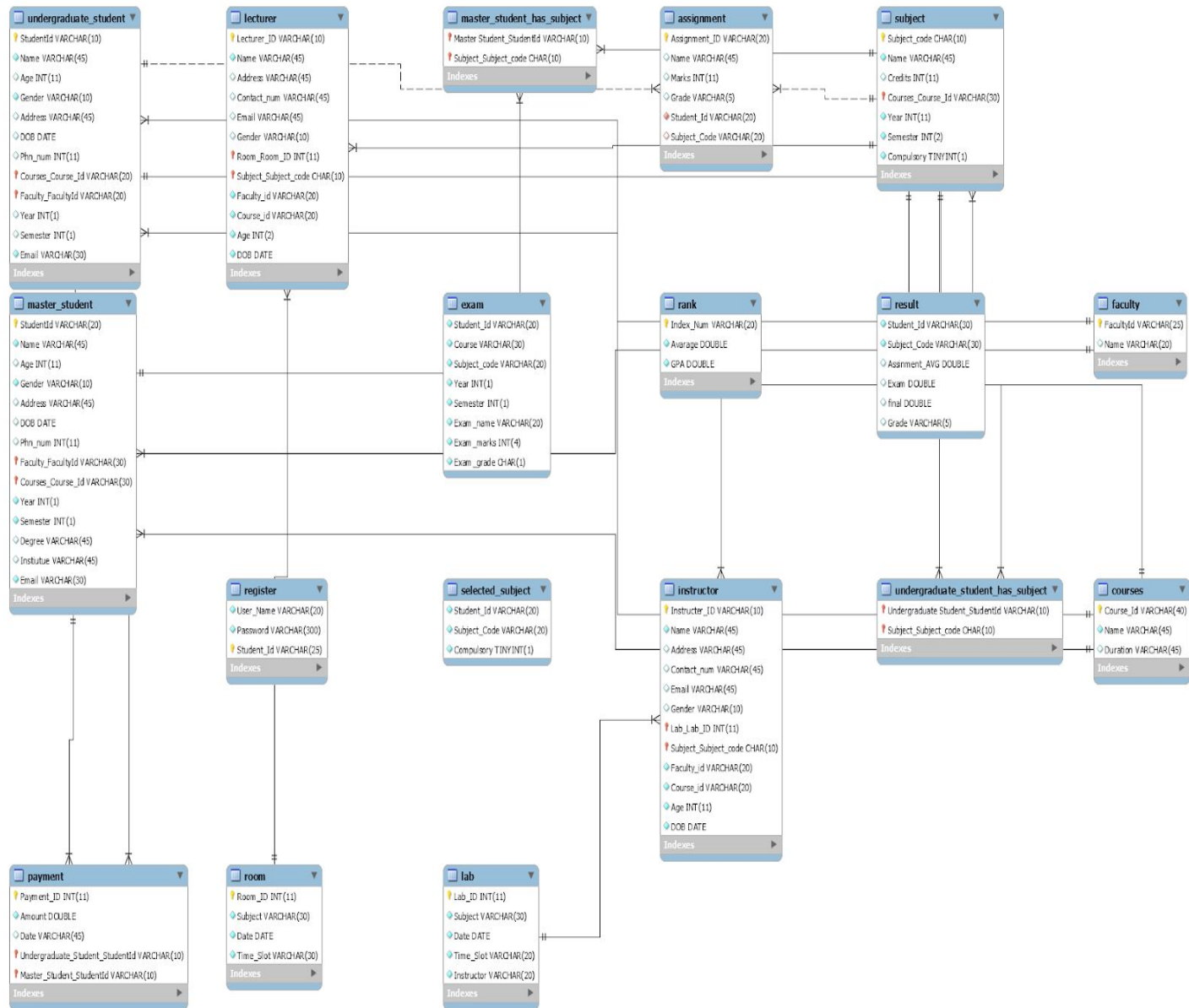
Classes of the System

- Undergraduate Student
- Postgraduate Student
- Lecturer
- Instructor
- Subject
- Selected Subject
- Practical Subject
- Payment
- Result
- Assignment
- Lab
- Room

Used Libraries

- Mysql JDBC Driver
- Javax.mail.jar
- JDK 1.8

ER Diagram

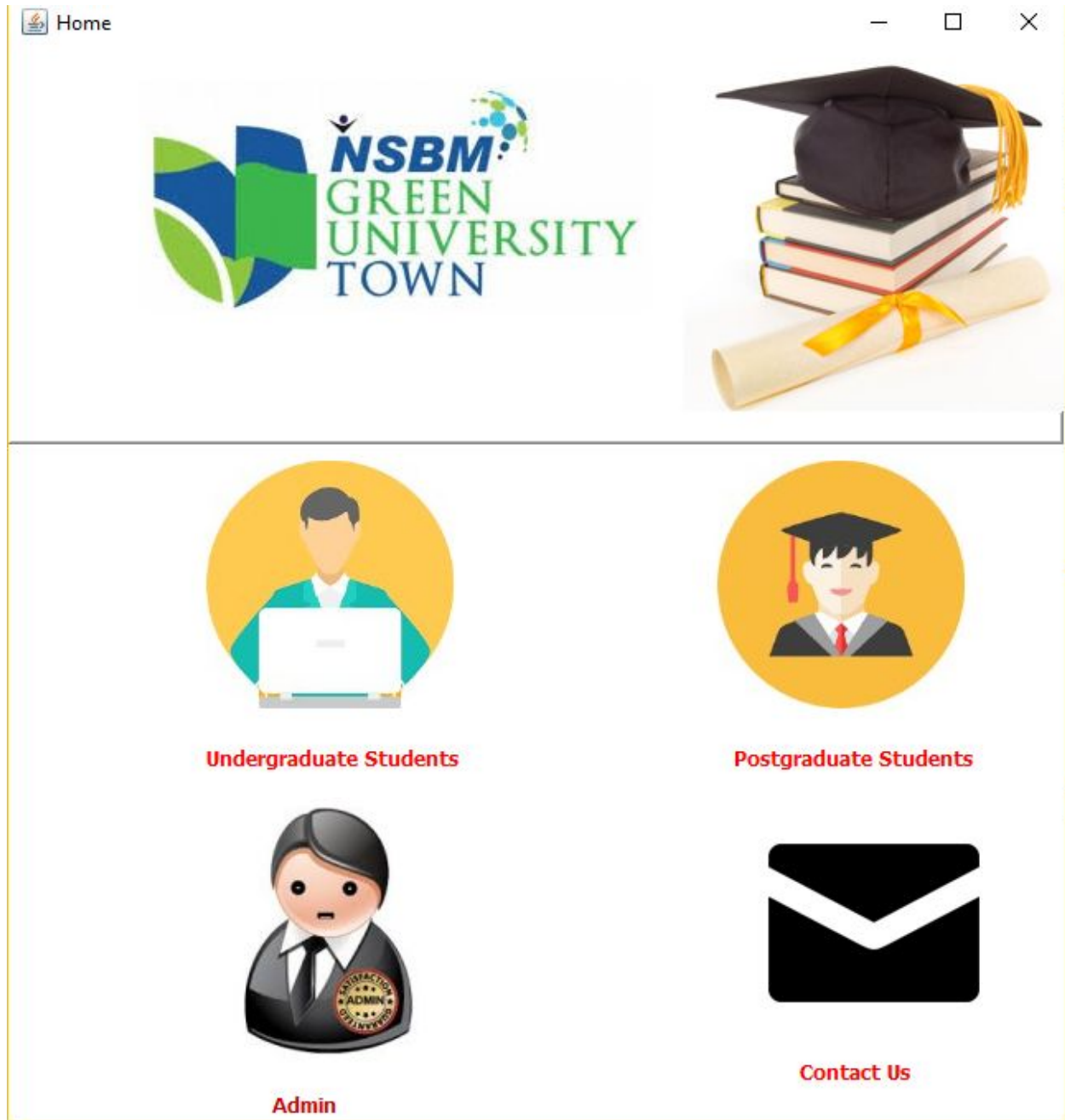


Functions of the System

- Admin
 - Register Students
 - Add Lectures and Instructors
 - Manage Informations of Students,Lectures,Instructors
 - Manage Results of Students.
 - Add Subjects
 - Manage Payments
 - Manage Class Allocation
 - Send Emails
 - Calculate GPA

- Students
 - View results of Exams,Assignments
 - Choose Subjects
 - Change Subjects

Home Page



Used Buttons

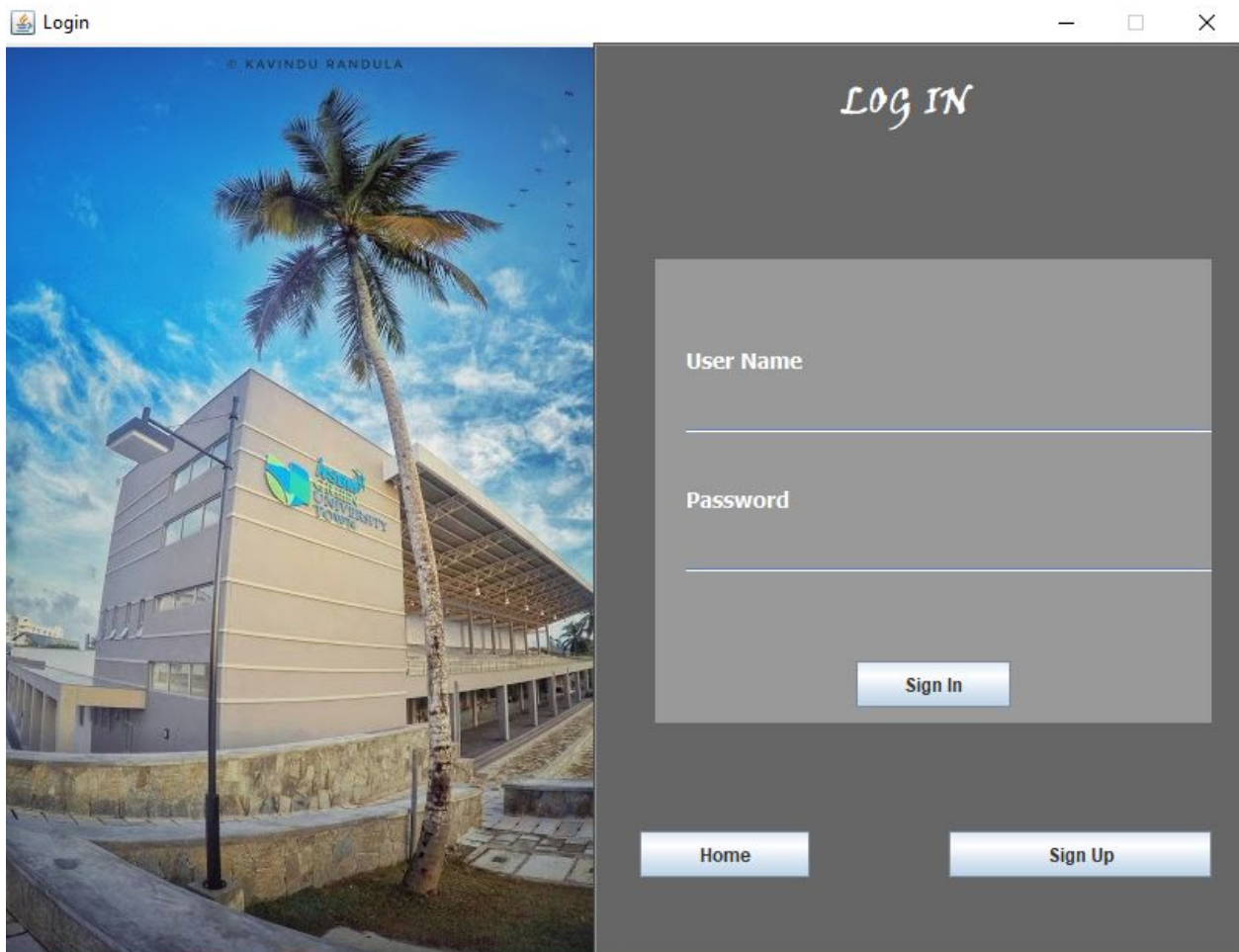
- **Undergraduate Student** : Enter to the System as Undergraduate Student.
- **Postgraduate Student** : Enter to the System as Postgraduate Student.
- **Admin** : Enter to the System as Admin

- **Contact Us** : Can view email and phone number of NSBM.

When the user enter to the System,

- User can click Relevant button and prompt Username and Password. After User can Enter to System

Login Page



- This is the login page
- Students cannot Create Account, if they didn't register in NSBM.
- Registration is done only by Admin

- After Registration Students can create Account and enter to the system.

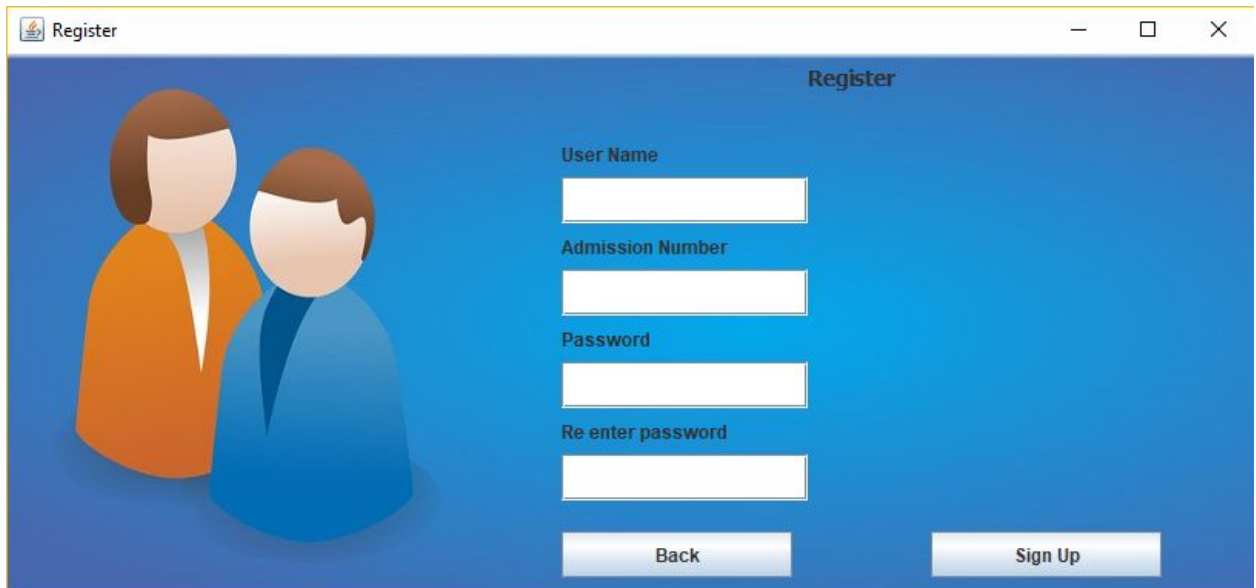
Used Methods

- **Sign_up_controll.search(username, password)** : Search database exists given username and password . if exists return true.if not return false.

Used Buttons

- **Sign In** : Call Sign_up_controll.search(username, password) method, if method return true enter to the system if not display error message.
- **Home**: return to the home page.
- **Sign up** : create a new account.

Register page



The screenshot shows a web browser window with the title "Register". The page has a blue background. On the left side, there is an illustration of two stylized figures, one in an orange jacket and one in a blue jacket. On the right side, there are four input fields labeled "User Name", "Admission Number", "Password", and "Re enter password". Below these fields are two buttons: "Back" and "Sign Up".

Used Methods

- **Sign_up_controll.checkvalidity(String id)** : Search database there exists Account from given username .if Account exists return true else return false
- **Sign_up_controll.add(username ,password,Student id)** : Add Username and password to database

Used Buttons

- **Sign up** : first call Sign_up_controll.checkvalidity(String id) method if it return true then call add(username ,password,Student id) method else display error message.
- **Back** : return to previous page.

Functions of the System

System Admin

System Admin is the only one who can register Students,Lectures,Instructors for NSBM.

Admin Functions

❑ Register Undergraduate

The screenshot shows a web application window titled "Register Undergraduate". Inside, there is a form titled "Register Student for Bachelor Course". The form is divided into two columns. The left column contains text input fields for "Student Id", "Full Name", "Age", "Email", "Contact Number", and "Date Of Birth". The right column contains radio buttons for "Gender" (Male and Female), a text area for "Address", a dropdown menu for "Faculty" (currently showing "School of Computing"), a dropdown menu for "Year" (currently showing "1"), radio buttons for "Semester" (1 and 2), and a dropdown menu for "Course" (currently showing "Computer Science"). At the bottom of the form are two buttons: "Back" and "Register".

Register Undergraduate

Register Student for Bachelor Course

Student Id

Full Name

Age

Email

Contact Number

Date Of Birth

Gender
☒ Male ☐ Female

Address

Faculty
School of Computing ▼

Year
1 ▼

Semester
☒ 1 ☐ 2

Course
Computer Science ▼

Back Register

Used Models

- Undergraduate Student

Used Methods

- `Undergraduate_Student_controll.Add(Undergraduate_Student)`: Add Undergraduate Student details to the database

Used Buttons

- **Back** : Goto Previous Page
- **Register** : Get details from text fields and create Undergraduate_student object after Call Add (Undergraduate_Student) function

❑ Register Postgraduate

Register Student for Master Course

Student Id

Full Name

Age

Email

Contact Number

Date Of Birth

Institute

Degree

Gender
☒ Male ☐ Female

Address

Faculty
 School of Business ▼

Year
 1 ▼

Semester
☒ 1 ☐ 2

Course
 Human Resource Management ▼

Back Register

Used Models

- Postgraduate Student

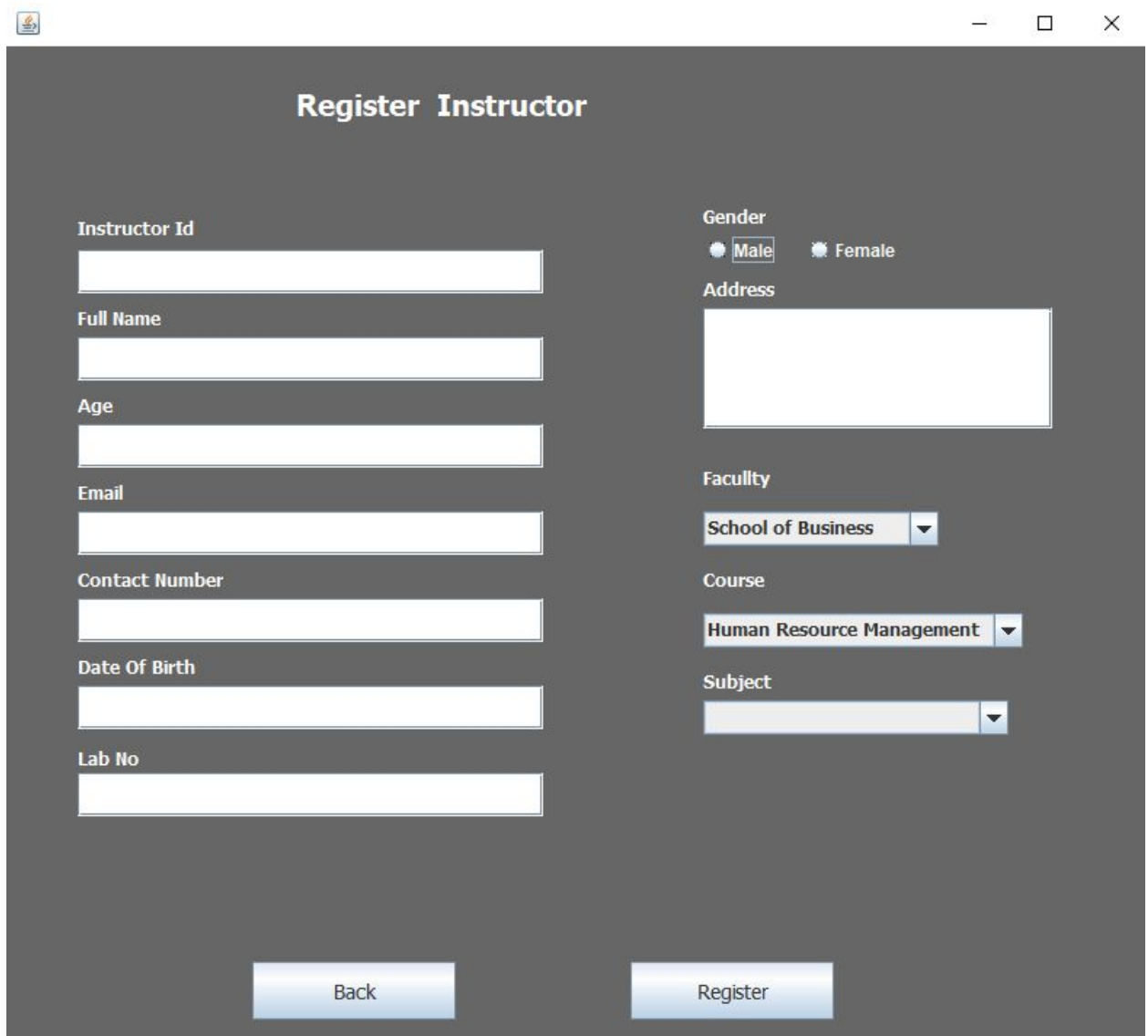
Used Methods

- **Postgraduate_student_controll.Add(Postgraduate_Student):** Add Postgraduate Student to Database

Used Buttons

- **Back** : Goto Previous Page
- **Register** :Get details from text fields and create Postgraduate_student object after Call Add (Postgraduate_Student) function

❑ Register Instructors



Register Instructor

Instructor Id

Full Name

Age

Email

Contact Number

Date Of Birth

Lab No

Gender
☒ Male ☐ Female

Address

Faculty
School of Business ▼

Course
Human Resource Management ▼

Subject
 ▼

Back **Register**

Used Models

- Instructor

Used Methods

- **Instructor_controll.Add(instructor)**: Add instructor to Database

Used Buttons

- **Back** : Goto Previous Page
- **Register** :Get details from text fields and create Instructor object after Call Add (Instructor) function

❑ Register Lecturer

Register Lecturer

Lecturer Id

Full Name

Age

Email

Contact Number

Date Of Birth

Room No

Gender
☒ Male ☐ Female

Address

Faculty
School of Business ▼

Course
Human Resource Management ▼

Subject
 ▼

Back Register

Used Models

- Lecturer

Used Methods

- **Lecturer_controll.Add(Lecturer)**: Add Lecturer to Database

Used Buttons

- **Back** : Goto Previous Page
- **Register** :Get details from text fields and create Lecturer object after Call Add (Lecturer) function

❑ Update Undergraduate Student ,PostGraduate,

Update Undergraduate

Insert Student ID: 2016cs129 [Search]

STUDENT ID: 2016cs129

STUDENT NAME: Lahiru Dulanjaya Senadheera

AGE: 22

GENDER: Male

ADDRESS: 301/5,Habarakada,Homagama

DATE OF BIRTH: 1995-03-03

PHONE NUMBER: 758034032

FACULTY: School of Computing

COURSE: Computer Science

YEAR: 1

SEMESTER: 1

[RESET] [UPDATE] [DELETE] [BACK]

Used Models

- Undergraduate Student

Used Methods

- **Undergraduate_Student_controll.search(Admission Number):**get details of Student from Database
- **Undergraduate_Student_controll.Undergraduate_Student_delete(Admission Number):**Delete Student details from Database
- **Undergraduate_Student_controll.Undergraduate_Student_update(Undergraduate Student):**Update Student details to Database

Used Buttons

- **Back** : Goto Previous Page
- **Reset** : set text fields to empty.
- **Update** :Call Undergraduate_Student_update(Undergraduate Student) method and update the details and store in the database.
- **Delete** : Call Undergraduate_Student_delete(Admission Number) method and Delete Student details from database.
- **Search** : get details of student from database and fill in the fields.

As Undergraduate Student , Admin can Update Postgraduate Details.

❑ Update Lecture and Instructors Details

Insert Lecturer ID

LECTURER ID

LECTURER NAME

AGE

GENDER

ADDRESS

DATE OF BIRTH

PHONE NUMBER

FACULTY

COURSE

SUBJECT

ROOM NO

EMAIL

Used Models

- Instructor

Used Methods

- **search(Instructor):**get details of Instructor from Database
- **Instructor_delete(Instructor):**Delete Instructor details from Database
- **Instructor_update(Instructor):**Update Instructor details to Database

Used Buttons

- **Back** : Goto Previous Page
- **Reset** : set text fields to empty.
- **Update** : call Instructor_update(Instructor) and update the details of instructor to database
- **Delete** : call Instructor_delete(Instructor) method Delete Instructor details.
- **Search** :call search(Instructor) method and get details of Instructor and fill the text fields

As Instructors,Admin can Update Details of Lectures.

❑ Admin can Show all Students in Faculties.

The screenshot shows a web application window titled "Show All" with standard window controls (minimize, maximize, close). Below the title bar, there is a "Faculty" section with three radio buttons: "School of Business", "School of Computing" (which is selected), and "School of Engineering". To the right of these buttons is a "Show Students" button. Below this section is a table with the following data:

Student Id	Full Name	Age	Gender	Address	Date of Birth	Phone number
2016cs121	Dulanjaya kasun	23	Male	303,Habarakada,...	1995-03-09	758234540
2016cs123	lalana pri	20	Male	22,kaduwala ,Ho...	1993-02-02	774234121
2016cs129	Lahiru Dulanjaya ...	22	Male	301/5,Habarakad...	1995-03-03	758034032

Below the table is a large empty rectangular area. At the bottom left of the window is a "Back" button.

Models

- Undergraduate Student

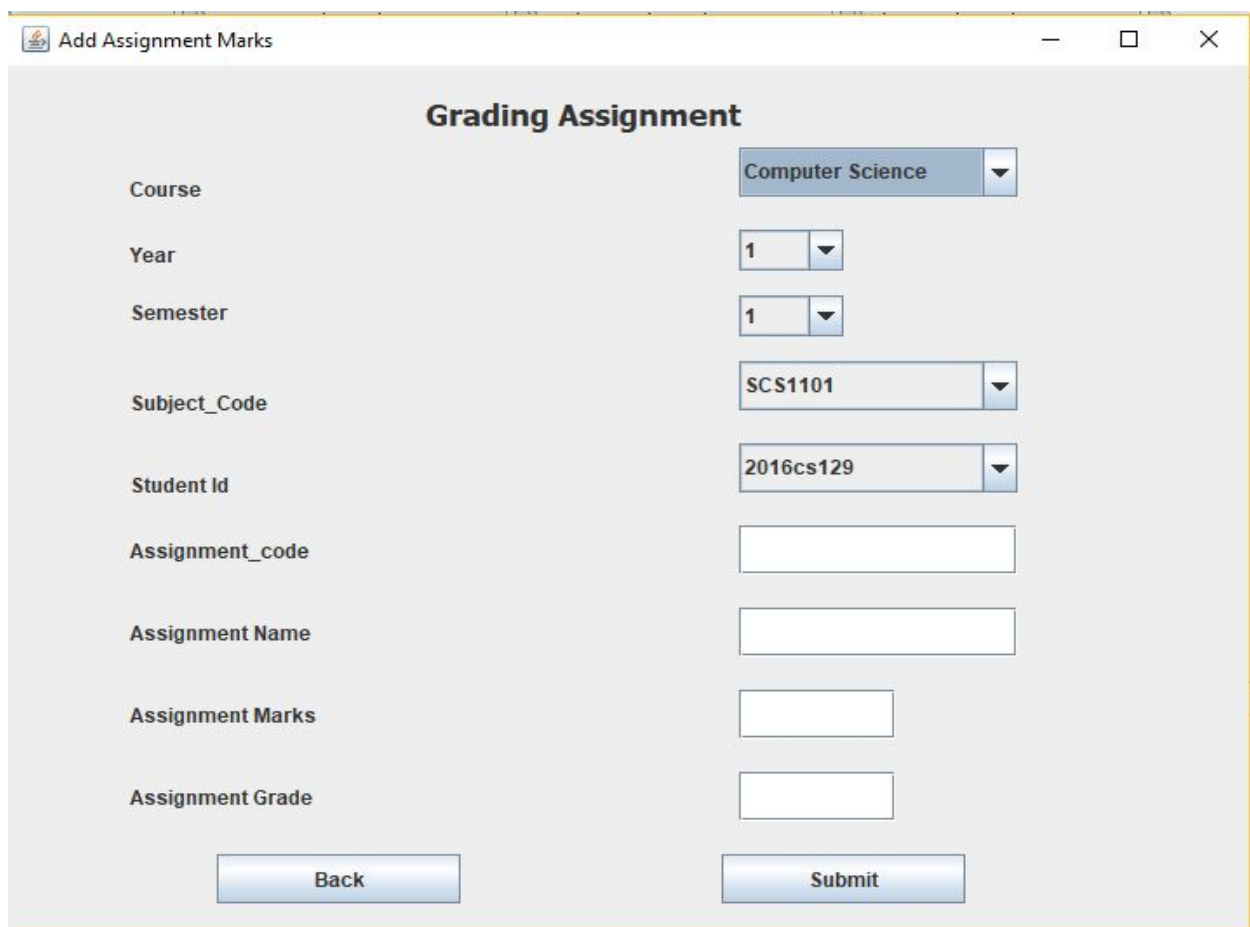
Methods

- **Undergraduate_Student_controll.getall()**:get details of all Students from Database

Buttons

- **Back** : Goto Previous Page
- **Search** : call getall() method and get details of students And fill the table.

❑ Admin can Add Assignments Marks and Exam Marks For Students



The screenshot shows a window titled "Add Assignment Marks" with a "Grading Assignment" form. The form contains the following fields and controls:

- Course**: A dropdown menu currently showing "Computer Science".
- Year**: A dropdown menu currently showing "1".
- Semester**: A dropdown menu currently showing "1".
- Subject_Code**: A dropdown menu currently showing "SCS1101".
- Student Id**: A dropdown menu currently showing "2016cs129".
- Assignment_code**: A text input field.
- Assignment Name**: A text input field.
- Assignment Marks**: A text input field.
- Assignment Grade**: A text input field.
- Back**: A button at the bottom left.
- Submit**: A button at the bottom right.

Used Models

- Assignment

- Undergraduate Student
- Postgraduate Student

Used Methods

- **Assignment_controll.Add(Assignment):** Add Assignment results to Database
- **fillcombo()** : Automatically fill course combo box
- **fillcombo1()** : Automatically fill Subject Code combo box
- **fillcombo2()** : Automatically fill Student id combo box

Used Buttons

- **Back** : Goto Previous Page
- **Submit** : Create “Assignment” object using details of text fields after Call Add(Assignment) method

The screenshot shows a Windows-style application window titled "Add Exam Marks". The window contains a form with the following fields and controls:

- Course:** A dropdown menu currently displaying "Computer Science".
- Year:** A dropdown menu currently displaying "1".
- Semester:** A dropdown menu currently displaying "1".
- Subject_Code:** A dropdown menu currently displaying "SCS1102".
- Student Id:** A dropdown menu currently displaying "2016cs129".
- Exam Name:** An empty text input field.
- Exam Marks:** An empty text input field.
- Exam Grade:** An empty text input field.
- Buttons:** Two buttons at the bottom, "Back" and "Submit", both with a light blue gradient and a slight shadow.

Used Models

- Exam
- Undergraduate Student

- Postgraduate Student

Used Methods

- **Exam_controll.Add(Exam):** Add Examresults to Database
- **fillcombo()** : Automatically fill course combo box
- **fillcombo1()** : Automatically fill Subject Code combo box
- **fillcombo2()** : Automatically fill Student id combo box

Used Buttons

- **Back** : Goto Previous Page
- **Submit** :Create “Exam” object using details of text fields Call Add(Exam) function

- ❑ Add average and GPA of each Students

GPA

CALCULATE GPA

Index Number

Index Number	Subject Code	Assignment Marks	Exam Marks	Final Marks	Grade	GPV Value
--------------	--------------	------------------	------------	-------------	-------	-----------

Average

GPA

Models

- Rank
- Result
- Exam
- Assignment
- Undergraduate Student
- Postgraduate Student

Methods

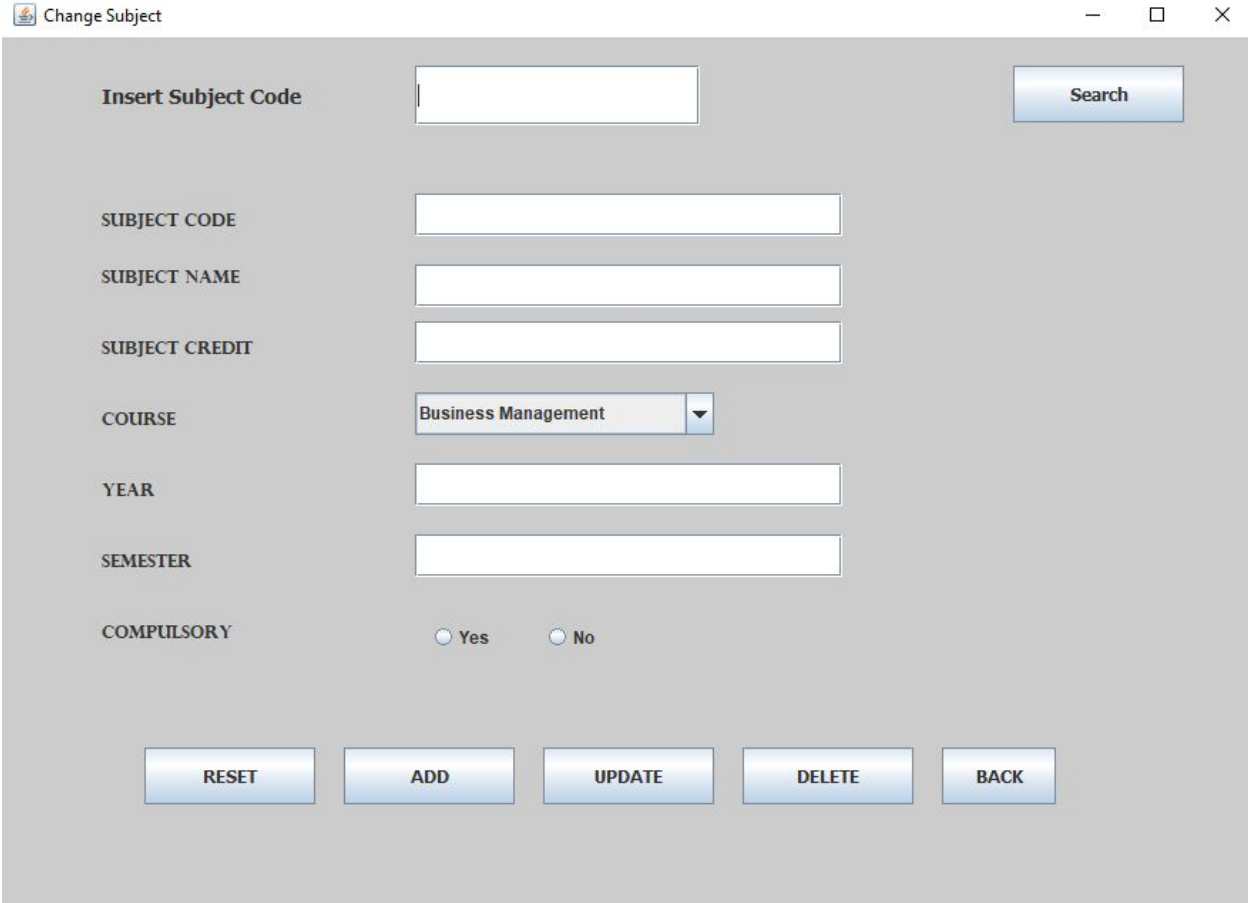
- **add(Student id,Average,GPA):** Add Final results(GPA,Average) to Database

Buttons

- **Back :** Go to Previous Page

- **Add** : Call add(Student id,Average,GPA) function

❑ Admin can also Update ,Insert Subjects.



The screenshot shows a window titled "Change Subject" with standard window controls (minimize, maximize, close) in the top right corner. The window contains the following elements:

- Insert Subject Code**: A text input field with a "Search" button to its right.
- SUBJECT CODE**: A text input field.
- SUBJECT NAME**: A text input field.
- SUBJECT CREDIT**: A text input field.
- COURSE**: A dropdown menu currently showing "Business Management".
- YEAR**: A text input field.
- SEMESTER**: A text input field.
- COMPULSORY**: Two radio buttons labeled "Yes" and "No".
- Buttons**: A row of five buttons at the bottom: "RESET", "ADD", "UPDATE", "DELETE", and "BACK".

Used Models

- Subject
- Course

Used Methods

- **Subject_controll.Add(Subject):** Add Examresults to Database
- **Subject_controll.getsub(Subject Code):** get details of subject from database
- **Subject_controll.Subject_update(Subject):** update details of subject and store in database
- **Subject_controll.Subject_delete(Subject Code):** delete details of subject from database
- **fillcombo()** : Automatically fill course combo box

Used Buttons

- **Back** : Goto Previous Page
- **Reset** : all the Text fields set empty
- **Update** : call Subject_update(Subject) method update the details
- **Delete** : call Subject_delete(Subject Code) method and Delete Subject details
- **Search** : get details of Subject and fill the text fields

- ❑ Admin can Send result sheet to Student Using Email.

The screenshot shows a web application window titled "Send Mails". It has two tabs: "Undergraduate" (selected) and "Postgraduate". The main heading is "Send Mails to Undergraduate". Below this, there are three input fields: "Enter Sudent Index Number" (with a "Check" button), "Student Email", and "Result" (a larger text area). At the bottom, there are "Back" and "Send" buttons.

Used Models

- Rank
- Result
- Undergraduate Student
- Postgraduate Student

Used Buttons

- **Back** : Goto Previous Page
- **Check** : get Email, and Student result from database according to the given Admission number and fill the fields
- **Send** : send result to relevant email address

- ❑ Admin can allocate class

The screenshot shows a Windows-style application window titled "Class Allocation". It has two tabs: "Class" (selected) and "Lab". The main area is titled "Class Allocation" and contains four input fields: "Room number" (text box), "Subject" (dropdown menu showing "BM1101"), "Date" (text box), and "Time Slot" (text box). At the bottom, there are two buttons: "Back" and "Allocate".

Used Models

- Room

Used Methods

- **class_allocation_controll.Add(Room)**: Add Room allocate details to Database
- **fillcombo1()**: Automatically fill Subject code combo box

Used Buttons

- **Back** : Goto Previous Page
- **Allocate**: Call Add(Room) function

The screenshot shows a window titled "Class Allocation" with two tabs: "Class" and "Lab". The "Lab" tab is active, displaying a form titled "Lab Allocation". The form contains the following fields and controls:

- Lab number: A text input field.
- Instructor Id: A dropdown menu with "ins-1" selected.
- Subject Code: A dropdown menu with "BM1101" selected.
- Date: A text input field.
- Time Slot: A text input field.
- Buttons: "Back" and "Allocate" buttons at the bottom.

Used Models

- Lab

Used Methods

- **class_allocation_controll.Add(Lab):** Add Lab allocate details to Database
- **fillcombo2():** Automatically fill instructor id combo box
- **fillcombo1():** Automatically fill Subject code combo box

Used Buttons

- **Back :** Goto Previous Page
- **Allocate:** Call Add(Lab) function

❑ Admin can Manage payment

- Add Payment

The screenshot shows a Windows application window with a title bar containing a small icon and standard window controls (minimize, maximize, close). The window has two tabs: 'Add Payment' (active) and 'Details'. The main area has a light blue background. It contains two text input fields. The first field is labeled 'Admission Numb...' and the second is labeled 'Subject Code'. Below the first field is a 'Back' button, and below the second field is an 'Add Payment' button.

Used Models

- Undergraduate Student
- Postgraduate Student
- Selected Subject

Used Methods

- **Payment_controll.Add(Payment):** Add Payment details to Database

Used Buttons

- **Back :** Goto Previous Page
- **Add Payment:** Call Add(Payment) function

- Get details of payments

Enter Index Number

Subject Code	Payment
SCS1103	paid
SCS1104	not paid
SCS1107	paid
SCS1109	paid
SCS1101	not paid
SCS1102	not paid
SCS1105	paid
SCS1106	paid

Used Models

- Undergraduate Student
- Postgraduate Student
- Selected Subject

Used Methods

- **Payment_controll.getpayment(Student id)** = Add Chosen Subject Codes to the text field from database.
- **Payment_controll.check(Student id)** = check whether Student paid or not

Used Buttons

- **Check** : Call Payment_controll.getpayment(Student id) and Payment_controll.check(Student id) function ,And fill the text fields

Students Functions

- ❑ Choose subjects.

The screenshot shows a web application window titled "Choose Subjects". It features a form for entering an "Admission Number" (2016cs129) and a "View Subjects" button. Below this, there are two tables. The first table, "Compulsory Subjects", lists four subjects: Data Structures & Algorithms-1, Programming-1, Maths-1, and Laboratory-1. The second table, "Select Subjects", lists six subjects with checkboxes for selection: Database-1, Computer System-1, Computer Architecture-1, Data communications and networks-1, Web Based Application Development-1, and Professional Development-1. At the bottom, there are "Back" and "Confirm" buttons.

Subject Code	Subject
SCS1101	Data Structures & Algorithms-1
SCS1102	Programming-1
SCS1105	Maths-1
SCS1106	Laboratory-1

Subject Code	Subject	Select
SCS1103	Database-1	<input type="checkbox"/>
SCS1104	Computer System-1	<input type="checkbox"/>
SCS1107	Computer Architecture-1	<input type="checkbox"/>
SCS1108	Data communications and networks-1	<input type="checkbox"/>
SCS1109	Web Based Application Development-1	<input type="checkbox"/>
SCS1110	Professional Development-1	<input type="checkbox"/>

Student can choose subjects according to the credits . so Student can select four subjects.if student select wrong number of subjects then the system get error message to student.

Used Models

- Subject
- Selected Subject

Used Methods

- **Selected_Subject_controll.add_subject(Selected_Subjects):** Add Selected Subject details to Database

- **Selected_Subject_controll.allreadyexists(Student id):**Check whether Student had already choose the subjects.

Used Buttons

- **Back :** Goto Previous Page
- **View Subject:** get all details about subjects and display in table.
call allreadyexists(Student id) method if it return true it's mean student already chosen the subjects, show message to "you already chosen" else Student can choose subjects.
- **Confirm:**call add_subject(Selected_Subjects) method

After chosen Student have few times to change their choice.After the time is over admin disable the ability of change subject.

[Change Subject](#)

Change Subjects

Admission Number

2016cs129

View Subjects

Compulsory Subjects

Subject Code	Subject
SCS1101	Data Structures & Algorithms-1
SCS1102	Programming-1
SCS1105	Maths-1
SCS1106	Laboratory-1

Select Subjects

Subject Code	Subject	Select
SCS1103	Database-1	<input type="checkbox"/>
SCS1104	Computer System-1	<input type="checkbox"/>
SCS1107	Computer Architecture-1	<input type="checkbox"/>
SCS1108	Data communications and networks-1	<input type="checkbox"/>
SCS1109	Web Based Application Development-1	<input type="checkbox"/>
SCS1110	Professional Development-1	<input type="checkbox"/>

Back

Change

Used Models

- Subject
- Selected Subject

Used Methods

- **Selected_Subject_controll.update_subject(Selected_Subjects):**
Update Selected Subject details to Database

Used Buttons

- **Back** : Goto Previous Page

- **View Subject:** get all details about subjects and display
- **Confirm :** call update_subject(Selected_Subjects) method

☐ View Exam marks.

Exam Grades

Admission Number

GET RESULT

SUBJECTS	GRADE	MARKS
SCS1101	90	A
SCS1102	60	c

Back

Used Models

- Exam
- Undergraduate Student
- Postgraduate Student


Used Methods

- **Exam_controll.getall(Student Id) :** Get all the Exam marks from the database.

Used Buttons

- **Back** : Goto Previous Page
- **Get Result**: Get Exam result details from database and fill the table.

❏ [View Assignment marks.](#)


— □ ×

Assignment Grades

Admission Number

Subject code
 ▼

GET RESULT

ASSIGNMENT	GRADE	MARKS
stack	69	B
b	55	C
c	60	C

Back

Used Models

- Assignment
- Undergraduate Student
- Postgraduate Student

Used Methods

- **Assignment_controll.getall(Student Id,subject_code)** : Get all the Assignments marks of given subject code from the database.

- **fillcombo()** : Automatically fill Subject Code combo box.

Used Buttons

- **Back** : Go to Previous Page
- **Get Result**: Get Assignment result details from database and fill the table.