# Code Explanation for Student Course Registration System

## 1. Controller

**CourseRegistorServlet.java**

- A class extend HTTP servlet
- Has **doPost()** method over HTTP Servlet request and HTTP Servlet response throw exception
- 1st, take a condition "**Is the request is null or not** "
- 2nd, if the request is not null, get the parameters from request
- 3rd, connect to the **index.jsp** file by **id**

**UserLoginServlet.java**

- A class extend HTTP servlet
- 1st, create a constructor and create a new **studentService** object and call **StudentService.java**
- Has **doGet()** method over HTTP Servlet request and HTTP Servlet response throw exception
- 2nd, get the parameters from request
- 3rd , connect to **dashboard.jsp** file

**UserLogoutServlet.java**

- Has **doGet()** method over HTTP Servlet request and HTTP Servlet response throw exception
- Create a new session or gain access to an existing session by using HTTP servlet request **getSession()**

**UserRegisterServlet.java**

- 1st, create a new **studentService** object and call **StudentService.java**
- Has **doPost()** method over HTTP Servlet request and HTTP Servlet response throw exception
- 2nd, send parameters directly to **users**

**UsersServlet.java**

- 1st, create a new **studentService** object and call **StudentService.java**
- Has **doGet()** method over HTTP Servlet request and HTTP Servlet response throw exception
- 2nd, create an **iterator** object to loop through collections by using **hasnext()** and **next()** method of iterator

## 2. Entity

**Course.java**

- Declare variables such as **id, name, department**
- Create **get()** and **set()** for each variables

**Department**

- Declare variables such as **id, name, phone**
- Create **get()** and **set()** for each variables

**Registration.java**

- Declare variables such as **id, course, student, semister, year, status**
- Create **get()** and **set()** for each variables

**Student**

- Declare variables such as **studentId, name, email, password, mobile**
- Create **get()** and **set()** for each variables

## 3. Exception

**DepartmentNotFound.java**

- Extend **RuntimeException** for checking the department is found or not
- Use **super()** to call constructor of super class

**IncorrectPasswordException.java**

- Extend **RuntimeException** for checking the password is correct or not
- Use **super()** to call constructor of super class

**InvalidEmailAddressException.java**

- Extend **RuntimeException** for email validation
- Use **super()** to call constructor of super class

UserNotFoundException.java

- Extend **RuntimeException** for checking user is found or not
- Use **super()** to call constructor of super class

## 4. Filter

**CourseRegistratorFilter.java**

- **CourseRegisterFilter** class implement the **Filter** interface
- Use default method of Filter, **doFilter()** for checking servlet request and response through IO Exception and Servlet Exception
- In doFilter(), check the request and response contain key or not by using **isEmpty()**

**RegisterFilter.java**

- **RegisterFilter** class implement the **Filter** interface
- Use default method of Filter, **doFilter()** for checking servlet request and response through IO Exception and Servlet Exception
- In doFilter(), check the request and response contain key or not by using **isEmpty()** and also match the email format with **match()**

**StudentFilter.java**

- **StudentFilter** class implement the **Filter** interface
- Use default method of Filter, **doFilter()** for checking servlet request and response through IO Exception and Servlet Exception
- In doFilter(), check the request and response contain key or not by using **isEmpty()** and also match the email format with **match()**

## 5. Service

**CourseService.java**

- Create objects call for **CourseRepository()** and **DepartmentRepository()**
- Has method **getCourseByDepartment()** to get course by department id and return the array list
- Also has **getCourse()** and **setCourse()** to get and set course in **CourseRepository**

**DepartmentService.java**

- Create an object call for **DepartmentRepository()**
- Has **getDepartment()** to return department from list

**RegistrationService.java**

- Create object calls for **CourseRepository(), StudentRepository(), RegistrationRepository()**
- Has **save()** to save registration information and return information by Student Id and registration Id to **getById()** and **getRegristrationByStudentId()**

**StudentService.java**

- Create an object call for **StudentRepository()**
- Save the student registration information and return the information to StudentRepository()
- Has **login()** user email and password is correct or not

## 6. WebContent

**confirm.jsp**

- Import **Student.java, RegristrationService.java, Regristration.java, CourseService.java**
- Check if the student's id is null or user request is null then response to the **index.jsp**
- Else get the parameters by registration id for student request session

**dashboard.jsp**

- Import **Course.java,CourseService.java, Department.java, DepartmentService.java, Student.java**
- Check if the user request is null then response to the **index.jsp**
- Else get the request from **getSession()**
- Also check the department id and course id are null return the null value to calling objects

**index.jsp**

- Get the user's request by using **getSession()**
- Send the response directly to **dashboard.java**

**registerList.java**

- Import **Regristration.java, RegristrationService.java, Student.java**
- Check if user's request is null then send the response directly to **index.java**
- Else get the parameter by **getSession()**