RegistrationController.java:

```
package com.unt.registration.controller;
import java.util.List;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.MediaType;
import org.springframework.web.bind.annotation.CrossOrigin;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.bind.annotation.RestController;

import com.unt.registration.util.Course;
import com.unt.registration.util.Department;
import com.unt.registration.util.EnrollObject;
import com.unt.registration.util.Enrollment;
import com.unt.registration.util.SelectCriteria;
import com.unt.registration.service.RegistrationServiceImpl;
import com.unt.registration.util.User;

@CrossOrigin(origins = "http://localhost:4200")
@RestController
@RequestMapping("/RegistrationController")
public class RegistrationController {

	@Autowired
	RegistrationServiceImpl registrationServiceImpl;

	@PostMapping(path = "/login", produces = { MediaType.APPLICATION_JSON_VALUE,
MediaType.APPLICATION_XML_VALUE })
```

```
	public User userValidate(@RequestBody User user) {
		return registrationServiceImpl.userValidate(user.getId(), user.getPassword());
	}

	@PostMapping(path = "/signup", produces = { MediaType.APPLICATION_JSON_VALUE,
MediaType.APPLICATION_XML_VALUE })
	public String signup(@RequestBody User user) {
		return registrationServiceImpl.signup(user);
	}

	@GetMapping(path = "/fetchDeptNames", produces = { MediaType.APPLICATION_JSON_VALUE,
			MediaType.APPLICATION_XML_VALUE })
	public List<Department> fetchAllDepartments() {
		return registrationServiceImpl.fetchAllDepartments();
	}

	@PostMapping(path = "/resetPassword", produces = { MediaType.APPLICATION_JSON_VALUE,
			MediaType.APPLICATION_XML_VALUE })
	public String resetPassword(@RequestBody User user) {
		return registrationServiceImpl.resetPassword(user);
	}

	@PostMapping(path = "/getCourses", produces = { MediaType.APPLICATION_JSON_VALUE,
MediaType.APPLICATION_XML_VALUE })
	public List<Course> getCourses(@RequestBody SelectCriteria selectcriteria) {
		return registrationServiceImpl.getCourses(selectcriteria);
	}

	@PostMapping(path = "/findCourse", produces = { MediaType.APPLICATION_JSON_VALUE,
MediaType.APPLICATION_XML_VALUE })
	public Course findCourse(@RequestBody String courseId) {
		return registrationServiceImpl.findCourse(courseId);
	}
```

```
	@PostMapping(path = "/enroll", produces = { MediaType.APPLICATION_JSON_VALUE,
MediaType.APPLICATION_XML_VALUE })
	public String enroll(@RequestBody EnrollObject enrollObject) {
		return registrationServiceImpl.enroll(enrollObject);
	}

	@PostMapping(path = "/dropCourse", produces = { MediaType.APPLICATION_JSON_VALUE,
MediaType.APPLICATION_XML_VALUE })
	public boolean drop(@RequestBody EnrollObject enrollObject) {
		return registrationServiceImpl.dropCourse(enrollObject);
	}

	@PostMapping(path = "/fetchEnrolledCourses", produces = { MediaType.APPLICATION_JSON_VALUE,
			MediaType.APPLICATION_XML_VALUE })
	public List<Course> fetchEnrolledCourses(@RequestBody User user) {
		return registrationServiceImpl.fetchEnrolledCourses(user);
	}
	@PostMapping(path = "/swap", produces = { MediaType.APPLICATION_JSON_VALUE,
			MediaType.APPLICATION_XML_VALUE })
	public List<Course> swap(@RequestBody User user) {
		return registrationServiceImpl.fetchEnrolledCourses(user);
	}

}
```

RegistrationDao.java

```
package com.unt.registration.dao;

import java.util.List;

import com.unt.registration.util.Course;
import com.unt.registration.util.Department;
import com.unt.registration.util.EnrollObject;
```

```
import com.unt.registration.util.SelectCriteria;
import com.unt.registration.util.User;

public interface RegistrationDao {

	public User userValidate(String id);
	public int userExists(String id);
	public int userIdProvided(String id);
	public Boolean signup(User user);
	public List<Department> fetchAllDepartments();
	public Boolean resetPassword(User user);
	public String getEmail(String id);
	public List<Course> getCourses(SelectCriteria selectCriteria);
	public Course findCourse(String courseId);
	public String enroll(EnrollObject enrollObject);
	public List<Course> fetchEnrolledCourses(User user);
	public boolean dropCourse(EnrollObject enrollObject);
	public boolean decreaseStrength(String id);
	public boolean increaseStrength(String id);
	public int checkPrerequisites(String userId, String courseId);
	public String getPrerequisites(String id);
}
```

RegistrationDaoImpl.java

```
package com.unt.registration.dao;

import java.sql.Types;
import java.util.Calendar;
import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.jdbc.core.BeanPropertyRowMapper;
import org.springframework.jdbc.core.JdbcTemplate;
```

```
import org.springframework.stereotype.Repository;

import com.unt.registration.service.RegistrationServiceImpl;
import com.unt.registration.util.Course;
import com.unt.registration.util.Department;
import com.unt.registration.util.EnrollObject;
import com.unt.registration.util.SelectCriteria;
import com.unt.registration.util.User;

@Repository
public class RegistrationDaoImpl implements RegistrationDao {

	@Autowired
	JdbcTemplate jdbcTemplate;
	RegistrationServiceImpl registrationServiceImpl;

	@Override
	public User userValidate(String id) {
		// TODO Auto-generated method stub
		String sql = "select * from \"Registration DB\".\"Userdetails\" where id=?";
		return jdbcTemplate.queryForObject(sql, new Object[] { id }, new
BeanPropertyRowMapper<>(User.class));
	}

	@Override
	public int userExists(String id) {
		// TODO Auto-generated method stub
		String sql = "SELECT COUNT(*) FROM \"Registration DB\".\"Userdetails\" where id=?";
		return jdbcTemplate.queryForObject(sql, new Object[] { id }, Integer.class);
	}

	@Override
```

```
	public int userIdProvided(String id) {
		// TODO Auto-generated method stub
		String sql = "SELECT COUNT(*) FROM \"Registration DB\".\"Universitydetails\" where id=?";
		return jdbcTemplate.queryForObject(sql, new Object[] { id }, Integer.class);
	}

	@Override
	public Boolean signup(User user) {
		// TODO Auto-generated method stub
		String sql = "INSERT INTO userdetails values(?,?,?,?,?,?,?,?)";
		Object[] args = { user.getId(), user.getFirstName(), user.getLastName(),
user.getEmail().toLowerCase(),
				user.getMobile(), user.getDeptId(), 0, user.getPassword() };
		int[] argTypes = { Types.VARCHAR, Types.VARCHAR, Types.VARCHAR, Types.VARCHAR,
Types.NUMERIC, Types.VARCHAR,
				Types.NUMERIC, Types.VARCHAR };
		if (jdbcTemplate.update(sql, args, argTypes) == 1)
			return true;
		else
			return false;
	}

	@Override
	public List<Department> fetchAllDepartments() {
		// TODO Auto-generated method stub
		String sql = "SELECT * FROM \"Registration DB\".\"Departments\"";
		return jdbcTemplate.query(sql, new BeanPropertyRowMapper<Department>(Department.class));
	}

	@Override
	public String getEmail(String id) {
		String sql = "SELECT email FROM \"Registration DB\".\"Userdetails\" where id=?";
		return jdbcTemplate.queryForObject(sql, new Object[] { id }, String.class);
```

```
	}

	@Override
	public Boolean resetPassword(User user) {
		// TODO Auto-generated method stub
		String sql = "UPDATE \"Registration DB\".\"Userdetails\" SET password=? where id=?";
		Object[] args = { user.getPassword(), user.getId() };
		int[] argTypes = { Types.VARCHAR, Types.VARCHAR };
		if (jdbcTemplate.update(sql, args, argTypes) == 1)
			return true;
		else
			return false;
	}

	public String getCurrentSemester() {
		String currentSemester;
		if ((Calendar.getInstance().get(Calendar.MONTH)) >= 1 &&
(Calendar.getInstance().get(Calendar.MONTH)) <= 4)
			currentSemester = "Spring";
		else if ((Calendar.getInstance().get(Calendar.MONTH)) >= 5 &&
(Calendar.getInstance().get(Calendar.MONTH)) <= 7)
			currentSemester = "Summer";
		else
			currentSemester = "Fall";
		return currentSemester;
	}

	@Override
	public List<Course> getCourses(SelectCriteria selectCriteria) {
		// TODO Auto-generated method stub

		String sql = "SELECT * FROM \"Registration DB\".\"Courses\" where degree=? and \"deptId\"=?
and \"isActive\"=? and semester=? and year=?";
```

```
		Object[] args = { selectCriteria.getDegree(), selectCriteria.getDeptId(), true,
this.getCurrentSemester(),
				Calendar.getInstance().get(Calendar.YEAR) };
		return jdbcTemplate.query(sql, args, new BeanPropertyRowMapper<Course>(Course.class));
	}

	@Override
	public Course findCourse(String courseId) {
		// TODO Auto-generated method stub
		String sql = "SELECT * FROM \"Registration DB\".\"Courses\" where \"courseId\"=? and
semester=? and year=?";
		return jdbcTemplate.queryForObject(sql,
				new Object[] { courseId, this.getCurrentSemester(),
Calendar.getInstance().get(Calendar.YEAR) },
				new BeanPropertyRowMapper<>(Course.class));
	}

	public List<Course> fetchEnrolledCourses(User user) {
		String sql = "select * from \"Registration DB\".\"Enrollments\" JOIN \"Registration
DB\".\"Courses\" ON \"Registration DB\".\"Enrollments\".\"courseId\"=\"Registration DB\".\"Courses\".\"courseId\"
where \"Registration DB\".\"Enrollments\".id=?";
		Object[] args = { user.getId() };
		return jdbcTemplate.query(sql, args, new BeanPropertyRowMapper<Course>(Course.class));
	}

	@Override
	public boolean dropCourse(EnrollObject enrollObject) {
		// TODO Auto-generated method stub
		String sql = "DELETE FROM \"Registration DB\".\"Enrollments\" WHERE id = '" +
enrollObject.getUserId()
				+ "' AND \"courseId\" = '" + enrollObject.getCourseId() + "' and semester='" +
this.getCurrentSemester()
				+ "'AND year=" + Calendar.getInstance().get(Calendar.YEAR);
		int rows = jdbcTemplate.update(sql);
		boolean strength = decreaseStrength(enrollObject.getCourseId());
```

```
		if (rows > 0 && strength) {
			return true;

		} else
			return false;
	}

	@Override
	public boolean decreaseStrength(String id) {
		String sql = "update \"Registration DB\".\"Courses\" set strength = strength-1 where \"courseId\"='" + id
				+ "' and semester='" + this.getCurrentSemester() + "'AND year="
				+ Calendar.getInstance().get(Calendar.YEAR);
		int rows = jdbcTemplate.update(sql);
		if (rows > 0)
			return true;
		else
			return false;

	}

	public int checkStrength(String courseId) {
		String sql = "SELECT count(*)   FROM \"Registration DB\".\"Courses\" where \"Registration DB\".\"Courses\".\"Strength\"  < \"Registration DB\".\"Courses\".\"CourseMaxStrength\" AND \"Registration DB\".\"Courses\".\"CourseID\"='"
				+ courseId + "'";
		return jdbcTemplate.queryForObject(sql, Integer.class);
	}

	@Override
	public boolean increaseStrength(String id) {
		String sql = "update \"Registration DB\".\"Courses\" set strength = strength+1 where \"courseID\"='" + id + "'";
		int rows = jdbcTemplate.update(sql);
```

```
		if (rows > 0)
			return true;
		else
			return false;

	}

	@Override
	public String enroll(EnrollObject enrollObject) {
		// TODO Auto-generated method stub
		int i;
		String id = getPrerequisites(enrollObject.getCourseId());
		if (id != "empty") {
			i = checkPrerequisites(enrollObject.getUserId(), id);
		} else
			i = 1;
		if (i > 0) {
			int checkStrength = checkStrength(enrollObject.getCourseId());
			if (checkStrength > 0) {
				String sql = "INSERT INTO  \"Registration DB\".\"Enrollments\" values(?,?,?,?,?)";
				Object[] args = { enrollObject.getUserId(), enrollObject.getCourseId(),
this.getCurrentSemester(),
						Calendar.getInstance().get(Calendar.YEAR), null };
				int[] argTypes = { Types.VARCHAR, Types.VARCHAR, Types.VARCHAR,
Types.NUMERIC, Types.CHAR };
				if (jdbcTemplate.update(sql, args, argTypes) == 1) {
					increaseStrength(enrollObject.getUserId());
					return "Enrolled";
				} else
					return "NotEnrolled";
			}

			else
```

```
                                        return "StrengthFull";
                        } else


                                return "NoPrerequisites";
        }


        @Override
        public int checkPrerequisites(String userId, String courseId) {
//                      String id=getPrerequisites(courseId);
                        String sql = "SELECT COUNT(*) FROM \"Registration DB\".\"Enrollments\" where id='" + userId
                                        + "' AND \"courseId\"='" + courseId + "' AND grade!=null";
                        return jdbcTemplate.queryForObject(sql, Integer.class);


        }


        @Override
        public String getPrerequisites(String id) {
                        // TODO Auto-generated method stub
                        String sql = "SELECT \"prerequisiteId\" FROM \"Registration DB\".\"Prerequisites\" where
\"courseId\"='" + id
                                        + "'";
                        if (jdbcTemplate.queryForObject(sql, String.class) != null)
                                return jdbcTemplate.queryForObject(sql, String.class);
                        else
                                return "empty";


        }


}
```

RegistrationServiceImpl.java

```
package com.unt.registration.service;

import java.util.List;
```

```
import javax.mail.internet.MimeMessage;
import javax.swing.JOptionPane;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.mail.javamail.JavaMailSender;
import org.springframework.mail.javamail.MimeMessageHelper;
import org.springframework.stereotype.Service;

import com.unt.registration.dao.RegistrationDaoImpl;
import com.unt.registration.util.Course;
import com.unt.registration.util.Department;
import com.unt.registration.util.EnrollObject;
import com.unt.registration.util.SelectCriteria;
import com.unt.registration.util.User;

@Service
public class RegistrationServiceImpl implements RegistrationService{

	@Autowired
	RegistrationDaoImpl registrationDaoImpl;
	@Autowired
	JavaMailSender javaMailSender;


	@Override
	public User userValidate(String id, String password) {
		// TODO Auto-generated method stub
		User user = new User();
		if (registrationDaoImpl.userExists(id)== 0)
		{
			user.setId(id);
			return user;
		} else {
			user = registrationDaoImpl.userValidate(id);
			user.setUserExists(true);
			if (password.equals(user.getPassword()))
				user.setValidUser(true);
			return user;
		}
	}

	@Override
	public String signup(User user) {
		if(registrationDaoImpl.userIdProvided(user.getId())!=0) {
			if(registrationDaoImpl.userExists(user.getId())==0) {
				if(registrationDaoImpl.signup(user)==true)
					return "signed up";
				else
					return "Unexpected Error";
			}
			else
				return "Already signed up";
		}
```

```
		else
			return "Invalid user";
	}

	@Override
	public List<Department> fetchAllDepartments() {
		// TODO Auto-generated method stub
		return registrationDaoImpl.fetchAllDepartments();
	}

	@Override
	public String resetPassword(User user) {
		// TODO Auto-generated method stub
		if(registrationDaoImpl.userExists(user.getId())!=0) {
			if(user.getEmail().equals(registrationDaoImpl.getEmail(user.getId())))
			{
				if(registrationDaoImpl.resetPassword(user)==true)
					return "Password reset";
				else
					return "Unexpected Error";
			}
			else
				return "invalid email";


		}
		else
		return "Invalid user ID";
	}

	@Override
	public List<Course> getCourses(SelectCriteria selectCriteria) {
		// TODO Auto-generated method stub
		return registrationDaoImpl.getCourses(selectCriteria);
	}

	@Override
	public Course findCourse(String courseId) {
		// TODO Auto-generated method stub
		return registrationDaoImpl.findCourse(courseId);
	}

	@Override
	public String enroll(EnrollObject enrollObject) {
		// TODO Auto-generated method stub
		return registrationDaoImpl.enroll(enrollObject);
	}

	@Override
	public List<Course> fetchEnrolledCourses(User user) {
		// TODO Auto-generated method stub
		return registrationDaoImpl.fetchEnrolledCourses(user);
	}
```

```
	@Override
	public boolean dropCourse(EnrollObject enrollObject) {
		// TODO Auto-generated method stub

		if(registrationDaoImpl.dropCourse(enrollObject))
		{
			sendEmail(enrollObject);
			return true;
		}
		return false;


	}
	@Override
	public boolean sendEmail(EnrollObject enrollObject) {

		 try {


			MimeMessage msg = javaMailSender.createMimeMessage();


		    MimeMessageHelper helper = new MimeMessageHelper(msg, true);

		    helper.setTo("sreekanthvobilishetty@gmail.com");

		    helper.setSubject("Notification for enrolling");
		    String s= enrollObject.getCourseId();


				helper.setText("<h2>Dropped Course  "+s +"Successfully!</h2>", true);



		    javaMailSender.send(msg);
		    return true;

		  } catch (Exception e) {
		    JOptionPane.showMessageDialog(null, e);
		    return false;
		  }
			}

	}
```

Student_index.html

```
<html>

  <head>

    <title> Home Page </title>

    <link rel="stylesheet" type="text/css" href="menu.css" />
```

```
</head>
<body>
  <div class="container">
    <div id="menu">
      <ul>
        <li><a href="#">Search</a></li>
        <li><a href="#">Enroll</a>
          <ul>
            <li><a href="/add">Add</a></li>
            <li><a href="Drop_course.html">Drop</a></li>
            <li><a href="#">Swap</a></li>
                                                    <li><a href="Course_curriculum.html">Course
Catalog</a>

            </li>
          </ul>
        </li>
        <li><a href="#">My Classes</a> <ul>
                                          <li><a href="#">Class Schedule
            <li><a href="#">View Grades</a></li>
            <li><a href="#">Mandatory Courses</a></li>

                                                    </ul>

            </li>
        <li><a href="#">Finance</a>
                                                <ul>
            <li><a href="#">Pay Bill</a></li>
            <li><a href="#">Past Payments</a></li>

          </ul>
                                                </li>
```

```
            <li><a href="Contact_us.html">Contact Us</a></li>
                                                    <li class="logout"><a href="/logout">Logout</a></li>
        </ul>
                                        <div>




                                        </div>
      </div>
    </div>
  </body>
</html>
```