Table of Contents

Introduction

This mini project's goal is to create a dynamic and responsive website for a student information management system (SIMS). This web application is a full-fledged logic application that uses latest technology in the market to provide full functionality, efficiency, and highest quality.

Development Environment and Programming Tools

Development Environment

The development environment we used to develop the mini system are: -

- **Eclipse IDE 2021-03: -** we prefer this IDE since It is the fastest Java IDE and This IDE will help us to create a dynamic web application.
- **Apache Tomcat 9.0: -** we prefer this server because it is open-source, and it's free to use. Easy to install and configure. Multiple applications can run at the same time without any issues.

Programming Language

- **Java Programming Language**
  We will use Java Server Page (JSP) and Servlet to implement our web application.
- **Hibernate Query Language (HQL)**
  Hibernate Query Language (HQL) is an object-oriented query language, similar to SQL, but instead of operating on tables and columns, HQL works with persistent objects and their properties. HQL queries are translated by Hibernate into conventional SQL queries, which in turns perform action on database.

Markup languages

- **HTML and CSS**
  we will use HTML (Hypertext markup language) to build the structure of our website and CSS to style our website.

Framework and Library

- Bootstrap
- mysql-connector-java-8.0.25
- antlr-2.7.7
- byte-buddy-1.10.22
- classmate-1.5.1
- FastInfoset-1.2.15
- hibernate-commons-annotations-5.1.2. Final
- hibernate-core-5.5.0. Final
- istack-commons-runtime-3.0.7
- jandex-2.2.3. Final
- javassist-3.27.0-GA
- javax.activation-api-1.2.0
- javax.persistence-api-2.2
- jaxb-api-2.3.1
- jaxb-runtime-2.3.1
- jboss-logging-3.4.2.Final
- jboss-transaction-api_1.2_spec-1.1.1.Final
- stax-ex-1.8
- txw2-2.3.1

System Requirement

On client-side

- Operating System (Any)
- Web Browser (IE, Mozilla Firefox, Google Chrome, Safari, Opera …)

On server-side

- Web Server (Apache Tomcat)
- Hibernate
- MySQL

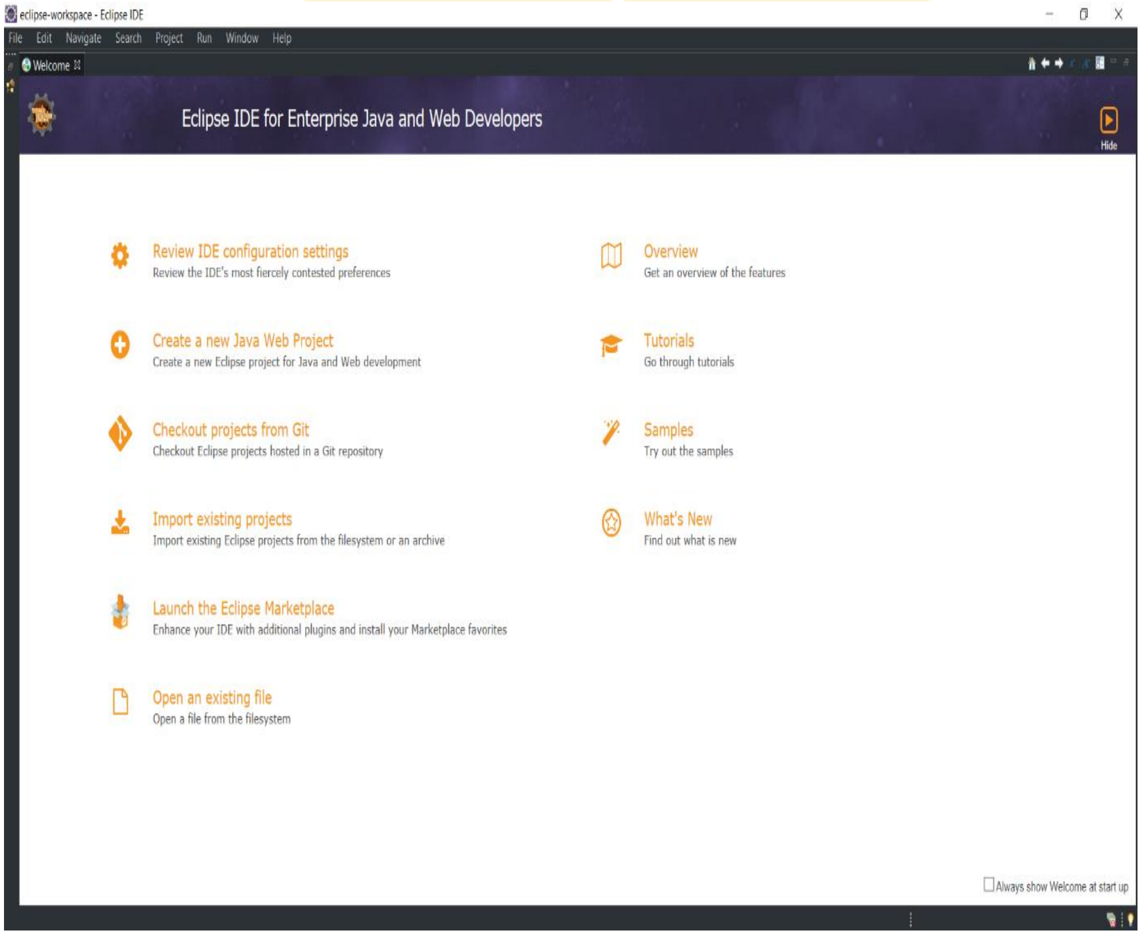
Features

- Login
- Create an Account
- Register Student.
- Display List of Student.
- Search Student
- Add Department
- Logout

Screenshots

Steps to create dynamic web project in eclipse

Step 1: Launch Eclipse IDE for Enterprise Java and Web Developers.


eclipse-workspace - Eclipse IDE
File Edit Navigate Search Project Run Window Help
Welcome
Hide
Always show Welcome at start up

Eclipse IDE for Enterprise Java and Web Developers

Review IDE configuration settings
Review the IDE's most fiercely contested preferences

Create a new Java Web Project
Create a new Eclipse project for Java and Web development

Checkout projects from Git
Checkout Eclipse projects hosted in a Git repository

Import existing projects
Import existing Eclipse projects from the filesystem or an archive

Launch the Eclipse Marketplace
Enhance your IDE with additional plugins and install your Marketplace favorites

Open an existing file
Open a file from the filesystem

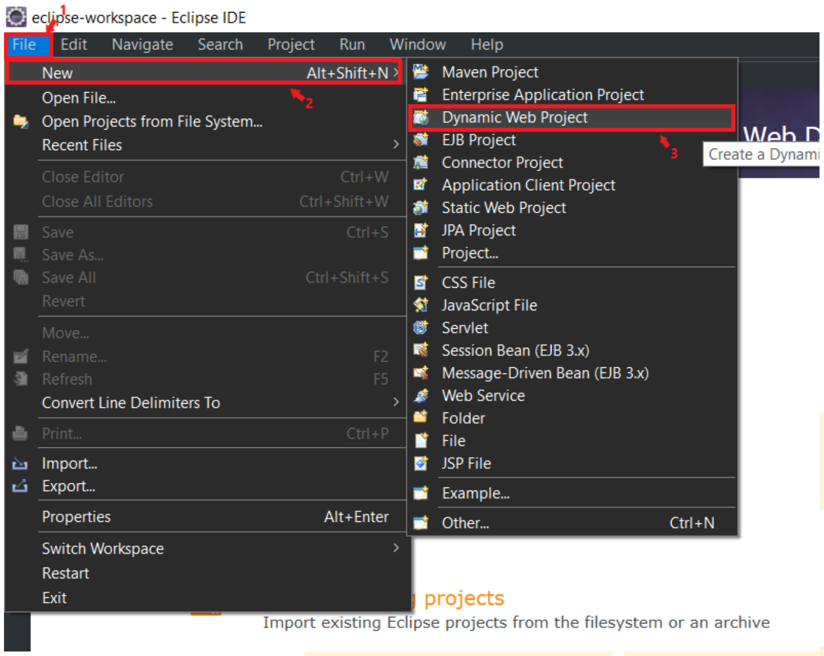
Overview
Get an overview of the features

Tutorials
Go through tutorials

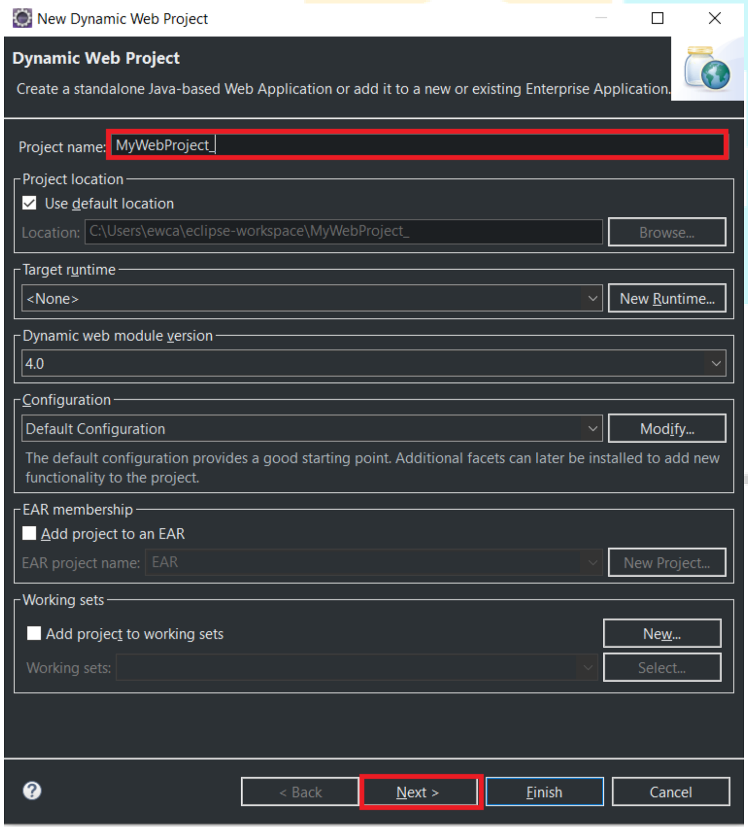
Samples
Try out the samples

What's New
Find out what is new

Step 2: Open Dynamic Web Project, Click File > New > Dynamic Web Project as shown in the figure.



Step: 3 Name the project e.g. MyWebProject.


New Dynamic Web Project

Dynamic Web Project

Create a standalone Java-based Web Application or add it to a new or existing Enterprise Application.

Project name: MyWebProject_

Project location
- [✓] Use default location
- Location: C:\Users\ewca\eclipse-workspace\MyWebProject_

Target runtime: <None>

Dynamic web module version: 4.0

Configuration: Default Configuration

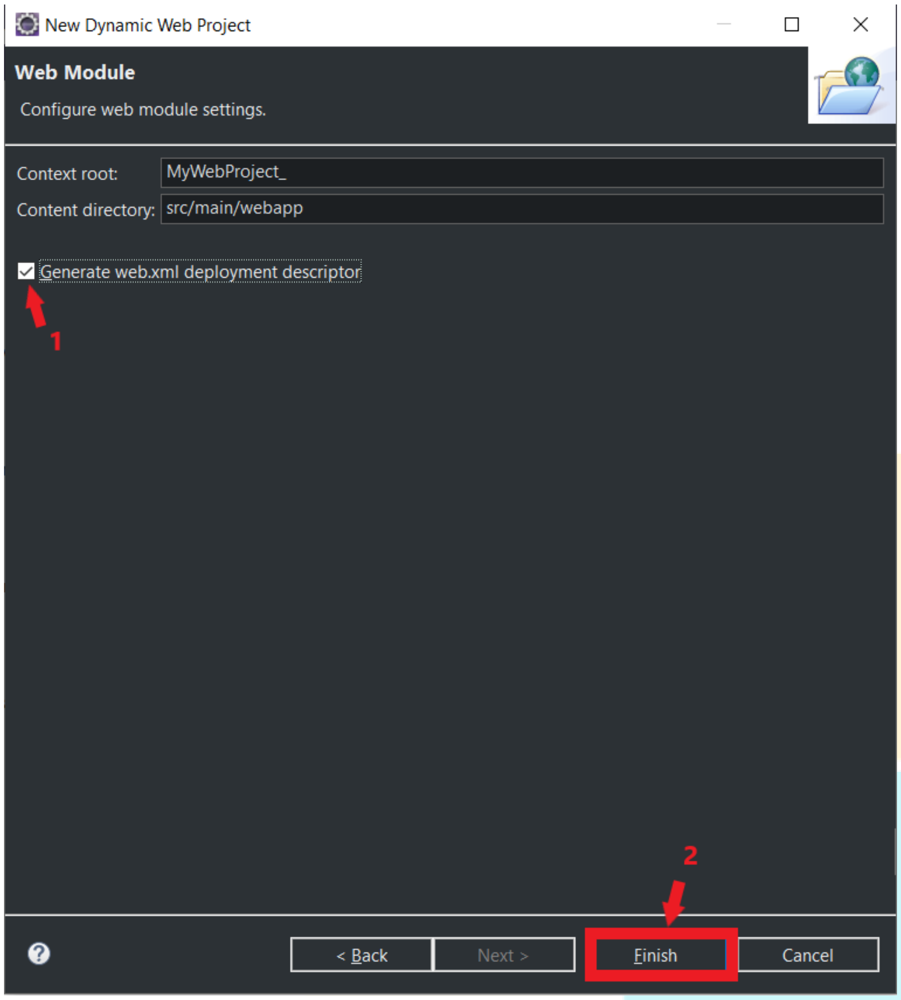
The default configuration provides a good starting point. Additional facets can later be installed to add new functionality to the project.

EAR membership
- [ ] Add project to an EAR
- EAR project name: EAR

Working sets
- [ ] Add project to working sets
- Working sets:

Then click “Next”.

After that select "Generate web.xml deployment descriptor" and click "Finish".



New Dynamic Web Project

Web Module

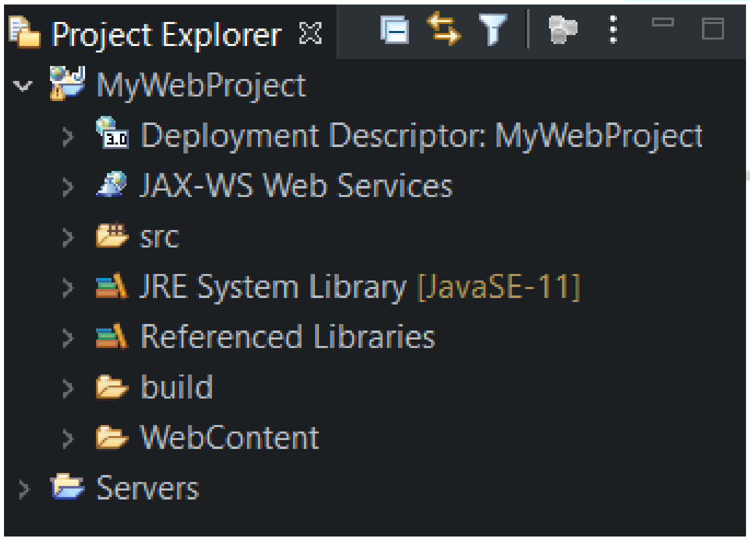
Configure web module settings.

Context root: MyWebProject_

Content directory: src/main/webapp

☑ Generate web.xml deployment descriptor

After that you will see something like this in the project explorer tab.

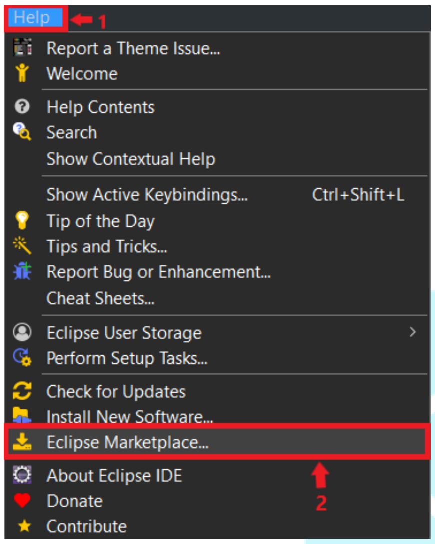


Project Explorer

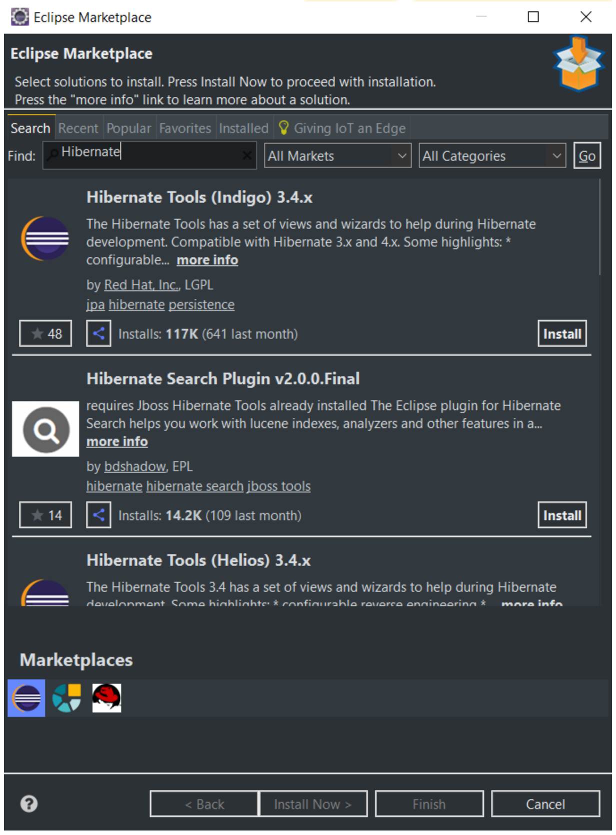
- MyWebProject
  - Deployment Descriptor: MyWebProject
  - JAX-WS Web Services
  - src
  - JRE System Library [JavaSE-11]
  - Referenced Libraries
  - build
  - WebContent
- Servers

Steps to install hibernate tools eclipse plugin

Step 1: Go to "Eclipse Marketplace" from the Help Menu, as shown in below image.



Step 2: Use the search option to find the "Hibernate Tools" plugin, hibernate plugins are Eclipse version specific. So make sure you choose the same one as your eclipse. You can find Eclipse version from "About Eclipse" popup page.


Eclipse Marketplace

Eclipse Marketplace

Select solutions to install. Press Install Now to proceed with installation.
Press the "more info" link to learn more about a solution.

Search | Recent | Popular | Favorites | Installed | Giving IoT an Edge

Find: Hibernate | All Markets | All Categories | Go

Hibernate Tools (Indigo) 3.4.x
The Hibernate Tools has a set of views and wizards to help during Hibernate development. Compatible with Hibernate 3.x and 4.x. Some highlights: * configurable... more info
by Red Hat, Inc., LGPL
jpa hibernate persistence
48 Installs: 117K (641 last month) Install

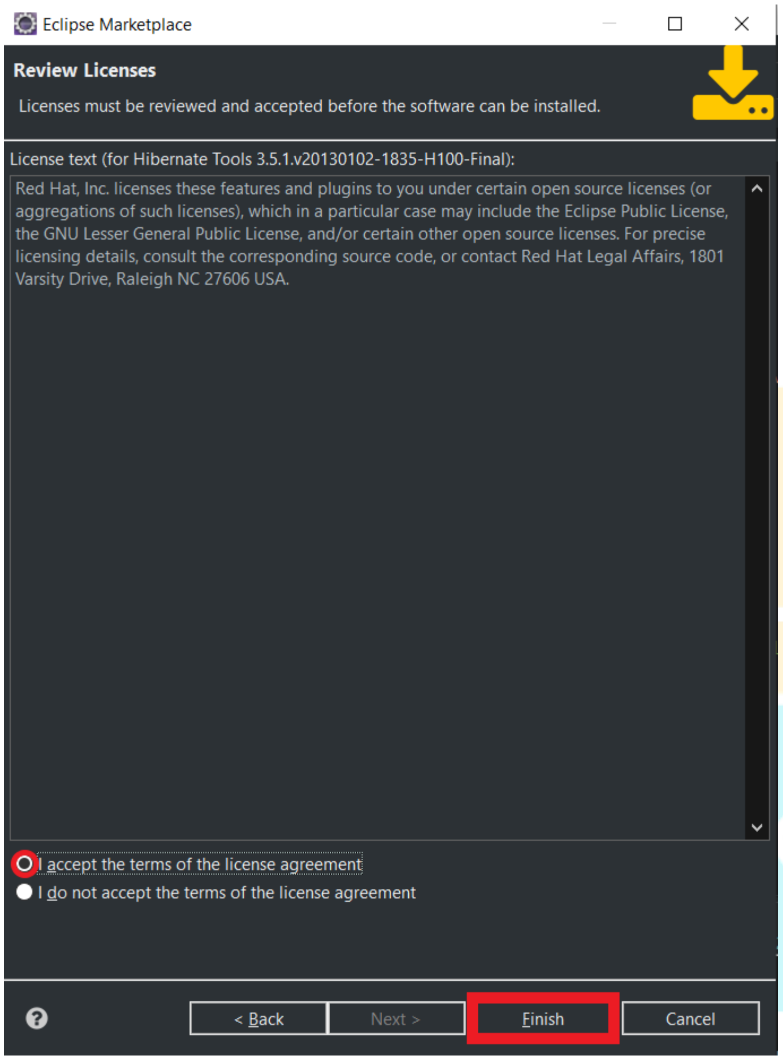
Hibernate Search Plugin v2.0.0.Final
requires Jboss Hibernate Tools already installed The Eclipse plugin for Hibernate Search helps you work with lucene indexes, analyzers and other features in a... more info
by bdshadow, EPL
hibernate hibernate search jboss tools
14 Installs: 14.2K (109 last month) Install

Hibernate Tools (Helios) 3.4.x
The Hibernate Tools 3.4 has a set of views and wizards to help during Hibernate development. Some highlights: * configurable reverse engineering * ... more info

Marketplaces

< Back | Install Now > | Finish | Cancel

Step 3: Click on the install button, make sure the "I accept the terms of the license agreement" radio box is selected and click on "Finish" button.



Licenses must be reviewed and accepted before the software can be installed.

License text (for Hibernate Tools 3.5.1.v20130102-1835-H100-Final):

Red Hat, Inc. licenses these features and plugins to you under certain open source licenses (or aggregations of such licenses), which in a particular case may include the Eclipse Public License, the GNU Lesser General Public License, and/or certain other open source licenses. For precise licensing details, consult the corresponding source code, or contact Red Hat Legal Affairs, 1801 Varsity Drive, Raleigh NC 27606 USA.

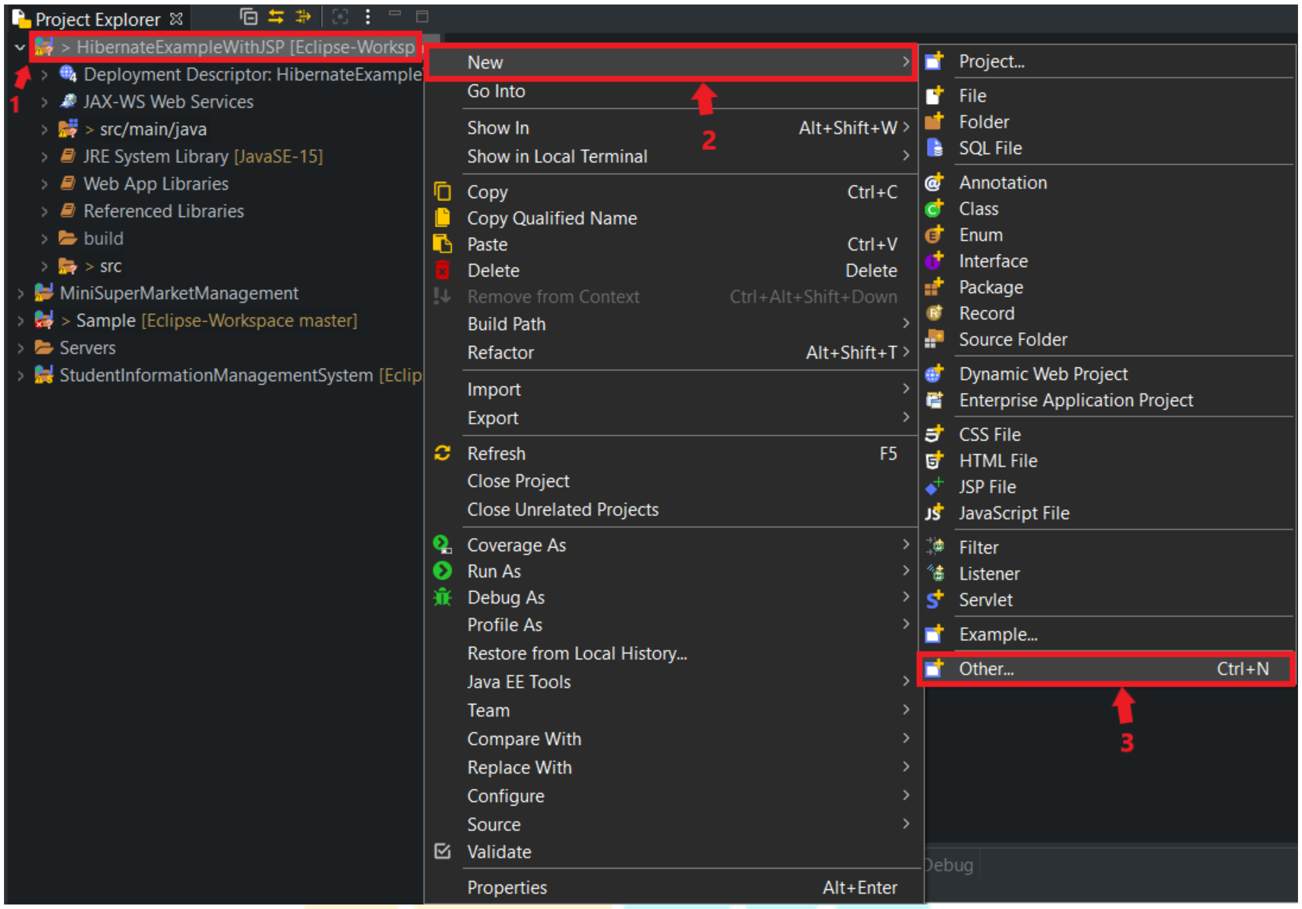
[✓ I accept the terms of the license agreement]
I do not accept the terms of the license agreement

Once the plugin is installed, it will ask to restart the Eclipse. Just restart it and you are ready to use it for hibernate projects.

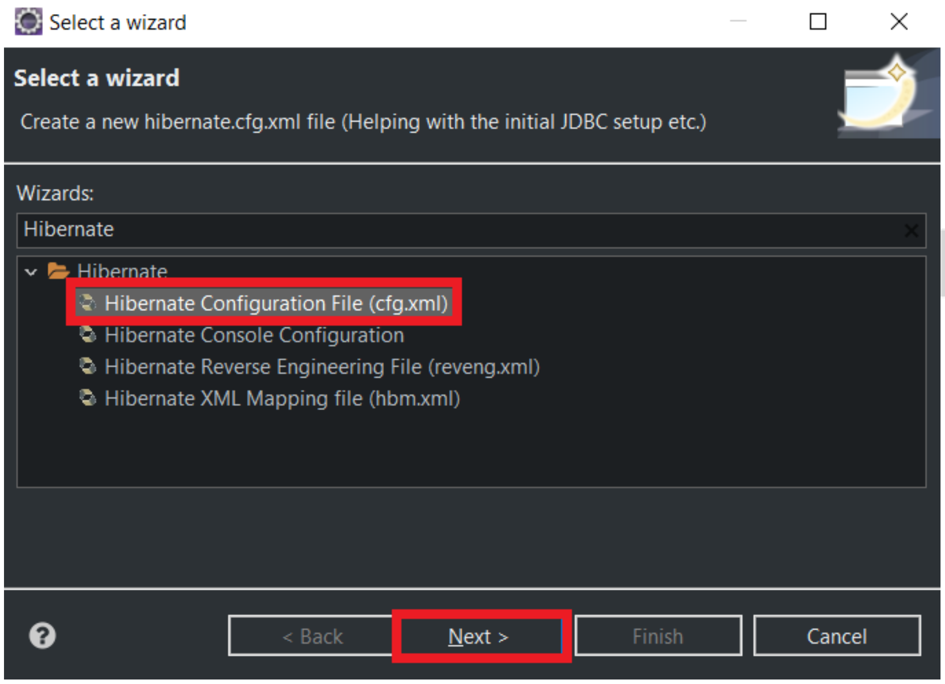
Steps to create hibernate configuration file using hibernate tools eclipse plugin

Now that hibernate tools plugin is installed, let's see how to create hibernate configuration file using it.

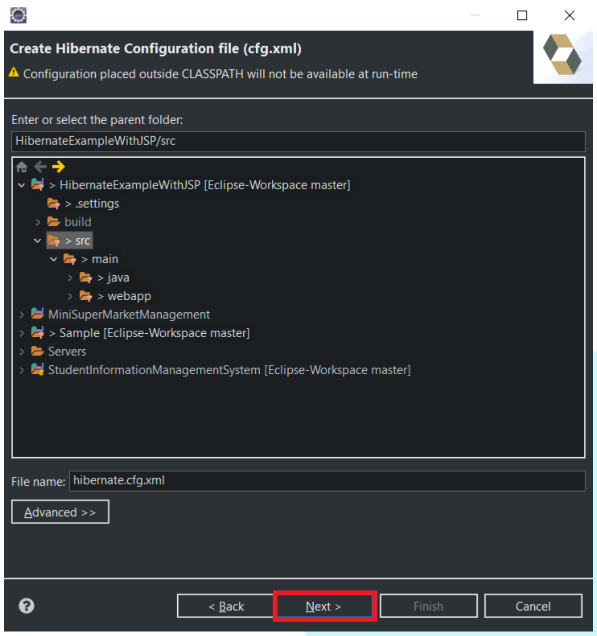
Step 1: Right click on the project and go to "New" > "Other" or you can open this window using keyboard Command+N (Mac), Ctrl+N (Windows).



In the popup window, select “Hibernate Configuration File” as shown in below image.


Select a wizard
Select a wizard
Create a new hibernate.cfg.xml file (Helping with the initial JDBC setup etc.)
Wizards:
Hibernate
Hibernate
Hibernate Configuration File (cfg.xml)
Hibernate Console Configuration
Hibernate Reverse Engineering File (reveng.xml)
Hibernate XML Mapping file (hbm.xml)
< Back
Next >
Finish
Cancel

Step 2: In the next window, you can select the parent folder and set the file name, as shown below.



Create Hibernate Configuration file (cfg.xml)

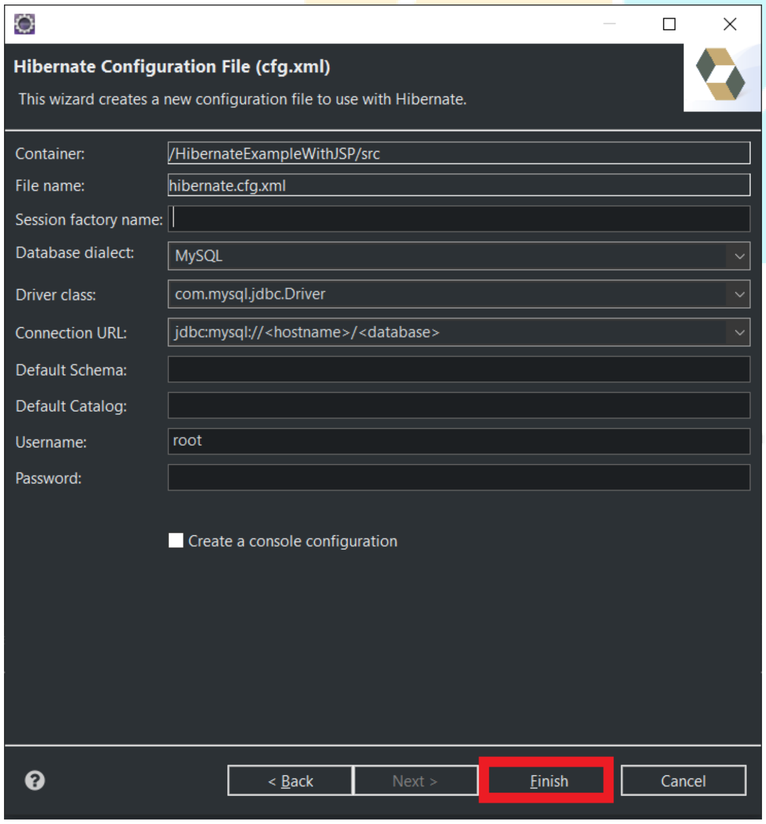
⚠ Configuration placed outside CLASSPATH will not be available at run-time

Enter or select the parent folder: HibernateExampleWithJSP/src

```
> HibernateExampleWithJSP [Eclipse-Workspace master]
    > .settings
    build
    > src
        > main
            > java
            > webapp
MiniSuperMarketManagement
> Sample [Eclipse-Workspace master]
Servers
StudentInformationManagementSystem [Eclipse-Workspace master]
```

File name: hibernate.cfg.xml

Step 3: The next window lets you set the database properties such as dialect, database user, password and connection URLs.



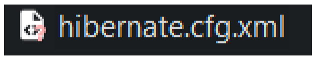
Hibernate Configuration File (cfg.xml)

This wizard creates a new configuration file to use with Hibernate.

Container:	/HibernateExampleWithJSP/src
File name:	hibernate.cfg.xml
Session factory name:	
Database dialect:	MySQL
Driver class:	com.mysql.jdbc.Driver
Connection URL:	jdbc:mysql://<hostname>/<database>
Default Schema:	
Default Catalog:	
Username:	root
Password:	

☐ Create a console configuration

Step 4: When you click “Finish” button, hibernate.cfg.xml file will get added in the parent folder where your java src is present.



hibernate.cfg.xml

For more tutorial Click Here.

UI (User Interface)

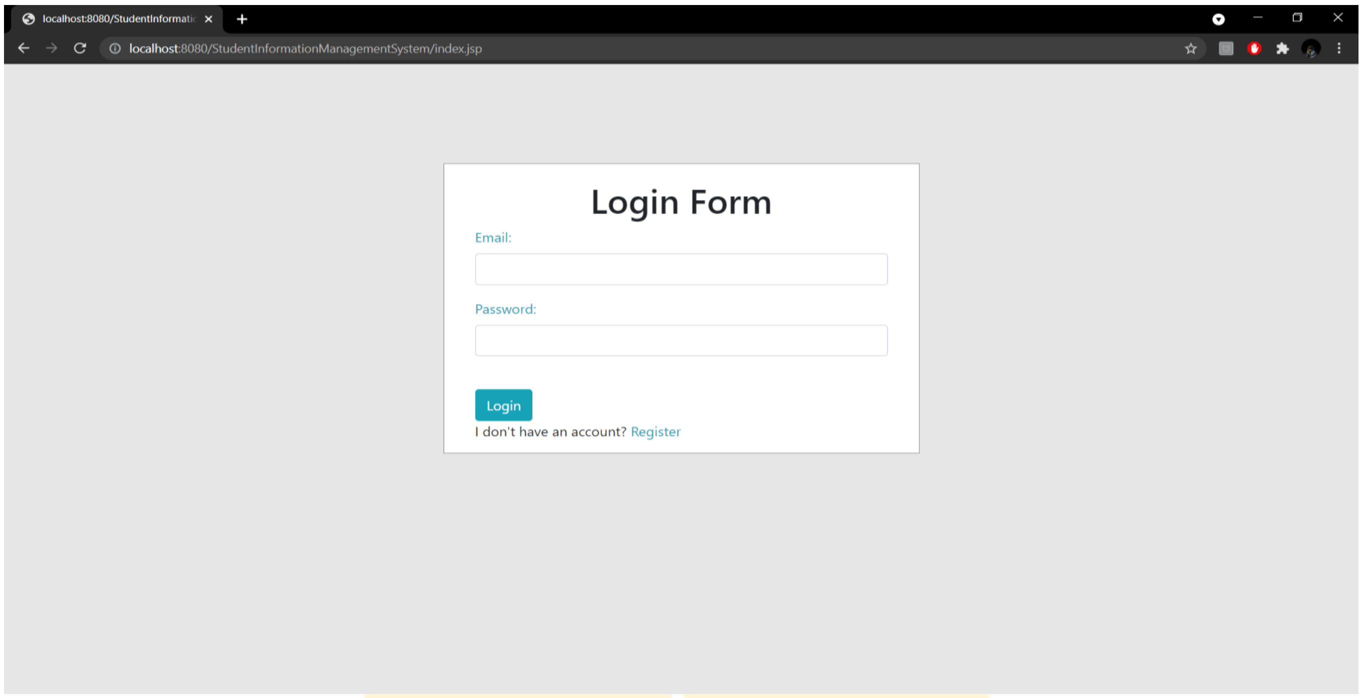

localhost:8080/StudentInformati
localhost:8080/StudentInformationManagementSystem/index.jsp
Login Form
Email:
Password:
Login
I don't have an account? Register

Figure 1: Login Page

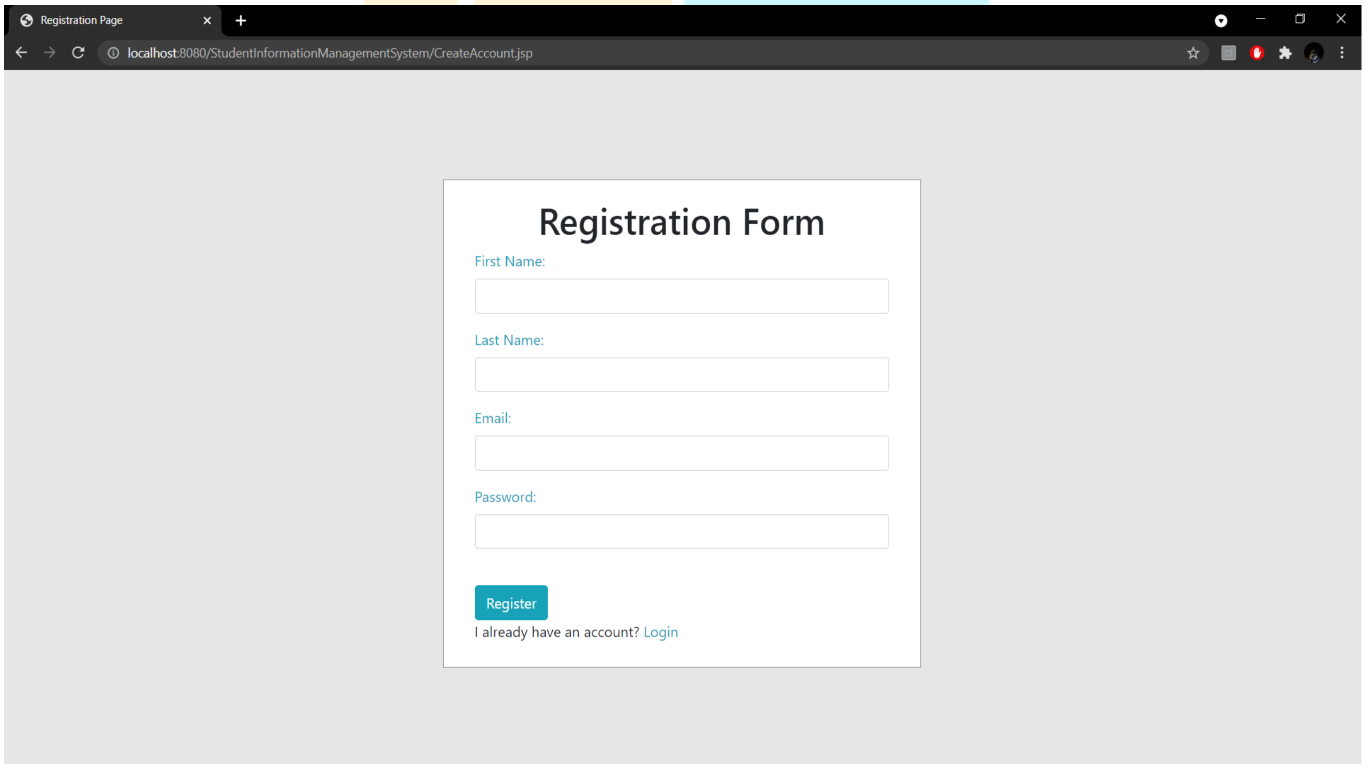

Registration Page
localhost:8080/StudentInformationManagementSystem/CreateAccount.jsp
Registration Form
First Name:
Last Name:
Email:
Password:
Register
I already have an account? Login

Figure 2: Registration Page

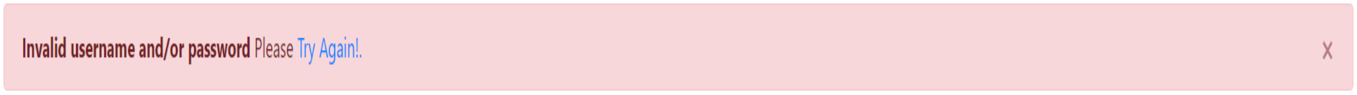
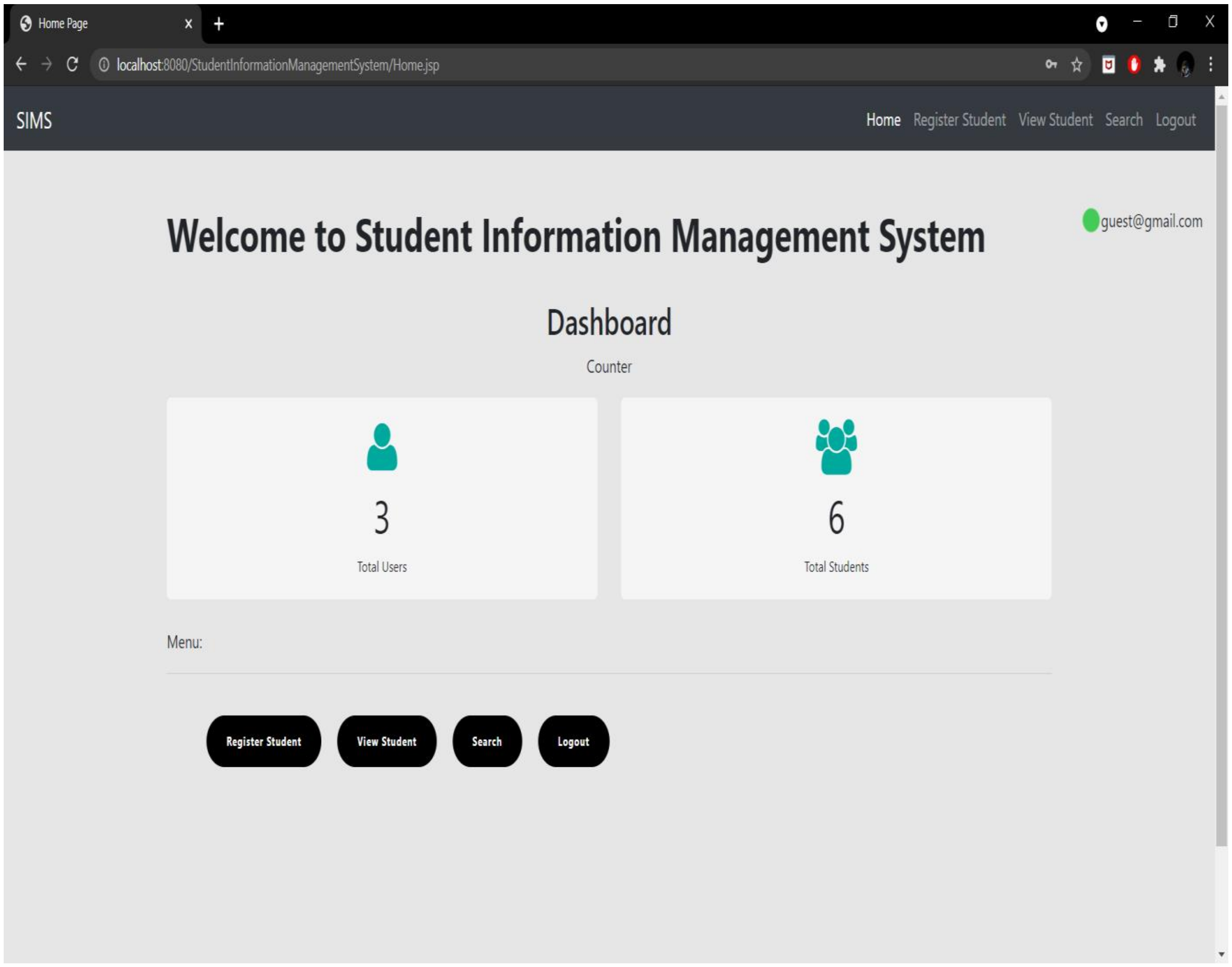

Invalid username and/or password Please Try Again!.

Figure 3: Error Page



Welcome to Student Information Management System

Dashboard

Counter

3

Total Users

6

Total Students

Menu:

Figure 4: Home Page

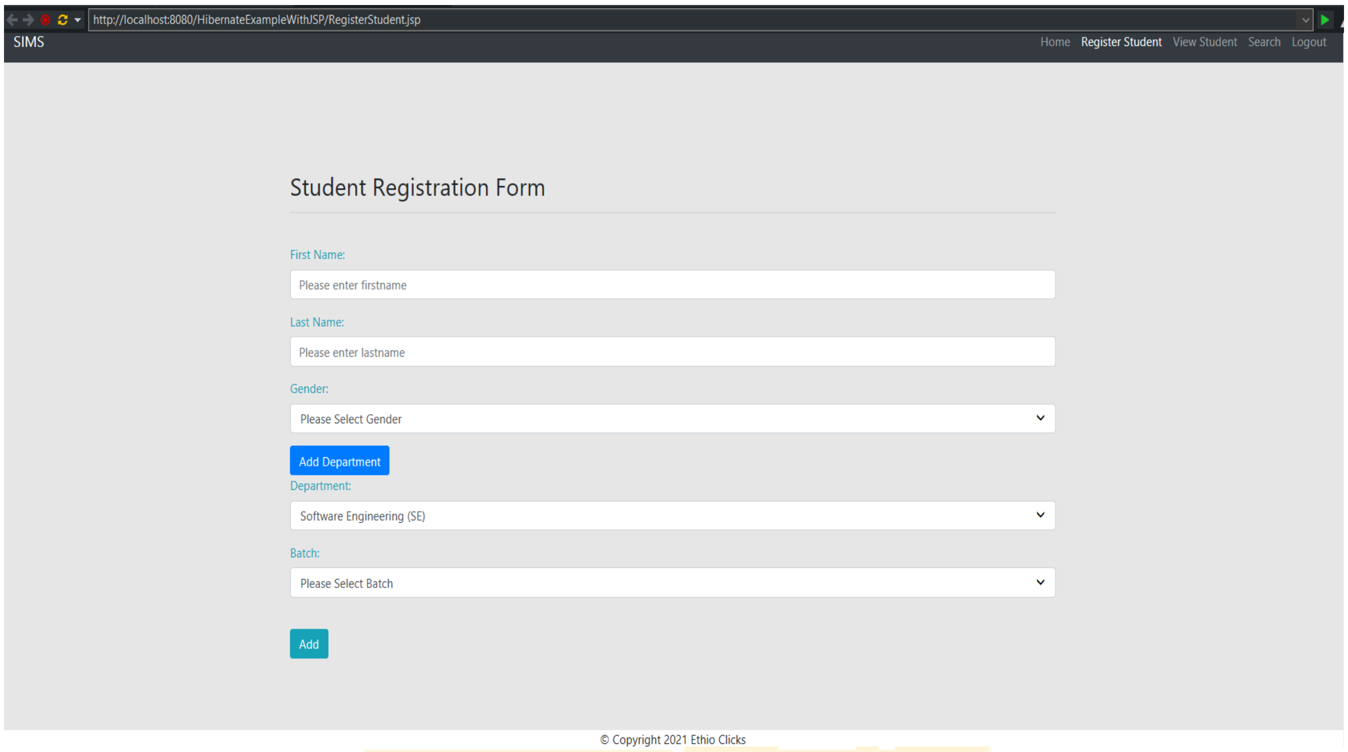


Figure 5: Register Student Page

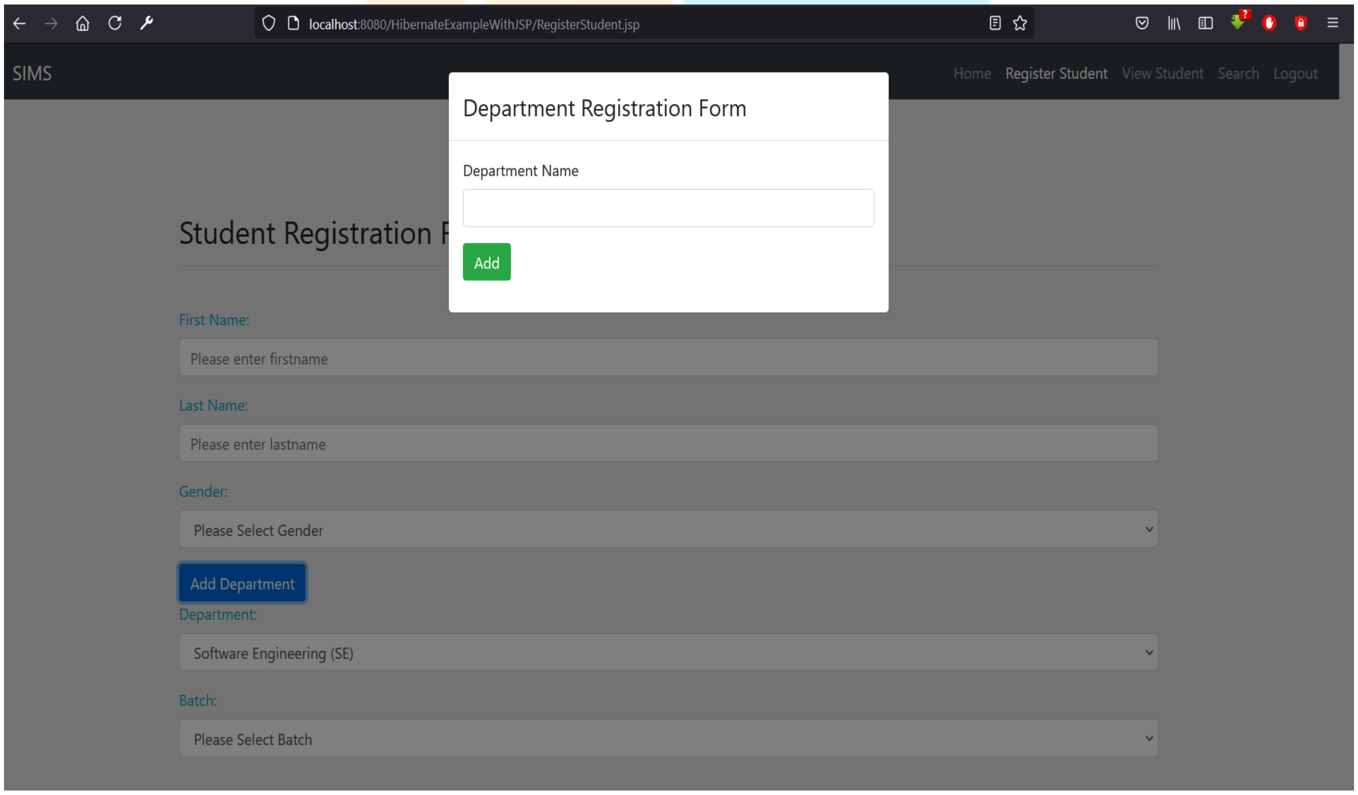
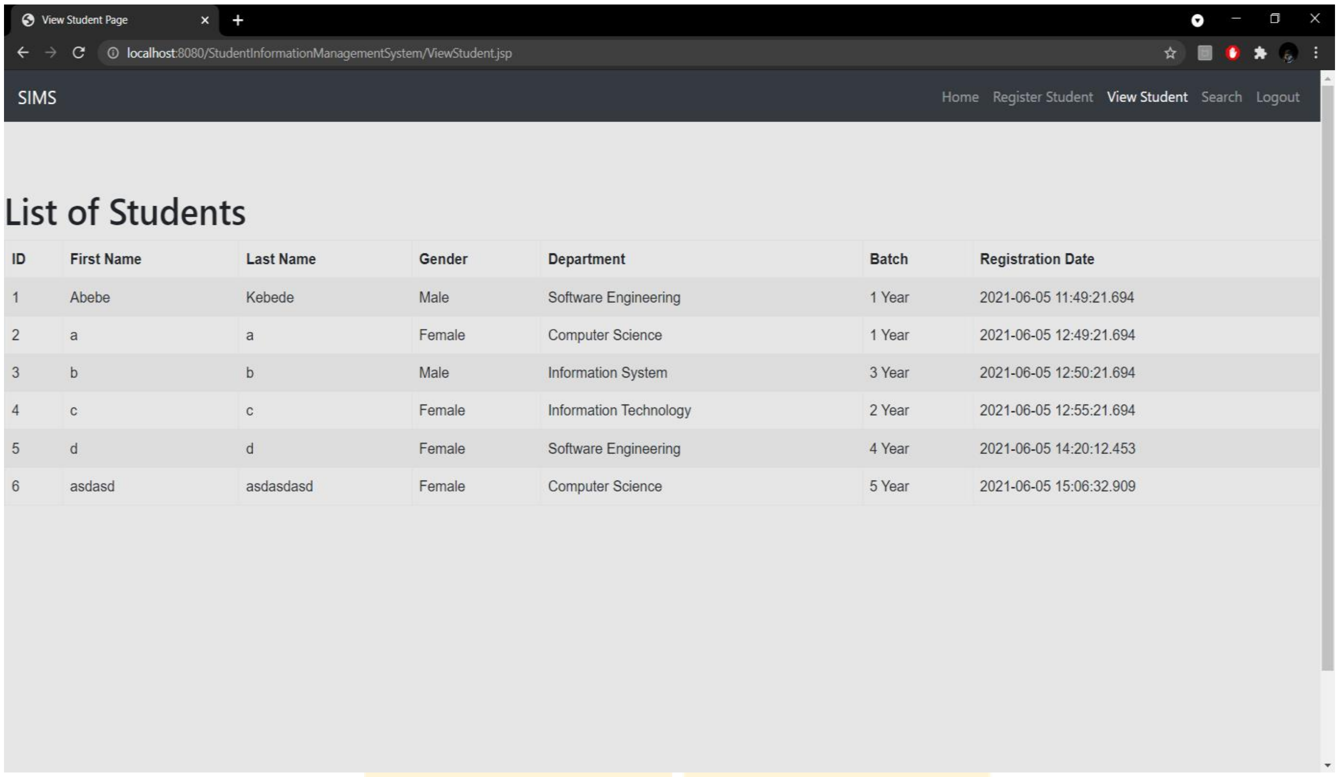


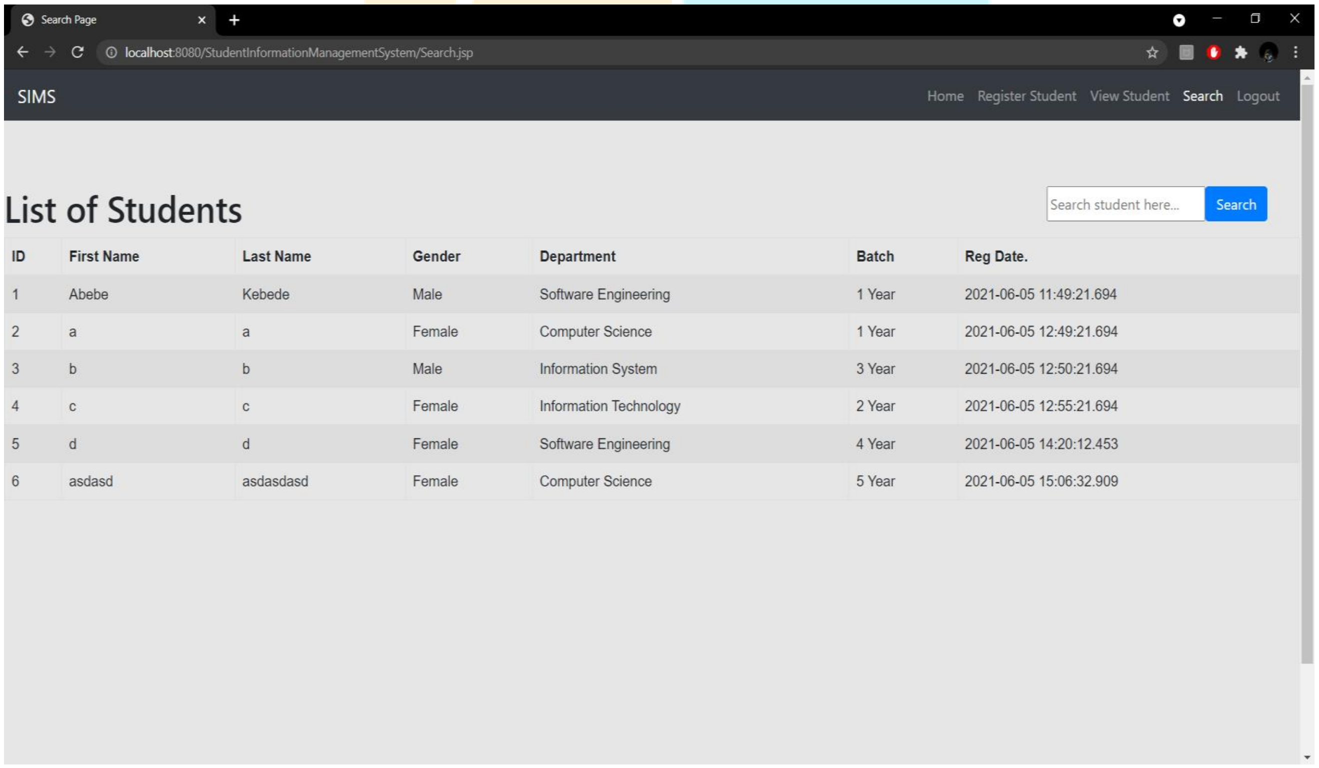
Figure 7: Add Department


View Student Page
localhost:8080/StudentInformationManagementSystem/ViewStudent.jsp
SIMS
Home Register Student View Student Search Logout

List of Students

ID	First Name	Last Name	Gender	Department	Batch	Registration Date
1	Abebe	Kebede	Male	Software Engineering	1 Year	2021-06-05 11:49:21.694
2	a	a	Female	Computer Science	1 Year	2021-06-05 12:49:21.694
3	b	b	Male	Information System	3 Year	2021-06-05 12:50:21.694
4	c	c	Female	Information Technology	2 Year	2021-06-05 12:55:21.694
5	d	d	Female	Software Engineering	4 Year	2021-06-05 14:20:12.453
6	asdasd	asdasdasd	Female	Computer Science	5 Year	2021-06-05 15:06:32.909

Figure 8: Display list of registered student.


Search Page
localhost:8080/StudentInformationManagementSystem/Search.jsp
SIMS
Home Register Student View Student Search Logout
Search student here...
Search

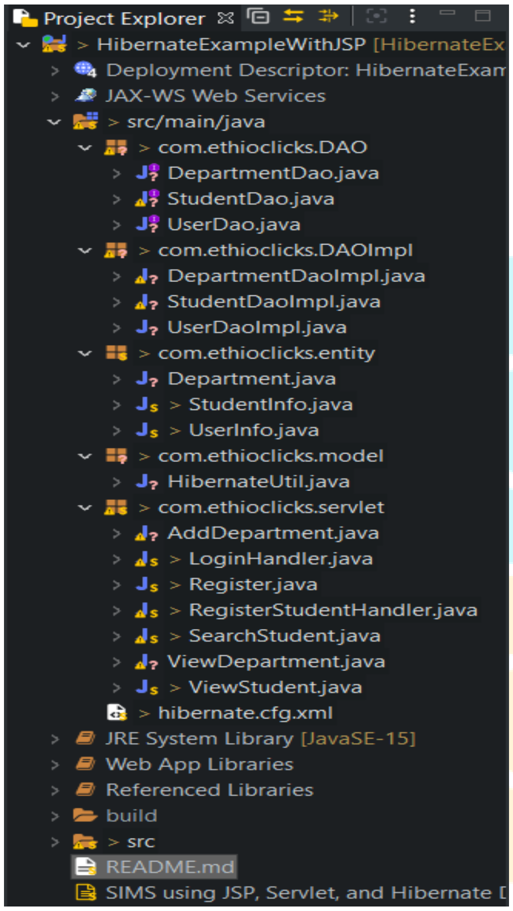
List of Students

ID	First Name	Last Name	Gender	Department	Batch	Reg Date.
1	Abebe	Kebede	Male	Software Engineering	1 Year	2021-06-05 11:49:21.694
2	a	a	Female	Computer Science	1 Year	2021-06-05 12:49:21.694
3	b	b	Male	Information System	3 Year	2021-06-05 12:50:21.694
4	c	c	Female	Information Technology	2 Year	2021-06-05 12:55:21.694
5	d	d	Female	Software Engineering	4 Year	2021-06-05 14:20:12.453
6	asdasd	asdasdasd	Female	Computer Science	5 Year	2021-06-05 15:06:32.909

Figure 9: Search student page.

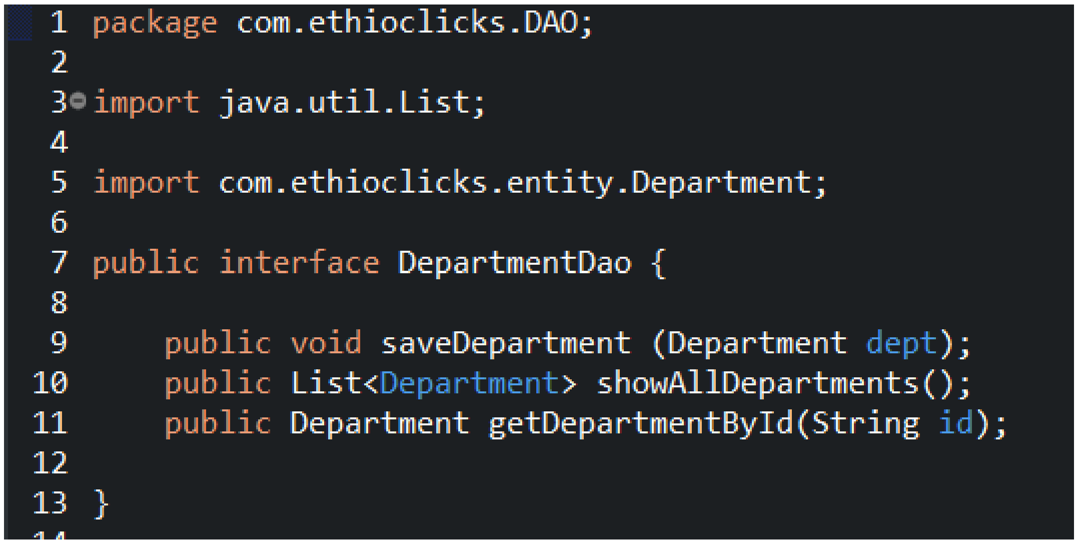
Source code

Project Structure



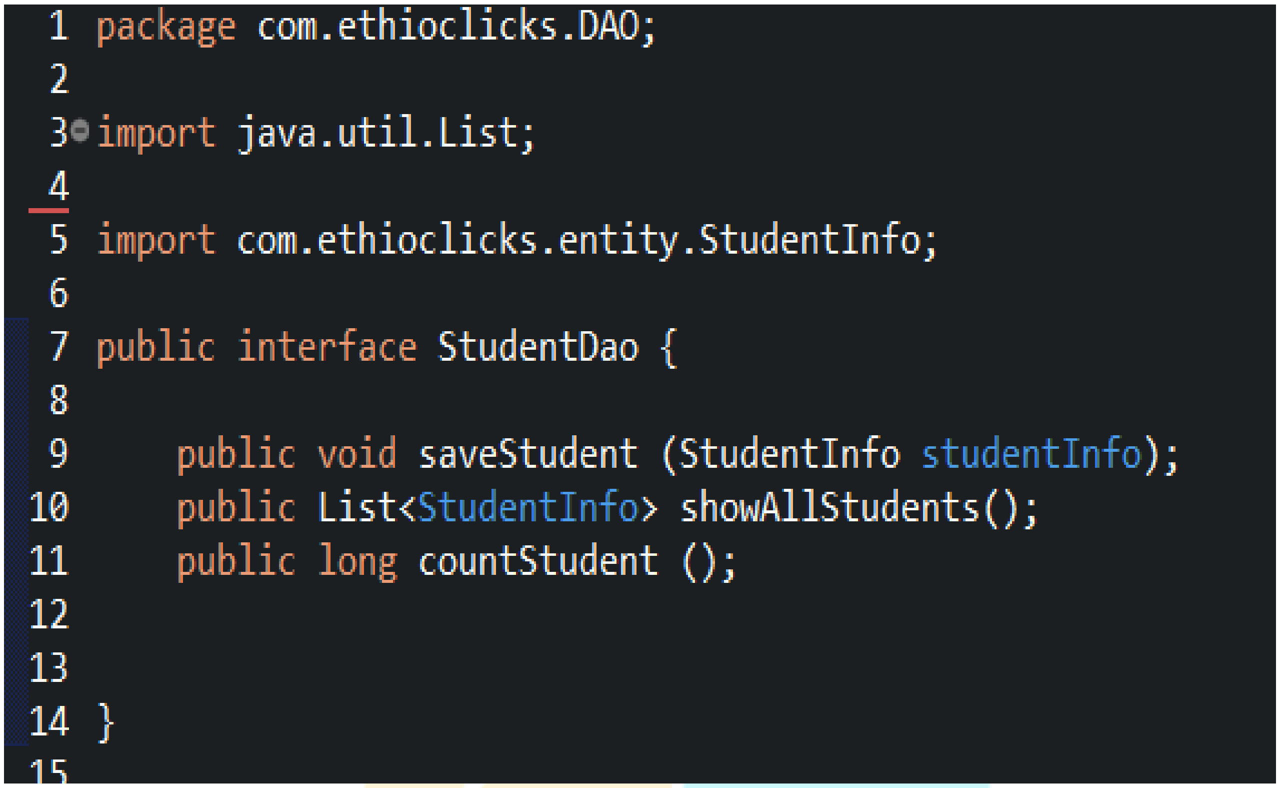
```
Project Explorer
HibernateExampleWithJSP [HibernateEx
  Deployment Descriptor: HibernateExam
  JAX-WS Web Services
  src/main/java
    com.ethioclicks.DAO
      DepartmentDao.java
      StudentDao.java
      UserDao.java
    com.ethioclicks.DAOImpl
      DepartmentDaoImpl.java
      StudentDaoImpl.java
      UserDaoImpl.java
    com.ethioclicks.entity
      Department.java
      StudentInfo.java
      UserInfo.java
    com.ethioclicks.model
      HibernateUtil.java
    com.ethioclicks.servlet
      AddDepartment.java
      LoginHandler.java
      Register.java
      RegisterStudentHandler.java
      SearchStudent.java
      ViewDepartment.java
      ViewStudent.java
    hibernate.cfg.xml
  JRE System Library [JavaSE-15]
  Web App Libraries
  Referenced Libraries
  build
  src
  README.md
  SIMS using JSP, Servlet, and Hibernate D
```

DepartmentDao.java



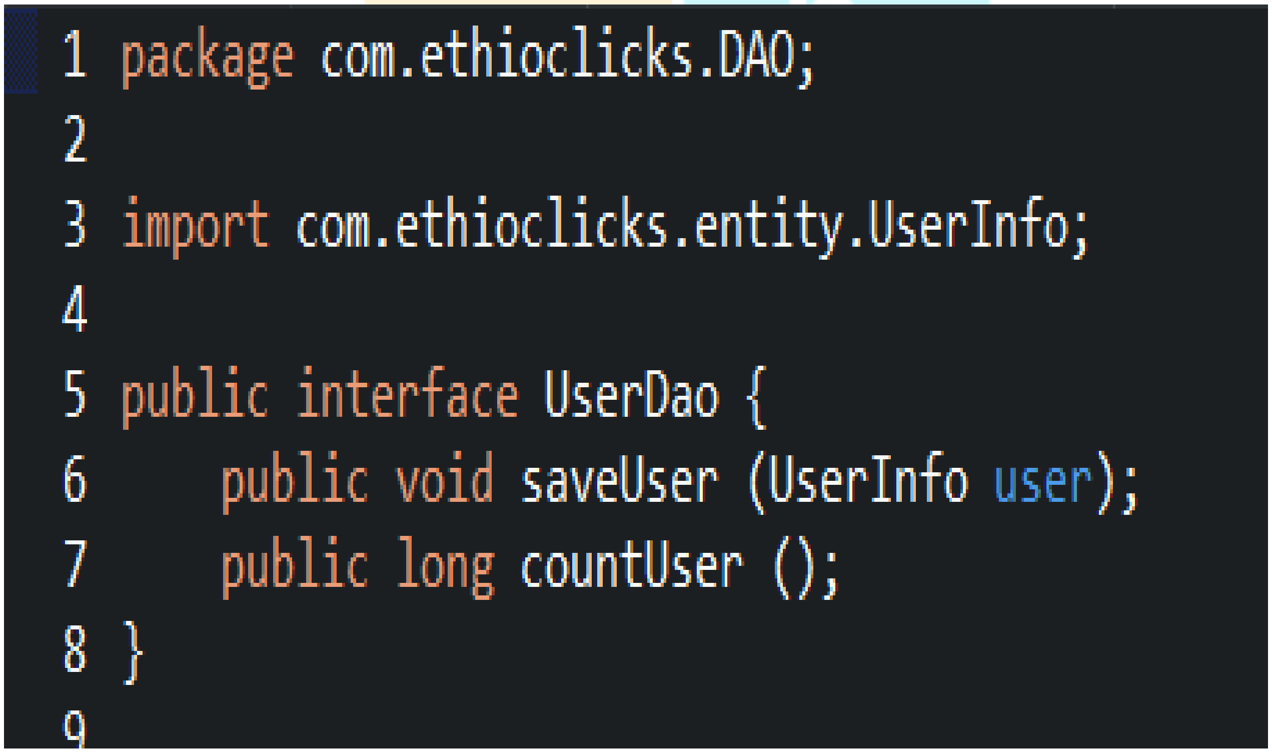
```java
package com.ethioclicks.DAO;

import java.util.List;

import com.ethioclicks.entity.Department;

public interface DepartmentDao {

    public void saveDepartment (Department dept);
    public List<Department> showAllDepartments();
    public Department getDepartmentById(String id);

}
```

StudentDao.java



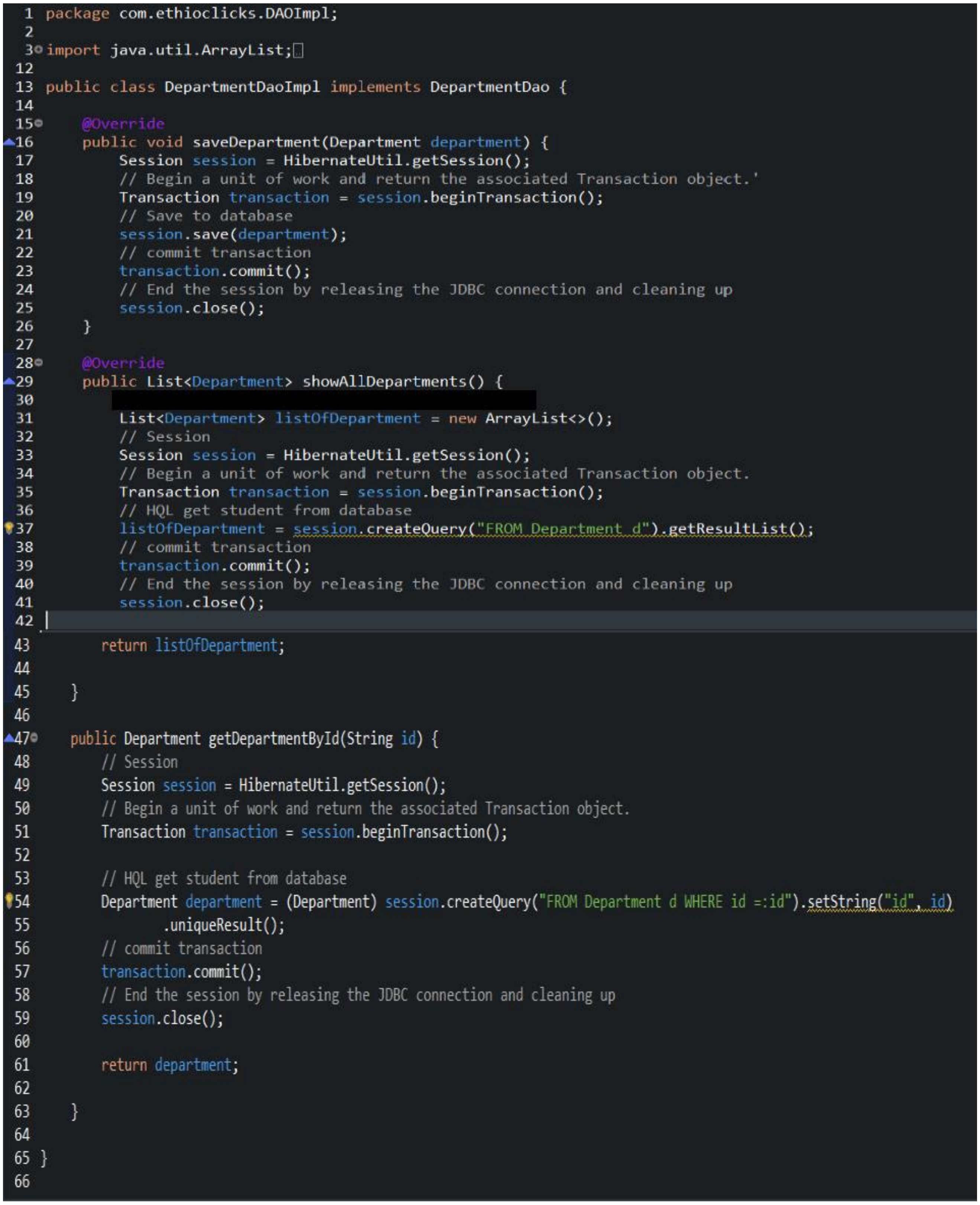
```java
package com.ethioclicks.DAO;

import java.util.List;

import com.ethioclicks.entity.StudentInfo;

public interface StudentDao {

    public void saveStudent (StudentInfo studentInfo);
    public List<StudentInfo> showAllStudents();
    public long countStudent ();


}
```

UserDao.java

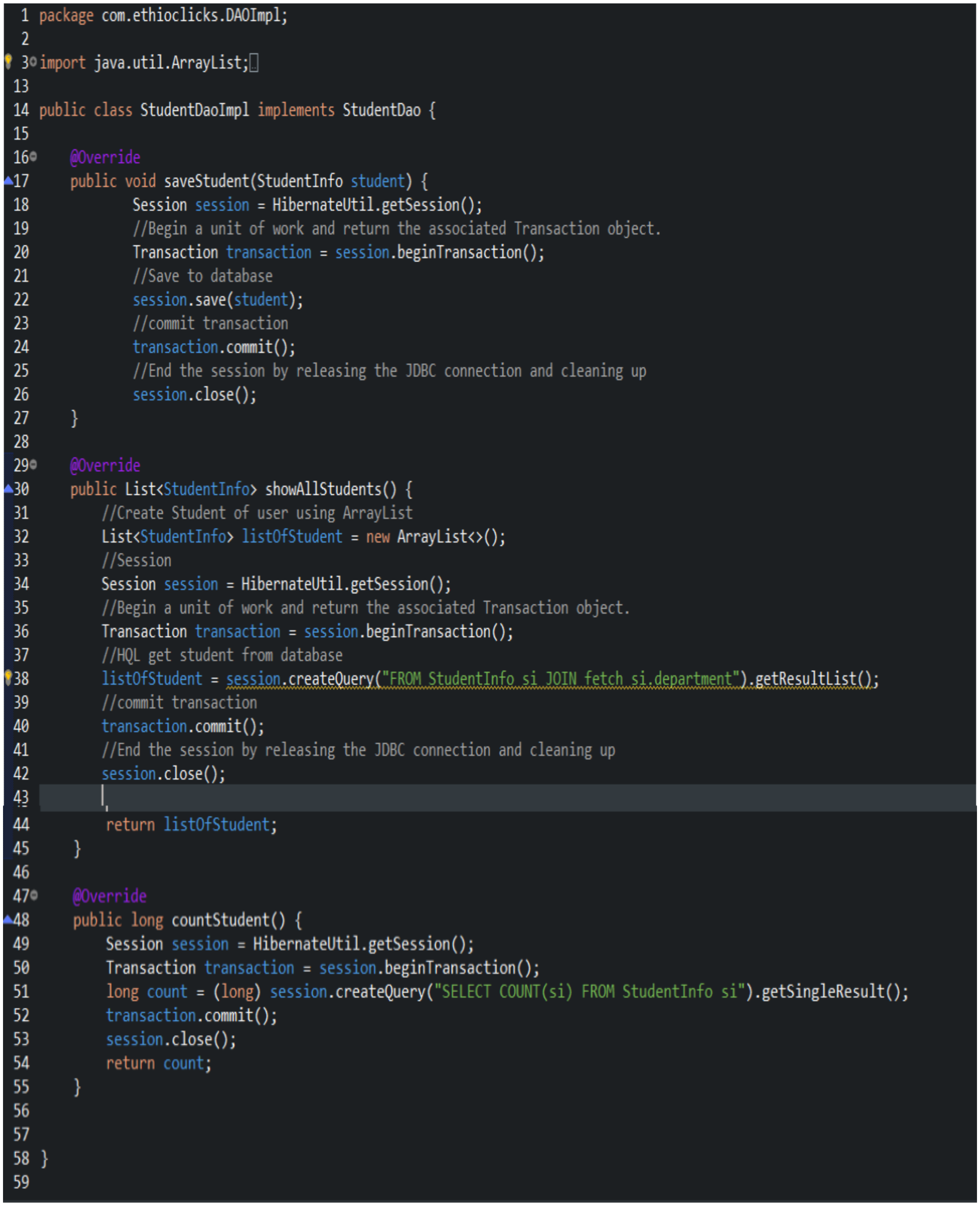


```java
package com.ethioclicks.DAO;

import com.ethioclicks.entity.UserInfo;

public interface UserDao {
    public void saveUser (UserInfo user);
    public long countUser ();
}
```

DepartmentDaoImpl.java

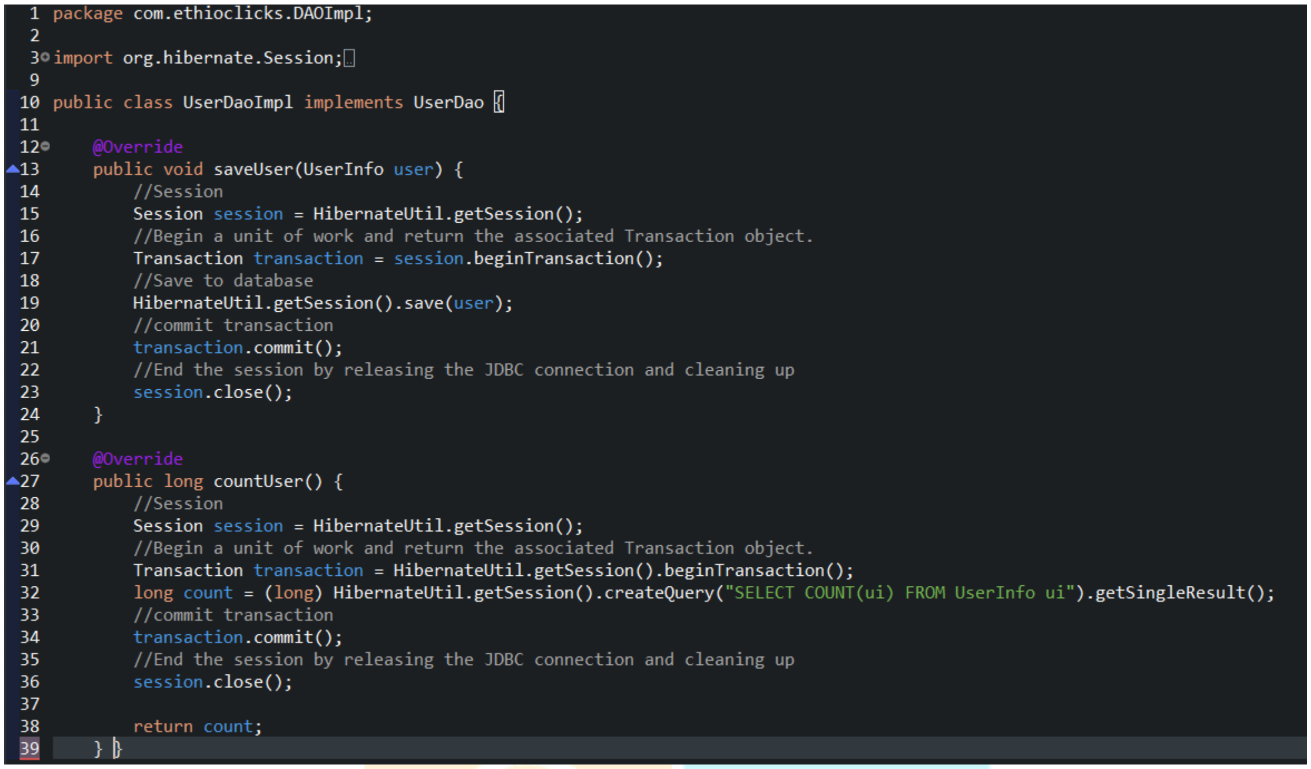

```
package com.ethioclicks.DAOImpl;

import java.util.ArrayList;

public class DepartmentDaoImpl implements DepartmentDao {

    @Override
    public void saveDepartment(Department department) {
        Session session = HibernateUtil.getSession();
        // Begin a unit of work and return the associated Transaction object.'
        Transaction transaction = session.beginTransaction();
        // Save to database
        session.save(department);
        // commit transaction
        transaction.commit();
        // End the session by releasing the JDBC connection and cleaning up
        session.close();
    }

    @Override
    public List<Department> showAllDepartments() {
        [redacted]
        List<Department> listOfDepartment = new ArrayList<>();
        // Session
        Session session = HibernateUtil.getSession();
        // Begin a unit of work and return the associated Transaction object.
        Transaction transaction = session.beginTransaction();
        // HQL get student from database
        listOfDepartment = session.createQuery("FROM Department d").getResultList();
        // commit transaction
        transaction.commit();
        // End the session by releasing the JDBC connection and cleaning up
        session.close();

        return listOfDepartment;

    }

    public Department getDepartmentById(String id) {
        // Session
        Session session = HibernateUtil.getSession();
        // Begin a unit of work and return the associated Transaction object.
        Transaction transaction = session.beginTransaction();

        // HQL get student from database
        Department department = (Department) session.createQuery("FROM Department d WHERE id =:id").setString("id", id)
                .uniqueResult();
        // commit transaction
        transaction.commit();
        // End the session by releasing the JDBC connection and cleaning up
        session.close();

        return department;

    }

}

```

StudentDaoImpl.java



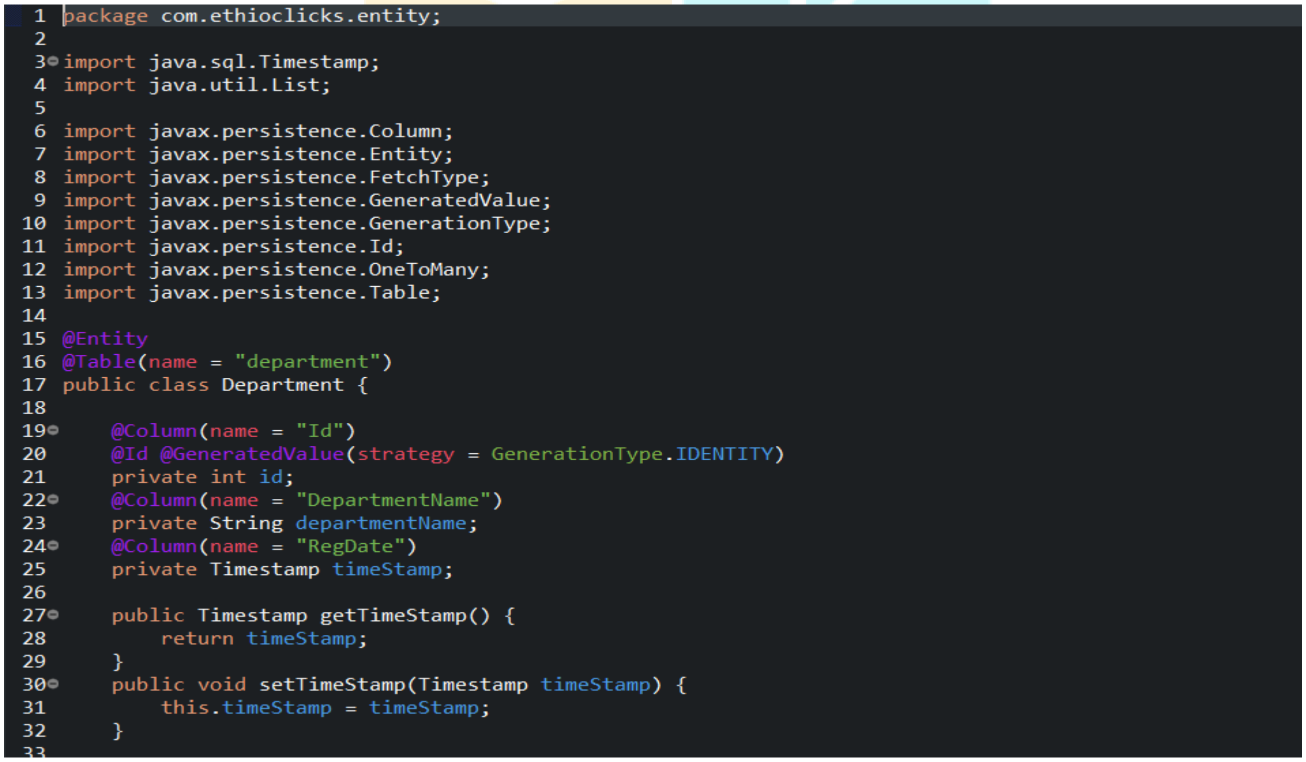
```java
package com.ethioclicks.DAOImpl;

import java.util.ArrayList;

public class StudentDaoImpl implements StudentDao {

    @Override
    public void saveStudent(StudentInfo student) {
            Session session = HibernateUtil.getSession();
            //Begin a unit of work and return the associated Transaction object.
            Transaction transaction = session.beginTransaction();
            //Save to database
            session.save(student);
            //commit transaction
            transaction.commit();
            //End the session by releasing the JDBC connection and cleaning up
            session.close();
    }

    @Override
    public List<StudentInfo> showAllStudents() {
        //Create Student of user using ArrayList
        List<StudentInfo> listOfStudent = new ArrayList<>();
        //Session
        Session session = HibernateUtil.getSession();
        //Begin a unit of work and return the associated Transaction object.
        Transaction transaction = session.beginTransaction();
        //HQL get student from database
        listOfStudent = session.createQuery("FROM StudentInfo si JOIN fetch si.department").getResultList();
        //commit transaction
        transaction.commit();
        //End the session by releasing the JDBC connection and cleaning up
        session.close();

        return listOfStudent;
    }

    @Override
    public long countStudent() {
        Session session = HibernateUtil.getSession();
        Transaction transaction = session.beginTransaction();
        long count = (long) session.createQuery("SELECT COUNT(si) FROM StudentInfo si").getSingleResult();
        transaction.commit();
        session.close();
        return count;
    }


}

```

UserInfoDaoImpl.java

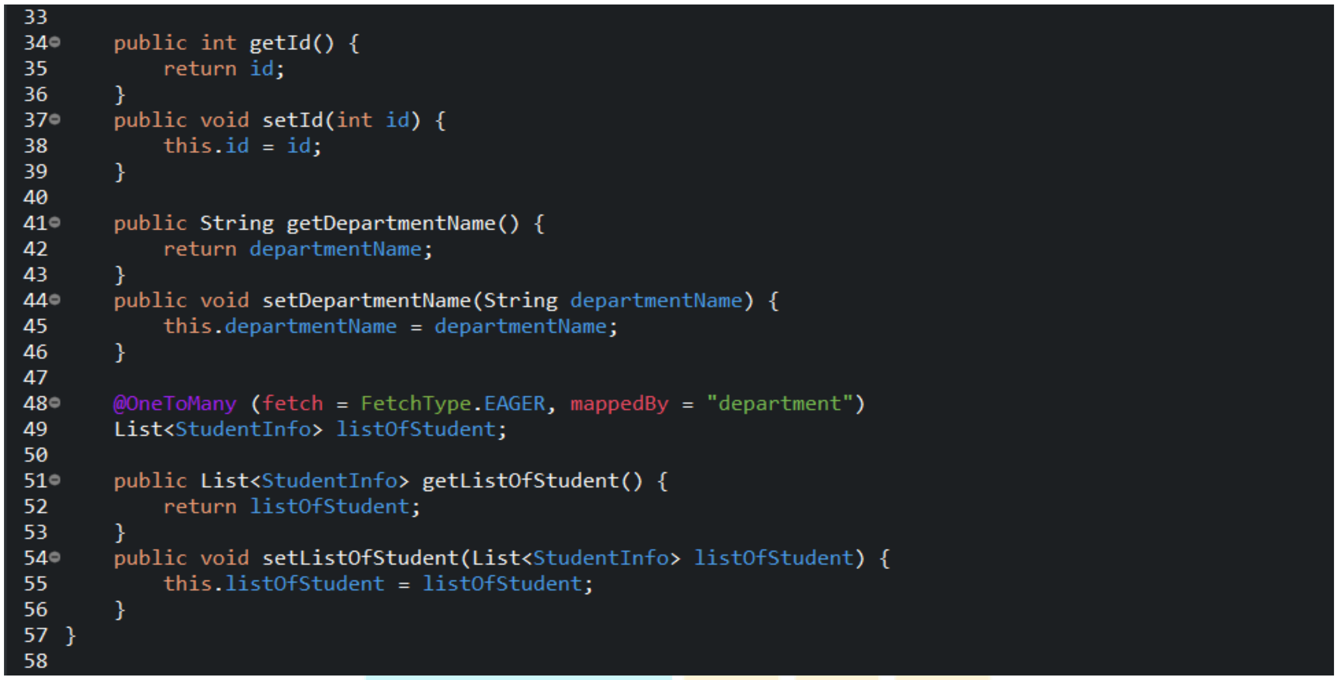


```
package com.ethioclicks.DAOImpl;

import org.hibernate.Session;

public class UserDaoImpl implements UserDao {

    @Override
    public void saveUser(UserInfo user) {
        //Session
        Session session = HibernateUtil.getSession();
        //Begin a unit of work and return the associated Transaction object.
        Transaction transaction = session.beginTransaction();
        //Save to database
        HibernateUtil.getSession().save(user);
        //commit transaction
        transaction.commit();
        //End the session by releasing the JDBC connection and cleaning up
        session.close();
    }

    @Override
    public long countUser() {
        //Session
        Session session = HibernateUtil.getSession();
        //Begin a unit of work and return the associated Transaction object.
        Transaction transaction = HibernateUtil.getSession().beginTransaction();
        long count = (long) HibernateUtil.getSession().createQuery("SELECT COUNT(ui) FROM UserInfo ui").getSingleResult();
        //commit transaction
        transaction.commit();
        //End the session by releasing the JDBC connection and cleaning up
        session.close();

        return count;
    } }
```

Department.java

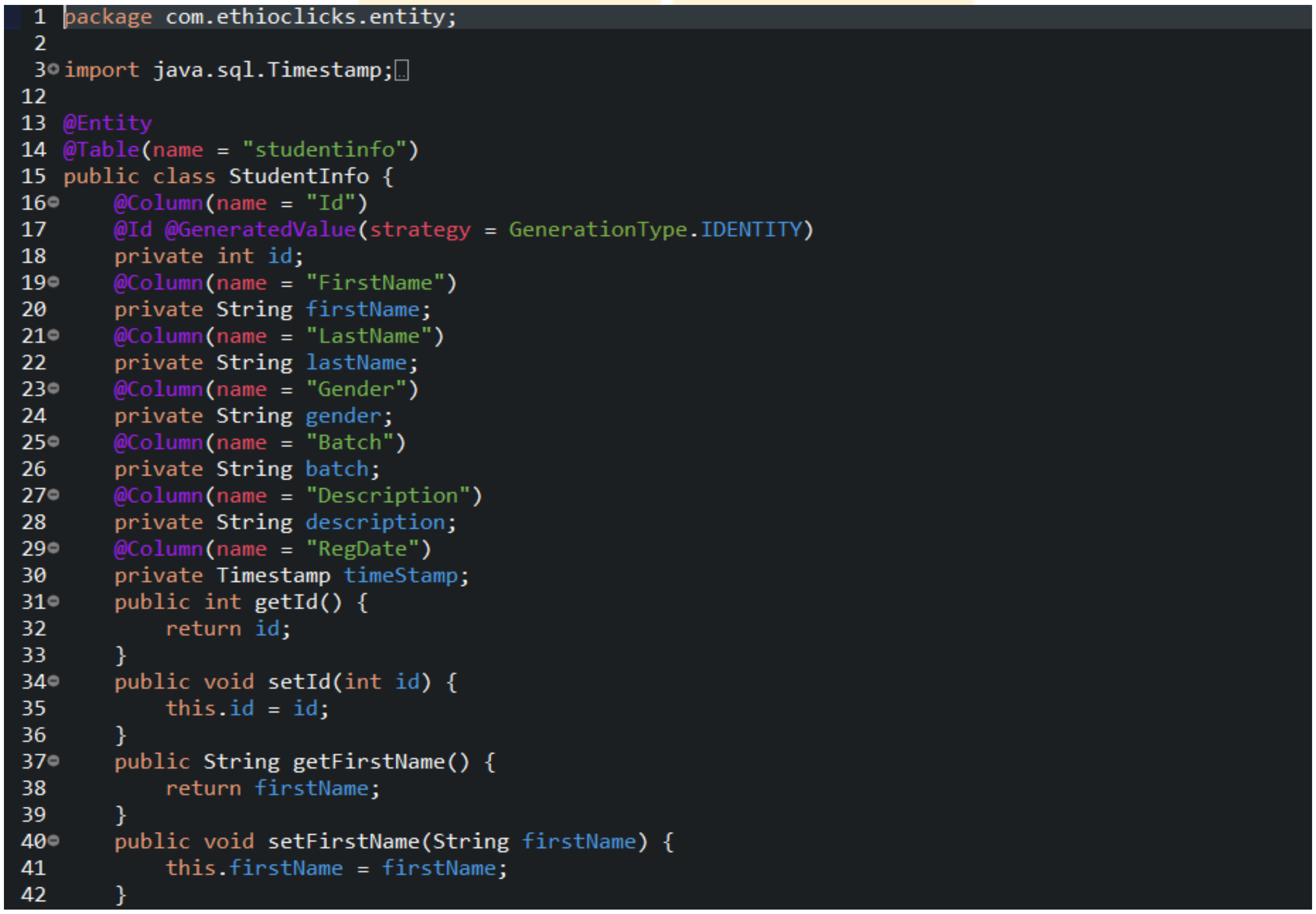


```
package com.ethioclicks.entity;

import java.sql.Timestamp;
import java.util.List;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.FetchType;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.OneToMany;
import javax.persistence.Table;

@Entity
@Table(name = "department")
public class Department {

    @Column(name = "Id")
    @Id @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int id;
    @Column(name = "DepartmentName")
    private String departmentName;
    @Column(name = "RegDate")
    private Timestamp timeStamp;

    public Timestamp getTimeStamp() {
        return timeStamp;
    }
    public void setTimeStamp(Timestamp timeStamp) {
        this.timeStamp = timeStamp;
    }
```

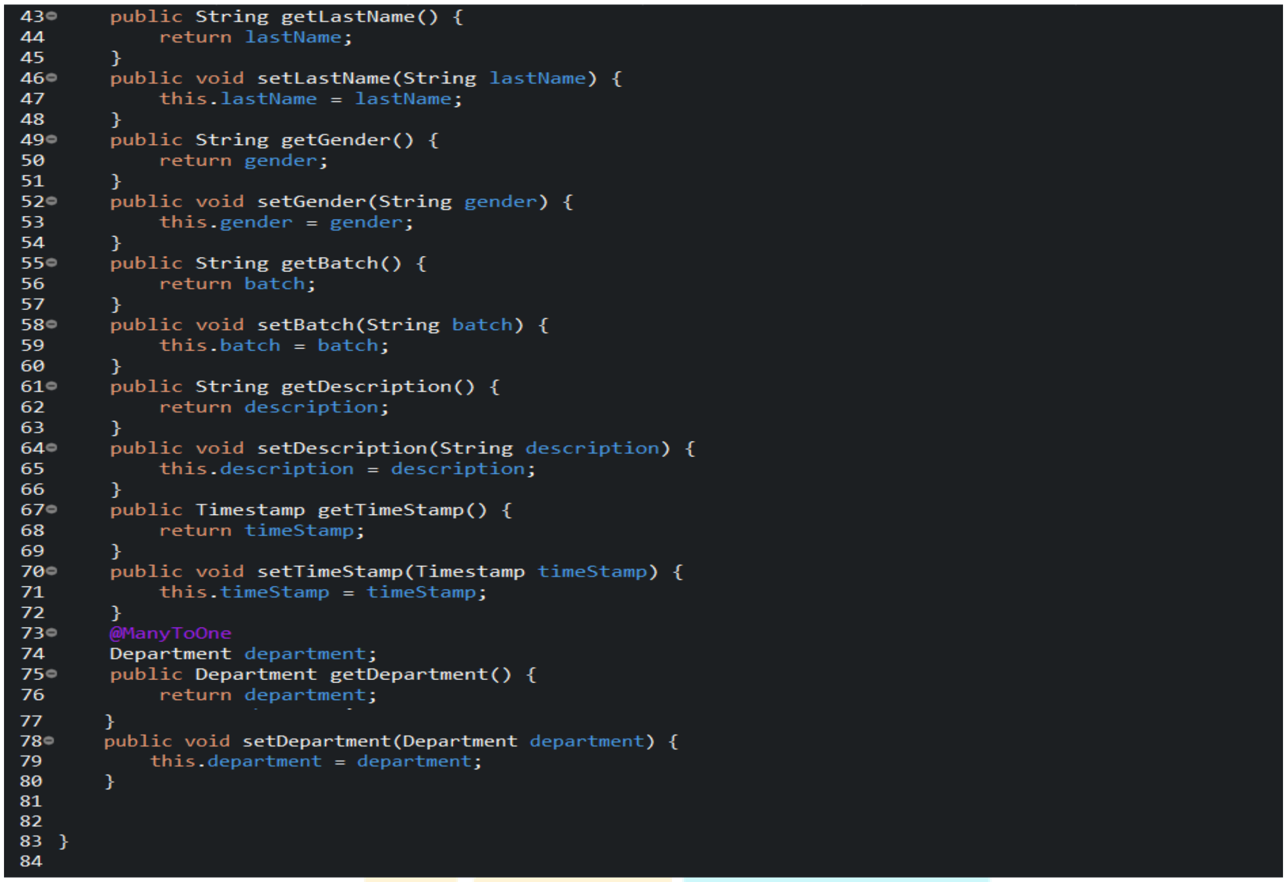


```java

    public int getId() {
        return id;
    }
    public void setId(int id) {
        this.id = id;
    }

    public String getDepartmentName() {
        return departmentName;
    }
    public void setDepartmentName(String departmentName) {
        this.departmentName = departmentName;
    }

    @OneToMany (fetch = FetchType.EAGER, mappedBy = "department")
    List<StudentInfo> listOfStudent;

    public List<StudentInfo> getListOfStudent() {
        return listOfStudent;
    }
    public void setListOfStudent(List<StudentInfo> listOfStudent) {
        this.listOfStudent = listOfStudent;
    }
}

```

StudentInfo.java

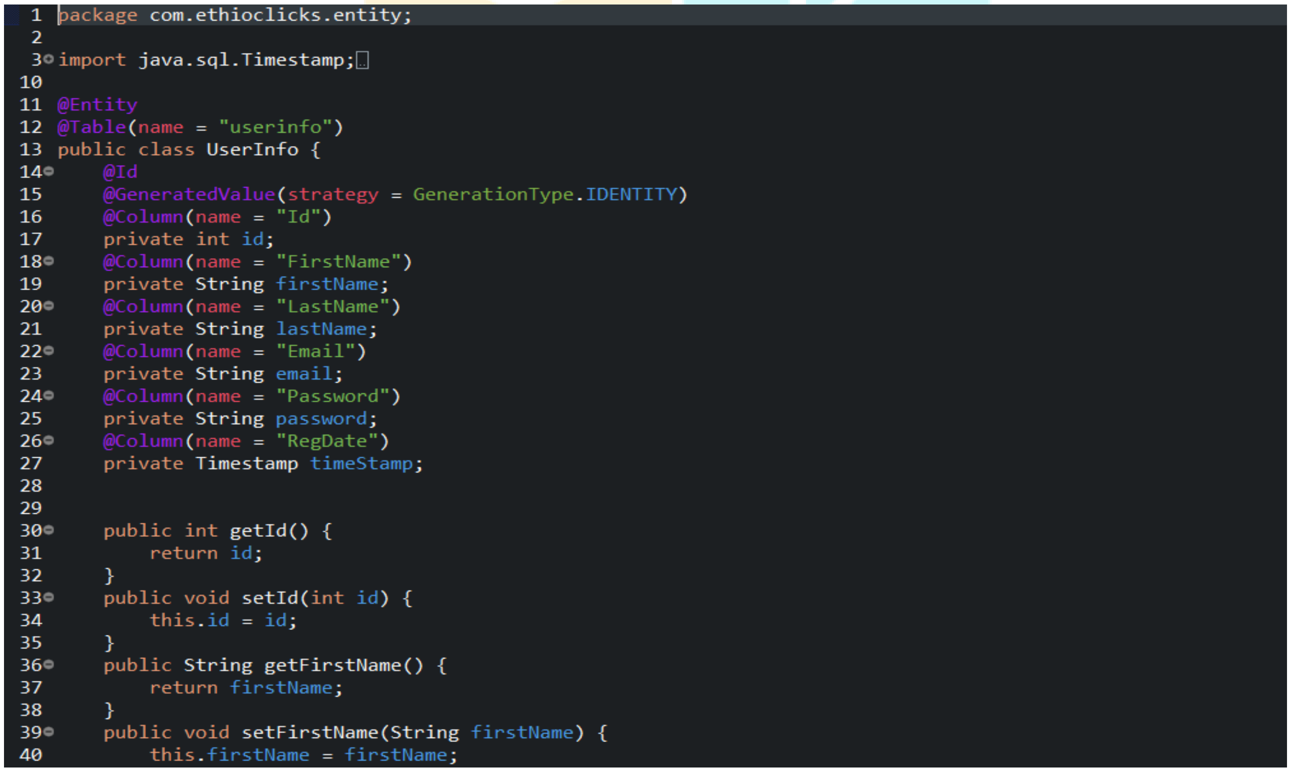


```java
package com.ethioclicks.entity;

import java.sql.Timestamp;

@Entity
@Table(name = "studentinfo")
public class StudentInfo {
    @Column(name = "Id")
    @Id @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int id;
    @Column(name = "FirstName")
    private String firstName;
    @Column(name = "LastName")
    private String lastName;
    @Column(name = "Gender")
    private String gender;
    @Column(name = "Batch")
    private String batch;
    @Column(name = "Description")
    private String description;
    @Column(name = "RegDate")
    private Timestamp timeStamp;
    public int getId() {
        return id;
    }
    public void setId(int id) {
        this.id = id;
    }
    public String getFirstName() {
        return firstName;
    }
    public void setFirstName(String firstName) {
        this.firstName = firstName;
    }
```

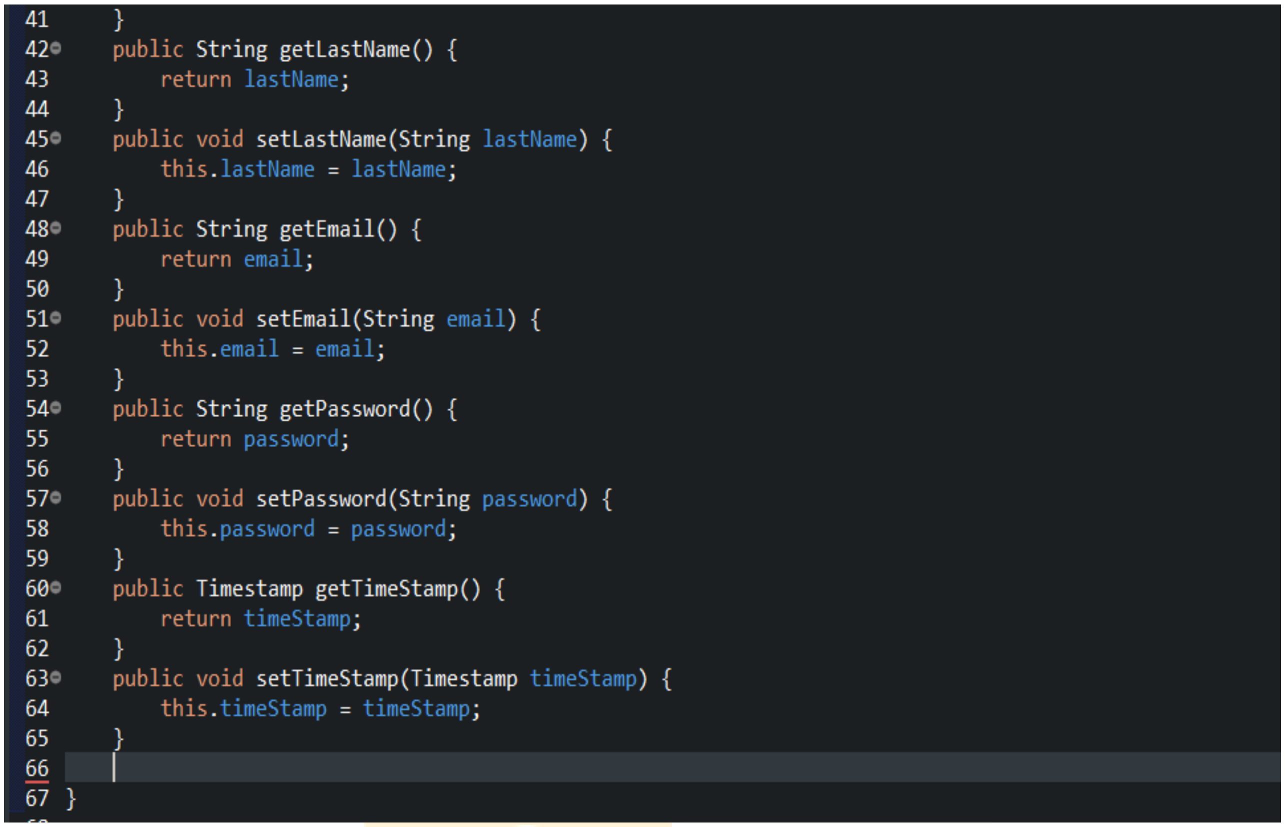


```
    public String getLastName() {
        return lastName;
    }
    public void setLastName(String lastName) {
        this.lastName = lastName;
    }
    public String getGender() {
        return gender;
    }
    public void setGender(String gender) {
        this.gender = gender;
    }
    public String getBatch() {
        return batch;
    }
    public void setBatch(String batch) {
        this.batch = batch;
    }
    public String getDescription() {
        return description;
    }
    public void setDescription(String description) {
        this.description = description;
    }
    public Timestamp getTimeStamp() {
        return timeStamp;
    }
    public void setTimeStamp(Timestamp timeStamp) {
        this.timeStamp = timeStamp;
    }
    @ManyToOne
    Department department;
    public Department getDepartment() {
        return department;
    }
    public void setDepartment(Department department) {
        this.department = department;
    }


}

```

UserInfo.java

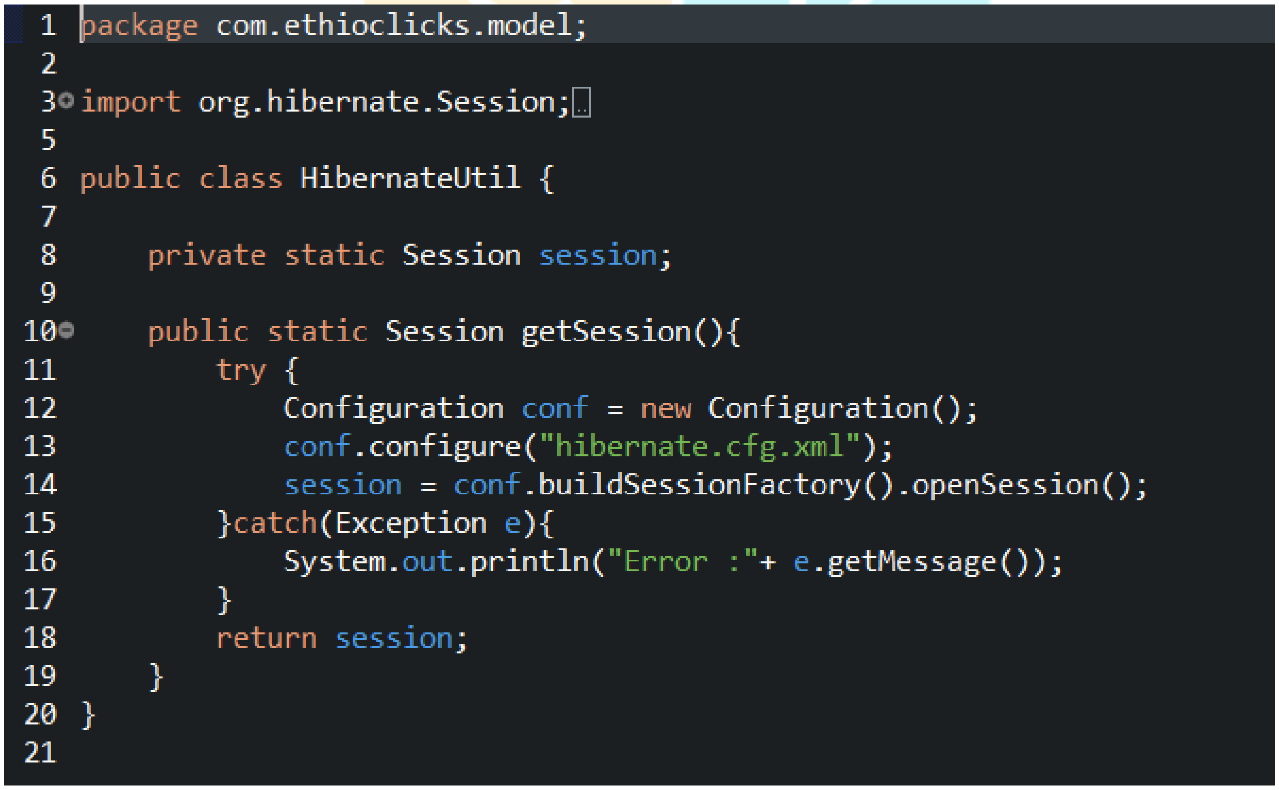


```
package com.ethioclicks.entity;

import java.sql.Timestamp;

@Entity
@Table(name = "userinfo")
public class UserInfo {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Column(name = "Id")
    private int id;
    @Column(name = "FirstName")
    private String firstName;
    @Column(name = "LastName")
    private String lastName;
    @Column(name = "Email")
    private String email;
    @Column(name = "Password")
    private String password;
    @Column(name = "RegDate")
    private Timestamp timeStamp;


    public int getId() {
        return id;
    }
    public void setId(int id) {
        this.id = id;
    }
    public String getFirstName() {
        return firstName;
    }
    public void setFirstName(String firstName) {
        this.firstName = firstName;
```



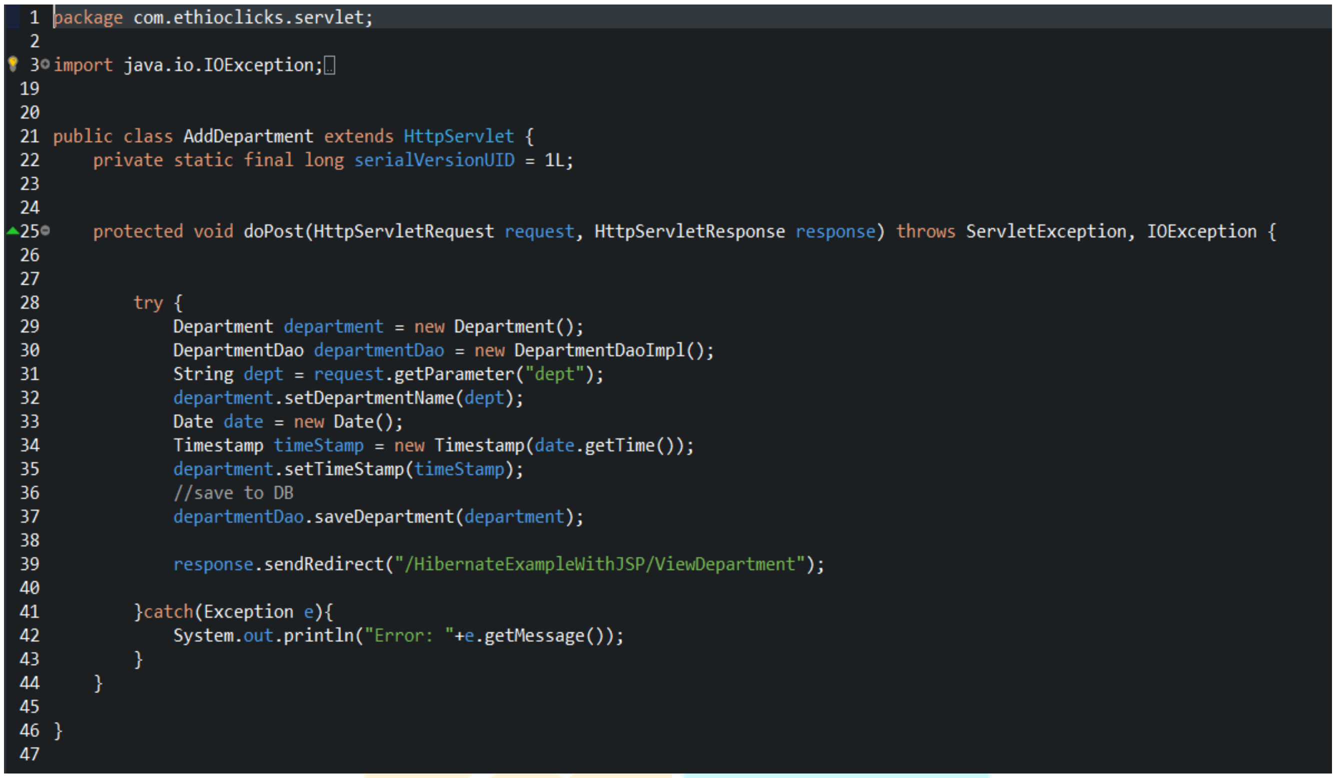
```
    }
    public String getLastName() {
        return lastName;
    }
    public void setLastName(String lastName) {
        this.lastName = lastName;
    }
    public String getEmail() {
        return email;
    }
    public void setEmail(String email) {
        this.email = email;
    }
    public String getPassword() {
        return password;
    }
    public void setPassword(String password) {
        this.password = password;
    }
    public Timestamp getTimeStamp() {
        return timeStamp;
    }
    public void setTimeStamp(Timestamp timeStamp) {
        this.timeStamp = timeStamp;
    }
    
}
```

HibernateUtil.java



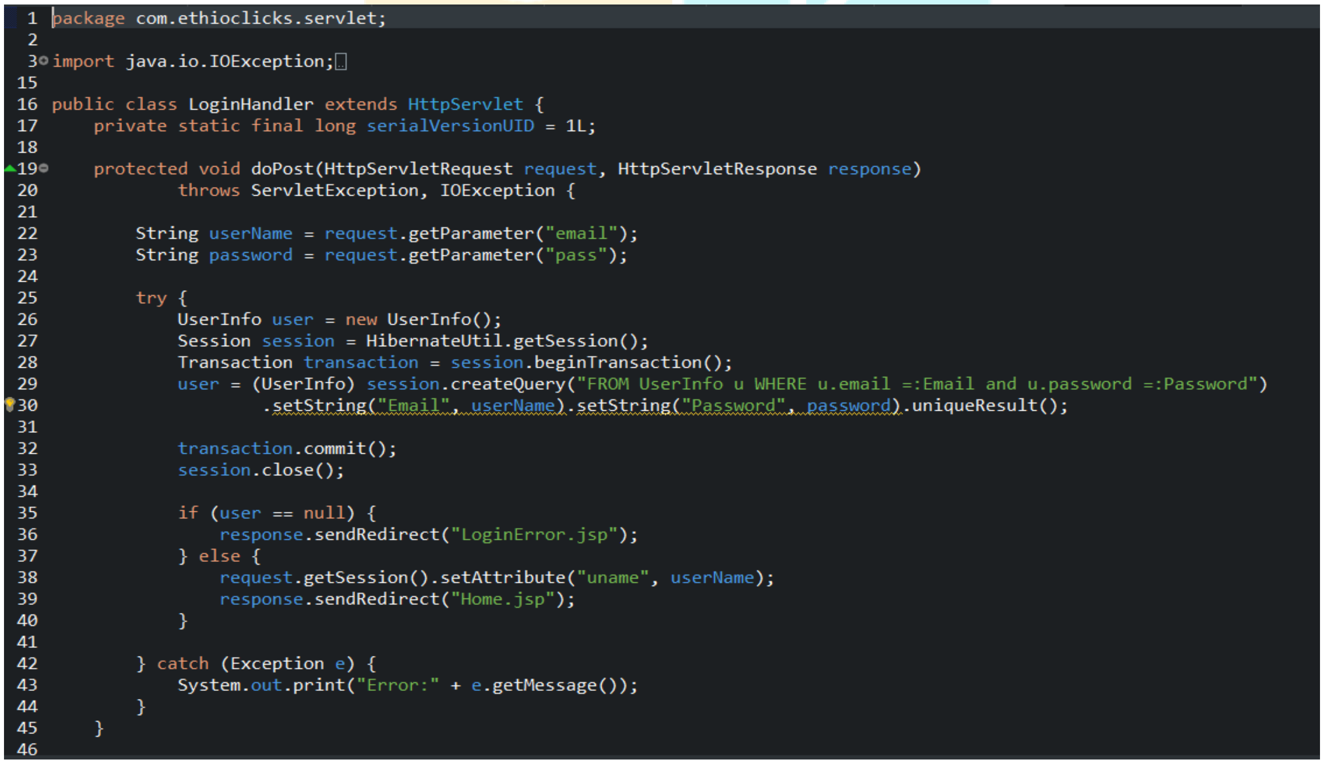
```
package com.ethioclicks.model;

import org.hibernate.Session;

public class HibernateUtil {

    private static Session session;

    public static Session getSession(){
        try {
            Configuration conf = new Configuration();
            conf.configure("hibernate.cfg.xml");
            session = conf.buildSessionFactory().openSession();
        }catch(Exception e){
            System.out.println("Error :"+ e.getMessage());
        }
        return session;
    }
}

```

AddDepartment.java



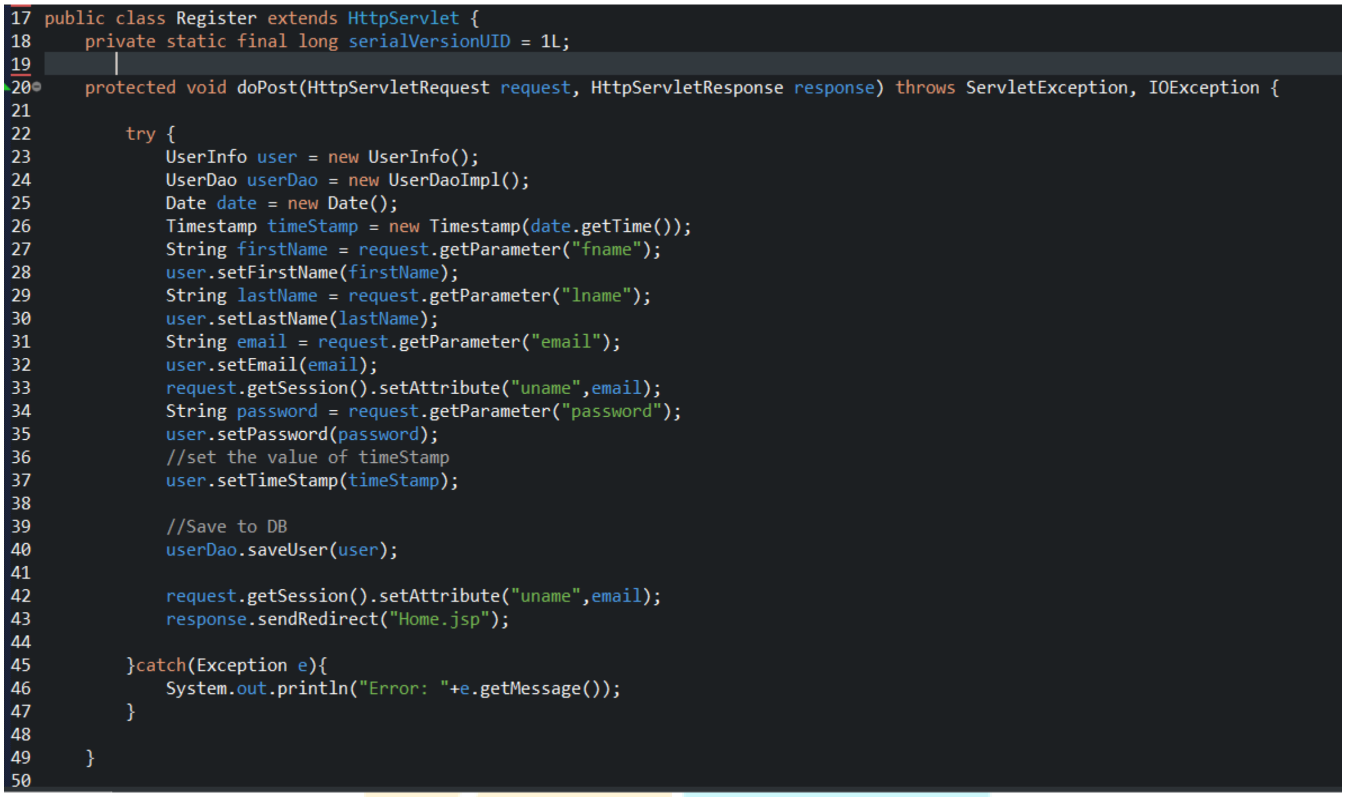
```java
package com.ethioclicks.servlet;

import java.io.IOException;


public class AddDepartment extends HttpServlet {
    private static final long serialVersionUID = 1L;


    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {


        try {
            Department department = new Department();
            DepartmentDao departmentDao = new DepartmentDaoImpl();
            String dept = request.getParameter("dept");
            department.setDepartmentName(dept);
            Date date = new Date();
            Timestamp timeStamp = new Timestamp(date.getTime());
            department.setTimeStamp(timeStamp);
            //save to DB
            departmentDao.saveDepartment(department);

            response.sendRedirect("/HibernateExampleWithJSP/ViewDepartment");

        }catch(Exception e){
            System.out.println("Error: "+e.getMessage());
        }
    }

}

```

LoginHandler.java



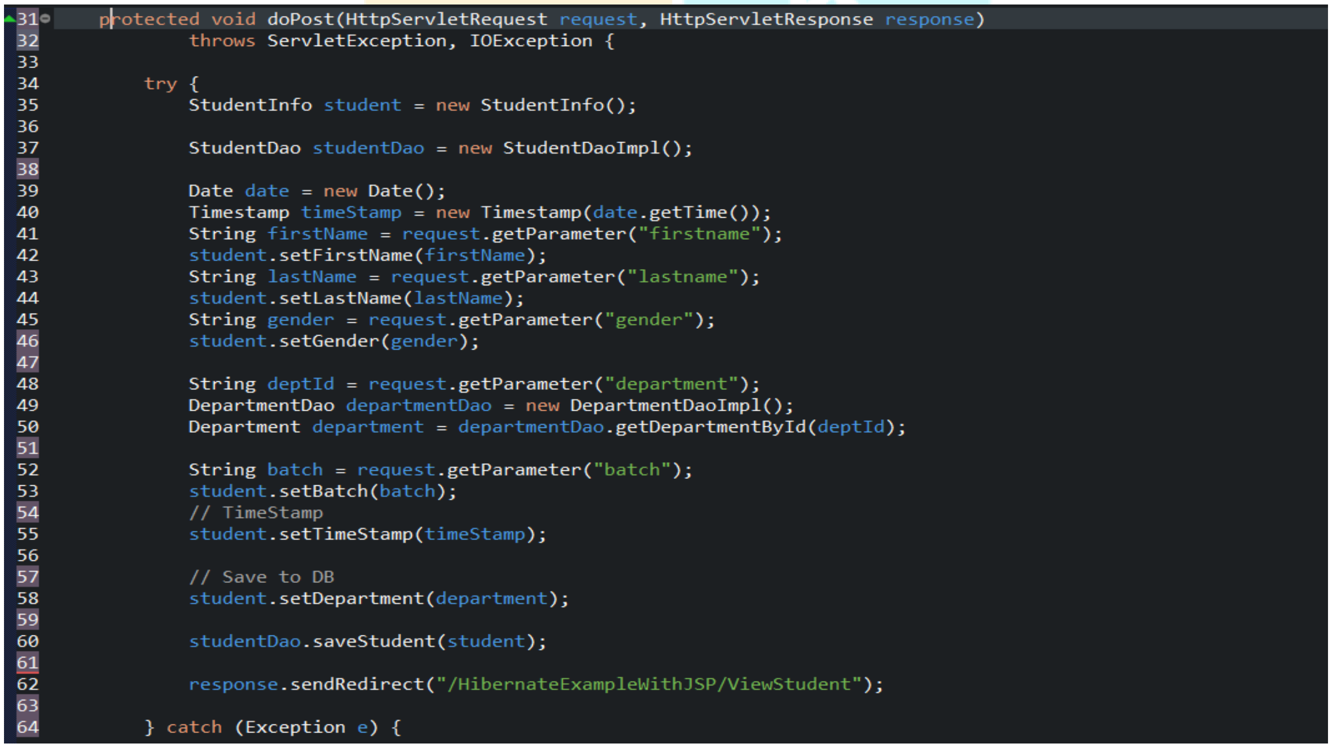
```java
package com.ethioclicks.servlet;

import java.io.IOException;

public class LoginHandler extends HttpServlet {
    private static final long serialVersionUID = 1L;

    protected void doPost(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {

        String userName = request.getParameter("email");
        String password = request.getParameter("pass");

        try {
            UserInfo user = new UserInfo();
            Session session = HibernateUtil.getSession();
            Transaction transaction = session.beginTransaction();
            user = (UserInfo) session.createQuery("FROM UserInfo u WHERE u.email =:Email and u.password =:Password")
                    .setString("Email", userName).setString("Password", password).uniqueResult();

            transaction.commit();
            session.close();

            if (user == null) {
                response.sendRedirect("LoginError.jsp");
            } else {
                request.getSession().setAttribute("uname", userName);
                response.sendRedirect("Home.jsp");
            }

        } catch (Exception e) {
            System.out.print("Error:" + e.getMessage());
        }
    }

```

Register.java

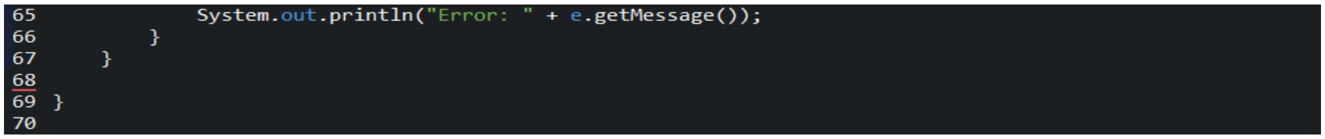


```
public class Register extends HttpServlet {
    private static final long serialVersionUID = 1L;

    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

        try {
            UserInfo user = new UserInfo();
            UserDao userDao = new UserDaoImpl();
            Date date = new Date();
            Timestamp timeStamp = new Timestamp(date.getTime());
            String firstName = request.getParameter("fname");
            user.setFirstName(firstName);
            String lastName = request.getParameter("lname");
            user.setLastName(lastName);
            String email = request.getParameter("email");
            user.setEmail(email);
            request.getSession().setAttribute("uname",email);
            String password = request.getParameter("password");
            user.setPassword(password);
            //set the value of timeStamp
            user.setTimeStamp(timeStamp);

            //Save to DB
            userDao.saveUser(user);

            request.getSession().setAttribute("uname",email);
            response.sendRedirect("Home.jsp");

        }catch(Exception e){
            System.out.println("Error: "+e.getMessage());
        }

    }

```

RegisterStudent.java

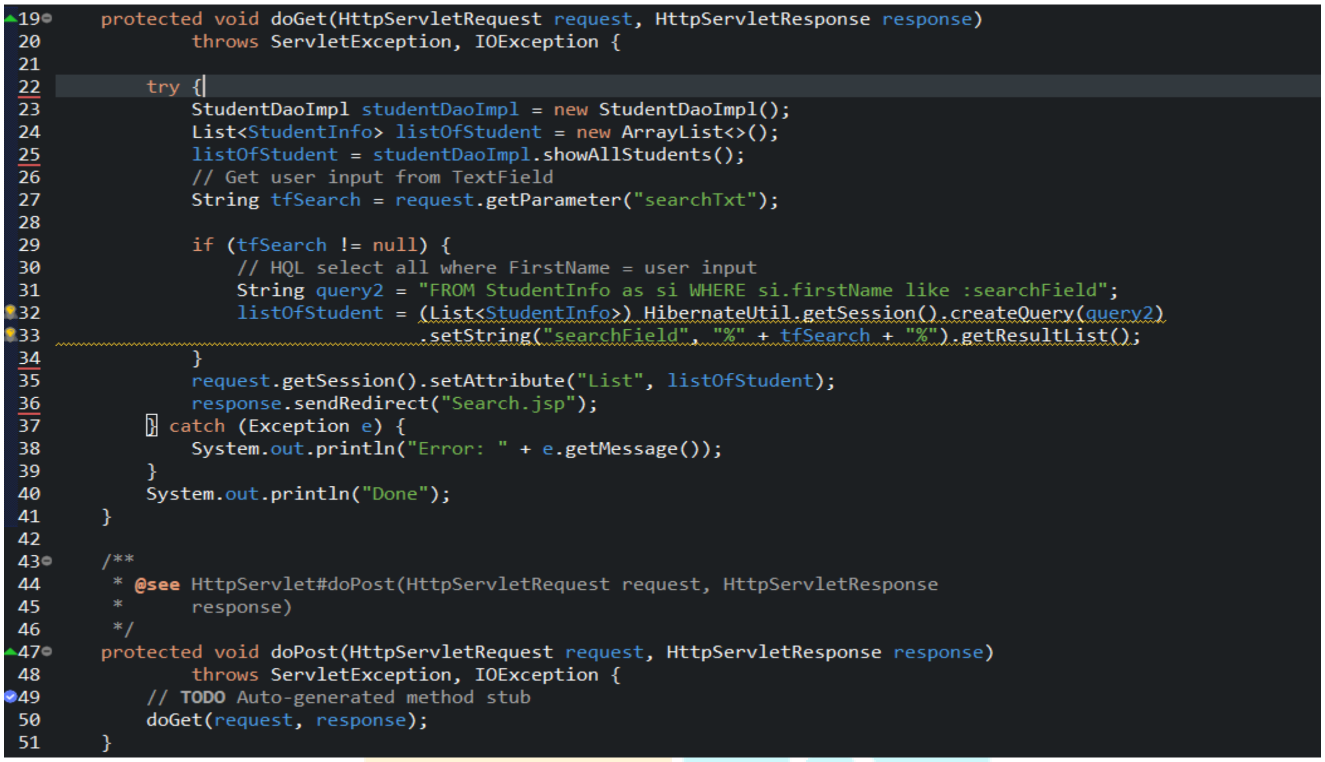


```
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {

        try {
            StudentInfo student = new StudentInfo();

            StudentDao studentDao = new StudentDaoImpl();

            Date date = new Date();
            Timestamp timeStamp = new Timestamp(date.getTime());
            String firstName = request.getParameter("firstname");
            student.setFirstName(firstName);
            String lastName = request.getParameter("lastname");
            student.setLastName(lastName);
            String gender = request.getParameter("gender");
            student.setGender(gender);

            String deptId = request.getParameter("department");
            DepartmentDao departmentDao = new DepartmentDaoImpl();
            Department department = departmentDao.getDepartmentById(deptId);

            String batch = request.getParameter("batch");
            student.setBatch(batch);
            // TimeStamp
            student.setTimeStamp(timeStamp);

            // Save to DB
            student.setDepartment(department);

            studentDao.saveStudent(student);

            response.sendRedirect("/HibernateExampleWithJSP/ViewStudent");

        } catch (Exception e) {
```



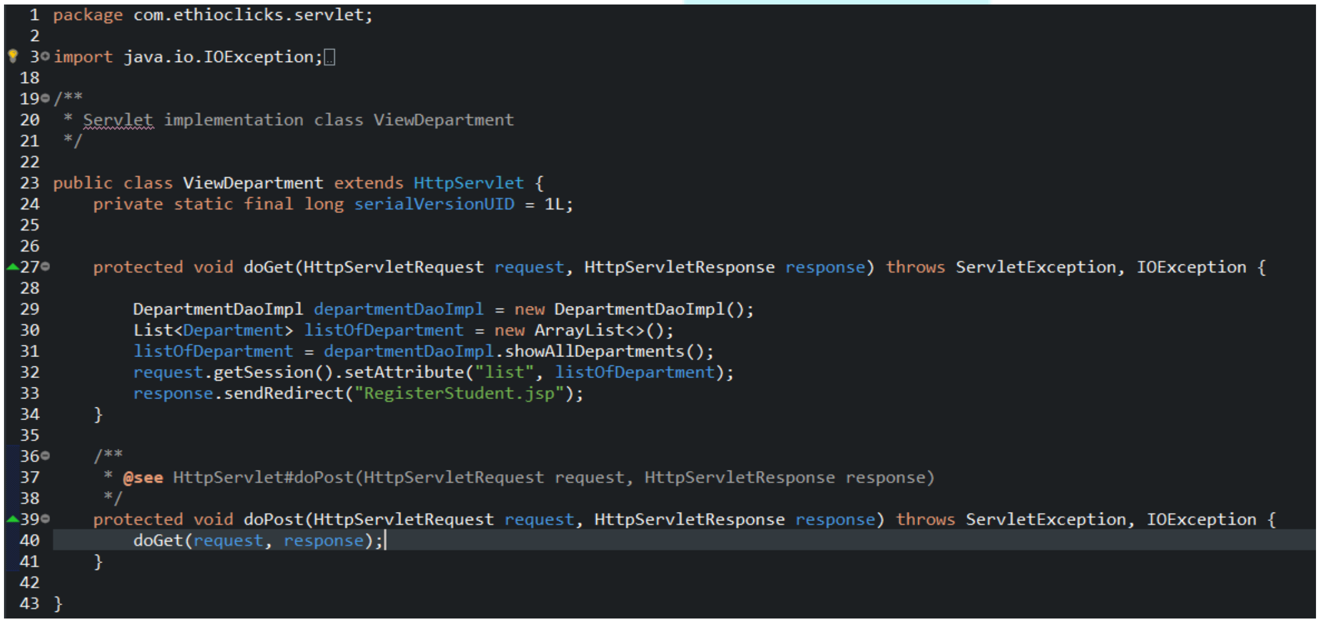
```
            System.out.println("Error: " + e.getMessage());
        }
    }

}
```

SearchStudent.java



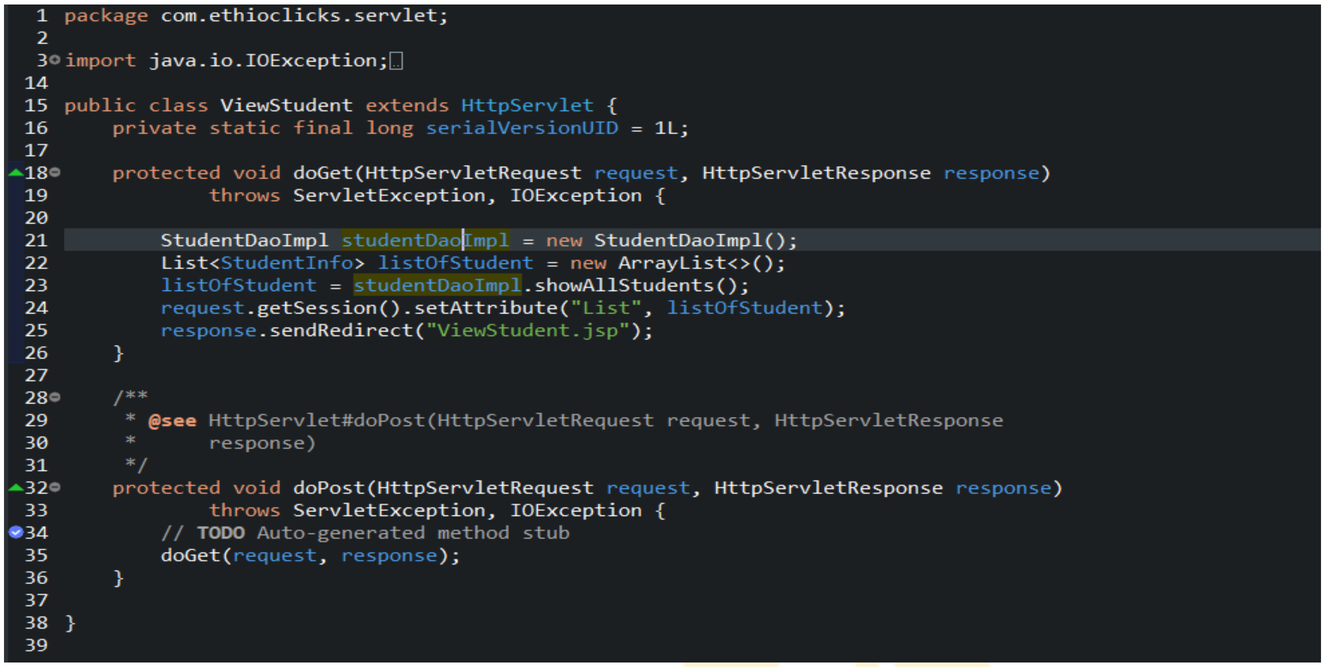
```
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {

        try {
            StudentDaoImpl studentDaoImpl = new StudentDaoImpl();
            List<StudentInfo> listOfStudent = new ArrayList<>();
            listOfStudent = studentDaoImpl.showAllStudents();
            // Get user input from TextField
            String tfSearch = request.getParameter("searchTxt");

            if (tfSearch != null) {
                // HQL select all where FirstName = user input
                String query2 = "FROM StudentInfo as si WHERE si.firstName like :searchField";
                listOfStudent = (List<StudentInfo>) HibernateUtil.getSession().createQuery(query2)
                                .setString("searchField", "%" + tfSearch + "%").getResultList();
            }
            request.getSession().setAttribute("List", listOfStudent);
            response.sendRedirect("Search.jsp");
        } catch (Exception e) {
            System.out.println("Error: " + e.getMessage());
        }
        System.out.println("Done");
    }

    /**
     * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse
     *      response)
     */
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        // TODO Auto-generated method stub
        doGet(request, response);
    }
```

ViewDepartment.java



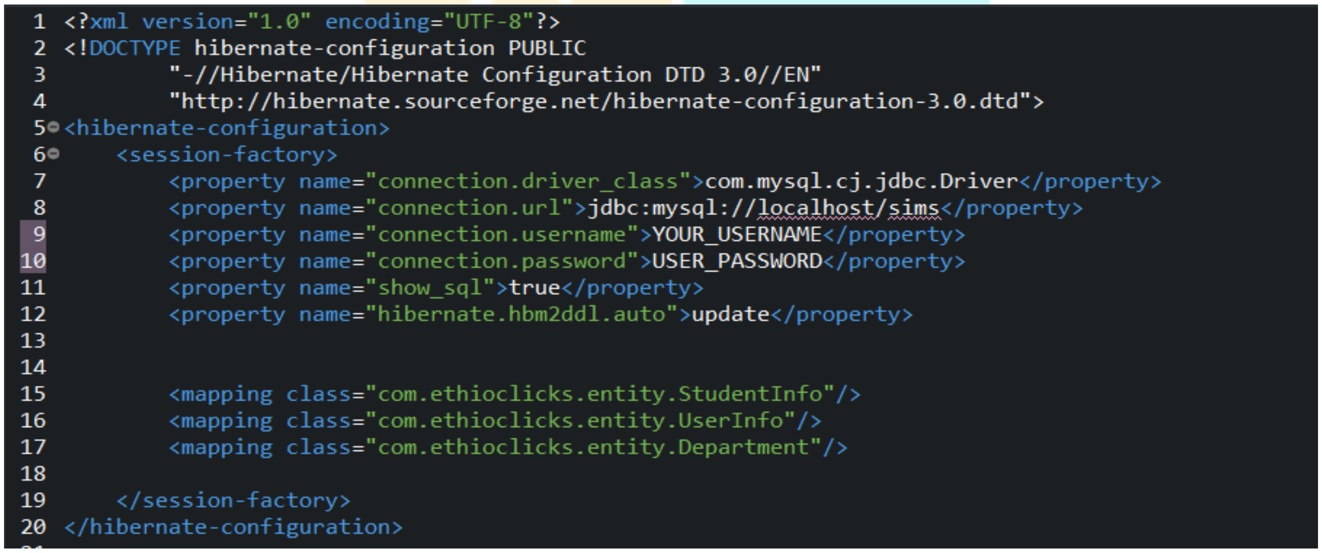
```
package com.ethioclicks.servlet;

import java.io.IOException;

/**
 * Servlet implementation class ViewDepartment
 */

public class ViewDepartment extends HttpServlet {
    private static final long serialVersionUID = 1L;


    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

        DepartmentDaoImpl departmentDaoImpl = new DepartmentDaoImpl();
        List<Department> listOfDepartment = new ArrayList<>();
        listOfDepartment = departmentDaoImpl.showAllDepartments();
        request.getSession().setAttribute("list", listOfDepartment);
        response.sendRedirect("RegisterStudent.jsp");
    }

    /**
     * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)
     */
    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        doGet(request, response);
    }

}
```

ViewStudent.java



```
package com.ethioclicks.servlet;

import java.io.IOException;

public class ViewStudent extends HttpServlet {
    private static final long serialVersionUID = 1L;

    protected void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {

        StudentDaoImpl studentDaoImpl = new StudentDaoImpl();
        List<StudentInfo> listOfStudent = new ArrayList<>();
        listOfStudent = studentDaoImpl.showAllStudents();
        request.getSession().setAttribute("List", listOfStudent);
        response.sendRedirect("ViewStudent.jsp");
    }

    /**
     * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse
     *      response)
     */
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        // TODO Auto-generated method stub
        doGet(request, response);
    }

}
```

Hibernate.cfg.xml



```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-configuration PUBLIC
        "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
        "http://hibernate.sourceforge.net/hibernate-configuration-3.0.dtd">
<hibernate-configuration>
    <session-factory>
        <property name="connection.driver_class">com.mysql.cj.jdbc.Driver</property>
        <property name="connection.url">jdbc:mysql://localhost/sims</property>
        <property name="connection.username">YOUR_USERNAME</property>
        <property name="connection.password">USER_PASSWORD</property>
        <property name="show_sql">true</property>
        <property name="hibernate.hbm2ddl.auto">update</property>


        <mapping class="com.ethioclicks.entity.StudentInfo"/>
        <mapping class="com.ethioclicks.entity.UserInfo"/>
        <mapping class="com.ethioclicks.entity.Department"/>

    </session-factory>
</hibernate-configuration>
```

Logout Page (Logout.jsp)

```
<%
        session.removeAttribute("uname");
        session.invalidate();
        response.sendRedirect("Login.jsp");
%>
```

Future Scope

- Action (Edit & Delete).
- Improving Security (Password Encryption).
- User validation using email and password recovery.
- Normalization.
- Print student information.