Developing A Backend Admin for Learner's Academy

This document contains sections for:

- Sprint planning and Task completion
- Core concepts used in project
- Flow of the Application.
- Demonstrating the product capabilities, appearance, and user interactions.
- Unique Selling Points of the Application
- Conclusions

The code for this project is hosted at https://github.com/AmitDhanorkar/LearnersAcademyAdministrativePortal

The project is developed by Amit Dhanorkar.

Development Environment

- Eclipse IDE for Enterprise Java Developers vOxygen Release (4.7.0)
- Apache Tomcat Server v9.0
- JRE: OpenJDK Runtime Environment 1.8.0_202
- All other software components are defined in the Project Object Model pom.xml

Sprints planning and Task completion

The project is planned to be completed in 3 sprints. Tasks assumed to be completed in the sprints are:

sprint 1:

• Creating the flow of the application

• Initializing git repository to track changes as development progresses.

• Writing the Java program to fulfil the requirements of the project and design UI

Sprint 2:

• Testing the Java program with different kinds of User input

• Pushing code to GitHub.

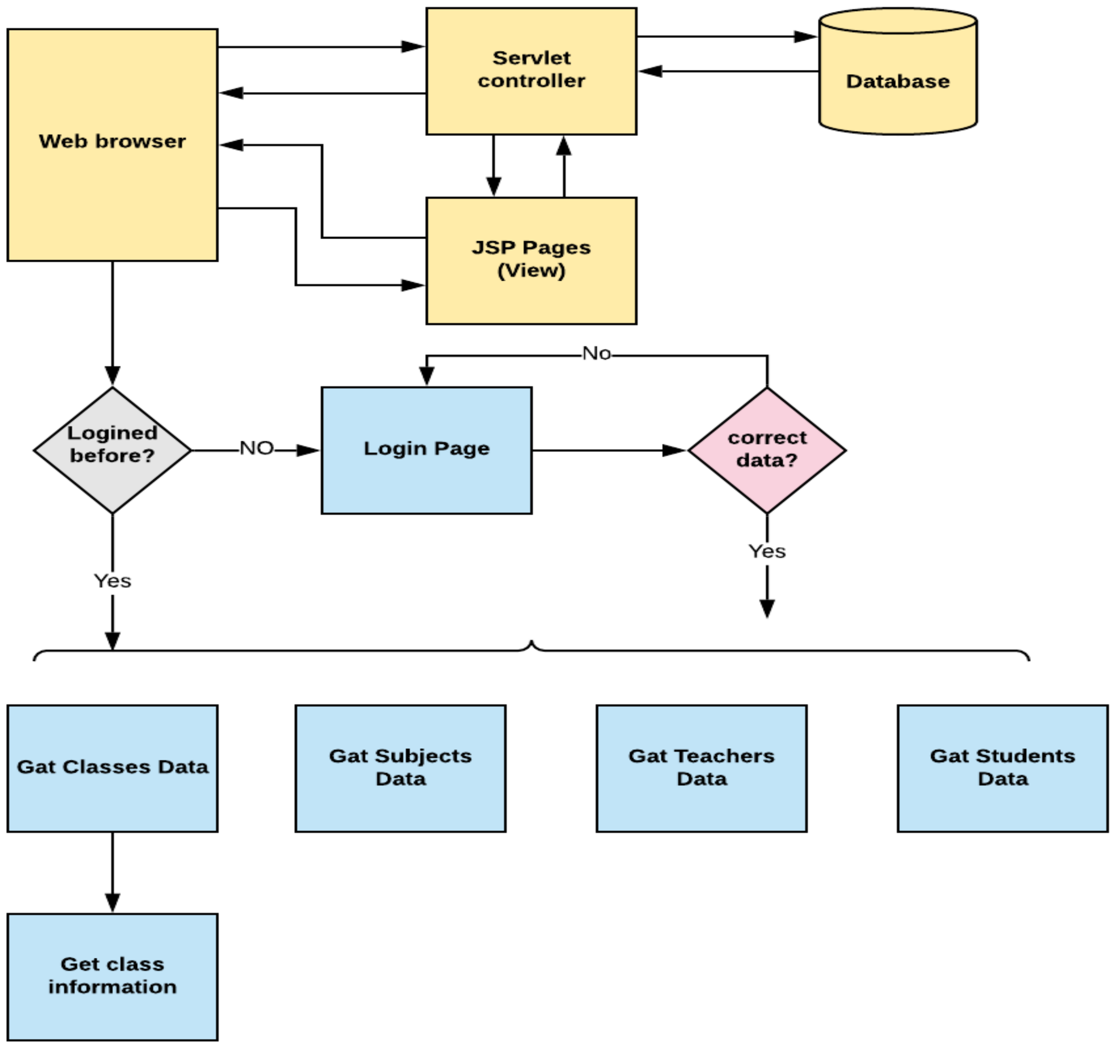
Sprint 3:

• Creating this specification document highlighting application capabilities, appearance, and user interactions.

Core concepts used in project

Servlets, Session-tracking, JSP, Hibernate, CSS

Flow of the Application



Demonstrating the product capabilities, appearance, and user interactions

To demonstrate the product capabilities, below are the sub-sections configured to highlight appearance and user interactions for the project:

1. Creating a maven project
2. Configuring pom.xml

3. Creating database and filling with sample data
4. Creating an Entity classes
5. Creating a DAO classes
6. Creating a servlet
7. Creating an JSP pages
8. Configuring Hibernate
9. Configuring web.xml
10. Building the project
11. Publishing and starting the project
12. Running the project
13. Pushing the code to your GitHub repositories

Step 1: Creating a maven project

- Open Eclipse
- Go the **File** menu. Choose **New->Maven Project**
- Check the checkbox **Use default Workspace location.** Click on **Next**
- Select an Archetype as **Maven Webapp project.** Click on **Next**
- Enter **Specify Archetype parameters like Group Id, Artifact Id etc.** and click on **Finish**
- This will create the project files in the Project Explorer

Step 2: Configuring pom.xml

- In the Project Explorer, expand **ValidationOfTheUserLogin**
- Double click on **pom.xml** to open it in the editor
- Enter the following script:

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/maven-
v4_0_0.xsd">
<modelVersion>4.0.0</modelVersion>
<groupId>com.learnersacademy</groupId>
<artifactId>LearnersAcademyAdministrativePortal</artifactId>
<packaging>war</packaging>
<version>0.0.1-SNAPSHOT</version>
<name>LearnersAcademyAdministrativePortal Maven Webapp</name>
<url>http://maven.apache.org</url>
<dependencies>
<!-- Servlets -->
<dependency>
<groupId>javax.servlet</groupId>
<artifactId>servlet-api</artifactId>
<version>2.5</version>
<scope>provided</scope>
</dependency>
<dependency>
<groupId>javax.servlet</groupId>
<artifactId>javax.servlet-api</artifactId>
<version>3.0.1</version>
<scope>provided</scope>
</dependency>

<!-- MySQL Connector -->
<dependency>
<groupId>mysql</groupId>
<artifactId>mysql-connector-java</artifactId>
<version>8.0.26</version>
</dependency>

<!-- PostGresSQL Connector -->
<dependency>
<groupId>org.postgresql</groupId>
<artifactId>postgresql</artifactId>
<version>42.2.14</version>
</dependency>


<!-- Hibernates -->
<dependency>
```

```
<groupId>org.hibernate</groupId>
<artifactId>hibernate-core</artifactId>
<version>5.5.7.Final</version>
</dependency>

<dependency>
<groupId>org.hibernate</groupId>
<artifactId>hibernate-entitymanager</artifactId>
<version>5.4.10.Final</version>
</dependency>

<!-- JSP JSTL Tags -->
<dependency>
<groupId>taglibs</groupId>
<artifactId>standard</artifactId>
<version>1.1.2</version>
</dependency>
<dependency>
<groupId>jstl</groupId>
<artifactId>jstl</artifactId>
<version>1.2</version>
<scope>compile</scope>
</dependency>

<dependency>
<groupId>javax.servlet.jsp.jstl</groupId>
<artifactId>jstl-api</artifactId>
<version>1.2</version>
</dependency>

<dependency>
<groupId>org.apache.taglibs</groupId>
<artifactId>taglibs-standard-spec</artifactId>
<version>1.2.5</version>
</dependency>
<dependency>
<groupId>org.apache.taglibs</groupId>
<artifactId>taglibs-standard-impl</artifactId>
<version>1.2.5</version>
</dependency>

<dependency>
<groupId>net.sf.json-lib</groupId>
<artifactId>json-lib</artifactId>
<version>2.4</version>
```

```
<classifier>jdk13</classifier>
</dependency>

</dependencies>
<build>
<finalName>LearnersAcademyAdministrativePortal</finalName>
</build>
</project>
```

Step 3: Creating database and filling with sample data

- MySQL is already installed in your machine.
- Login to the MySQL command line console
- Type **CREATE DATABASE learners_academy** and press **Enter**
- Type **USE learners_academy** and press **Enter**
- Enter the following script and execute it:

```
CREATE TABLE `class` (
  `id` int NOT NULL AUTO_INCREMENT,
  `className` varchar(255) NOT NULL,
  `classTeacherName` varchar(255) DEFAULT NULL,
  `createdDt` datetime DEFAULT NULL,
  `roomNo` varchar(255) DEFAULT NULL,
  `sectionName` varchar(255) DEFAULT NULL,
  `totalNumberOfStudents` int NOT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `className` (`className`)
) ENGINE=InnoDB AUTO_INCREMENT=12 DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_0900_ai_ci

INSERT INTO `class` (`id`, `className`, `classTeacherName`, `createdDt`, `roomNo`,
`sectionName`, `totalNumberOfStudents`) VALUES
(12, '10th Standard', 'Raghavendra', '2022-02-12 21:35:06', '101', 'A', 30);

CREATE TABLE `student` (
  `id` int NOT NULL AUTO_INCREMENT,
  `address` varchar(255) DEFAULT NULL,
  `age` int NOT NULL,
  `bloodGroup` varchar(255) NOT NULL,
  `className` varchar(255) NOT NULL,
  `createdDt` datetime DEFAULT NULL,
  `email` varchar(255) DEFAULT NULL,
  `emergencyContactNumber` varchar(255) NOT NULL,
  `gender` varchar(255) DEFAULT NULL,
  `name` varchar(255) NOT NULL,
  `updateDt` datetime DEFAULT NULL,
  `classId` int NOT NULL,
  PRIMARY KEY (`id`),
  KEY `students_ibfk_1` (`classId`),
  CONSTRAINT `students_ibfk_1` FOREIGN KEY (`classId`) REFERENCES `class` (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci

INSERT INTO `student` ('id', 'address', 'age', 'bloodGroup', 'className', 'createdDt', 'email',
'emergencyContactNumber', 'gender', 'name', 'updateDt', 'classId') VALUES
(1, 'At Hyderabad', 16, 'A+', '10th Standard', '2022-02-12 21:36:15', 'sai@gmail.com',
'1234567890', 'male', 'Sai Kishore', '2022-02-12 21:36:15', 12),
(2, 'At Hyderabad', 16, 'B+', '10th Standard', '2022-02-12 21:37:24', 'sandeep@gmail.com',
'1478523690', 'male', 'Sandeep L', '2022-02-12 21:37:24', 12),
(3, 'At Hyderabad', 16, 'AB+', '10th Standard', '2022-02-12 21:38:15', 'david@gmail.com',
'8523697410', 'male', 'David B', '2022-02-12 21:38:15', 12),
(4, 'At Hyderabad', 16, 'AB-', '10th Standard', '2022-02-12 21:39:13', 'aakash@gmail.com',
'5236987410', 'male', 'Aakash C', '2022-02-12 21:39:13', 12);


CREATE TABLE `subject` (
  `id` int NOT NULL AUTO_INCREMENT,
  `createdDt` datetime DEFAULT NULL,
  `subjectDescription` varchar(255) DEFAULT NULL,
  `subjectName` varchar(255) NOT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `subjectName` (`subjectName`)
) ENGINE=InnoDB AUTO_INCREMENT=7 DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_0900_ai_ci

INSERT INTO `subject` (`id`, `createdDt`, `subjectDescription`, `subjectName`) VALUES
(7, '2022-02-12 21:44:52', 'Mathematics', 'Math'),
(8, '2022-02-12 21:45:12', 'Chemistry subject', 'Chemistry'),
(9, '2022-02-12 21:45:31', 'Biology sub', 'Biology'),
(10, '2022-02-12 21:45:48', 'Physics subject', 'Physics');


CREATE TABLE `class_subject_mapping` (
  `id` int NOT NULL AUTO_INCREMENT,
  `classId` int NOT NULL,
  `className` varchar(255) DEFAULT NULL,
  `createdDt` datetime DEFAULT NULL,
  `subjectId` int NOT NULL,
  `subjectName` varchar(255) DEFAULT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `classId` (`classId`,`subjectId`),
  KEY `subjectId` (`subjectId`),
  CONSTRAINT `classes_subjects_mapping_ibfk_1` FOREIGN KEY (`classId`) REFERENCES
`class` (`id`),
  CONSTRAINT `classes_subjects_mapping_ibfk_2` FOREIGN KEY (`subjectId`)
REFERENCES `subject` (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci


INSERT INTO `class_subject_mapping` (`id`, `classId`, `className`, `createdDt`, `subjectId`,
`subjectName`) VALUES
(1, 12, '10th Standard', '2022-02-12 21:49:32', 10, 'Physics'),
(2, 12, '10th Standard', '2022-02-12 21:49:39', 7, 'Math'),
(3, 12, '10th Standard', '2022-02-12 21:49:46', 8, 'Chemistry'),
(4, 12, '10th Standard', '2022-02-12 21:50:01', 9, 'Biology');



CREATE TABLE `teacher` (
  `id` int NOT NULL AUTO_INCREMENT,
  `age` int NOT NULL,
  `contactNumber` varchar(255) NOT NULL,
  `createdDt` datetime DEFAULT NULL,
  `emailId` varchar(255) NOT NULL,
  `firstName` varchar(255) NOT NULL,
  `gender` varchar(255) DEFAULT NULL,
  `lastName` varchar(255) DEFAULT NULL,
  `martialStatus` varchar(255) DEFAULT NULL,
  `qualification` varchar(255) NOT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `firstName` (`firstName`,`contactNumber`,`emailId`)
) ENGINE=InnoDB AUTO_INCREMENT=4 DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_0900_ai_ci

INSERT INTO `teacher` ('id', 'age', 'contactNumber', 'createdDt', 'emailId', 'firstName', 'gender',
'lastName', 'martialStatus', 'qualification') VALUES
(4, '35', '1023045687', '2022-02-12 21:46:40', 'raghav@gmail.com', 'Raghavendra', 'male', 'G',
'Married', 'BE'),
(5, '27', '7531594862', '2022-02-12 21:47:21', 'monali@gmail.com', 'Monali', 'female', 'D',
'Married', 'BE'),
(6, '30', '2631598745', '2022-02-12 21:48:15', 'amit@gmail.com', 'Amit', 'male', 'D', 'Married',
'BE'),
(7, '32', '1598475203', '2022-02-12 21:48:51', 'virat@gmail.com', 'Virat', 'male', 'K', 'Single', 'BE');

CREATE TABLE `teacher_class_subject_mapping` (
  `id` int NOT NULL AUTO_INCREMENT,
  `classId` int NOT NULL,
  `className` varchar(255) DEFAULT NULL,
  `createdDt` datetime DEFAULT NULL,
  `subjectId` int NOT NULL,
  `subjectName` varchar(255) DEFAULT NULL,
  `teacherId` int NOT NULL,
  `teacherName` varchar(255) DEFAULT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `teacherId` (`teacherId`,`classId`,`subjectId`),
  KEY `teachers_classes_subjects_mapping_ibfk_1` (`classId`),
  KEY `teachers_classes_subjects_mapping_ibfk_2` (`subjectId`),
  CONSTRAINT `teachers_classes_subjects_mapping_ibfk_1` FOREIGN KEY (`classId`)
REFERENCES `class` (`id`),
  CONSTRAINT `teachers_classes_subjects_mapping_ibfk_2` FOREIGN KEY (`subjectId`)
REFERENCES `subject` (`id`),
  CONSTRAINT `teachers_classes_subjects_mapping_ibfk_3` FOREIGN KEY (`teacherId`)
REFERENCES `teacher` (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci


INSERT INTO 'teacher_class_subject_mapping' ('id', 'classId', 'className', 'createdDt',
'subjectId', 'subjectName', 'teacherId', 'teacherName') VALUES
(1, 12, '10th Standard', '2022-02-12 21:50:16', 10, 'Physics', 4, 'Raghavendra'),
(2, 12, '10th Standard', '2022-02-12 21:50:26', 8, 'Chemistry', 5, 'Monali'),
(3, 12, '10th Standard', '2022-02-12 21:50:33', 7, 'Math', 6, 'Amit'),
(4, 12, '10th Standard', '2022-02-12 21:50:44', 8, 'Chemistry', 7, 'Virat');
```

Step 4: Creating an Entity classes

- In the Project Explorer, expand **LearnersAcademyAdministrativePortal/JavaSource**
- Right click on **JavaSource** and choose **New->Servlet**
- In **Package**, enter com.learnersacademy.entity and in **Name** enter ClassRoom and click on **Finish**
- Enter the following code:

```
package com.learnersacademy.entity;
import java.util.Date;
import java.util.List;
import javax.persistence.CascadeType;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.JoinColumn;
import javax.persistence.JoinTable;
import javax.persistence.ManyToMany;
import javax.persistence.OneToMany;
import javax.persistence.Table;
import org.hibernate.annotations.CreationTimestamp;

@Entity
@Table(name = "class")
public class ClassRoom {

@Id
@GeneratedValue(strategy = GenerationType.IDENTITY)
private int id;
private String className;
private String sectionName;
private int totalNumberOfStudents;
private String roomNo;
private String classTeacherName;
@CreationTimestamp
private Date createdDt;

public int getId() {
return id;
}
public String getClassName() {
return className;
}
public void setClassName(String className) {
this.className = className;
}

public String getSectionName() {
return sectionName;
}
public void setSectionName(String sectionName) {
this.sectionName = sectionName;
}
public int getTotalNumberOfStudents() {
return totalNumberOfStudents;
}
public void setTotalNumberOfStudents(int totalNumberOfStudents) {
this.totalNumberOfStudents = totalNumberOfStudents;
}
public String getRoomNo() {
return roomNo;
}
        public void setRoomNo(String roomNo) {
this.roomNo = roomNo;
}
public String getClassTeacherName() {
return classTeacherName;
}
public void setClassTeacherName(String classTeacherName) {
this.classTeacherName = classTeacherName;
}
public Date getCreatedDt() {
return createdDt;
}
public void setCreatedDt(Date createdDt) {
this.createdDt = createdDt;
}
public ClassRoom() {
super();
}
public ClassRoom(String className, String sectionName, int totalNumberOfStudents, String
roomNo,
String classTeacherName) {
super();
this.className = className;
this.sectionName = sectionName;
this.totalNumberOfStudents = totalNumberOfStudents;
this.roomNo = roomNo;
this.classTeacherName = classTeacherName;
}
}
```

- In **Package**, enter com.learnersacademy.entity and in **Name** enter TeacherClassSubjectMapping and click on **Finish**

```
package com.learnersacademy.entity;
```

```
import java.util.Date;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Table;
import org.hibernate.annotations.CreationTimestamp;

@Entity
@Table(name = "teacher_class_subject_mapping")
public class TeacherClassSubjectMapping {

@Id
@GeneratedValue(strategy = GenerationType.IDENTITY)
private int id;
private int classId;
private String className;
private int subjectId;
private String subjectName;
private int teacherId;
private String teacherName;
@CreationTimestamp
private Date createdDt;

public int getId() {
return id;
}
public void setId(int id) {
this.id = id;
}
public int getClassId() {
return classId;
}
public void setClassId(int classId) {
this.classId = classId;
}
public String getClassName() {
return className;
}
public void setClassName(String className) {
this.className = className;
```

```
}
public int getSubjectId() {
return subjectId;
}
public void setSubjectId(int subjectId) {
this.subjectId = subjectId;
}
public String getSubjectName() {
return subjectName;
}
public void setSubjectName(String subjectName) {
this.subjectName = subjectName;
}
public int getTeacherId() {
return teacherId;
}
public void setTeacherId(int teacherId) {
this.teacherId = teacherId;
}
public String getTeacherName() {
return teacherName;
}
public void setTeacherName(String teacherName) {
this.teacherName = teacherName;
}
public Date getCreatedDt() {
return createdDt;
}
public void setCreatedDt(Date createdDt) {
this.createdDt = createdDt;
}
public TeacherClassSubjectMapping() {
super();
}
public TeacherClassSubjectMapping(int classId, String className, int subjectId, String subjectName,
int teacherId,
String teacherName) {
super();
this.classId = classId;
this.className = className;
this.subjectId = subjectId;
this.subjectName = subjectName;
this.teacherId = teacherId;
this.teacherName = teacherName;
}
@Override
public String toString() {
return "TeacherClassSubjectMapping [teacherId=" + teacherId + ", teacherName=" + teacherName +
", subjectId=" + subjectId+ ", subjectName=" + subjectName + "]";
}

}
```

- In **Package**, enter com.learnersacademy.entity and in **Name** enter ClassSubjectMapping and click on **Finish**

```
package com.learnersacademy.entity;
import java.util.Date;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Table;

import org.hibernate.annotations.CreationTimestamp;

@Entity
@Table(name = "class_subject_mapping")
public class ClassSubjectMapping {

@Id
@GeneratedValue(strategy = GenerationType.IDENTITY)
private int id;
private int classId;
private String className;
private int subjectId;
private String subjectName;
@CreationTimestamp
private Date createdDt;

public int getId() {
return id;
}
public int getClassId() {
return classId;
}
public void setClassId(int classId) {
this.classId = classId;
}
public String getClassName() {
return className;
}
public void setClassName(String className) {
this.className = className;
}
public int getSubjectId() {
return subjectId;
}
public void setSubjectId(int subjectId) {
this.subjectId = subjectId;
}

public String getSubjectName() {
return subjectName;
}
public void setSubjectName(String subjectName) {
this.subjectName = subjectName;
}
public Date getCreatedDt() {
return createdDt;
}
public void setCreatedDt(Date createdDt) {
this.createdDt = createdDt;
}
public ClassSubjectMapping() {
super();
}
public ClassSubjectMapping(int classId, String className, int subjectId, String subjectName) {
super();
this.classId = classId;
this.className = className;
this.subjectId = subjectId;
this.subjectName = subjectName;
}

}
```

- In **Package**, enter com.learnersacademy.entity and in **Name** enter Student and click on **Finish**

```
package com.learnersacademy.entity;
```

```
import java.util.Date;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Table;
import org.hibernate.annotations.CreationTimestamp;
import org.hibernate.annotations.UpdateTimestamp;

@Entity
@Table(name = "student")
public class Student {

@Id
@GeneratedValue(strategy = GenerationType.IDENTITY)
private int id;
private String name;
private String email;
private String emergencyContactNumber;
private String bloodGroup;
private String gender;
private int age;
private int classId;
private String className;
private String address;
@CreationTimestamp
private Date createdDt;
@UpdateTimestamp
private Date updateDt;

public int getId() {
return id;
}
public String getName() {
return name;
}
public void setName(String name) {
this.name = name;
}
public String getEmail() {
return email;
```

```
}
public void setEmail(String email) {
this.email = email;
}
public String getEmergencyContactNumber() {
return emergencyContactNumber;
}
public void setEmergencyContactNumber(String emergencyContactNumber) {
this.emergencyContactNumber = emergencyContactNumber;
}
public String getBloodGroup() {
return bloodGroup;
}
public void setBloodGroup(String bloodGroup) {
this.bloodGroup = bloodGroup;
}
public String getGender() {
return gender;
}
public void setGender(String gender) {
this.gender = gender;
}
public int getAge() {
return age;
}
public void setAge(int age) {
this.age = age;
}
public int getClassId() {
return classId;
}
public void setClassId(int classId) {
this.classId = classId;
}
public String getClassName() {
return className;
}
public void setClassName(String className) {
this.className = className;
}
public String getAddress() {
```

```
return address;
}
public void setAddress(String address) {
this.address = address;
}
public Date getCreatedDt() {
return createdDt;
}
public void setCreatedDt(Date createdDt) {
this.createdDt = createdDt;
}
public void setId(int id) {
this.id = id;
}
public Date getUpdateDt() {
return updateDt;
}
public void setUpdateDt(Date updateDt) {
this.updateDt = updateDt;
}
public Student() {
super();
}
public Student(String name, String email, String emergencyContactNumber, String bloodGroup,
String gender, int age,
int classId, String className, String address) {
super();
this.name = name;
this.email = email;
this.emergencyContactNumber = emergencyContactNumber;
this.bloodGroup = bloodGroup;
this.gender = gender;
this.age = age;
this.classId = classId;
this.className = className;
this.address = address;
}
}
```

- In **Package**, enter com.learnersacademy.entity and in **Name** enter Subject and click on **Finish**

```
package com.learnersacademy.entity;
import java.util.Date;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Table;
import org.hibernate.annotations.CreationTimestamp;

@Entity
@Table(name = "subject")
public class Subject {

@Id
@GeneratedValue(strategy = GenerationType.IDENTITY)
private int id;
   private String subjectName;
   private String subjectDescription;
   @CreationTimestamp
   private Date createdDt;
public int getId() {
return id;
}
public String getSubjectName() {
return subjectName;
}
public void setSubjectName(String subjectName) {
this.subjectName = subjectName;
}
public String getSubjectDescription() {
return subjectDescription;
}
public void setSubjectDescription(String subjectDescription) {
this.subjectDescription = subjectDescription;
}
public Date getCreatedDt() {
return createdDt;
}
public void setCreatedDt(Date createdDt) {
this.createdDt = createdDt;
}
public Subject() {
super();
}
public Subject(String subjectName, String subjectDescription) {
super();
this.subjectName = subjectName;
this.subjectDescription = subjectDescription;
}
@Override
public String toString() {
return "Subject [id=" + id + ", subjectName=" + subjectName + ", createdDt=" + createdDt + "]";
}}
```

- In **Package**, enter com.learnersacademy.entity and in **Name** enter Teacher and click on **Finish**

```
package com.learnersacademy.entity;
import java.util.Date;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Table;
import org.hibernate.annotations.CreationTimestamp;

@Entity
@Table(name = "teacher")
public class Teacher {

@Id
@GeneratedValue(strategy = GenerationType.IDENTITY)
private int id;
private String firstName;
   private String lastName;
   private String contactNumber;
   private String emailId;
   private String qualification;
   private String gender;
   private int age;
   private String martialStatus;
   @CreationTimestamp
```

```
    private Date createdDt;
public int getId() {
return 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 getContactNumber() {
return contactNumber;
}
public void setContactNumber(String contactNumber) {
this.contactNumber = contactNumber;
}
public String getEmailId() {
return emailId;
}
public void setEmailId(String emailId) {
this.emailId = emailId;
}
public String getQualification() {
return qualification;
}
public void setQualification(String qualification) {
this.qualification = qualification;
}
public String getGender() {
return gender;
}
public void setGender(String gender) {
this.gender = gender;
}
public int getAge() {
```

```
return age;
}
public void setAge(int age) {
this.age = age;
}
public String getMartialStatus() {
return martialStatus;
}
public void setMartialStatus(String martialStatus) {
this.martialStatus = martialStatus;
}
public Date getCreatedDt() {
return createdDt;
}
public void setCreatedDt(Date createdDt) {
this.createdDt = createdDt;
}
public Teacher() {
super();
}
public Teacher(String firstName, String lastName, String contactNumber, String emailId, String
qualification,
String gender, int age, String martialStatus) {
super();
this.firstName = firstName;
this.lastName = lastName;
this.contactNumber = contactNumber;
this.emailId = emailId;
this.qualification = qualification;
this.gender = gender;
this.age = age;
this.martialStatus = martialStatus;
}
  }
```

Step 5: Creating a servlet

- In the Project Explorer, expand **LearnersAcademyAdministrativePortal/JavaSource**
- Right click on **JavaSource** and choose **New->Servlet**

- In **Package**, enter com.learnersacademy.servlets and in **Name** enter ClassReportDetailsServlet and click on **Finish**
- Enter the following code:

```
package com.learnersacademy.servlets;

import java.io.IOException;
import java.util.List;

import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;

import com.learnersacademy.dao.ClassReportDetailsDao;
import com.learnersacademy.entity.Student;
import com.learnersacademy.entity.TeacherClassSubjectMapping;

/**
* Servlet implementation class ClassReportDetailsServlet
*/
public class ClassReportDetailsServlet extends HttpServlet {
private static final long serialVersionUID = 1L;

private ClassReportDetailsDao classReportDetailsDao;

public ClassReportDetailsServlet() {
super();
}

public void init() {
classReportDetailsDao = new ClassReportDetailsDao();
}
protected void doGet(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {

String action = request.getParameter("action");
try {
switch (action) {
```

```
case "classReportDetails":
generateClassReportDetails(request, response);
break;
}
} catch (Exception e) {
e.printStackTrace();
}
}

private void generateClassReportDetails(HttpServletRequest request, HttpServletResponse
response) {
String classId = request.getParameter("classId");
List<TeacherClassSubjectMapping> teacherClassSubjectMappings =
classReportDetailsDao.getTeacherClassSubjectMappingsDetails(Integer.parseInt(classId));
List<Student> studentDetails = classReportDetailsDao.getStudentDetails(Integer.parseInt(classId));
try {
HttpSession session = request.getSession();
session.setAttribute("teacherClassSubjectMappings", teacherClassSubjectMappings);
session.setAttribute("studentDetails", studentDetails);
RequestDispatcher dispatcher = request.getRequestDispatcher("pages/listClassDetailsReport.jsp");
dispatcher.forward(request, response);
} catch (Exception e) {
e.printStackTrace();
}
}

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

}
```

- In **Package**, enter com.learnersacademy.servlets and in **Name** enter ClassRoomServlet and click on **Finish**
- Enter the following code:

```
package com.learnersacademy.servlets;
import java.io.IOException;
import java.util.List;
import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import com.learnersacademy.dao.ClassRoomDao;
import com.learnersacademy.entity.ClassRoom;
import net.sf.json.JSONObject;
public class ClassRoomServlet extends HttpServlet {
private static final long serialVersionUID = 1L;

private ClassRoomDao classRoomDao;

public ClassRoomServlet() {
super();
}
public void init() {
classRoomDao = new ClassRoomDao();
}
private ClassRoom getClassRoom(HttpServletRequest request, HttpServletResponse response) {
String classRoomId = request.getParameter("id");
ClassRoom classRoom = classRoomDao.getClassRoom(Integer.parseInt(classRoomId));
HttpSession session = request.getSession();
session.setAttribute("classRoom", classRoom);
return classRoom;
}
private List<ClassRoom> getClassRooms(HttpServletRequest request, HttpServletResponse
response, boolean delFlag) {
List<ClassRoom> classRooms = classRoomDao.getAllClassRooms();
String reportFlag = request.getParameter("reportFlag");
try {
HttpSession session = request.getSession();
session.setAttribute("classRooms", classRooms);
session.setAttribute("delFlag", delFlag);
if("Y".equals(reportFlag)) {
RequestDispatcher dispatcher = request.getRequestDispatcher("pages/listClassReport.jsp");
dispatcher.forward(request, response);
} else {
RequestDispatcher dispatcher = request.getRequestDispatcher("pages/list-classRooms.jsp");
dispatcher.forward(request, response);
}
} catch (Exception e) {
```

```
e.printStackTrace();
}
return classRooms;
}

public List<ClassRoom> getAllClassDetails(HttpServletRequest request, HttpServletResponse
response) throws IOException {
List<ClassRoom> classRooms = classRoomDao.getAllClassRooms();
String flag = request.getParameter("servletName");
try {
HttpSession session = request.getSession();
session.setAttribute("classRooms", classRooms);
if(!"mappingServlet".equals(flag)) {
RequestDispatcher dispatcher = request.getRequestDispatcher("pages/addStudent.jsp");
dispatcher.forward(request, response);
}

} catch (Exception e) {
e.printStackTrace();
}
return classRooms;
}

private ClassRoom createClassRoom(HttpServletRequest request, HttpServletResponse response) {
String className = request.getParameter("className");
String sectionName = request.getParameter("sectionName");
int totalNumberOfStudents = Integer.parseInt(request.getParameter("totalNumberOfStudents"));
String roomNo = request.getParameter("roomNo");
String classTeacherName = request.getParameter("classTeacherName");

ClassRoom classRoomModel = new ClassRoom(className, sectionName, totalNumberOfStudents,
roomNo, classTeacherName);
ClassRoom newClassRoom = classRoomDao.saveClassRoom(classRoomModel);
getClassRooms(request, response, false);
return newClassRoom;
}

private void deleteClass(HttpServletRequest request, HttpServletResponse response) {
int classId = Integer.parseInt(request.getParameter("id"));
classRoomDao.deleteClass(classId);
getClassRooms(request, response, true);
```

```
}

private void redirectToAddJsp(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
RequestDispatcher dispatcher = request.getRequestDispatcher("pages/addClass.jsp");
dispatcher.forward(request, response);
}

protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
String action = request.getParameter("action");
try {
switch (action) {

case "new":
createClassRoom(request, response);
break;

case "list":
getClassRooms(request, response, false);
break;

case "delete":
deleteClass(request, response);
break;

case "listClassName":
getAllClassDetails(request, response);
break;

case "classById":
getClassRoom(request, response);
break;

case "addJsp":
redirectToAddJsp(request, response);
break;

}
} catch (Exception e) {
e.printStackTrace();
}
}


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

}
```

- In **Package**, enter com.learnersacademy.servlets and in **Name** enter ClassSubjectMappingServlet and click on **Finish**
- Enter the following code:

```
package com.learnersacademy.servlets;

import java.io.IOException;
import java.util.List;

import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;

import com.learnersacademy.dao.ClassSubjectMappingDao;
import com.learnersacademy.entity.ClassRoom;
import com.learnersacademy.entity.ClassSubjectMapping;
import com.learnersacademy.entity.Student;
import com.learnersacademy.entity.Subject;


/**
* Servlet implementation class ClassSubjectMappingServlet
*/
public class ClassSubjectMappingServlet extends HttpServlet {
private static final long serialVersionUID = 1L;

private ClassSubjectMappingDao classSubjectMappingDao;
```

```
public ClassSubjectMappingServlet() {
super();
}

public void init() {
classSubjectMappingDao = new ClassSubjectMappingDao();
}

private List<ClassSubjectMapping> getClassSubjectMappings(HttpServletRequest request,
HttpServletResponse response, boolean delFlag) {
List<ClassSubjectMapping> classSubjectMappings =
classSubjectMappingDao.getAllClassSubjectMapping();
try {
HttpSession session = request.getSession();
session.setAttribute("classSubjectMappings", classSubjectMappings);
session.setAttribute("delFlag", delFlag);
RequestDispatcher dispatcher =
request.getRequestDispatcher("pages/listClassSubjectMapping.jsp");
dispatcher.forward(request, response);
} catch (Exception e) {
e.printStackTrace();
}
return classSubjectMappings;
}

private ClassSubjectMapping createClassSubjectMapping(HttpServletRequest request,
HttpServletResponse response) {
int classId = Integer.parseInt(request.getParameter("classesNameCombo"));
int subjectId = Integer.parseInt(request.getParameter("subjectsNameCombo"));
String className = request.getParameter("className");
String subjectName = request.getParameter("subjectName");

ClassSubjectMapping mappingModel = new ClassSubjectMapping(classId, className, subjectId,
subjectName);
ClassSubjectMapping newMapping =
classSubjectMappingDao.saveClassSubjectMapping(mappingModel);
getClassSubjectMappings(request, response, false);
return newMapping;
}
```

```
private void deleteClassSubjectMapping(HttpServletRequest request, HttpServletResponse
response) {
int mappingId = Integer.parseInt(request.getParameter("id"));
classSubjectMappingDao.deleteClassSubjectMapping(mappingId);
getClassSubjectMappings(request, response, true);
}

private void getSubjectClassDetails(HttpServletRequest request, HttpServletResponse response) {
RequestDispatcher dispatcher;
try {
dispatcher =
request.getRequestDispatcher("classRoom?action=listClassName&servletName=mappingServlet");
dispatcher.include(request, response);
HttpSession session = request.getSession(false);
List<ClassRoom> classRooms=  (List<ClassRoom>) session.getAttribute("classRooms");

dispatcher = request.getRequestDispatcher("subject?action=list&servletName=mappingServlet");
dispatcher.include(request, response);
List<Subject> subjects=  (List<Subject>) session.getAttribute("subjects");

dispatcher = request.getRequestDispatcher("pages/addClassSubjectMapping.jsp");
dispatcher.forward(request, response);
} catch (ServletException e) {
e.printStackTrace();
} catch (IOException e) {
e.printStackTrace();
}
}

protected void doGet(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
String action = request.getParameter("action");
try {
switch (action) {

case "new":
createClassSubjectMapping(request, response);
break;

case "list":
getClassSubjectMappings(request, response, false);
break;

case "delete":
deleteClassSubjectMapping(request, response);
break;
case "mappingDetails":
getSubjectClassDetails(request, response);
break;
}
} catch (Exception e) {
e.printStackTrace();
}
}

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

}
```

- In **Package**, enter com.learnersacademy.servlets and in **Name** enter LoginServlet and click on **Finish**
- Enter the following code:

```
package com.learnersacademy.servlets;

import java.io.IOException;

import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;


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


public LoginServlet() {
super();
}


protected void doGet(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
String username = request.getParameter("username");
String password = request.getParameter("password");

if(username.equals("admin") && password.equals("admin@123")) {
request.getSession().setAttribute("loggedInUser", "admin");
RequestDispatcher dispatcher = request.getRequestDispatcher("pages/dashboard.jsp");
dispatcher.forward(request, response);
} else {
RequestDispatcher dispatcher = request.getRequestDispatcher("pages/error.jsp");
dispatcher.forward(request, response);
}
}

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

}
```

- In **Package**, enter com.learnersacademy.servlets and in **Name** enter LogoutServlet and click on **Finish**
- Enter the following code:

```
package com.learnersacademy.servlets;

import java.io.IOException;

import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

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

```
    public LogoutServlet() {
        super();
    }

protected void doGet(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
RequestDispatcher dispatcher = request.getRequestDispatcher("pages/login.jsp");
dispatcher.forward(request, response);
}

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

}
```

- In **Package**, enter com.learnersacademy.servlets and in **Name** enter StudentServlet and click on **Finish**
- Enter the following code:

```
package com.learnersacademy.servlets;

import java.io.IOException;
import java.util.Date;
import java.util.List;

import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;

import com.learnersacademy.dao.StudentDao;
import com.learnersacademy.entity.ClassRoom;
import com.learnersacademy.entity.Student;

/**
* Servlet implementation class StudentServlet
*/
public class StudentServlet extends HttpServlet {
```

```
private static final long serialVersionUID = 1L;

private StudentDao studentDao;

public StudentServlet() {
super();
}

public void init() {
studentDao = new StudentDao();
}

private Student getStudent(HttpServletRequest request, HttpServletResponse response) {
String studentId = request.getParameter("id");
Student student = studentDao.getStudent(Integer.parseInt(studentId));
return student;
}

private List<Student> getStudents(HttpServletRequest request, HttpServletResponse response,
boolean delFlag) {
List<Student> students = studentDao.getAllStudents();
try {
HttpSession session = request.getSession();
session.setAttribute("students", students);
session.setAttribute("delFlag", delFlag);
RequestDispatcher dispatcher = request.getRequestDispatcher("pages/list-students.jsp");
dispatcher.forward(request, response);
} catch (Exception e) {
e.printStackTrace();
}
return students;
}

private Student createStudent(HttpServletRequest request, HttpServletResponse response) {
String name = request.getParameter("studentName");
String email = request.getParameter("email");
String emergencyContactNumber = request.getParameter("emergencyContactNumber");
String bloodGroup = request.getParameter("bloodGroup");
String gender = request.getParameter("gender");
int age = Integer.parseInt(request.getParameter("age"));
int classId = Integer.parseInt(request.getParameter("classesNameCombo"));
```

```
String address = request.getParameter("address");
Student newStudent = null;
try {
RequestDispatcher dispatcher =
request.getRequestDispatcher("classRoom?action=classById&id="+classId);
dispatcher.include(request, response);
HttpSession session = request.getSession(false);
ClassRoom classRoom = (ClassRoom) session.getAttribute("classRoom");
String className = classRoom.getClassName() != null
&& !classRoom.getClassName().isEmpty()?classRoom.getClassName():"";
Student studentModel = new Student(name, email, emergencyContactNumber, bloodGroup, gender,
age, classId, className, address);
newStudent = studentDao.saveStudent(studentModel);
getStudents(request, response, false);
} catch (ServletException e) {
e.printStackTrace();
} catch (IOException e) {
e.printStackTrace();
}
return newStudent;
}

private void deleteStudent(HttpServletRequest request, HttpServletResponse response) {
int studentId = Integer.parseInt(request.getParameter("id"));
studentDao.deleteStudents(studentId);
getStudents(request, response, true);
}


protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
String action = request.getParameter("action");
try {
switch (action) {

case "new":
createStudent(request, response);
break;

case "list":
getStudents(request, response, false);
break;

case "delete":
deleteStudent(request, response);
break;
}
} catch (Exception e) {
e.printStackTrace();
}
}

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

}
```

- In **Package**, enter com.learnersacademy.servlets and in **Name** enter SubjectServlet and click on **Finish**
- Enter the following code:

```
package com.learnersacademy.servlets;

import java.io.IOException;
import java.util.List;

import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;

import com.learnersacademy.dao.SubjectDao;
import com.learnersacademy.entity.Subject;


/**
* Servlet implementation class SubjectServlet
*/
public class SubjectServlet extends HttpServlet {
```

```
private static final long serialVersionUID = 1L;

private SubjectDao subjectDao;

public SubjectServlet() {
super();
}

public void init() {
subjectDao = new SubjectDao();
}

private Subject getSubject(HttpServletRequest request, HttpServletResponse response) {
String subjectId = request.getParameter("id");
Subject subject = subjectDao.getSubject(Integer.parseInt(subjectId));
return subject;
}

private List<Subject> getSubjects(HttpServletRequest request, HttpServletResponse response,
boolean delFlag) {
List<Subject> subjects = subjectDao.getAllSubjects();
String flag = request.getParameter("servletName");
try {
HttpSession session = request.getSession();
session.setAttribute("subjects", subjects);
session.setAttribute("delFlag", delFlag);
if(!"mappingServlet".equals(flag)) {
RequestDispatcher dispatcher = request.getRequestDispatcher("pages/listSubjects.jsp");
dispatcher.forward(request, response);
}
} catch (Exception e) {
e.printStackTrace();
}
return subjects;
}

private Subject createSubject(HttpServletRequest request, HttpServletResponse response) {
String subjectName = request.getParameter("subjectName");
String subjectDescription = request.getParameter("subjectDescription");

Subject subjectModel = new Subject(subjectName, subjectDescription);
Subject newSubject = subjectDao.saveSubject(subjectModel);
getSubjects(request, response, false);
return newSubject;
}

private void deleteSubject(HttpServletRequest request, HttpServletResponse response) {
int subjectId = Integer.parseInt(request.getParameter("id"));
subjectDao.deleteSubjects(subjectId);
getSubjects(request, response, true);
}

private void redirectToAddJsp(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
RequestDispatcher dispatcher = request.getRequestDispatcher("pages/addSubject.jsp");
dispatcher.forward(request, response);
}

protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
String action = request.getParameter("action");
try {
switch (action) {

case "new":
createSubject(request, response);
break;

case "list":
getSubjects(request, response, false);
break;

case "delete":
deleteSubject(request, response);
break;

case "addJsp":
redirectToAddJsp(request, response);
break;
}
} catch (Exception e) {
e.printStackTrace();
}
}

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

}
```

- In **Package**, enter com.learnersacademy.servlets and in **Name** enter TeacherClassSubjectMappingServlet and click on **Finish**
- Enter the following code:

```
package com.learnersacademy.servlets;
import java.io.IOException;
import java.util.List;
import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import com.learnersacademy.dao.TeacherClassSubjectMappingDao;
import com.learnersacademy.entity.ClassRoom;
import com.learnersacademy.entity.Subject;
import com.learnersacademy.entity.Teacher;
import com.learnersacademy.entity.TeacherClassSubjectMapping;

public class TeacherClassSubjectMappingServlet extends HttpServlet {
private static final long serialVersionUID = 1L;
private TeacherClassSubjectMappingDao teacherClassSubjectMappingDao;
public TeacherClassSubjectMappingServlet() {
super();
}
public void init() {
teacherClassSubjectMappingDao = new TeacherClassSubjectMappingDao();
}
private List<TeacherClassSubjectMapping> getTeacherClassSubjectMappings(HttpServletRequest
request, HttpServletResponse response, boolean delFlag) {
```

```
List<TeacherClassSubjectMapping> teacherClassSubjectMappings =
teacherClassSubjectMappingDao.getAllTeacherClassSubjectMapping();
try {
HttpSession session = request.getSession();
session.setAttribute("teacherClassSubjectMappings", teacherClassSubjectMappings);
session.setAttribute("delFlag", delFlag);
RequestDispatcher dispatcher =
request.getRequestDispatcher("pages/listTeacherClassSubjectMapping.jsp");
dispatcher.forward(request, response);
} catch (Exception e) {
e.printStackTrace();
}
return teacherClassSubjectMappings;
}
private TeacherClassSubjectMapping createTeacherClassSubjectMapping(HttpServletRequest
request, HttpServletResponse response) {
int classId = Integer.parseInt(request.getParameter("classesNameCombo"));
String className = request.getParameter("className");
int subjectId = Integer.parseInt(request.getParameter("subjectsNameCombo"));
String subjectName = request.getParameter("subjectName");
int teacherId = Integer.parseInt(request.getParameter("teachersNameCombo"));
String teacherName = request.getParameter("teacherName");
TeacherClassSubjectMapping mappingModel = new TeacherClassSubjectMapping(classId,
className, subjectId, subjectName, teacherId, teacherName);
TeacherClassSubjectMapping newMapping =
teacherClassSubjectMappingDao.saveTeacherClassSubjectMapping(mappingModel);
getTeacherClassSubjectMappings(request, response, false);
return newMapping;
}
private void deleteTeacherClassSubjectMapping(HttpServletRequest request, HttpServletResponse
response) {
int mappingId = Integer.parseInt(request.getParameter("id"));
teacherClassSubjectMappingDao.deleteTeacherClassSubjectMapping(mappingId);
getTeacherClassSubjectMappings(request, response, true);
}
private void getTeacherSubjectClassDetails(HttpServletRequest request, HttpServletResponse
response) {
RequestDispatcher dispatcher;
try {
dispatcher =
request.getRequestDispatcher("classRoom?action=listClassName&servletName=mappingServlet");
```

```
dispatcher.include(request, response);
HttpSession session = request.getSession(false);

dispatcher = request.getRequestDispatcher("subject?action=list&servletName=mappingServlet");
dispatcher.include(request, response);

dispatcher = request.getRequestDispatcher("teacher?action=list&servletName=mappingServlet");
dispatcher.include(request, response);

dispatcher = request.getRequestDispatcher("pages/addTeacherClassSubjectMapping.jsp");
dispatcher.forward(request, response);
} catch (ServletException e) {
e.printStackTrace();
} catch (IOException e) {
e.printStackTrace();
}
}
protected void doGet(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
String action = request.getParameter("action");
try {
switch (action) {

case "new":
createTeacherClassSubjectMapping(request, response);
break;

case "list":
getTeacherClassSubjectMappings(request, response, false);
break;

case "delete":
deleteTeacherClassSubjectMapping(request, response);
break;

case "mappingDetails":
getTeacherSubjectClassDetails(request, response);
break;
}
} catch (Exception e) {
e.printStackTrace();
```

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

- In **Package**, enter com.learnersacademy.servlets and in **Name** enter TeacherServlet and click on **Finish**
- Enter the following code:

```
package com.learnersacademy.servlets;
import java.io.IOException;
import java.util.List;
import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import com.learnersacademy.dao.TeacherDao;
import com.learnersacademy.entity.Teacher;

public class TeacherServlet extends HttpServlet {
private static final long serialVersionUID = 1L;
private TeacherDao teacherDao;
  public TeacherServlet() {
    super();
  }
public void init() {
teacherDao = new TeacherDao();
}
private Teacher getTeacher(HttpServletRequest request, HttpServletResponse response) {
String teacherId = request.getParameter("id");
Teacher teacher = teacherDao.getTeacher(Integer.parseInt(teacherId));
return teacher;
}
private List<Teacher> getTeachers(HttpServletRequest request, HttpServletResponse response,
boolean delFlag) {
List<Teacher> teachers = teacherDao.getAllTeachers();
```

```
String flag = request.getParameter("servletName");
try {
HttpSession session = request.getSession();
session.setAttribute("teachers", teachers);
session.setAttribute("delFlag", delFlag);
if(!"mappingServlet".equals(flag)) {
RequestDispatcher dispatcher = request.getRequestDispatcher("pages/listTeacher.jsp");
dispatcher.forward(request, response);
}
} catch (Exception e) {
e.printStackTrace();
}
return teachers;
}
private Teacher createTeacher(HttpServletRequest request, HttpServletResponse response) {
String firstName = request.getParameter("firstName");
String lastName = request.getParameter("lastName");
String contactNumber = request.getParameter("contactNumber");
String emailId = request.getParameter("emailAddress");
String qualification = request.getParameter("qualification");
int age = Integer.parseInt(request.getParameter("age"));
String martialStatus = request.getParameter("martialStatus");
String gender = request.getParameter("gender");
Teacher teacherModel = new Teacher(firstName, lastName, contactNumber, emailId, qualification,
gender, age, martialStatus);
Teacher newTeacher = teacherDao.saveTeacher(teacherModel);
getTeachers(request, response, false);
return newTeacher;
}
private void deleteTeacher(HttpServletRequest request, HttpServletResponse response) {
int teacherId = Integer.parseInt(request.getParameter("id"));
teacherDao.deleteTeacher(teacherId);
getTeachers(request, response, true);
}
private void redirectToAddJsp(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
RequestDispatcher dispatcher = request.getRequestDispatcher("pages/addTeacher.jsp");
dispatcher.forward(request, response);
}
protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
```

```
String action = request.getParameter("action");
try {
switch (action) {

case "new":
createTeacher(request, response);
break;

case "list":
getTeachers(request, response, false);
break;

case "delete":
deleteTeacher(request, response);
break;

case "addJsp":
redirectToAddJsp(request, response);
break;
}
} catch (Exception e) {
e.printStackTrace();
}
}
protected void doPost(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
doGet(request, response);
}
}
```

Step 6: Creating a DAO classes

- In the Project Explorer, expand **LearnersAcademyAdministrativePortal/JavaSource**
- Right click on **JavaSource** and choose **New->Class**
- In **Package**, enter com.learnersacademy.dao and in **Name** enter ClassReportDetailsDao and click on **Finish**
- Enter the following code:

```
package com.learnersacademy.dao;
import java.util.ArrayList;
import java.util.Iterator;
import java.util.List;
import org.hibernate.Criteria;
import org.hibernate.Session;
import org.hibernate.Transaction;
import org.hibernate.criterion.Projections;
import org.hibernate.criterion.Restrictions;
import com.learnersacademy.entity.Student;
import com.learnersacademy.entity.TeacherClassSubjectMapping;
import com.learnersacademy.util.HibernateUtil;
public class ClassReportDetailsDao {
public List<TeacherClassSubjectMapping> getTeacherClassSubjectMappingsDetails(int classId) {

Transaction transaction = null;
List<TeacherClassSubjectMapping> getTeachersHandlingSubjectsList = new
ArrayList<TeacherClassSubjectMapping>();

try (Session session = HibernateUtil.getSessionFactory().openSession()) {
transaction = session.beginTransaction();
Criteria teachersHandlingSubjectsCriteria  =
session.createCriteria(TeacherClassSubjectMapping.class);
teachersHandlingSubjectsCriteria.add(Restrictions.eq("classId", classId));
teachersHandlingSubjectsCriteria.setProjection(Projections.distinct(Projections.projectionList()
 .add(Projections.property("teacherId"))
 .add(Projections.property("teacherName"))
 .add(Projections.property("subjectId"))
 .add(Projections.property("subjectName"))
));
List results = teachersHandlingSubjectsCriteria.list();
if(results != null && results.size() > 0) {
Iterator<Object> it = results.iterator();
while(it.hasNext()) {
Object[] row = (Object[]) it.next();
TeacherClassSubjectMapping teachersClassesSubjectsMappingObj = new
TeacherClassSubjectMapping();
teachersClassesSubjectsMappingObj.setTeacherId((Integer) row[0]);
teachersClassesSubjectsMappingObj.setTeacherName((String) row[1]);
teachersClassesSubjectsMappingObj.setSubjectId((Integer) row[2]);
teachersClassesSubjectsMappingObj.setSubjectName((String) row[3]);
getTeachersHandlingSubjectsList.add(teachersClassesSubjectsMappingObj);
}
}
```

```
transaction.commit();
} catch (Exception e) {
if (transaction != null) {
transaction.rollback();
}
e.printStackTrace();
}
return getTeachersHandlingSubjectsList;
}

public List<Student> getStudentDetails(int classId) {

Transaction transaction = null;
List<Student> getStudentInTheClassList = new ArrayList<Student>();

try (Session session = HibernateUtil.getSessionFactory().openSession()) {
transaction = session.beginTransaction();
Criteria studentsCriteria  = session.createCriteria(Student.class);
studentsCriteria.add(Restrictions.eq("classId", classId));
studentsCriteria.setProjection(Projections.distinct(Projections.projectionList()
 .add(Projections.property("id"))
 .add(Projections.property("name"))
 .add(Projections.property("emergencyContactNumber"))
 .add(Projections.property("bloodGroup"))
 .add(Projections.property("gender"))
));
List results = studentsCriteria.list();
if(results != null && results.size() > 0) {
Iterator<Object> it = results.iterator();
while(it.hasNext()) {
Object[] row = (Object[]) it.next();
Student student = new Student();
student.setId((Integer) row[0]);
student.setName((String) row[1]);
student.setEmergencyContactNumber((String) row[2]);
student.setBloodGroup((String) row[3]);
student.setGender((String) row[4]);
getStudentInTheClassList.add(student);
}
}
transaction.commit();
```

```
} catch (Exception e) {
if (transaction != null) {
transaction.rollback();
}
e.printStackTrace();
}
return getStudentInTheClassList;

}

}
```

- In **Package**, enter com.learnersacademy.dao and in **Name** enter ClassRoomDao and click on **Finish**
- Enter the following code:

```
package com.learnersacademy.dao;
import java.util.List;
import org.hibernate.Session;
import org.hibernate.Transaction;
import com.learnersacademy.entity.ClassRoom;
import com.learnersacademy.util.HibernateUtil;
public class ClassRoomDao {
public ClassRoom getClassRoom(int id) {
Transaction transaction = null;
ClassRoom classRoom = null;

try (Session session = HibernateUtil.getSessionFactory().openSession()) {
transaction = session.beginTransaction();
classRoom = session.get(ClassRoom.class, id);
transaction.commit();
} catch (Exception e) {
if (transaction != null) {
transaction.rollback();
}
e.printStackTrace();
}
return classRoom;
}

public ClassRoom saveClassRoom(ClassRoom classRoom) {
Transaction transaction = null;
```

```
ClassRoom createdClassRoom = null;
Session session = null;
try {
session = HibernateUtil.getSessionFactory().openSession();
transaction = session.beginTransaction();
session.save(classRoom);
transaction.commit();
} catch (Exception e) {
if (transaction != null) {
transaction.rollback();
}
e.printStackTrace();
} finally {
session.close();
}
return createdClassRoom;
}

@SuppressWarnings("unchecked")
public List<ClassRoom> getAllClassRooms() {
Transaction transaction = null;
List<ClassRoom> listOfClassRooms = null;
try (Session session = HibernateUtil.getSessionFactory().openSession()) {
transaction = session.beginTransaction();
listOfClassRooms = session.createQuery("from ClassRoom").getResultList();
                    transaction.commit();
} catch (Exception e) {
if (transaction != null) {
transaction.rollback();
}
e.printStackTrace();
}
return listOfClassRooms;
}

public void deleteClass(int id) {
Transaction transaction = null;
Session session = null;
try {
session = HibernateUtil.getSessionFactory().openSession();
transaction = session.beginTransaction();
ClassRoom classRoomObj = session.get(ClassRoom.class, id);
session.delete(classRoomObj);
transaction.commit();
} catch (Exception e) {
if (transaction != null) {
transaction.rollback();
}
e.printStackTrace();
} finally {
session.close();
}
}
}
```

- In **Package**, enter com.learnersacademy.dao and in **Name** enter ClassSubjectMappingDao and click on **Finish**
- Enter the following code:

```
package com.learnersacademy.dao;
import java.util.List;
import org.hibernate.Session;
import org.hibernate.Transaction;
import com.learnersacademy.entity.ClassSubjectMapping;
import com.learnersacademy.util.HibernateUtil;
public class ClassSubjectMappingDao {

public ClassSubjectMapping saveClassSubjectMapping(ClassSubjectMapping classSubjectMapping) {
Transaction transaction = null;
ClassSubjectMapping createdClassSubjectMapping = null;
Session session = null;
try {
session = HibernateUtil.getSessionFactory().openSession();
transaction = session.beginTransaction();
session.save(classSubjectMapping);
transaction.commit();
} catch (Exception e) {
if (transaction != null) {
transaction.rollback();
}
e.printStackTrace();
} finally {
session.close();
}
return createdClassSubjectMapping;
}

@SuppressWarnings("unchecked")
public List<ClassSubjectMapping> getAllClassSubjectMapping() {
Transaction transaction = null;
List<ClassSubjectMapping> listOfClassSubjectMappings = null;
try (Session session = HibernateUtil.getSessionFactory().openSession()) {
transaction = session.beginTransaction();
listOfClassSubjectMappings = session.createQuery("from ClassSubjectMapping").getResultList();
                                        transaction.commit();
} catch (Exception e) {
if (transaction != null) {
transaction.rollback();
}
e.printStackTrace();
}
return listOfClassSubjectMappings;
}

public void deleteClassSubjectMapping(int id) {
Transaction transaction = null;
Session session = null;
try {
session = HibernateUtil.getSessionFactory().openSession();
transaction = session.beginTransaction();
ClassSubjectMapping mappingObj = session.get(ClassSubjectMapping.class, id);
session.delete(mappingObj);
transaction.commit();
} catch (Exception e) {
if (transaction != null) {
transaction.rollback();
}
e.printStackTrace();
} finally {
session.close();
}
}
}
```

- In **Package**, enter com.learnersacademy.dao and in **Name** enter StudentDao and click on **Finish**
- Enter the following code:

```
package com.learnersacademy.dao;
import java.util.List;
import org.hibernate.Session;
import org.hibernate.Transaction;
import com.learnersacademy.entity.Student;
import com.learnersacademy.util.HibernateUtil;
public class StudentDao {
public Student getStudent(int id) {
Transaction transaction = null;
Student student = nul
try (Session session = HibernateUtil.getSessionFactory().openSession()) {
transaction = session.beginTransaction();
student = session.get(Student.class, id);
transaction.commit();
} catch (Exception e) {
if (transaction != null) {
transaction.rollback();
}
e.printStackTrace();
}
return student;
}

public Student saveStudent(Student student) {
Transaction transaction = null;
Student createdStudent = null;
Session session = null;
try {
session = HibernateUtil.getSessionFactory().openSession();
transaction = session.beginTransaction();
session.save(student);
transaction.commit();
} catch (Exception e) {
if (transaction != null) {
transaction.rollback();
}
e.printStackTrace();
} finally {
```

```
session.close();
}
return createdStudent;
}

@SuppressWarnings("unchecked")
public List<Student> getAllStudents() {
Transaction transaction = null;
List<Student> listOfStudents = null;
try (Session session = HibernateUtil.getSessionFactory().openSession()) {
transaction = session.beginTransaction();
listOfStudents = session.createQuery("from Student").getResultList();
                                    transaction.commit();
} catch (Exception e) {
if (transaction != null) {
transaction.rollback();
}
e.printStackTrace();
}
return listOfStudents;
}

public void deleteStudents(int id) {
Transaction transaction = null;
Session session = null;
try {
session = HibernateUtil.getSessionFactory().openSession();
transaction = session.beginTransaction();
Student studentsObj = session.get(Student.class, id);
session.delete(studentsObj);
transaction.commit();
} catch (Exception e) {
if (transaction != null) {
transaction.rollback();
}
e.printStackTrace();
} finally {
session.close();
}
}
}
```

- In **Package**, enter com.learnersacademy.dao and in **Name** enter SubjectDao and click on **Finish**
- Enter the following code:

```
package com.learnersacademy.dao;
import java.util.List;
import org.hibernate.Session;
import org.hibernate.Transaction;
import com.learnersacademy.entity.Subject;
import com.learnersacademy.util.HibernateUtil;
public class SubjectDao {
public Subject getSubject(int id) {
Transaction transaction = null;
Subject subject = null;
try (Session session = HibernateUtil.getSessionFactory().openSession()) {
transaction = session.beginTransaction();
subject = session.get(Subject.class, id);
transaction.commit();
} catch (Exception e) {
if (transaction != null) {
transaction.rollback();
}
e.printStackTrace();
}
return subject;
}

public Subject saveSubject(Subject subject) {
Transaction transaction = null;
Subject createdSubject = null;
Session session = null;
try {
session = HibernateUtil.getSessionFactory().openSession();
transaction = session.beginTransaction();
session.save(subject);
transaction.commit();
} catch (Exception e) {
if (transaction != null) {
transaction.rollback();
}
e.printStackTrace();
} finally {
session.close();
}
return createdSubject;
}

@SuppressWarnings("unchecked")
public List<Subject> getAllSubjects() {
Transaction transaction = null;
List<Subject> listOfSubjects = null;
try (Session session = HibernateUtil.getSessionFactory().openSession()) {
transaction = session.beginTransaction();
listOfSubjects = session.createQuery("from Subject").getResultList();
                                transaction.commit();
} catch (Exception e) {
if (transaction != null) {
transaction.rollback();
}
e.printStackTrace();
}
return listOfSubjects;
}

public Subject updateStudent(Subject subject) {
Transaction transaction = null;
Subject createdSubject = null;
Session session = null;
try {
session = HibernateUtil.getSessionFactory().openSession();
transaction = session.beginTransaction();
session.update(subject);
transaction.commit();
} catch (Exception e) {
if (transaction != null) {
transaction.rollback();
}
e.printStackTrace();
} finally {
session.close();
}
return createdSubject;
```

```
}

public void deleteSubjects(int id) {
Transaction transaction = null;
Session session = null;
try {
session = HibernateUtil.getSessionFactory().openSession();
transaction = session.beginTransaction();
Subject subjectsObj = session.get(Subject.class, id);
session.delete(subjectsObj);
transaction.commit();
} catch (Exception e) {
if (transaction != null) {
transaction.rollback();
}
e.printStackTrace();
} finally {
session.close();
}
}
}
```

- In **Package**, enter com.learnersacademy.dao and in **Name** enter TeacherClassSubjectMappingDao and click on **Finish**
- Enter the following code:

```
package com.learnersacademy.dao;
import java.util.List;
import org.hibernate.Session;
import org.hibernate.Transaction;
import com.learnersacademy.entity.TeacherClassSubjectMapping;
import com.learnersacademy.util.HibernateUtil;
public class TeacherClassSubjectMappingDao {
public TeacherClassSubjectMapping saveTeacherClassSubjectMapping(TeacherClassSubjectMapping
teacherClassSubjectMapping) {
Transaction transaction = null;
TeacherClassSubjectMapping createdTeacherClassSubjectMapping = null;
Session session = null;
try {
session = HibernateUtil.getSessionFactory().openSession();
transaction = session.beginTransaction();
session.save(teacherClassSubjectMapping);
```

```
transaction.commit();
} catch (Exception e) {
if (transaction != null) {
transaction.rollback();
}
e.printStackTrace();
} finally {
session.close();
}
return createdTeacherClassSubjectMapping;
}

@SuppressWarnings("unchecked")
public List<TeacherClassSubjectMapping> getAllTeacherClassSubjectMapping() {
Transaction transaction = null;
List<TeacherClassSubjectMapping> listOfTeacherClassSubjectMappings = null;
try (Session session = HibernateUtil.getSessionFactory().openSession()) {
transaction = session.beginTransaction();
listOfTeacherClassSubjectMappings = session.createQuery("from
TeacherClassSubjectMapping").getResultList();
                                        transaction.commit();
} catch (Exception e) {
if (transaction != null) {
transaction.rollback();
}
e.printStackTrace();
}
return listOfTeacherClassSubjectMappings;
}


public void deleteTeacherClassSubjectMapping(int id) {
Transaction transaction = null;
Session session = null;
try {
session = HibernateUtil.getSessionFactory().openSession();
transaction = session.beginTransaction();
TeacherClassSubjectMapping mappingObj = session.get(TeacherClassSubjectMapping.class, id);
session.delete(mappingObj);
transaction.commit();
} catch (Exception e) {
```

```
if (transaction != null) {
transaction.rollback();
}
e.printStackTrace();
} finally {
session.close();
}
}
}
```

- In **Package**, enter com.learnersacademy.dao and in **Name** enter TeacherDao and click on **Finish**
- Enter the following code:

```
package com.learnersacademy.dao;
import java.util.List;
import org.hibernate.Session;
import org.hibernate.Transaction;
import com.learnersacademy.entity.Teacher;
import com.learnersacademy.util.HibernateUtil;

public class TeacherDao {

public Teacher getTeacher(int id) {
Transaction transaction = null;
Teacher teacher = null;

try (Session session = HibernateUtil.getSessionFactory().openSession()) {
transaction = session.beginTransaction();
teacher = session.get(Teacher.class, id);
transaction.commit();
} catch (Exception e) {
if (transaction != null) {
transaction.rollback();
}
e.printStackTrace();
}
return teacher;
}

public Teacher saveTeacher(Teacher teacher) {
Transaction transaction = null;
```

```
Teacher createdTeacher = null;
Session session = null;
try {
session = HibernateUtil.getSessionFactory().openSession();
transaction = session.beginTransaction();
session.save(teacher);
transaction.commit();
} catch (Exception e) {
if (transaction != null) {
transaction.rollback();
}
e.printStackTrace();
} finally {
session.close();
}
return createdTeacher;
}

@SuppressWarnings("unchecked")
public List<Teacher> getAllTeachers() {
Transaction transaction = null;
List<Teacher> listOfTeachers = null;
try (Session session = HibernateUtil.getSessionFactory().openSession()) {
transaction = session.beginTransaction();
listOfTeachers = session.createQuery("from Teacher").getResultList();
                                        transaction.commit();
} catch (Exception e) {
if (transaction != null) {
transaction.rollback();
}
e.printStackTrace();
}
return listOfTeachers;
}
public void deleteTeacher(int id) {
Transaction transaction = null;
Session session = null;
try {
session = HibernateUtil.getSessionFactory().openSession();
transaction = session.beginTransaction();
Teacher teacherObj = session.get(Teacher.class, id);
session.delete(teacherObj);
transaction.commit();
} catch (Exception e) {
if (transaction != null) {
transaction.rollback();
}
e.printStackTrace();
} finally {
session.close();
}
}
}
```

Step 7: Creating an JSP pages

- In the Project Explorer, expand the project
  ' LearnersAcademyAdministrativePortal '
- Expand **src/main/webapp**. Create new folder **pages.** Right click on **pages.** Choose **New->JSP File**
- Enter the filename as login.jsp and click on **Finish**
- Enter the following code:

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Login Page</title>
<link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css" />
</head>
<body>
<center>
<h1>Learner's Academy</h1>
<div class="container fluid">
<div>
<h4>Login Form</h4>
<form action="/LearnersAcademyAdministrativePortal/login" class="was-validated">
```

```
<div class="form-group">
<label for="email">Username</label> <input type="text"
class="form-control" id="email" name="username"
placeholder="Enter your username" required />
<div class="valid-feedback">Valid.</div>
<div class="invalid-feedback">Please fill out this field</div>
</div>
<div class="form-group">
<label for="password">Password</label> <input type="password"
class="form-control" id="password" name="password"
placeholder="Enter your password" required minlength="3" />
<div class="valid-feedback">Valid.</div>
<div class="invalid-feedback">Please fill out this field</div>
</div>
<div class="form-group form-check">
<label class="form-check-label"></label><input type="checkbox"
class="form-check-input" name="remember" required /> I agree
terms and conditions.
<div class="valid-feedback">Valid.</div>
<div class="invalid-feedback">Check this checkbox to
continue.</div>
</div>
<button type="submit" class="btn btn-primary">Login</button>
</form>
<hr />
</div>
</div>
</center>
</body>
</html>
```

- Click on the **Save** icon

- Expand **src/main/webapp**. Create new folder **pages.** Right click on **pages.** Choose **New->JSP File**
- Enter the filename as dashboard.jsp and click on **Finish**
- Enter the following code:

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
```

```
<!DOCTYPE html>
<jsp:include page="menu.jsp" />
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Welcome To Learner's Academy</title>
<style type="text/css">
html {
  background: url(/LearnersAcademyAdministrativePortal/images/img1.jpg) no-repeat center center
fixed;
  -webkit-background-size: cover;
  -moz-background-size: cover;
  -o-background-size: cover;
  background-size: cover;
}
</style>
</head>
<body>
</body>
</html>
```

- Click on the **Save** icon

- Expand **src/main/webapp**. Create new folder **pages.** Right click on **pages.** Choose **New->JSP File**
- Enter the filename as addClass.jsp and click on **Finish**
- Enter the following code:

```
        <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css" />
<div class="container-fluid">
<div class="card-header py-3">
<h6 class="m-0 font-weight-bold text-primary"><a href="pages/dashboard.jsp">Home</a>
&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;
<a href="/LearnersAcademyAdministrativePortal/logout" style='float:right'>Logout</a></h6>
</div>
<div class="d-sm-flex align-items-center justify-content-between mb-4">
<h1 class="h3 mb-0 text-gray-800">Add Classes</h1> </div>
<div class="row">
```

```
<div class="col-lg-12">
<form class="addClasses" action="/LearnersAcademyAdministrativePortal/classRoom"
method="post">
<div class="form-row">
<div class="form-group col-md-5">
<label for="className">Class Name</label>
<input type="text" class="form-control" id="className" name="className" placeholder="Enter
Class Name" required="required"> </div>
<div class="form-group col-md-5">
<label for="sectionName">Section Name</label>
<input type="text" class="form-control" id="sectionName" name="sectionName" placeholder="Enter
Section Name"> </div>
<div class="form-group col-md-5">
<label for="totalNumberOfStudents">Total No of Students</label>
<input type="number" class="form-control" id="totalNumberOfStudents"
name="totalNumberOfStudents" placeholder="Enter Total Number of Students"
required="required"> </div>
<div class="form-group col-md-5">
<label for="roomNo">Room Name</label>
<input type="text" class="form-control" id="roomNo" name="roomNo" placeholder="Enter Room
Name"> </div>
<div class="form-group col-md-5">
<label for="classTeacherName">Class Teacher Name</label>
<input type="text" class="form-control" id="classTeacherName" name="classTeacherName"
placeholder="Enter Class Teacher Name"> </div>
</div>
<div class="form-group">
<button type="reset" id="resetFormAddClasses" name="resetFormAddClasses" class="btn btn-
danger">clear</button>
<button type="submit" id="submitFormAddClasses" name="submitFormAddClasses" class="btn btn-
primary">Submit</button>
</div>
<input type="hidden"  name="action" value="new"/>
</form>
</div>
</div>
</div>
```

- Click on the **Save** icon

- Expand **src/main/webapp**. Create new folder **pages.** Right click on **pages.** Choose **New->JSP File**
- Enter the filename as addClassSubjectMapping.jsp and click on **Finish**
- Enter the following code:

```
<%@page import="com.learnersacademy.entity.ClassRoom"%>
<%@page import="com.learnersacademy.entity.Subject"%>
<%@page import="java.util.List"%>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.1.0/jquery.min.js"></script>
<script src="http://ajax.aspnetcdn.com/ajax/jquery.validate/1.9/jquery.validate.js"></script>
<link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css" />
<%
List<ClassRoom> classRooms = (List<ClassRoom>) session.getAttribute("classRooms");
List<Subject> subjects =  (List<Subject>) session.getAttribute("subjects");
%>
<div class="container-fluid">
<div class="card-header py-3">
<h6 class="m-0 font-weight-bold text-primary"><a href="pages/dashboard.jsp">Home</a>
&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;
<a href="/LearnersAcademyAdministrativePortal/logout" style='float:right'>Logout</a></h6>
</div>
<div class="d-sm-flex align-items-center justify-content-between mb-4">
<h1 class="h3 mb-0 text-gray-800">Add Class Subject Mapping</h1> </div>
<div class="row">
<div class="col-lg-12">
<form class="addClassesSubjectsMapping"
action="/LearnersAcademyAdministrativePortal/classSubjectMapping" method="post">
<div class="form-row">
<div class="form-group col-md-5">
<label for="subjectsNameCombo">Subject Name</label>
<select class="form-control" id="subjectsNameCombo" name="subjectsNameCombo">
<option value= 0 selected>Select Subject Name </option>
<% if(subjects != null && subjects.size()>0){%>
<%	for(Subject sub  : subjects){
%>
<option value="<%= sub.getId()%>"><%= sub.getSubjectName()%></option>
<%}
}%>
</select>
</div>
<div class="form-group col-md-5" style="display : none;">
<label for="subjectId">Subject Id</label>
<input type="text" class="form-control" id="subjectId" name="subjectId" placeholder="Enter Subject
Id"> </div>
<div class="form-group col-md-5" style="display : none;">
<label for="subjectName">Subject Name</label>
<input type="text" class="form-control" id="subjectName" name="subjectName" placeholder="Enter
Subject Name"> </div>
</div>
<div style="display: none" id="errorSub">
<label><span style="color: red">Please Select Valid Subject Name</span></label>
</div>
<div class="form-row">
<div class="form-group col-md-5">
<label for="classesNameCombo">Class Name</label>
<select class="form-control" id="classesNameCombo" name="classesNameCombo">
<option value= 0 selected>Select Class Name </option>
<% if(classRooms != null && classRooms.size()>0){%>
<% for(ClassRoom cr  : classRooms){
%>
<option value="<%= cr.getId()%>"><%= cr.getClassName()%></option>
<%}
}%>
</select>
</div>
<div class="form-group col-md-5" style="display : none;">
<label for="classId">Class Id</label>
<input type="text" class="form-control" id="classId" name="classId" placeholder="Enter Class Id">
</div>
<div class="form-group col-md-5" style="display : none;">
<label for="className">Class Name</label>
<input type="text" class="form-control" id="className" name="className" placeholder="Enter
Class Name"> </div>

</div>
<div style="display: none" id="errorClass">
<label><span style="color: red">Please Select Valid Class Name</span></label>
</div>
```

```
<div style="display: none" id="errorSubmit">
<label><span style="color: red">Please Select Valid Option</span></label>
</div>
<div class="form-group">
<button type="reset" id="resetFormAddClassesSubjectsMapping"
name="resetFormAddClassesSubjectsMapping" class="btn btn-danger">clear</button>
<button type="submit" id="submitFormAddClassesSubjectsMapping"
name="submitFormAddClassesSubjectsMapping" class="btn btn-primary">Submit</button>
</div>
<input type="hidden"  name="action" value="new"/>
</form>
</div>
</div>
</div>
<script type="text/javascript">
$(document).ready(function() {
$("#subjectsNameCombo").click(function() {
var subjectId = $("#subjectsNameCombo option:selected").val();
if(subjectId == 0){
document.getElementById("errorSub").style.display="inline";
document.getElementById("errorClass").style.display="none";
document.getElementById("errorSubmit").style.display="none";
return false;
} else {
document.getElementById("errorSub").style.display="none";
}
});

});
$(document).ready(function() {
$("#classesNameCombo").click(function() {
var classId = $("#classesNameCombo option:selected").val();
if(classId == 0){
document.getElementById("errorClass").style.display="inline";
document.getElementById("errorSub").style.display="none";
document.getElementById("errorSubmit").style.display="none";
return false;
} else {
document.getElementById("errorClass").style.display="none";
}
});
});
$(document).ready(function() {
$("#submitFormAddClassesSubjectsMapping").click(function() {
var subjectId = $("#subjectsNameCombo option:selected").val();
var classId = $("#classesNameCombo option:selected").val();
document.forms[0].elements["subjectName"].value = $("#subjectsNameCombo
option:selected").text();
document.forms[0].elements["className"].value = $("#classesNameCombo option:selected").text();
if(subjectId == 0 || classId == 0){
document.getElementById("errorSubmit").style.display="inline";
document.getElementById("errorClass").style.display="none";
document.getElementById("errorSub").style.display="none";
return false;
} else {
document.getElementById("errorSubmit").style.display="none";
}
});

});
</script>
```

- Click on the **Save** icon

- Expand **src/main/webapp**. Create new folder **pages.** Right click on **pages.** Choose **New->JSP File**
- Enter the filename as addStudent.jsp and click on **Finish**
- Enter the following code:

```
<%@page import="com.learnersacademy.entity.ClassRoom"%>
<%@page import="java.util.List"%>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.1.0/jquery.min.js"></script>
<script src="http://ajax.aspnetcdn.com/ajax/jquery.validate/1.9/jquery.validate.js"></script>
<link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css" />
<%
List<ClassRoom> classRooms = (List<ClassRoom>) session.getAttribute("classRooms");
%>
<div class="container-fluid">
```

```
<div class="card-header py-3">
<h6 class="m-0 font-weight-bold text-primary"><a href="pages/dashboard.jsp">Home</a>
&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;
<a href="/LearnersAcademyAdministrativePortal/logout" style='float:right'>Logout</a></h6>
</div>
<div class="d-sm-flex align-items-center justify-content-between mb-4">
<h1 class="h3 mb-0 text-gray-800">Add Students</h1> </div>
<div class="row">
<div class="col-lg-12">
<form class="addStudents" id="addStudents"
action="/LearnersAcademyAdministrativePortal/student" method="POST">
<div class="form-row">
<div class="form-group col-md-5">
<label for="studentName">Student Name</label>
<input type="text" class="form-control" id="studentName" name="studentName"
placeholder="Enter Student Name" required="required"> </div>
<div class="form-group col-md-5">
<label for="fathersName">Email Address</label>
<input type="email" class="form-control" id="email" name="email" placeholder="Enter Email
Address"> </div>
<div class="form-group col-md-5">
<label for="emergencyContactNumber">Emergency Contact Number</label>
<input type="text" class="form-control" id="emergencyContactNumber"
name="emergencyContactNumber" placeholder="Enter Emergency Contact Number"
required="required" maxlength="10"> </div>
<div class="form-group col-md-5">
<label for="bloodGroup">Blood Group</label>
<input type="text" class="form-control" id="bloodGroup" name="bloodGroup" placeholder="Enter
Blood Group" required="required"> </div>
<div class="form-group col-md-5">
<label for="age">Age</label>
<input type="number" class="form-control" id="age" name="age" placeholder="Enter Age"> </div>
<div class="form-group col-md-5">
<label for="gender">Gender</label>
<select class="form-control" id="gender" name="gender">
<option selected>Enter Gender </option>
<option value="male">Male</option>
<option value="female">Female</option>
<option value="others">Others</option>
</select>
</div>
```

```
<div class="form-group col-md-5">
<label for="classesNameCombo">Class Name</label>
<select class="form-control" id="classesNameCombo" name="classesNameCombo">
<option value= 0 selected>Select Class Name </option>
<% if(classRooms != null && classRooms.size()>0){
for(ClassRoom cr  : classRooms){
%>
<option value="<%= cr.getId()%>"><%= cr.getClassName()%></option>
<%}
}%>
</select>
</div>
<div style="display: none" id="errorClass">
<label><span style="color: red">Please Select Valid Class Name</span></label>
  </div>
<div class="form-group col-md-5" style="display : none;">
<label for="classId">Class Id</label>
<input type="text" class="form-control" id="classId" name="classId" placeholder="Enter Class Id">
</div>
<div class="form-group col-md-5" style="display : none;">
<label for="className">Class Name</label>
<input type="text" class="form-control" id="className" name="className" placeholder="Enter
Class Name"> </div>
<div class="form-group col-md-10">
<label for="address">Address</label>
<textarea class="form-control" id="address" name="address" rows="3"></textarea>
</div>
</div>
<div class="form-group">
<button type="reset" id="resetFormAddStudents" name="resetFormAddStudents" class="btn btn-
danger">clear</button>
<button type="submit" id="submitFormAddStudents" name="submitFormAddStudents" class="btn
btn-primary">Submit</button>
</div>
<input type="hidden" name="action" value="new">
</form>
</div>
</div>
</div>
<script type="text/javascript">
$(document).ready(function() {
```

```
$("#classesNameCombo").click(function() {
var classId = $("#classesNameCombo option:selected").val();
if(classId == 0){
document.getElementById("errorClass").style.display="inline";
return false;
} else {
document.getElementById("errorClass").style.display="none";
}
});

});
$(document).ready(function() {
$("#submitFormAddStudents").click(function() {
var classId = $("#classesNameCombo option:selected").val();
if(classId == 0){
document.getElementById("errorClass").style.display="inline";
return false;
} else {
document.getElementById("errorClass").style.display="none";
}
});

});
</script>
```

- Click on the **Save** icon

- Expand **src/main/webapp**. Create new folder **pages.** Right click on **pages.** Choose **New->JSP File**
- Enter the filename as addSubject.jsp and click on **Finish**
- Enter the following code:

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css" />
<div class="container-fluid">
<div class="card-header py-3">
<h6 class="m-0 font-weight-bold text-primary"><a href="pages/dashboard.jsp">Home</a>
```

```
&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;
<a href="/LearnersAcademyAdministrativePortal/logout" style='float:right'>Logout</a></h6>
</div>
<div class="d-sm-flex align-items-center justify-content-between mb-4">
<h1 class="h3 mb-0 text-gray-800">Add Subjects</h1>
</div>
<div class="row">
<div class="col-lg-12">
<form class="addSubjects "action="/LearnersAcademyAdministrativePortal/subject"
method="POST">
<div class="form-row">
<div class="form-group col-md-5">
<label for="subjectName">Subject Name</label> <input type="text"
class="form-control" id="subjectName" name="subjectName"
placeholder="Enter Subject Name" required="required">
</div>
<div class="form-group col-md-10">
<label for="subjectDescription">Subject Description</label>
<textarea class="form-control" id="subjectDescription"
name="subjectDescription" rows="3" maxlength="500"></textarea>
</div>
</div>
<div class="form-group">
<button type="reset" id="resetFormAddSubjects"
name="resetFormAddSubjects" class="btn btn-danger">clear</button>
<button type="submit" id="submitFormAddSubjects"
name="submitFormAddSubjects" class="btn btn-primary">Submit</button>
</div>
<input type="hidden"  name="action" value="new"/>
</form>
</div>
</div>
</div>
```

- Click on the **Save** icon

- Expand **src/main/webapp**. Create new folder **pages.** Right click on **pages.** Choose **New->JSP File**
- Enter the filename as addTeacher.jsp and click on **Finish**
- Enter the following code:

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
```

```
pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css" />
<div class="container-fluid">
<div class="card-header py-3">
<h6 class="m-0 font-weight-bold text-primary"><a href="pages/dashboard.jsp">Home</a>
&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;
<a href="/LearnersAcademyAdministrativePortal/logout" style='float:right'>Logout</a></h6>
</div>
<div class="d-sm-flex align-items-center justify-content-between mb-4">
<h1 class="h3 mb-0 text-gray-800">Add Teachers</h1>
</div>
<div class="row">
<div class="col-lg-12">
<form class="addTeachers" action="/LearnersAcademyAdministrativePortal/teacher"
method="POST">
<div class="form-row">
<div class="form-group col-md-5">
<label for="firstName">First Name</label> <input type="text"
class="form-control" id="firstName" name="firstName"
placeholder="Enter First Name" required="required">
</div>
<div class="form-group col-md-5">
<label for="lastName">Last Name</label> <input type="text"
class="form-control" id="lastName" name="lastName"
placeholder="Enter Last Name">
</div>
<div class="form-group col-md-5">
<label for="Contact Number">Contact Number</label> <input
type="text" class="form-control" id="contactNumber"
name="contactNumber" placeholder="Enter Contact Number" required="required"
maxlength="10">
</div>
<div class="form-group col-md-5">
<label for="emailAddress">Email Address</label> <input
type="email" class="form-control" id="emailAddress"
name="emailAddress" aria-describedby="emailHelp"
placeholder="Enter Email Address" required="required">
</div>
<div class="form-group col-md-5">
```

```
<label for="qualification">Qualification</label> <input
type="text" class="form-control" id="qualification"
name="qualification" placeholder="Enter Qualification" required="required">
</div>
<div class="form-group col-md-5">
<label for="age">Age</label> <input type="number"
class="form-control" id="age" name="age" placeholder="Enter Age">
</div>
<div class="form-group col-md-5">
<label for="martialStatus">Martial Status</label> <select
class="form-control" id="martialStatus" name="martialStatus">
<option selected>Enter Martial Status </option>
<option value="Single">Single</option>
<option value="Married">Married</option>
</select>
</div>
<div class="form-group col-md-5">
<label for="gender">Gender</label> <select class="form-control"
id="gender" name="gender">
<option selected>Enter Gender </option>
<option value="male">Male</option>
<option value="female">Female</option>
<option value="others">Others</option>
</select>
</div>
</div>
<div class="form-group">
<button type="reset" id="resetFormAddTeachers"
name="resetFormAddTeachers" class="btn btn-danger">clear</button>
<button type="submit" id="submitFormAddTeachers"
name="submitFormAddTeachers" class="btn btn-primary">Submit</button>
</div>
<input type="hidden"  name="action" value="new"/>
</form>
</div>
</div>
</div>
```

- Click on the **Save** icon

- Expand **src/main/webapp**. Create new folder **pages.** Right click on **pages.** Choose **New->JSP File**
- Enter the filename as addTeacherClassSubjectMapping.jsp and click on **Finish**
- Enter the following code:

```
<%@page import="com.learnersacademy.entity.ClassRoom"%>
<%@page import="com.learnersacademy.entity.Subject"%>
<%@page import="com.learnersacademy.entity.Teacher"%>
<%@page import="java.util.List"%>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.1.0/jquery.min.js"></script>
<script src="http://ajax.aspnetcdn.com/ajax/jquery.validate/1.9/jquery.validate.js"></script>
<link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css" />
<%
List<ClassRoom> classRooms = (List<ClassRoom>) session.getAttribute("classRooms");
List<Subject> subjects =  (List<Subject>) session.getAttribute("subjects");
List<Teacher> teachers =  (List<Teacher>) session.getAttribute("teachers");
%>
<div class="container-fluid">
<div class="card-header py-3">
<h6 class="m-0 font-weight-bold text-primary"><a href="pages/dashboard.jsp">Home</a>
&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;
<a href="/LearnersAcademyAdministrativePortal/logout" style='float:right'>Logout</a></h6>
</div>
<div class="d-sm-flex align-items-center justify-content-between mb-4">
<h1 class="h3 mb-0 text-gray-800">Add Teachers Classes Subjects Mapping</h1> </div>
<div class="row">
<div class="col-lg-12">
<form class="addTeachersClassesSubjectsMapping"
action="/LearnersAcademyAdministrativePortal/teacherClassSubjectMapping" method="post">
<div class="form-row">
<div class="form-group col-md-5">
<label for="teachersNameCombo">Teacher Name</label>
<select class="form-control" id="teachersNameCombo" name="teachersNameCombo">
<option value= 0 selected>Select Teacher Name </option>
<% if(teachers != null && teachers.size()>0){%>
<% for(Teacher teacher  : teachers){
%>
<option value="<%= teacher.getId()%>"><%= teacher.getFirstName()%></option>
<%}
}%>
</select>
</div>
<div class="form-group col-md-5" style="display : none;">
<label for="teacherId">Teacher Id</label>
<input type="text" class="form-control" id="teacherId" name="teacherId" placeholder="Enter
Teacher Id"> </div>
<div class="form-group col-md-5" style="display : none;">
<label for="teacherName">Teacher Name</label>
<input type="text" class="form-control" id="teacherName" name="teacherName"
placeholder="Enter Teacher Name"> </div>
</div>
<div style="display: none" id="errorTea">
<label><span style="color: red">Please Select Valid Teacher Name</span></label>
</div>
<div class="form-row">
<div class="form-group col-md-5">
<label for="classesNameCombo">Class Name</label>
<select class="form-control" id="classesNameCombo" name="classesNameCombo">
<option value= 0 selected>Select Class Name </option>
<% if(classRooms != null && classRooms.size()>0){%>
<% for(ClassRoom cr  : classRooms){
%>
<option value="<%= cr.getId()%>"><%= cr.getClassName()%></option>
<%}
}%>
</select>
</div>
<div class="form-group col-md-5" style="display : none;">
<label for="classId">Class Id</label>
<input type="text" class="form-control" id="classId" name="classId" placeholder="Enter Class Id">
</div>
<div class="form-group col-md-5" style="display : none;">
<label for="className">Class Name</label>
<input type="text" class="form-control" id="className" name="className" placeholder="Enter
Class Name"> </div>

</div>
<div style="display: none" id="errorClass">
```

```
<label><span style="color: red">Please Select Valid Class Name</span></label>
</div>
<div class="form-row">
<div class="form-group col-md-5">
<label for="subjectsNameCombo">Subject Name</label>
<select class="form-control" id="subjectsNameCombo" name="subjectsNameCombo">
<option value= 0 selected>Select Subject Name </option>
<% if(subjects != null && subjects.size()>0){%>
<%	for(Subject sub  : subjects){
%>
<option value="<%= sub.getId()%>"><%= sub.getSubjectName()%></option>
<%}
}%>
</select>
</div>
<div class="form-group col-md-5" style="display : none;">
<label for="subjectId">Subject Id</label>
<input type="text" class="form-control" id="subjectId" name="subjectId" placeholder="Enter Subject
Id"> </div>
<div class="form-group col-md-5" style="display : none;">
<label for="subjectName">Subject Name</label>
<input type="text" class="form-control" id="subjectName" name="subjectName" placeholder="Enter
Subject Name"> </div>
</div>
<div style="display: none" id="errorSub">
<label><span style="color: red">Please Select Valid Subject Name</span></label>
</div>
<div style="display: none" id="errorSubmit">
<label><span style="color: red">Please Select Valid Option</span></label>
</div>
<div class="form-group">
<button type="reset" id="resetFormAddTeachersClassesSubjectsMapping"
name="resetFormAddTeachersClassesSubjectsMapping" class="btn btn-danger">clear</button>
<button type="submit" id="submitFormAddTeachersClassesSubjectsMapping"
name="submitFormAddTeachersClassesSubjectsMapping" class="btn btn-
primary">Submit</button>
</div>
<input type="hidden"  name="action" value="new"/>
</form>
</div>
</div>
```

```
</div>
<script type="text/javascript">
$(document).ready(function() {
$("#teachersNameCombo").click(function() {
var teacherId = $("#teachersNameCombo option:selected").val();
if (teacherId == 0) {
document.getElementById("errorTea").style.display = "inline";
document.getElementById("errorSub").style.display = "none";
document.getElementById("errorClass").style.display = "none";
document.getElementById("errorSubmit").style.display = "none";
return false;
} else {
document.getElementById("errorTea").style.display = "none";
}
});

});

$(document).ready(function() {
$("#subjectsNameCombo").click(function() {
var subjectId = $("#subjectsNameCombo option:selected").val();
if (subjectId == 0) {
document.getElementById("errorSub").style.display = "inline";
document.getElementById("errorTea").style.display = "none";
document.getElementById("errorClass").style.display = "none";
document.getElementById("errorSubmit").style.display = "none";
return false;
} else {
document.getElementById("errorSub").style.display = "none";
}
});

});

$(document).ready(function() {
$("#classesNameCombo").click(function() {
var classId = $("#classesNameCombo option:selected").val();
if (classId == 0) {
document.getElementById("errorClass").style.display = "inline";
document.getElementById("errorTea").style.display = "none";
document.getElementById("errorSub").style.display = "none";
document.getElementById("errorSubmit").style.display = "none";
return false;
} else {
document.getElementById("errorClass").style.display = "none";
}
});

});
$(document).ready(function() {
$("#submitFormAddTeachersClassesSubjectsMapping").click(function() {
var subjectId = $("#subjectsNameCombo option:selected").val();
var classId = $("#classesNameCombo option:selected").val();
var teacherId = $("#teachersNameCombo option:selected").val();
document.forms[0].elements["subjectName"].value = $("#subjectsNameCombo
option:selected").text();
document.forms[0].elements["className"].value = $("#classesNameCombo option:selected").text();
document.forms[0].elements["teacherName"].value = $("#teachersNameCombo
option:selected").text();
if (subjectId == 0 || classId == 0 || teacherId == 0) {
document.getElementById("errorSubmit").style.display = "inline";
document.getElementById("errorTea").style.display = "none";
document.getElementById("errorSub").style.display = "none";
document.getElementById("errorClass").style.display = "none";
return false;
} else {
document.getElementById("errorSubmit").style.display = "none";
}
});
});
</script>
```

- Click on the **Save** icon

- Expand **src/main/webapp**. Create new folder **pages.** Right click on **pages.** Choose **New->JSP File**
- Enter the filename as error.jsp and click on **Finish**
- Enter the following code:

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
```

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Learner's Academy</title>
</head>
<body>
<center>
<h1>Learner's Academy</h1>
<h2>
<a href="pages/login.jsp">Login</a>
<p>Looks like your credentials are incorrect!, Please try again! </p>
</h2>
</center>
</body>
</html>
```

- Click on the **Save** icon

- Expand **src/main/webapp**. Create new folder **pages.** Right click on **pages.** Choose **New->JSP File**
- Enter the filename as listClassDetailsReport.jsp and click on **Finish**
- Enter the following code:

```
<%@page import="com.learnersacademy.entity.TeacherClassSubjectMapping"%>
<%@page import="com.learnersacademy.entity.Student"%>
<%@page import="java.util.List"%>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<%@taglib prefix="c" uri="http://java.sun.com/jstl/core"%>
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt"%>
<%@ taglib prefix="fn" uri="http://java.sun.com/jsp/jstl/functions"%>
<!DOCTYPE html>
<link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css" />
  <%
List<TeacherClassSubjectMapping> teacherClassSubjectMappings =
(List<TeacherClassSubjectMapping>) session.getAttribute("teacherClassSubjectMappings");
        List<Student> studentDetails = (List<Student>) session.getAttribute("studentDetails");
%>
<div class="container-fluid">
<div class="card-header py-3">
<h6 class="m-0 font-weight-bold text-primary"><a href="pages/dashboard.jsp">Home</a>
```

```
&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;
<a href="/LearnersAcademyAdministrativePortal/logout" style='float:right'>Logout</a></h6>
</div>
<div class="d-sm-flex align-items-center justify-content-between mb-4">
<h1 class="h3 mb-0 text-gray-800">List Class Details Report</h1> </div>
<div class="row">
<div class="col-lg-12">
          <div class="card shadow mb-4">
            <div class="card-header py-3">
              <h6 class="m-0 font-weight-bold text-primary">Teachers and Handling
Subjects</h6>
            </div>
            <div class="card-body">
              <div class="table-responsive">
                <table class="table table-bordered teachersHandlingSubjectsList" id="dataTable
teachersHandlingSubjectsList" width="100%" cellspacing="0">
                  <thead>
                    <tr>
<th>Teacher Name</th>
                     <th>Subject Name</th>
                    </tr>
                  </thead>
                  <tfoot>
                    <tr>
                                          <c:forEach var="teacherClassSubjectMapping"
items="${teacherClassSubjectMappings}">
    <tr>
<td><c:out value="${teacherClassSubjectMapping.teacherName}" /></td>
<td><c:out value="${teacherClassSubjectMapping.subjectName}" /></td>
    </tr>
    </c:forEach>
                    </tr>
                  </tfoot>
                </table>
              </div>
            </div>
          </div>

          <!-- DataTales Example -->
          <div class="card shadow mb-4">
            <div class="card-header py-3">
```

```
                <h6 class="m-0 font-weight-bold text-primary">Students in the Class</h6>
              </div>
              <div class="card-body">
                <div class="table-responsive">
                  <table class="table table-bordered studentsInTheClassList" id="dataTable
studentsInTheClassList" width="100%" cellspacing="0">
                    <thead>
                      <tr>
                       <th>Student Name</th>
                       <th>Gender</th>
                       <th>Contact Number</th>
                       <th>Blood Group</th>
                      </tr>
                    </thead>
                    <tfoot>
                      <tr>
                       <c:forEach var="student" items="${studentDetails}">
<tr>
<td><c:out value="${student.name}" /></td>
<td><c:out value="${student.gender}" /></td>
<td><c:out value="${student.emergencyContactNumber}" /></td>
<td><c:out value="${student.bloodGroup}" /></td>
</tr>
  </c:forEach>
                      </tr>
                    </tfoot>
                  </table>
                </div>
              </div>
            </div>
</div>
</div>
</div>
```

- Click on the **Save** icon

- Expand **src/main/webapp**. Create new folder **pages.** Right click on **pages.** Choose **New->JSP File**
- Enter the filename as listClassReport.jsp and click on **Finish**
- Enter the following code:

```
<%@page import="com.learnersacademy.entity.ClassRoom"%>
<%@page import="java.util.List"%>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<%@taglib prefix="c" uri="http://java.sun.com/jstl/core"%>
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt"%>
<%@ taglib prefix="fn" uri="http://java.sun.com/jsp/jstl/functions"%>
<!DOCTYPE html>
<link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css" />
<%
List<ClassRoom> classRooms = (List<ClassRoom>) session.getAttribute("classRooms");
%>
<div class="container-fluid">
<div class="card-header py-3">
<h6 class="m-0 font-weight-bold text-primary"><a href="pages/dashboard.jsp">Home</a>
&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;
<a href="/LearnersAcademyAdministrativePortal/logout" style='float:right'>Logout</a></h6>
</div>
<div class="d-sm-flex align-items-center justify-content-between mb-4">
<h1 class="h3 mb-0 text-gray-800">List Classes Report</h1>
</div>
<div class="row">
<div class="col-lg-12">
<div class="card shadow mb-4">
<div class="card-header py-3">
<h6 class="m-0 font-weight-bold text-primary">Classes Report</h6>
</div>
<div class="card-body">
<div class="table-responsive">
<table class="table table-bordered classesListReport"
id="dataTable" width="100%" cellspacing="0">
<thead>
<tr>
<th>Class Name</th>
<th>Action</th>
</tr>
</thead>
<tfoot>
<tr>
```

```
<c:forEach var="classroom" items="${classRooms}">
<tr>
<td><c:out value="${classroom.className}" /></td>
<td><a
href="/LearnersAcademyAdministrativePortal/classReport?action=classReportDetails&classId=<c:out
value='${classroom.id}' />">Generate Report</a></td>
</tr>
</c:forEach>
</tr>
</tfoot>
</table>
</div>
</div>
</div>
</div>
</div>
</div>
```

- Click on the **Save** icon

- Expand **src/main/webapp**. Create new folder **pages.** Right click on **pages.** Choose **New->JSP File**
- Enter the filename as list-classRooms.jsp and click on **Finish**
- Enter the following code:

```
<%@page import="com.learnersacademy.entity.ClassRoom"%>
<%@page import="com.learnersacademy.entity.Student"%>
<%@page import="java.util.List"%>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<%@taglib prefix="c" uri="http://java.sun.com/jstl/core"%>
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt"%>
<%@ taglib prefix="fn" uri="http://java.sun.com/jsp/jstl/functions"%>
<!DOCTYPE html>
<link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css" />
<%
List<ClassRoom> classRooms = (List<ClassRoom>) session.getAttribute("classRooms");
boolean delFlag = (boolean)session.getAttribute("delFlag");
%>
<div class="container-fluid">
```

```
<div class="card-header py-3">
<h6 class="m-0 font-weight-bold text-primary"><a href="pages/dashboard.jsp">Home</a>
&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;
<a href="/LearnersAcademyAdministrativePortal/logout" style='float:right'>Logout</a></h6>
</div>
<div class="d-sm-flex align-items-center justify-content-between mb-4">
<h1 class="h3 mb-0 text-gray-800">List Classes</h1>
</div>
<div class="row">
<div class="col-lg-12">

<!-- DataTales Example -->
<div class="card shadow mb-4">
<div class="card-header py-3">
<h6 class="m-0 font-weight-bold text-primary">Classes</h6>
</div>
<div class="card-body">
<div class="table-responsive">
<table class="table table-bordered classesList" id="dataTable"
width="100%" cellspacing="0">
<thead>
<tr>
<th>Class Name</th>
<th>Total Number of Students</th>
<th>Created Dt</th>
 <th>Action</th>
</tr>
</thead>
<tfoot>
<tr>
<c:forEach var="classroom" items="${classRooms}">
<tr>
<td><c:out value="${classroom.className}" /></td>
<td><c:out value="${classroom.totalNumberOfStudents}" /></td>
<td><c:out value="${classroom.createdDt}" /></td>
<td><a
href="/LearnersAcademyAdministrativePortal/classRoom?action=delete&id=<c:out
value='${classroom.id}' />">Delete</a></td>
</tr>
</c:forEach>
```

```
</tr>
</tfoot>
</table>
</div>
</div>
<% if(delFlag){%>
<div>
<p>
<span style="color: red;">Record deleted successfully!</span>
</p>
</div>
<%}%>
</div>
</div>
</div>
</div>
```

- Click on the **Save** icon

- Expand **src/main/webapp**. Create new folder **pages.** Right click on **pages.** Choose **New->JSP File**
- Enter the filename as listClassSubjectMapping.jsp and click on **Finish**
- Enter the following code:

```
<%@page import="com.learnersacademy.entity.ClassSubjectMapping"%>
<%@page import="java.util.List"%>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<%@taglib prefix="c" uri="http://java.sun.com/jstl/core"%>
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt"%>
<%@ taglib prefix="fn" uri="http://java.sun.com/jsp/jstl/functions"%>
<!DOCTYPE html>
<link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css" />
<%
List<ClassSubjectMapping> classSubjectMappings = (List<ClassSubjectMapping>)
session.getAttribute("classSubjectMappings");
boolean delFlag = (boolean)session.getAttribute("delFlag");
%>
<div class="container-fluid">
<div class="card-header py-3">
<h6 class="m-0 font-weight-bold text-primary"><a href="pages/dashboard.jsp">Home</a>
```

```
&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;
<a href="/LearnersAcademyAdministrativePortal/logout" style='float:right'>Logout</a></h6>
</div>
<div class="d-sm-flex align-items-center justify-content-between mb-4">
<h1 class="h3 mb-0 text-gray-800">List Classes Subjects Mapping</h1>
</div>
<div class="row">
<div class="col-lg-12">
<div class="card shadow mb-4">
<div class="card-header py-3">
<h6 class="m-0 font-weight-bold text-primary">Classes Subjects
Mapping</h6>
</div>
<div class="card-body">
<div class="table-responsive">
<table class="table table-bordered classesSubjectsMappingList"
id="dataTable" width="100%" cellspacing="0">
<thead>
<tr>
<th>Class Name</th>
<th>Subject Name</th>
<th>Created Dt</th>
<th>Action</th>
</tr>
</thead>
<tfoot>
<tr>
<c:forEach var="classSubjectMapping" items="${classSubjectMappings}">
<tr>
<td><c:out value="${classSubjectMapping.className}" /></td>
<td><c:out value="${classSubjectMapping.subjectName}" /></td>
<td><c:out value="${classSubjectMapping.createdDt}" /></td>
<td><a href="/LearnersAcademyAdministrativePortal/classSubjectMapping?action=delete&id=<c:out
value='${classSubjectMapping.id}' />">Delete</a></td>
</tr>
</c:forEach>

 </tr>
                                    </tfoot>
                                </table>
                            </div>
```

```
                </div>
                <% if(delFlag){%>
                   <div><p><span style="color: red;">Record deleted successfully!</span></p></div>
                <%}%>
</div>
</div>
</div>
</div>
```

- Click on the **Save** icon

- Expand **src/main/webapp**. Create new folder **pages.** Right click on **pages.** Choose **New->JSP File**
- Enter the filename as list-students.jsp and click on **Finish**
- Enter the following code:

```
<%@page import="com.learnersacademy.entity.Student"%>
<%@page import="java.util.List"%>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<%@taglib prefix="c" uri="http://java.sun.com/jstl/core"%>
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt"%>
<%@ taglib prefix="fn" uri="http://java.sun.com/jsp/jstl/functions"%>
<!DOCTYPE html>
<link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css" />
<%
List<Student> students = (List<Student>) session.getAttribute("students");
boolean delFlag = (boolean)session.getAttribute("delFlag");
%>
<div class="container-fluid">
<div class="card-header py-3">
<h6 class="m-0 font-weight-bold text-primary"><a href="pages/dashboard.jsp">Home</a>
&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;
<a href="/LearnersAcademyAdministrativePortal/logout" style='float:right'>Logout</a></h6>
</div>
<div class="d-sm-flex align-items-center justify-content-between mb-4">
<h1 class="h3 mb-0 text-gray-800">List Students</h1> </div>
<div class="row">
<div class="col-lg-12">
              <div class="card shadow mb-4">
```

```
                        <div class="card-header py-3">
                            <h6 class="m-0 font-weight-bold text-primary">Students</h6>
                        </div>
                        <div class="card-body">
                            <div class="table-responsive">
                                <table class="table table-bordered studentsList" id="dataTable" width="100%"
cellspacing="0">
                                    <thead>
                                        <tr>
                                          <th>Student Name</th>
                                            <th>Class Name</th>
                                            <th>Contact Number</th>
                                            <th>Blood Group</th>
                                            <th>Created Dt</th>
                                            <th>Action</th>
                                        </tr>
                                    </thead>
                                    <tfoot>
                                        <tr>
<c:forEach var="student" items="${students}">
<tr>
<td><c:out value="${student.name}" /></td>
<td><c:out value="${student.className}" /></td>
<td><c:out value="${student.emergencyContactNumber}" /></td>
<td><c:out value="${student.bloodGroup}" /></td>
<td><c:out value="${student.createdDt}" /></td>
<td><a href="/LearnersAcademyAdministrativePortal/student?action=delete&id=<c:out
value='${student.id}' />">Delete</a></td>
</tr>
</c:forEach>

 </tr>
                                    </tfoot>
                                </table>
                            </div>
                        </div>
                        <% if(delFlag){%>
                            <div><p><span style="color: red;">Record deleted successfully!</span></p></div>
                        <%}%>
                    </div>
</div>
```

```
</div>
</div>
```

- Click on the **Save** icon

- Expand **src/main/webapp**. Create new folder **pages.** Right click on **pages.** Choose **New->JSP File**
- Enter the filename as listSubjects.jsp and click on **Finish**
- Enter the following code:

```
<%@page import="com.learnersacademy.entity.ClassRoom"%>
<%@page import="com.learnersacademy.entity.Student"%>
<%@page import="com.learnersacademy.entity.Subject"%>
<%@page import="java.util.List"%>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<%@taglib prefix="c" uri="http://java.sun.com/jstl/core"%>
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt"%>
<%@ taglib prefix="fn" uri="http://java.sun.com/jsp/jstl/functions"%>
<!DOCTYPE html>
<link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css" />

   <%
List<Subject> subjects = (List<Subject>) session.getAttribute("subjects");
boolean delFlag = (boolean)session.getAttribute("delFlag");
%>

<div class="container-fluid">
<div class="card-header py-3">
<h6 class="m-0 font-weight-bold text-primary"><a href="pages/dashboard.jsp">Home</a>
&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;
<a href="/LearnersAcademyAdministrativePortal/logout" style='float:right'>Logout</a></h6>
</div>
<div class="d-sm-flex align-items-center justify-content-between mb-4">
<h1 class="h3 mb-0 text-gray-800">List Subjects</h1>
</div>
<div class="row">
<div class="col-lg-12">
<div class="card shadow mb-4">
<div class="card-header py-3">
<h6 class="m-0 font-weight-bold text-primary">Subjects</h6>
```

```
</div>
<div class="card-body">
<div class="table-responsive">
<table class="table table-bordered subjectsList" id="dataTable"
width="100%" cellspacing="0">
<thead>
<tr>
<th>Subject Name</th>
<th>Created Dt</th>
<th>Action</th>
</tr>
</thead>
<tfoot>
<tr>
<c:forEach var="subject" items="${subjects}">
<tr>
<td><c:out value="${subject.subjectName}" /></td>
<td><c:out value="${subject.createdDt}" /></td>
<td><a href="/LearnersAcademyAdministrativePortal/subject?action=delete&id=<c:out
value='${subject.id}' />">Delete</a></td>
</tr>
</c:forEach>
</tr>
</tfoot>
</table>
</div>
</div>
<% if(delFlag){%>
<div>
<p>
<span style="color: red;">Record deleted successfully!</span>
</p>
</div>
<%}%>
</div>
</div>
</div>
</div>
```

- Click on the **Save** icon

- Expand **src/main/webapp**. Create new folder **pages.** Right click on **pages.** Choose **New->JSP File**
- Enter the filename as listTeacher.jsp and click on **Finish**
- Enter the following code:

```
<%@page import="com.learnersacademy.entity.Teacher"%>
<%@page import="java.util.List"%>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<%@taglib prefix="c" uri="http://java.sun.com/jstl/core"%>
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt"%>
<%@ taglib prefix="fn" uri="http://java.sun.com/jsp/jstl/functions"%>
<!DOCTYPE html>
<link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css" />
<%
List<Teacher> teachers = (List<Teacher>) session.getAttribute("teachers");
boolean delFlag = (boolean)session.getAttribute("delFlag");
%>
<div class="container-fluid">
<div class="card-header py-3">
<h6 class="m-0 font-weight-bold text-primary"><a href="pages/dashboard.jsp">Home</a>
&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;
<a href="/LearnersAcademyAdministrativePortal/logout" style='float:right'>Logout</a></h6>
</div>
<div class="d-sm-flex align-items-center justify-content-between mb-4">
<h1 class="h3 mb-0 text-gray-800">List Teachers</h1>
</div>
<div class="row">
<div class="col-lg-12">
<div class="card shadow mb-4">
<div class="card-header py-3">
<h6 class="m-0 font-weight-bold text-primary">Teachers</h6>
</div>
<div class="card-body">
<div class="table-responsive">
<table class="table table-bordered teachersList" id="dataTable"
width="100%" cellspacing="0">
<thead>
<tr>
```

```
<th>Teacher Name</th>
<th>Contact Number</th>
<th>Qualification</th>
<th>Gender</th>
<th>Created Dt</th>
<th>Action</th>
</tr>
</thead>
<tfoot>
<tr>
<c:forEach var="teacher" items="${teachers}">
<tr>
<td><c:out value="${teacher.firstName}, ${teacher.lastName}" /></td>
<td><c:out value="${teacher.contactNumber}" /></td>
<td><c:out value="${teacher.qualification}" /></td>
<td><c:out value="${teacher.gender}" /></td>
<td><c:out value="${teacher.createdDt}" /></td>
<td><a
href="/LearnersAcademyAdministrativePortal/teacher?action=delete&id=<c:out value='${teacher.id}'
/>">Delete</a></td>
</tr>
</c:forEach>

</tr>
</tfoot>
</table>
</div>
</div>
<% if(delFlag){%>
<div>
<p>
<span style="color: red;">Record deleted successfully!</span>
</p>
</div>
<%}%>
</div>
</div>
</div>
</div>
```

- Click on the **Save** icon

- Expand **src/main/webapp**. Create new folder **pages.** Right click on **pages.** Choose **New->JSP File**
- Enter the filename as listTeacherClassSubjectMapping.jsp and click on **Finish**
- Enter the following code:

```
<%@page import="com.learnersacademy.entity.TeacherClassSubjectMapping"%>
<%@page import="java.util.List"%>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<%@taglib prefix="c" uri="http://java.sun.com/jstl/core"%>
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt"%>
<%@ taglib prefix="fn" uri="http://java.sun.com/jsp/jstl/functions"%>
<!DOCTYPE html>
<link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css" />
<%
List<TeacherClassSubjectMapping> teacherClassSubjectMappings =
(List<TeacherClassSubjectMapping>) session.getAttribute("teacherClassSubjectMappings");
boolean delFlag = (boolean)session.getAttribute("delFlag");
%>
<div class="container-fluid">
<div class="card-header py-3">
<h6 class="m-0 font-weight-bold text-primary"><a href="pages/dashboard.jsp">Home</a>
&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;
<a href="/LearnersAcademyAdministrativePortal/logout" style='float:right'>Logout</a></h6>
</div>
<div class="d-sm-flex align-items-center justify-content-between mb-4">
<h1 class="h3 mb-0 text-gray-800">List Teachers Classes Subjects Mapping</h1>
</div>
<div class="row">
<div class="col-lg-12">
<div class="card shadow mb-4">
<div class="card-header py-3">
<h6 class="m-0 font-weight-bold text-primary">Teachers Classes Subjects Mapping</h6>
</div>
<div class="card-body">
<div class="table-responsive">
<table
class="table table-bordered teachersClassesSubjectsMappingList"
```

```
id="dataTable" width="100%" cellspacing="0">
<thead>
<tr>
<th>Teacher Name</th>
<th>Class Name</th>
<th>Subject Name</th>
<th>Created Dt</th>
<th>Action</th>
</tr>
</thead>
<tfoot>
<tr>
<c:forEach var="teacherClassSubjectMapping" items="${teacherClassSubjectMappings}">
<tr>
<td><c:out value="${teacherClassSubjectMapping.teacherName}" /></td>
<td><c:out value="${teacherClassSubjectMapping.className}" /></td>
<td><c:out value="${teacherClassSubjectMapping.subjectName}" /></td>
<td><c:out value="${teacherClassSubjectMapping.createdDt}" /></td>
<td><a
href="/LearnersAcademyAdministrativePortal/teacherClassSubjectMapping?action=delete&id=<c:out
value='${teacherClassSubjectMapping.id}' />">Delete</a></td>
</tr>
</c:forEach>

</tr>
</tfoot>
</table>
</div>
</div>
<% if(delFlag){%>
<div>
<p>
<span style="color: red;">Record deleted successfully!</span>
</p>
</div>
<%}%>
</div>
</div>
</div>
</div>
```

- Click on the **Save** icon

- Expand **src/main/webapp**. Create new folder **pages.** Right click on **pages.** Choose **New->JSP File**
- Enter the filename as menu.jsp and click on **Finish**
- Enter the following code:

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>

<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-
awesome.min.css">
<link href="/LearnersAcademyAdministrativePortal/css/menuNavigation.css" rel="stylesheet"
type="text/css">
<header>
<div class="navbar">
<div class="dropdown">
<button class="dropbtn">Student<i class="fa fa-caret-down"></i></button>
<div class="dropdown-content">
<a href="/LearnersAcademyAdministrativePortal/classRoom?action=listClassName">Add
Student</a>
<a href="/LearnersAcademyAdministrativePortal/student?action=list">List Students</a>
</div>
</div>
<div class="dropdown">
<button class="dropbtn">ClassRoom<i class="fa fa-caret-down"></i></button>
<div class="dropdown-content">
<a href="/LearnersAcademyAdministrativePortal/classRoom?action=addJsp">Add Class</a>
<a href="/LearnersAcademyAdministrativePortal/classRoom?action=list">List Classes</a>
</div>
</div>
<div class="dropdown">
<button class="dropbtn">Teacher<i class="fa fa-caret-down"></i></button>
<div class="dropdown-content">
<a href="/LearnersAcademyAdministrativePortal/teacher?action=addJsp">Add Teachers</a>
<a href="/LearnersAcademyAdministrativePortal/teacher?action=list">List Teachers</a>
</div>
</div>
<div class="dropdown">
```

```
<button class="dropbtn">Subject<i class="fa fa-caret-down"></i></button>
<div class="dropdown-content">
<a href="/LearnersAcademyAdministrativePortal/subject?action=addJsp">Add Subjects</a>
<a href="/LearnersAcademyAdministrativePortal/subject?action=list">List Subjects</a>
</div>
</div>
<div class="dropdown">
<button class="dropbtn">Classes Mapping<i class="fa fa-caret-down"></i></button>
<div class="dropdown-content">
<a
href="/LearnersAcademyAdministrativePortal/classSubjectMapping?action=mappingDetails">Class
Subject Mapping</a>
<a href="/LearnersAcademyAdministrativePortal/classSubjectMapping?action=list">List Mapped
Class-Subject</a>
</div>
</div>
<div class="dropdown">
<button class="dropbtn">Teachers Mapping<i class="fa fa-caret-down"></i></button>
<div class="dropdown-content">
<a
href="/LearnersAcademyAdministrativePortal/teacherClassSubjectMapping?action=mappingDetails"
>Teacher Class Mapping</a>
<a href="/LearnersAcademyAdministrativePortal/teacherClassSubjectMapping?action=list">List
Mapped Teacher-Class</a>
</div>
</div>
<div class="dropdown">
<button class="dropbtn">Class Reports<i class="fa fa-caret-down"></i></button>
<div class="dropdown-content">
<a href="/LearnersAcademyAdministrativePortal/classRoom?action=list&reportFlag=Y">Class Report
Details</a>
</div>
</div>
<a href="/LearnersAcademyAdministrativePortal/logout">Log out</a>
</div>
</header>
```

- Click on the **Save** icon

Step 8: Configuring Hibernate

- In the Project Explorer, expand **LearnersAcademyAdministrativePortal/JavaSource**
- Right click on **JavaSource** and choose **New->Class**
- In **Package**, enter com.learnersacademy.util and in **Name** enter HibernateUtil and click on **Finish**
- Enter the following code:

```
package com.learnersacademy.util;
import java.util.Properties;
import org.hibernate.SessionFactory;
import org.hibernate.boot.registry.StandardServiceRegistryBuilder;
import org.hibernate.cfg.Configuration;
import org.hibernate.cfg.Environment;
import org.hibernate.service.ServiceRegistry;
import com.learnersacademy.entity.ClassRoom;
import com.learnersacademy.entity.ClassSubjectMapping;
import com.learnersacademy.entity.Student;
import com.learnersacademy.entity.Subject;
import com.learnersacademy.entity.Teacher;
import com.learnersacademy.entity.TeacherClassSubjectMapping;
public class HibernateUtil {

private static SessionFactory sessionFactory;
public static SessionFactory getSessionFactory() {

if(sessionFactory ==  null) {
try {
Configuration configuration = new Configuration();
Properties hibernateProperties = new Properties();
hibernateProperties.put(Environment.DRIVER, "com.mysql.cj.jdbc.Driver");
hibernateProperties.put(Environment.URL, "jdbc:mysql://localhost:3306/learners_academy");
hibernateProperties.put(Environment.USER, "root");
hibernateProperties.put(Environment.PASS, "admin");
hibernateProperties.put(Environment.DIALECT, "org.hibernate.dialect.MySQL5InnoDBDialect");
hibernateProperties.put(Environment.SHOW_SQL, "true");
hibernateProperties.put(Environment.FORMAT_SQL, "true");
hibernateProperties.put(Environment.HBM2DDL_AUTO, "update");
configuration.setProperties(hibernateProperties);
configuration.addAnnotatedClass(Student.class);
configuration.addAnnotatedClass(ClassRoom.class);
configuration.addAnnotatedClass(Teacher.class);
configuration.addAnnotatedClass(Subject.class);
configuration.addAnnotatedClass(ClassSubjectMapping.class);
configuration.addAnnotatedClass(TeacherClassSubjectMapping.class);

ServiceRegistry serviceRegistry = new StandardServiceRegistryBuilder()
.applySettings(configuration.getProperties()).build();

sessionFactory = configuration.buildSessionFactory(serviceRegistry);
}
catch (Exception e) {
e.printStackTrace();
}
}
return sessionFactory;
}
}
```

Step 9: Configuring web.xml

- In the Project Explorer, expand **LearnersAcademyAdministrativePortal ->src->main->webapp->WEB-INF**
- Double click on **web.xml** to open it in the editor
- Enter the following script:

```
<!DOCTYPE web-app PUBLIC
"-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
"http://java.sun.com/dtd/web-app_2_3.dtd" >

<web-app>
  <display-name>Archetype Created Web Application</display-name>
  <servlet>
          <servlet-name>LoginServlet</servlet-name>
          <display-name>LoginServlet</display-name>
          <description></description>
          <servlet-class>com.learnersacademy.servlets.LoginServlet</servlet-class>
```

```
</servlet>
<servlet>
        <servlet-name>ClassRoomServlet</servlet-name>
        <display-name>ClassRoomServlet</display-name>
        <description></description>
        <servlet-class>com.learnersacademy.servlets.ClassRoomServlet</servlet-class>
</servlet>
<servlet>
        <servlet-name>StudentServlet</servlet-name>
        <display-name>StudentServlet</display-name>
        <description></description>
        <servlet-class>com.learnersacademy.servlets.StudentServlet</servlet-class>
</servlet>
<servlet>
        <servlet-name>LogoutServlet</servlet-name>
        <display-name>LogoutServlet</display-name>
        <description></description>
        <servlet-class>com.learnersacademy.servlets.LogoutServlet</servlet-class>
</servlet>
<servlet>
        <servlet-name>TeacherServlet</servlet-name>
        <display-name>TeacherServlet</display-name>
        <description></description>
        <servlet-class>com.learnersacademy.servlets.TeacherServlet</servlet-class>
</servlet>
<servlet>
        <servlet-name>SubjectServlet</servlet-name>
        <display-name>SubjectServlet</display-name>
        <description></description>
        <servlet-class>com.learnersacademy.servlets.SubjectServlet</servlet-class>
</servlet>
<servlet>
        <servlet-name>ClassSubjectMappingServlet</servlet-name>
        <display-name>ClassSubjectMappingServlet</display-name>
        <description></description>
        <servlet-class>com.learnersacademy.servlets.ClassSubjectMappingServlet</servlet-class>
</servlet>
<servlet>
        <servlet-name>TeacherClassSubjectMappingServlet</servlet-name>
        <display-name>TeacherClassSubjectMappingServlet</display-name>
        <description></description>
```

```
		<servlet-class>com.learnersacademy.servlets.TeacherClassSubjectMappingServlet</servlet-
class>
  </servlet>
  <servlet>
		<servlet-name>ClassReportDetailsServlet</servlet-name>
		<display-name>ClassReportDetailsServlet</display-name>
		<description></description>
		<servlet-class>com.learnersacademy.servlets.ClassReportDetailsServlet</servlet-class>
  </servlet>
  <servlet-mapping>
		<servlet-name>LoginServlet</servlet-name>
		<url-pattern>/login</url-pattern>
  </servlet-mapping>
  <servlet-mapping>
		<servlet-name>ClassRoomServlet</servlet-name>
		<url-pattern>/classRoom</url-pattern>
  </servlet-mapping>
  <servlet-mapping>
		<servlet-name>StudentServlet</servlet-name>
		<url-pattern>/student</url-pattern>
  </servlet-mapping>
  <servlet-mapping>
		<servlet-name>LogoutServlet</servlet-name>
		<url-pattern>/logout</url-pattern>
  </servlet-mapping>
  <servlet-mapping>
		<servlet-name>TeacherServlet</servlet-name>
		<url-pattern>/teacher</url-pattern>
  </servlet-mapping>
  <servlet-mapping>
		<servlet-name>SubjectServlet</servlet-name>
		<url-pattern>/subject</url-pattern>
  </servlet-mapping>
  <servlet-mapping>
		<servlet-name>ClassSubjectMappingServlet</servlet-name>
		<url-pattern>/classSubjectMapping</url-pattern>
  </servlet-mapping>
  <servlet-mapping>
		<servlet-name>TeacherClassSubjectMappingServlet</servlet-name>
		<url-pattern>/teacherClassSubjectMapping</url-pattern>
  </servlet-mapping>
```

```
 <servlet-mapping>
         <servlet-name>ClassReportDetailsServlet</servlet-name>
         <url-pattern>/classReport</url-pattern>
 </servlet-mapping>
 <welcome-file-list>
  <welcome-file>pages/login.jsp</welcome-file>
 </welcome-file-list>
</web-app>
```

Step 10: Building the project

- From the **Project** menu at the top, click on **Build**
- If any compile errors are shown, fix them as required

Step 11: Publishing and starting the project

- If you do not see the **Servers** tab near the bottom of the IDE, go to **Window** menu and click on **Show View->Servers**
- Right click on the **Server** entry and choose **Add and Remove**
- Click the **Add** button to move **LearnersAcademyAdministrativePortal** from the **Available** list to the **Configured** list
- Click on **Finish**
- Right click on the **Server** entry and click on **Publish**
- Right click the **Server** entry and click on **Start**
- This will start the server

Step 12: Running the project

- To run the project, open a web browser and type: **http://localhost:8080/LearnersAcademyAdministrativePortal/**

Step 13: Pushing the code to your GitHub repositories

- Open your command prompt and navigate to the folder where you have created your files.

```
cd <folder path>
```

- Initialize your repository using the following command:

```
git init
```

- Add all the files to your git repository using the following command:

  git add .

- Commit the changes using the following command:

git commit . -m "Changes have been committed."

- Push the files to the folder you initially created using the following command:

git push -u origin master

Unique Selling Points of the Application

1. The application is designed to add Student, Class, Teacher and Subject related details where we can map these details as per required.

2. Application is also provided the option to generate Class report where we will see details about respective class.

3. The user is able to seamlessly switch between options or return to home or logout even after performing adding/mapping details

Conclusions

- Admin have option to edit Student, Teacher and Classes related details.

- Also have user authorization with database not hardcoded.

- Generate XML file of class report