Learner’s Academy. Source Code

```
package com.simplilearn.admin;


import java.io.IOException;

import java.util.List;


import javax.annotation.Resource;

import javax.servlet.RequestDispatcher;

import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.Cookie;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.sql.DataSource;


import com.simplilearn.models.Student;

import com.simplilearn.models.Subject;

import com.simplilearn.models.Teacher;

import com.simplilearn.models.Class;


/**

* Servlet implementation class AdminControllerServlet

*/

@WebServlet("/AdminControllerServlet")

public class AdminControllerServlet extends HttpServlet {

private static final long serialVersionUID = 1L;


private DbRetrieve dbRetrieve;
```

```
@Resource(name = "administretive_portal")
private DataSource datasource;

@Override
public void init() throws ServletException {

super.init();

// create instance of db util, to pass in conn pool object
try {
dbRetrieve = new DbRetrieve(datasource);

} catch (Exception e) {
throw new ServletException(e);
}

}

/**
 * @see HttpServlet#HttpServlet()
 */
public AdminControllerServlet() {
super();
// TODO Auto-generated constructor stub
}

@Override
protected void doPost(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {
```

```
doGet(req, resp);
}

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

// read the "command" parameter
String command = request.getParameter("command");

if (command == null) {
command = "CLASSES";
}

// if no cookeies
if (!getCookies(request, response) && (!command.equals("LOGIN"))) {

response.sendRedirect("/Administrative-Portal/login.jsp");
}

else {

// if there is no command, how to handle
// route the data to the appropriate method
switch (command) {

case "STUDENTS":
studentsList(request, response);
break;

case "TEACHERS":
teachersList(request, response);
break;

case "SUBJECTS":
subjectList(request, response);
break;

case "CLASSES":
classestList(request, response);
break;

case "ST_LIST":
classStudentsList(request, response);
break;

case "LOGIN":
login(request, response);
break;

default:
```

```
classestList(request, response);


}

}

} catch (Exception e) {

throw new ServletException(e);

}

// response.getWriter().append("Served at: ").append(request.getContextPath());

}


private void studentsList(HttpServletRequest request, HttpServletResponse response)
throws Exception {

// get students from db util

List<Student> students = dbRetrieve.getStudents();


// add students to the request

request.setAttribute("STUDENT_LIST", students);


// send it to the jsp view page

RequestDispatcher dispatcher = request.getRequestDispatcher("/list-students.jsp");

dispatcher.forward(request, response);


}


private void teachersList(HttpServletRequest request, HttpServletResponse response)
throws Exception {

// get students from db util

List<Teacher> teachers = dbRetrieve.getTeachers();


// add students to the request
```

```
request.setAttribute("TEACHERS_LIST", teachers);

// send it to the jSP view page
RequestDispatcher dispatcher = request.getRequestDispatcher("/teachers-list.jsp");
dispatcher.forward(request, response);

}

private void subjectList(HttpServletRequest request, HttpServletResponse response) throws Exception {
// get subjects from db util
List<Subject> subjects = dbRetrieve.getSubjects();

// add subjects to the request
request.setAttribute("SUBJECTS_LIST", subjects);

// send it to the jSP view page
RequestDispatcher dispatcher = request.getRequestDispatcher("/subjects-list.jsp");
dispatcher.forward(request, response);

}

private void classestList(HttpServletRequest request, HttpServletResponse response) throws Exception {
// get subjects from db util
List<Class> classes = dbRetrieve.getClasses();

// add subjects to the request
request.setAttribute("CLASSES_LIST", classes);
```

```
// send it to the jSP view page
RequestDispatcher dispatcher = request.getRequestDispatcher("/classes-list.jsp");
dispatcher.forward(request, response);

}

private void login(HttpServletRequest request, HttpServletResponse response) throws Exception {
String username = request.getParameter("username");
String password = request.getParameter("password");

if (username.toLowerCase().equals("admin") && password.toLowerCase().equals("admin")) {

Cookie cookie = new Cookie(username, password);

// Setting the maximum age to 1 day
cookie.setMaxAge(86400); // 86400 seconds in a day

// Send the cookie to the client
response.addCookie(cookie);
classestList(request, response);
} else {
RequestDispatcher dispatcher = request.getRequestDispatcher("/login.jsp");
dispatcher.forward(request, response);
}

}
```

```
private void classStudentsList(HttpServletRequest request, HttpServletResponse response)
throws Exception {


int classId = Integer.parseInt(request.getParameter("classId"));

String section = request.getParameter("section");

String subject = request.getParameter("subject");


// get subjects from db util

List<Student> students = dbRetrieve.loadClassStudents(classId);


// add subjects to the request

request.setAttribute("STUDENTS_LIST", students);

request.setAttribute("SECTION", section);

request.setAttribute("SUBJECT", subject);


// send it to the jSP view page

RequestDispatcher dispatcher = request.getRequestDispatcher("/class-students.jsp");

dispatcher.forward(request, response);


}


private boolean getCookies(HttpServletRequest request, HttpServletResponse response)
throws Exception {


boolean check = false;

Cookie[] cookies = request.getCookies();

// Find the cookie of interest in arrays of cookies

for (Cookie cookie : cookies) {


if (cookie.getName().equals("admin") && cookie.getValue().equals("admin")) {
```

```
check = true;
break;
}

}



return check;

}

}
```

DBReterive.java

```
package com.simplilearn.admin;

import java.sql.Connection;
import java.sql.ResultSet;
import java.sql.Statement;
import java.util.ArrayList;
import java.util.List;

import javax.sql.DataSource;

import com.simplilearn.models.Student;
import com.simplilearn.models.Subject;
import com.simplilearn.models.Teacher;
import com.simplilearn.models.Class;

public class DbRetrieve {
```

```
private DataSource dataSource;

public DbRetrieve(DataSource dataSource) {
this.dataSource = dataSource;
}

public List<Student> getStudents() {

List<Student> students = new ArrayList<>();

Connection myConn = null;
Statement myStmt = null;
ResultSet myRs = null;

try {

// get a connection
myConn = dataSource.getConnection();

// create sql stmt
String sql = "SELECT * FROM students";
myStmt = myConn.createStatement();

// execute query
myRs = myStmt.executeQuery(sql);

// process result
while (myRs.next()) {
```

```
// retrieve data from result set row
int id = myRs.getInt("id");
String firstName = myRs.getString("fname");
String lastName = myRs.getString("lname");
int age = myRs.getInt("age");
int aclass = myRs.getInt("class");

// create new student object
Student tempStudent = new Student(id, firstName, lastName, age, aclass);

// add it to the list of students
students.add(tempStudent);

}

} catch (Exception e) {
// TODO: handle exception
} finally {
// close JDBC objects
close(myConn, myStmt, myRs);
}
return students;

}

public List<Teacher> getTeachers() {

List<Teacher> teachers = new ArrayList<>();
```

```
Connection myConn = null;
Statement myStmt = null;
ResultSet myRs = null;

try {

// get a connection
myConn = dataSource.getConnection();

// create sql stmt
String sql = "SELECT * FROM teachers";
myStmt = myConn.createStatement();

// execute query
myRs = myStmt.executeQuery(sql);

// process result
while (myRs.next()) {

// retrieve data from result set row
int id = myRs.getInt("id");
String firstName = myRs.getString("fname");
String lastName = myRs.getString("lname");
int age = myRs.getInt("age");

// create new student object
Teacher temp = new Teacher(id, firstName, lastName, age);
// add it to the list of students
teachers.add(temp);

}

} catch (Exception e) {
// TODO: handle exception
} finally {
// close JDBC objects
close(myConn, myStmt, myRs);
}
return teachers;

}

public List<Subject> getSubjects() {

List<Subject> subjects = new ArrayList<>();

Connection myConn = null;
Statement myStmt = null;
ResultSet myRs = null;

try {

// get a connection
myConn = dataSource.getConnection();

// create sql stmt
```

```
String sql = "SELECT * FROM subjects";
myStmt = myConn.createStatement();

// execute query
myRs = myStmt.executeQuery(sql);

// process result
while (myRs.next()) {

// retrieve data from result set row
int id = myRs.getInt("id");
String name = myRs.getString("name");
String shortcut = myRs.getString("shortcut");

// create new student object
Subject temp = new Subject(id, name,shortcut);

// add it to the list of students
subjects.add(temp);


}


} catch (Exception e) {
// TODO: handle exception
} finally {
// close JDBC objects
close(myConn, myStmt, myRs);
}
return subjects;
```

```
}

public List<Class> getClasses() {

List<Class> classes = new ArrayList<>();

Connection myConn = null;
Statement myStmt = null;
ResultSet myRs = null;

try {

// get a connection
myConn = dataSource.getConnection();

// create sql stmt
String sql = "SELECT * FROM classes";
myStmt = myConn.createStatement();

// execute query
myRs = myStmt.executeQuery(sql);

// process result
while (myRs.next()) {

// retrieve data from result set row
int id = myRs.getInt("id");
int section = myRs.getInt("section");
int subject = myRs.getInt("subject");
int teacher = myRs.getInt("teacher");
String time = myRs.getString("time");

Teacher tempTeacher = loadTeacher(teacher);
Subject tempSubject = loadSubject(subject);

String teacher_name = tempTeacher.getFname() + " " + tempTeacher.getLname();

// create new student object
Class temp = new Class(id, section, teacher_name, tempSubject.getName(), time);

// add it to the list of students
classes.add(temp);

}

} catch (Exception e) {
// TODO: handle exception
} finally {
// close JDBC objects
close(myConn, myStmt, myRs);
}
return classes;

}

public Teacher loadTeacher(int teacherId) {
```

```
Teacher theTeacher = null;

Connection myConn = null;
Statement myStmt = null;
ResultSet myRs = null;

try {

// get a connection
myConn = dataSource.getConnection();

// create sql stmt
String sql = "SELECT * FROM teachers WHERE id = " + teacherId;
myStmt = myConn.createStatement();

// execute query
myRs = myStmt.executeQuery(sql);

// process result
while (myRs.next()) {

// retrieve data from result set row
int id = myRs.getInt("id");
String fname = myRs.getString("fname");
String lname = myRs.getString("lname");
int age = myRs.getInt("age");
theTeacher = new Teacher(id, fname, lname, age);

}
```

```
} catch (Exception e) {
// TODO: handle exception
} finally {
// close JDBC objects
close(myConn, myStmt, myRs);
}
return theTeacher;

}

public Subject loadSubject(int subjectId) {

Subject theSubject = null;

Connection myConn = null;
Statement myStmt = null;
ResultSet myRs = null;

try {

// get a connection
myConn = dataSource.getConnection();

// create sql stmt
String sql = "SELECT * FROM subjects WHERE id = " + subjectId;
myStmt = myConn.createStatement();

// execute query
```

```
myRs = myStmt.executeQuery(sql);

// process result
while (myRs.next()) {

// retrieve data from result set row
int id = myRs.getInt("id");
String name = myRs.getString("name");
String shortcut = myRs.getString("shortcut");

theSubject = new Subject(id, name,shortcut);

}

} catch (Exception e) {
// TODO: handle exception
} finally {
// close JDBC objects
close(myConn, myStmt, myRs);
}
return theSubject;

}

public Class loadClass(int classId) {

Class theClass = null;

Connection myConn = null;
```

```
Statement myStmt = null;
ResultSet myRs = null;

try {

// get a connection
myConn = dataSource.getConnection();

// create sql stmt
String sql = "SELECT * FROM clasess WHERE id = " + classId;
myStmt = myConn.createStatement();

// execute query
myRs = myStmt.executeQuery(sql);

// process result
while (myRs.next()) {

// retrieve data from result set row
int id = myRs.getInt("id");
int section = myRs.getInt("section");
int subject = myRs.getInt("subject");
int teacher = myRs.getInt("teacher");
String time = myRs.getString("time");

Teacher tempTeacher = loadTeacher(teacher);
Subject tempSubject = loadSubject(subject);

String teacher_name = tempTeacher.getFname() + " " + tempTeacher.getLname();
}

} catch (Exception e) {
// TODO: handle exception
} finally {
// close JDBC objects
close(myConn, myStmt, myRs);
}
return theClass;

}

public List<Student> loadClassStudents(int classId) {

List<Student> students = new ArrayList<>();

Connection myConn = null;
Statement myStmt = null;
ResultSet myRs = null;

try {

// get a connection
myConn = dataSource.getConnection();

// create sql stmt
String sql = "SELECT * FROM students WHERE class = " + classId;
myStmt = myConn.createStatement();
```

```
// execute query
myRs = myStmt.executeQuery(sql);

// process result
while (myRs.next()) {

// retrieve data from result set row
int id = myRs.getInt("id");
String firstName = myRs.getString("fname");
String lastName = myRs.getString("lname");
int age = myRs.getInt("age");
int aclass = myRs.getInt("class");

// create new student object
Student tempStudent = new Student(id, firstName, lastName, age, aclass);
students.add(tempStudent);

}

} catch (Exception e) {
// TODO: handle exception
} finally {
// close JDBC objects
close(myConn, myStmt, myRs);
}
return students;

}
```

```
private void close(Connection myConn, Statement myStmt, ResultSet myRs) {

try {
if (myRs != null) {
myRs.close();
}
if (myStmt != null) {
myStmt.close();
}
if (myConn != null) {
myConn.close();
}

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

}

}
```

Class.java

```
package com.simplilearn.models;

public class Class {

private int id;
private int section;
private String teacher;
private String subject;
private String time;


public Class(int id, int section, String teacher, String subject, String time) {
super();
this.id = id;
this.section = section;
this.teacher = teacher;
this.subject = subject;
this.time = time;
}


public int getId() {
return id;
}
public void setId(int id) {
this.id = id;
}
public int getSection() {
return section;
}
public void setSection(int section) {
this.section = section;
}
public String getTeacher() {
return teacher;
}
public void setTeacher(String teacher) {
this.teacher = teacher;
}
public String getSubject() {
return subject;
}
public void setSubject(String subject) {
this.subject = subject;
}
public String getTime() {
return time;
}
public void setTime(String time) {
this.time = time;
}



}
```

Student.java

```
package com.simplilearn.models;

public class Student {

private int id;
private String fname;
private String lname;
private int age;
private int aclass;



public Student(int id, String fname, String lname, int age, int aclass) {
super();
this.id = id;
this.fname = fname;
this.lname = lname;
this.age = age;
this.aclass = aclass;
}


public int getId() {
return id;
}
public void setId(int id) {
this.id = id;
}
public String getFname() {
return fname;
}
public void setFname(String fname) {
this.fname = fname;
}
public String getLname() {
return lname;
}
public void setLname(String lname) {
this.lname = lname;
}
public int getAge() {
return age;
}
public void setAge(int age) {
this.age = age;
}
public int getAclass() {
return aclass;
}
public void setAclass(int aclass) {
this.aclass = aclass;
}


@Override
public String toString() {
return "Student [id=" + id + ", fname=" + fname + ", lname=" + lname + ", age=" + age + ",
aclass=" + aclass
+ "]";
}



}
```

Subject.java

```
package com.simplilearn.models;
```

```
public class Subject {

private int id;
private String name;
private String shortcut;

public Subject(int id, String name, String shortcut ) {
super();
this.id = id;
this.name = name;
this.shortcut = shortcut;
}

public int getId() {
return id;
}

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

public String getShortcut() {
return shortcut;
}

public void setShortcut(String shortcut) {
this.shortcut = shortcut;
}
public String getName() {
return name;
}

public void setName(String name) {
this.name = name;
}




}
```

Teachers.java

```
package com.simplilearn.models;

public class Teacher {

private int id;
private String fname;
private String lname;
private int age;

public Teacher(int id, String fname, String lname, int age) {
super();
this.id = id;
this.fname = fname;
this.lname = lname;
this.age = age;
}

public int getId() {
return id;
}

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

public String getFname() {
return fname;
}

public void setFname(String fname) {
this.fname = fname;
}

public String getLname() {
return lname;
}

public void setLname(String lname) {
this.lname = lname;
}

public int getAge() {
return age;
}
```

```
public void setAge(int age) {
this.age = age;
}
```

Login.css

```
Body {
  font-family: Calibri, Helvetica, sans-serif;
  background-color: pink;

}
button {
  justify-content: center;
       background-color: rgb(128, 128, 255);
       width: 100%;
        color: white;
        padding: 15px;
        margin: 10px 0px;
        border: none;
        cursor: pointer;
         }

form {
        border: 1.4px solid black;
width: 45%;
margin: 0 auto;
    }
input[type=text], input[type=password] {
```

```
justify-content: center;
        width: 100%;
        margin: 8px 0;
        padding: 12px 20px;
        display: inline-block;
        border: 2px solid rgb(0, 255, 0);
        box-sizing: border-box;
    }
button:hover {
        opacity: 0.7;
    }




.container {

  justify-content: center;
        padding: 15px;
        background-color: #FFF8DC;
    }
}
```

Style.css

```
html, body{

padding:0px;
font-family:Verdana, Arial, Helvetica, sans-serif;
margin-left: 103px; /* Same as the width of the sidenav */
}
```

```
table {
border-collapse:collapse;
border:1px solid gray;
font-family: Tahoma,Verdana,Segoe,sans-serif;
width:72%;

}

th {
border-bottom:1px solid gray;
background:none repeat scroll 0 0 rgb(128, 128, 128);
padding:10px;
color: #FFFFFF;
}

tr {
border-top:1px solid gray;
text-align:center;
}

tr:nth-child(even) {background: #FFFFFF}
tr:nth-child(odd) {background: #BBBBBB}

#wrapper {width: 100%; text-align: center; }
#header {width: 72%; background: rgb(128, 128, 128); margin-top: 0px; padding:5px 0px 15px 0px;}
#header h3 {width: 100%; margin:auto; color: #FFFFFF;}
#container {width: 100%; margin:auto}
#container h3 {color: #000;}
#container #content {margin-top: 20px;}
```

```
.add-student-button {
border: 1px solid #666;
border-radius: 5px;
padding: 4px;
font-size: 12px;
font-weight: bold;
width: 120px;
padding: 5px 10px;

margin-bottom: 15px;
background: #cccccc;
}



.sidenav  {
  height: 100%;
  width: 200px;
  border-color: #FFFFFF;
  position: fixed;
  z-index: 1;
  top: 0;
  left: 0;
  background-color: #000080;
  overflow-x: hidden;
  padding-top: 20px;
}
```

```
.sidenav a {
  padding: 6px 6px 6px 32px;
  text-decoration: none;
  font-size: 25px;
  color: white;
  display: block;
}

.sidenav a:hover {
  color: blue;
}

@media screen and (max-height: 450px) {
  .sidenav {padding-top: 15px;}
  .sidenav a {font-size: 18px;}
}

#page{

  height: 100%;



}



#logo{
font-family: 'Trebuchet MS', sans-serif;
text-align: center;
color: white;
}

.bar-item{


border-color: #FFFFFF;
border-width: 3px;
border-bottom: .5px solid rgba(255, 255, 255, 0.247);

}
```

Classeclist.jsp

```
html, body{

padding:0px;
font-family:Verdana, Arial, Helvetica, sans-serif;
margin-left: 103px; /* Same as the width of the sidenav */
}

table {
border-collapse:collapse;
border:1px solid gray;
font-family: Tahoma,Verdana,Segoe,sans-serif;
width:72%;

}

th {
border-bottom:1px solid gray;
```

```
background:none repeat scroll 0 0 rgb(128, 128, 128);
padding:10px;
color: #FFFFFF;
}

tr {
border-top:1px solid gray;
text-align:center;
}

tr:nth-child(even) {background: #FFFFFF}
tr:nth-child(odd) {background: #BBBBBB}

#wrapper {width: 100%; text-align: center; }
#header {width: 72%; background: rgb(128, 128, 128); margin-top: 0px; padding:5px 0px 15px 0px;}
#header h3 {width: 100%; margin:auto; color: #FFFFFF;}
#container {width: 100%; margin:auto}
#container h3 {color: #000;}
#container #content {margin-top: 20px;}

.add-student-button {
border: 1px solid #666;
border-radius: 5px;
padding: 4px;
font-size: 12px;
font-weight: bold;
width: 120px;
padding: 5px 10px;
```

```
margin-bottom: 15px;
background: #cccccc;
}


.sidenav  {
  height: 100%;
  width: 200px;
  border-color: #FFFFFF;
  position: fixed;
  z-index: 1;
  top: 0;
  left: 0;
  background-color: #000080;
  overflow-x: hidden;
  padding-top: 20px;
}

.sidenav a {
  padding: 6px 6px 6px 32px;
  text-decoration: none;
  font-size: 25px;
  color: white;
  display: block;
}

.sidenav a:hover {
  color: blue;
```

```
}

@media screen and (max-height: 450px) {
  .sidenav {padding-top: 15px;}
  .sidenav a {font-size: 18px;}
}

#page{

  height: 100%;



}



#logo{
font-family: 'Trebuchet MS', sans-serif;
text-align: center;
color: white;

}

.bar-item{



border-color: #FFFFFF;
border-width: 3px;
border-bottom: .5px solid rgba(255, 255, 255, 0.247);
```

```
}
```

Class-student.jsp

```
html, body{

padding:0px;
font-family:Verdana, Arial, Helvetica, sans-serif;
margin-left: 103px; /* Same as the width of the sidenav */
}

table {
border-collapse:collapse;
border:1px solid gray;
font-family: Tahoma,Verdana,Segoe,sans-serif;
width:72%;

}

th {
border-bottom:1px solid gray;
background:none repeat scroll 0 0 rgb(128, 128, 128);
padding:10px;
color: #FFFFFF;
}

tr {
border-top:1px solid gray;
text-align:center;
}
```

```
tr:nth-child(even) {background: #FFFFFF}
tr:nth-child(odd) {background: #BBBBBB}

#wrapper {width: 100%; text-align: center; }
#header {width: 72%; background: rgb(128, 128, 128); margin-top: 0px; padding:5px 0px 15px 0px;}
#header h3 {width: 100%; margin:auto; color: #FFFFFF;}
#container {width: 100%; margin:auto}
#container h3 {color: #000;}
#container #content {margin-top: 20px;}

.add-student-button {
border: 1px solid #666;
border-radius: 5px;
padding: 4px;
font-size: 12px;
font-weight: bold;
width: 120px;
padding: 5px 10px;

margin-bottom: 15px;
background: #cccccc;
}



.sidenav  {
  height: 100%;
  width: 200px;
  border-color: #FFFFFF;
  position: fixed;

  z-index: 1;

  top: 0;

  left: 0;

  background-color: #000080;

  overflow-x: hidden;

  padding-top: 20px;

}


.sidenav a {

  padding: 6px 6px 6px 32px;

  text-decoration: none;

  font-size: 25px;

  color: white;

  display: block;

}


.sidenav a:hover {

  color: blue;

}


@media screen and (max-height: 450px) {

  .sidenav {padding-top: 15px;}

  .sidenav a {font-size: 18px;}

}


#page{


  height: 100%;
```

```
}


#logo{
font-family: 'Trebuchet MS', sans-serif;
text-align: center;
color: white;

}

.bar-item{


border-color: #FFFFFF;
border-width: 3px;
border-bottom: .5px solid rgba(255, 255, 255, 0.247);

}




<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>

<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
```

```
<title>Login</title>

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

</head>

<body style="background-image: url('css/background.jpg');">


  <center> <h1> Admin Login </h1> </center>

    <form action="AdminControllerServlet" method="POST">

        <div class="container">

             <input type="hidden" name="command" value="LOGIN" />

            <label>Username : </label>

            <br/>

            <input type="text" placeholder="Enter Username" name="username" required>

            <br/>

            <label>Password : </label>

            <br/>

            <input type="password" placeholder="Enter Password" name="password" required>

            <br/>

            <button type="submit">Login</button>

            <br/>

            <input type="checkbox" checked="checked"> Remember me


        </div>

    </form>




</body>

</html>
```

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>List of Teachers</title>
<link type="text/css" rel="stylesheet" href="css/style.css">
</head>
<body style="background-image: url('css/background.jpg');">
<div id="page">
<jsp:include page="left-list.jsp" />



<div id="wrapper">

<div id="header">
<h3>Subjects</h3>
</div>
</div>



<div id="container">

<div id="content">

<table>

<tr>
```

```
<th>Name</th>
<th>Shortcut</th>



</tr>


<c:forEach var="tempSubject" items="${SUBJECTS_LIST }">
<tr>


<td>${tempSubject.name}</td>
<td>${tempSubject.shortcut}</td>



</tr>



</c:forEach>


</table>
</div>
</div>
</div>


</body>
</html>





<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
```

```
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>List of Teachers</title>
<link type="text/css" rel="stylesheet" href="css/style.css">
</head>
<body style="background-image: url('css/background.jpg');">
<div id="page">
<jsp:include page="left-list.jsp" />


<div id="wrapper">

<div id="header">
<h3>Teachers</h3>
</div>
</div>


<div id="container">

<div id="content">

<table>

<tr>

<th>First Name</th>
```

```
<th>Last Name</th>
<th>age</th>

</tr>

<c:forEach var="tempStudent" items="${TEACHERS_LIST }">
<tr>

<td>${tempStudent.fname}</td>
<td>${tempStudent.lname}</td>
<td>${tempStudent.age}</td>



</tr>



</c:forEach>

</table>
</div>
</div>
</div>

</body>
</html>
```