Admincontrollerservlet.java

package main.java.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 main.java.model.Student;

import main.java.model.Subject;

import main.java.model.Teacher;

import main.java.model.Class;

@WebServlet("/AdminControllerServlet")

public class AdminControllerServlet extends HttpServlet {

private static final long serialVersionUID = 1L;

private DbRetrieve dbRetrieve;

@Resource(name = "Products")

private DataSource datasource;

@Override

public void init() throws ServletException {

super.init();

try {

dbRetrieve = new DbRetrieve(datasource);

} catch (Exception e) {

throw new ServletException(e);

}

}

public AdminControllerServlet() {

super();

}

@Override

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

doGet(req, resp);

}

protected void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

try {

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

if (command == null) {

command = "CLASSES";

}

if (!getCookies(request, response) && (!command.equals("LOGIN"))) {

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

}

else {

// 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 {

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

request.setAttribute("STUDENT\_LIST", students);

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

dispatcher.forward(request, response);

}

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

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

request.setAttribute("TEACHERS\_LIST", teachers);

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

dispatcher.forward(request, response);

}

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

List<Subject> subjects = dbRetrieve.getSubjects();

request.setAttribute("SUBJECTS\_LIST", subjects);

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

dispatcher.forward(request, response);

}

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

List<Class> classes = dbRetrieve.getClasses();

request.setAttribute("CLASSES\_LIST", classes);

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

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");

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

request.setAttribute("STUDENTS\_LIST", students);

request.setAttribute("SECTION", section);

request.setAttribute("SUBJECT", subject);

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();

for (Cookie cookie : cookies) {

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

check = true;

break;

} }

return check; }

}

DbRetrieve.java

package main.java.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 main.java.model.Student;

import main.java.model.Subject;

import main.java.model.Teacher;

import main.java.model.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 {

myConn = dataSource.getConnection();

String sql = "SELECT \* FROM classes";

myStmt = myConn.createStatement();

myRs = myStmt.executeQuery(sql);

while (myRs.next()) {

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();

Class temp = new Class(id, section, teacher\_name, tempSubject.getName(), time);

classes.add(temp);

}

} catch (Exception e) {

} finally {

close(myConn, myStmt, myRs);

}

return classes;

}

public Teacher loadTeacher(int teacherId) {

Teacher theTeacher = null;

Connection myConn = null;

Statement myStmt = null;

ResultSet myRs = null;

try {

myConn = dataSource.getConnection();

String sql = "SELECT \* FROM teachers WHERE id = " + teacherId;

myStmt = myConn.createStatement();

myRs = myStmt.executeQuery(sql);

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) {

} finally {

close(myConn, myStmt, myRs);

}

return theTeacher;

}

public Subject loadSubject(int subjectId) {

Subject theSubject = null;

Connection myConn = null;

Statement myStmt = null;

ResultSet myRs = null;

try {

myConn = dataSource.getConnection();

String sql = "SELECT \* FROM subjects WHERE id = " + subjectId;

myStmt = myConn.createStatement();

myRs = myStmt.executeQuery(sql);

while (myRs.next()) {

int id = myRs.getInt("id");

String name = myRs.getString("name");

String shortcut = myRs.getString("shortcut");

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

}

} catch (Exception e) {

} finally {

close(myConn, myStmt, myRs);

}

return theSubject;

}

public Class loadClass(int classId) {

Class theClass = null;

Connection myConn = null;

Statement myStmt = null;

ResultSet myRs = null;

try {

myConn = dataSource.getConnection();

String sql = "SELECT \* FROM clasess WHERE id = " + classId;

myStmt = myConn.createStatement();

myRs = myStmt.executeQuery(sql);

while (myRs.next()) {

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) {

} finally {

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 {

myConn = dataSource.getConnection();

String sql = "SELECT \* FROM students WHERE class = " + classId;

myStmt = myConn.createStatement();

myRs = myStmt.executeQuery(sql);

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");

Student tempStudent = new Student(id, firstName, lastName, age, aclass);

students.add(tempStudent);

}

} catch (Exception e) {

} finally {

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();

}

}

}

Testservlet.java

package main.java.admin;

import java.io.IOException;

import java.io.PrintWriter;

import java.sql.Connection;

import java.sql.ResultSet;

import java.sql.Statement;

import javax.annotation.Resource;

import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.sql.DataSource;

@WebServlet("/TestServlet")

public class TestServlet extends HttpServlet {

private static final long serialVersionUID = 1L;

//Define datasource/connection pool for reference

@Resource(name="Products")

private DataSource dataSource;

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

// Set the printwriter

PrintWriter out = response.getWriter();

response.setContentType("text/plain");

// establish connection to the DB

Connection myConn = null;

Statement myStmt = null;

ResultSet myRs = null;

try {

myConn = dataSource.getConnection();

//create a sql statement

String sql = "select \* from students";

myStmt = myConn.createStatement();

//execute the sql statement

myRs = myStmt.executeQuery(sql);

//process the resultset

while(myRs.next()) {

String fname = myRs.getString("fname");

out.println(fname);

}

}

catch(Exception e) {

e.printStackTrace();

}

}

}

Class.java

package main.java.model;

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 main.java.model;

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 main.java.model;

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;

}

}

Teacher.java

package main.java.model;

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;

}

}