**ЛАБОРАТОРНАЯ РАБОТА №4**

**по дисциплине «Разработка Java-приложений управления телекоммуникациями»**

Выполнил:

студент 3 курса

группы ИКПИ-15

Целоусов Григорий Андреевич

Проверила:

Белая Татьяна Иоанновна

**Оглавление**

[**1.** **Постановка задачи** 4](#_Toc153109967)

[**3.** **Описание программы** 5](#_Toc153109968)

[**4.** **Ход работы** 6](#_Toc153109969)

[4.1. Таблицы 6](#_Toc153109970)

[4.2. Работа с БД 6](#_Toc153109971)

[4.3. Сервлеты 7](#_Toc153109972)

[4.4. Веб-страницы 8](#_Toc153109973)

[**5.** **Вывод** 11](#_Toc153109974)

[**6.** **Листинг** 12](#_Toc153109975)

[6.1. model.User.java 12](#_Toc153109976)

[6.2. model.Request.java 12](#_Toc153109977)

[6.3. model.Service.java 13](#_Toc153109978)

[6.4. servlet.UserServlet.java 13](#_Toc153109979)

[6.5. servlet.RequestServlet.java 15](#_Toc153109980)

[6.6. servlet.ServiceServlet.java 16](#_Toc153109981)

[6.7. service.DBService.java 18](#_Toc153109982)

[6.8. service.HashService.java 19](#_Toc153109983)

[6.9. service.UserDBService.java 19](#_Toc153109984)

[6.10. service.RequestDBService.java 22](#_Toc153109985)

[6.11. service.ServiceDBService.java 23](#_Toc153109986)

[6.12. web.xml 24](#_Toc153109987)

[6.13. index.jsp 25](#_Toc153109988)

[6.14. 404.jsp 25](#_Toc153109989)

[6.15. user/users.jsp 26](#_Toc153109990)

[6.16. user/addUser.jsp 27](#_Toc153109991)

[6.17. user/editUser.jsp 28](#_Toc153109992)

[6.18. request/requests.jsp 29](#_Toc153109993)

[6.19. request/addRequest.jsp 30](#_Toc153109994)

[6.20. request/editRequest.jsp 31](#_Toc153109995)

[6.21. service/services.jsp 32](#_Toc153109996)

[6.22. service/addService.jsp 33](#_Toc153109997)

[6.23. service/editService.jsp 34](#_Toc153109998)

[6.24. styles/style.css 35](#_Toc153109999)

[6.25. pom.xml 37](#_Toc153110000)

# **Постановка задачи**

Цель работы – познакомиться с работой сервлетов, изучить аспекты работы с базами данных в Java.

Задача:

1. Продумать таблицы (минимум 3 таблицы) по своей теме (каждая тема выбирается и фиксируется за студентом);
2. Создать CRUD для работы с этими таблицами;
3. Создать клиентское приложение, которое работает с таблицами через сервлеты.

Вариант 21:

Тема – Система управления заявками на техническую поддержку.

# **Описание программы**

Программа написана на языке Java версии 20 для операционной системы Windows. Для разработки были использованы различные инструменты среды программирования IntelliJ Idea Ultimate, система контроля версий Git и удаленное хранилище, предоставленное сервисом GitHub.­

Программа представляет собой клиент-серверное приложение, CRUD-систему для работы с таблицами. В работе были использованы три таблицы, основанные на главных сущностях, разработанных для курсового проекта.

Таблицы:

1. Пользователи;
2. Заявки;
3. Оказанные услуги.

# **Ход работы**

## Таблицы

Ниже представлена информация о таблицах, выбранных для реализации в данной лабораторной работе (табл. 1-3). Для каждой сущности был реализован собственный класс модели поведения (package model). В данных классах созданы описанные ниже поля и реализованы конструкторы и методы доступа к полям.

Таблица 1. Пользователи (users).

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Атрибут** | id | name | email | login | password | role |
| **Тип** | int | String | String | String | String | int |

Таблица 2. Заявки (request).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Атрибут** | id | client | date | description | status |
| **Тип** | int | int | Date | String | int |

Таблица 3. Оказанные услуги (service).

|  |  |  |  |
| --- | --- | --- | --- |
| **Атрибут** | id | request | master |
| **Тип** | int | int | int |

## Работа с БД

В программе был разработан класс DBService, использующий JDBC API для подключения к серверу PostgreSQL. В классе реализованы методы для выполнения основных запросов к БД (select, insert, update, delete). Для каждого класса модели поведения сущностей был разработан класс формирования SQL-запросов (UserDBService, RequestDBService, ServiceDBService). Данные классы по переданным данным формируют запросы и с помощью DBService отправляют их на исполнение СУБД.

## Сервлеты

Для работы с каждой таблицей был разработан собственный сервлет (UserServlet, RequestServlet, ServiceServlet). Рассмотрим функции создания, чтения, редактирования и удаления (CRUD) записей на примере таблицы users.

В качестве URL-паттерна для сервлета UserServlet был выбран «/users/\*». Таким образом, сервлет будет отвечать за все адреса, начинающиеся с «/users».

При получении GET-запроса сервлет проверяет относительный адрес:

String path = req.getPathInfo();

Если path пустой или равен «/», сервлет получает список всех пользователей и перенаправляет запрос на страницу «/user/users.jsp», передавая атрибутом список пользователей. JSP-страница отображает список пользователей в виде таблицы.

Если path равен «/new» (вызывается при нажатии кнопки «Add user» на странице «/user/users.jsp»), запрос перенаправляется на страницу «/user/addUser.jsp», содержащую форму для создания нового пользователя. Если все данные были введены корректно, отправляется POST-запрос на создание пользователя.

Если path равен «/N» (вызывается при нажатии на любую запись таблицы), где N – id существующего пользователя, сервлет получает от БД данные этого пользователя и перенаправляет запрос на страницу «/user/editUser.jsp», передавая параметром данные пользователя. Если все данные были введены корректно, отправляется POST-запрос на редактирование (удаление) пользователя.

При получении POST-запроса сервлет проверяет относительный адрес:

String path = req.getPathInfo();

Если path равен «/new», сервлет проверяет, была ли нажата кнопка «Add», и с помощью UserDBService отправляет к БД запрос на создание пользователя, используя данные, полученные из формы.

Если path равен «/N», где N – id существующего пользователя, сервлет сервлет проверяет, была ли нажата кнопка «Edit», и с помощью UserDBService отправляет к БД запрос на редактирование пользователя, используя данные, полученные из формы. Если же была нажата кнопка «Delete», сервлет отправляет к БД запрос на удаление пользователя.

После всего этого сервлет перенаправляет запрос на страницу «result.jsp», передавая параметром текст результата выполнения запроса.

По такому же принципу работают сервлеты RequestServlet и ServiceServlet.

## Веб-страницы

Главной страницей веб-приложения является страница «/index.jsp», содержащая приветственную надпись и панель навигации со ссылками, на страницы с таблицами (рис. 1).

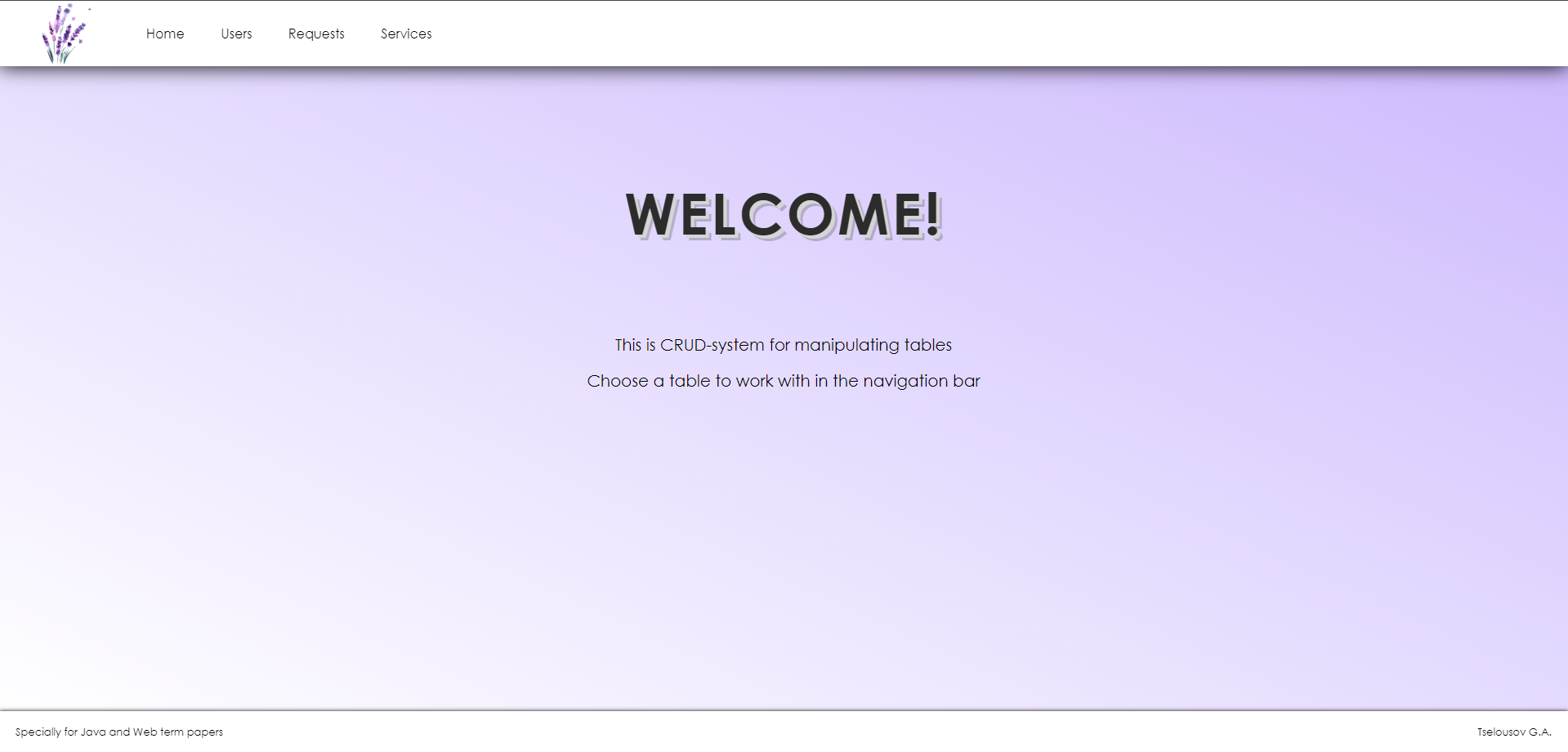


Рисунок 1 – index.jsp

Страницы для работы с таблицами рассмотрим на примере users. На странице «/user/users.jsp» (рис. 2) располагается таблица пользователей. Записи таблицы строятся с помощью атрибута users, полученного от сервлета UserServlet. Каждая запись является ссылкой на страницу редактирования/удаления записи. Под таблицей располагается кнопка добавления нового пользователя.

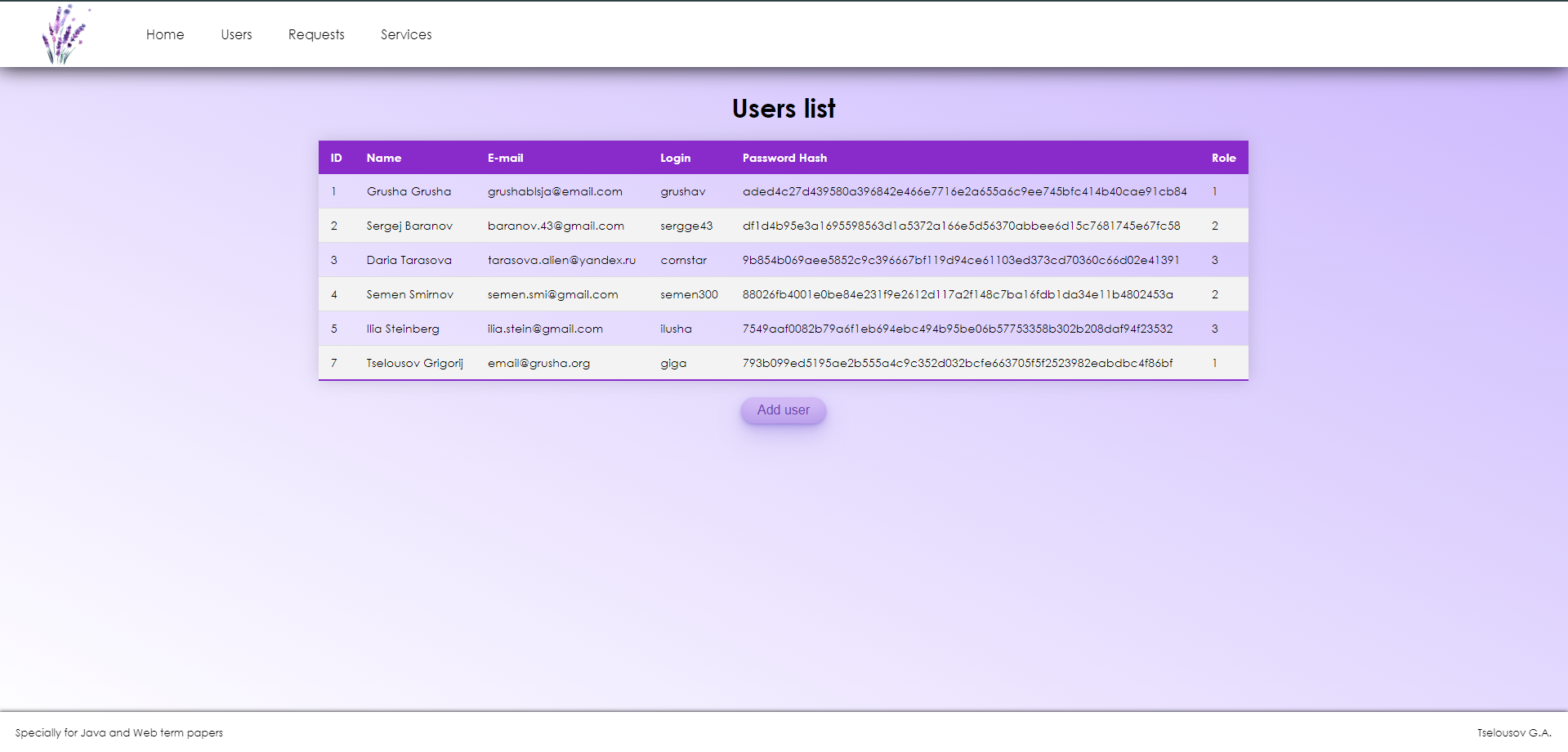


Рисунок 2 – user/users.jsp

Страница «/user/addUser.jsp» содержит форму для добавления нового пользователя в БД (рис. 3). Средствами HTML реализована валидация вводимых данных. При нажатии на кнопку «Add» сервлету на обработку отправляется POST-запрос добавления пользователя. Рядом располагается кнопка «Cancel» для возврата на страницу с таблицей пользователей.



Рисунок 3 – user/addUser.jsp

Страница «/user/edit.jsp» содержит форму для редактирования пользователя в БД (рис. 4). Средствами HTML реализована валидация вводимых данных. При нажатии на кнопку «Edit» сервлету на обработку отправляется POST-запрос редактирования пользователя. При нажатии на кнопку «Delete» сервлету на обработку отправляется POST-запрос удаления пользователя. Рядом располагается кнопка «Cancel» для возврата на страницу с таблицей пользователей.



Рисунок 4 – user/editUser.jsp

Идентичная логика реализована для работы с таблицами request и service.

Страница «result.jsp» содержит текст (результат обработки запроса) кнопку для возврата к таблице (рис. 5). Эти данные она получает от сервлета в качестве атрибутов.

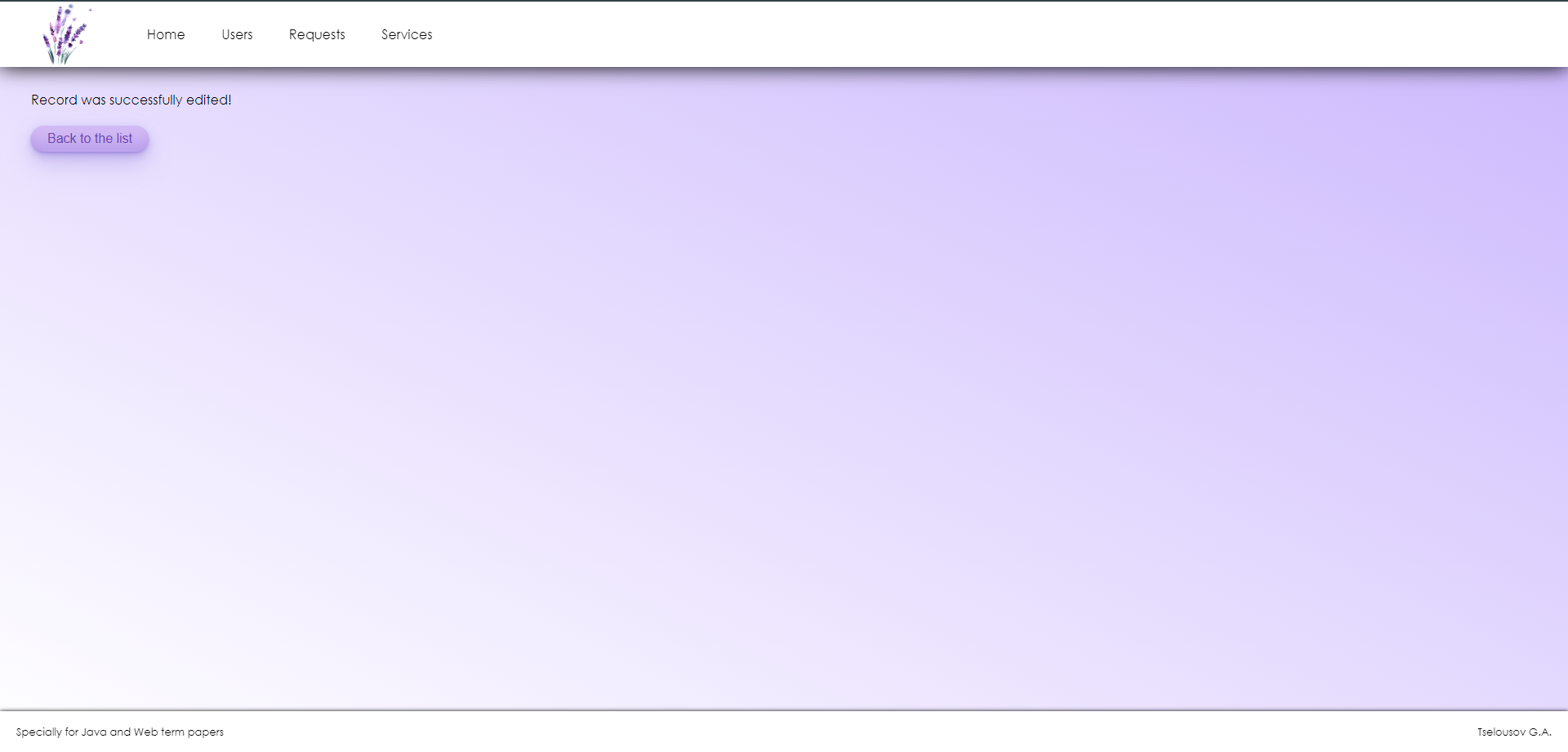


Рисунок 5 – result.jsp

# **Вывод**

Поставленная задача успешно проанализирована и решена, получены навыки разработки программного обеспечения на языке Java, а также навыки отладки и тестирования программы. В ходе работы были изучены различные аспекты языка Java, такие как: работа с сервлетами, установление соединения с базами данных и отправка запросов к БД, работа со встроенными классами для организации сервлетов и работы с БД. Получены навыки разработки клиент-серверного приложения.

В ходе работы была написана программа, представляющая собой клиент-серверное приложение, CRUD-систему для работы с таблицами. Были использованы три таблицы, основанные на главных сущностях, разработанных для курсового проекта.

# **Листинг**

## model.User.java

package model;  
  
public class User {  
 private int id;  
 private String name;  
 private String email;  
 private String login;  
 private String password;  
 private int role;  
  
 public User(int id, String name, String email, String login, String password, String role) {}  
 public User(int id, String name, String email, String login, String password, int role) {  
 this.id = id;  
 this.name = name;  
 this.email = email;  
 this.login = login;  
 this.password = password;  
 this.role = role;  
 }  
 public int getId() { return id; }  
 public void setId(int id) { this.id = 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 getLogin() { return login; }  
 public void setLogin(String login) { this.login = login; }  
 public String getPassword() { return password; }  
 public void setPassword(String password) { this.password = password; }  
 public int getRole() { return role; }  
 public void setRole(int role) { this.role = role; }  
}

## model.Request.java

package model;  
  
import java.text.SimpleDateFormat;  
import java.sql.Date;  
  
public class Request {  
 private int id;  
 private int client;  
 private Date date;  
 private String description;  
 private int status;  
  
 public Request(int id, int client, Date date, String description, int status) {  
 this.id = id;  
 this.client = client;  
 this.date = date;  
 this.description = description;  
 this.status = status;  
 }  
 public int getId() { return id; }  
 public void setId(int id) { this.id = id; }  
 public int getClient() { return client; }  
 public void setClient(int client) { this.client = client; }  
 public Date getDate() { return date; }  
 public String getDateString() { return new SimpleDateFormat("yyyy-MM-dd").format(date); }  
 public void setDate(Date date) { this.date = date; }  
 public String getDescription() { return description; }  
 public void setDescription(String description) { this.description = description; }  
 public int getStatus() { return status; }  
 public void setStatus(int status) { this.status = status; }  
}

## model.Service.java

package model;  
  
public class Service {  
 private int id;  
 private int request;  
 private int master;  
  
 public Service(int id, int request, int master) {  
 this.id = id;  
 this.request = request;  
 this.master = master;  
 }  
 public int getId() { return id; }  
 public void setId(int id) { this.id = id; }  
 public int getRequest() { return request; }  
 public void setRequest(int request) { this.request = request; }  
 public int getMaster() { return master; }  
 public void setMaster(int master) { this.master = master; }  
}

## servlet.UserServlet.java

package servlet;  
  
import model.User;  
import service.UserDBService;  
  
import javax.servlet.ServletException;  
import javax.servlet.http.HttpServlet;  
import javax.servlet.http.HttpServletRequest;  
import javax.servlet.http.HttpServletResponse;  
import java.io.IOException;  
import java.util.Arrays;  
import java.util.List;  
  
public class UserServlet extends HttpServlet {  
 @Override  
 protected void doGet(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {

UserDBService userDB = new UserDBService();  
String path = req.getPathInfo();  
  
if (path == null || path.isEmpty() || "/".equals(path)) {  
 List<User> users = userDB.getAll();  
 if (users != null) {  
 req.setAttribute("users", users);  
 req.getRequestDispatcher("/user/users.jsp").forward(req, resp);  
 }  
} else {  
 try {  
 String[] pathList = path.split("/");  
 String userID = pathList[1];  
 switch (userID) {  
 case "new" -> {  
 req.getRequestDispatcher("/user/addUser.jsp").forward(req, resp);  
 }  
 default -> {  
 User user = userDB.getByID(Integer.*parseInt*(userID));  
 if (user != null) {  
 req.setAttribute("user", user);  
 req.getRequestDispatcher("/user/editUser.jsp").forward(req, resp);  
 } else {  
 resp.sendRedirect("/404");  
 return;  
 }  
 }  
 }  
 } catch (Exception e) {

resp.sendRedirect("/404");  
 return;  
 }  
}  
resp.setContentType("text/html");  
super.doGet(req, resp);  
}  
  
 @Override  
 protected void doPost(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {

req.setCharacterEncoding("UTF-8");  
String path = req.getPathInfo();  
  
if (path == null || path.isEmpty()) {  
 return;  
}  
  
String result = "Nothing happened.";  
String userID = path.split("/")[1];  
switch (userID) {  
 case "new" -> {  
 if (req.getParameter("add") != null && req.getParameter("cancel") == null) {  
 result = (new UserDBService().create(new User(  
 0,  
 req.getParameter("name"),  
 req.getParameter("email"),  
 req.getParameter("login"),  
 req.getParameter("password"),  
 Integer.*parseInt*(req.getParameter("role"))  
 )))  
 ? "Record was successfully added!"  
 : "Record was not added, error occurred.";  
 }  
 }  
 default -> {  
 if (req.getParameter("edit") != null) {  
 result = (new UserDBService().edit(Integer.*parseInt*(userID), new User(  
 0,  
 req.getParameter("name"),  
 req.getParameter("email"),  
 req.getParameter("login"),  
 req.getParameter("password"),  
 Integer.*parseInt*(req.getParameter("role"))  
 )))  
 ? "Record was successfully edited!"  
 : "Record was not edited, error occurred.";  
 }  
 if (req.getParameter("delete") != null) {  
 result = (new UserDBService().delete(Integer.*parseInt*(userID)))  
 ? "Record was successfully deleted!"  
 : "Record was not deleted, error occurred.";  
 }  
 }  
}  
req.setAttribute("result", result);  
req.setAttribute("table", "users");  
req.getRequestDispatcher("/result.jsp").forward(req, resp);   
}  
}

## servlet.RequestServlet.java

package servlet;  
  
import model.Request;  
import service.RequestDBService;  
  
import javax.servlet.ServletException;  
import javax.servlet.http.HttpServlet;  
import javax.servlet.http.HttpServletRequest;  
import javax.servlet.http.HttpServletResponse;  
import java.io.IOException;  
import java.sql.Date;  
import java.util.List;  
  
public class RequestServlet extends HttpServlet {  
 @Override  
 protected void doGet(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {

RequestDBService requestDB = new RequestDBService();  
String path = req.getPathInfo();  
  
if (path == null || path.isEmpty() || "/".equals(path)) {  
 List<Request> requests = requestDB.getAll();  
 if (requests != null) {  
 req.setAttribute("requests", requests);  
 req.getRequestDispatcher("/request/requests.jsp").forward(req, resp);  
 }  
} else {  
 try {  
 String[] pathList = path.split("/");  
 String requestID = pathList[1];  
 switch (requestID) {  
 case "new" -> {  
 req.getRequestDispatcher("/request/addRequest.jsp").forward(req, resp);  
 }  
 default -> {  
 Request request = requestDB.getByID(Integer.*parseInt*(requestID));  
 if (request != null) {  
 req.setAttribute("request", request);  
 req.getRequestDispatcher("/request/editRequest.jsp").forward(req, resp);  
 } else {  
 resp.sendRedirect("/404");  
 return;  
 }  
 }  
 }  
 } catch (Exception e)

resp.sendRedirect("/404");  
 return;  
 }  
}  
resp.setContentType("text/html");  
super.doGet(req, resp);  
}  
  
 @Override  
 protected void doPost(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {

req.setCharacterEncoding("UTF-8");  
String path = req.getPathInfo();  
  
if (path == null || path.isEmpty()) {  
 return;  
}  
  
String result = "Nothing happened.";  
String requestID = path.split("/")[1];  
switch (requestID) {  
 case "new" -> {  
 if (req.getParameter("add") != null && req.getParameter("cancel") == null) {  
 result = (new RequestDBService().create(new Request(  
 0,  
 Integer.*parseInt*(req.getParameter("client")),  
 Date.*valueOf*(req.getParameter("date")),  
 req.getParameter("description"),  
 Integer.*parseInt*(req.getParameter("status"))  
 )))  
 ? "Record was successfully added!"  
 : "Record was not added, error occurred.";  
 }  
 }  
 default -> {  
 if (req.getParameter("edit") != null) {  
 result = (new RequestDBService().edit(Integer.*parseInt*(requestID), new Request(  
 0,  
 Integer.*parseInt*(req.getParameter("client")),  
 Date.*valueOf*(req.getParameter("date")),  
 req.getParameter("description"),  
 Integer.*parseInt*(req.getParameter("status"))  
 )))  
 ? "Record was successfully edited!"  
 : "Record was not edited, error occurred.";  
 }  
 if (req.getParameter("delete") != null) {  
 result = (new RequestDBService().delete(Integer.*parseInt*(requestID)))  
 ? "Record was successfully deleted!"  
 : "Record was not deleted, error occurred.";  
 }  
 }  
}  
  
req.setAttribute("result", result);  
req.setAttribute("table", "requests");  
req.getRequestDispatcher("/result.jsp").forward(req, resp);  
}  
}

## servlet.ServiceServlet.java

package servlet;  
  
import model.Service;  
import service.ServiceDBService;  
  
import javax.servlet.ServletException;  
import javax.servlet.http.HttpServlet;  
import javax.servlet.http.HttpServletRequest;  
import javax.servlet.http.HttpServletResponse;  
import java.io.IOException;  
import java.util.Arrays;  
import java.util.List;  
  
public class ServiceServlet extends HttpServlet {  
 @Override  
 protected void doGet(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {  
 ServiceDBService serviceDB = new ServiceDBService();  
 String path = req.getPathInfo();  
  
 if (path == null || path.isEmpty() || "/".equals(path)) {  
 List<Service> services = serviceDB.getAll();  
 if (services != null) {  
 req.setAttribute("services", services);  
 req.getRequestDispatcher("/service/services.jsp").forward(req, resp);  
 }  
 } else {  
 try {  
 String[] pathList = path.split("/");  
 String serviceID = pathList[1];  
 switch (serviceID) {  
 case "new" -> {  
 req.getRequestDispatcher("/service/addService.jsp").forward(req, resp);  
 }  
 default -> {  
 Service service = serviceDB.getByID(Integer.*parseInt*(serviceID));  
 if (service != null) {  
 req.setAttribute("service", service);  
 req.getRequestDispatcher("/service/editService.jsp").forward(req, resp);  
 } else {  
 resp.sendRedirect("/404");  
 return;  
 }  
 }  
 }  
 } catch (Exception e) {

resp.sendRedirect("/404");  
 return;  
 }  
 }  
  
 resp.setContentType("text/html");  
 super.doGet(req, resp);  
 }  
  
 @Override  
 protected void doPost(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {  
 req.setCharacterEncoding("UTF-8");  
 String path = req.getPathInfo();  
  
 if (path == null || path.isEmpty()) {  
 return;  
 }  
  
 String result = "Nothing happened.";  
 String serviceID = path.split("/")[1];  
 switch (serviceID) {  
 case "new" -> {  
 if (req.getParameter("add") != null && req.getParameter("cancel") == null) {  
 result = (new ServiceDBService().create(new Service(  
 0,  
 Integer.*parseInt*(req.getParameter("request")),  
 Integer.*parseInt*(req.getParameter("master"))  
 )))  
 ? "Record was successfully added!"  
 : "Record was not added, error occurred.";  
 }  
 }  
 default -> {  
 if (req.getParameter("edit") != null) {  
 result = (new ServiceDBService().edit(Integer.*parseInt*(serviceID), new Service(  
 0,  
 Integer.*parseInt*(req.getParameter("request")),  
 Integer.*parseInt*(req.getParameter("master"))  
 )))  
 ? "Record was successfully edited!"  
 : "Record was not edited, error occurred.";  
 }  
 if (req.getParameter("delete") != null) {  
 result = (new ServiceDBService().delete(Integer.*parseInt*(serviceID)))  
 ? "Record was successfully deleted!"  
 : "Record was not deleted, error occurred.";  
 }  
 }  
 }  
  
 req.setAttribute("result", result);  
 req.setAttribute("table", "services");  
 req.getRequestDispatcher("/result.jsp").forward(req, resp);   
 }  
}

## service.DBService.java

package service;  
  
import java.sql.\*;  
import java.util.ArrayList;  
import java.util.List;  
  
public class DBService {  
 private Connection getConnect() {  
 final String url = "jdbc:postgresql://217.107.219.154:49307/bonch\_2105432";  
 final String user = "bonch\_2105432";  
 final String password = "XnZ2eMhC2gw=";  
  
 Connection con = null;  
 try {  
 Class.*forName*("org.postgresql.Driver");  
 con = DriverManager.*getConnection*(url, user, password);  
 } catch (ClassNotFoundException e) {  
 System.*out*.println(e.getMessage());  
 } catch (SQLException e) {  
 System.*out*.println(e.getMessage());  
 }  
  
 return con;  
 }  
  
 public ResultSet select(String sql) {  
 ResultSet rs = null;  
 try {  
 Connection con = getConnect();  
 Statement st = con.createStatement(ResultSet.*TYPE\_SCROLL\_SENSITIVE*, ResultSet.*CONCUR\_UPDATABLE*);  
 rs = st.executeQuery(sql); } catch (SQLException e) {  
 System.*out*.println(e.getMessage());  
 }  
  
 return rs;  
 }  
  
 public boolean insert(String sql){  
 Statement st = null;  
 try {  
 Connection con = getConnect();  
 st = con.createStatement();  
 int rowsAffected = st.executeUpdate(sql);  
 con.close();  
 if (rowsAffected > 0) {  
 return true;  
 }  
 } catch (Exception e) {  
 System.*out*.println(e.getMessage());;  
 }  
 return false;  
 }  
  
 public boolean update(String sql) {  
 try {  
 Connection con = getConnect();  
 int rowsAffected = con.createStatement(ResultSet.*TYPE\_SCROLL\_SENSITIVE*, ResultSet.*CONCUR\_UPDATABLE*).executeUpdate(sql);  
 con.close();  
 if (rowsAffected > 0) {  
 return true;  
 }  
 } catch (SQLException e) {  
 System.*out*.println(e.getMessage());  
 }  
 return false;  
 }  
  
 public boolean delete(String sql) {  
 try {  
 Connection con = getConnect();  
 int rowsAffected = con.createStatement().executeUpdate(sql);  
 con.close();  
 if (rowsAffected > 0) {  
 return true;  
 }  
 } catch (SQLException e) {  
 System.*out*.println(e.getMessage());  
 }  
 return false;  
 }  
}

## service.HashService.java

package service;  
  
import com.google.common.hash.Hashing;  
import java.nio.charset.StandardCharsets;  
  
public class HashService {  
 public String hash(String toBeHashed) {  
 return Hashing.*sha256*().hashString(toBeHashed, StandardCharsets.*UTF\_8*).toString();  
 }  
}

## service.UserDBService.java

package service;  
  
import model.User;  
  
import java.sql.ResultSet;  
import java.sql.SQLException;  
import java.util.ArrayList;  
import java.util.List;  
  
public class UserDBService {  
 public List<User> getAll() {  
 List<User> users = new ArrayList<>();  
  
 try {  
 ResultSet rs = new DBService().select("SELECT \* FROM users");  
 while (rs.next()) {  
 users.add(new User(  
 rs.getInt("id"),  
 rs.getString("name"),  
 rs.getString("email"),  
 rs.getString("login"),  
 rs.getString("password"),  
 rs.getInt("role")  
 ));  
 }  
 } catch (SQLException e) {  
 System.out.println(e.getMessage());  
 }  
  
 return users;  
 }  
  
 public User getByID(int id) {  
 User user = null;  
  
 try {  
 ResultSet rs = new DBService().select("SELECT \* FROM users WHERE id=" + id);  
 if (rs.next()) {  
 user = new User(  
 rs.getInt("id"),  
 rs.getString("name"),  
 rs.getString("email"),  
 rs.getString("login"),  
 rs.getString("password"),  
 rs.getInt("role")  
 );  
 }  
 } catch (SQLException e) {  
 System.out.println(e.getMessage());  
 }  
  
 return user;  
 }  
  
 public User getByLogin(String login) {  
 User user = null;  
  
 try {  
 ResultSet rs = new DBService().select("SELECT \* FROM users WHERE login=" + login);  
 if (rs.next()) {  
 user = new User(  
 rs.getInt("id"),  
 rs.getString("name"),  
 rs.getString("email"),  
 rs.getString("login"),  
 rs.getString("password"),  
 rs.getInt("role")  
 );  
 }  
 } catch (SQLException e) {  
 System.out.println(e.getMessage());  
 }  
  
 return user;  
 }  
  
 public boolean create(User user) {  
 try {  
 ResultSet rs = new DBService().select("SELECT login FROM users WHERE login=" + user.getLogin());  
 if (rs.next()) {  
 throw new SQLException("User with this login already exists");  
 }  
 } catch (SQLException e) {  
 System.out.println(e.getMessage());  
 return false;  
 } catch (NullPointerException e) {  
 ;  
 }  
  
 String sql = "INSERT INTO users(name, email, login, password, role) values(\'"  
 + user.getName() + "\', \'"  
 + user.getEmail() + "\', \'"  
 + user.getLogin() + "\', \'"  
 + new HashService().hash(user.getPassword()) + "\', "  
 + user.getRole()  
 + ")";  
  
 return new DBService().insert(sql);  
 }  
  
 public boolean edit(int id, User user) {  
 boolean result = true;  
  
 try {  
 ResultSet rs = new DBService().select("SELECT login FROM users WHERE id!=" + id + " AND login=\'" + user.getLogin() + "\'");  
 if (rs.next()) {  
 throw new SQLException("User with this login already exists");  
 }  
 } catch (SQLException e) {  
 System.out.println(e.getMessage());  
 return false;  
 } catch (NullPointerException e) {  
 ;  
 }  
  
 try {  
 ResultSet rs = new DBService().select("SELECT \* FROM users WHERE id=" + id);  
 if (rs.next()) {  
 rs.updateString("name", user.getName());  
 rs.updateString("email", user.getEmail());  
 rs.updateString("login", user.getLogin());  
 rs.updateInt("role", user.getRole());  
 rs.updateRow();  
 } else throw new SQLException();  
 } catch (SQLException e) {  
 System.out.println(e.getMessage());  
 return false;  
 }  
  
 return result;  
 }  
  
 public boolean delete(int id) {  
 String sql = "DELETE FROM users WHERE id = " + id;  
 return new DBService().delete(sql);  
 }  
}

## service.RequestDBService.java

package service;  
  
import model.Request;  
import model.User;  
  
import java.sql.Date;  
import java.sql.ResultSet;  
import java.sql.SQLException;  
import java.util.ArrayList;  
import java.util.List;  
  
public class RequestDBService {  
 public List<Request> getAll() {  
 List<Request> requests = new ArrayList<>();  
  
 try {  
 ResultSet rs = new DBService().select("SELECT \* FROM request");  
 while (rs.next()) {  
 requests.add(new Request(  
 rs.getInt("id"),  
 rs.getInt("client"),  
 rs.getDate("request\_date"),  
 rs.getString("description"),  
 rs.getInt("status")  
 ));  
 }  
 } catch (SQLException e) {  
 System.out.println(e.getMessage());  
 }  
  
 return requests;  
 }  
  
 public Request getByID(int id) {  
 Request request = null;  
  
 try {  
 ResultSet rs = new DBService().select("SELECT \* FROM request WHERE id=" + id);  
 if (rs.next()) {  
 request = new Request(  
 rs.getInt("id"),  
 rs.getInt("client"),  
 rs.getDate("request\_date"),  
 rs.getString("description"),  
 rs.getInt("status")  
 );  
 }  
 } catch (SQLException e) {  
 System.out.println(e.getMessage());  
 }  
  
 return request;  
 }  
  
 public boolean create(Request request) {  
 String sql = "INSERT INTO request(client, request\_date, description, status) values("  
 + request.getClient() + ", \'"  
 + request.getDateString() + "\', \'"  
 + request.getDescription() + "\', \'"  
 + request.getStatus()  
 + "\')";  
 return new DBService().insert(sql);  
 }  
  
 public boolean edit(int id, Request request) {  
 boolean result = true;  
  
 try {  
 ResultSet rs = new DBService().select("SELECT \* FROM request WHERE id=" + id);  
 if (rs.next()) {  
 rs.updateInt("client", request.getClient());  
 rs.updateDate("request\_date", request.getDate());  
 rs.updateString("description", request.getDescription());  
 rs.updateInt("status", request.getStatus());  
 rs.updateRow();  
 } else throw new SQLException();  
 } catch (SQLException e) {  
 System.out.println(e.getMessage());  
 return false;  
 }  
  
 return result;  
 }  
  
 public boolean delete(int id) {  
 String sql = "DELETE FROM request WHERE id=" + id;  
 return new DBService().delete(sql);  
 }  
}

## service.ServiceDBService.java

package service;  
  
import model.Service;  
  
import java.sql.ResultSet;  
import java.sql.SQLException;  
import java.util.ArrayList;  
import java.util.List;  
  
public class ServiceDBService {  
 public List<Service> getAll() {  
 List<Service> services = new ArrayList<>();  
  
 try {  
 ResultSet rs = new DBService().select("SELECT \* FROM service");  
 while (rs.next()) {  
 services.add(new Service(  
 rs.getInt("id"),  
 rs.getInt("request"),  
 rs.getInt("master")  
 ));  
 }  
 } catch (SQLException e) {  
 System.out.println(e.getMessage());  
 }  
 return services;  
 }  
  
 public Service getByID(int id) {  
 Service service = null;  
  
 try {  
 ResultSet rs = new DBService().select("SELECT \* FROM service WHERE id=" + id);  
 if (rs.next()) {  
 service = new Service(  
 rs.getInt("id"),  
 rs.getInt("request"),  
 rs.getInt("master")  
 );  
 }  
 } catch (SQLException e) {  
 System.out.println(e.getMessage());  
 }  
  
 return service;  
 }  
  
 public boolean create(Service service) {  
 String sql = "INSERT INTO service(request, master) values("  
 + service.getRequest() + ", "  
 + service.getMaster()  
 + ")";  
  
 return new DBService().insert(sql);  
 }  
  
 public boolean edit(int id, Service service) {  
 boolean result = true;  
  
 try {  
 ResultSet rs = new DBService().select("SELECT \* FROM service WHERE id=" + id);  
 if (rs.next()) {  
 rs.updateInt("request", service.getRequest());  
 rs.updateInt("master", service.getMaster());  
 rs.updateRow();  
 } else throw new SQLException();  
 } catch (SQLException e) {  
 System.out.println(e.getMessage());  
 return false;  
 }  
  
 return result;  
 }  
  
 public boolean delete(int id) {  
 String sql = "DELETE FROM service WHERE id=" + id;  
 return new DBService().delete(sql);  
 }  
}

## web.xml

<?xml version="1.0" encoding="UTF-8"?>  
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xmlns="http://java.sun.com/xml/ns/javaee"  
 xsi:schemaLocation="http://java.sun.com/xml/ns/javaee  
http://java.sun.com/xml/ns/javaee/web-app\_3\_0.xsd"  
 version="3.0">

<servlet>  
 <servlet-name>userServlet</servlet-name>  
 <servlet-class>servlet.UserServlet</servlet-class>  
 </servlet>  
 <servlet-mapping>  
 <servlet-name>userServlet</servlet-name>  
 <url-pattern>/users/\*</url-pattern>  
 </servlet-mapping>  
   
 <servlet>  
 <servlet-name>requestServlet</servlet-name>  
 <servlet-class>servlet.RequestServlet</servlet-class>  
 </servlet>  
 <servlet-mapping>  
 <servlet-name>requestServlet</servlet-name>  
 <url-pattern>/requests/\*</url-pattern>  
 </servlet-mapping>  
   
 <servlet>  
 <servlet-name>serviceServlet</servlet-name>  
 <servlet-class>servlet.ServiceServlet</servlet-class>  
 </servlet>  
 <servlet-mapping>  
 <servlet-name>serviceServlet</servlet-name>  
 <url-pattern>/services/\*</url-pattern>  
 </servlet-mapping>  
  
 <error-page>  
 <error-code>404</error-code>  
 <location>/404.jsp</location>  
 </error-page>  
  
</web-app>

## index.jsp

<!DOCTYPE html>  
<html>  
<head>  
 <meta charset="UTF-8">  
 <meta name="viewport" content="width=device-width, initial-scale=1.0">  
 <link rel="stylesheet" type="text/css" href="styles/style.css">  
 <title>Home</title>  
</head>  
<body>  
 <header>  
 <img src="images/whitelogo.jpg">  
 <nav>  
 <ul>  
 <li><a href="/">Home</a></li>  
 <li><a href="/users">Users</a></li>  
 <li><a href="/requests">Requests</a></li>  
 <li><a href="/services">Services</a></li>  
 </ul>  
 </nav>  
 </header>  
  
 <div id="content">  
 <h1 class="retroshadow">Welcome!</h1>  
 <p>This is CRUD-system for manipulating tables</p>  
 <p>Choose a table to work with in the navigation bar</p>  
 </div>  
  
 <footer>  
 <p>Specially for Java and Web term papers</p>  
 <p>Tselousov G.A.</p>  
 </footer>  
  
</body>  
</html>

## 404.jsp

<!DOCTYPE html>  
<html>  
<head>  
 <meta charset="UTF-8">  
 <meta name="viewport" content="width=device-width, initial-scale=1.0">  
 <link rel="stylesheet" type="text/css" href="styles/style.css">  
 <title>Not found</title>  
</head>  
<body>  
<header>  
 <img src="images/whitelogo.jpg">  
 <nav>  
 <ul>  
 <li><a href="/">Home</a></li>  
 <li><a href="/users">Users</a></li>  
 <li><a href="/requests">Requests</a></li>  
 <li><a href="/services">Services</a></li>  
 </ul>  
 </nav>  
</header>  
  
<div id="content">  
 <h1 class="retroshadow">404</h1>  
 <p>Sorry! This is not the web page you are looking for.</p>  
</div>  
  
<footer>  
 <p>Specially for Java and Web term papers</p>  
 <p>Tselousov G.A.</p>  
</footer>  
  
</body>  
</html>

## user/users.jsp

<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>  
<%@ page contentType="text/html;charset=UTF-8" language="java" %>  
<html>  
<head>  
 <meta charset="UTF-8">  
 <meta name="viewport" content="width=device-width, initial-scale=1.0">  
 <link rel="stylesheet" type="text/css" href="../styles/style.css">  
 <title>Пользователи</title>  
</head>  
<body>  
 <header>  
 <img src="../images/whitelogo.jpg">  
 <nav>  
 <ul>  
 <li><a href="/">Home</a></li>  
 <li><a href="/users">Users</a></li>  
 <li><a href="/requests">Requests</a></li>  
 <li><a href="/services">Services</a></li>  
 </ul>  
 </nav>  
 </header>  
  
 <div class="recordList">  
 <h1>Users list</h1>  
 <table class="recordTable">  
 <thead>  
 <tr>  
 <th>ID</th>  
 <th>Name</th>  
 <th>E-mail</th>  
 <th>Login</th>  
 <th>Password Hash</th>  
 <th>Role</th>  
 </tr>  
 </thead>  
 <tbody>  
 <c:forEach items="${users}" var="user">  
 <tr class="record" onclick="window.location.href='http://localhost:8080/users/${user.id}'">  
 <td>${user.id}</td>  
 <td>${user.name}</td>  
 <td>${user.email}</td>  
 <td>${user.login}</td>  
 <td>${user.password}</td>  
 <td>${user.role}</td>  
 </tr>  
 </c:forEach>  
 </tbody>  
 </table>  
 <input type="button" value="Add user" onclick="window.location.href='http://localhost:8080/users/new'">  
 </div>  
  
 <footer>  
 <p>Specially for Java and Web term papers</p>  
 <p>Tselousov G.A.</p>  
 </footer>  
  
</body>  
</html>

## user/addUser.jsp

<%@ page contentType="text/html;charset=UTF-8" language="java" %>  
<html>  
<head>  
 <meta charset="UTF-8">  
 <meta name="viewport" content="width=device-width, initial-scale=1.0">  
 <link rel="stylesheet" type="text/css" href="../styles/style.css">  
 <title>Add user</title>  
</head>  
<body>  
 <header>  
 <img src="../images/whitelogo.jpg">  
 <nav>  
 <ul>  
 <li><a href="/">Home</a></li>  
 <li><a href="/users">Users</a></li>  
 <li><a href="/requests">Requests</a></li>  
 <li><a href="/services">Services</a></li>  
 </ul>  
 </nav>  
 </header>  
  
 <div>  
 <form method="post">  
 <p>You can add user here</p>  
 <table>  
 <tr>  
 <td>Name</td>  
 <td><input type="text" name="name"  
 pattern="^[A-Z]{1}[a-z]{0,} [A-Z]{1}[a-z]{0,}$"  
 title="Name contains two words beginning with uppercase letters"  
 required></td>  
 </tr>  
 <tr>  
 <td>E-mail</td>  
 <td><input type="email" name="email"  
 pattern="^[a-z]{1,}[a-z0-9.]{0,}[a-z0-9]{1,}@[a-z]{1,}[a-z0-9]{0,}\.[a-z]{1,}$"  
 title="E-mail is a sequence of letters, digits and points separated by @"  
 required></td>  
 </tr>  
 <tr>  
 <td>Login</td>  
 <td><input type="text" name="login"  
 pattern="^[a-zA-Z0-9\_]{1,}$"  
 title="Login is a sequence of letters, digits and underscore symbols"  
 required></td>  
 </tr>  
 <tr>  
 <td>Password</td>  
 <td><input type="password" name="password"  
 pattern="^(?=.\*[A-Za-z])(?=.\*\d)(?=.\*[@$!%\*#?&-\_])[A-Za-z\d@$!%\*#?&-\_]{8,}$"  
 title="Password must contain minimum eight characters, at least one letter, one number and one special character"  
 required></td>  
 </tr>  
 <tr>  
 <td>Role</td>  
 <td><input type="number" name="role" value="1"  
 min="1"  
 title="Role is a positive number (ID of a role)"  
 required></td>  
 </tr>  
 </table>  
 <input type="submit" name="add" value="Add">  
 <input type="button" onclick="window.location='http://localhost:8080/users'" name="cancel" value="Cancel">  
 </form>  
 </div>  
  
 <footer>  
 <p>Specially for Java and Web term papers</p>  
 <p>Tselousov G.A.</p>  
 </footer>  
  
</body>  
</html>

## user/editUser.jsp

<%@ page contentType="text/html;charset=UTF-8" language="java" %>  
<html>  
<head>  
 <meta charset="UTF-8">  
 <meta name="viewport" content="width=device-width, initial-scale=1.0">  
 <link rel="stylesheet" type="text/css" href="../styles/style.css">  
 <script type="text/javascript" src="../scripts/userValidation.js"></script>  
 <title>Edit user</title>  
</head>  
<body>  
 <header>  
 <img src="../images/whitelogo.jpg">  
 <nav>  
 <ul>  
 <li><a href="/">Home</a></li>  
 <li><a href="/users">Users</a></li>  
 <li><a href="/requests">Requests</a></li>  
 <li><a href="/services">Services</a></li>  
 </ul>  
 </nav>  
 </header>  
  
 <div>  
<%-- <form name="userValidation" action="/users" method="post" onSubmit="return userValidation(this);">--%>  
 <form method="post">  
 <p>You can change something and save record or delete it</p>  
 <table>  
 <tr>  
 <td>Name</td>  
 <td><input type="text" name="name" value="${user.name}"  
 pattern="^[A-Z]{1}[a-z]{0,} [A-Z]{1}[a-z]{0,}$"  
 title="Name contains two words beginning with uppercase letters"  
 required></td>  
 </tr>  
 <tr>  
 <td>E-mail</td>  
 <td><input type="email" name="email" value="${user.email}"  
 pattern="^[a-z]{1,}[a-z0-9.]{0,}[a-z0-9]{1,}@[a-z]{1,}[a-z0-9]{0,}\.[a-z]{1,}$"  
 title="E-mail is a sequence of letters, digits and points separated by @"  
 required></td>  
 </tr>  
 <tr>  
 <td>Login</td>  
 <td><input type="text" name="login" value="${user.login}"  
 pattern="^[a-zA-Z0-9\_]{1,}$"  
 title="Login is a sequence of letters, digits and underscore symbols"  
 required></td>  
 </tr>  
 <tr>  
 <td>Role</td>  
 <td><input type="number" name="role" value="${user.role}"  
 min="1"  
 title="Role is a positive number (ID of a role)"  
 required></td>  
 </tr>  
 </table>  
 <input type="submit" name="edit" value="Save changes">  
 <input type="button" onclick="window.location='http://localhost:8080/users'" name="cancel" value="Cancel">  
 </form>  
 <form method="post">  
 <input type="submit" name="delete" value="Delete">  
 </form>  
 </div>  
  
 <script>  
 </script>  
  
 <footer>  
 <p>Specially for Java and Web term papers</p>  
 <p>Tselousov G.A.</p>  
 </footer>  
  
</body>  
</html>

## request/requests.jsp

<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>  
<%@ page contentType="text/html;charset=UTF-8" language="java" %>  
<html>  
<head>  
 <meta charset="UTF-8">  
 <meta name="viewport" content="width=device-width, initial-scale=1.0">  
 <link rel="stylesheet" type="text/css" href="../styles/style.css">  
 <title>Requests</title>  
</head>  
<body>  
 <header>  
 <img src="../images/whitelogo.jpg">  
 <nav>  
 <ul>  
 <li><a href="/">Home</a></li>  
 <li><a href="/users">Users</a></li>  
 <li><a href="/requests">Requests</a></li>  
 <li><a href="/services">Services</a></li>  
 </ul>  
 </nav>  
 </header>  
  
 <div class="recordList">  
 <h1>Requests list</h1>  
 <table class="recordTable">  
 <thead>  
 <tr>  
 <th>ID</th>  
 <th>Client</th>  
 <th>Date</th>  
 <th>Description</th>  
 <th>Status</th>  
 </tr>  
 </thead>  
 <tbody>  
 <c:forEach items="${requests}" var="request">  
 <tr class="record" onclick="window.location.href='http://localhost:8080/requests/${request.id}'">  
 <td>${request.id}</td>  
 <td>${request.client}</td>  
 <td>${request.date}</td>  
 <td>${request.description}</td>  
 <td>${request.status}</td>  
 </tr>  
 </c:forEach>  
 </tbody>  
 </table>  
 <input type="button" value="Add request" onclick="window.location.href='http://localhost:8080/requests/new'">  
 </div>  
  
 <footer>  
 <p>Specially for Java and Web term papers</p>  
 <p>Tselousov G.A.</p>  
 </footer>  
  
</body>  
</html>

## request/addRequest.jsp

<%@ page contentType="text/html;charset=UTF-8" language="java" %>  
<html>  
<head>  
 <meta charset="UTF-8">  
 <meta name="viewport" content="width=device-width, initial-scale=1.0">  
 <link rel="stylesheet" type="text/css" href="../styles/style.css">  
 <title>Add request</title>  
</head>  
<body>  
<header>  
 <img src="../images/whitelogo.jpg">  
 <nav>  
 <ul>  
 <li><a href="/">Home</a></li>  
 <li><a href="/users">Users</a></li>  
 <li><a href="/requests">Requests</a></li>  
 <li><a href="/services">Services</a></li>  
 </ul>  
 </nav>  
</header>  
  
<div>  
 <form method="post">  
 <p>You can add request here</p>  
 <table>  
 <tr>  
 <td>Client</td>  
 <td><input type="number" name="client" min="1" value="1"  
 title="Client is a positive number (id of a user)" required></td>  
 </tr>  
 <tr>  
 <td>Date</td>  
 <td><input type="date" name="date" required></td>  
 </tr>  
 <tr>  
 <td>Description</td>  
 <td><input type="text" name="description" required></td>  
 </tr>  
 <tr>  
 <td>Status</td>  
 <td><input type="number" name="status" min="1" value="1"  
 title="Status is a positive number (is of a status)" required></td>  
 </tr>  
 </table>  
 <input type="submit" name="add" value="Add">  
 <input type="button" onclick="window.location='http://localhost:8080/requests'" name="cancel" value="Cancel">  
 </form>  
</div>  
  
<footer>  
 <p>Specially for Java and Web term papers</p>  
 <p>Tselousov G.A.</p>  
</footer>  
  
</body>  
</html>

## request/editRequest.jsp

<%@ page contentType="text/html;charset=UTF-8" language="java" %>  
<html>  
<head>  
 <meta charset="UTF-8">  
 <meta name="viewport" content="width=device-width, initial-scale=1.0">  
 <link rel="stylesheet" type="text/css" href="../styles/style.css">  
 <title>Add request</title>  
</head>  
<body>  
<header>  
 <img src="../images/whitelogo.jpg">  
 <nav>  
 <ul>  
 <li><a href="/">Home</a></li>  
 <li><a href="/users">Users</a></li>  
 <li><a href="/requests">Requests</a></li>  
 <li><a href="/services">Services</a></li>  
 </ul>  
 </nav>  
</header>  
  
<div>  
 <form method="post">  
 <p>You can change something and save record or delete it</p>  
 <table>  
 <tr>  
 <td>Client</td>  
 <td><input type="number" min="1" value="${request.client}"  
 title="Client is a positive number (id of a user)" required></td>  
 </tr>  
 <tr>  
 <td>Date</td>  
 <td><input type="date" value="${request.date}" required></td>  
 </tr>  
 <tr>  
 <td>Description</td>  
 <td><input type="text" value="${request.description}" required></td>  
 </tr>  
 <tr>  
 <td>Status</td>  
 <td><input type="number" value="${request.status}" min="1" required></td>  
 </tr>  
 </table>  
 <input type="submit" name="edit" value="Save changes">  
 <input type="button" onclick="window.location='http://localhost:8080/requests'" name="cancel" value="Cancel">  
 </form>  
 <form method="post">  
 <input type="submit" name="delete" value="Delete">  
 </form>  
</div>  
  
<footer>  
 <p>Specially for Java and Web term papers</p>  
 <p>Tselousov G.A.</p>  
</footer>  
  
</body>  
</html>

## service/services.jsp

<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>  
<%@ page contentType="text/html;charset=UTF-8" language="java" %>  
<html>  
<head>  
 <meta charset="UTF-8">  
 <meta name="viewport" content="width=device-width, initial-scale=1.0">  
 <link rel="stylesheet" type="text/css" href="../styles/style.css">  
 <title>Services</title>  
</head>  
<body>  
<header>  
 <img src="../images/whitelogo.jpg">  
 <nav>  
 <ul>  
 <li><a href="/">Home</a></li>  
 <li><a href="/users">Users</a></li>  
 <li><a href="/requests">Requests</a></li>  
 <li><a href="/services">Services</a></li>  
 </ul>  
 </nav>  
</header>  
  
<div class="recordList">  
 <h1>Services list</h1>  
 <table class="recordTable">  
 <thead>  
 <tr>  
 <th>ID</th>  
 <th>Request</th>  
 <th>Master</th>  
 </tr>  
 </thead>  
 <tbody>  
 <c:forEach items="${services}" var="service">  
 <tr class="record" onclick="window.location.href='http://localhost:8080/services/${service.id}'">  
 <td>${service.id}</td>  
 <td>${service.request}</td>  
 <td>${service.master}</td>  
 </tr>  
 </c:forEach>  
 </tbody>  
 </table>  
 <input type="button" value="Add service" onclick="window.location.href='http://localhost:8080/services/new'">  
</div>  
  
<footer>  
 <p>Specially for Java and Web term papers</p>  
 <p>Tselousov G.A.</p>  
</footer>  
  
</body>  
</html>

## service/addService.jsp

<%@ page contentType="text/html;charset=UTF-8" language="java" %>  
<html>  
<head>  
 <meta charset="UTF-8">  
 <meta name="viewport" content="width=device-width, initial-scale=1.0">  
 <link rel="stylesheet" type="text/css" href="../styles/style.css">  
 <title>Add service</title>  
</head>  
<body>  
<header>  
 <img src="../images/whitelogo.jpg">  
 <nav>  
 <ul>  
 <li><a href="/">Home</a></li>  
 <li><a href="/users">Users</a></li>  
 <li><a href="/requests">Requests</a></li>  
 <li><a href="/services">Services</a></li>  
 </ul>  
 </nav>  
</header>  
  
<div>  
 <form method="post">  
 <p>You can add service here</p>  
 <table>  
 <tr>  
 <td>Request</td>  
 <td><input type="number" name="request" min="1" value="1"  
 title="Request is a positive number (id of a request)" required></td>  
 </tr>  
 <tr>  
 <td>Master</td>  
 <td><input type="number" name="master" min="1" value="1"  
 title="Master is a positive number (id of a master)" required></td>  
 </tr>  
 </table>  
 <input type="submit" name="add" value="Add">  
 <input type="button" onclick="window.location='http://localhost:8080/services'" name="cancel" value="Cancel">  
 </form>  
</div>  
  
<footer>  
 <p>Specially for Java and Web term papers</p>  
 <p>Tselousov G.A.</p>  
</footer>  
  
</body>  
</html>

## service/editService.jsp

<%@ page contentType="text/html;charset=UTF-8" language="java" %>  
<html>  
<head>  
 <meta charset="UTF-8">  
 <meta name="viewport" content="width=device-width, initial-scale=1.0">  
 <link rel="stylesheet" type="text/css" href="../styles/style.css">  
 <title>Add service</title>  
</head>  
<body>  
<header>  
 <img src="../images/whitelogo.jpg">  
 <nav>  
 <ul>  
 <li><a href="/">Home</a></li>  
 <li><a href="/users">Users</a></li>  
 <li><a href="/requests">Requests</a></li>  
 <li><a href="/services">Services</a></li>  
 </ul>  
 </nav>  
</header>  
  
<div>  
 <form method="post">  
 <p>You can change something and save record or delete it</p>  
 <table>  
 <tr>  
 <td>Request</td>  
 <td><input type="number" name="request" min="1" value="${service.request}"  
 title="Request is a positive number (id of a request)" required></td>  
 </tr>  
 <tr>  
 <td>Master</td>  
 <td><input type="number" name="master" min="1" value="${service.master}"  
 title="Master is a positive number (id of a master)" required></td>  
 </tr>  
 </table>  
 <input type="submit" name="edit" value="Save changes">  
 <input type="button" onclick="window.location='http://localhost:8080/services'" name="cancel" value="Cancel">  
 </form>  
 <form method="post">  
 <input type="submit" name="delete" value="Delete">  
 </form>  
</div>  
  
<footer>  
 <p>Specially for Java and Web term papers</p>  
 <p>Tselousov G.A.</p>  
</footer>  
  
</body>  
</html>

## styles/style.css

/\* --------------------------- Общие стили всех страниц --------------------------- \*/  
body {  
 font-family: "Avant Garde", Avantgarde, "Century Gothic", CenturyGothic, "AppleGothic", sans-serif;  
 background: linear-gradient(to top right, white, #cbb7fd);  
 color: black;  
 height: 80vh;  
 background-size: cover;  
 margin-top: 80px;  
 padding: 30px;  
}  
  
header img {  
 height: 80px;  
 margin-left: 40px;  
}  
  
header {  
 background-color: white;  
 position: fixed;  
 top: 0;  
 left: 0;  
 right: 0;  
 height: 80px;  
 display: flex;  
 align-items: center;  
 box-shadow: 0 0 25px 0 black;  
}  
  
header \* {  
 display: inline;  
}  
  
header li {  
 margin: 20px;  
}  
  
header li a {  
 color: black;  
 text-decoration: none;  
 transition-property: color;  
 transition-duration: 0.5s;  
 transition-timing-function: ease-out;  
}  
  
header li a:hover {  
 color: #996ee0;  
 transition-property: color;  
 transition-duration: 0.5s;  
 transition-timing-function: ease-out;  
}  
  
footer {  
 background-color: white;  
 position: fixed;  
 bottom: 0;  
 left: 0;  
 right: 0;  
 height: 30px;  
 display: flex;  
 align-items: center;  
 justify-content: space-between;  
 box-shadow: 0 0 5px 0 black;  
 padding: 10px 20px 10px 20px;  
 font-size: 13px;  
}  
  
footer \* {  
 display: inline;  
}  
  
#content h1 {  
 font-family: "Avant Garde", Avantgarde, "Century Gothic", CenturyGothic, "AppleGothic", sans-serif;  
 font-size: 70px;  
 padding: 60px 50px;  
 text-align: center;  
 text-transform: uppercase;  
 text-rendering: optimizeLegibility;  
  
 &.retroshadow {  
 color: #2c2c2c;  
 /\*background-color: #d5d5d5;\*/  
 letter-spacing: .05em;  
 text-shadow:  
 4px 4px 0px #d5d5d5,  
 7px 7px 0px rgba(0, 0, 0, 0.2);  
 }  
}  
  
#content p {  
 font-size: 20px;  
 text-align: center;  
}  
  
input[type=submit], input[type=button], input[type=reset] {  
 margin-top: 5px;  
 background-color: #d2baf6;  
 border-radius: 100px;  
 box-shadow: rgba(80, 44, 187, 0.2) 0 -25px 18px -14px inset, rgba(80, 44, 187, 0.15) 0 1px 2px, rgba(80, 44, 187, 0.15) 0 2px 4px, rgba(80, 44, 187, 0.15) 0 4px 8px,rgba(80, 44, 187, .15) 0 8px 16px,rgba(80, 44, 187, .15) 0 16px 32px;  
 color: #6b41b2;  
 cursor: pointer;  
 display: inline-block;  
 /\*font-family: CerebriSans-Regular,-apple-system,system-ui,Roboto,sans-serif;\*/  
 padding: 7px 20px;  
 text-align: center;  
 text-decoration: none;  
 transition: all 250ms;  
 border: 0;  
 font-size: 16px;  
 user-select: none;  
 -webkit-user-select: none;  
 touch-action: manipulation;  
}  
  
input[type=submit]:hover, input[type=button]:hover, input[type=reset]:hover {  
 box-shadow: rgba(80, 44, 187,.35) 0 -25px 18px -14px inset,rgba(80, 44, 187,.25) 0 1px 2px,rgba(80, 44, 187,.25) 0 2px 4px,rgba(80, 44, 187,.25) 0 4px 8px,rgba(80, 44, 187,.25) 0 8px 16px,rgba(80, 44, 187,.25) 0 16px 32px;  
 transform: scale(1.05) rotate(-1deg);  
}  
  
/\* ------------------------------- Списки записей ------------------------------- \*/  
div.recordList {  
 text-align: center;  
}  
  
.recordTable {  
 margin: 0 auto 15px auto;  
 border-collapse: collapse;  
 /\*margin: 25px 0;\*/  
 font-size: 0.9em;  
 /\*font-family: sans-serif;\*/  
 min-width: 400px;  
 box-shadow: 0 0 20px rgba(0, 0, 0, 0.15);  
}  
  
tr.record {  
 cursor: pointer;  
}  
  
.recordTable thead tr {  
 background-color: #892bcb;  
 color: #ffffff;  
 text-align: left;  
}  
  
.recordTable th,  
.recordTable td {  
 padding: 12px 15px;  
}  
  
.recordTable tbody tr {  
 border-bottom: 1px solid #dddddd;  
}  
  
.recordTable tbody tr:nth-of-type(even) {  
 background-color: #f3f3f3;  
}  
  
.recordTable tbody tr:last-of-type {  
 border-bottom: 2px solid #892bcb;  
}

## pom.xml

<?xml version="1.0" encoding="UTF-8"?>  
<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/xsd/maven-4.0.0.xsd">  
 <modelVersion>4.0.0</modelVersion>  
  
 <groupId>org.bonch</groupId>  
 <artifactId>javacoursach</artifactId>  
 <version>1.0-SNAPSHOT</version>  
 <packaging>war</packaging>  
  
 <properties>  
 <maven.compiler.source>20</maven.compiler.source>  
 <maven.compiler.target>20</maven.compiler.target>  
 <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>  
 </properties>  
  
 <dependencies>  
 <dependency>  
 <groupId>javax.servlet</groupId>  
 <artifactId>javax.servlet-api</artifactId>  
 <version>3.1.0</version>  
 </dependency>  
  
 <dependency>  
 <groupId>org.postgresql</groupId>  
 <artifactId>postgresql</artifactId>  
 <version>42.5.1</version>  
 </dependency>  
  
 <dependency>  
 <groupId>jstl</groupId>  
 <artifactId>jstl</artifactId>  
 <version>1.2</version>  
 </dependency>  
  
 <dependency>  
 <groupId>com.google.guava</groupId>  
 <artifactId>guava</artifactId>  
 <version>32.1.3-jre</version>  
 </dependency>  
 </dependencies>  
  
</project>