

Оглавление

Цели и задачи лабораторной работы.....	2
Формулировка индивидуального задания	2
Перечень библиотек, основных классов и методов, использованных в программе.....	2
Результаты работы программы.....	6
Анализ полученных результатов.....	6
Выводы о проделанной работе.....	8
Список использованных источников	9
Приложение А (обязательное) Листинги программы.....	10
Приложение Б (обязательное) Схемы программы	21
Приложение В (обязательное) Скриншоты интерфейса.....	32

Цели и задачи лабораторной работы

Целью данной лабораторной работы является написание простейшего серверного приложения на языке программирования Java.

Формулировка индивидуального задания

Задание по вариантам предполагает создание страницы с аутентификацией для своего проекта. Необходимо предусмотреть как минимум два вида пользователей с разными итоговыми страницами, на которые осуществляется переход в зависимости от введенного логина и пароля.

Спроектировать обработку ввода неправильного логина или пароля с выводом соответствующего сообщения как о несуществующем логине, так и о вводе несоответствующего пароля для выбранного логина.

Вариант №5. Образовательный сайт по ИТ-программированию.

Перечень библиотек и основных классов, методов, использованных в программе

При написании данного приложения использовались 9 классов, которые условно можно разделить на три различных группы-пакета: пакет сервлетов (*servlets*), пакет фильтров (*filters*) и пакет данных (*data*).

1. В пакете данных находится единственный класс *Data*, который использует для своей работы неупорядоченные коллекции *HashMap*, которые хранят элементы (объекты, записи данных) в виде пар ключ-значение. В работе класса используется две коллекции: *loginInfo* – для хранения пары «имя пользователя – пароль» и *roleInfo* – для хранения пары «имя пользователя – роль».

Переменная *selector* задается текущим активным сервлетом и используется файлом *_menu.jsp* для корректного отображения (выделения цветом) текущего пункта меню.

Переменная *currentRole* определяет текущую роль конкретного пользователя, влияющую на права доступа к каждой странице.

Также класс имеет статический инициализатор (создает трех различных пользователей с соответствующими именами, паролями и ролями), 2 метода-сеттера и 4 метода-геттера.

2. Второй пакет (*filters*) включает в себя два однотипных класса *AdminFilter* и *UserFilter*, каждый из которых реализует интерфейс *javax.servlet.Filter*.

Оба класса включают в себя три метода – *init* (инициализация фильтра), *destroy* (выгрузка фильтра) и *doFilter* (собственно обработка фильтра – фильтрация запроса).

Класс *UserFilter* принимает запрос при обращении по адресу */userTask*, после чего задает соответствующее значение переменной *Data.selector* и проводит сравнение текущей роли пользователя со значениями “*user*” и “*admin*”. При положительном результате сравнения происходит перенаправление запроса по запрашиваемому адресу на сервлет *UserTaskServlet*, при отрицательном – на страницу *accessDeniedView.jsp* с передачей странице соответствующего текстового атрибута-сообщения.

Аналогично класс *AdminFilter* принимает запрос при обращении по адресу */adminTask*, после чего задает соответствующее значение переменной *Data.selector* и проводит сравнение текущей роли пользователя с единственным значением “*admin*”. При положительном результате сравнения происходит перенаправление запроса по запрашиваемому адресу на сервлет *AdminTaskServlet*, при отрицательном – на страницу *accessDeniedView.jsp* с передачей странице соответствующего текстового атрибута-сообщения.

3. Третий пакет (*servlets*) включает в себя шесть однотипных классов, каждый из которых расширяет абстрактный класс *jakarta.servlet.http.HttpServlet*, имеет методы *doGet* и *doPost*, но имеет свои особенности реализации.

- 1) *AdminTaskServlet* – принимает запрос от фильтра *AdminFilter*, задает соответствующее значение переменной *Data.selector* и перенаправляет запрос на страницу *adminTaskView.jsp*.
- 2) *HomeServlet* – принимает запрос при обращении по адресам */index* и */homeView.jsp*, создает новую сессию со стартовым атрибутом *name* и его значением по умолчанию *guest*, задает соответствующее значение переменной *Data.selector* и перенаправляет запрос на страницу *homeView.jsp*.
- 3) *LoginServlet*
 - метод *doGet* – принимает запрос при обращении по адресу */login* задает соответствующее значение переменной *Data.selector*, и перенаправляет запрос на страницу *loginView.jsp*;
 - метод *doPost* – принимает запрос после заполнения формы на странице *loginView.jsp* (имя пользователя и пароль); проводит сравнение полученных значений с данными, хранящимися в коллекции *Data.loginInfo*; при совпадении обоих значений происходит завершение предыдущей сессии, установление новой с новым значением атрибута имени и определением новой роли; перенаправляет запрос на соответствующую роли страницу; при несовпадении данных перенаправляет запрос на страницу *accessDeniedView.jsp* с передачей передаче странице соответствующего текстового атрибута-сообщения.
- 4) *LogoutServlet* – принимает запрос при обращении по адресу */logout*, завершает текущую сессию, задает соответствующее значение переменной *Data.selector* и перенаправляет запрос на домашнюю страницу.
- 5) *UserInfoServlet* – принимает запрос при обращении по адресу */userInfo*, задает соответствующее значение переменной *Data.selector* и перенаправляет запрос на страницу *userInfoView.jsp*.

- 6) *UserTaskServlet* – принимает запрос от фильтра *UserFilter*, задает соответствующее значение переменной *Data.selector* и перенаправляет запрос на страницу *userTaskView.jsp*.

Результаты работы программы

Для взаимодействия с пользователем и визуализации результатов работы приложения использовались 6 страниц *jsp*, каждая из которых включает в себя общий навигационный элемент меню *_menu.jsp*.

1. Страница *accessDeniedView.jsp* – принимает и выводит на экран передаваемые другими классами сообщения о причинах отказа в доступе к защищенным страницам (прил. В, рис. 2, 3, 7, 9, 10, 11).

2. Страница *adminTaskView.jsp* – отображает информацию обо всех пользователях при наличии авторизации в качестве администратора (прил. В, рис. 13).

3. Страница *homeView.jsp* – домашняя страница, предлагающая всем пользователям сайта краткую информацию о некоторых образовательных ресурсах с возможностью непосредственного перехода на указанные сайты (прил. В, рис. 1).

4. Страница *loginView.jsp* – предлагает для пользователей сайта форму (поля имени пользователя и пароля) авторизации на сайте (прил. В, рис. 5).

5. Страница *userInfoView.jsp* – отображает информацию о текущем пользователе сайта (прил. В, рис. 6, 12).

6. Страница *userTaskView.jsp* – отображает защищенный раздел пользователя, позволяющий зарегистрированным пользователям возможность просмотра и скачивания четырех книг в формате pdf (прил. В, рис. 4, 8, 14).

Анализ полученных результатов

В результате выполненной работы получено простейшее серверное приложение, выполненное на основе языка Java – сервлетов и jsp-страниц.

Приложение представляет из себя набор веб-страниц с тремя различными уровнями доступа:

- 1) уровень “guest” присваивается любому неавторизованному пользователю по умолчанию и позволяет просматривать только главную страницу, содержащую ссылки на информационные ресурсы и текущие сведения о самом пользователе (при нажатии на его имя в меню);
- 2) уровень “user” присваивается после авторизации на сайте уже зарегистрированным пользователям и позволяет просматривать помимо информации, доступной предыдущему уровню, дополнительную страницу пользователя, содержащую ссылки на 4 книги, доступные для просмотра и скачивания;
- 3) уровень “admin” присваивается после авторизации на сайте уже зарегистрированным пользователям и позволяет просматривать помимо информации, доступной предыдущим двум уровням, дополнительную страницу администратора, содержащую информацию обо всех зарегистрированных на сайте пользователях.

В случае необходимости имеющийся каркас веб-приложения легко масштабируется путем добавления дополнительных страниц, дополнительных ролей пользователей, дополнительных фильтров, а также – собственно дополнительных пользователей.

Выводы о проделанной работе

1. Отработаны основные этапы написания простейшего серверного приложения на языке программирования Java.
2. Реализована система аутентификации пользователей приложения, включающая в себя три вида пользователей с разными итоговыми страницами, а также обработку ввода неправильного логина или пароля с выводом соответствующих сообщений.
3. Разработано серверное приложение «Образовательный сайт по ИТ-программированию».

Список использованных источников

1. Морозова Ю.В. Разработка интернет-приложений: методические указания по выполнению лабораторной работы и организации самостоятельной работы для студентов, обучающихся с применением ДОТ. Томск: ФДО ТУСУР, 2020. 60 с.
2. Руководства Java Servlet/JSP [Электронный ресурс] // Сайт: betacode.net. – Режим доступа: <https://betacode.net/10979/servlet-jsp> (дата обращения: 20.08.2024).

Приложение А

(обязательное)

Листинги программы

```
package sidminik.data;

import java.util.HashMap;

public class Data {
    private static String selector;
    private static String currentRole;
    private static HashMap<String, String> loginInfo = new
HashMap<>();
    private static HashMap<String, String> roleInfo = new
HashMap<>();

    static {
        loginInfo.put("guest", "");
        roleInfo.put("guest", "guest");

        loginInfo.put("user_1", "123");
        roleInfo.put("user_1", "user");

        loginInfo.put("admin_1", "789");
        roleInfo.put("admin_1", "admin");
    }

    public static String getSelector() {
        return selector;
    }

    public static void setSelector(String selector) {
        Data.selector = selector;
    }

    public static String getCurrentRole() {
        return currentRole;
    }

    public static void setCurrentRole(String currentRole) {
        Data.currentRole = currentRole;
    }

    public static HashMap<String, String> getLoginInfo() {
        return loginInfo;
    }

    public static HashMap<String, String> getRoleInfo() {
        return roleInfo;
    }
}
```

```

package sidminik.filters;

import sidminik.data.Data;

import jakarta.servlet.*;
import jakarta.servlet.annotation.WebFilter;

import java.io.IOException;

@WebFilter("/adminTask")
public class AdminFilter implements Filter {
    @Override
    public void init(FilterConfig fConfig) throws
ServletException {
    }

    @Override
    public void destroy() {
    }

    @Override
    public void doFilter(ServletRequest request, ServletResponse
response, FilterChain chain)
        throws IOException, ServletException {

        Data.setSelector("admin");
        if ("admin".equals(Data.getCurrentRole())) {
            chain.doFilter(request, response);
        }
        else {
            String errorMessage = "Для входа на данную страницу
необходимо обладать правами администратора. Пожалуйста,
авторизируйтесь!";

            request.setAttribute("errorMessage", errorMessage);

            RequestDispatcher dispatcher //
                =
request.getServletContext().getRequestDispatcher("/WEB-
INF/pages/accessDeniedView.jsp");

            dispatcher.forward(request, response);
        }
    }
}

```

```

package sidminik.filters;

import sidminik.data.Data;

import jakarta.servlet.*;
import jakarta.servlet.annotation.WebFilter;

import java.io.IOException;

@WebFilter("/userTask")
public class UserFilter implements Filter {
    @Override
    public void init(FilterConfig fConfig) throws
ServletException {
    }

    @Override
    public void destroy() {
    }

    @Override
    public void doFilter(ServletRequest request, ServletResponse
response, FilterChain chain)
        throws IOException, ServletException {

        Data.setSelector("user");
        if ("user".equals(Data.getCurrentRole()) ||
"admin".equals(Data.getCurrentRole())) {
            chain.doFilter(request, response);
        }
        else {
            String errorMessage = "Для входа на данную страницу
необходимо обладать правами зарегистрированного пользователя.
Пожалуйста, авторизируйтесь!";

            request.setAttribute("errorMessage", errorMessage);

            RequestDispatcher dispatcher //
                =
request.getServletContext().getRequestDispatcher("/WEB-
INF/pages/accessDeniedView.jsp");

            dispatcher.forward(request, response);
        }
    }
}

```

```

package sidminik.servlets;

import jakarta.servlet.RequestDispatcher;
import jakarta.servlet.ServletException;
import jakarta.servlet.annotation.WebServlet;
import jakarta.servlet.http.HttpServlet;
import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpServletResponse;
import sidminik.data.Data;

import java.io.IOException;

@WebServlet("/adminTask")
public class AdminTaskServlet extends HttpServlet {
    public AdminTaskServlet() {
        super();
    }

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

        Data.setSelector("admin");
        RequestDispatcher dispatcher //
            = this.getServletContext()//
                .getRequestDispatcher("/WEB-INF/pages/adminTaskView.jsp");

        dispatcher.forward(request, response);
    }

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

        doGet(request, response);
    }
}

```

```

package sidminik.servlets;

import jakarta.servlet.RequestDispatcher;
import jakarta.servlet.ServletException;
import jakarta.servlet.annotation.WebServlet;
import jakarta.servlet.http.HttpServlet;
import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpServletResponse;
import jakarta.servlet.http.HttpSession;
import sidminik.data.Data;

import java.io.IOException;

@WebServlet({"/index", "/homeView.jsp"})
public class HomeServlet extends HttpServlet {
    public HomeServlet() {
        super();
    }

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

        HttpSession session = request.getSession();

        String name = (String) session.getAttribute("name");
        if (name == null) {
            session.setAttribute("name", "guest");
        }

        if (Data.getCurrentRole() == null) {
            Data.setCurrentRole("guest");
        }

        Data.setSelector("home");
        RequestDispatcher dispatcher //
            =
this.getServletContext().getRequestDispatcher("/WEB-INF/pages/homeView.jsp");

        dispatcher.forward(request, response);
    }

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

        doGet(request, response);
    }
}

```

```

package sidminik.servlets;

import jakarta.servlet.RequestDispatcher;
import jakarta.servlet.ServletException;
import jakarta.servlet.annotation.WebServlet;
import jakarta.servlet.http.HttpServlet;
import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpServletResponse;
import jakarta.servlet.http.HttpSession;
import sidminik.data.Data;

import java.io.IOException;

@WebServlet("/login")
public class LoginServlet extends HttpServlet {
    public LoginServlet() {
        super();
    }

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

        Data.setSelector("login");
        RequestDispatcher dispatcher //
            =
this.getServletContext().getRequestDispatcher("/WEB-INF/pages/loginView.jsp");

        dispatcher.forward(request, response);
    }

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

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

        if (Data.getLoginInfo().containsKey(userName) &&
password.equals(Data.getLoginInfo().get(userName))) {
            request.getSession().invalidate();
            HttpSession session = request.getSession();
            session.setAttribute("name", userName);

Data.setCurrentRole(Data.getRoleInfo().get(userName));
            RequestDispatcher dispatcher;
            if ("user".equals(Data.getCurrentRole())) {
                Data.setSelector("user");
                dispatcher =

```

```

request.getServletContext().getRequestDispatcher("/WEB-INF/pages/userTaskView.jsp");
    } else {
        Data.setSelector("admin");
        dispatcher =
request.getServletContext().getRequestDispatcher("/WEB-INF/pages/adminTaskView.jsp");
    }
    dispatcher.forward(request, response);
} else {
    String errorMessage;
    if (!Data.getLoginInfo().containsKey(userName) &&
!Data.getLoginInfo().containsValue(password)) {
        errorMessage = "Указаны неверные имя
пользователя и пароль! Пожалуйста, попробуйте еще раз.";
    } else if
(!Data.getLoginInfo().containsKey(userName)) {
        errorMessage = "Указано неверное имя
пользователя! Пожалуйста, попробуйте еще раз.";
    } else {
        errorMessage = "Указан неверный пароль!
Пожалуйста, попробуйте еще раз.";
    }
    request.setAttribute("errorMessage", errorMessage);
    RequestDispatcher dispatcher //
    =
request.getServletContext().getRequestDispatcher("/WEB-INF/pages/accessDeniedView.jsp");
    dispatcher.forward(request, response);
}
}
}

```



```

package sidminik.servlets;

import jakarta.servlet.ServletException;
import jakarta.servlet.annotation.WebServlet;
import jakarta.servlet.http.HttpServlet;
import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpServletResponse;
import sidminik.data.Data;

import java.io.IOException;

@WebServlet("/logout")
public class LogoutServlet extends HttpServlet {
    public LogoutServlet() {
        super();
    }

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

        request.getSession().invalidate();
        Data.setCurrentRole("guest");

        response.sendRedirect(request.getContextPath() + "/");
    }

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

```

```

package sidminik.servlets;

import jakarta.servlet.RequestDispatcher;
import jakarta.servlet.ServletException;
import jakarta.servlet.annotation.WebServlet;
import jakarta.servlet.http.HttpServlet;
import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpServletResponse;
import sidminik.data.Data;

import java.io.IOException;

@WebServlet("/userInfo")
public class UserInfoServlet extends HttpServlet {
    public UserInfoServlet() {
        super();
    }

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

        Data.setSelector("info");
        RequestDispatcher dispatcher //
            =
this.getServletContext().getRequestDispatcher("/WEB-INF/pages/userInfoView.jsp");

        dispatcher.forward(request, response);
    }

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

        doGet(request, response);
    }
}

```

```

package sidminik.servlets;

import jakarta.servlet.RequestDispatcher;
import jakarta.servlet.ServletException;
import jakarta.servlet.annotation.WebServlet;
import jakarta.servlet.http.HttpServlet;
import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpServletResponse;

import java.io.IOException;

@WebServlet("/userTask")
public class UserTaskServlet extends HttpServlet {
    public UserTaskServlet() {
        super();
    }

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

        RequestDispatcher dispatcher //
            = this.getServletContext()//
                .getRequestDispatcher("/WEB-INF/pages/userTaskView.jsp");

        dispatcher.forward(request, response);
    }

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

        doGet(request, response);
    }
}

```

_menu.jsp

```
<%@ page import="sidminik.data.Data" %>
<%@ page pageEncoding="UTF-8" contentType="text/html" %>

<ul class="menu">
    <li class="<%=("home".equals(Data.getSelector())) ?
"selected":"none") %>">
        <a
href="${pageContext.request.contextPath}/index">Главная</a>
    </li>
    <li class="<%=("user".equals(Data.getSelector())) ?
"selected":"none") %>">
        <a
href="${pageContext.request.contextPath}/userTask">Раздел
пользователя</a>
    </li>
    <li class="<%=("admin".equals(Data.getSelector())) ?
"selected":"none") %>">
        <a
href="${pageContext.request.contextPath}/adminTask">Раздел
администратора</a>
    </li>
    <li class="none" style="float:right">
        <a
href="${pageContext.request.contextPath}/logout">Выход</a>
    </li>
    <li class="<%=("login".equals(Data.getSelector())) ?
"selected":"none") %>" style="float:right">
        <a
href="${pageContext.request.contextPath}/login">Вход</a>
    </li>
    <li class="<%=("info".equals(Data.getSelector())) ?
"selected":"none") %>" style="float:right">
        <a href="${pageContext.request.contextPath}/userInfo">[
<span><%= session.getAttribute("name") %></span> ]</a>
    </li>
</ul>
```

accessDenied.jsp

```
<%@ page pageEncoding="UTF-8" contentType="text/html" %>

<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <style><%@ include file="../../style.css"%></style>
    <title>Доступ запрещен</title>
  </head>
  <body>
    <jsp:include page="../../elements/_menu.jsp"></jsp:include>
    <br/><br/>
    <h3 style="color:red;"><%=
request.getAttribute("errorMessage") %></h3>
  </body>
</html>
```

adminTaskView.jsp

```
<%@ page import="sidminik.data.Data" %>
<%@ page pageEncoding="UTF-8" contentType="text/html" %>

<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <style><%@ include file="../../../style.css"%></style>
        <title>Администратор</title>
    </head>

    <body>
        <jsp:include page="../../../elements/_menu.jsp"></jsp:include>
        <h3>Информационная страница администратора сайта</h3>
        <table>
            <tr>
                <th>Имя</th>
                <th>Пароль</th>
                <th>Роль</th>
            </tr>
            <tr>
                <td>guest</td>
                <td><%= Data.getLoginInfo().get("guest") %></td>
                <td><%= Data.getRoleInfo().get("guest") %></td>
            </tr>
            <tr>
                <td>user_1</td>
                <td><%= Data.getLoginInfo().get("user_1")
%></td>
                <td><%= Data.getRoleInfo().get("user_1") %></td>
            </tr>
            <tr>
                <td>admin_1</td>
                <td><%= Data.getLoginInfo().get("admin_1")
%></td>
                <td><%= Data.getRoleInfo().get("admin_1")
%></td>
            </tr>
        </table>
    </body>
</html>
```

homeview.jsp

```
<%@ page import="sidminik.data.Data" %>
<%@ page pageEncoding="UTF-8" contentType="text/html" %>

<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <style><%@ include file="../../../style.css"%></style>
        <title>Администратор</title>
    </head>

    <body>
        <jsp:include page="../../../elements/_menu.jsp"></jsp:include>
        <h3>Информационная страница администратора сайта</h3>
        <table>
            <tr>
                <th>Имя</th>
                <th>Пароль</th>
                <th>Роль</th>
            </tr>
            <tr>
                <td>guest</td>
                <td><%= Data.getLoginInfo().get("guest") %></td>
                <td><%= Data.getRoleInfo().get("guest") %></td>
            </tr>
            <tr>
                <td>user_1</td>
                <td><%= Data.getLoginInfo().get("user_1")
%></td>
                <td><%= Data.getRoleInfo().get("user_1") %></td>
            </tr>
            <tr>
                <td>admin_1</td>
                <td><%= Data.getLoginInfo().get("admin_1")
%></td>
                <td><%= Data.getRoleInfo().get("admin_1")
%></td>
            </tr>
        </table>
    </body>
</html>
```

loginView.jsp

```
<%@ page pageEncoding="UTF-8" contentType="text/html" %>

<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <style><%@ include file="../../style.css"%></style>
    <title>Вход на сайт</title>
  </head>
  <body>
    <jsp:include page="../../elements/_menu.jsp"></jsp:include>
    <h3>Страница входа</h3>
    <p style="color: red;">${errorString}</p>
    <form method="POST"
action="${pageContext.request.contextPath}/login">
      <input type="hidden" name="redirectId"
value="${param.redirectId}" />
      <table>
        <tr>
          <td>Имя пользователя</td>
          <td><input type="text" name="userName"
value= "${userName}" /> </td>
        </tr>
        <tr>
          <td>Пароль</td>
          <td><input type="password" name="password"
value= "${password}" /> </td>
        </tr>
        <tr>
          <td colspan="2">
            <input style="float: right"
type="submit" value= "Войти" />
          </td>
        </tr>
      </table>
    </form>
    <p style="color:red;">Возможен вход под следующими
учетными записями:</p>
    <p>Имя пользователя: user_1/ Пароль: 123</p>
    <p>Имя пользователя: admin_1/ Пароль: 789</p>
  </body>
</html>
```


userInfoView.jsp

```
<%@ page import="sidminik.data.Data" %>
<%@ page pageEncoding="UTF-8" contentType="text/html" %>

<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <style><%@ include file="../../../style.css"%></style>
        <title>Информация о пользователе</title>
    </head>
    <body>
        <jsp:include page="../../../elements/_menu.jsp"></jsp:include>
        <h3>Информация о пользователе:</h3>
        <table>
            <tr>
                <th>Имя</th>
                <td><%= session.getAttribute("name") %></td>
            </tr>
            <tr>
                <th>Пароль</th>
                <td><%=
Data.getLoginInfo().get(session.getAttribute("name")) %></td>
            </tr>
            <tr>
                <th>Роль</th>
                <td><%=
Data.getRoleInfo().get(session.getAttribute("name")) %></td>
            </tr>
        </table>
    </body>
</html>
```

userTaskView.jsp

```
<% @ page pageEncoding="UTF-8" contentType="text/html" %>

<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <style><% @ include file="../../style.css"%></style>
    <title>Пользователь</title>
  </head>
  <body>
    <jsp:include page="../../elements/_menu.jsp"></jsp:include>
    <h3>Добрый день! Это защищенная страница пользователя!</h3>
    <table class="site">
      <tr>
        <td class="col_1">
          <a
href="${pageContext.servletContext.contextPath}/books/2005_Swing.pdf">
            
          </a>
        </td>
        <td class="col_2">
          <p>Портянкин И. Swing: Эффектные пользовательские
интерфейсы, 2011. </p>
        </td>
      </tr>
      <tr>
        <td class="col_1">
          <a
href="${pageContext.servletContext.contextPath}/books/2020_Java_Concurrency
_на_практике.pdf">
            
          </a>
        </td>
        <td class="col_2">
          <p>Гетц Б. и соавт. Java Concurrency на практике, 2020</p>
        </td>
      </tr>
    </table>
```

```

        <td class="col_1">
            <a
href="\${pageContext.servletContext.contextPath}/books/2022_Классические_зад
ачи_на_Java.pdf">
                
            </a>
        </td>
        <td class="col_2">
            <p>Копец Д. Классические задачи Computer Science на языке
Java, 2022.</p>
        </td>
    </tr>
    <tr>
        <td class="col_1">
            <a
href="\${pageContext.servletContext.contextPath}/books/2023_Программируем_н
а_Java.pdf">
                
            </a>
        </td>
        <td class="col_2">
            <p>Лой М., Нимайер П., Лук Д. Программируем на Java,
2023.</p>
        </td>
    </tr>
</table>
</body>
</html>

```

style.css

```
* {  
    font-family: Verdana,Geneva,sans-serif;  
}  
  
h1, h2, h3, h4, h5, h6, p, input {  
    margin: 10px;  
}  
  
.menu {  
    list-style-type: none;  
    margin: 0;  
    padding: 0;  
    overflow: hidden;  
    background-color: #333;  
}  
  
.menu li {  
    float: left;  
}  
  
.menu li a {  
    display: block;  
    color: white;  
    text-align: center;  
    padding: 14px 16px;  
    text-decoration: none;  
}  
  
.none > a:hover {  
    background-color: #111;  
}  
  
.selected, .selected > a:hover {  
    background-color: darkred;  
}  
  
table {  
    border: 3px solid grey;  
    margin: 10px;  
    width: 600px;
```

```

}

th, td {
    height: 100px;
    border: 1px solid grey;
}

.site {
    width: 99%;
}

.col_1 {
    width: 100px;
}

img {
    margin: 10px;
    height: 120px;
    width: 120px;
    box-shadow: 0px 0px 10px 10px black;
}

td > a:hover > img {
    box-shadow: 0px 0px 10px 10px darkred;
}

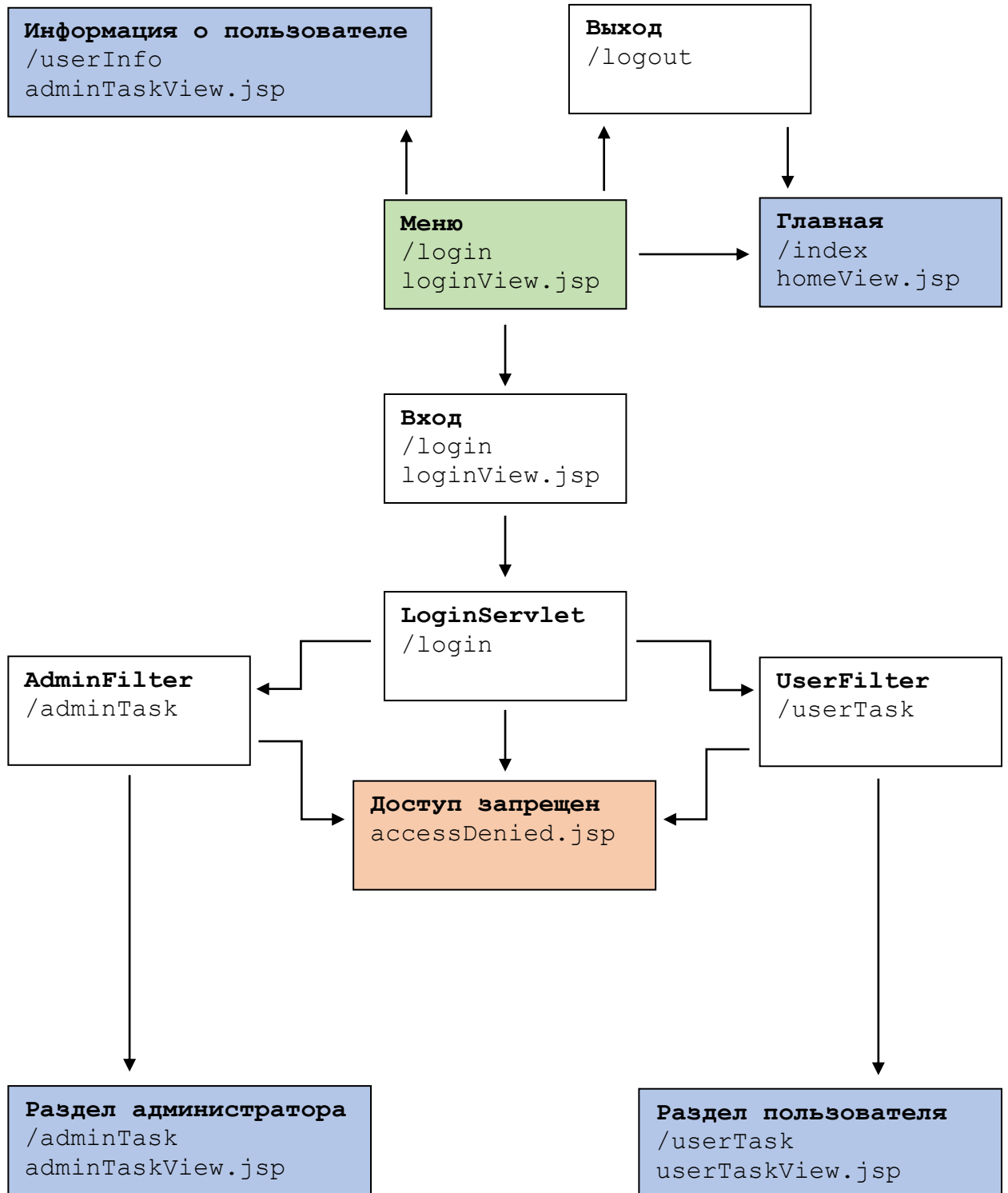
```

web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns="https://jakarta.ee/xml/ns/jakartaee"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="https://jakarta.ee/xml/ns/jakartaee
https://jakarta.ee/xml/ns/jakartaee/web-app_5_0.xsd"
  version="5.0">

  <welcome-file-list>
    <welcome-file>
      homeView.jsp
    </welcome-file>
  </welcome-file-list>
</web-app>
```

Приложение Б
(обязательное)
Схема программы



Приложение В

(обязательное)

Скриншоты интерфейса

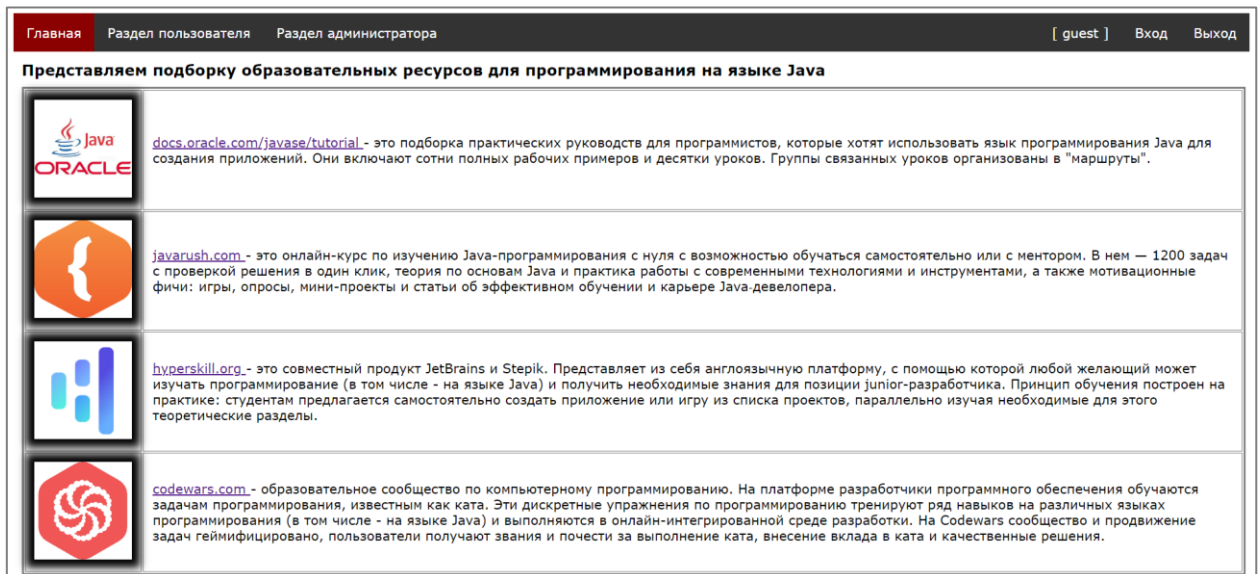


Рисунок 1 – Главная страница, незарегистрированный пользователь (guest)

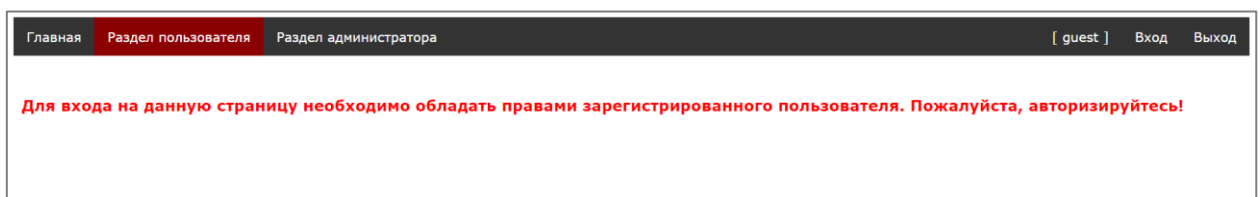


Рисунок 2 – Раздел пользователя, незарегистрированный пользователь (guest)

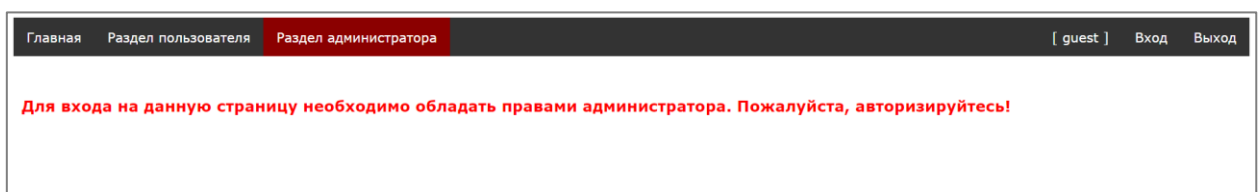


Рисунок 3 – Раздел администратора, незарегистрированный пользователь (guest)

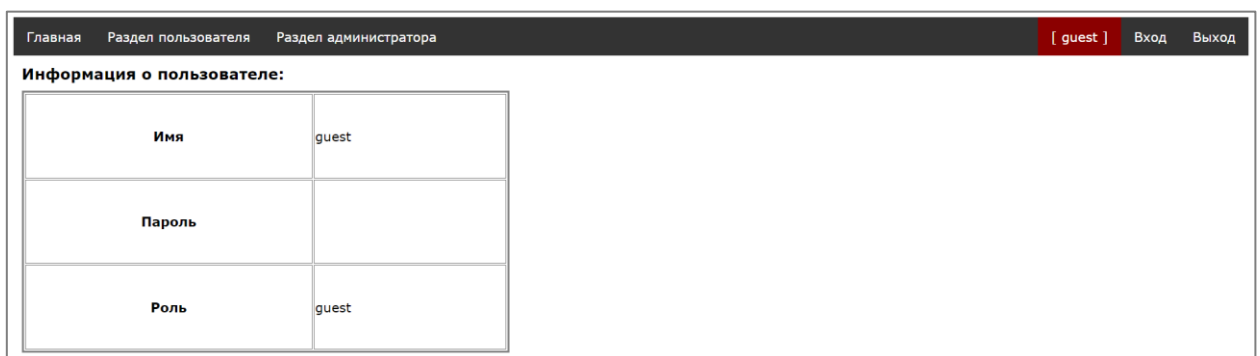


Рисунок 4 – Информация о пользователе, незарегистрированный пользователь (guest)

Главная	Раздел пользователя	Раздел администратора	[guest]	Вход	Выход
---------	---------------------	-----------------------	-----------	------	-------

Страница входа

Имя пользователя	<input type="text"/>
Пароль	<input type="password"/>
<input type="button" value="Войти"/>	

Возможен вход под следующими учетными записями:

Имя пользователя: user_1/ Пароль: 123

Имя пользователя: admin_1/ Пароль: 789

Рисунок 5 – Страница авторизации на сайте, незарегистрированный пользователь (guest)

Главная	Раздел пользователя	Раздел администратора	[user_1]	Вход	Выход
---------	---------------------	-----------------------	------------	------	-------

Добрый день! Это защищенная страница пользователя!





	Портянкин И. Swing: Эффектные пользовательские интерфейсы, 2011.
	Гетц Б. и соавт. Java Concurrency на практике, 2020
	Копец Д. Классические задачи Computer Science на языке Java, 2022.
	Лой М., Нимайер П., Лук Д. Програмируем на Java, 2023.

Рисунок 6 – Раздел пользователя после авторизации на сайте, зарегистрированный пользователь (user_1)

Главная	Раздел пользователя	Раздел администратора	[user_1]	Вход	Выход
---------	---------------------	-----------------------	------------	------	-------

Для входа на данную страницу необходимо обладать правами администратора. Пожалуйста, авторизуйтесь!

Рисунок 7 – Раздел администратора, зарегистрированный пользователь (user_1)

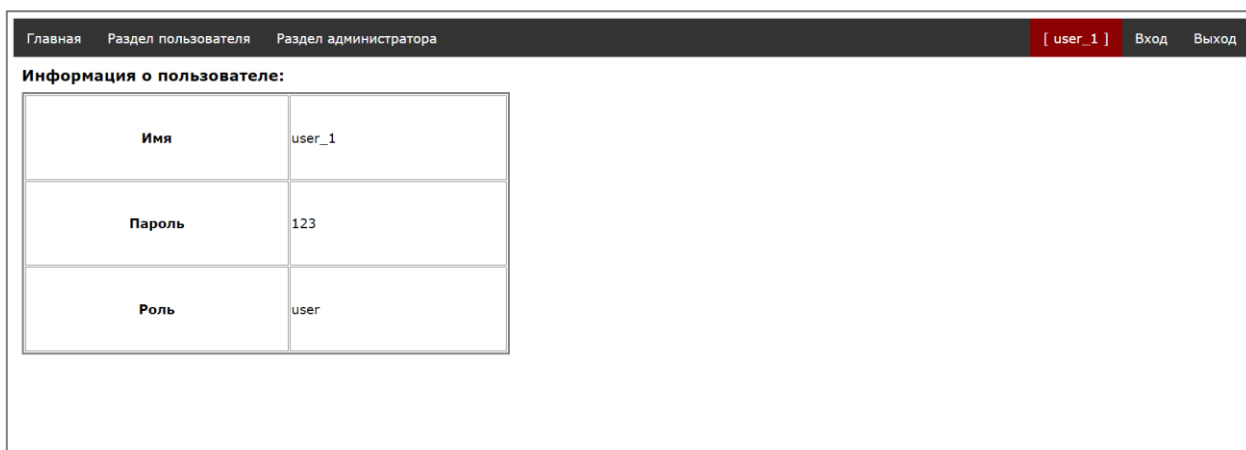


Рисунок 8 – Информация о пользователе, зарегистрированный пользователь (user_1)

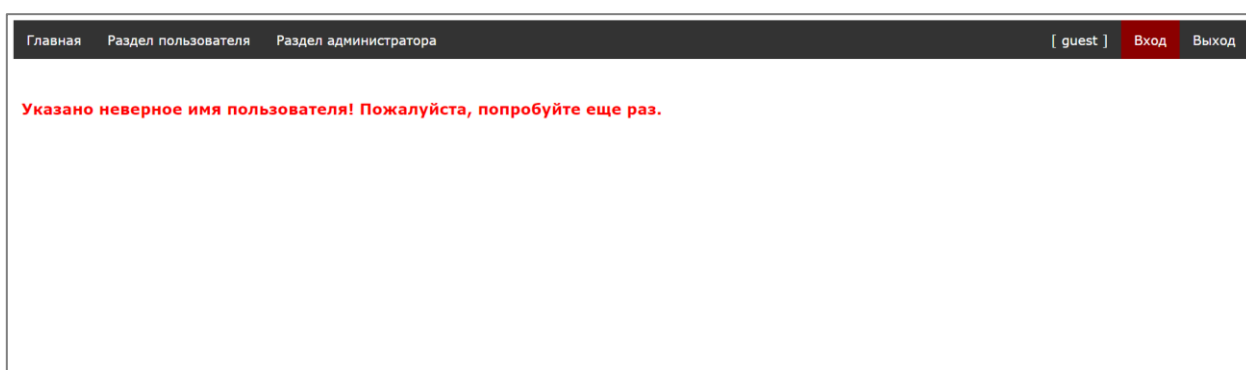


Рисунок 9 – Информация о неверно введенном имени пользователя



Рисунок 10 – Информация о неверно введенном пароле

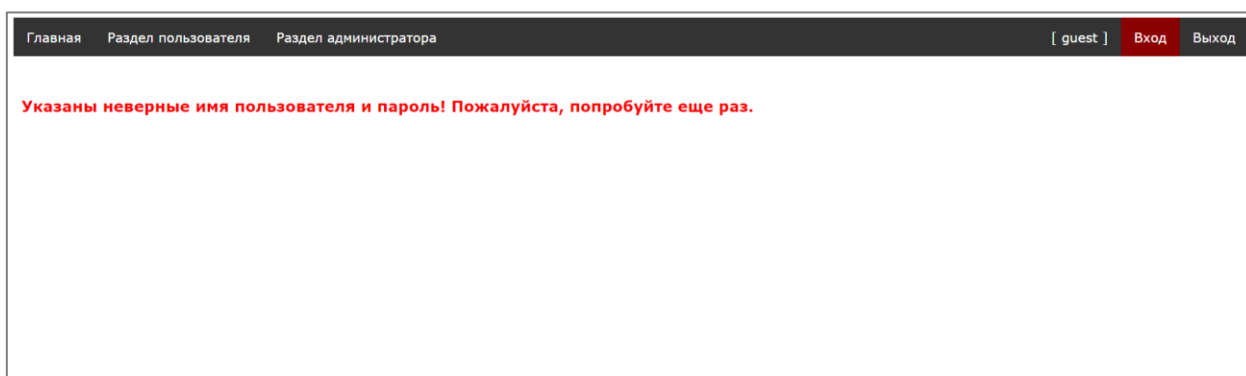


Рисунок 11 – Информация о неверно введенном имени пользователя и пароле

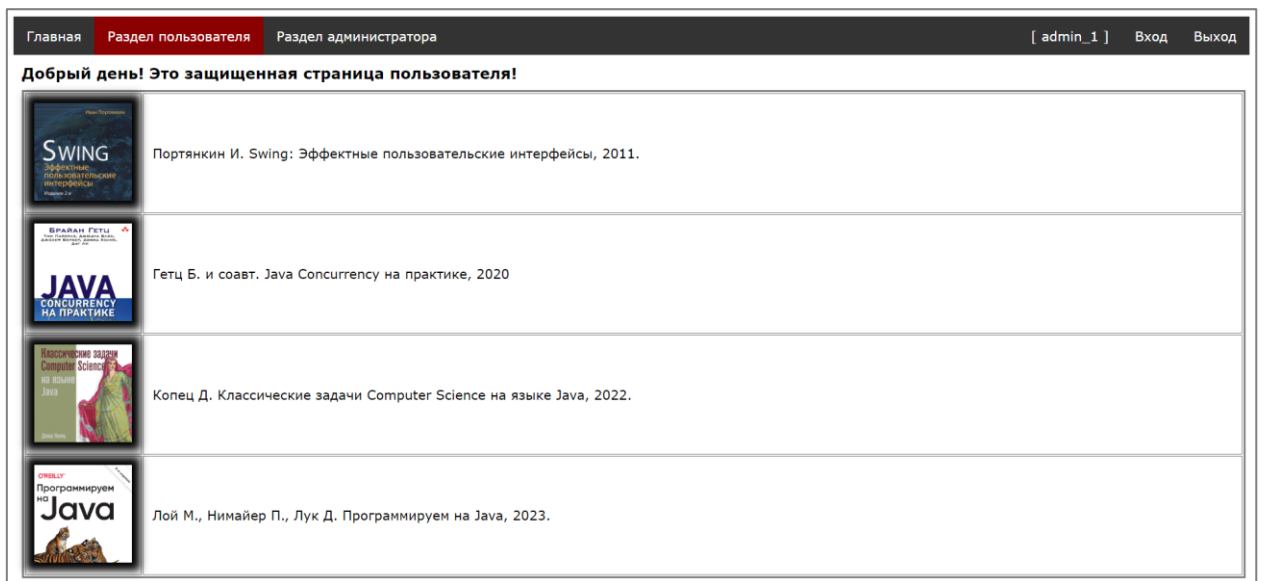


Рисунок 12 – Раздел пользователя после авторизации на сайте, зарегистрированный пользователь (admin_1)

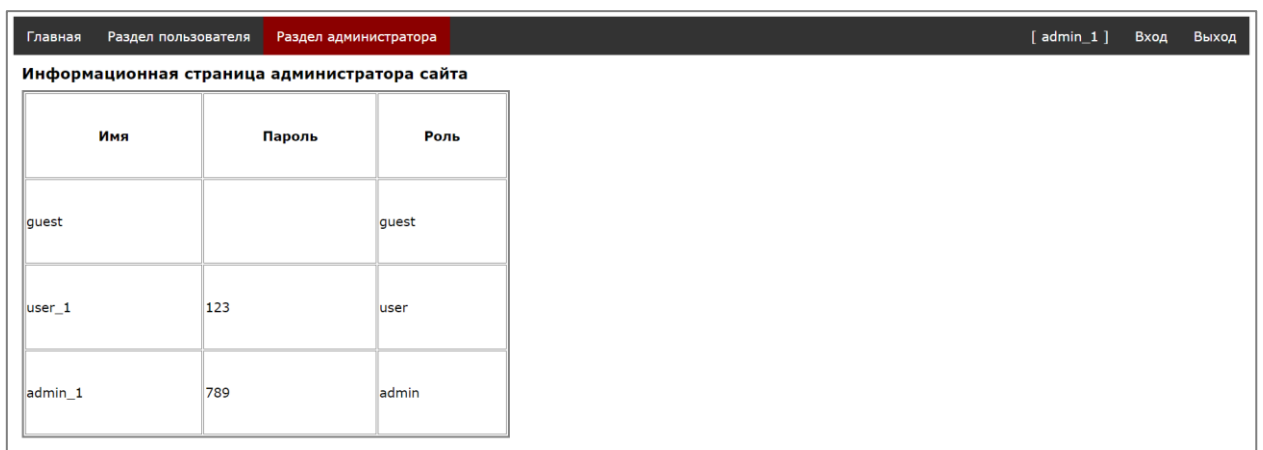


Рисунок 13 – Раздел администратора, зарегистрированный пользователь (admin_1)

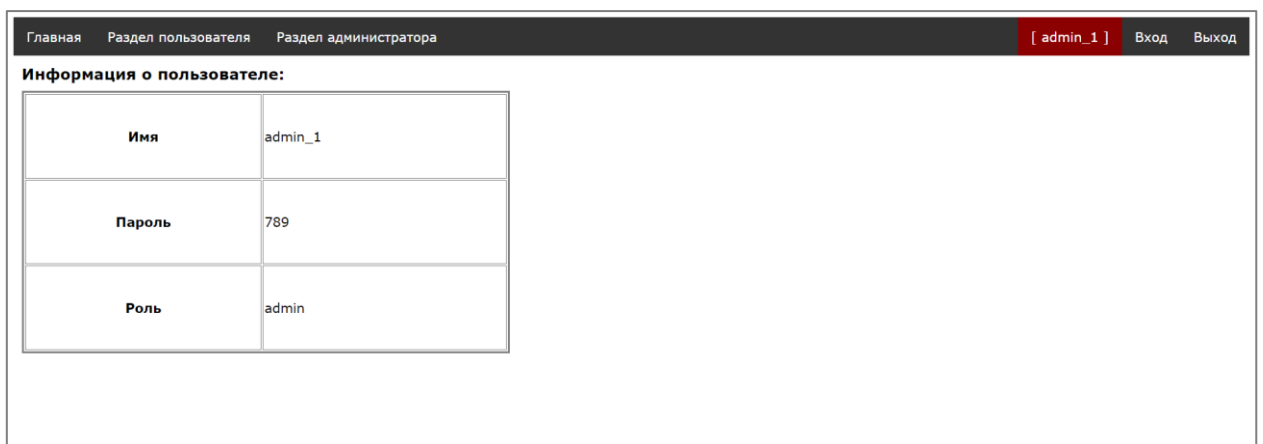


Рисунок 14 – Информация о пользователе, зарегистрированный пользователь (admin_1)