МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение
высшего образования «Самарский национальный исследовательский университет
имени академика С.П. Королева»
(Самарский университет)

Институт информатики и кибернетики
Кафедра технической кибернетики

ОТЧЕТ

по лабораторной работе №4
Дисциплина «Технологии сетевого программирования»

Выполнили студенты
группы 6301-010302D
Жданова В.И.
Русяева С.П.

САМАРА 2025

Созданы html страницы:

- login.html – страница авторизации (поля для ввода email и пароля, ссылка на страницу регистрации);
- register.html – страница регистрации (поля для ввода email, пароля, имени, номера телефона);
- home.html – домашняя страница приложения;
- header.html – шапка страницы с основными ссылками (каталог, корзина, заказы, профиль для пользователя и каталог, бренды, заказы для администратора; кнопка logout);
- products.html – каталог товаров с полем для фильтрации (с возможностью добавления, удаления, редактирования товаров для администратора);
- cart.html – корзина пользователя (доступно только пользователю);
- orders.html – страница личных заказов пользователя (доступно только пользователю);
- brands.html – список всех брендов, возможность редактирования, добавления, удаления бренда (доступно только администратору);
- ordersAdmin.html – страница заказов всех пользователей с возможностью изменения статуса заказа (доступно только администратору);
- profile.html – профиль пользователя с возможностью удалить профиль, изменить пароль, изменить номер телефона и имя (доступно только пользователю).

Пример страницы регистрации:

```
<!DOCTYPE html>

<html xmlns:th="http://www.thymeleaf.org">

<head><title>Register</title>

  <link href="https://cdn.jsdelivr.net/npm/bootswatch@5.3.0/dist/lux/bootstrap.min.css"
rel="stylesheet">

  <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.10.0/font/bootstrap-icons.css">

</head>

<body>
```

```
<div th:if="${error}" style="color:red;">
  <p th:text="${error}"></p>
</div>
<div class="container mt-5">
  <form  th:action="@{/api/auth/register}" method="post" class="mx-auto" style="max-width: 400px;">
    <fieldset class="p-4 border rounded-3 bg-light shadow-sm">
      <legend class="text-center mb-4">Регистрация</legend>
      <div class="mb-3">
        <label for="exampleInputEmail1" class="form-label">Email</label>
        <input type="email" name="email" class="form-control" id="exampleInputEmail1" aria-describedby="emailHelp" placeholder="Введите email" required>
      </div>
      <div class="mb-4">
        <label for="exampleInputPassword1" class="form-label">Пароль</label>
        <input type="password" name="password" class="form-control" id="exampleInputPassword1" placeholder="Введите пароль" autocomplete="off" required>
      </div>
      <div class="mb-4">
        <label for="exampleInputPassword1" class="form-label">Имя пользователя</label>
        <input type="text" name="username" class="form-control" id="exampleUserName" placeholder="Введите пароль" autocomplete="off" required>
      </div>
      <div class="mb-4">
        <label for="examplePhone" class="form-label">Телефон</label>
        <input type="text" name="phone" class="form-control" id="examplePhone"
          placeholder="+79999999999" autocomplete="off" required
          pattern="^\+7\d{10}$"
          title="Телефон должен быть в формате +79999999999">
      </div>
```

```
        <div class="d-grid gap-2 mb-3">
            <button type="submit" class="btn btn-primary py-2">Зарегистрироваться</button>
        </div>
    </fieldset>
  </form>
</div>
</body>
</html>
```

Взаимодействие с серверной частью:

- обработка форм через @Controller;
- динамическое обновление данных с помощью AJAX.

Пример контроллера (AuthController):

```
package com.example.WebApp.controller;

import com.example.WebApp.config.JwtProvider;

import com.example.WebApp.dto.AuthDto;

import com.example.WebApp.dto.JwtResponseDto;

import com.example.WebApp.dto.RefreshTokenDto;

import com.example.WebApp.dto.UsersDto;

import com.example.WebApp.exeption.ObjectNotFoundException;

import com.example.WebApp.model.RefreshToken;

import com.example.WebApp.service.AuthService;

import com.example.WebApp.service.BlackListService;

import com.example.WebApp.service.RefreshTokenService;

import jakarta.servlet.http.Cookie;

import jakarta.servlet.http.HttpServletRequest;

import jakarta.servlet.http.HttpServletResponse;

import lombok.AccessLevel;

import lombok.RequiredArgsConstructor;
```

```
import lombok.experimental.FieldDefaults;
import lombok.extern.slf4j.Slf4j;
import org.springframework.http.ResponseEntity;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.*;

@Controller
@RequestMapping("/api/auth")
@FieldDefaults(level = AccessLevel.PRIVATE, makeFinal = true)
@RequiredArgsConstructor
@Slf4j
public class AuthController {

  AuthService authService;
  RefreshTokenService refreshTokenService;
  BlackListService blackListService;
  JwtProvider jwtProvider;

  @PostMapping("/register")
  public String register(
        @RequestParam String email,
        @RequestParam String password,
        @RequestParam String username,
        @RequestParam String phone,
        HttpServletResponse response
  ) {
     UsersDto userDto = new UsersDto(username, email, password, phone);
     authService.save(userDto);
```

```
    JwtResponseDto jwt = authService.authenticate(new AuthDto(email, password));
    Cookie cookie = new Cookie("access_token", jwt.getAccessToken());
    cookie.setPath("/");
    cookie.setHttpOnly(true);
    response.addCookie(cookie);
    Cookie refreshCookie = new Cookie("refresh_token", jwt.getRefreshToken());
    refreshCookie.setHttpOnly(true);
    refreshCookie.setPath("/");
    response.addCookie(refreshCookie);
    return "redirect:/";
}
@PostMapping("/login")
public String loginFromForm(
        @RequestParam String email,
        @RequestParam String password,
        HttpServletResponse response
) {
    JwtResponseDto jwt = authService.authenticate(new AuthDto(email, password));
    Cookie cookie = new Cookie("access_token", jwt.getAccessToken());
    cookie.setPath("/");
    cookie.setHttpOnly(true);
    response.addCookie(cookie);

    Cookie refreshCookie = new Cookie("refresh_token", jwt.getRefreshToken());
    refreshCookie.setHttpOnly(true);
    refreshCookie.setPath("/");
    response.addCookie(refreshCookie);
```

```
        return "redirect:/";
    }
    @PostMapping("/refresh")
    public ResponseEntity<JwtResponseDto> refreshToken(@RequestBody RefreshTokenDto
refreshTokenDto) {
        return ResponseEntity.ok(refreshTokenService.refresh(refreshTokenDto));
    }
    @PostMapping("/logout")
    public String logout(
            HttpServletRequest request,
            HttpServletResponse response,
            @CookieValue(name = "access_token", required = false) String accessToken,
            @CookieValue(name = "refresh_token", required = false) String refreshToken
    ) {

        if (refreshToken != null) {
            refreshTokenService.findByToken(refreshToken)
                    .ifPresent(refreshTokenService::delete);
        }

        if (accessToken != null) {
            blackListService.addToBlacklistToken(accessToken,
jwtProvider.extractExpiration(accessToken));
        }

        Cookie cookie = new Cookie("access_token", null);
        cookie.setPath("/");
        cookie.setHttpOnly(true);
        cookie.setMaxAge(0);
```

```
        response.addCookie(cookie);

        Cookie refreshCookie = new Cookie("refresh_token", null);
        refreshCookie.setPath("/");
        refreshCookie.setHttpOnly(true);
        refreshCookie.setMaxAge(0);
        response.addCookie(refreshCookie);

        return "login";
    }
}
```

Пример AJAX-запроса (удаление профиля):

```
function deleteUser() {
    if (confirm('Вы уверены, что хотите удалить профиль?')) {
        fetch('/api/shop/users/me', {
            method: 'DELETE',
            credentials: 'same-origin'
        })
        .then(response => {
            if (response.ok || response.status === 204) {
                window.location.href = '/api/auth/login';
            } else {
                alert('Ошибка при удалении аккаунта');
            }
        })
        .catch(error => {
            console.error('Ошибка:', error);
            alert('Сетевая ошибка');
```

```
        });
    }
}
```

Макеты страниц создавались с помощью шаблонизатора Thymeleaf. Для стилизации использовался Bootstrap.