



**Федеральное государственное бюджетное образовательное учреждение высшего образования  
«МИРЭА – Российский технологический университет»**

**РТУ МИРЭА**

**Институт информационных технологий**

**Кафедра инструментального и прикладного программного обеспечения**

**Дисциплина «Разработка клиент-серверных приложений»**

## **КУРСОВАЯ РАБОТА**

### **Фуллстек-приложение для доставки продуктов**

Студент: Фомичев Р.А.

Группа: ИКБО-20-21

Руководитель: старший преподаватель Коваленко М.А.

Москва 2023

# 1 Цель

Проектирование и разработка клиент-серверного фуллстек-приложения для доставки продуктов

## 2 Задачи

1. Провести анализ предметной области и сформировать основные требования к приложению
2. Обосновать выбор средств ведения разработки
3. Разработать приложение с использованием выбранной технологии и инструментария
4. Провести тестирование приложения
5. Развернуть клиент-серверное приложение в облаке
5. Оформить пояснительную записку по курсовой работе
6. Провести анализ текста на антиплагиат
7. Создать презентацию по выполненной курсовой работе

### 3 Технологии разработки

Java

Spring

HTML5

CSS3

PostgreSQL

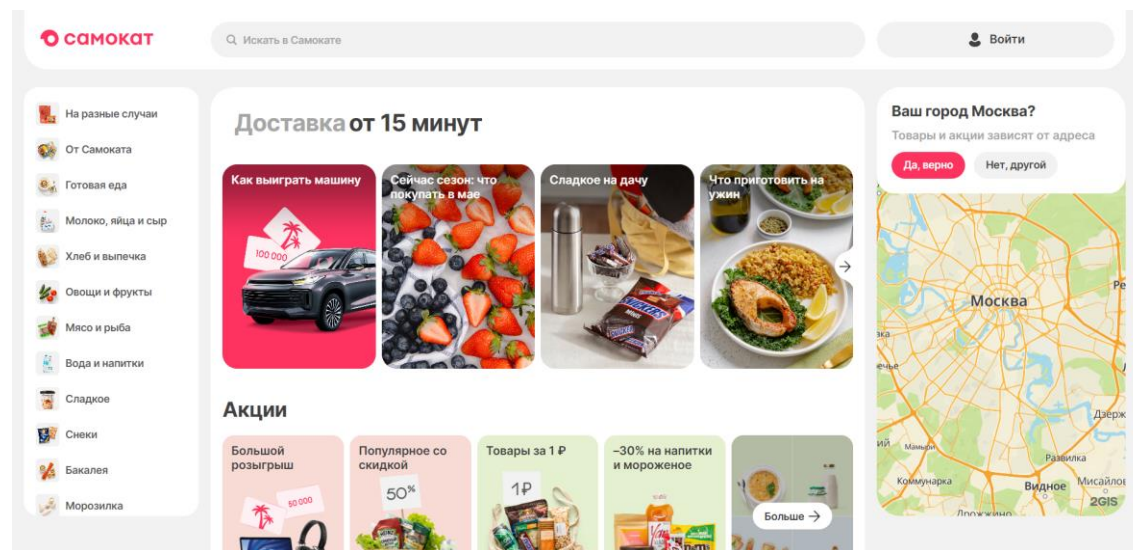
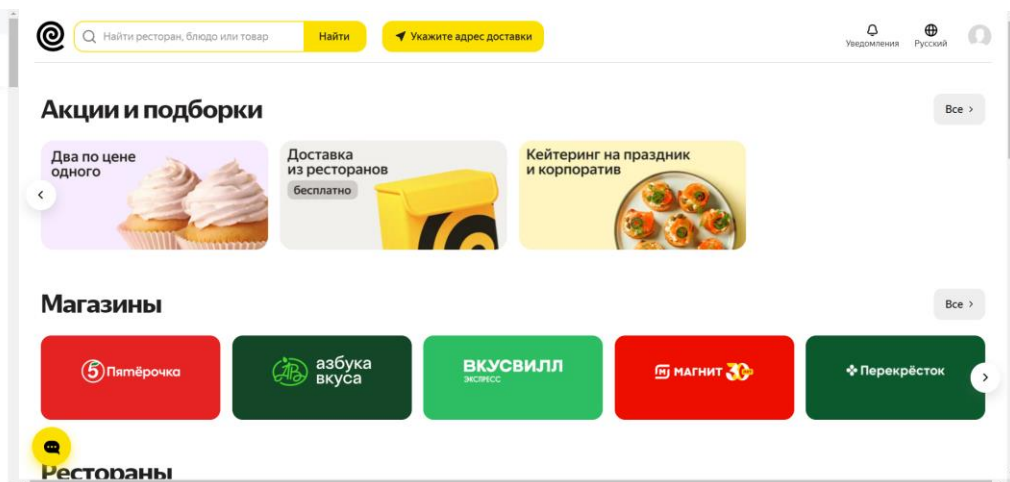
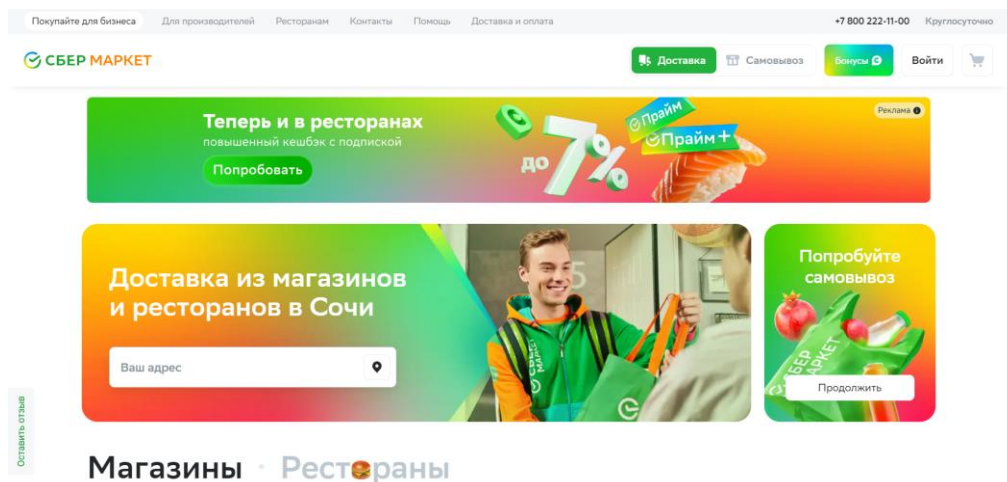
HTTP

REST

Maven

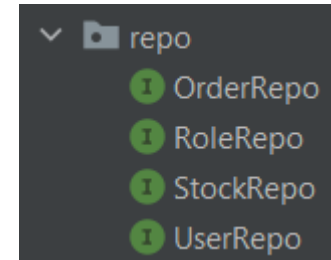
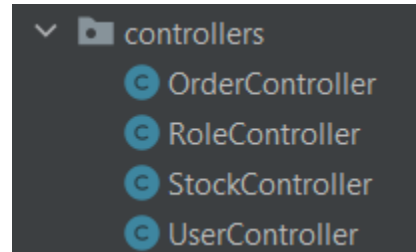
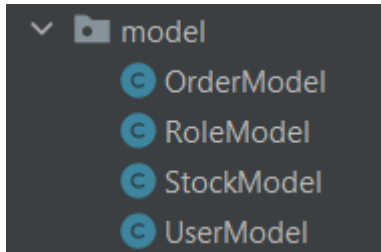
Git

# Анализ предметной области



# 4 Серверная часть приложения приложения

Модели, контроллеры, репозитории



```
@Bean
public SecurityFilterChain securityFilterChain(HttpSecurity http) throws Exception {
    return http.csrf(AbstractHttpConfigurer::disable)
        .cors(AbstractHttpConfigurer::disable)
        .authorizeHttpRequests(auth ->
            auth.requestMatchers(...patterns: "api/**", "login").permitAll())
        .formLogin(AbstractAuthenticationFilterConfigurer::permitAll).httpBasic(Customizer.withDefaults())
        .build();
}

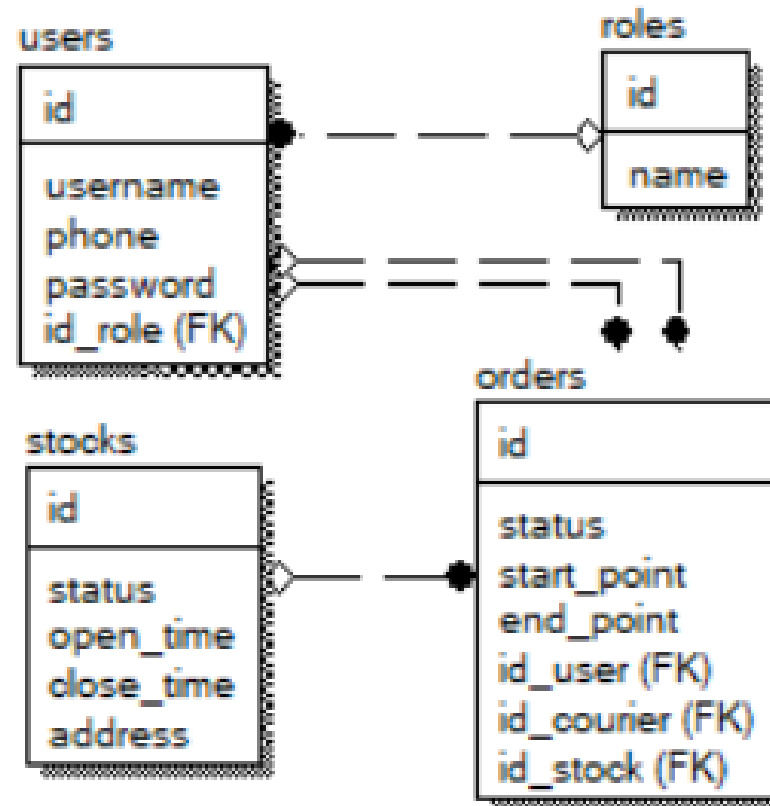
@Bean
public AuthenticationProvider authenticationProvider() {
    DaoAuthenticationProvider provider = new DaoAuthenticationProvider();
    provider.setUserDetailsService(userDetailsService());
    provider.setPasswordEncoder(passwordEncoder());
    return provider;
}
```

```
@GetMapping("/{user}/{id}")
@PreAuthorize("hasAuthority('3') || hasAuthority('1')")
public List<OrderModel> gerUserOrders(@PathVariable String id) {
    return orderRepo.findByIdUser(Integer.valueOf(id));
}

@GetMapping("/{courier}/{id}")
@PreAuthorize("hasAuthority('3') || hasAuthority('2')")
public List<OrderModel> gerCourierOrders(@PathVariable String id) {
    List<OrderModel> orders = orderRepo.findByIdCourier(Integer.valueOf(id));
    List<OrderModel> createdOrders = orderRepo.findByStatus("Created");
    orders.addAll(createdOrders);
    return orders;
}
```

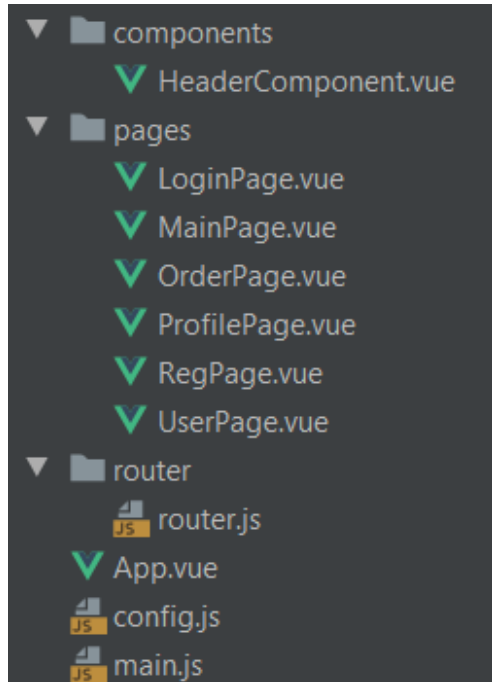
# 5 База данных

## Структура базы данных



# 6 Клиентская часть приложения

Структура, маршруты



```
import {createRouter, createWebHistory} from "vue-router";
import MainPage from "../pages/MainPage.vue";
import LoginPage from "../pages/LoginPage.vue";
import OrderPage from "../pages/OrderPage.vue";
import ProfilePage from "../pages/ProfilePage.vue";
import RegPage from "../pages/RegPage.vue";
import UserPage from "../pages/UserPage.vue";

const routes = [
  {path: '/'...},
  {path: '/login'...},
  {path: '/register'...},
  {path: '/order'...},
  {path: '/users'...},
  {path: '/profile'...}
]
```

# 7 Тестирование

## Регистрация

newUser

....

88005553535

Зарегистрироваться



# 8 Тестирование

Авторизация

Войти

Зарегистрироваться

# 9 Тестирование

Страница пользователя

**ProDel**

**Пользователи** **Заказы** **Профиль** **Logout**

**admin**

Логин: admin

Телефон: 0987654321

Роль доступа: Администратор(3)

**История заказов**

**#5** *Создан*

# 10 Тестирование

Доступ к странице администрирования

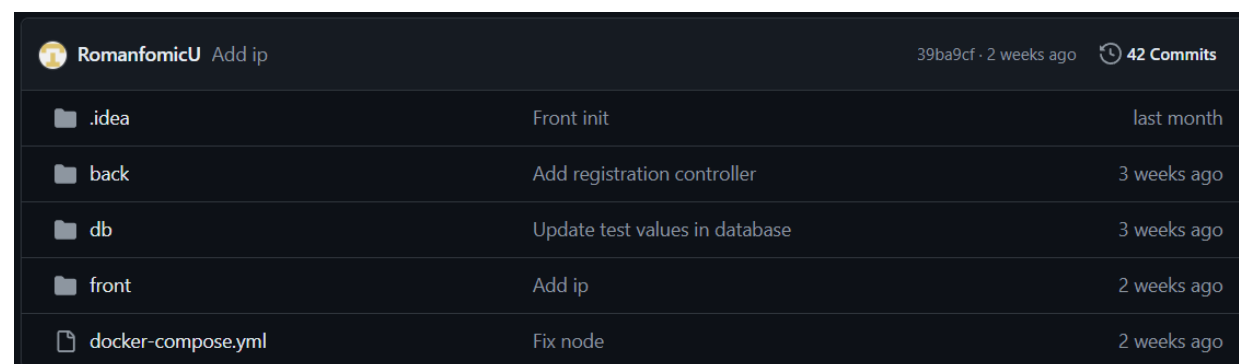
ProDel				Пользователи		Заказы	Профиль	Logout
id	username	phone	role					
#3	admin	0987654321	3	+	-			
#1	user	1234567890	1	+	-			
#4	NewUser	88005553535	1	+	-			
#2	courier	6543217890	2	+	-			

# 11 Результаты

- Разработана база данных
- Разработана серверная часть приложения
- Разработана клиентская часть приложения
- Написано 1500 строк кода серверной части и 800 строк кода клиентской части приложения
- Обеспечена отправка запросов клиентской части на сервер, их обработка и отправка ответа с серверной части.
- Обеспечена связь сервера с базой данных
- Приложение было размещено в облаке

URL репозитория на GitHub:

<https://github.com/RomanfomicU/ProductsDelivery>



RomanfomicU Add ip		39ba9cf · 2 weeks ago	🕒 42 Commits
📁 .idea	Front init	last month	
📁 back	Add registration controller	3 weeks ago	
📁 db	Update test values in database	3 weeks ago	
📁 front	Add ip	2 weeks ago	
📄 docker-compose.yml	Fix node	2 weeks ago	

**СПАСИБО ЗА ВНИМАНИЕ!**