|  |
| --- |
|  |
| МИНОБРНАУКИ РОССИИ |
| Федеральное государственное бюджетное образовательное учреждение высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** |

**Институт информационных технологий (ИТ)**

**Кафедра инструментального и прикладного программного обеспечения (ИиППО)**

**КУРСОВАЯ РАБОТА**

по дисциплине: Шаблоны программных платформ языка Джава

по профилю: Разработка программных продуктов и проектирование информационных систем

направления профессиональной подготовки: 09.03.04 «Программная инженерия»

Тема: Приложение «Ресторан»

Студент: Зубков Михаил Витальевич Группа: ИКБО-30-20

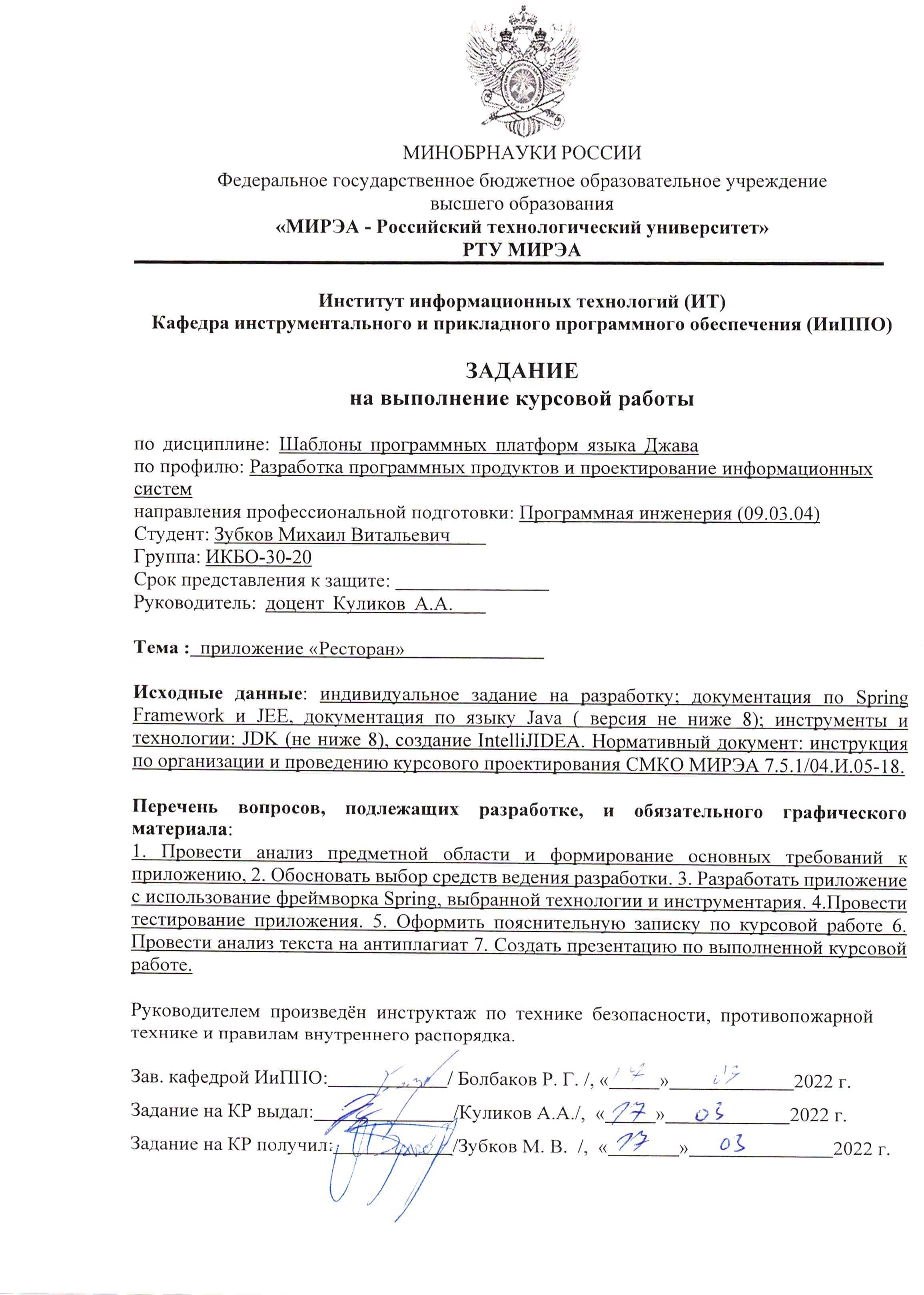
Работа представлена к защите (дата) / /

Руководитель: доцент Куликов А. А.

Работа допущена к защите (дата) /Куликов А. А./ Оценка по итогам защиты:

/

1



УДК 004.04

Зубков М. В. Веб-приложение на тему «Ресторан» / Курсовая работа по дисциплине «Шаблоны программных платформ языка Джава» профиля

«Разработка программных продуктов и проектирование информационных систем» направления профессиональной подготовки бакалавриата 09.03.04.

«Программная инженерия» (4-ий семестр) / руководитель старший преподаватель Куликов А. А. / кафедра ИиППО Института ИТ РТУ МИРЭА.

Целью курсовой работы является проведение анализа предметной области, формирование требований к разрабатываемому веб-приложению, разработка и тестирование веб-приложения, а также оформление документов сопутствующей документации.

М. МИРЭА. Ин-т ИТ. Каф. ИиППО. 2022 г. Зубков М. В.

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 4](#_Toc104659124)

[1. СБОР И АНАЛИЗ ТРЕБОВАНИЙ К ПРОГРАММНОМУ ПРОДУКТУ 5](#_Toc104659125)

[1.1 Описание предметной области 5](#_Toc104659126)

[ВЫВОДЫ К РАЗДЕЛУ 1 7](#_Toc104659127)

[2. РАЗРАБОТКА ВЕБ-ПРИЛОЖЕНИЯ 8](#_Toc104659128)

[2.1 Выбор технологий 8](#_Toc104659129)

[2.2 Структура веб-приложения 9](#_Toc104659130)

[2.3 Процесс создания веб-приложения 11](#_Toc104659131)

[2.3.1 Создание проекта и добавление зависимостей 11](#_Toc104659132)

[2.3.2 Создание сущностей и заполнение базы данных. Entity. 13](#_Toc104659133)

[2.3.3 Создание репозиториев. Repository. 17](#_Toc104659134)

[2.3.4 Создание контроллеров. Controllers. 18](#_Toc104659135)

[2.3.5 Создание клиентской части. Bootstrap. 19](#_Toc104659136)

[2.3.6 Создание сервисов. Services. 23](#_Toc104659137)

[2.4 Диаграмма классов и таблиц серверной части 23](#_Toc104659138)

[ВЫВОДЫ К РАЗДЕЛУ 2 24](#_Toc104659139)

[3. ТЕСТИРОВАНИЕ ВЕБ-ПРИЛОЖЕНИЯ 25](#_Toc104659140)

[3.1 Технология проведения тестирования 25](#_Toc104659141)

[3.2 Проблемы с тестированием регистрации и аутентификации 27](#_Toc104659142)

[ВЫВОДЫ К 3 РАЗДЕЛУ 27](#_Toc104659143)

[ЗАКЛЮЧЕНИЕ 28](#_Toc104659144)

[СПИСОК ИСПОЛЬЗОВАНЫХ ИСТОЧНИКОВ 29](#_Toc104659145)

[ПРИЛОЖЕНИЕ 30](#_Toc104659146)

**СПИСОК ИСПОЛЬЗОВАННЫХ СОКРАЩЕНИЙ**

ЯП – Язык программирования

СУБД – Система управления базами данных

# ВВЕДЕНИЕ

В современном мире подавляющее количество человек используют интернет для получения информации о местах проведения досуга. В связи с этим актуальность веб-приложений для поиска информации о них невероятна высока. В ходе данной курсовой работы, будет разработано веб-приложение на тему «Ресторан». Рестораны пользуются большой популярностью у людей с различными целями, будь то работник, забегающий перед работой за стаканчиком бодрящего эспрессо, или же дружная компания, заходящая перекусить вкусной еды.

Целью курсовой работы является создание современного веб-приложения с возможностью просмотра ассортимента и цен на продукцию ресторана.

Для выполнения данной цели будут выполнены следующие шаги:

1. Разработка веб-приложения с помощью Spring Framework и СУБД
2. Создание клиентской части веб-приложения
3. Проведение тестирования

Для получения информации о продуктах будут использованы сайты со схожей тематикой.

# СБОР И АНАЛИЗ ТРЕБОВАНИЙ К ПРОГРАММНОМУ ПРОДУКТУ

## Описание предметной области

Предметная область данной курсовой работы – исследование веб- ресурсов, которые представляют деятельность ресторанов. В данной части будут рассмотрены примеры сайтов ресторанов.

Примером сайта с хорошей реализацией является «Шоколадница» (Рисунок 1.1.1). На сайте представлен ассортимент и имеется личный кабинет, где есть возможность посмотреть баллы по программе лояльности.

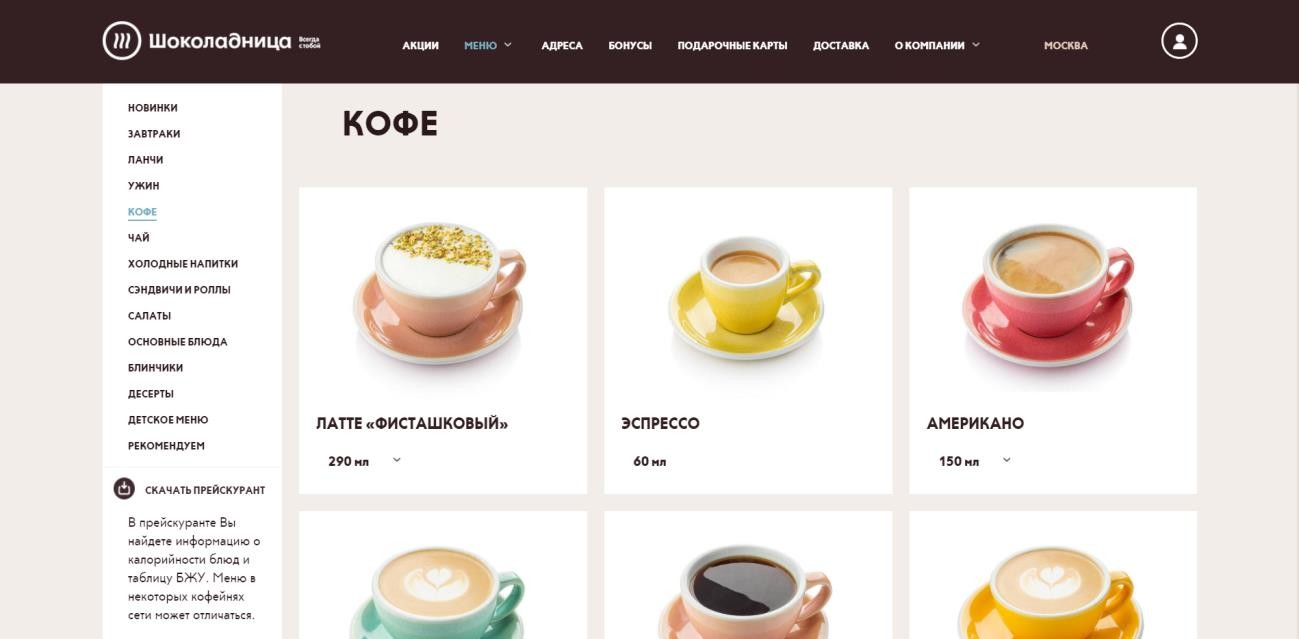


Рисунок 1.1.1 – Раздел «Меню» «Шоколадницы»

Не таким хорошим примером является сайт сети ресторанов «One Price», так как на нем нет возможности перейти в личный кабинет и проверить баллы по программе лояльности, а также нет визуализации меню (Рисунок 1.1.2)

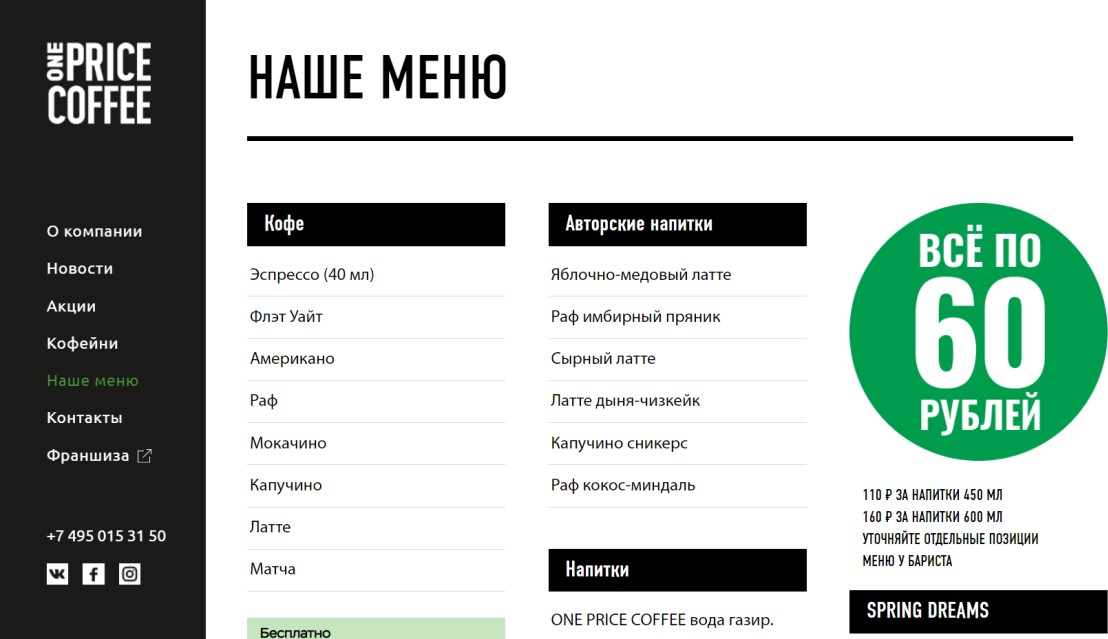


Рисунок 1.1.2 – Сайт «One Price»

Также, хорошим примером является сайт ресторанов быстрого питания «Burger King», но с одним недостатком – это невозможность посмотреть список заказанного (Рисунок 1.1.3)

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 1.1.3 – Раздел «Мои заказы» «Burger King»

## ВЫВОДЫ К РАЗДЕЛУ 1

В ходе анализа предметной области, были выявлены плюсы и минусы сайтов разных ресторанов, в связи с этим были сформированы требования к будущему веб-приложению «Ресторан»:

1. Наличие меню;
2. Визуализация ассортимента меню;
3. Наличие цен в меню;
4. Наличие личного кабинета пользователя;
5. Наличие возможности просмотреть список заказанного;

Следовательно, для реализации нам понадобится

1. Создать базу данных пользователей;
2. Создать базу данных для меню;
3. Применение технологии Hibernate для осуществления авторизации пользователей;

# 2. РАЗРАБОТКА ВЕБ-ПРИЛОЖЕНИЯ

## Выбор технологий

При разработке необходимо осуществить не только удобство пользователя, но и разработчика, в связи с этим при разработке приложения будут использованы следующие технологии, позволяющие сделать разработку легче, а использование будущего веб-приложения приятнее:

1. Intellij IDEA – интегрированная среда разработки программного обеспечения для многих языков программирования, в частности Java, JavaScript, Python, разработанная компанией JetBrains. Она способна редактировать html- файлы, а также использовать сторонние фреймворки, в нашем случае Spring;
2. Java 14 – строго типизированный объектно-ориентированный ЯП, позволяющий написать в данной курсовой работе серверную часть;
3. Spring ORM – модуль Spring, включающий в себя Spring Data JPA и Hibernate, которые будут нужны для реализации базы данных;
4. PostgreSQL – свободная объектно-реляционная система управления базами данных. Существует в реализациях для множества UNIX-подобных платформ, включая AIX, различные BSD-системы, HP-UX, IRIX, Linux, macOS, Solaris/OpenSolaris, Tru64, QNX, а также для Microsoft Windows;
5. Maven – фреймворк для автоматизации сборки проектов на основе описания их структуры в файлах на языке POM (англ. Project Object Model), являющемся подмножеством XML;
6. Github – репозиторий работ, позволяющий хранить проект удаленно в целях его мобильности, а также безопасности в случае удаления всех данных с локальной машины;
7. Bootstrap – свободный набор инструментов для создания сайтов и веб-приложений. Включает в себя HTML- и CSS-шаблоны оформления для типографики, веб-форм, кнопок, меток, блоков навигации и прочих компонентов веб-интерфейса, включая JavaScript-расширения
8. Postman – ПО, используемое в рамках тестирования веб-приложений

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 2.1.1 – Зависимость Hibernate в файле pom.xml

## Структура веб-приложения

Проект состоит из 4 основных частей:

1. Файлы серверной части, написанные с использованием Java и Spring Framework (Рисунок 2.2.1), включающие в себя контроллеры, модели, сервисы;

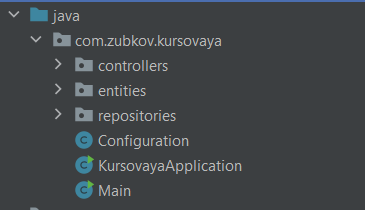


Рисунок 2.2.1 – Файлы, осуществляющие серверную логику

1. Файлы клиентской части, включающие в себя файлы Bootstrap Framework (Рисунок 2.2.2);

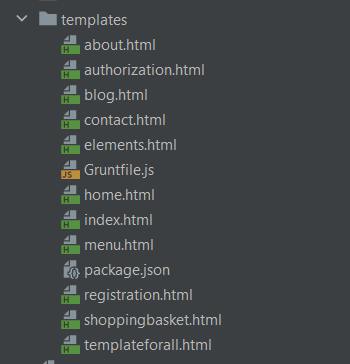


Рисунок 2.2.2 – Файлы клиентской части веб-приложения

1. Файлы Apache Maven, сборщика проекта (Рисунок 2.2.3);



Рисунок 2.2.3 – Файлы Maven, в которых происходит добавление зависимостей в проект

4. База данных PostgreSQL, в которой осуществляется хранение данных пользователей, задач и решений (Рисунок 2.2.4).

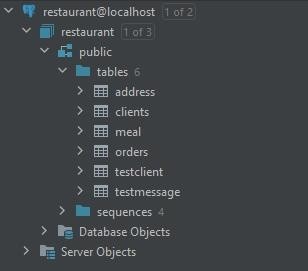


Рисунок 2.2.4 – База данных PostgreSQL

## Процесс создания веб-приложения

### Создание проекта и добавление зависимостей

При создании веб-приложения, первым делом через Spring Initializr[1] (Рисунок 2.3.1.1), сайт, позволяющий автоматизировать процесс создания проекта, был инициализирован проект. Далее были подключены зависимости проекта в maven-файл pom.xml (Рисунок 2.3.1.2), включающие в себя:

1. Spring Data JPA для дальнейшей работы с БД;
2. Thymeleaf для связи серверной части с клиентской;
3. Lombok для облегчения кода и автоматизации разработки;
4. PostgreSQL, Hibernate для реализации базы данных, где будут находится данные о пользователях, задачах и решениях.

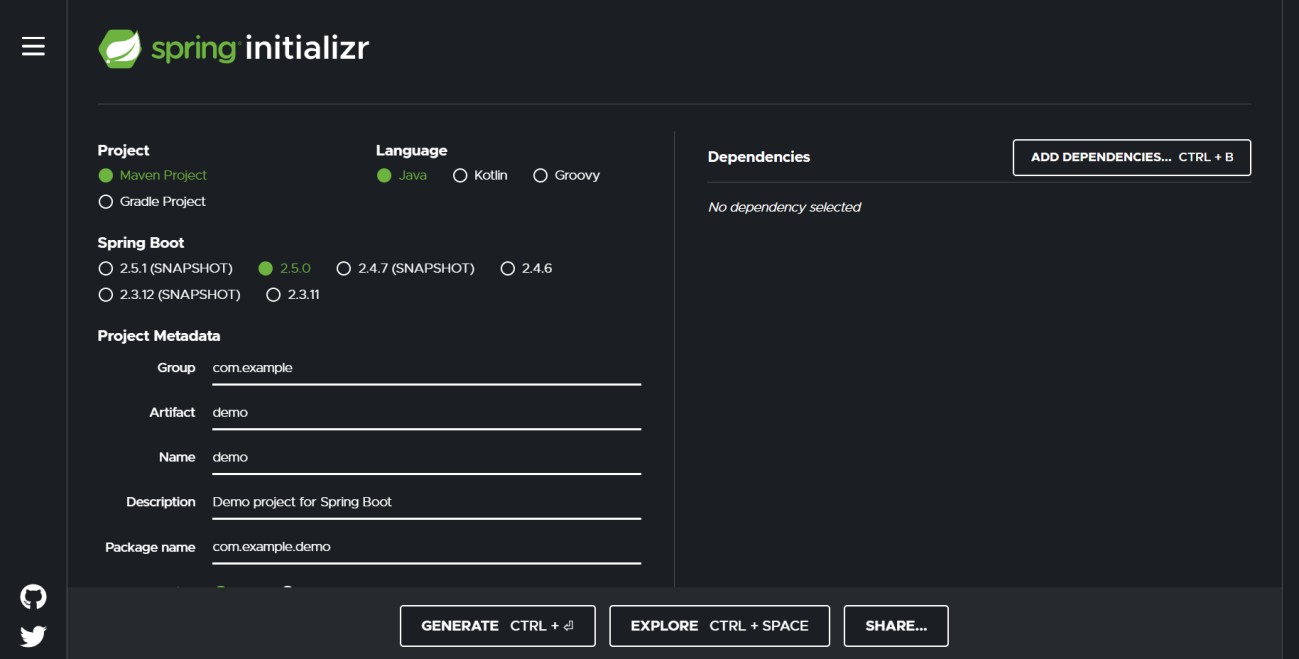


Рисунок 2.3.1.1 – Сайт Spring Initializr



Рисунок 2.3.1.2 – Часть файла pom.xml с частью представленных зависимостей

### Создание сущностей и заполнение базы данных. Entity.

После добавления зависимостей в проект можно начинать основную работу.

В начале работы осуществлялось параллельное создание и базы данных [2], и задание сущностей с отсылкой на них.

Сущность Client для задания данных о пользователей содержит 5 полей (Рисунок 2.3.2.1):

1. id – для задания id пользователя;
2. email – почта пользователя;
3. psswrd – пароль;
4. fk\_address – экземпляр класса адреса пользователя;
5. orders – поле списка заказов пользователя.

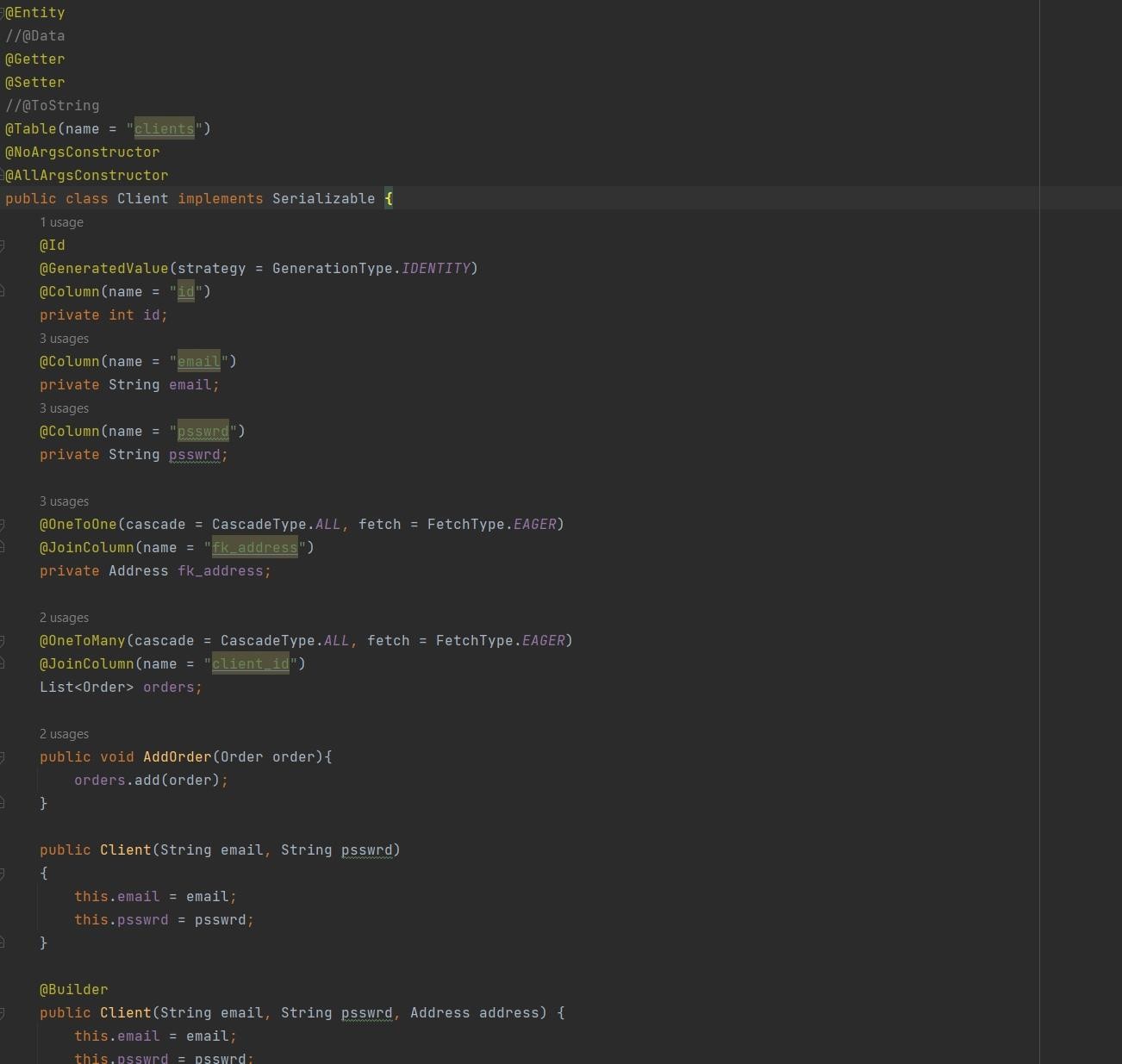


Рисунок 2.3.2.1 – Файл Client.java

Сущность Meal для еды из меню содержит 5 полей (Рисунок 2.3.2.2):

1. id – id еды;
2. name – наименование еды;
3. price – цена еды;
4. description – описание еды;
5. mealTime – время приема пищи.

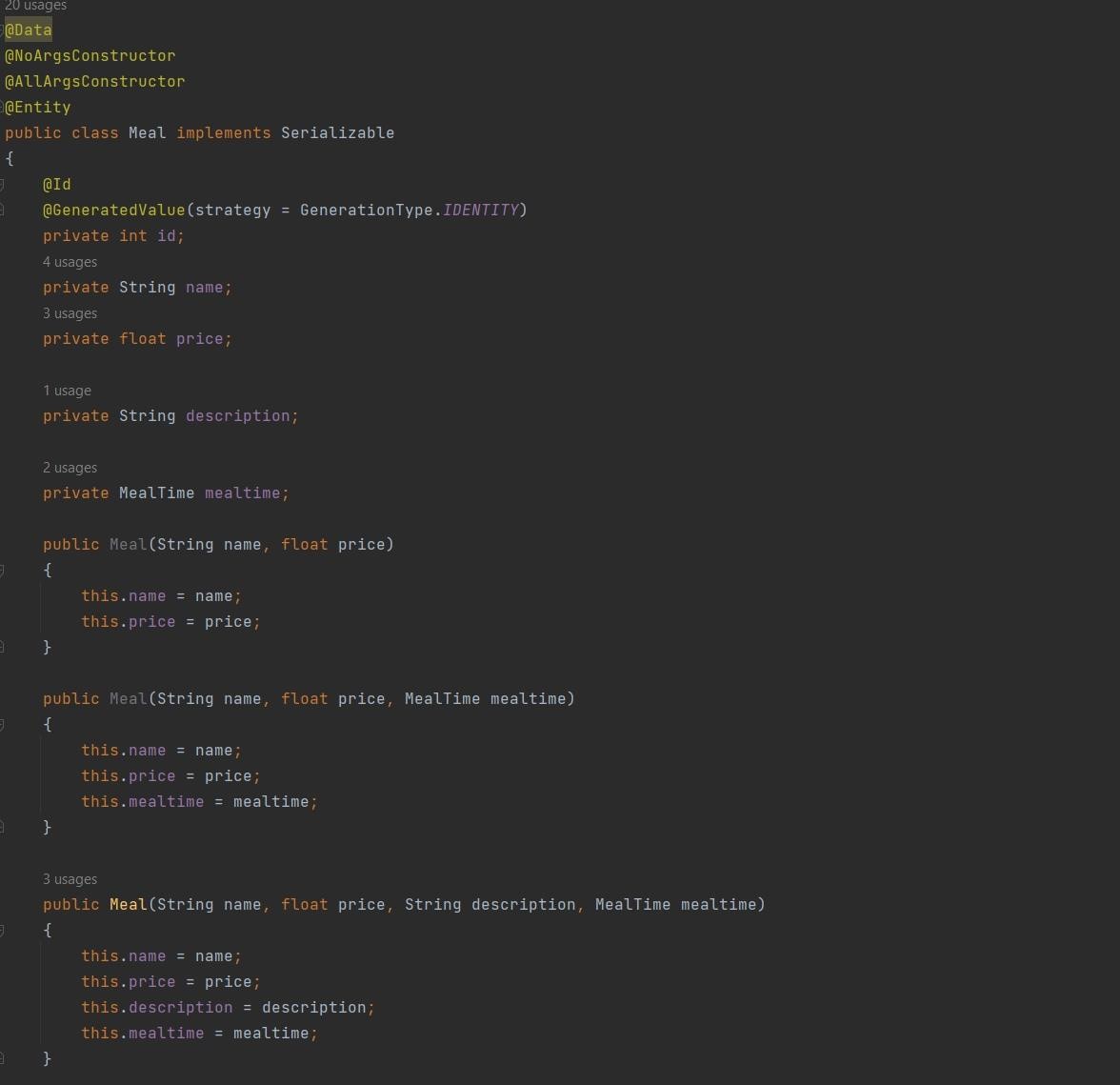


Рисунок 2.3.2.2 – Сущность «Meal»

Сущность Order, которая содержит 5 полей (Рисунок 2.3.2.3):

1. order\_id – id заказа;
2. price –общая цена заказа;
3. meals\_id – id еды, которая содержится в заказе;
4. client – клиент, который оформил заказ;

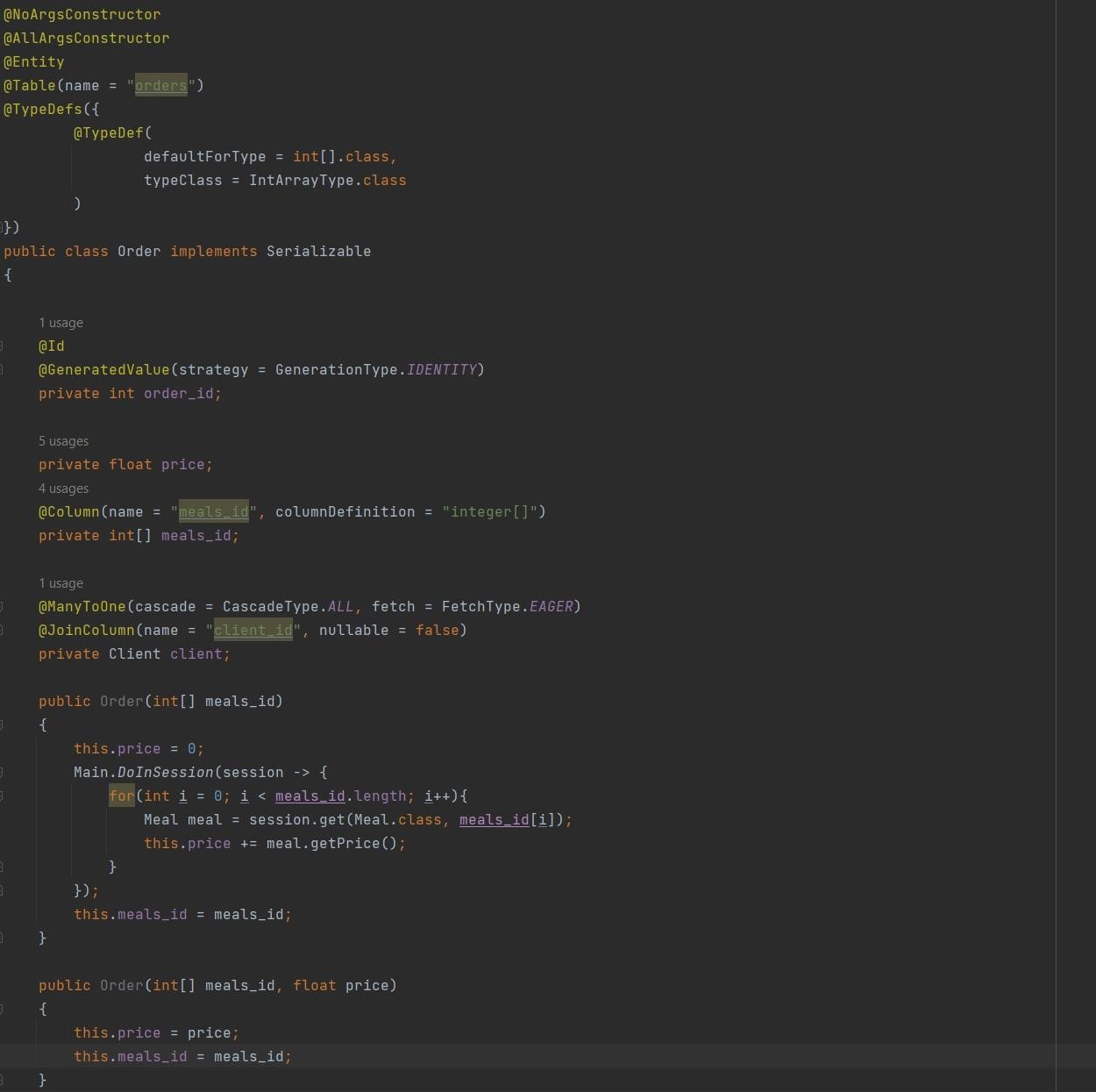


Рисунок 2.3.2.3 – Сущность «Order»

Сущность Address, которая содержит 6 полей (Рисунок 2.3.2.4):

1. id – id адреса;
2. city – город;
3. street – улица;
4. house – дом;
5. flat – номер дома;
6. client – ссылка на клиента данного адреса.

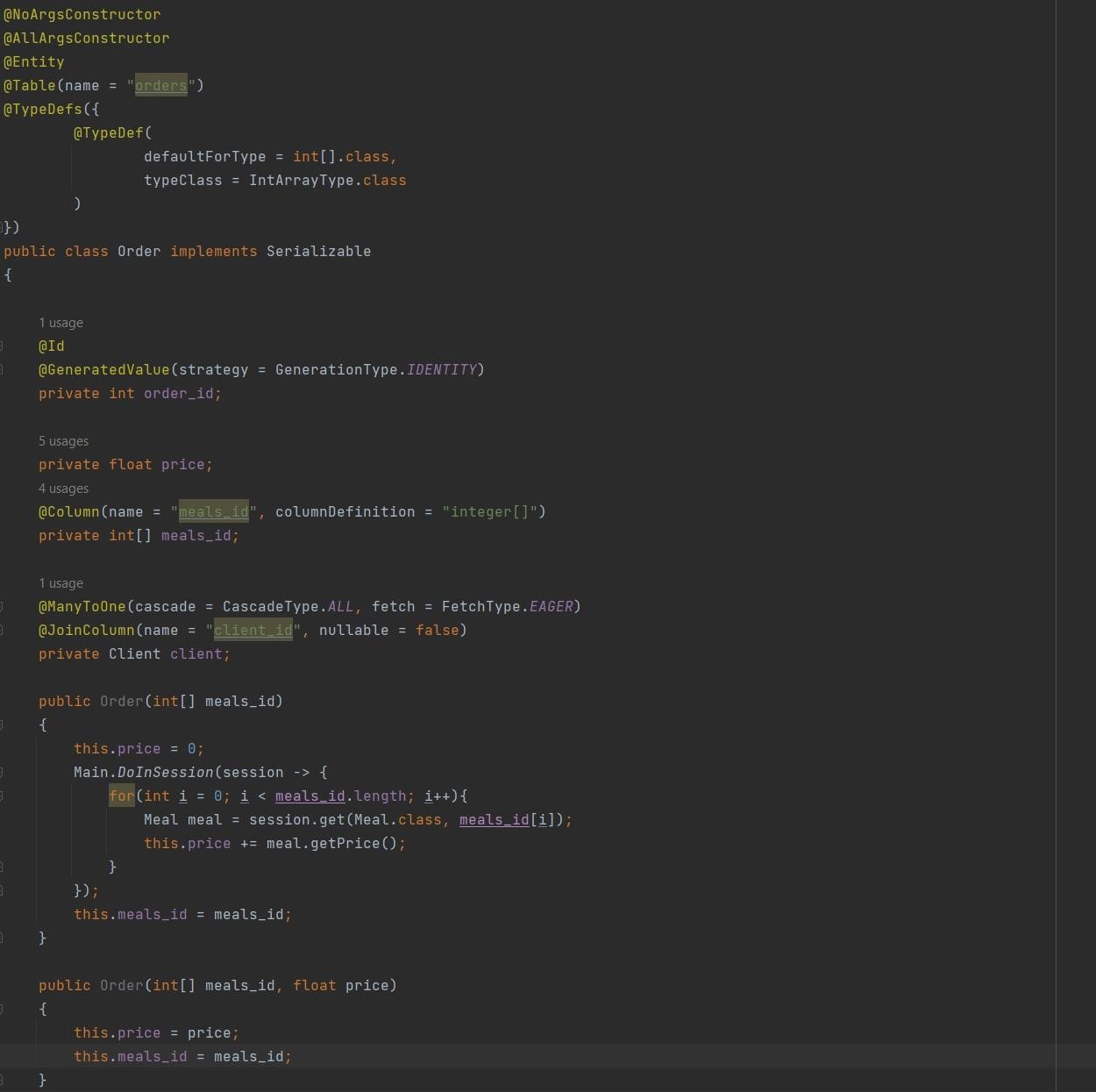


Рисунок 2.3.2.4 – Сущность «Address»

### Создание репозиториев. Repository.

После задания баз данных и создания сущностей для ее связи с прописываемой логикой серверной части, следующим шагом является создание репозиториев (Рисунки 2.3.3.1-2). Spring Data JPA [3] реализует репозитории, осуществляющие работу с базами данных, а именно: сохранение, удаление и поиск нужных данных. В предыдущем пункте через PostgreSQL уже была инициализирована база данных, связанная с помощью сущностей.

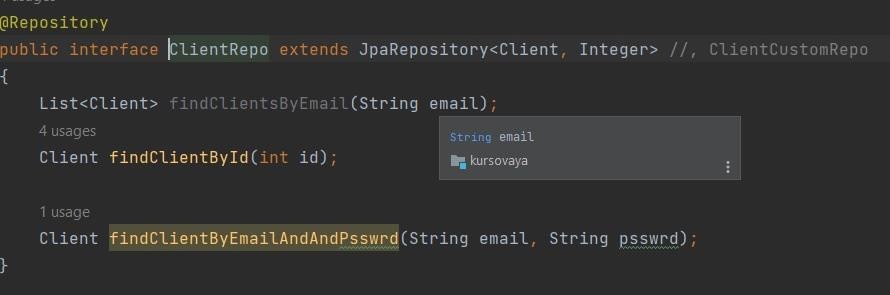


Рисунок 2.3.3.1 – Репозиторий для сущности Client



Рисунок 2.3.3.2 – Репозиторий для сущности Meal

### Создание контроллеров. Controllers.

После создания репозиториев, требуется создать контроллеры. Контроллеры осуществляют связь серверной части с клиентской, осуществление запросов post и get происходит именно в этой части кода. Так в коде можно увидеть такую аннотацию как GetMapping, которая «вызывает» страницу по заданному URL и PostMapping, возвращающую значение при определенном условии (Рисунок 2.3.4.1). Всего получилось 2 контроллера. (Рисунок 2.3.4.2)

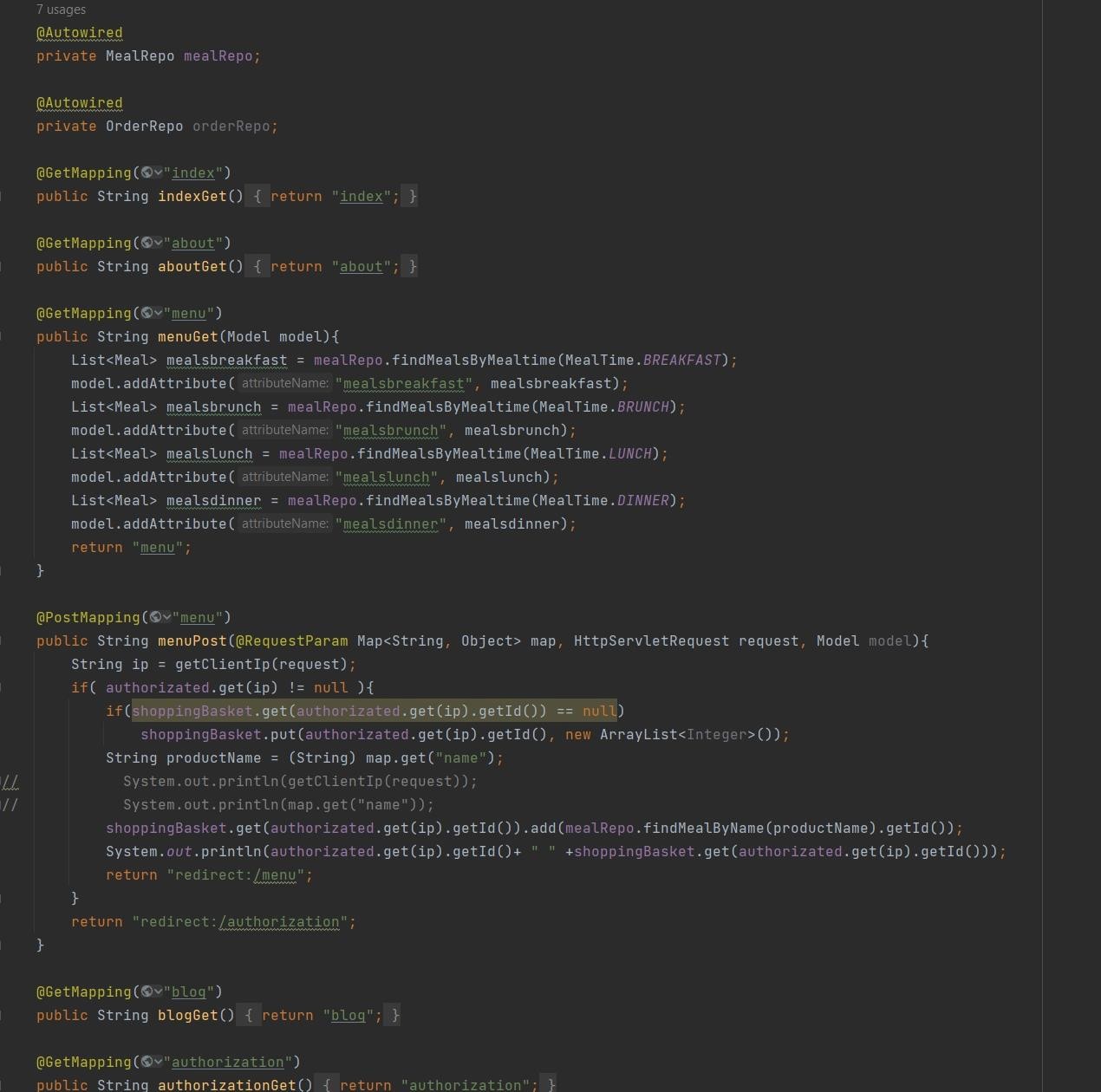


Рисунок 2.3.4.1 – Пример кода из одного из контроллеров с аннотациями

GetMapping и PostMapping



Рисунок 2.3.4.2 – Контроллеры

### Создание клиентской части. Bootstrap.

Для проверки работоспособности серверной части веб-приложения требуется провести тестирование с помощью клиентской части, в связи с этим необходимо было создать фронтенд веб-приложения. В связи с тем, что большинство пользователей смотрит сайты через телефон, клиентская часть будет выполнена в стиле mobile-first [4]. Так как целью курсовой работы является создание серверной части, реализацию клиентской части было решено облегчить, в связи с этим был использован фреймворк Bootstrap.

Клиентская часть была разработана для следующих страниц:

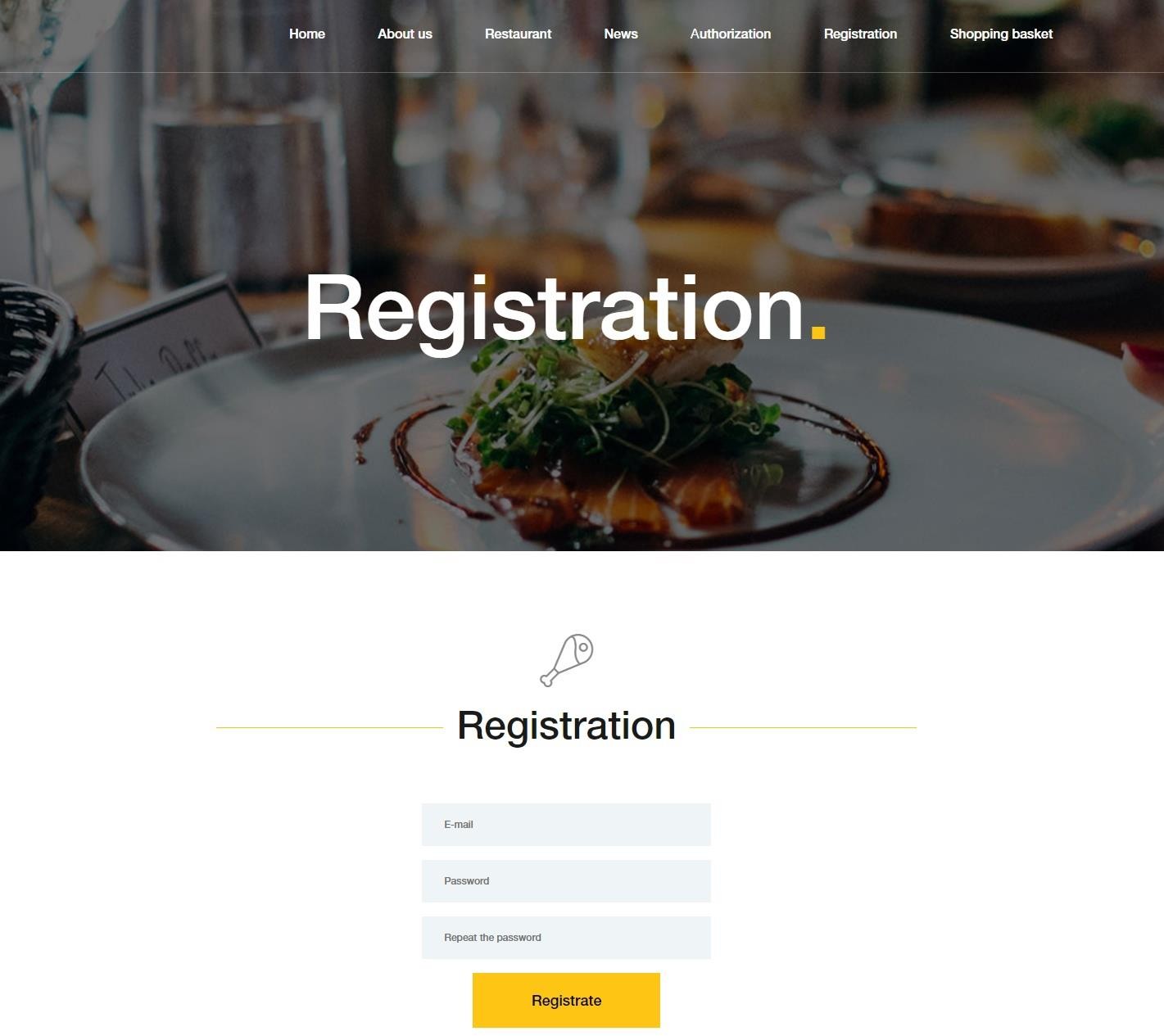
1. Окно регистрации (Рисунок 2.3.5.1)
2. Главная страница с которой происходит переход на страницу регистрации, входа и меню (Рисунок 2.3.5.2);
3. Окно входа (Рисунок 2.3.5.3);
4. Меню (Рисунок 2.3.5.4)
5. Личный кабинет (Рисунок 2.3.5.5)

Рисунок 2.3.5.1 – Окно регистрации

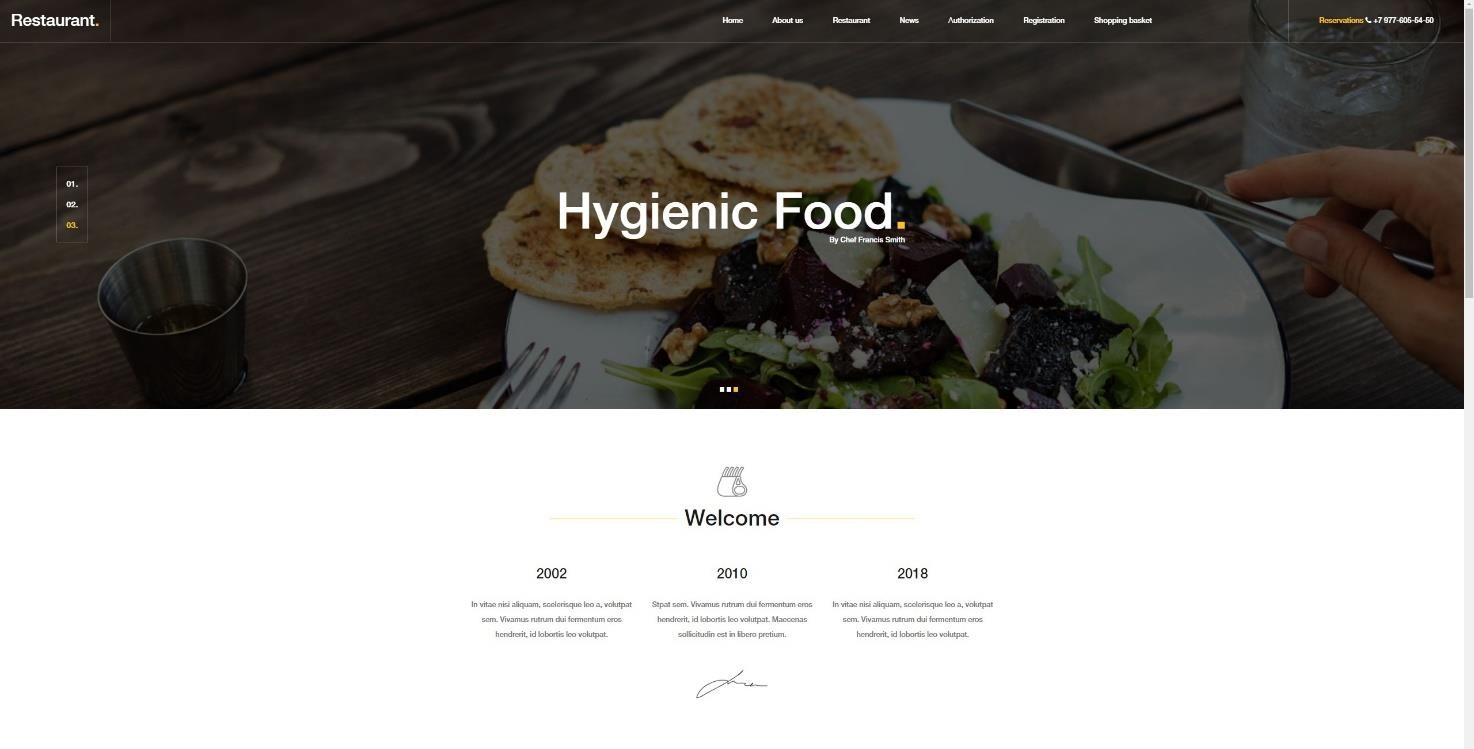


Рисунок 2.3.5.2 – Главная страница

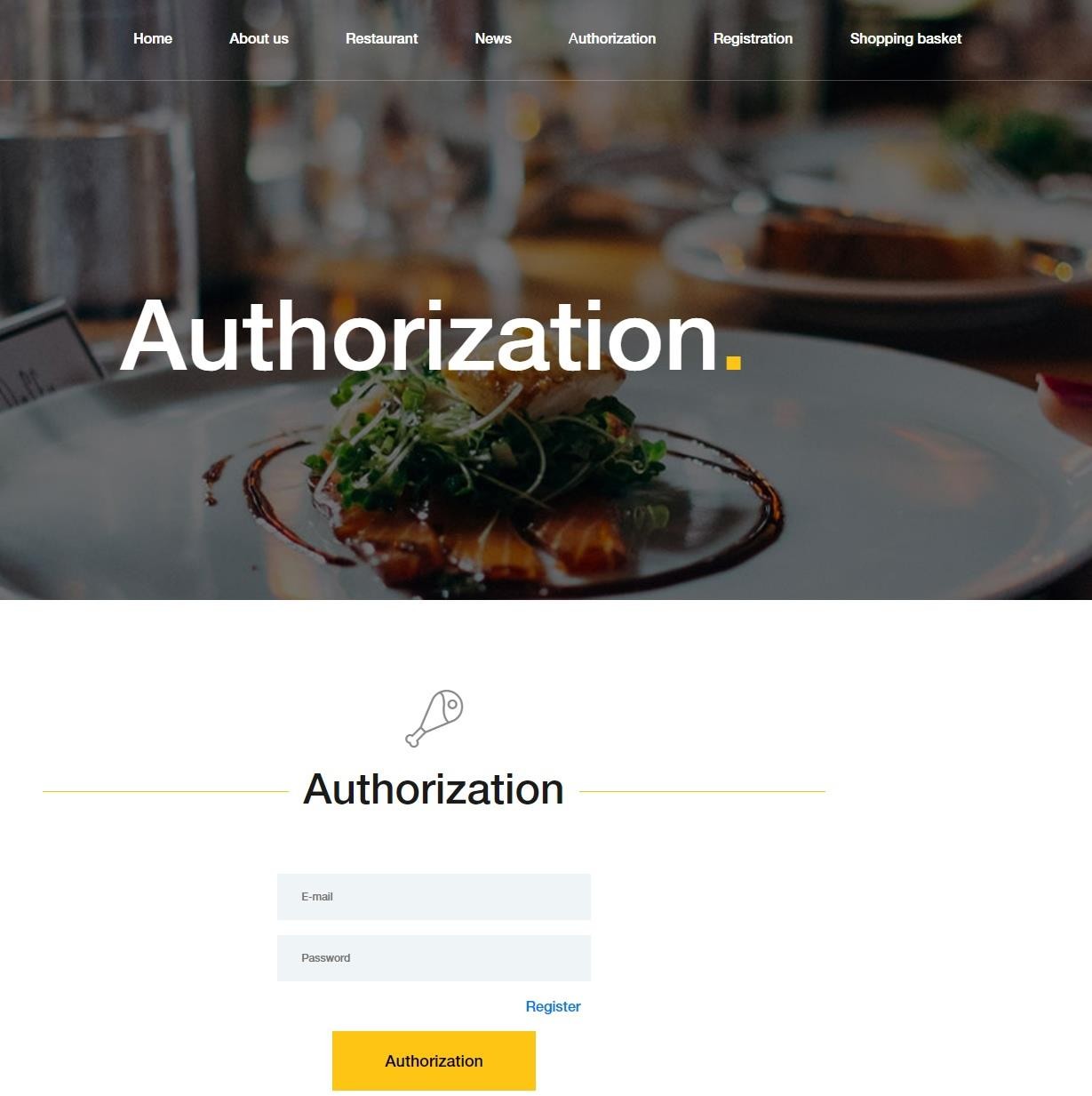


Рисунок 2.3.5.3 – Окно входа

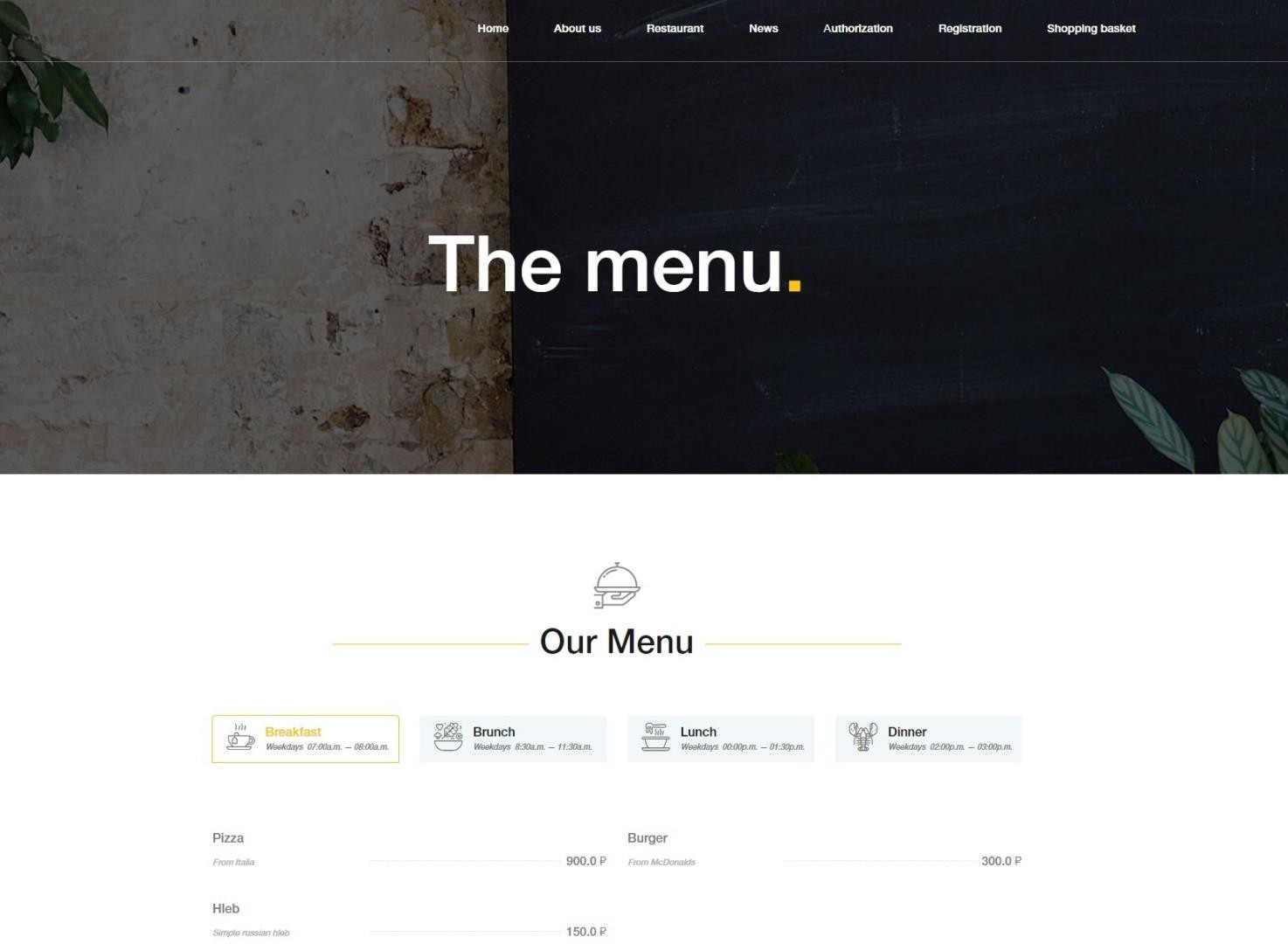


Рисунок 2.3.5.4 – Страница «Меню»

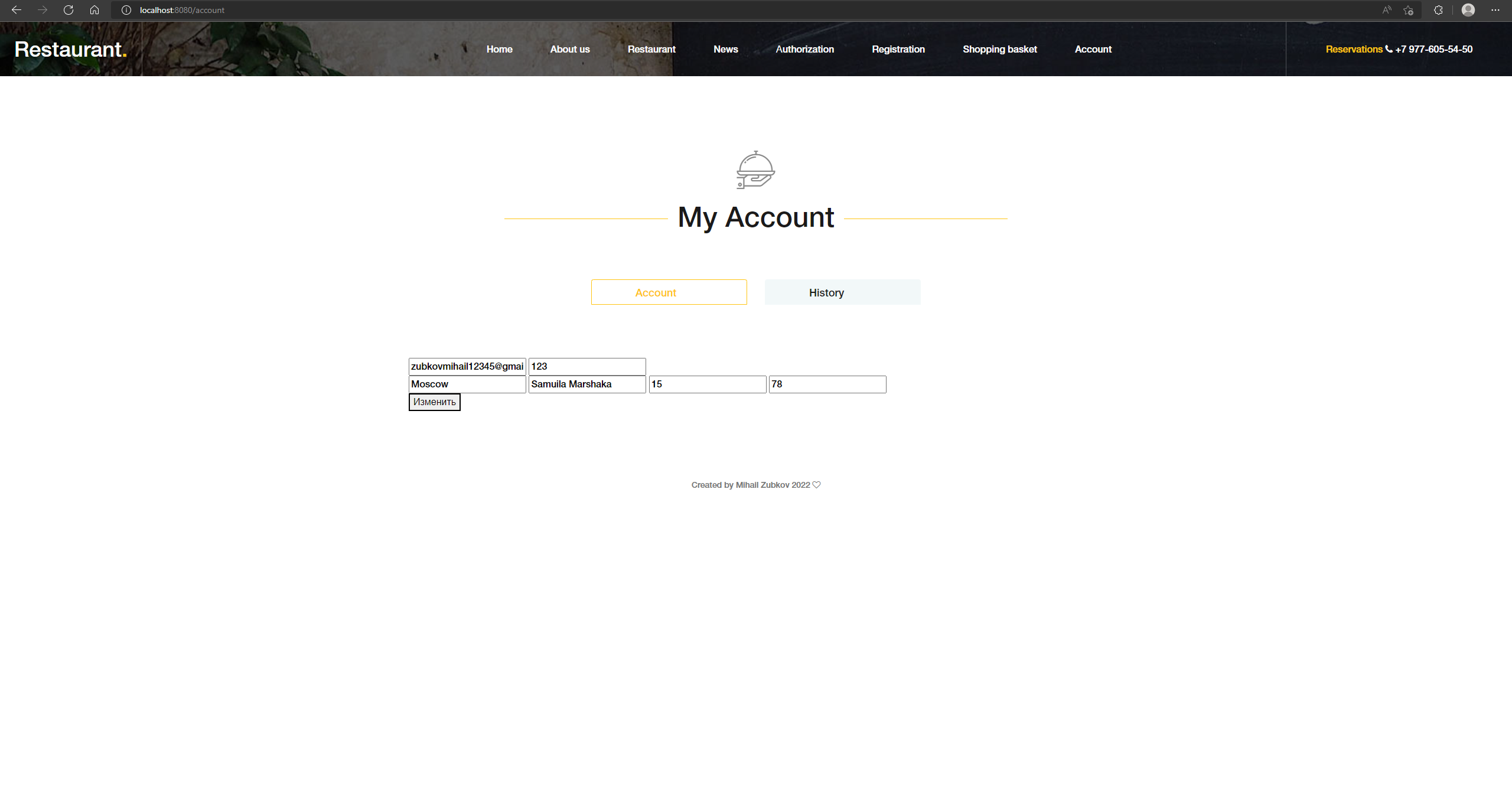


Рисунок 2.3.5.5 – Страница «Личный кабинет»

### Создание сервисов. Services.

Далее были подключены сервисы [5] (Рисунок 2.3.6.1). Данная часть кода помогает осуществить бизнес-логику серверной части, в ней прописывается та работа, которая не была прописана в контроллерах, а также запуск приложения

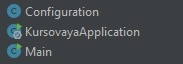


Рисунок 2.3.6.1 – Сервисы

## 2.4 Диаграмма классов и таблиц серверной части

Ниже представлена диаграмма таблиц базы данных(Рисуник 2.4.1) и диаграмма классов проекта(Рисуник 2.4.2).

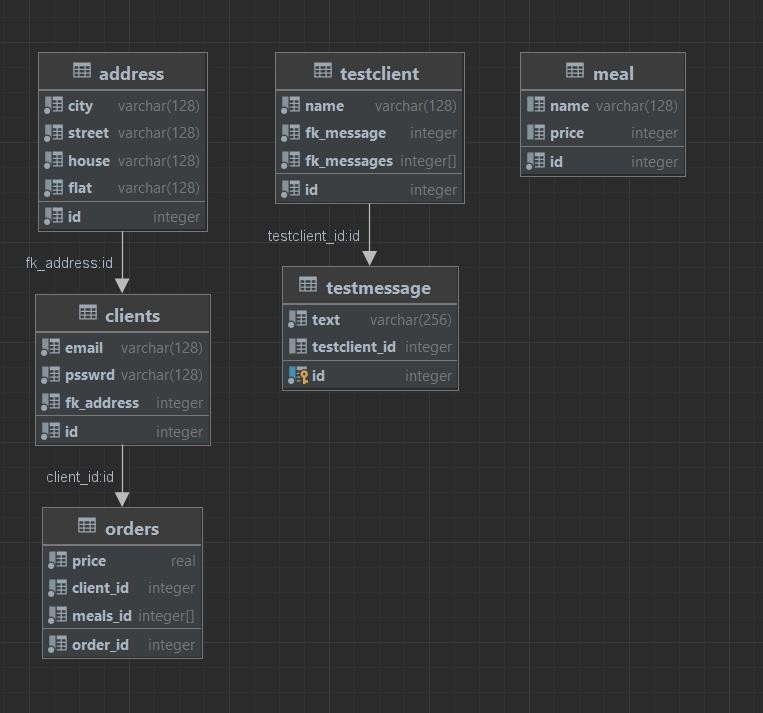


Рисунок 2.4.1 – Диаграмма таблицы базы данных

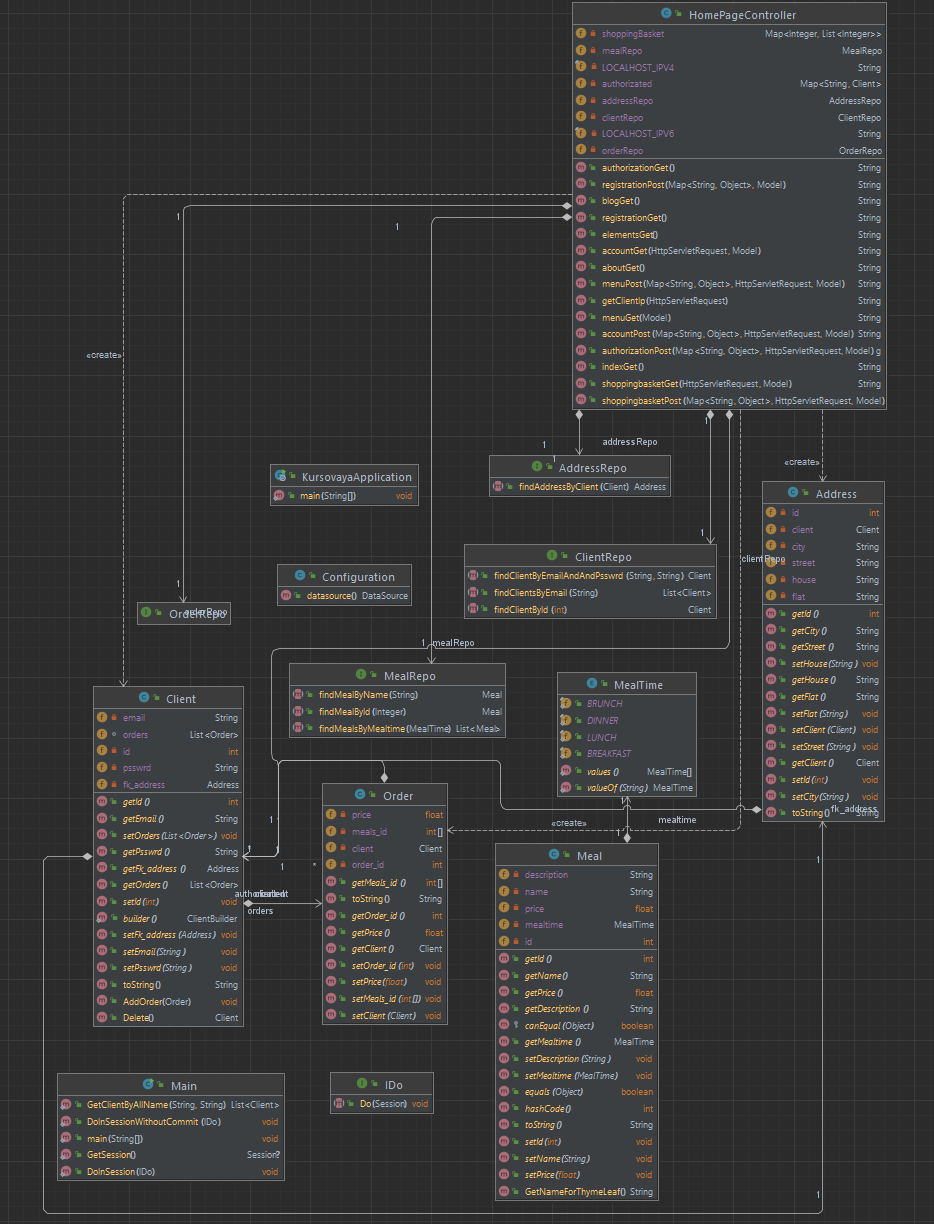


Рисунок 2.4.2 – Диаграмма классов проекта

## ВЫВОДЫ К РАЗДЕЛУ 2

В ходе анализа технологий разработки и формирования требований к комфорту разработчика и пользователей, был произведен выбор технологий разработки.

Было разработано веб-приложение с использованием фреймворков Spring и Bootstrap.

# 3. ТЕСТИРОВАНИЕ ВЕБ-ПРИЛОЖЕНИЯ

В данном разделе производится тестирование веб-приложения с помощью Postman

## Технология проведения тестирования

Postman – ПО, позволяющие тестировать веб-приложение, посредством отправки HTTP-запросов. В настройках запроса задаются данные для отправки и после отправляется сам запрос.

Как видно на рисунках 3.1.1-4, все запросы были успешны.

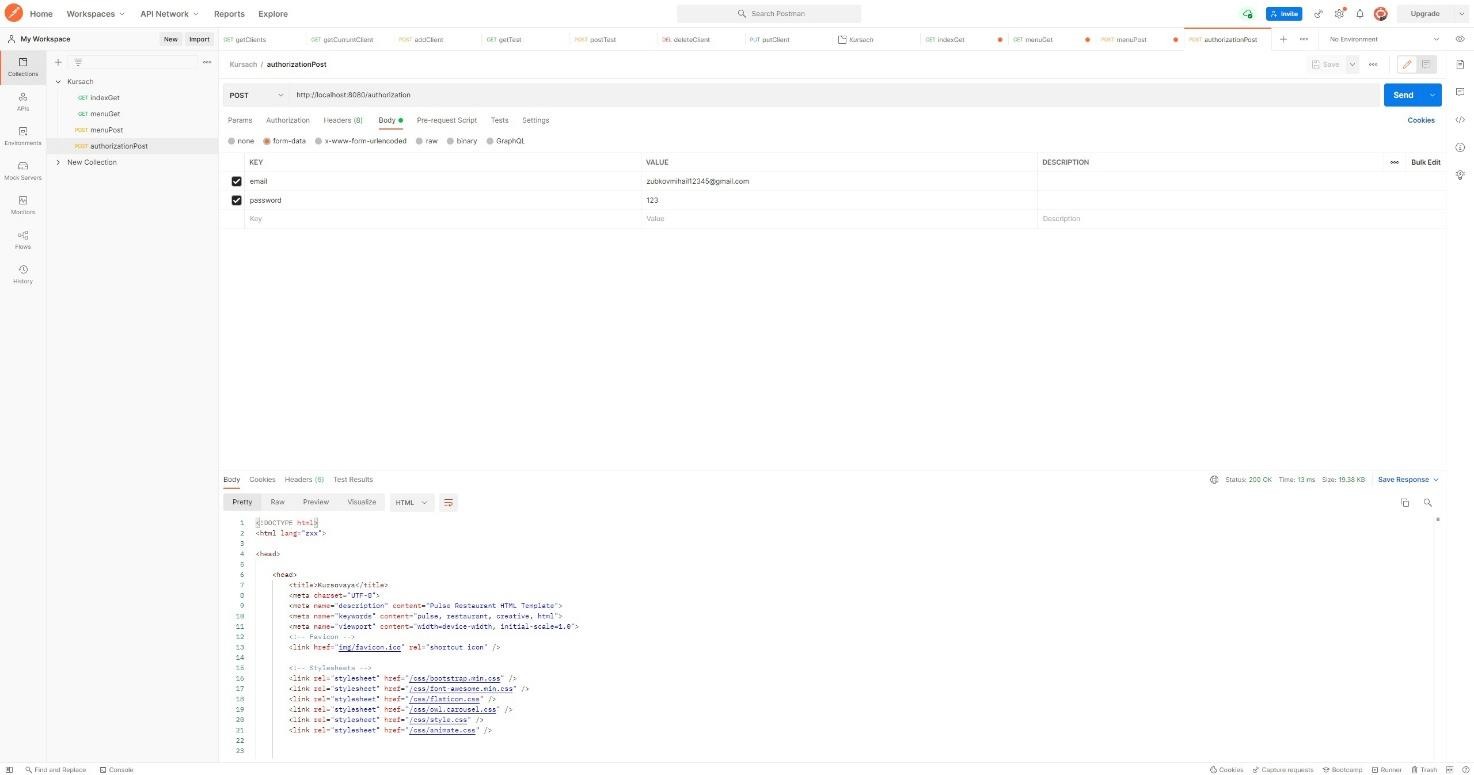


Рисунок 3.1.1 – Успешный запрос

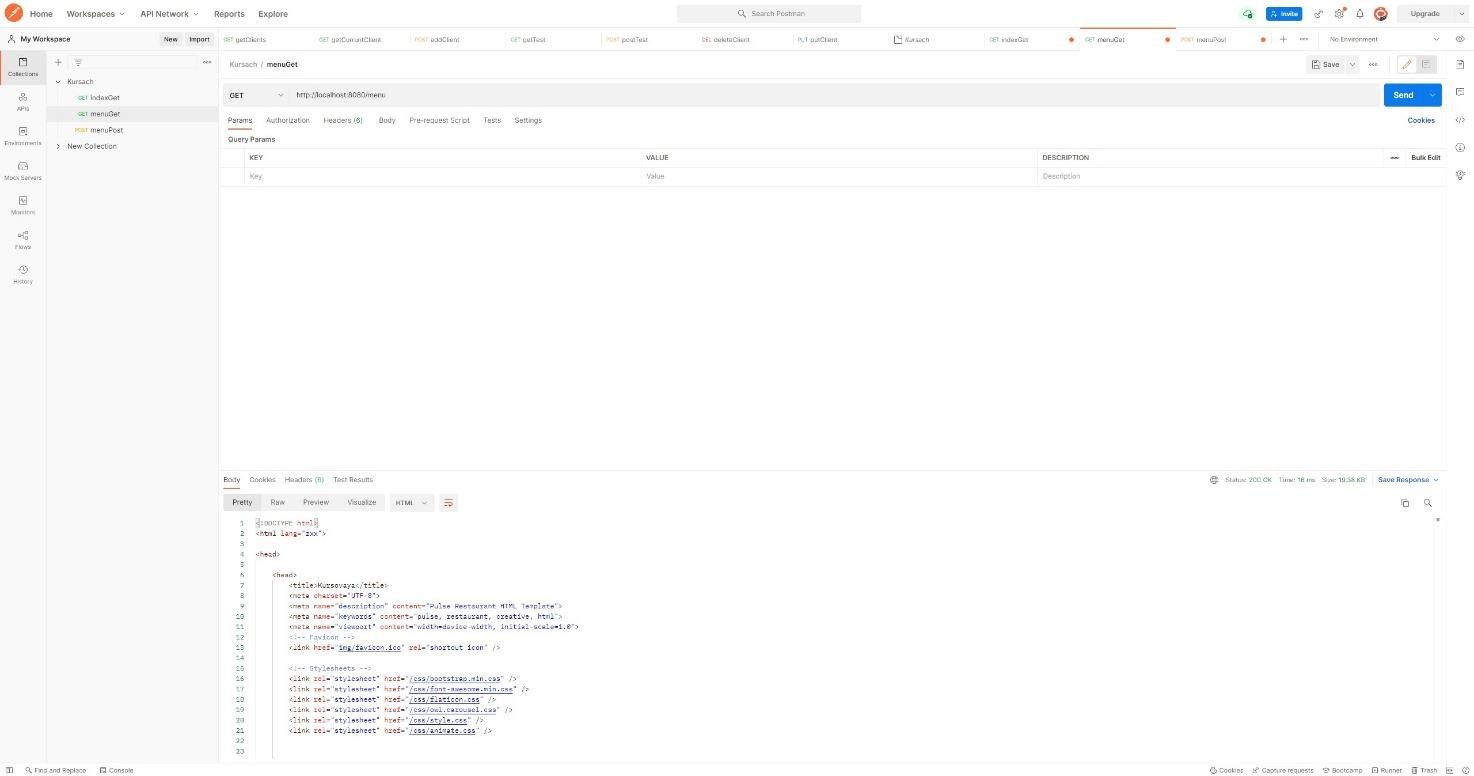


Рисунок 3.1.2 – Успешный запрос

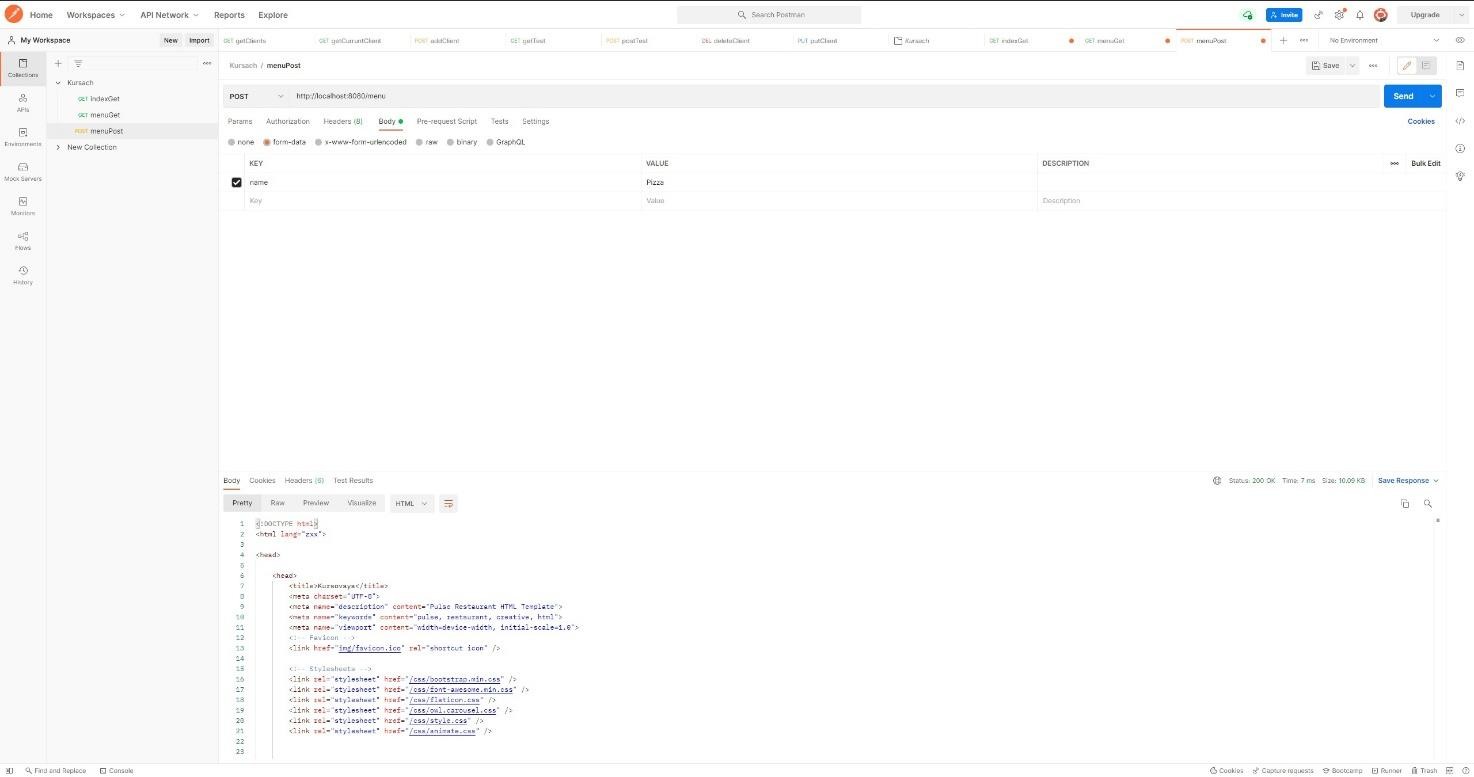


Рисунок 3.1.3 – Успешный запрос

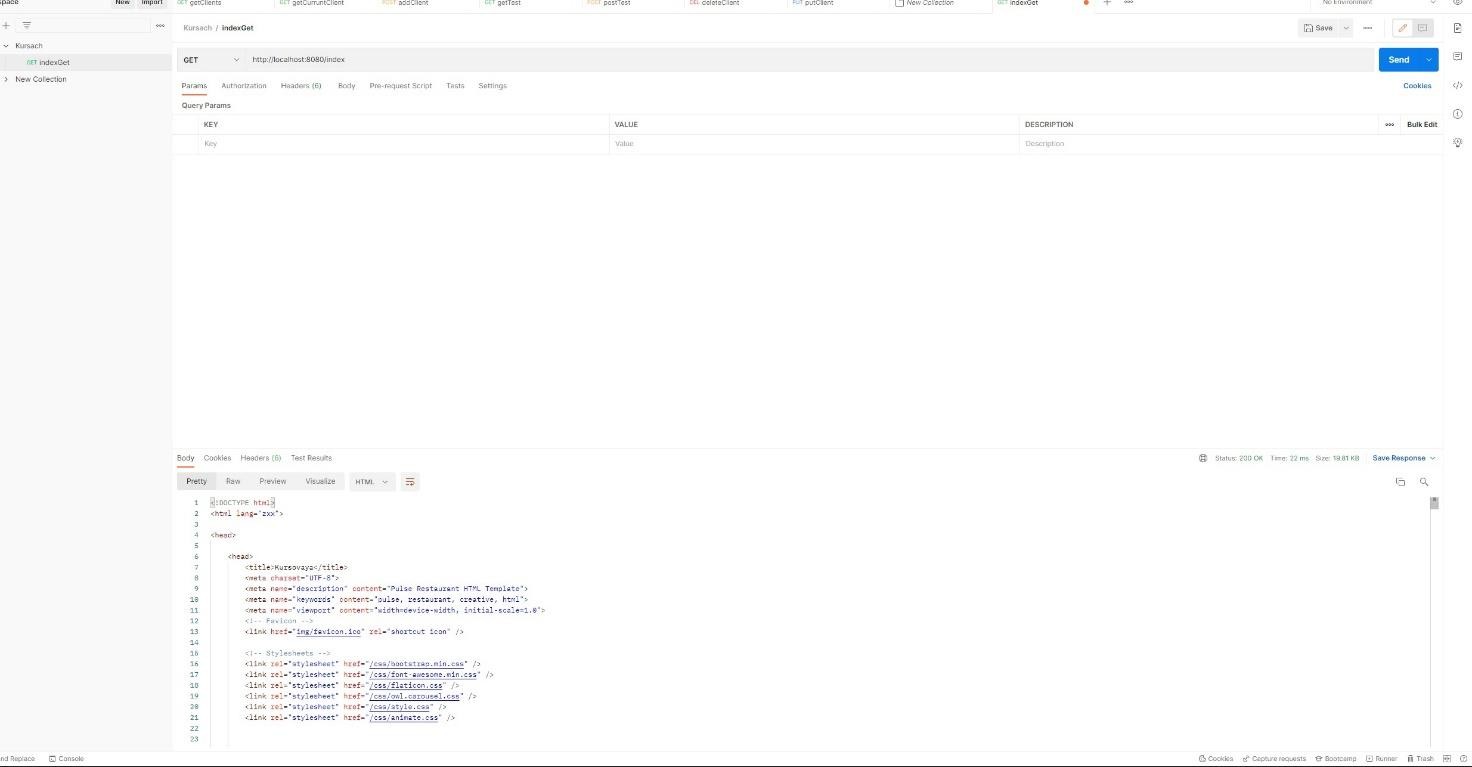


Рисунок 3.1.4 – Успешный запрос

## Проблемы с тестированием регистрации и аутентификации

К сожалению, тестирование регистрации и аутентификации посредством отправки запроса с данными для регистрации невозможно без отключения автоматически подключаемой защиты веб-приложения, защищающей ее от CSRF-атак.

## ВЫВОДЫ К 3 РАЗДЕЛУ

Было проведено тестирование разработанного веб-приложения.

# ЗАКЛЮЧЕНИЕ

В процессе выполнения данной курсовой работы был проведен анализ предметной области, в ходе которого были рассмотрены различные интернет- ресурсы на тему «Ресторан». Исходя из анализа, были составлены требования к веб-приложению.

Был проведен анализ технологий в ходе которого были сформированы требования и выбраны инструменты для разработки веб-приложения.

Было разработано веб-приложение с использованием Spring Framework и его компонентов, а также с использованием фреймворка для фронтенд разработки Bootstrap.

Было проведено тестирование веб-приложения, посредством работы с Postman.

В процессе работы над курсовой работой были приобретены следующие компетенции:

ПК-1 - способен выполнять разработку и интеграцию программных модулей и компонент, верификацию выпусков программных продуктов информационных систем

ПК-1.1 - Знать: методы и средства сборки модулей и компонент программного обеспечения при создании информационных систем

ПК-1.12 - Уметь: применять методы и средства создания программных интерфейсов информационных систем

ПК-1.14 - Владеть: разработкой процедур сборки модулей и компонент программного обеспечения при внедрении информационных систем

# СПИСОК ИСПОЛЬЗОВАНЫХ ИСТОЧНИКОВ

1. Spring Initializr [Электронный ресурс] Режим доступа: https://start.spring.io/, свободный (Дата последнего обращения: 15.05.2022)
2. PostgreSQL Documentation [Электронный ресурс] URL: https://[www.postgresql.org/docs/,](http://www.postgresql.org/docs/) свободный (Дата последнего обращения: 15.05.2022)
3. Spring Data JPA [Электронный ресурс] URL: https://spring.io/projects/spring-data-jpa/, свободный (Дата последнего обращения: 15.05.2022)
4. Принцип Mobile First при разработке дизайна сайтов [Электронный ресурс] URL: https://[www.site2b.ua/web-blog/mobile-first-website-design.html,](http://www.site2b.ua/web-blog/mobile-first-website-design.html) свободный (Дата последнего обращения: 15.05.2022)

5. Spring Data JPA. Пишем DAO и Services. [Электронный ресурс] URL: https://devcolibri.com/spring-data-jpa-%D0%BF%D0%B8%D1%88%D0%B5%D0%BC-dao-%D0%B8-services- %D1%87%D0%B0%D1%81%D1%82%D1%8C-2/, свободный (Дата последнего обращения: 15.05.2022)

# ПРИЛОЖЕНИЕ

В связи с объемностью работы, код будет размещен Github - https://github.com/s1ngle0f/kursovaya