Изображение выглядит как текст, коллекция картинок

Автоматически созданное описание

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: Разработка серверных частей интернет-ресурсов

по профилю: Разработка программных продуктов и проектирование информационных систем

направления профессиональной подготовки: Программная инженерия (09.03.04)

Тема: Серверная часть веб-приложения «Ресторан»

Студент: Зубков Михаил Витальевич

Группа: ИКБО-30-20

Работа представлена к защите (дата) /Зубков М. В. /

(подпись и ф.и.о. студента)

Руководитель: Синицын Анатолий Васильевич, старший преподаватель

Работа допущена к защите (дата) /Синицын А.В. /

(подпись и ф.и.о. рук-ля)

Оценка по итогам защиты:

/ /

/ /

(подписи, дата, ф.и.о., должность, звание, уч. степень двух преподавателей, принявших защиту)

М. РТУ МИРЭА. 2022 г.

Изображение выглядит как текст

Автоматически созданное описание

# АННОТАЦИЯ

Зубков М. В. Веб-приложение на тему «Ресторан». Курсовая работа по направлению подготовки бакалавров 09.03.04 «Программная инженерия» на тему «Ресторан» с использованием технологий HTML5, CSS3, Java, Jet brains Intellij IDEA, PostgreSQL СУБД.

В ведении обосновывается актуальность выбранной темы, а также для какого типа людей будет интересен веб-приложение «Ресторан».

В отчете детально были разобраны методы разработки веб-приложения «Ресторан». Основываясь на выбранном стеке технологий, были детально описано методы взаимодействия различных сервисов и проделанная работа с ними. В конце работы был реализован веб-приложение «Ресторан».

В заключении описываются какие цели были достигнуты и рассказывается какая работа была проделана для этого.

**СОДЕРЖАНИЕ**

[АННОТАЦИЯ 3](#_Toc121694889)

[ВВЕДЕНИЕ 7](#_Toc121694890)

[СБОР И АНАЛИЗ ТРЕБОВАНИЙ К ПРОГРАММНОМУ ПРОДУКТУ 8](#_Toc121694891)

[1.1 Анализ предметной области 8](#_Toc121694892)

[ВЫВОДЫ К РАЗДЕЛУ 1 10](#_Toc121694893)

[ОПРЕДЕЛЕНИЕ ТЕХНОЛОГИЙ РАЗРАБОТКИ СЕРВИСА 11](#_Toc121694894)

[2.1 Технологии разработки 11](#_Toc121694895)

[2.2 Паттерн проектирования 12](#_Toc121694896)

[ВЫВОДЫ К РАЗДЕЛУ 2 13](#_Toc121694897)

[РАЗРАБОТКА АРХИТЕКТУРЫ ПРИЛОЖЕНИЯ 14](#_Toc121694898)

[3.1 Разработка архитектуры на основе выбранного паттерна 14](#_Toc121694899)

[ВЫВОД К РАЗДЕЛУ 3 17](#_Toc121694900)

[РАЗРАБОТКА СЕРВЕРНОЙ ЧАСТИ ИНТЕРНЕТ-РЕСУРСА 18](#_Toc121694901)

[4.1 Создание проекта и добавление зависимостей 18](#_Toc121694902)

[4.2 Структура веб-приложения 18](#_Toc121694903)

[4.3 Процесс создания веб-приложения 20](#_Toc121694904)

[4.3.1 Создание проекта и добавление зависимостей 20](#_Toc121694905)

[4.3.2 Создание сущностей и заполнение базы данных. Entity. 22](#_Toc121694906)

[4.3.3 Создание репозиториев. Repository. 26](#_Toc121694907)

[4.3.4 Создание контроллеров. Controllers. 27](#_Toc121694908)

[4.4 Создание клиентской части. Bootstrap. 28](#_Toc121694909)

[4.5 Создание сервисов. Services. 32](#_Toc121694910)

[4.6 Диаграмма классов и таблиц серверной части 32](#_Toc121694911)

[ВЫВОДЫ К РАЗДЕЛУ 4 34](#_Toc121694912)

[ЗАКЛЮЧЕНИЕ 35](#_Toc121694913)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 36](#_Toc121694914)

[ПРИЛОЖЕНИЕ 38](#_Toc121694915)

**СПИСОК ИСПОЛЬЗОВАННЫХ СОКРАЩЕНИЙ**

ЯП – Язык программирования

СУБД – Система управления базами данных

MVC – Паттерн проектирования (Model-View-Controller)[6]

HTML – HyperText Markup Language

БД – База данных

CSS – Cascading Style Sheets

# 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

## Анализ 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»[11] (Рисунок 1.1.1). На сайте представлен ассортимент и имеется личный кабинет, где есть возможность посмотреть баллы по программе лояльности.

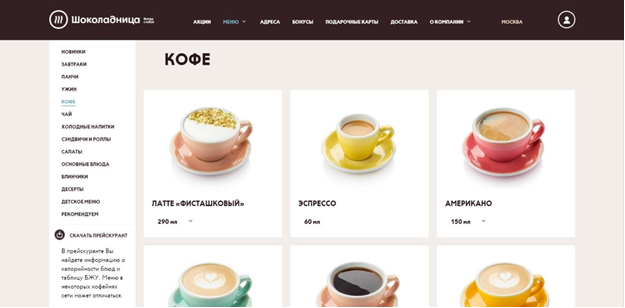


Рисунок 1.1.1 – Раздел «Меню» «Шоколадницы»

Не таким хорошим примером является сайт сети ресторанов «One Price»[12], 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»[13], но с одним недостатком – это невозможность посмотреть список заказанного (Рисунок 1.1.3)

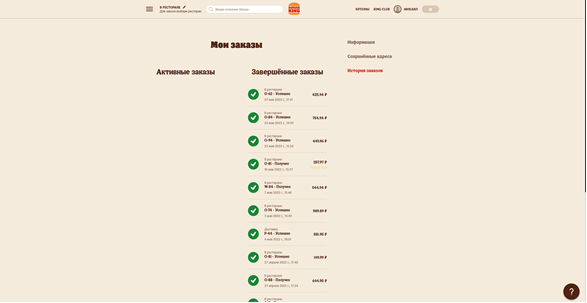


Рисунок 1.1.3 – Раздел «Мои заказы» «Burger King»

## 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;
6. Наличие ролей пользователей;

Следовательно, для реализации нам понадобится

1. Создать базу данных пользователей;
2. Создать базу данных для меню;
3. Создать базу данных для адресов;
4. Создать базу данных для заказов;
5. Осуществить авторизацию пользователей;

# ОПРЕДЕЛЕНИЕ ТЕХНОЛОГИЙ РАЗРАБОТКИ СЕРВИСА

## 2.1 Технологии разработки

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[14] – 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[7] 14 – строго типизированный объектно-ориентированный ЯП, позволяющий написать в данной курсовой работе серверную часть. Почему выбор пал именно на Java, а не на PHP, есть несколько моментов: Java является строготипизированным языком, что позволяет реже допускать ошибки с типизацией данных, язык Java с его реализацией бэкенда позволяет легче расширять функционал программы, также Java является более надежным языком в плане безопасности, больший набор библиотек.
3. Spring ORM[19] – модуль Spring, включающий в себя Spring Data JPA и Hibernate[18], которые будут нужны для реализации базы данных;
4. PostgreSQL – свободная объектно-реляционная[20] 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[16] – фреймворк для автоматизации сборки проектов на основе описания их структуры в файлах на языке POM (англ. Project Object Model), являющемся подмножеством XML;
6. Github[15] – репозиторий работ, позволяющий хранить проект удаленно в целях его мобильности, а также безопасности в случае удаления всех данных с локальной машины;
7. Bootstrap[17] – 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

## 2.2 Паттерн проектирования

Далее рассмотрим наиболее подходящий паттерн проектирования для веб-приложения «Ресторан», который будет сочетаться с технологиями, представленными выше. В качестве паттерна проектирования был выбран MVC (Model-View-Controller). Этот паттерн позволяет разделить логику, и создавать части проекта, отвечающие каждый за свою обязанность, отдельно друг от друга.

## ВЫВОДЫ К РАЗДЕЛУ 2

Подводя итог всего вышесказанного, мною были выбраны оптимальные технологии разработки, с помощью которых будет реализовано веб-приложение «Ресторан».

# РАЗРАБОТКА АРХИТЕКТУРЫ ПРИЛОЖЕНИЯ

## 3.1 Разработка архитектуры на основе выбранного паттерна

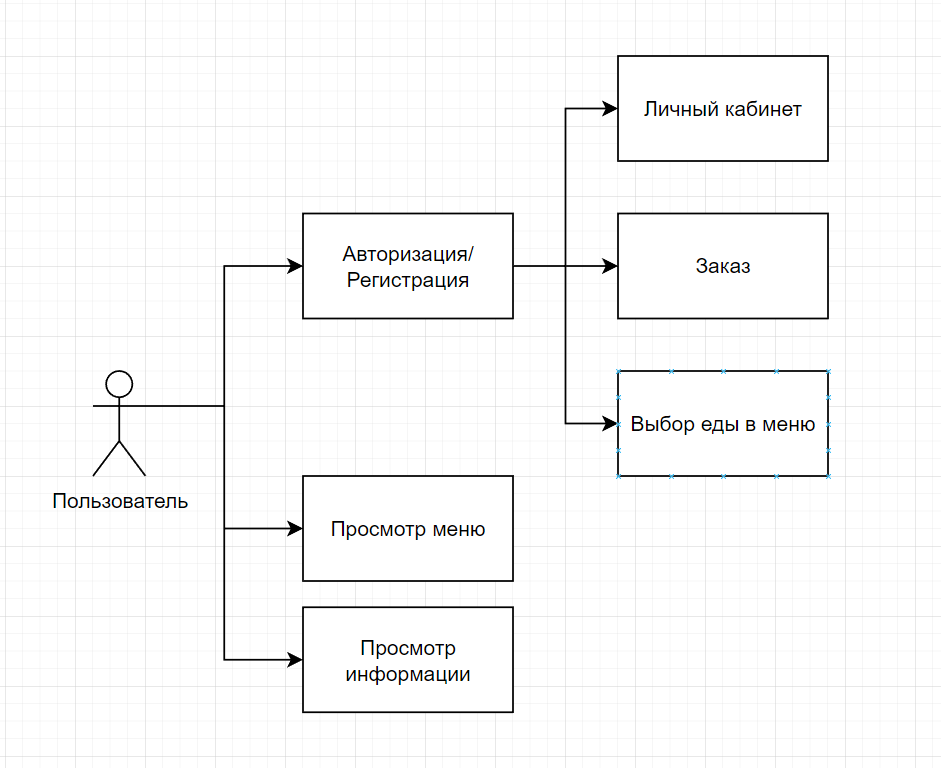
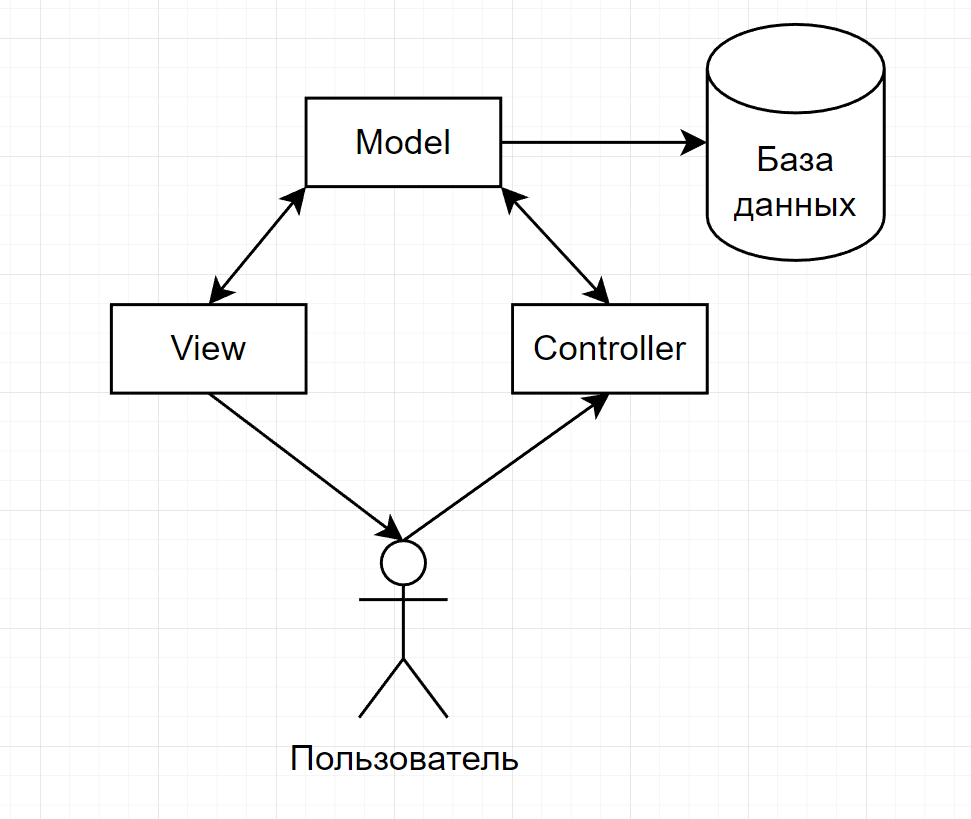
Перед началом разработки архитектуры интернет-ресурса необходимо спроектировать алгоритм работы сервиса путем создания диаграммы доступного функционала для пользователя (Рисунок 3.1)

Рисунок 3.1 – Функционал пользователя

Далее отобразим работу паттерна проектирования, в моем случае MVC, в условиях веб-приложения. Отличие данного паттерна от остальных заключается в том, что проект можно разделить на три части: логика проекта, контроллеры, представление для пользователя (Рисунок 3.2).

Рисунок 3.2 – Паттерн проектирования MVC

Рассмотрим работу элементов, представленных на диаграмме:

* Модель (Model) – является так называем мостом между представлением (view) и контроллером (controller). Она используется для обеспечения доступа к данным для из просмотра, записи новых значений, удаления и т.д.
* Представление (View) – используется для отображения данных пользователю, после получения данных из контроллера и модели. Они не обращаются к базе данных, не работают с данными, которые были получены из запроса пользователя.
* Контроллер (Controller) – основной компонент данного паттерна проектирования. Он связывает модели, представления и другие вспомогательные компоненты в одно целое. Контроллеры отвечают за обработку запросов от пользователя.

При использовании паттерна проектирования MVC интернет-ресурс делится на 3 слоя:

* Слой данных[8] – хранение данных
* Слой представления[8] – взаимодействие с пользователем
* Слой приложения[8] – посредник между моделью и представлением

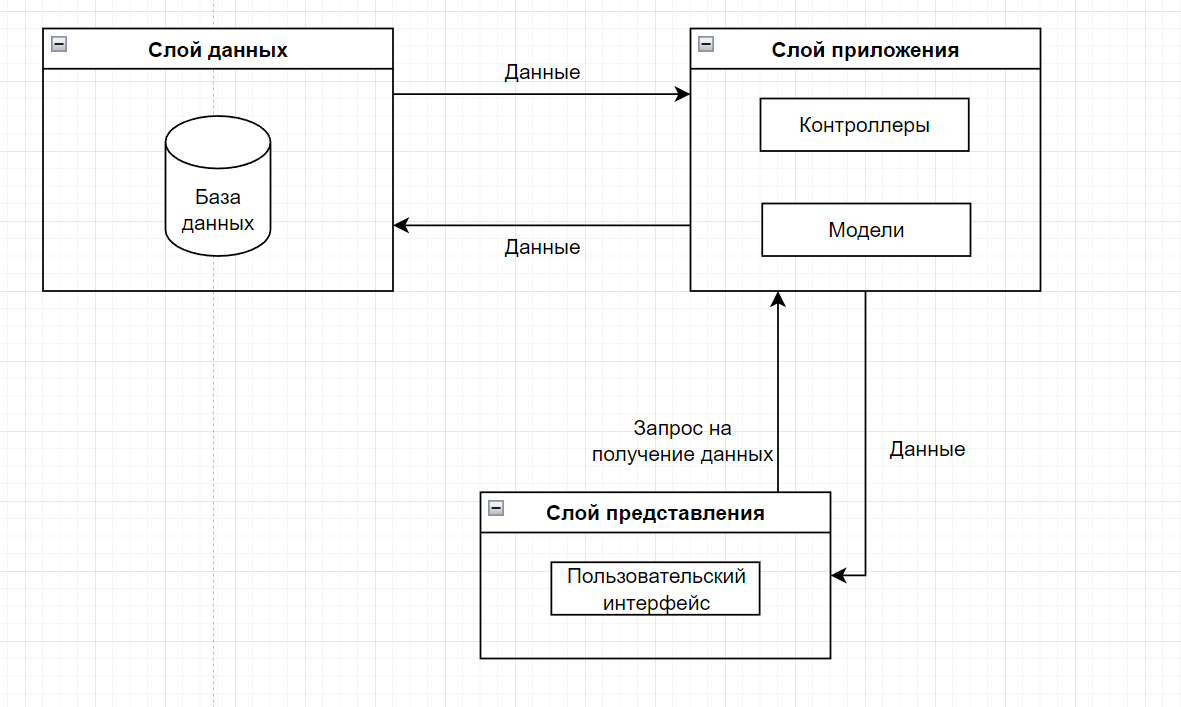
Визуальное представление данной архитектуры интернет-ресурса на основе паттерна MVC представлено на Рисунке 3.3

Рисунок 3.3 – Архитектура интернет-ресурса

## ВЫВОД К РАЗДЕЛУ 3

Итогом является выбор паттерна MVC, из-за простоты использования, разделения обязанностей между слоями проекта, а также именно этот паттерн больше всего подходит под данный интернет-ресурс.

# РАЗРАБОТКА СЕРВЕРНОЙ ЧАСТИ ИНТЕРНЕТ-РЕСУРСА

## 4.1 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[3] для дальнейшей работы с БД;
2. Thymeleaf[9] для связи серверной части с клиентской;
3. Lombok[10] для облегчения кода и автоматизации разработки;
4. PostgreSQL[2], Hibernate для реализации базы данных, где будут находится данные о пользователях, задачах и решениях.

## 4.2 Структура веб-приложения

Проект состоит из 4 основных частей:

1. Файлы серверной части, написанные с использованием Java и Spring Framework (Рисунок 4.2.1), включающие в себя контроллеры, модели, сервисы;

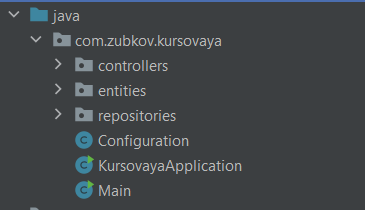


Рисунок 4.2.1 – Файлы, осуществляющие серверную логику

1. Файлы клиентской части, включающие в себя файлы Bootstrap Framework (Рисунок 4.2.2);

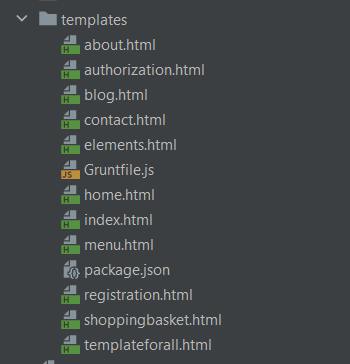


Рисунок 4.2.2 – Файлы клиентской части веб-приложения

1. Файлы Apache Maven, сборщика проекта (Рисунок 4.2.3);

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 4.2.3 – Файлы Maven, в которых происходит добавление зависимостей в проект

4. База данных PostgreSQL, в которой осуществляется хранение данных пользователей, задач и решений (Рисунок 4.2.4).

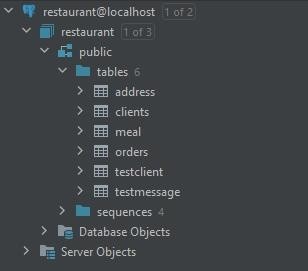


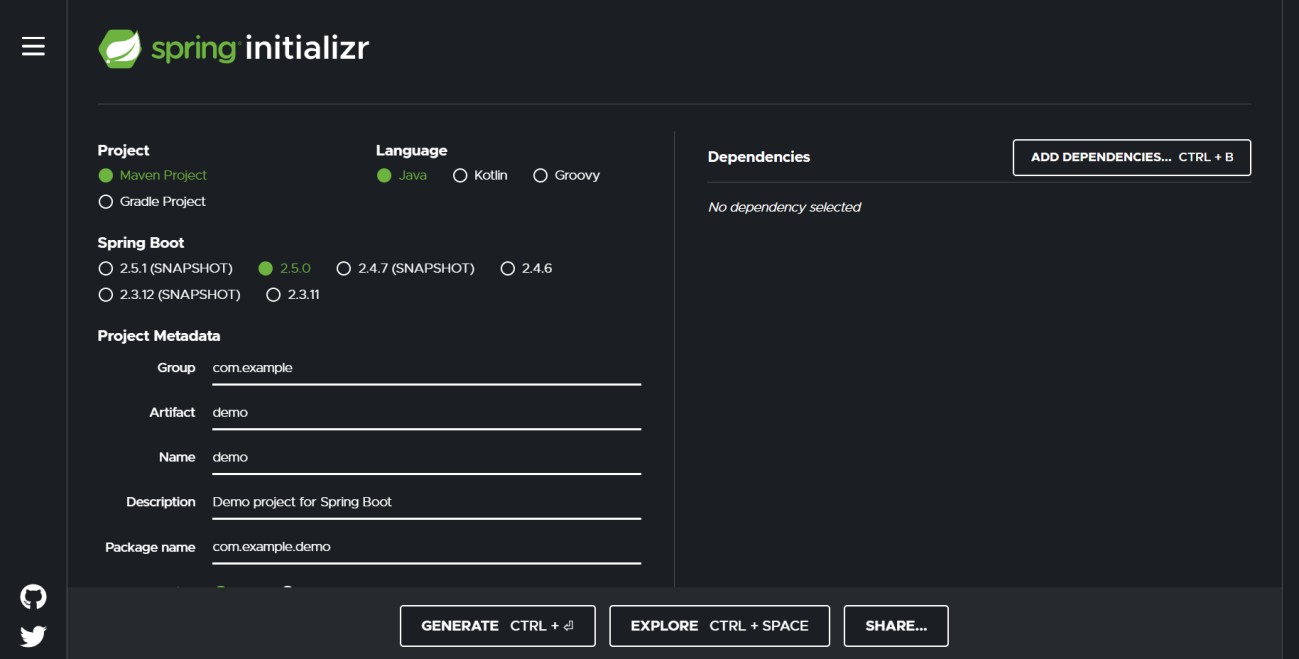
Рисунок 4.2.4 – База данных PostgreSQL

## 4.3 Процесс создания веб-приложения

## 4.3.1 Создание проекта и добавление зависимостей

При создании веб-приложения, первым делом через Spring Initializr[1] (Рисунок 4.3.1.1), сайт, позволяющий автоматизировать процесс создания проекта, был инициализирован проект. Далее были подключены зависимости проекта в maven-файл pom.xml (Рисунок 4.3.1.2), 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.



Изображение выглядит как текст

Автоматически созданное описаниеРисунок 4.3.1.1 – Сайт Spring Initializr

Рисунок 4.3.1.2 – Часть файла pom.xml с частью представленных зависимостей

## 4.3.2 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 , и задание сущностей с отсылкой на них.

Сущность Client для задания данных о пользователей содержит 5 полей (Рисунок 4.3.2.1):

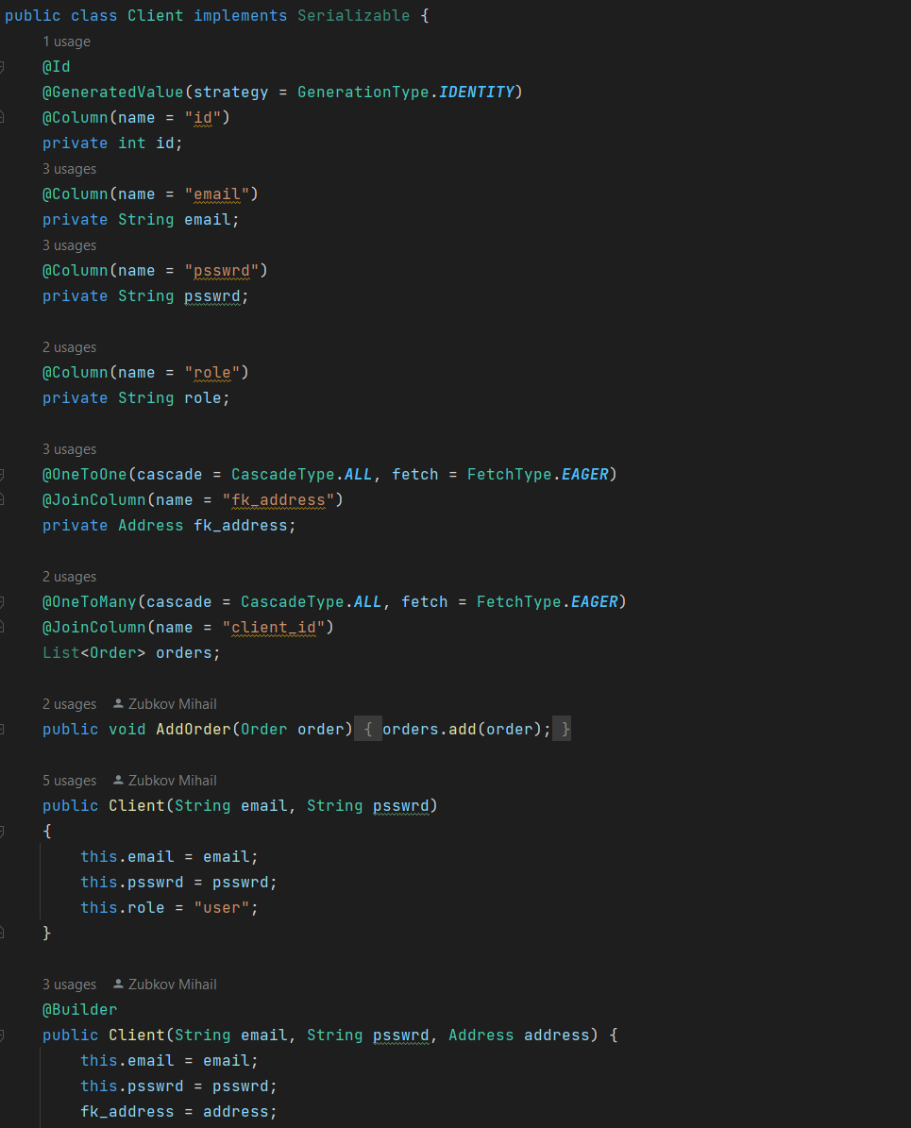
1. id – для задания id пользователя;
2. email – почта пользователя;
3. psswrd – пароль;
4. fk\_address – экземпляр класса адреса пользователя;
5. orders – поле списка заказов пользователя.
6. role – поле, отвечающее за возможности клиента

Рисунок 4.3.2.1 – Файл Client.java

Сущность Meal для еды из меню содержит 5 полей (Рисунок 4.3.2.2):

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

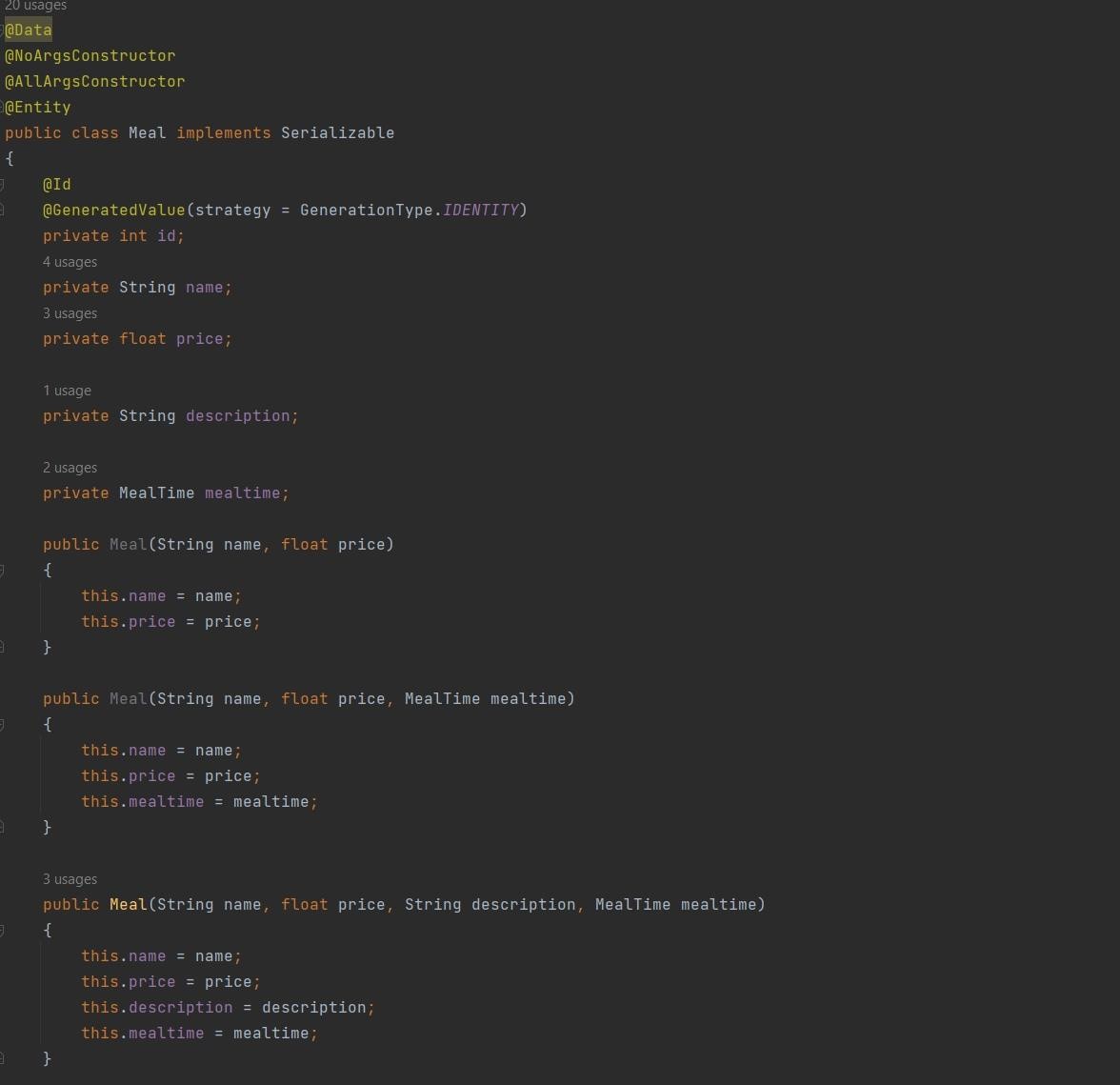


Рисунок 4.3.2.2 – Сущность «Meal»

Сущность Order, которая содержит 5 полей (Рисунок 4.3.2.3):

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;

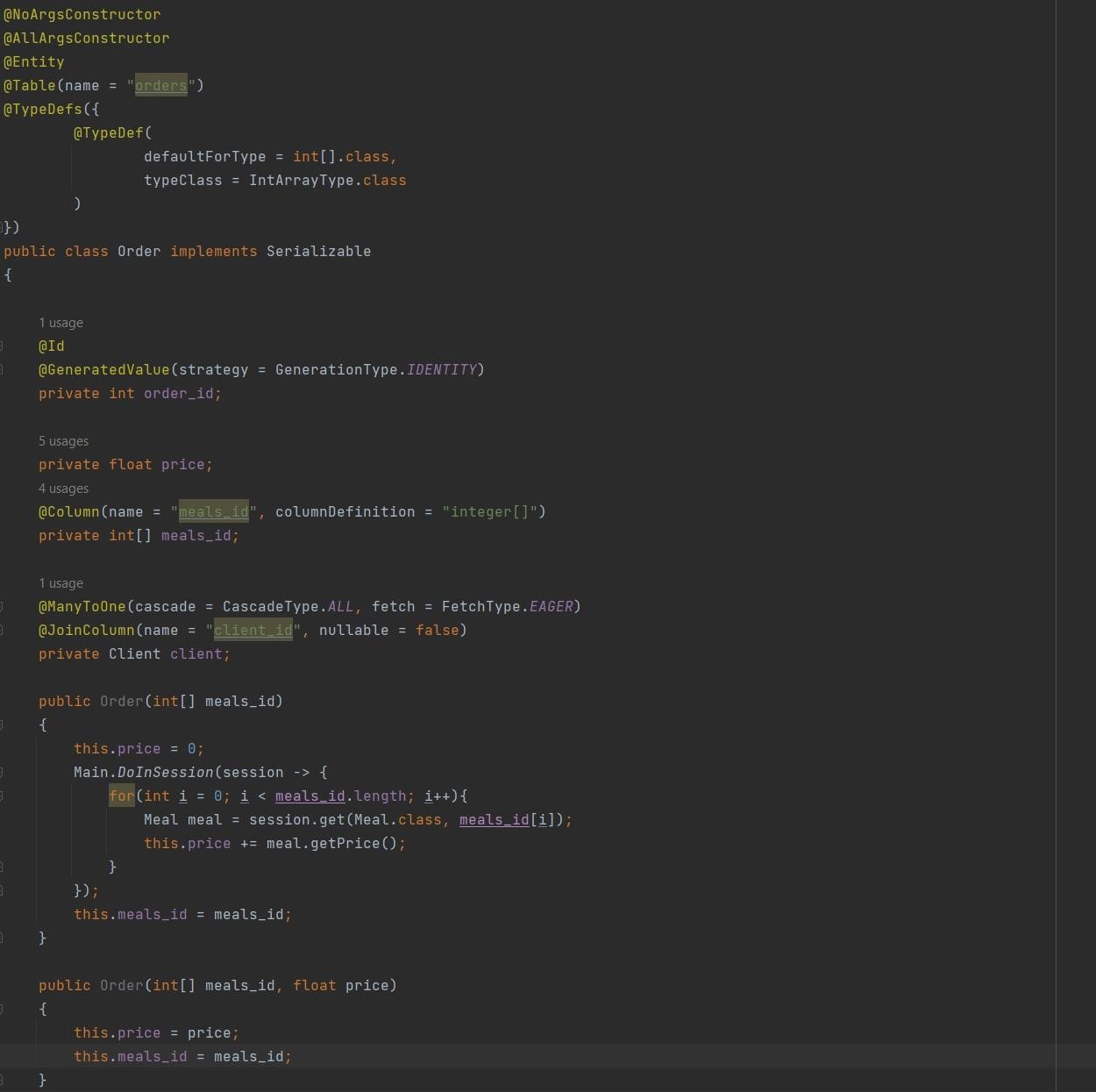


Рисунок 4.3.2.3 – Сущность «Order»

Сущность Address, которая содержит 6 полей (Рисунок 4.3.2.4):

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

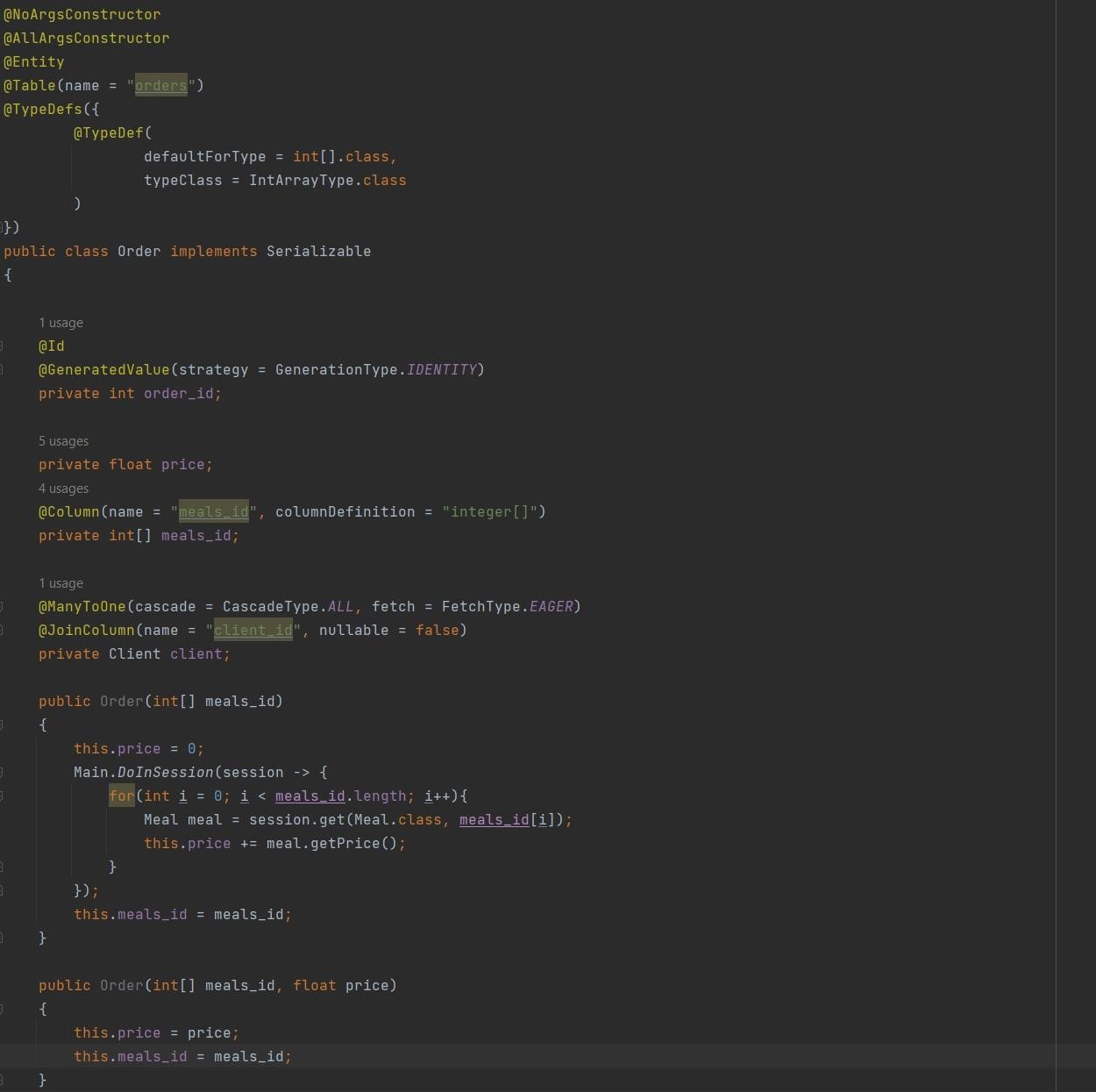


Рисунок 4.3.2.4 – Сущность «Address»

## 4.3.3 Создание репозиториев. Repository.

После задания баз данных и создания сущностей для ее связи с прописываемой логикой серверной части, следующим шагом является создание репозиториев (Рисунки 4.3.3.1-2). 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.

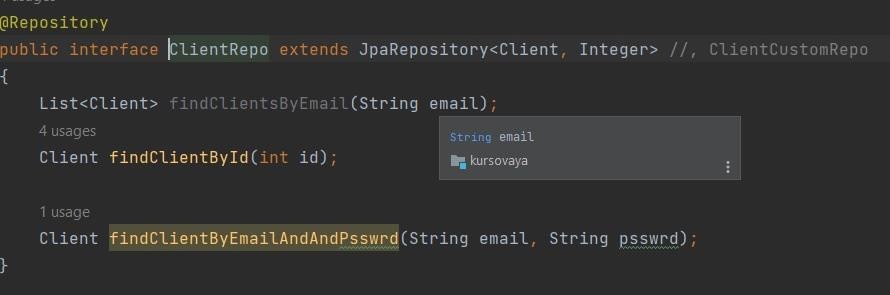


Рисунок 4.3.3.1 – Репозиторий для сущности Client



Рисунок 4.3.3.2 – Репозиторий для сущности Meal

## 4.3.4 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 4.3.4.1). Всего получилось 2 контроллера. (Рисунок 4.3.4.2)

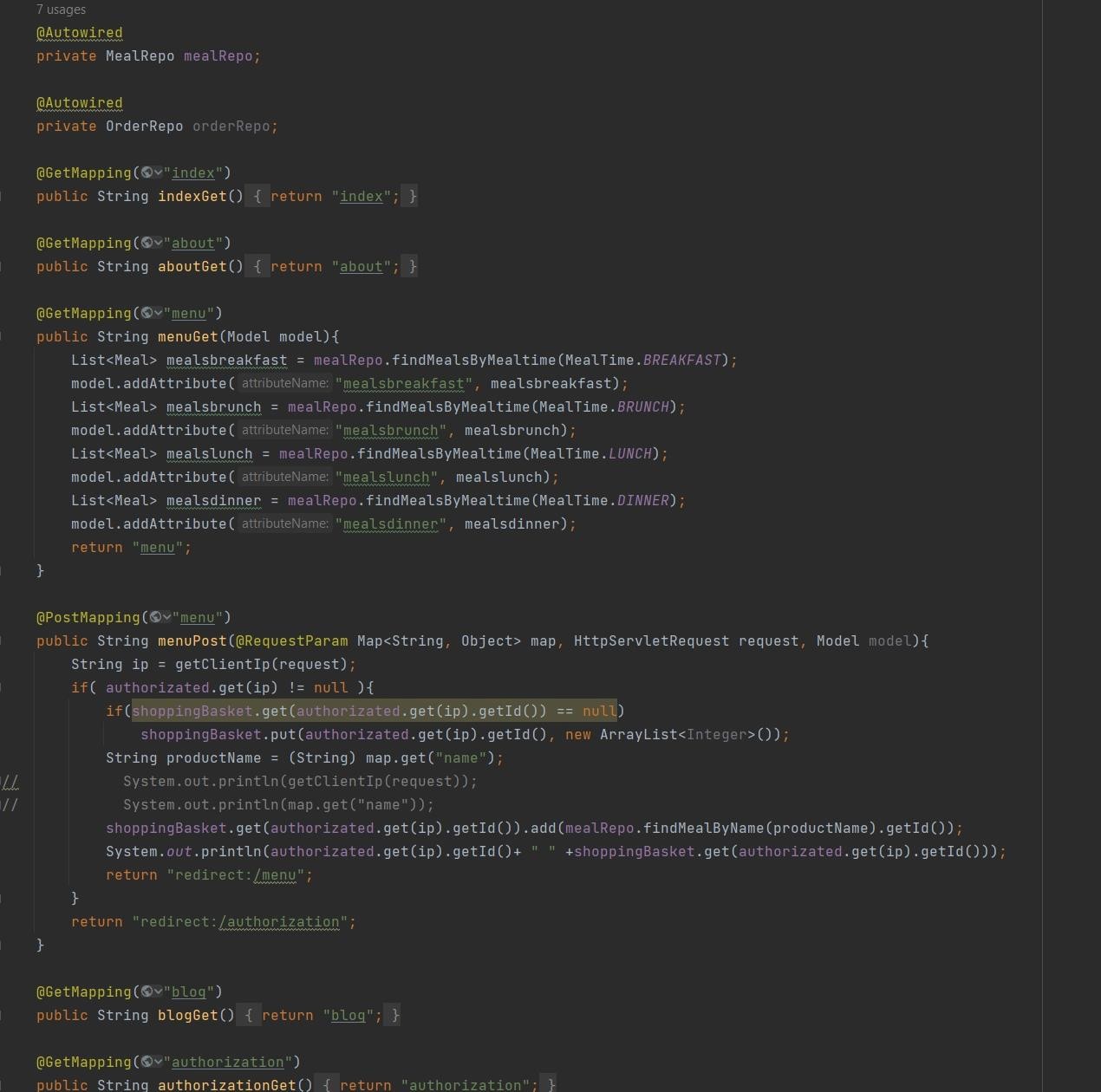


Рисунок 4.3.4.1 – Пример кода из одного из контроллеров с аннотациями

GetMapping и PostMapping



Рисунок 4.3.4.2 – Контроллеры

## 4.4 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:

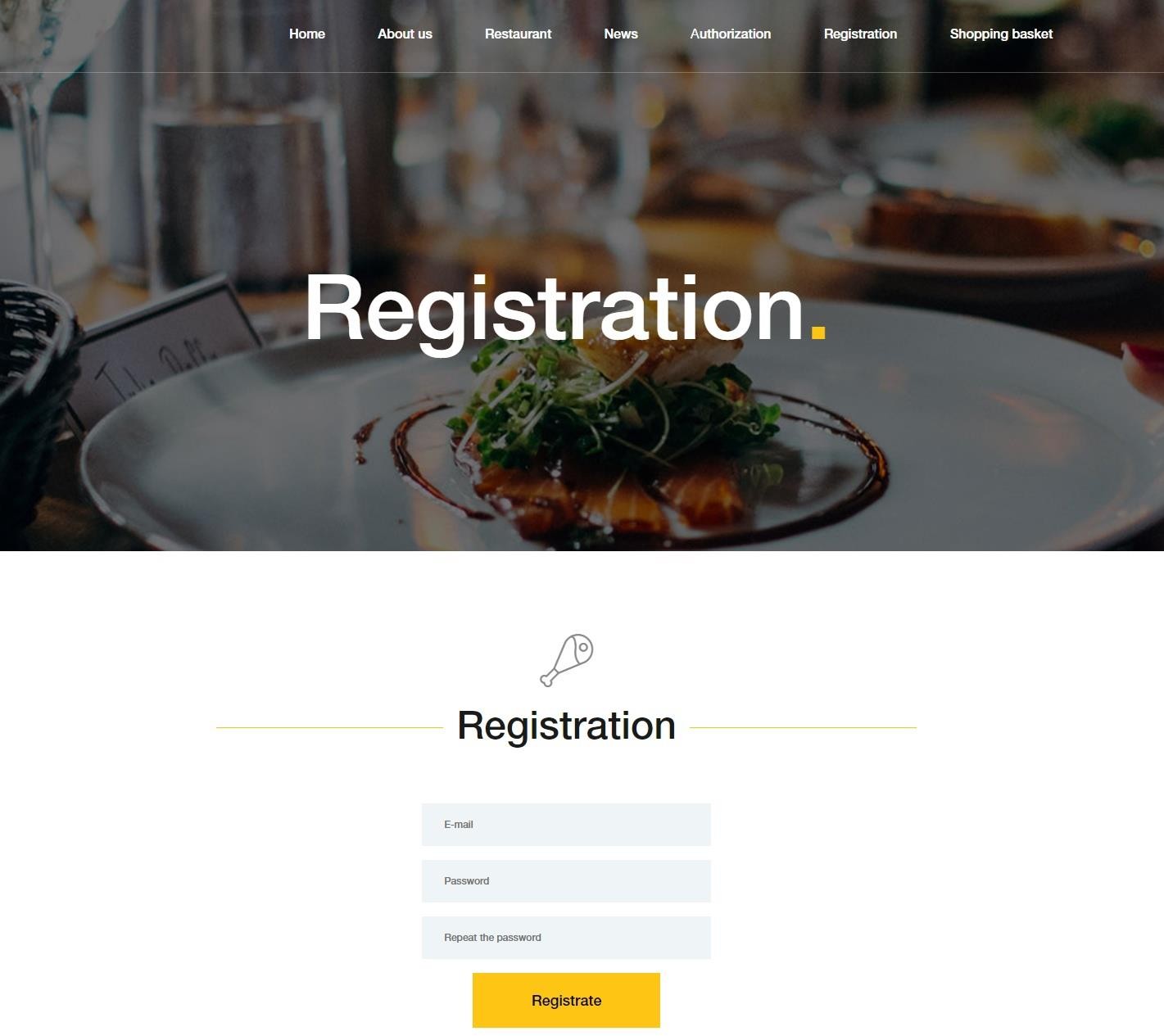
1. Окно регистрации (Рисунок 4.4.1)
2. Главная страница с которой происходит переход на страницу регистрации, входа и меню (Рисунок 4.4.2);
3. Окно входа (Рисунок 4.4.3);
4. Меню (Рисунок 4.4.4)
5. Личный кабинет (Рисунок 4.4.5)

Рисунок 4.4.1 – Окно регистрации

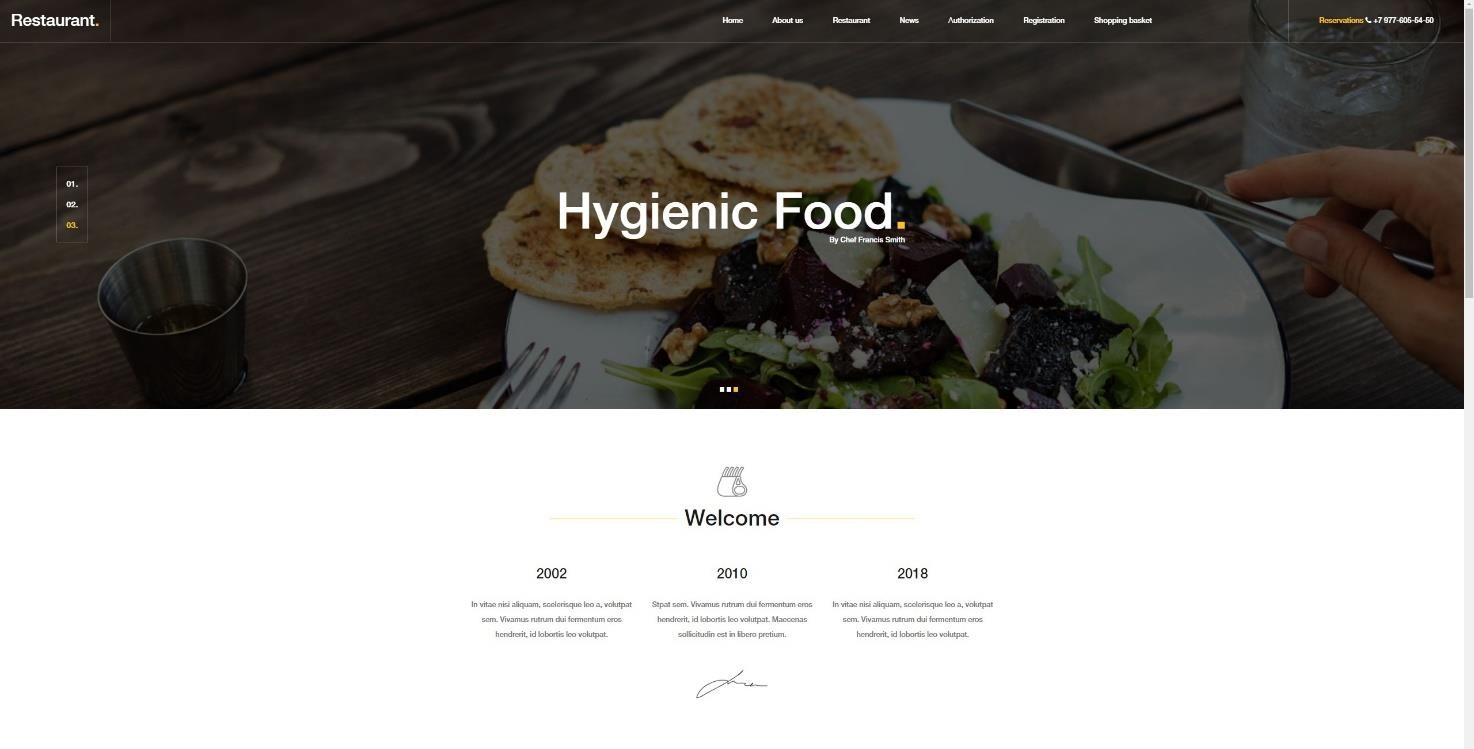


Рисунок 4.4.2 – Главная страница

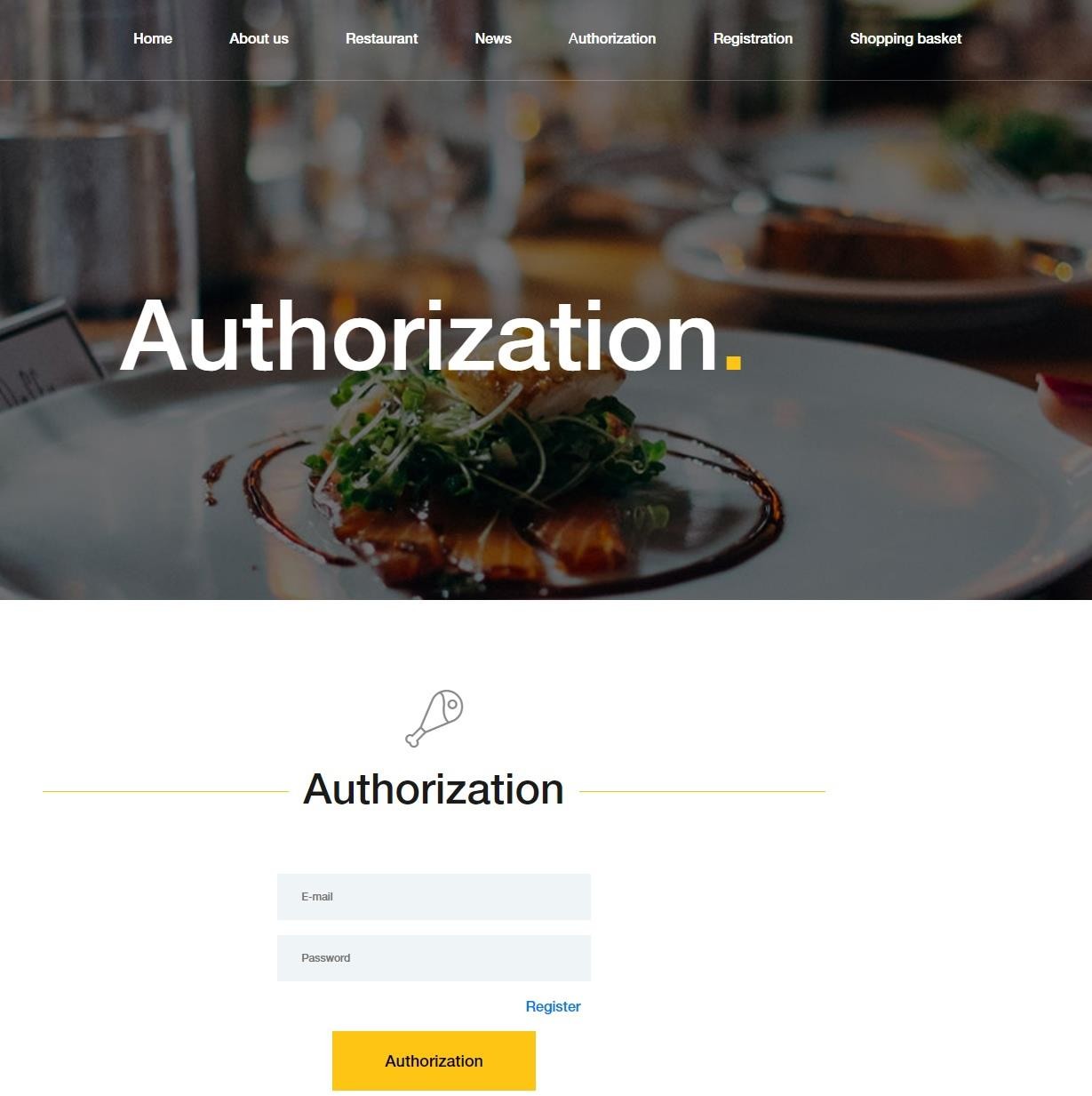


Рисунок 4.4.3 – Окно входа

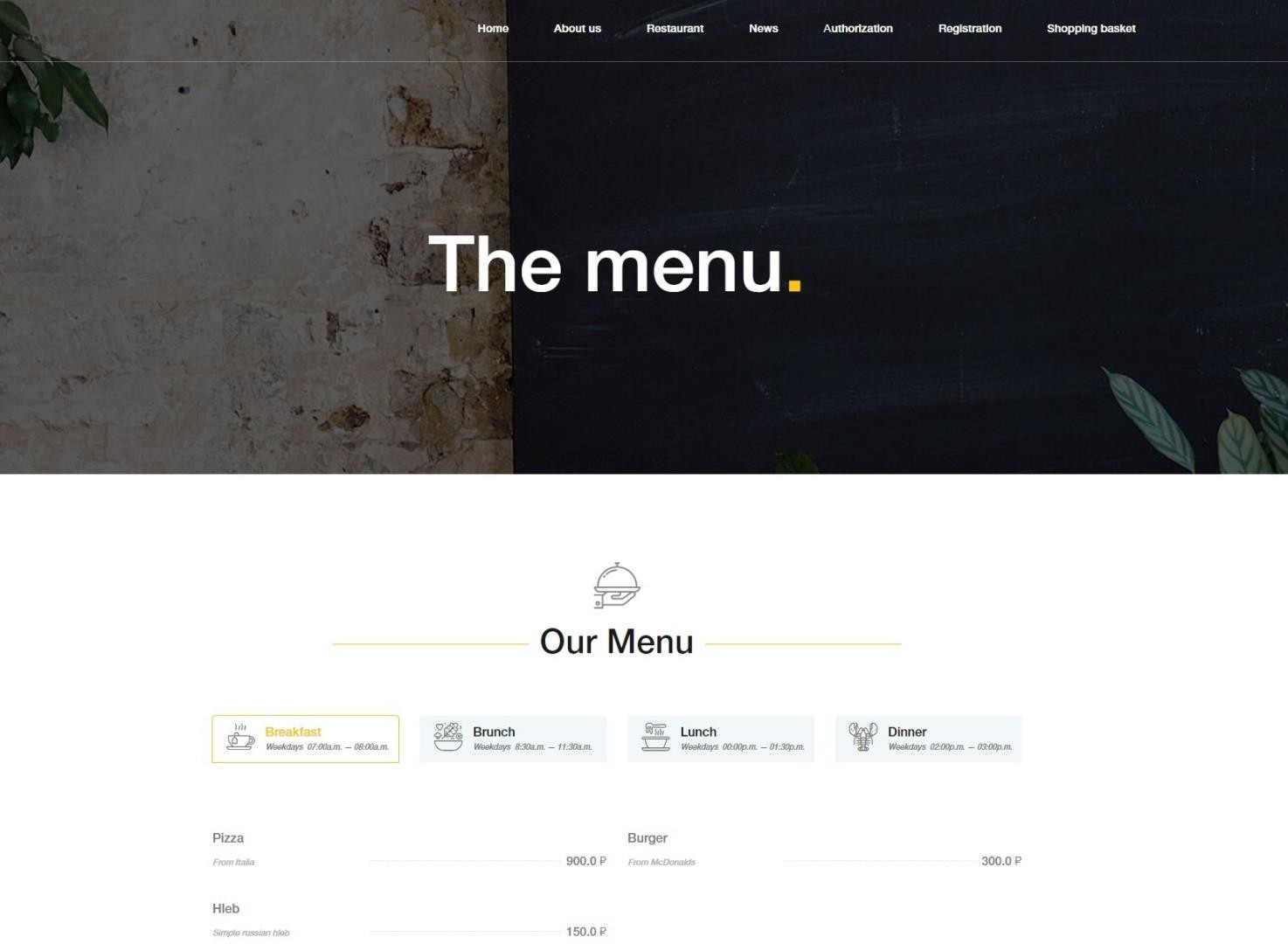


Рисунок 4.4.4 – Страница «Меню»

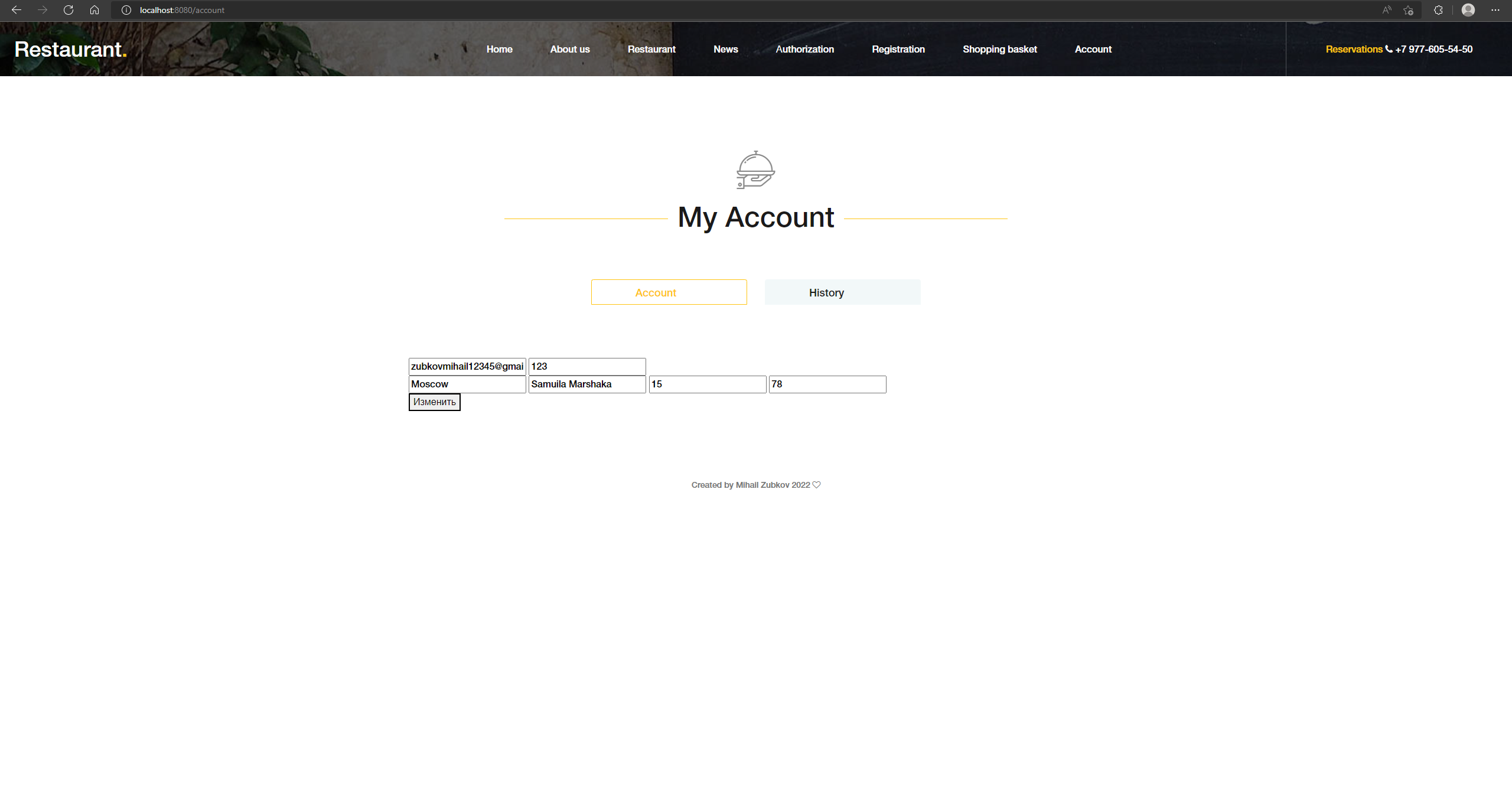


Рисунок 4.4.5 – Страница «Личный кабинет»

## 4.5 Создание сервисов. Services.

Далее были подключены сервисы [5] (Рисунок 4.5.1). Данная часть кода помогает осуществить бизнес-логику серверной части, в ней прописывается та работа, которая не была прописана в контроллерах, а также запуск приложения

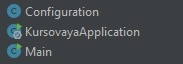


Рисунок 4.5.1 – Сервисы

## 4.6 Диаграмма классов и таблиц серверной части

Ниже представлена диаграмма таблиц базы данных (Рисуник 4.6.1) и диаграмма классов проекта (Рисуник 4.6.2).

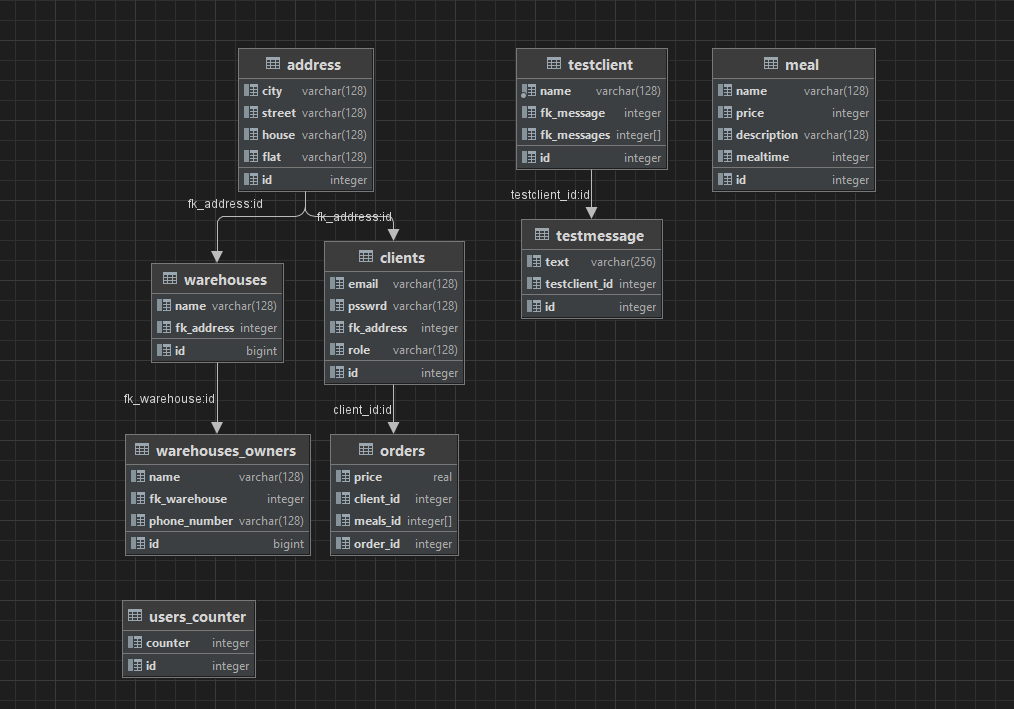


Рисунок 4.6.1 – Диаграмма таблицы базы данных

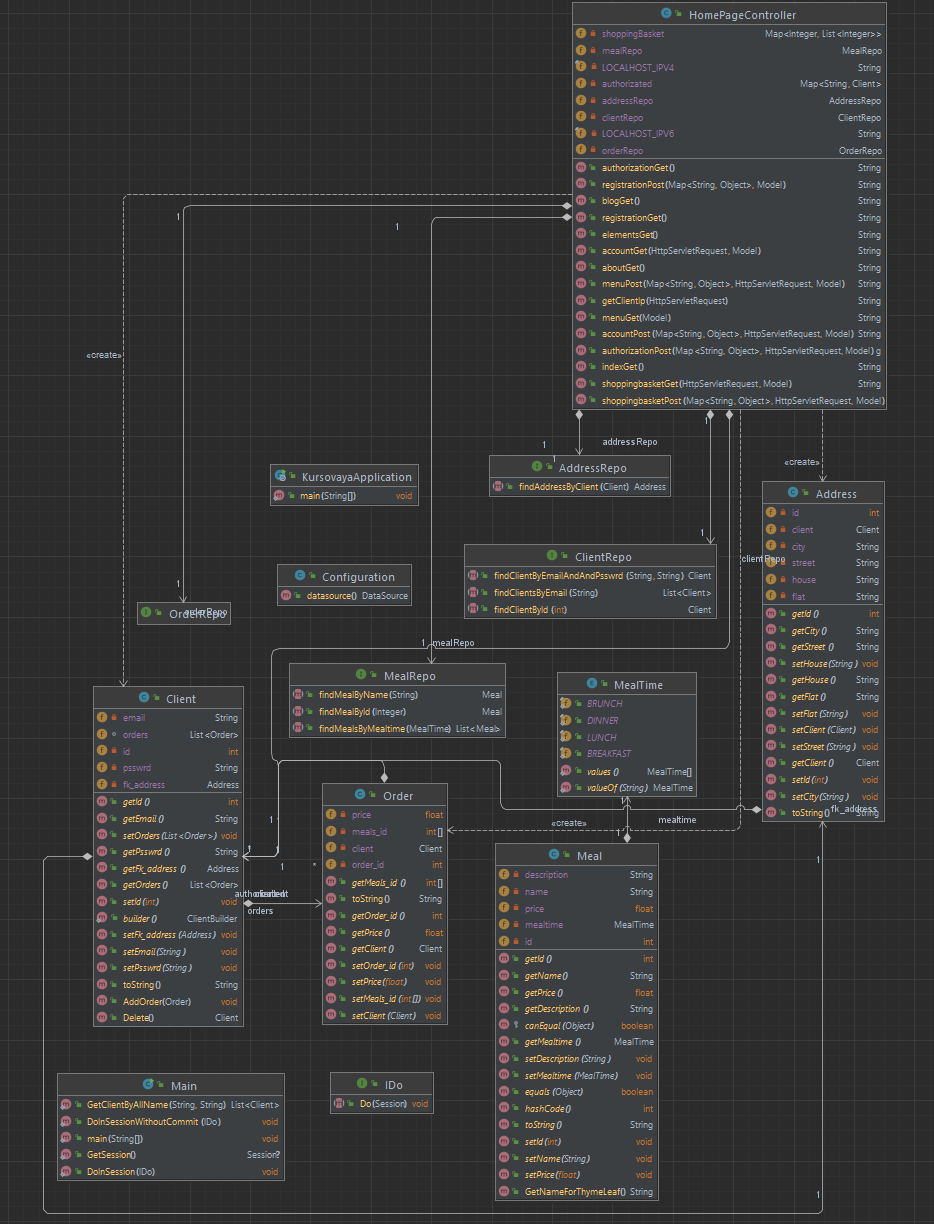


Рисунок 4.6.2 – Диаграмма классов проекта

## ВЫВОДЫ К РАЗДЕЛУ 4

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.

# ЗАКЛЮЧЕНИЕ

В процессе выполнения курсовой работы был проведён анализ предметной области, в ходе которого были рассмотрены различные ресурсы чатов. По результатам анализа были составлены требования к необходимому функционалу для веб-приложения «Ресторан».

В результате выполнения курсовой работы была разработана архитектура веб-приложения на основе выбранного паттерна проектирования, реализован слой серверной логики веб-приложения с применением выбранной технологии, реализован слой логики базы данных, разработан слой клиентского представления веб-приложения.

А также были получены навыки для обеспечения процесса разработки серверных частей интернет-ресурсов для формирования обобщенных трудовых функций и профессиональных компетенций в соответствии с профессиональными стандартами "Архитектор программного обеспечения", "Программист".

# СПИСОК ИСПОЛЬЗОВАННЫХ 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: 10.10.2022)
3. Spring Data JPA [Электронный ресурс] URL: https://spring.io/projects/spring-data-jpa/, свободный (Дата последнего обращения: 15.10.2022)
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: 15.10.2022)
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: 15.10.2022)
6. MVC [Электронный ресурс] URL: <https://habr.com/ru/post/215605/>, свободный (Дата последнего обращения 01.11.2022)
7. Java | Oracle: официальный сайт [Электронный ресурс] URL: https://www.java.com/ru/, (Дата последнего обращения 01.11.2022)
8. Архитектура слоя данных [Электронный ресурс] URL: <https://habr.com/ru/company/surfstudio/blog/655389/> (Дата последнего обращения 01.11.12)
9. Thymeleaf: официальный сайт [Электронный ресурс] URL: https://www.thymeleaf.org (Дата последнего обращения 01.11.2022)
10. Lombok: официальный сайт [Электронный ресурс] URL: <https://projectlombok.org/> (Дата последнего обращения 01.11.2022)
11. Шоколадница: официальный сайт [Электронный ресурс] URL: <https://shoko.ru/> (Дата последнего обращения 15.10.2022)
12. One Price Coffee: официальный сайт [Электронный ресурс] URL: https://onepricecoffee.com/ (Дата последнего обращения 15.10.2022)
13. Burger King: официальный сайт [Электронный ресурс] URL: https://burgerkingrus.ru/ (Дата последнего обращения 15.10.2022)
14. Intellij IDEA: официальный сайт [Электронный ресурс] URL: https://www.jetbrains.com/ru-ru/idea/ (Дата последнего обращения 15.10.2022)
15. Github: официальный сайт [Электронный ресурс] URL: https://github.com/ (Дата последнего обращения 15.10.2022)
16. Maven: официальный сайт [Электронный ресурс] URL: https://maven.apache.org/ (Дата последнего обращения 15.10.2022)
17. Bootstrap: официальный сайт [Электронный ресурс] URL: https://getbootstrap.com/ (Дата последнего обращения 15.10.2022)
18. Hibernate: официальный сайт [Электронный ресурс] URL: https://hibernate.org/ (Дата последнего обращения 15.10.2022)
19. Spring ORM [Электронный ресурс] URL: <https://springframework.net/doc-latest/reference/html/orm.html> (Дата последнего обращения 15.10.2022)
20. Объектно-реляционная база данных [Электронный ресурс] URL: <https://translated.turbopages.org/proxy_u/en-ru.ru.b6a6dbec-639635ff-6c6657a8-74722d776562/https/en.wikipedia.org/wiki/Object-relational> (Дата последнего обращения 15.10.2022)

# ПРИЛОЖЕНИЕ

Ссылка на репозиторий программного кода - https://github.com/s1ngle0f/kursovaya