Международная технологическая школа Tel-Ran

Техническое задание/ Спецификация на разработку приложения: “Онлайн-сервис заказа пиццы вашей мечты "PizzaCraft"

Выпускная квалификационная работа

разработка: студент группы 110723-m-be
Франчук Светлана

преподаватель: Костяной Юрий

2024

Содержание

1. Схема работы приложения

1.1 Назначение приложения

Владелец сети пиццерий принял решение о расширении спектра услуг, предоставляемых клиентам его заведений. В рамках этого решения был проведен анализ и собраны статистические данные (пользовательские истории приведены в Приложении 1), после чего было принято решение о разработке онлайн-приложения для заказа пиццы. Главной целью проекта является повышение качества предоставляемого сервиса, расширение клиентской базы и автоматизация процесса изготовления пиццы в соответствии с индивидуальными пожеланиями клиентов.

1.2 Пользователи приложения

Данное приложение будет использоваться пользователями, с разделением доступа по следующим ролям:

ADMINISTRATOR

1) ведение справочников/таблиц приложения:

- рецепты теста основы для пиццы - - с указанием веса, калорийности и цены в расчете на минимальный размер пиццы;
- ингредиенты для начинки пиццы - с указанием веса, калорийности и цены в расчете на минимальный размер пиццы;
- пиццы - рецепты пицц. Администратор вводит стандартные рецепты пицц, доступные в дальнейшем пользователям для выбора;
- заказы - внесение информации об оплате (в будущем возможно расширение проекта и интеграция с платежной системой);
- пользователи - блокирование пользователю возможности оставлять сообщения сроком на 1 месяц, за недопустимые сообщения
- отзывы - модерация сообщений;

2) получение списка пользователей, по определенным параметрам и выполнение рассылки о проводимых акциях (в будущем возможно расширение системы и интеграция с почтовым клиентом);

MANAGER - сбор статистической информации и ее обработка для:

- дальнейшего улучшения качества и спектра предоставляемых услуг;
- контроля за количеством использованных продуктов при приготовлении пицц;
- планирования работы кухни в зависимости от периодичности заказов и загруженности поваров;

CLIENT :

1) заказ пиццы, используя любой из предложенных вариантов (заказ пиццы, приготовленной по стандартному рецепту; заказ пиццы, приготовленный по индивидуальному рецепту; заказ пиццы, приготовленный по понравившемуся ранее рецепту и добавленному в избранное);
2) добавление оценки и отзывов приложению/ пиццерии.

1.3 Выгоды от разработки и использования приложения

Разработка приложения позволит:

владельцу приложения:

1) расширить круг пользователей (посетителей пиццерии) за счет:
- предоставления возможности сделать заказ онлайн и получить доставку по указанному заказу в указанное время;
- предоставления возможности сделать заказ по индивидуальному рецепту;
- предоставления пользователям возможности накапливать и использовать бонусы;
2) стимулировать пользователей к совершению дополнительного заказа за счет ведения базы пользователей (email, на который направляется информация о дополнительных акциях и скидках);
3) планировать работу кухни и периоды закупки ингредиентов за счет сбора и обработки статистической информации;

клиенту:

1) быстро и удобно осуществлять заказ пиццы онлайн (за счет возможности заказа на стандартный адрес регистрации, на дополнительный адрес доставки, на конкретную дату и время);
2) попробовать пиццу, приготовленную по индивидуальному рецепту, соответствующую допустимому уровню калорий, соответствующую желаемой сумме
3) сохранять понравившиеся рецепты и использовать их повторно для быстрого заказа в следующий раз
4) иметь возможность узнавать оценки и отзывы других посетителей для принятия решений о заказе какого либо рецепта

1.4 Описание схемы работы приложения

Для онлайн-приложения была предложена следующая структурная схема:

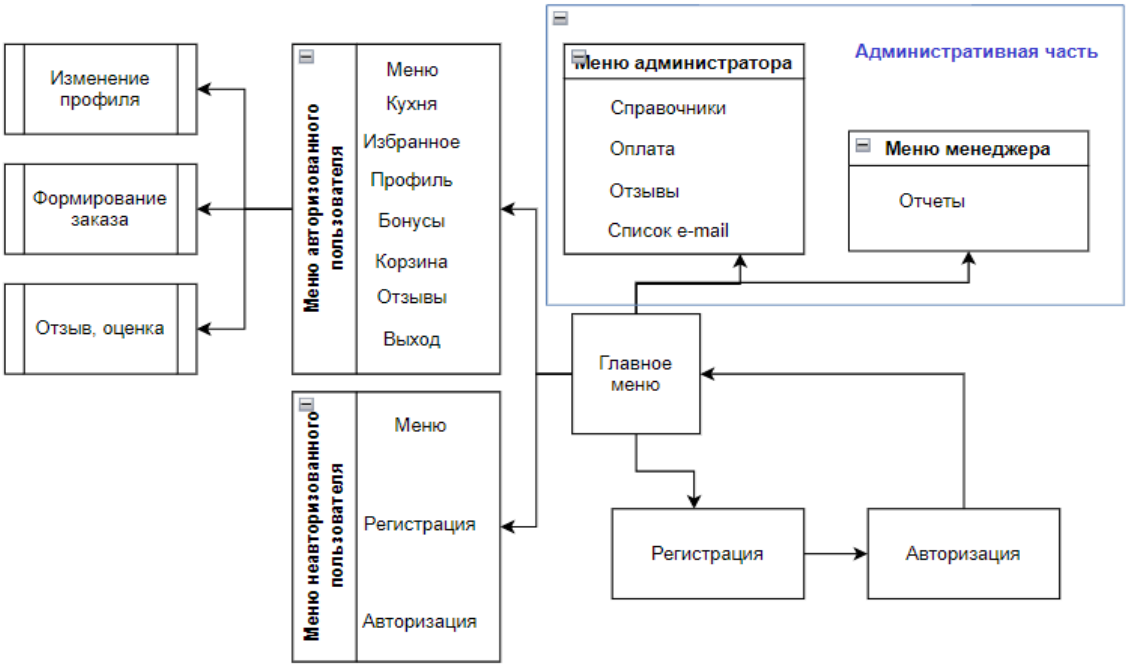


рис.1 Структурная схема

Схема работы клиента приведена на рис.2:

1) Пользователь регистрируется. При регистрации пользователь вводит следующую информацию:

- логин;
- пароль;
- E-mail - для получения в будущем информации о программах и скидках, и для восстановления пароля (при необходимости);
- дату рождения - для получения бонусных скидок при заказе в день рождения;

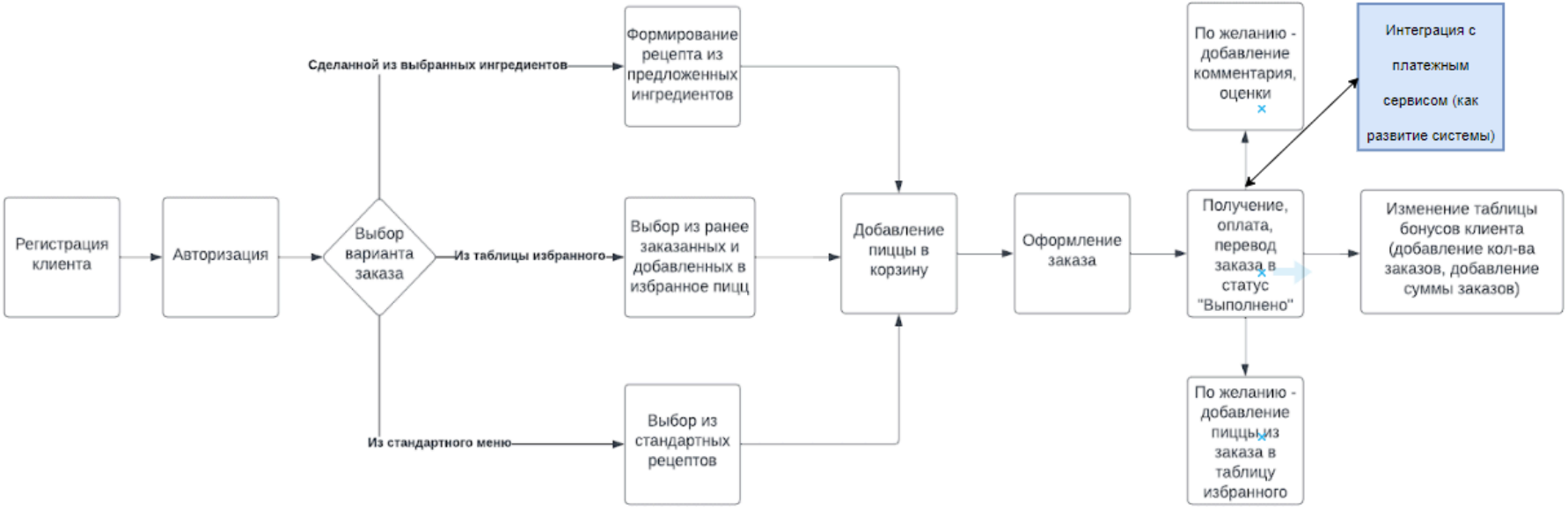


рис.2 Схема работы клиента

- адрес доставки (используется, как адрес доставки по умолчанию), в момент оформления заказа можно при необходимости указать другой адрес;
- контактный телефон для связи
- дополнительно, сохраняется дата регистрации пользователя.

2) В дальнейшем для авторизации используется логин и пароль.

3) С главной страницы авторизованный пользователь имеет доступ к следующим разделам:

- меню (классические рецепты с разбивкой на виды пицц: классическая итальянская, американская, специальная (гриль или фруктовая));
- кухня (раздел, где клиент самостоятельно “готовит” пиццу, выбирая тип приготовления, тесто и ингредиенты);
- избранное (раздел в котором сохранены рецепты ранее заказанных пицц и отмеченных для сохранения в избранное);
- профиль (раздел в котором пользователь может изменить информацию о себе: адрес, телефон или пароль. Остальные поля не доступны для изменения);
- бонусы (раздел в котором видны предлагаемые бонусы, условия для их получения и накопительный текущий статус клиента. Например: для получения бесплатной пиццы нужно сделать заказ на 100 Евро. На данный момент клиент имеет историю с 57 Евро согласно предыдущих заказов);
- корзина (раздел с выбранными пиццами). Если, корзина пользователя содержит список пицц, добавленных ранее и не оформленных в заказ, то после авторизации отображается соответствующее сообщение;
- заказы (раздел с историей заказов клиентов (содержит информацию с датой заказа, суммой и списком пицц). В списке пицц доступна возможность отметки пиццы, как избранного рецепта);
- отзывы клиентов (раздел, где пользователь может просмотреть отзывы клиентов и оценки, которые они оставили. Для пользователя есть возможность оставить отзыв и поставить свою оценку заведению)

4) Меню - страница на которой пользователь может:

- просмотреть все предлагаемые рецепты (в каждом рецепте указано: наименование, описание, калорийность и цена для минимального размера пиццы)
- отфильтровать рецепты по виду пиццы, по наполнению пиццы (мясная, рыбная, овощная, сырная, фруктовая)
- сделать заказ, выбрав понравившийся вариант, указав количество и размер пиццы (по умолчанию 1 шт и размер Small)

5) Кухня - страница на которой клиент осуществляет выбор:

- типа приготовления (классическая, глубокая, на гриле);
- типа теста для основы (сицилийское, неаполитанское, нью-йоркское, из цельнозерновой муки, из кукурузной муки);
- далее набор ингредиентов (соусы, основная начинка (мясо, рыба, колбаса), дополнительная начинка (овощи, грибы, сыры, фрукты, зелень). Наполнение ограничено следующими параметрами: максимум 3 соуса, максимум 4 вида основной начинки, максимум 7 видов дополнительной начинки). Для каждого ингредиента указано: наименование, калорийность и цена для минимального размера пиццы
- затем выбор размера пиццы и указание количества пицц (по умолчанию 1 шт и размер Small)
- клиент должен указать наименование пиццы и по желанию ввести краткое описание. По умолчанию: название (логин клиента+дата): описание - пусто.

Реализация конструктора пиццы позволит: повысить лояльность клиента к ассортименту пиццерии, при этом не потребуется человеческого труда для предоставления и сбора информации о составе пиццы, что сокращает трудозатраты и количество ошибок в заказах

6) Избранное - в данном разделе отображаются все сохраненные пользователем рецепты. Клиент может:
- посмотреть варианты пицц (в каждом рецепте указано: наименование, описание, калорийность и цена для минимального размера пиццы);
- отфильтровать рецепты по виду пиццы, по наполнению пиццы (мясная, рыбная, овощная, сырная, фруктовая);
- сделать заказ, выбрав понравившийся вариант, указав количество и размер пиццы (по умолчанию 1 шт и размер Small)

7) После выбора пицц пользователь может перейти в раздел корзина, просмотреть и при необходимости откорректировать заказ. В корзине отображаются: пицца, количество, размер, калорийность на выбранный размер пиццы, сумма на выбранный размер пиццы и общая сумма заказа. Далее можно приступить к оформлению заказа

8) Раздел оформление содержит:
- перечень пицц;
- общую сумму заказа;
- бонус, если клиент достиг бонусного предела по сумме, количеству заказов, либо у него день рождения в день доставки, либо у заведения проходит какая-либо акция в текущий момент. Списание бонуса происходит автоматически при достижении предела;
- дополнительно информацию в разделе можно дополнить временем и датой доставки. По умолчанию период доставки в пределах ближайшего часа.

- также можно указать адрес доставки, если он отличается от адреса по умолчанию.

Оплата производится за пределами данного приложения. В будущем приложение будет доработано интеграцией с системой оплаты . На данном этапе администратор вносит информацию с фактом оплаты вручную, после получения подтверждения оплаты курьером.

9) После внесения информации об оплате в таблицу бонусов клиента добавляется информация о сумме заказа. Количество заказов клиента также увеличивается.

10) Неавторизованный пользователь может видеть только главную страницу с одним разделом Меню, где он может только просматривать стандартные виды пиццы (наименование, описание, калорийность, цена пиццы минимального размера) и не имеет возможность сделать заказ.

Для пользователя с ролью **администратора** предложена следующая **схема,** рис.3:

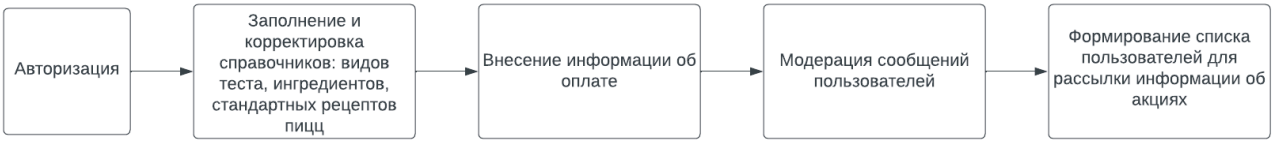


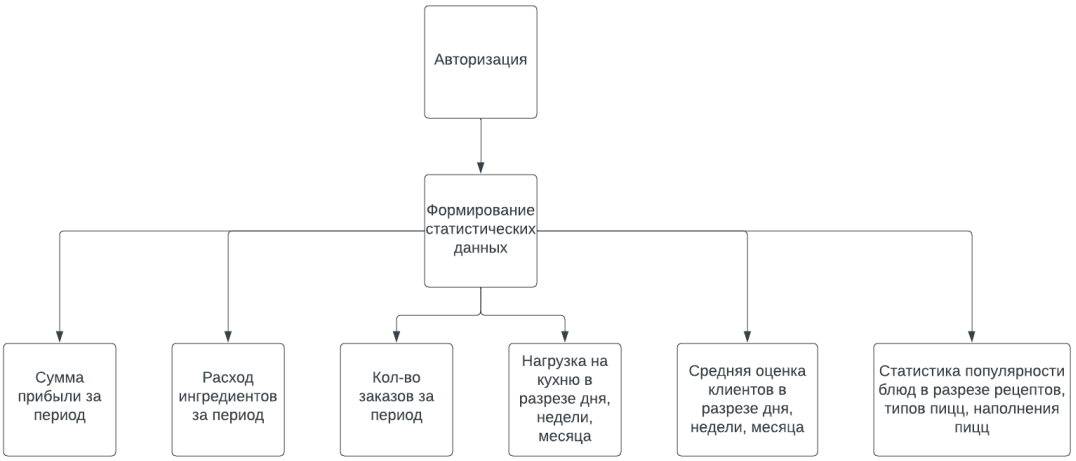
рис.3 - схема работы пользователя с ролью Администратор

Пользователи с ролью Администратор должны быть зарегистрированы в системе и внесены в базу данных. По умолчанию все новые пользователи регистрируются с ролью “Клиент”. Изменить роль на “Администратор” необходимо в Базе данных.

Для администратора доступны следующие пункты меню:

1) Тесто - раздел в котором можно добавлять и корректировать справочник видов теста;
2) Ингредиенты - раздел в котором можно добавлять и корректировать справочник ингредиентов;
3) Пиццы - раздел в котором можно добавлять и корректировать виды пицц
4) Оплата - раздел в котором вносится информация об оплате заказа
5) Чат - раздел в котором отображаются сообщения пользователей. Здесь же есть возможность удаления и блокирования сообщений пользователя на определенный период (по умолчанию на месяц)
6) Статистика - раздел в котором формируется список email адресов пользователей для рассылки информации о проводимых акциях.

Для пользователя с ролью **Менеджера** доступна следующая **схема**:



Пользователи с ролью Менеджер должны быть зарегистрированы в системе и внесены в базу данных. По умолчанию все новые пользователи регистрируются с ролью “Клиент”. Изменить роль на “Менеджер” необходимо непосредственно в Базе данных.

Для пользователя с ролью Менеджер доступен пункт меню статистика, в котором есть возможность сформировать следующие отчеты:

- Отчет о сумме прибыли за период
- Отчет о расходе ингредиентов за период
- Отчет о количестве заказов, через приложение за период
- Отчет об оценке клиентов в разрезе: дня, недели, месяца
- Отчет о статистической популярности блюд в разрезе: рецептов, типов пицц, типов наполнения пицц

2. Доменная модель

2.1 Описание сущностей в структуре приложения

Перечень сущностей доменной модели (структура сущностей описана далее):

- **UserApp** - информация о пользователях (логин, пароль, электронный адрес, адрес, телефон, дата рождения);
- **Role** - enum: ADMIN, MANAGER, CLIENT;
- **Bonus** - накопительная информация о бонусах клиента. Доступны следующие опции 50% скидка при общей накопленной сумме заказов 100 Евро, 30% скидка при заказе 31 пиццы, 100% скидка одновременном заказе на 100 Евро или 10 пицц (количество заказов, сумма заказов) - встроенная в UserApp таблица;
- **Address** - информация об адресе клиента, используется, как адрес доставки по умолчанию (город, наименование улицы, номер дома, номер квартиры) - встроенная в UserApp таблица;
- **ContactInformation** - информация о контакте клиента (номер телефона) встроенная в UserApp таблица;
- **Review** - информация об оценках и отзывах клиентов;
- **Dough** - информация о видах теста, которые используются в пиццерии (тип пиццы, вес, калорийность и сумма для минимального размера пиццы);
- **TypeDough** - enum: CLASSICA, PAN_PIZZA, SICILIAN, NEW_YORK_STYLE, NEAPOLITAN, WHOLE_WHEAT_FLOUR, CORNMEAL;
- **Ingredient** - информация об ингредиентах, которые используются в пиццерии (наименование, вес, калорийность, цена, группа, к которой принадлежит ингредиент: соус, основная начинка, дополнительная начинка);
- **GroupIngredient** - enum: SAUCE, BASIC, EXTRA;
- **Pizza** - информация о рецептах, по которым готовится пицца (наименование, описание, тип пиццы, тип наполнения, размер, признак стандартного рецепта, цена, калорийность);
- **Styles** - enum: CLASSIC_ITALIAN, AMERICAN, SPECIALITY;
- **ToppingsFillings** - enum: MEAT, VEGETABLES, CHEESE, SEAFOOD, FRUIT;
- **TypeBySize** - enum: SMALL(coefficient 1), MEDIUM(coefficient 1.3), LARGE(coefficient 1.7). Коэффициент используется для пересчета количества необходимого теста и ингредиентов, а также для расчета калорийности блюда и его себестоимости;

- **Favorites** - информация об избранных рецептах пользователя;
- **Basket** - информация о содержании корзины пользователя, содержит список пицц, которые пользователь предполагает заказать;
- **OrderDetails** - информация с деталями заказа (дата и время доставки, список пицц, бонус, если накоплен предел);
- **TypeBonus** - enum: DISCOUNT_30, DISCOUNT_50, DISCOUNT_100;
- **Order** - общая информация о заказе (дата заказа, общая сумма заказа, статус заказа, адрес доставки - по умолчанию равен адресу, указанному при регистрации).
- **StatusOrder** - enum: NEW, CANCELED, PAID.

Регистрация пользователя

Для работы в приложении необходима первоначальная регистрация пользователя. В дальнейшем работа в приложении доступна после авторизации. Одному пользователю доступна одна роль. Для обеспечения этого функционала используется сущность **UserApp**:

№ п.п	**Имя**	**Тип**	**Ограничения**	**Описание**
1	id	Bigint	Primary key, Not null, Unique, Auto Increment	Уникальный идентификатор
2	userName	Varchar (25)	size (5-25), Not null, Unique	Имя пользователя
3	password	Varchar(15)	size (5-15), Not null	Пароль
4	email	Varchar (50)	size (5-50), Not null, Unique	email
5	birthdate	Timestamp	Past	Дата рождения
6	date_registration	Timestamp	Present	Дата регистрации, заполняется автоматически
7	city	Varchar (75)	size (0-75)	Встраиваемая структура - Адрес, состоящая из следующих полей: город, название улицы, номер дома и номер квартиры
8	street_name	Varchar (75)	size (0-75)	
9	house_number	Varchar (5)	size (0-5)	
10	apartment_number	Varchar (5)	size (0-5)	

11	phone_number	Varchar (15)	size (5-15)	Номер телефона, встраиваемая таблица - Контактная информация
12	count_orders	integer	min(0)	Встраиваемая структура для расчета бонусов (количество и сумма заказов)
13	sum_orders	decimal(15, 2)	min(0)	
14	is_blocked	Boolean		Блокировка возможности оставлять сообщения в течение месяца с даты последнего сообщения
15	role	Enum		роль (CLIENT, MANAGER, ADMIN)
16	basket_id		Foreign key	Ссылка на таблицу Basket. Связь *один к одному*

Встраиваемая таблица **Address**

№ п.п	Имя	Тип	Ограничения	Описание
1	city	Varchar (75)	size (0-75)	Наименование города
2	street_name	Varchar (75)	size (0-75)	Название улицы
3	house_number	Varchar (5)	size (0-5)	Номер дома
4	apartment_number	Varchar (5)	size (0-5)	Номер квартиры

Встраиваемая таблица **ContactInformation**

№ п.п	Имя	Тип	Ограничения	Описание
1	phone_number	Varchar (15)	size (0-15)	Номер телефона

Административная часть

Этот функционал доступен пользователям с ролью **ADMIN** или **MANAGER**;

- Выполняемые **ADMIN**-ом функции:

1. Наполнение меню стандартными рецептами. В создании рецепта участвуют следующие сущности (все данные вносятся из расчета на минимальный размер пиццы. В случае выбора пользователем другого размера, все умножается на коэффициент пересчета, который находится в справочнике размеров пицц):

Ingredient

№ п.п	Имя	Тип	Ограничения	Описание
1	id	BigInt	Primary key, Not null, Unique, Auto Increment	Уникальный идентификатор
2	name	Varchar (75)	size (5-75), Not null, Unique	Наименование ингредиента
3	weight	integer	min(0), max(200), Not null	Вес ингредиента в пицце, грамм
4	nutrition	integer	min(0), max(600), Not null	Кол-во калорий в пицце
5	price	Decimal(19, 2)	min(0), Not null	Цена ингредиента в пицце
6	group	enum	Not null	Группа (SAUCE, BASIC, EXTRA)

Dough - рецепт теста основы для пиццы. Количественные показатели приведены для пиццы минимального размера. При оформлении заказа и выборе размера пиццы, все показатели пересчитываются согласно коэффициента, указанного в таблице размеров пиццы.

№ п.п	Имя	Тип	Ограничения	Описание
1	id	integer	Primary key, Not null, Unique, Auto Increment	Уникальный идентификатор

№ п.п	Имя	Тип	Ограничения	Описание
2	type_dough	Enum	not null	Типы теста(CLASSICA, PAN_PIZZA, SICILIAN, NEW_YORK_STYLE, NEAPOLITAN, WHOLE_WHEAT_FLOUR, CORNMEAL)
3	small_weight	integer	min(0), max(200), Not null	вес на пиццу минимального размера
4	small_nutrition	integer	min(0), max(600), Not null	кол-во калорий на пиццу минимального размера
5	small_price	Decimal(19, 2)	min(0), Not null	цена теста для пиццы минимального размера

Pizza

№ п.п	Имя	Тип	Ограничения	Описание
1	id	BigInt	Primary key, Not null, Unique, Auto Increment	Уникальный идентификатор
2	title	Varchar (35)	size (5-35), Not null, Unique	Наименование пиццы
3	description	Varchar (255)	size (0-255), Not null	Описание
4	styles	Enum	not null	Вид пиццы: CLASSIC_ITALIAN, AMERICAN, SPECIALITY
5	topping_fillings	Enum	not null	Тип начинки пиццы: MEAT, VEGETABLES, CHEESE, SEAFOOD, VEGETARIAN
6	size	Enum	not null	Размер пиццы:

				SMALL(coefficient 1), MEDIUM(coefficient 1.3), LARGE (coefficient 1.7)
7	ingredient_list		Foreign key	Лист ингредиентов/ Связь с таблицей ингредиентов многие ко многим
8	dough_id		Foreign key	Тесто. связь с таблицей Тесто Многие к одному
9	is_standard_recipe	boolean	not null, default - false	Признак того, что рецепт стандартный. По умолчанию - ложь
10	amount	DOUBLE(19,2)	min(0)	Сумма для реализации пиццы (округленная до ближайшего целого числа(себестоимость ингредиентов+себестоимость теста)*1,6)
11	nutrition	integer	min(0)	суммарная калорийности ингредиентов и теста

Для организации связи многие ко многим между таблицами Pizzas и Ingredients используется служебная таблица pizzas_ingredients

№ п.п	Наименование
1	ingredients_list_id
2	pizza_set_id

2. Внесение информации об оплате - изменение статуса в таблице заказы. Сущность Order будет описана ниже, в разделе функций клиента
3. Формирование списка клиентов. В выборку должны попадать все пользователи из сущности UserApp с ролью CLIENT. Состав полей для выборки:
    - login
    - email

4. Модерация сообщений пользователей, и блокирование пользователям возможности оставлять сообщения за нарушение правил общения на период - месяц с момента последнего общения (поле `is_Blocked` таблицы UserApp).

- выполняемые **MANAGER**-ом функции - формирование по необходимости отчетов, перечисленных ниже. Для формирования отчетов используются сущности **Order** и **OrderDetails**:

Order_details

№ п.п	Имя	Тип	Ограничения	Описание
1	id	BigInt	Primary key, Not null, Unique, Auto Increment	Уникальный идентификатор
2	order_id	BigInt	Foreign key	Ссылка на таблицу Order. Связь Многие к одному.
3	pizzas	ArrayList	Foreign key	Лист пицц. Односторонняя связь Многие к одному
5	quantity	Integer	NotNull, Positive	Количество пицц

Orders

№ п.п	Имя	Тип	Ограничения	Описание
1	id	bigint	Primary key, Not null, Unique, Auto Increment	Уникальный идентификатор
2	order_date_time	Timestamp	Present	Время формирования заказа
3	sum	decimal(19, 2)	min(0)	Общая сумма заказа
4	status_order	Enum	not null	Статус заказа, при необходимости может изменен администратором
5	order_details_id			Ссылка на таблицу Order_details. Связь Один ко многим.
6	user_app_id			Связь с таблицей UserApp - многие к одному
7	city	Varchar (75)	size (0-75)	Встраиваемая структура - Адрес доставки, состоящая из следующих полей: город, название улицы, номер дома и номер квартиры. По умолчанию, если адрес не заполнен, то доставка будет на адрес, указанный в профиле пользователя
8	street_name	Varchar (75)	size (0-75)	
9	house_number	Varchar (5)	size (0-5)	
10	apartment_number	Varchar (5)	size (0-5)	
11	type_bonuses	enum		Тип бонуса(DISCOUNT_30, DISCOUNT_50, DISCOUNT_100)
12	delivery_date_time	Timestamp	Future	Дата и время доставки

Встраиваемая таблица **DeliveryAddress**

№ п.п	Имя	Тип	Ограничения	Описание
1	city	Varchar (75)	size (0-75)	Наименование города
2	street_name	Varchar (75)	size (0-75)	Название улицы
3	house_number	Varchar (5)	size (0-5)	Номер дома
4	apartment_number	Varchar (5)	size (0-5)	Номер квартиры

Для организации связи многие ко одному между таблицами Users и orders используется служебная таблица users_orders

№ п.п	Наименование
1	user_app_id
2	orders_id

Перечень отчетов:

1) Отчет о сумме прибыли за период. Выборка заказов по полю orderDateTime таблицы Orders. Состав отчета: Дата, Сумма оплаченных пицц, сумма затраченных ингредиентов;
2) Отчет о расходе ингредиентов за период. Выборка из объединения таблиц OrderDetails, Pizza, Ingredient информации, ограниченной периодом дата начала и дата конца (поле `delivery_date_time`) . Состав отчета информация сгруппированная по ID ингредиента: ID ингредиента, наименование ингредиента, суммарный вес;
3) Отчет о количестве заказов, через приложение за период. Выборка из таблицы Order информации, ограниченной периодом дата начала и дата конца (поле `orderDateTime`) . Состав отчета: количество ID заказов за заданный период;
4) Отчет об оценке клиентов в заданном периоде. Выборка из таблицы Review информации, ограниченной периодом дата начала и дата конца (поле `review_date`) . Состав отчета: средняя оценка;
5) Отчет о статистической популярности блюд в заданном периоде в разрезе: рецептов, типов пицц, типов наполнения пицц. Выборка из объединения таблиц OrderDetails и Pizza информации,

ограниченной периодом дата начала и дата конца (поле `delivery_date_time`). Состав отчета - информация сгруппированная по ID пиццы: ID пиццы, наименование пиццы, тип пиццы, тип наполнения пиццы, количество заказов.

Клиентская часть

1) Страница Меню. Пользователю отображается выборка из таблицы Pizza всех рецептов с признаком `is_standard_recipe = true`
   Просматривая страницу Пользователь может выбрать пиццу, и ее размер. Информация попадает в корзину. Содержимое корзины отражает сущность **Basket**

№ п.п	Имя	Тип	Ограничения	Описание
1	id	Bigint	Primary key, Not null, Unique, Auto Increment	Уникальный идентификатор
2	id_user	Bigint	Foreign key	Ссылка на таблицу UserApp. Связь Один к одному.
3	pizzas	ArrayList	Foreign key	Лист пицц. Односторонняя связь Один ко многим

Для организации связи один ко многим между таблицами Pizzas и Basket используется служебная таблица baskets_pizzas

№ п.п	Наименование
1	basket_id
2	pizzas_id

2) Страница Кухня. Раздел, в котором пользователь последовательно выбирает:
   - тип приготовления Enum Styles;
   - типа теста для основы Enum TypeDough;
   - ингредиенты из таблицы Ingredients, сгруппированные по полю `group_ingredient` (соусы, основная начинка , дополнительная начинка. Для каждого ингредиента указано: наименование, калорийность и цена для минимального размера пиццы

- затем выбор размера пиццы Enum TypeBySize
- в завершении клиент должен указать наименование пиццы и по желанию ввести краткое описание. По умолчанию: название (логин клиента+дата): описание - пусто.

По завершению формирования рецепта, пицца попадает в корзину

3) Страница Избранное. Если пользователь отметил ранее какие-либо рецепты, как понравившиеся, то на данной странице будут отображены данные из таблицы **Favorites**

№ п.п	Имя	Тип	Ограничения	Описание
1	id	bigint	Primary key, Not null, Unique, Auto Increment	Уникальный идентификатор
2	id_user	bigint	Foreign key	Ссылка на таблицу UserApp. Связь Один к одному.
3	pizzas	ArrayList	Foreign key	Лист пицц. Односторонняя связь Многие ко многим

Для организации связи многие ко многим между таблицами Pizzas и Favorites используется служебная таблица pizzas_favorites

№ п.п	Наименование
1	pizzas_id
2	favorites_id

Пользователь может выбрать размер пиццы и количество пицц. После чего заказ попадает в корзину.

4) Страница Корзина. На данной странице отображаются:

- все пиццы, которые клиент отметил, как те, которые он хочет заказать;
- количество пицц, размер, калорийность и сумма.

После проверки заказа пользователь может перейти к оформлению заказа. Если к моменту оформления заказа пользователь накопил достаточное количество бонусов, то во время оформления заказа происходит автоматическое использование бонуса. Информация о бонусах хранится в таблице **Bonuses.** Данная таблица является встраиваемой в таблицу UserApp:

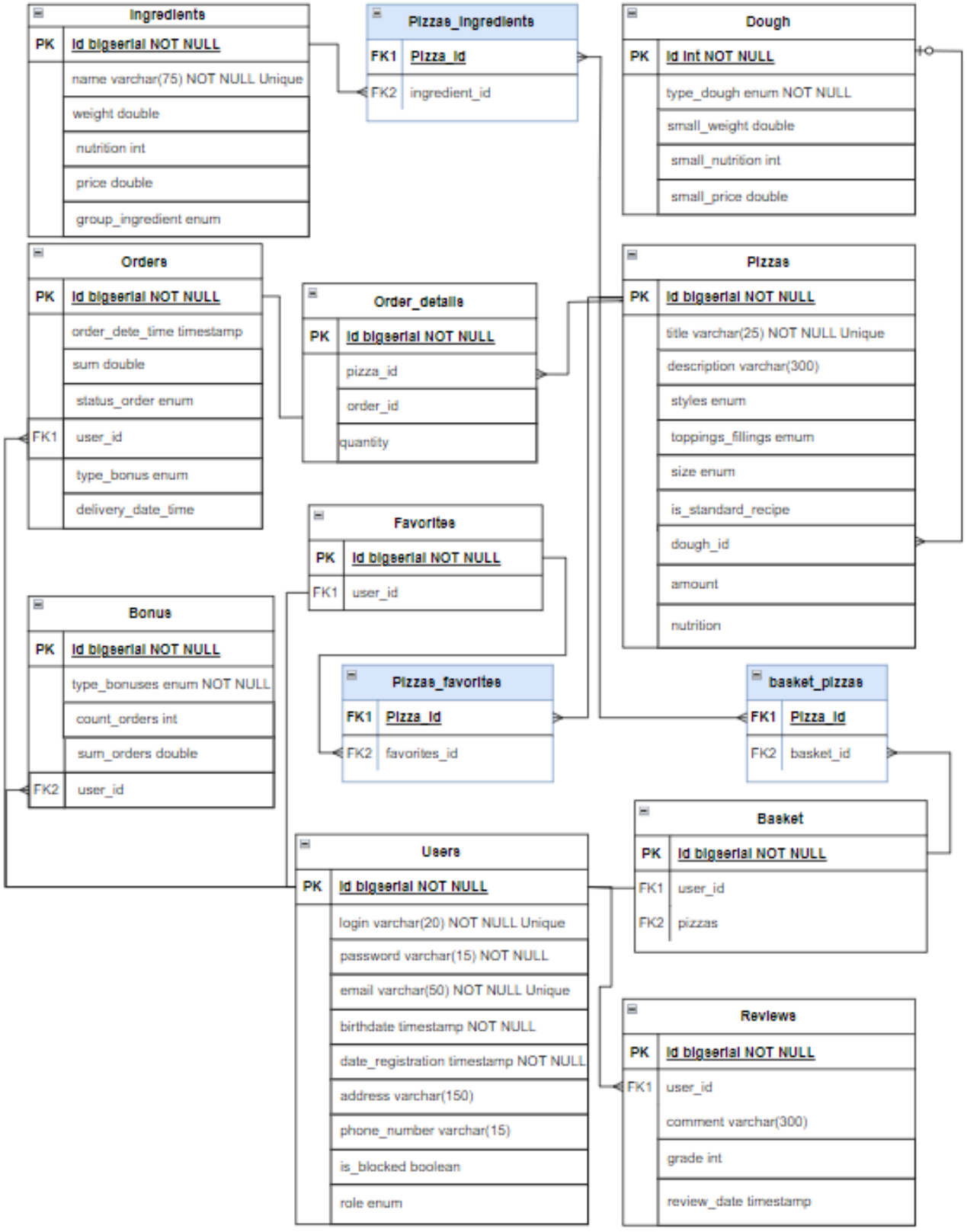
№ п.п	Имя	Тип	Ограничения	Описание
1	count_orders	integer	min(0)	Количество заказов. При использовании бонуса, связанного с этим полем, количество обнуляется
2	sum_orders	decimal(19, 2)	min(0)	Сумма заказанных пицц. При использовании бонуса, связанного с этим полем, сумма обнуляется

5) Страница Профиль - на этой странице пользователь может изменить свои данные: адрес, телефон или пароль. Изменения происходят в таблице UserApp

6) Страница Отзывы. Информация об отзывах и оценках пользователей содержится в таблице **Reviews**

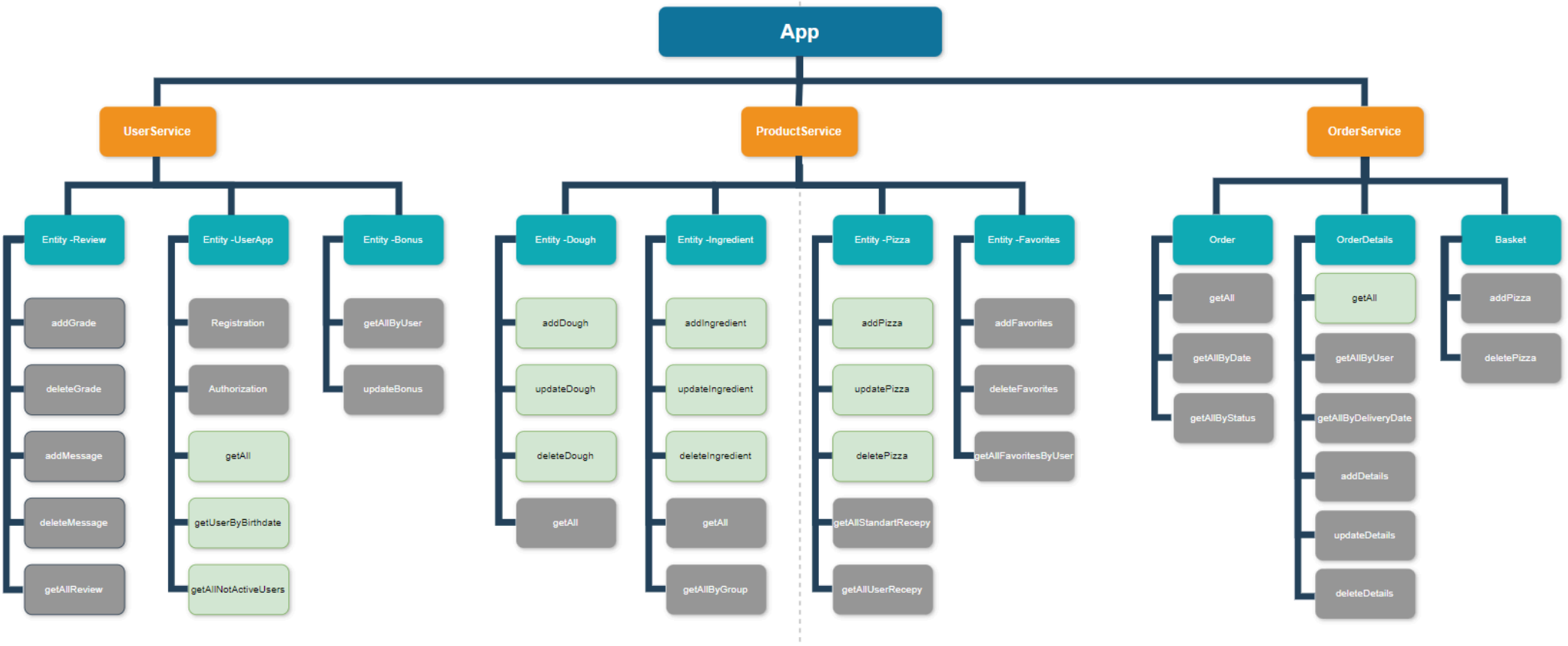
№ п.п	Имя	Тип	Ограничения	Описание
1	id	BigInt	Primary key, Not null, Unique, Auto Increment	Уникальный идентификатор
2	comment	varchar(255)	size(0-255)	Комментарий
3	grade	integer	min(0), max(10)	оценка
4	review_date	Timestamp	not null	Дата внесения информации в таблицу
5	user_app_id			Связь с таблицей UserApp - многие к одному

Диаграмма связи сущностей проекта приведена на схеме ниже.



* файл *Диаграмма связи сущностей (4).drawio* находится в папке главной ветки проекта

Схема доменной модели



* файл *Доменная модель (1).drawio* находится в папке главной ветки проекта.
В схеме приведены основные функции сервисов. Более подробное описание представлено ниже:

2.2 Описание сервисов

Для реализации алгоритмов приложения используются следующие сервисы:

- **AuthenticationService** - через данный сервис реализуется функция регистрации, авторизации и обновления информации о пользователе;

- **UserService** - через данный сервис реализуются функции:
    1. сохранение пользователя;
    2. получение пользователя по имени;
    3. получение текущего пользователя;
    4. получения информации о пользователе (профиль пользователя);
    5. изменение информации в профиле пользователя;
    6. получение списка пользователей с датой рождения в заданную дату;
    7. получение полного списка пользователей с ролью CLIENT;
    8. получение списка пользователей со статусом isBlocked;
    9. изменение статуса пользователя;
    10. получение списка бонусов клиента;
    11. изменение списка бонусов клиента;

- **ReviewService**- через данный сервис реализуются функции:
    1. добавление оценки/отзыва клиента;
    2. удаление оценки/отзыва клиента;
    3. изменение оценки/отзыва клиента;
    4. получение списка всех отзывов;
    5. получение списка отзывов за определенный период;
    6. получение списка отзывов пользователя.

- **DoughService** - через данный сервис реализуются функции:
    1. добавления нового рецепта теста;
    2. изменения параметров существующего рецепта теста;
    3. удаление существующего рецепта теста;
    4. получения списка всех существующих рецептов теста для администратора (с ценой себестоимости);
    5. получения списка возможных вариантов основы для теста для клиента (без цены);

- **IngredientService** - через данный сервис реализуются функции:

1. добавление нового ингредиента;
2. изменение параметров существующего ингредиента;
3. удаление существующего ингредиента;
4. получение полного списка ингредиентов (для Администратора);
5. получение списка ингредиентов в разрезе групп;
6. получение списка ингредиентов в разрезе рецепта пиццы;

- **PizzaService** - через данный сервис реализуются функции:
  1. добавление нового рецепта пиццы (при этом происходит автоматический расчет стоимости пиццы для реализации, равный себестоимости входящих ингредиентов и теста умноженной на 1,6 и количество калорий);
  2. изменение рецепта пиццы;
  3. удаление рецепта пиццы;
  4. получение списка стандартных рецептов пиццы;
  5. получение списка стандартных рецептов пицц в разрезе типов приготовления;
  6. получение списка стандартных рецептов пицц в разрезе типов наполнения;
  7. получение списка стандартных рецептов пицц в разрезе типов приготовления и типов наполнения;

- **FavoritesService** - через данный сервис реализуются функции:
  1. добавление рецепта в таблицу избранного;
  2. удаление рецепта из таблицы избранного;
  3. получение списка избранных рецептов пользователя;
  4. получение пиццы по ID

- **OrderService** - через данный сервис реализуются функции:
  1. формирования корзины пользователя - добавление пиццы в корзину;
  2. получение информации о корзине пользователя;
  3. изменение пиццы в корзине;
  4. удаление пиццы из корзины;
  5. перемещение информации из корзины в заказ (создание заказа);
  6. удаление заказа;
  7. изменение заказа;
  8. изменение статуса заказа;
  9. получение списка всех заказов пользователя;
  10. получение информации о заказе пользователя;
  11. получение списка всех заказов в разрезе статусов;
  12. получение списка всех заказов за период.

- **StatisticService** - через данный сервис реализуются функции:
    1. формирования информации о прибыли (сумма оплаченных пицц и сумма ингредиентов в этих пиццах);
    2. формирования информации о количестве использованных ингредиентов;
    3. формирование информации о количестве заказов за период;
    4. формирование информации о средней оценке за период;
    5. формирование информации о популярности пицц.

3. Хранение данных

Для хранения информации используется реляционная база данных Pizzeria.

Стандартные CRUD-методы реализованы в JPA-репозитории, связанном с определенной таблицей в базе данных. Ниже приведен перечень репозиториев и дополнительные методы, используемые в классах :

- **UserAppRepository** функции:
  READ - вернуть пользователя по логину, вернуть список пользователей по дате рождения, вернуть пользователя по логину и email, вернуть список заблокированных пользователей, вернуть список пользователей по роли;

- **DoughRepository**
  READ - вернуть тесто по типу теста и цене;

- **IngredientRepository** функции:
  READ:
  - чтение списка ингредиентов по группе принадлежности,
  - поиск ингредиента по имени и цене;

- **PizzaRepository** функции:
  READ:
  - вернуть список пицц в разрезе типов приготовления,
  - вернуть список стандартных пицц,
  - найти пиццу по имени,
  - вернуть список пицц в разрезе типов наполнения,
  - вернуть список пицц в разрезе типов приготовления и типов наполнения,
  - вернуть список рецептов пицц, использующих определенный вид теста,
  - вернуть список пицц, в составе которых есть определенный ингредиент

- **BasketRepository** функции:
  READ - чтение содержания корзины по ИД пользователя;

- **OrderRepository** функции:
  READ - возврат списка всех заказов пользователя, возврат списка заказов по статусу, возврат списка заказов за указанный период дат, возврат информации по заказу пользователя;

- **OrderDetailsRepository** функции:
  READ - возврат информации о деталях заказа по заказу, возврат списка заказов в которых присутствует определенная пицца;

- **FavoritesRepository** функции:
  READ - возврат списка избранного для пользователя;

- **ReviewRepository** функции:
  READ - получение информации об оценках и отзывах пользователей; получение списка отзывов за период;

4. Обмен данными

Для реализации данной функциональности необходимо создать следующие DTO:

1. UserRegisterRequestDTO (для регистрации пользователя):
- String userName(логин пользователя);
- String password (пароль пользователя);
- String email (e-mail пользователя);
- LocalDate birthday (дата рождения пользователя);
- String addressCity (город доставки по умолчанию);
- String addressStreetName (улица доставки по умолчанию);
- String addressHouseNumber (номер дома доставки);
- String addressApartmentNumber (номер квартиры доставки);
- String phoneNumber (контактный телефон);

2. UseRequestDTO (для изменения профиля пользователя):
- String userName(логин пользователя);
- String password (пароль пользователя);
- String email (e-mail пользователя);
- LocalDate birthday (дата рождения пользователя);
- String addressCity (город доставки по умолчанию);
- String addressStreetName (улица доставки по умолчанию);
- String addressHouseNumber (номер дома доставки);
- String addressApartmentNumber (номер квартиры доставки);
- String phoneNumber (контактный телефон);

3. UserResponseDTO:
- String userName(логин пользователя);
- String email (e-mail пользователя);
- LocalDate birthday (дата рождения пользователя);
- Address address (адрес доставки по умолчанию);
- String phoneNumber (контактный телефон);

4. UserBonusDTO:
- Long userId (идентификатор пользователя);
- int countOders (количество накопленных заказов, для будущего списания);
- double sumOrders (количество накопленной суммы заказов, для будущего списания);

5. UserLoginFormRequestDTO:
- userName (имя пользователя);

- password (пароль пользователя);

6. UserBlockedresponseDto:
- id - идентификатор пользователя;
- userName - имя пользователя;
- isBloked - признак заблокированности;
- reviewDate - дата последнего отзыва;

7. DoughCreateRequestDto (для создания строки справочника):
- Enum typeDough(тип теста);
- int weight (вес теста);
- int nutrition (калорийность теста);
- int price (цена теста);

8. DoughUpdateRequestDto (для обновления строки справочника):
- int weight (вес теста);
- int nutrition (калорийность теста);

9. DoughResponseDto (для ведения справочников):
- int id (идентификатор теста);
- Enum typeDough(тип теста);
- int weight (вес теста);
- int nutrition (калорийность теста);
- int price (цена теста);

10. DoughResponseClientDto (для отображения клиенту):
- int id (идентификатор теста);
- Enum typeDough(тип теста);
- int weight (вес теста);
- int nutrition (калорийность теста);

11. IngredientRequestDto (для ведения справочников):
- String name(наименование ингредиента)
- int weight (вес ингредиента)
- int nutrition (калорийность)
- int price (цена)
- groupIngredient (группа ингредиентов)

12. IngredientResponseDto (для ведения справочников):
- Long id (идентификатор ингредиента);
- String name(наименование ингредиента)
- int weight (вес ингредиента)
- int nutrition (калорийность)
- int price (цена)

- groupIngredient (группа ингредиентов)

13. IngredientResponseClientDto (для отображения клиенту):
- Long id (идентификатор ингредиента);
- String name(наименование ингредиента)
- int weight (вес ингредиента)
- int nutrition (калорийность)
- groupIngredient (группа ингредиентов)

14. PizzaRequestDto: (для ввода рецептов пиццы)
- String title(наименование пиццы)
- String description (описание пиццы)
- Enum styles (тип приготовления)
- Enum topingsFillings (тип наполнения)
- Enum size (размер)
- String Dough (тесто)
- List<IngredientResponseClientDto > (перечень соусов)
- List<IngredientResponseClientDto > (перечень основной начинки)
- List<IngredientResponseClientDto > (перечень дополнительной начинки)

15. PizzaResponseDto
 String title(наименование пиццы)
- String description (описание пиццы)
- Enum styles (тип приготовления)
- Enum topingsFillings (тип наполнения)
- Enum size (размер)
- DoughResponseClientDto dough (тесто основа)
- List<IngredientResponseClientDto > ingredientList (перечень ингредиентов)
- int nutrition (калорийность)
- double amount (цена)

16. PizzaToBasketrequestDto:
- pizzaId - ИД пиццы;
- countPizza - поличество пицц;

17. BasketRequestDto:
- Long userAppID - идентификатор пользователя
- List<PizzaResponseDto> лист пицц в корзине пользователя

18. BasketResponseDto:
- List<PizzaResponseDto> лист пицц в корзине пользователя

19. OrderRequestDto:
-String deliveryCity,

- String streetName,
- String houseNumber,
- String apartmentNumber,
- OrderDetailsResponseDto orderDetails,
- Long userId

20. OrderResponseDto:
- DeliveryAddress deliveryAddress,
- OrderDetailsResponseDto orderDetails,
- Long userId

21. OrderStatusResponseDto:
- id - идентификатор заказа;
- sum - сумма заказа;
- statusOrder - статус заказа;
- orderDateTime - дата заказа;
- userAppId - идентификатор пользователя

22. FavoritesResponseDto:
- List<PizzaResponseDto> pizzas (пиццы)

23. ReviewRequesDto:
- String comment (комментарий)
- Integer grade (оценка)
- Long UserId (идентификатор пользователя);

24. ReviewResponseDto:
- String comment (комментарий)
- Integer grade (оценка)
- String userName (имя пользователя);
- LocalDateTime reviewDate (дата добавления отзыва)

Также необходимо создать соответствующие контроллеры для обработки запросов от пользователя:

AuthUserController:
- registerUser (UserRegisterRequestDto userRegisterRequestDto) - метод регистрации пользователя;
- authentication(UserLoginFormRequestDto userLoginFormRequestDto) - метод аутентификации пользователя;
- updateUser(Long id, UserRequestDTO userRequestDTO) - метод для изменения информации о пользователе

UserController:

- getUserById (Long id) - получение пользователя по идентификатору;
- getUserBonus(Long id) - метод для получения списка бонусов клиента
- updateUserBonus(Long id, int count, double sum) - метод для изменения списка бонусов клиента;

ReviewController:

- addReview(ReviewRequestDTO reviewRequestDTO) - метод для добавления оценки/отзыва клиента;
- updateReview(Long ig, Long userId, ReviewRequestDto reviewRequestDto) - обновление отзыва пользователя;
- delete Reviews(Long id) - метод для удаления оценки/отзыва клиента;
- getAllReviews() - метод для получения списка оценок/отзывов клиента;
- getAllReviewByUser(Long id) - метод для получения всех отзывов пользователя;
- getAllReviewByPeriod(LocalDateTime startDate, LocalDateTime endDate) - метод для получения всех отзывов за период

UserControllerAdmin:

- getUserByBirthdate(LocalDate date) - метод для получения списка пользователей с датой рождения в заданную дату;
- getUsersByClientRole () - метод для получения полного списка пользователей с ролью CLIENT
- getUserBlocked() - метод для получения списка пользователей со статусом isBlocked и датой последнего сообщения
- changeBlockingUser(Long id, Bolean isBlocked) - изменение статуса блокировки пользователя

DoughController:

- addDough (DoughCreateRequestDto) - метод для добавления нового рецепта теста;
- updateDough (DoughUpdateRequestDto, Id) - метод для изменения параметров существующего рецепта теста;
- deleteDough (id) - метод для удаления существующего рецепта теста;
- getAllDoughForAdmin() - метод для получения списка всех существующих рецептов теста (для администратора с ценой себестоимости);
- getAllDoughForClient() - метод для получения списка возможных вариантов теста-основы для пиццы;

IngredientController:

- addIngredient (IngredientDTO ingredientDTO) - метод для добавление нового ингредиента;
- udateIngredient (IngredientDTO ingredientDTO) - метод для изменения параметров существующего ингредиента;

- deleteIndredient (IngredientDTO ingredientDTO) - метод для удаления существующего ингредиента;
- getAllIngredientForAdmit() - метод для получения полного списка ингредиентов;
- getIngredientByGroup (GroupIngredient group) - метод для получения списка ингредиентов в разрезе групп;
- getIngredientForPizza (Long idPizza) - метод для получения списка ингредиентов в разрезе рецепта пиццы;

PizzaController:

- addPizza (PizzaRequestDto pizzaRequestDto) - метод для добавления нового рецепта пиццы;
- updatePizza (PizzaDTORequest pizzaDTORequest, Long id) - метод для изменения рецепта пиццы;
- deletePizzaRecepy (Long id) - метод для удаления рецепта пиццы;
- getAllPizzaStandardRecepy() - метод для получения списка стандартных рецептов пиццы;
- getPizzaStandardRecepyByStyle (Style style) - метод для получения списка стандартных рецептов пицц в разрезе типов приготовления;
- getPizzaStandardRecepyByTopping (ToppingFilling toppingFilling) - метод для получения списка стандартных рецептов пицц в разрезе типов наполнения;
- getPizzaStandardRecepyByStyleAndTopping (Style style, ToppingFilling toppingFilling) - метод для получения списка стандартных рецептов пицц в разрезе типов приготовления и типов наполнения;

FavoritesController:

- addPizzaToUserFavorite (Long userId, Long pizzaId)- метод для добавления рецепта в таблицу избранного;
- deletePizzaFromUserFavorite (Long pizzaId, Long userId) - метод для удаления рецепта из таблицы избранного;
- getAllFavoritePizzaByUser (Long userId) - метод для получения списка избранных рецептов пользователя
- getPizza(id) - метод получения информации по пицце

OrderController

- addPizzaToBasket(Long userId, PizzaToBasketrequestDto pizzaToBasketrequestDto) формирования корзины пользователя - метод для добавления пиццы в корзину;
- getBasketByUser(Long id) - метод для получения информации о состоянии корцины пользователя;
- changePizzasInBasket(BasketRequestDto basketRequestDto) - изменение пиццы в корзине;

- moveDetailsBasketToOrder(Long id) - перемещение информации из корзины в заказ;
- updateOrderAndOrderDetails(Long id, OrderRequestDto orderRequestDto) - обновление информации по заказу
- deleteOrder(Long id) - метод для удаления заказа;
- updateOrderStatus(Long id, StatusOrder status) - обновить статус заказа;
- getAllOrdersByUser(Long id) - метод для получения списка всех заказов;
- getOrderById(Long orderId) - получение информации о заказе по идентификатору;
- getOrdersStatus(StatusOrder status) - метод для получения списка всех заказов в разрезе статусов;
- getOrdersByPeriod(LocalDate startDate, LocalDate endDate) - метод для получения списка всех заказов в разрезе дат.

StatisticController:

- getProfitInformation (LocalDateTime startDate, LocalDateTime endDate) - получение информации о прибыли;
- getIngredientConsumptionInfo (LocalDateTime startDate, LocalDateTime endDate) - получение информации о расходе ингредиентов;
- getCountOrdersInfo (LocalDateTime startDate, LocalDateTime endDate) - получение информации о количестве заказов за период;
- getAverageGrade(LocalDateTime startDate, LocalDateTime endDate) - получение средней оценки пользователей за период;
- getPopularityPizzasInfo(LocalDateTime startDate, LocalDateTime endDate) - получение информации о популярности пицц за период.

5. Представления

В рамках реализации данного проекта предполагается разработка части бэкэнда. На следующей стадии развития проекта будет вестись разработка части фронтенда.

6. Безопасность

Так как приложение хранит и обрабатывает личные данные пользователей (адрес, телефон), необходимо предусмотреть достаточный уровень защиты информации.

Предполагается следующий процесс аутентификации: пользователи проходят аутентификацию при вводе своего логина и пароля. Этот процесс позволяет приложению проверить подлинность пользователя и разрешить доступ к его аккаунту. После успешной аутентификации генерируется JWT токен, который будет использоваться для последующей авторизации и доступа к защищенным ресурсам.

JWT токен содержит информацию о пользователе и его правах доступа. После аутентификации токен генерируется с использованием секретного ключа и заданного срока действия. При каждом запросе клиента к защищенным ресурсам приложения токен передается в заголовке запроса. На сервере токен проверяется на подлинность и целостность. В случае успешной проверки пользователю разрешается доступ к ресурсам.

Срок действия токена должен быть ограничен определенным временем, например, 24 часами с момента его генерации. После истечения этого срока токен становится недействительным, и пользователь должен будет повторно пройти процесс аутентификации для получения нового токена.

7. Интеграции со сторонними сервисами

В рамках развития системы в будущем предполагаются следующие шаги:

- Интеграция с платежной системой для выполнения оплаты непосредственно при заказе;
- Интеграция с почтовым клиентом для автоматической рассылки информации о проводимых акциях

8. Проверка качества кода

Для проверки качества кода разрабатываемого приложения необходимо провести:

- Unit-тесты (для сервисов, публичных методов классов доменной модели, мапперов)
- интеграционные тесты (репозитории, контроллеры)

9. Развертывание

Требования к развертыванию:

- Операционная система: Windows 10 и выше.
- Требования к аппаратному обеспечению: минимум 2 ГБ оперативной памяти, двухъядерный процессор, 20 ГБ свободного дискового пространства.
- Сетевые требования: доступ к интернету для загрузки необходимых пакетов.

Инструкции по развертыванию:

- Установите Java Runtime Environment версии 21.
- Установите и настройте сервер приложений Apache Tomcat версии 10.
- Скопируйте WAR файл приложения в директорию развёртывания Tomcat.
- Настройте конфигурационные файлы приложения, указав параметры подключения к базе данных.
- Запустите сервер приложений Tomcat.

Миграция данных:

- Для миграции данных перед первоначальным запуском приложения выполните следующие шаги:
    - Подготовьте скрипт для создания базы данных "Pizzeria".
    - Запустите скрипт на сервере базы данных

10. Стек технологий

Функционал	Применяемые технологии
Основной фреймворк для организации приложения	Spring Framework 6 с модулем быстрого конфигурирования Spring Boot 3
Хранение данных	
СУБД	PostgreSQL
Доступ к данным	Репозитории Spring Data JPA на базе Hibernate
Обмен данными с пользователем	
Организация контроллеров	Spring Web MVC
Маппинг объектов в JSON	Jackson
Безопасность	
Организация аутентификации и доступа по ролям	Spring Security, с использованием JWT токена
Проверка качества кода	
Тестирование кода	JUnit5, Mockito
Оценка покрытия кода тестами	Средства IntelliJ IDEA
Code-review	Средства GitHub
Развертывание	
Наполнение БД (миграция)	Liquibase
Документирование	
Документирование кода	JavaDoc
Документирование API	OpenAPI v3 (Swagger), SpringDoc

Приложение 1 - User story

1. Я как пользователь, хотел бы иметь возможность регистрации в приложении, чтобы начать его использовать; 2. Я как пользователь, хотел бы иметь доступ к моему кабинету для анализа статистики (когда были заказы, что было заказано); 3. Я как пользователь, хотел бы иметь возможность добавления понравившегося блюда в избранное, для быстрого заказа в следующий раз; 4. Я как пользователь, хочу иметь возможность восстановления пароля через электронную почту, для уверенности, что я не потеряю доступ к моему кабинету; 5. Я как пользователь, хотел бы иметь возможность добавлять заказ в корзину, чтобы сформировать заказ; 6. Я как пользователь, хотел бы иметь возможность выбора пиццы со своим набором ингредиентов, чтобы сформировать заказ в соответствии с моими пожеланиями; 7. Я как пользователь, хотел бы иметь информацию о составе продуктов в пицце и их калорийности, для формирования заказа, отвечающего моим потребностям в заботе о своем здоровье; 8. Я как пользователь, хотел бы иметь возможность заказать пиццу, ограниченную моим бюджетом, чтобы всегда иметь возможность заказать еду; 9. Я как пользователь, хотел бы иметь возможность заказать не только классическую пиццу, но и другие варианты, для разнообразия моих заказов; 10. Я как пользователь, который заботится о своем здоровье, хотел бы иметь возможность заказать вегетарианскую пиццу; 11. Я как пользователь, хотел бы пользоваться гибкой системой бонусов, для того чтобы хотелось пользоваться только этим приложением; 12. Я как пользователь, хотел бы иметь возможность оставлять отзывы о заведении, качестве предоставляемых услуг, вариантах заказанных блюд, а также читать отзывы остальных клиентов, для совершения быстрого выбора; 13. Мне как пользователю, было бы приятно получать поздравления с праздниками от заведения, для формирования положительного впечатления о заведении; 14. Мне как пользователю, хотелось бы получать рассылку от заведения о проводимых акциях и скидках, для осуществления заказов по выгодным ценам
15. Я как администратор системы, хотел бы иметь возможность простого добавления информации об ингредиентах, их калорийности и стоимости, для экономии моего рабочего времени; 16.Я как администратор системы, хотел бы иметь возможность легкого поиска и корректировки информации о пиццах, их составах, ингредиентах, для экономии моего рабочего времени и минимизации ошибок; 17. Я как администратор системы, хотел бы иметь возможность легкой фильтрации информации о пользователях для предоставления им обратной связи (поздравления, рассылка информации об акциях), для выполнения моих обязанностей;

18. Я как администратор системы, хотел бы иметь статистику активности пользователей для стимулирования их активности, для повышения прибыли заведения; 19. Я как администратор системы, хотел бы иметь возможность блокирования пользователю доступа к возможности оставлять комментарии за нарушение правил, чтобы поддерживать порядок в системе
20. Я как владелец заведения, хотел бы иметь статистику в разрезе периодов нагрузки на заведение, для планирования загрузки поваров; 21. Я как владелец заведения, хотел бы иметь статистику в разрезе популярности блюд, используемых ингредиентов, для планирования закупаемых продуктов; 22. Я как владелец заведения, хотел бы иметь информацию о суммах прибыли за определенные периоды, для планирования расширения услуг; 23. Я как владелец заведения, хотел бы иметь возможность анализа отзывов о заведении, для улучшения качества обслуживания