



**Министерство науки и высшего образования Российской Федерации**  
**Федеральное государственное бюджетное образовательное учреждение**  
**высшего образования**  
**«Московский государственный технический университет**  
**имени Н.Э. Баумана**  
**(национальный исследовательский университет)»**  
**(МГТУ им. Н.Э. Баумана)**

**РАСПРЕДЕЛЁННАЯ СИСТЕМА ДЛЯ ЗАКУПКИ И ПРОДАЖИ  
ИНГРЕДИЕНТОВ ДЛЯ СОЗДАНИЯ ПАРФЮМЕРИИ КЛАССА ЛЮКС**

**Техническое задание**

**Листов 10**

Инв.№ подл.	Подп. и дата	Взлам.инв.№	Инв.№ дубл.	Подп. и дата

## ВВЕДЕНИЕ

По оценкам аналитиков DISCOVERY Research Group, в I пол.2021 г. объем рынка парфюмерии и косметики составил 256,9 млрд. руб. В условиях повышения цен на импортные товары, на рынке парфюмерии и косметики увеличилась доля российской продукции. По итогам 2020 г. наблюдался рост объемов производства по отношению к данным 2019 г. В I пол. 2021 г. было произведено 145,9 тыс. парфюмерии. Структура производства парфюмерии в России такова, что в I пол. 2021 г. основная доля пришлась на туалетную воду [1].

К тому же, с учетом пандемии коронавируса и перераспределения рынка, многие люди занялись производством парфюма (и не только) на дому, что тоже явилось прибыльным бизнесом, о чем свидетельствуют много разных источников, и бизнес такого рода продолжает набирать обороты.

Производство туалетной воды и духов – прибыльный бизнес. Мало кто знает, что в стоимости магазинных духов цена компонентов составляет порядка 10%, а все остальное – цена упаковки, зарплата рабочих и прибыль владельца бизнеса. Ингредиенты для духов не столь дороги, и технология производства не слишком сложна [2].

Производство духов в домашних условиях, как и любой другой бизнес на дому, имеет свои преимущества и недостатки, а также некоторые нюансы, которые важно учесть при рассмотрении бизнес-идеи.

Преимущества такого производства заключаются в высокой рентабельности, уникальности и неповторимости каждого авторского аромата (такие духи нельзя купить в масс-маркете или парфюмерном магазине), экологичности и натуральности продукта, отсутствии синтетических компонентов и не сезонном спросе.

Учитывая вышеперечисленные особенности производства, можно сделать вывод о том, что рынок парфюмерии довольно развит прибылен, предполагается выдвинуть платформу покупки/продажи ингредиентов для создания парфюма категории люкс на международный уровень, что позволит крупным кампаниям и индивидуальным предпринимателям, использующим платформу, создавать парфюм класса люкс из качественных неповторимых ингредиентов.

Данное техническое задание составлено для разработки распределённой системы для закупки и продажи ингредиентов для создания парфюмерии класса люкс. Техническое задание выполнено на основе ГОСТ 19.201–78 «ЕСПД. Техническое задание. Требования к содержанию и оформлению».

## **Глоссарий**

1. Парфюмерная основа – основой ингредиент парфюма, закрепляющий аромат готового продукта. Спирт (этиловый 96%) или масло (виноградных косточек, миндальное, кокосовое, масло жожоба).
2. Парфюмерная нота – это дескриптор аромата, который можно ощутить при нанесении духов. Ноты делятся на три класса: верхние / головные ноты, средние / сердечные ноты и базовые ноты, которые обозначают группы ароматов, которые можно ощутить в зависимости от времени после нанесения парфюма.
3. Группа ароматов – это ароматы с похожими нотами и запахом.
4. Ингредиенты для парфюмерии класса люкс – исключительно натуральные масла и компоненты, никакой синтетики.
5. Субстантивность – способность компонента долгое время передавать аромат с поверхности, на которую он нанесен.
6. ПВЗ – пункт выдачи заказов.
7. REST (Representational State Transfer) – архитектурный стиль взаимодействия компонентов распределённого приложения в сети.
8. Фронтенд – серверное приложение, принимающее запросы от пользователя. На каждый из типов запросов определяется, как организовать его выполнение. Принимает запросы, анализирует их и в соответствии с заложенным алгоритмом выполняет запросы к бекенду.
9. Бекенд – серверное приложение, выполняющее определенную задачу, например, взаимодействие с СУБД. Бекенды принимают запросы от фронтенда.
10. ПО – программное обеспечение.

## **Основания для разработки**

Разработка ведётся в рамках выполнения лабораторных работ по курсу «Методология программной инженерии» на кафедре «Программное обеспечение ЭВМ и информационные технологии» факультета «Информатика и системы управления» МГТУ им. Н.Э. Баумана.

## **Назначение разработки**

Разрабатываемая система должна предоставлять пользователям возможность покупки ингредиентов для создания парфюма из разных уголков мира. Должен быть предусмотрен поиск подходящих компонентов по таким параметрам, как тип продукта, его стоимость, ноты в парфюме (при наличии), страна производства, объем в наличии.

## **Существующие аналоги**

На сегодняшний день закупка ингредиентов для создания парфюмерии представлена следующими основными сайтами (порталами):

- PARFCLUB – <https://parfclub.shop/>;
- Лавка Парфюмера – <https://parflavka.ru/>;
- Дом Ароматов – <https://domaromatov.ru/>;
- Fleuron – <https://fleuron.ru/>.

У вышеперечисленных сайтов (порталов) есть определенные недостатки:

- возможность закупки компонентов только со склада, а не напрямую от производителя (4);
- основная ориентированность – продажа готовой парфюмерии, большинство продукции синтетика или заменители оригинальных продуктов (2);

- отсутствие информации о стране (заводе) производителе сырья (3);
- отсутствие информации о принадлежности к нотам парфюма (2);
- отсутствие информации о принадлежности к группе ароматов (2);
- отсутствие информации о субстантивности продукции (1);
- ограниченность тремя и менее производителями (2);
- отсутствие возможности закупки парфюмерной основы (масел, спирта и т. д.) (1);

По сравнению с существующими аналогами разрабатываемый проект должен иметь следующие преимущества:

- в основе должна лежать микросервисная архитектура, решающая сложности с масштабированием, обслуживанием и внесением изменений в функциональность;
- работа напрямую с производителем;
- наличие информации о стране/заводе производителе;
- наличие информации о ноте в парфюмерной композиции;
- наличие информации о группе ароматов;
- наличие информации о субстантивности продукта;
- возможность закупки парфюмерной основы.

## **Описание системы**

Разрабатываемый сервис должен представлять собой распределённую систему для закупки ингредиентов для создания парфюма класса люкс. Если клиент хочет оформить заказ, ему необходимо зарегистрироваться, указав информацию: фамилия, имя, отчество, номер телефона, электронная почта. В случае, если зарегистрированному ранее пользователю нужно получить

информацию о статусе заказа, ему нужно авторизоваться. Для неавторизованных пользователей доступен только просмотр общей информации. На рисунке 1 отображена схема предметной области.

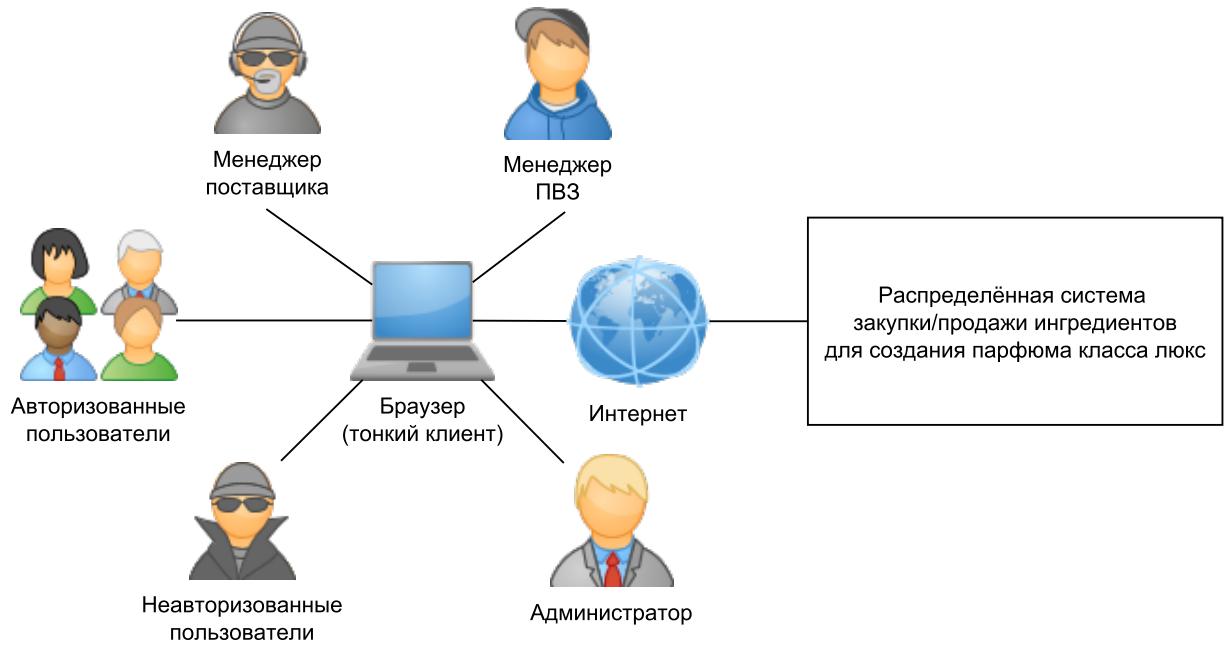


Рисунок 1 – Схема предметной области

## **Общие требования к системе**

Разрабатываемая система должна отвечать следующим требованиям.

1. Разрабатываемое ПО должно поддерживать функционирование системы в режиме 24 часов, 7 дней в неделю, 365 дней в году ( $24/7/365$ ) со среднегодовым временем доступности не менее 99.9%. Допустимое время, в течении которого система недоступна, за год должна составлять  $24 \cdot 365 \cdot 0.001 = 8.76$  ч.
2. Время восстановления системы после сбоя не должно превышать 15 минут.
3. Каждый узел должен автоматически восстанавливаться после сбоя.
4. Система должна поддерживать возможность «горячего» переконфигурирования системы. Необходимо предусмотреть поддержку добавления нового узла во время работы системы без рестарта.
5. Обеспечить безопасность работоспособности за счёт отказоустойчивости узлов.

## **Требования к функциональным характеристикам**

1. По результатам работы модуля сбора статистики медиана времени отклика системы на запросы пользователя на получение информации не должна превышать 3 секунд.
2. По результатам работы модуля сбора статистики медиана времени отклика системы на запросы, добавляющие или изменяющие информацию на портале не должна превышать 7 секунд.
3. Медиана времени отклика системы на действия пользователя должна быть менее 0.8 секунд при условии работы на рекомендованной аппаратной конфигурации, задержках между взаимодействующими сервисами менее 0.2 секунды и одновременном числе работающих пользователей менее 100 на каждый сервер, обслуживающий внешний интерфейс.

4. Система должна обеспечивать возможность запуска в современных браузерах: не менее 85% пользователей Интернета должны пользоваться ей без какой-либо деградации функционала.

### **Функциональные требования к системе с точки зрения пользователя**

Система должна обеспечивать реализацию нижеуказанных функций.

1. Регистрация и авторизация пользователей с валидацией вводимых данных как через интерфейс приложения.
2. Аутентификация пользователей.
3. Разделение всех пользователей на три роли:
  - Пользователь (неавторизованный пользователь);
  - Клиент (авторизованный пользователь);
  - Менеджер поставщика;
  - Менеджер ПВЗ;
  - Администратор.
4. Предоставление возможностей **Пользователю, Клиенту, Менеджеру поставщика, Менеджеру ПВЗ, Администратору** представленных в таблице 1.

Таблица 1 – Функции пользователей

Пользователь	<ol style="list-style-type: none"><li>1. просмотр списка доступных ингредиентов для закупки (включая поиск, фильтрацию);</li><li>2. регистрация в системе;</li><li>3. авторизация в системе;</li></ol>
--------------	--

Клиент	<ol style="list-style-type: none"> <li>1. просмотр списка доступных ингредиентов для закупки (включая поиск, фильтрацию);</li> <li>2. авторизация в системе;</li> <li>3. получение и изменение информации текущего аккаунта;</li> <li>4. просмотр всех заказов, созданных/обрабатываемых/отмененных/- полученных на имя авторизованного клиента</li> <li>5. получение детальной информации по конкретному заказу на имя авторизованного клиента;</li> <li>6. оформление заказа на имя авторизованного клиента;</li> <li>7. отмена заказа (при возможности), оформленного на имя авторизованного клиента;</li> <li>8. получение информации о статусе текущего пользователя в программе лояльности.</li> </ol>
Менеджер поставщика	<ol style="list-style-type: none"> <li>1. просмотр списка доступных ингредиентов для закупки (включая поиск, фильтрацию) в ячейке поставщика;</li> <li>2. авторизация в системе;</li> <li>3. просмотр и редактирование всех доступных ингредиентов для закупки в ячейке поставщика;</li> <li>4. просмотр и редактирование всех оформленных заказов в ячейке поставщика;</li> <li>5. получение детальной информации по конкретному заказу в ячейке поставщика;</li> </ol>
Менеджер ПВЗ	<ol style="list-style-type: none"> <li>1. авторизация в системе;</li> <li>2. просмотр и редактирование всех оформленных заказов;</li> <li>3. получение детальной информации по конкретному заказу;</li> <li>4. отмена любого оформленного заказа;</li> </ol>

<b>Администратор</b>	<ol style="list-style-type: none"> <li>1. просмотр списка доступных ингредиентов для закупки (включая поиск, фильтрацию);</li> <li>2. авторизация в системе;</li> <li>3. получение и редактирование информации о любом клиенте, зарегистрированном в системе;</li> <li>4. просмотр и редактирование всех оформленных заказов;</li> <li>5. получение детальной информации по конкретному заказу;</li> <li>6. отмена любого оформленного заказа;</li> </ol>
----------------------	---

## Входные данные

Входные параметры системы представлены в таблице 2.

Таблица 2 – Входные данные

Сущность	Входные данные
Клиент / Менеджер поставщика / Менеджер ПВЗ / Адми- нистратор	<ol style="list-style-type: none"> <li>1. <i>фамилия, имя, отчество (при наличии)</i> не более 256 символов каждое поле;</li> <li>2. <i>логин</i> не менее 10 символов и не более 128;</li> <li>3. <i>пароль</i> не менее 8 символов и не более 128, как минимум одна заглавная и одна строчная буква, только латинские буквы, без пробелов, как минимум одна цифра;</li> <li>4. <i>номер телефона</i>;</li> <li>5. <i>электронная почта</i>;</li> </ol>
Поставщик	<ol style="list-style-type: none"> <li>1. <i>идентификатор</i>;</li> <li>2. <i>страна</i> не более 256 символов;</li> <li>3. <i>наименование поставщика</i> не более 256 символов;</li> <li>4. <i>описание</i> не более 2048 символов.</li> </ol>

*Продолжение на следующей странице*

<b>Сущность</b>	<b>Входные данные</b>
Товар	1. <i>идентификатор</i> ; 2. <i>название</i> не более 256 символов; 3. <i>идентификатор поставщика</i> ; 4. <i>нота</i> ( <i>при наличии</i> ); 5. <i>группа ароматов</i> ( <i>при наличии</i> ); 6. <i>субстантивность</i> ( <i>при наличии</i> ); 7. <i>объем</i> единицы товара (мл); 8. <i>цена</i> за единицу товара; 9. <i>количество</i> ингредиента доступное для заказа; 10. <i>описание</i> не более 2048 символов.
Заказ	1. <i>идентификатор</i> ; 2. <i>идентификатор клиента</i> ; 3. <i>статус</i> (IN CART/ CREATED/ IN PROGRESS/ DONE/ PAID/ RECEIVED/ CANCELED);
Позиция заказа	1. <i>идентификатор</i> ; 2. <i>идентификатор заказа</i> ; 3. <i>идентификатор товара</i> ; 4. <i>количество</i> ; 5. <i>возврат</i> (TRUE/FALSE); 6. <i>статус</i> (APPROVED/ ASSEMBLED/ IN DELIVERY/ DELIVERED);
Оплата	1. <i>идентификатор</i> ; 2. <i>идентификатор заказа</i> ; 3. <i>статус</i> (NOT PAID/PAID); 4. <i>сумма</i> ; 5. <i>дата и время</i> .

### **Выходные параметры**

Выходными параметрами системы являются web-страницы. В зависимости от запроса и текущей роли пользователя они содержат следующую информацию (таблица 3).

Таблица 3 – Выходные параметры

Пользователь	1. список всех доступных для заказа товаров, с указанием: <ul style="list-style-type: none"> <li>● <i>наименование товара;</i></li> <li>● <i>страна производства;</i></li> <li>● <i>поставщик;</i></li> <li>● <i>нота;</i></li> <li>● <i>группа ароматов;</i></li> <li>● <i>субстантивность;</i></li> <li>● <i>объем единицы товара;</i></li> <li>● <i>цена;</i></li> <li>● <i>описание.</i></li> </ul>
Клиент	1. список всех доступных для заказа товаров, с указанием: <ul style="list-style-type: none"> <li>● <i>наименование товара;</i></li> <li>● <i>страна производства;</i></li> <li>● <i>поставщик;</i></li> <li>● <i>нота;</i></li> <li>● <i>группа ароматов;</i></li> <li>● <i>субстантивность;</i></li> <li>● <i>объем единицы товара;</i></li> <li>● <i>цена;</i></li> <li>● <i>количество доступное для заказа;</i></li> <li>● <i>описание.</i></li> </ul>
	2. детальная информация о пользователе, вошедшем в систему; <ul style="list-style-type: none"> <li>● <i>фамилия, имя, отчество;</i></li> <li>● <i>логин;</i></li> <li>● <i>номер телефона;</i></li> <li>● <i>электронная почта.</i></li> </ul>

	<p>3. список оформленных заказов на пользователя, вошедшего в систему, предоставляется информация:</p> <ul style="list-style-type: none"> <li>● <i>идентификатор заказа;</i></li> <li>● <i>позиции заказа;</i></li> <li>● <i>статус позиций заказа;</i></li> <li>● <i>статус заказа;</i></li> <li>● <i>общая сумма заказа.</i></li> </ul>
Администратор	<p>1. список всех доступных для заказа товаров, с указанием:</p> <ul style="list-style-type: none"> <li>● <i>наименование товара;</i></li> <li>● <i>страна производства;</i></li> <li>● <i>поставщик;</i></li> <li>● <i>нота;</i></li> <li>● <i>группа ароматов;</i></li> <li>● <i>субстантивность;</i></li> <li>● <i>объем единицы товара;</i></li> <li>● <i>цена;</i></li> <li>● <i>количество доступное для заказа;</i></li> <li>● <i>описание.</i></li> </ul> <p>2. детальная информация о всех пользователях:</p> <ul style="list-style-type: none"> <li>● <i>фамилия, имя, отчество;</i></li> <li>● <i>логин;</i></li> <li>● <i>номер телефона;</i></li> <li>● <i>электронная почта.</i></li> </ul> <p>3. список заказов всех пользователей, с указанием:</p> <ul style="list-style-type: none"> <li>● <i>логин пользователя;</i></li> <li>● <i>идентификатор заказа;</i></li> <li>● <i>позиции заказа;</i></li> <li>● <i>статус позиций заказа;</i></li> <li>● <i>статус заказа;</i></li> <li>● <i>общая сумма заказа.</i></li> </ul>

<b>Менеджер поставщика</b>	<p>1. список всех доступных для заказа товаров в ячейке поставщика, с указанием:</p> <ul style="list-style-type: none"> <li>● <i>наименование товара;</i></li> <li>● <i>страна производства;</i></li> <li>● <i>поставщик;</i></li> <li>● <i>нота;</i></li> <li>● <i>группа ароматов;</i></li> <li>● <i>субстантивность;</i></li> <li>● <i>объем единицы товара;</i></li> <li>● <i>цена;</i></li> <li>● <i>количество доступное для заказа;</i></li> <li>● <i>описание.</i></li> </ul> <p>2. список позиций заказов пользователей с указанием:</p> <ul style="list-style-type: none"> <li>● <i>идентификатор позиции;</i></li> <li>● <i>идентификатор товара;</i></li> <li>● <i>наименование товара;</i></li> <li>● <i>количество товара в позиции;</i></li> <li>● <i>статус позиции.</i></li> </ul>
<b>Менеджер ПВЗ</b>	<p>1. список заказов всех пользователей, с указанием:</p> <ul style="list-style-type: none"> <li>● <i>идентификатор заказа;</i></li> <li>● <i>ФИО пользователя;</i></li> <li>● <i>логин пользователя;</i></li> <li>● <i>позиции заказа;</i></li> <li>● <i>статус позиций заказа;</i></li> <li>● <i>статус заказа;</i></li> <li>● <i>общая сумма заказа.</i></li> </ul>

## Топология Системы

На рисунке 2 изображён один из возможных вариантов топологии разрабатываемой распределенной Системы.

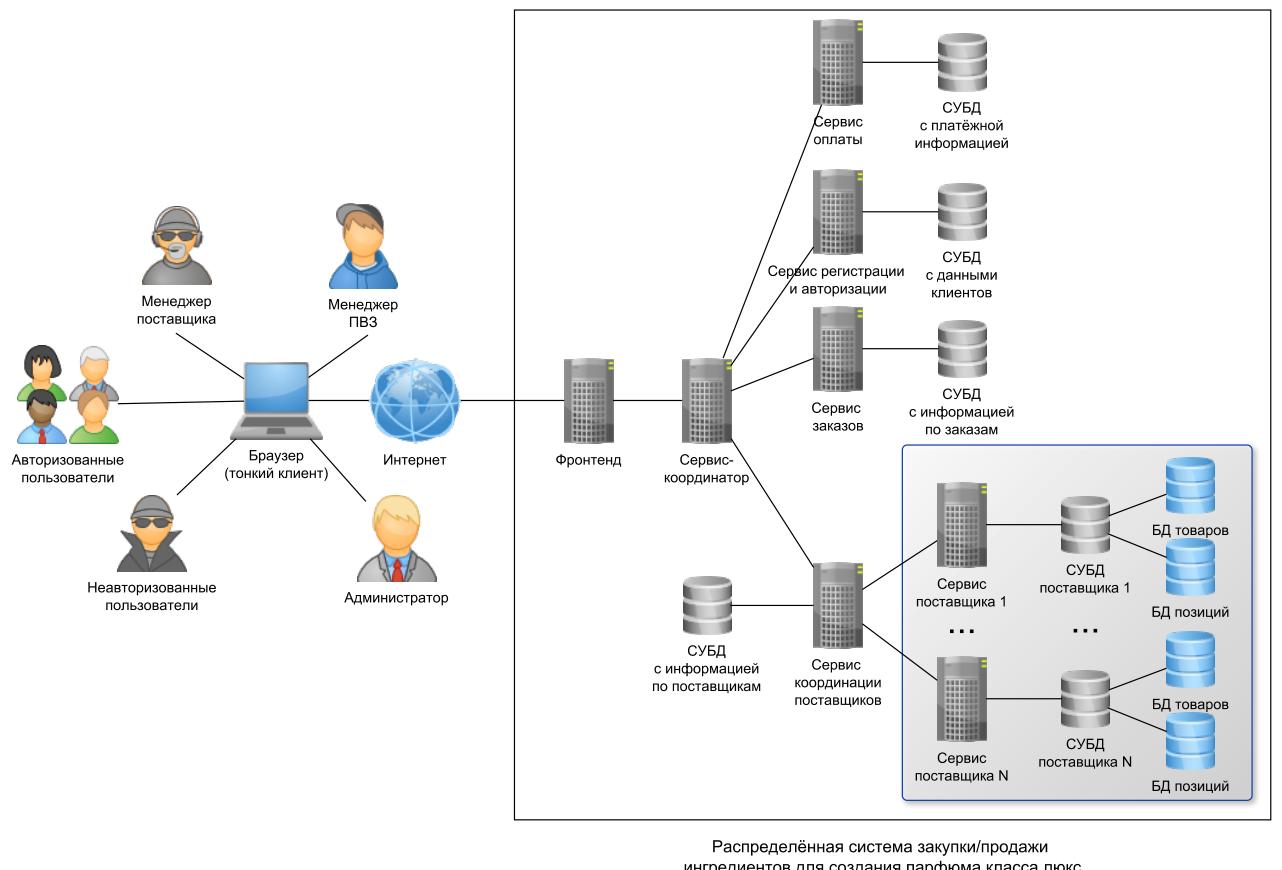


Рисунок 2 – Топология системы

Система будет состоять из фронтенда и 5 подсистем и N дополнительных сервисов:

- сервис-координатор;
- сервис регистрации и авторизации;
- сервис оплаты;
- сервис заказов;
- сервис координации поставщиков;
- сервисы поставщиков.

**Фронтенд** принимает запросы от пользователей по протоколу HTTP и анализирует их. На основе проведенного анализа выполняет запросы к мик-

росервисам бекенда, агрегирует ответы и отсылает их пользователю.

**Сервис-координатор** – единая точка входа и межсервисной коммуникации.

**Сервис-регистрации и авторизации** отвечает за:

- возможность регистрации нового клиента;
- аутентификацию пользователя (клиента/ администратора/ менеджера поставщика/ менеджера ПВЗ);
- авторизацию пользователя;
- выход из сессии.

**Сервис оплаты** реализует функции:

- проведение платежа от клиента к системе;
- получение статуса оплаты;
- отмену платежа.

**Сервис заказов** реализует следующие функции:

- получение списка всех заказов пользователя;
- получение информации о конкретном заказе;
- создание, отмена заказа.

**Сервис координации поставщиков** реализует следующие функции:

- получение списка всех поставщиков;
- получение информации о конкретном поставщике;

- перенаправление частей заказа пользователя конкретным поставщикам.

**Сервисы поставщиков** отвечают за:

- получение списка всех товаров поставщика;
- получение информации о конкретном товаре;
- получение списка всех позиций, направленных поставщику;
- получение информации о конкретной позиции, направленной поставщику;
- создание, изменение товара;
- создание, изменение позиции заказа;
- перенаправление частей заказа пользователя конкретным поставщикам.

### **Требования к программной реализации**

1. Требуется использовать СОА (сервис-ориентированную архитектуру) для реализации системы.
2. Система состоит из микросервисов. Каждый микросервис отвечает за свою область логики работы приложения и должны быть запущены изолированно друг от друга.
3. При необходимости, каждый сервис имеет своё собственное хранилище, запросы между базами запрещены.
4. При разработке базы данных необходимо учитывать, что доступ к ней должен осуществляться по протоколу TCP.
5. Необходимо реализовать один web-интерфейс для фронта. Интерфейс должен быть доступен через тонкий клиент (браузер).

6. Для межсервисного взаимодействия использовать HTTP (придерживаться RESTful).
7. Выделить Gateway Service как единую точку входа и межсервисной коммуникации. В системе не должно осуществляться горизонтальных запросов.
8. При недоступности систем портала должна осуществляться деградация функционала или выдача пользователю сообщения об ошибке.
9. Необходимо предусмотреть авторизацию пользователей, как через интерфейс приложения, так и через популярные социальные сети.
10. Валидацию входных данных необходимо проводить и на стороне пользователя, и на стороне фронтенда. Микросервисы бекенда не должны валидировать входные данные, поскольку пользователь не может к ним обращаться напрямую, они должны получать уже отфильтрованные входные данные.
11. Для запросов, выполняющих обновление данных на нескольких узлах распределенной системы, в случае недоступности одной из систем, необходимо выполнять полный откат транзакции.
12. Приложение должно поддерживать возможность горизонтального и вертикального масштабирования за счет увеличения количества функционирующих узлов и совершенствования технологий реализации компонентов и всей архитектуры системы.
13. Код хранить на Github, для сборки использовать Github Actions.
14. Gateway Service должен запускаться на порту 8000, остальные сервисы запускать на портах 8010, 8020, 8030, 8040, 8050-N (для сервисов поставщиков).
15. Каждый сервис должен быть завернут в docker.

## **Функциональные требования к подсистемам**

Подсистемы: фронтенд, бекенд-координатор, бекенд регистрации и авторизации, бекенд бронирования, бекенд оплаты, бекенд лояльности.

**Фронтенд** – серверное приложение, предоставляет пользовательский интерфейс и внешний API системы, при разработке которого нужно учитывать следующее:

- должен принимать запросы по протоколу HTTP и формировать ответы пользователям в формате HTML;
- в зависимости от типа запроса должен отправлять последовательные запросы в соответствующие микросервисы;
- запросы к микросервисам необходимо осуществлять по протоколу HTTP;
- данные необходимо передавать в формате JSON.

**Сервис-координатор** – серверное приложение, через которое проходит весь поток запросов и ответов, должен соответствовать следующим требованиям разработки:

- принимать и возвращать данные в формате JSON по протоколу HTTP;
- накапливать статистику запросов, в случае, если система не ответила  $N$  раз, то в  $N + 1$  раз вместо запроса сразу отдавать fallback. Через некоторое время выполнить запрос к реальной системе, чтобы проверить её состояние;
- выполнять проверку существования клиента, также регистрацию и аутентификацию пользователей;
- получение информации и обновление данных о зарегистрированном пользователе;
- получение информации из списка всех заказов, так и от заказов конкретного пользователя;
- оформление и отзыв созданного ранее заказа;

- получение информации из списка всех поставщиков, их редактирование;
- получение списка всех товаров, позиций от сервиса координации поставщиков и их изменение.

**Сервис координации поставщиков** должен реализовывать следующие функциональные возможности:

- принимать и возвращать данные в формате JSON по протоколу HTTP;
- получение списка всех товаров, позиций;
- перенаправление запросов от сервиса-координатора конкретным сервисам поставщиков.

Хранимая в базе данных сущность, ассоциированная с сервисом, детально представлена в таблице 7.

Таблица 4 – Состав сущностей

Сущность	Поля	Обязательность
Поставщик	<i>идентификатор</i> , является первичным ключом	да
	<i>страна</i> , не более 256 символов	да
	<i>наименование поставщика</i> , не более 256 символов	да
	<i>описание</i> , не более 2048 символов	нет

Каждый из **сервисов поставщиков** должен реализовывать следующие функциональные возможности:

- принимать и возвращать данные в формате JSON по протоколу HTTP;
- изменять данные о товарах и позициях;
- проверка возможности сбора позиции заказа по наличию товара у поставщика.

Ассоциированная с этим сервисом базы данных содержат сущности, детально представленная в таблице 8.

Таблица 5 – Состав сущностей

<b>Сущность</b>	<b>Поля</b>	<b>Обязательность</b>
Товар	<i>идентификатор</i> , является первичным ключом	да
	<i>название</i> , не более 256 символов	да
	<i>идентификатор поставщика</i>	да
	<i>нота</i>	нет
	<i>группа ароматов</i>	нет
	<i>субстантивность</i>	нет
	<i>объем</i>	да
	<i>цена</i>	да
	<i>количество</i>	да
Позиция заказа	<i>идентификатор</i> , является первичным ключом	да
	<i>идентификатор заказа</i>	да
	<i>идентификатор товара</i>	да
	<i>количество</i>	да
	<i>возврат</i> TRUE/FALSE	да
	<i>статус</i> APPROVED/ ASSEMBLED/ IN DELIVERY/ DELIVERED	да

**Сервис заказов** должен реализовывать следующие функциональные возможности:

- принимать и возвращать данные в формате JSON по протоколу HTTP;
- создавать заказ;
- изменять данные (статус) заказа.

Ассоциированная с этим сервисом база данных содержит сущность, детально представленная в таблице 8.

Таблица 6 – Состав сущностей

Сущность	Поля	Обязательность
Заказ	<i>идентификатор</i> , является первичным ключом	да
	<i>идентификатор клиента</i>	да
	<i>статус</i> IN CART/ CREATED/ IN PROGRESS/ DONE/ PAID/ RECEIVED/ CANCELED	да

**Сервис-регистрации и авторизации** должен реализовывать следующие функциональные возможности:

- принимать и возвращать данные в формате JSON по протоколу HTTP;
- возможность регистрации нового клиента и обновление данных уже существующего;
- проверка существования клиента;
- обеспечение авторизации пользователя через аккаунт в системе.

Хранимая в базе данных сущность, ассоциированная с сервисом, детально представлена в таблице 7.

Таблица 7 – Состав сущностей

Сущность	Поля	Обязательность
Аккаунт	<i>фамилия</i> , не более 256 символов	да
	<i>имя</i> , не более 256 символов	да
	<i>отчество</i> , не более 256 символов	нет
	<i>логин</i> , является первичным ключом	да
	<i>захешированный пароль</i>	да
	<i>номер телефона</i>	да
	<i>электронная почта</i>	да

**Сервис оплаты** реализует функции:

- получение и отправка данных в формате JSON по протоколу HTTP;

- предоставления информации об оплате по её идентификатору;
- проведения оплаты;
- получения и обновления статуса оплаты.

Ассоциированная с этим сервисом база данных содержит сущность, детально представленная в таблице 8.

Таблица 8 – Состав сущностей

Сущность	Поля	Обязательность
Платёж	<i>идентификатор</i> , является первичным ключом	да
	<i>идентификатор заказа</i>	да
	<i>статус</i> , NOT PAID/PAID	да
	<i>сумма</i>	да
	<i>дата</i>	да

## Алгоритм работы системы

Для обычного неавторизованного пользователя доступен только каталог товаров.

Сервис-координатор принимает **запрос на каталог** от пользователя, перенаправляет запрос сервису координации поставщиков, который в свою очередь перенаправляет запрос на каждый из сервисов поставщиков, где каждый сервис предоставляет каталог своих товаров, направляет ответ сервису координации поставщиков, который агрегирует все в одну таблицу и отправляет сервису-координатору, который передает каталог пользователю.

Если пользователь решил **зарегистрироваться**, сервис-координатор перенаправляет запрос на сервис регистрации и авторизации, происходит внесение пользователя в базу данных, ответ возвращается сообщением об успешной регистрации пользователя.

Если пользователь решил **авторизоваться**, сервис-координатор перенаправляет запрос на сервис регистрации и авторизации, происходит сверка предоставленными пользователем данными и имеющимися в базе данными, ответ возвращается сообщением об успешной авторизации пользователя.

Если клиент заходит на **станицу заказов**, происходит выгрузка информации о заказах. Сервис-координатор направляет запрос к сервису заказов, получает набор заказов пользователя, после чего, по полученным данным обращается к сервису координации поставщиков, который в свою очередь перенаправляет запрос на все сервисы поставщиков, которые по запросу выдают позиции заказов по идентификатору заказа и возвращают таблицу сервису координации поставщиков, который агрегирует полученные данные в одну таблицу и передает сервису-координатору, который передает ответ клиенту.

**Заказы**, хранящиеся у клиента **в корзине**, хранятся в кэше временно на стороне клиента. После **оформления заказа** запрос отправляется на сервис-координатор, который перенаправляет часть запроса (заказ) на сервис заказов, где вносится соответствующая информация, другую часть запроса (позиции заказа) перенаправляет сервису координации поставщиков, который перенаправляет части запроса определенным в позициях сервисам поставщиков, где также вносится соответствующая информация. При этом, информация о заказе удаляется из кэша пользователя.

**Менеджер поставщика** через сервис-координатор получает **список позиций поставщика** с возможностью изменения статуса позиций. Как только позиция сменила статус на «*IN DELIVERY*», менеджер поставщика больше не имеет к ней доступ.

**Менеджер ПВЗ** через сервис-координатор получает **список заказов клиента и позиций данного заказа** с возможностью изменения статуса заказа и изменение статуса позиций с «*IN DELIVERY*» на «*DELIVERED*». Как только все позиции заказа имеют статус «*DELIVERED*», статус заказа меняется на «*DONE*».

После того как клиент приходит забрать заказ, менеджер ПВЗ инициирует запрос к сервису-координатору, который передает запрос сервису заказов, получая ответ передает его сервису координации поставщиков, и т.д. получая позиции по заказу, высчитывается сумма заказа, и предается **сервису оплаты**. Как только пользователь оплатил заказ, статус отплаты меняется на «*PAID*». Автоматически по запросу статус заказа меняется на «*PAID*», и только после этого менеджер ПВЗ может выдать заказ пользователю и сменить его статус на «*RECEIVED*».

## **Требования к составу и параметрам технических средств**

Все серверные приложения должны потреблять суммарно не более 2 Гбайт оперативной памяти и работать на сервере с процессором Intel(R) Core(TM) i7-10510U CPU 1.80GHz.

## **Требования к надёжности**

Система должна работать в соответствии с данным техническим заданием без рестарта. Необходимо использовать «зеркальные серверы» для всех подсистем, которые будут держать нагрузку в случае сбоя до тех пор, пока основной сервер не восстановится.

## **Требования к документации**

Исполнитель должен подготовить и передать заказчику руководство:

- для Администратора Системы;
- для Пользователя Системы;
- для Клиента Системы;
- для Менеджера Поставщика;
- для Менеджера ПВЗ;
- по развёртыванию Системы.

## **Концептуальный дизайн**

Концептуальный дизайн позволяет рассмотреть создаваемую систему с точки зрения пользователей. На рисунке 3 отображена контекстная диаграмма верхнего уровня, которая обеспечивает наиболее общее или абстрактное описание работы системы. Данный вид диаграммы позволяет формализовать описание запросов пользователя и ответов системы на них, отобразив её в виде «чёрного ящика».

Для уточнения деталей по операции оформления заказов, отображённой на диаграмме верхнего уровня, используется дочерняя диаграмма, которая изображена на рисунке 4. Она определяет последовательность выполнения операций в системе при обработке запроса клиента.

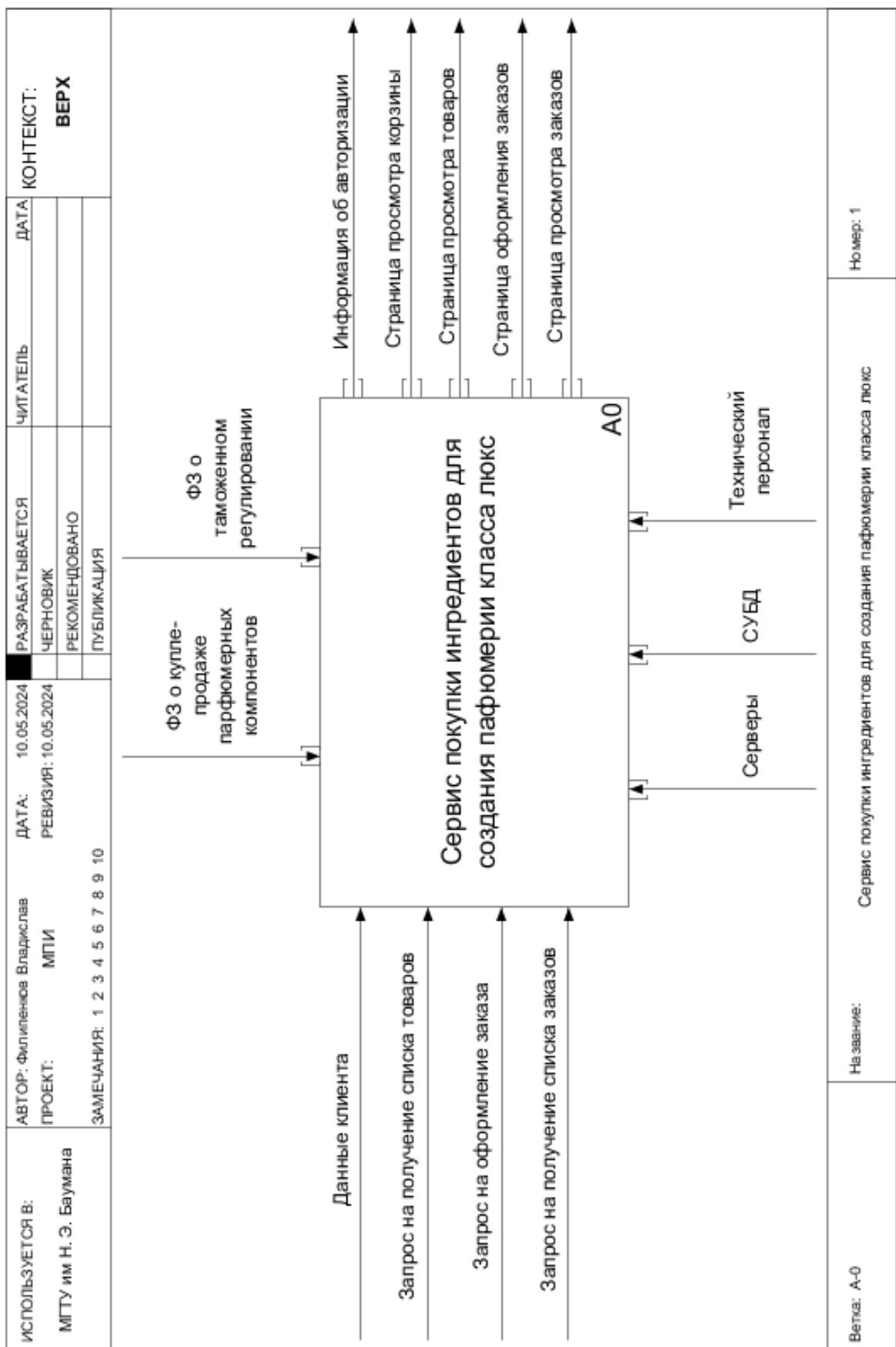


Рисунок 3 – Концептуальная модель системы в нотации IDEF0

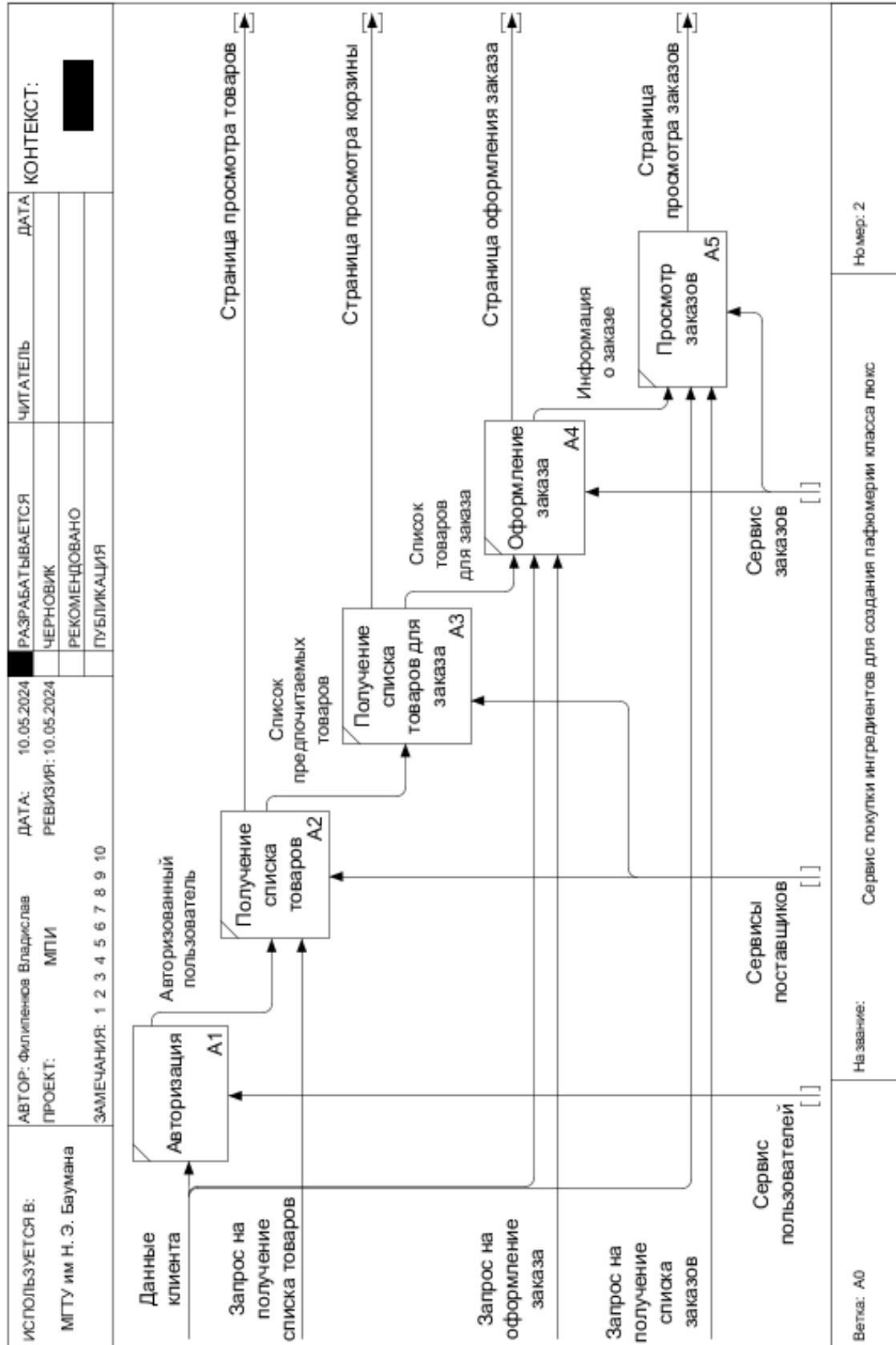


Рисунок 4 – Детализированная концептуальная модель системы в нотации IDEF0

## **Сценарии функционирования системы**

### **Регистрация клиента**

1. Пользователь нажимает на кнопку «Зарегистрироваться» в интерфейсе.
2. Пользователь перенаправляется на страницу, которая содержит поля для заполнения его данных.
3. Пользователь вводит данные в форму и для завершения регистрации нажимает на кнопку «Готово», тем самым подтверждая верность своих данных, а также согласие на их обработку и хранение.
4. Если пользователь с введенным для регистрации логином уже существует, то клиент перенаправляется с уведомлением на страницу авторизации. При успешной регистрации клиент попадает на страницу своего профиля в системе.

### **Авторизация клиента**

1. Пользователь нажимает на кнопку «Войти» в интерфейсе.
2. Пользователь перенаправляется на страницу авторизации, которая содержит поля для заполнения логина и пароля.
3. Пользователь завершает работу с формой авторизации нажатием кнопки «Готово».
4. При обнаружении ошибки в данных, пользователь перенаправляется на страницу ошибки; при совпадении данных с записью в базе данных аккаунтов пользователь получает доступ к системе.

### **Оформление заказа (Расширенная версия)**

1. Клиент нажимает кнопку «Главная/Каталог».

2. Клиент перенаправляется на страницу, которая содержит список товаров.
3. Клиент может применить фильтры для поиска желаемых товаров (ценовой диапазон, категория товара, страна производства, поставщик и т. п.).
4. Клиент нажимает на понравившуюся позицию и попадает на страницу с полным описанием товара, где он может добавить желаемое/доступное количество товара в корзину и вернуться на страницу товаров.
5. Добавив необходимые товары в корзину, клиент нажимает кнопку «Корзина», клиент перенаправляется в окно просмотра корзины, в которой будут отображаться товары и их количество.
6. Сверив товары из корзины, с тем что клиент хотел приобрести из изменив (при желании) товары перед оформлением заказа, клиент нажимает кнопку «Оформить заказ», после чего перенаправляется на страницу оформления заказов.
7. Окончательно сверив полный список товаров для заказа, клиент нажимает кнопку «Готово», выражая своё согласие на оформление заказа.
8. Если клиент не хочет оформлять заказ, он нажимает на кнопку «Отмена», затем автоматически перенаправится в окно просмотра корзины.

## Диаграммы прецедентов

В системе выделены 5 ролей: Пользователь, Клиент, Администратор, Менеджер Поставщика, Менеджер ПВЗ. На рисунках 5-9 представлены диаграммы прецедентов для каждой из ролей.

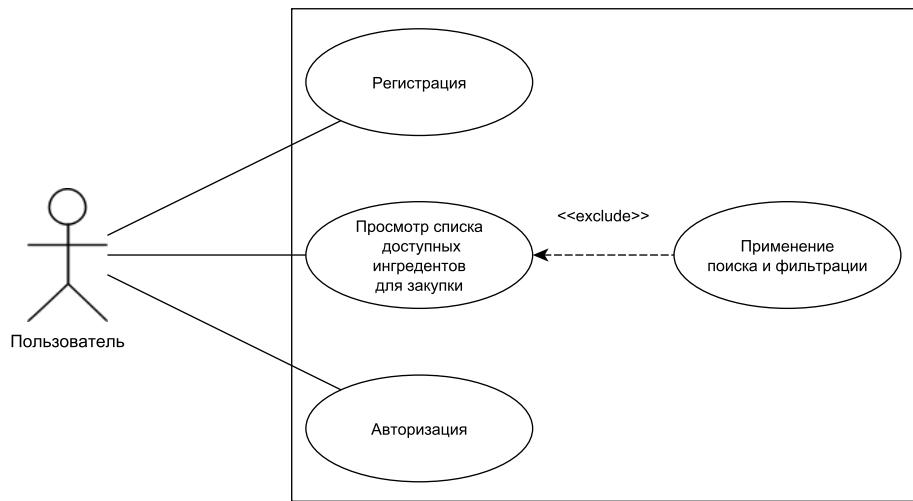


Рисунок 5 – Диаграмма прецедентов с точки зрения пользователя

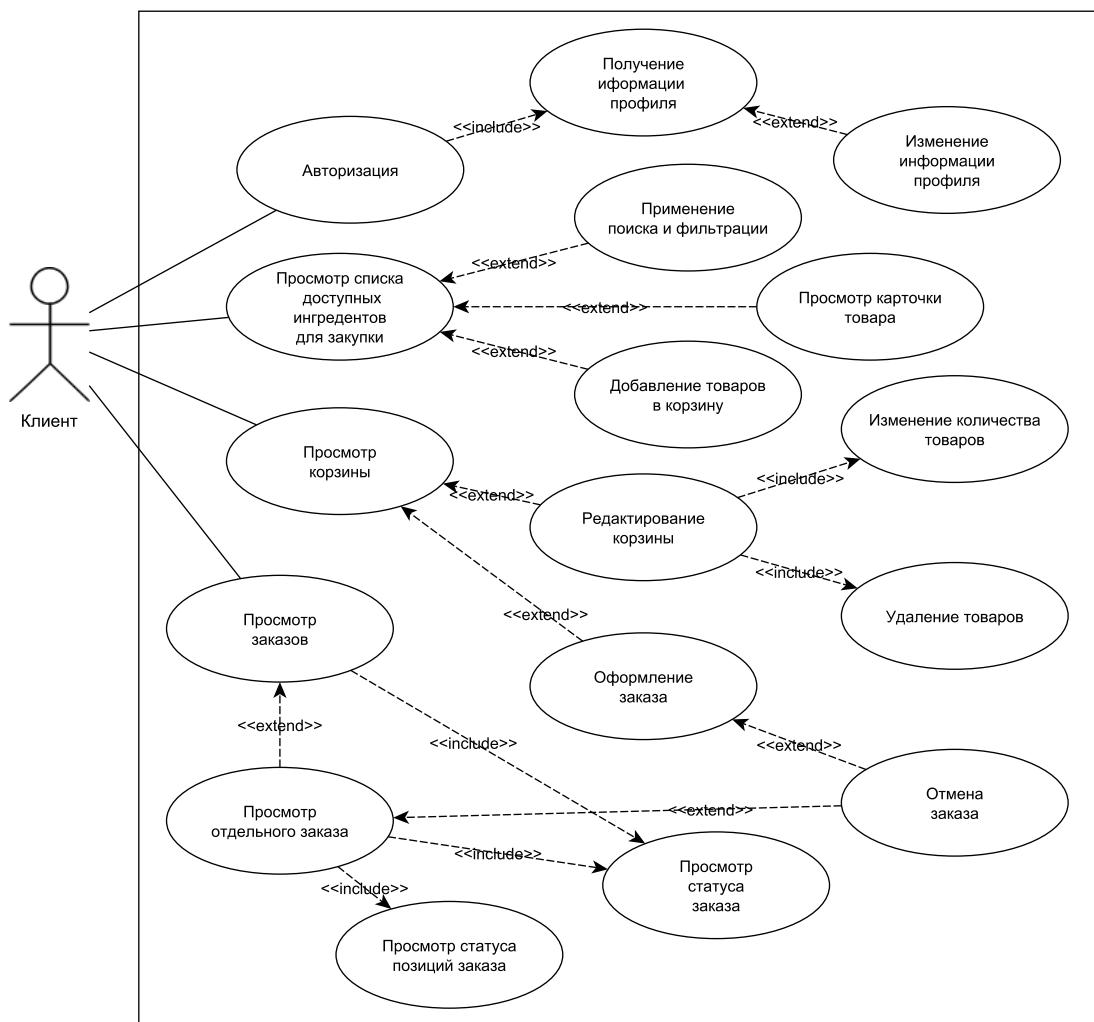


Рисунок 6 – Диаграмма прецедентов с точки зрения клиента

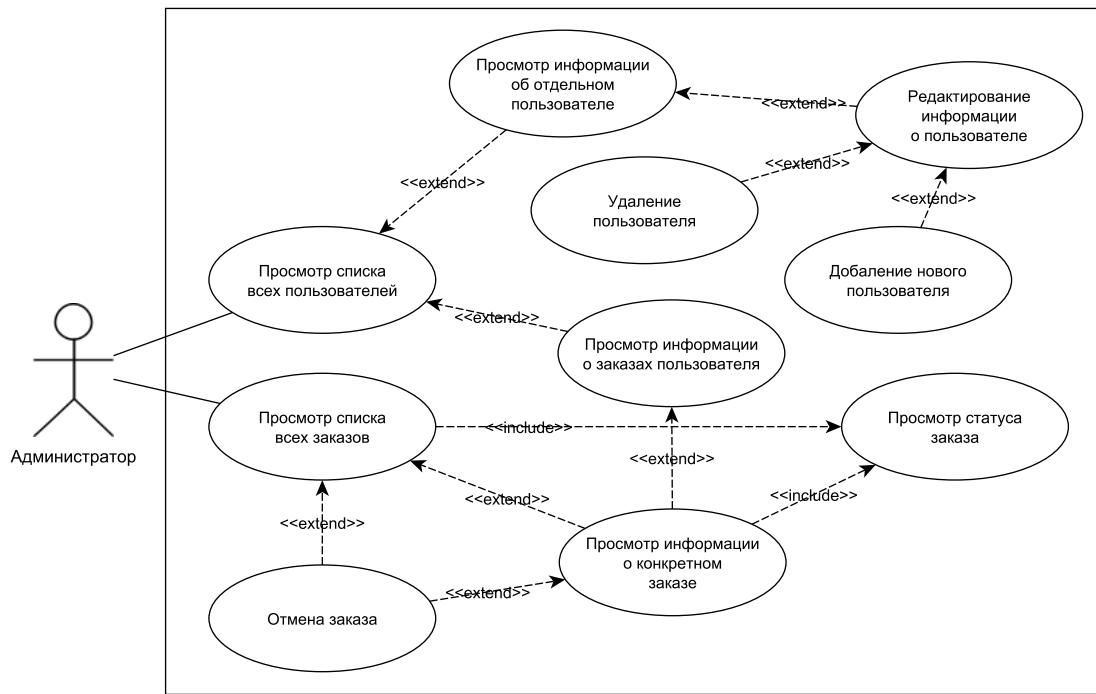


Рисунок 7 – Диаграмма прецедентов с точки зрения администратора

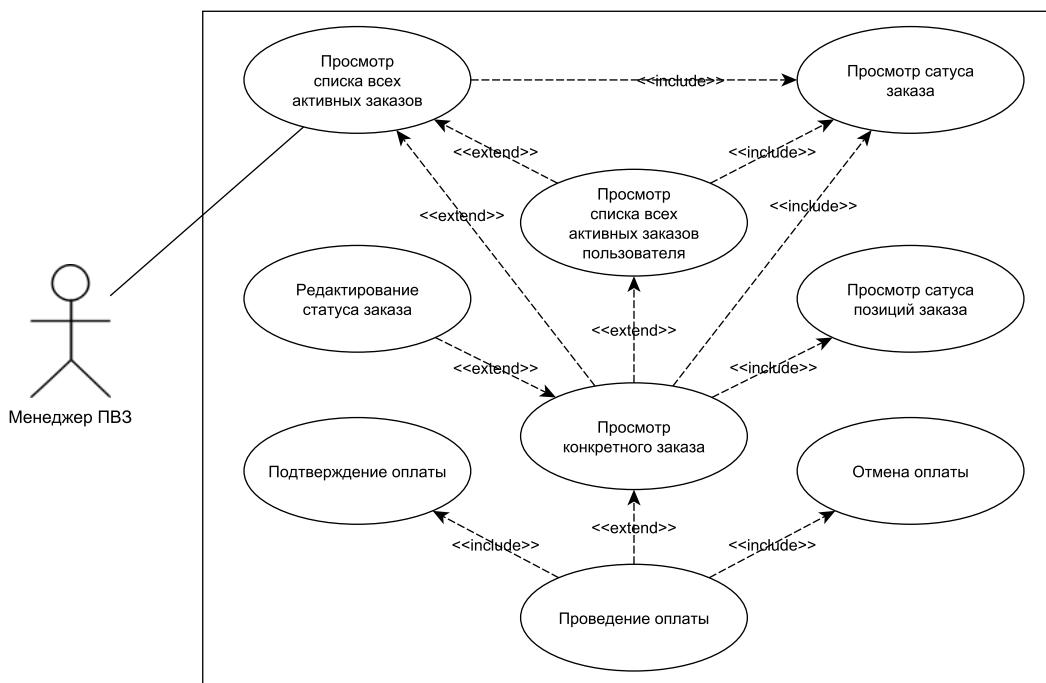


Рисунок 8 – Диаграмма прецедентов с точки зрения менеджера ПВЗ

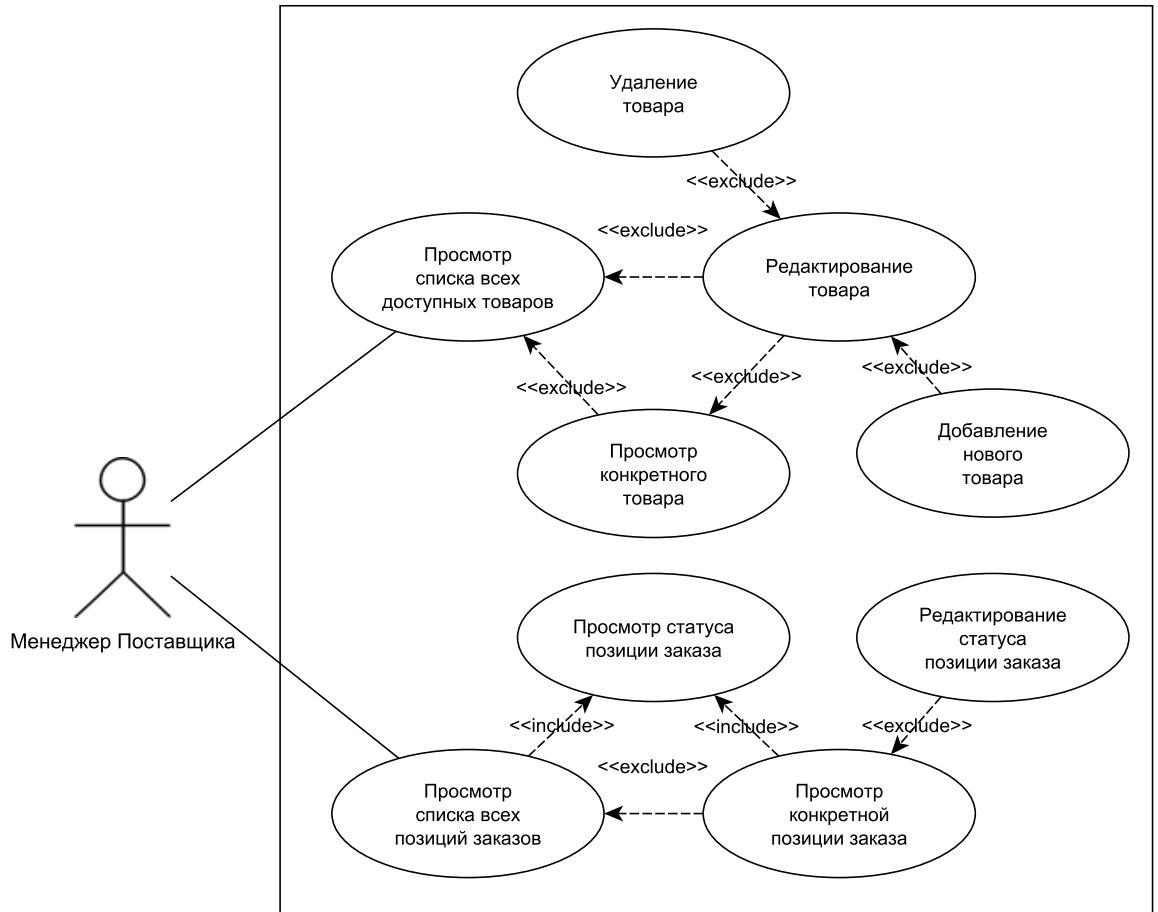


Рисунок 9 – Диаграмма прецедентов с точки зрения менеджера поставщика.

## Спецификации сценариев

Ниже, в таблицах 9-XXX приведены спецификации основных сценариев.

Таблица 9 – Спецификация сценария регистрации

Нормальный ход сценария	
Действия пользователя	Отклик системы
Пользователь нажимает кнопку «Зарегистрироваться»	Система предоставляет пользователю форму для регистрации, в которой нужно заполнить ФИО, дату рождения, логин, пароль, номер телефона, электронную почту

*Продолжение на следующей странице*

<b>Действия пользователя</b>	<b>Отклик системы</b>
Пользователь заполняет форму и даёт согласие на обработку данных нажав на кнопку «Подтвердить»	Пользователей с указанным логином/номером телефона/электронной почтой в клиентской базе не найдены. Данные пользователя регистрируются в системе. Пользователь перенаправляется на главную страницу (все дальнейшие действия под учетной записью зарегистрировавшегося пользователя)
<b>Альтернативный ход сценария</b>	
Пользователь нажимает кнопку «Зарегистрироваться»	Система предоставляет пользователю форму для регистрации, в которой нужно заполнить ФИО, дату рождения, логин, пароль, номер телефона, электронную почту
Пользователь заполняет форму и даёт согласие на обработку данных нажав на кнопку «Подтвердить»	Пользователь с указанным логином/номером телефона/электронной почтой уже есть в клиентской базе. Пользователю предлагается перейти на страницу авторизации
<b>Альтернативный ход сценария</b>	
Пользователь нажимает кнопку «Зарегистрироваться»	Система предоставляет пользователю форму для регистрации, в которой нужно заполнить ФИО, дату рождения, логин, пароль, номер телефона, электронную почту
Пользователь заполняет форму и даёт согласие на обработку данных нажав на кнопку «Подтвердить», но оставляет хотя бы одно необходимое для заполнения поле пустым	Пользователю предлагается заполнить необходимые для заполнения поля.

Таблица 10 – Спецификация сценария авторизации

<b>Нормальный ход сценария</b>	
<b>Действия пользователя</b>	<b>Отклик системы</b>
Пользователь нажимает кнопку «Войти»	Система предоставляет пользователю форму для авторизации, в которой нужно заполнить логин и пароль
Пользователь заполняет форму и нажимает кнопку «Войти»	Пользователь с указанным логином и паролем найден в клиентской базе. Пользователь перенаправляется на главную страницу (все дальнейшие действия под учетной записью авторизованного пользователя)
<b>Альтернативный ход сценария</b>	
Пользователь нажимает кнопку «Войти»	Система предоставляет пользователю форму для регистрации, в которой нужно заполнить ФИО, дату рождения, логин, пароль, номер телефона, электронную почту
Пользователь заполняет форму и нажимает кнопку «Войти»	Пользователь с указанным логином и паролем в клиентской базе не найден. Пользователю предлагается заполнить поля верно или пройти процедуру регистрации
<b>Альтернативный ход сценария</b>	
Пользователь нажимает кнопку «Войти»	Система предоставляет пользователю форму для регистрации, в которой нужно заполнить ФИО, дату рождения, логин, пароль, номер телефона, электронную почту
Пользователь заполняет форму и нажимает кнопку «Войти», но оставляет хотя бы одно из полей пустым	Пользователю предлагается заполнить необходимые для заполнения поля

Таблица 11 – Спецификация сценария просмотра товаров

<b>Нормальный ход сценария</b>	
<b>Действия клиента</b>	<b>Отклик системы</b>
Просмотр списка всех товаров, доступных для покупки	Система предоставляет клиенту список всех товаров, доступных для покупки
Клиент настраивает параметры поиска и фильтрации и нажимает кнопку «Обновить»	Система предоставляет клиенту список товаров, доступных для покупки, с учетом параметров поиска и фильтрации, указанные клиентом
Клиент выбирает понравившийся товар и нажимает на его карточку	Система предоставляет подробную информацию о выбранном товаре
<b>Альтернативный ход сценария</b>	
Просмотр списка всех товаров, доступных для покупки	Система предоставляет клиенту список всех товаров, доступных для покупки
Клиент настраивает параметры поиска и фильтрации и нажимает кнопку «Обновить»	Ни один из имеющихся товаров не соответствует параметрам поиска и фильтрации, указанным клиентом. Выводится сообщение «По вашему запросу ничего не найдено»

Таблица 12 – Спецификация сценария добавления товаров в корзину

<b>Нормальный ход сценария</b>	
<b>Действия клиента</b>	<b>Отклик системы</b>
Клиент выбирает товар, который хочет поместить в корзину (из списка товаров или зайдя в карточку товара) и нажимает кнопку «Добавить в корзину»	Система добавляет одну единицу товара в корзину пользователя, а кнопка «Добавить в корзину» заменяется на панель < Кнопка «–» Поле <кол-во товара в корзине> Кнопка «+» >

*Продолжение на следующей странице*

<b>Действия клиента</b>	<b>Отклик системы</b>
Клиент настраивает необходимое кол-во товара нажатиями на кнопки «+» или «-», или введя с клавиатуры необходимое значение в поле посередине и нажав клавишу «ENTER»	Система редактирует количество товара в корзине с учетом значения поля между кнопками «+» и «-», а также редактирует значение поля на странице с учетом манипуляций клиента
<b>Альтернативный ход сценария</b>	
Клиент выбирает товар, который хочет поместить в корзину (из списка товаров или зайдя в карточку товара) и нажимает кнопку «Добавить в корзину»	Система добавляет одну единицу товара в корзину пользователя, а кнопка «Добавить в корзину» заменяется на панель < Кнопка «-» Поле <кол-во товара в корзине> Кнопка «+» >
Клиент настраивает необходимое кол-во товара нажатиями на кнопки «+» или «-», или введя с клавиатуры необходимое значение в поле посередине и нажав клавишу «ENTER» (С учетом манипуляций клиента кол-во товара стало меньше либо равным нулю)	Система полностью удаляет товар из корзины, панель < Кнопка «-» Поле <кол-во товара в корзине> Кнопка «+» > заменяется на кнопку «Добавить в корзину»
<b>Альтернативный ход сценария</b>	
Клиент выбирает товар, который хочет поместить в корзину (из списка товаров или зайдя в карточку товара) и нажимает кнопку «Добавить в корзину»	Система добавляет одну единицу товара в корзину пользователя, а кнопка «Добавить в корзину» заменяется на панель < Кнопка «-» Поле <кол-во товара в корзине> Кнопка «+» >

*Продолжение на следующей странице*

<b>Действия клиента</b>	<b>Отклик системы</b>
Клиент настраивает необходимое кол-во товара нажатиями на кнопки «+» или «-», или введя с клавиатуры необходимое значение в поле посередине и нажав клавишу «ENTER» (С учетом манипуляций клиента кол-во товара стало равно количеству товара возможному для заказа)	Система редактирует количество товара в корзине с учетом значения поля между кнопками «+» и «-», а также редактирует значение поля на странице с учетом манипуляций клиента, при этом кнопка «+» на панели блокируется
<b>Альтернативный ход сценария</b>	
Клиент выбирает товар, который хочет поместить в корзину (из списка товаров или зайдя в карточку товара) и нажимает кнопку «Добавить в корзину»	Система добавляет одну единицу товара в корзину пользователя, а кнопка «Добавить в корзину» заменяется на панель < Кнопка «-»> Поле <кол-во товара в корзине> Кнопка «+»>
Клиент настраивает необходимое кол-во товара нажатиями на кнопки «+» или «-», или введя с клавиатуры необходимое значение в поле посередине и нажав клавишу «ENTER» (В поле кол-во товаров клиент вводит большее число чем доступно для заказа)	Система выдает сообщение «Невозможно добавить в корзину больше <макс. кол-во товара доступное к покупке>!», а также редактирует количество товара на <макс. кол-во товара доступное к покупке>, а также редактирует значение поля на странице на <макс. кол-во товара доступное к покупке>. Кнопка «+» на панели блокируется

Таблица 13 – Спецификация сценария оформления заказа

<b>Нормальный ход сценария</b>	
<b>Действия клиента</b>	<b>Отклик системы</b>
Клиент нажимает на кнопку «Корзина»	Система перенаправляет клиента на страницу просмотра корзины и предоставляет список товаров в корзине и кол-во каждой из позиций для заказа с возможностью изменения количества товара при помощи панели (см. <i>Спецификация сценария добавления товара в корзину</i> )
Клиент нажимает на кнопку «Оформить заказ»	Система перенаправляет клиента на страницу оформления заказа и предоставляет окончательный список товаров в корзине и кол-во каждой из позиций для заказа. Система сверяет кол-во товаров к заказу с кол-вом доступным к заказу на данный момент времени. Если кол-во товара к заказу не превышает допустимое – появляется кнопка «Подтвердить»
Клиент нажимает на кнопку «Подтвердить»	Система вносит заказ в базу, корзина клиента очищается
<b>Альтернативный ход сценария</b>	
Клиент нажимает на кнопку «Корзина»	Система перенаправляет клиента на страницу просмотра корзины. В корзине нет товаров. Выводится сообщение «Корзина пуста». Кнопка «Оформить заказ заблокирована»
<b>Альтернативный ход сценария</b>	
<i>Продолжение на следующей странице</i>	

<b>Действия клиента</b>	<b>Отклик системы</b>
Клиент нажимает на кнопку «Корзина»	Система перенаправляет клиента на страницу просмотра корзины и предоставляет список товаров в корзине и кол-во каждой из позиций для заказа с возможностью изменения количества товара при помощи панели (см. <i>Спецификация сценария добавления товара в корзину</i> )
Клиент нажимает на кнопку «Оформить заказ»	Система перенаправляет клиента на страницу оформления заказа и предоставляет окончательный список товаров в корзине и кол-во каждой из позиций для заказа. Система сверяет кол-во товаров к заказу с кол-вом доступным к заказу на данный момент времени. Если кол-во товара в корзине превышает допустимое, выводится сообщение «Невозможно оформить заказ. Превышено допустимое к заказу кол-во товара» и перенаправляет клиента на страницу просмотра корзины для редактирования кол-ва товара для дальнейшего заказа.
<b>Альтернативный ход сценария</b>	
Клиент нажимает на кнопку «Корзина»	Система перенаправляет клиента на страницу просмотра корзины и предоставляет список товаров в корзине и кол-во каждой из позиций для заказа с возможностью изменения количества товара при помощи панели (см. <i>Спецификация сценария добавления товара в корзину</i> )

*Продолжение на следующей странице*

<b>Действия клиента</b>	<b>Отклик системы</b>
Клиент нажимает на кнопку «Оформить заказ»	Система перенаправляет клиента на страницу оформления заказа и предоставляет окончательный список товаров в корзине и кол-во каждой из позиций для заказа. Система сверяет кол-во товаров к заказу с кол-вом доступным к заказу на данный момент времени. Если кол-во товара к заказу не превышает допустимое – появляется кнопка «Подтвердить»
Клиент нажимает на кнопку «Подтвердить»	По каким-либо причинам заказ не вносится в базу. Появляется сообщение «Ошибка оформления заказа. Попробуйте позже». Клиент перенаправляется на страницу просмотра корзины

## Логический дизайн

На рисунке 11 представлена схема базы данных разрабатываемой системы для покупки ингредиентов для создания парфюмерии класса люкс.

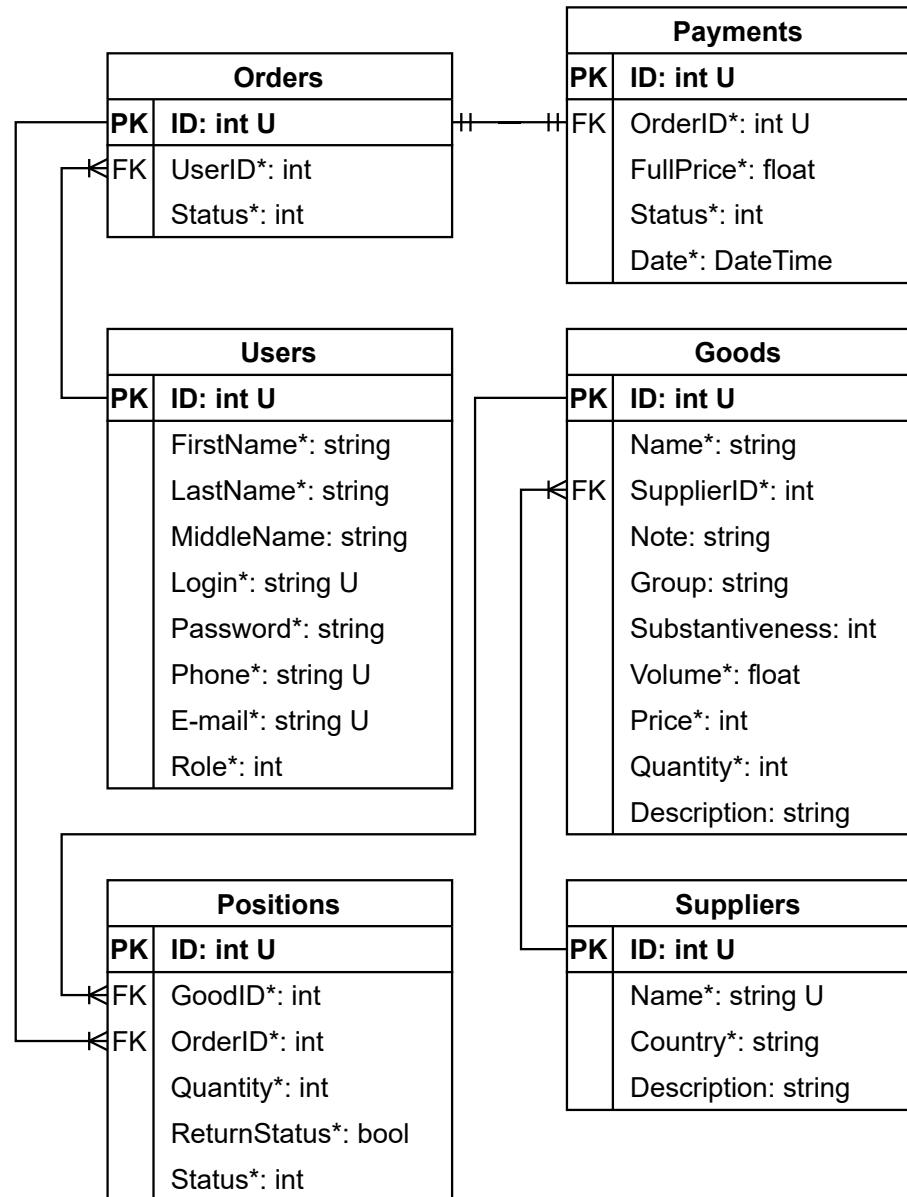


Рисунок 10 – Схема базы данных системы

## Спецификация классов

Ниже, на рисунке 11 представлена диаграмма классов разрабатываемой системы.

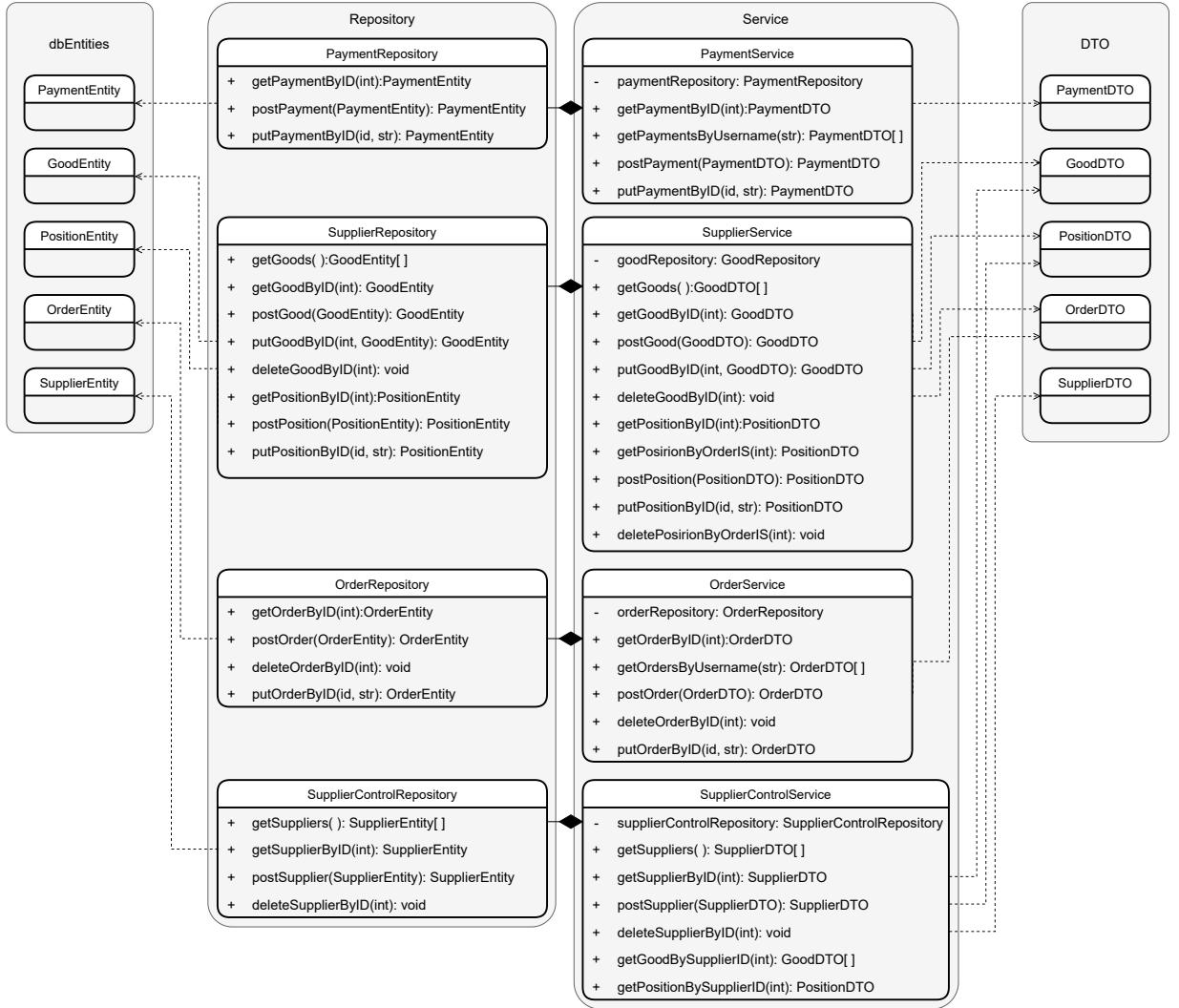


Рисунок 11 – Диаграмма классов

Классы типа *Entity* представляют легковесные объекты бизнес-логики, ассоциированные с соответствующими сущностями базы данных.

Классы типа *DTO* представляют объекты для передачи данных между классами бизнес-логики. Атрибутивный состав аналогичен составу ассоциированных с ними сущностей базы данных.

Классы типа *Repository* отвечают за взаимодействие микросервиса с базой данных.

Классы типа *Service* реализуют основную бизнес-логику микросервиса, преобразование данных и передачу их на последующий слой, непосредственно связанный с базой данных, поэтому предоставляемые ими методы схожи с методами классов типа *Repository*.

## Последовательность действий и потоки данных

На рисунке 12 представлена наиболее важная для системы динамическая модель, представленная в виде диаграммы последовательности действий.

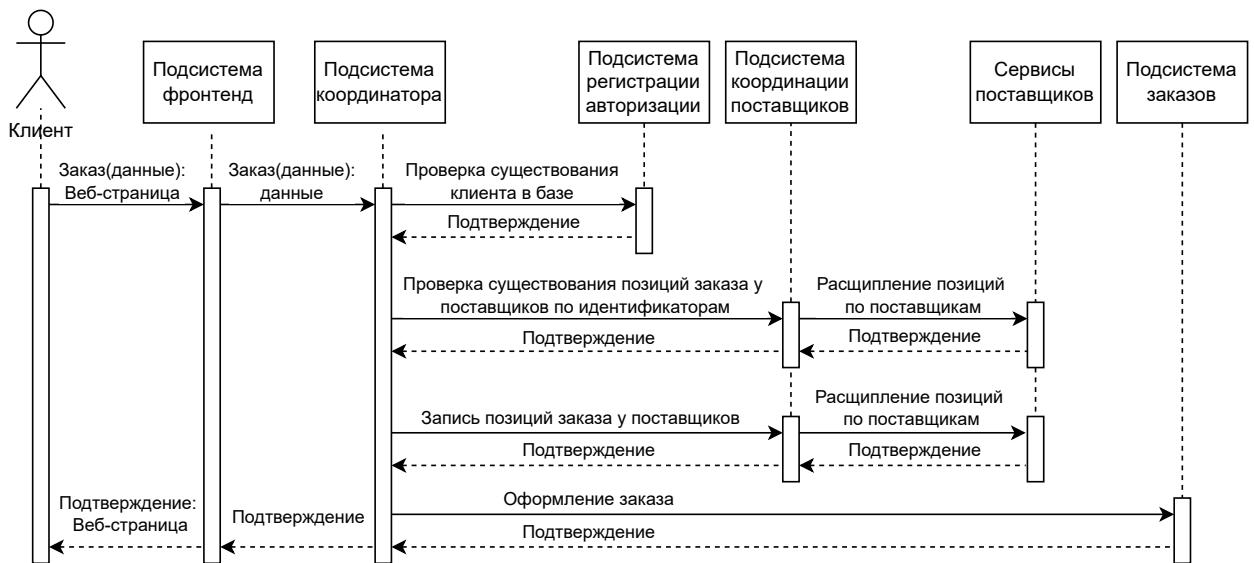


Рисунок 12 – Диаграмма последовательности действий при оформлении заказа

В системе предполагается распределенное хранение данных. Диаграмма потоков данных, представленная на рисунке 13, позволяет описать распределение сохраняемой информации в хранилищах данных. Каждое хранилище данных будет реализовано в виде базы данных.

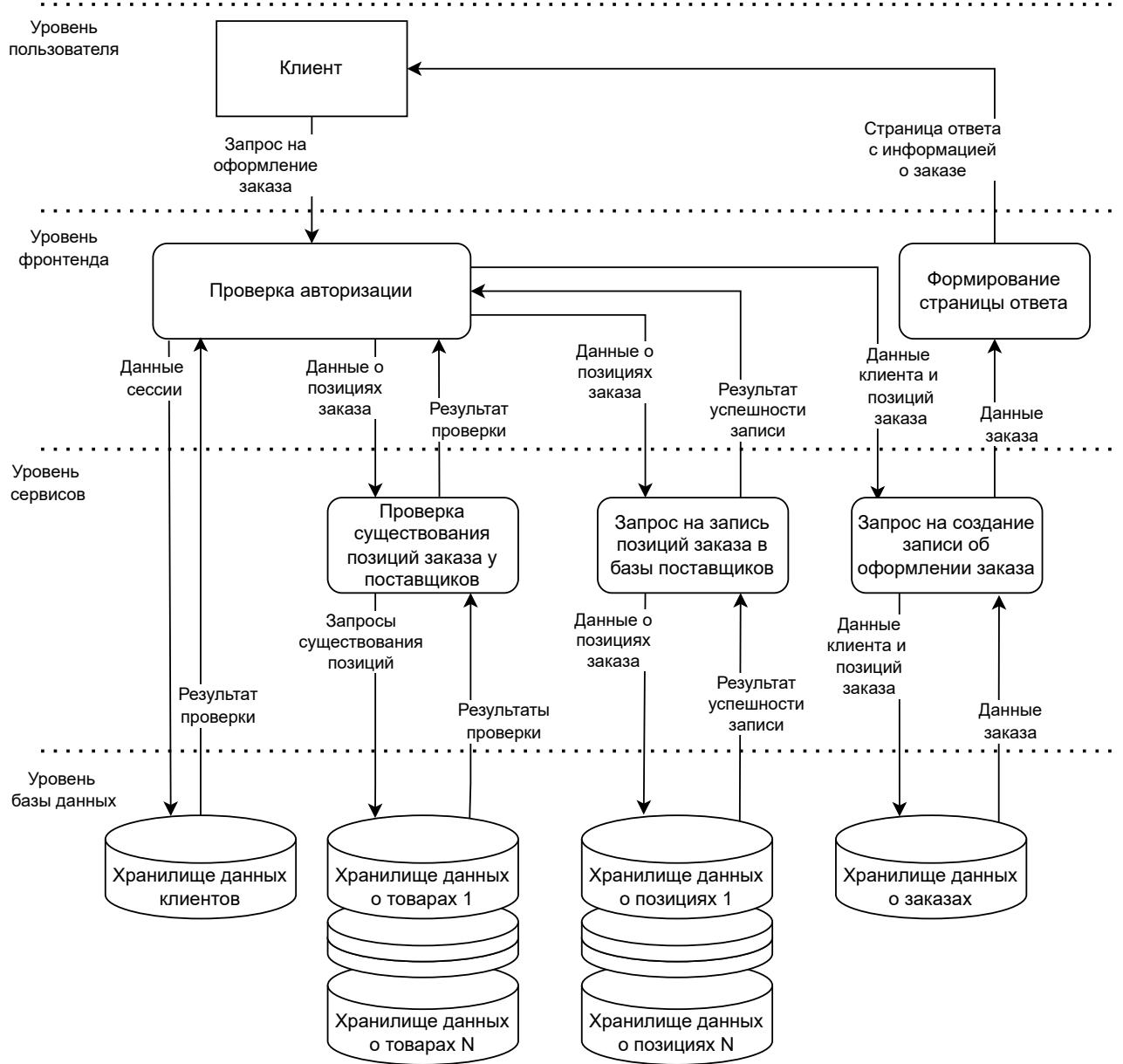


Рисунок 13 – Диаграмма потоков данных при оформлении заказа.

## **Высокоуровневый дизайн пользовательского интерфейса**

Пользовательский интерфейс в разрабатываемой системе представляет собой web-интерфейс, доступ к которому осуществляется через браузер (тонкий клиент).

Страница системы состоит из «шапки», основной части и футера. «Шапка» — это верхняя часть страницы, в которой находятся логотип и верхнее меню со ссылками на основные разделы портала. Футер — это нижняя часть страницы, в которой обычно размещают ссылки на редко посещаемые, но необходимые страницы, например, страницы с пользовательским соглашением.

Обобщенно структуру страниц системы можно представить следующим образом:

- главная страница со списком товаров и краткой информацией о каждом дом;
- страница авторизации;
- страница регистрации;
- страница с детальной информацией о выбранном товаре;
- страница корзины пользователя;
- страница с информацией по заказам пользователя;
- страница личного кабинета клиента системы;
- о проекте
  - контакты (администратор/ПВЗ);
  - время работы;
  - нормативные документы;
  - правила организации.

Ниже, на рисунках 14-17 приведены такие формы системы, как Главная страница, Страница авторизации, Страница регистрации, Страница с детальной информацией о выбранном товаре.

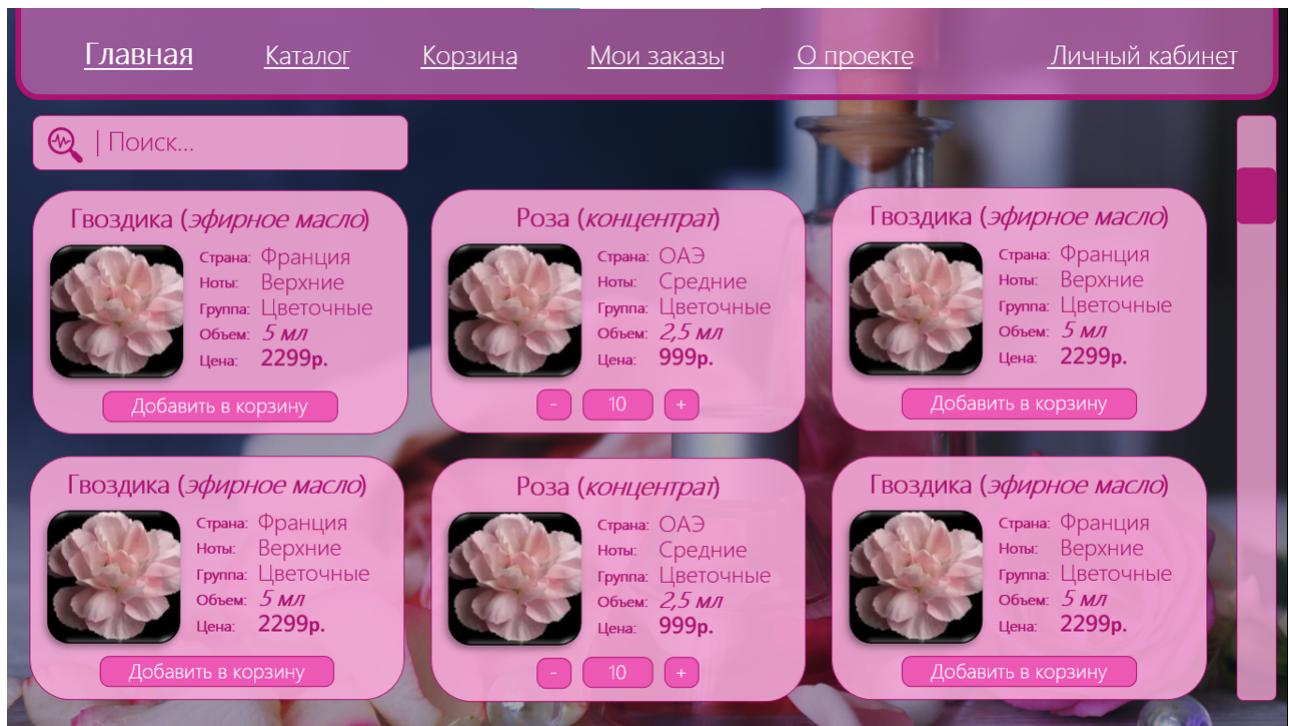


Рисунок 14 – Главная страница.

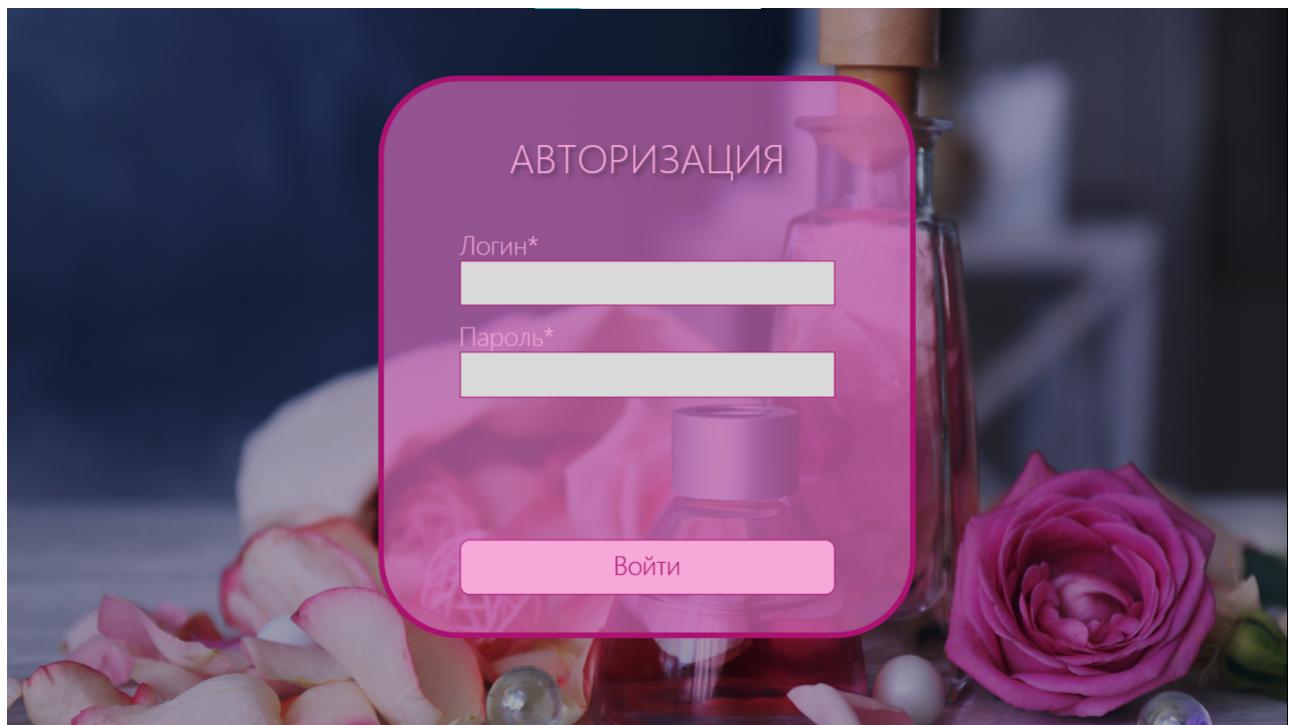


Рисунок 15 – Страница авторизации.

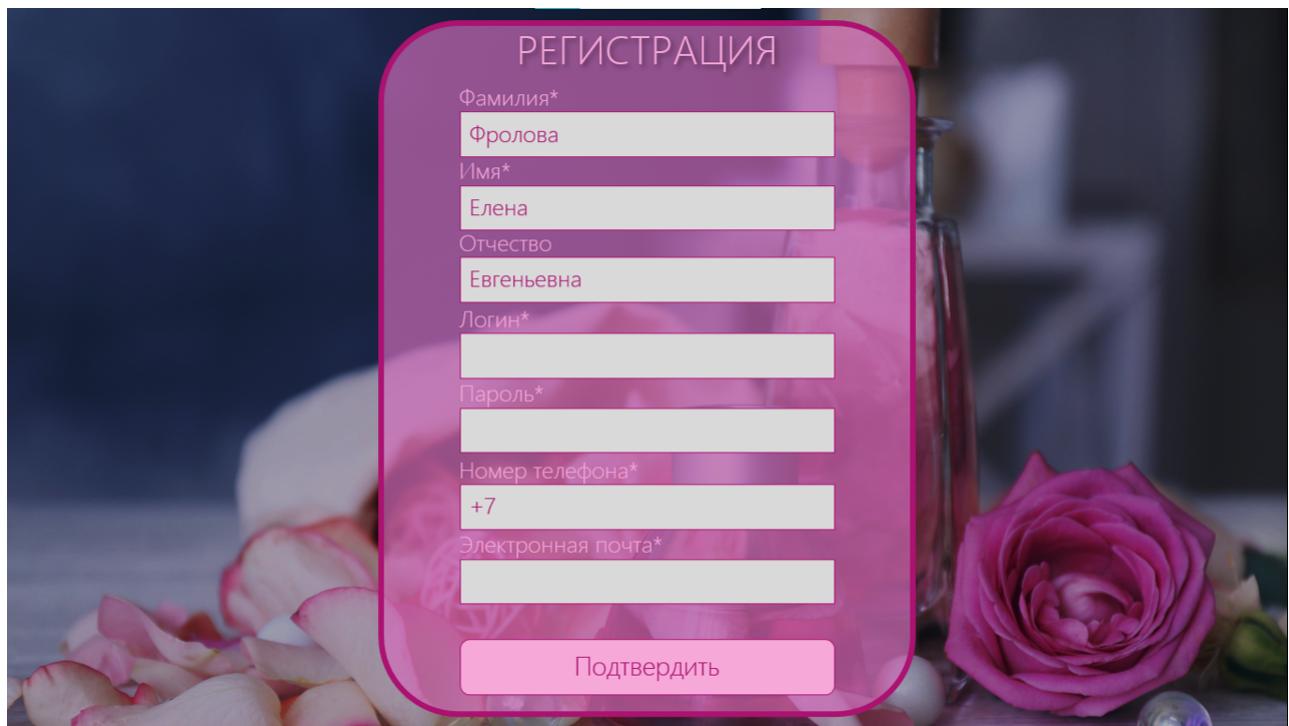


Рисунок 16 – Страница регистрации.

Главная Каталог Корзина Мои заказы О проекте Личный кабинет

### Лаванда (эфирное масло)



Страна: Франция, Прованс  
Ноты: Верхние  
Группа: Цветочные  
Объем: 5 мл  
Цена: 2299р.  
Производитель: *Histories' des fèves*  
Субстантивность: 8 часов на блоттере

Описание:

Эфирное масло лаванды настоящей аромат самосовершенствования и покоя, устраниет перевозбуждение, депрессию, плаксивость и истерические реакции. Сочетается с: апельсином, базиликом, гвоздикой, геранью, лимоном, можжевельником, мятой, фенхелем, эвкалиптом.

Добавить в корзину

Рисунок 17 – Страница с детальной информацией о выбранном товаре.

## Физический дизайн

### Выбор системы развертывания компонентов распределенной системы

Требования для системы развертывания компонентов.

- **Изоляция.** Компоненты не должны иметь доступа друг к другу, за исключением доступа, предусмотренного протоколом взаимодействия
- **Управление ресурсами.** Компонент должен потреблять ограниченное число ресурсов (процессорного времени, оперативной памяти, места на жестком диске).
- **Отслеживание зависимостей.** Компонент должен разворачиваться монолитно со всеми своими библиотеками. Это требование гарантирует, что компонент будет использоваться с теми же библиотеками, с которыми был протестирован.
- **Низкие накладные расходы.** Использование системы развертывания не должно нести высокие накладные расходы для распределенной системы.

Полностью удовлетворяет представленным требованиям такой инструмент визуализации, как программа *Docker* [3]. Согласно определению, Docker — программное обеспечение для автоматизации развертывания и управления приложениями в среде виртуализации на уровне операционной системы Linux. Позволяет «упаковать» приложение со всем его окружением в контейнер, который может быть перенесен на любую Linux-систему, а также предоставляет среду по управлению контейнерами. Разрабатывается и поддерживается одноименной компанией, распространяется как свободное программное обеспечение под лицензией Apache 2.0.

*Контейнером* называется образ изолируемой части Системы, содержащий приложение со всеми его зависимостями. В контейнере могут содержаться приложения и их среда выполнения: нужные библиотеки, конфигурационные файлы. Для облегчения работы используют единый репозиторий (хранилище) образов. Это позволяет применять уже готовые образы, которые

необходимо только доработать для нужной задачи. Docker эффективно используют для следующих задач:

- тестирование распределённой системы;
- отслеживание зависимостей между компонентами разрабатываемой системы.

Преимуществами Docker по сравнению с виртуальными машинами (Oracle VM VirtualBox, Microsoft Hyper-V) являются:

- низкие накладные расходы;
- наличие репозитория.

Использование программного обеспечения Docker позволит обеспечить высокую производительность при создании виртуальных вычислительных узлов и уменьшить потребление процессорных ресурсов. Недостатком Docker является то, что он работает только под ОС Linux.

## **Выбор операционной системы**

Согласно требованиям технического задания, разрабатываемый портал должен обладать высокой доступностью, работать на типичных архитектурах ЭВМ (Intel x86, Intel x64), а так же быть экономически недорогим для сопровождения. Таким образом, требования к ОС следующие.

- **Распространенность.** На рынке труда должно быть много специалистов, способных администрировать распределенную систему, работающую под управлением выбранной операционной системы.
- **Надежность.** Операционная система должна широко использоваться в стабильных проектах, таких как Mail.Ru, Vk.com, Google.com. Эти компании обеспечивают высокую работоспособность своих сервисов, и на их опыт можно положиться.
- **Наличие требуемого программного обеспечения.** Выбор операционной системы не должен ограничивать разработчиков в выборе программного обеспечения, библиотек.

- **Цена.**

Под данные требования лучше всего подходит ОС Ubuntu [4]. *Ubuntu* — это дистрибутив, использующий ядро Linux. Как и все дистрибутивы Linux, Ubuntu является ОС с открытым исходным кодом, бесплатным для использования. Поставляется как в клиентской (с графическим интерфейсом), так и в серверной (без графического интерфейса) версиями. Ubuntu поставляется с современными версиями ПО. Преимуществом Ubuntu являются низкие требования к квалификации системных администраторов. Однако Ubuntu менее стабильна в работе.

## Выбор СУБД

В соответствии с техническим заданием разработка бекенда предусматривает следующие требования.

- **Безопасность хранения данных.** Несанкционированный доступ к данным клиентам должен быть невозможен.
- **Транзакционность.** Должен соблюдаться принцип «ACID» (Atomicity — Атомарность, Consistency — Согласованность, Isolation — Изолированность, Durability — Надежность). Атомарность гарантирует, что транзакция не может быть зафиксирована частично. Согласованность — что успешное завершение транзакции оставит систему в согласованном состоянии. Изолированность — что параллельно выполняемые транзакции не будут влиять друг на друга. Надежность — что успешно завершенная транзакция будет зафиксирована, а в случае сбоя, после восстановления системы, результаты транзакции не будут потеряны.
- **Масштабируемость.** Выбранная СУБД должна поддерживать репликацию, шардирование.

PostgreSQL [5] — реляционная система управления базами данных. Она является некоммерческим ПО с открытым исходным кодом. Для работы с этой СУБД существуют библиотеки для таких распространенных языков программирования, как Python, Ruby, Perl, PHP, C, C++, Java, C#, Go. Она работает под управлением многих операционных систем: Linux, MacOS, Windows, FreeBSD, Solaris и др. По сравнению с MySQL система PostgreSQL

лучше работает с репликацией, так как в ней существует журнал (средство восстановления системы в случае сбоя) физической модификации страниц. PostgreSQL осуществляет асинхронную репликацию типа «ведущий — ведомый».

Выбор СУБД PostgreSQL для хранения данных разрабатываемой системы обеспечит надежность, безопасность и масштабируемость.

### **Выбор языка разработки компонент портала**

Исходя из приведенных требований к системе, можно выявить следующие требования к языку программирования.

- **Наличие разнообразных библиотек.** Использование готовых библиотек ускоряет разработку программного обеспечения. Также важно, что благодаря использованию распространенных оттестированных библиотек снижается вероятность ошибки. Это повышает надежность программного обеспечения.
- **Совместимость с выбранными ранее технологиями.** Выбранный язык должен уметь взаимодействовать с ОС Linux, СУБД PostgreSQL.
- **Высокая скорость разработки.** На ранних этапах разработки портала технические требования часто меняются. Язык программирования должен позволять как можно быстрее вносить изменения в коды программ.

Под данные требования хорошо подходит язык Java [6]. Это универсальный объектно-ориентированный язык со строгой типизацией. В нём реализован принцип WORA (от английского: write once, run anywhere). Это позволяет запускать приложения везде, где есть среда исполнения JRE (от английского: Java Runtime Environment). При этом не имеет значения, какая операционная система установлена на устройстве.

У Java есть ряд преимуществ.

- **Простота** – чёткие синтаксические правила и понятная семантика.
- **Объектно-ориентированный подход.**

- **Безопасность.** Обойти или взломать механизмы защиты крайне сложно.
- **Производительность.** Новые версии динамических компиляторов Java не уступают традиционным из других платформ. При необходимости те или иные приёмы оптимизации включаются или отменяются JIT-компилятором.
- **Надёжность.** Компилятор способен выявить ошибки ещё до выполнения кода, то есть на ранних стадиях.
- **Независимость от аппаратной части и ОС.** Важно лишь наличие исполняющей среды и JVM. Байт-код легко интерпретируется на любой машине. Кроссплатформенностью отличается также интерфейс, реализованный в системных библиотеках.
- **Динамичность и адаптируемость.** При необходимости можно добавить в библиотеки новые объекты, методы.
- **Удобные и эффективные сетевые возможности.** Приложения умеют находить объекты в сети и открывать к ним доступ. Предоставляется обширная программная библиотека для передачи данных по самым распространённым протоколам: FTP, HTTP, TCP/IP. Работает механизм вызова удалённых методов.

## Выбор фреймворка для разработки портала

К выбору фреймворка предъявляют те же требования, что и к выбору языка программирования:

- большое число стандартных возможностей;
- совместимость с выбранными ранее технологиями;
- высокая скорость разрабатываемого портала.

Для разработки «Распределённой системы бронирования номеров гостиниц сети Apartlux» должны быть использованы фреймворки, основанные на парадигме MVC:

- модель представляет объекты, хранимые в системе: информацию об аккаунтах, гостиницах, номерах, бронированиях;
- вид отвечает за визуализацию моделей;
- элемент управления отвечает за взаимодействие пользователя и программного обеспечения.

*Spring Boot* [7] — это фреймворк на основе Java с открытым исходным кодом. Благодаря быстродействию и простоте работы он стал популярным решением для создания развертываний в виде архива веб-приложений (WAR) и автономных Java-приложений.

Он позволяет избавиться от трудоемкой первоначальной установки и настройки среды развертывания. Основные преимущества:

- быстрая и легкая разработка приложений;
- автоконфигурация всех компонентов;
- готовые встроенные серверы, обеспечивающие ускоренное и более продуктивное развертывание приложений;
- отсутствие конфигурации XML;
- большой выбор плагинов, облегчающих работу со встроенными базами данных, легкий доступ к ним и службам очередей.
- подробная документация.

К недостаткам этого фреймворка относятся:

- создает множество неиспользуемых зависимостей, что приводит к большому размеру файла развертывания;
- не подходит для создания монолитных приложений.

Таким образом, в результате проведённого анализа в качестве системы развертывания компонентов был выбран Docker, ОС – Ubuntu, СУБД – PostgreSQL, язык программирования – Java, фреймворк – Spring Boot.

## **Обеспечение надежности портала**

В рассматриваемой «Распределённой системе покупки/продажи ингредиентов для создания парфюмерии класса люкс» для обеспечения надежности функционирования СУБД будет применяться репликация типа «ведущий – ведомый» и шардинг. Для обеспечения надежности данных СУБД необходимо разработать скрипт для автоматического создания резервной копии базы данных по расписанию.

Для фронтенда и бекендов целесообразно применить зеркалирование. Это обеспечит отказоустойчивость системы: в случае сбоя любого из ее узлов запросы на чтение данных будут выполняться.

Отказоустойчивость фронтенда возможно за счет его зеркалирования на несколько серверов. Выбор одного из зеркалируемых серверов осуществляется с помощью прописывания в DNS (система доменных имен) записях адресов всех серверов, на которых располагаются фронтенды.

Также должна быть настроена система мониторинга на предмет возникновения ошибок в системе, в случае их появления оповещение должно прийти в Telegram-канал для наиболее оперативного реагирования.

## **ЗАКЛЮЧЕНИЕ**

В ходе выполнения лабораторных работ по курсу «Методология программной инженерии» была разработана распределённая система закупки ингредиентов для создания парфюмерии класса люкс.

Перед началом разработки сформулировано техническое задание с общими и функциональными требованиями, а также требованиями к надежности и документации. Помимо требований к системе в техническом задании приведена топология системы, на основе которой разработаны требования к подсистемам.

## СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Анализ рынка парфюмерии в России (итоги первой половины 2021г.) [Электронный ресурс]. – Режим доступа: <https://drgroup.ru/components/com-jshopping/files/demo-products/Demo.Analiz-ryntka-parfyumerii-v-Rossii.pdf> (Дата обращения: 21.03.2024).
2. Производство туалетной воды как аналог бизнеса на дому [Электронный ресурс]. – Режим доступа: <https://hiterbober.ru/beginners/proizvodstvo-tualetnoj-vody.html> (Дата обращения: 21.03.2024).
3. Docker Documentation [Электронный ресурс]. – Режим доступа: <https://docs.docker.com/> (Дата обращения: 20.04.2023).
4. Ubuntu Documentation [Электронный ресурс]. – Режим доступа: <https://ubuntu.com/> (Дата обращения: 20.04.2023).
5. PostgreSQL Documentation [Электронный ресурс]. – Режим доступа: <https://www.postgresql.org/docs/> (Дата обращения: 20.04.2023).
6. Java Documentation [Электронный ресурс]. – Режим доступа: <https://docs.oracle.com/en/java/> (Дата обращения: 20.04.2023).
7. Spring Boot Documentation [Электронный ресурс]. – Режим доступа: <https://docs.spring.io/spring-boot/docs/current/reference/htmlsingle/> (Дата обращения: 20.04.2023).