**4 ПРОГРАММНАЯ ПОДДЕРЖКА ПРОЦЕССА ОФОРМЛЕНИЯ ЗАКАЗА В ОНЛАЙН-МАГАЗИНЕ САХАРНОЙ ПРОДУКЦИИ**

**4.1 Описание предметной области и основного процесса**

**предметной области**

В настоящее время заказы на продукцию сахарного завода оформляются с помощью веб-сайта и связи с менеджерами магазина завода. Сайт представляет собой информационный канал, посредством которого клиенты получают сведения о каталоге продукции завода. В частности: названия товаров, описания, пищевая ценность, а также информация об упаковках. Данная информация хранится в базе данных и контролируется менеджерами.

После выбора интересующего товара клиенты связываются с менеджерами сахарного завода с помощью различных каналов связи и оформляют заказ, указывая необходимые параметры заказа, оставляют данные для обратной связи и т.д.

Данный процесс был усовершенствован путём преобразования веб-сайта в интернет-магазин. Основная особенность и ключевой функционал заключается в возможности для клиентов совершать заказы онлайн напрямую через каталог и использовать собственный личный кабинет. Это позволяет значительно повысить удобство для клиентов, расширить возможности по взаимодействию с персоналом сахарного завода и повысить конкурентоспособность компании на рынке.

После оформления заказа он передаётся на обработку менеджерам сахарного завода. Они проводят мониторинг заказов, контроль их жизненных циклов, управление статусами. Все изменения относительно своего заказа менеджеры видят в личном кабинете.

Для использования всех возможностей сайта клиенту необходимо вначале пройти процедуру регистрации. В рамках регистрации клиент указывает личные данные и сведения о фирме, представителем которой он является. После создания личного кабинета клиент может оформлять заказы и контролировать их.

Для представления процесса оформления заказа была разработана

функциональная модель приложения IDEF0.

IDEF0 – метод функционального моделирования, а также графическая нотация, которая используется для описания и формализации бизнес-процессов. Особенность IDEF0 заключается в том, что эта методология ориентирована на соподчиненность объектов.

Главным процессом предметной области является оформление заказа. Контекстная диаграмма верхнего уровня (рисунок 4.1) иллюстрирует

функциональную модель «Оформить заказ».

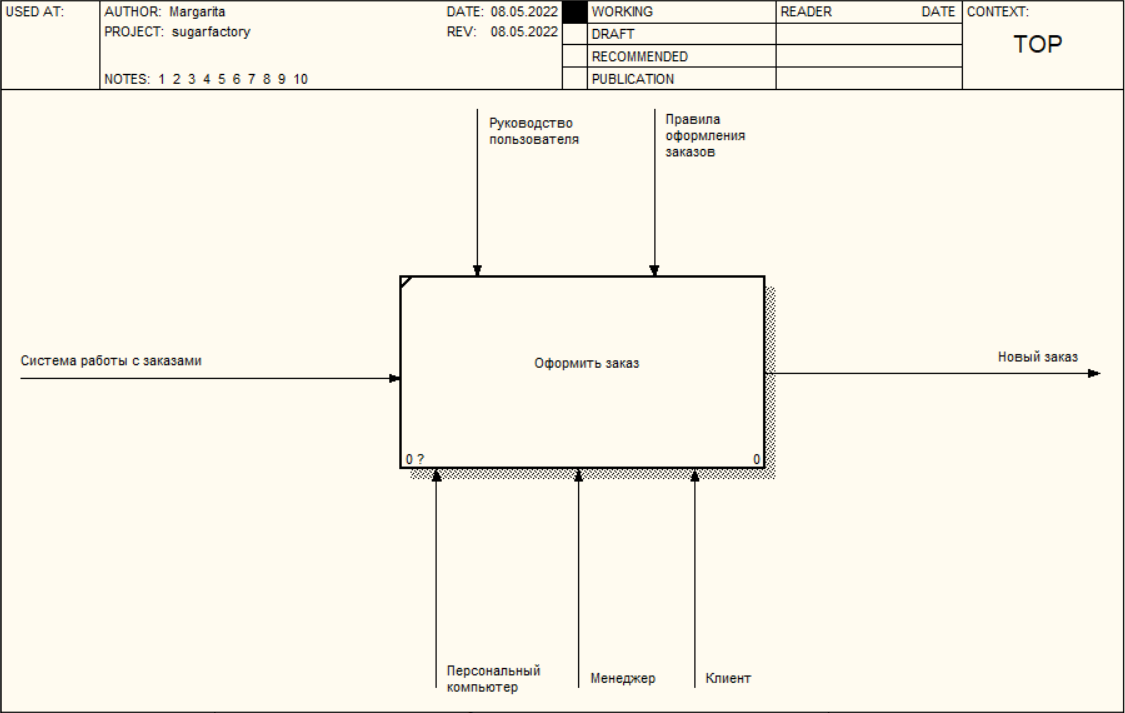


Рисунок 4.1 – Контекстная диаграмма

В качестве входных данных на диаграмме выступает система работы с заказами. Управлениями служат руководство пользователя и правила оформления заказов. Механизмы в системе – клиент, менеджер и персональный компьютер. В результате процесса формируется новый заказ.

Процесс оформления заказа декомпозируется на следующие процессы (рисунок 4.2):

- Войти в аккаунт;

- Перейти в каталог;

- Указать данные заказа.

Декомпозиция процесса «Указать данные заказа» представлена на рисунке 4.3. Она представляет собой следующие этапы:

- Выбрать товар;

- Указать количество;

- Выбрать упаковку.

Декомпозиция процесса «Выбрать упаковку» разделяется на следующие этапы:

- Получить список упаковок;

- Указать упаковку.

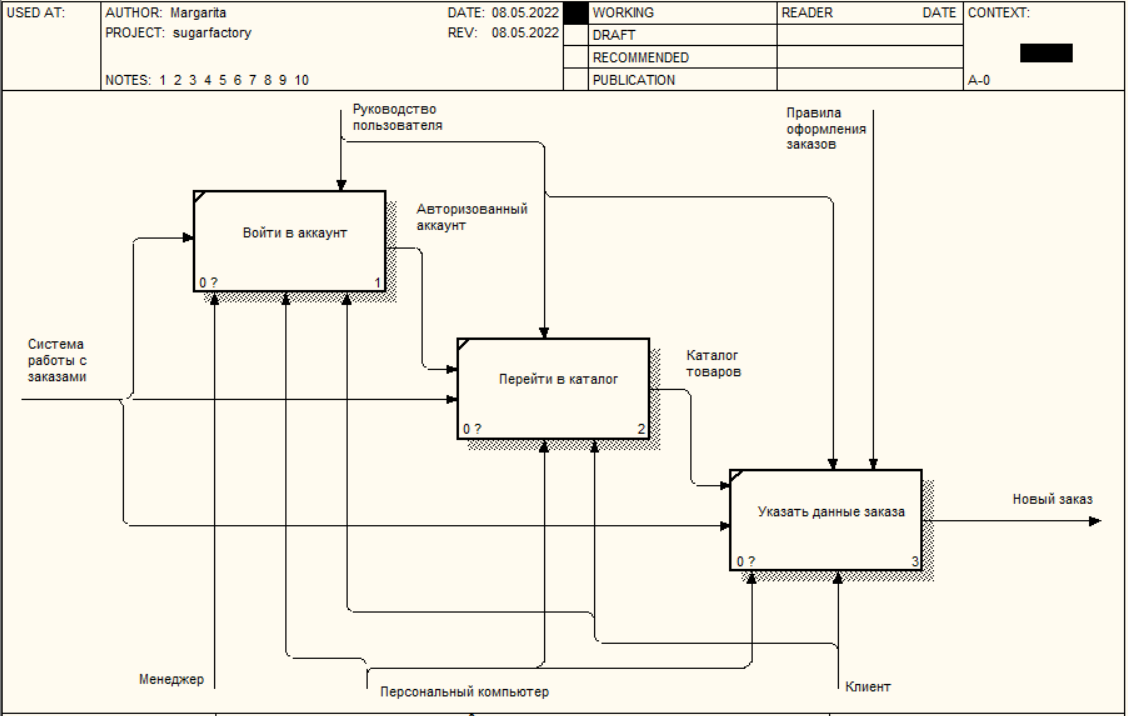


Рисунок 4.2 – Декомпозиция контекстной диаграммы

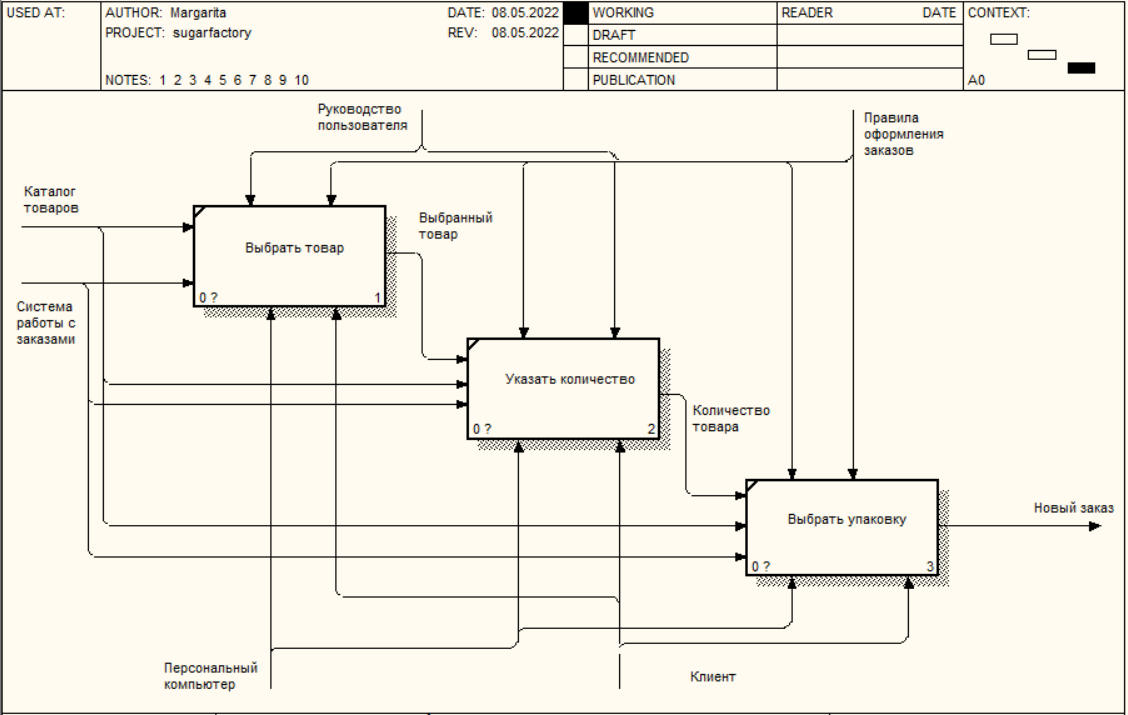


Рисунок 4.3 – Декомпозиция блока «Указать данные заказа»

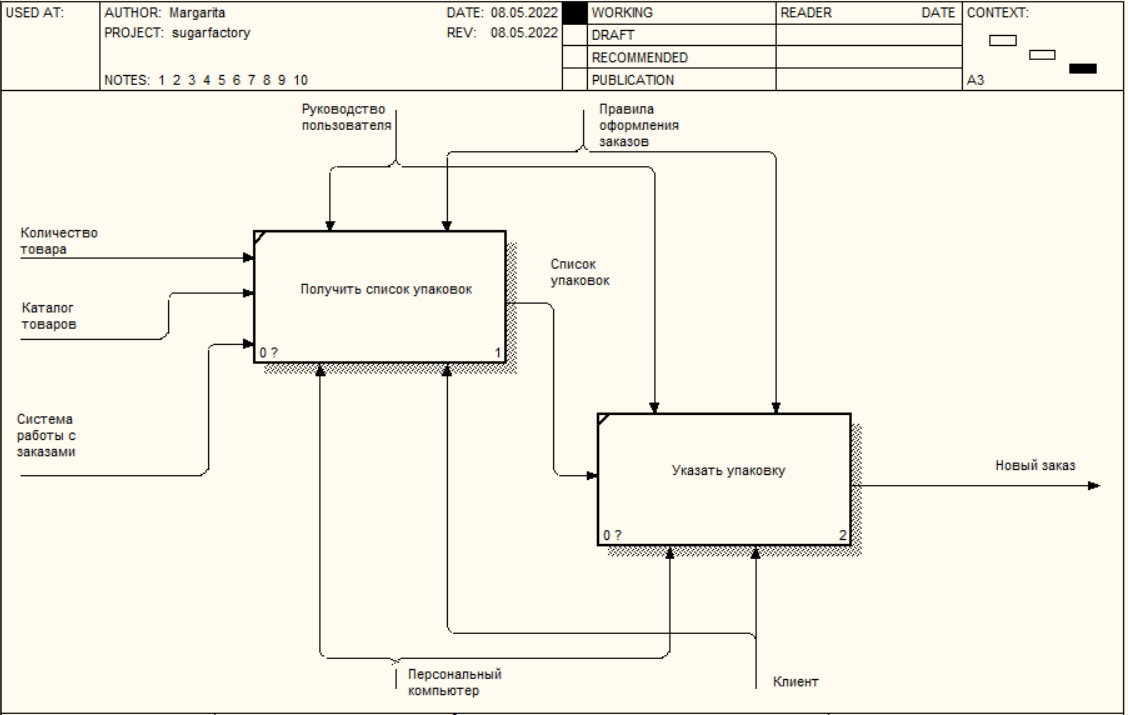


Рисунок 4.4 – Декомпозиция блока «Выбрать упаковку»

**4.2 Информационная модель системы и ее описание**

Информационная модель является представлением следующих элементов: сущности, атрибуты и связи. В базе данных сущности соответствует таблица, экземпляру сущности – строка (запись) в таблице, а атрибуту – колонка (поле) таблицы.

Процесс построения информационной модели состоит из следующих

шагов:

- определение сущностей;

- определение зависимостей между сущностями;

- задание первичных и альтернативных ключей;

- определение атрибутов сущностей;

- приведение модели к требуемому уровню нормальной формы;

- генерация базы данных.

Информационная модель системы представлена на рисунке 4.5.

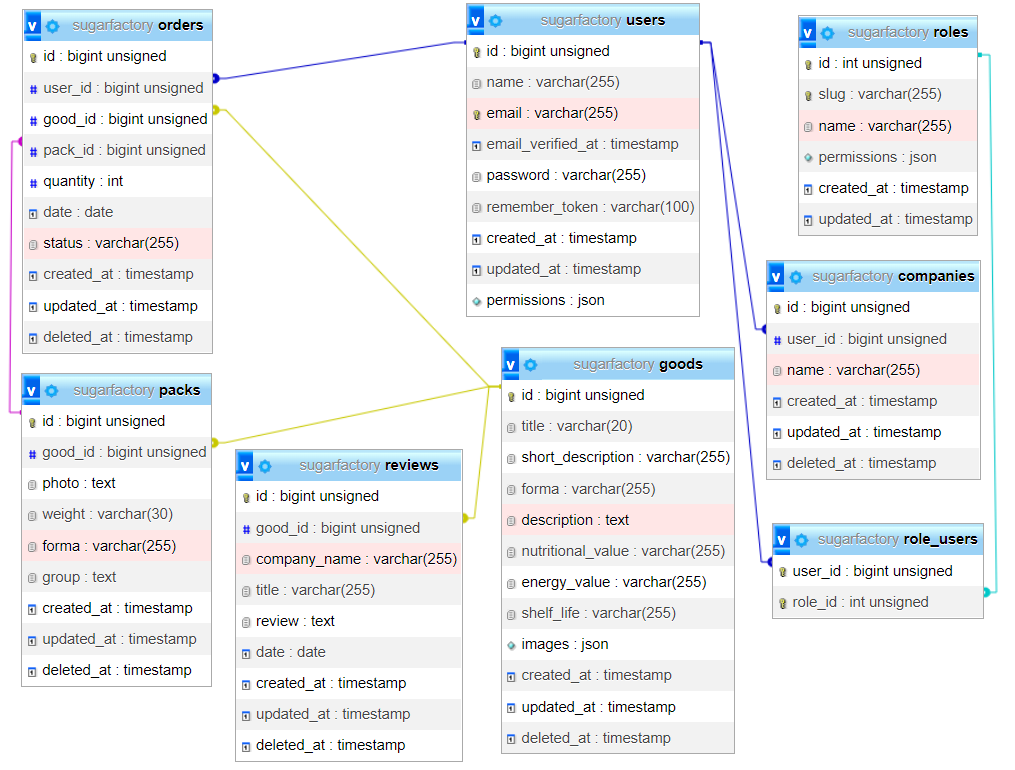


Рисунок 4.5 – Информационная модель системы

Далее сведения о сущностях и атрибутах в системе:

1) Сущность Goods представляет собой сущность товара в системе. Содержит следующие поля: title (название товара), short\_description (краткое описание товара), forma (форма товара), description (описание товара), nutritional\_value (пищевая ценность), energy\_value (энергетическая ценность), shelf\_life (срок годности), images (фото товара), а также поля created\_at, updated\_at, deleted\_at – поля дат создания, редактирования и удаления записи. Первичным ключом является поле id.

2) Сущность Reviews представляет собой сущность отзыва о товаре от клиента. Имеет следующие поля: good\_id (внешний ключ, таблица товара), company\_name (название компании), title (заголовок отзыва), review (текст отзыва), date (дата отзыва), а также поля created\_at, updated\_at, deleted\_at, аналогично с прошлой сущностью.

3) Сущность Packs является сущностью упаковки товара. Имеет следующие поля: good\_id (внешний ключ, таблица товара), photo (путь к фото упаковки в системе), weigth (вес упаковки), forma (форма), group (групповая упаковка), а также поля created\_at, updated\_at, deleted\_at.

4) Сущность Orders представляет собой сущность заказа. Имеет следующий набор полей: user\_id (внешний ключ, таблица пользователей), good\_id (внешний ключ, таблица товара), pack\_id (внешний ключ, таблица упаковок), quantity (количество), date (дата), status (статус заказа), а также поля created\_at, updated\_at, deleted\_at.

5) Сущность Users хранит в себе данные о пользователях в системе. Поля: name (имя пользователя), email (электронная почта), email\_verified\_at (дата подтверждения email), password (хеш пароля), remember\_token (токен для авторизации), поля created\_at, updated\_at, а также permissions (список разрешений для пользователя). Первичным ключом является id.

6) Сущность Roles – сущность роли пользователя в системе. Поля: slug (слаг роли), name (название роли), permissions (разрешения, предусмотренные для данной роли), поля created\_at и updated\_at. Первичным ключом выступает поле id.

7) Сущность role\_users – сущность связи для таблиц пользователей и ролей. Полями являются внешние ключи – id записи пользователя и роли.

8) Companies – сущность, представляющая компанию, связанную с пользователем. Поля: user\_id (внешний ключ, таблица пользователей), name (название компании), а также поля created\_at, updated\_at, deleted\_at, аналогично с предыдущими сущностями.

**4.3 Спецификация вариантов использования системы**

Use-case диаграмма – это диаграмма, отражающая отношения между акторами и прецедентами и являющаяся составной частью модели прецедентов, позволяющей описать систему на концептуальном уровне. В разработанной системе присутствуют 3 роли – клиент, менеджер и гость. Диаграммы вариантов использования для данных ролей представлены на рисунках 4.6, 4.7 и 4.8.

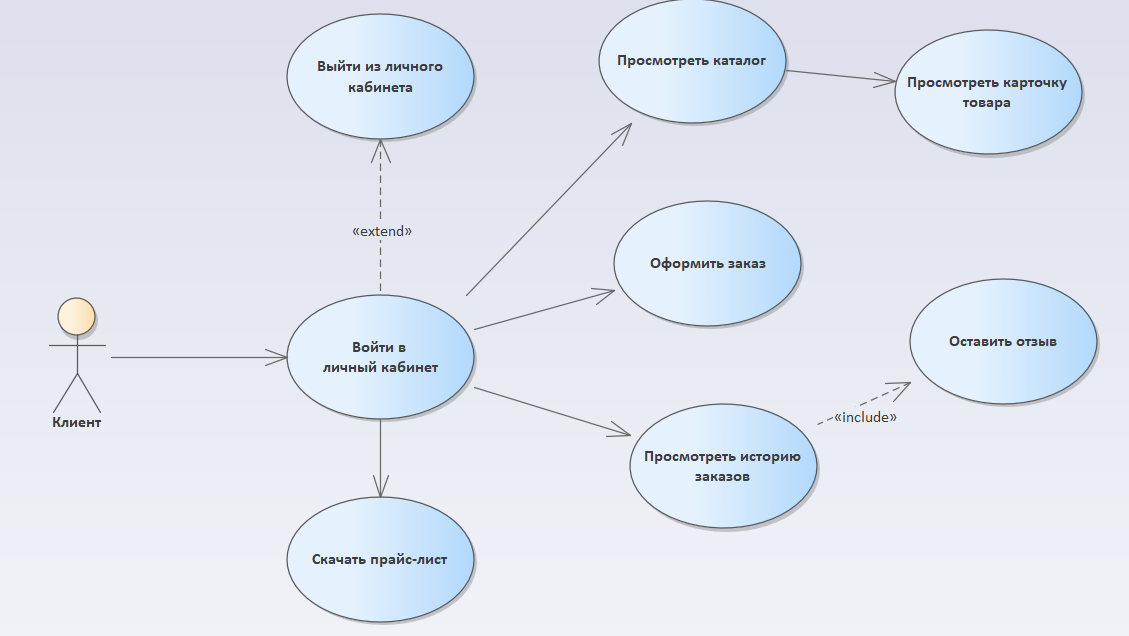


Рисунок 4.7 – Диаграмма вариантов использования для клиента

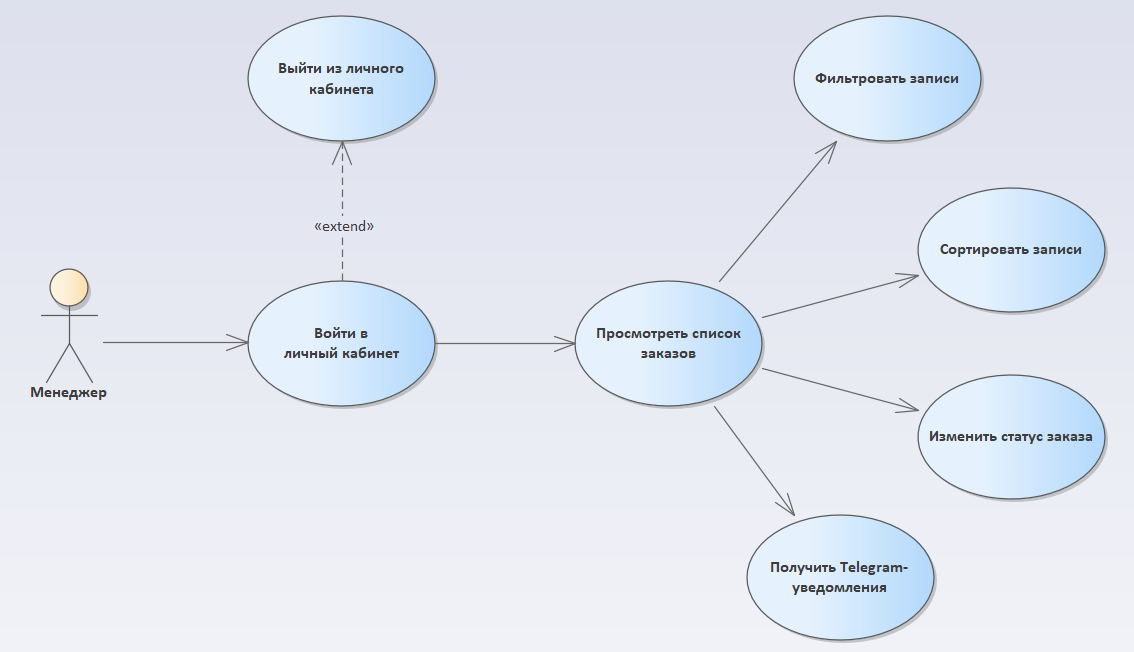


Рисунок 4.8 – Диаграмма вариантов использования для менеджера

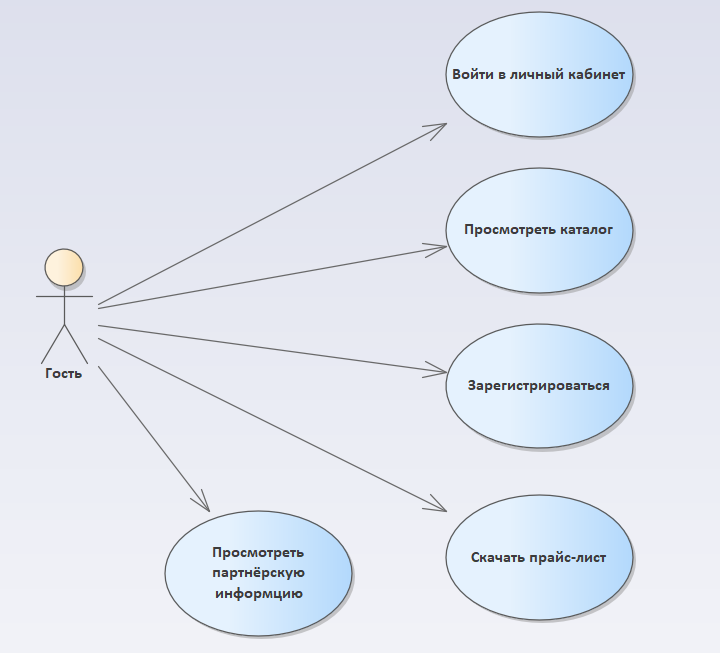


Рисунок 4.9 – Диаграмма вариантов использования для гостя

**4.4 Модели представления системы и их описание**

**4.4.1** Диаграмма классов – это UML-диаграмма, которая описывает систему, визуализируя различные типы объектов внутри системы и виды статических связей, которые существуют между ними. Он также иллюстрирует операции и атрибуты классов. Обычно они используются для изучения концепций области, понимания требований к программному обеспечению и описания подробных проектов. Классы представляют собой центральные объекты в системе. Он представлен прямоугольником с 3 отсеками. Первый показывает имя класса, а средний – атрибуты класса, которые являются характеристиками объектов. Символ интерфейса на диаграммах классов обозначает набор операций, которые детализируют ответственность класса. Диаграмма классов разработанной системы представлена на рисунке 4.9.

Разработанная система была построена на основе паттерна MVC (Model-View-Controller). На диаграмме показаны основные компоненты приложения: контроллеры, модели, сервисы. В рамках паттерна MVC модели представляют собой классы для описания сущностей предметной области, которым соответствуют таблицы в базе данные. Сервисы – вспомогательные классы, отвечающие за работу с бизнес-логикой приложения. Контроллеры – классы, связывающие другие компоненты и отдающие View (разметку-представление, которое видит пользователь).

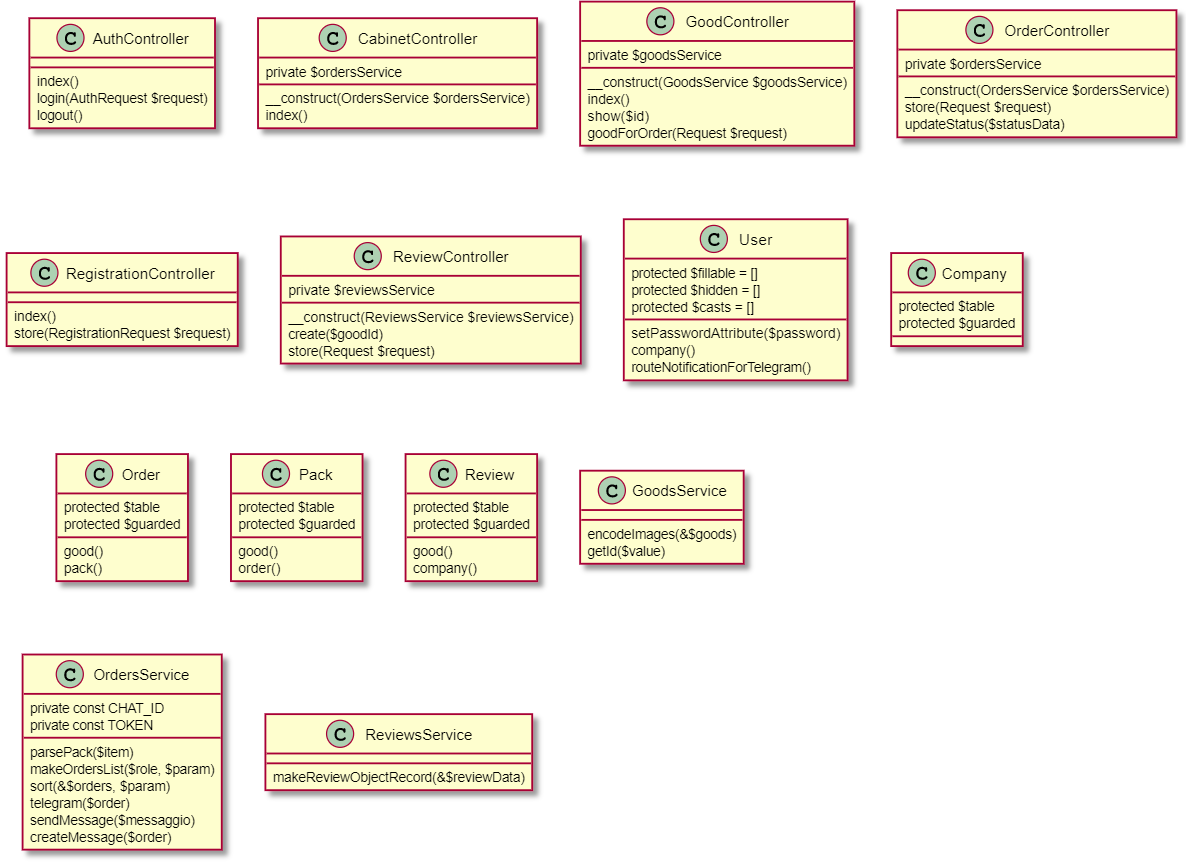


Рисунок 4.10 – Диаграмма классов

**4.4.2** Диаграмма компонентов (Component Diagram) используется для визуализации организации компонентов системы и зависимостей между ними. Они позволяют получить высокоуровневое представление о компонентах системы. Компонентами могут быть программные компоненты, такие как база данных или пользовательский интерфейс; или аппаратные компоненты, такие как схема, микросхема или устройство или бизнес-подразделение, такое как поставщик, платежная ведомость или доставка.

В качестве компонентов представленной системы выступают: фронтенд-часть приложения (то, что исполняется в браузере), а также компоненты серверной (backend) части. Среди компонентов сервера выделяются: модели (классы, описывающие сущность предметной области), контроллеры (для организации взаимодействия между другими компонентами и контроля отображений View), Request-классы (валидация), сервисы (бизнес-логика), классы роутинга и т.д. Связь между клиентом и сервером осуществляется по протоколу HTTP. Диаграмма компонентов представлена на рисунке 4.11.

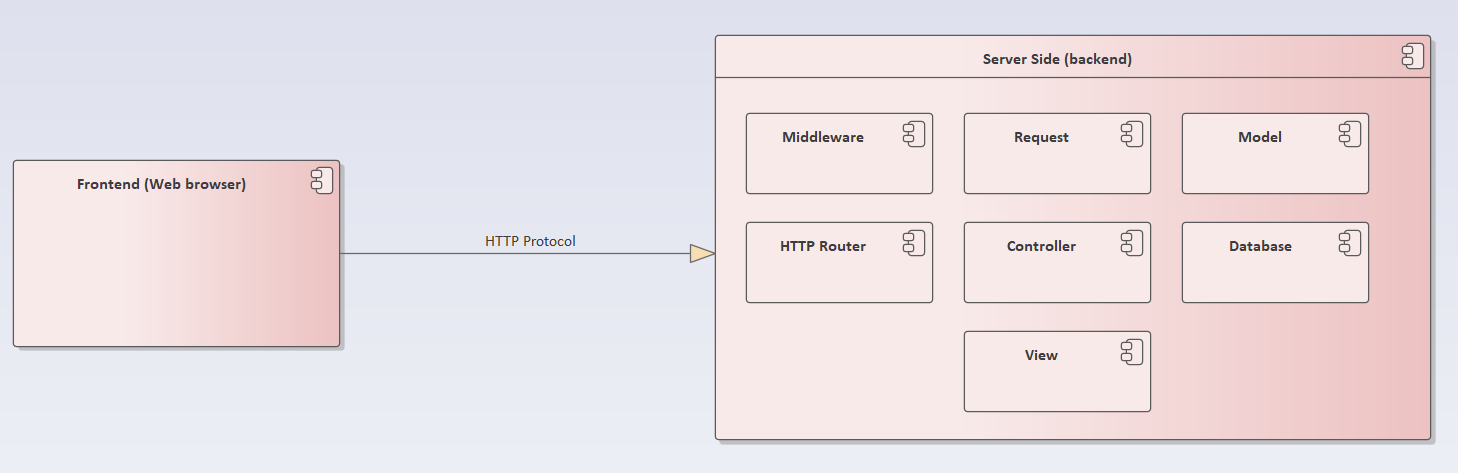


Рисунок 4.11 – Диаграмма компонентов

**4.4.3** Диаграмма развертывания используется для моделирования физических аспектов объектно-ориентированной системы. Такая диаграмма показывает конфигурацию узлов, где производится обработка информации, и то, какие компоненты размещены на каждом узле. Диаграммы развертывания используются для моделирования статического вида системы с точки зрения развертывания. В основном под этим понимается моделирование топологии аппаратных средств, на которых выполняется система. По существу, диаграммы развертывания - это просто диаграммы классов, сосредоточенные на системных узлах. Диаграммы развертывания важны не только для визуализации, специфицирования и документирования встроенных, клиент-серверных и распределенных систем, но и для управления исполняемыми системами с использованием прямого и обратного проектирования. Диаграмма развёртывания настоящей системы представлена на рисунке 4.12.

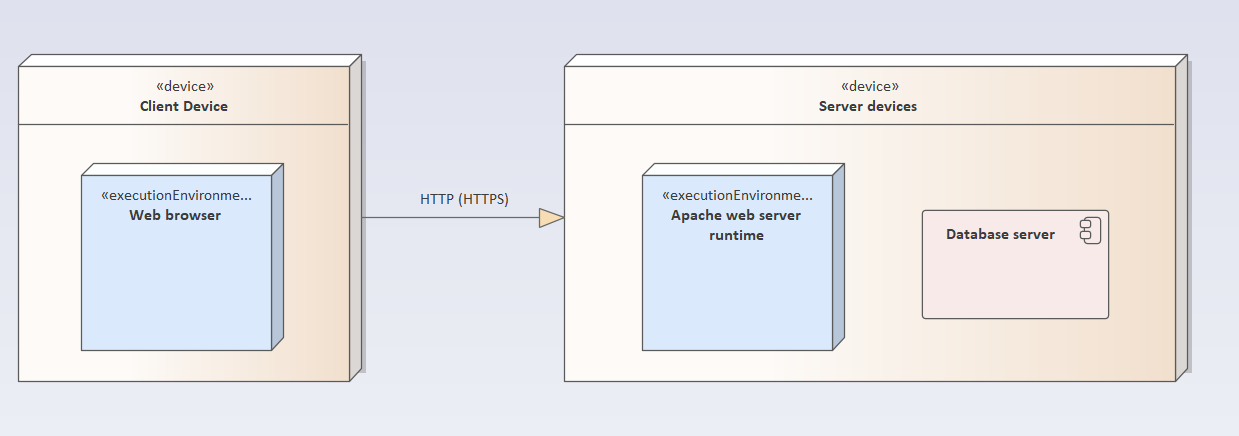


Рисунок 4.12 – Диаграмма развёртывания

**4.4.4** Диаграмма последовательности является одной из разновидности диаграмм взаимодействия и предназначена для моделирования взаимодействия объектов системы во времени, а также обмена сообщениями между ними. На диаграмме последовательности объекты в основном представляю экземпляры класса или сущности, обладающие поведением. В качестве объектов могут выступать пользователи, инициирующие взаимодействие, классы, обладающие поведением в системе или программные компоненты, а иногда и системы в целом. Диаграмма последовательности отображена на рисунке 4.13. На ней представлен основной процесс в системе – оформление заказа клиентом.

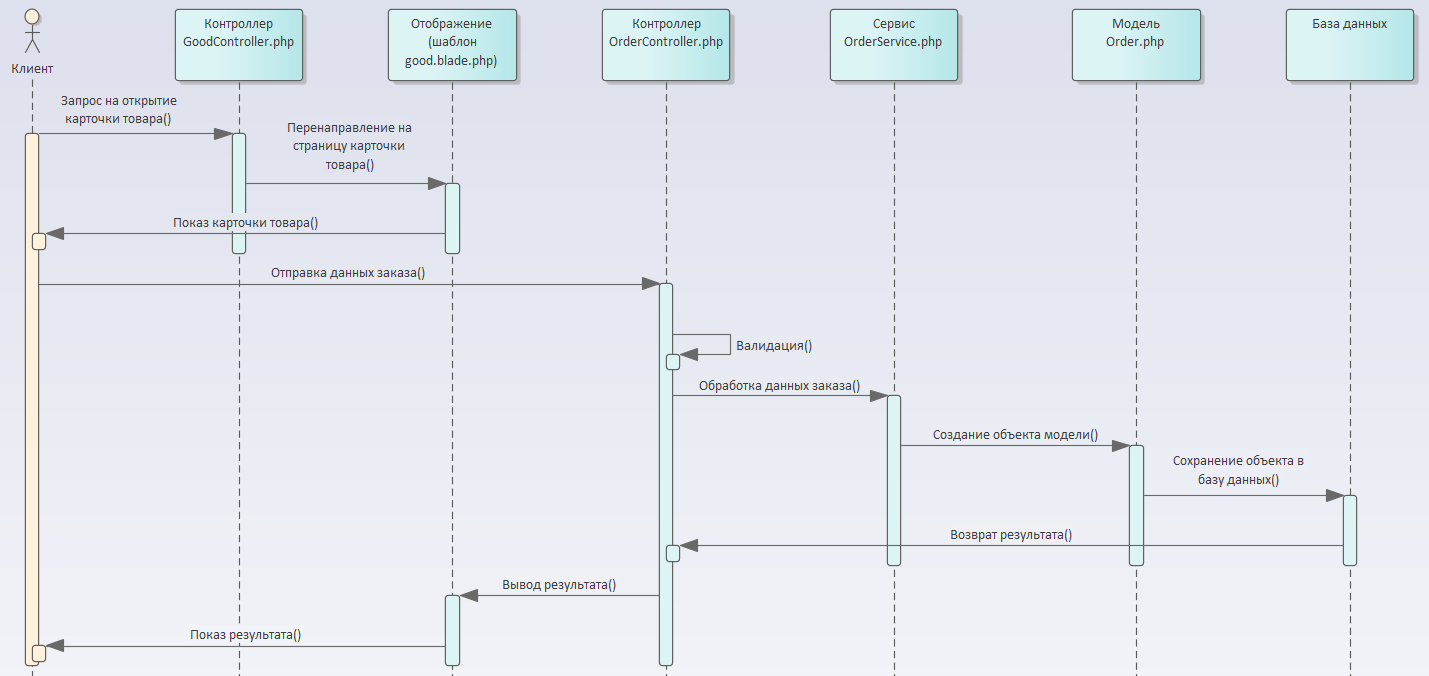


Рисунок 4.13 – Диаграмма развёртывания

**4.4.5** Диаграмма состояний – это один из пяти видов диаграмм UML, предназначенных для моделирования динамических аспектов поведения систем. Диаграмма состояний показывает конечный автомат. И диаграммы деятельности, и диаграммы состояний подходят для моделирования жизненного цикла объекта. Однако в то время, как диаграмма деятельности демонстрирует поток управления от одной деятельности к другой через множество объектов, диаграмма состояний отображает поток управления от состояния к состоянию внутри отдельного объекта. Диаграммы состояния применяются для моделирования динамических аспектов поведения систем.

Диаграмма состояний изображена на рисунке 4.14.

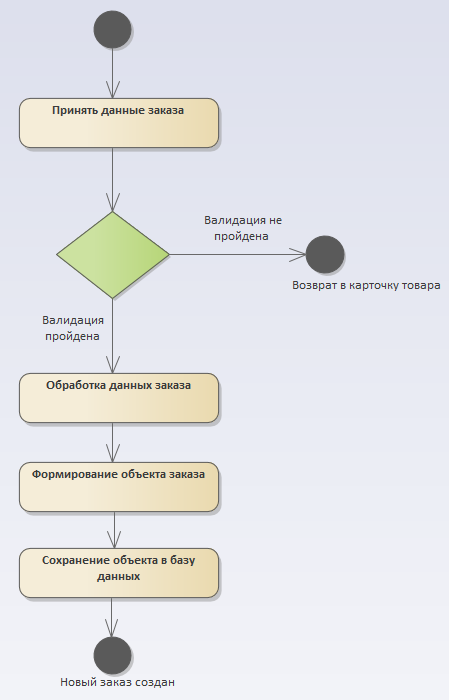


Рисунок 4.14 – Диаграмма состояний

**4.5 Описание алгоритмов, реализующих бизнес логику системы**

В разработанной системе функционалом оформления заказа обладает только авторизованный пользователь (клиент). Процедура оформления заказа включает в себя следующие этапы: авторизация в системе, выбор товара из каталога, указание количества товара, выбор упаковки. После отправления данных заказа в системе происходит валидация данных. В случае успеха на сайте создаётся новый заказ, передаваемый на обработку и контроль менеджерам. Блок-схема алгоритма системы представлена на рисунке 4.15.

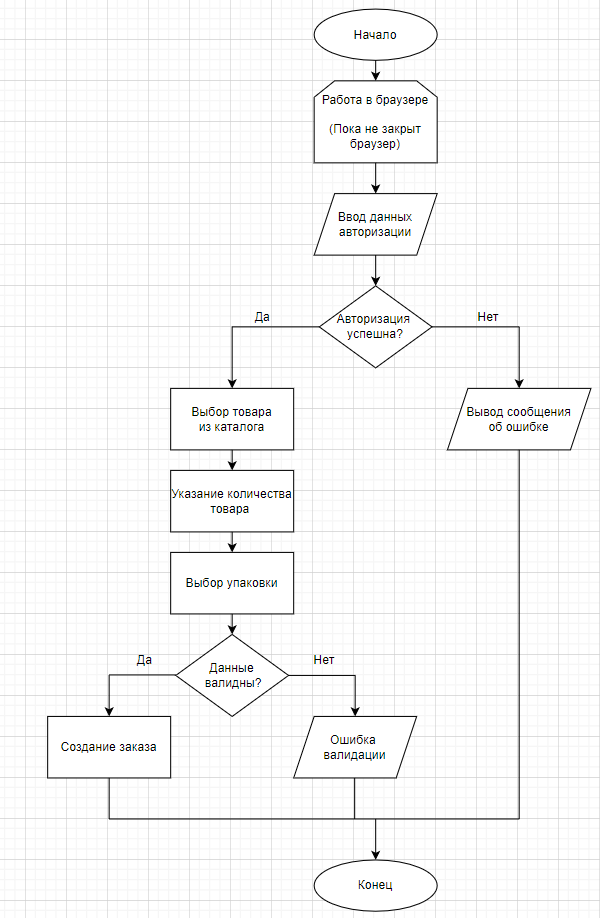


Рисунок 4.15 – Блок-схема алгоритма оформления заказа