**1.1 Бизнес-требования**

**1.1.1. Описание автоматизируемых бизнес-процессов**

**Процесс покупки товаров:**

1. Покупатель заходит на сайт интернет-магазина.
2. Просматривает товары по категориям или с помощью поиска.
3. Добавляет выбранные товары в корзину.
4. Переходит в корзину и подтверждает заказ.
5. Выполняет авторизацию или регистрацию (если не выполнена).
6. Выбирает способ оплаты (передается внешней системе проведения платежей).
7. Получает подтверждение об успешной оплате и оформлении заказа.
8. Заказ отправляется на обработку в складскую систему и систему логистики.

**Процесс администрирования:**

1. Администратор входит в систему через специальное клиентское приложение.
2. Управляет товарами (добавление, редактирование, удаление).
3. Управляет пользователями (просмотр, блокировка, разблокировка).
4. Просматривает отчеты о продажах и активности пользователей.
5. Настраивает права доступа и роли пользователей.
6. Следит за состоянием системы через подсистему мониторинга.

**1.1.2. Описание бизнес-требований (в формате UserStory)**

**Покупатель:**

1. Как покупатель, я хочу просматривать и искать товары, чтобы найти нужный мне продукт.
2. Как покупатель, я хочу добавлять товары в корзину, чтобы потом оформить заказ.
3. Как покупатель, я хочу просматривать и изменять содержимое корзины, чтобы подготовить её к заказу.
4. Как покупатель, я хочу регистрироваться и авторизоваться, чтобы получить доступ к персональным данным и истории заказов.
5. Как покупатель, я хочу оплачивать заказ, чтобы получить товары.

**Администратор:**

1. Как администратор, я хочу управлять товарами в магазине, чтобы поддерживать актуальность ассортимента.
2. Как администратор, я хочу управлять пользователями, чтобы поддерживать порядок в системе.
3. Как администратор, я хочу просматривать отчеты о продажах и активности, чтобы анализировать эффективность работы магазина.
4. Как администратор, я хочу настраивать права доступа и роли пользователей, чтобы обеспечить безопасность данных.
5. Как администратор, я хочу мониторить состояние системы, чтобы оперативно реагировать на сбои и проблемы.

**1.1.3. Описание типовых верхнеуровневых сценариев использования средств из состава магазина**

**Сценарий покупателя:**

1. Покупатель заходит на сайт интернет-магазина.
2. Выполняет поиск интересующего товара.
3. Добавляет товар в корзину.
4. Переходит к оформлению заказа.
5. Заполняет необходимые данные для доставки и оплаты.
6. Подтверждает заказ.

**Сценарий администратора:**

1. Администратор входит в систему через клиентское приложение.
2. Переходит в раздел управления товарами.
3. Добавляет новый товар, заполняет его характеристики и цену.
4. Сохраняет изменения.
5. Переходит в раздел управления пользователями.
6. Просматривает список пользователей и, при необходимости, блокирует нарушителя.

**1.1.4. Описание GUI веб-приложения**

**Покупательское приложение:**

* **Главная страница:** поиск по товарам, категории товаров, специальные предложения.
* **Страница товара:** изображение товара, описание, цена, кнопка "Добавить в корзину".
* **Корзина:** список добавленных товаров, возможность изменить количество, кнопка "Оформить заказ".
* **Регистрация/Авторизация:** формы для ввода данных, кнопки "Зарегистрироваться" и "Войти".
* **Личный кабинет:** история заказов, персональные данные, настройки.

**Администраторское приложение:**

* **Главная страница:** краткая информация о текущем состоянии магазина (продажи, активные пользователи).
* **Управление товарами:** список товаров, кнопки "Добавить", "Редактировать", "Удалить".
* **Управление пользователями:** список пользователей, кнопки "Блокировать", "Разблокировать".
* **Отчеты:** графики и таблицы с данными о продажах и активности.
* **Настройки:** управление ролями и правами доступа.

**1.1.5. Верхнеуровневые нефункциональные требования к системе**

1. **Производительность:** система должна обрабатывать до 1000 одновременных пользователей без снижения скорости ответа.
2. **Безопасность:** система должна обеспечивать защиту данных пользователей, в том числе шифрование конфиденциальной информации.
3. **Надежность:** система должна обеспечивать доступность не менее 99,9% времени в год.
4. **Масштабируемость:** система должна поддерживать возможность горизонтального и вертикального масштабирования.
5. **Легкость в использовании:** интерфейс должен быть интуитивно понятен пользователю.
6. **Сопровождаемость:** система должна быть легко модифицируема и обслуживаема, включая обновления и исправления.

**Архитектура проекта**

**1. Клиентское приложение покупателя**

* React.js для создания пользовательского интерфейса.
* Redux для управления состоянием приложения.
* Axios для работы с API.

**2. Клиентское приложение администратора**

* Angular или Vue.js для административного интерфейса.
* NgRx/Vuex для управления состоянием приложения.
* Axios или Fetch для работы с API.

**3. Подсистема разграничения прав доступа пользователей**

* JWT (JSON Web Token) для аутентификации и авторизации.
* RBAC (Role-Based Access Control) для управления правами доступа.

**4. Подсистема хранения данных**

* PostgreSQL или MySQL для хранения данных.
* Redis для кеширования.

**5. Подсистема журналирования клиентских и системных событий**

* ELK Stack (Elasticsearch, Logstash, Kibana) для сбора и анализа логов.
* Log4j или аналог для журналирования на серверной стороне.

**6. Подсистема мониторинга состояния составных частей системы**

* Prometheus и Grafana для мониторинга метрик.
* Alerts в случае сбоев или аномалий.

**Внешние системы:**

* Система проведения платежей.
* Клиентское приложение продавца.
* Складская система.
* Система логистики и организации доставки товара.
* Аналитическая система (продуктовая аналитика).