**Тема: Разработка системы управления запасами для склада**

**1. Введение**

Целью данного технического задания (ТЗ) является разработка системы управления запасами для складов, которая позволит эффективно отслеживать и управлять товарными запасами, минимизировать потери и оптимизировать процессы учета. Система должна обеспечивать автоматизацию процессов поступления, хранения, отгрузки и учета товаров.

**2. Цели и задачи**

**- Цели:**

- Обеспечить точный учет товарных запасов на складе.

- Упрощение процессов управления запасами и повышения эффективности работы склада.

**- Задачи:**

- Разработать интерфейс для ввода и редактирования данных о товарах.

- Реализовать функционал для отслеживания движений товаров (поступление, отгрузка, возврат).

- Обеспечить возможность генерации отчетов о состоянии запасов.

- Внедрить систему уведомлений о низком уровне запасов.

**3. Функциональные требования**

**1. Регистрация и авторизация**

- Пользователи должны иметь возможность зарегистрироваться и авторизоваться в системе.

**2. Управление товарами**

- Добавление, редактирование и удаление товаров из базы данных.

- Указание характеристик товара (название, описание, количество, цена, категория).

**3. Управление запасами**

- Учет поступления товаров на склад.

- Учет отгрузки товаров со склада.

- Учет возвратов товаров.

**4. Отчеты и аналитика**

- Генерация отчетов о текущем состоянии запасов.

- Анализ движения товаров по категориям и периодам.

**5. Уведомления**

- Уведомления о низком уровне запасов.

- Уведомления о предстоящих сроках годности товаров.

**4. Нефункциональные требования**

- Платформы: Веб-приложение, доступное через браузер.

- Безопасность: Защита данных пользователей и товаров (шифрование, безопасное хранение паролей).

- Производительность: Быстрая загрузка и отзывчивость интерфейса.

- Юзабилити: Интуитивно понятный интерфейс для пользователей.

**5. Технические требования**

- Язык программирования: Python (Django) или JavaScript (Node.js).

- База данных: PostgreSQL или MySQL для хранения данных.

- API:RESTful API для интеграции с другими системами (например, ERP).

**6. Этапы разработки**

1. Анализ требований – 2 недели

2. Проектирование архитектуры – 3 недели

3. Разработка интерфейса – 4 недели

4. Кодирование функционала – 6 недель

5. Тестирование – 3 недели

6. Внедрение и публикация – 2 недели

7. Сопровождение и обновления – по мере необходимости

**Заключение**

Разработка системы управления запасами для склада позволит улучшить учет товарных запасов, минимизировать потери и оптимизировать процессы на складе. Система должна быть интуитивно понятной и обеспечивать высокую степень безопасности данных.

**1. Языки программирования**

**Python (с использованием Django)**

- Почему:

- Простота и читаемость кода: Python известен своей простотой, что облегчает разработку и поддержку кода.

- Быстрая разработка: Django — это мощный фреймворк, который позволяет быстро разрабатывать веб-приложения благодаря встроенным функциям (например, аутентификация, админка, ORM).

- Сообщество и поддержка: Python имеет большое сообщество, что облегчает поиск решений для возникающих проблем.

**JavaScript (с использованием Node.js)**

- Почему:

- Асинхронная обработка: Node.js позволяет обрабатывать запросы асинхронно, что улучшает производительность приложения, особенно при работе с большим количеством пользователей.

- Единый стек: Использование JavaScript как на клиентской, так и на серверной стороне упрощает разработку и позволяет использовать одни и те же библиотеки и инструменты.

- Поддержка реального времени: Node.js хорошо подходит для приложений, требующих обновления данных в реальном времени, что может быть полезно для уведомлений о запасах.

**2. Базы данных**

**PostgreSQL**

- Почему:

- Надежность и производительность: PostgreSQL — это мощная объектно-реляционная СУБД, которая обеспечивает высокую производительность и надежность.

- Расширяемость: Поддерживает расширения и пользовательские типы данных, что позволяет адаптировать базу под конкретные нужды приложения.

- Поддержка сложных запросов: PostgreSQL позволяет выполнять сложные запросы и операции с данными, что может быть полезно для анализа запасов.

**MySQL**

- Почему:

- Широкое использование: MySQL является одной из самых популярных СУБД, что облегчает поиск разработчиков и поддержку.

- Производительность: Хорошо подходит для приложений с высокой нагрузкой и большим объемом данных.

- Поддержка репликации: MySQL поддерживает репликацию, что может быть полезно для обеспечения отказоустойчивости.

**3. Frontend технологии**

**HTML/CSS**

- Почему:

- Стандарт для веб-разработки:\*\* HTML и CSS являются основными технологиями для создания веб-страниц, необходимых для разработки пользовательского интерфейса.

**JavaScript (с использованием фреймворков, таких как React или Vue.js)**

- Почему:

- Интерактивность: JavaScript позволяет создавать динамичные и интерактивные интерфейсы, что улучшает пользовательский опыт.

- Компонентный подход: Фреймворки, такие как React или Vue.js, позволяют создавать переиспользуемые компоненты, что упрощает разработку и поддержку интерфейса.

**4. API**

**RESTful API**

- Почему:

- Стандартизация: RESTful API является стандартом для взаимодействия между клиентом и сервером, что упрощает интеграцию с другими системами.

- Гибкость: RESTful API позволяет легко расширять функциональность системы, добавляя новые эндпоинты по мере необходимости.

**5. Дополнительные технологии**

**Docker**

- Почему:

- Контейнеризация: Docker позволяет упаковывать приложение и все его зависимости в контейнеры, что упрощает развертывание и управление приложением.

- Изоляция окружений: Обеспечивает изоляцию окружений разработки, тестирования и продакшн, что минимизирует проблемы совместимости.

**Git**

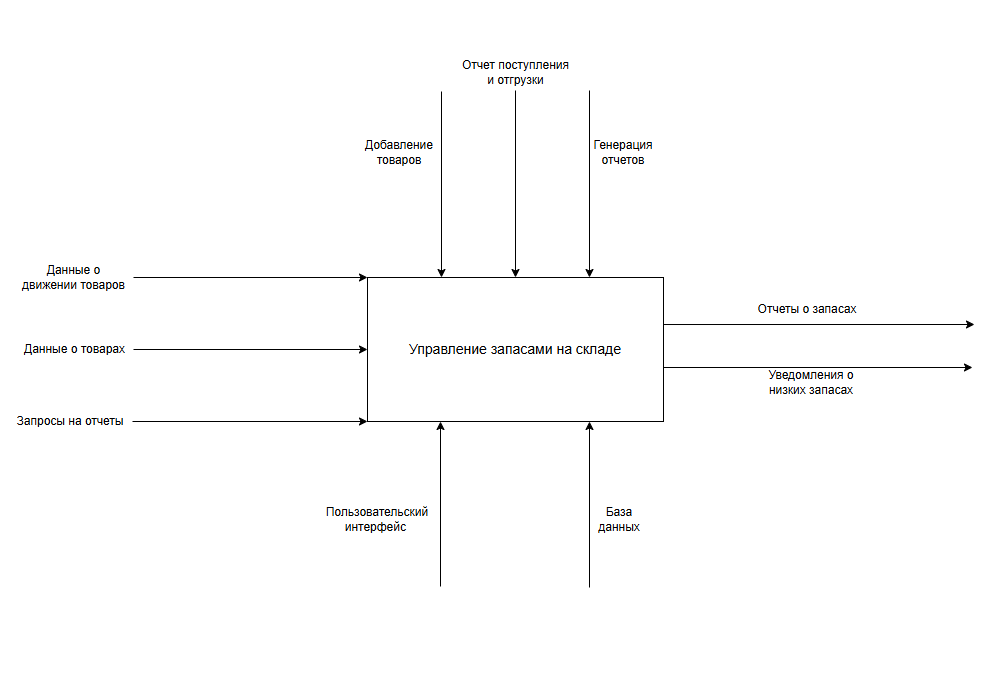
- Почему:

- Системы контроля версий: Git позволяет отслеживать изменения в коде, что упрощает совместную работу команды разработчиков и управление версиями приложения.

**Заключение**

Выбор этих языков и технологий обеспечит надежную, производительную и масштабируемую систему управления запасами, которая будет легко развивать и поддерживать.

**IDEF0 схема**



**ER диаграмма**

