**Описание монолитного приложения**

Пример приложения: Система управления проектами.

**Структура приложения:**

**1. Модуль пользователей (User Module)**

- Регистрация и аутентификация пользователей

- Управление профилем пользователя

- Роли и права доступа

**2. Модуль проектов (Project Module)**

- Создание и редактирование проектов

-Управление задачами внутри проектов

- Установка сроков и приоритетов

**3. Модуль задач (Task Module)**

- Создание, редактирование и удаление задач

- Назначение задач пользователям

- Комментарии и вложения к задачам

**4. Модуль отчетности (Reporting Module)**

- Генерация отчетов по проектам и задачам

- Визуализация данных (графики, диаграммы)

**5. Модуль уведомлений (Notification Module)**

- Уведомления о изменениях в задачах и проектах

- Email и push-уведомления

**6. Модуль интеграций (Integration Module)**

- Интеграция с внешними сервисами

- API для доступа к данным приложения

**Основные возможности:**

Управление проектами и задачами.

Система ролей и прав доступа.

Уведомления о событиях.

Генерация отчетов.

Интеграция с другими сервисами.

**Узкие горлышки**

**База данных:** При высокой нагрузке может возникнуть проблема с производительностью запросов к базе данных, особенно если не оптимизированы индексы или структура данных.

**Сетевые запросы:** Если приложение делает много внешних API-запросов, это может замедлить работу при высоких нагрузках.

**Обработка данных:** Сложные операции обработки данных могут занимать много времени и ресурсов, что влияет на отклик приложения.

**Память и ресурсы сервера:** Если приложение не оптимизировано по использованию памяти, это может привести к утечкам памяти и снижению производительности.

**План по выделению модулей и сервисов**

**Модуль пользователей**

* Выделить в отдельный сервис (User Service) для управления пользователями и аутентификацией.
* Использовать отдельную базу данных для хранения информации о пользователях.

**Модуль проектов и задач**

* Создать отдельные сервисы для управления проектами и задачами.
* Реализовать API для взаимодействия между сервисами.

**Модуль отчетности**

* Выделить в отдельный сервис (Reporting Service) для генерации отчетов.
* Использовать кэширование для часто запрашиваемых отчетов.

**Модуль уведомлений**

* Создать сервис уведомлений для обработки и отправки уведомлений.
* Реализовать асинхронную обработку уведомлений для повышения производительности.

**Модуль интеграций**

* Выделить в отдельный сервис для управления интеграциями с внешними сервисами.
* Использовать очередь сообщений для обработки запросов к внешним API.