Санкт-Петербургский Национальный
Исследовательский Университет Информационных
технологий, механики и оптики

Лабораторная работа 3

Проектирование архитектуры программного продукта

Выполнил: Фисенко
Максим Вячеславович
Группа № К34211
Проверил: Иванов
Сергей Евгеньевич

Санкт-Петербург
2024

Цель работы

Изучить методику создания архитектуры программного продукта.

Задачи

- Описать многозвенную клиент-серверную архитектуру приложения;
- Построить схему архитектуры и подробно описать ее компоненты.

Ход работы

Задание 1. Описание многозвенной клиент-серверной архитектуры программного продукта

Для выполнения данного задания в первую очередь необходимо указать сам программный продукт, описание архитектуры которого будет приводиться далее. В качестве такого продукта было взято веб-приложение автоматизации процесса рекрутинга для HR-специалистов.

Архитектура данного приложения будет включать в себя 4 уровня:

- *Уровень представления*. Данный уровень представляет из себя интерфейс, по которому пользователь взаимодействует с приложением;
- *Уровень бизнес-логики*. На данном уровне выполняется обработка данных, а также на нем реализована бизнес-логика приложения;
- *Уровень данных*. Данный уровень отвечает за хранение данных и управление ими;
- *Уровень интеграции*. На данном уровне проводится интеграция с внешними системами, необходимыми для работы нашего приложения.

Задание 2. Построение схемы архитектуры и подробное описание ее компонентов

Для выполнения данного задания первым делом необходимо подробно описать каждый уровень архитектуры приложения, также описать их функции и технологии, применяемые для его реализации.

Уровень представления

Это уровень, с которым взаимодействует конечный пользователь (кандидаты, HR-специалисты и администраторы). Он обеспечивает интерфейсы для удобного ввода и вывода информации.

Интерфейсы

- Веб-интерфейс для HR-специалистов

  Веб-страницы для выполнения ключевых функций (например, управления задачами, взаимодействия с процессами или просмотра аналитики). Используются технологии *HTML*, *CSS*, *JavaScript*, а также фреймворки типа *React*, *Angular*, *Vue.js* для создания динамических страниц.
- Интерфейс для администраторов

  Упрощенная панель управления для внесения данных, назначения статусов, просмотра отчетности и т. д.

Функции уровня представления

- Обеспечение пользователя доступом к функциональности системы (например, оформление действий, ввод данных, получение уведомлений);
- Взаимодействие с сервером через HTTP/HTTPS-запросы;
- Отображение данных, полученных от серверной части (результаты операций, отчеты, статус действий и т. д.);
- Аутентификация и авторизация пользователей (вход по логину и паролю).

Уровень бизнес-логики

Это ключевой уровень, где реализуются бизнес-процессы проекта. На этом уровне выполняются операции обработки данных и взаимодействия между клиентским интерфейсом и хранилищем.

Основные компоненты

- Сервер приложений

Центральная часть системы, где реализуются все правила обработки данных. Возможные технологии разработки: *Go, Python, Node.js, Java* и т. д.

- Бизнес-логика

  В данном компоненты реализуются такие функции, как обработка пользовательских данных, валидация операций, управление процессами, а также реализована аутентификация и авторизация (с помощью *JWT, OAuth* и т. д.), а также шифрование данных.

- API

  Предоставляет интерфейсы для взаимодействия клиентской и серверной частей (REST или GraphQL).

Функции уровня логики

- Обработка всех запросов клиентов;
- Реализация сложной бизнес-логики (например, расчет аналитики, распределение ресурсов);
- Управление сессиями пользователей;
- Взаимодействие с базой данных для получения и сохранения данных;
- Мониторинг и управление транзакциями.

Уровень данных

Этот уровень отвечает за хранение всех данных в системе, управление ими и доступ к ним.

Основные компоненты

- Реляционные базы данных (SQL)

  Данный компонент отвечает за хранение основной информации: данные о пользователях, операциях, отчетах и других сущностях проекта. Примеры технологий: *PostgreSQL, MySQL, Microsoft SQL Server*.

- NoSQL базы данных

  Для хранения неструктурированных данных, таких как логи активности или метаданные. Примеры: *MongoDB, Cassandra, DynamoDB*.

- Системы кэширования

  Ускоряют доступ к часто запрашиваемым данным. Примеры: *Redis, Memcached.*

Функции уровня данных

- Хранение информации обо всех объектах системы (пользователи, действия, статистика);
- Управление транзакциями для обеспечения корректности операций;
- Поддержка поиска, фильтрации и аналитики данных;
- Интеграция с уровнем приложений через безопасные подключения;

Уровень интеграции

Уровень интеграции отвечает за взаимодействие с внешними сервисами и платформами, необходимыми для выполнения функций рекрутинговой системы, таких как анализ, уведомления, и работа с внешними базами данных.

Основные компоненты

- Интеграция с сервисами видеоконференций

  Позволяет автоматически планировать и организовывать онлайн-собеседования для кандидатов через платформы, такие как *Zoom, Microsoft Teams или Google Meet*.

- Интеграция с сервисами массовой рассылки

  Например, *SendGrid* или *Mailgun* для автоматической отправки уведомлений кандидатам о статусе их заявок (принято, отклонено, этап собеседования и т. д.).

- Интеграция с системами управления персоналом

Интеграция с популярными HR-системами, такими как *Workday* или *BambooHR*, для автоматической передачи данных о нанятых кандидатах в базу сотрудников компании.

Функции уровня интеграции

- Обеспечение связи с внешними платформами;
- Сбор данных и отправка в сторонние аналитические инструменты;
- Поддержка поиска, фильтрации и аналитики данных;
- Логирование взаимодействий для упрощения диагностики и исправления ошибок;

Схема архитектуры со всеми уровнями изображена ниже на рисунке 1.

Рисунок 1 - Схема архитектуры приложения

Вывод

В ходе выполнения данной лабораторной работы была разработана и описана многозвенная клиент-серверную архитектура веб-приложения автоматизации процесса рекрутинга для HR-специалистов. Были выявлены и

описаны все уровни архитектуры данного приложения, а также составлена диаграмма, описывающая схему архитектуры приложения.