Моя интерпретация технического задания ввиду его неоднозначности:

Требуется разработать систему состоящую из нескольких частей:

1. Пользовательского интерфейса для сайта бизнеса доступ к которой будет у любого посетителя сайта, который хочет устроиться в компанию на сайте которой он находится. С помощью этого интерфейса в виде чата, соискатель генерирует для себя сопроводительное письмо. Для этого пользователю необходимо отправить в чат свое CV(резюме) и описание желаемой вакансии, после чего система сгенерирует сопроводительное письмо на основе предоставленных данных. При этом, резюме кандидата сохраняется в базе данных компании(CRM Bitrix).

2. Пользовательского интерфейса для HR менеджеров компании, которые работают непосредственно с уже существующим интерфейсом Битрикс. Этот интерфейс необходимо дополнить кастомным виджетом, который также будет представлен в виде окна мессенджера. HR менеджер отправляет в чат этого мессенджера описание вакансии, на которую необходимо нанять человека. В ответ чат-бот ищет подходящее резюме из списка тех, которые загрузили соискатели в пункте 1. Если подходящего резюме нет, система генерирует демонстрационное резюме по указанному шаблону.

3. Со стороны бекенда функционалом пункта 1. занимается скрипт на Python, который, используя REST API ChatGPT, генерирует сопроводительные письма на основе данных полученных с помощью REST API Bitrix.

4. Со стороны бекенда функционалом пункта 2. занимается скрипт на PHP, который, используя REST API ChatGPT, генерирует демонстрационное резюме а также находит подходящее под вакансию резюме используя REST API Bitrix.

Предлагаемый выбор технологий:

1. Фронтенд

Для создания пользовательского интерфейся предлагается использовать HTML, CSS, JS и React фреймворк. React фреймворк позоволит обеспечить модульный, читаемый код, который легко подерживать и переиспользовать.

2. Связь между фронтендом и бекендом предлагается сделать в форме REST API. Это значит, что как для PHP, так и для Python скрипта будет необходимо реализовать свое REST API. Это позволит обеспечить независимость бекенда от пользовательского интерфейса. Также это обеспечит достаточную масшабируемость в будущем, для расширения функционала приложения. В случае Python предлагается использовать Fast API. В случае PHP предлагается использовать Slim. Оба фреймворка позиционируются как легковесные фреймворки для создания REST API.

Архитектура приложения

- Пользовательский интерфейс на React

- Python REST API

- PHP REST API

