**1. Архитектура и выбор технологий**

1.1. Бэкенд-фреймворк

* Текущий стек: Flask (Python).

Flask — микрофреймворк для Python, который предоставляет минималистичный набор инструментов для создания веб-приложений. Он гибкий и легковесный, гораздо более прост для изучения (в сравнении с Django), что позволяет быстро адаптировать его под конкретные задачи. Flask не навязывает строгую структуру проекта, что, несомненно, является плюсом.

Выбор обусловлен достаточно небольшим функционалом и сложностью основного веб-приложения. А также, комьюнити фреймворка очень большое и найти решение для возникшей проблемы не составит труда.

* Решение: оставить Flask, так как библиотека позволяет быстро адаптировать проект под специфические требования.

1.2. СУБД

* Проблема текущего решения: хранение данных в файлах не обеспечивает надежность и масштабируемость.
* Решение: использовать реляционную базу данных PostgreSQL.
  + - Поддержка сложных запросов и используется достаточно часто, что облегчит работу, так как много материалов по работе.

Примерная схема данных:

* Таблица Users: хранение данных о пользователях (ID, имя, email, хеш пароля и т.д.).
* Таблица Agents: хранение данных о агентах (ID, имя, связанный пользователь и т.д.).
* Таблица ChatHistory: хранение истории диалогов (ID чата, ID пользователя, сообщения, время и т.д.).
* Таблица KnowledgeFiles: хранение метаданных о файлах (ID файла, путь, тип, размер и т.д.).

1.3. Хранение файлов

* Проблема: локальное хранение файлов не подходит для выпуска готового приложения.
* Решение:
  + Использовать облачное хранилище. В базе данных хранить только метаданные файлов (путь, тип, размер).

**2. Безопасность**

2.1. Аутентификация и авторизация

* Пароли: заменить хранение в открытом виде на хеширование с использованием bcrypt — алгоритм хеширования, специально разработанный для безопасного хранения паролей. Он использует соль для дополнительной защиты и замедляет процесс подбора паролей.

2.2. Защита API-ключей

* OpenAI API Key: хранить в переменных окружения, а не в коде. В таком случае достаточно будет изменить переменную окружения, не меняя внутренний код. А также это поможет сохранить код в случае взлома репозитория или утечки исходного текста.

**3. Интеграция с Telegram**

3.1. Технологии

* Библиотека: python-telegram-bot.
* Сохранять историю диалогов в таблице ChatHistory, что позволит в будущем анализировать весь диалог, находить сообщения в конкретный момент времени и, возможно, развивать агента.