**Этап 0: Подготовка (Неделя 1)**

**Цель:** Подготовить рабочее окружение на своем компьютере. Всё, что нужно установить и настроить.

| Шаг | Что делать? | Детали и комментарии |
| --- | --- | --- |
| **1. Установить Python** | Скачать и установить последнюю версию Python 3.11+ с официального сайта [python.org](https://python.org/). | Обязательно отметить галочку "Add python.exe to PATH" при установке. |
| **2. Установить IDE/Редактор кода** | Рекомендую **Visual Studio Code** (VS Code) — бесплатный, мощный, с кучей расширений для Python. | Альтернативы: PyCharm Community Edition (тоже бесплатен). |
| **3. Установить Git** | Скачать и установить Git с [git-scm.com](https://git-scm.com/). | Нужен для контроля версий вашего кода. Вы будете делать коммиты после каждой успешной задачи. |
| **4. Настроить виртуальное окружение** | Создать папку проекта filin\_aib. В терминале (командной строке) перейти в нее и выполнить: python -m venv venv | Это создаст изолированную среду для Python-библиотек проекта. Активируется командой: # Для Windows: .\venv\Scripts\activate |
| **5. Установить базовые библиотеки** | В активированном виртуальном окружении выполнить: pip install pyside6 langchain chromadb sentence-transformers | Это основа. sentence-transformers нужен для преобразования текста в векторы. |

**Итог этапа:** У вас готово к работе чистое программистское окружение.

**Этап 1: "Сердце системы" — Работа с локальной LLM (Недели 2-4)**

**Цель:** Убедиться, что вы можете запускать локальную модель и получать от нее ответы.

| Шаг | Что делать? | Детали и сроки |
| --- | --- | --- |
| **1. Установить Ollama** | Скачать и установить Ollama с официального сайта [ollama.ai](https://ollama.ai/). | Займет 5 минут. После установки она будет работать как фоновая служба. |
| **2. Скачать модель** | В терминале выполнить команду: ollama pull llama3.1:8b | **Внимание:** Это займет время (размер модели ~5 ГБ). Это **самый важный шаг**. Модель llama3.1:8b — хороший баланс между скоростью и качеством для ваших задач. |
| **3. Протестировать модель** | Написать простейший Python-скрипт, который через библиотеку requests отправляет запрос на http://localhost:11434/api/generate и получает ответ от модели. | **Задача на 1-2 дня.** Цель — получить от модели любой ответ в вашей программе на Python. |
| **4. Интегрировать LangChain** | Переписать тестовый скрипт, используя LangChain для общения с Ollama. LangChain упрощает дальнейшую работу. | **Задача на 1 день.** from langchain\_community.llms import Ollama llm = Ollama(model="llama3.1:8b") print(llm.invoke("Расскажи анекдот")) |

**Итог этапа:** Вы доказали, что "движок" работает. Ваша программа на Python может запрашивать и получать ответы от мощной LLM, работающей локально.

**Этап 2: "Память системы" — База знаний (Недели 5-8)**

**Цель:** Научить систему загружать ваши документы и отвечать на вопросы по ним.

| Шаг | Что делать? | Детали и сроки |
| --- | --- | --- |
| **1. Разобраться с ChromaDB** | Написать скрипт, который создает коллекцию в ChromaDB и добавляет в нее несколько текстовых строк, преобразованных в векторы. | **Задача на 2-3 дня.** Цель — понять принцип работы векторной БД. |
| **2. Реализовать загрузку PDF/TXT** | Написать функцию, которая извлекает простой текст из PDF и TXT-файлов и разбивает его на чанки (блоки). | Используйте библиотеку PyPDF2 для PDF. **Задача на 3-4 дня.** |
| **3. Добавить поддержку OCR (EasyOCR)** | Написать функцию, которая принимает изображение (JPG/PNG) или PDF-скан, распознает текст с помощью easyocr и возвращает его. | pip install easyocr. Может быть сложно с зависимостями. **Выделите неделю.** |
| **4. Создать цепочку RAG** | С помощью LangChain объединить все вместе: 1. Получить вопрос от пользователя. 2. Найти в ChromaDB релевантные чанки из ваших документов. 3. Передать и вопрос, и найденные чанки в локальную LLM (Ollama). 4. Получить и вывести ответ. | **Это ключевая задача этапа. Заложите 2 недели** на эксперименты и отладку. |

**Итог этапа:** У вас работает прототип самого ценного: система загружает ваши документы и точно отвечает на вопросы по ним, используя локальный ИИ. Уже сейчас это мощный инструмент.

**Этап 3: "Лицо системы" — Интерфейс (Недели 9-12)**

**Цель:** Создать удобное оконное приложение вместо консольных скриптов.

| Шаг | Что делать? | Детали и сроки |
| --- | --- | --- |
| **1. Освоить основы PySide6** | Создать простейшее окно с кнопкой и текстовым полем. По нажатию кнопки что-то происходит. | **Задача на 2-3 дня.** Посмотрите туториалы по PySide6. |
| **2. Спроектировать интерфейс** | Нарисовать от руки или в Figma макет: область для загрузки файлов, чат-интерфейс (поле ввода, кнопка "Отправить", область истории сообщений). | **Задача на 1 день.** |
| **3. Связать интерфейс с логикой** | Перенести функции из Этапов 1 и 2 в приложение PySide6. Сделать так, чтобы при нажатии кнопки в интерфейсе запускалась цепочка RAG. | **Самая сложная задача этапа. Заложите 2-3 недели.** Не бойтесь, что интерфейс будет выглядеть просто. Главное — работающая логика. |

**Итог этапа:** У вас есть первое настоящее рабочее приложение с кнопками и окнами, которое делает всё то же, что и консольные скрипты, но уже удобно.

**Дальнейшие этапы (кратко):**

* **Этап 4: Безопасность и данные (~1.5 мес.):** Добавить шифрование SQLite и ChromaDB, систему логинов и паролей.
* **Этап 5: Внешние запросы (~1 мес.):** Реализовать "двухконтурную" архитектуру для безопасного доступа в интернет.
* **Этап 6: Автоматизация (~1.5 мес.):** Разработать 1-2 модуля автоматизации (например, для анализа отзывов или расчета цен).
* **Этап 7: Лицензирование и упаковка (~1 мес.):** Написать сервер лицензий, создать инсталлятор для приложения.

**Важно:** Этот план — ориентир. Где-то вы сможете управиться быстрее, где-то возникнут сложности, и этап затянется. Не переживайте. Главное — начать с Этапа 0 и двигаться последовательно.

**Начните с Этапа 0 прямо сегодня.** Как только настроите окружение и у вас заработает команда ollama run llama3.1:8b в терминале — вы почувствуете огромный прогресс и мотивацию двигаться дальше. У вас все получится.