

1 - Репозиторий в шести файлах (Apache-2.0, livekit-agents 1.x)

src/agent.py
src/transcriber.py
src/notes.py
src/recording.py
src/token_server.py
tests/test_notes.py

Точка входа: AgentServer + диспатч + связка + резюме на завершении
По одной STT-сессии на участника (multi-user transcriber)
TranscriptStore + один проход LLM + запись .md / .json
Опционально: запись комнаты через LiveKit Egress
FastAPI: явный диспатч + выдача токена входа
Юнит-тесты слоя заметок (без сети)

2 - Быстрый старт (около пяти минут)

```
uv sync                                # установить зависимости
cp .env.example .env.local             # ключи LIVEKIT / DEEPGRAM / OPENAI
uv run python src/agent.py download-files # один раз скачать веса VAD
uv run python src/agent.py console      # пообщаться в терминале
uv run python src/agent.py dev          # воркер против вашего сервера
```

3 - Почему он listen-only (и дешевле)

raise StopResponse() + audio_output=False -> агент никогда не говорит.
По STT-сессии на участника -> атрибуция строк без шага диаризации.
Нет text-to-speech -> самый дорогой слой исчезает.

4 - Структура затрат (LiveKit Cloud, listen-only)

Счёт: минуты участников WebRTC + минуты сессии агента + исходящие ГБ
Плюс: speech-to-text за минуту (~\$0.006/мин). Строки TTS нет.
Сверьте тарифы на livekit.io/pricing; полный пример — в уроке.

5 - Чек-лист build-vs-buy

- ☐ Ноутейкер живёт внутри вашего приложения? Стройте на этом репо (Cloud или self-host).
- ☐ Строгая приватность или резидентность данных? Self-host open-source сервера LiveKit.
- ☐ Нужен лишь транскрипт звонка в Zoom? Купите бота встреч (например, Recall.ai ~\$0.50/час).
- ☐ Нужно, чтобы агент говорил? Добавьте стадию TTS - и ждите примерно удвоения затрат.
- ☐ Запись на Cloud падает? Используйте file_outputs=[...], а не одиночное поле 'output'.