Speak2MD — API документация (MVP)

Эта документация описывает публичные эндпоинты для загрузки аудио, отслеживания прогресса, получения результата, а также базовую аутентификацию и работу с транскриптами. Основано на проектных материалах по MVP: поддержка форматов mp3/m4a/wav, лимиты 50 МБ и 120 минут.

Базовые URL

- Прод: `https://api.speak2md.example.com`
- Локально: `http://localhost:8000`

Аутентификация

Используется **JWT Bearer**. Получите `access_token` через `/api/auth/login`, передавайте его в заголовке:

```
Authorization: Bearer <access_token>
```

Статусы задач

- `processing` — в обработке
- `ready` — готово
- `error` — ошибка

Быстрый старт

1. Зарегистрируйтесь → **POST** `/api/auth/register`
2. Войдите → **POST** `/api/auth/login` → получите `access_token`
3. Загрузите аудио → **POST** `/api/upload` → получите `job_id`
4. Подключитесь к WS → **WS** `/api/ws/{job_id}` чтобы видеть прогресс
5. Проверьте статус → **GET** `/api/status/{job_id}`
6. Заберите результат → **GET** `/api/result/{job_id}?format=markdown|json`

Endpoints

POST /api/upload

Загрузить аудиофайл и создать задачу.

Request (multipart/form-data):

- `file` — бинарный файл (mp3/m4a/wav), **обязателен**
- `language` — предпочитаемый язык (например, `ru`, `en`), опционально
- `diarization` — включить определение говорящих, опционально

Response 200 (application/json):

```
{ "job_id": "9f9a9f8c-4f31-4f7a-9c9b-21b2a80d2b65", "status": "processing" }
```

Ошибки: 400, 401, 404, 500

Пример (cURL):

```
curl -X POST http://localhost:8000/api/upload   -F
"file=@/path/to/meeting.m4a"   -F "language=ru"
```

GET /api/status/{job_id}

Получить текущий статус обработки.

Response 200:

```
{ "status": "processing", "progress": 18 }
```

Ошибки: 400, 401, 404, 500

Пример:

```
curl http://localhost:8000/api/status/9f9a9f8c-4f31-4f7a-9c9b-21b2a80d2b65
```

GET /api/result/{job_id}

Скачать результат (Markdown или JSON). Выбор формата через `?format=markdown|json` или заголовок `Accept`.

Response 200 (text/markdown):

```
# Итоги встречи
- Релиз сдвинут на 2 недели
...
```

Response 200 (application/json):

```
{ "title": "Стенограмма совещания", "sections": [{ "heading": "Итоги",
"bullets": ["..."] }] }
```

Ошибки: 400, 401, 404, 500

Пример:

```
curl -H "Accept: text/markdown" http://localhost:8000/api/result/<job_id>
```

WS /api/ws/{job_id}

WebSocket-подключение для событий по задаче.

Типы событий:

- `progress`: `{ "event": "progress", "data": { "progress": 73 } }`
- `status_change`: `{ "event": "status_change", "data": { "old_status": "processing", "new_status": "ready" } }`
- `completed`: `{ "event": "completed", "data": { "result_url": "..." } }`
- `error`: `{ "event": "error", "data": { "message": "..." } }`

Пример (JavaScript):

```
const ws = new WebSocket("ws://localhost:8000/api/ws/9f9a9f8c-4f31-4f7a-9c9b-
21b2a80d2b65");
ws.onmessage = (e) => console.log("WS:", e.data);
```

Авторизация и профиль

POST /api/auth/register

Request:

```
{ "email": "user@example.com", "password": "StrongPassw0rd!", "full_name":
"Алексей Иванов" }
```

Response 200:

```
{ "id": "9f9a9f8c-4f31-4f7a-9c9b-21b2a80d2b65", "email": "user@example.com",
"full_name": "Алексей Иванов", "plan": "free", "usage": {"minutes_used": 0,
"jobs_active": 0} }
```

Ошибки: 400, 401, 404, 500

POST /api/auth/login

Request:

```
{ "email": "user@example.com", "password": "StrongPassw0rd!" }
```

Response 200:

```
{ "access_token": "<jwt>", "refresh_token": "<refresh>", "token_type":
"bearer", "expires_in": 3600 }
```

Ошибки: 400, 401, 404, 500

POST /api/auth/refresh

Request:

```
{ "refresh_token": "<refresh>" }
```

Response 200:

```
{ "access_token": "<jwt>", "refresh_token": "<refresh>", "token_type":
"bearer", "expires_in": 3600 }
```

Ошибки: 400, 401, 404, 500

GET /api/me

Headers: `Authorization: Bearer <access_token>`

Response 200:

```
{ "id": "<uuid>", "email": "user@example.com", "full_name": "Алексей Иванов",
"plan": "free", "usage": {"minutes_used": 12, "jobs_active": 1} }
```

Ошибки: 400, 401, 404, 500

Транскрипты

GET /api/transcripts

Список транскриптов текущего пользователя.

Query: `project`, `tag`, `q`, `limit`, `offset`

Response 200:

```
{
  "total": 2,
  "items": [
    {"id": "<uuid>", "title": "Стенограмма 1", "created_at": "2025-09-
15T10:00:00Z", "language": "ru", "duration_sec": 3600, "tags": ["meeting"]},
    {"id": "<uuid>", "title": "Стенограмма 2", "created_at": "2025-09-
20T10:00:00Z", "language": "en", "duration_sec": 1800, "tags": ["roadmap"]}
  ]
}
```

Ошибки: 400, 401, 404, 500

GET /api/transcripts/{id}

Возврат Markdown, JSON или только метаданных (через `?format=markdown|json|meta`).

Response 200 (application/json):

```
{ "id": "<uuid>", "title": "Стенограмма", "created_at": "2025-09-
15T10:00:00Z", "content_url": "https://.../result.md", "json_url":
"https://.../result.json" }
```

Ошибки: 400, 401, 404, 500

POST /api/transcripts/{id}/tags

Добавить/заменить теги транскрипта.

Request:

```
{ "tags": ["finance", "okrs"] }
```

Response 200: объект транскрипта с обновлёнными тегами.
Ошибки: 400, 401, 404, 500

Пример:

```
curl -X POST http://localhost:8000/api/transcripts/<id>/tags   -H
"Authorization: Bearer $TOKEN"   -H "Content-Type: application/json"   -d
'{"tags": ["finance","okrs"]}'
```

DELETE /api/transcripts/{id}

Удалить транскрипт.

Response 200:

```
{ "deleted": true }
```

Ошибки: 400, 401, 404, 500

Обработка ошибок

Все ошибки возвращаются в формате:

```
{ "error": "Сообщение об ошибке" }
```

Коды: `400` (ошибка запроса), `401` (неавторизован), `404` (не найдено), `500` (внутренняя ошибка).

Ограничения и совместимость

- Форматы: mp3/m4a/wav
- Лимиты MVP: ≤ 50 МБ, ≤ 120 минут
- Бэкенд: FastAPI + Uvicorn; хранилище: Postgres; обработка: ffmpeg, VAD, ASR, постпроцессинг; сборка Markdown.