Генерация плейлиста по выбранным трекам

Костров Вячеслав
Ганыч Даниил
Сараев Никита

Описание задачи

Алгоритм действий пользователя

01

Пользователь авторизуется с аккаунтом Spotify

02

Пользователь выбирает список своих треков для генерации плейлиста

03

Пользователю возвращается сгенерированный плейлист

04

Пользователю предлагается автоматически добавить плейлист к себе в аккаунт

Состав команды

Олег Шевченко

Роль: куратор

Выполненные задачи:

- Ревью результатов работы команды.
- Помощь команде с выбором используемых подходов и технологий.

Костров Вячеслав

Роль: разработчик

Выполненные задачи:

- Создание и настройка репозитория.
- Реализация функционала для работы с данными.
- Реализация сервиса.

Ганыч Даниил

Роль: разработчик

Выполненные задачи:

- Реализация загрузчика треков в MP3.
- Реализация функционала для работы с моделью.
- Реализация финальной версии бейзлайна.

Сараев Никита

Роль: разработчик

Выполненные задачи:

- Реализация парсера меты Spotify.
- Выбор и реализация метрик.
- Реализация фронтенд интерфейса.

Командный workflow

- Репозиторий (remote): **GitHub.**
- Менеджер зависимостей: **Poetry.**
- Виртуальные окружения: **Poetry & conda.**
- Качество кода: **ruff & pre-commit.**
- CI проверки: **GitHub Actions.**

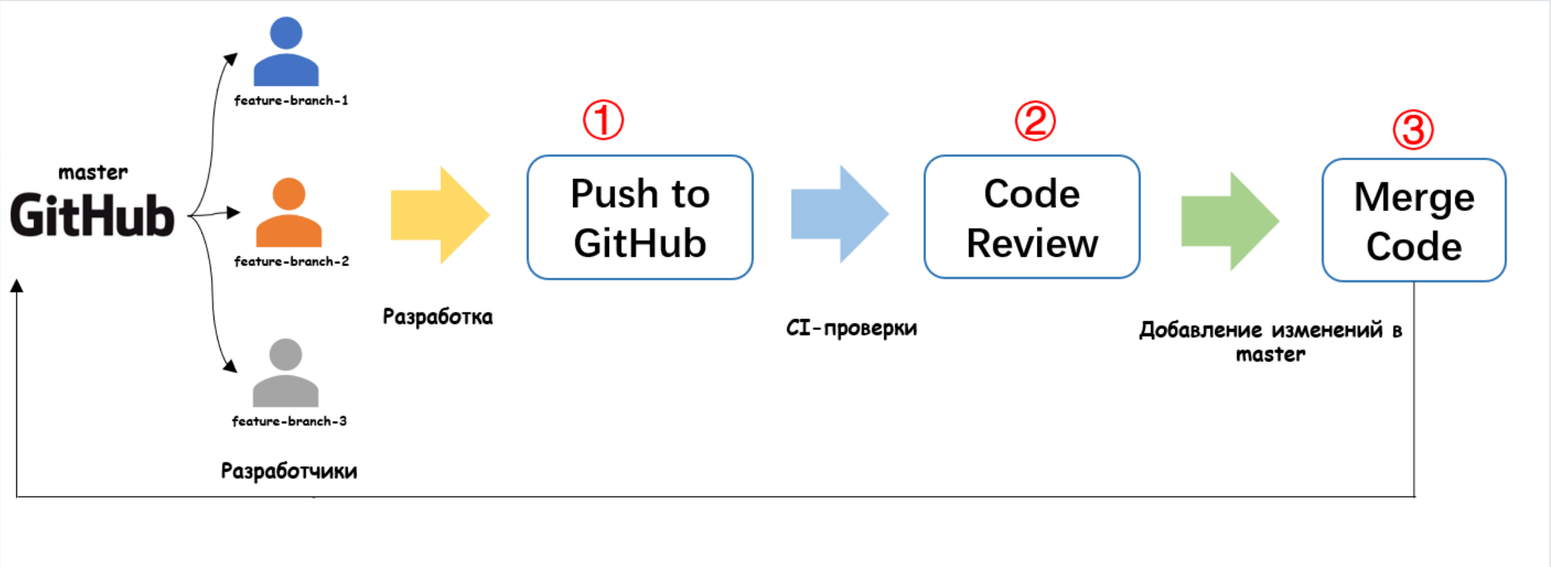


Рисунок 1. Git-flow

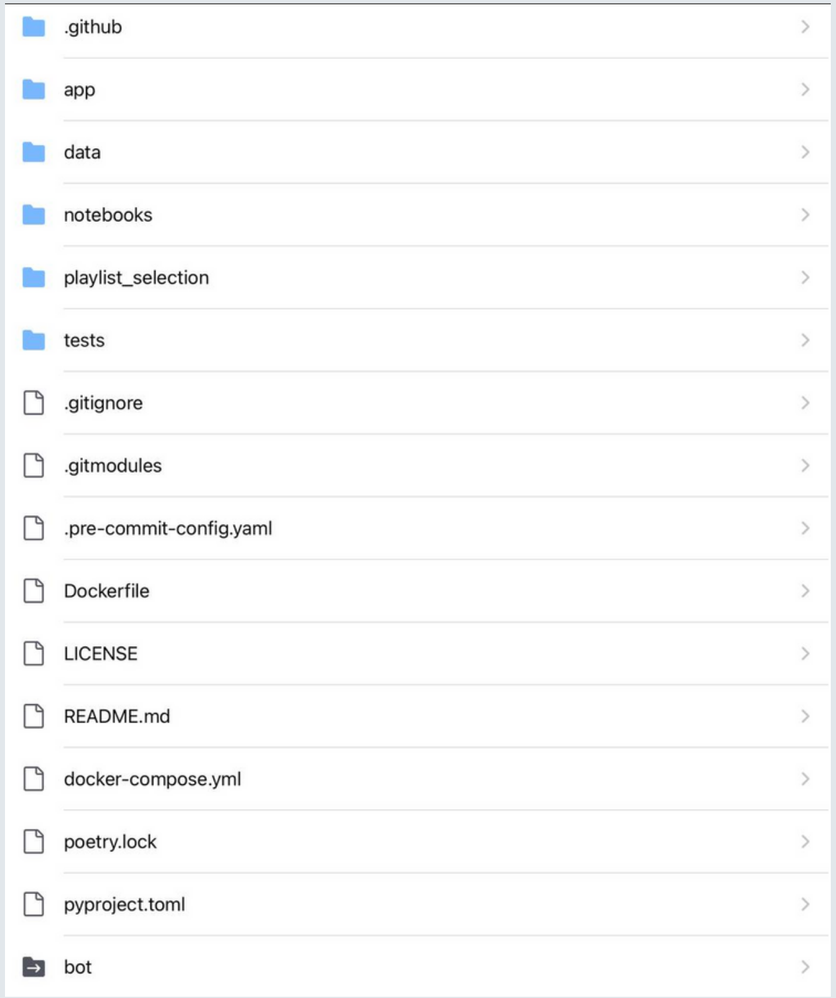
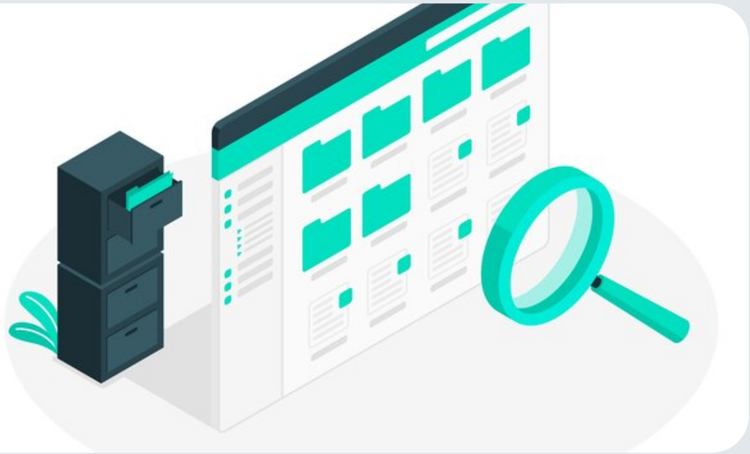


Рисунок 2. Структура репозитория

Описание данных

Мета Spotify

В данный момент основным источником признаков является мета информация о треках, собранная из Spotify API и в дальнейшем обработанная для извлечения дополнительных признаков.



Собранные признаки:

- Базовая информация о треках (исполнитель, альбом, год выпуска и тд).
- Дополнительное признаковое описание Spotify: популярность, громкость, «танцевальность», «энергия» и др.
- Собранные после постобработки: количество и средняя длительность сегментов/секций/битов.
- MP3 дорожка трека.

Хранение данных

Все данные хранятся на Yandex S3.
Данные лежат в формате tracks/<genre>/<artist_track>/{mp3, meta}

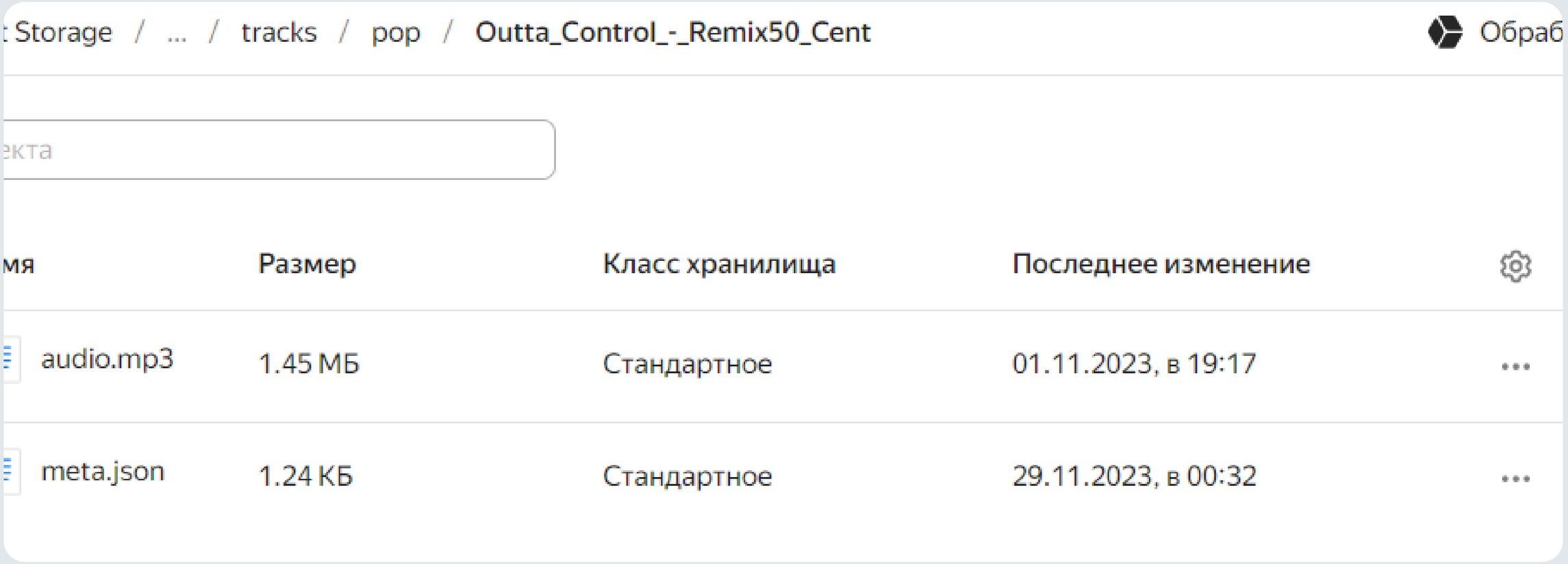


Рисунок 3. Хранение данных в S3

Валидация данных

Валидация данных осуществляется при помощи pydantic models. Реализованы модели для базовой информации о треке (исполнитель, альбом и т.д), а так же дополнительная вложенная модель для более подробных аудио признаков Spotify.

```python
65
66  class TrackMeta(BaseModel):
67      """Track meta info."""
68
69      album_name: str | None = Field(default=None, repr=True)
70      album_id: str | None = Field(default=None, repr=False)
71      album_release_date: str | None = Field(default=None, pattern=r'\d{4}-\d{2}-\d{2}', repr=False)
72      artist_name: list[str] = Field(default_factory=list, repr=True)
73      artist_id: list[str] = Field(default_factory=list, repr=False)
74      track_id: str = Field(default="unknown", repr=False)
75      track_name: str = Field(default="unknown", repr=True)
76      # TODO: Подумать как будем собирать жанры, в Spotify есть только для альбомов и очень не для всех
77      genres: list[str] = Field(default_factory=lambda : ["unknown"], repr=True)
78      track_details: TrackDetails = Field(default_factory=TrackDetails, repr=False)
```

Рисунок 4. Pydantic модель для мета-информации трека

Валидация собранных признаков

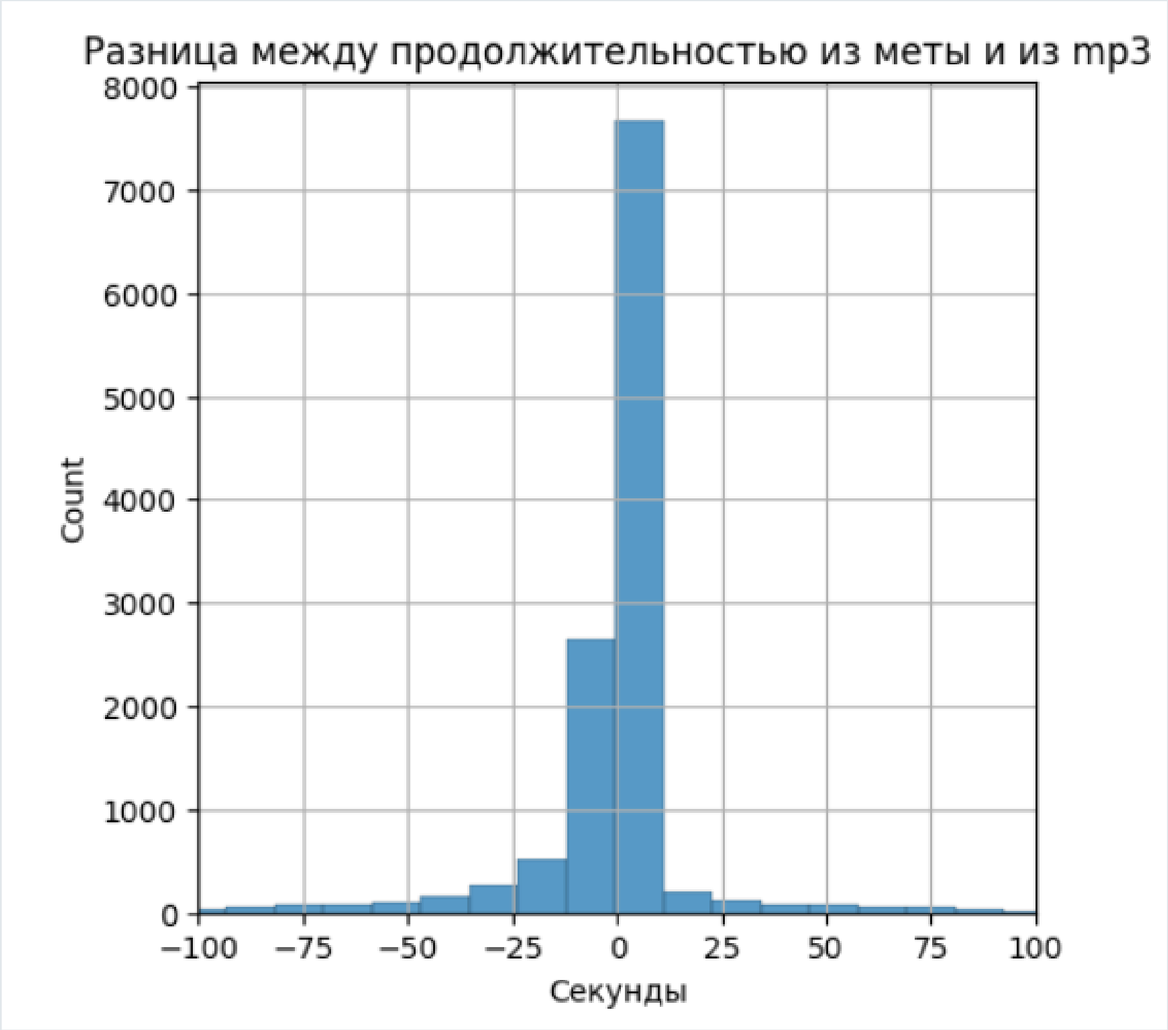


Рисунок 5. Разница продолжительности

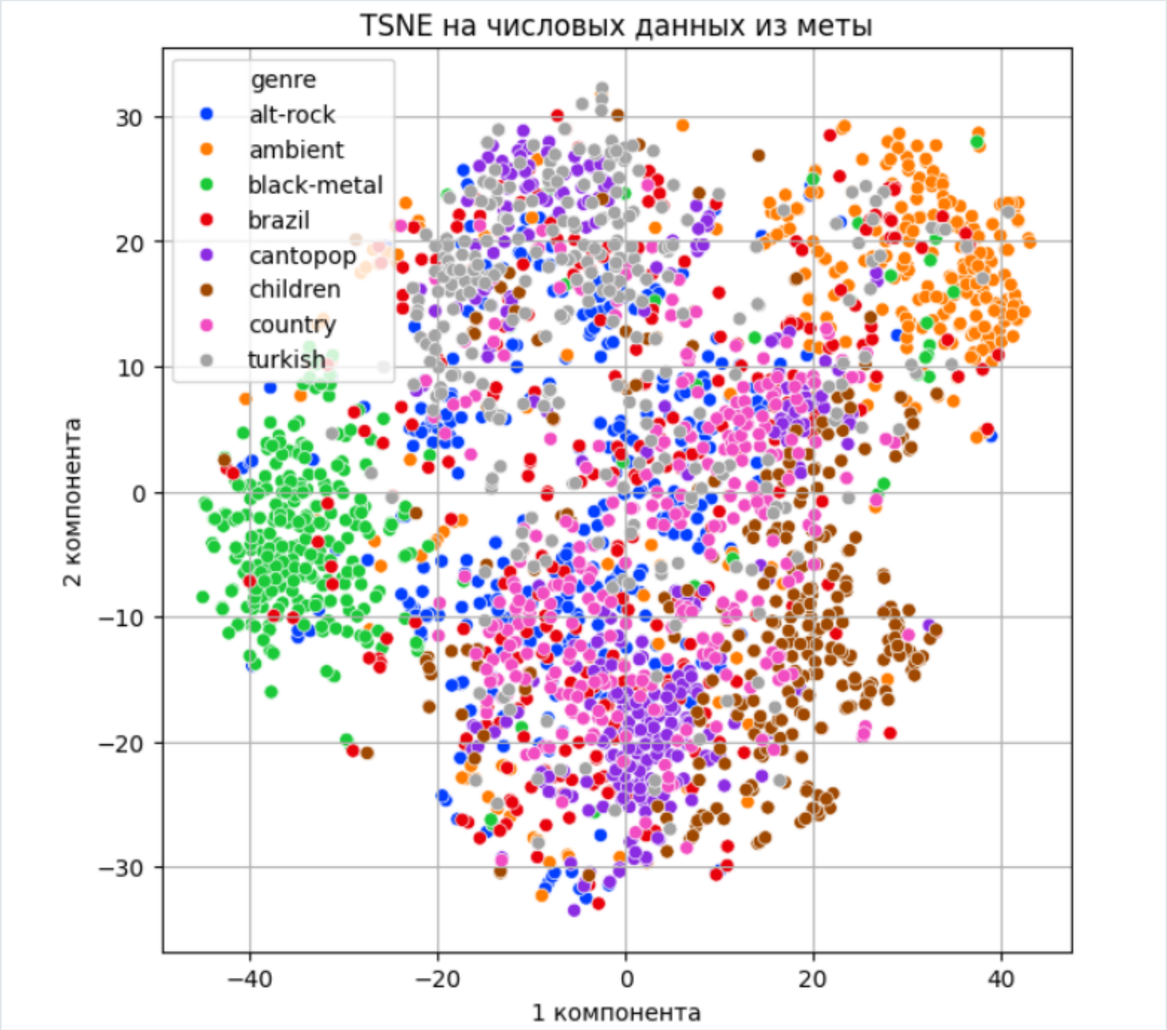


Рисунок 6. TSNE для числовых признаков

Распределение по жанру

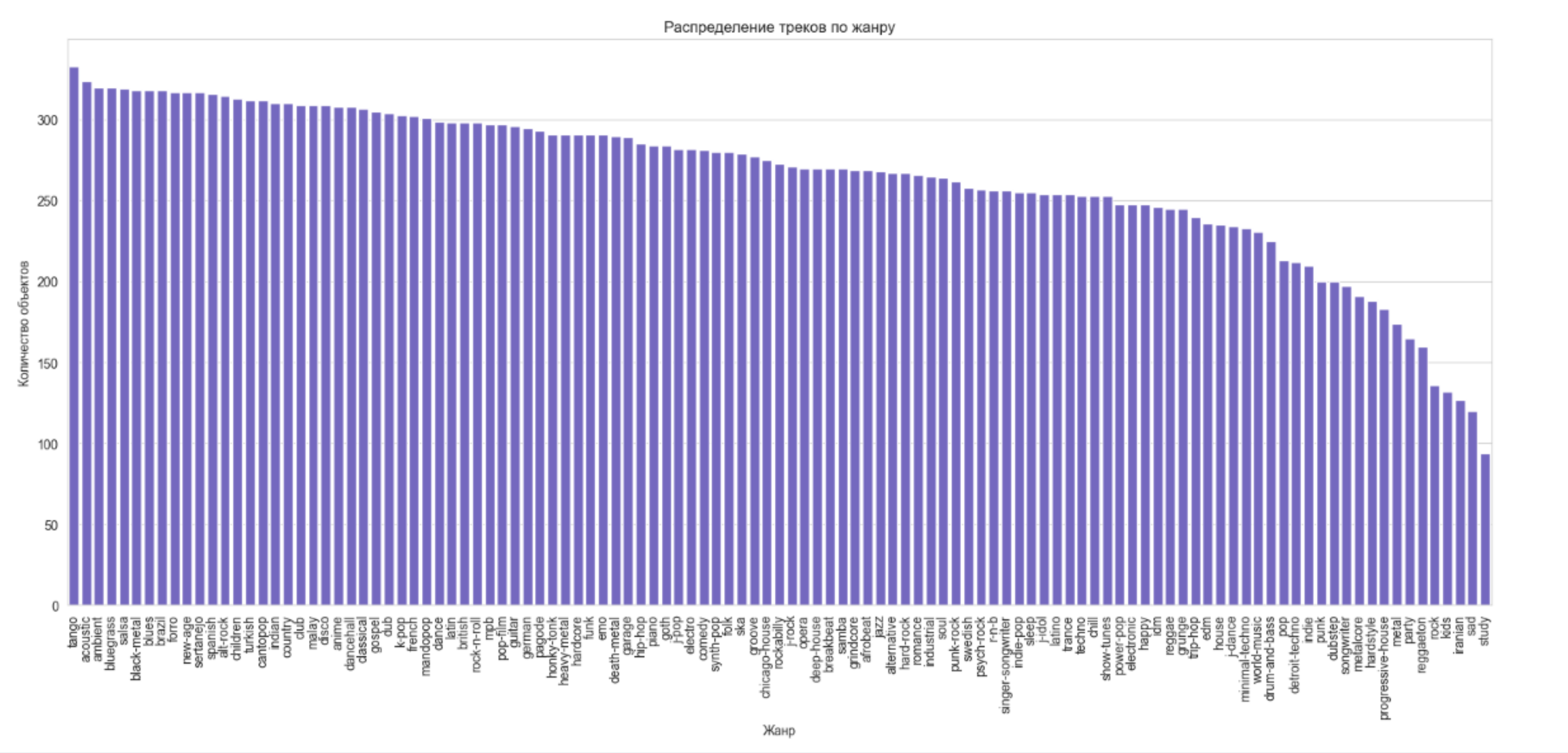


Рисунок 7. Распределение треков по жанрам

Распределение по году выпуска

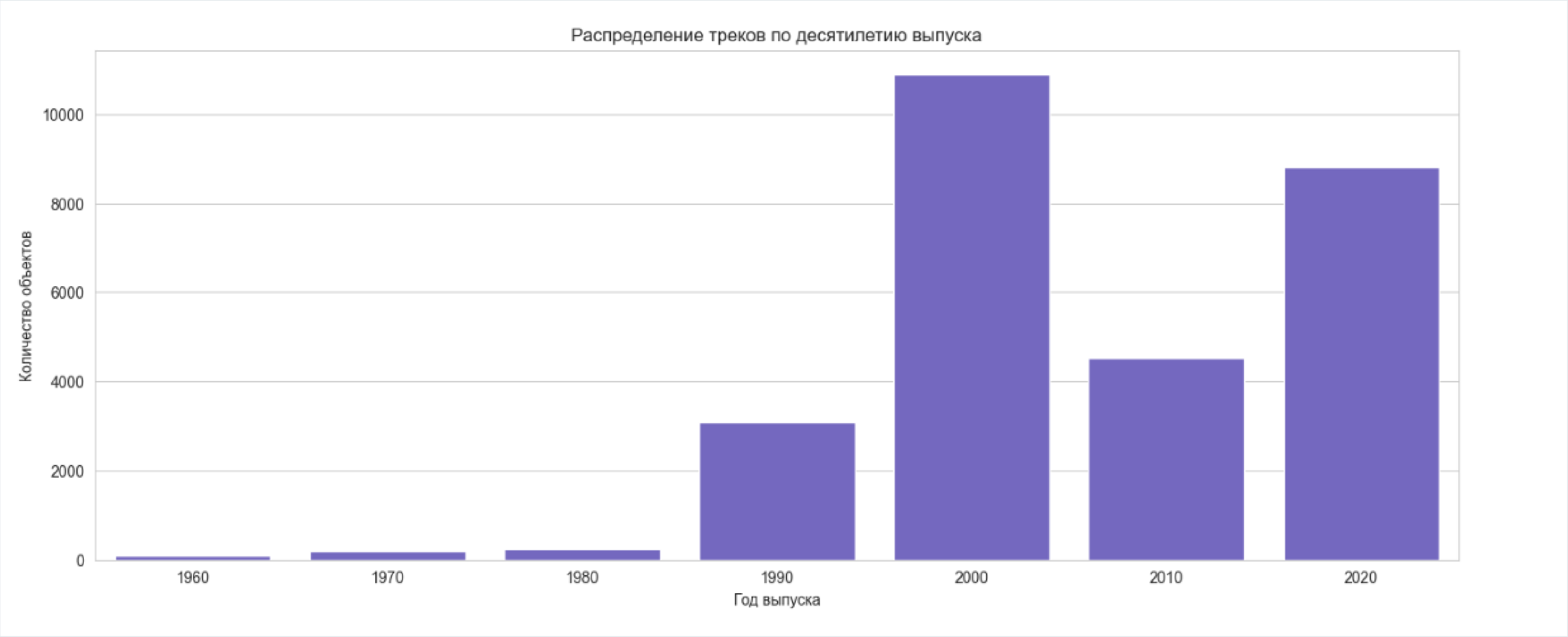


Рисунок 8. Распределение треков по году выпуска

Распределение по длительности

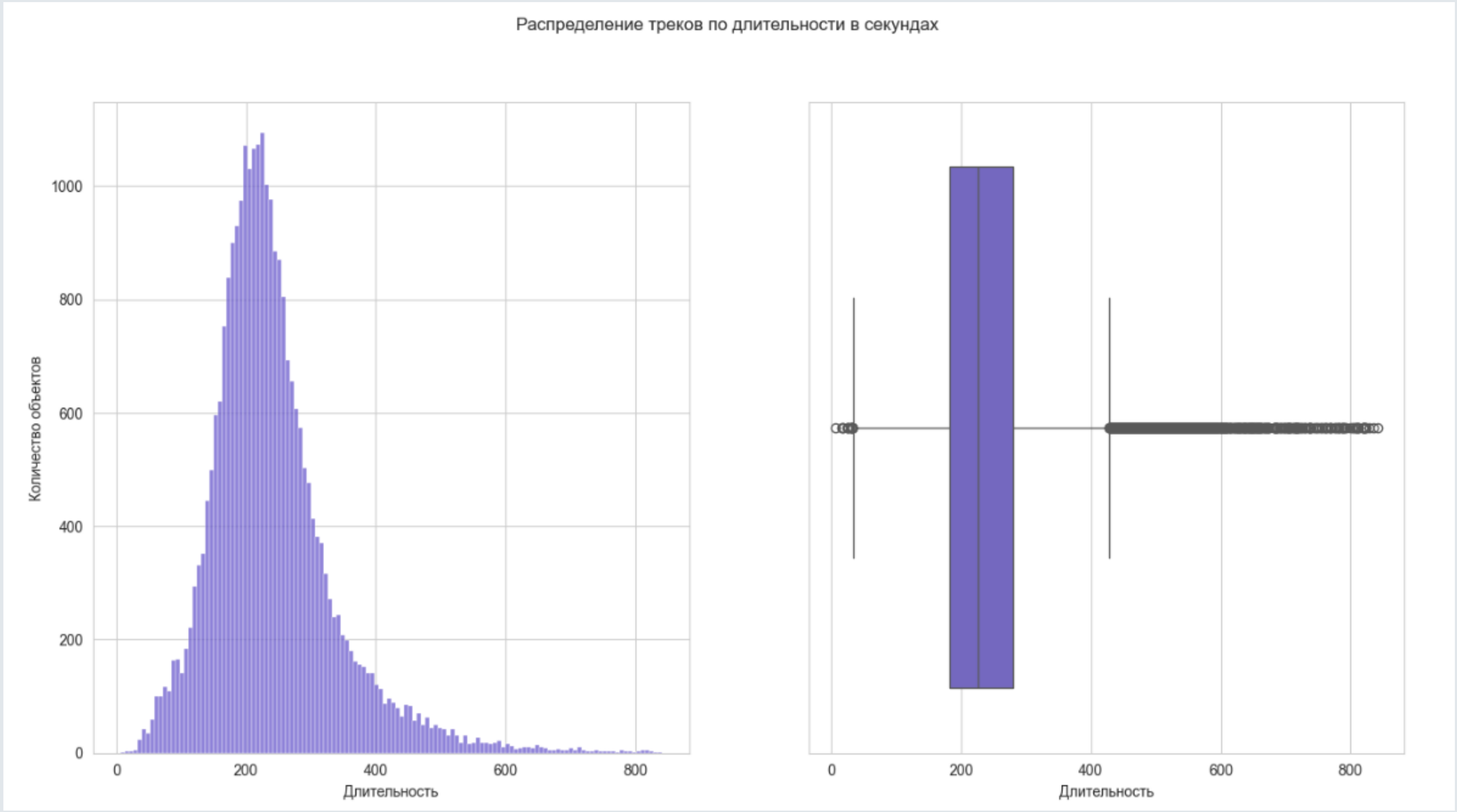


Рисунок 9. Распределение треков по длительности

Распределение аудио-фичей

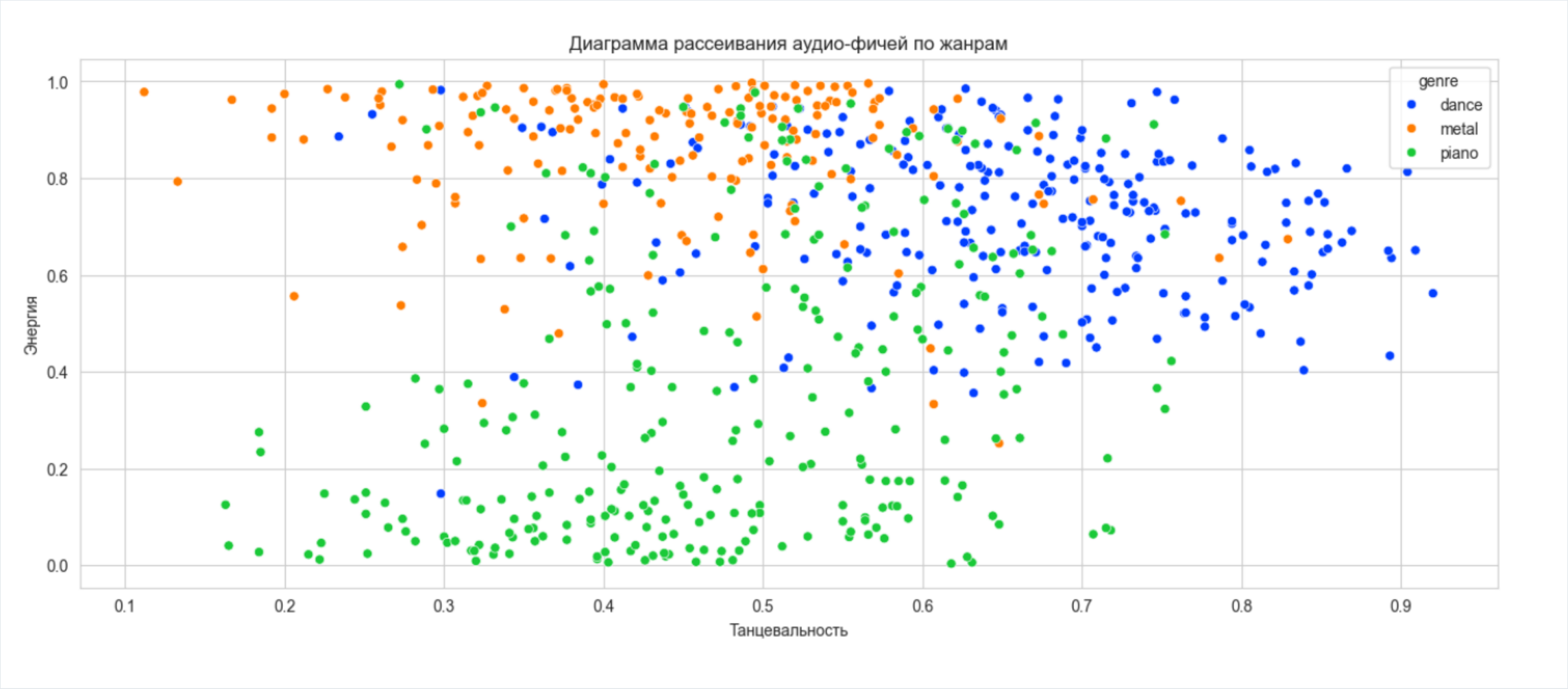


Рисунок 10. Диаграмма рассеивания для “энергии” и “танцевальности”

Описание метрик

Название метрики	Описание
CorrectGenreShare	Доля правильно угаданных жанров
CorrectAlbumShare	Доля правильно угаданных альбомов
CorrectArtistShare	Доля объектов, где угадан хотя бы один исполнитель
YearMeanDiff	Средняя разница в годе релиза
SoundParametersDiff	L2 норма для относительной разницы параметров звука

Описание модели

Пайплайн предобработки

Из всех признаков оставляются аудио-признаки Spotify, жанр и год выпуска трека.

Обработка категориальных признаков:

1. Используется OneHotEncoding

Обработка числовых признаков:

1. NaN значения заменяются средним
2. Признаки нормируются на среднее и дисперсию

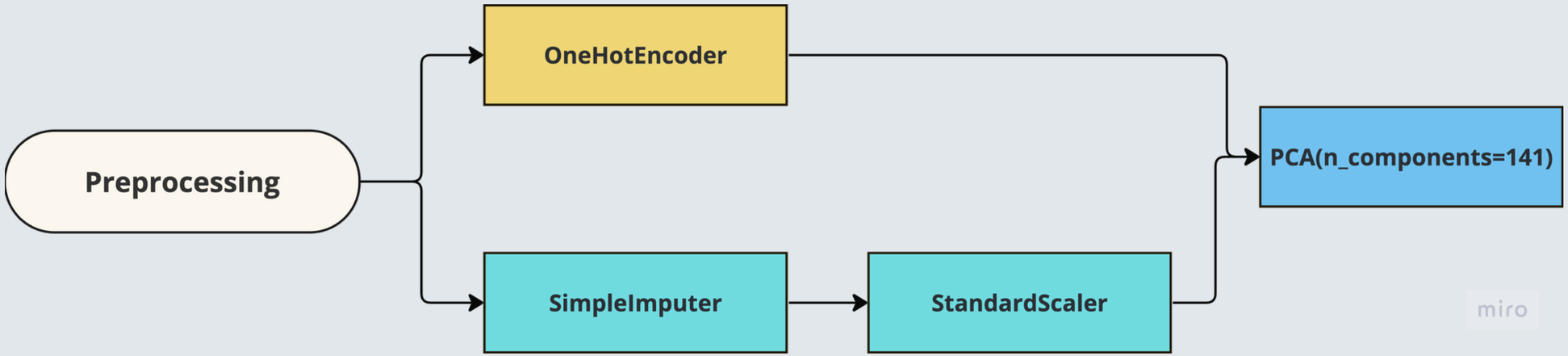
Далее для полученных признаков применяется PCA, оставляя 141 признак.

Реализация модели

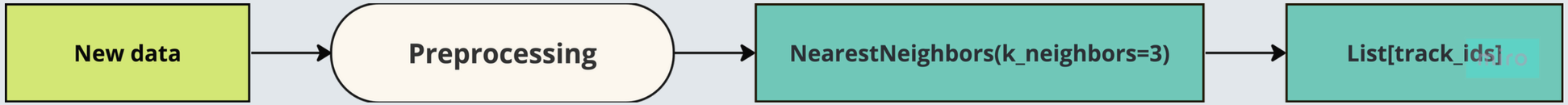
Как модель используется sklearn реализация поиска ближайших соседей с манхэттенской метрикой. В данный момент для каждого трека возвращается 3 соседа.

Описание модели

Пайплайн предобработки

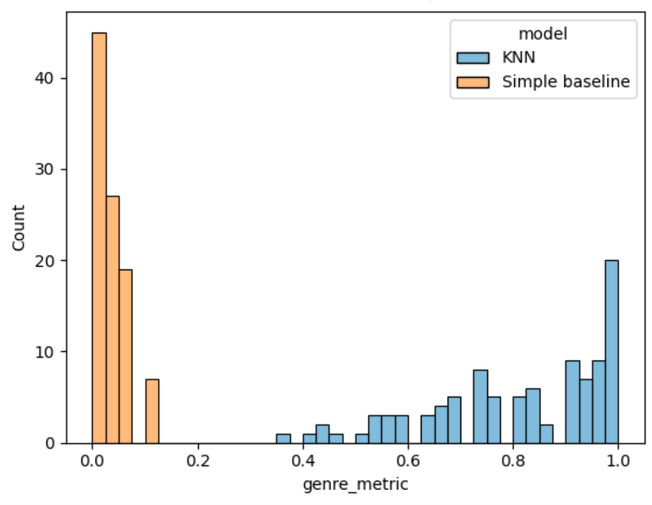


Реализация модели

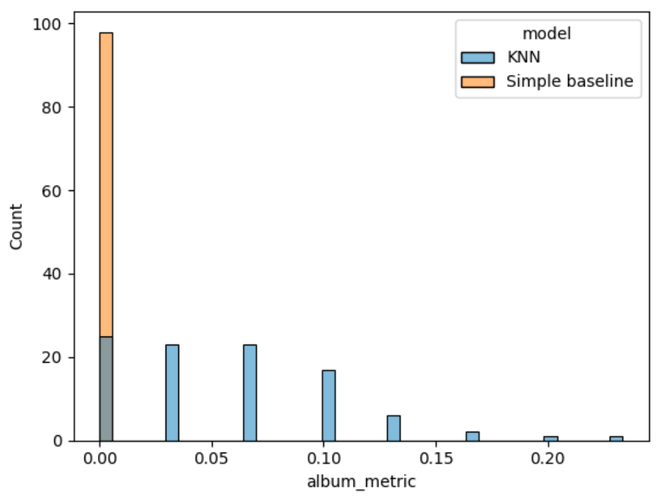


Метрики (KNN vs Random)

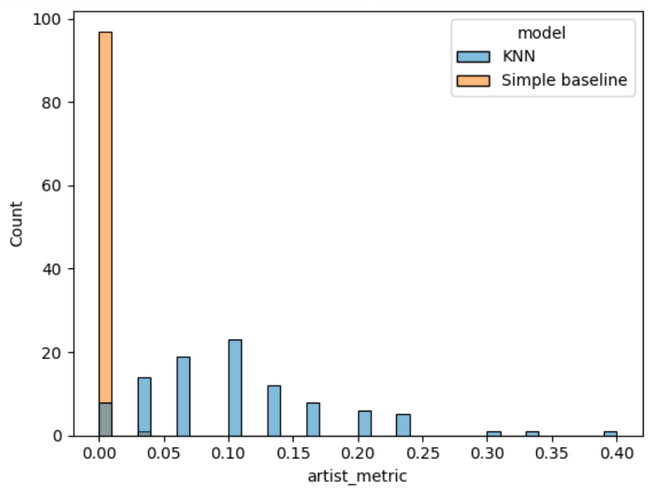
Угадан жанр



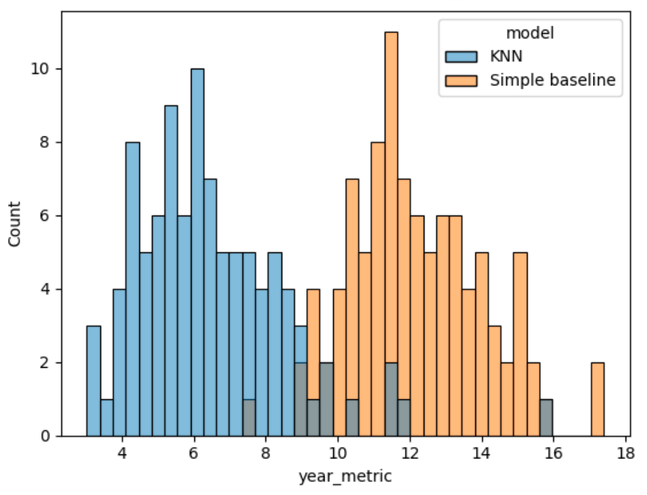
Угадан альбом



Угадан артист



Разница в годе выпуска



L2 норма разницы параметров звука

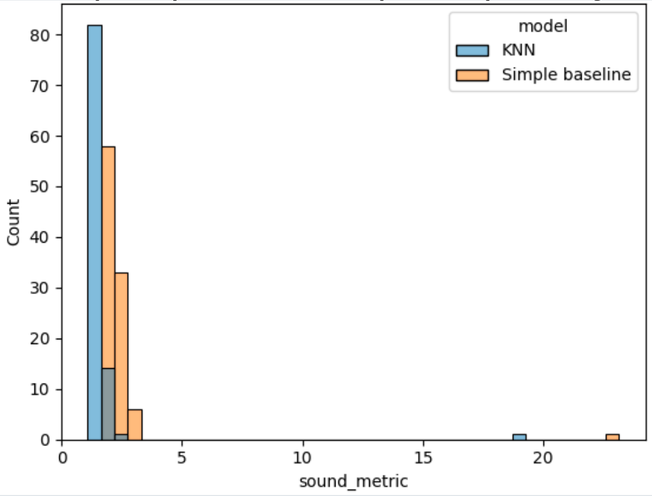
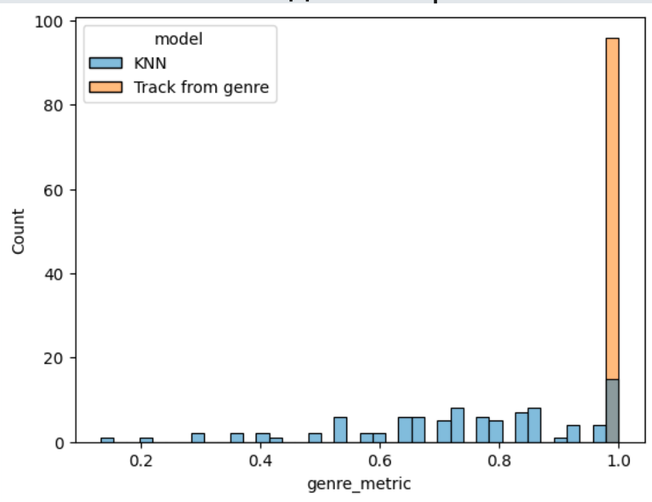


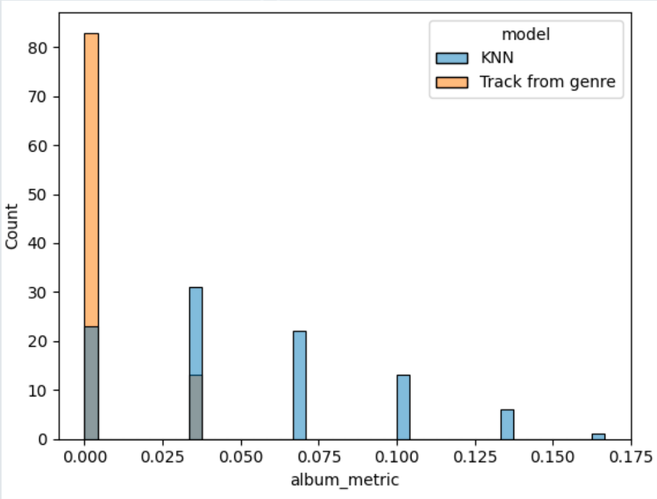
Рисунок 11. Сравнение метрик KNN и случайной рекоммендации

Метрики (KNN vs Genre tracks)

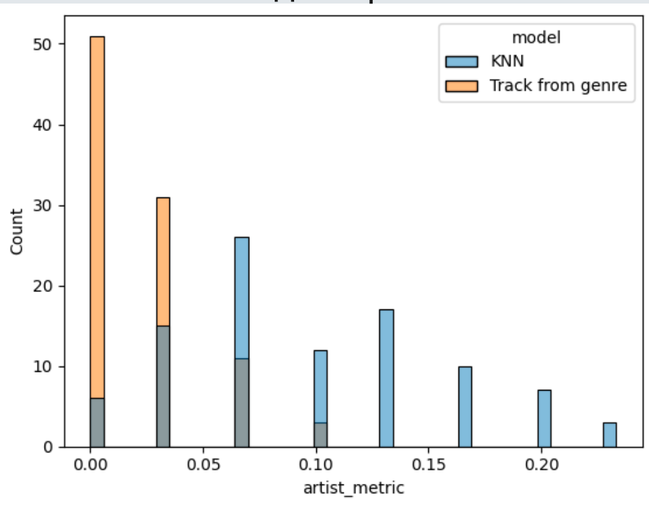
Угадан жанр



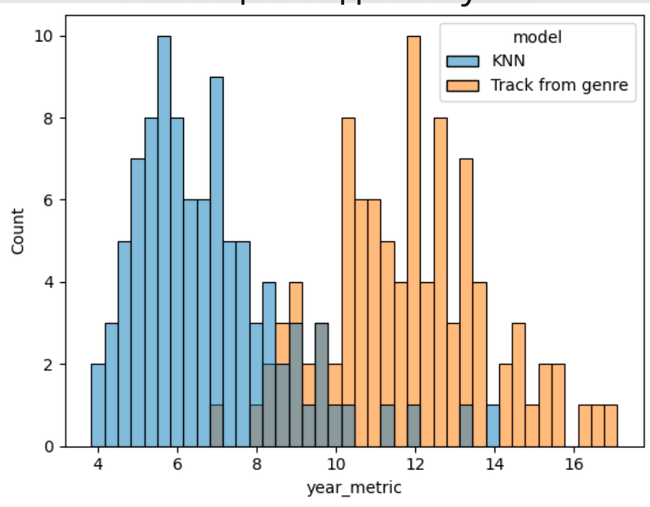
Угадан альбом



Угадан артист



Разница в годе выпуска



L2 норма разницы параметров звука

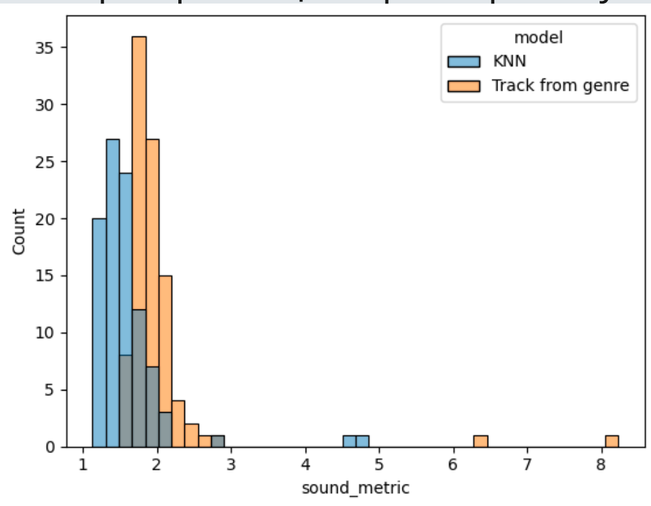


Рисунок 12. Сравнение метрик KNN и рекоммендации по жанру

Описание сервиса

Принцип работы

Для получения информации о треках пользователя требуется Spotify токен, который хранится в БД сервиса.
Для конкретного пользователя ключ к БД хранится в cookie.

При поступлении запроса на генерацию:

1. Подтягивается модель из S3
2. С помощью парсера собирается информации о входных треках
3. Происходит инференс и передача треков на фронтенд пользователю

Используемые технологии

FastAPI - реализация сервиса
Redis - встроенная БД
Jinja - интеграция фронтенда
Yandex S3 - хранение модели и данных

Авторизация (OAuth)

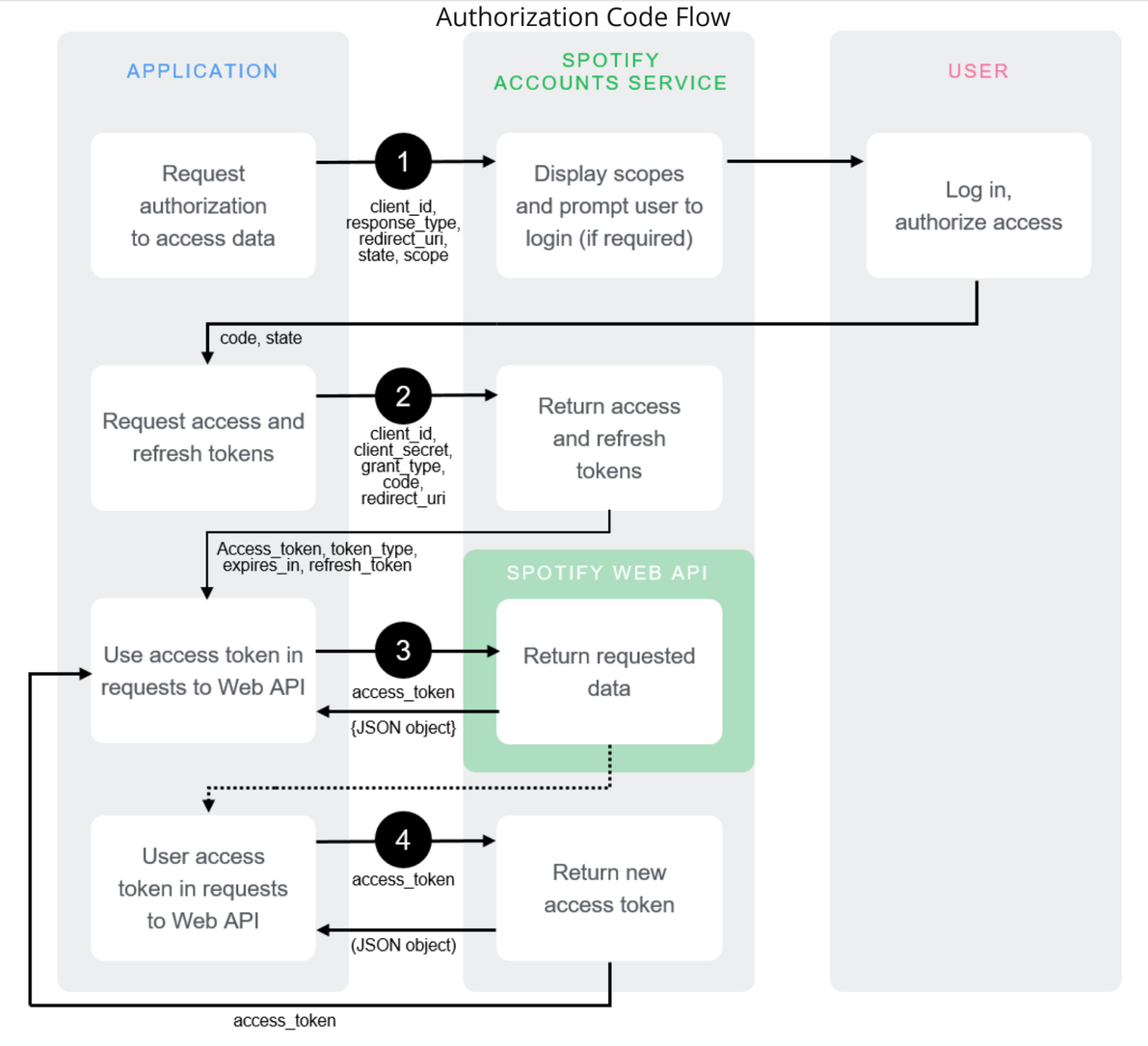


Рисунок 13. Схема авторизации

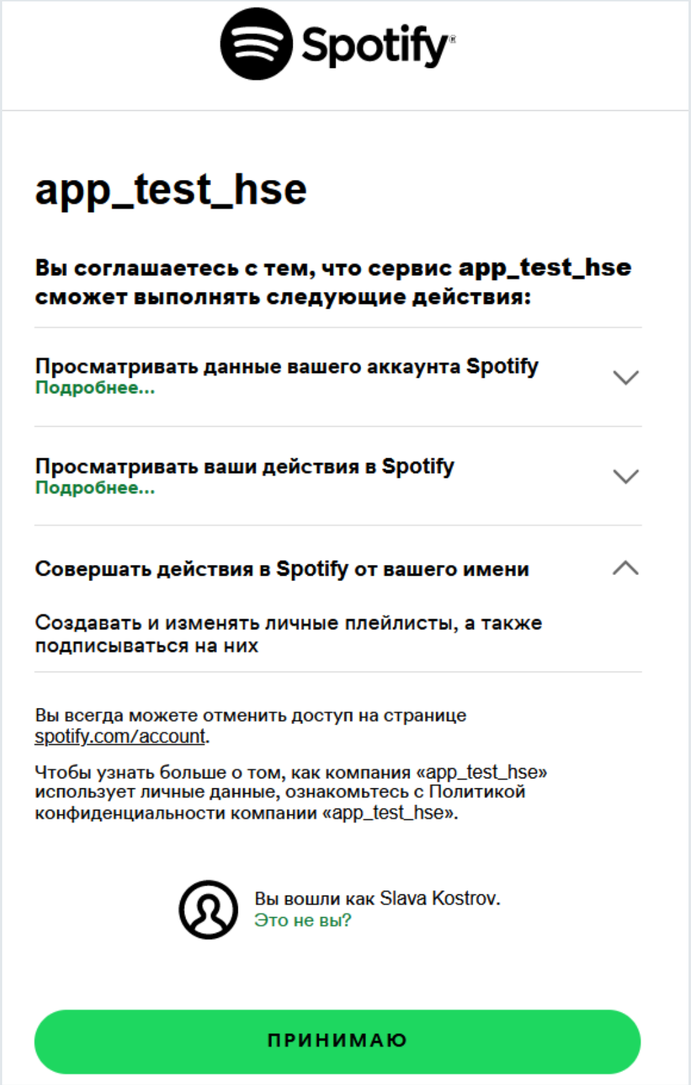


Рисунок 14. Пример авторизации

Архитектура сервиса

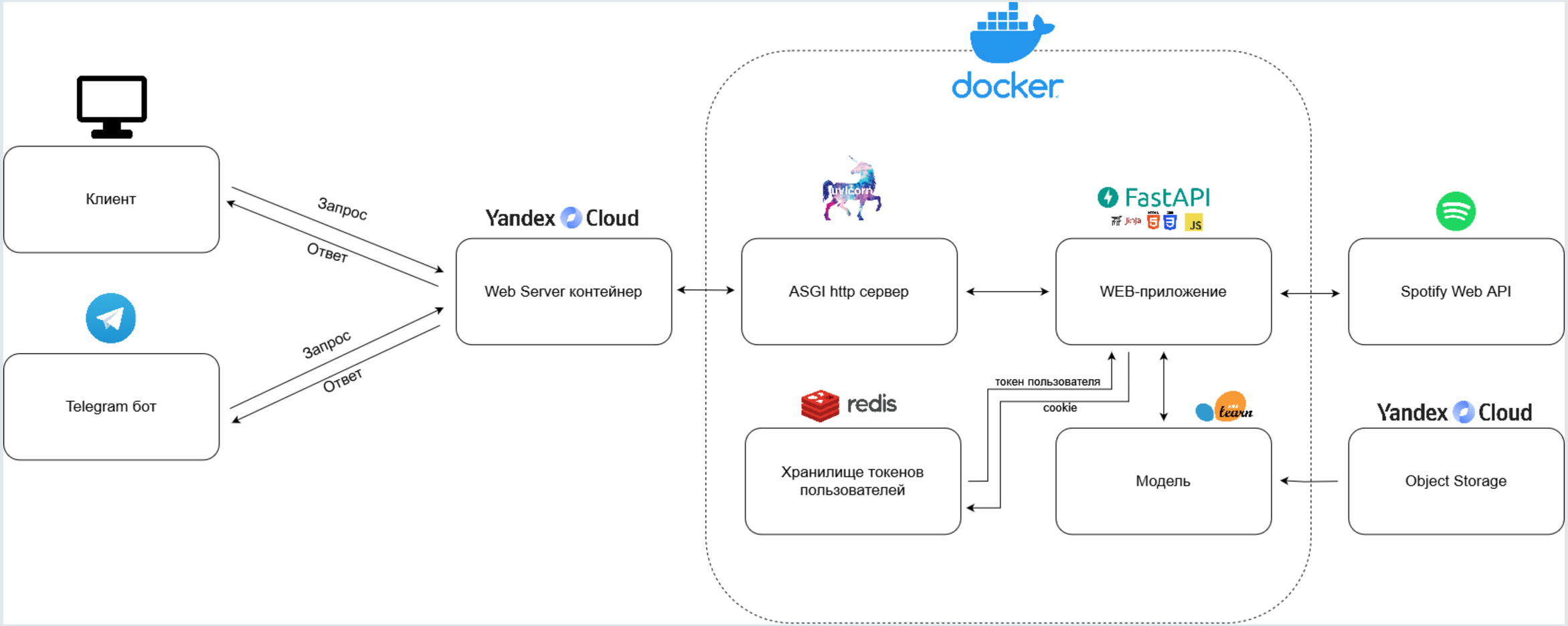
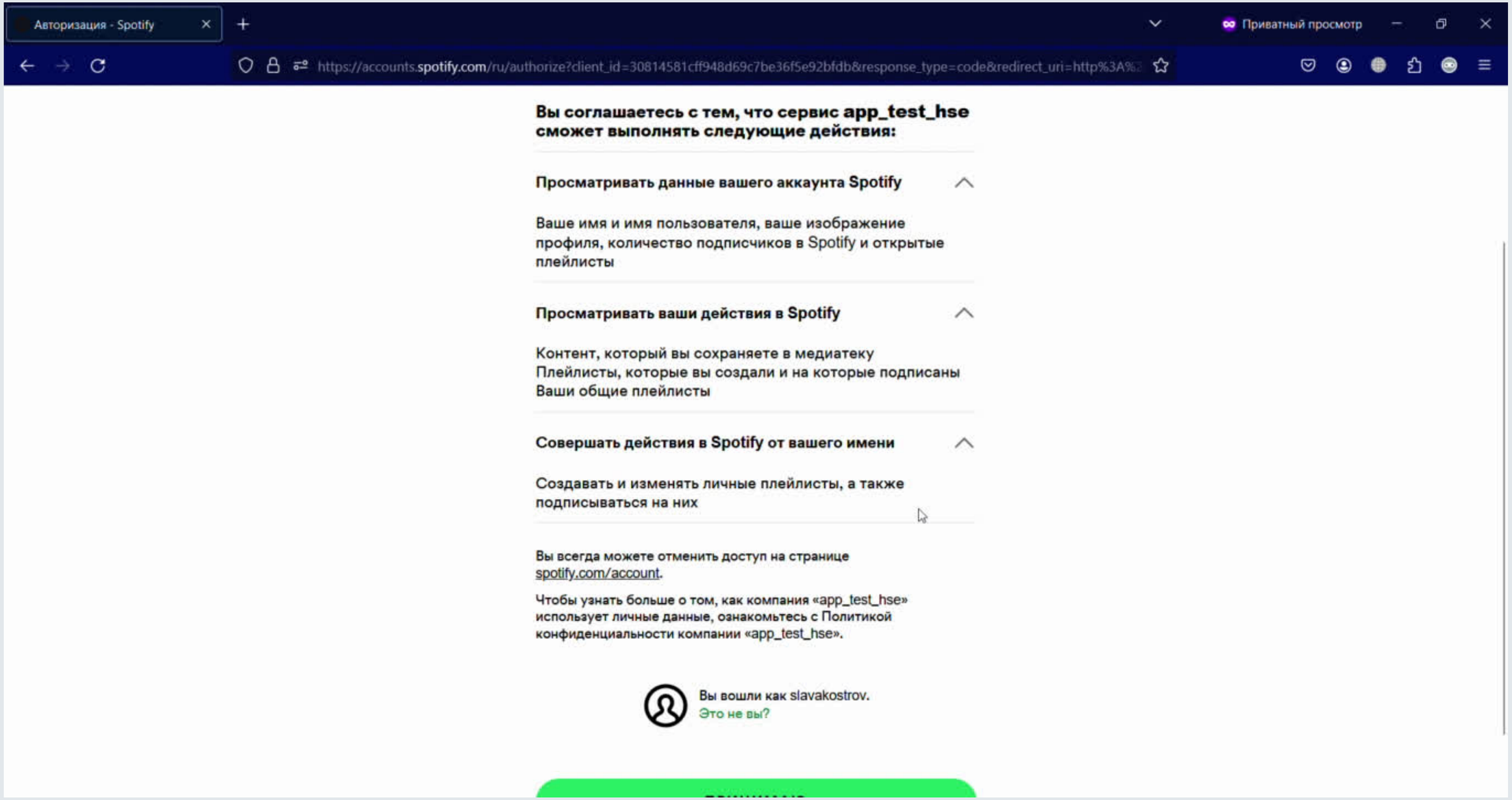


Рисунок 15. Архитектурная схема

Пример работы



Telegram бот

@playlist_selection_bot

Основные функции:

/predict *<artist - track_name>* - предсказывает треки по указанному треку или списку

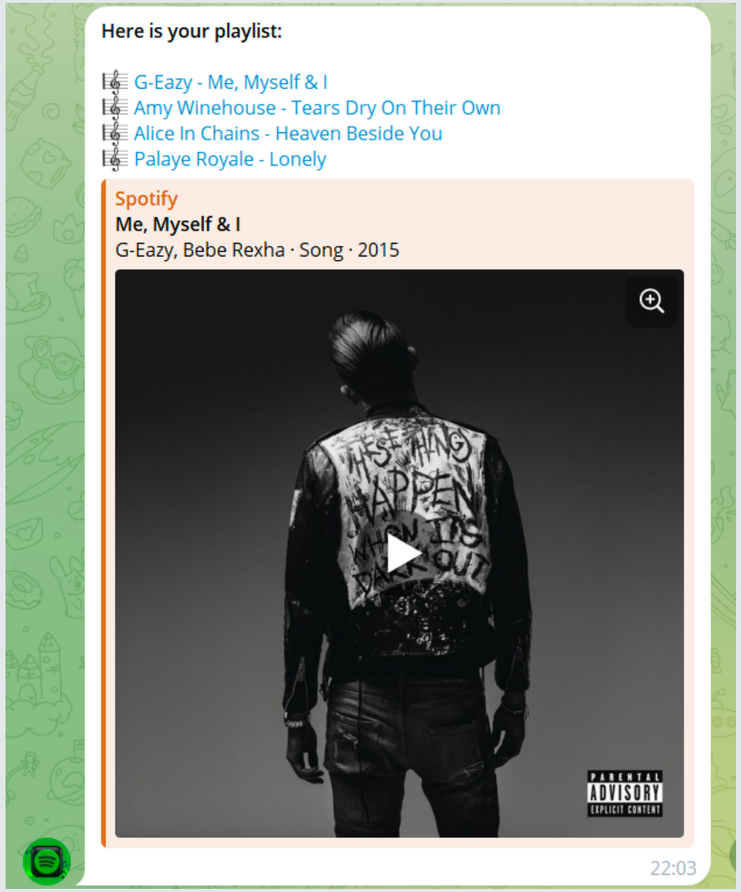


Рисунок 16. Пример работы /predict

/describe *<artist - track_name>* - получить информацию о треке

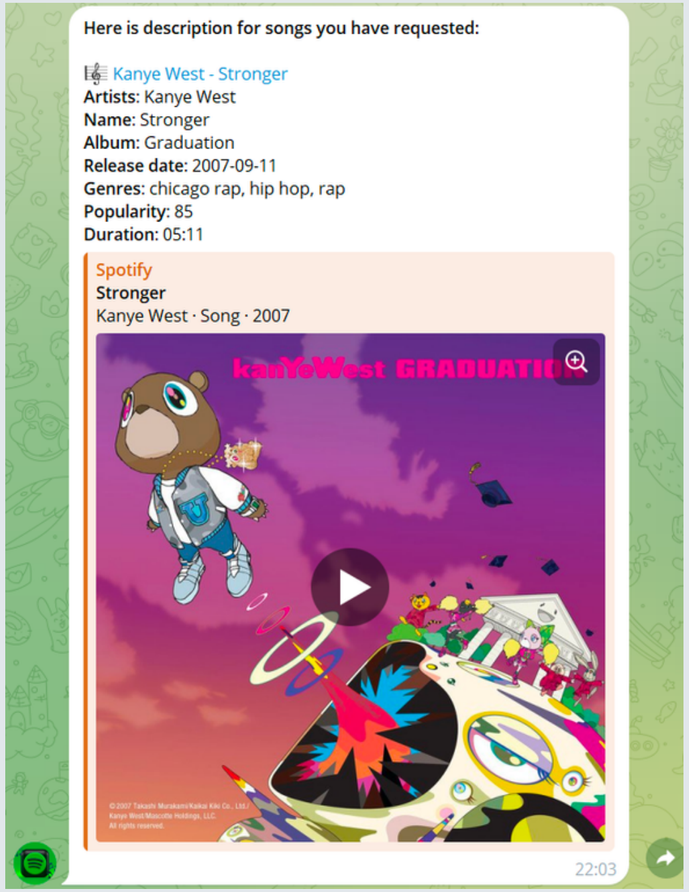
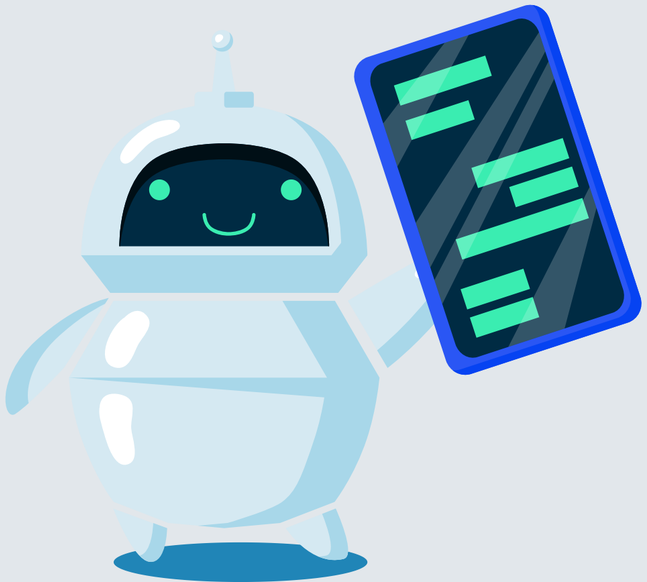


Рисунок 17. Пример работы /describe



Дальнейшие планы

Модель

1. Реализовать модель и метрики ранжирования
2. Обучить и добавить в модель эмбеддинги аудио дорожек

Сервис

1. Оптимизировать векторный поиск, например faiss
2. Улучшить UI
3. *Добавить интеграцию с другими сервисами по образу Spotify

Данные

1. Добрать наиболее популярных треков
2. Добрать разметку для существующих треков

Вопросы