Как с этим работать

Структура проекта

Состоит из Бэкенда -- music_predictor_backend и фронтеда -- music_predictor_stramlit.

Бэкенда структура

```
1. dto - модели для api
2. repository - папка для работы с моделями
3. routes - пути api
4. services - бизнес-логика приложения.
Преобразования данных пользователя в данные приложения
5. settings - содержит глобальную конфигурацию бэкенда
```

Streamlit структура

```
1. dto - модели для api
2. services - бизнес-логика приложения. Кнопочки
3. settings - содержит глобальную конфигурацию
```

API

Бэкенда

Состоит из ручек:

1. /api/v1/upload_dataset - загрузка датасета на сервер
2. /api/v1/make_eda - делает обработку датасета
3. /api/v1/fit_model - обучает модель. Дает результаты обучения
4. /api/v1/get_labels - получить классы датасета
5. /api/v1/set_dataset_name - сохранить датасет с именем
6. /api/v1/get_datasets_name - получить имена датасетов
7. /api/v1/models_names - получить имена моделей на сервере
8. /api/v1/save_predict_file - сохранить модель с имененм
9. /api/v1/predict - предсказать имя модели
10. /api/v1/save_model_name - сохранить модель по имени

Всё можно посмтреть в swagger.yaml

Docker

Собрать и запустить

```
docker compose --profile monitoring up -d --build
```

Без системы мониторинга

```
docker compose up -d --build
```

Мониторинг

Сервис отправляет логи в Loki.

Для просмотра логово, используется сервис Графана.

Нативно

Запуск

```
poetry shell
poetry install
python music_predictor_backend/main.py
streamlit run music_predictor_stramlit/client.py
```

Редактирование

Добавление либы

```
poetry add lib
```

Подготовка кода к коммиту

```
black music_predictor_backend
black music_predictor_streamlit

pre-commit
```