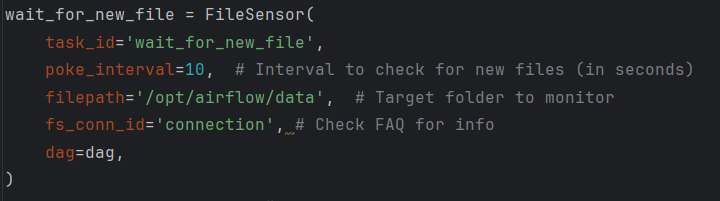
Отчёт 2 лабораторной работы по дисциплине “Инженерия данных” *выполнил Доружинский Дмитрий из группы 6233-010402D.*

# Пайплайн для инференса данных

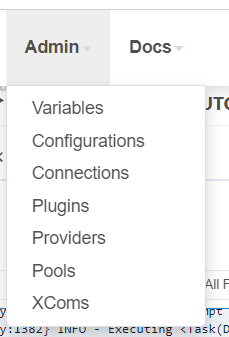
В 1 части лабораторной работы предлагается подстроить пайплайн со следующими этапами:

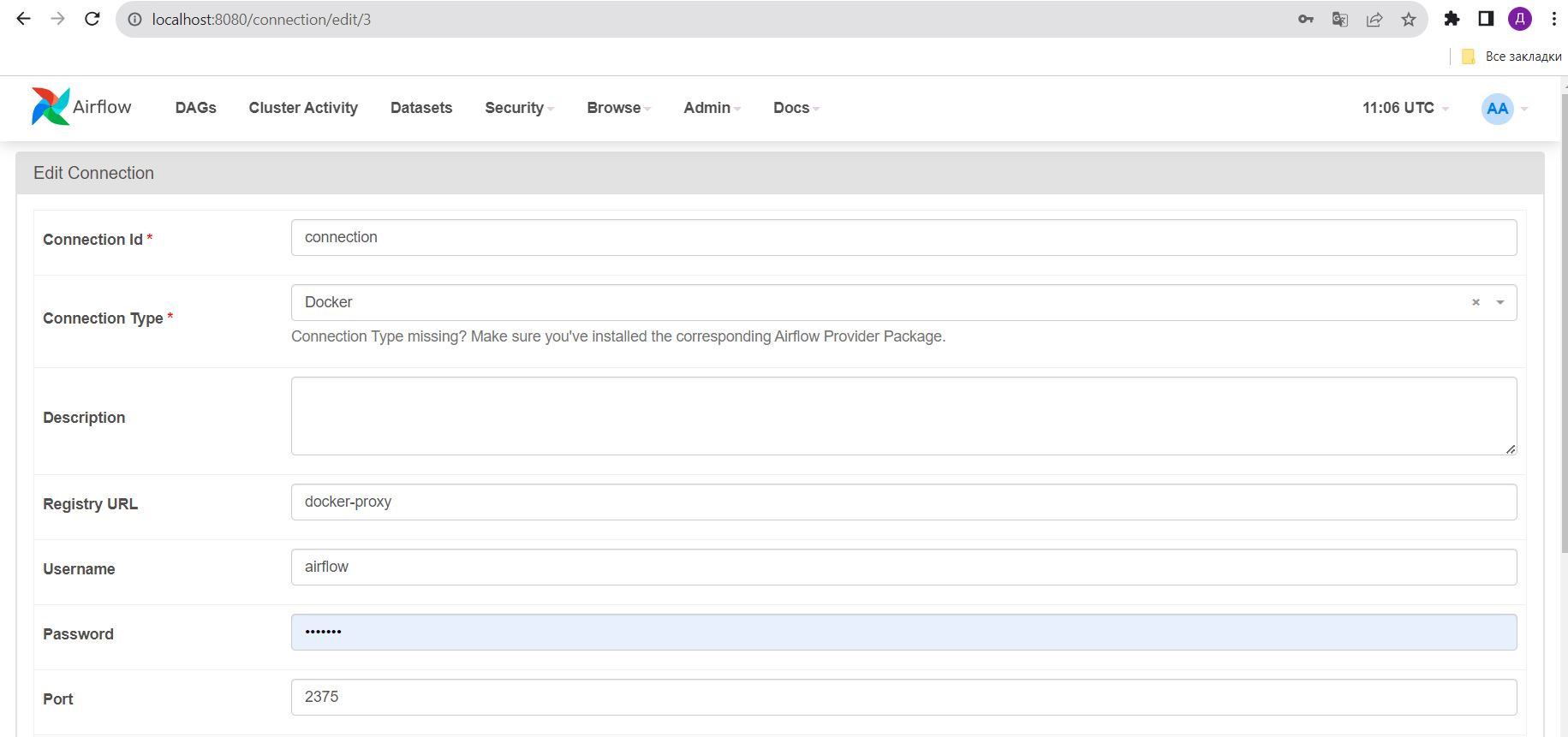
1. Производить мониторинг целевой папки на предмет появления новых видеофайлов.
2. Извлекать аудиодорожку из исходного видеофайла.
3. Преобразовывать аудиодорожку в текст с помощью нейросетевой модели.
4. Формировать конспект на основе полученного текста.
5. Формировать выходной .pdf файл с конспектом.

Пройдусь по каждому из них. В 1 пункте, как было предложено использовалась следующая конструкция.

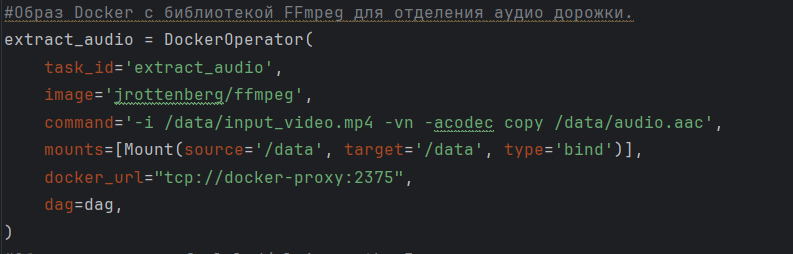


В параметре fs\_conn\_id необходимо указать имя, которое после необходимо указать при создание connections в выпадающем списке Admin графического интерфейса Airflow. Параметр poke\_interval задает интервал времени в секундах обновления просмотра новых файлов



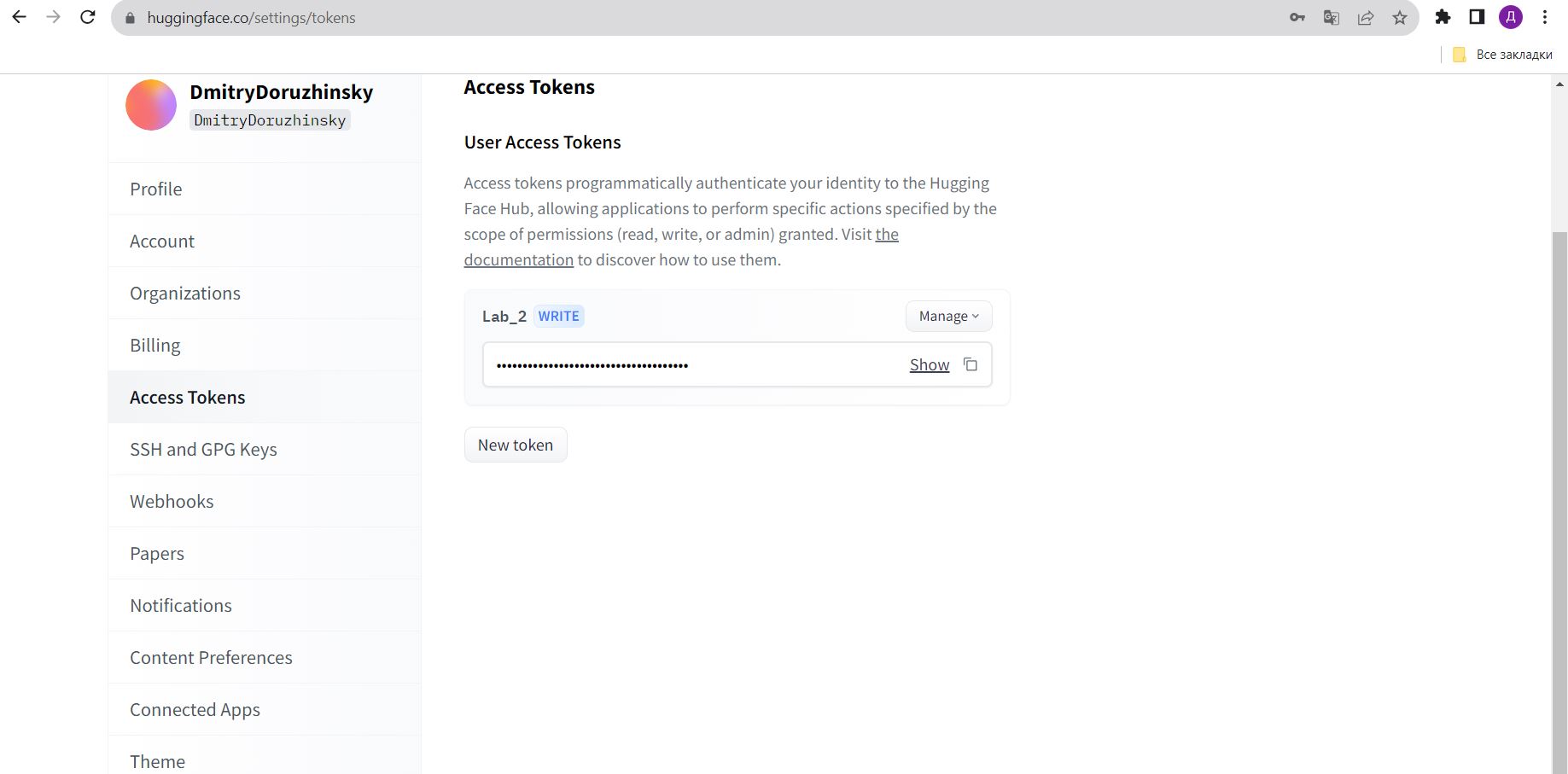


Извлекать аудио дорожку будет при помощи библиотеки ffmpeg и оператора DockerOperator. Хочу отметить, что это очень удобно, пользоваться DockerOperator, необходимо лишь найти нужный контейнер c библиотекой на DockerHub, либо создать его самостоятельно. В данном пункте используется DockerHub https://hub.docker.com/r/jrottenberg/ffmpeg

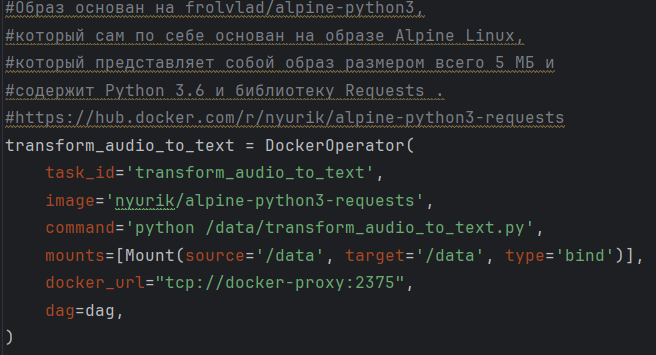


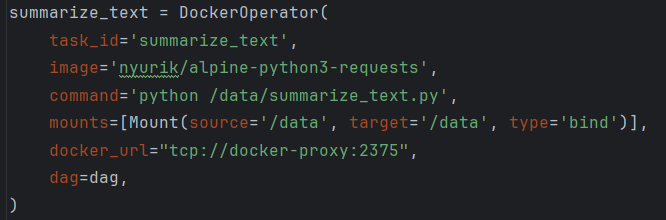
Далее предлагается использовать нейронную сеть <https://huggingface.co/openai/whisper-small> для преобразования аудио в текст. Нужно отметить, что для этого необходимо подготовить следующий файл .py.

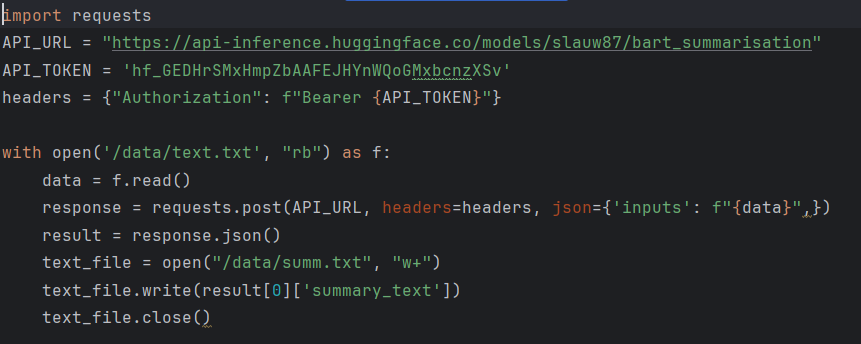


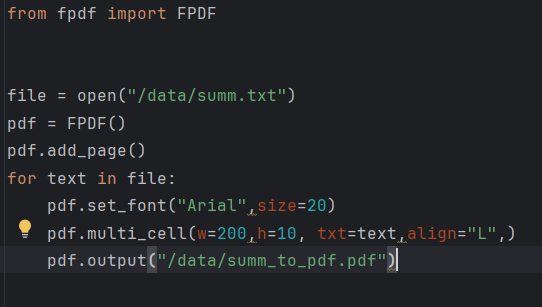
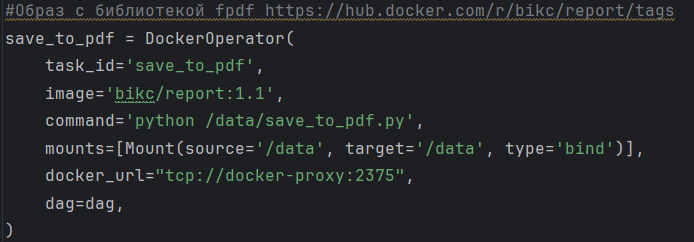
Нужно отметить, что для корректной работы необходимо получить личный токен на сайте www.huggingface.co после полной регистрации. И добавить его в переменную API\_TOKEN

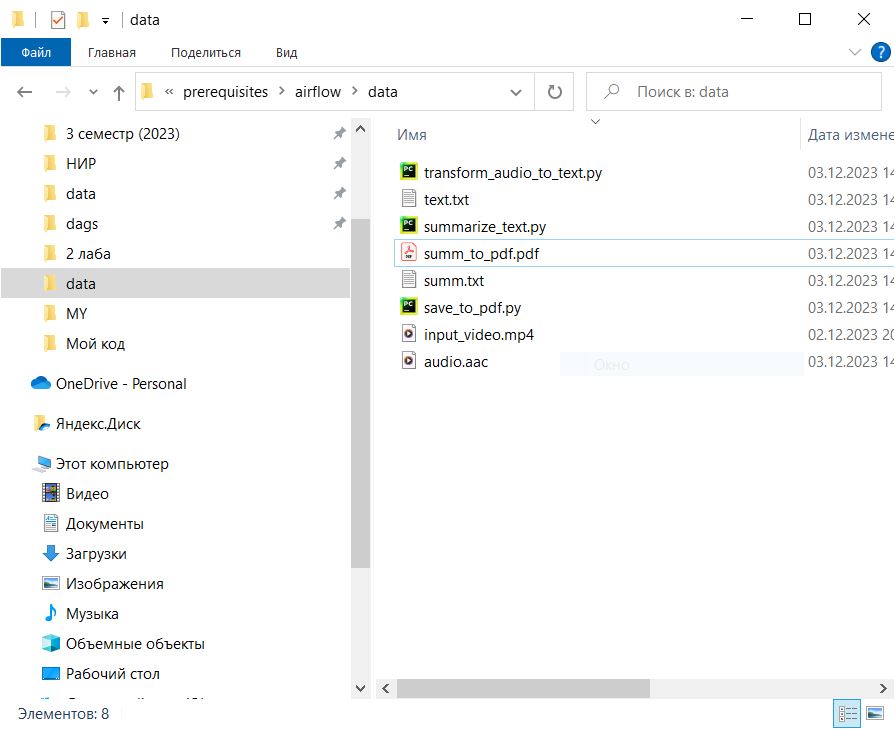
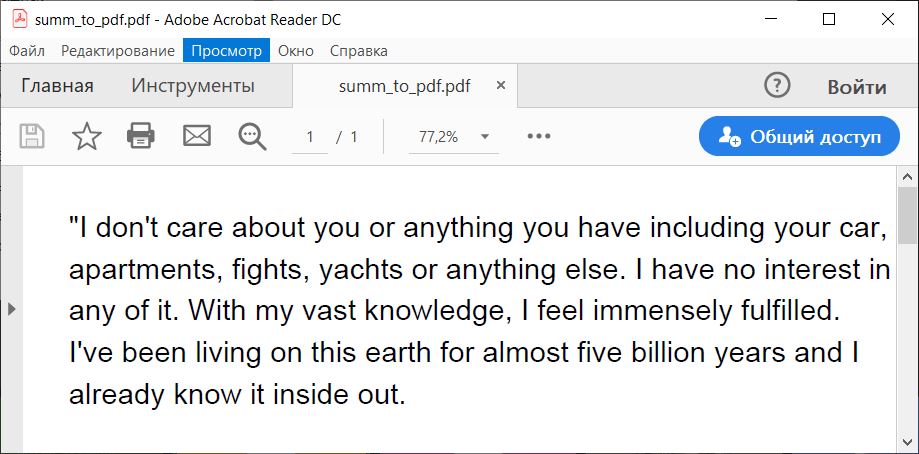
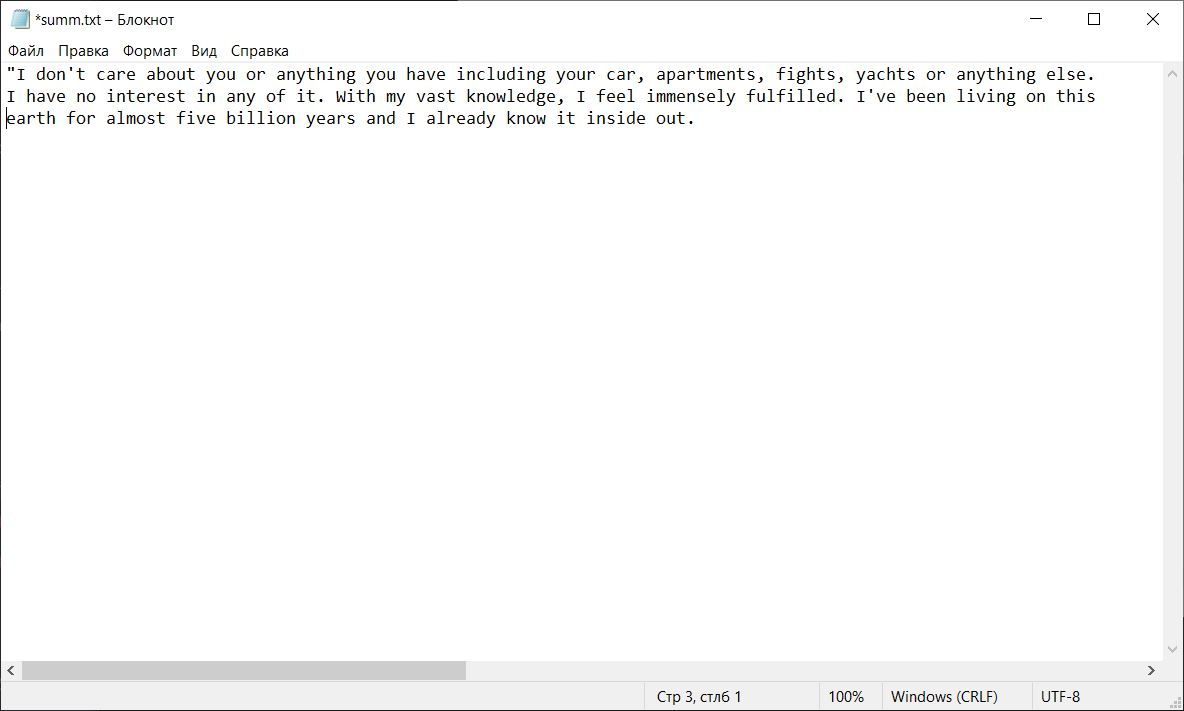
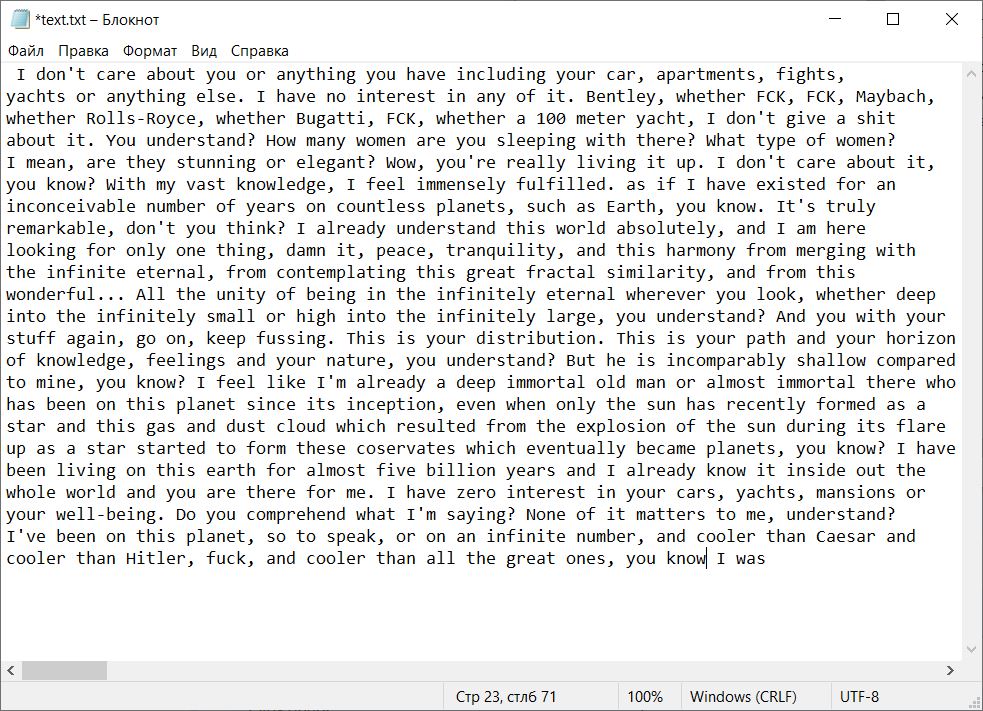
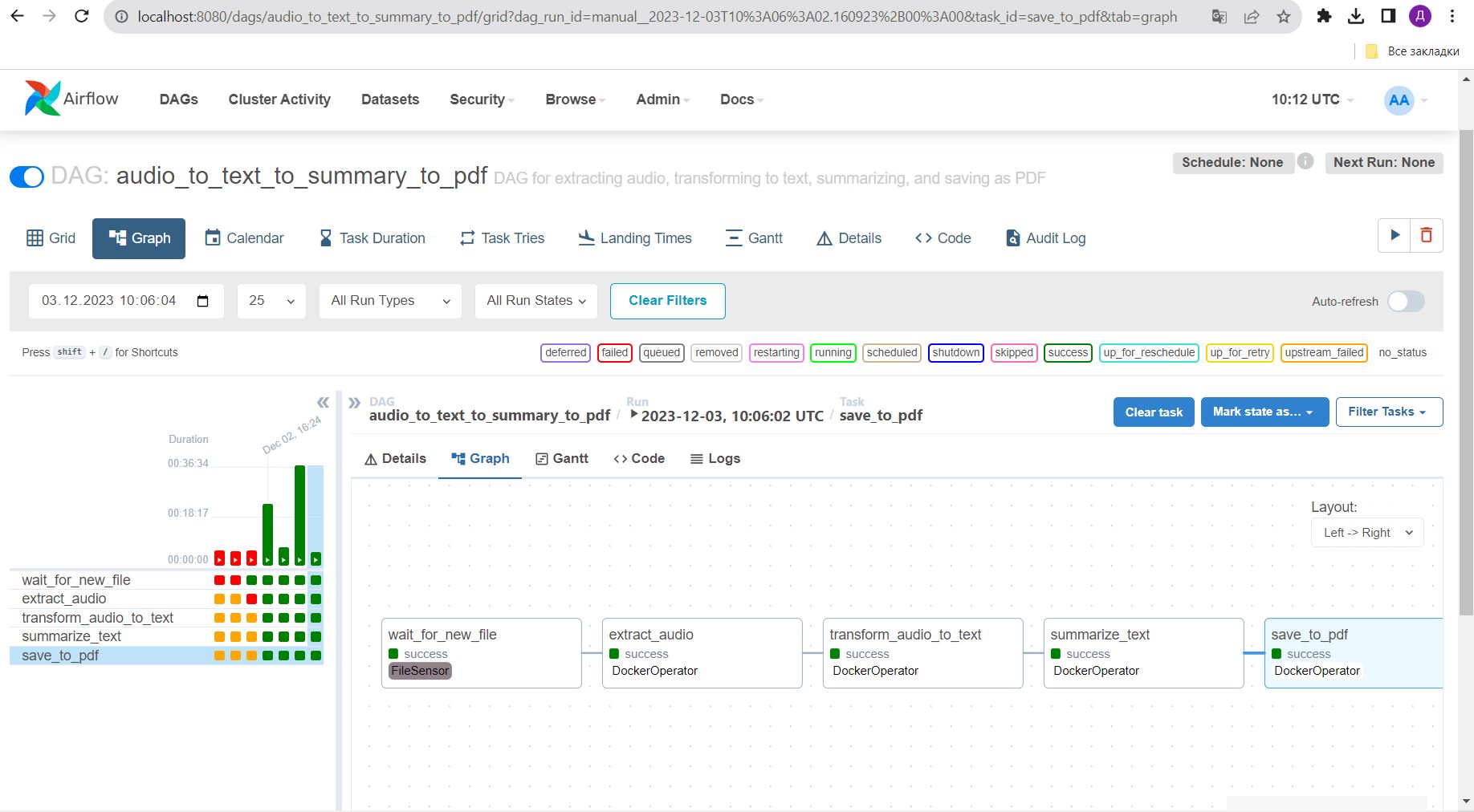
На этом шаге используем библиотеку request, для этого будет подкачивать следующий образ Docker, <https://hub.docker.com/r/nyurik/alpine-python3-requests>.



Далее необходимо при помощи следующей нейросети сделать краткую выжимку из получившегося текста (<https://huggingface.co/slauw87/bart_summarisation>). Этот этап похож на предыдущий. 



После необходимо сохранить получившийся текст сохранить в pdf для этого использовал бибилотеку fpdf, был найден на DockerHub образ для этого и подготовлен файл .py

В итоге путём проб и ошибок получилось всё это заставить работать. В качестве входного видео было использован ролик “Идущий к реке” (https://www.youtube.com/watch?v=VHDwukiQPuE&ab\_channel=%D0%9A%D0%BE%D0%BD%D1%81%D1%82%D0%B0%D0%BD%D1%82%D0%B8%D0%BD%D0%9A%D0%BE%D0%BD%D0%BE%D0%B2%D0%B0%D0%BB%D0%BE%D0%B2) переведенный заранее на английский. Текст из аудио передан довольно точно и сформирован конспект. Результаты 1 части привожу ниже. 

# Пайплайн для обучения модели

Во второй части было необходимо выполнить следующие шаги:

1. Читать набор файлов из определенного источника (файловой системы, сетевого интерфейса и т.д.).
2. Формировать пакет данных для обучения модели.
3. Обучать модель.
4. Сохранять данные результатов обученя (логи, значения функции ошибки) в текстовый файл

В качестве обучаемой модели была выбрана нейронная сеть с стохастическим градиентным спуском. Модель простая, код взят из лично выполненной лабораторной работы по курсу “Нейронные сети и глубокое обучение”. Набор данных MNIST загружается из папки airflow/data. В ходе выполнения столкнулся с несколькими проблемами. Первое долго не мог найти подходящий образ Docker не большого размеры, но в итоге был найден huanjason/scikit-learn:latest, размером 1GB. В лабораторной работе использовался алгоритм OneHotEncoder для подготовки y\_train и y\_test, но в контейнер успешно запустить его не получилось, поэтому пришлось в файле .ipynb сохранить y\_train и y\_test в файл .csv и так же его считать уже в airflow. В итоге всё отработало успешно. Ограничил кол-во эпох и x\_train из-за системных требований. Ниже привожу скриншоты выполнения, а также файла с сохраненными результатами.