## 4 Дополнительная функциональность

### 4.1 Распределение задач

Я и Грачев Дмитрий Александрович получили задачу реализовать получение субтитров не напрямую через YouTube, а через загрузку видео и обработки файла локально. Обязанности не были распределены лидером проекта, так что разбиение на задачи проводили мы сами. Всего было поставлено 3 задачи.

– Реализовать загрузку видео с YouTube,

– Реализовать получение субтитров с загруженного файла,

– Реализовать получение данных о видео: Заголовок, Описание, Канал и т.д.

Мы договорились распределить следующим образом: Дмитрий Александрович выполняет первое задание, а я все остальные.

### 4.2 Возможность загрузки MP4-файлов

Возникла проблема, заключающаяся в том, что, для того чтобы получать субтитры из загруженного файла, нужно иметь загруженный файл. Однако, посоветовавшись, мы пришли к решению, что для реализации этой части не нужно иметь видео именно с YouTube, а можно тестировать на обычном MP4 файле. Все проблемы были решены и стало возможно работать параллельно.

В качестве примера, на котором будет тестироваться модуль, был выбран 15-секундный видеоролик с одной фразой: “А я всё это вместе”.

Для получения субтитров из файла нужно было либо полностью написать свою программу, либо изучить имеющиеся варианты. Было принято решение остановиться на втором пункте. С помощью тимлида был найдет Репозиторий “yt-dlp” — многофункциональный командный загрузчик аудио/видео.

Для нашей задачи yt-dlp – слишком большой модуль, поэтому было приятно решение углубленно его изучить и выделить для себя лишь необходимый функционал.

На GitHub был изучен исходный код проекта. Была найдена необходимая функция.

Возникла проблема – у этой функции было слишком много зависимостей в проекте. Из-за этого было потрачено время на отрезание лишний ветвей и на максимальное сокращение кода.

В результате была написана функция, которая получает субтитры из MP4 файла.

Написанный модуль был перенесен в корень проекта в отдельную директорию.

### 4.3 Получение метаданных из YouTube видео

Для получения метаданных из YouTube видео также был задействован модуль “yt-dlp”. В этом модуле была найдена команда, которая принимает на вход уникальный идентификатор видео, и выводит всю информацию о видео в виде Json. Для запуска функции была использована библиотека subprocess. Для запуска функции были использованы параметры '--dump-json', '--no-playlist', которые отвечают за формат вывода и работу с плейлистом, в котором находится искомый видеоролик. Дальше нужно было обработать полученный Json-материал. Было принято решение исключить работу с тегами видео и оставить только заголовок, название канала, дату загрузки видео, длительность видео, количество просмотров на текущий момент, описание. Так же был добавлена возможность ловить ошибки при получении данных.

Эта функция была добавлена в общий файл работы с ссылками.

### 4.4 Перспективы расширения

Данные модули на момент релиза проекта не были использвоаны, так как возникли проблемы с proxy. Однако они позволяют добавить огромный функционал боту.

Благодаря этим решениям теперь возможна работа не только с YouTube, но и с обычными видео и даже кружочками. Это на самом деле большой функционал, поскольку на данный момент телеграмм кружочки нельзя расшифровывать без платной подписки, а зачастую нет времени или возможностей посмотреть срочный кружочек. Реализации получения субтитров с разных типов видео уже может выполниться в один шаг, поскольку все необходимые модули для этого есть.

Стоит заметить, что получение метаданных видео тоже очень важно для анализа видео, поскольку данные о канале, просмотрах, названию дают больше понимания нейросети о смысле видео, об аудитории, они помогают составить более точную картину. Для улучшения анализа видео сейчас требуется лишь изменить запрос, добавив в него полученные метаданные, тем самым сильно повысив качество ответа.

Метаданные так же могут помочь в дальнейшей реализации RAQ-системы. Поскольку сейчас стало возможность получить информацию о длительности видео, тем самым можно заранее подготовить разбиение по блокам, по участкам видео, тем самым сократив время анализа видео.