**Пояснительная записка**

**Название проекта:** Barrelgram

**Задание проекта:**

* Регистрация пользователя (данные хранятся в бд)
* Подтверждение через электронную почту
* Сам мессенджер (отправка текстовых сообщений)
* Поиск собеседников по имени
* Возможность отправки аудио, видео, фото
* Возможность обращения в поддержку

**Принцип работы:** пользователь заходит на сайт, регистрируется, указывая свой email и затем подтверждая его (письмо в 99 процентов случаев в спаме, но оно и неудивительно). Затем пользователь загружает фотографию своего профиля, после чего переходит на страницу /download, с которой может изменить данные профиля, а также скачать приложение, в котором уже и осуществляются остальные функции мессенджера. Сайт сделан с помощью фреймворка flask, приложение с помощью PyQt5.

Необходимые библиотеки:

* flask
* pyqt5
* sqlalchemy
* Pillow
* requests
* smtplib
* sqlite3

**Программный код:**

1. Сайт (сервер): на сайте пользователю доступна только часть с регистрацией и настройкой профиля. Остальное выполняется с помощью приложения, которому с помощью requests отправляются json ответы. Некоторые страницы, например главная, взаимодействуют и с приложением, и с браузером клиента. В таком случае передаётся булевой параметр app\_client, значение true которого означает, что используется приложение. Основные страницы сайта:

* / - главная страница. Здесь выполняется вход
* /reg - Регистрация нового пользователя
* /email\_accept и /email\_confirm - всё что связано с подтверждением почты
* /settings - настройки личных данных
* /avatar - настройка аватара
* /main - возвращает диалоги в формате json (1 раз при входе в аккаунт)
* /main/dialogues - возвращает то же, что и main, но уже постоянно, для обновления
* /main/dialogue/<id> - возвращает историю диалога (GET) или отправляет сообщение в диалог (POST)
* /main/dialogue/<id>/<type>/<key> - Отправка файла
* /main/search - Поиск пользователей
* /go\_out - Выход

1. Приложение: приложение состоит из нескольких классов окон pyqt и потоков (была задействована многопоточность). Поток RequestThread раз в полсекунды обновляет данные на экране мессенджера (если SEND\_GET\_REQUESTS = True). Подробное описание в комментариях в коде (наверное)