# Аннотация

.

СОДЕРЖАНИЕ

[Аннотация 2](#_Toc80715214)

[ВВЕДЕНИЕ 4](#_Toc80715215)

[1 Основная часть 5](#_Toc80715216)

[1.1 Постановка задачи 5](#_Toc80715217)

[1.2 Выбор технологий 5](#_Toc80715218)

[1.3 Анализ типов и выбор СУБД 7](#_Toc80715219)

[1.4 Предварительная обработка запроса пользователя 8](#_Toc80715220)

[ЗАКЛЮЧЕНИЕ 9](#_Toc80715221)

[Список использованных источников 10](#_Toc80715222)

# ВВЕДЕНИЕ

Поиск информации в интернете в наше время стал одной из самых обыденных вещей в повседневной жизни.

Но поиск зачастую не затрагивает некоторые ниши, которые бывают нужны при запросах в интернет, например, поиск фильмов по фразе, сказанной там. Вы не можете вспомнить название фильма, но помните цитату из него. Если цитата не стала крылатым выражением или просто обсуждаемой фразой, вы не получите ответа на вопрос, что же это был за фильм. Не даст вам поиск ответа и в том случае, если цитата оборвана, или используется где-то кроме этого фильма, имеет множество продолжений и др.

Целью данной проектной работы является разработка системы, позволяющей находить фильм по тексту из него.

Для достижения данной цели потребуется решить такие задачи как:

* создать базу данных фильмов и их текстов;
* разработать алгоритм предобработки запроса пользователя и поиска информации по базе данных;
* создать сайт, демонстрирующий работу системы;
* добавить функции, делающие сайт применимым вне проектной работы.

# Основная часть

## Постановка задачи

Требуется разработать приложение, которое обладает следующей функциональностью.

* Регистрация пользователя.
* Поиск фильмов по фразе.
* Сохранение истории поиска.
* Возможность добавления фильмов в избранное.
* Телеграмм-бот для двухфакторной авторизации.

Система должна иметь веб- и телеграмм- интерфейс для поиска по базе фильмов.

## Выбор технологий

Для выполнения проекта был выбран язык программирования Python. Python — это высокоуровневый язык программирования общего назначения, который используется в том числе и для разработки веб-приложений. Язык ориентирован на повышение производительности разработчика и читаемости кода.

Документация: <https://docs.python.org/3/>

Для разработки веб-приложения используется фреймворк Flask. Flask — фреймворк для создания веб-приложений на языке программирования Python, использующий набор инструментов Werkzeug, а также шаблонизатор Jinja2. Относится к категории так называемых микрофреймворков— минималистичных каркасов веб-приложений, сознательно предоставляющих лишь самые базовые возможности.

Документация: <https://flask.palletsprojects.com/en/2.0.x/>

Для разработки телеграмм приложения используется PyTelegramBotApi (telebot)

PyTelegramBotApi – обертка Telegram Bot Api (<https://core.telegram.org/bots/api>) для Python.

Документация: <https://github.com/eternnoir/pyTelegramBotAPI>

Для приведения текста запросов в начальную форму используется модуль pymorphy2.

Документация: <https://pymorphy2.readthedocs.io/en/stable/>

Для устранения опечаток в словах используется модуль AutoCorrect.

Документация: <https://github.com/filyp/autocorrect>

Для получения случайных чисел используется встроенный в Python модуль Random.

Документация: <https://docs.python.org/3/library/random.html>

Для шифровки паролей используется встроенный в Python модуль hashlib.

Документация: <https://docs.python.org/3/library/hashlib.html>

Для хранения времени сессии используется встроенный в Python модуль datetime.

Документация: <https://docs.python.org/3/library/datetime.html>

Для получения переменных окружения используется встроенный в Python модуль os.

Документация: <https://docs.python.org/3/library/os.html>

Для отправки на сервер Flask запросов из Telegram используется модуль requests.

Документация: <https://docs.python-requests.org/en/master/>

Для работы веб сайта используется язык разметки HTML, таблица стилей CSS, скриптовый язык для страниц Javascript.

Для развертывания приложения на сервере было использовано программное обеспечение Docker.

Документация: <https://docs.docker.com/>

## Анализ типов и выбор СУБД

<https://proglib.io/p/kak-podruzhit-python-i-bazy-dannyh-sql-podrobnoe-rukovodstvo-2020-02-27> (здесь почему sqlite)

<https://habr.com/ru/post/348220/> (здесь про реляционные и нереляционные)

Попробуй сравнить их, <https://github.com/NastyRu/Database_Coursework/blob/main/report/rpz.pdf> здесь можешь посмотреть пример моего сравнения

Это как вывод:

В моей проектной работе используется СУБД SQLite.

SQLite — компактная встраиваемая реляционная база данных. Исходный код библиотеки передан в общественное достояние. Является чисто реляционной базой данных.

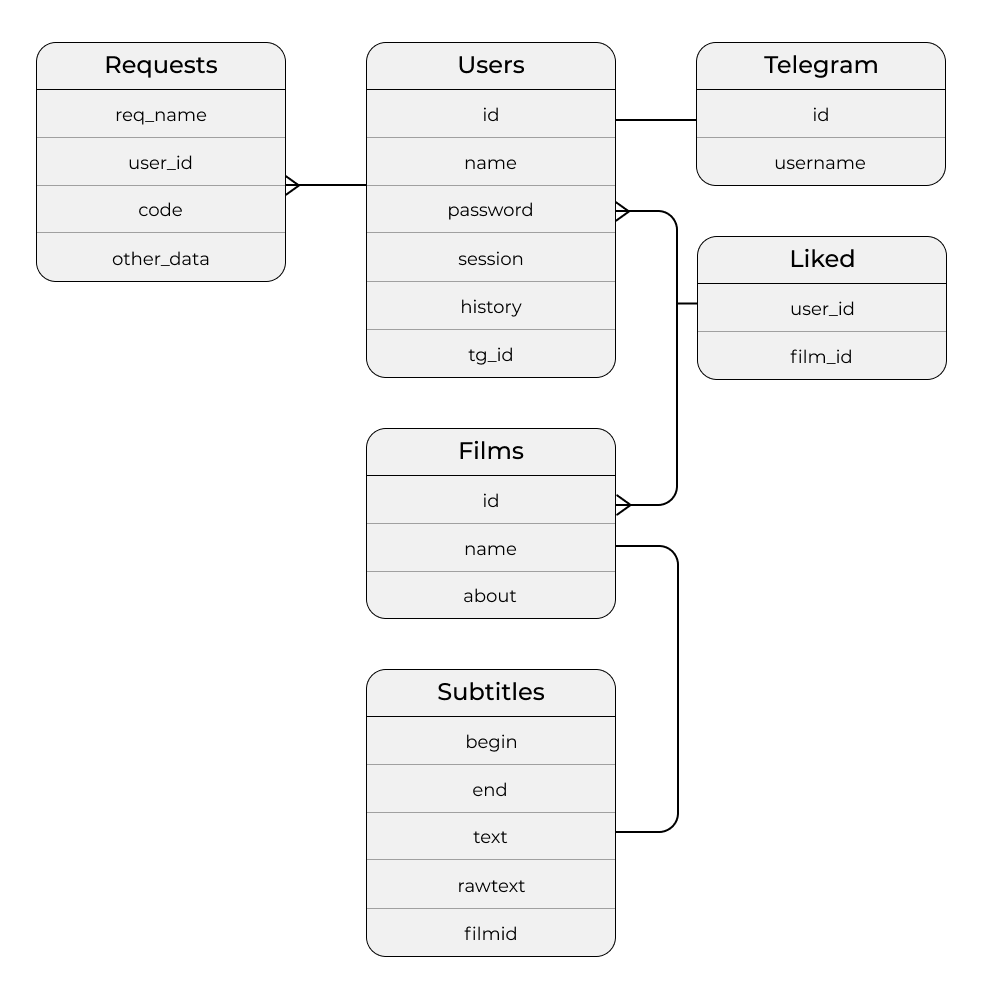
Преимуществом SQLite является скорость исполнения, простота кода, встраиваемость в Python.

Рис. 1

## Предварительная обработка запроса пользователя

На вход подается текст, набранный пользователем, являющийся фрагментом текста одного из фильмов и обрабатывается по методу на рис. 2.

Рис. 2

# ЗАКЛЮЧЕНИЕ

# Список использованных источников

O SQLite: <https://lecturesdb.readthedocs.io/databases/sqlite.html>

О Python: <https://web-creator.ru/articles/python>

O Flask: <https://ru.wikipedia.org/wiki/Flask_(веб-фреймворк)>