**Документация ТЗ (позиция Data Engineer в Farpost)**

Описание сущностей БД:

* Пользователь – пользователь сайта (id, username, пароль, email, дата и время регистрации). Для анонимов (незалогиненных пользователей) все поля пустые, кроме id и даты и времени регистрации.
* Тип действия – тип действия, возможное в логах (id, тип).
* Лог – лог пользователя на сайте (id, id пользователя, id типа действия, дата и время действия, успешен ли). Успешен ли – булевый атрибут (0 – отказ сервера, 1 – принятие сервером).
* Контент – супер-сущность контент сайта (id, тип, дата и время создания, дата и время удаления). Тип – атрибут, хранящий либо “Тема”, либо “Сообщение”.
* Контент Лог – для связи логов и контента (id, id лога, id контента)
* Тема – тема сайта (id, название)
* Сообщение – сообщение на сайте (id, id темы, содержание)

Связи сущностей:

* Тема – Контент (1к1)
* Сообщение – Контент (1к1)
* Тема – Cообщение (1кМ)
* Контент – Контент Лог (1кМ)
* Лог – Контент Лог (1кМ)
* Тип Действия – Лог (1кМ)
* Пользователь – Лог (1кМ)

Описание кода:

**sql/create\_tables.sql**

Файл содержит SQL-код, который удаляет схему public и создаёт её заново (очистка БД), добавляет таблицы, индексы и связи между таблицами.

**sql/agg\_funcs.sql**

Файл содержит SQL-запросы, которые использовались для формирования таблицы с агрегационной информацией по дням.

**.env**

Файл содержит переменные окружения (для подключения к PostgreSQL и pgAdmin4), которые использует Docker при запуске контейнера.

**Dockerfile**

Файл используется для загрузки библиотек, находящихся в requirements.txt, а также запускает python-код.

**docker-compose.yaml**

Файл используется для запуска 3 сервисов: PostgreSQL по порту 5435 (но внутри контейнера используется 5432), pgAdmin4 по порту 5050, python с использованием Dockerfile

**main.py**

Файл создаёт подключение к базе данных, запускает генерацию данных в базу данных с помощью класса DataGenerator, создаёт файл с агрегационной информацией по дням с помощью класса DataAggregation

**scripts/generate\_data.py**

Файл содержит класс DataGenerator, который генерирует данные за месяц (январь 2025 года). Для генерации текстовых данных использовался Faker. Алгоритм генерировал по минимум по 5 действий каждого типа за день, минимум 2 ошибки при создании темы анонимом.

**scripts/agg\_funcs.py**

Файл содержит класс DataAggregation, который формирует файл csv с агрегационной информацией о логах по дням. Колонки таблицы:

1. new\_accounts – кол-во новых аккаунтов, созданных за день (аккаунт – залогиненный пользователь)
2. new\_messages – кол-во новых сообщений, созданных за день
3. anonymous\_messages\_pct – отношение анонимных сообщений ко всем сообщениям за день (в процентах)
4. new\_topics\_pct – отношение кол-ва новых тем за день ко всем уже существующим (в процентах)

**data/actions.json**

Файл содержит названия всех типов действий, которые затем будут добавлены в базу данных.

**data/agg\_data.csv**

Файл содержит агрегационную информацию о логах по дням (формируется после запуска python-кода).