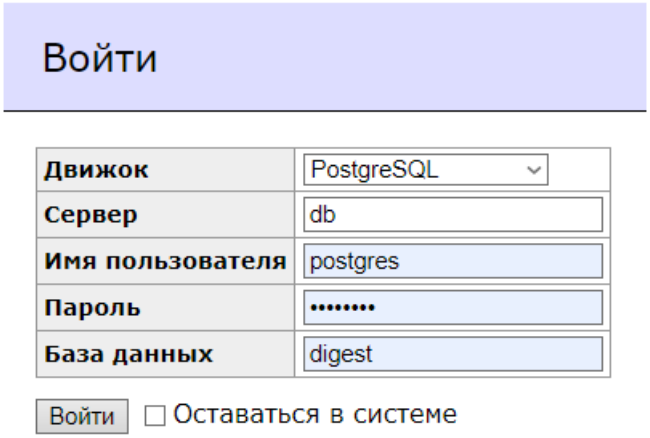
Все взаимодействия с базой данных, происходят путем использования класса {username}DAO(Base), вот пример такого класса ```class UserDAO(BaseDAO):
model = User

```
@classmethod
def create_user(cls, user_id, first_name, username, utm_source):
    with session_maker() as session:
        query = insert(User).values(
            user_id=user_id,
            first_name=first_name,
            username=username,
            chatgpt_flag=False,
            balance=0.0,
            created_at=func.now() + timedelta(hours=3),
            updated_at=func.now() + timedelta(hours=3),
            utm_source=utm_source,
        )
        session.execute(query)
        session.commit()

        return cls.find_one_or_none(user_id=user_id)
```

```
Открывать сессию вне класса СТРОГО запрещено

[[Требования к бэкэнду]]
[[Как запустить проект в первый раз]]
[[Документация/Сервисы/Админка/Информация для бэкэнда
Админки/Взаимодействие с базой данных]]

## Установка проекта
Клонируйте репозиторий и установите его в удобную вам папку. Далее
запросите у коллег файл виртуального окружения(.env) для запуска
проекта
Для запуска установите docker dekstop
## Запуск проекта
Проект можно запустить локально на компьютере таким образом:
#### 1. Создать файл postgres.yaml(заполнить его можно таким образом)
```

name: digest
services:
db:
container_name: db
image: postgres
restart: always
user: postgres
volumes:
- digest-db-data:/var/lib/postgresql/data
ports:
- “5432:5432”
command: -p 5432
environment:
- POSTGRES_DB=digest
- POSTGRES_PASSWORD=postgres
healthcheck:
test: [“CMD”, “pg_isready”]
interval: 1s
timeout: 1s
retries: 50
adminer:
image: adminer
restart: always
ports:
- “8080:8080”
environment:
- ADMINER_DEFAULT_SERVER=db
- ADMINER_DEFAULT_PORT=5432
volumes:
digest-db-data: ``` Этот файл запускает два контейнера: 1) База данных 2) Админка для регулировки этой базы данных(доступна по ссылке localhost:8080) Ниже пример входа



После того как он создан впишите команду в консоли `docker-compose -f postgres.yaml up -d` #### 2. Установить зависимости Пример для windows(команды вписать в консоли, в директории проекта)

```
python -m venv venv
pip install -r requirements.txt
```

3. Запустить проект

Запустить проект можно вписать в консоли(в корневой папке) команду `python .\main.py` #### 4. Подготовка базы данных

```
from sqlalchemy import create_engine

from models import Base


DATABASE_URL =
"postgresql+psycopg2://postgres:postgres@localhost/digest"
engine = create_engine(DATABASE_URL)

Base.metadata.drop_all(engine)
Base.metadata.create_all(engine)

print("Таблицы успешно созданы!")
```

Запустите этот код, далее зайдите в базу данных и заполните таблицу workers, без нее бот не сможет обрабатывать каналы!

Данные требование необходимо соблюдать в будущем!

Код

1. Названия функций и переменных должно отражать их предназначение.
2. Названия всех функций и методов классов, которые должны быть приватными в пакете/классе должны начинаться либо с `_`, либо с `__`
3. Каждая функция, класс должны иметь документацию в определенном формате. Данный формат должен быть обсужден
4. Наименования функций, классов должно иметь единый стиль в пакете.
5. Не стоит зависеть от общих классов. Исключением может являться объект, являющийся репрезентацией сущности с которой мы работаем. Для каждого хендлера должен быть собственный класс запроса и ответа, даже если мы имеем одинаковые ответы в разных хендлерах, классы ответов должны быть разными.
6. Все функции и методы должны быть покрыты UNIT-тестами. Желательно иметь покрытие 60-80%.
7. Для каждого хендлера желательно иметь подробное описание всех возможных ответов с примером ответа.

   ## GIT
8. Все ветки должны иметь в своем названии информацию по типу ветки, краткому описанию и идентификатору задачи в таск менеджеру

- `{prefix}/{name}-{identifier}` Префикс может быть один из следующих:

- `feature`
- `fix`
- `update`

1. Перед выпуском обновления в prod необходимо провести старшему специалисту code review

   # Особенности кода
2. При взаимодействии со временем необходимо переводить UTC время в МСК. По умолчанию в проекте у нас время по МСК
3. При запуске проекта необходимо добавить воркера в базу данных, в ином случае проект запустится, но не сможет обрабатывать каналы