## Модуль db.py

|  |
| --- |
| from sqlalchemy.ext.asyncio import create\_async\_engine, AsyncSession, async\_sessionmaker from sqlalchemy.orm import declarative\_base from faststream import Depends from typing import Annotated import os  DATABASE\_URL = os.getenv("DATABASE\_URL")  engine = create\_async\_engine(DATABASE\_URL) SessionLocal = async\_sessionmaker(bind = engine, class\_= AsyncSession) Base = declarative\_base()  async def get\_session():  async with SessionLocal() as session:  yield session  SessionDep = Annotated[AsyncSession, Depends(get\_session)] |

engine – выступает в качестве центрального источника подключений к конкретной БД

AsyncSession – управляет операциями сохранения для объектов

async\_sessionmaker – фабрика, которая генерирует новые объекты AsynSession

Base = declarative\_base() – создает базовый класс, от которого будут наследоваться все модели (таблицы) БД

async def get\_session():

async with SessionLocal() as session:

yield session

Асинхронная функция: Эта функция создает контекстный менеджер для управления жизненным циклом сессии.

async with SessionLocal() as session: Открывает новую сессию и гарантирует ее закрытие после использования.

yield session: Возвращает созданную сессию вызывающему коду. Это позволяет использовать эту функцию в качестве зависимости в FastAPI или других фреймворках.

SessionDep = Annotated[AsyncSession, Depends(get\_session)]

Здесь создается аннотированный тип SessionDep, который указывает, что это зависимость типа AsyncSession, получаемая через функцию get\_session. Это удобно для использования в обработчиках маршрутов или других функциях, где требуется доступ к базе данных.