Базы данных. Flask. Тестирование.

Небольшой разговор о БД и тестах, а также самые полезные инструменты Flask, сопутствующие пакеты и все, что с этим связано.

Клоков Стас 16.03.2023

Базы данных

Базы данных

Самое общее

- Какие бывают: реляционные и не реляционные(сетевые/иерархические)
- Задача о выборе представления данных. Зачем нам подходы, какие есть проблемы?
- Почему бы nosql?
- А почему бы sql?
- Сложности и распределенность

Проектирование реляционной базы

Что мы делаем, когда думаем про проектирование базы?

- Сущности в базе - вводим все необходимые «типы» объектов, с которыми планируем работать.
- Когда все типы есть, мы можем начать разговоры об их взаимодействии.
- Типы взаимодействий: (1-1), (1-M/M-1), (M-M)
- Таблица для связи M-M и как оно перекладывается на жизнь в orm.

Миграции

Что если надо что-то менять

- Про миграции уже успели много что сказать, но идея лишь в том, что база, как и сервис, данные которого в ней лежат, меняется, а изменениями нужно уметь управлять.
- Миграции схем базы - миграции, меняющие вид базы.
- Миграции данных - миграции, меняющие сами данные.
- Миграции - часто тяжелая задача и под их проведение поднимают репликации и/или отдельные тачки.

Транзакции

Что такое, зачем?

- Транзакция - операция, собирающая исполнение нескольких инструкций и имеющая смысл только при полном ее завершении.

Сессии

Как ими управлять и зачем они?

- Сессии нужны, чтобы уменьшать нагрузку.
- Управлять ими самим нам не требуется.

Основные DDL и DML операторы

Что и как мы делаем с базой?

- DDL - создание и редакция самой базы.
- DML - пресловутые SELECT, INSERT, DELETE, UPDATE для работы с данными

ACID

Просто чуть еще полезной теории по принципам.

- Atomicity - атомарность (транзакция либо вся, либо вообще не выполнится)
- Consistency - согласованность (из предыдущего, состояния базы всегда согласованные)
- Isolation - изолированность (изоляция и ее слои в базах - здесь речь про то, что транзакции изолированы друг от друга и при параллельном исполнении не ломают работу системы)
- Durability - надежность (завершенность транзакции - уверенность в результате)

СУБД

Системы управления базами данных

- Тут все очень просто - системы управления. Чтобы выбрать нужную, надо понимать задачу, тип базы и оценивать поддержку в дальнейшем.
- Самые распространённые MySQL, PostgreSQL
- SQLlite - если чисто поиграться и оно будет в дз

Flask и его инструменты

Flask app. Сервер на фласк.

Что он может и что дает?

- Он управляет ответами по протоколу
- Занимается роутингом
- Содержит полезные настройки по типу логирования, настроек CORS, и тд

Flask-шаблоны. Jinja2

Зачем нам шаблонизатор и что он позволяет делать?

- Шаблоны нужны для того, чтобы возвращать готовые страницы
- Шаблоны умеют исполнять всякую тупую логику
- Шаблоны умеют наследоваться

Flask route methods. Обработка форм.

Как мы можем разруливать формочки и типы запросов?

- Формы в HTML - самый просто вариант отправки запросов разного типа из веб-интерфейса и наша апишка в роутинге должна уметь разруливать типы.
- Методы описываются в роутинге на каждую ручку(-и)

Flask-sqlalchemy. ORM.

Что по орм во фласке, как живем, что важного?

- ORM - алхимия, очень приятная и гибкая.
- Позволяет довольно удобно работать с классами под таблицы
- M-M через таблицу посредника
-

Flask-wtf. Формы по-взрослому.

Зачем нам формы и как мы с этим работаем?

- Работаем с формами из кода и подставляем вызовы функций в шаблоны
- Валидируем данные и настраиваем вариацию из коробки
- Настраиваем типы, а шаблон поймет все сам

Flask login. Работа с аутентификацией.

Что нам предоставляет фласк?

- Позволяет делать процессы аутентификации и авторизации из коробки
- Умеет чекать токены из сессионных кук для авторизации
- Умеет сам хэшировать пароли

Flask cookie

Работа с куками во фласке.

- Очень полезный тул для работы с куками напрямую из кода
- Простейший интерфейс для работы с куками как со словарем

Flask микросервисы. Blueprint.

Декомпозиция фласка на аппы.

- Blueprint позволяет разделить общую аппу на составляющие
- Нужно для разделения ответственности и в целом для декомпозиции MVC

Flask миграции. Alembic.

Механики для миграций для фласка.

- Удобный тулик для работы с миграциями, но к сожалению не всемогущий - есть некоторый ряд проблем.

Тестирование

Тестирование.

О двойной работе и важности тестов.

- Пишем тесты - делаем одно и то же дважды.
- Помогаем себе не умирать в будущем.
- Либо: utittest, pytest

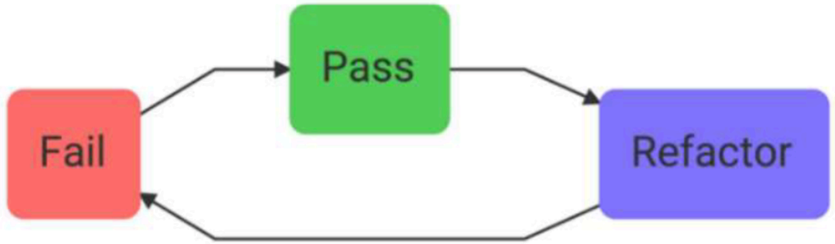
TDD

Базовые принципы Test Driven Development.

Вот основные принципы применения TDD:

1. Прежде чем писать код реализации некоей возможности, пишут тест, который позволяет проверить, работает ли этот будущий код реализации, или нет. Прежде чем переходить к следующему шагу, тест запускают и убеждаются в том, что он выдаёт ошибку. Благодаря этому можно быть уверенным в том, что тест не выдаёт ложноположительные результаты, это — своего рода тестирование самих тестов.
2. Создают реализацию возможности и добиваются того, чтобы она успешно прошла тестирование.
3. Выполняют, если это нужно, рефакторинг кода. Рефакторинг, при наличии теста, который способен указать разработчику на правильность или неправильность работы системы, вселяет в разработчика уверенность в его действиях.

TDD расшифровывается как Test Driven Development (разработка через тестирование). Процесс, реализуемый в ходе применения этой методологии очень прост:



Формат AAA

Arrange Act Assert.

- Пресет сначала
- Затем действие, которое тестим
- Проверяем, что все ок

Уровни тестирования.

Немного об уровнях конкретно в бэкенде на питоне.

- Уровень юнитов
- Уровень АПИ

Фикстуры и моки

Что к чему с инфрой тестов

- Фикстуры - инфра внешних сущностей
- Моки - данные для тестирования

Фабрики тестовых объектов

О том как в питоне генерить объекты на все случаи жизни.

- Фабрика из пакета - factory-boy