Introduction

Part 1, Chapter 1

Changelog »

App Overview

By the end of this course, you will have built an asynchronous text summarization API with Test-Driven Development. The API itself will follow RESTful design principles, using the basic HTTP verbs: GET, POST, PUT, and DELETE.

Endpoint	HTTP Method	CRUD Method	Result
/summaries	GET	READ	get all summaries
/summaries/:id	GET	READ	get a single summary
/summaries	POST	CREATE	add a summary
/summaries:id	PUT	UPDATE	update a summary
/summaries/:id	DELETE	DELETE	delete a summary

Along with Python and FastAPI, we'll use Docker to quickly set up our local development environment and simplify deployment. Tortoise ORM, an async ORM (Object Relational Mapper), will be used to interact with a Postgres database. We'll use pytest instead of unittest for writing unit and integration tests to test the API. Finally, we'll store the code on a GitHub and utilize GitHub Actions to run tests before deploying to Heroku.

Before diving in, let's take a minute to go over why some of the above tools are being used.

FastAPI

While Flask and Django are the two most popular Python web frameworks, FastAPI has surged in popularity in late 2019 and early 2020 since it supports async out-of-the box and has an advanced data validation feature based on Python type hints.

Put simply, FastAPI is a modern, batteries-included Python web framework that's perfect for building RESTful APIs. It can handle both synchronous and asynchronous requests and has built-in support for data validation, JSON serialization, authentication and authorization, and OpenAPI (version 3.0.2 as of writing) documentation.

Highlights:

1. Heavily inspired by Flask, it has a lightweight microframework feel with support for Flask-like route decorators.
2. It takes advantage of Python type hints for parameter declaration which enables data validation (via Pydantic) and OpenAPI/Swagger documentation.
3. Built on top of Starlette, it supports the development of asynchronous APIs.
4. It's fast. Since async is much more efficient than the traditional synchronous threading model, it can compete with Node and Go with regards to performance.
5. Because it's based on and fully compatible with OpenAPI and JSON Schema, it supports a number of powerful tools, like Swagger UI.
6. It has amazing documentation.

Feedback

Review the Features guide from the official docs for more info. It's also encouraged to review Alternatives, Inspiration, and Comparisons, which details how FastAPI compares to other web frameworks and technologies, for context.

Docker

Docker is a container platform used to streamline application development and deployment workflows across various environments. It's used to create the infrastructure required -- like installing Linux, configuring system-level dependencies, and running Python -- for the web app within a lightweight container that can be moved from your development machine to the production server quickly and easily.

Pytest

pytest is a test framework for Python that makes it easy (and fun!) to write, organize, and run tests. When compared to unittest, from the Python standard library, pytest:

1. Requires less boilerplate code so your test suites will be more readable.
2. Supports the plain `assert` statement, which is far more readable and easier to remember compared to the `assertSomething` methods -- like `assertEquals`, `assertTrue`, and `assertContains` -- in unittest.
3. Is updated more frequently since it's not part of the Python standard library.
4. Simplifies setting up and tearing down test state with its fixture system.
5. Uses a functional approach.

Tortoise ORM

Tortoise ORM is a async ORM inspired by the Django ORM that's designed for ease of use. It's familiar constructs help make it easier for Python developers to switch over to the world of async.

Besides Tortoise, there are a number of other ORMs and query builders in active development that support async operations. Just keep in mind that the ecosystem is not mature yet, so things are moving quickly. Breaking changes are to be expected.

ORMs:

- FastAPI SQLAlchemy
- GINO
- ORM

Query Builders:

- asyncpgsa
- Databases

GitHub

GitHub Actions is a continuous integration and delivery (CI/CD) solution, fully integrated with GitHub. Along with it, we'll also use GitHub Packages, a package management service, to store Docker images.

There are a number of platforms that include remote version control along with both CI and package management:

1. GitLab
2. AWS (CodeCommit, CodeBuild, CodeDeploy, ECR)
3. GCP (Cloud Source Repositories, Cloud Build, Container Registry)
4. Azure (Repos, Pipelines, Container Registry)

Each of them have similar features and pricing models so you really can't go wrong with any of them. If you're used to using a different platform, feel free to continue to use it if that's your preference. Or, check your understanding, and try a new platform.

Heroku

Heroku is a cloud Platform as a Service (PaaS) that provides hosting for web applications. They offer abstracted environments where you don't have to manage the underlying infrastructure, making it easy to manage, deploy, and scale web applications. With just a few clicks you can have your app up and running, ready to receive traffic.

Feedback

Changelog »

✓ Mark as Completed

Feedback