Server Development with Flask

Our Learning Goals

- Use Flask to create an API server
- Describe the role of API servers in web development

Discussion:

What Are APIs Made Of?

Let's recap, from a user perspective, what an API needs to include for a program to be able to successfully interact with it.

The Web Development Eco-System

When programming web applications, we typically classify technologies as either **front-end** or **back-end**.

Front-End technologies include HTML, CSS, and JavaScript, and manage everything a regular user would interact with in their web browser.

Back-End technologies include servers, databases, and other application logic that power a web application.

Web Development with Flask

API development is crucial to making applications that users can interact with. Most web applications, even those with a user-friendly front-end website, rely on making API calls to interact with a server, where the application logic is based.

Flask is a popular, minimal Python library for creating servers. It can:

- Create and write the entire back-end in Python!
- Do small tasks (e.g., create a microblog or stand up a simple API).
- Manage complex applications (e.g., Pinterest's API or create a Twitter clone).

Basic Flask Server

```
from flask import Flask # import the Flask class to create an app

app = Flask(__name__) # invoke the Flask clas

@app.route('/') # define the first route, the home route
def index(): # define the function that responds to the above route
  return 'Hello, World!'

if __name__ == '__main__':
    app.run() # Start the server listening for requests
```


Guided Walk-Through:

1. Running a Flask Application

15 minutes

Let's run through starting up a Flask server together.

Execute the first block of example code, then make a request in your browser to localhost:5000 to see the response from your server.

While 5000 is the default port for Flask applications, you can set a different port in the app.run() method.

Discussion:

HTTP Requests and Methods

Our Flask application's goal is to respond to HTTP requests from users.

Before we go further, what are the properties of an HTTP request?

Creating a Route in Flask

```
@app.route("/posts", method="GET")
def index():
        return { "response_code": 200, "data": "Hello from index" }
```

The `url` parameter sets up the url path a user must provide to get a response from this route.

All routes start with "/", so that even "www.website.com" would be listed as:

```
@app.route("/")
```

Creating a Route in Flask

```
@app.route("/posts/<post_id>", method="GET")
def index(post_id):
        return { "response_code": 200, "data": f"Hello from {post_id}"
}
```

The **url path parameters** are used to create dynamic routes that can transfer the path's value into an argument within the handler function, in this case called **post_id**.

Creating a Route in Flask

```
@app.route("/posts", method="GET")
def index():
        return { "response_code": 200, "data": "Hello from index" }
```

The **method** keyword argument limits a route to requests with the specified method.

Creating a Route in Flask

```
@app.route("/posts", method="GET")
def index():
        return { "response_code": 200, "data": "Hello from index" }
```

The **route handler function**, defined beneath the route decorator, will be invoked whenever your application receives a request matching the route.

Creating a Route in Flask

```
@app.route("/posts", method="GET")
def index():
        return { "response_code": 200, "data": "Hello from index" }
```

In this example, we're returning a dictionary. Flask will automatically translate this into a JSON object.

The Seven RESTful Routes

URL Path	Method	Name	Purpose
"/cats"	GET	index	Show the home page for this collection
"/cats"	POST	create	Add a new item to a collection
"/cats/<id>"	GET	show	Show details of a specific item
"/cats/<id>"	PUT	update	Update details of a specific item
"/cats/<id>"	DELETE	delete	Delete an item
"/cats/new"	GET	new	Show the form to create a new item
"/cats/<id>/edit"	GET	edit	Show the form to update a specific item

Group Exercise:
2. API Routes

30 minutes

Create the seven routes listed in table for a restaurant reviews application. Responses should be strings that describe what the route does.

"/reviews"	GET	index	Show the home page for this collection
"/reviews"	POST	create	Add a new item to a collection
"/reviews/<id>"	GET	show	Show details of a specific item
"/reviews/<id>"	PUT	update	Update details of a specific item
"/reviews/<id>"	DELETE	delete	Delete an item
"/reviews/new"	GET	new	Show the form to create a new item
"/reviews/<id>/edit"	GET	edit	Show the form to update a specific item

Server Development with Flask

Wrapping Up

Recap

In today's class, we…

- Used lists and list methods to manage collections of data.
- Used index-based retrieval to access and manipulate list items.
- Used dictionaries to represent data with multiple properties.

Looking Ahead

On your own:

- Ensure that you've completed the Python pre-work and pre-work quiz.

Next Class:

Conditionals

Don’t Forget: Exit Tickets!


GA