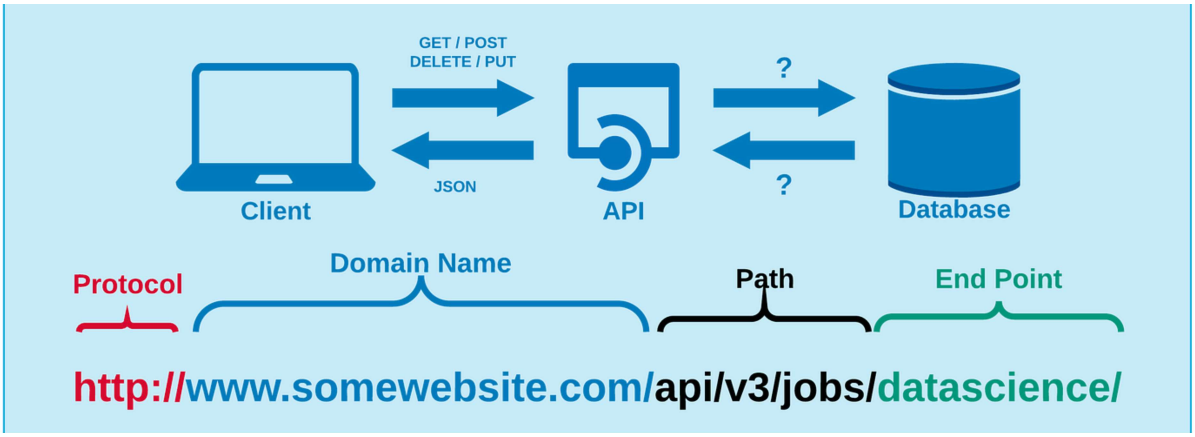
Web APIs - Flask

Web APIs are tools for making information and application functionality accessible over the internet.
In programming more generally, the term API, short for Application Programming Interface, refers to a part of a computer program designed to be used or manipulated by another program, as opposed to an interface designed to be used or manipulated by a human. Computer programs frequently need to communicate amongst themselves or with the underlying operating system, and APIs are one way they do it. In this tutorial, however, we'll be using the term API to refer specifically to web APIs.



API Terminology

When using or building APIs, you will encounter these terms frequently:

- **HTTP (Hypertext Transfer Protocol)** is the primary means of communicating data on the web. HTTP implements a number of "methods," which tell which direction data is moving and what should happen to it. The two most common are GET, which pulls data from a server, and POST, which pushes new data to a server.
- **URL (Uniform Resource Locator)** - An address for a resource on the web, such as https://programminghistorian.org/about. A URL consists of a protocol (http://), domain (programminghistorian.org), and optional path (/about). A URL describes the location of a specific resource, such as a web page. When reading about APIs, you may see the terms URL, request, URI, or endpoint used to describe adjacent ideas. This tutorial will prefer the terms URL and request to avoid complication. You can follow a URL or make a GET request in your browser, so you won't need any special software to make requests in this tutorial.
- **JSON (JavaScript Object Notation)** is a text-based data storage format that is designed to be easy to read for both humans and machines. JSON is generally the most common format for returning data through an API, XML being the second most common.

- **REST (REpresentational State Transfer)** is a philosophy that describes some best practices for implementing APIs. APIs designed with some or all of these principles in mind are called REST APIs. While the API outlined in this lesson uses some REST principles, there is a great deal of disagreement around this term. For this reason, I do not describe the example APIs here as REST APIs, but instead as web or HTTP APIs.

Flask

is a web framework for Python, meaning that it provides functionality for building web applications, including managing HTTP requests and rendering templates.
To install flask, use the command: **`pip install flask`**

```
from flask import Flask
import pickle

app = Flask(__name__)
```

- We've imported `Flask` class, to which we'll pass the name of the app.We can now develop API endpoints as per our need.
- For a particular request we need to send it via some `url`, and corresponding to that `url`, we need some piece of code, that will respond when invoked.
- For this, we've written a function `ping()`, in which we will write a simple message written
- To define the `url`, we'll use a **decorator** `@app.route` to which you'll pass the url. You can also specify the type of url request such as **get**, **post**, etc.
- A decorator in python allows a user to add new functionality to an existing object without modifying its structure.

```
from flask import Flask
import pickle

app = Flask(__name__)

@app.route("/ping", methods=['GET'])
def ping():
    return {"message": "Hi there, I'm working!!"}
```

Flask cheatsheet:
https://drive.google.com/file/d/1705hDOsBCShGwEGSOetvfejYeUFtWFzl/view?usp=sharing
Flask docs: https://flask.palletsprojects.com/en/2.3.x/