Q&A Session

Presented by: @TA_Yaya

Flask : Introduction | Rendering and Forms

Defintion of Flask

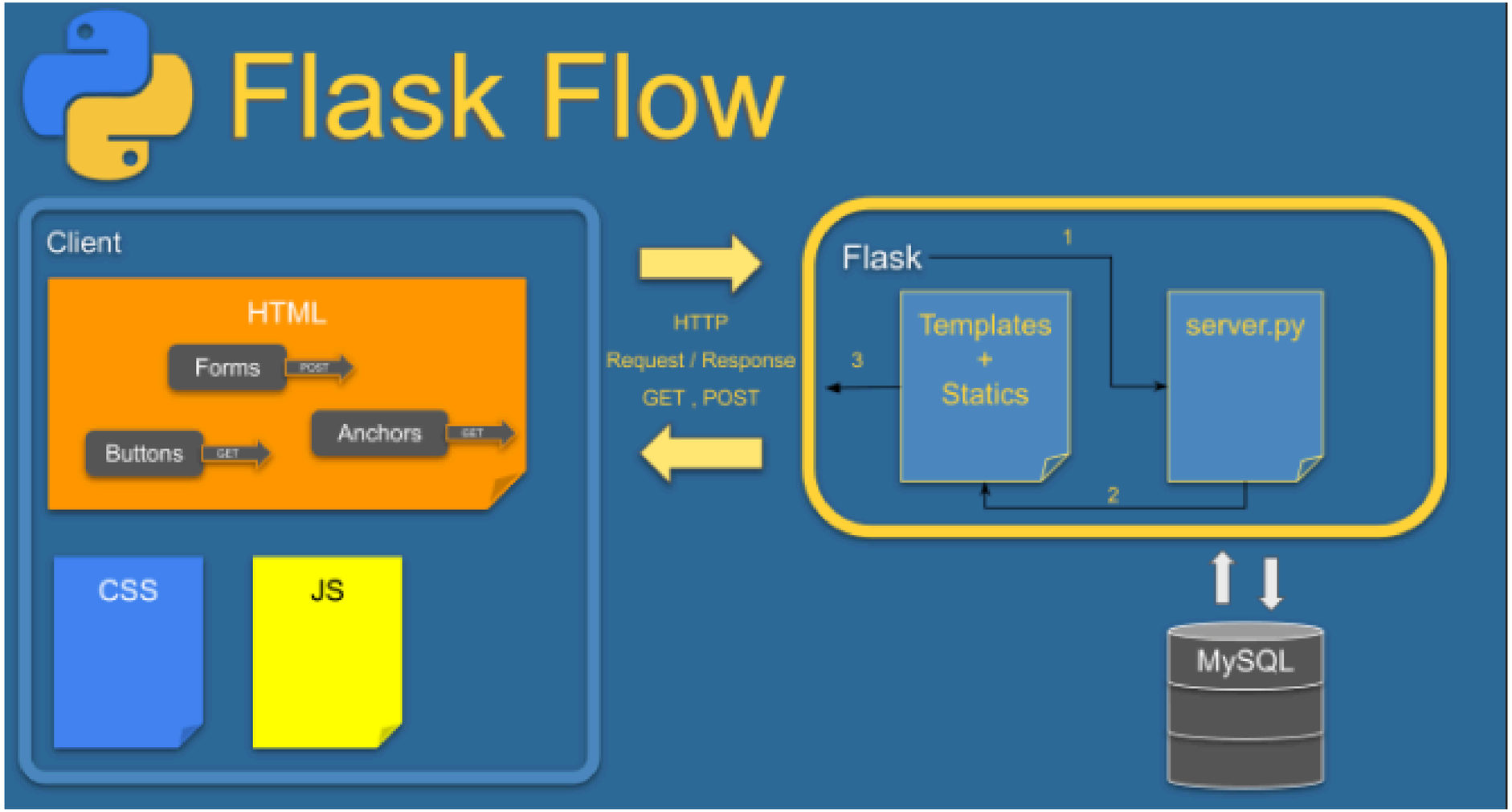
Flask is a lightweight and flexible web framework for Python.

It allows developers to build web applications quickly and easily by providing essential tools and features, such as routing, templates, and request handling.

Flask is designed to be simple, enabling developers to create scalable and maintainable applications without unnecessary complexity.

It also supports extensions, which add functionality to your applications as needed.

Flask Flow Diagram



Explanation of the Flask Flow Diagram

1) Client :

- **HTML**: This is the markup language used to structure web pages. It contains elements like forms, buttons, and anchors (links).
- **CSS**: Cascading Style Sheets are used to define the visual appearance of HTML elements, styling the web page.
- **JS (JavaScript)**: A scripting language used to add interactivity to web pages, such as animations, form validation, etc.

2) Flask (Backend) :

- **server.py**: This is the main script of your Flask application.
  It contains the server logic and handles HTTP requests (GET, POST) sent by the client.

- **Templates + Statics**:
  - **a)Templates**: These are HTML files with placeholders that will be filled in by Flask. Flask uses Jinja2 as the templating engine.

  - **b) Statics**: These are static files like images, CSS, and JavaScript files.

Explanation of the Flask Flow Diagram

3) Interaction Between Client and Server:

- **HTTP Requests (GET, POST)**:

  a)GET: Used to request data from the server, such as clicking links or submitting forms without sensitive data.

  b)POST: Used to send data to the server, such as when submitting forms containing sensitive information.

- **HTTP Response**: The Flask server responds with the requested data, often in the form of HTML pages rendered by the templates.

4) Database (MySQL) :

- **server.py**: This is the main script of your Flask application. It contains the server logic and handles HTTP requests (GET, POST) sent by the client.

- **Templates + Statics**:

  a)Templates: These are HTML files with placeholders that will be filled in by Flask. Flask uses Jinja2 as the templating engine.

  b) Statics: These are static files like images, CSS, and JavaScript files.

Typical Workflow

1. **The client sends an HTTP request (e.g., by clicking a link or submitting a form).**
2. **The Flask server receives the request and processes it in server.py.**
3. **The server may interact with the MySQL database to fetch or update data.**
4. **The Flask server renders an HTML template, integrating the necessary data, and sends it back to the client.**
5. **The client receives the response and displays the updated web page in the browser.**

Defintion of Virtual Environments

A virtual environment is a tool that creates an isolated space to install project-specific dependencies without affecting other projects or system-wide libraries.

A virtual environment is an essential tool for professional Flask application development, as it ensures the isolation, management, and security of project-specific dependencies.

This helps maintain a clean and reproducible development environment.

Flask app Structure

<Let’s Code />