Open-Source Report

Proof of knowing your stuff in CSE312

Guidelines

Provided below is a template you must use to write your reports for your project.

Here are some things to note when working on your report, specifically about the **General Information & Licensing** section for each technology.

- **Code Repository**: Please link the code and not the documentation. If you'd like to refer to the documentation in the **Magic** section, you're more than welcome to, but we need to see the code you're referring to as well.
- **License Type**: Three letter acronym is fine.
- **License Description**: No need for the entire license here, just what separates it from the rest.
- **License Restrictions**: What can you *not* do as a result of using this technology in your project? Some licenses prevent you from using the project for commercial use, for example.

Also, feel free to extend the cell of any section if you feel you need more room.

If there's anything we can clarify, please don't hesitate to reach out! You can reach us using the methods outlined on the course website or see us during our office hours.

[Flask]

General Information & Licensing

Code Repository	https://github.com/pallets/flask
License Type	BSD-3-Clause license
License Description	It is an open-source software license used by the Flask project and many other open-source projects. The license allows for free use, modification, and distribution of the software, as long as certain conditions are met.
License Restrictions	1. Redistributions of the source code must retain the copyright notice, the list of conditions, and the disclaimer. 2. Redistributions in binary form must reproduce the copyright notice, the list of conditions, and the disclaimer in the documentation and/or other materials provided with the distribution. 3. The name of the copyright holder and the names of its contributors cannot be used to endorse or promote products derived from this software without specific prior written permission.

Magic

Brief description:

app.run() is a method in the Flask framework that starts the development server. When we call **app.run()**, we are starting the Flask development server that listens for incoming HTTP requests on the specified IP address and port number. The **app.run()** method takes several optional parameters, such as **host**, **port**, **debug**, and others, to configure the behavior of the server.

The whole chain from framework code to HW code:

1. We call app.run() at: https://github.com/fhuang566/CSE-312/blob/18667025ec6a622b1183c1ae52ea05297bc5569b/code/flask_server.py - L23
    - Flask's **run** method is responsible for starting the Werkzeug development server.

2. Flask's **app.run()** method calls Werkzeug's **run_simple():** https://github.com/pallets/werkzeug/blob/1bfd5deb6b2eb5bf369c089796a6669cf89f7bd7/src/werkzeug/serving.py - L917
    - The **run_simple()** function in Werkzeug creates and starts a server instance based on the given configuration.

3. Werkzeug's **run_simple()** function calls **srv.serve_forever():**https://github.com/pallets/werkzeug/blob/1bfd5deb6b2eb5bf369c089796a6669cf89f7bd7/src/werkzeug/serving.py - L1075
    - The **serve_forever()** method is responsible for running the server in a loop, accepting incoming connections and handling requests.

4. Werkzeug uses Python's built-in **socketserver** module to create the underlying server. The **BaseServer** class in Werkzeug is a subclass of **socketserver.TCPServer:** https://github.com/python/cpython/blob/4f5e1cb00a914692895c1c16e446c8d2ab3efb7e/Lib/socketserver.py - L216
    - Werkzeug's **BaseServer** inherits from **socketserver.TCPServer** and customizes it for serving WSGI applications.

5. **socketserver.TCPServer.serve_forever()** is called when Werkzeug's **serve_forever()** is invoked:https://github.com/python/cpython/blob/4075e0166fcae0eef5e3abe1a97b3c227ce6861c/Lib/socketserver.py - L215
    - The **serve_forever()** method in **socketserver.TCPServer** runs the server in a loop, accepting connections and processing requests.