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.

FastAPI

General Information & Licensing

Code Repository	https://github.com/tiangolo/fastapi
License Type	MIT License
License Description	• Permissive license • Allows unrestricted use without any limitation free of charge • Protects the copyright holders from liability for any consequences caused by anything utilizing the licensed code
License Restrictions	• Derivative code must also be licensed under the MIT License

Magic

Dispel the magic of this technology. Replace this text with some that answers the following questions for the above tech:

- How does this technology do what it does? Please explain this in detail, starting from after the TCP socket is created
- Where is the specific code that does what you use the tech for? You ***must*** provide a link to the specific file in the repository for your tech with a line number or number range.
    - If there is more than one step in the chain of calls *(hint: there will be)*, you must provide links for the entire chain of calls from your code, to the library code that actually accomplishes the task for you.
    - Example: If you use an object of type HttpRequest in your code which contains the headers of the request, you must show exactly how that object parsed the original headers from the TCP socket. This will often involve tracing through multiple libraries and you must show the entire trace through all these libraries with links to all the involved code.

*This section will likely grow beyond the page

//Explanation here

Our project creates a WebSocket object in main.py on line 106.

We then call the WebSocket.accept() in main.py on line 114 to establish a websocket connection.

accept() calls WebSocket.recieve() to establish the websocket connection if it isn't already established in websockets.py on line 98

WebSocket.recieve()

(Deleted but keeping just in case)
websocket() calls add_api_websocket_route()

add_api_websocket_route() calls APIRouter's add_api_websocket_route()

add_api_websocket_route() creates a new APIWebSocketRoute object

APIWebSocketRoute calls websocket_session() in routing.py on line 74 of the constructor
https://github.com/encode/starlette/blob/master/starlette/routing.py#L74

websocket_session() creates a new WebSocket object

Look into WebSocketState class and HTTPConnection class when back from class