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 License
License Description	● A BSD license are a family of free software licenses ● A BSD license imposes minimal restrictions on the use and distribution of the software. ● These do not have share-alike requirements like other licenses.
License Restrictions	● Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. ● Redistributions in binary form must reproduce the above

	copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. • Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

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

This library is a lightweight web application framework. This allows for web servers to pass requests to web application or other frameworks. Flask accomplishes its purpose through multiple different aspects. We will examine how Flask operates to create a TCP socket connection.
The process of creating a TCP socket connection begins when we call the app.run() method in our app.py server code. From this call the code is then directed to the flask library. Specifically it navigates to the run method in the app.py file in the library (https://github.com/pallets/flask/blob/d0bf462866289ad8bfe29b6e4e1e0f531003ab34/src/flask/app.py#L773). This method set the server name and port. If the server name and port are not set as a parameter then the run method sets them to default values. Additionally, inside of this method there is a method called “run_simple()” (https://github.com/pallets/flask/blob/d0bf462866289ad8bfe29b6e4e1e0f531003ab34/src/flask/app.py#L889) . This method comes the werkzeug.serving library in the serving.py file (https://github.com/pallets/flask/blob/fae889d467a0a19385978f7e979234771a0eb10d/src/werkzeug/serving.py#L907) . When this method is called it intializes all of the necessary information for a socket. Once this method sets all of the values, the server_forever() method is then called (https://github.com/pallets/flask/blob/fae889d467a0a19385978f7e979234771a0eb10d/src/wer

kzeug/serving.py#L1069). This is a helper function (https://github.com/pallets/flask/blob/fae889d467a0a19385978f7e979234771a0eb10d/src/werkzeug/serving.py#L764) written in this same file that called the super().server_forver() method. This is the call which calls the server forever method in the python library (https://github.com/python/cpython/blob/27419a71b5aa18baf24f4e640c5a6e8df9338928/Lib/socketserver.py#L218) . This method is the code inside of the python libraries that create and start the TCP connection. Within this same file there are other methods that handle stopping, handling requests, and other functions. That is the path from our call in our code to the python library that creates the TCP connection.

We use this technology (Flask) to create a TCP socket connection for our app to communicate between the server and client.