The three Python files here combined into a rewrite CS340 final. The original intended to demonstrate my ability to design and then employ a REST API in a client to server to DB-backend relationship. This version is built with the web framework Flask instead. Creating a web app is a way of orienting the code towards a end-user as opposed to an API for a developer.

Much of the original code was over permissioned, to demonstrate the ability to perform CRUD functions upon a DB for example. The improvements have either stripped or rewritten these methods to focus on delivering curated data to the client and reducing the number of writes to the DB. In the original code, the client effectively had indirect write-access to the DB, a major security flaw for a publicly available database. The DB-interaction code will store and cache more data, and in a production environment, serve as a private place to store a company’s proprietary data.

Cross-site scripting attacks concern any site dynamically generating HTML and JavaScript. In the context of this application, the risk is lower, as nowhere in the application is the user ever uploading data that the server would then store. But additionally, Flask escapes all text generated through its templating system, which one prevent any attacker’s code from being executed (Pallets).

I would combat many other common attacks at different OSI layers. Basic DoS attacks from a single IP, or few IPs are easily handled through a firewall filter. For DDoS attacks I would recommend expensing for a dedicated DDoS protection service. The magnitude of attacks from threat actors like the original Mirai botnet can be beyond even large enterprises to prevent. I would instead point a company concerned about DDoS attacks to services like Cloudflare, services that have specifically built out infrastructure to combat DDoS attacks.

I improved my skills with JavaScript this project, it’s not a language I’m very familiar with. Firefox produced a reference for the language to use, but it is possible that other browsers would render the pages differently. Those other browsers would need testing before deployment. Likewise, Flask is new to me, but the API is well documented, and I did want to learn a web framework going into this; as I expect web apps to only gain popularity as a user interface in the future. The greatest trouble I had was trying to understand how Flask passes Python variables to JavaScript. It prints out the data into the source code of the HTML.

Pallets. (2020). *Security Considerations — Flask Documentation (1.1.x)*. Flask.palletsprojects.com. Retrieved 23 November 2020, from https://flask.palletsprojects.com/en/1.1.x/security/.