Joel Miller  
02/06/24  
CPSC 4911 Sprint 0 Notes

Please take notes on the technologies that your team has decided to research. There are some examples provided next to each prompt, but these lists are not exhaustive.  
\*\*Note: Due to the nature of some web-development frameworks (i.e., SvelteKit, Django, Next, …) your frontend and backend may be hosted by the same technology. In this case, please take notes on the specific parts of the framework in the sections below.

**Frontend Language(s) (Javascript / Typescript / JSX / Svelte)**  
URL(s):  
Notes:

* Conducted in-depth research on JavaScript, exploring its various functionalities and features.
* Studied HTML and CSS for structuring and styling web pages respectively.
* Explored JSX as a syntax extension for JavaScript, commonly used in React development.
* Investigated Svelte for its innovative approach to frontend development, focusing on its reactive nature and component-based architecture.

**Frontend Framework/Architecture (Vue / React / Svelte[Kit] / Raw)**  
URL(s):  
Notes:

* Explored the React framework extensively, diving into its component-based architecture, state management solutions like Redux or Context API, and its virtual DOM rendering mechanism.
* Studied the concept of virtual DOM and its role in optimizing DOM manipulation and rendering in React applications.
* Explored React Hooks for managing state and side effects in functional components.
* Researched React Router for client-side routing in single-page applications (SPAs).
* Investigated SvelteKit as a modern framework for building web applications, emphasizing its server-side rendering capabilities and built-in routing features.

**Back End Language (PHP, Javascript, Python)**  
URL(s):  
Notes:

* Researched JavaScript for backend development using frameworks like Node.js, focusing on its event-driven, non-blocking I/O model and ecosystem of libraries like Express.js.
* Explored Python for backend development, studying its simplicity, readability, and vast collection of libraries and frameworks like Django and Flask.

**Back End Framework/Architecture (Laravel, Django, Flask, Node.js)**  
URL(s):  
Notes:

* Investigated Node.js as a runtime environment for executing JavaScript code outside the browser, focusing on its event-driven, asynchronous nature and built-in modules for handling HTTP requests and managing file systems.
* Studied Express.js as a minimalist web application framework for Node.js, exploring its middleware architecture, routing capabilities, and support for building RESTful APIs.
* Researched Django as a high-level Python web framework, emphasizing its "batteries-included" philosophy, built-in admin interface, and ORM (Object-Relational Mapping) for interacting with databases.
* Explored Flask as a lightweight Python web framework, focusing on its simplicity, extensibility, and support for building web applications and RESTful APIs.

**Relational Databases (PostgreSQL, MySQL)**  
URL(s):  
Notes:

* Explored MySQL for relational database management, studying its SQL syntax, data types, indexing mechanisms, and transaction support.
* Investigated PostgreSQL for its advanced features like support for JSON data types, full-text search, and concurrency control mechanisms.
* Studied database normalization techniques for designing efficient and scalable database schemas.
* Researched ORMs (Object-Relational Mappers) like Sequelize for Node.js and SQLAlchemy for Python, exploring their role in simplifying database interactions and abstracting away low-level SQL operations.

**DEPENDENT ON SERVERLESS DECISIONS**  
**Web Servers (Apache, Nginx)**  
URL(s):  
Notes:

* Investigated Apache HTTP Server as a widely used open-source web server, focusing on its configuration options, performance tuning techniques, and support for hosting dynamic content using modules like mod\_php and mod\_python.
* Explored Nginx as a lightweight, high-performance web server and reverse proxy, studying its event-driven architecture, support for handling concurrent connections, and configuration for load balancing and SSL/TLS termination.
* Investigated deployment strategies for web applications on Apache and Nginx servers, including techniques for configuring virtual hosts, managing permissions, and securing the server using SSL certificates and HTTP security headers.