Clint Monroe

CS 499 Computer Science Capstone

Southern New Hampshire University

17 November 2024

Enhancement One: Software Design and Engineering

The primary artifact for my ePortfolio is an interactive animal shelter management dashboard, a web application developed to support a fictional client's need for organizing and analyzing rescue animal data. This dashboard was initially created in a previous course as a Python-based application using the Dash framework. For my capstone project, I enhanced the application by transitioning its architecture to React for the frontend and FastAPI for the backend, implemented with SQLite as the database. These upgrades significantly improve performance, scalability, and the user experience. The enhancements I've made this week include setting up a React and FastAPI environment, configuring CORS, implementing Axios for API requests, establishing database communication, and setting up basic CRUD operations. Together, these components provide a functional MVP that allows users to add and view animal records.

I selected this dashboard for my ePortfolio because it represents a comprehensive application of my skills in software design, algorithms, and database management. It demonstrates my ability to apply industry-standard technologies and best practices, such as asynchronous processing, component-based architecture, and ORM-based data management. These improvements showcase my capability in designing a solution that not only meets user needs but also aligns with software engineering principles. By moving sorting and filtering operations to the database layer, optimizing data structures with ORM models, and implementing

asynchronous API requests, I'm showcasing my ability to create efficient and maintainable applications.

The enhancements align closely with the course outcomes I aimed to achieve in Module One. I demonstrated effective software design and the application of tools and techniques, building an MVP that emphasizes algorithmic efficiency and data handling. I plan to further align with security-focused outcomes by implementing measures to mitigate vulnerabilities in FastAPI. Additionally, the structured communication between React and FastAPI highlights my progress in achieving a collaborative environment, setting up the application for scalability and teamwork.

Throughout the enhancement process, I learned a great deal about handling frontend-backend communication in a full-stack environment, particularly with asynchronous requests in FastAPI. The initial setup of React and FastAPI presented challenges, especially in configuring CORS and ensuring seamless data transfer through Axios. However, overcoming these issues improved my understanding of real-world integration challenges, especially in aligning frontend and backend components in a scalable and responsive way. This experience has prepared me to tackle additional optimizations and security configurations in future sprints, with a focus on delivering a final product that is robust, secure, and user-friendly.

I would like to note that due to the interconnected nature of building a full-stack application, I've found it challenging to focus exclusively on Software Design and Engineering without addressing Algorithms, Data Structures, and Database considerations simultaneously. Recognizing this overlap, I have structured the work into four distinct sprints, covering weeks three through six, to ensure a balanced approach across all areas of development. Since the project requires milestones in the initial three weeks, I've allocated the final sprint for backlog

tasks and polish to refine the application. Each sprint has defined goals, tasks, and milestones, aligning with both user and developer stories that I crafted to guide development. The plan for each sprint is as follows:

- **Sprint 1: Project Initialization & Environment Setup** – This sprint focuses on setting up the project plan and environment, with basic configurations in React, FastAPI, and SQLAlchemy.
- **Sprint 2: Core Feature Development** – Emphasizes frontend interactivity and backend efficiency through asynchronous API endpoints and interactive data features.
- **Sprint 3: Data Optimization & Security** – Optimizes data structures and queries within SQLAlchemy to enhance performance and security.
- **Sprint 4: Backlog & Polish** – Reserved for completing remaining items, enhancing the UI/UX, and final testing to ensure a seamless, reliable user experience.