The artifact I selected for enhancement was the dashboard from my CS-340 project, which was originally built in a Jupyter Notebook using Python Dash. The dashboard visualized animal rescue data and queried a MongoDB database using a custom CRUD module. Although it was fully functional, the original setup had several limitations that impacted maintainability, security, and potential deployment outside the notebook environment.

To address these concerns, I transformed the notebook into a standalone Python application by refactoring the code into an app.py script. I replaced hardcoded database credentials with environment variables loaded from a .env file, significantly improving the project's security and flexibility. This change also aligned the application more closely with industry standards for configuration management.

Another major enhancement was switching from JupyterDash to dash.Dash, which allowed the app to run as a standalone server, independent of Jupyter. This made the project more modular and deployable in real-world settings. I also set up a secure, locally hosted MongoDB instance with authentication to replace the previous cloud-based setup. All core dashboard functionality, including filters, interactive charts, and map visualizations ,was preserved and fully functional in the new version.

This enhancement clearly aligns with Course Outcome #4, demonstrating the use of well-founded and innovative techniques in computing practices. It also supports Course Outcome #3 by improving the design structure for long-term maintainability and showcasing thoughtful trade-off management in engineering decisions. Overall, I learned how architectural improvements, like removing hardcoded values and isolating configuration , can dramatically improve software quality, security, and readiness for deployment.