**7-2 Project Two Submission**

**README**

**Vincent Yanez**

**CS-340-Client-Server-Development-7-2-Project-Two-Submission**

Describe the required functionality of the project. Include the screenshots or screencast taken while testing and deploying your dashboard (Step 6) as proof that you have achieved the required functionality.

Grazioso Salvare Dashboard Project

The project involves creating a dashboard that connects to a MongoDB database to display and interact with animal data. The dashboard lets users search and filter this data easily. To show that the dashboard works correctly, screenshots or screencasts of the dashboard in action should be included as proof of successful testing and deployment, demonstrating that the required features work as intended.

Describe the tools used to achieve this functionality and a rationale for why these tools were used.

Tools Used: Python: Python was selected for its ease of use, clear syntax, and rich library ecosystem, making development fast and efficient.

Jupyter Notebook: Jupyter Notebook was used to write and test code in an interactive environment, which helped streamline the development process.

Dash: Dash was used to create interactive data visualizations and build the web application's user interface. Its Python-based nature allowed for easy integration with the rest of the project.

Plotly: Plotly was used to generate dynamic charts and graphs within the Dash application, providing interactive data visualization features that are crucial for analyzing rescue data.

.Be sure to explain why MongoDB was used as the model component of the development, including what specific qualities or capabilities it provides for interfacing with Python.

MongoDB: MongoDB was chosen for its flexibility with unstructured data and easy integration with Python, making storing and retrieving data easy. MongoDB's strong indexing and querying support made it ideal for efficiently managing rescue event data.

.Be sure to explain the Dash framework that provides the view and controller structure for the web application.

Dash makes it easy to build interactive web apps using Python, handling both the user interface and the logic behind it. It allows for smooth integration of data visualizations and controls without needing HTML or JavaScript.

.Be sure to include links to any resources or software applications that were accessed or used.

Links Used: Python: Python.org Jupyter Notebook: Jupyter.org Dash: Dash by Plotly MongoDB: MongoDB

Explain the steps that were taken to complete the project.

Here are the simplified steps taken to complete the project:

-Set Up Environment: Installed necessary software and libraries, including Python, Dash, and MongoDB.

-Develop Backend: Created a MongoDB database to store rescue event data and set up data processing scripts.

-Create Frontend: Designed interactive dashboards using Dash for visualizing data.

-Integrate Components: Connected the Dash frontend with the MongoDB backend to display dynamic data.

-Testing: Tested the application for functionality and made adjustments as needed.

-Deploy: Published the application for access and use.

Identify any challenges that were encountered and explain how those challenges were overcome.

Challenges and Solutions: Server Issue: Encountered a server port conflict several times. Resolved by closing the conflicting process and using a different port.

Data Integration: Had trouble fully integrating and displaying mountain rescue and disaster data. Managed to get a basic map and graph working by verifying data formats and sources, but some issues remain unresolved.