**README File for Grazioso Salvare Dashboard Project**

**1. Project Functionality**

For this project, I designed a dynamic dashboard tailored for Grazioso Salvare. This dashboard allows users to seamlessly access, query, and visualize various datasets.

**Proof of Functionality:**  
You can verify the dashboard's functionality through the screenshots and screencast I took during its testing and deployment:  
  
A screenshot of a computer

Description automatically generatedA screenshot of a dashboard

Description automatically generatedA screenshot of a computer

Description automatically generatedA white rectangular object with a black border

Description automatically generatedA screenshot of a computer

Description automatically generated

**2. Tools I Used & Rationale**

* **MongoDB:** I chose MongoDB, a NoSQL database, as the model component for this project. My reasons:
  + Its flexible, schema-less nature.
  + Its ability to handle vast data volumes.
  + The efficient support it offers for Python, aiding in data analytics.
* **Dash Framework:** I employed Dash by Plotly for the web application, offering both a view and controller structure. With Dash, I found:
  + Crafting interactive, web-based data visualizations is intuitive.
  + It integrates smoothly with Python, eliminating the need for advanced web development know-how.

**3. Resources & Software Applications**

During the development, I frequently referred to and used these resources and applications:

* MongoDB
* Python

**4. Steps I Took to Complete the Project**

1. I began by setting up MongoDB and designing the database schema.
2. After gathering and cleaning the necessary data, I populated the MongoDB database.
3. I set up a Python environment and installed all the necessary packages.
4. I meticulously designed the dashboard's UI layout.
5. I then integrated this dashboard with MongoDB to retrieve and present the data.
6. I added interactive features and visualizations
7. After thorough testing for various scenarios, I refined certain aspects for optimal performance.
8. Finally, I deployed the dashboard to our chosen platform/server.

**5. Challenges I Encountered & How I Overcame Them**

* **Data Inconsistency:** Some data sources had mismatched formats. I tackled this by crafting data validation and transformation scripts in Python.
* **UI Responsiveness:** I noticed the dashboard wasn't displaying well on mobiles. To rectify this, I utilized the responsive attributes of Dash components and tweaked the CSS accordingly.
* **Data Retrieval Speed:** As the datasets grew, some queries started lagging. To counteract this, I optimized our MongoDB queries and indexed the fields we accessed regularly.