### **Deployment Plan**

#### **Overview**

The goal of this data product is to provide an interactive platform for data analysis and visualization, tailored specifically for diabetes data. It enables users to upload datasets, perform preprocessing tasks, visualize data, resample datasets, and evaluate machine learning models. The software will be deployed on a cloud platform (Heroku) to ensure scalability and accessibility for authorized users across different devices.

#### **Assumptions, Dependencies, Constraints**

* **Assumptions**: Users have basic understanding of data analysis and machine learning.
* **Dependencies**: The software depends on Python libraries such as Dash, Plotly, pandas, scikit-learn, and matplotlib. Reliable internet connectivity is required for cloud operations.
* **Constraints**: The performance might be limited by the hardware specifications of the cloud server, especially for large datasets and intensive computing tasks.

#### **Operational Readiness**

* **Testing**: Before deployment, comprehensive testing is performed to ensure every function operates correctly. This includes unit tests for data processing functions and integration tests for user interactions and data flow.
* **Preparations**: Configuration of cloud resources, setting up security measures like SSL/TLS for secure data transmission, and preparing backup systems for data redundancy.

#### **Data Conversion**

Initial data to be used by the system will be loaded and validated through an automated script that checks data integrity and formats it according to the application's requirements.

#### **Phased Rollout**

* **Phase 1**: Deploy basic functionality like data upload, visualization.
* **Phase 2**: Incorporate more complex features such as machine learning model evaluations and resampling.
* **Phase 3**: Add the help function to app after successful deployment and add Styling to the app. Currently styling is minimal.
* **Phase 4:** Introduce a log in and log out measures to the app once confirmed it is working to restrict it a specific user group (in development)

#### **Support**

* **Help Feature**: A comprehensive "Help" section will be integrated within the application, offering step-by-step guides on using each feature.

**Release Planning**

* Methods:
  + Agile Methodology: We'll use Agile methodologies, incorporating iterative development and frequent feedback loops to ensure continuous improvement and adaptability.
  + Version Control: Utilizing GitHub for version control to manage changes efficiently and collaborate effectively.
  + CI/CD Pipeline: Implementing Continuous Integration and Continuous Deployment (CI/CD) pipelines with GitHub Actions to automate testing, building, and deployment processes. This ensures that every code change is verified and deployed seamlessly.
* Tasks:
  + Initial Setup:
    - Configure the Heroku environment and create the necessary resources (e.g., database, add-ons).
    - Set up environment variables and configurations required for the application.
  + Development:
    - Implement core features of the Diabetes Dash app, such as data visualization, user authentication, and data management.
    - Ensure all code is thoroughly tested using unit tests and integration tests.
  + Testing:
    - Conduct thorough testing including unit tests, integration tests, and user acceptance testing (UAT) to ensure functionality and performance.
    - Address any bugs or issues identified during testing phases.
  + Deployment:
    - Automate deployment to Heroku using GitHub Actions.
    - Set up automated monitoring and logging to track the health and performance of the application.
* Contingency Plans:
  + Backup and Recovery:
    - Regularly back up the database and critical data to ensure recovery in case of data loss or corruption.
    - Implement disaster recovery plans to quickly restore services in case of major failures.
  + Rollback Strategy:
    - Maintain a rollback strategy to revert to the previous stable version of the app in case a new release introduces critical issues.
    - Use feature flags to enable or disable features without requiring full deployments.
  + Monitoring and Alerts:
    - Set up comprehensive monitoring using tools like Heroku Metrics, New Relic, or Datadog to track application performance and health.
    - Configure alerts for critical metrics such as response time, error rates, and resource utilization to proactively address issues.
  + Scaling Plan:
    - Design the application to scale horizontally and vertically to handle increased load and ensure high availability.
    - Use Heroku's auto-scaling features to adjust resources based on demand.

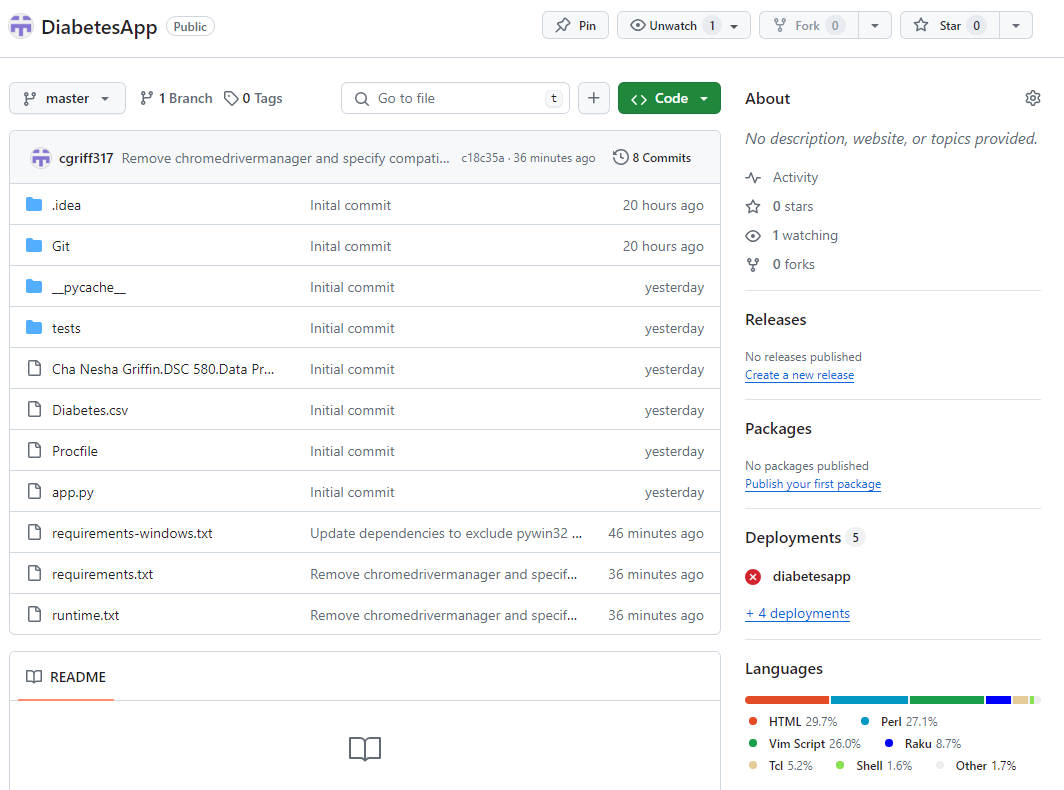
#### **Choosing a Cloud Platform**

Heroku would be suitable for hosting this application given their robust scalability options, support for Python-based applications, and extensive machine learning libraries. Heroku is cost effective and can automate the deployment, from capacity provisioning and load balancing to auto-scaling.

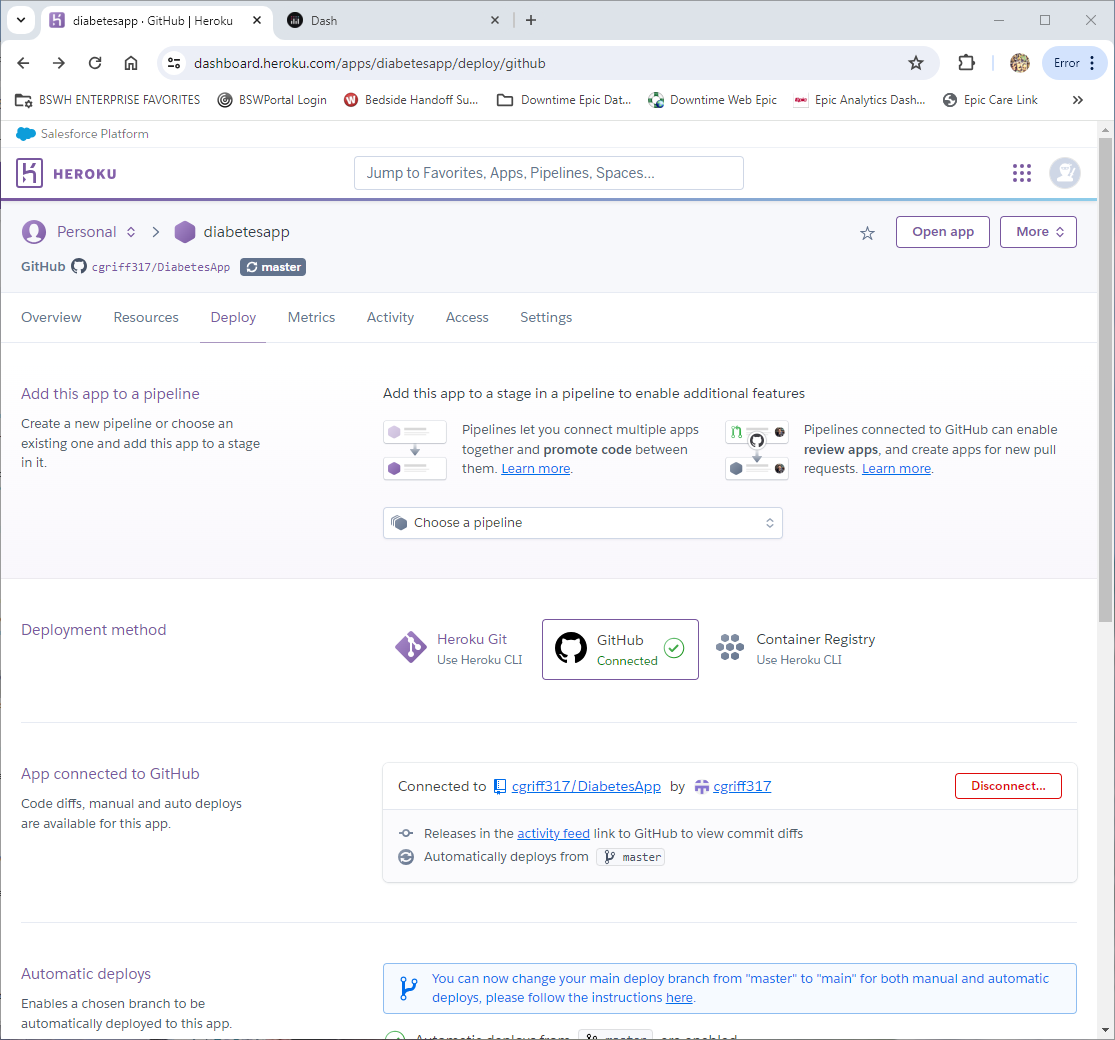
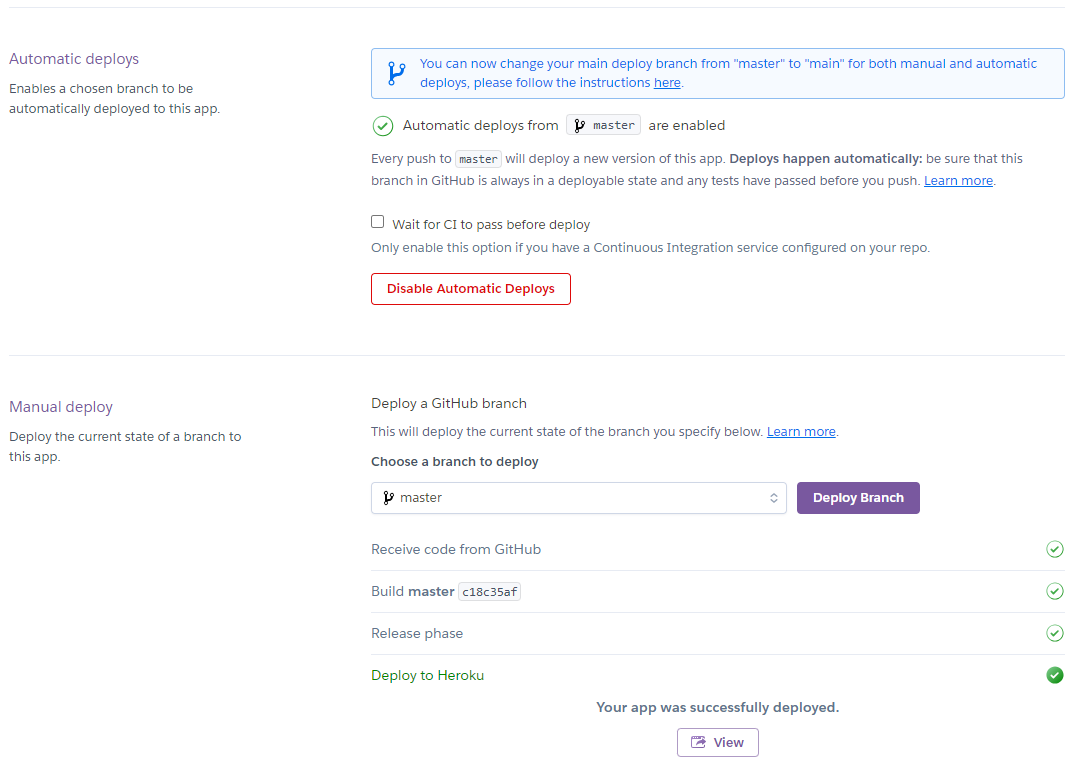
**Demonstration of Product Deployment**

1. Show the data product installed and running on a cloud-based platform. Provide clear evidence of the product being installed and running. This could be a server installed on your hosting account, a cloud service provider, or something similar.

* Step 1: Prepare GitHub Repository
  + Push Python app to a GitHub repository. Ensure repository includes necessary files like requirements.txt, runtime and Procfile.



* Step 2: Connect GitHub to Heroku
  + Log in to Heroku account and go to the Dashboard.
  + Create a new app:
    - Click on "New" and then "Create new app".
    - Enter a unique app name and choose region.
    - Click "Create app".
* Step 3: Deploy via GitHub
  + Go to your app's Deploy tab.
  + In the Deployment method section, click on "GitHub".
  + Connect GitHub account to Heroku.
  + Once connected, search for repository name and connect it.
  + Choose the branch wanted to deploy (usually main or **master**).
  + Click on "Deploy Branch" to manually deploy the app from GitHub to Heroku.
    - If any errors arise when trying to deploy, Heroku will show the error in the Build Mater window, and you can debug.

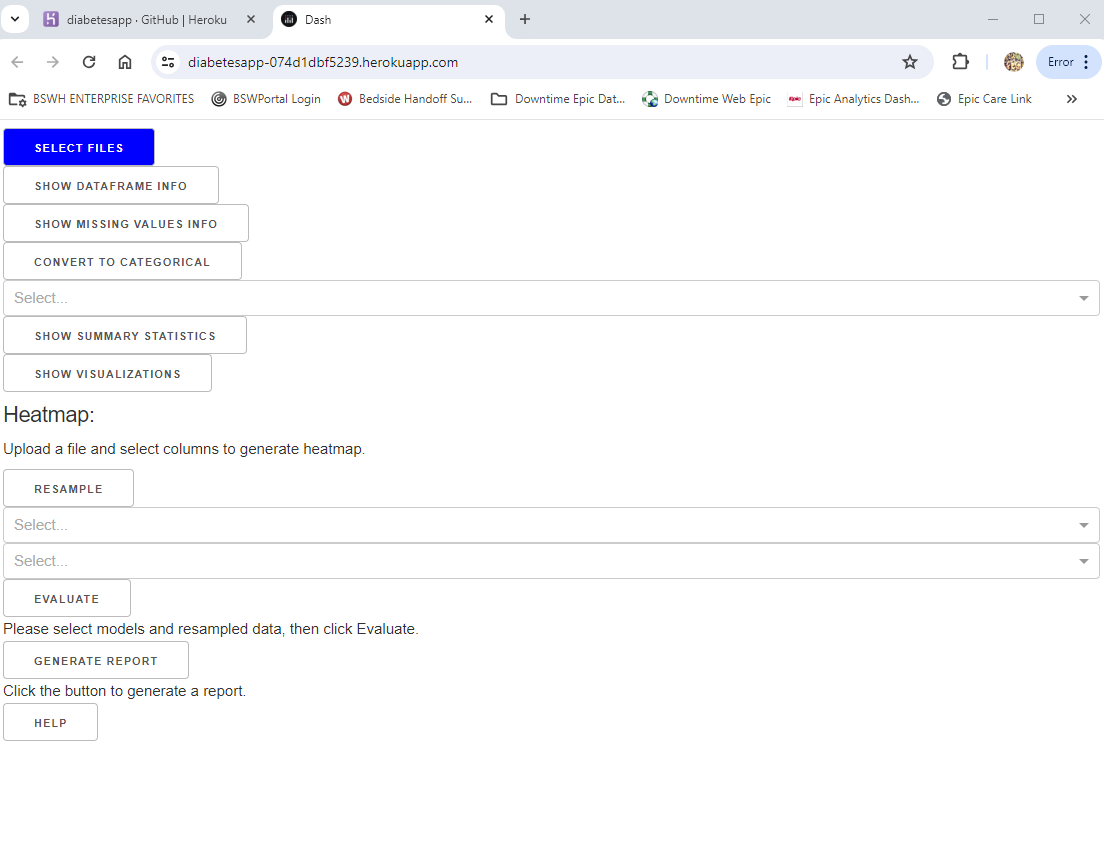
* Step 6: Access the App
  + Once the deployment is complete, access the app by clicking on the "View" button on the Heroku dashboard.

1. **Provide a URL where one can access the data product. The product must be hosted on a website that allows the instructor to test it. It could be a web-hosting account, shiny.io, or a cloud service provider.**

<https://diabetesapp-074d1dbf5239.herokuapp.com/>

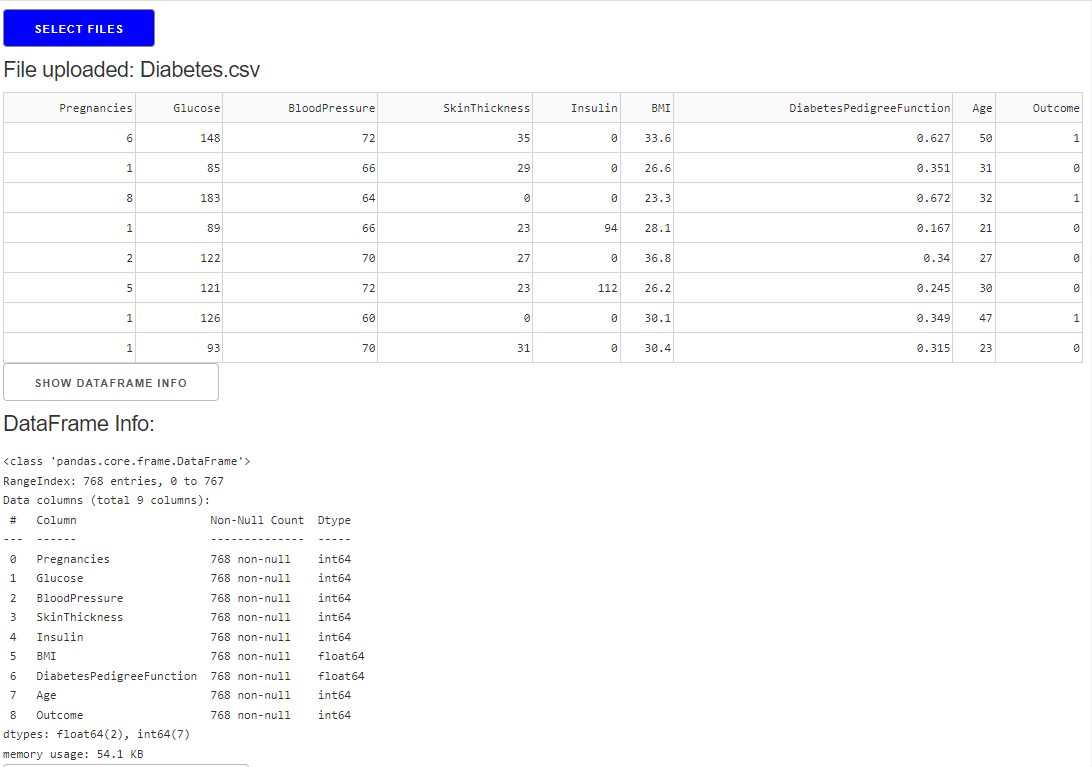
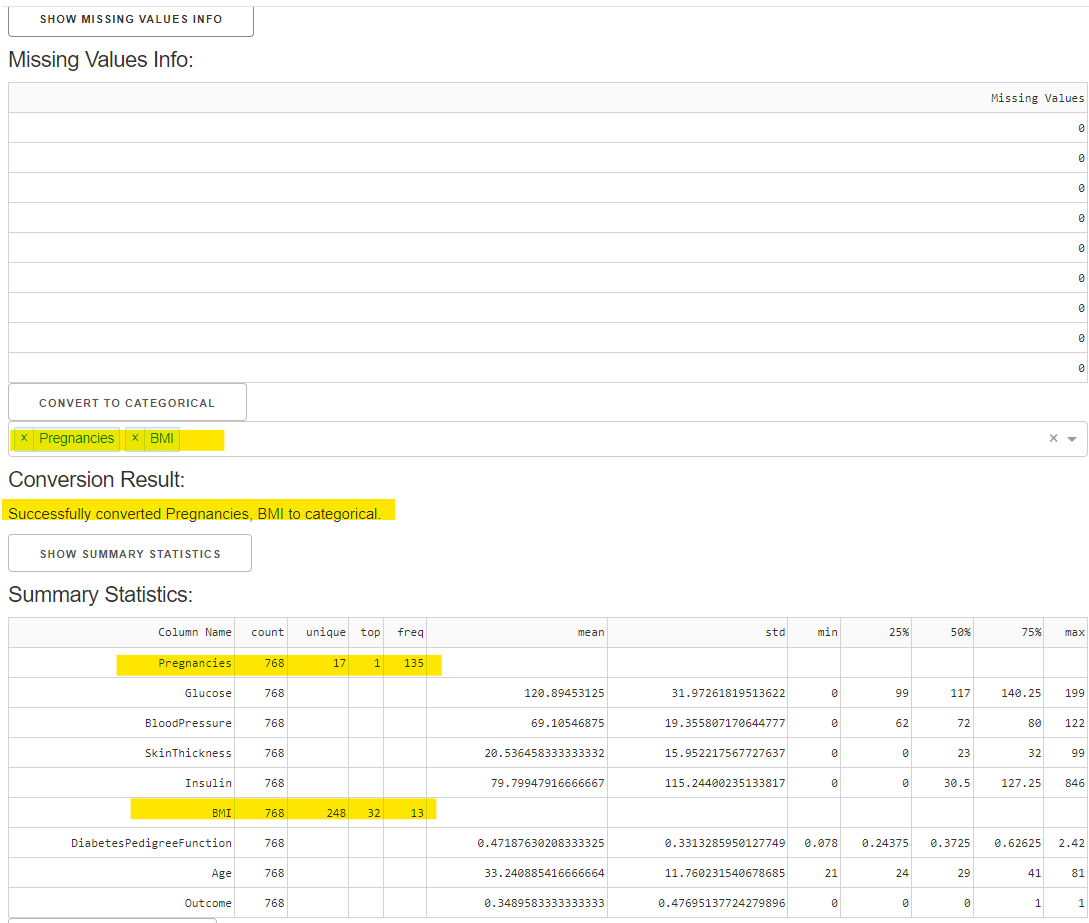
1. **Implementation of all features defined and described in previous topics.**

* The screenshot below shows all the implementations that were required from previous topics:
  + Ability to select files to upload
  + Option to show information that is in the Dataframe
  + Option to see missing values
  + Option to convert categorical data for analysis
  + Ability to see Summary Statistics
  + Ability to see Visualizations (i.e. charts, graphs etc.) of the data being analyzed
  + Option to generate reports that will be more defined in Topic 8
  + A Help button that instructs users on how to interact with the Dashboard

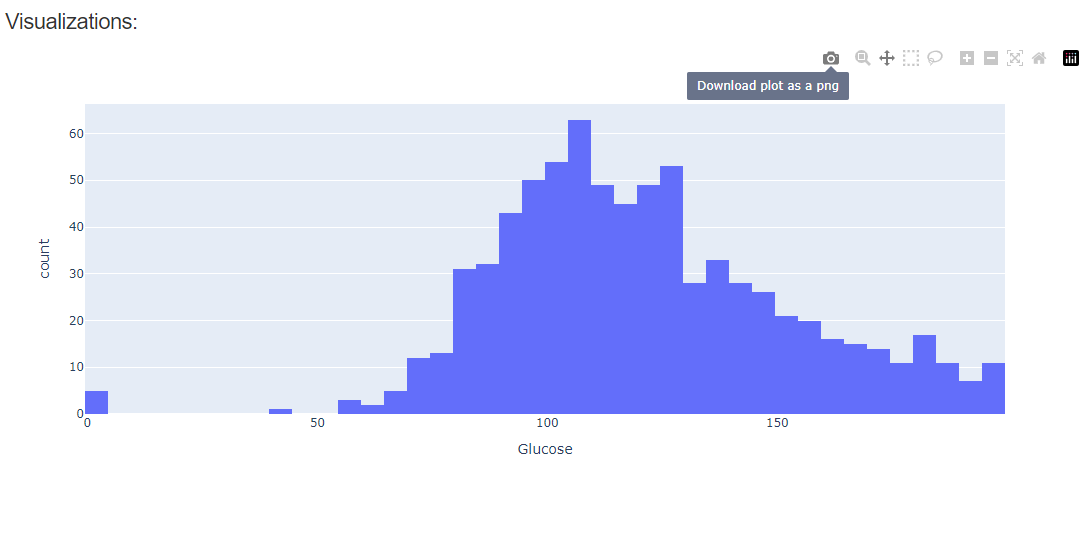


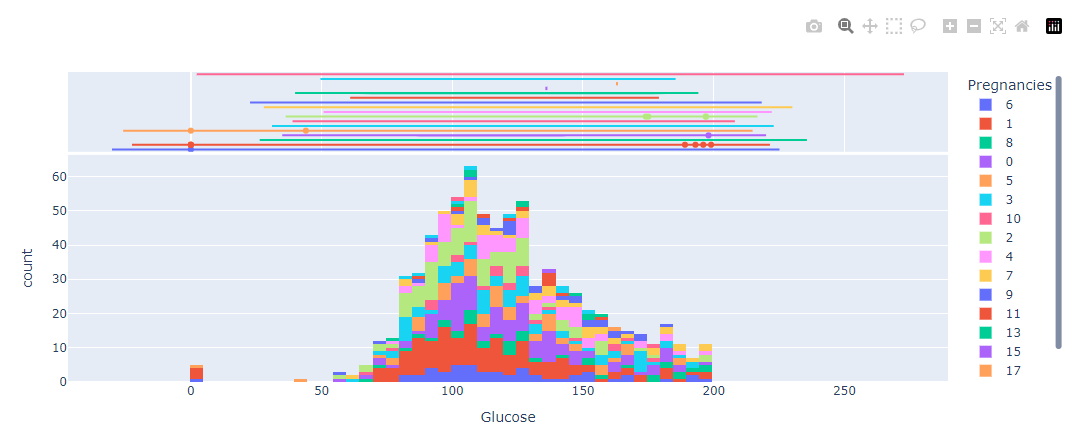
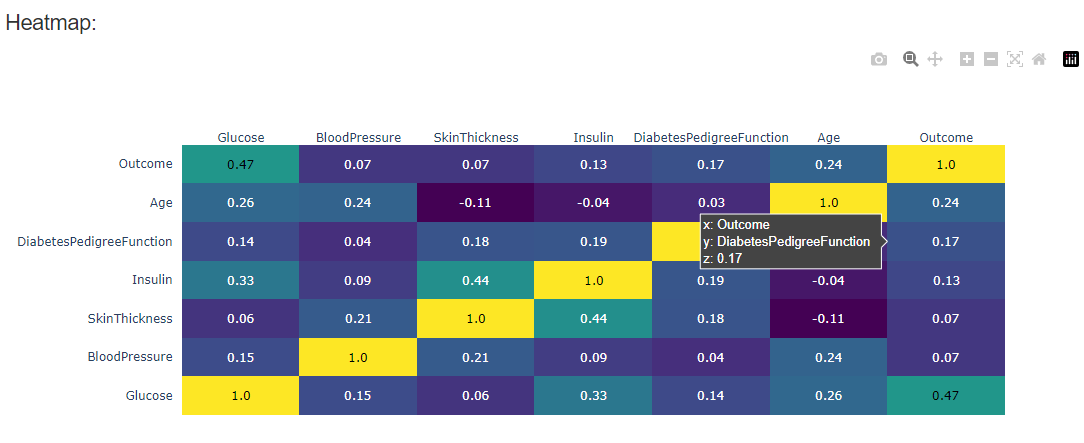
1. **Interactive data mining and analysis. Demonstrate how the product enables one to mine data, analyze it, and visualize various steps and results.**

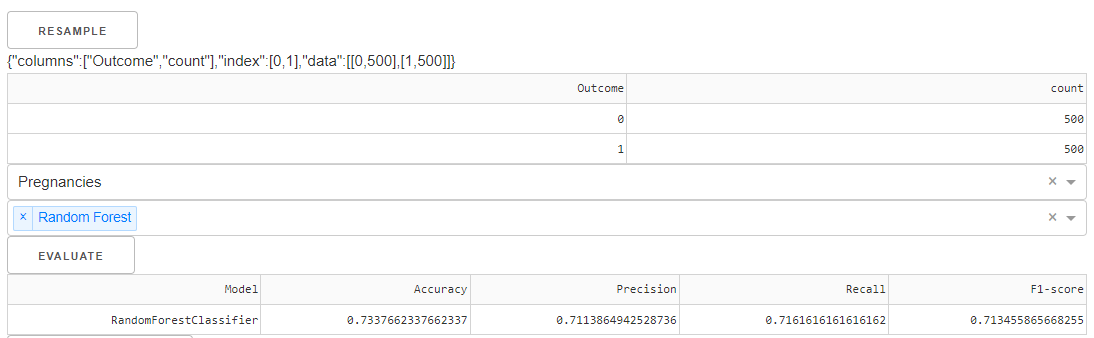
* Upload CSV file and interact with the data, checking Dataframe info, review missing values and transform categorical data

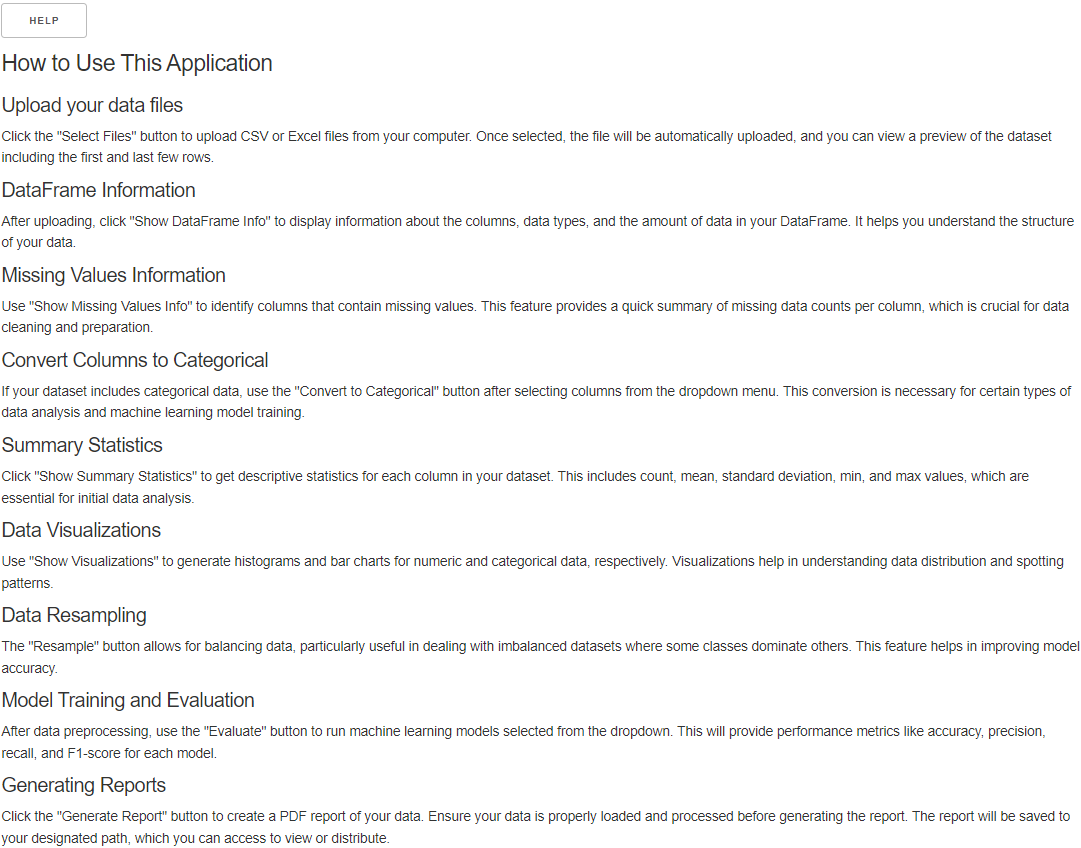
 

* Visualize and interact with visualizations. There are many options like down loading a plot/chart, zooming, select blocks etc.

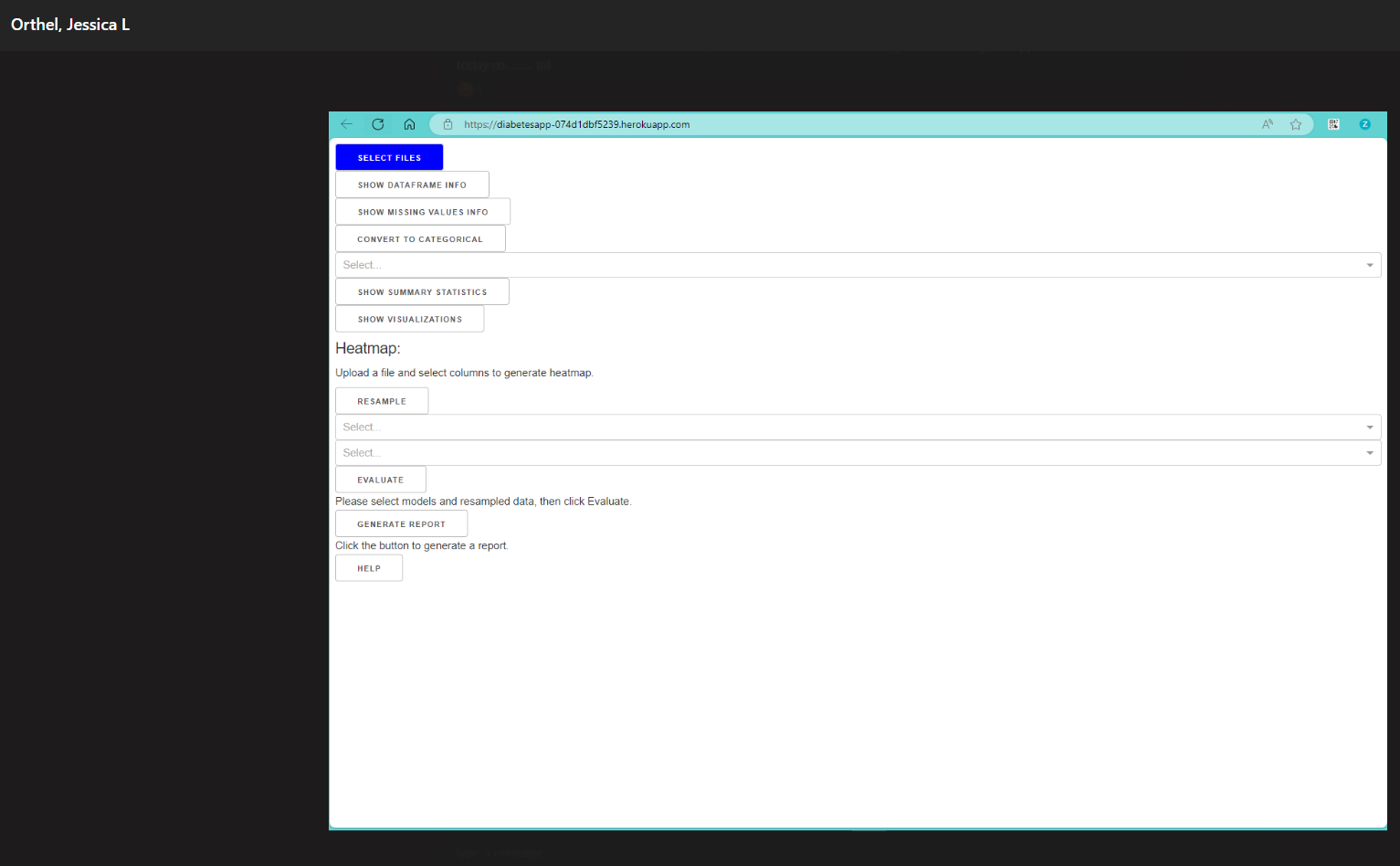
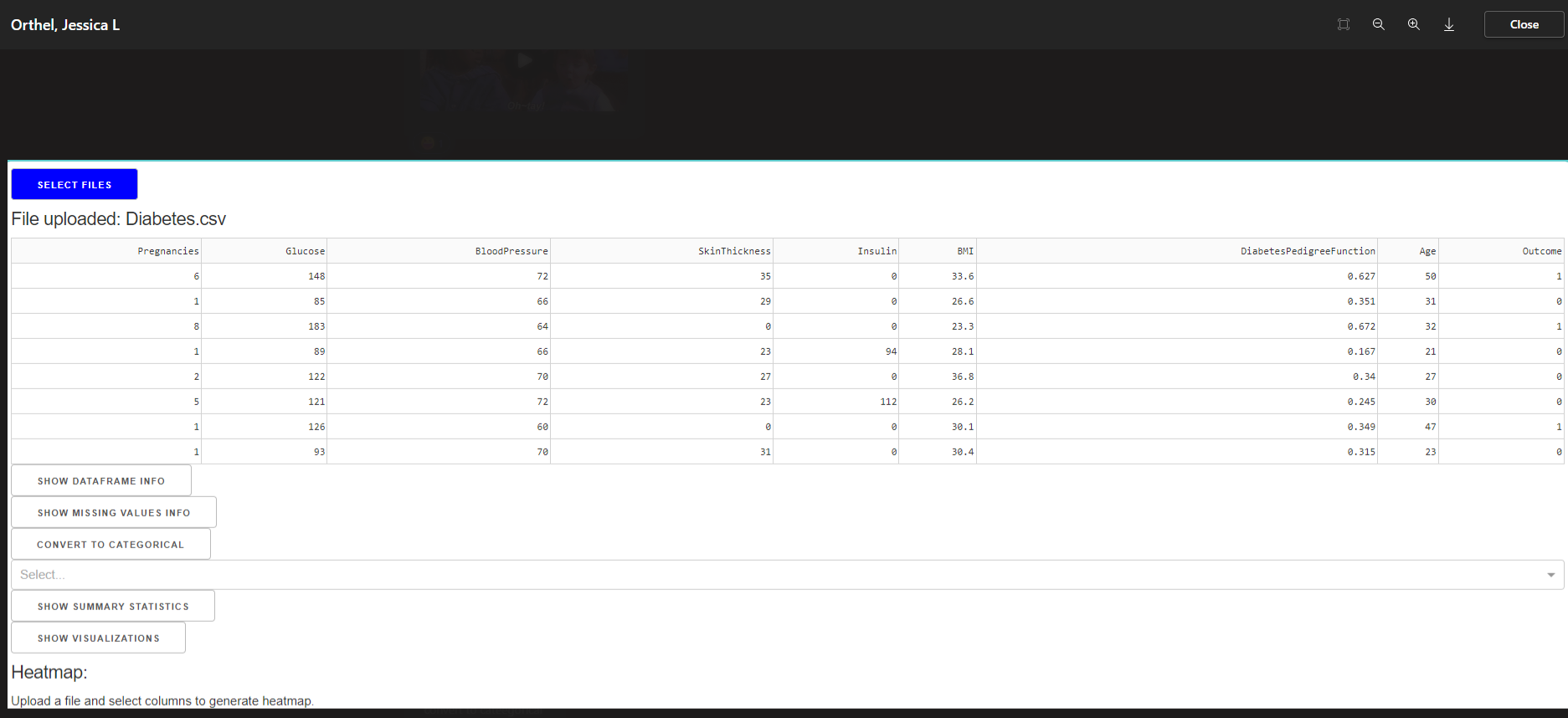
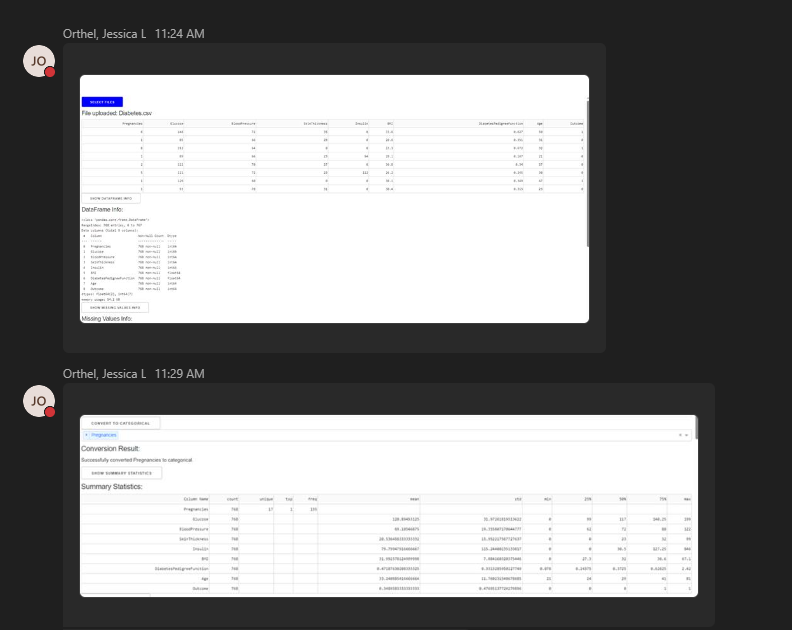






1. **Support for multiple simultaneous users. Demonstrate how multiple simultaneous users can access the website hosting your product.**

* I had a co-worker access my Dash app via the link provided (<https://diabetesapp-074d1dbf5239.herokuapp.com/>) at the same time I was in the app interacting with it:  

1. **A placeholder for web-based report generation. Indicate where on the webpage reports will be presented. Show an example of a report being requested, produced, and presented.**

* This is the place holder for report generation. Currently I am getting an error when I select a button, so I am working to debug. Reports will be visible at the bottom of the Dashboard application right before the HELP button:

