**Interactive Data Exploration Dashboard**

An intuitive tool for data visualization, exploration, and analysis

Prepared by: Himakar Raju  
Date: 2024-11-06



# Introduction

**Objective:**

The Interactive Data Exploration Dashboard is designed to simplify data exploration and visualization for analysts and decision-makers by allowing easy upload, exploration, and reporting of datasets.

**Audience:**

This document is intended for data analysts and management who seek efficient ways to derive insights from datasets.

## Tool Overview and Interface

**Main Functionalities:**

* File upload
* Data visualization
* Report generation

## Technologies Used:

1. Streamlit: For the web application framework.
2. Plotly Express: For creating dynamic and interactive charts.
3. Pandas: For data manipulation and analysis.
4. Sweetviz and ydata-profiling: For generating in-depth exploratory data analysis reports.
5. D-Tale: For an interactive data exploration environment.
6. User Interface Design: Streamlit’s layout features make the dashboard visually engaging, with a wide mode, light theme, and centered text.

# Step-by-Step Workflow

## Setting the Interface

Description: The code sets the layout to a wide mode, loads the company logo as an icon, and applies a serif font style with centered elements to improve readability.

Code Block with Annotations:  
import streamlit as st  
from PIL import Image  
st.set\_page\_config(  
 layout="wide",  
 initial\_sidebar\_state="collapsed",  
 page\_title="Interactive Data Exploration Dashboard",  
 page\_icon=Image.open(r"C:\Projects\LOGO.png"),  
)

## File Upload

Functionality: Allow users to upload CSV and Excel files.

Process:  
- The file is validated to ensure it's in a supported format.  
- Upon successful upload, a folder is created (if not already present) with the filename for saving reports and charts.

uploaded\_file = st.file\_uploader("Upload a CSV or Excel file", type=["csv", "xlsx", "xls"])

## Data Display and Graphing

Data Preview: Display a preview of the uploaded dataset for initial inspection.

Graphing Options: Users can select chart types (scatter, line, bar, pie) and assign columns to axes.

st.subheader("Graph Options:")  
graph\_type = st.selectbox("Select a graph type", ["Scatter", "Line", "Bar", "Pie"])

## Data Exploration Features

Descriptive Statistics: Provides a checkbox option to view statistical summaries.  
Duplicate and Missing Value Analysis: Users can identify duplicates and missing values.  
Correlation Matrix and Heatmap: Users can explore relationships among numerical variables via correlation matrix and heatmap.

## Report Generation

Sweetviz Report:

import sweetviz as sv  
report = sv.analyze(df)  
report.show\_html(filepath=f"{folder\_name}/sweetviz\_report.html", layout='vertical', open\_browser=True)

ydata-profiling Report:  
from ydata\_profiling import ProfileReport  
profile = ProfileReport(df)  
profile.to\_file(f"{folder\_name}/ydata\_profiling\_report.html")

D-Tale Integration:  
import dtale  
d = dtale.show(df)  
d.open\_browser()

# Error Handling and Feedback Mechanisms

Error Messages: Errors are captured and displayed to the user in a user-friendly format, aiding troubleshooting and user guidance.  
try:  
 # Data processing logic  
except Exception as e:  
 st.error(f"An error occurred: {e}")

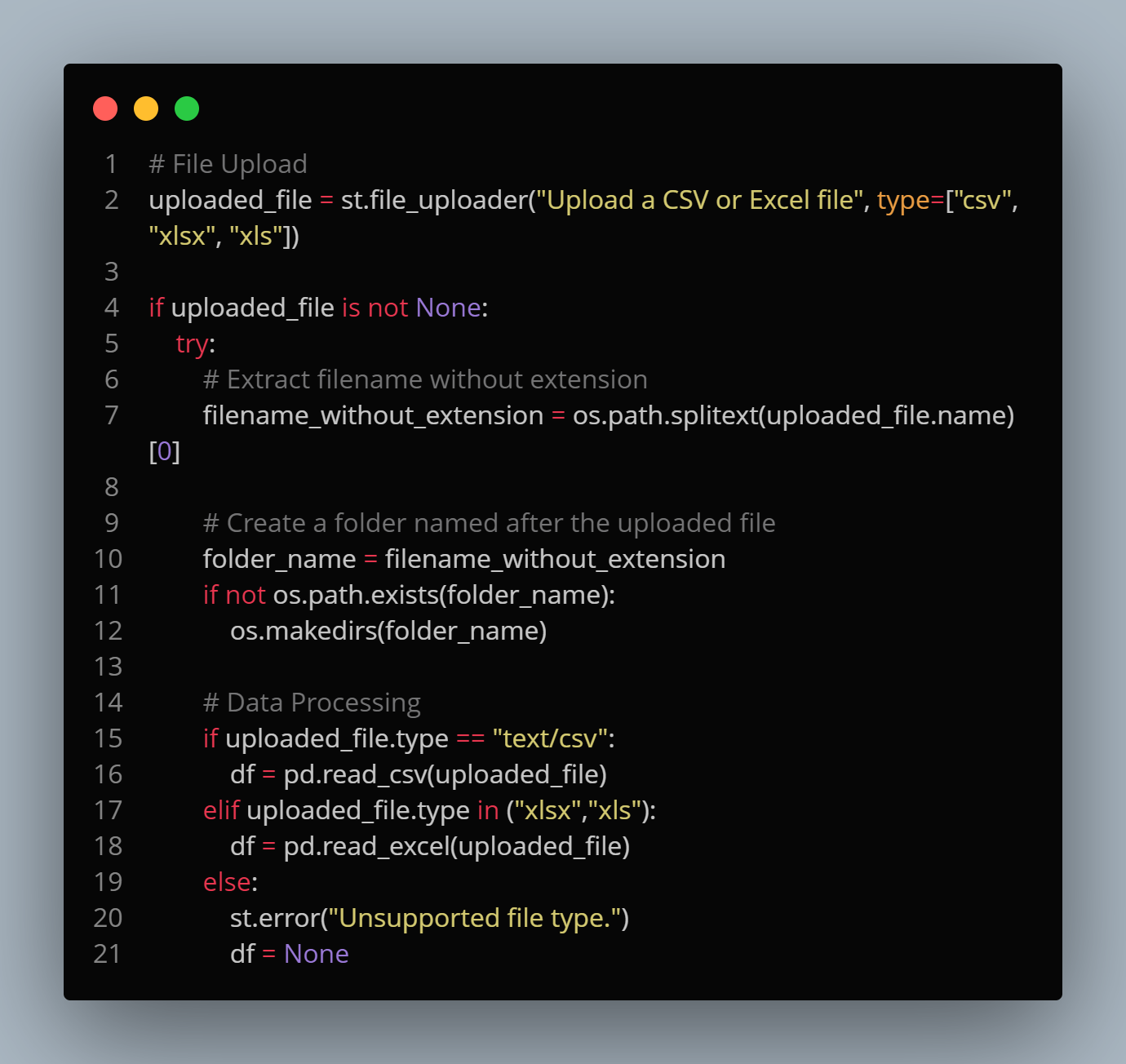
# Appendix

Python Code Reference: Complete code listing for technical review.

**Imports**

**Setting up the webpage view**

**File Upload and Data Processing**



**Data Display and Visualization**



**Data Exploration and Transformation**



**Report Generation**



# Conclusion

This dashboard significantly improves data exploration efficiency and decision-making.

**Possible Future Implementations:**

Potential expansions could include machine learning integration for predictive insights.