Project Documentation: CSV Data Cleaning Script

1. Project Overview

The purpose of this project is to create a Python script that automatically cleans a messy CSV dataset. Real-world datasets often contain missing values, duplicates, and incorrect data types that can negatively affect data analysis and machine learning models.

2. Objectives

- Handle missing values by replacing them with meaningful values (mean, median, or placeholders).
- Remove duplicate rows to maintain data integrity.
- Convert columns into their correct data types.
- Export the cleaned dataset for further analysis.

3. Methodology

The project follows these steps:

Step 1: Import the dataset

- The CSV file is read into a Pandas DataFrame using pd.read_csv().

Step 2: Handle missing values

- Age Column
    - Converted to numeric using pd.to_numeric().
    - Missing values replaced with the mean age of the dataset.
    - Column converted to integer type.

- Fare Column
    - Missing values are replaced with "NA" (string placeholder).

- Other Columns
    - Missing values replaced with "Missing" to ensure no null entries remain.

Step 3: Remove duplicate rows

- Used drop_duplicates() to eliminate redundant entries.

Step 4: Data type conversion

- Ensures numeric fields (like Age) are correctly typed as integers.

- Other fields retain consistent data types.

Step 5: Export cleaned data

- The final cleaned dataset is saved into a new CSV file (Titanic_Dataset_Cleaned.csv).

Challenges Faced & Solutions

- Challenge 1: Handling inconsistent data types
  Some columns (like *Age*) contained strings instead of numbers.
  Solution: Used pd.to_numeric(errors="coerce") to convert invalid values into NaN, which could then be filled with the mean.

- Challenge 2: Missing values in different columns
  Not all columns could be treated the same way.
  Solution: Applied specific strategies:
    - Mean imputation for numeric columns (Age).
    - Placeholder "NA" for Fare.
    - "Missing" for categorical/text columns.

- Challenge 3: Duplicates in dataset
  Duplicate entries reduced data quality.
  Solution: Used Pandas drop_duplicates() to remove redundant

Conclusion

This project demonstrates how **data preprocessing** can be automated using Python. The script provides a general-purpose cleaning pipeline that can be reused on other datasets with minimal changes. Clean data is now ready for analysis and visualization