

Samsung Innovation Campus GSSSIETW, Mysore

Project Title: Employee Salary Data Cleaning and Aggregation for HR Insights

Description: The project “Employee Salary Data Cleaning and Aggregation for HR Insights” focuses on converting raw salary data into reliable and structured information for HR use. It involves cleaning missing and inconsistent values, standardizing records, and aggregating data such as average salary and departmental distribution. Visualizations like charts and graphs are used to highlight trends and patterns. This project supports HR teams in making data-driven decisions for workforce and compensation planning.

Submitted By:

Preethi K M

5th semester

AI & DS Department

E-mail: kmpreethi08@gmail.com

Table of the content:

- Problem statement
- Objectives and features
- UML
- Detailed explanation
- Code explanation
- Output and explanation
- Usage instruction
- Troubleshooting
- Bibliography

Problem Statement: The HR department of a company maintains employee records, including salaries, departments, and job titles. However, the dataset contains duplicates, missing values, and inconsistent data formats.

The goal is to clean the employee dataset using Pandas, perform aggregation to extract useful HR insights, and export the results for reporting

Project Objectives

- **Data Cleaning:** Automated removal of duplicates and standardization of employee records
- **Data Enrichment:** Calculate derived metrics like years of service
- **Analysis:** Generate department and job title performance summaries
- **Visualization:** Interactive dashboard for HR insights and reporting

- **Export Capabilities:** CSV export functionality for further analysis

Features :

Key Performance Indicators

- Total Employees: Current employee count with filtering
- Average Salary: Mean salary across selected filters
- Total Salary Cost: Sum of all salaries
- Average Years of Service: Mean employee tenure

Interactive Filters

- Department Filter: Dropdown to select specific department or view all
- Real-time Updates: All metrics and charts update automatically

Visualizations

1. Salary Distribution Histogram

- Shows salary range distribution
- Identifies salary clusters and outliers
- Updates based on department filter

2. Department Salary Comparison

- Bar chart comparing average salaries by department
- Helps identify compensation disparities
- Visual department performance comparison

3. Salary vs Experience Scatter Plot

- Relationship between tenure and compensation
- Color-coded by department
- Identifies salary progression patterns

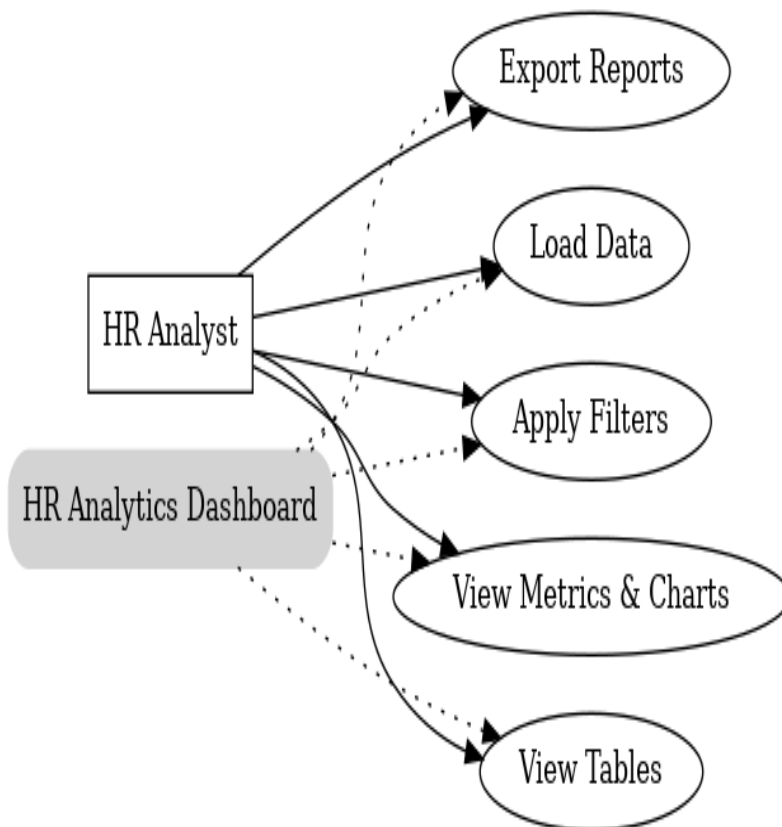
Export Functionality

4. CSV Downloads: Export any table or filtered dataset
5. Custom File Names: Descriptive export file naming
6. Real-time Data: Always exports current filter state

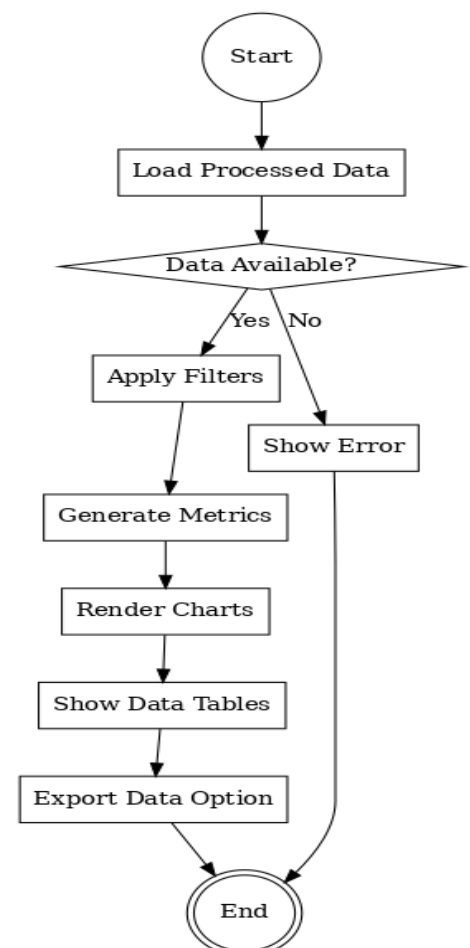
Export Functionality

- CSV Downloads: Export any table or filtered dataset
- Custom File Names: Descriptive export file naming
- Real-time Data: Always exports current filter state

UML Diagram:



Use case diagram



Activity diagram

Structure of the project:

mini_project/

|

| —  data/

| └─ employees.csv # Raw employee dataset

|

| —  scripts/

| └─ step1_load_inspect.py # Load & inspect dataset

| └─ step2_clean_standardize.py # Clean & standardize records

| └─ step3_add_derived_column.py # Add calculated fields (e.g., yearly salary)


| └─ step4_group_by_department.py # Department-level salary insights

| └─ step5_highest_paid_per_department.py # Identify top earners per department

| └─ step6_group_by_jobtitle.py # Job title-wise analytics

| └─ step7_all_exports.py # End-to-end pipeline execution

|

| —  dashboard/

| └─ simple_dashboard.py # Streamlit interactive dashboard

|

| —  output/

| └─ cleaned_employees.csv # Cleaned dataset

| └─ employees_enriched.csv # Dataset + derived columns

| └─ dept_summary.csv # Departmental insights

| └─ highest_paid.csv # Top paid employees summary

| └─ job_summary.csv # Job title analytics results

|

| —  docs/

| └─ README.md # Project documentation

|

| —  requirements.txt # Python dependencies

Detailed Explanation:

Technical Stack

Core Technologies

- **Python 3.7+:** Main programming language
- **Pandas:** Data manipulation and analysis
- **Streamlit:** Interactive web dashboard framework
- **Matplotlib:** Data visualization
- **NumPy:** Numerical computing

Development Environment

- **VS Code:** Primary IDE
- **Command Line:** Script execution and package management
- **Git:** Version control

Installation & Setup

Prerequisites

- Python 3.13
- pip (Python package installer)

Step 1: Install Required Packages

`pip install pandas streamlit matplotlib numpy seaborn`

Step 2: Project Setup

1. Create project directory: `D:\learning\gsss_sic\mini_project`

2. Place your employees.csv file in the project directory
3. Copy all Python scripts to the project directory
4. Input Data Format

The system expects a CSV file named `employees.csv` with the following structure:

Column	Data Type	Description	Example
EmpID	String	Unique employee identifier	E001, E002
Name	String	Employee full name	Alice Wong
Department	String	Department name	HR, IT, Finance, Sales
JobTitle	String	Employee job title	HR Manager, Software Engineer
Salary	Integer	Annual salary in dollars	60000, 75000
JoiningDate	Date	Employee joining date	2019-03-15

Code Explanation:

Step 1: Data Loading & Inspection

File: `step1_load_inspect.py`

Purpose: Initial data exploration and quality assessment

Operations:

- Load CSV data into pandas DataFrame
- Display dataset information (`info()`, `head()`, `describe()`)
- Identify data types, missing values, and basic statistics

Output: Console display of dataset overview

Step 2: Data Cleaning & Standardization

File: step2_clean_standardize.py

Purpose: Clean and standardize raw data

Operations:

- Remove exact duplicate rows
- Handle duplicate Employee IDs (keep first occurrence)
- Fill missing salary values with median
- Replace missing job titles with "Unknown"
- Standardize department names (HR, IT, Finance, Sales)
- Standardize job title variations

Output: cleaned_employees.csv

Step 3: Data Enrichment

File: step3_add_derived_column.py

Purpose: Add calculated columns for enhanced analysis

Operations:

- Convert JoiningDate to datetime format
- Calculate Years of Service based on current date
- Handle date parsing errors gracefully

Output: employees_enriched.csv

Step 4: Department Analysis

File: step4_group_by_department.py

Purpose: Generate department-wise performance metrics

Operations:

- Group employees by department
- Calculate average salary per department
- Calculate total salary expenditure per department
- Count employees per department

Output: dept_summary.csv

Step 5: Top Performers by Department

File: step5_highest_paid_per_department.py

Purpose: Identify highest-paid employee in each department

Operations:

- Find maximum salary employee per department
- Extract detailed employee information
- Sort results by department

Output: highest_paid.csv

Step 6: Job Title Analysis

File: step6_group_by_jobtitle.py

Purpose: Analyze salary patterns by job title

Operations:

- Group employees by job title
- Calculate average salary per job title
- Calculate total salary cost per job title
- Count employees per job title

Output: job_summary.csv

Step 7: Automated Pipeline Execution

File: step7_all_exports.py

Purpose: Execute entire data processing pipeline

Operations:

- Run steps 2-6 sequentially
- Handle execution errors
- Provide pipeline status updates

Output: All CSV files generated automatically

Step 8: Dashboard Visualization

File: dashboard.py

Purpose: Provide an interactive dashboard for HR to view aggregated salary insights and anomalies

Operations:

- Load processed datasets (payroll_enriched.csv, dept_summary.csv, anomalies.csv)
- Display KPI metrics (total payroll, avg salary, median salary, headcount)
- Visualize charts:
 - Salary distribution (histogram)
 - Average salary by department (bar chart)
 - Headcount by department (bar chart)
- Allow export of filtered views (CSV)

Output:

Interactive dashboard tables, and downloadable reports for HR managers

Code all_exports.py:

```
step7_all_exports.py > ...
1 # Runs steps 2 -> 6 in one go. Assumes employees.csv is present.
2 import subprocess, sys
3 steps = [
4     "step2_clean_standardize.py",
5     "step3_add_derived_column.py",
6     "step4_group_by_department.py",
7     "step5_highest_paid_per_department.py",
8     "step6_group_by_jobtitle.py",
9 ]
10 for s in steps:
11     print(f"\n=== Running {s} ===")
12     ret = subprocess.call([sys.executable, s])
13     if ret != 0:
14         print(f"{s} failed with code {ret}")
15         sys.exit(ret)
16 print("\n✅ All outputs ready: cleaned_employees.csv, employees_enriched.csv, dept_summary.csv, highest_paid.csv, job_summary.csv")
17
```

Output and Explanation:

Output of step7_all_exports.py:

Explanation:

Step 2 cleans and standardizes raw employee data into a consistent format with basic employee information like ID, name, department, job title, salary, and joining date.

Step 3 adds a calculated "Years of Service" column to the cleaned data, showing how long each employee has been with the company.

Step 4 groups employees by department and creates summary statistics showing average salary, total salary costs, and employee count for each department.

Step 5 identifies the highest-paid employee in each department, revealing top earners like Marcus Allen (IT Data Scientist) at \$99,000.

Step 6 analyzes data by job title across all departments, showing salary ranges and employee counts for each position type.

he Finance department has the highest average salary, while IT has the most employees with 15 staff members.

Data Scientists earn the highest average salary at \$97,000, while Sales Executives are the most common job title with 8 employees.

The pipeline generates 5 CSV files providing different analytical views of employee data for HR decision-making and compensation analysis.

```
D:\learning\mini_project>python all_exports.py
python: can't open file 'D:\\learning\\mini_project\\all_exports.py': [Errno 2] No such file or directory

D:\learning\mini_project>python step7_all_exports.py

=== Running step2_clean_standardize.py ===
✅ Cleaned & standardized -> cleaned_employees.csv
EmpID      Name      Department      JobTitle      Salary      JoiningDate
0  E001    Alice Wong      HR             HR Manager    60000        2019-03-15
1  E002    Bob Smith       IT             Software Engineer  75000        2020-06-20
2  E003    Carol Lee       Finance        Accountant     55000        2018-11-12
3  E004    David Kim      IT             Data Scientist 95000        2021-02-10
4  E005    Eva Brown      Sales          Sales Executive 50000        2019-07-01

=== Running step3_add_derived_column.py ===
✅ Added YearsOfService -> employees_enriched.csv
EmpID      Name      Department      JobTitle      JoiningDate      YearsOfService
0  E001    Alice Wong      HR             HR Manager    2019-03-15        6
1  E002    Bob Smith       IT             Software Engineer  2020-06-20        5
2  E003    Carol Lee       Finance        Accountant     2018-11-12        6
3  E004    David Kim      IT             Data Scientist  2021-02-10        4
4  E005    Eva Brown      Sales          Sales Executive  2019-07-01        6

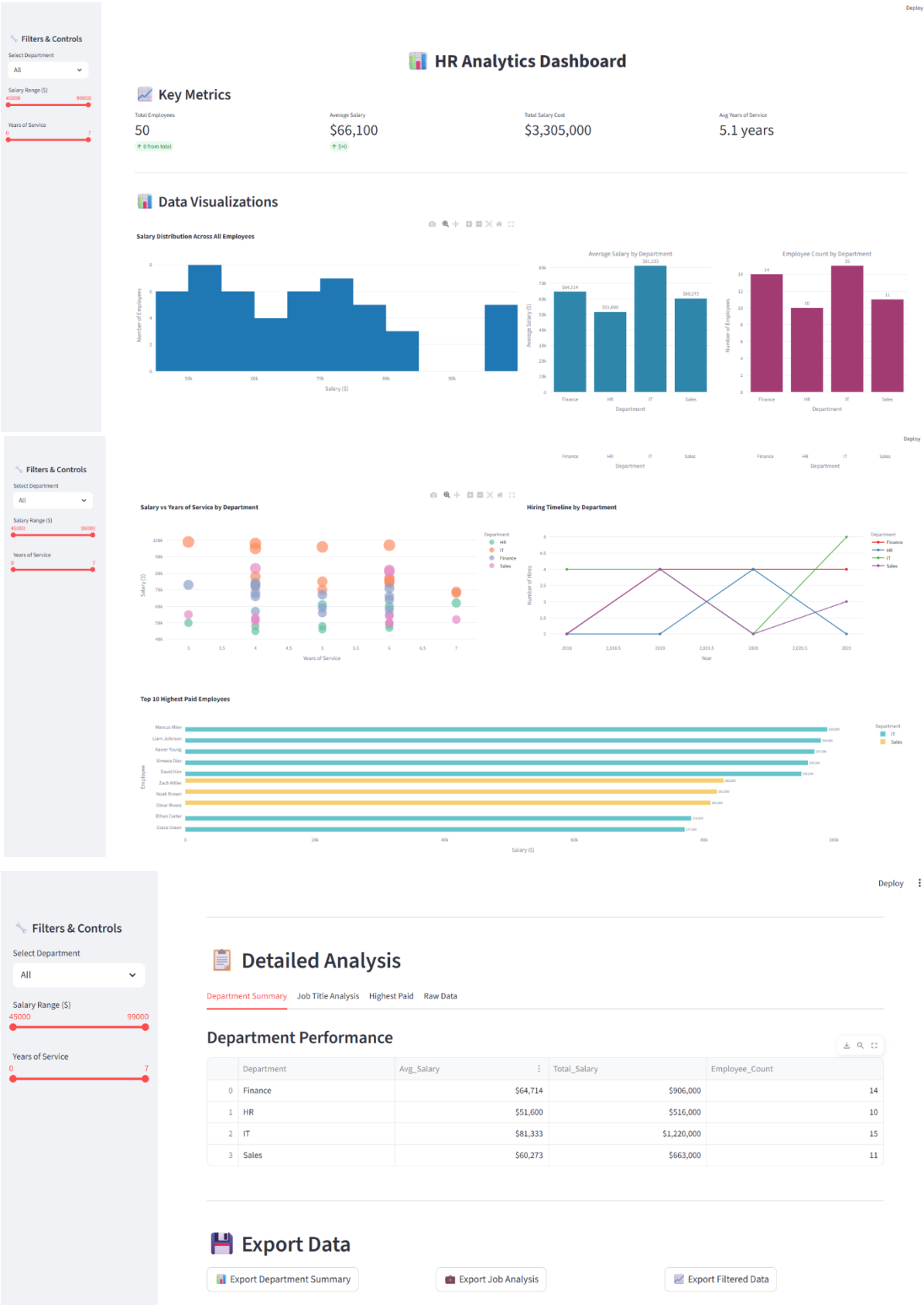
=== Running step4_group_by_department.py ===
✅ Department summary -> dept_summary.csv
Department      Avg_Salary      Total_Salary      Employee_Count
0  Finance        64714.285714     906000           14
1  HR             51600.000000     516000           10
2  IT             81333.333333     1220000          15
3  Sales          60272.727273     663000           11

=== Running step5_highest_paid_per_department.py ===
✅ Highest paid per dept -> highest_paid.csv
Department EmpID      Name      JobTitle      Salary
49  Finance  E050      Yusuf Khan    Senior Accountant  74000
37  HR       E038      Lily Perez    HR Manager         62000
38  IT       E039      Marcus Allen  Data Scientist     99000
25  Sales    E026      Zach Miller    Sales Manager      83000

=== Running step6_group_by_jobtitle.py ===
✅ Job title summary -> job_summary.csv
JobTitle      Avg_Salary      Total_Salary      Employee_Count
1  Data Scientist  97000.000000     485000            5
7  Sales Manager   82000.000000     246000            3
9  Software Engineer  75428.571429     528000            7
8  Senior Accountant  72500.000000     290000            4
10 System Administrator  69000.000000     207000            3
2  Financial Analyst  66200.000000     331000            5
4  HR Manager     61000.000000     183000            3
0  Accountant     57000.000000     285000            5
6  Sales Executive 52125.000000     417000            8
3  HR Executive   49000.000000     147000            3
5  Recruiter      46500.000000     186000            4

✅ All outputs ready: cleaned_employees.csv, employees_enriched.csv, dept_summary.csv, highest_paid.csv, job_summary.csv
```

Output of dashboard.py:



Explanation:

This HR Analytics Dashboard represents the visual output of the employee data processing pipeline, transforming the raw CSV files into an interactive business intelligence interface.

Dashboard Overview The system displays comprehensive HR metrics with a total of 50 employees, an average salary of \$66,100, total salary costs of \$3.305 million, and an average tenure of 5.1 years across the organization.

Interactive Features The left sidebar provides filtering capabilities allowing users to select specific departments and adjust salary ranges (\$45,000-\$99,000) and years of service (1-7 years) to customize their analysis views.

Key Visualizations The first screen shows three primary charts: a salary distribution histogram revealing the spread of compensation across all employees, average salary comparisons by department showing IT leading at \$81,333, and employee count distribution indicating IT as the largest department with 15 employees.

Advanced Analytics The second screen presents correlation analysis between salary and years of service by department, hiring timeline trends from 2018-2021 showing departmental growth patterns, and a top 10 highest-paid employees ranking with Marcus Allen leading at \$99,000.

Detailed Data Tables The third screen provides tabular views of the processed data, including department performance metrics showing Finance with the highest average salary (\$64,714) and total costs, alongside IT having the most employees and highest total salary expenditure (\$1.22 million).

Export Functionality The dashboard includes data export options for Department Summary, Job Analysis, and Filtered Data, enabling users to download specific datasets for further analysis or reporting purposes.

This comprehensive dashboard effectively translates the backend data processing pipeline into actionable HR insights for strategic workforce management and compensation analysis.

Usage Instructions

Running the Data Pipeline

1. Single Step Execution:

```
python step1_load_inspect.py
```

```
python step2_clean_standardize.py
```

continue with other steps

2. Complete Pipeline:

```
python step7_all_exports.py
```

Launching the Dashboard

1. Start the Dashboard:

```
streamlit run dashboard.py
```

2. Access the Application:

- Open browser to <http://localhost:8501>
- Dashboard loads automatically

3. Using the Dashboard:

- Select filters from sidebar
- Explore different visualizations
- Navigate between data tables using tabs
- Download data using export button

Troubleshooting Common Issues

The common issues you can face while running the files are listed here

Data Loading Errors:

Problem: FileNotFoundError: employees.csv not found

Solution: Ensure employees.csv is in the project directory

- Verify file name spelling and extension
- Check file permissions

Missing Dependencies

Problem: ModuleNotFoundError: No module named 'streamlit'

Solution: pip install streamlit pandas matplotlib numpy

Dashboard Import Errors

Problem: ImportError: DLL load failed

Solution: Use the simple dashboard version, Avoid plotly dependencies, Restart terminal

Empty Data Results

Problem: Dashboard shows no data

Solution: Run the complete data pipeline first, Verify CSV files are generated, Check data file paths

Bibliography

- Pandas Documentation: <https://pandas.pydata.org/docs/>
- NumPy Documentation: <https://numpy.org/doc/>
- Matplotlib Documentation:
<https://matplotlib.org/stable/contents.html>
- Plotly Python Graphing Library: <https://plotly.com/python/>
- Streamlit Documentation: <https://docs.streamlit.io/>
- Python Official Documentation: <https://docs.python.org/3/>