

System Requirements

Specification Index

For

Creating a New Column Based on Conditional Statements

(Topic: Advanced Pandas)

Version 1.0

Employee Data Analysis Console

Project Abstract

In this project, the task is to analyze employee data stored in a CSV format. The **EmployeeAnalysis** Python console application is designed to load, process, and update employee salary information. This application reads the employee data from a CSV file, calculates salary levels based on certain criteria, and allows users to export the modified data to a new CSV file. The goal is to provide a tool for quick analysis and categorization of employee salaries, which will help in generating salary insights and reporting for human resource management. This application is crucial for better understanding and managing employee compensation within the organization.

Business Requirements:

- 1. Employee Data Input:**
The system should be capable of accepting employee data in CSV format, where each employee's details include **Name**, **Department**, **Salary**, and other relevant attributes.
The employee data must be processed and classified into categories based on salary.
- 2. Salary Analysis:**
The system should allow for calculating a new column **Salary_Level** that classifies employee salaries as 'High', 'Medium', or 'Low', based on predefined salary thresholds.
- 3. Export Functionality:**
The system must provide the functionality to export the updated employee data with the calculated **Salary_Level** column into a new CSV file.

Constraints

- 1. Input Requirements:**

The employee data file must be in CSV format and should contain at least the following columns:

- **Name** (string): The name of the employee
- **Department** (string): The department of the employee
- **Salary** (float): The annual salary of the employee

2. Output Requirements:

The output must be a CSV file that contains the original data along with a new **Salary_Level** column, classifying each employee into 'High', 'Medium', or 'Low' salary categories.

Process and Operations

1. Data Loading:

- The system must load employee data from a CSV file using the Pandas **read_csv** method.
- If the file is not found or contains invalid data, the system should raise an appropriate error.

2. Salary Classification:

- The **Salary_Level** column should be populated based on the following criteria:
 - 'High' for salaries greater than 60,000
 - 'Medium' for salaries between 30,000 and 60,000
 - 'Low' for salaries less than or equal to 30,000
- The classification should be applied to each employee's salary.

3. Export Process:

- The system should save the updated data (including the

`Salary_Level` column) into a new CSV file. The file must be named as `updated_employee_data.csv` or a user-defined name.

- Ensure the CSV file is saved without the index column.
-

Required Output Format

The system must display the following outputs:

1. **First 5 Rows of Data:**
Show the first 5 rows of the employee data to the user for preview.
 2. **DataFrame Information:**
Display the column names and data types of the employee dataset.
 3. **Salary Level Classification:**
Include the newly calculated `Salary_Level` in the dataset. Each employee's salary should be classified as 'High', 'Medium', or 'Low' based on the salary thresholds.
 4. **Exported CSV File:**
The updated employee data, including the `Salary_Level` column, should be saved to a new CSV file.
-

Template Code Structure:

1. **Data Loading Functions:**
 - `load_employee_data(file_path)`: Load CSV data into a Pandas DataFrame.
 - `display_head()`: Return the first 5 rows of the DataFrame.
2. **Data Processing Functions:**
 - `dataframe_info()`: Display DataFrame column names and

data types.

- **calculate_salary_level()**: Classify salary levels into 'High', 'Medium', and 'Low'.

3. Export Function:

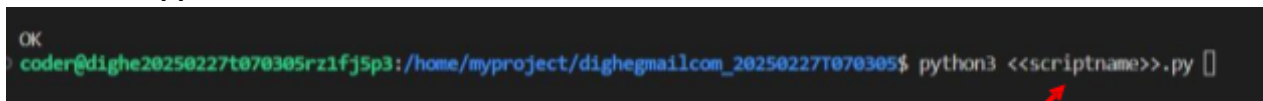
- **export_updated_csv(output_file)**: Save the modified DataFrame with salary classifications to a new CSV file.

Execution Steps to Follow:

- All actions like build, compile, running application, running test cases will be through Command Terminal.
- To open the command terminal the test takers, need to go to Application menu (Three horizontal lines at left top) -> Terminal -> New Terminal
- This editor Auto Saves the code
- If you want to exit(logout) and continue the coding later anytime (using Save & Exit option on Assessment Landing Page) then you need to use **CTRL+Shift+B** -command compulsorily on code IDE. This will push or save the updated contents in the internal git/repository. Else the code will not be available in the next login.
- These are time bound assessments the timer would stop if you logout and while logging in back using the same credentials the timer would resume from the same time it was stopped from the previous logout.
- To setup environment:
You can run the application without importing any packages
- To launch application:
python3 mainclass.py
- To run Test cases:
python3 -m unittest
- Before Final Submission also, you need to use **CTRL+Shift+B** - command compulsorily on code IDE, before final submission as well. This will push or save the updated contents in the internal git/repository, and will be used to evaluate the code quality.

Screen shot to run the program


To run the application

A screenshot of a terminal window with a dark background. The prompt is 'coder@dighe20250227t070305rz1fj5p3: /home/myproject/dighegmailcom_20250227T070305\$'. The command 'python3 <<scriptname>>.py' is entered. A red arrow points from the text 'python3 mainclass.py' below to the '<<scriptname>>.py' part of the command in the terminal.

```
OK
coder@dighe20250227t070305rz1fj5p3: /home/myproject/dighegmailcom_20250227T070305$ python3 <<scriptname>>.py
```

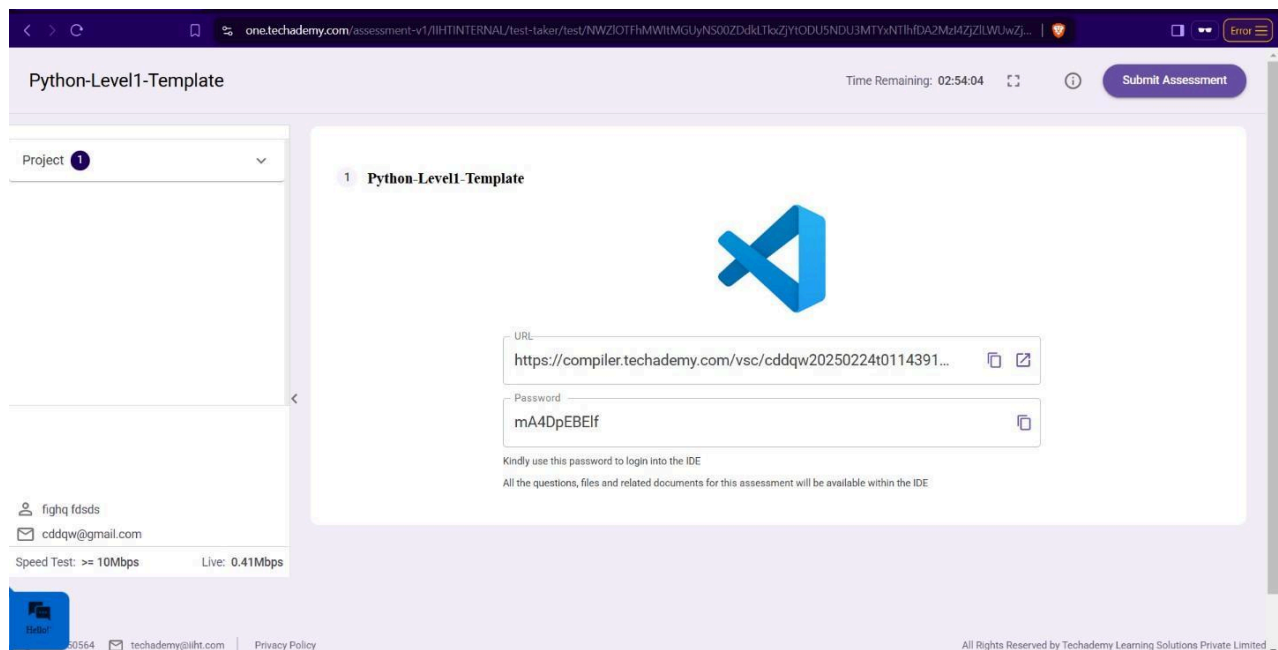
python3 mainclass.py

```
● coder@dighe20250227t070305rz1fj5p3:/home/myproject/dighegmailcom_20250227T070305$ python3 -m unittest
TestBoundary = Passed
.TestExceptional = Passed
.TestCalculateTotalDonations = Failed
.TestCalculateTotalStockValue = Failed
.TestCheckFrankWhiteDonated = Failed
```



To run the testcase

`python3 -m unittest`



- **Once you are done with development and ready with submission, you may navigate to the previous tab and submit the workspace. It is mandatory to click on “Submit Assessment” after you are done with code.**