**Employee Data Management System**

### **Project Description:**

Create a Python program that allows users to manage employee records using a single class (EmployeeManager). This project will reinforce your understanding of core Python concepts, including:

* Data structures (dictionaries, lists)
* File handling using the csv module
* Functions and conditional logic
* Basic command-line interfaces (CLI)

### **📋 Requirements:**

You are required to implement a menu-based employee management system with the following features:

#### **1. Add Employee**

* Collect employee details: ID, Name, Position, Salary, and Email
* Store them in memory using a dictionary
* Save them to a CSV file for future use
  + <https://www.pythontutorial.net/python-basics/python-write-csv-file/>

#### **2. View All Employees**

* List all employees in a readable format (use a loop)
* Data should be loaded from memory

#### 

#### **3. Update Employee**

* Allow user to input the employee ID and update any of the fields (Name, Position, Salary, Email)
* Fields left empty should not be changed
* Save the updated data back to the CSV

#### **4. Delete Employee**

* Allow user to delete a specific employee by ID
* Update the CSV file accordingly

#### **5. Search Employee**

* Search and display an employee’s details by their unique ID

#### **6. Exit**

* Cleanly exit the program

**Technical Requirements:**

* Use a single class called EmployeeManager
* Use a dictionary to store all employee data in memory
* Use the csv module to read/write data from/to employees.csv
* Handle invalid input and ensure basic data validation (e.g., salary is numeric)

### **How It Works:**

1. Start the Program:  
   The user is presented with a menu of actions (add, update, delete, search, list, exit).
2. Perform an Action:  
   Depending on the selected option, the program performs the corresponding task (e.g., adding or updating an employee).
3. Save Data:  
   Changes are saved to a CSV file, ensuring the data is persistent even after the program is closed.
4. Retrieve Data:  
   Employee details are loaded from the CSV file each time the program starts.

### **Grading Criteria for the Project Remark: if use chatGPT you get Zero**

1. **Functionality (50 points)**
   * Menu Options (10 points):  
     Verify that the main menu displays all options (Add, Update, Delete, Search, List, Exit) and correctly accepts user input.
   * Add Employee (10 points):  
     Check if the program successfully adds a new employee and saves the details in the CSV file.
   * Update Employee (10 points):  
     Confirm the program allows users to update specific fields of an employee and reflects the changes correctly.
   * Delete Employee (10 points):  
     Ensure employees can be deleted by their ID, and the CSV file updates correctly.
   * Search Employee (10 points):  
     Validate the search functionality retrieves the correct employee or returns "not found" if the ID doesn’t exist.
2. **Code Quality (20 points)**
   * Readability (5 points):  
     Check for clear variable names, organized code structure, and proper use of comments.
   * Efficiency (5 points):  
     Evaluate if the program avoids unnecessary computations (e.g., iterating only when required).
   * Modularity (5 points):  
     Ensure the code uses functions and methods effectively without redundant logic.
   * Error Handling (5 points):  
     Verify the program handles invalid input gracefully (e.g., invalid ID or non-numeric salary).
3. **Use of OOP Principles (20 points)**
   * Class Design (10 points):  
     Check if EmployeeManager class are designed properly, encapsulating relevant data and logic.
   * Reusability (5 points):  
     Assess if the code can be easily extended (e.g., adding more features without refactoring the entire codebase).
   * Encapsulation & Abstraction (5 points):  
     Confirm if the program uses proper encapsulation (e.g., methods for accessing/updating employee data) and hides unnecessary implementation details.
4. **File Handling (10 points)**
   * CSV Integration (5 points):  
     Ensure the program correctly reads and writes employee data to a CSV file.
   * Data Persistence (5 points):  
     Validate that changes (add, update, delete) are retained across program runs by saving and reloading the file.
5. **Bonus Points (Optional)**

* Validation (5 points):  
  If the program validates fields like email or ensures salary is numeric.
* User Experience (5 points):  
  For adding a clear and user-friendly interface or instructions.

6. **Important Notice – Project Submission (10 point)**

* Please make sure to **submit your project only via Google Classroom**.
* ✅ You are required to submit **only the GitHub repository link**.  
   No need to upload any files.
* 🕒 Submissions outside Google Classroom will **not be accepted**.

### **Sample Grading Table**

| **Criteria** | **Maximum Points** | **Earned Points** | **Comments** |
| --- | --- | --- | --- |
| **Functionality** | **50** |  |  |
| **Code Quality** | **20** |  |  |
| **Use of OOP Principles** | **20** |  |  |
| **File Handling** | **10** |  |  |
| **Bonus** | **10** |  |  |
| **Github** | **10** |  |  |
| **Total** | **110** |  |  |