EMS Group Project

Problem Statement:

You need to create an Employee Management System for a company. This EMS should be able to **add new Employees**, **update Employee information**, **remove Employees**, and **list Employee information**. The user interface (U/I) will be the terminal console, where you can print the program information, and take in inputs from the user's standard input (the keyboard). You will need to have a basic menu with options to navigate your application for your users to interact with. Since this is a group project, be sure to organize your team and design your project development pipeline BEFORE you start writing code. You will be using Git with this project, so be aware that merge conflicts and other issues may arise and plan accordingly. It's recommended that you designate one person to accept pull requests and resolve merge conflicts.

Requirements:

- Use the principles of OOP in your design. I.e. encapsulate relevant code together. Break down methods and classes with separation of concerns (not everything in one big main method), so that each part of your code is focused on doing one task, and doing it well.
- Employees should store the following at minimum: first name, last name, employee id (needs to be unique), Date of Employment, salary, and department as a String (Bonus below for Department as its own class).
- Use the file methods to store and retrieve (read and write) your data for Employees and Departments to a file format of your choice (CSV, JSON, Other)
- Implement the use of at least one custom exception, and use exception handling in all appropriate places.
- Use at least one List Comprehension.

- Have a console based U/I menu system that allows a user to navigate through your app and perform all of the CRUD operations listed above, at a minimum.
- Use Git and Github for version control. Use Git best practices to make coordinating with teammates easier. Each team member should have their own branch and submit pull requests to merge onto the main branch.

Bonus:

- Create a Department class and make sure to store information a department might have (name, budget, phone, etc.). You will also be able to **add, update, remove, and list the different departments**. You should also be able to ***keep track of all the departments in the company and which employees are within them***.
- Set up testing for code using unittest.