This project is an **extension** of your first practical project, but now it requires more structure and additional features. Here’s a simple breakdown of what you **need to do**:

**1. Improve Your Project Structure (N-Layered or MVC)**

* Instead of having all your logic in one or two files, you need to **separate your code into different layers**:
  + **Presentation Layer** (User Interface) – Handles user interactions (menu, input, output).
  + **Business Layer** – Manages how data is processed in memory (stores and modifies records).
  + **Persistence Layer** – Handles reading and writing data from/to a CSV file.
  + **Model (Entity) Layer** – Defines the data structure (your existing TrafficDataRecord class).
* **Create separate folders** for these layers and move the respective Python files into them.

**2. Extend Functionality**

You need to add **new features** to your program:

1. **Read & Store First 100 Records** – When the program starts, read up to 100 records from the CSV and store them in memory (list/array).
2. **Display Name on Screen** – Your name should always be visible or appear every 10 records when displaying multiple records.
3. **Interactive Menu for User Input**:
   * **Reload Data**: Read from the CSV again and replace in-memory data.
   * **Save Data to a New CSV**: Save current in-memory data to a new CSV file with a unique name (use a GUID/UUID).
   * **Search & Display Records**: Let the user pick and display one or multiple records.
   * **Add a New Record**: Allow the user to input new data and add it to the in-memory list.
   * **Edit a Record**: Modify an existing record in the list.
   * **Delete a Record**: Remove a record from the list.

**3. Use GitHub for Version Control**

* **Commit & Tag your progress step by step** (e.g., one commit for each major feature).
* Use **tag “v2.0”** for this version.

**4. Implement Unit Testing**

* Write a **single unit test** to check if at least **one function works correctly**.
  + Example: Verify if a record is **correctly added, edited, or deleted**.

**5. Follow Best Practices**

* **Use exception handling** (e.g., handle missing files, incorrect data input).
* **Write docstrings/comments** in all Python files.
* **Structure your files into folders** for clarity.

**6. Provide Screenshots & Documentation**

* Take **screenshots of your program in action** (reading CSV, user interactions, saving files, etc.).
* Document your learning experience and describe how your program runs.

**Summary of What You Need to Do:**

✅ **Reorganize** your project into structured layers (folders).  
✅ **Modify** your existing code to handle up to **100 records** in memory.  
✅ **Add user interaction options** (reload, save, search, add, edit, delete).  
✅ **Use exception handling** for file errors and invalid input.  
✅ **Write one unit test** to confirm functionality.  
✅ **Commit & tag** each major change in GitHub.  
✅ **Take screenshots** of your program running & document your process.

This project is **more structured and interactive**, but since you already built a Python program in **Practical Project 1**, you just need to **expand and organize** it. Start step by step, and I’ll help you along the way! 🚀