**Name Manager System Project**

**Objective**: Build a modular Python application that allows users to manage (add, view, search, and delete) names in a persistent .dat file using object-oriented programming principles. Photos are to be included in file submission.

**Skills Covered**

| **Skill** | **Description** |
| --- | --- |
| **Basic Syntax** | Variables, loops, conditionals |
| **Functions** | Modular code organization |
| **File I/O** | Read/write binary files. Bonus if using pickle or shelve |
| **OOP** | Classes, inheritance, encapsulation, polymorphism |
| **Exception Handling** | Robust error catching |
| **Multiple Files** | Modular structure using packages or simple modules |
| **CLI Interaction** | Input/output in the terminal |
| **Data Validation** | Ensuring valid names only |

**Project Overview**

Create a command-line tool for managing a database of names. The program supports:

1. Adding a name
2. Viewing all names
3. Searching for a name
4. Deleting a name
5. Saving and loading names from a .dat file

**Pickle Code Example**

import pickle

# Suppose you have a list of Person objects

people = [Person("Alice", "Smith"), Person("Bob", "Jones")]

# Saving to a file (serialization)

with open('names.dat', 'wb') as file:

pickle.dump(people, file)

# Loading from a file (deserialization)

with open('names.dat', 'rb') as file:

loaded\_people = pickle.load(file)

**Shelve Code Example**

import shelve

# Suppose you have a list of Person objects

people = [Person("Alice", "Smith"), Person("Bob", "Jones")]

# Saving to shelve file

with shelve.open('names\_shelve') as db:

db['people'] = people # store the list under the key 'people'

# Loading from shelve file

with shelve.open('names\_shelve') as db:

loaded\_people = db['people']

**How to Run the Program**

1. Save all files in the same name\_manager folder.
2. Open terminal or command prompt.
3. Navigate to the folder.
4. Run: python main.py

**Submission**

* GitHub Link with the following:
  + Code
  + README.md
* README.md (on GitHub):
  + Project title and description
  + Setup instructions
  + How to run/demo the project
* Photos
  + Screenshots of your application in action
  + Optional: photos of setup/hardware (if applicable)
* Documentation (typically a PDF or DOCX):
  + Project overview and goals
  + System architecture / design diagrams
  + Implementation details (including technologies used)
  + The IDE and how to run the code
  + Testing strategy and results – including testing units
  + Any challenges faced and how they were solved
  + Future work or improvements