Intern Project Assignments (Python + Tkinter)

General Rules (Mandatory for All Projects)

• Modular Architecture: You must have at least three files: main.py (GUI), logic.py (Data Processing), and models.py (OOP Classes).

• No-Crash Policy: Every user input must be validated. Invalid inputs must show a Tkinter messagebox instead of crashing.

• Data Persistence: Use CSV or JSON files so data is preserved after restarting the application.

• Documentation: Every function must include a proper docstring explaining its purpose.

Project 1: Aegis – IT Asset Management System

Goal

Build a professional desktop application to track company hardware throughout its lifecycle.

Problem Statement

Nexus Corp is losing track of its laptops and monitors. They need a system to register hardware, assign it to employees, and track availability.

Functional Requirements

• Create an Asset class with attributes: Asset_ID, Type, Brand, Purchase_Date, Status.

• Ensure Asset_ID is unique; block duplicate entries.

• Prevent assigning an already assigned asset.

• Load and save asset data using a CSV file.

GUI Requirements (Tkinter)

• Treeview widget for displaying assets.

• Buttons: Add Asset, Assign to Employee, Delete Retired Asset.

• Real-time search bar to filter assets.

Project 2: PulseCare – Clinic Appointment Manager

Goal

Manage patient queues efficiently using smart data structures.

Problem Statement

City Health Clinic has multiple doctors but one waiting room. Patients need clarity on their turn based on urgency and arrival time.

Functional Requirements

• Use a dictionary mapping Doctor Name to a list of Patient objects.

• Emergency patients must be inserted at the front of the queue.

• Track patient status: Waiting, With Doctor, Completed.

• Patient class must be defined in models.py.

GUI Requirements (Tkinter)

• Combobox to select doctor.

• Next Patient button to move patients to history.

• Emergency patients displayed in bold red text.

Project 3: StockStream – Retail Inventory & Billing

Goal

Create a POS system handling inventory, billing, and taxation.

Problem Statement

Urban Closet needs a system to calculate bills, apply tax, and automatically update inventory after each sale.

Functional Requirements

• Use a dictionary to manage inventory with product details.

• Check stock availability before selling.

• Apply 12% GST on the total bill.

• Disable Add to Cart if stock drops below 5 units.

GUI Requirements (Tkinter)

• Active Cart listbox.

• Checkout button showing final amount with tax.

• Generate a receipt text file with timestamp.

Project 4: EduTrack – Student Performance Analytics

Goal

Convert student marks into actionable academic insights.

Problem Statement

Greenwood Academy needs a system to identify weak students and top performers using analytical logic.

Functional Requirements

• Store student data using nested dictionaries.

• Calculate average marks and grades.

• Identify class toppers.

• Handle invalid marks using try-except blocks.

GUI Requirements (Tkinter)

• Dashboard showing total students, class average, topper.

• Filter button for students scoring below 40%.

• Optional bar graph using Canvas.

NOTE:

If I or other interns look at your code and can't understand it in 60 seconds, you have failed the assignment—even if the app runs perfectly.

ADDITIONAL INSTRUCTIONS:

must use Git.

One student will create a Repository and will invite their partner and **me** as collaborators.

The Commit History: I will check the commit history. If I see only one big commit at the end called "project finished," they lose points. I want to see "Initial UI setup," "Fixed bug in logic.py," etc.

Add the README.md file:

Project Title & Team Members.

How to Run: (e.g., python main.py).

Key Features: A bulleted list of what the app can do.

The "Logic Highlights": A short paragraph explaining how they handled a specific challenge (like the "Priority Queue" or "Low Stock Alert").