**Day 05 Task Allocation: -**

**To create a simple Asset Inventory Management System in Python that interacts with a PostgreSQL database, I'll provide you with a basic outline and code. The system will include a dashboard interface using Tkinter for the first page and a PostgreSQL connection to manage the asset data.**

**1. Install Required Libraries**

**Before you start, make sure to install the necessary libraries:**

**Source Code: -**

**pip install psycopg2 tkinter**

### 2. Database Setup

First, you'll need to create a PostgreSQL database and table for storing asset information.

**SQL to create a table:**

**Source Code: -**

**CREATE TABLE assets (**

**id SERIAL PRIMARY KEY,**

**asset\_name VARCHAR(100) NOT NULL,**

**asset\_type VARCHAR(50),**

**purchase\_date DATE,**

**asset\_value NUMERIC,**

**asset\_status VARCHAR(20)**

**);**

**3. Python Code for Asset Management System: -**

**import tkinter as tk**

**from tkinter import messagebox**

**import psycopg2**

**from psycopg2 import sql**

**# PostgreSQL connection**

**def connect\_db():**

**try:**

**conn = psycopg2.connect(**

**dbname="your\_database\_name",**

**user="your\_username",**

**password="your\_password",**

**host="localhost",**

**port="5432"**

**)**

**return conn**

**except Exception as e:**

**messagebox.showerror("Database Error", f"Error connecting to database: {e}")**

**return None**

**# Function to insert data into the database**

**def insert\_asset(asset\_name, asset\_type, purchase\_date, asset\_value, asset\_status):**

**conn = connect\_db()**

**if conn:**

**cursor = conn.cursor()**

**query = sql.SQL("INSERT INTO assets (asset\_name, asset\_type, purchase\_date, asset\_value, asset\_status) VALUES (%s, %s, %s, %s, %s)")**

**try:**

**cursor.execute(query, (asset\_name, asset\_type, purchase\_date, asset\_value, asset\_status))**

**conn.commit()**

**messagebox.showinfo("Success", "Asset added successfully!")**

**except Exception as e:**

**messagebox.showerror("Error", f"Failed to add asset: {e}")**

**finally:**

**cursor.close()**

**conn.close()**

**# Function to show the dashboard**

**def show\_dashboard():**

**def add\_asset():**

**asset\_name = asset\_name\_entry.get()**

**asset\_type = asset\_type\_entry.get()**

**purchase\_date = purchase\_date\_entry.get()**

**asset\_value = asset\_value\_entry.get()**

**asset\_status = asset\_status\_entry.get()**

**if not all([asset\_name, asset\_type, purchase\_date, asset\_value, asset\_status]):**

**messagebox.showwarning("Input Error", "Please fill out all fields.")**

**return**

**insert\_asset(asset\_name, asset\_type, purchase\_date, asset\_value, asset\_status)**

**dashboard = tk.Tk()**

**dashboard.title("Asset Management Dashboard")**

**tk.Label(dashboard, text="Asset Name").grid(row=0, column=0, padx=10, pady=10)**

**asset\_name\_entry = tk.Entry(dashboard)**

**asset\_name\_entry.grid(row=0, column=1, padx=10, pady=10)**

**tk.Label(dashboard, text="Asset Type").grid(row=1, column=0, padx=10, pady=10)**

**asset\_type\_entry = tk.Entry(dashboard)**

**asset\_type\_entry.grid(row=1, column=1, padx=10, pady=10)**

**tk.Label(dashboard, text="Purchase Date (YYYY-MM-DD)").grid(row=2, column=0, padx=10, pady=10)**

**purchase\_date\_entry = tk.Entry(dashboard)**

**purchase\_date\_entry.grid(row=2, column=1, padx=10, pady=10)**

**tk.Label(dashboard, text="Asset Value").grid(row=3, column=0, padx=10, pady=10)**

**asset\_value\_entry = tk.Entry(dashboard)**

**asset\_value\_entry.grid(row=3, column=1, padx=10, pady=10)**

**tk.Label(dashboard, text="Asset Status").grid(row=4, column=0, padx=10, pady=10)**

**asset\_status\_entry = tk.Entry(dashboard)**

**asset\_status\_entry.grid(row=4, column=1, padx=10, pady=10)**

**tk.Button(dashboard, text="Add Asset", command=add\_asset).grid(row=5, column=0, columnspan=2, pady=20)**

**dashboard.mainloop()**

**# Run the dashboard**

**show\_dashboard()**

### 4. Explanation

* **Database Connection (connect\_db)**: Establishes a connection with the PostgreSQL database.
* **Insert Asset (insert\_asset)**: Takes asset details as input and inserts them into the assets table.
* **Dashboard (show\_dashboard)**: Displays a simple interface using Tkinter where users can input asset details.

### 5. Running the Code

When you run this script, a simple GUI window will pop up where you can enter the asset details. After entering the details, clicking "Add Asset" will store the data in the PostgreSQL database.

**Note:** Replace your\_database\_name, your\_username, and your\_password with your actual PostgreSQL credentials.

This is a basic starting point. You can expand this by adding more features such as viewing, editing, and deleting assets, or by improving the user interface.