**Simple asset inventory management system in Python requires defining a basic structure to store and manage the assets. Below is a Python script that can be used to manage an inventory of assets, allowing the user to add, view, update, and delete assets.**

**Source Code: -**

import json

# Asset Inventory Management System

class AssetInventory:

def \_\_init\_\_(self):

self.assets = []

def add\_asset(self, asset\_id, asset\_name, asset\_type, purchase\_date, value, owner):

asset = {

'Asset ID': asset\_id,

'Asset Name': asset\_name,

'Asset Type': asset\_type,

'Purchase Date': purchase\_date,

'Value': value,

'Owner': owner

}

self.assets.append(asset)

print(f"Asset '{asset\_name}' added successfully.")

def view\_assets(self):

if not self.assets:

print("No assets in the inventory.")

return

for asset in self.assets:

print(json.dumps(asset, indent=4))

def update\_asset(self, asset\_id, \*\*kwargs):

for asset in self.assets:

if asset['Asset ID'] == asset\_id:

for key, value in kwargs.items():

if key in asset:

asset[key] = value

print(f"Asset ID '{asset\_id}' updated successfully.")

return

print(f"Asset ID '{asset\_id}' not found.")

def delete\_asset(self, asset\_id):

for asset in self.assets:

if asset['Asset ID'] == asset\_id:

self.assets.remove(asset)

print(f"Asset ID '{asset\_id}' deleted successfully.")

return

print(f"Asset ID '{asset\_id}' not found.")

# Main function

def main():

inventory = AssetInventory()

while True:

print("\nAsset Inventory Management System")

print("1. Add Asset")

print("2. View Assets")

print("3. Update Asset")

print("4. Delete Asset")

print("5. Exit")

choice = input("Enter your choice: ")

if choice == '1':

asset\_id = input("Enter Asset ID: ")

asset\_name = input("Enter Asset Name: ")

asset\_type = input("Enter Asset Type: ")

purchase\_date = input("Enter Purchase Date (YYYY-MM-DD): ")

value = input("Enter Asset Value: ")

owner = input("Enter Owner: ")

inventory.add\_asset(asset\_id, asset\_name, asset\_type, purchase\_date, value, owner)

elif choice == '2':

inventory.view\_assets()

elif choice == '3':

asset\_id = input("Enter Asset ID to update: ")

print("Enter new details (leave blank if no change):")

asset\_name = input("Enter new Asset Name: ")

asset\_type = input("Enter new Asset Type: ")

purchase\_date = input("Enter new Purchase Date (YYYY-MM-DD): ")

value = input("Enter new Asset Value: ")

owner = input("Enter new Owner: ")

update\_data = {

'Asset Name': asset\_name,

'Asset Type': asset\_type,

'Purchase Date': purchase\_date,

'Value': value,

'Owner': owner

}

# Filter out empty inputs

update\_data = {k: v for k, v in update\_data.items() if v}

inventory.update\_asset(asset\_id, \*\*update\_data)

elif choice == '4':

asset\_id = input("Enter Asset ID to delete: ")

inventory.delete\_asset(asset\_id)

elif choice == '5':

print("Exiting the system.")

break

else:

print("Invalid choice. Please try again.")

if \_\_name\_\_ == "\_\_main\_\_":

main()

**Explanation:**

1. **Asset Inventory Class (AssetInventory)**:
   * **Attributes**: Stores assets as a list of dictionaries.
   * **Methods**:
     + add\_asset: Adds a new asset with the required details.
     + view\_assets: Displays all assets in the inventory.
     + update\_asset: Updates details of a specific asset.
     + delete\_asset: Removes an asset from the inventory by its ID.
2. **Main Function (main)**:
   * Provides a menu-driven interface to add, view, update, or delete assets.
3. **Usage**:
   * Run the script and follow the on-screen prompts to manage assets.

This script is a basic implementation. Depending on your needs, you can extend it to include features like data persistence (e.g., saving to a file or database), input validation, and more complex reporting.