Internship project

-june 2025

Asset managment

Code

include <iostream>

#include <cstring>

using namespace std;

const int MAX\_ASSETS = 100;

struct Employee

{

int id;

char name[50];

char department[50];

};

struct Location

{

int id;

char name[50];

char address[100];

};

struct Depreciation

{

float rate;

int usefulLife;

};

struct Maintenance

{

int id;

char description[100];

char scheduledDate[20];

bool completed;

};

struct Asset

{

int id;

char name[50];

char description[100];

char acquisitionDate[20];

float purchaseCost;

char status[30];

Depreciation depreciation;

Maintenance maintenance;

Location location;

Employee assignedTo;

bool isAssigned;

};

Asset assets[MAX\_ASSETS];

int assetCount = 0;

void addAsset()

{

if (assetCount >= MAX\_ASSETS) {

cout << "Asset limit reached!\n";

return;

}

Asset& a = assets[assetCount];

cout << "Enter Asset ID: ";

cin >> a.id;

cin.ignore();

cout << "Enter Name: ";

cin.getline(a.name, 50);

cout << "Enter Description: ";

cin.getline(a.description, 100);

cout << "Enter Acquisition Date (dd-mm-yyyy): ";

cin.getline(a.acquisitionDate, 20);

cout << "Enter Purchase Cost: ";

cin >> a.purchaseCost;

cin.ignore();

cout << "Enter Status: ";

cin.getline(a.status, 30);

cout << "Enter Depreciation Rate (%): ";

cin >> a.depreciation.rate;

cout << "Enter Useful Life (years): ";

cin >> a.depreciation.usefulLife;

cout << "Enter Maintenance ID: ";

cin >> a.maintenance.id;

cin.ignore();

cout << "Enter Maintenance Description: ";

cin.getline(a.maintenance.description, 100);

cout << "Enter Scheduled Date: ";

cin.getline(a.maintenance.scheduledDate, 20);

a.maintenance.completed = false;

cout << "Enter Location ID: ";

cin >> a.location.id;

cin.ignore();

cout << "Enter Location Name: ";

cin.getline(a.location.name, 50);

cout << "Enter Address: ";

cin.getline(a.location.address, 100);

a.isAssigned = false;

assetCount++;

cout << "Asset added successfully!\n";

}

void assignAssetToEmployee() {

int id;

cout << "Enter Asset ID to assign: ";

cin >> id;

for (int i = 0; i < assetCount; i++)

{

if (assets[i].id == id)

{

cout << "Enter Employee ID: ";

cin >> assets[i].assignedTo.id;

cin.ignore();

cout << "Enter Employee Name: ";

cin.getline(assets[i].assignedTo.name, 50);

cout << "Enter Department: ";

cin.getline(assets[i].assignedTo.department, 50);

assets[i].isAssigned = true;

cout << "Asset assigned successfully.\n";

return;

}

}

cout << "Asset not found.\n";

}

void updateAssetStatus()

{

int id;

char newStatus[30];

cout << "Enter Asset ID: ";

cin >> id;

cin.ignore();

cout << "Enter New Status: ";

cin.getline(newStatus, 30);

for (int i = 0; i < assetCount; i++)

{

if (assets[i].id == id)

{

strcpy(assets[i].status, newStatus);

cout << "Status updated.\n";

return;

}

}

cout << "Asset not found.\n";

}

float calculateDepreciation(Asset a) {

return a.purchaseCost - (a.purchaseCost \* (a.depreciation.rate / 100) \* a.depreciation.usefulLife);

}

void removeAsset() {

int id;

cout << "Enter Asset ID to remove: ";

cin >> id;

for (int i = 0; i < assetCount; i++)

{

if (assets[i].id == id)

{

for (int j = i; j < assetCount - 1; j++)

{

assets[j] = assets[j + 1];

}

assetCount--;

cout << "Asset removed.\n";

return;

}

}

cout << "Asset not found.\n";

}

void displayAllAssets() {

for (int i = 0; i < assetCount; i++) {

Asset& a = assets[i];

cout << "\nAsset ID: " << a.id

<< "\nName: " << a.name

<< "\nStatus: " << a.status

<< "\nDepreciation Value: " << calculateDepreciation(a)

<< "\nAssigned: " << (a.isAssigned ? "Yes" : "No") << "\n";

}

}

int main() {

int choice;

do {

cout << "\nAsset Management Menu\n";

cout << "1. Add Asset\n";

cout << "2. Assign Asset\n";

cout << "3. Update Status\n";

cout << "4. Remove Asset\n";

cout << "5. Show All Assets\n";

cout << "6. Exit\n";

cout << "Enter your choice: ";

cin >> choice;

switch (choice) {

case 1: addAsset();

break;

case 2: assignAssetToEmployee();

break;

case 3: updateAssetStatus();

break;

case 4: removeAsset();

break;

case 5: displayAllAssets();

break;

case 6: cout << "Exiting...\n";

break;

default:

cout << "Invalid choice!\n";

}

} while (choice != 6);

return 0;

}

