Case Study: Digital Asset Management Application Documentation

~ P325_Sarvesh Sharma

Project Overview:

- **Purpose:** The Digital Asset Management System (DAMS) is designed to manage company assets efficiently, track asset allocation, maintenance, and reservations, and maintain records associated with assets and employees.
- **Technologies Used:** Python, MSSQL

Directory Structure: S:\GIT\Hexawarwe_CaseStudy

- **main:** Contains the main execution module **`AssetManagementApp.py`**
- **dao:** Data access object classes for interacting with the database (e.g., **`AssetManagementService.py`**, **`AssetManagementServiceImpl.py`**)
- **entity:** Classes representing the tables defined in SQL (e.g., **`Asset.py`**, **`Employee.py`**, **`MaintenanceRecord.py`**, **`AssetAllocation.py`**, **`Reservation.py`**)
- **myexceptions:** Custom exceptions (e.g., **`AssetNotFoundException.py`**, **`AssetNotMaintainException.py`**)
- **util:** Contains utility classes (e.g., `DBConnection.py` for database connection)
- **SQL File: `Case_Study.sql`** for setting up the database schema

```
S:\GIT\Hexaware_CaseStudy\

├── Case_Study.sql            # SQL file with database schema

├── config.properties         # Configuration file

├── venv                      # Virtual Environment

│

├── entity                    # Entity package for business objects

│   ├── __init__.py
│   ├── Employee.py                # Employee class
│   ├── Asset.py                   # Asset class
│   ├── MaintenanceRecord.py      # MaintenanceRecord class
│   ├── AssetAllocation.py        # AssetAllocation class
│   └── Reservation.py             # Reservation class
│
├── dao                                    # DAO package for database interaction
│   ├── __init__.py
│   ├── AssetManagementService.py          # Service interface
│   └── AssetManagementServiceImpl.py    # Implementation of service interface
│
├── myexceptions                        # Custom exception handling package
│   ├── __init__.py
│   ├── AssetNotFoundException.py       # Exception for asset not found
│   ├── AssetNotMaintainException.py    # Exception for assets not maintainable
│
├── util                           # Utility package for database connection
│   ├── __init__.py
│   └── DBConnection.py            # Database connection management
│
└── app                              # Application package (main application logic)
   ├── __init__.py
   └── AssetManagementApp.py       # Main application logic with menu
```

Project Flow:

1. Database Setup

- **SQL File**: Created an SQL file **`Case_Study.sql`** to define the database schema for the Digital Asset Management system.
- **Tables**:
    - **`Assets`**: Stores information about different assets, including AssetID, Name, Type, SerialNumber, PurchaseDate, Location, Status, and OwnerID.
    - **`Employees`**: Stores employee details, including EmployeeID, Name, Role, and Salary.
    - **`Asset_Allocations`**: Tracks the allocation of assets to employees, including AllocationID, AllocationDate, ReturnDate, AssetID, and EmployeeID.
    - **`Maintenance_Records`**: Logs maintenance activities for assets, including MaintenanceID, MaintenanceDate, AssetID, and Notes.
    - **`Reservations`**: Handles reservations made for assets, including ReservationID, ReservationDate, StartDate, EndDate, AssetID, and EmployeeID.

2. Entity Classes

- Defined classes in the `entity` package:
    - **`Asset`**: Attributes include AssetID, Name, Type, SerialNumber, PurchaseDate, Location, Status, and OwnerID.
    - **`Employee`**: Attributes include EmployeeID, Name, Role, and Salary.
    - **`AssetAllocation`**: Tracks when assets are allocated or deallocated, with attributes like AllocationID, AssetID, EmployeeID, AllocationDate, and ReturnDate.
    - **`MaintenanceRecord`**: Stores maintenance activities, with attributes like MaintenanceID, AssetID, MaintenanceDate, and Notes.
    - **`Reservation`**: Manages reservations, with attributes like ReservationID, ReservationDate, StartDate, EndDate, AssetID, and EmployeeID.

3. DAO Implementation

- **Interfaces and Implementation**:
    - Developed the interface `AssetManagementService` and its implementation `AssetManagementServiceImpl`.
    - Methods implemented include:
        - `addAsset()`: Add new assets to the system.
        - `updateAsset()`: Modify asset details.
        - `deleteAsset()`: Remove an asset from the system.
        - `allocateAsset()`: Assign an asset to an employee.

        - `deallocateAsset()`: Deallocate an asset and return it to the available pool.
        - `performMaintenance()`: Log maintenance activities for an asset.
        - `reserveAsset()`: Reserve an asset for future use.
        - `withdrawReservation()`: Cancel a reservation for an asset.

4. Utility Class

- **`DBConnection.py`**:
    - Manages connections to the SQL database.
    - Uses connection pooling to handle multiple database requests.
    - Handles connection setup and teardown with methods like `get_connection()` and `close_connection()`.

5. Exception Handling

- Custom exceptions implemented in the `myexceptions` package:
    - **`AssetNotFoundException`**: Raised when an asset ID does not exist in the system.
    - **`AssetNotMaintainException`**: Raised when an asset is not eligible for maintenance (e.g., currently in use or in repair).

6. Main Application Logic

- Developed a **menu-driven interface** in **`AssetManagementApp.py`** that allows users to:
    - Add, update, and delete assets.
    - Allocate or deallocate assets to/from employees.
    - Perform maintenance on assets.
    - Make and withdraw reservations.
    - Retrieve details for assets, allocations, maintenance records, and reservations.

7. Sample Data

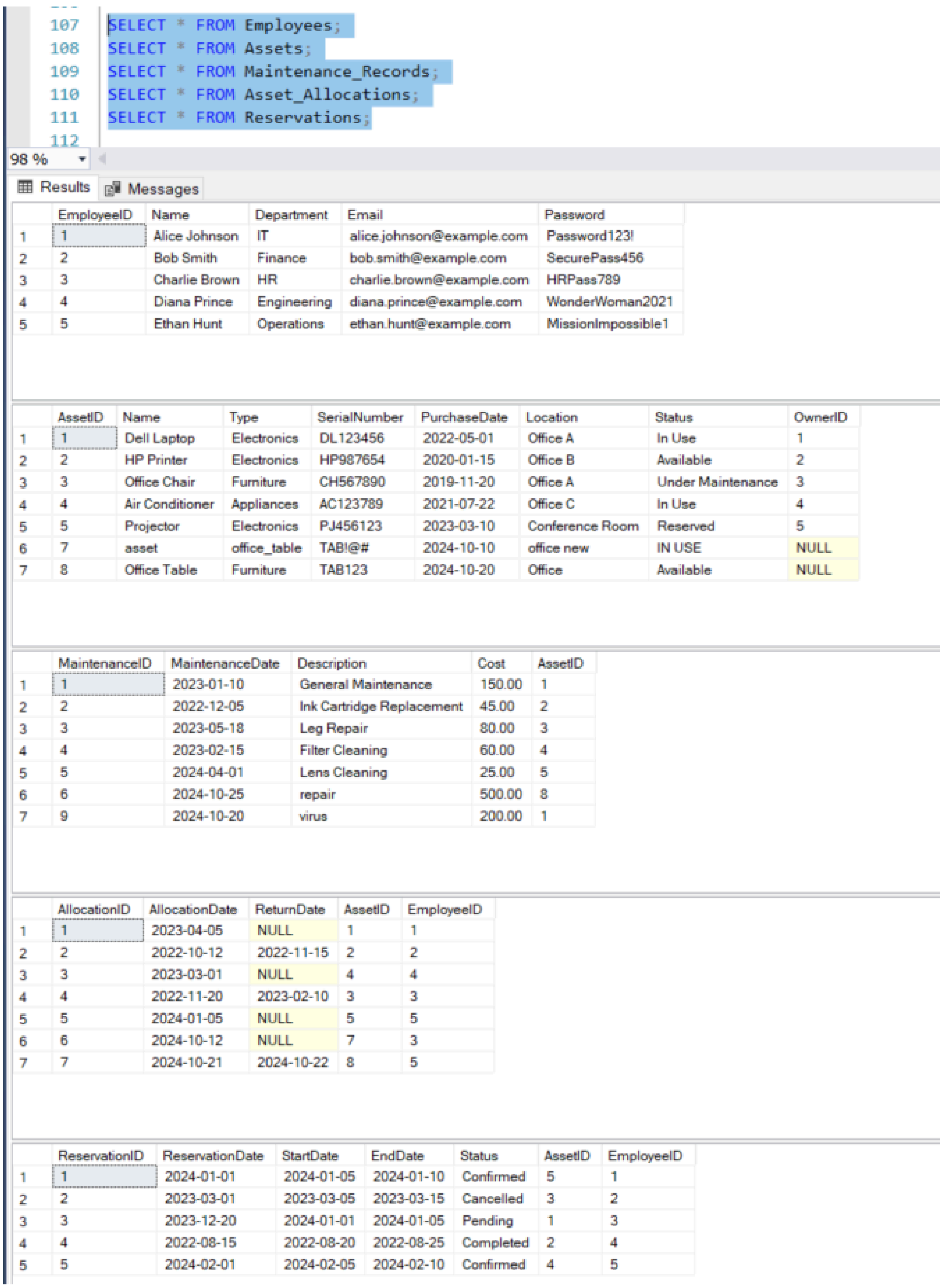
- Inserted sample data into the database for testing purposes:
    - Preloaded assets, employees, and some initial reservations and allocations.
    - Sample queries for adding assets, scheduling maintenance, and reserving assets.

8. Testing and Validation

- **Unit Testing**: Performed unit tests on service methods to ensure they handle all possible scenarios, including valid and invalid inputs.

Outputs:

=> Initial table and their values



```
SELECT * FROM Employees;
SELECT * FROM Assets;
SELECT * FROM Maintenance_Records;
SELECT * FROM Asset_Allocations;
SELECT * FROM Reservations;
```

98 %

Results | Messages

	EmployeeID	Name	Department	Email	Password
1	1	Alice Johnson	IT	alice.johnson@example.com	Password123!
2	2	Bob Smith	Finance	bob.smith@example.com	SecurePass456
3	3	Charlie Brown	HR	charlie.brown@example.com	HRPass789
4	4	Diana Prince	Engineering	diana.prince@example.com	WonderWoman2021
5	5	Ethan Hunt	Operations	ethan.hunt@example.com	MissionImpossible1

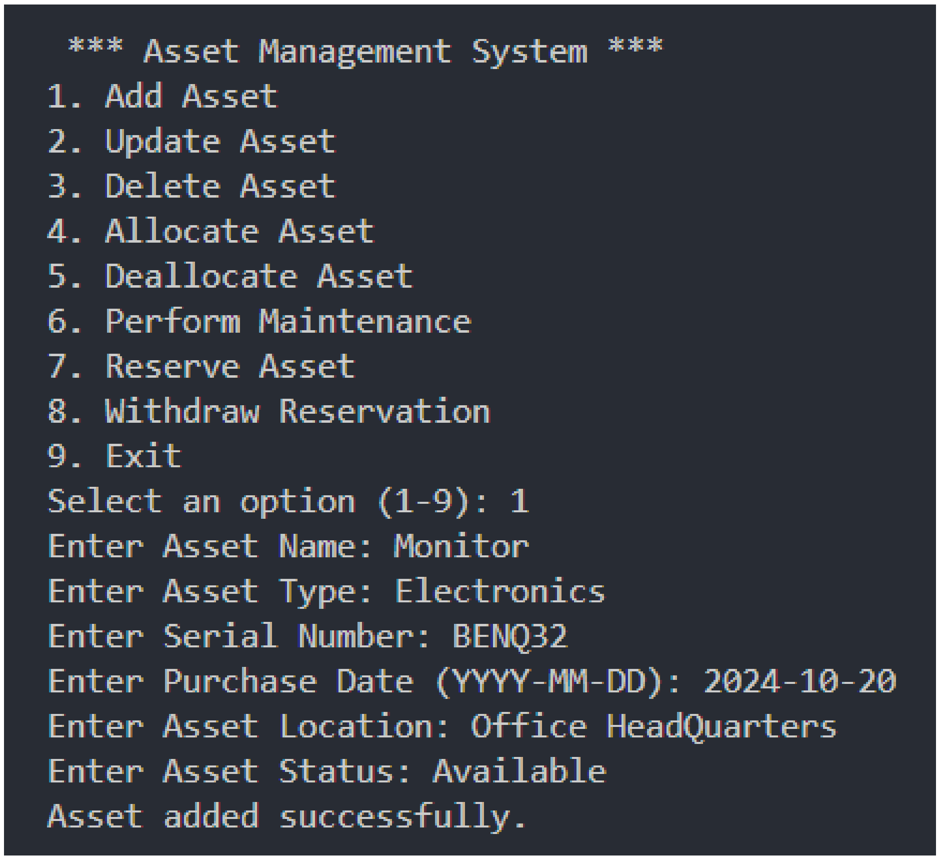
	AssetID	Name	Type	SerialNumber	PurchaseDate	Location	Status	OwnerID
1	1	Dell Laptop	Electronics	DL123456	2022-05-01	Office A	In Use	1
2	2	HP Printer	Electronics	HP987654	2020-01-15	Office B	Available	2
3	3	Office Chair	Furniture	CH567890	2019-11-20	Office A	Under Maintenance	3
4	4	Air Conditioner	Appliances	AC123789	2021-07-22	Office C	In Use	4
5	5	Projector	Electronics	PJ456123	2023-03-10	Conference Room	Reserved	5
6	7	asset	office_table	TAB!@#	2024-10-10	office new	IN USE	NULL
7	8	Office Table	Furniture	TAB123	2024-10-20	Office	Available	NULL

	MaintenanceID	MaintenanceDate	Description	Cost	AssetID
1	1	2023-01-10	General Maintenance	150.00	1
2	2	2022-12-05	Ink Cartridge Replacement	45.00	2
3	3	2023-05-18	Leg Repair	80.00	3
4	4	2023-02-15	Filter Cleaning	60.00	4
5	5	2024-04-01	Lens Cleaning	25.00	5
6	6	2024-10-25	repair	500.00	8
7	9	2024-10-20	virus	200.00	1

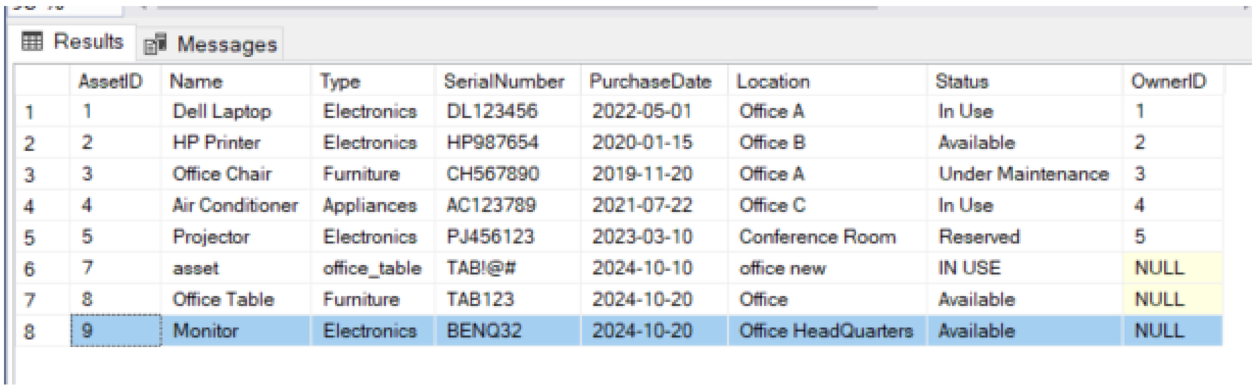
	AllocationID	AllocationDate	ReturnDate	AssetID	EmployeeID
1	1	2023-04-05	NULL	1	1
2	2	2022-10-12	2022-11-15	2	2
3	3	2023-03-01	NULL	4	4
4	4	2022-11-20	2023-02-10	3	3
5	5	2024-01-05	NULL	5	5
6	6	2024-10-12	NULL	7	3
7	7	2024-10-21	2024-10-22	8	5

	ReservationID	ReservationDate	StartDate	EndDate	Status	AssetID	EmployeeID
1	1	2024-01-01	2024-01-05	2024-01-10	Confirmed	5	1
2	2	2023-03-01	2023-03-05	2023-03-15	Cancelled	3	2
3	3	2023-12-20	2024-01-01	2024-01-05	Pending	1	3
4	4	2022-08-15	2022-08-20	2022-08-25	Completed	2	4
5	5	2024-02-01	2024-02-05	2024-02-10	Confirmed	4	5

=> 1. Add Asset



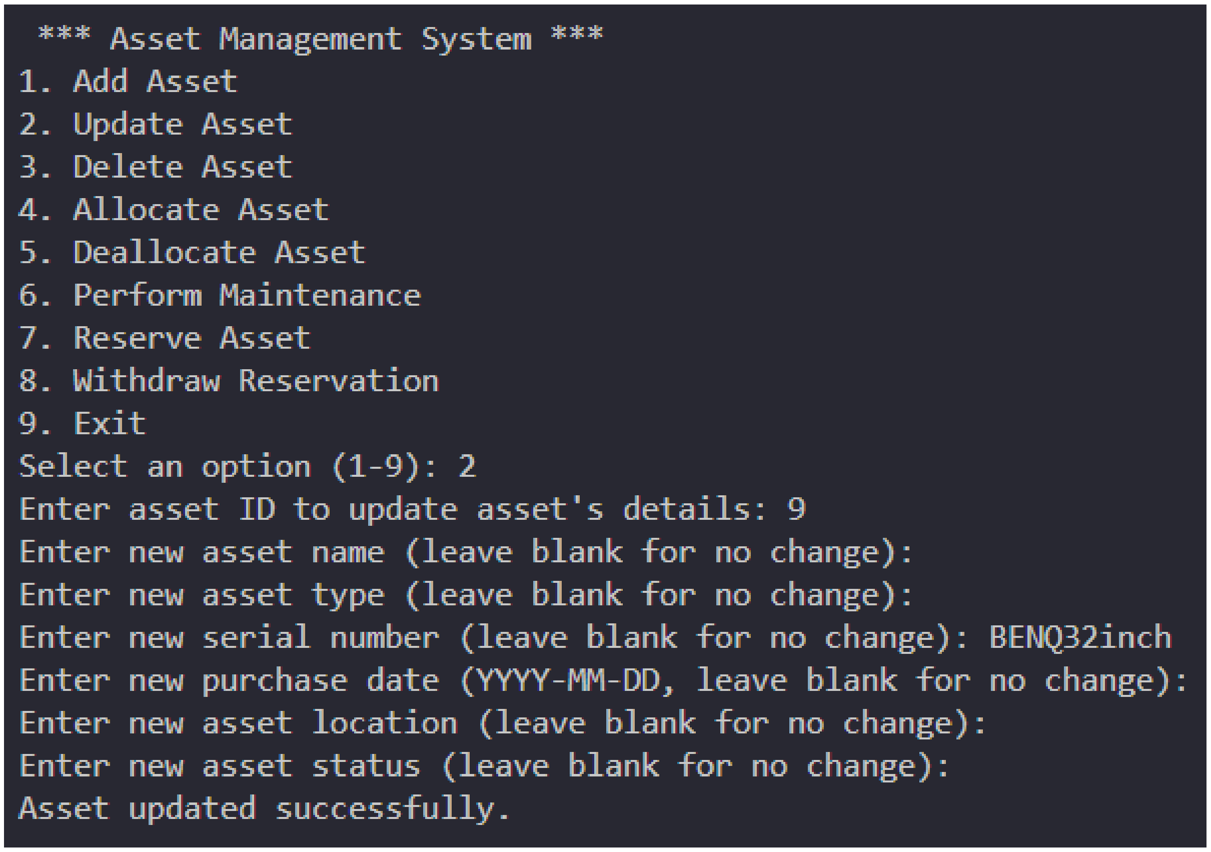
```
 *** Asset Management System ***
1. Add Asset
2. Update Asset
3. Delete Asset
4. Allocate Asset
5. Deallocate Asset
6. Perform Maintenance
7. Reserve Asset
8. Withdraw Reservation
9. Exit
Select an option (1-9): 1
Enter Asset Name: Monitor
Enter Asset Type: Electronics
Enter Serial Number: BENQ32
Enter Purchase Date (YYYY-MM-DD): 2024-10-20
Enter Asset Location: Office HeadQuarters
Enter Asset Status: Available
Asset added successfully.
```



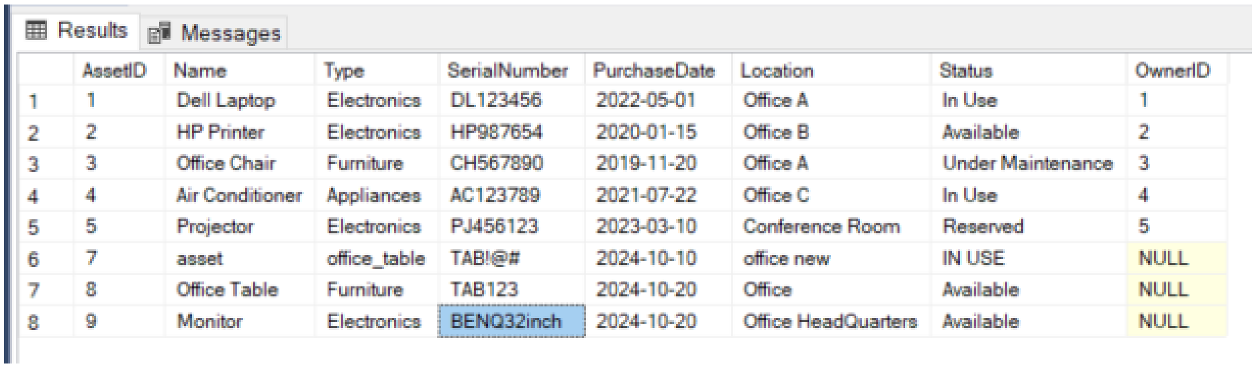
Results Messages

	AssetID	Name	Type	SerialNumber	PurchaseDate	Location	Status	OwnerID
1	1	Dell Laptop	Electronics	DL123456	2022-05-01	Office A	In Use	1
2	2	HP Printer	Electronics	HP987654	2020-01-15	Office B	Available	2
3	3	Office Chair	Furniture	CH567890	2019-11-20	Office A	Under Maintenance	3
4	4	Air Conditioner	Appliances	AC123789	2021-07-22	Office C	In Use	4
5	5	Projector	Electronics	PJ456123	2023-03-10	Conference Room	Reserved	5
6	7	asset	office_table	TAB!@#	2024-10-10	office new	IN USE	NULL
7	8	Office Table	Furniture	TAB123	2024-10-20	Office	Available	NULL
8	9	Monitor	Electronics	BENQ32	2024-10-20	Office HeadQuarters	Available	NULL

=> 2. Update Asset



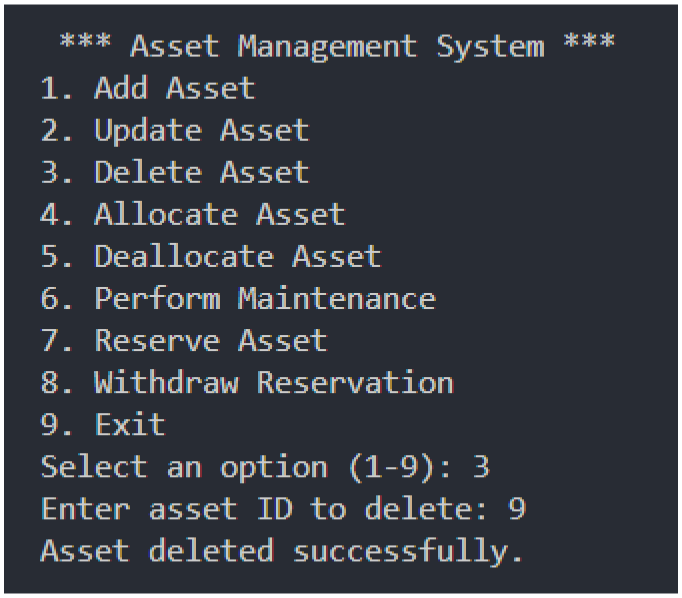
```
 *** Asset Management System ***
1. Add Asset
2. Update Asset
3. Delete Asset
4. Allocate Asset
5. Deallocate Asset
6. Perform Maintenance
7. Reserve Asset
8. Withdraw Reservation
9. Exit
Select an option (1-9): 2
Enter asset ID to update asset's details: 9
Enter new asset name (leave blank for no change):
Enter new asset type (leave blank for no change):
Enter new serial number (leave blank for no change): BENQ32inch
Enter new purchase date (YYYY-MM-DD, leave blank for no change):
Enter new asset location (leave blank for no change):
Enter new asset status (leave blank for no change):
Asset updated successfully.
```



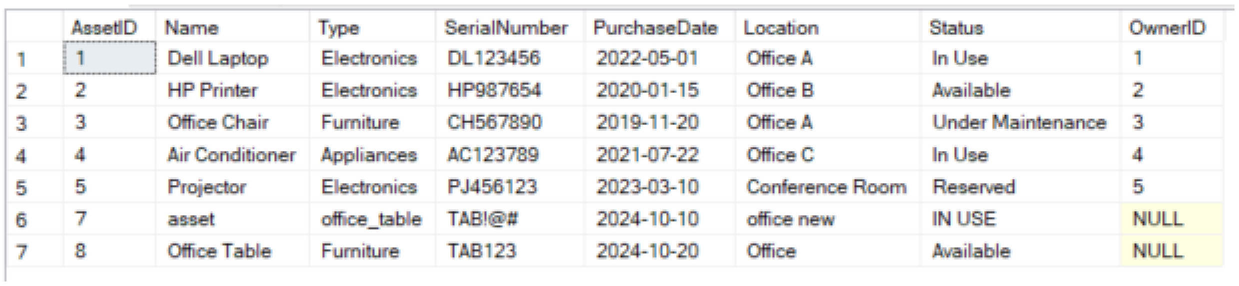
Results | Messages

	AssetID	Name	Type	SerialNumber	PurchaseDate	Location	Status	OwnerID
1	1	Dell Laptop	Electronics	DL123456	2022-05-01	Office A	In Use	1
2	2	HP Printer	Electronics	HP987654	2020-01-15	Office B	Available	2
3	3	Office Chair	Furniture	CH567890	2019-11-20	Office A	Under Maintenance	3
4	4	Air Conditioner	Appliances	AC123789	2021-07-22	Office C	In Use	4
5	5	Projector	Electronics	PJ456123	2023-03-10	Conference Room	Reserved	5
6	7	asset	office_table	TAB!@#	2024-10-10	office new	IN USE	NULL
7	8	Office Table	Furniture	TAB123	2024-10-20	Office	Available	NULL
8	9	Monitor	Electronics	BENQ32inch	2024-10-20	Office HeadQuarters	Available	NULL

=> 3. Delete Asset

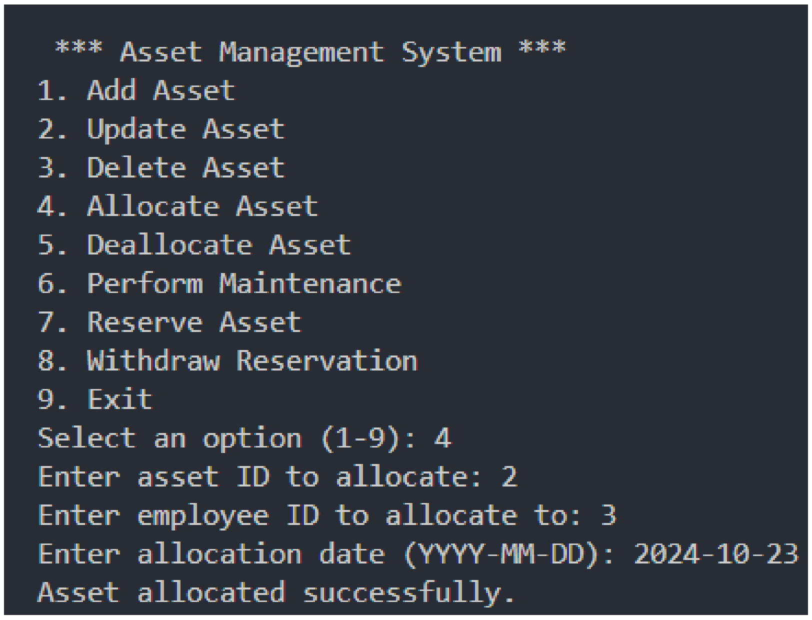


```
 *** Asset Management System ***
1. Add Asset
2. Update Asset
3. Delete Asset
4. Allocate Asset
5. Deallocate Asset
6. Perform Maintenance
7. Reserve Asset
8. Withdraw Reservation
9. Exit
Select an option (1-9): 3
Enter asset ID to delete: 9
Asset deleted successfully.
```

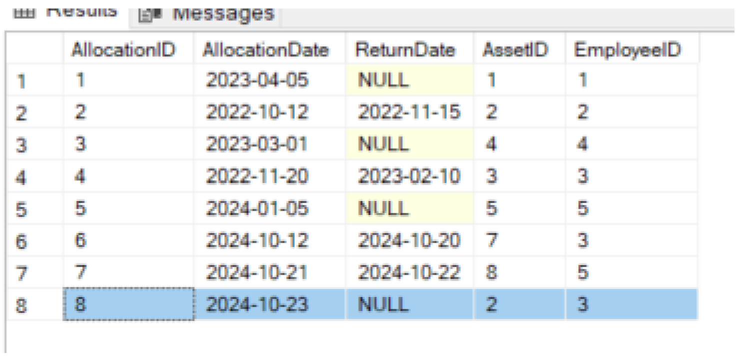


	AssetID	Name	Type	SerialNumber	PurchaseDate	Location	Status	OwnerID
1	1	Dell Laptop	Electronics	DL123456	2022-05-01	Office A	In Use	1
2	2	HP Printer	Electronics	HP987654	2020-01-15	Office B	Available	2
3	3	Office Chair	Furniture	CH567890	2019-11-20	Office A	Under Maintenance	3
4	4	Air Conditioner	Appliances	AC123789	2021-07-22	Office C	In Use	4
5	5	Projector	Electronics	PJ456123	2023-03-10	Conference Room	Reserved	5
6	7	asset	office_table	TAB!@#	2024-10-10	office new	IN USE	NULL
7	8	Office Table	Furniture	TAB123	2024-10-20	Office	Available	NULL

=> 4. Allocate Asset



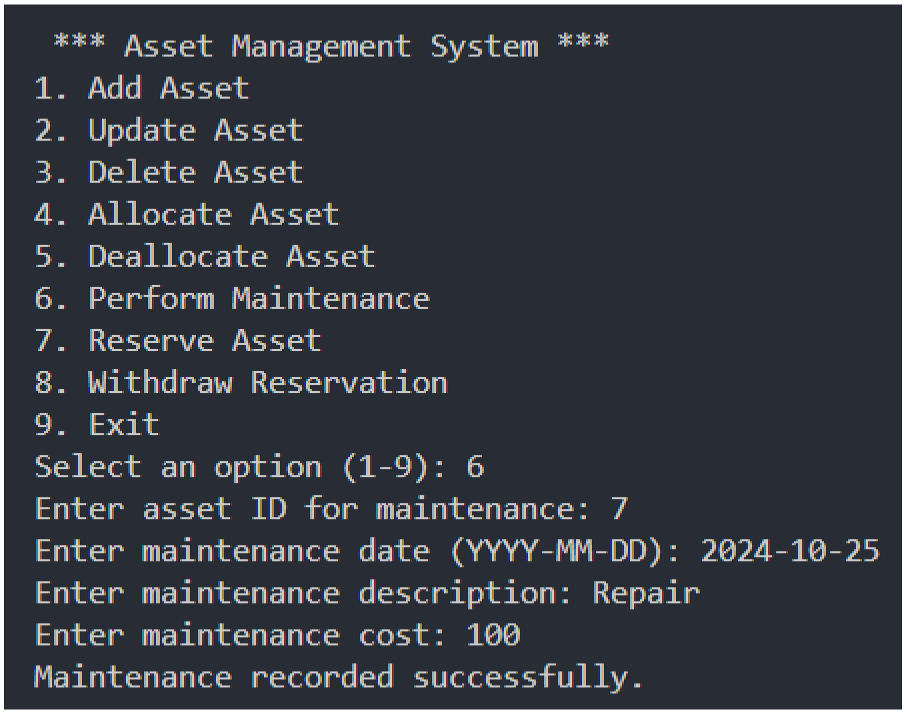
```
 *** Asset Management System ***
1. Add Asset
2. Update Asset
3. Delete Asset
4. Allocate Asset
5. Deallocate Asset
6. Perform Maintenance
7. Reserve Asset
8. Withdraw Reservation
9. Exit
Select an option (1-9): 4
Enter asset ID to allocate: 2
Enter employee ID to allocate to: 3
Enter allocation date (YYYY-MM-DD): 2024-10-23
Asset allocated successfully.
```



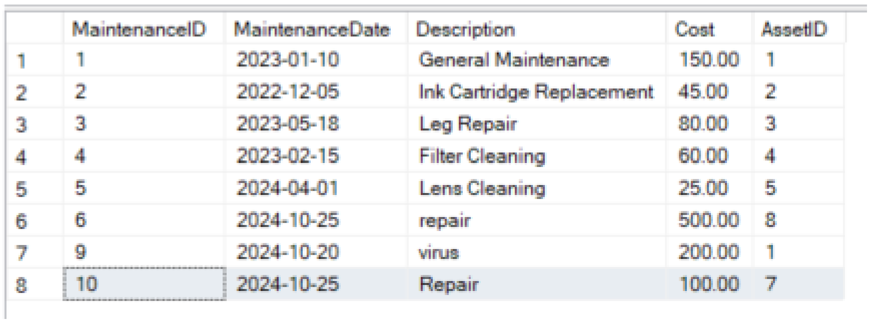
Results Messages

	AllocationID	AllocationDate	ReturnDate	AssetID	EmployeeID
1	1	2023-04-05	NULL	1	1
2	2	2022-10-12	2022-11-15	2	2
3	3	2023-03-01	NULL	4	4
4	4	2022-11-20	2023-02-10	3	3
5	5	2024-01-05	NULL	5	5
6	6	2024-10-12	2024-10-20	7	3
7	7	2024-10-21	2024-10-22	8	5
8	8	2024-10-23	NULL	2	3

=>6. Perform Maintenance

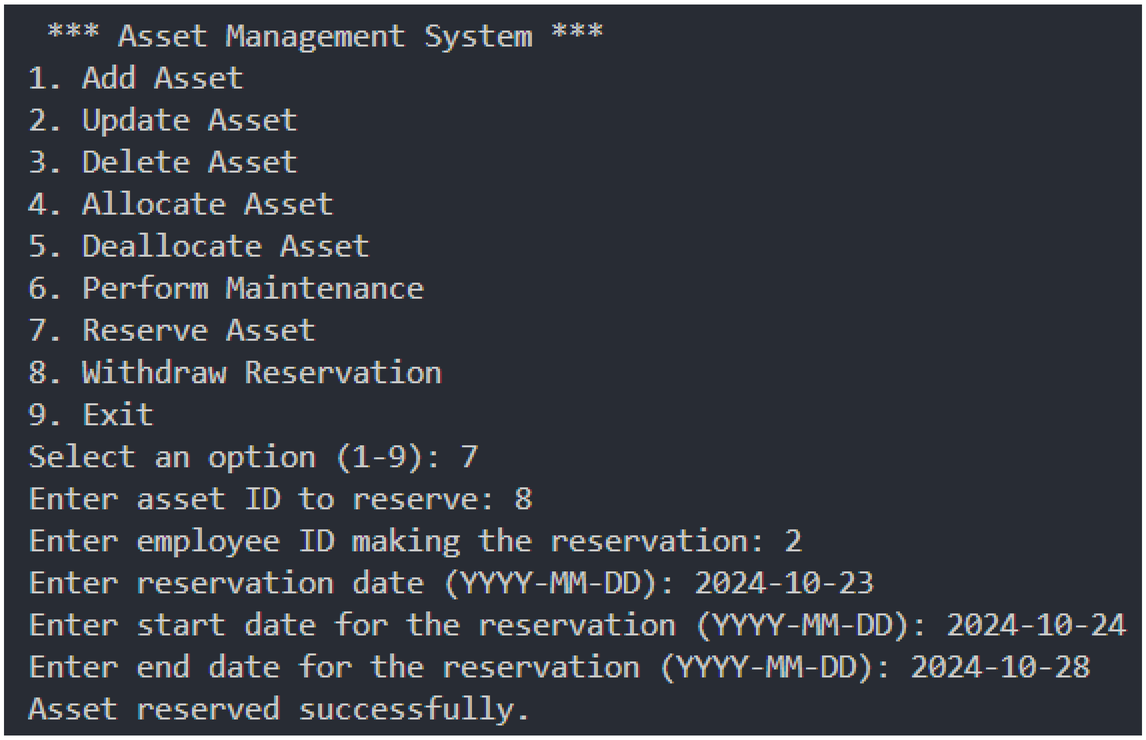


```
 *** Asset Management System ***
1. Add Asset
2. Update Asset
3. Delete Asset
4. Allocate Asset
5. Deallocate Asset
6. Perform Maintenance
7. Reserve Asset
8. Withdraw Reservation
9. Exit
Select an option (1-9): 6
Enter asset ID for maintenance: 7
Enter maintenance date (YYYY-MM-DD): 2024-10-25
Enter maintenance description: Repair
Enter maintenance cost: 100
Maintenance recorded successfully.
```

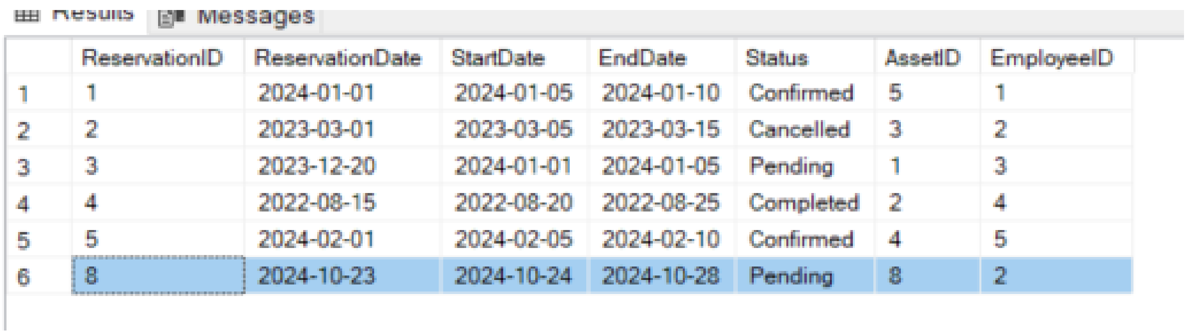


	MaintenanceID	MaintenanceDate	Description	Cost	AssetID
1	1	2023-01-10	General Maintenance	150.00	1
2	2	2022-12-05	Ink Cartridge Replacement	45.00	2
3	3	2023-05-18	Leg Repair	80.00	3
4	4	2023-02-15	Filter Cleaning	60.00	4
5	5	2024-04-01	Lens Cleaning	25.00	5
6	6	2024-10-25	repair	500.00	8
7	9	2024-10-20	virus	200.00	1
8	10	2024-10-25	Repair	100.00	7

=> 7. Reserve Asset



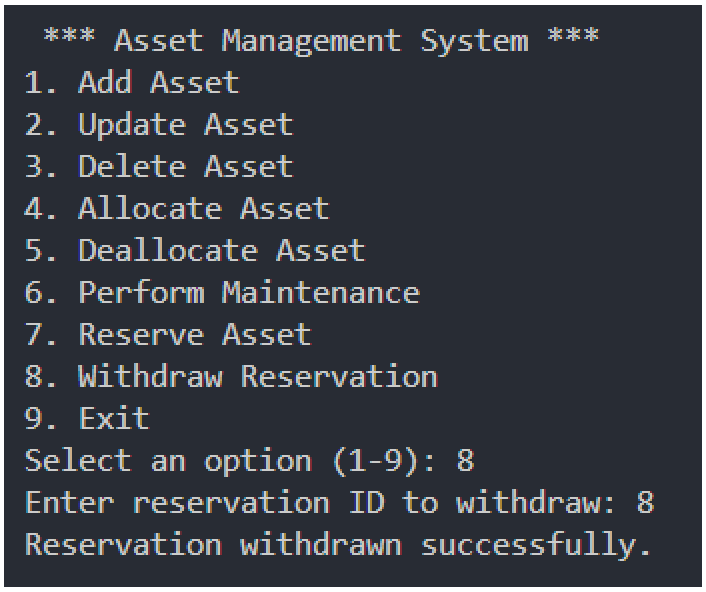
```
 *** Asset Management System ***
1. Add Asset
2. Update Asset
3. Delete Asset
4. Allocate Asset
5. Deallocate Asset
6. Perform Maintenance
7. Reserve Asset
8. Withdraw Reservation
9. Exit
Select an option (1-9): 7
Enter asset ID to reserve: 8
Enter employee ID making the reservation: 2
Enter reservation date (YYYY-MM-DD): 2024-10-23
Enter start date for the reservation (YYYY-MM-DD): 2024-10-24
Enter end date for the reservation (YYYY-MM-DD): 2024-10-28
Asset reserved successfully.
```



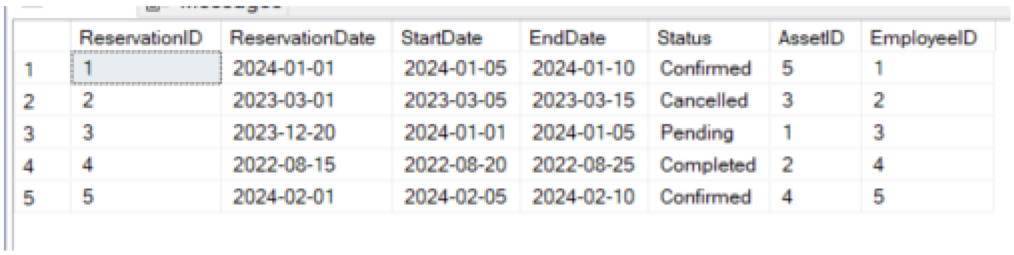
Results Messages

	ReservationID	ReservationDate	StartDate	EndDate	Status	AssetID	EmployeeID
1	1	2024-01-01	2024-01-05	2024-01-10	Confirmed	5	1
2	2	2023-03-01	2023-03-05	2023-03-15	Cancelled	3	2
3	3	2023-12-20	2024-01-01	2024-01-05	Pending	1	3
4	4	2022-08-15	2022-08-20	2022-08-25	Completed	2	4
5	5	2024-02-01	2024-02-05	2024-02-10	Confirmed	4	5
6	8	2024-10-23	2024-10-24	2024-10-28	Pending	8	2

=> 8. Withdraw Reservation

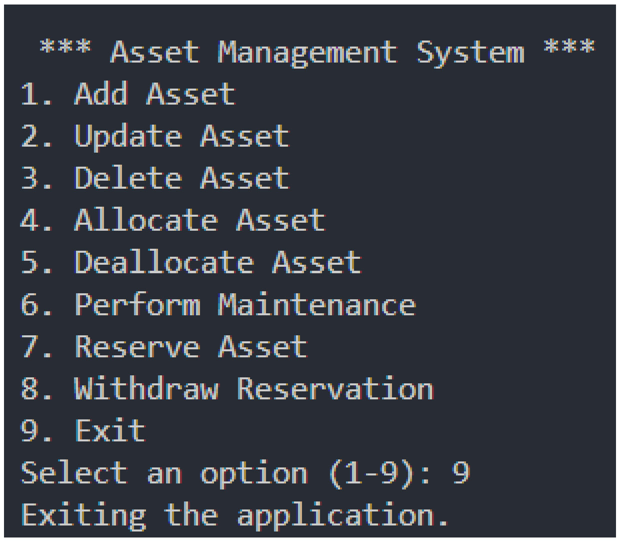


```
 *** Asset Management System ***
1. Add Asset
2. Update Asset
3. Delete Asset
4. Allocate Asset
5. Deallocate Asset
6. Perform Maintenance
7. Reserve Asset
8. Withdraw Reservation
9. Exit
Select an option (1-9): 8
Enter reservation ID to withdraw: 8
Reservation withdrawn successfully.
```



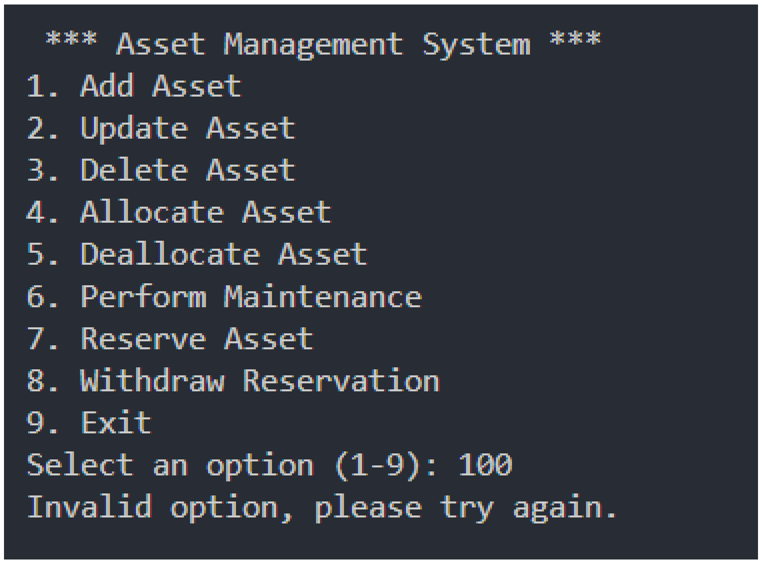
	ReservationID	ReservationDate	StartDate	EndDate	Status	AssetID	EmployeeID
1	1	2024-01-01	2024-01-05	2024-01-10	Confirmed	5	1
2	2	2023-03-01	2023-03-05	2023-03-15	Cancelled	3	2
3	3	2023-12-20	2024-01-01	2024-01-05	Pending	1	3
4	4	2022-08-15	2022-08-20	2022-08-25	Completed	2	4
5	5	2024-02-01	2024-02-05	2024-02-10	Confirmed	4	5

=>9. Exit



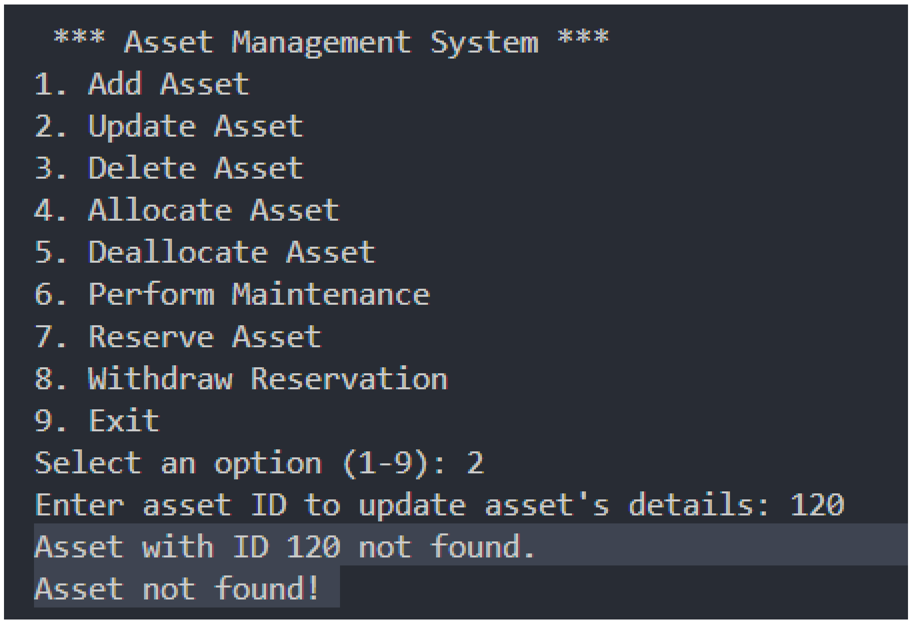
```
 *** Asset Management System ***
1. Add Asset
2. Update Asset
3. Delete Asset
4. Allocate Asset
5. Deallocate Asset
6. Perform Maintenance
7. Reserve Asset
8. Withdraw Reservation
9. Exit
Select an option (1-9): 9
Exiting the application.
```

=>10. Invalid Input



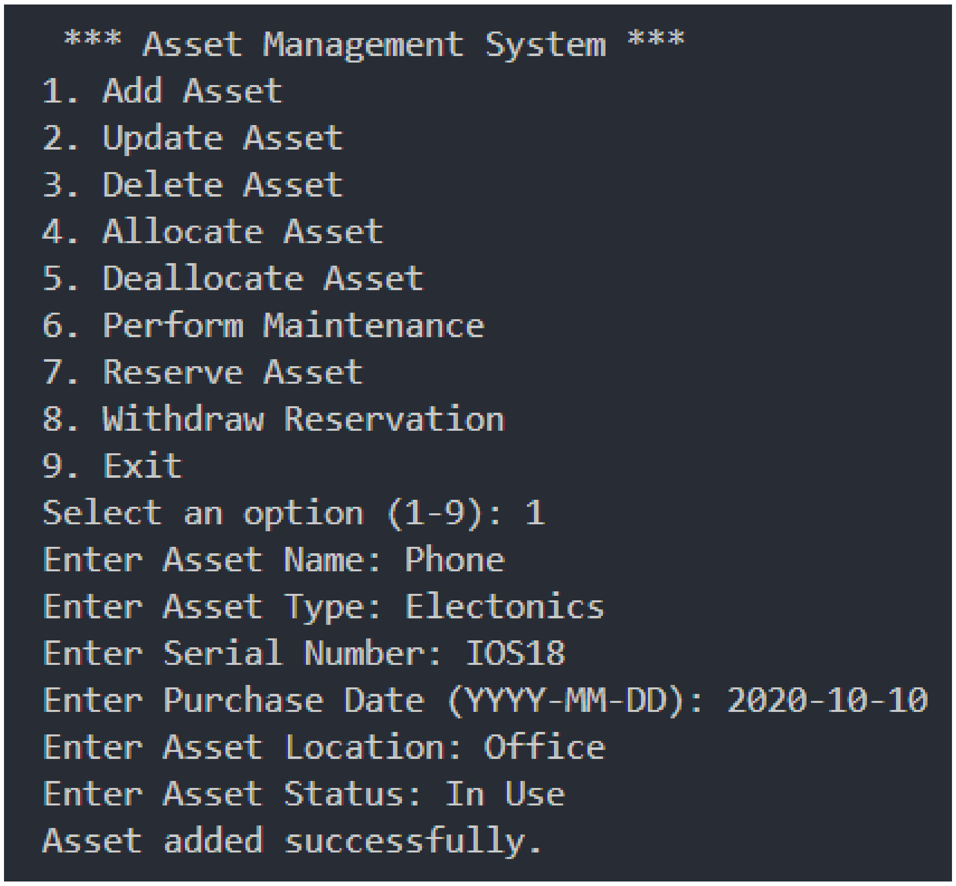
```
 *** Asset Management System ***
1. Add Asset
2. Update Asset
3. Delete Asset
4. Allocate Asset
5. Deallocate Asset
6. Perform Maintenance
7. Reserve Asset
8. Withdraw Reservation
9. Exit
Select an option (1-9): 100
Invalid option, please try again.
```

=>11. AssetNotFoundException.py

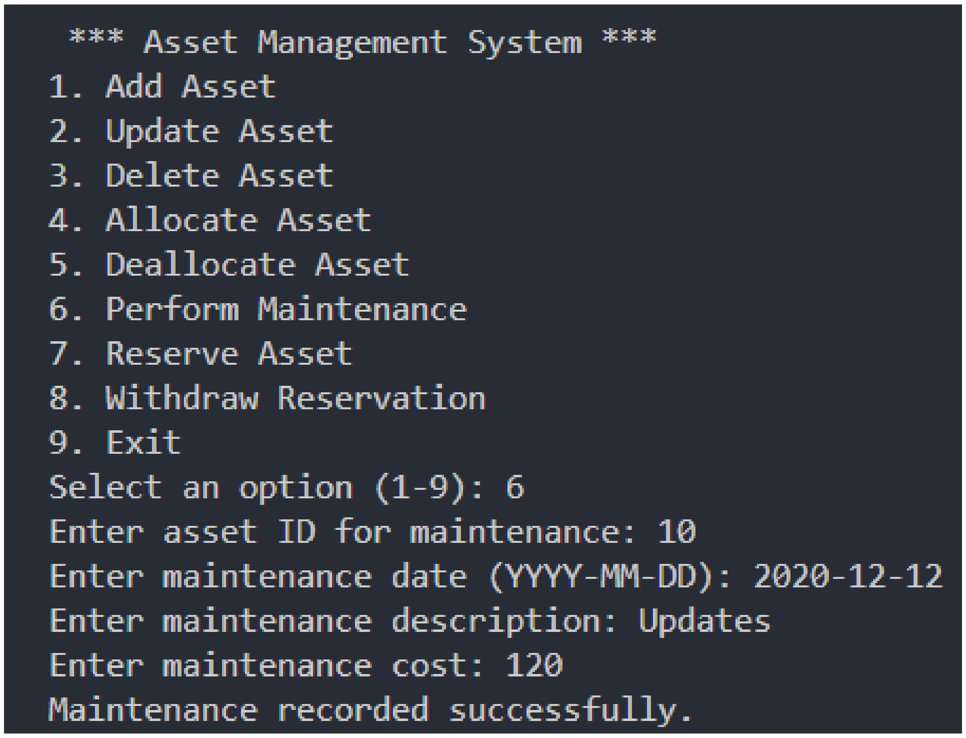


```
 *** Asset Management System ***
1. Add Asset
2. Update Asset
3. Delete Asset
4. Allocate Asset
5. Deallocate Asset
6. Perform Maintenance
7. Reserve Asset
8. Withdraw Reservation
9. Exit
Select an option (1-9): 2
Enter asset ID to update asset's details: 120
Asset with ID 120 not found.
Asset not found!
```

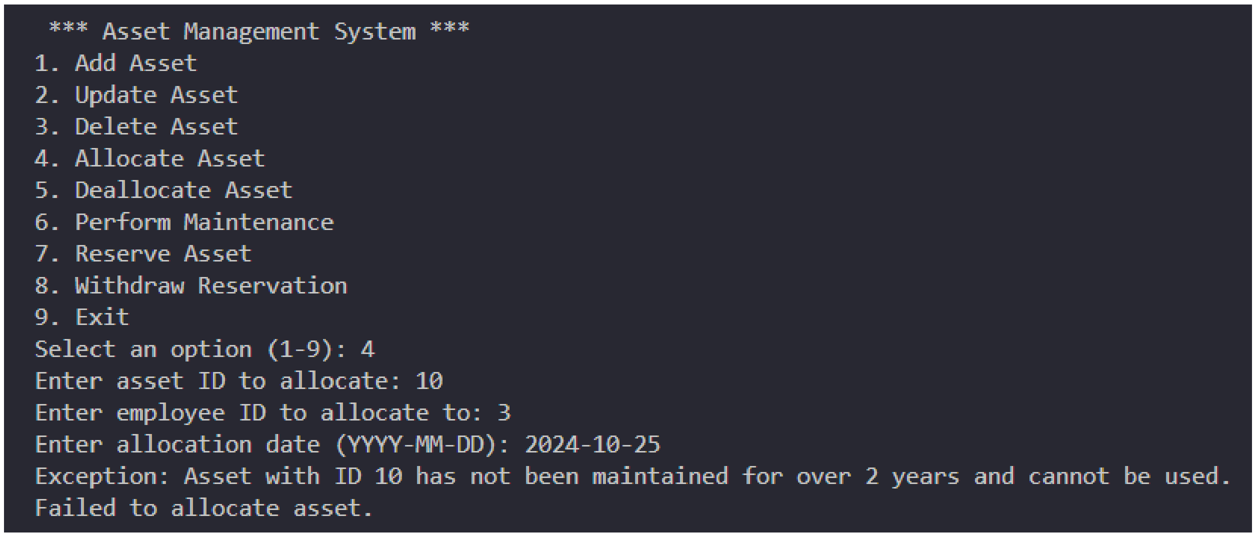
=>11. AssetNotMaintainException.py



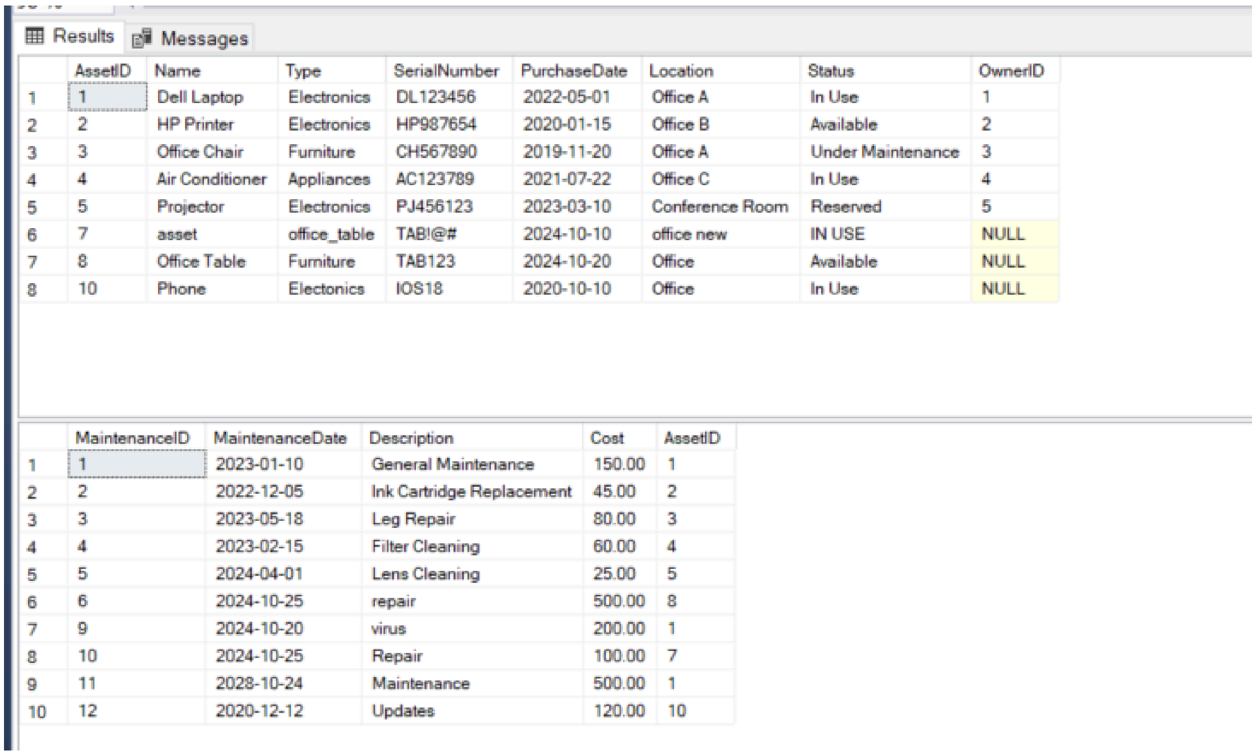
```
 *** Asset Management System ***
1. Add Asset
2. Update Asset
3. Delete Asset
4. Allocate Asset
5. Deallocate Asset
6. Perform Maintenance
7. Reserve Asset
8. Withdraw Reservation
9. Exit
Select an option (1-9): 1
Enter Asset Name: Phone
Enter Asset Type: Electonics
Enter Serial Number: IOS18
Enter Purchase Date (YYYY-MM-DD): 2020-10-10
Enter Asset Location: Office
Enter Asset Status: In Use
Asset added successfully.
```



```
 *** Asset Management System ***
1. Add Asset
2. Update Asset
3. Delete Asset
4. Allocate Asset
5. Deallocate Asset
6. Perform Maintenance
7. Reserve Asset
8. Withdraw Reservation
9. Exit
Select an option (1-9): 6
Enter asset ID for maintenance: 10
Enter maintenance date (YYYY-MM-DD): 2020-12-12
Enter maintenance description: Updates
Enter maintenance cost: 120
Maintenance recorded successfully.
```



```
 *** Asset Management System ***
1. Add Asset
2. Update Asset
3. Delete Asset
4. Allocate Asset
5. Deallocate Asset
6. Perform Maintenance
7. Reserve Asset
8. Withdraw Reservation
9. Exit
Select an option (1-9): 4
Enter asset ID to allocate: 10
Enter employee ID to allocate to: 3
Enter allocation date (YYYY-MM-DD): 2024-10-25
Exception: Asset with ID 10 has not been maintained for over 2 years and cannot be used.
Failed to allocate asset.
```

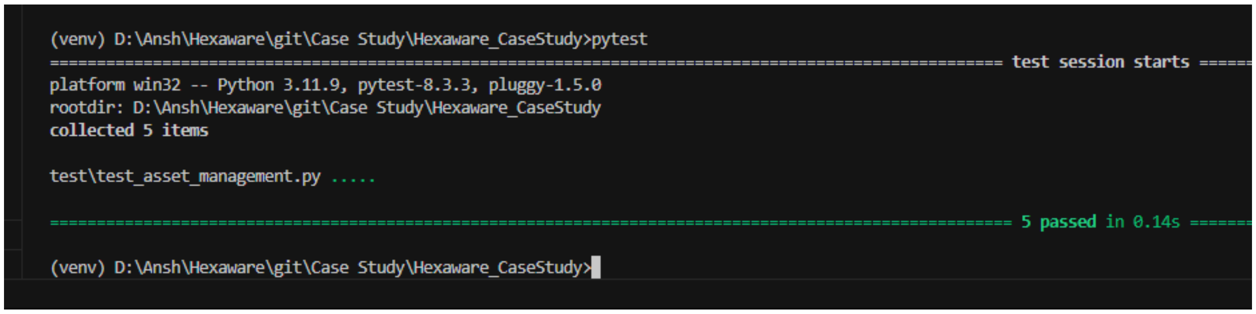


	AssetID	Name	Type	SerialNumber	PurchaseDate	Location	Status	OwnerID
1	1	Dell Laptop	Electronics	DL123456	2022-05-01	Office A	In Use	1
2	2	HP Printer	Electronics	HP987654	2020-01-15	Office B	Available	2
3	3	Office Chair	Furniture	CH567890	2019-11-20	Office A	Under Maintenance	3
4	4	Air Conditioner	Appliances	AC123789	2021-07-22	Office C	In Use	4
5	5	Projector	Electronics	PJ456123	2023-03-10	Conference Room	Reserved	5
6	7	asset	office_table	TAB!@#	2024-10-10	office new	IN USE	NULL
7	8	Office Table	Furniture	TAB123	2024-10-20	Office	Available	NULL
8	10	Phone	Electonics	IOS18	2020-10-10	Office	In Use	NULL

	MaintenanceID	MaintenanceDate	Description	Cost	AssetID
1	1	2023-01-10	General Maintenance	150.00	1
2	2	2022-12-05	Ink Cartridge Replacement	45.00	2
3	3	2023-05-18	Leg Repair	80.00	3
4	4	2023-02-15	Filter Cleaning	60.00	4
5	5	2024-04-01	Lens Cleaning	25.00	5
6	6	2024-10-25	repair	500.00	8
7	9	2024-10-20	virus	200.00	1
8	10	2024-10-25	Repair	100.00	7
9	11	2028-10-24	Maintenance	500.00	1
10	12	2020-12-12	Updates	120.00	10

Test Cases

- Write test case to test asset created successfully or not.
- Write test case to test asset is added to maintenance successfully or not.
- Write test case to test asset is reserved successfully or not.
- write test case to test exception is thrown correctly or not when employee id or asset id not found in database.



```
(venv) D:\Ansh\Hexaware\git\Case Study\Hexaware_CaseStudy>pytest
================================================================================================= test session starts ======
platform win32 -- Python 3.11.9, pytest-8.3.3, pluggy-1.5.0
rootdir: D:\Ansh\Hexaware\git\Case Study\Hexaware_CaseStudy
collected 5 items

test\test_asset_management.py .....

================================================================================================== 5 passed in 0.14s ======

(venv) D:\Ansh\Hexaware\git\Case Study\Hexaware_CaseStudy>
```