Case Study on Asset Management System

Team 7

Melvin Jones E

Navadharshini J

Project Overview :

The Digital Asset Management System is designed to streamline the tracking, maintenance, and allocation of assets within an organization. The application allows users to perform CRUD operations on assets, track their maintenance history, allocate them to employees, and manage reservations. The project was developed using Java with JDBC for database interaction, SQL for the schema, and JUnit for testing.

Schema Design :

Created tables for employees, assets, Maintenance Records, Reservations, Asset Allocations.

Employees Table:

```
mysql> desc employees;
+-------------+--------------+------+-----+---------+-------+
| Field       | Type         | Null | Key | Default | Extra |
+-------------+--------------+------+-----+---------+-------+
| employee_id | int          | NO   | PRI | NULL    |       |
| name        | varchar(50)  | YES  |     | NULL    |       |
| department  | varchar(50)  | YES  |     | NULL    |       |
| email       | varchar(100) | YES  |     | NULL    |       |
| password    | varchar(255) | YES  |     | NULL    |       |
+-------------+--------------+------+-----+---------+-------+
5 rows in set (0.02 sec)
```

Assets Table schema :

```
mysql> desc assets;
+---------------+-------------+------+-----+---------+-------+
| Field         | Type        | Null | Key | Default | Extra |
+---------------+-------------+------+-----+---------+-------+
| asset_id      | int         | NO   | PRI | NULL    |       |
| name          | varchar(20) | YES  |     | NULL    |       |
| type          | varchar(50) | YES  |     | NULL    |       |
| serial_number | varchar(30) | YES  | UNI | NULL    |       |
| purchase_date | date        | YES  |     | NULL    |       |
| location      | varchar(50) | YES  |     | NULL    |       |
| status        | varchar(50) | YES  |     | NULL    |       |
| owner_id      | int         | YES  | MUL | NULL    |       |
+---------------+-------------+------+-----+---------+-------+
8 rows in set (0.01 sec)
```

Asset Allocations schema :

```
mysql> desc asset_allocations;
+----------------+------+------+-----+---------+----------------+
| Field          | Type | Null | Key | Default | Extra          |
+----------------+------+------+-----+---------+----------------+
| allocation_id  | int  | NO   | PRI | NULL    | auto_increment |
| asset_id       | int  | YES  | MUL | NULL    |                |
| employee_id    | int  | YES  | MUL | NULL    |                |
| allocation_date | date | YES  |     | NULL    |                |
| return_date    | date | YES  |     | NULL    |                |
+----------------+------+------+-----+---------+----------------+
5 rows in set (0.01 sec)
```

Maintenance Records Schema :

```
mysql> desc maintenance_records;
+-----------------+---------------+------+-----+---------+----------------+
| Field           | Type          | Null | Key | Default | Extra          |
+-----------------+---------------+------+-----+---------+----------------+
| maintenance_id  | int           | NO   | PRI | NULL    | auto_increment |
| asset_id        | int           | YES  | MUL | NULL    |                |
| maintenance_date | date          | YES  |     | NULL    |                |
| description     | varchar(255)  | YES  |     | NULL    |                |
| cost            | decimal(10,2) | YES  |     | NULL    |                |
+-----------------+---------------+------+-----+---------+----------------+
5 rows in set (0.01 sec)
```

Reservations Schema :

```
mysql> desc reservations;
+-----------------+-------------+------+-----+---------+----------------+
| Field           | Type        | Null | Key | Default | Extra          |
+-----------------+-------------+------+-----+---------+----------------+
| reservation_id  | int         | NO   | PRI | NULL    | auto_increment |
| asset_id        | int         | YES  | MUL | NULL    |                |
| employee_id     | int         | YES  | MUL | NULL    |                |
| reservation_date | date        | YES  |     | NULL    |                |
| start_date      | date        | YES  |     | NULL    |                |
| end_date        | date        | YES  |     | NULL    |                |
| status          | varchar(50) | YES  |     | NULL    |                |
+-----------------+-------------+------+-----+---------+----------------+
7 rows in set (0.01 sec)
```

Project Structure :

1. **Entity :** Created Entity package to these classes and added getter, setter methods

   → Employee Class
   → Asset Class
   → Asset Allocation Class
   → Perform Maintenance Class
   → Reservation Class

2. **Dao :** Created Interface called AssetManagementService and Implementation file called AssetMangementServiceImpl to implement the interface methods.

   → AssetManagementService Interface

   → AssetMangementServiceImpl class

3. **Exception :** Created exceptions for Asset, Employee not found and check if asset is not maintained for 2 years.

   → AssetNotFound Exception

   → AssetNotMaintain Exception

   → EmployeeNotFound Exception

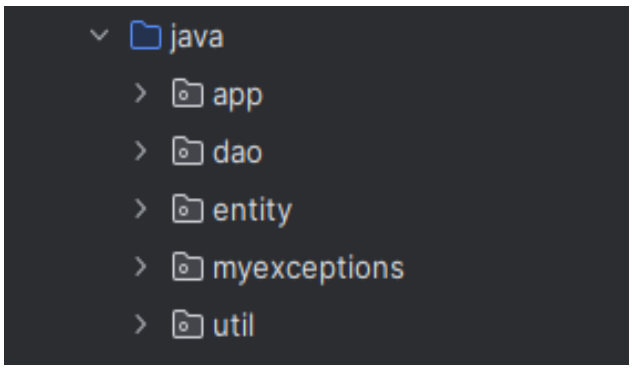
4. **Util :** Created Util package to Connect to the database.

   → DBConnUtil class

   → DBPropertyUtil class

5. **Main :** Main class to trigger all the methods and get input from user

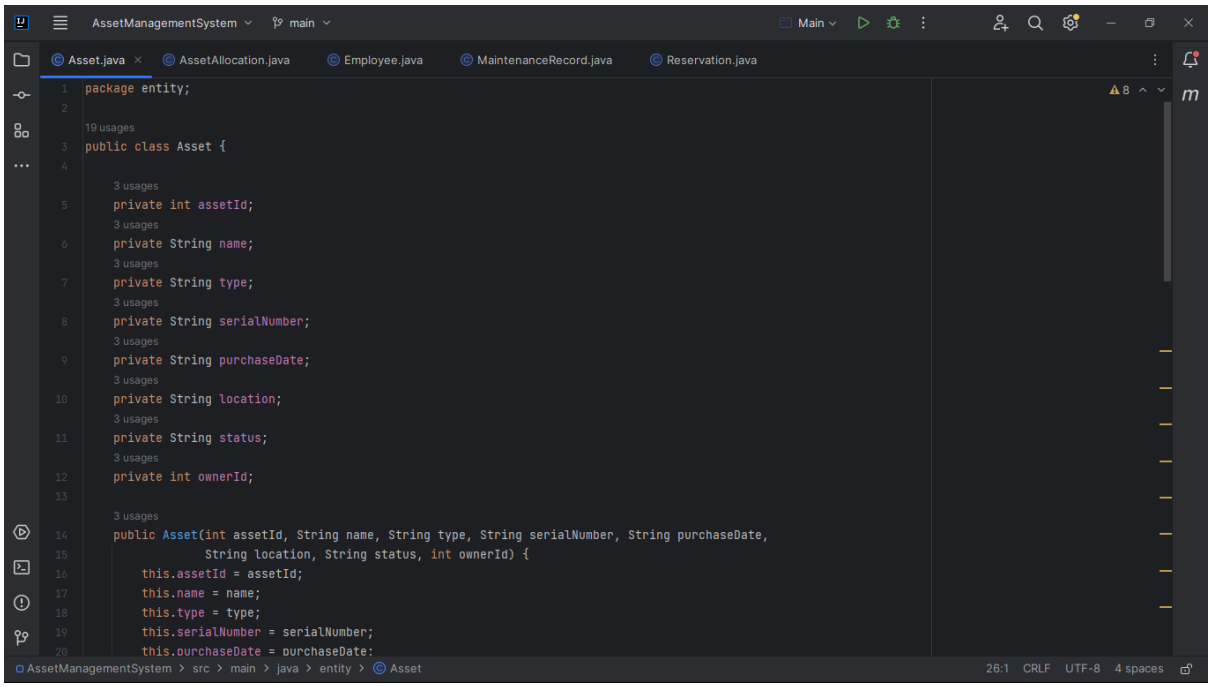
   → Main class



Entity Package :

Created entity package for Employee, Assets, Asset Allocations, Maintenance Records and Reservations.

Asset.java



```java
package entity;

public class Asset {

    private int assetId;
    private String name;
    private String type;
    private String serialNumber;
    private String purchaseDate;
    private String location;
    private String status;
    private int ownerId;

    public Asset(int assetId, String name, String type, String serialNumber, String purchaseDate,
                 String location, String status, int ownerId) {
        this.assetId = assetId;
        this.name = name;
        this.type = type;
        this.serialNumber = serialNumber;
        this.purchaseDate = purchaseDate;
```

AssetAllocation.java

```java
package entity;

public class AssetAllocation {

    private int allocationId;
    private int assetId;
    private int employeeId;
    private String allocationDate;
    private String returnDate;

    public AssetAllocation(int allocationId, int assetId, int employeeId, String allocationDate, String returnDate) {
        this.allocationId = allocationId;
        this.assetId = assetId;
        this.employeeId = employeeId;
        this.allocationDate = allocationDate;
        this.returnDate = returnDate;
    }


    public int getAllocationId() { return allocationId; }
```

Employee.java

```java
package entity;

public class Employee {

    private int employeeId;
    private String name;
    private String department;
    private String email;
    private String password;

    public Employee(int employeeId, String name, String department, String email, String password) {
        this.employeeId = employeeId;
        this.name = name;
        this.department = department;
        this.email = email;
        this.password = password;
    }


    public int getEmployeeId() { return employeeId; }
```

MaintenanceRecords.java

```java
    private String description;
    private double cost;

    public MaintenanceRecord(int maintenanceId, int assetId, String maintenanceDate, String description, double cost) {
        this.maintenanceId = maintenanceId;
        this.assetId = assetId;
        this.maintenanceDate = maintenanceDate;
        this.description = description;
        this.cost = cost;
    }

    public int getMaintenanceId() { return maintenanceId; }

    public void setMaintenanceId(int maintenanceId) { this.maintenanceId = maintenanceId; }

    public int getAssetId() { return assetId; }

    public void setAssetId(int assetId) { this.assetId = assetId; }

    public String getMaintenanceDate() { return maintenanceDate; }
```

Reservation.java

```java
package entity;

public class Reservation {

    private int reservationId;
    private int assetId;
    private int employeeId;
    private String reservationDate;
    private String startDate;
    private String endDate;
    private String status;

    public Reservation(int reservationId, int assetId, int employeeId, String reservationDate,
                       String startDate, String endDate, String status) {
        this.reservationId = reservationId;
        this.assetId = assetId;
        this.employeeId = employeeId;
        this.reservationDate = reservationDate;
        this.startDate = startDate;
        this.endDate = endDate;
        this.status = status;
    }

    public int getReservationId() { return reservationId; }

    public void setReservationId(int reservationId) { this.reservationId = reservationId; }

    public int getAssetId() { return assetId; }

    public void setAssetId(int assetId) { this.assetId = assetId; }
```

Util Package :

Created Util package to establish connection between database with the help of Maven dependency.

DBConnUtil.java

```java
package util;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;

8 usages
public class DBConnUtil {

    8 usages
    public static Connection getConnection() {

        Connection con = null;
        String fileName = "db.properties";
        try {
            String url = DBPropertyUtil.getConnectionString(fileName);
            con = DriverManager.getConnection(url);


        } catch (SQLException e) {
            e.printStackTrace();
        }

        return con;
    }
}
```

DBPropertyUtil.java

```java
package util;

import java.io.FileInputStream;
import java.io.IOException;
import java.util.Properties;

1 usage
public class DBPropertyUtil {
    1 usage
    public static String getConnectionString(String fileName) {
        String url = null;
        Properties props = new Properties();

        try (FileInputStream fis = new FileInputStream(fileName)) {
            props.load(fis);

            url = props.getProperty("url") + "/" + props.getProperty("database")
                    + "?user=" + props.getProperty("user")
                    + "&password=" + props.getProperty("password");

        } catch (IOException e) {
            e.printStackTrace();
        }

        return url;
    }
}
```

db.properties file to store data of database

```
url = jdbc:mysql://localhost:3306
user = root
password = melvin@sql
database = application
```

Dao Package :

Created an Interface AssetManagementService to show methods to be implemented

AssetManagementService.java

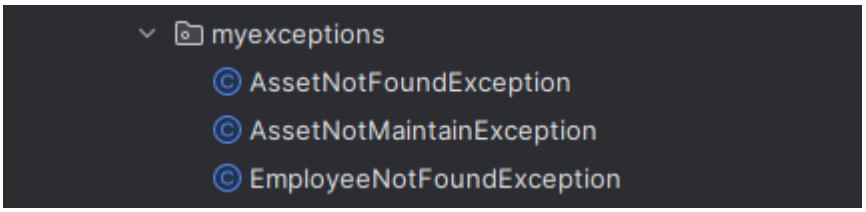
```
package dao;
import entity.Asset;

public interface AssetManagementService {


    boolean addAsset(Asset asset);

    boolean updateAsset(Asset asset);

    boolean deleteAsset(int assetId);

    boolean allocateAsset(int assetId,int employeeId, String allocationDate);

    boolean deallocateAsset(int assetId, int employeeId, String returnDate);

    boolean performMaintenance(int assetId, String maintenanceDate, String description, double cost);


    boolean reserveAsset(int assetId, int employeeId, String reservationDate, String startDate, String endDate, String status);


    boolean withdrawReservation(int reservationId);

}
```

And implemented those methods in AssetManagementImpl class by inheriting the interface .

```java
public class AssetManagementServiceImpl implements AssetManagementService{



    public boolean addAsset(Asset asset) {

        String query="INSERT INTO assets(asset_id,name,type,serial_number,purchase_date,location,status,owner_id) VALUES(?,?,?,?,?,?,?,?)";

        try(Connection con=DBConnUtil.getConnection()) {

            PreparedStatement p=con.prepareStatement( sql: "Select * from employees where employee_id=?");
            p.setInt( parameterIndex: 1,asset.getOwnerId());
            ResultSet rs=p.executeQuery();
            if(!rs.next()) throw new EmployeeNotFoundException("Employee with id "+ asset.getOwnerId()+" is not found");

            p=con.prepareStatement(query);
            p.setInt( parameterIndex: 1, asset.getAssetId());
            p.setString( parameterIndex: 2,asset.getName());
            p.setString( parameterIndex: 3, asset.getType());
            p.setString( parameterIndex: 4,asset.getSerialNumber());
            p.setString( parameterIndex: 5, asset.getPurchaseDate());
            p.setString( parameterIndex: 6,asset.getLocation());
            p.setString( parameterIndex: 7, asset.getStatus());
            p.setInt( parameterIndex: 8, asset.getOwnerId());
            p.executeUpdate();
        }
        catch(EmployeeNotFoundException | SQLException e) {
            System.err.println(e.getMessage());
```

Myexceptions Package

Created User defined exceptions whenever an Asset Id or Employee Id is not found, By inheriting the Exception class.



AssetNotFoundException.java

```java
package myexceptions;


public class AssetNotFoundException extends RuntimeException {

    public AssetNotFoundException(String message) { super(message); }
}
```

AssetNotMaintainException.java

```java
package myexceptions;


public class AssetNotMaintainException extends RuntimeException {

    public AssetNotMaintainException(String message) { super(message); }
}
```

EmployeeNotFoundException.java

```java
package myexceptions;


public class EmployeeNotFoundException extends Exception{

        public EmployeeNotFoundException(String message) { super(message); }
}
```

Added these Exceptions in required methods , which might throw errors and caught them .

Example :

```java
    @Override
    public boolean allocateAsset(int assetId, int employeeId, String allocationDate) {

        String query="INSERT INTO asset_allocations(asset_id,employee_id,allocation_date) VALUES (?,?,?)";

        try(Connection con = DBConnUtil.getConnection()) {
            PreparedStatement p=con.prepareStatement( sql: "Select * from assets where asset_id=?");

            p.setInt( parameterIndex: 1,assetId);
            ResultSet rs=p.executeQuery();
            if(!rs.next()) throw new AssetNotFoundException("Asset with id "+assetId+" is not found");

            p=con.prepareStatement( sql: "Select * from employees where employee_id=?");
            p.setInt( parameterIndex: 1,employeeId);
            rs=p.executeQuery();
            if(!rs.next()) throw new EmployeeNotFoundException("Employee with id " +employeeId+" is not found");

            PreparedStatement psmt = con.prepareStatement(query);
            psmt.setInt( parameterIndex: 1,assetId);
            psmt.setInt( parameterIndex: 2,employeeId);
            psmt.setString( parameterIndex: 3,allocationDate);
            psmt.executeUpdate();
        }
        catch(AssetNotFoundException | EmployeeNotFoundException |SQLException e) {
            System.err.println(e.getMessage());
            return false;
        }
        return true;
    }
```

Created Main.java File to trigger the Implemented Methods.

```java
public class Main {

    public static void main(String[] args) {

        AssetManagementService imp = new AssetManagementServiceImpl();
        Scanner sc = new Scanner(System.in);

        while (true){
            System.out.println("--- Digital Asset Management System ---");
            System.out.println("1. Add Asset");
            System.out.println("2. Update Asset");
            System.out.println("3. Delete Asset");
            System.out.println("4. Allocate Asset");
            System.out.println("5. Deallocate Asset");
            System.out.println("6. Perform Maintenance");
            System.out.println("7. Reserve Asset");
            System.out.println("8. Withdraw Asset");
            System.out.println("9. Exit");
            System.out.print("Enter the operation to perform: ");

            int choice = sc.nextInt();
            sc.nextLine();

            switch (choice) {
                case 1:
                    addAsset(imp, sc);
                    break;
                case 2:
                    updateAsset(imp, sc);
```

Results :

Case 1 : Add

```
C:\Users\91739\.jdks\openjdk-20.0.2\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.1.1\lib\idea_rt.jar=6288:C:\Pr
--- Digital 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 Asset
9. Exit
Enter the operation to perform: 1
Enter the assetId: 13
Enter the Asset Name: sony laptop
Enter the type of the asset (laptop, vehicle, equipment, etc..): laptop
Enter the Serial number of an asset: asd123890
Enter the Purchase Date (yyyy-mm-dd): 2024-11-23
Enter the Location: TamilNadu
Enter the Status of an asset (in use, decommissioned, under maintenance): in use
Enter the ownerId: 2
Asset Added Successfully
```

Database check whether the asset added or not

```
mysql> select * from assets
    -> ;
+----------+---------------------+-------------+---------------+---------------+---------------+-------------------+----------+
| asset_id | name                | type        | serial_number | purchase_date | location      | status            | owner_id |
+----------+---------------------+-------------+---------------+---------------+---------------+-------------------+----------+
|        1 | tv                  | electronics | 12312         | 2024-09-01    | chennai       | new               |        1 |
|        2 | iphone              | mobile      | 43892         | 2027-01-01    | USA           | new               |        2 |
|        5 | samsung             | mobile      | a12daa        | 2024-01-05    | chennai       | new               |        6 |
|       10 | table               | equipment   | alsd2342      | 2024-12-01    | pune          | in use            |        6 |
|       13 | sony laptop         | laptop      | asd123890     | 2024-11-23    | TamilNadu     | in use            |        2 |
```

Case 2 : Update

```
C:\Users\91739\.jdks\openjdk-20.0.2\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.1.1\lib\idea_rt.jar=6305:C:\Pr
--- Digital 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 Asset
9. Exit
Enter the operation to perform: 2
Enter the Asset id to update: 13
Enter the Asset Name to update: Asus tuf
Enter the type of the asset to update (laptop, vehicle, equipment): laptop
Enter the Serial number of an asset to update: 439sagae3
Enter the Purchase Date (yyyy-mm-dd) to update: 2024-8-1
Enter the Location to update: chennai
Enter the Status of an asset (in use, decommissioned, under maintenance) to update: in use
Enter the ownerId to update: 4
Asset Updated Successfully
--- Digital Asset Management System ---
1. Add Asset
2. Update Asset
3. Delete Asset
4. Allocate Asset
5. Deallocate Asset
```

Database Check

```
MySQL 8.0 Command Line Cli

mysql> select * from assets;
+----------+---------------------+-------------+---------------+---------------+---------------+--------------------+----------+
| asset_id | name                | type        | serial_number | purchase_date | location      | status             | owner_id |
+----------+---------------------+-------------+---------------+---------------+---------------+--------------------+----------+
|        1 | tv                  | electronics | 12312         | 2024-09-01    | chennai       | new                |        1 |
|        2 | iphone              | mobile      | 43892         | 2027-01-01    | USA           | new                |        2 |
|        5 | samsung             | mobile      | a12daa        | 2024-01-05    | chennai       | new                |        6 |
|       10 | table               | equipment   | alsd2342      | 2024-12-01    | pune          | in use             |        6 |
|       13 | Asus tuf            | laptop      | 439sagae3     | 2024-08-01    | chennai       | in use             |        4 |
```

Similarly for all the cases

```
AssetManagementSystem  main                                  Main
Main.java
Run  Main
C:\Users\91739\.jdks\openjdk-20.0.2\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.1.1\lib\idea_rt.jar=6325:C:\Pr
--- Digital 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 Asset
9. Exit
Enter the operation to perform: 3
Enter the Asset Id to delete: 13
Asset Deleted Successfully
--- Digital 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 Asset
9. Exit
Enter the operation to perform:
AssetManagementSystem > src > main > java > app > Main > deleteAsset   25:33  CRLF  UTF-8  4 spaces
```

```
AssetManagementSystem  main                                  Main
Main.java
Run  Main
C:\Users\91739\.jdks\openjdk-20.0.2\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.1.1\lib\idea_rt.jar=6329:C:\Pr
--- Digital 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 Asset
9. Exit
Enter the operation to perform: 4
Enter the Asset Id to Allocate Asset: 5
Enter the Employee Id to Allocate Asset: 4
Enter the Date to Allocate Asset: 2024-1-4
Asset Allocated Successfully
--- Digital 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 Asset
9. Exit
Enter the operation to perform:
AssetManagementSystem > src > main > java > app > Main > deleteAsset   27:33  CRLF  UTF-8  4 spaces
```

Exception is thrown If Asset Id or Employee Id is not Found

```
C:\Users\91739\.jdks\openjdk-20.0.2\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.1.1\lib\idea_rt.jar=6334:C:\Pr
--- Digital 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 Asset
9. Exit
Enter the operation to perform: 4
Enter the Asset Id to Allocate Asset: 4
Enter the Employee Id to Allocate Asset: 2
Enter the Date to Allocate Asset: 2024-11-4
Asset is not Allocated
Asset with id 4 is not found
--- Digital 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 Asset
9. Exit
```

Implemented Unit testing with the help of Junit library, Used Maven to add the library.

```java
public class AssetManagementTest {

    static AssetManagementService imp = null;


    @BeforeAll
    static void beforeAll() { imp = new AssetManagementServiceImpl(); }


    private Asset createAsset(int assetId, String name, String type, String serialNumber, String purchaseDate, String location, String status, int owner
        return new Asset(assetId, name, type, serialNumber, purchaseDate, location, status, ownerId);
    }


    @Test
    void testAddAsset() {
        Asset asset1 = createAsset( assetId: 201,  name: "HP Spectre x360",  type: "Laptop",  serialNumber: "HP1234XYZ",  purchaseDate: "2023-05-22",  location: "Boston
        Asset asset2 = createAsset( assetId: 202,  name: "Audi Q7",  type: "Vehicle",  serialNumber: "AUDI4567AB",  purchaseDate: "2021-11-05",  location: "Los Angeles"
        Asset asset3 = createAsset( assetId: 203,  name: "Epson L3150 Printer",  type: "Equipment",  serialNumber: "EPSON7890LM",  purchaseDate: "2022-04-13",  locatio
        Asset asset4 = createAsset( assetId: 204,  name: "Lenovo ThinkPad X1",  type: "Laptop",  serialNumber: "LENOVO1122XP",  purchaseDate: "2023-08-15",  location: "
        Asset asset5 = createAsset( assetId: 205,  name: "Toyota Corolla",  type: "Vehicle",  serialNumber: "TOYOTA7777XY",  purchaseDate: "2023-12-01",  location: "Dal

        assertTrue(imp.addAsset(asset1),  message: "Asset 1 (HP Spectre x360) should be added successfully");
        assertTrue(imp.addAsset(asset2),  message: "Asset 2 (Audi Q7) should be added successfully");
        assertTrue(imp.addAsset(asset3),  message: "Asset 3 (Epson L3150 Printer) should be added successfully");
        assertTrue(imp.addAsset(asset4),  message: "Asset 4 (Lenovo ThinkPad X1) should be added successfully");
        assertTrue(imp.addAsset(asset5),  message: "Asset 5 (Toyota Corolla) should be added successfully");
    }
```

```java
        assertTrue(imp.addAsset(asset1), message: "Asset 1 (HP Spectre x360) should be added successfully");
        assertTrue(imp.addAsset(asset2), message: "Asset 2 (Audi Q7) should be added successfully");
        assertTrue(imp.addAsset(asset3), message: "Asset 3 (Epson L3150 Printer) should be added successfully");
        assertTrue(imp.addAsset(asset4), message: "Asset 4 (Lenovo ThinkPad X1) should be added successfully");
        assertTrue(imp.addAsset(asset5), message: "Asset 5 (Toyota Corolla) should be added successfully");
    }


    @Test
    void performMaintenance() {

        assertTrue(imp.performMaintenance( assetId: 101, maintenanceDate: "2024-05-10", description: "Battery Replacement", cost: 150.00), message: "Maintenance f
        assertTrue(imp.performMaintenance( assetId: 102, maintenanceDate: "2023-11-25", description: "Tire Change", cost: 200.00), message: "Maintenance for asset
        assertTrue(imp.performMaintenance( assetId: 103, maintenanceDate: "2022-09-15", description: "Printer Cartridge Replacement", cost: 50.00), message: "Main
        assertTrue(imp.performMaintenance( assetId: 104, maintenanceDate: "2024-03-30", description: "Screen Repair", cost: 300.00), message: "Maintenance for ass
    }


    @Test
    void testReserve() {

        assertTrue(imp.reserveAsset( assetId: 1, employeeId: 1, reservationDate: "2024-11-25", startDate: "2024-11-27", endDate: "2024-11-29", status: "approved"),
        assertTrue(imp.reserveAsset( assetId: 2, employeeId: 2, reservationDate: "2024-11-26", startDate: "2024-11-28", endDate: "2024-11-30", status: "canceled"),
        assertTrue(imp.reserveAsset( assetId: 5, employeeId: 3, reservationDate: "2024-11-23", startDate: "2024-11-26", endDate: "2024-11-28", status: "approved"),
        assertTrue(imp.reserveAsset( assetId: 1, employeeId: 4, reservationDate: "2024-11-25", startDate: "2024-11-27", endDate: "2024-11-29", status: "approved"),
        assertTrue(imp.reserveAsset( assetId: 1, employeeId: 5, reservationDate: "2024-11-25", startDate: "2024-11-27", endDate: "2024-11-29", status: "pending"),
    }


    @Test
```

Test case check :

Passed

```
Tests passed: 1 of 1 test – 1 sec 136 ms
C:\Users\91739\.jdks\openjdk-20.0.2\bin\java.exe ...

Process finished with exit code 0
```

Let us see if it fails if the employee id is invalid and check

```
AssetManagementSystem  main                                   AssetManagementT...AddAsset
Main.java   AssetManagementTest.java

ing serialNumber, String purchaseDate, String location, String status, int ownerId) {
, location, status, ownerId);

, type: "Laptop", serialNumber: "LENOVO5678AB", purchaseDate: "2023-04-11", location: "Chicago", status: "In Use", ownerId: 800); // this emp id is not valid
hicle", serialNumber: "BMWX56789CD", purchaseDate: "2021-02-20", location: "Miami", status: "Decommissioned", ownerId: 8);
"Camera", serialNumber: "NIKON9876EF", purchaseDate: "2020-11-25", location: "Los Angeles", status: "Under Maintenance", ownerId: 9);
0", type: "Laptop", serialNumber: "DELL8765GH", purchaseDate: "2022-07-30", location: "Houston", status: "In Use", ownerId: 10);
"Bike", serialNumber: "YAMAHA0987IJ", purchaseDate: "2021-09-18", location: "Dallas", status: "In Use", ownerId: 11);

ad 5) should be added successfully");
ld be added successfully");
should be added successfully");
e 7420) should be added successfully");
hould be added successfully");

24-05-10", description: "Battery Replacement", cost: 150.00), message: "Maintenance for asset 101 should be successful");
23-11-25", description: "Tire Change", cost: 200.00), message: "Maintenance for asset 102 should be successful");
22-09-15", description: "Printer Cartridge Replacement", cost: 50.00), message: "Maintenance for asset 103 should be successful");
AssetManagementSystem > src > test > java > AssetManagementTest > testAddAsset        24:156  CRLF  UTF-8  4 spaces
```

```
AssetManagementSystem  main                                   AssetManagementT...AddAsset
Main.java   AssetManagementTest.java
Run   Main   AssetManagementTest.testAddAsset

Tests failed: 1 of 1 test – 1 sec 51 ms
AssetManagementTest 1 sec 51 ms
  testAddAsset()  1 sec 51 ms

C:\Users\91739\.jdks\openjdk-20.0.2\bin\java.exe ...
Employee with id 800 is not found

org.opentest4j.AssertionFailedError: Asset 1 (Lenovo IdeaPad 5) should be added successfully ==>
Expected :true
Actual   :false
<Click to see difference>

> <3 internal lines>
>    at AssetManagementTest.testAddAsset(AssetManagementTest.java:30) <29 internal lines>
>    at java.base/java.util.ArrayList.forEach(ArrayList.java:1511) <9 internal lines>
>    at java.base/java.util.ArrayList.forEach(ArrayList.java:1511) <25 internal lines>

Process finished with exit code -1
AssetManagementSystem > src > test > java > AssetManagementTest > testAddAsset        CRLF  UTF-8  4 spaces
```