Java Project

Asset Management System (AMS)

Objective :

The objective of the Asset Management System (AMS) is to efficiently manage and track organizational assets, including hardware and employee interactions. The system aims to provide accurate and timely information about asset availability, assignment, and overall lifecycle, ensuring effective utilization and management of resources.

1. Admin

- **Represents**: System administrators who manage the asset management system.
- **Attributes**:
  - `adminId`: A unique identifier for the admin.
  - `username`: The username of the admin.
  - `passwordHash`: A hashed version of the admin's password.
  - `fullName`: The full name of the admin.
  - `email`: The email address of the admin.
- **Role**: Manages system settings, user roles, and overall administration of the asset management system.

2. Employee

- **Represents**: Employees who use the asset management system.
- **Attributes**:
  - `employeeId`: A unique identifier for the employee.
  - `employeeName`: The name of the employee.
  - `email`: The email address of the employee.
  - `phoneNumber`: The phone number of the employee.
  - `designation`: The job title or position of the employee.
- **Role**: Tracks personal information, roles, and contact details of employees who will be assigned assets.

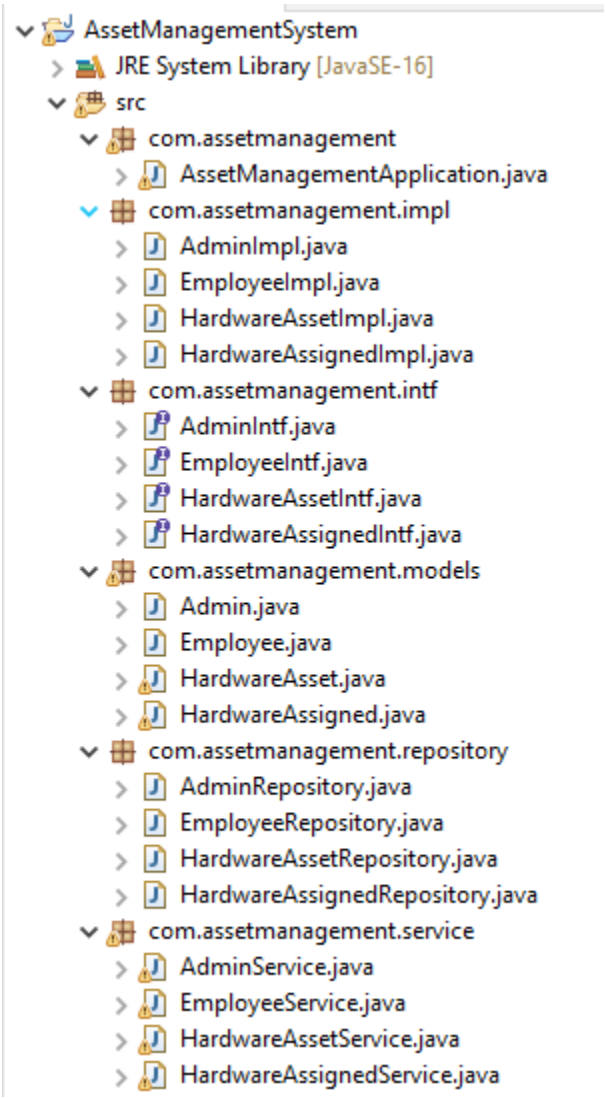
3. HardwareAsset

- **Represents**: Physical or digital assets managed by the system.
- **Attributes**:
  - `assetId`: A unique identifier for the hardware asset.
  - `assetName`: The name of the asset.
  - `description`: A detailed description of the asset.
  - `purchaseDate`: The date when the asset was purchased.
  - `status`: The current status of the asset (e.g., Available, In Use, Under Maintenance).
- **Role**: Manages the details and status of each asset within the system.

4. HardwareAssigned

- **Represents**: The relationship between hardware assets and employees.
- **Attributes**:
  - `assignmentId`: A unique identifier for the hardware assignment.
  - `employeeId`: The unique identifier of the employee to whom the asset is assigned.
  - `assetId`: The unique identifier of the asset assigned.
  - `assignmentDate`: The date when the asset was assigned to the employee.
  - `returnDate`: The date when the asset was returned (if applicable).
- **Role**: Tracks which assets are assigned to which employees and manages the status of these assignments.

Project Structure:



AMS Implementation :

AssetManagementApplication :

```
package com.assetmanagement;

import com.assetmanagement.repository.*;
import com.assetmanagement.service.AdminService;
import com.assetmanagement.service.EmployeeService;
import com.assetmanagement.service.HardwareAssetService;
import com.assetmanagement.service.HardwareAssignedService;

import java.util.InputMismatchException;
import java.util.Scanner;

public class AssetManagementApplication {

    private static AdminService admin;
    private static EmployeeService employee;
    private static HardwareAssetService hardware;
    private static HardwareAssignedService assigned;

    private static Scanner scanner = new Scanner(System.in);

    public static void main(String[] args) {

          EmployeeRepository employeeRepo = new EmployeeRepository();
        AdminRepository adminRepo = new AdminRepository();
        HardwareAssetRepository hardwareAssetRepo = new
HardwareAssetRepository();
        HardwareAssignedRepository hardwareAssignedRepo = new
HardwareAssignedRepository();

        employee = new EmployeeService(employeeRepo);
        admin = new AdminService(adminRepo);
        hardware = new HardwareAssetService(hardwareAssetRepo);
        assigned = new HardwareAssignedService(hardwareAssignedRepo);
```

```
    boolean running = true;
    while (running) {
        System.out.println("Asset Management System");
        System.out.println("1. Admin Operations");
        System.out.println("2. Hardware Asset Operations");
        System.out.println("3. Employee Operations");
        System.out.println("4. Hardware Assigned Operations");
        System.out.println("5. Exit");

        System.out.print("Choose an option: ");
        int choice = getValidInt();

        switch (choice) {
            case 1:
                handleAdminOperations();
                break;
            case 2:
                handleHardwareAssetOperations();
                break;
            case 3:
                handleEmployeeOperations();
                break;
            case 4:
                handleHardwareAssignedOperations();
                break;
            case 5:
                running = false;
                System.out.println("Exiting...");
                break;
            default:
                System.out.println("Invalid option, please try again.");
        }
    }
}

private static void handleAdminOperations() {
```

```
        System.out.println("Admin Operations");
        System.out.println("1. Add Admin");
        System.out.println("2. Remove Admin");
        System.out.println("3. Update Admin");
        System.out.println("4. Get Admin");
        System.out.println("5. List All Admins");
        System.out.println("6. Back to Main Menu");

        int choice = getValidInt();
        switch (choice) {
            case 1:
                admin.addAdmin();
                break;
            case 2:
                admin.removeAdmin();
                break;
            case 3:
                admin.updateAdmin();
                break;
            case 4:
                admin.getAdmin();
                break;
            case 5:
                admin.listAllAdmins();
                break;
            case 6:
                return;
            default:
                System.out.println("Invalid option, please try again.");
        }
    }

    private static void handleHardwareAssetOperations() {
        System.out.println("Hardware Asset Operations");
        System.out.println("1. Add Hardware Asset");
        System.out.println("2. Remove Hardware Asset");
        System.out.println("3. Update Hardware Asset");
```

```
        System.out.println("4. Get Hardware Asset");
        System.out.println("5. List All Hardware Assets");
        System.out.println("6. Back to Main Menu");

        int choice = getValidInt();
        switch (choice) {
            case 1:
                hardware.addHardwareAsset();
                break;
            case 2:
                hardware.removeHardwareAsset();
                break;
            case 3:
                hardware.updateHardwareAsset();
                break;
            case 4:
                hardware.getHardwareAsset();
                break;
            case 5:
                hardware.listAllHardwareAssets();
                break;
            case 6:
                return;
            default:
                System.out.println("Invalid option, please try again.");
        }
    }

    private static void handleEmployeeOperations() {
        System.out.println("Employee Operations");
        System.out.println("1. Add Employee");
        System.out.println("2. Remove Employee");
        System.out.println("3. Update Employee");
        System.out.println("4. Get Employee");
        System.out.println("5. List All Employees");
        System.out.println("6. Back to Main Menu");
```

```
        int choice = getValidInt();
        switch (choice) {
            case 1:
                employee.addEmployee();
                break;
            case 2:
                employee.removeEmployee();
                break;
            case 3:
                employee.updateEmployee();
                break;
            case 4:
                employee.getEmployee();
                break;
            case 5:
                employee.listAllEmployees();
                break;
            case 6:
                return;
            default:
                System.out.println("Invalid option, please try again.");
        }
    }

    private static void handleHardwareAssignedOperations() {
        System.out.println("Hardware Assigned Operations");
        System.out.println("1. Add Hardware Assigned");
        System.out.println("2. Remove Hardware Assigned");
        System.out.println("3. Update Hardware Assigned");
        System.out.println("4. Get Hardware Assigned");
        System.out.println("5. List All Hardware Assigned");
        System.out.println("6. Back to Main Menu");

        int choice = getValidInt();
        switch (choice) {
            case 1:
                assigned.addHardwareAssigned();
```

```
                break;
            case 2:
                assigned.removeHardwareAssigned();
                break;
            case 3:
                assigned.updateHardwareAssigned();
                break;
            case 4:
                assigned.getHardwareAssigned();
                break;
            case 5:
                assigned.listAllHardwareAssigned();
                break;
            case 6:
                return;
            default:
                System.out.println("Invalid option, please try again.");
        }
    }

    private static int getValidInt() {
        while (true) {
            try {
                return scanner.nextInt();
            } catch (InputMismatchException e) {
                System.out.println("Invalid input, please enter an integer.");
                scanner.next(); // clear the invalid input
            }
        }
    }
}
```

AdminImpl :

```
package com.assetmanagement.impl;

import com.assetmanagement.models.Admin;
import com.assetmanagement.repository.AdminRepository;
```

```
import com.assetmanagement.intf.AdminIntf;

import java.util.List;

public class AdminImpl implements AdminIntf {
    private AdminRepository adminRepo;

    public AdminImpl(AdminRepository adminRepo) {
        this.adminRepo = adminRepo;
    }

    @Override
    public void addAdmin(Admin admin) {
        adminRepo.addAdmin(admin);
    }

    @Override
    public void removeAdmin(int adminID) {
        adminRepo.removeAdmin(adminID);
    }

    @Override
    public void updateAdmin(Admin admin) {
        adminRepo.updateAdmin(admin);
    }

    @Override
    public Admin getAdmin(int adminID) {
        return adminRepo.getAdmin(adminID);
    }

    @Override
    public List<Admin> getAllAdmins() {
        return adminRepo.getAllAdmins();
    }
}
```

EmployeeImpl :

```
package com.assetmanagement.impl;

import com.assetmanagement.models.Employee;
import com.assetmanagement.repository.EmployeeRepository;
import com.assetmanagement.intf.EmployeeIntf;

import java.util.List;

public class EmployeeImpl implements EmployeeIntf {
    private EmployeeRepository employeeRepo;

    public EmployeeImpl(EmployeeRepository employeeRepo) {
        this.employeeRepo = employeeRepo;
    }

    @Override
    public void addEmployee(Employee employee) {
        employeeRepo.addEmployee(employee);
    }

    @Override
    public void removeEmployee(int employeeID) {
        employeeRepo.removeEmployee(employeeID);
    }

    @Override
    public void updateEmployee(Employee employee) {
        employeeRepo.updateEmployee(employee);
    }

    @Override
    public Employee getEmployee(int employeeID) {
        return employeeRepo.getEmployee(employeeID);
    }

    @Override
```

```
    public List<Employee> getAllEmployees() {
        return employeeRepo.getAllEmployees();
    }
}
```

HardwareAssetImpl :

```
package com.assetmanagement.impl;

import com.assetmanagement.models.HardwareAsset;
import com.assetmanagement.repository.HardwareAssetRepository;
import com.assetmanagement.intf.HardwareAssetIntf;

import java.util.List;

public class HardwareAssetImpl implements HardwareAssetIntf {
    private HardwareAssetRepository hardwareAssetRepo;

    public HardwareAssetImpl(HardwareAssetRepository hardwareAssetRepo) {
        this.hardwareAssetRepo = hardwareAssetRepo;
    }

    @Override
    public void addHardwareAsset(HardwareAsset asset) {
        hardwareAssetRepo.addHardwareAsset(asset);
    }

    @Override
    public void removeHardwareAsset(int assetID) {
        hardwareAssetRepo.removeHardwareAsset(assetID);
    }

    @Override
    public void updateHardwareAsset(HardwareAsset asset) {
        hardwareAssetRepo.updateHardwareAsset(asset);
    }

    @Override
```

```
    public HardwareAsset getHardwareAsset(int assetID) {
        return hardwareAssetRepo.getHardwareAsset(assetID);
    }

    @Override
    public List<HardwareAsset> getAllHardwareAssets() {
        return hardwareAssetRepo.getAllHardwareAssets();
    }
}
```

HardwareAssignedImpl :

```
package com.assetmanagement.impl;

import com.assetmanagement.models.HardwareAssigned;
import com.assetmanagement.repository.HardwareAssignedRepository;
import com.assetmanagement.intf.HardwareAssignedIntf;

import java.util.List;

public class HardwareAssignedImpl implements HardwareAssignedIntf {
    private HardwareAssignedRepository hardwareAssignedRepo;

    public HardwareAssignedImpl(HardwareAssignedRepository
hardwareAssignedRepo) {
        this.hardwareAssignedRepo = hardwareAssignedRepo;
    }

    @Override
    public void addHardwareAssigned(HardwareAssigned assigned) {
        hardwareAssignedRepo.addHardwareAssigned(assigned);
    }

    @Override
    public void removeHardwareAssigned(int assignedID) {
        hardwareAssignedRepo.removeHardwareAssigned(assignedID);
    }
```

```
    @Override
    public void updateHardwareAssigned(HardwareAssigned assigned) {
        hardwareAssignedRepo.updateHardwareAssigned(assigned);
    }

    @Override
    public HardwareAssigned getHardwareAssigned(int assignedID) {
        return hardwareAssignedRepo.getHardwareAssigned(assignedID);
    }

    @Override
    public List<HardwareAssigned> getAllHardwareAssigned() {
        return hardwareAssignedRepo.getAllHardwareAssigned();
    }
}
```

AdminIntf :

```
package com.assetmanagement.intf;

import com.assetmanagement.models.Admin;
import java.util.List;

public interface AdminIntf {
    void addAdmin(Admin admin);
    void removeAdmin(int adminID);
    void updateAdmin(Admin admin);
    Admin getAdmin(int adminID);
    List<Admin> getAllAdmins();
}
```

EmployeeIntf :

```
package com.assetmanagement.intf;

import com.assetmanagement.models.Employee;
import java.util.List;
```

```
public interface EmployeeIntf {
    void addEmployee(Employee employee);
    void removeEmployee(int employeeID);
    void updateEmployee(Employee employee);
    Employee getEmployee(int employeeID);
    List<Employee> getAllEmployees();
}
```

HardwareAssetIntf :

```
package com.assetmanagement.intf;

import com.assetmanagement.models.HardwareAsset;
import java.util.List;

public interface HardwareAssetIntf {
    void addHardwareAsset(HardwareAsset asset);
    void removeHardwareAsset(int assetID);
    void updateHardwareAsset(HardwareAsset asset);
    HardwareAsset getHardwareAsset(int assetID);
    List<HardwareAsset> getAllHardwareAssets();
}
```

HardwareAssignedIntf :

```
package com.assetmanagement.intf;

import com.assetmanagement.models.HardwareAssigned;
import java.util.List;

public interface HardwareAssignedIntf {
    void addHardwareAssigned(HardwareAssigned assigned);
    void removeHardwareAssigned(int assignedID);
    void updateHardwareAssigned(HardwareAssigned assigned);
    HardwareAssigned getHardwareAssigned(int assignedID);
    List<HardwareAssigned> getAllHardwareAssigned();
}
```

Admin :

```
package com.assetmanagement.models;

public class Admin {
    private int adminID;
    private String adminName;

    public Admin(int adminID, String adminName) {
        this.adminID = adminID;
        this.adminName = adminName;
    }

    public int getAdminID() {
        return adminID;
    }

    public void setAdminID(int adminID) {
        this.adminID = adminID;
    }

    public String getAdminName() {
        return adminName;
    }

    public void setAdminName(String adminName) {
        this.adminName = adminName;
    }

    @Override
    public String toString() {
        return "Admin{" +
                "adminID=" + adminID +
                ", adminName='" + adminName + '\'' +
                '}';
    }
}
```

Employee :

```
package com.assetmanagement.models;

public class Employee {
    private int employeeID;
    private String employeeName;
    private String email;
    private String phoneNumber;
    private String designation;

    public Employee(int employeeID, String employeeName, String email, String
phoneNumber, String designation) {
        this.employeeID = employeeID;
        this.employeeName = employeeName;
        this.email = email;
        this.phoneNumber = phoneNumber;
        this.designation = designation;
    }

    public int getEmployeeID() {
        return employeeID;
    }

    public void setEmployeeID(int employeeID) {
        this.employeeID = employeeID;
    }

    public String getEmployeeName() {
        return employeeName;
    }

    public void setEmployeeName(String employeeName) {
        this.employeeName = employeeName;
    }

    public String getEmail() {
        return email;
```

```
    }

    public void setEmail(String email) {
        this.email = email;
    }

    public String getPhoneNumber() {
        return phoneNumber;
    }

    public void setPhoneNumber(String phoneNumber) {
        this.phoneNumber = phoneNumber;
    }

    public String getDesignation() {
        return designation;
    }

    public void setDesignation(String designation) {
        this.designation = designation;
    }

    @Override
    public String toString() {
        return "Employee{" +
                "employeeID=" + employeeID +
                ", employeeName='" + employeeName + '\'' +
                ", email='" + email + '\'' +
                ", phoneNumber='" + phoneNumber + '\'' +
                ", designation='" + designation + '\'' +
                '}';
    }
}
```

HardwareAsset :

```
package com.assetmanagement.models;
```

```
import java.time.LocalDate;
import java.util.Date;

public class HardwareAsset {
    private int hardwareAssetID;
    private String assetName;
    private String assetType;
    private LocalDate purchaseDate;
    private double purchasePrice;
    private int warrantyPeriod;
    private String assetStatus;

    public HardwareAsset(int hardwareAssetID, String assetName, String
assetType, LocalDate purchaseDate, double purchasePrice, int warrantyPeriod,
String assetStatus) {
        this.hardwareAssetID = hardwareAssetID;
        this.assetName = assetName;
        this.assetType = assetType;
        this.purchaseDate = purchaseDate;
        this.purchasePrice = purchasePrice;
        this.warrantyPeriod = warrantyPeriod;
        this.assetStatus = assetStatus;
    }

    public int getHardwareAssetID() {
        return hardwareAssetID;
    }

    public void setHardwareAssetID(int hardwareAssetID) {
        this.hardwareAssetID = hardwareAssetID;
    }

    public String getAssetName() {
        return assetName;
    }

    public void setAssetName(String assetName) {
```

```
    this.assetName = assetName;
}

public String getAssetType() {
    return assetType;
}

public void setAssetType(String assetType) {
    this.assetType = assetType;
}

public LocalDate getPurchaseDate() {
    return purchaseDate;
}

public void setPurchaseDate(LocalDate purchaseDate) {
    this.purchaseDate = purchaseDate;
}

public double getPurchasePrice() {
    return purchasePrice;
}

public void setPurchasePrice(double purchasePrice) {
    this.purchasePrice = purchasePrice;
}

public int getWarrantyPeriod() {
    return warrantyPeriod;
}

public void setWarrantyPeriod(int warrantyPeriod) {
    this.warrantyPeriod = warrantyPeriod;
}

public String getAssetStatus() {
    return assetStatus;
```

```
    }

    public void setAssetStatus(String assetStatus) {
        this.assetStatus = assetStatus;
    }

    @Override
    public String toString() {
        return "HardwareAsset{" +
                "hardwareAssetID=" + hardwareAssetID +
                ", assetName='" + assetName + '\'' +
                ", assetType='" + assetType + '\'' +
                ", purchaseDate=" + purchaseDate +
                ", purchasePrice=" + purchasePrice +
                ", warrantyPeriod=" + warrantyPeriod +
                ", assetStatus='" + assetStatus + '\'' +
                '}';
    }
}
```

HardwareAssigned :

```
package com.assetmanagement.models;

import java.time.LocalDate;
import java.util.Date;

public class HardwareAssigned {
    private int hardwareAssignedID;
    private int hardwareAssetID;
    private int employeeID;
    private String assignedBy;
    private LocalDate assignedDate;
    private String assignedStatus;

    public HardwareAssigned(int hardwareAssignedID, int hardwareAssetID, int
employeeID, String assignedBy, LocalDate assignedDate, String assignedStatus)
{
```

```
    this.hardwareAssignedID = hardwareAssignedID;
    this.hardwareAssetID = hardwareAssetID;
    this.employeeID = employeeID;
    this.assignedBy = assignedBy;
    this.assignedDate = assignedDate;
    this.assignedStatus = assignedStatus;
}

public int getHardwareAssignedID() {
    return hardwareAssignedID;
}

public void setHardwareAssignedID(int hardwareAssignedID) {
    this.hardwareAssignedID = hardwareAssignedID;
}

public int getHardwareAssetID() {
    return hardwareAssetID;
}

public void setHardwareAssetID(int hardwareAssetID) {
    this.hardwareAssetID = hardwareAssetID;
}

public int getEmployeeID() {
    return employeeID;
}

public void setEmployeeID(int employeeID) {
    this.employeeID = employeeID;
}

public String getAssignedBy() {
    return assignedBy;
}

public void setAssignedBy(String assignedBy) {
```

```
        this.assignedBy = assignedBy;
    }

    public LocalDate getAssignedDate() {
        return assignedDate;
    }

    public void setAssignedDate(LocalDate assignedDate) {
        this.assignedDate = assignedDate;
    }

    public String getAssignedStatus() {
        return assignedStatus;
    }

    public void setAssignedStatus(String assignedStatus) {
        this.assignedStatus = assignedStatus;
    }

    @Override
    public String toString() {
        return "HardwareAssigned{" +
                "hardwareAssignedID=" + hardwareAssignedID +
                ", hardwareAssetID=" + hardwareAssetID +
                ", employeeID=" + employeeID +
                ", assignedBy='" + assignedBy + '\'' +
                ", assignedDate=" + assignedDate +
                ", assignedStatus='" + assignedStatus + '\'' +
                '}';
    }
}
```

AdminRepository :

```
package com.assetmanagement.repository;

import com.assetmanagement.models.Admin;
```

```
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

public class AdminRepository {
    private Map<Integer, Admin> adminMap = new HashMap<>();

    public void addAdmin(Admin admin) {
        adminMap.put(admin.getAdminID(), admin);
    }

    public void removeAdmin(int adminID) {
        adminMap.remove(adminID);
    }

    public void updateAdmin(Admin admin) {
        adminMap.put(admin.getAdminID(), admin);
    }

    public Admin getAdmin(int adminID) {
        return adminMap.get(adminID);
    }

    public List<Admin> getAllAdmins() {
        return new ArrayList<>(adminMap.values());
    }
}
```

EmployeeRepository :

```
package com.assetmanagement.repository;

import com.assetmanagement.models.Employee;

import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
```

```
import java.util.Map;

public class EmployeeRepository {
    private Map<Integer, Employee> employeeMap = new HashMap<>();

    public void addEmployee(Employee employee) {
        employeeMap.put(employee.getEmployeeID(), employee);
    }

    public void removeEmployee(int employeeID) {
        employeeMap.remove(employeeID);
    }

    public void updateEmployee(Employee employee) {
        employeeMap.put(employee.getEmployeeID(), employee);
    }

    public Employee getEmployee(int employeeID) {
        return employeeMap.get(employeeID);
    }

    public List<Employee> getAllEmployees() {
        return new ArrayList<>(employeeMap.values());
    }
}
```

HardwareAssetRepository :

```
package com.assetmanagement.repository;

import com.assetmanagement.models.HardwareAsset;

import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

public class HardwareAssetRepository {
```

```
    private Map<Integer, HardwareAsset> hardwareAssetMap = new
HashMap<>();

    public void addHardwareAsset(HardwareAsset asset) {
        hardwareAssetMap.put(asset.getHardwareAssetID(), asset);
    }

    public void removeHardwareAsset(int assetID) {
        hardwareAssetMap.remove(assetID);
    }

    public void updateHardwareAsset(HardwareAsset asset) {
        hardwareAssetMap.put(asset.getHardwareAssetID(), asset);
    }

    public HardwareAsset getHardwareAsset(int assetID) {
        return hardwareAssetMap.get(assetID);
    }

    public List<HardwareAsset> getAllHardwareAssets() {
        return new ArrayList<>(hardwareAssetMap.values());
    }
}
```

HardwareAssignedRepository :

```
package com.assetmanagement.repository;

import com.assetmanagement.models.HardwareAssigned;

import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

public class HardwareAssignedRepository {
    private Map<Integer, HardwareAssigned> hardwareAssignedMap = new
HashMap<>();
```

```
    public void addHardwareAssigned(HardwareAssigned assigned) {
        hardwareAssignedMap.put(assigned.getHardwareAssignedID(), assigned);
    }

    public void removeHardwareAssigned(int assignedID) {
        hardwareAssignedMap.remove(assignedID);
    }

    public void updateHardwareAssigned(HardwareAssigned assigned) {
        hardwareAssignedMap.put(assigned.getHardwareAssignedID(), assigned);
    }

    public HardwareAssigned getHardwareAssigned(int assignedID) {
        return hardwareAssignedMap.get(assignedID);
    }

    public List<HardwareAssigned> getAllHardwareAssigned() {
        return new ArrayList<>(hardwareAssignedMap.values());
    }
}
```

AdminService :

```
package com.assetmanagement.service;

import com.assetmanagement.impl.AdminImpl;
import com.assetmanagement.models.Admin;
import com.assetmanagement.repository.AdminRepository;

import java.util.*;

public class AdminService {

        private static AdminImpl adminImpl;
        private static Scanner scanner = new Scanner(System.in);

        public AdminService(AdminRepository adminRepo) {
```

```
        this.adminImpl = new AdminImpl(adminRepo);
    }

          public static void addAdmin() {
        System.out.print("Enter Admin ID: ");
        int id = getValidInt();
        System.out.print("Enter Admin Name: ");
        String name = scanner.next();
        adminImpl.addAdmin(new Admin(id, name));
        System.out.println("Admin added successfully.");
    }

          public static void removeAdmin() {
        System.out.print("Enter Admin ID to remove: ");
        int id = getValidInt();
        adminImpl.removeAdmin(id);
        System.out.println("Admin removed successfully.");
    }

    public static void updateAdmin() {
        System.out.print("Enter Admin ID to update: ");
        int id = getValidInt();
        System.out.print("Enter new Admin Name: ");
        String name = scanner.next();
        adminImpl.updateAdmin(new Admin(id, name));
        System.out.println("Admin updated successfully.");
    }

    public static void getAdmin() {
        System.out.print("Enter Admin ID to retrieve: ");
        int id = getValidInt();
        Admin admin = adminImpl.getAdmin(id);
        if (admin != null) {
            System.out.printf("%-10s %-20s%n", "ID", "Name");
            System.out.println("----------------------------");
            System.out.printf("%-10d %-20s%n", admin.getAdminID(),
admin.getAdminName());
```

```
        } else {
            System.out.println("Admin not found.");
        }
    }

    public static void listAllAdmins() {
        List<Admin> admins = adminImpl.getAllAdmins();
        // Print table header
        System.out.printf("%-10s %-20s%n", "ID", "Name");
        System.out.println("----------------------------");
        // Print each record in the table
        if (!admins.isEmpty()) {
            for (Admin admin : admins) {
                System.out.printf("%-10d %-20s%n", admin.getAdminID(),
admin.getAdminName());
            }
        } else {
            System.out.println("No admins to display.");
        }
    }

    public static int getValidInt() {
        while (true) {
            try {
                return scanner.nextInt();
            } catch (InputMismatchException e) {
                System.out.println("Invalid input, please enter an integer.");
                scanner.next(); // clear the invalid input
            }
        }
    }

}
```

EmployeeService :

```
package com.assetmanagement.service;
```

```
import com.assetmanagement.impl.EmployeeImpl;
import com.assetmanagement.models.Employee;
import com.assetmanagement.repository.EmployeeRepository;

import java.util.*;

public class EmployeeService {

        private static EmployeeImpl employeeImpl;
        private static Scanner scanner = new Scanner(System.in);


        public EmployeeService(EmployeeRepository employeeRepo) {
       this.employeeImpl = new EmployeeImpl(employeeRepo);
   }


        public static void addEmployee() {
                System.out.print("Enter Employee ID: ");
                int id = getValidInt();
                System.out.print("Enter Employee Name: ");
                String name = scanner.next();
                System.out.print("Enter Email: ");
                String email = scanner.next();
                System.out.print("Enter Phone Number: ");
                String phone = scanner.next();
                System.out.print("Enter Designation: ");
                String designation = scanner.next();
                employeeImpl.addEmployee(new Employee(id, name, email, phone,
designation));
                System.out.println("Employee added successfully.");
            }

        public static void removeEmployee() {
                System.out.print("Enter Employee ID to remove: ");
                int id = getValidInt();
                employeeImpl.removeEmployee(id);
```

```
            System.out.println("Employee removed successfully.");
        }

    public static void updateEmployee() {
            System.out.print("Enter Employee ID to update: ");
            int id = getValidInt();
            System.out.print("Enter new Employee Name: ");
            String name = scanner.next();
            System.out.print("Enter new Email: ");
            String email = scanner.next();
            System.out.print("Enter new Phone Number: ");
            String phone = scanner.next();
            System.out.print("Enter new Designation: ");
            String designation = scanner.next();
            employeeImpl.updateEmployee(new Employee(id, name, email,
phone, designation));
            System.out.println("Employee updated successfully.");
        }

    public static void getEmployee() {
            System.out.print("Enter Employee ID to retrieve: ");
            int id = getValidInt();
            Employee employee = employeeImpl.getEmployee(id);
            if (employee != null) {
                System.out.printf("%-10s %-20s %-25s %-15s %-20s%n",
                        "ID", "Name", "Email", "Phone", "Designation");

System.out.println("----------------------------------------------------------------------");
                System.out.printf("%-10d %-20s %-25s %-15s %-20s%n",
                        employee.getEmployeeID(),
                        employee.getEmployeeName(),
                        employee.getEmail(),
                        employee.getPhoneNumber(),
                        employee.getDesignation());
            } else {
                System.out.println("Employee not found.");
            }
```

```
        }

    public static void listAllEmployees() {
        List<Employee> employees = employeeImpl.getAllEmployees();
        // Print table header
        System.out.printf("%-10s %-20s %-25s %-15s %-20s%n",
                "ID", "Name", "Email", "Phone", "Designation");

System.out.println("--------------------------------------------------------------------------------");
        // Print each record in the table
        if (!employees.isEmpty()) {
            for (Employee employee : employees) {
                System.out.printf("%-10d %-20s %-25s %-15s %-20s%n",
                        employee.getEmployeeID(),
                        employee.getEmployeeName(),
                        employee.getEmail(),
                        employee.getPhoneNumber(),
                        employee.getDesignation());
            }
        } else {
            System.out.println("No employees to display.");
        }
    }

    public static int getValidInt() {
        while (true) {
            try {
                return scanner.nextInt();
            } catch (InputMismatchException e) {
                System.out.println("Invalid input, please enter an integer.");
                scanner.next(); // clear the invalid input
            }
        }
    }

}
```

HardwareAssetService :

```
package com.assetmanagement.service;

import com.assetmanagement.impl.HardwareAssetImpl;
import com.assetmanagement.models.HardwareAsset;
import com.assetmanagement.repository.HardwareAssetRepository;

import java.time.LocalDate;
import java.util.*;

public class HardwareAssetService {

        private static HardwareAssetImpl hardwareAssetImpl;
        private static Scanner scanner = new Scanner(System.in);

        public HardwareAssetService(HardwareAssetRepository
HardwareAssetRepo) {
              this.hardwareAssetImpl = new
HardwareAssetImpl(HardwareAssetRepo);
          }

        public static void addHardwareAsset() {
      System.out.print("Enter Hardware Asset ID: ");
      int id = getValidInt();
      System.out.print("Enter Asset Name: ");
      String name = scanner.next();
      System.out.print("Enter Asset Type: ");
      String type = scanner.next();
      System.out.print("Enter Purchase Date (YYYY-MM-DD): ");
      String[] parts = scanner.next().split("-");
      LocalDate date = LocalDate.of(Integer.parseInt(parts[0]),
Integer.parseInt(parts[1]), Integer.parseInt(parts[2]));
      System.out.print("Enter Purchase Price: ");
      double price = scanner.nextDouble();
      System.out.print("Enter Warranty Period (months): ");
      int warranty = getValidInt();
      System.out.print("Enter Asset Status: ");
```

```
        String status = scanner.next();
        hardwareAssetImpl.addHardwareAsset(new HardwareAsset(id, name, type,
date, price, warranty, status));
        System.out.println("Hardware asset added successfully.");
    }

        public static void removeHardwareAsset() {
        System.out.print("Enter Hardware Asset ID to remove: ");
        int id = getValidInt();
        hardwareAssetImpl.removeHardwareAsset(id);
        System.out.println("Hardware asset removed successfully.");
    }

        public static void updateHardwareAsset() {
        System.out.print("Enter Hardware Asset ID to update: ");
        int id = getValidInt();
        System.out.print("Enter new Asset Name: ");
        String name = scanner.next();
        System.out.print("Enter new Asset Type: ");
        String type = scanner.next();
        System.out.print("Enter Purchase Date (YYYY-MM-DD): ");
        String[] parts = scanner.next().split("-");
        LocalDate date = LocalDate.of(Integer.parseInt(parts[0]),
Integer.parseInt(parts[1]), Integer.parseInt(parts[2]));
        System.out.print("Enter new Purchase Price: ");
        double price = scanner.nextDouble();
        System.out.print("Enter new Warranty Period (months): ");
        int warranty = getValidInt();
        System.out.print("Enter new Asset Status: ");
        String status = scanner.next();
        hardwareAssetImpl.updateHardwareAsset(new HardwareAsset(id, name,
type, date, price, warranty, status));
        System.out.println("Hardware asset updated successfully.");
    }

        public static void getHardwareAsset() {
        System.out.print("Enter Hardware Asset ID to retrieve: ");
```

```
        int id = getValidInt();
        HardwareAsset asset = hardwareAssetImpl.getHardwareAsset(id);
        if (asset != null) {
            System.out.printf("%-10s %-20s %-15s %-15s %-15s %-20s %-10s%n",
                              "ID", "Name", "Type", "Purchase Date", "Price", "Warranty
(months)", "Status");

System.out.println("----------------------------------------------------------------------------
----------------");
            System.out.printf("%-10d %-20s %-15s %-15s %-15.2f %-20d %-10s%n",
                              asset.getHardwareAssetID(),
                              asset.getAssetName(),
                              asset.getAssetType(),
                              asset.getPurchaseDate(),
                              asset.getPurchasePrice(),
                              asset.getWarrantyPeriod(),
                              asset.getAssetStatus());
        } else {
            System.out.println("Hardware asset not found.");
        }
    }

       public static void listAllHardwareAssets() {
        List<HardwareAsset> assets = hardwareAssetImpl.getAllHardwareAssets();
        // Print table header
        System.out.printf("%-10s %-20s %-15s %-15s %-15s %-20s %-10s%n",
                          "ID", "Name", "Type", "Purchase Date", "Price", "Warranty
(months)", "Status");

System.out.println("----------------------------------------------------------------------------
----------------");
        // Print each record in the table
        if (!assets.isEmpty()) {
            for (HardwareAsset asset : assets) {
                System.out.printf("%-10d %-20s %-15s %-15s %-15.2f %-20d
%-10s%n",
                                  asset.getHardwareAssetID(),
```

```
                        asset.getAssetName(),
                        asset.getAssetType(),
                        asset.getPurchaseDate(),
                        asset.getPurchasePrice(),
                        asset.getWarrantyPeriod(),
                        asset.getAssetStatus());
            }
        } else {
            System.out.println("No hardware assets to display.");
        }
    }

        public static int getValidInt() {
        while (true) {
            try {
                return scanner.nextInt();
            } catch (InputMismatchException e) {
                System.out.println("Invalid input, please enter an integer.");
                scanner.next(); // clear the invalid input
            }
        }
    }

}
```

HardwareAssignedService :

```
package com.assetmanagement.service;

import com.assetmanagement.impl.HardwareAssignedImpl;
import com.assetmanagement.models.HardwareAssigned;
import com.assetmanagement.repository.HardwareAssignedRepository;

import java.time.LocalDate;
import java.util.*;

public class HardwareAssignedService {
```

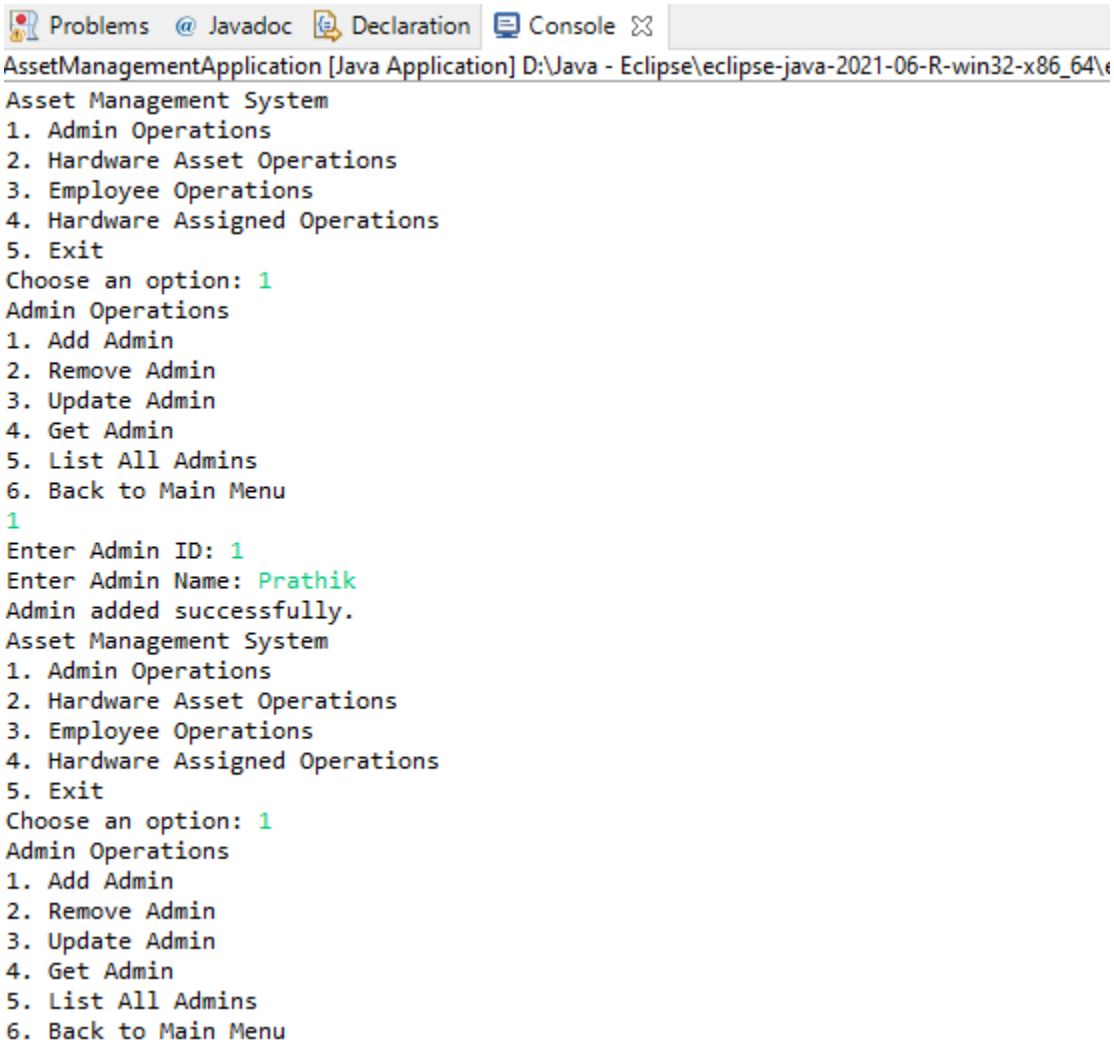
```
        private static HardwareAssignedImpl hardwareAssignedImpl;
          private static Scanner scanner = new Scanner(System.in);

          public HardwareAssignedService(HardwareAssignedRepository
HardwareAssignedRepo) {
              this.hardwareAssignedImpl = new
HardwareAssignedImpl(HardwareAssignedRepo);
          }

          public static int getValidInt() {
      while (true) {
         try {
            return scanner.nextInt();
         } catch (InputMismatchException e) {
            System.out.println("Invalid input, please enter an integer.");
            scanner.next(); // clear the invalid input
         }
      }
   }

          public static void addHardwareAssigned() {
      System.out.print("Enter Hardware Assigned ID: ");
      int id = getValidInt();
      System.out.print("Enter Hardware Asset ID: ");
      int assetID = getValidInt();
      System.out.print("Enter Employee ID: ");
      int employeeID = getValidInt();
      System.out.print("Enter Assigned By: ");
      String assignedBy = scanner.next();
      System.out.print("Enter Assigned Date (YYYY-MM-DD): ");
      String[] parts = scanner.next().split("-");
      LocalDate date = LocalDate.of(Integer.parseInt(parts[0]),
Integer.parseInt(parts[1]), Integer.parseInt(parts[2]));
      System.out.print("Enter Assigned Status: ");
      String status = scanner.next();
      hardwareAssignedImpl.addHardwareAssigned(new HardwareAssigned(id,
assetID, employeeID, assignedBy, date, status));
```

```
        System.out.println("Hardware assigned added successfully.");
    }

        public static void removeHardwareAssigned() {
        System.out.print("Enter Hardware Assigned ID to remove: ");
        int id = getValidInt();
        hardwareAssignedImpl.removeHardwareAssigned(id);
        System.out.println("Hardware assigned removed successfully.");
    }

        public static void updateHardwareAssigned() {
        System.out.print("Enter Hardware Assigned ID to update: ");
        int id = getValidInt();
        System.out.print("Enter new Hardware Asset ID: ");
        int assetID = getValidInt();
        System.out.print("Enter new Employee ID: ");
        int employeeID = getValidInt();
        System.out.print("Enter new Assigned By: ");
        String assignedBy = scanner.next();
        System.out.print("Enter new Assigned Date (YYYY-MM-DD): ");
        String[] parts = scanner.next().split("-");
        LocalDate date = LocalDate.of(Integer.parseInt(parts[0]),
Integer.parseInt(parts[1]), Integer.parseInt(parts[2]));
        System.out.print("Enter new Assigned Status: ");
        String status = scanner.next();
        hardwareAssignedImpl.updateHardwareAssigned(new
HardwareAssigned(id, assetID, employeeID, assignedBy, date, status));
        System.out.println("Hardware assigned updated successfully.");
    }

        public static void getHardwareAssigned() {
        System.out.print("Enter Hardware Assigned ID to retrieve: ");
        int id = getValidInt();
        HardwareAssigned assigned =
hardwareAssignedImpl.getHardwareAssigned(id);
        if (assigned != null) {
            System.out.printf("%-10s %-15s %-15s %-20s %-20s %-15s%n",
```

```
                        "ID", "Asset ID", "Employee ID", "Assigned By", "Assigned
Date", "Status");

System.out.println("----------------------------------------------------------------------------
------");
            System.out.printf("%-10d %-15d %-15d %-20s %-20s %-15s%n",
                        assigned.getHardwareAssignedID(),
                        assigned.getHardwareAssetID(),
                        assigned.getEmployeeID(),
                        assigned.getAssignedBy(),
                        assigned.getAssignedDate(),
                        assigned.getAssignedStatus());
        } else {
            System.out.println("Hardware assigned not found.");
        }
    }

             public static void listAllHardwareAssigned() {
        List<HardwareAssigned> assignedList =
hardwareAssignedImpl.getAllHardwareAssigned();
        // Print table header
        System.out.printf("%-10s %-15s %-15s %-20s %-20s %-15s%n",
                        "ID", "Asset ID", "Employee ID", "Assigned By", "Assigned Date",
"Status");

System.out.println("----------------------------------------------------------------------------
------");
        // Print each record in the table
        if (!assignedList.isEmpty()) {
            for (HardwareAssigned assigned : assignedList) {
                System.out.printf("%-10d %-15d %-15d %-20s %-20s %-15s%n",
                            assigned.getHardwareAssignedID(),
                            assigned.getHardwareAssetID(),
                            assigned.getEmployeeID(),
                            assigned.getAssignedBy(),
                            assigned.getAssignedDate(),
                            assigned.getAssignedStatus());
```

```
            }
        } else {
            System.out.println("No hardware assignments to display.");
        }
    }

}
```

Results :



```
3
Enter Admin ID to update: 1
Enter new Admin Name: Balaji
Admin updated successfully.
Asset Management System
1. Admin Operations
2. Hardware Asset Operations
3. Employee Operations
4. Hardware Assigned Operations
5. Exit
Choose an option: 1
Admin Operations
1. Add Admin
2. Remove Admin
3. Update Admin
4. Get Admin
5. List All Admins
6. Back to Main Menu
4
Enter Admin ID to retrieve: 1
ID          Name
------------------------------
1           Balaji
Asset Management System
1. Admin Operations
2. Hardware Asset Operations
3. Employee Operations
4. Hardware Assigned Operations
5. Exit
Choose an option: 1
Admin Operations
1. Add Admin
2. Remove Admin
3. Update Admin
4. Get Admin
5. List All Admins
6. Back to Main Menu

5
ID          Name
------------------------------
1           Balaji
Asset Management System
1. Admin Operations
2. Hardware Asset Operations
3. Employee Operations
4. Hardware Assigned Operations
5. Exit
Choose an option: 2
Hardware Asset Operations
1. Add Hardware Asset
2. Remove Hardware Asset
3. Update Hardware Asset
4. Get Hardware Asset
5. List All Hardware Assets
6. Back to Main Menu
```

```
1
Enter Hardware Asset ID: 1
Enter Asset Name: Laptop
Enter Asset Type: Hardware
Enter Purchase Date (YYYY-MM-DD): 2024-08-13
Enter Purchase Price: 15000
Enter Warranty Period (months): 12
Enter Asset Status: Active
Hardware asset added successfully.
Asset Management System
1. Admin Operations
2. Hardware Asset Operations
3. Employee Operations
4. Hardware Assigned Operations
5. Exit
Choose an option: 2
Hardware Asset Operations
1. Add Hardware Asset
2. Remove Hardware Asset
3. Update Hardware Asset
4. Get Hardware Asset
5. List All Hardware Assets
6. Back to Main Menu
5
ID         Name                Type            Purchase Date   Price           Warranty (months)    Status
-------------------------------------------------------------------------------------------------------
1          Laptop              Hardware        2024-08-13      15000.00        12                   Active
```

```
Asset Management System
1. Admin Operations
2. Hardware Asset Operations
3. Employee Operations
4. Hardware Assigned Operations
5. Exit
Choose an option: 3
Employee Operations
1. Add Employee
2. Remove Employee
3. Update Employee
4. Get Employee
5. List All Employees
6. Back to Main Menu
1
Enter Employee ID: 1
Enter Employee Name: Prathik
Enter Email: prathik.b@payoda.com
Enter Phone Number: 9876543210
Enter Designation: Software Engineer
Employee added successfully.
Asset Management System
1. Admin Operations
2. Hardware Asset Operations
3. Employee Operations
4. Hardware Assigned Operations
5. Exit
Choose an option: 3
Employee Operations
1. Add Employee
2. Remove Employee
3. Update Employee
4. Get Employee
5. List All Employees
6. Back to Main Menu
```

```
3
Enter Employee ID to update: Invalid input, please enter an integer.
1
Enter new Employee Name: Prathik
Enter new Email: prathik.b@gmail.com
Enter new Phone Number: 876543234
Enter new Designation: Software Tester
Employee updated successfully.
Asset Management System
1. Admin Operations
2. Hardware Asset Operations
3. Employee Operations
4. Hardware Assigned Operations
5. Exit
```

```
Asset Management System
1. Admin Operations
2. Hardware Asset Operations
3. Employee Operations
4. Hardware Assigned Operations
5. Exit
Choose an option: 3
Employee Operations
1. Add Employee
2. Remove Employee
3. Update Employee
4. Get Employee
5. List All Employees
6. Back to Main Menu
1
Enter Employee ID: 1
Enter Employee Name: Prathik
Enter Email: prathik.b@gmil.com
Enter Phone Number: 9876543210
Enter Designation: Software Developer
Employee added successfully.
Asset Management System
1. Admin Operations
2. Hardware Asset Operations
3. Employee Operations
4. Hardware Assigned Operations
5. Exit
Choose an option: 3
Employee Operations
1. Add Employee
2. Remove Employee
3. Update Employee
4. Get Employee
5. List All Employees
6. Back to Main Menu
```

```
1
Enter Employee ID: 2
Enter Employee Name: Sanjith S
Enter Email: sanjith@gmail.com
Enter Phone Number: 987654333
Enter Designation: Tester
Employee added successfully.
Asset Management System
1. Admin Operations
2. Hardware Asset Operations
3. Employee Operations
4. Hardware Assigned Operations
5. Exit
Choose an option: 3
Employee Operations
1. Add Employee
2. Remove Employee
3. Update Employee
4. Get Employee
5. List All Employees
6. Back to Main Menu
1
Enter Employee ID: 3
Enter Employee Name: Srinith
Enter Email: srinith@gmail.com
Enter Phone Number: 5678998765
Enter Designation: QA
Employee added successfully.
Asset Management System
1. Admin Operations
2. Hardware Asset Operations
3. Employee Operations
4. Hardware Assigned Operations
5. Exit
```

```
Choose an option: 3
Employee Operations
1. Add Employee
2. Remove Employee
3. Update Employee
4. Get Employee
5. List All Employees
6. Back to Main Menu
5
ID         Name                Email                     Phone           Designation
-------------------------------------------------------------------------------
1          Prathik             prathik.b@gmil.com        9876543210      Software Developer
2          Sanjith S           sanjith@gmail.com         987654333       Tester
3          Srinith             srinith@gmail.com         5678998765      QA
Asset Management System
1. Admin Operations
2. Hardware Asset Operations
3. Employee Operations
4. Hardware Assigned Operations
5. Exit
```

```
Choose an option: 4
Hardware Assigned Operations
1. Add Hardware Assigned
2. Remove Hardware Assigned
3. Update Hardware Assigned
4. Get Hardware Assigned
5. List All Hardware Assigned
6. Back to Main Menu
1
Enter Hardware Assigned ID: 1
Enter Hardware Asset ID: 1
Enter Employee ID: 1
Enter Assigned By: Balaji
Enter Assigned Date (YYYY-MM-DD): 2024-08-13
Enter Assigned Status: Active
Hardware assigned added successfully.
Asset Management System
1. Admin Operations
2. Hardware Asset Operations
3. Employee Operations
4. Hardware Assigned Operations
5. Exit
Choose an option: 4
Hardware Assigned Operations
1. Add Hardware Assigned
2. Remove Hardware Assigned
3. Update Hardware Assigned
4. Get Hardware Assigned
5. List All Hardware Assigned
6. Back to Main Menu
4
Enter Hardware Assigned ID to retrieve: 1
ID          Asset ID        Employee ID     Assigned By          Assigned Date        Status
-------------------------------------------------------------------------------------------
1           1               1               Balaji               2024-08-13           Active
```

```
Asset Management System
1. Admin Operations
2. Hardware Asset Operations
3. Employee Operations
4. Hardware Assigned Operations
5. Exit
Choose an option: 5
Exiting...
```