EMPLOYEE MANAGEMENT SYSTEM

Project Documentation

Project Overview

1. Project Title

Employee Management System using Java

1.2 Project Description

The Employee Management System (EMS) is a **menu-driven, console-based Java application** designed to manage employee records efficiently.
It allows users to **add, view, search, update, and delete employee data**, with persistent storage using file handling.

The system uses **ArrayList** for storing employee objects and **HashMap** for fast employee lookup using employee ID.
It also includes **salary reports**, **department-wise summaries**, and **exception handling** to ensure reliability.

1.3 Objectives

- To manage employee records digitally
- To implement CRUD operations in Java
- To use efficient data structures (ArrayList & HashMap)
- To demonstrate file handling using serialization
- To generate salary and department reports
- To apply exception handling and validation

---

2. Setup Instructions

2.1 Software Requirements

- Operating System: Windows / Linux / macOS
- Java JDK: **Java 8 or higher**
- IDE: VS Code / IntelliJ IDEA / Eclipse

2.2 Installation Steps

1. Install Java JDK
2. Set JAVA_HOME environment variable
3. Open IDE and create a new Java project
4. Create the following files:

- Employee.java
- EmployeeManagementSystem.java
- EmployeeFileHandler.java
- EmployeeReportGenerator.java

2.3 Running the Project

1. Compile all .java files
2. Run EmployeeManagementSystem.java
3. Use menu options to interact with the system

---

3. Code Structure

3.1 Project Folder Structure

EmployeeManagementSystem/

|

├── Employee.java

├── EmployeeManagementSystem.java

├── EmployeeFileHandler.java

├── EmployeeReportGenerator.java

└── employees.dat

3.2 File Descriptions

File Name	**Description**
Employee.java	Employee class with attributes & serialization
EmployeeManagementSystem.java	Main class with menu and CRUD operations
EmployeeFileHandler.java	Handles file save/load operations
EmployeeReportGenerator.java	Generates salary & department reports
employees.dat	Serialized employee data file

---

4. Data Format Specification

4.1 Employee Data Fields

Field	Data Type	Description
id	String	Unique employee ID
name	String	Employee name
department	String	Department name
position	String	Job role
salary	double	Monthly salary
joinDate	LocalDate	Date of joining

4.2 File Format

- File Name: employees.dat
- Format: **Serialized Java Object**
- Contains: ArrayList<Employee>

---

5. File Handling Procedures

5.1 Saving Data

- Uses ObjectOutputStream
- Serializes employee list into employees.dat
- Automatically saves after adding employee

5.2 Loading Data

- Uses ObjectInputStream
- Reads employee data on application startup
- Handles missing file using exception handling

---

6. Employee Management Workflow

6.1 Add Employee

1. User enters employee details
2. System validates unique ID
3. Data stored in ArrayList & HashMap
4. Data saved to file

6.2 View Employees

- Displays all employee records in tabular format

6.3 Search Employee

- Search by:
    - Employee ID
    - Name
    - Department

6.4 Update Employee

- Updates salary using employee ID

6.5 Delete Employee

- Removes employee from system & HashMap

6.6 Generate Reports

- Salary statistics
- Department-wise summary

---

7. Technical Details

7.1 Algorithms Used

- Linear search for name & department
- HashMap lookup for employee ID
- Stream API for salary calculations

7.2 Data Structures Used

Structure	Purpose
ArrayList	Stores employee objects
HashMap	Fast lookup using employee ID
Map	Department grouping

7.3 Architecture

- **Layered architecture**
    - Model → Employee
    - Service → Management & Reports
    - Persistence → File Handler

---

8. Exception Handling

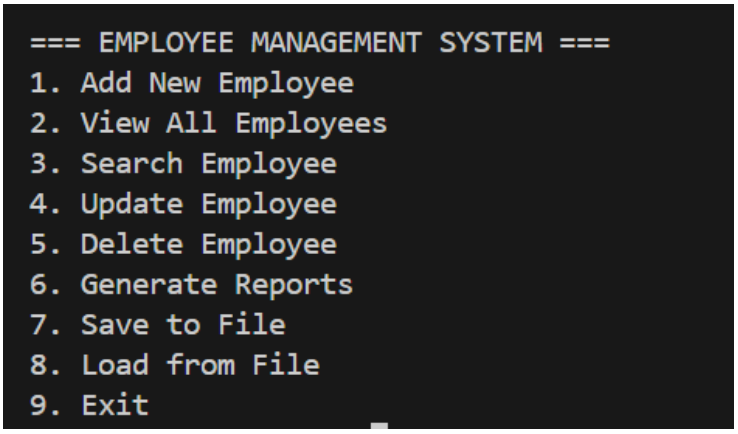
Scenario	Exception Handled
Invalid input	NumberFormatException
File not found	FileNotFoundException
Class mismatch	ClassNotFoundException
IO failure	IOException

---

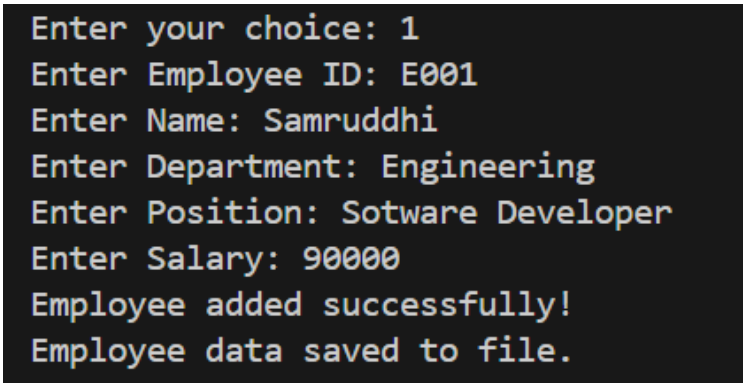
9. Visual Documentation (Screenshots to Include)

Mandatory Screenshots

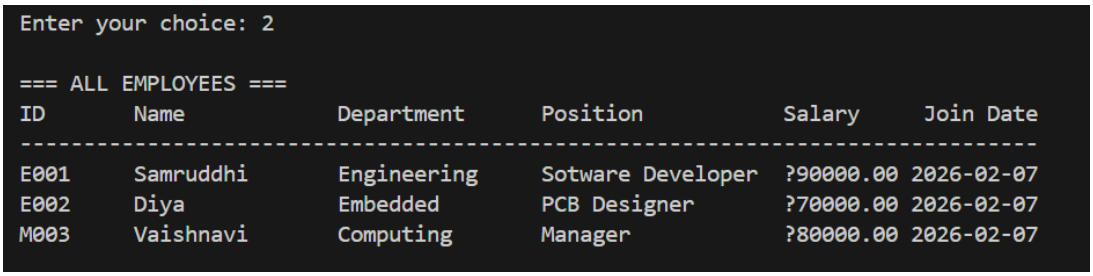
1. Main menu screen



2. Add employee



3. View employees



4. Search employee

```
Enter your choice: 3

1. Search by ID
2. Search by Name
3. Search by Department
1
Enter value: E001
ID: E001, Name: Samruddhi, Dept: Engineering, Position: Sotware Developer, Salary: ?90000.0, Joined: 2026-02-07
```

5. Generate salary report

```
Enter your choice: 6

1. Department-wise Summary
2. Salary Statistics
1

? DEPARTMENT SUMMARY:
? Engineering: 1 employees, Average: ?90000.00
? Embedded: 1 employees, Average: ?70000.00
? Computing: 1 employees, Average: ?80000.00
```

6. Data saved confirmation

```
=== EMPLOYEE MANAGEMENT SYSTEM ===
1. Add New Employee
2. View All Employees
3. Search Employee
4. Update Employee
5. Delete Employee
6. Generate Reports
7. Save to File
8. Load from File
9. Exit
Enter your choice: 7
Employee data saved to file.

=== EMPLOYEE MANAGEMENT SYSTEM ===
1. Add New Employee
2. View All Employees
3. Search Employee
4. Update Employee
5. Delete Employee
6. Generate Reports
7. Save to File
8. Load from File
9. Exit
Enter your choice: 8
Employee data loaded from file.
```

10. Testing Evidence

10.1 Sample Test Cases

Test Case	Input	Expected Output
Add Employee	Valid data	Employee added successfully
Duplicate ID	Existing ID	Error message
Search by Dept	Engineering	Matching employees
Delete Employee	Valid ID	Employee removed
Salary Report	Employees exist	Correct statistics

10.2 Validation

- All inputs validated
- Duplicate IDs prevented
- File persistence verified