Vityarthi Project Report
(Programming in Java)

Academic Year – 2024-2025

Project – Marks Manager System

Name – Aryan Singh Dangi Thakur

Course- Integrated M.Tech (AI)

Registration No. – 24MIM10121

Introduction-

The Marks Manager System is a Java-based application designed to simplify the process of recording and analysing student marks. The project demonstrates practical use of object-oriented programming principles, Java Swing for GUI development, and file handling for persistent storage.

The motivation behind building this system is the common difficulty educators face in manually recording marks, calculating totals and grades, tracking class performance, and generating insights. By automating this process, the Marks Manager System ensures reliability, accuracy, and ease of use.

The system provides two interfaces:

1. **Console interface** for quick testing.
2. **GUI interface** using Java Swing for a user-friendly experience.

Problem Statement-

Traditional methods of maintaining student marks using physical registers or spreadsheets are prone to:

- Human errors
- Repetitive calculations
- Lack of instant insights
- Difficulty updating and maintaining data
- No automated statistics or analytics

The problem is the absence of a lightweight, offline, easy-to-use tool that can:

- Store student marks
- Compute total, percentage, grade, and pass/fail
- Provide class analytics
- Save/load data
- Display results cleanly in a GUI

The Marks Manager System solves these issues efficiently.

Functional Requirements

1. Add Student Record
2. Edit/Update Student Record
3. Delete Record
4. Search Student by Name
5. Calculate Total, Percentage, Grade, Result
6. Display All Students
7. Compute Class Average
8. Identify Topper
9. Count Pass/Fail Students
10. Save Data to CSV File
11. Load Data from CSV File
12. GUI (Swing) with Buttons and Forms

Non-Functional Requirements

- **Usability:** Clean UI, clear buttons, validations
- **Portability:** Runs on any OS with Java
- **Maintainability:** Modular code structure
- **Performance:** Efficient for many records
- **Reliability:** Accurate calculations
- **Scalability:** Support for future extensions
- **Data Integrity:** Safe file operations

System Architecture

The system is divided into four layers:

1. Data Layer

Student.java

- Stores name and marks
- Computes total, percentage, grade, result

2. Business Logic Layer

StudentManager.java

- Handles adding, updating, deleting
- Supports analytics (topper, average, pass/fail count)

3. Storage Layer

MarksStorage.java

- Saves list of students to CSV
- Loads list of students from CSV

4. Presentation Layer

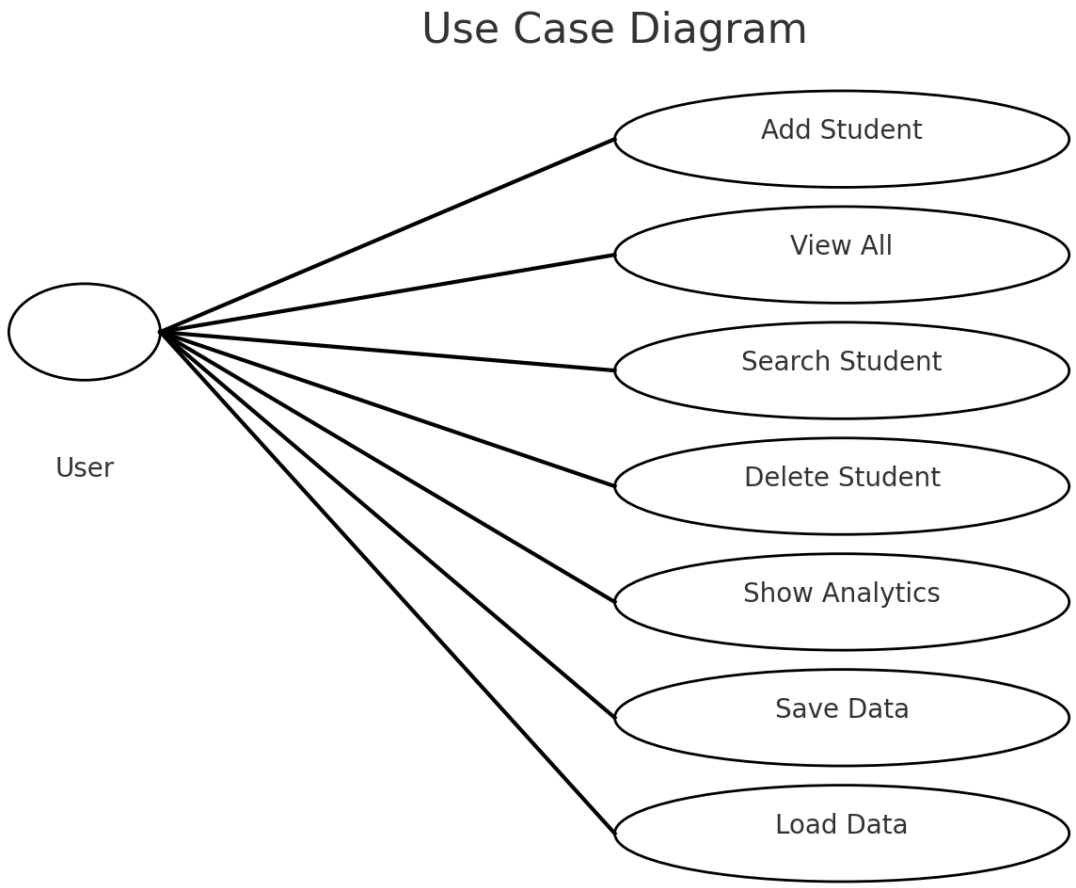
ConsoleApp.java

MarksManagerGUI.java

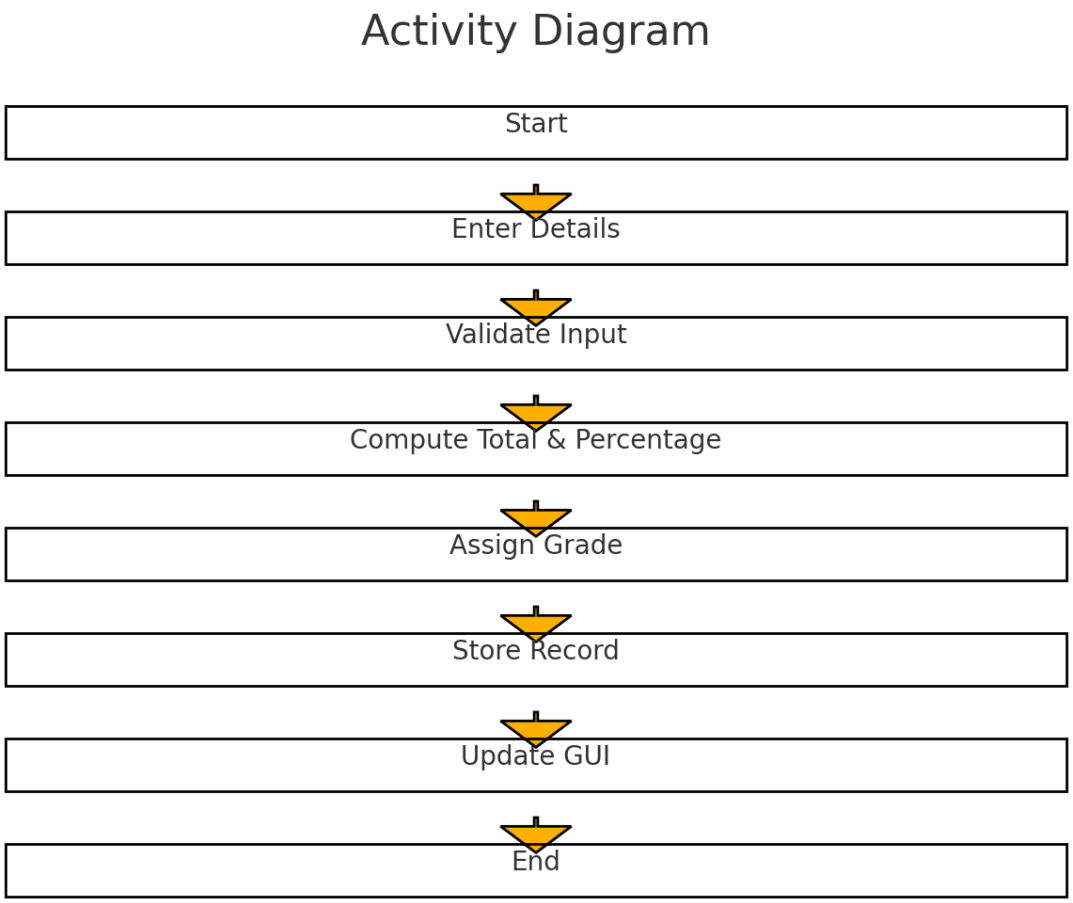
- Console interface
- Swing GUI (Buttons, TextFields, Dialogs)

Design Diagrams

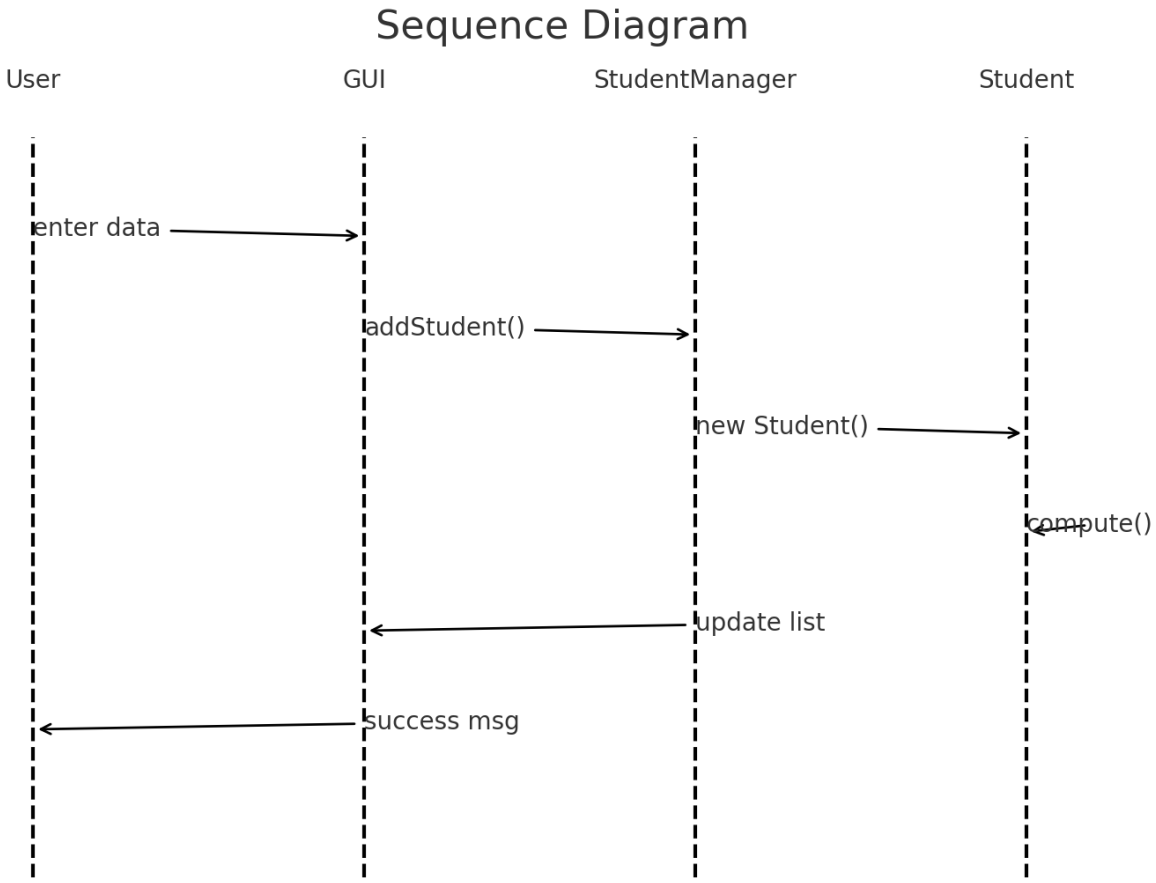
- **Use Case Diagram**



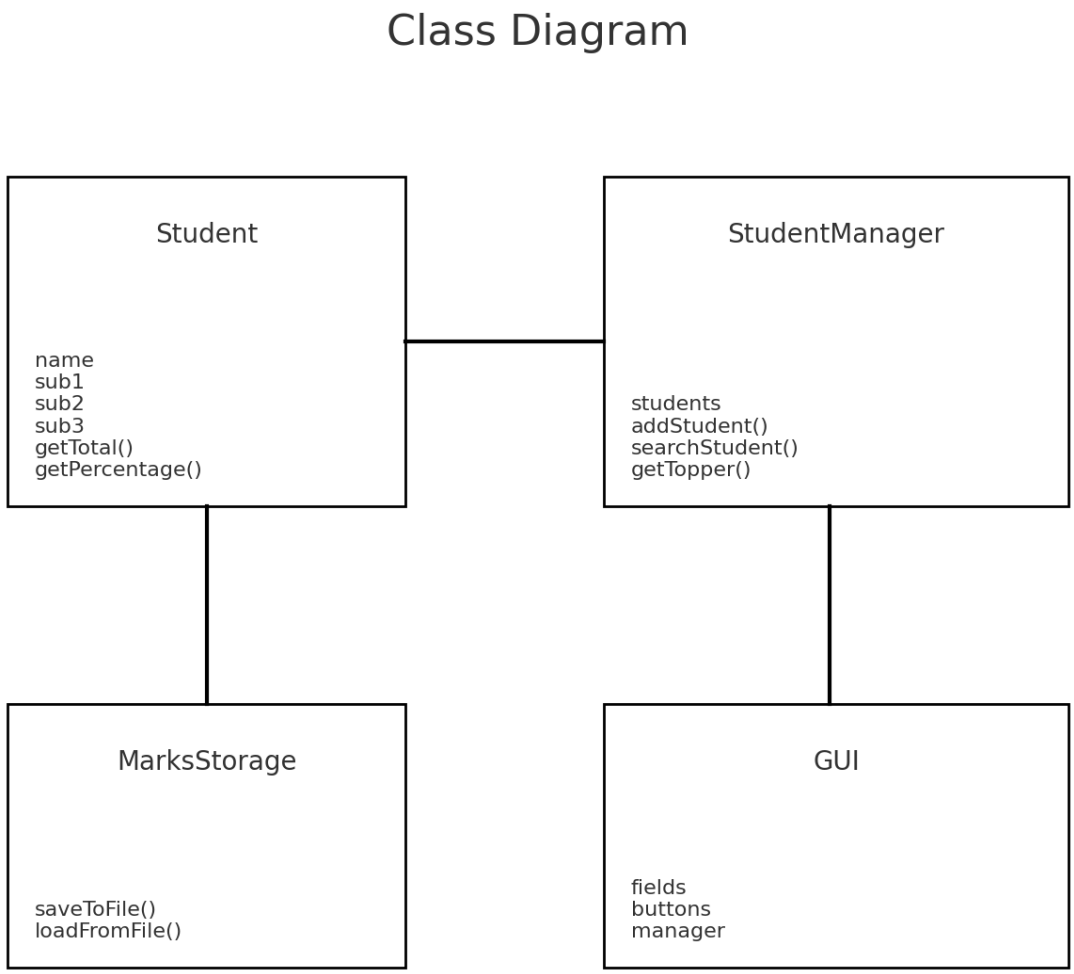
- **Workflow/Activity Diagram**



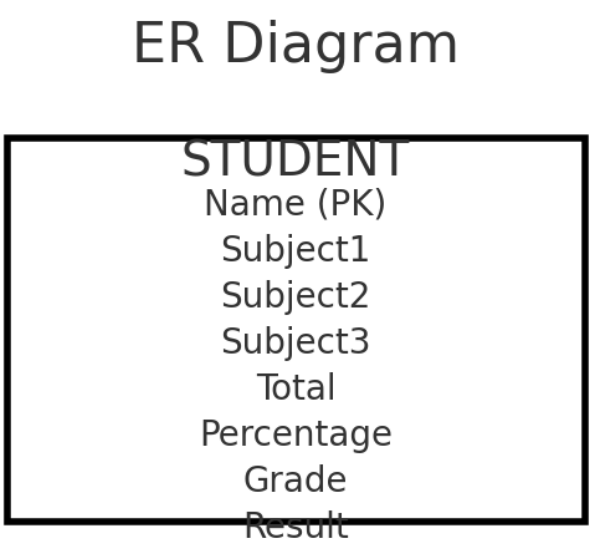
- **Sequence Diagram**



- **Class Diagram**



- **ER Diagram**



Design Decisions & Rationale

- **Java** chosen for portability and OOP support
- **Swing** chosen for built-in GUI support
- **ArrayList** for flexible record storage
- **CSV** chosen for simple external storage
- **Modular design** ensures maintainability
- **Validation** ensures safe and correct input

Implementation Details

Student.java

- Holds marks and student info
- Calculates grade and percentage

StudentManager.java

- Core logic
- CRUD operations
- Analytics (topper, stats)

MarksStorage.java

- File I/O using CSV
- Saves and loads records securely

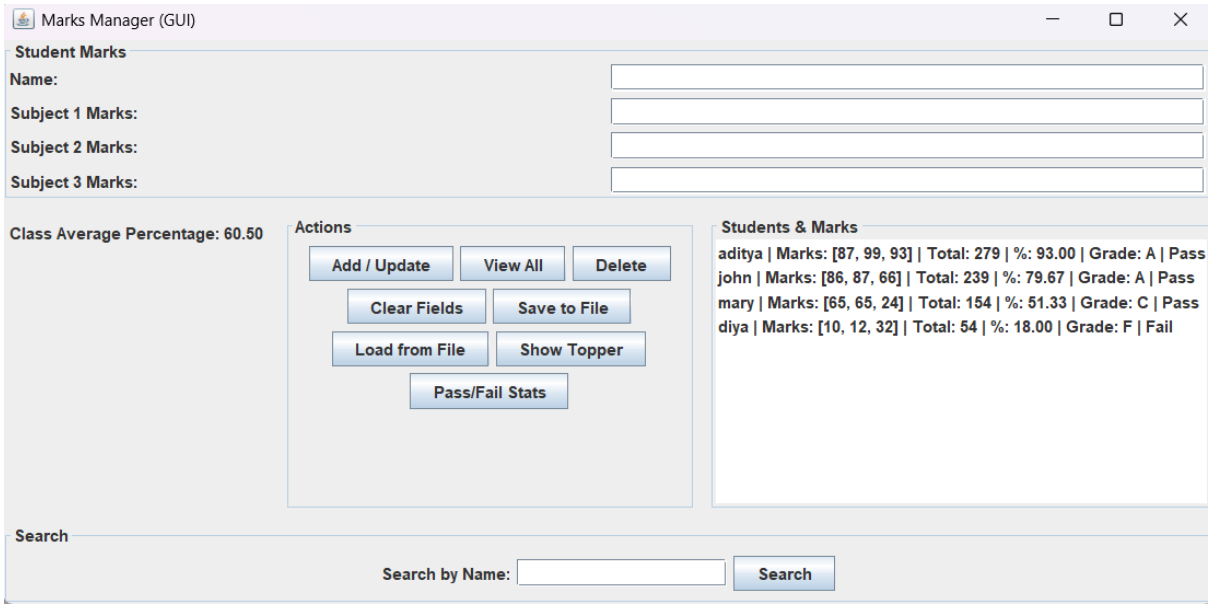
ConsoleApp.java

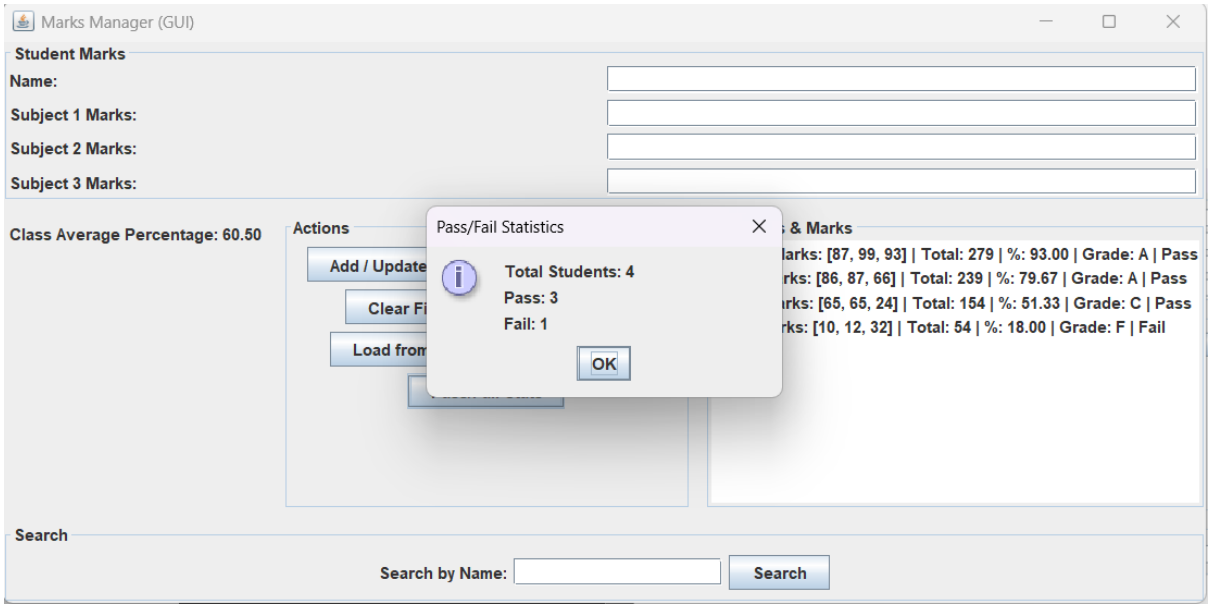
- Menu-driven interface

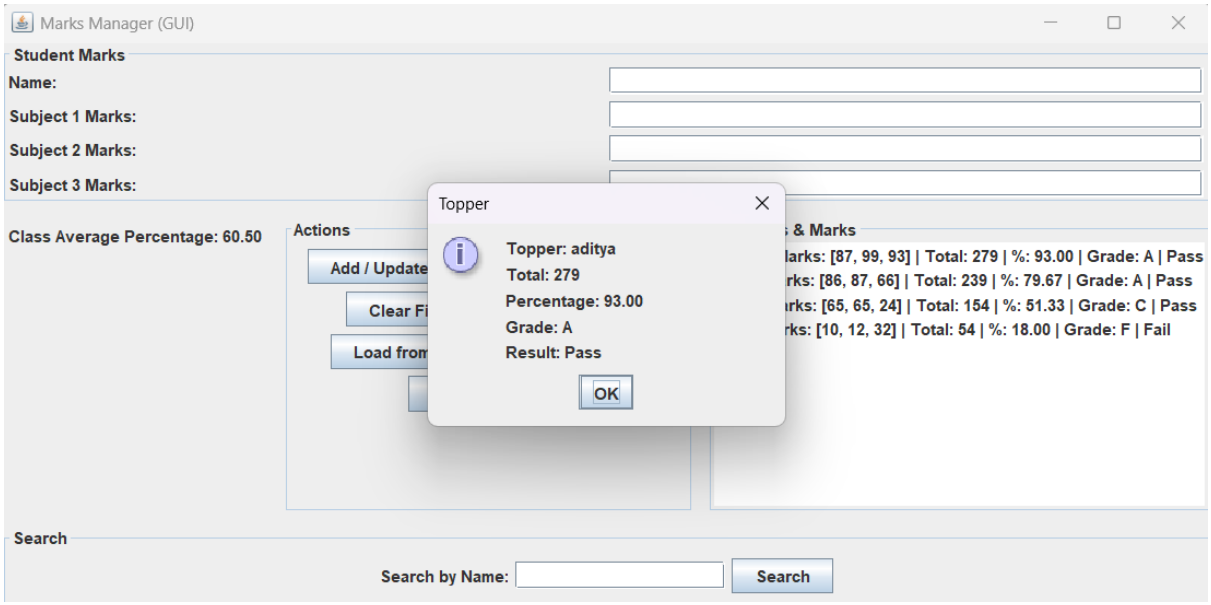
MarksManagerGUI.java

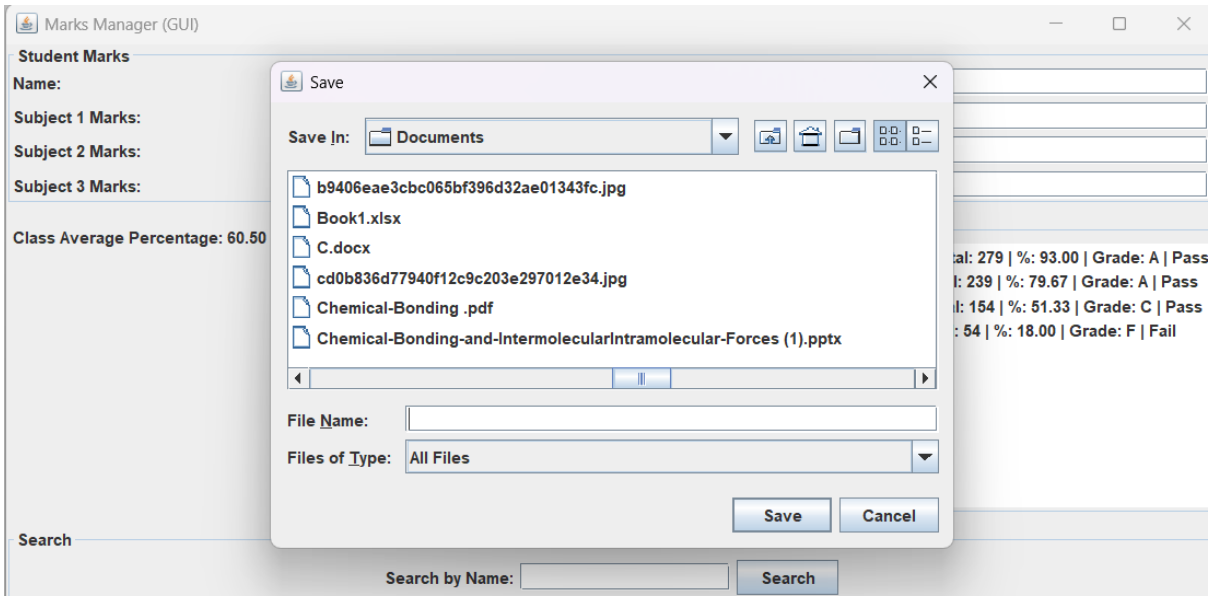
- Swing-based GUI
- Buttons for all operations
- List display for students
- Dialog popups for search, topper, stats

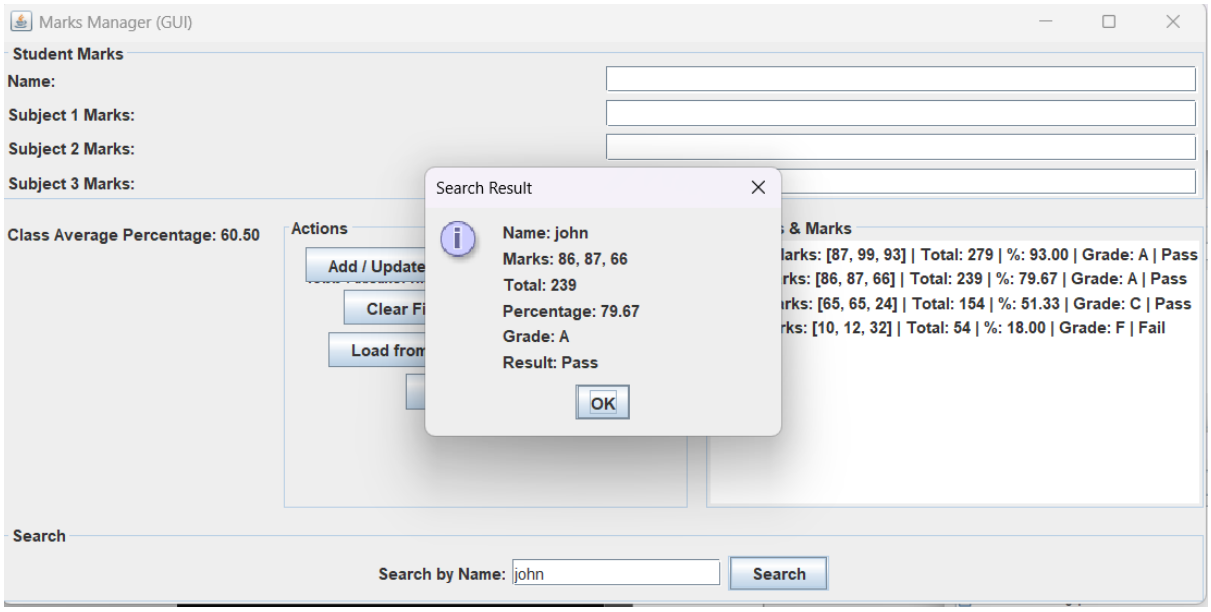
Screenshot/Results

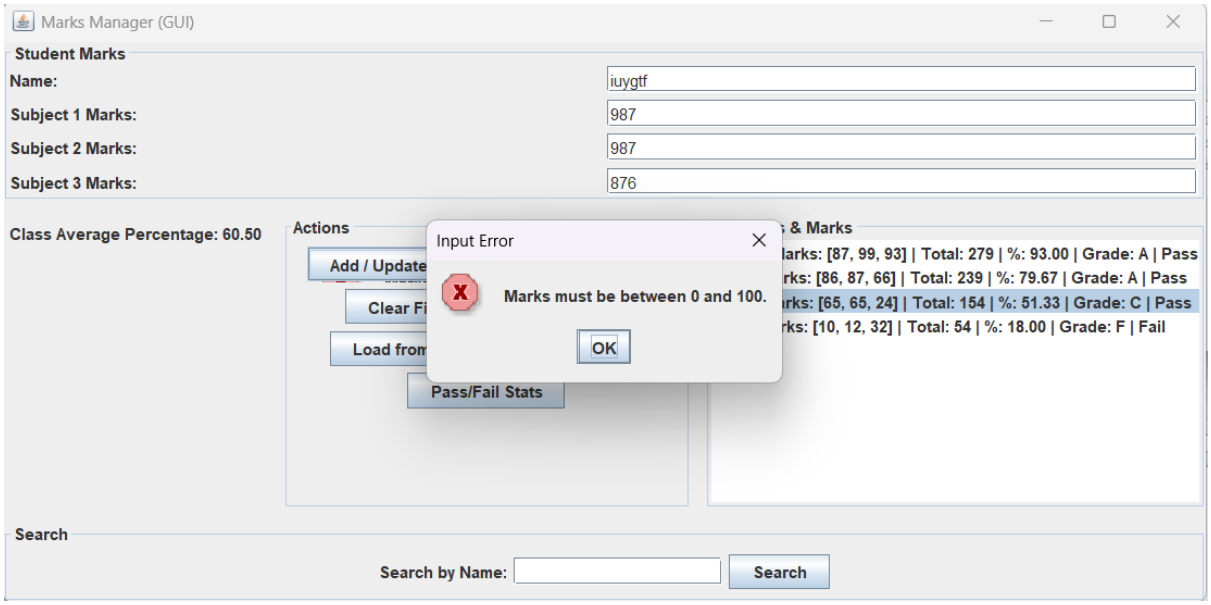





Marks Manager (GUI)
Student Marks
Name:
Subject 1 Marks:
Subject 2 Marks:
Subject 3 Marks:
Class Average Percentage: 60.50
Actions
Add / Update
Clear Fi
Load from
& Marks
arks: [87, 99, 93] | Total: 279 | %: 93.00 | Grade: A | Pass
rks: [86, 87, 66] | Total: 239 | %: 79.67 | Grade: A | Pass
rks: [65, 65, 24] | Total: 154 | %: 51.33 | Grade: C | Pass
rks: [10, 12, 32] | Total: 54 | %: 18.00 | Grade: F | Fail
Topper
Topper: aditya
Total: 279
Percentage: 93.00
Grade: A
Result: Pass
OK
Search
Search by Name:
Search


Marks Manager (GUI)
Student Marks
Name:
Subject 1 Marks:
Subject 2 Marks:
Subject 3 Marks:
Class Average Percentage: 60.50
al: 279 | %: 93.00 | Grade: A | Pass
l: 239 | %: 79.67 | Grade: A | Pass
l: 154 | %: 51.33 | Grade: C | Pass
: 54 | %: 18.00 | Grade: F | Fail
Save
Save In:
Documents
b9406eae3cbc065bf396d32ae01343fc.jpg
Book1.xlsx
C.docx
cd0b836d77940f12c9c203e297012e34.jpg
Chemical-Bonding .pdf
Chemical-Bonding-and-IntermolecularIntramolecular-Forces (1).pptx
File Name:
Files of Type:
All Files
Save
Cancel
Search
Search by Name:
Search


Marks Manager (GUI)
Student Marks
Name:
Subject 1 Marks:
Subject 2 Marks:
Subject 3 Marks:
Class Average Percentage: 60.50
Actions
Add / Update
Clear Fi
Load from
& Marks
arks: [87, 99, 93] | Total: 279 | %: 93.00 | Grade: A | Pass
rks: [86, 87, 66] | Total: 239 | %: 79.67 | Grade: A | Pass
rks: [65, 65, 24] | Total: 154 | %: 51.33 | Grade: C | Pass
rks: [10, 12, 32] | Total: 54 | %: 18.00 | Grade: F | Fail
Search Result
Name: john
Marks: 86, 87, 66
Total: 239
Percentage: 79.67
Grade: A
Result: Pass
OK
Search
Search by Name: john
Search


Marks Manager (GUI)
Student Marks
Name: iuygtf
Subject 1 Marks: 987
Subject 2 Marks: 987
Subject 3 Marks: 876
Class Average Percentage: 60.50
Actions
Add / Update
Clear Fi
Load from
Pass/Fail Stats
& Marks
arks: [87, 99, 93] | Total: 279 | %: 93.00 | Grade: A | Pass
rks: [86, 87, 66] | Total: 239 | %: 79.67 | Grade: A | Pass
rks: [65, 65, 24] | Total: 154 | %: 51.33 | Grade: C | Pass
rks: [10, 12, 32] | Total: 54 | %: 18.00 | Grade: F | Fail
Input Error
Marks must be between 0 and 100.
OK
Search
Search by Name:
Search

Testing Approach

1. Unit Testing

- Tested computation methods
- Boundary testing (0, 100, 40, etc.)

2. Functional Testing

- Add → View → Save → Load
- Search/Delete flows

3. Negative Testing

- Non-numeric input
- Out-of-range marks

4. File Handling Tests

- Missing file
- Corrupted CSV
- Repeated saves

Challenges Faced

- GUI alignment issues with layouts
- Handling user input errors
- Managing CSV load/save formatting
- Java package/path errors during compilation
- Keeping UI synced with data updates

Learnings & Key Takeaways

- Stronger OOP understanding
- Swing GUI design principles
- Collections & dynamic data structures
- Reliable exception handling
- File I/O techniques
- Layered architecture design

Future Enhancements

- Add database support (MySQL/SQLite)
- Add more subjects
- Add charts (bar/pie graphs)
- Add login system
- Add export to PDF directly
- Add mobile/web version

References

- Oracle Java Documentation
- Java Swing official guide
- GeeksforGeeks (Java, OOP, Swing, File I/O)
- StackOverflow solutions
- Classroom/Lab notes