📚 Student Information System (SIS) Project Documentation

Project Overview

Clear Description of Project Goals and Objectives

The goal of this Student Information System (SIS) project is to provide a robust, command-line interface for managing student records. The system utilizes Object-Oriented Programming (OOP) principles and Java Collections to handle essential administrative functions.

Key Objectives:

- **CRUD Operations:** Implement the four fundamental database operations (Create, Read, Update, Delete) for student records.
- **Data Validation:** Ensure data integrity by validating user inputs for critical fields like age (positive integer) and grade (A-F or 1-10 format).
- **Modular Design:** Separate concerns into dedicated classes (Student, StudentManager, ValidationUtils) for maintainability and clarity.
- **User Experience:** Provide a clear, menu-driven interface for administrators to interact with the system.

Setup Instructions

Step-by-step installation and configuration guide

This project is a pure Java console application and requires only a Java Development Kit (JDK) to run.

1. **Install Java:** Ensure you have the Java Development Kit (JDK) installed on your system (Version 8 or higher is recommended).
2. **Save Source Files:**
   - Create a directory named Student_SIS.
   - Save all four files (Student.java, StudentManager.java, ValidationUtils.java, StudentInformationSystem.java) inside this directory.
3. **Compile the Project:**
   - Open your terminal or command prompt.
   - Navigate to the project directory: cd Student_SIS
   - Compile all four Java source files: javac *.java
4. **Run the Application:**
   - Execute the main class: java StudentInformationSystem
   - The application menu should now appear in the console.

Code Structure

Well-organized code with clear file hierarchy

The project follows a modular, three-tier structure to maintain separation of concerns.

- Student_SIS/
  - Student.java (Data Model)
  - StudentManager.java (Business Logic/Controller)
  - ValidationUtils.java (Utility/Helper Class)
  - StudentInformationSystem.java (Application Entry Point/View)

File Name	Purpose	Key Concepts
Student.java	Defines the blueprint for a single student entity. Contains private fields and public Getters/Setters.	OOP (Encapsulation), Data Model
StudentManager.java	Manages the collection of Student objects and implements all core CRUD operations (Add, View, Search, Update, Delete).	Collections (ArrayList), Business Logic
ValidationUtils.java	Provides static methods for standardized and validated user input. Handles input exceptions and format checks.	Static Utility Methods, Input Validation, Exception Handling
StudentInformationSystem.java	Contains the main() method. Initializes the StudentManager and runs the main application loop and menu interface.	Application Entry Point, Control Flow (switch, while)

Visual Documentation

Conceptual Console Outputs Demonstrating Functionality

1. Main Menu Display

===== STUDENT INFORMATION SYSTEM =====

1. Add Student

2. View All Students

...

6. Exit

Enter your choice:

2. Add Student with Validation

Enter Name: Alex Smith

Enter Age: -5

❌ Age must be a positive number!

Enter Age: 20

Enter Grade (A-F or 1-10): G

❌ Invalid grade! Enter A-F or 1-10.

Enter Grade (A-F or 1-10): B

Enter Student ID: S1001

Enter Contact Number: 555-1234

✔ Student Added Successfully!

3. View All Students

Student ID	Name	Age	Grade	Contact
S1001	Alex Smith	20	B	555-1234
S1002	Jane Doe	22	A	555-4321

Technical Details

Explanation of algorithms, data structures, and architecture

A. Data Structures

- **ArrayList<Student> (students in StudentManager.java):**
  - Used as the in-memory database to store all Student objects.
  - Chosen for its dynamic size and ease of use for sequential storage and retrieval.

B. Object-Oriented Principles (OOP)

- **Encapsulation (Student.java):** The student data fields are declared private and accessed only through public **Getters and Setters**. This protects the data from unauthorized or inconsistent direct modification.
- **Separation of Concerns:** The project clearly divides responsibilities into Model, Controller/Logic, and View/I/O classes.

C. Core Algorithms

1. **Linear Search (Search, Update, Delete):**
   - The searchStudent(), updateStudent(), and deleteStudent() methods all employ a **linear search algorithm**.
   - They iterate through the ArrayList<Student> to find a matching studentId or name.
   - If a match is found, the loop is immediately exited (return).
2. **Input Validation (Regex):**
   - The ValidationUtils.validateGrade() method uses **Regular Expressions (Regex)** to enforce strict formatting for the grade input:
     - grade.matches("[A-Fa-f]"): Matches a single letter from A through F (case-insensitive).
     - grade.matches("[1-9]|10"): Matches numbers 1 through 9, or the number 10.

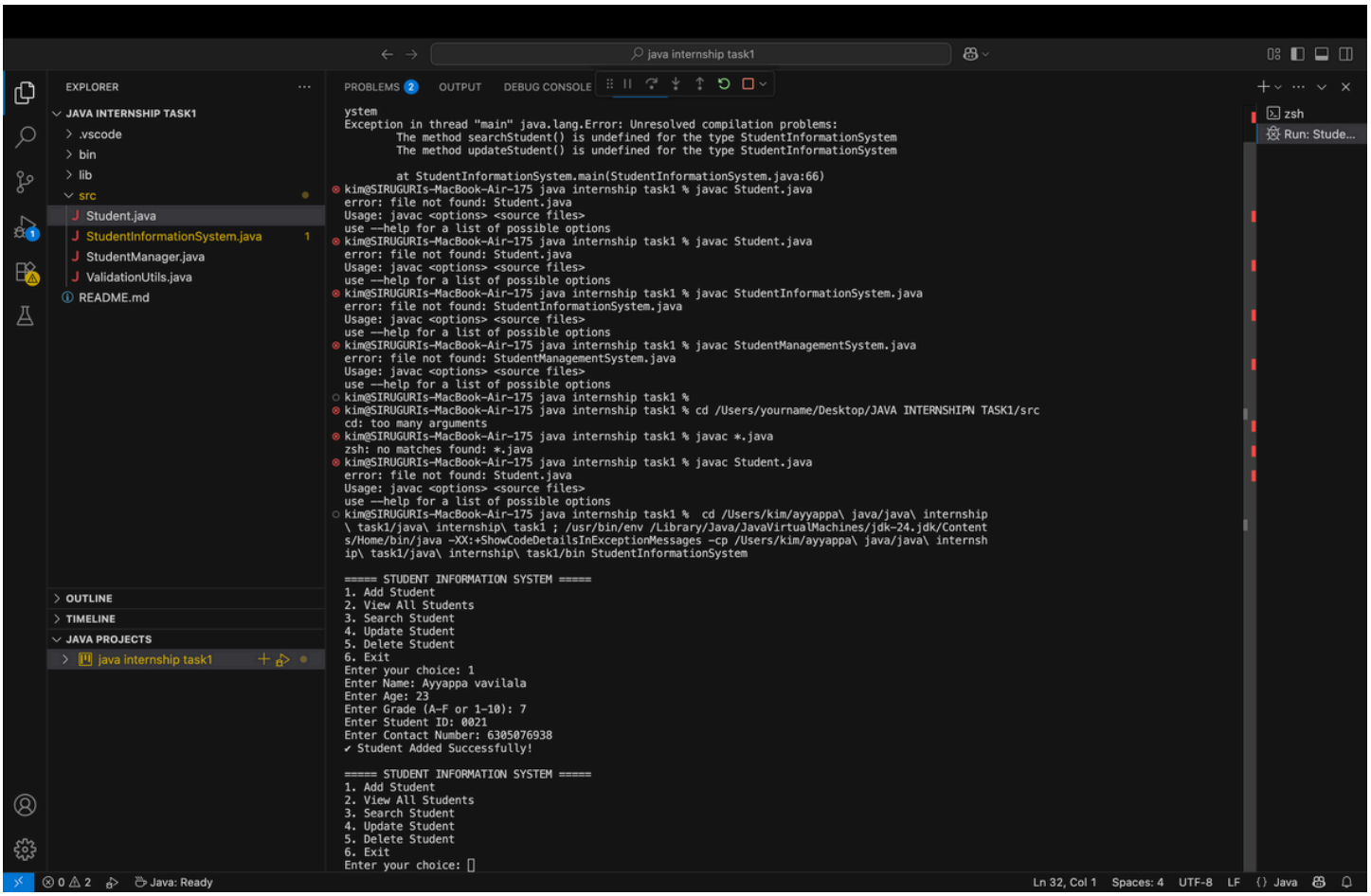
Testing Evidence

Examples of test cases and validation

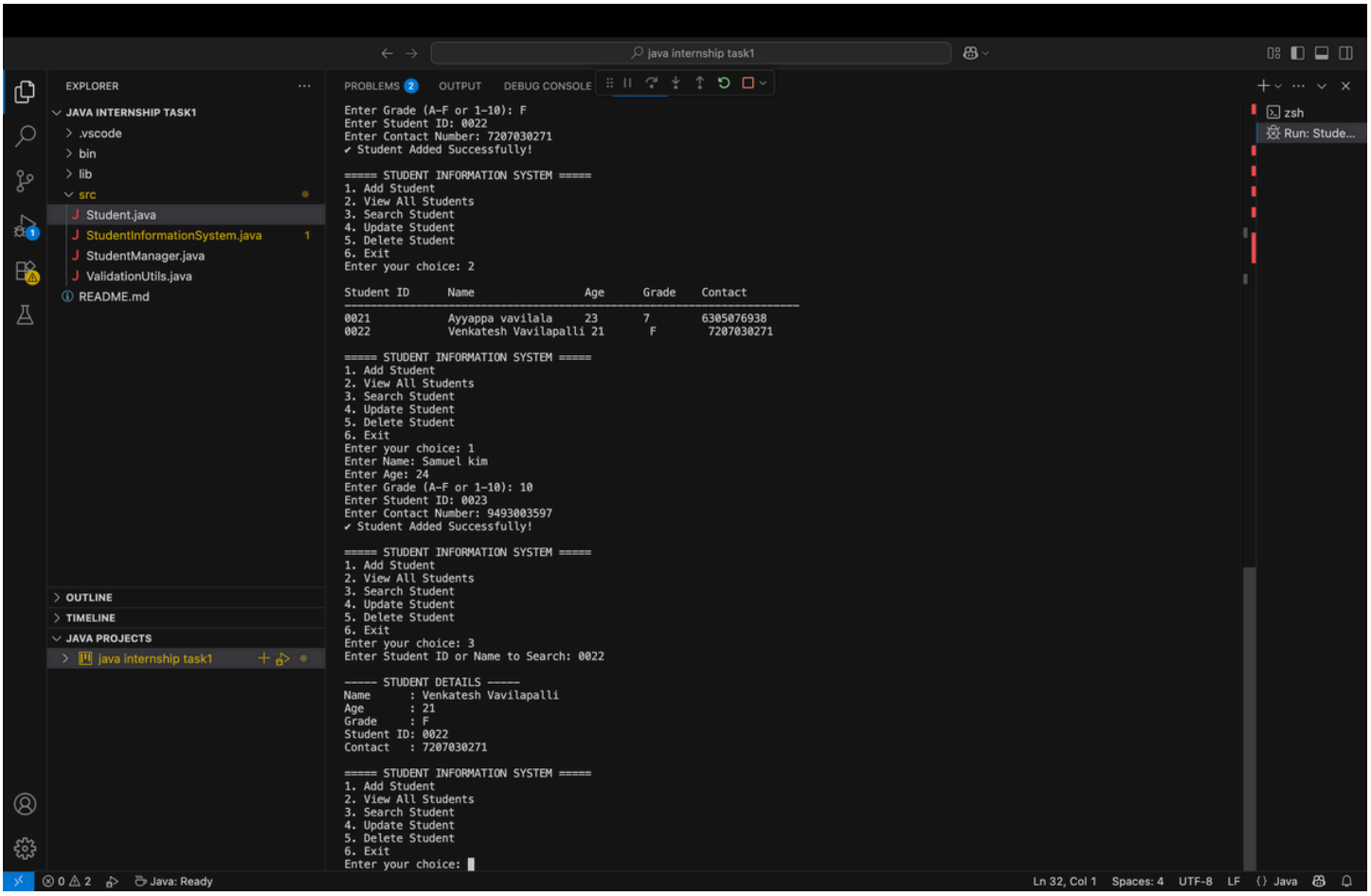
Test Case #	Feature	Action Steps	Expected Result
TC-001	Add Student (Valid)	1. Choose Option 1. 2. Enter valid data (Name: Mark, Age: 18, Grade: 9, ID: M101).	System prints: "✔ Student Added Successfully!"
TC-002	Add Student (Invalid Age)	1. Choose Option 1. 2. Enter Age: -5. 3. Enter Age: abc. 4. Enter Age: 19.	System rejects -5 and abc with error messages, then accepts 19.
TC-003	View All	1. Choose Option 2	Prints the formatted table header and the details of student M101.
TC-004	Search by ID	1. Choose Option 3. 2. Enter Search Key: M101.	Prints the full details of student M101 in the student details block.
TC-005	Update	1. Choose Option 4. 2. Enter ID to Update: M101. 3. Change Grade to A.	System prints: "✔ Student Updated Successfully!". TC-003 re-run should show Grade A.
TC-006	Delete	1. Choose Option 5. 2. Enter ID to Delete: M101. 3. Choose Option 2 (View All).	System prints: "✔ Student Deleted Successfully!". View All prints "❌ No student records found!"
TC-007	Invalid Choice	1. Enter choice 9. 2. Enter choice xyz.	System prints "❌ Invalid choice! Try again." for 9. System prints "❌ Invalid input! Enter a number." for xyz.

VISUAL REPRESENTATION OF SAMPLE OUTPUT

OUTPUT-1


java internship task1
EXPLORER
JAVA INTERNSHIP TASK1
.vscode
bin
lib
src
Student.java
StudentInformationSystem.java
StudentManager.java
ValidationUtils.java
README.md
OUTLINE
TIMELINE
JAVA PROJECTS
java internship task1
PROBLEMS 2
OUTPUT
DEBUG CONSOLE
ystem
Exception in thread "main" java.lang.Error: Unresolved compilation problems:
The method searchStudent() is undefined for the type StudentInformationSystem
The method updateStudent() is undefined for the type StudentInformationSystem
at StudentInformationSystem.main(StudentInformationSystem.java:66)
kim@SIRUGURIs-MacBook-Air-175 java internship task1 % javac Student.java
error: file not found: Student.java
Usage: javac <options> <source files>
use --help for a list of possible options
kim@SIRUGURIs-MacBook-Air-175 java internship task1 % javac Student.java
error: file not found: Student.java
Usage: javac <options> <source files>
use --help for a list of possible options
kim@SIRUGURIs-MacBook-Air-175 java internship task1 % javac StudentInformationSystem.java
error: file not found: StudentInformationSystem.java
Usage: javac <options> <source files>
use --help for a list of possible options
kim@SIRUGURIs-MacBook-Air-175 java internship task1 % javac StudentManagementSystem.java
error: file not found: StudentManagementSystem.java
Usage: javac <options> <source files>
use --help for a list of possible options
kim@SIRUGURIs-MacBook-Air-175 java internship task1 %
kim@SIRUGURIs-MacBook-Air-175 java internship task1 % cd /Users/yourname/Desktop/JAVA INTERNSHIPN TASK1/src
cd: too many arguments
kim@SIRUGURIs-MacBook-Air-175 java internship task1 % javac *.java
zsh: no matches found: *.java
kim@SIRUGURIs-MacBook-Air-175 java internship task1 % javac Student.java
error: file not found: Student.java
Usage: javac <options> <source files>
use --help for a list of possible options
kim@SIRUGURIs-MacBook-Air-175 java internship task1 % cd /Users/kim/ayyappa\ java/java\ internship
\ task1/java\ internship\ task1 ; /usr/bin/env /Library/Java/JavaVirtualMachines/jdk-24.jdk/Content
s/Home/bin/java -XX:+ShowCodeDetailsInExceptionMessages -cp /Users/kim/ayyappa\ java/java\ internsh
ip\ task1/java\ internship\ task1/bin StudentInformationSystem
===== STUDENT INFORMATION SYSTEM =====
1. Add Student
2. View All Students
3. Search Student
4. Update Student
5. Delete Student
6. Exit
Enter your choice: 1
Enter Name: Ayyappa vavilala
Enter Age: 23
Enter Grade (A-F or 1-10): 7
Enter Student ID: 0021
Enter Contact Number: 6305076938
✔ Student Added Successfully!
===== STUDENT INFORMATION SYSTEM =====
1. Add Student
2. View All Students
3. Search Student
4. Update Student
5. Delete Student
6. Exit
Enter your choice:
zsh
Run: Stude...
Java: Ready
Ln 32, Col 1
Spaces: 4
UTF-8
LF
Java

OUTPUT-2


java internship task1
EXPLORER
JAVA INTERNSHIP TASK1
.vscode
bin
lib
src
Student.java
StudentInformationSystem.java
StudentManager.java
ValidationUtils.java
README.md
OUTLINE
TIMELINE
JAVA PROJECTS
java internship task1
PROBLEMS 2
OUTPUT
DEBUG CONSOLE
Enter Grade (A-F or 1-10): F
Enter Student ID: 0022
Enter Contact Number: 7207030271
✔ Student Added Successfully!
===== STUDENT INFORMATION SYSTEM =====
1. Add Student
2. View All Students
3. Search Student
4. Update Student
5. Delete Student
6. Exit
Enter your choice: 2
Student ID Name Age Grade Contact
0021 Ayyappa vavilala 23 7 6305076938
0022 Venkatesh Vavilapalli 21 F 7207030271
===== STUDENT INFORMATION SYSTEM =====
1. Add Student
2. View All Students
3. Search Student
4. Update Student
5. Delete Student
6. Exit
Enter your choice: 1
Enter Name: Samuel kim
Enter Age: 24
Enter Grade (A-F or 1-10): 10
Enter Student ID: 0023
Enter Contact Number: 9493003597
✔ Student Added Successfully!
===== STUDENT INFORMATION SYSTEM =====
1. Add Student
2. View All Students
3. Search Student
4. Update Student
5. Delete Student
6. Exit
Enter your choice: 3
Enter Student ID or Name to Search: 0022
----- STUDENT DETAILS -----
Name : Venkatesh Vavilapalli
Age : 21
Grade : F
Student ID: 0022
Contact : 7207030271
===== STUDENT INFORMATION SYSTEM =====
1. Add Student
2. View All Students
3. Search Student
4. Update Student
5. Delete Student
6. Exit
Enter your choice:
zsh
Run: Stude...
Java: Ready
Ln 32, Col 1
Spaces: 4
UTF-8
LF
Java