Automated Student Grading System – Documentation

By – Aryan Gahlaut (Delhi Technological University)

Overview:

This is an Application-based automated student grading system in C++ language that enables an admin of an university to handle student's records for CRUD operations (insert, view, update, delete) according to their branch and semester. It employs CSV files at the backend to save admin credentials and student information (name, roll number ,semseter, date of birth, phone number, per subject grades ...etc).

Data Structures Used:

Following Data structures are used for the purpose

*1. **Struct: **Student*

Purpose: To store and operate on individual student records.

Data Members:

- `string name`
- `string roll`
- `string phone`
- `string dob`
- `string semester`
- `string branch`
- `vector<string> grades`

Member Functions:

- `string getGrade(int mark)` – Converts numeric marks into letter grades (O, A+, etc.) based to pre defined mapping.
- `string serialize()` – Converts student data into a CSV-formatted string.
- `void display()` – Displays student data in readable format.

*2. **Class: **GradingSystem*

Purpose: Handles admin authentication and all operations on student data.

Data Members:

- `string adminEmail, adminPass` – Stores admin login credentials.
- `vector<Student> students` – List of students in the selected branch/semester.
- `string adminFile` – Filename to store admin login data (`admin.csv`).
- `string selectedSemester, selectedBranch` – Current semester and branch context.
- `string targetFile` – Filename derived from semester and branch for saving student data.

Member Functions:

- `bool isValidRollForBranch(const string &roll, const string &branch)` – Validates roll number format for a given branch.
- `void loadAdmin()` – Loads admin credentials from `admin.csv`.
- `bool login()` – Performs admin login verification.
- `void loadStudents()` – Loads students from the target branch-semester file.
- `void saveStudents()` – Saves current student list to the CSV file.
- `void setSemesterAndBranch()` – Prompts user to set the context (semester and branch).
- `void insertStudent()` – Adds a new student after validation.
- `void viewStudent()` – Displays details of a student by roll number.
- `void modifyStudent()` – Deletes and re-inserts a student to modify record.
- `void deleteStudent()` – Removes a student record.
- `void menu()` – Main menu loop for admin operations.

Supporting Validation Functions:

- `isValidName(const string&)` - To check validity of name entered by admin.
- `isValidPhone(const string&)` - To check validity of Phone number entered by admin.

- `isValidDOB(const string&)` - To check validity of Date of Birth of a student.
- `isValidGrade(const string&)` - To check validity of Grades assigned to student.

Files Used:

1. admin.csv

- Format: `email,password`
- Stores admin login details.
- If not found, it creates one with default credentials: `nearfuture343@gmail.com,tokyo123`

2. <branch>_<semester>.csv

- Example: `computer_5.csv`, `electrical_3.csv`
- Each file contains records of students in the specified branch and semester.
- Format per student:

```
name,roll,phone,dob,semester,branch,grade1,grade2,...
```

Operations:

1. Insert Student

- Prompts for name, roll number, phone, DOB, number of subjects, and marks.
- Validates all inputs.
- Calculates and stores grades.
- Saves to appropriate `<branch>_<semester>.csv` file.
- If the file is not found , a new file is created on the fly and entry is made into this newly created file

2. Modify Student

- Deletes student based on roll number if found.
- Reuses `insertStudent()` to add updated data.

3. Delete Student

- Removes student record matching the roll number from memory and CSV if found , otherwise returns error.

4. View Student

- Displays student details by roll number from the loaded memory if found , otherwise displays appropriate message.

Flow of Execution:

1. Load admin credentials from `admin.csv`
2. Admin login authentication.
3. Display menu options (Insert, Modify, Delete, View).
4. Each student operation prompts to select branch and semester.
5. Relevant CSV is loaded/saved accordingly.

Sample File Usage:

- `computer_5.csv`: Contains records for Computer Engineering students in 5th semester.
- `chemical_3.csv`: Contains records for Chemical Engineering students in 3rd semester.

Every CSV file is separated by (branch, semester) pair for modular data storage and simpler handling of records.

Application:

QT C++ framework is used to develop full fledged working application (with the help of C++ code structure) by designing UI/UX for smooth user friendly navigation between tabs , built with the help of built-in libraries in QT .

Conclusion:

This student grading module, based on validation and modularity, provides secure, organized handling of data using CSV files and fundamental C++ features. Every part is engineered to cope with real-world input inaccuracies while ensuring proper separation of student and admin records.