**School Management System**

**[Strictly adhere to the object oriented programming specifications given in the problem statement. Template code is provided to ease the input output process. Template code will not compile. You need to fill in the missing code.]**

**Business Requirement:**

Your task is to create a basic School management System where students can register to courses, and view the course assigned to them.

**Work-Flow:**

Only students with right credentials can login. Otherwise, a message is display stating: “Wrong Credentials”.

1. Valid students are able to see the courses they are registered.
2. Valid students are able to register to any course in the system.

**Requirement 1:**

**CSV Files**

Create three Comma Separated Values (csv) files that contain columns specified in the tables below. The tables will be in the following format:

**Format:**

|  |  |  |
| --- | --- | --- |
| **Datatype** | **Name** | **Description** |
| The type of data contained in this column | The name of the column | The description of what this column will contain |

**File 1 – Students.csv:**

|  |  |  |
| --- | --- | --- |
| **Datatype** | **Name** | **Description** |
| String | email | Student’s current school email |
| String | name | The full name of the student |
| String | pass | Student’s password in order to login |

**File 2 – Courses.csv:**

|  |  |  |
| --- | --- | --- |
| **Datatype** | **Name** | **Description** |
| int | id | Unique Course Identifier |
| String | name | Provides the name of the course |
| String | Instructor | Provides the name of the instructor |

**File 3 – Attending.csv:**

|  |  |  |
| --- | --- | --- |
| **Datatype** | **Name** | **Description** |
| int | courseID | Unique Course Identifier |
| String | email | Student’s current school email |

**Requirement 2:**

**Model Class:**

Create a package in the src folder named: **CoreJava.Models**, in this package you will create every model class.

Every Model class must contain the following general two requirements:

1. First constructor takes no parameters and it initializes every members to an initial value.
2. Second constructor must initialize every private member with a parameter provided to the constructor.

Create a class Student with the private member variables specified in **TABLE 1**. These private members must have **GETTERS** and **SETTERS** methods.

The purpose of the Student class is to carry data related to one student.

**TABLE 1:**

|  |  |  |
| --- | --- | --- |
| **Datatype** | **Name** | **Description** |
| String | email | Student’s current school email |
| String | name | The full name of the student |
| String | pass | Student’s password in order to login |

Create a class Course with the private member variables specified in **TABLE 2**. These private members must have **GETTERS** and **SETTERS** methods.

The purpose of the Course class is to carry data related to one Course.

**TABLE 2:**

|  |  |  |
| --- | --- | --- |
| **Datatype** | **Name** | **Description** |
| String | courseID | Unique Course Identifier (ex: CIS101) |
| String | courseName | Provides the name of the course |
| String | InstructorName | Provides the name of the instructor |

Create a class Attending with the private member variables specified in **TABLE 3**. These private members must have **GETTERS** and **SETTERS** methods.

The purpose of the Attending class is to carry data related to which Students are attending which Courses.

**TABLE 3:**

|  |  |  |
| --- | --- | --- |
| **Datatype** | **Name** | **Description** |
| String | courseID | Unique Course Identifier (ex: CIS101) |
| String | studentEmail | Student’s school email |

**Requirement 2:**

**Data Access Objects**

Under the package named: **CoreJava.DAO**, create a class and call it **StudentDAO**. This class is going to be used to search the csv files for student’s information only.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No. | Return Type | Class Name | Method Name | Input Parameters |
| 1 | List<Student> | StudentDAO | getStudents - This method reads the Students.csv file and returns the data as a List<Student> | None |
| 2 | Student | StudentDAO | getStudentByEmail – This method takes a Student’s email as a String and the List of Students as an ArrayList and parses the List for a Student with that email and returns a Student Object. | List<Student> studentList,  String email |
| 3 | boolean | StudentDAO | validateUser – This method takes the List of Students and two other parameters: the first one is the user email and the second one is the password from the user input. Return whether or not student was found | List<Student> studentList,  String email,  String pass |

Under the package named: **CoreJava.DAO**, create a class and call it **CourseDAO**. This class is going to be use to query the database for course’s information only.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No. | Return Type | Class Name | Method Name | Input Parameters |
| 1 | List<Course> | CourseDAO | getAllCourses – This method takes no parameter and returns every Course in the table. | None |

Under the package named: **CoreJava.DAO**, create a class and call it **AttendingDAO**. This class is going to be use to query the database for Attending’s information.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No. | Return Type | Class Name | Method Name | Input Parameters |
| 1 | List<Attending> | AttendingDAO | getAttending –  This method reads the Attending.csv file and returns the data as a List<Attending> | None |
| 2 | void | AttendingDAO | registerStudentToCourse – This method takes a Student’s email and a Course ID. It checks if a Student with that Email is currently attending a Course with that ID.  If the Student is not attending that Course, add a new Attending object with the Student’s Email and Course ID to the List. | List<Attending> attending,  String student\_email,  int course\_id |
| 3 | List<Course> | AttendingDAO | getStudentCourse –  This method takes a Student’s Email as a parameter and would search the Attending List for all the courses a student is registered to base on the Id.  Each of these is added to a new List of courses. This list of courses the Student is attending is returned | List<Attending> attending,  String studentEmail |
| 4 | Void | AttendingDAO | saveAttending –  This method overwrites the original Attending.csv file with the new data | List<Attending> attending |

**Requirement 3:**

**Main Entry**

Inside the package named: **CoreJava.MainEntryPoint**, there is a class named: **MainRunner**. When your code is complete, this class will be used to run the School Management System. None of the code in this class should be modified, and it should therefore only be used to test your code after you’ve finalized everything.

In the same package, there is also a class named **TestRunner**. Feel free to use this class to test your code as much as you’d like. Feel free to make changes. The content of the TestRunner class will not be factored into or used at all for your grade. Feel free to copy any of the code **from** MainRunner **into this class** if you’d like to try making any modifications.

**Sample:** Students. Once a student is logged in, the student is able to see all the courses she/he is registered to. Two options are available 1. Register to Class and 2. Logout. If option 1 is selected, then the student is able to see all the courses and register to any of them.

**Example Workflow:**

Are you a(n)

1. Student

2. quit

Please, enter 1 or 2.

1

Enter Your Email:

J@gmail.com

Enter Your Password:

333

My Classes:

# COURSE NAME INTRUCTOR NAME

1 GYM Mark

2 Math Luke

1. Register to Class

2. Logout

1

All Courses:

ID COURSE NAME INSTRUCTOR NAME

1 GYM Mark

2 Math Luke

3 Science Stephanie

4 English Lisa

Which Course?

3

My Classes:

# COURSE NAME INTRUCTOR NAME INSTRUCTOR EMAIL

1 GYM mark mark@gmail.com

2 Math Luke luke@gmail.com

3 Science Stephanie Stephanie@gmail.com

You have been signed out.