BẢN BÁO CÁO ASSIGNMENT – COURSE REGISTRATION SYSTEM

Nhóm 4
Thành viên: Kiên, Duy, Khiêm, Đức
Lớp: SE2042

---

1. Giới thiệu

Đề tài Course Registration System là một ứng dụng Java dạng ***menu-driven***, cho phép người dùng quản lý danh sách sinh viên, môn học, lớp học phần và việc đăng ký môn học của sinh viên.
Mục tiêu của hệ thống là hỗ trợ việc tra cứu, quản lý thông tin học tập một cách nhanh chóng và hiệu quả, đồng thời có thể đọc/ghi dữ liệu từ file **`.txt`** để đảm bảo tính lưu trữ lâu dài.

2.Yêu cầu hệ thống

Hệ thống được thiết kế để đáp ứng các yêu cầu sau:

- Quản lý thông tin sinh viên, môn học, lớp học phần: thêm, sửa, xóa.
- Đăng ký hoặc hủy đăng ký lớp học phần cho sinh viên.
- Giới hạn sinh viên không được đăng ký quá 20 tín chỉ trong một học kỳ.
- Cho phép xem điểm trung bình học kỳ hoặc điểm trung bình toàn khóa.
- Sắp xếp sinh viên theo điểm trung bình.
- Đọc/ghi dữ liệu từ file (.txt).
- Có menu điều hướng chính và menu con cho từng nhóm chức năng.

---

3. Tổ chức package của dự án

Dự án được tổ chức theo mô hình module hóa rõ ràng nhằm dễ bảo trì và mở rộng, gồm các package chính:

```
courseregistrationsystem/                # Package chính của chương trình
└── CourseRegistrationSystem.java        # Lớp khởi chạy (main class)

data/                                    # Package chứa các lớp mô tả dữ liệu
(Data/Classes)
```

```
│
├── Person.java                  # Lớp cha trừu tượng cho các đối tượng người
├── Student.java                 # Lớp mô tả thông tin sinh viên
├── Subject.java                 # Lớp mô tả thông tin môn học
├── CourseSection.java           # Lớp mô tả thông tin lớp học phần
└── Registration.java            # Lớp mô tả thông tin đăng ký học phần

enums/                           # Package chứa các kiểu liệt kê (Enumeration)
│
├── DayOfWeek.java               # Ngày học trong tuần (MON → SAT)
├── RegistrationStatus.java      # Trạng thái đăng ký (ENROLLED, PASSED,
FAILED, WITHDRAWN)
└── StudentStatus.java           # Trạng thái sinh viên (ACTIVE, INACTIVE,
GRADUATED)

interfaces/                      # Package chứa các interface hỗ trợ
│
├── Displayable.java             # Interface hiển thị thông tin đối tượng
├── FileSerializable.java        # Interface hỗ trợ đọc/ghi dữ liệu từ file
└── Identifiable.java            # Interface bắt buộc đối tượng có getId()

manager/                         # Package xử lý logic quản lý dữ liệu
│
├── Management.java              # Lớp generic quản lý danh sách đối tượng
(List<T>)
├── StudentManager.java          # Quản lý sinh viên
├── SubjectManager.java          # Quản lý môn học
├── CourseManager.java           # Quản lý lớp học phần
└── RegistrationManager.java     # Quản lý đăng ký học phần

util/                            # Package chứa các tiện ích (Utilities)
│
├── FileHandler.java             # Đọc và ghi dữ liệu từ file(.txt)
├── Menu.java                    # Hiển thị menu chính, điều hướng chương trình
└── Validator.java               # Kiểm tra và xác thực dữ liệu đầu vào (input
validation).
```

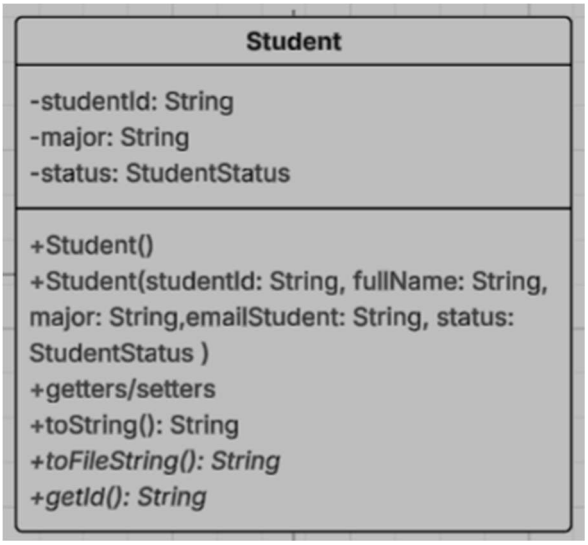
---

4. Thiết kế hệ thống

Hệ thống được thiết kế theo mô hình **hướng đối tượng (OOP)**, gồm các nhóm lớp chính:

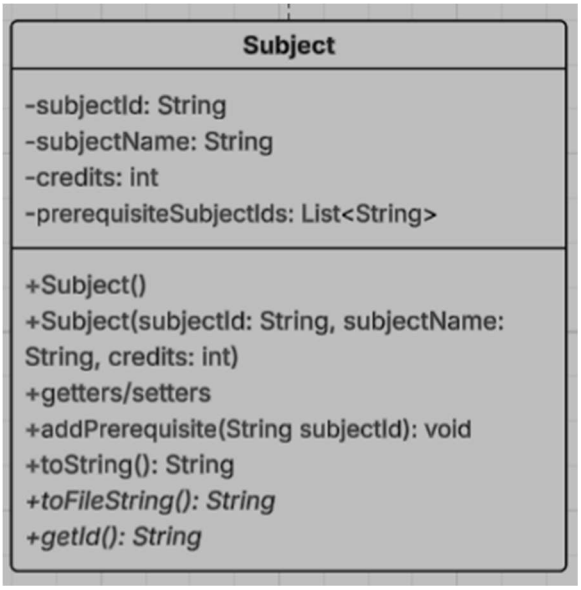
POJO (Đối tượng Java cơ bản)

- **Student**



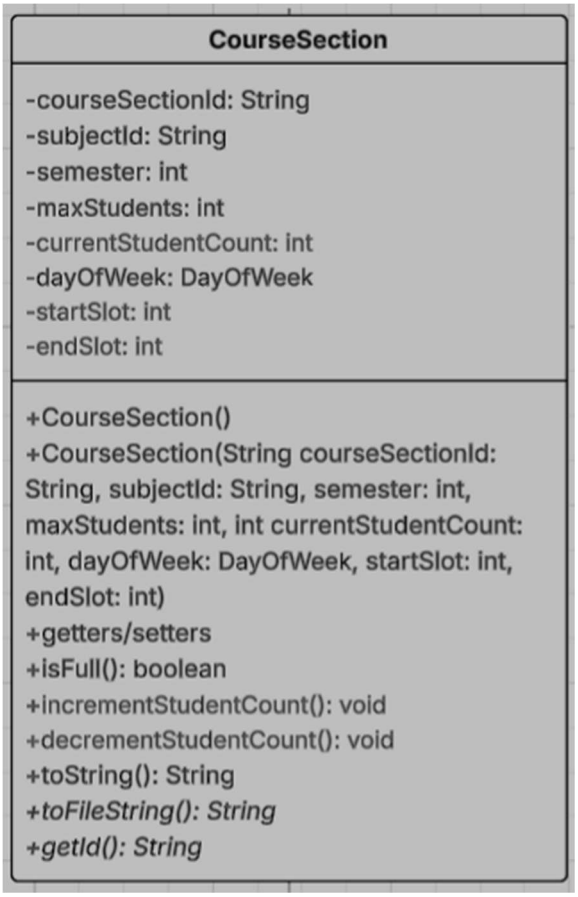
  - **Thuộc tính**: `studentId`, `fullName`, `major`, `email`, `status: StudentStatus`
  - **Chức năng**: Lưu thông tin sinh viên, xuất dữ liệu ra file, triển khai `Identifiable` và `FileSerializable`.

- **Subject**



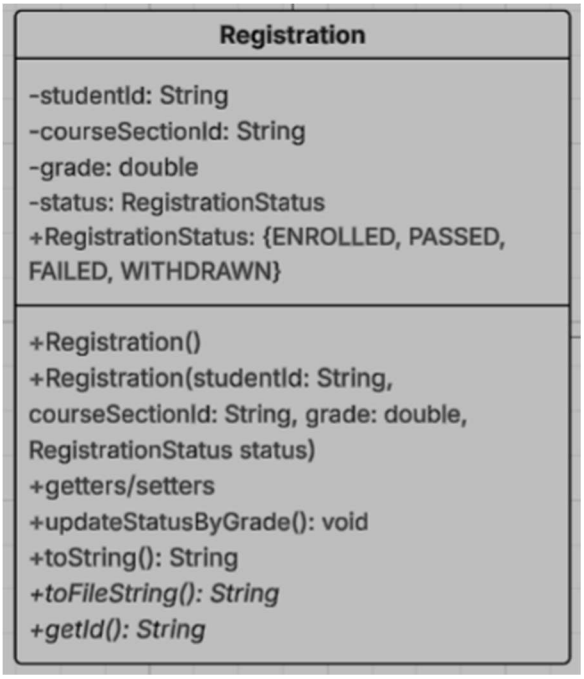
  - **Thuộc tính**: `subjectId`, `subjectName`, `credits`, `prerequisiteSubjects`
  - **Chức năng**: Quản lý thông tin môn học và các môn tiên quyết.

- **CourseSection**



  - **Thuộc tính**: **`courseSectionId`**, **`subjectId`**, **`semester`**, **`maxStudents`**, **`currentStudentCount`**, **`dayOfWeek`**, **`startSlot`**, **`endSlot`**
  - **Chức năng**: Quản lý thông tin lớp học phần, kiểm tra sĩ số, lịch học.

- **Registration**

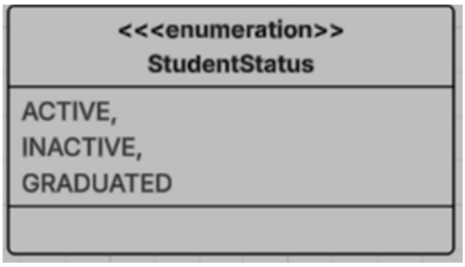


- **Thuộc tính**: `studentId`, `courseSectionId`, `grade`, `status: RegistrationStatus`
- **Chức năng**: Ghi nhận việc sinh viên đăng ký môn học và cập nhật điểm.

Lớp tổng hợp: `Enumerations`

Mục đích: Lớp `Enumerations` (hoặc package `enums`) được tạo ra để chứa toàn bộ các enum được sử dụng xuyên suốt hệ thống — giúp mã nguồn có tổ chức, dễ bảo trì và rõ ràng hơn.

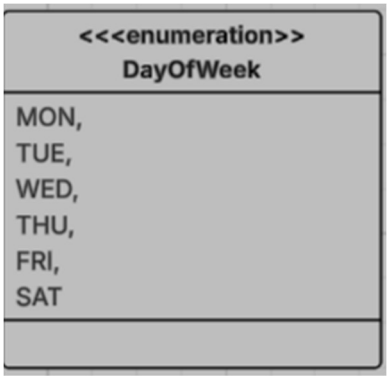
1. Enumeration Class: StudentStatus



- Được Sử dụng trong:
  `Student` → Thuộc tính `status: StudentStatus`
- **Thuộc tính:**
  - **ACTIVE:** Sinh viên đang học.

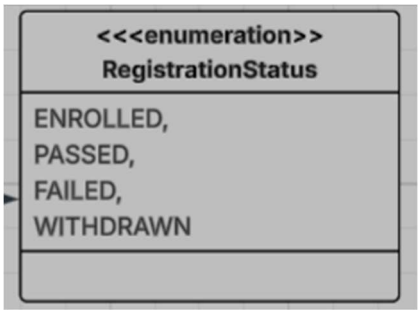
- **INACTIVE:** Sinh viên tạm ngưng học.
- **GRADUATED:** Sinh viên đã tốt nghiệp.

2. Enumeration Class: DayOfWeek



- Được Sử dụng trong:
  `CourseSection` → Thuộc tính `dayOfWeek: DayOfWeek`
- **Thuộc tính:**
  - **MON**: Thứ hai.
  - **TUE**: Thứ ba.
  - **WED**: Thứ tư.
  - **THU**: Thứ năm.
  - **FRI**: Thứ sáu.
  - **SAT**: Thứ bảy.

3. Enumeration Class: RegistrationStatus

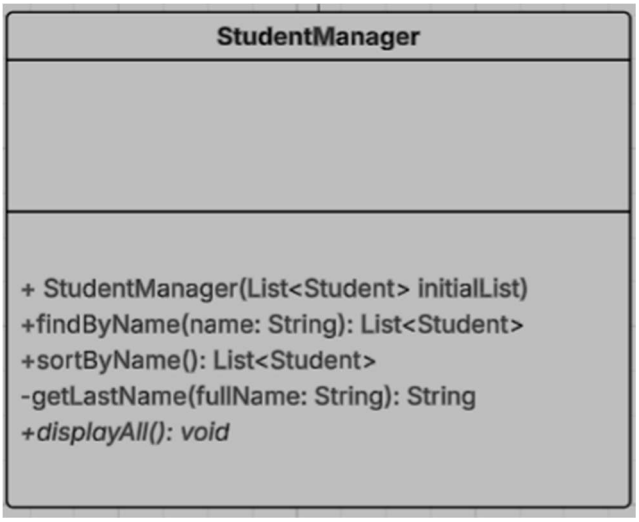


- Được Sử dụng trong:
  `Registration` → Thuộc tính `status: RegistrationStatus`
- **Thuộc Tính:**
  - **ENROLLED:** Học sinh đang học.
  - **PASSED:** Học sinh đã qua môn.
  - **FAILED:** Học sinh đã rớt môn.

- **WITHDRAWN:** Học sinh đã rút môn khi mà chỉ đăng ký vào lớp nhưng chưa học.

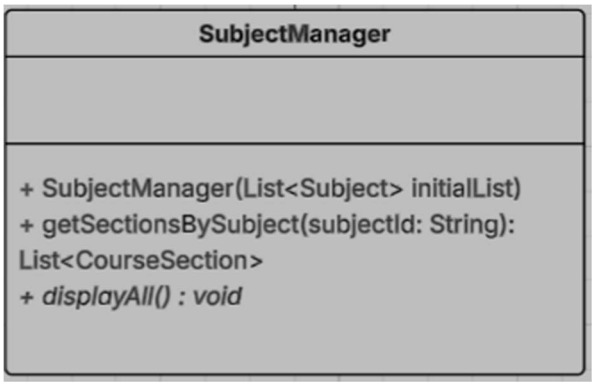
Managers (Lớp quản lý)

1. StudentManager



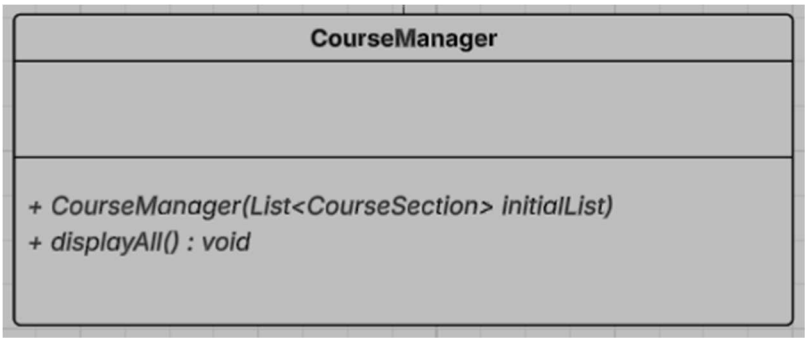
- **Chức năng:**
  Quản lý danh sách sinh viên và cung cấp các thao tác tìm kiếm, sắp xếp, và hiển thị thông tin sinh viên.
- **Nhiệm vụ chính:**
  - Tìm kiếm các sinh viên có tên hoặc họ tên chứa chuỗi ký tự được nhập vào.
  - Sắp xếp danh sách sinh viên theo thứ tự bảng chữ cái, hỗ trợ việc hiển thị và tra cứu thuận tiện.
  - Lấy phần tên cuối cùng của sinh viên để phục vụ mục đích sắp xếp và hiển thị.
  - Hiển thị toàn bộ danh sách sinh viên ra màn hình.

2. SubjectManager



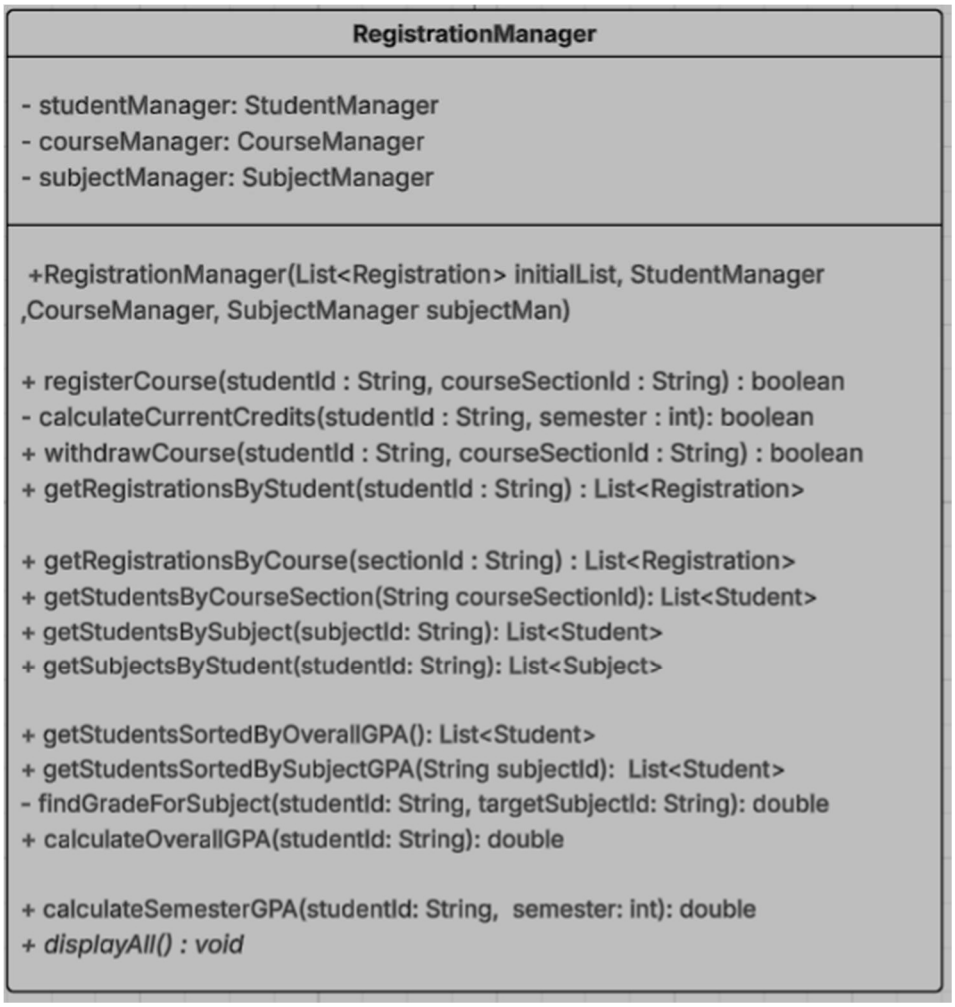
- **Chức năng:** Quản lý các học phần và cung cấp khả năng truy xuất danh sách lớp học (`CourseSection`) theo từng môn học.
- **Nhiệm vụ chính:**
    - Lấy danh sách các lớp học (course sections) thuộc một môn học cụ thể thông qua **`subjectId`.**
    - Hiển thị toàn bộ danh sách môn học và các lớp học tương ứng.

3. CourseManager



- **Chức năng:** Quản lý danh sách các lớp học phần (`List<CourseSection>`).
- **Nhiệm vụ chính:**
    - Giữ dữ liệu về các lớp học phần, bao gồm ngày học (`dayOfWeek`), khung giờ (`startSlot`, `endSlot`) , sĩ số (`maxStudents`), và danh sách sinh viên đã đăng ký.
    - Cung cấp phương thức lấy thông tin thời khóa biểu của lớp (`getScheduleInfo()`).
    - Kiểm tra sĩ số lớp học (`isFull()`) .
    - Cho phép thêm sinh viên mới vào lớp khi đăng ký thành công.

4. RegistrationManager



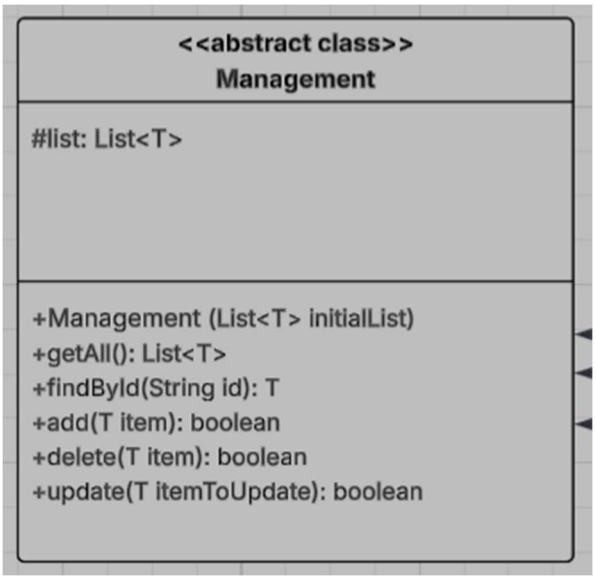
- **Chức năng:** Quản lý toàn bộ hoạt động đăng ký học phần của sinh viên, bao gồm việc đăng ký, hủy đăng ký, và tính toán kết quả học tập (GPA, tín chỉ).

- **Nhiệm vụ chính:**

    - Cho phép sinh viên đăng ký / hủy đăng ký vào một lớp học phần.
    - Tính tổng số tín chỉ hiện tại mà sinh viên đã đăng ký trong một học kỳ.
    - Lấy danh sách đăng ký học phần của một sinh viên cụ thể.
    - Lấy danh sách sinh viên đang học trong một môn học cụ thể.
    - Tính điểm trung bình tích lũy (GPA toàn khóa) của sinh viên.
    - Tính điểm trung bình học kỳ cho sinh viên.
    - Hiển thị toàn bộ danh sách đăng ký học phần trong hệ thống.

Management<T> (Lớp trừu tượng):

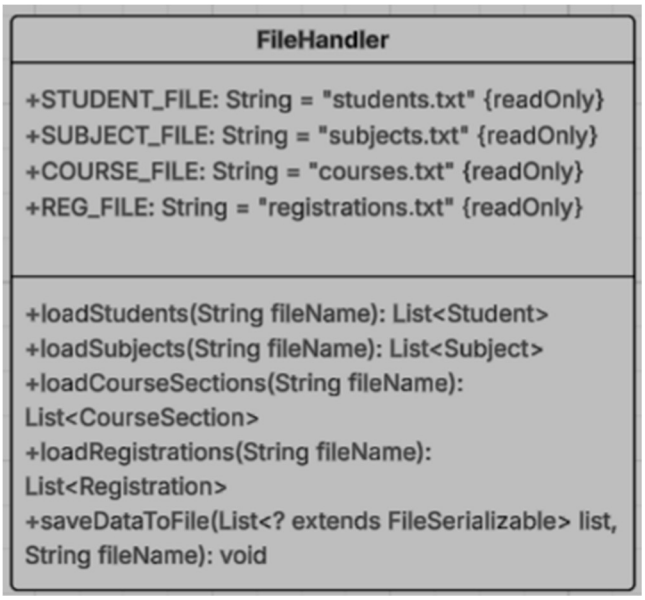
Lớp **`Management<T>`** được thiết kế dưới dạng **lớp trừu tượng (abstract)** và sử dụng **Generic (T)**, cho phép tái sử dụng linh hoạt cho nhiều kiểu dữ liệu khác nhau.



- **Chức năng:** Cung cấp các phương thức quản lý chung cho các lớp con như **`StudentManager`**, **`SubjectManager`**, và **`RegistrationManager`**.
- **Nhiệm vụ chính:**
  - Trả về toàn bộ danh sách đối tượng hiện có.
  - Tìm kiếm một đối tượng theo ID duy nhất.
  - Thêm một đối tượng mới vào danh sách.
  - Xóa một đối tượng ra khỏi danh sách.
  - Cập nhật thông tin đối tượng trong danh sách.

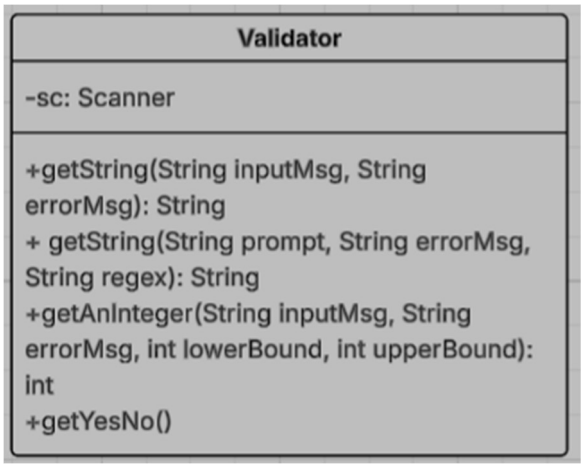
Utilities (Lớp tiện ích)

- **FileHandler**:



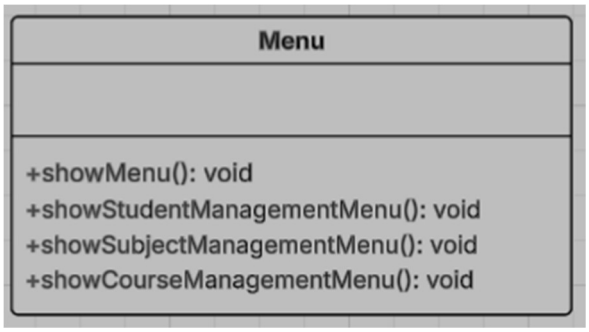
- **Chức năng:**
  Đảm nhiệm việc đọc và ghi dữ liệu giữa hệ thống và các tệp văn bản (text files), đảm bảo quá trình lưu trữ và khôi phục dữ liệu diễn ra nhất quán và an toàn.
- **Nhiệm vụ chính:**
  - Đọc dữ liệu từ tệp sinh viên và chuyển đổi thành danh sách đối tượng **`Student`.**
  - Đọc dữ liệu từ tệp môn học và tạo danh sách **`Subject`**.
  - Đọc thông tin lớp học phần từ tệp và chuyển đổi thành danh sách **`CourseSection`**.
  - Đọc danh sách đăng ký học phần của sinh viên từ tệp.
  - Ghi dữ liệu của danh sách đối tượng (có thể là **`Student`**, **`Subject`**, **`CourseSection`** hoặc **`Registration`**) trở lại file tương ứng, giúp lưu trữ thông tin thay đổi của hệ thống.

- **Validator**



- **Chức năng:**
  Xử lý việc kiểm tra, xác thực và lấy dữ liệu đầu vào từ người dùng, đảm bảo dữ liệu hợp lệ trước khi đưa vào hệ thống.
- **Nhiệm vụ chính:**
  - Nhập chuỗi từ bàn phím, kiểm tra rỗng và hiển thị thông báo lỗi nếu không hợp lệ.
  - Nhập chuỗi và xác thực theo biểu thức chính quy (regex).
  - Nhập số nguyên nằm trong khoảng giới hạn cho phép.
  - Xác nhận câu trả lời Yes/No từ người dùng.

- **Menu**:

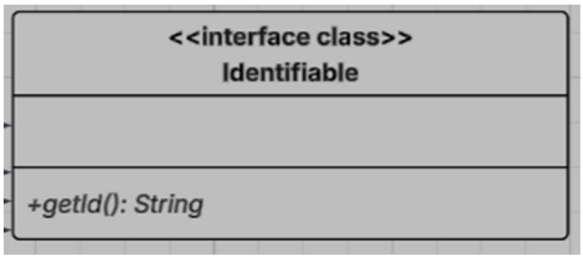


- **Chức năng**: Lớp `Menu` chịu trách nhiệm hiển thị và điều hướng toàn bộ giao diện dòng lệnh (console interface) của hệ thống.
- **Nhiệm vụ chính:**
  - Hiển thị **menu chính** của chương trình, bao gồm các lựa chọn như "Quản lý sinh viên", "Quản lý môn học", "Quản lý lớp học phần", "Đăng ký học phần", và "Thoát chương trình".
  - Hiển thị menu con dành cho **quản lý sinh viên**, cho phép người dùng thêm, xóa, sửa, sắp xếp hoặc tìm kiếm sinh viên theo tên.

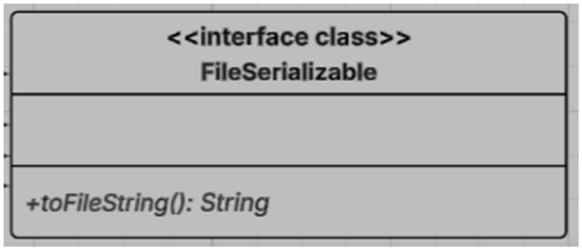
- Hiển thị menu con dành cho **quản lý môn học**, bao gồm các chức năng thêm môn, chỉnh sửa thông tin, hoặc xem danh sách các lớp học phần của một môn.
- Hiển thị menu con dành cho **quản lý lớp học phần**, nơi người dùng có thể xem danh sách lớp học phần, kiểm tra sĩ số, lịch học và các thông tin khác liên quan đến từng lớp.

Interfaces & Abstracts

- **Identifiable**:

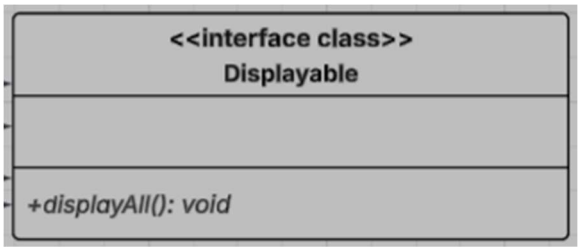


  - **Mục đích**: Xác định các đối tượng có thuộc tính ID duy nhất trong hệ thống.
  - **Các phương thức chính**:
    - String **getId()**: Lấy ID của đối tượng.
  - **Được sử dụng trong**: `Student, Subject, CourseSection, Registration.`
- **FileSerializable**:



  - **Mục đích:** Hỗ trợ đối tượng chuyển đổi thành **chuỗi định dạng** để ghi ra file `.txt`.
  - **Các phương thức chính:**
    - `String toFileString():` Chuyển đổi tượng thành chuỗi lưu trữ.
  - **Được sử dụng trong:** Tất cả các lớp dữ liệu (POJO) như `Student`, `Subject`, `CourseSection`, `Registration`.

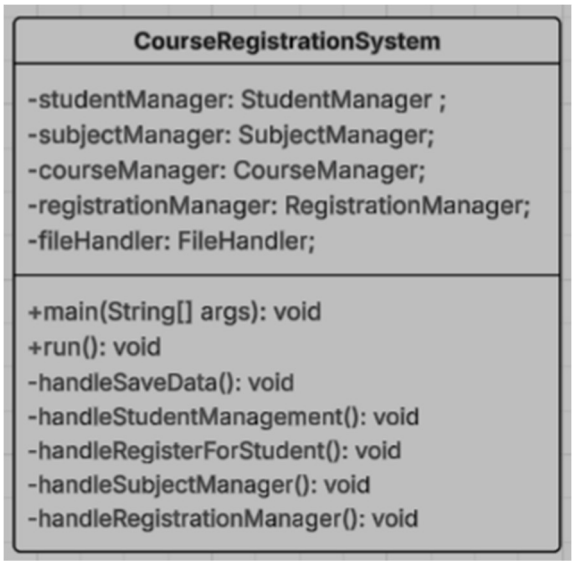
- **Displayable**:



- **Mục đích:** Quy định hành vi hiển thị thông tin của đối tượng trên màn hình console.
- **Các phương thức chính:**
  - `void display()`**:** In thông tin chi tiết của đối tượng ra màn hình.
- **Được sử dụng trong:** `Student`**,** `Subject`**,** `CourseSection`**,** `Registration`**,** để chuẩn hóa cách xuất dữ liệu.

Main Controller

- **CourseRegistrationSystem**: Là lớp điều phối chính, chịu trách nhiệm khởi tạo dữ liệu, hiển thị menu, xử lý luồng điều khiển và ghi lại dữ liệu khi thoát chương trình.
  - **Thuộc tính (Attributes):**
    - `studentManager`: quản lý danh sách sinh viên và các thao tác liên quan.
    - `subjectManager`: quản lý danh sách môn học và lớp học phần.
    - `courseManager`: quản lý thông tin lớp học phần.
    - `registrationManager`: quản lý đăng ký học phần của sinh viên.
    - `fileHandler`: xử lý đọc/ghi dữ liệu từ file `.txt`.
  - **Phương thức (Methods):**
    - `main(String[] args)`: điểm vào chương trình, khởi chạy hệ thống.
    - `run()`: hiển thị menu và điều khiển luồng xử lý chính.
    - `handleSaveData()`: lưu toàn bộ dữ liệu hiện tại ra file.
    - `handleStudentManagement()`: điều phối các chức năng quản lý sinh viên.
    - `handleRegisterForStudent()`:điều phối chức năng đăng ký/hủy đăng ký học phần.
    - `handleSubjectManager()`: điều phối các chức năng quản lý môn học.
    - `handleRegistrationManager()`: điều phối các thao tác quản lý đăng ký học phần.



---

5. Các chức năng chính của chương trình

5.1. Quản lý CRUD (Thêm, Sửa, Xóa)

- **Mô tả:** Cho phép người dùng thêm, sửa, xóa sinh viên, môn học, hoặc lớp học phần.
- **Cách Triển khai:**
  - Dùng **`Menu`** để lấy lựa chọn người dùng.
  - Dùng **`Validator`** để kiểm tra dữ liệu đầu vào.
  - Gọi các phương thức từ **`StudentManager`**, **`SubjectManager`**, hoặc **`CourseManager`** để thực hiện thao tác.

5.2. Đăng ký và Hủy đăng ký môn học

- **Mô tả:** Sinh viên có thể đăng ký lớp học phần nếu thỏa mãn các điều kiện (không trùng lịch, không quá tín chỉ, đủ tiên quyết...).
- **Cách Triển khai:**
  - Thực hiện qua **`RegistrationManager`**, có sự phối hợp giữa **`StudentManager`**, **`SubjectManager`**, và **`CourseManager`**.
  - Kiểm tra tuần tự 5 điều kiện: trùng lịch, tiên quyết, giới hạn tín chỉ, sĩ số, trạng thái sinh viên.

5.3. Xử lý logic nghiệp vụ

- **Mô tả:** Thực hiện các kiểm tra và tính toán trong hệ thống như tổng tín chỉ, xếp loại, điểm trung bình, hoặc danh sách môn đã qua.
- **Cách Triển khai:** Toàn bộ do các lớp ***Manager*** đảm nhận — đặc biệt là **`RegistrationManager`**.

5.4. Sắp xếp và Tìm kiếm

- **Mô tả:** Cho phép sắp xếp sinh viên theo điểm trung bình hoặc tìm kiếm sinh viên/môn học theo ID.
- **Cách Triển khai:**
  - Dùng `Collections.sort()` hoặc `Comparator` trong các lớp Manager.
  - Các thao tác tìm kiếm dựa trên `findById()` của `Management<T>`.

5.5. Đọc/Ghi File

- **Mô tả:** Lưu trữ toàn bộ dữ liệu hệ thống vào file `.txt` và đọc lại khi khởi chạy.
- **Cách Triển khai:**
  - Dùng `FileHandler` để đọc/ghi dữ liệu, kết hợp `FileSerializable` để ghi định dạng thống nhất.

---

6. Luồng đi của chương trình

Luồng hoạt động tổng thể

1. Khi chương trình khởi chạy, **FileHandler** đọc dữ liệu từ các file `.txt`.
2. Dữ liệu được truyền vào 4 lớp Manager tương ứng.
3. `CourseRegistrationSystem.run()` gọi `Menu` để hiển thị giao diện điều khiển.
4. Người dùng thao tác CRUD, đăng ký, sắp xếp, hoặc xem thông tin.
5. Khi thoát, toàn bộ dữ liệu được ghi đè lại vào file.

Phân tầng hệ thống

- **Tầng UI (Giao diện):** `CourseRegistrationSystem`, `Menu`, `Validator`.
- **Tầng Manager (Xử lý logic):** 4 lớp Manager thừa kế `Management<T>`.
- **Tầng Data (Dữ liệu):** `Student`, `Subject`, `CourseSection`, `Registration`.
- **Tầng IO (Lưu trữ):** `FileHandler`.

Hệ thống **Course Registration System** được chia thành 4 luồng xử lý chính, thể hiện quá trình vận hành của toàn bộ chương trình từ khi khởi chạy đến khi kết thúc.

Luồng 1: Load Data (Nạp dữ liệu)

1. Khi chương trình bắt đầu, phương thức `main()` trong lớp **CourseRegistrationSystem** được gọi.
   Câu lệnh duy nhất trong `main()` là: `new CourseRegistrationSystem().run();`
2. Câu lệnh này khởi tạo đối tượng `CourseRegistrationSystem`, đồng thời gọi **constructor (hàm khởi tạo)**.
3. Trong constructor, chương trình tạo một đối tượng **FileHandler**, sau đó lần lượt gọi các hàm:
   `loadStudents()`, `loadSubjects()`, `loadCourseSections()`, và `loadRegistrations()`.
4. Mỗi hàm đọc dữ liệu từ file `.txt`, chuyển đổi nội dung thành các đối tượng tương ứng, rồi trả về **List** dữ liệu.
5. Các danh sách này được truyền vào bốn lớp quản lý tương ứng:
   - `StudentManager(listStudents)`
   - `SubjectManager(listSubjects)`
   - `CourseManager(listCourses)`
   - `RegistrationManager(listRegistrations)`
6. Sau khi hoàn tất, toàn bộ dữ liệu từ file đã được nạp vào bộ nhớ (RAM). Hệ thống sẵn sàng xử lý.

Luồng 2: Xử lý chính (Main Processing Flow)

1. Sau khi khởi tạo xong, phương thức `run()` bắt đầu thực thi.
2. Một vòng lặp `while(true)` được dùng để hiển thị menu và chờ người dùng thao tác.
3. Lớp **Menu** chịu trách nhiệm hiển thị giao diện lựa chọn (Main Menu và các menu con).
4. Lớp **Validator** được sử dụng để kiểm tra dữ liệu đầu vào từ người dùng, đảm bảo hợp lệ.
5. Sau khi người dùng chọn chức năng, hệ thống dùng **cấu trúc** `switch-case` để điều phối đến các module tương ứng, ví dụ:
   - Quản lý sinh viên → `StudentManager`
   - Quản lý môn học → `SubjectManager`
   - Quản lý lớp học phần → `CourseManager`
   - Đăng ký môn học → `RegistrationManager`

Luồng 3: Xử lý logic đăng ký môn học (Business Logic Flow)

1. Khi người dùng chọn chức năng **Đăng ký môn học**, hệ thống gọi lớp **Validator** để nhập vào `studentId` và `courseId`.
2. Sau đó, lớp **CourseRegistrationSystem** gọi đến phương thức `registerCourse(studentId, courseId)` trong **RegistrationManager**.
3. Tại đây, hệ thống lần lượt thực hiện **5 bước kiểm tra logic nghiệp vụ** trước khi cho phép đăng ký:

   a. Kiểm tra trùng lịch học:

- Gọi **`CourseManager`** để lấy thông tin lịch học (ngày, tiết bắt đầu, tiết kết thúc).
- So sánh với danh sách lớp mà sinh viên đã đăng ký trong cùng kỳ để phát hiện trùng lịch.

b. Kiểm tra môn tiên quyết:

- Gọi **`SubjectManager`** để lấy danh sách các môn tiên quyết (**`prerequisiteSubjects`**).
- Kiểm tra trong lịch sử học tập xem sinh viên đã hoàn thành các môn này hay chưa.

c. Kiểm tra giới hạn tín chỉ:

- Tính tổng số tín chỉ sinh viên đã đăng ký trong kỳ hiện tại cộng với môn mới.
- Nếu vượt quá 20 tín chỉ thì hệ thống từ chối đăng ký.

d. Kiểm tra sĩ số lớp học phần:

- Kiểm tra **`currentStudentCount`** < **`maxStudents`**. Nếu lớp đã đủ, không cho phép đăng ký thêm.

e. Kiểm tra trạng thái sinh viên:

- Gọi **`StudentManager`** để kiểm tra trạng thái sinh viên (**`StudentStatus`**).
- Nếu sinh viên đang bị đình chỉ hoặc tạm dừng học thì sẽ không được phép đăng ký.

4. Chỉ khi tất cả 5 điều kiện trên đều hợp lệ, hệ thống mới tạo đối tượng **`Registration`** mới và thêm vào danh sách đăng ký.
5. Phương thức **`registerCourse()`** trả về **true** nếu đăng ký thành công.
6. **`CourseRegistrationSystem`** nhận kết quả và hiển thị thông báo **"Đăng ký thành công!"** cho người dùng.

Luồng 4: Kết thúc và lưu dữ liệu (Save Data Flow)

1. Khi người dùng chọn **"0. Thoát"**, chương trình sẽ gọi đến phương thức `handleSaveData()` trong **CourseRegistrationSystem**.
2. Phương thức này lần lượt gọi **FileHandler** để ghi đè dữ liệu ra 4 file `.txt`:
   - **`students.txt`**
   - **`subjects.txt`**
   - **`courseSections.txt`**
   - **`registrations.txt`**
3. Cụ thể, FileHandler được gọi như sau:

```
fileHandler.saveDataToFile(studentManager.getAll(), "students.txt");
```

4. Phương thức `saveDataToFile()` sử dụng **tính đa hình (Polymorphism)** của Java:
    - Nó nhận tham số là `List<? extends FileSerializable>.`
    - Khi gọi `item.toFileString(),` Java tự động xác định phương thức phù hợp (Student, Subject, CourseSection…).
5. Sau khi ghi đè toàn bộ dữ liệu xuống file, chương trình kết thúc quá trình chạy.
6. Hệ thống tắt an toàn, đảm bảo dữ liệu luôn được cập nhật mới nhất.

---

7. Kết quả đạt được

- Hoàn thiện hệ thống **quản lý học phần** bằng Java.
- Thực hiện thành công các chức năng **CRUD, đăng ký học phần, ghi/đọc file, và xác thực dữ liệu**.
- Áp dụng **đầy đủ nguyên lý OOP**: kế thừa, đa hình, trừu tượng, đóng gói.
- Hệ thống có cấu trúc rõ ràng, dễ mở rộng và bảo trì.

---

8. Kết luận

Dự án **Course Registration System** giúp nhóm hiểu rõ hơn về lập trình hướng đối tượng trong Java, cách tổ chức lớp, xử lý dữ liệu và quản lý tệp tin.