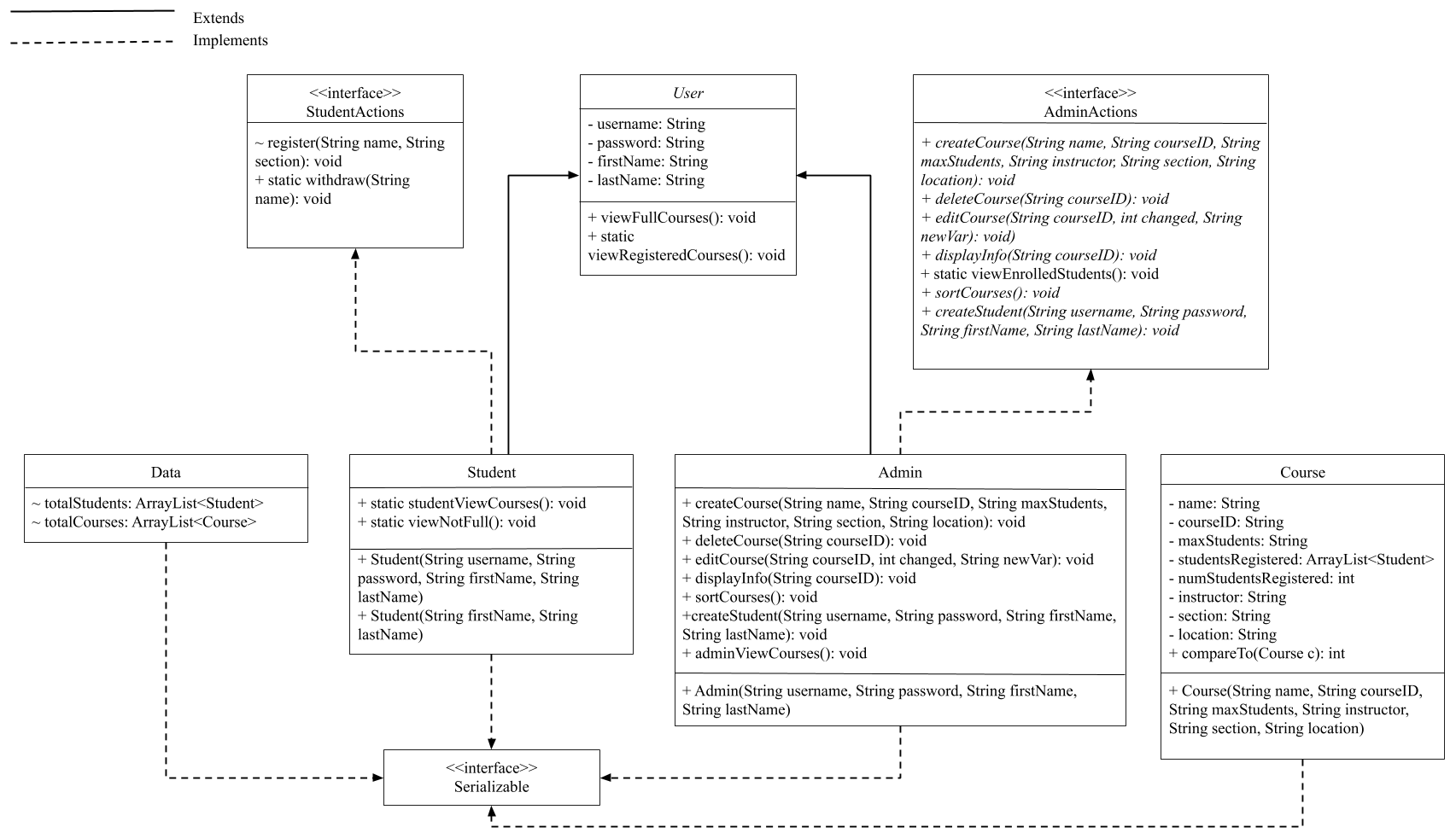
This course registration system consists of eight .java files: two interfaces, one abstract class, four concrete classes, and one used for running the program. Below is a UML diagram of the program.



The program starts by running the testing.java file. In the main method, the following is performed:

If serialized courses exist, call oldStart(), else call newStart(). Regardless, call logIn().

Methods within testing.java:

+ static newStart(): void

* Deserializes the provided “MyUniversityCourses.csv” file and stores the information into Data.totalCourses

+ static oldStart(): void

* Calls loadCourses and loadStudents

+ static save(): void

* Serializes Data.totalCourses and Data.totalStudents into “Courses.ser” and “Students.ser” respectively.

+ static loadCourses(String file): void

* Deserializes the “Courses.ser” file created during a previous session and stores the information into Data.totalCourses

+ static loadStudents(String file): void

* Deserializes the “Students.ser” file created during a previous session and stores the information into Data.totalStudents

+ static logIn(): void

* Asks the user if they are an admin or a student, then requests a username and password. If they are an admin, listAdminActions is called. If they are a student, listStudentActions is called. There is also the option to exit the system, which calls save() and then exits the program.

+ static listAdminActions(): void

* Provides the user with the option to choose between 11 options:

1: Create Course //calls createCourse (in Admin class)

2: Delete Course //calls deleteCourse (in Admin class)

3: Edit Course //calls editCourse (in Admin class)

4: Display Info On Specific Course //calls displayInfo (in Admin class)

5: View All Courses //calls adminViewCourses (in Admin class)

6: View Enrolled Students in a Course //calls viewEnrolledStudents (in AdminActions)

7: Create a New Student //calls createStudent (in Admin class)

8: View Full Courses //calls viewFullCourses (in User class)

9: View Courses a Student has Registered For //calls viewRegisteredCourses (in User class)

10: Sort Courses //calls sortCourses (in Admin class)

11: Log Out //calls save and logIn (both in testing class)

\*Every function except “Log Out” calls listAdminActions at the end

+ static listStudentActions(): void

* Provides the user with the option to choose between 6 options:

1: Register in a course //calls register (in StudentActions)

2: Withdraw from a course //calls withdraw (in StudentActions)

3: View all courses //calls studentViewCourses (in Student class)

4: View all courses that are not full //calls viewNotFull (in Student class)

5: View all courses you are registered in //calls viewRegisteredCourses (in User class)

6: Log Out //calls save and logIn (both in testing class)

\*Every function except “Log Out” calls listStudentActions at the end