Registration System

Software Design Patterns

CPIT-252

Students:

Yousif Bogari - 2136374.

Abdulaziz saddig jastanieh - 2135813.

Instructor:

Dr. Rizwan Qureshi.

Introduction:.....2
Problem Definition:.....2
Suggested solution:.....3
Main Functionalities:.....3
Class Diagrams:.....4
.....6
Code:.....7
Expected Results:.....7

Introduction:

At the start of every semester, in the first few weeks, students must always go through the process of registering courses. This process Is anxiety inducing and stressful, as students always fear missing out on good Sections or Good lecture times. So students Rush to the faculty, to go to the Student affairs who are responsible for registering courses for students, and to the waiting line at the start of every semester. And every student that goes in starts asking the same questions: “what are the sections available?” , “what are their times?” “What Are The Good College Free Courses?” and so , This process can take so much time, and the students' precious time is being wasted. And Doctors have to delay the start of the lectures as not all the students have registered for the course, which causes a delay in the Schedule.

Problem Definition:

The problem is that there isn’t a Simple, Good and easy to use Registration system that the student can access anytime and anywhere in order to register a course, the systems that are being used by the students to register the courses are not suitable for Mobile use. And so, when the students try to use them from their mobile phones, they feel clunky and hard to use. That's because they are made with the intent that they will be used from a laptop or a personal computer, and not a mobile phone. And that is not the only issue, Using them from your mobile phone can take a lot of your phone's hardware resources E.g: CPU , RAM.

Suggested solution:

We will Design and build a Simple Registration system with Our focus being on Mobile use and User experience, as well as Reducing the usage of mobile phone resources, so that students can register courses from anywhere at any time with ease.

Main Functionalities:

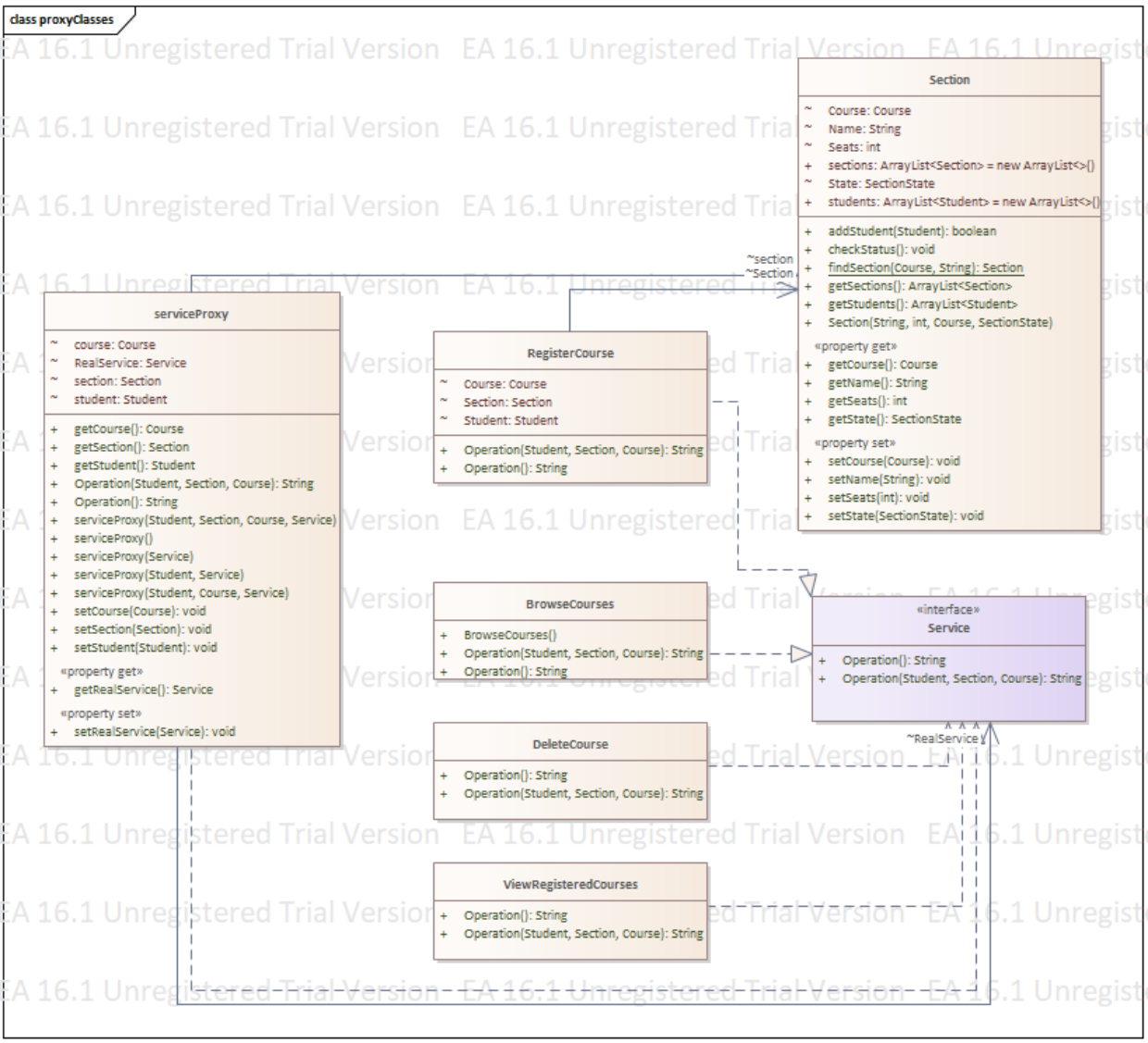
1- The student can Browse through the list of courses Available for Registration.

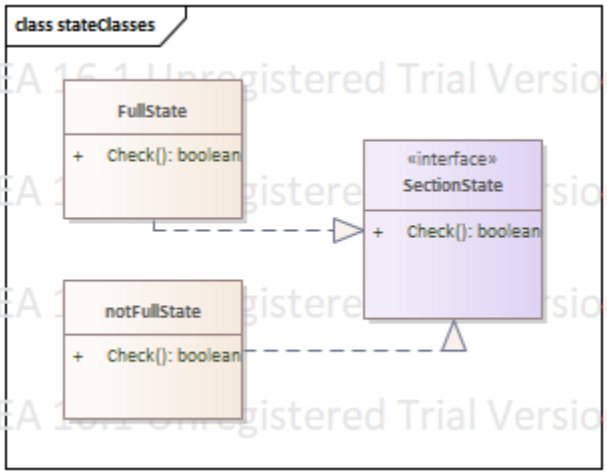
2- The student can Register a Course at a certain Section.

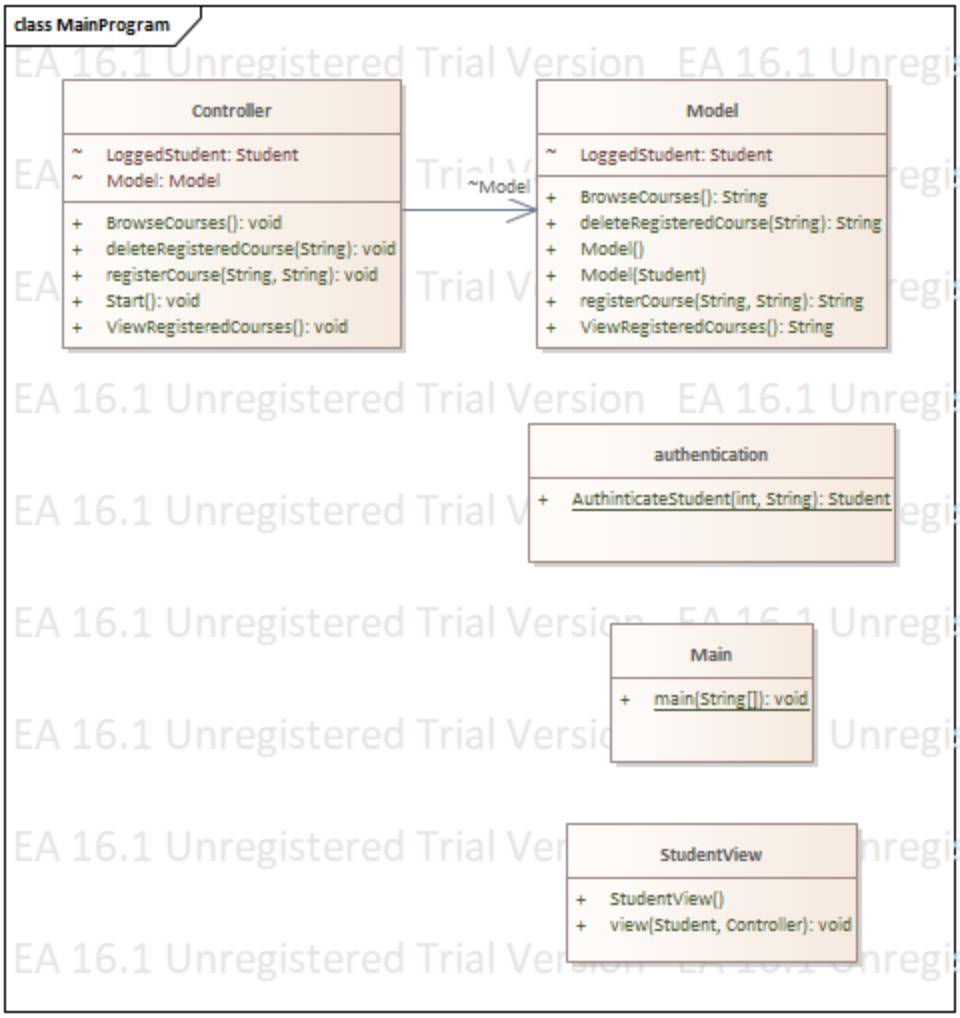
3- The student can see the list of Registered Courses.

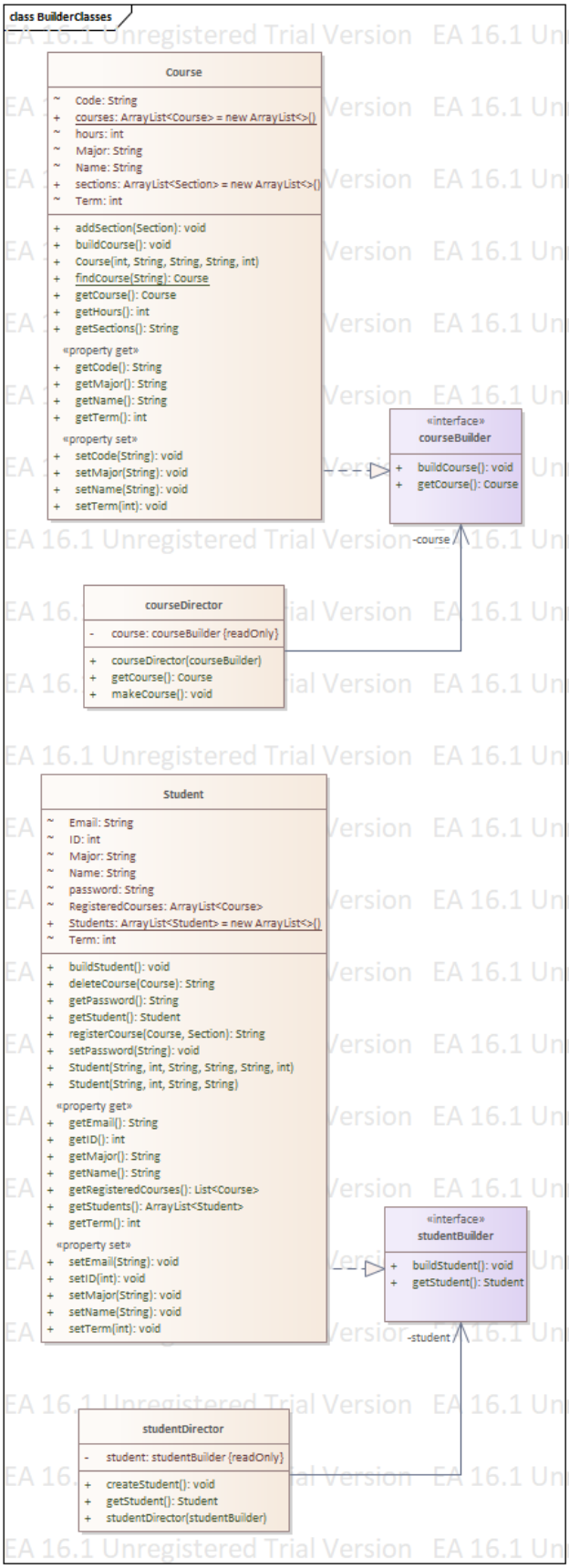
4- The student can Delete a Registered Course.

Class Diagrams:






class MainProgram
Controller
~ LoggedStudent: Student
~ Model: Model
+ BrowseCourses(): void
+ deleteRegisteredCourse(String): void
+ registerCourse(String, String): void
+ Start(): void
+ ViewRegisteredCourses(): void
~Model
Model
~ LoggedStudent: Student
+ BrowseCourses(): String
+ deleteRegisteredCourse(String): String
+ Model()
+ Model(Student)
+ registerCourse(String, String): String
+ ViewRegisteredCourses(): String
authentication
+ AuthinticateStudent(int, String): Student
Main
+ main(String[]): void
StudentView
+ StudentView()
+ view(Student, Controller): void


class BuilderClasses
Course
~ Code: String
+ courses: ArrayList<Course> = new ArrayList<>()
~ hours: int
~ Major: String
~ Name: String
+ sections: ArrayList<Section> = new ArrayList<>()
~ Term: int
+ addSection(Section): void
+ buildCourse(): void
+ Course(int, String, String, String, int)
+ findCourse(String): Course
+ getCourse(): Course
+ getHours(): int
+ getSections(): String
«property get»
+ getCode(): String
+ getMajor(): String
+ getName(): String
+ getTerm(): int
«property set»
+ setCode(String): void
+ setMajor(String): void
+ setName(String): void
+ setTerm(int): void
«interface»
courseBuilder
+ buildCourse(): void
+ getCourse(): Course
-course
courseDirector
- course: courseBuilder {readOnly}
+ courseDirector(courseBuilder)
+ getCourse(): Course
+ makeCourse(): void
Student
~ Email: String
~ ID: int
~ Major: String
~ Name: String
~ password: String
~ RegisteredCourses: ArrayList<Course>
+ Students: ArrayList<Student> = new ArrayList<>()
~ Term: int
+ buildStudent(): void
+ deleteCourse(Course): String
+ getPassword(): String
+ getStudent(): Student
+ registerCourse(Course, Section): String
+ setPassword(String): void
+ Student(String, int, String, String, String, int)
+ Student(String, int, String, String)
«property get»
+ getEmail(): String
+ getID(): int
+ getMajor(): String
+ getName(): String
+ getRegisteredCourses(): List<Course>
+ getStudents(): ArrayList<Student>
+ getTerm(): int
«property set»
+ setEmail(String): void
+ setID(int): void
+ setMajor(String): void
+ setName(String): void
+ setTerm(int): void
«interface»
studentBuilder
+ buildStudent(): void
+ getStudent(): Student
-student
studentDirector
- student: studentBuilder {readOnly}
+ createStudent(): void
+ getStudent(): Student
+ studentDirector(studentBuilder)

Code:

The Program and codes will be Available at Github:

https://github.com/usiifo/CPIT-252-Project

Expected Results:

Students will be able to register courses easily without the need to visit the students affairs and waste so much time, which will result in students being Satisfied and happy, and Doctors can start giving lectures early and be on schedule.