

**Cairo University**

**Faculty of Computers and Artificial Intelligence**

**Department of Computer Sciences**

**A Better E-com**

*Implemented by*

|  |  |
| --- | --- |
| 20170191 | فارس سيد حسن السيد |
| 20170301 | منير صبري يوسف أحمد |
| 20170307 | مينا بطرس فرحات فلتس |
| 20170174 | عمر حسن حسني عبد العال |
| 20170198 | كريم محمد فتحي مسعود |

Supervised by

**Dr. Basheer**

**TA. Hassan Mourad**

Graduation Project

Academic Year 2020-2021

Final Documentation

Abstract

The old-bylaw E-com current features are not working well, in addition to that it misses important features that the stakeholders of the E-com and us as students need.

The new-bylaw E-com or simply the new E-com fixed some of the old E-com features while making the user interface simpler and better looking.

However, the current (new) E-com focuses mainly on the E-community part while ignoring almost completely the E-learning part.

For example, the current E-com does not support material, assignments, or exams.

This led us as stakeholders of the E-com to use external websites like Acadox, Google Classroom, and Blackboard.

Those websites provided E-community services like course materials, releasing, and submitting assignments, holding quizzes and exams, some of them even support online lectures, however, these websites designed to be generic to be accessible from all different kind of universities and faculties with different and contradicting rules.

Therefore, these websites fail on the E-com services part like registering students to courses with specific rules or recording the attendance and grades of the students.

This project and the work done in it is a step on the long journey to create a software product for our faculty that combines the features from the E-community focused website like the old and new E-com and the E-learning focused websites like blackboard that provides the E-community services that forces our faculty rules and bylaw while having enough E-learning services so that the E-com become a self-sufficient website that service all the stakeholders needs in one user-friendly environment.

# Introduction

In this chapter, we will cover the environment of the project, the motivation behind our work, the existing problem, the goal of the project, the timeline and steps done in the process of development.

## Project Area

The world wide web (Internet) has made sharing and communicating with other people a lot more convenient.

You can communicate with someone in the other side of the globe while being at your home. Thus, Web development (the process of making a web site) has huge demand on it, especially at the current time of the pandemic.

In the area of E-learning, many websites have been developed to use the power of the Internet in this field like Google Classroom, Blackboard, etc. Probably all universities have their own websites.

Our college website, E-com, is lacking in both the look and the features, with missing features and non-working ones. There was an attempt in the new E-com to overcome those issues but there are still some issues to solve.

Our college is not an exception, it has its own website, the old-bylaw E-com, the old-bylaw E-com has served the college for many years before finally got replaced by the new-bylaw E-com.

## 1.2 Motivation

Through our years in the faculty, we moved back and forth between different official and external websites for handling the learning process of our courses.

We as students used the official old-bylaw E-com for registering the courses, viewing our attendance and grades, we used several websites at the same time for viewing and downloading materials, delivering assignments, taking exams, and attending online lectures.

The process of tracking all these websites at the same time was fatiguing and time wasting, it also increased the gap between the students and the instructors since it made the process of communication between each other slow and unresponsive.

## Problem Definition

The old-bylaw E-com was a quick-developed solution for our faculty to make use of the Internet to enhance the learning process.

It has a lot of features that the are either do not work well or underdeveloped, for example, the website handles the course registration, however, the process of finding the suitable courses for the student and registering the student into the course is manually done.

It supports adding materials and exchanging messages between the stakeholders; however, the execution of the features is not user-friendly or easy.

The new-bylaw E-com (or simply the new E-com) has finally came and replaced the old E-com, however, the problem is it focuses almost completely on the E-community part like registering students and recording attendance and grades.

So that, we as stakeholders had to use external websites like Acadox, Google Classroom, Blackboard to handle the process of distributing materials, delivering assignments, and holding online quizzes and exams.

The problem with these external websites that they focus mainly on the E-learning part while ignoring almost completely the E-community part, which prevents us from being able to completely migrate to one of these third-party websites.

## Project Objective

The goal of the project is to build a software solution that combines the elements of the E-community and the E-learning in one environment.

This environment should satisfy the needs of the stakeholders of the E-com from enforcing the faculty rules and bylaw to managing materials and holding exams.

The software product of this project should be self-sufficient to its stakeholders and be user-friendly.

## Gantt Chart

## 1.6 Used Tools

In developing the project, we wanted to use the latest technologies to prepare ourselves for post-graduation work, we considered many different viable options to do the specified requirements like PHP (Laravel framework), JAVA (JSP or SPRING) and C# (asp.net or .net core) we decided to design the system on restful API called by front end framework and a database server we decided it has to be remote database for flexibility we considered firebase and MySQL.

Technologies

After careful consideration we decided to use C# (.net core) for the backend API being called by Angular front end as they are the most used and very good and solid technologies, for the database we used MySQL remote database server remotemysql.com.

IDEs

As for the IDEs for developing C# backend we used Visual Studio 2019 Community Edition since they work really well together and to benefit of the powerful debugger for Angular front end, we used Visual Studio Code and For

Database development we used MySQL Workbench and MySQL shell.

Other Software

That’s not all the programs that we used there are also programs we used for Testing the API and designing diagrams and use case tables.

For testing the API, we used postman for its wide range of features and automated testing, for designing diagrams we used Visual Paradigm, for the use case tables we used Microsoft word, and for source controlling and sharing code we used GitHub.

## 1.7 Report Organization

Related Work

In the following pages we will describe discuss similar products to ours

That we were inspired by and learned from.

System Analysis

We will describe our project’s specifications the functional requirements we compiled through looking and analyzing various sources like (E-comm, Google Classes, Acadox, Blackboard), then we will specify the non-functional requirements we found necessary like (Security, Usability, Robustness), Then we will show the Use case Diagrams of each actor in out system.

System Design

We will show an overall picture of how our system is designed, what parts it consists of and how they interact with each other with Component Diaram, the classes in the system and their relations with the class diagrams, how the database is structured with the entities in it with the ERD, and we will show screenshots of how we designed the graphical user interface of our system.

Implementation and Testing

We will show samples of our system while it’s working and applied test cases of the many test cases, we used through developing the system.

# 2. Related Work

E-comm

As the name of our project suggests the closest project to ours is the current E-comm (old-bylaw) it’s the one we used through our college years and the main motivation behind our project we inspired a lot of the functionality from E-comm we added some functionality that we wished existed like a feed four each individual course, Handling attendances individually as a student would know which Lab or Section, he/she was recorded as absent on.

Others

We looked at many E-learning platforms most notably Black board of course as it’s the one we currently use, Acadox as we used it a lot also and google classrooms There was a lot to note and be impressed by the extensive features that they provide however they didn’t handle department or suggest the courses to you and the course registration process was rather long as they need to handle all different faculties not just FCAI.

We wanted to take what is useful to students of FCAI from the similar systems from E-comm we implemented the department, attendance, course recommendation and the general feed among other things, from other E-learning platforms we really found that each course students should have isolated course feed instead of having to use other means.

# 4. System Analysis

In this chapter we will show the analysis of our project with Functional requirements we collected, Non-functional requirements we found important and use case diagrams.

## 3.1 Project Specification

### 3.1.1 Functional Requirements

|  |  |
| --- | --- |
| FR1 | The admin must be able to insert new student into the system with the student high school type, national id, name, birth date. |
| FR2 | The admin must be able to insert instructor into the system with the instructor national id, name, birth date, department, university and graduation year |
| FR3 | The admin must be able to create account for student that already exist in the database and initial password is national id |
| FR4 | The admin must be able to create account for instructor that already exist in the database and initial password is national id |
| FR5 | The admin must be able to reset the credentials of the account of a student by setting its password to national id |
| FR6 | The admin must be able to reset the credentials of the account of an instructor by setting its password to national id |
| FR7 | All user type must be able to Login the system using their password initially set to national id |
| FR8 | All users must be able to view the info of their profile after they log in to the system |

|  |  |
| --- | --- |
| FR9 | All Users must be able to change their Email Address, Address, Phone number, Mobile number and their additional info at any time or set them to empty. |
| FR10 | All Users must be able to change their password with providing their old password and the new one. |
| FR11 | The system must recommend the applicable Courses provided he/she finished its prerequisites |
| FR 12 | The admin must be able to add course to department with course code, name and description |
| FR13 | All Users must be able to get the code info and prerequisites of a course using course code |
| FR 14 | All Users must be able to search course name and get course info of all courses with given name |
| FR 15 | The admin must be able to update course department code, course academic year, course description, with course code |
| FR16 | The admin must be able to update the course Prerequisites by giving a full new list of Prerequisites. |
| FR17 | The admin must be able to update the course department applicability by giving a list |
| FR18 | The admin must be able to set the archived status of a course by giving status |
| FR19 | The admin must be able to set the department of a registered student |

|  |  |
| --- | --- |
| FR20 | The admin must be able to add an instance of a course by giving course code, year, current term |
| FR21 | All users must be able to view whether an instance of a course is open for registration |
| FR22 | The admin must be able to mark and instance of a course as closed for registration |
| FR23 | The admin must be able to mark an instance of a course as open for registration |
| FR24 | The student must be able to submit a priority list for their desired departments |
| FR25 | The admin must be able to view the department priority list belonging to any student |
| FR26 | The student must be able to view only their own department priority list |
| FR27 | The student must be able to update their submitted department priority list |
| FR28 | All users must be able to view all courses given by a department giving the department code. |
| FR29 | The student must be able to view the courses available for them to register |
| FR30 | The admin must be able to view the courses available for any student to register given student id |

|  |  |
| --- | --- |
| FR31 | All users must be able to view the available instances of a given course by course code |
| FR32 | The student must be able to view the courses in which he/she is registered in |
| FR33 | The admin must be able to view the course in which a give student is registered in by student id |
| FR34 | The student must be able to view the instances of courses that he/she is registered in |
| FR35 | The admin must be able to view the instances of courses that any given student is registered in |
| FR36 | The admin must be able to view if normal course registration is open in the faculty |
| FR37 | The student must be able toe view if normal course registration is open in the faculty |
| FR38 | The admin must be able to view if late course registration is open in the faculty |
| FR39 | The student must be able to view if late course registration is open in the faculty |
| FR40 | The student must be able to register themselves to an instance of an available course |
| FR41 | The admin must be able to register any student to an instance of an available course |

|  |  |
| --- | --- |
| FR42 | The student must be able to drop themselves of a course instance if it’s open for registration |
| FR43 | The admin must be able to drop any student from an instance of a course if it’s open for registration |
| FR44 | All users must be able to view students registered in a given course instance |
| FR45 | The admin must be able to view all late course registration requests submitted by students |
| FR46 | The admin must be able to view the late course registrations request by student id |
| FR47 | The student must be able to submit late course registration request |
| FR48 | The student must be able to delete their late course registration request |
| FR49 | The admin must be able to set the status of a late course registration request |
| FR50 | The student can view their status regarding a given instance of a course |
| FR51 | The admin and instructor must be able to view the status of a student regarding a given instance of a course |
| FR52 | The admin must be able to set the status of a given student regarding an instance of a course |

|  |  |
| --- | --- |
| FR53 | The instructor must be able to view the courses in which he/she is registered in |
| FR54 | The instructor must be able to view the courses in which any instructor is registered in |
| FR55 | The instructor must be able to view the course instance in which he/she is registered in |
| FR56 | The admin must be able to view the course instance in which any instructor is registered in |
| FR57 | The instructor must be able to view the course instances of a given course in which he/she is registered in |
| FR58 | The admin must be able to view the course instances of a given course in which any instructor is registered in |
| FR59 | The admin must be able to register the instructor into an instance of a course |
| FR60 | The admin must be able to drop the instructor from an instance of a course |
| FR61 | All users must be able to view all instructors registered in a given instance of a course |
| FR62 | All users must be able to view whether or not an instance of a course is set to read only |
| FR63 | The admin must be able to set any instance of a given course as read only |

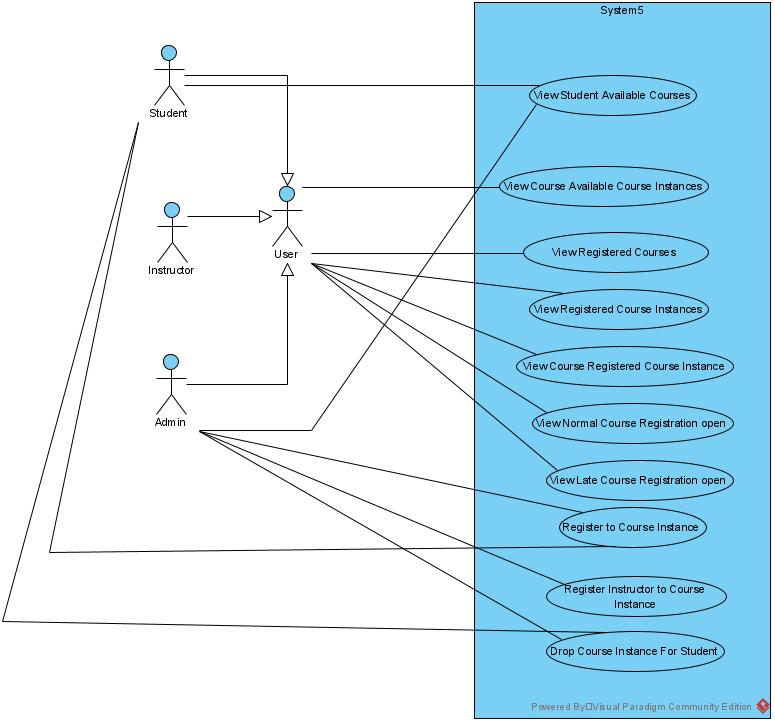
|  |  |
| --- | --- |
| FR64 | The instructor must be able to set course instance in which he/she is registered in as read only |
| FR65 | All users must be able to view the faculty general feed without needing to login |
| FR66 | The admin must be able to type a post on the general feed of the faculty |
| FR67 | The admin must be able to delete a post from the general feed of the faculty |
| FR68 | The student must be able to view the feed of an instance of course that he/she is registered in |
| FR69 | The instructor must be able to view the feed of an instance of a course that he/she is registered in |
| FR70 | The admin must be able to view the feed of any instance of any course without needing to be registered |
| FR71 | The instructor must be able to add to the feed of an instance of a course the he/she is registered in |
| FR72 | The admin must be able to add to the feed of any instance of any course |
| FR73 | The instructor must be able to delete a post from the feed of an instance of a course that he/she is registered in |
| FR74 | The admin must be able to delete a post from the feed of any instance of any course |

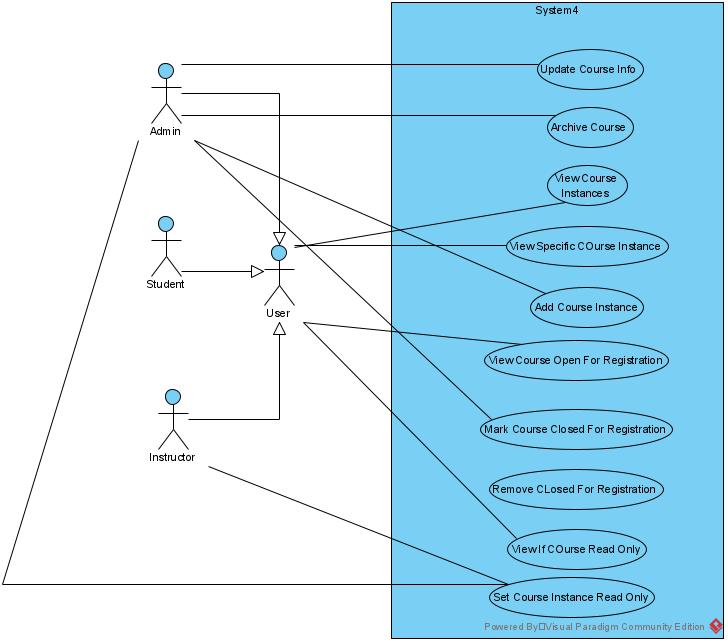
|  |  |  |
| --- | --- | --- |
| FR65 | | The student must be able to view their attendance item in an instance of a course that they are registered in |
| FR66 | | The instructor must be able to view the attendance items of any student in an instance of a course that they are registered in |
| FR67 | | The admin must be able to view the attendance items of any student of an instance of any course |
| FR68 | | The instructor must be able to add an attendance item to an instance of a course that they are registered in |
| FR69 | | The admin must be able to add an attendance item to an instance of any course |
| FR70 | | The instructor must be able to set the attendance of a student in an attendance item in an instance of a course that they are registered in |
| FR 71 | | The admin must be able to set the attendance of a student in an attendance item in an instance of any course |
| FR72 | | The instructor must be able to delete an attendance item in an instance of a course that they are registered in |
| FR73 | | The admin must be able to delete an attendance item in an instance of any course |
| FR74 | | The student must be able to view their own grade regarding an instance of a course that they registered in |
| FR75 | | The instructor must be able to view the grade of any student in an instance of a course that they are registered in |
| FR76 | The admin must be able to view the grad of any student in any instance of any course | |
| FR77 | The admin must be able to set the grade of any student in any instance of any course | |
| FR78 | The student must be able to view their own GPA | |
| FR79 | The admin must be able to view the GPA of any student in that system | |

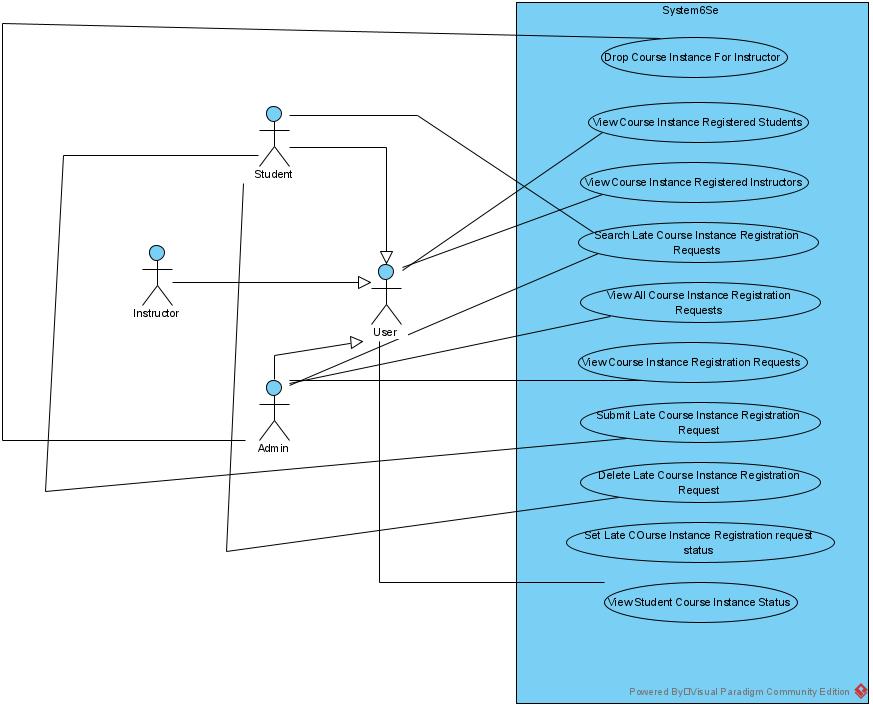
### 3.1.2 Non-functional Requirements

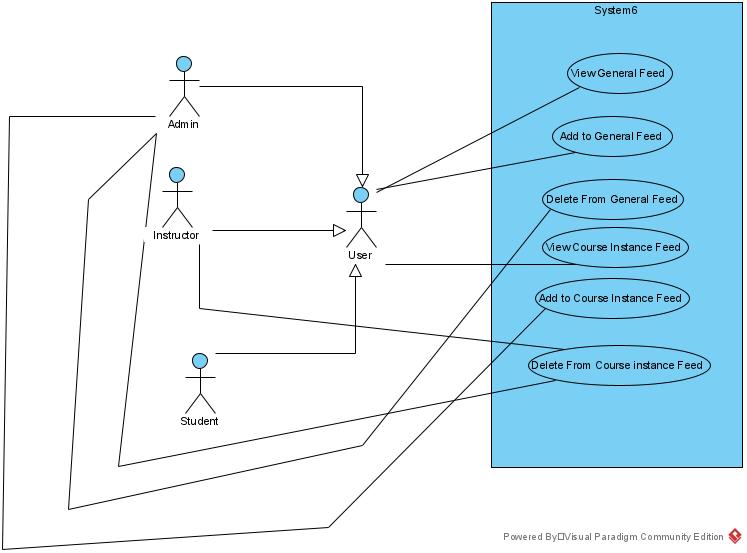
1. **Usability:** System should be easy even for new students that are not tech savvy.
2. **Performance:** The system should handle assignment turn-in in workload without a problem.
3. **Consistency:** Functions that have similar steps should have similar interfaces.
4. **User Friendly:** The design and colors of the interface should be suitable for all ages and for the user’s eyes.
5. **Robustness:** The system should be able to handle invalid user input.
6. **Security:**  The passwords should be hashed before storing them in the database and API invokes must carry authentication token given at login

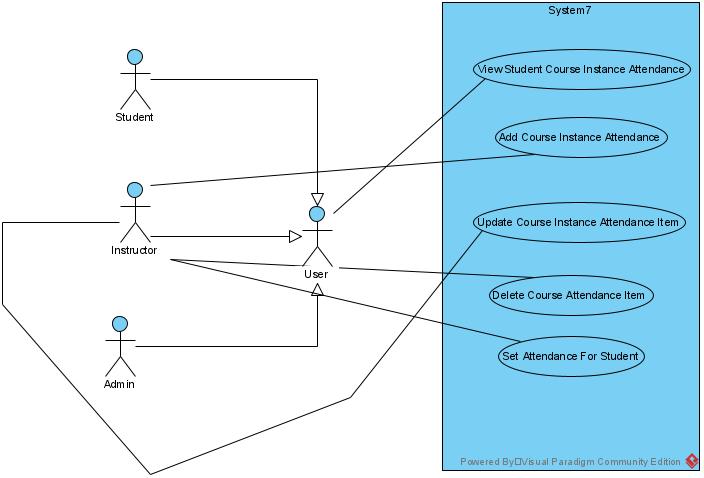
## 3.2 Use case Diagrams

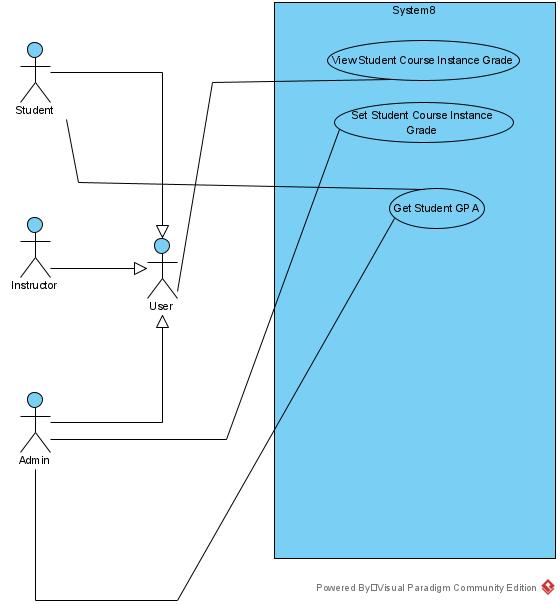






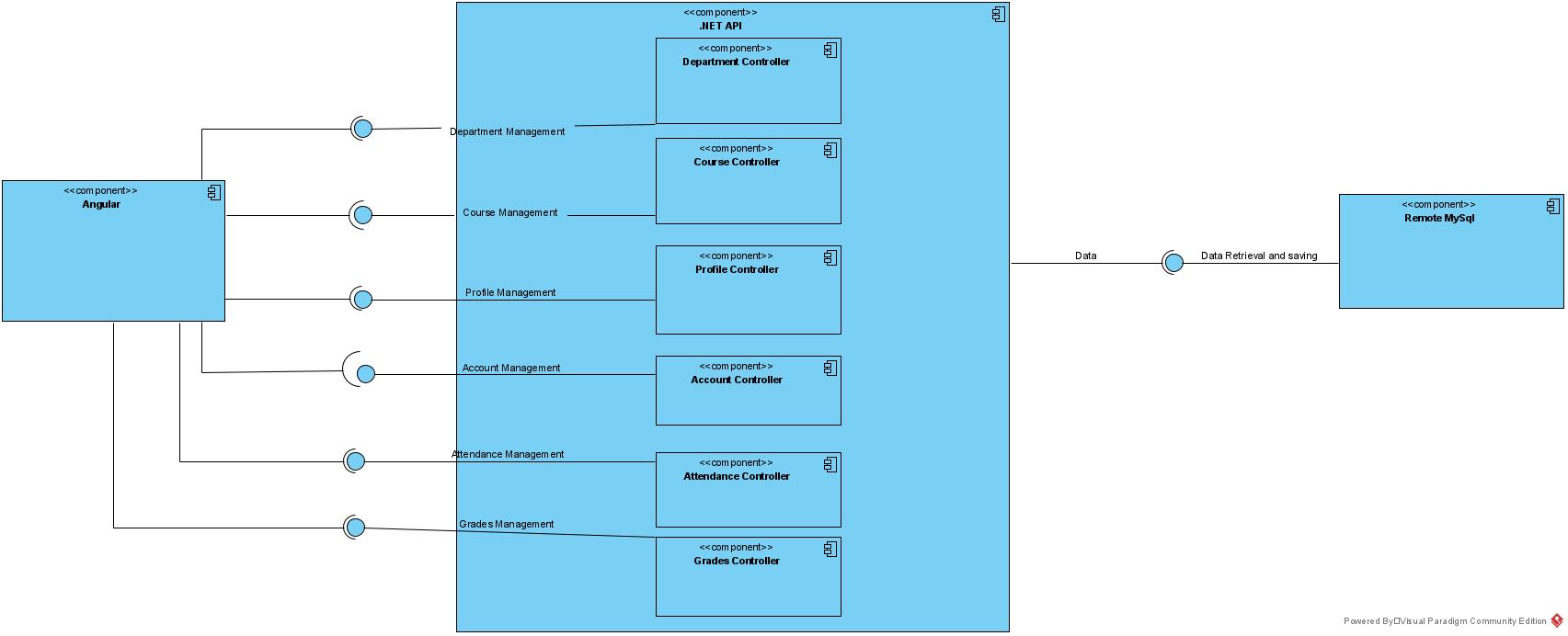






# 4. System Design

### Component Diagram



### Class Diagrams