Team K

Authors: John Badji | Fherdinand J. Corcuera | Kelvin Wu

CUNY Zero
Software Requirements Specification
For a Graduate Program Management System

Version 1.0

Revision History

Date	Version	Description	Author(s)
24/10/21	1.0	Initial draft	KW, FC, JB

Table of Contents

Software Requirements Specification

1. Introduction

This document contains the software requirements specifications for the CUNY Zero software system. In this section, the system purpose and scope, definitions, references, and a general overview of the document can be found.

1.1 Purpose

This document lays out the functional and nonfunctional requirements allocated to the development of a brand-new software named CUNY Zero by the company Team K. The next sections include specific requirements, constraints, and other factors along with use cases that describe and detail the varieties of user interactions that CUNY Zero software must deliver.

There are multiple intended audiences for this document. It serves as a basis for an agreement between the development and the marketing team. In addition, developers should refer to this document in order to get a clear and thorough understanding of what is expected from the CUNY Zero project and finally, this document will be reviewed and approved by the project management team.

1.2 Scope

This software requirements specification applies to the web application being developed called CUNY Zero. CUNY Zero is an online platform which provides numerous tools for managing academic aspects for a graduate program in a college. This software allows students to manage their academic such as searching, enrolling, and dropping courses. It gives faculty or instructors a means of managing and organizing the students and the courses they are teaching. It gives registrars a way to administer the ins and outs of the graduate institution that this software applies to. And anyone can find out about general basic information related to courses, students, faculty, as well as enrollment options through this platform.

1.3 Definitions, Acronyms, and Abbreviations

***GPA*:** Grade Point Average, a number that indicates how well or how high a student in their courses on average.

Participating Actors: The people or the system who interact with the use case

***Flow of Events*:** Step by step action and interaction between the participating actors and the system

***Instructor**:* someone who is licensed to be able teach at the University.

***Postconditions*:** What happens or the result when the process completes successfully

***Preconditions*:** Things that need to have before the use case can initialize the process.

***Registrar**:* someone who is responsible for the flow of the System

***Student**:* someone who is registered to take courses at the University.

***Title*:** Communicates the goal and purpose of the use case.

1.4 References

Team K. (2021). *Use-Case* Diagram. Visual Paradigm Online. https://online.visual-paradigm.com/drive/#diagramlist:proj=0&new=UseCaseDiagram

The City College of New York. (n.d). *GradSchoolZero: a graduate program management system*. Proj_spec.docx. https://www.dropbox.com/s/8uudhe6x021thug/proj_spec.docx?dl=0

1.5 Overview

This document is intended for selected individuals including stakeholders, project management team, marketing team and software developers.

Section 2 of this document provides profound details on the CUNY Zero system from an overall point of view. Product functions, user characteristics, constraints, assumptions, dependencies, and general requirements are defined from the system-level perspective.

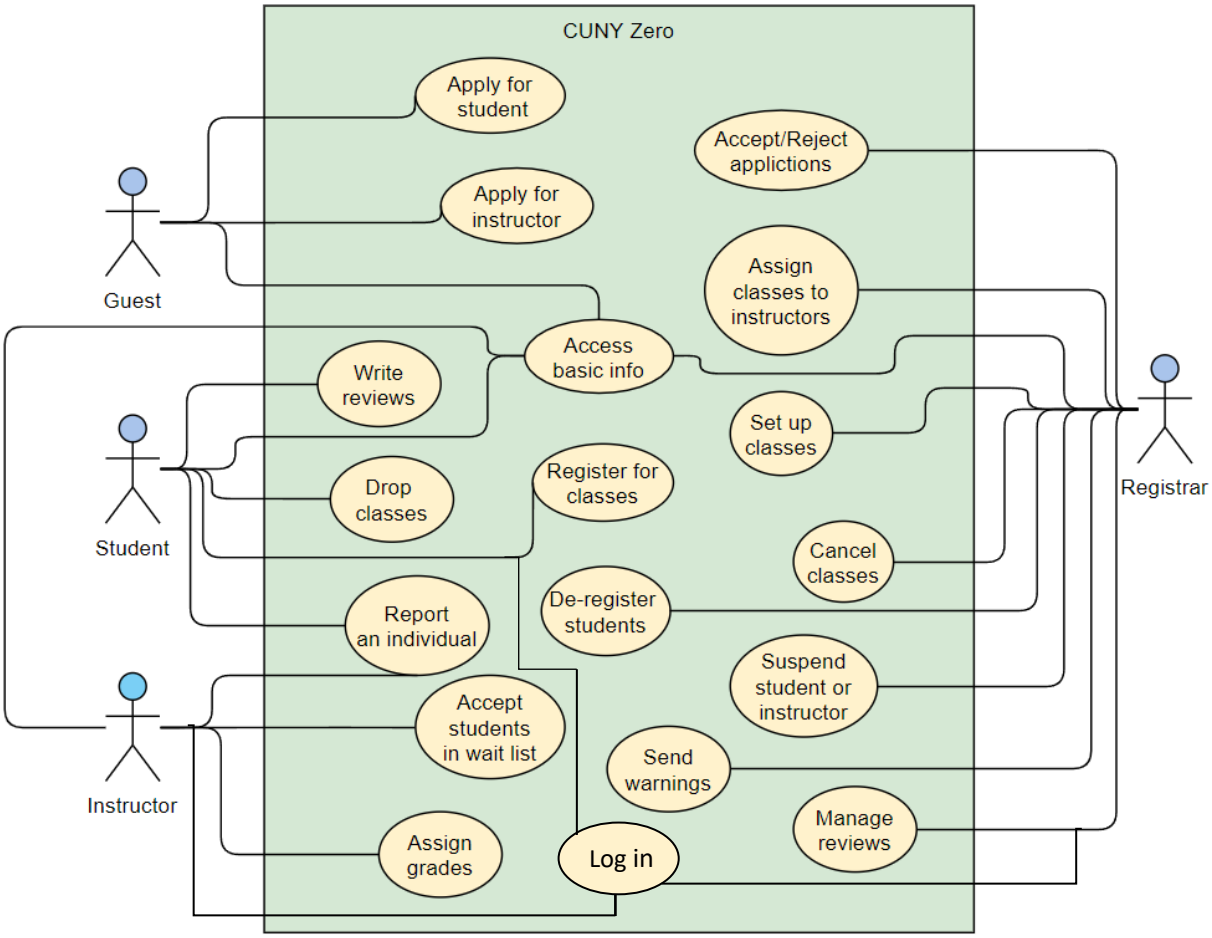
Section 3 explores deeper into the specific requirements of the system under development. Interfaces and features are listed and sufficiently detailed to enable knowledgeable developers/designers to craft a system that satisfies all requirements.

Section 4 of this document provides a guide for better accessing information presented in this software requirements specification.

2. Overall Description

This section provides a general picture of the CUNY Zero system and its use cases, assumptions, and dependencies.

2.1 Use-Case Model Survey



2.2 Assumptions and Dependencies

It is assumed that guests, students, instructors, and registrars have access to a stable connection to the internet in order to be able to make use of this software. It is also assumed that the aforementioned users have a certain degree of proficiency in using a computer, mouse, keyboard, and navigating through a web site.

3. Specific Requirements

This section documents the specific requirements of the CUNY Zero system. Subsection 3.1 describes the major functions of the software depicted by the use case model while subsection 3.2 explores deeper into the very small details of every major function.

3.1 Use-Case Reports

***Title*: Registrar** communicates with the Applicant (**Guest**)
Participating Actors: **Registrar** and **Guest**

Flow of Events:

- The **Registrar** accesses the software through their terminal and reviews the applicant's document.

Preconditions:

- The Guest, if accepted, will be given an account to access the system

Postconditions:

- The Guest is now either an instructor or a student and can access all the available features in the system.

Title: Registered **Student** register for a class
Participating Actors: **Student**, **Registrar**, **Instructor**

Flow of Events:

- **Registrar** set up classes
- **Student** will access the systems to register for available classes

Preconditions:

- Before the use case begins, the **Student** must be logged onto the system.

Postconditions:

- The **Student** can now register for classes.

Title: **Instructor** assign grades to students
Participating Actors: **Instructor**, **Student**, **Registrar**

Flow of Events:

- **Instructor** will access the system and will be directed to the "Instructor's Page"
- The **Instructor** will be able to see student's academic information.
- The **Instructor** will then assign grades to the students

Preconditions:

- The Instructor must log onto the system to see student's academic record

Postconditions:

- The Instructor can now see the student's information and can now assign grades.

***Title*:** Report System for **Instructor** and **Student**
Participating Actors: **Student**, **Instructor**, **Registrar**

Flow of Events:

- **Student** or **Instructor** will access the system using their account
- They will be directed to a page where they can report about a student or an instructor
- The **Registrar** will review the submitted complaint and decide on what to do

Preconditions:

- The **Instructor** and **Student** must be logged onto the system for the use case to begin.

Postcondition:

- Instructor or Student can now write a complain

***Title*:** Dropping Class(es)
Participating Actors: **Student** and **Registrar**

Flow of Events:

- Once the **Student** log in through the system, the system will go to the homepage that shows the academic record and the class schedule.
- The **Student** will be able to drop classes

Preconditions:

- **Student** must be logged onto the system to drop classes.

Postconditions:

- Student will be able to access their schedule and can now drop classes

***Title*:** Student Reviews
Participating Actors: **Student** and **Registrar**

Flow of Events:

- **Student** will access the system by logging in and will go to a page where they can write a review about the **Instructor.**

Preconditions:

- **Student** must be logged onto the system to write a review.

Postconditions:

- **Student** will be able to write a review and rate the instructor's performance and class.

3.2 [1.]Supplementary Requirements

***Title*:** **Registrar** communicates with the Applicant (**Guest**)
Participating Actors: **Registrar** and **Guest**

Flow of Events:

- The **Registrar** accesses the software through their terminal and performs the following actions:
    - Look into the Students' application and check if:
        - GPA > 3.0, then the applicant will be accepted
        - If GPA < 3.0, then the applicant will be rejected will be given a reason why they are rejected.
    - Look into Instructors' applications.

Preconditions:

I. The Guest, if accepted as a student, will receive a Student ID and a one-time password which must be changed after logging in.
II. The Guest, if accepted as an instructor, will be given an ID and password to access the system.

Postconditions:

I. The **Student** can now register for classes
II. The **Registrar** will assign classes along with the size and schedule of the classes to the **Instructor**.

Title: Registered **Student** register for a class
Participating Actors: **Student**, **Registrar**, **Instructor**

Flow of Events:

- **Registrar** set up classes along with the schedule, name of **Instructor** and the size of the class
- **Student** will access the systems and can search for these available classes and can register for 2 – 4 courses.

Preconditions:

I. **Student** must not have time conflict among classes
II. The class that the **Student** wants to take is not full
III. The **Registrar** cancels a class/course during class running period
IV. The course has less than five students or all classes are cancelled

Postconditions:

I. **Student** will be registered for the course
II. If the **Student** is waitlisted, the **Instructor** will be able to let the **Student** register for that class.
III. **Student** will be able to choose another course for cancelled class by the **Registrar**
IV. The **Instructor** will be given a warning from the **Registrar** if the class is less than five students and suspended if all classes are cancelled.

Title: **Instructor** assign grades to students
Participating Actors: **Instructor**, **Student**, **Registrar**

Flow of Events:

- **Instructor** will access the system and will be directed to the "Instructor's Page"
- The page will show all the basic information of the students including name, email, ID, and gender.
- The **Instructor** will look through the academic record of the **Student** registered for the class.
- The **Instructor** will then assign grades to the students
- Through the system, a **Student** who is finishing 8 classes can be able to apply for graduation. **Registrar** checks all requirements by looking at the student's academic record on the student's page.

Preconditions:

As a **Student**:

I. **Student** with GPA less than 2.0 or failed the same class twice
II. **Student** with GPA between 2 and 2.25
III. **Student** with a GPA greater than 3.75 in a semester or overall GPA greater than 3.5
IV. **Registrar** checks student's application for graduation.

As an **Instructor**:

I. The **Instructor** did not assign grades for ALL students
II. If the overall GPA of the class is greater than 3.5 or less than 2.5, the **Registrar** will question the **Instructor**.

Postconditions:

As a **Student**:

I. The **Student** will be terminated

II. The **Student** will be given a warning and will be interviewed by the **Registrar**
III. The **Student** will be registered as an Honor Roll Student.

As an **Instructor**:

I. The **Instructor** will be given a warning by the **Registrar**
II. If there is no adequate justification for the decision, the **Registrar** will warn the **Instructor** or terminate the **Instructor** right away.

***Title*:** Report System for **Instructor** and **Student**
Participating Actors: **Student**, **Instructor**, **Registrar**

Flow of Events:

- **Student** or **Instructor** can access the system using their account
- They will be directed to a page where they can report about a student or an instructor
- The **Registrar** will review the submitted complaint and decide on what to do

Postconditions:

As a **Student**:

I. **Student** writes a complaint about another student or instructor.

As an **Instructor**:

I. **Instructor** can also write a complain to the **Registrar** to de-register a student.

Precondition:

As a **Student**:

I. **Registrar** will give the **Student** three warning and will result to suspension for one semester and must pay a fine.

As an **Instructor**:

I. **Registrar** will give the **Instructor** a warning.

Title: Dropping Class(es)
Participating Actors: **Student** and **Registrar**

Flow of Events:

- Once the **Student** logs in through the system, the **student** will be directed to the homepage which shows the academic record and the class schedule.
- The **Student** will be able to drop classes as long as it is between registration period and grading period.

Preconditions:

I. **Student** must drop class(es) between registration period and grading period.

Postconditions:

I. **Registrar** will assign **Student** who dropped the class a W grade.
II. If the **Student** dropped all classes, **Registrar** would automatically suspend the **Student** for one semester.

***Title*:** Student Reviews
Participating Actors: **Student** and **Registrar**

Flow of Events:

- **Students** will go to a page where they can write a review about the **Instructor** and will be posted anonymously
- The **Registrar** as the admin will be the only one who can see who did the review.
- The **Registrar** can then review the student's writing.
- Once the revision is done the **Registrar** will decide if the review is acceptable or not.

Preconditions:

I. **Student** cannot write a review after grades are posted.
II. **Student** post more than or equal to two vulgar words.
III. The Rating of the class is less than 2
IV. The **Registrar** gives the **Instructor** 3 or more warning for the bad rating.

Postconditions:

I. **Registrar** will send a warning per review to the **Student** if found posting vulgar words. In some cases, if the post has a lot of vulgar words, the post will not be posted and the **Student** will be given two warnings.
II. The **Instructor** will also be given a warning if the class rating is less than 2
III. A suspension will be given if the **Instructor** received 3 or more warnings.

[1.]Note: Section 3.2 is a more detailed report about the use-case diagram. It has all the necessary requirements for the actors (Registrar, Instructor, and Student) to study and implement so that the use-case can be executed properly.

4. Supporting Information

4.1 Index