Software Requirements Specification

for

Online Self-Service Course Enrollment Tool (OSSCET)

Version 2.0 approved

Prepared by Jesse Neubauer

University of Arizona Global Campus

August 28th, 2023

Table of Contents

Table of Contents ii

Revision History ii

1. Introduction 1

1.1 Purpose 1

1.2 Document Conventions 1

1.3 Intended Audience and Reading Suggestions 1

1.4 Product Scope 1

1.5 References 1

2. Overall Description 2

2.1 Product Perspective 2

2.2 Product Functions 2

2.3 User Classes and Characteristics 2

2.4 Operating Environment 2

2.5 Design and Implementation Constraints 2

2.6 User Documentation 2

2.7 Assumptions and Dependencies 3

3. External Interface Requirements 3

3.1 User Interfaces 3

3.2 Hardware Interfaces 3

3.3 Software Interfaces 3

3.4 Communications Interfaces 3

4. System Features 4

4.1 System Feature 1 4

4.2 System Feature 2 (and so on) 4

5. Other Nonfunctional Requirements 4

5.1 Performance Requirements 4

5.2 Safety Requirements 5

5.3 Security Requirements 5

5.4 Software Quality Attributes 5

5.5 Business Rules 5

6. Other Requirements 5

Appendix A: Glossary 5

Appendix B: Analysis Models 5

Appendix C: To Be Determined List 6

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
|  |  |  |  |
|  |  |  |  |

# Introduction

## Purpose

This project will address the problem of students enrolling into online courses.

## Document Conventions

Text which is ~~stricken through~~ indicates a feature which is not implemented in this stage of the project, but may yet be developed for later use in a portfolio.

## Intended Audience and Reading Suggestions

The intended reading audience for this document is Dr. Charmelia Butler, the instructor of this course.

## Product Scope

* New user registration that includes account and profile creation.
* Each new user should have a unique ID associated with a password. The system should guard against two users using the same ID for registration.
* Profiles must include some key information about the applicant including name, phone, email, and any other information you may see necessary.
* Post registration, users can login to the system at any time using the ID and the password created during the registration process.
* Online courses run through three semesters per year (spring/summer/fall), and students can list the courses that will be offered during any semester, as not all courses will be offered in every semester.
* Each course should have a maximum number of enrollments that may be different depending on the course.
* If a user wants to enroll into a course and the course is full, the student can add themselves onto a waiting list.
* A user can also cancel the enrollment from any course that they are enrolled in, and the system should inform the first in the waiting list (if any) that they ~~can~~ enroll into the class.

## References

None

# Overall Description

## Product Perspective

This project is inspired by the course-enrollment tools I have used in the past, during the completion of my first Bachelor’s degree at a traditional college. If implemented at UAGC, a system like this would allow students far more flexibility and involvement in their own education.

## Product Functions

* New user registration that includes account and profile creation.
* Each new user should have a unique ID associated with a password. The system should guard against two users using the same ID for registration. The system also guards against two users having the same ID after editing their own profile.
* Profiles must include some key information about the applicant including name, phone, email, and any other information you may see necessary.
* Post registration, users can login to the system at any time using the ID and the password created during the registration process.
* Online courses run through three semesters per year (spring/summer/fall), and students can list the courses that will be offered during any semester, as not all courses will be offered in every semester.
* Each course should have a maximum number of enrollments that may be different depending on the course.
* If a user wants to enroll into a course and the course is full, the student can add themselves onto a waiting list.
* A user can also cancel the enrollment from any course that they are enrolled in, and the system should inform the first in the waiting list (if any) that they ~~can~~ enroll into the class.

## User Classes and Characteristics

Users will consist of three classes. Students, Teachers, and Administrators.

## Operating Environment

The software will operate on common desktop browsers, such as Chrome, Firefox, and Edge.

## Design and Implementation Constraints

Design and implementation was constrained to the PHP, MySQL, and Bootstrap libraries dictated by the course assignments, and is limited to what I was able to complete during the three-week development phase of the course’s duration.

## User Documentation

Users must document or otherwise recall their own login information, as there is no password reset functionality at this time.

## Assumptions and Dependencies

None

# External Interface Requirements

## User Interfaces

The user interface consists of links, buttons, forms, scrollbars, and tables. At this time, its visual appeal is minimal and utilitarian.

## Hardware Interfaces

At this time, all communications are performed via web browser, mouse, and keyboard.

## Software Interfaces

The project includes its own database, and does not interact with any other systems at this time. It could, hypothetically, integrate with a tuition/salary payment system. It could also integrate with a grades-tracking system, although there is a framework in place for implementing such a feature in this project in the future.

## Communications Interfaces

At this time, all communications are performed via web browser, mouse, and keyboard. There is an internal messaging system built into the project which allows users to receive messages from the system ~~and from other users~~.

# System Features

To differentiate: a course is a set curriculum that may be repeated multiple times, and could be taught by different teachers in the department. There may even be multiple instances of the same course taught simultaneously. A ‘class,’ also sometimes referred to as an offering, is a single instance of a course, taught for the duration of one semester.

## User login/registration/authentication

4.1.1 Description and Priority:

New user registration that includes account and profile creation. This is of utmost priority.

4.1.2 Stimulus/Response Sequences

Users must select between logging in or creating a new profile. New users will be prompted to select their role: Student or Teacher

If an unregistered username is used for logging in, the system will say “Invalid credentials.” If an incorrect password is provided, the system will *also* say “Invalid credentials.”

If a registered username is used for creating a new profile, the system will say “Username unavailable.”

If and only if a valid username and password is provided, the system will grant the user access to their home page and account.

4.1.3 Functional Requirements

REQ-1: Each new user should have a unique username associated with a password. The system should guard against two users using the same username for registration.

REQ-2: The system will not allow access to any user page without authentication.

## User account profile information

4.2.1 Description and Priority:

User account information and personal settings. This is of secondary priority.

4.2.2 Stimulus/Response Sequences

Profiles must include some key information about the applicant including name, phone, email, mailing address, tuition/salary, and date of enrollment/hiring.

Users may choose to update these fields as applicable. (The last two fields are not able to be updated by non-Admin users.)

4.2.3 Functional Requirements

REQ-3: The user’s information will be collected during the account creation process.

REQ-4: The system will display the information under the user profile section once logged in.

REQ-5: The user may change this information at any time.

## Class creation and enrollment

4.3.1 Description and Priority:

Course offerings created and listed as classes. This is a primary priority.

4.3.2 Stimulus/Response Sequences

Online courses run through three semesters per year (spring/summer/fall), and Teachers (non-Admin staff) may create a course and/or list a class offering in any/all of the ~~three upcoming~~ semesters.

~~If a teacher attempts to create an offering that overlaps with the timing of another of their offerings, the system will display “Time conflict: [previous course name]”~~.

Students can view a list of the courses that will be offered during any semester, as not all courses will be offered in every semester.

~~If a student attempts to enroll in a class that overlaps with the timing of another of their classes, the system will display “Time conflict: [previous course name]”.~~

4.3.3 Functional Requirements

REQ-6: There is a view of all classes offered ~~in a given semester~~.

REQ-7: Users have access to see this view ~~of the current semester~~. Users can *interact* with the view ~~of the following three semesters~~.

## Course capacity

4.4.1 Description and Priority:

Each course offering includes a maximum number of enrolled students. This is a secondary priority.

4.4.2 Stimulus/Response Sequences

Each course should have a maximum number of enrollments, and this number may be different depending on the course. When offered, the class begins with this value as the number of vacant seats. (If offered in person, this value will be based on the number of seats in the room where the class takes place, if smaller than the default course occupancy.)

When a student enrolls in a class, the number of vacancies for that class decreases by one until it reaches 0 and the class is considered full.

If a student attempts to enroll into a class and the class is full, the student is given the option to add themselves onto a waiting list.

~~If there is a non-full offering of the same course in the same semester, and it does not conflict with the student’s schedule, the system will notify the student and provide a link to this other class.~~

4.4.3 Functional Requirements

REQ-8: Each class has a vacancy/occupancy attribute which is updated each time a student enrolls in the class.

REQ-9: A list of first-in-first-out students is created for classes which reach a vacancy level of 0.

## Cancelling enrollments

4.5.1 Description and Priority:

A student can withdraw from a class they’ve enrolled in, thus freeing up class capacity. This is a tertiary priority.

4.5.2 Stimulus/Response Sequences

A student may cancel their enrollment from any class that they are enrolled in.

The system should inform the first student in the waiting list (if any) that they have now been enrolled into the class.

4.5.3 Functional Requirements

REQ-10: A student needs to be able to view a list of the classes they’ve enrolled in. Each one must have a “withdraw” option available. ~~Teachers and administrators may also withdraw a student from a class, though this will be highly audited and regulated.~~

REQ-11: Withdrawing from the class increases the vacancy number of that class. If the vacancy goes from 0 to 1, the waiting list protocol is engaged.

REQ-12: The student at the top of the waiting list receives a notification within their messages tab.

REQ-13: The class will still appear as full to students not on the waiting list, and those students may join the bottom of the waiting list. The student at the top of the waiting list will ~~gain access to enroll in the class~~ be automatically enrolled in the class.

~~REQ-14:~~ ~~If the first-notified student declines the enrollment, they are removed from the waiting list, and the next student on the list (if any) is notified.~~

~~REQ-15:~~ ~~If the first-notified student does not respond to the notification in a timely manner, the next student on the list is notified, and so on. Both/all notified students now have access to enroll in the class until such time as one of them accepts the enrollment. At this time, all other offers will be rescinded, and the other notified students will return to their original spot on the waiting list.~~

# Other Nonfunctional Requirements

## Performance Requirements

Each page must take no longer than five seconds to load completely.

## Safety Requirements

There are no safety requirements, as the data contained within the database is fictional and does not include any PII.

## Security Requirements

There is a password verification system, but since the only user of this software will be Dr. Butler, and since it was not a requirement of this course, no additional security measures or authentication practices have been implemented.

## Software Quality Attributes

Correctness and operability are the attributes of this program with the highest priority.

## Business Rules

A teacher may only cancel a class if there are no students enrolled in it. A student may enroll in any class, but if the class offering has no vacancies, the student will be placed at the bottom of a waiting list. When a vacancy opens up in a full class, the student at the top of the waiting list will be automatically enrolled, and notified of this development.

# Other Requirements

Appendix A: Glossary

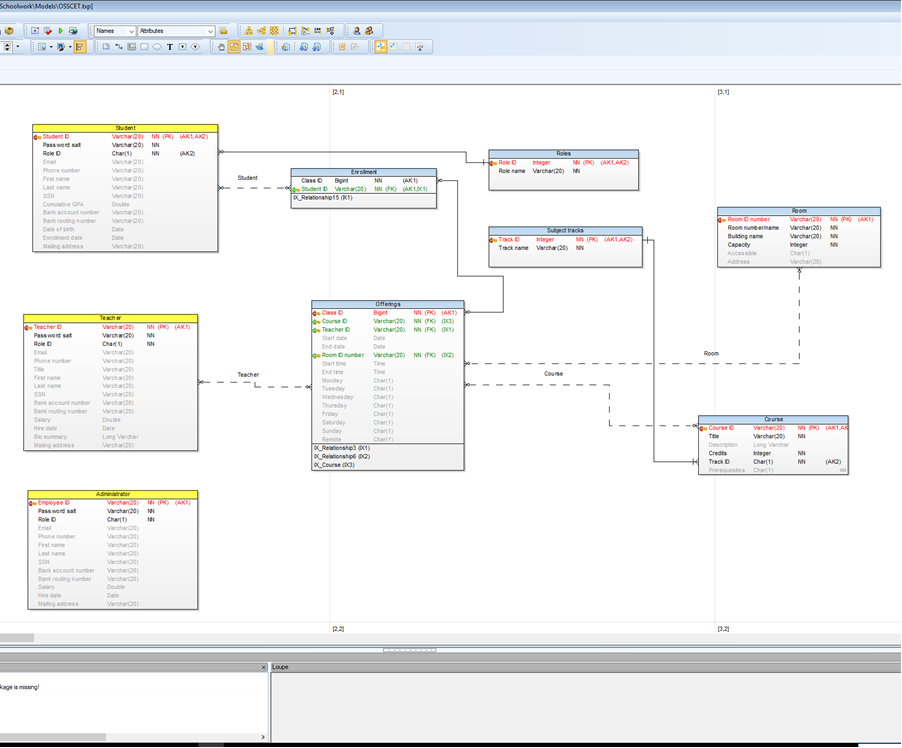
Administrator: A user of the system who is neither a student nor a teacher. These may include advisors, deans, heads of departments, customer service representatives, technicians, automated systems, and more.

Student: A user of the system who is learning. They pay tuition to the school, enroll in and attend classes, and receive grades which may contribute to a degree.

Teacher: A user of the system who is mentoring. They earn a salary from the school, create and/or offer classes, teach any classes they’ve offered, and provide grades and feedback to students.

Course: The format of a defined syllabus that can be taught If we were to draw parallels between the terms used in this project and the terms used in programming, a Course would be equivalent to what would be called a “Class” in Java.

Class/Offering: The scheduled time and place during which a specific course will be taught. Using the same parallel as above, a Class would be equivalent to an instantiated object in Java.

Note: the ”Enrollment” entity should also include attributes: Date enrolled (in the class, which is different from the date the student enrolled in the school), status (active, wait-list, or withdrawn), and grade (represented as a percentage).