**Hilltop Online**

**Software Requirement Specification**

prepared by Jason Morris, Carlos Alpuche, Adrian Leonard

Christopher Inzana, and Stephanie Thrash

COSC 3339

Version 1.0

Thursday, February 16th 2017

# Table of Contents

[**Table of Contents**](#_f6kpzuxpkd42) **2**

[**Revision History**](#_huc10se78nji) **4**

[**References**](#_5ytwdddiedx8) **4**

[**Introduction**](#_t5d1nxhlu6w) **5**

[Purpose](#_yfi820ikefqi) 5

[Intended Audience and Stakeholders](#_uioy6f19y8i) 5

[Product Scope](#_z80i8w8m6tcp) 6

[**User Requirements Documentation**](#_p6v7nifgv6dc) **7**

[Actor Identification](#_o5muk3z2p5zi) 7

[Students](#_o16l3x4siuqc) 7

[Instructors](#_hd8gakq0uk2o) 7

[Registrar](#_kt91msjgunbp) 7

[Use Case Diagram](#_s4h80he4biuu) 8

[Use Cases](#_am4asmfb1534) 9

[View Section History](#_dzez9g7q58e2) 9

[Drop Student](#_wfcbxeh1zmut) 9

[Section Details](#_iu0xthu6hp9s) 9

[View Assignments](#_5o7sjm2a2ecd) 10

[Add Assignments](#_kobn9bljs2qv) 10

[Edit Assignments](#_wa3n0moeydu3) 11

[Remove Assignment](#_rogb9c5ksp87) 11

[View/Edit Assignment Grading Table](#_g352kvmj537k) 12

[Submit Final Grades](#_utxbzsi1nqbc) 12

[Profile Page](#_88dsd3j139ka) 13

[View Course Catalog](#_qifdcwkgflva) 13

[Login](#_fu6vj31jryz0) 14

[Logout](#_3qj5xj316onq) 14

[View Current Schedule](#_6de0ekubm1tc) 14

[Add Section to Schedule](#_rniy0ntqpqgf) 15

[Drop Section from Schedule](#_feqw4qf9xopg) 15

[View Grades](#_3tx5h51zgnx1) 16

[View Transcript](#_xs2r5yvznbbq) 16

[Add Terms](#_eqw3hsbmg1x6) 16

[Edit Terms](#_vpsk28nw60vj) 17

[Remove Terms](#_e0dgvd9kjcfj) 17

[Add Course Catalog Entries](#_r5oay2mktmye) 18

[Edit Course Catalog Entries](#_yeior0hewkuk) 18

[Remove Course Catalog Entries](#_u2wjkd9wi1e6) 18

[Add Sections to Terms](#_h7tf8x9e7tb2) 19

[Edit Sections in Terms](#_7d5fdvy39x1i) 19

[Remove Sections from Terms](#_9voea61eh67d) 20

[Add User](#_1zjla4mevcss) 20

[Edit User](#_9bl7wzvinh1) 21

[Remove User](#_2cs96bv3mzxc) 21

[View User Info](#_vhrv70q2r1oe) 21

[Change Password](#_cbfmgsotbf1g) 22

[**System Architecture**](#_jqgg5inhhzty) **23**

[System Modules](#_jx1gkwhom19n) 23

[System Module Description](#_hrk5shk9eety) 23

[**System Requirement Specification**](#_iwmqoiya3kdm) **25**

[Sequence Diagrams](#_797elmadvyqd) 25

[View Section History](#_my2kxqclpad4) 25

[Drop Student](#_my2kxqclpad4) 25

[Section Details](#_my2kxqclpad4) 26

[View Assignments](#_r6bgvqkto3lh) 26

[Add Assignments](#_sek6v89vrax7) 27

[Edit Assignments](#_3duanhunvc5k) 27

[Remove Assignment](#_bbtpf73wb3lz) 28

[View/Edit Assignment Grading Table](#_bwc3ifh69dmy) 29

[Submit Final Grades](#_c9d4389fwad) 30

[Profile Page](#_my2kxqclpad4) 31

[View Course Catalog](#_my2kxqclpad4) 31

[Login](#_79o4di74v2j7) 31

[Logout](#_my2kxqclpad4) 32

[View Current Schedule](#_my2kxqclpad4) 33

[Add Section to Schedule](#_my2kxqclpad4) 33

[Drop Section from Schedule](#_my2kxqclpad4) 34

[View Current Grades](#_my2kxqclpad4) 34

[View Transcript](#_my2kxqclpad4) 35

[Add terms](#_my2kxqclpad4) 35

[Edit terms](#_my2kxqclpad4) 36

[Remove terms](#_my2kxqclpad4) 36

[Add Course Catalog Entries](#_my2kxqclpad4) 37

[Edit Course Catalog Entries](#_my2kxqclpad4) 37

[Remove Course Catalog Entries](#_my2kxqclpad4) 38

[Add Sections to Terms](#_my2kxqclpad4) 38

[Edit Sections in Terms](#_my2kxqclpad4) 39

[Remove Sections from Terms](#_my2kxqclpad4) 39

[Add User](#_my2kxqclpad4) 40

[Edit User](#_my2kxqclpad4) 40

[Remove User](#_my2kxqclpad4) 41

[View User Info](#_ub3365tfsszf) 41

[Change Password](#_my2kxqclpad4) 42

[Nonfunctional Requirements](#_tsba2ru9lkfl) 42

[**Glossary**](#_pj8lbvdni4id) **43**

[**Appendix**](#_5nftksud56c6) **43**

[Appendix A: Grade Calculations](#_u8oyv4hudl83) 43

# Revision History

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

# References

[*https://web.cs.dal.ca/~hawkey/3130/srs\_template-ieee.doc*](https://web.cs.dal.ca/~hawkey/3130/srs_template-ieee.doc)

*Software Engineering, 10th Edition, Ian Sommerville*

# Introduction

### Purpose

The requirements described in this document are for a standalone university grading and registration system version 1.0. This system will give registrars, instructors, and students the ability to access needed information regarding grades, courses, and sections. Registrars are the administrators of the system and are responsible for managing users, updating course information, and creating sections. Instructors are capable of viewing information regarding their own sections, assigning grades, and managing students within their sections. Lastly, students are allowed viewable access to current and past grades and are capable of performing enrollment actions such as adding or dropping sections. This system is intended for use by schools to help streamline and unify the enrollment and grading processes.

### Intended Audience and Stakeholders

This document is intended for use by all stakeholders in our system, including: Accreditation Bodies, University Curriculum Reviewers, University Student Information Regulators, University Budgeting Committees, University Registrars, University Instructors, University Students, Software Developers, Project Managers, Software Testers, and Operations Teams including University IT Staff. The following SRS includes instructions on software implementation, business and operations rules, overall system design and function and non functional requirements. The main sections should be read sequentially as each builds off the other. Other parts of the document, including the Appendices and Glossary, are meant to clarify terms and should be read first by any non-technical readers. Other stakeholders, like regulators, may find more value in the implementation details of the system architecture.

### Product Scope

Our software helps universities have a unified grading and registration system system. It will help students register for classes, keep track of their schedules, see their grades, and get a transcript. It will help instructors keep grading separate for each section they teach, submit final grades, view which students they have in each section and allow them to drop a student from a section. The registrar will have the ability to keep the course catalog up to date, create new terms and available sections, and create user accounts for students and instructors. Our goal is to help all parties save time in the registration system by developing a minimal interface and straightforward actions.

# User Requirements Documentation

### Actor Identification

#### Students

Students can access the system by logging in using their email and an auto-generated password. Students can access the enrollment module, as well as the grading module to view their grades. In the enrollment module, Students can add/drop sections of a course. Students can view the course catalog. Students can view their own profile, their transcript, and their current schedule.

#### Instructors

Instructors can access the system by logging in using their email and an auto-generated password. Instructors depend on the system to view their schedules and perform grading for their sections.

#### Registrar

Registrars manage the Users of the system, available terms and sections for registration, and course entries in the Course Catalog.

### Use Case DiagramUse Case Model.png

### Use Cases

#### View Section History

|  |  |
| --- | --- |
| **Actors** | Instructor |
| **Description** | Displays tables of sections taught by instructor, separated by term and year. |
| **Data** | InstructorID, authorization key |
| **Response** | A list of tables ordered chronologically containing details of each section taught by instructor |
| **Stimulus** | Actor clicked *View Section History* link on profile page |
| **Comments** |  |

#### Drop Student

|  |  |
| --- | --- |
| **Actors** | Instructor |
| **Description** | Allows Instructor to drop the selected student from a section’s enrollment |
| **Data** | InstructorID, SectionID, StudentID, authorization key |
| **Response** | A confirmation message informing instructor of successful drop |
| **Stimulus** | Actor clicked *Drop Student* link next to student’s name in section roster |
| **Comments** |  |

#### Section Details

|  |  |
| --- | --- |
| **Actors** | Instructor |
| **Description** | Provides instructor with a view of the details of a section they are teaching. These details include: day, time, location, and number students enrolled. |
| **Data** | InstructorID, sectionID, authorization key |
| **Response** | Displays a header containing the section day, time, location and number. Beneath the header is a table containing the names of students enrolled in the section with a button to drop each student |
| **Stimulus** | Instructor clicked the *Section Details* button. |
| **Comments** |  |

#### View Assignments

|  |  |
| --- | --- |
| **Actors** | Instructor |
| **Description** | Allows Instructors to view a list of all assignments separated into tables within their assignment groups. |
| **Data** | InstructorID, sectionID, authorization key |
| **Response** | Displays a table of assignment groups and their associated assignments. |
| **Stimulus** | Actor clicks *View Assignments* link on window for the currently open section tab |
| **Comments** |  |

#### Add Assignments

|  |  |
| --- | --- |
| **Actors** | Instructor |
| **Description** | Allows Instructor to add a new assignment to an existing assignment group or create a new assignment group. Each assignment or assignment group contains a name and a weight percentage. The total weight in each assignment group should total 100, and the total weight among assignment groups should also total 100. |
| **Data** | InstructorID, SectionID, authorization key |
| **Response** | Adds the new assignment to an assignment group or creates a new assignment group. |
| **Stimulus** | Instructor is on the *ViewAssignments* page and clicked the plus sign next to an empty assignment group table or inside of an assignment group table. |
| **Comments** | When assignments are created for a section, they are stored for each student in that section with a default grade of null. This allows student grades to be viewed for completed assignments only without a score for incomplete assignments that could impact the total section grade calculation. See *View/Edit Assignment Grading Table* for more |

#### Edit Assignments

|  |  |
| --- | --- |
| **Actors** | Instructor |
| **Description** | Allows the instructor to change the name or weights of an assignment or assignment group. |
| **Data** | instructorID, sectionID, assignmentGroupID OR assignmentID, authorization key |
| **Response** | Displays a table containing the list of assignment groups, assignments within the group and all respective weights. |
| **Stimulus** | Instructor is on the *ViewAssignments* page and hovered over an existing assignment group name, assignment name, assignment group weight, or assignment weight. After editing, the update action takes place on the press of the *enter* key. |
| **Comments** | This is a single use case for assignment groups and assignments because they are dependent on each other. An assignment must belong to an assignment group. |

#### Remove Assignment

|  |  |
| --- | --- |
| **Actors** | Instructor |
| **Description** | Allows the Instructor to remove an assignment from an assignment group or remove an entire assignment group and all of it’s assignments. |
| **Data** | instructorID, sectionID, assignmentGroupID OR assignmentID, authorization key |
| **Response** | The assignment or assignment group disappears from the *ViewAssignments* table. |
| **Stimulus** | The use clicks the minus sign next to an assignment group or an assignment. |
| **Comments** | This is a single use case for assignment groups and assignments because they are dependent on each other. An assignment must belong to an assignment group. |

#### View/Edit Assignment Grading Table

|  |  |
| --- | --- |
| **Actors** | Instructor |
| **Description** | An instructor inputs grades in a table with each row representing a student and each column representing an assignment. The grades are in a numerical format.  The last column in every row represents a cumulative grade in plus/minus format for all assignments weighted using the assignment groups in use case *View/Edit/Add assignments*. |
| **Data** | An Instructor id, section id, and authorization key. |
| **Response** | An interactive grading table that allows changes of individual assignments and validation prompts when those grades are successfully updated. |
| **Stimulus** | Actor clicked Assignment Table link. |
| **Comments** | The table is an interactive Nx(M+2) grid, where N is the number of rows representing students and M is the number of assignments. The extra 2 columns are the first and last column to be populated with student grade and cumulative section grade for every student. Each cell for assignments can be changed to represent a grade change. All grade cells are 0 by default. |

#### Submit Final Grades

|  |  |
| --- | --- |
| **Actors** | Instructor |
| **Description** | A three column table with student names and cumulative grades in both numerical and letter formats. The numerical grade is calculated from grades in the Assignment Grading Table.  The numerical grade in the middle column can be changed, which updates the letter grade in last column. When finished verifying grades, the instructor uses a submit button to store the final grades in the database for use by students and the registrar. |
| **Data** | An Instructor username, section id, and authorization key. |
| **Response** | An interactive final grades table that allows changes of cumulative numerical grades and a validation prompt when those grades are successfully stored. |
| **Stimulus** | Instructor clicks on “Submit Final Grade” button. |
| **Comments** | Unlike assignments, who have a default value of null, final term grades have no default value, and are only stored in the database after an instructor submits them. |

#### Profile Page

|  |  |
| --- | --- |
| **Actors** | Student, Instructor |
| **Description** | Displays list of editable personal details about an instructor or student. Some fields, like userID, are immutable. |
| **Data** | A userID and authorization key. |
| **Response** | Student or Instructor details are displayed for the given username. |
| **Stimulus** | Actor is redirected to this page after login or actor clicked *View Profile* link. |
| **Comments** |  |

#### View Course Catalog

|  |  |
| --- | --- |
| **Actors** | Student, Instructor, Registrar |
| **Description** | A searchable interface for all courses available at the university. Each course shall contain a summary of the course’s objectives, the normal terms in which the course is offered, and any pre or co-requisites that the course requires. |
| **Data** | In the search field, enter a course name or unique course id. |
| **Response** | If found, return course information. If not found, display error. |
| **Stimulus** | Actor clicked on “View Course Catalog” link. Available search field to search for specific courses. |
| **Comments** | This use case is a two-step implementation. First, implement a searchable interface over the available courses. Then, implement a course information display. |

#### Login

|  |  |
| --- | --- |
| **Actors** | Student, Instructor, Registrar |
| **Description** | The entrypoint into all pages in the application. An Actor must input their username and password to gain access to all pages. |
| **Data** | Username, Password |
| **Response** | If a valid combination, an authentication key is stored by the client in a cookie. If not a valid combination, shows a login an error message. |
| **Stimulus** | Actor clicked on login button on the login page of the system. This is a pre-req for any other interaction with the system |
| **Comments** | Authentication key can be stored multiple ways. See [Glossary](#_pj8lbvdni4id) |

#### Logout

|  |  |
| --- | --- |
| **Actors** | Instructor, Student, Registrar |
| **Description** | The Actor currently logged in can log out of the system. |
| **Data** | NA |
| **Response** | The cookie containing user information is destroyed and the Actor is redirected to the login page. |
| **Stimulus** | An Actor presses the *Logout* button. |
| **Comments** |  |

#### View Current Schedule

|  |  |
| --- | --- |
| **Actors** | Instructor, Student |
| **Description** | A Graphical, single-week calendar is showed for a 7 day week in which students and instructors may or may not have classes. Classes show up as blocks on days associated with their meeting times. |
| **Data** | A user id associated with the logged in student or instructor.. |
| **Response** | A calendar with blocked time frames for classes associated with the user’s taught or registered classes. |
| **Stimulus** | Actor clicked on a link/tab that will display their current schedule for current term |
| **Comments** | See calendar on myEdu.com for example graphical format. |

#### Add Section to Schedule

|  |  |
| --- | --- |
| **Actors** | Student |
| **Description** | A student may add sections from their list of current enrollments. |
| **Data** | StudentID, sectionID |
| **Response** | The section is added to the student’s enrollments. |
| **Stimulus** | Actor clicks on *Add Section* button next to a section entry in the list of sections offered in the current term. |
| **Comments** |  |

#### 

#### Drop Section from Schedule

|  |  |
| --- | --- |
| **Actors** | Registrar |
| **Description** | A student may drop sections of offered courses to their current enrolled sections. |
| **Data** | StudentID, sectionID |
| **Response** | The section is dropped from the student’s enrollments. |
| **Stimulus** | Actor clicks on *Drop Section* button next to a section entry in their *View Current Schedule* view |
| **Comments** | When dropping a student, completely delete the enrollment record. The drop should not be recorded on the transcript. |

#### View Grades

|  |  |
| --- | --- |
| **Actors** | Student |
| **Description** | A Student will be able to access a list of all grades in an enrolled section. |
| **Data** | A studentID, sectionID, authorization key |
| **Response** | A table of assignments grades, weighted assignment group totals, and a operall section grade is shown for each assignment in which the student is currently enrolled. |
| **Stimulus** | A Student clicks on the *View Grades* button while on the *View Schedule* screen. |
| **Comments** | Each assignment grade is to be shown as a raw score and as a weighted score. Total grade calculations can be found in the Appendix. |

#### View Transcript

|  |  |
| --- | --- |
| **Actors** | Student |
| **Description** | Students can view a history of enrollments and grades listed by sections and terms. |
| **Data** | StudentID, authorization key |
| **Response** | A table separated by semesters and years each having the specific classes that were in that semester with a final grade |
| **Stimulus** | Student clicks on *View Transcript* |
| **Comments** | Grades on the transcript are listed by section by term and only include the final grade for the section. |

#### Add Terms

|  |  |
| --- | --- |
| **Actors** | Registrar |
| **Description** | The Registrar can add a new term to the system. A term is the combination of a semester, year, and active boolean. |
| **Data** | RegistrarID, termSemester, termYear, activeBoolean, authorization key |
| **Response** | The new term is added to the database and is shown on other menu items in the system, such as *Add Sections*. |
| **Stimulus** | Registrar clicks on the *Add Term* button. |
| **Comments** | A term is active if sections in the term are still holding class periods meetings. |

#### Edit Terms

|  |  |
| --- | --- |
| **Actors** | Registrar |
| **Description** | The Registrar can edit the active boolean, semester, or year attributes of a term. |
| **Data** | RegistrarID, termID, authorization key, optional(activeBoolean, semester, year) |
| **Response** | The new term information is displayed. |
| **Stimulus** | Registrar clicks the *Edit Term* button. |
| **Comments** | If the section is closed, it should show up in other menu items as *Semester Year (Closed)* - ie Fall 2016 (Closed). |

#### Remove Terms

|  |  |
| --- | --- |
| **Actors** | Registrar |
| **Description** | The Registrar can remove a term from the system. |
| **Data** | RegistrarID, termID, authorization key |
| **Response** | The term is no longer listed anywhere in the system. |
| **Stimulus** | Registrar clicks on the *Remove Term* button. |
| **Comments** | When a term is removed, all data associated with it should also be removed, such as offered Sections. |

#### Add Course Catalog Entries

|  |  |
| --- | --- |
| **Actors** | Registrar |
| **Description** | The Registrar can add Courses to the Course Catalog. The main attributes of a course are course number, name, department, description, and credit hours. |
| **Data** | registrarID, authorization key, courseNumber, courseName, courseDepartment, courseDescription, and courseHours. |
| **Response** | A new course record is created in the database and includes all course attributes. The course will appear in the course catalog. |
| **Stimulus** | Actor clicks on *Add Course Catalog Entry* link |
| **Comments** | There may need to be an enforced length of course description depending on size of database fields. |

#### Edit Course Catalog Entries

|  |  |
| --- | --- |
| **Actors** | Registrar |
| **Description** | The Registrar can edit course entries in the Course Catalog. Only course name, department, description, and credit hours may be changed. |
| **Data** | registrarID, authorization key, courseID, optional(courseName, courseDepartment, courseDescription, courseHours) |
| **Response** | The changes are stored in the database course record courseID and he course changes will appear in the course catalog. |
| **Stimulus** | Actor clicks on *Edit* button next to a course in the Course Catalog. |
| **Comments** |  |

#### Remove Course Catalog Entries

|  |  |
| --- | --- |
| **Actors** | Registrar |
| **Description** | The Registrar can remove coarse entries from the course catalog. |
| **Data** | registrarID, authorization key, CourseId |
| **Response** | The course record in the database has a false value on it’s available boolean attribute. The course should not appear in the course catalog. |
| **Stimulus** | Actor clicks on *Remove* button next to a course in the Course Catalog and completes its confirmation dialog. |
| **Comments** | If a course was completely dropped from the table, then sections would need to be deleted or nullified. To remediate this issue, removing a course should set an availability boolean on that record to false, that way it does not appear in the course catalog but does not break section functionality. |

#### Add Sections to Terms

|  |  |
| --- | --- |
| **Actors** | Registrar |
| **Description** | The registrar can add sections to a particular term. Each section has an instructor, meeting time, and is an instance of a course from the course catalog. |
| **Data** | registrarID, authorization key, courseID, InstructorID, meetingTime |
| **Response** | The section created will be added to the database and is viewable by students, instructors, and reigstrars. |
| **Stimulus** | Actor clicked on *Add Section* link. |
| **Comments** | Attributes of a section that are most commonly seen, such as title and course number, are inherited from their course parent and are not specified by the registrar at the time of section creation. The only thing required. |

#### Edit Sections in Terms

|  |  |
| --- | --- |
| **Actors** | Registrar |
| **Description** | The registrar can edit attributes of a section. |
| **Data** | registrarID, authorization key, sectionID, optional(courseID, InstructorID, meetingTime) |
| **Response** | The section created will be edited in the database and it’s changes will be viewable by students, instructors, and reigstrars. |
| **Stimulus** | Actor clicked on *Edit Section* link. |
| **Comments** | When this action is invoked, none or all of the fields from the Section object may be specified. Only update the fields specified. |

#### Remove Sections from Terms

|  |  |
| --- | --- |
| **Actors** | Registrar |
| **Description** | The registrar can remove sections from a particular term. |
| **Data** | registrarID, authorization key, sectionID |
| **Response** | Sections are no longer viewable |
| **Stimulus** | Actor clicked on *Remove Sections* link. |
| **Comments** | When the section is removed, it’s available boolean field is set to false. It should not appear in the available sections list from a term. If the section record was removed from the database entirely, the students in that section would lose that record on their transcript. |

#### Add User

|  |  |
| --- | --- |
| **Actors** | Registrar |
| **Description** | Registrar can add a User to the system. They must specify attributes related to that user, such as role, date of birth, name, address, and telephone. |
| **Data** | registrarID, authorizationKey, userName, userAddress, userDoB, userTelephone, userRole |
| **Response** | User added to the database. Respond with userEmail and userPassword |
| **Stimulus** | Registrar clicks on *Add Use*r link. |
| **Comments** |  |

#### Edit User

|  |  |
| --- | --- |
| **Actors** | Registrar |
| **Description** | Registrar can edit a User to the system. Editable attributes include date of birth, name, address, and telephone. |
| **Data** | registrarID, authorizationKey, userID, optional(userName, userAddress, userDoB, userTelephone) |
| **Response** | User added to the database. Respond with userEmail and userPassword |
| **Stimulus** | Registrar clicks on *Edit Use*r link. |
| **Comments** | This method should not allow the registrar to change the role of a student. |

#### Remove User

|  |  |
| --- | --- |
| **Actors** | Registrar |
| **Description** | The Registrar can remove a User from the system. |
| **Data** | registrarID, authorization key, userID |
| **Response** | The user is removed from the database and the Registrar recieves a confirmation dialog that the user was removed. |
| **Stimulus** | Actor clicked on *Remove User* link |
| **Comments** | Removing a user should not cascade to other portions of the database. For example, remove a Student should not remove their transcript. |

#### View User Info

|  |  |
| --- | --- |
| **Actors** | Registrar |
| **Description** | A registrar can view a User’s information. This includes a search function with autofill by user id or name. |
| **Data** | registrarID, authorization key, userID |
| **Response** | A page showing User’s information including User ID, User Name, DoB, Address, Telephone Number, and Role. |
| **Stimulus** | Actor clicked on *View User Info* link. |
| **Comments** | For design considerations, see the St. Edward’s Directory search page. |

# 

#### Change Password

|  |  |
| --- | --- |
| **Actors** | Instructor, Student, Registrar |
| **Description** | All actors should be able to change their passwords. |
| **Data** | UserID, OldPassword, NewPassword |
| **Response** | Password should be changed in the database and the user should be prompted to re-login. |
| **Stimulus** | User clicks the Change Password button. |
| **Comments** | User must re-login after a password change. Please refer to Logout use case to see how to log a user out. |

# System Architecture

### System Modules

System Modules.png

### System Module Description

|  |  |
| --- | --- |
| **System Module** | **Description** |
| User Management | Prompts for and processes username and password entered by user. After successful login, user sent to a specific module based on their role. No access given and error displayed on failed login. This module also tracks user roles. |
| Enrollment | These functions include: adding terms by registrar, creating/editing sections, adding/dropping sections for students, drop student option for instructors and viewing sections currently taught or enrolled. |
| Grading | Contains all functions associated with student grades. These include: creating/editing assignments and their weights, inputting/updating assignment grades, submitting final grades and viewing assignment and section grades/transcripts. |
| Course Catalog | Provides a detailed view of all courses offered by the university and their descriptions. Includes functionality for registrar to add/edit/delete course information. |
| Student Profile | Includes display of student’s personal information, ability to view student’s weekly schedule, and contains links to other modules. |
| Instructor Profile | Includes display of instructor’s personal information, ability to view instructor’s weekly schedule, and contains links to other modules/functions. |

# 

# System Requirement Specification

### Sequence Diagrams

#### View Section HistoryViewSectionHistory.png

#### Drop StudentDrop Student.png

#### Section DetailsSection Details.png

#### View AssignmentsViewAssignments.png

#### Add Assignments AddAssignment.png

#### Edit Assignments

EditAssignment.png

#### Remove Assignment

RemoveAssignment.png

#### View/Edit Assignment Grading Table

View/Edit GradingTable.png

#### Submit Final Grades

SubmitFinalGrades.png

#### Profile PageProfile Page.png

#### View Course CatalogView Course Catalog.png

#### LoginLoginSequenceDiagram.png

#### Logout

#### Logout.png

#### View Current ScheduleView Current Schedule.png

#### Add Section to ScheduleAdd Section to Schedule.png

#### Drop Section from ScheduleDrop Section from Schedule.png

#### View Current GradesView Grades.png

#### View TranscriptView Transcript.png

#### Add termsAdd Term.png

#### Edit termsEdit Term.png

#### Remove terms

#### Add Course Catalog EntriesAdd Course to Catalog.png

#### Edit Course Catalog EntriesEdit Course Catalog.png

#### Remove Course Catalog Entries

#### Add Sections to TermsAdd Section to Term.png

#### Edit Sections in TermsEdit Section in Term.png

#### Remove Sections from TermsRemove Sections from Terms.png

#### Add UserAdd User.png

#### Edit UserEdit User.png

#### Remove UserRemove User.png

#### View User InfoView User.png

#### Change PasswordChange Password.png

### Nonfunctional Requirements

* The system must respond to page requests in under 3 seconds.
* The system should be able to handle a minimum of 1000 students simultaneously in enrollment modules.
* Passwords will be stored as hashes and all traffic will be SSL encrypted.
* Page access is restricted by role type and roles are checked on each call.
* Users and Registrar must have a valid email address and password to log in to the system.
* Users must be enrolled in or employed by the University.
* Students may view their own grades only.
* Students have the option to change their password.
* Registrar generates the username for a new User. //this has changed (deleted pw)
* Instructors can drop a Student at any time.

# Glossary

**User(s)**: Used to describe both Students and Instructors in the system.

**Section(s)**: The individual instances of a course being offered. Sections differ by date, time, location, and instructor. Students register for the sections offered in a term.

**Course Catalog**: A page where Users can view all Courses offered by the University.

**authorization key**: An identifier to a specific user id stored in a browser after authentication.

# Appendix

### Appendix A: Grade Calculations

percentage format

An assignment grade represented in 0 through 100 percentage points; ie. a student may earn 25 points out of 50 on an exam, and their grade in percentage format is 50%.

cumulative format

An assignment, section, or term grade represented in the following ranges. It is either lettered or on a 4.0 scale.

|  |  |  |
| --- | --- | --- |
| **Letter Grade** | **4.0 Scaled Grade** | **Percentage Format** |
| A | 4.00 | 94 - 100 |
| A- | 3.67 | 90 - 93.9 |
| B+ | 3.33 | 87 - 79.9 |
| B | 3.00 | 84 - 86.9 |
| B- | 2.67 | 80 - 83.9 |
| C+ | 2.33 | 77 - 79.9 |
| C | 2.0 | 70 - 76.9 |
| D | 1.0 | 60 - 69.9 |
| F | 0.0 | 0 - 59.9 |

grade calculations

Grade is calculated from assignment weights and assignment group weights. Here is the pseudocode for calculating a total grade for a student.

total = 0;

For each assignment group:

groupUnweightedTotal = 0;

For each assignment:

groupUnweightedTotal += assignemnt\_grade \* assignment weight;

total += groupUnweightedTotal \* groupWeight

Total grade percentage is total/100

# Index

**A**

Actor, 2, 7, 9, 10, 12, 13, 14, 15, 16, 19, 20, 21, 22, 23

add/drop, 7

Assignment, 2, 3, 10, 11, 12, 13, 16, 17, 24, 29, 30, 46, 47

assignment group, 10, 11, 12, 46

authorization key, 9, 10, 11, 12, 13, 16, 17, 18, 19, 20, 21, 22, 23, 45

**C**

course, 5, 7, 14, 18, 19, 20, 25, 45

Course Catalog, 2, 3, 7, 14, 18, 19, 20, 24, 32, 39, 40, 45

courses, 5, 14, 16, 24

**E**

email, 7, 45

enrollment, 5, 7, 9, 16, 45

**G**

grade, 11, 12, 13, 16, 17, 46, 47

grading, 4, 5, 7, 12

**I**

instructor, 9, 11, 12, 13, 15, 20, 25, 45

Instructors, 2, 5, 7, 10, 45

**P**

password, 7, 14, 23, 24, 45

Passwords, 45

**R**

Registrar, 2, 5, 7, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 45

Registrars, 5, 7

registration, 4, 5, 7

**S**

Section, 2, 3, 5, 9, 10, 11, 12, 13, 15, 16, 17, 18, 20, 21, 24, 26, 27, 34, 35, 45, 46

student, 5, 9, 10, 11, 12, 13, 15, 16, 22, 24, 25, 46

Students, 2, 4, 5, 7, 10, 12, 13, 15, 17, 20, 21, 24, 45

**T**

term, 9, 13, 15, 16, 17, 18, 20, 21, 45, 46

**U**

University, 5, 45

User, 2, 3, 4, 7, 21, 22, 23, 24, 42, 43, 44, 45

Users, 7, 45

**W**

weight, 10, 11, 47

**Y**

year, 9, 17, 18