# Software Requirements Specification

## Co-op Evaluation System

*Senior Project 2014-2015*

**Team Members:**
Tyler Geery
Maddison Hickson
Casey Klimkowsky
Emma Nelson

**Faculty Coach:**
Samuel Malachowsky

**Project Sponsors:**
Jim Bondi (OCSCE)
Kim Sowers (ITS)

## Table of Contents

## Revision History

| Version | Primary Author(s) | Description of Version | Date Completed |
|---|---|---|---|
| v1.0 | Emma Nelson, Maddison Hickson, Casey Klimkowsky, Tyler Geery | Initial revision | October 6, 2014 |
| v1.1 | Maddison Hickson | Update after the Requirements phase receiving more information from Jim | October 16, 2014 |
| v1.2 | Casey Klimkowsky, Tyler Geery, Maddison Hickson | Update after receiving feedback from Jim on 10/23/14 | October 28, 2014 |
| v1.3 | Casey Klimkowsky, Tyler Geery, Maddison Hickson | Update after receiving feedback from Jim on 10/28/14 | October 30, 2014 |
| v1.4 | Emma Nelson | Updated VM Specification | October 30, 2014 |
| v1.5 | Casey Klimkowsky, Tyler Geery, Maddison Hickson | Update after receiving feedback from Kim on 10/28/14 | November 3, 2014 |
| v1.6 | Emma Nelson | Verified changes and added priority description | November 5, 2014 |
| v1.7 | Emma Nelson | Update to match changes going into development | February 18, 2015 |
| v1.8 | Emma Nelson | Removed redundant requirements and updated the priority of a few requirements | March 15, 2015 |
| v1.9 | Emma Nelson | Final version of the artifact before release | May 16, 2015 |

# 1 Introduction

## 1.1 Purpose

The purpose of the Co-op Evaluation System (CES) is to allow students to provide feedback on their most recent co-op, and for employers to provide feedback on a student's performance during their most recent co-op. Additionally, the system is used by faculty to approve or fail a student's co-op, and is also used by OCSCE to gather data on students' co-ops.

This SRS describes the entire scope of the new Co-op Evaluation System, which is elaborated upon in 1.3.

## 1.2 Problem

RIT's current Co-op Evaluation System, an application used by OCSCE, has a number of performance, reliability, usability, and maintainability issues. Among others, session timeouts and submission timeouts are inherit problems of the current Co-op Evaluation System. A new version started from scratch with up-to-date technologies needs to be developed.

## 1.3 Document Conventions

Each requirement statement is to have its own priority, which is listed next to the requirement itself. A priority of "High" indicates that the given requirement is top priority by the development team and is key to having a functional system. "Medium" priority requirements will be secondary; however, the development team still expects to complete as many of the associated features as time allows in order to deliver a functionally equivalent system. Requirements labelled with a "Low" priority are stretch goals and depend on the time available at the end of the development period. Most of these requirements are additional features beyond the scope of the current feature set. They are documented for use in future development either by ITS or another senior project team.

## 1.4 Project Scope

One of our primary goals is that by the conclusion of this project, we will supply OCSCE and ITS with a product that is functionally equivalent to the existing system, with fewer performance, reliability, and maintainability issues. Time permitting, we hope to implement a small number of enhancements, as defined by our project sponsors, as well. If the scope ends up being too much, our primary focus will shift to getting forms working end-to-end from creation and assignment all the way through to approval or rejection. We plan to design and implement the system with extensibility in mind, so that in the future, other developers may implement additional features that we were unable to implement during the duration of this project.

Several features that exist in the current system, but need to be drastically improved, are report generation, form generation, and error messages. Even though we are aiming for functional equivalence for these features, we intend to greatly improve their usability. We plan to address other major pain points, such as the ability to change a student's supervisor at a later date, as well.

## 1.5 Intended Audience

This document is intended to help the development team to validate that they are building the right application, and verify that the features they have built are built correctly. It is also intended for the project sponsors to sign off on as a definitive list of requirements. Finally, the project coach can use this document to validate the development team is meeting the agreed upon requirements during his evaluation of the team's efforts.

## 1.6 References

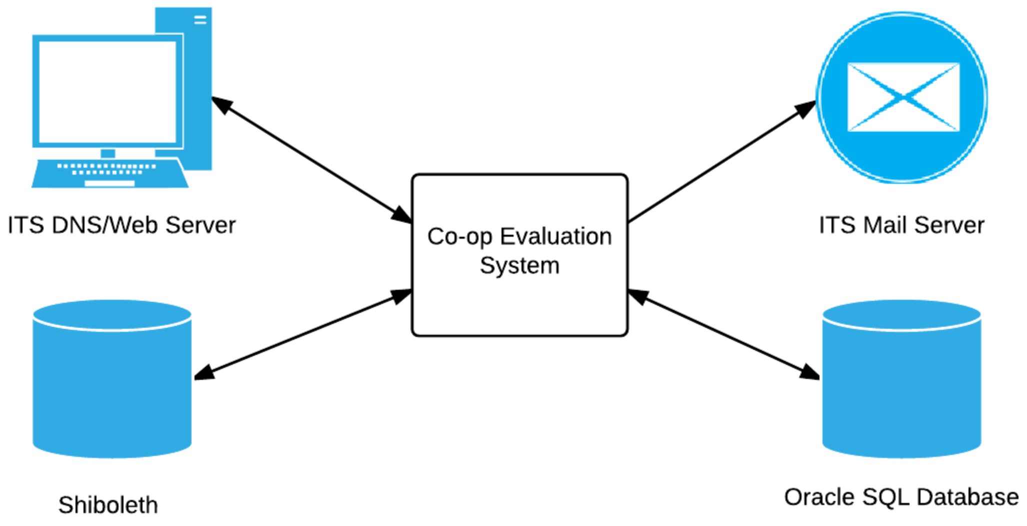
We were given the following materials to help us define the requirements of the system:

| Reference | Description |
|---|---|
| ITS Wiki Page | Created by the ITS team, and contains sample applications as well as guidelines we must abide by |
| Project Proposal | Written by the sponsors, and contains much of the background and high-level goals of the project. |
| Collection of all previous work on the project | This includes documentation and code from the previous senior project teams that worked on the Co-op Evaluation System. |
| RIT Web Standards | This document defines a set of standards that create a framework for proper usage with regard to language, graphics, and navigational architecture on all RIT-related websites. |

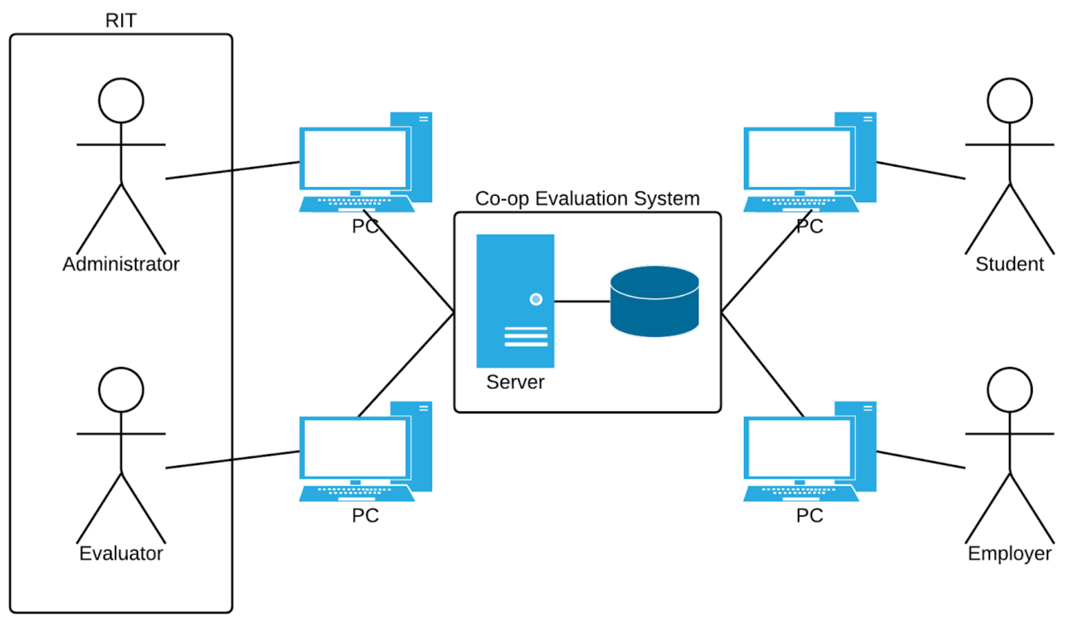
# 2 Overall Description

## 2.1 Product Perspective

The purpose of this project is to re-engineer the Co-op Evaluation System in order to leverage newer web technologies while also improving performance and user interaction. The current system uses outdated, under-documented technology, which makes it difficult to maintain. Furthermore, the random errors that occur do not give users confidence that their information was submitted properly. Significant improvements to the user interface will need to be made, but the existing database structures can be used as a reference for modifications.



The above diagram outlines the major components of the overall system, subsystem interconnections, and external interfaces, which are elaborated upon in Section 5.



The diagram above show a high-level view of the user interaction with the system as well as the interaction between technologies involved.

## 2.2 User Classes and Characteristics

| User | Description of Use | Frequency of Use |
|---|---|---|
| Student | Uses application to fill out a Work Report | 1 time per co-op block |
| Employer | Uses application to fill out an Employer Evaluation | 1 time per student per co-op block |
| Evaluator | Uses application to review the student's and employer's survey responses to determine the student's grade (S or F) | 1 time per student being evaluated |
| Administrator | Uses application to gather statistical data from survey responses | As needed to generate reports, create new forms, and perform other administrative tasks |

## 2.3 Evaluation Status States

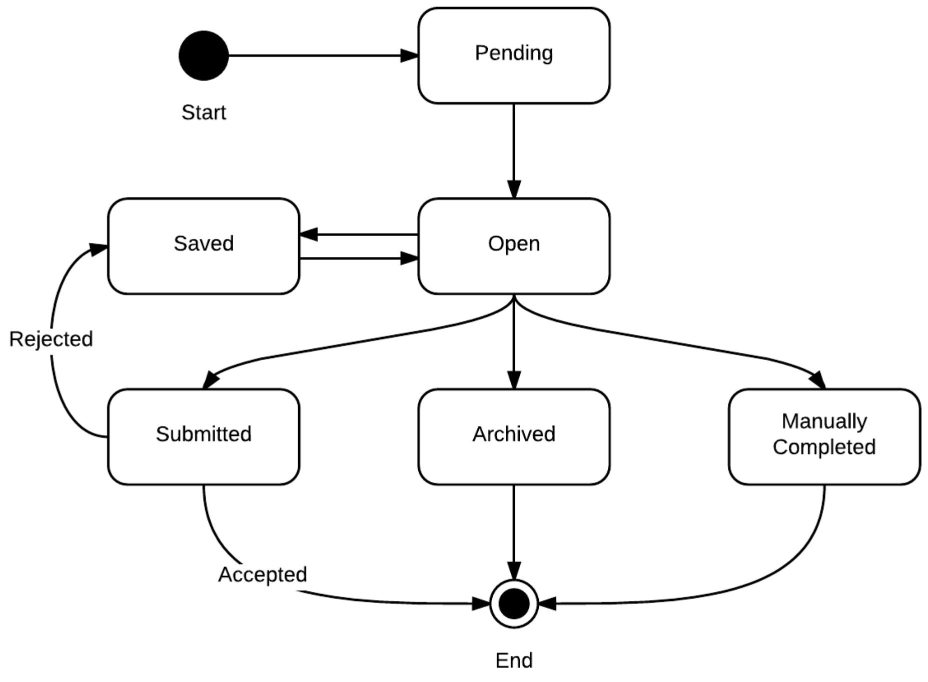
Evaluations, both Employer and Student, will start out as Pending. In the pending state, the evaluation appears in the system, but cannot be filled out yet.

Three weeks from the end of the term, all evaluations will be changed from Pending to Open. An evaluation in the Open state can be filled out by an Employer or Student that it is assigned to. An open evaluation can be changed to Saved, Submitted, Manually Completed, or Archived state.

If the user saved the evaluation, it is changed to the Saved state. In this state, the user can open the evaluation and continue filling out answers. They can continue saving the evaluation until they are finished. The Saved sate will be referred to as “In Progress” in the user interface. The evaluation can be changed to Submitted, Manually Completed, or Archived state from the Saved state.

The Submitted state is reached by submitting the form from either the Open state or the Saved state. In this state, the evaluation cannot be changed by any class of user. At this point, the only change that can be made to the evaluation is by an evaluation approval change. If the evaluation is approved, nothing else will happen. If the evaluation is rejected, the evaluation will go back to the Saved state.

There are two other states that an evaluation can end up in: Manually Completed and Archived. If the evaluation is completed in some way other than the normal process, an evaluation can be changed to the Manually Completed state. If the evaluation will never be completed, and the user does not want to receive messages telling them to fill out the evaluation, the evaluation can be marked as Archived. The Archived state was known as the Past Pending state in the previous version of the CES.



## 2.4 Operating Environment

Since the product will be a web-based application, the software is required to be accessible by numerous browsers, and different versions of each browser. The browser in which the application is accessed may be from a desktop computer, or a mobile device.

The required browsers and versions of each in which the system must be accessible from are as follows:

- Google Chrome, latest version
- Mozilla Firefox, latest version
- Safari, latest version
- Internet Explorer, 9+

The system itself shall be deployed and will operate on a VM provided by ITS. The system specifications of this VM are as follows:

- The server is currently running Tomcat 7.0.55. The version will be upgraded to 7.0.56 in January.
- The amount of memory will be adjusted to suit the needs of our application.
- The current configuration has both the minimum and maximum heap size is set to 512MB. If more space is required, another Tomcat instance will be spun up to distribute the load across two servers controlled by the same host.
- Hardware is shared between many applications and monitored by the ITS application support team.
- Both TEST and PROD servers are within the RIT network and will require use of the RIT VPN in order to conduct testing.

## 2.5 Design and Implementation Constraints

The system must comply with the development guidelines provided to us by ITS, as defined by the EWA Student Development Guidelines wiki page. At a high level, these guidelines include approved application frameworks, build tools, application server technologies, database standards, and several other technology standards.

The system must also comply with the RIT Web standards document, which defines the standards for RIT-related websites in regard to language, graphics, and navigational architecture.

## 2.6 Assumptions and Dependencies

Our biggest personal assumption is that our own experiences on co-op and with the old CES are representative of every student. However, we know this is not the case, so we will do our best to talk to our friends in other majors to glean their perspectives. We will use our experiences and those of other users as examples of what can happen, not as hard facts.

JobZone may become a potential dependency. Right now employers can log in there without an RIT computer account, and the hope is that the CES can get the authentication token from

JobZone to authenticate the user on the Co-op Evaluation System. However, this is still up in the air; there may be some re-planning around finding an alternative solution for Employer Authentication.

SIS was another potential tie-in on the Evaluator side; however, this is out of scope for this project. The sponsor was looking into finding out if the Evaluator role can be shifted to SIS for an easier interaction experience for faculty and staff who use SIS on a daily basis. In this case, the system would need to provide a portal for SIS to access information as needed, and to send messages of acceptance or rejection for Student Evaluations.

# 3 System Features

## 3.1 System Requirements

### 3.1.1 Form Administration

| Number | Priority | Requirement |
|---|---|---|
| F-01 | High | Student and Employer evaluations shall have the following progress statuses: Submitted, Saved, Open, Pending, Manually Completed, and Archived. Refer to Section 2.3 for more detail. |
| F-02 | High | Student evaluations shall have the following evaluation approval statuses: Submitted, Approved, and Rejected. |
| F-03 | Medium | The system shall record an evaluation approval status change trail. |
| F-04 | Medium | The system shall automatically change the progress status of all evaluations from Pending to Open after a configurable number of weeks before the end of a Student’s co-op. |
| F-05 | High | The system shall store evaluations in a database. |

### 3.1.2 Evaluations

| Number | Priority | Requirement |
|---|---|---|
| E-01 | Low | The system shall be able to automatically save non-submitted evaluation forms. |
| E-02 | High | The system shall be able to validate an evaluation for completeness. |
| E-03 | Medium | The system shall be able to validate an evaluation for correctness (e.g. email validation) the client side. |

### 3.1.3 Submissions

| Number | Priority | Requirement |
|---|---|---|
| S-01 | Medium | The system shall display the submissions in a format that is printable. |
| S-02 | High | The system shall be able to search forms based on student last name, student first name, student ID, company name, term, department, advisor's RIT Computer Account user name, student progress status, evaluation progress status, and employer evaluation status. |

### 3.1.4 Reports

| Number | Priority | Requirement |
|---|---|---|
| R-01 | Out of Scope | The system shall generate reports based on a user-defined selection of submissions and statistics. |
| R-02 | Out of Scope | The system shall produce statistics on all questions that have numeric answers. |
| R-03 | Out of Scope | The system shall produce statistics on all questions that have qualitative answers ("Comments Only"). |
| R-04 | Out of Scope | The system shall use an third-party service to generate reports that, at a minimum, supports the reports generated by the current system. |
| R-05 | Out of Scope | The system shall be able to accommodate the addition of a more reports as defined by the sponsor and other users of the CES. |
| R-06 | Out of Scope | The system should provide an easy way to select items to be included or excluded from the report. |
| R-07 | Out of Scope | The system should provide the ability to run reports across multiple year-levels (i.e. all undergraduate students). |
| R-08 | Out of Scope | The system should be able to export the SQL view created for the report. |
| R-09 | Out of Scope | The system shall be able to export the report data in CSV format for use in any spreadsheet application. |
| R-10 | Out of Scope | The system should be able to run reports on qualitative data. |

| | | |
|---|---|---|
| R-11 | Out of Scope | The system should display the qualitative questions on forms as selectable options when choosing the report settings. |
| R-12 | Out of Scope | The system shall take no longer than 3 minutes to generate a report. |
| R-13 | Out of Scope | The system shall have the ability run a report by academic year. |
| R-14 | Out of Scope | The system should have the ability to run a report by form. |

### 3.1.5 Email Notifications

| Number | Priority | Requirement |
|---|---|---|
| N-01 | Out of Scope | The system shall be able to send generated email notifications to Students and Evaluators manually. |
| N-02 | Out of Scope | The system shall be able to send generated email notifications to Students and Evaluators automatically. |
| N-03 | Out of Scope | The system shall be able to generate evaluation notifications to all Employers and Students a configurable number of weeks before the Student’s end date. |
| N-04 | Out of Scope | The system shall be able to generate a student confirmation email to Students and Employers. |
| N-05 | Out of Scope | The system shall be able to display notification statuses for Student and Employer notifications. (Use Case 4.1.4) |
| N-06 | Out of Scope | The system shall be able to display failed emails and sent emails in the notifications statuses. |
| N-07 | Out of Scope | The system shall send a notification email to a Student or Employer when their evaluation has been rejected. |
| N-08 | Out of Scope | The system shall send an email notification to Students confirming their supervisor, start date, and end date. |
| N-09 | Out of Scope | The system shall send a notification email to Students when their work report was successfully persisted. |
| N-10 | Out of Scope | The system shall send a notification email to Employers when their evaluation was successfully persisted. |

|  |  |  |
|---|---|---|
| N-11 | Out of Scope | The system shall send an email notification to Students and Employers when an evaluation has been Saved, but not submitted for a period of two weeks. |
| N-12 | Out of Scope | The system shall send an email notification to Evaluators when their students submit a work report. |

### 3.1.6 Users

| Number | Priority | Requirement |
|---|---|---|
| U-01 | High | The system shall use the user information captured from existing OCSCE database to provide authorization for employers. |
| U-02 | Medium | The system shall be able to automatically initialize the next term for Students and Employers based on the RIT academic calendar. |
| U-03 | High | The system shall be able to update the status of submissions when supplied with a new status (manually or automatically). |
| U-04 | High | The system shall use Shibboleth to authorize students, faculty, and OCSCE staff. |

## 3.2 User Requirements

### 3.2.1 Administrator

| Number | Priority | Requirement |
|---|---|---|
| AD-01 | High | An Administrator shall be able to search Student and Employer evaluations. |
| AD-02 | High | An Administrator shall be able to search Student and Employer submissions. |
| AD-03 | Medium | An Administrator shall be able to compare Student and Employer submissions, side-by-side. |
| AD-04 | High | An Administrator shall be able to create Student/Employer forms. |
| AD-05 | High | An Administrator shall be able to view Student/Employer forms. |
| AD-06 | High | An Administrator shall be able to update Student/Employer forms. |

| | | |
|---|---|---|
| AD-07 | Low | An Administrator shall be able to archive Student/Employer forms. |
| AD-08 | Medium | An Administrator shall be able to delete forms that have no submissions associated with them. |
| AD-09 | High | An Administrator shall be able to assign forms to departments. |
| AD-10 | Low | An Administrator shall be able to update the status of any groups of evaluations. |
| AD-11 | High | An Administrator shall be able to view the status of any evaluation. |
| AD-12 | Out of Scope | An Administrator shall be able to create and edit notifications. |
| AD-13 | Out of Scope | An Administrator shall be able to resend failed notifications. |
| AD-14 | Out of Scope | An Administrator shall be able to view the configurations of notifications. |
| AD-15 | Out of Scope | An Administrator shall be able to update the configurations of notifications. |
| AD-16 | Out of Scope | An Administrator shall be able to generate reports. |
| AD-17 | Low | An Administrator shall be able to copy the contents of an existing Student or Employer form to a new form. |
| AD-28 | High | An Administrator shall be able to add/remove OCSCE users to the system. |
| AD-19 | High | An Administrator shall be able to add/remove Academic Department (Evaluator) users from the system. |
| AD-20 | High | An Administrator shall be able to add/remove colleges and departments to the system. |
| AD-21 | Low | An Administrator shall be able to transfer Academic Department user privileges to another user. |
| AD-22 | High | An Administrator shall be able to import a file containing evaluations. |

### 3.2.2 Evaluator

| Number | Priority | Requirement |
|---|---|---|
| EV-01 | High | The Evaluator shall be able to view Student submissions associated with their department. |
| EV-02 | High | The Evaluator shall be able to view Employer submissions associated with their department. |
| EV-03 | Low | The Evaluator shall be able to view Student evaluations associated with double majors in their department. |
| EV-04 | Low | The Evaluator shall be able to view Employer evaluations associated with double majors in their department. |
| EV-05 | Low | The Evaluator shall be able to view Student submissions associated with double majors in their department. |
| EV-06 | Low | The Evaluator shall be able to view Employer submissions associated with double majors in their department. |
| EV-07 | Medium | The Evaluator shall be able to compare current Student and Employer submissions. |
| EV-09 | High | The Evaluator shall be able to accept or reject a submission. |
| EV-10 | High | The Evaluator shall be able to view the status for all evaluations associated with their department. |
| EV-11 | Out of Scope | The Evaluator shall be able to create notifications for their department. |
| EV-12 | Out of Scope | The Evaluator shall be able to view the configuration of the notifications. |
| EV-13 | Low | An Evaluator shall be able to change the status of a work report from archived to open. |

### 3.2.3 Student

| Number | Priority | Requirement |
|---|---|---|
| ST-01 | High | A Student shall be able to view their own work reports. |
| ST-02 | High | A Student shall be able to update their own work reports before submission unless it has been rejected. |
| ST-03 | High | A Student shall be able to submit their own work reports. |

| | | |
|---|---|---|
| ST-04 | High | A Student shall be able to view their own submissions. |
| ST-05 | High | A Student shall be able to view the submissions of their Employers. |
| ST-06 | High | A Student shall be able to view the status of their own work reports. |
| ST-07 | High | A Student shall be able to view the status of their Employers' evaluations. |

### 3.2.4 Employer

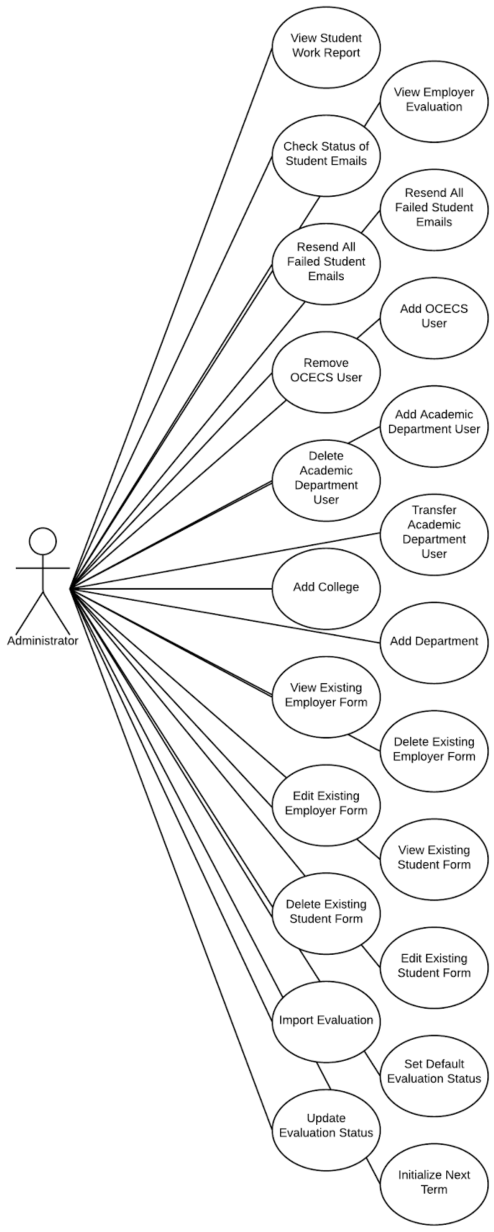
| Number | Priority | Requirement |
|---|---|---|
| EM-01 | High | The Employer shall be able to view their evaluations. |
| EM-02 | High | The Employer shall be able to update their evaluations. |
| EM-04 | High | The Employer shall be able to submit an evaluation. |
| EM-05 | High | The Employer shall be able to view their submissions. |
| EM-06 | High | The Employer shall be able to view the status of their evaluations. |

## 4 Use Cases

Refer to the User Analysis and Workflows document for a complete description of the human actors involved in the system.

## 4.1 Administrator

### 4.1.1 Use Case Context


View Student Work Report
View Employer Evaluation
Check Status of Student Emails
Resend All Failed Student Emails
Resend All Failed Student Emails
Add OCECS User
Remove OCECS User
Add Academic Department User
Delete Academic Department User
Transfer Academic Department User
Administrator
Add College
Add Department
View Existing Employer Form
Delete Existing Employer Form
Edit Existing Employer Form
View Existing Student Form
Delete Existing Student Form
Edit Existing Student Form
Import Evaluation
Set Default Evaluation Status
Update Evaluation Status
Initialize Next Term

### 4.1.2 View Student Work Report

| | |
|---|---|
| **Actors** | Administrator |
| **Description** | This use case describes how an Administrator uses the Co-op Evaluation System to view a Student's submitted work report. |
| **Trigger** | Administrator desires to look over a student's co-op evaluation. |

|  |  |
|---|---|
| **Pre-conditions** | 1. The Administrator has an Internet connection.<br>2. The Administrator is authenticated and signed into the system. |
| **Post-conditions** | The desired student's work report is displayed to the Administrator. |
| **Normal Flow** | 1. The Administrator selects "Evaluations".<br>2. The Administrator selects "Search Submitted/Pending Evaluations"<br>3. The Administrator inputs student information.<br>4. The Administrator clicks "Search" button.<br>5. The Administrator selects desired student for the chosen co-op block.<br>6. The use case ends successfully. |
| **Alternative Flows** | N/A |
| **Exceptions** | **Student is Not in the System**<br>If in step 4 of the normal flow the desired student does not exist in the system, then<br>1. The system shall display a message stating that the student does not exist.<br>2. The use case ends with a failure condition. |

### 4.1.3 View Employer Evaluation

|  |  |
|---|---|
| **Actors** | Administrator |
| **Description** | This use case describes how an Administrator uses the Co-op Evaluation System to view an Employer's submitted evaluation. |
| **Trigger** | Administrator desires to look over a student's co-op evaluation. |
| **Pre-conditions** | 1. The Administrator has an Internet connection.<br>2. The Administrator is authenticated and signed into the system. |
| **Post-conditions** | The desired employer's co-op evaluation of the student is displayed to the Administrator. |
| **Normal Flow** | 1. The Administrator selects "Evaluations".<br>2. The Administrator selects "Search Submitted/Pending Evaluations"<br>3. The Administrator clicks "Search" button.<br>4. The Administrator selects desired employer evaluation link for the chosen student and co-op block.<br>5. The Administrator selects the submission for the specific semester.<br>6. The use case ends successfully. |

| | |
|---|---|
| **Alternative Flows** | N/A |
| **Exceptions** | **Employer is Not in the System**<br>If in step 4 of the normal flow the desired employer does not exist in the system, then<br>1. There is no action to be taken by the Administrator.<br>2. The use case ends with a failure condition. |

### 4.1.4 Check Status of Emails

| | |
|---|---|
| **Actors** | Administrator |
| **Description** | This use case describes how an Administrator uses the Co-op Evaluation System to check the status of the reminder emails sent out to students and employers. |
| **Trigger** | Administrator desires to look over the status of emails sent to students and employers. |
| **Pre-conditions** | 1. The Administrator has an Internet connection.<br>2. The Administrator is authenticated and signed into the system. |
| **Post-conditions** | Administrator is viewing the given status of student and employer emails. |
| **Normal Flow** | 1. The Administrator selects “Email” from the navbar.<br>2. The Administrator selects “Email Status” from the dropdown.<br>3. The Administrator selects either “Student” or “Employer” to display the associated emails.<br>4. The Administrator views scheduled emails as well as the date and time for which the emails are scheduled.<br>5. The Administrator views failed emails and the reason for failure.<br>6. The use case ends successfully. |
| **Alternative Flows** | N/A |
| **Exceptions** | N/A |

### 4.1.5 Resend All Failed Emails

| | |
|---|---|
| **Actors** | Administrator |
| **Description** | This use case describes how an Administrator uses the Co-op Evaluation System to send out any email notifications that failed to send to students and employers. |

| | |
|---|---|
| **Trigger** | Administrator desires to send out any email notifications that failed to send to students and employers. |
| **Pre-conditions** | 1. The Administrator has an Internet connection.<br>2. The Administrator is authenticated and signed into the system. |
| **Post-conditions** | The Administrator resent all “failed” emails. |
| **Normal Flow** | 1. The Administrator selects “Email” from the navbar.<br>2. The Administrator selects “Email Status” from the dropdown.<br>3. The Administrator selects either “Student” or “Employer” to display the associated emails.<br>4. The Administrator sees a list of all emails that failed to send.<br>5. The Administrator selects “Resend All Failed Emails” from page.<br>6. The use case ends successfully. |
| **Alternative Flows** | N/A |
| **Exceptions** | **No Failed Emails in System**<br>If in step 4 of the normal flow there are no emails that failed to send, then<br>1. The Administrator does nothing.<br>2. The use case ends with a failure condition. |

### 4.1.6 Add and Administrator

| | |
|---|---|
| **Actors** | Administrator |
| **Description** | This use case describes how an administrator uses the Co-op Evaluation System to add an administrator to the system. |
| **Trigger** | Administrator desires to add an administrator to the system. |
| **Pre-conditions** | 1. The Administrator has an Internet connection.<br>2. The Administrator is authenticated and signed into the system. |
| **Post-conditions** | Administrator successfully added the administrator to the system. |
| **Normal Flow** | 1. The Administrator accesses “Users” in the navigation bar.<br>2. The Administrator chooses “Administrators” from the dropdown menu.<br>3. The Administrator clicks the “Add new” button.<br>4. The Administrator enters the user’s RIT account information.<br>5. The Administrator selects the “Add” button.<br>6. The Administrator verifies user has been added to the system.<br>7. The use case ends successfully. |

| | |
|---|---|
| **Alternative Flows** | N/A |
| **Exceptions** | **User Already in the System**<br>If in step 2 of the normal flow the user is already an administrator, then<br>1. There is no action to be taken by the Administrator.<br>2. The use case ends with a failure condition. |

### 4.1.7 Remove an Administrator

| | |
|---|---|
| **Actors** | Administrator |
| **Description** | This use case describes how an administrator uses the Co-op Evaluation System to remove an administrator from the system. |
| **Trigger** | Administrator desires to remove an administrator to the system. |
| **Pre-conditions** | 1. The Administrator has an Internet connection.<br>2. The Administrator is authenticated and signed into the system. |
| **Post-conditions** | Administrator successfully removed the administrator from the system. |
| **Normal Flow** | 1. The Administrator accesses “Users” in the navigation bar.<br>2. The Administrator chooses “Administrators” from the dropdown menu.<br>3. The Administrator searches for the user in the table listing all current users.<br>4. The Administrator locates user to be removed from the system.<br>5. The Administrator clicks the “Remove” button in the row listing the selected user.<br>6. The use case ends successfully. |
| **Alternative Flows** | N/A |
| **Exceptions** | N/A |

### 4.1.8 Add Academic Department User

| | |
|---|---|
| **Actors** | Administrator |
| **Description** | This use case describes how an Administrator uses the Co-op Evaluation System to add an Academic Department User to the system. |
| **Trigger** | Administrator desires to add an Academic Department User to the system. |
| **Pre-conditions** | 1. The Administrator has an Internet connection. |

|  |  |
|---|---|
|  | 2. The Administrator is authenticated and signed into the system. |
| **Post-conditions** | Administrator successfully added an Academic Department User to the system. |
| **Normal Flow** | 1. The Administrator accesses “Users” in the navigation bar.<br>2. The Administrator chooses “Department Users” from the dropdown menu.<br>3. The Administrator clicks the “Add new” button.<br>4. The Administrator enters the user’s RIT account information.<br>5. The Administrator selects the “Add” button.<br>6. The Administrator verifies user has been added to the system.<br>7. The use case ends successfully. |
| **Alternative Flows** | N/A |
| **Exceptions** | **Department User Already in the System**<br>If in step 5 of the normal flow the user is already an Academic Department User, then<br>1. There is no action to be taken by the Administrator.<br>2. The use case ends with a failure condition. The system returns with an error message. |

### 4.1.9 Delete Academic Department User

|  |  |
|---|---|
| **Actors** | Administrator |
| **Description** | This use case describes how an Administrator uses the Co-op Evaluation System to remove an Academic Department User from the system. |
| **Trigger** | Administrator desires to remove an Academic Department User from the system. |
| **Pre-conditions** | 1. The Administrator has an Internet connection.<br>2. The Administrator is authenticated and signed into the system. |
| **Post-conditions** | Administrator successfully deletes an Academic Department User from the system. |
| **Normal Flow** | 1. The Administrator accesses “Users” in the navigation bar.<br>2. The Administrator chooses “Department Users” from the dropdown menu.<br>3. The Administrator searches for the user in the table listing all current users.<br>4. The Administrator locates user to be removed from the system. |

| | |
|---|---|
| | 5. The Administrator clicks the “Remove” button in the row listing the selected user.<br>6. The use case ends successfully. |
| **Alternative Flows** | N/A |
| **Exceptions** | N/A |

### 4.1.10 Transfer Academic Department User

| | |
|---|---|
| **Actors** | Administrator |
| **Description** | This use case describes how an Administrator uses the Co-op Evaluation System to transfer an existing user’s privileges in the system to another user. |
| **Trigger** | Administrator desires to transfer existing user in the system to another department. |
| **Pre-conditions** | 1. The Administrator has an Internet connection.<br>2. The Administrator is authenticated and signed into the system. |
| **Post-conditions** | Administrator successfully transferred user to another department. |
| **Normal Flow** | 1. The Administrator accesses “Users” in the navigation bar.<br>2. The Administrator chooses “Transfer Department Users” from the dropdown menu.<br>3. The Administrator selects the source college the department is currently under.<br>4. The Administrator selects the department.<br>5. The Administrator chooses the destination college the department users will be under.<br>6. The Administrator selects the new department.<br>7. The Administrator clicks the “Transfer” button.<br>8. The use case ends successfully. |
| **Alternative Flows** | N/A |
| **Exceptions** | N/A |

### 4.1.11 Add College

| | |
|---|---|
| **Actors** | Administrator |
| **Description** | This use case describes how an Administrator uses the Co-op Evaluation System to add a new college to the system. |

| | |
|---|---|
| **Trigger** | Administrator desires to add a new college to the system. |
| **Pre-conditions** | 1. The Administrator has an Internet connection.<br>2. The Administrator is authenticated and signed into the system. |
| **Post-conditions** | Administrator successfully added a new college to the system. |
| **Normal Flow** | 1. The Administrator accesses “Users” in the navigation bar.<br>2. The Administrator chooses “College Management” from the dropdown menu.<br>3. The Administrator clicks the “Add New” button.<br>4. The Administrator enters desired college name.<br>5. The Administrator selects the “Add” button.<br>6. The use case ends successfully. |
| **Alternative Flows** | N/A |
| **Exceptions** | **College Already in the System**<br>If in step 5 of the normal flow the college is already in the system, then<br>1. There is no action to be taken by the Administrator.<br>2. The use case ends with a failure condition. The system will return with an error message. |

### 4.1.12 Remove College

| | |
|---|---|
| **Actors** | Administrator |
| **Description** | This use case describes how an Administrator uses the Co-op Evaluation System to remove a college from the system. |
| **Trigger** | Administrator desires to remove a college from the system. |
| **Pre-conditions** | 1. The Administrator has an Internet connection.<br>2. The Administrator is authenticated and signed into the system. |
| **Post-conditions** | Administrator successfully removed a college from the system. |
| **Normal Flow** | 1. The Administrator accesses “Users” in the navigation bar.<br>2. The Administrator chooses “College Management” from the dropdown menu.<br>3. The Administrator selects the college to be removed.<br>4. The Administrator selects the “Remove” button.<br>5. The use case ends successfully. |
| **Alternative Flows** | N/A |
| **Exceptions** | **Form Submission for College** |

|  |  |
|---|---|
|  | If in step 3 of the normal flow the college to be deleted has an existing form submission associated with it, then<br>1. The system displays a message saying that college cannot be deleted because there is an existing form submission under that college.<br>2. The use case ends with a failure condition. |

### 4.1.13 Add Department

|  |  |
|---|---|
| **Actors** | Administrator |
| **Description** | This use case describes how an Administrator uses the Co-op Evaluation System to add a new department to the system. |
| **Trigger** | Administrator desires to add a new department to the system. |
| **Pre-conditions** | 1. The Administrator has an Internet connection.<br>2. The Administrator is authenticated and signed into the system. |
| **Post-conditions** | Administrator successfully added a new department to the system. |
| **Normal Flow** | 1. The Administrator accesses “Users” in the navigation bar.<br>2. The Administrator chooses “Department Management” from the dropdown menu.<br>3. The Administrator selects the college to which the new department will belong.<br>4. The Administrator clicks the “Add New” button.<br>5. The Administrator enters desired department name.<br>6. The Administrator enters desired department code.<br>7. The Administrator clicks the “Add” button.<br>8. The Administrator verifies department was added.<br>9. The use case ends successfully. |
| **Alternative Flows** | N/A |
| **Exceptions** | **Department Already in the System**<br>If in step 7 of the normal flow the department is already in the system, then<br>1. There is no action to be taken by the Administrator.<br>2. The use case ends with a failure condition. The system will return with an error message. |

### 4.1.14 Remove Department

|  |  |
|---|---|
| **Actors** | Administrator |
| **Description** | This use case describes how an Administrator uses the Co-op |

| | |
|---|---|
| | Evaluation System to remove a department from the system. |
| **Trigger** | Administrator desires to remove a department from the system. |
| **Pre-conditions** | 1. The Administrator has an Internet connection.<br>2. The Administrator is authenticated and signed into the system. |
| **Post-conditions** | Administrator successfully removed a department from the system. |
| **Normal Flow** | 1. The Administrator accesses "Users" in the navigation bar.<br>2. The Administrator chooses "Department Management" from the dropdown menu.<br>3. The Administrator selects the college to which the new department will belong.<br>4. The Administrator locates the department name/code desired to be removed.<br>5. The Administrator selects the "Remove" button.<br>6. The use case ends successfully. |
| **Alternative Flows** | N/A |
| **Exceptions** | **Form Submission for Department**<br>If in step 5 of the normal flow the department to be deleted has an existing form submission associated with it, then<br>1. The system displays a message saying that department cannot be deleted because there is an existing form submission under that department.<br>2. The use case ends with a failure condition. |

### 4.1.15 Create a New Employer Form

| | |
|---|---|
| **Actors** | Administrator |
| **Description** | This use case describes how an Administrator uses the Co-op Evaluation System to create a new Employer form. |
| **Trigger** | Administrator desires to create a new Employer form. |
| **Pre-conditions** | 1. The Administrator has an Internet connection.<br>2. The Administrator is authenticated and signed into the system. |
| **Post-conditions** | Administrator successfully created a new Employer form. |
| **Normal Flow** | 1. The Administrator selects "Forms" from the navigation menu.<br>2. The Administrator selects "Employer Forms" under the Forms dropdown.<br>3. The Administrator clicks the "Add New" button. |

| | |
|---|---|
| | 4. The Administrator types in the name for the new form.<br>5. The Administrator creates the new blank form.<br>6. The use case ends successfully. |
| **Alternative Flows** | N/A |
| **Exceptions** | **Form Failed to Submit**<br>If in step 5 of the normal flow an error occurs when the Administrator attempts to submit the form, then<br>1. The system displays an error message stating what went wrong.<br>2. The use case ends with a failure condition. |

### 4.1.16 View Existing Employer Form

| | |
|---|---|
| **Actors** | Administrator |
| **Description** | This use case describes how an Administrator uses the Co-op Evaluation System to view an existing form for a given department. |
| **Trigger** | Administrator desires to view existing form for given department. |
| **Pre-conditions** | 1. The Administrator has an Internet connection.<br>2. The Administrator is authenticated and signed into the system. |
| **Post-conditions** | Administrator successfully viewed an existing employer form. |
| **Normal Flow** | 1. The Administrator selects “Forms” from the navigation menu.<br>2. The Administrator selects “Employer Forms” under the Forms dropdown.<br>3. The Administrator clicks the name of an existing employer form.<br>4. The use case ends successfully. |
| **Alternative Flows** | N/A |
| **Exceptions** | N/A |

### 4.1.17 Delete Existing Employer Form

| | |
|---|---|
| **Actors** | Administrator |
| **Description** | This use case describes how an Administrator uses the Co-op Evaluation System to delete an existing form for a given department. |
| **Trigger** | Administrator desires to delete Existing form for given department |
| **Pre-conditions** | 1. The Administrator has an Internet connection. |

|  |  |
|---|---|
|  | 2. The Administrator is authenticated and signed into the system. |
| **Post-conditions** | Administrator successfully deleted an existing employer form. |
| **Normal Flow** | 1. The Administrator selects “Forms” from the navigation menu.<br>2. The Administrator selects “Employer Forms” under the Forms dropdown.<br>3. The Administrator locates the form.<br>4. The Administrator clicks the “Delete” button.<br>5. The use case ends successfully. |
| **Alternative Flows** | N/A |
| **Exceptions** | **Employer Submission for Selected Form**<br>If in step 3 of the normal flow the form to be deleted has an existing employer submission, then<br>1. The system displays a message saying that form cannot be deleted because there is an existing submission for that form.<br>2. The use case ends with a failure condition. |

### 4.1.18 Edit Existing Employer Form

|  |  |
|---|---|
| **Actors** | Administrator |
| **Description** | This use case describes how an Administrator uses the Co-op Evaluation System to edit an existing form for a given department. |
| **Trigger** | Administrator desires to edit existing form for given department. |
| **Pre-conditions** | 1. The Administrator has an Internet connection.<br>2. The Administrator is authenticated and signed into the system. |
| **Post-conditions** | Administrator successfully edited an existing employer form. |
| **Normal Flow** | 1. The Administrator selects “Forms” from the navigation menu.<br>2. The Administrator selects “Employer Forms” under the Forms dropdown.<br>3. The Administrator selects the desired form.<br>4. The Administrator clicks the “Edit” button.<br>5. The Administrator makes desired changes to the form.<br>6. The Administrator scrolls to the bottom of the form, and clicks the “Save Changes” button.<br>7. The use case ends successfully. |
| **Alternative Flows** | **Cancel Changes**<br>If in step 5 of the normal flow the Administrator desires to cancel their changes, then |

| | |
|---|---|
| | 1. The Administrator scrolls to the bottom of the form, and clicks the “Cancel” button.<br>2. The use case ends successfully. |
| **Exceptions** | N/A |

### 4.1.19 Create a New Student Form

| | |
|---|---|
| **Actors** | Administrator |
| **Description** | This use case describes how an Administrator uses the Co-op Evaluation System to create a new Student form. |
| **Trigger** | Administrator desires to create a new Student form. |
| **Pre-conditions** | 1. The Administrator has an Internet connection.<br>2. The Administrator is authenticated and signed into the system. |
| **Post-conditions** | Administrator successfully created a new Student form. |
| **Normal Flow** | 1. The Administrator selects “Forms” from the navigation menu.<br>2. The Administrator selects “Student Forms” under the Forms dropdown.<br>3. The Administrator clicks the “Add New” button.<br>4. The Administrator adds a name to the form.<br>5. The Administrator creates the blank form.<br>6. The use case ends successfully. |
| **Alternative Flows** | N/A |
| **Exceptions** | **Form Failed to Submit**<br>If in step 5 of the normal flow an error occurs when the Administrator attempts to submit the form, then<br>1. The system displays an error message stating what went wrong.<br>2. The use case ends with a failure condition. |

### 4.1.20 View Existing Student Form

| | |
|---|---|
| **Actors** | Administrator |
| **Description** | This use case describes how an Administrator uses the Co-op Evaluation System to view an existing form for a given department. |
| **Trigger** | Administrator desires to view existing form for given department. |
| **Pre-conditions** | 1. The Administrator has an Internet connection.<br>2. The Administrator is authenticated and signed into the system. |

| | |
|---|---|
| **Post-conditions** | Administrator successfully viewed an existing student form. |
| **Normal Flow** | 1. The Administrator selects “Forms” from the navigation menu.<br>2. The Administrator selects “Student Forms” under the Forms dropdown.<br>3. The Administrator selects the form.<br>4. The Administrator clicks the “View” button.<br>5. The use case ends successfully. |
| **Alternative Flows** | N/A |
| **Exceptions** | N/A |

### 4.1.21 Delete Existing Student Form

| | |
|---|---|
| **Actors** | Administrator |
| **Description** | This use case describes how an Administrator uses the Co-op Evaluation System to delete an existing form for a given department. |
| **Trigger** | Administrator desires to delete existing form for given department. |
| **Pre-conditions** | 1. The Administrator has an Internet connection<br>2. The Administrator is authenticated and signed into the system. |
| **Post-conditions** | Administrator successfully deleted an existing student form. |
| **Normal Flow** | 1. The Administrator selects “Forms” from the navigation menu.<br>2. The Administrator selects “Student Forms” under the Forms dropdown.<br>3. The Administrator selects the form.<br>4. The Administrator clicks the “Delete” button.<br>5. The use case ends successfully. |
| **Alternative Flows** | N/A |
| **Exceptions** | **Student Submission for Selected Form**<br>If in step 4 of the normal flow the form to be deleted has an existing student submission, then<br>3. The system displays a message saying that form cannot be deleted because there is an existing submission for that form.<br>4. The use case ends with a failure condition. |

### 4.1.22 Edit Existing Student Form

| | |
|---|---|
| **Actors** | Administrator |

| | |
|---|---|
| **Description** | This use case describes how an Administrator uses the Co-op Evaluation System to edit an existing form for a given department. |
| **Trigger** | Administrator desires to edit existing form for given department. |
| **Pre-conditions** | 1. The Administrator has an Internet connection.<br>2. The Administrator is authenticated and signed into the system. |
| **Post-conditions** | Administrator successfully edited an existing student form. |
| **Normal Flow** | 1. The Administrator selects “Forms” from the navigation menu.<br>2. The Administrator selects “Student Forms” under the Forms dropdown.<br>3. The Administrator selects the desired form.<br>4. The Administrator clicks the “Edit” button.<br>5. The Administrator makes desired changes to the form.<br>6. The Administrator clicks the “Save Changes” button.<br>7. The use case ends successfully. |
| **Alternative Flows** | **Cancel Changes**<br>If in step 6 of the normal flow the Administrator desires to cancel their changes, then<br>1. The Administrator scrolls to the bottom of the form, and clicks the “Cancel” button.<br>2. The use case ends successfully. |
| **Exceptions** | N/A |

### 4.1.23 Import Evaluation

| | |
|---|---|
| **Actors** | Administrator |
| **Description** | This use case describes how an Administrator uses the Co-op Evaluation System to import evaluation information for students. |
| **Trigger** | Administrator desires import a newly registered co-ops . |
| **Pre-conditions** | 1. The Administrator has an Internet connection.<br>2. The Administrator is authenticated and signed into the system.<br>3. The Administrator has an import file. |
| **Post-conditions** | Administrator successfully imported evaluations into the system. |
| **Normal Flow** | 1. The Administrator selects “Evaluations” from the navigation menu.<br>2. The Administrator selects “Evaluation Management” under the Evaluations dropdown. |

| | |
|---|---|
| | 3. Under “Import Evaluations”, the Administrator chooses a file to import.<br>4. The Administrator clicks the “Upload” button.<br>5. The use case ends successfully. |
| **Alternative Flows** | N/A |
| **Exceptions** | **File is Not Properly Formatted**<br>If in step 3 of the normal flow the file uploaded is not in the correct format, then<br>1. The system displays a message saying that the file is in an invalid format.<br>2. The use case ends with a failure condition. |

### 4.1.24 Set Default Evaluation Status

| | |
|---|---|
| **Actors** | Administrator |
| **Description** | This use case describes how an Administrator uses the Co-op Evaluation System to set a default evaluation status for a college. |
| **Trigger** | Administrator desires needs to change the default status for a specific college. |
| **Pre-conditions** | 1. The Administrator has an Internet connection.<br>2. The Administrator is authenticated and signed into the system.<br>3. The college and department already exists in the system.<br>4. The evaluation status already exists. |
| **Post-conditions** | Administrator successfully changed the default evaluation status for a specific college or department. |
| **Normal Flow** | 1. The Administrator selects “Evaluations” from the navigation menu.<br>2. The Administrator selects “Evaluation Management” under the Evaluations dropdown.<br>3. Under “Default Evaluation Status”, the Administrator chooses a college from the drop-down menu.<br>4. The Administrator chooses a department from the dropdown.<br>5. The Administrator chooses the new default evaluation status.<br>6. The Administrator clicks the “Set Status” button.<br>7. The use case ends successfully. |
| **Alternative Flows** | N/A |
| **Exceptions** | N/A |

### 4.1.25 Update Evaluation Status

| | |
|---|---|
| **Actors** | Administrator |
| **Description** | This use case describes how an Administrator uses the Co-op Evaluation System to update the evaluation status for a college. |
| **Trigger** | Administrator desires to update the evaluation status for a college. |
| **Pre-conditions** | 1. The Administrator has an Internet connection.<br>2. The Administrator is authenticated and signed into the system.<br>3. The college and department already exists in the system.<br>4. The evaluation status already exists. |
| **Post-conditions** | Administrator successfully updated the evaluation status for a college. |
| **Normal Flow** | 1. The Administrator selects “Utilities” from navigation bar.<br>2. The Administrator selects “Update States” from the dropdown menu.<br>3. The Administrator chooses the current evaluation status.<br>4. The Administrator chooses the new evaluation status.<br>5. The Administrator chooses the evaluation type.<br>6. The Administrator enters the semester id number.<br>7. The Administrator chooses any number of departments from the multi-select box.<br>8. The Administrator clicks the “Update” button.<br>9. The use case ends successfully. |
| **Alternative Flows** | N/A |
| **Exceptions** | N/A |

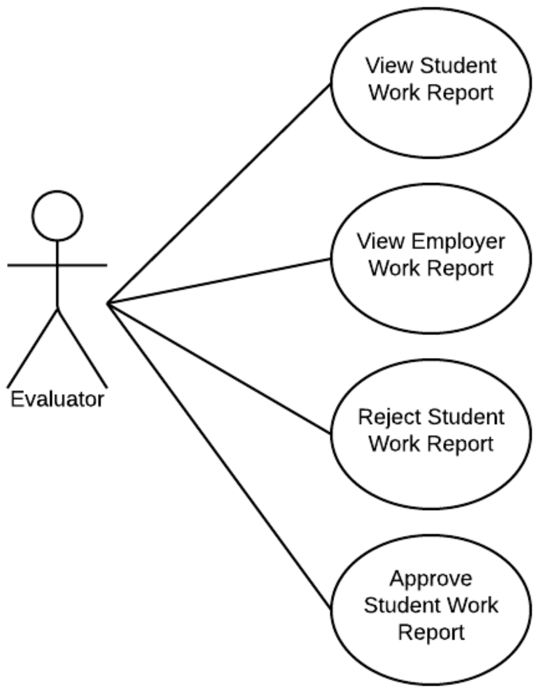
### 4.1.26 Initialize Next Term

| | |
|---|---|
| **Actors** | Administrator |
| **Description** | This use case describes how an Administrator uses the Co-op Evaluation System to initialize the next term in which students will be on co-op. |
| **Trigger** | Administrator desires to initialize the next term that students will be on co-op. |
| **Pre-conditions** | 1. The Administrator has an Internet connection.<br>2. The Administrator is authenticated and signed into the system.<br>3. The term has not already been initialized. |

| | |
|---|---|
| **Post-conditions** | Administrator successfully initialized the next term. |
| **Normal Flow** | 1. The Administrator selects “Utilities” from navigation bar.<br>2. The Administrator selects “Initialize Next Term” from the dropdown menu.<br>3. The Administrator selects whether to initialize the term for students and/or employers.<br>4. The Administrator clicks the “Initialize Term” button.<br>5. The use case ends successfully. |
| **Alternative Flows** | N/A |
| **Exceptions** | N/A |

## 4.2 Evaluator

### 4.2.1 Use Case Context


Evaluator
View Student Work Report
View Employer Work Report
Reject Student Work Report
Approve Student Work Report

### 4.2.2 View Student Work Report

| | |
|---|---|
| **Actors** | Evaluator |
| **Description** | This use case describes how an Evaluator uses the Co-op Evaluation System to view a Student’s submitted work report. |

| | |
|---|---|
| **Trigger** | Evaluator has received a notification about the students completion and desires to look over a student's co-op evaluation. |
| **Pre-conditions** | 1. Evaluator has an Internet connection.<br>2. Evaluator is authenticated and signed into the system.<br>3. The student has already finished their work report. |
| **Post-conditions** | Evaluator is viewing the given student's work report evaluation. |
| **Normal Flow** | 1. The Evaluator accesses "Evaluations" from the nav bar.<br>2. The Evaluator inputs student information.<br>3. The Evaluator clicks the "Search" button.<br>4. The Evaluator selects the student work report for the specific semester.<br>5. The use case ends successfully. |
| **Alternative Flows** | N/A |
| **Exceptions** | **Student is Not in the System**<br>If in step 3 of the normal flow the desired student does not exist in the system, then<br>1. The system shall display a message stating that the student does not exist.<br>2. The use case ends with a failure condition. |

### 4.2.3 View Employer Evaluation

| | |
|---|---|
| **Actors** | Evaluator |
| **Description** | This use case describes how an Evaluator uses the Co-op Evaluation System to view an Employer's submitted evaluation. |
| **Trigger** | Evaluator has received a notification that the Employer has completed their evaluation and desires to look it over. |
| **Pre-conditions** | 1. Evaluator has an Internet connection.<br>2. Evaluator is authenticated and signed into the system.<br>3. Employer has already completed their evaluation of the student. |
| **Post-conditions** | Evaluator is viewing the given employer's evaluation. |
| **Normal Flow** | 1. The Evaluator accesses "Evaluations" from the nav bar.<br>2. The Evaluator inputs desired search criteria.<br>3. The Evaluator clicks the "Search" button.<br>4. The Evaluator selects the employer evaluation for the specific semester.<br>5. The use case ends successfully. |

| | |
|---|---|
| **Alternative Flows** | N/A |
| **Exceptions** | **Employer is Not in the System**<br>If in step 4 of the normal flow the desired employer does not exist in the system, then<br>1. The system shall display a message stating that the employer does not exist.<br>2. The use case ends with a failure condition. |

### 4.2.4 Reject Student Work Report

| | |
|---|---|
| **Actors** | Evaluator |
| **Description** | This use case describes how an Evaluator uses the Co-op Evaluation System to reject the student's work report and have it sent back to the student to redo. |
| **Trigger** | Evaluator is not satisfied with the student's work report, and there is a desire for the student to re-do their submission. |
| **Pre-conditions** | 1. Evaluator has an Internet connection.<br>2. Evaluator is authenticated and signed into the system.<br>3. Student has already submitted their work report. |
| **Post-conditions** | Evaluator has changed the status of the work report from Pending Approval to Rejected. |
| **Normal Flow** | 1. The Evaluator accesses "Evaluations" from the nav bar.<br>2. The Evaluator inputs student information.<br>3. The Evaluator clicks the "Search" button.<br>4. The Evaluator selects the desired student.<br>5. The Evaluator selects "Reject" for the work report for the specific semester.<br>6. The use case ends successfully. |
| **Alternative Flows** | N/A |
| **Exceptions** | **Student is Not in the System**<br>If in step 3 of the normal flow the desired student does not exist in the system, then<br>1. The system shall display a message stating that the student does not exist.<br>2. The use case ends with a failure condition. |

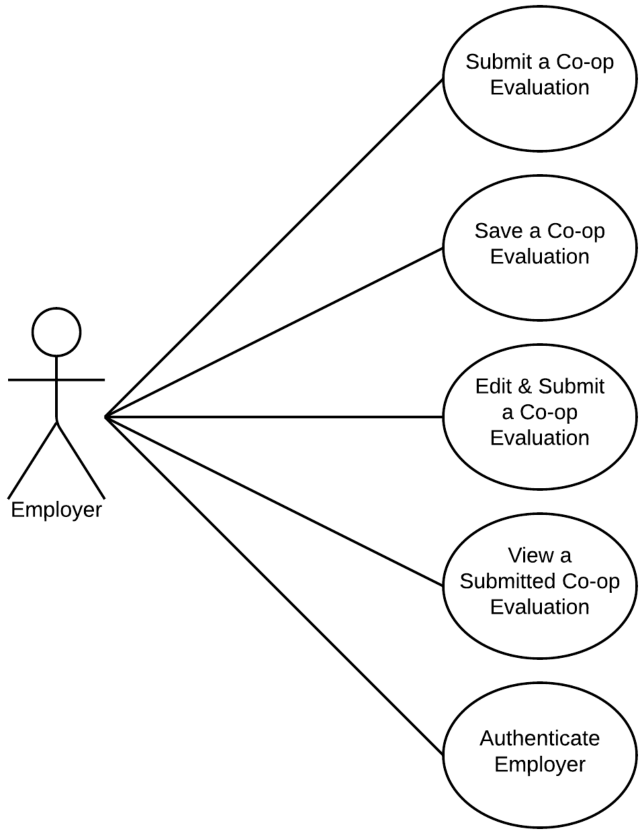
### 4.2.5 Accept Student Work Report

| | |
|---|---|
| **Actors** | Evaluator |

|  |  |
|---|---|
| **Description** | This use case describes how an Evaluator uses the Co-op Evaluation System to accept the student's work report. |
| **Trigger** | Evaluator is satisfied with the student's work report after reviewing it, and wants to accept it. |
| **Pre-conditions** | 1. Evaluator has an Internet connection.<br>2. Evaluator is authenticated and signed into the system.<br>3. Student has already submitted their work report. |
| **Post-conditions** | Evaluator has changed the status of the work report from Pending Approval to Approved. |
| **Normal Flow** | 1. The Evaluator accesses "Evaluations" from the nav bar.<br>2. The Evaluator inputs student information<br>3. The Evaluator clicks the "Search" button<br>4. The Evaluator selects for the desired student.<br>5. The Evaluator selects "Accept" for the work report for the specific semester.<br>6. The use case ends successfully. |
| **Alternative Flows** | N/A |
| **Exceptions** | **Student is Not in the System**<br>If in step 3 of the normal flow the desired student does not exist in the system, then<br>1. The system shall display a message stating that the student does not exist.<br>2. The use case ends with a failure condition. |

## 4.3 Employer

### 4.3.1 Use Case Context


Submit a Co-op Evaluation
Save a Co-op Evaluation
Employer
Edit & Submit a Co-op Evaluation
View a Submitted Co-op Evaluation
Authenticate Employer

### 4.3.2 Submit a Co-op Evaluation

| | |
|---|---|
| **Actors** | Employer |
| **Description** | This use case describes how the Employer uses the Co-op Evaluation System to submit a new Co-op Evaluation. |
| **Trigger** | Employer has a Co-op Student for which an Evaluation needs to be submitted and has received an email telling them the form is available. |
| **Pre-conditions** | 1. The Employer has an active Internet connection.<br>2. The Employer has a registered Co-op Student. |
| **Post-conditions** | The Co-op Evaluation has been submitted for the Co-op Student. |
| **Normal Flow** | 1. The Employer locates the panel for the student in question under “Current Co-op Students”. |

| | |
|---|---|
| | 2. The Employer selects the “Open” link next “Employer Eval” in the given student co-op panel.<br>3. The Employer completes the Co-op Evaluation.<br>4. The Employer submits the Co-op Evaluation.<br>5. The system displays a message saying that the Co-op Evaluation has been submitted successfully.<br>6. The use case ends successfully. |
| **Alternative Flows** | **All Required Fields Have Not Been Completed**<br>If in step 5 of the normal flow the Employer submits the Co-op Evaluation without having completed all required fields, then<br>1. The system shall display an error message, and ask the Employer to fill out all required fields that have not been completed.<br>2. The use case resumes at step 4. |
| **Exceptions** | **Invalid User**<br>If the use case Authenticate Employer does not complete successfully, then<br>1. The use case ends with a failure condition.<br><br>**Error Submitting Work Report**<br>If in step 4 of the normal flow there was an error with the submission of the Co-op Evaluation, then<br>1. The system shall display an error message, saying there was a problem with Co-op Evaluation submission.<br>2. The use case ends with a failure condition. |

### 4.3.2 Save a Co-op Evaluation

| | |
|---|---|
| **Actors** | Employer |
| **Description** | This use case describes how the Employer uses the Co-op Evaluation System to save a Co-op Evaluation to be completed at a later date. |
| **Trigger** | Employer has a Co-op Student for which a Co-op Evaluation needs to be submitted, and they wish to finish the Co-op Evaluation at a later date. |
| **Pre-conditions** | 1. The Employer has an active Internet connection.<br>2. The Employer has a registered Co-op Student. |
| **Post-conditions** | A Co-op Evaluation has been saved for the respective Co-op Student. |
| **Normal Flow** | 1. The Employer locates the panel for the student in question under “Current Co-op Students”.<br>2. The Employer selects the “Open” link next “Employer Eval” in the given student co-op panel. |

| | |
|---|---|
| | 3. The Employer partially completes the Co-op Evaluation.<br>4. The Employer saves the Co-op Evaluation.<br>5. The system displays a message saying that the Co-op Evaluation has been saved successfully.<br>6. The use case ends successfully. |
| **Alternative Flows** | N/A |
| **Exceptions** | **Invalid User**<br>If the use case Authenticate Employer does not complete successfully, then<br>1. The use case ends with a failure condition.<br><br>**Error Saving Work Report**<br>If in step 4 of the normal flow there was an error with the saving of the Co-op Evaluation, then<br>1. The system shall display an error message, saying there was a problem with saving the Co-op Evaluation.<br>2. The use case ends with a failure condition. |

### 4.3.3 Edit & Submit a Co-op Evaluation

| | |
|---|---|
| **Actors** | Employer |
| **Description** | This use case describes how the Employer uses the Co-op Evaluation System to update a Co-op Evaluation that was previously saved. |
| **Trigger** | Employer has a co-op for which a Co-op Evaluation has been started, and they wish to finish the Co-op Evaluation and submit it. |
| **Pre-conditions** | 1. The Employer has an active Internet connection.<br>2. The Employer has previously registered a co-op.<br>3. The Employer has a previously saved a Co-op Evaluation. |
| **Post-conditions** | A Co-op Evaluation has been saved for the respective Co-op Student. |
| **Normal Flow** | 1. The Employer locates the panel for the student in question under “Current Co-op Students”.<br>2. The Employer selects the “Open” link next “Employer Eval” in the given student co-op panel.<br>3. The Employer completes the Co-op Evaluation, if unfinished.<br>4. The Employer submits the Co-op Evaluation.<br>5. The system displays a message saying that the Co-op Evaluation has been submitted successfully.<br>6. The use case ends successfully. |
| **Alternative Flows** | **Save Work Report**<br>If in step 4 of the normal flow the Employer chooses to save the Co-op Evaluation again instead of submitting it, then |

|  |  |
|---|---|
|  | 1. The Employer partially completes the Co-op Evaluation.<br>2. The Employer saves the Co-op Evaluation.<br>3. The system displays a message saying that the Co-op Evaluation has been saved successfully.<br>4. The use case ends successfully.<br><br>**All Required Fields Have Not Been Completed**<br>If in step 5 of the normal flow the Employer submits the Co-op Evaluation without having completed all required fields, then<br>1. The system shall display an error message, and ask the Employer to fill out all required fields that were not completed.<br>2. The use case resumes at step 4. |
| **Exceptions** | **Invalid User**<br>If the use case Authenticate Employer does not complete successfully, then<br>1. The use case ends with a failure condition.<br><br>**Error Submitting Co-op Evaluation**<br>If in step 4 of the normal flow there was an error with the submitting of the Co-op Evaluation, then<br>1. The system shall display an error message, saying there was a problem with submitting the Co-op Evaluation.<br>2. The use case ends with a failure condition. |

### 4.3.4 View a Submitted Co-op Evaluation

|  |  |
|---|---|
| **Actors** | Employer |
| **Description** | This use case describes how the Employer uses the Co-op Evaluation System to view a Co-op Evaluation that was previously submitted. |
| **Trigger** | Employer has a Co-op Evaluation that has been submitted, and they want to view. |
| **Pre-conditions** | 1. The Employer has an active Internet connection.<br>2. The Employer has a previously submitted Co-op Evaluation for a given student. |
| **Post-conditions** | The desired Co-op Evaluation is displayed to the Employer. |
| **Normal Flow** | 1. The Employer locates the panel for the student in question under “Current Co-op Students”.<br>2. The Employer selects the “Open” link next “Employer Eval” in the given student co-op panel.<br>3. The Employer selects a previously submitted Co-op Evaluation for the student.<br>4. The system displays the Co-op Evaluation to the Employer. |

| | |
|---|---|
| | 5. The use case ends successfully. |
| **Alternative Flows** | N/A |
| **Exceptions** | **Invalid User**<br>If the use case Authenticate Employer does not complete successfully, then<br>1. The use case ends with a failure condition.<br><br>**Error Displaying Work Report**<br>If in step 4 of the normal flow there was an error displaying the Co-op Evaluation, then<br>1. The system shall display an error message, saying there was a problem with displaying the Co-op Evaluation.<br>2. The use case ends with a failure condition. |

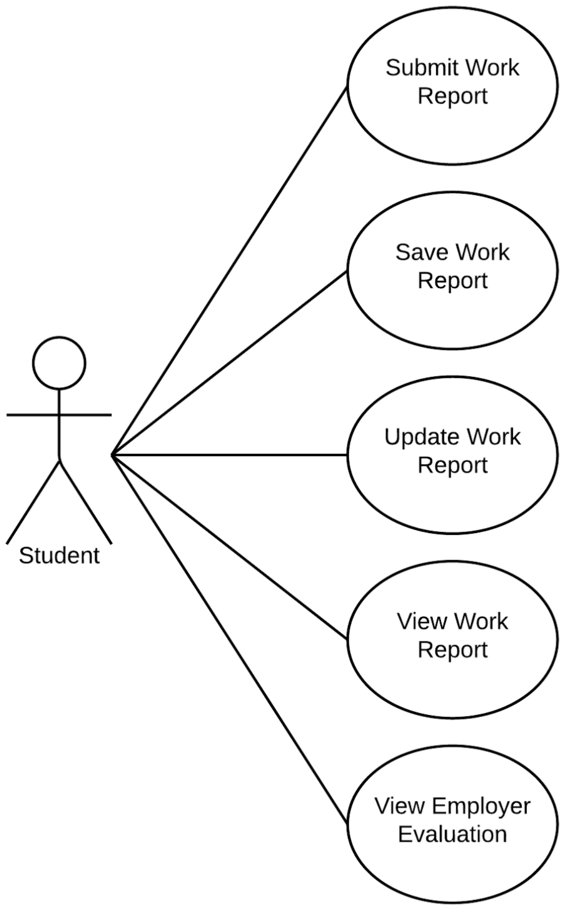
### 4.3.5 Authenticate Employer

This use case is meant to demonstrate what will happen if the employers cannot authenticate through Simplicity. It may or may not be the case in the final system, but it is important to plan for possibility at this stage in the process.

| | |
|---|---|
| **Actors** | Employer |
| **Description** | This use case describes how the Employer authenticates with the Co-op Evaluation System. |
| **Trigger** | Employer has a action to complete on the Co-op Evaluation System. |
| **Pre-conditions** | 1. The Employer has an active Internet connection.<br>2. The Employer has hired at least one student to work a co-op. |
| **Post-conditions** | The Employer is successfully logged into the Co-op Evaluation System. |
| **Normal Flow** | 1. The Employer opens the website on a supported browser.<br>2. The Employer selects “Employer Login”.<br>3. The Employer types in their user name.<br>4. The Employer types in their password.<br>5. The Employer submits their credentials.<br>6. The system authenticates the user.<br>7. The system displays the Employer’s homepage.<br>8. The use case ends successfully. |
| **Alternative Flows** | N/A |
| **Exceptions** | **Invalid User**<br>If in step 7 of the normal flow does not complete successfully, then<br>1. The use case ends with a failure condition. |

## 4.4 Student

### 4.4.1 Use Case Context



### 4.4.2 Submit Work Report

| | |
|---|---|
| **Actors** | Student |
| **Description** | This use case describes how the Student uses the Co-op Evaluation System to submit a new Work Report. |
| **Trigger** | Student has a co-op for which a Work Report needs to be submitted. |
| **Pre-conditions** | 1. The Student has an active Internet connection.<br>2. The Student has previously registered a co-op. |
| **Post-conditions** | A Work Report has been submitted for the respective Student's co-op. |
| **Normal Flow** | 1. The Student locates the panel for the co-op in question. |

| | |
|---|---|
| | 2. The Student selects the “Open” link next “Work Report” in the given co-op panel.<br>3. The Student completes the Work Report.<br>4. The Student submits the Work Report.<br>5. The system displays a message saying that the Work Report has been submitted successfully.<br>6. The use case ends successfully. |
| **Alternative Flows** | **All Required Fields Have Not Been Completed**<br>If in step 4 of the normal flow the Student submits the Work Report without having completed all required fields, then<br>1. The system shall display an error message, and ask the Student to fill out all required fields that have not been completed.<br>2. The use case resumes at step 3. |
| **Exceptions** | **Error Submitting Work Report**<br>If in step 4 of the normal flow there was an error with the submission of the Work Report, then<br>1. The system shall display an error message, saying there was a problem with Work Report submission.<br>2. The use case ends with a failure condition. |

### 4.4.3 Save Work Report

| | |
|---|---|
| **Actors** | Student |
| **Description** | This use case describes how the Student uses the Co-op Evaluation System to save a Work Report to be completed at a later date. |
| **Trigger** | Student has a co-op for which a Work Report needs to be submitted, and they wish to finish the Work Report at a later date. |
| **Pre-conditions** | 1. The Student has an active Internet connection.<br>2. The Student has previously registered a co-op. |
| **Post-conditions** | A Work Report has been saved for the respective Student’s co-op. |
| **Normal Flow** | 1. The Student locates the panel for the co-op in question.<br>2. The Student selects the “Open” link next “Work Report” in the given co-op panel.<br>3. The Student partially completes the Work Report.<br>4. The Student saves the Work Report.<br>5. The system displays a message saying that the Work Report has been saved successfully.<br>6. The use case ends successfully. |
| **Alternative Flows** | N/A |

| | |
|---|---|
| **Exceptions** | **Error Saving Work Report**<br>If in step 4 of the normal flow there was an error with the saving of the Work Report, then<br>1. The system shall display an error message, saying there was a problem with saving the Work Report.<br>2. The use case ends with a failure condition. |

### 4.4.4 Update Work Report

| | |
|---|---|
| **Actors** | Student |
| **Description** | This use case describes how the Student uses the Co-op Evaluation System to update a Work Report that was previously saved. |
| **Trigger** | Student has a co-op for which a Work Report has been started, and they wish to finish the Work Report and submit it. |
| **Pre-conditions** | 1. The Student has an active Internet connection.<br>2. The Student has previously registered a co-op.<br>3. The Student has a previously saved a Work Report. |
| **Post-conditions** | A Work Report has been saved for the respective Student’s co-op. |
| **Normal Flow** | 1. The Student locates the panel for the co-op in question.<br>2. The Student selects the “Save MM/DD/YYYY” link next “Work Report” in the given co-op panel.<br>3. The Student completes the Work Report.<br>4. The Student submits the Work Report.<br>5. The system displays a message saying that the Work Report has been submitted successfully.<br>6. The use case ends successfully. |
| **Alternative Flows** | **Save Work Report**<br>If in step 3 of the normal flow the Student chooses to save the Work Report again instead of submitting it, then<br>1. The Student partially completes the Work Report.<br>2. The Student saves the work report.<br>3. The system displays a message saying that the Work Report has been saved successfully.<br>4. The use case ends successfully.<br><br>**All Required Fields Have Not Been Completed**<br>If in step 4 of the normal flow the Student submits the Work Report without having completed all required fields, then<br>1. The system shall display an error message, and ask the Student to fill out all required fields that have not been completed. |

|  |  |
|---|---|
|  | 2. The use case resumes at step 3. |
| **Exceptions** | **Error Submitting Work Report**<br>If in step 6 of the normal flow there was an error with the submitting of the Work Report, then<br>1. The system shall display an error message, saying there was a problem with submitting the Work Report.<br>2. The use case ends with a failure condition. |

### 4.4.5 View Work Report

|  |  |
|---|---|
| **Actors** | Student |
| **Description** | This use case describes how the Student uses the Co-op Evaluation System to view a Work Report that was previously submitted. |
| **Trigger** | Student has a Work Report that has been submitted, and they want to view. |
| **Pre-conditions** | 1. The Student has an active Internet connection.<br>2. The Student has previously submitted a Work Report. |
| **Post-conditions** | The desired Work Report is displayed to the Student. |
| **Normal Flow** | 1. The Student locates the panel for the co-op in question.<br>2. The Student selects the "Submitted MM/DD/YYYY" link next "Work Report" in the given co-op panel.<br>3. The system displays the Work Report to the Student.<br>4. The use case ends successfully. |
| **Alternative Flows** | N/A |
| **Exceptions** | **Error Displaying Work Report**<br>If in step 3 of the normal flow there was an error displaying the Work Report, then<br>1. The system shall display an error message, saying there was a problem with displaying the Work Report.<br>2. The use case ends with a failure condition. |

### 4.4.6 View Employer Evaluation

|  |  |
|---|---|
| **Actors** | Student |
| **Description** | This use case describes how the Student uses the Co-op Evaluation System to view an Employer Evaluation that was previously submitted. |

| | |
|---|---|
| **Trigger** | Student has an Employer Evaluation that has been submitted, and they want to view. |
| **Pre-conditions** | 1. The Student has an active Internet connection.<br>2. The Student has a previously submitted Employer Evaluation. |
| **Post-conditions** | The desired Employer Evaluation is displayed to the Student. |
| **Normal Flow** | 1. The Student locates the panel for the co-op in question.<br>2. The Student selects the “Submitted MM/DD/YYYY” link next “Employer Eval” in the given co-op panel.<br>3. The system displays the Employer Evaluation to the Student.<br>4. The use case ends successfully. |
| **Alternative Flows** | N/A |
| **Exceptions** | **Error Displaying Work Report**<br>If in step 3 of the normal flow there was an error displaying the Employer Evaluation, then<br>1. The system shall display an error message, saying there was a problem with displaying the Employer Evaluation.<br>2. The use case ends with a failure condition. |

# 5 External Interface Requirements

## 5.1 User Interfaces

The system’s user interfaces will need to look and feel like they belong to RIT. This means that they should have a similar layout, color scheme, and other visual aspects as other school sites. However, the system should also utilize modern web design standards of layout and interaction.

Since the system is a web application, the user interface must be consistent across all modern desktop browsers, specifically Chrome, Firefox, Safari, and Internet Explorer 9+. Additionally, the system will take advantage of Twitter Bootstrap to create a responsive user interface, thus allowing users to access the system on a variety of devices. This in-turn requires that the system be accessible and usable on mobile platforms (e.g. smart phones and tablets). If a feature is not accessible from a mobile device the user interface must display an error message that clearly explains what cannot be done and why.

## 5.2 Software Interfaces

The product will interact with Shibboleth in order to authenticate users. Our system will take in log-in credentials from the user, and send those credentials to Shibboleth to verify. Shibboleth will then return the type of user, which will be used by our program to give the user specific permissions.

## 5.3 Communications Interfaces

Submission of form data must persist to the database. The website must inform the the user that their data was not properly stored in the database, and allow for resubmission. Hibernate and JPA will be used for database access from our system.

The system must interface with ITS mail servers in order to send out notification emails to students and employers.

The system will communicate over the standard protocols of HTTP and TCP/IDP. For accessing secure sections of the system, HTTPS will be used. Shibboleth will be used to authenticate users.

# 6 Non-functional Requirements

## 6.1 Performance Requirements

| Number | Priority | Requirement |
|---|---|---|
| PR-01 | High | The system shall respond to requests within 3 seconds. |
| PR-02 | Medium | The system shall be able to handle up to 6,000 users at a time. |
| PR-03 | Medium | The system shall load all web pages in less than 30 seconds, with the exception of reports. |

## 6.2 Security Requirements

| Number | Priority | Requirement |
|---|---|---|
| SE-01 | High | The system shall require users to authenticate against Shibboleth with their RIT user IDs in order to access the system. |
| SE-02 | High | The system shall not allow employers to view evaluations for a student that is not currently working for him/her. |

## 6.3 Safety Requirements

| Number | Priority | Requirement |
|---|---|---|
| SA-01 | High | The system shall ensure integrity of its data upon database failure. |
| SA-02 | High | Inputted user data shall not be lost. |

## 6.4 Usability Requirements

| Number | Priority | Requirement |
|---|---|---|
| UR-01 | Medium | The system shall be usable from all major up-to-date web browsers (latest version of Chrome, latest version of Firefox, latest version of Safari, and IE 9+). |
| UR-02 | Low | The system shall be accessible from mobile devices. |

## 6.5 Availability Requirements

| Number | Priority | Requirement |
|---|---|---|
| AR-01 | High | The system shall be available 95% of the time. |

| | | |
|---|---|---|
| AR-02 | High | The system shall not time out during the evaluation submission more than 5% of the time. |

## 6.6 Modifiability Requirements

| Number | Priority | Requirement |
|---|---|---|
| MR-01 | High | The system shall be modifiable for future updates. |
| MR-02 | High | The system shall allow new colleges to be added to the system. |
| MR-03 | High | The system shall allow new departments to be added to the system. |

## 6.7 Documentation Requirements

| Number | Priority | Requirement |
|---|---|---|
| DR-01 | Medium | The system shall include a test plan, including up to 95% code coverage. |
| DR-02 | High | The system shall include a user manual that describes to users how to use the system. |

## 6.8 Environment Requirements

| Number | Priority | Requirement |
|---|---|---|
| EN-01 | High | The system shall run on the ITS-provided VM at final release. |

## Appendix A: Glossary

| Term | Definition |
|---|---|
| CES | Co-op Evaluation System |
| Evaluation | A form that is currently being filled out by a student or by an employer |
| EWA | Enterprise Web Applications, a division of ITS |
| Form | A template that is used to generate an evaluation for a department |
| ITS | Information and Technology Services |
| Notification | An email message that will be generated and sent to students and/or employees. |
| OCSCE | Office of Cooperative Education and Career Services |
| Report | An aggregation of submissions used to display statistics |
| RIT | Rochester Institute of Technology |
| SRS | Software Requirements Specification |
| Submission | A form that has been completed and submitted to the evaluator |
| Status | The current state of the evaluation |