Self and Peer Assessment System

Self and Peer Assessment system is an online tool and it is primarily designed to support evaluation activities where students work in teams.

Software Engineering Individual project. Dr. Qingxiong Ma

**Abid A. Ahmadi**

**November 29 2017**

**Table of Contents**

1. Scope of project -----------------------------------------------------------------------1
   1. Statement problem ------------------------------------------------------------1
2. Functional requirements -------------------------------------------------------------2
   1. Domain classes ------------------------------------------------------------------2
3. Domain modeling ----------------------------------------------------------------------3
4. Use case modeling ---------------------------------------------------------------------4
   1. List of use cases -----------------------------------------------------------------4
   2. Use case diagram: Faculty and Student ------------------------------------5
   3. Use case diagram: Admin -----------------------------------------------------6
   4. Use case description, web mock-ups, and UI prototype----------------7
      1. User login ----------------------------------------------------------------7
      2. User login-----------------------------------------------------------------8
      3. Student Evaluate Peer description----------------------------------9
      4. Student Evaluate Peer screen mockups----------------------------9
      5. Student Evaluate Peer UI prototype ------------------------------10
      6. Student View Peer Assessment description ---------------------11
      7. Student View Peer Assessment screen mock-ups --------------11
      8. Student View Peer Assessment UI prototype -------------------12
      9. Student Make Comment on Evaluation description------------13
      10. Student Make Comment on Evaluation web page screen mock-ups---------------------------------------------------------------13
      11. Student Make Comment on Evaluation UI prototype ----------14
      12. Faculty Create Team description----- -----------------------------15
      13. Faculty Create Team web screen mock ups ---------------------15
      14. Faculty Create Team UI prototype --------------------------------16
      15. Admin Add Student use case description ------------------------17
      16. Admin Add Student web screen mock ups ----------------------17
      17. Admin Add Student UI prototype ---------------------------------18
      18. Admin Add Faculty use case description-------------------------19
      19. Admin Add Faculty web screen mock ups-----------------------19
      20. Admin Add Faculty UI prototype ----------------------------------20
      21. Admin Add Class use case description ----------------------------21
      22. Admin Add Class UI prototype--------------------------------------22
      23. Admin Add Rubrics use case description-------------------------23
      24. Admin Add Rubrics web screen mock ups-----------------------24

A. Ahmadi

November 29 2017

1. Analysis Class Diagram --------------------------------------------------------------25
   1. Available classes --------------------------------------------------------------25
   2. Class diagram ------------------------------------------------------------------26
2. Robustness Diagram -----------------------------------------------------------------27
   1. User login -----------------------------------------------------------------------27
   2. Student Evaluate Peer -------------------------------------------------------28
   3. Student View Peer Assessment---------------------------------------------29
   4. Student Make Comment on Evaluation ----------------------------------30
   5. Faculty Create Team ---------------------------------------------------------31
   6. Admin Add Student ----------------------------------------------------------32
   7. Admin Add Faculty -----------------------------------------------------------33
   8. Admin Add Class --------------------------------------------------------------34
   9. Admin Add Class---------------------------------------------------------------35
   10. Admin Create Rubrics -------------------------------------------------------36

A. Ahmadi

November 29 2017

**Scope of Project**

Self and peer assessment system is an online tool and it is primarily designed to support evaluation activities where students work in groups or teams. The purpose of this online tool is to collect group peer evaluation data, do analysis, and generate reports. Self and peer assessment system allows academics to easily set up and manage the process of obtaining and collating self and peer assessment reports. The Self and Peer Assessment online tool is designed to generate a fully personalized feedback to inspire and improve team member’s skills.

**Problem Statement**

The mentioned scope of the project derived from the following problem. Group project helps IT students apply system design knowledge, solve real world business problems, enhance learning effective team skills, improve students’ business communication skills, and serve back to the community. Therefore, students are required to do group-projects in their classes.

During the semester, students are asked to do multiple self- and peer- assessments based on the provided rubrics. The rubrics for the self- and peer assessment were developed based on previous studies and department faculty brainstorming. The primary focus of self- and peer- assessment is on the learning process of team skills. It should be used to enhancing team skill development. It can also be used as a reference or supplementary documents when faculty evaluates the final project.

The current manual process of administering and analysing the assessment is time-consuming for faculty members to implement the assessment. The paper-based assessment is difficult for students to get an instant feedback from previous stage of project to improve their group performance. This problem caused the establishment of an online self and peer assessment system.

**Function Requirements**

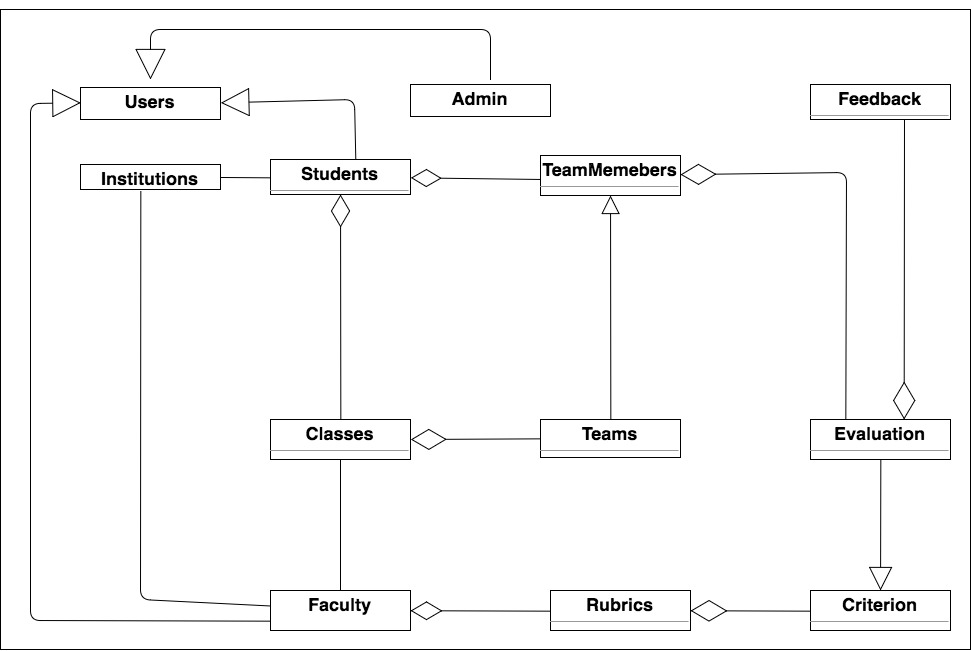
* 1. The system should allow faculty, students, and administrator to log in by selecting their role.
  2. The system must allow admin to add student, delete student, and edit student information
  3. The system must allow admin to view students and faculty by school
  4. They system must allow admin to add faculty, delete faculty, and edit faculty information
  5. The system must allow admin to view classes by school and should also allow admin to search classes by class number.
  6. The system must allow admin to create, delete, and modify classes.
  7. The system must allow admin to create rubrics.
  8. They system must allow admin to modify rubric’s criteria.
  9. The system must allow admin to add, edit and delete rubrics.
  10. The system must allow students to perform evaluation on team members.
  11. The system must allow students to view peer evaluation in order to get instant feedback but the peer evaluation should be displayed anonymously.
  12. The system must allow faculty to create project team and assign enrolled students in a team.
  13. The system must allow students to make comment on peer’s assessment.
  14. The web system must analyze the evaluation and generate different reports for individual student, team and class.
  15. The system must allow any department and institution to use the system.

**Domain Classes**

1. Institution
2. User
3. Admin
4. Faculty
5. Student
6. Team
7. Team Members
8. Classes
9. Rubrics
10. Criteria
11. Feedback
12. Evaluation

**DOMAIN MODELING**

Domain Modeling for 12 Identified Domain Classes



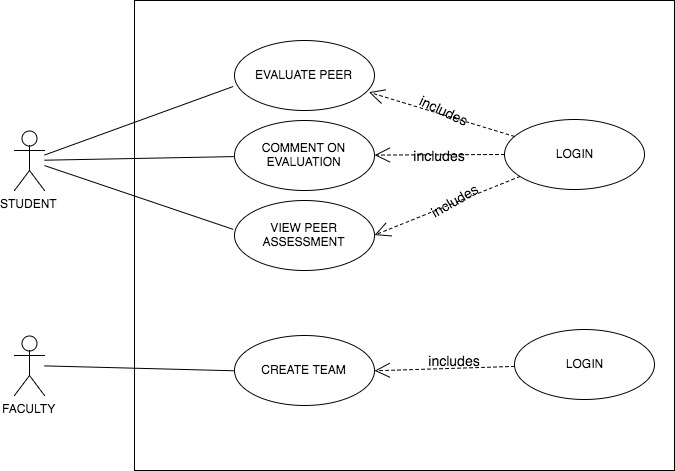
**USE CASE MODELING:** USE CASE LIST

**Use Case List Self and Peer Evaluation System**

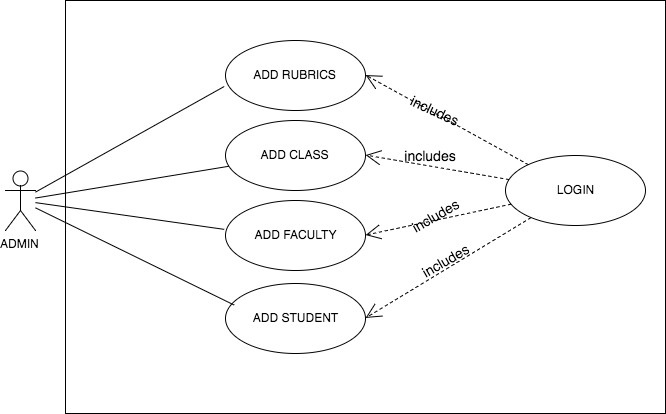
|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Actor** | **Use Case** | **Goal** |
| 1 | Admin, Student, Faculty | User Login | Allows user to login by role |
| 2 | Student | View Peer Assessment | Student can get instant feedback and make a comment on peer assessments |
| 3 | Student | Evaluate Peer | Allow student to evaluate team members |
| 4 | Student | Make Comment on Evaluation | Student can comment on peer assessment if student disagrees |
| 5 | Faculty | Create Team | Enables faculty to create project team |
| 6 | Admin | Add Student | Enables admin to add new student |
| 7 | Admin | Add Faculty | Enables admin to add new faculty |
| 8 | Admin | Add Class | Enables admin to add new class |
| 9 | Admin | Add Rubrics | Allows admin to create new rubrics for faculties |

**USE CASE MODELING:** USE CASE DIAGRAM FOR FACULTY AND STUDETN

**Use Case Diagram for Actors: Student and Faculty**

**USE CASE MODELING:** USE CASE DIAGRAM FOR ADMIN

**Use Case Diagram for Actors: ADMIN**



**USE CASE MODELING:** USE CASE DESCRIPTION, WEB PAGE SCREEN MOCK-UPS & UI PROTOTYPE

**Use case 1: User Login:**

Actor: Admin, Faculty, or Student

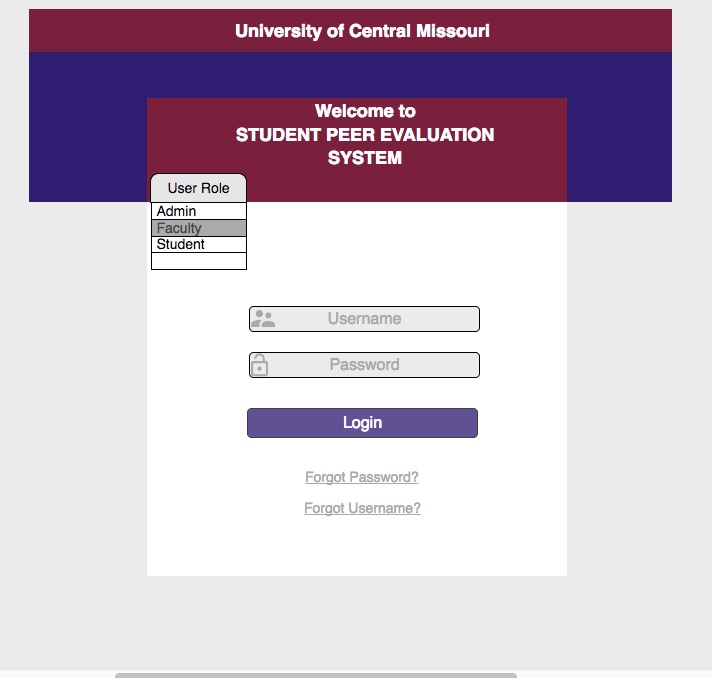
**Basic Course**

User enters the self and peer evaluation URL into the browser. The system opens self and peer evaluation login page. User clicks on the “Select Role” tab. The system opens a drop down menu. User selects her/his role from the drop down menu. The system closes the drop down menu. User types in her/his password and username then clicks on the login button. The system checks to see if the username exists in the “Users”. If an account is found, the system compares the password in the system database. If comparison is successful, the system retrieves the user information, based on the user role. The system displays the user’s landing page.

**Alternate Course**

Invalid username or password: The system checks to see if the username exists in the “Users”. If the username is not found, the system displays a message “Invalid username. Click on “OK” to try again.” The user clicks on the “OK” button. The system redisplays login page. User clicks on the forgot username link. The system asks user to enter his/her email address. User enters the email address associated with his/her account. The system displays a message ”Username was sent to your email.”

**Use case mock-up web page: (****Use case: User Login)**

****

**USE CASE MODELING:** USE CASE DESCRIPTION, WEB PAGE SCREEN MOCK-UPS & UI PROTOTYPE

**Use case 2: Student Evaluate Peer:**

Actor: Student

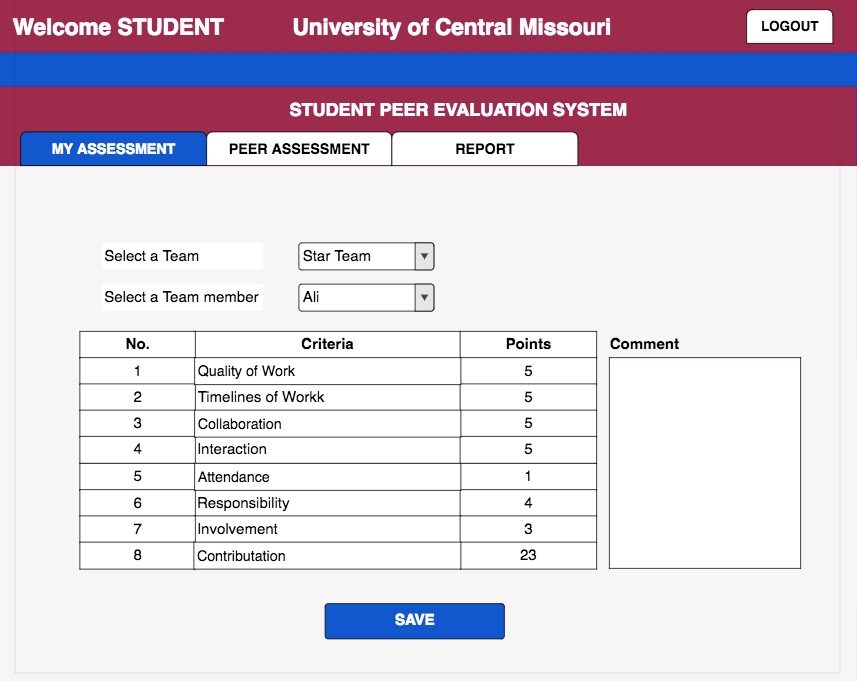
**Basic: Course:**

Student clicks on the “Evaluate Peer” button (from My Assessment page). The system displays list of teams in a drop box. Student selects a team. The system displays list of team members in a drop box. Student selects a team member. The system displays rubric criteria. Student performs the assessment by typing a point to each criterion. Student can optionally write a comment. Student clicks on the “Save” button. The system ensures if the points are within allowable range. If the entered points are within allowable range, the system saves the assessment and redisplays view my assessment (Student landing page) page.

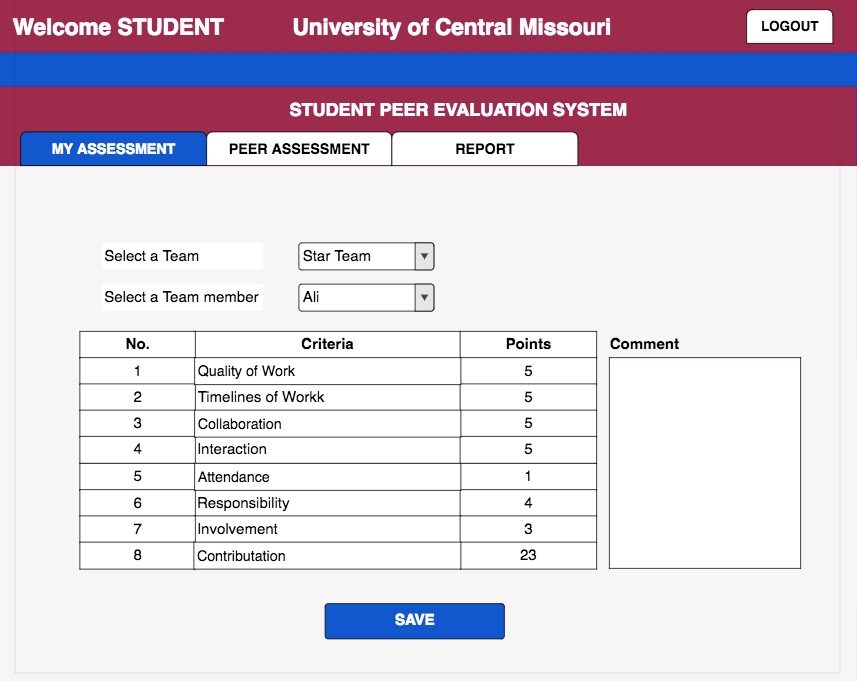
**Alternate Course**

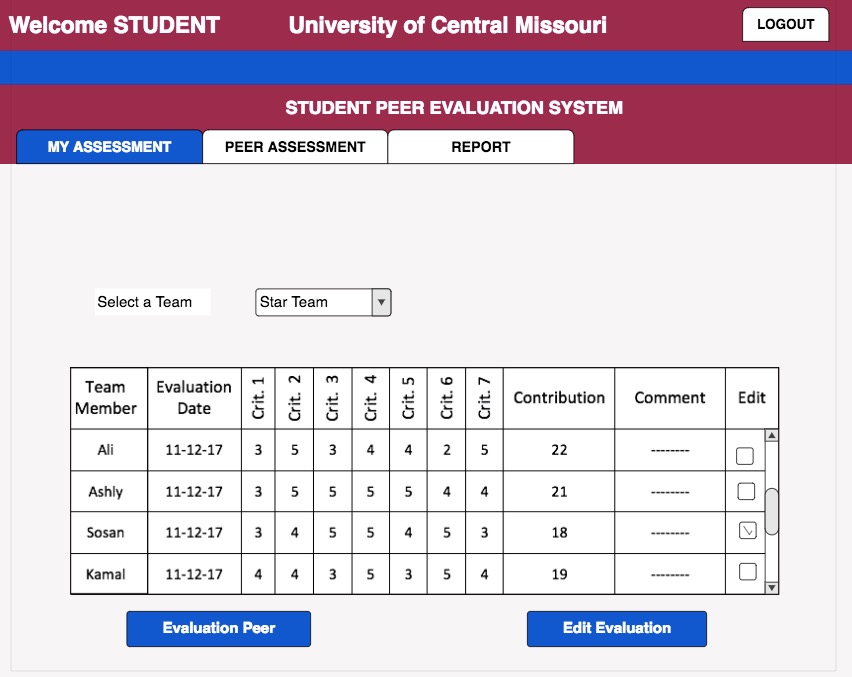
Grading point out of range: The system prompts “Grade not within allowable range (1-5). Click on ”Ok” to try again or on “Cancel” to return to student landing page.

**Use case mock-up web page: (Use case2: Student Evaluate Peer)**

**Student Evaluate Peer UI Prototype**

Student Landing Page

**USE CASE MODELING:** USE CASE DESCRIPTION, WEB PAGE SCREEN MOCK-UPS & UI PROTOTYPE



Evaluate Peer

**Use case 3: Student View Peer Assessment:**

Actor: Student

**Basic Course:**

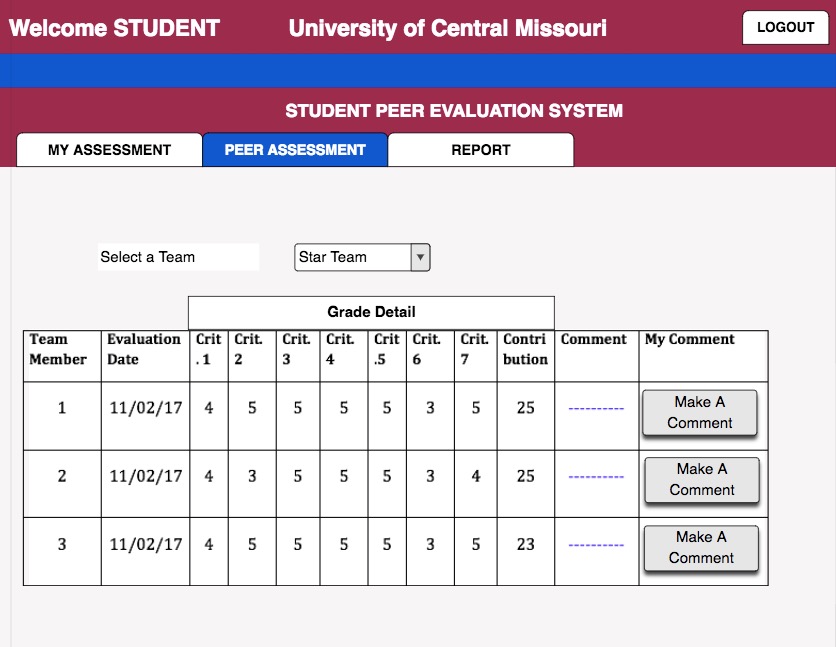
Student clicks on the “Peer Assessment” button (from student landing page).

The system displays the teams the student is assigned to. Student selects a team. The system displays assessments performed by team members.

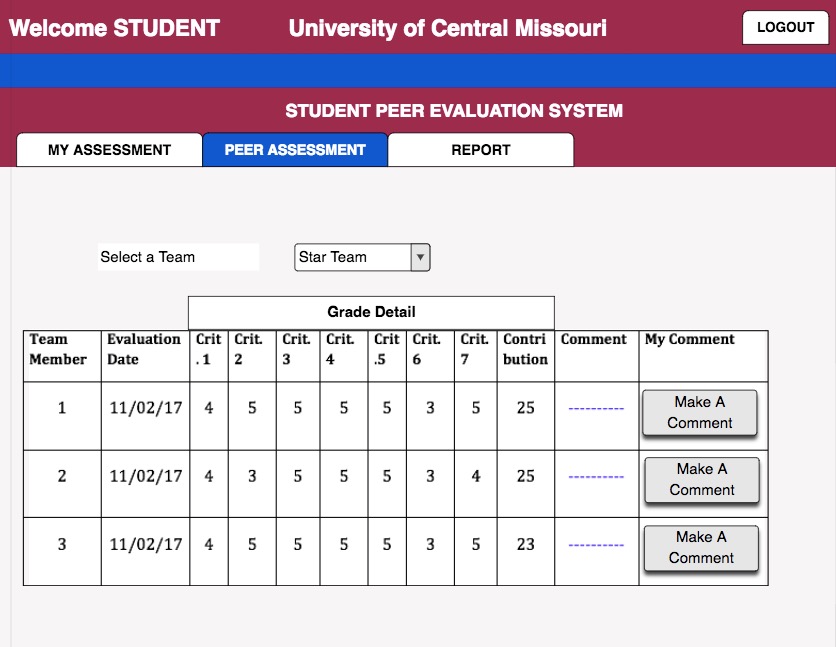
**Alternate Course**

Assessment not performed yet: If assessment does not exist, the system displays a message “Team members have not performed evaluation yet. Please come back at a later time. Click on “OK” to select another team or on “Cancel” to return to student landing page.”

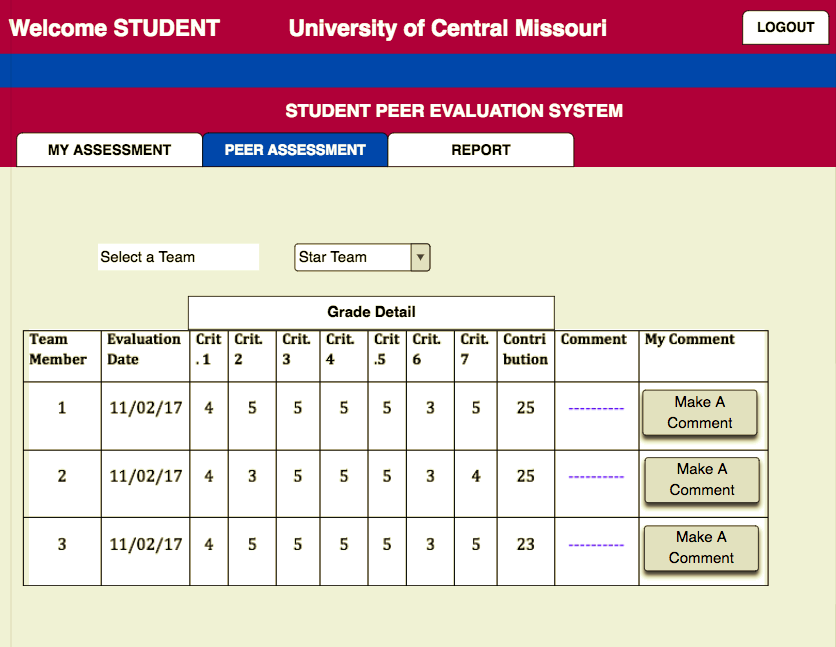
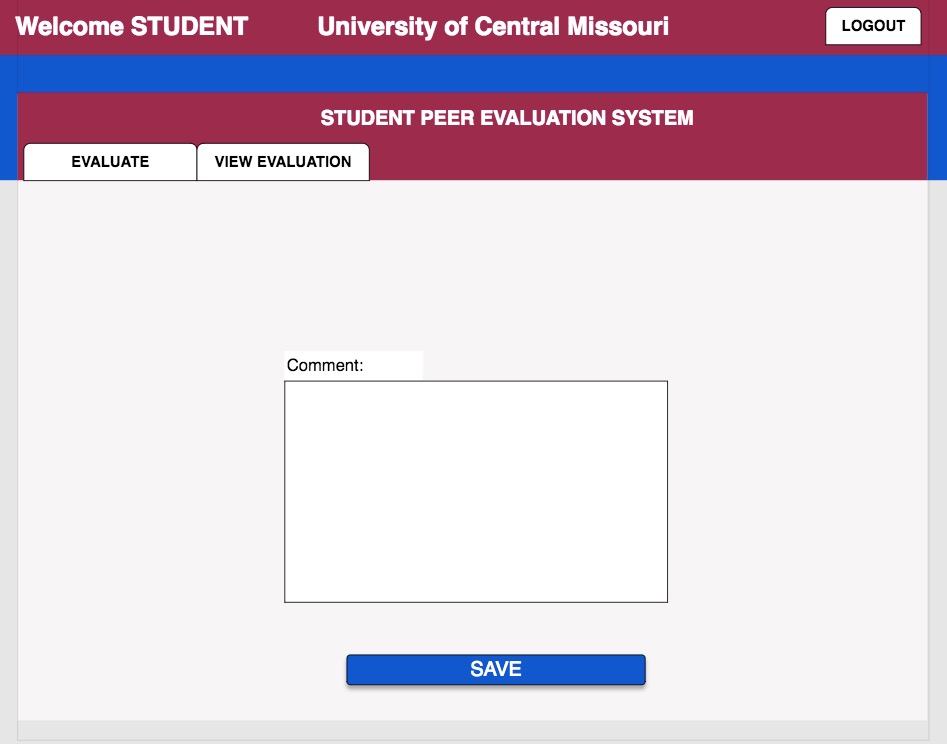
**Use case mock-up web page: (Use case: Student View Peer Assessment)**

**Student View Peer Assessment – UI Prototype**

Student View Peer Assessment Page

****

Student Landing Page



**USE CASE MODELING:** USE CASE DESCRIPTION, WEB PAGE SCREEN MOCK-UPS & UI PROTOTYPE

**Use case 4: Student Make Comment on Evaluation:**

Actor: Student

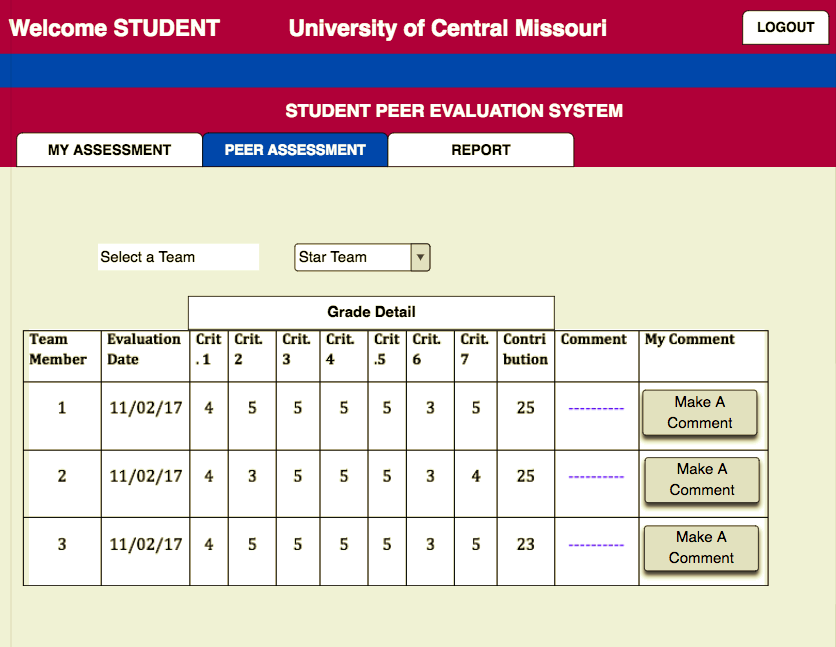
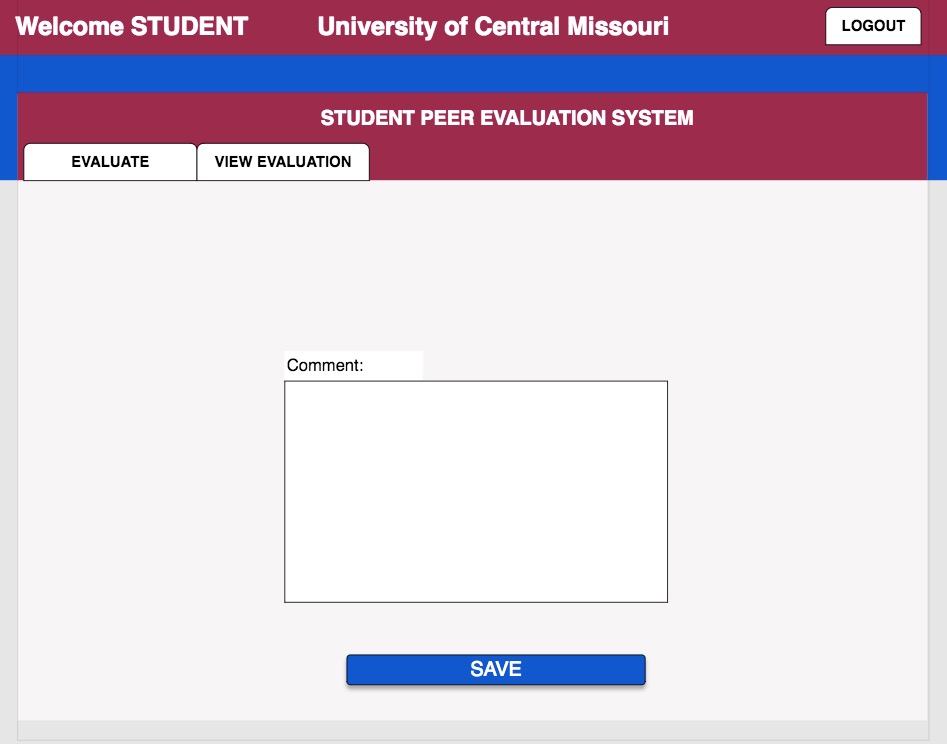
**Basic Course:**

Student clicks on the “Make A Comment” button (from view peer assessment page). The system displays a comment window associated to the desired team member. Student writes a comment on the comment box. Student clicks on the “Save” button. The system ensures the comment length is okay. If the comment length is within allowable range, the system saves the comment and redisplays the “Peer Assessment” page.

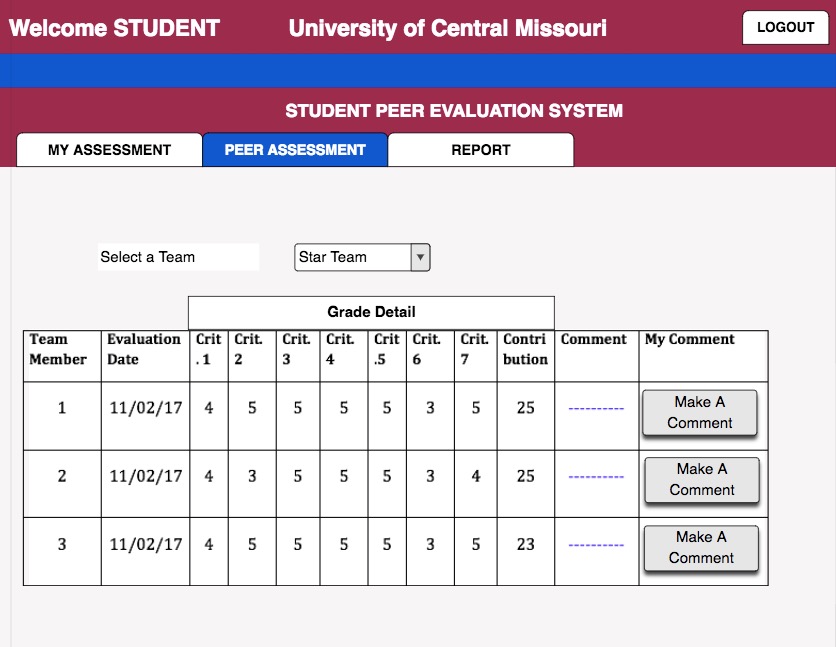
**Alternate Course**

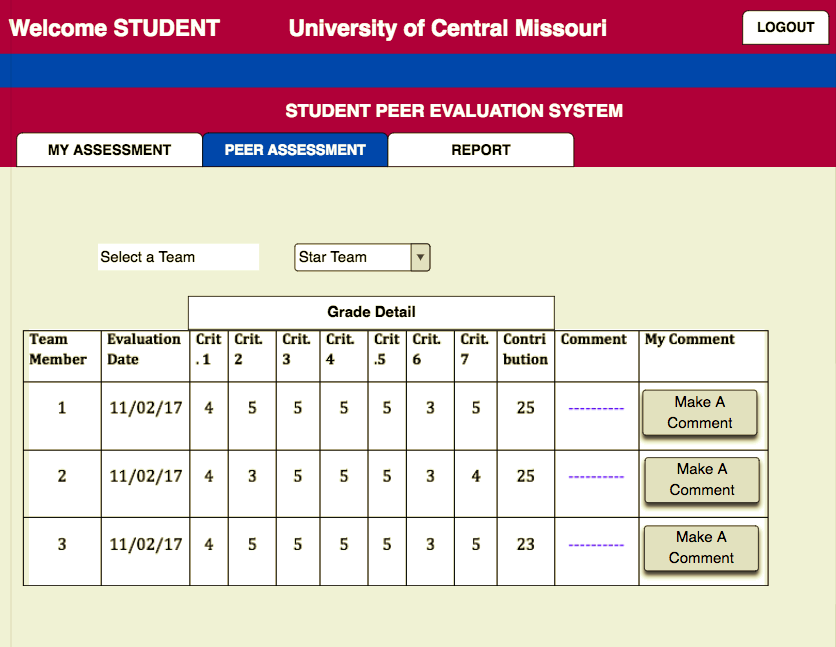
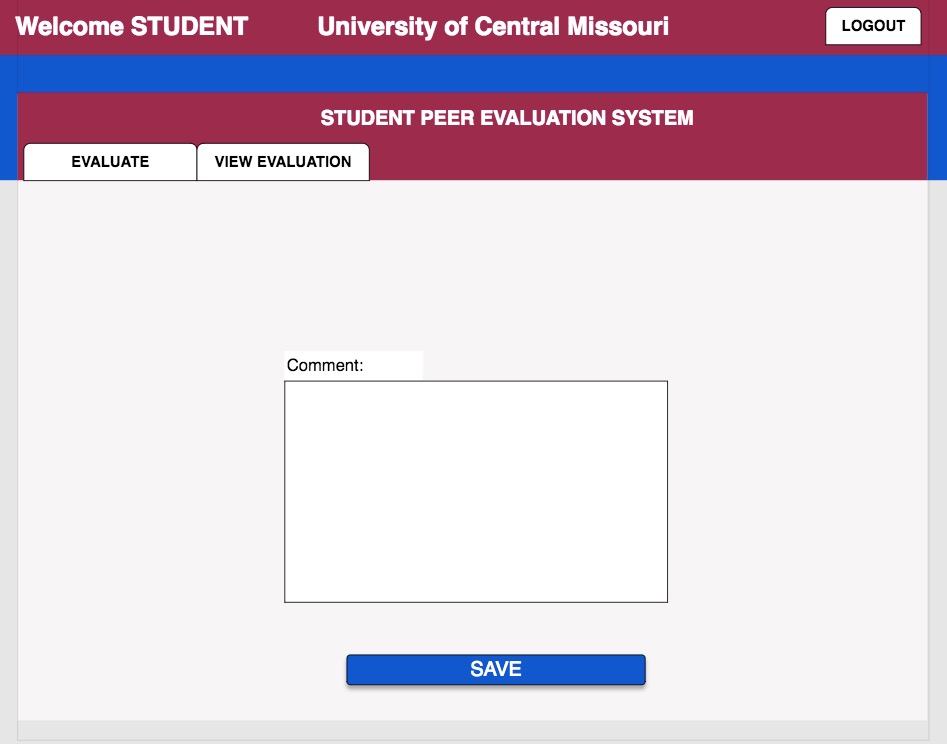
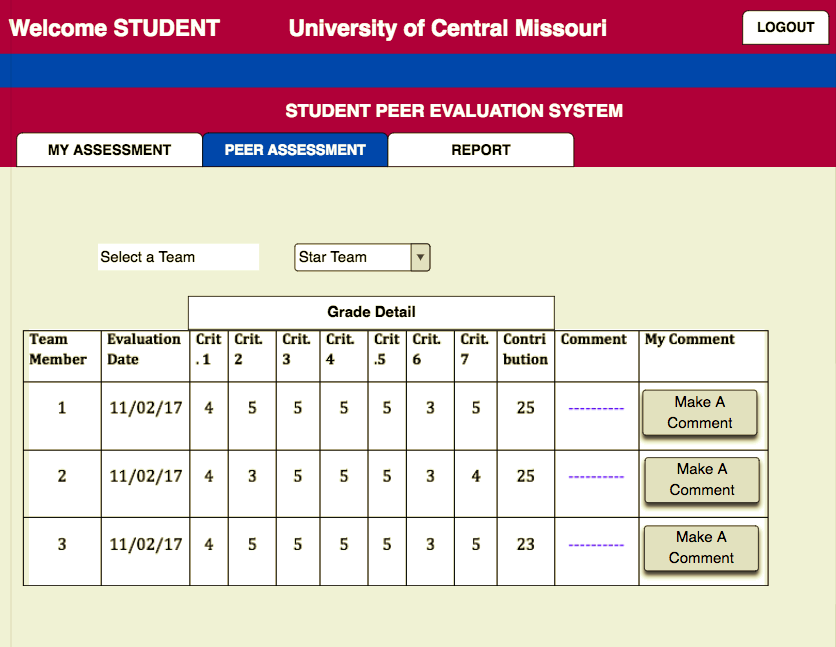
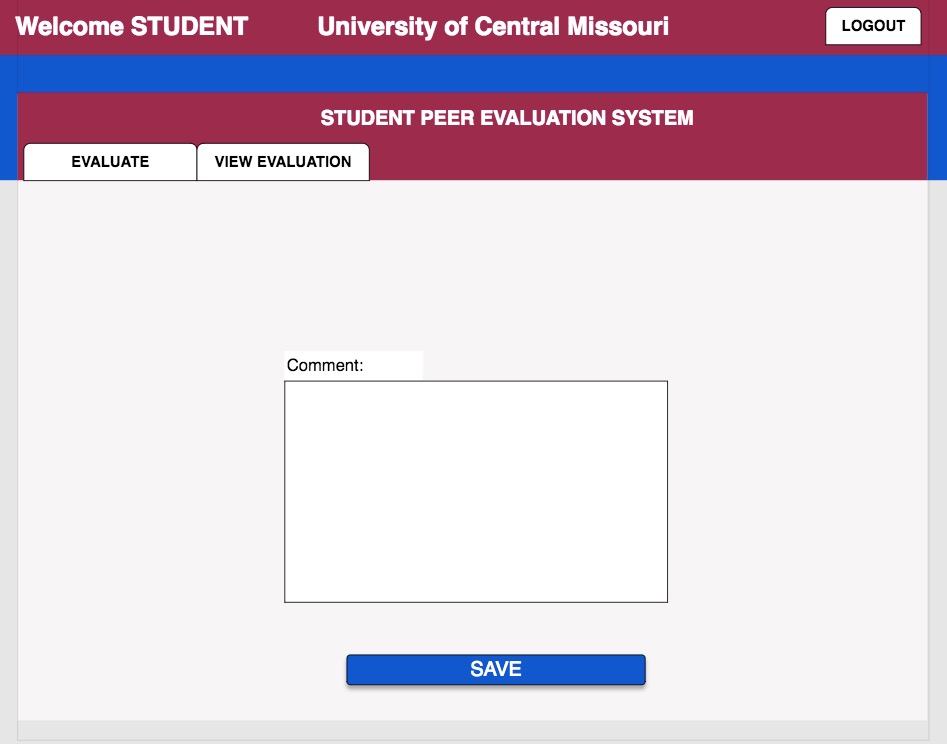
Too long or short comment: The system displays a message “Comment length is out side the allowable range. Click on “OK” to try again.”

**Use case mock-up web page: (Use case : Student Make Comment on Evaluation)**



**Student Comment On Evaluation – UI Prototype**

**USE CASE MODELING:** USE CASE DESCRIPTION, WEB PAGE SCREEN MOCK-UPS & UI PROTOTYPE



**Use case 5: Faculty Create Team:**

Actor: Faculty

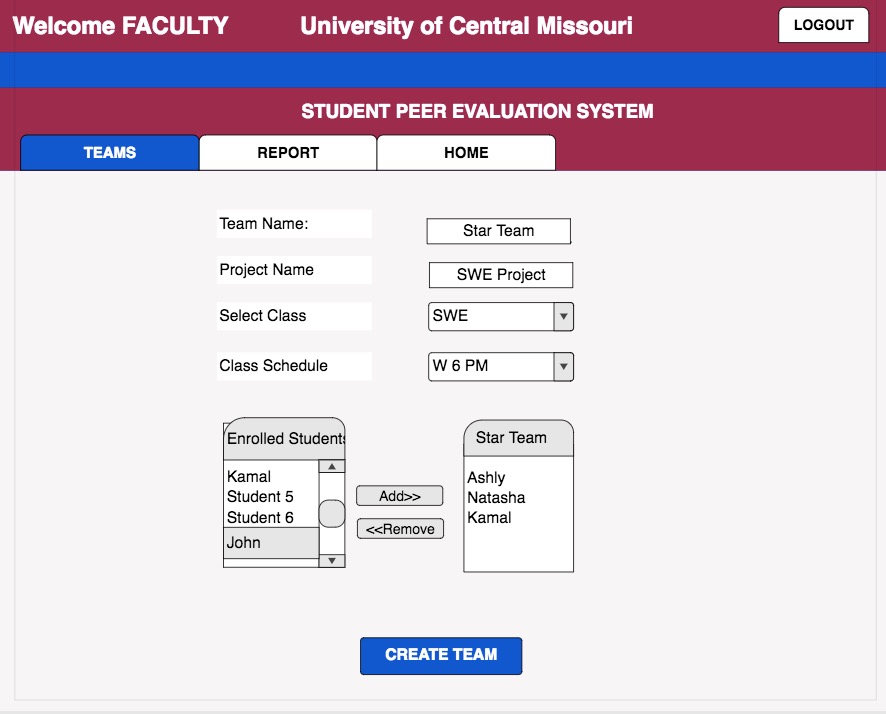
**Basic Course**

From the faculty landing page faculty clicks on the “Create Team” button. The system also displays a new team form along with the list of classes the faculty is teaching. Faculty fills out the form by writing a team name and a project name. Faculty selects a class from the drop down menu. The system displays the list of enrolled students. Faculty selects a student from the list and adds the selected student into the newly created team. Faculty clicks on the “Create Team” button. The system saves the new team. The system redisplays the team view page.

**Alternate Course**

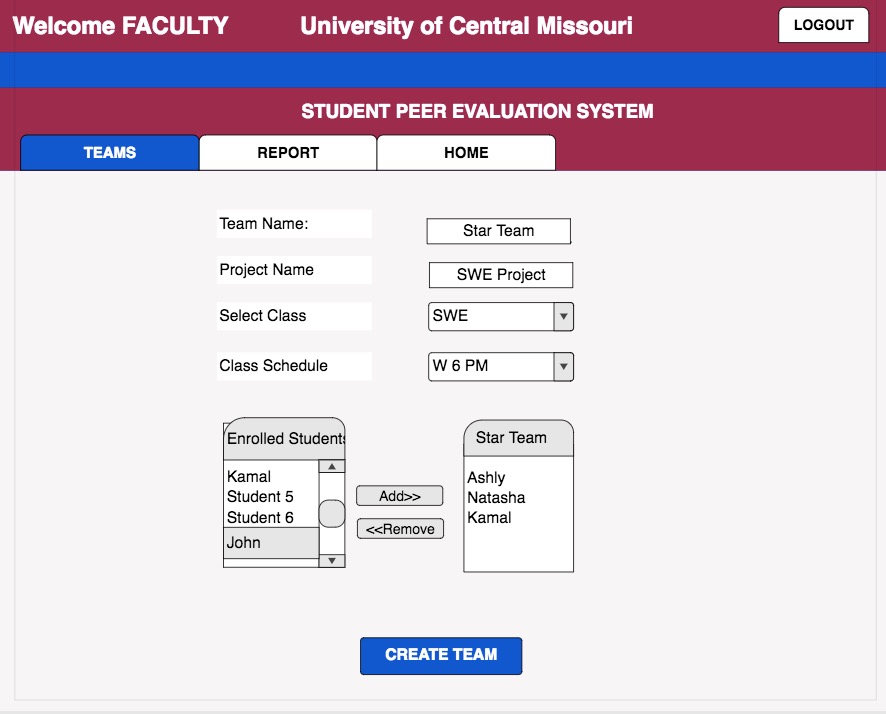
Duplicate Team name: If a duplicate team name exists, the system displays a message “Team name already exists. Click on the “OK” button to enter a new team name.”.

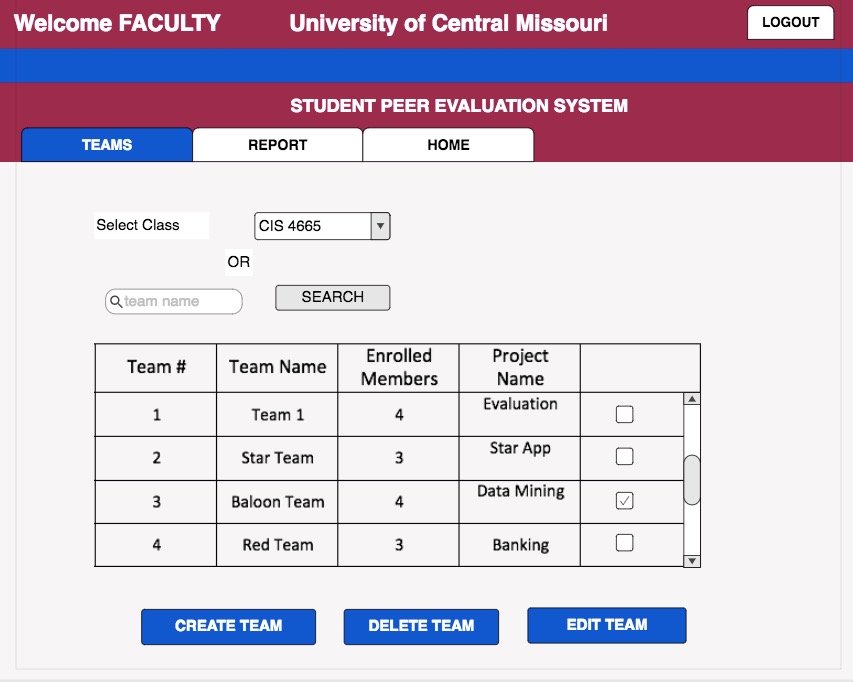
**Use case mock-up web page: (Use case: Faculty Create Team)**

****

**Faculty Create Team – UI Prototype**

Faculty Create Team

****

**USE CASE MODELING:** USE CASE DESCRIPTION, WEB PAGE SCREEN MOCK-UPS & UI PROTOTYPE

Faculty Landing Page

**Use case 6: Admin Add student:**

Actor: Admin

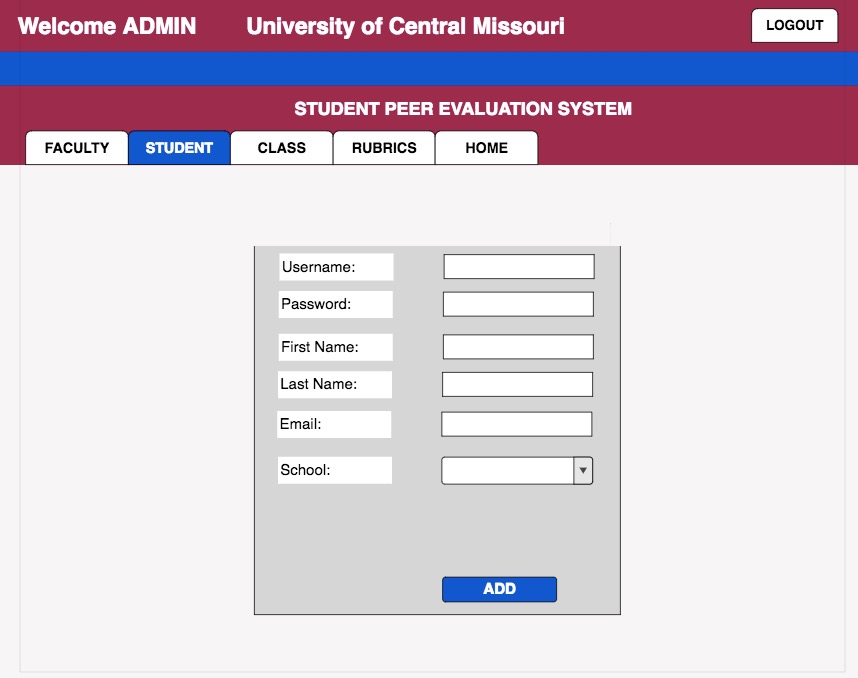
**Basic Course**

Admin clicks on the “Add New Student” button from the student viewer page. The system opens a form along with a list of schools. Admin fills out the form with the new student information such as username, password, first name, last name, phone #, and email address. Admin also selects a school. Admin clicks on the “Add” button. The system checks the users to see if a duplicate record exists. If duplicate record does not exist, the system adds the new student record into the student user and redisplays student view page.

**Alternative Course**

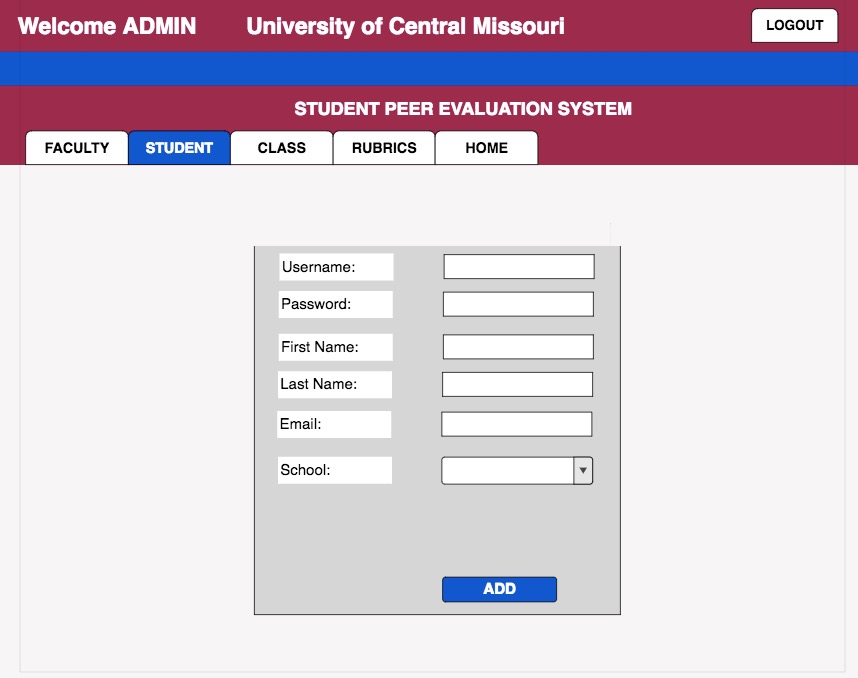
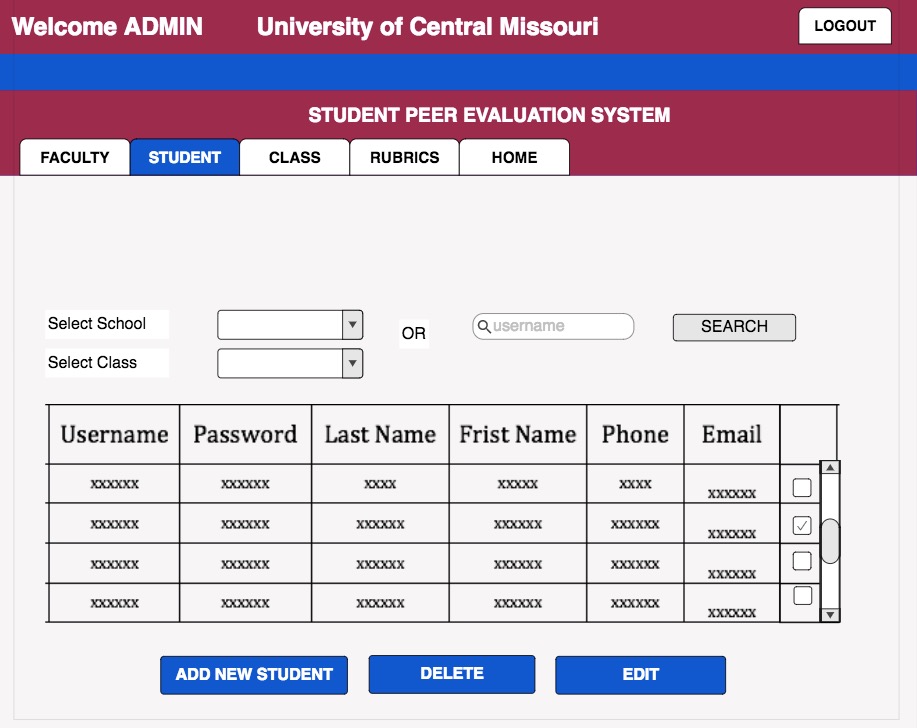
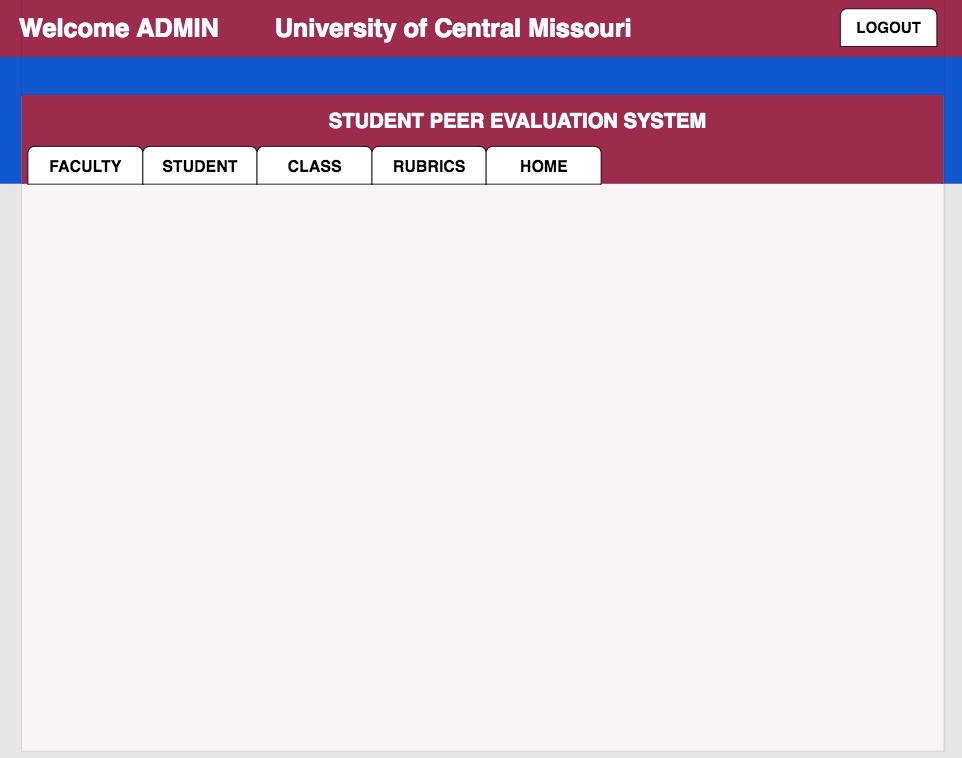
Duplicate Student: If the system finds a match record, the system displays an alert message “Student already exist in the system. Click on “Ok” to add a new student or on “Cancel” to return to the student viewer page.”

**Use case mock-up web page: (Use case: Admin Add Student)**



**Admin Add Student – UI Prototype**

Student View Page



Faculty Add Student

Admin Landing Page

**USE CASE MODELING:** USE CASE DESCRIPTION, WEB PAGE SCREEN MOCK-UPS & UI PROTOTYPE

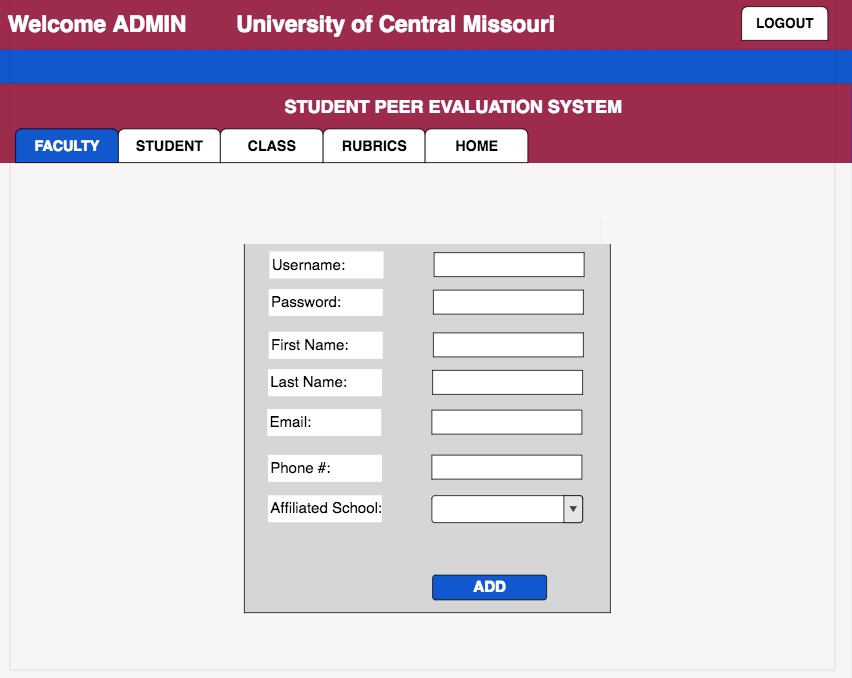
**Use case 7: Admin Add Faculty:**

Actor: Faculty

**Basic Course**

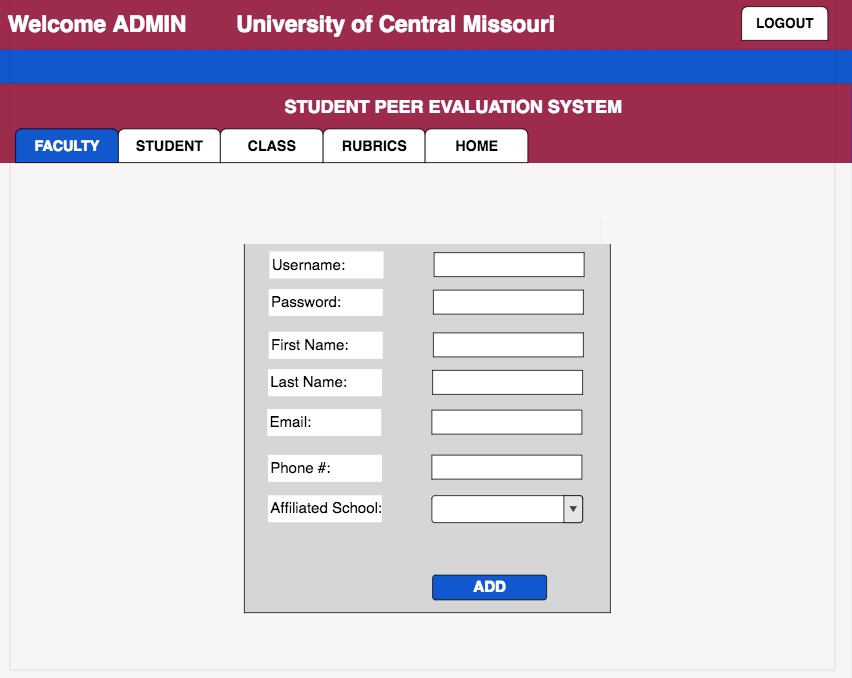
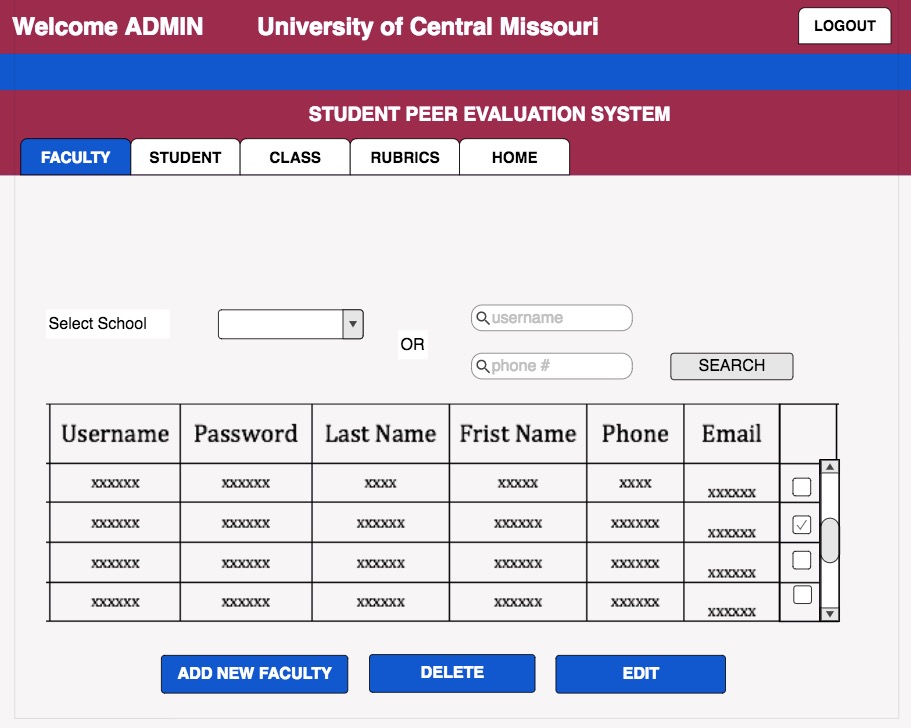
Admin clicks on the “Add New Faculty” button from the faculty viewer page. The system opens a form. Admin fills out the form with the new faculty information such as username, password, first name, last name, phone, email and affiliated school. Admin clicks on the “Add” button. The system checks the users to see if a duplicate record exists. If duplicate record does not exist, the system adds the new faculty record into the list of faculty and redisplays faculty view page.

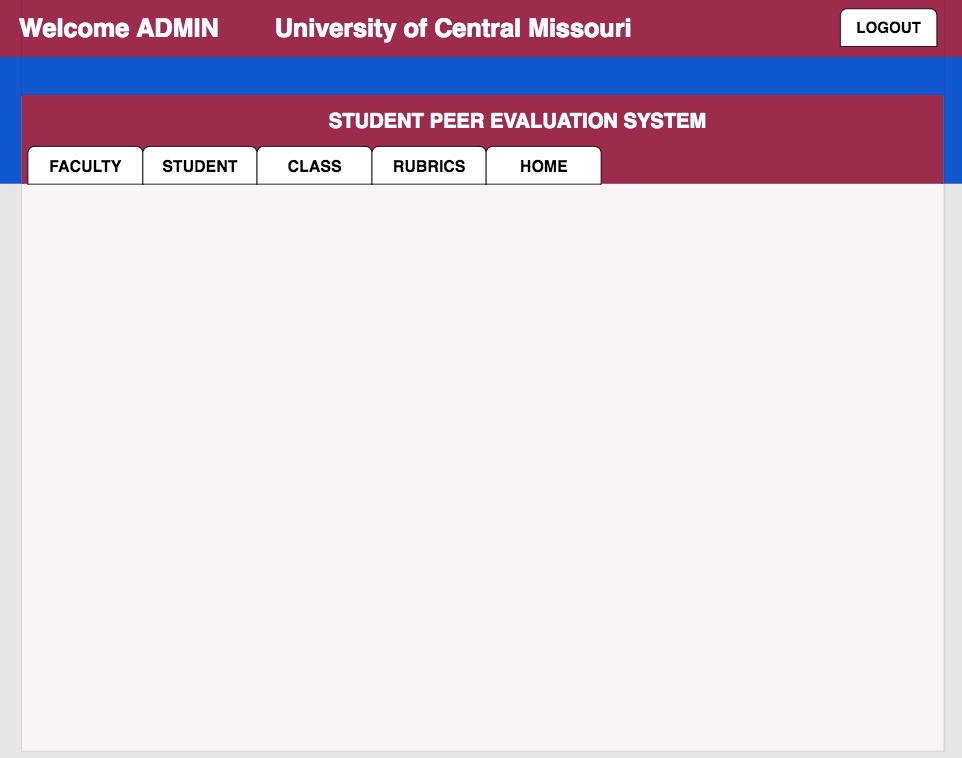
**Alternative Course**

Duplicate Faculty: If the system finds a match record, the system displays an alert message “Faculty already exist on the system. Click on “Ok” to add another faculty or on “Cancel” to return to the faculty page.”

**Use case mock-up web page: (Use case: Admin Add Faculty)**

**Admin Add Faculty – UI Prototype**



****

**USE CASE MODELING:** USE CASE DESCRIPTION, WEB PAGE SCREEN MOCK-UPS & UI PROTOTYPE

**Use case 8: Admin Add Class:**

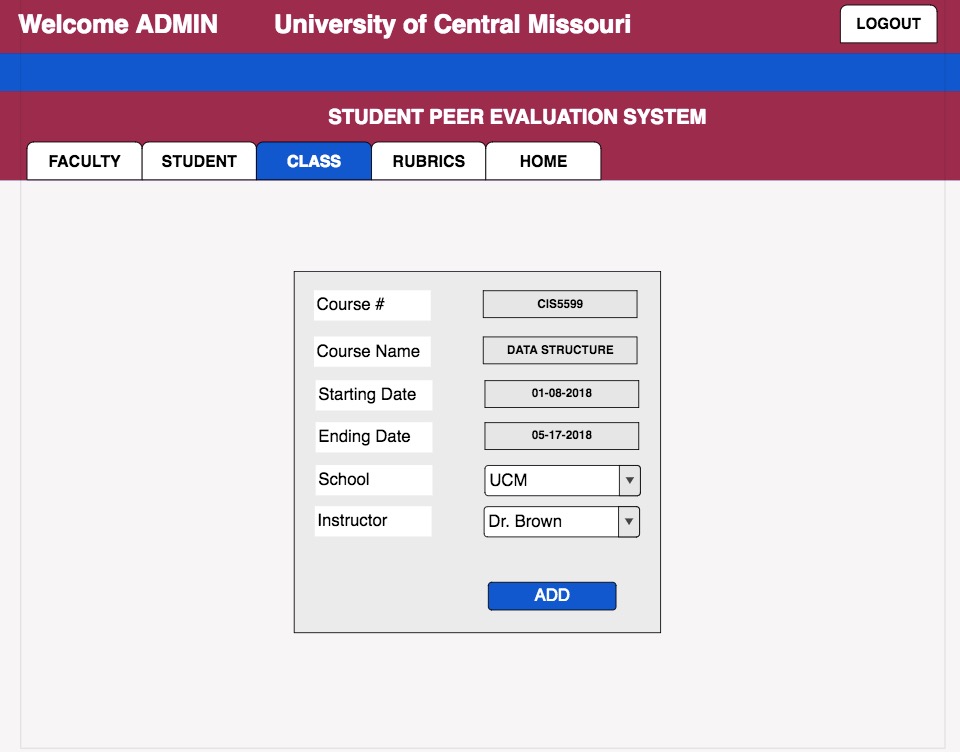
Actor: Admin

**Basic Course**

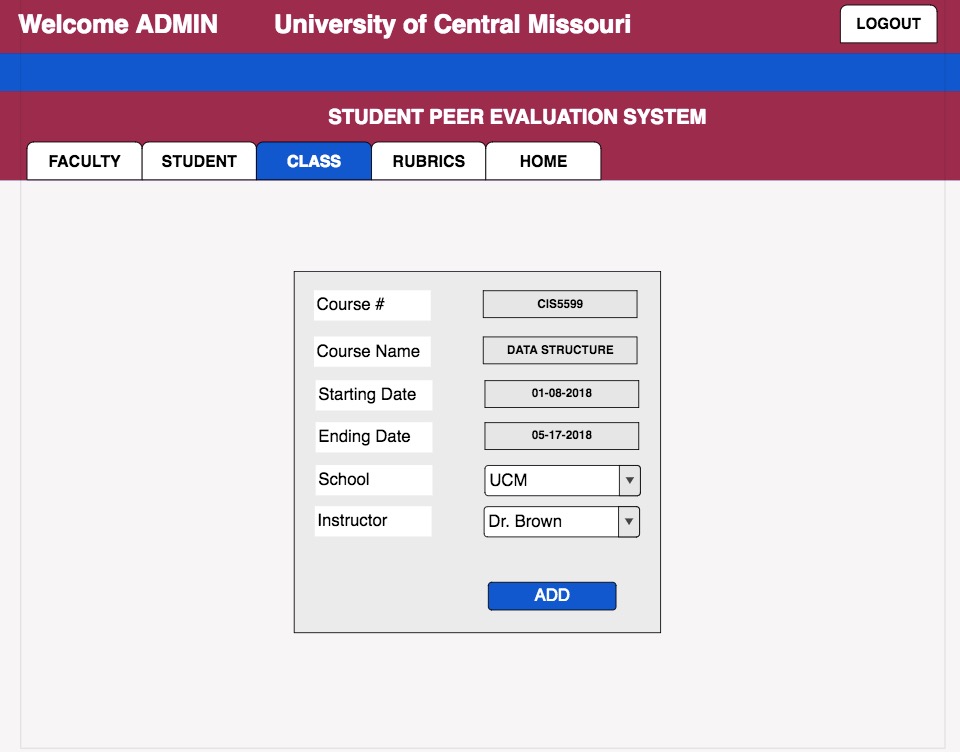
Admin clicks on the “Add New Class” button from the class view page. The system obtains faculty information from the faculty user and displays it along with a form. Admin fills out the form with the following information, course #, course name, starting date, ending date, and school. Admin selects a faculty from the list of faculty. Admin clicks on the “Add” button. The system ensures whether a duplicate class exists or not. If a duplicate class does not exist, the system saves the new class and redisplays the class viewer page.

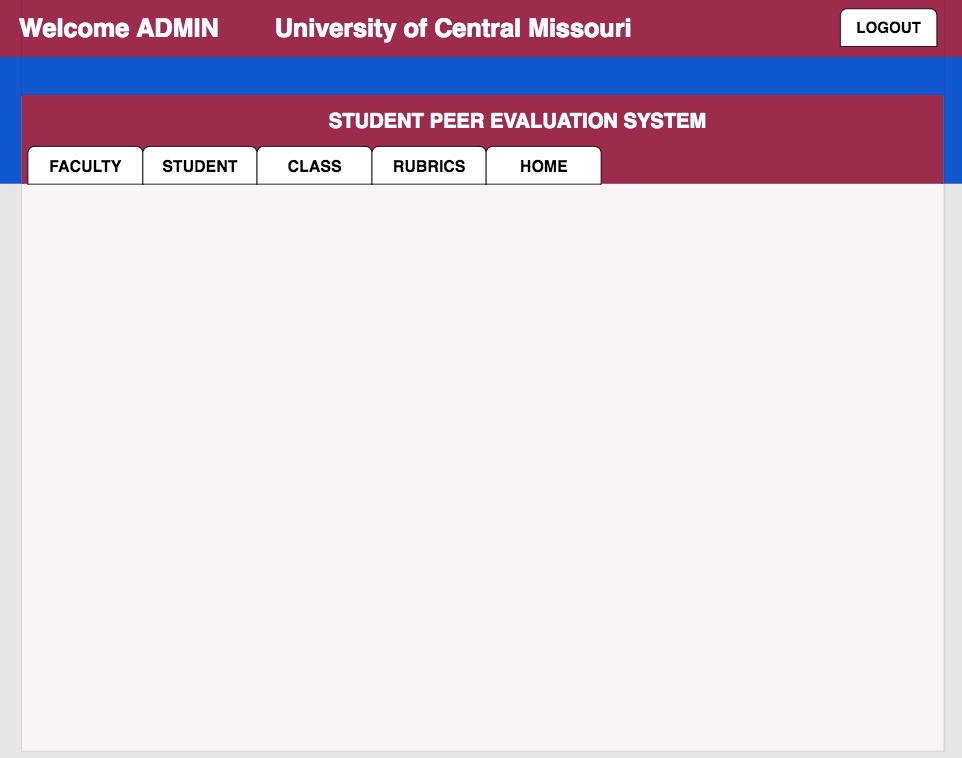
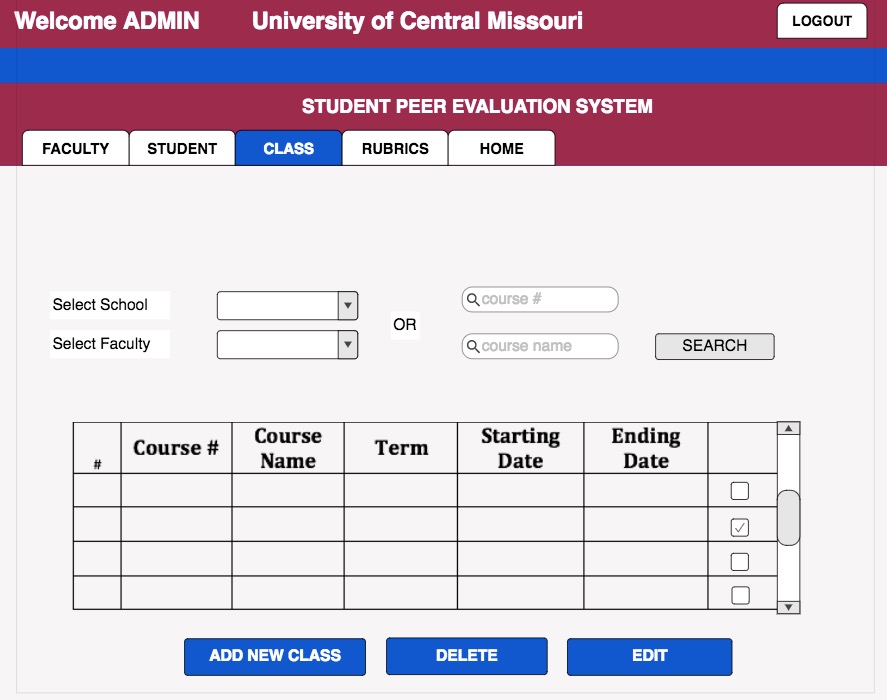
**Alternate Course**

Duplicate Class: If a duplicate class exists, the system displays a message “This class already exist in the system. Click on “OK” to add another class or “Cancel” to return to the class viewer page.

 **Use case mock-up web page: (Use case: Admin Add Class)**

**Admin Add Class – UI Prototype**



******USE CASE MODELING:** USE CASE DESCRIPTION, WEB PAGE SCREEN MOCK-UPS & UI PROTOTYPE

**Use case 9: Admin Add Rubrics:**

Actor: Admin

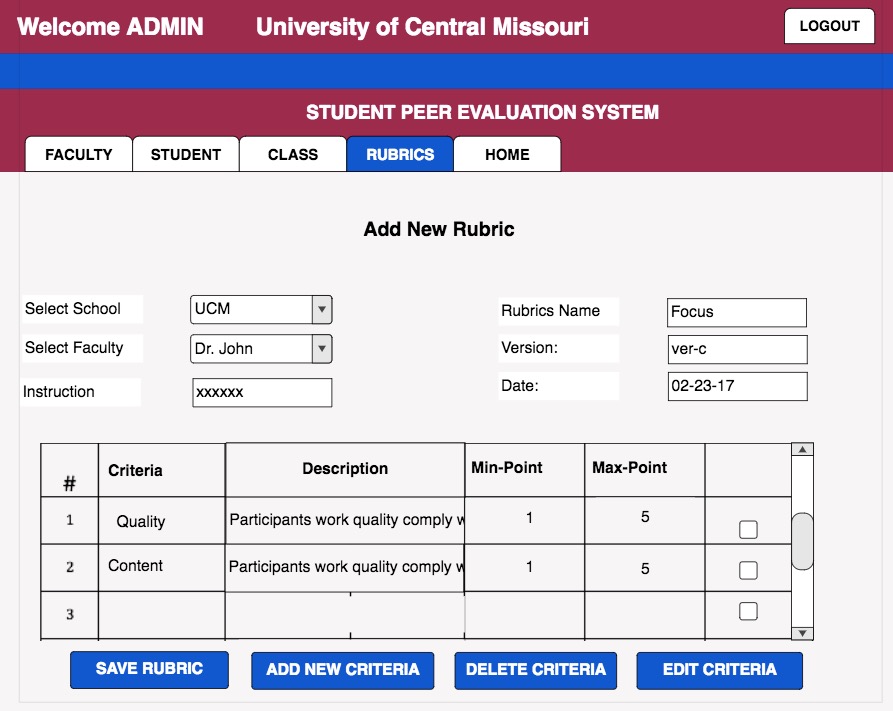
**Basic Course**

Admin clicks on the “Add New Rubrics” button from the rubric view page. The system obtains the list faculties and displays it along with new rubric form. Admin selects a faculty from the list and fills out the remaining of the form with the following information, school, rubric name, version, instruction and the date. Admin clicks on “Add Criteria” button. The system displays criteria form. Admin fills out the rubric criteria. Admin clicks on the “Save Rubric” button. The system ensures that the rubric version is not duplicate. If rubric version is not duplicate, the system saves both the rubric and the rubric criteria.

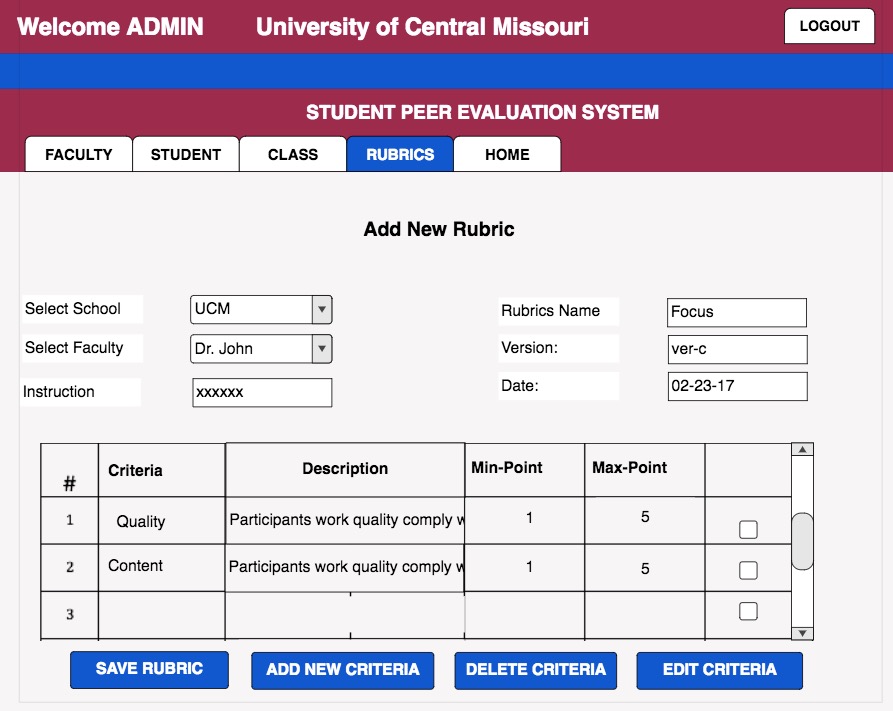
**Alternate Course**

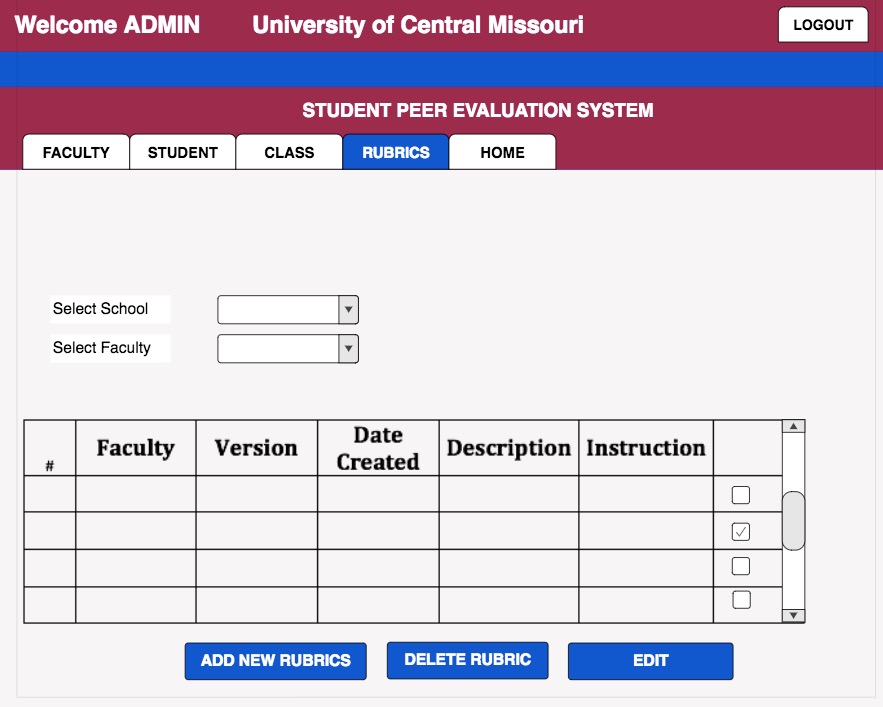
Duplicate rubrics version: If a duplicate rubric version already exist, the system displays a message “The rubric version you entered already exist, please enter a new rubric version name. Click on “OK” to enter a new rubric version.”

**Mock-up web page: (Use case: Admin Add Rubrics)**



**Admin Add Rubrics – UI Prototype**





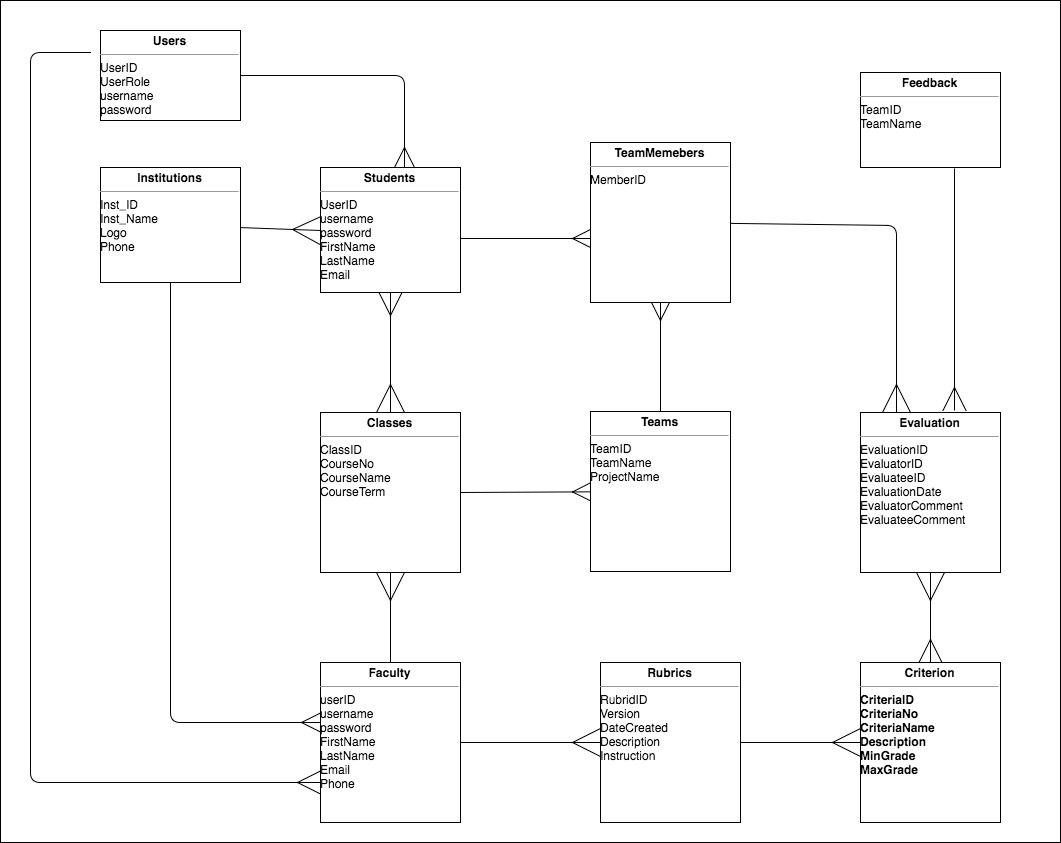
**ANALYSIS CLASS DIAGRAM:**

Available Classes

1. Users
2. Admin
3. Faculty
4. Student
5. Institution
6. Classes
7. Rubrics
8. Criteria
9. Team Members
10. Teams
11. Evaluation
12. Feedback

**ANALYSIS CLASS DIAGRAM:**

**Analysis Class Diagram-Self & Peer Evaluation System**



**ROBUSTNESS DIAGRAM**

**Use case: User Login:**

Actor: Admin, Faculty, or Student

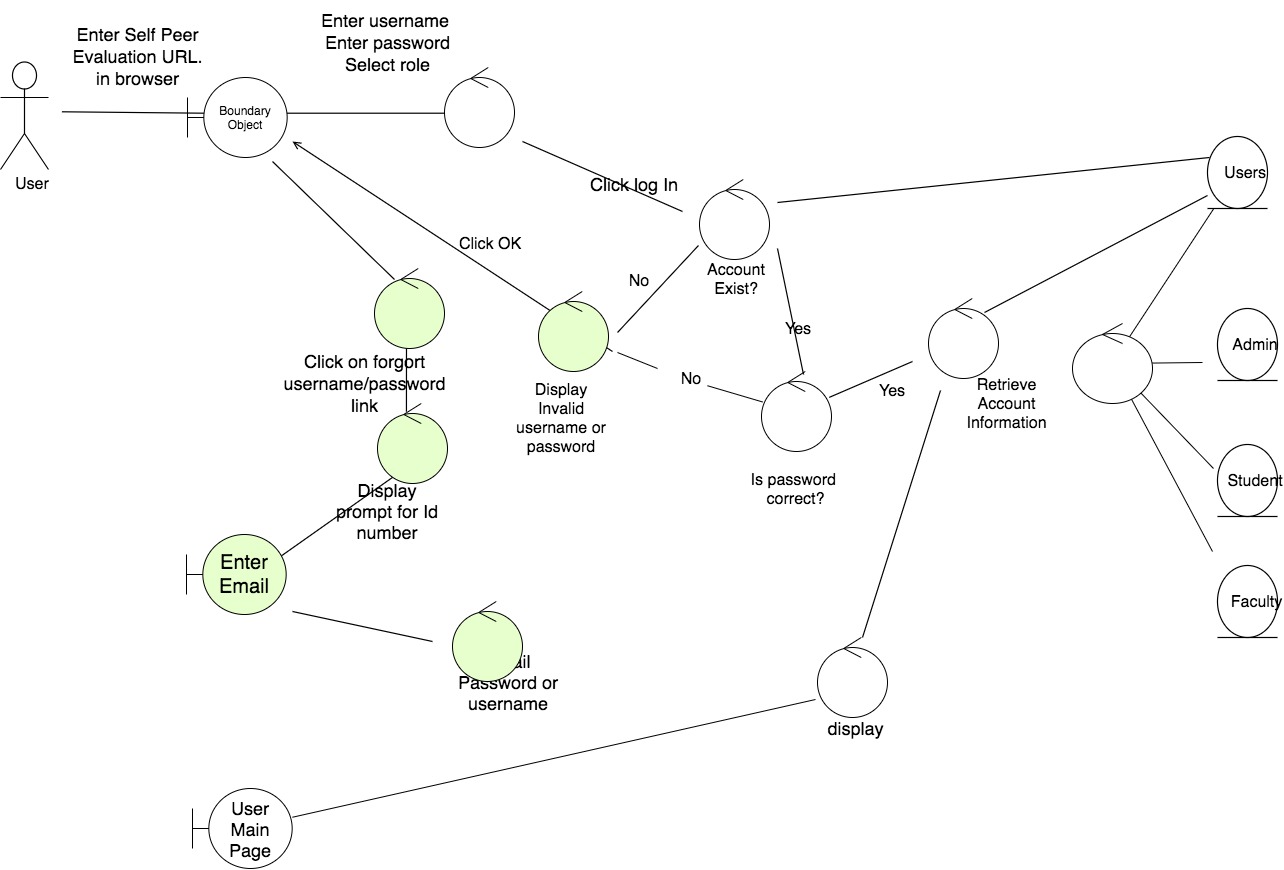
**Basic Course**

User enters the self and peer evaluation URL into the browser. The system opens self and peer evaluation login page. User clicks on the “Select Role” tab. The system opens a drop down menu. User selects her/his role from the drop down menu. The system closes the drop down menu. User types in her/his password and username then clicks on the login button. The system checks to see if the username exists in the “Users”. If an account is found, the system compares the password in the system database. If comparison is successful, the system retrieves the user information, based on the user role. The system displays the user’s landing page.

**Alternate Course**

Invalid username or password: The system checks to see if the username exists in the “Users”. If the username is not found, the system displays a message “Invalid username. Click on “OK” to try again.” The user clicks on the “OK” button. The system redisplays login page. User clicks on the forgot username link. The system asks user to enter his/her email address. User enters the email address associated with his/her account. The system displays a message ”Username was sent to your email.”

**Robustness Diagram for use case USER LOGIN**

****Use case: Student Evaluate Peer:**

Actor: Student

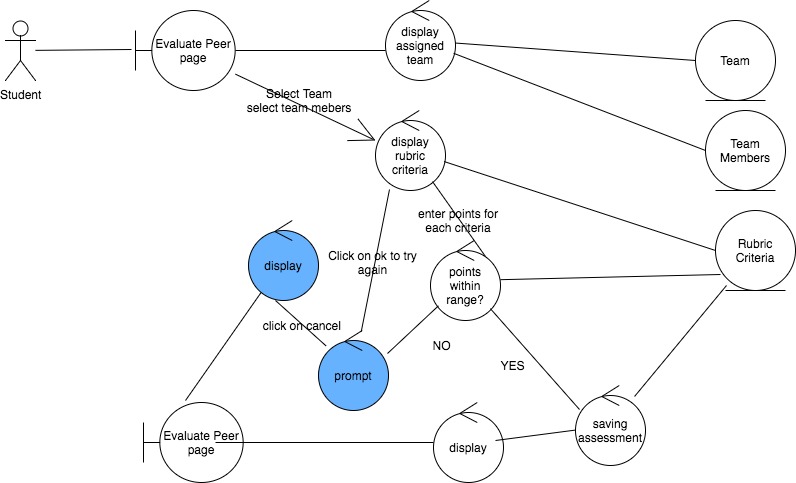
**Basic: Course:**

Student clicks on the “Evaluate Peer” button (from My Assessment page). The system displays list of teams in a drop box. Student selects a team. The system displays list of team members in a drop box. Student selects a team member. The system displays rubric criteria. Student performs the assessment by typing a point to each criterion. Student can optionally write a comment. Student clicks on the “Save” button. The system ensures if the points are within allowable range. If the entered points are within allowable range, the system saves the assessment and redisplays view my assessment (Student landing page) page.

**Alternate Course**

Grading point out of range: The system prompts “Grade not within allowable range (1-5). Click on ”Ok” to try again or on “Cancel” to return to student landing page.

**Robustness Diagram for use case STUDENT EVALUATE PEER**

****

**Use case: Student View Peer Assessment:**

Actor: Student

**Basic Course:**

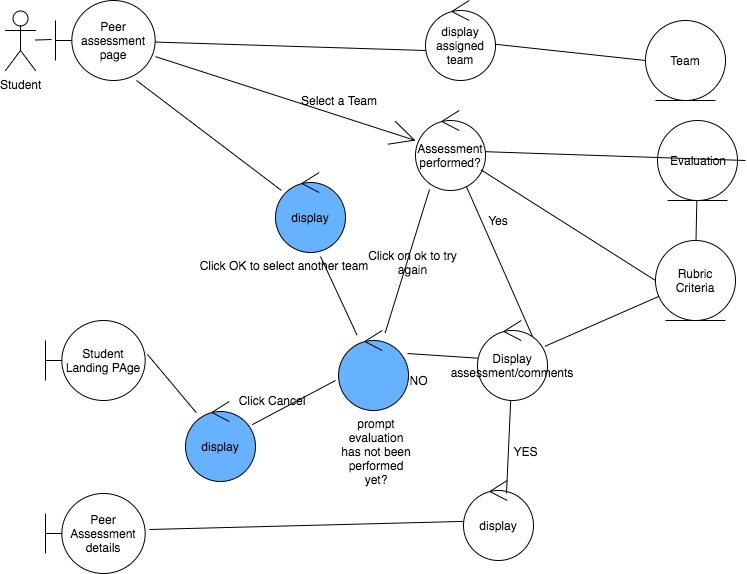
Student clicks on the “Peer Assessment” button (from student landing page).

The system displays the teams the student is assigned to. Student selects a team. The system displays assessments performed by team members.

**Alternate Course**

Assessment not performed yet: If assessment does not exist, the system displays a message “Team members have not performed evaluation yet. Please come back at a later time. Click on “OK” to select another team or on “Cancel” to return to student landing page.”

**Robustness Diagram for use case STUDENT VIEW PEER ASSESSMENT**

****

**Use case: Student Make Comment on Evaluation:**

Actor: Student

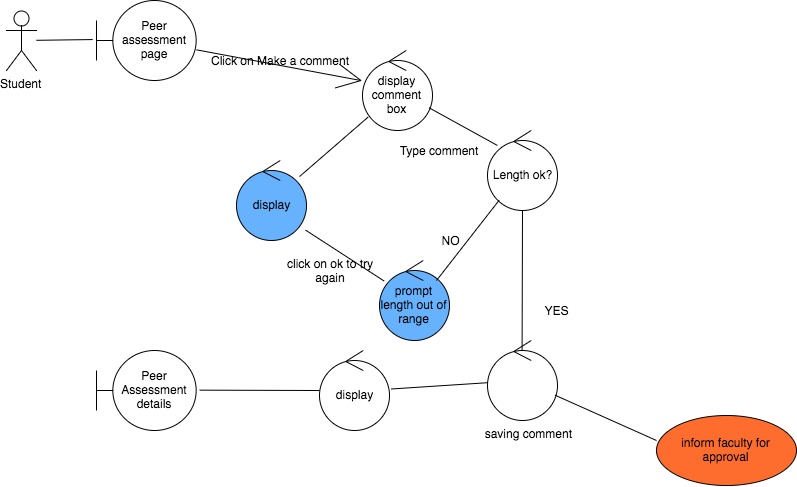
**Basic Course:**

Student clicks on the “Make A Comment” button (from view peer assessment page). The system displays a comment window associated to the desired team member. Student writes a comment on the comment box. Student clicks on the “Save” button. The system ensures the comment length is okay. If the comment length is within allowable range, the system saves the comment and informs the faculty for approval. The system redisplays the “Peer Assessment” page.

**Alternate Course**

Too long or short comment: The system displays a message “Comment length is out side the allowable range. Click on “OK” to try again.”

**Robustness Diagram for use case STUDENT COMMENT ON EVALUATION**



**Use case: Faculty Create Team:**

Actor: Faculty

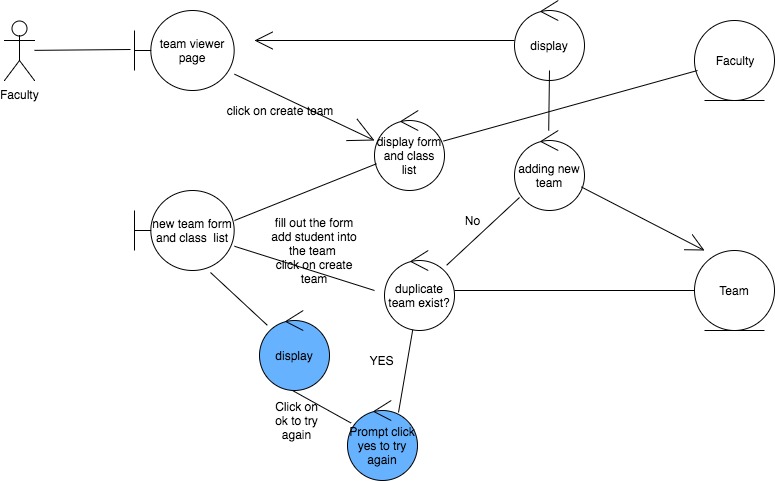
**Basic Course**

From the faculty landing page faculty clicks on the “Create Team” button. The system also displays a new team form along with the list of classes the faculty is teaching. Faculty fills out the form by writing a team name and a project name. Faculty selects a class from the drop down menu. The system displays the list of enrolled students. Faculty selects a student from the list and adds the selected student into the newly created team. Faculty clicks on the “Create Team” button. The system saves the new team. The system redisplays the team view page.

**Alternate Course**

Duplicate Team name: If a duplicate team name exists, the system displays a message “Team name already exists. Click on the “OK” button to enter a new team name.”.

**Robustness Diagram for use case FACULTY CREATE TEAM**



**Use case: Admin Add Student:**

Actor: Admin

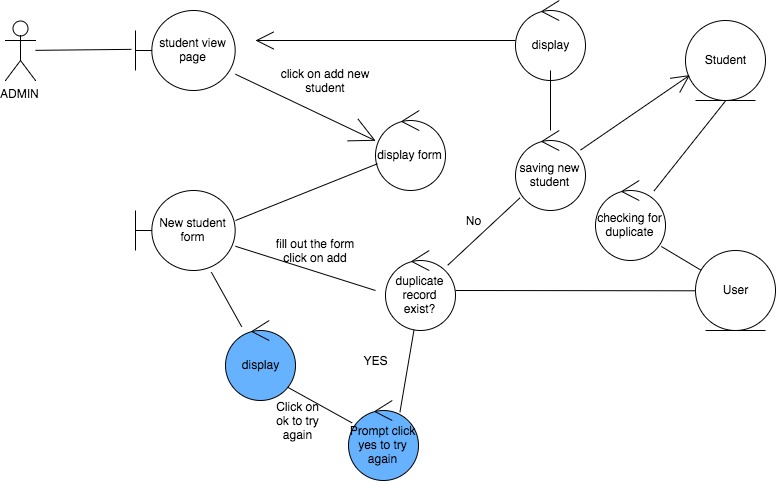
**Basic Course**

Admin clicks on the “Add New Student” button from the student viewer page. The system opens a form along with a list of schools. Admin fills out the form with the new student information such as username, password, first name, last name, phone #, and email address. Admin also selects a school. Admin clicks on the “Add” button. The system checks the users to see if a duplicate record exists. If duplicate record does not exist, the system adds the new student record into the student user and redisplays student view page.

**Alternative Course**

Duplicate Student: If the system finds a match record, the system displays an alert message “Student already exist in the system. Click on “Ok” to add a new student or on “Cancel” to return to the student viewer page.”

**Robustness Diagram for use case ADMIN ADD STUDENT**

**Use case: Admin Add Faculty:**

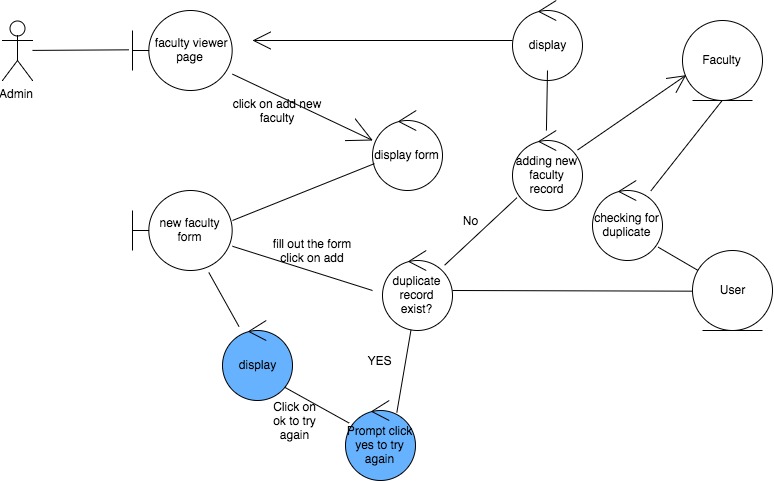
Actor: Faculty

**Basic Course**

Admin clicks on the “Add New Faculty” button from the faculty viewer page. The system opens a form. Admin fills out the form with the new faculty information such as username, password, first name, last name, phone, email and affiliated school. Admin clicks on the “Add” button. The system checks the users to see if a duplicate record exists. If duplicate record does not exist, the system adds the new faculty record into the list of faculty and redisplays faculty view page.

**Alternative Course**

Duplicate Faculty: If the system finds a match record, the system displays an alert message “Faculty already exist on the system. Click on “Ok” to add another faculty or on “Cancel” to return to the faculty page.”



**Robustness Diagram for use case ADMIN ADD FACULTY**

**Use case: Admin Add Class:**

Actor: Admin

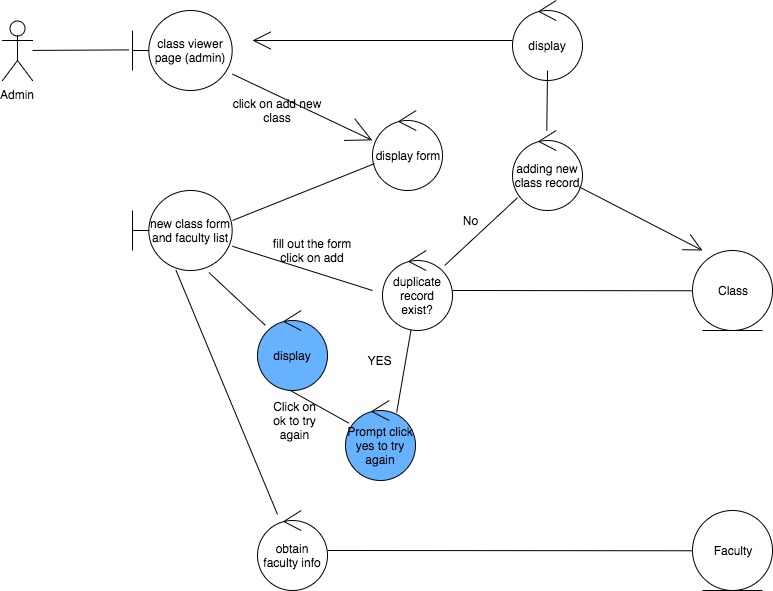
**Basic Course**

Admin clicks on the “Add New Class” button from the class view page. The system obtains faculty information from the faculty user and displays it along with a form. Admin fills out the form with the following information, course #, course name, starting date, ending date, and school. Admin selects a faculty from the list of faculty. Admin clicks on the “Add” button. The system ensures whether a duplicate class exists or not. If a duplicate class does not exist, the system saves the new class and redisplays the class viewer page.

**Alternate Course**

Duplicate Class: If a duplicate class exists, the system displays a message “This class already exist in the system. Click on “OK” to add another class or “Cancel” to return to the class viewer page.

**Robustness Diagram for use case ADMIN ADD CLASS**

**Use case: Admin Add Rubrics:**

Actor: Admin

**Basic Course**

Admin clicks on the “Add New Rubrics” button from the rubric view page. The system obtains the list faculties and displays it along with new rubric form. Admin selects a faculty from the list and fills out the remaining of the form with the following information, school, rubric name, version, instruction and the date. Admin clicks on “Add Criteria” button. The system displays criteria form. Admin fills out the rubric criteria. Admin clicks on the “Save Rubric” button. The system ensures that the rubric version is not duplicate. If rubric version is not duplicate, the system saves both the rubric and the rubric criteria.

**Alternate Course**

Duplicate rubrics version: If a duplicate rubric version already exist, the system displays a message “The rubric version you entered already exist, please enter a new rubric version name. Click on “OK” to enter a new rubric version.”

**Robustness Diagram for use case ADMIN ADD RUBRICS**

