**Documentation for Gaming Website Designing  
  
  
  
Deepak K Jain  
  
  
  
  
  
  
  
CIS 5690 -Advanced Systems Project (Fall, 2019)  
Dr. Qingxiong Ma  
December 08, 2019**

**Table of Contents**

[**Problem Statement** 2](#_Toc26608698)

[**Project Scope** 2](#_Toc26608699)

[**List of Functional Requirements** 2](#_Toc26608700)

[**Object Domain Model** 3](#_Toc26608701)

[**Step 1. Identifying Primary List of Objects** 3](#_Toc26608702)

[**Step 2. Eliminating Duplicated and Unnecessary Items** 3](#_Toc26608703)

[**Step 3. Building Aggregation Relationships** 5](#_Toc26608704)

[**Step 4. Identify Further Domain Objects** 6](#_Toc26608705)

[**Step 5. Building Generalization Relationships** 6](#_Toc26608706)

[**Step 6. Building Domain Model** 7](#_Toc26608707)

[**Glossary of The Domain Objects:** 7](#_Toc26608708)

[**List of Use Cases** 8](#_Toc26608709)

[**Identifying Actors’ List** 8](#_Toc26608710)

[**Identifying the List of Goals:** 8](#_Toc26608711)

[**Making List of Use Cases:** 8](#_Toc26608712)

[**Use Case Descriptions and Screen Mock-Ups** 9](#_Toc26608713)

[**Use Case Diagrams** 21](#_Toc26608714)

[**Use Case Diagram: Student and Instructor** 21](#_Toc26608715)

[**Use Case Diagram: Student and Admin** 22](#_Toc26608716)

[**Things to keep in mind for use case diagram** 22](#_Toc26608717)

[**Analysis Class diagram** 23](#_Toc26608718)

[**Activity Diagrams** 24](#_Toc26608719)

[**Robustness Diagrams** 33](#_Toc26608720)

[**References 42**](#_Toc26608721)

# **Problem Statement**

It is important to be happy. According to lifecoach Expert, - “We become more creative, witty, energetic and fun to be around and it can also lead us to become more financially successful.”

The people can achieve happiness by utilizing their spare time in doing something they like to do; like playing games or gardening or cooking. There are a lot of people in today’s world that feel happy in playing games.

# **Project Scope**

The project should be able to deliver some games that can be played using mouse clicks and keyboard buttons. There should not be any additional hardware required for the user to play the games. There should be a homepage where all the games are hosted and linked to homepage. At the homepage there should be a screen that shows the look and feel of each of the game.

The user should be able to navigate to the individual game pages. When the user clicks on the screens for individual game page, he should be able to navigate to the game page and there must also be links available on the individual game page for the user to navigate among the games and the all pages should be linked to the game homepage.

# **List of Functional Requirements**

Below are the functional requirements of the project.

|  |  |
| --- | --- |
| **Sr. No.** | **Functional requirements** |
| 1 | System should be able to **create** login for the Instructors, students and Admin. |
| 2 | Admin should **manage** Instructors, students, classes, and rubrics defined. |
| 3 | Students from different classes can **participate** in different projects. |
| 4 | System should allow Instructors to **manage** project teams for their classes. |
| 5 | System will also be **used** by another departments” Instructors. |
| 6 | System should allow Instructor to **assign** students to different project teams. |
| 7 | System should allow students to **perform** peer and self-assessments. |
| 8 | Student should be able to **view** self-assessments done by them and peer-assessments performed by other team members. |
| 9 | Students should **give** **feedback** for activity of their group. |
| 10 | Students can **make comments** on the performance of other students. |
| 11 | System should **generate reports** of member performance individual, class and team level. |

# **Object Domain Model**

To create the object domain model, I followed the below steps:

**Step 1. Identifying Primary List of Objects** - The below objects are primarily looks useful for the project. First, we will list down all the objects and then in the next step we will eliminate the objects that are not necessary for the project.

**Primary list of objects is as under:**

Instructors

Project teams

Students

Projects

Class

Users

Administrator

Login Level

Department

Programs

Courses

Rubric information

Self-assessment

Peer-assessment

Feedback

Open ended question

Close ended question

Reports

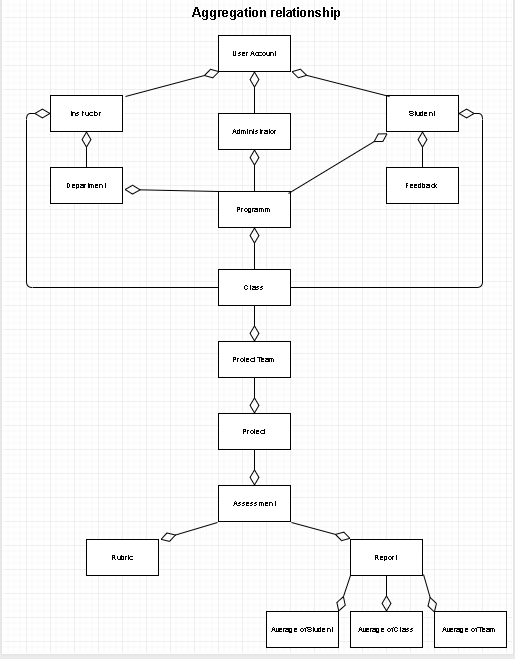
Individual

Comments

**Step 2. Eliminating Duplicated and Unnecessary Items** - In this step we are eliminating duplicate and unnecessary items and making an updated list-

|  |  |  |
| --- | --- | --- |
| **Identified** | | **Eliminated** |
| Instructors | Programs | Users |
| Project teams | Courses | Open ended question |
| Students | Rubric information | Close ended question |
| Projects | Self-assessment | Individual |
| Class | Peer-assessment |  |
| Administrator | Feedback |  |
| Login Level | Reports |  |
| Department | Comments |  |

**Step 3. Building Aggregation Relationships -** In this step we are building aggregation relationships in the domain model-



**Step 4. Identify Further Domain Objects** – In this step we identify further domain objects that were not identified in the initial requirements-

Average of individual

Average of team

Average of class

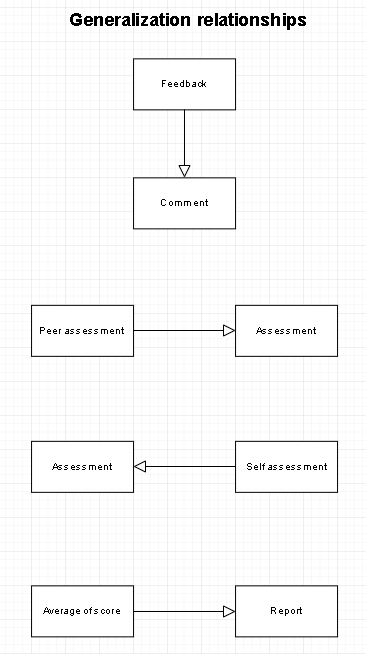
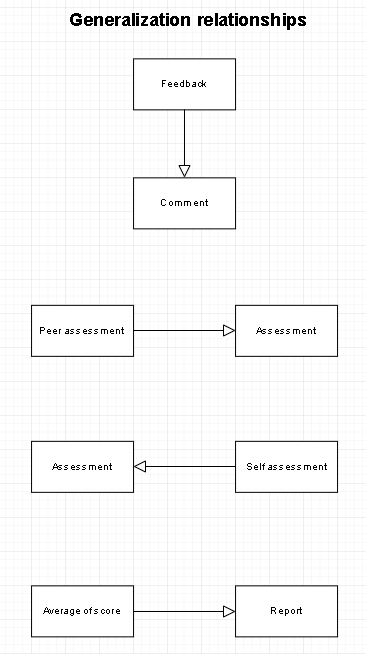
Submit

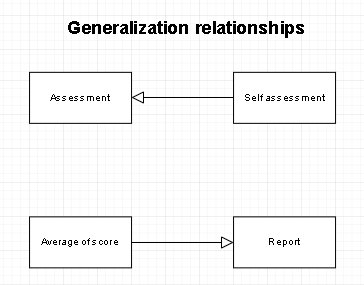
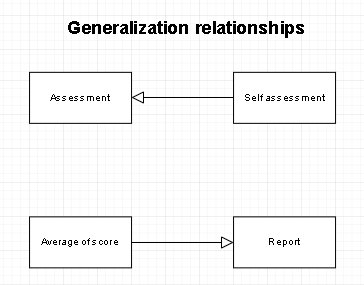
Retrieve

Plagiarism check

Chat area

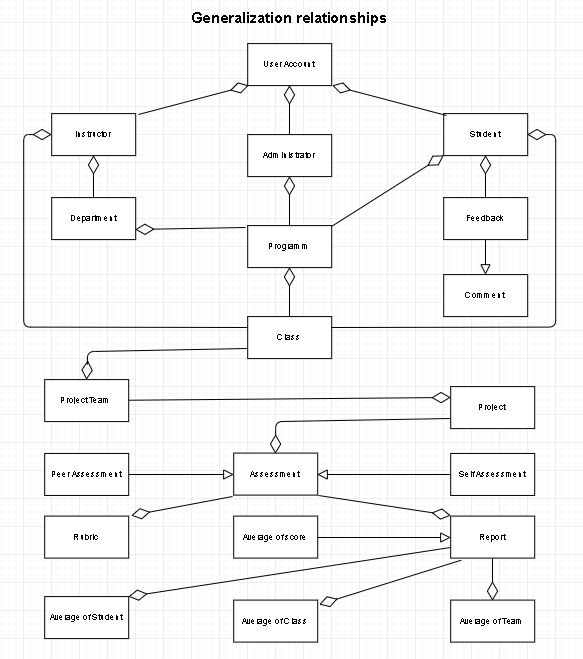
**Step 5. Building Generalization Relationships –** In this step we are building generalization relationships in the domain model-





**Step 6. Building Domain Model –** In this step we are building domain model based on both aggregation and generalization relationships-

**Domain Model**



## **Glossary of The Domain Objects:**

**Administrator–** All the access controls stay with this object in the system.

**Instructors–**They will assign projects to the students and create teams and are responsible for management of class.

**Students–**The students are the objects who perform the self and peer assessments. They are also called team members.

**Department–** This object has the entire information related to the departments of the university.

**Classes–** Various classes attended by the students of different departments. Instructor can teach more than one class and students can also attend more than one class.

**Projects–**The projects are assigned by the Instructors to students and students will perform based on each individual project.

**Team–**The instructor willfor a team of 3 to 4 students for the completion of each project throughout the semester.

**Self-Assessment–** Self-Assessmentis done by the team members (students) for themselves.

**Peer Evaluation–** Peer Evaluation is a team assessment performed by students for other teams.

**Reports–** Reports are created by system for instructors for improving the students’ performance.

**Feedback**–The feedback from students about the performance of other students in several individual projects throughout the semester.

**Comments–** Studentcomments are provided by the students on team member performance in each group project. Instructors” comments are provided by instructors on the student comments.

# **List of Use Cases**

**Identifying Actors’ List:** There are main 3 actors for this project.

1. Administrator
2. Instructors
3. Students

## **Identifying the List of Goals:**

1. **Administrator**

* Login Page for Admin
* Managing Rubrics
* Manage Users
* Managing Classes

1. **Instructors**

* Login Page for Instructors
* Adding the projects
* Assigning the projects
* Make teams
* Make comments
* View Reports

1. **Students**

* Login Page for Students
* Perform Peer and Self-Assessments
* Make comments

## **Making List of Use Cases:**

1. User logins to the system
2. Admin Adds Students
3. Admin Updates Students
4. Admin Updates Classes
5. Admin Adds Classes
6. Instructor Updates Teams
7. Instructor Adds Teams
8. Instructor Approves comments
9. Team member makes assessments

# **Use Case Descriptions and Screen Mock-Ups**

Full use case description with numbered event flows and screen mock-ups are below:

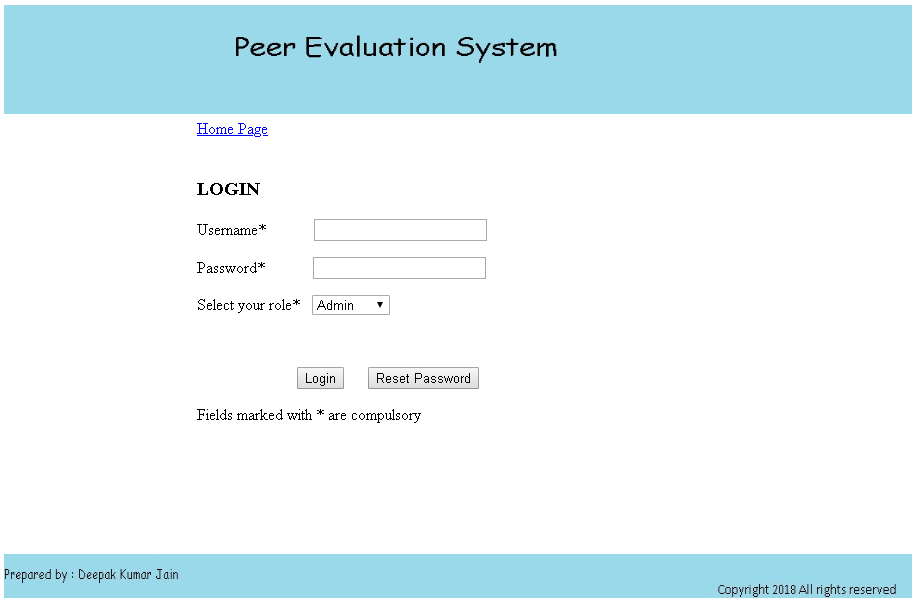
**Use case 1 -** User logins to the system**:**

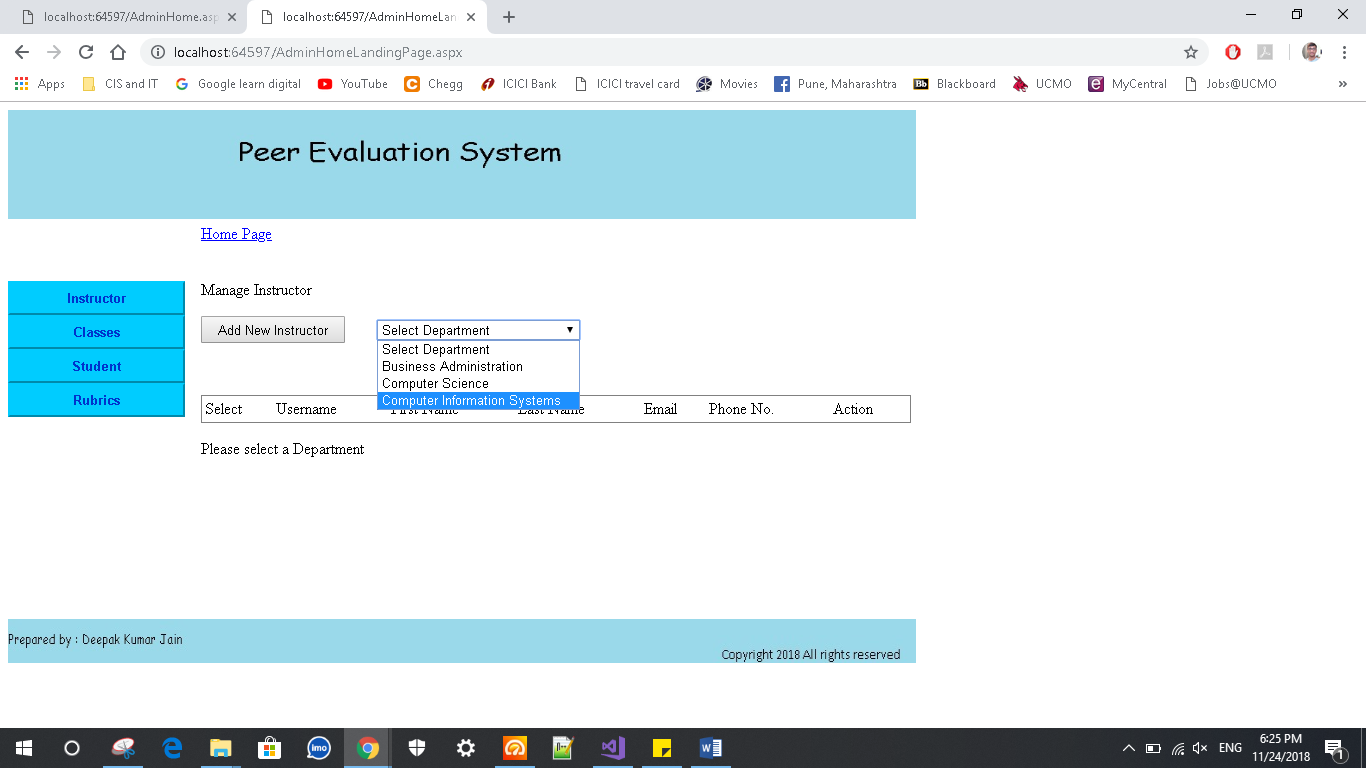
**Main flow:**

1. **Primary actor**: User.
2. **Precondition**: User is on the Home page.
3. **Main success scenario**:
   1. The user clicks on the link to navigate to the login page.
   2. The user enters username and password.
   3. The user selects the role.
   4. The user clicks on the login button.

**Alternative flow:**

1. **Primary actor**: User.
2. **Precondition**: User is on the Homepage.
3. **Main failure scenario:** 
   1. The user clicks on the link to navigate to the login page.
   2. The user enters the wrong password.
   3. The user is redirected to the “User settings page” where the user has to set new password along with a new security question for authentication.
   4. After setting the new password, the user will be able to login again with the right credentials.

The user clicks on login link and then the below screens appears. The user provides username, password and selects role and clicks login button.

Once the user clicks login button, the system will check the database and shows the below screen if correct username and password are entered.

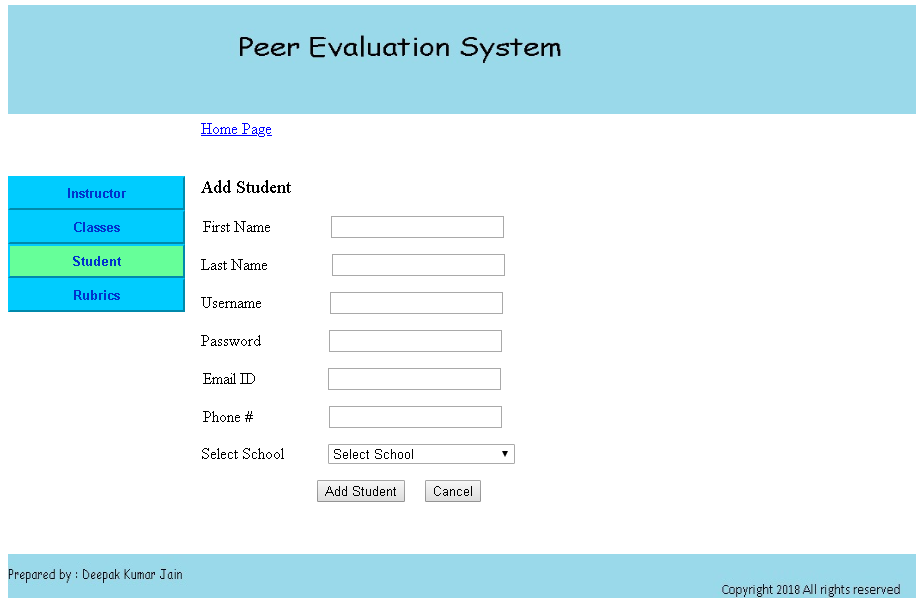
**Use case 2 -** Admin Adds Students

**Main flow:**

1. **Primary actor**: Admin
2. **Precondition**: The admin clicks the “Adds Student” Button
3. **Main success scenario:** 
   1. The admin selects the roles for Student.
   2. The admin adds student details and clicks “Add Student” Button.
   3. The admin selects school and instructor.
   4. The student will be added to the database.

**Alternative flow:**

1. **Primary actor:** Admin
2. **Precondition**: The admin clicks the “Adds New Student” Button.
3. **Main success scenario:** 
   1. The admin selects the roles for Student.
   2. The admin adds student details and clicks “Add New Student” Button.
   3. The admin selects school and instructor.
   4. The admin gets error as the student details are incomplete.

The admin clicks the “Adds New Student” Button on Student page and reaches below screen. Then admin will enter first name, last name, username, password, email ID, phone number and then select school and click “Add Student” button 

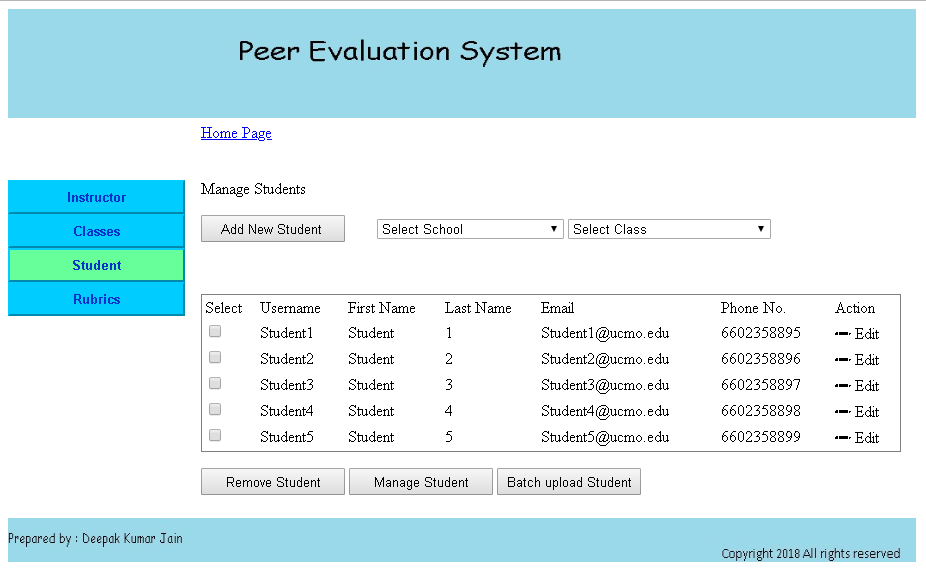
**Use case 3 -** Admin Updates Students

**Main flow:**

1. **Primary actor:** Admin
2. **Precondition**: The admin clicks the “Update Student” Button
3. **Main success scenario:** 
   1. The admin selects the roles for Student.
   2. The admin update student details and clicks “Manage Student” Button.
   3. The student details will be updated in the database.

**Alternative flow:**

1. **Primary actor:** Admin
2. **Precondition**: The admin clicks the “Update Student” Button.
3. **Main success scenario:** 
   1. The admin selects the roles for Student.
   2. The admin adds student details and clicks “Manage Student” Button.
   3. The admin gets error as the student details are incomplete.

The admin will click “Update students” button and comes to Manage Students page as shown below 

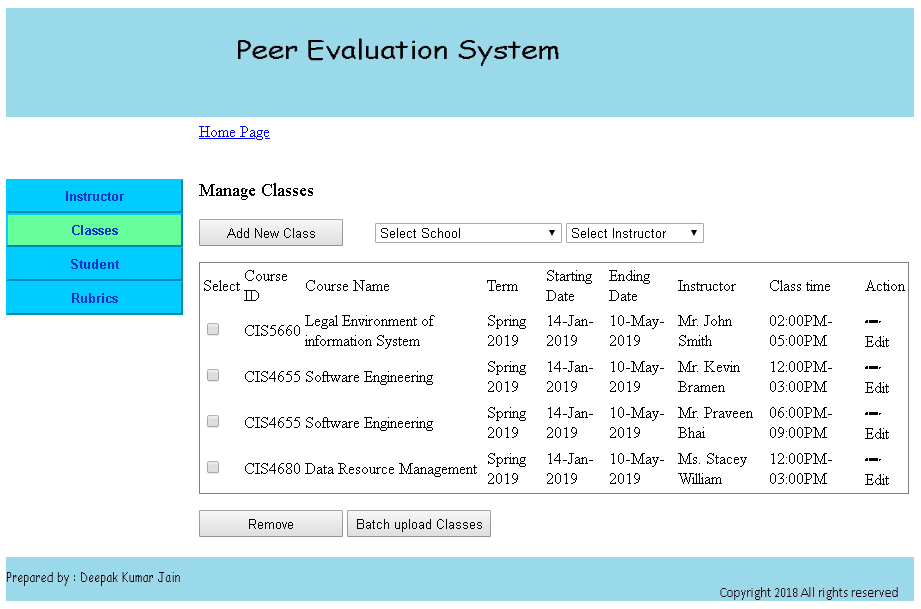
**Use case 4 -** Admin Updates Classes

**Main flow:**

1. **Primary actor:** Admin
2. **Precondition**: The admin clicks the “Update Classes” Button
3. **Main success scenario:** 
   1. The admin selects school.
   2. The admin selects instructor.
   3. The admin selects the class line checkbox to update class.
   4. The admin update class details and clicks “Update Class” Button.
   5. The class details will be updated in the database.

**Alternative flow:**

1. **Primary actor:** Admin
2. **Precondition**: The admin clicks the “Update Classes” Button.
3. **Main success scenario:** 
   1. The admin selects school.
   2. The admin selects instructor.
   3. The admin selects the class line checkbox to update class.
   4. The admin update class details and clicks “Update Class” Button.
   5. The admin gets error as the class details are incomplete.

The admin will click “Update Class” button and comes to Manage Classes page as shown below 

**Use case 5 -** Admin Adds Classes

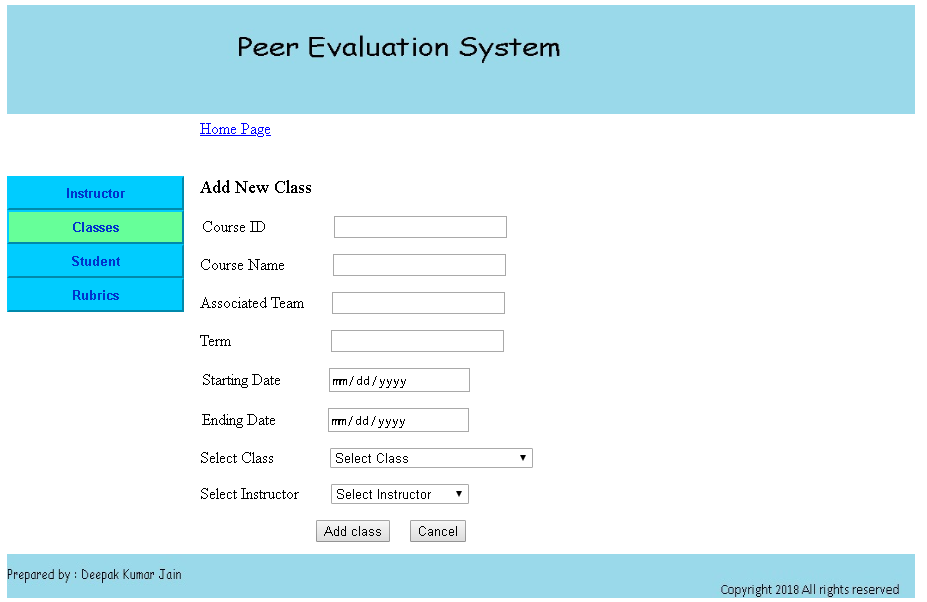
**Main flow:**

1. **Primary actor:** Admin**.**
2. **Precondition**: The admin clicks the “Add New Class” Button.
3. **Main success scenario:** 
   1. The admin selects school.
   2. The admin selects instructor.
   3. The admin enters details (Course ID, Course Name, Associated Team, Term, Starting Date, Ending Date, Class and Instructor) to add class.
   4. The admin clicks “Add Class” Button.
   5. The class details will be added in the database.

**Alternative flow:**

1. **Primary actor:** Admin
2. **Precondition**: The admin clicks the “Add New Class” Button.
3. **Main success scenario:** 
   1. The admin selects school.
   2. The admin selects instructor.
   3. The admin enters details (Course ID, Course Name, Associated Team, Term, Starting Date, Ending Date, Class and Instructor) to add class.
   4. The admin clicks “Add Class” Button.
   5. The admin gets error as the class details are incomplete.

The admin clicks the “Adds New Class” Button on Classes page and reaches below screen. Then admin will enter Course ID, Course Name, Associated Team, Term, Starting Date, Ending Date, Class and Instructor and then select class and click “Add Class” button



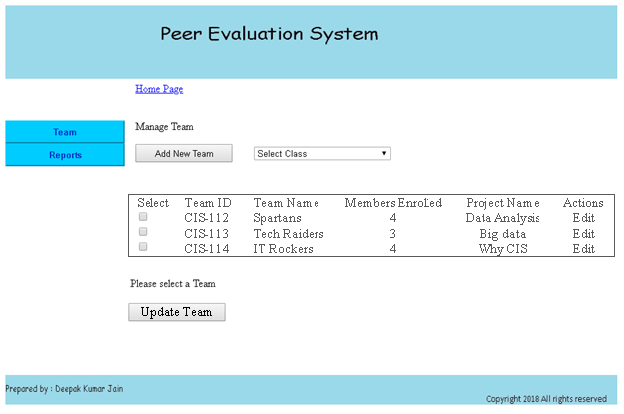
**Use case 6 -** Instructor Updates Teams

**Main flow:**

1. **Primary actor:** Instructor**.**
2. **Precondition**: The Instructor clicks the “Update Teams” Button.
3. **Main success scenario:** 
   1. The Instructor selects class.
   2. The Instructor selects the team line checkbox to update team.
   3. The Instructor selects member names to update team.
   4. The Instructor clicks “Update Team” Button.
   5. The Team details will be updated in the database.

**Alternative flow:**

1. **Primary actor:** Instructor
2. **Precondition**: The Instructor clicks the “Update Teams” Button.
3. **Main success scenario:** 
   1. The Instructor selects class.
   2. The Instructor selects the team line checkbox to update team.
   3. The Instructor selects member names to update team.
   4. The Instructor clicks “Update Team” Button.
   5. The Instructor cannot find the required student name in the list.
   6. The Instructor gets error as the team details are incomplete.

The instructor will click update teams button to reach below screens. Here he will select a team and clicks on “Update Teams” Button.****

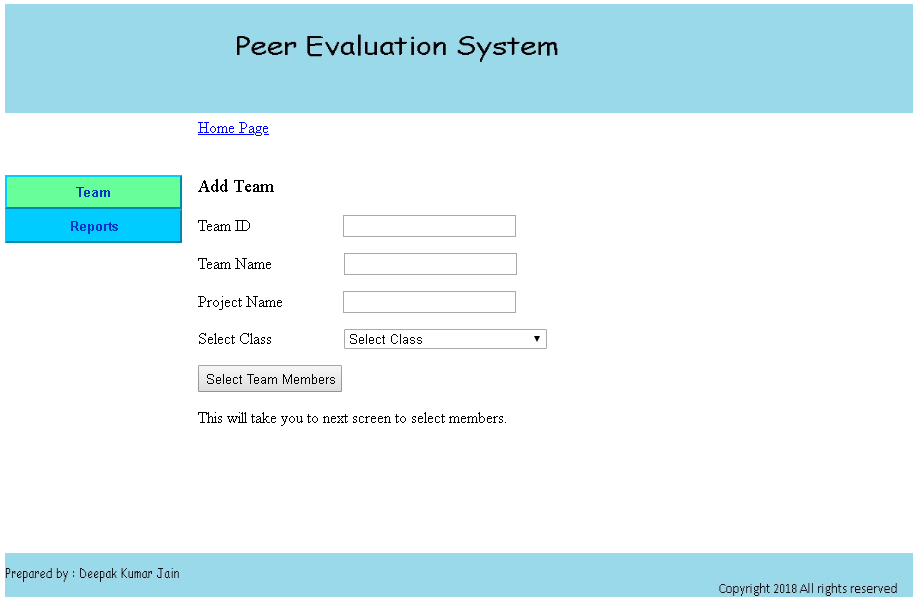
**Use case 7 -** Instructor Adds Teams

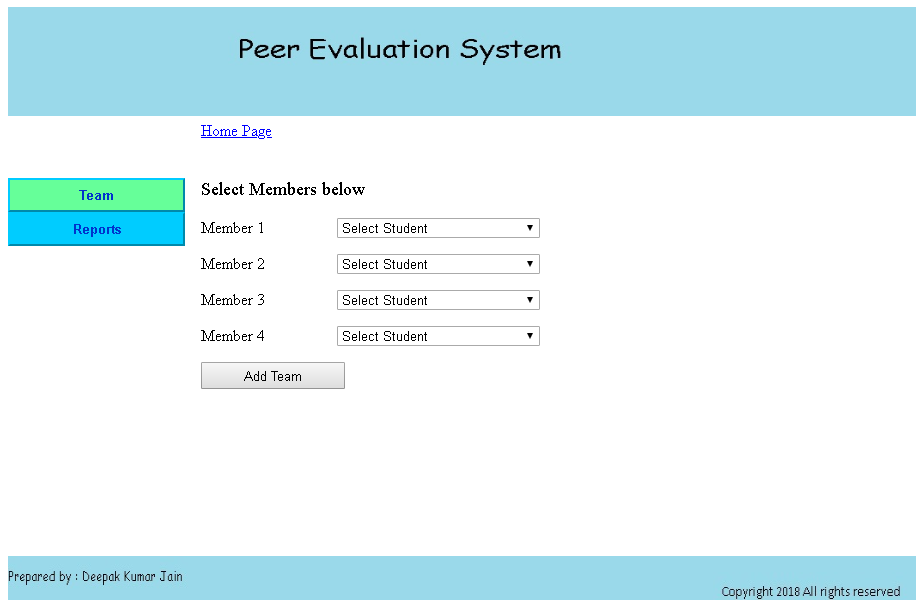
**Main flow:**

1. **Primary actor:** Instructor**.**
2. **Precondition**: The Instructor clicks the “Add New Team” Button.
3. **Main success scenario:** 
   1. The Instructor selects class.
   2. The Instructor clicks “Add Team” Button.
   3. The Instructor enters details (Team ID, Team Name, Project Name and Class) to add team.
   4. The Instructor clicks “Select Team Members” Button.
   5. The Instructor select up to 4 Team Members.
   6. The Instructor clicks “Add Team” Button.
   7. The Team is added in the database.

**Alternative flow:**

1. **Primary actor:** Instructor**.**
2. **Precondition**: The Instructor clicks the “Add New Team” Button.
3. **Main success scenario:** 
   1. The Instructor selects class.
   2. The Instructor clicks “Add Team” Button.
   3. The Instructor enters details (Team ID, Team Name, Project Name and Class) to add team.
   4. The Instructor gets error that team details are incomplete when he clicks “Select Team Members” Button.
   5. After rechecking and adding all the details he clicks “Select Team Members” Button.
   6. The Instructor may not find team members in the list.
   7. After member is added to the list, the Instructor clicks “Add Team” Button.
   8. The Instructor gets error as the team details are incomplete.

The Instructor enters details (Team ID, Team Name, Project Name and Class) in below page and click on Select Team Members.

By clicking Select Team Members button he is redirected to below page**.** Here he will choose team members and then click on Add Team button to add a team. 

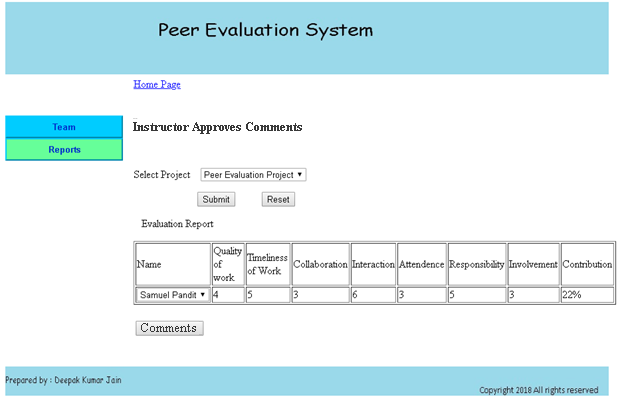
**Use case 8 -** Instructor Approves comments

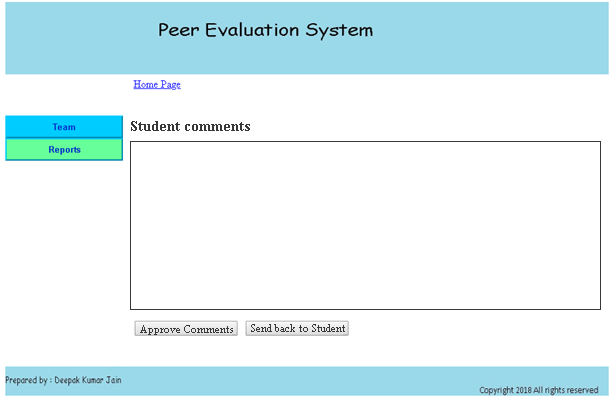
**Main flow:**

1. **Primary actor:** Instructor**.**
2. **Precondition**: The Instructor clicks the “View Assessment” Button.
3. **Main success scenario:** 
   1. The Instructor selects Class.
   2. The Instructor selects Project.
   3. The Instructor clicks “View Assessment” Button.
   4. The Instructor approves comments.

**Alternative flow:**

1. **Primary actor:** Instructor**.**
2. **Precondition**: The Instructor clicks the “View Assessment” Button.
3. **Main success scenario:** 
   1. The Instructor selects Class.
   2. The Instructor selects Project.
   3. The Instructor clicks “View Assessment” Button.
   4. The Instructor finds inappropriate comments, so he will click “Send to Student” button.
   5. The Instructor provides his comments and sends back to student.

In below screen the instructor clicks on “Comments” Button and it opens a student comments screen.

In below screen the instructor clicks the “Approve Comment” button if he finds the comment is ok, otherwise he will click “Send back to Student” button to send back.

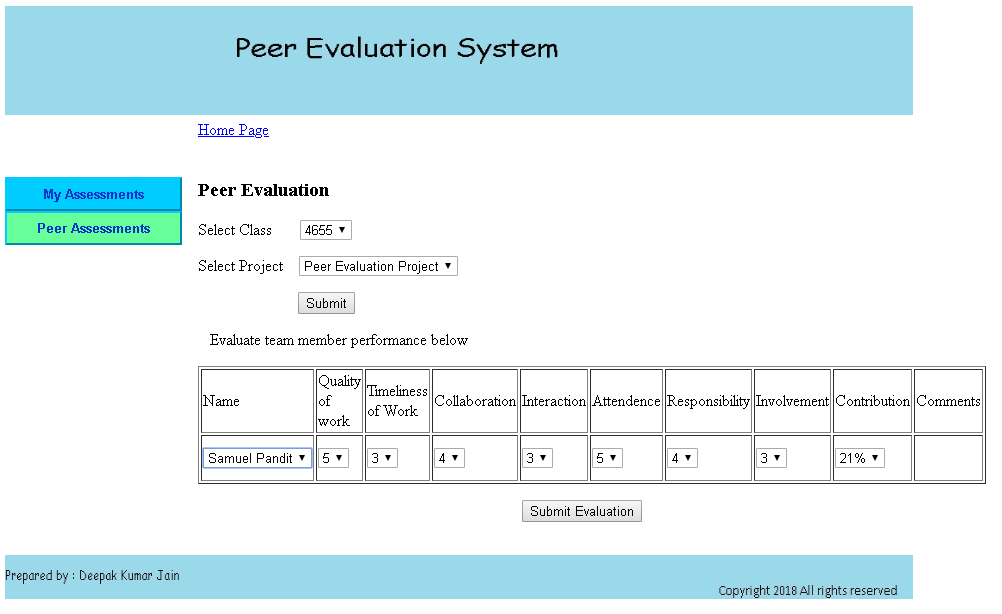
**Use case 9 -** Team member makes assessments

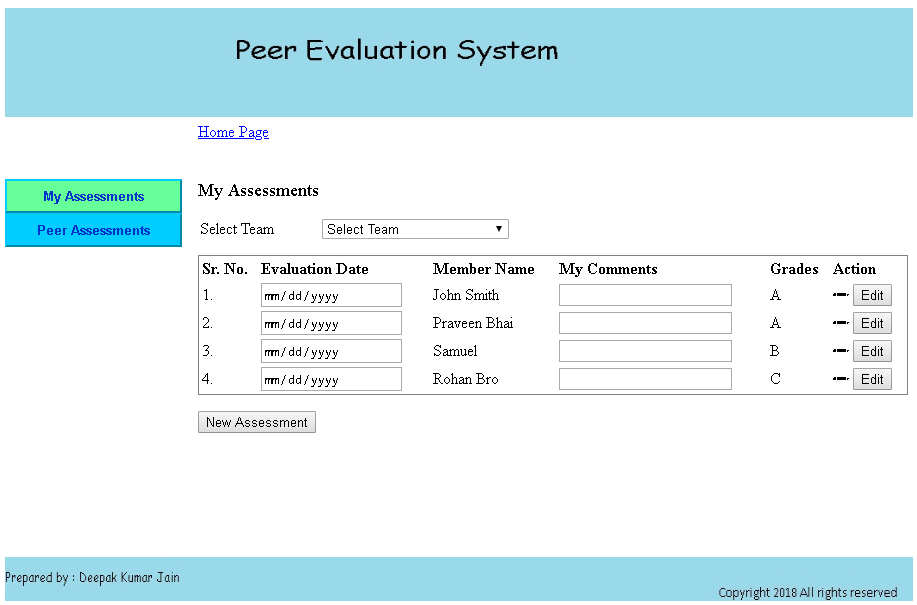
**Main flow:**

1. **Primary actor:** Team member**.**
2. **Precondition**: The Team member clicks the “Peer Assessment” Button.
3. **Main success scenario:** 
   1. The Team member selects Class.
   2. The Team member selects Project.
   3. The Team member clicks “Submit” Button.
   4. The Team member selects a member name from the list.
   5. The Team member selects rating for each parameter.
   6. The Team member provides their comments in last column.
   7. The Team member clicks “Submit Evaluation” Button.

**Alternative flow:**

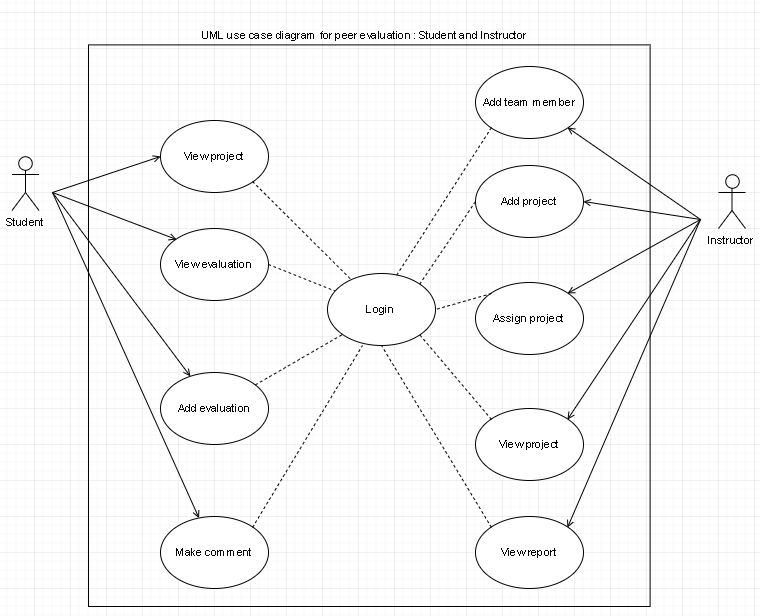
1. **Primary actor:** Team member**.**
2. **Precondition**: The Team member clicks the “Peer Assessment” Button.
3. **Main success scenario:** 
   1. The Team member selects Class.
   2. The Team member not able to select project as it is not in the list.
   3. The Team member clicks “Submit” Button once error is solved.
   4. The Team member selects a member name from the list.
   5. The Team member not able to select member as it is not in the list.
   6. The Team member selects rating for each parameter once error is solved.
   7. The Team member provides their comments in last column.

The Team member selects Class and Project and click submit button. Then he will select name and gives his opinions and gives his comments and clicks on Submit Evaluation button

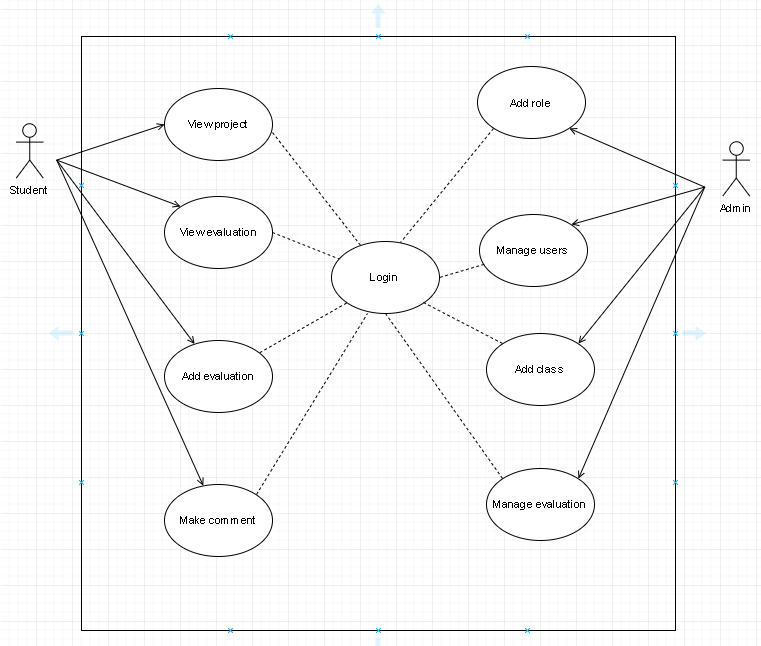
The team member will see assessments done by him in the screen below by clicking “My Assessments” button. 

# **Use Case Diagrams**

## **Use Case Diagram: Student and Instructor**



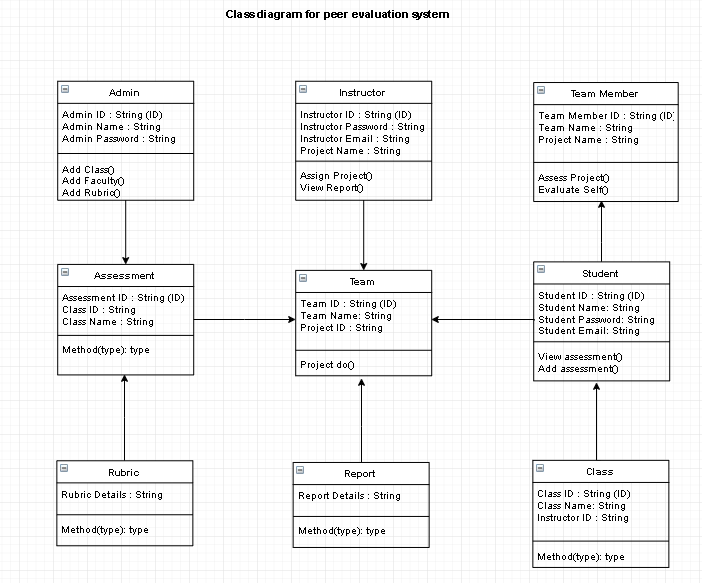
## **Use Case Diagram: Student and Admin**



## **Things to keep in mind for use case diagram**

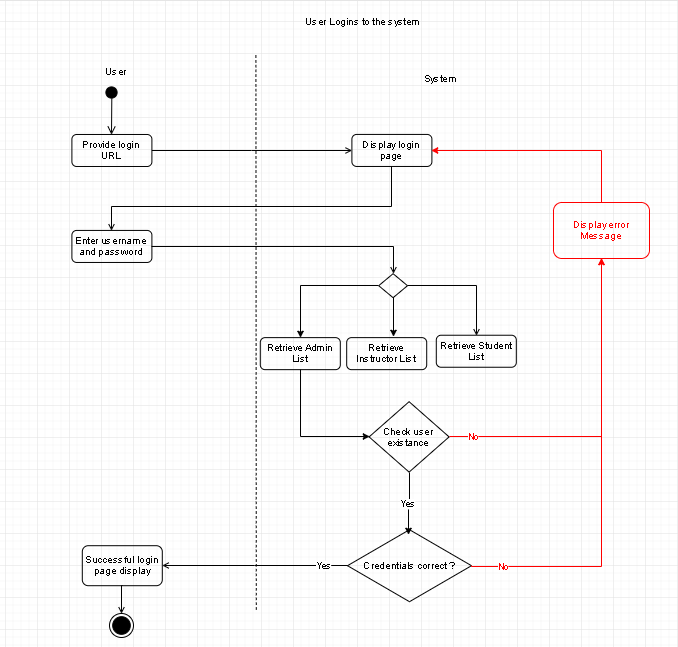
* After the functional descriptions, prototypes and use-cases have been refined, the use case description also has to be changes accordingly.
* Check that the use cases should match the prototypes that have been designed.
* All the changes have to be incorporated in the use case diagram.

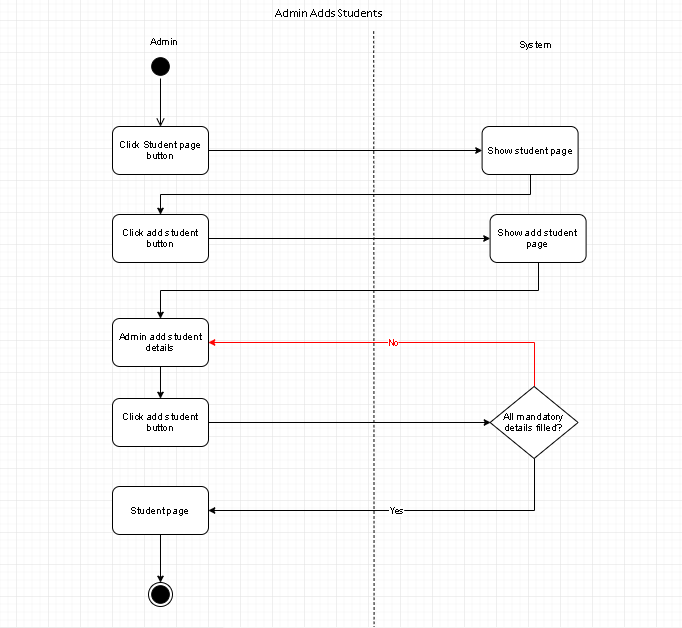
# **Analysis Class diagram**

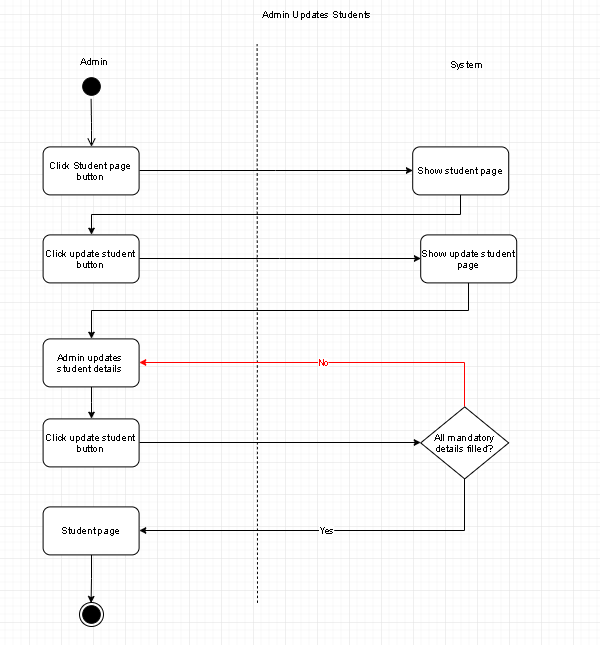


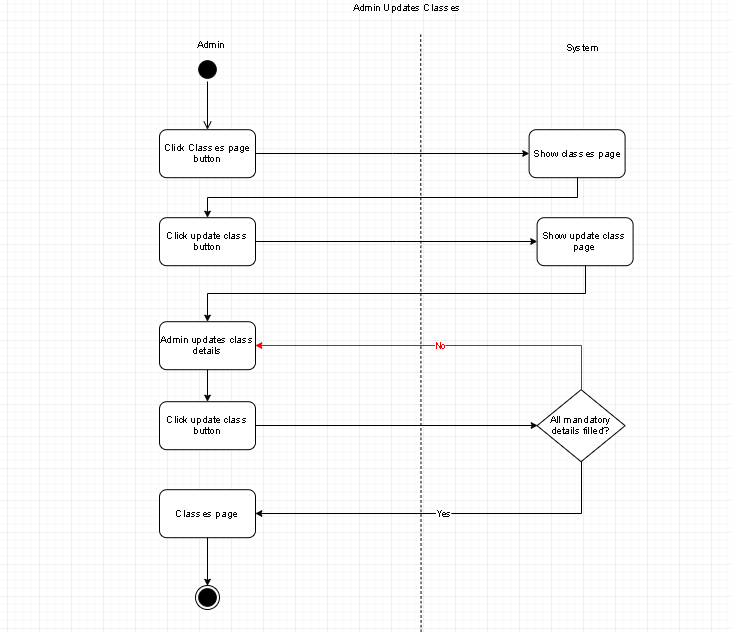
# **Activity Diagrams**

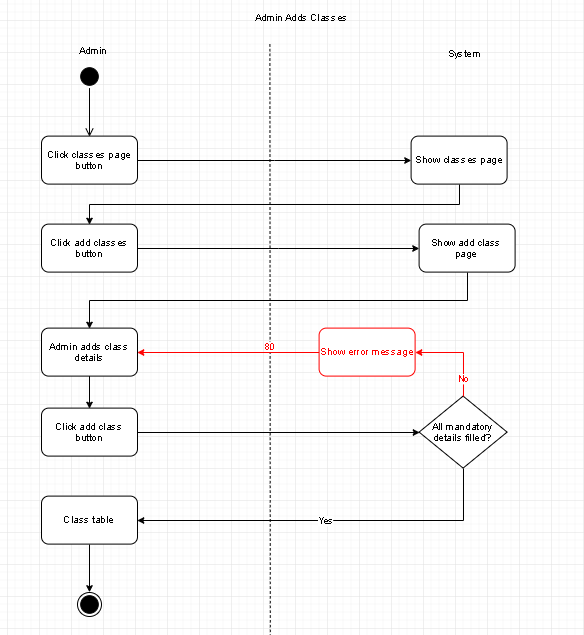
Starting below are the Activity diagrams for each use case

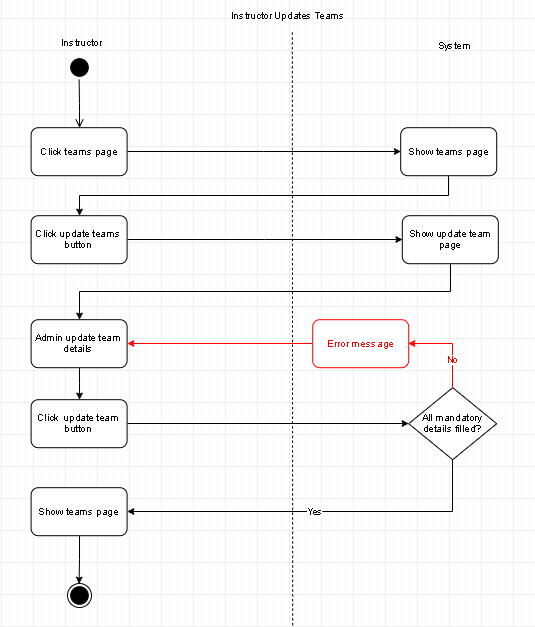


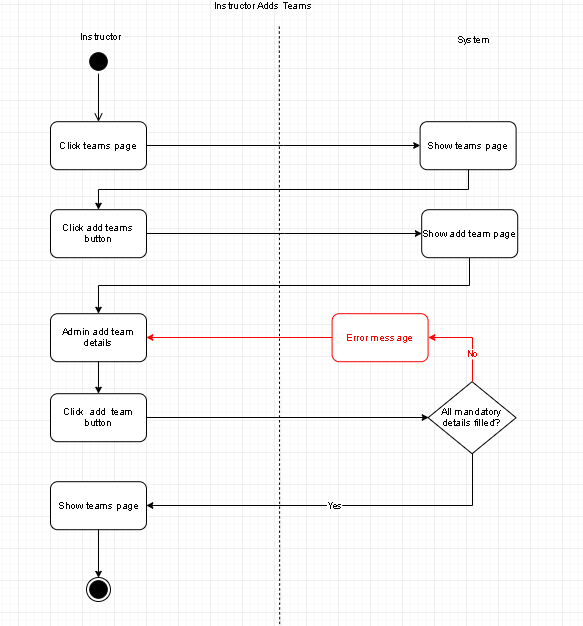


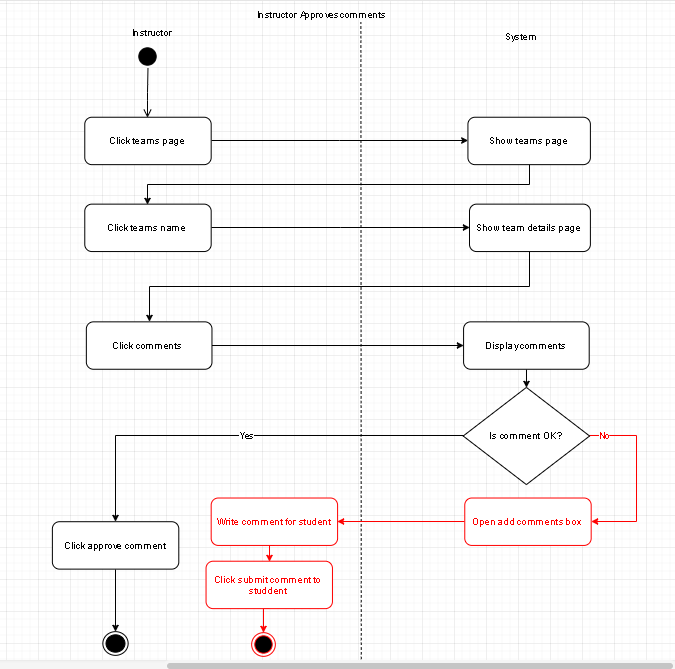


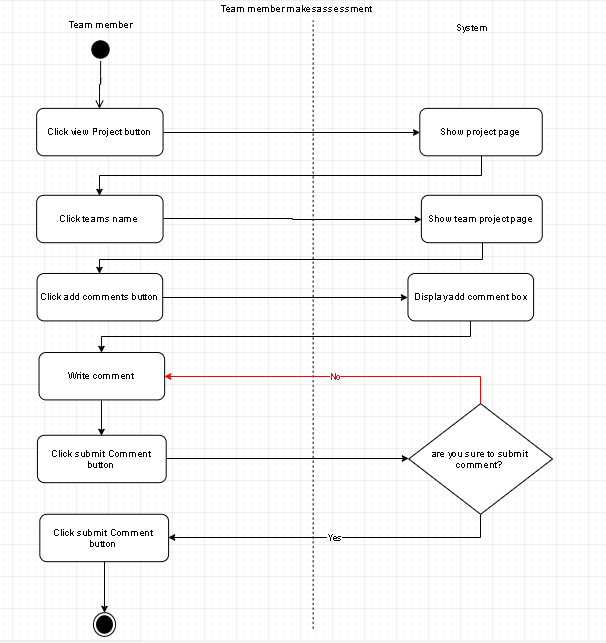






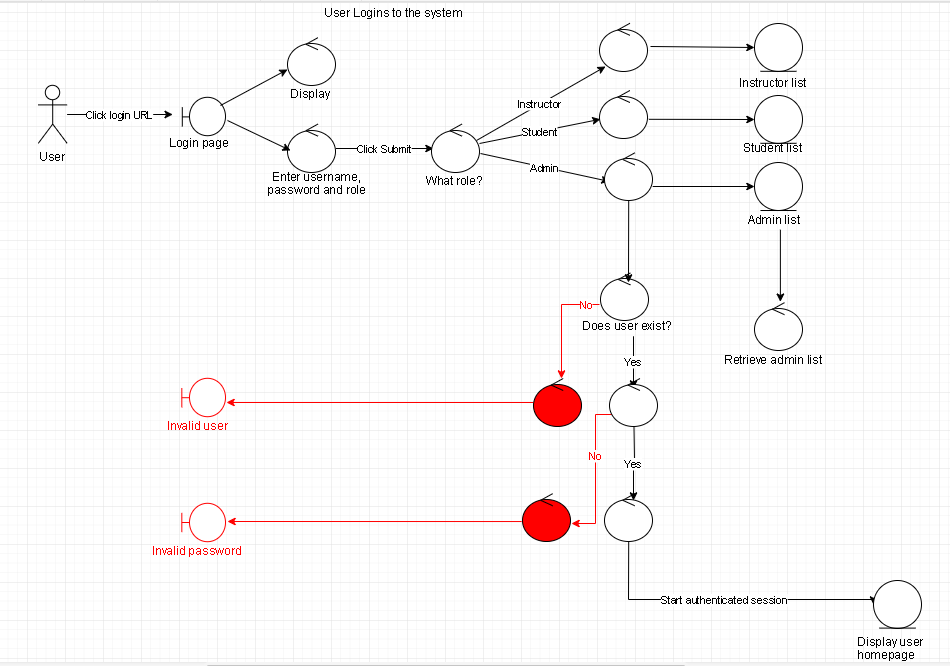


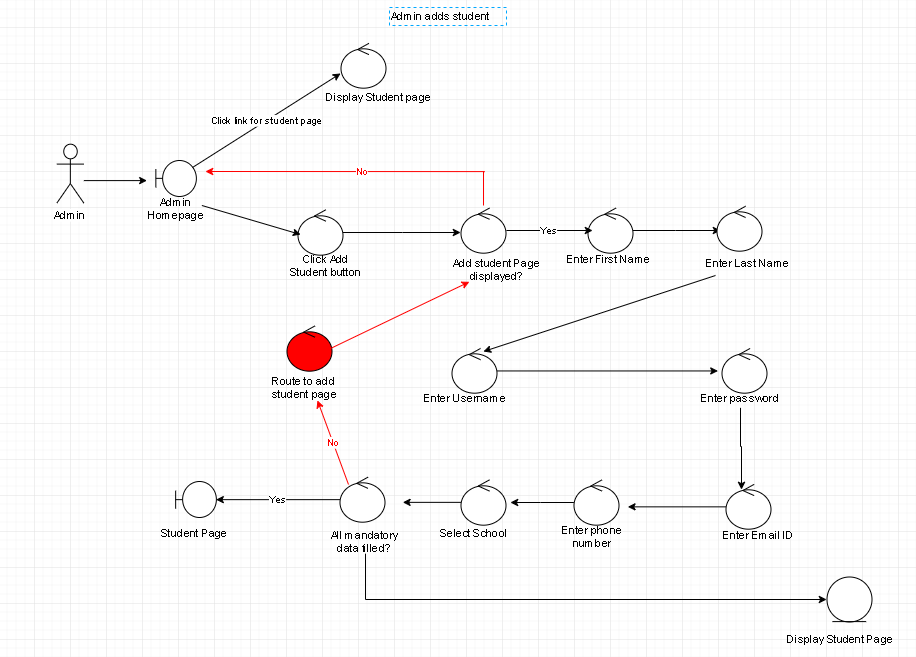


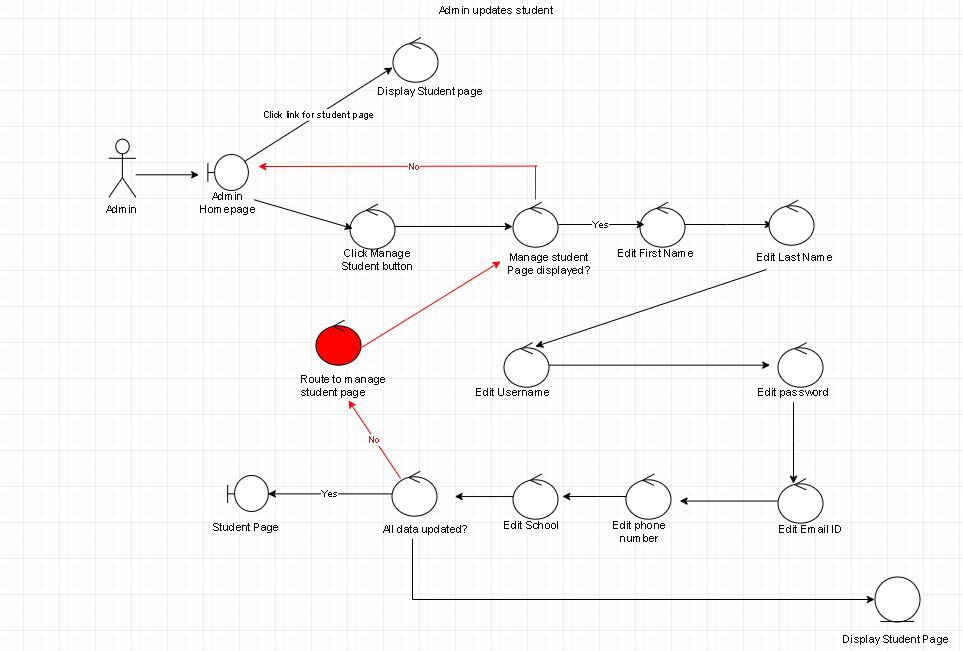


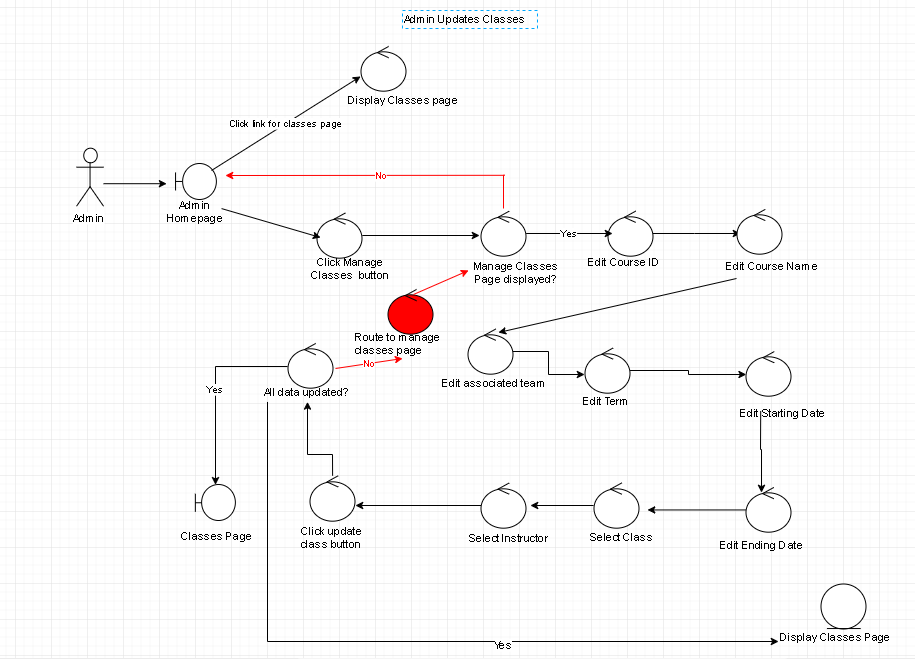
# **Robustness Diagrams**

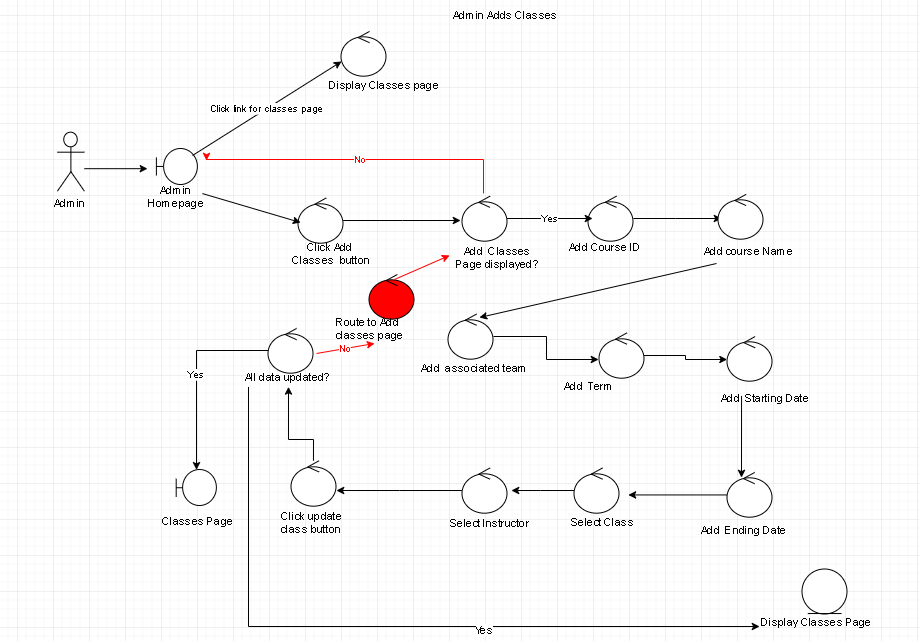
Starting below are the Robustness diagrams for each use case

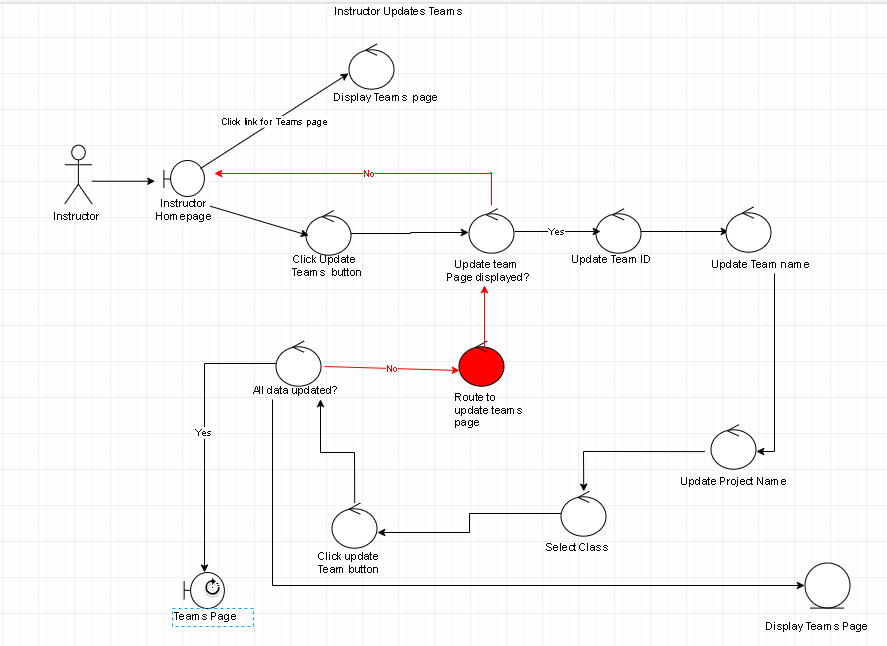


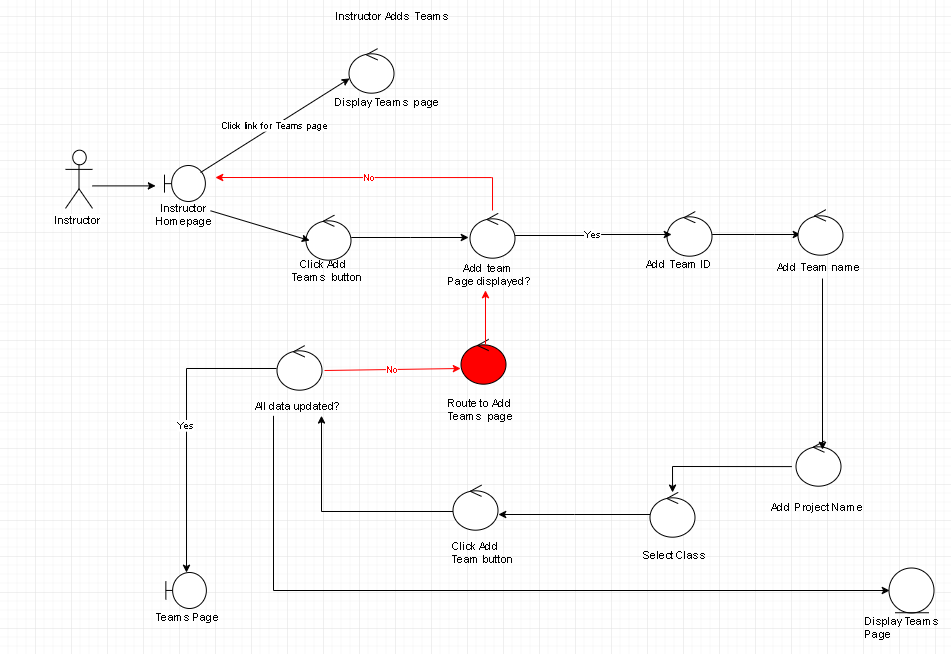


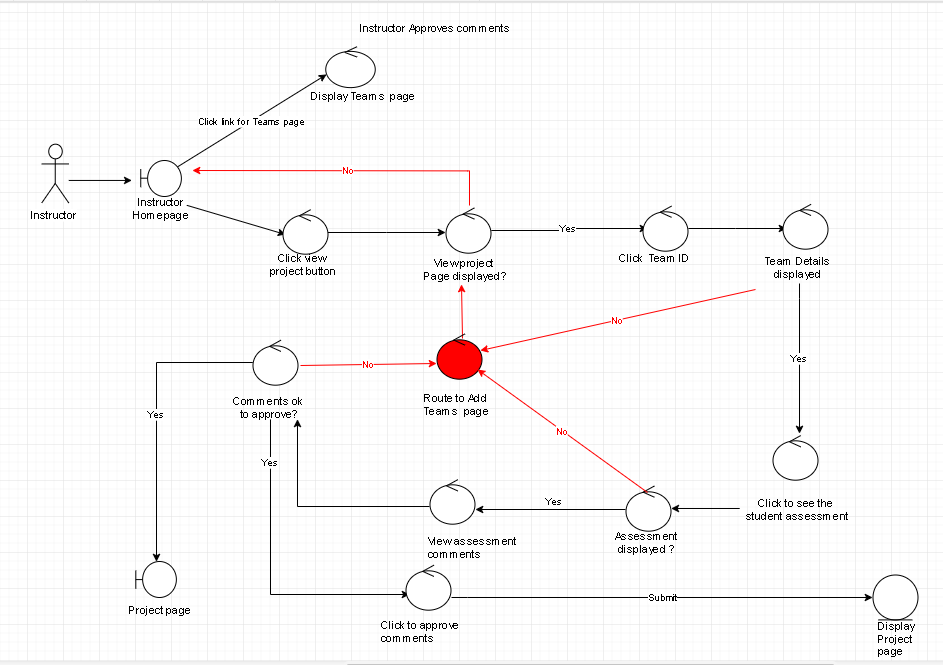


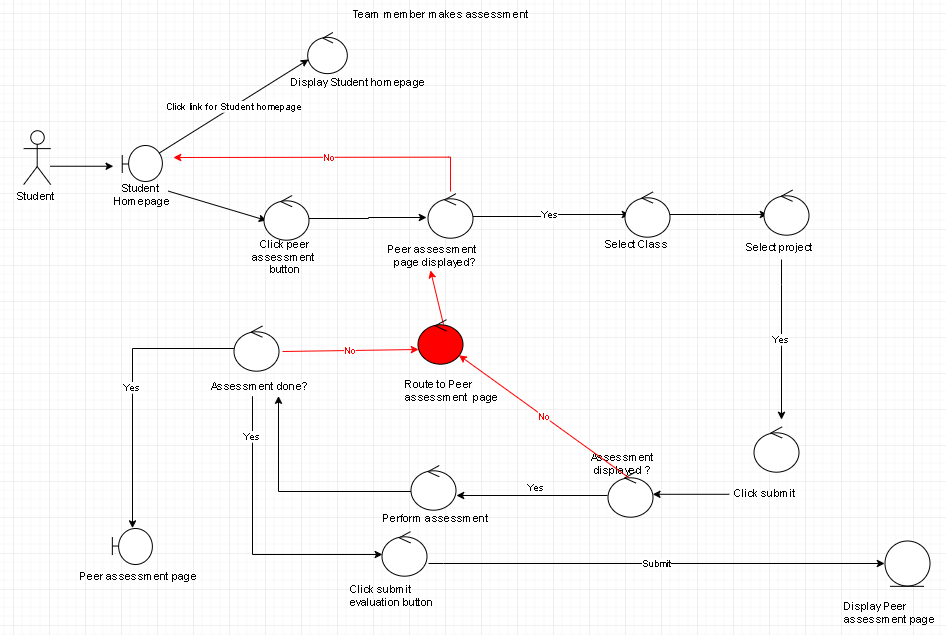












# **References**

1. Durham, Jeff. "Why is Happiness So Important?" Lifecoach Expert. Last modified September 16, 2019. <http://www.lifecoachexpert.co.uk/whyishappinesssoimportant.html>.
2. Finn, ED. "What Makes a Country Happy?" The Independent. Last modified November 24, 2017. <https://theindependent.ca/2017/11/24/makes-country-happy/>.