**Team Name:**

The Coding Company

**Team Members:**

Daniel Martinez

Francisco Fierro

Kyle Pamintuan

Matthew Le

**Team Leader:**

Matthew Le

**Requirements Analysis Document**

1. **Introduction**
   1. **Purpose**: The client, Michael Smith, is a CEO of an education-based company who is in need of an interactive online chat room app that can be used in school. Mr. Smith wants an app that can serve as a platform where instructors are able to hold classroom lectures and where all students will feel encouraged to participate. Also, he wants an app that can make it easier for teachers to take roll and track participation points.
   2. **Goal**: The most important aspect of the system should be to help instructors and students interact more. The client believes that the more confident students are to ask questions, the more they are willing to learn and succeed. This is accomplished through a chat room app that lets students post comments/messages anonymously. The secondary most important aspect of the system should be to help make teaching duties easier for the instructor. Finally, the overall goal is to improve how the classroom operates as a whole. That is, instructors are able to teach better and students are able to learn better.
   3. **Scope of the System**:
      1. The system should provide a way for instructors to create an online account where he/she is able to manage course and student data. Also, the teacher should be able to create chatroom sessions for lectures and be given features to help he/she administer those sessions as needed.
      2. For students, the system should provide a way to let them create an online account, add/drop a course, and access course chat room sessions created by the instructor. Also, the student should be able to interact with instructor/students easily and have full anonymity for all communication in the chat room. All student comments/posts are to be filtered with a profanity check of some kind to prevent inappropriate behavior.
      3. Lastly, all functionality of the system should be done online. Access to the online software should be accomplished via web app. Both apps should have notification capabilities.
   4. **Objectives and Fulfillment Criteria**: The system requires functional and nonfunctional requirements be met and that the following are accurate and timely:
      1. **Instructor**:
         1. Register as an Instructor
         2. Enter/Delete/Update account information
         3. Edit account settings
         4. Create a course
         5. View/Edit course list
         6. Add a student
         7. View/Edit student list
         8. Create/Delete a chat room session
         9. Post a message/comment
         10. Kick a student
         11. Delete a post
      2. **Student**:
         1. Register as a Student
         2. Enter/Delete/Update account information
         3. Edit account settings
         4. Add/Drop a course
         5. Access chat room session
         6. Post a message/comment
         7. Upvote/Downvote a message/comment
      3. **System**:
         1. Profanity check for messages/comments
         2. Notifications
      4. **Synopsis**: The summaries and detailed descriptions of the current and proposed systems as well as the functional and nonfunctional requirements will be addressed to in this documentation. Various possible scenarios, use cases, mockups or reports and software will be included as well.
2. **Current System**
   1. **BeachBoard**: Beachboard is an online portal through which instructors can share assignments, deadlines, display grades and provide feedback, and provide additional resources for a course. Each course also has a discussion section on which instructors may create threads for students to discuss a topic, or share and provide feedback to their work.
   2. **Instructors**: Instructors are able to create discussion boards and threads, post on the threads, and reply to students.
   3. **Students**: Students can create threads on the discussion boards created by the instructor, post on the threads created by the instructor or another student. They are also able to reply to posts created by another student, or the instructor.
3. **Proposed System**
   1. **Overview**
      1. The system should allow registration of instructors/students that don’t already exist in the system. Also, user should be able to manage account info or account settings.
      2. The system should allow instructors to manage courses, students in the courses, and chat room sessions.
      3. The system should allow students to manage their registered courses and access chat room sessions.
      4. Instructors and students should be able to communicate in the chat room sessions by posting messages/comments approved by profanity checks.
      5. The system should be able to send necessary notifications to Instructor/Student
   2. **Functional Requirements**
      1. **Instructor removes a student**
         1. For an instructor to remove a student, they must first be registered and successfully logged into their instructor account.
         2. The instructor must have also created a course and a chat room session.
         3. For a student to be removed from the chat room session, a student must have already be registered and successfully logged into their student account.
         4. The student must be enrolled in the instructor’s course.
         5. The student must also have access to the chat room by entering in the correct access code.
         6. Once all the above requirements are met, the instructor can then select the option to kick a student and choose the student to remove from the chat room.
      2. **Instructor deletes a comment**
         1. For an instructor to delete a comment, they must first be registered and successfully logged into their instructor account.
         2. The instructor must have also created a course and a chat room session.
         3. For a student’s comment to be deleted, a student must have already be registered and successfully logged into their student account.
         4. The student must be enrolled in the instructor’s course.
         5. The student must also have access to the chat room by entering in the correct access code.
         6. The student must have posted a comment that was successfully approved by any profanity checks.
         7. Once all the above requirements are met, the instructor can then select the option to view the student’s comment and remove the comment from the chat room.
      3. **Student enters a chat room**
         1. Firstly, an instructor must have already registered and successfully logged into their instructor account.
         2. The instructor must have already created a course and chat room session for the course.
         3. Now, for a student to enter the chat room session, they must have already registered and successfully logged into their student account.
         4. The student must be enrolled in the course relating to the chat room.
         5. The student must also have access to the chat room by entering in the correct access code.
         6. Once all the requirements above are met, the student can then enter the chatroom.
      4. **Student picks a name**
         1. Firstly, an instructor must have already registered and successfully logged into their instructor account.
         2. The instructor must have already created a course and chat room session for the course.
         3. Now, for a student to enter the chat room session, they must have already registered and successfully logged into their student account.
         4. The student must be enrolled in the course relating to the chat room.
         5. The student must also have access to the chat room by entering in the correct access code.
         6. Once all the requirements above are met, the student can then enter the chatroom and choose to pick a name.
      5. **Student votes on comment and receive points**
         1. Firstly, an instructor must have already registered and successfully logged into their instructor account.
         2. The instructor must have already created a course and chat room session for the course.
         3. Now, at least two students must have already registered and successfully logged into their student accounts.
         4. The students must be enrolled in the course relating to the chat room.
         5. The students must also have access to the chat room by entering in the correct access code.
         6. One student must have successfully posted a comment and been approved by the profanity checks.
         7. Once all the requirements above have been met, the other student can then choose to upvote/downvote the first student’s comment. And by doing so, they receive points for participating in the chat room.
      6. **System blocks messages that contains profanity**
         1. Firstly, an instructor must have already registered and successfully logged into their instructor account.
         2. The instructor must have already created a course and chat room session for the course.
         3. Now, for a student to enter the chat room session, they must have already registered and successfully logged into their student account.
         4. The student must be enrolled in the course relating to the chat room.
         5. The student must also have access to the chat room by entering in the correct access code.
         6. The student can then enter the chatroom and attempt to post a comment.
         7. The comment must contain any type of profanity: bad words, inappropriate words, aggression, harassment, etc.
         8. Once all the requirements above have been met, the system can then block the message that contains profanity. The person posting the message gets an error message.
      7. **User creates an account**
         1. A valid .edu email is required to register an account. An email that has already been used to register another account cannot be used
         2. An account can be designated as either a Student account or Instructor account. The user must choose which account type they want. If they choose an Instructor account, they must go through additional verification steps
         3. Once the email has been provided, a password will also need to be provided. Passwords must contain at least one uppercase letter (A-Z), one lower case letter (a-z), a number (0-9), a special character (!-~), and must be at least 8 characters in length. A warning should be displayed beside the password field if the current entry does not meet these guidelines.
         4. Once both the email and password fields have been provided, the user can confirm their information and the system will check to see if the information and format is correct
         5. The system will send a verification email to the email that was provided.
         6. The user must click on the verification link in the email in order to complete their account registration.
         7. If the email has already been used to register an account, or if the password does not meet the format requirements, an error message is displayed.
      8. **User creates a chatroom**
         1. In order to create a chatroom, the user must first be logged into an Instructor account
         2. The user should choose the “Start a chat room” option and input any optional information
         3. Once the user launches the chat room, an access code is generated and displayed to the user
         4. The access code is required by students in order to enter the chat room
      9. **User posts a message**
         1. The user must be logged into an account and be in a chat room in order to post a message
         2. The user taps on the large text box and types out their message
         3. When the message is completely typed out, the user presses the “send” button
         4. If the message does not contain profanity, it is posted into the chat room
      10. **Instructor views class list**
          1. Each student must be displayed in the following format:

[student name] [student username] [student id]

1. Student name: The student’s full name as registered with the school
2. Student username: The student’s screen name as registered with the system
3. Student ID: The student’s statewide identification number
   * + 1. System must be able to handle a null list of students and indicate that there are not any students in the list
       2. System must be able to handle students without registered accounts/usernames. These students will have “N/A” as their entry for their username.
     1. **System sends notifications**
        1. System must report the triggering event in the notification
        2. For students, system must identify in the notification, the actor that performed the triggering event by their screenname. For instructors, the actor will be identified by their student name and student id.
        3. Notification must persist in the notifications drawer until dismissed by the user
        4. Notification must contain a timestamp (hh:mm [timezone]) of the time it was sent
     2. **User updates account information**
        1. User must be able to provide current username and password
        2. For the information being updated, the new information must not be identical to the old information. If the information matches, a warning should be displayed after the user attempts to submit the changes. And the relevant fields should be cleared.
        3. Passwords must contain at least one uppercase letter (A-Z), one lower case letter (a-z), a number (0-9), a special character (!-~), and must be at least 8 characters in length. A warning should be displayed beside the password field if the current entry does not meet these guidelines.
        4. There will be two fields into which the user must enter the new password. These fields must match to confirm the new password.
        5. Usernames must not contain explicit language, must be at least 6 characters in length, and must be unique.. A warning should be displayed beside the username field if the current entry does not meet these guidelines. The particular requirement not met should be specified in the warning.
        6. Updating a username will require the entry of the current password.
        7. User should be notified of successful updating of information with a confirmation screen.
   1. **Non-Functional Requirements** 
      1. **User interface and human factors:**
4. The application should be in English only.
5. Each button or input text box should be clearly labeled.
6. The user should not have to tap or click more than 4 times in order to get where they want to go.
7. Invalid inputs or error messages should be clearly communicated with the user using distinct colors and/or larger font
8. A message should have the capacity of no less than 140 characters and the amount of characters left in a message should be clearly communicated to the user in real time
   * 1. **Performance:**
9. The app should be able to handle up to 500 chat rooms at any given time, each chat room holding up to 500 students.
10. It should be able to handle all of this while still responding within 10 milliseconds of user input.
11. The system should be able to perform at this level from 8am to 8pm every day from Monday to Friday
12. Any e-mail sent to a user by the system should be delivered within 5 minutes
    * 1. **Error handling and extreme conditions:**
13. The application should be able to handle invalid inputs
14. Under extreme conditions, the system should be able to recover within 5 minutes
15. In order to avoid spamming, a user should not be able to post more than 15 messages in a 1 minute span of time
    * 1. **Hardware:**
16. The system should be able to run on multiple smartphone configurations
    * 1. **Documentation:**
17. Each input textbox should clearly communicate the valid format it requires
18. A help button should be available on every menu that explains to the user what they are seeing and how to proceed
    * 1. **Training:**
19. There should be one administrator trained on how to reboot the system in case of failure
    * 1. **Backup:**
20. The last 3 chat rooms of every professor will be backed up on the system memory
    * 1. **System Maintenance and Setup:**
21. Regular maintenance on the system will be performed once a week
22. A one-time installation is required in order to get the system running
    * 1. **Security:**
23. All login information should be secure and encrypted with at least 128 bits of security
    1. **Scenarios**
       1. **Instructor creates account**

|  |  |
| --- | --- |
| Scenario Name: | a. Instructor creates account |
| Actors: | Instructor |
| Flow of Control: | 1. Instructor inputs information including name, .edu email and college name 2. System sends instructor a verification email 3. Instructor clicks on verification link 4. Account is verified     Instructor name: Brian Hatfield  Instructor e-mail: brianhatfield@csulb.edu  College name: Long Beach State University |



|  |  |
| --- | --- |
| Scenario Name: | b. Instructor creates account |
| Actors: | Instructor |
| Flow of Control: | 1. Instructor inputs information including name, .edu email and college name 2. System sends instructor a verification email 3. Instructor clicks on verification link 4. Account is verified     Instructor name: Bruce Simpson  Instructor e-mail: brucesimpson@ucla.edu  College name: University of California, Los Angeles |



|  |  |
| --- | --- |
| Scenario Name: | c. Instructor creates account |
| Actors: | Instructor |
| Flow of Control: | 1. Instructor inputs information including name, .edu email and college name 2. System sends instructor a verification email 3. Instructor clicks on verification link 4. Account is verified     Instructor name: Ezra Martinez  Instructor e-mail: ezramartinez@ua.edu  College name: University of Alabama |

* + 1. **Instructor creates a chatroom**

|  |  |
| --- | --- |
| Scenario Name: | a. Instructor Creates a Chatroom |
| Actors: | Instructor |
| Flow of Control: | 1. Instructor accesses application, logs into it, and chooses to create a chatroom. 2. System prompts for class information. 3. System loads appropriate class and time information to generate a chatroom and maintain the chatroom during the appropriate times. 4. System automatically generates access code for chatroom. 5. Access code is displayed on instructor's screen     Instructor Name: Frank Murgolo  Course: CECS 491A SEM  Section: 03  Course Num: 7771  Class Days: T TH  Class Times: 3:30pm - 4:20pm  Access code: CS82315 |



|  |  |
| --- | --- |
| Scenario Name: | b. Instructor Creates a Chatroom |
| Actors: | Instructor |
| Flow of Control: | 1. Instructor accesses application, logs into it, and chooses to create a chatroom. 2. System prompts for class information. 3. System loads appropriate class and time information to generate a chatroom and maintain the chatroom during the appropriate times. 4. System automatically generates access code for chatroom. 5. Access code is displayed on instructor's screen     Instructor Name: Derek Wilson  Course: CECS 478  Section: 01  Course Num: 7738  Class Days: T TH  Class Times: 8:00pm - 9:15pm  Access code: CS67124 |



|  |  |
| --- | --- |
| Scenario Name: | c. Instructor Creates a Chatroom |
| Actors: | Instructor |
| Flow of Control: | 1. Instructor accesses application, logs into it, and chooses to create a chatroom. 2. System prompts for class information. 3. System loads appropriate class and time information to generate a chatroom and maintain the chatroom during the appropriate times. 4. System automatically generates access code for chatroom 5. Access code is displayed on instructor's screen     Instructor Name: Lorraine Carson  Course: CDFS 319  Section: 15  Course Num 6427  Class Days: M W  Class Times: 5:30pm - 6:45pm  Access code: FS23504 |

* + 1. **Instructor removes a student**

|  |  |
| --- | --- |
| Scenario Name: | a. Instructor removes a student |
| Actors: | Instructor, Student |
| Flow of Control: | 1. Instructor chooses student 2. Instructor chooses “remove student” option 3. System removes selected student from the chatroom 4. Student is banned from chatroom until next lecture     Instructor name: Luis Academia  Student name: Antonio Hilson  Student username: Anonymous841 |



|  |  |
| --- | --- |
| Scenario Name: | b. Instructor removes a student |
| Actors: | Instructor, Student |
| Flow of Control: | 1. Instructor chooses student 2. Instructor chooses “remove student” option 3. System removes selected student from the chatroom 4. Student is banned from chatroom until next lecture     Instructor name: Bill Pax  Student name: Frank Zapato  Student username: Anonymous689 |



|  |  |
| --- | --- |
| Scenario Name: | c. Instructor removes a student |
| Actors: | Instructor, Student |
| Flow of Control: | 1. Instructor chooses student 2. Instructor chooses “remove student” option 3. System removes selected student from the chatroom 4. Student is banned from chatroom until next lecture     Instructor name: Oliver Grand  Student name: Sidney Wells  Student username: Anonymous121 |

* + 1. **Instructor deletes a post**

|  |  |
| --- | --- |
| Scenario Name: | a. Instructor deletes a post |
| Actors: | Instructor |
| Flow of Control: | 1. Instructor selects post 2. Instructor chooses “delete” option 3. System deletes the message from the chatroom     Instructor name: Connor Smith  Student name: Fig Newton  Student message: “Anyone want to do the homework for me? I’ll pay” |



|  |  |
| --- | --- |
| Scenario Name: | b. Instructor deletes a post |
| Actors: | Instructor |
| Flow of Control: | 1. Instructor selects post 2. Instructor chooses “delete” option 3. System deletes the message from the chatroom     Instructor name: Will Jensen  Student name: Oscar Isaac  Student message: “Did you guys watch the game last night?” |



|  |  |
| --- | --- |
| Scenario Name: | c. Instructor deletes a post |
| Actors: | Instructor |
| Flow of Control: | 1. Instructor selects post 2. Instructor chooses “delete” option 3. System deletes the message from the chatroom     Instructor name: Hubert Han  Student name: Sean Curry  Student message: “Here’s a link to the hw answers: http://fakeurl.com” |

* + 1. **Student creates account**

|  |  |
| --- | --- |
| Scenario Name: | a. Student creates account |
| Actors: | Student |
| Flow of Control: | 1. Student downloads app 2. Student chooses “create new account option” 3. Student inputs name, email, password and school 4. System sends verification email to student 5. Student clicks verification link and account is created     Student name: Jason Bourne  Student email:jbhello@gmail.com  Student password: mynameisjason123  Student school: Long Beach State University |



|  |  |
| --- | --- |
| Scenario Name: | b. Student creates account |
| Actors: | Student |
| Flow of Control: | 1. Student downloads app 2. Student chooses “create new account” option 3. Student inputs name, email, password and school 4. System sends verification email to student 5. Student clicks verification link and account is created     Student name: Dustin Miller  Student email:dmillz6789@gmail.com  Student password: theycallmemiller  Student school: Harvard University |



|  |  |
| --- | --- |
| Scenario Name: | c. Student creates account |
| Actors: | Student |
| Flow of Control: | 1. Student downloads app 2. Student chooses “create new account” option 3. Student inputs name, email, password and school 4. System sends verification email to student 5. Student clicks verification link and account is created     Student name: Tim Bosco  Student email: TimBoss41@gmail.com  Student password: password123  Student school: Oregon State University |

* + 1. **Student resets password**

|  |  |
| --- | --- |
| Scenario Name: | a.Student resets password |
| Actors: | Student |
| Flow of Control: | 1. Student opens app 2. Student chooses “reset password” option 3. Student inputs email 4. System sends email to student with password reset link 5. Student inputs new password     Student name: Chris Bacon  Student email:cb49ers@csulb.edu  New password: iforgotmyoldpassword |



|  |  |
| --- | --- |
| Scenario Name: | b. Student resets password |
| Actors: | Student |
| Flow of Control: | 1. Student opens app 2. Student chooses “reset password” option 3. Student inputs email 4. System sends email to student with password reset link 5. Student inputs new password     Student name: Ronald Donald  Student email: ronthedon@gmai.com  New password: newpassword123 |



|  |  |
| --- | --- |
| Scenario Name: | c.Student resets password |
| Actors: | Student |
| Flow of Control: | 1. Student opens app 2. Student chooses “reset password” option 3. Student inputs email 4. System sends email to student with password reset link 5. Student inputs new password     Student name: John Patterson  Student email:itsjohnny@yahoo.com  New password: ilovedogs15 |

* + 1. **Student enters a chatroom**

|  |  |
| --- | --- |
| Scenario Name: | a. Student Enters a Chatroom |
| Actors: | Student, Instructor |
| Flow of Control: | 1. Student accesses application and is prompted for an access code. 2. Instructor shares access code with Student 3. Student enters access code.     Access Code: CE12345 |



|  |  |
| --- | --- |
| Scenario Name: | b. Student Enters a Chatroom |
| Actors: | Student1, Student2, Instructor |
| Flow of Control: | 1. Student1 accesses application and is prompted for an access code. 2. Instructor shares access code with Student2 3. Student2 shares access code with Student1 4. Student1 enters access code.     Access Code: FS56789 |



|  |  |
| --- | --- |
| Scenario Name: | c. Student Enters a Chatroom |
| Actors: | Student, Instructor |
| Flow of Control: | 1. Student accesses application and is prompted for an access code. 2. Instructor broadcasts access code to the classroom. 3. Student enters access code.     Access Code: EE91012 |

* + 1. **Student picks a name**

|  |  |
| --- | --- |
| Scenario Name: | 1. Student picks a name |
| Actors: | Student |
| Flow of Control: | 1. Student gains access to chatroom session 2. Student decides to rename their username 3. Student chooses from a given pool of randomized names     Old Username: Anonymous123  New Username: spongebob |



|  |  |
| --- | --- |
| Scenario Name: | b. Student picks a name |
| Actors: | Student |
| Flow of Control: | 1. Student gains access to chatroom session 2. Student decides to rename their username 3. Student chooses from a given pool of randomized names     Old Username: Anonymous456  New Username: patrickStar |



|  |  |
| --- | --- |
| Scenario Name: | c. Student picks a name |
| Actors: | Student |
| Flow of Control: | 1. Student gains access to chatroom session 2. Student decides to rename their username 3. Student chooses from a given pool of randomized names     Old Username: Anonymous789  New Username: squidward |

* + 1. **Student choose anonymous for name**

|  |  |
| --- | --- |
| Scenario Name: | 1. Student choose anonymous for name |
| Actors: | Student |
| Flow of Control: | 1. Student gains access to chatroom session 2. Student is automatically given anonymous name     Student name: Fred Fries  Username: Anonymous089 |



|  |  |
| --- | --- |
| Scenario Name: | b.Student choose anonymous for name |
| Actors: | Student |
| Flow of Control: | 1. Student gains access to chatroom session 2. Student is automatically given anonymous name     Student name: Luke Walker  Username: Anonymous537 |



|  |  |
| --- | --- |
| Scenario Name: | c. Student choose anonymous for name |
| Actors: | Student |
| Flow of Control: | 1. Student gains access to chatroom session 2. Student is automatically given anonymous name     Student name: Steve Gilroy  Username: Anonymous295 |

* + 1. **Student posts a comment**

|  |  |
| --- | --- |
| Scenario Name: | a. Student posts a comment |
| Actors: | Student |
| Flow of Control: | 1. Student submits a comment into a chatroom 2. System checks for unauthorized content 3. Comment is displayed in chatroom     Student username: Anonymous901  Comment: “Can you elaborate?” |



|  |  |
| --- | --- |
| Scenario Name: | b. Student posts a comment |
| Actors: | Student |
| Flow of Control: | 1. Student submits a comment into a chatroom 2. System checks for unauthorized content 3. Comment is found to contain inappropriate language 4. Student removes inappropriate language and re-submits 5. System checks for unauthorized content 6. Comment is displayed     Student username: spongebob  Comment: “What the hell does that mean?”  New Comment: “What do you mean by that?” |



|  |  |
| --- | --- |
| Scenario Name: | c. Student posts a comment |
| Actors: | Student |
| Flow of Control: | 1. Student submits a comment into a chatroom 2. System checks for unauthorized content 3. Comment is found to contain links to external sites 4. Student removes links and re-submits 5. System checks for unauthorized content 6. Comment is displayed     Student username: Anonymous437  Comment: “Here’s a link to explain multithreading: [www.google.com/search/multithreading](http://www.google.com/search/multithreading)”    New Comment: “Google Search: Multithreading - a technique by which a single set of code can be used by several processors at different stages of execution.” |

* + 1. Student upvotes a comment

|  |  |
| --- | --- |
| Scenario Name: | 1. Student upvotes a comment |
| Actors: | Student1, Student2, Instructor |
| Flow of Control: | 1. Student1 & Student2 gain access to chatroom session 2. Instructor asks a question 3. Student1 posts answer via chatroom (Authorized) 4. Student2 acknowledges by upvoting Student1’s answer     Instructor: “1+1 = ?”  Student1: “2”  \*\*\* Student2 upvotes Student1’s post \*\*\* |



|  |  |
| --- | --- |
| Scenario Name: | b. Student upvotes a comment |
| Actors: | Student1, Student2, Instructor, Student3 |
| Flow of Control: | 1. Student1 & Student2, Student3 gain access to chatroom session 2. Instructor asks a question 3. Student1 posts answer via chatroom (Authorized) 4. Student2 comments on Student1’s post (Authorized) 5. Student3 acknowledges Student2’s comment by upvoting     Instructor: “1+1 = ?”  Student1: “2”  Instructor: “Correct!”  Student2: “How did you get that?”  \*\*\* Student3 upvotes Student2’s comment \*\*\* |



|  |  |
| --- | --- |
| Scenario Name: | c. Student upvotes a comment |
| Actors: | Student1, Student2 |
| Flow of Control: | 1. Student1 & Student2 gain access to chatroom session 2. Student1 posts question on chatroom (Authorized) 3. Student2 acknowledges question by upvoting     Student1: “Professor when is the first midterm?”  \*\*\* Student2 upvotes Student1’s post \*\*\* |

* + 1. **Student downvotes a comment**

|  |  |
| --- | --- |
| Scenario Name: | a. Student downvotes a comment |
| Actors: | Student1, Student2, Instructor |
| Flow of Control: | 1. Student1 & Student2 gain access to chatroom session 2. Instructor asks a question 3. Student1 posts answer via chatroom (Authorized) 4. Student2 disagrees by downvoting Student1’s answer     Instructor: “1+1 = ?”  Student1: “2”  \*\*\* Student2 downvotes Student1’s post \*\*\* |



|  |  |
| --- | --- |
| Scenario Name: | b. Student downvotes a comment |
| Actors: | Student1, Instructor |
| Flow of Control: | 1. Student1 gain access to chatroom session 2. Instructor posts a comment on chatroom (Authorized) 3. Student1 downvotes the Instructor’s comment     Instructor: “How do you guys feel about having the Midterm on Thursday?”  \*\*\* Student1 downvotes Professor Murgolo’s post \*\*\* |



|  |  |
| --- | --- |
| Scenario Name: | c. Student downvotes a comment |
| Actors: | Student1, Student2 |
| Flow of Control: | 1. Student1 & Student2 gain access to chatroom session 2. Student1 posts comment on chatroom (Authorized) 3. Student2 disagrees by downvoting Student1’s post     Student1: “Can we have a midterm this week instead of next week?”  \*\*\* Student2 downvotes Student1’s post \*\*\* |

* + 1. **System blocks messages that contains profanity**

|  |  |
| --- | --- |
| Scenario Name: | a. System blocks messages that contains profanity |
| Actors: | System, StudentA |
| Flow of Control: | 1. StudentA enters comment containing profanity 2. System detects profanity word(s) 3. System prevent message from sending 4. System sends warning message to Student 5. System logs warning count |



|  |  |
| --- | --- |
| Scenario Name: | b. System blocks messages that contains profanity |
| Actors: | System, StudentB |
| Flow of Control: | 1. StudentB enters comment containing profanity 2. System detects profanity word(s) 3. System prevent message from sending 4. System sends warning message to Student 5. System logs warning count |



|  |  |
| --- | --- |
| Scenario Name: | c. System blocks messages that contains profanity |
| Actors: | System, StudentC |
| Flow of Control: | 1. StudentC enters comments containing profanity 2. System detects profanity word(s) 3. System prevent message from sending 4. System sends warning message to Student 5. System logs warning count |

* + 1. **System warns student for using profanity**

|  |  |
| --- | --- |
| Scenario Name: | a. System warns student for using profanity |
| Actors: | System, StudentA |
| Flow of Control: | 1. System detected comment containing profanity 2. System prevent message from sending 3. System sends warning message to student     System: “This comment contains inappropriate wording. This is your first warning.” |



|  |  |
| --- | --- |
| Scenario Name: | b. System warns student for using profanity |
| Actors: | System, StudentA |
| Flow of Control: | 1. StudentA continues to use profanity 2. System detected comment containing profanity 3. System prevent message from sending 4. System sends warning message to student     System: “This comment contains inappropriate wording. This is your second warning.” |



|  |  |
| --- | --- |
| Scenario Name: | c. System warns student for using profanity |
| Actors: | System, StudentA |
| Flow of Control: | 1. StudentA continues to use profanity 2. System detected comment containing profanity 3. System prevent message from sending 4. System sends warning message to student     System: “This comment contains inappropriate wording. This is your third warning.” |

* + 1. **Instructor views class list**

|  |  |
| --- | --- |
| Scenario Name: | a. Instructor views spectators |
| Actors: | Instructor |
| Flow of Control: | 1. Instructor selects to view menu options 2. System displays menu 3. Instructor selects option to view spectators 4. System displays list of spectator students     Instructor: Frank Murgolo  Course: CECS 491A SEM  Course Num: 7771  Students:  Francisco Fierro  Matthew Le  Kyle Pamintuan  Daniel Martinez |



|  |  |
| --- | --- |
| Scenario Name: | b. Instructor views spectators |
| Actors: | Instructor |
| Flow of Control: | 1. Instructor selects to view menu options 2. System displays menu 3. Instructor selects option to view spectators 4. System displays list of students     Instructor: Ratana Ngo  Course: CECS 326 SEM  Course Num: 7823  Students:  Dania Wareh |



|  |  |
| --- | --- |
| Scenario Name: | c. Instructor views spectators |
| Actors: | Instructor |
| Flow of Control: | 1. Instructor selects to view menu options 2. System displays menu 3. Instructor selects option to view spectators 4. System displays lack of students     Instructor: Lorraine Carson  Course: CDFS 319  Course Num: 6427  Students:  None |

* + 1. **Student receives notifications**

|  |  |
| --- | --- |
| Scenario Name: | a. Student receives notification |
| Actors: | Student1, Student2 |
| Flow of Control: | 1. Student1 replies to a post made by Student2 2. System notifies Student2 of reply     Student1 comment: “How’s it going guys?”  Student 2 reply: “It’s going well”  Notification: "A classmate has replied to your comment!” |



|  |  |
| --- | --- |
| Scenario Name: | b. Student receives notification |
| Actors: | Student, Instructor |
| Flow of Control: | 1. Instructor replies to a comment made by Student 2. System notifies Student of reply     Student1 comment: “Is the hw hard?”  Student 2 reply: “Not too bad”  Notification: "A classmate has replied to your comment!” |



|  |  |
| --- | --- |
| Scenario Name: | c. Student receives notification |
| Actors: | Student, Instructor |
| Flow of Control: | 1. Instructor kicks Student out of chatroom 2. Student receives notification of expulsion from chatroom     Student1 comment: “Anyone have the notes?”  Student 2 reply: “Yea I’ll email them to you”  Notification: "A classmate has replied to your comment!” |

* 1. **Use Cases**
     1. **Instructor removes a student**

|  |  |
| --- | --- |
| Use-case Name: | Instructor removes a student |
| Actors: | Instructor, Student |
| Pre-Conditions: | 1. Internet connection 2. Instructor logged onto account 3. Student logged onto account 4. Instructor created chatroom session 5. Student accessed chatroom |
| Flow of Control: | 1. Instructor creates chatroom session 2. Student accesses chatroom 3. Instructor selects a student 4. Instructor chooses option to kick student from chatroom |
| Post-Conditions: | 1. Student no longer has access to chatroom |
| Error-Conditions: | 1. Accounts not yet created 2. Account username or password entered incorrectly 3. Access code entered incorrectly |
| Non-Functional Requirements: | See Section 3.3 Non-Functional Requirements |

* + 1. **Instructor deletes a post**

|  |  |
| --- | --- |
| Use-case Name: | Instructor deletes a comment |
| Actors: | Instructor, Student |
| Pre-Conditions: | 1. Internet connection 2. Instructor logged onto account 3. Student logged onto account 4. Instructor created chatroom session 5. Student access chatroom |
| Flow of Control: | 1. Instructor creates chatroom session 2. Student access chatroom 3. Student posts on chatroom 4. Instructor select a comment 5. Instructor chooses option to delete the comment |
| Post-Conditions: | 1. Student’s comment is deleted and can no longer be seen in the chatroom |
| Error-Conditions: | 1. Accounts not yet created 2. Account username or password entered incorrectly 3. Access code entered incorrectly 4. Student’s post is denied because of profanity |
| Non-Functional Requirements: | See Section 3.3 Non-Functional Requirements |

* + 1. **Student enters a chatroom**

|  |  |
| --- | --- |
| Use-case Name: | Student enters a chatroom |
| Actors: | Instructor, Student |
| Pre-Conditions: | 1. Internet connection 2. Instructor logged onto account 3. Student logged onto account 4. Instructor created chatroom session 5. Student does not yet have access to the chatroom |
| Flow of Control: | 1. Instructor creates a chatroom session 2. Student clicks add class 3. Student finds the chatroom the instructor created 4. Student enters access code from instructor 5. Student has access to chatroom |
| Post-Conditions: | 1. Student now has access and can enter the chatroom |
| Error-Conditions: | 1. Accounts not yet created 2. Account username or password entered incorrectly 3. Access code entered incorrectly |
| Non-Functional Requirements: | See Section 3.3 Non-Functional Requirements |

* + 1. **Student picks a name**

|  |  |
| --- | --- |
| Use-case Name: | Student picks a name |
| Actors: | Student |
| Pre-Conditions: | 1. Internet connection 2. Created account 3. Logged onto account 4. Have access to chatroom |
| Flow of Control: | 1. Student access the chatroom 2. Student picks a name from a list of pre-made names |
| Post-Conditions: | 1. Student enter chatroom with selected name |
| Error-Conditions: | 1. No internet connection 2. Student have not made account 3. Student does not have access code to chatroom |
| Non-Functional Requirements: | See Section 3.3 Non-Functional Requirements |

* + 1. **Student votes on comment and receive points**

|  |  |
| --- | --- |
| Use-case Name: | Student votes on comment and receive points |
| Actors: | Student |
| Pre-Conditions: | 1. Internet connection 2. Student made account 3. Student access chatroom 4. Student does not use profanity |
| Flow of Control: | 1. Student enters chatroom 2. Student type in comment area and enters it into the chatroom 3. Student sees other comments 4. Student choose either the up arrow bottom or the down arrow button next to the comments |
| Post-Conditions: | 1. Student receives points depending on the type of votes they received on the comment |
| Error-Conditions: | 1. No internet connection 2. Student uses profanity 3. upvote/downvote system doesn’t work |
| Non-Functional Requirements: | See Section 3.3 Non-Functional Requirements |

* + 1. **System blocks messages that contains profanity**

|  |  |
| --- | --- |
| Use-case Name: | System blocks contains that contains profanity |
| Actors: | System |
| Pre-Conditions: | 1. Internet connection 2. Profanity Check System list implemented 3. Student enter profanity in comment area |
| Flow of Control: | 1. Student enter a comment 2. System check comment if it contains profanity 3. Comment contains profanity 4. System blocks comment from displaying 5. System asks student to enter comment without profanity |
| Post-Conditions: | 1. Comment that enter containing profanity does not go through 2. Student has to re-enter comment |
| Error-Conditions: | 1. Student found a way to trick the profanity check system 2. Profanity check system contain bugs |
| Non-Functional Requirements: | See Section 3.3 Non-Functional Requirements |

* + 1. **User creates an account**

|  |  |
| --- | --- |
| Use-case Name: | User creates an account |
| Actors: | Instructor or Student |
| Pre-Conditions: | 1. Internet connection 2. E-mail must not be registered to another Speak Up account |
| Flow of Control: | 1. User goes to web app and chooses the “create account” option 2. User chooses either “Student account” or “Instructor account” 3. User enters e-mail in the “Enter e-mail” text box 4. User creates password in the “Enter password” text box 5. User chooses “confirm” option 6. System sends verification e-mail to user 7. User clicks verification link 8. System verifies and completes user’s account |
| Post-Conditions: | 1. A new account is created for the user |
| Error-Conditions: | 1. E-mail has already been used to create an account 2. Password does not meet format requirements |
| Non-Functional Requirements: | See Section 3.3 Non-Functional Requirements |

* + 1. **User creates a chatroom**

|  |  |
| --- | --- |
| Use-case Name: | User creates a chatroom |
| Actors: | Instructor |
| Pre-Conditions: | 1. Internet connection 2. User is logged into an Instructor account |
| Flow of Control: | 1. User chooses “Start a chatroom” option 2. User has option to title chatroom using the “Title (optional)” text box 3. System generates access code 4. Access code is displayed to user |
| Post-Conditions: | 1. A chatroom is created 2. An access code is created and in use |
| Error-Conditions: | 1. User is using a student account instead of instructor account |
| Non-Functional Requirements: | See Section 3.3 Non-Functional Requirements |

* + 1. **User posts a message**

|  |  |
| --- | --- |
| Use-case Name: | User posts a message |
| Actors: | User |
| Pre-Conditions: | 1. Internet connection 2. User is logged into a valid account 3. User is in a chat room |
| Flow of Control: | 1. User clicks text box titled “Message” 2. User types in their message 3. User clicks “Post message” button 4. Message is posted into the chat |
| Post-Conditions: | 1. A chat message is created |
| Error-Conditions: | 1. The message contains profanity or links |
| Non-Functional Requirements: | See Section 3.3 Non-Functional Requirements |

* + 1. **Instructor views class list**

|  |  |
| --- | --- |
| Use-case Name: | Instructor views class list |
| Actors: | Students, Instructor |
| Pre-Conditions: | 1. Internet connection.  2. Application services are up  3. Instructor has created chatroom |
| Flow of Control: | 1. Instructor accesses menu and selects to view the list of attendees/absentees/spectators 2. System displays appropriate list   [Instructor Unique ID]  [Course Name and Section]  [Student Name(s)] |
| Post-Conditions: | 1. A list of student’s name is shown |
| Error-Conditions: | 1. The chatroom is empty |
| Non-Functional Requirements: | See Section 3.3 Non-Functional Requirements |

* + 1. **System sends notifications**

|  |  |
| --- | --- |
| Use-case Name: | System sends notifications |
| Actors: | Users |
| Pre-Conditions: | 1. Internet connection. 2. Application services are up 3. Users are registered with the app 4. User have access to a chatroom 5. User enables notifications |
| Flow of Control: | 1. Chatroom opens up and/or student post a new comment 2. System finds all registered users with chatroom 3. System sends notification to the users |
| Post-Conditions: | 1. Notification persists until dismissed by user |
| Error-Conditions: | 1. System opens notification from a chatroom that is no longer in session |
| Non-Functional Requirements: | See Section 3.3 Non-Functional Requirements |

* + 1. **User updates account information**

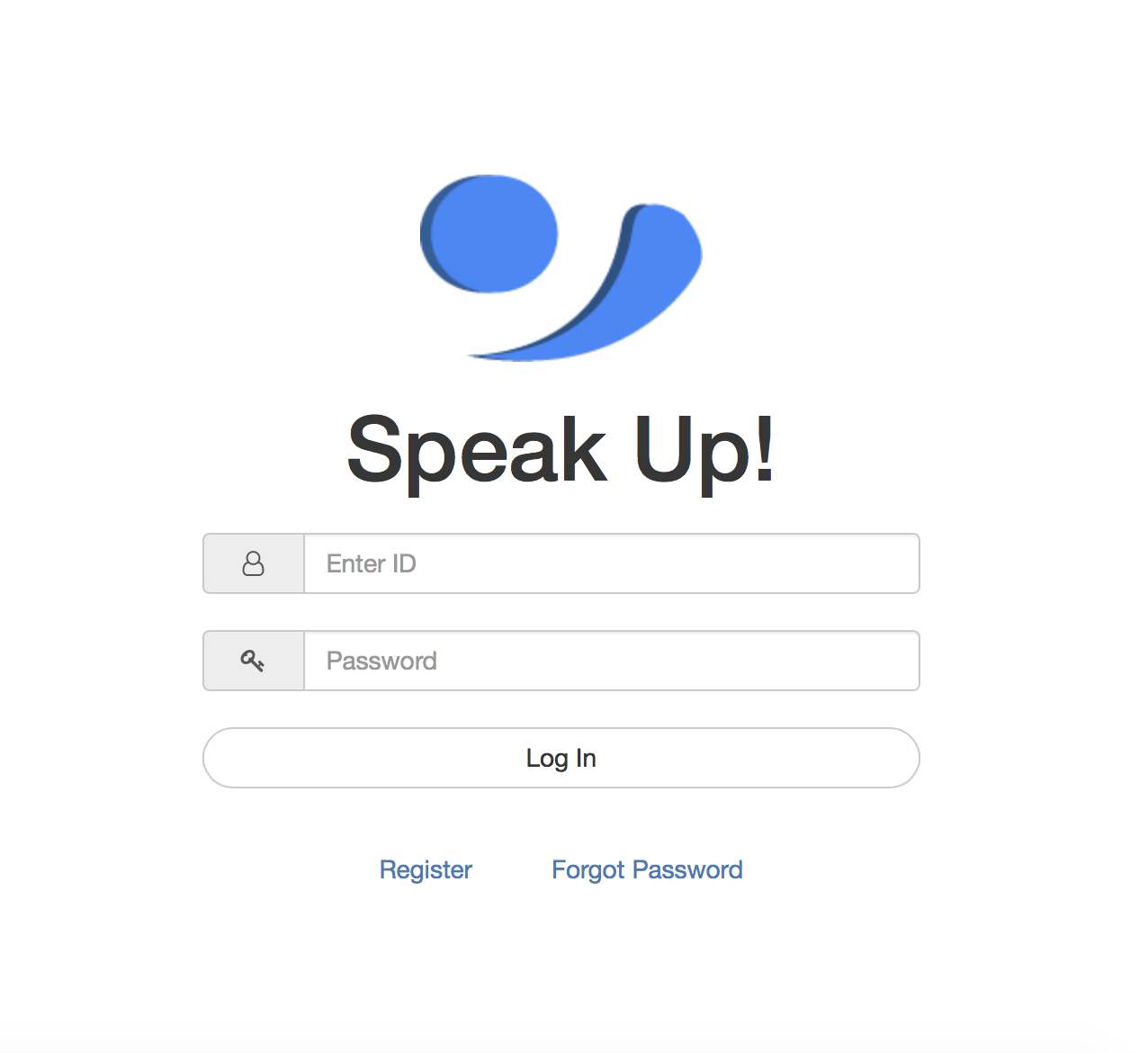
|  |  |
| --- | --- |
| Use-case Name: | User updates account information |
| Actors: | User |
| Pre-Conditions: | 1. Internet connection.  2. Application services are up  3. User is registered with the app and enrolled in courses |
| Flow of Control: | 1. User accesses account management tools 2. User updates account information 3. System updates database   [Unique User ID]  [User Information] |
| Post-Conditions: | 1. The database is updated with the new user information |
| Error-Conditions: | 1. System failure  2. User does not provide valid information |
| Non-Functional Requirements: | See Section 3.3 Non-Functional Requirements |

**3.6 Mockup of GUI:**

These are the basic mockups of the user interface and layout. The positions and designs are subject to change as the project progresses.

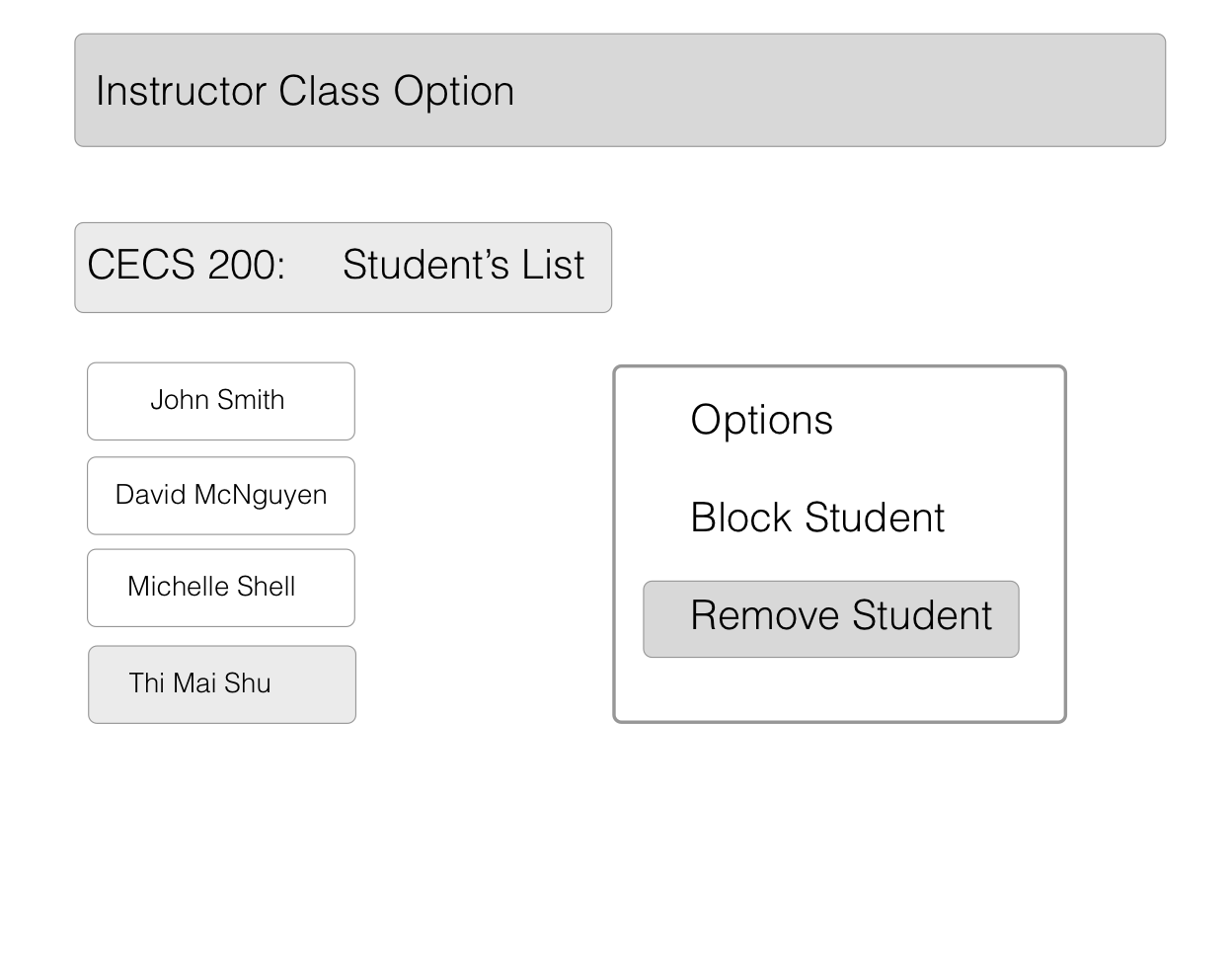
**Login Screen**

Enter user ID and password then press login to enter.

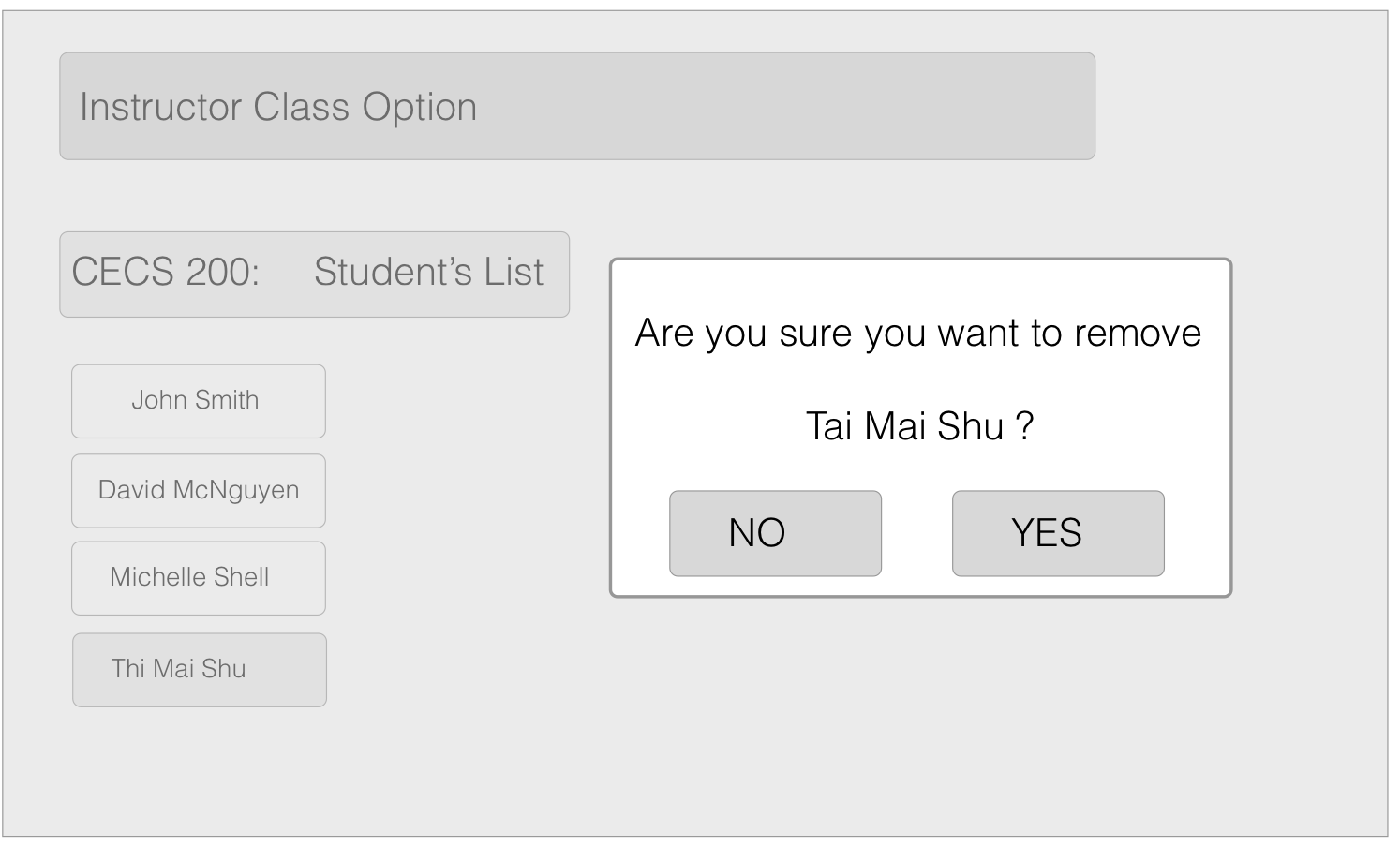


I**nstructor removes a student**

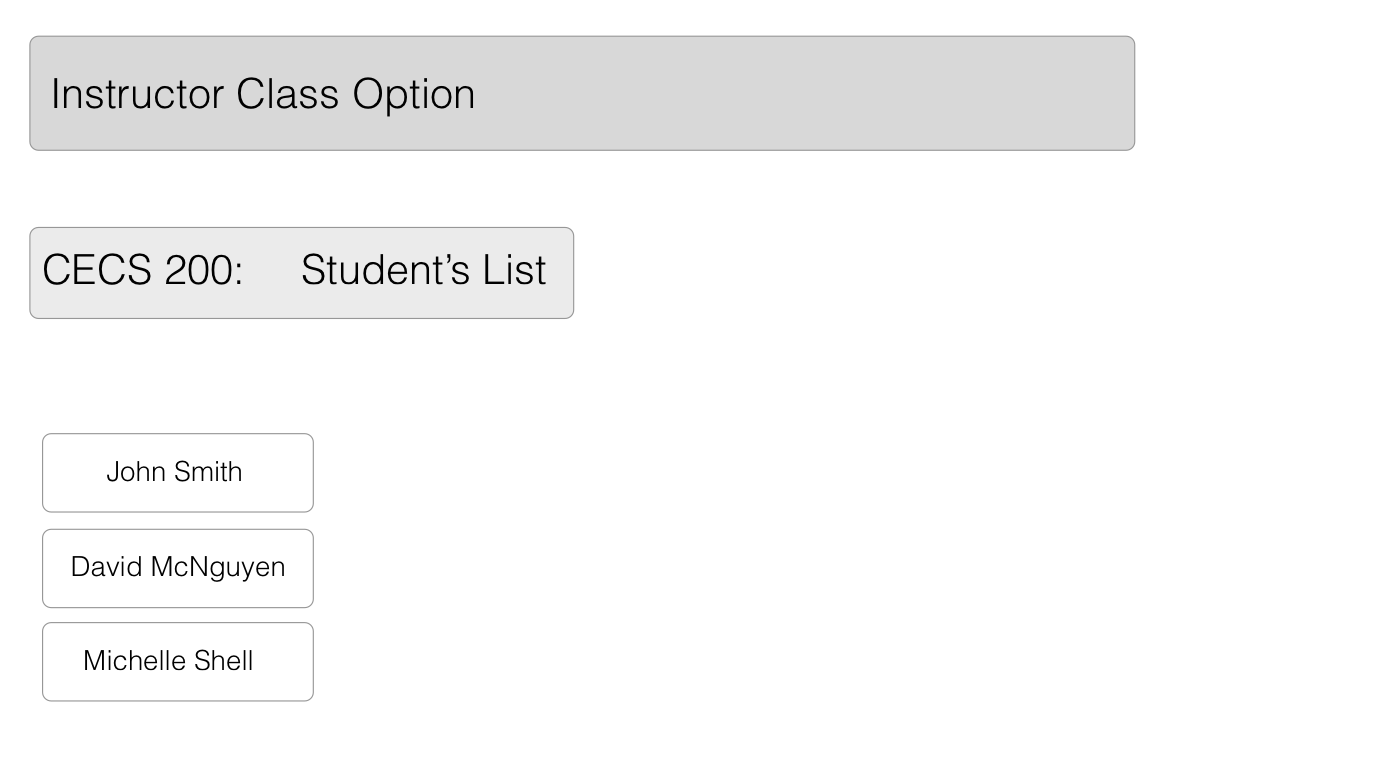
The instructor goes into the class they want to remove a student from. Then the class’s student list would populate and they can click on the student and a popup menu would pop up.



A pop up confirmation will ask if the teacher wants to remove the student.

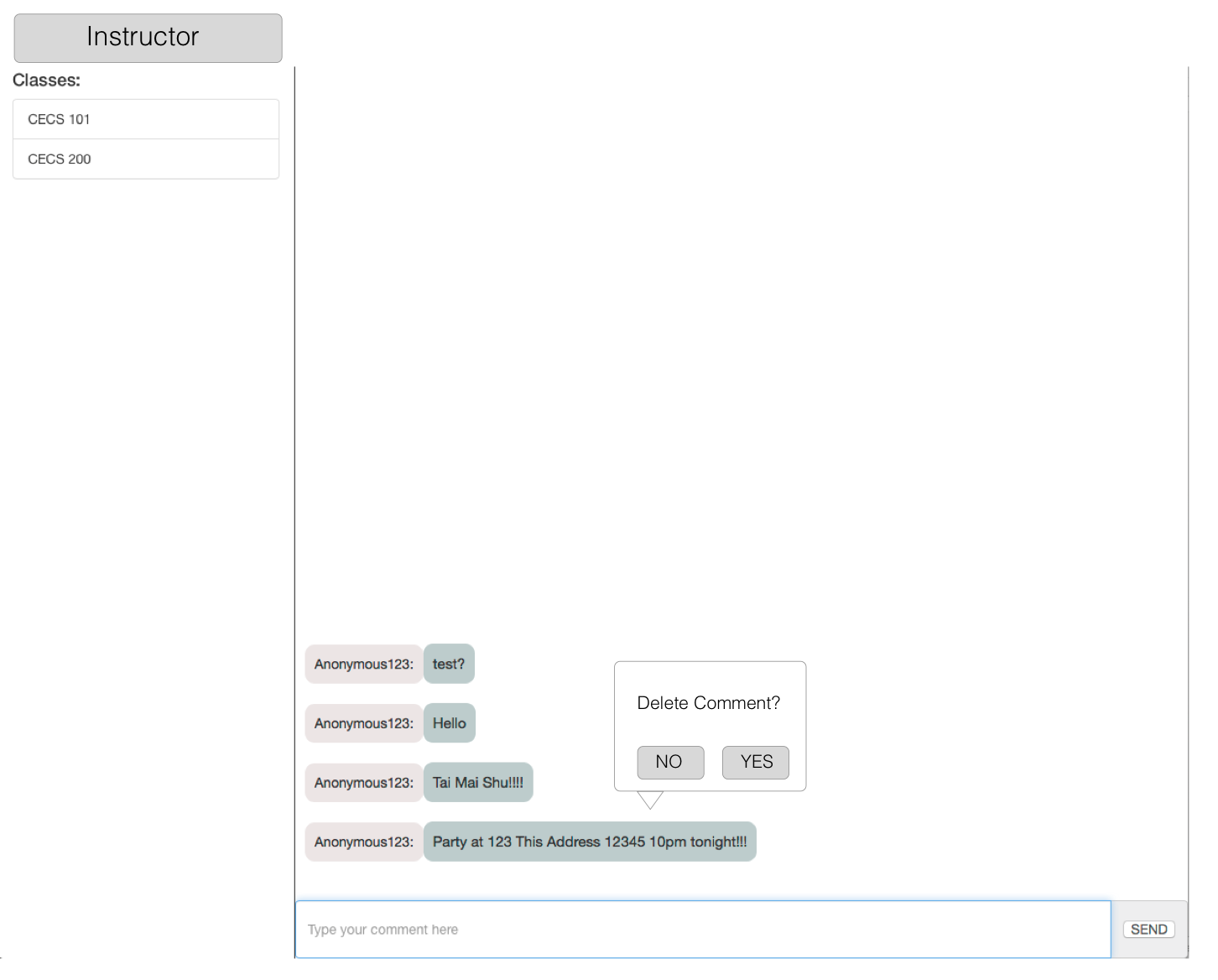


Pop up disappears and student is removed.

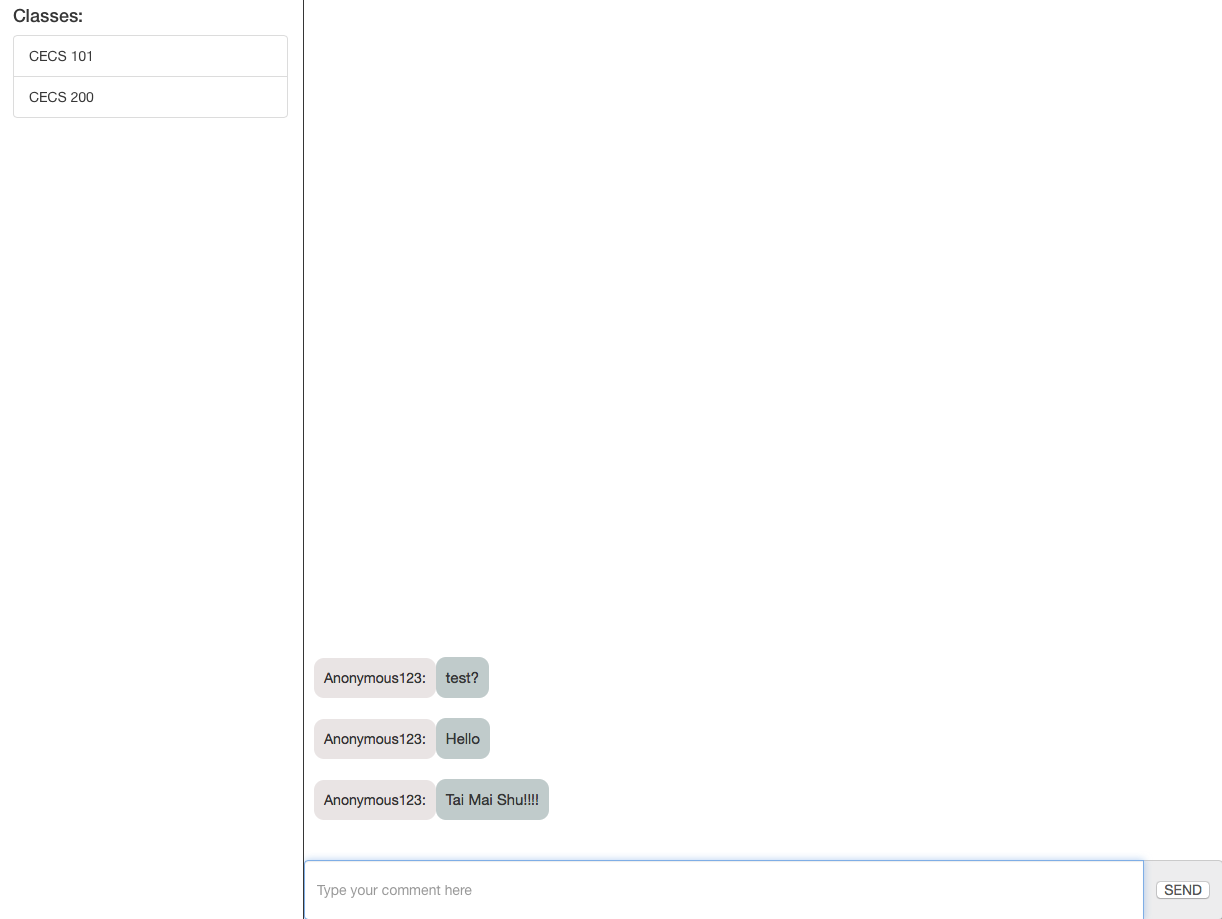


**Instructor deletes a post**

Instructor clicks on a comment and a small pop up window then instructor choose option to delete comment

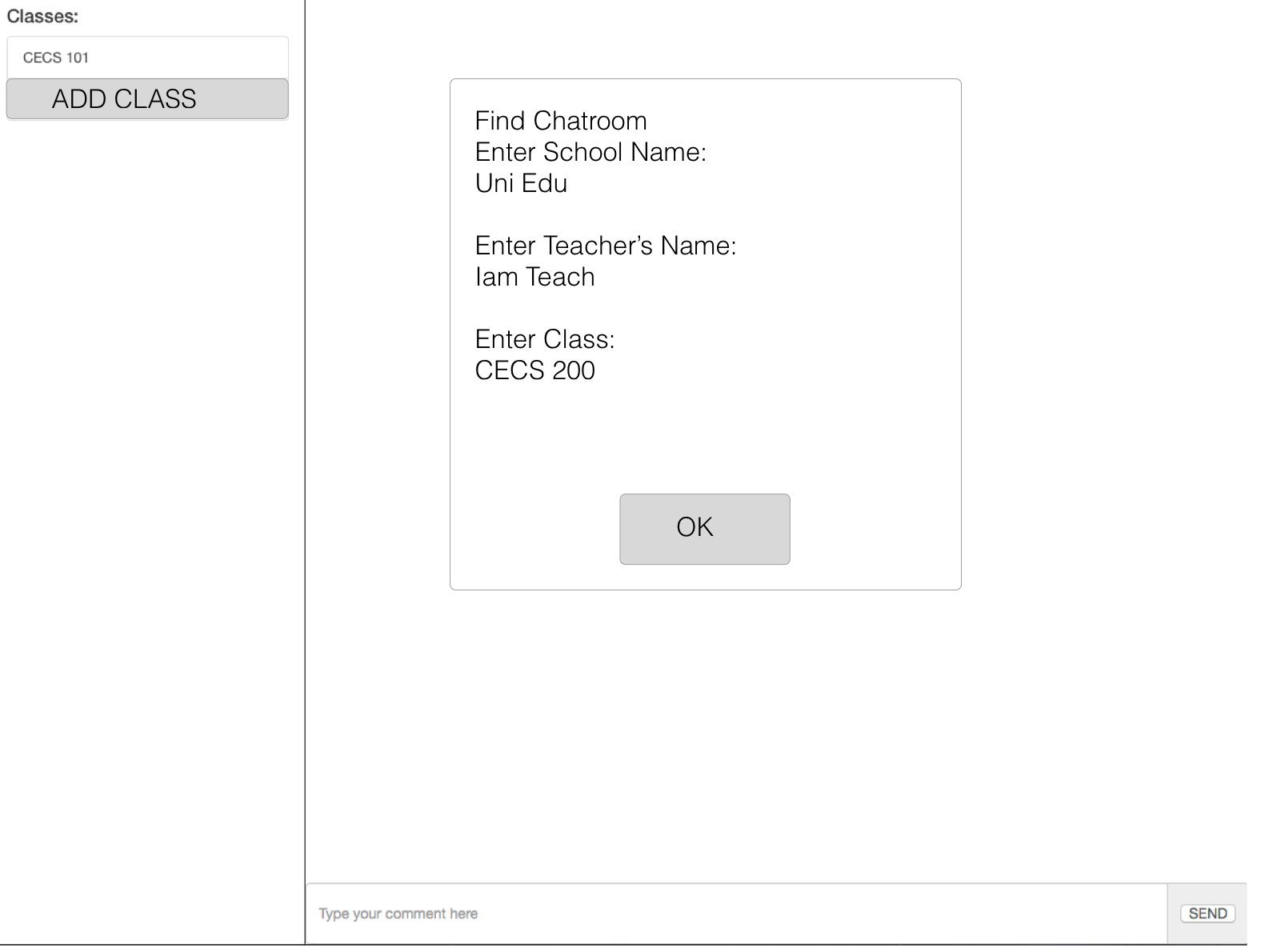


The comment is removed from the chatroom

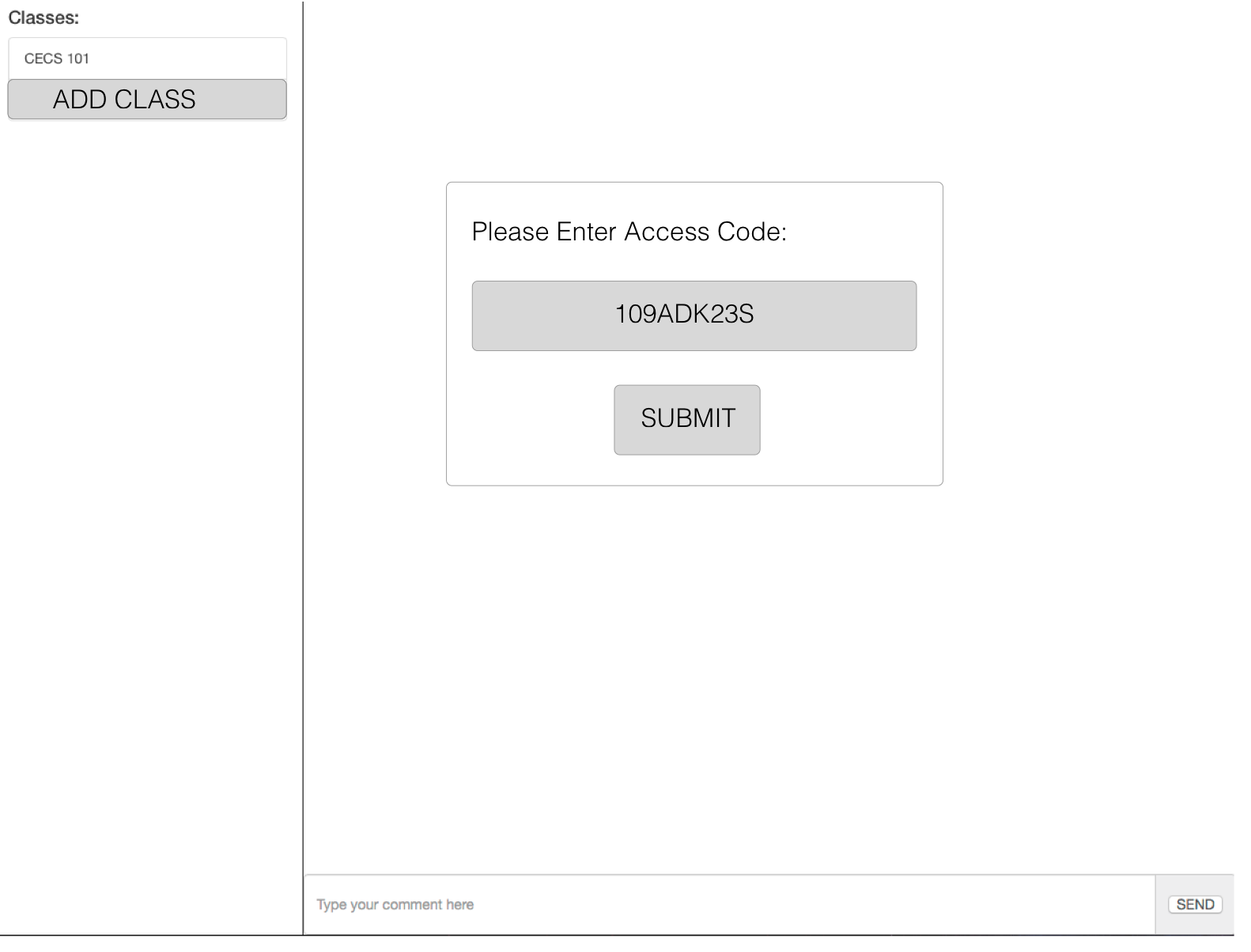


**Student enters a chatroom**

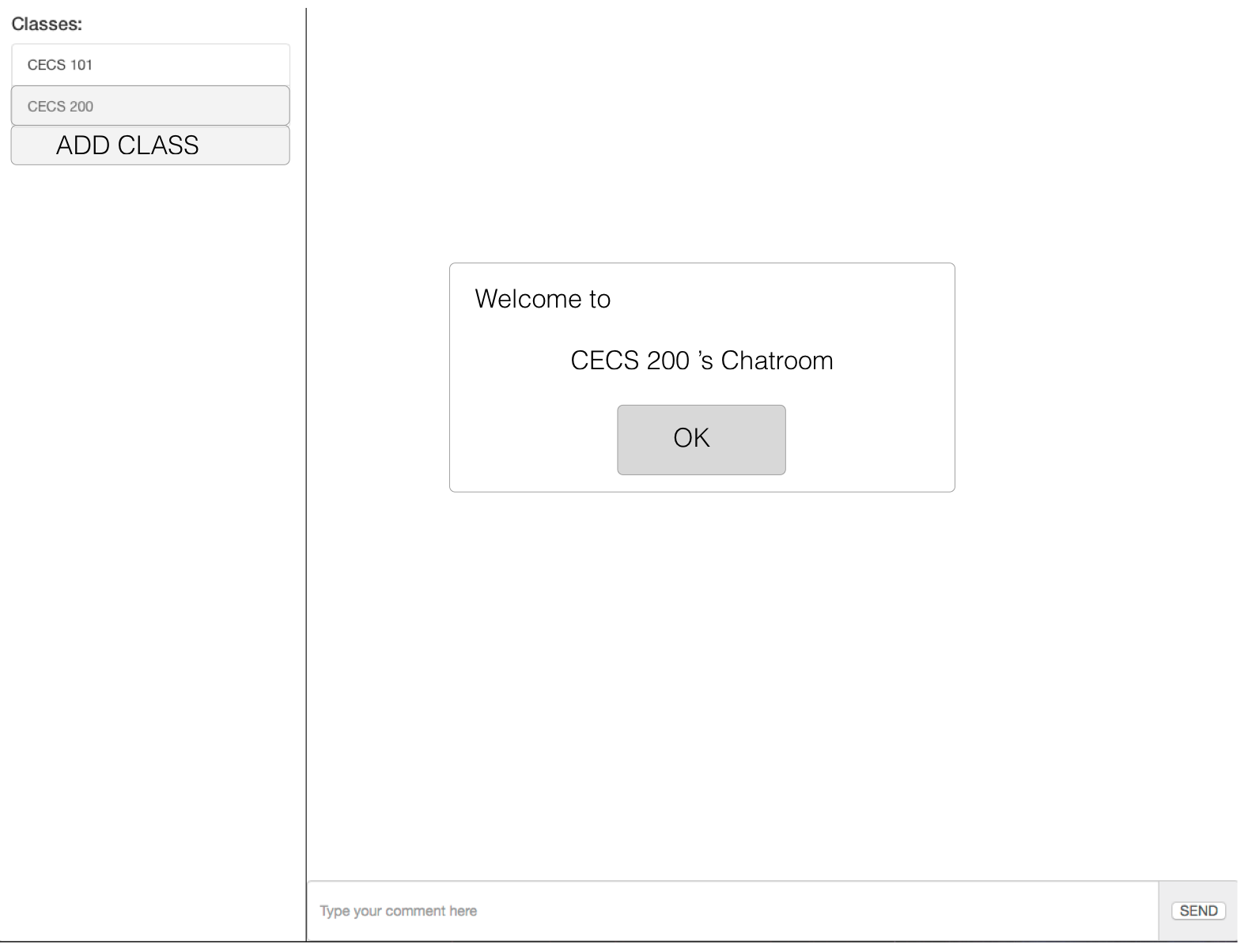
Student finds the chatroom the professor created.



Student selects a class and enters the access code the first time.

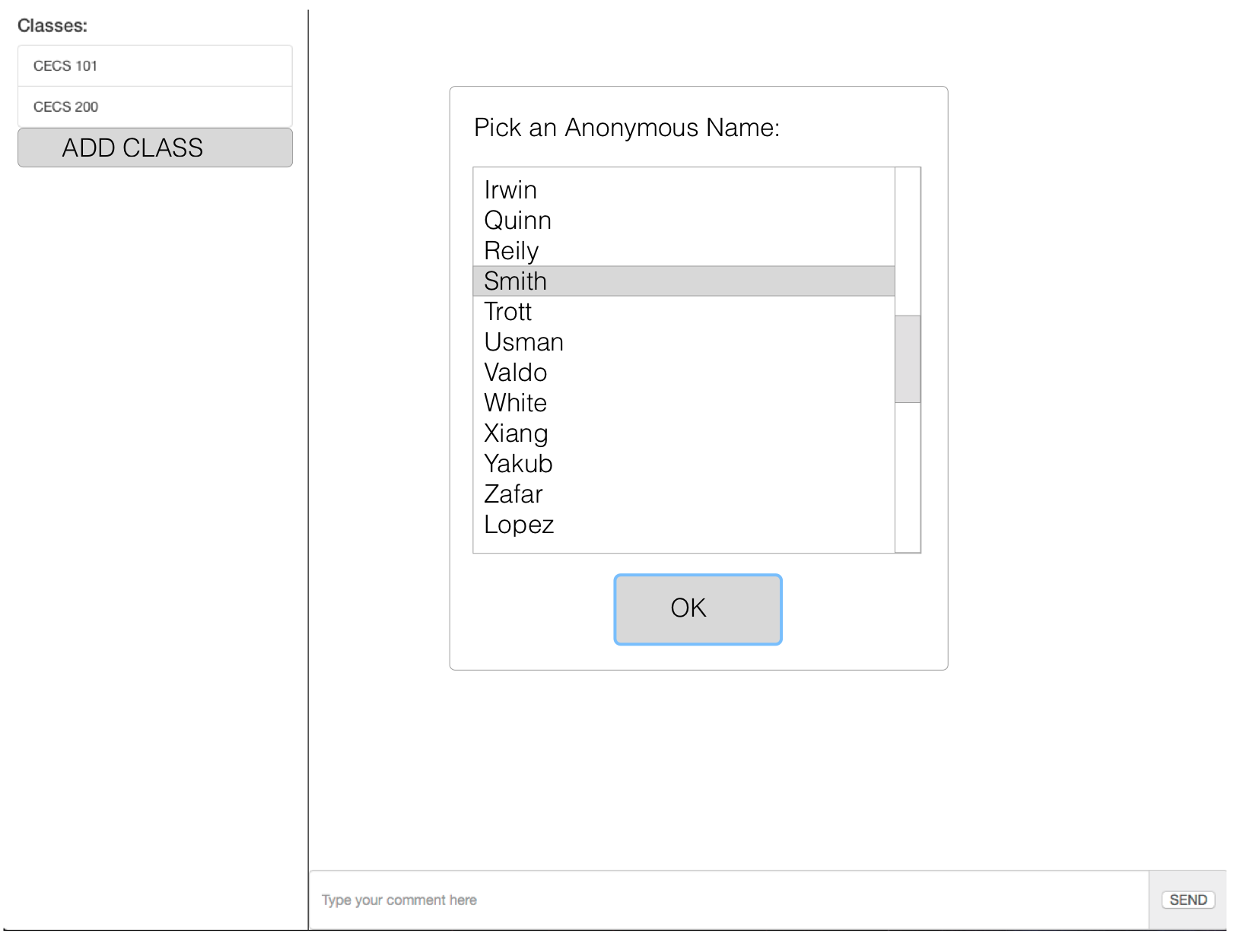


Once the access code is entered the student has access to the chatroom.



**Student picks a name**

Student is given a list of premade names that they can pick from.



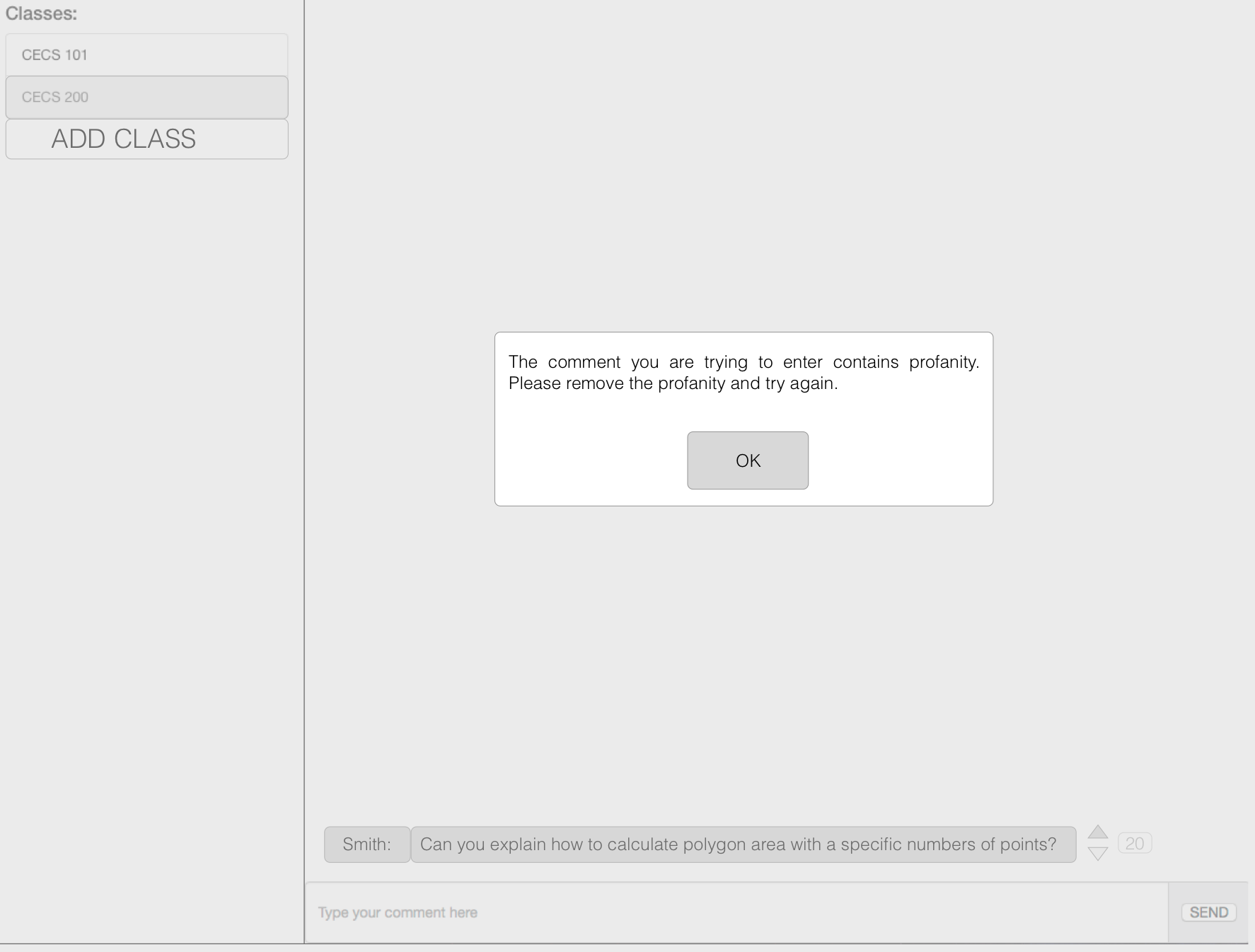
**Student votes on comment and receive points**

Students are able to receive points on the comments. Other students are able to vote on the comments by clicking the up arrow or the down arrow.



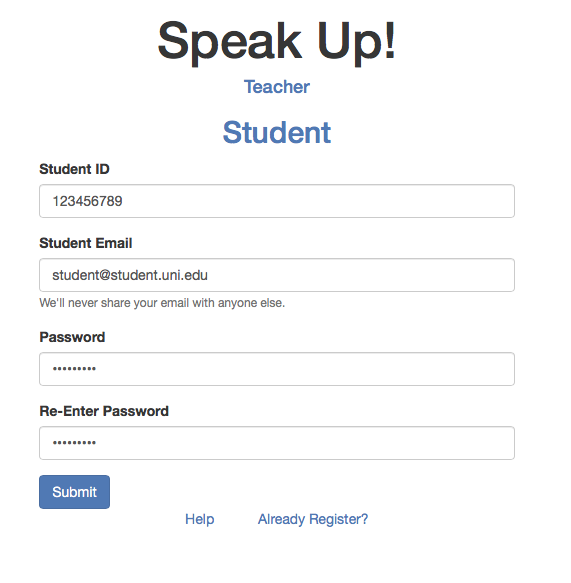
**System blocks comments that contains profanity**

The system detects the comment that contains profanity and then blocks it prompting the user to retry.



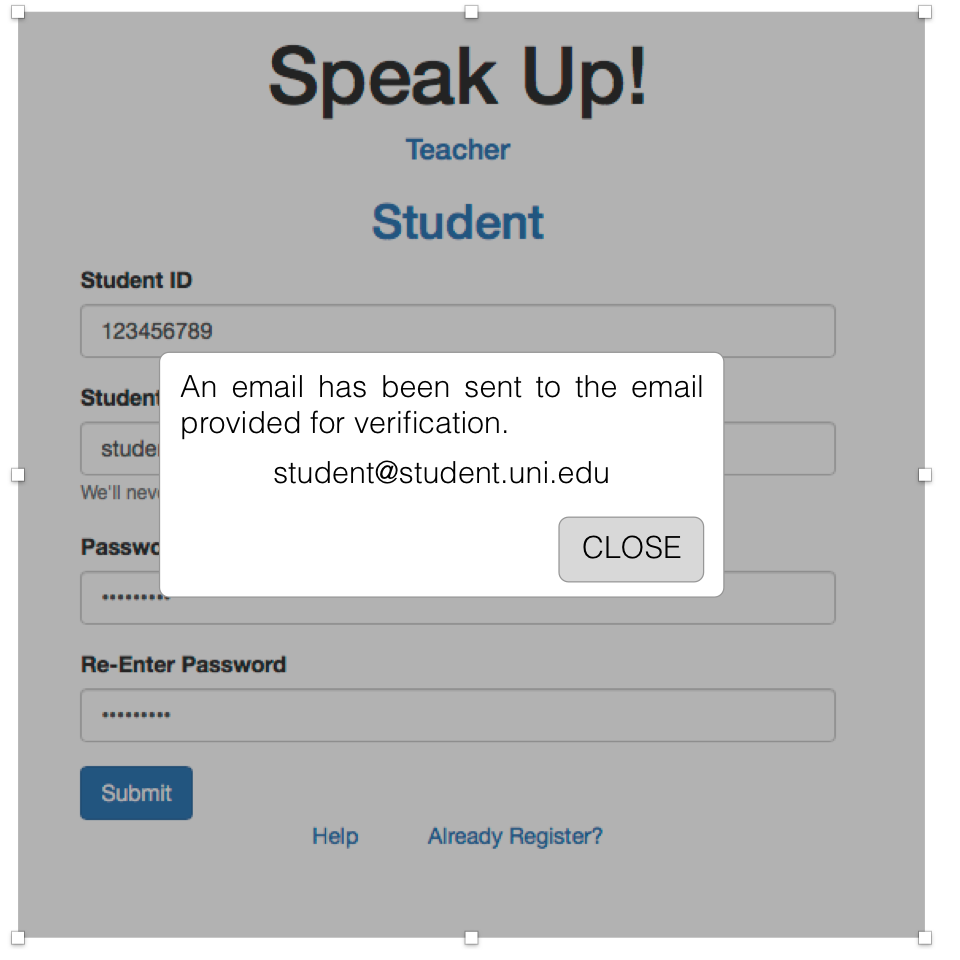
**User creates an account**

User selects register and enter their information and press submit if they are a teacher, they would register as a teacher instead.



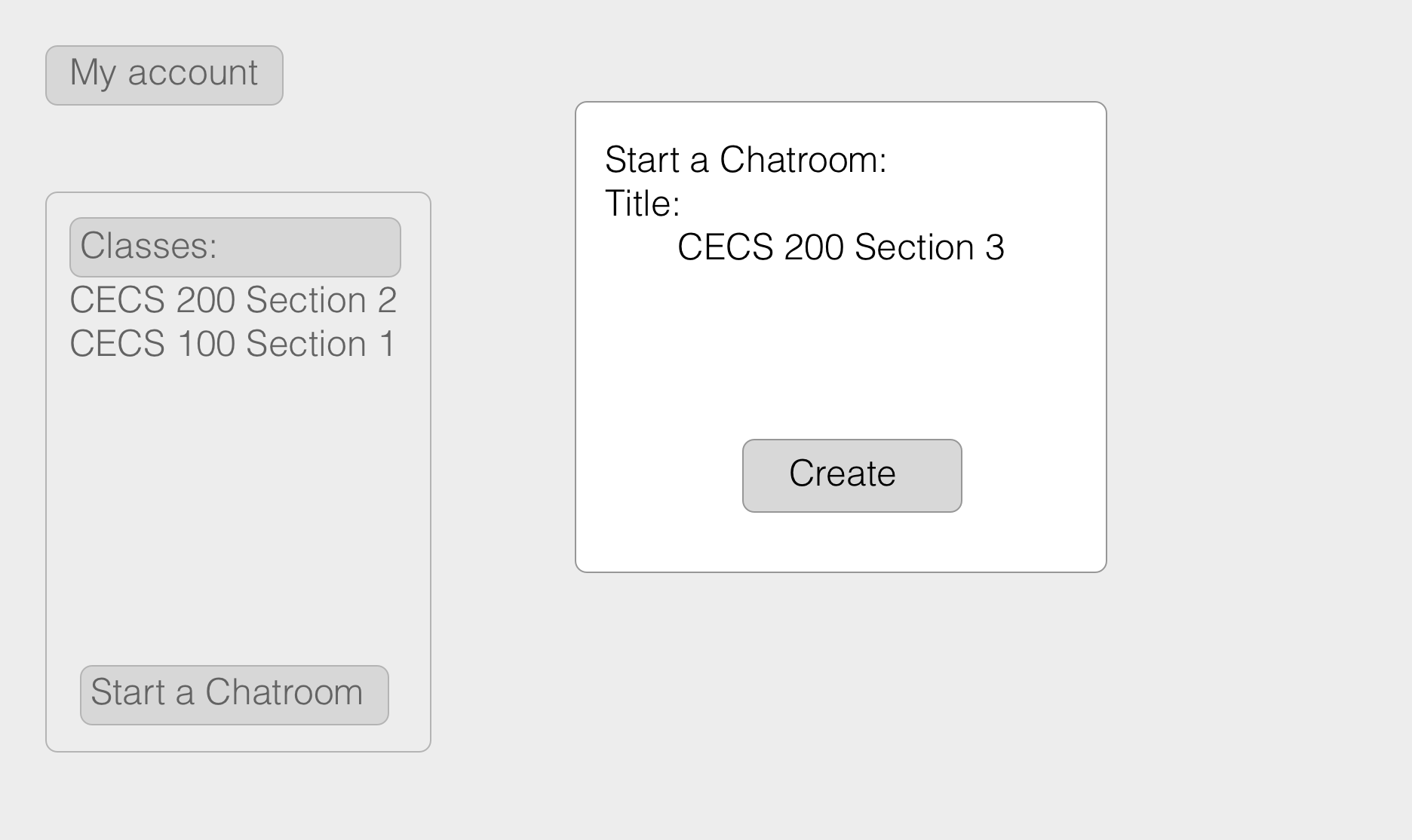


A pop up will show that the System has sends a verification e-mail to user after they have submitted the information.

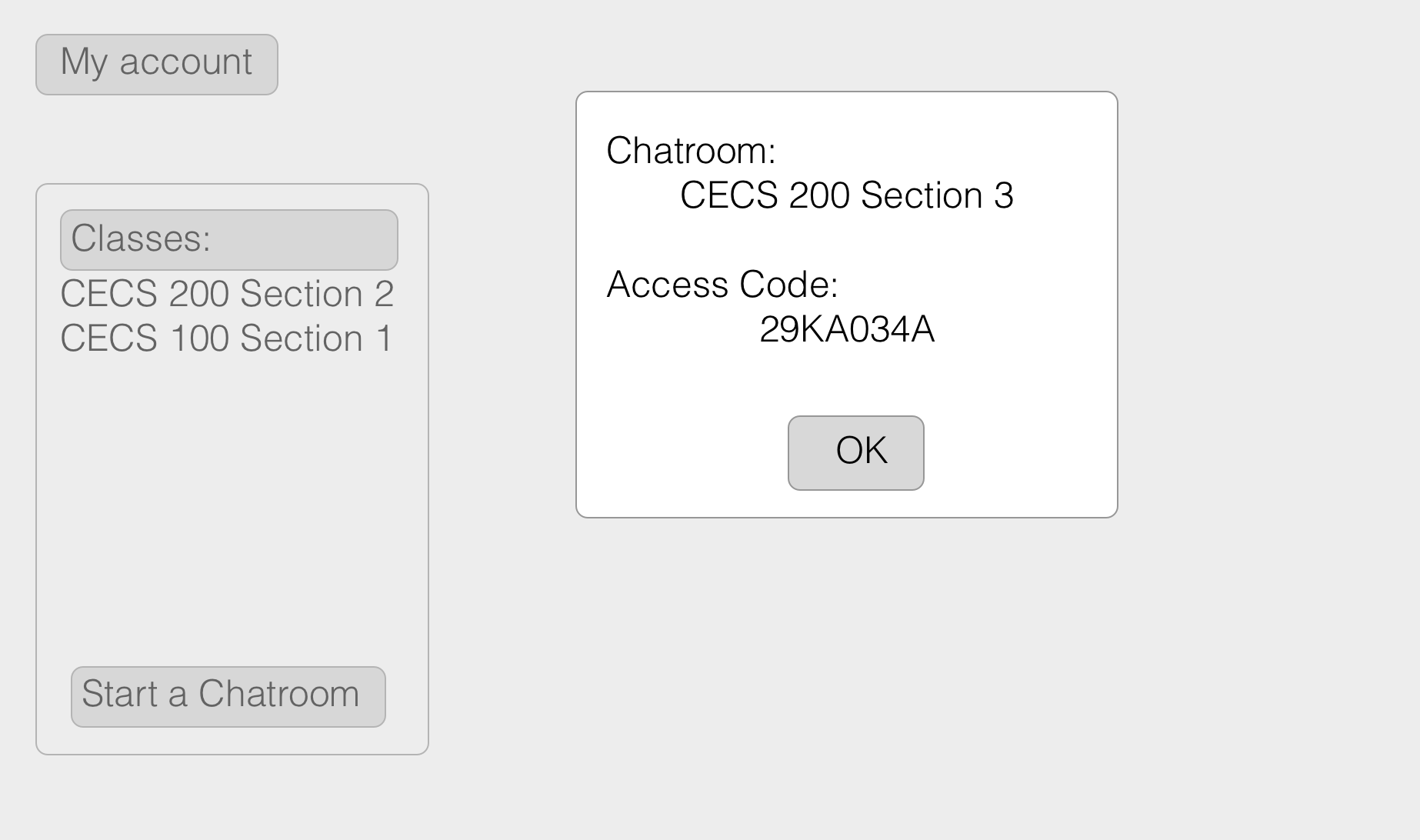


**User Creates a chatroom**

If the user is the instructor they are able to create a chatroom for their student. They would go into their account settings and click “Start a Chatroom” and a popup window shows up asking for title.

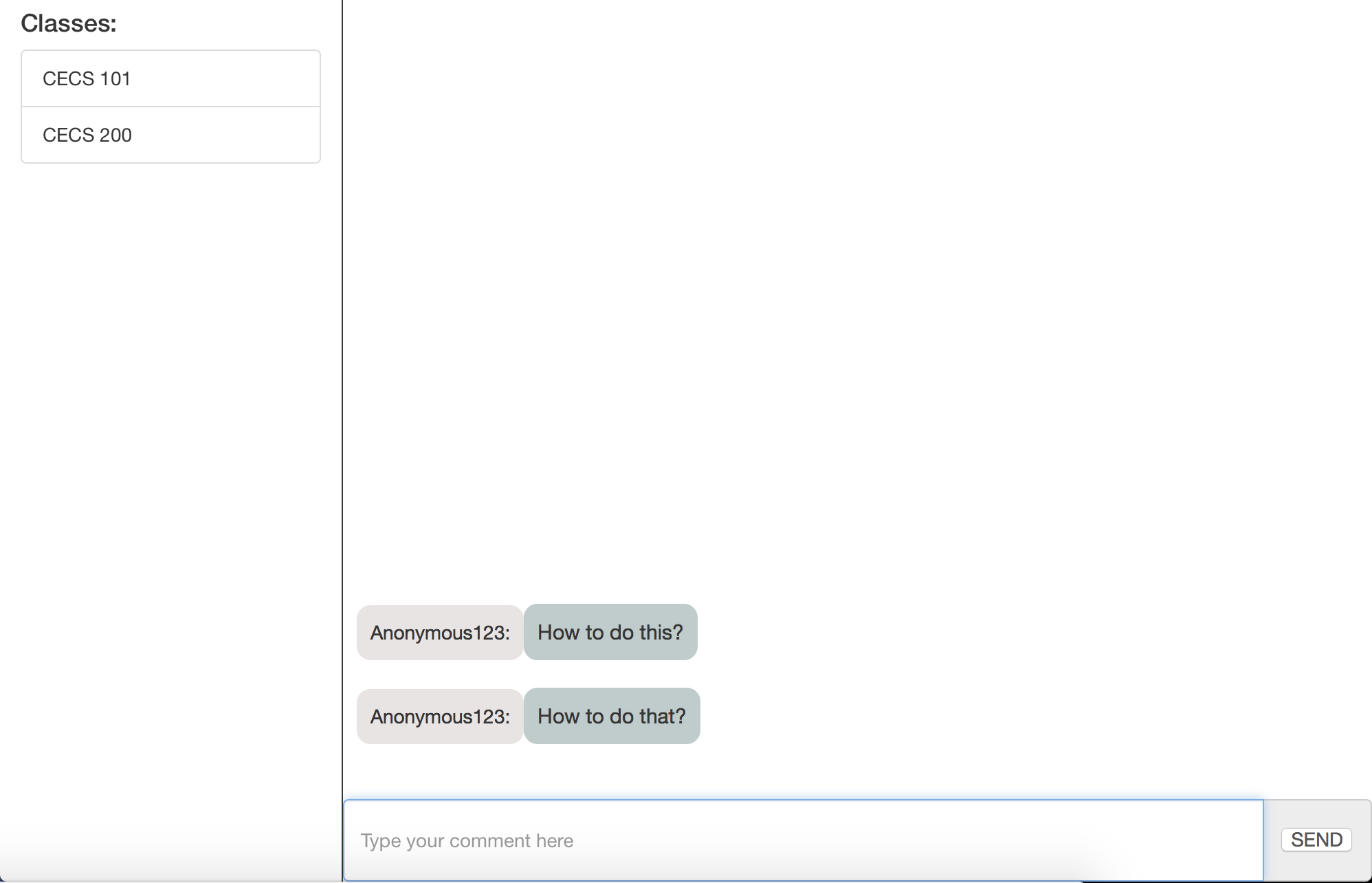


Once the instructor clicks “Create” a new pop up window will display the access code for students to enter.



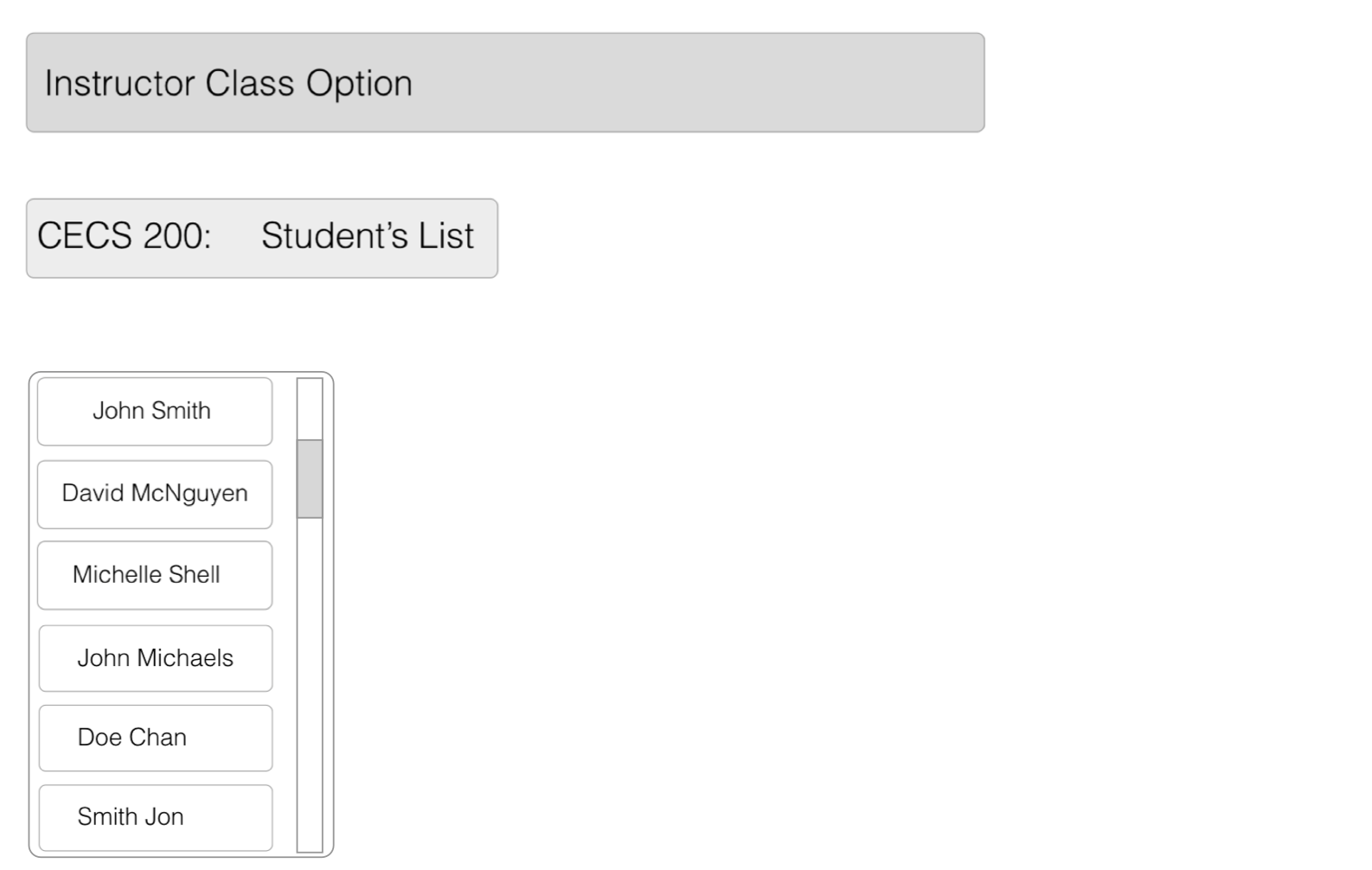
**Users posts a message**

Student can type their comment in the text area and press enter or send. Clicking on a different class will show them a different chatroom with the same GUI.



**Instructor views class list**

The instructor is able to go to their account and click on a classroom to view the students have have register into the classroom



**User updates account information**

The user is able to update their account information by going into their account settings.

