**Team Name:**

The Coding Company

**Team Members:**

Francisco Fierro

Kyle Pamintuan

Matthew Le

Daniel Martinez

**Team Leader:**

Matthew Le

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: | 1. Option menu should pop up next to student name 2. Message should pop up to confirm if instructor wants to kick student |

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: | 1. Option menu should pop up next to comment 2. Message window should pop up to confirm instructor wants to delete a comment |

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 enters access code from instructor 3. 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: | 1. Message should pop up if student try to enter chatroom without access code. 2. Message should pop up if student enter wrong access code. 3. Message should be similar to “Please enter current access code”. |

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: | 1. A large list of names should appear after student have access to chatroom 2. Student should be able to select name and display on username input |

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: | 1. Arrow icon pointing up 2. Arrow icon pointing down 3. Each comment should be separated from each other 4. Each comment should have a set of their own up and down arrows |

1. System blocks messages that contains profanity

|  |  |
| --- | --- |
| Use-case Name: | System blocks messages 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: | 1. System message to student should be similar to “Please enter a comment without use of profanity” 2. Message should be a modal |

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: | 1. Verification e-mail should be sent to user within 15 minutes of sign-up 2. User email and password info should be secure and encrypted |

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: | 1. Access code must be unique to each active chat room 2. Chat room must be able to hold up to 400 users 3. System must be able to support up to 200 chat rooms at once |

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: | 1. A student should not be able to post more than 15 messages within a one-minute span 2. A message should appear if student are spamming messages |

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: | 1. The list should have different columns for attendees/absentees/spectators 2. List should display up to 300 names |

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: | 1. There should be a switch button or option for user to enable/disable notifications in their account settings. 2. Notification message should be similar to “There is a new comment in [Class Name]” 3. Or “[Class Name]” is now open.” |

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: | 1. Database should have the appropriate user account information. 2. Each input should have the corresponding database data. 3. There should be a save button that will update the database with the new input information. |