Second Iteration

Team WCJ

Christopher Coyne ctc2141

Phu D Pham pdp2121

William Chiu wnc2105

Julie Song js5360

PART 1:

User Stories:

1. FRESHMEN: As a freshman, I want to view the Columbia campus and spatially understand its layout so that I will be less disoriented when I eventually come to campus for the first time.  
     
   My conditions of satisfaction are:

Common cases:

* 1. The chat rooms must be linked to different locations on Columbia campus, and link to one another in a way consistent with campus’s layout.

One special case is:

1. If the user tries to access a location, or a chatroom associated with a location which we do not support, we will reject any such request.
2. INCOMING STUDENT: As an incoming student (freshmen/transfer student), I want to be able to communicate with other new students so that I can build an auxiliary social network before physically meeting them.

My conditions of satisfaction are:

Common cases:

1. There must be a communication functionality in the form of chat that allows me to send messages to other people on the platform
2. CLUB MEMBERS: As a member of an in person club on campus, I want a virtual meeting place where I can meet up with other members so that we can continue to get to know each other and have a fun way to socialize, even though we are not meeting on campus.

My conditions of satisfaction are:

Common cases:

1. Users should be able to move around the room so that people can know who is talking.

Special cases:

1. If a user tries to move to an invalid spot in a chat room, via an API request, their move will be invalidated.

Difference: Due to time constraints, we were not able to integrate the friend’s functionality discussed in the first iteration report “If a user tries to private message another user which they are not friends with, their message will result in an error, and will be rejected (the message won’t be sent).”

PART 2:

**function passport\_callback (accessToken, refreshToken, profile, done)**

Purpose:

Middleware function to check login credentials and utilize Google’s API.

Equivalence classes:

Invalid Equivalence classes:

* Profile’s email is undefined.

Test case: “check invalid email credentials (empty)”

* Profile’s email does not end with “columbia.edu” or “banard.edu”.

Test case: “check invalid email credentials (not columbia)”

* User has already logged in.

Test case: "check already logged in"

Valid Equivalence classes:

* Profile’s has not logged in, has a defined email address ending in “columbia.edu” or “banard.edu”.

Test case: “check valid email credentials”

**function loggedIn(email)**

Purpose: check valid socket connection.

Valid Equivalence Classes:

* Socket is connected for the client with the associated email.

Test case: “check valid socket connection”

Invalid Equivalence Classes:

* Socket is not connected for the client with the associated email.

Test case: “check invalid socket connection”

**function moveCheck(email, location, socket)**

Valid Equivalence Classes:

X and Y coordinates of location is between 0 and 100, The Lions object has the email parameter as a property, and this email property has a non null room as a property.- testcase movecheck branch 9

This represents if a user with an existing email and room moves his lion avatar around the screen normally.

Invalid Equivalence classes:

Location is not a valid object - testcase: movecheck branch 1

Location has more than 2 coordinates - testcase: movecheck branch 2

X coordinate of location < 0 - testcase: movecheck branch 3

X coordinate of location > 100 - testcase: movecheck branch 4

Lions object does not contain the email parameter as a property -testcase: movecheck branch 7

There is no room property of the email attribute of Lions- movecheck branch 8

Y coordinate of location < 0 - testcase: movecheck branch 5

Y coordinate of location > 100 - testcase: movecheck branch 6

**Boundary Analysis:** These equivalence classes all generate error messages if the character tries to move outside the coordinate plane, or if they do not have a valid room associated with their avatar (so the backend doesn't know what room to move them around in), or if they don't have a valid email property, which contains their room. Since the coordinate plane is from [0,0] to [100, 100], an x or y value that is below 0 or above 100 would place the user's character outside of the screen, which would render them invisible and provide for a confusing user experience. In our testing, we made sure to account for invalid coordinate pairings that are below 0 or above 100.

**function checkDisconnect(socket, email)**

Valid Equivalence classes:

The Email parameter is a property of the Lions object - testcase: Check checkdisconnect branch1

If a player disconnects, and their email exists in our object of lion avatars, the function will remove their email from the associated room and from the Lions object.

Invalid Equivalence classes:

The Email parameter is not a property of the Lions object. - testcase: Check checkdisconnect branch2

This means that the player disconnected, but in an error, their email was not part of the Lions object. The function will manually loop through each room and delete the user's email.

**function fetchEmailLocation(room)**

Purpose: Update the room for all clients after an action of another client.

Valid Equivalence Classes:

* The requested room is not empty.

Test case: “check handleNewSocket”

Invalid Equivalence Classes:

* The requested room is empty.

**function changeRoomCallback(Rooms,room,email, socket)**

Purpose: Move a client from a room to another room, and update the rooms information for people in the old and new rooms.

Valid Equivalence Classes:

* Requested room is in the room database.

Test case: "check if a move from room to room happens"

Invalid Equivalence Classes:

* Requested room is not in the room database.

Test case: “check if a move from room to room happens”

**function onConnection(socket, debug)**

Test case: Test io init

Valid equivalence classes:

* Handle connection to room endpoint.

Test case: “check handle room”

* Handle connection to move endpoint.

Test case: “check handle move”

* Handle connection to chat endpoint.

Test case: “check handle chat”

**function initRooms()**

Valid equivalence classes:

* MongoDB is connected and rooms datatable can be fetched.

Test case: "Test initialization of database"

Invalid equivalence classes:

* MongoDB is not connected or rooms datatable cannot be fetched.

**function handleNewSocket(socket)**

Test case: handlenewsocket

Valid Equivalence classes:

* User is not logged in
  + The database will search for the user's email and add them to a room.

Invalid equivalence classes:

* User is already logged in
  + The socket the user is connected to will be disconnected

PART 3: Link to test report:

<https://github.com/WilliamChiu/ASE-Team-Project/tree/master/coverage/lcov-report>

File name: app.js.html

Branch coverage: 94.87%

PART 4: Link to Travis configuration:

<https://github.com/WilliamChiu/ASE-Team-Project/blob/master/.travis.yml>

Link to Travis Report:

<https://travis-ci.com/github/WilliamChiu/ASE-Team-Project>