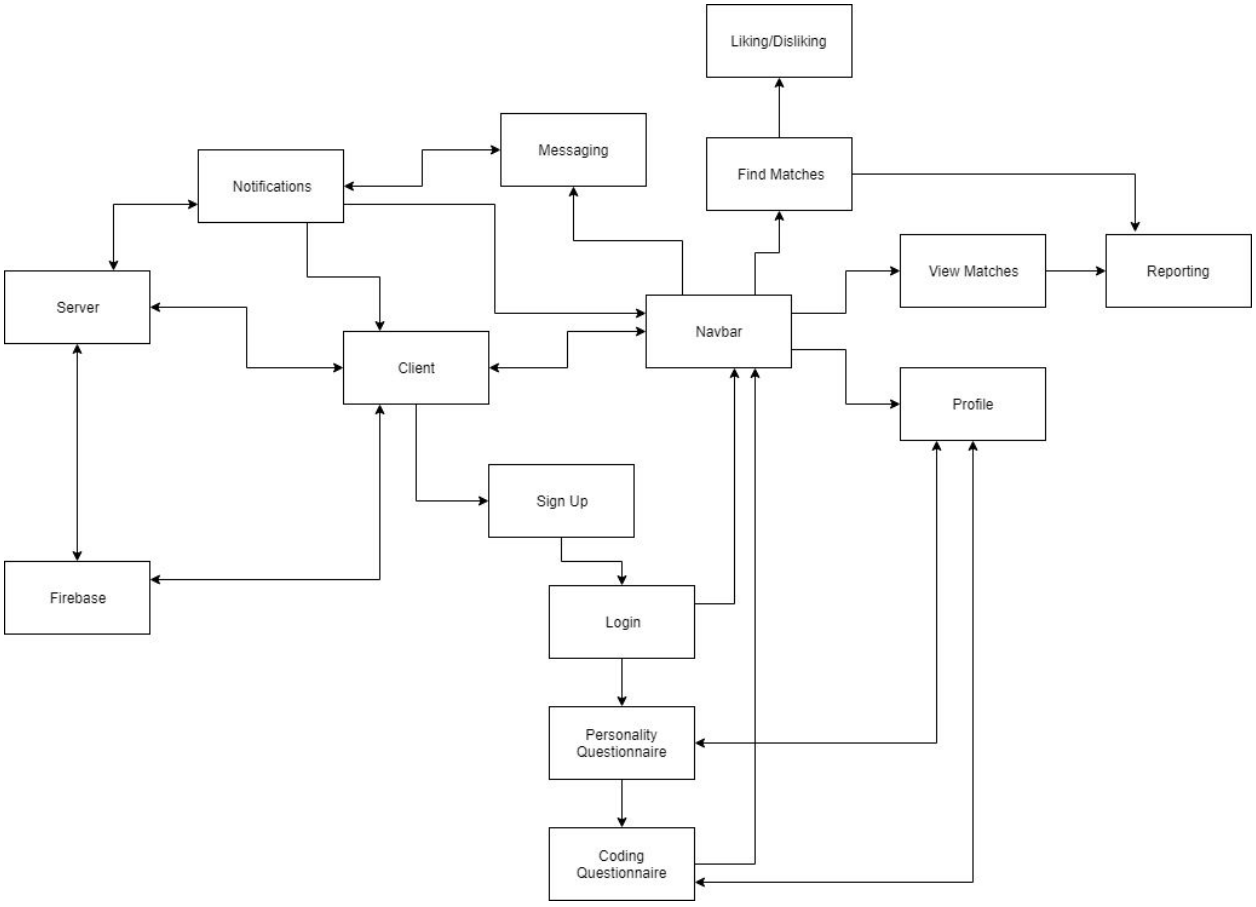
Incremental and Regression Testing

Team 8

Team Members: David Carr, Shivan Desai, Reid Kippenbrock, Laura Kristie, Paul Krivacka

1.Classification of Components



Firebase:

Function: Stores and retrieves the data for our website.

Interaction(s):

Server: The server requests data from and stores data in firebase.

Client App: Uses firebase's authentication to track whether or not the user is logged in, and their account if they are.

Navbar: It lets users log in or sign up, both of which send and get data from firebase.

Personality Questionnaire: Receives the user's answers for the personality questionnaire and stores them

Coding Questionnaire: Receives the user's answers for the coding questionnaire and stores them

Matching Page: Returns possible matches data for the user to the matching page.

Messaging: Stores, loads, and adds to messages between users

Liking/Disliking: Adds and removes people from the like list of users

View Matches: Returns the accounts a user has matched with

Input: Incoming request with JSON-formatted information in the body

Output: A "success" or the requested information

Dependent Components: Server, Sign Up, Login, Navbar, Personality Questionnaire, Coding Questionnaire, Matching Page, Profile, Messaging, Liking/Disliking, Matches

Dependent On: None

Server:

Function: Host the website, store data in the database, retrieve data from the database, and the matching algorithm.

Interaction(s):

Firebase: The server requests data from and stores data in firebase.

Personality Questionnaire: Sorts through a user's personality questionnaire answers to determine who they will match to.

Coding Questionnaire: Sorts through a user's coding questionnaire answers to determine who they will match to.

Client API: Handles requests from the client to get lists for potential matches and matched users. Also handles liking/disliking/unliking and reporting users

Notifications: Sends and receives socket.io notifications to the correct users

Input: A user client connecting with the server

Output: The information the client requested/a response back

Dependent Components: Client as a whole, including Sign Up, Login, Navbar, Personality Questionnaire, Coding Questionnaire, Matching Page, Notifications, Messaging

Dependent On: Firebase

Nav Bar:

Function: Acts as a navigator for the site. A link to most of the main pages is in the nav bar. Users can use the nav bar to create or log in to their accounts.

Interaction(s):

Firebase: Requires the user authentication from firebase to see whether or not a user is logged in. It also sends the login and sign up requests to firebase and receives a response.

Matching Page: Appears on the matching page.

Personality Questionnaire: Links to the personality questionnaire

Messaging: Links to the messaging page

Notifications: Listens for new message notifications and displays a count of unread message notifications
Profile: Links to the profile page
Find Matches: Links to the 'Find Matches' page
View Matches: Links to the 'View Matches' page

Input: User Click/Typing
Output: Redirecting the user to a new page or sending something to firebase
Dependent Components: Matching Page, Profile, Personality Questionnaire, Coding Questionnaire, Notifications, Find Matches, View Matches
Dependent On: Client, Server, Firebase

Personality Questionnaire:

Function: The function of the personality questionnaire is to get information about the user and their preferences so it can be used in matching them to other people.
Interaction(s):

Firebase: Stores the answers to its questions in firebase.
Matching Page/all user interaction pages: (Indirectly) Provides the data with which the matching page and algorithm depend on, necessary for all user interactions
Coding Questionnaire: Once the personality questionnaire is completed, it redirects to the coding questionnaire.

Input: User Click/Typing
Output: Redirecting the user to the coding questionnaire and storing their choices in the database
Dependent Components: Matching Page
Dependent On: Client, Server, Firebase

Coding Questionnaire:

Function: The function of the coding questionnaire is to get information about the user and their coding preferences so it can be used in matching them to other people.
Interaction(s):

Firebase: Stores the answers to its questions in firebase.
Matching Page: (Indirectly) Provides the data with which the matching page and algorithm depend on.

Input: User Click/Typing
Output: Redirecting the user to the coding questionnaire and storing their choices in the database
Dependent Components: Matching Page
Dependent On: Client, Server, Firebase, Personality Questionnaire

Profile Page:

Function: The function of the profile page is to display the profile information to the user, and allow them to edit information or change their profile pictures or personal pictures.
Interaction(s):

Firebase: Fetches the required user information from firebase, and updates firebase database according to edited information.

Questionnaires: Allows users to edit their answers on the questionnaires.

Navbar: Links to the user's profile page

Input: User Click/Typing

Output: Redirecting the user to a new page or sending something to firebase

Dependent Components: Coding Questionnaire, Personality Questionnaire

Dependent On: Client, Server, Firebase, Navbar

Find Matches:

Function: The function of matching is to pair together users based on the answers to their personality and coding questionnaires and their preferences.

Interaction(s):

Firebase: Fetches information needed from Firebase and stores the current matches and previously matched users to Firebase.

Questionnaires: (Indirectly) The questionnaires provide the data needed to match users together.

Liking/Disliking: The Find Matches page allows the user to like/dislike other accounts

Reporting: You can report other users on the Find Matches page

Input: User Click/Typing

Output: Redirecting the user to the potential matches page along with the list of matched users.

Dependent Components: Potential Matches Page, Liking/Disliking, Reporting

Dependent On: Client, Server, Firebase, Navbar, Personality Questionnaire

Liking/Disliking:

Function: The function of liking/disliking is to allow users to choose who they like and would be interested in matching with and messaging.

Interaction(s):

Find Matches: (Indirectly) You can only like/dislike people from the 'Find Matches' page

Server: The request to like/dislike someone is sent to the server

Input: User Click

Output: A request to the server to add someone to the Liked or Viewed (but not liked) list in the database

Dependent Components: None directly

Dependent On: Client, Server, Firebase, Navbar, Find Matches

Messaging:

Function: Allows users who have liked each other to communicate.

Interaction(s):

Firebase: The messages users send to each other are stored in and retrieved from firebase

Navbar: Links to the messaging page

Notifications: Messages the user receives show up in notifications if they are not currently on the messages page

Server: Messages are sent to the server to be stored and loaded

Input: User Click/Typing
Output: A request to the server to add a message to the conversation history between two users
Dependent Components: Notifications
Dependent On: Client, Server, Firebase, Navbar

Notifications:

Function: To show when a user is sent a message.
Interaction(s):
Navbar: Displays the notification count
Messages: Creates the message that is then sent as a notification to the recipient's open tabs/windows
Server: Sent the notification to the recipient's tabs/windows
Input: A socket.io socket.emit() function
Output: An increased unread message count on the navbar
Dependent Components: Navbar
Dependent On: Client, Server, Firebase, Navbar, Messages

View Matches:

Function: View the users you have matched with (when you both like each other).
Interaction(s):
Navbar: Displays the view matches page
Server: Gets a request to get the matches and returns a user's matches
Firebase: Retrieves a user's matches data
Input: A click on the Navbar/page refresh
Output: The user accounts a user has matched with
Dependent Components: Reporting, Liking / Disliking
Dependent On: Client, Server, Firebase, Navbar

Reporting:

Function: To report a user when they break the rules or act inappropriately.
Interaction(s):
Find and View Matches: The pages where users can report other users
Server: Report request sent to the server
Input: Clicking/Typing
Output: An email to the account that was reported
Dependent Components: N/A
Dependent On: Client, Server, Firebase, Navbar, Find Matches, View Matches

Form of Incremental Testing

We used bottom-up testing to test our project. Each module was first tested individually with various inputs and test cases. For example, input boxes would be checked locally on the client

during development, and we would use drivers to test the algorithms. After the individual module testing was done, we tested the system as a whole.

2. Incremental and Regression Testing

Module	**Component A: Liking/Disliking/Reporting Users**

Incremental Testing

Defect #	**Description**	**Severity**	**How to Correct**
1	There should not be an error in disliking a user when you are not on the other user's liked list.	2	Add a check that the user is either on the list of likedUsers or potentialMatches of the disliked user.
2	Storing potentialMatches, likedUsers, and matchedUsers should not create any circular references in the database.	1	Change storage to store references to the users instead of a copy of their information
3	Page crashes when an empty list of users is returned (Matchies & matching)	1	Added a check for an empty list and renders a separate page empty
4	Users with empty data are being sent as matches, crashing the page	1	Added a check for the returned list, to ensure that all users have valid information
5	Whenever a user is like/dislike/unlike/reported nothing happens on the page until the database request is fulfilled	3	Added a spinning loading icon to all buttons to let user know that something is happening

Regression Testing

Defect #	**Description**	**Severity**	**How to Correct**
1	Fixing the circular reference caused an issue with calculating the highest scores in potentialMatches	2	Calculate the scores earlier before storing them.

2	Fixing the circular reference caused an issue with previous matches not being properly checked for.	2	Iterate through users previous matches when generating new potential matches.
3	The loading icon is not being removed when a user cancels out of the reporting menu.	3	Added a function to remove the loading indicator when a user cancels out of the reporting menu.
4	The Returned list from reporting a user throws a promise error.	2	Check if the promise returns an error, if it does, try to fetch matches normally

Module	**Component B: Profile Picture/ Picture Upload**

Incremental Testing

Defect #	**Description**	**Severity**	**How to Correct**
1	The header shows the incorrect navbar as if the user isn't logged in.	2	Make sure the authorized user is passed to the header component correctly.
2	The default picture doesn't appear when a new user creates an account.	3	Changed the picture name for the initial call to the database where picture url is saved.
3	User's uploaded pictures don't fit within the matches/ matching pages panels.	3	Added additional css to images within the miniprofile.

Regression Testing

Defect #	**Description**	**Severity**	**How to Correct**
1	Picture Upload doesn't check if upload file is a picture.	2	Add regex expression to check if the file has a .jpg, .png, .bmp or .gif.

Module	**Component C: Editing Questionnaires**

Incremental Testing

Defect #	Description	Severity	How to Correct
1	The update personality questionnaire button is not visible.	1	Make sure the class name is capitalized for edit questionnaire component.

Regression Testing

Defect #	Description	Severity	How to Correct
1	Ensure data is correctly updated in database.	1	Adjusted variable name for doc id for user to store in the database.

Module	Component D: Notifications

Incremental Testing

Defect #	Description	Severity	How to Correct
1	When multiple tabs are open, the same notification will happen on each tab and increment the number	3	Make the counter local for each tab, so when it goes to each one it increments to the same number everywhere.
2	Window crashes if the count element does not exist upon an update	1	Run the notification code upon page load.

Regression Testing

Defect #	Description	Severity	How to Correct
1	Ensure that different components' socket areas do not overlap and display things on the client twice	3	Do not allow the socket io events to modify anything outside of their component

Module	**Component F: Messaging**

Incremental Testing

Defect #	**Description**	**Severity**	**How to Correct**
1	Socket.io creates a socket for every component that uses it instead of referencing the same socket throughout	3	Send the socket data back for each component, and classify the component using the socket on the server
2	Dynamically created Nametags using JavaScript cannot be clicked/interacted with	2	Create different react components to load the various parts of the message page
3	When a User presses Enter to send a message, their message does not show up in the chat window and they are bombarded with alerts (Enter is registered multiple times)	2	Instead of using pure javascript event listeners, used react onkeypress event for the messagetextarea
4	If a user gets a message from another user while a chat window is open, it'll add it to the chat list even if its from a different person	2	Added a check for incoming messages and only add them to the list if the sender is in the chatroom

Regression Testing

Defect #	**Description**	**Severity**	**How to Correct**
1	Ensure that one user did not get another's socket event	1	Add a backend check to ensure that the socket is the user's and not someone else's

Module	**Component E: Account Creation**

Incremental Testing

Defect #	**Description**	**Severity**	**How to Correct**

1	When a login Fails the user is not informed what the error is	3	Used firebase's error message in the error message for logins
2	When registering fails, the user is not informed what the error is	3	Used firebase's error message in the error message for registering

3. Update Product Backlog

Functional Requirements

Backlog Id	Functional Requirements	Hours	Status
1	As a user I would like to create a HƐ>LO WORLD account.	4 hours	<Completed in Sprint 1>
2	As a user I would like to login and logout of my HƐ>LO WORLD account.	4 hours	<Completed in Sprint 1>
3	As a user, I would like to fill out a personality questionnaire.	10 hours	<Completed in Sprint 1>
4	As a user, I would like to fill out a fun coding questionnaire.	8 hours	<Completed in Sprint 1>
5	As a user, I would like to be able to edit my questionnaire answers.	10 hours	<Completed in Sprint 2>
6	As a user, I would like to be able to upload pictures of myself and my interests to show on my profile.	8 hours	<Completed in Sprint 2>
7	As a user, I would like to be able to set/change my name, profile description, and preferences on my account.	6 hours	<Completed in Sprint 1>
8	As a user, I would like to be matched with people who have similar personalities to my own.	10 hours	<Completed in Sprint 1>
9	As a user, I would like to view my potential matches and our match compatibility score.	10 hours	<Completed in Sprint 1>
10	As a user, I would like to "like" other accounts to display my interest in getting to know them.	8 hours	<Completed in Sprint 2>

11	As a user, I would like to be able to chat with users I've been matched with.	12 hours	<Completed in Sprint 2>
12	As a user, I would like to view a list of others who have been matched with me.	8 hours	<Completed in Sprint 2>
13	As a user, I would like to receive notifications for when I receive a new chat message.	8 hours	<Completed in Sprint 2>
14	As a user, I would like to be able to "unlike" someone if they are not right for me.	6 hours	<Completed in Sprint 2>
15	As a user, I would like to be able to report other users.	4 hours	<Completed in Sprint 2>
16	As a developer, I would like to be able to verify if a user is logged in on all required pages.	8 hours	<Completed in Sprint 1>
17	As a user, I would like to be able to easily navigate between web pages.	5 hours	<Completed in Sprint 1>
18	As a user, I would like a secure login page (HTTPS).	8 hours	<Completed in Sprint 1>
19	As a user, I would like a Matching webpage where I can view potential matches.	6 hours	<Completed in Sprint 2>
20	As a user, I would like a Matches webpage where I can view people I have been matched with.	6 hours	<Completed in Sprint 2>
21	As a user, I would like to have homepage that explains what HƐ>LO WORLD is.	5 hours	<Completed in Sprint 1>
22	As a user, I would like to be able to see the online status of my matched users.	8 hours	<Completed in Sprint 2>
23	As a user, I would like to select a picture as my profile picture to display on the matching page.	6 hours	<Completed for Sprint 2>