Product Backlog

Web Chat Client with Chatbot

10th March 2017

Team 8

Atul Aneja, Jalaleldeen Aref, Tarang Khanna, Wyatt Larkey, and Joel Van Auken

Problem Statement

Today, a significant portion of communication between people take place in online chat rooms. While these instant messaging platforms perform include lots of fun, interesting features, they tend to stifle productivity and become a distraction. We plan to create a new chat client that people can use to communicate with friends and clients, but also include some productivity tools, namely a chat bot that can retrieve and display stock market information. This way users can continue to enjoy the convenience of a chat room but also have access to useful productivity tools.

Background Information

For the last 25 years or so, instant messengers and chat rooms have provided a convenient way for people to communicate over the internet. Today, millions of people communicate with instant messengers all over the world. Instant messengers have dozens of features including links to social media, sending videos and hyperlinks, as well as much more. For professionals, instant messengers have become a workplace tool, to facilitate communication between team members. But sometimes all these features can lead to workplace distractions and lost productivity.

What we want to do is provide a chatroom that provides features that promote productivity. The chatroom will have basic functionality including text messaging, but also include work tools that include a chatbot which helps the user find current stock data.

Environment

We plan to develop our browser based application using a typical IDE like Atom or Sublime along with using HTML, CSS, and javascript to create the front-end. To store data we will use an online database service called firebase. For the chatbot we will use a service called ai.api to parse user input. Our server runtime will be implemented using python.

Functional Requirements

Backlog ID	Functional Requirement	Hours	Status
1	As a user, I can create a username	15	Completed in Sprint 1
2	As a user, I can secure my username with a password	5	Completed in Sprint 1
3	As a user, I can join a room	10	Completed in Sprint 2
4	As a user, I can post text messages to a chatroom	5	Completed in Sprint 1
5	As a user I can tag a bot to send it commands	5	Completed in Sprint 1
6	As a user I can create rooms	10	In-progress: Sprint 2
7	As a user I can protect my rooms with a password	10	In-progress: Sprint 2
8	As a user, I always get meaningful responses from bots	5	Completed in Sprint 1
9	As a user I can enter a chat room with a guest account that is not persistent	5	Completed in Sprint 1
10	As a user I can reset my password with my email address	5	In-prog

			ress: Sprint 2
11	As a user, I can write and use hyperlinks in chat rooms	5	Completed in S2
12	As a user, I can receive alerts from the application from tagged messages	5	Completed in Sprint 2
13	As a user, If I leave a chatroom and come back I will still be able to see previous messages	10	Completed in Sprint 2
14	As a user, I can see who all is in a chatroom	10	Completed in Sprint 1

Non-Functional Requirements

1. **Server initializes rooms when the application is started.**
2. **Database stores user login information, emails and data from chatrooms**
3. **Bots use neural networks to gather stock data via an API**
4. **Front end is responsive, and can comfortably be used on desktops and mobile devices**

Use Cases

Backlog ID/ Name	Action	System Response
1 Create User	1. Open application in browser 3. User enters username and password or signs in as guest 4.User hits submit button	2. Server responds and shows login page 5. Server accepts data and takes user to chathub
2 Secure User	1. Press “secure handle” button 3. User types in a username and password combo 4. User hits submit button	2. Handle secure page appears 5. Page responds with success/failure message
3 Join Room	1. User selects a chat room 5. User sees chatroom and can interact with it	2. Server checks to see if chatroom has password authentication 3. If it does, server displays password dialog box 4. If not puts user into chatroom
4 Post Text	1. User types message in message box 3. User presses “send” button	2. User sees message appear in chat log
5 Bot Chat	1. User sends message that contains a flag which sends their message to a chat bot	2. Chatbot receives the user's message 3. Chatbot parses input and generates meaningful response

	4. User sees chatbots response in their current chat room. Other users can see the chat bots response.	
6 Create Room	**1. User presses “create room” button 3. User inputs room name in textbox 4. User presses submit button**	**2. Room creation page appears 5. Server displays success/failure message 6. Server displays new room after entering user into that room**
7 Password Protect Room	**1. User creates a chatroom 3. User selects the chat room options button, a dropdown appears and the user selects “make chatroom private” 5. User makes password for chatroom and selects submit 7. User receives confirmation that chat room is secure.**	**2. Server generates chatroom for user 4. Server sends user a dialog box to create password for the chat room 6. Server sets chatroom so that all users will have to enter the password to get in the chatroom**
8 Meaningful Bot	**1. User tags chatbot in message**	**2. Chat bot responds with appropriate response**
9 Guest Access	**1. Open application in browser 3. User enters username and password or signs in as guest 4.User hits submit button**	**2. Server responds and shows login page 5. Server accepts data and takes user to chathub**
10 Reset Password	**1. User forgets password 2. User presses “forgot password” button 4. User inputs email address 6. User follows email instructions**	**3. System displays password recovery page 5. System sends user an email with password reset instructions**
11 Chat Links	**1. User copies and pastes a link into a chatroom 3. User can click the link and the browser will open a new tab and take them there.**	**2. Server determines that the text is a link and provides link functionality to the text**
12 Chat Tags	**1. Another user tags user in message**	**2. System sends notification to user**

13 Chat History	**1. User is in roomA 2. User enters roomB 3. User enters roomA again**	**4. System displays all messages from roomA including those written before the user entered roomB**
14 Chat User Status	**1. When a user is in a chatroom, there is a window on the side that shows all the users currently in the chatroom.**	**2. The server keeps track of all users in the chatroom and displays them for all users to see.**