Visit at **https://hkchatroom.herokuapp.com/** and sign in with **test01** (username) & **123456** (password).

**Please allow extra time for first click while awaiting start-up of heroku free server.

HKChatroom

Online Chatroom Web Application with MERN Framework

V1.1 | Feb 25 2018
By Jeffrey Chung

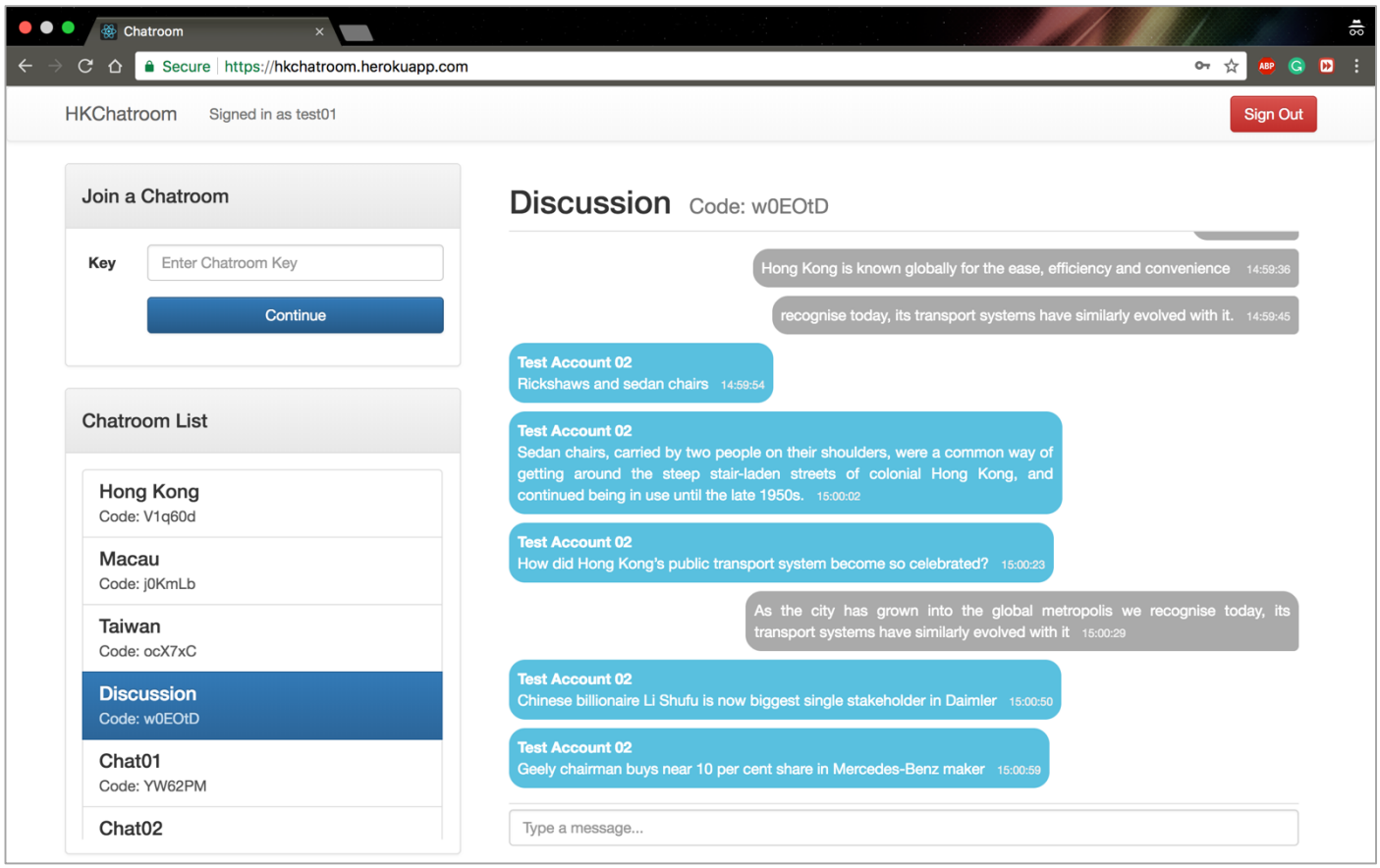
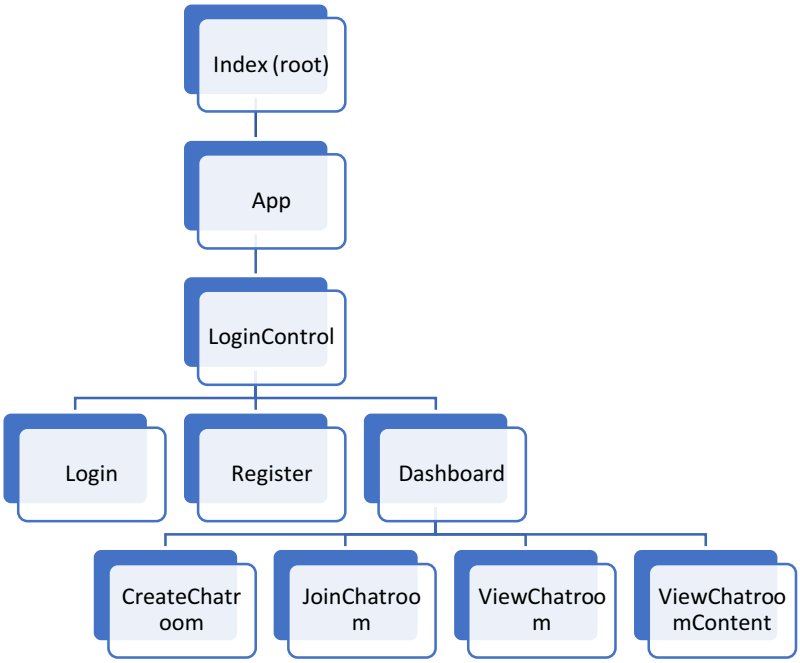


Table of Contents

Front-end: ReactJS

- Hosted on: https://hkchatroom.herokuapp.com/
- The source code follows a hierarchical order:



- Source Code File Arrangement

Name	Location
index.js	/chat-client/src/index.js
App.js	/chat-client/src/App.js
LoginControl.js	/chat-client/src/Authentication/LoginControl.js
Login.js	/chat-client/src/Authentication/Login.js
Register.js	/chat-client/src/Authentication/Register.js
Dashboard.js	/chat-client/src/Authentication/Dashboard.js
CreateChatroom.js	/chat-client/src/Authentication/Chatroom/CreateChatroom.js
JoinChatroom.js	/chat-client/src/Authentication/Chatroom/JoinChatroom.js
ViewChatroom.js	/chat-client/src/Authentication/Chatroom/ViewChatroom.js
ViewChatroomContent.js	/chat-client/src/Authentication/Chatroom/ViewCharroomContent.js

- Source Code File Functionalities

Name	Function
index.js	• Set page title • Map **App.js** to root of webpage
App.js	• Include Bootstrap 3.3.7 online source code reference for CSS • Include **LoginControl.js** for this class

LoginControl.js	• Manage authentication status of end user and fetch respective pages with the following **states** and **methods:**
	  ○ States
	  ▪ username
	  ▪ displayName
	  ▪ chatroomList
	  ▪ isLoggedIn
	  ▪ isToRegister
	  ○ Methods

Name	Function
	▪ setLogin(isLoggedIn, username, displayName, chatroomList) ▪ getUsername() ▪ getDisplayName() ▪ getChatroomList() ▪ appendChatroomList(chatroomName, chatroomKey) ▪ login() ▪ logout() ▪ register() • Include **Login.js, Register.js & Dashboard.js** for this class • Display the top navigation bar
Login.js	• Implements login interface with **states**, **methods** and **inherited methods**: ○ States ▪ usernameReg ▪ displayNameReg ▪ passwordReg ▪ confirmPasswordReg ▪ error ○ Methods ▪ clearForm() ○ Inherited Methods ▪ setLogin()
Register.js	• Implements Registration interface with **states** and **inherited methods**: ○ States ▪ username ▪ password ▪ error ○ Inherited Methods ▪ login() ▪ setLogin()
Dashboard.js	• Implements Dashboard interface (i.e. page after signed in) with **states**, **methods** and **inherited methods**: ○ States ▪ isDashboard ▪ activeChatroomID ▪ activeChatroomName ▪ activeStringKey ▪ activeLastMsgId ▪ activeChatroomMsgPack ○ Methods ▪ setViews(isDashboard) ▪ setActiveChatroom(chatroomID, chatroomName, stringKey, lastMsgId) ▪ getActiveChatroomID() ▪ getActiveChatroomName() ▪ getActiveStringKey() ▪ setActiveLastMsgId(lastMsgId)

Name	Function
	▪ getActiveLastMsgId() ▪ setActiveChatroomMsgPack(msgPack) ▪ getActiveChatroomMsgPack() ○ Inherited Methods ▪ getUsername() ▪ getDisplayName() ▪ getChatroomList() ▪ appendChatroomList() • Include **CreateChatroom.js, JoinChatroom.js, ViewChatroom.js & ViewChatroomContent.js** for this class • Manage application views of end user and fetch respective chatroom/dashboard pages
CreateChatroom.js	• Display on **Right Div** of **Dashboard Page** • Implements Chatroom Creation interface (i.e. input form) with **states** and **inherited methods**: ○ States ▪ chatroomName ▪ error ○ Inherited Methods ▪ setViews () ▪ setActiveChatroom () ▪ appendChatroomList ()
JoinChatroom.js	• Display on **Left Div** of **Dashboard & Chatroom Page** • Implements Joining Chatroom by Code and Accessing Chatroom List with **states**, **methods** and **inherited methods**: ○ States ▪ chatroomKeyerror ○ Methods ▪ setViewChatroom(key, name) ○ Inherited Methods ▪ setViews () ▪ setActiveChatroom () ▪ getActiveChatroomID() ▪ getActiveChatroomName() ▪ getActiveStringKey ()
ViewChatroom.js	• Display on **Upper Right Div** of **Chatroom Page** • Implements Chatroom Name & Code Display with **states** and **inherited methods**: ○ States ▪ activeChatroomID ▪ activeChatroomName ▪ activeStringKey ○ Inherited Methods ▪ setViews () ▪ setActiveChatroom () ▪ getActiveChatroomID() ▪ getActiveChatroomName() ▪ getActiveStringKey ()
ViewChatroomContent.js	• Display on **Lower Right Div** of **Chatroom Page**

Name	Function

	• Implements Chatroom Name & Code Display with **states**, **methods** and **inherited methods**:
	  ○ States
	  ▪ activeChatroomID
	  ▪ activeChatroomName
	  ▪ activeStringKey
	  ○ Methods
	  ▪ clearActiveChatroomTypeBox()
	  ▪ checkMsg()
	  ▪ refreshMsgPack()
	  ○ Inherited Methods
	  ▪ getActiveChatroomMsgPack ()
	  ▪ getActiveLastMsgId()
	  ▪ getActiveStringKey ()
	  ▪ getActiveChatroomID()
	  ▪ setActiveChatroom ()
	  ▪ setActiveChatroomMsgPack ()
	  ▪ setActiveLastMsgId()

Back-end: ExpressJS

- Hosted on: https://chatroom-hk.herokuapp.com/api/
- Routes are coded in **/chatroom-backend/routes/api.js**
- Route Functionalities

Name	Route	Attributes	Function

/load	GET	-	• Determine if end user has signed in by session variables
			  ○ If yes, make query to database to obtain username, chatroom list and display name of the user
			  ○ If no, fetch error message

/login	POST	username; password	• Determine if end user possess correct login credentials by making query to database
			  ○ If yes, set session variables **userId, displayName & username** for the user, and fetch JSON response composing of username, chatroom list and display name of the user
			  ○ If no, fetch error message

/logout	GET	-	• Unset session variables **userId, displayName & username** for the user, and fetch null JSON response
/register	POST	username; password; displayName	• Determine if the provided username is already taken by making query to user database

Name	Route	Attributes	Function
			○ If taken, fetch JSON response with corresponding error message ○ If not taken, make query to create a new user in the database with the supplied information, and send JSON response with username, chatroom list and display name of the user ▪ If unsuccessful, fetch error message
/createchatroom	POST	name; session.username	• Generate a new 6-digit Chatroom Code by giving a random string that consists of lower case, uppercase letters and numbers • Determine if the newly generated is already taken by making query to user database ○ If taken, repeat the generation process ○ If not taken, make query to create a new chatroom in the database with the supplied information, and send JSON response with string key, chatroom name and chatroom id for the user • If unsuccessful, fetch error message
/appendChatroomToList	POST	chatroomName; chatroomKey; session.username	• "Upsert" the provided chatroom information (i.e. Chatroom Name & Key) to the corresponding user identified by username ○ If successful, send JSON response with result: success for the user ○ If unsuccessful, fetch error message
/getchat/:key	GET	params.key; session.username	• Load chatroom information and all messages of that chatroom to the user by making query to user, chatroom and messages database ○ If successful, send JSON response with chatroomName, chatroomID, stringKey, largestMsgId and msgPack

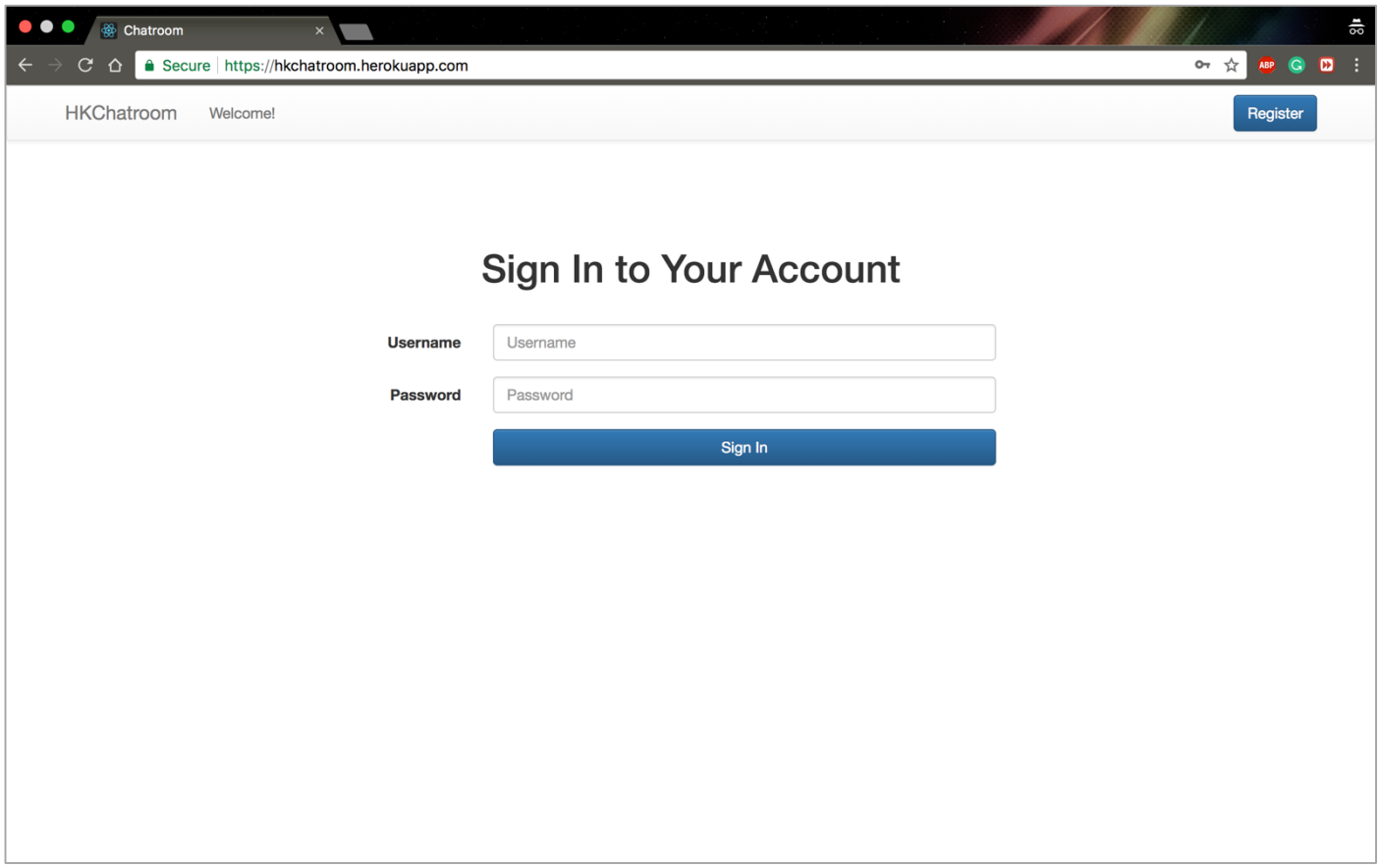
Name	Route	Attributes	Function
			(i.e. an array containing message id, isSelf toggle, isDateAppeared toggle, sender id, sender name, content, date and time for each message ○ If unsuccessful, fetch error message
/checkNewMsg/:key	GET	params.key; session.username	• Provide a quick and low network usage API to check if end user has the most updated messages for the chatroom current viewing by making query to chatroom and messages database ○ If successful, send JSON response with a MsgId indicating the ID of the most recently created message at the corresponding chatroom ○ If unsuccessful, fetch error message
/postmessage/:chatroomid	POST	content; date; time; params.chatroomid; session.userId session.username; session.displayName	• Insert the provided message contents (i.e. content, date, time) with the corresponding chatroom information (indicated by params) and user information (indicated by session variables) ○ If successful, send JSON response with message ID of the newly inserted message ○ If unsuccessful, fetch error message

Database: mLab MongoDB

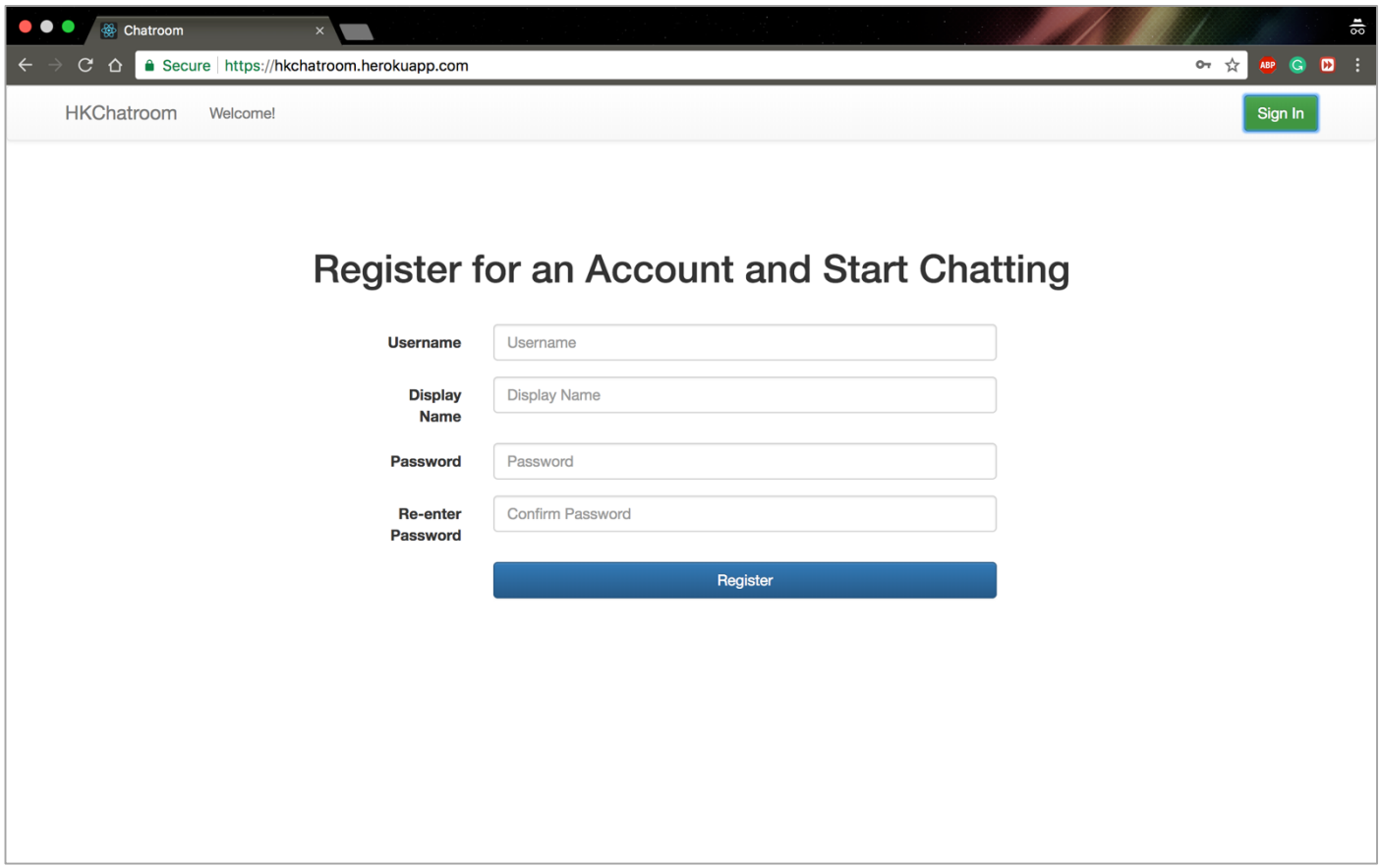
- Hosted on: mongodb://<dbuser>:<dbpassword>@ds233238.mlab.com:33238/chat101
- Free Sandbox account
- 0.5GB storage space

User Interface Snapshots

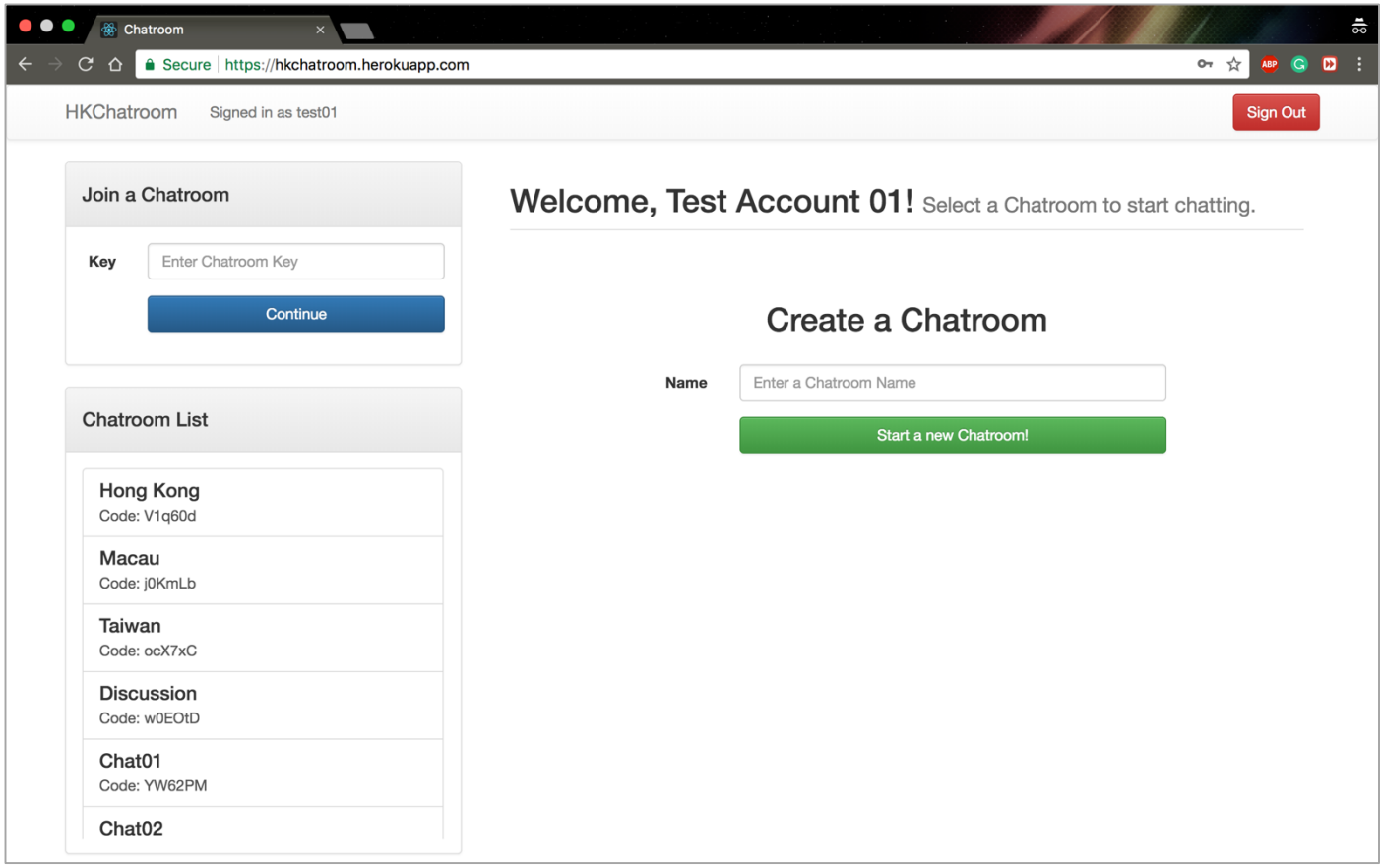
- Sign In Page



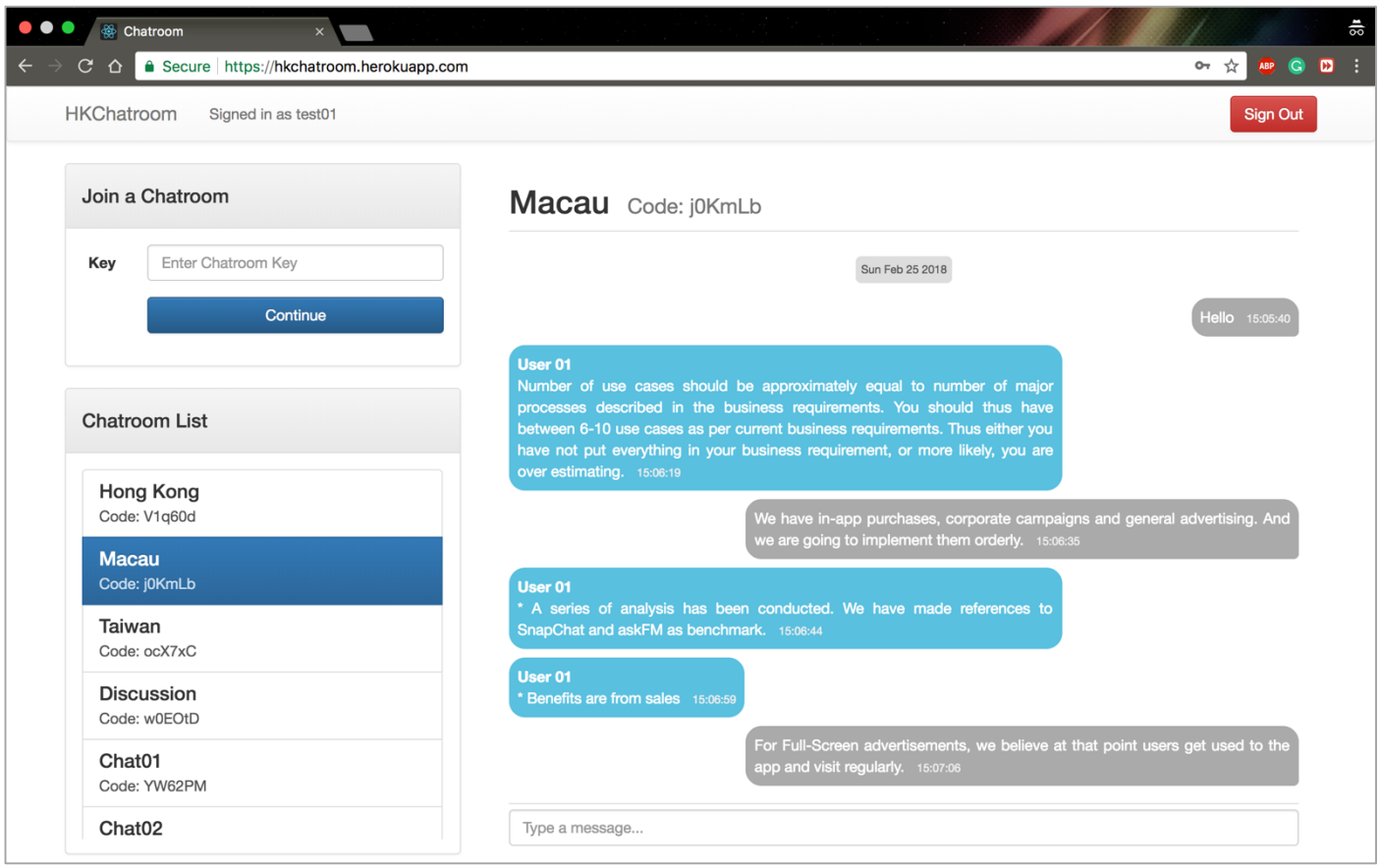
- Registration Page



- Dashboard Page



- Chatroom Page



Test Accounts

- Several accounts have been set up with the deployed application to facilitate views

No.	Username	Password	Remarks
1	test01	123456	Multiple Chatrooms, Chatroom named “Discussion”
2	test02	123456	Chatroom named “Discussion”
3	test03	123456	Clean account

Remarks

- HKChatroom is deployed on free accounts of Heroku and mLab
- In case of any behavioural abnormalities, please contact administrator at jeffreywlchung@gmail.com for assistance