**Link to code:** <http://brockweekley.com/chat/dist/chat>

## I created an angular frontend app that relies on a firestore database to store messages entered by users in order to synchronize messages in realtime, essentially creating a group chatting system. Visiting the site will display the chat log and all of the current messages sent by users. You can submit your own message by pressing enter and clear the log for all users with the clear button.

## What framework did you choose?

I chose to work with angular and firestore as I have worked with angular in the past and was interested in learning a noSQL database that can be used as a backend for the angular framework.

**What did you learn about this framework?**

I learned a great deal about firestore and some new familiarizations with angular. In generating the angular app, I learned about production files and the dist folder, which I am hosting on my server. I also learned a great deal about firestore and angularfire. Firestore is an online database managment solution that integrates into angular through angularfire. I created a new firestore database and learned about document management, which is essentially the way that data is linked together. Each document nests into other documents and every document has an id. For my database, I am storing messages that users send on the firestore and then pulling them back down for every user to see, creating a colaborative real-time chat log. Firestore is mostly used for syncing data in real time and this is achieving that entirely. I also used an http call to a free API called timezone in order to get a global time that the user is sending the message.

In terms of angular, I have an understanding of component based architecture, where all components are generated to display on one dynamic page based on routing. I also have an understanding of html angular such as ngFor and ngIf as well as ngModel, which are tags used to inject typescript into html. Finally, I understand services and creating get/post requests to API’s through HTTP and how the angular environment functions.

## What resources did you utilize?

### Angular

[https://angular.io](https://angular.io/) - Basic angular information and syntax <https://angular.io/guide/http> - Angular information on HTTP calls <https://angular-templates.io/tutorials/about/angular-crud-with-firebase> - Angularfire tutorial for CRUD operations, I followed this and made CRD operations <https://github.com/angular/angularfire2/blob/master/docs/rtdb/lists.md> - Angularfire examples <https://alligator.io/angular/cloud-firestore-angularfire/> - Introductory Angularfire usage

### Firestore

<https://firebase.google.com/docs/firestore/> - Basic firestore information and syntax <https://firebase.google.com/docs/reference/js/firebase.firestore> - Using firestore vs firebase <https://medium.com/@coderonfleek/firebase-firestore-and-angular-todo-list-application-d0fe760f6bca> - Firestore tutorial that helped me fix some operations in my CRD <https://expertise.jetruby.com/firebase-realtime-database-the-best-way-to-sync-your-data-192ac2b17d3> - Firestore tutorial on synchronizing data <https://itnext.io/how-to-crud-in-angular-firebase-firestore-456353d7c62> - Angularfire tutorial that helped with CRUD operations

### Timezone

<https://timezonedb.com/references/get-time-zone> - Free API to get current time for CDT

## What problems did you run into?

The first problem I ran into was hosting the angular app on my AWS server. I was able to do this with Professor Wergeles help by building the project to production on local and then hosting the dist folder on the server.

The second problem I had was creating a link to my firestore database. The tutorials above about crud operations were able to show me how to make a successful link after adding the header information to my environment, such as the API key.

The third problem that I had was a bug I discovered after launching on my server. I had a friend hop on to my chat and send some messages with me, when I discovered some of the messages were out of order. I discovered this was because the program was gathering our local system clocks time whenever it was sending the time with our messages, because of this, our clocks were out of sync and messages could display before already sent ones if the system clocks were ahead or behind. To fix this, I found a free API called timezone that gets the current time for whatever timezone you select. I selected our central time and was able to synchronize all users’ times on the app.