Okra Team
Startup 07 Submission

Special Server Setup Procedure

Our server does not require additional setup besides npm install and node src/server.js.

Main Feed- Nick Achin

The main feed now uses the ranking algorithm taking “Jane Doe’s” skills and interests and matching them on the job openings available. The main feed uses the “feed” collections for the feeds and jobItems/notificationItems to populate the feed.

BUGS: Unfortunately, our transition to the database did not go smoothly and the main feed is not properly populating the UI.

Job Board - Josh T.

Project Page - Thomas Cloutier

The project page now pulls all of it’s data from the online database. The specific collections that the page uses are the “project” collections for all of the specific project data, the “positions” collection for all of the open and filled position information for the project, and the “notificationItems” for all of the specific updates for the project.

Dropped Features:

-For this page the apply button does change upon interaction but fails to do anything based on the fact that we never implemented user sign-in and user creation figures. Ideally we would have like this to be used as a way for individual users to apply to projects with open positions that the user felt

- The following button was dropped for many of the same reasons. Also populating the user’s feed with random project updates made the main-feed too cluttered

-Messages to projects were dropped as we felt that having users message projects was not a good feature. Also time constraints would not allow for it

Create Project - Ryan LeCours

Profile Page - Ryan Minichiello

The profile page successfully pulls the user data from the mongo database. Specifically, the user collection for "Jane Doe" is pulled. It is also rendering the project pills just as the sidebar is.

Josh and I implemented the ranking algorithm, and I successfully ported it to the job board and the main feed. This was the central feature of the application, so it felt good to finally implement it and get it working the way we intended. It's a bit naive and not scalable because it will scan all items in the database.

R = $((user\ skills \cup user\ interests) \cap (project\ tags).length) / (project\ tags).length)$)

If (R > .66 ranking = GOLD, R < .33 ranking = BRONZE, else R = ranking = SILVER)

We as a team configured the database, and once the first domino of serving from mongo fell with profile and project, the rest of the components fell into place.

<u>Dropped Features</u>

WIth the dropping of the inbox feature, the user's messaging function will no longer work. User "Jane" is pretty hard coded throughout the application, but as a single user application requirement for demo, her data flows down to components not only used by her, but throughout the rest of the components in the application.

Inbox Page - Caroline Kim

- We ended up dropping the Inbox Page.
- But before it was dropped, it was sending chat data to mongo_express, and the banner on the right side wasn't static anymore.
- The main reason for dropping this page was because we couldn't implement the onPost or onClick to click around different chats or send any messages. Due to this problem, there was no purpose of having this inbox page. So we removed it from the top navbar and got rid of all the inbox components.

After getting rid of the inbox page, I helped other people put up the data in mongodb and rendering from there.