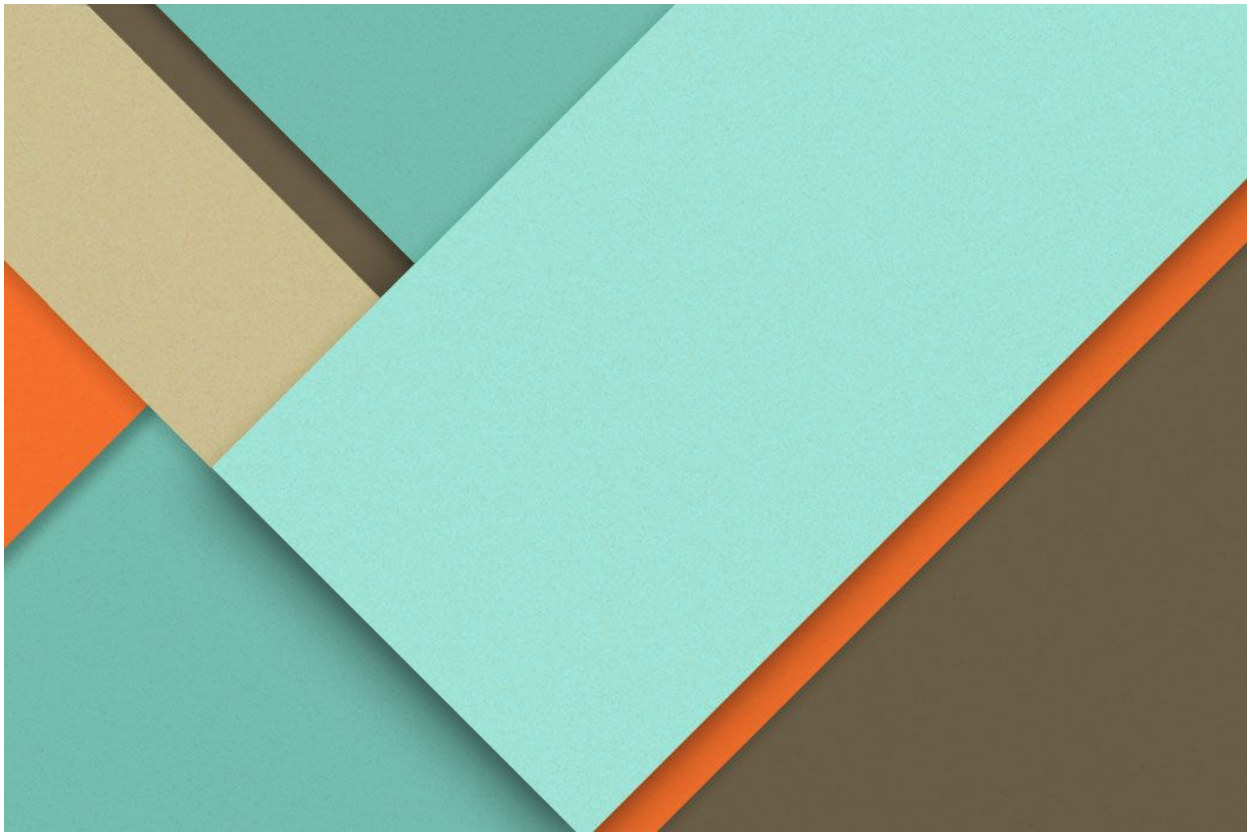

Imperium

Team 15

Jeremy Potratz
CJ Enright
Anthony Popilov
Peter Reckamp
Jacob Szarzynski

What went well

This sprint we finished almost all of our objectives. We were able to work cohesively as a group to get things done, and are pretty happy with the end result. At this point the app itself is starting to come together with a plethora of features that make both using it and developing it quite nice. For the next sprint we'll have to work just as hard to keep this level of quality, but are hopeful that we can do it.

Git

Git has once again proven itself invaluable. We had a lot of time over spring break where we just worked away on our own tasks without the need for coordination because it was all being handled by git. For future sprint we may consider using feature branches to avoid some merge conflicts and other VCS overhead, but as a whole it made us much more productive as developers.

React

React has allowed us to make composable elements, which has saved development time and led to a smaller code base. For example, we could define a generic card element, and use it in any user interface that needs a card component. Not only does this save development time, but it also means our design can become more cohesive.

User Stories

User Story 1

I would like to upload my resume so that companies can learn more about me

Description	Assignee	Work Estimate
Create view to accept resume uploads	Jacob	3 hours
Create handler for resume uploads in the backend	Jeremy	4 hours
Create logic to store uploaded resumes in Firebase	CJ	2 hours
Write tests to ensure the upload resume handler functions correctly	Peter	2 hours

Completed: A user is able to upload resumes as a pdf to their profile as a student. The resumes are uploaded with express-fileupload then stored in Firebase. We update the resumes in the

accounts tab and then send the updated resume to Firebase. The resumes are viewable by both the students, and the prospective employers while in the matching process.

User Story 2

As a user, I would like to upload a photo of myself/my company so that companies can see more about me.

Description	Assignee	Work Estimate
Create view to accept pictures via Camera or uploading from the user's computer.	Jacob/Anthony	2
Create handler for photos in the backend	CJ	2
Create logic to store uploaded photos in Firebase	Jeremy	2
Create view to see/update uploaded photo	Anthony	3
Write tests to ensure that the photo was successfully updated	Peter	2

Completed: A user is able to upload photos to their profile as both a student and employer. The photos are uploaded with express-fileupload then stored in Firebase. We update the photos in the accounts tab and then send the updated photo to Firebase. While looking for a match, all users are able to see each individual's photos.

User Story 3

I would like to view student profiles so that I can get a better understanding of a student

Description	Assignee	Work Estimate
Create view for viewing student profiles	Jacob/Anthony	6 hours
Write handler for retrieving data for student view to use	CJ	3 hours
Write logic to retrieve student data from firebase	Jeremy	3 hours
Write tests to ensure retrieving student works and doesn't send sensitive data	Peter	3 hours

Completed: After properly signing up as an employer, the user is able to look through a group of student users that match their criteria for a future employee. This is done at the home page

where an array of student users are pulled from Firebase after matching criteria like wage, major, start date, and more. All of the students information is viewable except for their private settings like their password.

User Story 4

I would like to be able to sign sign up as an employer with a criteria instead of bio.

Description	Assignee	Work Estimate
Create view for signing up as an employer	Jacob/Anthony	1.5 hours
Update handler to store employer data in firebase	Jeremy	1 hours
Update logic to retrieve employer data from firebase	CJ	2 hours
Add a list for employer's criteria and which major they are primarily seeking to employ	Jacob/Anthony	1.5 hours
Write tests to ensure successful signup as employer	Peter	2 hours

Completed: In order to implement this we had to change the sign up and preferences view to show certain fields based on the type of user. React ended up making this pretty straightforward. In the database students and employers are stored in the same table, but are differentiated by a "persona" field.

User Story 5

I would like to be able to view employer profiles so that I can see which companies are hiring and what their preferences are

Description	Assignee	Work Estimate
Create view for viewing employers profiles	Anthony/Jacob	4 hours
Write handler for retrieving data for employers view to use	CJ	3 hours
Write logic to retrieve employer data from firebase	Jeremy	2 hours
Write tests to ensure retrieving employer works and doesn't send sensitive data	Peter	3 hours

Completed: After signing up as a student, a user is able to view employers that match their preferences on their home page. The matching process is done on the home page and only shows employers that match criteria of the student like desired wage, start date, and more.

User Story 6

I would like to create job specific preferences about what I want so that companies can more effectively find me

Description	Assignee	Work Estimate
Add to the account view options for requested wage, full or part time, and start and end dates	Jacob	2 hours
Write a handler for retrieving data for the new options	CJ	3 hours
Add logic to receive more options in the firebase	Jeremy	2 hours
Write tests to ensure the options are updated correctly in the database	Peter	2 hours

Completed: This involved adding a bunch of fields to students so that we can better match them up with employers. This also complicated how we find those matches, adding a lot more constraints, but that ended up making our matching a lot more effective for finding suitable jobs.

User Story 7

I would like to specify a radius of where I am interested in having a job

Description	Assignee	Work Estimate
Add an option where the user can select which regions they would like to look for jobs	Jacob	2 hours
Write a handler for retrieving data for the regions	Jeremy	3 hours
Write logic to receive regions in the firebase	CJ	2 hours
Write tests to ensure the region is correctly stored in the database	Peter	2 hours

Completed: When signing up for an account, a user is prompted with four check boxes to decide what region they are looking for a job in. These regions are the South, West, Midwest, and

Northeast. By selecting the different boxes, the user is able to designate more than one possible region to work in.

User Story 8

I would like to specify whether I want a job, internship, or co-op

Description	Assignee	Work Estimate
Add an option where the employer can specify whether they want a job, internship, or co-op	Jacob	2 hours
Write a handler for storing the type of job	Jeremy	3 hours
Write logic to store type of job in the firebase	CJ	2 hours
Write tests to ensure the type of job is correctly stored in the database	Peter	2 hours

Completed: This one was pretty straightforward, but again required us to add more constraints to our match finder. The user can only specify one type of job, so that's stored as a string in the user's row in the user table.

User Story 9

I would like to specify that I am interested in jobs for more than 1 major

Description	Assignee	Work Estimate
Add an option where the student can specify which major they are	Jacob	2 hours
Add an option where students can specify which minor they are	CJ	2 hours
Add on the view a button that will allow the user to add more than one major or minor if necessary	CJ	2 hours
Write a handler for storing majors and minors	Jeremy	3 hours
Write logic to store majors and minors to firebase	Anthony	2 hours
Write tests to ensure the majors and minors are successfully added in the database	Peter	2 hours

Completed: We ended up solving this by asking users to separate their majors by commas. This is a pretty common pattern on websites so we figured users should be somewhat accustomed to

it. When searching for matches, we turn the individual majors into an array of strings and see if any of them match with a given job.

User Story 10

I would like to be able to like employers that I'm interested in so that I can show interest in those employers

Description	Assignee	Work Estimate
Add to the home page view a photo of the company as well as basic information from their bio	Jacob	2 hours
Create buttons to either like or dislike	Jacob	2 hours
After liking or disliking have the next company card show up	Jacob	2 hours
Write a handler for storing the like or dislike on the employer	Jeremy	3 hours
Write logic to store the like or dislike from firebase	Jeremy	2 hours
Write tests to ensure the process of liking or disking is executed correctly	Peter	4 hours

Completed: Within the home page we enabled student users to like an employer profile when they were matched in order to show the company that they are either interested or disinterested in them. We achieved this by creating "Like" and "Dislike" buttons that added the employer to your list of liked or disliked companies.

User Story 11

I would like to 'favorite' up to 3 companies to show that I am extra interested

Description	Assignee	Work Estimate
Add a button where a user can favorite a company	Anthony	1 hour
Write logic to keep track that only 3 companies may be favorited at a time	CJ	3 hour
Write logic to store that an employer has been favorited to firebase	Jeremy	2 hours
Add an option to remove favorites	CJ	2 hour

Write tests to ensure that the process of favoriting is executed correctly	Peter	3 hours

Completed: This was accomplished in a similar way to liking a user story, but had more constraints attached to it. Because of that it took a bit longer than expected. A student is able to favorite an employer, but an employer cannot favorite a student.

User Story 12

I would like to unmatch with an employer so that I can adjust my interests over time

Description	Assignee	Work Estimate
Add a button where students can unmatch with employers	Jacob	1 hour
Write a handler for updating the matched status	CJ	2 hours
Write logic that will update the match in the firebase	Jeremy	3 hours
Write test cases that will make sure users are successfully unmatched	Peter	2 hours

Completed: After a student user and employer show mutual interest in each other through liking each others profiles, the users show up in their respective matches category. The matches page shows all the people that you have liked and that have liked you. In that page, you are given an option on the users matched card to unmatch them. This shows up as an "unmatch" button on the right hand side of the matched users profile. After selecting that button, it removes the user from your matched list.

User Story 13

As an employer I would like to create a bio for the specific job/internship

Description	Assignee	Work Estimate
Add a text box on the sign up page for the company to describe the position which they are advertising	Jacob	1 hours
Write a handler for storing the bio of the position	Jeremy	2 hours
Write logic to store the bio from firebase	CJ	3 hours
Write test cases to ensure that the process of creating a	Anthony	2 hours

position bio is executed correctly		

Completed: The bio for the job position was very similar to the implementation we have in the student user. We were able to use similar code for the bio of a user to create and update a bio for the job. We achieved this by making a text box in the sign up page for employers, in which they could add their positions bio.

User Story 14

I would like to be able to like students that I'm interested so that we can begin to build a professional relationship

Description	Assignee	Work Estimate
Add to the home page view a photo of students as well as basic information from their bio	Jacob	2 hours
Create buttons to either like or dislike	Jacob	2 hours
After liking or disliking have the next student card show up	Jacob	2 hours
Write a handler for storing the like or dislike on the student	Jeremy	3 hours
Write logic to receive the like or dislike from firebase	Jeremy	2 hours
Write tests to ensure the process of liking or disking is executed correctly	Anthony	4 hours

Completed: Within the home page we enabled employers users to like an student profile when they were matched in order to show the company that they are either interested or disinterested in them. We achieved this by creating "Like" and "Dislike" buttons that added the student to your list of liked or disliked companies

What did not go well?

This sprint we ended up not finishing a user story, this was mainly due to our tests not being robust enough and not testing everything a reasonable amount of time before the sprint demo. As we started to run through everything and practice for the presentation, a lot of problems arose, such as profile photos not all being the same size. We ended up getting overwhelmed and ran out of time and missed the opportunity to re-test our favorite

feature. Unfortunately it wasn't until during the demo that we found out it wasn't fully functional. Additionally, our pacing was bad this sprint. Most of the work was done during spring break and during the final week of the sprint. Also our user story planning was poor this time around. We did not include some user stories that would have worked really well with some of the stories we had on the plan. Consequently we ended up essentially having to do those stories anyways for the sake of functionality.

User story 11

Write logic to keep track that only 3 companies may be favorited at a time	CJ	3 hour

Not completed: This was implemented in the backend, so a user cannot have more than 3 favorites in the database. However, we did not correctly handle/display the errors sent from the backend on the frontend. This allowed a user to appear as if they favorited more than 3 companies, but in reality they are not.

How should you improve?

1. In order to effectively complete all the user stories on time, we as a group should manage our time more effectively. Nearing the end of the past two sprints, our group has saved a large chunk of work for the day or two before our sprint review. Efficient use of time will result in easier completion of user stories in a less rushed, more professional manner.
2. Another important aspect to note, is that we should be more careful when filling out our sprint planning. We must be careful to pursue all the necessary user stories, while also not biting off more than we can chew. As a result of a poorly done Sprint 2 Plan, some aspects of the sprint were not properly communicated within the group, and as a result, were not properly completed in a time-effective manner.
3. One more key idea for improvement, is to figure out directly which user stories take priority over others. Those that prioritize functionality over aesthetics are obviously more crucial, but it is imperative that each member of the group knows his role and what to work on first. For this sprint there were occasions where work could not progress until a blocking user story was finished. We'd like to avoid that in the future.