# Table of contents

Page 2: IT Professional Skills – project proposal.

Page 3: Possible technologies.

Page 4: Issues encountered.

Page 4: Architecture overview.

Page 6: Technologies used.

Page 7: Future development.

Page 8: Sprint 1 review.

Page 9: Sprint 2 review.

Page 9: Features of implementation.

Page 10: Middleware.

Page 11: Development lifecycle.

Page 11: Architecture of the solution

Page 13: Design methodology

Page 15: Testing

Page 17: Conclusion

Page 18: Appendix of OneNote

# IT Professional skills - Project proposal

Sean Skelton: G00361169

-Fionn Browne: G00373152

Supervisor: Joseph Corr

GitHub: <https://github.com/swordslinger/Covid-19-Statistics-App>

## ==>Overall Description<==

This will be a web application created from JavaScript HTML, basic CSS and react, it will display a map of Ireland and provide information on COVID-19 which will be read in from an API. The user will also be able to Register an account on the web page that will be saved to a database.

We plan on this application being a single user system. Also, as we are not handling any financial, private, copyrighted material or sensitive data there was no strict rules or regulations we had to comply with when building this application.

The application will produce the webpages. The Database will be MongoDB This system will require static updates; the program will be localized to English speaking demographic.

## ==>Architecture diagram<==

Graphical user interface, application

Description automatically generated

## ==>Technologies we plan to use<==

We plan on using JavaScript, mongo DB, visual studio code, react.js, CSS, HSE Covid-19 API, react, react-leaflet and node with assorted packages, GitHub for source control and OneNote and Jira for collaboration and documentation of this project.

## ==>Issues the project is trying to solve<==

The main problem this project is trying to solve is to provide Centralized view of for the daily COVID-19 cases in Ireland with user friendliness in mind.

## ==>Project benefits<==

It provides a straightforward platform for people of any age group too interact with and keep on top of the daily cases giving indications as to how close or far away we are to returning too pre-Covid-19 times.

## ==>Project Objectives<==

The design of the application has simplicity at its core. The Objective of this project is to provide user friendly view of the COVID statistics that web app that is usable by any generation.

To make all people more informed through the display of number of cases, total cases, number of deaths and total deaths, hospitalization rates etc. To make people aware of how it is spread three data screens for entry with the Login, Register an account & search. The System will produce the webpages. The Database will be MongoDB that will save users Email and password for data entry.

## ==>Data Entry<==

There will be three data entry screens which are as follows Login, Registration and an API search. Login will take in an email and password from the user as will Registration. The API search will take a string that is 3 characters long, ensure it is numeric characters.

## ==>Minimum hardware requirements<==

The hardware requirements for this application to run is 5MB of memory allocation you will Windows 7, Windows 8, Windows 8.1, Windows 10 or later An Intel Pentium 4 processor or later that's SSE3 capable.

## ==>Limitations<==

The number of COVID-19 infections is only updated every two days in the API thus you may have to wait an extra day to get the number of infections. Difficulty to work on the project together as we cannot work in the same room due to lockdown. Multiple assignments and exams on during development time for this project.

## ==>Milestone's<==

- Finish writing user stories and begin development.

- Complete core functionality.

- Have application fully developed.

- Have everything documented for the final report.

- Have manual testing for any large section of code.

# Possible technologies we looked at and compared.

## GeoEncoding

-ArcGIS – map building/ did not have time to build the skill set to use the tool

## Backend

-Technologies we looked at were Ruby, php and node.js

We decided to use Node.js as backend framework as it was the one, we were had the most familiarity and experience using, it integrates very well with non-relational databases whereas PHP is best suited for relational databases such as SQL, its lacking in libraries in comparison to other backend languages. Ruby is quite robust and modern but as this project was under a time constraint, we felt the time taking to learn this language and integrate into our project would have left us with an unfinished application.

## Database

-We chose to use Mongo DB as it would have the fattest databases too write and take information from and as we are dealing with minimal input fields and this application having a low user count in mind we do not need to be as worried about the structure or working with larger datasets that MySQL is suited for.

## Frontend

-So, we mainly stuck within JavaScript as this language Is one of the most used for web applications and the backend npm depends on using JavaScript, we looked at using either angular or react.

The reason we chose react is react is it was UI component design in mind and as the UI is central to this project and angular JS comes with a lot of fluff and specification files that we did not feel we were going to utilize in our project.

# Issues encountered.

We had a lot of issues to get Jira to work on our machines that when we had a meeting, we decided that Sean would do the testing with Jira and I would do manual testing until he could check my code as too much time had passed trying to get it to work.

I was trying to create a database with SQL and wamp, But I ran into some issues where the code would not connect to the database and would not return the name entered for the database.

In this project some of the issues we encountered revolved around the difficulties we had with being able to work together since with covid we are not able to be in the same room. And how much boiler plate JavaScript required.

Connecting to the database and retrieving data proved to cause some issues when I attempted to make it through wamps database, so I ended up using mongo dB to store account details for the user.

# Architecture overview of the system

For the server I used body parser to intercept the body of a http messaged passed as post request.

Then I used Cors to allow for multi-server connection from localhost:4000 to localhost:3000 so that I can make it into 1 service.

I used Mongoose to connect to the mongo DB and to take in the password & email from it where I made a schema so that would read in the email and password of each account.

I made a build of the project so that Localhost:4000 would get the front-end functionality where it is sending a file and joining two paths.

## Bot.js

Is used as a component I used to display the local time and date for each user using Card which formatted the output nicely and only required an opening and closing tag on a page after being imported.

## Home.js

Reads in the data from the Json blob API, It the stores the API data in a variable, it then uses dependency injection to pass the data from home to covid19.js.

## covid19.js

For splitting the array of data into single pieces and passing to covid19Item.js

## covid19Item.js

For displaying the individual pieces of COVID-19 data

## Edit.js

Edit is very similar to register.js however it differs by overwriting existing data in the database and saving the changes done to the account.

 .

## Images.js

Utilizing the Carousal JavaScript package, it puts 3 navigate able images at the top of screen on rotation.

## login.js

takes in users email and password and persist it to the database.

## register.js

Imports registerStyle.css for styling the pages attributes. It then takes in values for email and password on submitting the details entered will be displayed as an alert on the page and then it sends the data to the server which then loads the data from there it creates a new account in mongo DB.

## registerStyle.css

To stylize the input and messages used in register.js.

## users.js

For splitting the array of data into single pieces and passing to useritem.js with a reload functionality

## UsersItem.js

For displaying the email and password of user with delete user functionality then reloads the page

## App.js

This uses a navbar too transverse all the different components in this application.

## MapOfIreland.js

This component takes covid 19 data from the API using an asynchronous call and saves it too it too a react state.

It then creates an instance of the leaflet map and places a tile layer over it, think of this as formatting for the map that contains details for the country such as roads, rivers etc. and then use json data from a file in the project to render a map on screen of Ireland drawn with polygons.

It then populates this map with a marker that is based on the CSS styling in mapOfIreland.css and we then add a popup to this marker so that when it clicked it displays the number of cases that day.

## Towns. Json

 just contains all the attributes from the API but we downloaded it locally to access the coordinates for the polygons too draw it on screen.

server. Js Used for persisting information to the database, to build the production of the application and for handling get, put, post, update and delete requests.

searchAPI.js Puts a js variable into a URL too search the API based on the FID that returns information from the relevant entry, the most recent value can be retrieved from the pop in the map of Ireland and then used to populate the search bar.

# Technologies used.

## ==>Technologies we plan to use<==

## JavaScript ->

**JavaScript** (or also called **JS**) is a lightweight, interpreted, object-oriented language with first-class functions, which is used a lot for scripting language for web pages.

## MongoDB ->

**MongoDB** is a document-oriented NoSQL database used for high volume data storage. It makes use of collections and documents. Documents consist of key-value pairs which are the basic unit of data in **MongoDB**.

## Visual Studio Code ->

Is a lightweight but powerful source **code** editor, it comes with built-in support for JavaScript, TypeScript and Node.

## React.js ->

IIs an open-source, front end, JavaScript library for building user interfaces or UI components. But is usually used for single page application development and some of its features include being fast, scalable and the capacity to manipulate data without leaving a webpage:CSS ->

Stands for Cascading Style Sheets. **CSS** describes how HTML elements are to be displayed on screen, paper, or in other media. **CSS** saves a lot of work. It can control the layout of multiple web pages all at once.

## API ->

Is a set of programming code that enables data transmission between two pieces of software products/applications. Technical specification describing the data exchange options between solutions with the specification done in the form of a request for processing and data delivery protocols.

## Jira ->

This software is used for bug tracking, issue tracking, and project management.

## Selenium ->

Is a portable framework for **testing** web applications. **Selenium** provides a playback tool for authoring functional **tests** without the need to learn a **test** scripting language.

## CSS->

Stands for Cascading Style Sheets. CSS describes how HTML elements are to be displayed on screen, paper, or in other media. CSS saves a lot of work. It can control the layout of multiple web pages all at once.

# Future development.

* In the future we were thinking of adding emailing system so that every forte night an updating email is sent out to registered accounts with the last 14-day cases of covid.

* Adding shading of a range from green to red with covid 19 cases density in each county for easy visuals for the users.

* Adding a feature to where you can donate to local hospitals to support nurses and doctors. Which would require business regulations of encrypting the user’s info and transaction and a receipt service.

* Adding a feature that would ask for user permission to track their current location.

* Pop up for each County displaying the 14-day incidence rate.

* IF an API became available for locations of test centers, populating the map with the said test centers.

* Hosting the site on AWS server so it is available globally.

* Tracking of number of people with 1st dose of vaccination and the total number of people vaccinated.

# Sprint 1 review.

## Stories went out of scope.

Some of our stories were a bit out of scope as we ran into issues with creating a reliable database for logging information and getting multiple API to work together when there is a limited amount available for Ireland.

## Issues with consistent commitments.

With exams and other assignments at the same time with other responsibilities we found it difficult to have a consistent commit to GitHub.  As we were still trying to sort out the bugs before we committed the code.

## Issues encountered with sprint.

Some issues encountered in this sprint were using new software that took us a bit longer to get accustomed like, not being able to work with each other side by side for when one of us got caught with a specific section of a user Story.  It was difficult to keep OneNote, GitHub and Jira all up to date while developing the code.

## What went well?

We got a much better understanding of the scope of the user stories and what is required of us to finish the project. We learned how to manage time more appropriately in order of the scale of work given and how to organize workloads. We communicated with each other nearly daily checking up on where we stood what we need done and what our goal is for the next meeting to have finished.

## What would we improve on?

As we were working on this project, we found that it be better if we shared our screens over discord to show exact sections of code giving errors for the user story or to show what we want to develop on. As we tried to do our own thing for the first half of the sprint with the user stories, we agreed on but without visually seeing the others code we did not fully understand each other’s issues with a user story, or the progress made on the code.

# Sprint 2 review.

## Finishing off user stories

We completed the user stories by the end of our second sprint, we were glad that we communicated well and organized our branches and user stories well so that we would not get any merge conflicts and that it would compile correctly.

## Checking each other branches of code.

As each of us were working on different user stories we made sure to work them on separate branches from each other so that we would be able to track work done better and in case we ran into an error or accidentally deleted some code wed still have a backup which did come in useful a couple of times. Over the second project we paid more attention to each other’s work so that were comfortable with the pace we went and would know if someone is having difficulties or issues with the work.

## What went well?

We managed to work through the remainder of the user stories and have the manual testing done for the user stories. We communicated far more during this sprint and helped each other with code or finding work arounds for certain issues. We delegated the tasks evenly between each other and managed to get on top of the workload.

## Learning outcome?

We are far more confident with our coding abilities and time management skills. We are more experienced in managing user stories and researching issues we have encountered with code.

# Features of implementation

## ->Features of the written code.

* We tried too attempted a Modular design pattern and tried to keep only the relevant code in relevant components.
* Decoupled
* special code algorithms
* extensibility

## -> Localization.

* Specified to Ireland alone with its time zone.

->Accessibility.

* Will require a pc or a system that can have visual studio code installed.

## ->Navigation on the website.

* User can go between home, login, register, map of Ireland or health info using the navbar.

## ->Search.

* The capacity to search an API with the Unix formatted date.

## ->Security.

* Currently we do not have the password given by the users encrypted but we plan on have that done with future implementation.

## ->Popups.

* We used popups for displaying the last 14-day cases of covid, notifying a user that there account was created and for displaying that they might not have the correct format for their email or password entered the registration.

# Middleware.

## Body parser->

Allows us to intercept body of a http/json message passed as a post request and parse the relevant data we need for our application.

## Cors ->

(Cross-Origin Resource Sharing) allowed us to work with the two servers to share resources.

## Mongoose ->

we used Mongoose to connect to our mongo DB which then we needed a schema to define Gmail and password to define the structure of the document.

## Path ->

We used it to join two directories together.

## Bootstrap ->

We used it for adding features such as carousel or cards to the project.

## Axios ->

Used to post to the server which then checks the server for existing credentials.

## Nodemon ->

We used this just to help with developing the code as it automatically restarts the node application when file changes in the directory are detected.

## Express ->

Is a routing middleware web framework which allowed us access to call back functions, it handles different http requests and helps with routing the application?

# Development life cycle use

We tried to apply the development life cycle of scrum that is a methodology within agile and we had weekly meetings, user stories and two time boxed 3 week sprints, we did sprint plans and sprint reviews, We struggled with keeping consistent with the time boxed 3 week sprints as the amount of work that we could do  each week varied and we often found that implementing a piece of functionality wasn’t as clear cut as we had initially thought and had to go back to the drawing board mid sprint which ended in some commits happening after a sprint had been completed

Using this methodology, it did give us a very clear vision of what we wanted this project to do and what the finished project will be.

# Architecture of the solution

Black lines with denotated with two arrow heads denote links between two pages.

Dashed lined with singular black arrowhead denotes a reference to another class with the component with the with the dashed line coming out of it containing a reference to the component on the receiving end of the arrow.

Chart, line chart, scatter chart

Description automatically generated

Double dashed lines with arrow heads denote transfer of data between components.

Chart, scatter chart

Description automatically generated

Diagram, engineering drawing

Description automatically generated

# Design methodology.

During this project we relied heavily on the use of the mind map to understand what the key objectives were for this project and what were our key values. This changed over time and we drifted a bit during the process of development & planning, but we always reviewed the diagram so that we could be grounded.

Diagram

Description automatically generated

 -> We then use Gantt diagram to plan out the time allocation for this project, we managed to stick well to this during the process of this project. We had to reformat this chart a couple times as our project coordinator gave us new input on what should be included and what we can leave out.  Where we went from initially this in the project proposal.

Chart

Description automatically generated with medium confidence

-> To this. With the Gantt diagram in particular we referred back to it when some of the user stories went out of scope for sprint one or when we were thinking of what area we should cover next in order to keep a steady workflow going whereby we kept chipping away at this project at all times.

Chart, bar chart

Description automatically generated

->We had weekly meetings with each other that started with discussion of project candidates and evolved into the covid statistics tracker and then these meetings became focused on updates with our progress and things we were trying to implement, what was working what was not and how we were going to fix it.

->We also had weekly meetings with our project coordinator which was centered around the planning of the project and helped steer us in the right direction and make sure we were keeping in scope and prioritizing the correct things.

Graphical user interface, application

Description automatically generated

->We used very basic architecture overview we used in the beginning to work out how the project is going to work and how the different parts will communicate with each other and used to then used this too aid in our development.

# Testing

## Manual Testing

The key concept of manual testing is to ensure that the application is error free, and it is working in conformance to the specified functional requirements. Our functional requirements in this project were the navbar and all its links work without any issue. This testing checks the quality of the system and delivers bug-free product to the customer. It is very tedious but very reliable.

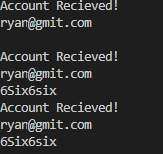
Initially tried using jest a unit testing framework for JavaScript but we felt the time needed to be invested into developing the skillet for this would have been best invested in developing, planning and research, we instead opted to use manual testing throughout the application as talked about above along with selenium tests below, a full suite of test and test plan in repository.

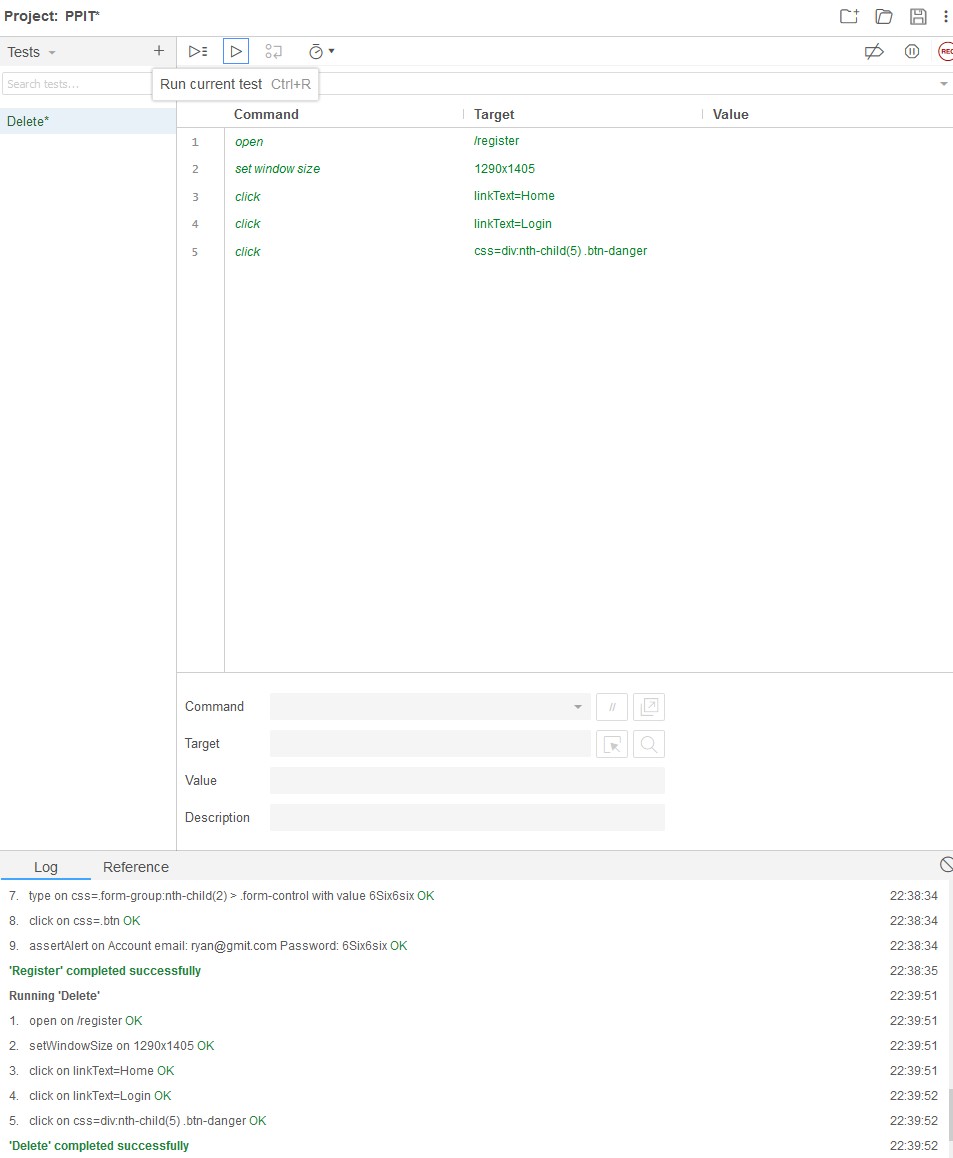
## Selenium

Testing Selenium is a test automation framework that allows you to automate web app testing. you can write test scripts to run against browsers.

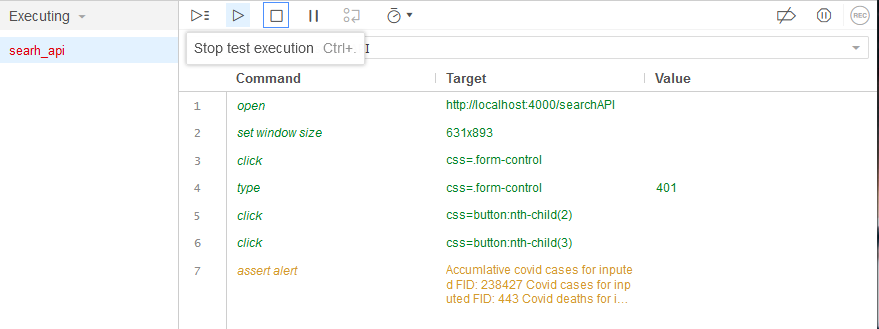
We used a culmination of manual tests and acceptance testing to ensure a functioning application, please see examples below.

Manual test to see if server received username and password.



Selenium test for navbar

And selenium test for mapOfIreland



Test cannot seem to reach alert, if the alert returns the data in the in the target it indicates it is working as intended.

# Conclusion.

## Summary

This project was a big formative learning experience for me and my colleague dealing with all aspects of the project from the planning and research to the more technical aspects of coding and problem solving and even soft skills such as teamwork, communication and working in cohesion.

## What went well?

We were working on this project consistently we were always communicating at a minimum once a week and only 1 meeting was missed by each member of them team, and other member filled in the other member on what was missed with the project coordinator, we never had any merge conflicts.

## What did not go Well?

One big issue was developing this project in the middle of pandemic so we could not meet face to face nor being able to be on campus, the implementation of agile/scrum life cycle as this was designed for larger teams and we felt that using the Kanban approach would have been much more manageable and streamlined for this kind of project.

## What we learned

We learned about the intricates of JavaScript and react and developing a fully-fledged web application, the importance of good practices such as the single responsibility, maintaining multiple branches on git so we could isolate each other's work and not conflict. the importance of understanding limitations of certain features and thorough research so you do not end up trying too code something that is not feasible from the beginning.

## What we would do differently

For the map we would have taking a more html orientated approach as a lot of the resources available for react-leaflet seem to use html more than JavaScript. We would have chosen a Kanban approach over scrum as a lot of time was spent trying to optimize our use of Jira and the iterative development was very difficult for us to manage and be consistent with, we felt it was ultimately detrimental to the quality of the project with the time available.

We would also have liked to do more graphical displays of data using data returned from API too draw interactive charts such as returning covid cases within a particular age range over a specified period.

# Appendix of OneNote

## Diary Entry

### Week-1

25th-31st January

In our first week Sean and I both agreed on being a team for this project, we discussed what things we are passionate about and our strengths and weaknesses.

We started to write down possible ideas for our project, we had such things in mind like a crypto currency trade simulator but all crypto API’s we could find were behind pay walls, so we were not able to get them. We also thought of the idea of making a game for this assignment and develop our skills in that area but realized that the gaming industry does not appeal to either of us, so we shelved that idea.

We then saw a notification of covid cases in Ireland and thought maybe making a covid app would be useful but more interactive with accurate information.

### Week-3

8th-14th February

This week we made an architecture diagram to show what our project would incorporate.

Graphical user interface, application

Description automatically generated

We then made a few 8 user stories to have our key functionality there and plan on adding more through the work process.

We then made our first draft of the project proposal which highlights the weeks of work and research, but we plan on asking our professor for tips on how to improve this and what areas of our documentation and development needs more attention and where we could make some changes to suit the industry standards better.

### Week-5

22nd- 28th February

We set up Jira and added the first complete user story that has been broken down into smaller tasks.

This week we wrote up some more user stories, some more epics and some tests.

Sean and I agreed to use Jira for its burndown chart and for the tests and user stories.

We updated the gnat diagram thanks to input from our coordinator.

Chart, bar chart

Description automatically generated

### Week-7

8th-14th March

This week Sean and I were working on developing and testing code for sprint 1. We encountered some setbacks as we had to spend some extra time revising for exams this week.

I have developed the home page and Sean has developed the map of Ireland. After our meeting on Wednesday, we felt like most of the documentation was up to the mark and we had to record the errors and issues encountered with the development of the code and merging it with the main branch in GitHub.

### Week-9

22nd-28th March

Due to responsibilities outside of this program we were unable to get much work done this week. But Sean and I still did the sprint review and planed the second sprint. We reflected what went well during the first sprint and what we could approve upon for sprint 2.

### Week-11

5th-11th April

This week Sean and I were finishing up the project, I managed to get the database and the backend server to work for taking in information from the user and then the user being able to edit or delete accounts.  Now when the user enters their password its encrypted and has restrictions such as 8 characters, one upper- and lower-case character and at least 1 number. And then the email requires the format of [\*\*@\*\*.com](mailto:**@**.com) and if the wrong input is given you will get a popup telling you what is required.

### Week-13

19th-25th April

This week we were finalizing the report that we have been adding to each week. We got some pointers from our coordinator on what we should add and what areas did we stray of topic.

We added work to future development, conclusion/retrospective as we learned of better approaches to our own after the fact and we got one of our friends to test out code and see what they thought. They gave us some good feedback on what features we should add in the future.