\n - Newline

Team 19

Deliverable 5

Sprint Backlog

Sprint 5

Duration: November 13 - November 19

The link below contains the sprint plan, execution, burndown chart and graph for sprint 5
https://docs.google.com/spreadsheets/d/1jQ8F4uXXfon3L4rK_rBWbQHDeP-phsp9FXXHslXHDFU/edit#gid=1924449463

Completed Tasks
16-F-1: Modify upload page to accommodate for both types of organizations
7-B-1: Create an endpoint for changing submission
6-B-1: Create an endpoint for uploading submission
13-B-1: Create an endpoint for editing a report template
8-B-1: Create endpoint for getting submission
12-B-1: Create an endpoint for storing and retrieving report templates
11-B-1: Create endpoint to get all created queries under an organization
11-B-2: Create endpoint to run queries
9-B-1: Create endpoint for deleting submission
10-B-1: Create a endpoint for submitting queries

Sprint 6

Duration: November 20 - November 26

The link below contains the sprint plan, execution, burndown chart and graph for sprint 6
https://docs.google.com/spreadsheets/d/1_fWW6vSPOVAMZour3fWSr3W7Ybu80duQ91Vg6D3R3wk/edit#gid=0

Completed Tasks
18-B-1: Create endpoint for getting all reports
5-B-1: Create Endpoint for adding/deleting table
5-F-1: Create option for user to add/delete table
15-B-1: Create endpoint for creating report
7-F-1: Create an interface for editing submitted data
13-F-1: Create an interface for editing a report
13-F-2: Edit metadata of templates
8-F-1: Create logic for submitted data
12-F-1: Create interface for creating and displaying a report template
11-F-1: Display list of queries created under organization
11-F-2: Create interface for displaying result from running a query
9-F-1: Create option to withdraw submission
10-F-1: Create a page for creating queries

Code Review

Code Review Strategy

Our guideline to do code review is to check if the code meets the following requirements.

- The code should have good readability and should be self explanatory without/very less comments.
- Every piece of code is written to meet the clients needs and should not contain any dead code.
- The design of every component is in such as way that it can be easily modified in the future.
- The code is implemented to be stable.
- The code does not contain any bloaters which means it should not have very long classes or very long methods.
- The code should not contain complex/multiple if statements or switch statements.
- The code should not contain duplicate code. The duplicate code should be replace by a method or a component (front-end).
- The code should not contain magic numbers. It should make use of constants instead.

Code Review Summary

Frederic Pun

Backend:

The backend, in my completely bias opinion, is created fairly efficiently. It has a lot of file bundling using index.js which allows for cleaner imports and automated route initializations. Due to our limited knowledge of backend technologies, there are many areas of improvement from architecture to use of technologies. The handlers of the routes could have more reusable components such as filter for different permission levels, which would further clean our code and reduce redundant code. As for technologies, mongoose would have allowed for more structured (as structured as our dynamic template creation system can be) models and easier implementation.

Frontend:

The front end, although has a very professional and polished front, is fairly messy behind the scenes. I think the reason for this is due to the scale and complexity of the pages being significantly more than expect. As a result, we have a mix of extremely big and complex components, and small abstracted components. A solution which can help reduce the size and complexity of the project would be the use of a state management system such as redux. Redux was not used initially to help reduce the complexity of the project for other members. However, considering the situation of the project, it would of been a good call.

Chaoyue Xi

Backend:

I think our design of backed code is good, considering both efficiency and readability. At the same time, it is convenient to modify or add new features. For unit test, I think, if we have more time, we should do more cases to cover as more situations as possible. The cases we have done are not enough. Besides, some specific methods in out backend code is not efficient enough, such as the data submission part. Each time I submit or modify the data, I need to wait quite long time because it need to go over the whole data. If we have more time, I believe we can design a more efficient algorithm. But for now, the algorithm we use is also good.

Qingtian Wang

Backend:

In my opinion, the amount of comment and the readability of code is at a good balance level. There could be more explanations for database operations but most of them are not hard to understand after gaining basic knowledge of JavaScript database APIs. I just refactored many text in the code to constants so hard-coding are not problems anymore.
However, unit testing cases can be more varied as some bugs cannot be found without some explicit test cases. We already found a bug that can should be uncovered by unit testing.

Frontend:

Some of the user interface can be improved with some friendly design. For example, I added a background image for login page such that the whole color seems mild, and an additional warning for incorrect login certificate will be useful since most messages are hidden in browser console.

Vishwaa Patel

Front-end:

In my opinion, the front end code is very complicated with lots of pages. However, the major components are well separated into different components such as lists, modals or forms for the layout of the different elements in the page. These components have been reused at many different places omitting duplicate code. Even though the component like the table has only been used once, it has been separated as a component to make the further development/ modification to the code base easier and faster. Moreover, the layout of the css files is made in a way that it is easily understandable without comments and easier to build on it.

Guidelines on building and running the product

Before both running, please ensure Node Js(LTS Version) is installed, and path of node js is added into system environment variable.

Back-end:

1. Download and install MongoDB(current release) from https://www.mongodb.com/download-center/community?jmp=docs
2. Switch working directory to Back-end
3. Run command: npm install
4. Run command: mongorestore --port 27017 -d test ./src/test/db
5. Build a local .env file with content:

PORT	3001
HASH	sha256
DB	mongodb://127.0.0.1:27017
COLLECTIONS	greencare

6. Run command: npm start

Front-end & Integrate:

1. Switch working directory to Front-end
2. Run command: npm install
3. Wait for installations to be finished
4. Build a local .env file with content:

REACT_APP_SERVER	http://localhost:3001

5. Run command: npm start

Guidelines on running the unit tests and integration tests

Before both test, please ensure Node Js(LTS Version) is installed, and path of node js is added into system environment variable.

Back-end:

7. Download and install MongoDB(current release) from https://www.mongodb.com/download-center/community?jmp=docs
8. Switch working directory to Back-end
9. Run command: npm install
10. Run command: mongorestore --port 27017 -d test ./src/test/db
11. Build a local .env file with content:

PORT	3001
HASH	sha256
DB	mongodb://127.0.0.1:27017
COLLECTIONS	test
REG_ORG	5be6fadc35e8e75238438ebd
SYS_ORG	5be5f396678020045400 44b1
SYS_ADMIN	UDCp31OmcSSyv/2D7bxp69F5A928IVH/RDdZ8owwpAk=
SYS_USER	x0rkxnniZGMjRzrfaIqLFREclATbgFvE1srO4aHd2Ks=
REG_ADMIN	S/i4rrJKPLCB6N3ly/cHpXnrtZ/reMpsvSEvZMGs56g=
REG_USER	Gqj1RoNU4/4WiLrOg5Ez/4zZWQsuP1ICDopv6uFTatk=

12. Run command: npm test
13. The unit tests will be run automatically

Front-end & Integrate:

6. Switch working directory to Front-end
7. Run command: npm install
8. Wait for installations to be finished
9. Build a local .env file with content:

REACT_APP_SERVER	http://localhost:3001

10. Run command: npm start
11. Test functionalities manually

For integration test, start backend and frontend simultaneously, then test frontend functionalities manually.

Acceptance Tests

These are the acceptance tests for our product.

The user starts with the login page.

Login Page

Successful Login

- The user enters the correct email and password.
- The input field border of the email and password will turn green when the email and password are of correct format.
- The user will click login button and will be logged in the system and redirected to the upload page.

Login Failure

- The user enters incorrect email and password.
- The input field border of the email and password will turn red when the email and password are not of correct format.
- The user will click login button but pop up a message of invalid email or password.

DashBoard

Dashboard contains links to different pages in the app such as Upload, Reports, Queries, Organizations, Users. The organization and queries page is only available for the TEQ users.

Upload Page

- The templates stored under the organization is displayed as a list.
- The submission for the template will appear in a table so that the users can edit it.

Uploading a file

- The user will select a template for which they want to upload a submission from the list.
- The user will click the upload button and a pop up will come up.
- The user will select the file that it want to upload either by dragging it or by browsing it in the computer.
- The user will click the submit button to upload the submission.
- If the user does not want to upload any submission, then the user can click the exit button and the pop up will disappear and it would not change anything.

Editing the submission

- The user will select a template for which they want to edit the submission from the list.

- The user will make the desired changes in the table for the data.
- The user will click the edit submission and the submission of the user will be updated.

Modifying the template

- The user will select the template for which they want to modify the template.
- The user will click the modify button and a pop up will come up where the user can modify the columns names and the requirement of the columns.
- The user will click submit to save changes.

Deleting the submission

- The user will select a template for which they want to upload a submission from the list.
- The user will click the delete button to delete the submission from the template.

Reports Page

Adding report

- The user will click the add report button and a pop up will come up.
- The user will enter the name of the report and click save to add the report to the reports list.
- The user can also click the delete button and the pop up will disappear and nothing will be added.
- The user will click add section and a pop up will come up.
- The user will add the the name of the section, select the query and select the type of chart and click save.
- The chart with the data from the query will be generated.
- The user can add multiple sections for a report.

Creating PDF of the report

- The user will select the report from the list of reports.
- The user will click the download button to download the report as a pdf.

Deleting the report

- The user will select the report from the list of reports.
- The user click the delete button and the report will be deleted from the list of reports

Modifying reports

- The user will select the report from the list of reports.
- The user will make modification to the report by adding/ deleting/modifying sections of charts into the report.
- The user will click the save button and the changes of the reports will be saved.

Queries Page

The queries stored under the organization is displayed as a list.

Adding a Query

- The TEQ user will click the add query button and a pop up comes up.
- The TEQ user will write the name of the query and the query and click the save button to save the query.
- The query will be add to the list of queries.
- If the user does not want to add any query, then the user can click the exit button and pop up will disappear and it would not change anything.

Changing a Query

- The TEQ user will click on the query from the list.
- The TEQ user will change the name of the query and/or the query and click save to save the changes for that query.

Deleting a Query

- The TEQ user will click on the query from the list.
- The TEQ user will click the delete button to delete the query. The query will be deleted from the list of queries.

Run Query

- The TEQ user will click on the query from the list.
- The TEQ user will click the run query button and the query will run and display the results below.

Organizations Page

- The organizations stored under TEQ is displayed as a list.
- A system organization is an organization like the TEQ who can access the data, generate queries.

Adding a normal Organization

- The TEQ user will click the add organization button and a pop up comes up.
- The TEQ user will write the name of the organization and set the system organization to false.
- The TEQ user will click the save button and the organization will be added to the organizations list.
- The TEQ user can click exit to not add an organization and the pop up will disappear without adding anything.

Adding a System Organization

- The TEQ user will click the add organization button and a pop up comes up.
- The TEQ user will write the name of the organization and set the system organization to true.

- The TEQ user will click the save button and the organization will be added to the organizations list.
- The TEQ user can click exit to not add an organization and the pop up will disappear without adding anything.

Deleting an organization

- The TEQ user will select the organization from the list.
- The TEQ user will click the delete button and it will delete the organization from the list of organizations.

Changing information of that organization

- The TEQ user will select the organization from the list.
- The TEQ user will change the name of the organization or change the organization type (System or Normal)
- The TEQ user will click the save button and the organization info is updated.

Adding Users for the organization

- The TEQ user will select the organization from the list.
- The TEQ user will click the add user button to add an user for that organization and a pop up will come up.
- The TEQ user will add all the information of the user it wants to add and click save button to add the user.
- The TEQ user can also click the exit button on the pop up and the pop will disappear without adding any users.

Users Page

Adding an administrator user

- The user can click the add user button and a pop up will come up
- The user will add the information of the user they want to add and click the save button to add the user to the users list for that organization.
- The user will set the Administrator to true.
- The user can also click the exit button and the pop up will disappear without adding anything.

Adding a normal user

- The user can click the add user button and a pop up will come up
- The user will add the information of the user they want to add and click the save button to add the user to the users list for that organization.
- The user will set the Administrator to false.
- The user can also click the exit button and the pop up will disappear without adding anything.

Deleting a user

- The user will select the user from the list
- The user will click the delete button to delete the user from the list

Modifying information of a user

- The user will select the user from the list
- The user will change the name,email or password of the user and click save to modify the user's information.

Logout

- The user clicks the logout button and is logged out of the system and is redirected to the login page.

Product Presentation Video

https://drive.google.com/file/d/1sBw1d7CM0Qvgjh8LQJFE5i9i3uU6pbBx/view?usp=sharing