A summary of your decisions and the options you considered for your component(s)

We figured that text answers would best be displayed in a Bootstrap accordion format because the design pattern seemed aesthetically pleasing – it's elegant and fairly minimalistic with smooth animations built in – and because the Accordion's collapsibility lets users scroll past answers more easily than just displaying them in a paragraph form. It's also simple to implement and extend – because it's a native Bootstrap component. Finally, we were planning to migrate Simulence's styling to Bootstrap anyway – and building new UI using Bootstrap lets us work towards that.

The accordion is split into two subcomponents – the outer Accordion container, and individual Accordion Items. This was done so that we can fill an accordion with items programmatically by just sending an array of answers to the outer accordion, taking advantage of Vue's reusability.

It made sense to display every other type of answer (Multiple Choice, Multi-Select, Linear Scale, and Dropdown questions) with Bar Graphs as it was a very intuitive and straightforward way to gather feedback on the quality of the game. The x-axis is each option in the question, and the y-axis indicates the number of people who selected that option.

We decided to set up Component Testing (for the purpose of testing UI component interactions), because we expect our components to be **interactive** (this is standard for good web applications), and **reusable** (because we want to take advantage of Vue's Component Reusability) to keep our code extensible. If we plan to re-use the same component over and over again, it makes sense to test that it responds to user input events correctly. We eschewed logical unit tests for this deliverable because our front-end UI doesn't really have much complicated logic to test, and so testing logic is not as critical as testing UI interactions. However, Jest, the testing library we set up to get component testing to work, supports unit testing out-of-the-box – although we haven't written any unit tests yet, the infrastructure to run and view them is already set up.

- **Individual contributions explaining who did what. You can keep it to at most one paragraph per person to highlight any work that is not captured in any of the repos.**

Michael Y. designed and implemented the components related to the Text Accordion, researched and set up testing infrastructure enabling unit and component tests for front end code, and wrote the first two component tests for the Accordion component. A lot of research was done to figure out why previous teams couldn't get tests working and overcome that issue.

Nealon S. designed and implemented the Bar Graph component, wrote unit tests for it, and designed the layout of the survey results page, deciding how to display each type of answer, as well as how the data for the results would be called in memory. Nealon also deployed the application to Firebase, and pulled multiple all-nighters working on the Text Accordion and unit tests, but the outdated codebase made this very difficult.

Both worked together to realize that the codebase was out of date, and come up with solutions to still implement code without breaking anything.

All the details and instructions needed for your TA to see and verify your work. You need to provide enough documentation so your TA can confirm:

1. Go on the website https://simulence-d2-2023-8c85a.web.app/#/login.
2. Log in with the developer email: devSimulence@gmail.com, and with the password 123123.
3. You'll be on the Projects page of the Developer. If there isn't a project, select Add New Project, fill in the data with anything you want, and Save.
4. Go back to the Dashboard and select view details on the Project you just created.
5. Go to the Analysis option on the side-bar, and select Add New Survey under Surveys.
6. Follow the existing steps to create a survey of your choice.
7. Go back to Analysis and press Publish on the survey you just created, then click Results.
8. You should be able to view sample questions and answers for the survey you just created, where:
    -Analytics for multiple choice, multi-select, Dropdown, and Linear Scale questions are shown through Bar Graphs
    -Responses to short-answer questions are given through a Text Accordion where each button is a playtester working on the game, clicking will reveal sample responses.