**Process**

**Good:**
- Effective communication channel established with the client.
- Regular team and client meetings
- Maintaining a Trello board was a good process, since it allowed us to keep track of task-wise progress.
- Adequate frequency of team meetings and meetings with the client.
- Everyone in the team attends every meeting on time. Although virtual, people are very cooperative and use tools like Slack, GitHub, Trello very efficiently.
- Division of documentation work went really well to produce high quality and understandable documents.
- Number of meetings to discuss development and progress/allocate tasks

**Bad:**
- Need to verify implementation feasibility before committing to the client.
- Creating branches on GitHub was an unsettling process since the work got very convoluted while merging and finalizing.
- We spend more time in discussions, instead of taking actions.
- The peer review process takes place long after the feature has been completed, which leads to messy code in turn leading to some performance issues which need to be addressed earlier.
- Communication via Slack for urgent discussions with the client delayed the decisions.
- Need to have everyone review documentation/diagram vs. only a group of people doing it to prevent errors or inconsistencies.

**Try:**
- Restrict the addition of more features by the client. For example, the client recently added a feature to enable the user to bookmark a product.
- Have a time limit to the team meetings
  - Over the next 4 weeks, we can try having team meetings of predefined duration. In this way, each team member will have an agenda to discuss during the meeting and will be well prepared with it. This will save the time spent in discussions and we can utilize this time effectively.
- Daily stand-up would be a good thing to try.
- Decide agenda of the meeting, pre-assign time to each point on the agenda to avoid driving away from the main point during discussion.
- We are putting a process into habit that as and when the development has been completed, we can have peer review sessions, unit testing and integration testing done for that particular feature.
- Use separate slack channels for frontend and backend development.
- Hold meetings specifically to discuss team documentation and reach a consensus.
People

Good:
- Great collaboration on development of core capabilities.
- All team members are learning and contributing proactively.
- Equal distribution of work according to each person’s strengths to get maximum output.
- Nice coordination of all team members with each other. Everyone is willing to learn and contribute.
- Client has been very supportive throughout until now in providing information, credentials for access, prototypes which helped us allocate the tasks faster and get it completed faster. All the team members are learning the new technology.
- Asking for help from each other proactively whenever anyone gets stuck.
- Everyone has worked hard to make the progress we have in terms of development.
- Everyone is committed to the project and constantly works on it despite having other classes.

Bad:
- All team members must be aware of what other team members are working on and should give status updates on their progress to the entire team.
- Delay in receiving food product data from the client.
- Conflicting schedules made it difficult to organize meetings.
- Everyone is new to the core technology stack being used.
- As people are new to technology and have other commitments, sometimes the assigned task takes a much longer time than estimated, which delays some features.
- Not paying attention to minute details while working on team assignments and presentations, avoiding or not taking charge of their own actions and tasks assigned.
- Some client delays in making decisions have stalled development, especially regarding the food API.

Try:
- Keep cameras on showing more enthusiasm for team meetings.
- Upfront communication with the client about receiving food product data and other feedback.
- Better communication between people working on the backend and the frontend.
- Assign tasks to people according to priority of tasks as well as taking into consideration the schedule constraints of all team members to avoid hassle as the deadline approaches.
- Short knowledge sharing sessions between team members so if any emergency arises development work would not hamper to a greater extent.
- Try integrating more testing into remaining sprints to ensure full functionality.
We can have 2 people who know the technology better, assigned for helping other team members, so that if a person is stuck with a feature, the people who know the technology can guide them through and pace up the process. This can be conducted twice a week in which doubts are resolved.

**Product**

Good:
- Great implementation progress made on the core capabilities.
- Submit a rating feature in the application was appreciated by the client.
- Database integration properly achieved, most frontend screens ready.
- The project is scoped down to be able to implement in one semester. The product developed until now is working properly. Client is satisfied with the demo.
- Login/Signup functionality was implemented well. The technology selected, namely React Native and Firebase, was a good decision for authorization and authentication, which forms a very important part of the application. All of the functionalities were appreciated by the client demonstrated in the ARB presentation.
- The system/app will be able to satisfy most of the features we want to develop.

Bad:
- Many parts of the developed code needs cleanup as there is a lot of commented code and some files which we are not using anymore which creates a lot of confusion.
- Food API could not be integrated with the mobile application.
- Finalizing the API was a big issue and a roadblock to success. We have not yet decided which food API to use and hence, need to proceed with a sample database provided by the client.

Try:
- We should use just one GitHub branch for keeping accurate track of progress made in development and push changes as often as possible to facilitate testing and faster implementation.
- Thorough review of the documentation to be submitted as a part of team assignments.
- Have a start to finish run-through or bug-bashing session and note every small bug so that we can address it.
- Some corner cases of implementation are not clearly discussed. So, drill down to the depth of each use case implementation to have a definite idea of what exactly is the expected end product of this use case.
- To make the process smooth after we get the API we are trying to replicate the data received from the API from the research done and client feedback, so that when we use the API we would not face much difficulty in terms of development.
- Use the USDA food API to extract nutritional information of food products
- Try to refine the planned features as much as possible while striving to maintain scalability so that the app can be improved by future teams.
Final Action Items:

- Create specific agendas for meetings and set meeting time limits (soft limits).
- Assign tasks to people according to priority of tasks as well as taking into consideration the schedule constraints of all team members to avoid hassle as the deadline approaches.
- Peer review and test features (with specific written test cases) within the same sprint that they are developed/finished, also to keep track of bugs/defects in order to address any issues and thoroughly debug the software.