**Sprint 2 Report**

**SlugFit**

**2/14/23**

**Actions to Stop Doing**

In Sprint 2, we were all able to work very efficiently/productively individually and as a

Team, however at the very end we weren’t able to merge all the PR since we had rushed the ending. We need to more efficiently communicate the changes we make such that merging can be more fluent. Our system of using Notion to track and divide the work of user stories/tasks kept us organized, but we could make sure to have more coordination in the planning of components. Also, dividing into groups allows members to have partners to assist them.

**Actions to Start Doing**

1. Documenting Functions

Any functions or large blocks of code must have comments to explain what that code does. This will help other members easily & quickly understand what the code does and how it works.

1. Assign People to Review PRs

Members who are more familiar with the technologies used will be tagged when a PR is ready to be reviewed. When PRs are reviewed more frequently, it will make the overall process more efficient as it allows members to correct errors or begin working on different tasks.

1. Getting PRs Out Sooner

Members should submit their PRs sooner to get reviewed, even if they’re not ready to

merge. This will allow reviewers to correct any logic flaws early on before the developer

has spent too much time in the wrong direction. Also helps if other components rely on the props of the feature.

1. Ensure that Pull Request are able to merge to main branch without error

With lots of code being written there tends to always be issues with new code overwriting old or not being able to merge smoothly. That should always be taken account of and local branches should always be working on the main branch as it gets updated.

**Actions to Keep Doing**

1. Assigning Partners

Since half of our team is unfamiliar with the technology that we are using, we had decided to create pairs with those who had experience and those who haven’t. This allowed for those who haven’t experience to learn as their partner was working on their task as they were able to follow along and ask questions.

1. Reach out if you have questions/problems

When a member is confused about something and had spent over 10-15 minutes trying

to solve the problem on their own, we always had someone available to help. Rather

than being stuck on a problem for too long, members were able to assist and teach

others, allowing us to be more efficient.

3. Specific Criteria/Format on PR

When a member is ready to do a PR on GitHub, he/she must make sure that their code is free of compiler errors/warnings. Once there are no more errors/warnings, the PR must follow a specific format providing useful information such as feature name, user story, etc. This will allow all PRs to be organized and easily understandable by any other member.

**Work Completed:** 19 User Stories, 53 Story Points

* Refactoring database calls, types, and naming (5)
* Make BlockActions (Delete, Duplicate, Rename, Insert Below or Above) (5)
* display loading wheel while fetching(2)
* Lock UI on load(1)
* Add green dots or checkmarks to the days of the calendar where the user completed a workout(3)
* Create a calendar view that displays the monthly calendar (they can go backwards in time)(4)
* Fetch the user’s past workouts and render them in the list (3)
* Create a PastWorkoutBlock that displays the date of the completed workout, the name, and how many of the working sets were completed (3)
* Create a horizontal scrolling flatlist of PastWorkoutBlock components(3)
* When the blocks are clicked, set the selectedConsumableWorkout to the one the user clicks then navigate to the ConsumeWorkoutPage(2)
* Create a SelectWorkoutPage that displays the users workouts in blocks(3)
* Create a ‘Start Workout’ button on the home page that navigates to the SelectWorkoutPage (1)
* Display the different blocks belonging to an exercise in order(2)
* Create a ConsumableNoteBlock component that shows notes for an exercise(3)
* Create a ConsumableRestBlock component that shows the rest time(3)
* As a gym rat, I want to specify the rest interval I should take after performing a set of an exercise so I can be adequately recovered for the next set (2)
* As a gym rat, I want to be able to add notes to my exercises that give me cues on form so that I can avoid injury (3)
* As a gym rat, I want to specify whether a set is a warm up or working set so I know to adjust my intensity (2)
* Distinguish between Warmup and Working sets(3)

**Not Completed:** 6 User Stories, 13 Story Points

* Create an EndWorkout button that sets the status of the workout to DONE in supabase and saves any last data the user entered and navigates away(3)
* When you click on an exercise in the ListView, have it open the CardView and then navigate to that exercise in the flatlist(1)
* Save and update the model in supabase when a user enters in their data → cascade changes back down to UI with a real-time listener(5)
* Create a ConsumableWorkout interface and table in Supabase(3)
* Create a ListView toggle button to switch between cards and one scrolling list to view the exercises(3)
* Create Next and Previous buttons to navigate the exercises list(1)

**Work Completion Rate**

Total # of user stories completed during Sprint 2: 19 User Stories.

Total # of estimated ideal work hours completed during Sprint 2: 53 Work Hours.

Total # of days during Sprint 2: 14 Days (2/1-2/14).

User stories/day and ideal work hours/day figures: 0.3-0.5 Stories per Day

