Table 1: Revision History

Date	Developer(s)	Change
Date1	Name(s)	Description of changes
Date2	Name(s)	Description of changes
...	...	...

Development Plan
REVITALIZE

Team 13, REVITALIZE
Bill Nguyen and nguyew3
Syed Bokhari and bokhars
Hasan Kibria and kibriah
Youssef Dahab and dahaby
Logan Brown and brownl33
Mahmoud Anklis and anklism

[Put your introductory blurb here. —SS]

1 Team Meeting Plan

2 Team Communication Plan

3 Team Member Roles

4 Workflow Plan

The main repository is named REVITALIZE. The implementation will follow the feature-branch model. Feature development will take place in a branch other than the main branch. This encapsulation makes it easy for our team to work on a feature without disturbing the main codebase. That way, the main branch will never contain broken code. The feature-branch model will make it simple for our team to initiate discussions around a branch by making pull requests to comment on each other's work.

The documentation will follow the centralized model. The main branch will serve as a point of entry for all changes to the documentation. All changes will be committed to this branch.

To report bugs and request modifications to implementation or documentation, we will create GitHub issues and assign them to the people working on a specific problem. This will also serve as a way of holding team members accountable for the issues they have to tackle in the future. Labels will be used to classify

issues and pull requests to help create a standard workflow in our repository.

If required, milestones will be created to track progress on groups of issues or pull requests in our repository to better manage our project through viewing milestones' view dates, completion percentages, and lists of open and closed issues and pull requests associated with the milestones. Using milestones could become a necessity as our project size and complexity increases with time.

5 Proof of Concept Demonstration Plan

What is the main risk, or risks, for the success of your project? What will you demonstrate during your proof of concept demonstration to convince yourself that you will be able to overcome this risk?

5.1 Potential Risks and Difficulties

5.1.1 Finding Primary Stakeholders

Currently, our team's primary stakeholders are adults who want to improve and keep track of their overall health and wellness via an easy to use, all in one application. However finding such people who will be willing to, participate in the evaluation of our project, and act as consultants who will meet with us and review our work could prove to be difficult.

5.1.2 Human Computer Interface and User Experience

As there are already existing implementations that are similar to our project, our team has to determine how to deliver this project it's stakeholders in a better way than the other existing applications. This entails improving the human computer interface and user experience aspects of our project. Determining how to tackle this issue entails answering many of the deepest questions and most important questions in computing such as can we design our project to achieve the optimal functionality, learnability, performance, error rates, trustability, retention, and accessibility all at the same time? What does it mean for our interface to be better at all? Easier to use, faster to get tasks done, or less mistakes made by user? Not answering such questions with accuracy can potentially be a blocker to how we plan to deliver our project to our stakeholders.

5.1.3 Design and Implementation

Some of our team members are not very familiar with the client-server architecture model or with the programming languages and API's that our team plans to use. There will be a learning curve that our team members are required to get over in a decent amount of time for us to finish the project on time.

5.1.4 Testing

The tests that our team requires to have a working deliverable entails testing for functionality, usability, interface, performance, and user experience. However, our main concern regarding testing our project is determining how to come up with an effective test plan that will test all the different aspects of our project.

5.1.5 Scope

Our team does not currently envision the scope to be a risk as the project seems doable as it is. However, further down the line, as we are designing, documenting, implementing and testing the project, we will be able to determine if the project size is too large and if have to modify our goals. Thus, our team views project scope as a potential future risk.

5.1.6 Future Goal Changes

The project seems reasonable the way we planned it out. However, from previous years of experience with software development, our team knows that future issues will arise, that we currently have missed or didn't think or know of, that might force us to change our goals or intended plan of design and implementation.

5.1.7 Libraries

Our team does not anticipate any concerns with library installations as we will be working mainly with API's.

5.1.8 Hardware

There are no hardware risks or concerns as this is purely a software project.

5.2 Plan to Overcome Risks

6 Technology

- Specific programming language

- Specific linter tool (if appropriate)

- Specific unit testing framework

- Investigation of code coverage measuring tools

- Specific plans for Continuous Integration (CI), or an explanation that CI is not being done

- Specific performance measuring tools (like Valgrind), if appropriate

- Libraries you will likely be using?

- Tools you will likely be using?

7 Coding Standard

8 Project Scheduling

[How will the project be scheduled? —SS]