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 to work on a feature without disturbing the main codebase. That way, the main branch will never contain broken code. The feature-branch model makes it simple for team members 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, GitHub issues will be created and assigned to 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 the used repositories.

If required, milestones will be created to track progress on groups of issues or pull requests to better manage the 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 project size and complexity increase with time.

5 Proof of Concept Demonstration Plan

5.1 Potential Risks and Difficulties

5.1.1 Human Computer Interface and User Experience

As there are already existing implementations that are similar to this project, the team has to determine how to deliver this project to it's stakeholders in a better way than the other existing applications. This entails improving the human computer interface and user experience aspects of the project. Determining how to tackle this issue entails answering many of the deepest questions and most important questions in computing such as how to design the project to achieve the optimal functionality, learnability, performance, error rates, trustability, retention, and accessibility all at the same time? What does it mean for the 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 the team plans to deliver the project to it's stakeholders.

5.1.2 Design and Implementation

Some of the team members are not very familiar with the client-server architecture model or with the programming languages and application programming interfaces that the team plans to use. There will be a learning curve that team members are required to get over quickly to deliver the project on time.

5.1.3 Testing

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

5.1.4 Scope

The 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 team members are designing, documenting, implementing and testing the project, they will be

able to determine if the project size is too large and if project goals have to be modified. Thus, project scope is potentially a future risk.

5.1.5 Future Goal Changes

The project seems reasonable the way it is planned out. However, from previous years of experience with software development, team members knows that future issues will arise, that currently have been missed or were not thought of or known of, that might force the team to change it's goals or original plan of design and implementation.

5.1.6 Libraries

The team does not anticipate any concerns with library installations as standardized frameworks will be used.

5.1.7 Hardware

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

5.2 Plan to Overcome Risks

5.2.1 Human Computer Interface and User Experience

Discussing with the stakeholders and devising user-acceptance testing should be the path to overcoming the interface-related risk. Also, the team aims to put one developer in charge of specifically researching methods of improving the app's user experience.

5.2.2 Design and Implementation

Frequent meeting and communication should be enough to ensure adequate transfer of the software-design-related knowledge and expertise some team members have more of than others.

5.2.3 Testing

The team aims to go beyond just unit testing and high code coverage statistics; end-to-end testing and user acceptance testing should also be considered and implemented to ensure smooth usability. As the project progresses, the team aims to dedicate special attention and time to devising a clear testing architecture.

5.2.4 Scope

It will be tried to minimize this risk by intelligently dividing tasks and providing realistic estimates for how long each task will take. If the team realizes some

things to be out of scope later down the line, then they can meet with stakeholders to narrow down on what should be prioritized and what should be left as a stretch goal.

5.2.5 Future Goal Changes

Trying to stay flexible and meet twice a week. If some goals change, then devising a plan with the time and effort scope at a later date should be a viable route. .

6 Technology

6.1 Specific Programming Language

For this mobile app, Javascript will be used as the main programming language, as it is aimed to use React Native and Express JS (two Javascript based frameworks) to build the app's front end and back end respectively.

6.2 Unit Testing Framework

The well resourced, well documented Javascript unit testing framework the development team has collectively expressed favour for is Jest, hence this will be the framework of choice.

6.3 Continuous Integration

As the development team is generally comfortable with Github and that is the version control platform that has been opted for, it makes sense to use Githiub's own Continuous Integration features to support real time integration of incoming code.

6.4 Performace Measuring Tool

After conducting research, it has been decided that Sentry will be the tool of choice due to the vast functionality it offers in regards to performance measuring and debugging.

6.5 Libraries

This would a dynamic, extensible list so it is better describred generally like this: React, Express, Node js all associated libraries with said frameworks.

6.6 Tools

Recipe Finder (https://www.edamam.com/), Workout Tracker(https://wger.de/en/software/api), Sleep Tracker (https://developers.google.com/location-context/sleep), IDE's: VS Code and Android Studio.

7 Coding Standard

The team can use the Google JS Style Guide as a benchmark for coding standard and stylistic specs: https://google.github.io/styleguide/jsguide.html

8 Project Scheduling

Gantt chart will be used for project scheduling, showing key deliverables and progress. Will be updated regularly throughout the project: https://github.com/BillNguyen1999/REVITALIZE/tree/main/projectschedule/REVITALIZE.pdf.