**CS673 Software Engineering**

**Team 4 - Project Name**

**Project Proposal and Planning**

|  |  |  |  |
| --- | --- | --- | --- |
| Team Member | Role(s) | Signature | Date |
| Ray Clark | Team Lead | *Ray Clark* | May 31, 2021 |
| Danny | Backup Team Lead | *Danny G* | May 13, 2021 |
|  | Backend Developer |  | May 13, 2021 |
| Chen Zhao | Requirement Lead | *Chen Zhao* | May 15, 2021 |
| Huimin Diao | Configuration Lead | *Huimin Diao* | May 15, 2021 |
| Josephine But | QA Leader | *Josephine But* | May 24, 2021 |
| Zixia Zhou | Security Leader | *Zixia Zhou* | May 17, 2021 |
|  |  |  |  |
|  |  |  |  |

**Revision history**

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Author** | **Date** | **Change** |
| Iteration 0 | Group | May 18, 2021 |  |
| Iteration 1 | Group | June 1, 2021 |  |
|  |  |  |  |
|  |  |  |  |

[Overview](#_87t9hln2vjz0)

[Related Work](#_mps353x5ezyl)

[Detailed Description](#_fg3z0hpd4q9v)

[Management Plan](#_ds8oyr75pnh1)

[Process Model](https://docs.google.com/document/d/107bVcXdAG-ogRr90PquFB8-aWGvTwSua8pu_O4Kmz6c/edit#heading=h.27177f40uci)

[Risk Management](https://docs.google.com/document/d/107bVcXdAG-ogRr90PquFB8-aWGvTwSua8pu_O4Kmz6c/edit#heading=h.a4oqwntk3mw)

[Monitoring and Controlling Mechanism](https://docs.google.com/document/d/107bVcXdAG-ogRr90PquFB8-aWGvTwSua8pu_O4Kmz6c/edit#heading=h.ywdoc2clc9yt)

[Schedule and deadline](#_tadq5mb0pici)

[Quality Assurance Plan](https://docs.google.com/document/d/107bVcXdAG-ogRr90PquFB8-aWGvTwSua8pu_O4Kmz6c/edit#heading=h.72e1f4uawy2r)

[Metrics](#_b2haznn3yyz2)

[Standard](https://docs.google.com/document/d/107bVcXdAG-ogRr90PquFB8-aWGvTwSua8pu_O4Kmz6c/edit#heading=h.vc72k6dweldv)

[Inspection/Review Process](#_f1c69ifi68h7)

[Testing](https://docs.google.com/document/d/107bVcXdAG-ogRr90PquFB8-aWGvTwSua8pu_O4Kmz6c/edit#heading=h.r5d5mhtlf0kq)

[Defect Management](https://docs.google.com/document/d/107bVcXdAG-ogRr90PquFB8-aWGvTwSua8pu_O4Kmz6c/edit#heading=h.54a4wuncjg1c)

[Process improvement process](#_jhct37ebxxpn)

[Configuration Management Plan](https://docs.google.com/document/d/107bVcXdAG-ogRr90PquFB8-aWGvTwSua8pu_O4Kmz6c/edit#heading=h.hw41vg4ykxen)

[Configuration items and tools](https://docs.google.com/document/d/107bVcXdAG-ogRr90PquFB8-aWGvTwSua8pu_O4Kmz6c/edit#heading=h.bwlb4d4vdox2)

[code commit guidelines](#_yyauft6zr9hw)

[References](https://docs.google.com/document/d/107bVcXdAG-ogRr90PquFB8-aWGvTwSua8pu_O4Kmz6c/edit#heading=h.8mva2050iy7t)

[Glossary](#_ty3i2nqffhtc)

# Overview

Being overweight is becoming more and more common in human society, and being overweight has a systemic and complex negative impact on a person’s health. Since diet control (calorie intake control) is one of the most important ways to fight obesity, our group will build an app that helps people calculate the calories of the food they eat and will provide tips, encouragement and warnings to help people fight obesity. Any person hoping to regulate their caloric intake is a potential user. The name of the app is **MyDietHub**.

# Related Work

Here are some websites that are similar to our project:

- <https://www.myfitnesspal.com/food/search>

* In myfitnesspal website, users can search a food’s calories, protein carbs and other nutritional info. Compared with our project, this website also assigns a grade for each food. In our meeting, we discussed this feature as well. We hope to also add a grading system, but it will be added only when all other essential features have been completed.
* - <https://loseitblog.com/>
* Same as loseit website, it adds a part called “Loss weight Tips”, this is also a feature that we would like to integrate in our application. We plan to add a recommendation feature for daily food and calories limitation.

# Proposed High level Requirements

* 1. Functional Requirements  
     1. Essential Features (the core features that you definitely need to finish):

(For each essential features, please give a rough estimation in terms of person hours or an range of person hours)

1. [Register]

As a user, I want to be able to register for an account so that I can have my own account.

1. [Log in]

As a user, I want to be able to log in to the app so that I can operate on my own account.

1. [Log out]

As a user, I want to be able to log out the app so that my account will be more secure.

1. [User Data]

As a user, I want to input my basic information (gender, height, weight), so that the application will calculate the reasonable amount of calories that I consume per day.

1. [Query calories]

As a user, I want to be able to query the calories of a given food item, given the name and quantity of the food item, so that I can track my caloric intake.

* + 1. Desirable Features (the nice features that you really want to have too):

1. [Estimation]

As a user, I want the application to help me determine the approximate weight of food by simple description (small, medium or large) when I do not know the approximate weight of food, so that I can get a rough calorific value without knowing the exact weight

1. [Rating]

As a user, I want the app to rate my diet so that I can understand the quality of the food I am eating.

* + 1. Optional Features (additional cool features that you want to have if there is time):

1. [Blog]

As a user, I want to share my experience or some other comments on the application, so that the application should integrate a feature similar to a blog.

1. [Chat]

As a user, I want to chat with others, so that the application should integrate an instant messaging feature.

1. [Calorie Tracker]

As a user, I want to have Calorie tracking over time, so that the application should store meals in a database and make a graph of caloric intake by day.

* 1. Nonfunctional Requirements
     1. Definition: A non-functional requirement is a specification that describes the system’s operation capabilities and constraints that enhance its functionality. These may be speed, security, reliability, etc.
        1. [Application Speed]
* When a user logins in, the page load time will not exceed 1 second.
* When the ‘Calculate Calories’ button is clicked, the total time the system takes to access the USDA database and return the data to the user will not exceed 5 seconds.
  + - 1. [Application Styling]
* The application will feature a graphics-based user interface for all screens.
* The application will utilize tabular data tables and bar and pie chart graphs to visually represent calorie data to the user.
  + - 1. [Security]
* The application will use a cryptographic hash function to hash the username and password to ensure the security of these user inputs.
  + - 1. [Reliability]
* The system will have a minimum 85% reliability for any given month meaning it will be up and operating at least 85% of the time.
  + - 1. [Maintainability]
* The system will have a 75 percent maintainability factor for any given 24 hour period meaning that there is a 75% probability that any system error will be addressed and corrected in a 24 hour time frame.
  + - 1. [Portability]
* The application will operate on Windows 10x and higher, macOS 11x and higher and iOS 14x and higher.

# Management Plan

## Process Model

(Please describe your software process model)

*Agile (fill in details)*

Because of the nature of the development team, the very short development timeline (7 weeks), the need for material revisions and enhancements to the application, and the distributed nature of the development group, we are using an Agile process model.

Agile - This dynamic development method utilizes small, self-organizing teams that engage in an iterative production process featuring regular communication, collaboration and feedback with regard to evolving designs and requirements. The hallmark of the method is a series of short development cycles called ‘sprints’ that focus on continuous improvement of the product. The method emphasizes individuals versus process, and places a high priority on working software and responsiveness to change versus a static method that uses extensive upfront documentation (Manifesto for Agile, 2001). For projects requiring speed and flexibility, the Agile method can be advantageous in that it provides for more rapid development and a high degree of adaptability given changing circumstances and requirements.

## Objectives and Priorities

The overarching theme of the objectives and priorities are that the core functionality of the app should take precedence over other features. With the saying of “something is better than nothing”, it’s important for us to focus our team effort on pushing the primary objectives before devoting time to other objectives. Within this framework, our early team focus will be on implementing the core features whereas later in the development cycle we can fill in missing documentation and non-blocking bugs. Test-writing can occur in parallel with code writing, but development of code should come first so we can achieve our MVP.

**Objectives (in order of importance)**

* + 1. Complete the essential feature of looking up and returning nutritional information for foods in USDA database
    2. User login portal for personalized tracking with BMI and target daily calories
    3. Estimation & extrapolation of food weight for calorie calculations
    4. Resolving bugs having to do with any of these 3 essential features
    5. Documentation, test writing & tracking of essential features and bugs
    6. Optional features (Graphs, etc)
    7. Optional feature bugs (ui/stylistic bugs). Primary focus should be on creating a MVP (minimum viable product) to start with

## Risk Management (need to be updated constantly)

## **Identified Risks for the Application**

1. Inadequate Number of Experienced Programmers – The existing development team has only one (1) programmer experienced with Flask, but all members are experienced with Python. Front-end interface will be limited to HTML/CSS & JavaScript, as no member is familiar with REACT. The need to learn Flask may cause time delays.

2. Distributed Development Team – Given the development team will be distributed throughout the world (U.S. west & east coast, and China) this may lead to poor communication and collaboration among team members due to time zone, cultural and language differences. There is a risk that communication issues will lead to delays and possibly a missed release date.

3. Aggressively Short Development Timeline – Because the app will need to be developed in just 7 weeks, base requirements will need to be fleshed out quickly and development will need to begin immediately for the timeline to be met. If the time delays are encountered, there is a risk of missing the release date.

**Risk Classification**

There are two types of risk classifications including organizational risks, which are described as those risks that are the result of people’s actions and the way in which they are organized, and technical risks, which are described as those risks resulting from specific hardware or software problems. In this case, the lack of experience with Flask among most team members would be classified as an organizational risk because the organization, team4, maintains this staffing deficiency. It is, therefore, the organization’s responsibility to ensure that all personnel on the development team learn and become proficient with this technology or assign this responsibility to specific team members who have the requisite capability.

**Risk Mitigation Approach**

There are two approaches to dealing with an identified risk. First, the team can employ a “conquest” strategy to the risk by facing it head-on. Alternatively, the team can elect to employ an “avoidance” strategy to the risk by seeking an alternative pathway or course of action.

**Limited Technology Experience** - For this Calorie Tracker App, we will address the lack of Flask experience through a conquest approach and assign the Flask code to two members who will either possess the skill in this technology or who agree to commit to learning it quickly.

**Distributed Development Team** – To limit the effects from the distributed development team, we will again use a conquest approach. We will establish three (3) pairs of programmers based on similar time zones to ensure adequate communication and collaboration among the team members.

**Aggressively Short Development Timeline** – To address the very short development timeline, we will ensure that we conduct at least two (2) stand-up meetings per week and communicate regularly on Slack to ensure that all development tasks stay on schedule.

Risk Management Sheet Link:

## 

## 

## 

## Monitoring and Controlling Tools and Mechanisms

We will use the following tools to facilitate group communication and monitor the project progress.

* + 1. Pivotaltracker Link:

Link : <https://www.pivotaltracker.com/n/projects/2498532>

A tool to keep track the user stories and the progress of the project.

* + 1. Slack Link: <https://app.slack.com/client/T02084K4P1B/C021UMH0E0Z/thread/C021UMH0E0Z-1621053305.113300>
* A channel for the team to communicate for different matters.
  + 1. Github Link:

<https://github.com/BUMETCS673/BUMETCS673OLSum21P4>

* All the team documentations, coding and all the files will submit to the github, except the presentation video.
  + 1. Zoom meeting

<https://us02web.zoom.us/j/89128285008?pwd=STRJUHBGYmtlTXZIWEZLbDdKSTRRZz09>

Create a new one before each meeting.

|  |
| --- |
|  |

* + 1. Weekly meeting time: <https://us02web.zoom.us/j/87594736737?pwd=MXBXZ2FKSzcyMmhrQWF6RUdKQmVqUT09>
    2. Google Drive:

1. This is a Document folder for the team for different iterations. All the team documentations will be stored in the subfolder of this folder.

<https://drive.google.com/drive/folders/1ycim_9muV95n0ubDR_IbJ9IAe5ShTqzZ>

1. Our team folder, any other shared documents or files will be stored here.

<https://drive.google.com/drive/folders/18SUpRb7TpjAeA7BixyKUDga1WlKJcZaI>

|  |
| --- |
|  |

## Timeline (need to be updated at the end of each iteration)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Iteration | Functional Requirements(E/D/O) | Tasks | Estimated/real person hours | Presentation Recording Link (5-10 minutes) |
| 0 | N/A | Define project requirements, team structure, issue tracking, env. setup | 15 | <https://drive.google.com/file/d/1xe5razhSfA8gTC-PqZQSviyHPXg0uMYg/view?usp=sharing> |
| 1 | Iteration 1 | Define project requirements, Software Architecture, Class Diagram, UI Layout, Registration, Login, API Development, Security Design, Database Design. |  | <https://drive.google.com/file/d/1-8lrzUlA9Bw0UK7WDwmKyrKusfcPsQNK/view?usp=sharing> |
| 3 | [Register] | Setting up the project framework | 20+ |  |
| 4 | [Log in]  [Log out] | API and database | 20+ |  |
| 5 | [User data]  [Calorie query]  Optional:  [Estimation]  [Rating]  [Chat]  [Blog] | user interaction | 30+ |  |

# 

# Quality Assurance Plan

## Metrics

* + 1. Product Metrics : In order to improve the quality software of our product, we use the following product metrics to measure and monitor our product quality.
* *Error Discovery Rate ( To determine the effectiveness of the test cases).*

Error Discovery Rate = (Total no. of defects found / Total no. of test cases executed) \* 100 %

* *Defect Fix Rate ( To know the quality of a build in terms of defect fixing).*

Defect Fix Rate = ( Defects fixed / Defects Reported) \* 100%

* *Defect Density ( To determine the stability of the application).*

Defect Density = Defect count / Line of Code

* + 1. Process Metrics : These metrics that pertain to process quality. They are used to measure the efficiency and effectiveness of various processes.
* *Test Case Preparation Productivity ( To calculate the number of test cases prepared and the effort spent for the preparation of the test cases).*

Test Case Preparation Productivity = No. of test cases prepared / labour of hours for test case preparation.

* *Test Case Execution Productivity ( To calculate the number of test cases executed and the effort spent for the execution of the test cases).*

Test Case Execution Productivity = No. of test cases executed / Total labour of hour to spend to execute the test case.

* *Test Execution Coverage ( To measure the amount of testing performed by a set of tests).*

Test Execution Coverage = ( Total no. of test cases executed / Total no. of test cases planned to execute) \* 100 %

* *Test Cases Passed (To measure the percentage no. of test cases passed).*

Test Cases Pass = ( Total no. of test cases passed / Total no. of test cases executed ) \* 100%

* *Test Cases Failed ( To measure the percentage no. of test cases failed).*

Test Cases Failed = (Total no. of test cases failed / Total no. of test cases executed) \* 100%

* *Test Design Efficiency ( To measure the efficiency of test design).*

Test Design Efficiency = Number of tests designed / Total time.

* *Testing Efficiency ( To measure the efficiency of the tests).*

Testing Efficiency = (no. of bugs resolved / Total no. of bugs raised) \*

100%

* Number of Defects Per Test Hour ( To measure the number of defects per test hour).

Number of Defects Per Test Hour = Total no. of defects / Total number of test hours

* + 1. Project Metrics : It is used to monitor the team’s progress and load the efforts to the project’s goals.
* Effort Variance ( To measure how much effort to complete the planned work).

Effort Variance = [( Actual labour hour - estimated effort) / (Estimated labour hour)] \* 100%

* If Effort Variance is positive, this means we took extra time(effort) to complete the planned work.
* If Effort Variance is negative, this means we took less time(effort) to complete the planned work.
* If Effort Variance is zero, this means we are on the estimated effort.
  + 1. Velocity Metrics : to predict how quickly a team can work towards a certain goal by comparing the average story points or hours committed to and completed in previous sprints.
* Average Story points committed to and completed in previous sprints.
  + 1. Results : Iteration0 - TBD
  1. Standard

1. Documentation Standard :

i. All documentations follow the procedures of the professor's standards and

guidelines.

ii. Progress Report : To keep track of individual user’s development and the

ongoing tasks.

iii. Meeting Minutes : To keep track of our meeting decisions and outcomes.

iv. SPPP : To keep track of the scope of the project. It includes everything

about the project idea, project vision and basic project management

planning.

v. Risk Management : To keep track the risks of the project and addresses the

risks.

vi. Testing Report : To keep track the progress of the testing, it contains the

test cases and test results.

vii. README.md : Need to update it consistently based on the project

progressing.

1. Coding Standard :

i. PEP8 is used for the coding standard.

## Inspection/Review Process

* Requirement Review
* Should be reviewed by each of the Agile Team members at the end of each iteration to assess progress and then adjust its backlog for the next iteration.
* Documentation Review
* Team Leader will be responsible for the documentation review at the end of each iteration. Team members will adjust or do changes for their documentations after the review.
* Code Review
* Should perform peer code reviews and all code must be reviewed before checked into the GitHub.
* Codes also can be reviewed by those stakeholders who were not involved in the development for a specific branch.
* All the code reviewers have to update the review status on Pivotal Tracker.
* Test Report Review
* The team members who were not involved in writing and developing test cases for a specific branch will review the test cases and contribute to different kinds of testing(e.g unit test, integration testing...etc).
* The team members who were not involved in writing and developing test cases should perform the execution of the tests and do the review on the testing results.

## Testing:

We will conduct four types of testing:

* **Unit Testing** : Unit testing is the first layer of the testing process. Each developer will be responsible for writing the unit testing for testing a method.

Developers who wrote the unit tests are responsible for executing the unit tests and record the test results in the testing report.

* **Functional Acceptance Testing :** To verify if our application satisfies its business requirements. Based on the acceptance test written for the user stories at PivotalTracker, testers will do the manual testing to verify if the functional features meet the acceptance tests for each story. If time is allowed, QA Developer will automate the acceptance test.
* **Integration Testing** : To test the modules/components when integrated to verify that they work as expected. The individual modules are first tested in isolation. Once the modules are unit tested, they are integrated one by one, till all the modules are integrated and validate whether the requirements are implemented correctly or not. QA Leader will be responsible for the integration testing. All other team members can also be involved in Integration Testing. Those members who performed the Integration Testing should record the test results in the testing report.
* **System Testing** : It is a level of testing that validates the complete and fully integrated software product. The purpose of a system test is to evaluate the end-to-end system specifications. Write some test cases and develop the automated test cases to verify that our application performs and functions correctly according to user specifications. Manual and automated tests can be performed for system testing. It should test the application with different browsers as well. QA Leader will be responsible for the system testing. All other team members can also be involved in the System Testing. Those members who performed the System Testing should record the test results in the testing report.
* **Regression Testing** : This testing is to confirm that a code change or addition has not adversely affected existing features. Previously executed test cases are re-executed in order to verify the impact of change. All team members can do the execution of the regression testing.Those members who performed the Regression Testing should record the test results in the testing report.

Link to the testing report : TBD

## Defect Management

* Git issues will be used to keep track of the defects.
* Testers that find bugs must raise a ticket with the Git issues and assign the bug to a specific developer who developed that functionality. It must includes

the following items :

* Description of the bug
* Step to reproduce the behavior
* Expected Behavior
* Screenshots
* Version number
* Additional Context.

The tester has to add a label for Severity. The tester has to update the test

result in the testing report.

* There will be two labels for the issues : bug and enhancement.
* The developer who received the bug issue has to fix the bug, after fixing the bug, the developer has to commit the change with the fix description and add a label to the bug as “Fixed” and assign it back to the tester.
* The tester who received the assigned “Fixed” bug will do the testing on this bug and close the bug if the test is passed, assign it back to the developer if the tests fails. The tester needs to update the test result in the testing report.
* Bugs and defects will be categorized by severity as described below:

a. **Critical** : This has to be fixed within a day. Any defect that needs

immediate attention which impacts the testing process or the issue

cause the application to crash immediately will be classified under

this category.

b. **High** : These issues affects major functionality.This is the defect or

issue which should be resolved before the release is made.

c. **Medium** : The defect affects minor functionality or non-critical ‘

data. This defect should be resolved after all the serious bugs are

fixed.

d. **Low** : The defect does not affect functionality or data. This defect

can be resolved in the future and does not need any immediate

attention.

# Configuration Management Plan

(For more details, please refer to SCMP document for encounter example)

## Configuration items and tools

Python flask will be used for our development environment, JavaScript will be used for frontend development. And we will update on github/git anytime based on group discussions. Pivotal tracker will be used to track requirements. We will use Pip & requirements.txt along with virualenv to define an environment for our app to run in.

## Change management and branch management

## 1.**Change management**

* Anyone can request a CR via the team Slack channel
* If the CR is not simple or purely cosmetic, team will decide if it requires a zoom group or subgroup meeting
* After the discussion the team leader will finally decide to approve or reject the CR.
* If the change request is approved, the relevant team members implement the change and record it in the team progress log.

**2. Branch management**

* We will have a main branch and a development branch. New features branches will be based off the development branch and merged back into the development branch after review and approval of others on the team. When tests are passing and functionality can be assured, the development branch will be merged into the main branch for major releases (iterations).

## Code commit guidelines

Verify the integrity and runnability of the code before committing.

The commit log should include a brief description of the commit, as well as a detailed description of what was changed and why.

## Integration and deployment plan

All code and artifacts will be stored in the project github repository, potential users will clone the repository and be able to set up a local environment to run the webapp using a setup file. This will create a virtual environment, install the necessary packages and launch a local webserver to view the application. Later on in development, we will explore the feasibility of using Heroku to deploy our application.

# References

1. PEP8 style guide <https://www.python.org/dev/peps/pep-0008/>
2. USDA API Reference: <https://fdc.nal.usda.gov/api-guide.html>
3. Flask User Guide: <https://flask.palletsprojects.com/en/2.0.x/>

(For more details, please refer to the encounter example in the book or the software version of the documents posted on blackboard. )

1. Non-Functional Requirements: <https://www.altexsoft.com/blog/non-functional-requirements/>

# Glossary