Foodify - an online Food Management System

[Project Plan]

Version 1.0

Anandarajan Sindini
Kolady Anamika Martin
Duong Ngoc Yen
Verma Nandini
Chua Jim Sean

RunTime Terror
School of Computer Science and Engineering
Nanyang Technological University, Singapore

Revision History

Revision Number	Date	Primary Author(s)	Comments
1.0	October 1st, 2022	Anandarajan Sindini, Duong Ngoc Yen, Chua Jim Sean	First Version

Table of Contents

1 Introduction

1.1 Project Overview

Foodify is a web application used for food management. It helps its users to record and track the expiry dates of thor food items. It also has a forum for users to exchange food items in case of inability to finish them in order to prevent them from expiring. Foodify uses methods such as recipe recommendations using the registered food items and recommending donating food at food donation drive locations to encourage them against letting their food item go to waste. Using the above functions, it will allow users to reduce their food wastage by managing food better.

1.2 Project Description and Scope

To use Foodify, anyone can simply use their email address to register as a user. A registered user can scan a food item to register it under his/her/their account. The expiry date of the item will be required for registration as well. When a food item has been consumed fully, the user can update the item and it will automatically be deleted. When foods are about to expire, Foodify uses the following methods to alert users and prevent food wastage.

1. **Reduce food wastage due to <u>negligence</u> towards expiry date of food items**

   In order to achieve the first objective, the user will receive notifications from the web application on the following occasions:

   I. 10 days before the expiry of food item
   II. 5 days before the expiry of food item
   III. Last 3 days before the expiry of food item

   These notifications are intended to reduce user negligence to the expiry date of food items by constantly reminding them.

   Along with notifications, users will also receive suggested recipes involving the food items that are about to expire to encourage consumption of the item before expiry. Upon expiry, the registered food item will automatically be removed from the user's account.

2. **Reduce food wastage due to <u>inability</u> to consume food items by the expiry date.**

   In case of user inability to consume food before expiry, a forum also exists to allow users to put up their soon-to-expire food items for sale in order to prevent the food from going to waste. Other users of the forum can express interest by clicking an 'Interested' in the food item, upon which the forum post will become private allowing the seller and the user who expressed interest button to negotiate a price compromise. Selling price of the food item(s) is to be negotiated between the seller and the buyer privately, the web application will not give any price suggestions. If the expected buyer were to become uninterested they could simply click the 'Interested' button again and the post will become public for other users to see. If unbought, upon expiry of the food item, any post regarding the food item will be automatically removed.

Locations of non-profit organisations which collect unwanted food items for donation will also be given to the user to suggest another way to prevent the wastage of food items.

By consistently using this web application, users are able to cut down on their food wastage by either remembering to consume foods or giving them away before expiration, thereby, fulfilling the objectives of the web application.

2 Project Organisation

2.1 Team Structure

The following shows the different team members and their roles in the project:

- Anandarajan Sindini - Project Manager
- Kolady Anamika Martin - Release Engineer and QA Manager
- Duong Ngoc Yen - QA Engineer
- Verma Nandini - Lead developer and Front-end developer
- Chua Jim Sean - Backend developer

2.2 Roles and Responsibilities

Project Manager: Anandarajan Sindini

- Oversees project progress
- Approves and executes project plan
- Assigns tasks and reports status of project to team members
- Manages and motivates team members

Release Engineer and QA Manager: Kolady Anamika Martin

- Oversees quality assurance of project
- Oversees and controls proper placement and deployment of source codes
- Plans and executes project quality assurance plan
- Handles code repositories

QA Engineer: Duong Ngoc Yen

- Plans and executes project quality assurance plan
- Prepares quality assurance tests
- Assists team on quality fixes

Lead developer and Front-end developer: Verma Nandini

- Oversees development team
- Develops user interface of project's web application

Backend developer: Chua Jim Sean

- Develops back-end of web application
- Manages data storage and site performance

2.3 Team Communication

Communication channels include:

- Weekly meetings held on Thursdays
- Group announcements and updates sent through group chat on telegram
- Online discussions are held in between weekly sessions whenever necessary

3 Process Definition

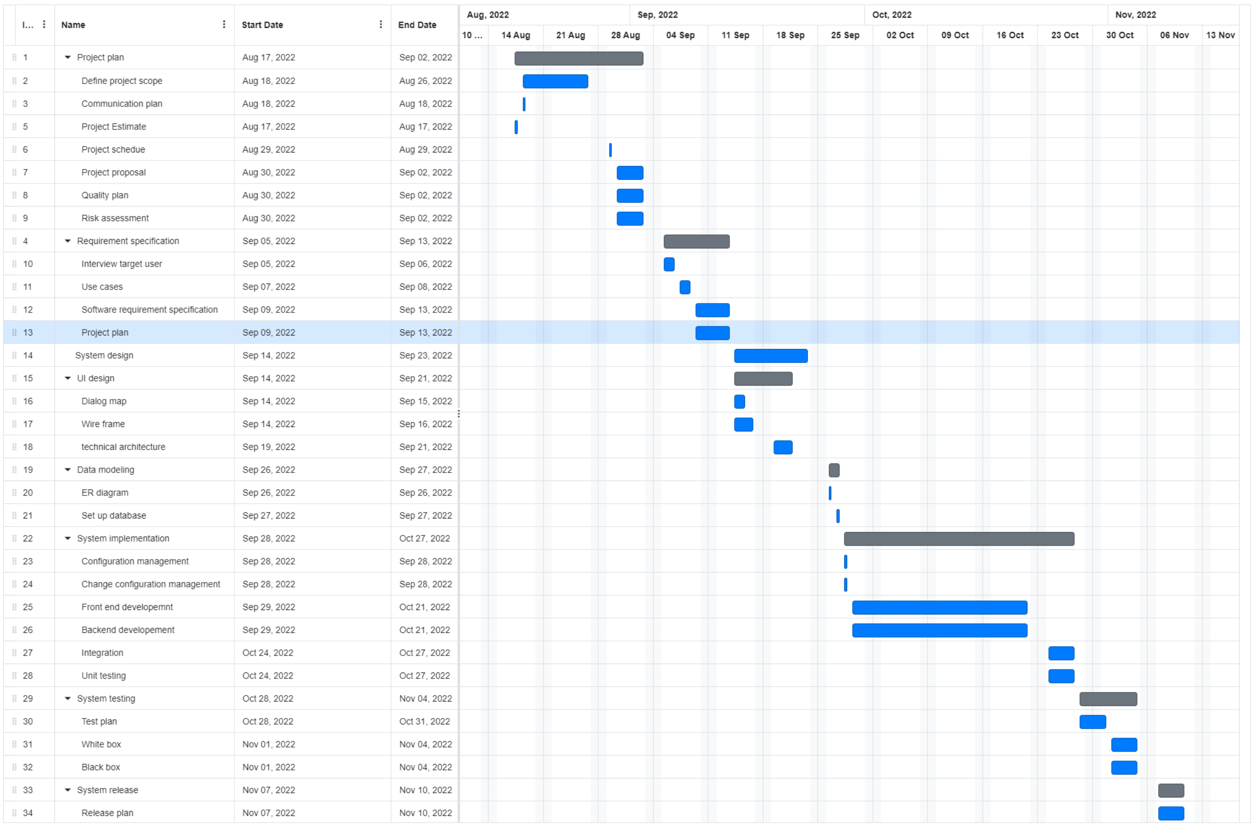
3.1 Lifecycle Model

RunTime Terror intends to utilise the Incremental Development Model for this project. This model was chosen for a number of reasons:

1. Considering the short time period of this project implementation, waterfall SDLC will not be a suitable option. As quality assurance and user acceptance tests get conducted on the project deliverable, further changes and revisions might be required. With waterfall this process will take very long as the whole SDLC process will have to restart for modifications to the project. Hence, using the Incremental Development Model will allow us to continuously improve on the product and save time used up for restarting SDLC.
2. Another usefulness of Incremental Development Model is that it generates working software earlier during the life cycle. This will allow customers to respond to each build of the web application, which will give the chance for developers to build the project as close to the desired outcome as possible while also improving the customer trust and satisfaction.
3. It allows for easier risk management as risks can be identified early with the earlier product versions and fixed more quickly, minimising the risk of the final product.

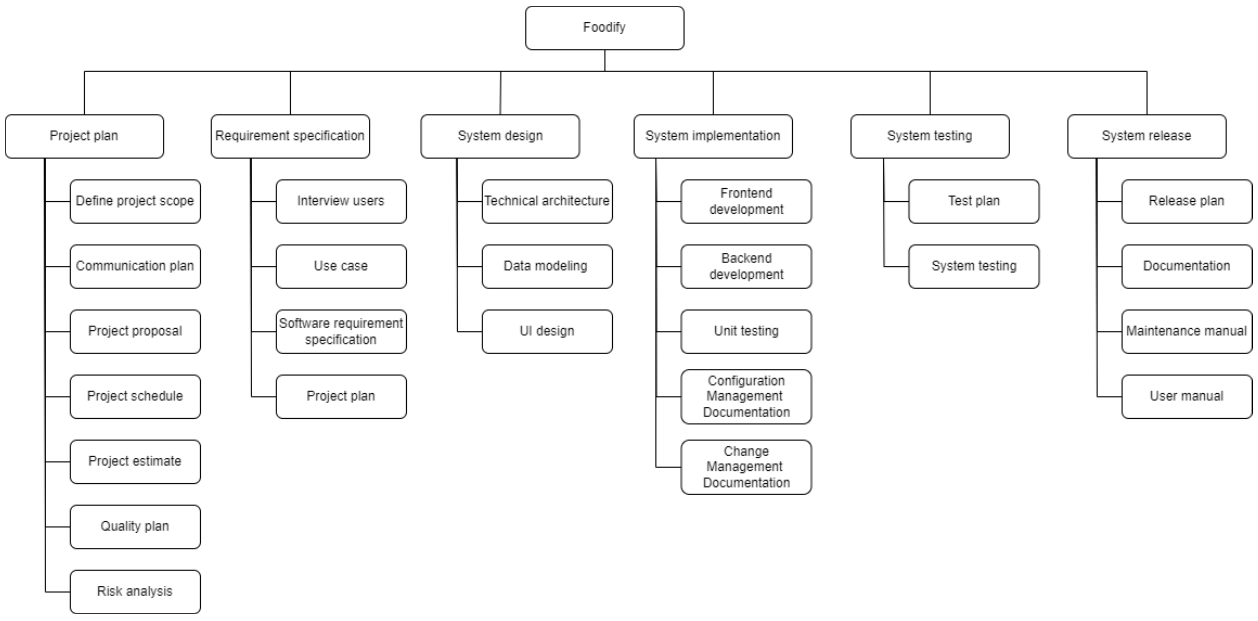
4 Schedule

4.1 Activity Dependencies and Schedule



I...	Name	Start Date	End Date
1	Project plan	Aug 17, 2022	Sep 02, 2022
2	Define project scope	Aug 18, 2022	Aug 26, 2022
3	Communication plan	Aug 18, 2022	Aug 18, 2022
5	Project Estimate	Aug 17, 2022	Aug 17, 2022
6	Project schedue	Aug 29, 2022	Aug 29, 2022
7	Project proposal	Aug 30, 2022	Sep 02, 2022
8	Quality plan	Aug 30, 2022	Sep 02, 2022
9	Risk assessment	Aug 30, 2022	Sep 02, 2022
4	Requirement specification	Sep 05, 2022	Sep 13, 2022
10	Interview target user	Sep 05, 2022	Sep 06, 2022
11	Use cases	Sep 07, 2022	Sep 08, 2022
12	Software requirement specification	Sep 09, 2022	Sep 13, 2022
13	Project plan	Sep 09, 2022	Sep 13, 2022
14	System design	Sep 14, 2022	Sep 23, 2022
15	UI design	Sep 14, 2022	Sep 21, 2022
16	Dialog map	Sep 14, 2022	Sep 15, 2022
17	Wire frame	Sep 14, 2022	Sep 16, 2022
18	technical architecture	Sep 19, 2022	Sep 21, 2022
19	Data modeling	Sep 26, 2022	Sep 27, 2022
20	ER diagram	Sep 26, 2022	Sep 26, 2022
21	Set up database	Sep 27, 2022	Sep 27, 2022
22	System implementation	Sep 28, 2022	Oct 27, 2022
23	Configuration management	Sep 28, 2022	Sep 28, 2022
24	Change configuration management	Sep 28, 2022	Sep 28, 2022
25	Front end developemnt	Sep 29, 2022	Oct 21, 2022
26	Backend developement	Sep 29, 2022	Oct 21, 2022
27	Integration	Oct 24, 2022	Oct 27, 2022
28	Unit testing	Oct 24, 2022	Oct 27, 2022
29	System testing	Oct 28, 2022	Nov 04, 2022
30	Test plan	Oct 28, 2022	Oct 31, 2022
31	White box	Nov 01, 2022	Nov 04, 2022
32	Black box	Nov 01, 2022	Nov 04, 2022
33	System release	Nov 07, 2022	Nov 10, 2022
34	Release plan	Nov 07, 2022	Nov 10, 2022

4.2 Work Breakdown Structure



4.3 Work Packages

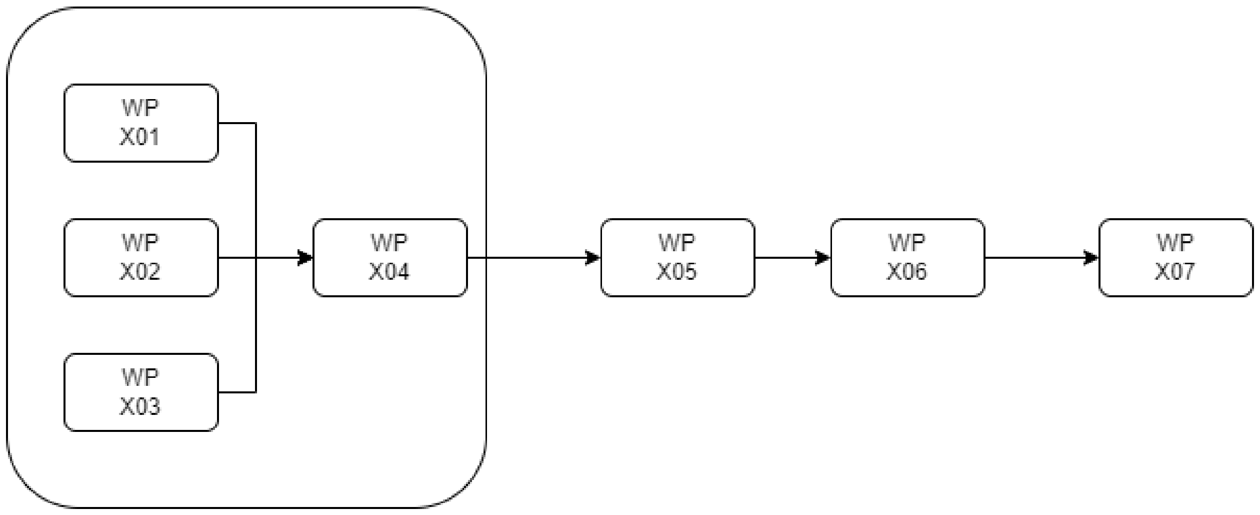
The project consists of multiple packages in the software development life cycle:

1. Project plan
2. Requirement specification
3. User interface
4. Technical architecture
5. Data modelling
6. Implementation & unit testing
7. System testing and quality assurance

4.4 Activity Dependencies

Work packages #	Work package description	Duration	Dependencies
X01	Project plan	8 days	NA
X02	Requirement specification	10 days	NA
X03	User interface	7 days	NA
X04	Technical architecture	14 days	X01, X02, X03
X05	Data modelling	7 days	X04
X06	Implementation & unit testing	30 days	X05
X07	System testing and quality assurance	30 days	X06

The above packages could be illustrated as this:



WPX05 depends on all packages before it. It means package WPX05 would not be started until package WPX01-04 has been finished.

4.5 Work Package Details

Work packages are listed below. A team member, indicated in bold, has been assigned as primarily responsible for each work package and will coordinate that package.

Project	Foodify
Work package	1 - Project Plan (1 of 7)
Assigned to	**Nandini**
Effort	8 days
Start date	20 August
Purpose	Provide an overview of the project
Inputs	None
Activities	Provide a brief overview of the project, objective, problem statements, deliverables throughout the software development cycle.
Outputs	Project plan documentation

Project	Foodify

Work package	2 - Requirement Specification (2 of 7)
Assigned to	**Sean**
Effort	10 days
Start date	20 August
Purpose	To understand customer need, align customer requirement with the developer team
Inputs	Customer's requirement
Activities	Interviewing customers, inspecting and doing surveys.
Outputs	Requirement Specification documentation

Project	Foodify
Work package	3 - User interface (3 of 7)
Assigned to	**Anamika, Yen**
Effort	7 days
Start date	20 August
Purpose	To make an easy-to-use interface
Inputs	User Information
Activities	Getting interface action, getting user information, managing interaction and displaying action between system and user, displaying the result of the request.
Outputs	User Interface

Project	Foodify
Work package	4 - Technical architecture (4 of 7)
Assigned to	**Sindini**
Effort	14 days

Start date	28 August
Purpose	To provide details on the architecture and other technical details for the software developer team.
Inputs	Project plan, Requirement specification, User Interface
Activities	Identifying type of architecture of the software system, components, interaction, hardware infrastructure, software tools, design practices, design pattern.
Outputs	Documentation of High level Design and Architectural Specification

Project	Foodify
Work package	5 - Data modelling (5 of 7)
Assigned to	**Sean**
Effort	7 days
Start date	28 August
Purpose	Plan, implement and build the database to store user's data
Inputs	Project plan, Requirement specification, User Interface, Technical Architecture
Activities	Analysing data flow, entity relationships, deciding database tools, and deploying the tools.
Outputs	Documentation of Data Modelling

Project	Foodify
Work package	6 - Implementation & Unit testing (6 of 7)
Assigned to	**Nandini, Sindini, Sean**
Effort	30 days
Start date	13 September
Purpose	Implement the system and make it into the real product. Conduct unit

	testing to test the functionalities of the system
Inputs	Project plan, Requirement specification, User Interface, Technical Architecture, Data modelling
Activities	Implement the components and the interaction between the components. Conduct unit testing and test the system with different test flow.
Outputs	Source code, Unit test report

Project	Foodify
Work package	7 - System testing and quality assurance (7 of 7)
Assigned to	**Yen, Anamika**
Effort	30 days
Start date	11 October
Purpose	Perform the overall testing to check if the whole system could satisfy the relevant quality standard
Inputs	Project plan, Requirement specification, User Interface, Technical architecture, data modelling, Source code, unit testing.
Activities	Perform black box testing, white box testing, analyse the testing result, identify the strategies for improving the software process,
Outputs	Test report, quality assurance report

5 Project Estimates

5.1 Code Size Estimation using Function Points

We calculated unadjusted function point based on the complexity of functions provided by this system. Code size is then estimated by adjusted function point.

5.1.1 Unadjusted Function Points

Supports the following proposed functions:

User:

- User account Sign Up
- User account Login
- User account management
    - Reset Password
    - Add Location
    - Edit Account
    - Delete
    - Logout
- Add food item to Listing (Sale/Giveaway)
- Add food item (Private)
- Indicate Interest on a food item Listing
- Get reminders on expiring food items
- Add recipes to Community page
- Like/Dislike Recipe Recommendations
- View Recipe Recommendations
- View/Modify/Delete food item listings

Administrator:

- Manage User Accounts
- Manage food item Listings

The measure of unadjusted function points is based on five primary component elements of these functions: Inputs, Outputs, Inquiries, Logical Files, and Interfaces. Each element ranges from Low Complexity, Medium Complexity to High Complexity. The detailed evaluation of the complexity is as follows:

Rating Inputs:

· User Account: (username, email, password, location)
· Food Item (product name, expiry date)
· Food Listing (Food Item, Type of Listing [Sell/Giveaway], Price, Quantity, Expiry Date, Opened/Unopened, Image)
· Recipe Ingredients (Ingredient name, Quantity)
· Recipe Recommendations (Like/Dislike)

Files Type Referenced (FTR)	**Data Elements**		
	1-4	5-15	Greater than 15
Less than 2	Low (3)	Low (3)	Average (4)
2	Low (3)	Average (4)	High (6)
Greater than 2	Average (4)	High (6)	High (6)

Rating Outputs:

· Displaying list of Food Items (private)
· Displaying Recipe Recommendations
· Displaying customer Food Item Listings

File Types Referenced (FTR)	**Data Elements**		
	1-5	6-19	Greater than 19
less than 2	Low (4)	Low (4)	Average (5)
2 or 3	Low (4)	Average (5)	High (7)
Greater than 3	Average (5)	High (7)	High (7)

Rating Inquiries:

· Selecting Food Listings according to the user's criteria
· Selecting Recipe Recommendations

File Types Referenced (FTR)	**Data Elements**		
	1-5	6-19	Greater than 19
less than 2	Low (3)	Low (3)	Average (4)
2 or 3	Low (3)	Average (4)	High (6)
Greater than 3	Average (4)	High (6)	High (6)

Rating Logical Files:

- Food Listings
- Community Page recipes
- User Account

Record Element Types (RET)	**Data Elements**		
	1 to 19	20 - 50	51 or More
1 RET	Low (7)	Low(7)	Average (10)
2 to 5 RET	Low (7)	Average (10)	High (15)
6 or More RET	Average (10)	High (15)	High (15)

Rating Interfaces:

- 2 External Files Referenced (FoodItems, UserAccount)

Record Element Types (RET)	**Data Elements**		
	1 to 19	20 - 50	51 or More
1 RET	Low (7)	Low(7)	Average (10)
2 to 5 RET	Low (7)	Average (10)	High (15)
6 or More RET	Average (10)	High (15)	High (15)

Summary of above analysis:

Element	**Complexity**	**Detail**
Inputs	Low	User Account
	Low	Food Item
	Average	Food Listing
	Low	Recipe
Logical Files	High	Food Listing
	Medium	Recipe Recommendations
	Medium	User Account
Outputs	Low	Display Food Items
	Average	Display Food Items Listings
	Low	Display Recipes
Inquiries	Low	Selecting Food Listing
	Low	Recipe Recommendations Ingredients
Interfaces	Average	Food Items, Recipe Recommendations
	Low	User Account

Calculation of Unadjusted Function Points:

Characteristic	**Low**		**Medium**		**High**	
Inputs	3	× 3	1	× 4	0	× 6
Outputs	2	× 4	1	× 5	0	× 7
Inquiries	2	× 3	0	× 4	0	× 6
Logical Files	0	× 7	2	× 10	1	× 15
Interfaces	1	× 5	3	× 7	0	× 10
Unadjusted FP	28		50		15	
Total=L+M+H	93					

5.1.2 Adjusted Function Points

Influence Factors	**Score**	**Detail**
Data Communications	5	Application is more than a front-end, and supports more than one type of teleprocessing communications protocol.
Distributed Functions	4	Distributed processing and data transfer are online and in both directions.
Performance	3	Response time or throughput is critical during all business hours. No special design for CPU utilisation was required. Processing deadline requirements with interfacing systems are constraining.
Heavily used	2	Some security or timing considerations are included.
Transaction rate	3	Daily peak transaction period is anticipated.
On-line data entry	5	More than 30% of transactions are interactive data entry
End-user efficiency	2	Four to five of the efficiency designs are included
On-line data update	3	Online update of major internal logical files is included.
Complex processing	1	Any one of the complex components
Reusability	4	The application was specifically packaged and/or documented to ease re-use, and the application is customised by the user at source code level.

Installation Ease	1	No special considerations were stated by the user *but* special setup is required for installation.
Operational Ease	1	Effective start-up, back-up, and recovery processes were provided, but no operator intervention is required (count as two items).
Multiple sites	0	User requirements do not require considering the needs of more than one user/installation site.
Facilitate change	3	Flexible query and report facility is provided that can handle complex requests, for example, *and/or* logic combinations on one or more internal logical files (count as three items).
Total score	37	
Influence Multiplier = Total score × 0.01 + 0.65 = 37 × 0.01 + 0.65 = 1.02		
Adjusted FP = Unadjusted FP × Influence Multiplier = 93 × 1.02 = 94.86		

Scoring (0 – 5)
0 = No influence
1 = Insignificant influence
2 = Moderate influence
3 = Average influence
4 = Significant influence
5 = Strong influence

5.1.3 Lines of Code

According to Capers Jones statistics, each Function Point requires 29 lines of code if the application is implemented using C++.
Therefore, we have: **Lines of Code** = 89.76 FP × 29 LOC/FP = **2603 LOC**

5.2 Efforts, Duration and Team Size Estimation

To estimate the effort and duration required for the project, we use function points as the basis to calculate Effort, Duration, Team size and finally the schedule. The estimates are expanded to account for project management and extra contingency time to obtain the total average effort estimates. From these averages, the duration of each work package in working days is estimated based on the following calculations.

- Working days include 5 days in a week.
- Effort = Size / Production Rate = (**2751** LOC) / (39 LOC/PD)[1] = 71 PD
- Duration = 3 × (Effort)$^{1/3}$ = 3 × (71)$^{1/3}$= 12.5 Days
- Initial schedule = 12.5 Days / 5 days a week = 2.5 Weeks
- Team size = 71 PD / 12.5 D = 5.68 P = 6 Persons
- Working hours include 8 hours in a working day.
- Total person-hours (PH) = 71 PD × 8 hours = 568 PH

5.2.1 Distribution of Effort

1990's Industry Data	**Work Package**	**Distribution**	**Estimates**
Preliminary Design 18 %	**Project Plan**	**9%**	**51.12**
	Requirement Specification	**9%**	**51.12**
Detailed Design 25 %	**User Interface**	**7%**	**39.76**
	Technical Architecture	**11%**	**62.48**
	Data Modelling	**7%**	**39.76**
Code & Unit Testing 26 %	**Code & Unit testing**	**21%**	**119.28**
	Online Documentation	**5%**	**28.4**
Integration & Test 31 %	**Integration & Quality Assurance**	**31%**	**176.08**
	Extrapolated total effort		**568**
	2% for project management		**11.36**
	3% for contingency		**17.04**
	Total effort		**596.4**

These duration estimates are based on the assumption that each team member works an equal amount on any given work package.

5.3 Cost Estimates

Hardware:

Developer workstations:

7 - Dell Precision Workstation 330	Total $0.00
Pentium IV 1.4GHz single processor	
256 MB RAM	
20.8GB IDE drive	

Software:

GNU, Apache, or Other Free Licence-based Software:

Apache Web Server	$0.00
Perl	$0.00

Software Licence Provided by Third Party:

Microsoft Office 2000	$0.00
Microsoft Project 2000	$0.00
Products ESTIMATE Professional	$0.00

Other Resources:

Staff:

7 Employees with 568 working hours with $18.00/hour	$10,224.00

Stationary:

Paper, photocopying and other miscellaneous cost	$50.00

Total:

$10,224.00

The customer will supply the required hardware and software necessary to run the SQL server and the E-Commerce web server. TLA is not responsible in any way for supplying said systems. TLA 's hardware and software responsibilities relate only to our own development needs to accomplish the project we have been asked to complete, and which has been described in the introduction section of this document. TLA will also demonstrate the completed product.

6 Product Checklist

The project deliverables will be delivered on the stated deadlines:

Project Deliverable	**Estimated Deadline**
Requirements Analysis	Sep 22nd, 2022
Project Plan	Oct 6th, 2022
Design Document	Oct 13th, 2022
Module /System Test Plan	Oct 13th, 2022
System Release (Demo)	Oct 20th, 2022

7 Best Practice Checklist

Practice	Check
Document everything that is done in a standardised format	☐
In requirement specification, avoid ambiguity, incompleteness, inaccuracy and inconsistency. Requirements should be documented clearly and must contain a complete functional specification.	☐
Avoid fancy or unnecessary product functions, product only needs to meet the customer requirements. Minimise mixing the data and tasks of individuals. Ensure that key information is communicated to all members of the team.	☐
Team must have a shared vision towards what they are building. Any doubts, clarifications or issues must hence be communicated immediately with team members. When giving review or suggestions on each other's works, team members must be honest and constructive so that the overall team can benefit from individual actions.	☐
Plan for continuous change. We must: • All manuals designs, test, source code should have revision numbers and dates revision history comments, change marks to indicate the changes • New revisions should be approved before being made and checked for quality and compliance after being made	☐
When planning any task, do not underestimate the time, effort and overhead required for it. Add an optimal buffer time to tasks so that the project progresses with the planned overall timeline.	☐
Code reviews are a much more efficient method to find software defects. Plan and manage code reviews between team members	☐
Testing must include both black box and white box testing. It must involve unit, functional, integrating and acceptance testing.	☐
Business and code integrity must always be upheld by all team members.	☐

8 Risk Management

The following risks (not including the risks under general risk management) are evaluated for the project:

Scope Creep - scope morphing into something it was initially not
Impact severity: High
Probability: 25%
Impact: Could cause the initial functional requirements and requirement specifications to become irrelevant. Project might be much different than the expected product and might not be an efficient product that satisfies customers as the vision that the team had was not met.
Risk Reduction: Be consistent with the requirement specification before every iterative round. If requirement is not specified, then ensure a thorough discussion with customer and proper new requirement specification has been completed before proceeding to work towards the new requirement(s).

Changes to team size or members before project completion
Impact severity: High
Probability: 5%
Impact: Workload of team members may change; parts of project uncomplete; managers might not be able to measure progress of team.
Risk Reduction: Follow communication plans as documented in section 2.3.

System size underestimated
Impact severity: Moderate
Probability: 30%
Impact: More effort will need to be put in on design and coding, could hinder project's progress
Risk Reduction: Update estimates often as the project progresses.

Budget Issues
Impact severity: Extreme
Probability: 25%
Impact: If budget breach occurs, customer satisfaction will be compromised. Project might struggle to get additional funding. Even if further funding is given, project duration will be lengthened and planning needs to be done again.
Risk Reduction: Plan budget well before the project starts. Strictly stick to the budget and monitor it constantly. All expenses must be known to all team members.

Project Cancellation
Impact severity: Very-extreme
Probability: 1%
Impact: Project efforts become completely wasted.
Risk Reduction: Before project commencement, ensure the customer did market research on the demand for this project. Constantly communicate with customers and let them know the progress to show that the project is going well.

9 Quality Assurance

A set standard for quality assurance will be followed by the team when doing quality tests on the project. The specifics of these test procedures can be found in the Quality Plan documentation.

Foodify will undergo these tests mainly:

- **Unit Testing** - testing of individual components of the software
- **System Testing** - testing of the whole system
- **Acceptance Testing** - testing for customer acceptance and satisfaction of the software

These methodologies were chosen as they will ensure success in the main aspects of Foodify.

- **System Functionality** - ensure that the system is able to function properly with short response time and is able to fulfil all functional requirements
- **UserAcceptance** - ensure the system is able to satisfy the users and customers in terms of design and functionality

The above-mentioned test methodologies will be done using practical test cases. As part of the final project deliverable, detailed test data will be provided.

10 Monitoring & Control

Some key methods of monitoring the progress of the software includes:

Timeline plan and following: During the timeline planning, underestimation of time required for tasks to be completed should not occur. Time allocated for each task should always have a buffer time, to make up for time that could be lost on any internal or external uncertainties. Timeline should be planned to be as detailed as possible and with every major or minor task accounted for. Schedule of the team members should be considered while assigning tasks so as to ensure that each individual can complete their tasks with the time they were allocated and follow through with the timeline.

Identifying major risks beforehand: Identifying potential risks beforehand will allow the team to plan the preventive measures to eliminate the risk beforehand as well, thereby minimising the chance of losing the project or delaying it due to the risks. Major risks and their preventive measures have been identified in the risk management section of this document in section 8.

Resource consumption measurement: Monitoring budget is very crucial and for that all resource consumption must be calculated, monitored and recorded. A quantitative measurement of resource consumption would make it easier for tracking and recording resource consumption. The percentage estimates of each milestone's resource requirements provided in this document allow for easy progress tracking.

Regular review of project progress: Regular review could be conducted through any means of communication as listed in section 2.3. These regular reviews will keep team members who are working on different aspects of the project to be aware of what the other team members have been doing and where they, as well as the whole team, stand in terms of their timeline and progression towards the final deliverables. Other than just progress, any new risks or hindrances to the project that has been recently identified should be discussed as well.