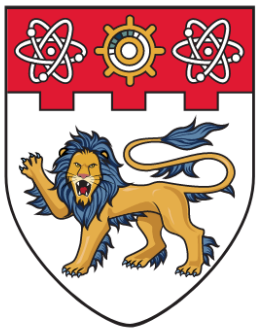

NANYANG TECHNOLOGICAL UNIVERSITY
SINGAPORE

Project Plan for Planit:

Team Members	Role
Ryan Tan Jinn En	Project Manager
Neo Yong Tai	Development Lead
Lee Yu Sheng Daniel	Back-End Developer
Mamuduri Paulani	Front-End Developer
Frankie Ye Htet Myat	Release Engineer / Manager
Chen Xueyao	QA Manager
Kundu Koushani	QA Engineer

Name of Design Team: **Team Syan**
School of Computer Science and Engineering, Nanyang Technological University

Version History

Version #	Implemented By	Revision Date	Reason for Revision
1.0	Ryan Tan	29/09/2021	First Version
1.1	Ryan Tan	09/10/2021	Added in relevant visuals

Table of Contents

1 Introduction

1.1 Project Overview

The Planit application will provide visual references for a variety of gardens, monitoring tools for specific crop growth, and streamlined guidelines plus solutions for easy user reference. Its features will remove the guesswork involved in gardening, ensuring gardeners with peace of mind.

1.2 Project Description and Scope

The Planit application is an application targeted towards plant owners from the general public in Singapore. Available as a web application, Planit takes advantage of the flexibility of a web application to reach out to more users. Planit aims to provide the average gardener with flexible gardening solutions, as well as plant monitoring and growth analysis tools.

Due to the nature of the system as well as time and budget constraints, the Planit application comes with some limitations:

- The plant database will contain mostly plants that can grow in Singapore's temperate climate. Gardeners who are growing exotic plants that require special growth conditions may not be able to find their plant within the database.
- Due to the limited time period, there is a lack of pre-release user testing and feedback.
- Due to time and budget constraints, Planit currently does not come as an iOS or Android application.

2 Project Organization

2.1 Team Structure

The following is the list of executive roles, as required by CMM level 3.

- Project Manager: Ryan Tan Jinn En
- Development Lead: Neo Yong Tai
- Front-End Developer: Lee Yu Sheng Daniel
- Back-End Developer: Mamuduri Paulani
- Release Engineer/Manager: Frankie Ye Htet Myat
- QA Manager: Chen Xueyao
- QA Engineer: Kundu Koushani

2.2 Roles and Responsibilities

Project Manager: Ryan Tan Jinn En

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

Development Lead: Neo Yong Tai

- Oversees code and development progress
- Approves and executes programming from both front/back-end designs
- Assigns tasks and reports status of QA planning and execution to QA Engineers

Front-End Developer: Lee Yu Sheng Daniel

- Build the prototype on React
- Construct product's front-end/ User Interface using React
- Design User Interface
- Ensure stability and response time of the system meet the requirements
- Creates User manual

Back-End Developer: Mamuduri Paulani

- Responsible for coding back-end in NodeJS
- Responsible for implementing database in MongoDB
- Develops concepts and algorithms required for general app functionality
- Ensure server deployed on AWS functions correctly
- Application reacts appropriately according to command inputs from the user

Release Engineer/Manager: Frankie Ye Htet Myat

- Manage, plan, schedule and control the Planit app through its various stages and

environments

- Required to test both the testing and quality for release purposes

QA Manager: Chen Xueyao

- Oversee QA progress
- Approve and execute QA plan
- Assigns tasks and report status of QA planning and execution to the QA engineers

QA Engineer: Kundu Koushani

- Develop and implement proper testing tools to ensure application functions properly
- Assess quality of specifications and technical design documents to ensure timeliness, relevancy and meaningful feedback
- Assist in planning and implementing strategies for quality management and testing

2.3 Team Communication

Syan communication channels include the following:

1. Weekly meetings are held on Mondays.
2. Group announcements and updates are sent through WhatsApp group chat
3. Discord discussions are held as necessary.
4. Split up into subgroups as necessary, in order to work more cooperatively on specific problems.
5. Github Board for communicating issues and code related changes

3 Process Definition

3.1 Lifecycle Model

Syan intends to use the Incremental Development Model throughout the Plantit project. This methodology is more flexible than the traditional Waterfall SDLC due to repeated iterations involving design, coding, unit testing, integration, and quality assurance. The Waterfall SDLC is not a viable choice due to the short timeline available for the Planit project to reach delivery quality.

Syan has chosen to avoid such methodologies as Spiral because of concerns over the short timeline. Should design procedures, for example, need to be revisited within the first release date, it is likely that the project will overshoot its critical schedule.

Syan intends to deliver the first iteration of functionality on the System Delivery date indicated in the Estimations section of this document. After further client interaction, further iterations should occur as necessary.

Due to the intelligent nature of this agent, further iterations interleaved with client interaction and live testing should hone the algorithms used such that they behave in an accurate and logical manner, providing a more effective search experience for users.

4 Schedule

4.1 Activity Dependencies and Schedule

Tasks	Project Timeline								
	23/08	30/08	06/09	13/09	20/09	27/09	04/10	11/10	18/10
Project planning and setup									
Design Sprint									
Establish Use Cases									
Plan epics, sprints and backlog									
Craft Project Proposal									
Product Design									
Low-Fi wireframe									
High-Fi wireframe									
Map User Journey									
Interactive Prototype									
Development									
Establish server and databases									
Backend implementation									
Login/Registration View									
Main Panel View									
Add Plant Info View									
Notifications and Alerts									
Edit/Delete Plant View									
QA & Testing									
Write Unit Tests									
Set criteria for User Acceptance Test									
Finalize UAT									
User checks UAT of initial prototype									
User checks UAT of final prototype									

Fig.1 Gantt Chart representing schedule for Planit Project

4.2 Work Breakdown Structure

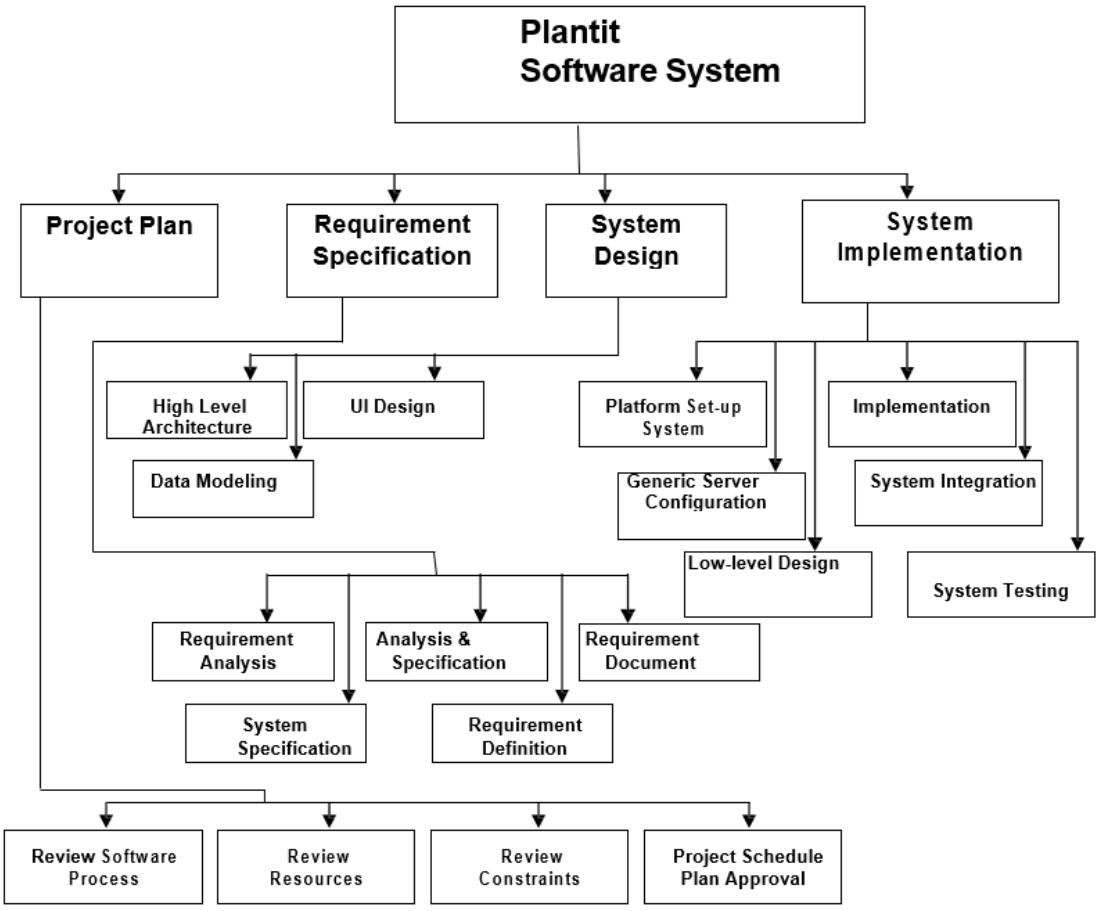

Fig. 2 Work breakdown structure for Planit Project

4.3 Work Packages

The entire project work is broken down by the important phases of the software development life cycle. They include the following:

1. Project Plan
2. Requirement Specification
3. User Interface
4. Technical Architecture
5. Coding & Unit Testing
6. Integration & Quality Assurance

4.4 Activity Dependencies

The following table describes the dependencies of the deliverable work packages:

Work Package #	Work Package Description	Duration	Dependencies
X01	Project Plan	14 days	--
X02	Requirement Specification	14 days	--
X03	User Interface	24 days	--
X04	Technical Architecture	15 days	X01,X02,X03
X05	Coding & Unit Testing	10 days	X04
X06	Integration & System Testing	30 days	X05

The following Activity Network Diagram describes the above in more graphical detail:

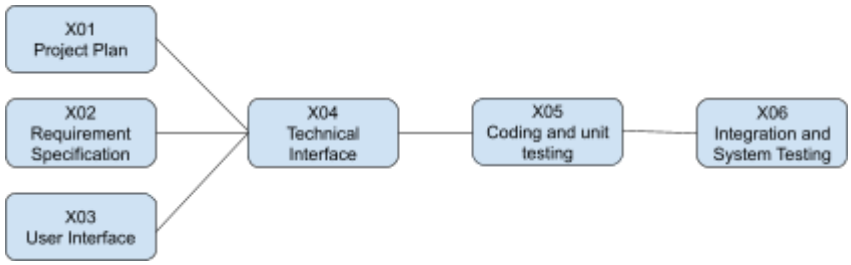

Fig. 3 Activity dependency diagram for Planit

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	Planit Application
Work Package	X01 - Project Plan (1 of 8)
Assigned To	Ryan Tan Jinn En, Mamuduri Paulani, Lee Yu Sheng Daniel, Neo Yong Tai, Frankie Ye Htet Myat, Kundu Koushani, Chen Xueyao
Effort	14 PD
Start Date	Monday, 23/08/21
Purpose	To determine an introductory overview of the project, to be refined in later work packages.
Inputs	None
Activities	This work package includes providing a brief overview of the project, its objectives, and a set of proposed project deliverables throughout the development of the software cycle. The people responsible for this work package will also be transcribing ideas brought up in the group meeting discussion into a formal report.
Outputs	A written document of the Project Plan Introduction.

Project	Planit Application
Work Package	X02 - Requirement Specification (2 of 8)
Assigned To	Ryan Tan Jinn En, Mamuduri Paulani, Lee Yu Sheng Daniel, Neo Yong Tai, Frankie Ye Htet Myat, Kundu Koushani, Chen Xueyao
Effort	14 PD
Start Date	Monday, 10/09/21
Purpose	To establish a common understanding between the customer and the

	software project team of the customers' requirements to be addressed by the project.
Inputs	Customer's requirements Outputs
Activities	Identify "the customer", write and inspect customer requirements and build requirements.
Outputs	A written document of the requirement specification

Project	Planit Application
Work Package	X03 - Quality Plan (3 of 8)
Assigned To	Ryan Tan Jinn En, Mamuduri Paulani, Lee Yu Sheng Daniel, Neo Yong Tai, Frankie Ye Htet Myat, Kundu Koushani, Chen Xueyao
Effort	14 PD
Start Date	Monday, 10/09/21
Purpose	To establish the goals, processes, and responsibilities required to implement effective quality assurance functions for the Planit project. To define and provide the necessary framework to ensure a consistent approach to software quality assurance throughout the project life cycle.
Inputs	-
Activities	Identify the processes that require quality assurance and identify product and process assessments to be performed during the development of Planit.
Outputs	A written document of the quality plan.

Project	Planit Application
Work Package	X04 - Application Prototype (4 of 8)
Assigned To	Neo Yong Tai, Ryan Tan Jinn En, Mamuduri Paulani, Lee Yu Sheng Daniel,

	Frankie Ye Htet Myat, Kundu Koushani, Chen Xueyao
Effort	21 PD
Start Date	Monday, 20/09/21
Purpose	To create a simple prototype of the application that will aid the team in identifying potential problems, prevent costly mistakes and encourage a faster and more effective design cycle.
Inputs	Requirement specifications
Activities	Create a prototype that simulates all major functions of the Planit application: 1. Login 2. Create, view and modify plants 3. View plant growth record and plant growth chart 4. Search web results
Outputs	A functional prototype of the Planit application

Project	Planit Application
Work Package	X05 - Technical Architecture (5 of 8)
Assigned To	Neo Yong Tai, Ryan Tan Jinn En, Mamuduri Paulani, Lee Yu Sheng Daniel, Frankie Ye Htet Myat, Kundu Koushani, Chen Xueyao
Effort	14 PD
Start Date	
Purpose	To do the high level architecture design
Inputs	Project Plan Work Packages (X01 to X03 inclusive).
Activities	High level design entails defining the architecture of the software system and identifying the various components and how they are inter-related to and interactive with each other. Designers also need to decide on the software and hardware infrastructures, such as what operating system on which the software is built, the language used to implement the

	software, and so on. Design topics including maintainability, portability, and reusability will be addressed here as well.
Outputs	High Level Design and Architectural Specification.

Project	Planit Application
Work Package	X05 - Risk Management Plan(5 of 8)
Assigned To	Neo Yong Tai, Ryan Tan Jinn En, Mamuduri Paulani, Lee Yu Sheng Daniel, Frankie Ye Htet Myat, Kundu Koushani, Chen Xueyao
Effort	14 PD
Start Date	20/9/2021
Purpose	To identify and evaluate risks impacting the scope, business, schedule and budget of the project. To reduce, mitigate and eliminate harmful threats in a timely manner to minimize the impact on the project.
Inputs	-
Activities	Identify and analyse the risk of the project. Create a risk response plan to handle the risk during the development of Plantit
Outputs	A written document of the risk management plan

Project	Planit Application
Work Package	X06 - Coding & Unit Testing (6 out of 8)
Assigned To	Neo Yong Tai, Ryan Tan Jinn En, Mamuduri Paulani, Lee Yu Sheng Daniel,

	Frankie Ye Htet Myat, Kundu Koushani, Chen Xueyao
Effort	28 PD
Start Date	Monday, 20/09/21
Purpose	To implement the system as per the requirements specification and other associated documents. This work package includes such additional activities as preliminary unit testing
Inputs	Work Packages X01-X05
Activities	Programmers will implement the modules according to the design specifications noted in the Specification document. Unit tests will be conducted to ensure the functionality of the modules.
Outputs	Source code and header files Unit test results

Project	Planit Application
Work Package	X07 - Integration & System Testing (6 out of 6)
Assigned To	Neo Yong Tai, Ryan Tan Jinn En, Mamuduri Paulani, Lee Yu Sheng Daniel, Frankie Ye Htet Myat, Kundu Koushani, Chen Xueyao
Effort	28 PD
Start Date	Monday, 11/10/21
Purpose	To identify and fix logical and syntactical errors produced during the implementation of the System, and setting up drivers and stubs to see how the module responds to various inputs. Black box testing as well as white box testing might be conducted to check for logical errors. All the testing procedures will be documented in the Test Plan report.
Inputs	Work Package X06
Activities	The Integration testing team may try to simulate how a user might interact with the system. Similar to Unit Testing, Integration Testing may require the development of stubs and drivers as well, but here this is

	more geared towards the higher (overall system) level. Testers may also examine issues such as system performance and integrity. Heuristics assessment plays an important role in this work package, as intelligence components will define eventual system success.
Outputs	Test report Fully functional Planit app Source code

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

Planit supports the following proposed functions:

User:

- Login
- View/Modify/Delete plants
- View/Complete routine and warning notifications
- Search for plants
- Search for answers for your queries online

Administrator:

- Delete users
- Edit notifications

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:

- Gathering plants information: (types of plants, start date and priority) and Constraints (i.e. sorting the results by priority, types and date)
- Gathering user information (email address, password, first name, last name)
- Gathering results for search queries online

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 a list of the results matching the user's search and filter criteria
- Displaying plant of user
- Displaying notifications of plant
- Displaying user account information

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 the plant based on filtering or searching
- Selecting notification of plant
- Selecting user account information

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:

- Plants Information
- Users Information

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:

- 3 External Files Referenced (PlantTypes, GrowthEnvironment, Priority)

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	Gathering Plant Information
	Low	Gathering User Information
Logical Files	High	Plant Information
	Medium	User Information
Outputs	High	Display Search Results
	Low	Display Plants
	Low	Display User Account Information
Inquiries	High	Selecting Plant
	Low	Selecting Plant Notification
	Low	Selecting User Account Information
Interfaces	Medium	PlantTypes, GrowthEnvironment
	Low	Priority

Calculation of Unadjusted Function Points:

Characteristic	Low		Medium		High	
Inputs	2	× 3	0	× 4	0	× 6

Outputs	2	× 4	0	× 5	1	× 7
Inquiries	2	× 3	0	× 4	1	× 6
Logical Files	1	× 7	0	× 10	1	× 15
Interfaces	1	× 3	4	× 7	0	× 10
Unadjusted FP	30		28		28	
Total=L+M+H	86					

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 utilization 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 customized 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 = 88 × 1.02 = 89.76		

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 Javascript.
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 = (2603 LOC) / (39 LOC/PD)[1] = 67 PD
- Duration = 3 × (Effort) $^{1/3}$ = 3 × (67) $^{1/3}$ = 12.2 Days
- Initial schedule = 12.2 Days / 5 days a week = 2.44 Weeks
- Team size = 67 PD / 12.2 D = 5.49 P = 6 Persons

- Working hours include 8 hours in a working day.
- Total person-hours (PH) = 67 PD × 8 hours = 536 PH

5.2.1 Distribution of Effort

1990's Industry Data	Work Package	Distribution	Estimates
Preliminary Design 18 %	Project Plan	9%	48.24
	Requirement Specification	9%	48.24
Detailed Design 25 %	User Interface	7%	37.52
	Technical Architecture	11%	58.96
	Data Modeling	7%	37.52
Code & Unit Testing 26 %	Code & Unit testing	21%	112.56
	Online Documentation	5%	26.8
Integration & Test 31 %	Integration & Quality Assurance	31%	166.16
	Extrapolated total effort		536
	2% for project management		10.72
	3% for contingency		16.08
	Total effort		562.8

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

[1] Lines of code per Person Day statistics based on Industrial Benchmarks, 1997: 31 LOC/PD for United States; 62 LOC/PD for Canada

5.3 Cost Estimates

Item	Supplier	Quantity	Unit Price	Total
Project Manager		1	$30,000	$30,000
Project Team Members		6	$10,000	$60,000
Computers	DELL	7	$1,000	$7,000
Publication License	Google, Apple	2	$100	$200

Server	AWS	1	$2,800	$2,800
Database	MongoDB	1	$15,000	$15,000
Office Rent		1	$6,000	$6,000
			Total	$121,000

Planit is not responsible in any way for supplying said systems. Planit'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. Planit will also demonstrate the completed product.

6 Product Checklist

The items listed below are planned to be delivered on the stated deadlines.

Project Deliverable	Estimated Deadline
Requirements Specification	20th Sep, 2021
Project Plan	11th Oct, 2021
Module/System Test Plan	01st Nov, 2021
System Release (Demo)	01st Nov, 2021

7 Best Practice Checklist

Practice
Document what we do and standardize all documentations.
Pay attention to requirements, checking for ambiguity, completeness, accuracy, and consistency. The requirement documentation must contain a complete functional specification for the application.
Keep it simple. Complexity management is one major challenge. Aim for the following: • Minimize interfaces between modules, procedures and data. • Minimize interfaces between people, otherwise exponential communication cost • Avoid fancy product functions; ensure functionality meets customer requirements, meaning it must be user-friendly
Require visibility. We must see what we build or else we can measure the progress and take management action. This includes: the manager must have good communication with his or her team members, require developers to make code available for review, and review design for appropriateness.
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 changes made • New revisions should be approved before being made and checked for quality and compliance after being made
Do not underestimate. Be careful in obtaining accurate estimates for: time, effort, overhead, meeting time, and especially effort on integration, testing, documentation and maintenance.
Code reviews are efficient for finding software defects. Plan and manage code reviews between team members.
Software testing will use both black box and white box testing. It will involve unit, functional, integrating and acceptance testing.

8 Risk Management

Besides the general risk management, the following risks have been identified for the Planit project:

Covid-19 Pandemic related delays
Impact severity: Moderate
Probability: 15%
Impacts: Delay in meeting up with stakeholders due to volatile Covid-19 situation and policies. Team members (not ill) under quarantine due to close contact with affected people. Thus, the schedule is pushed back.
Risk Reduction: Account and plan for alternative methods of meeting. Keep up with the news regularly to be prepared for changes.

Changes to requirements
Impact severity: High
Probability: 20%
Impacts: Depends on the stage at which changes occur. This ranges from needing to update the requirements documentation to doing a complete redesign.
Risk Reduction: Be rigorous in eliciting requirements. Make stakeholders aware of potential repercussions of requirement changes.

Specification delays
Impact severity: High
Probability: 10%
Impacts: Delay in finalizing the specification will push back schedule for all following stages of the project.
Risk Reduction: Stringent monitoring of specification progress.

Underestimating system size
Impact severity: Moderate
Probability: 35%
Impacts: More work will need to be spent on design and coding, negatively impacting schedule.
Risk Reduction: Update estimates regularly as the project proceeds.

Group coordination issues
Impact severity: Moderate
Probability: 40%
Impacts: Members may be unaware of what is expected of them. Managers may not be

able to measure progress adequately. Thus, portions of projects are not completed.
Risk Reduction: Follow communication plans as documented in section 2.3.

Customer cancels project
Impact severity: Extreme
Probability: 1%
Impacts: All work done will have been wasted.
Risk Reduction: Keep in close contact with the customer, ensuring that they have done sufficient market research indicating a demand for this product.

9 Quality Assurance

We will achieve quality assurance by following the standard set by the company. The specific procedures and details shall be provided in the Quality Plan.

Specific test procedures and details shall be provided in the Module/System Test Plan.

In addition, Planit shall make use of two testing methodologies:

- **Unit Testing** involves testing system components individually.
- **In-Place Testing** involves testing of the whole system as a unit.

Furthermore, these methodologies will be used to test two important aspects of the Planit:

- **System Function** will be tested to ensure that software flaws are eliminated, and
- **Algorithmic Function** will be tested to ensure that heuristic aspects of the project (such identifying the needs of the based plant health and needs) perform realistically to provide value to the users.

The methodology makes broad use of realistic test cases. Detailed test data is an important part of the final project delivery. Syan shall provide a comprehensive and detailed subset of this data for testing purposes.

10 Monitoring & Control

Many procedures are required in order to be able to successfully monitor the progress of a software project. Some of the most important are:

Quantitative measurement of resource consumption: Estimates of Planit project's resource requirements, primarily in terms of human resources, can provide a quantitative measurement of project progress when compared to progress in terms of project milestones. The percentage estimates of each milestone's resource requirements provided in this document allow for easy progress tracking.

Identification of major project risks: Early identification of major risks to the project allows for placement of preventative measures before problems can develop. Major risks have been identified in the Risk Management section of this document, along with the measures being taken to avoid them.

Regular reviews of project progress: Throughout the duration of Planit project, we shall meet weekly to review the progress of all project tasks, including management, planning, analysis, development, and testing.

Timeline Planning and task decomposition: This document outlines an estimated timeline for the project. A reasonably accurate timeline can be assembled by hierarchically decomposing tasks into measurable subcomponents and estimating requirements for each. At the same time, this decomposition can assist in task assignment and balancing. Throughout the implementation phase, these subcomponents can allow for fine-grained measurement of progress. Project subcomponents and timeline estimates are included in the Estimates and Work Breakdown Structure sections of this document.