NANYANG TECHNOLOGICAL UNIVERSITY
SINGAPORE

QMePls

Project Plan

Date: 23rd September 2021

Seow Jing Hng Aloysius
Jacob Law Zhen
Jolene Tan
Soh Qian Yi
Samuel Lee Si En
Zeta Chua Hui Shi

Team Titans
School of Computer Science and Engineering

Revision History

Name	Date	Changes	Version
Jolene	23/9/2021	Uploaded Initial Template	1.0
Jolene	25/9/2021	Introduction, Project Organisation	1.1
Aloysius	27/9/2021	Quality Assurance, Risk Management	1.2
Jacob	30/9/2021	Product Checklist	1.3
Qian Yi	5/10/2021	Schedule	1.4
Samuel	10/10/2021	Monitoring & Control	1.5
Zeta	11/10/2021	Best Practice Checklist	1.6
All	12/10/2021	Project Estimates	1.7
Qian Yi	13/10/2021	Format Document	1.8

Table of Contents

1. Introduction

1.1 Project Overview

With the rise of COVID19 cases, further gathering restrictions and more people taking Pre Event Tests (PET), clinics will start to be more crowded and waiting times will surge. Our application aims to enhance the queuing experience by allowing patients or people taking their PETs to join a queue at their selected clinic.

1.2 Project Description and Scope

QMePls is an application aiming to reduce congestion, boost efficiency and provide convenience for patients or people taking their Pre Event Tests (PET) at nearby clinics. Clinic goers are able to check the queue statuses of nearby clinics, add their symptoms to a report card and join the current queue, thus only needing to turn up at the clinic when their turn is nearing.

Furthermore, clinics are able to control this queue to stop taking patients, allow patients to book or even push late patients down the queue line.

Our application is limited to mainly a queuing system which can be manipulated and managed by the clinic staff themselves respectively. All other administrative matters such as payment, consultation and collection of medication cannot be done through this application and must be carried out in person.

The system consists of a database for the patients and one for symptoms where it is stored through Firebase. We are also using XML for the frontend and Java for the backend development.

2. Project Organization

2.1 Team Structure

Name	Role(s)
Aloysius Seow	Quality Assurance Manager/Engineer
Jacob Law	Lead Developer
Jolene Tan	Project Manager
Soh Qian Yi	Release Manager/Engineer
Samuel Lee	Back-end Developer
Zeta Chua	Front-end Developer

2.2 Roles and Responsibilities

Project Manager: Jolene Tan

- Makes the project proposal, project planning and scheduling
- Overseas the project progress
- Assign tasks and organise meetings and discussions
- Manages and motivates team members

Lead Developer: Jacob Law

- Plans product development and timelines.
- Discuss and aid project managers to ensure that the development team and technologies selected are aligned with the business' goals and vision of the product and its implementation.

Front-End Developer: Zeta Chua

- Implement the product's end front development using Android Studios (Java)
- Liaise with backend to facilitate necessary data exchange between frontend and backend

Back-End Developer: Samuel Lee

- Implements the product's back end infrastructure based on the detailed design documents including the System Requirement Specification (SRS) document
- Liaise with frontend to facilitate necessary data exchange between frontend and backend

Quality Assurance Manager/Engineer: Aloysius Seow

- Helps with software quality assurance plan and risk management plan
- Ensures acceptable software quality
- Designs testing strategies
- Executes the test procedures

Release Engineer/Manager: Soh Qian Yi

- Ensures accurate test cases results for the system with reference to functional requirements
- Organise meetings for the team for review and analysing of test results
- Manages release processes and ensures an error-free system before public release
- Ensures that all versions of the system are properly documented.

2.3 Team Communication

Team "Titans" communication channels includes the following:

- Bi-weekly physical meetings held on Thursday mornings (8.30am to 10.30am) when all members are available
- Group announcements/reminders are sent through project Telegram group
- Weekly online Microsoft Team discussion held
- Splitting into subgroups to work more cooperatively on specific problems

3. Process Definition

3.1 Lifecycle Model

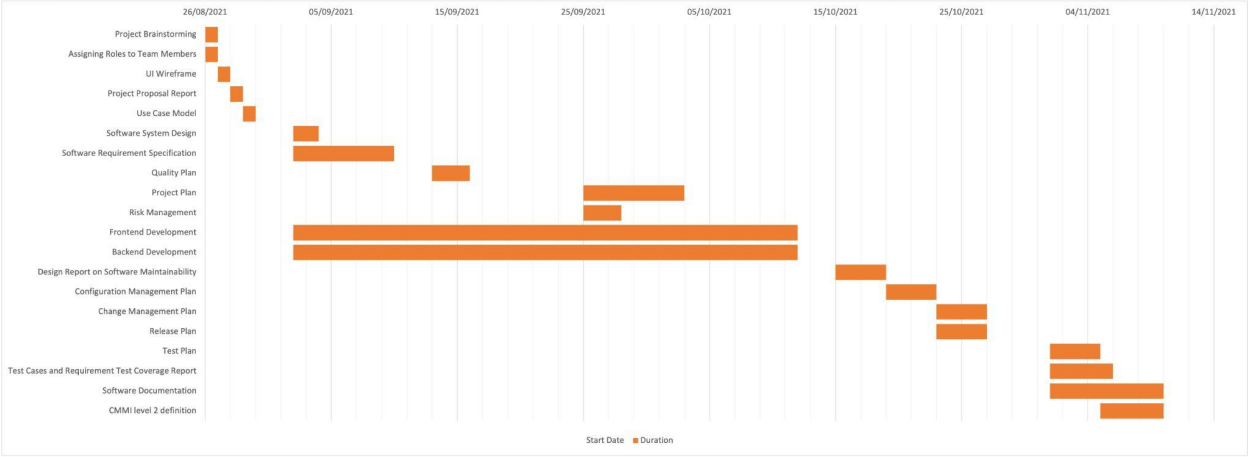
The QMePls project team intends to use the SCRUM Agile software development methodology throughout the project's lifecycle. SCRUM employs an iterative, incremental approach to optimize predictability and to control risk. SCRUM engages groups of people who collectively have all the skills and expertise to do the work and share or acquire such skills as needed.

SCRUM's entire development process consists of several short iteration cycles known as "Sprints", each sprint may be 2 to 4 weeks long, depending on the SCRUM team's needs. A "product backlog" is kept to manage the product's needs and is sorted according to the priority of business values of the product. In each sprint, the SCRUM team would select the highest priority requirements from the product backlog and send it to the sprint backlog during the sprint planning process. These selected product backlog items (PBIs) are analyzed, discussed and estimated at the sprint planning meeting to get a list of tasks known as the sprint backlog to be included in the next sprint. Once the SCRUM team completes the selected tasks in the sprint backlog list, the current sprint ends and proceeds on to the next sprint iteration cycle.

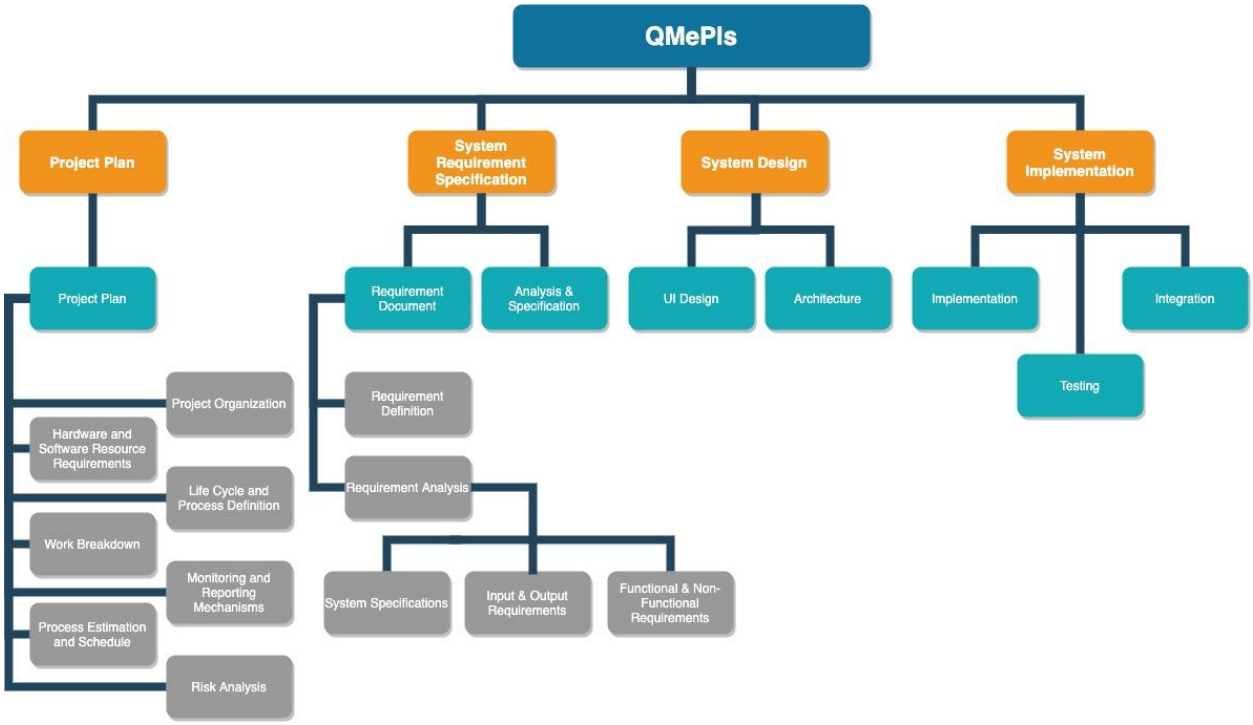
SCRUM sprints allow for a more dynamic and fluid lifecycle should any changes or additions be made to the requirements. These changes can be reflected quickly in the product backlog surrounding the sprints. This fluidity and the rapid nature of agile development methodologies will allow the QMePls team to quickly adapt, develop and deliver iteratively in a rapid manner while keeping up with changes in business and stakeholder requirements.

4. Schedule

4.1 Activity Dependencies and Schedule



4.2 Work Breakdown Schedule



4.3 Work Packages

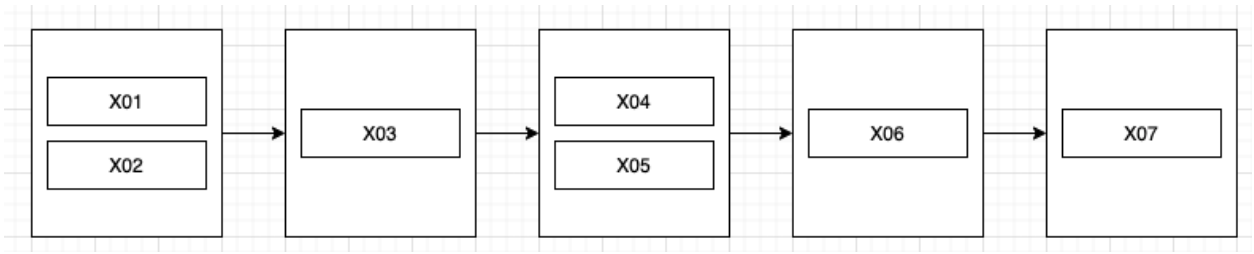
The QMePls project can be broken down to several important phases of the software development life cycle, which includes the following:

1. Project Plan
2. System Requirement Specification
3. Software and Technical Architecture
4. User Interface Design
5. Coding & Unit Testing
6. Management Plan
7. Integration Testing & Quality Assurance

4.4 Activity Dependencies

Work Package #	Description	Duration	Dependencies
X01	Project Plan	8 days	-
X02	System Requirement Specification	8 days	-
X03	Software and Technical Architecture	7 days	X01, X02
X04	User Interface Design	30 days	X03
X05	Coding & Unit Testing	30 days	X03
X06	Management Plan	4 days	X04, X05
X07	Integration Testing & Quality Assurance	14 days	X06

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



4.5 Work Package Details

Project	QMePls Clinic Queue Management System
Work Package	X01— Project Plan (1 of 7)
Assigned To	Seow Jing Hng Aloysius, Jacob Law Zhen, Jolene Tan, Soh Qian Yi, Samuel Lee Si En, Zeta Chua Hui Shi
Effort	8PD
Start Date	25/09/2021
Purpose	To determine and document the draft plan of the project, to be refined for later work packages
Inputs	None
Activities	This work package includes providing a brief overview of the project, its objectives, a set of processes and work packages, cost estimation and working schedules throughout the software development cycle
Outputs	A written document of the Project Plan.

Project	QMePls Clinic Queue Management System
Work Package	X02— System Requirement Specification (2 of 7)
Assigned To	Seow Jing Hng Aloysius, Jacob Law Zhen, Jolene Tan, Soh Qian Yi, Samuel Lee Si En, Zeta Chua Hui Shi
Effort	8PD
Start Date	02/09/2021
Purpose	To establish and determine system requirements of project
Inputs	Requirements
Activities	Acquire and analyse requirements. Provide features and functions that need to be implemented and specify requirements.
Outputs	A written document of the system requirement specification.

Project	QMePls Clinic Queue Management System
Work Package	X03— Software and Technical Architecture (3 of 7)
Assigned To	Seow Jing Hng Aloysius, Jacob Law Zhen, Jolene Tan, Soh Qian Yi, Samuel Lee Si En, Zeta Chua Hui Shi
Effort	7PD
Start Date	10/09/2021
Purpose	To build & design the technical architecture
Inputs	Project Plan Work Packages (X01 and X02).
Activities	Defining the architecture of the software system and identifying the various components and how they are inter-related to and interactive with each other. Deciding on the software and hardware infrastructures Design topics including maintainability, portability, and reusability will be addressed here as well.
Outputs	Software and Technical Architecture

Project	QMePls Clinic Queue Management System
Work Package	X04— User Interface Design (4 of 8)
Assigned To	Zeta Chua Hui Shi, Soh Qian Yi
Effort	30PD
Start Date	18/09/2021
Purpose	To design and develop the user interface between the system and the users, to make it easy use (i.e. user friendly)
Inputs	Project Plan Work Packages (X03)
Activities	To get the user information, user request, display the dialog between system and user, display the result of request
Outputs	User Interface

Project	QMePls Clinic Queue Management System
Work Package	X05— Coding & Unit Testing (5 of 7)
Assigned To	Seow Jing Hng Aloysius, Jacob Law Zhen, Jolene Tan, Samuel Lee Si En
Effort	30PD
Start Date	18/09/2021
Purpose	To design and implement the system as per the systems requirements specification and other associated documents, design the backend service APIs & features, save and provide data. This work package includes such additional activities as preliminary unit testing.
Inputs	Project Plan Work Packages (X03)
Activities	Programmers will implement the modules according to the design specifications noted in the relevant documents.
Outputs	Back-end services & features, as well as source code

Project	QMePls Clinic Queue Management System
Work Package	X06— Management Plan (6 of 7)
Assigned To	Seow Jing Hng Aloysius, Jacob Law Zhen, Jolene Tan, Soh Qian Yi, Samuel Lee Si En, Zeta Chua Hui Shi
Effort	4PD
Start Date	23/10/2021
Purpose	Plan & document the process management
Inputs	Project Plan Work Package (X04 and X05).
Activities	To check if software development is progressing as planned and meeting expectations. Feedback is utilized to keep management plans up to date.
Outputs	Management Plan

Project	QMePls Clinic Queue Management System
Work Package	X07— Testing & Quality Assurance (7 of 7)
Assigned To	Seow Jing Hng Aloysius, Jacob Law Zhen, Jolene Tan, Soh Qian Yi, Samuel Lee Si En, Zeta Chua Hui Shi
Effort	14PD
Start Date	29/10/2021
Purpose	To identify any errors within each component of the system as well as the system as a whole, and to ensure that the integration and implementation of the systems does not create any errors. This is done by conducting unit testing for individual components of the system, followed by an system/acceptance test on the entire system. Errors are to be fixed immediately upon identification. The test plan is documented in the Test Plan Report.
Inputs	Project Plan Work Package X06.
Activities	Test case scenarios will be brainstormed to identify any possible areas for error occurrence. Unit testing is then conducted on the individual test scenarios. After this is done, system testing will be conducted at the end of project implementation.
Outputs	Integrated software and test report.

5. Project Estimates

5.1 Unadjusted Function Points

QMePls supports the following proposed functions:

Patient:

- Login/User Authentication
- Book Appointment System
- Queue Management System
- Symptom Selector
- Notification System
- Clinic Map

Clinic:

- Login/User Authentication
- Queue Management System

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

Inputs:

Item	**Complexity**
Registration (Email, Password, Name)	Low
Login (Email, Password, Domain)	Low
List of Symptoms (Symptom Search, Selected Symptom)	High
Clinic Map (Selected Clinic)	Medium
Book Appointment (Select Clinic, Join Queue)	Medium
Manage Queue (Next Patient, Remove Patient, Skip Patient, Add Walk-In)	Medium

Outputs:

Item	**Complexity**
Displaying successfully registered dialog	Low
Displaying list of symptoms	High
Display selected symptoms	Medium
Displaying notification	Medium
Display queue status	High
Display number of people queuing in front of the user	High
Display confirm appointment booking popup	Low

Inquiries:

Item	**Complexity**
Filter only logged in patient's report card	Medium
Filter only logged in clinic's queue	Medium
Filter only Singapore shown on map	Low

Logical Files:

Item	**Complexity**
Registration and login access control	Low
Handling adding and removing patient from queue	High
Handling addition and deletion of symptoms	High
Handling return nearest 3 clinics	Medium
Handling cancellation of appointment	Medium

Interfaces:

Item	**Complexity**
Firebase	Low
Google Maps API	Medium
Algolia Fuzzy Search API	High

Calculation of Unadjusted Function Points:

Characteristic	**Low Complexity**		**Medium Complexity**		**High Complexity**	
# Inputs	2	x3	3	×4	1	×6
# Outputs	2	×4	2	×5	3	×7
Inquiries	0	×3	2	×4	1	×6
Logical Files	1	×7	2	×10	2	×15
Interfaces	1	×5	1	×7	1	×10
Unadjusted Function Points	6		57		73	
Total = L+M=H	6 + 57 + 73 = 136					

5.1.1 Adjusted Function Points

The measure of Adjusted Function Points includes 14 identified factors that affect the complexity of the code and is scored from 1 to 5 depending on its influence. The following formulas are used for calculation of Adjusted Function Points:

Total score = sum of influence factors
Influence multiplier = (Total score) x 0.01 + 0.65
Total Adjusted Function Points = (Unadjusted total) x (Influence multiplier)

Scoring:

0	1	2	3	4	5
No influence	Insignificant influence	Moderate influence	Average influence	Significant influence	Strong influence

Influence Factors	Score (0-5)	Detail
Data Communications	3	Application is more than a front-end, and supports more than one type of teleprocessing communications protocol.
Distributed Functions	2	Distributed processing and data transfer are online and in both directions.
Performance	4	Application must be smooth enough for the user.
Heavily Used	1	Concurrent users should be able to use it without issues.
On-line Data Entry	5	About 50% of transactions are interactive data entry.
Transaction Rate	2	Daily peak transactions are anticipated.
End-user Efficiency	4	Three to four of the efficiency designs are included.
On-line Data Update	3	Database requires constant backup.
Complex Processing	3	Handling adding and removing patients from the queue.
Reusability	2	The application was specifically packaged and/or documented to ease re-use.
Installation Ease	1	Users can use any Android mobile device to access the application.
Operational Ease	1	Backup is required, but otherwise no operator intervention is needed.
Multiple Sites	0	User requirements do not require needs for multiple instances of the application.
Facilitate Change	0	Changes are unlikely to occur.

Total Score	31
Influence Multiplier = Total score × 0.01 + 0.65 = 31 × 0.01 + 0.65 = 0.96	
Adjusted FP = 136 × 0.96 = 130.56	

5.1.2 Lines of Code

According to Capers Jones and Quantitative Software Management (QSM) statistics, each function point requires on average 53 lines of code if the application is implemented using Java.

Therefore, we have:
Lines of Code: 130.56 FP × 53 LOC/FP = 6919.68 LOC

Note that with each 3rd party code library used, the number of LOC required per function point may be reduced significantly. However, we will not take that into account as those are difficult to estimate, hence, the above LOC metrics are used as the worst case required per function point.

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.
(62 LOC/PD is the Canada average in 1997)

- Working days include 5 days in a week
- Effort = Size / Production Rate = (6919.68 LOC) / (62 LOC/PD) = 111.61 PD
- Duration = $3 \times (111.61)^{1/3}$ = 14.44 Days
- Initial schedule = 14.44 Days/ 5 Days a week = 2.89 Days
- Team Size = Effort / Duration = 111.61 / 14.44 = 7.73 P= 8 Persons
- Working hours include 8 hours in a working day.
- Total person-hours (PH) = 111.61 × 8 = 892.88 PH

5.2.1 Distribution of Effort

	Work Package	**Distribution**	**Estimates**
Preliminary Design 18%	Project Plan	9%	80.36
	System Requirement Specification	9%	80.36
Detailed Design 25%	Software and Technical Architecture	11%	98.22
	User Interface Design	7%	62.5
	Data Modeling	7%	62.5
Code & Testing 26%	Management Plan	21%	187.50
	Online Documentation	5%	44.64
Integration & Test 31%	Integration & Quality Assurance	31%	276.79
	Extrapolated total effort		892.87
	2% for project management		17.86
	3% for contingency		26.79
	Total effort		937.52

5.3 Cost Estimates

Category	**Item**	**Supplier**	**Quantity**	**Unit Price**	**Total**
Personnel	Project Manager	N.A	1	$30,000.00	$30,000.00
	Project Team Members		5	$3,000.00	$15,000.00
Equipment	Computers	Dell	6	$1,500.00	$9,000.00
Technologies	Google Maps Services	Google	-	$350.00 per month	$700.00 (2 months)
	Google Play Services	Google	1	$33.65	$33.65
Utility Costs	Office Rental	NTU	1	$6,000.00 per month	$6,000.00
	Transportation	Comfort Delgro	1	$1,000.00	$1,000.00
Total					**$61,733.65**

6. Product Checklist

The plan is that the items listed below will be delivered on the stated deadlines.

Deliverables	**Estimated Deadline**
Project Proposal	8 September 2021
System Requirement Specification	22 September 2021
Quality Plan	22 September 2021
Project Plan	13 October 2021
Risk Management	13 October 2021
Software Prototype	13 October 2021
Design Report on Software Maintainability	24 October 2021
Configuration Management Plan	25 October 2021
Change Management Plan	27 October 2021
Release Plan	27 October 2021
Test Plan	8 November 2021
Test Cases and Requirement Test Coverage Report	9 November 2021
Software Documentation	10 November 2021
CMMI level 2 definition	10 November 2021

7. Best Practice Checklist

In this section, a list of best practices will be devised and listed according to the larger Software Engineering communities and our own team's configuration. These best practices will help in maintaining quality in the QMePls project. The list will include project management and then software engineering best practices.

Practice	10
Document what we do; all documentation must be in a standardized format. ● IEEE SQA 730-2014 ● IEEE 829 Test Plan	
Pay attention to requirements, check for ambiguity, completeness, accuracy, and consistency. The requirement documentation must contain a complete functional specification.	
Keep it simple. Complexity management is one of the major challenges. Strive to: • Minimize interfaces between modules, procedures and data. • Minimize interfaces between people, otherwise exponential communication cost • Avoid fancy product functions, design as long as the functionality meets the customer requirements	
Require Visibility. We must see what we build otherwise we can measure the progress and take management action. ● Task management software - Trello used to allow team to keep track of each other's progress ● Weekly meeting on Microsoft Teams + Bi-weekly physical meetings to ensure communication between team members ● Version control software - Github used so that code is always available for review by team	
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 ● Use a configuration management system and make processes ● Required maintenance	
Don't underestimate. We must be careful to obtain accurate estimates for: time, effort, overhead, meeting time, and especially effort on integration, testing, documentation and maintenance.	
Usage and adherence to naming conventions to allow for swift and correct identifications and reading of code. ● Oracle Naming Conventions for Java	
Code reviews are a much more efficient method to find software defects. Plan and manage code reviews between team members.	

Ensure a roll back strategy with the version control system for code base and databases. Always ensure that there is a way to roll back to a previous working version.

Software testing will use black box testing. It will involve unit, system and acceptance testing.

8. Risk Management

Besides the general risk management plan, the following risks have been identified for the QMePls project.

1. **Fully Loaded Database**
   Impact severity: Medium
   Probability of occurrence: Medium
   Consequences:
   - Slow fetching times
   - Inability to add new Patient Users/Clinics

   Risk Reduction:
   - Set warning notifications when server capacity is about 80% loaded
   - Subscribe and acquire more server data space from Google Cloud Services

2. **Malicious attacks on server**
   Impact severity: High
   Probability of occurrence: Medium
   Consequences:
   - Disclosure of confidential information

   Risk Reduction:
   - Administrative credentials to the database server should only be held by Project Manager
   - ‘Least Privilege' Access
   - Networks between server and client to be encrypted

3. **Loss of server data**

   Impact severity: High

   Probability of occurrence: Low

   Consequences:

   - Loss of confidential information
   - Loss of functionality of system

   Risk Reduction:

   - Engage in third party cloud service provider to store a separate copy of server data

4. **Lack of awareness of project objectives**

   Impact severity: Medium

   Probability of occurrence: Medium

   Consequences:

   - Project does not meet business requirements
   - Miscommunication and confusion within the team

   Risk Reduction:

   - Weekly reporting of tasks and responsibilities
   - Project Manager to check up on individual teammate's progress every week

5. **Mid-project changes in objectives/management**

   Impact severity: Medium

   Probability of occurrence: Medium

   Consequences:

   - Waste of time and effort
   - Schedule of project must be revised

   Risk Reduction:

   - Review and validate requirements during weekly meetings
   - Follow change management logs in the event of a change

6. **Insufficient Budget**

   Impact severity: High

   Probability of occurrence: Low

   Consequences:

   - Inability to continue with project
   - Cut costs from other areas

   Risk Reduction:

   - Acceptance

7. **Time required to develop the project is underestimated**

   Impact severity: Medium

   Probability of occurrence: Medium

   Consequences:

   - Loss of Client's trust and support
   - Further costs incurred

   Risk Reduction:

   - Weekly meetings to include task fulfilment based on project timeline
   - Re-assess project timeline at an early stage if really needed

8. **Legal obligations**

   Impact severity: High

   Probability of occurrence: Low

   Consequences:

   - Project has to be changed to meet regulations
   - Schedule of project must be revised
   - Cost estimate must be revised

   Risk Reduction:

   - Ensure project abides by industrial standards
   - Identify any potential grey areas in terms of legality

9. **Absence of personnel holding critical roles**

   Impact severity: High

   Probability of occurrence: Low

   Consequences:

   - Specific component of project put on hold
   - Delay project schedule

   Risk Reduction:

   - Weekly updating of tasks by different roles in the team
   - Ensure every single member is up to task based on project timeline

10. **Poor team communications and teamwork**
    Impact severity: Medium
    Probability of occurrence: Medium
    Consequences:
    - Delay Project schedule
    - Failure to meet project requirements

    Risk Reduction:
    - Team bonding activities before/during project timeline to build cohesiveness
    - Buddy system to check on another team member

9. Quality Assurance

The project 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 will be provided in the Test Plan.
In addition, the QMePls will utilise the following testing methodologies:

- Black Box Testing
    - Involves testing based functionalities/components of the system without knowing the code implementation.

- System/Background Testing
    - Involves testing of all the components combined into one system to ensure the cohesiveness of the code implementations.

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

- System Function will be tested to ensure that all the functional requirements are met and that all the software flaws are eliminated

- Algorithmic Function will be tested to ensure that the heuristic aspect such the Queue Management System is provided to Users reliably.

10. Monitoring & Control

Monitoring and Control are processes needed to track, review and regulate the progress and performance of a project. Some of the most important are:

Control Risks:
Risks can potentially impact your project's timeline, performance or budget. Early identification of major risks to the project allows for placement of preventative measures before problems can develop. Major risks and measures taken to avoid them have been identified in the Risk Management document.

Controlling Resource Consumption:
Estimates of the QMePls' 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. With the Project Estimates in the document, we can easily track our progress in the project

Control Quality:
Throughout the project, we must ensure each project deliverable is to the standard defined in the project management plan. Members of the project must meet weekly, either physically or remotely, review the progress of all project tasks

Control Schedule:
Through the Product Checklist, we can see the estimated timeline for the project. We must ensure that the project work is performed according to schedule, and that project deadlines are met. Project subcomponents are included in the Work Breakdown Structure.