Department of Information and Communication Technology
Faculty of Technology
University of Ruhuna

ICT 3262
Software Verification and Quality Assurance
Assessment

Submitted to: Ms. R. S. Wickrama Arachchi

Submitted by:

TG/2019/491 - P.L.N.N. Dilshan
TG/2019/502 - M.M.J. Hasangi
TG/2019/513 - E.A.P. Lakshitha
TG/2019/516 - R.M.U.H.P. Rathnayake
TG/2019/525 - J.L. Rasanjana
TG/2019/527 - S.D. Madarasingha

TEST PLAN FOR THE MCMS WEB AND MOBILE APPLICATION

Abstract

The abstract outlines the development of a comprehensive Medical Center Management System (MCMS) for K.G.N. Medi House in Galle, spearheaded by Dr. Nilantha. Motivated by the need to replace an outdated POS system, the proposed solution combines a web interface (React, Node JS) and a mobile application (Flutter). The web interface caters to detailed information and patient scheduling, with the doctor in the admin role. Simultaneously, the mobile app provides quick access to essential data, stock status, notifications, and reports, with the assistant managing patient scheduling efficiently. This integrated system aims to enhance operational efficiency and provide a seamless experience for both medical staff and patients.

VERSION HISTORY

Registration number	Write by	Approval Date	Approved By	Submitted Date	Outline
TG/2019/491	P.L.N.N. Dilshan	2023.01.09	Ms. R. S. Wickrama Arachchi	2023.09.11	Test Deliveries
TG/2019/502	M.M.J. Hasangi	2023.01.09	Ms. R. S. Wickrama Arachchi	2023.09.11	Test Strategy
TG/2019/513	E.A.P. Lakshita	2023.01.09	Ms. R. S. Wickrama Arachchi	2023.09.11	Schedule and Estimation
TG/2019/516	R.M.U.H.P. Ratnayake	2023.01.09	Ms. R. S. Wickrama Arachchi	2023.09.11	Test Objective, Test Criteria, Resource Planning
TG/2019/525	J.L. Rasanjana	2023.01.09	Ms. R. S. Wickrama Arachchi	2023.09.11	Test Environment
TG/2019/527	S.D. Madarasingha	2023.01.09	Ms. R. S. Wickrama Arachchi	2023.09.11	Test Strategy

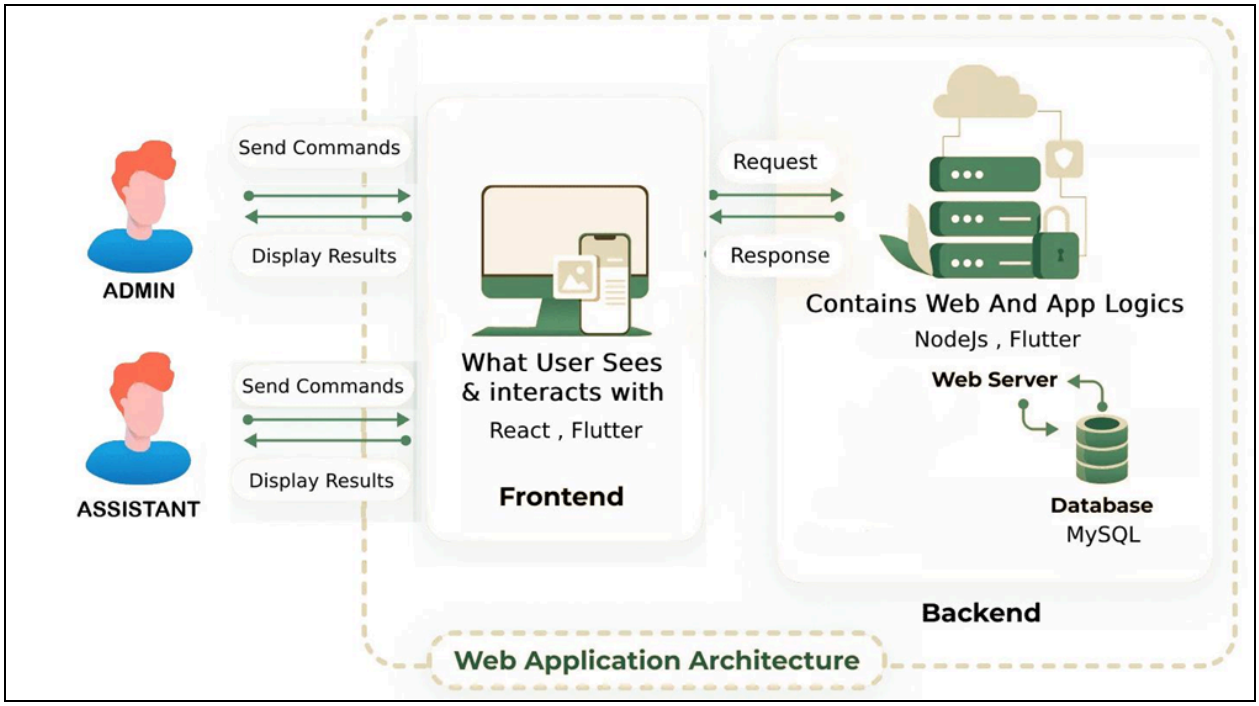
1 Introduction

1.1 Problem Specification

The client is currently using a legacy java application built around 2010. The software was developed for inventory management and billing using java and an access database. The current system is incompatible with newer versions of windows and very slow to respond. There are critical vulnerabilities in the current system. The current system is built for work offline, and no backup system is built to back up the database. Doctor channeling, patient management and attendance marking of pharmacy assistance are done manually. The current system only generates a daily report of drug usage; there are no options to view monthly or quarterly usage checks. No warnings are shown for short, expired products, and notify the customer in the current system.

1.2 Solution Outline

A web-based system is being developed as the solution, with the front end being created using React and the back end being created using Node.js. In addition, for the convenience of the customer, a mobile application is being developed using Flutter (Dart). In addition to the current limitations, our new web-based system will have a number of new features including doctor channeling, patient management, and attendance marking of pharmacy assistance which will be systematic. Our system will also have the ability to generate monthly and quarterly usage reports, which will give the client a better understanding of their inventory management. The system will also have built-in warnings for short-expired products and notifications to the customer, ensuring they are aware of any potential issues with their inventory. Additionally, the web-based system will be compatible with newer versions of windows and will be much faster and responsive. The new system will also have a built-in backup system to protect against data loss. The mobile application will also be an added advantage to the customer as they can easily monitor their stocks and inventory on the go.



2. Test Strategy

2.1 Scope of Testing

The scope of a test defines what areas of a customer's product are supposed to get tested, what functionalities to focus on, what bug types the customer is interested in, and what areas or features should not be tested by any means.

2.1.1 Features to be tested

Module Name	Applicable Roles	Description
Appointment Management	Patient Receptionist Medical Staff	Patients: Patients can book, reschedule, or cancel appointments with specific doctors or departments. Medical Staff: Medical staff can view their schedules, confirm, or reject appointments, and set their availability timings. Receptionist: Receptionists can manage the appointment calendar, assist patients in booking appointments, and handle schedule changes.

Billing and Payments	Patient Receptionist Administrator	Patient: Patients can view their medical bills, make payments online, and download invoices for insurance purposes. Accounting Staff: Accounting staff generates invoices, processes payments, and maintains billing records for patients and insurance companies. Administrator: Administrators have access to financial reports, monitor transactions, and ensure the smooth operation of billing processes.
Stock Management	Inventory Manager Medical Staff	Inventory Manager: Inventory managers oversee stock levels, order medical supplies, manage inventory databases, and ensure that the medical center has adequate supplies. Medical Staff: Medical staff can request necessary medical supplies and equipment through the system, specifying the quantity and urgency. They can also view stock availability for essential items. Medical Staff will also get informed when the stocks run out via the mobile application.
Report Management	Medical Staff Administrator	Medical Staff: Medical staff can generate patient reports, view test results, and access medical history reports to aid in diagnosis and treatment decisions. Administrator: Administrators have access to various reports such as patient demographics, appointment schedules, billing reports, and other statistical data to assess the medical center's performance and make informed decisions.

2.1.2 Features not to be tested

The following features will not be tested since they are not included in the software requirements specification.

- User Interfaces
- Hardware
- Database logic
- Website Performance
- Communication Interfaces
- App

2.2 Test Type

- Integration Testing (Individual software modules are combined and tested as a group)
- System Testing: Conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements.
- API testing: Test all the APIs created for the software under tested.

2.3 Risk and Issues

Risk	Mitigation
Medical errors, infections, misdiagnosis, and other issues can compromise patient safety.	Implement robust training programs for healthcare staff. Foster a culture of open communication where staff can report errors without fear of reprisal.
Patient data breaches and unauthorized access to medical records.	Employ encryption and strict access controls to protect patient information. Regularly update security protocols and conduct security audits.
Revenue loss due to billing errors, insurance claim denials, or changes in healthcare policies.	Invest in robust billing and revenue cycle management systems. Stay updated with healthcare regulations and

	insurance policies.
Negative publicity due to patient dissatisfaction or public perception issues.	Implement a robust patient feedback system to address concerns promptly. Engage in community outreach programs to improve the medical center's image. Have a crisis communication plan in place to address negative publicity effectively.

2.4 Test Logistics

2.4.1 Who will test

All the developers who is working on the group project will participate for the testing.

2.4.2 When will the test occurs

The testing will start by the developers when all the following inputs are ready.

- During the development of a component
- After a component is finished
- Software is available for testing
- Test Specification is created

3. Test Objective

The main objective of this testing phase is to verify all the components of the Medical Center Management System(MCMS) are functioning well. All the operations such as appointment handling, billing, report generation etc. should be working normally in a real business environment.

4. Test Criteria

4.1 Suspension Criteria

If the team members report that there are 40% of test cases failed, suspend testing until the development team fixes all the failed cases.

4.2 Exit Criteria

Specifies the criteria that denote a successful completion of a test phase

- Run rate is mandatory to be 100% unless a clear reason is given.
- Pass rate is 80%, achieving the pass rate is mandatory.

5. Resource Planning

5.1 System Resource

No.	Resources	Description
1.	Computer	There should be enough requirements to run all the development tools and testing software in a computer. There should be at least 3 computers to execute the texting.
2.	Network	There should be a good connection of at least 5Mb/s speed with a lan connection
3.	Server	A server to host the website to test and deploy the website for the real business world
4.	Test tool	A test tool that supports Jira testing to track down and ensure all the components are working correctly

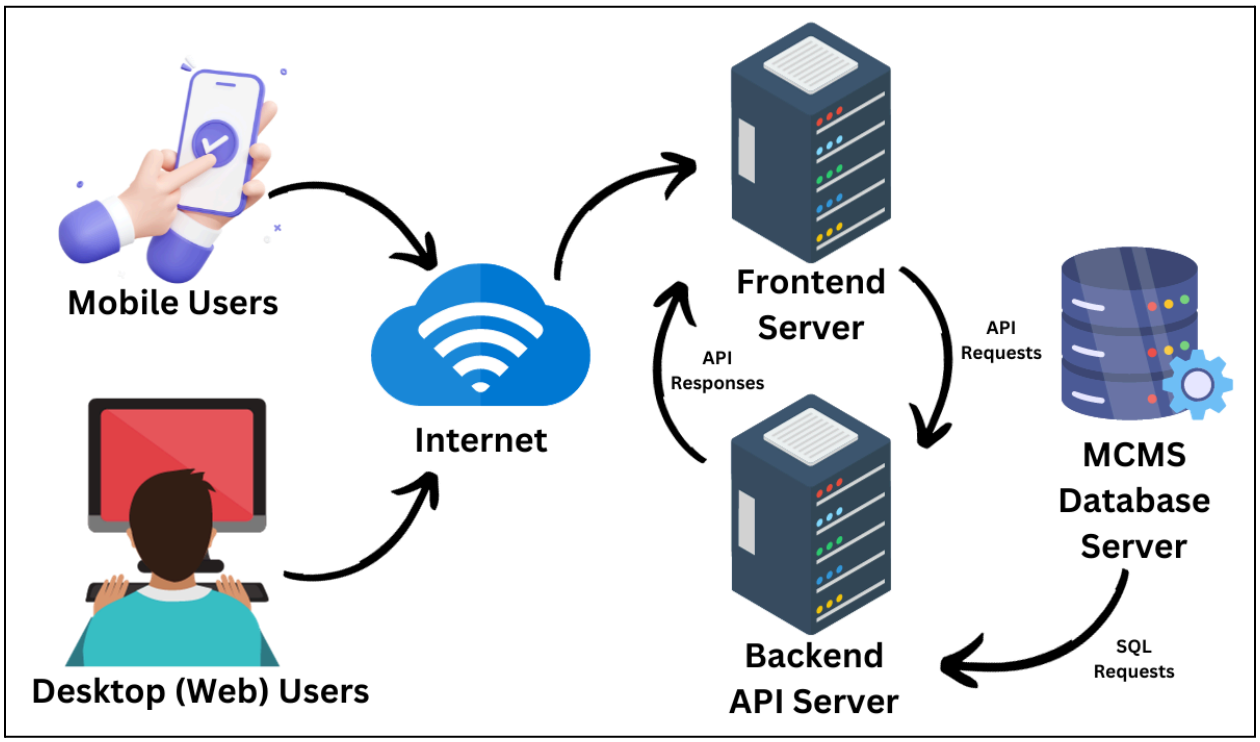
5.2 Human Resource.

No.	Resources	Description
1.	Project Manager	Manage the whole project Define project directions

		Acquire appropriate resources
2.	Developers	Develop components for the system Test the component after it was finished Execute test cases, results, report defects
3.	Supervisor	Direct team members with supervision Give suggestion for the development

6. Test Environment

The Test Environment should be setup as figure below



7. SCHEDULE & ESTIMATION

All project task and estimation

Website

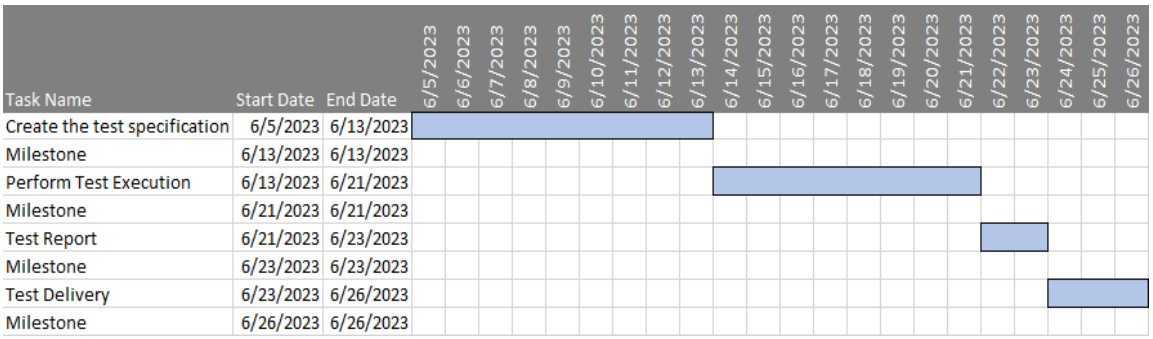
Task	Members	Estimate effort
Create the test specification	Test Designer	40 man-hour
Perform Test Execution	Tester	25 man-hour
Test Report	Tester	5 man-hour
Test Delivery		10 man-hour
Total		80 man-hour

Application

Task	Members	Estimate effort
Create the test specification	Test Designer	35 man-hour
Perform Test Execution	Tester	15 man-hour
Test Report	Tester	3 man-hour
Test Delivery		5 man-hour
Total		58 man-hour

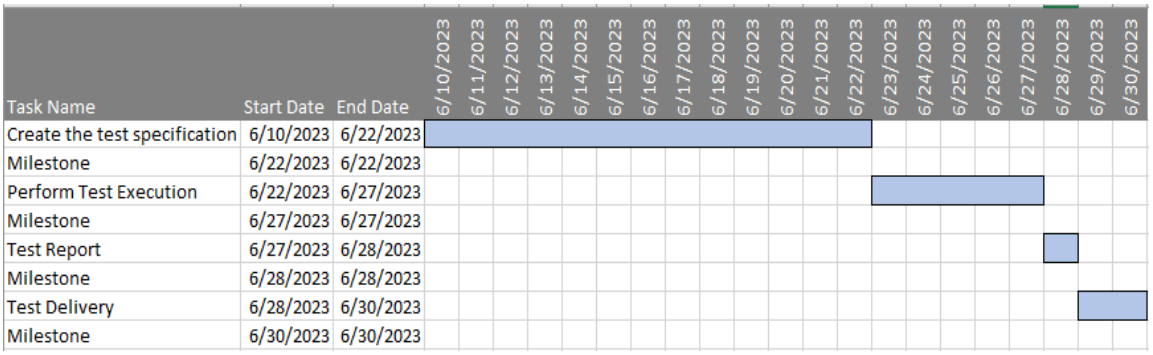
Schedule to complete these tasks

Website



Task Name	Start Date	End Date
Create the test specification	6/5/2023	6/13/2023
Milestone	6/13/2023	6/13/2023
Perform Test Execution	6/13/2023	6/21/2023
Milestone	6/21/2023	6/21/2023
Test Report	6/21/2023	6/23/2023
Milestone	6/23/2023	6/23/2023
Test Delivery	6/23/2023	6/26/2023
Milestone	6/26/2023	6/26/2023

Application



Task Name	Start Date	End Date
Create the test specification	6/10/2023	6/22/2023
Milestone	6/22/2023	6/22/2023
Perform Test Execution	6/22/2023	6/27/2023
Milestone	6/27/2023	6/27/2023
Test Report	6/27/2023	6/28/2023
Milestone	6/28/2023	6/28/2023
Test Delivery	6/28/2023	6/30/2023
Milestone	6/30/2023	6/30/2023

8. TEST DELIVERABLES

Test deliverables are provided as below

8.1 Before testing phase

- Test plans document.
    - Scope: Frontend, backend, mobile app, APIs.
    - Testing types: Unit, integration, system, UAT.
    - Estimated timelines: 4 weeks for testing.
    - Tools: Jira, Selenium, Postman.
- Test cases documents.
    - 50 detailed test cases covering appointment management, billing, reporting.
    - Test cases have preconditions, test data, steps, expected result.
- Test Design specifications
    - Test environment details:
        - OS: Windows 10, Windows 11, Ubuntu 20, Ubuntu 22.
        - Browsers: Chrome, Firefox, Edge.
        - Devices: Pixel 4, Apple iPhone X.
    - Test data strategy: Mix of production clone and synthetic data.
    - Tools selected: Jira for defects, Selenium for automation.

8.2 During the testing

- Test Tool - Simulators:
    - Jira tool implemented for tracking defects and test cases.
    - APIs simulated using Postman.
- Test Data: Test data created covering different scenarios and edge cases.
- Test Trace-ability Matrix: Traceability matrix linking all defined test cases to requirements.
- Error logs and execution logs: Defects logged and tracked actively in Jira.

8.3 After the testing cycles is over

- Test Results/reports:
    - Executed 45 test cases, Passed: 40, Failed: 5
    - Open Defects: 5 (2 critical, 3 minor)
    - Defect root cause analysis indicated frontend bugs
- Defect Report - Installation/ Test procedures guidelines.
    - Deployment guide created with setup instructions.
- Release notes:

- Release notes listing new features, fixes, known issues.