Software Test Plan for ... (Group #1)

1. Introduction

1.1. Purpose

This document will outline our software testing plan and approach to ensure functionality, integration, and performance for our Hospital Appointment Booking Website.

1.2. Scope

* The scope encompasses the essential functions necessary for the website to attain functional viability and serve as a viable substitute for the hospital's existing appointment booking systems.

1.2.1 In Scope

* Calendar
* reservation/cancelation
* Staff accounts
* Regular user account
* Reminders

1.2.2 Out of Scope

* Features that will not be tested because they are not part of the initial scope of the project
* Website on mobile

1.3. References

* Product Proposal
* Project Roadmap
* System Requirements Document
* Software Specification Documents

2. Test Strategy

2.1. Objective

* The primary objective of these assessments is to ascertain the absence of runtime errors within our website while rigorously validating their functionality through a multi-tiered testing approach, encompassing unit, component, and system testing. Each discrete section of the website undergoes scrutiny to ensure its adherence to its designated functionality. When these individual components are seamlessly integrated through comprehensive component testing, we strive to demonstrate the harmonious operation of the entire system, culminating in a cohesive and operational website.

2.2. Approach

* Traditional manual testing will be used first to test the website functions in parts and as a whole. We will also be using automation programs such as Selenium to test website and browser functionality. But this will come after we receive acceptable results from manual testing

2.3. Tools

* Selenium for automated website and browser testing.
* JMeter for load and performance testing on software
* Apache Jmeter for load, stress, and functional database testing
* SqlMap to test database security vulnerabilities
* Google Test for unit testing
* Sonarqube for automated code quality testing
* Manual bottom-up integration testing

2.4. Environments

* We will develop separate testing environments for Unit, Component, System, and Integration testing to ensure that our Website works at every level of development.

2.5. Entry and Exit Criteria

Entry:

* All development activities for the appointment booking system are completed.
* The test environment is set up and configured.
* Test data, including patient information and appointment scenarios, are prepared and available.
* Test cases and test scripts are created, reviewed, and ready for execution.
* Test resources, including hardware, software, and personnel, are allocated and available.

Exit:

* All test cases and test scripts have been executed, and the results are documented.
* Defects, if any, have been identified, logged, tracked, and their resolutions verified.
* Test summary reports, including test coverage and pass/fail metrics, are generated.
* The system is considered ready for production release based on the test results and stakeholder approval.

3. Scope of Testing

3.1. Functional Testing

3.1.1 Unit Testing:

* Ensure that each unit performs as expected in isolation.
* We will make sure individual functions produce the expected results and look out for any logic errors to fix them.
* We will also test these in component form to see if they interact as intended.

3.1.2 Integrated Unit Testing:

* We will test how components perform when integrated with other components in the system to ensure that everything is integrated correctly.
* We will test how the website functions when integrated with other pieces of technology to see that their performance is satisfactory such as cellphones for notifications

3.1.3 System Integration Testing:

* Test the integration of all system components.
* Verify end-to-end functionality, including appointment booking, patient management, and database interactions.
* Ensure all integrations work as planned (google maps, google calendar, phone notification system)

3.1.4 User Acceptance Testing (UAT):

* Our Product Manager(us) will get a small sample of users(friends) to test the website to make sure all requirements have been met
* Ensure that the system aligns with real-world usage and user expectations.

3.1.5 Regression Testing:

* Re-run selected tests to ensure that new changes have not negatively impacted existing functionality.
* Verify that fixes for previously identified defects do not introduce new issues.

3.1.6 Smoke Testing:

* Conduct initial, shallow tests to check if the critical functionalities of the system are working as expected.
* Performed before more comprehensive testing phases.

3.2 Non-Functional Testing

3.2.1 Coding Standard Testing:

* Check if the code adheres to coding standards and best practices agreed on by the team.
* Ensure code quality and maintainability.
* Automatedly tested with sonarqube

3.2.2 Security Testing:

* Assess the system's security measures to identify vulnerabilities.
* Ensure that patient data is protected and that access controls are in place.
* Automated testing will be done using SqlMaps and SonarQube

3.2.3 Usability Testing:

* Evaluate the user interface for ease of use and user-friendliness.
* Ensure that users can navigate the system comfortably.
* Automated Testing will be done with Google Tests

3.2.4 Performance and Load Testing:

* Test the system's performance under different loads and usage scenarios.
* Assess response times, scalability, and resource utilization.
* Automated testing will be done using Selenium

4. Consideration of Infrastructure

4.1. Server Configuration:

* Ensure that server hardware and software meet system requirements.
* Verify load balancing, redundancy, and failover mechanisms for high availability.

4.2. Database:

* Verify the correctness of the database schema and data integrity.
* Test database performance under load.
* Implement backup and recovery procedures.
* Sqlmaps for security testing

5. Risks or Mitigation Plan

5.1. Risks:

* Identify potential risks, such as data breaches, system downtime, or regulatory non-compliance.
* Assess the impact and likelihood of each risk.
* Loss of important information in data transfers

5.2. Mitigation:

* Regular Database Backups
* Have data encrypted when stored to prevent breaches
* Weekly/monthly server maintenance for performance
* Assign responsible individuals or teams for risk mitigation.
* Develop contingency plans in case mitigation efforts fail.

6. Resourcing

6.1. Team Composition:

* Specify the roles and responsibilities of team members involved in testing, including testers, developers, and subject matter experts.
* Ensure that the necessary skills and expertise are available within the team.
* Each member will take turns in the sprint to be Test Manager so that all members get equal experience.

7. Milestones and Deliverables

7.1 Milestones:

* Define key milestones in the testing process, such as test planning, test execution, and test closure.
* Set specific dates or deadlines for each milestone.

Sprint 1:

* Research Test Tools
* Create Test Plan

Sprint 2:

* Manual regression testing on new stories and features as developed
* Begin Learning Automated Testing Tools listed

Sprint 3-4:

* Continue regression testing on any new functions
* Use google tests to automate unit testing
* Security testing on databases with SqlMaps
* Begin implementing sonarqube for code quality testing

Sprint 5-6:

* Begin integration testing with rough deliverable of project
* Continue regression testing on any new functions
* Use google tests to automate unit testing
* Security testing on databases with SqlMaps
* Continue using sonarqube for code quality testing

(next PI)Sprint 7-8:

* Start using Selenium for Performance and load testing of website
* Continue all previous tests

Sprint 9-10

* User Acceptance Testing
* Begin UAT Fixes
* Continue all previous tests

Sprint 11-12

* UAT Fixes
* Final rounds of all previous testing as we finalize the project
* Final Report

7.2. Deliverables:

* List the documents and reports that will be produced as part of the testing process, such as test plans, test cases, test logs, defect reports, and test summary reports.
* Specify the format and distribution of these deliverables.