Master Test Plan

March 31, 2016

Submitted By:
Alvin
Andi Darmawan
Fakhria Nur Shabrina
Muhammad Farhan Mubarak

Table of Contents

1 Introduction

This document is a high-level overview defining our testing strategy for the Public Accommodation Location Service (PALS) System. Its objective is to communicate project-wide quality standards and procedures. It portrays a snapshot of the project as of the end of the planning phase. This document will address the different standards that will apply to the unit, integration and system testing of the specified system. We will utilize testing criteria under the white box, black box, and system-testing paradigm. This paradigm will include, but is not limited to, the testing criteria, methods, and test cases of the overall design. Throughout the testing process we will be applying the test documentation specifications described in the IEEE Standard 829-1983 for Software Test Documentation.

1.1 Team Interaction

The following describes the level of team interaction necessary to have a successful product.

- The Test Team will work closely with the Development Team to achieve a high quality design and user interface specifications based on customer requirements. The Test Team is responsible for visualizing test cases and raising quality issues and concerns during meetings to address issues early enough in the development cycle.
- The Test Team will work closely with Development Team to determine whether or not the application meets standards for completeness. If an area is not acceptable for testing, the code complete date will be pushed out, giving the developers additional time to stabilize the area.
- Since the application interacts with a back-end system component, the Test Team will need to include a plan for integration testing. Integration testing must be executed successfully prior to system testing.

2 Test Objective

The objective our test plan is to find and report as many bugs as possible to improve the integrity of our program. Although exhaustive testing is not possible, we will exercise a broad range of tests to achieve our goal. We will be testing PALS Application and its Administrator Website utilizing a pre-order traversal format. For the PALS Application, there will be 4 main menus and several feature, that are:

- Find Location By Category (ATM, Gas Station, and Repair Shop).
  - Nearby radius: 300 meters to 1.5 km with increment 300 meters.
  - View all location for each category.
  - View location information.
  - Get direction from current location to place location.
  - Make a phone call (just for repair shop).

- User Input Location required to be confirmed by administrator before it can show in maps).
- Help Menu
- About Menu

And as for the Administrator Website, there will be seven main functions that are:

- Administrator Login
- View Data (ATM, Gas Station, Repair Shop)
- Insert Data (ATM, Gas Station, Repair Shop)
- Update Data (ATM, Gas Station, Repair Shop)
- Delete Data (ATM, Gas Station, Repair Shop)
- Edit User Inputted Data (ATM, Gas Station, Repair Shop)
- Confirm User Inputted Data (ATM, Gas Station, Repair Shop)
- Reject User Inputted Data (ATM, Gas Station, Repair Shop)

Our user interface to utilize these feature and functions is designed to be user-friendly and provide ease of use. The PALS Application will be released in Google Play Store, where as for Administrator Website will be hosted to web hosting. So we would like to ensure that it could be run from a variety of Android devices and browsers.

3 Process Overview

The following represents the overall flow of the testing process:

1. Identify the requirements to be tested. All test cases shall be derived using the current Program Specification.
2. Identify which particular test(s) will be used to test each module.
3. Review the test data and test cases to ensure that the unit has been thoroughly verified and that the test data and test cases are adequate to verify proper operation of the unit.
4. Identify the expected results for each test.
5. Document the test case configuration, test data, and expected results.
6. Perform the test(s).
7. Document the test data, test cases, and test configuration used during the testing process. This information shall be submitted via the Unit/System Test Report (STR).
8. Successful unit testing is required before the unit is eligible for component integration/system testing.
9. Unsuccessful testing requires a Bug Report Form to be generated. This document shall describe the test case, the problem encountered, its possible cause, and the sequence of events that led to the problem. It shall be used as a basis for later technical analysis.
10. Test documents and reports shall be submitted. Any specifications to be reviewed, revised, or updated shall be handled immediately.

4 Testing Process

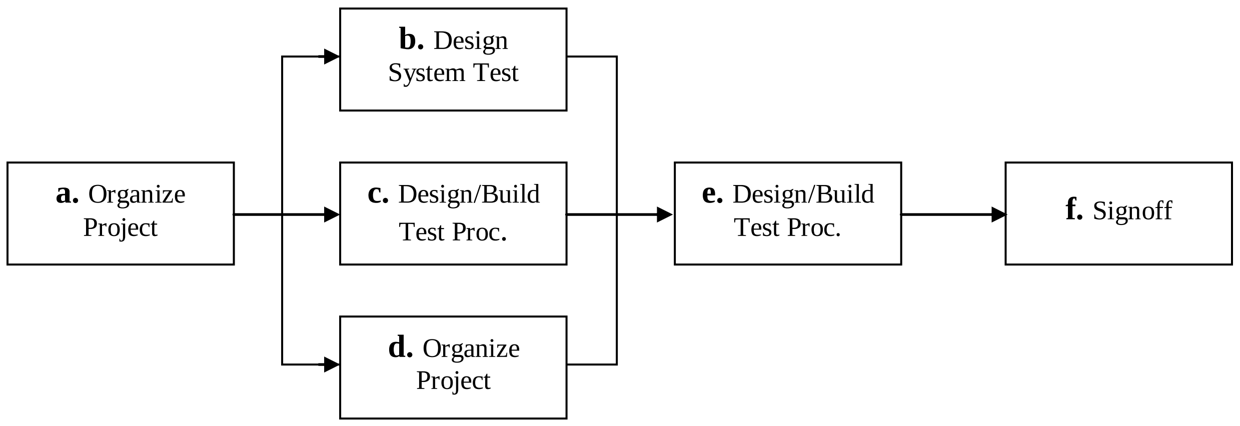

Figure 1: Test Process Flow

The diagram above outlines the Test Process approach that will be followed.

a. **Organize Project** involves creating a System Test Plan, Schedule & Test Approach, and assigning responsibilities.
b. **Design/Build System Test** involves identifying Test Cycles, Test Cases, Entrance & Exit Criteria, Expected Results, etc. In general, test conditions/expected results will be identified by the Test Team in conjunction with the Development Team. The Test Team will then identify Test Cases and the Data required. The Test conditions are derived from the Program Specifications Document.
c. **Design/Build Test Procedures** includes setting up procedures such as Error Management systems and Status reporting.
d. **Build Test Environment** includes requesting/building hardware, software and data set-ups.
e. **Execute System Tests** – The tests identified in the Design/Build Test Procedures will be executed. All results will be documented and Bug Report Forms filled out and given to the Development Team as necessary.
f. **Signoff** - Signoff happens when all pre-defined exit criteria have been achieved.

5 Testing Strategy

The following outlines the types of testing that will be done for unit, integration, and system testing. While it includes what will be tested, the specific use cases that determine how the testing is done will be detailed in the Test Design Document. The template that will be used for designing use cases is shown in Figure 2.

Tested By:	
Test Type	
Test Case Number	
Test Case Name	
Test Case Description	
Item(s) to be tested	
1	
2	
Specifications	
Input	**Expected Output/Result**
Procedural Steps	
1	
2	
3	
4	
5	
6	
7	

Figure 2: Test Case Template

5.1 Unit Testing

Unit Testing is done at the source or code level for language-specific programming errors such as bad syntax, logic errors, or to test particular functions or code modules. The unit test cases shall be designed to test the validity of the programs correctness.

5.1.1 White Box Testing

In white box testing, the UI is bypassed. Inputs and outputs are tested directly at the code level and the results are compared against specifications. This form of testing ignores the function of the program under test and will focus only on its code and the structure of that code. Test case designers shall generate cases that not only cause each condition to take on all possible values at least once, but that cause each such condition to be executed at least once. To ensure this happens, we will be applying Branch Testing.

Each function of the PALS application and Administrator Website is executed independently; therefore, a program flow for each function has been derived from the code.

5.1.1.1 Branch Testing

Using the program flow graph for each function, we will be able to determine all of the branches that will need to tested and will be used to develop the corresponding test cases.

5.1.2 Black Box Testing

Black box testing typically involves running through every possible input to verify that it results in the right outputs using the software as an end-user would. We have decided to perform Equivalence Partitioning and Boundary Value Analysis testing on our application.

5.1.2.1 Equivalence Partitioning

In considering the inputs for our equivalence testing, the following types will be used:

- Legal input values – Test values within boundaries of the specification equivalence classes. This shall be input data the program expects and is programmed to transform into usable values.
- Illegal input values – Test equivalence classes outside the boundaries of the specification. This shall be input data the program may be presented, but that will not produce any meaningful output.

The equivalence partitioning technique is a test case selection technique in which the test designer examines the input space defined for the unit under test and seeks to find sets of input that are, or should be, processed identically. The following table represents our equivalence classes, both valid and invalid.

Input/Output Event	Valid Equivalence Classes	Invalid Equivalence Classes
atm_name	maximum 100 character	> 100 character empty input
atm_address	Text data type that can contains maximum 65535 character	empty input
bank_name	maximum 20 character	> 20 character empty input
gas_station_name	maximum 35 character	> 35 character empty input
gas_station_address	Text data type that can contains maximum 65535 character	empty input
gas_station_company	maximum 12 character	> 12 character empty input
repair_shop_name	maximum 35 character	> 35 character empty input
repair_shop_address	Text data type that can contains maximum 65535 character	empty input
repair_shop_telephone	maximum 13 number character	> 13 number character non-number character
vehicle_brand	maximum 15 character	> 15 character empty input
latitude	Double data type up to 16 digits, which is a 10 digit after the decimal point	> 16 character empty input
longitude	Double data type up to 16 digits, which is a 10 digit after the decimal point	> 16 character empty input
internet connection	Have internet access with decent speed	No internet access Slow internet access
location service	Device location service enabled	Device location service disabled

5.1.2.2 Boundary Value Testing

The selectable range of radius values was set by the development team that is 300 metres to 1.5 kilometers with multiples of 300 meters. However, users can also see all the location of the object if within that range of the radius, location of the object cannot be found. Due to limitation on the scope, the application only stores data objects (ATM, gas station, and repair shop) that are located in Depok and surrounding area. Boundary value testing will be conducted to ensure the relevancy of search results based on the radius selected.

5.2 Integration Testing

There are two applications that will need to be integrated: PALS Android Application and the Administrator Website. Those two applications, once integrated, will form the complete PALS System. The following describes these applications as well as the steps that will need to be taken to achieve complete integration.

Application 1 - PALS Android Application

This application provides a user interface where the user can perform the different actions (functions). This application will be tested separate from the administrator website to check if each interface (e.g. button, spinner, maps, etc) is functioning properly, and in general, to test if the click-event actions are working properly.

Application 2 – Administrator Website

The Administrator Website provides the function that used by administrator to maintain and manage the database. The website is also contains the web services which will be used as connection between PALS Android Application and the database server.

When the PALS Android Application integrated with web services in Administrator Website, we will have a complete PALS System.

5.3 System Testing

The goals of system testing are to detect faults that can only be exposed by testing the entire integrated system or some major part of it. Generally, system testing is mainly concerned with areas such as performance, compatibility, security, validation, load/stress, and configuration sensitivity. But in our case well focus only on function validation and compatibility. And in both cases we will use the black-box method of testing.

5.3.1 *Function Validation Testing*

The integrated “PALS System” will be tested based on the requirements to ensure that we built the right application. In doing this test, we will try to find the errors in the inputs and outputs, that is, we will test each function to ensure that it can run properly. The behavior of each functions are contained in the Software Program Specification.

Function	**Expected Behavior**
Find ATM location near by particular radius (Android Application)	see Software Program Specification
Find Gas Station location near by particular radius (Android Application)	see Software Program Specification
Find Repair Shop location near by particular radius (Android Application)	see Software Program Specification
Get Direction (Android Application)	see Software Program Specification
Make a Phone Call (Android Application)	see Software Program Specification
View Location Information (Android Application)	see Software Program Specification
Input Location (Android Application)	see Software Program Specification
View About (Android Application)	see Software Program Specification
View Help (Android Application)	see Software Program Specification
Login (Web Application)	see Software Program Specification
Add Data (Web Application)	see Software Program Specification
Edit Data (Web Application)	see Software Program Specification
Delete Data (Web Application)	see Software Program Specification
Search Data (Web Application)	see Software Program Specification
Confirm User Inputted Data (Web Application)	see Software Program Specification
Reject User Inputted Data (Web Application)	see Software Program Specification

In addition, we will test:

- The interfaces to ensure they are functioning as desired (i.e. check if each interface is behaving as expected, specifically verifying the appropriate action is associated with each click event).

5.3.2 Compatibility Testing

This test will be conducted to ensure compatibility of PALS android application with various android devices screen size and platforms. It will be done by testing in various android device types. And this will be performed by:

- Check if the PALS application can be run on Android OS 4.2 (Jelly Bean) – Android OS 6.0 (Marshmallow).
- Checking position of layout, button, spinner, maps, etc.
- Check if each interface (e.g. button, spinner, maps, etc) is functioning properly.

6 Entry and Exit Criteria

This section describes the general criteria by which testing commences, temporarily stopped, resumed and completed within each testing phase. Different features/components may have slight variation of their criteria, in which case, those should be mentioned in the feature test plan. The testing phase also maps to the impact level definition when a defect is entered in the bug-tracking phase.

6.1 Unit Testing

Unit Testing is done at the source or code level for language-specific programming errors such as bad syntax, logic errors, or to test particular functions or code modules. The unit test cases shall be designed to test the validity of the programs correctness.

6.1.1 Black Box Phase

Black box testing typically involves running through every possible input to verify that it results in the right outputs using the software as an end-user would. We will use Equivalence Partitioning and Boundary Value Analysis complexity metrics in order to quantifiably determine how many test cases needed to achieve maximum code coverage.

*6.1.1.1 **Black Box Entry Criteria***

The Black Box Entry Criteria will rely on the component specification, and user interface requirements. Things that must be done on entry to the Black Box stage:

- All PALS Android Application functions, Find (ATM, Gas Station, Repair Shop), Add New, Get Direction, Make Phone Call, View Information, View Help, View About, must be coded.
- All Administrator Website function, Login, Add, Edit, Delete, Confirm, Reject, Search, Logout, must be coded.
- The type of Black Box testing Methods will be determined upon entry. We will use Equivalency Partition, and Boundary Value Analysis.
- Equivalency Partition will include, String, Text, and Double data types,.
- Boundary Value Analysis will include, Double data type as radius, and the values will have a boundary value of (0.3 - 1.5) with increment of 0.3.

*6.1.1.2 **Black Box Exit Criteria***

The Black Box Exit Criteria listed below explains what needs to be completed in-order to exit Black Box phase. To exit the Black Box phase 100% success rate must be achieved. Things that must be done upon exiting the Black Box stage:

- The Equivalence Classes will have been created for the valid and invalid input values.
- The Equivalency Partition Method will have generated Test Cases based on the Equivalence Classes of inputs. The invalid input domain values for Equivalence classes will include inputting an empty input data file and inputting character exceed the maximum length.
- Another set of Test Cases will have been created based on Partition Class of internet connection. The Test Cases will test in normal internet speed, slow internet speed, and no internet connection.
- Another set of Test Cases will have been created based on Partition Class of location service. The Test Cases will test with location service enabled and disabled.
- Boundary Value Analysis will have generated Test Cases based on the boundary values of coverage radius (0.3 - 1.5 km).
- All code bugs that are exposed are corrected.

6.1.2 White Box Phase

The White Box criteria apply for purposes of focusing on internal program structure, and discover all internal program errors. Defects will be categorized and the quality of the product will be assessed.

6.1.2.1 *White Box Entry Criteria*

The White Box Entry Criteria will rely on the QA engineers verifying that the major features work alone but not necessarily in combination; exception handling will not be implemented. The design and human interface are stable. Things that must be done on entry to the White Box stage:

- All PALS Android Application functions, Find (ATM, Gas Station, Repair Shop), Add New, Get Direction, Make Phone Call, View Information, View Help, View About, must be coded.
- All Administrator Website function, Login, Add, Edit, Delete, Confirm, Reject, Search, Logout, must be coded.
- The type of White Box testing Methods will be determined upon entry. We will use Basis Path Testing and Function Validation testing on all PALS system properties.
- Black Box Testing should be in its late stages.

After the White Box criteria have been met, the product enters the White Box stage. During White Box stage Development Engineering's emphasis is on refining the product and fixing defects. Information Design's emphasis is on developing product user documentation.

6.1.2.2 *White Box Exit Criteria*

The PALS System in the White Box stage should have a generally stable feel to it. White Box testing continues until the Black Box or next milestone criteria are met. To exit the White Box phase 100% success rate must be achieved. The following describes the state of the product upon exit from the White Box Stage:

- All PALS Android Application functions, Find (ATM, Gas Station, Repair Shop), Add New, Get Direction, Make Phone Call, View Information, View Help, and View About, are implemented, operational and tested.
- All Administrator Website function, Login, Add, Edit, Delete, Confirm, Reject, Search, Logout, are implemented, operational and tested.
- All Branch Testing test cases will be generated. The test cases will be generated from the Control Flow diagrams of all functions.
- Both PALS Android application and Administrator Website graphical interface have been reviewed and found to satisfactory by development Engineers, and QA Engineers, and is stable, that is, no further changes to dialog boxes or other interface elements are planned. Minor changes (word-smiting, UI adjustment, etc.) are acceptable, but must be arranged with the Development and Test Engineers.
- All code bugs that are exposed are corrected.

6.2 Integration Test

There are two applications that will be integrated for Integration Testing. The two applications are PALS Android Application and Administrator Website. The following describes the entry and exit criteria for Integration testing.

6.2.1 *Integration Test Entry Criteria*

The Integration Test Entry Criteria will rely on both applications to be operational, and the web services module in Administrator Website can fetch data from database. Things that must be done on entry to the Integration Test stage:

- All PALS Android Application functions, Find (ATM, Gas Station, Repair Shop), Add New, Get Direction, Make Phone Call, View Information, View Help, View About, must be coded.
- All Administrator Website function, Login, Add, Edit, Delete, Confirm, Reject, Search, Logout, must be coded.
- Big Bang approach will be implemented in integration test, where both applications are integrated together at once, then tested.
- Black Box Testing should either be in its late stages or completed.
- White Box Testing should have begun.

6.2.2 *Integration Test Exit Criteria*

The Integration Test Exit Criteria will rely on both applications to be operational, and the web services module in Administrator Website can fetch data from database. To exit the Integration Testing phase 100% success rate must be achieved. Things that must be done on exit from the Integration Test stage:

- All code bugs that are exposed are corrected.
- The PALS Android Application and Administrator Website's web services will interact together to complete a user request to fetch data or insert data, according to the System Specification Design.
- Both Modules are ready for System Testing.
- Black Box Testing is completed.
- White Box Testing should either be in its late stages or completed.

6.3 System Test

The System Test criteria apply for purposes of categorizing defects and the assessing the quality level of the product. All elements of the PALS Android application and Administrator Website are meshed together and tested as a whole. System test focuses on functions and performance, reliability, compatibility, installation, and behavior during special conditions.

6.3.1 *System Test Entry Criteria*

The Entrance Criteria specified by the Development Engineers, should be fulfilled before System Test can commence. In the event, that any criterion has not been achieved, the System Test may commence if both Development and Test Engineers are in full agreement that the risk is manageable.

- The Graphical User Interface and the PALS Android application and Administrator Website must be fully functional.
- All developed code must be unit tested. Unit and Link Testing must be completed and signed off by the development team.
- All test hardware and environments must be in place, and free for System test use.
- All Black Box testing must be completed and exposed bugs must be corrected.
- All White Box testing must be completed and exposed bugs must be corrected.
- Integration Testing must be completed and exposed bugs must be corrected
- Function Validation Testing is the accepted method of testing for all PALS Android application functions: Find (ATM, Gas Station, Repair Shop), Add New, Get Direction, Call, View Information, View Help, View About and all Administrator Website function, Login, Add, Edit, Delete, Confirm, Reject, Search, Logout.
- Development and Test Engineers agree that Function Validation Testing will cover function performance, reliability, and compatibility testing.

6.3.2 System Exit Criteria

The Exit Criteria must satisfy all the criteria listed below. This verifies that all elements of the project mesh properly. This is to make sure that all the system functions and performs according to the System Specification Document.

- All Function Validation Testing is 100 percent successful. Testing for all PALS Android application functions: Find (ATM, Gas Station, Repair Shop), Add New, Get Direction, Call, View Information, View Help, View About and all Administrator Website function, Login, Add, Edit, Delete, Confirm, Reject, Search, Logout, interact with complete accuracy.
- No degradation of System performance in PALS Android application across different Android platforms and screen sizes will be affected. (Android 4.2 or above is acceptable) and also No degradation of System performance in Administrator Website across different web browsers.
- The Graphical User Interface performs to System Specification Requirements.
- All the PALS System (both Android and Administrator Website) properties are expressed correctly through the Graphical User Interface.
- All input fields on the Graphical User Interface are working correctly.
- All high priority errors from System Testing must be fixed and tested.
- If any medium or low-priority errors are outstanding – the Development Engineers and Test manager must sign off the implementation risk as acceptable.

6.4 Shipping or Live Release

The PALS application testing is scaled down into phase Function Complete and follows the release criteria.

6.4.1 Shipping/Live Release Entry Criteria

The criteria for entering the final stages are QA verifies that all open product defects, regardless of fixed defects, documented, deferred, or otherwise addressed.

The software is frozen when the product passes its final milestone. If any code changes are made after the final milestone, the features fixed must be re-tested. QA, and Development Engineers closely monitor fixes that go into the final build to minimize risk. After the final milestone criteria have been met, the product enters the Live Release stage.

6.4.2 Shipping/Live Release exit Criteria

The Shipping/Live Release stage is when the product is ready for general availability to the public and the user documentation is final. The product must fully satisfy its release specifications and the user documentation must adequately describe the product's functionality. Both should be ready for use by the end user.

- QA tests the final product version to verify that the product to be released to the general public is of the utmost quality and satisfies original design specifications.
- The product must receive approval from the product team.
- QA and Development must prepare Release Notes.

The product is now ready to ship or published to production environment.

7 Bug Tracking/ Bug Process

During testing, the testing team members normally encounter behavior that goes against a specified or implied design requirement in the product. When this happens, we will document and reproduce the bugs for the developers.

Expectation of a bug:

- Keep track of what version of the application the bug is found
- Determine if bug has already been written up
- Indicate the steps to reproduce the bug – write enough details for others looking at the bug to be able to duplicate it; exclude unnecessary steps (i.e. If access point is irrelevant, be more general in your steps).

- Actual results – be specific on your findings.
- Expected results – how the product should behave based on the specified or implied requirements.
- Implications – How does the defect affect the quality of the product?

The following chart defines the impact levels to be used when entering bugs.

Impact	Definitions
1 – Fatal	**Test Stopper:** If you can't access a function and need the bug to be fixed immediately. The defect prevents QA from testing the feature area, sub-area or functionality of the feature.
2 – Serious	**Beta Stopper:** This is a bug that users would experience such as: data corruption, calculation errors, incorrect data, UE's and system crash on common user scenarios, significant QA risk, and major UI defects.
3 – Minor	**Live Release:** A bug that must be fixed before the product is officially completed, UE's or crashes, content, and UI and graphic changes required for release.

7.1 Various Roles in Bug Resolution

- **Author –** The person who wrote the bug; this will be someone on the QA team
- **Resolver –** Normally an Engineer assigned to a specific area of the application.
- **Verifier –** normally a QA Engineer responsible for testing the fix and closing the bug.

7.2 Bug Report Form

VHTN Software Development Problem Report #: ________

Program ____________________ Release ______________ Version ________

Report Type (1-6)_____	Severity(1-3)_____	Attachments (Y/N) ________
1 - Coding error	*1 - Fatal*	*If yes, describe:*
2 - Design issue	*2 - Serious*	____________________
3 - Suggestion	*3 - Minor*	____________________
4 - Documentation		
5 - Hardware		
6 - Query		

Problem Summary ____

Can you reproduce the problem? (Y/N) ________

Problem & how can it be reproduced?

____
____
____

Suggested fix (optional input)

____
____
____

Reported By: ____________________ Date: ____________

Items below are for use only by the development team

Functional Area:____________________ Assigned To:____________________

Comments:

____
____

Status_____ Prioirity(1-5)_____

1 - open *2 - closed*

Resolution(1-9)_____ Resolution Version ________

1 - Pending	*4 - Deferred*	*7 - Withdrawn by reporter*
2 - Fixed	*5 - As designed*	*8 - Need more info*
3 - Irreproducible	*6 - Can't be fixed*	*9 - Disagree with suggestion*

Resolved By: ____________________ Date: ____________

Tested By: ____________________ Date: ____________

Treat as Deferred (Y/N)_____

8 Roles and Responsibilities

8.1 Development Team

Code Development Project Leader – Andi Darmawan

- Ensure Phase 1 is delivered to schedule and quality
- Ensure exit criteria are achieved prior to system test signoff
- Regularly review testing progress with test controller.
- Raise and manage issues/risks relating to project or outside test teams control.
- Review and sign off test approach, plans and schedule.

SQA Project Leader – M. Farhan Mubarak

- Ensure Phase 1 is delivered to schedule and quality
- Regularly review testing progress
- Manage issues/risks relating to System Test Team
- Provide resources necessary for completing system test

8.2 Testing Team

Test Planner / Controller – Alvin

- Ensure Phase 1 is delivered to schedule and quality
- Produce high level and detailed test conditions
- Produce expected results
- Report progress at regular status reporting meetings
- Co-ordinate review and signoff of test conditions
- Manage individual test cycles and resolve tester queries/problems.

Lead Tester – Fakhria Nur Shabrina

- Identify test data
- Execute test conditions and mark-off results
- Prepare software error reports
- Administrate error measurement system
- Ensure test systems outages/problems are reported immediately and followed up.
- Ensure entrance criteria are achieved prior to system test start.
- Ensure exit criteria are achieved prior to system test signoff.

9 Test Schedule

The section contains the overall project schedule. It discusses the phases and key milestones as they relate to quality assurance. It discusses the testing goals and standards that we'd like to achieve for each phase of testing that will be deployed, e.g., Usability Testing, Code Complete Acceptance, Beta Testing, Integration Testing, Regression Testing, System Testing.
The key dates for overall Binary Tree development and Testing are outlined below. For details on the schedule, refer to the Binary Tree Project Schedule (this document). For details on general Engineering QA deliverables, refer to the test plan document.

Binary Tree Program Milestones	**End Date**	**Notes**	**QA Deliverables/Roles**
Planning and Analysis Phase	16/03/16	At this Milestone, the high level planning should be completed. Some of the deliverables are: Project Plan, System Definition, Program function specifications / requirements.	High-level test planning activities, which include preliminary development of Master QA Plan (this document, QA schedule, and requirement document).
Design Phase	30/03/16	This is a feature-driven milestone where the requirements and initiatives are further defined and solutions are finalized. The deliverables for this phase are UML diagram, navigation structure, system wireframe and mockup.	Development and Test engineers participate actively in feature design by inspecting and reviewing the requirements and design documents. As the design documents are completed, the test engineers are encouraged to start working on the Test Plan document and test design planning.
Code Complete -Infrastructure	15/04/16	This milestone is when all infrastructure development and functions should be complete. The testing team should have preformed unit & integration testing before checking the code into any build.	The Test Engineers should have completed or in the final stages of their preliminary Infrastructure Test Plan, test cases and other QA documents related to test execution for each feature or component such as test scenarios, expected results, data sets, test procedures, scripts and applicable testing tools.
Code Complete -Function	22/04/16	This milestone includes unit testing and code review of each function component prior to checking the code into the test phase. The deliverables include system-testing specification, Unit testing specifications, Integration plan.	The Test Engineers should have provided Code Complete Assessment Test to Development Engineer. The Test Engineers should also have completed or in the final stages of their preliminary White Box Test Plan, test cases and other QA documents related to test execution for each feature or component such as test scenarios, expected results, data sets, test procedures, scripts and applicable testing tools.

Binary Tree Program Milestones	**End Date**	**Notes**	**QA Deliverables/Roles**
Feature Complete	29/04/16	This phase allows for feature clean up to verify remaining bug fixes.	All bugs verified and QA documentation is finalized.
Ship/Live	30/04/16	Product is out.	Any unfinished Testing documents should be complete.

The Microsoft Project schedule is included at the end of this document.

10 Deliverables

- System Design Documentation.
- System Function Specifications.
- PALS Android Application and Administrator Website source code
- Test plan document - this document should address testing objectives, criteria, standards, schedule and assignments, and testing tools.
  - Unit Testing Plan
  - Integration Plan
  - System Testing Plan
- Test Design Document
  - Unit white-box test design – covers white testing criteria, methods and test cases
  - Unit black-box test design – covers black-box testing criteria, methods and test cases
  - System test design – covers system test criteria, methods, and test cases.
- Test report document
  - Unit white-box test report – covers unit white box test results, problems, summary and analysis
  - Unit black-box test report – covers unit black box test results, problems, summary and analysis
  - System Test report – covers system test results, problems, summary and analysis

11 References

Pressman, Roger S. Software Engineering - A Practitioner's Approach. Fifth edition. The McGraw-Hill companies, Inc.

Kaner, C., Falk, J., Nguyen, H.-Q. Testing Computer Software. Wiley Computer Publishing, 1999.