**Software Quality Assurance Plan**

Mandatory Artifact for a web based centralized e-attendance System

Department of Information Technology, Government of Nepal

Version 1.1, 8-Aug-2021

Submitted by

XYZ CO

Document Change Control

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Date | Version | Author | Reviewed by | Approved by | Description |
| 2021-01-20 | 1.0 | XYZ |  |  |  |
| 2021-08-08 | 1.0 | XYZ |  |  |  |

Table of Contents

[1. Introduction 5](#_Toc79678774)

[2. Scope of Work 6](#_Toc79678775)

[3. Test Strategy 7](#_Toc79678776)

[3.1. Overall Test Strategy 7](#_Toc79678777)

[3.2. Testing Types 9](#_Toc79678778)

[3.2.1. Integration Testing 10](#_Toc79678779)

[3.2.2. System Testing 11](#_Toc79678780)

[3.2.2.1 User Interface Testing 11](#_Toc79678781)

[3.2.2.2 Functional Testing 14](#_Toc79678782)

[3.2.3. Regression Testing 16](#_Toc79678783)

[3.2.5 Data Conversion Testing 17](#_Toc79678784)

[3.2.6. Performance Testing 18](#_Toc79678785)

[3.2.7 Security and Access Control Testing 18](#_Toc79678786)

[3.2.8. User Acceptance Testing 19](#_Toc79678787)

[3.3 Tools 20](#_Toc79678788)

[3.4. Defect Management 21](#_Toc79678789)

[3.5. Environment 23](#_Toc79678790)

[3.5.1. Environment Details 23](#_Toc79678791)

[4. Resources 24](#_Toc79678792)

[4.1. Roles & Responsibilities 24](#_Toc79678793)

[5. Project Milestone 25](#_Toc79678794)

[5.1. High Level Test Schedule 25](#_Toc79678795)

[5.2. Project Build/Release Plan 25](#_Toc79678796)

[01/05/2021 25](#_Toc79678797)

[5.3. High Level Build Schedule 25](#_Toc79678798)

[6. Deliverables 26](#_Toc79678799)

[7. Communication 26](#_Toc79678800)

[8. Assumptions 27](#_Toc79678801)

[9. Risk & Mitigation 28](#_Toc79678802)

[10. Glossary of Terms 29](#_Toc79678803)

# 1. Introduction

***1.1. Background***

Most of the Government Organization here in Nepal have large number of employees working under various sections/departments/divisions/units etc. For all these employees, human resource department of organization needs to track their daily attendance which directly influence organization's productivity. Currently, in Department of Information Technology (DOIT) there is traditional way of logging the daily attendance manually in register sheet. This traditional method is so much time consuming and little bit hassle to manage. Thus, since Information Technology is now emerging like wave across the world contributing to make things easy to manage, DOIT is adopting the IT solution by Developing **"E-Attendance System".**

The Employee E-Attendance System will help Human Resource (HR) Department to see who is checked in and in what time. By doing this, we will be benefitted by keeping track of employee working hours. We will be ensured that we can manage employee's time effectively so that we can get most out of their time. We can maintain complete records of sick leaves and holidays, as well ensure that our employees are being paid correctly.

The E-Attendance System will include Employee Profile Management and Leave Management as Core Functional Module and will retrieve the Daily Attendance Data from Biometric Device using Web API.

***1.2. Purpose/Objective***

The main purpose of this Test Strategy Document for the proposed **"E-Attendance System"** is to document and define the “Software Quality Assurance Plan” in regards to the overall Project scope, including the following:

1. High level Testing methodology for the project implementation
2. Identification of the various Testing Phases, Types of Testing, Testing environments, Testing guidelines and Issue resolution
3. Identification of the Testing activities/Deliverables
4. Plan for Test execution
5. Description as regards to participation and expectations for the test phases
6. Approach to verify the work products/new system/all deliverables
7. Resource requirements/allocations, including hardware/software, and staff responsibilities
8. Requirements for environment and setup, for the test team
9. High level business process scenarios that will be covered during testing
10. An overview of the business flows, architecture, and testing infrastructure

The Testing activity will be carried over across the key phases as identified below:

1. Test Planning
2. Test Design and Development
3. Functional and Regression Test Execution
4. Defect Reporting and Management
5. Test Summary and Sign-Off

The intended audience for the document is as follows:

1. DOIT
2. Development Team
3. QA Team

# 2. Scope of Work

Following identified functional requirements of the proposed **"E-Attendance System"** are considered to be in scope of testing:

1. Login Management Module
2. Kaaj module Module
3. Leave Management Module
4. Profile Management Module
5. Employee Attendance Module
6. Report Module
7. Leave Apply Management Module

# 3. Test Strategy

The strategy adopted for testing the proposed **"E-Attendance System"** is based on a “phase-wise” approach as detailed in the following sections.

## 3.1. Overall Test Strategy

The test Strategy includes following steps:

***Step1: Review of Requirements***

In this phase, the QA team will study the requirements from a testing point of view to identify testable requirements. They will understand the requirement in terms of what they will test and figure out the testable requirements.

During this phase the team will interact with clients to analyze and understand their requirements in detail and test the legality as well.

Requirement Analysis phase includes following steps:

1. Identifying types of test to be performed.
2. Gathering details about testing priorities and focus.
3. Identifying test environment details where testing is supposed to be carried out.

***Step2: Test Planning***

This phase is the most important phase where all testing strategies will be defined. In this phase, the QA manager will get involved to determine the test plan strategy along with efforts and cost estimation of the project.

This phase involves:

1. Proper analysis of the E-attendance system ,
2. Test Strategy will be designed
3. Test objectives will be defined
4. Proper Resource planning
5. Plan Test Environment
6. Schedule & Estimation
7. Determining Test Deliverables

***Step3: Test Case Development Phase***

This phase involves creation, verification and rework of test cases and test scripts after the test plan gets ready. QA team will note the detailed test cases. Along with the test cases, the team will also prepares the test data for testing. Once the test cases are ready they will be reviewed by the QA lead.

A good test case is the one which is effective at finding defects and also covers most of the scenarios on the system under test.

A good test case should:

1. Be simple and transparent
2. Be Created with end user in mind
3. Avoid test case repetition
4. Not assume functionality and features of the software application
5. Ensure 100% coverage of software requirements
6. Name the test case id such that they are identified easily while tracking defects
7. Implement testing techniques
8. Return the Test Environment to the pre-test state
9. Generate the same results every time
10. The peers should be able to uncover defects in test case design

***Step4: Test Environment Setup***

In this phase, the development team will set-up the environment for the QA test team to execute the test cases. QA team will monitor the readiness of the given test environment and acquire the required environment details from the development team required for the testing.

***Step5: Test Execution***

The approved test cases will be provided to the QA test team. Then the QA team will start testing the E-attendance system based on the prepared test plans and test cases. QA team will then perform different types of testing to ensure the quality of the system.

***Step6: Test Cycle Closure***

In this phase the QA team completes the all the testing. The QA team will be called for meeting to evaluate cycle completion criteria based on test coverage, quality, time, cost, software, business objectives. QA team analyses the test artifacts such as test cases, defect reports etc.

The QA lead will then prepare the document at the end of the release by giving all the relevant information regarding the testing process throughout the release.

### 3.2. Testing Types

QA test team will perform various type of testing to make sure that the changes in the code are working as expected. Each types of testing have its own advantages.

Given below are some of the tests that will be performed by the QA test team.

Figure 1 Types of Testing

### 3.2.1. Integration Testing

Every module are already unit tested by the development team and after they are integrated the QA team will start Integration testing. In this step, the proposed QA team will identify the various modules of the “**E-Attendance**” system that needs integration testing. Different modules of E-Attendance system are:

1. Login Management Module
2. Leave Management Module
3. Profile Management Module
4. Report Module
5. Employee Attendance Management Module
6. Kaaj Management Module

These individual modules will be integrated then the QA team will verify the performance of the complete system. Data flow will be validated end-to-end to ensure smooth functioning of the system as a whole.

In case of E-attendance system, the employee will make the attendance using the biometric device. The data from the biometric device will be verified in the Report module of the system.

|  |  |  |
| --- | --- | --- |
| # | Item | Description |
| 1 | Test Objective | Ensures that the E-attendance system is working as expected with all the different modules integrated at the end. |
| 2 | Technique | Check for Interfacing points between each of the modules completed:  Verify for end-to-end data flow across modules  Verify the functional requirement across modules |
| 3 | Environment | Will be carried out in QA environment |
| 4 | Entry Criteria | Completion of the development and system Testing activities for all the modules  Integration Test points have been identified and Test Cases are prepared |
| 5 | Exit Criteria | All the Test cases have been executed and Test coverage status is 100%  All the Test cases are passed with no severe bugs/issues. |
| 6 | Special Considerations | Sufficient ramp up time required for the testers to understand the design and data flow.  A well-planned coordination is needed between teams to carry out such Testing.  Bug fixing effort and retesting effort will impact the total effort estimated for testing |

Table2 Integration Testing Details

### 3.2.2. System Testing

In this phase of testing, the QA team will

1. Evaluate the end-to-end specification of the system
2. Verify both functional and non-functional requirements
3. Verify proper data acceptance, processing and retrieval and appropriate implementation of business rule

This type of testing will be based on Black-Box technique; that is verifying the applications and its internal processes by interacting with the application via the Graphical User Interface (GUI) and analyzing the output or the results.

System Testing will include the following:

1. User Interface Testing
2. Functional Testing

## 3.2.2.1 User Interface Testing

In User Interface (UI) testing the testing team will verify user’s interaction with E-Attendance system. The team will test the UI functionalities and verify whether they meet the specifications or not.

QA will perform the following tests:

1. Placement of logout, profile and change password button
2. Highlighting notification message in dashboard
3. System menus in the sidebar
4. Display of controls, font, color, size etc.

|  |  |  |
| --- | --- | --- |
| # | Item | Description |
| 1 | Test Objective | To verify the following   * Window objects and characteristics such as menu, size, position, state and focus conform to standards. * UI of the pages should be as per the requirements. It would include various things like look-and-feel of the page, navigation within page, navigation to reports, display of controls, font, color, size etc. * UI of the Administrative screen should be as per the requirements of the UI guideline of the project. It would include look-and-feel of the page, navigation, font, color, size etc. |
| 2 | Technique | Create tests for each screen, pages and reports to verify proper navigation. |
| 3 | Environment | Will be carried out in QA environment. |
| 4 | Entry Criteria | * Integration testing is complete * UI checklist, System test plan and Test specifications are ready. |
| 5 | Exit Criteria | * All planned tests have been executed.   • All identified defects have been addressed. |
| 6 | Special Considerations | Bug fixing effort and retesting effort will impact the total effort estimated for testing. |

Table 1 User Interface Testing Details

## 3.2.2.2 Functional Testing

The goal of this test is to validate the software system against the functional requirements/specifications. It tests each function of the software application by providing the appropriate input, verifying the output against the Functional requirements. Also included are the testing of the screens for data, navigation and functionalities of the application. Focus will be on testing the dashboard and all the reports for data, report formats and navigation.

At first, the QA test team will

1. Determine the main functionalities of the system, Error Conditions and messages, usability testing i.e. whether the product is user-friendly or not etc.
2. Prepare number of test cases as per the functionality.
3. Create the input data for the functionalities to be tested.
4. Then the expected result will be determined Expected Result for the functionality that needs to be tested.
5. Finally after the execution of the prepared test cases, the Actual output and the Expected output are compared to find whether the functionality is working as expected or not.

Identify test Input (Test Data)

Compute the expected outcomes with the selected test input values

Execute Test Cases

Comparison of Actual and Expected Result

|  |  |  |
| --- | --- | --- |
| # | Item | Description |
| 1. | Test Objectives | Ensure proper target-of-test functionality, including navigation, data entry, processing, workflow and retrieval |
| 2. | Technique | This phase will be executed in each of the iteration.  Execute each use-case flow or function using valid and invalid data, to verify the following:   * The expected results occur when valid data is used * The appropriate error or warning messages are displayed when invalid data is used. * All the workflows will send notification to assignee as and when required. * Reports will be verified for correct data population * Additionally, Front end data will be verified through data verification from the database. |
| 3 | Environment | Will be carried out in QA environment |
|  | Entry Criteria | * Integration testing is complete. * System test plan is ready. * Data Plan is ready. |
| 5 | Exit Criteria | * All planned tests have been executed. * All identified defects have been addressed. |
| 6 | Special Considerations | * Testing team will verify the business rules and functions specified in the documents provided. * Any change of business rule and functions will have an impact on the test plan and testing cycle. * Bug fixing effort and retesting effort will impact the total effort estimated for testing. |

Table3 Functional Testing Details

### 3.2.3. Regression Testing

Regression Testing will be carried out by the QA test team in the application in different iterations. The objective of this testing will be to ensure that the software is remained intact.

The team will perform regression testing by re-executing the tests to analyze which part of the system might be impacted with the modified code.

It will help the team to prioritize the vast number of test cases by identifying areas of focus, priority and strategy for test execution. They will perform this test to ensure that the modified system continues to meet the requirements.

Examine and analyze the requirement of code modifications

Make sure the system is stable and operates properly

Select and prioritize the test case for regression testing

Detect and Report Bugs

Figure 2 Regression Testing Process

|  |  |  |
| --- | --- | --- |
| # | Item | Description |
| 1 | Test Objective | Ensure that the functionalities implemented in each of the iterations for different modules/functionalities remains intact after the implementation of subsequent modules. |
| 2 | Technique | Functionalities for each of the modules will be implemented in each of the iteration, as defined earlier. So it is imperative that modules implemented in the later iterations will be tested along with some of the functionalities from earlier iterations. |
| 3 | Environment | Will be carried out in QA environment. |
| 4 | Entry Criteria | * Functional Testing is complete for current module * Regression suite is prepared for earlier modules. |
| 5 | Exit Criteria | * All the Test cases are passed with no severe bugs/issues |
| 6 | Special Considerations | None |

Table4 Regression Testing Details

## 3.2.5 Data Conversion Testing

In data Conversion Testing, the QA test team will perform the system data study from where the data migration will be done. The team will perform series of tasks:

1. The team will validate the critical data contained in the current system.
2. Capture any additional data required for the initial operation of New system.
3. Data mapping of the data elements in the old and new system will be done to determine which fields needs to be updated prior to migration.

### 3.2.6. Performance Testing

During this testing phase the QA team will ensure that the application is performing properly and performs to the need of the business.

1. The team will understand the application response time.
2. They will run load test at the peak hours the system will be utilized the most to check the maximum number of users that the application can handle before it crashes.
3. They will detect the bottlenecks of the system like CPU utilization, memory utilization, Caching, bandwidth/network utilization, Disk usage etc. that can slow down the entire system
4. Verify response time of the application under low, normal, moderate and heavy load conditions.

Without Performance Testing, software is likely to suffer from issues such as: running slow while several users use it simultaneously, inconsistencies across different operating systems and poor usability.

### 3.2.7 Security and Access Control Testing

Security testing of any system is for searching all the possible loopholes and weaknesses of the system which might result in a breach in the security. The main focus of this testing is to keep the system away from any vulnerabilities or threats so that the system does not get exploited.

The main focus of the team will be to keep the system away from any vulnerabilities or threats so that the system does not get exploited.

The major focus areas of Security and Access Control Testing are:

**1. Network Security**

The test team will look for the vulnerabilities in network infrastructure

**2. System Software security**

The team will assess the weaknesses in various software’s like operating system database system etc. on which the system depends.

**3. Client Side Application Security**

Testers will test and ensure that no manipulation will be made through the browser.

**4. Server-Side Application Security**

The test team will perform this test to ensure that the server side is strong enough to block any vulnerabilities.

### 3.2.8. User Acceptance Testing

In this test the actual users will get involved in testing the E-attendance system to validate the system as per the requirement. The actual users will give their feedbacks after performing the test. The users will provide their feedbacks after they use the system.

They may provide positive comments, report bugs or defects or request changes. The received feedbacks will be communicated to the concern Project Manager. The QA team will track all the feedbacks provided from the users and then report to the concerned development team. QA will also give the UAT report after completion of User Acceptance Testing.

User Acceptance Testing (UAT) focuses mainly on the functional requirements of the application and performed by the real users of the system.

|  |  |  |
| --- | --- | --- |
| # | Item | Description |
| 1 | Test Objectives | To verify   * End system behaves in the desired way as specified in the specification requirement document. |
| 2 | Technique | * Identify a set of comprehensive Test cases to be the candidate for UAT * Run the TEST set to measure the coverage of User requirements. |
| 3. | Environment | Will be carried out in Pre-Production environment |
| 4. | Entry Criteria | * System Testing along with all other Testing is complete. * Test data specific to user and role are available to the testing team. * Users are identified to perform the Testing |
| 5. | Exit Criteria | * For each known actor type the appropriate function or data are available * All the Tests are completed with sufficient no of iterations. |
| 6. | Special Considerations | * Test data should be available specific to user/group level and role. * Bug fixing effort and retesting effort will impact the total effort estimated for testing |

Table5 User Acceptance Testing Details

### 3.3 Tools

The following tools will be employed for the Testing activities involved in the project:

|  |  |  |  |
| --- | --- | --- | --- |
| # | Item | Tools | Vendor/ In-House |
| 1. | Test Management | TestRail, JIRA | In-House |
| 2. | Defect Tracking | JIRA | In-House |
| 3. | Version Control | GitHub | In-House |
| 4. | Project Management | JIRA | In-House |

Table6 List of Tools for Testing Activities

### 3.4. Defect Management

Defect management would be taken care in JIRA. All the defects and issues during the Testing lifecycle will be logged in JIRA by the QA. All the developers and QA (Quality Assurance Engineers) will be provided access to the respective Project in JIRA according to work on the Defects.

The workflow that would be followed for Defect Management is as follows:

1. QA will typically log all the Defects and issues in JIRA. The defect detected by the QA for the first time will be posted in JIRA as **‘NEW’.**
2. The bug/defect with status **‘NEW’** will be assigned to developers and they will look into the bug to check whether the bug is valid or invalid.

If the bug is invalid the status will be changed to invalid. If the bug is valid then the status will be changed to **‘Assigned’** with proper “**Steps to reproduce”**. Any requirement clarifications will need to be assigned to Business Analyst.

1. Developers will keep an eye on the assigned issues/defects each day. They will change the status to **‘In Progress’** once they start working.
2. The status will then be changed to **‘Fixed** upon fixing the Defects.
3. Once the bug is fixed, QA will **‘Re-test’** the defects and change the status as **‘Closed’** or **‘Re-Open’** accordingly.
4. If the bug no-longer exists the QA will then change the status to **‘Closed’**



Figure 3 Defect Management

## 3.5. Environment

### 3.5.1. Environment Details

The details of the environment, where the testing will be conducted, are as follows:

|  |  |  |
| --- | --- | --- |
| # | Testing Type | Environment |
| 1 | Integration Testing | QA |
| 2 | System Testing | QA |
| 3 | Regression Testing | QA |
| 4 | Performance Testing | Staging |

Table7 Environment Details

# 4. Resources

### 4.1. Roles & Responsibilities

The following table describes the Roles/Responsibilities, as regards to the testing activities, for the proposed **“E-attendance”** System:

|  |  |  |
| --- | --- | --- |
| # | Item | Description |
| 1 | Test Lead | • Identifies, prioritizes, and implements testing activity  • Responsibilities:  - Overall review of system design , process  - Create Test strategy and Test specification  - Generate test plan  - Generate test model  - Test Execution  - Report to upper management  - Asses Project strength  - Mitigate Risks |
| 2. | Test System Administrator | •Ensures test environment and assets are managed and maintained.  • Responsibilities:  - Administer Test Management System  - Install and Manage access to Test Systems |
| 3. | Database Administrator, Database Manager | • Ensures Test Data (database) environment and assets are managed and maintained  • Responsibilities:  - Administer test data (database) |
| 4 | Tool Admin | Handles any issues related to Tools |

Table8 Roles/Responsibilities

# 5. Project Milestone

### 5.1. High Level Test Schedule

Below are the QA activities that we will carry out in order to perform the application testing:

|  |  |  |
| --- | --- | --- |
| Activities | From Date | To Date |
| Developing Quality Assurance Plan | 01/03/2021 | 01/05/2021 |
| Setting up goals of Project assurance |  |  |
| Assigning responsibilities to the members of quality team and determining the hierarchy of management |  |  |
| Gathering relevant information on the project standards and defining compliance criteria |  |  |
| Identifying the set of measurements and metrics to be used to determine quality levels and performance |  |  |
| Auditing Project Quality |  |  |
| Identifying and correcting any deficiencies in project activities |  |  |
| Analyzing Project Quality |  |  |

### 5.2. Project Build/Release Plan

### 01/05/2021

### 5.3. High Level Build Schedule

***01/04/2021***

# 6. Deliverables

The deliverables will be

1. Test Strategy and Test Plans
2. Test Cases
3. Test Execution Logs

# 7. Communication

The following are the proposed communication for the proposed ‘E-Attendance system testing:

1. Daily stand-up will be conducted with the developer and QA team to inform about the works both the team will be performing on the day.
2. Weekly meetings will be scheduled among all the team member to discuss the test plans and the progress of the test.
3. All the issues will be tracked by Jira on regular basis.

# 8. Assumptions

The functional testing strategy for the proposed **“E-Attendance”** system has been created and documented based on the following assumptions:

|  |  |  |
| --- | --- | --- |
| # | Item | Assumptions |
| 1 | Application | * The developer team should provide each build one day prior to the QA team so that the team can perform at least one round of smoke test before the final test executions could be performed. * Every build should escort by a Build release plan containing the build contains in details |
| 2 | Data and Database | * The input data must be provided by the respective application team. * The database must be equivalent to production in terms of data volume, hardware and software configuration * Application team to help functional test team write test cases for validating the correct execution of a given scenario and for profiling |
| 3 | Schedule | * Schedule to be altered by functional Test Lead in concurrence with the Project Plan |

*Table9 Assumptions*

# 9. Risk & Mitigation

The following are the risks and mitigation from the proposed **“E-Attendance”** system Functional testing perspectives:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # | Item | Risks | Severity | Mitigation |
| 1 | Application, Database, Infrastructure | Ambiguous Requirement | High | Test Team members to discuss with the Development Team to understand the requirement and application whereabouts |
| Inadequate mapping of Business to Functional | High | Test Team members to discuss with the Development Team to understand the requirement and application whereabouts |
| Changes incorporated even after Requirement freezing | Medium | Impact of such functionalities to be kept out of scope |
| Typical Data dependency | High | Typical data to be supplied prior test execution start |
| Environmental dependency | Medium | Version controlling need to be strictly maintained and monitored to prevent any violation |
| 2 | Project Risks | Test Failures | Medium | Use Friday nights and weekends to make up |
| Test pre-requisites not ready | Medium | Execute other non-dependent test cases |
| System Crashes during test | Medium | Action for Immediate restore and recovery activity |
| Re-Test due to Reports not available | High | Some Key tests to be repeated to assess impact of reports |

*Table10 Risk & mitigation*

# 10. Glossary of Terms

|  |  |  |
| --- | --- | --- |
| # | Term | Description |
| 1 | Bug | See Defect |
| 2 | Conversion Testing | Confirms the accuracy of the conversion procedures needed to initially load the data into the system. Also validates the usage of the data during day-to-day production activity. Performed during the System test level. |
| 3 | Defect/Bug | The deviation of an actual result from the expected result during the application testing. A flaw in the software with potential to cause a failure which is raised by a tester and is meant for a developer to fix. If the defect cannot be resolved only by developers, then the item would be considered an issue.  Defect Severities:  **Severity 1**: **Critical** - Catastrophic defect that causes total failure of the software or unrecoverable data loss. There is no work around. In general, a severity 1 defect would prevent the product from being released.  Examples: defects that cause the system to crash, massive performance degradation, data corruption, data loss, security violation or completely disrupt service.  **Severity 2: High** - Defect results in severely impaired functionality. A work around may exist but its use is unsatisfactory. In general, you would not release the product with such a defect. Examples: operational error, data integrity, some performance degradation, loss of functionality (no workaround)  **Severity 3: Medium** - Defect causes failure of non-critical aspects of the system. There is a reasonably satisfactory work around. The product may be released if the defect is documented, but the existence of the defect may cause customer dissatisfaction.  Example: a non-Client Financial Report is not recognizing an option correctly, but if a filter is set, the report can be generated with the proper output.  **Severity 4: Low** - Defect of minor significance. A work around exists or, if not, the impairment is slight. Generally, the product could be released and most customers would be unaware of the defect's existence or only slightly dissatisfied.  Example: A button or button set is slightly off center on a data screen, or the problem is purely cosmetic and not easily recognizable, minor problem, misspelling, UI layout, rare occurrence. |
| 4 | End-to-End Testing | Additional interface testing from the beginning to the end of a process including all upstream and downstream impacted systems that receive data, whether direct or indirect from the primary system. Performed during the System test level. |
| 5 | Entry Criteria | Metrics specifying the condition that must be met in order to begin testing at the next stage or level. |
| 6 | Environment | The collection of hardware, software, data and personnel that comprise a level of test. |
| 7 | Exit Criteria | Metrics specifying the conditions that must be met in order to promote a software product to the next stage or level. |
| 8 | Functional Testing | See System Testing. |
| 9 | Integration Testing | The objective of Integration Testing is to test the interaction of related data interface components in order to confirm that these components function properly when integrated together. This serves to identify and resolve major interface defects before starting System Testing. Typically conducted by the development team. |
| 10 | Interfacing System | Downstream or upstream system that may require change due to the primary system. |
| 11 | Issue | An issue can be any discrepancies, deficiencies or other abnormal system behavior noted during the entire phase of the software development life cycle. Issues are not limited to hardware, database, network, and integration issues alone. Issues are often process related as well. An issue is anything that hampers the normal workflow during SDLC. Items are considered to be issues if they involve more than a developer to resolve. An issue can be raised by anyone related with the concerned application during any phase of the SLDC |
| 12 | Primary System | Core application being developed or modified. |
| 13 | QA | Quality Assurance |

*Table11 Glossary of Term*