Test Strategy

SysAnatomy

Author : Saranya Radhakrishnan, Surya Valluri, Sourav Bhowmik
Date : 03 April 2016
Version: 1.0

Document Control

Document location

Location
https://github.com/rsaranya/Project-SysAnatomy

Author

Position	Name	CWID
Test Lead	Saranya Radhakrishnan	20062589
QA Tester	Sourav Bhowmik	20061723
QA Tester	Surya Valluri	20062574

Stakeholders and other contributors

Position	Name
Project Sponsor	Anthony Giorgio
Developer	Saranya Radhakrishnan
Developer	Sourav Bhowmik
Developer	Surya Valluri

Revision history

Version	Issue date	Author/editor	Description/Summary of changes
1	04/03/16	Sourav Bhowmik	First draft
1.1	04/05/16	Surya Valluri & Saranya	Covered few strategies and approaches
1.2	04/07/16	Sourav & Surya	Elaborated draft
1.3	04/07/16	Saranya	Final document changes

Reviewed by

Version	Issue date	Name	Position	Review date
1	04/03/16			

Approvals

Version	Issue date	Name	Position	Approval date
1	04/03/16			

Related documents

Document	Location
Design Document	https://github.com/rsaranya/Project-SysAnatomy/tree/master/Documents

Table of Contents

SysAnatomy

Introduction

This is the Test Approach Document for the project SysAnatomy. It contains high level test strategy for the project. The test effort will be prioritised and carried out depending on the priorities of the project which are defined in the Project Deign Document. This is a living document that may be edited as the project moves forward. The Project Sponsor shall review and approve the final version of this document.

Purpose

This document is meant to be completed and used by the project test team to record and guide how testing will be managed and carried forward for this project. The team's target is to develop a shared understanding of the overall approach, test and timings of test activities. This will give us a clear view of how to evaluate the system.

Objective

The objectives of this document are to clearly specify the testing procedures that would be used. We aim to achieve high quality and shorter lead times with minimum overhead, frequent deliveries and close teamwork in the team.

Scope

The scope of the project will consist of:

- Unit Testing
- Code analysis(static and dynamic)
- System Testing
- Environment testing
- Performance and availability testing
- Regression testing
- Acceptance testing

Test Strategy

Testing Methodology

- **Specification based / Black box testing**: This will include Boundary value analysis, State Transitions and Use case testing.
- **Structure based / white box testing:** This involves Statement coverage, Decision coverage, Condition coverage and Multi condition coverage.
- **Experience based techniques:** consists of exploratory testing and Error guessing.

Test Plan

The purpose of master test plan is to identify the testing plans for each release of the application. As the product being developed in stages, every module deployed will be tested. The following content must be included in the plan:

- **Testing approach for every release:** Based on the delivery and the module the approach of testing varies. For initial releases testing will be mainly focused on components functionality. In later cycles the focus will be mainly on does this add to the final outcome and how its performance can be enhanced. Once all individual components are tested thoroughly then integration testing and performance testing.
- Even though all these cycles of development and testing are carried out, the underlying rule is to perform regression testing and make sure the system is working fine.

While performing every phase of testing two key points to be focused are dependencies and risks. As we will be testing the entire system, dependency of one component with another plays a key role. At times few changes will be made at the risk of something working well and all such will be properly evaluated to make sure that dependency and risk management are efficient in the project.

Detailed Test Plan

Keeping the different modules in mind and the timeline of the deliverables we should design tests plans for: ***User Interface, Database*** and ***Reports***. Initially each module will be tested individually till everything is deployed into one system. In such case Interface, database and reporting modules can be tested while developing just to make sure correct functionalities. The following can be the goals for testing the system.

Testing User Interface:

- Test cases for user inputs.
- Test cases for user interaction components like buttons, text boxes, etc.
- Gathering reviews and suggestions from peers for the look and feel of the product.

Testing Database:

- Internally verifying all the columns definitions.
- Verifying free flow of data from the system to database without any discrepancies.
- Validating functionality of all functions and testing processed data.

Testing Reports:

- Generating sample reports on some garbage data and then reporting on sliced data.
- Then validating reports based on lookup with data from database.

Once testing has started all the failed test cases should be documented and tickets will be created, to make sure that they tracked correctly and rectified.

Testing Types

Unit / Component Testing

Each file will be tested individually before integrating it in their respective modules. This is to ensure correctness of individual forms and prevent unit or component errors on integration into the system. Junit will be used at this stage to check the correctness of individual Java functions. Developers will test the controls on UI to ensure correctness.

System Testing

Modules developed will be integrated into a system and tested to make sure that they are working fine as a system. This plays a key role, because individual performance doesn't matter if they won't work in coordination. For every phase of development, the system will be tested to make sure that the results are as the team expected.

Regression Testing

The project is being implemented in stages and the system keeps on getting upgraded constantly incorporating all changes. So for every cycle of new changes the test cases should get updated and older test cases should also run successfully. Team should make sure that new change in the system does not affect the previous working version of the system.

Integration Testing

The main plan of this testing is to make sure that components interact well and make sure that expected data and calls are exchanged between them. In detail testing emphasis will be more focused towards 'what' information is being sent to the data base and 'what' information is being sent for report generation. All such links will be tested thoroughly.

User Acceptance Testing

The working product will be shown to selective peers in order to test the correctness of the product. The product will be tested for its look and feel, usability and confinement to the requirement specification.

Testing Approach

Testing Objectives

The main objective of testing this product will be to ensure that the product conforms to the requirements specified by the client.

Test Execution

Test cases will be written for every component in the product. These test cases will then be executed for every component till it's been integrated into the module. Tests will be performed at every stage of integration until deployment.

Testing Challenges

- To test the product's performance in heavy traffic.
- To test the working of the product after system failure.

Key Roles, Accountability and Responsibilities

Proposed Test Team Structure

- Saranya Radhakrishnan will be testing the User Interface.
- Sourav Bhowmik will be testing the Database Structure, Working and Performance.
- Surya Valluri will be testing the Business Logic of the product.

Staffing and Training Needs

The following tools will be required for testing:

- Junit
- Git Bug Tracking