2022

TEST PLAN

PUSL2020 – RELEASE 1.0

GROUP 77

ChangeLog

Version	Change Date	By	Description
1.0	29/04/2022	Galaketiye Hasan	Initial Write
1.1	30/04/2022	Dedugala Bandara	Test methodology
1.2	01/05/2022	Dedugala Bandara	Finalize

Table of Contents

1. Introduction

1.1 Overview

The website is to be developed is a traffic accident reporting system for Sri Lankan general public. Clients of this project include, the Road Development Authority (RDA) of Sri Lanka, the Sri Lankan Police and Insurance Companies.

This test plan has been tailored to fulfil the customer requirement by implementing adequate test strategies in different stages of the development.

1.2 Scope

1.2.1 In Scope

Following functional requirements have been chosen and will be tested upon the first release of the application.

Functional Requirements

ID	Requirement under Test
Reporter User	
RF_1	A guest user should be able to register to the system by providing identity details and login information
RF_2	Should be able log in to the system using their login credentials
RF_3	Should be able to add their vehicle details

RF_4	Should be able to view list of vehicles they have added previously
RF_5	Should be able to report an accident
RF_6	Should be able to manage previously submitted accident reports
RF_7	Should be able to attach images to accident
RDA / Police / Insurance Employee User	
EF_1	Should be able to log in to back-office portal
EF_2	Should be able to view list of reported accidents
EF_3	Should be able to view a single accident in detail
EF_4	Should be able to approve or reject an accident
Police Employee	
EPF_1	Should be able to view graphical representations of approved accidents
Webmaster User	
WF_1	Should be able to log into admin panel
WF_2	Should be able to view list of employee users
WF_3	Should be able to add new employee by institution
WF_4	Should be able to manage and exiting employee

Non-Functional Requirements

ID	Requirement under Test
Authorization	
AU_1	The reporter dashboard should only be accessible to logged in reporters
AU_2	A reporter should not be able to view another's accident report
AU_3	The back-office dashboard is only accessible to employee users
AU_4	Admin panel is only accessible to webmaster users
AU_5	Uploaded images should only be visible to reported user and relevant employees
Validation	
VAL_1	Web forms must be validated according to data types and format
VAL_2	System should display error messages when a form validation fails with reasoning
Navigation	
NAV_1	Table views should display 10 records per page
NAV_2	Table should be sortable by columns

Target of the current release is to develop the application adhering to above critical requirements. At the end of the development, user acceptance testing will be carried out by domain users.

Application Tests

- Validation of NIC and Vehicle license format
- Image uploads to Minio object storage instance
- Every public method invocation of service layer representing a command in the domain logic.

1.2.2 Out of Scope

Requirements listed below will be not tested under current release.

1. Employee and Webmaster authentication pages are excluded from UI testing These interfaces are generated by framework defaults.
2. Identification information validation against government databases for authenticity
3. Browser compatibility testing
4. Volume including simultaneous users, database and file storage
5. Availability of the system

1.3 Quality Objectives

- The developed application must fulfil all the user-stories, functional and non-functional requirements documented in the requirement specification document.

- The team should be confident in the quality of the product. Stakeholders should be able make informed decisions regarding the quality of the application based on tests results.

- Ability to find defects and failures of the system as much as possible in order to reduce the level of risk.

- Bugs / issues are identified and fixed before customer sign off.

- Reduce the possibility of failure under specific condition(s)

- Avoid possible legal issues with the customer.

1.4 Roles & Responsibilities

Role	Responsibility
Business consultant	• Document requirements and build user acceptance test cases • Communicate test results, bugs and status of them with the client • Conduct UAT with the client
Project Manager	• Validate test reports before customer hand off • Track defects throughout the bug life cycle • Prioritize issues by severity
Technical Lead	• Allocate appropriate testing tools • Review code coverage and code quality • Communicate bugs/defects with developers
Developer	• Write unit tests • Make sure new source code (commit) would not brake previously written tests
DevOps Engineer	• Run unit and integration tests as part of continues integration • First configure and deploy application to UAT environment • Share system error log with development team for analysis
QA Analyst	• Develop functional test cases with test tools • Review UAT report • Develop test strategies for legal and ethical requirements

2. Test Methodology

2.1 Overview

The application under development is to be developed by V-model software development life cycle. It has been chosen due to following reasons.

1. Requirements of the application is clearly defined and fixed.
2. Test designs are made before the development phase
3. Product testing is clearly defined for each step, thereby allowing QA analysts to measure quality metrics during the entire life cycle of the project

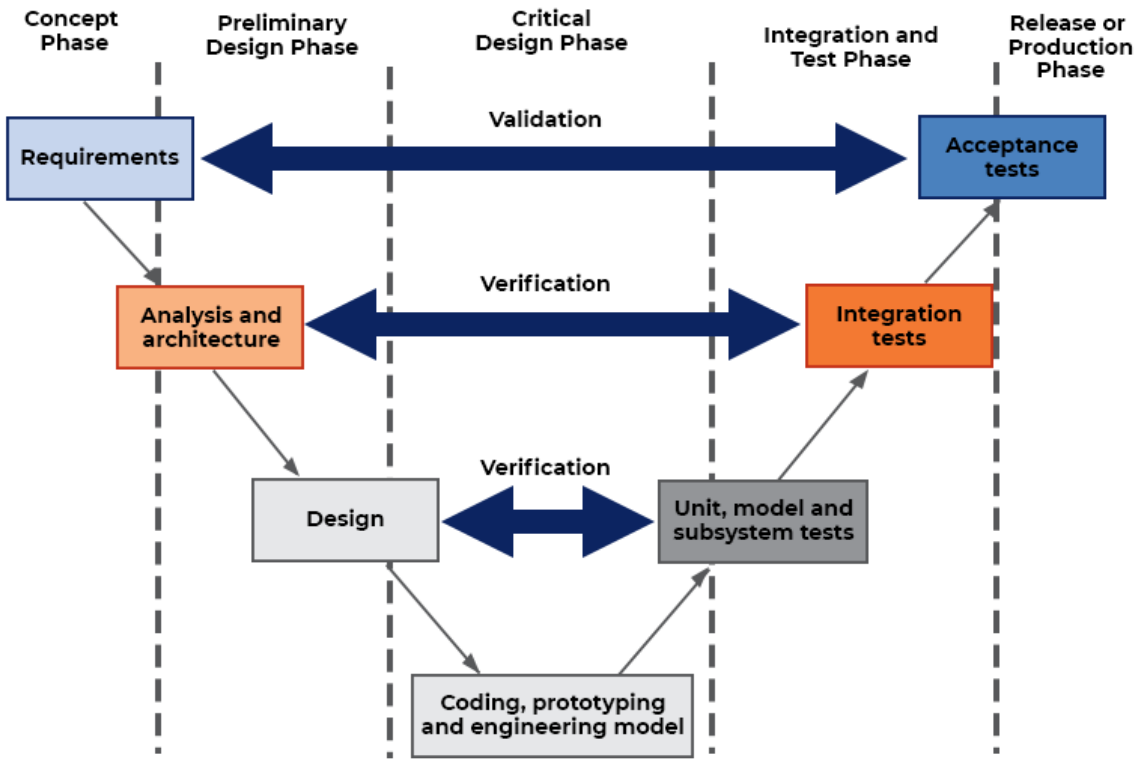

Figure 1 V-model

As seen in the Figure 1, User acceptance test cases are derived from requirements specification created at the beginning of the system. Analysis of the system,

including database and file storage are tested through integration testing. Domain and service layer is tested through unit tests.

However, team has agreed to incorporate agile practices to the above model by using SCRUM framework.

Doing so

- Each issue/bug raised will be added to the backlog.

- During a sprint meeting agile team can discuss the severity of the issue with product owner and project manager. Then the team will be able to identify critical issues and choose them for development in the upcoming sprint.

- QA analyst can provide insight into product quality and test metrices to customer and development team during sprint retrospective.

Solution Architecture

The structure of the application is modeled around clean architecture practices. By separating into layers, development team plan to test each layer with appropriate test strategy.

Project	Test Project	Description
.Domain	.Domain.Tests	Domain Entities, Value Objects, Enums and Validators (NIC, License, Phone, Vehicle Reg)
.Application	.Application.Tests	Contracts of service and data access objects (repositories)
.Infrastructure	.Infrastructure.Tests	Implementations of services and repositories
.Web	.Web.Tests	Controllers, View Models, Authentication

2.2 Test Levels

2.2.1 Unit Tests

- An individual method invocation is executed programmatically by a test.
- Verity of input parameters are seeded and output of the method is validated to check whether they reflect the expected outcome
- Both negative and positive test cases will be included
- Following value objects will be tested using unit tests.
    - NIC
    - Driver License
    - Vehicle Registration Number
    - Email
    - Phone
- Service classes depends on repository interfaces for data access. Therefore, repository interfaces will be mocked and injected to them.

2.2.2 Integration Tests

- External service compatibility, connectivity and functionality is tested in these tests.
- Prior to running tests, dependent services must be on-line and reachable to the application.
- This web application uses MS SQL Server 2019 and MinIO object server as external systems. Services can be initialized by using docker-compose deployment.
- Since integration tests are require dedicated environment to test, these tests are carried out by DevOps engineers at the end of a sprint release.
- Image upload functionality will be tested by integrating MinIO and Kestrel Web server.

2.2.3 Functional Tests

- Represent a user story
- Test automation is carried out by automating a browser
- Validates the functionality of the application from a customer point of view
- Require application to be deployed
- Helps identifying bugs before UATC
- Available technologies
    - Selenium – Cross language
    - Cypress – JavaScript only
    - Playwright – Light weight
- Functional tests for AUT will be carries out using Playwright and Specflow framework due to their tight integration with C# and .NET framework.
- Test are written in Gherkin syntax; then each step is programmed. Placeholders are used to test positive and negative test cases.

Gherkin Syntax Example

Feature	ReporterLogin
Background	**GIVEN** Clear the browser cookies before starting the scenario
Scenario	**GIVEN** I am in Reporter Login Page **AND** I see the login form **WHEN** I fill {email} and {password} field **AND** I click on Login button **THEN** I will be redirected to the reporter dashboard page

2.2.4 User Acceptance Tests

- A User Acceptance Test Case plan document is to be prepared by QA analyst.
- UATC plan will be divided into following modules
    - Reporter
    - Employee
    - Web Master
- Each module contains number of test cases, which are written in following format
    - Action identifying Test ID
    - Title explaining the user-story under test
    - Preconditions
    - Steps to be carried out
    - Data to be used for inputs
    - Expected results and actual result
    - Severity of the test case
    - Priority given
    - By whom the test was carried out
- Test plan document will also include the date of which it is carried out and reviewed by whom

2.3 Bug Triage

- Team will use You Track bug tracking tool to report and manage bugs
- When a bug is discovered; it is added to the sprint backlog with adequate details including,
    - Title
    - Description
    - Steps to generate the bug
    - Expected outcome
    - Actual outcome
    - Source Code path, class and or method
    - Exception occurred if any
- Bugs in the backlog is discussed during the daily standup meeting
- Depending on the severity, bugs are either added to the current sprint or deferred to the next sprint or release
- Bugs discovered from unit and integration tests are not shared with the customer
- Customer may also report bugs through You Track
- Customer and/or users will receive the feedback from QA regarding the status of tests carried out in UATC phase.

2.4 Suspension Criteria and Resumption Requirements

- After each iteration of the development branch, previously written tests are re-executed. If all unit tests are passing, no further unit testing is needed.

- When code coverage fell below 100%, uncovered methods should be tested. Tests should be written and caried out until full coverage is achieved.

- When a bug is added to the current sprint, it will be solved by the developer team and appropriate test cases will be created to validate the completeness. Once reviewed by the project manager or customer, bug will be marked as completed.

2.5 Test Completeness

- 100% test coverage of domain and service layer
- All continues tests are passing and application build action is successful
- All UATC test cases are marked as successful
- All open bugs are fixed. Non critical bugs will be fixed in next release

3. Test Deliverables

Artifact	Internal Team	Customer
Requirement Specification	X	X
Test Strategy Document	X	X
Bug Report	X	
User Acceptance Test Cases	X	X
Unit / Integration Test Metrics	X	
Functional Test Recordings	X	X
Application Error Log	X	
GitHub CI Log	X	
Customer Sign Off	X	X
Requirement Traceability Matrix	X	
Test Plan	X	

4. Resource & Environment Needs

4.1 Testing Tools

Tool	Test Level	Description
xUnit	Unit / Integration	• Add unit testing functionality to .NET 6 projects • Include single [Test], [Fact] test cases • Multiple inputs can be carried out via [Theory] • Dependency injection container support can be added using Collection Fixtures
Moq	Unit	• Provide ability to create mock objects for interfaces
Serilog	ALL	• Provide structured logging • Logging adapters are available for console and file-base logging
SpecFlow	Functional	• Support gherkin syntax style test cases
Playwright	Functional	• Provide headless browser instance for browser testing
Docker	Integration / Functional	• Containerization support • Deploy application in UAT with minimal server configuration • Build application
GitHub Actions	Unit	• Continues integration • Continues testing • Run unit tests as a build action
CodeCov	Unit	• Generate coverage metrices

YouTrack	UAT	• Bug reporting tool • Project management with SCRUM tools • Allow limited access to customer

4.2 Test Environment

4.2.1 Hardware

Minimum hardware requirement for UAT environment.

REQUIRMENT	REASON
X64 bit Processor with Virtualization enabled	Docker
Minimum 4 logical cores	Run multiple docker instances. MS SQL server require dedicated core.
8 GB RAM	MS SQL server join / aggregation operations
Network Card	LAN connectivity
8 Mbps Network	Web Application

4.2.2 Software

- HTML5 compatible web browser (preferably Google Chrome / Microsoft Edge)
- Docker Engine
- JetBrains Rider – For application debugging

5. Terms/Acronyms

TERM	DEFINITION
UATC	User acceptance test case