**Project name**

**Test Plan**

*This is a template to the project test plan,*

*It should be submitted as a Word file and as PDF.*

**Submitted by**:

**Supervised by:**

Table of Contents

[1. Introduction 3](#_Toc499375966)

[1.1 Purpose 3](#_Toc499375967)

[1.2 Project overview 3](#_Toc499375968)

[2. Test strategy 3](#_Toc499375969)

[2.1 Test objectives 3](#_Toc499375970)

[2.2 Test assumptions 3](#_Toc499375971)

[2.3 Data approach 3](#_Toc499375972)

[2.4 Scope and Levels of Testing 3](#_Toc499375973)

[2.4.1 Functional Test 3](#_Toc499375974)

[2.4.2 User Acceptance Test (UAT) 3](#_Toc499375975)

[3. Validation and Defect Management 4](#_Toc499375976)

[4. TEST ENVIRONMENT 5](#_Toc499375977)

[5. Test cases 5](#_Toc499375978)

[5.1 Function tests 5](#_Toc499375979)

[5.1.1 MY\_first\_test 5](#_Toc499375980)

[5.2 UAT tests 5](#_Toc499375981)

# 

# Introduction

## Purpose

This test plan describes the testing approach and overall framework that will drive the testing of the BTW Project. The document introduces:

* Test Strategy: rules the test will be based on, including the givens of the project (e.g.: start / end dates, objectives, assumptions); description of the process to set up a valid test (e.g.: entry / exit criteria, creation of test cases, specific tasks to perform, scheduling, data strategy).
* Execution Strategy: describes how the test will be performed.

## Project overview

……

# Test strategy

## Test objectives

The objective of the test is to verify that the functionality of BTW works according to the specifications. The final product of the test is twofold:

* A production-ready software;
* A set of stable test scripts that can be reused for Functional test execution.

## Test assumptions

…….

## Scope and Levels of Testing

### Functional Test

**PURPOSE**: Functional testing will be performed to check the functions of application. The functional testing is carried out by feeding the input and validates the output from the application.

**METHOD**: The test will be performed according to Functional scripts/Test procedures with a well-defined PASS/FAIL criteria.

### User Acceptance Test (UAT)

**PURPOSE**: this test allows the end users to complete one final review of the system prior to deployment.

**METHOD**: Will be performed manually by team members according to written test cases.

# Validation and Defect Management

It is the responsibility of the tester to open the defects, link them to the corresponding script, assign an initial severity and status,

It is the responsibility of the developer to retest after a fix is provided and close the defect.

Defects will be categorized according to the following severity status:

|  |  |
| --- | --- |
| Severity | Impact |
| 1 (Critical) | * This bug is critical enough to crash the system, cause file corruption, or cause potential data loss * It causes an abnormal return to the operating system (crash or a system failure message appears). * It causes the application to hang and requires re-booting the system. |
| 2 (High) | * It causes a lack of vital program functionality with workaround. |
| 3 (Medium) | * This Bug will degrade the quality of the System. However there is an intelligent workaround for achieving the desired functionality - for example through another screen. * This bug prevents other areas of the product from being tested. However other areas can be independently tested. |
| 4 (Low) | * There is an insufficient or unclear error message, which has minimum impact on product use |
| 5 (Cosmetic) | * There is an insufficient or unclear error message that has no impact on product use. |

# TEST ENVIRONMENT

….

# Test cases

## Function tests

### DataBsae Tests:

|  |  |  |  |
| --- | --- | --- | --- |
| **Objective** | **Enter:** | **Exit:** | **Defect Categorization:** |
| Insert Road information | Use SQL query to insert new road with legal values. | Road table affected, new tuple appears. | Critical – tuple isn't created, wrong values appear. |
| Insert Crossroads information | Use SQL query to insert new Crossroads with legal values. | Crossroads table affected, new tuple appears. | Critical – tuple isn't created, wrong values appear. |
| Insert Passageway information | Use SQL query to insert new Passageway with legal values. | Passageway table affected, new tuple appears. | Critical – tuple isn't created, wrong values appear. |
| Insert Weight information | Use SQL query to insert new Weight with legal values. | Weight table affected, new tuple appears. | Critical – tuple isn't created, wrong values appear. |
| Insert Place information | Use SQL query to insert new Place with legal values. | Place table affected, new tuple appears. | Critical – tuple isn't created, wrong values appear. |
| Can’t create illegal Crossroads | 1. Use SQL query to insert new crossroads. 2. Supply illegal Passageways id. | Crossroads table isn’t affected. | Critical – Old data table affected.  High – tuple is added to the table. |
| Can’t create illegal Passageway | 1. Use SQL query to insert new Passageway. 2. Supply illegal crosses roads id. | Passageway table isn’t affected. | Critical – Old data table affected.  High – tuple is added to the table. |
| Can’t create illegal Place | 1. Use SQL query to insert new Place. 2. Supply illegal road id. | Place table isn’t affected. | Critical – Old data table affected.  High – tuple is added to the table. |
| Retrieve Road Information | Use SQL query to get road data by road id. | Road table isn’t affected.  Correct information received. | Critical – Old data table affected.  High – received wrong information. |
| Retrieve Crossroads Information | Use SQL query to get Crossroads data by id. | Crossroads table isn’t affected.  Correct information received. | Critical – Old data table affected.  High – received wrong information. |
| Retrieve Passageway Information | Use SQL query to get Passageway data by id. | Passageway table isn’t affected.  Correct information received. | Critical – Old data table affected.  High – received wrong information. |
| Retrieve Place Information | Use SQL query to get Place data by Place id. | Place table isn’t affected.  Correct information received. | Critical – Old data table affected.  High – received wrong information. |
| Can’t get information with wrong id | Use SQL query to receive road information, with unknown id. | Road table isn’t affected.  Error message – unknown road. | Critical – Old data table affected.  High – received wrong information. |
| DB knows how to get information from JSON | 1. Create legal json file with system information. 2. Use DB functionality to keep the information from the json file. | The json file is recognized correctly, all data spread and inserted into the correct tables by columns. | Critical – Old data table affected.  High – received wrong information. |
| Get maps from DB | Use SQL query to receive map information by map id. | The correct json file describing the map id received. | Critical – Old data table affected.  High – received wrong information. |

## 5.2 UAT tests