**Machine Learning Using Deep Learning and Natural Language Processing**

**Peraton**

**www.peraton.com**

Testing Plan

**Date:**

11/12/2021

**Team Members:**

Rolando Gonzalez

Elshaday Alemayehu

Yusuf Siddiqui

Arial Carvalho

Eduardo Heredia

# Abstract

This document details the agreement between the sponsor and the development team. The test plan for our system’s implementation provides the approach to ensuring that our software follows the intended behavior and identifies defects in the implementation. This document outlines the objectives and processes for testing our system’s software and hardware. Our test plan ensures that all functional and design requirements are implemented as specified in the documentation by identifying test methods for unit and system testing. This test plan document shows whether our software works correctly as per requirements and the acceptance criteria as defined in accordance with stakeholders satisfaction. In this deliverable, we will see a detailed report that lays out the test procedure, testing targets, techniques used for testing, test plan, and test expectations to evaluate our product to see if it meets the functionality presented in the requirement. This includes specification based system level test cases, traceability of test cases to use cases, and techniques for test generation. This document is not final and is subject to change as needed by the sponsor.

Table of Contents

[Abstract 2](#_Toc87624776)

[List of Tables 4](#_Toc87624777)

[Introduction 5](#_Toc87624778)

[Requirements/Specifications-Based System Level Test Cases 6](#_Toc87624779)

[Traceability of Test Cases to Use Cases 12](#_Toc87624780)

[Techniques for Test Generation 13](#_Toc87624781)

[Evidence the Document Has Been Placed Under Configuration Management 14](#_Toc87624782)

[References 15](#_Toc87624783)

# List of Tables

Table 1: Login-1 6

Table 2: Login-2 7

Table 3: Upload-1 8

Table 4: Upload-2 8

Table 5: Contract Favorability Skew-1 9

Table 6: Contract Favorability Skew-2 10

Table 7: View ML Model performance-1 10

Table 8: View ML Model performance-2 11

Table 9: Traceability of Test Cases to Use Cases 12

# Introduction

This test plan document presents a series of techniques that will be used to satisfy testing requirements. This includes unit testing, integration testing, functional testing, end-to-end testing, acceptance testing, and performance testing. The purpose of this document is to demonstrate how the various testing techniques will be conducted and the various aspects of the system that will be tested to ensure that the system follows the intended behavior.

The structure of the test plan document is as follows:

1. Requirements/specifications-based system level test cases - shows test scenarios, prerequisites, actions, inputs, expected outputs, and test results
2. Traceability of test cases to use cases - shows the relationship between the test and use cases
3. Techniques for test generation - describes which techniques and provides a brief description about the techniques. Also specifies the criteria used to measure the quality of our tests

# Requirements/Specifications-Based System Level Test Cases

Table : Login-1

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Scenario ID** | | Login-1 | | |  | | |
| **Test Case Description** | | Login – Valid test cases | | |  | | |
| **Prerequisite** | | A valid user account | | |  | | |
| **Scenario Number** | **Action** | | **Inputs** | **Expected Output** | | **Test Result** | **Test Comments** |
| 1 | Launch application | | Open desktop app/ UI | Peraton App login page | | Pass | Launch successful |
| 2 | Enter correct Email & Password and hit login button | | Email id : test@peraton.com  Password: \*\*\*\*\*\* | Login success | | Pass | Login successful |

Table : Login-2

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Scenario ID** | | Login-2 | | |  | | |
| **Test Case Description** | | Login – Invalid test case | | |  | | |
| **Prerequisite** | | N/A | | |  | | |
| **Scenario Number** | **Action** | | **Inputs** | **Expected Output** | | **Test Result** | **Test Comments** |
| 1 | Enter invalid Email & any Password and hit login button | | Email id : invalid@gmail.com  Password: \*\*\*\*\*\* | The email address or phone number that you've entered doesn't match any account. Sign up for an account | | Fail | Invalid login attempt stopped |
| 2 | Enter valid Email & incorrect Password and hit login button | | Email id : valid@peraton.com  Password: \*\*\*\*\*\* | The password that you've entered is incorrect. Forgot Password? | | Fail | Invalid login attempt stopped |

Table : Upload-1

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Scenario ID** | | Upload-1 | | |  | | |
| **Test Case Description** | | Upload – Valid test case | | |  | | |
| **Prerequisite** | | User account with permission to upload | | |  | | |
| **S.No** | **Action** | | **Inputs** | **Expected Output** | | **Test Result** | **Test Comments** |
| 1 | Upload contract in pdf format | | Contract format .pdf | Contract successfully uploaded | | Pass | upload successful |
| 2 | Upload contract in word format | | Contract format .docx | Contract successfully uploaded | | Pass | upload successful |

Table : Upload-2

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Scenario ID** | | Upload-2 | | |  | | |
| **Test Case Description** | | Upload – Negative test case | | |  | | |
| **Prerequisite** | | User account with permission to upload | | |  | | |
| **S.No** | **Action** | | **Inputs** | **Expected Output** | | **Test Result** | **Test Comments** |
| 1 | Upload contract in excel format | | Contract Format extension .xlsx | Please enter the correct contract format. (word document or pdf format) | | Fail | upload unsuccessful |
| 2 | Upload corrupted file | | Corrupted file | Unable to read file. | | Fail | Upload unsuccessful |

Table : Contract Favorability Skew-1

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Scenario ID** | | Contract Favorability Skew-1 | | |  | | |
| **Test Case Description** | | Contract Favorability Skew– Valid test case | | |  | | |
| **Prerequisite** | | Processed Data (Contract) | | |  | | |
| **S.No** | **Action** | | **Inputs** | **Expected Output** | | **Test Result** | **Test Comments** |
| 1 | View Favorability Skew | | Processed document | Confidence Level - 80%  Favorable - 70%  Unfavorable - 30% | | Pass | Confidence level of model meets the minimum threshold |

Table : Contract Favorability Skew-2

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Scenario ID** | | Contract Favorability Skew-2 | | |  | | |
| **Test Case Description** | | Contract Favorability Skew– Invalid Test case | | |  | | |
| **Prerequisite** | | Processed Data (Contract) | | |  | | |
| **S.No** | **Action** | | **Inputs** | **Expected Output** | | **Test Result** | **Test Comments** |
| 1 | View Favorability Skew | | Processed document | Confident % - 40%  Favorable - 30%  Unfavorable - 70% | | fail | Confidence level of model does not meet the minimum threshold |

Table : View ML Model performance-1

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Scenario ID** | | View ML Model performance-1 | | |  | | |
| **Test Case Description** | | View ML model performance – Valid Test case | | |  | | |
| **Prerequisite** | | Processed Data (Contract) | | |  | | |
| **S.No** | **Action** | | **Inputs** | **Expected Output** | | **Test Result** | **Test Comments** |
| 1 | View ML model performance results | | Processed document | Most recent ML model performance compared to previously trained model | | Pass | View ML model performance successful |
| 2 | View ML model performance results | | Processed document | The words that didn’t meet the threshold parameters are: “examples..” | | Pass | View ML model performance successful |

Table : View ML Model performance-2

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Case Description** | | View ML Model performance – Invalid Test case | | |  | | |
| **Prerequisite** | | Processed Data (Contract) | | |  | | |
| **S.No** | **Action** | | **Inputs** | **Expected Output** | | **Test Result** | **Test Comments** |
| 1 | View ML model performance with empty results | | Processed document | No available previously trained model | | fail | View ML model performance unsuccessful |
| 2 | View ML model performance with empty result | | Processed document | Couldn’t show words that didn’t meet the threshold parameters specified by user | | fail | View ML model performance unsuccessful |

# Traceability of Test Cases to Use Cases

Table 9: Traceability of Test Cases to Use Cases

|  |  |  |
| --- | --- | --- |
| **Requirement** | **Test Scenario ID** | **S.No** |
| R02 | Login-1 | 1,2 |
| R02 | Login-1A | 1,2 |
| E02 | Upload-1 | 1 |
| E02 | Upload-1A | 1 |
| E03 | Text Extraction-1 | 1 |
| E03 | Text Extraction-1A | 1 |
| Y04 | Data Processing-1 | 1 |
| Y04 | Data Processing-1A | 1 |
| Y01 | Contract Favorability Skew-1 | 1 |
| Y01 | Contract Favorability Skew-1A | 1 |
| Y02 | View ML model performance-1 | 1,2 |
| Y02 | View ML model performance-1A | 1,2 |

# Techniques for Test Generation

Unit testing

The unit is the smallest and indivisible part in the software that is tested to not have analysis or programming errors. This white-box technique is considered one of the most popular in software development and is supported by many testing tools.

Statement Coverage

This white-box technique is widely used in the software testing industry. The statement coverage is given by the coverage percentage from the formula: actual number of statements covered divided by the total number of statements and the number of unreachable statements in the program. The statement coverage does not subsume decision and condition coverage.

Integration Testing

This black-box technique is performed when two or more components are combined to create a new subsystem or a system. This technique is intended to reveal any error that may exist during communications among the components being integrated. Combined with unit testing, these two become a powerful testing bundle for most software. Differently from the previous two methods, the integration testing is designed to determine whether the system meets the requirements.

Acceptance Testing

Testing conducted to determine if a system satisfies its acceptance criteria and to enable the customer to determine whether to accept the system. This method is also a black-box based testing.

The quality of the testing techniques are evaluated by the following criteria:

1. Number of test cases generated

Testing methods that the number of test cases grows exponentially is less desired than those that grow in a linear fashion.

1. Number of defects revealed

There is not a single testing technique that covers 100% of the program. Some techniques are able to cover more portions of the code than others.

1. Duration to complete testing analysis

Testing methods that take longer to complete their analysis are very time-consuming during the testing phase, e.g., mutation testing versus statement coverage.

# Evidence the Document Has Been Placed Under Configuration Management

Graphical user interface, text, application, email

Description automatically generated

# References

1. A. Sweigart, “Chapter 13 – working with PDF and word documents,” *Automate the Boring Stuff with Python*. [Online]. Available: https://automatetheboringstuff.com/chapter13/. [Accessed: 08-Oct-2021].
2. “Natural language toolkit,” *Natural Language Toolkit - NLTK 3.6.3 documentation*. [Online]. Available: https://www.nltk.org/. [Accessed: 08-Oct-2021].
3. “Pandas,” pandas. [Online]. Available: https://pandas.pydata.org/. [Accessed: 08-Oct-2021].
4. “scikit-learn,” scikit-learn. [Online]. Available: https://scikit-learn.org/stable/. [Accessed: 08-Oct-2021].