Quality Assurance Test Plan

A. Overview

1. Software Design Plan Summary

Endothon Finance is facing a software error on their customer facing web app. Their small company is focused on providing business loans and they have recently upgraded their systems to a new solution. In this new web app, the potential customer enters their information to be collected, sorted, and sent via **Endothon Finance** to themselves and other loan lenders that they work in parallel to. This application process asks for the applicant companies' last 5 years of financial records, and for companies that are less than 5 years old, all of their previous financial data and then also projects covering a 5 year span. This part of the application process on the web app is having an error that asks for the incorrect years records when prompting for the applicants financial information.

The software will receive a bug-fix that patches the area of code that has been causing the wrong prompting of years to the customers when requesting financial data from their companies. This new/fixed code will be able to be updated into the system to have the software perform as intended.

To accomplish this deliverable, the bug will be assessed through looking into the area of back-end code that is responsible for prompting the correct year to the potential customer using the web app. Next, either a rewriting of this code will need to be done or the front-end may need to be checked for the error of years and the html code fixed in that section of code. Then, the web app form will be tested to ensure accuracy of the fixes done to the system. Finally, the software will be fully updated and this solution will be complete.

2. Functional Requirements Objective

The overall objective is to update the current software to accurately prompt the financial questions by year that it prints those seeking to apply for a loan using the web app.

update the year prompting system for companies that have 5 years of financial history available. The prompt options for companies that are less than 5 years of age must appropriately suggest asking for the most current years of operation that the company does have, as well as a forecast of future financials for the remainder of the unfinished years in the 5 year period.

2a. Functional requirements objective metrics

Namely, to show the requirements above, the updating of the current software must include the proper 5 years of financial information being prompted to those companies that are 5 or more years old at time of applying. Also, for companies that do not have 5 years of financial history, the web app should accurately ask for the years that they do have and then also a forecast that covers the future up to the 5 year mark.

3. Non-Functional Requirements Objective

The overall objective of the non-functional requirements is to create an environment where the functionality of the program can be used as intended by the user and accessed also as needed by the developers and managers of the software.

for the web application has to have a proper uptime and be able to be readily used. Also, the companies back-end processes have to also be up and running and working in tandem to the web application front-end.

3a. Non-functional requirements objective metrics

The application must have an uptime that is constant and dependable for when the web app access is needed. The back-end also has to be running in tandem to the front end web application so that the entire program runs properly. For this, the hardware of the servers and computers that run both the front and back-end parts of the application also need to be operating correctly.

B. Scope

1. In-Scope Functional Requirements

The application needs to update the year prompting system for companies that have 5 years of financial history available. The prompt options for companies that are less than 5 years of age must appropriately suggest asking for the most current years of operation that the company does have, as well as a forecast of future financials for the remainder of the unfinished years in the 5 year period.

2. In-Scope Non-Functional Requirements

The web application has to have a proper uptime and be able to be readily used. Also, the companies back-end processes have to also be up and running and working in tandem to the web application front-end.

3. Out-of-scope Functionalities

This design plan only focuses on the software sections that are responsible for the prompting of the yearly financials for both 5+ year old companies and less than 5 year old ones. The front-end sections of code that reveal the years will also be checked to assess if it is a problem to do with the front-end display or just the back-end functions.

3a. Out-of-scope functionalities explanation

Any other aspect of the code that is not directly related to the years for financial information being prompted is not included in the scope of this design plan. No other front-end functions will be checked other than the aforementioned parts that show the prompting of years for loan applicants.

C. Test Strategy

1. Testing Overview

Test Case Table				
Test Type	**Description of Test**	**Objective**	**Test Owner**	**Environment**
Unit Test	Front-end test of company age being input as less than 5 years. The business age prompt will be filled into a test instance of the web app with "3", and the system is expected to then prompt for 2 years of profit forecasting.	*To show if the front-end is responsible for <5 code errors.*	***Dev***	*Local*
Unit Test	Back-end test of company age being input as less than 5 years. The business age prompt will be filled into a test instance of the back-end application with "3", and the system is expected to then prompt for 2 years of profit forecasting.	*To show if the back-end is responsible for <5 code errors.*	***Dev***	*Local*
Unit Test	Front-end test of company age being input as 5 or more years. The business age prompt will be filled into a test instance of the web app with "5", and the system is expected not to prompt for profit forecasting.	*To show if the front-end is responsible for 5+ code errors.*	***Dev***	*Local*
Unit Test	Back-end test of company age being input as 5 or more years. The business age prompt will be filled into a test instance of the back-end application with "5", and the system is expected not to prompt for profit forecasting.	*To show if the back-end is responsible for 5+ code errors.*	***Dev***	*Local*
End-to-End	Tests the application in the user's perspective for companies less than 5 years of age. The input of "3" will be put in a running instance of the web application (both front and back ends active). The expected result will be prompting for 2 years of profit forecasting.	*To show app functionality for <5 years old companies.*	***QA***	*UAT*
End-to-End	Tests the application in the users perspective for companies 5 or more years old. The input of "5" will be put in a running instance of the web application (both front and back ends active). The system is expected not to prompt for profit forecasting.	*To show app functionality for 5+ years old companies.*	***QA***	*UAT*

2. Sequence of Testing

The tests should be sequenced in the order of the above chart. This is because each one would sequentially tell if there needs to be a break to assess the code if it fails to perform the expected function. (Unit tests can be done in any order, but all should be done before the End-to-End tests are initiated, as Unit tests would reveal the internal happenings of the program that would need to be assessed if there were any functionality discrepancies).