**Commerce Bank Web App Test Plan**

**Team AAFK**

December 4, 2022

**Team Members**

Andrew Floyd

Kevin Floyd

Forrest Spencer

Andres Garcia

# 

# 

# **Introduction**

This document outlines the test plan for the Commerce Bank Web Application design for a user generated fundraising platform. As outlined in the project Requirements Document, this platform needs to provide a medium for all users to request funding for individual causes which other users may contribute. The testing activities discussed in this document will verify that the software for the creation and donation systems meet the needs of the users by verifying that the requirements for this platform, as outlined in the Requirements Document, are met.

The testing routine will test the Fundraiser, User, Donation, and Payment Information system’s response to creating, updating, and deleting, as well as browsing through previously generated fundraisers, and donations. The results of this testing procedure will enable the creators of this system to gauge project success as outlined in the Macro Project Plan.

# **Terminology**

Throughout this document the terms user, fundraiser, donation, system, site and username/password will be used frequently therefore, formal definitions will be given.

System- the database and controller that supports the backend of the website

User- people who have created an account on the website (those with a username and password) who will exclusively use only the front-end of the website

Fundraiser - A user generated cause they are raising funds for, which other users may donate.

Donation - A dollar amount contributed to an individual cause (fundraiser) by a user.

Site- the front-end website (main page)

Username/Password- unique identifiers that authenticate and validates a user

# **Items Tested**

Items that will be tested during the testing phase as laid out by the Macro Project Plan will be but are not limited to:

· Indexing of Fundraisers, Payments, and Payment Info

Test Case: 1.1, 2.1, 3.1

· Creation of Fundraisers, Payments, and Payment Info

Test Case: 1.2, 2.2, 3.2

· Viewing details of Fundraisers, Payments, and Payment Info

Test Case: 1.3, 2.3, 3.3

· Editing of Fundraisers, Payments, and Payment Info

Test Case: 1.4, 2.4, 3.4

· Deleting of Fundraisers, Payments, and Payment Info

Test Case: 1.5, 2.5, 3.5

# **Items Not Tested**

There are features that will not be included in the current testing procedure. This does not mean that these features will not be implemented, but that they are not available for testing. Those features include but are not limited to:

* Querying the Database (tests were made using mock data)
* Administrative Features that include adding/removing a user, signaling on call status and selecting email preferences

# **Approach**

The overall method to this testing procedure is manual system testing. Each test case created will have a direct link to the requirements as laid out in the Requirements Document. Each test case will test the security feature with invalid and valid data (null and non null values) to ensure that user requirement of valid data for these features is met.

The features that specify searching by keyword will be tested in separate test cases. Searching by keyword through previously asked questions and answers will verify the system’s ability to produce results that an individual can validate visually.

Manual system testing will continue throughout the second and third iteration of the project. For each iteration, both old and newly implemented features will be tested. Adding new features or functionality can sometimes interfere with the functionality of old features and to ensure product/project success, all features implemented should function as intended throughout the life of the software.

# **Item Pass/Fail Criteria**

The minimum requirements for this software system were laid out in the Requirements Document and the Macro Project Plan outlined what the creators of the software considered project success.

Implemented features that meet the requirements as determined by the customer, meaning the feature does what the user wants it to do with very little difficulty, passes the testing procedure. Difficulty, as used here, is determined by user comprehension and user ability to use the feature with little to no training.

Features that contain major defects will fail the testing procedure and will be documented via an incident report and turned over to the developer for investigation and revision.

# **Test Deliverables**

In addition to the Test Plan, other test deliverables include the Test Specification which outlines the specific test cases and expected results of each test, and Test reports which is comprised of Incidents, Defects and Changes.

# **Testing Tasks**

The following list the testing deliverables and the activities required to produce the deliverable.

| **Deliverables** | **Activities** |
| --- | --- |
| **Test Plan** | * Analyze Requirements for System Features * Determine Testable/Non-Testable Features * Develop Approach/Method for testing * Determine Task and Estimate Efforts * Develop Schedule for Testing |
| **Test Specifications** | * Analyze Requirements * Define Test Cases for Testable Features as Outlined by the Test Plan |
| **Test reports** | * Implement Test Cases as Outlined by the Test Specifications * Document Incidents and Defects * Determine Severity of Incidents and Defects * Determine Changes that Need to be Made to System * Document and Submit Change Request to Developer |