

**Table of Contents**

[***1***](#_heading=h.gjdgxs) ***Introduction 2***

[**1.1**](#_heading=h.30j0zll) **Overview 2**

[**1.2**](#_heading=h.1fob9te) **Goals and Objectives 2**

[**1.3**](#_heading=h.3znysh7) **Scope 2**

[**1.4**](#_heading=h.tyjcwt) **Definitions 3**

[**1.5**](#_heading=h.3dy6vkm) **Document Conventions 3**

[**1.6**](#_heading=h.1t3h5sf) **Assumptions 3**

[***2***](#_heading=h.4d34og8) ***General Design Constraints 3***

[**2.1**](#_heading=h.2s8eyo1) **Product Environment 3**

[**2.2**](#_heading=h.17dp8vu) **User Characteristics 4**

[**2.3**](#_heading=h.3rdcrjn) **Mandated Constraints 4**

[**2.4**](#_heading=h.26in1rg) **Potential System Evolution 5**

[***3***](#_heading=h.lnxbz9) ***Nonfunctional Requirements 5***

[**3.1**](#_heading=h.35nkun2) **Usability Requirements 5**

[**3.2**](#_heading=h.1ksv4uv) **Operational Requirements 5**

[**3.3**](#_heading=h.44sinio) **Performance Requirements 5**

[**3.4**](#_heading=h.2jxsxqh) **Security Requirements 6**

[**3.5**](#_heading=h.z337ya) **Safety Requirements 6**

[**3.6**](#_heading=h.3j2qqm3) **Legal Requirements 6**

[**3.7**](#_heading=h.1y810tw) **Other Quality Attributes 6**

[**3.8**](#_heading=h.4i7ojhp) **Documentation and Training 6**

[**3.9**](#_heading=h.2xcytpi) **External Interface 6**

[3.9.1](#_heading=h.1ci93xb) User Interface 6

[3.9.2](#_heading=h.3whwml4) Software Interface 7

[***4***](#_heading=h.2bn6wsx) ***System Features 7***

[**4.1**](#_heading=h.qsh70q) **Feature: <title> 7**

[4.1.1](#_heading=h.3as4poj) Description and Priority 8

[4.1.2](#_heading=h.1pxezwc) Use Case: <title> 8

[4.1.3](#_heading=h.49x2ik5) Additional Requirements 8

[**4.2**](#_heading=h.2p2csry) **Feature: <title> 8**

[4.2.1](#_heading=h.147n2zr) Description and Priority 8

[4.2.2](#_heading=h.3o7alnk) Use Case: <title> 8

[4.2.3](#_heading=h.23ckvvd) Additional Requirements 8

**Revision History**

| **Version** | **Date** | **Name** | **Description** |
| --- | --- | --- | --- |
| 1 | 12/02/22 | Andres Garcia | Initial Document |
|  |  |  |  |

# Introduction

## Overview

The commerce fundraiser web application will be available on all devices as it is deployed on Azure meaning it is live to view on any device using the link provided in the user guide document. This application allows any person to access our website to make donations by signing up, they will be able to see all fundraisers available, make donations, edit their fundraisers, and delete their fundraisers.

This document defines the requirements for the web application for Commerce Bank. The purpose of this document is to represent the system requirements and project requirements for the Web Application presented by Commerce Bank. Our task was to create a web application that allows users to log in, set up a fundraiser, track their donations, and view and donate to other users’ fundraisers. All data used to simulate a donation is fake or dummy data.

To find instructions on setting up an account and getting started, refer to the user guide. The user guide offers a more in-depth explanation of how to create an account and be able to set up your credit card or bank info to make donations. Our user guide also goes in depth through the donation page for other users to donate towards whatever donation page you have created.

Our web application is inspired by Go Fund Me which is a crowdfunding and donation website that allows the user to create a donation page for other users to donate towards the cause.

## Goals and Objectives

The main goals for our web application are:

1. Provide a website for users to create a donation/organization page to allow for crowdfunding and donations.
2. User can create, update, and delete their donation page
3. Other users can make donations towards a fundraiser page created by another user.

## Scope

Our web application allows users to be able to create a fundraiser page. Our website, however, only allows registered users to create a fundraiser page or donate after signing up first. Users do, however, make their donations anonymously when contributing to other users’ fundraisers. This is to follow the requirements and constraints of Commerce Bank.

## Definitions

**Commerce Fundraiser Application:**

The website that is being described here. The software system described in this document.

**Use Case:**

Describes a goal-oriented interaction between the system and an actor. A use case may define several variants called scenarios that result in different paths through the use case and usually different outcomes.

**Scenario**:

one path through a use case

**Actor**:

user or other software system that receives value from a use case.

**Role**:

category of users that share similar characteristics.

**Product**:

what is being described here; the software system specified in this document.

**Project**:

activities that will lead to the production of the product described here. Project issues are described in a separate project plan.

**Controls**:

the individual elements of a user interface such as buttons and checkboxes.

**CRUD**:

Create, Read, Update, Delete

## Document Conventions

Portions of this document that are incomplete will be marked with TBD. Each TBD item will have an owner and estimated date for resolving the issue.

## Assumptions

It is assumed that the client has access to the internet and can access the web application.

# General Design Constraints

## Commerce Fundraiser Web Application Environment

The Commerce Fundraiser Web Application will be deployed on Azure’s cloud services to make it available for all platforms. The Application is created using the .Net Core Framework including Entity Framework and written in C# with HTML and CSS. The Application is worked on and pushed onto GitHub to collaborate changes with colleagues. Those updates are then reviewed and pushed onto Azure and then published on Azure’s Web Services. This then publishes any changes live to the website and then it is restarted.





## User Characteristics

**Commerce Fundraiser Users:**

Any user that signs up to our web application. They have little to no technical experience.

## Mandated Constraints

The application is created to be viewed preferably on laptop. The application is written in C#, HTML and CSS, however if the device does not support these languages, then it may not run locally. You will need access to the internet to view the website application.

## Potential System Evolution

The application can evolve to become a mobile application for each mobile device available.

# Nonfunctional Requirements

## Usability Requirements

Users can easily navigate the website with little instructions or reference to the user guide.

Users can view and edit their active fundraisers.

Users can update personal information and passwords.

## Operational Requirements

Applications should be quick at responding to the users’ requests.

## Performance Requirements

Any changes made to the source code can be easily published on GitHub and then those changes be updated quickly with little downtime when uploading to Azure.

Application should run smoothly with no more than 5 second delay when redirected.

## Security Requirements

Users and Non-registered users should not get any hints when inputting a wrong login, the error will just mention that either username or password is incorrect.

Users do not have access to the database, it is locked behind Azure’s web service with a login needed to view the database and the information in it.

Users cannot edit and delete other users’ fundraisers.

## Safety Requirements

Application uses Model-View-Controller (MVC) architectural pattern to separate all components to individual components to limit any external threats to security.

Users do not have access to the database.

## Legal Requirements

Personal information like credit card info and bank account info is protected within the database

## Other Quality Attributes

Easily accessible to anyone that has an internet connection

## Documentation and Training

The Commerce Web Application will be available to users without the user guide or training. A user guide and system documentation will be provided to anyone that may want to understand how the application is used.

## External Interface

Database is connected through MSSQL Azure’s SQL server and connected to the website.

### User Interface

90% of users will be able to use 90% of the features within 20 seconds without prior training.

### Software Interface

Web application deployed on Azure’s Web services.

Database is connected through Azure.

Website is accessible through the given link.

# Functional Requirements

**4.1 Required Features**

**4.1.1 Use Case: 1**

**Description: User Register / Login**

Actors: Fundraiser User

Value = high

Cost = high

Basic Path

1. User clicks tab for Register on Webpage
2. System presents user to select an option tab: Login or Register
3. User Clicks on Sign Up
4. System prompts the user to enter their email and password.
5. User enters a systematic correct email and password.
6. System registers new user and then requests them to login with newly created account
7. User clicks Logoff.
8. System exits.

**4.1.2 Use Case: 2**

**Description: Create Fundraiser**

Actors: Fundraiser User

Value = high

Cost = high

Basic Path

1. Logged in user clicks “Make Lemonade” button.
2. System navigates the user to create a fundraiser form.
3. Users must enter a name of the fundraiser, a goal amount, a description and a URL image link.
4. The user then hits the create button and the system will process and add the fundraiser to the database.
5. The user can view their created fundraiser on the homepage or their Your Fundraisers page

**4.1.3 Use Case: 3**

**Description: Donate to Fundraiser**

Actors: Fundraiser User

Value = high

Cost = high

Basic Path

1. From the individual fundraiser details page, a user can click on the Donate button.
2. This button will navigate the user to the Donation Form page.
3. The system will prompt the user to enter a payment amount, and select or add a Credit Card or Bank Account information.
4. If the user selects to add, they will be taken to the add payment details page to enter their payment name, Card or Bank account number, CVC number if applicable, with expiration date, and billing address.
5. Once a user has entered their payment choice and payment amount, they click the donate button.
6. The system then processes the donation and adds it to the fundraiser page and donation goal amount.