Software Requirements Specification

For

Team 4

April 3, 2021

Version 2

Prepared by:

Ruby Rios

**Table of Contents**

[Revision History 3](#_Toc65519209)

[1 Introduction 4](#_Toc65519210)

[1.1 Overview 4](#_Toc65519211)

[1.2 Goals and Objectives 4](#_Toc65519212)

[1.3 Scope 4](#_Toc65519213)

[1.4 Definitions 4](#_Toc65519214)

[1.5 Document Conventions 5](#_Toc65519215)

[1.6 Assumptions 5](#_Toc65519216)

[2 General Design Constraints 5](#_Toc65519217)

[2.1 Product Environment 5](#_Toc65519218)

[2.2 User Characteristics 6](#_Toc65519219)

[2.3 Mandated Constraints 6](#_Toc65519220)

[2.4 Potential System Evolution 6](#_Toc65519221)

[3 Nonfunctional Requirements 6](#_Toc65519222)

[3.1 Usability Requirements 6](#_Toc65519223)

[3.2 Operational Requirements 6](#_Toc65519224)

[3.3 Performance Requirements 6](#_Toc65519225)

[3.4 Security Requirements 7](#_Toc65519226)

[3.5 Safety Requirements 7](#_Toc65519227)

[3.6 Legal Requirements 7](#_Toc65519228)

[3.7 Other Quality Attributes 7](#_Toc65519229)

[3.8 Documentation and Training 7](#_Toc65519230)

[3.9 External Interface 7](#_Toc65519231)

[3.9.1 User Interface 7](#_Toc65519232)

[3.9.2 Software Interface 7](#_Toc65519233)

[4 System Features 8](#_Toc65519234)

[4.1 Required Features 8](#_Toc65519235)

[4.1.1 Use Case 1 8](#_Toc65519236)

[4.1.2 Use Case 2 8](#_Toc65519237)

[4.1.3 Use Case 3 9](#_Toc65519238)

[4.1.4 Use Case 4 10](#_Toc65519239)

[4.1.5 Use Case 5 11](#_Toc65519240)

[4.1.6 Use Case 6 12](#_Toc65519241)

[4.2 Optional Features 12](#_Toc65519242)

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Name** | **Description** |
| 1 | 02/28/2021 | Ruby Rios | Initial Document |
| 2 | 04/03/2021 | Ruby Rios | Updated to account for change in tools and decisions about stretch goals |

# Introduction

## Overview

The Commerce Bank Project will be a web application to allow Commerce Bank customers to get updated about their recent bank account activity. Or, more specifically, the web application will allow users to login to their Commerce Bank account, to view details about their transactions, to set and receive notifications about transactions based on the user’s preferences, and to manually enter in their own transactions.

This document provides information about the requirements for the Commerce Bank Project web application. The purpose of this document is to represent the system requirements in a readable way so that the client can understand them and verify them for correctness, but with enough detail that developers can design and implement a software system from them. Project goals, scope and definitions are given in the introduction.

Design constraints and application environment are described in the following section. Non-functional requirements are outlined for later verification. Functional requirements are given to show the system features and expected user interaction.

Project constraints will be included in separate documentation. The Software Project Management Plan will give specifics on project budget and schedule. A separate Test Plan document will address test specifications and procedures.

## Goals and Objectives

The three main goals of the Commerce Bank Project are to:

1. Allow users to login to their Commerce Bank account
2. Set and receive notifications about transactions
3. Manually enter in transaction details

## Scope

The Commerce Bank Project will provide users the ability to get updated bank account information through a web application. Through this application, users will be able to login to their Commerce Bank account, set and receive notifications about transactions, manually enter in transaction details, and have an intuitive way to view their notifications.

## Definitions

**Commerce Bank Project**- the product that is being described. This is the web application specified in this document.

**User** – the person or persons who will actually interact with the Commerce Bank Project.

**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.

**Client** – the person or organization for which the Commerce Bank Project web application is being built. For the product being described, this would be Commerce Bank.

**Role** – a category of users that share similar characteristics.

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

**Shall** – adverb used to indicate importance; indicates the requirement is mandatory. “Must” and “will” are synonyms for “shall”.

**Should** – adverb used to indicate importance; indicates the requirement is desired but not mandatory.

**May** – adverb used to indicate an option. For example, “The system may be taken offline for up to one hour every evening for maintenance.” Not used to express a requirement, but rather to specifically allow an option.

**Developer** – the person or organization developing the system.

**Controls** – the individual elements of a user interface such as buttons and check boxes.

## 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

Our team is assuming that Commerce Bank will host and maintain the product past the product release. Transaction information and login information will be updated and integrated into Commerce systems by the bank. Performing full security scans and adding security enhancements to the product will also be performed by Commerce Bank. Ensuring that our project follows all legal governances for storing and maintaining personal information will be the responsibility of Commerce Bank.

# General Design Constraints

## Product Environment

The Commerce Bank Project web application is a web application attached to Microsoft SQL Sever. The software to be used in the development of the Commerce Bank Project web application is a Blazor Server (C#), Blazor Pages with Razor, Boostrap, Dapper, and a Microsoft SQL Server..

## User Characteristics

Commerce Bank Project Users: Commerce Bank customers with computer access and Internet access. Proficiency with computers may vary amongst these users.

## Mandated Constraints

Any Commerce branding used in the application shall adhere to style guidelines provided. Commerce Bank has directly stated that they want for the product to be a web application, and to use an SQL Server (2012 or above).

## Potential System Evolution

Beyond this project, this system could be updated to accommodate more of the stretch goals listed on the project requirements page, such as adding in features to allow for users to retrieve their password if they have forgotten it, adding in different types of notifications, remembering sessions, etc. For system evolutions beyond features, this project could eventually be connected to a mobile application, or the web application could be updated to allow for other accounts or for bill tracking.

# Nonfunctional Requirements

## Usability Requirements

Our goal is to get constant feedback during this process for usability from both Commerce Bank staff, UMKC classmates, and other individuals to ensure usability. While individuals may still need additional help based on technical competency, our goal is that 85% of users will be able to use the Commerce Bank Project web application without assistance.

## Operational Requirements

The Commerce Bank Project web application will rely mostly on visual information as to hopefully increase the security of the product. While our team hopes to implement features for accessibility, due to concerns of audio screen readers reading personal financial information out loud in public settings, the system shouldn’t depend on the user hearing audible output.

## Performance Requirements

We would like for the performance of our application to be as reasonable as possible for the user, and we will get user and stakeholder feedback about performance throughout the creation of the Commerce Bank Project web application.

## Security Requirements

Information for user login and transaction details should be stored in a secure database. As stated, security is assumed to be handled by Commerce Bank.

## Safety Requirements

The team does not believe there are any safety requirements that need implementation with this product.

## Legal Requirements

As stated in section 1.6, Commerce Bank is to ensure that our project follows all legal governances for storing and maintaining personal information.

## Other Quality Attributes

The team does not believe there are any other quality attribute requirements that need implementation with this product.

## Documentation and Training

The Commerce Bank Project web application will be delivered to users without documentation or training. A user guide and system documentation will be provided to project stakeholders.

## External Interface

### User Interface

The interface should be professional and intuitive. The design should be simplistic, helping the user to stay focused and easily complete tasks. To make the interface more fun and appealing, occasional use of color from Commerce Bank’s approved branding colors will be used. Accessibility will be a priority for this team, and fonts, font sizes, and differentiation of color will be considered in designing this application. It is expected that 85% of users will be able to use the Commerce Bank Project web application without help or guidance.

### Software Interface

The software to be used in the development of the Commerce Bank Project web application are a Blazor Server (C#), Blazor Pages with Razor, Boostrap, Dapper, and a Microsoft SQL Server.

# System Features

## Required Features

### Use Case 1

***Description: User Login/Checks Transaction Notifications***

Cost: High

Risk: Medium

Value: High

Basic Path:

1. User opens up the web application.
2. The system prompts user to enter in a username and password.
3. User enters a correct username and password.
4. System displays the home page, which has information on transaction details.
5. User clicks Log off.
6. System Exits.

Alternate Path:

1. User opens up the web application
2. The system prompts users to enter in a username and password
3. User enters an incorrect username or password
4. System displays an error message “Invalid Email and/or Password.”
5. User may choose to login again, returning to step 2, or exit.
6. System Exits

### Use Case 2

***Description: Enter in a Transaction Manually***

Cost: Low

Risk: Low

Value: Low

Basic Path:

1. User successfully logs in.
2. User clicks icon for “Enter in a transaction”.
3. System displays a new page entry boxes for entering in transaction details, such as (but not limited to) date of transaction, where the transaction was made, amount of transaction, etc.
4. User enters in these details and then clicks “Enter”.
5. The system saves transaction details into the transaction database and returns user to home page.
6. User clicks log off.
7. System exits.

Alternative Path:

1. User successfully logs in.
2. User clicks icon for “Enter in a transaction”.
3. System displays a new page entry boxes for entering in transaction details, such as (but not limited to) date of transaction, where the transaction was made, amount of transaction, etc.
4. User does not fill in all of the details necessary and clicks “Enter”.
5. The system displays an error message “Please fill in all transaction fields.”
6. User may choose to fix the missing fields, returning to step 2, or exit.
7. System exits.

### Use Case 3

***Description: Set Notifications***

Cost: Medium

Risk: Medium

Value: Medium

Basic Path:

1. User successfully logs in.
2. User clicks icon for “Edit Notifications”.
3. System displays a new page with current notifications and buttons to delete and edit current notification, and a button to set a new notification.
4. User clicks to set notifications.
5. System displays a new page with different buttons for various notification options (notification based on time, notifications based on amount, notifications to display at a certain time, etc.).
6. Depending on which option the user selects, new options will be displayed to help the user with setting that type of notification and an enter button.
7. User fill in all of the information required and then clicks enter.
8. The system saves this information in the notification database, and returns the user to the homepage.
9. User clicks log off.
10. System exits.

Alternative Path:

1. User successfully logs in.
2. User clicks icon for “Edit Notifications”.
3. System displays a new page with current notifications and buttons to delete and edit current notification, and a button to set a new notification.
4. User clicks to set notifications.
5. System displays a new page with different buttons for various notification options (notification based on time, notifications based on amount, notifications to display at a certain time, etc.).
6. Depending on which option the user selects, new options will be displayed to help the user with setting that type of notification and an enter button.
7. User does not fill in all of the information required and then clicks enter.
8. The system displays an error message “Please fill in all notification fields.”
9. User may choose to enter the missing fields, returning to step 4, or exit.
10. System exits.

### Use Case 4

***Description: Set Notifications***

Cost: Medium

Risk: Medium

Value: Medium

Basic Path:

1. User successfully logs in.
2. User clicks icon for “Edit Notifications”.
3. System displays a new page with current notifications and buttons to delete and edit current notification, and a button to set a new notification.
4. User clicks to set a new notification
5. System displays a new page with different buttons for various notification options (notification based on time, notifications based on amount, notifications to display at a certain time, etc.).
6. Depending on which option the user selects, new options will be displayed to help the user with setting that type of notification and an enter button.
7. User fills in all the information required and then clicks enter.
8. The system saves this information in the notification database and returns the user to the homepage.
9. User clicks log off.
10. System exits.

Alternative Path:

1. User successfully logs in.
2. User clicks icon for “Edit Notifications”.
3. System displays a new page with current notifications and buttons to delete and edit current notification, and a button to set a new notification.
4. User clicks to set a new notification.
5. System displays a new page with different buttons for various notification options (notification based on time, notifications based on amount, notifications to display at a certain time, etc.).
6. Depending on which option the user selects, new options will be displayed to help the user with setting that type of notification and an enter button.
7. User does not fill in all the information required and then clicks enter.
8. The system displays an error message “Please fill in all notification fields.”
9. User may choose to enter the missing fields, returning to step 4, or exit.
10. System exits.

### Use Case 5

***Description: Edit Notifications***

Cost: Medium

Risk: Medium

Value: Medium

Basic Path:

1. User successfully logs in.
2. User clicks icon for “Edit Notifications”.
3. System displays a new page with current notifications and buttons to delete and edit current notification, and a button to set a new notification.
4. User clicks to edit a notification.
5. System displays a new page with different buttons for various notification options.
6. Depending on which option the user selects, new options will be displayed to help the user with setting that type of notification and an enter button.
7. User fills in all the information required and then clicks enter.
8. The system saves this information in the notification database and returns the user to the homepage.
9. User clicks log off.
10. System exits.

Alternative Path:

1. User successfully logs in.
2. User clicks icon for “Edit Notifications”.
3. System displays a new page with current notifications and buttons to delete and edit current notification, and a button to set a new notification.
4. User clicks to edit a notification.
5. System displays a new page with different buttons for various notification options.
6. Depending on which option the user selects, new options will be displayed to help the user with setting that type of notification and an enter button.
7. User does not fill in all the information required and then clicks enter.
8. The system displays an error message “Please fill in all notification fields.”
9. User may choose to enter the missing fields, returning to step 4, or exit.
10. System exits.

### Use Case 6

***Description: Delete Notifications***

Cost: Low

Risk: Low

Value: Low

Basic Path:

* 1. User successfully logs in.
  2. User clicks icon for “Edit Notifications”.
  3. System displays a new page with current notifications and buttons to delete and edit current notification, and a button to set a new notification.
  4. User clicks to delete a notification.
  5. System displays a pop up, asking whether the user would really like to delete the notification, with buttons for yes and no.
  6. User clicks yes.
  7. The system deletes the notification from the notification database and returns the user to the homepage.
  8. User clicks log off.
  9. System exits.

Alternative Path:

1. User successfully logs in.
2. User clicks icon for “Edit Notifications”
3. System displays a new page with current notifications and buttons to delete and edit current notification, and a button to set a new notification.
4. User clicks to delete a notification.
5. System displays a pop up, asking whether the user would really like to delete the notification, with buttons for yes and no.
6. User clicks no.
7. The system does not edit the notification database and returns the user to the homepage.
8. User clicks log off.
9. System exits.

## Optional Features

None currently scheduled. Will be updated by Ruby if that changes.