Software Requirements Specification
For
[Commerce Banking Transaction Project]

[May 02, 2021]
[Version 2]

Prepared by:
Ozzie Loewen (Primary), Thomas Starr-Timberlake,
Kole Keeney, Sushant Acharya

Table of Contents

Revision History

Version	Date	Name	Description
1	03/02/21	Initial Requirements	This is the initial base of requirements for the Commerce Bank dealing with creating a transaction web application for the bank's users.
2	05/01/21	Completed Updated Requirements	This is the finalized version of the Requirements document with updated deadlines.

1 Introduction

1.1 Overview

This document defines the requirements for the Commerce Bank project related to creating a web application for Commerce Bank customers to check their monetary transactions and receive important notifications. The purpose of this document is to present the system requirements in a readable and understandable way for the clients and stakeholders to verify with enough detail that developers can design and implement a software system from them.

This document does not address project issues such as schedule, cost, development methods, development phases, deliverables or testing procedures. Those are addressed in a separate project document and quality assurance test plan.

The Commerce Transaction application is web based and used by the bank's members so they can view their transaction history, make modifications and receive notifications. It provides a way for authors to get direct and specific feedback from readers while imposing minimal additional work for authors and readers.

1.2 Goals and Objectives

The goals of the Commerce Bank transaction application are:

1. Be in web application form (using a newer web development framework, such as .Net).
2. Utilize a database.
3. User Experience should be easy enough for anyone to use.
    a. Text visible in font size and color.
    b. Should utilize everyday language (ex. 2:00p.m. instead of 14:00:00).
    c. Date should be readable instead of a timestamp format.
    d. Text in tables should be properly aligned.
        i. Numeric should be right aligned.
        ii. Alphanumeric should be left aligned.
    e. White space should be intentional and not sporadic.
4. Should be viewable on both desktop and mobile device.
5. Provide a reliable login page for the users to navigate to their personal account.
6. Front-End Development
    a. Make the application responsive and aesthetically pleasing.
    b. Front-end framework/libraries are up to you but must be included in project (aka no external resources), but you must use at least one CSS framework such as Bootstrap.
    c. Use Commerce Bank styling.

Dark Green: #006747
Green: #4FA800
Blue: #007AA3
Yellow Orange: #FFB300
Red: #E30000

Fonts: Poppins for Headings, Open Sans for Body Text.

7. **Login Page**
    - **a.** Simple login and passwords fields.
        - **i.** Mask the password field.
        - **ii.** Password minimum requirements:
            1. 8 characters
            2. 1 upper case letter
            3. 1 symbol
            4. 1 number
            5. 1 lower case letter
    - **b.** Login button
8. **Notification Rules** - Tool should allow for configurable notification rules to be created to notify user when transactions fit into a set of criteria.
    - a. Users should be able to Add/Edit/Delete notification rules without technical assistance.
    - b. These are examples (you have the liberty of coming up with what types of notification rules there are and how they are implemented):
        - i. Minimum of 3 notification rules added to system.
            1. Out of state transactions
            2. Timeframe usage
            3. Categories
9. **Notifications based on Notification Rules**
    - a. Main Requirement: Notification when the user logs in, examples:
        - i. Transaction in Alaska
        - ii. Transaction at 2AM
        - iii. Transaction from Spa
10. **Transaction Summary**
    - a. Transaction list sorted by date.
        - i. Don’t need to worry about searching/filtering.
    - b. Users should have the ability to add transactions here, which should automatically trigger any associated notification rules.
11. **Home Page**
    - **a.** Dashboard (Summary for triggered notification rules):
        - i. Number of times each notification rule has been triggered over the past month and year.

            1. Daily screen should be easy to read, easy to use, and provide a snapshot of data.
        ii. Ability to hide notification rules where the times tripped is zero.
    b. Ability to pull/compare notification rule different timeframes.
    c. Ability to export to spreadsheet.

12. **Stretch Goals for Group 1 Project**
    a. Options for user if they forget their username or password.
        i. Plans to email the user and have them reset with a set of steps.
    b. Use the Peer Review Method
        i. Two team members need to check the submitted code before implementing it. Should there be an inconclusive result, The Project Manger will have the final say.
    c. Remember Function
        i. When a user logs in, they will have the option to have their information stored for faster log in.

1.3 Scope

This project will provide banking members of Commerce with the ability to login to their account and view their transaction history. It should be an individual user's account so that they don't have access to other people's private banking information, and vice versa, other members don't have access to the user's information. A user should also be able to view notifications sent to them from the application. The user will be allowed to add transactions to their account themselves should they wish to make an update.

1.4 Definitions

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.

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.

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

1.5 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 or will be noted that it may be developed later as more information is learned.

1.6 Assumptions

It is assumed that the test data given to the group at the beginning of this project is not real customer data. It is assumed that the test data does not contain any sensitive information that would be detrimental to the company should it be accessed publicly in some way. It is assumed that the contract and provisions made by both company and team members will be upheld. It is assumed that the necessary steps taken to provide this learning experience have been approved by the proper authorities and entities. It is assumed that all software being used is compliant with coding standards. It is assumed that all team members, instructors, clients, sponsors, and stakeholders will act in accordance with the standards set by 'The ACM Code of Ethics and Professional Conduct' to the best of their ability.

2 General Design Constraints

2.1 Product Environment

The banking transaction web application will be a component of the bank's member's database as well as customer guidelines and verifications.

The commerce notification system will include a scalable web app designed to function within most web browsers, including desktop and mobile. The online frontend will interact with an SQL database with the transaction history of each user. There will also be some sort of admin view or backend access via the terminal to allow for efficient maintenance of the transaction database, allowing for the addition/deletion of transactions without accessing the frontend.

2.2 User Characteristics

The users for the transaction's web application are customers of Commerce Bank. They have an account set up and affiliated with Commerce. They are members who have signed up to use this application by associating their email/login identification along with a password to be able to view their transaction history and notifications associated with the account they have with Commerce Bank.

2.3 Mandated Constraints

The web application will be developed using ASP.NET 5.0.3, with most frontend development done utilizing the bootstrap CSS framework. The database will be created/maintained using SQL Server 2019. Development will be done using visual studio Community 2019 and testing will be done on a windows system, with the web frontend being manually tested using Google Chrome.

2.4 Potential System Evolution

The following transaction application could be easily extendable if given a good foundation for growth. Standards in this would be eliminating magic numbers, building off of previous code and not making code that needs to be fixed together. There are some side effects that occur when creating code based off of legacy items, such as creating around the initial set up, and having a hard time integrating new techniques, or another programmer's input. The way to minimize the hardships that comes with extending code and further development starts with setting a standard for coding conventions that every team member abides by and is able to incorporate into their coding practices. By doing this we can create documentation that allows for programmers to come in and reference the documentation and more easily pick up and understand the conventions of the project.

Specific options to extend the transactions application are other stretch goals:

1. Deploy the project into a windows server/cloud instance.

    a. Practice configuration properties for different environments

2. Use an open-source reporting tool/business intelligence suite for all the reporting and its data visualization.

3. Use pull requests/code review approved by a group member within your source control.

4. Security scan your application and fix Critical issues at a minimum. OWASP ZAP is a good open-source option.

5. Session for remembering user if they close their browser and then log in again.

6. Notifications via messaging center in the app

7. Notifications via email or text

Other options like creating a mobile app, rather than mobile compliant application is an option. Being able to update security as more advancements in that area are developed. Or developing using a more advanced framework. Going back through and making the existing code more efficient to use or take up less memory.

Creating an application that is able to be functionable for users who have vision impairment would also be a natural next step. It should also be noted that 'green' is often a color that is hard for people with visual color impairment to see. Creating a black and white, or grey scale version of the site would be prudent for creating a site that would be more easily viewed for Commerce customers.

It is understood that there are more ways to create extensible code and there are more and various options to extend. More to this section will be added should more ideas be presented through the development process.

3 Nonfunctional Requirements

3.1 Usability Requirements

1. Must be within the color scheme of Commerce bank.
    a. Options for colors are listed in the 2.2 section of this document and should be considered while creating the Front End or User based portion of this application.
2. Must use natural language.
    a. The site must use language that everyone can understand and interpret (ex. Writing out the date with the month, number and year; January 12th, 2021, rather than a 01/12/21 format.
    b. Time understanding, 2:00 p.m., rather than 14:00:00.
3. Formatting in a sensical way
    a. Any table formatting should maintain the same format throughout the site
    b. Numeric will be right aligned.
    c. Alphanumeric will be left aligned.
    d. Menus and notification options will appear on the left side of the screen, following the reading conventions of the United States.

3.2 Operational Requirements

The users' environment is noisy so the system shouldn't depend on the user hearing audible output. The user may only be operating on a cellphone and may need to access the site from a mobile device, or they may only be able to use a computer and have it be accessible on a desktop. The user will also need to be able to access the web application.

3.3 Performance Requirements

Maintainability: The database for each user should be able to be modified without altering any frontend code and without any knowledge of the users' login credentials (except for their username).

Speed: The application should finish most if not all database queries in less than 5 seconds with a database of 250 items (transactions).

3.4 Security Requirements

Access to data and features may be limited to specific users. There may also be a requirement to keep an audit trail of system use. This section describes the security requirements including the levels and what needs to be protected.

Access to any user's database should be limited to only the user and an administrator. Login information should also be limited to only the user and the administrator, ideally with the password being obfuscated from the administrator by using some form of encryption.

3.5 Safety Requirements

Our safety comes in the form of internet security as well as safe practices using electronics. The team's contribution for keeping users safe involves our login page feature and having each user create a unique account to access their banking information, no user will have the same username as another user.

We will be incorporating our SQL database without injecting the material to not create any weak points in our security. We will also be doing our best to keep the user safe by writing code that is efficient and runnable on the user's machines. We don't want to create a program that ends up consuming too much memory or processing power for the typical user.

3.6 Legal Requirements

A user's personal information or financial records should be kept confidential from all other users and should only be available to an administrator if they need to modify the database for any reason.

3.7 Other Quality Attributes

Portability: The web application should be able to be easily deployed on any compatible system by simply cloning the git repository and running the necessary commands to initialize the dotnet. SQL servers. Ideally, a batch file will be provided to simplify this process, requiring the administrator/server maintainer to initialize the web application with only one command.

3.8 Documentation and Training

The Commerce transaction notification application will be available to users without documentation or training. A minimal readme file will be provided with commands on how to set up the system and steps to modify the database may also be provided to server administrators. A user guide and aforementioned documentation will also be provided to stakeholders.

3.9 External Interface

External interfaces may be user interfaces or software interfaces.

3.9.1 User Interface

The user interface should be appealing and eye-catching. The user interface should also match the Commerce Bank green/blue aesthetic as requested by the stakeholders. There should be smooth yet consistent animations that give the site a distinctive and professional visual identity.

The user interface will be intuitive. The various functions of the website should be accessible through thoughtfully placed buttons/switches that mimic the design conventions of other websites that the user may be familiar with to maximize ease of use. At least 80% of users should be able to view and search transactions with zero training and 70% of users should also be able to create their own notification rules with minimal training.

3.9.2 Software Interface

The web frontend will interface with the SQL database to determine what notifications to trigger. There should be a web API that allows administrators to also interfaces with the SQL database without direct access to the web frontend.

4 System Features

4.1 Feature: Login

4.1.1 Description and Priority

1. Administrative view
    a. This view will be readable and writeable.
    b. Changes to accounts can be made in this view.
    c. Administrators will be software developers who can update changes to the site and the account viewing.
        i. This will ideally be based on the administrator’s experience and actions they find hard or unnatural to execute.
    d. Administrator’s will be able to view their previous transactions.
2. User View
    a. This will be the standard operation of Commerce Bank customers.

b. They will have access to their account with the option of adding transactions they made.
c. Users will be able to view their previous transactions.

This section will be expanded as development of the ‘Login’ feature is undergone.

All users, whether administrative or customer based, will be using the login page to access their account. However, administrator view, and user viewer differ on a few points.

Cost: medium
Risk: low
Value: high

4.1.2 Use Case: User Logs in

Actors: Commerce Bank Customer (user), admin

Preconditions: A user has accessed the web application and is viewing the login page

Basic Flow:

1. The user inputs their email address into the associated field.
2. The user inputs their associated password into the associated field.
3. The user clicks the Login button.
4. The system grants access to associated user home page and all allowed features.

Alternate Flow:

1. User account does not exist.

    a. The user will be directed to signing up with an administrator.

    b. System will display window and user will provide email address and valid password to finish creating account.

2. The user enters an incorrect password.

    a. The system shows a warning, and the user can try to login again.

        i. After multiple incorrect logins a notification will be triggered by system to the associated account.

b. The user forgets their password.

i. The user will have the option to request to change their password through their associated email or contact admin.

4.2 Feature: Home

4.2.1 Description and Priority

The user wants a comprehensive view of their account and notification settings on a singular page.

Cost: medium
Risk: low
Value: high

4.2.2 Use Case: Home Screen Navigation

Actors: Commerce Bank Customer (user)

Preconditions: User has successfully logged into their account

Basic Flow:

1. User will be able to view a dashboard of their notifications.
    a. This includes the number of times notifications have been triggered as well as associated types.
2. User will be able to sort available notifications by times in which system will update its display.
3. User will be able to select the transaction page in which system will change page accordingly.

Alternate Flow

1. User has no notifications to view.
    a. Dashboard will display a message to user relaying no notifications are available.

4.3 Feature: Transactions

4.3.1 Description and Priority

The user for Commerce Banking has successfully logged into the app. The user wants to add, view, or modify their financial transactions.

Cost: high

Risk: high
Value: high

4.3.2 Use Case: Viewing Transactions

Actors: Commerce Bank Customer (user)

Preconditions: User has successfully logged into their account; user has logged in and navigated to this screen

Basic Flow:

1. System will display table of transactions sorted by date.
2. User will be able to view all associated account transactions with a description and running total as well.

Alternate Flow:

1. User has no transactions.
    1. System will display a message relaying the account has no transactions.

4.3.3 Use Case: Adding Transactions

Actors: Commerce Bank Customer (user)

Preconditions: User has logged in and successfully navigated to this screen, user has a transaction to add

Basic Flow:

1. User selects option to add transaction from transaction page.
2. System displays window of form for user to fill out.
3. User enters all required data (Date, transaction type, amount, optional description).
4. System closes window, adds transaction to transaction table and refreshes.
5. System triggers a transaction added notification for the account.

4.4 Feature: Notifications

4.4.1 Description and Priority

User wants to be able to modify what criteria is necessary for a notification to be displayed.

Cost: medium
Risk: medium
Value: high

4.4.2 Use Case: Notification Tool

Actors: Commerce Bank Customer (user)

Preconditions: User has logged into account and selected settings from notification dashboard

Basic Flow:

1. User is able to view all notification settings.
2. User has the option to disable or modify any notification rules in which system will update and display accordingly after.
3. User is able to modify time-based notification rules.
4. User is able to modify location-based notification rules.
5. User is able to modify what kinds of notifications are triggered (in app, external).

4.5

Note: All features and use cases are subject to change and as the project continues additions and modifications are anticipated.