Software Project Management Plan

Commerce Bank Web Application

10/10/2021

Team Members

Edward Krulewich

James Ta

Jingfan Bai

Colin Turnham

Alex Norris

James Helm

FCBI
5110 Rockhill Road, Kansas City, Missouri

Document Control

Change History

Revision	Change Date	Description of changes
V1.0	10/10/2021	Initial pre-release

Document Storage

This document is stored in the project's SVN repository at:
https://github.com/UMKC-CS451R/cs415r_f21_groupproject-fcbi

Document Owner

Alex Norris is responsible for developing and maintaining this document.

Table of Contents

1 Overview

1.1 Purpose and Scope

Provide an attractive User Experience that is easy to understand and does not require a tutorial or guide to navigate through. Online registry should be easy, but secure and dependent on bank accounts that were made by an admin and proof of identity.

1.2 Goals and Objectives

Project goals:

1. Design a web application design that scales well to tablets and smartphones.
2. Create an easy-to-use web application to view bank account activity.

Project objectives:

1. Create a secure database that makes displaying account information safe and easy.
2. Build a client software that provides access to the database from all devices.
3. Create a feature for allowing transfers and transactions.
4. Create customizable notification and authentication options.

1.3 Project Deliverables

The following items will be delivered to the customer on or before 12/6/2021:

1. Full project source code.
2. Administrator's Manual
3. Plan for Quality Assurance
4. Unit testing
5. Database migration plan.

1.4 Assumptions and Constraints

Assumptions:

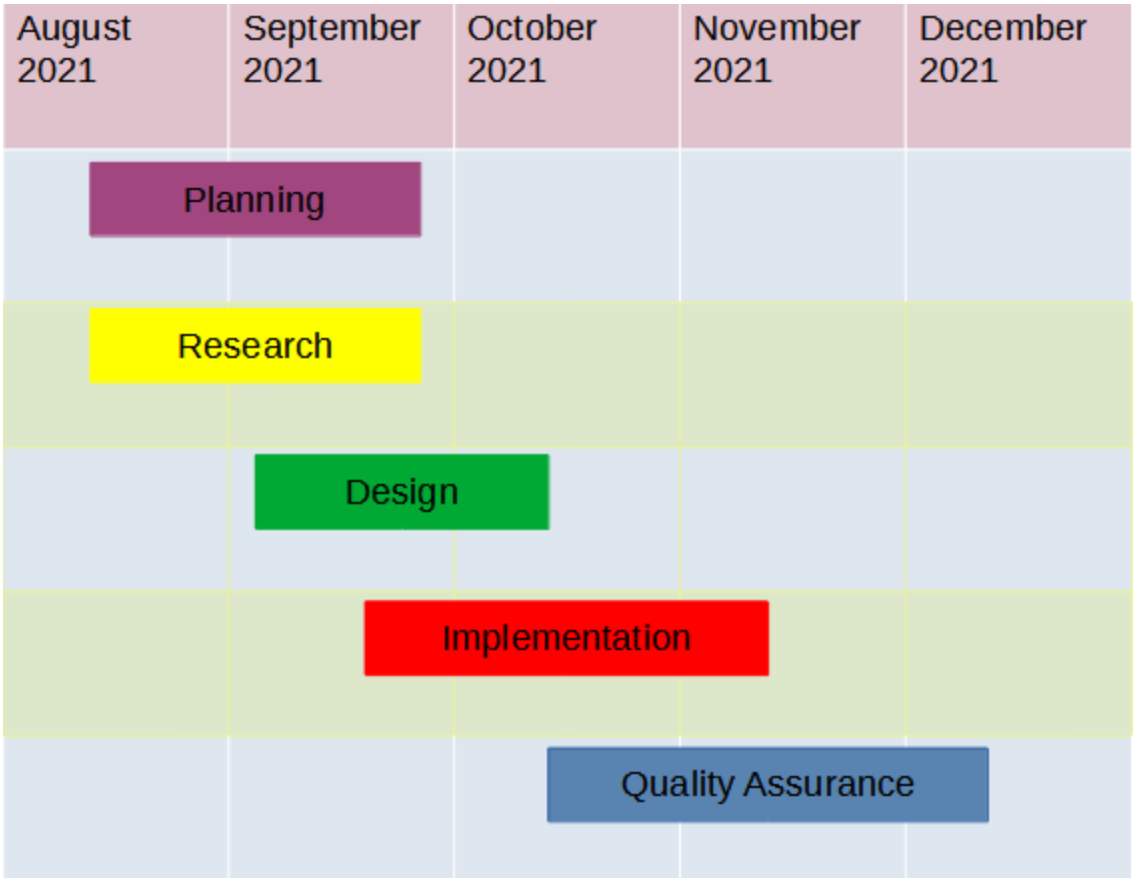
1. Full web application runs on test devices.
2. We will have two back-end engineers, two front-end engineers, and two quality assurance engineers .
3. The Windows & Microsoft SQL Server server will be delivered 12/6/2021.
4. Facilities for our work will be provided by ourselves
5. Full implementation of the server connection will be adapted to Commerce Bank's current system.
6. Our implementation will give the formatting and functionality for which Commerce Bank may connect their own systems for usage.

Constraints:

1. The software must run in a web browser.
2. The database must be secure.
3. The software must be ready by 12/6/2021.
4. Approval is required from Commerce Bank for any changes in implementation of features.
5. Any Commerce Bank symbol usage and page formatting must adhere to Commerce Bank styling constraints.

1.5 Schedule

- 10/9/2021 - Basic database implementation and front-end foundation.
- 10/25/2021 - Solidified database design, ReactJS implementation, Finish account settings UI, implement Unit testing framework.
- 11/8/2021 - Fully optimize for mobile and tablets.
- 11/22/2021 - Refactor code and deeper quality assurance.
- 12/6/2021 - Push final release.



1.6 Success Criteria

- Schedule is not compromised.
- Communication is maintained between engineers.
- Most important functionality is implemented by Dec 6, 2021.
- Trello board is properly maintained.

1.7 Definitions

- Front-end engineer - Programmer tasked with building user interfaces.
- Back-end engineer - Programmer tasked with building database and server routines.
- Quality Assurance engineer - Programmer tasked with developing software tests.

1.8 Evolution of the Project Plan

Project plan will be updated before and after each iteration to reflect possible issues or deliverables being available earlier than expected.

We will assess risks each time before updating the project plan. This will also serve as our time to further expand upon manual and unit testing.

2 Startup Plan

2.1 Team Organization

Project Manager: The project manager will maintain task assignment, inspect quality assurance for inaccuracies. Our project manager will also serve as a front-end engineer.

Back-end Engineers (2): Back-end engineers are tasked with server maintenance and database implementations as well as API request interfacing.

Front-end Engineers (2): Front-end engineers will be in charge of maintaining user interfaces, Razor Pages, and ReactJS implementations.

Quality Assurance Engineers (2): Quality assurance engineers will write unit tests and perform manual tests.

2.2 Project Communications

Information will be gathered through talks with Commerce Bank representatives and documentation for the software stack used. Our team will communicate through messages via Discord and email. Weekly meetings will be held to assess progress.

2.3 Technical Process

We will be following an integration of the Agile process. We will use iterations as sprints, and each sprint should result in a full application with each one resulting in more functionality. We spent two weeks planning and researching, then started designing during this phase. After we were done with the initial research we quickly moved to implementation and will be in implementation until November. After implementation, we will focus on quality assurance to make sure we have an acceptable final product.

2.4 Tools

- Programming Language – C#, JavaScript
- Version Control – GitHub source code repository
- Defect tracking – Manual testing and unit testing
- Build tools – local and main builds will be created using Visual Studio.
- Automated testing – unit tests will be implemented with the xUnit testing for C# within Visual Studio

3 Work Plan

3.1 Activities and Tasks

Task	Description	Owner	Effort Estimate	Actual Effort	Planned Start/Stop	Actual Start/Stop	Dependency
Database Design	Design the database	Edward, James H	50 hr		9/15 - 10/31	9/15 - In Progress	N/A
Database requests in front-end	Update user interface based on database	Alex N, James Ta	20 hr		9/27 - 11/8	9/30 - In Progress	Database Design
Implement Full UI/UX	Make front-end pretty	Alex N, James Ta	30 hr		9/15 - 11/8	9/15 - In Progress	N/A
Quality Assurance	Write unit test and perform manual tests	Jingfan Bai, Colin Turnham	50 hr		9/30 - 12/6	9/27 - In Progress	All tasks

3.2 Release Plan

- 10/9/2021 - First pre-release version
- 11/8/2021 - Initial prototype of final product
- 12/6/2021- Final product release build

3.3 Iteration Plans

- 10/9/2021 - Basic database implementation and front-end foundation.
- 10/25/2021 - Solidified database design, ReactJS implementation, Finish account settings UI, implement Unit testing framework.
- 11/8/2021 - Fully optimize for mobile and tablets.
- 11/22/2021 - Refactor code and deeper quality assurance.
- 12/6/2021 - Push final release.

3.4 Budget

1 project manager at 4 hours per week for 8 weeks 32 hours * $50/hr = $1600
2 requirements engineer at 4 hours per week for 8 weeks 32 hours * $40/hr = $2560
3 software engineers at 4 hours per week each for 8 weeks 64 hours * $40/hr = $3840
128 hours total, $8000 total, avg, $62.50 per hour

4 Control Plan

4.1 Monitoring and Control

Weekly/Wednesdays at 7PM – Team meeting. Project participants report status, progress and potential problems.

09/09/2021 - Gather requirements
09/15/2021 - Project Charter Complete
09/22/2021 - Iteration #1 Plan Complete
09/25/2021 - Project Plan Complete
09/30/2021 - Iteration #1 Complete
10/14/2021 - Test Report Complete
10/22/2021 - Iteration #2 Complete
11/17/2021 - User Guide and System Administration Manual Complete
12/07/2021 - Product Released

4.2 Project Measurements

Phase	Measurement	Source
End of each sprint	Record effort estimates for product features	Project Manager
Sprint Planning	Record effort estimates for scheduled tasks Update effort estimates for product features Update estimated dates in release plan	Project Manager
Sprint Closeout	Record actual effort for scheduled tasks Record actual effort for product features Record LOC count for modules written Address bugs and features to add	Engineering team
System Test	Record the rate at which errors are found. Write unit tests and perform manual tests. Report findings to back-end and front-end engineers	Quality Assurance Engineers
Project Closeout	Write documentation, upload to GitHub and push a release version.	Project Manager
Ongoing	Record defects found during implementation and maintenance phases. Prioritize bugs that risk entire system	Full team

5 Supporting Process Plans

5.1 Risk Management Plan

For our risk management plan, we will prioritize risks that could potentially expose private account details and account numbers. The security of this information should always be a top priority to ensure a secure banking application. Consequences of failing to address these risks can end with threat actors stealing private information and, in the worst case scenario, illicitly redirecting funds. In the event that these risks go unnoticed and are exploited, database connection should immediately be cut off until the exploit is resolved.

5.2 Configuration Management Plan

- cSource code will be saved to a GitHub repository and during initial implementation, we will run our system using local databases and Visual Studio.
- We will be using the standard Microsoft defined naming conventions for the C# language.
- All documents will be posted to the GitHub repository.
- During weekly meetings, items in our code that require changes will be discussed to determine how it will be done.
- If creative differences regarding code occur, a majority rules vote will determine the tie-break.

5.3 Verification and Validation Plan

Unit testing will be conducted to validate and verify our methods are properly performing their tasks. For more abstract methods, we will perform manual testing.

5.4 Product Acceptance Plan

Acceptable product quality would include an easy-to-use interface that doesn't take a long time to load onto your browser. The website should be navigable without the need to use a guide. It should also be secure so risks to account details and privacy should be minimal to nonexistent.