Software Project Management Plan

Commerce Web App

10/8/2020

Team Members

David Johnson

Brittney MacLennan

Evan Wike

Tyler Wheaton

Ashish Sharma

Document Control

Change History

Revision	Change Date	Description of changes
V1.0	10/8/2020	Initial release

Document Storage

This document is stored in the group's Github repository as well as in Google documents.

Document Owner

Evan Wike is responsible for developing and maintaining this document as the project manager.

Table of Contents

1 Overview

1.1 Purpose and Scope

The purpose for this project is to create an easy to use web application that customers can easily interact with to accomplish their goals, such as tracking transaction history and receiving notifications based on their preferences.

The scope of the project will allow customers to add, edit, and delete transactions notifications trigger, in addition to adding transactions manually, while also allowing customers to view transactions history and export it to spreadsheet.

1.2 Goals and Objectives

Project Goals:

1. The application functions in an intuitive manner.
2. The application is responsive and aesthetically pleasing.

Project Objectives:

1. Unit Testing covers at least 10% code.
2. Provide customers with an interface that allows them to easily receive alerts as well as manually add transactions.
3. Configurable notification rules that notify users when transactions fit a set of criteria.
4. Ability to pull/compare notification rules with different timeframes and be able to export to a spreadsheet.

1.3 Project Deliverables

The following items will be delivered to the customer on or before 12/09/2020.

1. Source code for both client and server portions of the project
2. User Guide
3. System Guide
4. Test Plan
5. Risk Management Report
6. Database information used for project

1.4 Assumptions and Constraints

Assumptions:

1. It is assumed that the user is online and has a stable connection.
2. It is assumed that the user has a web browser compatible with the web application.
3. It is assumed the user has an account with the bank at the time of login.
4. It is assumed the user has the valid login credentials.

Constraints:
This application must:

1. be a web application (rather than a desktop application).
2. be built in a “newer” web development framework.
3. utilize at least one CSS framework.
4. ensure user passwords contain at least:
    a. 8 characters
    b. 1 uppercase letter
    c. 1 symbol
    d. 1 number
5. provide the following features:
    a. Home Page (including a Dashboard w/ notification summary)
    b. Login
6. achieve at least 10% code coverage for unit tests.

1.5 Schedule and Budget Summary

Our schedule summary is as follows:

10/05/2010 - Iteration #1 Plan Complete

10/19/2020 - Iteration #2 Complete

11/02/2020 - Iteration #3 Complete

11/16/2020 - Iteration #4 Complete

12/07/2020 - Iteration #5 Complete

12/09/2020 - Project Complete

The budget summary can be found on the following page, broken down into categories.

Staffing

1 project manager at 4 hours per week for 14 weeks 56 hours * $66/hr = $3696

1 requirements engineer at 4 hours per week for 14 weeks 56 hours * $33/hr = $1848

2 software engineers at 4 hours per week each for 14 weeks 112 hours * $48/hr = $5376

224 hours total, $10,920 total, avg, $48.75 per hour

Equipment

4 computers at $800 $800 * 4 = $3200

Total = $10,920 + $3200 = **$14,120**

1.6 Success Criteria

- Requirements detailed in the Project Requirements documents are delivered on or before December 9.
- Total project cost doesn't exceed the budget outlined in section 1.5.
- Customers are able to receive notifications based on their settings.
- Customers are able to view and export their transaction history.

1.7 Definitions

actor	user, or other software system, that receives value from a use case.
BaaS	"Backend as a Service," a cloud service model that automates server side development and takes care of the cloud infrastructure.
customer	end user, the intended user of this software.
controls	the individual input elements in a user interface, such as buttons and checkboxes.
end user	customer, the intended user of this software.
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 to be used to express a requirement, but rather to specifically allow an option.
mobile-first	design philosophy that places mobile devices at the forefront of both design strategy and implementation; focuses on designing for the smallest screens first, before working back to laptops and desktops.
pagination	the process of splitting the contents of a website, or a section of its contents, into discrete pages.
product	the software system described in this document.

project	activities leading up to the production of the product described here. Project issues are described in a separate document.
RAD	"Rapid App Development," a form of agile software development methodology that prioritizes rapid prototype releases and iterations.
responsive	an approach to web design that allows web pages to render well on a variety of devices and screen sizes.
role	category of users sharing similar characteristics.
scenario	one path through a use case.
shall	adverb; used to indicate importance. Indicates the requirement is mandatory. Synonymous with *must* and *will* for the purposes of this document.
should	adverb; used to indicate importance. Indicates the requirement is desired, though not mandatory.
use case	describes a goal-oriented interaction between the system and actor, may define several variants, known as *scenarios*, that result in different paths through the use case, and usually in different outcomes.

1.8 Evolution of the Project Plan

At the start of each iteration, the project plan will be updated with the tasks that the group hopes to complete for the upcoming iteration. Upon the conclusion of the iteration, the project plan will be updated to reflect what was completed and the actual effort of each completed task. Risk mitigation will also be a part of each iteration and will be evaluated at the start, with severe risks being addressed and analyzed as soon as they become known to the project.

2 Startup Plan

2.1 Team Organization

Role	Responsible for...
Project Manager	Organizing team members, tracking tasks, managing risks, leading weekly team meetings, and providing monthly status reports.
Architect	System design, integration testing, and documenting System Architecture.

Developer	Designing, developing, and unit testing. Also responsible for documenting individual components.
UI Designer	Designing the UI and defining the overall theme of the application. The UI Designer will also define styles for individual components.
UI Developer	Implementing the *UI Designer's* theme and styles on both the application and the individual components.
Database Designer	Planning, designing, and developing the database. Also responsible for designing and documenting the Data Access Layer.
Database Developer	Implementing the design for the Data Access Layer and database testing.

2.2 Project Communications

Information for the group is gathered and distributed through two different methods:

1. Discord is used to communicate in a quick and effective manner while,
2. Google Drive (Docs, Sheets, etc.) is used to collaborate on documentation.

2.3 Technical Process

An agile development methodology is being used for this project, and will follow an iteration schedule that lasts for a period of two weeks for each iteration. Milestones and goals for each iteration will be recorded in this document, and actual effort for each task will be evaluated at the iteration closeout.

2.4 Tools

- **Programming Language:** Typescript
- **Client-side Frameworks:** Angular, Bootstrap
- **CSS Preprocessor:** Sass
- **Database:** user data and user authentication will be handled by Firebase using the Firebase SDK.

- **Version Control:** source code and written artifacts will be stored in a Git repository hosted on GitHub.

3 Work Plan

3.1 Activities and Tasks

Work Breakdown

1. Project Setup	
1.1 Project Initialization	
Description:	Install project dependencies, setup project structure, setup project IDE preferences, push to GitHub.
Owner:	Evan Wike (Project Manager)
Est. Effort:	2 hours
Actual Effort:	-
Planned Start-Stop:	10/12 - 10/14
Actual Start-Stop:	-
Task Dependencies:	None.
1.2 Theme Design	
Description:	Design the overall theme of the application. This includes picking primary and secondary colors, fonts, etc.
Owner:	Brittany MacLennan (UI Designer)
Est. Effort:	3 hours
Actual Effort:	-
Planned Start-Stop:	10/12 - 10/18
Actual Start-Stop:	-
Task Dependencies:	None.

2. Login	
2.1 Design	
Description:	Design the Login Page, define component styles.
Owner:	Brittney MacLennan (UI Designer)
Est. Effort:	2 hours
Actual Effort:	-
Planned Start-Stop:	10/12 - 10/16
Actual Start-Stop:	-
Task Dependencies:	None.
2.2 Data Access Layer	
Description:	Design and implement the Data Access Layer for use with testing the Login page.
Owner:	Tyler Wheaton (Database Designer)
Est. Effort:	5 hours
Actual Effort:	-
Planned Start-Stop:	10/12 - 10/20
Actual Start-Stop:	-
Task Dependencies:	None.
2.3 Write Unit Tests	
Description:	Write unit tests for UI methods.
Owner:	Evan Wike (Developer)
Est. Effort:	2 hours
Actual Effort:	-
Planned Start-Stop:	10/12 - 10/14
Actual Start-Stop:	-
Task Dependencies:	None.
2.4 Build	
Description:	Build the Login page, implement functionality.
Owner:	Evan Wike (Developer)
Est. Effort:	15 hours
Actual Effort:	-
Planned Start-Stop:	10/16 - 10/30
Actual Start-Stop:	-
Task Dependencies:	2.3 Write Unit Tests

3. Registration Page	
3.1 Design	
Description:	Design the Registration Page, define component styles.
Owner:	Brittney MacLennan (UI Designer)
Est. Effort:	3 hours
Actual Effort:	-
Planned Start-Stop:	10/14 - 10/16
Actual Start-Stop:	-
Task Dependencies:	None.
3.2 Write Unit Tests	
Description:	Write unit tests for Registration methods.
Owner:	Evan Wike (Developer)
Est. Effort:	5 hours
Actual Effort:	-
Planned Start-Stop:	10/16 - 10/18
Actual Start-Stop:	-
Task Dependencies:	None.
3.3 Build	
Description:	Build the Registration Page, implement functionality.
Owner:	Tyler Wheaton (Developer)
Est. Effort:	20 hours
Actual Effort:	-
Planned Start-Stop:	10/20 - 10/30
Actual Start-Stop:	-
Task Dependencies:	3.2 Write Unit Tests
3.4 Integration	
Description:	Connect Registration page to the Login page.
Owner:	Evan Wike (Architect)
Est. Effort:	4 hours
Actual Effort:	-
Planned Start-Stop:	10/25 - 10/30
Actual Start-Stop:	-
Task Dependencies:	3.3 Build
3.5 Integration Testing	
Description:	Write integration tests for the Login and Registration components.
Owner:	Brittney MacLennan (UI Designer)
Est. Effort:	3 hours
Actual Effort:	-
Planned Start-Stop:	8/25 - 8/30
Actual Start-Stop:	-
Task Dependencies:	3.4 Integration

4. Dashboard	
4.1 Design	
Description:	Design the Dashboard, define component styles.
Owner:	Brittney MacLennan (UI Designer)
Est. Effort:	3 hours
Actual Effort:	-
Planned Start-Stop:	10/25 - 10/30
Actual Start-Stop:	-
Task Dependencies:	None.
4.2 Integration	
Description:	Integrate Login component with the Dashboard.
Owner:	Evan Wike (Architect)
Est. Effort:	3 hours
Actual Effort:	-
Planned Start-Stop:	10/30 - 11/5
Actual Start-Stop:	-
Task Dependencies:	None.
4.3 Integration Testing	
Description:	Write Integration Tests for the Login and Dashboard componeonts.
Owner:	Evan Wike (Developer)
Est. Effort:	4 hours
Actual Effort:	-
Planned Start-Stop:	10/30 - 11/5
Actual Start-Stop:	-
Task Dependencies:	4.2 Integration

3.2 Release Plan

This section will contain a loose plan for the release schedule that we expect to follow over the course of this project. The dates reflected both in the release plan as well as the iteration plan may change if we are unable to finish goals within the previously specified time. However, if these dates are changed or if these goals are moved to other iterations, that change will be reflected in updated versions of the product plan.

- Create Basic UI
    - Create home page (11/02/2020)
        - Create notification summary (11/16/2020)
    - Create registration page (11/02/2020)
        - Password requirements: 8 characters, 1 uppercase letter, 1 symbol, 1 number (11/02/2020)

        - Verification email (11/16/2020)
        - Create account with google/facebook account (11/16/2020)
    - Create login page (11/02/2020)
        - Verify user exists (11/02/2020)
        - Allow password resets (11/16/2020)
    - Create transaction page (11/02/2020)
        - Allow adding transactions manually (11/16/2020)
        - Allow exporting of transactions (11/16/2020)
        - Sorting transactions by date (11/02/2020)
- Create database (11/02/2020)

3.3 Iteration Plans

This section covers past iterations and what was completed during them as well as what is planned for future iterations. Just as the release plan in section 3.2 is subject to change, the iteration plans can be changed in future versions of the project plan to reflect what was accomplished during each iteration.

Iteration #1 (ends 10/05):

- Project Charter
- Requirements Document
- Turned in Contract

Iteration #2 (ends 10/19):

- Project Plan
- Risk Management Report

Iteration #3 (11/02):

- Architecture Document
- Create home page
- Create registration page
    - Implement password requirements
- Create login page
    - Verify user exists in database before allowing them to login
- Create transaction page
    - Sort transactions by date
- Create database

Iteration #4 (ends 11/16):

- Create notification summary
- Implement verification emails

- Allow creation of accounts through Google/Facebook Accounts
- Allow password resets
- Allow transactions to be added manually
- Allow exporting of transactions
- Test Plan
- User and System Guide

Iteration #5 (ends 12/07):

- Gather all information and present to Commerce Bank

3.4 Budget

Based on the budget summary provided in section 1.5, we expect the project budget to be **$14,120.** This amount will be used as a ceiling value that we will try not to go above, and potentially be able to come in under.

4 Control Plan

4.1 Monitoring and Control

Progress tracking will be handled through the Discord server for the convenience of our means of communication. We will hold meetings toward the end of the week to see where we stand in the current iteration. By the end of the iteration, we evaluate on what was completed and how well we performed during the iteration.

4.2 Project Measurements

Phase	Measurement	Role
Release Planning	Record effort estimates for product features	Project Manager

Iteration Planning	Record effort estimates for scheduled tasks Update effort estimates for product features Update estimated dates in release plan	Project Manager
Iteration Closeout	Record actual effort for scheduled tasks Record actual effort for product features	All Roles
Project Closeout	Archive project performance data in process database. (See process database definition for a list of measures to record.)	Project Manager
Ongoing	Record defects found from integration testing. Assign each defect to one of the following categories: blocker, critical, major, minor or trivial. Keep track of the state of each defect: open, assigned, fixed, closed.	Manager/ Developer/ UI Developer/ Database Developer

5 Supporting Process Plans

5.1 Risk Management Plan

The risks that the team will likely come across are scope of knowledge of tools, running into crunch time, over or underestimating tasks.

Actions that we can take to minimize scope of knowledge of tools is learning what the tools do and how they're going to function. For tasks, it's deducing the subtasks that make up the task. Getting a headstart on our tasks is often good to mitigate crunch time so the team has more

time to fix other things that could be an issue. Should these become a problem, the team can continue what wasn't finished into the next iteration.

5.2 Configuration Management Plan

Configuration management plans for this document and other baselined work products including review procedures and change management procedures.

1. All work products will be stored in a Github repository running on a central server.

2. The naming convention for documents will be: DocName-v#.suffix where DocName is the name of the document, v# denotes the version number of the document and 'suffix' is the standard/normal suffix for the document type. For example, the second version of the requirements document created as a Microsoft Word document might be labeled: Requiremets-v2.docx.

3. All project (work products) items (documents, source code, test cases, program data, test data, etc) will be stored in the Github repository but not all will be under change control (subject to formal change control procedures.) Only the system requirements, project plan and source code will be baselined and under configuration control.

4. Items that are subject to change control will be considered baselined after a group review at the end of the life cycle phase during which they are created. Baselined here means that the product has undergone a formal review and can only be changed through the prescribed change control procedures.

5. The change control procedure once a product is baselined is when someone that wants to make a change to an item would reach out to the person in the group that was working on that part of the project, propose the change that you want to make, and reach an agreement on what improves the original.

6. A version history to record the various changes of documents and source code for project progress and other purposes such as troubleshooting, falling back on an older revision, etc. The version includes either what was changed and/or what was fixed.

5.3 Verification and Validation Plan

During the iterations, each module being created will be tested by the developer creating it as well as other members of the team to verify that it functions as it should. Using the password requirement as an example, both the developer in charge of that module as well as the other team members will attempt to create passwords outside of the specified rules to see if it will be

allowed. Should a member of the team besides the developer find an issue with a module during the iteration, the error will be logged in the “errors” channel in our communication channel.

5.4 Product Acceptance Plan

For each individual project module to be considered acceptable, they must function as intended by the time the project is handed over at the end of the semester.

Should a previous requirement prove unobtainable before its iteration deadline, the requirement will be reevaluated and moved to another iteration if necessary.

If a stretch goal is unable to be implemented before the project deadline, it will be reevaluated and the team will decide to continue and attempt to add it to the project or if it should be dropped entirely.