**Software Project Management Plan**

**Commerce Bank Budgeting Tool**

28 February 2017

**Team Members**

Jon Whipple

Jeremiah Trahern

Derek Wilson

Alex Sweeney

Colin Zeiler

Daniel McNary

Emily Thomas

Jeremy Troshynski

Document Control

**Change History**

|  |  |  |
| --- | --- | --- |
| **Change Date** | **Editor** | **Description of changes** |
| 2/28/2017 | Jeremy Troshynski | Initial creation, various small edits |
| 2/28/2017 | Derek Wilson | Communications, Assumptions/Constraints, Budget, Some definitions, Evolution |
| 3/1/2017 | Emily Thomas | Purpose and Scope, Goals and Objectives, random edits |
| 3/1/2017 | Jeremiah Trahern | Technical Process, Configuration Management Plan, Verification and Validation Plan, Risk Management Plan, Project Deliverables, some Definitions |
| 3/1/2017 | Colin Zeiler | Management and Control, Schedule and Budget Summary, Success Criteria, Iteration Plan, Various edits |
| 3/1/2017 | Alex Sweeney | Product Acceptance Plan |
| 3/1/2017 | Jeremy Troshynski | Editing and Reviewing, Release Plan |
| 3/1/2017 | Jonathan Whipple | Added Gantt chart, reformatted document, last preparation for upload to GitHub |

**Document Storage**

This document is stored in the project’s Google Drive Folder:

<https://docs.google.com/document/d/1cwZEG6r4sBdu7sCEwDKyK7i9UndMhHx5rSU1SCUPNCI/edit?usp=sharing>

As well as on GitHub:

<https://github.com/cs451/Spring2017Team6>

**Document Owner**

All team members are responsible for developing and maintaining this document.

**Table of Contents**

[1 Overview 4](#_Toc476166452)

[1.1 Purpose and Scope 4](#_Toc476166453)

[1.2 Goals and Objectives 4](#_Toc476166454)

[1.3 Project Deliverables 4](#_Toc476166455)

[1.4 Assumptions and Constraints 5](#_Toc476166456)

[1.5 Schedule and Budget Summary 6](#_Toc476166457)

[1.6 Success Criteria 7](#_Toc476166458)

[1.7 Definitions 7](#_Toc476166459)

[1.8 Evolution of the Project Plan 7](#_Toc476166460)

[2 Startup Plan 9](#_Toc476166461)

[2.1 Team Organization 9](#_Toc476166462)

[2.2 Project Communications 9](#_Toc476166463)

[2.3 Technical Process 10](#_Toc476166464)

[2.4 Tools 10](#_Toc476166465)

[3 Work Plan 11](#_Toc476166466)

[3.1 Activities and Tasks 11](#_Toc476166467)

[3.2 Release Plan 11](#_Toc476166468)

[3.3 Iteration Plans 11](#_Toc476166469)

[3.4 Budget 11](#_Toc476166470)

[4 Control Plan 13](#_Toc476166471)

[4.1 Monitoring and Control 13](#_Toc476166472)

[4.2 Project Measurements 13](#_Toc476166473)

[5 Supporting Process Plans 14](#_Toc476166474)

[5.1 Risk Management Plan 14](#_Toc476166475)

[5.2 Configuration Management Plan 14](#_Toc476166476)

[5.3 Verification and Validation Plan 15](#_Toc476166477)

[5.4 Product Acceptance Plan 15](#_Toc476166478)

# Overview

## Purpose and Scope

Our team is creating a gamified web application for the Commerce Bank website that Commerce Bank users can access through their online account to set multiple savings or budget goals. Achieving a goal earns them points that they can show off to friends or use to purchase virtual rewards.

The application will provide visuals of a user’s budget plans, with divided subsections for budgets that have caps on categories of expenses. It will display lists of savings plans, transactions within specified time periods, spending trends within specified time periods, and total points earned throughout the duration of the user’s account.

The application can be used on any online device, as it will be provided through the Commerce Bank website. It will use Commerce Bank’s SQL database provided on the website server.

The user interface will be on a dashboard, where all elements of the application are on display and where the user can toggle the display options. 90% of new users should navigate the application with ease without referring to any instructions, although both user-side and administrative-side documentation will be provided.

## Goals and Objectives

Project goals:

1. Develop an application that meets all the core requirements.
2. Learn more about writing programs with frameworks like ASP.NET and Entity.
3. Alleviate the headaches of finances for the users.

Project objectives:

1. Develop an application whose dashboard can quickly be picked up and navigated.
2. Make the code easy to understand so the Commerce Bank developers have an easier time integrating the application into their system.
3. Make the points system flexible and scalable so that incremental features can be added without delay.

## Project Deliverables

The following items will be delivered to the customer on or before 5/1/2017:

1. All source code for the system
2. User’s guide
3. All documentation
4. System test cases
5. All plans completed by the deadline

## Assumptions and Constraints

Assumptions:

1. Security is not within the scope of the project.
2. The team’s proxy server can easily be replaced with the Commerce Bank online server.

Constraints:

1. The software must be able to integrate with Commerce Bank’s current online-banking system.
   1. The software must be built using the .Net Framework.
   2. The database must be implemented using SQL Server 2012.
2. Our team can use whichever client-side frameworks/libraries we see fit, but they must be included in the project.
3. The software must be turned in by 5/1/2017 - no major changes to the code are allowed, only bug fixes - and ready to be presented on 5/5/2017.

## Schedule and Budget Summary

|  |  |  |
| --- | --- | --- |
| Schedule Item | Start Date | End Date |
| Iteration 1 | 2/20/2017 | 3/5/2017 |
| Technical Prototype | 2/20/217 | 2/27/2017 |
| Iteration 2 | 3/6/2017 | 3/19/2017 |
| Customer Approved UI Prototype | 2/20/2017 | 3/8/2017 |
| Iteration 3 | 3/20/2017 | 4/9/2017 |
| Iteration 4 | 4/10/2017 | 4/23/2017 |
| Usability Testing | 3/20/2017 | 4/19/2017 |
| Iteration 5 | 4/24/2017 | 5/1/2017 |

We have no monetary budget as we are not being paid to develop this software. We do, however, have a time budget in that we must have all aspects of the software finished by 5/1/2017.

## Success Criteria

The software must be developed using the .NET framework, and compatible with SQL Server 2012.

The software must be able to display user transaction data in a chart.

The software must allow for user goals to be created, tracked, removed, and completed.

The software must have a system implemented that allows users to receive points for meeting budget goals.

The software must be responsive on desktops and smaller screens - tablets, mobile phones, etc.

## Definitions

ER diagrams - The visual representation of the database

Server - Machine that user computers get their application resources from

Stories - Tasks which are meant to be completed, or worked on, during a particular iteration.

Story points - The measure of effort relative to other stories in this project

UI - User interface, the screens in which users of the product interact with

## Evolution of the Project Plan

Before the start of an iteration, the project plan will be updated to include a schedule of detailed tasks for the upcoming iteration. At the conclusion of an iteration, the project plan will be updated to include the actual effort for each completed task.

Risk mitigation efforts will be evaluated at the start of each iteration. Severe risks will be analyzed and added to the project plan as soon as they materialize.

For Iteration 1, our tasks consist of:

* Everyone learning the .Net Framework
* Writing the ER diagrams and materializing the schema for the database
* Making sure connection to the server is achieved
* Writing code for making the tables in the database
* Creating the home page UI
* Creating the graphs for the home page of the UI
* Integrating the data from the database into the graphs for the home page

Our biggest complication we were faced with for the first couple weeks was trying to connect to the server. We had a plan set to store the data locally on all of our machines until connection could be established by all parties.

# Startup Plan

## Team Organization

Project Manager: (Alex) The project manager is responsible for managing risks, running the weekly team meeting, providing monthly status reports to senior management, and communicating with the client.

Database Team (4): (Colin, Derek, Daniel, Emily) The database team is responsible for creating, maintaining, and adding functionality to the network database. They’re also expected to participate in document creation and team meetings.

Frontend Team (4): (Jeremy, Alex, Jon, Jeremiah) The frontend team is responsible for creating a functional web application UI. They’re also expected to participate in document creation and team meetings.

## Project Communications

Our team meets every Friday to discuss what we have done in the week before and what needs to be done before the next meeting. Sub-team meetings are called if the sub-team feels that meeting face-to-face is necessary. Our team has also set up multiple team pages across the web.

We have a Slack team page where communication between all members and sub-team members happens. We use it to plan meetings, to discuss ideas or concerns that arise when not in team meetings, and to collaborate on aspects of the project.

We also have a Google Drive team page where members post and edit project documents collectively. It also contains a list of tasks, with specific tasks delegated to each team member, that need to be accomplished within the week between meetings.

A Trello board has been set up as well. It features our product backlog, our current sprint plan, and the status of each task within the sprint plan. The Trello board gives us a good visual representation of the progress of the current iteration, as well as the project as a whole.

## Technical Process

The team will follow an agile development process.

The project is being developed in iterations, which last 2 weeks. The team will communicate what stories we should focus on as we prepare and start each iteration. Mid-iteration status updates will be given through slack and during Friday team meetings, as well as merging to the GitHub master branch whenever a major story is completed.

Documentation will be created in parallel with code, during each iteration according to the deadlines specified in the project calendar. Before completing each phase, team members will meet to discuss their progress, and any issues they are having (this will most likely be during a Friday meeting).

Major milestones for each iteration:

* **Iteration 1:** technical prototype will be made, testing the technology stack as well as team cooperation.
* **Iteration 2:** customer approved UI prototype will be completed.
* **Iteration 3:** NONE
* **Iteration 4:** usability test.
* **Iteration 5:** system documentation completed.

## Tools

* **Programming Language** – C#, HTML/CSS, JavaScript
* **Version Control** – source code and written artifacts will be stored in a GitHub repository.
* **Build tools** – local and main builds will be done using Visual Studio.

# Work Plan

## Activities and Tasks

The following attributes will be recorded for each task:

* Task name
* Task Description
* Owner
* Effort estimate
* Actual effort
* Planned start and stop dates
* Actual start and stop dates
* Dependencies among other tasks

## Release Plan

|  |  |  |
| --- | --- | --- |
| Schedule Item | Internal/External | Submission Date |
| Technical Prototype | Internal | 2/27/2017 |
| UI Prototype | External | 3/8/2017 |
| Usability Test Report | Internal | 4/19/2017 |
| User Guide | Internal | 4/26/2017 |
| Product Release | External | 5/1/2017 |

## Iteration Plans

An iteration plan is a short-term fine-grained plan that shows the tasks to be completed during an iteration.

Iteration 1:

* Create and populate the database with user information.
* Create queries to extract necessary budget information from the database
* Create database abstraction layer to convert query information to JSON
* Create basic user interface to display data

## Budget

Since we are a team comprised of college students, we will not be getting paid for this project, so we don’t have a monetary budget. We do, however, have a time budget. The project is to be finished by 5/1/2017 and presented on 5/5/2017.

2/28/2017

So far we have spent a few hours working to complete all of the project documents needed: the project charter, product backlog, use cases, and project management plan. The database team has spent a handful of hours working on creating our server, making sure that we can connect to the server, conceptualizing and building the database, and populating the tables with data. The front-end team worked on making the UI for the home page.

# Control Plan

## Monitoring and Control

Weekly – Team meeting. Project participants report status, progress and potential problems.

2/27/2017 – Technical prototype due. Technical risks should be addressed and resolved.

3/8/2017 – Customer approved UI Prototype due.

3/17/2017 – Sprint Review with Commerce.

4/26/2017 – User Guide and System Documentation due.

5/1/2017 – Final Product due.

## Project Measurements

|  |  |  |
| --- | --- | --- |
| Phase | Measurement | Source |
| Release Planning | Record effort estimates for product features | PM |
| Iteration Planning | Record effort estimates for scheduled tasks  Update effort estimates for product features  Update estimated dates in release plan | PM |
| Iteration Closeout | Record actual effort for scheduled tasks  Record actual effort for product features  Record LOC count for modules written | PM & Team |
| System Test | Record the rate at which errors are found. | QA |
| Project Closeout | Archive project performance data in process database. (See process database definition for a list of measures to record.) | DB Team |
| Ongoing | Record defects found from integration testing through first year of release.  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. | Team |

# Supporting Process Plans

## Risk Management Plan

Major risks during this project:

1. Problems with team cooperation with respect to code maintenance, version control, and meeting deadlines according to the rest of the team’s expectations. This could result in halting development, falling behind, and reverting the project to an older state. In order to avoid this, team members should communicate their concerns, or any problems in getting their work done as soon as possible, and team meetings will be held to get an idea of everyone’s progress.
2. Merging back to the master branch causes too much trouble, with too many commits being made by different people. Some methods to reduce this risk is to integrate often, and make sure changes one makes do not go too long without updating their own branch with other member’s commits. Should the state of the repository become so altered that it cannot be easily debugged, a new project might be created, and pieced together by a few team members with all current changes.
3. Commerce Bank changes requirements, or requirements are not understood correctly. Communication with Commerce is important to mitigate this risk, and this will mostly be done during the demos, and through prototypes. If questions arise about how something should be implemented, emailing Commerce with those questions should make it easier to follow their requirements. If they outright change their requirements, a contingency plan will be made then, according to how the team feels.

## Configuration Management Plan

1. All work products will be stored in a shared GitHub repository.
2. Documents and Code will be kept in separate folders on the repository, under the folders *Project Documents* and *Budgeting Application*. All items in the repository are subject to change unless it is agreed upon by the group that an item is complete and should not be changed. Should this happen, the item will be marked and kept on someone’s local machine in case someone changes the version in the repository.
3. If someone needs to change a baselined document or piece of code, they will message everyone through Slack, and get approval by all those who worked on that document/code, as well as the project manager. They will need to explain the reasons for the change, and what their changes will do to improve/fix the piece in question. If anyone does object to the change, the reason for objecting will be discussed at a meeting where everyone is invited to attend and voice their opinion. At the end of the meeting a democratic vote will be held to decide whether or not the change should be allowed.
4. A change history on major documents will be kept, allowing for individuals to show their contributions and they will specify on which day their contributions were made. For everything else, the change history will just be given through GitHub commits.

## Verification and Validation Plan

Every team member is expected to contribute to quality assurance for both code and documents. Everyone participating in the project should understand all of the code, even if it is not the responsibility of their respective team. In order to validate code before major milestones and demos, test cases will be drawn up for each module being integrated, and the teams will do their best to create as many useful test cases in order to assure demos run smoothly, and each component remains modular. More extensive testing and quality assurance is expected should this project be integrated as a product for Commerce Bank’s website, so our focus will be on making sure the code is functional, efficient, and provides readable/visually appealing results.

## Product Acceptance Plan

Acceptable project performance will at a minimum be considered a passing grade for the Software Capstone class. In the best case scenario Commerce Bank will accept our system and work on implementing it. In order for these goals to occur the system shall satisfy our project goals in function, performance, and design. It must run correctly according to our project guidelines and result in no errors. It must pass the validation plan, wherein all test cases are run smoothly.