Software Project Management Plan

Financial Budgeting App: ClariFI

9/26/2025

Team Members

Abdul Mohammed
Haley Nilsen
Kaleb Flores Lopez
Sid Nsude

Document Control

Change History

Revision	Change Date	Description of changes
V1.0	9/26/2025	Initial release

Document Storage

This document is stored in the project's repository at:
https://github.com/snsude/Commerce-Bank-Financial-App/tree/main

Document Owner

Abdul Mohammed is responsible for developing and maintaining this document.

Table of Contents

1 Overview

1.1 Purpose and Scope

ClariFi is a financial banking application that empowers users to take control of their financial future through goal-oriented tracking and AI assistance. This web-based application is being developed as part of UMKC's Software Engineering Capstone class in partnership with Commerce Bank. ClarFi will serve as a modern financial management platform that bridges traditional banking services and personalized financial planning, offering users an intuitive interface to monitor their progress toward achieving their financial aspirations. The project leverages artificial intelligence integration and responsive web design to deliver a secure, user-friendly experience that meets the evolving needs of today's digital budgeting customers.

The purpose of this project is to streamline personal financial management and goal achievement by providing users with a centralized platform that transforms complex financial data into actionable insights and clear progress indicators. ClariFi actively supports users in reaching their financial objectives through intelligent monitoring and AI-powered assistance. This project aims to enhance user engagement with their financial health while reducing the complexity and friction typically associated with personal finance management.

The scope of this project compasses the complete design, development, testing, and deployment of a financial web application for Commerce Bank that will serve as a comprehensive financial goal-tracking platform. The application will provide secure user authentication and account management capabilities, robust budget tracking functionality, and sophisticated data visualization tools that present financial information in clear, actionable formats. A key component of the scope includes the integration of an AI-powered chatbot that will offer personalized assistance, financial insights, and recommendations to enhance the user experience. Additionally, the application will feature a comprehensive tutorial system designed to onboard new users and ensure they can effectively utilize all available features. The project scope explicitly does not include mobile app development for iOS or Android platforms, integration of third-party plugins beyond pre-approved libraries and frameworks, or support for international customers and their specific regulatory requirements at this initial stage of development.

1.2 Goals and Objectives

Goals and objectives define expected project outcomes. Goals are broad and inspirational. Objectives are narrow and measurable.

Project goals generally related project outcomes to business objectives (reduced cost, increased revenue, improved quality, etc).

A well-worded objective is SMART: Specific, Measurable, Attainable/Achievable, Realistic and Time-bound.

Partial Example

Project goals:
1.

Project objectives:
1.

1.3 Project Deliverables

This section lists the outputs of the project that are delivered to the customer.

Partial Example

The following items will be delivered to the customer on or before 12/12/2025:

1. Source Code – Completed and well documented codebase, for frontend, backend and database all hosted on a GitHub repository.
2. Technical Documentation – Documents like the architecture/design, test report, requirements, test plan, and project charter all available in the GitHub repository.
3. User Documentation & Tutorial – The user guide document along with a mp4 file showcasing the project and it's features.
4. Final Report & Presentation – A detailed report about the project's design, implementations, challenges and outcomes along with a presentation for a live demo.

1.4 Assumptions and Constraints

Assumptions are conditions, usually outside the control of the project team, that are taken for granted. Project plans (i.e. estimates) typically depend on certain assumptions being true. Assumptions that turn out to be false, may jeopardize project success. In order to reduce project risk, the project manager may elect to validate certain assumptions as part of the risk management process.

This is also a good place to document verbal promises or assurances given to you.

Constraints are limits or restrictions on freedom. Projects may have technical as well as non-technical constraints. Priorities for schedule and budget can impose non-technical constraints on a project. Restrictions on programming language or

delivery platform are examples of technical constraints that limit design and implementation options.

Partial Example

Assumptions:

1. **The user has a working internet connection**
2. **The user has a working laptop or desktop with a functioning browser**
3. **The user has a valid email address**

Constraints:

1. **Cannot use CMS tool**
2. **Must use a CSS framework**
3. **Must complete two stretch goals**
4. **Must have an LLM integration within the web application**
5. **Have till the end of the semester to complete**
6. **Have a $0 budget**

Note, the following is not a reasonable assumption for inclusion in this section: "We assume that our group has the necessary skills and knowledge needed to complete the project." This might be something you are taking for granted, but it is not something worth documenting in the project plan. The assumptions you want to list here are those that are outside your control. Once the development team is established, it is their responsibility to possess or develop the skills and knowledge needed to complete the project. If there is a concern that the existing team doesn't have the skills and knowledge needed to complete the project successfully, add it as a risk and develop a plan for mitigating the risk.

1.5 Schedule and Budget Summary

The schedule summary shows start and end dates for high-level activities ending in major milestones or deliverables. Milestones are major events in the project life cycle that are used to measure progress.

A Gantt chart is an excellent tool for visualizing the start and stop dates of major scheduled activities.

(Temporary) - Schedule Summary

- **Week 1–2: Project Planning & Figma Designs**
  - Requirement analysis, design wireframe, draft documents, database schema, develop skeleton pages, research ai agent implementations, research data visualization tools,
- **Week 3–5: Core Development – Sprint 1**
  - Start developing frontend, have database set up, llm/ai agent environment ready to go, start agent development, data visualization tool decided, complete user authentication, basic dashboard ready, start connecting backend, start creating automated tests

- **Week 6–8: Core Development – Sprint 2**
    - Implement goal tracking feature, integrate ai agent, start testing app functionality and continue writing automated tests
- **Week 8–10: Core Development – Sprint 3**
    - Bug fixing & testing
- **Week 11-12: Polish & Present**
    - Deploy finalized app, finalize documentation, rerun tests, prepare presentation and report

The budget summary shows total project cost, possibly broken down into separate categories for such things as salaries, equipment, travel, overhead, etc.

Budget Summary

- **Software & Tools = $0**
- **Personnel = $0**

1.6 Success Criteria

Success criteria spell out what has to happen before the project can be considered a success. Having explicit success criteria serve two purposes. First, during a project success criteria help to focus attention on what is important. Second, at the conclusion of a project (project closure) success criteria are used to assess whether or not the goals and objectives of the project have been achieved.

To be effective in both of these endeavors, success criteria must be defined in a way that is both quantifiable and verifiable.

For more advise on how to define the success criteria for a project, I recommend: *Success Criteria Breed Success*, by Karl Wiegers. It is available on the web.

Partial Example

- **Project Completion:** the final deliverable is submitted on/before scheduled date
- **User Experience:** The app is easy to use, doesn't have a huge learning curve, and provides accessibility to all types of users
- **Requirements: Met** All of the functional/ non-functional requirements are met
- **Agent Performance:** AI agent is able to perform its duties to accurately
- **Critiqued Features Updated:** After the mid semester meeting with Commerce bank, the critiqued features are updated to ensure that the standards are met

1.7 Definitions

This section should define potentially unfamiliar or ambiguous words, acronyms and abbreviations.

- **LLM** : a type of artificial intelligence trained on vast amount of text data to generate human- like language and answer questions
- **FastAPI** : a high performance web framework for building APIs with Python
- **PostgreSQL** : an open source object-relational database system
- **React** : a JS library for building user interfaces, specifically single-page applications
- **Tailwind CSS** : a CSS framework for rapidly building custom user designs

1.8 Evolution of the Project Plan

This section describes plans for updating the project plan throughout the project.

Partial Example

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.

2 Startup Plan

2.1 Team Organization

This section explains project roles and the authorities and responsibilities associated with these roles. Lines of communication, authority and reporting relationships are often shown with an org chart. If development team is known, actual names can be associated with roles.

Partial Example

Project Manager:	The project manager is responsible for creating the project plan (with input from those doing the work), managing risks, running the weekly team meeting and providing monthly status reports to senior management.
Programmers (3):	Programmers are primary responsible for coding and unit testing modules. They are also expected to take part in architecture planning and review meetings.

2.2 Project Communications

This section contains the project communications plan. The communications plan describes how information is gathered and distributed.

- **Internal Team Communcation:** communication will take place through Microsoft Teams, we will have a weekly meeting to discuss /provide updates on progress

- **Jira Dashboard:** will have tasks assigned to each team member

2.3 Technical Process

This section describes the software development methodology or conventions the team agrees to live by. When following an organization standard process, this section will refer to the standard process and state any deviations that are planned for this project. In the absence of an organization standard process, this section will define planned phases, entry and exit criteria for each phase, major milestones, workflows, and other aspects of the proposed development process.

- **Sprints**
- **Sprint Reviews**
- **Version Control**
- **Weekly Meetings**

2.4 Tools

This section specifies the development tools the team will be using to perform their work.

Partial Example

- **Version Control**: GitHub
- **Frontend**: React, Tailwind CSS
- **Backend**: FastAPI, Python
- **Database**: PostgreSQL
- **LLM API**: Hugging Face Inference API
- **Communication:** Microsoft Teams
- **Design & Prototyping**: Figma
- **Automated testing** – unit tests will be implemented with the JUnit testing framework.

3 Work Plan

3.1 Activities and Tasks

A work breakdown structure is an excellent tool for identifying a complete list of tasks.

Depending on the needs of the project, some or all of 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

3.2 Release Plan

For day-to-day project management the release and iteration plans (described in the next section) are probably the two most important project management artifacts.

The release plan lists expected completion dates for major milestones and delivery dates of key work products. The project's technical development process to a certain extent will dictate the choice and timing of milestones and deliverables. For example, projects following the Rational Unified Process will have four major milestones: life-cycle objectives, life-cycle architecture, initial operation capability, and product release.

3.3 Iteration Plans

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

3.4 Budget

The project budget is the projected cost of the project as a function of time. At any point in the project you should be able to say how much money has already been spent and estimate of the amount of money needed to complete the project.

4 Control Plan

4.1 Monitoring and Control

Include in this section plans and procedures for tracking progress and controlling performance. Included here will be the approximate dates of technical as well as managerial reviews. Typically each major milestone or project phase will end in a review.

For projects that don't have a separate communication plan, this section may also include information on the timing and content of status reports and other project review documentation.

Partial Example

Weekly	–	Team meeting. Project participants report status, progress and potential problems.
3/1/2008	–	Critical Design Review. Formal inspection of product architecture.
5/15/2008	–	Executive Review. The project manager presents current project status to project sponsor and senior executives.

4.2 Project Measurements

Product and process measures support project management and estimation by analogy. At the beginning of a project, estimates are made for product size, project cost and delivery dates. During a project, progress is tracked with measures of actual effort, integrated lines of code and actual expenditures. Keeping track of

estimates and actuals during a project helps to calibrate whatever technique is being used to make estimates. Storing project performance data on completed projects provides a rich source of data for estimating future projects.

Example

Phase	Measurement	Source
Release Planning	Record effort estimates for product features	Mgr
Iteration Planning	Record effort estimates for scheduled tasks Update effort estimates for product features Update estimated dates in release plan	Mgr
Iteration Closeout	Record actual effort for scheduled tasks Record actual effort for product features Record LOC count for modules written	Mgr/Pgr
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.)	Mgr
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.	Mgr/Pgr/QA

5 Supporting Process Plans

5.1 Risk Management Plan

Identify technical and managerial risks. Prioritize risks. Consider the probability of each risk turning into a problem and the likely consequences. For the highest priority risks, what actions will be taken to minimize the probability of the risk turning into a problem and the resulting consequences? What are the contingency plans for selected risks that do become a problem? Identify processes for monitoring risks and updating the risk management plan.

5.2 Configuration Management Plan

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

Partial Example

1. All work products will be stored in a centralized CVS repository running on a central server.
2. The naming convention for documents will be: NNN-VVV.suffix where NNN is a mnemonic that reflects the function of the document, VVV is a 3 digit version number, 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: REQ-002.doc.
3. All project (work products) items (documents, source code, test cases, program data, test data, etc) will be stored in the CVS 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: (1) anyone wanting to make a change to a baselined item sends an email to the rest of the group describing the change, reason for the change, expected impact, and timeline for integrating the change. (2) if no one responds to the group within 2 days with a reason for why the change request shouldn't be permitted, it will be considered accepted and the person proposing the change may proceed with the change. 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.
6. Including a change history with all documents is encouraged but only required for baselined documents. The change history should be at the front of the work item and include: (1) the name of the person making the change, (2) brief description of what has changed, (3) reason for the change, and (4) the date the change was integrated.

5.3 Verification and Validation Plan

The verification and validation plan defines what actions are being taken to assure the quality of the development process and resulting software products.

Partial Example

5.4 Product Acceptance Plan

The product acceptance plan defines what is acceptable in terms of product quality and product functionality. Acceptance criteria should be objective and measurable. Note product success is one aspect of project success. Teams wanting to establish a clear understanding of what will be considered acceptable project performance may

want to define a more general plan for project success that includes quantitative goals for delivery date, cost, etc.

Partial Example