Brock University

Computer Science Department
COSC 4P02 – Software Engineering 2

Release Planning Document

Team Members
Anthony Colosimo - ac19fa@brocku.ca
Jacob Drobena - jd20ym@brocku.ca
Rouvin Rebello - rr20jk@brocku.ca
Sangmitra Madhusudan - smadhusudan@brocku.ca
Skye Reid - ug20lq@brocku.ca
Tyler McDonald - tmcdonald3@brocku.ca

Professor: Dr. Naser Ezzati-Jivan

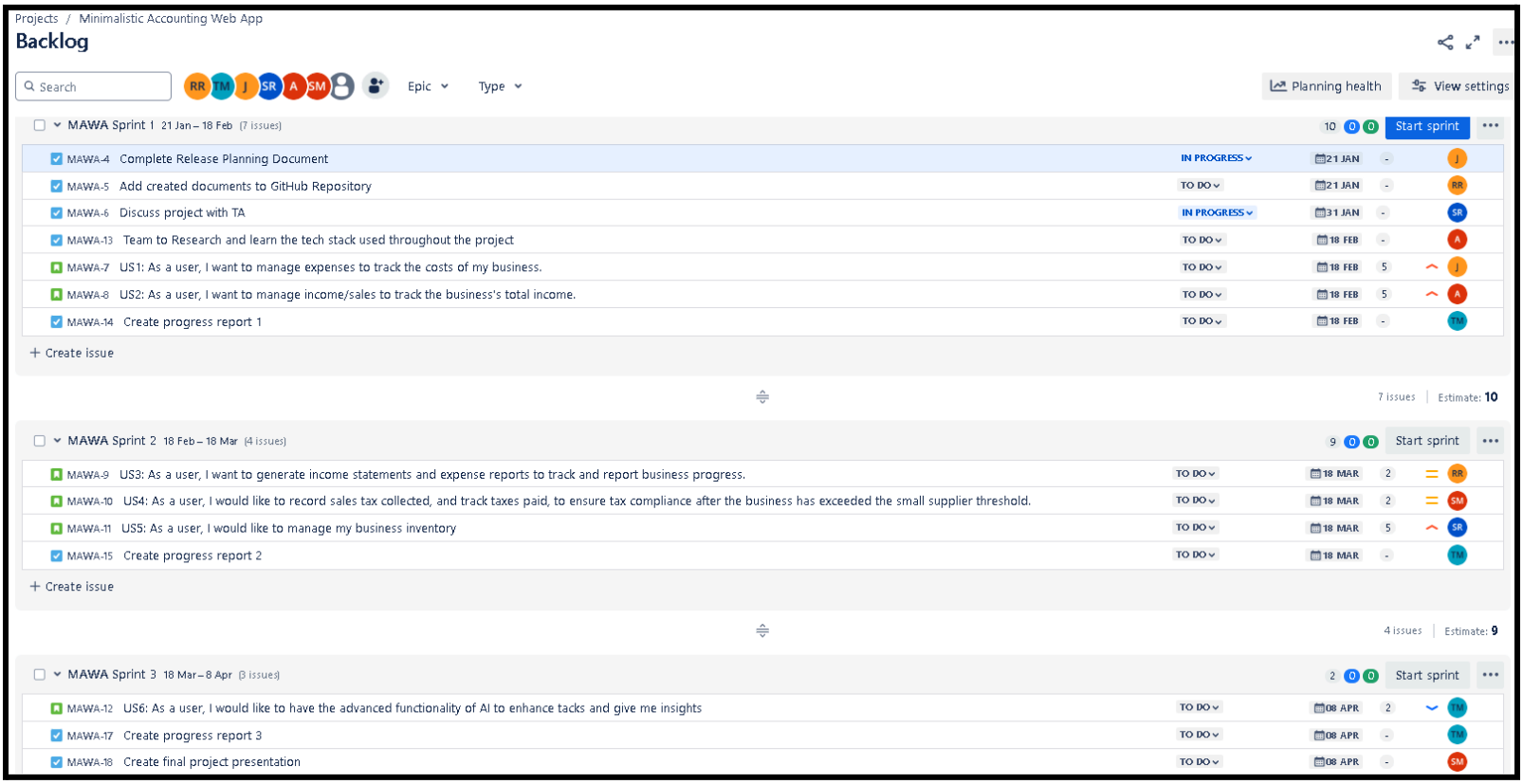
Sunday, 19th January 2025

Introduction

Our team aims to design a minimalistic accounting web application that offers small business owners a streamlined and affordable accounting solution. This release plan outlines the project backlog, user stories, and the required research needed during development. For scheduling, our product planning will organize our progress using our Jira Board, which will contain user stories and release information. We have structured sprints using agile development to focus on incremental feature design while ensuring quality and reliability at each step. This app will reduce the overhead experienced through complex software and empower small businesses to manage their finances efficiently. Our team has prioritized simplicity, accessibility, and scalability for the initial development steps of our product, ensuring that it effectively meets the needs of our end users.

Project Backlog

The product backlog is a comprehensive repository of every user story spanning the entire project. Our product backlog is visible below:



User Story Overviews

The user stories highlight the core functionalities and features critical to delivering an all-in-one accounting application for small business owners. These specific stories provide an outline for the development process, ensuring that the app meets the intended deployment goals.
While these stories highlight the foundation of our app, *the timing and implementation details are subject to change as the project evolves.* This is to accommodate supervisor feedback, sprint retrospectives, and to allow for changes during agile development.

- **User Story 1:** As a user, I want to manage expenses to track the costs of my business.
    - This involves:
        - Adding expenses (individually and in bulk)
            - Adding product expenses should require the user to type in an item number that will add to their inventory amount
        - Removing expenses
        - Editing expenses
        - Viewing all expenses
        - Categorizing Expenses
    - Tasks:
        - Task 1: Create a database that holds the expenses
        - Task 2: Create a UI that handles CRUD operations for the expenses database
        - Task 3: Test this user story and associated classes with Jest along with manual testing
    - Members:
        - Front end - Anthony, Sangmitra, Skye
        - Back end - Tyler
    - Story points: 5
    - Priority: High (necessary for minimum viable application)
- **User Story 2:** As a user, I want to manage income/sales to track the business's total income.
    - This involves:
        - Adding income as sales
            - Sales should deduct from the inventory of the item
        - Adding income arbitrarily
        - Removing income/sales
        - Editing income transactions

        - Viewing all income transactions
        - Categorizing Income
    - Tasks:
        - Task 1: Create a database that holds sales (tied to inventory)
        - Task 2: Create a UI that handles CRUD operations for the income + inventory database
        - Task 3: Test this user story and associated classes with Jest along with manual testing
    - Members:
        - Front end - Anthony, Sangmitra, Skye
        - Back end - Tyler, Rouvin
    - Story points: 5
    - Priority: High
- **User Story 3:** As a user, I want to generate income statements and expense reports to track and report business progress.
    - This involves:
        - Generating income statements for the specified duration
        - Generating expense reports for the specified duration
        - Creating visuals for income and expenses in the dashboard
    - Tasks:
        - Task 1: Create a UI feature where users can select the type of report (income or expense) and specify the period for the same.
        - Task 2: Develop back-end functionality using endpoints to handle requests for generating requested reports and logic to fetch the data from the database.
        - Task 3: Utilize a library to format and generate reports and graphs from retrieved information.
        - Task 4: Test the user story using Jest
    - Members:
        - Front end - Anthony, Sangmitra, Skye
        - Back end - Tyler
    - Story points: 2
    - Priority: High/Medium
- **User Story 4:** As a user, I would like to record sales tax collected, and track taxes paid, to ensure tax compliance after the business has exceeded the small supplier threshold.
    - This involves:
        - Allowing the user to set their sales tax rate
        - Add text fields for adding income/expense sales tax
        - Sales tax calculation

        - Generating Sales tax report
    - Tasks:
        - Task 1: Create a form to input sales tax that is sent to income and expense databases
        - Task 2: Develop back-end functionality using endpoints to handle logic to fetch sales tax rate, income, and expense data from the database after which sales tax can be calculated
        - Task 3: Utilize a library to format and generate reports from retrieved information
    - Members:
        - Front end - Skye, Anthony, Sangmitra (primarily for visualizations)
        - Back end - Rouvin, Tyler
    - Story points: 3
    - Priority: Medium
- **User Story 5:** As a user, I would like to manage my business inventory
    - This involves:
        - Adding an item (item #, Name of item, Cost, Sales Tax, Amount in Stock)
        - Deducting from an inventory item when a product is sold (in the income section)
        - Adding an inventory item when a product is purchased for the business (in the expense section)
        - Generate inventory report (amount of each item, total value of each item, total assets in stock)
    - Tasks:
        - Task 1: Create a database for the inventory
        - Task 2: Attach/integrate with the income and expense databases
        - Task 3: Use Jest and manual testing to ensure the integration works correctly with income/expense databases
    - Members:
        - Front end: Rouvin, Skye
        - Back end: Tyler, Jacob
    - Story points: 5
    - Priority: High/Medium
- **User Story 6:** As a user, I would like to have the advanced functionality of AI to give me insights about my business.
    - This involves: (pro user feature only)
        - Creating roles for pro users so that only they gain access to this feature.
        - Integrating either:

                - Commercial LLM APIs (OpenAI, Anthropic, etc.)
                - On-premises LLMs with locally offered APIs (open-source LMs such as Llama/Phi)
            - Creating a chat window/panel with pre-written prompts to enable discussion with the LLM.
            - Creating & managing an additional setting to toggle the presence of AI insights.
            - Creating easy-access buttons to generate view-specific insights (reports, inventory, etc.).
    - Tasks:
        - Task 1: Implement the external API of our choosing, while experimenting with prompt engineering to create appropriate responses.
        - Task 2: Implement the chat/insights interface as well as the per-page buttons to generate quick responses.
        - Task 3: Create a toggle for this feature while gating it behind Pro access.
        - Task 4: Further testing to ensure the model remains fair and appropriate to all users in all situations.
    - Members:
        - Front end + Back end: Anthony, Rouvin, Tyler
    - Story points: 2
    - Priority: Low

Sprint Summary

- **Sprint One [Jan 21st - Feb 18th]:** The goal of this sprint is to develop a basic implementation of the application. This includes a database and UI for simple expense and income management, correlating to the implementation of User Stories 1 and 2.
    - Tasks for this sprint:
        - Research and learn the tech stack necessary for this implementation
        - Create a basic front end (dashboard, landing page, login) for the application.
        - User Story 1:
            - Create a database that holds the expenses
            - Create a UI that handles CRUD operations for the expenses database

            - Test this user story and associated classes with Jest along with manual testing
        - User Story 2:
            - Create a database that holds sales/income transactions (tied to inventory)
            - Create a UI that handles CRUD operations for the income + inventory database
            - Test this user story and associated classes with Jest along with manual testing
        - Create progress report one and ensure all tests/meetings/user stories are recorded

- **Sprint Two [Feb 18th - March 18th]:** The goal of this sprint is to implement inventory tracking, sales tax, and expense report generation. These features will require a menu for the user to manage and view their inventory, generate expense reports, and a form for users to specify details regarding sales tax. These user actions will then be reflected in the database as necessary.
    - Tasks for this sprint:
        - Researching how to integrate databases and unify CRUD operations between them
        - Researching the best sales tax practices and laws so that the software automatically makes users tax-compliant
        - Ensure each subsection is accessible and compatible with the dashboard
    - User Story 3 - Report Generation
        - Create a UI feature where users can select the type of report (income or expense) and specify the period for the same.
        - Develop back-end functionality using endpoints to handle requests for generating requested reports and logic to fetch the data from the database.
        - Utilize a library to format and generate reports and graphs from retrieved information.
        - Test the user story using Jest
    - User Story 4 - Sales Tax Collection + Remittance
        - Create a form to input sales tax that is sent to income and expense databases
        - Develop back-end functionality using endpoints to handle logic to fetch sales tax rate, income, and expense data from the database after which sales tax can be calculated

        - Utilize a library to format and generate reports from retrieved information
    - User Story 5 - Inventory
        - Create a database for the inventory
        - Attach/integrate with the income and expense databases
        - Use Jest and manual testing to ensure the integration works correctly with income/expense databases
    - Create progress report two and ensure all tests/meetings/user stories are recorded
- **Sprint Three [March 18th - April 8th]:** This sprint aims to focus on implementing user story 6, system-level testing, and Creating the final presentation for our project.
    - Tasks for this sprint:
        - User Story 6: As a user, I would like to have the advanced functionality of AI to give me insights about my business.
            - This involves: (pro user feature only)
                - Creating roles for pro users so that only they gain access to this feature.
                - Integrating either:
                    - Commercial LLM APIs (OpenAI, Anthropic, etc.)
                    - On-premises LLMs with locally offered APIs (open-source LMs such as Llama/Phi)
                - Creating a chat window/panel with pre-written prompts to enable discussion with the LLM.
                - Creating & managing an additional setting to toggle the presence of AI insights.
                - Creating easy-access buttons to generate view-specific insights (reports, inventory, etc.).
            - Task 1: Implement the external API of our choosing, while experimenting with prompt engineering to create appropriate responses.
            - Task 2: Implement the chat/insights interface as well as the per-page buttons to generate quick responses.
            - Task 3: Create a toggle for this feature while gating it behind Pro access.
            - Task 4: Further testing to ensure the model remains fair and appropriate to all users in all situations.
        - Integration & System-level testing
        - Create documentation for testing results

    - Create progress report 3 and ensure all tests/meetings/user stories are recorded.
    - Create final presentation

Team Member Contribution:

Each member of group 10 contributed equally to this release planning document. Our product owner, Rouvin, collaborated with Jacob to build and organize the Jira boards into sprints. Team member Skye documented the meeting minutes while providing details on sprints and user stories. Anthony, Sangmitra and Tyler provided expertise in crafting the user stories while collaborating with other team members.
Tasks achieved include:

- Completion of release planning document
- Jira backlog and sprint organization
- Documentation of progress achieved while looking at future requirements