Sprint Report

Book loan management project

By Eugene ETOUNDI II

26/01/2024

—

SOFTWARE ENGINEERING PROCESSES

—

Supervised by JOSÉ ANTONIO GALLUD LÁZARO

Contents

I. Introduction and Objectives of the Project

As part of the Software Engineering course, I have undertaken the development of a sophisticated Loan Management Web Application. This project addresses the intricate requirements associated with managing loans of books in a seamless and organized manner.

The primary focus of our application is to streamline the processes related to the borrowing and lending of books within a library or similar facility. The system will cater to the needs of both users seeking to borrow books and administrators responsible for overseeing the lending process. By leveraging cutting-edge technologies and following industry best practices, I aim to deliver a robust and scalable solution that meets the highest standards of software engineering.

Key Features:

- User Management: Efficient handling of user registrations, logins.
- Book Catalog: A comprehensive cataloging system that allows administrators to manage books, their availability, and associated details.
- Loan :
    - Create a book
    - Create a loan
    - Return a book
    - Add copy(ies) of a book
    - Remove copy(ies) of a book
    - Search a book and a loan with a key word
    - Notification to every connected user about a new book added
    - Log management
- Improvements on the view.

Through this project, I aim not only to develop a functional loan management system but also to hone our skills as aspiring software engineers, ready to contribute to the technological landscape. I invite you to embark on this journey with me as I explore the intricacies of software engineering in the development of our Loan Management Web Application.

II. Sprint reports

A. Sprint 1: Implement the base architecture of a generic software as a service (SaaS)

1. Sprint prep meeting:

a. With the product owner: objective/user stories/acceptances tests

- Objective:
    - The objective of Sprint 1 is to implement the initial version of user management functionality, focusing on core features that allow basic user interactions.
- User Stories:

  - As a new user, I want to be able to register with a nickname to access the system.
  - As an existing user, I want to log in using my registered nickname.
  - As an administrator, I want to retrieve a list of system users.
  - As an administrator, I want to check if a user is active.
  - As an administrator, I want the ability to delete a user by their nickname.
  - As a developer, I want to perform various queries related to users for system maintenance purposes.
- Acceptance Tests:
  - User Registration:
    - Scenario 1: User successfully registers with a unique nickname.
    - Scenario 2: User registration fails if the nickname is already in use.
  - User Login:
    - Scenario 1: User successfully logs in with a valid nickname.
    - Scenario 2: User login fails with an invalid nickname or password.
  - Retrieve User List:
    - Scenario 1: Administrator successfully retrieves a list of users.
    - Scenario 2: Retrieving the user list fails for unauthorized users.
  - Check User Activity:
    - Scenario 1: Administrator successfully checks if a user is active.
    - Scenario 2: Checking user activity fails for unauthorized users.
  - Delete User:
    - Scenario 1: Administrator successfully deletes a user by nickname.
    - Scenario 2: Deleting a user fails for unauthorized users.

b. From the team: detailed estimated tasks/hours

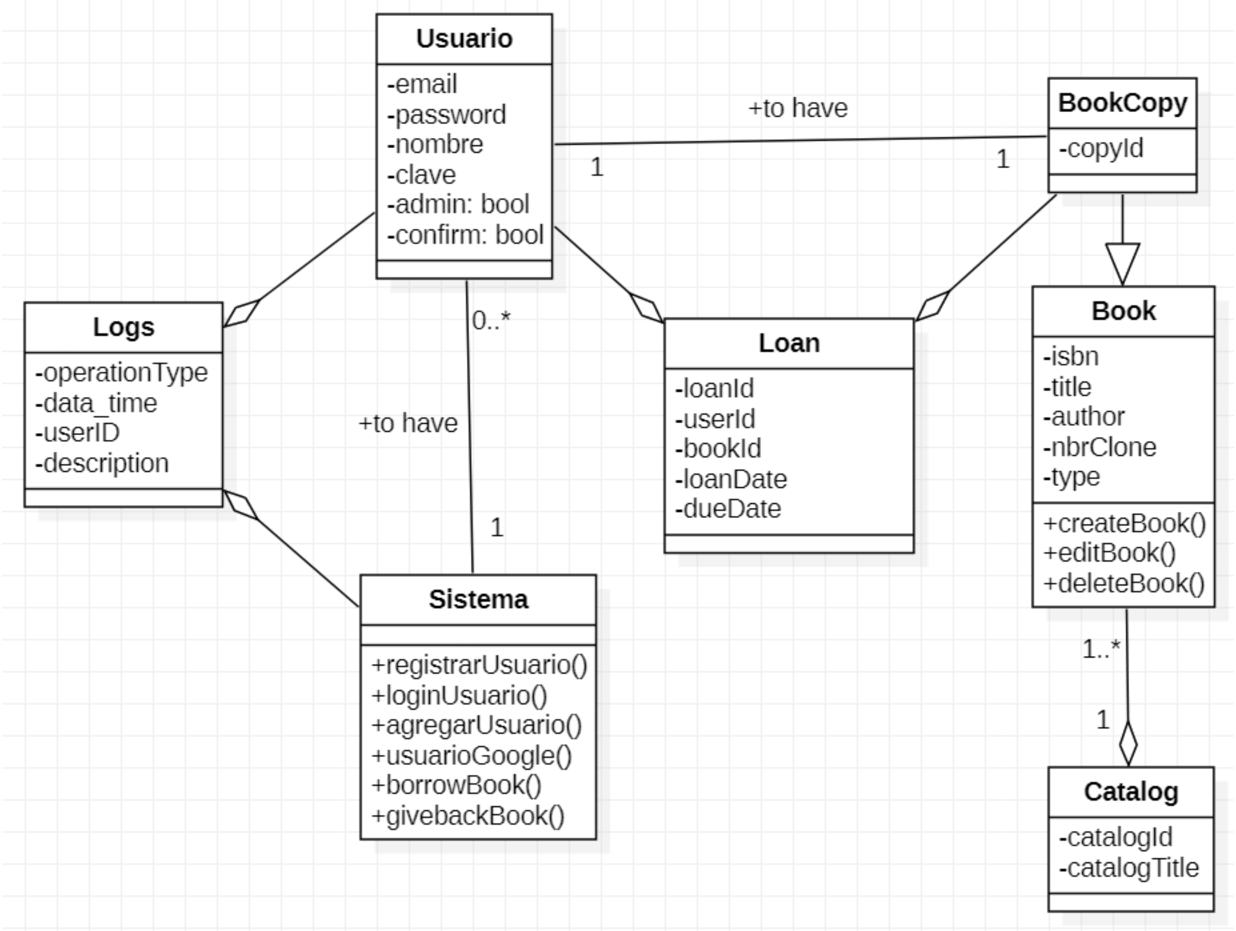
- User Management Module Setup:
  - Set up the initial structure for the user management module.
  - Estimated Hours: 4
- User Registration and Login Implementation:
  - Implement the functionality for user registration and login.
  - Estimated Hours: 12
- Retrieve User List Implementation:
  - Implement the functionality to retrieve a list of users.
  - Estimated Hours: 8
- Check User Activity Implementation:
  - Implement the functionality to check if a user is active.
  - Estimated Hours: 6
- Delete User Implementation:
  - Implement the functionality to delete a user by nickname.
  - Estimated Hours: 8

2. Development of the sprint:

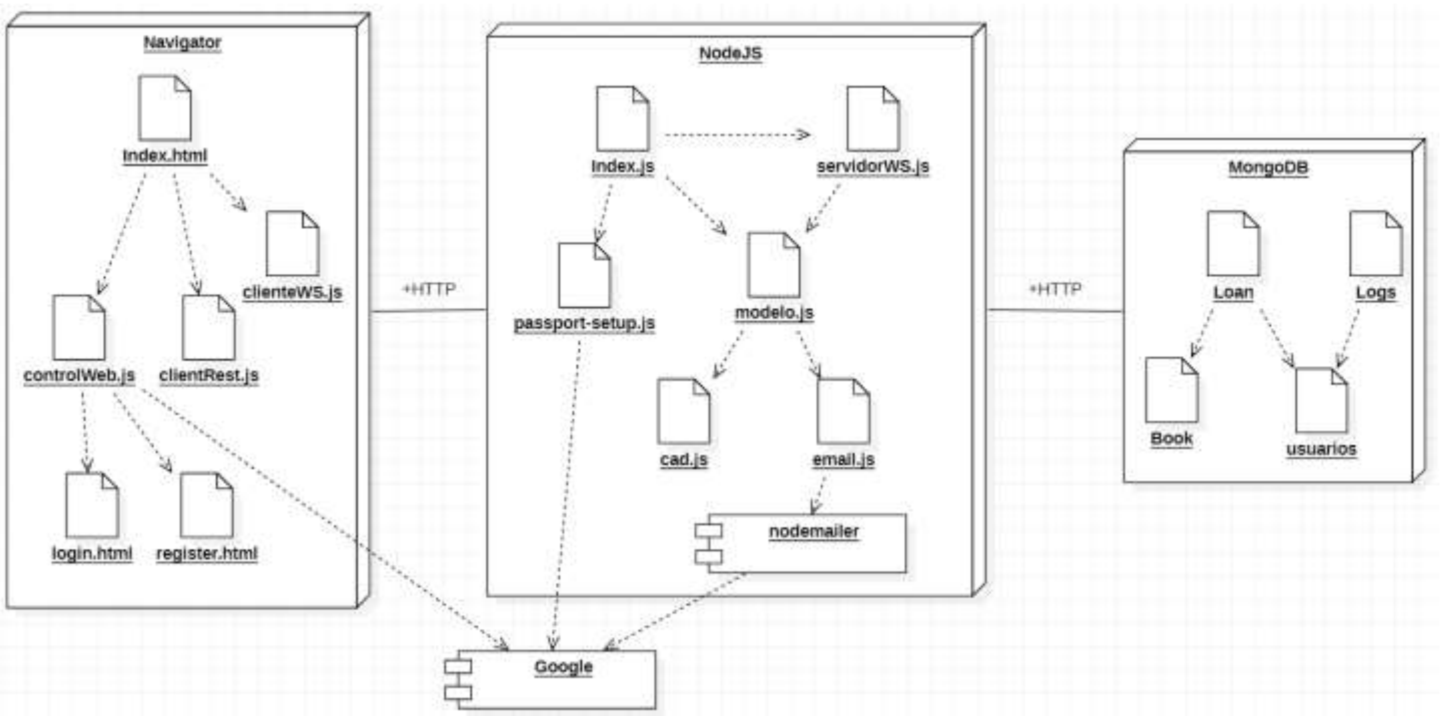
- Testing with Jasmine:
  - Test Automation Setup:
    - Configured Jasmine as the testing framework for automated testing.
    - Set up test suites and specifications to organize test cases.

- Unit Testing:
  - Conducted unit tests for individual components of the user management module.
  - Ensured that functions like user registration, login, retrieval, and deletion were tested in isolation.

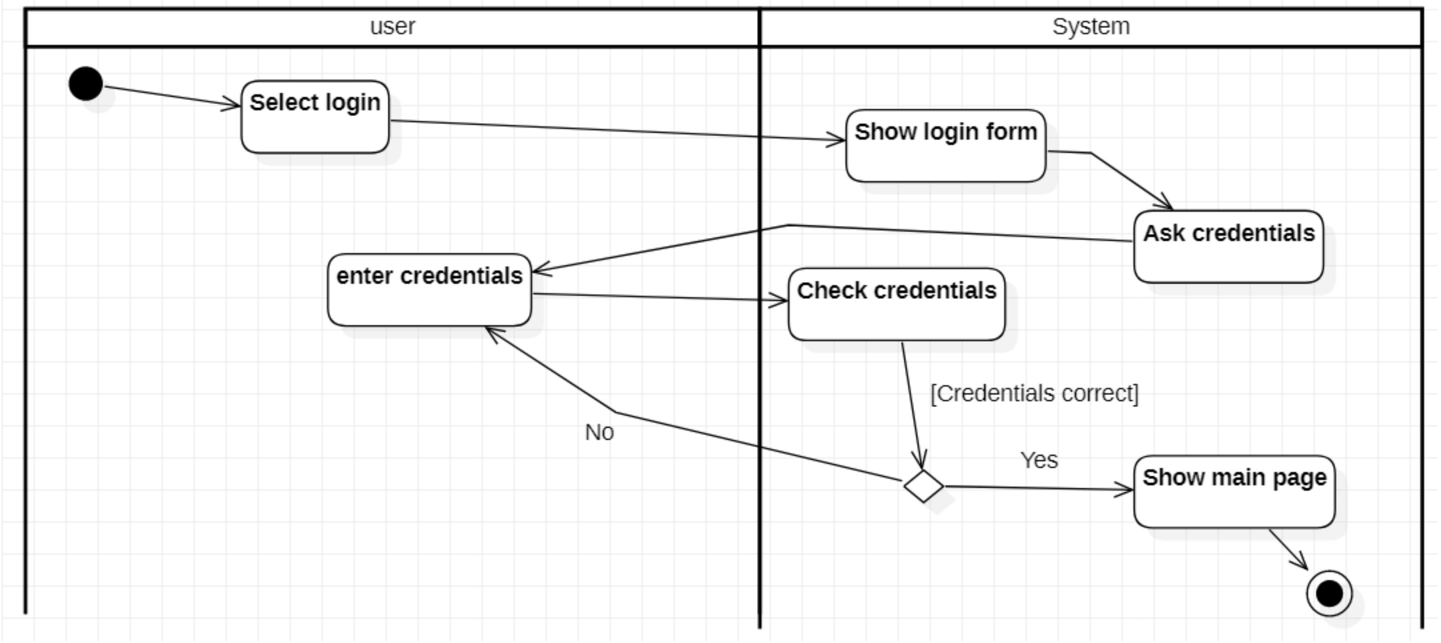
- Diagram:
  - Class Diagram:



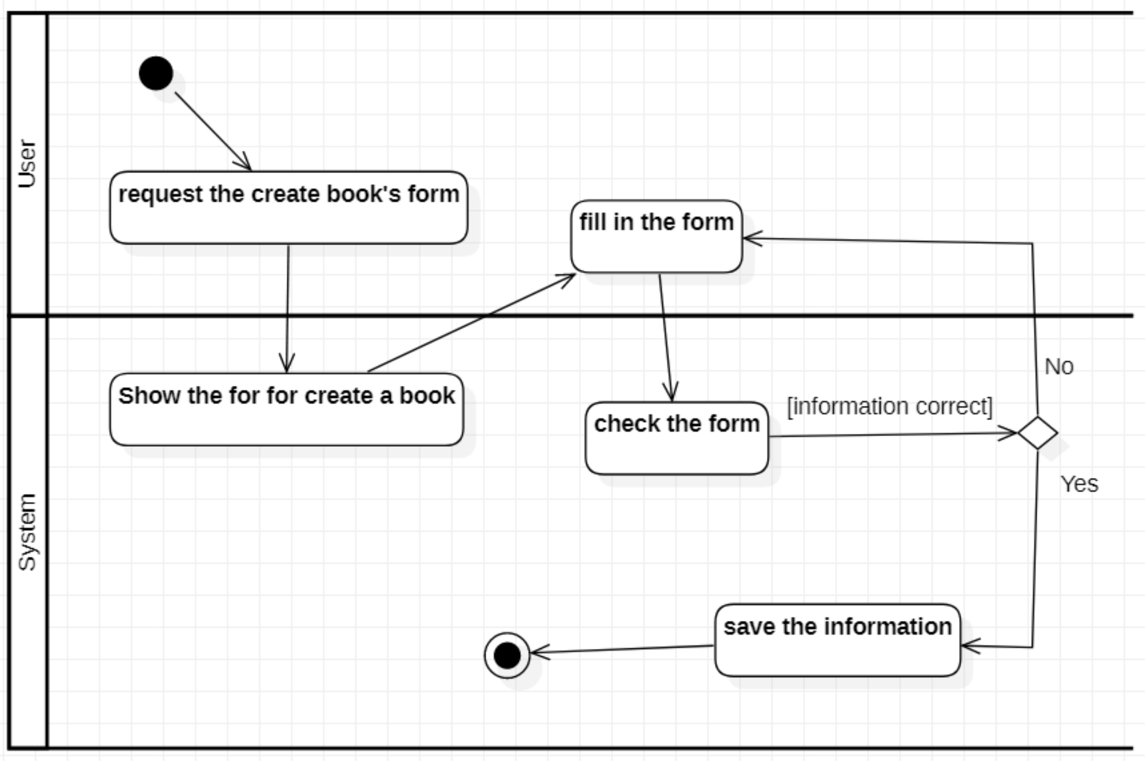
  - Architecture Diagram:



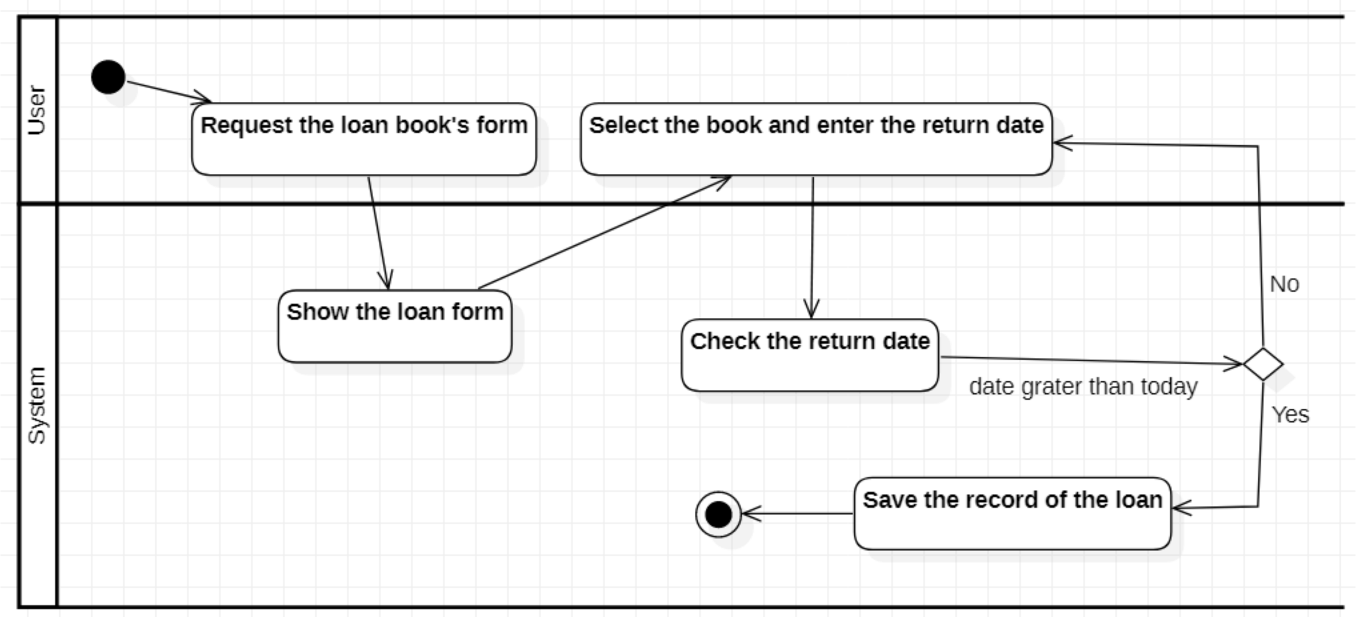
- Activity Diagram login:



- Activity Diagram createBook



- Activity diagram loanBook


User
System
Request the loan book's form
Select the book and enter the return date
Show the loan form
Check the return date
date grater than today
No
Yes
Save the record of the loan

3. Meeting at the end of the sprint:

a. Review (with the product owner): show the increase.

- Present the functionality allowing users to register with a nickname and log in.
- Highlight any additional features or improvements made based on the teacher's input.
- Demonstrate the ability to retrieve a list of users, check user activity, and delete users.
- Address any questions or concerns raised by the teacher regarding these operations.

b. Retrospective (of the team): sprint burndown is discussed.

- Display the Sprint Burndown chart depicting planned versus actual work completed throughout the sprint.
- Discuss the team's achievements during the sprint, such as completing planned tasks and meeting objectives.

B. Sprint 2: Users

1. Sprint prep meeting:

a. With the product owner: objective/user stories/acceptances tests

- Objective:
  - The objective of Sprint 2 is to enhance the application's functionality by managing the application state with cookies, implementing third-party authorization (Google), securing data access with MongoDB, and introducing features for user registration, confirmation, encryption, security, and logout.
- User Stories:
  - As a user, I want my application state to be managed efficiently using cookies for a seamless experience.
  - As a user, I want the option to log in using my Google credentials for quick and secure access.
  - As a developer, I want to establish a data access layer with MongoDB for efficient and scalable data storage.
  - As a user, I want a simplified registration process with the option to confirm my account via a confirmation link received by email.
  - As a user, I want my password to be securely encrypted to protect my account information.
  - As a user, I want access to secure routes, ensuring only registered users can use the system.
  - As a user, I want the ability to log out securely to protect my session.
- Acceptance Tests:
  - Application State with cookies:
    - Verify that the application state is managed effectively using cookies.
    - Ensure that users can navigate seamlessly between different sections without loss of data.
  - Third-Party Authorization (Google):
    - Confirm that users can log in using their Google credentials securely.
    - Validate that user information is retrieved and stored appropriately.
  - Data Access Layer with MongoDB:
    - Ensure that MongoDB is integrated into the application for efficient data storage.
    - Verify that data retrieval and manipulation operations work as expected.
  - Local User Registration:
    - Test the local user registration process, including form validation and user creation.
    - Confirm that user details are stored securely in the database.

    - Account Confirmation by Email:
        - Validate that users receive a confirmation email with a link for account verification.
        - Confirm that clicking the verification link successfully activates the user account.
    - Password Encryption:
        - Test the encryption of user passwords to ensure security.
        - Validate that encrypted passwords match the expected format.
    - Secure Routes:
        - Test the implementation of secure routes to prevent unauthorized access.
        - Ensure that only registered users can access protected sections.
    - Logout Functionality:
        - Validate the secure logout functionality.
        - Confirm that users are logged out, and session data is appropriately cleared.

b. From the team: detailed estimated tasks/hours

- Manage Application State with Cookies:
    - Implement client-side and server-side logic for managing the application state with cookies.
    - Estimated Hours: 10
- Implement Third-Party Authorization (Google):
    - Integrate Google OAuth for user authentication.
    - Implement the necessary backend and frontend components.
    - Estimated Hours: 16
- Implement Data Access Layer with MongoDB:
    - Set up MongoDB integration, including schema design and CRUD operations.
    - Ensure proper handling of data retrieval and storage.
    - Estimated Hours: 20
- Implement Google One Tap:
    - Integrate Google One Tap for a streamlined authentication experience.
    - Implement frontend components for Google One Tap.
    - Estimated Hours: 12
- Implement Local User Registration:
    - Develop the user registration form and backend logic.
    - Include form validation and error handling.
    - Estimated Hours: 14
- Implement Account Confirmation by Email:
    - Set up email services for sending confirmation emails.
    - Implement backend logic to process confirmation links.
    - Estimated Hours: 18
- Encrypt User Password:
    - Implement password encryption for user security.
    - Ensure that encrypted passwords adhere to security standards. - Estimated Hours: 8

2. Development of the sprint:

- Ensure that the application state is managed effectively using cookies.
- Validate that users can securely log in using their Google credentials.

- Confirm the proper functioning of data access with MongoDB.
- Verify the integration of Google One Tap for streamlined authentication.
- Validate the local user registration process.
- Confirm that users receive and successfully confirm their accounts via email.
- Ensure the secure encryption of user passwords.
- Confirm the implementation of secure routes.
- Validate the logout functionality.

3. Meeting at the end of the sprint:

a. Review (with the product owner): show the increase

- Present the application state management using cookies, emphasizing a seamless user experience.
- Demonstrate Google OAuth integration for secure and convenient user authentication.
- Highlight the functionality of the MongoDB data access layer, emphasizing efficiency and scalability.
- Showcase additional features, including local user registration, account confirmation via email, password encryption, secure routes, and logout.

b. Retrospective (of the team): sprint burndown is discussed

- Evaluate the progress made during the sprint by reviewing the Sprint Burndown chart.
- Display and analyse the Sprint Burndown chart, comparing planned versus actual progress.
- Identify any deviations or trends that require attention or improvement.

C. Sprint 3: Project Design

1. Sprint prep meeting:

a. With the product owner: objective/user stories/acceptances tests

- Objective:
  - The objective of Sprint 3 is to enhance the system's security by implementing secret keys management for sensitive information. Additionally, improvements to the user interface, WebSocket integration for real-time communication, end-to-end testing with Playwright, activity logging, and decoupling the client and server will be addressed.
- User Stories:
  - As a system administrator, I want sensitive information to be managed securely with the implementation of secret keys.
  - As a user, I want an improved user interface with the addition of modal elements for enhanced user experience.
  - As a user, I want real-time updates on system events, such as book additions, through WebSocket integration.
  - As a QA engineer, I want to perform end-to-end testing using Playwright to ensure comprehensive test coverage.
  - As a system administrator, I want to log user actions for auditing and monitoring purposes.

  - As a developer, I want to decouple the client and server components for better maintainability and scalability.
  -
- Acceptance Tests:
  - Secret Keys Management:
    - Verify that sensitive information is appropriately managed with the implementation of secret keys.
    - Ensure that access to sensitive data is restricted and follows best security practices.
  - Improved User Interface:
    - Test the user interface improvements, including the addition of modal elements.
  - WebSocket Integration:
    - Validate real-time communication between the server and clients through WebSocket integration.
    - Test scenarios where clients receive instant notifications, such as book additions.
  - End-to-End Testing with Playwright:
    - Create Playwright scripts to cover end-to-end testing scenarios for critical system functionalities.
    - Ensure that Playwright tests cover a wide range of user interactions and edge cases.
  - Activity Logging:
    - Test the implementation of activity logging to save user actions.
    - Verify that logs capture relevant information for auditing purposes.
  - Decoupling Client and Server:
    - Test the decoupling of client and server components to ensure independence.
    - Confirm that changes on one side do not negatively impact the other.

b. From the team: detailed estimated tasks/hours

- Secret Keys Management:
  - Implement a secure mechanism for managing secret keys.
  - Estimated Hours: 12
- User Interface Improvements:
  - Add modal elements to the user interface for additional improvements.
  - Estimated Hours: 8
- WebSocket Integration:
  - Integrate WebSocket communication for real-time updates.
  - Implement server and client components for WebSocket functionality.
  - Estimated Hours: 16
- End-to-End Testing with Playwright:
  - Develop Playwright scripts to cover end-to-end testing scenarios.
  - Execute tests and address any issues identified during testing.
  - Estimated Hours: 20
- Activity Logging Implementation:
  - Implement activity logging to record user actions.
  - Ensure logs contain relevant information for auditing.
  - Estimated Hours: 14

- Decoupling Client and Server:
  - Modify the architecture to achieve decoupling between client and server.
  - Refactor code and update dependencies to support the decoupling.
  - Estimated Hours: 18

2. Development of the sprint:

- Ensure the secure management of sensitive information with secret keys.
- Validate the enhanced user interface with modal elements.
- Verify real-time communication through WebSocket integration.
- Ensure comprehensive end-to-end test coverage using Playwright.
- Validate the implementation of activity logging for user actions.
- Test the decoupling of client and server components.
- Conduct end-to-end testing to ensure the seamless integration of all implemented features.

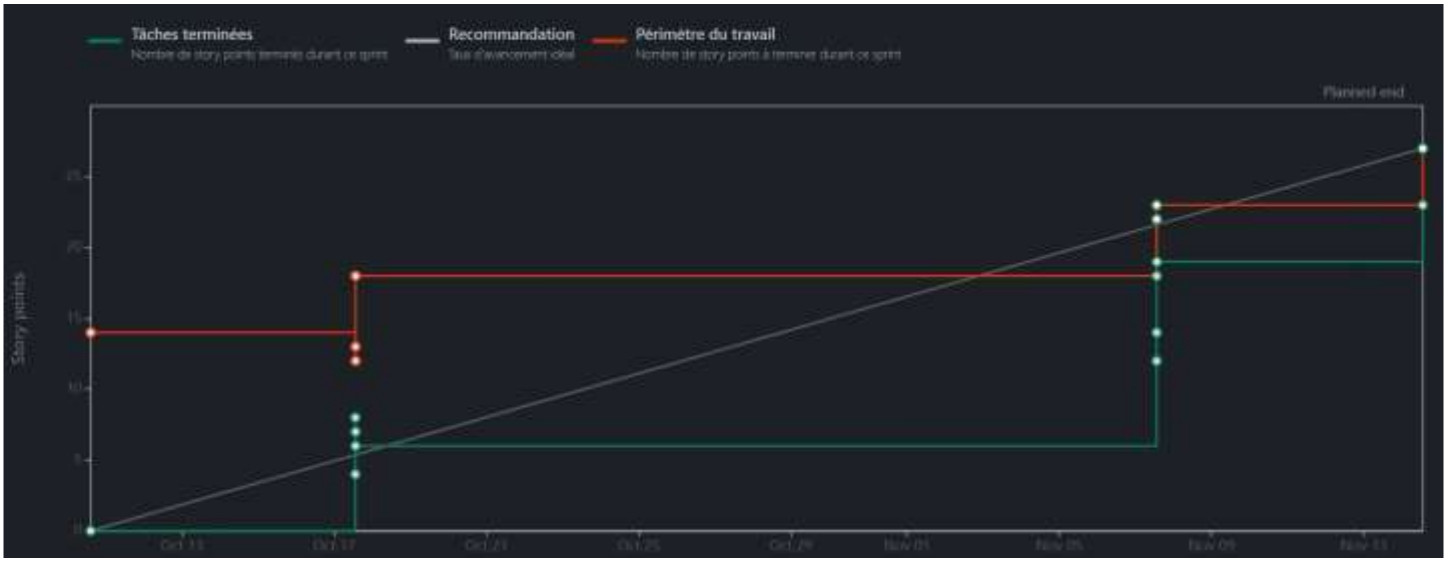
3. Meeting at the end of the sprint:

Review

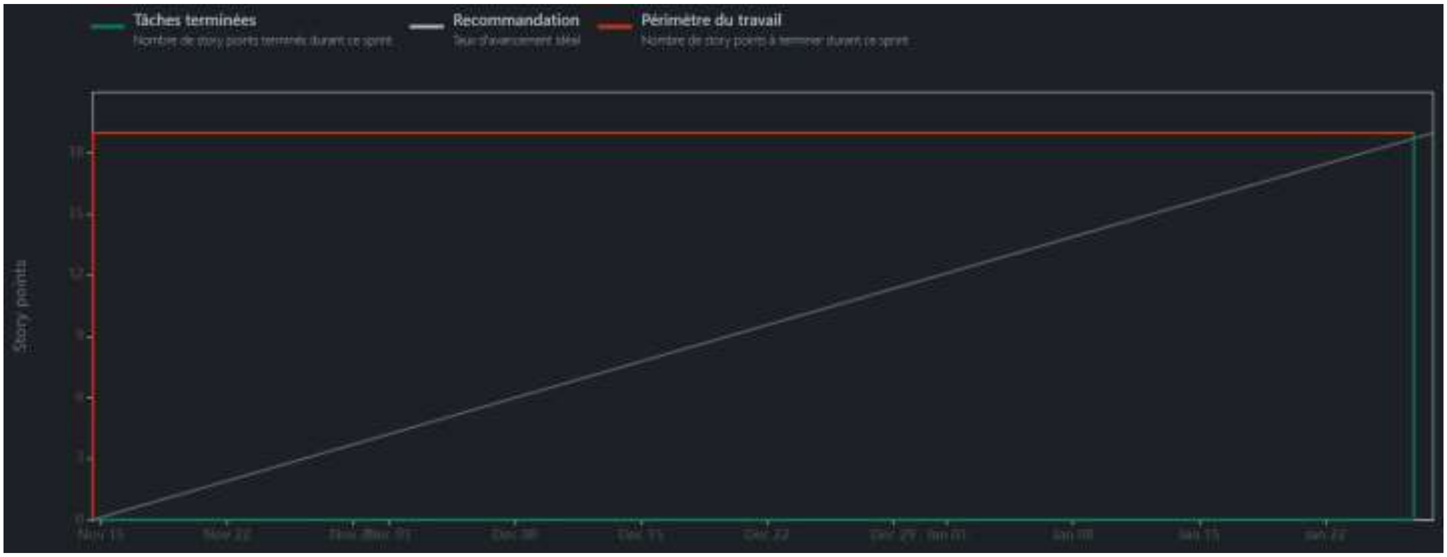
- Submission and evaluation of the project
- Making a video presentation

III. Attachments

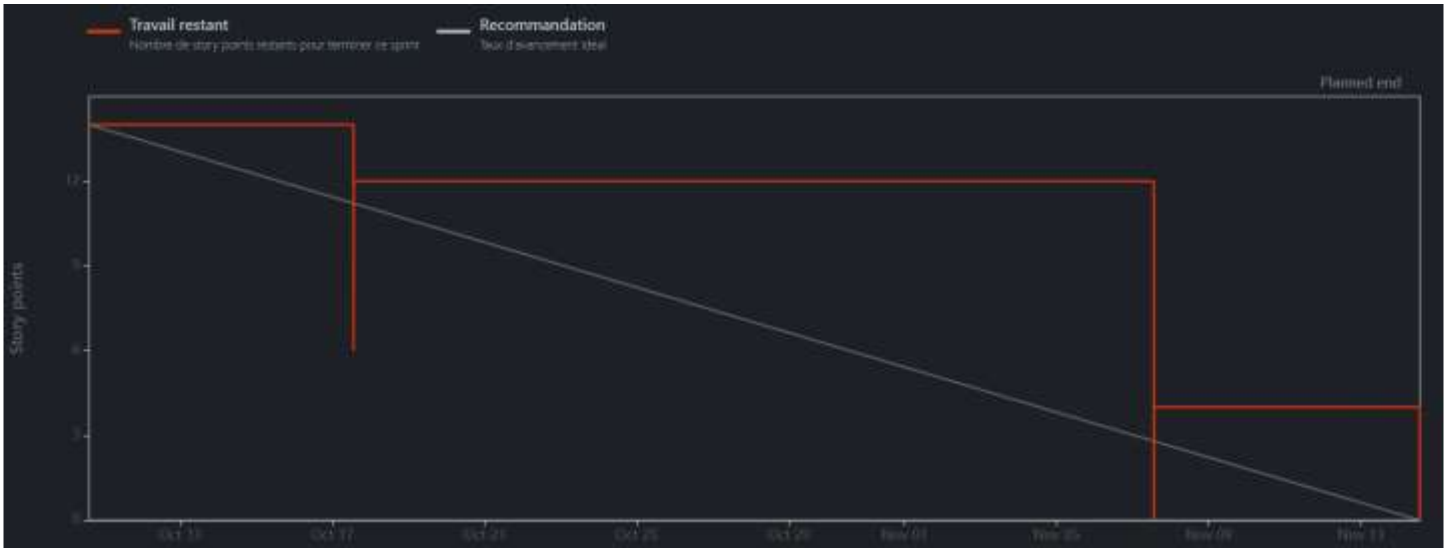
1. Burnup sprint 2 report



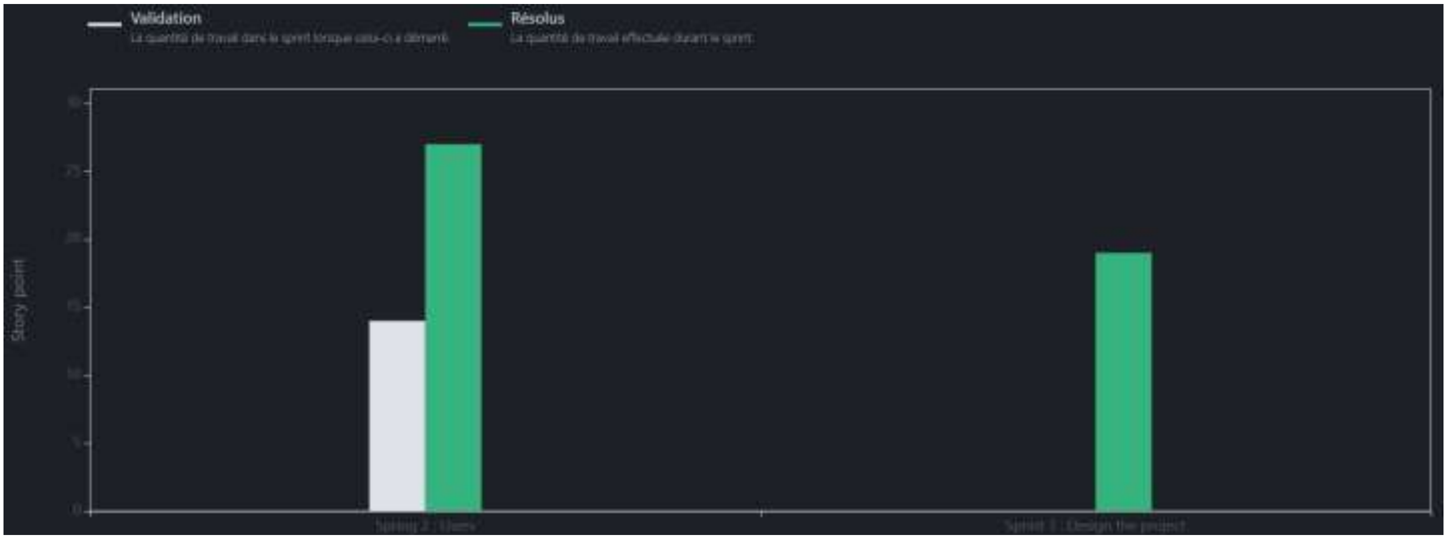
2. Burnup sprint 3 report



3. Sprint 2 Burndown graph



4. Velocity ratio


Validation
La quantité de travail dans le sprint lorsque celui-ci a démarré.
Résolus
La quantité de travail effectuée durant le sprint.
Story point
0
5
10
15
20
25
30
Spring 2 : Users
Sprint 1 : Design the project