Project Milestone 2

By
Ethan Surber
Menase Yirdaw
Ian Yoon
Yasser Algburi

Process Deliverable I - Agile

Agile: submit a summary of retrospective (retrospective) and prioritized tasks for the next milestone (sprint planning)

Process Deliverable I: Retrospective Summary

- **What went well**:
  - **Clear Understanding of Project Goals**: The team has a solid grasp of the project's main objectives, including prioritizing features that enhance productivity, task management, and real-time notifications.
  - **Effective Initial Planning**: Team communication has been productive, and planning sessions have outlined key features for FocusBot, laying the groundwork for development.
- **Challenges Encountered**:
  - **Uncertain Dependencies**: With some features dependent on others, particularly notifications and analytics, it's been challenging to finalize a logical development order.
  - **Task Prioritization**: There's been some uncertainty in deciding which core features should be implemented first to maximize project momentum and progress.
- **Improvements for the Next Sprint**:
  - **Focus on Foundational Features First**: Prioritize implementing core functionality, like user registration and the basic to-do list, to provide a solid foundation for additional features.
  - **Map Dependencies More Clearly**: Identify feature dependencies early on, so task assignments avoid bottlenecks.
  - **End-of-Sprint Reviews**: Schedule brief reviews to identify and address issues sooner, ensuring alignment and progress tracking.

Sprint Planning: Prioritized Tasks for Next Milestone

1. **User Registration System** (High Priority)

    - **Goal**: Create an account setup where users can register with basic information (username, email, password).
    - **Tasks**: Design the registration form, implement form validation, ensure the system saves account data securely, and configure a basic authentication process.
    - **Dependencies**: Authentication setup will form a basis for secure user data handling in future features.
2. **To-Do List Core Functionality** (High Priority)
    - **Goal**: Enable users to create, edit, and delete tasks, forming the foundation of the productivity system.
    - **Tasks**: Set up a basic to-do list interface, define task fields (title, description, priority), and allow users to manage tasks.
    - **Dependencies**: Essential for implementing notifications and analytics, as it's the primary data source for tracking productivity.
3. **Pomodoro Timer Basic Setup** (Medium Priority)
    - **Goal**: Provide a timer that allows users to start and stop work intervals.
    - **Tasks**: Implement a timer UI, basic start/stop functionality, and create a placeholder for future customization of intervals.
    - **Dependencies**: None initially, but customization will later depend on this basic setup.
4. **Outline Analytics Dashboard Design** (Low Priority)
    - **Goal**: Begin planning an analytics dashboard to track productivity data for team managers.
    - **Tasks**: Draft the layout, decide on metrics to be displayed (e.g., tasks completed, Pomodoro sessions), and identify data sources.
    - **Dependencies**: Basic data from to-do lists and Pomodoro sessions will be essential once metrics are tracked.
5. **Establish Security Standards for Data Encryption** (Low Priority)
    - **Goal**: Outline encryption standards to ensure secure data handling.
    - **Tasks**: Define encryption requirements, consider secure data storage methods, and establish basic security practices.
    - **Dependencies**: Complements the user registration feature, ensuring sensitive user information is protected.

Requirements Analysis

Based on the results of your requirements elicitation, goals for your project, and course materials, please complete the following tasks:

- Provide an example of five hypothetical non-functional requirements for your system. Be sure to include the specific type of requirement discussed in class, with each requirement coming from a unique category.
- Provide an example of five hypothetical functional requirements for your system.

- Write five formal use cases for your system and provide use case or sequence diagrams to represent each use case.

Requirements Analysis

Hypothetical non-functional requirements:

- The System must respond to all user actions within a certain timeframe, such as 200 milliseconds. This is to ensure a seamless user experience.
- The user interface must be intuitive and straightforward enough such that more than 80% of users can complete tasks without needing any external guidance.
- FocusBot must have an uptime of at least 99.5% within a 24 hour period. This is to ensure that users can always rely on the service and to ensure that FocusBot is not constantly offline.
- FocusBot must be able to handle upwards of 10,000 users at any given time without any sort of performance degradation.
- In order to preserve security and privacy, all User data must be encrypted and safely stored.

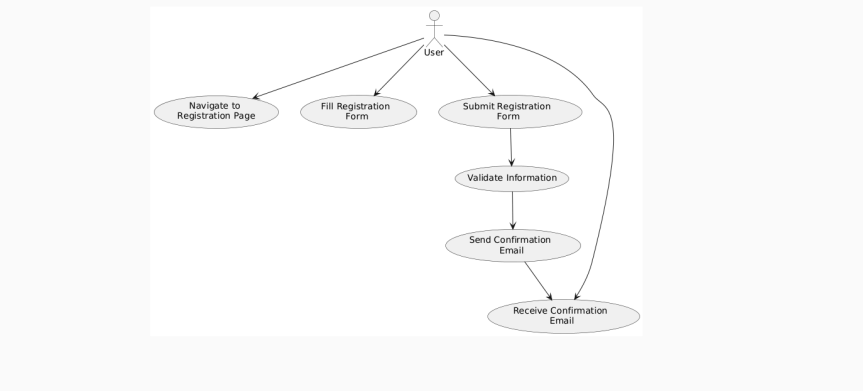
Hypothetical functional requirements:

- The system must allow users to create, edit, and delete certain or all items in their to-do lists, as well as specify the priority for each task
- FocusBot must implement the Pomodoro technique. This can be done by enabling users to set their own work and break intervals, and can use notifications to signify start/stops.
- FocusBot must send real-time notifications to users for task reminders. The option to customize the frequency and type of notifications must also be included.
- FocusBot must allow users to integrate with other third-party applications.

Five Formal Use Cases

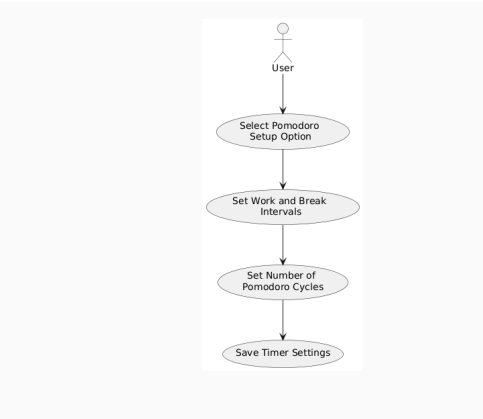
Use Case 1: Register for FocusBot

- **Actors**: User
- **Preconditions**: The user has internet access and can access the registration page.
- **Steps**:
    1. User navigates to the registration page.
    2. User fills out the username, password, and email fields.
    3. User submits the registration form.
    4. System validates the information and checks for any errors.
    5. System sends a confirmation email to the user.
- **Postconditions**: User receives a confirmation email, and their account is created.



Use Case 2: Set Up a Custom Pomodoro Timer

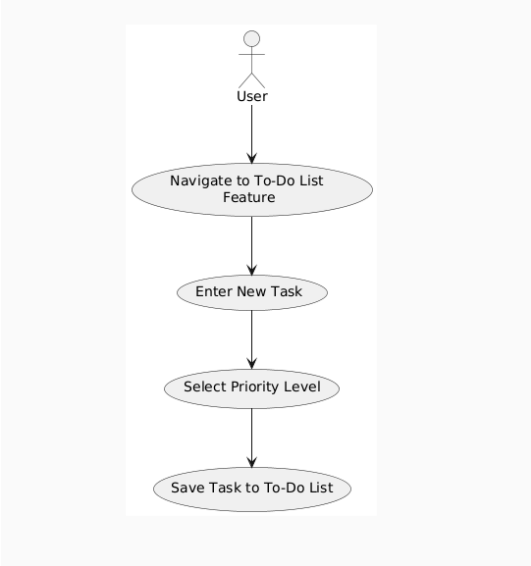
- **Actors**: User
- **Preconditions**: The user is logged in.
- **Steps**:
    1. User selects the Pomodoro setup option.
    2. User sets work and break intervals.
    3. User sets the desired number of Pomodoro cycles.
    4. User saves the timer settings.
- **Postconditions**: Pomodoro timer is configured, ready to start when the user presses the “Start” button.



Use Case 3: Add and Prioritize a Task in To-Do List

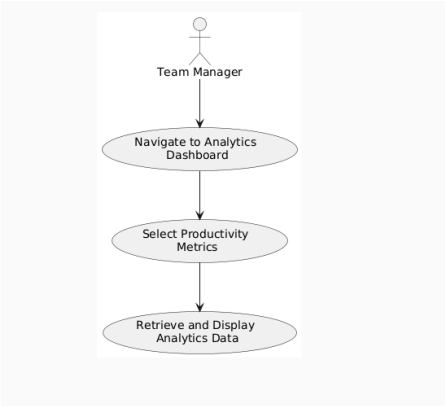
- **Actors**: User
- **Preconditions**: The user is logged in.
- **Steps**:
    1. User navigates to the to-do list feature.
    2. User enters a new task in the task input field.
    3. User selects the priority level of the task.
    4. User saves the task to the to-do list.

- **Postconditions**: The new task is added to the list with the selected priority.



Use Case 4: View Team Analytics (for Managers)

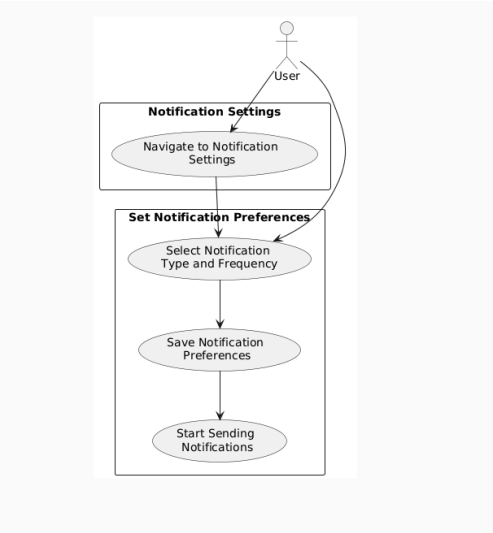
- **Actors**: Team Manager
- **Preconditions**: The team manager is logged in and has team permissions.
- **Steps**:
    1. Manager navigates to the analytics dashboard.
    2. Manager selects options to view productivity metrics.
    3. System retrieves and displays real-time analytics (e.g., Pomodoro sessions completed, tasks completed).
- **Postconditions**: Manager views the analytics data for their team.



Use Case 5: Enable Real-Time Notifications

- **Actors**: User
- **Preconditions**: User is logged in and has at least one active task.

- **Steps**:
    1. User navigates to the notification settings.
    2. User selects options for notification type and frequency.
    3. User saves notification preferences.
    4. System starts sending notifications as per the selected preferences.
- **Postconditions**: User receives notifications according to their setting



Requirements Specification

Based on the results of your requirements elicitation, goals for your project, and course materials, please complete the following tasks:

- Write four user stories from the perspective of at least two different actors. Provide the acceptance criteria for these stories.
- For each user story mentioned above, estimate the amount of effort needed to complete relevant subtasks using function points. Explain your answer.

Requirements Specification

User Story: As a user, I want to be able to register for the Focus Bot, so I can have access to the features the program provides.

Acceptance Criteria

- Users can only submit the registration form when the username, password, and email fields have been filled.
- The user must provide a valid email address.
- Users will receive an email after successful registration to confirm their account.
- Passwords must meet complexity requirements.

Function Points: I would estimate around 20 function points due to the security implementation for storing the username and password along with the validation of the password and email.

User Story: As a team manager, I want to be able to view my team's analytics, so I can see how they are utilizing FocusBot.

Acceptance Criteria

- Team leaders can access a dashboard showing individual and team productivity metrics.
- Metrics shown include Pomodoro sessions completed, tasks completed, and time spent on these tasks.
- The dashboard will update in real-time.

User Story: As a user, I want to be able to set a custom Pomodoro timer for work and break times, so I can manage my workload more efficiently.

Function Points: I would estimate this to be around 40 function points. Real-time updates would be a complex task to do. In addition to this, implementing this requires some form of data aggregation along with a various number of UI components.

Acceptance Criteria

- The user must provide a valid time (in minutes) for the Pomodoro.
- The session begins when the “Start” button is pressed, and ends whenever the user presses the “End” button.
- Users can set a specific number of cycles before the Pomodoro session ends.
- After four small cycles, the user is given a larger break.

Function Points: I would estimate this to be around 15 function points. Implementing this feature would be relatively simple as it is mostly UI based.

User Story: As a user, I want to add a task to my to-do list, and then prioritize this task so I can complete my most important tasks first.

Acceptance Criteria:

- The user must provide a non-empty string to input in the task.
- To prioritize the task, the user must click the three dots to the side of the task, then select the priority level.
- Task’s should be color coded to reflect priority.
- System confirms when a task is added and the priority of this task.

Function Points: Implementing this would be around 25 function points. Various UI components need to be implemented along with adding a prioritization feature to the to-do-list system.