Process Deliverable: Agile

- What went well: Through our surveying process we managed to perform our requirements elicitation and receive good feedback and insight on what our product should and shouldn't include.
- What did not go as planned: However, our surveying process took longer than expected. We had some difficulty getting timely responses. This was due to factors such as respondent availability.
- What can we do better: For our next feedback collection process, we will begin our survey earlier to accommodate for response times and ask for a quick response. We will also look to potentially use other forms of feedback such as focus groups and interviews to mitigate survey delays.

- Prioritized tasks (sprint planning): Product iteration and redesigning based on requirements analysis.

Requirements Analysis:

Non-Functional Requirements

- **Usability**: The system should allow users to easily navigate and filter TODO comments through an intuitive UI, ensuring that new users can complete basic tasks with minimal training.
- **Reliability**: The TODO list must be consistently available, with no more than 0.1% downtime over a one-month period, to ensure developers can rely on it for tracking tasks.
- **Performance**: The centralized TODO list should compile and display TODO comments across open files in under 1 second, with real-time updates for any modifications.
- **Scalability**: The system should support a codebase with up to 10,000 TODO comments across multiple files without performance degradation.
- **Maintainability**: The system's codebase should be modular, allowing for new features (such as integrations with additional project management tools) to be added with minimal disruption to existing functionality.

Functional Requirements

- The system must detect all TODO comments in open files within the VSCode editor and display them in a centralized list.
- The system must allow users to filter TODO items by tags (e.g., "Onboarding") or assigned developers.
- The system should enable integration with Jira for syncing TODO comments with Jira tasks, allowing users to link comments to Jira tickets.
- The system should provide a feature to categorize TODO comments by priority, allowing users to focus on high-priority items.

- The system must allow users to click on a TODO item in the centralized list to automatically navigate to the corresponding line in the source code.

Use Cases and Diagrams

- **Use Case 1: Display Centralized TODO List**
  - **Actor**: Software Developer
  - **Goal**: To view all TODO comments across open files in a single, centralized list.
  - **Main Success Scenario**:
    1. The Software Developer opens the TODO management extension in VSCode.
    2. The system compiles all TODO comments in the currently open files.
    3. The system displays the compiled TODO comments in a centralized list, organized by file.
  - **Diagram**: Use case diagram showing the Software Developer accessing the “Display Centralized TODO List” feature.
- **Use Case 2: Filter TODO Comments by Assigned Developer**
  - **Actor**: Entry-Level Developer
  - **Goal**: To filter TODO comments based on tasks assigned to them to streamline onboarding.
  - **Main Success Scenario**:
    1. The Entry-Level Developer selects the filter option within the TODO list extension.
    2. The system provides a dropdown to filter by tags or assigned developer.
    3. The Entry-Level Developer selects their name.
    4. The system updates the list to show only tasks assigned to the selected developer.
  - **Diagram**: Use case diagram showing the Entry-Level Developer interacting with the filter feature.
- **Use Case 3: Integrate TODO Comments with Jira**
  - **Actor**: Product Manager
  - **Goal**: To integrate TODO comments with Jira for task tracking and progress monitoring.
  - **Main Success Scenario**:
    1. The Product Manager accesses the TODO list extension and selects the Jira integration feature.
    2. The system prompts the Product Manager to log in to Jira.
    3. The Product Manager links a TODO comment to an existing Jira ticket or creates a new ticket.
    4. The system syncs the TODO comment with Jira and displays a link to the ticket.
  - **Diagram**: Sequence diagram illustrating the steps from accessing Jira integration to syncing with Jira.
- **Use Case 4: Categorize TODO Comments by Priority**
  - **Actor**: Product Manager

    - **Goal**: To categorize and prioritize TODO comments for efficient task management.
    - **Main Success Scenario**:
        1. The Product Manager opens the TODO list in VSCode.
        2. The Product Manager selects the option to categorize by priority.
        3. The system displays options for marking each TODO item by priority (e.g., high, medium, low).
        4. The Product Manager assigns priorities to specific TODO items.
        5. The system updates the display to organize TODO items by priority level.
    - **Diagram**: Use case diagram showing prioritization of TODO items.
- **Use Case 5: Navigate to TODO Comment Location**
    - **Actor**: Software Developer
    - **Goal**: To navigate directly to the location of a TODO comment within the code.
    - **Main Success Scenario**:
        1. The Software Developer clicks on a TODO item in the centralized list.
        2. The system navigates to the line in the code file where the TODO comment is located.
        3. The TODO item is highlighted for easy identification.
    - **Diagram**: Sequence diagram illustrating the process from clicking a TODO item to navigating to its location.

Requirements Specification:

User Story One:

“As a Software Developer, I want to see all TODO comments from my code files in a centralized list so that I can efficiently manage my tasks without needing to search through individual files. This will allow me to keep track of outstanding items across various sections of the codebase in one place, making my workflow more organized and streamlined. With this centralized view, I can prioritize and address tasks more effectively, ensuring nothing gets overlooked during development.”

Acceptance Criteria:

- The extension compiles all TODO comments across open code files in VSCode into a single, centralized list.
- The list updates in real time as TODO comments are added, edited, or removed.

Effort Estimation:

Centralized List for TODO Comments
Effort: Medium

Creating and updating a centralized list requires parsing code files for TODO comments and displaying them in the UI, with real-time updates.

User Story Two:

“As a New Entry-Level Developer who is onboarding, I want to filter TODO comments based on my assigned tasks so that I can focus on what’s relevant to me during my initial learning phase. This feature would allow me to gradually familiarize myself with specific parts of the codebase without feeling overwhelmed by unrelated tasks. By concentrating on my assigned TODOs, I can progress through onboarding in a structured way and become more productive in a shorter amount of time.”

Acceptance Criteria:

- The extension provides a filter to view only tasks tagged as "Onboarding" or assigned to the specific developer.
- Tasks specific to onboarding are highlighted or grouped separately in the list to help the developer prioritize relevant work.

Effort Estimation:

Filtering for Onboarding Tasks for New Developer
Effort: Medium

Implementing filtering functionality requires setting up criteria for tags or assigned developers and making these accessible in the UI.

User Story Three:

“As a Product Manager, I want to integrate tasks from the VSCode extension with Jira so that I can track developer progress and ensure alignment with overall project goals. Having TODO comments synced with Jira tickets would provide better visibility into task completion rates, helping me identify potential bottlenecks or areas where additional support might be needed. This integration would enable a seamless workflow between development and project management, keeping everyone informed and aligned with project milestones.”

Acceptance Criteria:

- The extension provides an option to link each TODO comment to a Jira ticket.
- Clicking a TODO item in the centralized list allows users to view or create a corresponding Jira ticket in Jira.

Effort Estimation:

Jira Integration
Effort: High

Integrating with Jira API for ticket creation and linking requires API calls, authentication handling, and synchronization features.

User Story Four:

“As a Product Manager, I want to view task tags and priorities for each TODO comment so that I can quickly assess which issues are most critical to the project’s success. This would allow me to distinguish between high-priority and low-priority tasks, helping to ensure that critical issues are addressed promptly. With clear visibility into task categorization, I can make more informed decisions and communicate priorities to the development team effectively.”

Acceptance Criteria:

- Clicking on a TODO item in the centralized list opens the corresponding code file in VSCode and navigates to the exact location of the TODO comment.
- This extension provides a simple interface for product managers to switch between tasks and review context efficiently.

Effort Estimation:

Navigation to TODO Comment Locations
Effort: Low.

Enabling navigation to specific locations in code files is achievable through the VSCode API, which supports navigating to specific lines or sections within files.