AGILE:

Theme: Get GiggleGit demo into a stable enough alpha to start onboarding some adventurous clients

Epic: Onboarding experience

User Story 1:

- As a vanilla git power-user that has never seen GiggleGit before, I want to quickly understand how to perform basic version control tasks so that I can efficiently use the system without losing my existing workflow.
- This includes being able to clone repositories, create branches, commit changes, and push to the remote with minimal friction.

Task: Create onboarding documentation for power-users.

Tickets:

- Title: Write onboarding documentation
- Details: Develop comprehensive documentation that explains how to perform essential tasks in GiggleGit for users familiar with standard Git. Include examples of commands, expected outcomes, and any differences from traditional Git behavior.

- Title: Create quick-start guide
- Details: Design a quick-start guide that provides step-by-step instructions for common version control actions in GiggleGit. This should be concise and highlight the unique features of GiggleGit that differentiate it from standard Git.

User Story 2:

- As a team lead onboarding an experienced GiggleGit user, I want to provide tailored training sessions so that my team can leverage GiggleGit's unique features effectively.
- This includes demonstrating how to use meme-based merge management and how to customize workflows within the platform.

Task:Schedule and conduct training sessions for experienced users.

Tickets:

- Title: Develop training materials
- Details: Create slide decks and interactive materials for training sessions aimed at experienced GiggleGit users. Focus on advanced features, including meme-based merges, and best practices for integrating GiggleGit into existing workflows.

- Title: Host live Q&A sessions

- Details:Organize and host live Q&A sessions to address questions and concerns from experienced users. This will provide an opportunity for hands-on learning and ensure users feel comfortable with the platform.

User Story 3:
- As a new user, I want to easily set up my environment so that I can start using GiggleGit without hassle.
- This includes being able to install the software, configure my user settings, and understand the initial setup process.

Task: Create an installation and setup guide.

Tickets:
- Title: Document installation steps
- Details: Write detailed instructions for installing GiggleGit on various operating systems (Windows, macOS, Linux). Include troubleshooting tips for common issues during installation.

- Title: Outline initial configuration process
- Details: Develop a guide that walks users through the initial configuration of their GiggleGit environment, including setting user preferences and connecting to repositories.

FORMAL REQUIREMENTS

1. Why this user story doesn’t work: This statement lacks context regarding the user's needs or goals in relation to their experience with GiggleGit. A user story should articulate who the user is, what they want, and why it’s important to them, providing insight into their motivations and expected outcomes. In its current form, it merely states a desire without explaining the user's specific objectives or the benefits they seek from the feature.

Goal and Non-Goal

Goal:
To gather user feedback on the SnickerSync diff tool to understand its usability, effectiveness, and overall user satisfaction.

Non-Goal:
To implement all requested features and improvements from the user studies immediately; the focus is solely on gathering insights for future iterations.

Non-Functional Requirements

1. User Access Control:
Access to the SnickerSync tool must be restricted based on user roles to ensure that only authorized users can modify snickering concepts.

Functional Requirements:
- User Role Management: Implement a role-based access control (RBAC) system that defines user roles (e.g., admin, standard user) and the corresponding permissions for using SnickerSync.
- Permission Verification: Ensure that the system verifies user permissions before allowing access to modify or create snickering concepts in the tool.

2. Random User Assignment for Studies:
The user study must include a mechanism for randomly assigning participants to control groups and variants to ensure unbiased results.

Functional Requirements:
- Randomization Algorithm: Develop a randomization algorithm that assigns users to different study groups while maintaining balance across groups based on user demographics or previous experiences.
- Group Assignment Tracking: Create a system that logs each participant's group assignment and allows for easy retrieval of this information for analysis after the study concludes.

These requirements will help ensure that the SnickerSync tool meets the necessary standards for security and usability while supporting effective user studies.