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 be able to easily navigate the interface and access common git commands so that I can start using it without a steep learning curve.

User story 2: As a team lead onboarding an experienced GiggleGit user, I want to make sure all the previous features are still accessible and easy to find.

User story 3: As a user of GiggleGit, I want to easily share my projects with my team so that we can collaborate without issues

Task: enable project sharing feature

1. Ticket: Create a Project Sharing Guide
    a. Write a simple guide that explains how users can share their projects with others in GiggleGit. Include easy-to-follow steps for inviting team members and managing permissions.
2. Ticket: Implement Project Sharing Function
    a. Develop a feature that allows users to invite team members to collaborate on their projects in GiggleGit. Ensure that users can set different permission levels, such as view-only or edit access, and that the sharing process is straightforward.

As a user I want to be able to authenticate on a new machine. The statement focuses on a specific feature (authentication) rather than an end goal or need. A user story should describe what the user wants to achieve rather than just the functionality itself. This statement lacks the "benefit" part, making it less effective in guiding development.

Formal requirements:

Goal: Conduct user studies to evaluate the usability and effectiveness of the existing vanilla interface for syncing with base GiggleGit packages.

Non-goal: Modify the vanilla interface capable of syncing with the base GiggleGit packages so that it will sync with a snicker.

Non-Functional Requirement 1:

Access Control: Only authorized Product Managers (PMs) should have access to modify and maintain the different snickering concepts within the SnickerSync tool.

Functional Requirement 1.1:

- User Role Management: The system must include a user role management feature that allows the assignment of roles (e.g., admin, PM, user) to individuals based on their access level.

Functional Requirement 1.2:

- Access Permissions: The system must implement access control mechanisms that restrict modification of snickering concepts to users with the PM role, ensuring unauthorized users cannot make changes.

Non-Functional Requirement 2:

Randomized User Assignment: The user study must ensure random assignment of participants between control groups and variants to eliminate bias in the results.

Functional Requirement 2.1:

- Randomization Algorithm: The system must include a randomization algorithm that assigns participants to control groups and variants without bias, ensuring an even distribution of users across groups.

Functional Requirement 2.2:

- Tracking Assignments: The system must maintain a log of user assignments to each group, enabling easy review and verification of the randomization process during and after the study.