1.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 have an easy-to-follow tutorial or guide to help me understand how merges are managed using memes so I can quickly start using the system without confusion.

Task:
Create a user guide for GiggleGit

Ticket 1:
Title: Write clear and concise documentation for merging with memes
Details: Write a step-by-step guide that walks through the core functionalities of GiggleGit with special focus on how meme-based merges work. Include screenshots or visuals where necessary to enhance understanding.

Ticket 2:
Title: Develop an in-app tutorial for first-time users
Details: Implement a built-in interactive tutorial that guides users through setting up their first repository, making commits, and understanding meme-based merges. The tutorial should be integrated into the interface for seamless onboarding.

User Story 2:

As a team lead onboarding an experienced GiggleGit user, I want to be able to easily add new team members to our project so they can start using the version control system without complex configurations.

Task:
Simplify user onboarding for team leads

Ticket 1:
Title: Streamline user invite system
Details: Create a feature that allows team leads to invite new users via email or username. New users should receive an email invitation with a link to join the project and be directed to an onboarding flow upon first login.

Ticket 2:
Title: Pre-configure repository settings for new users

Details: Automatically apply standard repository settings (like branch protection, meme-merge rules) for new users when they join a project to reduce manual configuration for the team lead.

User Story 3:

As a project manager, I want to be able to track the history of merges and see which memes were used in the process so I can ensure accountability within the team.

Task:
Enable merge history tracking

Ticket 1:
Title: Implement meme-merge history tracking
Details: Develop a feature that records the memes used in each merge and displays them in a log alongside the merged commits. This should be easily accessible from the repository history page.

Ticket 2:
Title: Provide export option for merge history
Details: Allow users to export the merge history, including the memes, as a CSV or PDF file for external analysis or documentation purposes.

Non-user Story Explanation:

The statement "As a user I want to be able to authenticate on a new machine" is not a user story. This is because it lacks the "why" or the end goal of the action. A proper user story should include the user's intent or value, such as: "As a user, I want to be able to authenticate on a new machine so that I can access my repositories securely from different devices."

2.Formal Requirements

Goal:

Provide a seamless, entertaining experience during merges that synchronizes version control changes with lighthearted, meme-based feedback.

Non-goal:

Ensure meme relevance or quality across different contexts.

Non-Functional Requirement 1: Security

Description: Ensure secure access control for SnickerSync users to prevent unauthorized usage or data tampering.

Functional Requirement 1:
Only authenticated users should be able to access the SnickerSync merge feature. Implement OAuth2 authentication for access to SnickerSync services.

Functional Requirement 2:
Enforce role-based access control (RBAC) so only project maintainers can modify SnickerSync settings, while contributors can only use the existing meme-based sync options.

Non-Functional Requirement 2: Randomization for User Studies

Description: Ensure users are randomly assigned to either the control or variant group for testing different snickering experiences.

Functional Requirement 1:
Develop a random assignment algorithm to split users between control and experimental groups. This should be triggered upon the user's first login to the study.

Functional Requirement 2:
Track which users were placed in which groups by logging assignments in a secure database, ensuring data consistency for future analysis. This log should also allow administrators to reassign users if necessary for troubleshooting.