Agile

1) Complete the User Stories:
    a) **As a vanilla git power-user that has never seen GiggleGit before, I want to** seamlessly integrate GiggleGit into my existing workflows so that I can manage repositories using familiar Git commands enhanced with meme-based merge management.
    b) **As a team lead onboarding an experienced GiggleGit user, I want to** have access to comprehensive onboarding tools and documentation so that I can efficiently train my team members and ensure smooth adoption of GiggleGit's unique features.
2) Create a third user story:
    a) User Story: As a developer, I want to customize the memes used during merge conflicts so that the merging process feels more engaging and aligns with my team's culture.
    b) Task: Enable Meme Customization
    c) Ticket 1: Develop a Meme Library Interface
        i) Create a user-friendly interface that allows users to browse, select, and preview memes for managing merge conflicts. Ensure the interface supports adding new memes and categorizing existing ones for easy access.
    d) Ticket 2: Implement a Meme-Selection API
        i) Build a backend API that handles the retrieval and storage of user-selected memes. Ensure the API integrates seamlessly with the frontend interface and supports future scalability for additional meme categories.
3) **As a user, I want to be able to authenticate on a new machine.**
    a) This is not a user story. This statement lacks specific context regarding the user's role and the underlying motivation or benefit. A proper user story should follow the format: "As a [user role], I want to [goal] so that [reason]." Without specifying who the user is and why authentication on a new machine is important, the statement remains an incomplete requirement.
    b) This statement is a **functional requirement** or a **feature request** that specifies a desired functionality, but does not encapsulate the user's perspective or the value it provides.

Formal Requirements

1) List one goal and one non-goal:
    a) Goal: Conduct comprehensive user studies to evaluate the effectiveness, usability, and user satisfaction of SnickerSync, the new diff tool, in managing merge conflicts using memes.
    b) Non-Goal: Develop advanced customization features for SnickerSync beyond the initial vanilla interface, such as user-generated meme uploads or integration with third-party meme repositories.
2) Two non-functional requirements:
    a) Access Control:
        i) Ensure that only authorized participants can access the SnickerSync user study interfaces and related data to maintain the integrity and confidentiality of the study.
    b) Maintainability:
        i) Allow Project Managers to easily update and manage different snickering concepts without requiring significant changes to the underlying codebase or disrupting the user interface.
3) Two functional requirements for each non-functional requirement:
    a) Access Control:
        i) Implement user authentication mechanisms, including multi-factor authentication, to verify the identity of the participants before granting access to the SnickerSync user study.
        ii) Develop role-based access controls (RBAC) to restrict specific functionalities and data access based on participant roles, ensuring that only designated users can modify study parameters or view sensitive data.
    b) Maintainability:
        i) Create an administrative dashboard that allows PMs to add, edit, or remove snickering concepts dynamically without requiring code deployments, facilitating quick updates based on user feedback.
        ii) Design the snickering concept module using a modular architecture, enabling easy integration of new meme types and concepts while ensuring compatibility with existing features and minimizing potential disruptions.