Agile

Complete the following tasks:

1. Complete these user stories:
    a. As a vanilla git power-user that has never seen GiggleGit before, I want to be able to track and manage changes to files over time with ease, just like in vanilla git.
        i. Task: Ensure basic git compatibility
            1. Ticket 1: Implement core git commands
                a. Ensure GiggleGit supports core git commands such as merge, pull and push. Write tests to ensure they behave correctly.
            2. Ticket 2: Create a simple tutorial
                a. Make a simple tutorial showing how GiggleGit works similarly to vanilla git
    b. As a team lead onboarding an experienced GiggleGit user, I want to be able to add and remove memes from the merge meme library for my team
        i. Task: Allow customization of meme libraries
            1. Ticket 1: Design and create a database for storing uploaded memes
                a. Use something like MongoDB
            2. Ticket 2: Implement different levels of access controls
                a. Create a "Team Lead" role with administrator privileges for managing the meme library.

2. Create a third user story, one task for this user story, and two associated tickets.
    a. Tasks should be a single phrase. (As should themes and epics. See those provided.)
    b. User stories should be one to three sentences.
    c. Tickets should have a title consisting of a single phrase and details that are long enough to sufficiently describe what needs to be done. You do not need to assign points to the tickets

User story 3:

- As a developer collaborating on a team project, I want to be notified when a merge conflict is resolved using a meme
    - Task: Add notifications
        - Ticket 1: Implement notification system (email, ping) for conflict resolution
        - Ticket 2: Customize notification to include meme image

3. This is not a user story. Why not? What is it?
    a. As a user I want to be able to authenticate on a new machine
        i. This is a requirement. It lacks the "who" and the "why" that user stories require for context.

Formal Requirements

Complete the following tasks:

1. List one goal and one non-goal
    a. Goal: Allow users to sync with a snicker in a way that improves collaboration and is more exciting than basic version control.
    b. Non-Goal: Suggest code optimizations when syncing
2. Create two non-functional requirements. Here are suggestions of things to think about
    a. Who has access to what
    b. PMs need to be able to maintain the different snickering concepts
    c. A user study needs to have random assignments of users between control groups and variants
3. For each non-functional requirement, create two functional requirements (for a grand total of four functional requirements).

Non-functional requirements:

- Permissions and access control
    - Functional requirements:
        - Ensure secure handling of merges by only allowing authorized users to sync
        - Implement different user roles: admin, developer, etc.
- User study setup
    - Functional requirements:
        - Randomly assign users between control groups and variants
        - Track user data within each variant