Agile

1. Complete these user stories:
    - As a vanilla git power-user that has never seen GiggleGit before, I want to…
        i. Explore GiggleGit's features with a low learning curve so I can decide if I want to adopt it.
    - As a team lead onboarding an experienced GiggleGit user, I want to…
        i. Have access to clear documentation and training materials so I can efficiently onboard my team members

2. Create a third user story, one task for this user story, and two associated tickets.
    - Tasks should be a single phrase. (As should themes and epics. See those provided.)
    - User stories should be one to three sentences.
    - 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: As a team member onboarding to GiggleGit for the first time, I want to have a guided tutorial on the platform so I can quickly learn how to perform version control so my workflow is impacted minimally.
    - Task: Implement guided tutorial for first-time users
        - Ticket 1: Create a basic tutorial
            - Create a step-by-step guide that introduces users to key features of GiggleGit
        - Ticket 2: Add progress tracking for tutorial
            - Implement functionality that allows users to track their progress through the tutorial and allow them to resume their progress where they left off if they exited the tutorial

3. This is not a user story. Why not? What is it?
    - As a user I want to be able to authenticate on a new machine

This isn't a user story because a purpose or why the user wants this functionality is not given. This is a requirement description.

Formal Requirements

Goal: Launch SnickerSync with different sync behaviors for testing. This will gather feedback on how “sync with a snicker” improves or detracts from the user experience

Non-Goal: Optimizing the performance of SnickerSync for large repositories

Non-Functional Requirement 1: User Study Access Control

- Only users in designated control groups should access the SnickerSync feature set based on their assignment
- Functional Requirement 1: Ensure users are randomly assigned to a control or variant group. Ensure they can only access behaviors applicable to their group
- Functional Requirement 2: Provide a mechanism for PMs to monitor and adjust group assignments

Non-Functional Requirement 2: SnickerSync Management

- PMs need to be able to configure different SnickerSync behaviors without needing developer intervention
- Functional Requirement 1: Create a backend interface for PMs to modify sync behavior and apply them to users
- Functional Requirement 2: Implement a logging system to track changes made, ensuring that results from different groups and behaviors can be correlated