CodeChuckle is a startup whose product is GiggleGit, a version control system “where merges are managed by memes."

Theme: Get GiggleGit demo into a stable enough alpha to start onboarding some adventurous clients
Epic: Onboarding experience

User Stories:

- *As a vanilla git power-user that has never seen GiggleGit before, I want to be be confident using GiggleGit as a tool.*

- *As a team lead onboarding an experienced GiggleGit user, I want to access the previous repo’s of this user.*

Create a third user story, one task for this user story, and two associated tickets:

As a brand new user who is unfamiliar with vanilla git, I want a quick introduction into GiggleGits functionality and syntax

***Task**: Implement a tutorial for GiggleGit’s most important functions*

Ticket 1: Create a tutorial
Create a file with explanations for all of GiggleGit’s primary functions sorted from most useful to least

Ticket 2: Make the tutorial accessible
Create a function that pulls up a list when the command “gigglegit help” is entered.

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

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 story has nothing to do with a specific user, it is true for all users who want to use the software. This makes it a functional requirement.*

1. List one goal and one non-goal
    - *Goal: Gain insight about what user's have common issues with in the SnickerSync interface.*
    - *Non-Goal: Implement new features based on user feedback*
2. Create two non-functional requirements. Here are suggestions of things to think about:
    - **Who has access to what**
    - **PMs need to be able to maintain the different snickering concepts**
    - **A user study needs to have random assignments of users between control groups and variants**
    - non-functional requirement 1: Randomness
        i. Assign users to control and variant groups using a randomly generated id value.
        ii. Ensure proper ratio of control participants to variant.
    - non-functional requirement 2: Privacy
        i. Implement secure user accounts that dont leave important information exposed
        ii. Create a database for users to submit feedback, sorted by control or variant group

3. For each non-functional requirement, create two functional requirements (for a grand total of four functional requirements).