Agile

CodeChuckle is a startup whose product is GiggleGit, a version control system “where merges are managed by memes." (It saddens me to say that this was a joke written by ChatGPT for 131)

You have just been hired as employee number n for some small number n. They have the dev chops to make a demo, but you are their first serious developer.

Here is a theme and an epic:

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

Complete the following tasks:

Complete these user stories:

- As a vanilla git power-user that has never seen GiggleGit before, I want to…
  Understand GiggleGit features quickly and easily
    - Task: display easy-to-understand features or a guide
        - Ticket 1: make an interactive tutorial for first-time users that walks through the important features and interface.
        - Ticket 2: write a clear documentation for troubleshooting or general information, that includes FAQs
- As a team lead onboarding an experienced GiggleGit user, I want to…
  Get my team set up with the GiggleGit demo and track/monitor progress
    - Task: provide team management features
        - Ticket 1: allow PM to monitor all employees, each of their progress and completion of tutorial
        - Ticket 2: allow PM to invite employees to projects

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

- As a user in a group that appreciates a specific kind of humor, I want to…
  Add more personalized and niche memes
    - Task: enable custom meme inputs
        - Ticket 1: create a meme upload feature as images or gifs
        - Ticket 2: create a library of all uploaded custom memes for organization or categorization

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 is not a user story because it is not specific to a particular kind of user and their specific wants based on context and situation. Instead, it is an overall feature that the product needs to deliver for all users. Therefore, it is a *requirement* instead.

Formal Requirements

CodeChuckle is introducing a new diff tool: SnickerSync—why merge in silence when you can sync with a snicker? The PMs have a solid understanding of what it means to "sync with a snicker" and now they want to run some user studies. Your team has already created a vanilla interface capable of syncing with the base GiggleGit packages.

Complete the following tasks:

List one goal and one non-goal

Goal: Create an easy-to-use tool that plays a snicker every time there is a merge.

Non-goal: Making sure a merge is successful and snickering based on success.

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

- NFR 1: Different roles have different levels of access
    - FR 1: PMs should be able to access and control snickering concepts/features/behavior
    - FR 2: Employees/regular users shouldn't have access to hidden features, only basic features
- NFR 2: Scalability
    - FR 1: The system should be able to handle large amounts of users without performance drops and bugs
    - FR 2: The system should be able to maintain low response times for all users