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:

1. Complete these user stories:
   - As a vanilla git power-user that has never seen GiggleGit before, I want to…

     have as many similar features to vanilla git so I can return to power-using GiggleGit as fast as possible

   - As a team lead onboarding an experienced GiggleGit user, I want to…

     have control over when memes can appear as to not side track employees during multiple merges.

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

   As a Project Leader I want to be able to see which employees have completed obboarding, and at which time. This is in order to ensure that the schedule is met and our team is functioning in a timely manner.

Ticket 1: Give permission structure that allows a project manager to view all onboarding success.

Ticket 2: Implement schedule by which each team can base deadlines off of.

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 is not a user story as they have not explained why they want what they want. Instead this is a requirement which the user would like the company to implement.

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:

1. List one goal and one non-goal

    Goal: Enhance fun for each employee by combining the memes of GiggleGit with the snickering of SnickerSync

    Non-Goal: Introduce a function which allows the boss to observe each snicker recieved.

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
        1) A security feature that only allows people with team permissions to operate and view certain code.

        a) It must check users permissions before they are allowed to access anything
        b) Permission can be given and rescinded based on changes in teams and status
    2) Track the speed which random users perform the same basic functions in SnickerSync vs in GiggleGit to ensure that functionality is not lost by the transition.
        a) The system must track typing speed and lines written
        b) The system must generate reports based on the increase and decrease in these times

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