Agile

- **User Story 1:** As a vanilla git power-user that has never seen GiggleGit before, I want to quickly learn the differences between Git and GiggleGit so that I can integrate it into my workflow efficiently.
- **User Story 2:** As a team lead onboarding an experienced GiggleGit user, I want to streamline the onboarding process so the user can start contributing to the project quickly without unnecessary redundant training on features with chich they are already familiar.
- **User Story 3:** As a new user, I want to access GiggleGit securely from multiple devices so that I can work from anywhere without compromising security.
    - **Task:** Implement multi-device authentication
        - **Ticket 1:** Design the user interface for multi-device authentication.
            - Allow users to authenticate on new devices using multi-factor authentication.
        - **Ticket 2:** Implement multi-device authentication logic in the backend.
            - Ensure the system recognizes and securely handles multiple device logins while preserving session data.
- **Not a User Story:** As a user I want to be able to authenticate on a new machine
    - This is not a valid user story because it lacks a clear reason or benefit. User stories should express the value the feature brings to the user.

Formal Requirements

- **Goal:** Enable users to use SnickerSync to compare and merge code while receiving humorous feedback.
- **Non-Goal:** Develop an advanced AI to generate humorous feedback
- Non-Functional Requirement 1: Security
    - Functional Requirement 1: Restrict access to SnickerSync settings based on the role of the person trying to access the settings
        - Only PMs should have access
    - Functional Requirement 2: Log all changes made to SnickerSync settings
        - All changes should be recorded in a log only accessible by administrators
- Non-Functional Requirement 2: Usability for User Studies
    - Functional Requirement 1: Implement a user assignment system that randomly allocates users between control and variant groups for SnickerSync studies.
        - Users should be evenly distributed between control group (regular merging) and variant group (merging with a snicker)
    - Functional Requirement 2: Create a user study dashboard for PMs to track user assignment and view results/feedback
        - he dashboard should display which users are in which group, and collect feedback on their experiences during code merges.