Agile

1. User stories:
    - As a vanilla git power-user that has never seen GiggleGit before, I want to ensure that my existing projects that use traditional git can be able to seamlessly transit to GiggleGit without altering the project structure or workflow, and the standard version control operation remains (e.g. commit, push, clone) so I can continue managing my repositories as before while exploring the new features with GiggleGit.
    - As a team lead onboarding an experienced GiggleGit user, I want to make sure that the GiggleGit's mem-based merging feature is easy to understand, or has some references to traditional git (commands), so that I can help my team smoothly adapt to use GiggleGit and maintain efficiency on our projects without spending too much time on learning the functionality of each specific memes.

2. The third user story, one task for this user story, and two associated tickets:

    User story:
    As a developer who needs to frequently update project versions across multiple branches, I want to customize my meme that correspond to one or multiple version control commands (git commands), so I can quickly identify and execute version-related tasks, reducing the time spent switching between branches and tagging versions.

    - **Task:** Allow customized meme that automatically performs a specific or series of version control commands when used.
        i. **Ticket 1:** Implement a meme command editor that allows user to map their customized meme to some specific version control commands.
        ii. **Ticket 2:** Implement a validation process that checks the validity of the set of commands mapped by the customized memes to avoid conflict commands, failures, or file damages. (Display the set of commands that are about to be mapped and ask for confirmation from the users before creating the customized meme.)
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 because it does not inform who the user is, rather, it only says "the user". Also, this sentence does not specify why that specific user (or group of users) wants to "be able to authenticate on a new machine". In short, it lacks the viewpoint of the specific user. This sentence is just a simple requirement raised to the GiggleGit project that only emphasizes what needs to be done.

Formal Requirements

Goal: Gather feedback from users on the effectiveness of SnickerSync (with different snickering concepts) by conducting user studies with randomized control and variant groups.

Non-goal: Additional features on the current SnickerSync interface should not be developed during user studies.

Non-Functional Requirements 1: There need to be an efficient and convenient Access control for different identity during the user study.

- **Functional Requirement 1:** Implement an access control system that gives different access based on identity, so that different people will have their corresponding accesses (e.g. PMs, developers, test users…)
- **Functional Requirement 2:** A simple management system/interface where the PMs can manage user assignments, update different snickering concepts, and monitor the user study.

Non-Functional Requirements 2: Ensure the validity of the user study (user study needs to be unbiased and needs to be persistent on the given version of SnickerSync to each individual)

- **Functional Requirement 3:** For a user study to be valid, the control and variant group should be randomly assigned, so we can implement an algorithm that randomly and proportionally assigns test users into control or variant groups.
- **Functional Requirement 4:** The system used for user study needs to keep track of the control and variant group assignment of the user during the entire user study to avoid re-assignment so that the users consistently use the same version of SnickerSync based on the assignment.