**Agile**

1. a. As a vanilla git power-user that has never seen GiggleGit before, I want to easily understand and use basic version control features so that I can manage my code repositories much more effectively and efficiently.
2. As a team lead onboarding an experienced GiggleGit user, I want to quickly show my team how to use advanced GiggleGit features to let my team can easily understand the features and simplify our collaborative development process.
3. **User Story 3:** As a new GiggleGit user, I want to set up my development environment easily, so that I can begin contributing quickly.

**Task for this user story:** Set up development environment for GiggleGit.

**Ticket 1:**

**Title:** **"Installation Guide for GiggleGit Setup"**

**Details:** Create a step-by-step guide for setting up GiggleGit on different operating systems (Windows, MacOS, Linux).

**Ticket 2:**

**Title:** **"Configuration of GiggleGit on Local Machine"**

**Details:** Provide instructions on configuring GiggleGit settings such as authentication keys, repository access.

1. Because a user story typically describes a need or feature from the user's perspective and includes the value or benefit it brings to the user, this statement only talks about an action rather than a broader need or problem the user wants to solve. This is more likely a requirement about the product of function of access control or authentication.

**Formal Requirements**

1. **Goal:** Enable SnickerSync to sync with the base GiggleGit packages to improve control efficiency for users’ changes.

**Non-Goal:** SnickerSync will not replace traditional merging functionalities within GiggleGit completely. It should complement them with additional visual and interactive tools.

1. **Access Control:** To limit access based on permissions, the system needs user roles. Access rights and snickering concepts can only be changed by users with admin roles. **User Assignment:** To enable control and variant groups for user studies, the system should allow users to be randomly assigned to various snickering concepts.
2. Create Functional Requirements for Each Non-Functional Requirement

* For Access Control:
  + **Functional Requirement 1:** Establish an admin dashboard where authorized users can control user roles and permissions to use SnickerSync features.
  + **Functional Requirement 2:** . Create a login system that applies access controls according to user roles, ensuring that the snickering concepts can only be altered by authorized users.
* For User Assignment:
  + **Functional Requirement 1:** Implement an randomization algorithm that assigns users to different snicker variants during user studies.
  + **Functional Requirement 2:** Create a user study management interface that allows product managers to view and adjust user assignments across control and variant groups.