**HW1 Requirements**

Aishwari Krishna

**Agile**Theme:Get the GiggleGit demo into a stable enough alpha to start onboarding some adventurous clients.

Epic:Onboarding experience.

User Stories:

1. As a vanilla git power-user that has never seen GiggleGit before, I want to understand how GiggleGit differs from traditional Git, so that I can see how it fits into my workflow.
2. As a team lead onboarding an experienced GiggleGit user, I want to ensure they have a quick and efficient setup process, so that they can contribute to projects without delay.
3. As a new user interested in GiggleGit, I want to access a guided tutorial with examples, so that I can quickly learn how to use key features.

Task for User Story 3: Develop an interactive tutorial for onboarding new users.

Tickets:

1. Create a step-by-step tutorial for new users.

Design and implement an onboarding guide that includes step-by-step instructions for setting up and using GiggleGit. The tutorial should cover installation, basic commands, and merging via memes.

1. Implement an interactive demo environment.

Set up a sandbox environment where new users can practice GiggleGit commands without affecting real repositories.

Not a User Story Explanation: “As a user, I want to be able to authenticate on a new machine.”

This is not a user story because it lacks a clear user motivation and benefit — a proper user story should explain why authentication is needed and how it benefits the user. Instead, it is a feature request because it describes a function of the system but does not explain why the user needs it or how it benefits them. A better version would be: “As a frequent traveler, I want to authenticate my GiggleGit account on a new machine, so that I can continue my work seamlessly from different locations.”

**Formal Requirements**

Goal: Ensure SnickerSync provides a seamless, secure, and intuitive syncing experience for GiggleGit users.

Non-Goal: Support for third-party version control systems beyond GiggleGit.

Non-Functional Requirements:

1. Maintainability: SnickerSync should be easy to update and manage as new snickering concepts are introduced.
2. User study integrity: the system must ensure that user study assignments are fair, unbiased, and prevent tampering.

Functional Requirements:

Related to maintainability:

1. Modular configuration: implement a configuration management system that allows PMs to add, modify, or remove snickering concepts without requiring developer intervention.
2. Version control for snickering concepts: maintain a history of snickering concept changes, allowing PMs to roll back to previous versions if needed.

Related to user study integrity:

1. Randomized assignment: automatically assign users to control and experimental groups using a cryptographically secure randomization algorithm.
2. Data logging and auditability: maintain an immutable log of all user study assignments and interactions to ensure transparency and reproducibility in A/B testing.