**Agile slide:**

* Theme: Get GiggleGit demo into a stable enough alpha to start onboarding some adventurous clients
* Epic: Onboarding experience

1. Complete these user stories
   1. As a vanilla git power-user that has never seen GiggleGit before, I want to be able to pick it up quickly and not have to have a long adjustment period.
   2. As a team lead onboarding an experienced GiggleGit user, I want to be able to display the benefits of GiggleGit easier in a presentation format or quick crash course training video.
2. Create a third user story, one task for this user story, and two associated tickets.
   1. As someone who has never learned git before, I want to be able to learn basic git functions quickly while enjoying the memes for merging.
      1. Ticket 1:
         1. Title: Git Commands that Everyone Should Know
         2. Details: Lists the most commonly used git commands, what they do, and what parameters they take.
      2. Ticket 2:
         1. Title: Downloading GiggleGit
         2. Details: Shows a step-by-step guide on how to download GiggleGit and how to configure it to your needs.
3. This is not a user story. Why not? What is it?
   1. As a user I want to be able to authenticate on a new machine
      1. This is not a user story because it only tells us about the issue they are facing but not their story. This can only qualify to be a requirement in a user story but not their experience and what they need.

**Formal Requirements:**

1. List one goal and one non-goal
   1. Goal: Create an easy-to-use interface that users can use to “sync with a snicker” with the base GiggleGit packages
   2. Non-Goal: SnickerSync has a way for users to input their thoughts and feedback so that we can improve.
2. Create two non-functional requirements
   1. Accessibility:
      1. Users should not have full access to the entire program. CodeChuckle employees should have admin permissions.
   2. Random Assignments:
      1. Different users should have different snickers to see which one is the most effective.
3. For each non-functional requirement, create two functional requirements (for a grand total of four functional requirements).
   1. Accessibility
      1. There needs to be an admin portal and user portal that separates the workers and the users testing things.
      2. Admin should be able to see user history and their interactions to weed out useless and useful feedback
   2. Random Assignments:
      1. Based on collected user data, personalize each person’s experience to the snicker they would probably like the most
      2. Allow admins to have a quick and easy way to collect data on each snicker used such as most liked snicker, trending snickers, or user added snickers.