Agile

1.

   a. As a vanilla git power-user that has never seen GiggleGit before, I want to understand how it works as quickly as possible. I do not want to spend a lot of time learning a new version control system because I value my time. This will help me stay productive with minimal disruption

   b. As a team lead onboarding an experienced GiggleGit user, I want the process to be straightforward. I need to ensure the experienced GiggleGit user adapts to the version control system as quickly as possible. This will allow them to contribute with minimal disruption.

2.

   - **Third User Story**: As a new user of GiggleFit, I want to access a comprehensive tutorial. This will help me understand the key features and functionalities. I prefer to learn at my own pace to ensure I grasp everything.
   - **Task**: Create a comprehensive tutorial for new users.
   - **Tickets**:
     1. **Develop tutorial content**: Produce content for the GiggleGit tutorial that covers all key features and functionalities. This should include step-by-step instructions, best practices, and tips for efficient use. The content should be structured to facilitate easy navigation for new users.
     2. **Implement tutorial interface**: Implement a user interface for the tutorial with GiggleGit. The interface should enable users to navigate sections easily, bookmark important parts, and track their progress. This will enhance the user experience and support effective learning.

3. The sentence, “As a user, I want to be able to authenticate on a new machine.” is not a user story. The sentence does not talk about the user’s need and the benefit the user is seeking. This is just a non-functional requirement.

Formal Requirements

1.

   - **Goal**: To gather user feedback on the SnickerSync tool to refine its functionality.
   - **Non-Goal**: To develop a version of SnickerSync with advanced features beyond the vanilla syncing capabilities.

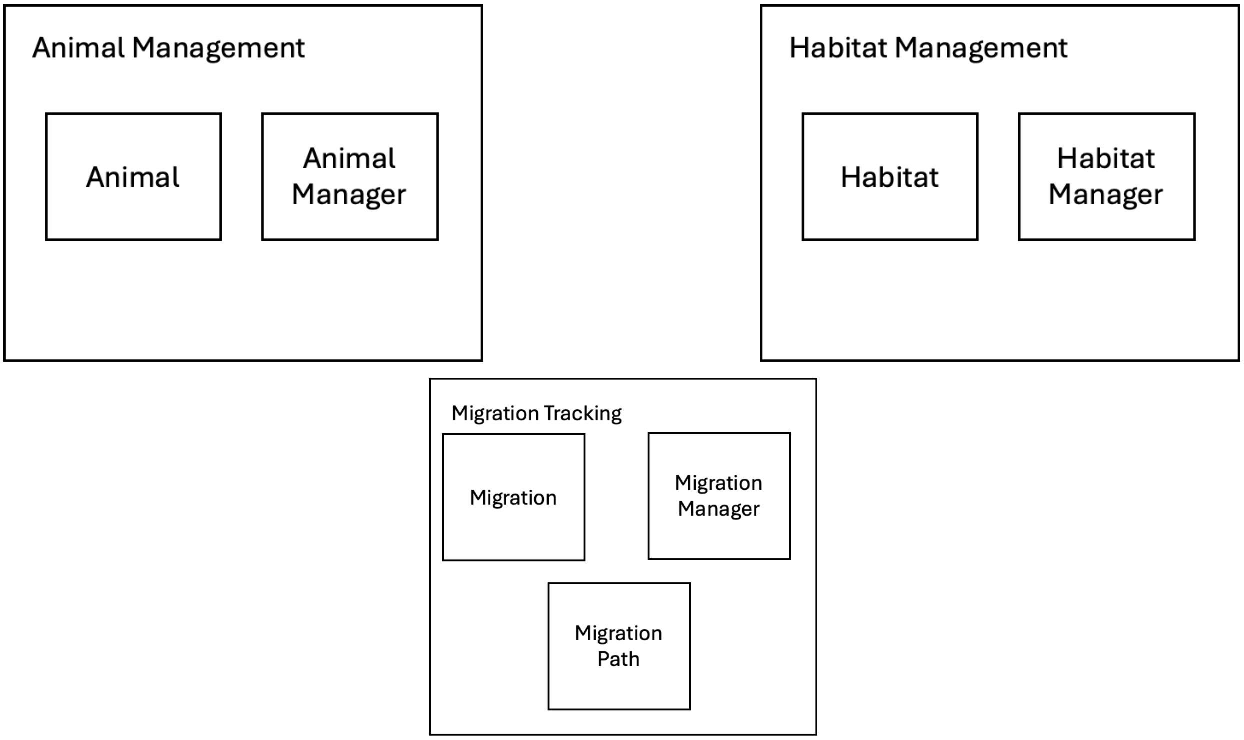
2.

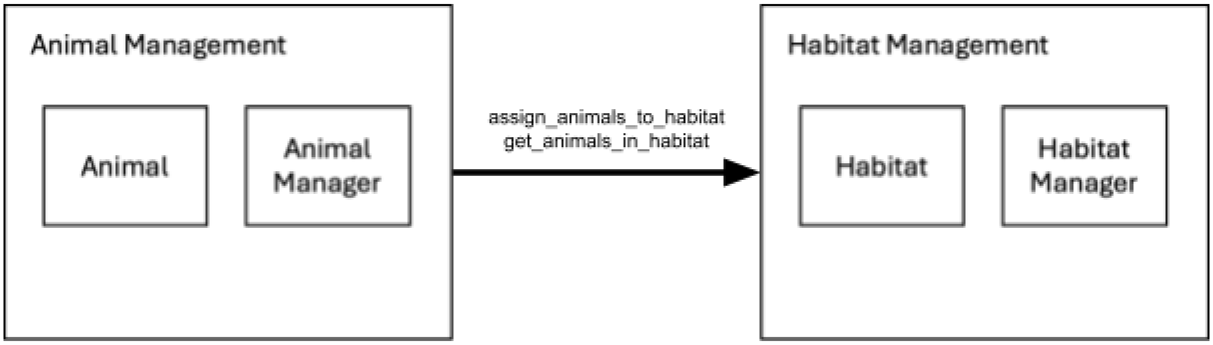
   a. Non-functional requirement 1: Security
   - **Functional requirement 1.1**: SnickerSync must implement secure authentication protocols to ensure that only authorized users can access the tool.
   - **Functional requirement 1.2**: All data transferred during the syncing process must be encrypted to protect against unauthorized access and data breaches.

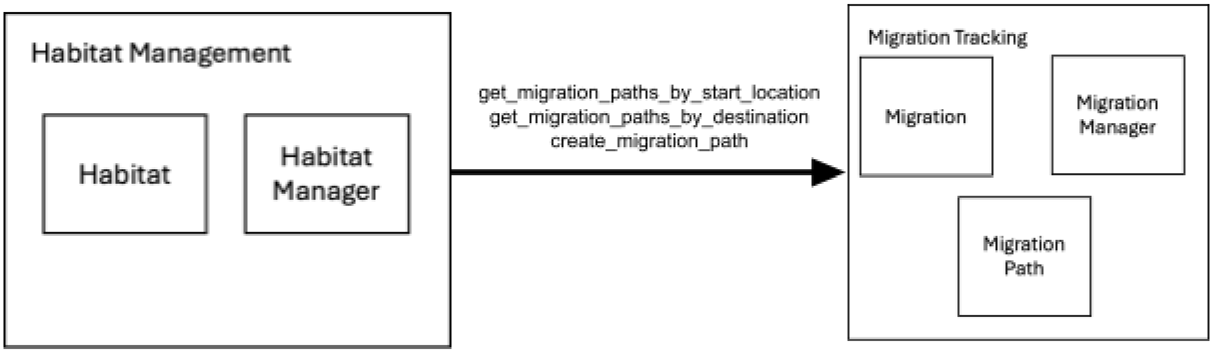
   b. Non-functional requirement 2: Scalability
   - **Functional requirement 2.1**: SnickerSync should be able to handle simultaneous user connections without the risk of performance degradation.

- **Functional requirement 2.2**: The architecture of SnickerSync must allow for easy integration of additional features in the future without requiring significant redesign.

Dependencies


Animal Management
Animal
Animal Manager
Habitat Management
Habitat
Habitat Manager
Migration Tracking
Migration
Migration Manager
Migration Path


Animal Management
Animal
Animal Manager
assign_animals_to_habitat
get_animals_in_habitat
Habitat Management
Habitat
Habitat Manager


Habitat Management
Habitat
Habitat Manager
get_migration_paths_by_start_location
get_migration_paths_by_destination
create_migration_path
Migration Tracking
Migration
Migration Manager
Migration Path