Agile Slides

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

1. **Complete these user stories:**
    a. As a vanilla git power-user that has never seen GiggleGit before, I want to have an easy transition from Git to GiggleGit.
    b. As a team lead onboarding an experienced GiggleGit user, I want to make adding them to my GiggleGit repositories easy.
2. **Create a third user story, one task for this user story, and two associated tickets.**

As a programmer with private projects, I want to GiggleGit to effectively protect my data so I can work on sensitive projects.

a. **Task:** Prevent Data Leakage
    i. **Ticket 1:** Safely separate different data
        - We need to find a way to make sure user data is stored securely so companies and individuals cannot have their new projects leaked.
    ii. **Ticket 2:** Implement 2-factor authentication
        - We need to have 2-factor authentication so it's less likely for user's accounts to be hackable and data stolen.

3. **This is not a user story. Why not? What is it?**
    a. As a user I want to be able to authenticate on a new machine

This is a requirement rather than a user story. That's because it lacks the reason or benefit for this task.

Formal requirements slide

1. **Create one goal and non-goal related to launching user studies for snickersync.**

Goal: Allow users to to synchronize their code changes with an audio feedback (snickers) to improve the user experience.

Non-Goal: To replace existing solutions (GiggleGit). SnickerSync will work with existing tools instead of creating a new one.

Non-Functional Requirement 1:

- **Functional Requirements:** User Control
- The system has to allow the product managers to create, edit, and delete user roles where each role has specific permissions related to control of the Snickers.

- Authentication and Authorization: The system must allow users to log in and authenticate themselves so only authorized users have access to Snickers management.

Non-Functional Requirement 2:

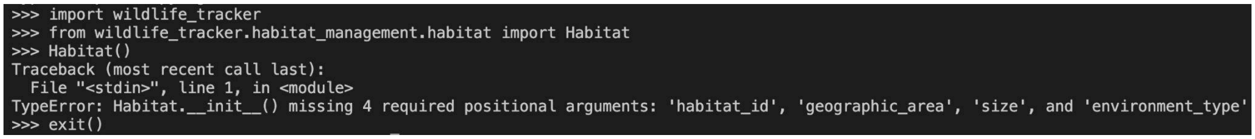
- **Functional Requirements:** Performance
- The system has to respond to user SnickerSync requests within 2 seconds for 99% of all operations.
- The system has to handle up to 5,000 users without slowing down performance or audio output.

Partitioning Functionality

Done

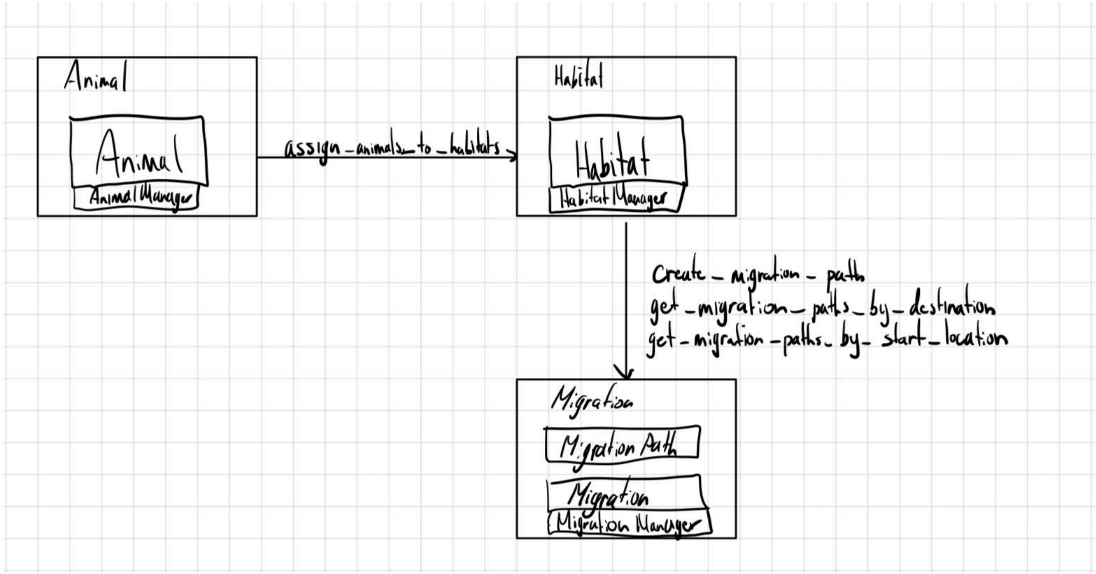
Importable Moduels

Done



```
>>> import wildlife_tracker
>>> from wildlife_tracker.habitat_management.habitat import Habitat
>>> Habitat()
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: Habitat.__init__() missing 4 required positional arguments: 'habitat_id', 'geographic_area', 'size', and 'environment_type'
>>> exit()
```

Dependencies:


Animal
Animal
Animal Manager
assign_animals_to_habitats
Habitat
Habitat
Habitat Manager
Create_migration_path
get_migration_paths_by_destination
get_migration_paths_by_start_location
Migration
Migration Path
Migration
Migration Manager