Agile

1. Complete User Stories
    - As a vanilla git power-user that has never seen GiggleGit before, I want to have clear documentation highlighting the differences between Git and GiggleGit, so that I can justify switching over and quickly adapt to the changes without disrupting my existing workflows.
    - As a team lead onboarding an experienced GiggleGit user, I want to provide them with our team's workflow guidelines, so that they can integrate seamlessly and collaborate effectively from day one.
2. Junior Developer New to Git
    - As a junior developer new to git, I want access to interactive tutorials and support resources, so that I can learn to use GiggleGit efficiently and contribute to projects ASAP.
    - **Task**: Develop onboarding resources for new GiggleGit users
    - **Ticket 1**: Create Interactive Onboarding Tutorial
        - Develop an engaging, step-by-step interactive tutorial within GiggleGit that gives an introduction to new users on basic commands and unique features like meme-managed merges. Tutorial should help users become more comfortable with the interface and main functionalities quickly.
    - **Ticket 2**: Create Comprehensive Onboarding Documentation
        - Write detailed onboarding docs such as setup guides, FAQs, and best practices for GiggleGit. Ensure documentation is easy to understand and accessible, helping new users in resolving common problems and learning advanced features at their own pace.
3. It is not a complete user story because it lacks the “so that…” clause that explains the benefit or motivation behind the user’s need. This would be a task.

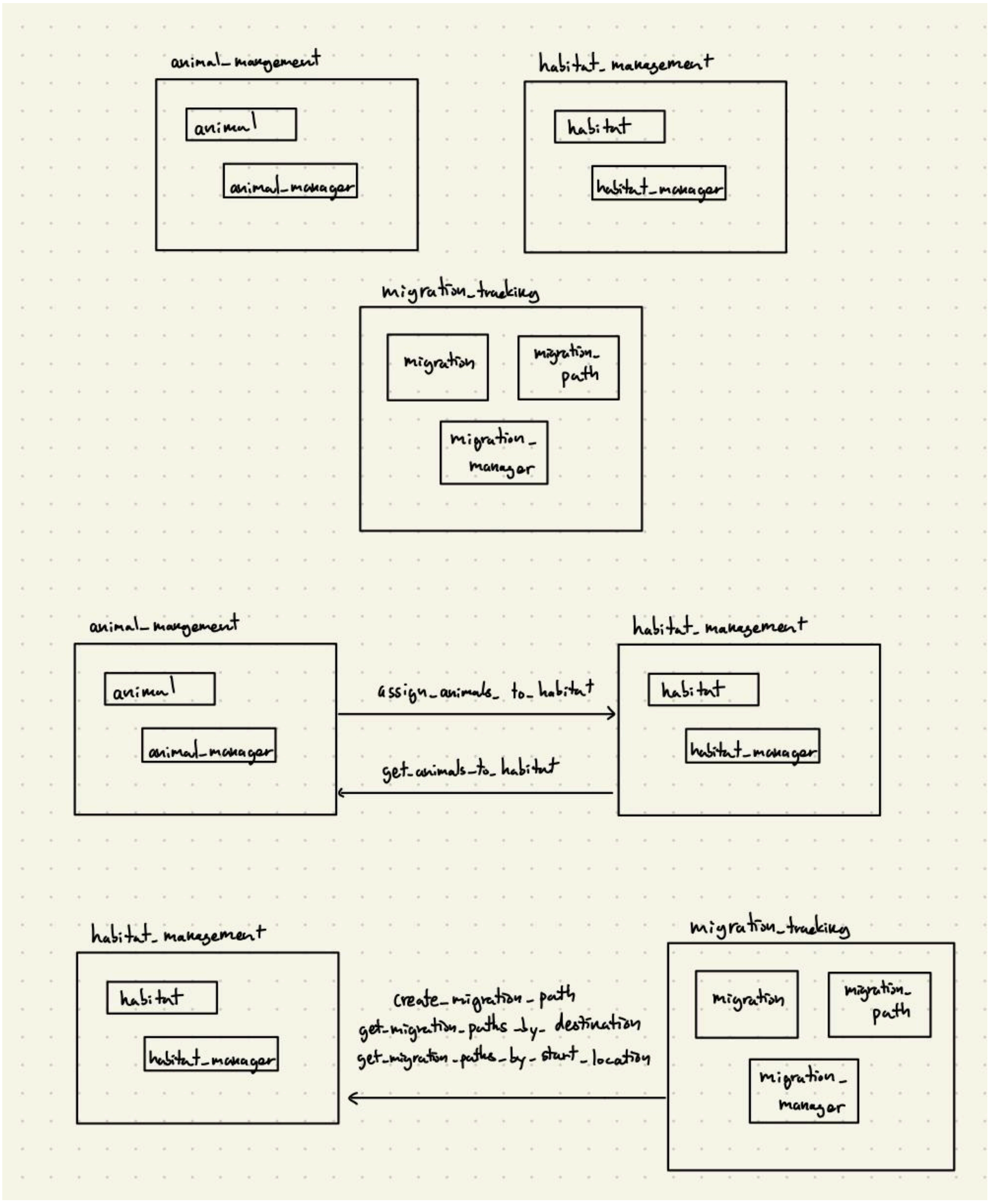
Formal Requirements

1. Goal and Non-Goal
    - Integrate SnickerSync into the GiggleGit packages and conduct user studies through questionnaire/response form to evaluate its usability and effectiveness. Add the form somewhere on the website or/and as a command in the CLI.
    - SnickerSync won’t focus on supporting third-party diff tools or overhauling existing GiggleGit architecture beyond what is necessary for SnickerSync integration.
2. NFRs
    - **Requirement 1**: Access Control
        - The system must ensure that only authorized users have access to specific features and data related to SnickerSync and the user studies.

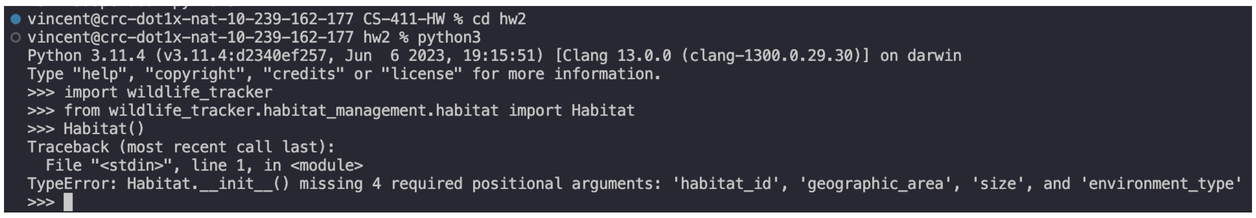
- **Requirement 2**: Random Assignment in User Studies
    - The system must randomly assign users to control groups and variant groups during user studies to ensure unbiased and statistically significant results.

3. Functional Requirements for NFRs
    - NFR1
        1. **Requirement 1**: User Auth
            - The system will require users to authenticate with valid credentials (Auth0 SSO) before granting access to SnickerSync features.
            - The system will implement role-based access control, allowing admins to assign roles (e.g., PMs, developers, researchers) with specific permissions to users.
        2. **Requirement 2**: Role-Based Permissions
    - NFR2
        1. **Requirement 1**: Randomization Algorithm
            - The system will utilize a randomization algorithm to assign incoming users to either the control group or one of the variant groups automatically
        2. **Requirement 2**: Assignment Logging
            - The system will record each user’s group assignment in a secure database (say Oracle) to maintain consistency and integrity of the user study data.

Dependencies



Screenshot of Python Script



```
● vincent@crc-dot1x-nat-10-239-162-177 CS-411-HW % cd hw2
○ vincent@crc-dot1x-nat-10-239-162-177 hw2 % python3
  Python 3.11.4 (v3.11.4:d2340ef257, Jun  6 2023, 19:15:51) [Clang 13.0.0 (clang-1300.0.29.30)] on darwin
  Type "help", "copyright", "credits" or "license" for more information.
  >>> 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'
  >>> 
```