# Plan of approach

This document explains how to integrate work from all teams in Unity using GitHub. It shows the steps for programming, art, sound, and gameplay integration.

## Recommended Tools

- **GitHub Desktop (Recommended)**

  User-friendly interface, easier for beginners.

- **Command Line (Optional)**

  Only recommended for experienced Git users.

## General Integration Steps

1. **Clone the repository**

   Only needed once when starting the project.

2. **Always pull the latest changes**

   Pull the latest version of the `dev` branch before working.

3. **Create your own branch**

   Each feature or task must have its own branch.

4. **After finishing your work:**

   - Test in Unity
   - Commit your changes
   - Push your branch to GitHub

5. **Create a Pull Request (PR)**

   PR should merge your branch into `dev`.

6. **Code review**

   Another team member checks and approves the PR before merging.

---

# Programming Integration

1. **Switch to the programmer's branch**

   Their feature should be in a separate branch.

2. **Open Unity and check for conflicts**
   - Unity may reimport assets or scripts
   - Fix compile errors if needed

3. **Continue their work**
   - Add missing code or functionality
   - Clean up scripts
   - Follow tasks listed in GitHub
   - **Note:** Programmers handle combining animations with assets under the supervision of the Art Integrator.

4. **Write clear commit messages**
   - Explain changes (don’t just write “fix”)

5. **Create a Pull Request**
   - Tag someone for review if possible

---

# Art Integration

1. **Switch to the art branch**

   Their assets, textures, models, or animations will be located there.

2. **Check the art files**
   - Ensure folder structure is correct
   - Confirm file names follow the project’s naming conventions

3. **Continue their work**
    - Prepare assets for programming or gameplay use
    - Update art documentation:
        - File locations
        - Prefab/animation names
        - How/where the asset should be used
    - **Note:** Art Integrator supervises asset usage in the scene but usually does not assemble the full scene themselves. Full scene assembly is a collaborative task with programming and gameplay integrators.
4. **Write clear commit messages**
    - Example: “Added player run animation files + updated art documentation”
5. **Create a Pull Request**
    - Tag someone for review if possible

---

# Sound Integration

1. **Switch to the sound branch**

    Their audio work should be located there.
2. **Open Unity and import audio files**
    - Check compression settings and file format
3. **Continue their work**
    - Assign audio to objects, triggers, animations, or events
    - Test if sounds play correctly during gameplay
    - **Note:** If the Sound Integrator lacks technical knowledge, a programmer can do this under supervision.
4. **Write clear commit messages**
    - Example: “Added door open sound to door prefab”
5. **Create a Pull Request**

- Tag someone for review if possible

---

## Gameplay Integration

1. **Switch to the gameplay branch**

   Their gameplay work should be in a separate branch.

2. **Open the gameplay document**
   - Contains all elements to integrate (art + programming)
   - Updating and maintaining this document is part of the integrator’s job
3. **Continue their work**
   - Merge elements created by programmers (scripts, mechanics, events)

     with elements created by artists (animations, UI, assets) inside the document
   - Ensure the document clearly describes how everything fits together
   - **Note:** Full scene integration is a **team effort** involving programming, art, and gameplay integrators.
4. **Write clear commit messages**
   - Example: “Updated gameplay document with new enemy attack sequence”
5. **Create a Pull Request**
   - Tag someone for review if possible