🧠 Practical GitHub Tasks for Software Development Training

🔹 Stage 1: Git Basics (Individual Level)

1. **Initialize a Git Repository**
   - Task: Create a local Git repo and push it to GitHub.
2. **Create a README File**
   - Task: Write a clear project description using Markdown (README.md).
3. **Commit Messages Practice**
   - Task: Make multiple commits with meaningful messages as they add files.
4. **Create Branches**
   - Task: Create a `feature/yourname` branch and switch between `main` and feature branches.
5. **Git Ignore Usage**
   - Task: Create a `.gitignore` file to exclude unnecessary files (e.g., `.env`, `node_modules`, `.DS_Store`).

🔹 Stage 2: Collaboration & Workflow

6. **Fork & Clone a Repository**
   - Task: Fork a sample repo (provided by trainer), clone it locally, and make changes.
7. **Pull Request Workflow**
   - Task: Make changes in a branch and open a **Pull Request** to merge it into the `main` branch.
8. **Code Review Simulation**

- Task: Review a peer’s pull request using GitHub's review tools (comments, suggestions, approvals).

9. **Merge Conflicts**
    - Task: Simulate a merge conflict and resolve it locally before pushing.
10. **Use Git Tags**
    - Task: Tag a versioned release (e.g., `v1.0.0`) after completing a milestone.

🔹 Stage 3: Real Development Tasks

11. **Bug Tracker with Issues**
    - Task: Create and assign GitHub Issues to track bugs or new features.
12. **Project Board**
    - Task: Use GitHub Projects (Kanban board) to manage to-dos and progress.
13. **Wiki Documentation**
    - Task: Create Wiki pages for project documentation.
14. **Markdown Formatting**
    - Task: Write a detailed `CONTRIBUTING.md` or `CHANGELOG.md` with proper formatting.
15. **Use Actions for CI/CD**
    - Task: Set up a simple GitHub Actions workflow (e.g., linting, tests on push).

🔹 Stage 4: Advanced Use (Optional)

16. **Collaborative Mini Project**
    - Task: Work in small teams to build a mini project (e.g., To-Do app, portfolio site) using GitHub for code collaboration.

17. **Release Management**
    - Task: Create a new release in GitHub with attached assets (like build files or documentation).
18. **Environment Variables**
    - Task: Use secrets in GitHub Actions to handle sensitive keys.
19. **Custom Branch Policies**
    - Task: Explore setting up protected branches, required reviews, or auto-deploy actions.
20. **Template Repository**
    - Task: Convert a repo into a template and use it to start new projects.

📌 Final Note (Add to All Tasks)

> All GitHub activities are part of your **training program** to prepare you for real software development.
> These tasks are **not for company projects or commercial use**. Practice responsibly.

Here's how freshers can approach these GitHub tasks both **individually** and in **groups**, along with the **option of using multiple accounts** for collaboration practice:

🧩 How to Approach GitHub Training Tasks

🔹 Individual Tasks (No Group Needed)

The following tasks can and should be done **solo**, using just one GitHub account:

1. ✅ Initialize a repo and push code
2. ✅ Create branches, commit, and merge
3. ✅ Write a README, CONTRIBUTING.md, etc.
4. ✅ Use `.gitignore`

5. ✅ Tag versions (`git tag`)
6. ✅ Create Issues (self-assigned)
7. ✅ Use GitHub Actions for automated workflows
8. ✅ Create project boards and wikis
9. ✅ Practice merge conflicts (by editing different branches)

✅ **They do not need multiple accounts or teammates for these.**
👨‍💻 Use local branches to simulate team workflows.

🔹 Group-Based Tasks (Optional but Valuable)

The following tasks are ideal for **2–3 member groups** or using **multiple GitHub accounts** (can be fake/test accounts if needed for training):

1. 🤝 **Fork & Pull Requests**
   - One person creates a repo; others fork it and open PRs.
2. 🧑‍🔧 **Code Review**
   - Teammates review each other's pull requests and comment/suggest changes.
3. ⚠️ **Merge Conflict Resolution**
   - Two users edit the same line on different branches and try to merge = conflict.
4. 🛠️ **Project Collaboration**
   - Work on a small app or page, assign issues/tasks to each other.
5. 📦 **Release & Tag Collaboration**
   - One person tags/releases; others download and verify.

🔄 If working solo, they can simulate this with **2-3 GitHub accounts and emails** on the same computer or browser in incognito windows.

💡 Using Multiple Accounts (Simulated Teamwork)

To simulate teamwork **without real groups**, each learner can:

- Create 2 or 3 GitHub accounts (e.g., `dev-uttam-1`, `dev-uttam-2`)
- Use them to:
  - Fork, push, and pull PRs
  - Simulate review workflows
  - Test permissions and roles

Git setup tip:
To switch accounts locally:

bash
CopyEdit
```
git config user.name "Test User"
git config user.email "test@example.com"
```

They can also use GitHub’s browser interface to fork, edit, and pull request without local setup if needed.

📌 Recommendation

- ✅ Start **individually** for the basics.
- 👥 Form **2–3 person groups** for collaboration tasks.
- 🧪 Or simulate teamwork with multiple accounts per person.