Here's a step-by-step tutorial on how to create a GitHub account, create a repository on GitHub, install Git on your local computer, and link your GitHub account to your local computer using SSH.

1. Creating a GitHub Account:

1. Go to GitHub's homepage.
2. Click on the "Sign up" button.
3. Fill in your desired username, email address, and password.
4. Click on the "Create an account" button.
5. Follow the on-screen instructions to complete the registration process.

2. Installing Git on Your Local Computer:

For Windows:

1. Go to the Git for Windows download page.
2. Download the installer.
3. Run the installer and follow the on-screen instructions.

For macOS:

1. If you have Homebrew installed, you can run: `brew install git`
2. If not, you can download the Git installer for macOS from the Git website.

For Linux:

1. Use your distribution's package manager. For example:
    - For Debian/Ubuntu: `sudo apt-get install git`
    - For Fedora: `sudo dnf install git`
    - For CentOS: `sudo yum install git`

3. Linking GitHub Account to Local Computer with SSH:

1. **Generate an SSH Key**:
    - Open a terminal or command prompt.
    - Type `ssh-keygen -t rsa -b 4096 -C "your_email@example.com"`, replacing `your_email@example.com` with your GitHub email address.
    - Press Enter to accept the default file location.
    - Enter a passphrase (optional) and confirm it.
2. **Add the SSH Key to the ssh-agent**:
    - Start the ssh-agent in the background: `eval "$(ssh-agent -s)"`
    - Add your SSH private key to the ssh-agent: `ssh-add ~/.ssh/id_rsa`
3. **Add the SSH Key to your GitHub account**:
    - Display the public key with: `cat ~/.ssh/id_rsa.pub`
    - Copy the displayed key.

- Go to GitHub and navigate to your account settings.
- Click on "SSH and GPG keys" in the sidebar.
- Click on "New SSH key".
- Paste your key into the "Key" field and give it a descriptive title.
- Click "Add SSH key".

4. **Test the connection**:

   - In your terminal or command prompt, type: `ssh -T git@github.com`
   - If everything is set up correctly, you should see a message like: "Hi [YourUsername]! You've successfully authenticated, but GitHub does not provide shell access."

Now, you're all set! You can clone, push, and pull from your GitHub repositories using SSH without having to enter your username and password every time.

4. Creating a Git Repository on GitHub:

1. Once logged in, click on the '+' icon in the top right corner.
2. Select "New repository".
3. Name your repository and provide a brief description.
4. Choose whether to make your repository public (visible to everyone) or private (only visible to you and collaborators).
5. Click on the "Create repository" button.

5. Making an Existing Directory on Your Local Computer a Git Repository:

1. **Navigate to Your Directory**:

   - Open a terminal or command prompt.
   - Navigate to your directory using the `cd` command. For example: `cd path/to/your/directory`

2. **Initialize the Directory as a Git Repository**:

   - Run the command: `git init`
   - This initializes a new Git repository and begins tracking an existing directory.

3. **Link Your Local Repository to the Remote GitHub Repository**:

   - Run the command: `git remote add origin git@github.com:YourUsername/YourRepositoryName.git`
     - Replace `YourUsername` with your GitHub username and `YourRepositoryName` with the name of the repository you created on GitHub.

4. **Commit Your Existing Files**:

   - Add all the files in the directory to the staging area: `git add .`
   - Commit the files: `git commit -m "Initial commit"`

5. **Push to GitHub**:

   - Push your commits to the remote repository: `git push -u origin master`

6. Installing Visual Studio Code (VSCode):

Using VSCode to sync your git repository is not necessary, but highly recomended (versus doing it from the command line).

For Windows, macOS, and Linux:

1. Go to the VSCode download page.
2. Choose the version suitable for your OS and follow the installation instructions.

7. Using VSCode to Sync Local Git Repo to Remote One on GitHub:

1. **Open Your Repository in VSCode**:
   - Launch VSCode.
   - Go to `File` > `Open Folder` and select your repository's directory.
2. **Install the GitHub Extension (Optional but Recommended)**:
   - Go to the Extensions view by clicking on the square icon on the sidebar or pressing `Ctrl+Shift+X`.
   - Search for "GitHub Pull Requests and Issues" and install it. This extension allows you to manage pull requests and issues directly from VSCode.
3. **Making Changes and Committing**:
   - After making changes to your files, you'll notice the Source Control icon (looks like a branch) on the sidebar has a badge indicating the number of changed files.
   - Click on the Source Control icon.
   - You'll see a list of changes. Hover over the changes and click the '+' icon to stage them.
   - Once staged, type a commit message in the text box at the top and press `Ctrl+Enter` (or `Cmd+Enter` on macOS) to commit.
4. **Pushing and Pulling from GitHub**:
   - In the Source Control view, you'll see a "..." icon at the top. Click on it.
   - From the dropdown, you can choose `Push` to push your commits to GitHub or `Pull` to pull any changes from GitHub.

With these steps, you can easily sync your local Git repository with the remote one on GitHub using Visual Studio Code.

Connecting to a high-performance cluster (like your FARM) and setting up GitHub access is a common task for researchers and developers who work with large-scale computations. Here's how you can set it up:

8. Connecting to the FARM and Setting Up GitHub Access:

1. **SSH into the FARM**:
   - Open a terminal or command prompt on your local machine.

- Use the `ssh` command to connect to the FARM. The general format is: `ssh YourUsername@farm.cse.ucdavis.edu`, where `YourUsername` is your username on the FARM.

2. **Generate an SSH Key on the FARM for GitHub**:

   - Once connected to the FARM, you can generate an SSH key specifically for GitHub access by following the same steps as before: `ssh-keygen -t rsa -b 4096 -C "your_email@example.com"`.
   - Remember to replace `your_email@example.com` with your GitHub email address.

3. **Add the SSH Key to the ssh-agent on the FARM**:

   - Start the ssh-agent in the background: `eval "$(ssh-agent -s)"`
   - Add your SSH private key to the ssh-agent: `ssh-add ~/.ssh/id_rsa`

4. **Add the SSH Key to your GitHub account**:

   - Display the public key with: `cat ~/.ssh/id_rsa.pub`
   - Copy the displayed key.
   - On your local machine, go to GitHub and navigate to your account settings.
   - Click on "SSH and GPG keys" in the sidebar.
   - Click on "New SSH key".
   - Paste your key into the "Key" field and give it a descriptive title (e.g., "FARM SSH Key").
   - Click "Add SSH key".

5. **Test the Connection from the FARM**:

   - On the FARM, type: `ssh -T git@github.com`
   - If everything is set up correctly, you should see a message like: "Hi [YourUsername]! You've successfully authenticated, but GitHub does not provide shell access."

9. Creating the `clone_github.sh` Script:

1. **Create the Script File**:

   - While still on the FARM, use a text editor (like `nano` or `vim`) to create the script. For example: `nano clone_github.sh`

2. **Enter the Script Content**:

   - Copy and paste the following into the editor (replacing USERNAME with your github account name and REPO with the name of the repository on github).

     ```
     #!/bin/bash

     rm -rf github
     git clone git@github.com:USERNAME/REPO.git
     mv tajima github
     chmod u+x github/code/*
     ```

3. **Save and Exit**:
   - If using `nano`, press `Ctrl+O` to write the changes, then `Enter` to confirm, and finally `Ctrl+X` to exit.
   - If using `vim`, press `Esc`, then type `:wq`, and press `Enter`.
4. **Make the Script Executable**:
   - Run the command: `chmod +x clone_github.sh`

Now, whenever you want to quickly update your code on the FARM, you can run the `./clone_github.sh` script. This will remove the old `github` directory, clone the repository from GitHub, rename the cloned directory to `github`, and make all the code files inside `github/code/` executable.