

# Create Account in GitHub

The image shows the GitHub landing page with a dark background. At the top, there's a navigation bar with a logo, 'Features', 'Explore', 'Pricing', a search bar labeled 'Search GitHub', and 'Sign in or Sign up'. Below the navigation, a large white text 'Built for developers' is displayed. To the right, there's a form for creating a new account. The form consists of three input fields: 'Pick a username', 'Your email address', and 'Create a password'. Below these fields is a note: 'Use at least one letter, one numeral, and seven characters.' A large green button at the bottom right is labeled 'Sign up for GitHub'. A red box highlights this button. At the very bottom, small text states: 'By clicking "Sign up for GitHub", you agree to our [terms of service](#) and [privacy policy](#). We'll occasionally send you account related emails.'

Features Explore Pricing

Search GitHub

Sign in or Sign up

# Built for developers

GitHub is a development platform inspired by the way you work. Host code, manage projects, and build software alongside millions of other developers.

Pick a username

Your email address

Create a password

Use at least one letter, one numeral, and seven characters.

Sign up for GitHub

By clicking "Sign up for GitHub", you agree to our [terms of service](#) and [privacy policy](#). We'll occasionally send you account related emails.

# Create Repository

## Create a new repository

A repository contains all the files for your project, including the revision history.

Owner

 sathyadevops

/ myproj 

Great repository names are short and memorable. Need inspiration? How about [automatic-octo-guacamole](#).

Description (optional)

 Public

Anyone can see this repository. You choose who can commit.

 Private

You choose who can see and commit to this repository.

Initialize this repository with a README

This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

Add .gitignore: **None**

Add a license: **None**  

**Create repository**

# Install Git on Ubuntu 14.04

## Step 1: Installation

```
#apt-get update  
#apt-get install git-core -y  
#git --version
```

## Step 2: Configuration

```
#git config --global user.name sathyadevops  
#git config --global user.email  
sathyadevops1@gmail.com  
#cat .gitconfig (or) #git config --list
```

## **Step 3: Create GIT repository**

```
#mkdir /repos  
#cd /repos  
#git init  
#ls -a  
#git clone  
https://github.com/sathyadevops/myproj.git
```

## **Step 4: Working with Git Repository**

```
#echo "Welcome to Git" >> README.md  
#git status
```

**to add a file to cache (staging Area)**

```
#git add README.md  
#git status
```

**to move a file from Staging Area to Local Repo**

```
#git commit -m "initial commit"
```

**to Add and Commit a file at a time**

```
#git commit -a -m "initial commit"
```

**to push the code to Central Repo(master)**

```
#git push -u origin master
```

**To changed files in your working repository**

```
#git status
```

**To show all git commits**

```
#git log
```

```
#git log -p
```

```
#git log --since=12-03-2017 --until=13-03-2017
```

```
#git log --oneline
```

**To made changes to tracked files**

#git diff

#git log

#git diff 57af6s43d..9wg5c2ys3

**To list all branches**

#git branch

**to work with branches:**

```
#git branch branch1
```

```
#git checkout branch1
```

```
#git branch
```

```
#vi index
```

new line from branch

```
#git commit -a -m "new line from branch"
```

```
#git push -u origin branch1
```

**check in browser → github**

**to merge the branch code into master**

```
#git checkout master
```

```
#git merge branch1
```

```
#cat index.html
```

```
#git push -u origin master
```

**to delete a Branch:**

```
#git branch -d branch1
```

**to delete a Branch without merging the Data:**

```
#git branch -D branch1
```

```
#git branch
```

## Git - Review Changes

```
# git diff
```

```
# git log
```

```
# git show
```

```
c0f455906befd100192848233fb896d081e22
```

```
84
```

## Git – Remote Server

```
#git remote -v
```

```
#git checkout -- . (to revert all the changes)
```

# Git Stash

- git stash temporarily shelves (or *stashes*) changes you've made to your working copy so you can work on something else, and then come back and re-apply them later on.
- Stashing is a way to pause what you're currently working on and come back to it later.

```
#vi index.html  
<h1> Hello World </h1>  
<h2> New line is added </h2>  
# git diff
```

**Stash your changes away with:**

```
# git stash (or)  
# git stash save "message"  
# git diff  
# cat index.html  
<h1> Hello World </h1>
```

**To List multiple layers of stashes**

```
# git stash list  
# git stash show
```

**You're back to your original working state**

```
# git stash apply  
# git stash apply stash@{0}  
# git stash pop  
# cat index.html  
<h1>Hello World </h1>  
<h2>New line is added </h2>
```

**We can manually delete stashes :**

```
# git stash drop stash@{1}
```

**delete all of the stored stashes**

```
# git stash clear
```