**Shravan More**

**Batch: T13**

**Roll No. 62**

# SEPM
# EXPERIMENT NO: 02

**Aim:** To Understand Version Control System, install Git and Create a Github Account.

## Theory:

### Version Control System (VCS)

A Version Control System (VCS) is a tool that helps developers track changes to files over time. It allows multiple developers to work on the same project simultaneously, while keeping track of changes, preventing conflicts, and ensuring that previous versions of files are always available for reference or rollback.

In simple terms, version control allows you to manage changes in your code, documentation, or any set of files. It records what changes were made, when they were made, and who made them.

Types of Version Control Systems:

1. Local Version Control System: This is a simple version control system that keeps track of changes on a local machine. One example is the RCS (Revision Control System).
2. Centralized Version Control System (CVCS): A centralized VCS has a central server that stores the main repository and all the changes. Developers can pull changes from this central repository and push their changes back. Examples: Subversion (SVN), CVS.
3. Distributed Version Control System (DVCS): In this system, each developer has a local repository on their own machine, which is synchronized with the central repository. Examples include Git, Mercurial, and Bazaar.

### Git: A Distributed Version Control System

Git is one of the most popular and widely used Distributed Version Control Systems. It allows multiple developers to work on a project, collaborate, and track changes efficiently. Git helps you:

- Track the history of all changes made to a project.
- Collaborate with other developers without interfering with each other's work.

- Revert back to previous versions of your project if needed.
- Manage branches and merge different versions of a project.

**Key Concepts in Git:**

1. Repository: A Git repository is a collection of files and directories that are being tracked by Git. It contains both the project's files and the entire history of changes.
2. Commit: A commit is a snapshot of your project at a specific point in time. Each commit is associated with a unique ID (SHA-1 hash).
3. Branch: A branch is a separate line of development in Git. By default, Git creates a "main" or "master" branch. Developers can create new branches to work on features or bug fixes without affecting the main codebase.
4. Merge: Merging combines changes from different branches into one.
5. Clone: A clone is a copy of a remote repository that you can work with locally.

## GitHub: A Hosting Service for Git Repositories

GitHub is a cloud-based hosting platform for Git repositories. It allows developers to store and manage their Git repositories online, collaborate with others, and even contribute to open-source projects.

With GitHub, you can:

- Share your code with others.
- Work on projects with collaborators.
- Use GitHub's features like pull requests and issues to manage changes and project tasks.

**Key Features of GitHub:**

- Remote Repositories: GitHub hosts your repositories online, making it easy to share and collaborate with others.
- Forking: You can create your own copy of a repository to make changes without affecting the original project.
- Pull Requests: After making changes in your forked repository, you can submit a pull request to propose your changes to the original repository.
- Issues: You can track bugs, feature requests, and tasks related to the project.

## Setting Up Git and GitHub

1. Install Git:
   Git can be installed on Windows, Mac, and Linux.

To install Git, visit the official Git website: https://git-scm.com/downloads and follow the instructions for your operating system.

2. Create a GitHub Account:

Go to https://github.com/ and sign up for an account.

After creating an account, you can create new repositories, fork existing ones, and start collaborating on projects.

3. Setting Up Git Locally:

After installing Git, configure it by setting your user name and email address with these commands:

```
git config --global user.name "Your Name"

git config --global user.email "your.email@example.com"
```

4. Clone a Repository:

To clone an existing repository from GitHub, use the following command:

```
git clone https://github.com/username/repository-name.git
```

5. Make Changes & Commit:

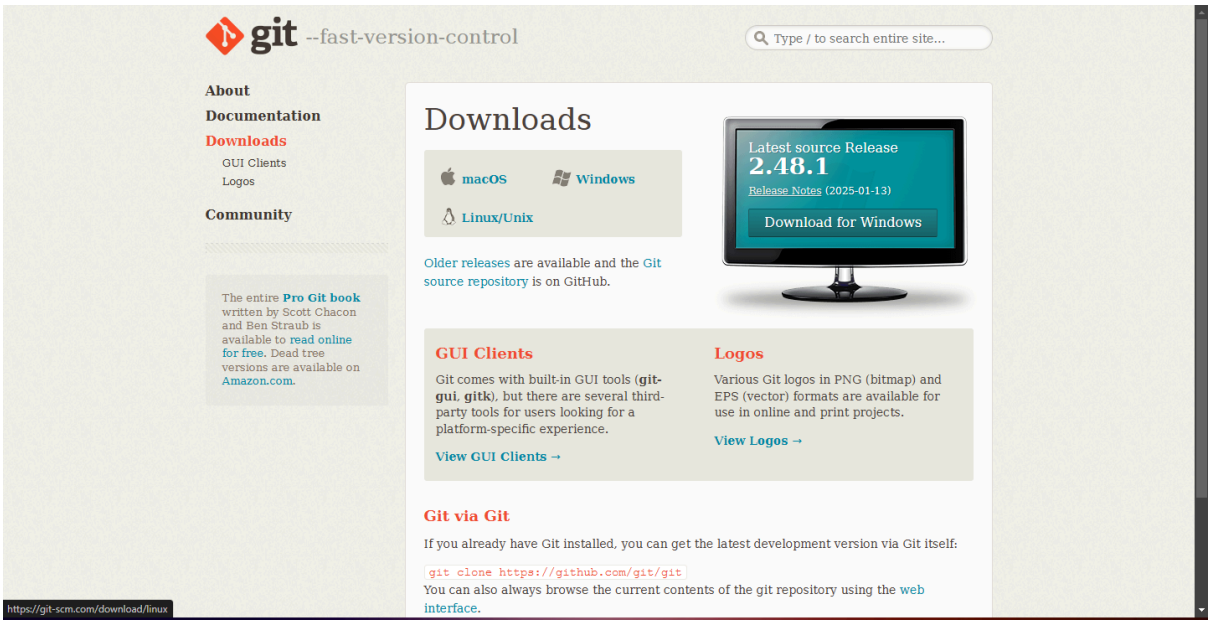
After modifying files in your local repository, you can commit those changes with the following commands:

```
git add .

git commit -m "Your commit message"
```

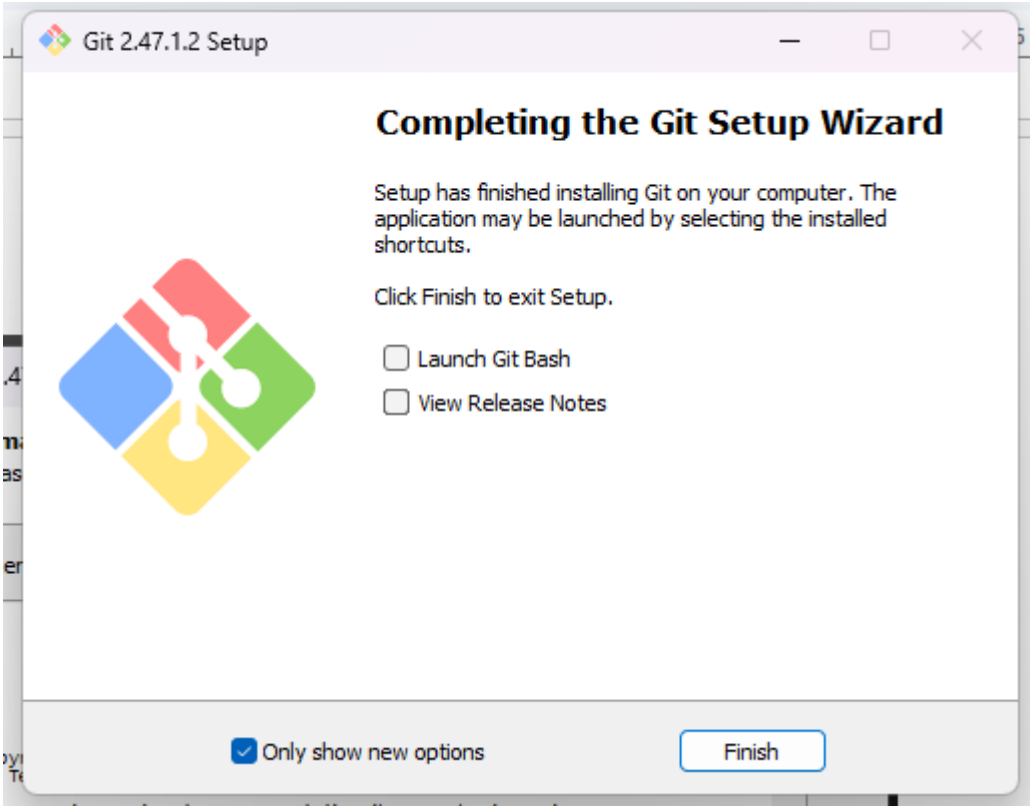
6. Push Changes to GitHub:

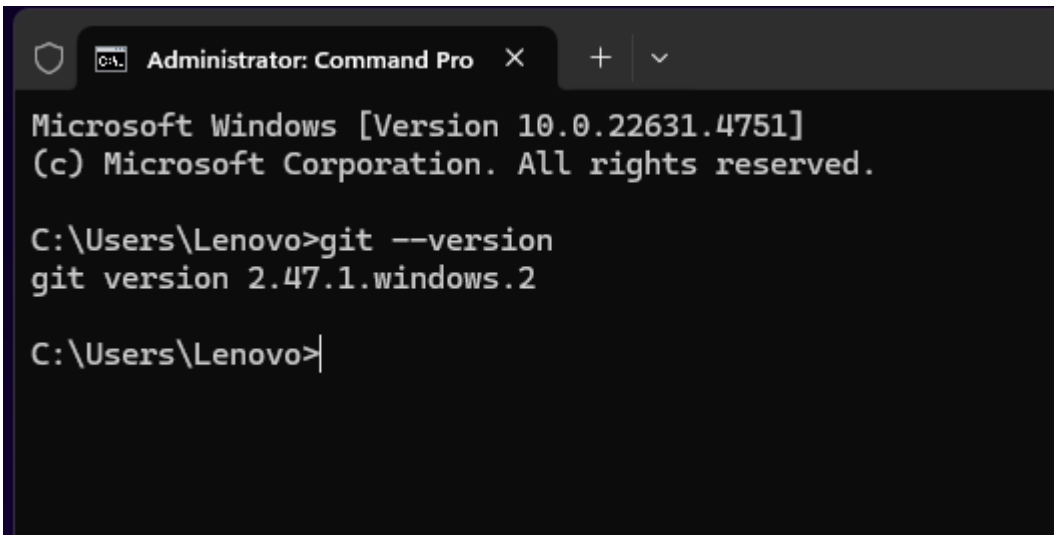
   After committing locally, you can push changes to the remote GitHub repository:
   ```
   git push origin main
   ```

# Screenshots :




Git 2.47.1.2 Setup
Information
Please read the following important information before continuing.
When you are ready to continue with Setup, click Next.
GNU General Public License
Version 2, June 1991
Copyright (C) 1989, 1991 Free Software Foundation, Inc.
59 Temple Place - Suite 330, Boston, MA 02111-1307, USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.
Preamble
The licenses for most software are designed to take away your
freedom to share and change it. By contrast, the GNU General Public
License is intended to guarantee your freedom to share and change
https://gitforwindows.org/
Only show new options
Install
Cancel


Git 2.47.1.2 Setup
Completing the Git Setup Wizard
Setup has finished installing Git on your computer. The application may be launched by selecting the installed shortcuts.
Click Finish to exit Setup.
Launch Git Bash
View Release Notes
Only show new options
Finish




```
Lenovo@202-018 MINGW64 ~
$ cd Desktop

Lenovo@202-018 MINGW64 ~/Desktop
$ mkdir git-docs

Lenovo@202-018 MINGW64 ~/Desktop
$ cd git-docs

Lenovo@202-018 MINGW64 ~/Desktop/git-docs
$ git config --global --list
user.name=Shubham
user.email=shubham78p@gmail.com

Lenovo@202-018 MINGW64 ~/Desktop/git-docs
$ mkdir git-demo-project

Lenovo@202-018 MINGW64 ~/Desktop/git-docs
$ cd git-demo-project/

Lenovo@202-018 MINGW64 ~/Desktop/git-docs/git-demo-project
$ git init
Initialized empty Git repository in C:/Users/Lenovo/Desktop/git-docs/git-demo-project/.git/
```

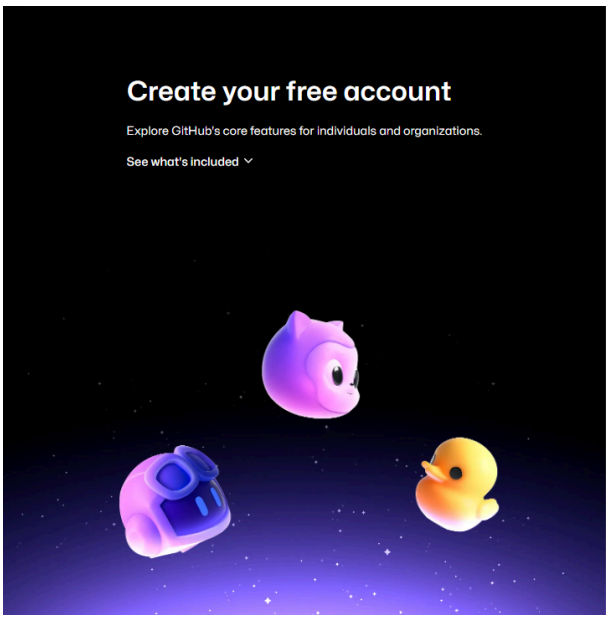
```
Lenovo@202-018 MINGW64 ~/Desktop/git-docs/git-demo-project (master)
$ git add .

Lenovo@202-018 MINGW64 ~/Desktop/git-docs/git-demo-project (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   SampleDOCS.txt


Lenovo@202-018 MINGW64 ~/Desktop/git-docs/git-demo-project (master)
$ git commit -m "First Commit"
[master (root-commit) 5a51523] First Commit
 1 file changed, 1 insertion(+)
 create mode 100644 SampleDOCS.txt
```

```
Lenovo@202-018 MINGW64 ~/Desktop/git-docs/git-demo-project (master)
$ git add .

Lenovo@202-018 MINGW64 ~/Desktop/git-docs/git-demo-project (master)
$ git commit -m "html_Page"
[master 693252a] html_Page
 1 file changed, 11 insertions(+)
 create mode 100644 index.html

Lenovo@202-018 MINGW64 ~/Desktop/git-docs/git-demo-project (master)
$ git status
On branch master
nothing to commit, working tree clean
```

```
Lenovo@202-018 MINGW64 ~/Desktop/git-docs/git-demo-project (master)
$ git status
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)
        statusTest.txt

nothing added to commit but untracked files present (use "git add" to track)

Lenovo@202-018 MINGW64 ~/Desktop/git-docs/git-demo-project (master)
$ git add statusTest.txt

Lenovo@202-018 MINGW64 ~/Desktop/git-docs/git-demo-project (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   statusTest.txt


Lenovo@202-018 MINGW64 ~/Desktop/git-docs/git-demo-project (master)
$ git commit -m "StatusTest"
[master d6746a0] StatusTest
 1 file changed, 1 insertion(+)
 create mode 100644 statusTest.txt
```

```
Lenovo@202-018 MINGW64 ~/Desktop/git-docs/git-demo-project (master)
$ git commit -am "Express Commit"
[master b51a73e] Express Commit
 1 file changed, 1 insertion(+), 1 deletion(-)

Lenovo@202-018 MINGW64 ~/Desktop/git-docs/git-demo-project (master)
$ git status
On branch master
nothing to commit, working tree clean
```

```
Lenovo@202-018 MINGW64 ~/Desktop/git-docs/git-demo-project (master)
$ git log
commit b51a73ed4150c7020500ee61c52620c2245815ac (HEAD -> master)
Author: Shubham <shubham78p@gmail.com>
Date:   Mon Feb 3 13:29:15 2025 +0530

    Express Commit

commit d6746a046827e94f0f400d79942cac8d97b0e53d
Author: Shubham <shubham78p@gmail.com>
Date:   Mon Feb 3 13:27:48 2025 +0530

    StatusTest

commit 693252ad41cf859fcaecaf6707fe23a29a711794
Author: Shubham <shubham78p@gmail.com>
Date:   Mon Feb 3 13:10:22 2025 +0530

    html_Page

commit 5a515233c7a5d60c33a77609d60b63b31f63d177
Author: Shubham <shubham78p@gmail.com>
Date:   Mon Feb 3 13:07:09 2025 +0530

    First Commit

Lenovo@202-018 MINGW64 ~/Desktop/git-docs/git-demo-project (master)
$ git log --oneline
b51a73e (HEAD -> master) Express Commit
d6746a0 StatusTest
693252a html_Page
5a51523 First Commit
```




Already have an account? Sign in →

## Sign up to GitHub

Email*

javachip005@gmail.com

Password*

••••••••••••

Password should be at least 15 characters OR at least 8 characters including a number and a lowercase letter.

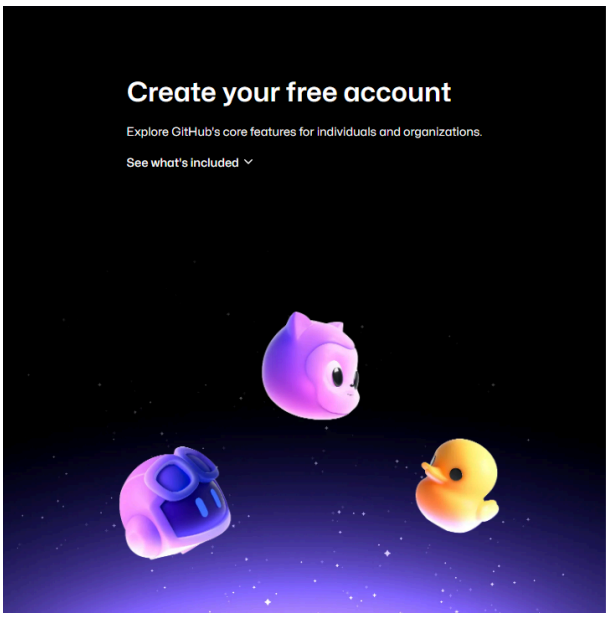
Username*

Java-Chip-Code

Username may only contain alphanumeric characters or single hyphens, and cannot begin or end with a hyphen.

Continue →

By creating an account, you agree to the Terms of Service. For more information about GitHub's privacy practices, see the GitHub Privacy Statement. We'll occasionally send you account-related emails.




Already have an account? Sign in →

## Confirm your email address

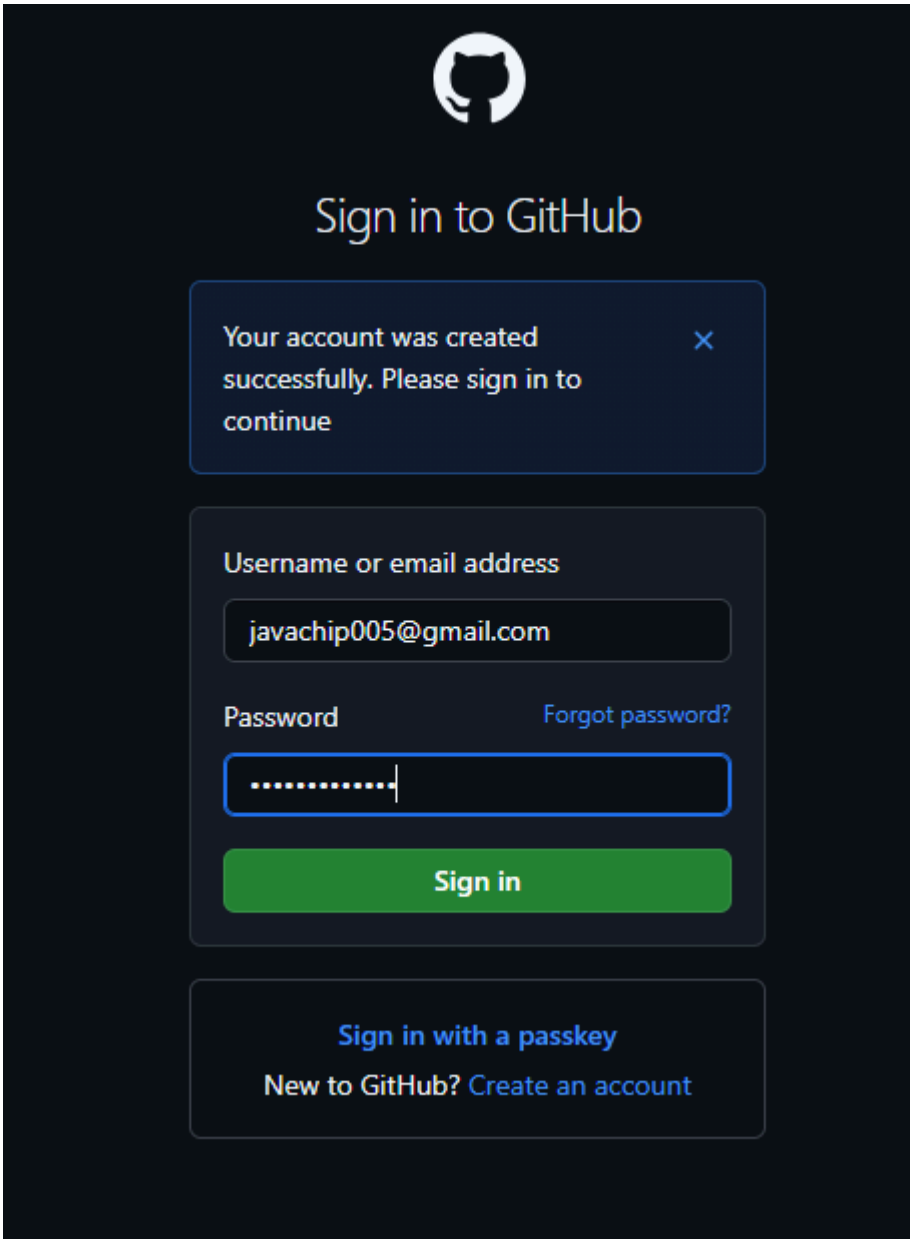
We have sent a code to **javachip005@gmail.com**

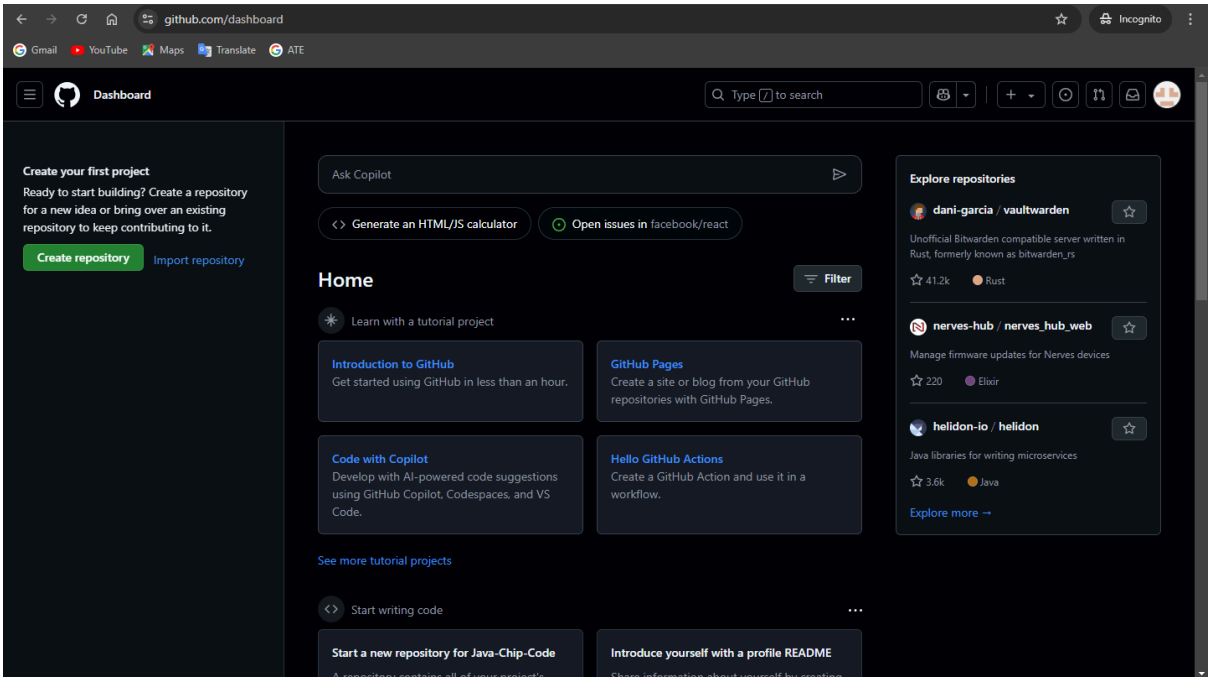
Enter code

Continue →

Didn't get your email? Resend the code or update your email address.

By creating an account, you agree to the Terms of Service. For more information about GitHub's privacy practices, see the GitHub Privacy Statement. We'll occasionally send you account-related emails.


Sign in to GitHub
Your account was created successfully. Please sign in to continue
Username or email address
javachip005@gmail.com
Password
Forgot password?
Sign in
Sign in with a passkey
New to GitHub? Create an account




**Conclusion :** Successfully Understood Version Control System, installed Git and Created a Github Account.