

Lab 14: Introduction to GitHub Cloning Repository

Lab 13: Introduction to GitHub and Cloning Repository

Objective

1. Understand the fundamentals of Git as a version control system.
2. Learn how to connect local projects to GitHub.
3. Practice committing changes, pushing code, pulling remote changes, and resolving conflicts.
4. Understand repository structure, staging, committing, branching, and remote synchronization.
5. Explore collaborative workflows used in software development teams.

Reference Material

- Lecture Notes: Introduction to Version Control
- Textbook: *Practical Git and GitHub*
- Classic Reference: *Pro Git* (Scott Chacon)

Latest References (2024–2025):

- GitHub Docs: *Getting Started with GitHub*
- Atlassian Git Tutorial
- Git Documentation – Command Reference

Activity Timeline

Activity Name	Time
Introduction & Overview (Git, GitHub, Workflow)	30 minutes
Walkthrough Task 1: Cloning + Project Setup	40 minutes
Walkthrough Task 2: Commit, Push & Pull Changes	50 minutes
Practice Tasks (Git Workflow Scenarios)	40 minutes
Conclusion & Discussion (Collaboration & CI/CD)	20 minutes
Total	180 minutes

Introduction:

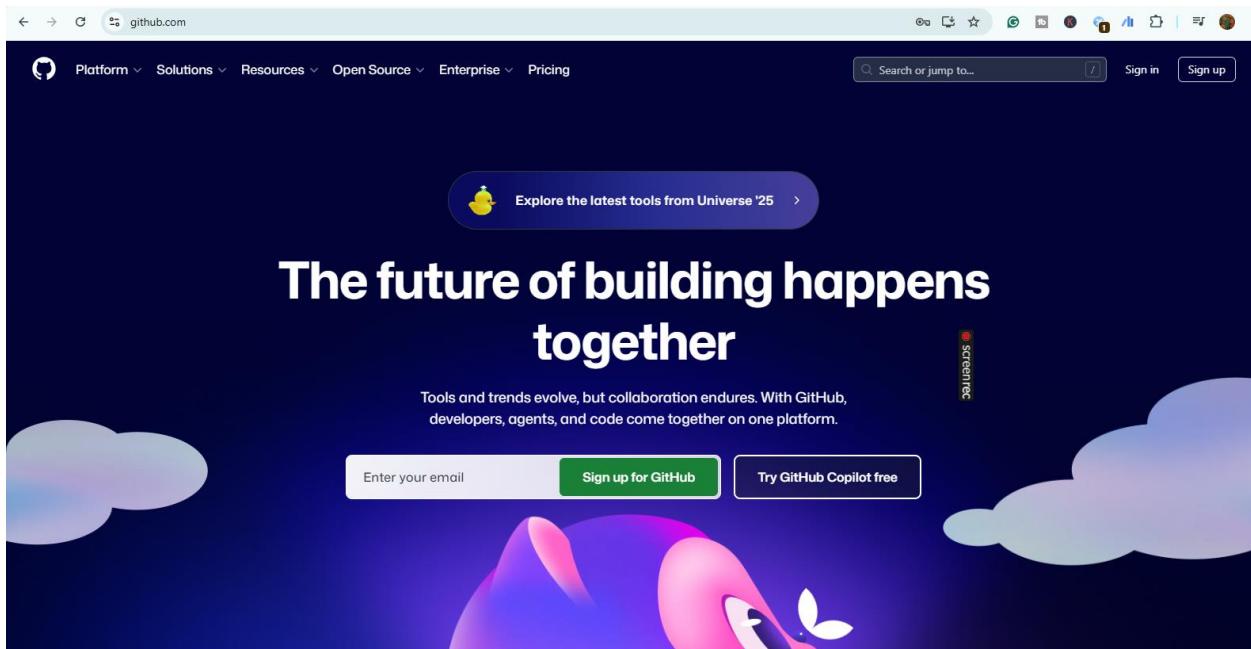
Version control is an essential component of modern software development. Git allows developers to track changes, maintain revisions, and collaborate efficiently across distributed environments.

GitHub is the most widely used hosting platform for Git repositories. It provides online repository management, collaboration features, pull requests, issue tracking, reviews, and integration pipelines.

In this lab, students will learn how to work with Git and GitHub, manage code versions, commit changes, and synchronize local projects with remote repositories.

Walkthrough Task 1:

Create account on Github



The dashboard of github will look like below

The screenshot shows the GitHub Home page. On the left, there's a sidebar with options to 'Create your first project' and buttons for 'Create repository' and 'Import repository'. The main area has a search bar and navigation buttons for 'Task', 'Create issue', 'Write code', 'Git', and 'Pull requests'. Below these are sections for 'Ask anything' and '+ Add repositories, files, and spaces'. A 'Feed' section displays trending repositories, including 'parcade/Continuous-Claude-v3' and 'snarktank/ralph'. To the right, a 'Latest from our changelog' sidebar lists recent updates: 'Gemini 3 Flash is now available in Visual Studio, JetBrains IDEs...', 'Reduced pricing for GitHub-hosted runners usage', 'Improved performance for GitHub Actions workflows page', and 'Control who can request apps for your organization now...'. At the bottom, there's a footer with copyright information and links to GitHub's Terms, Privacy, Security, Status, Community, Docs, Contact, Manage cookies, and Do not share my personal information.

Create a repository

The screenshot shows the 'Create a repository' form. It consists of two main sections: 'General' and 'Configuration'.

General: This section includes fields for 'Owner' (set to 'hafuumahmood-hub'), 'Repository name' (set to 'SCD Lab 14'), and a note that the repository will be created as 'SCD-Lab-14'. There's also a description field containing 'This will cover the basics of how to create the repository and commit changes in it'.

Configuration: This section includes fields for 'Choose visibility' (set to 'Public'), 'Add README' (with a note about READMEs), 'Add .gitignore' (with a note about ignoring files), and 'Add license' (with a note about licenses). All configuration fields have their default values selected ('Off', 'No .gitignore', 'No license').

A large green 'Create repository' button is located at the bottom right of the configuration section.

After that, invite the collaborators (teammates that can make changes to your project)

The screenshot shows the 'General' settings page for a GitHub repository. The left sidebar has sections like 'Access', 'Code and automation', 'Security', and 'Integrations'. The 'Access' section is expanded, showing 'Collaborators' (selected), 'Moderation options', and 'Direct access' (0 collaborators). The 'Manage access' section says 'You haven't invited any collaborators yet' with a 'Add people' button.

Add your code files inside the project.

The screenshot shows a GitHub repository page for 'SCD-Lab-14'. The top navigation bar includes links for Code, Issues, Pull requests, Actions, Projects, Wiki, Security, Insights, and Settings. The repository details show it's public and created by 'hafuumahmood-hub'. It has 1 branch, 0 tags, 1 commit, and 0 forks. The 'About' section contains a brief description: 'This will cover the basics of how to create the repository and commit changes in it'. Below this are sections for Activity, Stars, Watching, Forks, Releases, Packages, and Languages. The Languages section shows Java at 100.0%.

Click on commits to see the commit history

The screenshot shows a GitHub repository page for 'hafuumahmood-hub / SCD-Lab-14'. The 'Commits' tab is selected, showing a single commit for 'Create BankAccount.java' made by 'hafuumahmood-hub' on Jan 11, 2026. The commit message is 'Create BankAccount.java'. The commit is verified and has a SHA-1 hash of 575f446. The interface includes a search bar, user navigation, and various repository management links.

Now do some changes in the code

The screenshot shows a GitHub code editor for 'BankAccount.java'. A modal dialog titled 'Commit changes' is open, prompting for a commit message. The message 'Update BankAccount.java' is entered. Below the message input is an 'Extended description' field with placeholder text 'Add an optional extended description...'. At the bottom of the dialog, there are two radio button options: 'Commit directly to the main branch' (selected) and 'Create a new branch for this commit and start a pull request'. There are also 'Cancel' and 'Commit changes' buttons. The background shows the Java code for the 'BankAccount' class.

```
class BankAccount {
    private String ownerName;
    private double balance;
    private boolean active;
    private int newvariable;

    public BankAccount(String ownerName, double initialBalance) {
        if (ownerName == null)
            throw new IllegalArgumentException("Owner name cannot be null");
        if (initialBalance < 0)
            throw new IllegalArgumentException("Initial balance cannot be negative");
        this.ownerName = ownerName;
        this.balance = initialBalance;
        this.active = true;
    }

    public void deposit(double amount) {
        if (!active)
            throw new IllegalStateException("Account is closed");
        balance += amount;
    }

    public void withdraw(double amount) {
        if (!active)
            throw new IllegalStateException("Account is closed");
        if (amount > balance)
            throw new IllegalStateException("Insufficient funds");
        balance -= amount;
    }

    public void close() {
        active = false;
    }

    public String toString() {
        return "BankAccount{" +
                "ownerName=" + ownerName +
                ", balance=" + balance +
                ", active=" + active +
                '}';
    }
}
```

Now you will see two commits on the main code page

The screenshot shows a GitHub repository interface. At the top, there are navigation links for 'main' (with a dropdown), '1 Branch', '0 Tags', 'Go to file' (with a search icon), 'Add file' (with a dropdown), and 'Code' (with a dropdown). Below this, a commit list is displayed:

- hafuumahmood-hub** Update BankAccount.java ··· 790a270 · now 2 Commits
- BankAccount.java Update BankAccount.java now

Below the commit list is a 'README' section with a 'Readme' icon and a 'Add a README' button. Under the 'Commits' heading, another commit list is shown:

- Commits on Jan 11, 2026
 - Update BankAccount.java ··· hafuumahmood-hub authored now (Verified) 790a270
 - Create BankAccount.java ··· hafuumahmood-hub authored 6 minutes ago (Verified) 575f446

Walkthrough Task 2:

Using GitHub Desktop

Download GitHub Desktop

The screenshot shows the GitHub Desktop download page. At the top, there is a navigation bar with 'GitHub Desktop' (selected), 'Download' (underlined), 'Release Notes', and 'Help'. The main content area has a dark background with white text. It features a large button labeled 'Download GitHub Desktop'.

Focus on what matters instead of fighting with Git. Whether you're new to Git or a seasoned user, GitHub Desktop simplifies your development workflow.

Download for Windows (64bit)

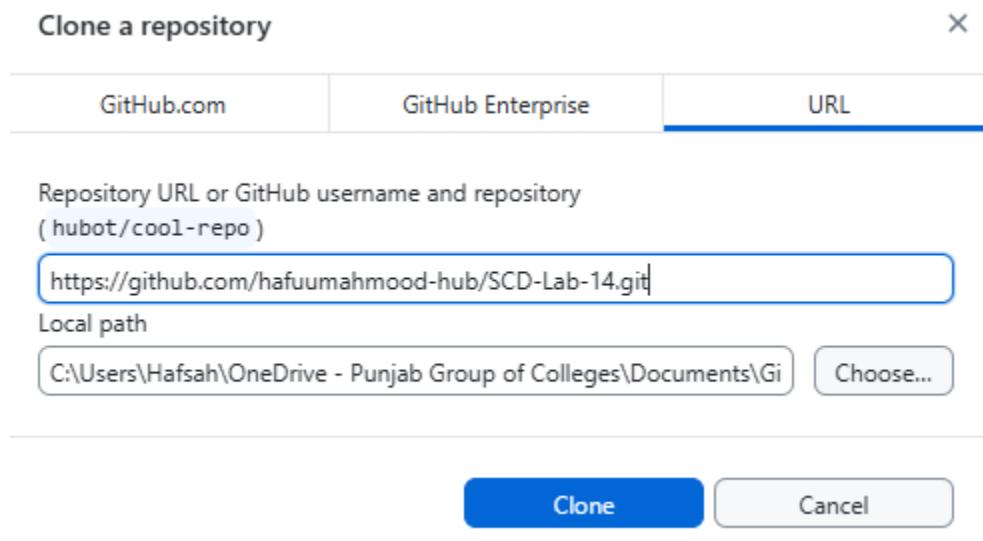
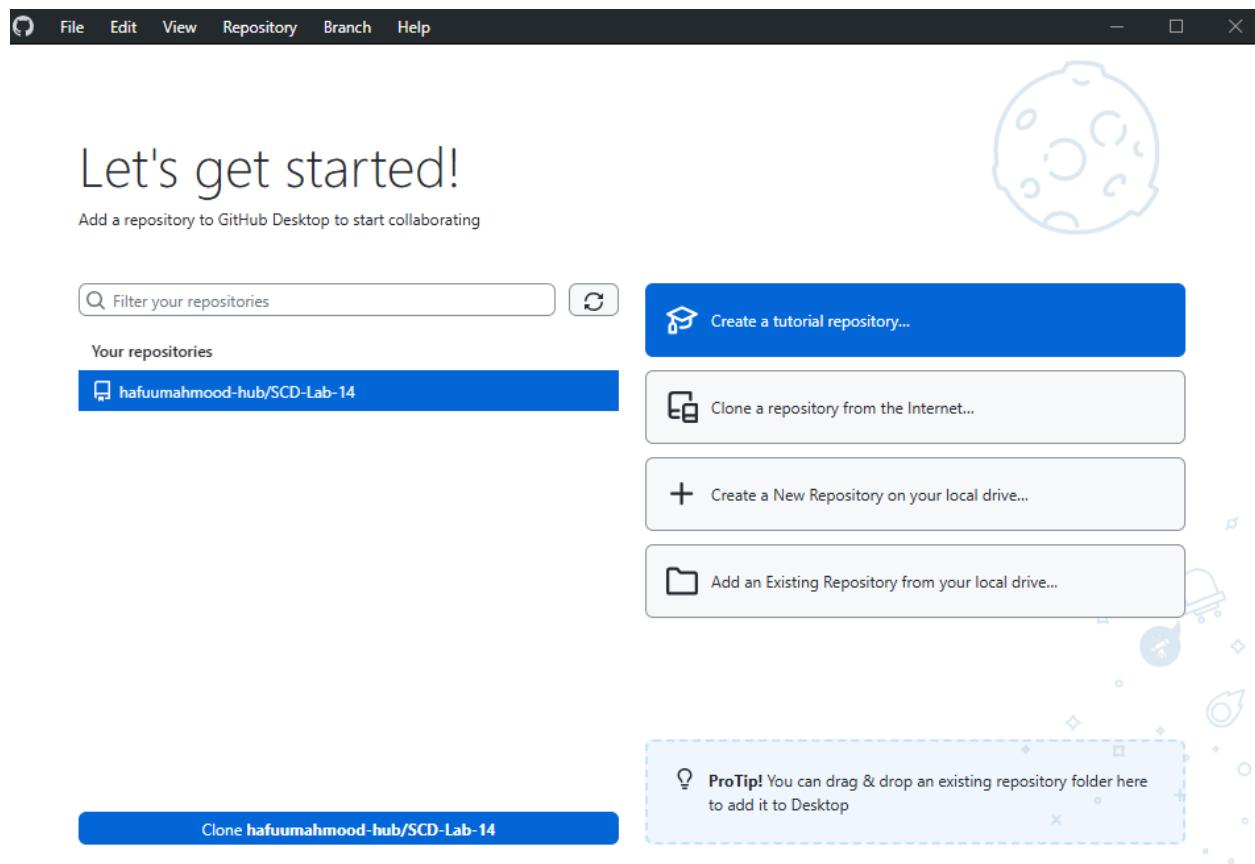
Try beta features and help improve future releases
Experience the latest features and bug fixes before they're released.
[Check out Beta](#)

Prefer the MSI?
Need to download the package to install across your organization?
[Download for Windows \(MSI\)](#)

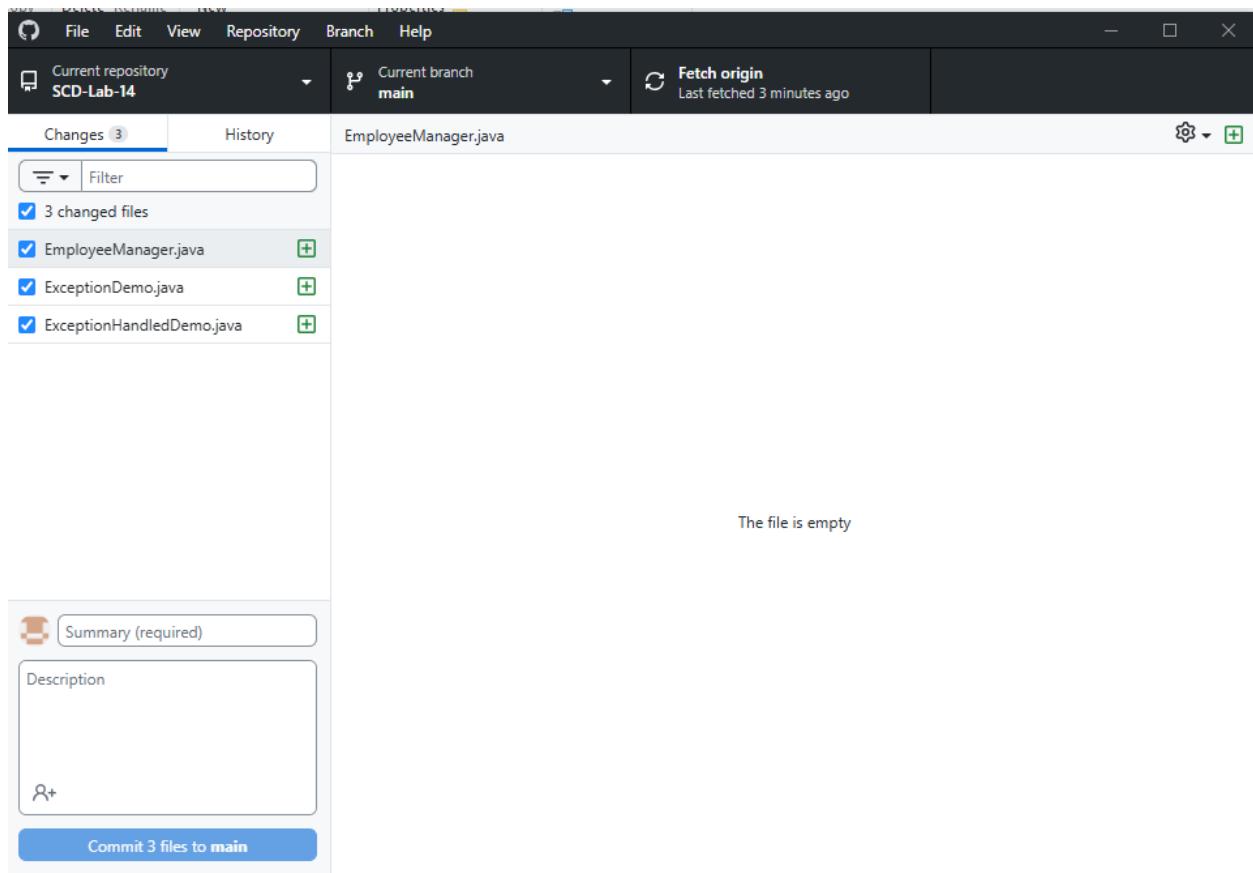
Mac?
Need to download for macOS?
[Download for macOS](#)

By downloading, you agree to the [Open Source Applications Terms](#).

Clone the repository that you created in your online github account to your laptop/PC



Now copy some of the files in your project folder



Now commit these new added files to main

The screenshot shows the GitHub Desktop application interface. At the top, the menu bar includes File, Edit, View, Repository, Branch, and Help. The current repository is set to "SCD-Lab-14" and the current branch is "main". A toolbar at the top right indicates "Push origin" with 1 commit pushed and "Last fetched 4 min...".

The main area displays a message: "No local changes". Below this, it says "There are no uncommitted changes in this repository. Here are some friendly suggestions for what to do next." A small icon of a person working on a laptop is shown.

Suggestions include:

- Push commits to the origin remote**: You have 1 local commit waiting to be pushed to GitHub. A "Push origin" button is available, along with keyboard shortcuts: Always available in the toolbar when there are local commits waiting to be pushed or **Ctrl + P**.
- Open the repository in your external editor**: Select your editor in [Options](#). A "Open in Visual Studio Code" button is shown.
- View the files of your repository in Explorer**: Repository menu or **Ctrl + Shift + F**. A "Show in Explorer" button is shown.
- Open the repository page on GitHub in your browser**: Repository menu or **Ctrl + Shift + G**. A "View on GitHub" button is shown.

On the left side, there's a sidebar with a "Summary (required)" section containing a "Description" field and a "Commit to main" button. Below this, it says "Committed just now" and "Three new files are added", with an "Undo" button.

Push the commits to origin remote

After that your original repository on the internet will also reflect the changes that you made inside your system.

The screenshot shows the GitHub repository page for "hafuumahmood-hub". The repository is public and has 1 branch and 0 tags. The main branch is "main".

The repository has 3 commits from the user "hafuumahmood-hub", all pushed 1 minute ago. The commits are:

- BankAccount.java: Update BankAccount.java, 36 minutes ago
- EmployeeManager.java: Three new files are added, 1 minute ago
- ExceptionDemo.java: Three new files are added, 1 minute ago
- ExceptionHandlerDemo.java: Three new files are added, 1 minute ago

On the left, there's a sidebar with a "README" file. On the right, there's an "About" section with a brief description: "This will cover the basics of how to create the repository and commit changes in it." It also shows statistics: Activity (0), Stars (0), Watching (0), Forks (0), and Releases (No releases published, Create a new release).

Commits

The screenshot shows the GitHub 'Commits' page for a repository named 'SCD-Lab-14'. The main commit is titled 'Three new files are added' by 'hafuumahmood-hub' committed 2 minutes ago. Two other commits are listed: 'Update BankAccount.java' and 'Create BankAccount.java', both authored by 'hafuumahmood-hub' 37 and 42 minutes ago respectively. All commits are verified.

Now if you make any changes on the script on your web GitHub you will need to pull those changes into your desktop app.

The screenshot shows the GitHub Desktop application interface. The top bar shows the repository 'SCD-Lab-14' and the current branch 'main'. A message in the center says 'No local changes' with a sub-message: 'There are no uncommitted changes in this repository. Here are some friendly suggestions for what to do next.' It lists three suggestions: 'Pull 1 commit from the origin remote', 'Open the repository in your external editor', and 'View the files of your repository in Explorer'. At the bottom left, there is a summary box with a 'Commit to main' button.

Click on pull origin and your both directories will be synced.

Practice Tasks:

Practice Task 1: Team-Based Git Collaboration on a Mini Application

Objective

To enable students to collaboratively build different modules of a small management system while using GitHub as the shared version control platform for integration, conflict handling, and synchronization

Task Description

Students are required to perform the following steps:

- 1. Form teams of 3–4 members.**
- 2. Select one mini application to develop**, such as (examples):
 - Library Management System
 - Inventory Management System
 - Course Registration System
 - Hostel/Room Allocation System
 - Clinic Appointment System
 - Vehicle Service Tracking System
 - Simple Payroll System
- 3. Assign modules to each member**, such as:
 - Member A → User/Student/Employee Module
 - Member B → Resource/Inventory/Asset Module
 - Member C → Records/Transactions/Bookings
 - Member D (optional) → Reporting / Authentication / Admin Panel
- 4. Create a single GitHub repository** for the team and invite all members as collaborators.
- 5. Each member must:**
 - Clone the repository
 - Develop their assigned module locally
 - Commit changes with meaningful messages
 - Push changes to the main project
- 6. Teams must create at least one separate branch per member**, such as:
 - a) module-user
 - b) module-inventory
 - c) module-transactions
 - d) module-report

7. System must include **at minimum**:
 - o Basic CRUD operations for each module
 - o Input validation & error handling
 - o Simple console or GUI interface
8. Final submission artifacts:
 - Repository link
 - Screenshots of commits
 - Screenshot of merges

Outcomes:

- Students will understand version control concepts using Git.
- Students will be able to stage, commit, push, and pull code using GitHub.
- Students will handle real-world synchronization issues and merge conflicts.
- Students will collaborate through GitHub repositories.
- Students will appreciate GitHub as a professional development, deployment, and teamwork tool.