Experiment No. 2:

Creating an Account on GitHub, Creating a Repository, and Uploading Files/Projects on GitHub.

Aim of the Experiment:

The aim of this experiment is to create an account on GitHub, set up a new repository, and upload files or projects to GitHub using both the GitHub web interface and Git commands. This experiment helps in understanding how to use GitHub for remote version control, collaboration, and project management.

Objective:

The objective of this experiment is to:

- Understand **GitHub** as a remote repository hosting service.
- Create an account on **GitHub**.
- Create a new **repository** on GitHub.
- Upload files or an entire project to GitHub.
- Understand the basic commands used for pushing a local repository to GitHub.

Theory:

What is GitHub?

GitHub is a cloud-based platform that allows developers to store, manage, share, and collaborate on projects using Git. It provides a remote repository where multiple users can contribute to the same project.

Features of GitHub:

- Remote repository hosting for Git-based projects.
- Collaboration among multiple developers.
- Version control for tracking changes.
- Pull requests and code reviews for teamwork.
- Issue tracking and project management.

Difference Between Git and GitHub:

Feature	Git	GitHub
Type	Version Control System	Cloud-based Git repository hosting service
Storage	Local	Remote (Cloud)
Purpose	Tracks changes and manages versions	Allows sharing, collaboration, and remote storage
Commands Used	git init, git commit, git branch, git merge	git push, git pull, git clone

Key Concepts:

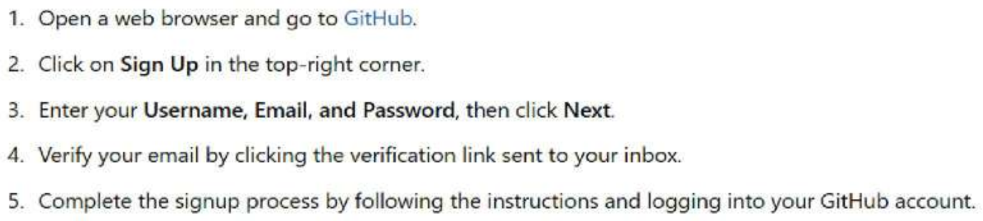
- **Repository**: A collection of files and folders managed by Git.
- **Commit**: A saved change in Git.
- **Staging Area**: A place where files are prepared for a commit.
- **Branch**: A separate version of a repository.

Requirements:

- A computer with internet access.
- A GitHub account.
- Git installed on your system (Refer to Experiment-1 for installation).
- A local repository for uploading files to GitHub.

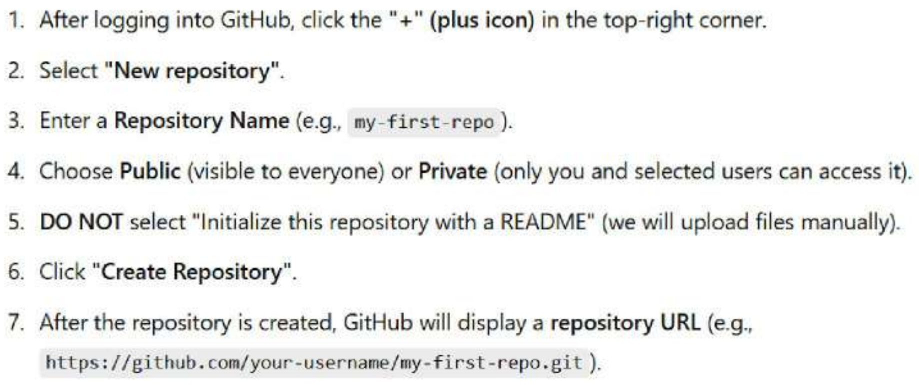
Procedure:

Step 1: Create a GitHub Account



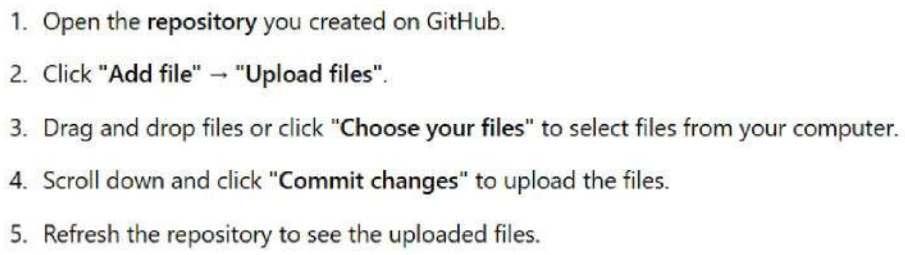
1. Open a web browser and go to GitHub.
2. Click on **Sign Up** in the top-right corner.
3. Enter your **Username, Email, and Password**, then click **Next**.
4. Verify your email by clicking the verification link sent to your inbox.
5. Complete the signup process by following the instructions and logging into your GitHub account.

Step 2: Create a New Repository on GitHub



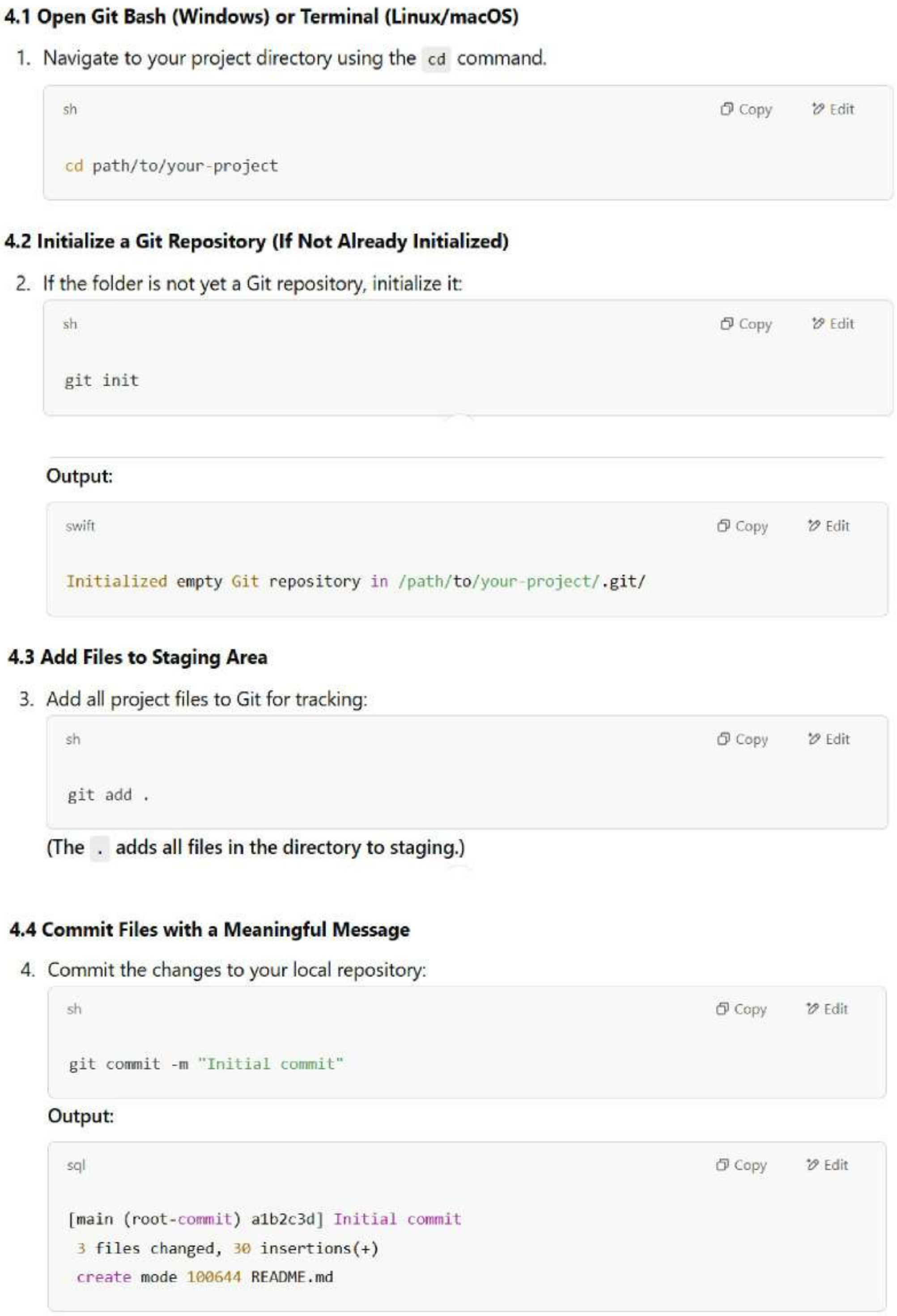
1. After logging into GitHub, click the **"+" (plus icon)** in the top-right corner.
2. Select **"New repository"**.
3. Enter a **Repository Name** (e.g., `my-first-repo`).
4. Choose **Public** (visible to everyone) or **Private** (only you and selected users can access it).
5. **DO NOT** select "Initialize this repository with a README" (we will upload files manually).
6. Click **"Create Repository"**.
7. After the repository is created, GitHub will display a **repository URL** (e.g., `https://github.com/your-username/my-first-repo.git`).

Step 3: Upload Files Using GitHub Web Interface (Easy Method - No Git Commands Needed)



1. Open the **repository** you created on GitHub.
2. Click **"Add file"** → **"Upload files"**.
3. Drag and drop files or click **"Choose your files"** to select files from your computer.
4. Scroll down and click **"Commit changes"** to upload the files.
5. Refresh the repository to see the uploaded files.

Step 4: Uploading a Local Project to GitHub Using Git Commands (Recommended Method - Command Line Approach)



4.1 Open Git Bash (Windows) or Terminal (Linux/macOS)

1. Navigate to your project directory using the `cd` command.

```sh
cd path/to/your-project
```

4.2 Initialize a Git Repository (If Not Already Initialized)

2. If the folder is not yet a Git repository, initialize it:

```sh
git init
```

Output:

```swift
Initialized empty Git repository in /path/to/your-project/.git/
```

4.3 Add Files to Staging Area

3. Add all project files to Git for tracking:

```sh
git add .
```

(The `.` adds all files in the directory to staging.)

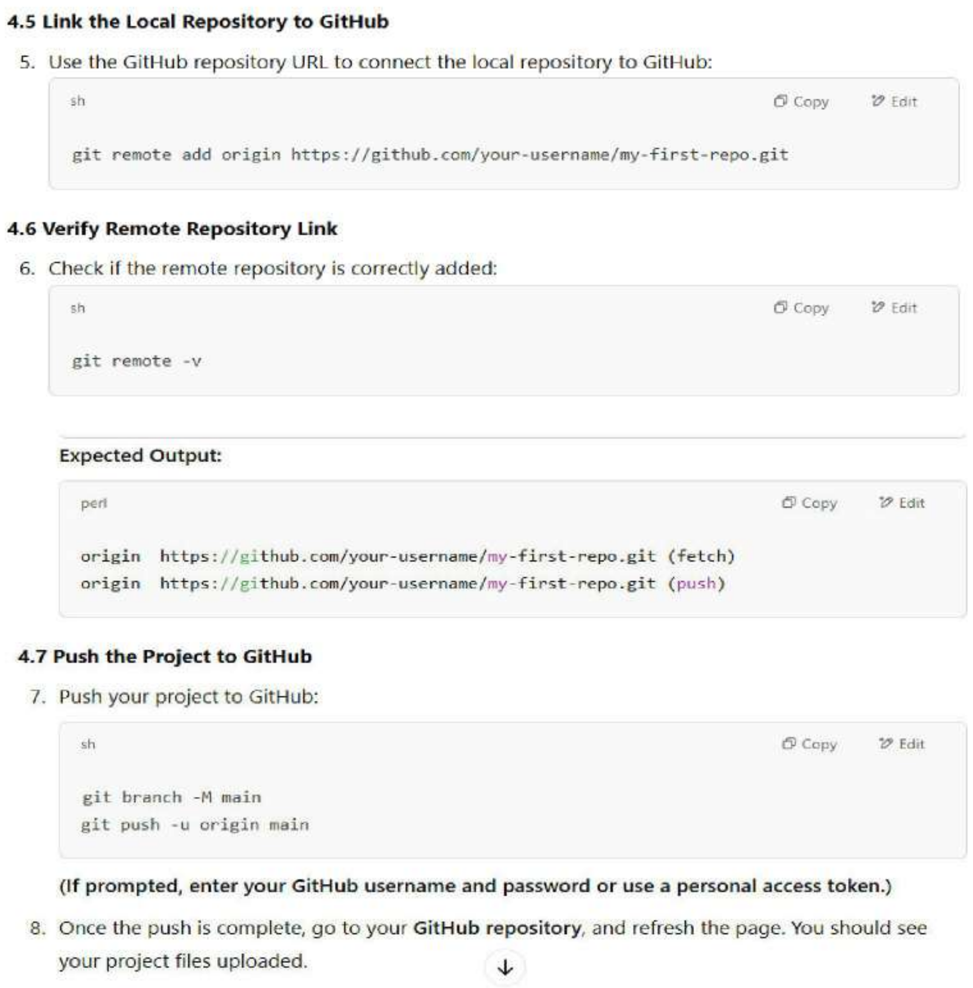
4.4 Commit Files with a Meaningful Message

4. Commit the changes to your local repository:

```sh
git commit -m "Initial commit"
```

Output:

```sql
[main (root-commit) a1b2c3d] Initial commit
 3 files changed, 30 insertions(+)
 create mode 100644 README.md
```



4.5 Link the Local Repository to GitHub

5. Use the GitHub repository URL to connect the local repository to GitHub:

```sh
git remote add origin https://github.com/your-username/my-first-repo.git
```

4.6 Verify Remote Repository Link

6. Check if the remote repository is correctly added:

```sh
git remote -v
```

Expected Output:

```perl
origin  https://github.com/your-username/my-first-repo.git (fetch)
origin  https://github.com/your-username/my-first-repo.git (push)
```

4.7 Push the Project to GitHub

7. Push your project to GitHub:

```sh
git branch -M main
git push -u origin main
```

(If prompted, enter your GitHub username and password or use a personal access token.)

8. Once the push is complete, go to your **GitHub repository**, and refresh the page. You should see your project files uploaded.

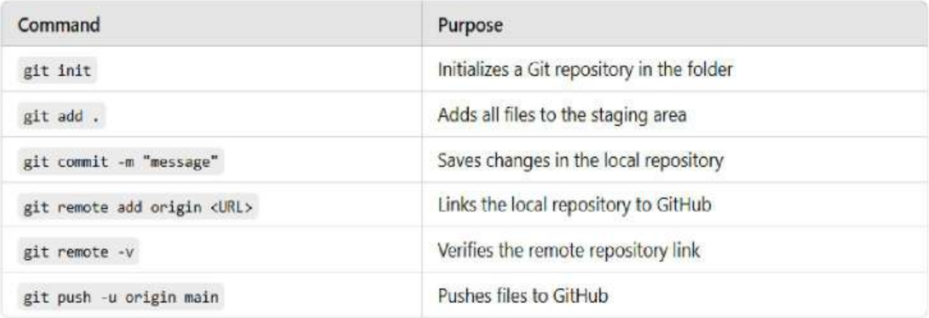
Step 5: Verify the Uploaded Files on GitHub

- Open GitHub and go to your repository.
- Check if all files have been uploaded successfully.
- Click on any file to view its content and commit history.

Expected Output:

- Successfully created a GitHub account.
- Set up a new GitHub repository.
- Uploaded files/Project using both the GitHub Web Interface and Git Commands.
- The `git push` command successfully updated the repository on GitHub.
- Verified that the files are available on GitHub for public or private access.

Important Git Commands Used in This Experiment:



Command	Purpose
`git init`	Initializes a Git repository in the folder
`git add .`	Adds all files to the staging area
`git commit -m "message"`	Saves changes in the local repository
`git remote add origin <URL>`	Links the local repository to GitHub
`git remote -v`	Verifies the remote repository link
`git push -u origin main`	Pushes files to GitHub

Conclusion:

This experiment successfully demonstrated how to:

- Create a GitHub account.
- Set up a repository on GitHub.
- Upload files using both GitHub Web Interface and Git Commands.
- Use Git commands to initialize, commit, and push changes to GitHub.

This knowledge is essential for collaborative software development and version control.

Viva Questions on GitHub

1. What is GitHub, and how does it differ from Git?
2. How do you create an account on GitHub?
3. What are the key features of GitHub?
4. How do you create a new repository on GitHub?
5. What is a remote repository in GitHub?
6. How do you upload files to GitHub using the web interface?
7. What is the difference between a public and private repository in GitHub?
8. What is the purpose of the README.md file in a GitHub repository?
9. How do you clone an existing repository from GitHub?
10. What is the purpose of a commit message in GitHub?
11. How do you connect a local Git repository to a GitHub repository?
12. What is the command to push changes to a GitHub repository?
13. How do you check if your repository is linked to a remote GitHub repository?
14. What is the purpose of a .gitignore file in GitHub?
15. How do you update an existing repository on GitHub with new changes?
16. How do you create a branch in a GitHub repository?
17. What is the purpose of a pull request in GitHub?
18. How do you fork a repository on GitHub?
19. What is GitHub Actions, and how is it used?
20. How do you delete a repository from GitHub?
21. What is the difference between git pull and git fetch in GitHub?
22. How do you invite collaborators to a GitHub repository?
23. What are GitHub Issues, and how are they used?
24. How do you revert a commit in GitHub?
25. What is GitHub Pages, and how is it used?
26. How do you check the remote repository link in Git?
27. How do you rename a branch in GitHub?
28. What happens if you push changes to GitHub without configuring your username and email?
29. How do you merge branches in GitHub?
30. What is the purpose of GitHub Gists, and how are they different from repositories?

Answer for Via Questions

Here are the **answers** to all the **Viva Questions** related to Git Hub:

1. What is GitHub, and how does it differ from Git?

Answer:

GitHub is a **cloud-based service** that allows users to store, manage, and collaborate on Git repositories. Git is a **version control system** that manages changes in source code. GitHub provides an online interface to store and share Git repositories.

Example:

Git is installed on your local machine and can track changes. GitHub allows you to upload and share your project with others online.

2. How do you create an account on GitHub?

Answer:

1. Go to GitHub in a web browser.
2. Click **Sign Up**.
3. Enter a **Username, Email, and Password**.
4. Click **Verify** and check your email for a confirmation link.
5. Complete the setup and log in.

3. What are the key features of GitHub?

Answer:

- **Repository hosting:** Store projects online.
- **Collaboration tools:** Pull Requests, Issues, and Discussions.
- **Version control:** Track changes in files.
- **Branching and merging:** Work on different versions of a project.
- **CI/CD support:** Automate builds using GitHub Actions.

Example:

A team of developers can work on the same project by pushing their changes to GitHub and using pull requests to merge them.

4. How do you create a new repository on GitHub?

Answer:

1. Log into GitHub.
2. Click **"+"** (top-right) → **"New repository"**.
3. Enter a **Repository Name** (e.g., my-project).
4. Select **Public** or **Private**.
5. Click **"Create Repository"**.

5. What is a remote repository in GitHub?

Answer:
A **remote repository** is a Git repository stored on a cloud server like GitHub.

Example:

```
git remote add origin https://github.com/user/my-project.git
```

This links your local project to GitHub.

6. How do you upload files to GitHub using the web interface?

Answer:

1. Open your repository on GitHub.
2. Click **"Add file"** → **"Upload files"**.
3. Select files from your computer.
4. Click **"Commit changes"**.

7. What is the difference between a public and private repository in GitHub?

Answer:

- **Public Repository**: Anyone can view the code.
- **Private Repository**: Only invited users can access it.

Example:
When creating a repository, select **"Public"** or **"Private"**.

8. What is the purpose of the README.md file in a GitHub repository?

Answer:
A README.md file provides a project description, installation instructions, and usage details.

Example:

```
# My Project

This is a sample project demonstrating GitHub usage.
```

9. How do you clone an existing repository from GitHub?

Answer:
Use:

```
git clone https://github.com/user/repository.git
```

This downloads the project to your local machine.

10. What is the purpose of a commit message in GitHub?

Answer:
A commit message describes the changes made in a commit.

Example:

```
git commit -m "Fixed a bug in the login module"
```

11. How do you connect a local Git repository to a GitHub repository?

Answer:

```
git remote add origin https://github.com/user/repository.git
```

This links your local project to GitHub.

12. What is the command to push changes to a GitHub repository?

Answer:

```
git push -u origin main
```

Uploads local commits to GitHub.

13. How do you check if your repository is linked to a remote GitHub repository?

Answer:

```
git remote -v
```

Example Output:

```
origin  https://github.com/user/repository.git (fetch)
origin  https://github.com/user/repository.git (push)
```

14. What is the purpose of a .gitignore file in GitHub?

Answer:
It tells Git to **ignore** specific files.

Example:
Create .gitignore and add:

```
node_modules/
.env
*.log
```

15. How do you update an existing repository on GitHub with new changes?

Answer:

```
git add .
git commit -m "Updated project"
git push origin main
```

16. How do you create a branch in a GitHub repository?

Answer:

```
git checkout -b new-branch
```

Creates and switches to a new branch.

17. What is the purpose of a pull request in GitHub?

Answer:
A **pull request** is used to propose code changes and request review before merging.

18. How do you fork a repository on GitHub?

Answer:
Click **"Fork"** on a repository's page to create a copy in your account.

19. What is GitHub Actions, and how is it used?

Answer:
GitHub Actions automates tasks like testing and deployment.

20. How do you delete a repository from GitHub?

Answer:
Go to **Settings** → **Danger Zone** → **Delete Repository**.

21. What is the difference between git pull and git fetch in GitHub?

Answer:

- git pull: Fetches and **merges** changes.
- git fetch: Fetches changes **without merging**.

22. How do you invite collaborators to a GitHub repository?

Answer:

Go to **Settings** → **Manage Access** → **Invite Collaborator**.

23. What are GitHub Issues, and how are they used?

Answer:

GitHub Issues track bugs, enhancements, and tasks.

24. How do you revert a commit in GitHub?

Answer:

```
git revert <commit-hash>
```

25. What is GitHub Pages, and how is it used?

Answer:

GitHub Pages hosts static websites from a repository.

26. How do you check the remote repository link in Git?

Answer:

```
git remote -v
```

27. How do you rename a branch in GitHub?

Answer:

```
git branch -m old-name new-name
```

28. What happens if you push changes to GitHub without configuring your username and email?

Answer:

Git will prompt an error. Set user details with:

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

29. How do you merge branches in GitHub?

Answer:

```
git checkout main
git merge feature-branch
```

30. What is the purpose of GitHub Gists, and how are they different from repositories?

Answer:

GitHub Gists allow you to share small code snippets publicly or privately, while repositories store full projects.