Hands-on Lab: Setting up the Environment

Skills Network

Estimated time needed: 15 minutes

What you will learn

In this lab, you will set up an online repository for the project and learn how to create folders and files in the **Skills Network Environment**. You will also understand how to view the output of your code.

Learning objectives

After completing this lab, you will be able to:

- Create an online repository.
- Create folders and files in **Skills Network Environment**.
- Run the code to see the output.
- Perform Git operations.

Prerequisites

- You must have completed the prerequisite courses, especially the **Getting Started with Git and GitHub** course.

- You must have a GitHub account. If you want to set up a GitHub account, click **here** for the detailed steps.

Step 1: Create an online repository and create personal access token

1. Create a blank public GitHub repository in your GitHub account without creating any README.md file for this. Make sure you set your repository to public.and name it accordingly.

2. You can find more information by clicking on this Create_Github_repository

3. **Personal Access Token** is now a mandatory part to push data to GitHub repositories to make sure your authentication is related to your GitHub account.

4. To create a **Personal Access Token**, go to your GitHub account and click on your profile icon located in the top-right corner. Then click on **Settings**.

javaScriptEssential

- 🖳 Your codespaces
- 🏢 Your organizations
- 🌐 Your enterprises
- ☆ Your stars
- ♡ Your sponsors
- ⟨⟩ Your gists

- ⬆ Upgrade
- 🌐 Try Enterprise
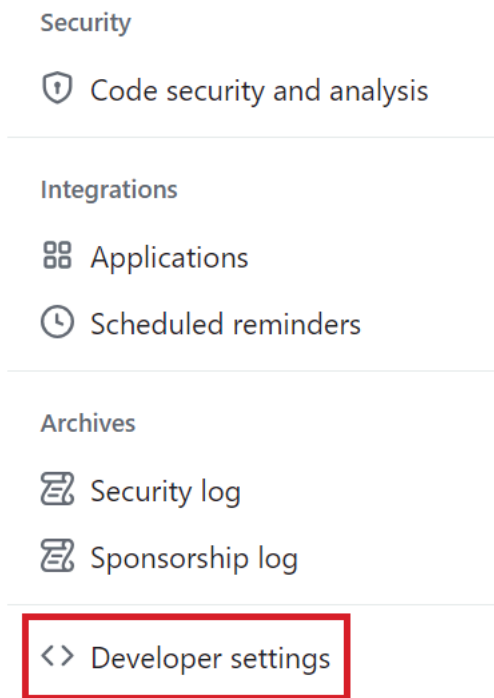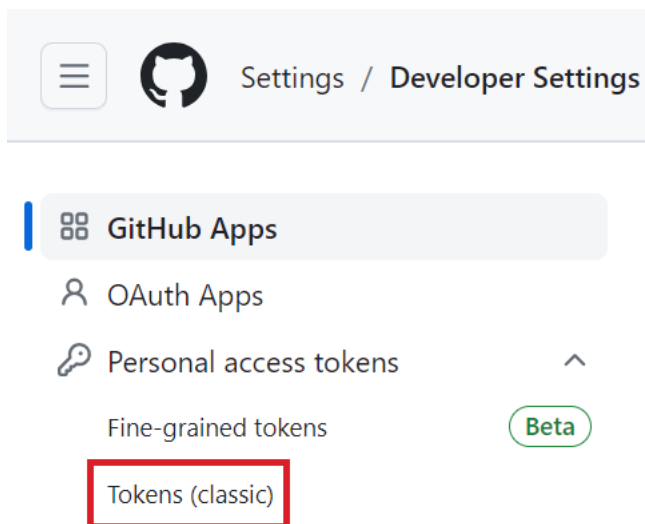- 🐙 Try Copilot
- ⚗ Feature preview
- ⚙ Settings

- 📖 GitHub Docs
- 👥 GitHub Support

5. Next, select **Developer settings**. This option is typically available towards the bottom of the window.

Security

🛡 Code security and analysis

Integrations

⊞ Applications

🕐 Scheduled reminders

Archives

🗐 Security log

🗐 Sponsorship log

〈〉 Developer settings

6. Navigate to **Tokens (classic)** under **Personal access tokens**.

☰ ⬤ Settings / **Developer Settings**

⊞ **GitHub Apps**

🗛 OAuth Apps

🔑 Personal access tokens   ⌃

   Fine-grained tokens   (Beta)

   Tokens (classic)

7. To generate an access token, click **Generate a personal access token**.

Personal access tokens (classic)

Need an API token for scripts or testing? Generate a personal access token for quick acc
API.

Personal access tokens (classic) function like ordinary OAuth access tokens. They can be used instead o
over HTTPS, or can be used to authenticate to the API over Basic Authentication.

8. In the **Generate token** page, fill in the required details and click the **repo** checkbox to enable access for `git` commands.

New personal access token (classic)

Personal access tokens (classic) function like ordinary OAuth access tokens. They can be password for Git over HTTPS, or can be used to authenticate to the API over Basic Auth

Note

> javaScriptEssential

What's this token for?

Expiration *

> 30 days   ⇅   The token will expire on Tue, Sep 5 2023

Select scopes

Scopes define the access for personal tokens. Read more about OAuth scopes.

- ☑ **repo**   Full control of private repositories
  - ☑ repo:status   Access commit status
  - ☑ repo_deployment   Access deployment status
  - ☑ public_repo   Access public repositories
  - ☑ repo:invite   Access repository invitations
  - ☑ security_events   Read and write security events

9. Then, click **Generate token**.

- ☐ **admin:ssh_signing_key**   Full control of public user SSH signing keys
  - ☐ write:ssh_signing_key   Write public user SSH signing keys
  - ☐ read:ssh_signing_key   Read public user SSH signing keys

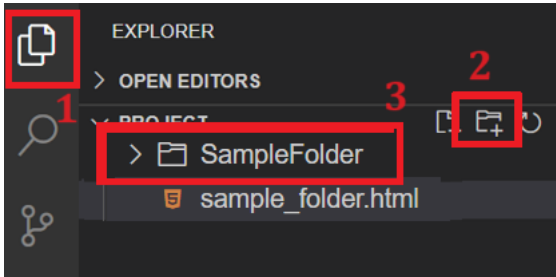**Generate token**   Cancel

10. Your personal access token will be generated. The token is only valid for **30 days**. You will need to generate a new token once the current token expires.

REMEMBER: Make sure to copy your personal access token now. You won't be able to see it again! In case you forgot to save it or misplaced it, delete the already created token and generate new one.

Note: This repository will be your main repository for all labs except the practice project lab and the final project lab.

Step 2: Create files in Skills Network Environment

1. In the right window pane, click on Explorer, as shown in the screenshot below (refer to number 1). Next, click on the project folder, and then click the icon highlighted in red (refer to number 2) in the screenshot. Enter the folder name as **SampleFolder**. This action will create a folder for you. Next, select the **SampleFolder** folder shown at number 3, right-click, and select **New File**. Enter the file name as **sample_folder.html** and click OK. This will create the HTML file within the **SampleFolder**.
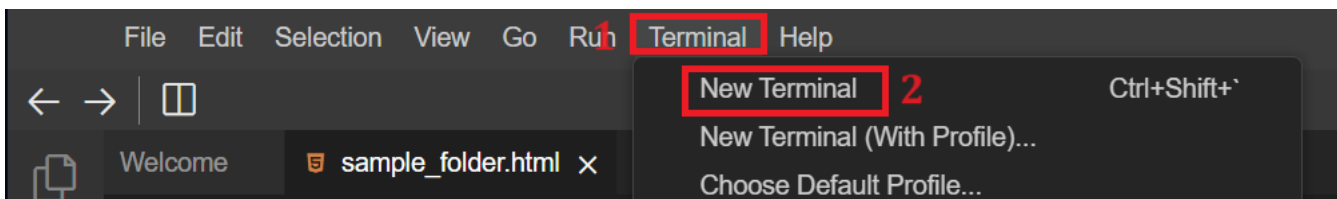

2. Create the basic template structure in the HTML file **sample_folder.html** by copying the provided content and pasting in the **sample_folder.html** file.

```
<!DOCTYPE html>
<html>
<head>
   <title>Sample Folder</title>
</head>
<body>
   <h1>Sample Folder</h1>
   <button>SampleFolder</button>
</body>
</html>
```

Note: After pasting the code, save your file.

Step 3: Perform Git commands in the terminal

1. Now, click on the "Terminal" tab at the top-right of the window, and then click on "New Terminal" as shown in the given screenshot.

2. Next, within the path of the terminal, initialize it to create a Git repository to perform all Git commands in this terminal. Use the provided command for this purpose in the terminal and press **Enter**.

```
git init
```

3. Then you need to execute this command in the terminal for performing necessary commands within the **Skills Network Environment** and press **Enter**.

```
git config --global --add safe.directory /home/project
```

Note: Above command will help you to work inside project folder environment with git commands.

4. Then,

   - set the `git config --global`for email address by performing given command in terminal by providing email address of your GitHub account within double qoutes instead of **you@example.com** and press **Enter**.

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

   - Again,set the `git config --global`for username by performing given command in terminal by providing username of your GitHub account within double qoutes instead of **Your Name** and press **Enter**.

```
git config --global user.name "Your Name"
```
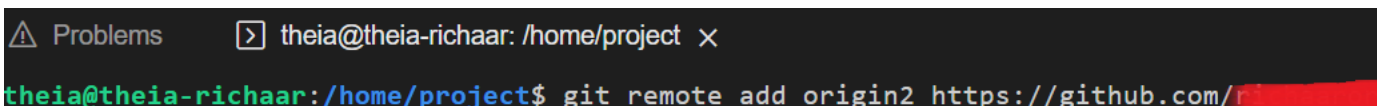
5. Next, perform `git add` and `git commit` to save the changes for GitHub repository by running the following commands one after another:

```
git add --a
```

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

6. Perform `git push` commands to push the files into your GitHub repository.

   - Add your GitHub repository URL in origin2 variable by writing given command in terminal.
   - Also replace entire `<git-repo-url>` with your GitHub repository URL such as `git remote add origin2` https://github.com//youraccountname//yourrepositoryname as shown in given screenshot and click **Enter**.


```
git remote add origin2 <git-repo-url>
```

   - Then, perform the given command in the terminal to push the content of your file in GitHub repository and click **Enter**.

```
git push origin2
```

7. While pushing the files in GitHub using `git push` command, it will ask you to enter the username for your GitHub account in the terminal. Enter your username and then press enter. Next, it will also ask for your password, here you need to paste the **Personal Access Token** that you generated in step 1.

   Note: Upon pasting your **Personal Access Token** into the terminal, it won't show for security reasons, but it's already there. Simply hit enter, and it will push your files and folders to the GitHub repository.

8. It will push all the files directly into your GitHub repository.

   Note: You can also refer to git_commands for more detailed instruction.

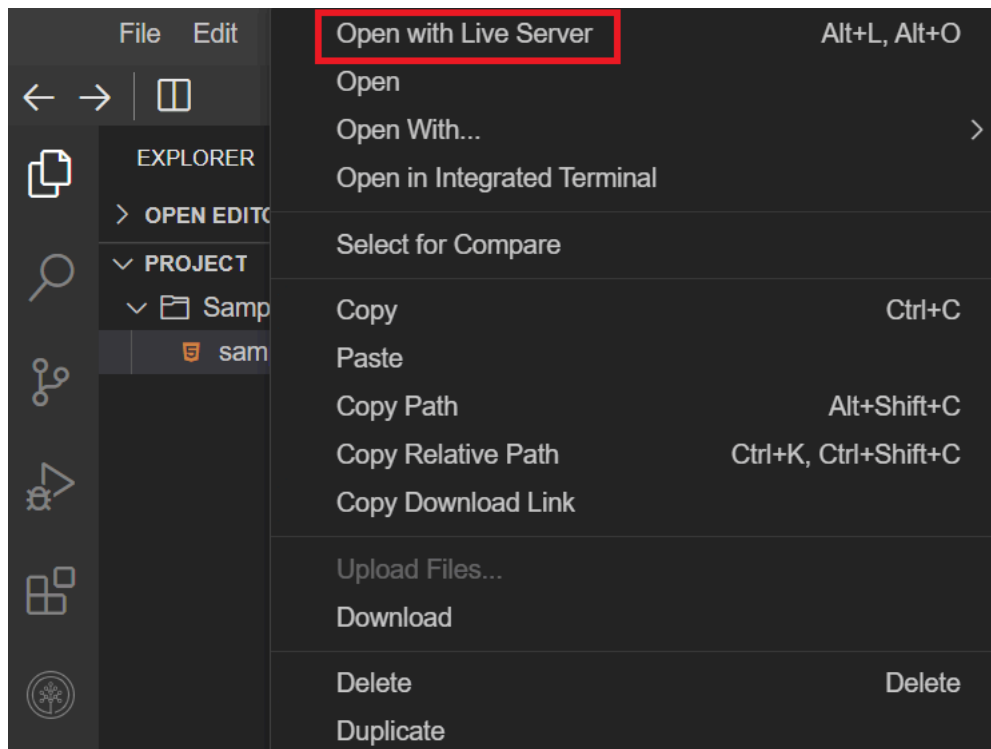# Step 4: Need to perform Git commands

It is crucial to follow a few essential steps to ensure the proper management and persistence of your data in a GitHub repository:

- **Regular updates:** Whenever you make changes or add new components to your project, adding, committing, and pushing the updates to your GitHub repository is essential. This process ensures that your latest work is safely stored and accessible to collaborators.

- **Session persistence:** During an active session, your data remains accessible. However, it's important to note that if your session expires or you log out, you will need to clone the repository again to resume work.

By adhering to these guidelines, you can maintain a well-organized and efficient GitHub repository, ensuring your work is securely stored and easily accessible to you and your collaborators.
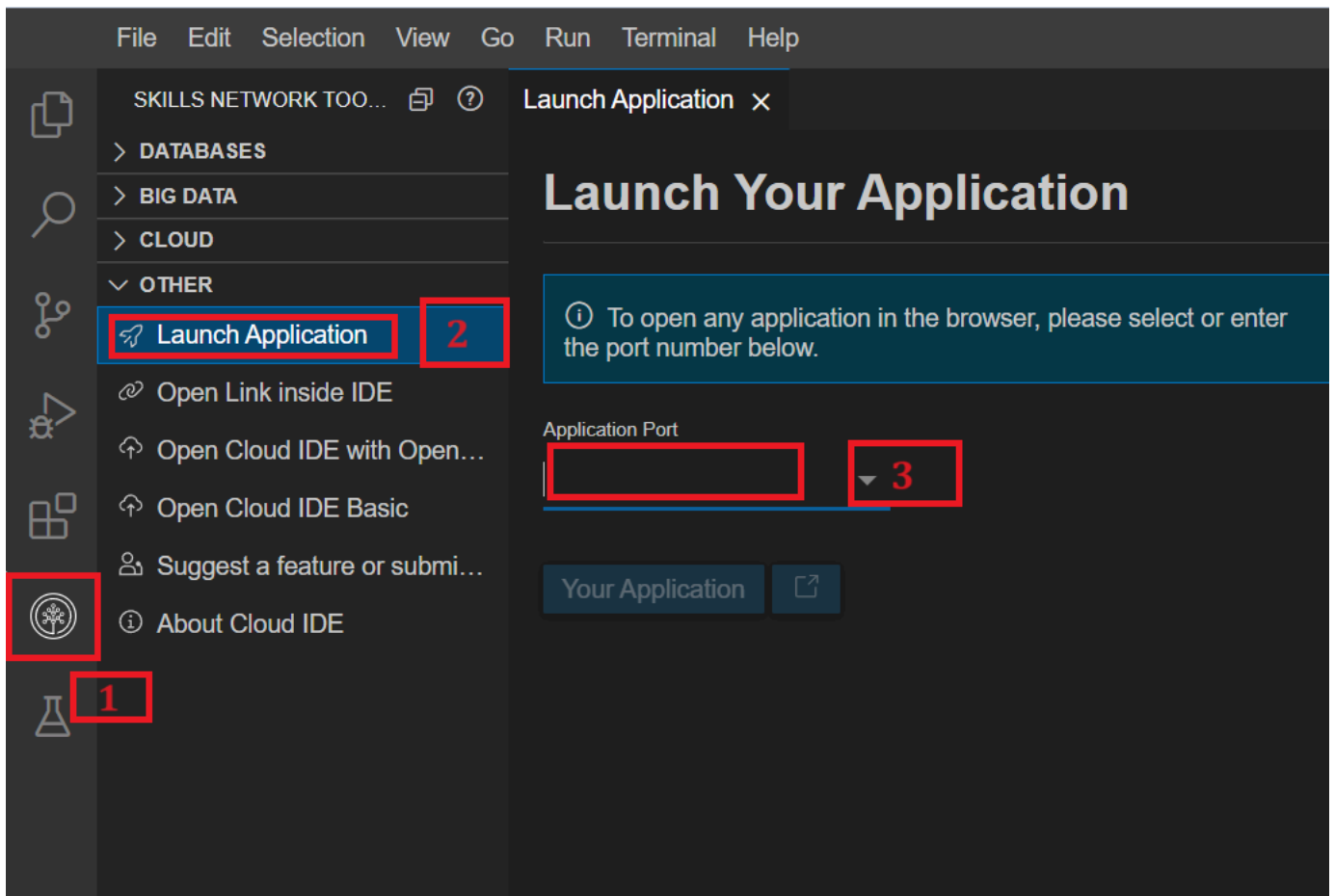
Step 5: Check the output

1. To view how your HTML page will be displayed in the browser, use the built-in Live Server extension. Select file **sample_folder.html** within **SampleFolder** folder and right click on that file and choose 'Open with Live Server'.
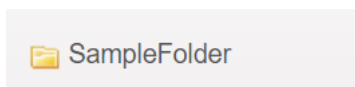

2. A notification will appear at the bottom right, indicating that the server has started on port 5500.

3. Then, click the Skills Network button on the left (refer to number 1). This action will open the "Skills Network Toolbox." Next, select "Launch Application"

   (refer to number 2). Once there, enter port number **5500** in "Application Port" (refer to number 3) and click this button .
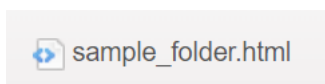
4. It will open your default browser, where you will see the **SampleFolder** folder name.

3. Click on that folder name. It will show you the available file names within that folder.

4. Then click on the **sample_folder.html** file, as shown below.

5. The front page will open, allowing you to see the output.

Note: Remember to save your files regularly. If you edit your code, refresh your browser, which runs through port number 5500. This way, there is no need to launch the application again. Also, push all the latest changes of code in GitHub repository as well by performing git add, git commit, and git push commands.