Lab: Getting Started with GitHub (Optional)

Objectives

- Create a GitHub account
- Add a private repository to your GitHub account
- Create and edit a file in your GitHub repository
- Create a new fork of an existing GitHub repository

Introduction

Welcome to the Getting Started with GitHub lab.

To understand GitHub, you should first understand Git. Git is an open-source project management application. Git is also a distributed version control system. In this system, each team member has a copy of the entire project on their computer. This copy, called a repository, includes the project's revision history. Although multiple team members might be working with the same file, each member can be sure that they are working from the current version.

GitHub is a popular online hosting service for Git repositories. With Git, you must use the command line to manage your project. But with GitHub, you can perform these tasks with the click of a mouse through GitHub's graphical user interface (GUI). GitHub also provides useful collaboration features like wikis and bug tracking.

In this lab, you will perform basic GitHub tasks. You will set up a GitHub account and create a repository on your GitHub page. Then you will modify your repository by creating and editing a new file. You will also discover how to commit, or save, each change.

Note that this lab is a prerequisite for the next lab: Scanning for Code Vulnerabilities with Snyk. In that lab, you will scan the code in your GitHub repositories for vulnerabilities.

What you need

You need only a web browser and a personal email account. GitHub provides everything else.

Task 1: Create a GitHub account

1. Access https://github.com/join using your browser.

2. On the Join GitHub page, provide the requested information to create a GitHub account. Next, click **Continue** and complete the simple puzzle to confirm that you are not a bot. Finally, click **Submit**.

3. When prompted to enter a launch (verification) code, check your email for the code, and then type it into the space provided.

4. The Welcome to GitHub page is displayed. Scroll to the end of the page, and then select **Skip personalization.**

5. The GitHub dashboard is displayed. The dashboard alerts you to updates on recent activities, and you can use it to track issues related to projects that you are working on.

MY NOTES *** This lab served as a prerequisite for the upcoming **Scanning for Code Vulnerabilities with Snyk** activity. Since I already had an existing GitHub account, I skipped Task 1 and began with Task 2: creating a new repository. The lab walked through the basic GitHub interface and setup process, preparing the environment for future vulnerability scanning and secure code practices.

Task 2: Create a repository

1. Click **Create repository.**

2. Now you will provide information to create your repository.
   • In the **Repository name** field, type 'test'.
   • Click **Private** to ensure that you can choose who can see and commit to this repository.

   MY NOTE *** For Task 2, I created a new public repository as part of the required setup. Although the lab instructions recommend creating a private repository, I chose to make mine public so that it can serve as part of my professional portfolio. A public repository allows instructors, peers, and potential employers to review my work and view version history, which supports both transparency and professional development.
   • Select the **Add a README file** checkbox.
   • Click **Create repository.**

3. The home page for your newly created repository is displayed. You can manage your repository's files and folders on this page.

Task 3: Create and edit a file

1. Click **Add File**, and then select **Create new file**

from the list.

2. In the test/ field, type **helloworld.py**, which will be your new file's name.

Note that the **.py** file extension indicates that this file is a Python file. Python is a popular, easy-to-learn scripting language for developing web and software applications. You can use it to accomplish nearly any cybersecurity task that you need, making Python the programming language of choice for cybersecurity professionals.

3. Scroll to the **Commit new file section**. Next, click **Commit directly to the main branch** if the option isn't selected. Finally, click **Commit new file**.

4. Your newly created file should be displayed on the test repository's home page. Click **helloworld.py**.

5. Now you'll edit the file. To do so, first, click the pencil icon.

6. The GitHub file editor is displayed. Type or copy and paste the following text into your editor:

```
# This file will print Hello, world!
Print ('Hello, world!')
```

7. Scroll to the **Commit changes** section. Next, click **Commit directly to the main branch** if the option isn't selected. Finally, click **Commit changes**.

By committing a change in GitHub, you indicate that this new code is a stable and valuable addition to the repository. If you were working with others on this project, you would next issue a pull request. A pull request (PR) is a mechanism by which you notify others that your changes are ready for review. After reviewing and approving these changes, the project manager would merge them into the repository.

8. The change you committed to helloworld.py should be displayed on the test repository's home page.

Task 4: Create a new fork

GitHub has an extensive collection of public repositories full of valuable resources for developers. You will now create a fork of one of these repositories. A fork is a copy of an existing repository. With a fork, you can change the repository's content without impacting the original project.

1. Access https://github.com/bitnami/containers from your browser.
   Bitnami provides installers, virtual machines, and other packages for running popular applications from standard operating systems.

2. The Bitnami Containers Library is displayed, which is an extensive repository of containerized applications available for developer use. Click **Fork**.

Note: A container is a package of application code and its dependencies that runs in a virtual environment.

3. The **Create a new fork** page is displayed. In the Repository name field, type **bitnami**. Next, click **Create fork**.
4. Click your user icon, and then select **Your repositories** from the list.
5. Two repositories should be displayed on your Repositories page: the test repository that you created and the bitnami repository that you forked.

Conclusion

Congratulations! You just took your first step toward learning GitHub. You created a GitHub account, and then you added your first repository. Next, you created and edited a new repository file. Finally, you forked an existing repository, and you can now modify this fork without altering the original repository.
You are now prepared to create and modify repositories on GitHub and explore other collaboration features that GitHub provides.

My Notes:

Lab Summary:

This lab introduced the foundational tasks for using GitHub and served as a prerequisite for future labs involving code vulnerability scanning (e.g. with Synk). While labeled optional, I completed the full lab sequence to ensure my environment was properly configured.

What I did:

- Skipped Task 1, as I already had a GitHub account.
- Started with Task 2, where I created a new repository named `test`. Although the instructions recommended making it private, I chose to create a public repository to support professional transparency and portfolio sharing.
- In task 3, I created a file (`helloworld.py`) directly in the GitHub interface. I added a short Python script "Hello World!" and committed the changes using the GitHub web editor.
- In task 4, I forked the Bitnami Containers repository (`bitnami/containers`) to my own GitHub account, enabling future exploration or modification of containerized application code without affecting the original source.

Why there are no Screenshots

All actions were performed within GitHub's web-based interface. No screenshots were needed or captured because the work can be publicly verified in my GitHub repositories. This approach keeps the documentation clean and audit-friendly, while allowing real-time validation of my contributions.