🛠️ GitHub Self-Hosted Runner Setup Guide (MoJ Project)

Assignment: `Week 9 ICE Prep` **Deliverable:** A screenshot of your "Idle" runner in the GitHub settings.

This guide will set up your local machine to act as a **CI Server (Runner)** for your team's `github.iu.edu` repository. This is a **mandatory, graded, individual assignment** that must be completed *before* Lecture 2.

The entire "CI/CR Fire Drill" workshop (ICE 2) will not work for your team if your runners are not configured and "Idle."

Step 0: Essential Pre-Flight Checklist

Before starting, ensure these two things are true:

1. **Python 3.10+:** You have Python 3.10 or newer installed and available in your system's `PATH`.
    - **Action:** Open your terminal and run `python3 --version`.
2. **Permissions:** You have a working terminal and permission to create directories in your user's home folder.

Step 1: Navigate to the Runner Configuration Page

1. Log in to `github.iu.edu`.
2. Navigate to your team's **private** repository (e.g., `FA25-SE1-TeamXX-moj`).
3. Go to the repository `Settings` tab.

❗ CRITICAL TROUBLESHOOTING: "I don't see the `Settings` tab!"

You have just hit our first real-world permissions problem! This is a perfect teachable moment.

- **The Problem:**
    1. In ICE 1, your `Repo Admin` added you as a **"Collaborator,"** which defaults to the **"Write"** role.
    2. This assignment requires you to add a **Self-Hosted Runner**.
    3. Only users with the **"Admin"** role can access the `Settings -> Actions -> Runners` page.
- **The Teachable Moment:** This is a classic "Role-Based Access Control" (RBAC) issue. We set up permissions for *writing code* ("Write") but forgot about *administering infrastructure* ("Admin").
- **THE FIX:**
    1. Contact your team's `Repo Admin` from ICE 1 *immediately*.
    2. Direct them to go to `Settings -> Collaborators`.
    3. Have them change your role from **"Write"** to **"Admin"**.

4. Once they have done this, refresh the page. You should now see the "Settings" tab and can proceed.

---

4. On the left-hand menu, click `Actions`, then `Runners`.
5. Click the green `New self-hosted runner` button.

Step 2: Create a Professional Runner Directory

We will create a scalable directory structure to manage this runner and any future runners you might create.

1. Open your terminal/command prompt.
2. Create a "parent" directory in your home folder to house *all* your runners:

```
mkdir -p ~/github-runners
```

3. Now, create a *project-specific* directory for your MoJ runner. **Use your exact repo name.**

```
mkdir -p ~/github-runners/FA25-SE1-TeamXX-moj
```

(Replace `XX` with your team number)

4. Navigate into your new project-specific runner directory. **All remaining steps will happen here.**

```
cd ~/github-runners/FA25-SE1-TeamXX-moj
```

Step 3: Configure and Start Your Runner

You will now see a page with instructions for `Download` and `Configure`. Follow them precisely.

1. **Select OS:** Choose the correct Operating System (macOS, Linux, Windows) and Architecture (e.g., `ARM64` for Apple Silicon Macs, `x64` for most other machines).

2. **Download:**

   - Copy and paste the `curl` command from the "Download" section to download the runner package *into your current directory*.

3. **Configure:**

   - **CRITICAL:** Copy the `config.sh ...` (or `config.cmd`) command from the "Configure" section on the GitHub page. This command includes your **unique, temporary registration token**.
   - Run this command in your terminal (e.g., `./config.sh --url ... --token ...`).
   - The script will ask you for a few settings:

        - **Enter the name of runner group to add this runner to:** Press **Enter** to accept the `default` group.
        - **Enter the name of runner:** `TeamXX-YourIUUsername` (e.g., `Team05-seiffert`).
        - **Enter any additional labels:** Press **Enter** to accept the defaults.
        - **Enter name of work folder:** Press **Enter** to accept the `_work` default.
    - After a moment, you should see `Settings Saved`.

4. **Start the Runner:**
    - In the same terminal (still inside `~/github-runners/FA25-SE1-TeamXX-moj`), run the script:

```
./run.sh
```

    - You should see the runner connect successfully, ending with:

```
√ Connected to GitHub
...Listening for Jobs
```

    - **Leave this terminal window open.** Your runner is now "Idle" and waiting for a job from ICE 2.

Step 4: Verify and Submit Your Evidence

This is the **Definition of Done** and the deliverable for your Canvas assignment.

1. With the `./run.sh` script still running in your terminal, go back to your browser.
2. Refresh the `Settings` $\rightarrow$ `Actions` $\rightarrow$ `Runners` page in your repository.
3. You should now see your named runner (e.g., `Team05-seiffert`) in the list with a **green "Idle" status dot** ✅ .
4. Take a single screenshot that **clearly shows** your runner with its **green "Idle" status**.
5. Submit this screenshot to the Canvas assignment **`Week 9 ICE Prep`**.

Example of a successful screenshot:

Step 5: Managing Your Runners (Going Forward)

- **To stop this runner:** Go to the terminal where `./run.sh` is running and press `Ctrl+C`. The runner will go "Offline" in GitHub.
- **To run this runner again (for ICE 2):** Open a terminal, `cd ~/github-runners/FA25-SE1-TeamXX-moj`, and run `./run.sh`.
- **To add a runner for another project:**
    1. Create a *new* directory (e.g., `mkdir ~/github-runners/new-project`).
    2. `cd` into that new directory.
    3. Go to the *new repo's* settings and follow this guide again from **Step 3**.

4. To run both at once, you will need **two separate terminal windows**, one for each runner.

Common Failures & Diagnostics

Symptom	Cause	Diagnostic Check
Runner is "Offline" (Red Dot)	The `./run.sh` script is not running, or your computer lost its connection.	**Check:** Is the `./run.sh` script still running in your terminal? If you re-run it, does the dot turn green?
Configuration Fails (Step 3)	The token expired (they only last 60 minutes).	**Check:** Refresh the "New self-hosted runner" page to get a **new token** and run the `./config.sh ...` command again.
`./config.sh: Permission denied`	The script is not executable (macOS/Linux).	**Check:** Run `chmod +x config.sh` and `chmod +x run.sh`, then try again.