Preparation for Lab 3

Lab 3 will be dealing wil IAM. IAM is a huge subject and can be complex. But we will be starting easily enough.

BEFORE we start with IAM, I want you to get ready to use your GitHub accounts.

Everyone still has their passwords to get into their GitHub? Did you set up MFA as well? Well, there's another secure way to interact with GitHub securely. SSH keys that you store in memory on your work machine or laptop to interact with the code versioning system.

Github is a lot of things as well as a code versioning tool. It's a document repository, it has pipelines for automated code reviews and supports complex deployment strategies. So it'll be expected that you will be using it for all your labs going forward. We will tie it out to your portfolios!

So read this over, assume bash commands are happening on your linux VM. But you can also use the same commands for Macs as well.

Ok, here's your guide to making your first (for many of you) GitHub commit, from preparing your project directory and walking you through creating an SSH-authenticated push and then a second commit to see the versioning magic.

Here’s the revised guide—including proper permissions, explicit steps to create a GitHub repo via the web UI, and a link to GitHub’s official quickstart tutorial.

---

1. Prepare your project directory & permissions

1. **Create and secure your folder**

```
mkdir my-first-repo
cd my-first-repo
chmod 755 .           # Owner can rwx, others can rx
```

2. **Add your project files**

```
echo "# My First Repo" > README.md
echo "print('Hello, world!')" > hello.py
```

---

2. Add a `.gitignore`

```
# Byte-compiled Python files
__pycache__/
*.py[cod]

# Local config / secrets
.env
config.yaml
secrets.json

# Editor & OS files
*.swp
.DS_Store
```

```
touch .gitignore
nano .gitignore    # Paste above, save
chmod 644 .gitignore   # Read/write owner, read others
```

---

3. Secrets / Token Management

- **Never commit** keys or passwords—add them to `.gitignore`.
- **Use environment variables**:

```
export API_TOKEN="your_real_token_here"
```

- **Consider secret-scanning** (e.g. GitHub's built-in scanner or `git-secrets`).

---

4. Generate, Secure & Load SSH Keys

1. **Generate a new key pair**. You can leave the passphrase question blank. Don't worry about doing that:

```
ssh-keygen -t ed25519 -C "you@example.com"
# Accept defaults: ~/.ssh/id_ed25519 and id_ed25519.pub
```

2. **Lock down permissions on your keys**:

```
chmod 700 ~/.ssh
chmod 600 ~/.ssh/id_ed25519
chmod 644 ~/.ssh/id_ed25519.pub
```

3. **Start the SSH agent** and load your private key:

```
eval "$(ssh-agent -s)"
ssh-add ~/.ssh/id_ed25519
```

4. **Copy & register** your public key on GitHub:

```
cat ~/.ssh/id_ed25519.pub
```

   - On GitHub.com: **Settings** → **SSH and GPG keys** → **New SSH key**, paste and save.

5. **Test your connection**:

```
ssh -T git@github.com
# Expect: "Hi username! You've successfully authenticated..."
```

---

5. Initialize Git & First Commit

1. **Initialize the repo**:

```
git init
```

2. **Stage files**:

```
git add README.md hello.py .gitignore
```

3. **Commit**:

```
git commit -m "Initial commit: project scaffold"
```

---

6. Create & Connect to a GitHub Repository

1. **On GitHub.com**, click **New** (green button) or go to https://github.com/new
2. **Fill in**:
   - **Repository name**: `my-first-repo`
   - **Description** (optional)
   - **Privacy**: Public or Private
   - **Do not** initialize with a README (you already have one)
3. **Click Create repository**
4. **Back in your terminal**, add the remote and push:

```
git remote add origin git@github.com:your-username/my-first-repo.git
git push -u origin master
```

---

7. Modify & Make a Second Commit

1. **Edit `hello.py`**:

```
nano hello.py
# Change to:
# print('Hello, GitHub!')
```

2. **Stage and commit**:

```
git add hello.py
git commit -m "Update greeting in hello.py"
git push
```

---

8. Further Reading

For more detail, see GitHub's official quickstart tutorial:
https://docs.github.com/en/get-started/quickstart/create-a-repo

---

You've now covered:

- Directory setup & correct permissions
- `.gitignore` best practices
- SSH key security & loading into memory
- Creating a GitHub repo via the web UI
- Make at least two clear commits with descriptive messages and save a screen shot to add to Lab 3 later.

Once that's done, GitHub will trust pushes over SSH from your VM/laptop. You can now `git push` without typing your username and password each time.

You're now comfortable with the core workflow: *prepare* → *track* → *commit* → *push* → *modify* → *commit again*. Welcome to GitHub!