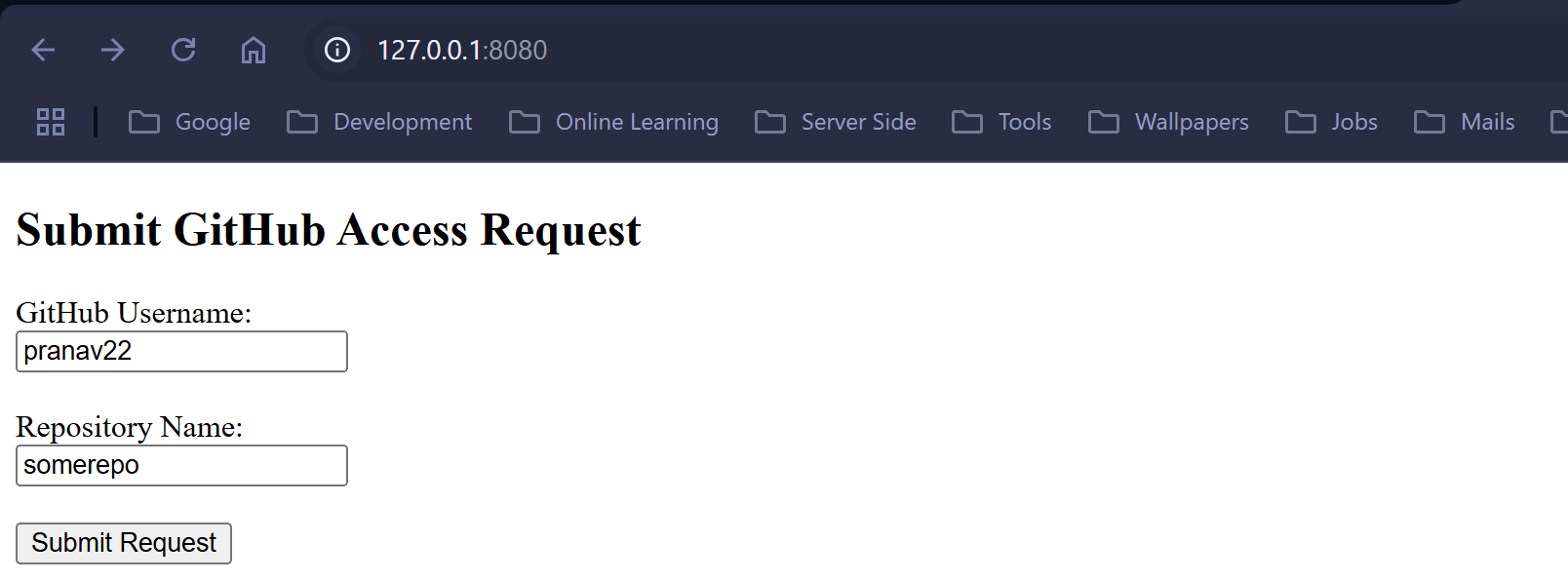
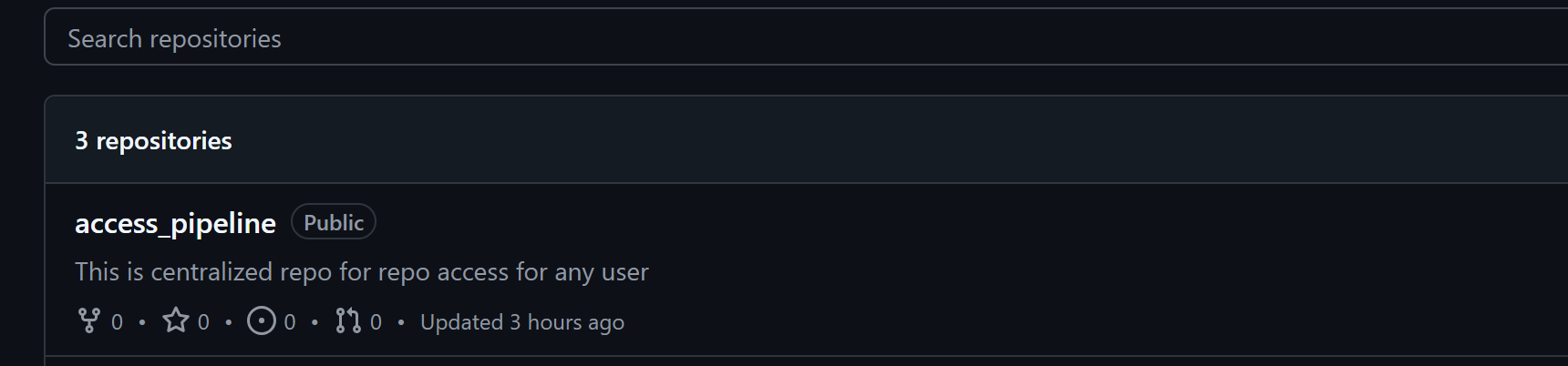
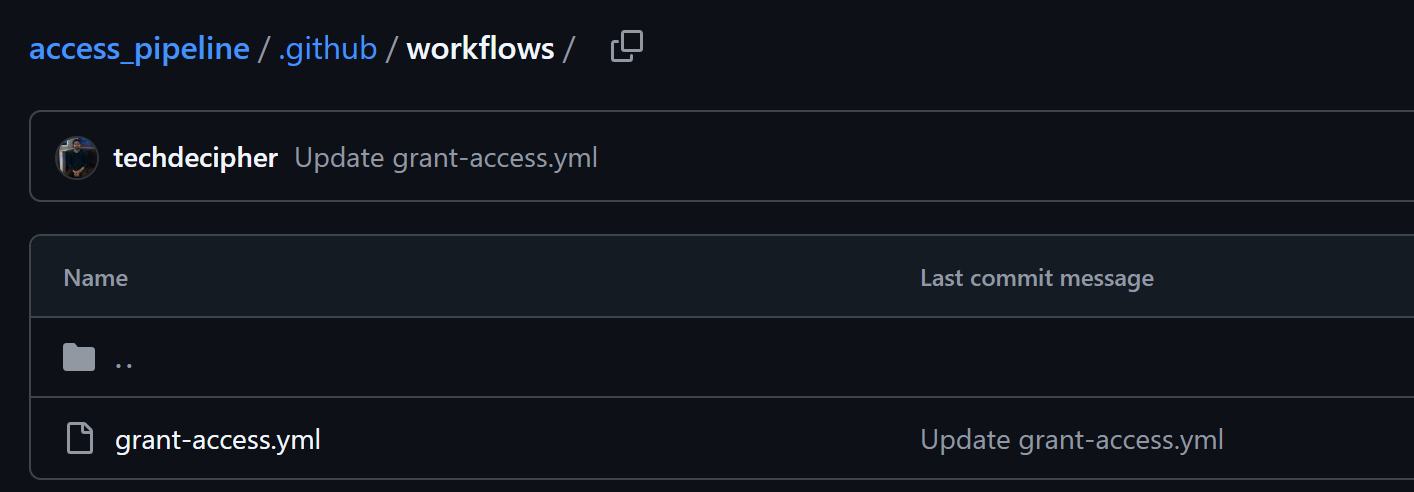
**POC for Github access automation**

**Step 1) Data Retrieval**  
Following example is just a mimic of an flask API which I is used to simulate how the required details are fed, retrieved and sent to trigger the github actions workflow. Note: This is just a mimic, the steps will differ depending on how the manager approval is done.****

**Step 2) Setup the repo, use case wise(granting repo access in this case)**



**Step 3) Setting up the workflow yaml in github actions in that repo**



A sample job yaml file: grant-access.yml

name: Grant GitHub Access

on:

workflow\_dispatch:

inputs:

username:

required: true

description: GitHub username to grant access

repo:

required: true

description: Repo name

jobs:

grant-access:

runs-on: ubuntu-latest

env:

GITHUB\_PAT: ${{ secrets.MY\_PAT }}

OWNER: my-org-tech-decipher

steps:

- name: Grant access using GitHub API

run: |

echo "Granting read access to ${{ github.event.inputs.username }} on ${{ github.event.inputs.repo }}"

curl -s -X PUT \

-H "Authorization: token $GITHUB\_PAT" \

-H "Accept: application/vnd.github+json" \

https://api.github.com/repos/$OWNER/${{ github.event.inputs.repo }}/collaborators/${{ github.event.inputs.username }} \

-d '{"permission":"pull"}'

**Step 4) Setup the secrets for the repo**

Going to Settings > Secrets and Variables > Secrets > Repository secrets  
 

And update the Secrete in the step 3 script grant-access.yml

**Step 5) Execution status**

Once the info is submitted from the step 1 which is a mimic phase, github actions will run, and provide access to the specific repo.

