Personal Blog Setup and Self-host using Hugo and Github Actions. 1

1. Irrespective of OS or propretairy software users, To cut short and jump stright in to the IDE.
2. For that, we will use Gitpod. few lines about Gitpod- all can use free tier and no credit card required.
3. check if git installed on the gitpod and rutn the below commands on the terminal

```
echo "# fosschennai_June2023" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/Sakh1l/fosschennai_June2023.git
git push -u origin main
```

4. Now, on the Readme file, add some text and commit the changes. once done, you can able to view the changes in the repository.
5. create the site with hugo command hugo new site my_blog
   move inside the folder my_blog and git clone
   git clone https://github.com/vaga/hugo-theme-m10c.git themes/m10c
6. and edit config.yml file with theme: "m10c"
7. Now run the hugo server to check the page in localhost
8. add "gh-pages" brach for the github actions
9. move to settings and under "Actions " tab on the left side select "general" tab and select "read and write permisson" and save the settings.

Github Actions CI/CD

```
git submodule add --depth=1 https://github.com/vaga/hugo-theme-m10c.git
themes/m10c
echo "# Test Read" >> README.md
git add README.md
git commit -m "first commit"
git branch -M main
sudo nano config.yml
git remote add origin https://github.com/Sakh1l/fosschennai_June2023
git push -u origin main
```

mkdir -p .github/workflows
sudo nano .github/workflows/deploy.yml
git commit -m "add the first test page"
git push

CI/CD work flow config

Create a repository on Github, create a readme file, add the remote address, and push your first commit

Manually Add the gh-pages Branch

- Manually add the gh-pages branch to the repository; otherwise the github actions will throw an error

Allow Read and Write Permissions on the Workflow

- Allow read and write permissions under Settings > Actions > General > Workflow permissions

Add a .github/workflows/deploy.yml file under the project root directory

```yaml
name: Publish to GH Pages
on:
  push:
    branches:
      - main
  pull_request:

jobs:
  deploy:
    runs-on: ubuntu-latest
    steps:
      - name: Checkout source
        uses: actions/checkout@v3
        with:
          submodules: true

      - name: Checkout destination
        uses: actions/checkout@v3
        if: github.ref == 'refs/heads/main'
        with:
          ref: gh-pages
```

```
          path: built-site

      - name: Setup Hugo
        run: |
          curl -L -o /tmp/hugo.tar.gz
'https://github.com/gohugoio/hugo/releases/download/v0.110.0/hugo_extended_0
.110.0_linux-amd64.tar.gz'
          tar -C ${RUNNER_TEMP} -zxvf /tmp/hugo.tar.gz hugo
      - name: Build
        run: ${RUNNER_TEMP}/hugo

      - name: Deploy
        if: github.ref == 'refs/heads/main'
        run: |
          cp -R public/* ${GITHUB_WORKSPACE}/built-site/
          cd ${GITHUB_WORKSPACE}/built-site
          git add .
          git config user.name 'githubusername'
          git config user.email 'youremailid@gmail.com'
          git commit -m 'Updated site'
          git push
```

- The first step checks out my repository under `$GITHUB_WORKSPACE` and `submodules:true` ensures that our submodule for the theme repository is fetched as well
- The second step allows us to reference the `gh-pages` branch via the `$GITHUB_WORKSPACE/built-site` directory, where our static sites will be stored in (Refer to the `Deploy` step)
- The third and fourth steps involve installing hugo and building the static pages in the `public` directory with the `hugo` command
- The last step copies the static sites into `${GITHUB_WORKSPACE}/built-site` and pushes the changes to the referenced branch `gh-pages`, which is a special branch that Github recognizes and uses to publish to your Github Pages site

Note: the content will be deployed to `https://<username>.github.io/<repository_name>/` by default if not configured otherwise. Update the `base_url` in config.yml to `"https://<username>.github.io/<repository_name>/"`

Link Custom Domain to Github Pages

-
- Add your custom domain under your Git repository's Settings > Pages > Custom Domain as shown in the image above.

Note: the DNS check will initially be unsuccessful

```
baseurl: "http://sakhil.in"
```

- Update the baseurl in config.yml with your domain. Please note that the HTTPS is enforced through Github Pages

Once the commit for the baseurl change is pushed, the Github Action will run the deploy job and you should be able to access the site via your custom domain. It might take some time for your DNS provider to update your domain and the newly-created records to take effect.