CATALOG TEST IN WINDOWS 10
Fernando Millán Villalobos

Contents

1. NOTES

This document is based on the instructions provided in below site:
https://github.com/wiederkehr/catalog-deployment-example-githubpages

Some parts have been enhanced with other steps which are not described in the above-mentioned document. Additional links are provided in each section.

Important notes

- GitHub pages which will require using the /docs in master branch must belong to an organization. This has an impact on the type of repository I create (organization).

- In order to configure the GitHub pages to use the /docs in master branch, the /docs folder must already exist. Otherwise it is not possible to do the configuration.
  https://help.github.com/en/github/working-with-github-pages/configuring-a-publishing-source-for-your-github-pages-site

  https://help.github.com/en/enterprise/2.14/user/articles/configuring-a-publishing-source-for-github-pages

  > **Tip:** The **master branch /docs folder** source setting **will not** appear as an option if the `/docs` folder doesn't exist on the `master` branch.

- To do the catalogue build, the package.json might need to be adapted to allow having the output to a specific output folder (docs) of a specific public-url (GitHub repository)
  https://github.com/interactivethings/catalog/issues/405

2. INSTALLATION OF NODE.JS and NPM

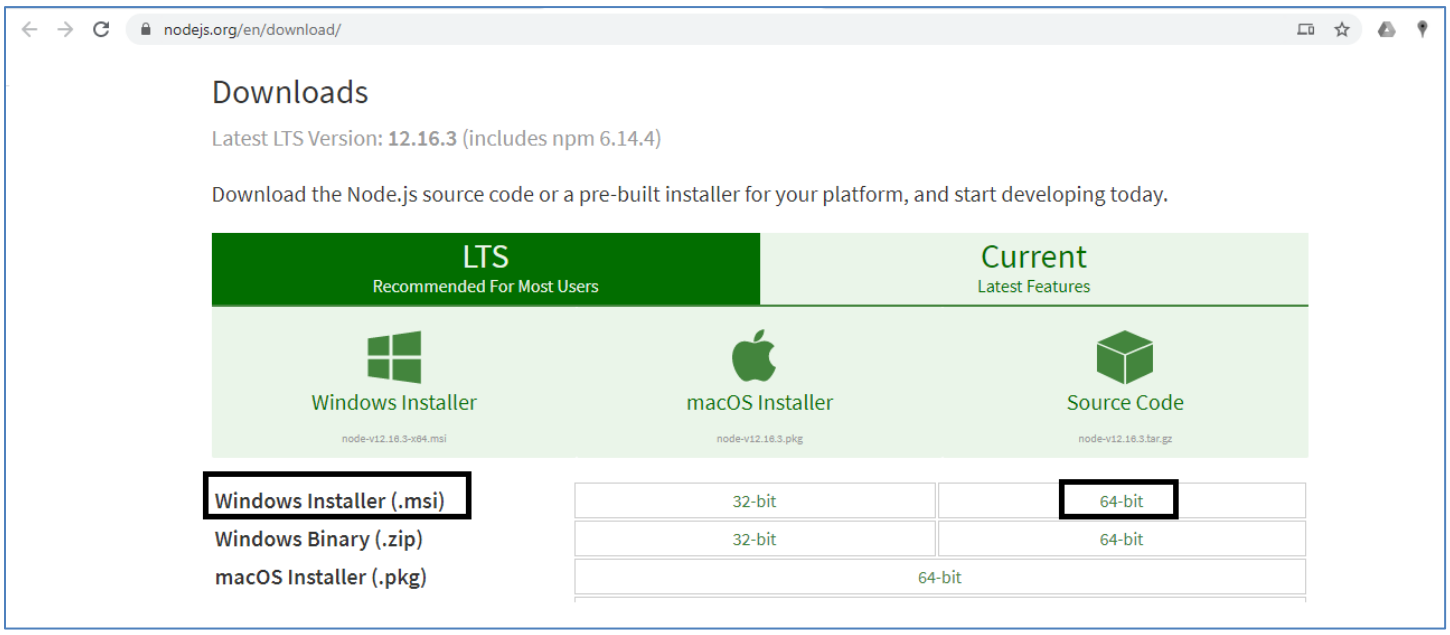
Please refer to https://docs.npmjs.com/downloading-and-installing-node-js-and-npm

Since the setup is done in a Windows machine, I follow the below instructions.

> **OS X or Windows Node installers**
>
> If you're using OS X or Windows, use one of the installers from the Node.js download page. Be sure to install the version labeled **LTS**. Other versions have not yet been tested with npm.

The link is https://nodejs.org/en/download/

I download the LTS version Windows installer (.msi) for 64-bit (after confirming in System settings of my machine that this is the bit version):

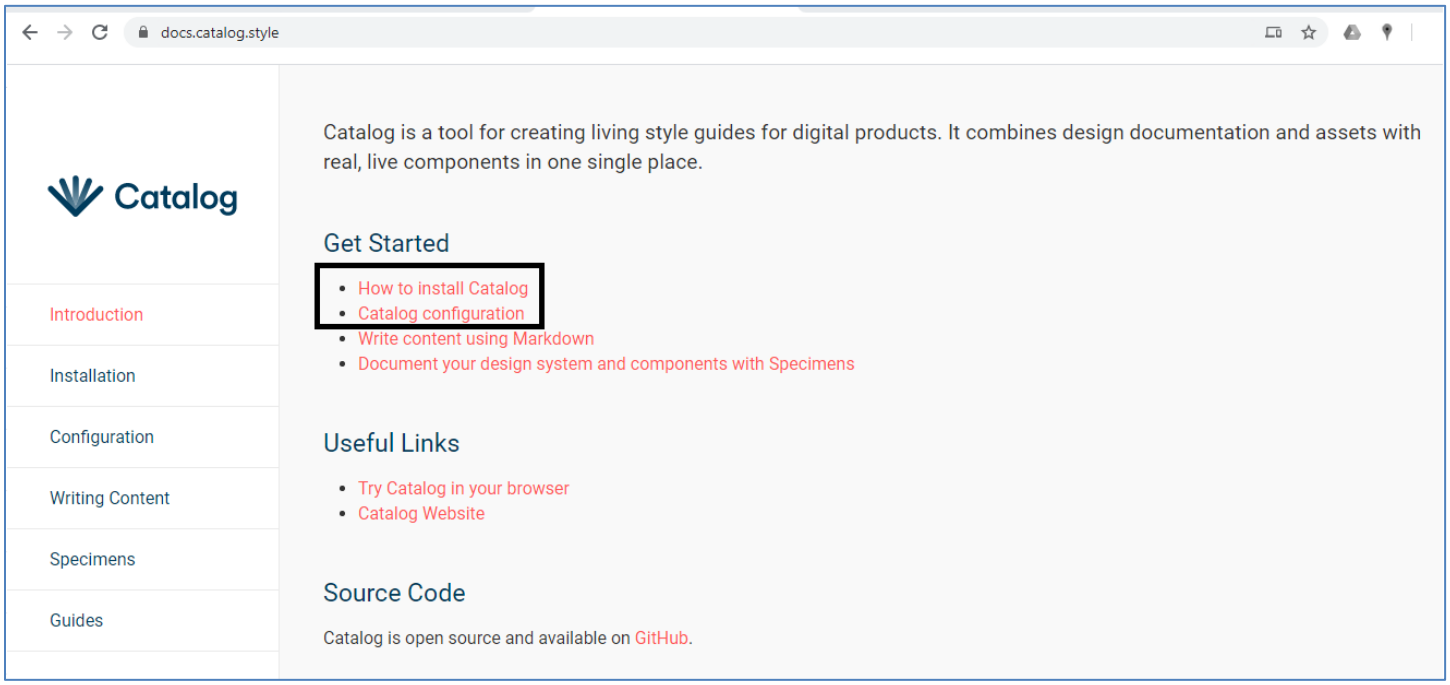


Once the .msi file is downloaded locally to the machine, I run it and let execute till everything is installed.

3. INSTALLATION OF CATALOG

Refer to https://www.catalog.style/
Section docs: https://docs.catalog.style/

How to install Catalog:

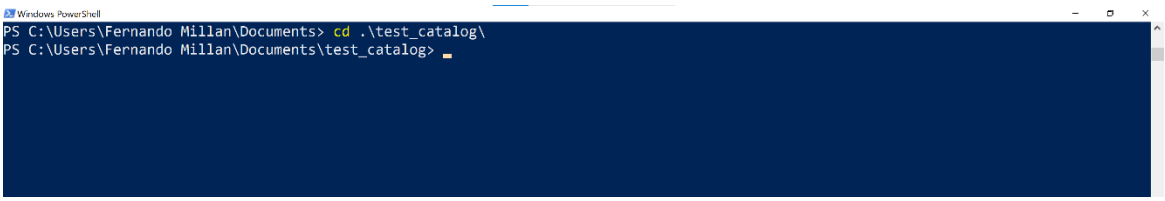


In PowerShell I execute the install command:

```
npm install -g create-catalog
```

4. CREATION OF A CATALOG AND RUNNING IT LOCALLY

In PowerShell navigate to the local directory where the catalog will be created.
Example: C:\Users\Fernando Millan\Documents\test_catalog

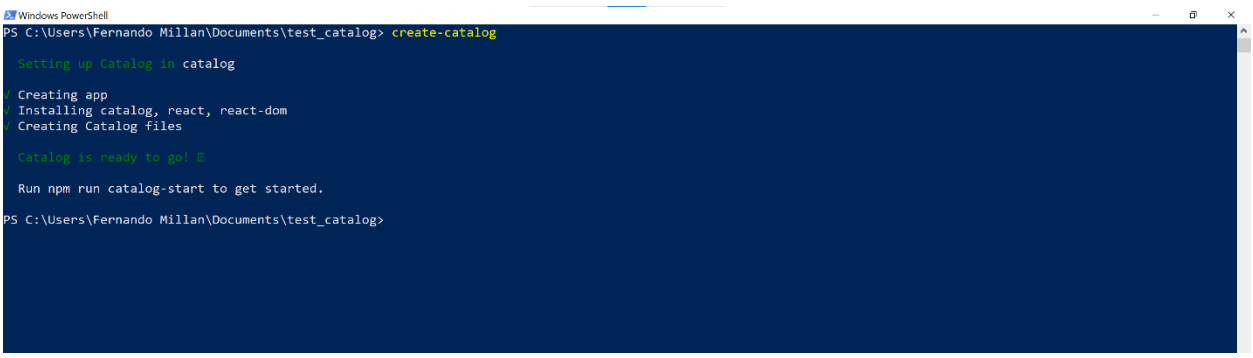


Then run the command

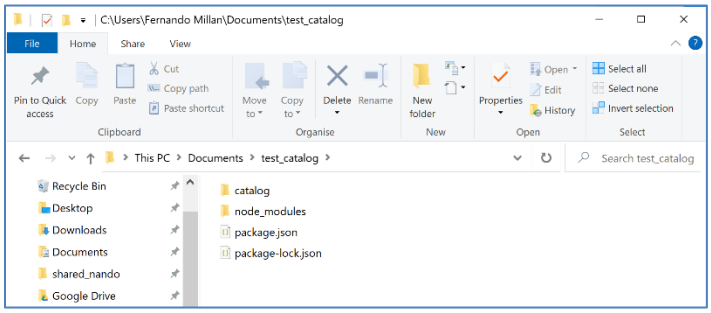
```
create-catalog <directory>
```

In our case

```
create-catalog C:\Users\Fernando Millan\Documents\test_catalog
```



Once the catalog is created, we can go to the local folder:

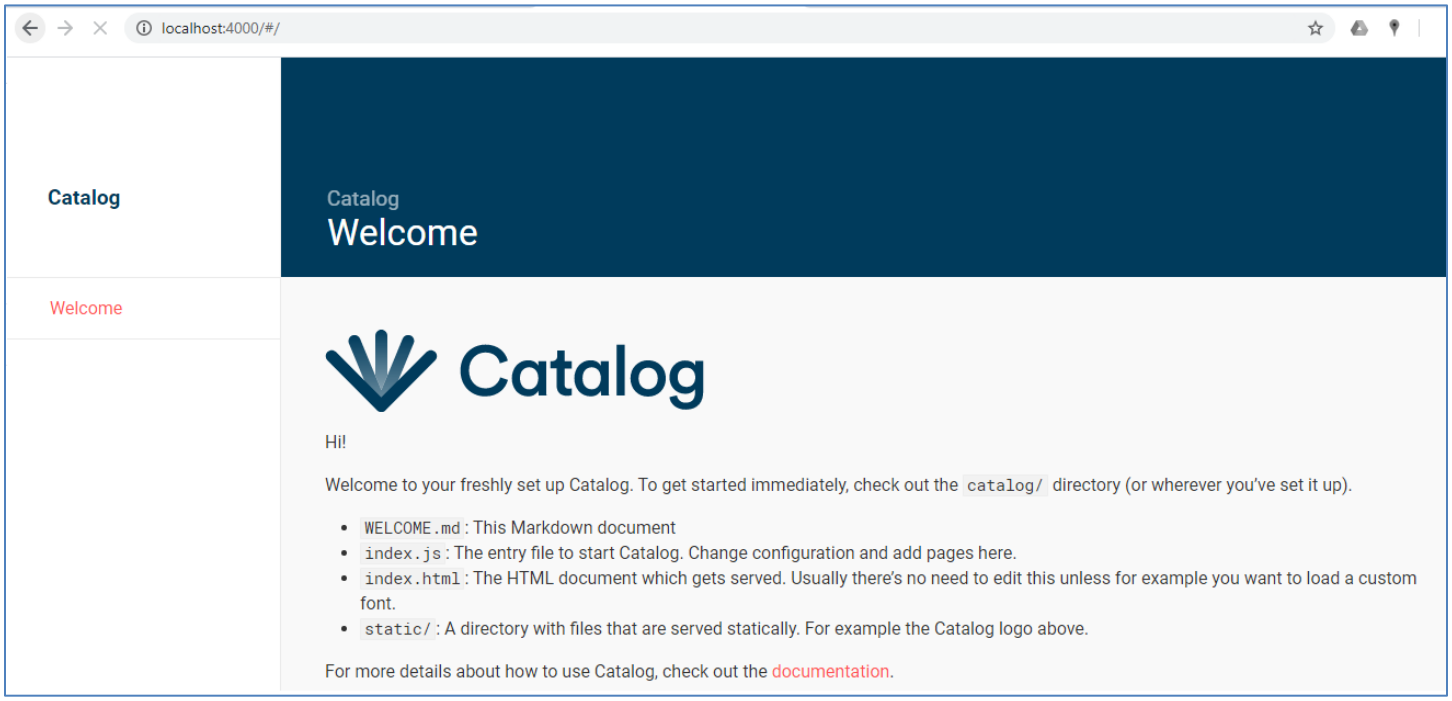


To run the catalog locally, execute the following command in PowerShell from the catalog folder:

```
npm run catalog-start
```

```
npm
DONE  Compiled successfully in 3852ms

I  Catalog is running at http://localhost:4000/
```

As long as the catalog is running in PowerShell, it will be visible locally in http://localhost:4000/



To stop do Ctrl-C and confirm:

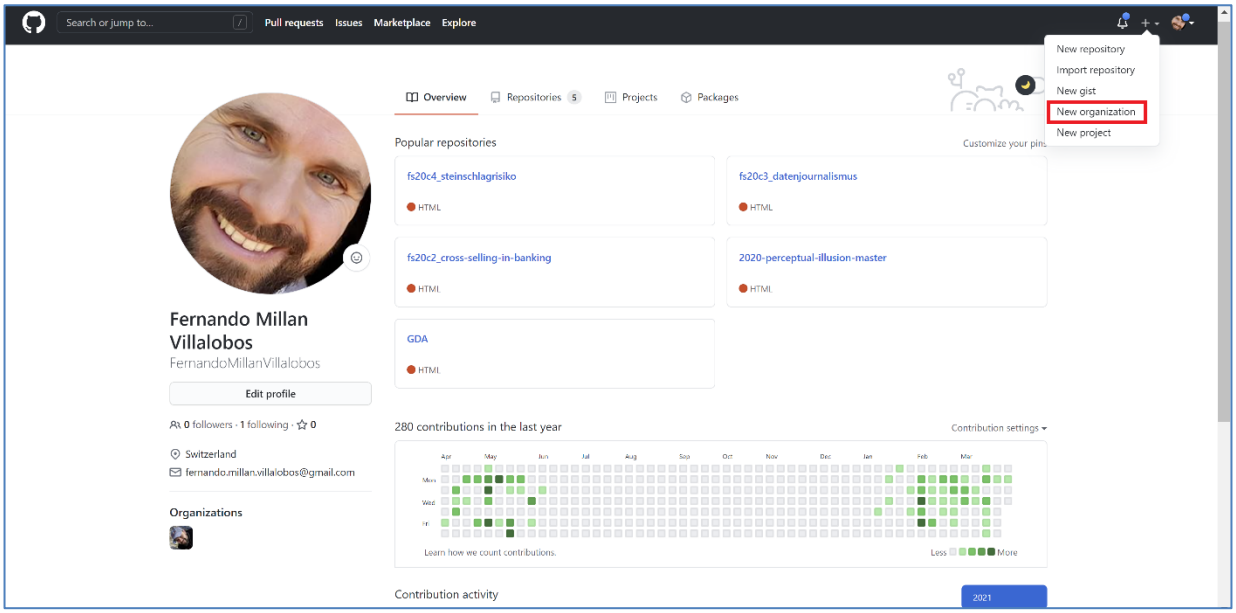
```
npm
DONE  Compiled successfully in 3852ms

I  Catalog is running at http://localhost:4000/
Terminate batch job (Y/N)?
```

5. CREATE NEW REPOSITORY IN GITHUB FOR AN ORGANIZATION

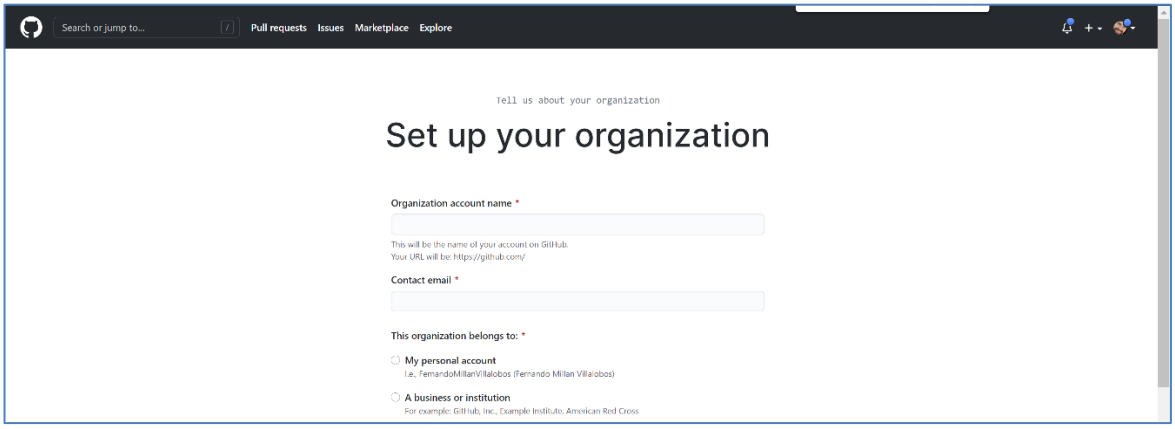
At a later stage we will need to use GitHub Pages and configure the same to synchronize from the master/doc folder. This is only possible for repositories that belong to an organization.
Hence when creating the repository in GitHub we need to create a repository for an organization.

a. Create new organization in GitHub

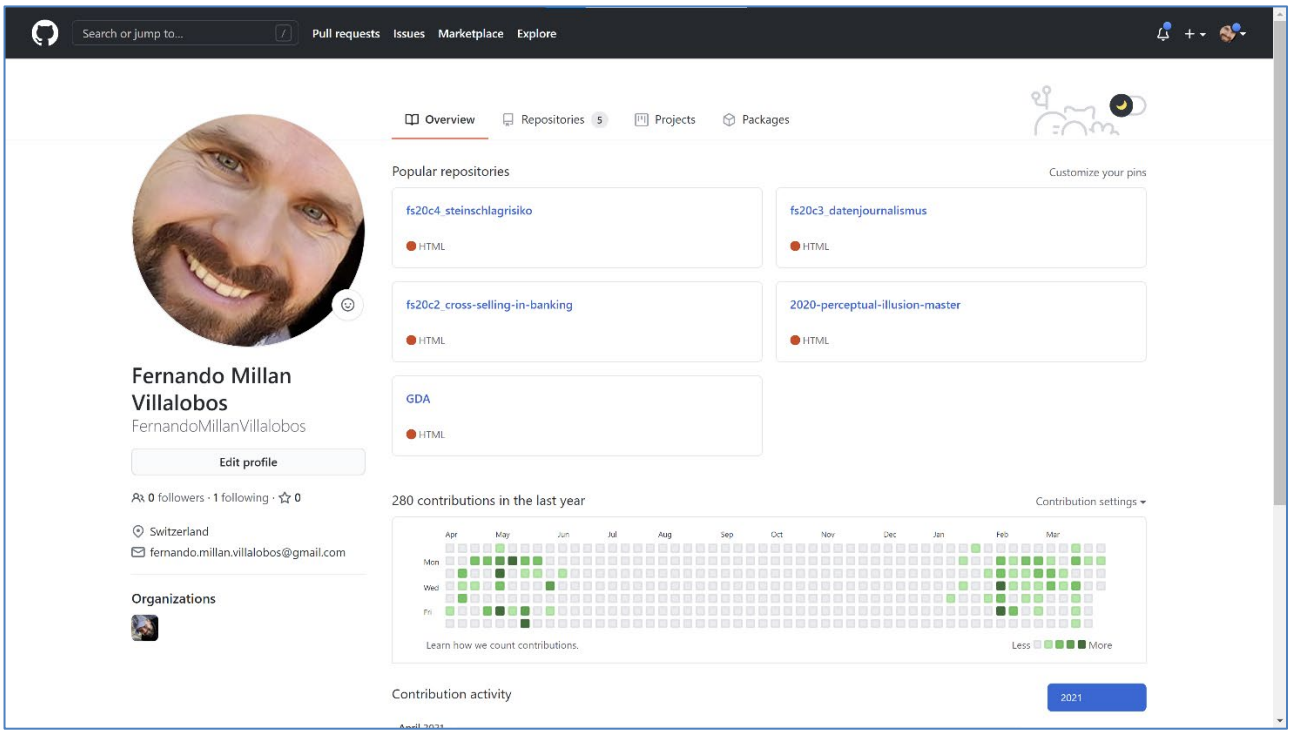
If your GitHub profile already has an organization, skip this step
Click on “New organization”



In the next window, select Free plan and then below page to setup the organization will appear:

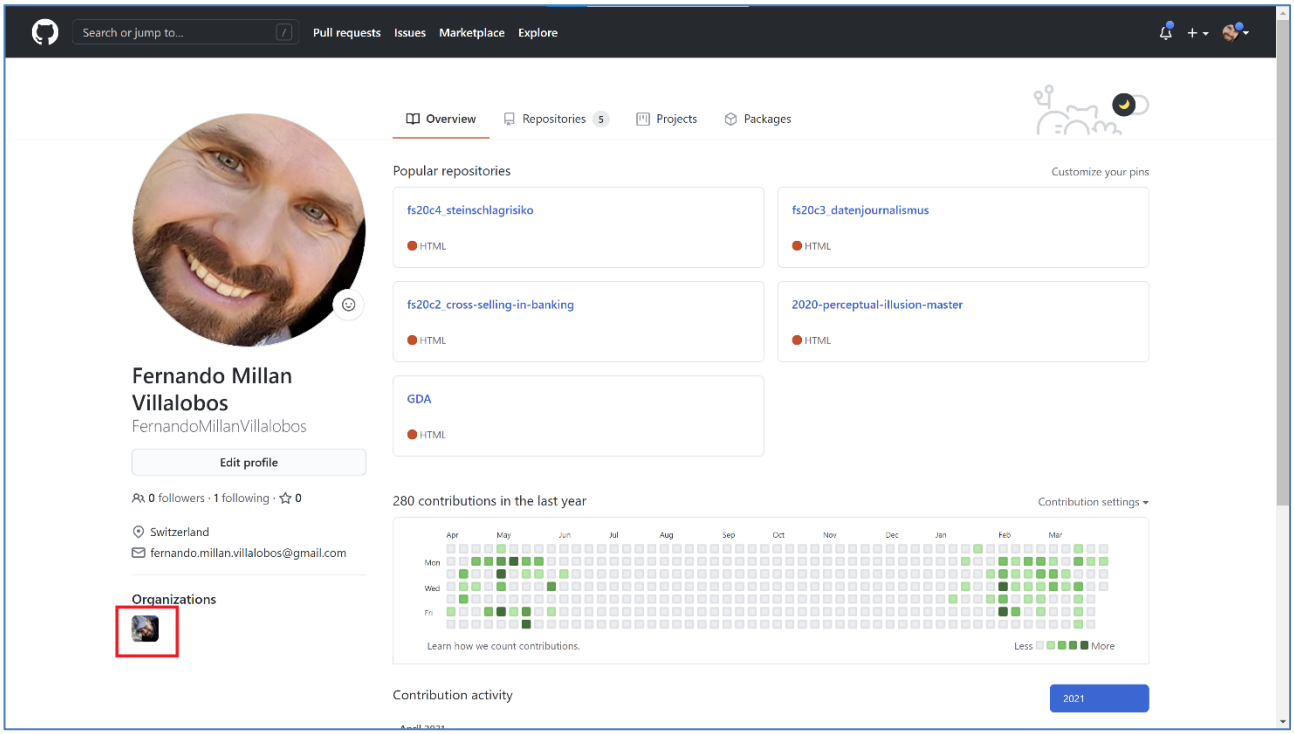


Provide the required data and complete the process.
Afterwards the new organization will be listed under “Organizations” in your GitHub profile:

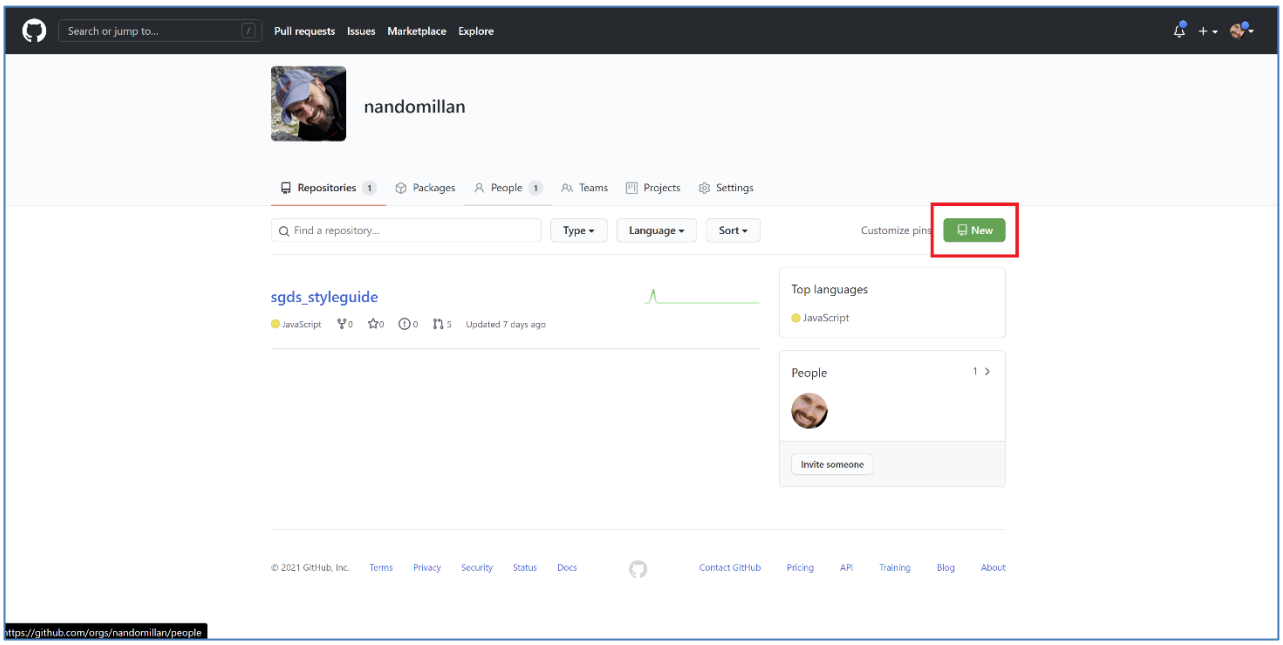


b. Create new repository for your organization

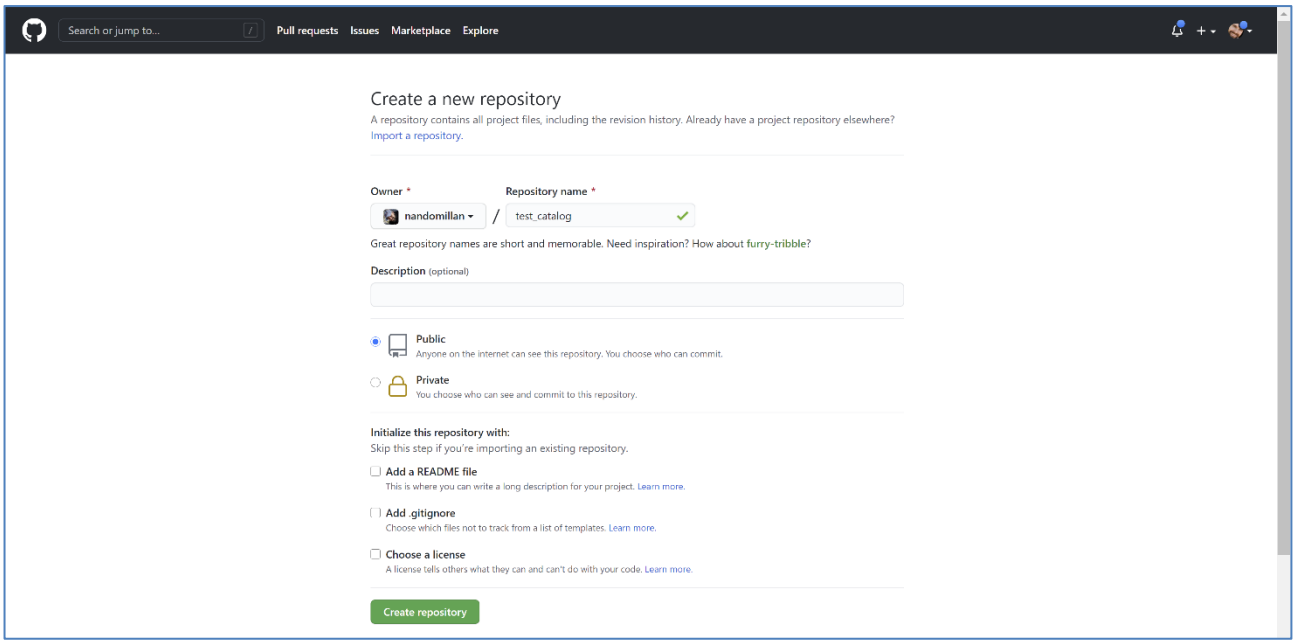
Navigate to your organization by clicking on its icon:



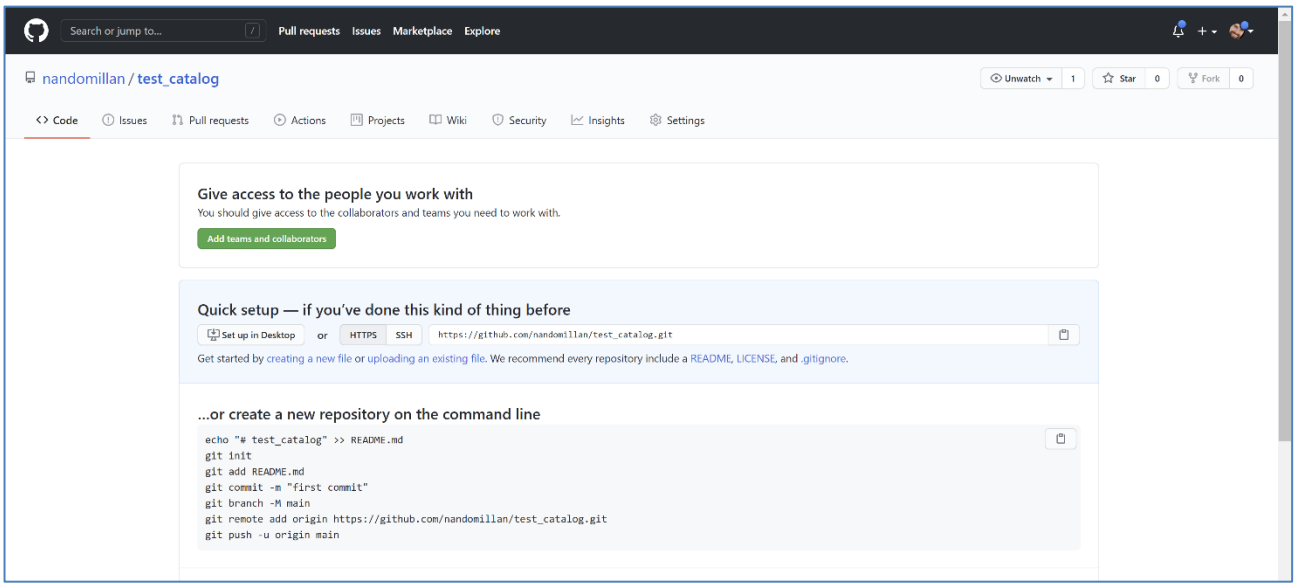
Then create a new repository:



Provide a name and create: test_catalog:

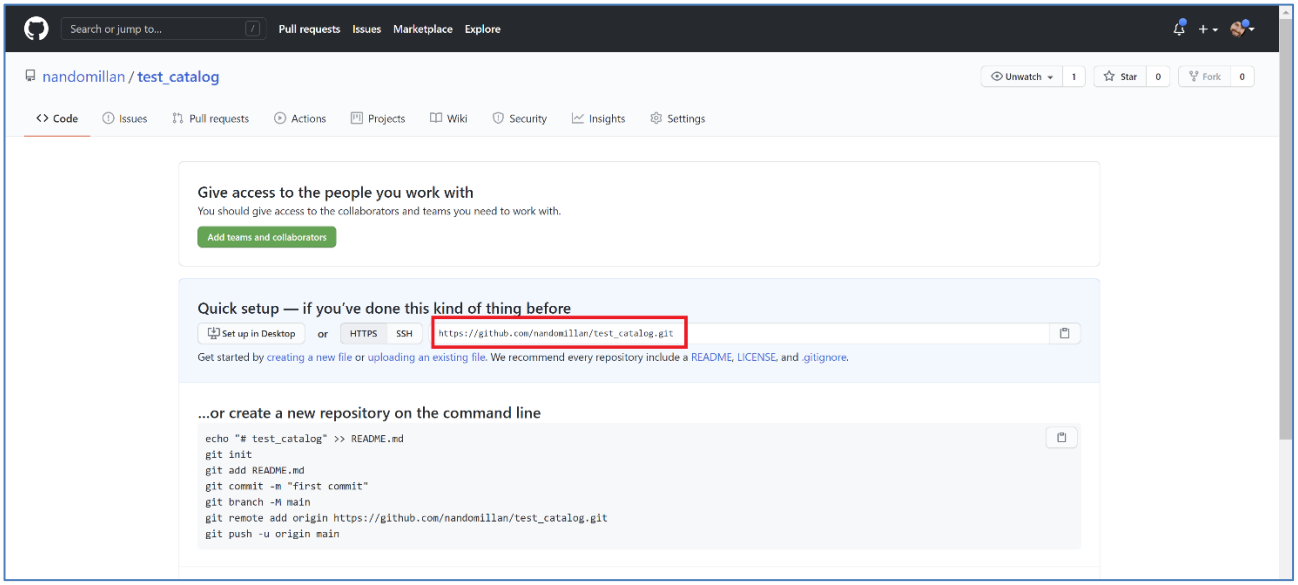


The new repository will be ready:



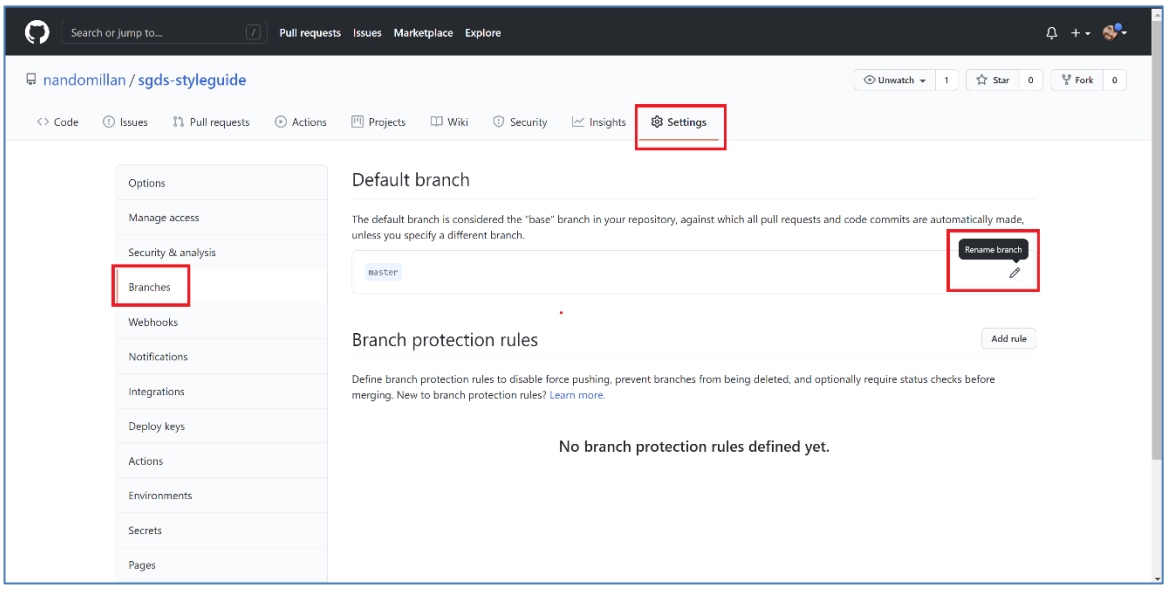
c. Copy the remote_repository_URL of your new repository

Copy the https link of your newly created repository: https://github.com/nandomillan/test_catalog



d. Change the name of your main branch to master

To avoid any confusion later, change the name of your main branch to master:

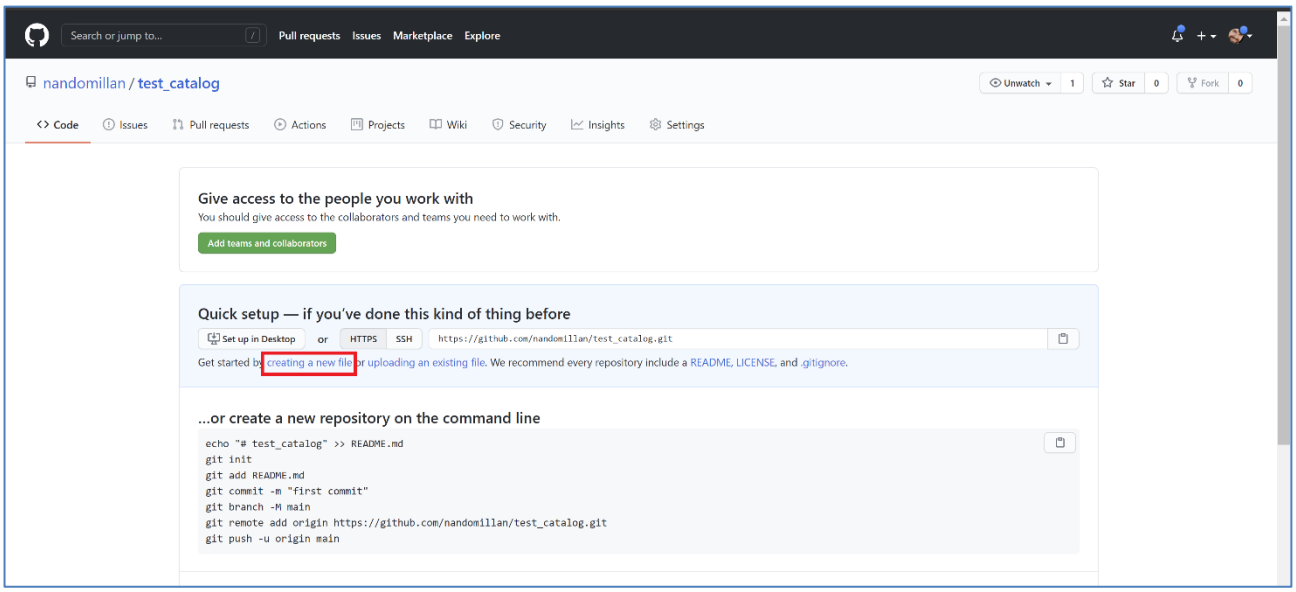


e. Create a docs folder in the new repository

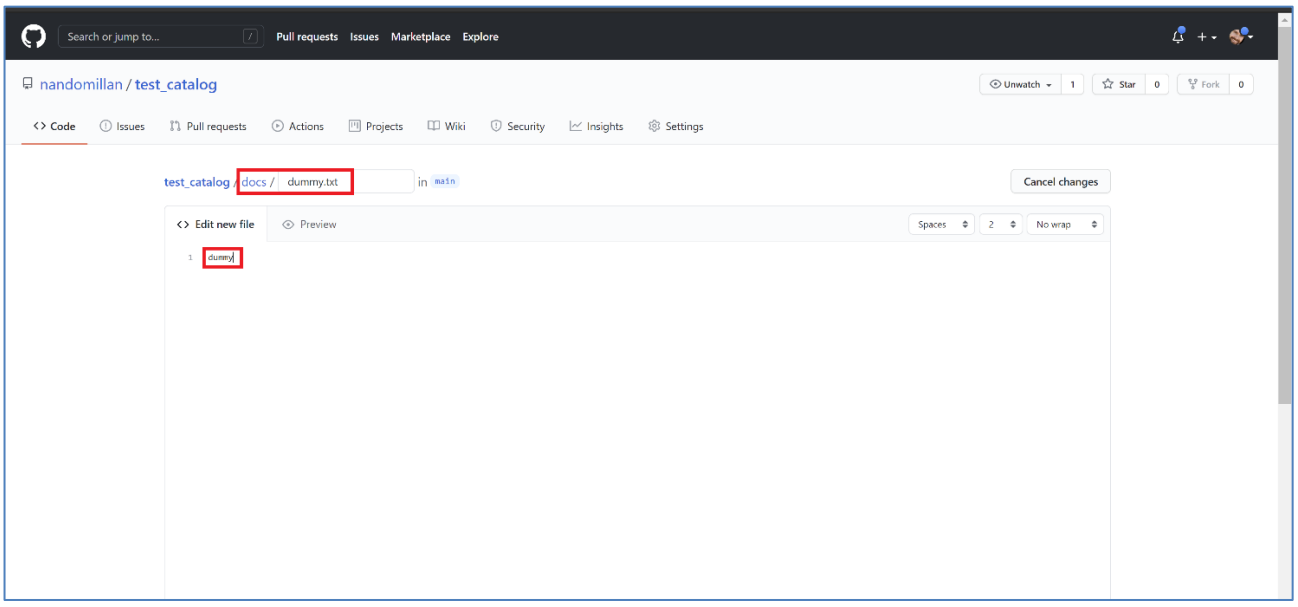
As we will see later, we will need to configure our GitHub Page to have as source the docs folder under master branch (refer to **CONFIGURE GITHUB PAGES**).

In order to do this, it is mandatory that the docs folder exists. A repository can't be empty, hence we will create a dummy file inside the folder.

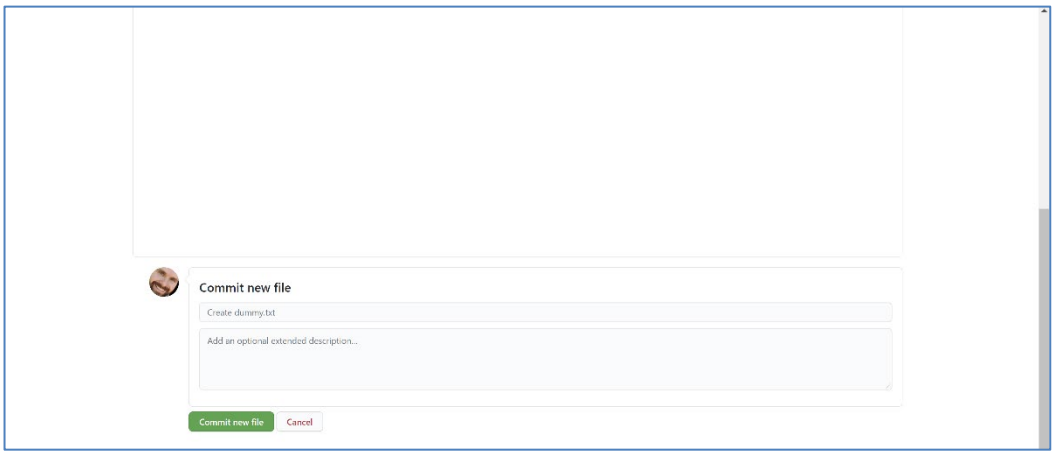
Click on “creating a new file”:



Enter **docs** and then “**/**”
This will create the repository (test_catalog/**docs**)
Then create the dummy file (test_catalog/docs/**dummy.txt**):



Commit the new file:



ADD YOUR CATALOG CODE TO YOUR GITHUB REPOSITORY

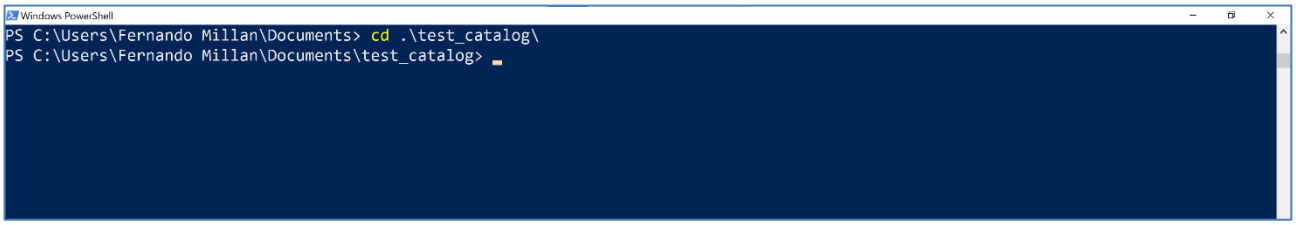
Below actions must be performed from the command line:

At this points we have following status

- New folder docs and document dummy.txt which exist in the GitHub repository need to be pulled to the local catalog repository.
- Documents in the local folder (including the adapted package.json) need to be pushed to the GitHub repository.

f. Change the current working directory to your local catalog folder

```
C:\Users\Fernando Millan\Documents\test_catalog
```



g. Initialize the local directory as a Git repository: git init

```
Windows PowerShell
PS C:\Users\Fernando Millan\Documents\test_catalog> git init
Reinitialized existing Git repository in C:/Users/Fernando Millan/Documents/test_catalog/.git/
PS C:\Users\Fernando Millan\Documents\test_catalog>
```

h. Set the new remote git repository: git remote add origin remote_repository_URL

Replace remote_repository_URL with your own (refer to **Copy the remote_repository_URL of your new repository**)

```
git remote add origin https://github.com/nandomillan/test_catalog.git
```

```
Windows PowerShell
PS C:\Users\Fernando Millan\Documents\test_catalog> git remote add origin https://github.com/FernandoMillanVillalobos/test_catalog.git
```

To verify if this is done properly, enter command:

```
git remote -v
```

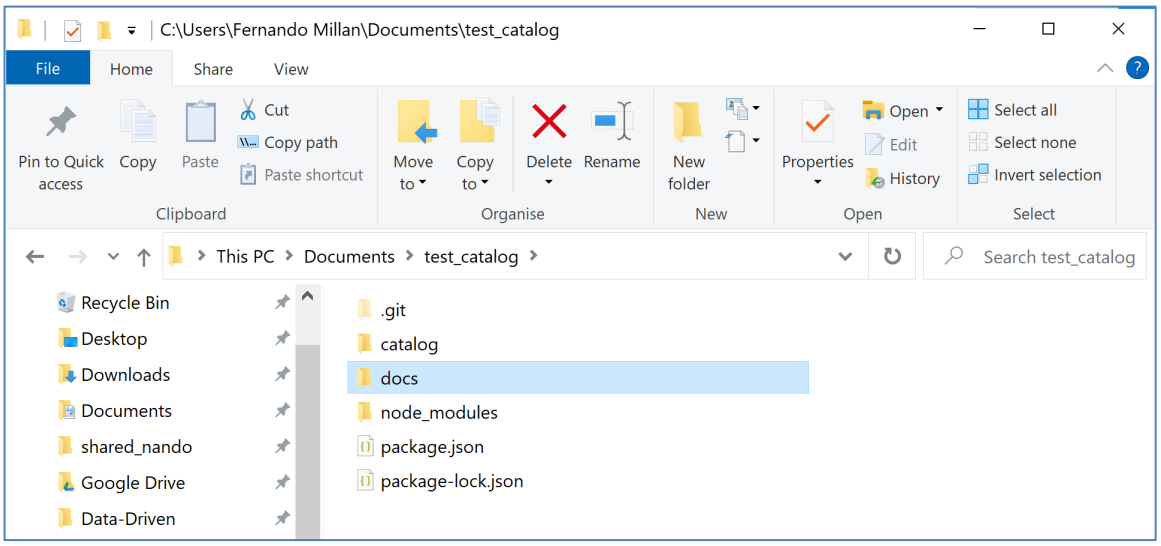
```
Windows PowerShell
PS C:\Users\Fernando Millan\Documents\test_catalog> git remote -v
origin  https://github.com/nandomillan/test_catalog.git (fetch)
origin  https://github.com/nandomillan/test_catalog.git (push)
PS C:\Users\Fernando Millan\Documents\test_catalog>
```

i. Pull the changes in GitHub to your local repository: git pull origin master

We do first a pull to. This will create the document docs/dummy.txt in your local repository:

```
Windows PowerShell
PS C:\Users\Fernando Millan\Documents\test_catalog> git pull origin main
From https://github.com/nandomillan/test_catalog
 * branch            main       -> FETCH_HEAD
Already up to date.
PS C:\Users\Fernando Millan\Documents\test_catalog> ls


    Directory: C:\Users\Fernando Millan\Documents\test_catalog


Mode                LastWriteTime         Length Name
----                -------------         ------ ----
d-----          4/7/2021   9:39 AM                catalog
d-----          4/7/2021   2:36 PM                docs
d-----          4/7/2021   9:43 AM                node_modules
-a----          4/7/2021   9:39 AM         384029 package-lock.json
-a----          4/7/2021   9:39 AM            246 package.json


PS C:\Users\Fernando Millan\Documents\test_catalog>
```

Evidence that the docs folder has been created:

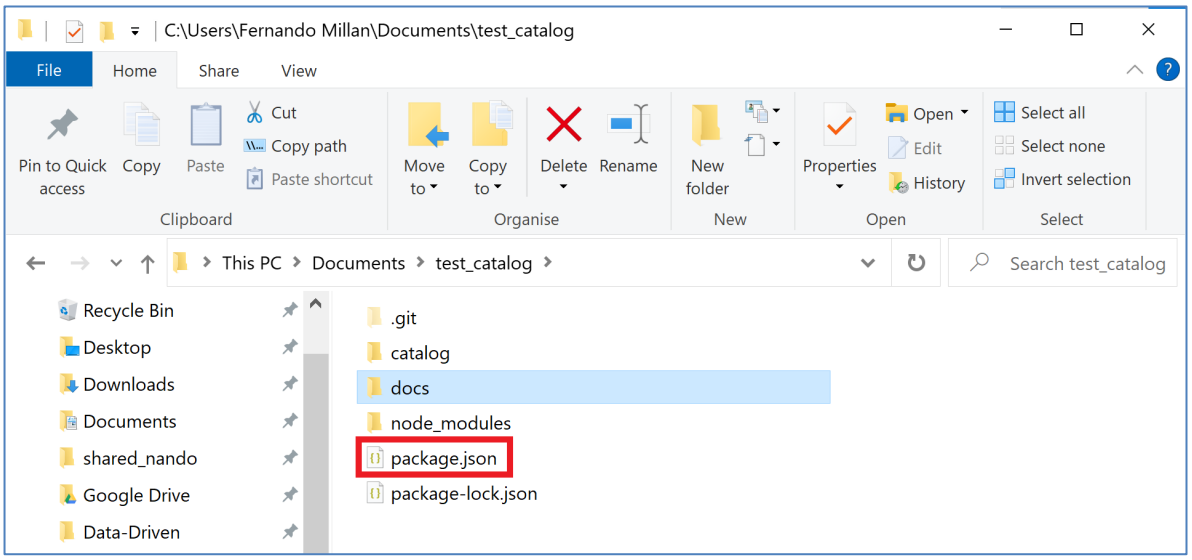


j. Edit the package.json to set it up for the build

This information was obtained from https://github.com/interactivethings/catalog/issues/405

Before we add the catalog to the git repository, we need to ensure that the package JSON is properly set up to do the build as per our needs (refer to **CONFIGURE GITHUB PAGES** and **PUBLISH TO GITHUB PAGES**).
When we do the build, we expect that this is done to the docs folder within our GitHub repository.

Open the package.json from the local folder containing your catalog:



Verify how the build will be performed:

```
{
  "name": "test_catalog",
  "description": "",
  "dependencies": {
    "catalog": "^3.6.0",
    "react": "^17.0.2",
    "react-dom": "^17.0.2"
  },
  "scripts": {
    "catalog-start": "catalog start",
    "catalog-build": "catalog build"
  }
}
```

We need to do the build to a specific output folder (docs) of a specific public-url (our GitHub repository)
Adapt that line of code to the following:
"catalog-build": "catalog build --out docs/ --public-url /XXXXX"
Where XXXXX is the name of the GitHub repository

In our case:
"catalog-build": "catalog build --out docs/ --public-url /Test_catalog_repo"

```
{
  "name": "test_catalog",
  "description": "",
  "dependencies": {
    "catalog": "^3.6.0",
    "react": "^17.0.2",
    "react-dom": "^17.0.2"
  },
  "scripts": {
    "catalog-start": "catalog start",
    "catalog-build": "catalog build --out docs/ --public-url /test_catalog"
  }
}
```

k. Add the files in your new local repository: git add .

```
PS C:\Users\Fernando Millan\Documents\test_catalog> git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   package.json

no changes added to commit (use "git add" and/or "git commit -a")
PS C:\Users\Fernando Millan\Documents\test_catalog> git add .
warning: LF will be replaced by CRLF in package.json.
The file will have its original line endings in your working directory
PS C:\Users\Fernando Millan\Documents\test_catalog> git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   package.json

PS C:\Users\Fernando Millan\Documents\test_catalog>
```

The first time it will take some time as there are many files to be added.

l. Commit the files that you've staged in your local repository: git commit -m "Commit comments"

```
PS C:\Users\Fernando Millan\Documents\test_catalog> git commit -m "First commit"
[master ac440ef8] First commit
 1 file changed, 2 insertions(+), 2 deletions(-)
PS C:\Users\Fernando Millan\Documents\test_catalog> git status
On branch master
nothing to commit, working tree clean
PS C:\Users\Fernando Millan\Documents\test_catalog>
```

m. Push the changes in your local repository to GitHub: git push origin main

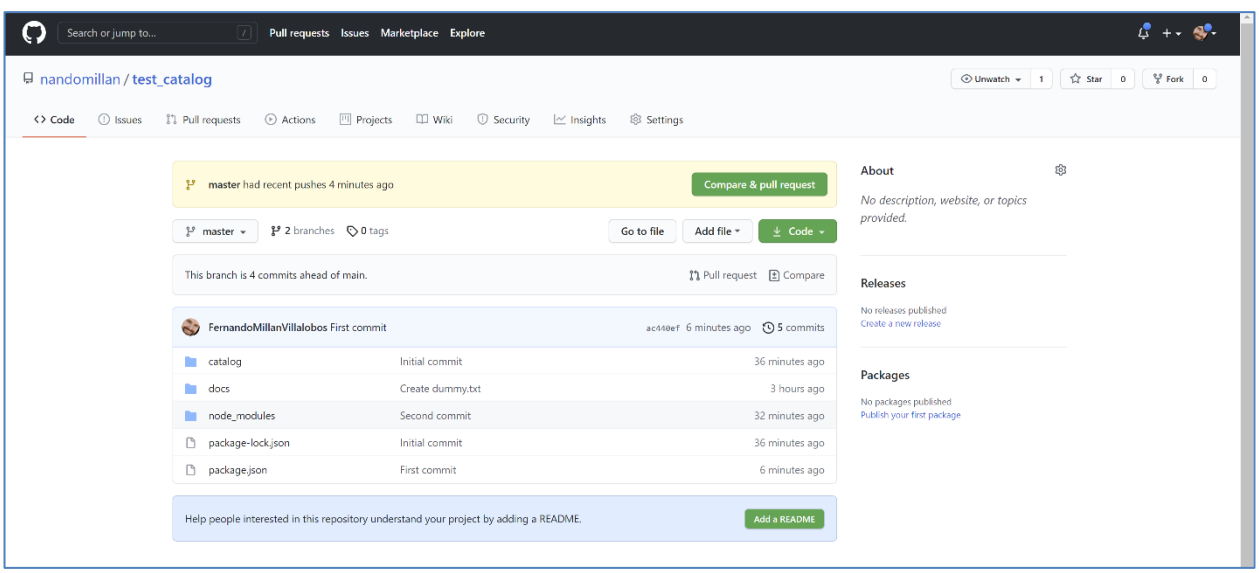
This will push all content of local repository to GitHub, including the new edited package.json

Commit the files that you've staged in your local repository: git commit -m "Commit comments"

Push the changes with git push origin master:

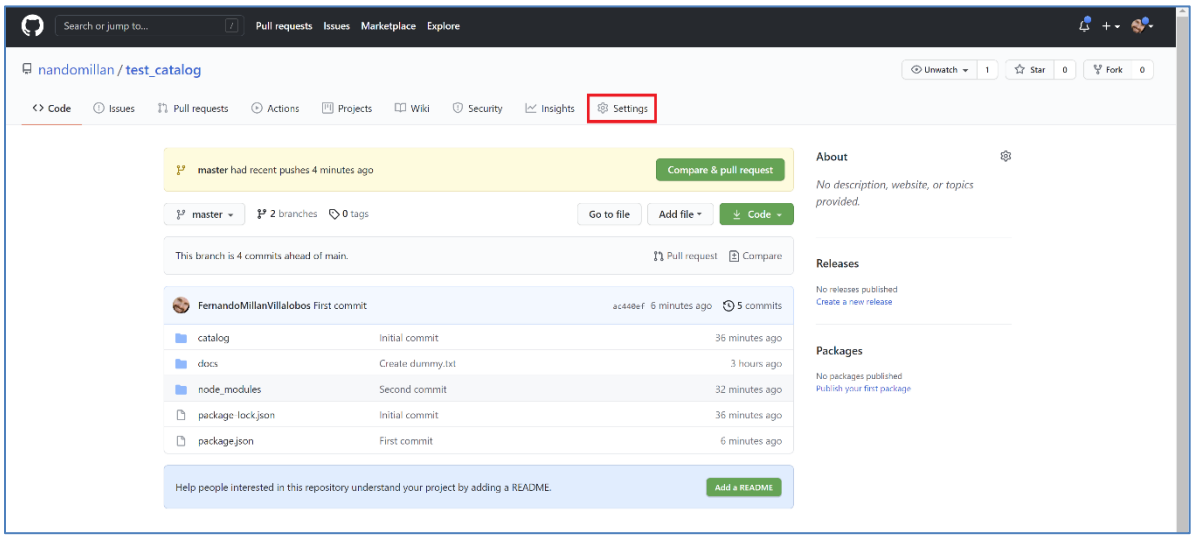
```
PS C:\Users\Fernando Millan\Documents\test_catalog> git push origin master
Enumerating objects: 17961, done.
Counting objects: 100% (17961/17961), done.
Delta compression using up to 12 threads
Compressing objects: 100% (13156/13156), done.
Writing objects: 100% (17960/17960), 17.70 MiB | 4.34 MiB/s, done.
Total 17960 (delta 4514), reused 17126 (delta 4075), pack-reused 0
remote: Resolving deltas: 100% (4514/4514), done.
remote:
remote: Create a pull request for 'master' on GitHub by visiting:
remote:      https://github.com/nandomillan/test_catalog/pull/new/master
remote:
To https://github.com/nandomillan/test_catalog.git
 * [new branch]        master -> master
PS C:\Users\Fernando Millan\Documents\test_catalog>
```

All catalog will be now available in the GitHub repository:

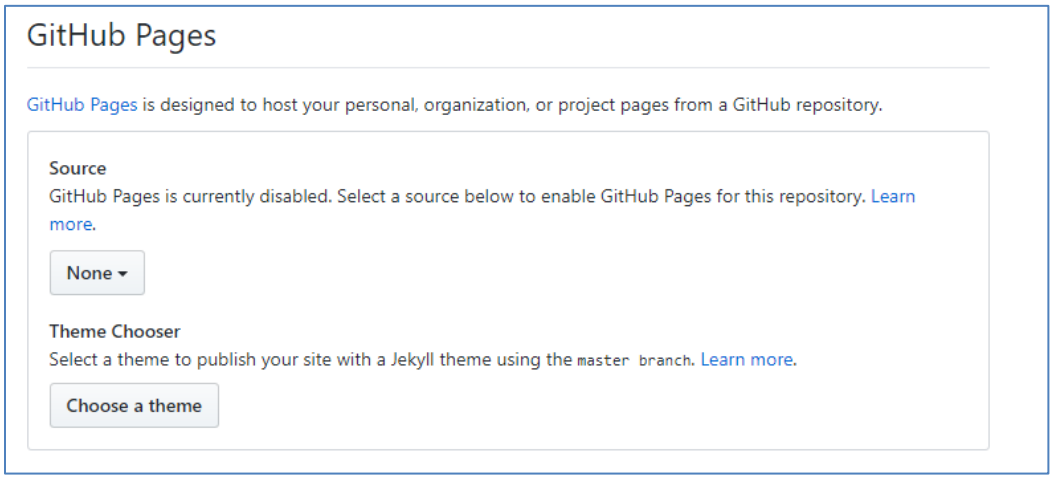


6. CONFIGURE GITHUB PAGES

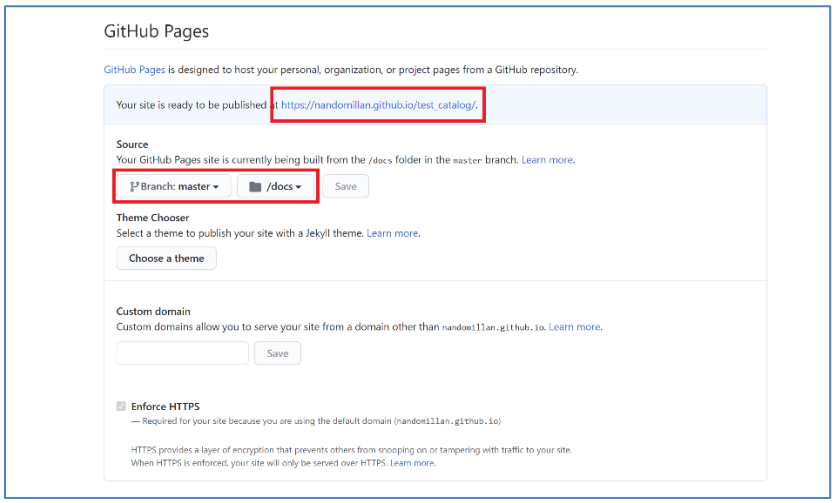
Setup GitHub Pages on the Settings page of your GitHub repository:



In the settings page, scroll down until reaching GitHub Pages section:



Select as source Master branch/docs folder:



Copy the URL where the Page will be published (https://nandomillan.github.io/test_catalog/.) and apply the changes.

PUBLISH TO GITHUB PAGES

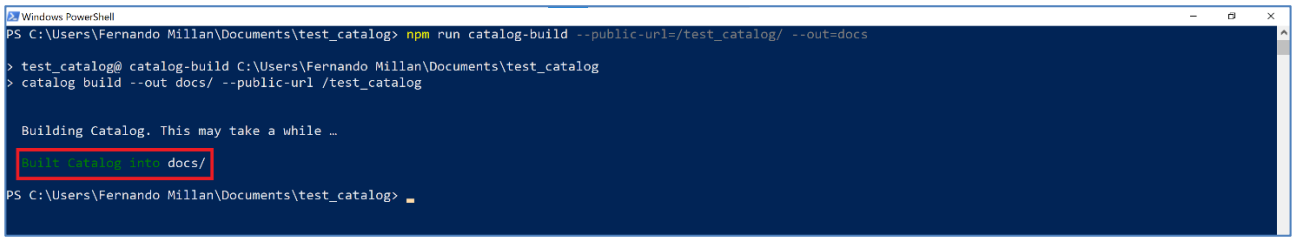
There are two ways to publish: locally or automatic via a GitHub Workflow.
The local build ensures that the build works properly but it requires a local build after each modification of the catalog.
Via a workflow, the build happens automatically evert time a change of the catalog is committed to the repository.

a. Build and Publish locally

i. Build your catalog locally from PowerShell

Use the command npm run catalog-build --public-url=/your-repo-name/ --out=docs
In our case

```
npm run catalog-build --public-url=/test_catalog/ --out=docs
```



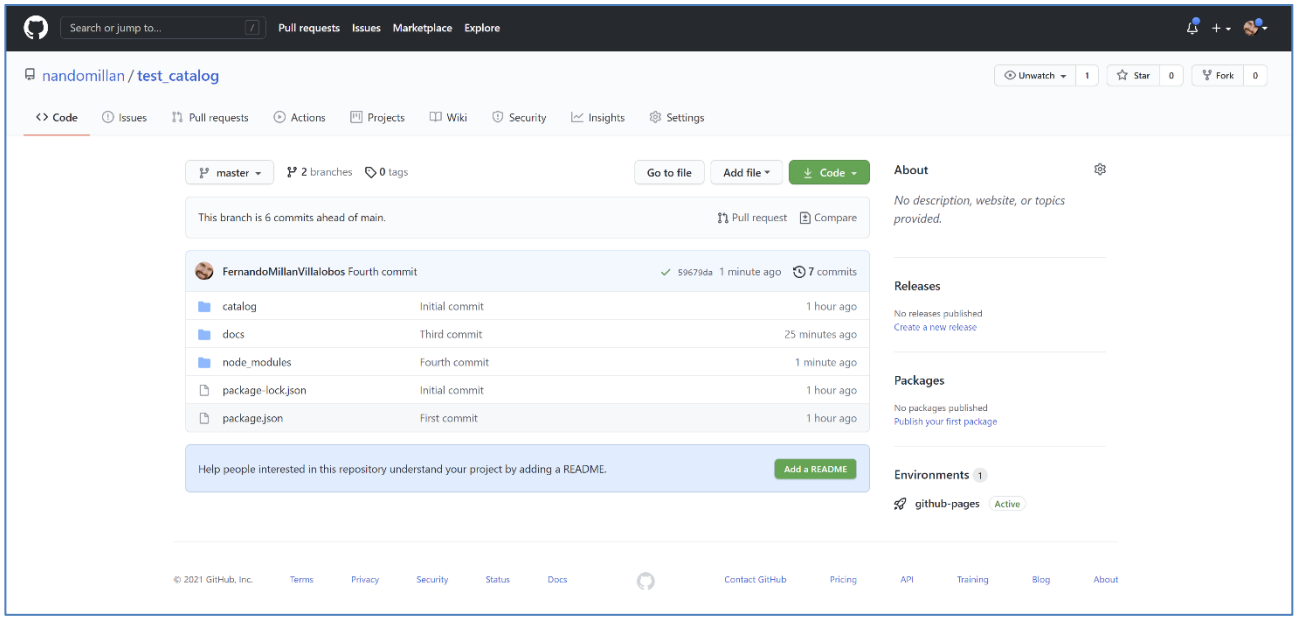
It is very important that the Built is done into docs. Should that not be the case, please review once more the package.json.

ii. Push the code changes to your repository on GitHub

Once the build is confirmed to have happened properly, we can push again the local changes in our catalog repository to the GitHub repository. Proceed as per below:

- **Add the files in your new local repository: git add .**
- **Commit the files that you've staged in your local repository: git commit -m "Commit comments"**
- **Push the changes in your local repository to GitHub: git push origin master**

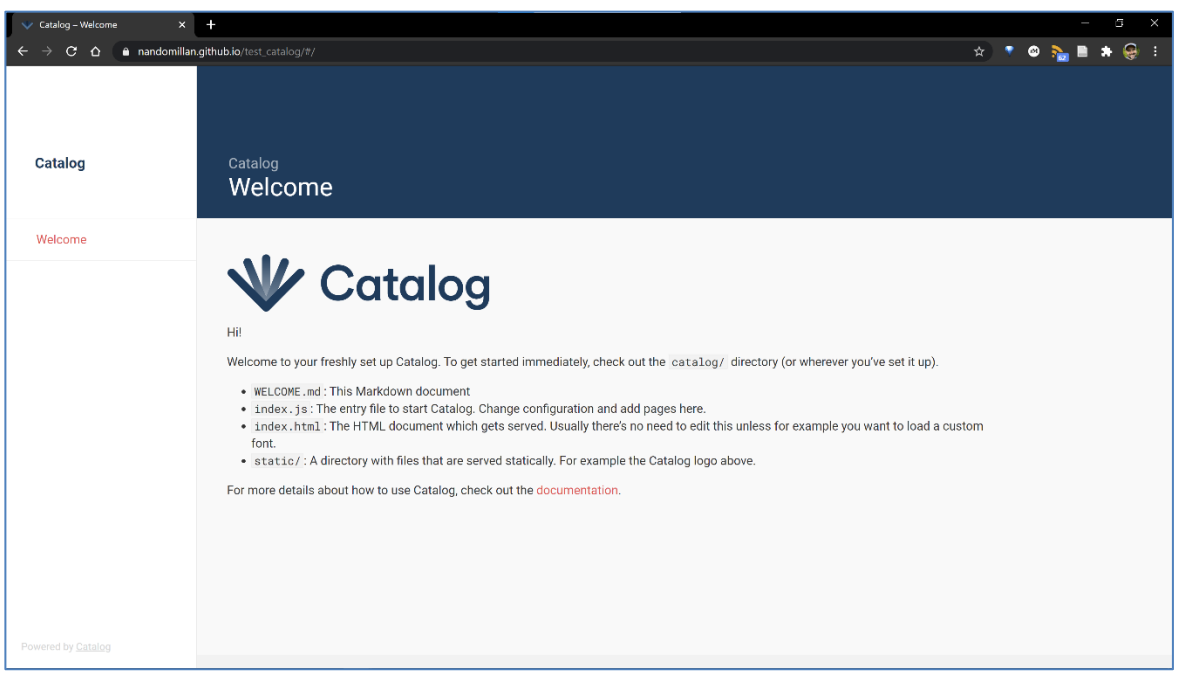
Once completed, the GitHub repository should have the latest version.



iii. Verify outcome in your GitHub pages

The catalog will be displayed in the index.html, hence the link should be GitHub page + index.html:

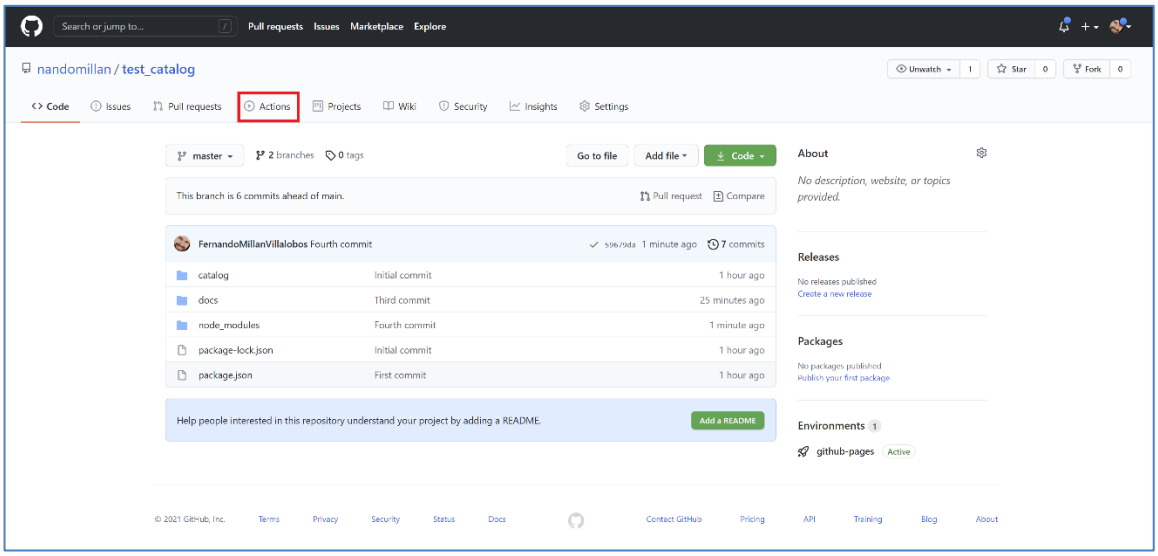
https:// https://nandomillan.github.io/test_catalog/#/



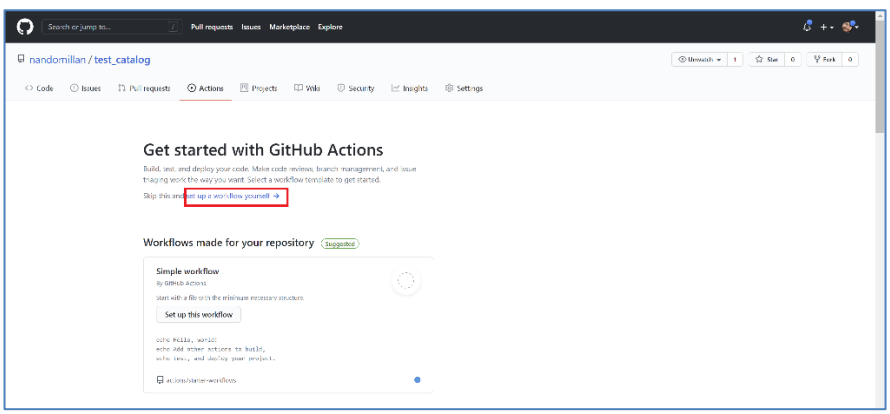
b. Build and publish with a workflow

Once we can confirm that the local build works, we can move forward to enable the automatic build in GitHub with a workflow.

To setup a new Workflow, go to Actions your Github repository:



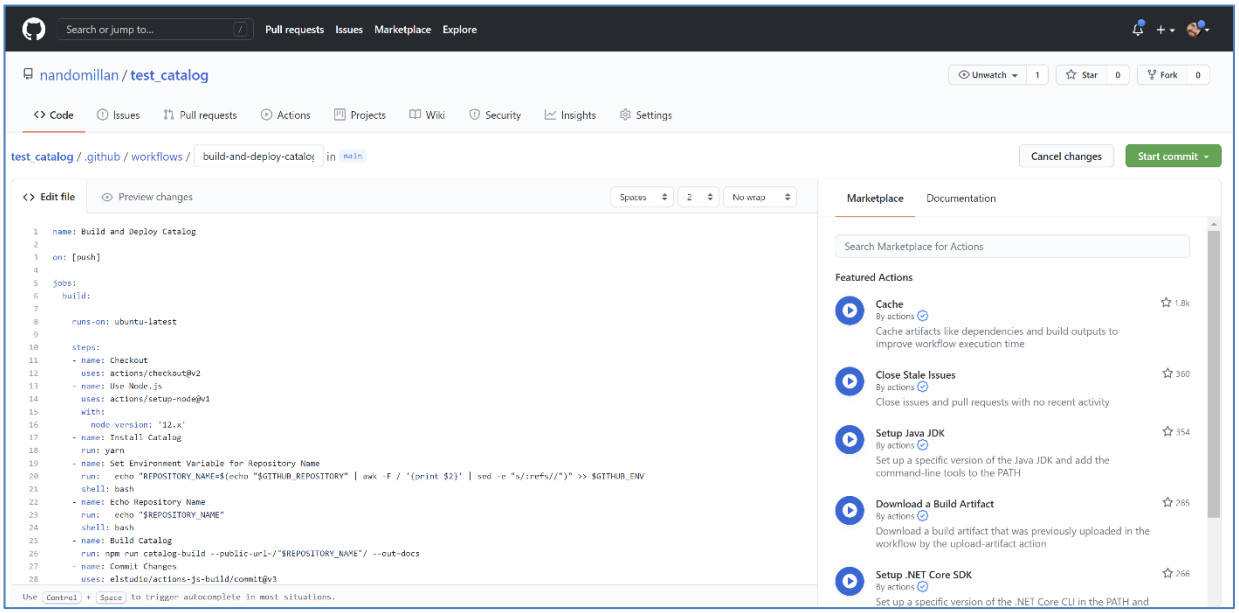
Skip the template selection and set up a fresh workflow for yourself:



Give your workflow a descriptive name (i.e. build-and-deploy-catalog.yml)
Copy and paste the following workflow (workflow has been taken from https://github.com/wiederkehr/catalog-deployment-example-githubpages but the highlighted rows has been edited as per below to use npm instead of yarn. By using yarn, the build was being done to catalog/build instead of repository/docs) and ENVIRONMENT VARIABLE set in the right way according to Github (set-env was deprecated): https://github.blog/changelog/2020-10-01-github-actions-deprecating-set-env-and-add-path-commands/

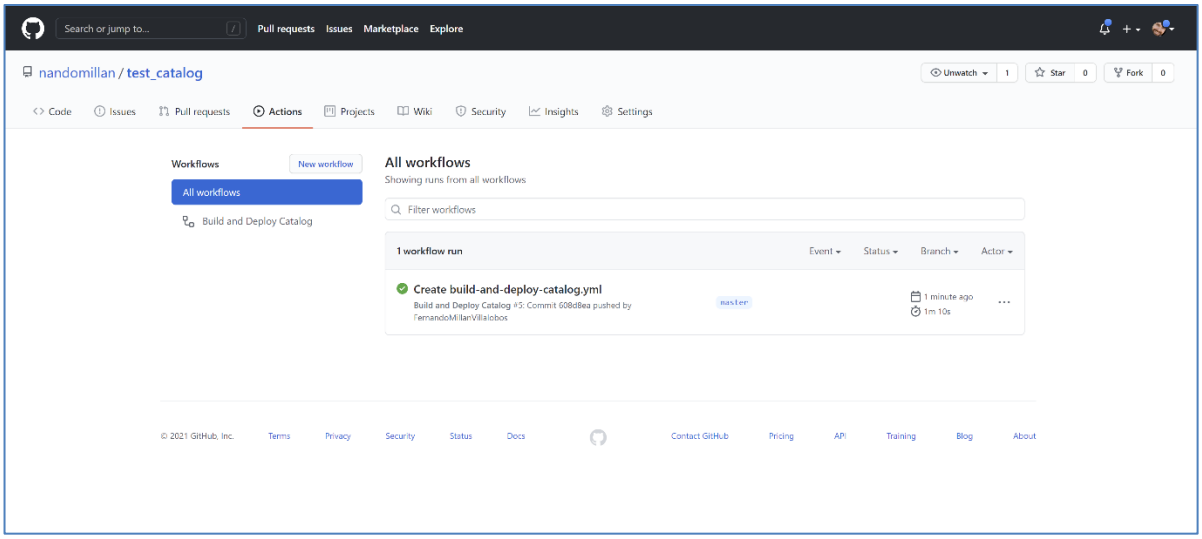
```
name: Build and Deploy Catalog

on: [push]

jobs:
  build:

    runs-on: ubuntu-latest

    steps:
    - name: Checkout
```

```
      uses: actions/checkout@v2
    - name: Use Node.js
      uses: actions/setup-node@v1
      with:
        node-version: '12.x'
    - name: Install Catalog
      run: yarn
    - name: Set Environment Variable for Repository Name
      run:   echo "REPOSITORY_NAME=$(echo "$GITHUB_REPOSITORY" | awk -F / '{print $2}' | sed
-e "s/:refs//")" >> $GITHUB_ENV
      shell: bash
    - name: Echo Repository Name
      run:   echo "$REPOSITORY_NAME"
      shell: bash
    - name: Build Catalog
      run: npm run catalog-build --public-url=/"$REPOSITORY_NAME"/ --out=docs
    - name: Commit Changes
      uses: elstudio/actions-js-build/commit@v3
      with:
        commitMessage: Build Catalog
```



To save the workflow,, click on “Start commit”

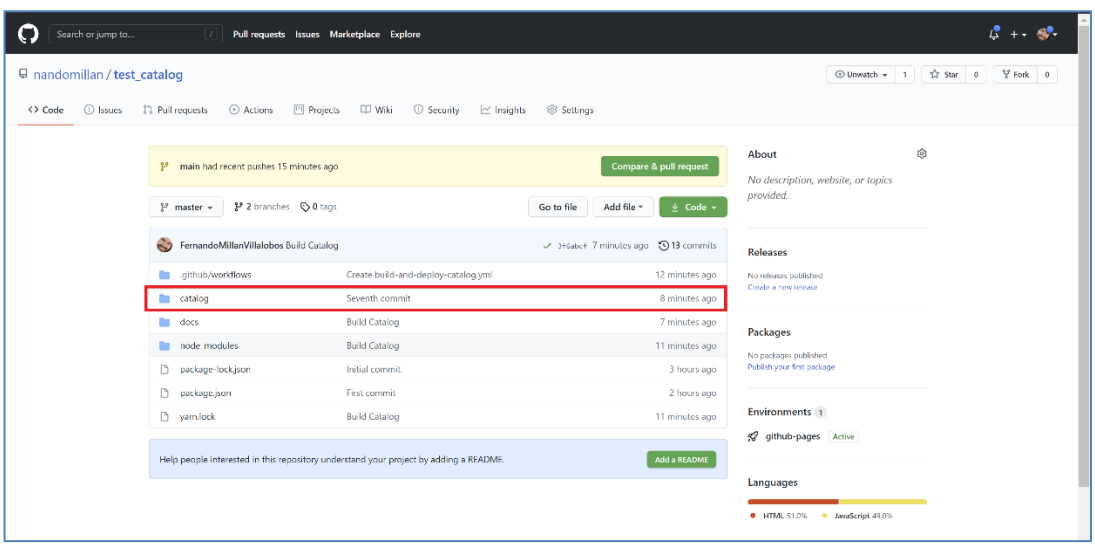
Navigate to Actions page of your repository to see your new workflow listed.



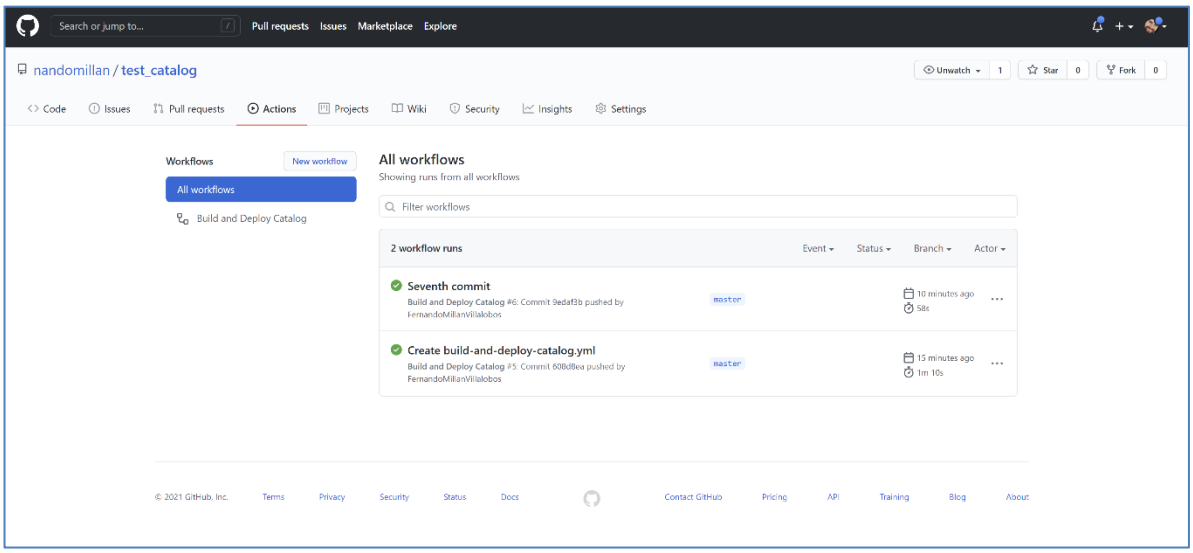
To ensure that this is working as expected, make and commit a code change to your Catalog to see your publication workflow in action (note: because we just created the workflow in GitHub, it will be required to do a pull before we push the changes).

Example, I added few lines to WELCOME.md

After committing the changes to the GitHub repository, validate in Github.

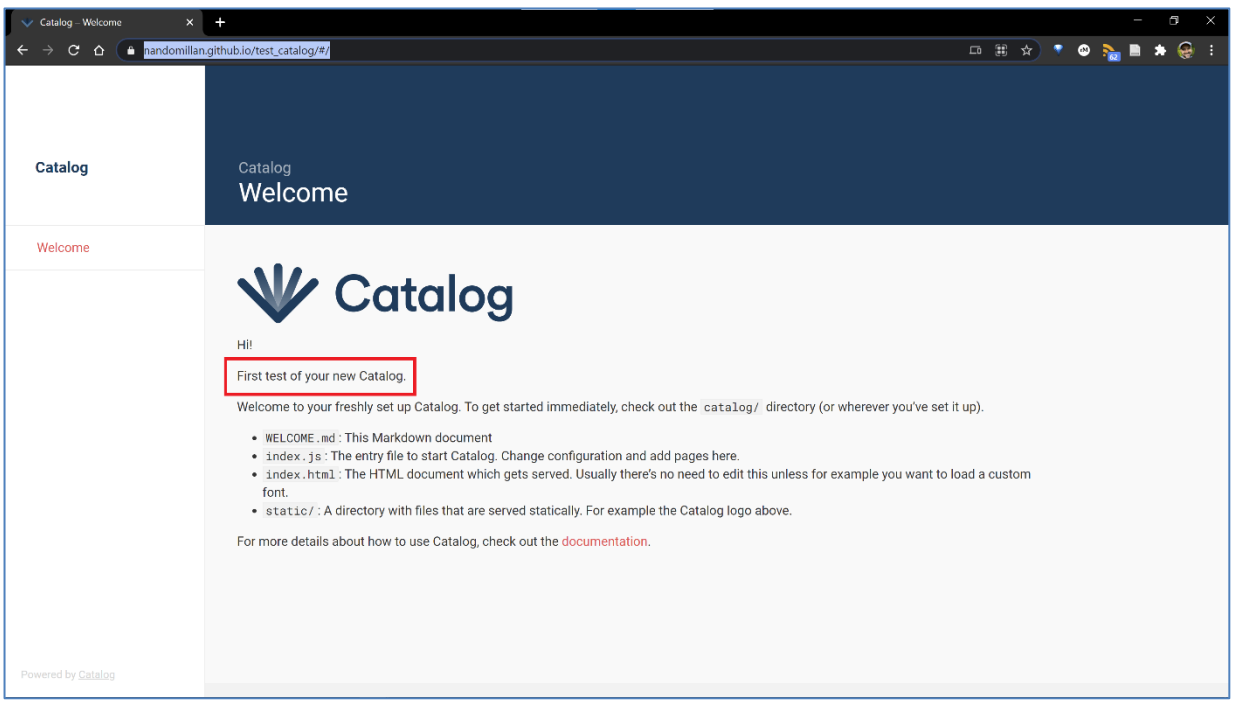


Go then the Actions to see the status of the Workflow:



Not only the workflow should finish ok, but the build should also be done to the docs folder as well.

If the workflow worked properly, verify once more in the GitHub Page:
https://nandomillan.github.io/test_catalog/#/



From now on, every new code change that you push to the master branch will publish a new version of Catalog.

IMPORTANT: because the automatic build is updating the repository, before committing any new changes from the catalog local folder to GitHub we need to do a pull.

7. APENDIX – LIST OF PARAMETERS/VARIABLES

Local folder where Catalog will be created	C:\Users\Fernando Millan\Documents\test_catalog
GitHub Organization	nandomillan
GitHub repository	test_catalog
Remote repository URL	https://github.com/nandomillan/test_catalog
GitHub Page	https://nandomillan.github.io/test_catalog/#/
FS20FUF_Styleguide Example GitHub repository	https://github.com/nandomillan/sgds_styleguide
FS20FUF_Styleguide Example GitHub Page	https://nandomillan.github.io/sgds_styleguide/#/