

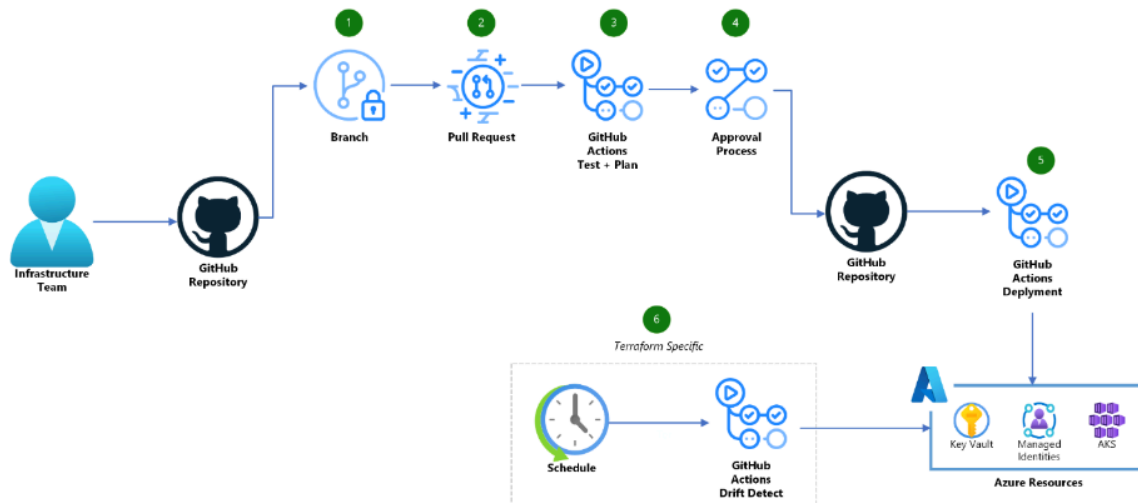
EXPERIMENT NO. 10

Aim: To study and implement deployment of Ecommerce PWA to GitHub Pages.

Theory:

GitHub Pages is a feature provided by GitHub that allows users to host static websites directly from their GitHub repositories. It's a convenient and straightforward way to deploy simple websites, documentation, portfolios, and even PWAs (Progressive Web Apps) like your E-commerce application. Here's some key information about GitHub Pages:

1. **Free Hosting:** GitHub Pages provides free hosting for static websites, making it an attractive option for developers looking to deploy their projects without incurring hosting costs.
2. **Support for Static Content:** GitHub Pages supports static content such as HTML, CSS, and JavaScript. This makes it suitable for hosting PWAs, which typically consist of client-side code that can run entirely in the browser.
3. **Custom Domains:** GitHub Pages allows you to use a custom domain for your website, providing a more professional appearance and easier branding. You can configure a custom domain through the repository settings.
4. **Automatic Deployment:** GitHub Pages automatically deploys your website whenever you push changes to the `gh-pages` branch of your repository. This simplifies the deployment process and ensures that your website stays up-to-date with your latest changes.
5. **HTTPS Support:** GitHub Pages provides HTTPS support for all websites hosted on its platform. This ensures that your website is secure and can be accessed over HTTPS, which is essential for modern web applications.
6. **Usage Limits:** While GitHub Pages is free to use, it does have some usage limits. For example, there are limits on bandwidth and file size, so you may need to consider these limitations when deploying large or high-traffic websites.
7. **Documentation and Support:** GitHub provides comprehensive documentation and support for GitHub Pages, making it easy to get started and troubleshoot any issues you encounter during deployment. You can find detailed guides and tutorials on the GitHub Help website.



Steps for Deploying Website using github pages:

1. Create a GitHub Repository:

- Access GitHub and create a new repository.
- Name the repository and choose its visibility (public or private).
- Optionally, initialize the repository with a README file.
- Create the repository.

2. Prepare Website Files:

- Ensure all website files are ready for deployment, including HTML, CSS, JavaScript, and assets.
- Organize files appropriately.

3. Push Website Files to Repository:

- Clone the repository to your local machine using Git.
- Add the website files to the local repository.
- Commit and push changes to the remote repository on GitHub.

4. Configure GitHub Pages:

- Navigate to the repository's "Settings" tab on GitHub.
- Scroll down to the "GitHub Pages" section.
- Choose the branch to use for GitHub Pages (e.g., `main`, `master`) or a specific folder.
- Save the changes.

5. Verify Deployment:

- GitHub Pages will automatically deploy the website based on the configured settings.
- Visit the provided URL (typically `https://<username>.github.io/<repository>`) to verify the deployment.

6. Custom Domain (Optional):

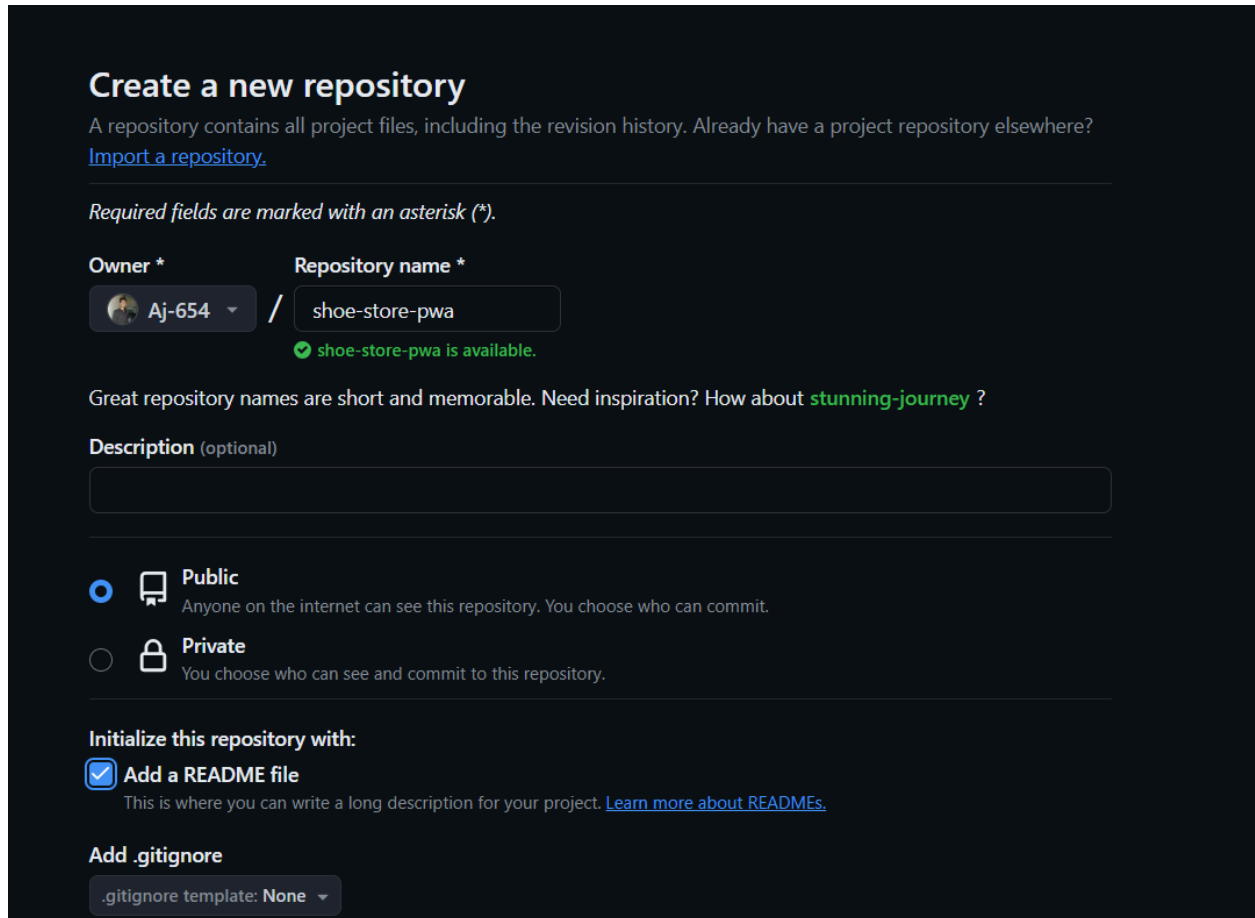
- Configure a custom domain for the website if desired.
- Follow GitHub's instructions to set up the custom domain and update DNS settings.

7. Further Development and Maintenance:

- Continue developing and updating the website as needed.
- Commit and push changes to the GitHub repository to trigger redeployment by GitHub Pages.

Implementation:

1. Create a new repository




Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Required fields are marked with an asterisk ().*

Owner * **Repository name ***

 Aj-654 / shoe-store-pwa

✔ shoe-store-pwa is available.

Great repository names are short and memorable. Need inspiration? How about [stunning-journey](#) ?

Description (optional)

☒ **Public**
Anyone on the internet can see this repository. You choose who can commit.

☐ **Private**
You choose who can see and commit to this repository.

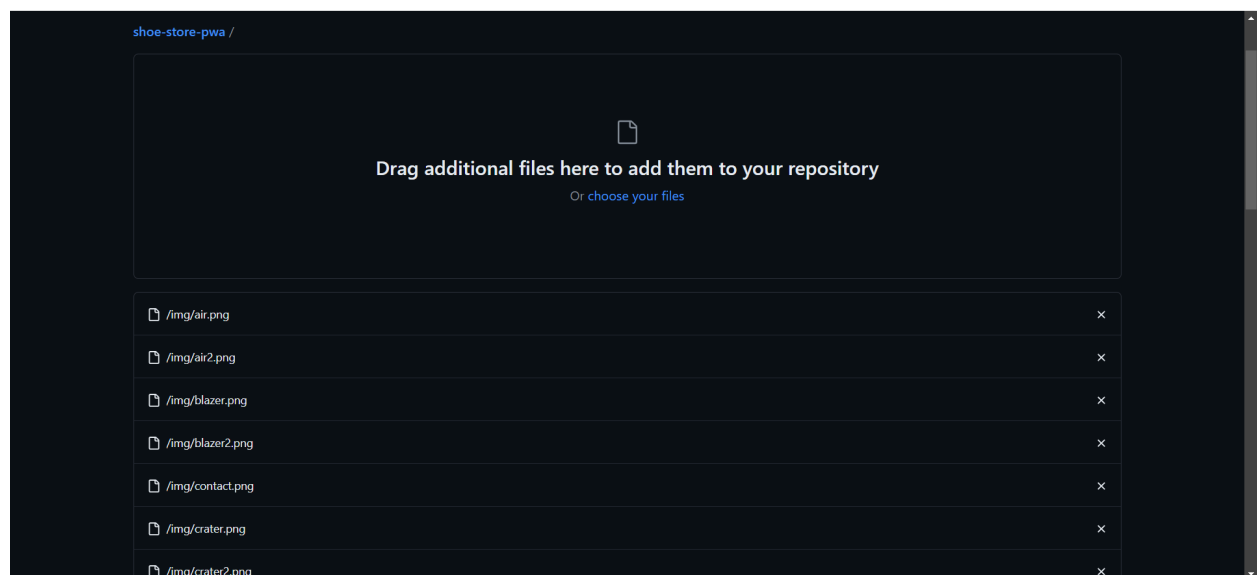
Initialize this repository with:

☒ **Add a README file**
This is where you can write a long description for your project. [Learn more about READMEs.](#)

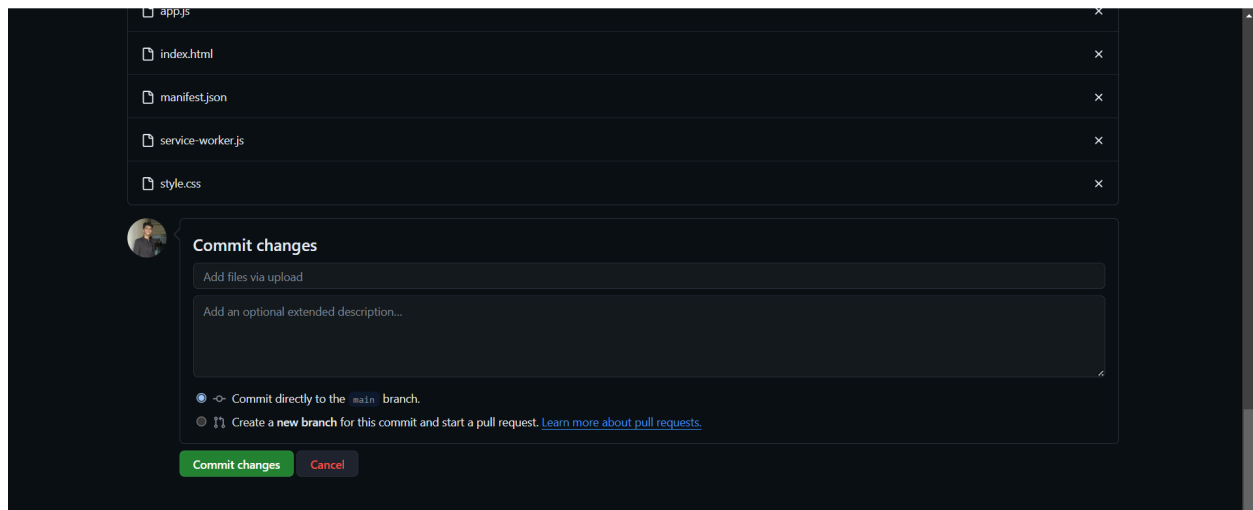
Add .gitignore

.gitignore template: **None**

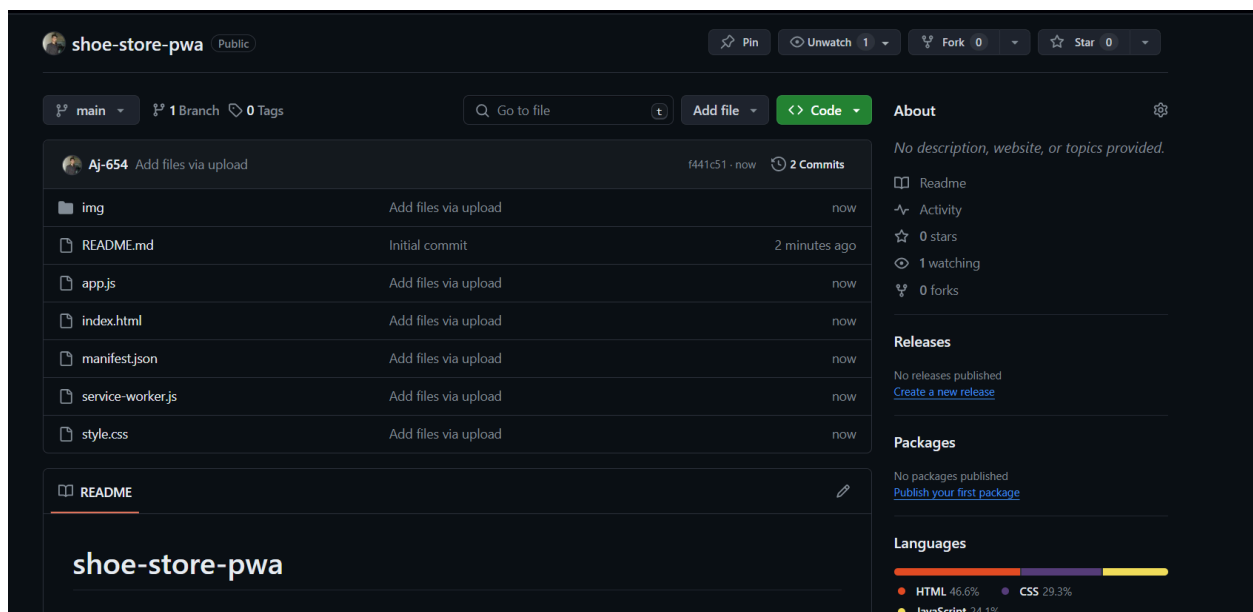
2. Upload our files into that repository



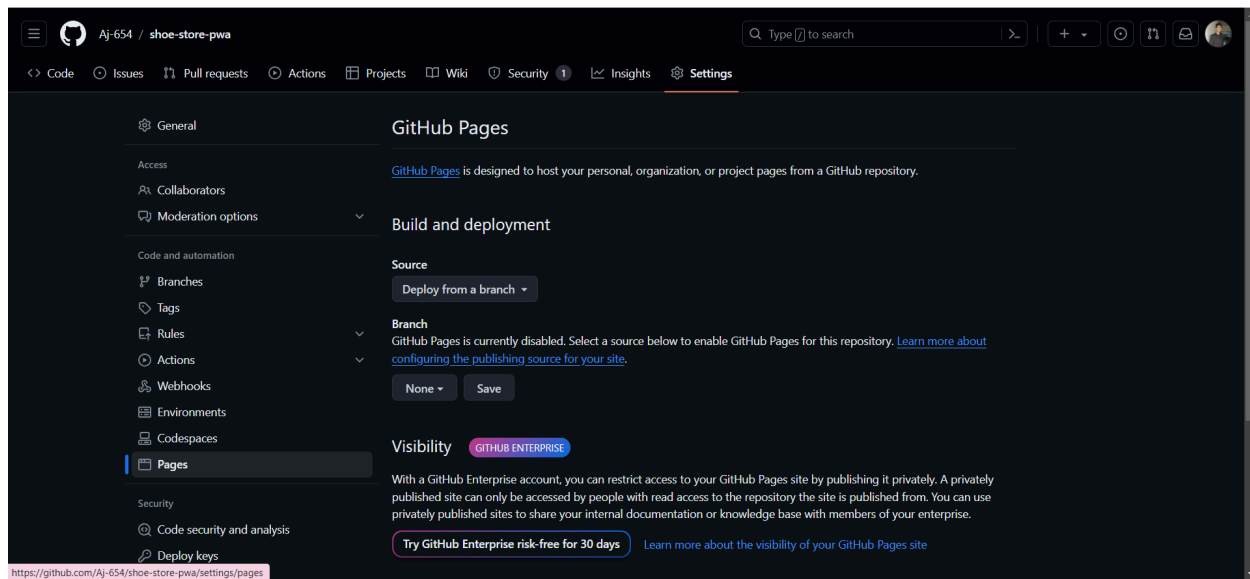
3. Press on Commit to commit the changes.



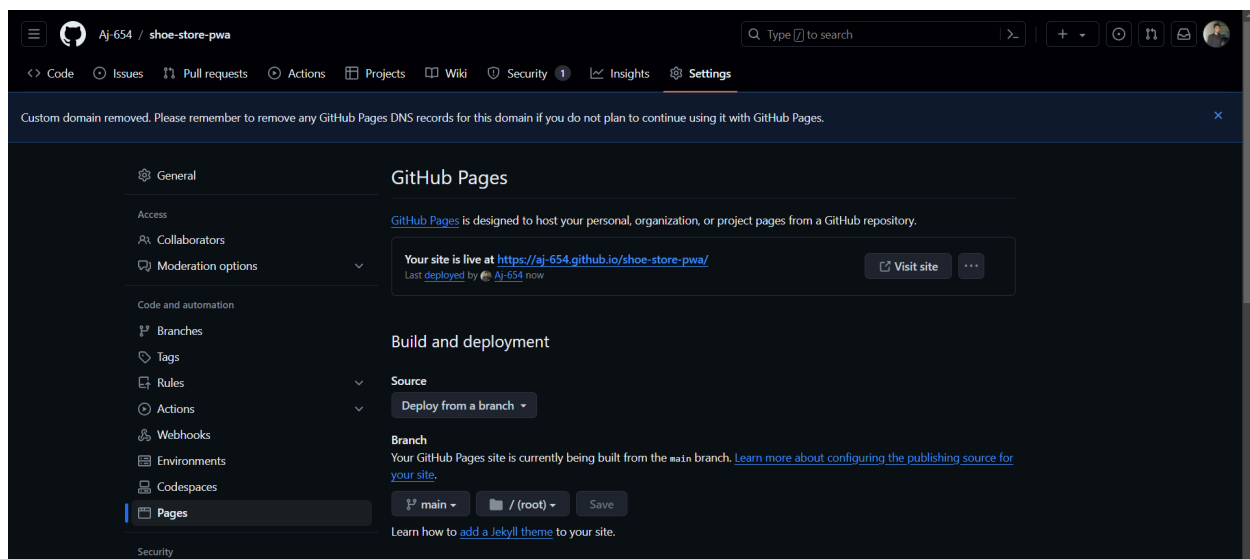
4. Go to settings



5. Select the appropriate branch



6. Click on save and the website has been deployed successfully



Github Repository link: <https://github.com/Aj-654/shoe-store-pwa>

Hosted Website link: <https://aj-654.github.io/shoe-store-pwa/>

Conclusion:

In conclusion, the process of deploying a website using GitHub Pages offers a straightforward and accessible method for hosting static web content. By following the outlined steps, individuals can seamlessly publish their websites online, making them readily available for visitors to access. GitHub Pages serves as a valuable platform, particularly for personal projects, portfolios, and documentation, providing free hosting and simplified deployment procedures. Additionally, the option to configure custom domains adds further flexibility and professionalization to the hosted websites. Overall, leveraging GitHub Pages empowers users to showcase their work and ideas to a global audience with ease, fostering creativity, collaboration, and knowledge sharing in the digital realm.