

Name : Niddhi Rijhwani
Class : D15B Roll No : 47
Subject : MPL Exp 10

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

Theory :

Deployment of Ecommerce PWA to GitHub Pages :

Progressive Web Apps (PWAs) are modern web applications that offer users a native app-like experience while maintaining the accessibility and performance of a web page. They are designed to work offline, load quickly, and provide a reliable experience, making them ideal for ecommerce platforms.

To deploy an Ecommerce PWA to GitHub Pages :

1. **Build the Ecommerce PWA :** Create your ecommerce app using HTML, CSS, JavaScript, or frameworks like React, Angular, or Vue.js. Implement key PWA features like service workers for offline functionality and a manifest.json file for app installation on user devices.
2. **Prepare for Deployment :** Ensure that your app is production-ready by building the project using commands like `npm run build`. This generates the static files (HTML, CSS, JS) that are ready for hosting.
3. **Push Code to GitHub :** Create a GitHub repository, push your project files (including the build folder), and set up the necessary configuration files like `package.json`.
4. **Configure GitHub Pages :** In the repository's settings, enable GitHub Pages and configure it to serve content from the `gh-pages` branch. This makes your app publicly accessible.
5. **Ensure Correct URLs and Caching :** Make sure URLs in your app are correctly linked to prevent issues with broken paths, especially when deployed. The service worker should handle caching of important assets for optimal offline performance.
6. **Test and Deploy :** Once deployed, test your PWA across different devices and browsers to ensure smooth functionality, including offline capabilities and quick loading.

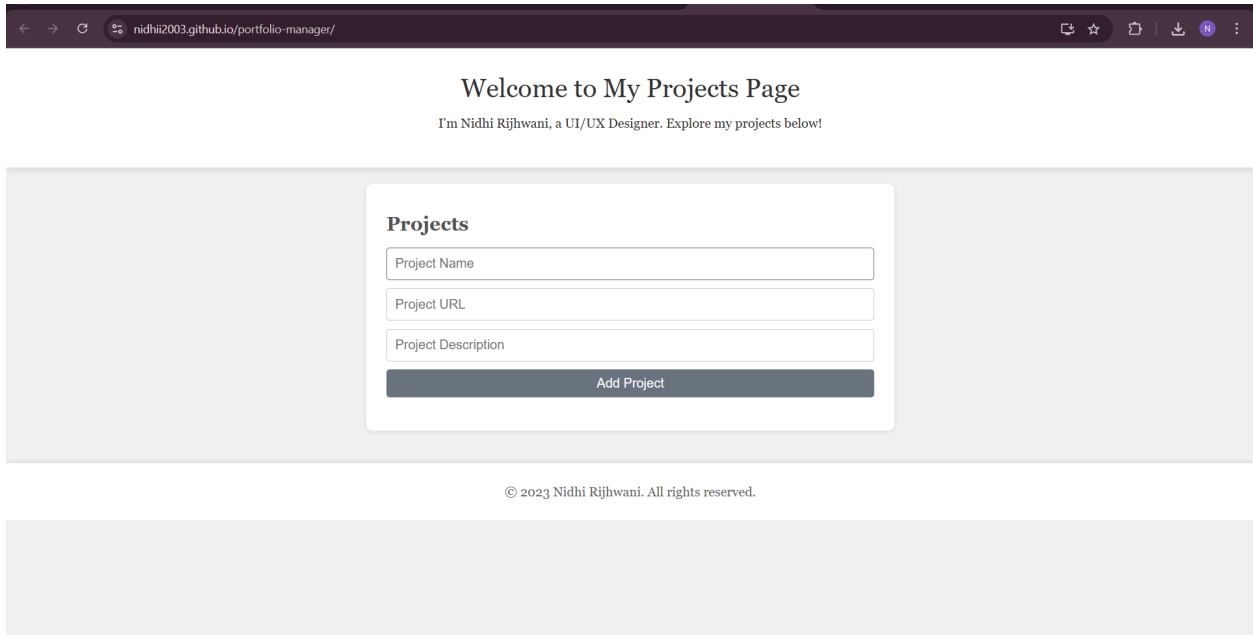
Output :

This screenshot shows the main page of the 'portfolio-manager' repository on GitHub. The repository is owned by 'nidhi2003' and is public. The file browser shows a list of files including 'icons', 'node_modules', 'app.js', 'index.html', 'manifest.json', 'package-lock.json', 'package.json', 'service-worker.js', 'styles.css', and 'vite.config.js'. All files were committed 2 minutes ago by 'Nidhi Rijhwani'. The right sidebar contains an 'About' section, a 'Releases' section with a 'Create a new release' button, a 'Packages' section with a 'Publish your first package' button, and a 'Languages' section showing a bar chart for JavaScript (45.0%), CSS (36.5%), and HTML (18.5%).

This screenshot shows the 'Settings' page for the 'portfolio-manager' repository, with the 'GitHub Pages' tab selected. The left sidebar lists various settings categories: General, Access, Collaborators, Moderation options, Code and automation, Branches, Tags, Rules, Actions, Webhooks, Environments, Codespaces, Pages (selected), Security, and Code Security. The main content area for 'GitHub Pages' includes a description, a 'Build and deployment' section with a 'Deploy from a branch' button, and a 'Visibility' section with a 'Start free for 30 days' button. The 'Branch' section indicates that GitHub Pages is currently disabled.

This screenshot shows the 'Settings' page for the 'portfolio-manager' repository, with the 'GitHub Pages' tab selected. The left sidebar is the same as the previous screenshot. The main content area for 'GitHub Pages' now shows that the site is live at 'https://nidhi2003.github.io/portfolio-manager/'. The 'Build and deployment' section shows the 'main' branch as the source. The 'Custom domain' section is also visible at the bottom.

Link : <https://nidhii2003.github.io/portfolio-manager/>



Welcome to My Projects Page

I'm Nidhi Rijhwani, a UI/UX Designer. Explore my projects below!

Projects

Project Name

Project URL

Project Description

Add Project

© 2023 Nidhi Rijhwani. All rights reserved.

Conclusion :

In conclusion, deploying an Ecommerce PWA to GitHub Pages offers a simple, cost-effective way to provide users with a fast, reliable, and app-like experience. By leveraging GitHub Pages for hosting, you can easily deploy a PWA that works offline, loads quickly, and offers smooth navigation, leading to better user engagement. This method also ensures your ecommerce site is accessible across all devices, enhancing the overall user experience. Using service workers and caching mechanisms further boosts performance, making this approach ideal for modern, scalable ecommerce platforms.