Developer Portfolio with Serverless Functions and Enhanced Features: Assignment 3 Write-Up

For Assignment 3, I expanded the functionality of my 11ty-based developer portfolio by integrating serverless functions alongside Ghost CMS. The primary goal was to dynamically display GitHub repository statistics (Stars, Forks, and Issues) for each project, leveraging serverless functions and ensuring a seamless user experience. This implementation showcases real-time project data while maintaining the scalability and flexibility introduced in Assignment 2.

Theme and Objective

The primary objective for Assignment 3 was to enhance the portfolio's interactivity and information depth by displaying dynamic GitHub stats for each project. This feature complements the existing CMS-based workflow, providing a richer and more engaging experience for users.

By combining serverless functions with Ghost CMS, the portfolio now demonstrates the ability to fetch and display external data dynamically while retaining the scalability of a static site.

Technologies

Ghost CMS:

- Manages project content, including titles, descriptions, tags, GitHub links, and technology stacks.
- Introduced a new tagging convention (repo:username/reponame) for associating GitHub repositories with projects.

Netlify Functions:

- Fetches real-time GitHub stats for each project using GitHub's REST API.
- Integrated seamlessly with the 11ty build process to display dynamic data on the portfolio.

11ty and Nunjucks:

- Extends the existing templating system to incorporate the GitHub stats UI.
- Ensures a cohesive and modern design for all project entries.

GitHub API:

- Provides live repository data, including Stars, Forks, and Issues, enriching each project's display.

New Features and Functionality

Dynamic GitHub Stats Integration
Each project now dynamically displays its real-time stats (Stars, Forks, and Issues) fetched from GitHub.
Implementation:

- Added a repo:username/reponame tag to Ghost CMS for each project.
- Utilized Netlify Functions to fetch data from the GitHub API based on the tag.
- Rendered stats dynamically in the project's layout with a new "Project Stats" section.

Project Stats UI

- Stats are displayed in a visually appealing, neon-glowing "Project Stats" section.
- The section includes clearly labeled counters for Stars, Forks, and Issues.

Alignment: Added CSS improvements from Chat GPT to align stats across projects, ensuring uniformity regardless of content size.

Filtering and Pagination Enhancements

- Maintained the filtering feature by technology tags (HTML, CSS, JavaScript, C++) introduced in Assignment 2.
- Pagination continues to dynamically adjust based on filtered results.

Content Modeling in Ghost CMS

Enhanced Tagging System:

- Introduced a repo: tag format to map each project to its respective GitHub repository.
- Tags like Projects, HTML, CSS, and JavaScript are used for filtering purposes.

GitHub Repository Link:

- Every project includes a direct link to its GitHub repository, complementing the stats display.

Data Fetching and Template Design

Netlify Functions:

- Deployed a serverless function (fetchGitHubStats.js) to Netlify.
- Fetches GitHub repository data at runtime.
- Returns real-time stats to be displayed on the portfolio.

11ty Integration:

- Updated the projects and projects-by-tag templates to include a "Project Stats" container.
- Stats are fetched dynamically using the serverless function during site interaction.

Conclusion

This lab demonstrates the integration of serverless functions into a CMS-powered static site, showcasing both dynamic content updates and live external data fetching. By adding GitHub stats and refining the UI, the portfolio now offers a richer, more interactive experience for users. This scalable setup ensures ease of maintenance and adaptability for future enhancements.