Online Code Editor with Live Preview - Project Report

Introduction

This report details the development of a browser-based Integrated Development Environment (IDE) designed for HTML, CSS, and JavaScript. The primary objective was to create a functional platform that enables users to write code in these languages and instantly view the rendered output in a live preview pane. This project serves as a practical demonstration of web development skills, encompassing front-end development, real-time data handling, and deployment strategies.

Abstract

The Online Code Editor is a web application that provides a seamless coding experience for front-end web technologies. It features separate editor panels for HTML, CSS, and JavaScript, coupled with an immediate live preview of the combined output. Key functionalities include code sharing via URL parameters, allowing users to save and share their work effortlessly, and the integration of pre-defined code templates for quick starts. A significant achievement was the implementation of a real-time console output from the live preview iframe, enhancing debugging capabilities. The project was successfully developed using React and deployed publicly on Netlify, showcasing a full development lifecycle from local setup to global accessibility.

Tools Used

The development and deployment of this project leveraged a combination of modern web technologies and development tools:

- **Front-end Framework:** React (managed via Create React App)
- **Code Editors:** Monaco Editor (`@monaco-editor/react`)
- **Live Preview:** HTML `<iframe>` elements
- **Version Control:** Git
- **Code Hosting:** GitHub
- **Deployment Platform:** Netlify

Steps Involved in Building the Project

The project development followed a structured approach, from initial setup to public deployment:

1. **Project Initialization:** The project was set up using Create React App (`npx create-react-app my-code-editor`), providing a solid foundation for a React application.
2. **Monaco Editor Integration:** The Monaco Editor, a powerful code editor used in Visual Studio Code, was integrated for each language (HTML, CSS, JavaScript) using the `@monaco-editor/react` wrapper. This enabled features like syntax highlighting and basic autocompletion.
3. **Live Preview Implementation:** An `<iframe>` element was used to render the combined HTML, CSS, and JavaScript code in real-time. The `srcDoc` attribute of the iframe was dynamically updated whenever changes were detected in any of the code editors.
4. **Code Sharing via URL:** Logic was implemented to encode the HTML, CSS, and JavaScript code into Base64 format and then append it to the URL as query parameters. This allows users to share their specific code snippets by simply sharing the URL. The application also parses these URL parameters on load to pre-populate the editors.
5. **Template Management:** A set of predefined code templates (e.g., default, simple card, flexbox example) were created and integrated. Users can select and load these templates to quickly start new projects, overwriting current editor content after a confirmation.
6. **Integrated Console/Error Output:** A critical enhancement involved capturing `console.log` , `console.error` , and `console.warn` messages, as well as uncaught JavaScript errors, from within the iframe. These messages are sent back to the parent React application using `window.parent.postMessage()` and displayed in a dedicated console output area, significantly aiding debugging.
7. **Production Build Generation:** Before deployment, the React application was compiled into an optimized production build using `npm run build` . This command generates a `build` folder containing static assets ready for hosting.
8. **Git Initialization and Configuration:** Git was installed and initialized in the project directory. Essential Git configurations, such as `user.email` and `user.name` , were set globally to enable proper commit authorship.
9. **GitHub Repository Setup:** A new public repository (`my-code-editor`) was created on GitHub.com. The local Git repository was then linked to this remote GitHub repository, and the project code was pushed to GitHub (`git push -u origin main`).
10. **Netlify Deployment:** The project was deployed using Netlify, a continuous deployment platform. Netlify was connected to the GitHub repository, automatically detecting the React build settings. A crucial `_redirects` file (`/* /index.html 200`) was added to the `public` folder and pushed to GitHub to correctly handle routing for the Single Page Application, resolving "site not found" errors. Upon successful deployment, the application became publicly accessible via a Netlify URL (e.g., `transcendent-snickerdoodle-5fdb6.netlify.app`).

Conclusion

The Online Code Editor project successfully delivers a functional and user-friendly browser-based IDE, fulfilling the core objective of providing an interactive environment for HTML, CSS, and JavaScript development. The integration of features like live preview, URL-based sharing, code templates, and a real-time console output significantly enhances its utility. The complete development and deployment cycle, including Git version control, GitHub integration, and Netlify hosting, demonstrates a comprehensive understanding of modern web development workflows. This project serves as a strong foundation, with potential for future enhancements such as code formatting, local storage persistence, and advanced layout options.