Industrial Engineering Journal
ISSN: 0970-2555
Volume : 54, Issue 7, July : 2025

CROSS-PLATFORM WINDOWS 11 DESKTOP SIMULATION WITH WEB TECHNOLOGIES

SOUMYA PRIYADARSHANI SAHOO 4th Year, Department of CSE, Gandhi Institute for Technology, BPUT, India soumyasahoo2021@gift.edu.in
CHANDRA SEKHAR GIRI 4th Year, Department of CSE, Gandhi Institute for Technology, BPUT, India csekhar2021@gift.edu.in

Abstract—
This project explores the development of a cross-platform Windows 11 desktop simulation using modern web technologies such as HTML5, CSS3, JavaScript, and frameworks like React.js and Electron. The simulation aims to replicate the look, feel, and interactivity of the Windows 11 desktop environment, including features like the Start Menu, taskbar, window management, and settings interface. By leveraging web technologies, the application can run seamlessly on multiple platforms—Windows, macOS, and Linux—without native code. The goal is to provide users, developers, and educators with a lightweight, customizable environment for learning, prototyping UI concepts, or enhancing accessibility to Windows-like interfaces. This simulation also supports touch and responsive design for use on tablets and mobile devices. The project highlights the versatility of web development tools in creating immersive, system-like user experiences without relying on traditional OS-level APIs or resources..

Keywords:
HTML, CSS, NODE-JS

I. INTRODUCTION

In the era of cloud computing and platform-independent applications, simulating a Windows 11 desktop environment using web technologies presents a powerful solution for accessibility, education, and lightweight computing. A cross-platform Windows 11 desktop simulation built with HTML, CSS, and JavaScript allows users to experience and interact with a familiar desktop interface directly through a web browser, without the need for native operating system installation. This approach enables seamless access across different devices and operating systems—including macOS, Linux, and mobile platforms—making it ideal for environments where native Windows access is limited or impractical. Utilizing modern front-end frameworks and APIs, developers can replicate key UI components like the Start Menu, taskbar, window management, and file explorer, providing an engaging and interactive user experience. Additionally, such simulations can serve as educational tools, testing platforms, or even lightweight remote desktops for specific tasks. This project highlights the capabilities of web technologies to deliver rich, OS-like experiences that are platform-agnostic, customizable, and accessible from virtually anywhere.

II. LITERATURE REVIEW

Recent advancements in web technologies have enabled the development of operating system simulations within browsers, leveraging HTML5, CSS3, and JavaScript. Projects like Windows 11 in React and WebDesktop demonstrate the feasibility of replicating desktop environments using front-end frameworks such as React and Vue.js. Research on cross-platform solutions emphasizes the benefits of browser-based environments for enhanced accessibility, device independence, and reduced resource consumption. Studies also highlight challenges in achieving full OS functionality, including file system emulation, multitasking, and security. Despite limitations, literature supports the growing role of web-based simulations in education, remote access, and lightweight computing environments.

III. SYSTEM DESIGN

The system design for a cross-platform Windows 11 desktop simulation using web technologies involves a modular, component-based architecture. The user interface is developed using HTML5 and styled with CSS3 to mimic the Windows 11 aesthetic. JavaScript, along with frameworks like React or Vue.js, manages dynamic behaviors and component interactions such as window movement, resizing, and taskbar updates. A virtual file system is simulated using browser storage APIs like LocalStorage or IndexedDB. The system is responsive and optimized for various devices and screen sizes. Backend services, if needed, are integrated via RESTful APIs, ensuring a seamless, interactive, and scalable desktop experience.

IV. IMPLEMENTATION

The implementation of a cross-platform Windows 11 desktop simulation begins with designing the UI using HTML5 and CSS3 to replicate the Start Menu, taskbar, desktop icons, and windows. JavaScript, along with a front-end framework like React or Vue.js, is used to handle state management, user interactions, and dynamic component rendering. Features like drag-and-drop windows, multi-window support, and a simulated file explorer are implemented using DOM manipulation and browser APIs such as LocalStorage or IndexedDB. Event listeners manage user inputs, while optional backend integration through RESTful APIs supports user authentication or file operations, enabling a fully interactive and responsive desktop simulation.

RESULTS

The cross-platform Windows 11 desktop simulation successfully replicates key features of the Windows 11 interface within a web browser. Users can interact with familiar components such as the Start Menu, taskbar, desktop icons, and resizable application windows. The system runs smoothly across multiple platforms—including Windows, macOS, Linux, and mobile devices—without the need for installation. Basic file operations are enabled through LocalStorage, providing a functional virtual file system. The simulation demonstrates responsive design, efficient performance, and cross-device compatibility. Overall, it serves as an effective proof of concept for delivering desktop-like environments using only web technologies, highlighting the potential for broader applications**.**

CONCLUSION

The cross-platform Windows 11 desktop simulation using web technologies effectively demonstrates the potential of HTML5, CSS3, and JavaScript in replicating a desktop environment within a web browser. The simulation successfully mirrors key features of Windows 11, offering an interactive and responsive experience across various devices and platforms. By leveraging web APIs and front-end frameworks, the project highlights the feasibility of creating lightweight, accessible, and cross-platform solutions for users without native OS installation. While limitations exist in achieving full OS functionality, this approach showcases the promise of web-based desktop simulations in education, remote access, and lightweight computing**.**

ACKNOWLEDGEMENT

We extend our sincere appreciation to all individuals and organizations whose contributions have been instrumental in the development of the real-time weather application. Special thanks to meteorological experts and researchers whose invaluable insights and advancements have enhanced our understanding of weather forecasting and data processing. We acknowledge the support of technology partners for their innovative solutions in sensor technologies and data acquisition. Furthermore, we express gratitude to the users whose feedback and preferences have guided the design and functionality of the application. This collaborative effort underscores our commitment to providing accurate, reliable, and accessible weather information to users worldwide.

REFERENCES

- http://www.wikipedia.com/
- http://www.w3schools.com/
- http://www.reactjs.org/
- https://dev.to/achowba/building-a-modal-in-react- https://dev.to/achowba/building-a-modal-in-react-15hg#%3A~%3AtargetText%3DOpen%20the%20Modal.js%20file%2C%7B%7B%20transform%3A%20props.show%20%3F