I selected the Trivia app artifact for this enhancement because it showcases a wide range of software development skills, from front-end design to efficient state management and component-based architecture. This project demonstrates my ability to build interactive, dynamic web applications, emphasizing user experience and functionality. Originally built with vanilla JavaScript, the app was significantly enhanced by transitioning to React and Vite, demonstrating my ability to modernize and optimize a codebase using modern web development frameworks and tools. This enhancement highlights my proficiency in component-based architecture, state management, and user interface design in React.

By using Vite as the build tool, I also improved the performance and developer experience of the application, showcasing my ability to implement cutting-edge tools and methodologies.

Transferring the trivia app from vanilla JavaScript to React introduced new challenges, particularly with managing component state and React’s one-way data flow. Handling state across multiple components while ensuring the quiz logic functioned correctly required a deeper understanding of React’s architecture. Incorporating Vite into the development process also introduced a learning curve, but it allowed me to set up a more efficient development environment, improving the app's performance. This experience reinforced my skills in modern web development and helped me apply design patterns that make the application more scalable and easier to maintain.

When transferring the trivia app from vanilla JavaScript to React + Vite, I broke the app down into smaller, reusable components and implemented React’s state management to handle dynamic changes. The first step was setting up the project with Vite, a modern build tool that offers a faster development environment. I then structured the app into a functional component using React’s useState hook to manage the quiz's state, such as the current question index, the user’s score, and whether the quiz was completed.

One of the key changes was moving from directly manipulating the DOM to React’s virtual DOM, which allows for more efficient updates. Each time a user selects an answer, the handleChoiceChange function updates the state, and the quiz dynamically renders the next question by incrementing the question index. The logic for checking answers and updating the score was managed within the checkAnswer function, which also controls the flow of the quiz by checking if there are more questions or if the quiz has finished.

This enhancement demonstrates my ability to apply well-founded and innovative techniques, skills, and tools in software development to create efficient and maintainable solutions. Specifically, it showcases software design principles such as modularity, reusability, and maintainability, aligning with outcomes related to software engineering best practices. By transitioning the app to React, I showcased my ability to adapt to new frameworks and design interactive, dynamic web applications using modern development techniques.