React Single Page Application Using Create React App

Front-End Development, Toolchain Configuration, Testing, and Production Deployment

ABSTRACT

Modern web application development demands user interfaces that are responsive, maintainable, and capable of delivering a seamless experience across devices and varying network conditions. As the complexity of front-end systems has grown, so too has the sophistication of the tooling required to build them effectively. Configuring a React project from scratch—integrating Babel for transpilation, Webpack for module bundling, ESLint for code quality enforcement, Jest for testing, and optimised build pipelines for production deployment—is a non-trivial undertaking that introduces significant friction, particularly for beginners and small development teams working under time constraints. To address these challenges, Create React App (CRA) was introduced as a command-line interface tool that abstracts away manual project configuration by providing a fully pre-configured, production-ready development environment with a single command. This project investigates the design, development, testing, and deployment of a React Single Page Application bootstrapped using CRA, treating the application not merely as a functional deliverable but as a structured evaluation of the CRA toolchain as a foundation for professional front-end development, examining its capabilities, trade-offs, workflow conventions, and positioning relative to modern alternatives such as Vite and Next.js.

KEYWORDS

React, Create React App, Single Page Application, Webpack, Babel, npm Scripts, Front-End Tooling, JavaScript, TypeScript, Jest, React Testing Library, Build Optimisation, Code Splitting, Lazy Loading, Vite, Next.js, CSS Modules, React Router.

SYNOPSIS

The application demonstrates the full breadth of modern front-end development practice within the CRA environment, covering component-based UI design using functional React components and hooks, client-side routing via React Router, local and global state management strategies, and production deployment to static hosting platforms. Performance optimisation through code splitting and lazy loading is explored alongside CRA's four primary npm workflow scripts: npm start for live-reloading development, npm test for Jest-powered watch-mode testing, npm run build for minified production bundle generation, and npm run eject for exposing the underlying Webpack and Babel configuration for advanced customisation. The project also situates CRA within the broader front-end tooling landscape by comparing it against traditional manual Webpack setups and modern alternatives including Vite and Next.js, evaluating each across build speed, configuration flexibility, ecosystem support, and suitability for projects of varying scale and complexity.

TECHNOLOGIES

Core Framework: React — component-based JavaScript library for building declarative, virtual DOM-powered user interfaces with functional components and hooks.

Project Scaffolding: Create React App (CRA) — zero-configuration CLI tool providing a pre-configured development environment, build pipeline, and test runner out of the box.

Transpilation & Bundling: Babel and Webpack — Babel transpiles modern JavaScript and JSX to browser-compatible ES5; Webpack bundles, tree-shakes, and optimises all module dependencies.

Routing: React Router — declarative client-side routing library enabling multi-page navigation within the Single Page Application without full page reloads.

Testing: Jest and React Testing Library — Jest provides the test runner and assertion framework; React Testing Library enforces behaviour-driven component testing aligned with user interactions.

Build Analysis: Lighthouse and Source Map Explorer — Lighthouse profiles runtime performance and accessibility; Source Map Explorer analyses production bundle composition for size optimisation.

Styling: CSS Modules — scoped, locally namespaced stylesheets preventing class name collisions and enforcing style encapsulation at the component level.

PROCEDURE

Step 1: Project Setup and Component Design

The project is initialized using Create React App (CRA), which sets up a ready-to-run React environment with