Senior Frontend Engineer Take-Home Exercise

This take-home exercise is designed to assess your ability to:

* Architect a scalable React application
* Implement clean, maintainable code with best practices
* Demonstrate proficiency in component design, state management, and testing
* Showcase UI/UX sensibility and performance optimizations

**Exercise Prompt**

Build a small dashboard application that:

1. Fetches a list of users from a public API (e.g., https://jsonplaceholder.typicode.com/users)
2. Displays the users in a searchable, sortable table
3. Shows detailed profile information in a modal when a row is clicked
4. Includes a form to add a new user (locally; no API POST needed)
5. Implements client-side routing to navigate between “User List” and “Add User” views

**Requirements**

1. **Architecture & Tech Stack**
   * Use React with functional components and hooks (e.g., useState, useEffect)
   * Use React Router for navigation
   * State management choice (Context API, Redux, or another tool) with rationale
2. **UI/UX**
   * Responsive design for desktop and mobile
   * Accessible components (ARIA attributes, keyboard navigation)
   * Clean, consistent styling (CSS-in-JS, Tailwind, or CSS Modules)
3. **Functionality**
   * Search filter by name and email
   * Sort by columns (ascending/descending)
   * Modal for viewing details: address, phone, company
   * Form validation on “Add User” (e.g., required fields, proper email format)
4. **Performance & Quality**
   * Optimize list rendering (e.g., virtualization if >50 items)
   * Code splitting and lazy loading for routes or heavy components
   * Error handling and loading states
5. **Testing**
   * Unit tests for key components (e.g., table, form)
   * Integration or E2E test for user flows (optional but encouraged)

**Evaluation Criteria**

| **Category** | **Weight** | **Focus Areas** |
| --- | --- | --- |
| Code Quality | 25% | Readability, modularity, naming, comments |
| Architecture & Patterns | 20% | Scalability, maintainability, and proper use of React patterns |
| Functionality | 20% | Completeness, correctness, edge cases |
| UI/UX & Accessibility | 15% | Design consistency, responsiveness, ARIA, keyboard support |
| Performance | 10% | Efficient rendering, lazy loading, bundle size considerations |
| Testing | 10% | Coverage, meaningful tests, and use of testing frameworks |

**Submission**

* Provide a public Git repository with commit history
* Include a README with setup instructions, architectural decisions, and any assumptions
* Optional: deploy to Netlify, Vercel, or GitHub Pages and share the URL