Campus Hiring Evaluation - Frontend

Afford Medical Technologies Private Limited

July 15, 2025

Terms & Conditions

This document and the associated assessment contain confidential and proprietary information of Afford Medical Technologies Private Limited ("Affordmed"). By accessing this material or participating in this assessment, you acknowledge receipt of this information for the sole purpose of evaluating your candidacy for an internship, contract, or employment with Affordmed. You hereby agree to the following:

- **Confidentiality:** You shall maintain the strict confidentiality of all information received and refrain from sharing, distributing, or disclosing any part of the information to any third party.
- **Non-Tampering:** You shall not tamper with, disrupt, or attempt to compromise any Affordmed or its vendor's cloud or software resources provided for this assessment.
- **Sole Use:** You shall use this material solely for the purpose stated herein and for no other purpose whatsoever.

Any unauthorised use, disclosure, or tampering will result in immediate disqualification from the candidacy process and may subject you to legal action. You consent to the exclusive jurisdiction of the Courts of Hyderabad/Secunderabad for any legal disputes arising from this agreement. Any reference to third parties within this document is purely coincidental and for illustrative purposes only, and does not constitute any endorsement or affiliation.

Deliverables & Evaluation Considerations

Time Limit: 2 Hours (No extra time for pushing to GitHub)

- **Code Implementation:** Deliver a fully functional, responsive React frontend web application that adheres to all specified requirements and general constraints, demonstrating robust error handling, high code quality, and efficient UI design suitable for a production environment.
- **Design Document:** Provide a concise document that outlines your architectural and code design choices for the React application. This should include key decisions, data modeling (especially for client-side persistence), technology selections with

justifications, routing strategy for URL handling/redirection, and any assumptions made, reflecting a comprehensive understanding of scalable, maintainable, and user-centric system design.

Develop a React URL Shortener Web App

Ensure youve followed the instructions and registered as provided in the Campus Hiring Evaluation - Pre-Test Setup document before starting.

Your task is to develop a user-friendly URL Shortener application that provides core URL shortening functionality and displays analytical insights, all managed within the client-side application.

Requirements & Constraints

- **Mandatory Logging Integration:** Your app MUST extensively use the Logging Middleware you created in the Pre-Test Setup stage. Use of inbuilt language loggers or console logging is not allowed.
- **Application Architecture:** Implement a React application.
- **Authentication:** For the purpose of this evaluation, assume users accessing your APIs are pre-authorized. Your application must not require user registration or login mechanisms for API access.
- **Short Link Uniqueness:** All generated short links within the application must be unique. The application must manage this uniqueness.
- **Default Validity:** If a user does not specify a validity period for a shortened URL, it must default to 30 minutes. Validity input from the user will always be provided as an integer representing minutes.
- **Custom Shortcodes:** Users may optionally provide a custom shortcode of their choice. If a shortcode is provided, your service must attempt to use it, ensuring it is unique and valid (e.g., alphanumeric, reasonable length). If no shortcode is provided, your service must automatically generate a unique shortcode.
- **Redirection:** When a user accesses a shortened URL (e.g., http://hostname:port/abcd1) from the short URL creation's result page or from the statistics page, the React application must handle the route and redirect them to the original long URL. This implies client-side routing and management of the URL mappings.
- **Error Handling:** Implement robust client-side error handling. Display appropriate user-friendly messages for invalid inputs (e.g., malformed URL, shortcode collision) and other operational issues.
- **Running Environment:** Your React application must run exclusively on http://localhost:3000.
- **User Experience:** Care must be taken to avoid cluttering the page. The UI must prioritize user experience, with a focus on highlighting key elements of each page.

- **Styling Framework:** Use Material UI only. If you are not familiar with Material UI, employ native CSS. Use of ShadCN or other CSS libraries is prohibited. Solely relying on native CSS or not using Material UI will result in lower scores.

Your application should consist of the following pages:

URL Shortener Page

- **Functionality:** This page should allow users to shorten up to 5 URLs concurrently. For each URL, the user should be able to provide:
  - The original long URL.
  - An optional validity period (in minutes).
- **Client-Side Validation:** Prior to making API calls to the backend, the users inputs should be validated based on the constraints already specified in the "General Requirements & Constraints" section (e.g., valid URL format, validity as an integer).
- **Display Results:** Upon successful creation of shortened URLs, the application must display the shortened links along with their respective expiry dates, clearly associated with each original URL provided by the user.

URL Shortener Statistics Page

- **Functionality:** This page should display a list of all shortened URLs created (either in the current session or historically if data persistence is implemented and accessible). For each listed shortened URL, it should clearly present:
  - The shortened URL itself and its creation and expiry date times.
  - The total number of times the short link has been clicked.
  - Detailed click data, including:
    - The timestamp of each click.
    - The source from which the click originated.
    - The coarse-grained geographical location of the click.