System Architecture

* App Component

This React application's App component manages routing using React Router. It conditionally renders components based on user authentication status, directing users to sign up or log in if not authenticated, and displaying course and contact components if authenticated.

* Signup Component

This React component SignUp renders a form for user registration, utilizing useSignup custom hook to handle form submission, including user input fields for name, email, password, phone number, and age, while displaying loading state during form submission.

* Login Component

This React component Login renders a login form that captures user email and password inputs. It uses the useLogin custom hook to handle form submission, allowing users to log in and navigate to the '/course' page upon successful authentication, with error handling for login failures.

* UseSignup hook

This custom hook useSignup handles user registration (signup) using Firebase authentication (createUserWithEmailAndPassword) and Firestore (db). It manages state for pending operations (isPending), potential errors (error), and cancellation handling (isCancelled). Upon successful signup, it dispatches a login action and navigates the user to the home page ('/').

* UseLogout hook

This useLogout custom hook manages user logout functionality by updating the user's online status in Firestore (db.collection('users').doc(user.uid).update({ online: false })) and signing the user out (projectAuth.signOut()). It handles state for pending operations (isPending), potential errors (error), and cleanup with cancellation handling.

* UseLogin hook

This useLogin custom hook handles user login functionality using Firebase authentication (signInWithEmailAndPassword). It updates user online status in Firestore upon successful login (db.collection('users').doc(res.user.uid).update({ online: true })) and manages state for pending operations (isPending) and potential errors (error) with cancellation handling.

* UseAuthContext

The useAuthContext custom hook retrieves the authentication context (AuthContext) using useContext from React. It ensures that the context is available and throws an error if it's not found, indicating that useAuthContext must be used within an AuthContextProvider. It then returns the retrieved context for use in components.

* FireBase config

This code initializes Firebase with the provided firebaseConfig settings and exports the necessary Firebase services (db, projectAuth, projectStorage, timestamp) for Firestore, authentication, storage, and timestamp functionality. The Firebase services are configured and ready to be used within the application for interacting with Firestore, authentication, and storage functionalities.

* Course

The Course component displays information about available courses with buttons to enroll in each course. It includes details like course names ("Maths Certification" and "Science Certification"), their prices (free), and buttons to enroll (onClick events to navigate to specific course pages /course\_one and /course\_two).

* Course\_one

This React component CourseOne presents a math certification quiz with dynamic questions and answer input fields. It tracks user answers, calculates total time taken for completion (rounded to 1 decimal), and provides feedback based on correctness. Upon successful completion, it generates a certificate with random analytics (total time, attempts, rank, certificate ID). The component encapsulates quiz functionality and analytics presentation within a single container.

* Course\_two

This React component CourseTwo renders a science certification quiz with dynamic questions and answer input fields. It tracks user answers, calculates total time taken for completion, and displays feedback based on correctness. Upon successful completion, it generates a certificate with random analytics (total time, attempts, rank, certificate ID). The component encapsulates quiz functionality and analytics presentation within a single container.

* Navbar

The Navbar component handles navigation and authentication actions, including links for signup, login, and logout based on the user's authentication status. It also includes elements for redirecting to contact information and external course pages.

* Footer

The Footer component renders a footer section for a website, displaying information about the website and contact details. It includes a logo, a brief description, and contact information (email and phone). The footer also shows the copyright notice with the current year and website name.

* Contact

The Contact component renders a contact page with information such as email, phone number, and address. Users can reach out for inquiries or assistance using the provided contact details.

Setup Instructions

* Open the front-end in terminal and do npm install.
* After npm install run the command `npm run dev`.
* Open the link in chrom/edge/brave or any other browser.