Objective - To Simulate SSO login to a platform using Okta.

Status Quo -

1. We have a website, where you can click on login using okta.
2. The user will be redirected to the okta page and login. Once the login is completed the end user will be successfully logged in to the website and access the portal.

High-Level Architecture

Component	**Technology**
Frontend	React.js
Auth Provider	Okta
SSO Protocols	SAML & OIDC
Backend	Your server (optional for token handling)

Overview Terms

Terms	Meaning	Who will provide to whom
Client ID **Client Secret** (if backend used)	Required for OIDC authentication. Needed for token exchange in Authorization Code Flow	Okta admin to provide
Issuer URL	Usually: https://{yourOktaDomain}/oauth2/default	Okta admin to provide
Authorization Endpoint	Redirect here for login	Okta admin to provide
Token Endpoint	To exchange code for tokens (used by backend)	Okta admin to provide

Redirect URI(s)	Whitelisted URLs to redirect users after login	SP will provide and Okta admin to confirm after configuration
Scopes Allowed	Usually openid, profile, email (attribute)	Okta admin and SP both should decide and agree.
Redirect URI (callback)	List of URLs Okta will redirect to after login (e.g., https://yourapp.com/login/callback)	SP to share with Okta admin
Post Logout Redirect URI	Where to redirect after logout	Optional
App Domain	So the Okta admin can register your SPA domain	SP to provide to okta admin

Step 1: User opens your login page

Example:

- They click "Login with Okta" (or your custom SSO button)
- https://yourapp.com/login

Step 2: Redirect to Okta Authorization Endpoint

- Your app redirects the browser to Okta
- GET https://{yourOktaDomain}/oauth2/default/v1/authorize ?client_id=YOUR_CLIENT_ID
- This is the Authorization Endpoint. User logs into Okta (if not already logged in)

Step 3: Okta redirects back to your app

- After successful login, Okta redirects to your **Redirect URI (callback)**:
- https://yourapp.com/callback?code=AUTH_CODE&state=xyz123

Step 4: Your backend exchanges code for tokens -

POST https://{yourOktaDomain}/oauth2/default/v1/token

This is a **secure step** that happens server-side

Step 5: Okta responds with tokens

```
{
  "access_token": "....",
  "id_token": "....",   // contains user identity
  "refresh_token": "...", // optional
  "token_type": "Bearer",
  "expires_in": 3600
}
```

Step 6: Your backend verifies the ID token

- Decode the id_token (a JWT) and validate:
    - issuer matches https://{yourOktaDomain}/oauth2/default
    - Audience matches your **Client ID**
    - nonce matches the original request - A random value included in the initial `/authorize` request, and must match in the `id_token`

Step 7: Create session / send JWT to frontend

Your app can now:

- Start a session (cookie-based)
- Or pass a frontend-safe JWT (for SPA)