Libraries or packages used:

Wallet frontend:

React (react, react-dom):

The core library for building the user interface. It's used to create reusable UI components and manage the application's state and lifecycle.

React Router (react-router-dom):

Used for handling routing in the React application. It allows for navigation between different components, creating a single-page application experience.

Axios:

A popular HTTP client for making API requests. It's likely used to communicate with the backend server, sending and receiving data for credential operations.

QR Code Libraries (qrcode.react, react-qr-reader):

- qrcode.react: Used to generate QR codes, likely for displaying credential requests or offers.

- react-qr-reader: Implements QR code scanning functionality, probably used in the wallet to scan credential offers or presentation requests.

JWT Libraries (jsonwebtoken, jwt-decode):

- jsonwebtoken: Used for creating and verifying JSON Web Tokens, which are likely used for secure communication between the frontend and backend.

- jwt-decode: Helps in decoding JWTs on the client-side, useful for extracting information from tokens without verification.

CORS:

While typically a backend middleware, its presence in the frontend dependencies suggests it might be used in a development server or proxy setup to handle Cross-Origin Resource Sharing.

Testing Libraries (@testing-library/react, @testing-library/jest-dom, @testing-library/user-event):

These are used for writing and running tests for React components, ensuring the application behaves as expected.

React Scripts:

Part of Create React App, it provides scripts for running, building, and testing the React application.

Web Vitals:

Used for measuring and reporting on core web vitals, helping to monitor the application's performance.

Wallet Backend:

Express:

This is a minimal and flexible Node.js web application framework. It's used to create the server-side application for the wallet. Express handles routing, middleware integration, and HTTP request/response management. In your wallet backend, it likely manages endpoints for storing credentials, retrieving credentials, and handling authentication.

Axios:

A popular promise-based HTTP client for making API requests. In the wallet context, it's probably used to communicate with external services, such as credential issuers or verifiers. It might be used to fetch credential offers, submit credential requests, or send verifiable presentations.

CORS (Cross-Origin Resource Sharing):

This middleware is used with Express to enable cross-origin requests. It's crucial for allowing the wallet frontend (which might be served from a different origin) to communicate with the wallet backend API securely.

Jose:

This is a JavaScript implementation of JSON Object Signing and Encryption (JOSE) standards. In the wallet, it's likely used for cryptographic operations related to verifiable credentials. This could include signing verifiable presentations, verifying signatures on received credentials, or encrypting sensitive data.

JSON Web Token (jsonwebtoken):

Used for creating and verifying JSON Web Tokens. In the wallet context, it might be used for secure session management, authenticating API requests, or as part of the credential issuance and verification process.

QR Code Libraries (qrcode, qrcode.react):

- qrcode: A server-side library for generating QR codes. It might be used to create QR codes for sharing wallet information or for credential exchange protocols.

- qrcode.react: A React component for generating QR codes. Its presence in the wallet backend dependencies suggests it might be used in a server-side rendering context or as part of a full-stack application structure.

Issuer Application:

Issuer Frontend:

React (react, react-dom):

These are the core libraries for building the user interface of your application. React is used to create reusable UI components and manage the application's state and lifecycle. React DOM is specifically for rendering React components in the browser.

React Scripts:

This is a set of scripts from Create React App (CRA) that helps you develop, build, and run your React application. It abstracts away the complex configuration for webpack, Babel, ESLint, etc., allowing you to focus on writing your application code.

Testing Libraries (@testing-library/react, @testing-library/jest-dom, @testing-library/user-event):

These libraries are used for writing and running tests for your React components. They help ensure that your application behaves correctly and maintains its functionality as you develop.

Web Vitals:

This library is used for measuring and reporting on core web vitals, which are important metrics for understanding the user experience of your web application. It helps you monitor things like loading performance, interactivity, and visual stability.

Issuer Backend:

Express:

This is a minimal and flexible Node.js web application framework. In the issuer context, it's used to create the server-side application that handles credential issuance. Express manages routing, middleware integration, and HTTP request/response handling for the issuer's API endpoints.

Axios:

A promise-based HTTP client for making API requests. In the issuer, it might be used to communicate with external services, such as identity providers or credential registries.

Body-parser:

Middleware for Express that parses incoming request bodies. It's crucial for handling JSON payloads in API requests.

CORS (Cross-Origin Resource Sharing):

Middleware used with Express to enable cross-origin requests, allowing the issuer to interact with frontends or other services hosted on different domains.

Jose:

A JavaScript implementation of JSON Object Signing and Encryption (JOSE) standards. In the issuer, it's likely used for cryptographic operations related to verifiable credentials, such as signing issued credentials.

JSON Web Token (jsonwebtoken):

Used for creating and verifying JSON Web Tokens. In the issuer context, it might be used for secure communication or as part of the credential issuance process.

QR Code (qrcode):

A library for generating QR codes. The issuer might use this to create QR codes for credential offers or to facilitate the credential issuance process.

UUID (uuidv4):

Used for generating unique identifiers. In the issuer, this might be used to create unique IDs for credentials, offers, or sessions.

WebSocket (ws):

A library for implementing WebSocket connections. The issuer might use this for real-time communication with clients during the credential issuance process.

Verifier Frontend:

React (react, react-dom):

These are the core libraries for building the user interface of your application. React is used to create reusable UI components and manage the application's state and lifecycle. React DOM is specifically for rendering React components in the browser.

Axios:

A popular promise-based HTTP client for making API requests. In your frontend, it's likely used to communicate with your backend server, sending and receiving data for credential operations.

QR Code Libraries (qrcode.react, react-qrcode-reader):

- qrcode.react: Used to generate QR codes, likely for displaying credential requests or offers.

- react-qrcode-reader: Implements QR code scanning functionality, probably used in the wallet to scan credential offers or presentation requests.

React Scripts:

This is a set of scripts from Create React App (CRA) that helps you develop, build, and run your React application. It abstracts away the complex configuration for webpack, Babel, ESLint, etc., allowing you to focus on writing your application code.

Testing Libraries (@testing-library/react, @testing-library/jest-dom, @testing-library/user-event):

These libraries are used for writing and running tests for your React components. They help ensure that your application behaves correctly and maintains its functionality as you develop.

Web Vitals:

This library is used for measuring and reporting on core web vitals, which are important metrics for understanding the user experience of your web application. It helps you monitor things like loading performance, interactivity, and visual stability.

Verifier Backend:

1. **cors**: This library is used to enable Cross-Origin Resource Sharing (CORS) in your Express application. It allows your server to handle requests from different domains, making it possible for your web application to communicate with your backend API securely.
2. **express**: A minimal and flexible Node.js web application framework that provides a set of features for building web and mobile applications. It's widely used for setting up REST APIs and handling HTTP requests.
3. **node-jose**: A JavaScript library that provides functionalities for handling JSON Object Signing and Encryption (JOSE), including JSON Web Tokens (JWT), JSON Web Encryption (JWE), and JSON Web Signatures (JWS). It helps in implementing secure communication and data exchange.
4. **uuid**: This library is used to generate unique identifiers (UUIDs). It's often used to create unique IDs for objects, sessions, or transactions within your application, ensuring they are distinct and not easily duplicated.