Full-Stack Project Dependencies **kjopmann**

This report provides an overview of the dependencies and packages used in our full-stack project, which combines React for the front-end, Postgre SQL for the database, and Node.js for the backend. By understanding the key dependencies utilized in our project, we gain insights into the technologies and tools leveraged to build a robust and scalable application. The report outlines the purpose and significance of each dependency, highlighting their roles in enhancing development efficiency, managing state, securing data, and facilitating seamless communication between the client and server.

# Front End:

1. **"react":** React is a JavaScript library used for building user interfaces. We used React as the foundation of our front-end development, allowing us to create reusable and modular UI components.
2. **"react-dom":** React DOM is a package that provides the integration between React and the browser's DOM (Document Object Model). We used it to render our React components to the browser.

**npm install react-dom**

1. **"react-icons":** This package provides a collection of icons that can be easily used in React applications. We utilized it to enhance our user interface by including various icons for different functionalities.

**npm install react-icons**

1. **"react-redux":** React Redux is a library that provides a predictable state container for managing the application's state. We utilized it to manage the global state of our React application, enabling efficient data flow between components.

**npm install react-redux**

1. **"react-router-dom":** This package provides routing capabilities for React applications. We used it to implement client-side routing, allowing us to navigate between different pages or views in our application.

**npm install react-router-dom**

1. **"axios":** Axios is a popular HTTP client library for making API requests. We used it to send HTTP requests from our React application to interact with the backend server and retrieve data.

**npm install axios**

1. **"@reduxjs/toolkit":** Redux Toolkit is an official package by Redux that simplifies the process of managing Redux state. We utilized it to streamline the Redux setup and reduce boilerplate code in our React application.

**npm install @reduxjs/toolkit**

# Backend:

1. **"bcrypt":** Bcrypt is a library used for password hashing and verification. We used it to securely hash and compare passwords in our Node.js backend, ensuring the protection of user credentials.  
   **“npm install bcrypt”**
2. **"cors":** CORS (Cross-Origin Resource Sharing) is a mechanism that allows resources (e.g., APIs) on a web page to be requested from another domain outside the domain from which the resource originated. We used the CORS package to enable cross-origin requests in our Node.js backend.

**npm install cors**

1. **"dotenv":** Dotenv is a module used to load environment variables from a .env file into process.env. We used it to store sensitive information, such as database credentials, in a separate file and access them securely within our Node.js application.

**npm install dotenv**

1. **"express":** Express is a popular web application framework for Node.js. We utilized it to build the backend server, handle routing, and handle HTTP requests and responses.

**npm install express**

1. **"express-async-handler":** This package simplifies error handling in asynchronous Express routes. We used it to handle asynchronous operations in our routes and streamline error handling.

**npm install express-async-handler**

1. **"jsonwebtoken":** JSON Web Token (JWT) is a standard for creating and verifying tokens. We used the jsonwebtoken package to implement token-based authentication and authorization in our Node.js backend

**npm install jsonwebtoken**