PROJECT TITLE :sonarhub

FRONTEND : reactjs

Vite - It's fast, simple, and widely adopted!

Vite is a build tool that aims to provide a faster and leaner development experience for modern web projects.

command:npm create vite@latest my-app -- --template react-ts

Variant : react , ts+SWC

Installed tools for frontend:

- npm install react-hook-form bootstrap react-bootstrap @types/react @types/react-dom @types/react-router-dom
- npm install react-router-dom @types/react-router-dom
- npm i --save-dev @types/node

LOGIN - SIGNUP FORM

- **react-hook -form:**React Hook Form takes a slightly different approach than other form libraries in the React ecosystem by using uncontrolled inputs with ref instead of depending on the state to control the inputs

  npm install react-hook-form

  Strict-type-check: This spread operator syntax is a new implementation to the library that enables strict type checking in forms with TypeScript.

  <input type="text" name="firstName" {...register('firstName')} />

  Old version use ref:

  <input type="text" name="firstName" ref={register} />

Diff between apollo-client vs @apollo/client

Apollo Client is the fully-featured, production ready caching GraphQL client for every UI framework and GraphQL server.-Apollo Client is designed for building complex, data-driven UIs in modern web applications.

`apollo-client` is the legacy version of Apollo Client.

`@apollo/client` is the current, actively maintained version of Apollo Client.

+Key differences:

`@apollo/client` has improved performance and stability.

`@apollo/client` has better TypeScript support.

`@apollo/client` is the recommended version for new projects.

Backend:

Installed tools:

- npm install express apollo-server-express graphql type-graphql typeorm reflect-metadata pg passport passport-github2 express-session cors dotenv
- npm install typescript ts-node nodemon @types/node @types/express @types/cors @types/passport @types/express-session --save-dev
- npm install sonar-scanner sonarqube-scanner
- npm install --save-dev ts-node
- npm i tsc
- tsc -init
- npm install @apollo/server
- npm install tsconfig-paths --save-dev --legacy-peer-deps
- npm install cross-env --legacy-peer-deps
- npm install --save-dev @types/express @types/cors @types/jsonwebtoken @types/bcryptjs @types/passport @types/passport-github2 --legacy-peer-deps
- npm install --save-dev @types/express-session --legacy-peer-deps

- npm install chart.js react-chartjs-2
-
-
-

Migrations:

- npm run generate:migration "src/database/migrations/AddedUserEntity"
- npm run start:local

env:

node -e "console.log(require('crypto').randomBytes(32).toString('hex'))"

3fb13f6477f886bdbcbfa187211d570f69fdadfcbb15874072ccc8edd1e82d15

```
DB_NAME = SonarHub

DB_HOST = localhost

DB_PORT = 5432

DB_USERNAME = postgres

DB_PASSWORD = 1234

GITHUB_CLIENT_ID=Ov23liwVPCtND4vUVfuG

GITHUB_CLIENT_SECRET=837e566dcc3abbd9ca2d0013e9b52dd651c16318

JWT_SECRET=d7380772bc0d4947ba0fb8b90e61ef22cd83bd7361d8802607df17594bfc3a50

SESSION_SECRET=3fb13f6477f886bdbcbfa187211d570f69fdadfcbb15874072ccc8edd1e82d15

BASE_URL=http://localhost:4000

FRONTEND_URL=http://localhost:5173

EMAIL_USER=arockia.santhiya@tringapps.com
```

node -v

v22.14.0

PS C:\sonarHub-backend> npm -v

10.9.2

https://docs.github.com/en/graphql/reference/queries

docker run -d --name sonar -p 9000:9000 sonarqube

npx ts-node -e "import dataSource from './src/database/data-source'; dataSource.initialize().then(() => console.log(dataSource.entityMetadatas.map(e => e.name))).catch(console.error);"

Tsc

del /s /q *.js

Why does `tsc` create a `.js` file for each `.ts` file?

When you run the TypeScript compiler (`tsc`), it transpiles each `.ts` file into a corresponding `.js` file because TypeScript itself is not natively understood by Node.js or browsers. The `.js` files are necessary for execution since JavaScript is the language that the runtime environment (Node.js) understands.

If you want to **compile all TypeScript files into a single JavaScript file**

select * from user_activity_logs;

SELECT * FROM information_schema.tables WHERE table_name = 'user_activity_logs';

Get-ChildItem -Path . -Recurse -Include *.ts | Select-String -Pattern 'import.*project.entity'

Why We Need `useMemo`:

The `useMemo` hook in React is used to **optimize performance** by **memoizing** the value of a computed variable. In your case:

This means **`sonarIssues` will only be recalculated when `data` changes**.

Without `useMemo`, every render of the component would **recompute** `sonarIssues`, even if `data` hasn't changed.

Sonar:
npm install -g sonarqube-scanner
sonar-scanner -Dsonar.host.url=http://localhost:9000 -Dsonar.token=YOUR_NEW_TOKEN

git rev-parse --is-inside-work-tree
true

```
sonar.projectKey=backend-ts
sonar.projectName=Backend TypeScript
sonar.sourceEncoding=UTF-8
sonar.language=ts
sonar.sources=src
sonar.exclusions=**/*.spec.ts,**/*.test.ts,node_modules/**,dist/**
sonar.typescript.lcov.reportPaths=coverage/lcov.info
sonar.host.url=http://localhost:9000
sonar.token=sqa_36cdaca266ea6b9836f13c07c2293798e06bfa04
```