Diario di bordo Progettino Autenticazione

12.11.2022

Creati progetti backend (nodejs express) e frontend (react).

14.11.2022

Installato bootstrap e react-bootstrap. Npn install.

Creati i files componenti login e registration js. Forms presi da react-bootstrap.

<https://react-bootstrap.github.io/forms/overview/>

Uso un hook per decidere se mostrare la pagina di login oppure quella di registrazione; ancora da programmare e testare. È un’idea, successivamente sarà probabilmente da modificare.

* Scrivere i test per login e registration!!!

15.11.2022

Cercato il come testare da

* <https://it.reactjs.org/docs/testing-recipes.html>
* <https://www.youtube.com/watch?v=6wbnwsKrnYU>
* <https://training.fabiobiondi.io/2019/12/28/react-typescript-unit-test-react-testing-library/>

Scritti dei test. Vado a verificare che i componenti vengano renderizzati correttamente ricercandone le parti nello schermo.

‘It’ e ‘test’ sono essenzialmente lo stesso “comando” <https://stackoverflow.com/questions/45778192/what-is-the-difference-between-it-and-test-in-jest> .

Attenzione: getByPlaceholderText() può causare problemi se esistono più placeholder con il medesimo testo o parte di esso (es. ‘Password’ e ‘Confirm password’). Infatti <https://testing-library.com/docs/queries/byplaceholdertext/> suggerisce di usare in alternativa ‘getByLabelText’, o quantomeno indica che ‘getByPlaceholderText’ non ne è un buon sostituto.

Per eseguire i test si impiega il comando **npm test**.

Rifattorizzato: creata la cartella “\_\_test\_\_” allo stesso livello della cartella “components”.

Il button con variazione ‘link’ non sembra essere recuperabile tramite getByRole (per recuperare il button <https://stackoverflow.com/questions/58408178/query-a-button-with-specific-text> ). Allora ho impiegato getByText <https://stackoverflow.com/questions/57827126/how-to-test-anchors-href-with-react-testing-library> .

16.11.2022

Validazione dei form in react con react-hook-form: <https://www.npmjs.com/package/react-hook-form> , <https://react-hook-form.com/> . Si tratta di una libreria da quasi 2.7 mio. di download settimanali.

Nota: in react per le proprietà si procede come segue, passando un oggetto <https://www.w3schools.com/react/react_css.asp> .

Palette di colori html <https://www.w3schools.com/colors/colors_names.asp> .

Inoltre sembra intelligente impiegare uno schema sfruttando Yup così da evitare di inserire nel form le varie validazioni. npm install @hookform/resolvers yup

Refactoring nello schema yup.

Specificare dei messaggi di errore <https://stackoverflow.com/questions/60356911/how-to-change-default-error-text-in-yup-formik> .

Aggiunti altri campi al form di registrazione: nome, cognome, descrizione di sé stessi, data di nascita (con datepicker (<https://www.npmjs.com/package/react-datepicker>) e possibilmente un’immagine ().

Validazione dei campi presenti.

Devo capire come validare la data e come impostare il datepicker affinché sia più immediato tornare indietro nel tempo (ora andrebbe di mese in mese, ma già per me è lunga tornare indietro rapidamente e sono nato nel ’93…).

18.11.2022

Angelo sembra apprezzare i nostri sforzi!

Installare Sequelize, bcrypt, jsonwebtoken e per lavorare con il file .env.

19.11.2022

Inserito “proxy”: <http://localhost:3000> nel package json del frontend in modo da collegarlo al backend.

Ora, si potrebbe impiegare uno useEffect per seguire la fetch verso l’api del backend, ma sembra leggermente più laborioso rispetto all’impiego di axios (37mio di download settimanali) <https://www.npmjs.com/package/axios> .

Tuttavia si segnala su cve un problema risalente al 2021 <https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2021-3749> in cui si parla di “axios is vulnerable to Inefficient Regular Expression Complexity”. Ossia “The product uses a regular expression with an inefficient, possibly exponential worst-case computational complexity that consumes excessive CPU cycles.” Da <https://cwe.mitre.org/data/definitions/1333.html> .

Problema del collegamento, forse dovuto alle politiche CORS, scarico cors <https://www.npmjs.com/package/cors> .

Dopo diversi inconvenienti, problemi, Axios funziona: frontend sembra collegato con il backend.

// Destruct necessary in order to manage the inputs/forms validation

const { register, handleSubmit, formState: { errors } } = useForm({ resolver: yupResolver(schema)});

// onSubmit function call on the nodejs backend

const onSubmit = data => {

axios.get("/login").then(response => {

console.log(response.data);

})

};

Le righe di codice di Axios sono state inserite nella funzione onSubmit presa grazie a yup.

Installati bcrypt, Sequelize, nodemon, jsonwebtoken, dotenv (per lavorare con il file .env) <https://www.npmjs.com/package/dotenv> .

Creato il database: due tabelle, verifications per la verifica della mail e user per gli utenti. <https://www.tutsmake.com/node-js-express-mysql-email-verification-tutorial/>