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/>

22.11.2022

Scritti i modelli in Sequelize di User e Verification.

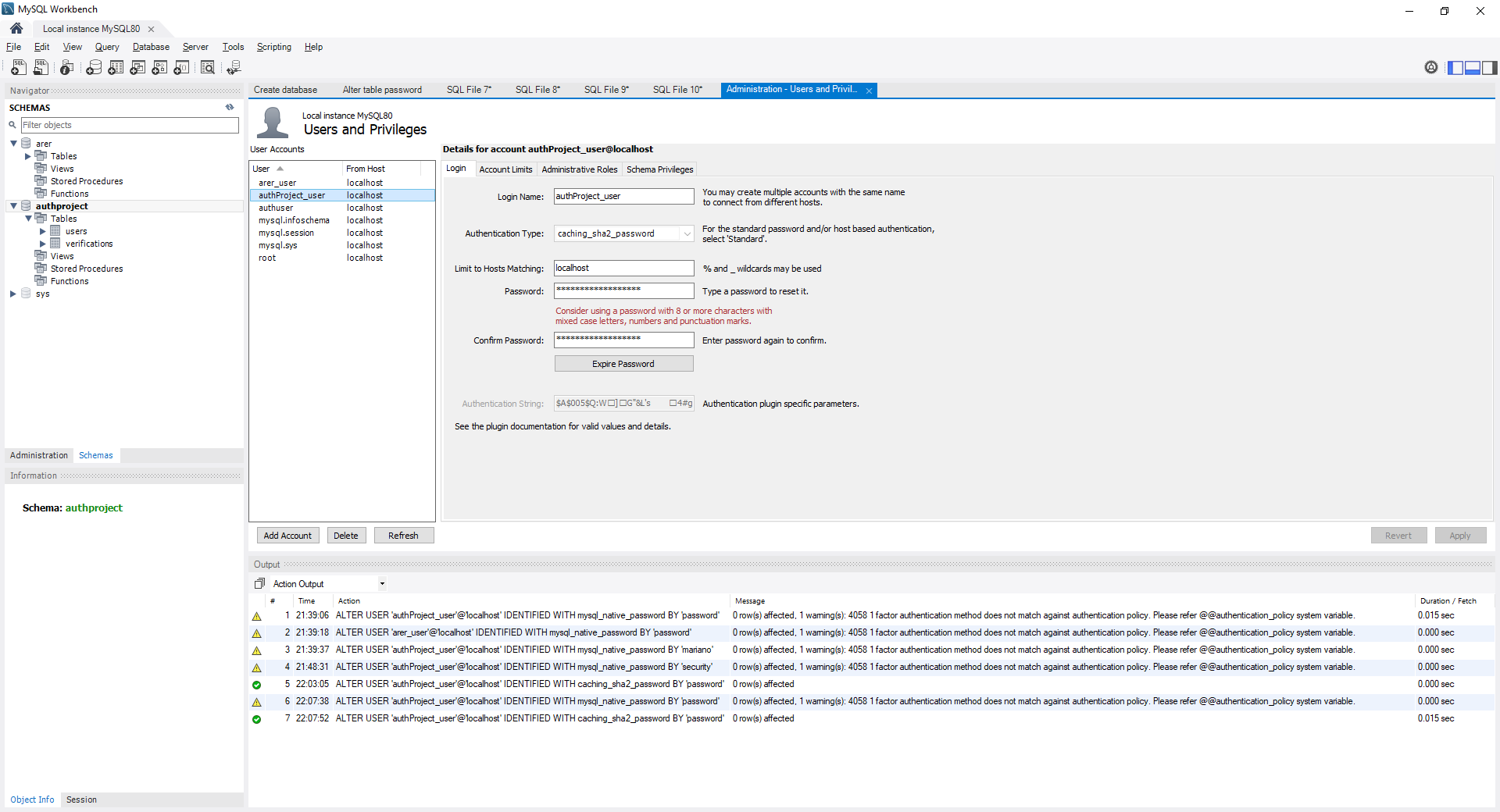
Filtrate nel Datepicker le date future rispetto a oggi (new Date()). Bisogna velocizzare il passaggio alle date passate per ora limitato al salto di mese in mese. E aggiungere il recaptcha.

Sistemo il backend.

Non riesco a collegare sequelize al database…

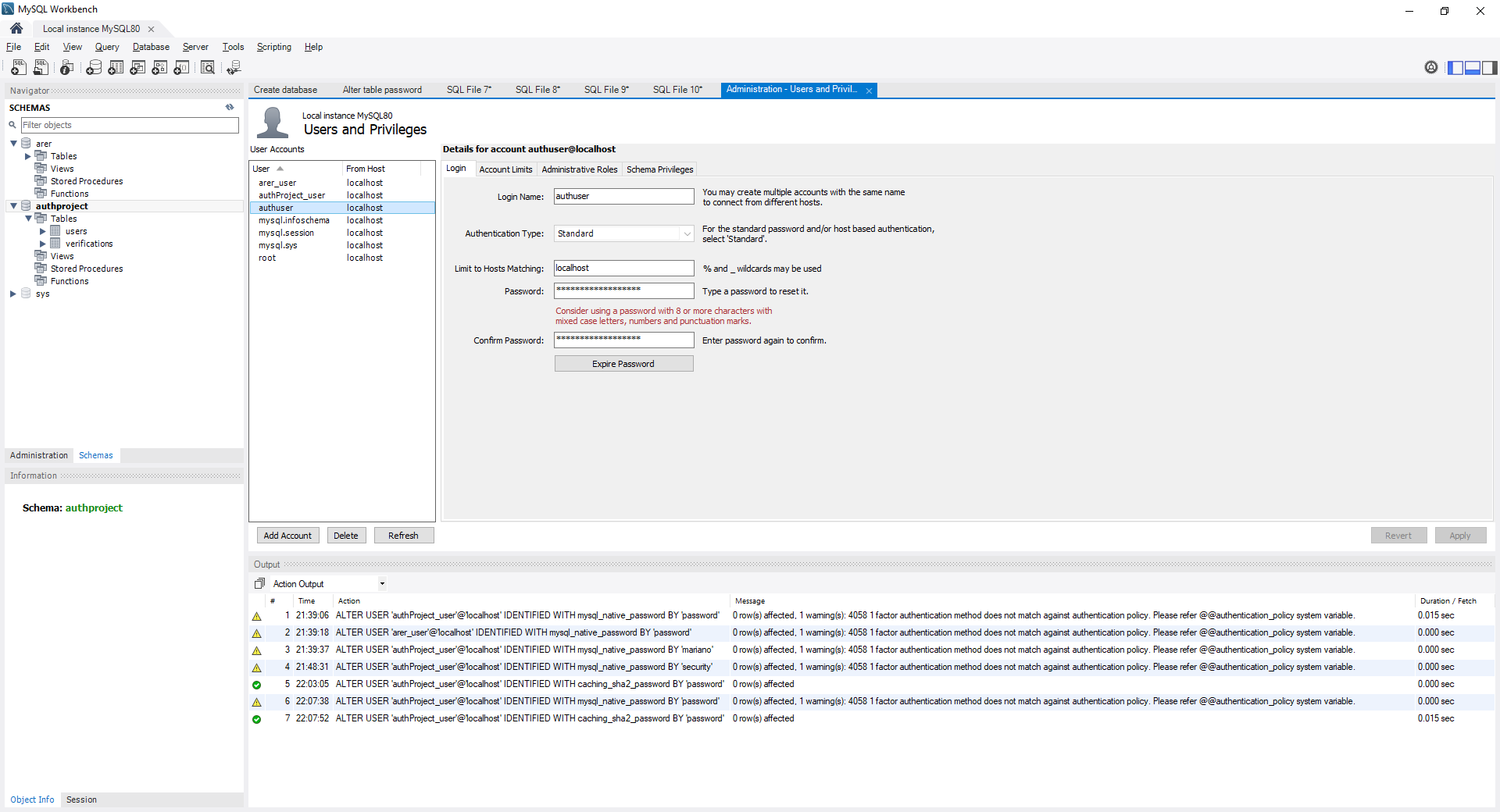


Il problema è stato risolto. Inserendo l’utente root e la rispettiva password il collegamento avveniva. Rispetto all’utente arer del progetto di diploma, l’utente che si cercava di autenticare



Aveva un tipo di autenticazione differente da quella standard.

Conseguentemente ho creato un utente ulteriore impiegando il tipo di autenticazione standard. Facendo partire il server non si sono verificati errori: Sequelize si connette con MySQL.



23.11.2022

Inserire la visualizzazione dell’utente loggato (card). Kitchen sink da <https://react-bootstrap.github.io/components/cards/> .

Creata la card utente, ma è da testare.

Passare al routing anche lato client <https://www.npmjs.com/package/react-router-dom> . Si cambia dalla single page application al routing.

Docs <https://v5.reactrouter.com/web/guides/quick-start> .

<https://www.geeksforgeeks.org/reactjs-router/#:~:text=React%20Router%20is%20a%20standard,how%20the%20React%20Router%20works>. Indica che Switch non è più un export della libreria ma è Routes.

Effettuato il routing

* “/” 🡪 login
* “/register” 🡪 form di registrazione
* “/user” 🡪 la scheda utente

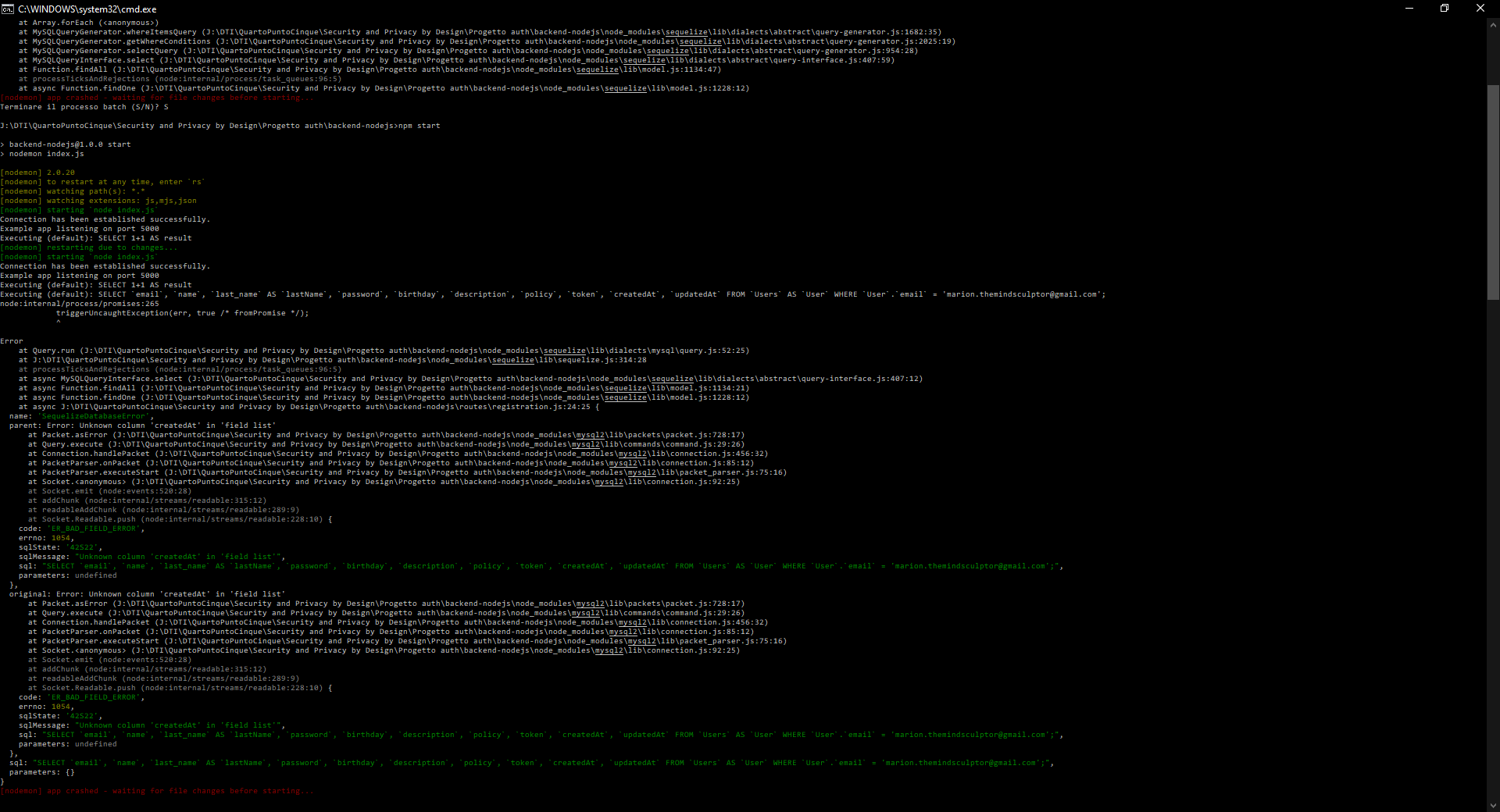
24.11.2022

Sistemo la registrazione. Probabilmente, per praticità, il tutto verrà poi gestito da una route unica; magari user.

Inserito il controllo per il match della password <https://www.positronx.io/add-confirm-password-validation-in-react-with-hook-form/> .

Ora scrivo la re-direzione dopo la corretta registrazione. <https://stackoverflow.com/questions/53900739/redirection-using-axios-and-react> . Installato react-router per importare Redirect.

Ho sistemato il post, in parte, l’errore generato è ora differente a livello di server



Sembra che sia dovuto al createdAt e updatedAt. Devo capire come risolvere, ossia come inserire automaticamente le date.

Ho creato le colonne nella tabella users in mysql workbench.

Nonostante l’inserimento delle colonne nel db l’errore persiste.



Provo anche a creare una verification. Continua a esserci il problema, allora ho disattivato il timestamps per l’intero database <https://stackoverflow.com/questions/20386402/sequelize-unknown-column-createdat-in-field-list> .

Così sembra funzionare, inoltre birthday nel db è stata cambiata in datetime da timestamp, per il modello con sequelize (<https://sequelize.org/docs/v6/core-concepts/model-basics/#data-types> ). New Date() crea l’istanza anche dell’orario, ma con toDateString() dovrei ottenere unicamente la data selezionata (<https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Date/toDateString> ), questo sembrerebbe funzionare meglio, testato in un interprete online (<https://stackoverflow.com/questions/2013255/how-to-get-year-month-day-from-a-date-object> ).

Redirect ora funziona <https://www.geeksforgeeks.org/how-to-redirect-to-a-relative-url-in-javascript/> .

25.11.2022

Gestito il login a livello di codice, da testare. In ordine: si controlla l’esistenza dell’utente, si controlla che l’account sia verificato, si controlla che la password sia corretta. Se tutti questi check sono corretti si crea e si invia il token.

Mi dedico al frontend per gestire il componente di login e i componenti di risposta. Devo aggiungere l’oggetto da assegnare al post.

Al json di risposta dal backend in caso di errori al momento del login ho aggiunto un campo legato al problema (se è legato alla password o alla verifica) poiché entrambi ritornerebbero lo stesso codice di errore 🡪 risolto: non eseguivo il send sulla risposta della route…

26.11.2022

Dopo il login, provo a inserire una semplice ridirezione a seconda della condizione. Nel componente App inserisco le route che mi servono per la ridirezione con i relativi componenti e i props necessari.

Hooks ed eventuali funzioni:

isUserLog (true/false), un hook per il messaggio e la relativa funzione per settarlo, un hook per il problema e la relativa funzione per settarlo.

Gli hook non funzionano, quindi ho creato delle pagine ad hoc.

Inoltre ho risolto il perché non si vedeva console.log() nella console del browser web. Axios divide quella che è la risposta da quello che è l’errore.

.then(response => {…} ) 🡪 per la risposta

.catch(error => {…} ) 🡪 per gli errori (es. 401, 404, ecc…)

Ora cerco di gestire l’invio della email.

Mischio un pochino <https://www.tutsmake.com/node-js-express-mysql-email-verification-tutorial/> , <https://www.youtube.com/watch?v=76tKpVbjhu8> e il relativo codice su github <https://github.com/benawad/graphql-express-template/tree/25_confirmation_email> .

Installo il nodemailer <https://www.npmjs.com/package/nodemailer> con oltre 2 mio di download settimanali.

Creata la route verifymail che gestisce il link di verifica che si invia via mail:

router.get('/verifymail/:token', async (req, res) => {

try {

const { user: { email } } = jwt.verify(req.params.token, process.env.MAIL\_TOKEN\_SECRET);

const verification = await Verification.findOne({where: {token: req.params.token}});

// Update verify flag

verification.verify = '1';

// Save updated verification element

verification.save();

} catch (e) {

res.send('error');

}

return res.redirect('http://localhost:3000/');

});

Creato un account mail volante su ethermail. La configurazione su nodejs è:

const transporter = nodemailer.createTransport({

host: 'smtp.ethereal.email',

port: 587,

auth: {

user: 'mitchel.schiller69@ethereal.email',

pass: 'YXk5E5MZJB74qdnHdt'

}

});

Qualora si andasse in produzione potrebbe essere corretto usare questa configurazione per distinguere test e produzione

var mailConfig;

if (process.env.NODE\_ENV === 'production' ){

// all emails are delivered to destination

mailConfig = {

host: 'smtp.sendgrid.net',

port: 587,

auth: {

user: 'real.user',

pass: 'verysecret'

}

};

} else {

// all emails are catched by ethereal.email

mailConfig = {

host: 'smtp.ethereal.email',

port: 587,

auth: {

user: 'ethereal.user@ethereal.email',

pass: 'verysecret'

}

};

}

let transporter = nodemailer.createTransport(mailConfig);

Provo la verifica della mail.

Salta fuori un errore: “error: self signed certificate in certificate chain”.

<https://stackoverflow.com/questions/45088006/nodejs-error-self-signed-certificate-in-certificate-chain> segnalano una soluzione, ma la soluzione viene smentita qui <https://github.com/nodemailer/nodemailer/issues/406> .

Sempre all’interno dell’ultimo link si suggerisce la seguente soluzione da aggiungere ai parametri del transporter.

tls: {

rejectUnauthorized: false

}

Effettivamente l’errore è sparito. Verificherò che l’email sia arrivata, inviata alle 22.46.