<https://www.youtube.com/watch?v=TiSGujM22OI&list=PLC3y8-rFHvwi1AXijGTKM0BKtHzVC-LSK>

ReactTS:

* TS > JS (adv: code scalability, static typing, better documentation, easier maintenance & code refactoring)

(dezadv: write more code, compiler keeps complaining about errors => more time to spend on them)

but think on the long run, ts is better and you’ll be more productive with the experience gained

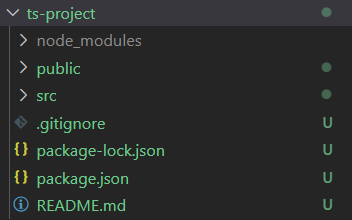
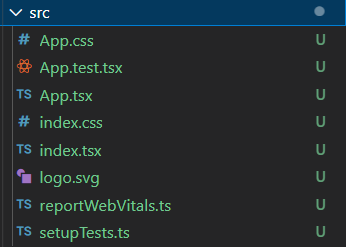
* Typescript is a superset of javascript, which lets you add type information to your code.
* typescriptul precizeaza tipul concret al datei => This then lets you **find bugs** in your code **quicker**, bc your IDE and build process can check the types to see if you've made mistakes.

NPX = npm package runner which gets installed when installing node => merge direct sa-i dat npx create-react-app fara a instala npm package global

NPM = intai se instaleaza global npm package (npm install create-react-app -g) si apoi poti genera proiectul (create-react-app <project\_name>) => trb updatat constant pachetul npm

* folosim npx ca e mai usor si la obiect

*Pas 1: npx create-react-app proiect --template typescript*

spre deosebire de tutorial: am package-lock.json, nu am tsconfig.json si nici yarn.lock, iar in src nu am env.d.ts

*npm start*

Obs: cand aplicatia ruleaza, modificarile apar automat, in real time in pagina

*Pas 2: creare folder components si fisier .tsx in el*

* ce trb pus prima data intr-un fisier.tsx?

export const Greet = () => {

    return (

        <div> <h2> Welcome to my page! </h2></div>

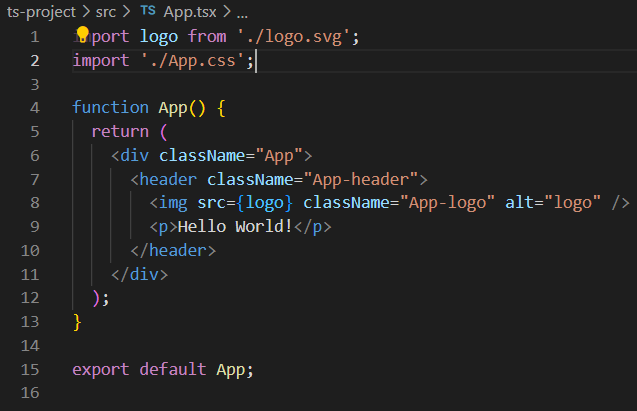
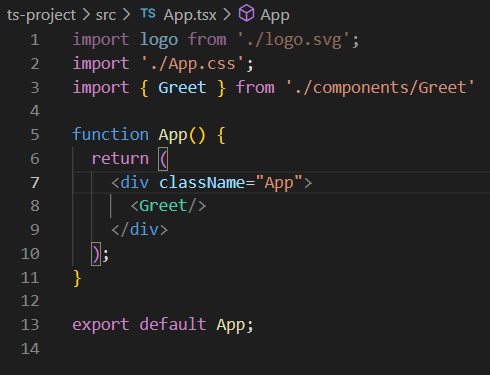
    )

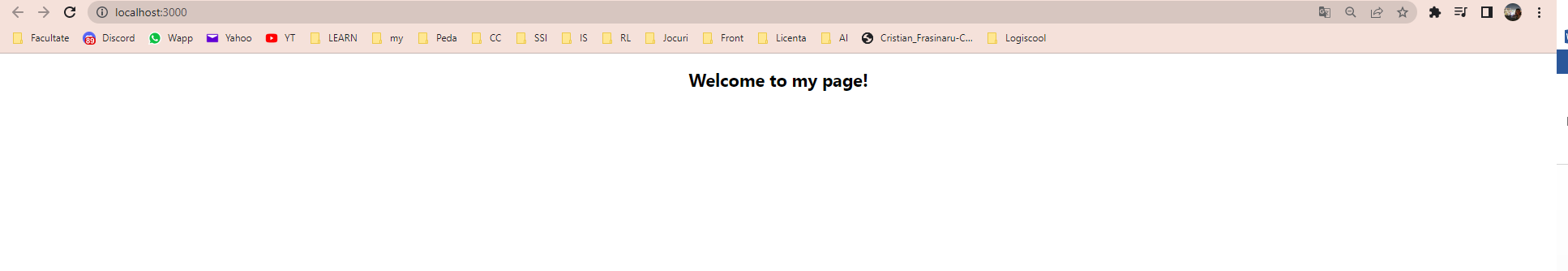
}

spre deosebire de fisier .js, nu are nev de import, iar functia se scrie diferit

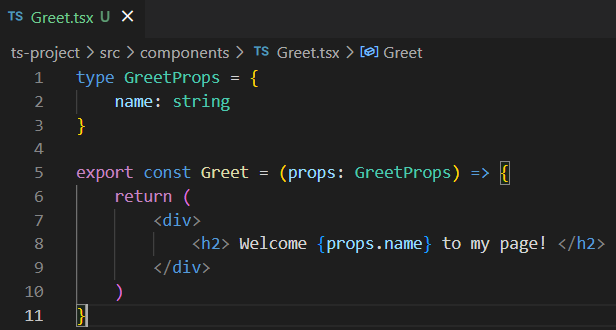
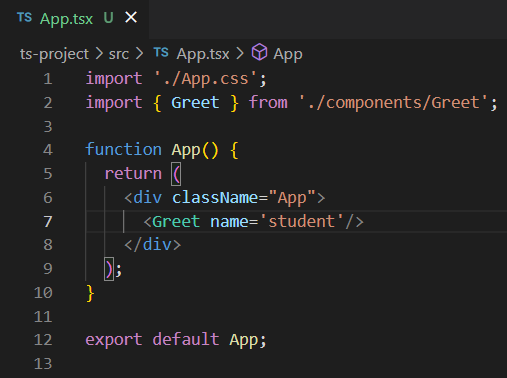
**rfce** + enter/tab => introduce template-ul de functie gen App

*Pas 3: include the .tsx file in App.tsx component*

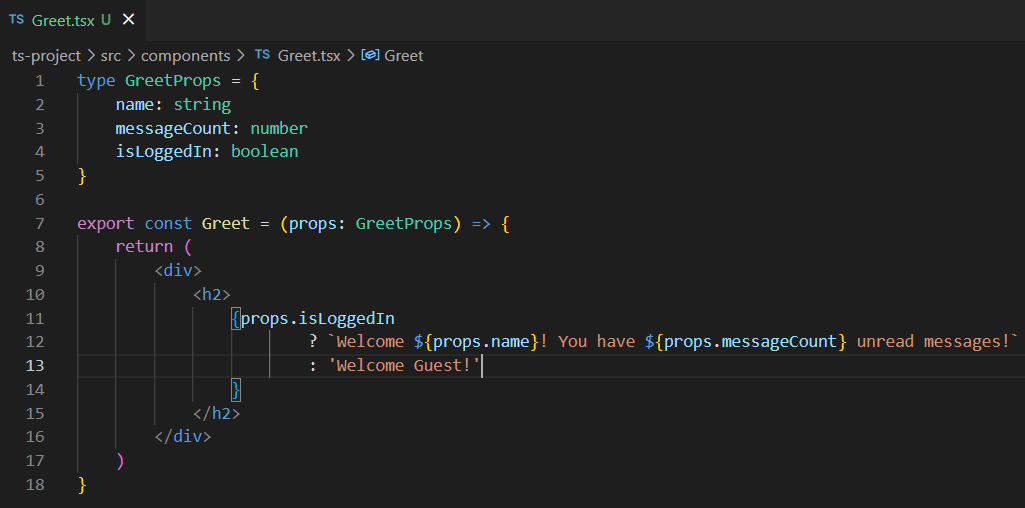
from: to:

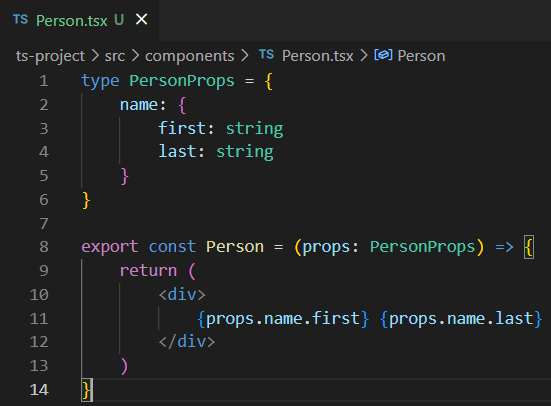
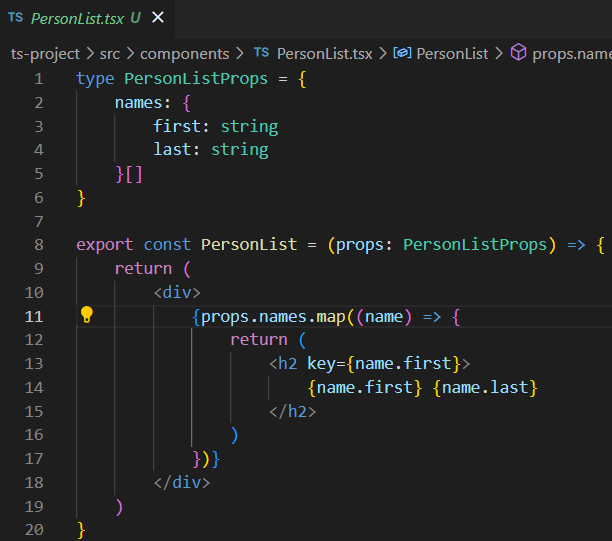
* 

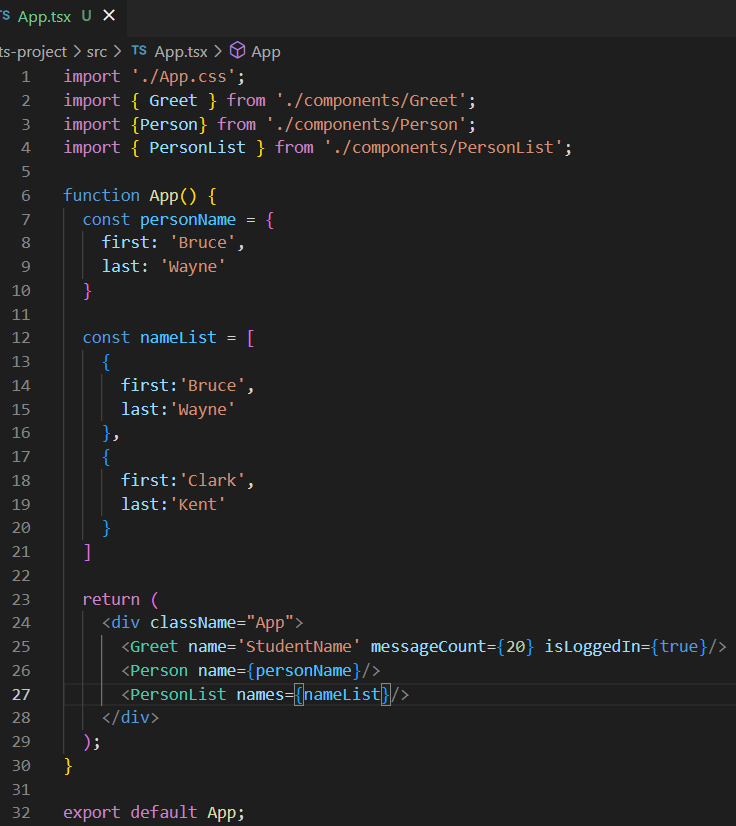
*Pas 4: typing props*

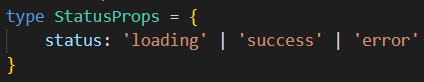
 

**Basic props examples:**

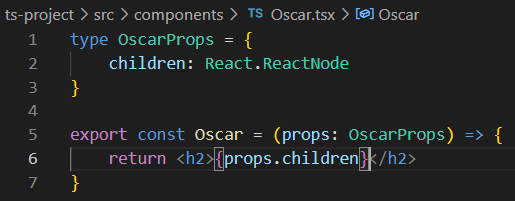
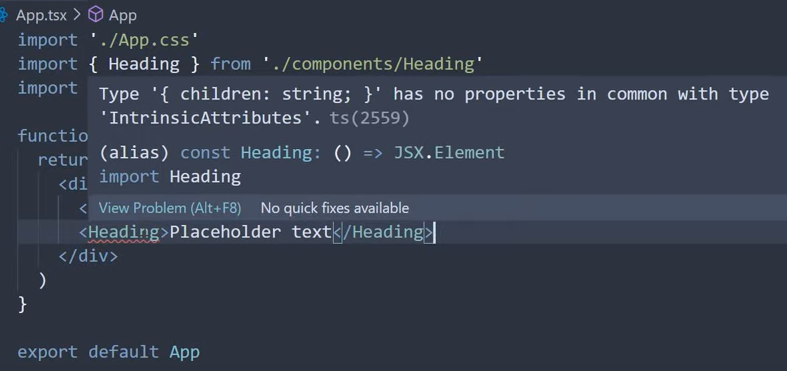


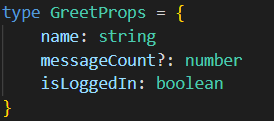
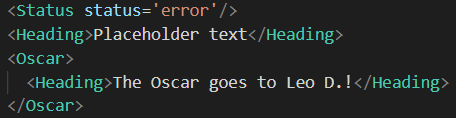


**Advanced props examples:**

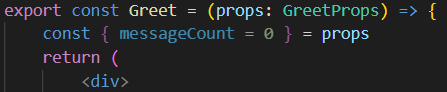
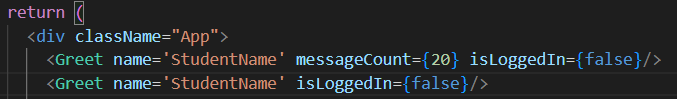
props type that does not allow for others strings than these:

solving an error with children props: 

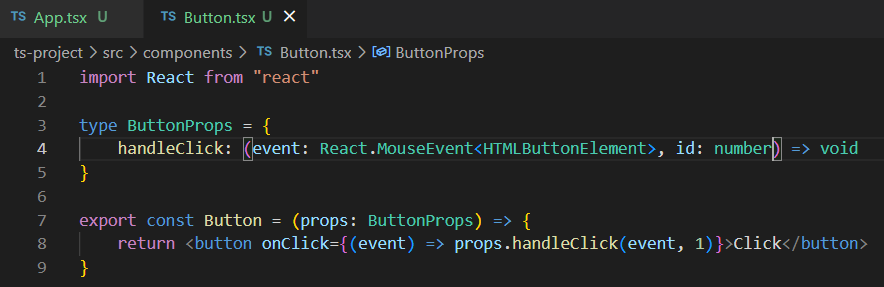
React.ReactNode children props type:

In App.tsx: 

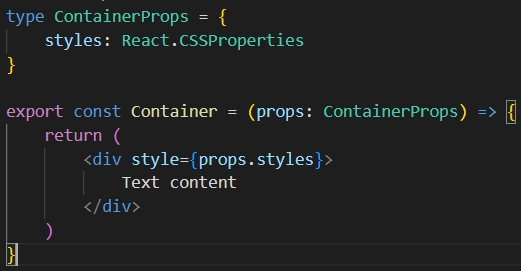
optional props:

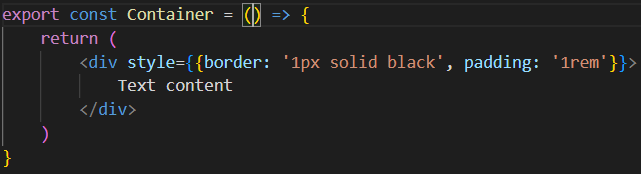
* putting ? when declaring it:
* in plus, initializing it with 0 when not sent in props: 
* 

**Event props:**

handleClick: 

handleChange: in input element

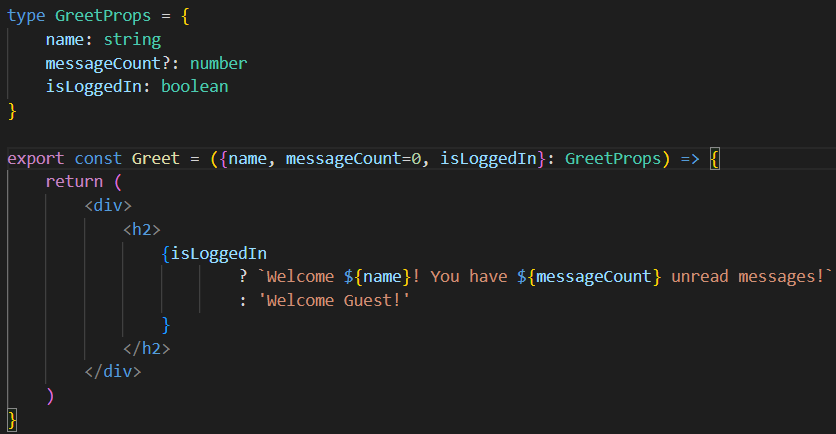
**Style props:**

in loc de:  punem:

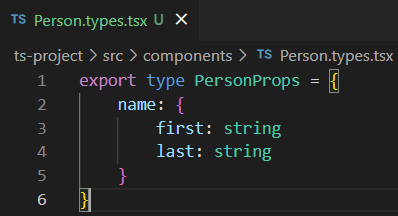
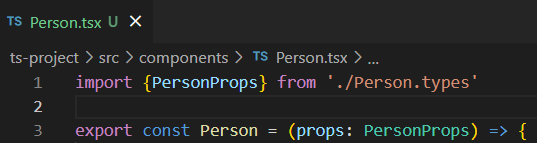
* React.CSSProperties

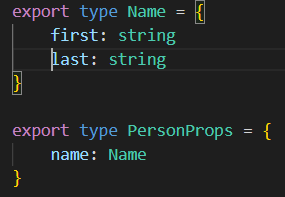
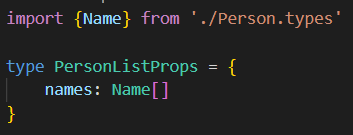
**Tips**:

* props dat ca parametru se poate inlocui cu {proprietatile sale} ex:



* move the types in separate file bc in complex project you’ll have a comp with many many types and it’s not ok to put them above the return function of that component

sau:  => 

cut from App.tsx

function App() {

  const personName = {

    first: 'Bruce',

    last: 'Wayne'

  }

  const nameList = [

    {

      first:'Bruce',

      last:'Wayne'

    },

    {

      first:'Clark',

      last:'Kent'

    }

  ]

  return (

<div className="App">

      <Greet name='StudentName' messageCount={20} isLoggedIn={false}/>

      <Greet name='StudentName' isLoggedIn={false}/>

      <Person name={personName}/>

      <PersonList names={nameList}/>

      <Status status='error'/>

      <Heading>Placeholder text</Heading>

      <Oscar>

        <Heading>The Oscar goes to Leo D.!</Heading>

      </Oscar>

      <Button handleClick = {(event, id) => {

        console.log('Button clicked', event, id)

      }}/>

      <Input value='' handleChange={event => console.log(event)}/>

      <Container styles={{border: '1px solid black', padding: '1rem'}}/>

    </div>

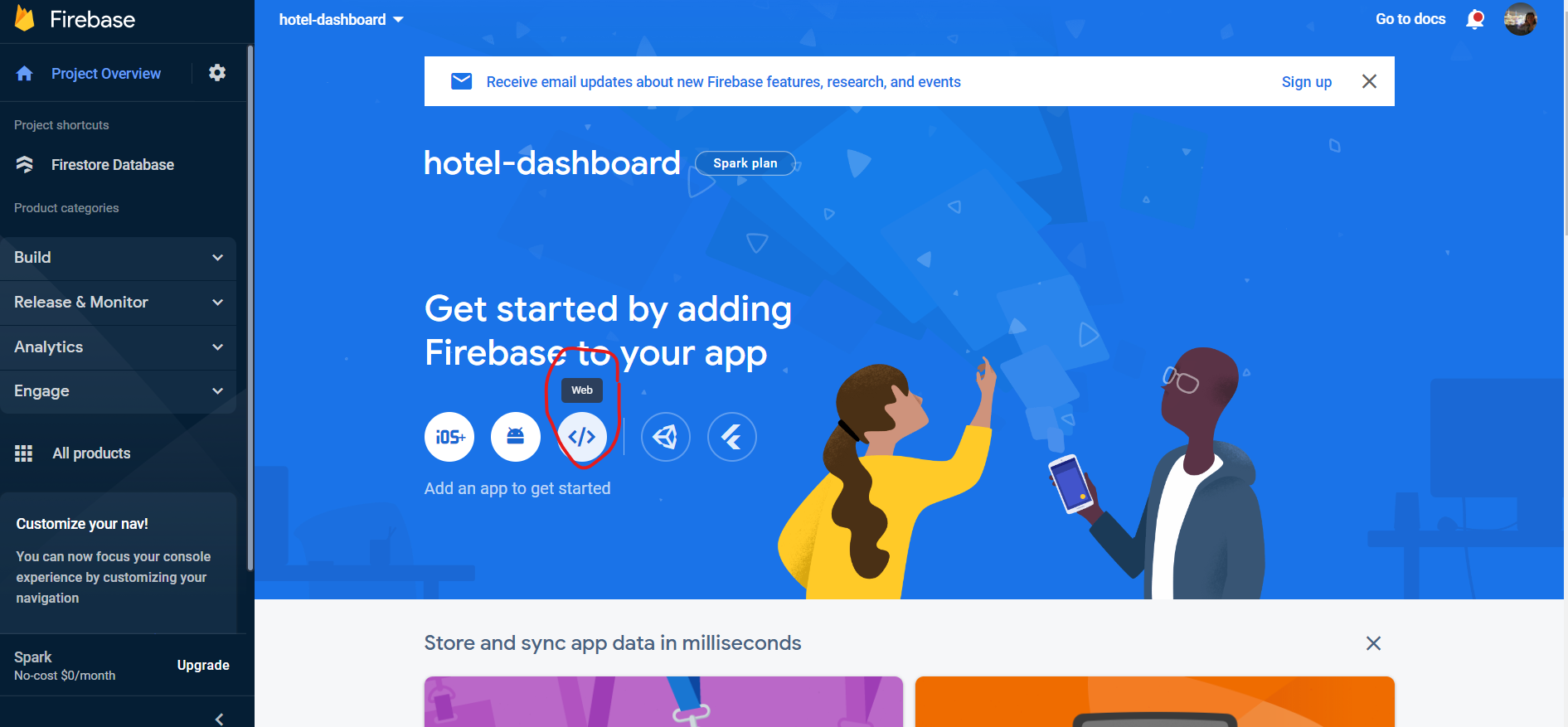
);

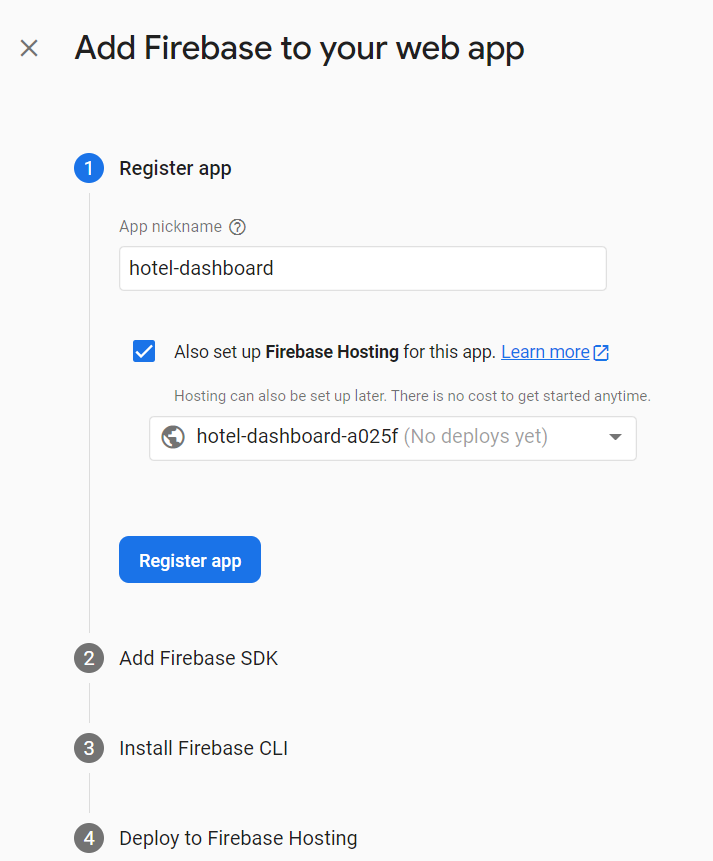
}

export default App;

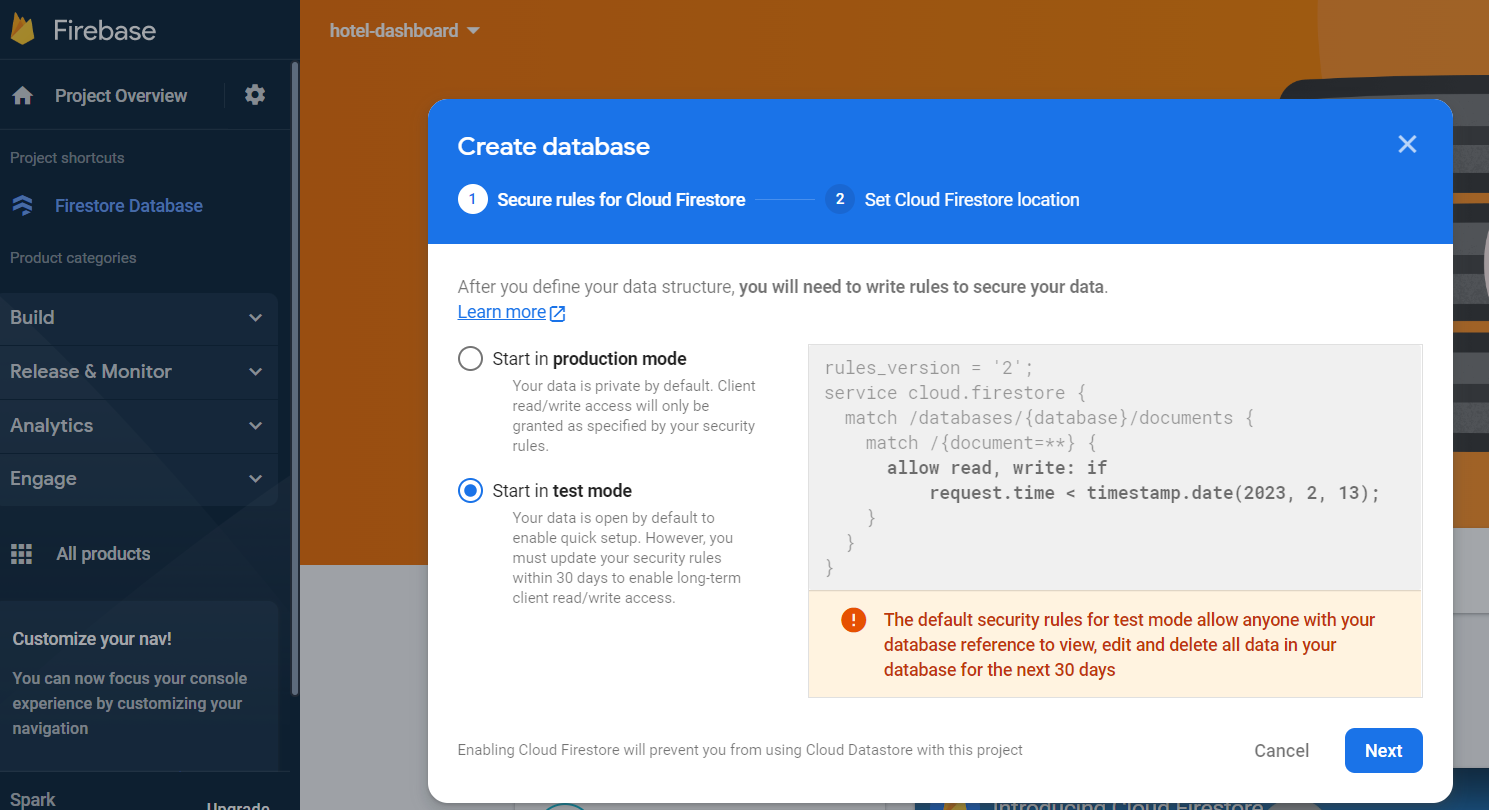
***Pas 5: firebase***

* create project : go to console, add project: hotel-dashboard, default firebase account
* click pe web



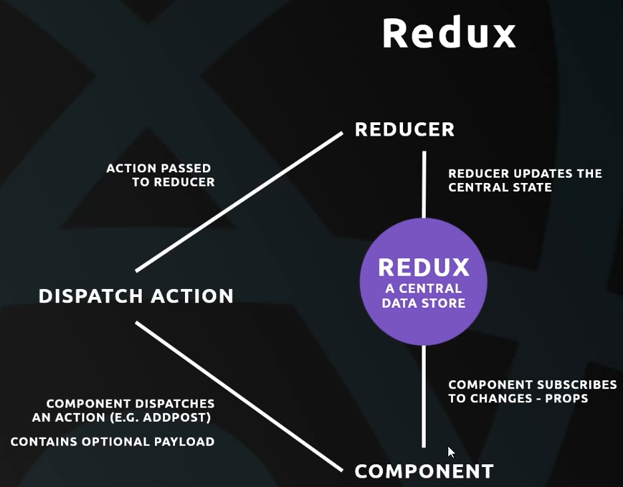


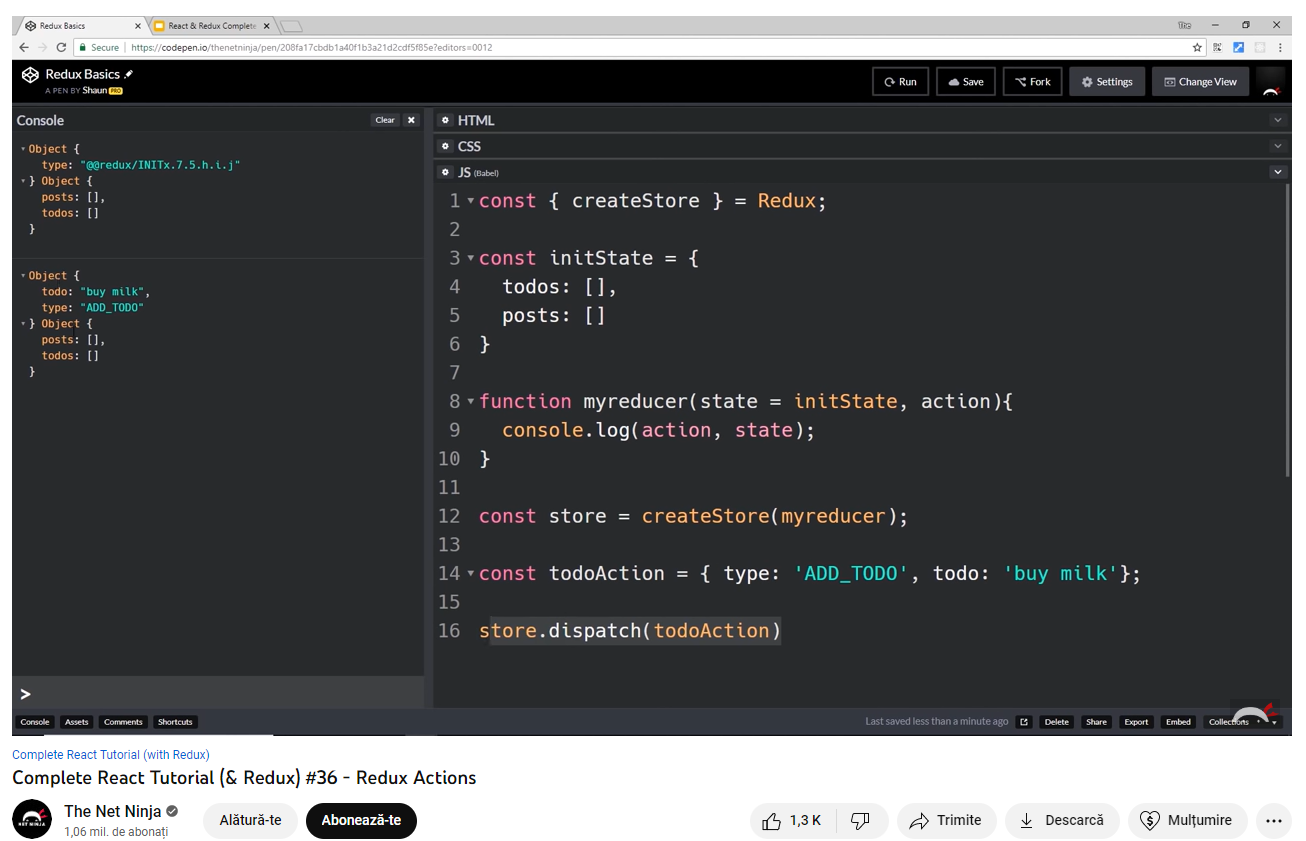
* create lib folder in src, create firebase.ts file in lib folder and paste the code from add firebase sdk and also run commands from here and install firebase cli, ignore the part with the deployment
* on the site: build -> firestore database -> create db -> start in test mode -> change the location to europe



* start a collection -> collection ID = hotels -> auto-id
* field: id -> value: auto id from above

***Redux***

******



**Adding Redux to the React App (with Typescript):**

yarn add redux react-redux

yarn add @types/react-redux

Tutorial: Add notes with Redux (Notes app) => students add notes to the courses