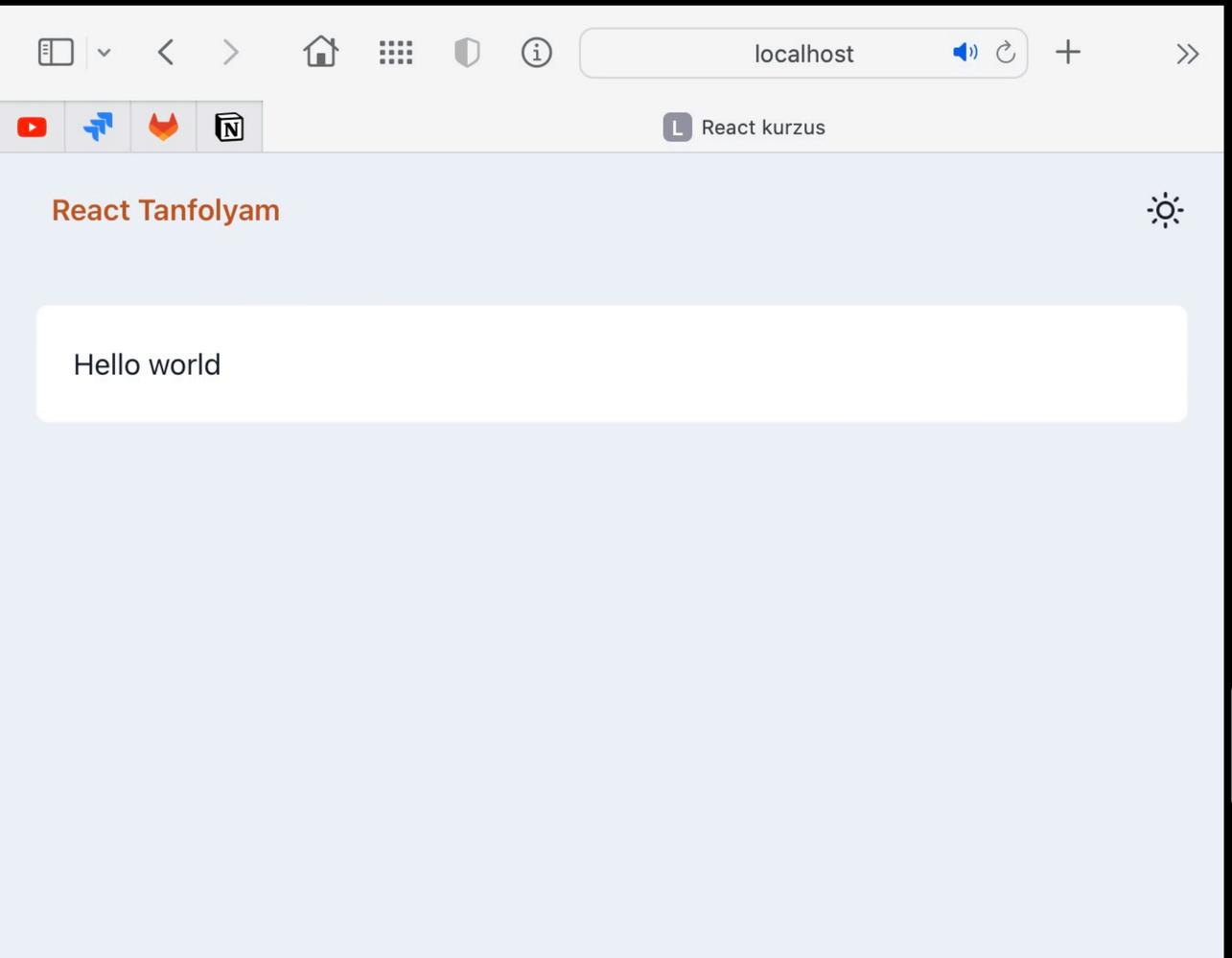


demo master start

jsx.tsx

```tsx
import ReactDOM from 'react-dom';

const element = <h1>Hello world with variable</h1>;
const myButton = (
  <button
    onClick={() => {
      alert('Kattttt');
    }}
  >
    Click
  </button>
);
const color = 'green';

export function renderDemo() {
  const root = document.getElementById( elementId: 'jsx') as HTMLEleme

  // Basic
  ReactDOM.render(<h1>Hello world</h1>, root);
  // Variable
  // ReactDOM.render(element, root)
  // Computed value
  // ReactDOM.render(<h1>Value: {5 + 5}</h1>, root)
  // Composition
  // ReactDOM.render(<h1>{myButton}</h1>, root)
  // Attributes
  // ReactDOM.render(
  //     <h1 style={{ color: color }} className="myHeading">
  //         Colorful wonderful
  //     </h1>,
  //     root
  // )
```

demo > src > examples > jsx > jsx.tsx 1:1 LF UTF-8

localhost

React kurzus

React Tanfolyam

React.FC alapú komponens üdvözöl

demo master start

jsx.tsx FCBased.tsx

```tsx
import {FunctionComponent} from "react";

type MyProps = {
  text: string;
};

export const FCBased: FunctionComponent<MyProps> = ({ text })=> {
  return <h1>{text}</h1>;
}
```

demo > src > examples > components > FCBased.tsx 10:1 LF UTF-8

localhost

React kurzus

React Tanfolyam

Hello

demo master start

jsx.tsx FCBased.tsx FuncBased.tsx

```tsx
export function FuncBased() {
  return <h1>Hello</h1>;
}
```

demo > src > examples > components > FuncBased.tsx 4:1 LF UTF-8

localhost

React kurzus

React Tanfolyam

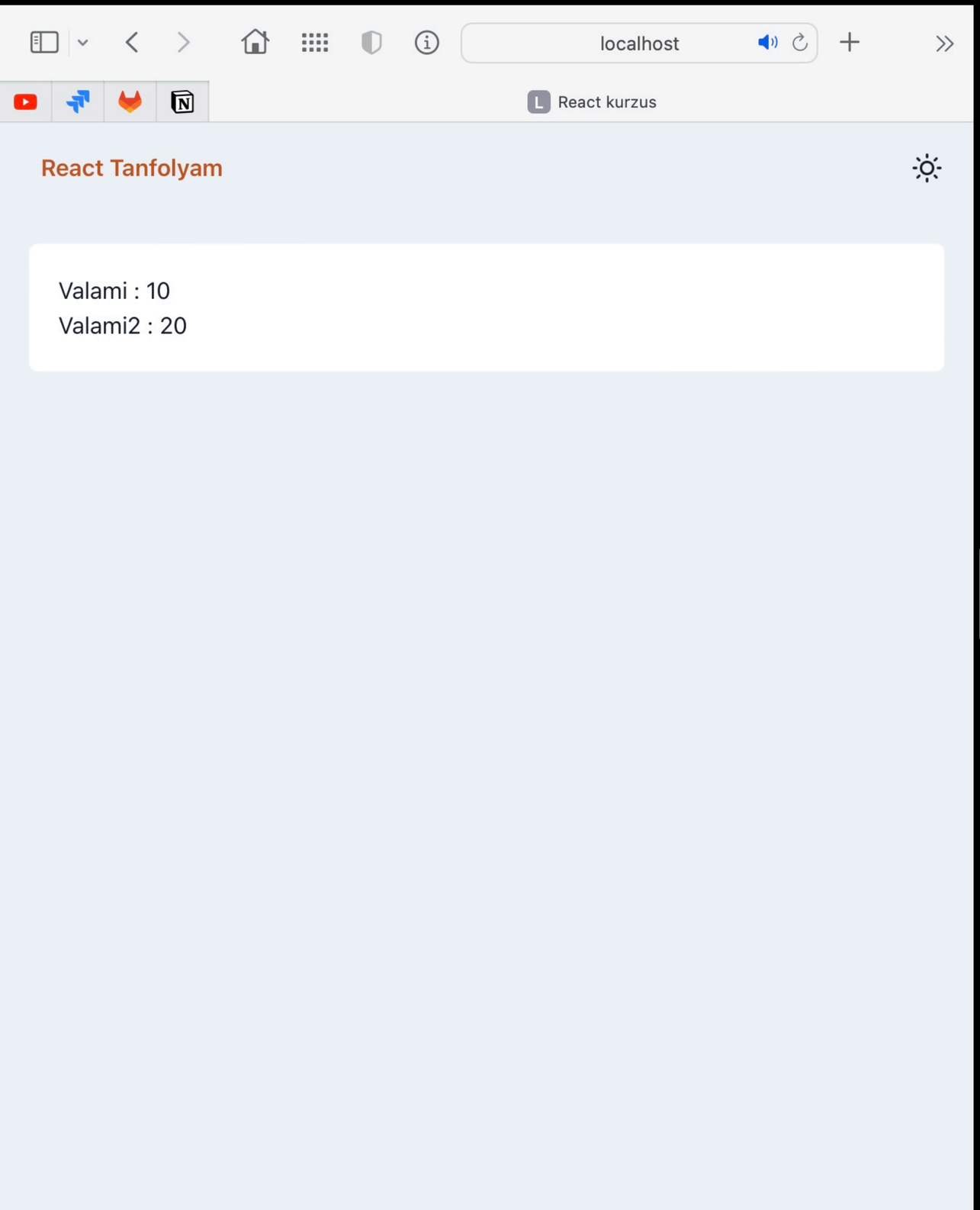
Komponálva vagyokHello

demo master start

Composition.tsx

```tsx
import { Button } from '@chakra-ui/react';
import { PropsWithChildren } from 'react';

export function Composition(props: PropsWithChildren) {
  return <Button colorScheme='orange'>{props.children}</Button>;
}

```

demo > src > examples > components > Composition.tsx 1:1 LF UTF-8



demo master start

PropsWrapper.tsx

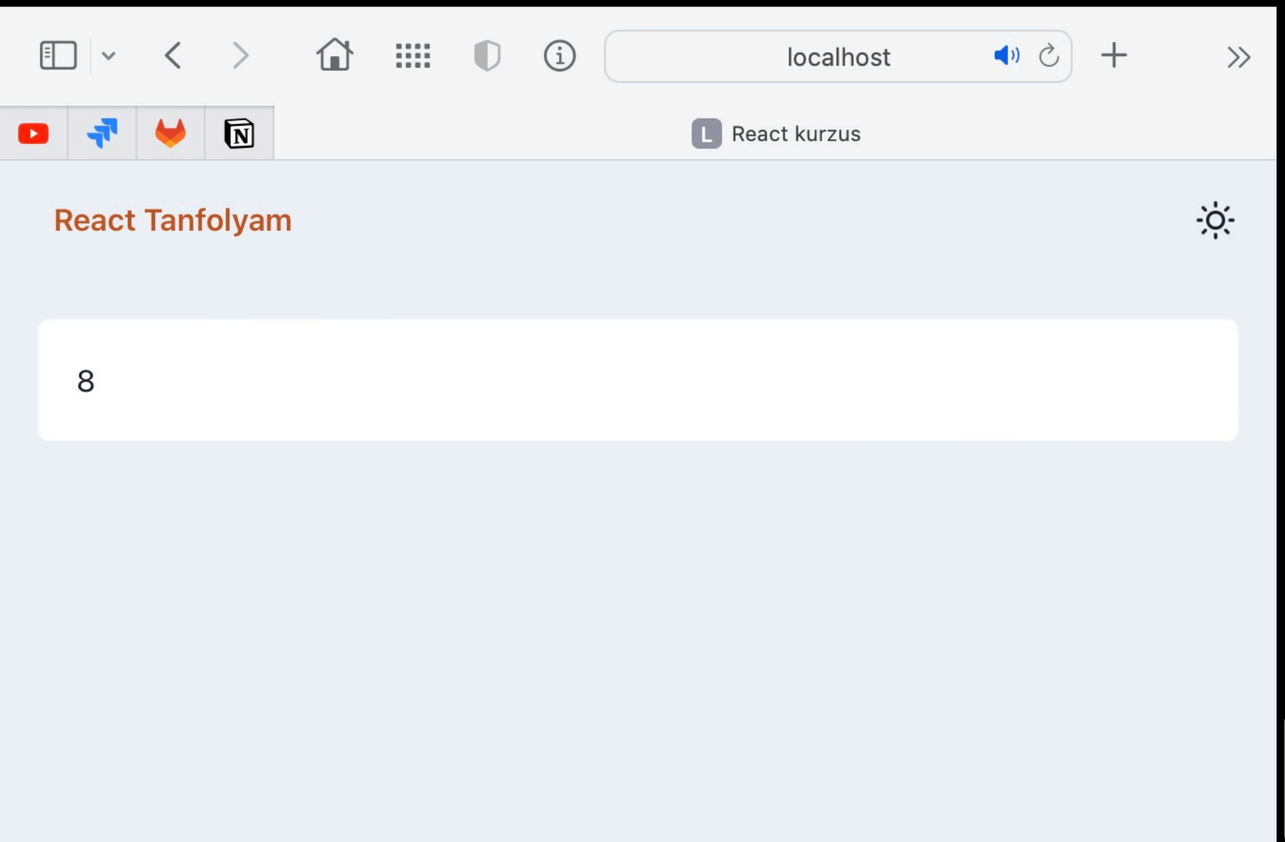
```tsx
import React from 'react';

import { Props } from './Props';

export function PropsWrapper() {
  return (
    <Props
      data={[
        { name: 'Valami', value: 10 },
        { name: 'Valami2', value: 20 },
      ]}
    />
  );
}
```

Props.tsx

```tsx
import React from 'react';

type SomeKindOfObject = {
  name: string;
  value: number | string;
};

type MyProps = {
  data: SomeKindOfObject[];
};

export const Props: React.FC<MyProps> = ({ data }) => {
  return (
    <div>
      {data.map((obj) => (
```

SomeKindOfObject

demo > src > examples > components > Props.tsx 6:3 LF UTF-8

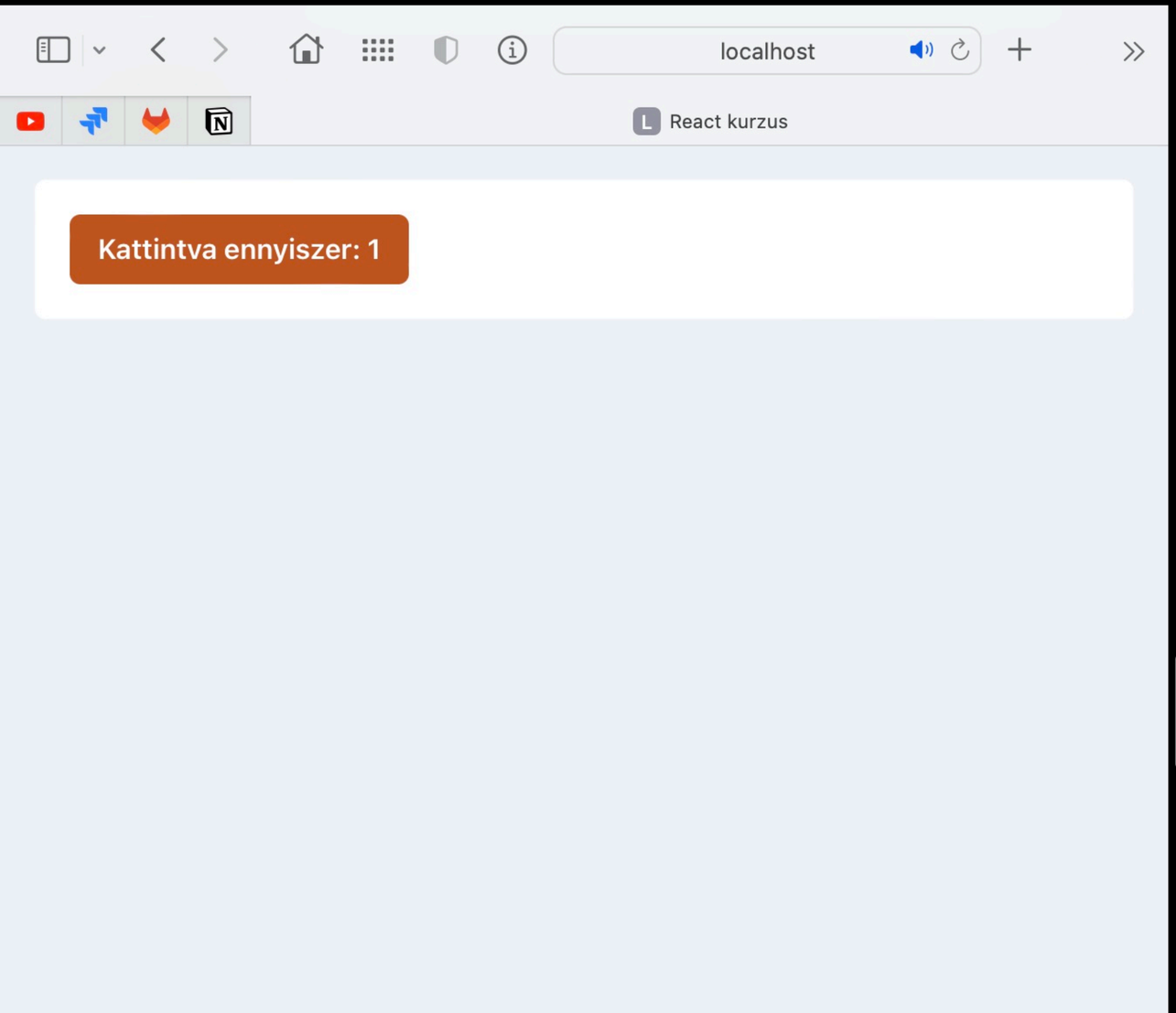


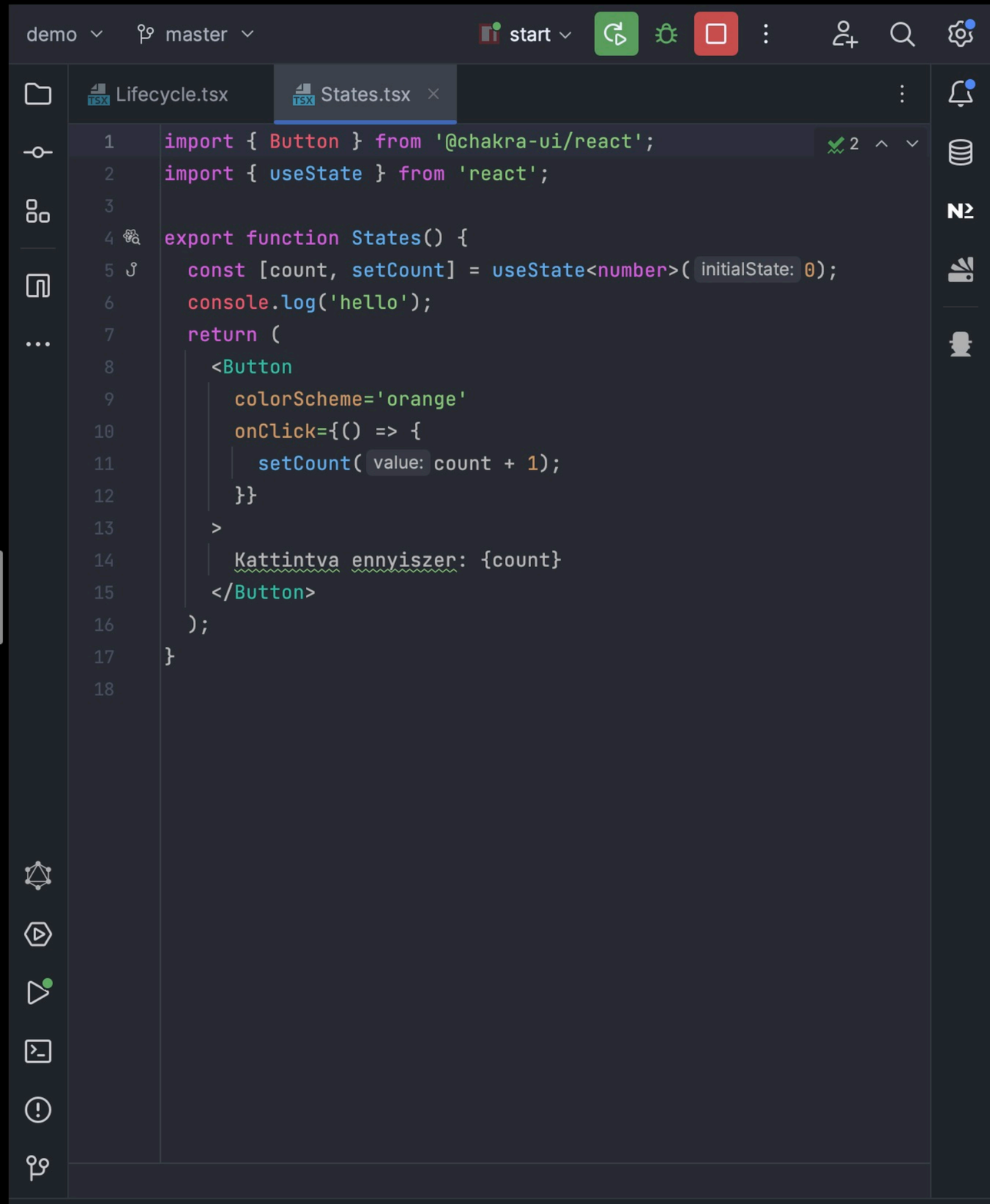
demo master start

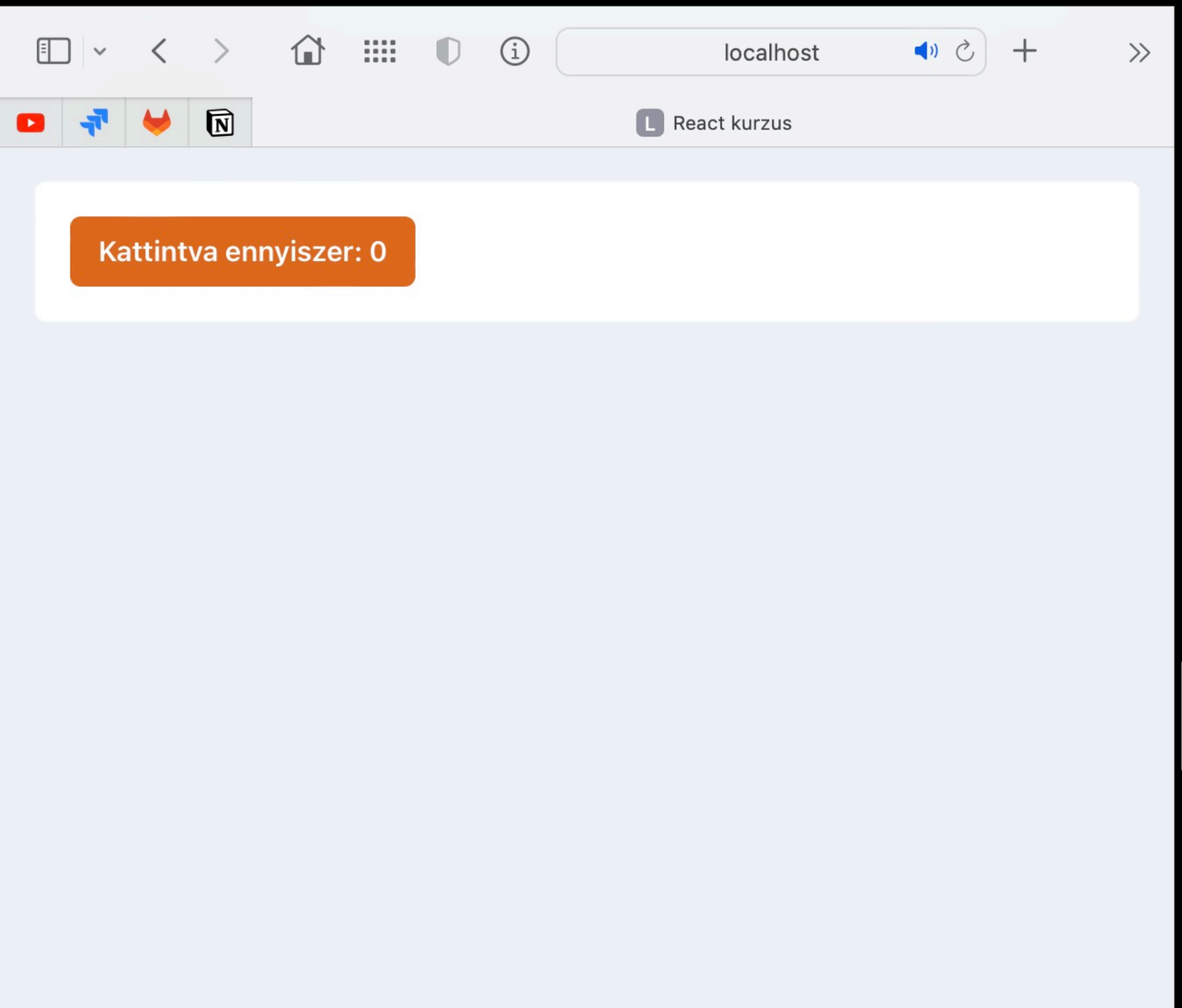
Lifecycle.tsx

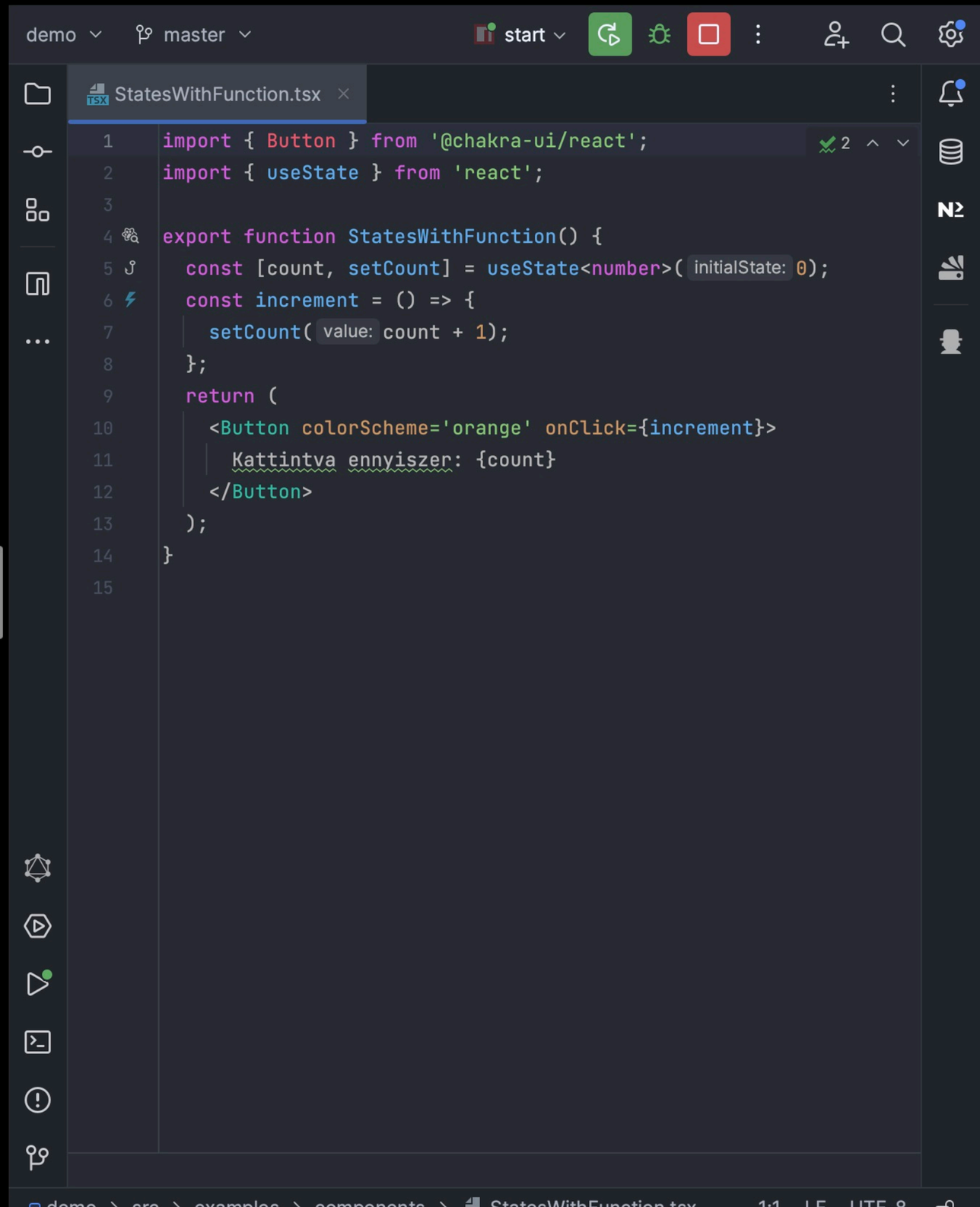
```tsx
import { useEffect, useState } from 'react';

export function Lifecycle() {
  const [count, setCount] = useState( initialState: 0);

  useEffect( effect: () => {
    console.log('MOUNT');

    const id = setInterval( callback: () => {
      console.log('Tick');
      setCount( value: (count) => count + 1);
    }, ms: 1000);

    return () => {
      console.log('UNMOUNT');
      clearInterval(id);
    };
  }, deps: []);

  return <p>{count}</p>;
}

```

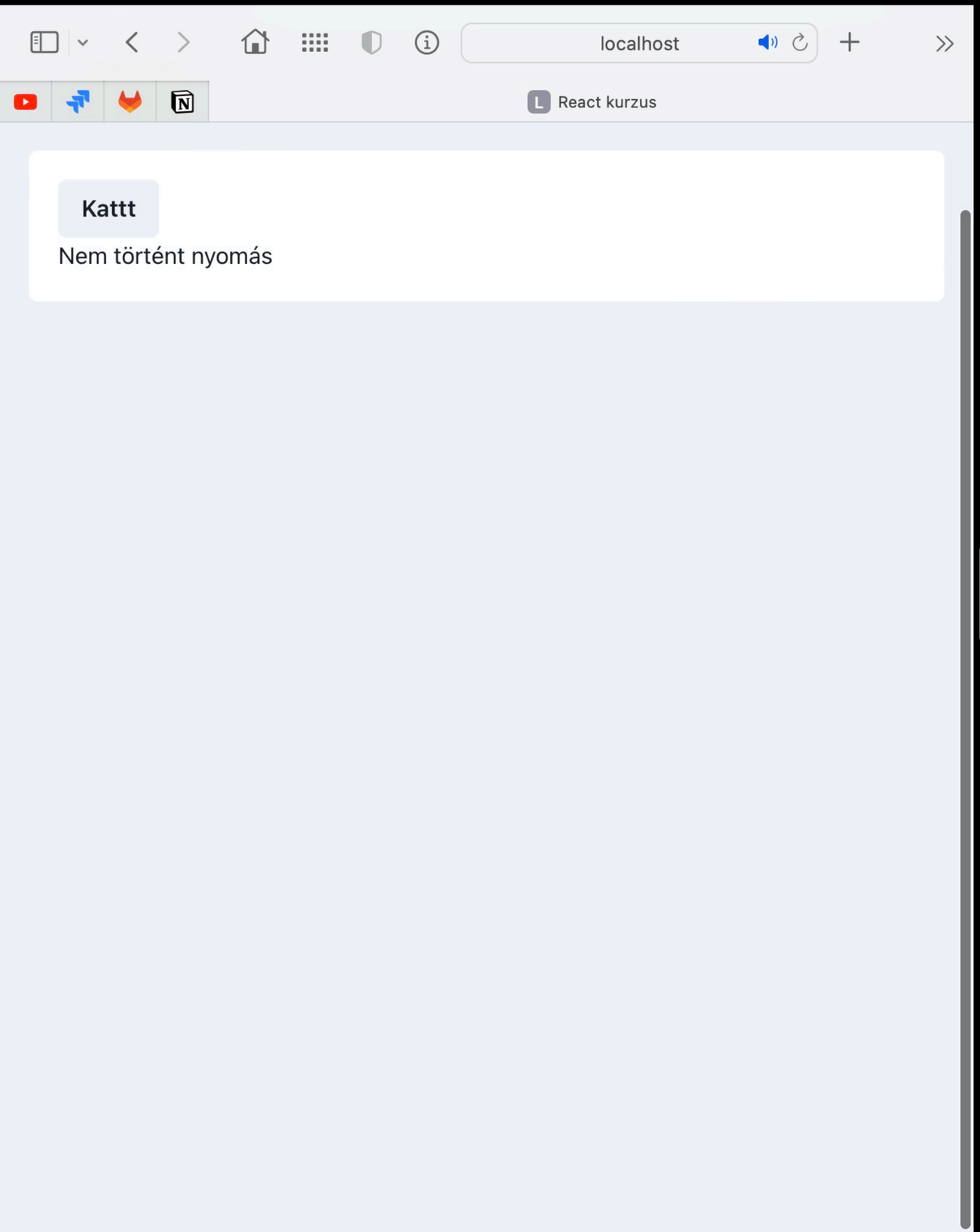
demo > src > examples > components > Lifecycle.tsx 1:1 LF UTF-8


localhost
React kurzus
Kattintva ennyiszer: 1


demo
master
start
Lifecycle.tsx
States.tsx
import { Button } from '@chakra-ui/react';
import { useState } from 'react';
export function States() {
const [count, setCount] = useState<number>(initialState: 0);
console.log('hello');
return (
<Button
colorScheme='orange'
onClick={() => {
setCount(value: count + 1);
}}
>
Kattintva ennyiszer: {count}
</Button>
);
}
demo > src > examples > components > States.tsx
1:1
LF
UTF-8


localhost
React kurzus
Kattintva ennyiszer: 0


demo
master
start
StatesWithFunction.tsx
import { Button } from '@chakra-ui/react';
import { useState } from 'react';
export function StatesWithFunction() {
const [count, setCount] = useState<number>(initialState: 0);
const increment = () => {
setCount(value: count + 1);
};
return (
<Button colorScheme='orange' onClick={increment}>
Kattintva ennyiszer: {count}
</Button>
);
}
demo
src
examples
components
StatesWithFunction.tsx
1:1
LF
UTF-8

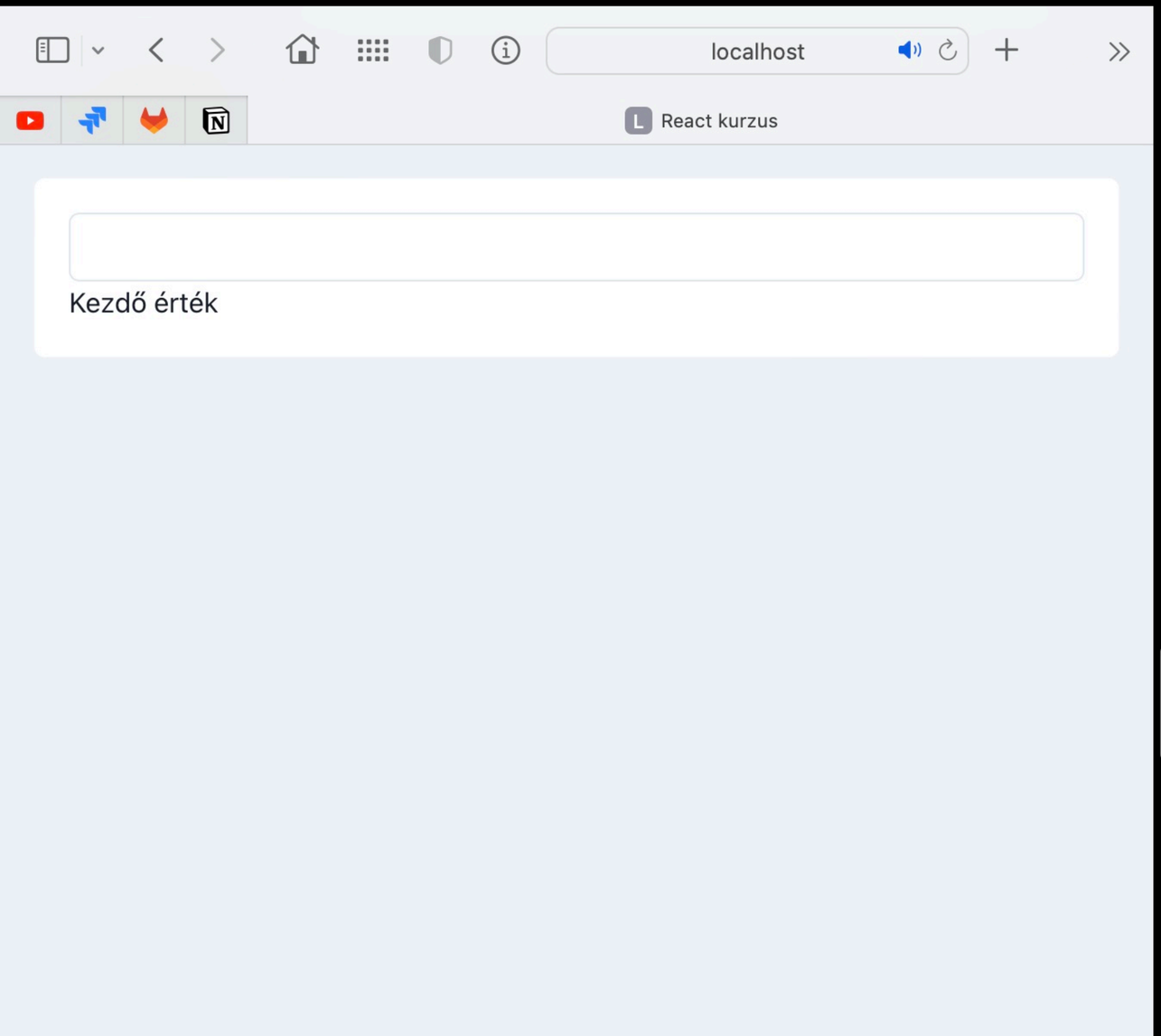


demo master start

StatesWithFunction.tsx FunctionInteraction.tsx

```tsx
import { Button } from '@chakra-ui/react';
import React, { useState } from 'react';

export function FunctionInteraction() {
  const [text, setText] = useState<string>( initialState: 'Nem történt
  const func = (newText: string) => {
    setText(newText);
  };
  return (
    <>
      <InnerComponent func={func} />
      <h1>{text}</h1>
    </>
  );
}

type MyProps = {
  func: (text: string) => void;
};

const InnerComponent: React.FC<MyProps> = ({ func }) => {
  return (
    <Button
      onClick={() => {
        func( text: 'hello');
      }}
    >
      Kattt
    </Button>
  );
};
```

demo > src > examples > interaction > FunctionInteraction.tsx 1:1 LF UTF-8



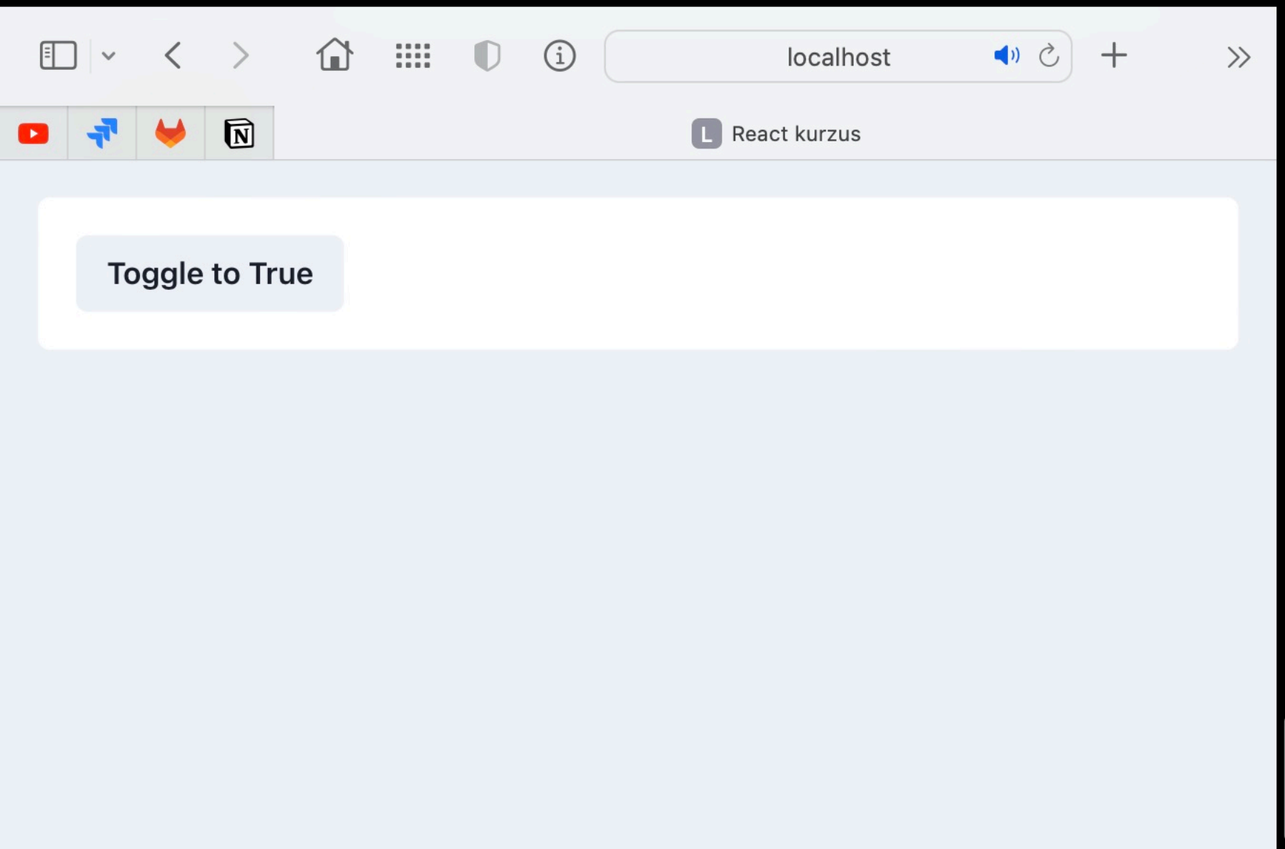
demo master start

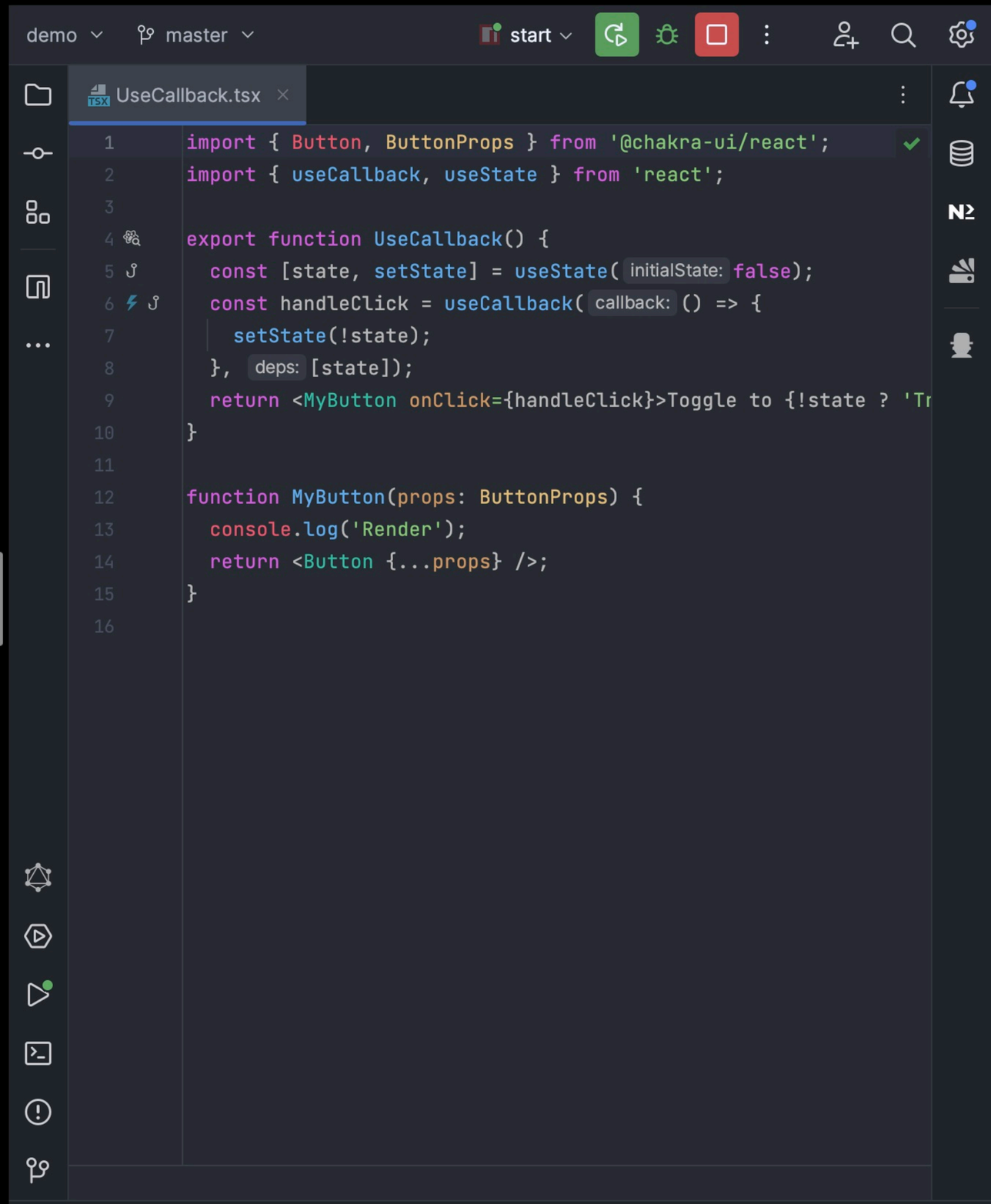
ContextInteraction.tsx

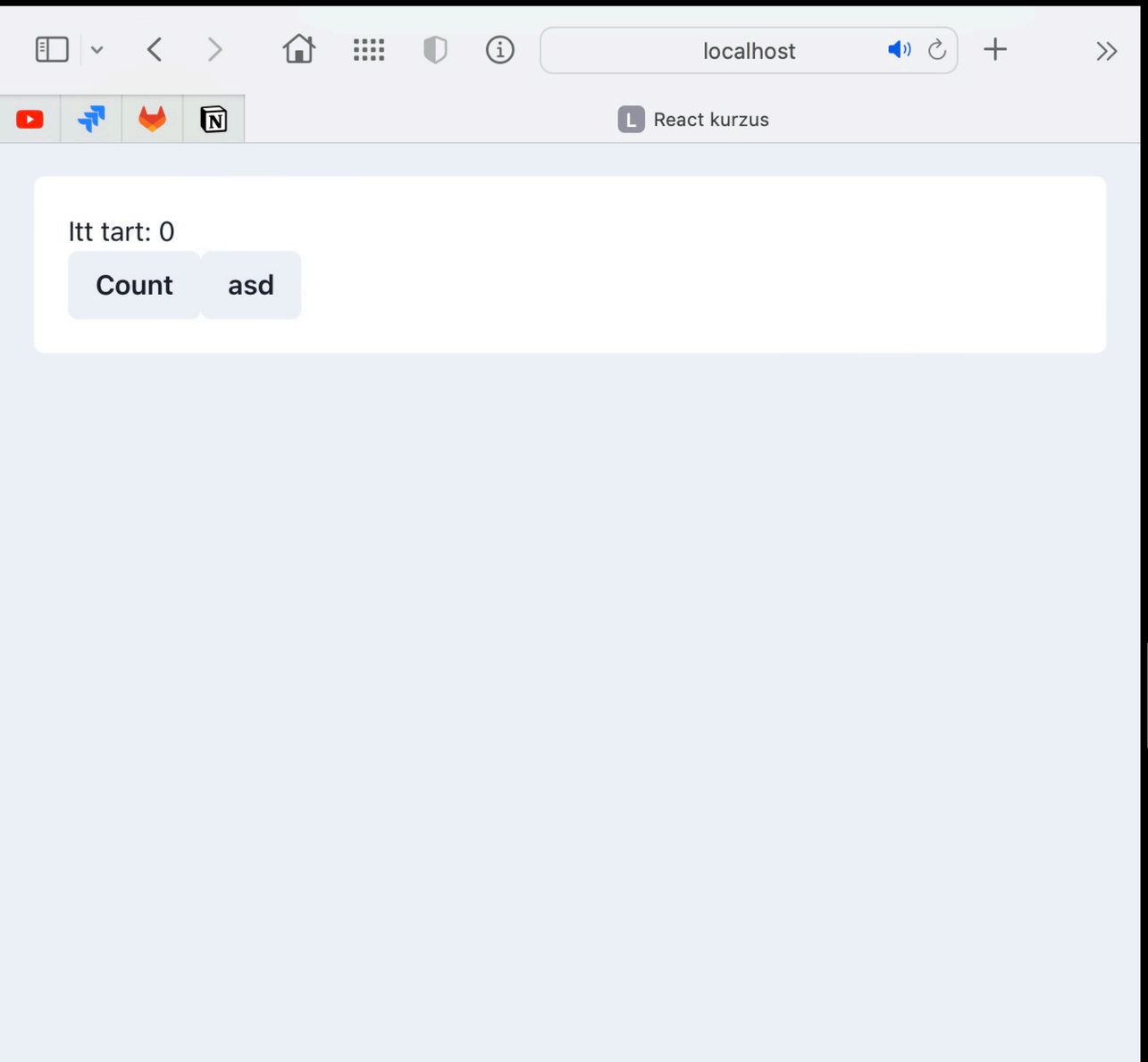
```tsx
export function ContextInteraction() {
  return (
    <MyContextProvider>
      <div>
        <div>
          <InputComponent />
        </div>
      </div>
      <OutputComponent />
    </MyContextProvider>
  );
}

type MyContextType = {
  value: string;
  setValue: (newValue: string) => void;
};

const MyContext = createContext<MyContextType>( defaultValue: {
  value: '',
  // eslint-disable-next-line @typescript-eslint/no-empty-functio
  setValue: () => {},
});

function MyContextProvider({ children }: PropsWithChildren) {
  const [value, setValue] = useState<string>( initialState: 'Kezdő érté

  return <MyContext.Provider value={{ value, setValue }}>{childre
}

function InputComponent() {
  const { setValue } = useContext<MyContextType>(MyContext);
```

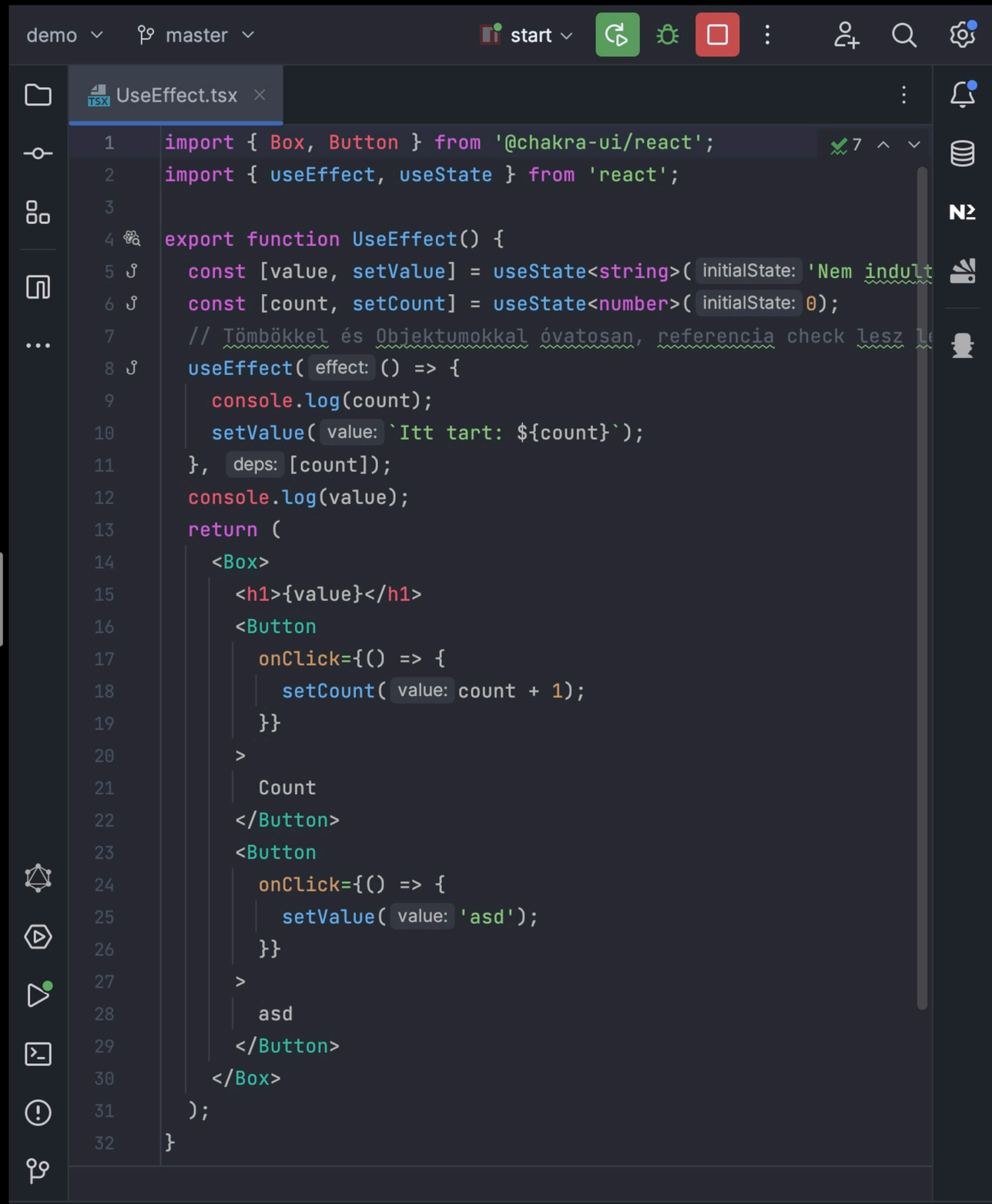
MyContext

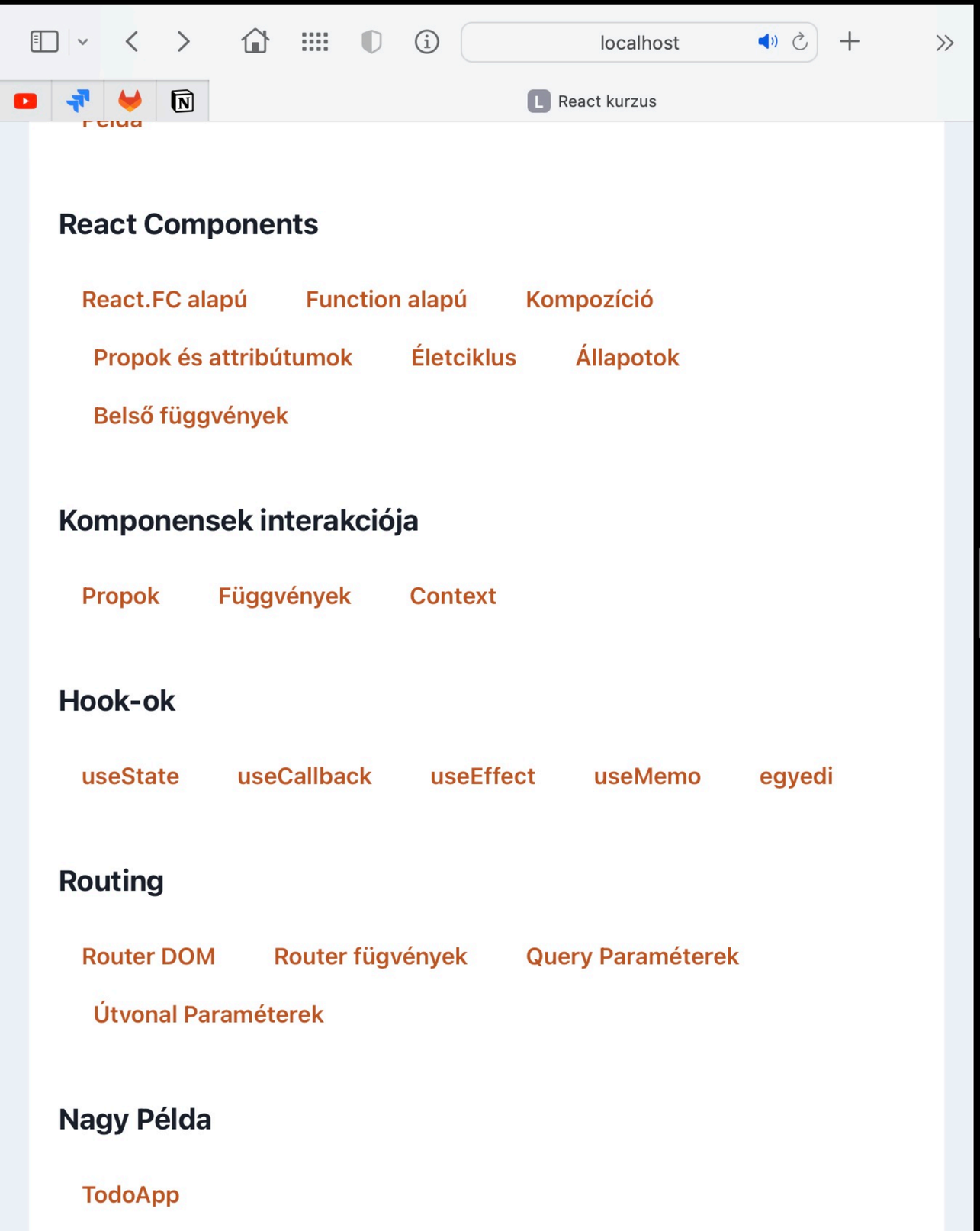
examples > interaction > ContextInteraction.tsx > MyContext 23:13 LF UTF-8

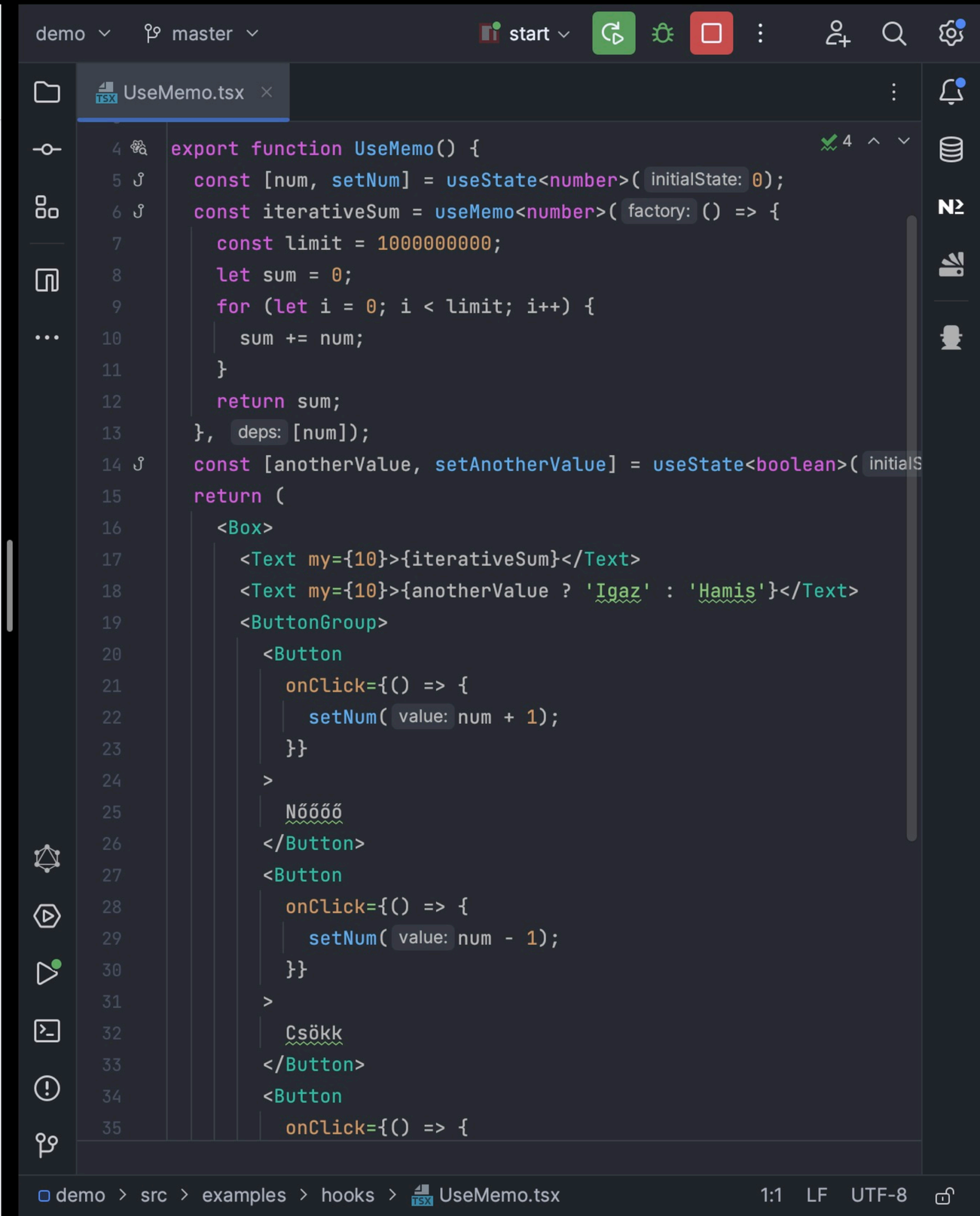

localhost
React kurzus
Toggle to True

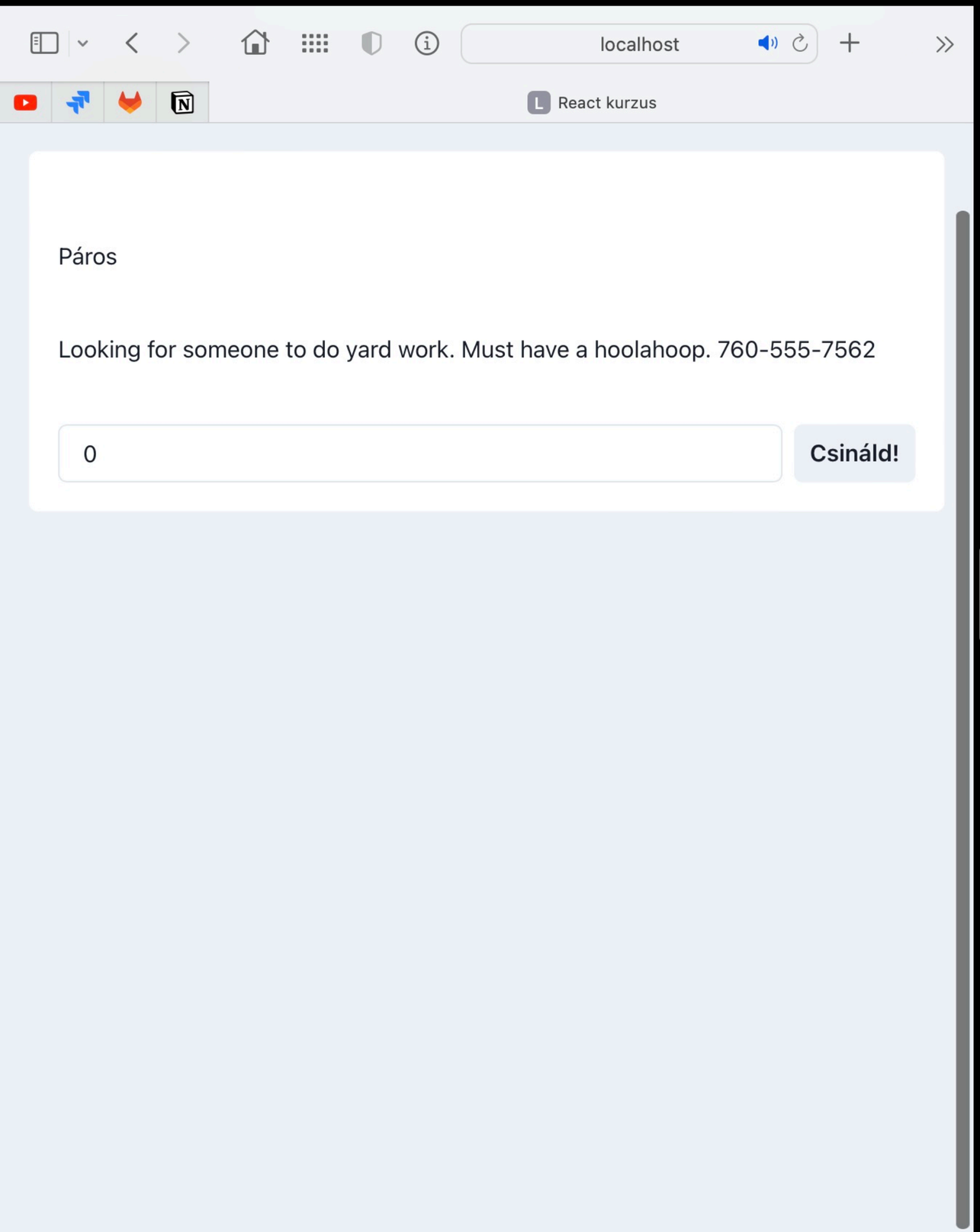

demo
master
start
UseCallback.tsx
import { Button, ButtonProps } from '@chakra-ui/react';
import { useCallback, useState } from 'react';
export function UseCallback() {
const [state, setState] = useState(initialState: false);
const handleClick = useCallback(callback: () => {
setState(!state);
}, deps: [state]);
return <MyButton onClick={handleClick}>Toggle to {!state ? 'Tr
}
function MyButton(props: ButtonProps) {
console.log('Render');
return <Button {...props} />;
}
demo > src > examples > hooks > UseCallback.tsx
1:1
LF
UTF-8

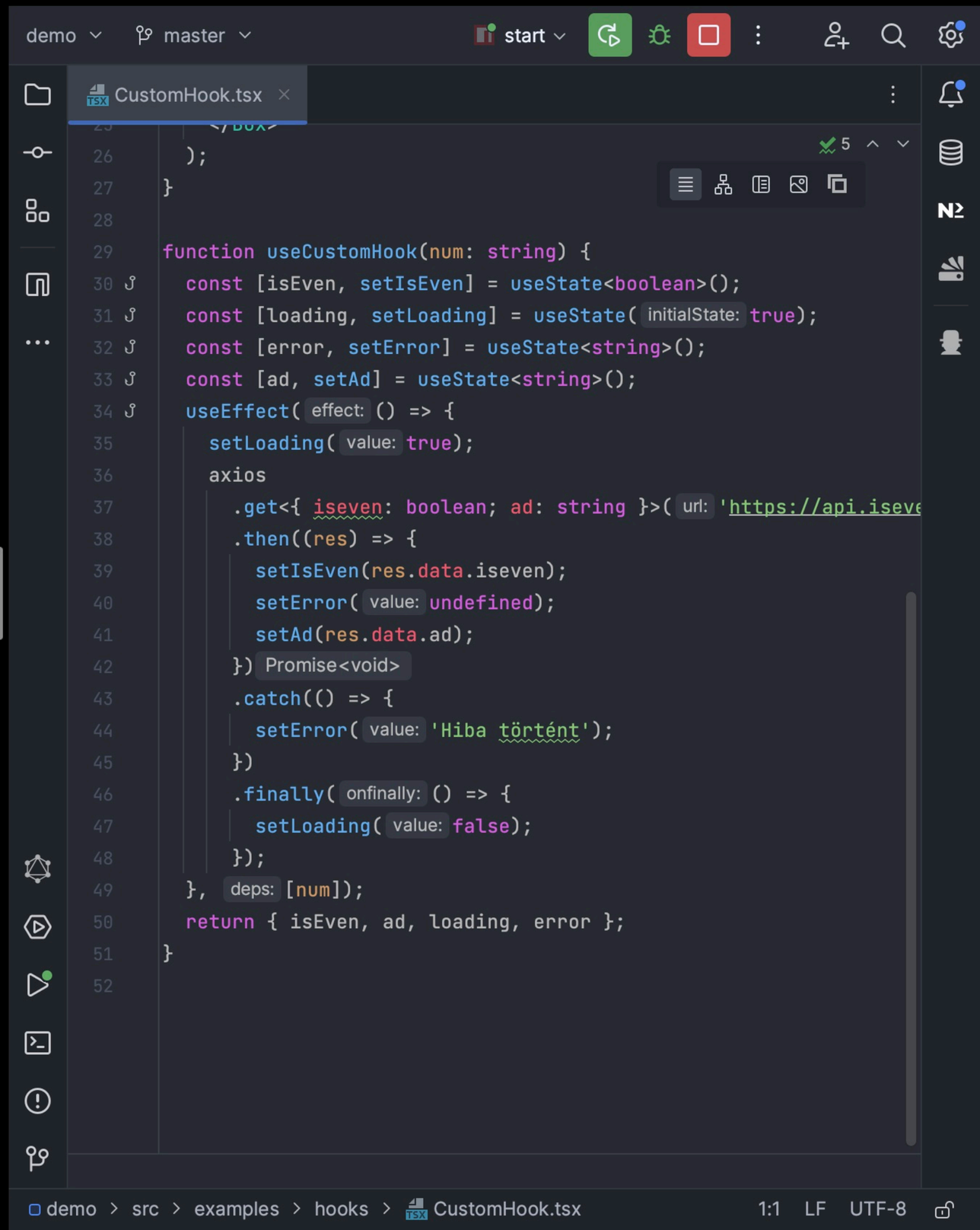

localhost
React kurzus
Itt tart: 0
Count
asd

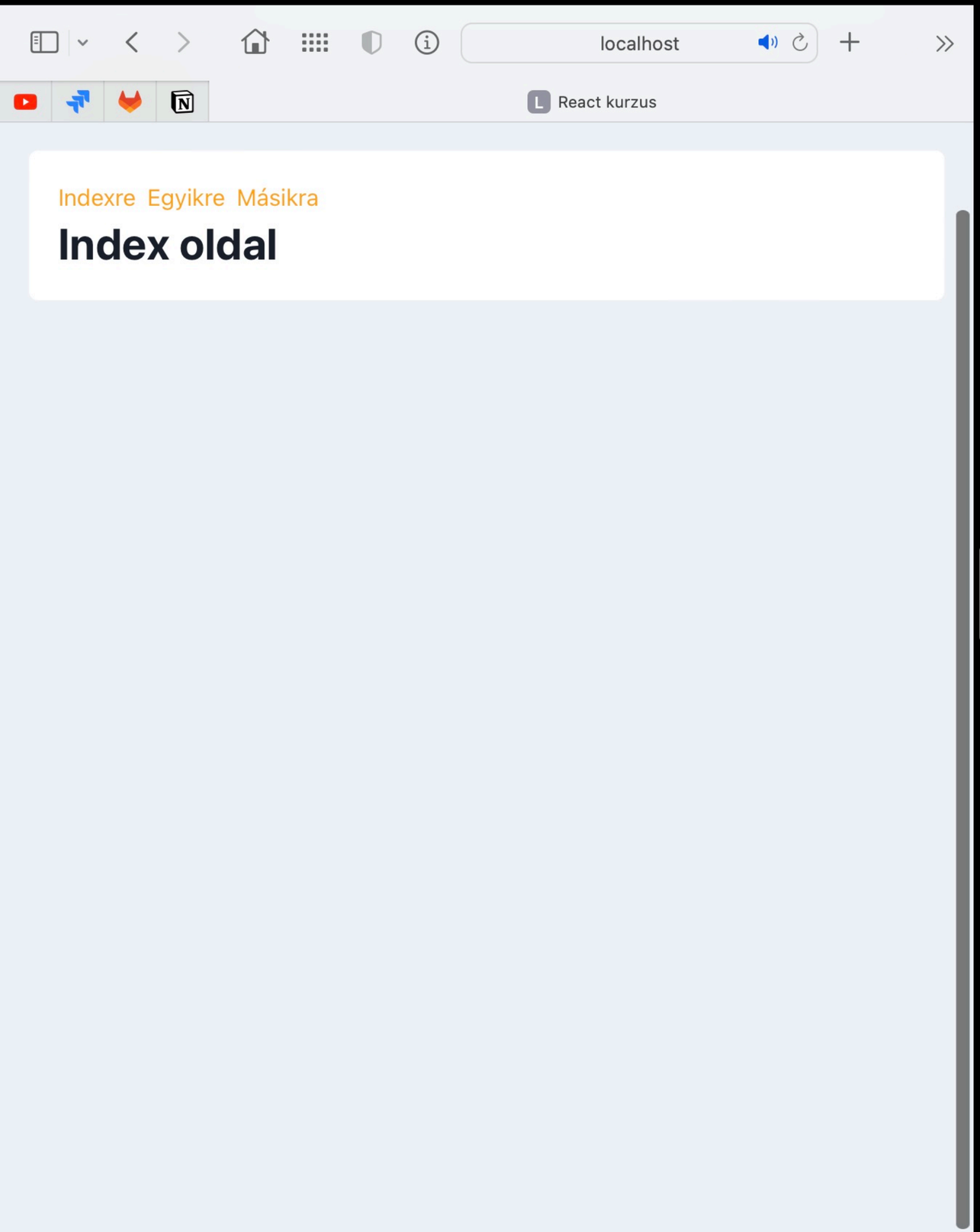

demo
master
start
UseEffect.tsx
import { Box, Button } from '@chakra-ui/react';
import { useEffect, useState } from 'react';
export function UseEffect() {
const [value, setValue] = useState<string>(initialState: 'Nem indult
const [count, setCount] = useState<number>(initialState: 0);
// Tömbökkel és Objektumokkal óvatosan, referencia check lesz
useEffect(effect: () => {
console.log(count);
setValue(value: `Itt tart: ${count}`);
}, deps: [count]);
console.log(value);
return (
<Box>
<h1>{value}</h1>
<Button
onClick={() => {
setCount(value: count + 1);
}}
>
Count
</Button>
<Button
onClick={() => {
setValue(value: 'asd');
}}
>
asd
</Button>
</Box>
);
}
demo > src > examples > hooks > UseEffect.tsx
1:1
LF
UTF-8


localhost
React kurzus
Példa
React Components
React.FC alapú
Function alapú
Kompozíció
Propok és attribútumok
Életciklus
Állapotok
Belső függvények
Komponensek interakciója
Propok
Függvények
Context
Hook-ok
useState
useCallback
useEffect
useMemo
egyedi
Routing
Router DOM
Router fügvények
Query Paraméterek
Útvonal Paraméterek
Nagy Példa
TodoApp


demo
master
start
UseMemo.tsx
export function UseMemo() {
const [num, setNum] = useState<number>(initialState: 0);
const iterativeSum = useMemo<number>(factory: () => {
const limit = 1000000000;
let sum = 0;
for (let i = 0; i < limit; i++) {
sum += num;
}
return sum;
}, deps: [num]);
const [anotherValue, setAnotherValue] = useState<boolean>(initialS
return (
<Box>
<Text my={10}>{iterativeSum}</Text>
<Text my={10}>{anotherValue ? 'Igaz' : 'Hamis'}</Text>
<ButtonGroup>
<Button
onClick={() => {
setNum(value: num + 1);
}}
>
Nőőőő
</Button>
<Button
onClick={() => {
setNum(value: num - 1);
}}
>
Csökk
</Button>
<Button
onClick={() => {
demo > src > examples > hooks > UseMemo.tsx
1:1
LF
UTF-8


localhost
React kurzus
Páros
Looking for someone to do yard work. Must have a hoolahoop. 760-555-7562
0
Csináld!


demo
master
start
CustomHook.tsx
);
}
function useCustomHook(num: string) {
const [isEven, setIsEven] = useState<boolean>();
const [loading, setLoading] = useState(initialState: true);
const [error, setError] = useState<string>();
const [ad, setAd] = useState<string>();
useEffect(effect: () => {
setLoading(value: true);
axios
.get<{ iseven: boolean; ad: string }>(url: 'https://api.iseve
.then((res) => {
setIsEven(res.data.iseven);
setError(value: undefined);
setAd(res.data.ad);
}) Promise<void>
.catch(() => {
setError(value: 'Hiba történt');
})
.finally(onfinally: () => {
setLoading(value: false);
});
}, deps: [num]);
return { isEven, ad, loading, error };
}
demo > src > examples > hooks > CustomHook.tsx
1:1
LF
UTF-8

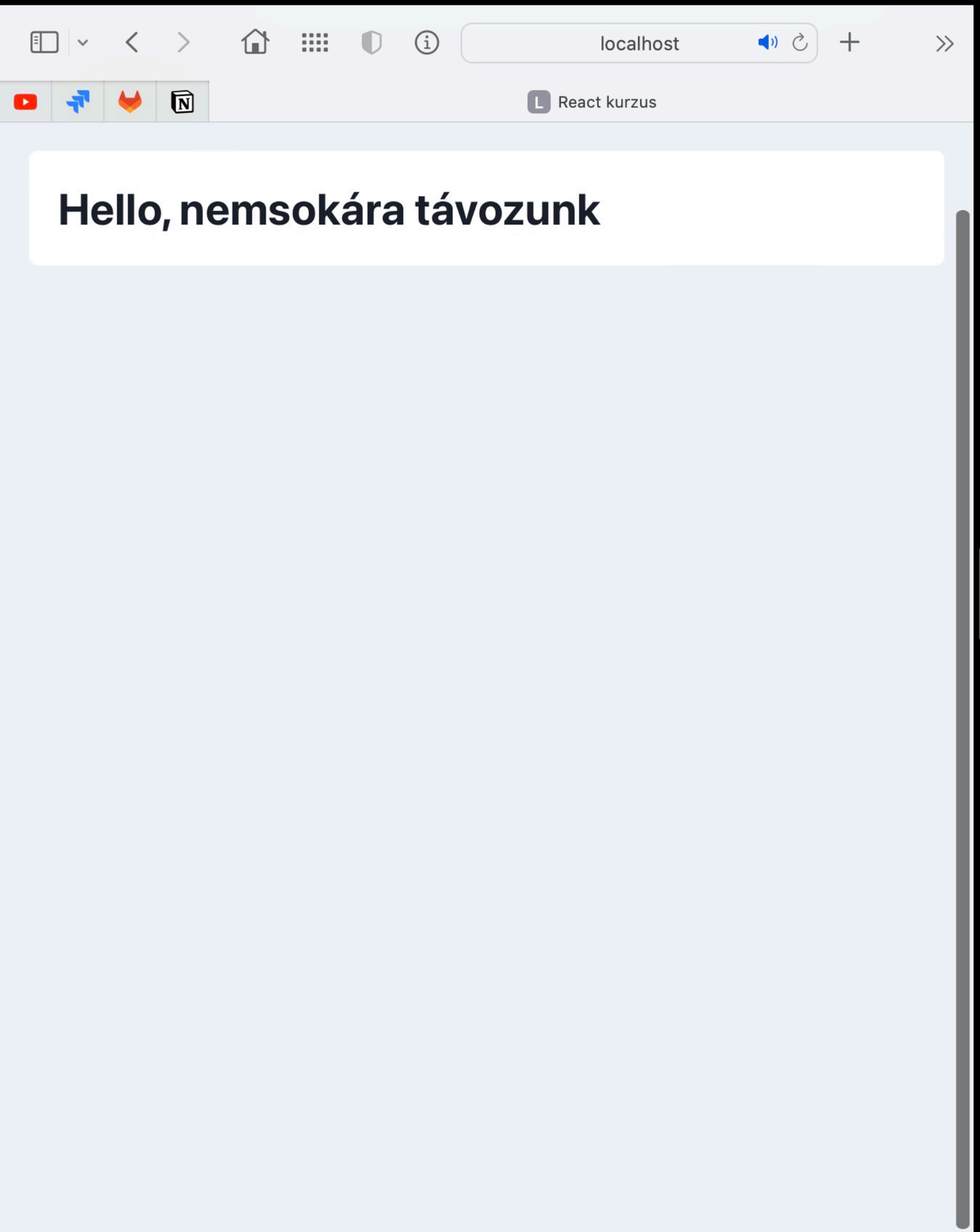


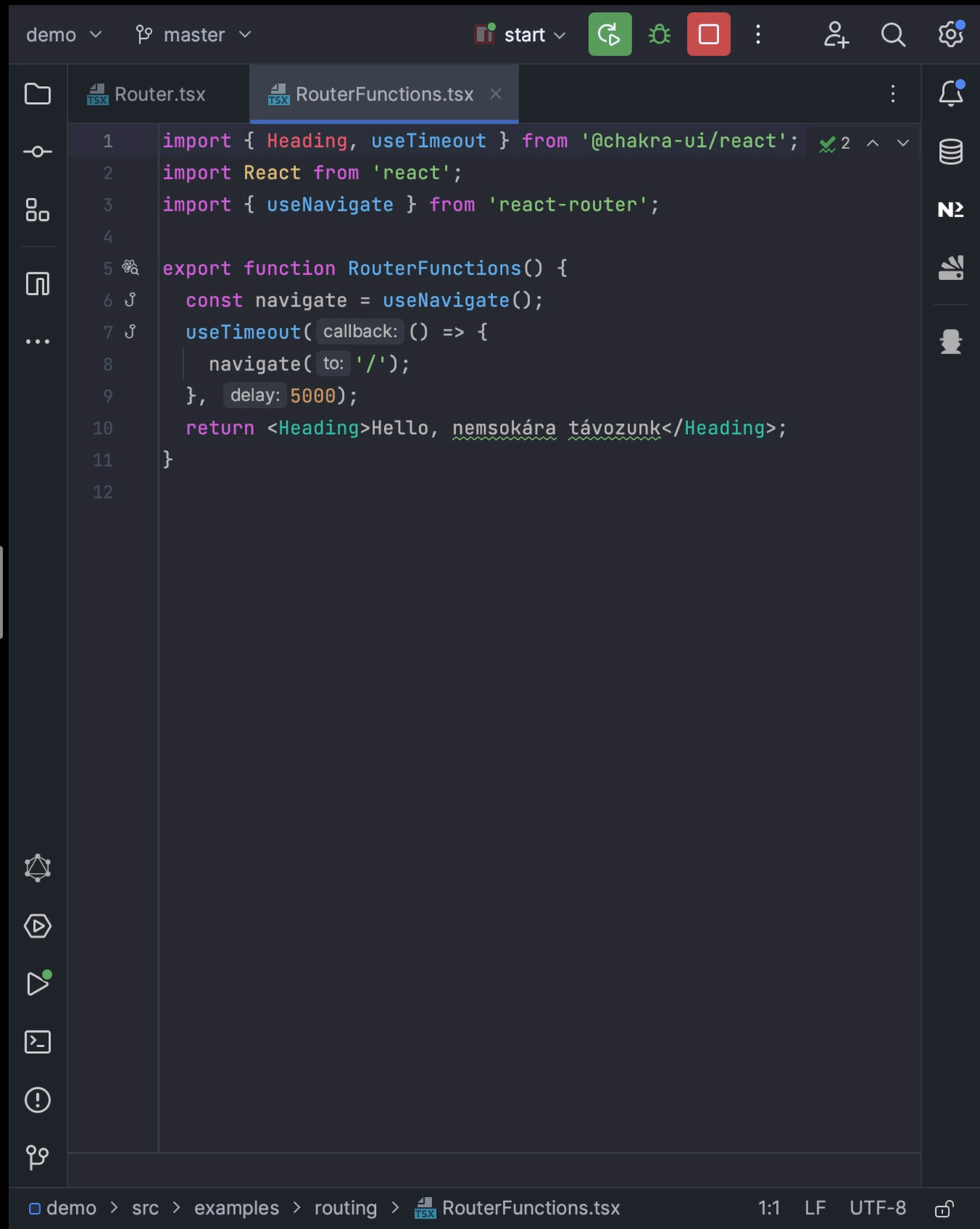
demo master start

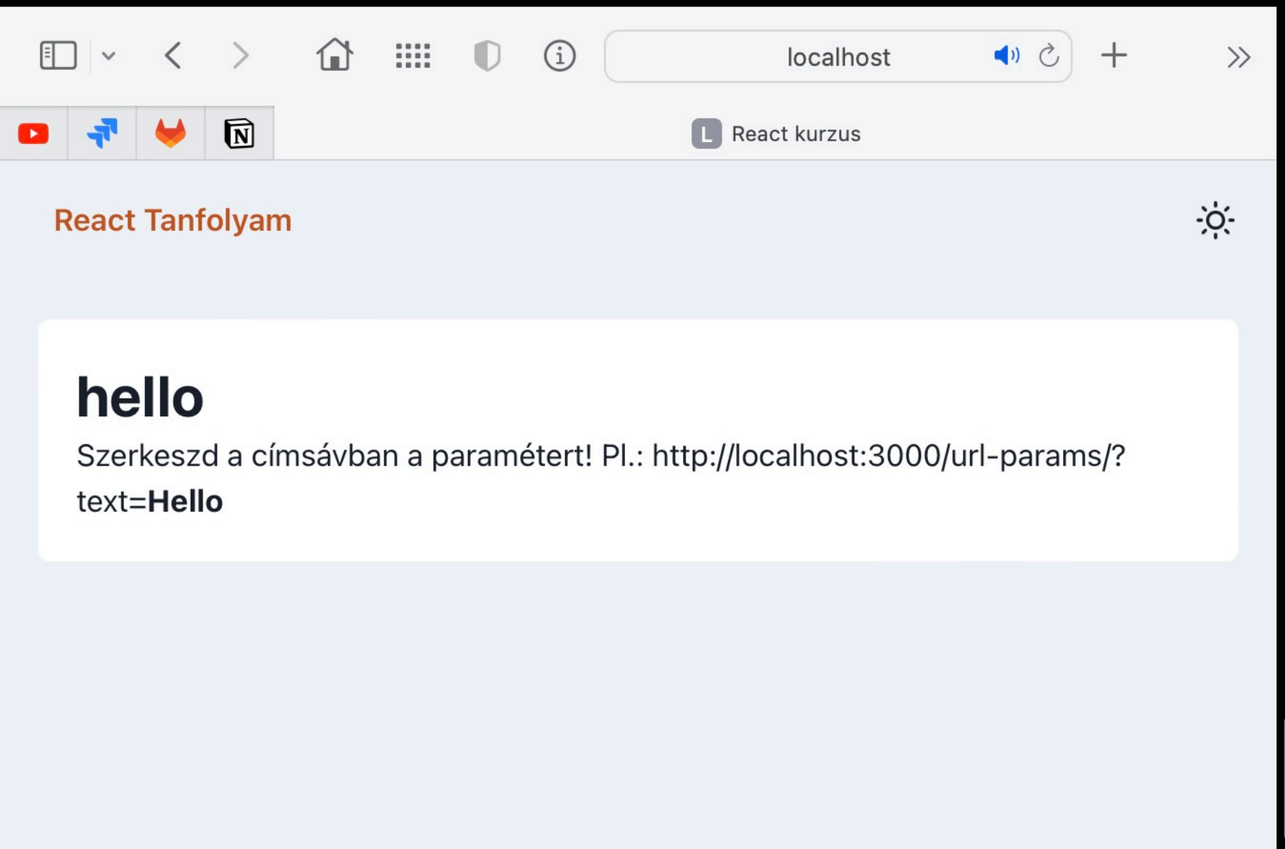
Router.tsx

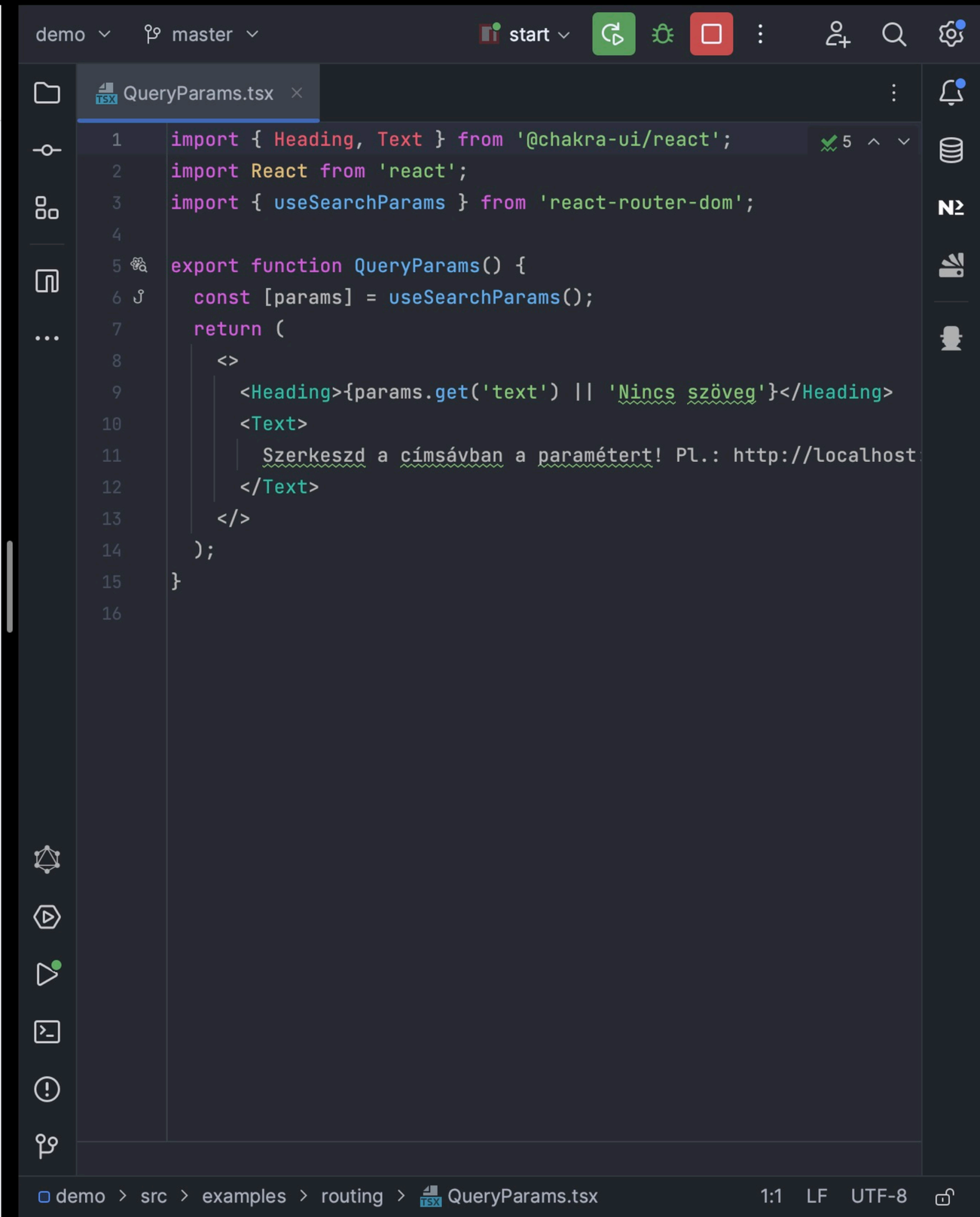
```tsx
import { ButtonGroup, Heading } from '@chakra-ui/react';
import React from 'react';
import { Link, Route, Routes } from 'react-router-dom';

export function Router() {
  return (
    // <BrowserRouter>
    <>
      <ButtonGroup color='orange'>
        <Link to='/router'>Indexre</Link>
        <Link to='/router/egyik'>Egyikre</Link>
        <Link to='/router/masik'>Másikra</Link>
      </ButtonGroup>
      <Routes>
        <Route path='/'>
          <Route path='egyik' element={<Heading>Egyik</Heading>} />
          <Route path='masik' element={<Heading>Másik</Heading>} />
          <Route index element={<Heading>Index oldal</Heading>} />
        </Route>
      </Routes>
    </>
    // </BrowserRouter>
  );
}
```

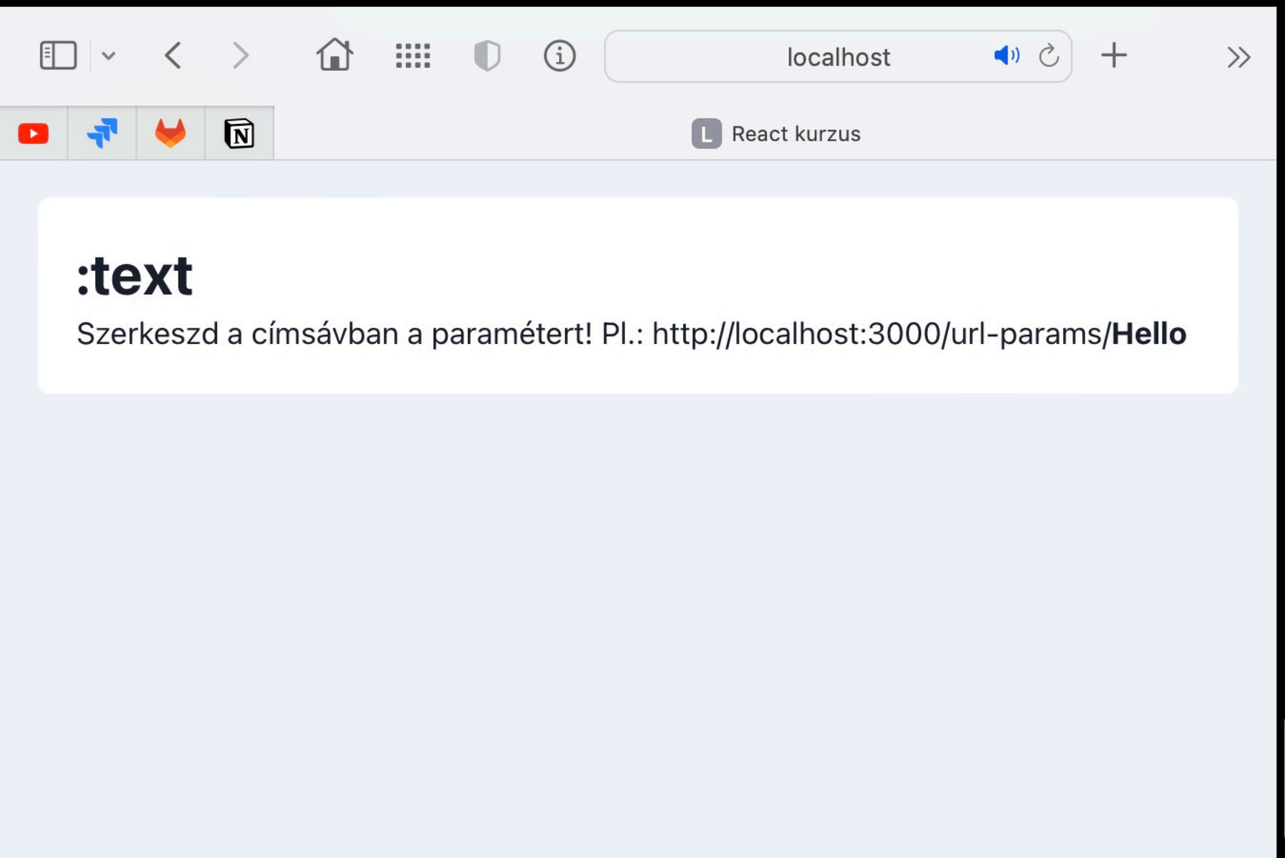
demo > src > examples > routing > Router.tsx 1:1 LF UTF-8


localhost
React kurzus
Hello, nemsokára távozunk


demo
master
start
Router.tsx
RouterFunctions.tsx
import { Heading, useTimeout } from '@chakra-ui/react';
import React from 'react';
import { useNavigate } from 'react-router';
export function RouterFunctions() {
const navigate = useNavigate();
useTimeout(callback: () => {
navigate(to: '/');
}, delay: 5000);
return <Heading>Hello, nemsokára távozunk</Heading>;
}
demo > src > examples > routing > RouterFunctions.tsx
1:1
LF
UTF-8

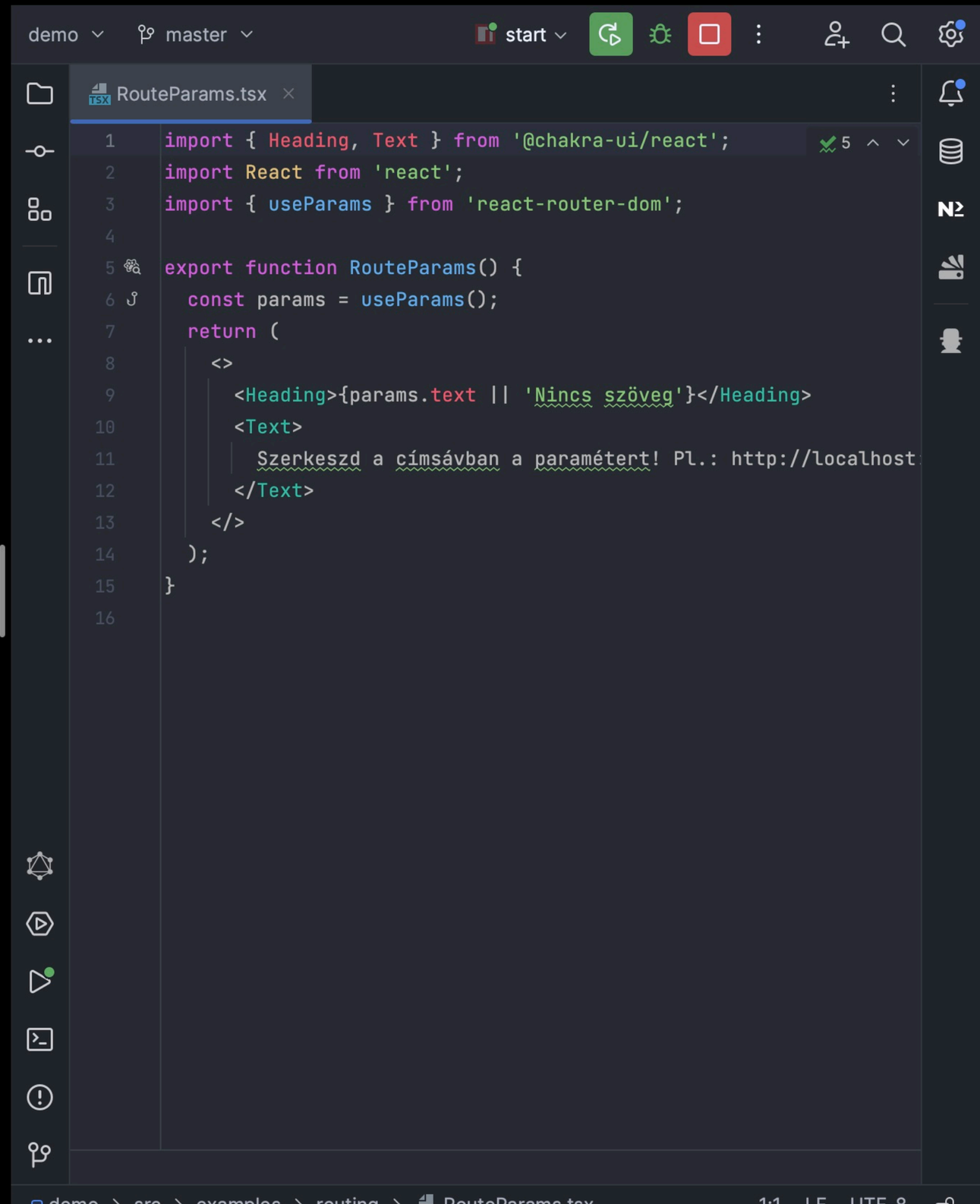

localhost
React kurzus
React Tanfolyam
hello
Szerkeszd a címsávban a paramétert! Pl.: http://localhost:3000/url-params/?text=Hello

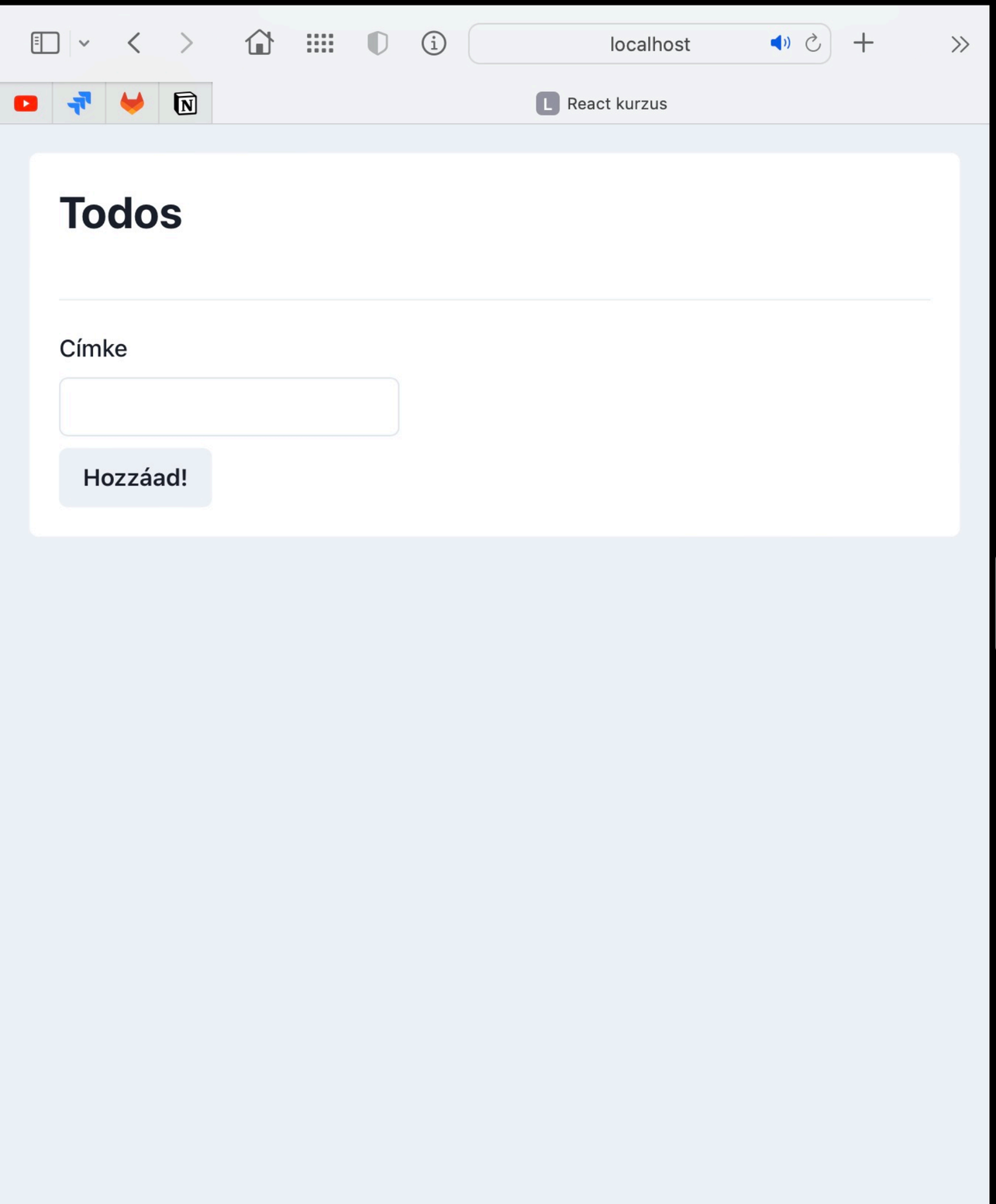

demo
master
start
QueryParams.tsx
import { Heading, Text } from '@chakra-ui/react';
import React from 'react';
import { useSearchParams } from 'react-router-dom';
export function QueryParams() {
const [params] = useSearchParams();
return (
<>
<Heading>{params.get('text') || 'Nincs szöveg'}</Heading>
<Text>
Szerkeszd a címsávban a paramétert! Pl.: http://localhost
</Text>
</>
);
}
demo > src > examples > routing > QueryParams.tsx
1:1
LF
UTF-8

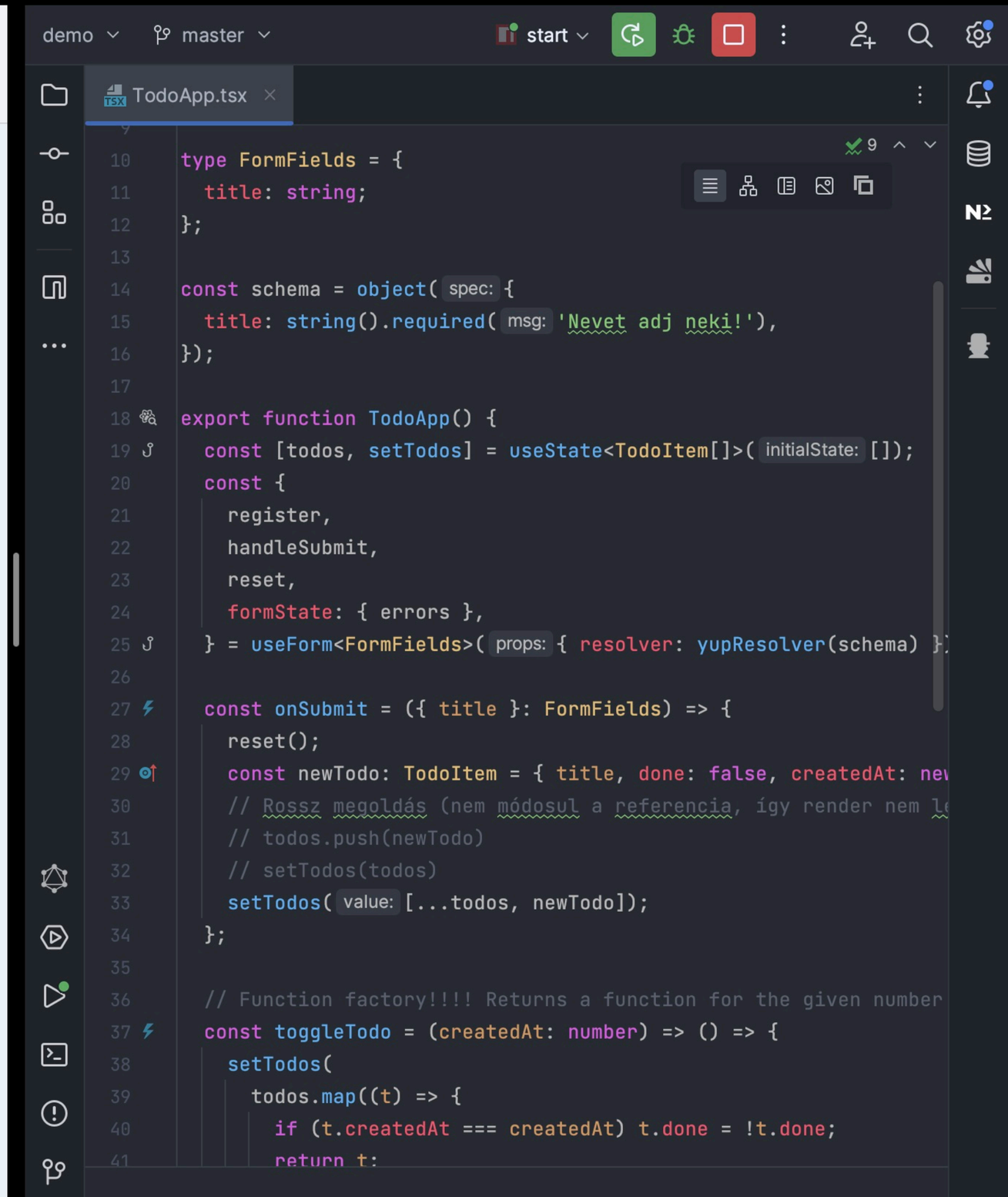


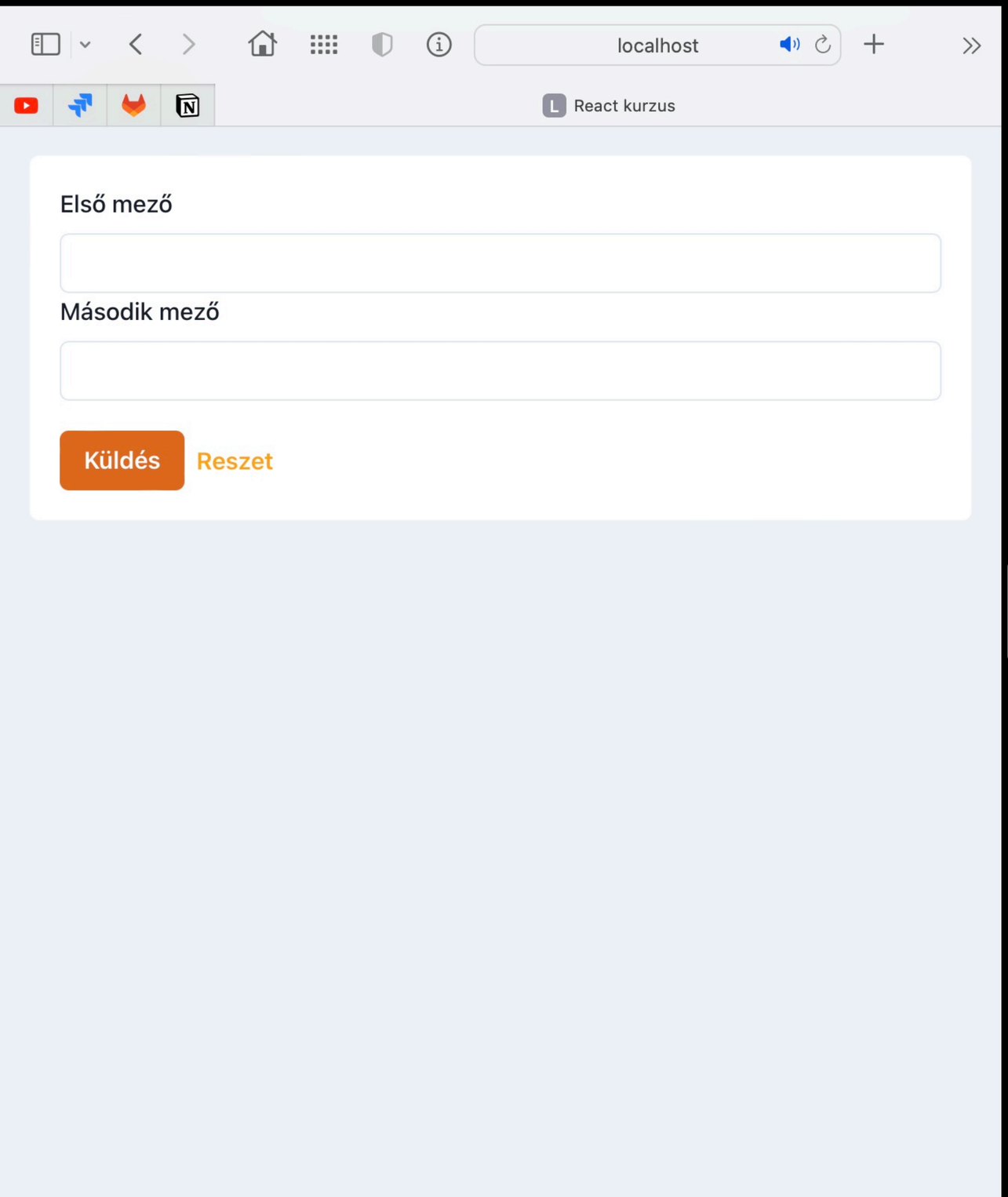
:text

Szerkeszd a címsávban a paramétert! Pl.: http://localhost:3000/url-params/**Hello**




localhost
React kurzus
Todos
Címke
Hozzáad!


demo
master
start
TodoApp.tsx
type FormFields = {
title: string;
};
const schema = object(spec: {
title: string().required(msg: 'Nevet adj neki!'),
});
export function TodoApp() {
const [todos, setTodos] = useState<TodoItem[]>(initialState: []);
const {
register,
handleSubmit,
reset,
formState: { errors },
} = useForm<FormFields>(props: { resolver: yupResolver(schema) }
const onSubmit = ({ title }: FormFields) => {
reset();
const newTodo: TodoItem = { title, done: false, createdAt: ne
// Rossz megoldás (nem módosul a referencia, így render nem l
// todos.push(newTodo)
// setTodos(todos)
setTodos(value: [...todos, newTodo]);
};
// Function factory!!!! Returns a function for the given number
const toggleTodo = (createdAt: number) => () => {
setTodos(
todos.map((t) => {
if (t.createdAt === createdAt) t.done = !t.done;
demo > src > examples > todoapp > TodoApp.tsx
1:1 LF UTF-8

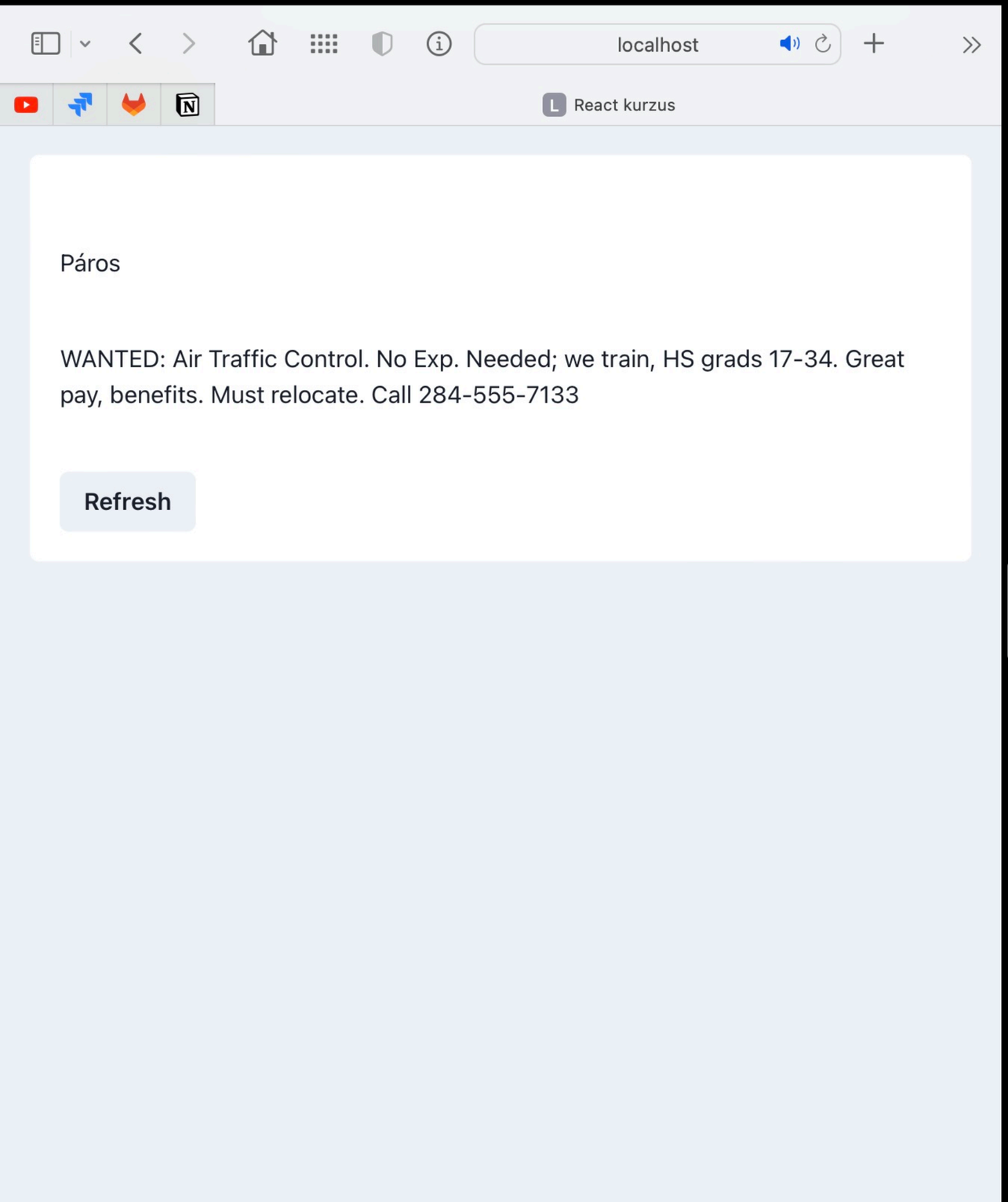


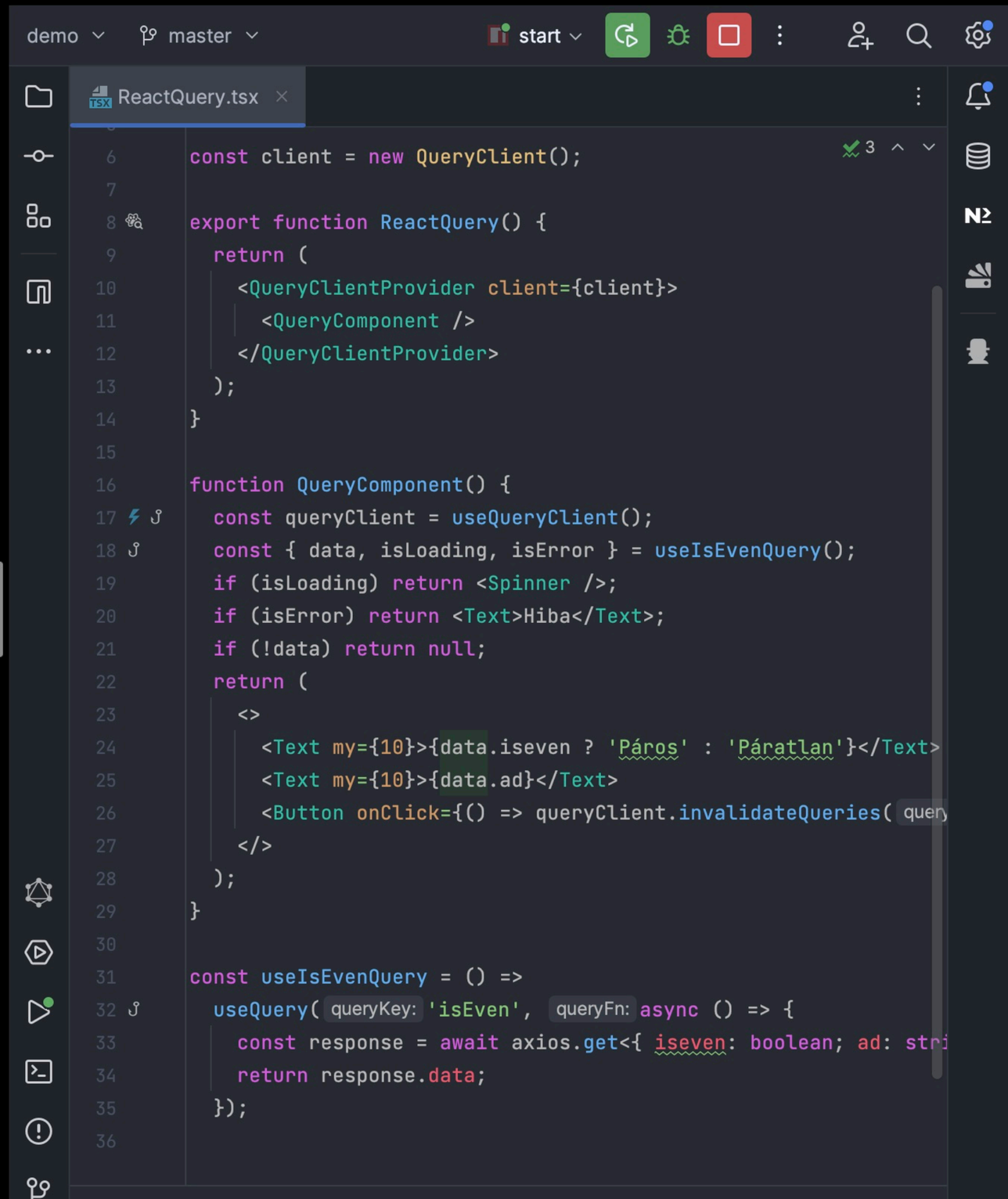
demo master start

TodoApp.tsx Form.tsx

```tsx
const schema = object( spec: {
  first: string().required( msg: 'Kötelezőőő
  second: number().required( msg: 'Kötelezőőőő').positive( msg: 'Mi
});

type FormFields = {
  first: string;
  second: number;
};

export function Form() {
  const {
    register,
    handleSubmit,
    reset,
    formState: { errors },
  } = useForm<FormFields>( props: { resolver: yupResolver(schema) }
  const onSubmit = (values: FormFields) => {
    alert(`${values.first} ${values.second}`);
  };

  return (
    <form onSubmit={handleSubmit(onSubmit)}>
      <FormControl isInvalid={!!errors.first}>
        <FormLabel>Első mező</FormLabel>
        <Input {...register( name: 'first')} />
        <FormErrorMessage>{errors.first?.message}</FormErrorMessa
      </FormControl>
      <FormControl isInvalid={!!errors.second}>
        <FormLabel>Második mező</FormLabel>
        <Input {...register( name: 'second')} />
        <FormErrorMessage>{errors.second?.message}</FormErrorMess
```

demo > src > examples > other > Form.tsx 1:1 LF UTF-8


localhost
React kurzus
Páros
WANTED: Air Traffic Control. No Exp. Needed; we train, HS grads 17-34. Great pay, benefits. Must relocate. Call 284-555-7133
Refresh


demo
master
start
ReactQuery.tsx
const client = new QueryClient();
export function ReactQuery() {
return (
<QueryClientProvider client={client}>
<QueryComponent />
</QueryClientProvider>
);
}
function QueryComponent() {
const queryClient = useQueryClient();
const { data, isLoading, isError } = useIsEvenQuery();
if (isLoading) return <Spinner />;
if (isError) return <Text>Hiba</Text>;
if (!data) return null;
return (
<>
<Text my={10}>{data.iseven ? 'Páros' : 'Páratlan'}</Text>
<Text my={10}>{data.ad}</Text>
<Button onClick={() => queryClient.invalidateQueries(
</>
);
}
const useIsEvenQuery = () =>
useQuery(queryKey: 'isEven', queryFn: async () => {
const response = await axios.get<{ iseven: boolean; ad: stri
return response.data;
});
demo > src > examples > other > ReactQuery.tsx
1:1 LF UTF-8


```tsx
import styled from 'styled-components';

export function StyledComponents() {
  return (
    <StyledButton>
      <p>Pöttyök</p>
    </StyledButton>
  );
}

const StyledButton = styled.div`
  border-radius: 600px;
  height: 200px;
  width: 200px;
  background-color: red;
  display: flex;
  align-items: center;
  justify-content: center;
  color: white;
  font-style: italic;
  font-weight: bold;
  font-size: 3rem;
  transition: transform 0.5s ease;
  :hover {
    transform: rotate(720deg) scale(2);
  }
`;
```

Egyéb

React Hook Forms
React Query
Styled Components
React Docs
create-react-app
Vite
Redux
ChakraUI
React Icons
NextJS