

CREAR UNA VENTANA DE LOGIN DENTRO DE UNA APLICACIÓN CON HOOKS

PASO 0 Crear la aplicación login	1
PASO 1. Instalar reactstrap	3
PASO 2. Crear el componente login	4
PASO 3. Crear el componente Menu	6
PASO 4. Cambiando el componente principal App	7
RESULTADO FINAL DEL COMPONENTE APP	9
PASO 5. MEJORANDO EL COMPONENTE LOGIN	11
ASPECTO DEL COMPONENTE HASTA AHORA	13
PASO 6 COMPLETANDO LA APLICACIÓN	15
Cambios en Menu.js	15
ASPECTO DE App.js HASTA AHORA	16

PASO 0 Crear la aplicación login

En este primer paso lo que vamos a hacer es crear una nueva aplicación que la vamos a llamar login.

Para ello utilizaremos la aplicación create-react-app para crear una estructura vacía

```
:~/react$ create-react-app login
```

```
Creating a new React app in /home/juancarlos/react/login.
```

Success! Created login at /home/juancarlos/react/login
Inside that directory, you can run several commands:

`npm start`

Starts the development server.

`npm run build`

Bundles the app into static files for production.

`npm test`

Starts the test runner.

`npm run eject`

Removes this tool and copies build dependencies, configuration files and scripts into the app directory. If you do this, you can't go back!

We suggest that you begin by typing:

`cd login`

`npm start`

Happy hacking!



Edit `src/App.js` and save to reload.

[Learn React](#)

Una vez creada la aplicación y lanzado el navegador el aspecto que tiene que presentar es el que se puede ver en la imagen anterior.

PASO 1. Instalar reactstrap

Nuestro proyecto va a utilizar la librería ReactStrap por lo tanto habrá que instalar no solamente esta librería sino la librería de Bootstrap en el proyecto.

A continuación se muestran los comandos que hay que lanzar desde la terminal si estamos utilizando Ubuntu Linux:

```
npm install reactstrap react react-dom
```

```
npm install --save bootstrap
```

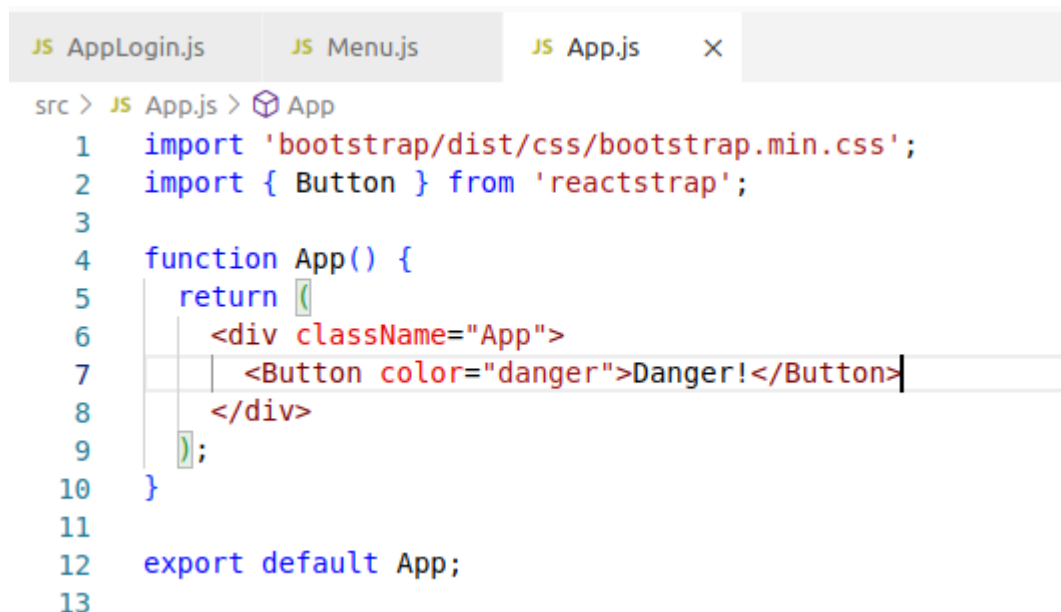
Además, habrá que importar la librería de Bootstrap dentro del fichero a App.js de nuestra aplicación:

```
import 'bootstrap/dist/css/bootstrap.min.css';
```

Para comprobar si funciona tanto la librería instalada como los componentes de la misma, lo que vamos a hacer es importar un botón de la librería y mostrarlo en el interfaz de usuario:

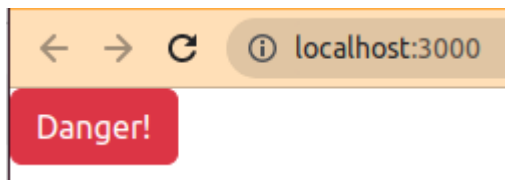
```
import { Button } from 'reactstrap';
```

```
<Button color="danger">Danger!</Button>;
```



```
JS AppLogin.js JS Menu.js JS App.js X
src > JS App.js > App
1  import 'bootstrap/dist/css/bootstrap.min.css';
2  import { Button } from 'reactstrap';
3
4  function App() {
5    return (
6      <div className="App">
7        <Button color="danger">Danger!</Button>
8      </div>
9    );
10 }
11
12 export default App;
13
```

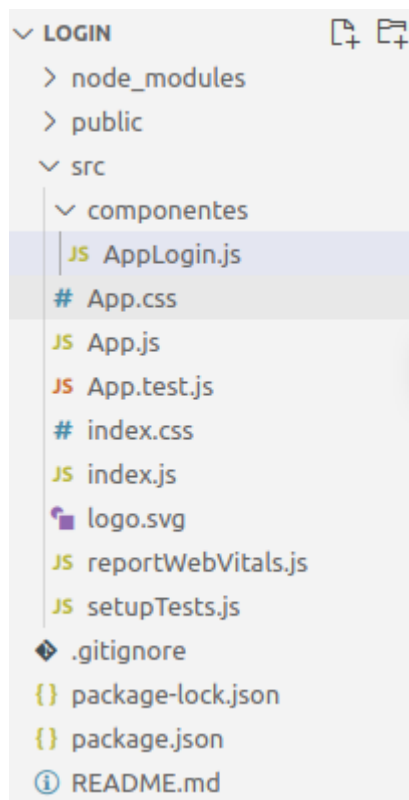
El código resultante se puede ver en la imagen anterior. Si todo va correctamente como es de esperar, el navegador deberá demostrar un botón rojo como el que se aprecia en la siguiente imagen:



PASO 2. Crear el componente login

Es el momento de empezar a crear nuevos componentes y para ello vamos a crear una ventana de login.

Utilizaremos para ello un nuevo componente llamado a AppLogin en el cual crearemos un pequeño formulario para que el usuario pueda introducir el usuario y la Password y validarse:



Es una buena práctica ubicar los componentes en un directorio carpeta aparte, porque si no se amontona el código y la reutilización del mismo es más complicada y poco efectiva.

El nuevo componente creado tendrá el siguiente código Fuente:

```
import React from 'react';
```

```

import
{Row, Col, Card, CardTitle, CardText, Form, FormGroup, Button, Label, Input}
from 'reactstrap';

export default function AppLogin(props) {

  return (
    <Row>
    <Col sm="4"></Col>
    <Col sm="4">
      <Card body>
        <CardTitle className="text-center" tag="h4">
          Log in
        </CardTitle>
        <Form inline>
          <FormGroup className="mb-2 me-sm-2 mb-sm-0">
            <Label className="me-sm-2" for="exampleEmail">User id</Label>
            <Input
              id="Telefono"
              name="telefono"
              placeholder="type your user id"
              type="email"
            />
          </FormGroup>
          <FormGroup className="mb-2 me-sm-2 mb-sm-0">
            <Label className="me-sm-2"
for="examplePassword">Password</Label>
            <Input
              id="Password"
              name="password"
              type="password"
            />
          </FormGroup>
          <br/>
          <Button color="primary" size="lg" block >
            <strong>Log in</strong>
          </Button>
          <CardText className="text-danger">{"INFORMACIÓN"}</CardText>

        </Form>
      </Card>
    </Col>
  </Row>

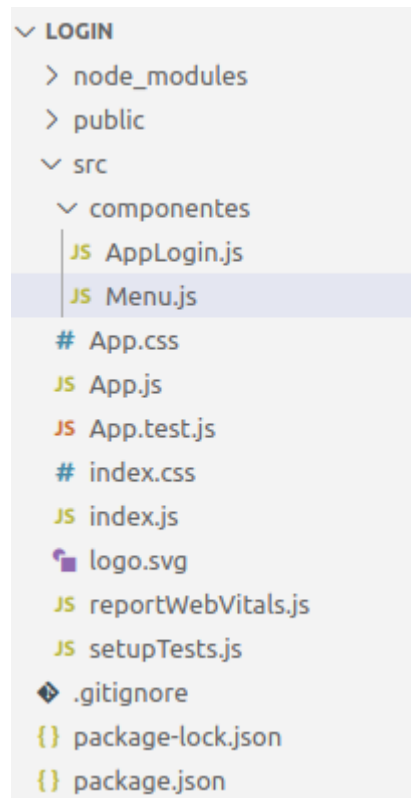
```

```
)  
}
```

Accesoriamente a este componente para validar el usuario y la Password, lo que vamos a crear es otro componente llamado menú el cual nos permitirá mostrar tres opciones al usuario. La codificación del componente se hará en el siguiente apartado de este tutorial:

PASO 3. Crear el componente Menu

Al igual que hicimos con el componente a AppLogin, ahora crearemos un nuevo archivo y dentro de el el componente menú que queremos diseñar:



Para la creación de este componente lo que vamos a utilizar son componentes de la librería **reactstrap** (**Navbar**, **NavbarBrand**, **NavLink**) que nos van a generar una barra de menú. El componente nuevo tendrá el siguiente código:

```
import React, { useState } from 'react';  
import {  
  Navbar,  
  NavbarBrand,  
  NavLink,  
  Button,  
} from 'reactstrap';  
  
export default function Menu(props) {
```

```

let colorUno = 'secondary'
let colorDos = 'secondary'
let colorTres = 'secondary'
switch (props.menuItem) {
  case 'UNO':
    colorUno = 'primary'
    break;
  case 'DOS':
    colorDos = 'primary'
    break;
  case 'TRES':
    colorTres = 'primary'
    break;
}
return (
  <div>
    <Navbar>
      <NavbarBrand href="/">MYFPSCHOOL</NavbarBrand>
      <NavLink>
        <Button color={colorUno}>UNO</Button>{" "}
        <Button color={colorDos}>DOS</Button>{" "}
        <Button color={colorTres}>TRES</Button>
      </NavLink>
    </Navbar>
  </div>
);
}

```

PASO 4. Cambiando el componente principal App

Ya es el momento de empezar a crear la aplicación en sí misma. Para ello lo primero que tenemos que hacer es cambiar el componente función por un componente de tipo Class.

```
class App extends Component {
```

A este componente se le va a añadir también un constructor en el cual vamos a guardar dos variables. La primera nos va a indicar si el usuario se ha logueado correctamente y la otra nos va a indicar la opción de menú elegida por el usuario:

```
constructor(props) {  
  super(props);  
  this.state = {  
    menuItem: "UNO",  
    logged: false,  
  }  
}
```

Vamos a generar una función que nos permita cambiar un elemento del estado de nuestro componente.

```
changeMenu(item){  
  this.setState({menuItem:item})  
}
```

Necesitaremos también una función que nos permita validar el usuario y la Password que introduce nuestro cliente de la aplicación. Como no vamos a utilizar una base de datos, la validación la haremos HardCoded.

En una aplicación real, el usuario junto con su Password habría que validarlo en algún tipo de servicio o bien en una base de datos.

Para simplificar nuestra aplicación utilizaremos un método más de andar por casa.

```
userLogin(telefono,password){  
  if (telefono=="Myfpschool" && password=="2023"){  
    this.setState({logged:true})  
  }  
}
```

Para renderizar la aplicación vamos a usar un re renderizado dependiente del estado de nuestra aplicación. Si el usuario se ha logueado correctamente mostrará una información y en caso contrario nos mostrará la ventana de login.


```

render(){
  let obj=<<Menu
  | | | | changeMenu={(item)=>this.changeMenu(item)}/></>

  if (!this.state.logged){
    obj= <AppLogin
    userLogin={(telefono,password)=>this.userLogin(telefono,password)}
    />
  }

  return (
    <div className="App">
      <Menu menuItem={this.state.menuItem} changeMenu={(item)=>this.changeMenu(item)}/>
      {obj}
    </div>
  );
}

```

RESULTADO FINAL DEL COMPONENTE APP

```

import 'bootstrap/dist/css/bootstrap.min.css';
import React, { Component } from 'react';
import Menu from './componentes/Menu'
import AppLogin from './componentes/AppLogin'

class App extends Component {
  constructor(props) {
    super(props);
    this.state = {
      menuItem: "UNO",
      logged: false,
    }
  }

  changeMenu(item) {
    this.setState({menuItem:item})
  }

  userLogin(telefono,password) {
    if (telefono=="Myfpschool" && password=="2023") {
      this.setState({logged:true})
    }
  }
}

```

```

    }

    render() {

        let obj=<><Menu
            changeMenu={ (item)=>this.changeMenu(item) }/></>

        if (!this.state.logged) {
            obj= <AppLogin
                userLogin={ (telefono,password)=>this.userLogin(telefono,password) }
            />
        }

        return (
            <div className="App">
                <Menu menuItem={this.state.menuItem}
                    changeMenu={ (item)=>this.changeMenu(item) }/>
                {obj}
            </div>
        );
    }

}

export default App;

```

Si todo funciona correctamente, el resultado de ejecutar el código anterior será el siguiente
 si todo correo funciona correctamente, el resultado de ejecutar el código anterior será el siguiente:

MYFPSCHOOL

UNO DOS TRES

Log in

User id

Password

Log in

INFORMACION

PASO 5. MEJORANDO EL COMPONENTE LOGIN

Es el momento de ir completando la aplicación y por lo tanto vamos a necesitar la utilización de Hooks en nuestra aplicación. Utilizaremos el Hook useState.

Lo primero que tenemos que hacer es importar lo de la librería React.

```
import React, { useState } from 'react';
```

A continuación vamos a crear los tres Hooks. Para el usuario lo que vamos a utilizar es un teléfono aunque al ser un campo de texto podría saber vivir también nombre de usuario.

```
const [password, setPassword] = useState('');  
const [telefono, setTelefono] = useState('');  
const [info, setInfo] = useState('');
```

Deberemos modificar también los campos de texto para poder recoger la información que el usuario va insertando en ellos con lo cual tendremos que utilizar un Listener para el evento o onChange:

```
<Input  
  id="Telefono"  
  name="telefono"  
  placeholder="type your user id"  
  type="email"  
  onChange={handleChange}  
/>
```

En este Listener lo que vamos a ir es guardando la información que vaya cumplimentando el usuario en los Hooks correspondientes:

```
const handleChange = (event) => {  
  setInfo('');  
  const target = event.target;  
  if (target.name === "password") {  
    setPassword(target.value);  
  }  
  if (target.name === "telefono") {  
    setTelefono(target.value);  
  }  
}
```

Para hacer una validación sencillita, lo que vamos a hacer es que cuando el usuario haga clic en el botón de login, se haga una primera validación para ver si los datos introducidos son correctos:

```
<Button color="primary" size="lg" block onClick={clicar}>
  <strong>Log in</strong>
</Button>
```

```
const clicar = ()=>{
  if (password===''||telefono===''){
    setInfo('Cumplimente todos los campos');
  }else{
    if (telefono=="Myfpschool" && password=="2023"){
      props.userLogin(telefono,password)
    }else{
      setInfo('DATOS INCORRECTOS')
    }
  }
}
```

En el caso de que se introduzcan datos incorrectos nos aparecerá un pequeño cartel debajo del botón indicando este error:

MYFPSCHOOL

UNO

DOS

TRES

Log in

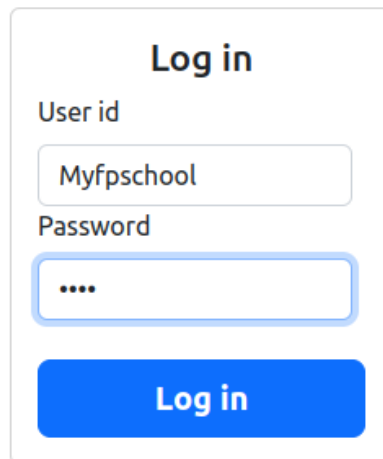
User id

Password

Log in

DATOS INCORRECTOS

En caso contrario, nuestra aplicación debería de dejar de mostrar la ventana de login y montar otra información que se mostraría al usuario:



The image shows a login form titled "Log in". It has two input fields: "User id" with the value "Myfpschool" and "Password" with four dots. Below the fields is a blue button labeled "Log in".

ASPECTO DEL COMPONENTE HASTA AHORA

```
import React, { useState } from 'react';
import
{Row, Col, Card, CardTitle, CardText, Form, FormGroup, Button, Label, Input}
from 'reactstrap';

export default function AppLogin(props) {
  const [password, setPassword] = useState('');
  const [telefono, setTelefono] = useState('');
  const [info, setInfo] = useState('');

  const handleChange = (event) => {
    setInfo('');
    const target = event.target;
    if (target.name == "password") {
      setPassword(target.value);
    }
    if (target.name == "telefono") {
      setTelefono(target.value);
    }
  }

  const clicar = () => {
    if (password == '' || telefono == '') {
      setInfo('Cumplimente todos los campos');
    } else {

```

```

        if (telefono=="Myfpschool" && password=="2023") {
            props.userLogin(telefono,password)
        }else{
            setInfo('DATOS INCORRECTOS')
        }
    }
}

return (
<Row>
<Col sm="4"></Col>
<Col sm="4">
    <Card body>
        <CardTitle className="text-center" tag="h4">
            Log in
        </CardTitle>
        <Form inline>
            <FormGroup className="mb-2 me-sm-2 mb-sm-0">
                <Label className="me-sm-2" for="exampleEmail">User id</Label>
                <Input
                    id="Telefono"
                    name="telefono"
                    placeholder="type your user id"
                    type="email"
                    onChange={handleChange}
                />
            </FormGroup>
            <FormGroup className="mb-2 me-sm-2 mb-sm-0">
                <Label className="me-sm-2"
for="examplePassword">Password</Label>
                <Input
                    id="Password"
                    name="password"
                    type="password"
                    onChange={handleChange}
                />
            </FormGroup>
            <br/>
            <Button color="primary" size="lg" block onClick={clicar}>
                <strong>Log in</strong>
            </Button>
            <CardText className="text-danger">{info}</CardText>

```

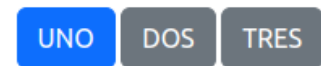
```

    </Form>
  </Card>
</Col>
</Row>
)
}

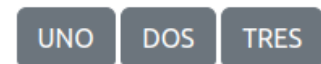
```

Una vez ejecutado el programa he introducido los datos correspondientes de forma correcta el resultado tiene que ser el siguiente. Como se puede observar se repite el mismo componente dos veces pero en el siguiente apartado vamos a corregir este funcionamiento no correcto.

MYFPSCHOOL



MYFPSCHOOL



PASO 6 COMPLETANDO LA APLICACIÓN

Necesitamos hacer ciertos cambios en nuestra aplicación en el componente menú. Se necesitaría recordar que opción del menú principal se ha seleccionado con lo cual habrá que capturar el evento clic de cada uno de los botones que se muestran en el menú.

Cambios en Menu.js

```

return (
  <div>
    <Navbar>
      <NavbarBrand href="/">MYFPSCHOOL</NavbarBrand>
      <NavLink>
        <Button color={colorUno} onClick={()=>props.changeMenu("UNO")}>UNO</Button>{" "}
        <Button color={colorDos} onClick={()=>props.changeMenu("DOS")}>DOS</Button>{" "}
        <Button color={colorTres} onClick={()=>props.changeMenu("TRES")}>TRES</Button>
      </NavLink>
    </Navbar>
  </div>
);

```

El resultado tiene que ser el que aparece a continuación tras pulsar los botones dos y tres puntos



ASPECTO DE App.js HASTA AHORA

```
import 'bootstrap/dist/css/bootstrap.min.css';
import React, { Component } from 'react';
import Menu from '../componentes/Menu'
import AppLogin from '../componentes/AppLogin'

class App extends Component {
  constructor(props) {
    super(props);
    this.state = {
      menuItem: "UNO",
      logged: false,
    }
  }

  changeMenu(item) {
    this.setState({menuItem: item})
  }

  userLogin(telefono, password) {
    if (telefono === "Myfpschool" && password === "2023") {
      this.setState({logged: true})
    }
  }

  render() {

    let obj=<></>
```



```
if (!this.state.logged) {
  obj= <AppLogin
  userLogin={ (telefono,password)=>this.userLogin(telefono,password) }
  />
}

return (
  <div className="App">
    <Menu menuItem={this.state.menuItem}
changeMenu={ (item)=>this.changeMenu(item) } />
    {obj}
  </div>
  );
}

}

export default App;
```