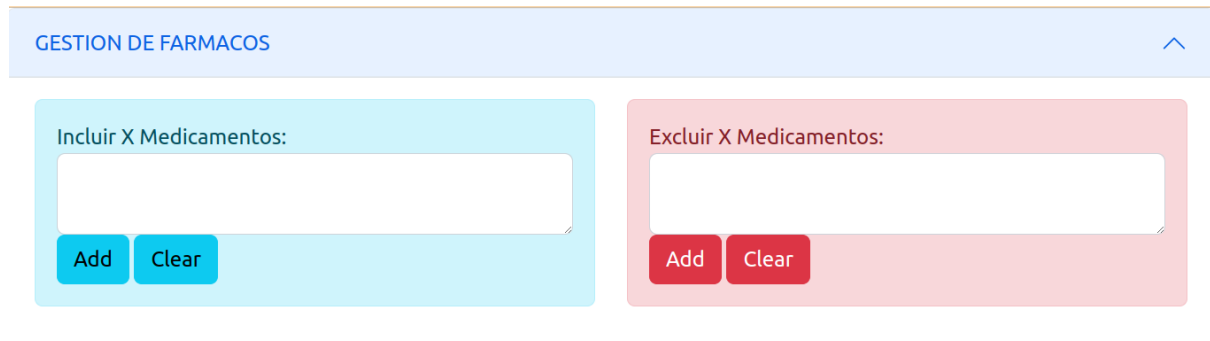


EXAMEN REACTJS. GESTIÓN DE FÁRMACOS

Se pide que hagas una pequeña aplicación que controle la entrada de fármacos en un campo. Para ello vamos a valernos de una ventana modal la cual permitirá filtrar los fármacos e incorporarlos a dos campos.

La aplicación tiene el siguiente aspecto:



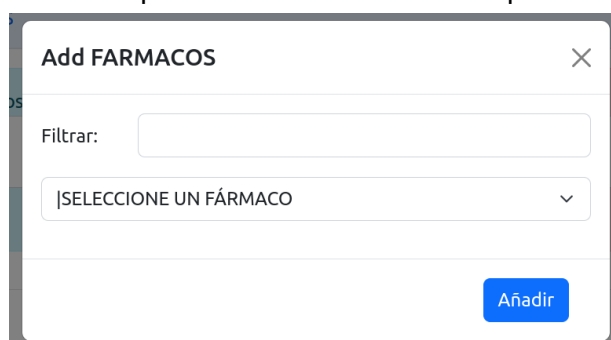
Los componentes reactstrap se proporcionarán al estudiante con lo cual no tienes que preocuparte del aspecto visual sino solamente de la programación.

Se proporcionará un archivo llamado datos.js que contendrá los datos de la aplicación. Este archivo tiene el siguiente formato:

```
export const FARMACOS = [
  {codATC:"",descATC:"SELECCIONE UN FÁRMACO"},
  {codATC:"01",descATC:"ALGODON"},
  {codATC:"01A",descATC:"ALGODON ARROLLADO"},
  {codATC:"01A01",descATC:"ALGODON ARROLLADO PURO"}]
```

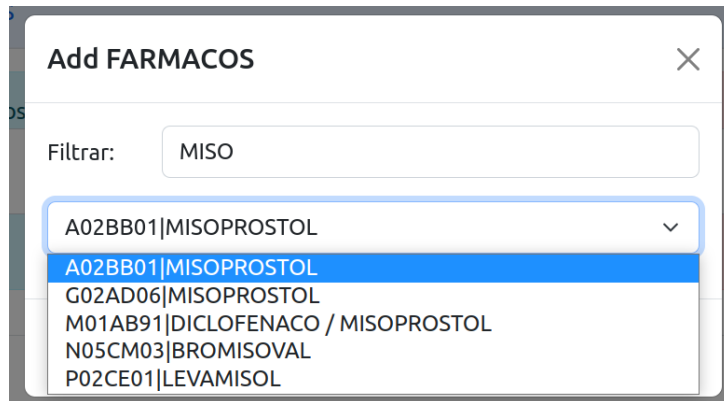
Obviamente el archivo proporcionado tiene muchos más registros.
Funcionalidad.

1 Cuando pulsamos los botones Add aparece nuestra ventana modal de fármacos:



En el combo aparecerán todos los fármacos disponibles.

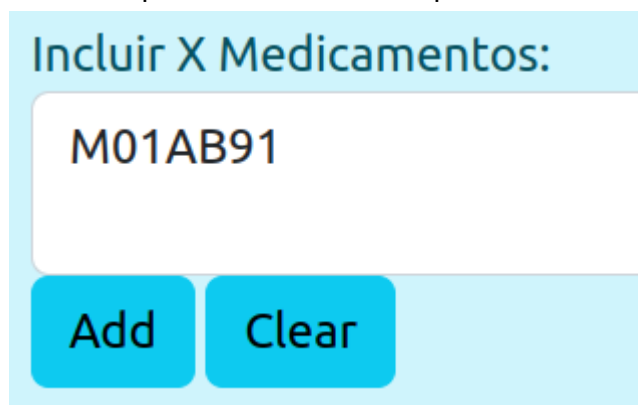
2 Cuando introduzco parte de un nombre en filtrar, solamente me aparecen los fármacos que contienen esa cadena:



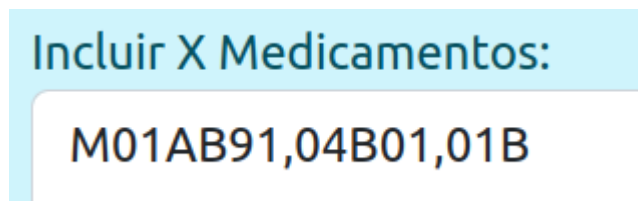
3 Cuando selecciono un fármaco concreto, aparece en la ventana modal al lado del botón de añadir:

M01AB91|DICLOFENACO / MISOPROSTOL Añadir

4 Si le pulso añadir se añade el **Código del fármaco** al campo donde pulsé el botón Add. Esto es importante añadir al campo correcto.



Si se añaden varios fármacos se van separando con comas para luego poder procesarlos.



5 Si pulso **clear** en incluir o excluir, borrará solamente el campo correspondiente.

Se proporcionará lo siguiente:

- 1 Fichero datos.js con todos los datos de medicamentos
- 2 Fichero App.js con la aplicación medio terminada. Tendrás que meterle la funcionalidad anteriormente descrita.

Cosas que se valorarán:

- Realiza la funcionalidad que se pide manteniendo el código proporcionado.
- En cuanto al código

- Legibilidad del código.
- Longitud del código.
- La modularidad (alta cohesión y bajo acoplamiento). Solamente tienes que completar el código.
- Que el código sea limpio, flexible, reutilizable y mantenible.
- No se valorará el aspecto estético de la web.
- Que no haya errores en la consola.
- La eficiencia.
- Sigue los principios SOLID.
- Otros criterios.

CÓDIGO STARTER (App.js)

```
import React, { useState, Component } from 'react';
import 'bootstrap/dist/css/bootstrap.min.css';
import {
  Alert, Row, Col, UncontrolledAccordion, AccordionItem,
  AccordionHeader, AccordionBody, Input, Button, Modal, ModalHeader,
  ModalBody, ModalFooter, FormGroup, Label
} from 'reactstrap';
import { FARMACOS } from '../componentes/datos';

const VentanaModalDiccionario = (props) => {
  const {
    className
  } = props;

  const handleChange = (event) => {
    // COMPLETA ESTA FUNCION
  }

  return (
    <div>
      <Modal isOpen={props.mostrar} toggle={props.toggle}
        className={className} onEntering={"/ESTO SE EJECUTA CUANDO MUESTRAS LA VENTANA"}>
        <ModalHeader toggle={props.toggle}>{props.titulo}</ModalHeader>
        <ModalBody>
          <FormGroup row>
```

EXAMEN REACTJS. GESTIÓN DE FÁRMACOS

```
    }  
  }  
  
  handleChange = (event) => {  
  }  
  
  add(datos) {  
    this.toggleModal();  
  }  
  
  setIsOpen(d) {  
    if (d==undefined) return;  
    this.setState({isOpen:d})  
  }  
  
  toggleModal() { this.setIsOpen(!this.state.isOpen); }  
  
  render() {  
    return (  
      <>  
      <div>  
        <UncontrolledAccordion  
          defaultOpen={ [  
            '1'  
          ] }  
          stayOpen  
        >  
          <AccordionItem>  
            <AccordionHeader targetId="1">  
              GESTION DE FARMACOS  
            </AccordionHeader>  
            <AccordionBody accordionId="1">  
              <Row>  
  
                <Col>  
                  <Alert color="info">  
                    Incluir X Medicamentos:  
                    <Input type="text" value={this.state.rxseleccionar} />  
                    <input type="button" value="Agregar" />  
                  </Alert>  
                </Col>  
              </Row>  
            </AccordionBody>  
          </AccordionItem>  
        </UncontrolledAccordion>  
      </div>  
    )  
  }  
}
```

```

        <Button
onClick={ ()=>{this.toggleModal()}} color="info">Add</Button>
        { " "}<Button color="info"
onClick={""}>Clear</Button>
        </Alert>
    </Col>
    <Col>
        <Alert color="danger">
            Excluir X Medicamentos:
            <Input type="text" name="rxenmascarar"
onChange={this.handleChange.bind(this)}
value={this.state.rxenmascarar}/>
            <Button
onClick={ ()=>{this.toggleModal()}} color="danger">Add</Button>
            { " "}<Button color="danger"
onClick={""}>Clear</Button>
            </Alert>
        </Col>
    </Row>
</AccordionBody>
</AccordionItem>
</UncontrolledAccordion>
</div>
    <VentanaModalDiccionario diccionario={this.state.diccionario}
add={ (datos)=>this.add(datos)} mostrar={this.state.isOpen} aceptar =
{"Añadir"} toggle={ ()=>this.toggleModal()} titulo={"Add
"+this.state.diccionario}/>
    <br/>
</>
    );
}
}

class App extends Component {
  render() {

    return (
      <div className="App">
        <Filter />
      </div>
    );
  }
}

```

```
}  
  
export default App;
```