FINIFI - MERN Stack Assignment

InvoiceFilter.js:

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

import { TextField, MenuItem, Select, FormControl, InputLabel, Button } from '@mui/material';

const statuses = [

  'Open', 'Awaiting Approval', 'Approved', 'Processing', 'Paid', 'Rejected', 'Vendor Not Found', 'Duplicate',
'Void'

];

export default function InvoiceFilter({ onFilterChange, onCreateInvoice }) {

  const [status, setStatus] = useState('');

  const [searchTerm, setSearchTerm] = useState('');

  const handleSearchChange = (e) => setSearchTerm(e.target.value);

  const handleStatusChange = (e) => setStatus(e.target.value);

  return (

    <div className="flex space-x-4 mb-6">

      <TextField

        label="Search by Vendor Name/Invoice #"

        variant="outlined"

        value={searchTerm}

        onChange={handleSearchChange}

        fullWidth

      />

      <FormControl variant="outlined" fullWidth>

        <InputLabel>Status</InputLabel>

        <Select value={status} onChange={handleStatusChange} label="Status">

          {statuses.map((status) =>
```

```
    <MenuItem key={status} value={status}>
            {status}
          </MenuItem>
        ))}
      </Select>
    </FormControl>
    <Button variant="contained" onClick={() => onFilterChange(searchTerm, status)}>Filter</Button>
    <Button variant="outlined" onClick={onCreateInvoice}>Create Invoice</Button>
  </div>
 );
}
```

models/Invoice.js:

```
const mongoose = require('mongoose');
const invoiceSchema = new mongoose.Schema({
 vendorName: { type: String, required: true },
 invoiceNumber: { type: String, required: true, unique: true },
 status: {
  type: String,
  enum: ['Open', 'Awaiting Approval', 'Approved', 'Processing', 'Paid', 'Rejected', 'Vendor Not Found',
'Duplicate', 'Void'],
  required: true
 },
 netAmount: { type: Number, required: true },
 invoiceDate: { type: Date, required: true },
 dueDate: { type: Date, required: true },
 department: { type: String, required: true },
```

```
  poNumber: { type: String },
  createdTime: { type: Date, default: Date.now },
  createdDate: { type: Date, default: Date.now },
});

module.exports = mongoose.model('Invoice', invoiceSchema);
```

CreateInvoiceModal.js:

```
import React, { useState } from 'react';
import { Dialog, DialogActions, DialogContent, DialogTitle, TextField, Button, MenuItem, FormControl, Select, InputLabel } from '@mui/material';

const statuses = [
  'Open', 'Awaiting Approval', 'Approved', 'Processing', 'Paid', 'Rejected', 'Vendor Not Found', 'Duplicate', 'Void'
];

export default function CreateInvoiceModal({ open, onClose, onSubmit }) {
  const [vendorName, setVendorName] = useState('');
  const [invoiceNumber, setInvoiceNumber] = useState('');
  const [status, setStatus] = useState('');
  const [netAmount, setNetAmount] = useState('');
  const [invoiceDate, setInvoiceDate] = useState('');
  const [dueDate, setDueDate] = useState('');
  const [department, setDepartment] = useState('');
  const [poNumber, setPoNumber] = useState('');

  const handleSubmit = () => {
```

```
  const data = { vendorName, invoiceNumber, status, netAmount, invoiceDate, dueDate, department, poNumber };

  onSubmit(data);

 };

 return (

  <Dialog open={open} onClose={onClose}>

   <DialogTitle>Create Invoice</DialogTitle>

   <DialogContent>

    <TextField label="Vendor Name" fullWidth value={vendorName} onChange={(e) => setVendorName(e.target.value)} />

    <TextField label="Invoice Number" fullWidth value={invoiceNumber} onChange={(e) => setInvoiceNumber(e.target.value)} />

    <FormControl fullWidth>

     <InputLabel>Status</InputLabel>

     <Select value={status} onChange={(e) => setStatus(e.target.value)} label="Status">

      {statuses.map((status) => (

       <MenuItem key={status} value={status}>

        {status}

       </MenuItem>

      ))}

     </Select>

    </FormControl>

    <TextField label="Net Amount" fullWidth value={netAmount} onChange={(e) => setNetAmount(e.target.value)} />

    <TextField label="Invoice Date" type="date" fullWidth value={invoiceDate} onChange={(e) => setInvoiceDate(e.target.value)} />

    <TextField label="Due Date" type="date" fullWidth value={dueDate} onChange={(e) => setDueDate(e.target.value)} />
```

```
      <TextField label="Department" fullWidth value={department} onChange={(e) =>
setDepartment(e.target.value)} />

      <TextField label="PO Number" fullWidth value={poNumber} onChange={(e) =>
setPoNumber(e.target.value)} />

    </DialogContent>

    <DialogActions>

      <Button onClick={onClose}>Cancel</Button>

      <Button onClick={handleSubmit}>Submit</Button>

    </DialogActions>

  </Dialog>

 );

}
```