Agregando Filtro de Gastos

- Creo el componente Filtros, le agrego unas clases y copio de Modal las opciones del select

```
const Filtros = () => {
  return (
    <div className="filtro sombra contenedor">
        <form >
            <div className="campo">
                <label>Filtrar Gastos</label>
                <select>
                    <option value="">--Seleccione</option>
                    <option value="ahorro">Ahorro</option>
                    <option value="comida">Comida</option>
                    <option value="casa">Casa</option>
                    <option value="gastos">Gastos Varios</option>
                    <option value="ocio">Ocio</option>
                    <option value="salud">Salud</option>
                    <option value="suscripciones">Suscripciones</option>
                </select>
            </div>
        </form>
    </div>
  )
}

export default Filtros
```

- Lo voy a usar en App.jsx, lo importo
- Como lo voy a mostrar antes del listado de gastos lo ubico correctamente
- Para no usar el .filter y tomar y devolver los arreglos filtrados con el objeto completo usaré el state filtro y setFiltro, iniciará como un arreglo vacío. Solo va a haber una opción. Si hubieran múltiples opciones podría hacerse con un arreglo
- Le paso filtro y setFiltro por props al componente Filtros
- Le coloco al select un value de filtro y en el onChange, cada vez que el usuario clique una opción lo colocaré en el setFiltro

```
<select
value={filtro}
onChange={(e)=>setFilter(e.target.value)}>
    <option value="">--Seleccione</option>
    <option value="ahorro">Ahorro</option>
    <option value="comida">Comida</option>
    <option value="casa">Casa</option>
    <option value="gastos">Gastos Varios</option>
    <option value="ocio">Ocio</option>
    <option value="salud">Salud</option>
    <option value="suscripciones">Suscripciones</option>
</select>
```

- De esta forma tengo el filtro aplicado en el App.jsx
- Puedo ir a components en devtools, App y puedo ver que si selecciono Comida, el último state se llena con comida
- Colocaré un useEffect para que escuche los cambios que sucedan en filtro
- Para que cuando cambie, voy a mandar llamar el arreglo de gastos y voy afiltrar según la categoría seleccionada
- Se va a ejecutar almenos una vez, por eso le pongo la condición
    - Si hay algo en el filtro, le pongo un console.log para probar
- Gastos filtrados será gastos, el arreglo que tiene todos los gastos.filter y que me retorne los gastos que pertenezcan a la categoría de filtro
- Lo hago con un return implicito, por eso está entre paréntesis

```
useEffect(()=>{
  if(filtro){
    const gastosFiltrados= gastos.filter(gasto => (gasto.categoria === filtro))
  }

}, [filtro])
```

- Hay que retornar esos gastos en un nuevo state, porque si lo hagio con gastos, voy a perder la referencia al resto de gastos.
- Creo otro state con gastosFiltrados, setGastosFiltrados
- Va a iniciar como un arreglo vacío, porque puede ser que haya más de un elemento
- Entonces, para cambiar el state uso el setGastosFiltrados con gastosFiltrados

```
<select
value={filtro}
onChange={(e)=>setFilter(e.target.value)}>
    <option value="">--Seleccione</option>
    <option value="ahorro">Ahorro</option>
    <option value="comida">Comida</option>
    <option value="casa">Casa</option>
    <option value="gastos">Gastos Varios</option>
    <option value="ocio">Ocio</option>
    <option value="salud">Salud</option>
    <option value="suscripciones">Suscripciones</option>
</select>
```

- El siguiente paso es : si hay un filtro activo, hay que pasar el arreglo de gastos filtrados y el filtro hacia el componente ListadoGastos

Le paso filtro y gastosFiltrados por props a ListadoGastos en App.jsx

- Le digo que si filtro existe, quiero iterar sobre los gastosFiltrados.
- Si no hay nada voy a filtrar sobre gastos

```
<div className="listado-gastos contenedor">
    <h2>{gastos.length ? "Gastos" : "no hay gastos aún"}</h2>

      {
        filtro ? (
          gastosFiltrados.map(gasto=>(
              <Gasto
              key={gastos.id}
              gasto={gasto}
              setGastoEditar={setGastoEditar}
              eliminarGasto={eliminarGasto}/>
          ))

        ): (
          gastos.map(gasto=>(
            <Gasto
            key={gastos.id}
            gasto={gasto}
            setGastoEditar={setGastoEditar}
            eliminarGasto={eliminarGasto}/>
        ))

        )
      }
</div>
```

- Ahora si elijo un gasto que no hay en el select aparece Gastos.
- Lo ideal sería que dijera: No hay gastos de esta categoría, o algo así.
- Para ello coloco la lógica de gastosFiltrados en un fragment
- Cómo está dentro de un fragment tengo que volverlo a meter entre llaves para decirle que es código de javascript
- Le coloco el ternario

```
{
  filtro ? (
    <>

    <h2>{gastosFiltrados.length ? "Gastos" : "no hay gastos aún"}</h2>
   {gastosFiltrados.map(gasto=>(
     <Gasto
     key={gastos.id}
     gasto={gasto}
     setGastoEditar={setGastoEditar}
     eliminarGasto={eliminarGasto}/>
     ))}
    </>
  ): (
    <>

      <h2>{gastos.length ? "Gastos" : "no hay gastos aún"}</h2>
    {gastos.map(gasto=>(
      <Gasto
      key={gastos.id}
      gasto={gasto}
      setGastoEditar={setGastoEditar}
      eliminarGasto={eliminarGasto}/>
  ))}
    </>
  )
}
>
```