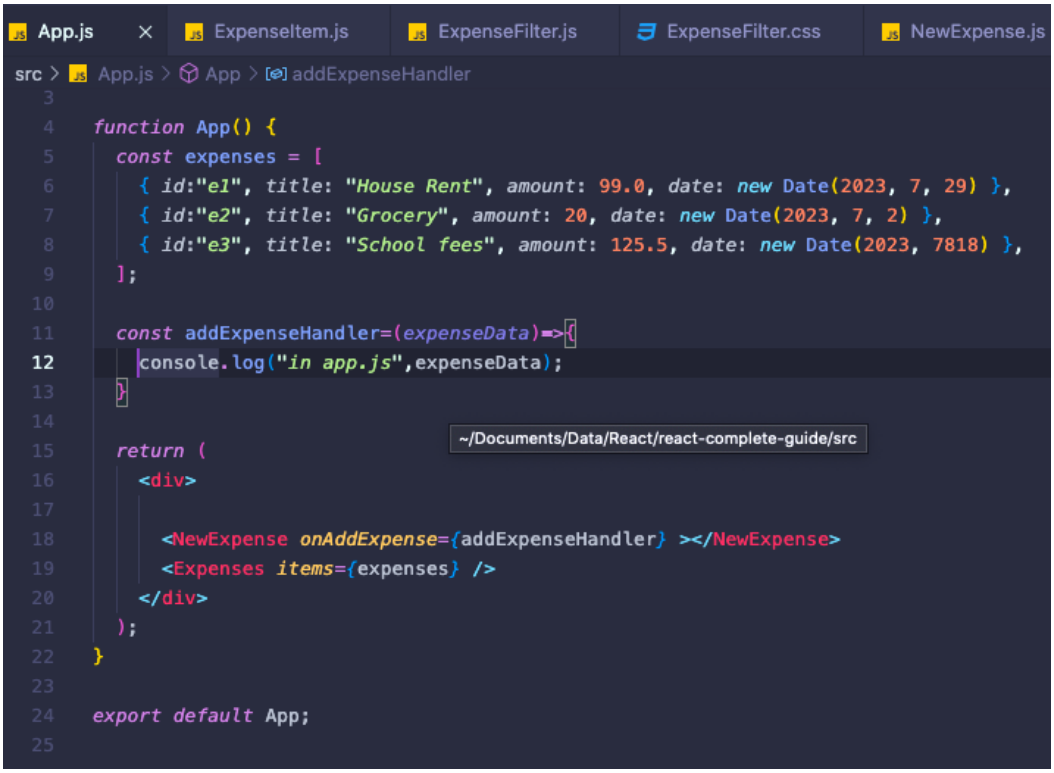
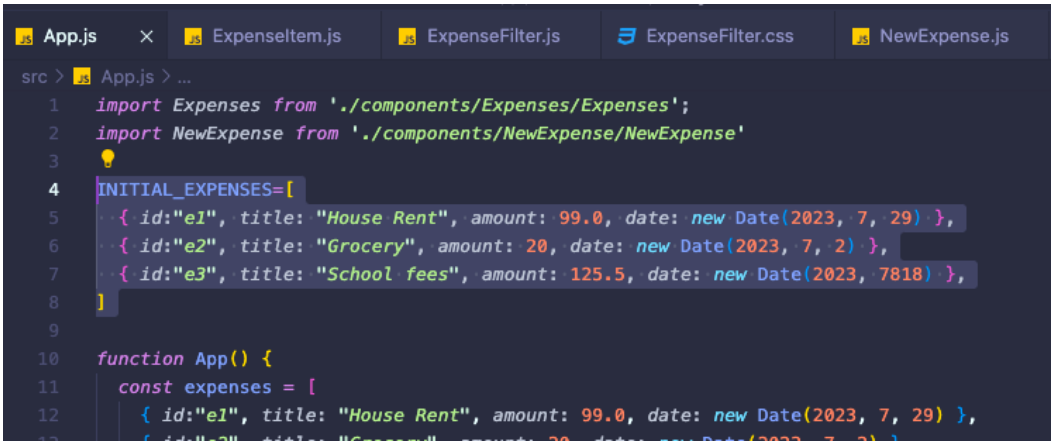
17. Statefull lists(updating the Expenses list when new expense item is added)

- Until now, in app.js we have a static list
  - 
    ```
    App.js | ExpenseItem.js | ExpenseFilter.js | ExpenseFilter.css | NewExpense.js
    src > App.js > App > addExpenseHandler
    3
    4   function App() {
    5     const expenses = [
    6       { id:"e1", title: "House Rent", amount: 99.0, date: new Date(2023, 7, 29) },
    7       { id:"e2", title: "Grocery", amount: 20, date: new Date(2023, 7, 2) },
    8       { id:"e3", title: "School fees", amount: 125.5, date: new Date(2023, 7818) },
    9     ];
    10
    11    const addExpenseHandler=(expenseData)=>{
    12      console.log("in app.js",expenseData);
    13    }
    14
    15    return (                    ~/Documents/Data/React/react-complete-guide/src
    16      <div>
    17
    18        <NewExpense onAddExpense={addExpenseHandler} ></NewExpense>
    19        <Expenses items={expenses} />
    20      </div>
    21    );
    22  }
    23
    24  export default App;
    25
    ```
  - Here we are just console logging the newly entered data from other component
  - Instead of console.log we have to update the expenses list so that it should display the new list in our page
- For this, let us have a initial list,
  - 
    ```
    App.js | ExpenseItem.js | ExpenseFilter.js | ExpenseFilter.css | NewExpense.js
    src > App.js > ...
    1   import Expenses from './components/Expenses/Expenses';
    2   import NewExpense from './components/NewExpense/NewExpense'
    3
    4   INITIAL_EXPENSES=[
    5     { id:"e1", title: "House Rent", amount: 99.0, date: new Date(2023, 7, 29) },
    6     { id:"e2", title: "Grocery", amount: 20, date: new Date(2023, 7, 2) },
    7     { id:"e3", title: "School fees", amount: 125.5, date: new Date(2023, 7818) },
    8   ]
    9
    10  function App() {
    11    const expenses = [
    12      { id:"e1", title: "House Rent", amount: 99.0, date: new Date(2023, 7, 29) },
    ```
- Now let us, create a state variable to hold a list,

-

App.js | ExpenseItem.js | ExpenseFilter.js | ExpenseFilter.css | NewExpen

src > App.js > App

```
4
5   let INITIAL_EXPENSES=[
6     { id:"e1", title: "House Rent", amount: 99.0, date: new Date(2023, 7, 29) },
7     { id:"e2", title: "Grocery", amount: 20, date: new Date(2023, 7, 2) },
8     { id:"e3", title: "School fees", amount: 125.5, date: new Date(2023, 7818) },
9   ]
10
11  function App() {
12    // const expenses = [
13    //   { id:"e1", title: "House Rent", amount: 99.0, date: new Date(2023, 7, 29
14    //   { id:"e2", title: "Grocery", amount: 20, date: new Date(2023, 7, 2) },
15    //   { id:"e3", title: "School fees", amount: 125.5, date: new Date(2023, 781
16    // ];
17    const [expenses, setExpenses]=useState(INITIAL_EXPENSES);
18
19    const addExpenseHandler=(expenseData)=>{
20      console.log("in app.js",expenseData);
21    }
22
23    return (
24      <div>
25
26        <NewExpense onAddExpense={addExpenseHandler} ></NewExpense>
27        <Expenses items={expenses} />
28      </div>
29    );
30  }
```

- Now instead of logging the newly entered expense item, we have to update the state variable

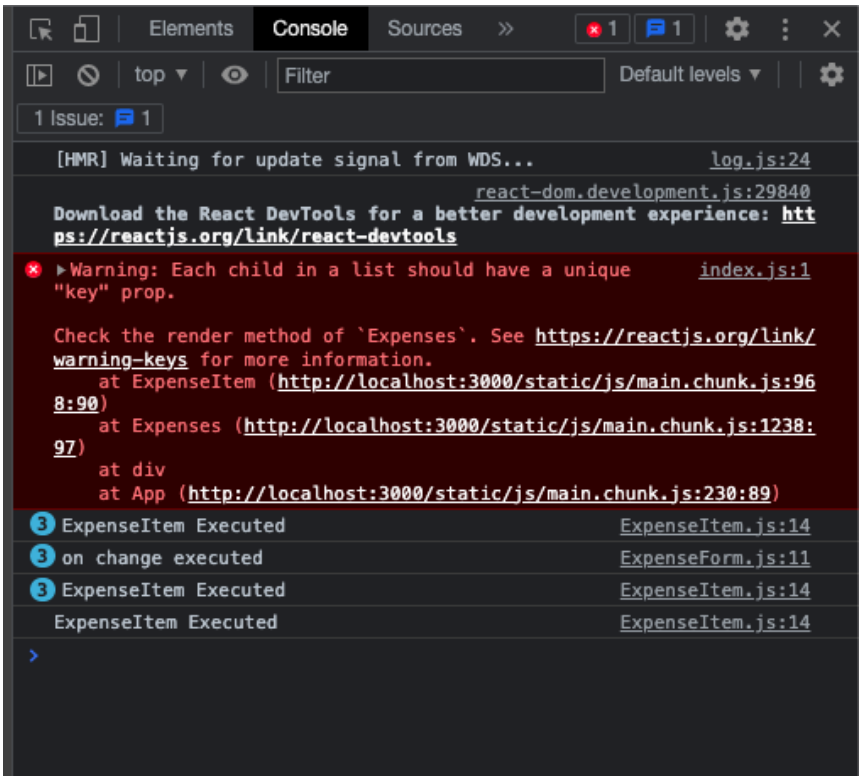
App.js | ExpenseItem.js | ExpenseFilter.js | ExpenseFilter.css | NewExpense.js

src > App.js > App > addExpenseHandler

```
11  function App() {
12    // const expenses = [
13    //   { id:"e1", title: "House Rent", amount: 99.0, date: new Date(2023, 7, 29) },
14    //   { id:"e2", title: "Grocery", amount: 20, date: new Date(2023, 7, 2) },
15    //   { id:"e3", title: "School fees", amount: 125.5, date: new Date(2023, 7818) },
16    // ];
17    const [expenses, setExpenses]=useState(INITIAL_EXPENSES);
18
19    const addExpenseHandler=(expenseData)=>{
20      // console.log("in app.js",expenseData);
21      // setExpenses([expenseData,...expenses]);
22      setExpenses((prevState)=>[expenseData,...prevState]);
23    }
24
25    return (
26      <div>
27
28        <NewExpense onAddExpense={addExpenseHandler} ></NewExpense>
29        <Expenses items={expenses} />
30      </div>
31    );
32  }
33
34  export default App;
35
```

- As we seen earlier, this is the proper method to update the state variable because react just schedules the state change, so to keep the data shown in frontend consistent we need to use the latest snapshot of the state.

- **NOTE:**
  - **It may not work properly if we haven't done this yet**

- 

- **So let us add key, while rendering the list**
-