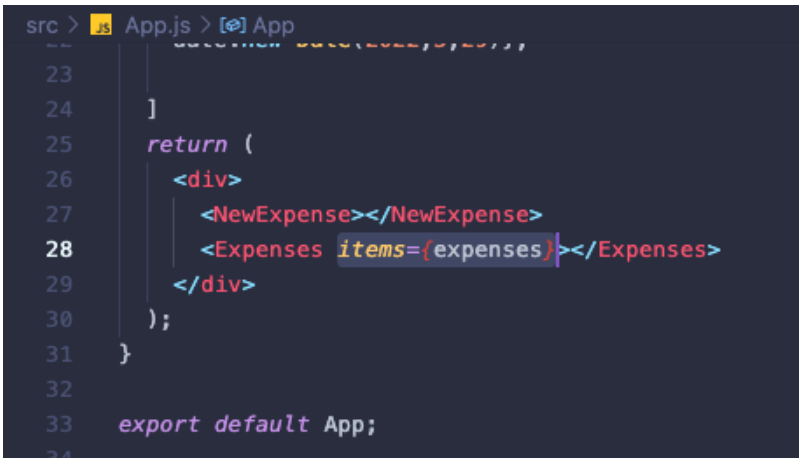
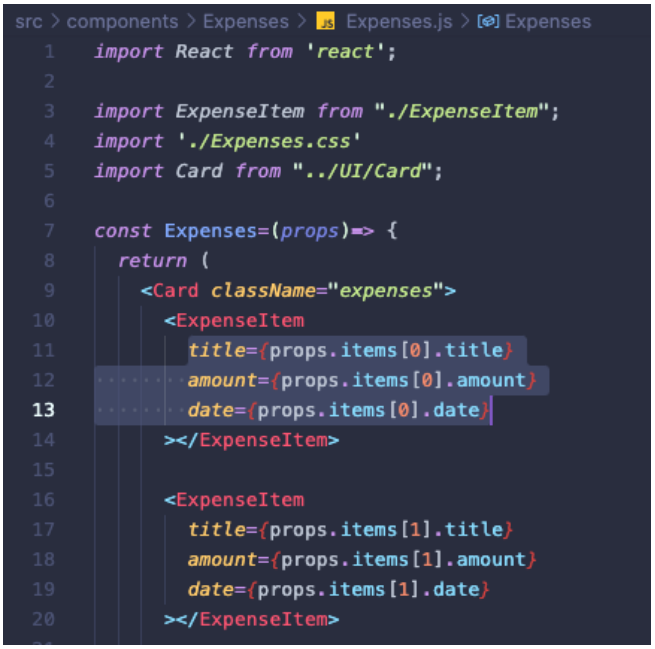
23. Passing props from child to parent(Bottom-up Approach)

- Until now we have learnt how to pass props from parent to child
  - Let us see the eg of parent to child
  - 

    - here in app.js we are passing the expenses list to Expenses component
  - 

    - Here in Expenses.js we are accessing the list which was passed from app.js to Expenses component and here in Expense.js we are passing each elements separately to expenseitem.js

src > components > Expenses > ExpenseItem.js > ExpenseItem

```
const ExpenseItem=(props)=> {

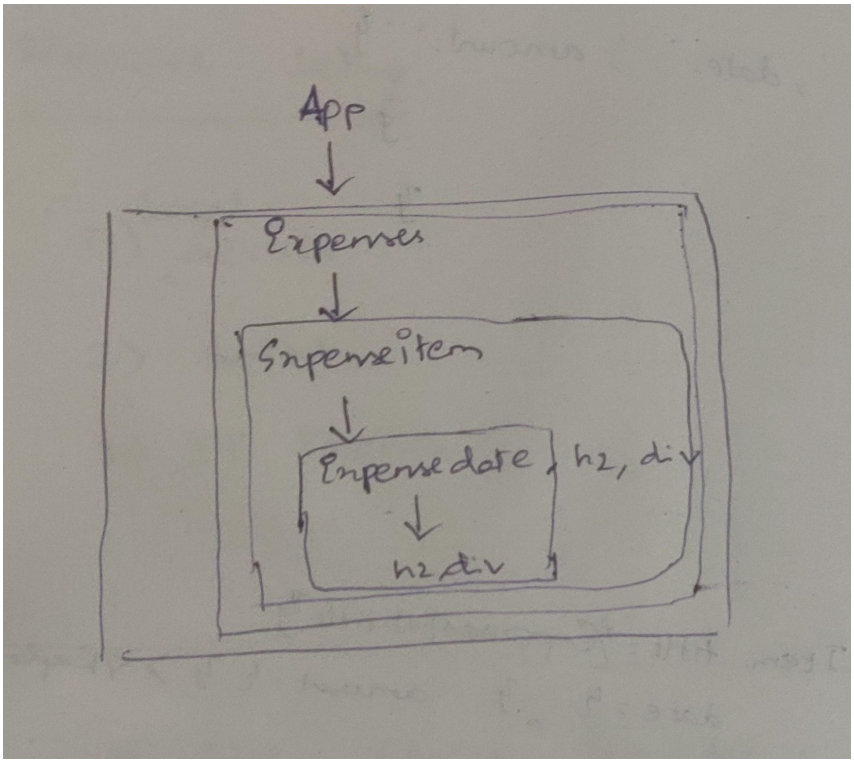
  const [title,setTitle]=useState(props.title);

  const clickHandler =()=>{
    setTitle("updated title");
    console.log("current title="+title);
  }

  return (
    <Card className="expense-item">
      <ExpenseDate date={props.date} ></ExpenseDate>
      <div className="expense-item__description">
        <h2>{title}</h2>
        <div className="expense-item__price">${props.amount}</div>
      </div>
      <button onClick={clickHandler} >Click Me</button>
    </Card>
  );
}

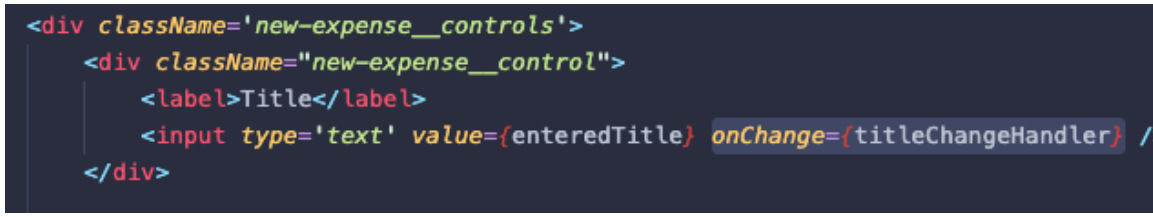
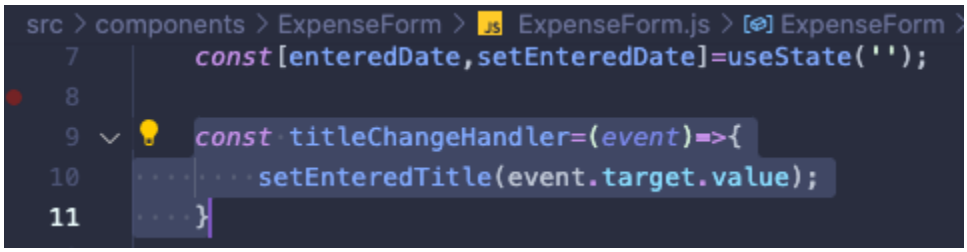
export default ExpenseItem;
```

- In expenseitem.js we are accessing each list elements which was passed by expenses.js



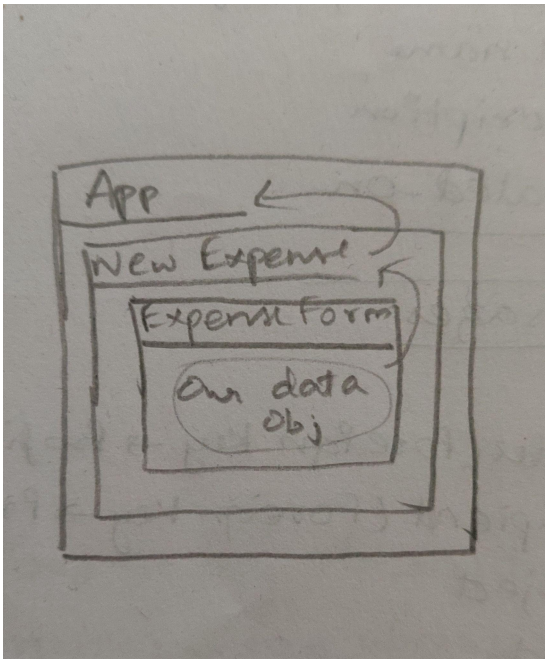
- As we see in this picture we are passing props from top to bottom, i.e from parent to child

- NOW HOW COULD WE DO THIS IN OPPSOITE DIRECTION, i.e. CHILD TO PARENT

    - It is very easy, we know that the source component file for default html components are the browser
    - Let it be the child of all our custom components
        - 
```
<form onSubmit={submitHandler}>
    <div className='new-expense__controls'>
        <div className="new-expense__control">
            <label>Title</label>
            <input type='text' value={enteredTitle} onChange={titleChangeHandler
        </div>
```
        - This input component is a default component which came from another child file(browser)
        - 
```
<div className='new-expense__controls'>
    <div className="new-expense__control">
        <label>Title</label>
        <input type='text' value={enteredTitle} onChange={titleChangeHandler} /
    </div>
```
        - Here we have used a eventhandler prop which takes in an function, in the same file itself we have defined that function
        - 
```
src > components > ExpenseForm > ExpenseForm.js > ExpenseForm
7       const[enteredDate,setEnteredDate]=useState('');
8
9       const titleChangeHandler=(event)=>{
10          setEnteredTitle(event.target.value);
11      }
```
            - Here we have accepted an argument,and used event.target.value. but where does that event argument came from
            - It came from the source file of the default component input
            - So here it is child to parent

SIMILARLY,

- But that is a pattern we can replicate for our own components as well. We can create our own event props, if we wanna call it like, and we can expect functions as values and that would allow us to pass a function from a parent component to a child component and then call that function inside of the child component. And when we then call a function, we can of course pass data to that function as a parameter and that's how we can communicate up from child to parent.

- In our project, we have to pass the new expenses that is added in the form to the app.js where we have our pre-defined expenses list
- This is our structure
    - 

    - **The object could go from expense from to app.js only in this path[expenseform → new Expense → app.js]**
    - ```
      src > App.js > App
      23      {id:'e4',
      24      title:'Fuel Expense',
      25      amount:200,
      26      date:new Date(2022,3,29)},
      27
      28    ]
      29    return (
      30      
      31        <NewExpense></NewExpense>
      32        <Expenses items={expenses}></Expenses>
      33      
      34    );
      35  }
      36
      37  export default App;
      38
      ```
        - This is where we need the new added expenses, which will be coming from NewExpens component
      ```

-
```
src > components > ExpenseForm > NewExpense.js > [e]
1   import React from 'react';
2   import ExpenseForm from './ExpenseForm';
3   import './NewExpense.css';
4
5   const NewExpense=()=>{
6       return (
7           <div className='new-expense'>
8               <ExpenseForm></ExpenseForm>
9           </div>
10      )
11  }
12
13  export default NewExpense;
```
    - New expense component will be getting the newly added expense data from the expense form component
- 

    - Here we have the expenseData which has to passed to the upper level to pass it to the top(app.js)

- Now lets dive into it
- Firstly we need to get the newly added expense data from expenseForm to newexpense component
- In new expense component
    - src > components > ExpenseForm > NewExpense.js > NewExpense

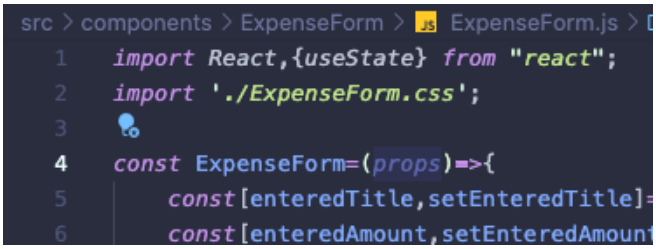
```
1  import React from 'react';
2  import ExpenseForm from './ExpenseForm';
3  import './NewExpense.css';
4
5  const NewExpense=()=>{
6      return (
7          <div className='new-expense'>
8              <ExpenseForm onSaveExpenseData={}></ExpenseForm>
9          </div>
10     )
11 }
12
13 export default NewExpense;
```

        - Here we have creating a new prop, we can name it in anyway
        - It is a convention to use ‘on’ when it accepts an function
        - Here we need it to accept a function because whenever something happens inside expenseForm component this function must be triggered which will accept some data
    - src > components > ExpenseForm > NewExpense.js > NewExpense > onSaveExpenseDataHandler

```
1  import React from 'react';
2  import ExpenseForm from './ExpenseForm';
3  import './NewExpense.css';
4
5  const NewExpense=()=>{
6      const onSaveExpenseDataHandler=(enteredExpenseData)=>{
7
8      }
9
10     return (
11         <div className='new-expense'>
12             <ExpenseForm onSaveExpenseData={onSaveExpenseDataHandler}></ExpenseForm>
13         </div>
14     )
15 }
16
17 export default NewExpense;
```

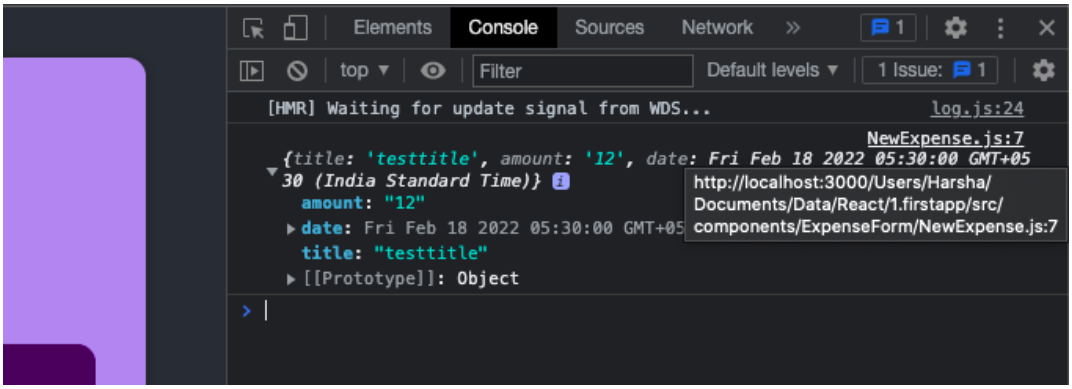
```
const onSaveExpenseDataHandler: (enteredExpenseData: any)
```

        - Here we have defined our function which will be passed inside the props
        - Here we have also specified an argument
            - This is the argument which will get the newly added expense data from the child component, let us see how it happens

- Now in expenseForm.js
  - 

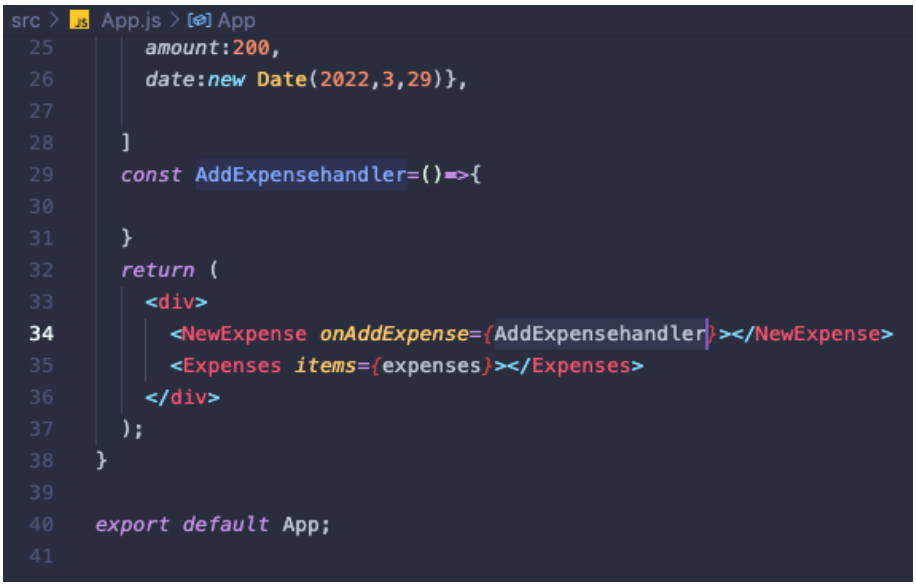
    - Here we have specified the argument props because while calling the expenseForm from the new expense component we have also included one attribute ‘onSaveExpenseData’, so to accept it here, we have specified this component to accept that using an argument
  - 

    - Here when form is submitted, submithandler will be executed in line 30 instead of printing the expensedata here we are calling the onSaveExpenseData function which is defined in parent component, using the props and we have also passed in the expenseData while calling

- Now let us accept this value in that onSaveExpenseHandler and print it there
  -
```
src > components > ExpenseForm > NewExpense.js > NewExpense > onSaveExpenseDataHandler
1   import React from 'react';
2   import ExpenseForm from './ExpenseForm';
3   import './NewExpense.css';
4
5   const NewExpense=()=>{
6       const onSaveExpenseDataHandler=(enteredExpenseData)=>{
7           console.log(enteredExpenseData);
8       }
9
10      return (
11          <div className='new-expense'>
12              <ExpenseForm onSaveExpenseData={onSaveExpenseDataHandler}></ExpenseForm>
13          </div>
14      )
15  }
16
17  export default NewExpense;
```
- Now when we entered a new value
  - 

    - Here we could see that the values entered in expenseForm is now logged from its parent component NewExpense.js

SIMILARLY, NOW WE ARE GOING TO PASS THE DATA FROM NEWEXPENSE TO APP.JS

- In app.js,
  - 
    - Here in app.js while calling the newExpense component we are passing a prop
  - So in newexpense.js let us add an prop argument in the function
    - 
    - Here we are accepting the props
    - In line 6 we have onSaveExpenseDataHandler which will be executed when the form inside ExpenseForm is submitted(refer line 12 here)
    - So when the form is submitted in the child component(ExpenseForm) it will be passed here(NewExpense and now we have to pass it to its parent App.js
    - So inside the same onSaveExpenseData we are going to call the props.onAddExpense()along with that passing in the

enteredExpenseData which is accessed from the ExpenseForm in Newexpense

```
src > components > ExpenseForm > NewExpense.js > NewExpense > onSaveExpenseDataHandler
import React from 'react';
import ExpenseForm from './ExpenseForm';
import './NewExpense.css';

const NewExpense=(props)=>{
    const onSaveExpenseDataHandler=(enteredExpenseData)=>{
        //console.log(enteredExpenseData);
        props.onAddExpense(enteredExpenseData);
    }

    return (
        <div className='new-expense'>
            <ExpenseForm onSaveExpenseData={onSaveExpenseDataHandler}></ExpenseForm>
        </div>
    )
}

export default NewExpense;
```

- Now the function that we defined in app.js will be called along with that enteredExpenseData being passed

- So in App.js

```
src > App.js > App > AddExpensehandler
    date:new Date(2022,3,3)},

    {id:'e4',
    title:'Fuel Expense',
    amount:200,
    date:new Date(2022,3,29)},

]
const AddExpensehandler=(newExpenseData)=>{
    console.log(newExpenseData);
}

return (
  <div>
    <NewExpense onAddExpense={AddExpensehandler}></NewExpense>
    <Expenses items={expenses}></Expenses>
  </div>
);
}

export default App;
```

- Here we are accepting an argument and also printing that

- Now we can clearly see that the entered from data is logged from app.js
  -