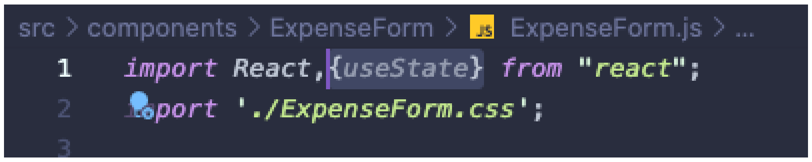
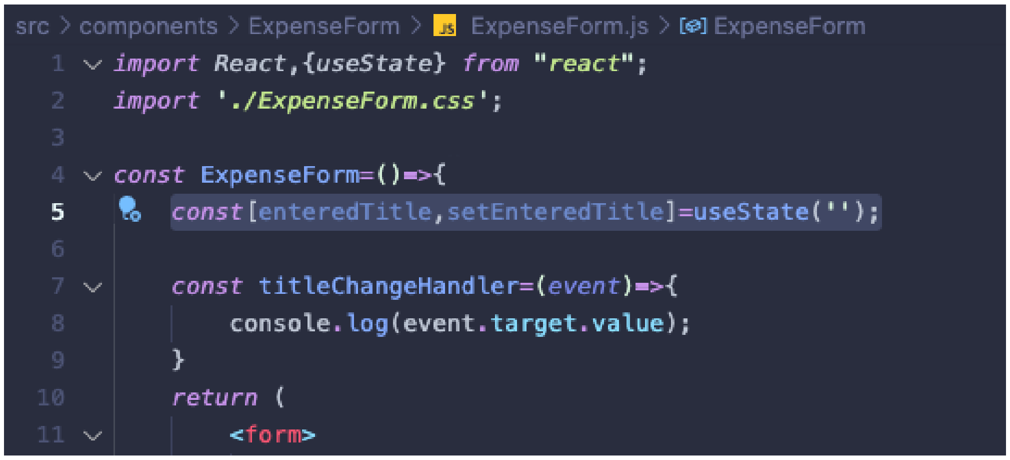
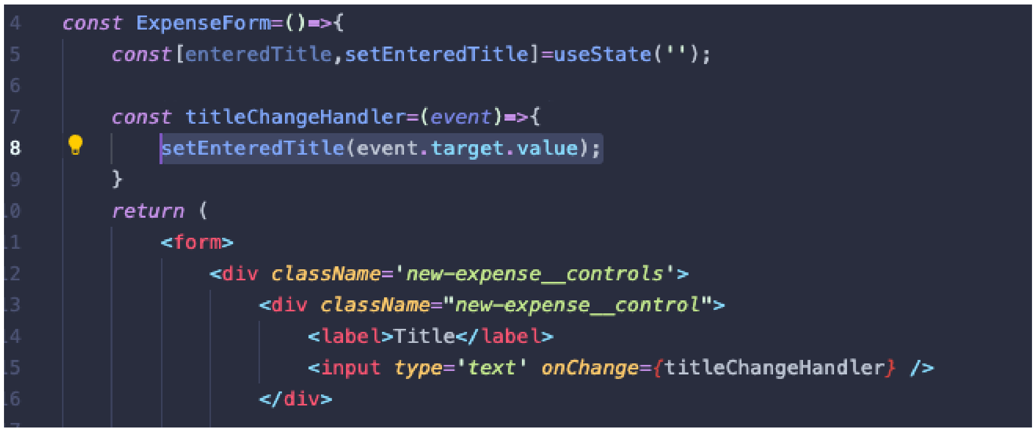
19.working with multiple states(separate maintenance approach)

- Until now we have done this
    - 1. Getting the entered value after each keystroke
- But now we need to store it somewhere, so that after submitting the form we could have that value for performing some operations
- And the other thing is we need to hold that value if suppose that component is re-evaluated or re-rendered for someother reasons(like switching page from pagination)
- For this we could use state variable which will satisfy both the two needs
    - 

      src > components > ExpenseForm > ExpenseForm.js > ...

      ```
      import React,{useState} from "react";
      import './ExpenseForm.css';
      ```
    - 

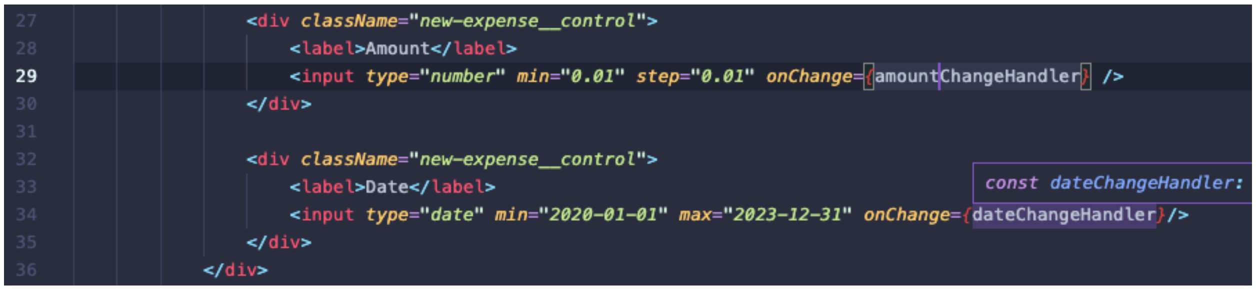
      src > components > ExpenseForm > ExpenseForm.js > ExpenseForm

      ```
      import React,{useState} from "react";
      import './ExpenseForm.css';

      const ExpenseForm=()=>{
          const[enteredTitle,setEnteredTitle]=useState('');

          const titleChangeHandler=(event)=>{
              console.log(event.target.value);
          }
          return (
              <form>
      ```
    - Here we are creating a state variable enteredTitle to store the input value
    - Initially the value of the state variable is an empty string
    - Now,
        - 

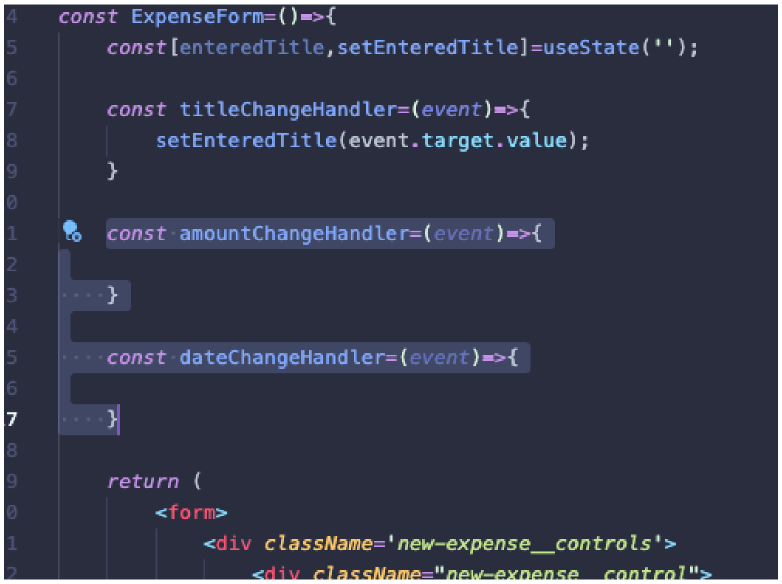
          ```
          const ExpenseForm=()=>{
              const[enteredTitle,setEnteredTitle]=useState('');

              const titleChangeHandler=(event)=>{
                  setEnteredTitle(event.target.value);
              }
              return (
                  <form>
                      <div className='new-expense__controls'>
                          <div className="new-expense__control">
                              <label>Title</label>
                              <input type='text' onChange={titleChangeHandler} />
                          </div>
          ```
        - Now the state variable enteredTitle will have the current input value because we are using the setEnteredTitle function inside the event handler function of onChange and passing in the value that is entered

- Now, here I'm not really doing it to update this component though the component will update. That will always happen when you update the State but I'm doing it to ensure that we're storing this in some variable, which is kind of detached from the life cycle of this component function. So that no matter how often this component function might execute again, this State is stored and survives.

- Now let us do the same for other input fields also

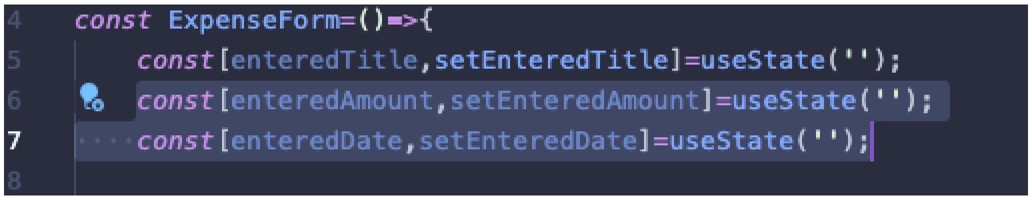


```
            <div className="new-expense__control">
                <label>Amount</label>
                <input type="number" min="0.01" step="0.01" onChange={amountChangeHandler} />
            </div>

            <div className="new-expense__control">
                <label>Date</label>
                <input type="date" min="2020-01-01" max="2023-12-31" onChange={dateChangeHandler}/>
            </div>
        </div>
```

- 

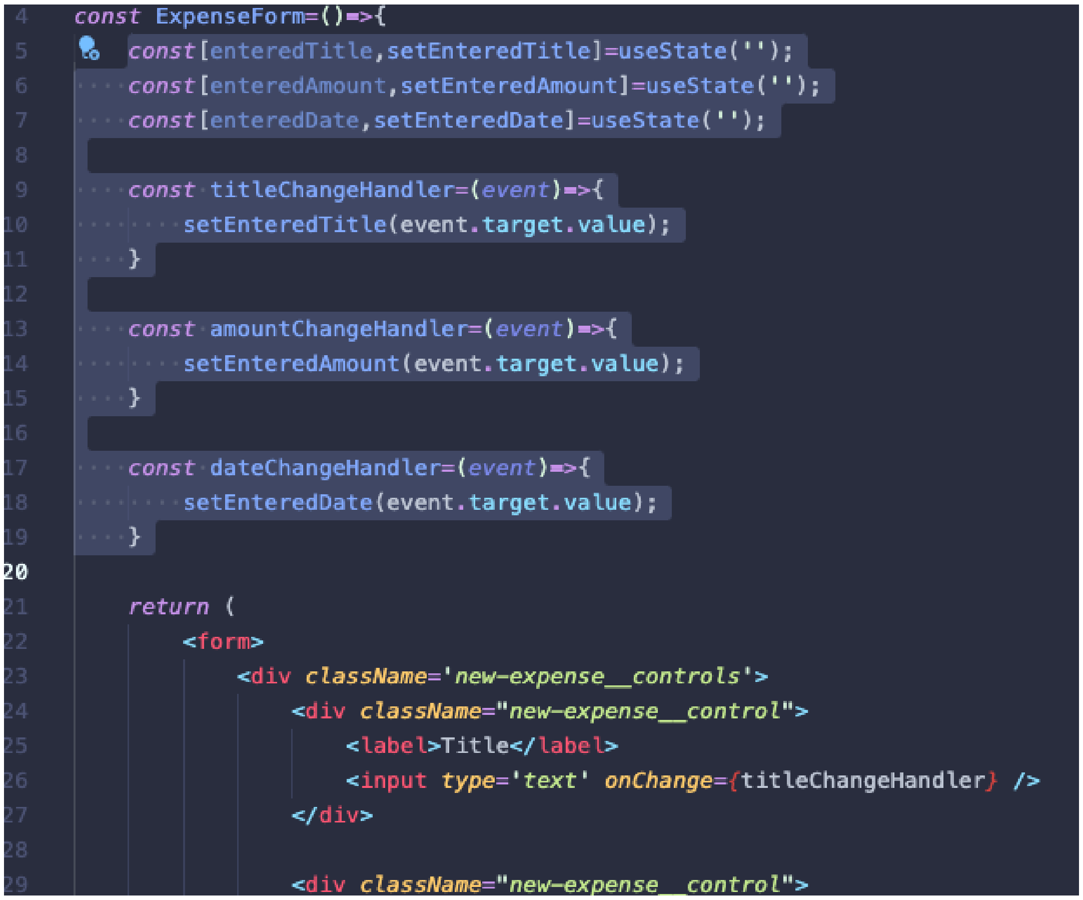
```
const ExpenseForm=()=>{
    const[enteredTitle,setEnteredTitle]=useState('');

    const titleChangeHandler=(event)=>{
        setEnteredTitle(event.target.value);
    }

    const amountChangeHandler=(event)=>{

    }

    const dateChangeHandler=(event)=>{

    }

    return (
        <form>
            <div className='new-expense__controls'>
                <div className="new-expense__control">
```

- Now its time to create state variables for other input fields as well



```
const ExpenseForm=()=>{
    const[enteredTitle,setEnteredTitle]=useState('');
    const[enteredAmount,setEnteredAmount]=useState('');
    const[enteredDate,setEnteredDate]=useState('');
```

-
    - Here we have the initial value of state variable as a string because by default, the events.target.avlue which is given by onChange event is of string type

- Now finally,
  - 

```
const ExpenseForm=()=>{
    const[enteredTitle,setEnteredTitle]=useState('');
    const[enteredAmount,setEnteredAmount]=useState('');
    const[enteredDate,setEnteredDate]=useState('');

    const titleChangeHandler=(event)=>{
        setEnteredTitle(event.target.value);
    }

    const amountChangeHandler=(event)=>{
        setEnteredAmount(event.target.value);
    }

    const dateChangeHandler=(event)=>{
        setEnteredDate(event.target.value);
    }

    return (
        <form>
            <div className='new-expense__controls'>
                <div className="new-expense__control">
                    <label>Title</label>
                    <input type='text' onChange={titleChangeHandler} />
                </div>

                <div className="new-expense__control">
```

- Here we have create three separate state variable which will act independent, any changes made to onestate will not affect the other states