

W16-P1: fetch cart data from url (API)

The screenshot displays a web application titled "UseReducer" running on localhost:3000. The main content area, titled "YOUR BAG", lists four items in the cart:

- Samsung Galaxy S8: \$399.99, remove button
- Google Pixel: \$499.99, remove button
- Xiaomi Redmi Note 2: \$699.99, remove button
- Samsung Galaxy S7: \$599.99, remove button

The total price at the bottom is \$2199.96. The background shows the Redux state in the browser's DevTools console:

```
cart: {
  items: [
    { id: 'rec86qchPxb62VJ75', title: 'google pixel', price: '499.99', amount: 1 },
    { id: 'recd8dsEidw2VU0', title: 'Xiaomi Redmi Note 2', price: '699.99', amount: 1 },
    { id: 'recuTol60XST3PIw', title: 'Samsung Galaxy S7', price: '599.99', amount: 1 },
    { id: 'rec1Zl7fCIBOPdcT2', title: 'Samsung Galaxy S8', price: '399.99', amount: 1 }
  ],
  loading: false
}
```

The code editor on the left shows the Redux state management logic:

```
const fetchData = async () => {
  dispatch({ type: 'LOADING' });
  const response = await fetch(url);
  const cart = await response.json();
  console.log('cart', cart);
  dispatch({ type: 'DISPLAY_ITEMS', payload: cart });
}

useEffect(() => {
  fetchData();
}, []);

return <AppContext_27.Provider value={{...state, clearCart, fetchData}} > {
  {children}
}</AppContext_27.Provider>
```

The Reducer logic for handling actions is also visible:

```
if(action.type === 'GET_TOTALS'){
  // ...
}
if(action.type === 'LOADING'){
  return { ...state, loading: true }
}
if(action.type === 'DISPLAY_ITEMS'){
  return { ...state, cart: action.payload, loading: false }
}
```

W16-P2: remove cart item

The screenshot shows the code editor with three files open: `Reducer_27.js`, `AppProvider_27.js`, and `CartItem_27.js`.

Reducer_27.js (lines 48-55) shows the `'REMOVE'` action handler:

```
if(action.type === 'REMOVE'){
  const filteredCart = state.cart.filter(
    (item) => item.id !== action.payload
  );
  return { ...state, cart: filteredCart };
}
```

AppProvider_27.js (lines 52-67) shows the `remove` function being passed to the context provider:

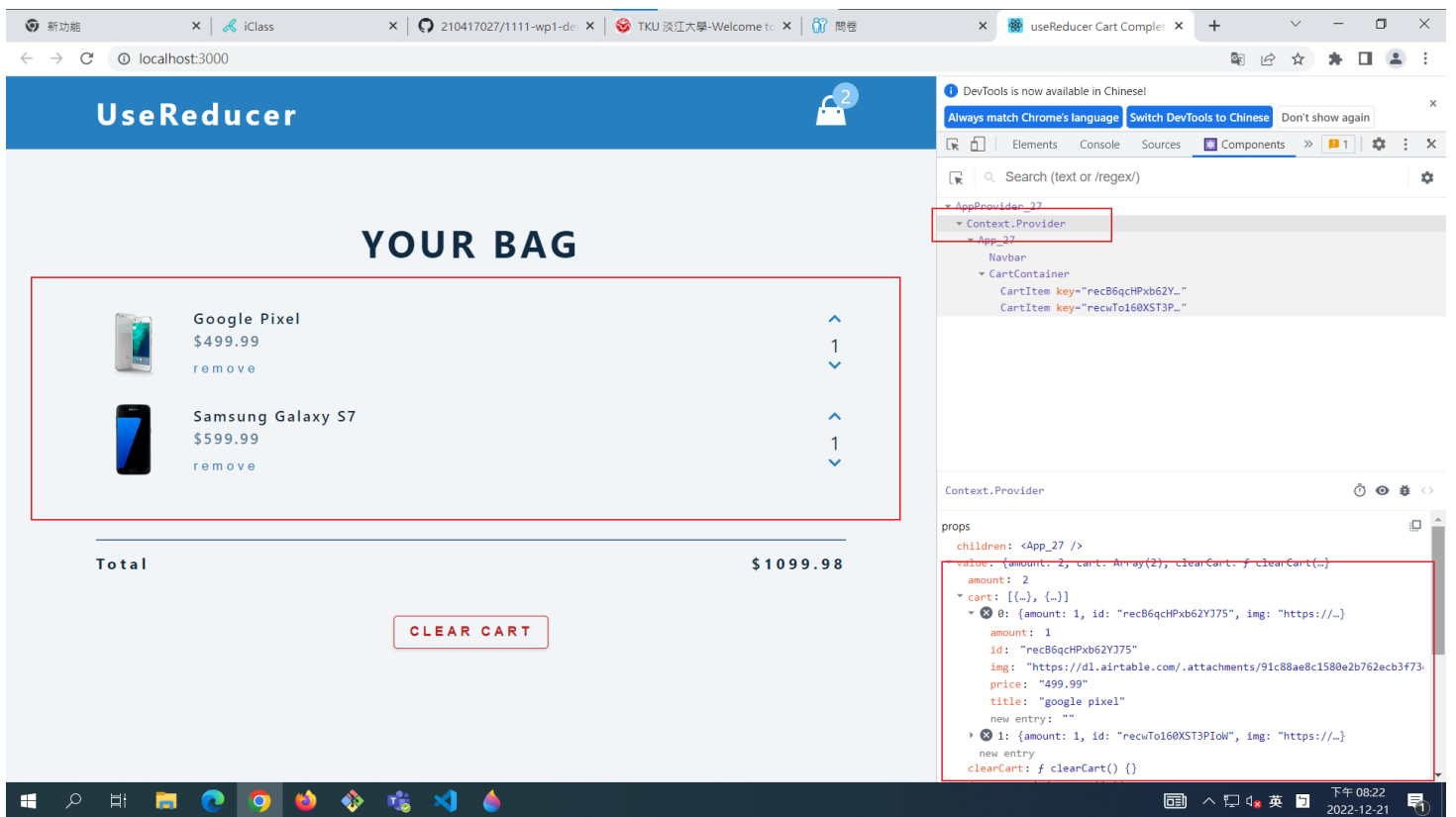
```
const useGlobalContext_xx = () => {
  return useContext(AppContext_27);
}

export { AppProvider_27, useGlobalContext_xx };
```

CartItem_27.js (lines 5-27) shows the `remove` function and the corresponding UI button:

```
const { increase, decrease, remove } =
  useGlobalContext_xx();

return (
  <article className='cart-item'>
    <img src={img} alt={title} />
    <div>
      <h4>{title}</h4>
      <h4 className='item-price'>${price}</h4>
      <div>
        <button className='remove-btn' onClick={
          () => remove(id)}>
          remove
        </button>
      </div>
    </div>
  </article>
);
```



W16-P3: decrease amount of a cart item

