

# Java Script : over all picture of the code :



```

EXPLORER
MOON-LIGHT
node_modules
public
src
component
customers
AddCustomers.js U
CustomerDetails.js U
Customers.js U
DeleteCustomer.js U
EditCustomer.js U
Footer.js U
Home.js U
Navbar.js U
OurServices.js U
PageNotFound.js U
App.css
App.js M
App.test.js
index.css
index.js M
logo.svg
reportWebVitals.js
setupTests.js
.gitignore
package-lock.json M
package.json M
README.md

App.test.js
ContactUs.js U
Footer.js U
AddCustomers.js U
OurServices.js U
CustomerDetails.js U
Customers.js U X
DeleteCustomer.js U
EditCustomer.js U

src > component > customers > Customers.js ...
7 function Customers() {
29   function showCustomers() {
38     (customers.map((cust) => <tr key={cust.id}>
48       </Link>
49       <Link to={"/customers/edit/${cust.id}"} className="btn btn-primary" title="Edit">
50         <i className="bi bi-pencil-square" ></i>
51       </Link>
52       <Link to={"/customers/delete/${cust.id}"} className="btn btn-danger" title="delete" >
53         <i className="bi bi-trash" ></i>
54       </Link>
55     </div>
56     </td>
57   </tr>
58 </tbody>
59 </table>
60 </div>
61 </tbody>
62 </div>
63 </div>
64 </div>
65 </div>
66 </div>
67 </div>
68 </div>
69 </div>
70 </div>
71 </div>
72 </div>
73 </div>
74 </div>
75 </div>
76 </div>
77 <div className="alert alert-primary text-center m-3">Customers</div>
78 <Link to={"/customer/add"} className="btn btn-success"> Add New Customer
79 </Link>
80 <div className="input-group m-3" style={{ width: "50%"}}>
81 <input type="text" className="form-control" onChange={handleSearchTextChange} />
82 <button className="btn btn-secondary" onClick={search}><i className="bi bi-search" ></i></button>
83 </div>
84 {showCustomers()}
85 </div>
86 </div>
87 </div>
88 </div>

```

detailed view of the logic :

```

function Customers() {
  let [customers, setCustomers] = useState([]); // we have intialized the customers witha an empty array
  async function getAllCustomers() {
    let allCustomers = await axios.get("http://localhost:5000/customers"); // to get the data from the API using the a
    //note that await makes the compiler wait for the data to be brought we did so as we are using the asynchronous prog
    // note that we have used the asynchronous programming to avoid many problems such as that the data may be too big a
    setCustomers(allCustomers.data); //note that we wrote .data as the allCustomers has other things rather tahnbn the d
    //we used the setCustomers to put the data in the customers var
  }
  useEffect(() => { getAllCustomers() }, []); // we used the use effect to call the function getAllCustomers that sets th
  let [searchText, setSearchText] = useState("");
  function handelSeachTextChange(event) {
    setSearchText(event.target.value);
  }
  async function search() {
    let searchResult = await axios.get(`http://localhost:5000/customers?q=${searchText}`);
    setCustomers(searchResult.data)
  }
  function showCustomers() {
    if (customers.length > 0) {
      return <div>
        <table className=" table table-bordered table-striped table-hover text-center caption-top " >
          <caption className="text-center fs-1"></caption>
          <thead><tr><th>Id</th> <th>Name</th> <th>Balance </th> <th>Image</th> <th>Action</th></tr></thead>
          <tbody>
            {customers.map((cust) => <tr key={cust.id}>
              <td>{cust.id}</td>
              <td>{cust.name}</td>
              <td>{cust.balance} </td>
              <td>img src={cust.imageUrl}</td>
              <td>
                <div className="btn-group btn-group-sm">

```

```

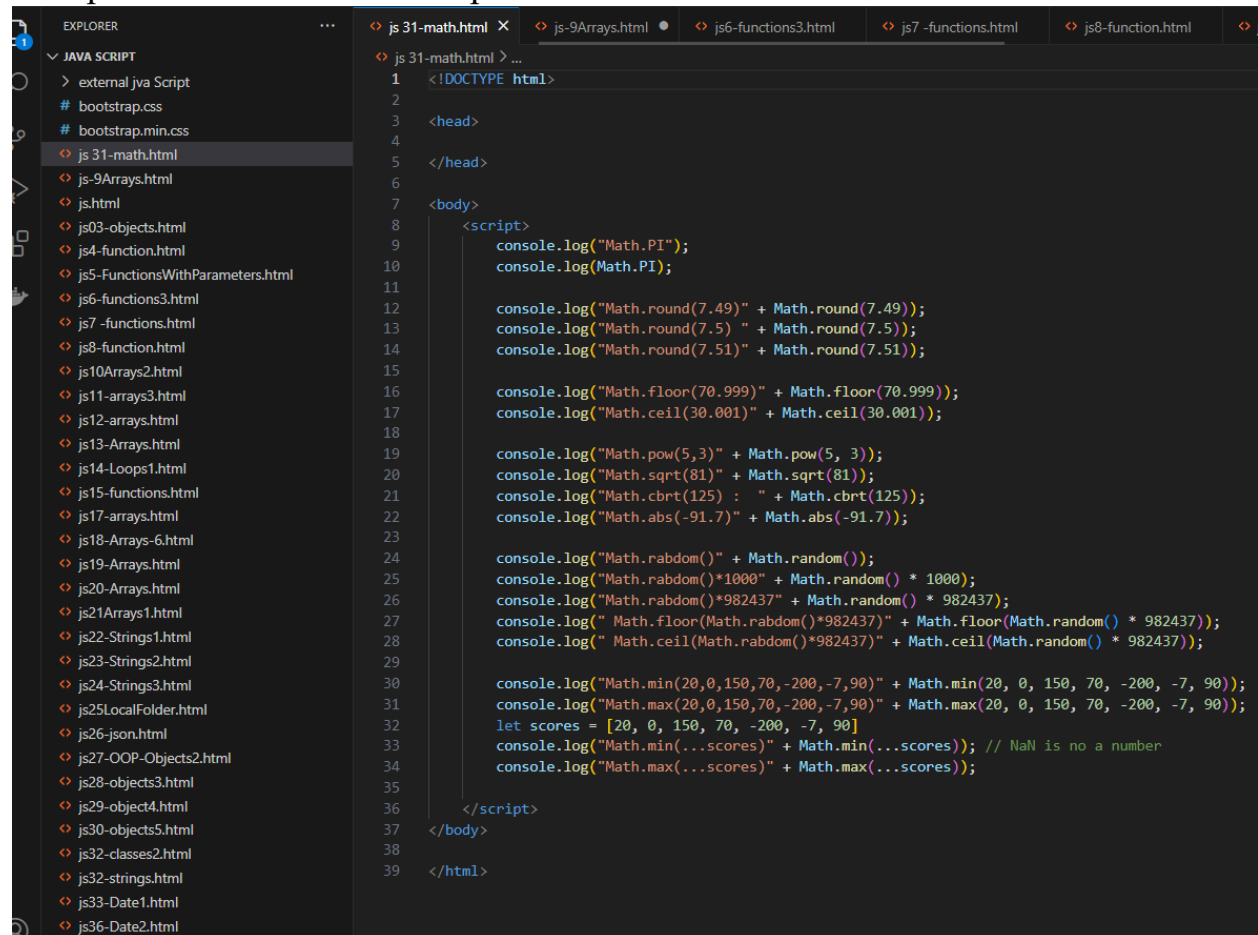
        <div className="btn-group btn-group-sm">
          <Link to={` /customers/details/${cust.id}`} className="btn btn-success" title="details">
            <i className="bi bi-list-ul" ></i>
          </Link>
          <Link to={` /customers/edit/${cust.id}`} className="btn btn-primary" title="Edit">
            <i className="bi bi-pencil-square" ></i>
          </Link>
          <Link to={` /customers/delete/${cust.id}`} className="btn btn-danger" title="delete" >
            < i className=" bi bi-trash"></i>
          </Link>
        </div>
        /* note that here the Link is exactly the same as the NavLink used in the NavBar but it is called
        t2reeban kda el Link henna bdal el anchor */
        /* note that the title is the thing that when we hover over the link appears the text written in th
      </td>
    </tr>}}
  </tbody>
</table>
</div>
);
}
else {
  return <h1 className="alert alert-danger text-center m-3" >No Customers To Display. </h1>;
}
}

return <div>
  <h1 className="alert alert-primary text-center m-3 ">Customers</h1>
  <Link to={` /customer/add`} className="btn btn-success"> Add New Customer /* note that the first forward slash me
  */>
  </Link>
  <div className="input-group m-3" style={{ width: "50%" }}>
    < input type="text" className="form-control" onChange={handelSeachTextChange} />
    <button className="btn btn-secondary" onClick={search}><i className="bi bi-search"></i></button>
  </div>
  <h1 className="alert alert-primary text-center m-3 ">Customers</h1>
  <Link to={` /customer/add`} className="btn btn-success"> Add New Customer /* note that the f
  */>
  </Link>
  <div className="input-group m-3" style={{ width: "50%" }}>
    < input type="text" className="form-control" onChange={handelSeachTextChange} />
    <button className="btn btn-secondary" onClick={search}><i className="bi bi-search"></i></button>
  </div>
  {showCustomers()}
</div>;
}
}
export default Customers;

```

note that the Java Script code is written between the script tags

`<script > // JS code ..... </ script>`



```
EXPLORER
  JAVA SCRIPT
    external java Script
    # bootstrap.css
    # bootstrap.min.css
    js 31-math.html
    js-9Arrays.html
    js.html
    js03-objects.html
    js4-function.html
    js5-FunctionsWithParameters.html
    js6-functions3.html
    js7 -functions.html
    js8-function.html
    js10Arrays2.html
    js11-arrays3.html
    js12-arrays.html
    js13-Arrays.html
    js14-Loops1.html
    js15-functions.html
    js17-arrays.html
    js18-Arrays-6.html
    js19-Arrays.html
    js20-Arrays.html
    js21Arrays1.html
    js22-Strings1.html
    js23-Strings2.html
    js24-Strings3.html
    js25LocalFolder.html
    js26-json.html
    js27-OOP-Objects2.html
    js28-objects3.html
    js29-object4.html
    js30-objects5.html
    js32-classes2.html
    js32-strings.html
    js33-Date1.html
    js36-Date2.html

js 31-math.html > ...
1  <!DOCTYPE html>
2
3  <head>
4
5  </head>
6
7  <body>
8
9      <script>
10         console.log("Math.PI");
11         console.log(Math.PI);
12
13         console.log("Math.round(7.49)" + Math.round(7.49));
14         console.log("Math.round(7.5)" + Math.round(7.5));
15         console.log("Math.round(7.51)" + Math.round(7.51));
16
17         console.log("Math.floor(70.999)" + Math.floor(70.999));
18         console.log("Math.ceil(30.001)" + Math.ceil(30.001));
19
20         console.log("Math.pow(5,3)" + Math.pow(5, 3));
21         console.log("Math.sqrt(81)" + Math.sqrt(81));
22         console.log("Math.cbrt(125) : " + Math.cbrt(125));
23         console.log("Math.abs(-91.7)" + Math.abs(-91.7));
24
25         console.log("Math.rabdom()" + Math.random());
26         console.log("Math.rabdom()*1000" + Math.random() * 1000);
27         console.log("Math.rabdom()*982437" + Math.random() * 982437);
28         console.log(" Math.floor(Math.rabdom()*982437)" + Math.floor(Math.random() * 982437));
29         console.log(" Math.ceil(Math.rabdom()*982437)" + Math.ceil(Math.random() * 982437));
30
31         console.log("Math.min(20,0,150,70,-200,-7,90)" + Math.min(20, 0, 150, 70, -200, -7, 90));
32         console.log("Math.max(20,0,150,70,-200,-7,90)" + Math.max(20, 0, 150, 70, -200, -7, 90));
33         let scores = [20, 0, 150, 70, -200, -7, 90]
34         console.log("Math.min(...scores)" + Math.min(...scores)); // NaN is no a number
35         console.log("Math.max(...scores)" + Math.max(...scores));
36     </script>
37 </body>
38
39 </html>
```

```
EXPLORER
  JAVA SCRIPT
    external java Script
    # bootstrap.css
    # bootstrap.min.css
    js 31-math.html
    js-9Arrays.html
    js.html
    js03-objects.html
    js4-function.html
    js5-FunctionsWithParameters.html
    js6-functions3.html
    js7 -functions.html
    js8-function.html
    js10Arrays2.html
    js11-arrays3.html
    js12-arrays.html
    js13-Arrays.html
    js14-Loops1.html
    js15-functions.html
    js17-arrays.html
    js18-Arrays-6.html
    js19-Arrays.html
    js20-Arrays.html
    js21Arrays1.html
    js22-Strings1.html
    js23-Strings2.html
    js24-Strings3.html
    js25LocalFolder.html
    js26-json.html
    js27-OOP-Objects2.html
    js28-objects3.html
    js29-object4.html
    js30-objects5.html
    js32-classes2.html
    js32-strings.html
    js33-Date1.html
    js36-Date2.html
    js36-typeOf.html

js19-Arrays.html > body > script
3 <head>
5 </head>
6
7 <body>
8 <script>
9     let employees = [
10         { name: "Osama Ahmed", position: "Developer", salary: 7500 },
11         { name: "Ahmed Ali", position: "Tester", salary: 6000000 },
12         { name: "Hossam Mahmoud", position: "Designer ", salary: 8000 },
13         { name: "Wael Mostafa", position: "Team Leader", salary: 18500 },
14         { name: "Bahaa hassan", position: "Developer", salary: 10500 },
15         { name: "Hussian", position: "Developer", salary: 10500 }];
16
17     // arr.every(predicate function)
18     // returns true if ALL the elemets meat the condition
19     console.log("every()");
20     let areAllEmployeesDevelopers = employees.every(function (emp) { return emp.position === "Developer" });
21     console.log(areAllEmployeesDevelopers);
22     let isEachEmployeesSalaryGreaterThanOrEqualTo5000 = employees.every(function (emp) { return emp.salary >= 5000 });
23     console.log("isEachEmployeesSalaryGreaterThanOrEqualTo5000");
24     console.log(isEachEmployeesSalaryGreaterThanOrEqualTo5000);
25     // arr.some(predicate function)
26     // returns true if SOME ( even if one ) the element meat the condition
27     console.log("Some ()");
28     let areSomeEmployeesDevelopers = employees.some(function (emp) { return emp.position === "Developer" });
29     console.log(areSomeEmployeesDevelopers);
30     console.log(areSomeEmployeesDevelopers);
31     let areSomeEmployeesSalaryGreaterThanOrEqualTo2500 = employees.some(function (emp) { return emp.salary >= 2500 });
32     console.log("areSomeEmployeesSalaryGreaterThanOrEqualTo2500");
33     console.log(areSomeEmployeesSalaryGreaterThanOrEqualTo2500);
34     // map --> we give the map a fun ction that is to be executed on all the elements of the array
35
36     console.log(" map")
37     let costs = [1000, 1500, 3000, 2000, 25000];
38     console.log("costs");
39     console.log(costs);
40
41     let prices = costs.map(function (cost) { return cost * 1.5 });
42     console.log("prices");
43     console.log(prices);
44 </script>
45 </body>
46
```

```
File Edit Selection View Go Run Terminal Help Java Script
EXPLORER
  JAVA SCRIPT
    external java Script
    # bootstrap.css
    # bootstrap.min.css
    js 31-math.html
    js-9Arrays.html
    js.html
    js03-objects.html
    js4-function.html
    js5-FunctionsWithParameters.html
    js6-functions3.html
    js7 -functions.html
    js8-function.html
    js10Arrays2.html
    js11-arrays3.html
    js12-arrays.html
    js13-Arrays.html
    js14-Loops1.html
    js15-functions.html
    js17-arrays.html
    js18-Arrays-6.html
    js19-Arrays.html
    js20-Arrays.html
    js21Arrays1.html
    js22-Strings1.html
    js23-Strings2.html
    js24-Strings3.html
    js25LocalFolder.html
    js26-json.html
    js27-OOP-Objects2.html
    js28-objects3.html
    js29-object4.html
    js30-objects5.html
    js32-classes2.html
    js32-strings.html
    js33-Date1.html
    js36-Date2.html
    js36-typeOf.html
  OUTLINE
  TIMELINE
  0 0 0 0 0

js29-object4.html > ...
1 <!DOCTYPE html>
2
3 <head>
4   <title> Some Methods of the Objects </title>
5 </head>
6
7 <body>
8   <script> // some methods of the object
9     let prod3 = { name: "SIS", decription: "Student Information System", prics: 5000 };
10    let customer1 = new Object();
11    customer1.name = "AlShams University ";
12    customer1.email = "email@yahoo.com";
13    console.log(" customer1 let customer1 = new object();");
14    console.log(customer1);
15
16    console.log("object.assign ( targetObject , sourceObject)");// copies an object in an
17    let prod4 = new Object();
18    console.log(" object 4 before bieng assigned to ");
19    console.log(prod4);
20    // Object.assign(target , source)
21    Object.assign(prod4, prod3);
22    console.log("prod4 After bieng assigned to : ");
23    console.log(prod4);
24
25    let prod5 = {}; // empty object
26    console.log("prod5");
27    console.log(prod5);
28
29    console.log("object entries (prod 3 )");// breaks the elements of the objects in an
30    let prod3Entries = Object.entries(prod3);
31    console.log("prod3Entries");
32    console.log(prod3Entries);
33    console.log(" for let entery of prod3Entries \n");
34    for (let entery of prod3Entries) {
35      console.log(entery);
36    }
37    console.log(" for ( let [property , value ] of prod3Entrise )");
38    for (let [property , value] of prod3Entries) {
39
40      console.log(property + ":" + value);
41    } </script>
42 </body>
43
44 </html>
```