LAB # 08

Statement Purpose:

To familiarize the students with

- HTML CRUD operations using JavaScript
- ToDo List
- ReactJs Introduction
- Installation, Basic example

```
<!DOCTYPE html>
<html lang="en">

<head>
   <meta charset="UTF-8">
   <meta http-equiv="X-UA-Compatible" content="IE=edge">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Document</title>
   <link rel="Stylesheet" href="crud.css">
</head>

<body>
   <div class="flex-container">
      <div class="flex-child magenta">
         <table class="table">
            <tr class="table-row">
               <td>
                  <form onsubmit="event.preventDefault(); onFormSubmit();">
                     <div class="container">
                        <h1 id="heading1">Register</h1>
                        <p id="p1">Please fill this form to create a Record.</p>
                        <hr>

                        <label id="label" for="Emp-Name"><b>Emp-Name</b></label>
                        <input type="text" placeholder="Enter Name" name="name"
id="name" required>

                        <label id="label" for="City"><b>City</b></label>
                        <input type="text" placeholder="Enter City" name="city" id="city"
required>

                        <label id="label" for="salary"><b>Salary</b></label>
                        <input type="number" name="salary" id="salary"
placeholder="Salary" required>
```

```
                        <hr>

                        <!-- <p>By creating an account you agree to our <a
href="#">Terms & Privacy</a>.</p> -->
                        <button type="submit" name="submit"
class="registerbtn">Submit</button>
                    </div>
                </form>
            </td>

        </tr>

    </table>
</div>
<div class="flex-child green">
    <td>
        <table id="empList">
            <h1 id="record">Record</h1>
            <thead id="tab_head">
                <tr>
                    <th>Emp-Name</th>
                    <th>City</th>
                    <th>Salary</th>
                    <th>Edit</th>
                    <th>Delete</th>
                </tr>
            </thead>

            <tbody id="tab_body">

            </tbody>

        </table>
    </td>
</div>
</div>
<script src="./crud.js"></script>
</body>

</html>
```

CRUD.js

```
let selectRow = null;

function onFormSubmit() {
    let formData = readFormData();
    if (selectRow == null) insertNewRecord(formData);
    else updateRecord(formData);
```

```
    resetForm();
}
// Getting value from User-----------------------------------------------
function readFormData() {
    var formData = {};
    formData["name"] = document.getElementById("name").value;
    formData["city"] = document.getElementById("city").value;
    formData["salary"] = document.getElementById("salary").value;
    //   console.log(formData);
    return formData;
}

// Inserting & Showing Record in Another Table-----------------------------------------------
function insertNewRecord(data) {
    let table = document
        .getElementById("empList")
        .getElementsByTagName("tbody")[0];
    let newRow = table.insertRow(table.length);
    cell1 = newRow.insertCell(0);
    cell1.innerHTML = data.name;
    cell2 = newRow.insertCell(1);
    cell2.innerHTML = data.city;
    cell3 = newRow.insertCell(2);
    cell3.innerHTML = data.salary;
    cell4 = newRow.insertCell(3);
    cell4.innerHTML = `<a onclick="onEdit(this)">Edit</a>`;
    cell5 = newRow.insertCell(4);
    cell5.innerHTML = `<a onclick="onDelete(this)">Delete</a>`;
}

// Reseting Form-----------------------------------------------------------------------
function resetForm() {
    document.getElementById("name").value = "";
    document.getElementById("city").value = "";
    document.getElementById("salary").value = "";
    selectRow = null;
}
// Editing Record ----------------------------------------------------------------------

function onEdit(td) {
    selectRow = td.parentElement.parentElement;
    document.getElementById("name").value = selectRow.cells[0].innerHTML;
    document.getElementById("city").value = selectRow.cells[1].innerHTML;
    document.getElementById("salary").value = selectRow.cells[2].innerHTML;
}

// Update Record-----------------------------------------------------------------------
function updateRecord(formData) {
    selectRow.cells[0].innerHTML = formData.name;
    selectRow.cells[1].innerHTML = formData.city;
```

```
    selectRow.cells[2].innerHTML = formData.salary;
}

// Dleteing Record------------------------------------------------------------------------
function onDelete(td) {
    if (confirm("Are you want to delete this record ?")) {
        row = td.parentElement.parentElement;
        document.getElementById("empList").deleteRow(row.rowIndex);
        resetForm();
    }
}
```

CRUD CSS

```
* {
    box-sizing: border-box;
}

.flex-container {
    display: flex;
}

.flex-child {
    flex: 1;
    text-align: -webkit-center;
    border: 2px solid rgb(255, 162, 162);
}

.flex-child:first-child {
    margin-right: 20px;
}

.container {
    padding: 16px;
}

input[type="text"],
input[type="number"] {
    width: 100%;
    padding: 15px;
    margin: 5px 0 22px 0;
    display: inline-block;
    border: none;
    background: #f1f1f1;
}

input[type="number"]:focus {
    background-color: #ddd;
```

```
    outline: none;
}

#heading1 {
    margin-top: 0px;
    text-align: center;
    font-size: 20px;
}

#p1 {
    margin-top: 0px;
    text-align: center;
    font-size: 20px;
}

#label,
#empList,
th,
td {
    border: 2px solid goldenrod;
    border-collapse: collapse;
}

th,
td {
    padding: 10px;
}

th {
    background-color: burlywood;
}

#empList tr:nth-child(even) {
    background-color: brown;
}

#empList tr:nth-child(odd) {
    background-color: brown;
}

#empList th {
    background-color: gray;
    color: white;
}

#record {
    display: flex;
    justify-content: center;
}
```

To Do List

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>

<body>
    <div class="container">
        <div class="row">
            <div class="intro col-12">
                <h1>Progress</h1>
                <div>
                    <div class="border">

                    </div>

                </div>
            </div>

            <div class="row">
                <div class="col-12">
                    <input type="text" id="userinput" placeholder="New Item"
maxlength="30">
                    <button id="enter">ENter</button>
                </div>
            </div>

            <div class="row">
                <div>
                    <ul class="col-12">

                    </ul>
                </div>

            </div>

        </div>
    </div>
    <script src="list.js"></script>
</body>

</html>
```

```
var enterButton = document.getElementById("enter");
var input = document.getElementById("userinput");
var ul = document.querySelector("ul");
var item = document.getElementsByTagName("li");

function inputLength() {
    return input.value.length;

}

function listLength() {
    return item.length;
}

function createListElement() {
    var li = document.createElement("li");
    li.appendChild(document.createTextNode(input.value));
    ul.appendChild(li);
    input.value = "";
}

function crossOut() {
    li.createList.toggle("done");
    li.addEventListener("click", crossOut);
    var deleteButton = document.createElement("button");
    deleteButton.appendChild(document.createTextNode("X"));
    li.appendChild(deleteButton);
    deleteButton.addEventListner("click", deleteListitem);
}

function deleteListitem() {
    li.classList.add("delete");

}

function addListAfterClick() {
    if (inputLength() > 0) {
        createListElement();
    }
}

function addListAfterKeypress() {
    if (inputLength > 0) {
        createListElement();
    }
}
```

```
enterButton.addEventListener("click", addListAfterClick);
input.addEventListener("keypress", addListAfterKeypress);
```

REACT Basics

React.JS

Extensions

- Community Material Theme
- es7 REact
- es6
- Auto Import
- Import Cost
- EslInt
- JSHInt
- Jslint
- bracket Pair Colorize
- VS Code Great Icon
- Code Spell Checker
- LIve Server
- Code Runner
- Html Snippets

Installation in VS Code

- npx create-react-app my-app
- cd my-app
- npm start

HTML

```
<div id="root">
   <!-- This element's contents will be replaced with your component. -->
</div>
```

import React from 'react';

import ReactDOM from 'react-dom';

```
ReactDOM.render(
  <h1>Hello, world!</h1>,
  document.getElementById('root')
);
```

Introducing JSX

Consider this variable declaration:

const element = <h1>Hello, world!</h1>;

This tag syntax is neither a string nor HTML.

It is called JSX, and it is a syntax extension to JavaScript. We recommend using it with React to describe what the UI should look like. JSX may remind you of a template language, but it comes with the full power of JavaScript

Why JSX?

React embraces the fact that rendering logic is inherently coupled with other UI logic: how events are handled, how the state changes over time, and how the data is prepared for display.

Instead of artificially separating technologies by putting markup and logic in separate files, React separates concerns with loosely coupled units called “components” that contain both. We will come back to components in a further section, but if you’re not yet comfortable putting markup in JS, this talk might convince you otherwise.

React doesn’t require using JSX, but most people find it helpful as a visual aid when working with UI inside the JavaScript code. It also allows React to show more useful error and warning messages.

we declare a variable called name and then use it inside JSX by wrapping it in curly braces:

```
const name = 'Junaid ;
const element = <h1>Hello, {name}</h1>;
```

You can put any valid JavaScript expression inside the curly braces in JSX. For example, 2 + 2, **user.firstName, or Name(user)** are all valid JavaScript expressions.

In the example below, we embed the result of calling a JavaScript function, **Name(user),** into an <h1> element.

```
import logo from './logo.svg';
import './App.css';
import React from 'react';
import ReactDOM from 'react-dom';

function Name(user) {
  return user.firstName + ' ' + user.lastName;
}

const user = {
  firstName: 'Muhammad',
  lastName: 'Junaid'
};

const element = (
  <h1>
```

```
    Welcome, {Name(user)}!
  </h1>
);

ReactDOM.render(
  element,
  document.getElementById('root')
);
export default Name;
```

Rendering an Element into the DOM

```
<div id="root">
  <!-- This div's content will be managed by React. -->
</div>
ReactDOM.render(
  <h1>Hello, world!</h1>,
  document.getElementById('root')
);
```

Updating the Rendered Element

React elements are immutable. Once you create an element, you can't change its children or attributes. An element is like a single frame in a movie: it represents the UI at a certain point in time.

With our knowledge so far, the only way to update the UI is to create a new element, and pass it to root.render().

```
import React from 'react';
import ReactDOM from 'react-dom';

function tick() {
  const element = (
    <div>
      <h1>Hello, world!</h1>
      <h2>It is {new Date().toLocaleTimeString()}.</h2>
    </div>
  );
  ReactDOM.render(
    element,
    document.getElementById('root')
  );
}

setInterval(tick, 1000);

export default tick;
```

```
import React from "react";
```

```
import ReactDOM from "react-dom";
ReactDOM.render( <
   h1 > Hello, world! < /h1>,
   document.getElementById('root')
);
```

```
function MyForm() {
   return ( < form >
      <label > Enter your name:
      <input type = "text" / >
      </label> < / form >
   )
}
ReactDOM.render( < MyForm / > , document.getElementById('root'));
```

```
const myelement = ( <table >
   <tr >
   <th > Name < /th> < /tr >
<tr >
   <td > John < /td>
< /tr > <tr >
   <td > Elsa < /td> < /tr > </table>
);
ReactDOM.render(myelement, document.getElementById('root'));
```

```
function Football() {
   const shoot = () => {
      alert("Great Shot!");
   }
   return ( <button onClick = { shoot } > Take the shot! < /button>
   );
}
ReactDOM.render( < Football / > , document.getElementById('root'));
```