Practical – 6

AIM 1:

Demonstrate Class, Properties, Methods and Objects.

Source Code:

(HTML)

```
<!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">
   <style>
        body {
            background-color: aquamarine;
        }
    </style>

    <title>P6.1</title>
</head>
<body>
    <script src="Pract.js"></script>
</body>
</html>
```

(JAVASCRIPT)

```
class Bike {
    constructor(type, avgspeed) {
        this.type = type;
        this.avgspeed = avgspeed;
        document.write(`<br> Type :- ${type}<br> Average-Speed:- ${avgspeed} in
bike`);
    }
    displayBike(){
        document.write(`<br> Inside bike class`);
    }
}
class Electric extends Bike {
    constructor(price, model) {
        super(price, model);
```

```
        this.price = price;
        this.model = model;
        document.write(`<br> Price :- ${this.price}<br> Model :- ${this.model} in
Electric`);
    }
    displayEle(){
        document.write(`<br> Inside Electric class`);
    }
}
let r = new Electric(200000,"20EC");
r.displayEle();
r.displayBike();
```

Output:

Conclusion:

In this practical I came to know about the class , object and the constructor of Javasctipt.

AIM 2:

Demonstrate how to work with JSON File.

Source Code:

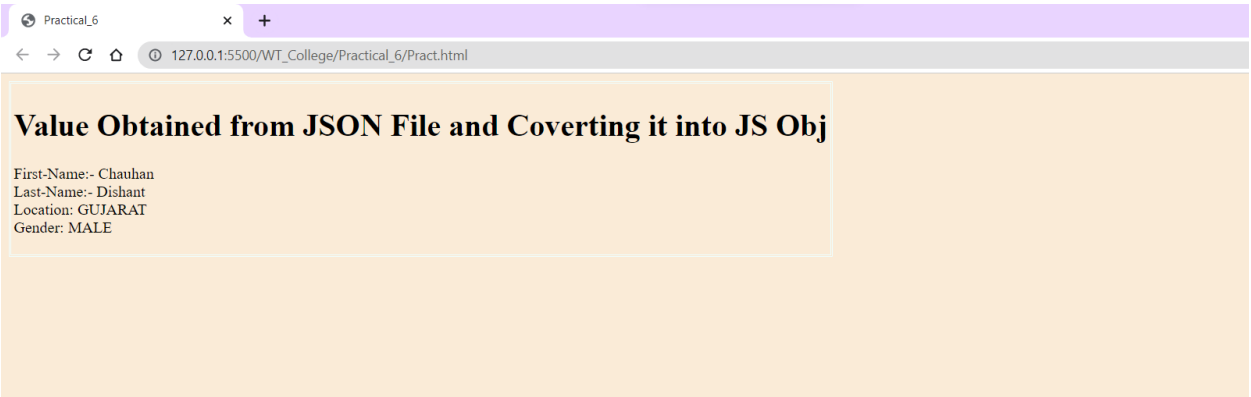
(HTML)

```
<!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>P6.2</title>
</head>
<body>
    <h1>Value fatched from json converting in js obj</h1>
    <p id="P"></p>
    <script src="Pract.js"></script>
</body>
</html>
```

(JAVSCRIPT)

```
var Dishant = '{"First_Name" : "Chauhan", "Last_Name" : "Dishant",
"Location" : "GUJARAT", "Gender" : "MALE"}';
const obj = JSON.parse(Dishant);
let t = `First-Name:- ${obj.First_Name} <br>  Last-Name:-
${obj.Last_Name} <br> Location: ${obj.Location} <br> Gender:
${obj.Gender} <br>`;
document.getElementById('P').innerHTML = t;
```

Output:



Conclusion:

In this practical I came to know about the working of the JSON file and I have learned that how the JSON file works.

AIM 3:

Do Spoural Registration Form Validation (Error object and try…catch..)

Source Code:

(HTML)

```
<!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>sP6.3</title>
</head>
<body>
    <div class="main">
        <h1>Spoural Registration Forms</h1>
        <form action="">
            <label for="name">Name:-</label>
            <input type="text" id="name">
            <br>
            <label for="id">Id  :-</label>
            <input type="text" id="id">
            <br>
            <label for="Branch">Branch</label>
            <input type="text" id="br">
            <br><input type="button" value="Submit" id="submit">
            <p id="p"></p>
        </form>
    </div>
    <style>
        .main {
            top: 25%;
            left: 25%;
            font-size: 30px;
            padding: 20px;
            width: 50%;
            height: 45%;
            background-color: #6098b0;
            position: absolute;
            display: block;
            text-align: center;
            justify-content: center;
            border: 2px solid gray;
```

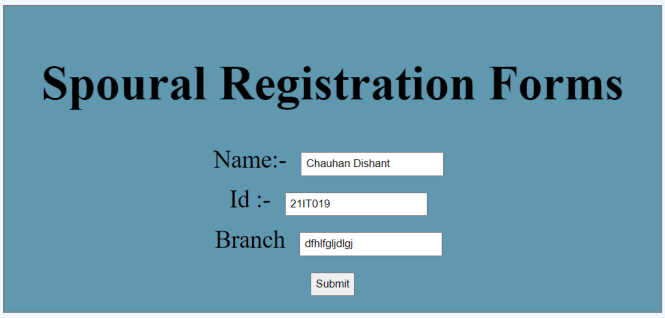
```
        }
        input {
            padding: 5px;
            margin: 10px;
        }
    </style>
    <script src="Pract.js"></script>
</body>
</html>
```

(JAVSCRIPT)

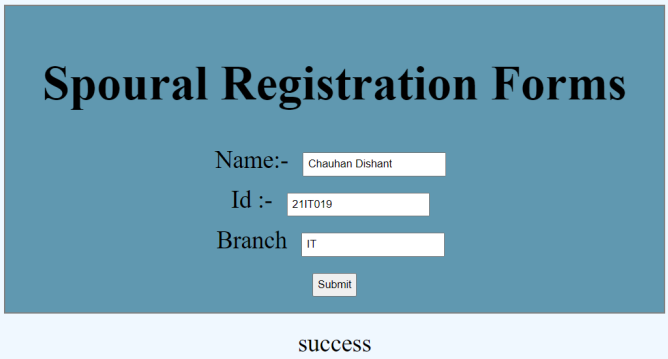
```
let btn = document.getElementById('submit');
        btn.onclick = () => {
            let name = document.getElementById('name');
            let id = document.getElementById('id');
            let br = document.getElementById('br');
            let p = document.getElementById('p');
            try {
                if (name.value == "" || id.value == "" || br.value == "") {
                    throw new Error("NULL Value NOT allowed")
                }
                else if (br.value != "IT" && br.value != "CE" && br.value !=
"CSE") {
                    throw new Error("invalid branch")
                }
                else{
                    p.innerHTML = "success"
                }
            } catch (error) {
                p.innerHTML = "Error: " + error.message;
            }
        }
```

Output:

Error: NULL Value NOT allowed



Error: invalid branch



Conclusion:

In this practical I have learned about the form validity and it's implementation and it's working.