Assignment - 3

Ques-1: Create a business logic with 20 objects and use map reduce and filter function.

Ans: A dataset is created for 20 employees working in an office.

businessLogic.js > ...

```
const firstNames = ["Amit", "Priya", "Rahul", "Neha", "Vivek", "Anjali", "Karan", "Sneha", "Ravi", "Pooja", "Sumit",
const lastNames = ["Sharma", "Verma", "Gupta", "Mehta", "Rastogi", "Singh", "Yadav", "Joshi", "Kapoor", "Chauhan", "M
var age = [28, 26, 32, 24, 29, 31, 27, 33, 30, 25, 26, 29, 35, 24, 32, 28, 31, 30, 27, 29];
var salary = [32000, 30000, 45000, 28000, 35000, 27000, 40000, 33000, 39000, 26000, 31000, 30000, 42000, 29000, 44000
let workingDays = [27, 29, 26, 30, 28, 25, 30, 26, 27, 28, 29, 30, 25, 26, 28, 27, 30, 29, 28, 30];

const employees = []; //null array

for(let i=0; i<20; i++){
    employees.push({
        firstNames : firstNames[i],
        lastNames : lastNames[i],
        age : age[i],
        salary : salary[i],
        workingDays : workingDays[i]
    }); //to enter data from above

}
console.log(employees); //to access all the employees details
```

map function

```
// map
const firstName1 = employees.map(emp => emp.firstName);
console.log(firstName1); // to get the firstname of all the employees

const lastName1 = employees.map(emp => emp.lastName);
console.log(lastName1); // to get the firstname of all the employees

const firstNameAndAge = employees.map(emp => ({Name:emp.firstName, Age:emp.age}));
console.log(firstNameAndAge); // // to get the firstname and age of all the employees

const summary = employees.map(emp => ({
  Name: emp.firstName,
  WorkingDays: emp.workingDays
}));
console.log(summary); // to get name and working days
```

reduce function

```
// reduce
const totalSalary = employees.reduce((acc,emp) => acc + emp.salary,0);
console.log("Total Salary: " + totalSalary); // To get the total salary of the employees

const avgSalary = employees.reduce((acc, emp, _, arr) => acc + emp.salary / arr.length, 0);
console.log("Average Salary:", avgSalary); // to get the average salary of the employees
```

filter function

```
// filter
const highEarners = employees.filter(emp => emp.salary > 35000).map(emp => emp.firstName + " " + emp.lastName);
console.log("High Salary Employees:", highEarners); // To know the highest earning employees

const hardWorkers = employees.filter(emp => emp.workingDays == 30).map(emp => emp.firstName + " " + emp.lastName);
console.log("Hard workers: ", hardWorkers); // To know the employees who work for the whole month
```

Ques-2 : E-commerce using callbacks and promises.
Ans: Using callback :-

```
function addToCart(product, callback) {···
} // Function to add product to the cart

function generateBill(product, callback) {···
} // Function to Generate the bill of the product

function placeOrder(product, price, callback) {
  console.log("Order placed for " + product + " costing ₹" + price);
  callback(product);
} // Function for placing an order

function confirmOrder(product, callback) {
  console.log("Order confirmed for " + product + ".");
  callback(product);
} // Function to confirm the order

function deliveryStatus(product) {
  console.log(product + " is out for delivery. Will arrive soon!");
} // Function to know the delivery status of the product

addToCart("Smartphone", function (product) {
  generateBill(product, function (product, price) {
    placeOrder(product, price, function (product) {
      confirmOrder(product, function (product) {
        deliveryStatus(product);
      });
    });
  });
}); //callback function
```

Using promises:-

```
function addToCart(product) {
  return new Promise(function (resolve) {
    console.log("Product " + product + " added to cart.");
    resolve(product);
  });
}

function generateBill(product) {
  return new Promise(function (resolve) {
    const price = Math.floor(Math.random() * 900) + 100; // ₹100 to ₹999
    console.log("Bill generated for " + product + ": ₹" + price);
    resolve({ product: product, price: price });
  });
}

function placeOrder(data) {
  return new Promise(function (resolve) {
    console.log("Order placed for " + data.product + " and the cost is ₹" + data.price);
    resolve(data.product);
  });
}

function confirmOrder(product) {
  return new Promise(function (resolve) {
    console.log("Order confirmed for " + product);
    resolve(product);
  });
}
```

```
function deliveryStatus(product) {
  console.log(product + " is out for delivery. Will arrive soon.");
}

addToCart("Smartphone")
  .then(generateBill)
  .then(placeOrder)
  .then(confirmOrder)
  .then(deliveryStatus)
  .catch(function (error) {
    console.log("Error: " + error);
  }); // promises
```

Ques-3 : Use fetch api to show your github profile using HTML, CSS and JavaScript.

```html
<!DOCTYPE html>
<html>
<head>
  <title>GitHub Profile</title>
  <style>
    body {
      font-family: sans-serif;
      text-align: center;
      padding-top: 40px;
      background: #f0f0f0;
    }
    #profile {
      background: white;
      padding: 20px;
      border-radius: 10px;
      display: inline-block;
      box-shadow: 0 0 10px rgba(0,0,0,0.1);
    }
    img {
      width: 100px;
      border-radius: 50%;
    }
    a {
      display: inline-block;
      margin-top: 10px;
      text-decoration: none;
      color: blue;
      font-weight: bold;
    }
  </style>
</head>
<body>

  <h1>GitHub Profile Viewer</h1>
  <div id="profile">Loading...</div>

  <script>
    fetch("https://api.github.com/users/Yateesh8")
      .then(res => res.json())
      .then(data => {
        document.getElementById("profile").innerHTML = `
          <img src="${data.avatar_url}" alt="Avatar">
          <h2>${data.name || data.login}</h2>
          <a href="${data.html_url}" target="_blank">
            <img src="https://github.githubassets.com/images/modules/logos_page/GitHub-Mark.png"
                 alt="GitHub" style="width: 20px; vertical-align: middle; margin-right: 5px;">
            Visit GitHub
          </a>
        `;
      });
  </script>

</body>
</html>
```