DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING (ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING)

II B.Tech I Semester

Subject Name: NODE JS/ REACT JS/ DJANGO LAB
Subject Code: C0522
Regulations: MR-22

Lab Manual

Academic Year: 2024-25

MALLA REDDY ENGINEERING COLLEGE (AUTONOMOUS)

MAIN CAMPUS

(An UGC Autonomous Institution, Approved by AICTE and Affiliated to JNTUH, Hyderabad, Accredited by NAAC with 'A++' Grade (III Cycle))

NBA Accredited Programmes – UG (CE, EEE, ME, ECE, & CSE), PG (CE-SE, EEE, EPS, ME-TE)

Maisammaguda(H), Gundlapochampally Village, Medchal Mandal,

Medchal-Malkajgiri District, Telangana State – 500100

MALLA REDDY ENGINEERING COLLEGE (AUTONOMOUS)

MR22 – ACADEMIC REGULATIONS (CBCS)

for B.Tech. (REGULAR) DEGREE PROGRAMME

Applicable for the students of B.Tech. (Regular) programme admitted from the Academic Year 2022-23 onwards

The B.Tech. Degree of Jawaharlal Nehru Technological University Hyderabad, Hyderabad shall be conferred on candidates who are admitted to the programme and who fulfill all the requirements for the award of the Degree.

VISION OF THE INSTITUTE

To be a premier center of professional education and research, offering quality programs in a socio-economic and ethical ambience.

MISSION OF THE INSTITUTE

- To impart knowledge of advanced technologies using state-of-the-art infrastructural facilities.
- To inculcate innovation and best practices in education, training and research.
- To meet changing socio-economic needs in an ethical ambience.

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING – ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

DEPARTMENT VISION

To attain global standards in Computer Science and Engineering education, training and research to meet the growing needs of the industry with socio-economic and ethical considerations.

DEPARTMENT MISSION

- To impart quality education and research to undergraduate and postgraduate students in Computer Science and Engineering.
- To encourage innovation and best practices in Computer Science and Engineering utilizing state-of-the-art facilities.
- To develop entrepreneurial spirit and knowledge of emerging technologies based on ethical values and social relevance.

PROGRAMME EDUCATIONAL OBJECTIVES (PEOs)

PEO1: Graduates will demonstrate technical skills, competency in AI & ML and exhibit team management capability with proper communication in a job environment

PEO2: Graduates will function in their profession with social awareness and responsibility

PEO3: Graduates will interact with their peers in other disciplines in industry and society and contribute to the economic growth of the country

PEO4: Graduates will be successful in pursuing higher studies in engineering or management

PROGRAMME OUTCOMES (POs)

PO1: Engineering knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

PO2: Problem analysis: Identify, formulate, review research literature and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

PO3: Design/development of solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

PO4: Conduct investigations of complex problems: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

PO5: Modern tool usage: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

PO6: The engineer and society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

PO7: Environment and sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

PO8: Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

PO9: Individual and team work: Function effectively as an individual and as a member or leader in diverse teams, and in multidisciplinary settings.

PO10: Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

PO11: Project management and finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

PO12: Life-long learning: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

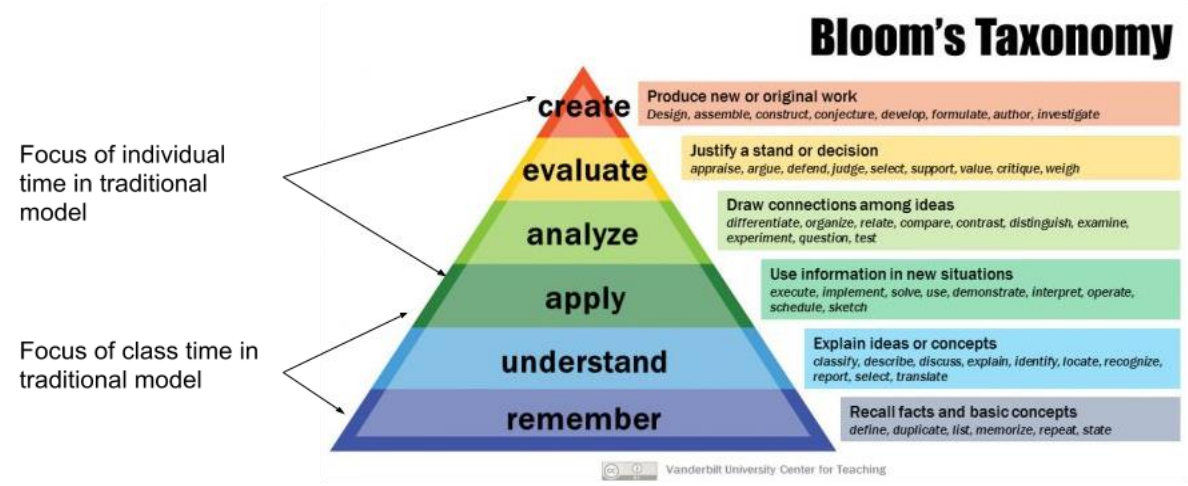
PROGRAMME SPECIFIC OUTCOMES (PSOs)

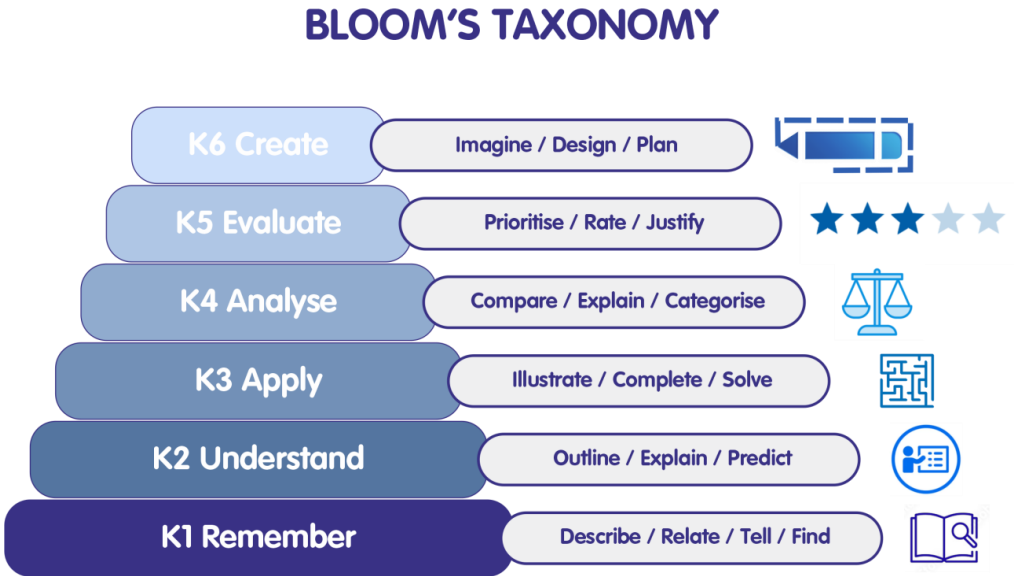
PSO1: Design and develop intelligent automated systems applying mathematical, analytical, programming and operational skills to solve real world problems

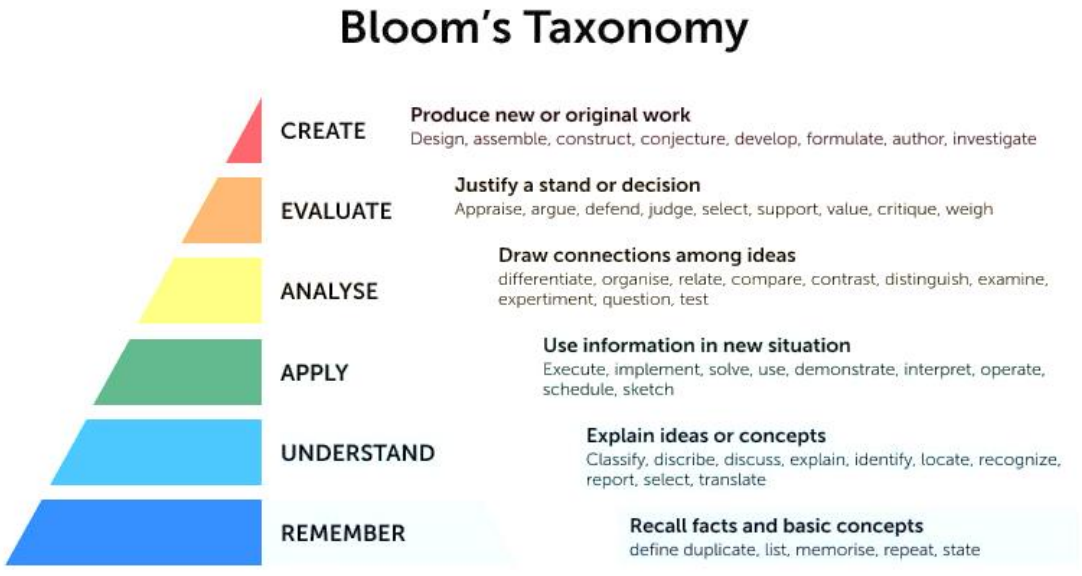
PSO2: Apply machine learning techniques, software tools to conduct experiments, interpret data and to solve complex problems

PSO3: Implement engineering solutions for the benefit of society by the use of AI and ML

BLOOM'S TAXONOMY (BT) TRIANGLE & BLOOM'S ACTION VERBS







BLOOM'S ACTION VERBS

REVISED Bloom's Taxonomy Action Verbs

Definitions	I. Remembering	II. Understanding	III. Applying	IV. Analyzing	V. Evaluating	VI. Creating
Bloom's Definition	Exhibit memory of previously learned material by recalling facts, terms, basic concepts, and answers.	Demonstrate understanding of facts and ideas by organizing, comparing, translating, interpreting, giving descriptions, and stating main ideas.	Solve problems to new situations by applying acquired knowledge, facts, techniques and rules in a different way.	Examine and break information into parts by identifying motives or causes. Make inferences and find evidence to support generalizations.	Present and defend opinions by making judgments about information, validity of ideas, or quality of work based on a set of criteria.	Compile information together in a different way by combining elements in a new pattern or proposing alternative solutions.
Verbs	• Choose • Define • Find • How • Label • List • Match • Name • Omit • Recall • Relate • Select • Show • Spell • Tell • What • When • Where • Which • Who • Why	• Classify • Compare • Contrast • Demonstrate • Explain • Extend • Illustrate • Infer • Interpret • Outline • Relate • Rephrase • Show • Summarize • Translate	• Apply • Build • Choose • Construct • Develop • Experiment with • Identify • Interview • Make use of • Model • Organize • Plan • Select • Solve • Utilize	• Analyze • Assume • Categorize • Classify • Compare • Conclusion • Contrast • Discover • Dissect • Distinguish • Divide • Examine • Function • Inference • Inspect • List • Motive • Relationships • Simplify • Survey • Take part in • Test for • Theme	• Agree • Appraise • Assess • Award • Choose • Compare • Conclude • Criteria • Criticize • Decide • Deduct • Defend • Determine • Disprove • Estimate • Evaluate • Explain • Importance • Influence • Interpret • Judge • Justify • Mark • Measure • Opinion • Perceive • Prioritize • Prove • Rate • Recommend • Rule on • Select • Support • Value	• Adapt • Build • Change • Choose • Combine • Compile • Compose • Construct • Create • Delete • Design • Develop • Discuss • Elaborate • Estimate • Formulate • Happen • Imagine • Improve • Invent • Make up • Maximize • Minimize • Modify • Original • Originate • Plan • Predict • Propose • Solution • Solve • Suppose • Test • Theory

Anderson, L. W., & Krathwohl, D. R. (2001). A taxonomy for learning, teaching, and assessing, Abridged Edition. Boston, MA: Allyn and Bacon.

2022-23 Onwards (MR-22)	MALLA REDDY ENGINEERING COLLEGE (AUTONOMOUS)	B.Tech. VI Semester		
Code: C6602	NODE JS/ REACT JS/ DJANGO LAB	L	T	P
Credits: 1		-	-	2

Course Objectives:

- To implement the static web pages using HTML and do client side validation using JavaScript.
- To design and work with databases using Java
- To develop an end to end application using java full stack.
- To introduce Node JS implementation for server side programming.
- To experiment with single page application development using React.

Software Requirements: JDK, Tomcat Server, PHP and WAMP Server.

Exercises:

1. Build a responsive web application for shopping cart with registration, login, catalog and cart pages using CSS3 features, flex and grid.
2. Make the above web application responsive web application using Bootstrap framework.
3. Use JavaScript for doing client – side validation of the pages implemented in experiment 1 and experiment 2.
4. Explore the features of ES6 like arrow functions, callbacks, promises, async/await. Implement an application for reading the weather information from openweathermap.org and display the information in the form of a graph on the web page.
5. Develop a java stand alone application that connects with the database (Oracle / mySql) and perform the CRUD operation on the database tables.
6. Create an xml for the bookstore. Validate the same using both DTD and XSD.
7. Design a controller with servlet that provides the interaction with application developed in experiment 1 and the database created in experiment 5.
8. Maintaining the transactional history of any user is very important. Explore the various session tracking mechanism (Cookies, HTTP Session)
9. Create a custom server using http module and explore the other modules of Node JS like OS,path, event.
10. Develop an express web application that can interact with REST API to perform CRUD operations on student data. (Use Postman)

11. For the above application create authorized end points using JWT (JSON Web Token).
12. Create a react application for the student management system having registration, login, contact, about pages and implement routing to navigate through these pages.
13. Create a service in react that fetches the weather information from openweathermap.org and the display the current and historical weather information using graphical representation usingchart.js
14. Create a TODO application in react with necessary components and deploy it into github.

Course Outcomes:

At the end of the course, the student will be able to,

- Build a custom website with HTML, CSS, and Bootstrap and little JavaScript.
- Demonstrate Advanced features of JavaScript and learn about JDBC
- Develop Server – side implementation using Java technologies like
- Develop the server – side implementation using Node JS.
- Design a Single Page Application using React.

Reference Books:

1. Jon Duckett, Beginning HTML, XHTML, CSS, and JavaScript, Wrox Publications, 2010
2. Bryan Basham, Kathy Sierra and Bert Bates, Head First Servlets and JSP, O'Reilly Media, 2ndEdition, 2008.
3. Vasan Subramanian, Pro MERN Stack, Full Stack Web App Development with Mongo, Express, React, and Node, 2nd Edition, A Press.

CO- PO, PSO Mapping
(3/2/1 indicates strength of correlation) 3-Strong, 2-Medium, 1-Weak

COs	Programme Outcomes (POs)												PSOs		
	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3
CO1	2	3	1									2	1		
CO2	2	2										2	2		
CO3	1	2										1	1		

1. Build a responsive web application for shopping cart with registration, login, catalog and cart pages using CSS3 features, flex and grid.

index.html

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

<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Shopify</title>
<link rel="stylesheet" href="./css/style.css">
<link rel="stylesheet" href="./css/responsive.css">
<link
    href="https://fonts.googleapis.com/css2?family=Poppins:ital,wght@1,900&display=
    swap" rel="stylesheet">
  <scriptsrc="https://kit.fontawesome.com/1ba2cf90f3.js"
  crossorigin="anonymous"></script>
<link rel="shortcut icon" href="./assests/Citycons_bag-512.webp" type="image/x-
    icon">
</head>
<body>
<section>
<nav>
<span><a href="index.html"><i class="fas fa-shopping-bag"></i>shopping
    cart</a></span>
<ul>
<li><a href="index.html">Home</a></li>
<li><a href="signup.html">signup</a></li>
<li><a href="login.html">login</a></li>
<li><span onclick="openBucket()"><i class="fas fa-shopping-cart"></i></span>
<span class="items-count"></span>
</li>
</ul>
<span onclick="openBucket()" class="cart-icon"><i class="fas fa-shopping-
    cart"></i></span>
</nav>
<main class="fruits-side">
<span>Buy Fruits</span>
<button><a href="fruit-aisle.html">Shop Now <i class="fas fa-caret-
    right"></i></a></button>
</main>
<div class="cart">
<header><span><a href="index.html"><i class="fas fa-shopping-bag"></i>
    Shopify</a></span><span
            class="exit"><i class="fas fa-times"></i></span></header>
<main>
<ol>
<li>Item-Name</li>
<li>Qty</li>
<li>Total</li>
</ol>
<section>
<ul id="ul"></ul>
```

```
</section>
</main>
</div>
</section>
<script>
      const top_items_count = document.querySelector('.items-count'),
         bottom_items_count = document.querySelector('.bottom-items-count'),
         exit = document.querySelector('.exit'),
         bucket = document.querySelector('.cart').style;
      top_items_count.innerHTML = 0;
      bottom_items_count.innerHTML = 0;
      function openBucket() {
         bucket.visibility = "visible";
         bucket.opacity = "1";
         bucket.zIndex = "9";
         bucket.transition = "all 0.5s";
      }
      exit.addEventListener('click', () => {
         bucket.visibility = "hidden";
         bucket.opacity = "0";
         bucket.zIndex = "-9";
         bucket.transition = "all 0.5s";
      });
      top_items_count.innerHTML = count;
      bottom_items_count.innerHTML = count;
</script>
</body>
    </html>
```

Registration: signup.html

```
<!DOCTYPE html>
<html lang="en">
<html>
<head>
<title>Sign Up</title>
<meta name="viewport" content="width=device-width, initial-scale=1.0" />
<meta charset="utf-8" />
<link rel="stylesheet" type="text/css" href="signup_style.css" />
<link
      rel="stylesheet"
      href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-
    awesome.min.css"
    />
<link
      href="https://fonts.googleapis.com/css?family=Titillium+Web:400,300,600"
      rel="stylesheet"
      type="text/css"
    />
<script                src="https://unpkg.com/@lottiefiles/lottie-player@latest/dist/lottie-
    player.js"></script>
<link  rel="stylesheet"  href="https://use.fontawesome.com/releases/v5.15.1/css/all.css"
    integrity="sha384-
```

```
    vp86vTRFVJgpjF9jiIGPEEqYqlDwgyBgEF109VFjmqGmIY/Y4HV4d3Gp2irVfcrp
    " crossorigin="anonymous">
</head>
<div class="login-page">
<div class="form">
<form>
<lottie-player

    src="https://assets4.lottiefiles.com/datafiles/XRVoUu3IX4sGWtiC3MPpFnJvZNq7l
    VWDCa8LSqgS/profile.json"
        background="transparent"
        speed="1"
        style="justify-content: center"
        loop
        autoplay
></lottie-player>
<input type="text" placeholder="full name" />
<input type="text" placeholder="email address" />
<input type="text" placeholder="pick a username" />
<input type="password" id="password" placeholder="set a password" />
<i class="fas fa-eye" onclick="show()"></i>
<br>
<br>
</form>
<form class="login-form">
<button type="button" onclick="window.location.href='login.html'">
        SIGN UP
</button>
</form>
</div>
</div>
</body>
<script>
    function show() {
      var password = document.getElementById("password");
      var icon = document.querySelector(".fas");
      // ========== Checking type of password ==========
      if (password.type === "password") {
        password.type = "text";
      } else {
        password.type = "password";
      }
    }
</script>
</html>
    </html>
```

SignIn : login.html<!DOCTYPE HTML>

```
<html lang="en" >
<html>
<head>
<title>Login</title>
<meta name="viewport" content="width=device-width, initial-scale=1.0">
```

```
        <meta charset="utf-8">
        <link rel="stylesheet" type="text/css" href="login_style.css">
        <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
            awesome/4.7.0/css/font-awesome.min.css">
        <link href='https://fonts.googleapis.com/css?family=Titillium+Web:400,300,600'
            rel='stylesheet' type='text/css'>
        <link href='https://fonts.googleapis.com/css?family=Titillium+Web:400,300,600'
            rel='stylesheet' type='text/css'>
        <script src="https://unpkg.com/@lottiefiles/lottie-player@latest/dist/lottie-
            player.js"></script>
        <link rel="stylesheet"
            href="https://use.fontawesome.com/releases/v5.15.1/css/all.css"
            integrity="sha384-
            vp86vTRFVJgpjF9jiIGPEEqYqlDwgyBgEF109VFjmqGmIY/Y4HV4d3Gp2irVf
            crp" crossorigin="anonymous">
        </head>
<div class="login-page">
        <div class="form">
        <form>
        <lottie-player
            src="https://assets4.lottiefiles.com/datafiles/XRVoUu3IX4sGWtiC3MPpFnJvZN
            q7lVWDCa8LSqgS/profile.json" background="transparent" speed="1"
            style="justify-content: center;" loop  autoplay></lottie-player>
        <input type="text" placeholder="&#xf007;  username"/>
        <input type="password" id="password" placeholder="&#xf023;  password"/>
        <i class="fas fa-eye" onclick="show()"></i>
        <br>
        <br>
        <button>LOGIN</button>
        <p class="message"></p>
        </form>
        <form class="login-form">
        <button type="button" onclick="window.location.href='signup.html'">SIGN
            UP</button>
        </form>
        </div>
        </div>
        <script>
           function show(){
            var password = document.getElementById("password");
            var icon = document.querySelector(".fas")
            // ========== Checking type of password ==========
            if(password.type === "password"){
              password.type = "text";
            }

        else {
              password.type = "password";
            }
           };
        </script>
        </body>
        </html>
```

Shopping cart: fruit-aisle.html(shopping fruits)

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Fruit-Aisle</title>
<link rel="stylesheet" href="./css/style3.css">
<link rel="stylesheet" href="./css/responsive.css">
<link
    href="https://fonts.googleapis.com/css2?family=Poppins:ital,wght@1,900&displa
    y=swap" rel="stylesheet">
<script                                    src="https://kit.fontawesome.com/1ba2cf90f3.js"
    crossorigin="anonymous"></script>
<link rel="shortcut icon" href="./assests/Citycons_bag-512.webp" type="image/x-
    icon">
</head>
<body>
<section>
<nav>
<span><a href="index.html"><i class="fas fa-shopping-bag"></i> Shopify shopping
    cart</a></span>
<ul>
<li><a href="index.html">Home</a></li>
<li><a href="signup.html">signup</a></li>
<li><a href="login.html">login</a></li>
<li><span onclick="openBucket()"><i class="fas fa-shopping-cart"></i></span>
<span class="items-count"></span>
</li>
</ul>
<span onclick="openBucket()" class="cart-icon"><i class="fas fa-shopping-
    cart"></i></span>
</nav>
<main>
<div class="items item-1" data-item="Apples" data-price="100"></div>
<div class="items item-2" data-item="Lemons" data-price="200"></div>
<div class="items item-3" data-item="Strawberries" data-price="300"></div>
<div class="items item-4" data-item="Bananas" data-price="400"></div>
</main>




<div class="cart">
<header><span><a href="index.html"><i class="fas fa-shopping-bag"></i>
    Shopify</a></span><span
            class="exit"><i class="fas fa-times"></i></span></header>
<main>
<ol>
<li>Item-Name</li>
<li>Qty</li>
```

```
        <li>Total</li>
        </ol>
        <section>
        <ul id="ul">
        </ul>
        </section>
        </main>
        </div>
        </section>
        <script src="./js/script3.js"></script>
        </body>
</html>
```

Responsive.css

```
        @media (min-width: 0px) and (max-width: 425px) {
          section > nav {
            grid-template-columns: auto 13vw;
          }
          section > nav > ul {
            display: none;
          }
          .cart-icon {
            color: #323232;
            font-size: 1.3rem;
          }
          .fruits-side > span {
            font-size: 3.3rem;
            padding-top: 80px;
          }
          .fruits-side > button {
            height: 10vh;
            width: 53vw;
            font-size: 1.2rem;
            letter-spacing: 1px;
            margin-bottom: 30px;
          }









         .cart {
            position: fixed;
            height: 100vh;
            width: 100%;
            top: 0%;
            left: 0%;
            transform: scale(0.9);
          }
          .cart > main > ol > li:nth-child(1) {
            padding: 0px 12px;
```

```
  }
 }
 @media (min-width: 426px) and (max-width: 1023px) {
  section > nav {
   grid-template-columns: auto 60vw;
  }
  section > nav > span {
   font-size: 1.6rem;
  }
  section > nav > ul > li > a {
   font-size: 1.1rem;
  }
  .fruits-side > button {
   height: 10vh;
   width: 25vw;
   font-size: 1.2rem;
   letter-spacing: 1px;
   margin-bottom: 30px;
  }
  .cart {
   position: fixed;
   height: 100vh;
   width: 100%;
   top: 0%;
   left: 0%;
   transform: scale(0.9);
  }
  .cart > main > ol > li:nth-child(1) {
   padding: 0px 12px;
  }
  section > footer {
   grid-template-columns: auto 60vw;
  }
  section > footer > ul > li {
   padding: 10px 15px;
  }
}
```

Style.css

```
 *,
 body {
  margin: 0;
  padding: 0;
  list-style: none;
  text-decoration: none;
  outline: none;
  letter-spacing: 1px;
  transition: all 0.5s;
  color: inherit;
  scroll-behavior: smooth;
```

```
  font-family: "Poppins", sans-serif;
}
section {
  position: relative;
  width: 100%;
  display: grid;
  grid-template-columns: 0.99fr;
  grid-template-rows: 15vh repeat(3, 80vh) 15vh;
  align-content: space-around;
  justify-content: space-evenly;
}
section > nav {
  display: grid;
  grid-template-columns: 15vw 50vw;
  grid-template-rows: 0.8fr;
  align-content: space-around;
  justify-content: space-around;
  background-color: whitesmoke;
}
section > nav > span {
  display: flex;
  align-items: center;
  justify-content: space-evenly;
  font-weight: bold;
  font-size: 2rem;
  color: grey;
  text-transform: capitalize;
}
section > nav > span > a > i {
  color: #323232;
}
section > nav > ul {
  display: flex;
  align-items: center;
  justify-content: space-evenly;
}



section > nav > ul > li {
  position: relative;
}
section > nav > ul > li > a {
  font-weight: bold;
  font-size: 1.3rem;
  letter-spacing: 1px;
  color: grey;
}
section > nav > ul > li > a > i {
  color: #323232;
  font-size: 2rem;
}
.cart-icon{
```

```
  display: none;
}
.items-count {
  position: absolute;
  padding: 5px;
  background-color: red;
  color: white;
  font-weight: bold;
  font-size: 0.9rem;
  border-radius: 30px;
  top: -40%;
}
.fruits-side {
  background-image: url("../assests/62.jpg");
  background-size: 100% 100%;
  display: grid;
  align-content: space-around;
  justify-content: space-evenly;
}
.fruits-side > span {
  font-weight: bold;
  font-size: 8rem;
  color: white;
  text-transform: capitalize;
  padding-top: 80px;
}
.fruits-side > button {
  height: 8.5vh;
  width: 13vw;
  background-color: transparent;
  border: 1.5px solid white;
  border-radius: 10px;
  font-weight: bold;
  font-size: 1.2rem;
  color: white;


 letter-spacing: 1px;
  justify-self: center;
  margin-bottom: 30px;
}
.fruits-side > button > a {
  display: flex;
  align-items: center;
  justify-content: space-evenly;
}
.fruits-side > button:hover {
  transition: all 0.5s;
  cursor: pointer;
  background-color: white;
  border: none;
  color: rgba(0, 0, 0, 0.5);
}
```

```
.fruits-side > button > a > i {
  font-size: 1.8rem;
}
.bottom-items-count {
  position: absolute;
  padding: 5px;
  background-color: red;
  color: white;
  font-weight: bold;
  font-size: 0.9rem;
  border-radius: 30px;
  bottom: 40%;
}
.cart {
  position: fixed;
  height: 98.5%;
  width: 25vw;
  background-color: whitesmoke;
  top: 0.5%;
  right: 0.5%;
  display: grid;
  grid-template-columns: 0.95fr;
  grid-template-rows: 10vh 60vh 10vh;
  align-content: space-around;
  justify-content: space-evenly;
  box-shadow: 0px 0px 10px 5px rgba(0, 0, 0, 0.5);
  border-radius: 5px;
  opacity: 0;
  visibility: hidden;
  z-index: -9;
  transition: all 0.5s;
}



.cart > header {
  display: flex;
  align-items: center;
  justify-content: space-around;
}
.cart > header > span {
  display: flex;
  align-items: center;
  justify-content: space-evenly;
  font-weight: bold;
  font-size: 1.5rem;
  color: grey;
  text-transform: capitalize;
}
.cart > header > span > a > i {
  color: #323232;
}
.exit > i {
```

```
  font-size: 2rem;
  color: #323232;
  cursor: pointer;
  transition: all 0.5s;
}
.cart > main {
  box-shadow: 0px 0px 10px 1px rgba(0, 0, 0, 0.5);
  border-radius: 5px;
  display: grid;
  grid-template-columns: 1fr;
  grid-template-rows: 5vh 1fr;
  align-content: space-around;
  justify-content: space-evenly;
  transition: all 0.5s;
}
.cart > main > ol {
  display: flex;
  align-items: center;
  justify-content: space-around;
}
.cart > main > ol > li {
  font-weight: lighter;
  border: 1px solid rgba(0, 0, 0, 0.5);
  padding: 0px 21px;
  color: grey;
}




.cart > main > section {
  overflow: scroll;
  transition: all 0.5s;
  display: grid;
  grid-template-columns: 1fr;
  grid-template-rows: 1fr;
  align-content: space-around;
  justify-content: space-evenly;
}
.cart > main > section > ul > li {
  display: flex;
  align-items: center;
  justify-content: space-between;
  height: 5vh;
  border-bottom: 1px solid rgba(0, 0, 0, 0.5);
  cursor: pointer;
  transition: all 0.5s;
}
```

Style3.css

```
*,
body {
  margin: 0;
  padding: 0;
  list-style: none;
  text-decoration: none;
  outline: none;
  letter-spacing: 1px;
  transition: all 0.5s;
  color: inherit;
  scroll-behavior: smooth;
  font-family: "Poppins", sans-serif;
}

section {
  width: 100%;
  display: grid;
  grid-template-columns: 0.99fr;
  grid-template-rows: 15vh 90vh 15vh;
  align-content: space-around;
  justify-content: space-evenly;
  transition: all 0.5s;
}




section > nav {
  display: grid;
  grid-template-columns: 15vw 50vw;
  grid-template-rows: 0.8fr;
  align-content: space-around;
  justify-content: space-around;
  background-color: whitesmoke;
}
section > nav > span {
  display: flex;
  align-items: center;
  justify-content: space-evenly;
  font-weight: bold;
  font-size: 2rem;
  color: grey;
  text-transform: capitalize;
}
section > nav > span > a > i {
  color: #323232;
}

section > nav > ul {
```

```
  display: flex;
  align-items: center;
  justify-content: space-evenly;
}

section > nav > ul > li {
  position: relative;
}
section > nav > ul > li > a {
  font-weight: bold;
  font-size: 1.3rem;
  letter-spacing: 1px;
  color: grey;
}
section > nav > ul > li > a > i {
  color: #323232;
  font-size: 2rem;
}
.items-count {
  position: absolute;
  padding: 5px;
  background-color: red;
  color: white;
  font-weight: bold;
  font-size: 0.9rem;
  border-radius: 30px;
  top: -40%;
}

.cart-icon {
  display: none;
}

section > main {
  display: grid;
  grid-template-columns: repeat(4, 23.5vw);
  grid-template-rows: 55vh;
  align-content: space-around;
  justify-content: space-evenly;
}

.items {
  border-radius: 5px;
  position: relative;
  transition: all 0.5s;
}
.item-1 {
  background-image:              url("../assests/fruit-img/close-up-photography-of-apples-
     672101.jpg");
  background-size: 100% 100%;
}
item-1::after {
  content: "Apples";
```

```
  position: absolute;
  bottom: -13.5%;
  left: 0;
  font-weight: bold;
  font-size: 1.75rem;
  color: #323232;
}
.item-1::before {
  content: "Add To Cart";
  position: absolute;
  font-weight: bold;
  font-size: 1.2rem;
  color: white;
  background-color: transparent;
  padding: 10px;
  border-radius: 20px;
  cursor: pointer;
  left: 27%;
  top: 45%;
  cursor: pointer;
  opacity: 0;
  visibility: hidden;
  transition: all 0.5s;
  border: 2px solid white;
}


.item-1:hover::before {
  opacity: 1;
  visibility: visible;
  transition: all 0.5s;
}

.item-1:hover {
  background-image: linear-gradient(rgba(0, 0, 0, 0.3), rgba(0, 0, 0, 0.3)),
    url("../assests/fruit-img/close-up-photography-of-apples-672101.jpg");
}

.item-2 {
  background-image: url("../assests/fruit-img/three-yellow-citrus-952360.jpg");
  background-size: 100% 100%;
}

.item-2::after {
  content: "Lemons";
  position: absolute;
  bottom: -13.5%;
  left: 0;
  font-weight: bold;
  font-size: 1.75rem;
  color: #323232;
}
```

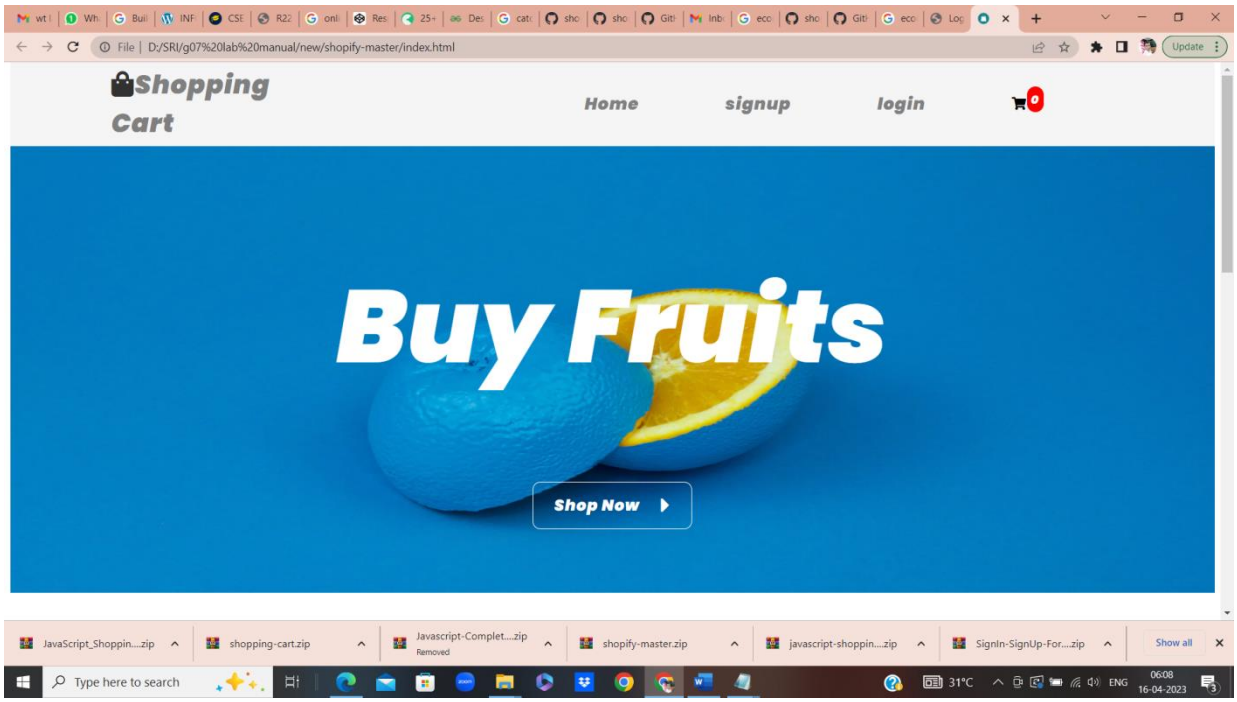
```
.item-2::before {
  content: "Add To Cart";
  position: absolute;
  font-weight: bold;
  font-size: 1.2rem;
  color: white;
  background-color: transparent;
  padding: 10px;
  border-radius: 20px;
  cursor: pointer;
  left: 27%;
  top: 45%;
  cursor: pointer;
  opacity: 0;
  visibility: hidden;
  transition: all 0.5s;
  border: 2px solid white;
}

.item-2:hover::before {
  opacity: 1;
  visibility: visible;
  transition: all 0.5s;
}

.item-2:hover {
  background-image: linear-gradient(rgba(0, 0, 0, 0.3), rgba(0, 0, 0, 0.3)),
    url("../assests/fruit-img/three-yellow-citrus-952360.jpg");
}

.item-3 {
  background-image:          url("../assests/fruit-img/white-bowl-of-whole-strawberries-
     89778.jpg");
  background-size: 100% 100%;
}

.item-3::after {
  content: "Strawberries";
  position: absolute;
  bottom: -13.5%;
  left: 0;
  font-weight: bold;
  font-size: 1.75rem;
  color: #323232;
}

.item-3::before {
  content: "Add To Cart";
  position: absolute;
  font-weight: bold;
  font-size: 1.2rem;
  color: white;
  background-color: transparent;
```

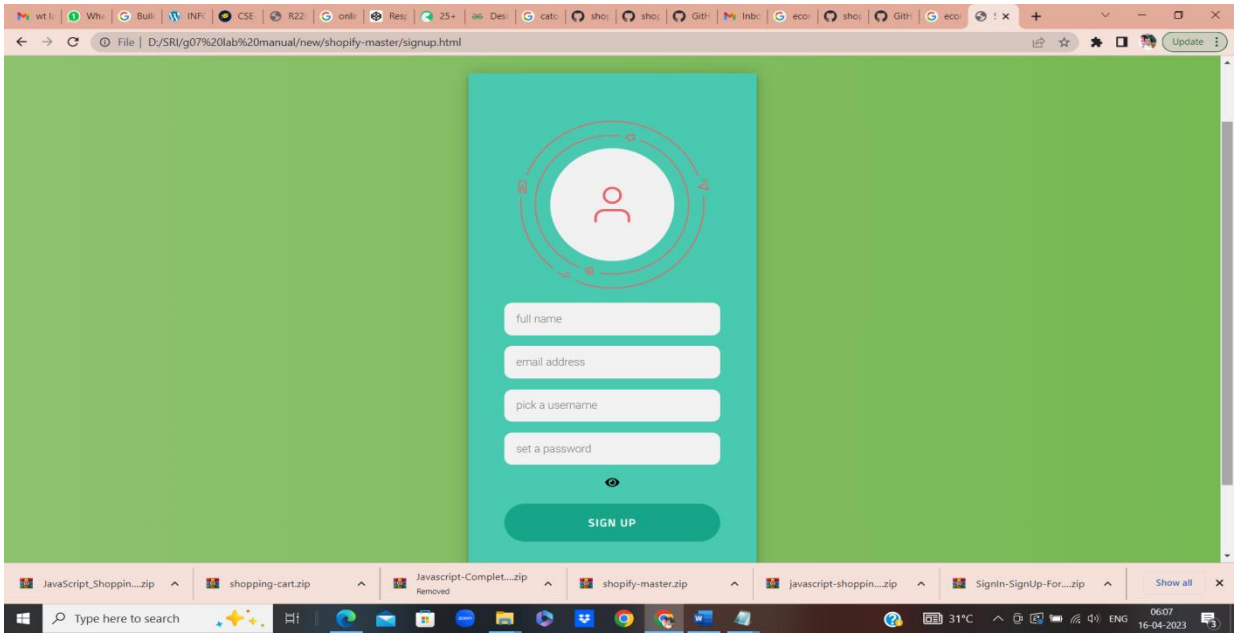
```
  padding: 10px;
  border-radius: 20px;
  cursor: pointer;
  left: 27%;
  top: 45%;
  cursor: pointer;
  opacity: 0;
  visibility: hidden;
  transition: all 0.5s;
  border: 2px solid white;
}

.item-3:hover::before {
  opacity: 1;
  visibility: visible;
  transition: all 0.5s;
}

.item-3:hover {
  background-image: linear-gradient(rgba(0, 0, 0, 0.3), rgba(0, 0, 0, 0.3)),
    url("../assests/fruit-img/white-bowl-of-whole-strawberries-89778.jpg");
}

.item-4 {
  background-image: url("../assests/fruit-img/yellow-bananas-61127.jpg");
  background-size: 100% 100%;
}

.item-4::after {
  content: "Bananas";
  position: absolute;
  bottom: -13.5%;
  left: 0;
  font-weight: bold;
  font-size: 1.75rem;
  color: #323232;
}

.item-4::before {
  content: "Add To Cart";
  position: absolute;
  font-weight: bold;
  font-size: 1.2rem;
  color: white;
  background-color: transparent;
  padding: 10px;
  border-radius: 20px;
  cursor: pointer;
  left: 27%;
  top: 45%;
  cursor: pointer;
  opacity: 0;
  visibility: hidden;
```

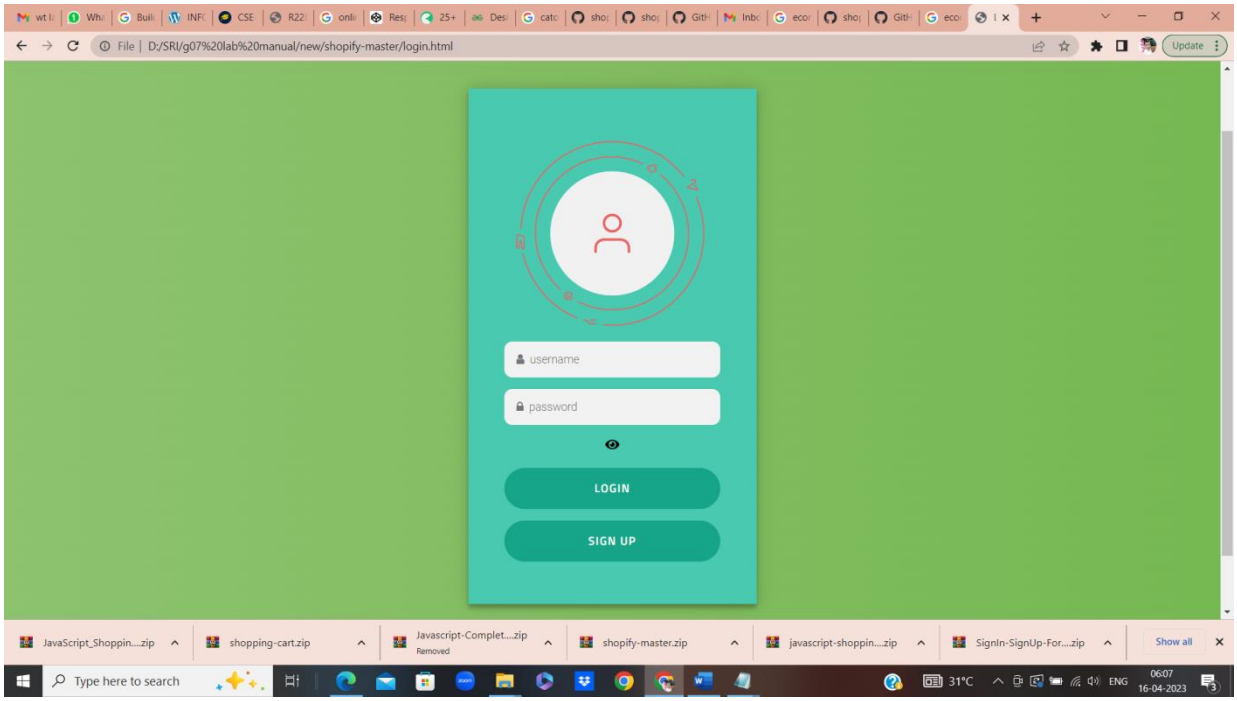
```
  transition: all 0.5s;
  border: 2px solid white;
}

.item-4:hover::before {
  opacity: 1;
  visibility: visible;
  transition: all 0.5s;
}

.item-4:hover {
  background-image: linear-gradient(rgba(0, 0, 0, 0.3), rgba(0, 0, 0, 0.3)),
    url("../assests/fruit-img/yellow-bananas-61127.jpg");
}

@media (min-width: 0px) and (max-width: 425px) {
  section {
    grid-template-rows: 15vh 250vh 15vh;
  }

  section > main {
    grid-template-columns: repeat(1, 85vw);
    grid-template-rows: repeat(4, 50vh);
  }

  .item-1::before,
  .item-2::before,
  .item-3::before,
  .item-4::before {
    padding: 10px 15px;
    left: 20%;
    top: 43%;
  }

  .item-1::after,
  .item-2::after,
  .item-3::after,
  .item-4::after {
    bottom: -18%;
    font-size: 1.5rem;
  }

  .item-4 {
    margin-bottom: 10%;
  }
}
@media (min-width: 426px) and (max-width: 1023px) {
  section {
    grid-template-rows: 15vh 140vh 15vh;
  }
  section > main {
    grid-template-columns: repeat(2, 35vw);
    grid-template-rows: repeat(2, 55vh);
```

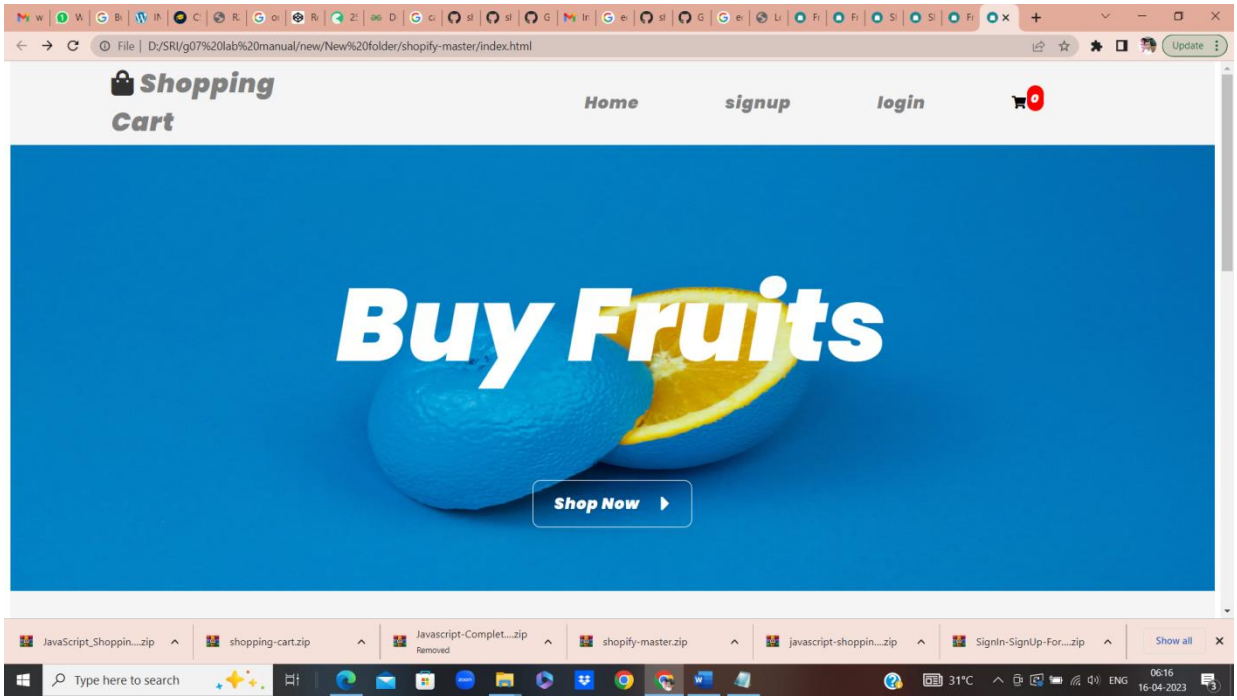
```
  }
 .item-1::before,
 .item-2::before,
 .item-3::before,
 .item-4::before {
   padding: 10px 15px;
   left: 20%;
   top: 43%;
 }
 .item-1::after,
 .item-2::after,
 .item-3::after,
 .item-4::after {
   bottom: -18%;
   font-size: 1.5rem;
 }
}
```

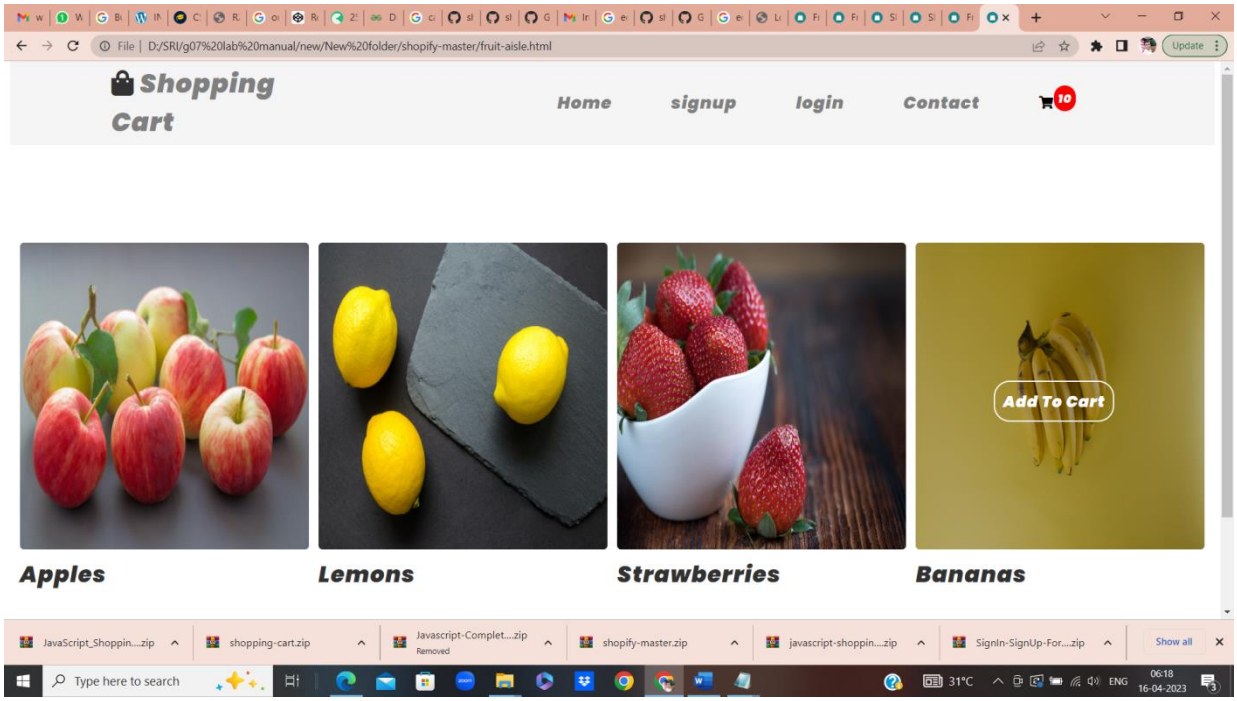
OUTPUT:




File | D:/SRI/g07%20lab%20manual/new/shopify-master/signup.html
Update
full name
email address
pick a username
set a password
SIGN UP
JavaScript_Shoppin....zip
shopping-cart.zip
Javascript-Complet....zip
Removed
shopify-master.zip
javascript-shoppin....zip
SignIn-SignUp-For....zip
Show all
Type here to search
31°C
ENG
06:07
16-04-2023


File | D:/SRI/g07%20lab%20manual/new/shopify-master/login.html
Update
username
password
LOGIN
SIGN UP
JavaScript_Shoppin....zip
shopping-cart.zip
Javascript-Complet....zip
Removed
shopify-master.zip
javascript-shoppin....zip
SignIn-SignUp-For....zip
Show all
Type here to search
31°C
ENG
06:07
16-04-2023


File | D:/SRI/g07%20lab%20manual/new/New%20folder/shopify-master/index.html
Update
Shopping Cart
Home
signup
login
0
Buy Fruits
Shop Now
JavaScript_Shoppin....zip
shopping-cart.zip
Javascript-Complet....zip
Removed
shopify-master.zip
javascript-shoppin....zip
SignIn-SignUp-For....zip
Show all
Type here to search
31°C
ENG
06:16
16-04-2023


File | D:/SRI/g07%20lab%20manual/new/New%20folder/shopify-master/fruit-aisle.html
Update
Shopping Cart
Home
signup
login
Contact
10
Add To Cart
Apples
Lemons
Strawberries
Bananas
JavaScript_Shoppin....zip
shopping-cart.zip
Javascript-Complet....zip
Removed
shopify-master.zip
javascript-shoppin....zip
SignIn-SignUp-For....zip
Show all
Type here to search
31°C
ENG
06:18
16-04-2023

2. Make the above web application responsive web application using Bootstrap framework

```
<!DOCTYPE html>
<html lang="en" >
<head>
<meta charset="UTF-8">
<title>Bootstrap 4 - Shopping Cart</title>
  <link rel="stylesheet" href="assets/bootstrap/bootstrap.min.css">
  <script src="assets/js/jquery.min.js"></script>
  <link rel="stylesheet" href="assets/fontawesome/css/all.min.css">
  <script src="assets/js/bootstrap.min.js"></script>
  <link rel="stylesheet" href="assets/css/style.css">
  <link rel="icon" href="images/favicon.ico" type="image/x-icon" />
  </head>
<body>
<!-- partial:index.partial.html -->
<!-- Nav -->
<nav class="navbar navbar-inverse bg-inverse fixed-top bg-faded">
<div class="row">
<div class="col">
                <span data-toggle="modal" data-target="#cart">
                        <i class="fas fa-shopping-cart fa-2x fa-border icon-
grey zoom-sm"><span class="badge total-count"></span></i>
                </span>
          <button class="clear-cart btn btn-danger ml-2" id="clearCart">
                <i class="fas fa-undo"></i> Clear
          </button>
          </div>
</div>
```

```
</nav>

<!-- Main -->
<div class="container">
  <br>
  <h3 class="text-center">Bootstrap 4 - Shopping Cart</h3>
<hr>


<div class="row">
    <div class="col-sm-12 col-md-6 col-xl-4">
        <figure class="card card-product zoom">
            <div class="img-wrap"><img src="images/item-0.jpg" class="rounded p-
    1"></div>
            <figcaption class="info-wrap">
                <h4 class="title">Beautiful dress</h4>
                <p class="desc">Some small description goes here</p>
                <div class="rating-wrap">
                    <div class="label-rating">132 reviews</div>
                    <div class="label-rating">154 orders </div>
                    <br/>
                    <small class="text-muted">&#9733; &#9733;
    &#9733; &#9733; &#9734;</small>
                </div><!-- rating-wrap.// -->
            </figcaption>
            <div class="bottom-wrap">
                <a href="" class="btn btn-sm btn-primary float-right add-to-cart
    show-toast" data-product_id="1" data-pimage="images/item-0.jpg" data-
    name="Beautiful dress" data-price="220">
                    <i class="fas fa-shopping-cart"></i> Add to cart
                </a>
                <div class="price-wrap h5">
                    <span class="price-new">$220</span><del class="price-
    old">$300</del>
                </div><!-- price-wrap.// -->
            </div><!-- bottom-wrap.// -->
        </figure>
    </div><!-- col // -->
    <div class="col-sm-12 col-md-6 col-xl-4">
        <figure class="card card-product zoom">
            <div class="img-wrap"><img src="images/item-1.jpg" class="rounded p-
    1"></div>
            <figcaption class="info-wrap">
                <h4 class="title">Elegant style</h4>
                <p class="desc">Some small description goes here</p>
                <div class="rating-wrap">
                    <div class="label-rating">132 reviews</div>
                    <div class="label-rating">154 orders </div>
                    <br/>
                    <small class="text-muted">&#9733; &#9733;
    &#9733; &#9733; &#9734;</small>
                </div><!-- rating-wrap.// -->
            </figcaption>
            <div class="bottom-wrap">
                <a href="" class="btn btn-sm btn-primary float-right add-
    to-cart show-toast" data-product_id="2" data-pimage="images/item-1.jpg" data-
```

```
        name="Elegant style" data-price="115">
                                        <i class="fas fa-shopping-cart"></i> Add to cart
                                    </a>
                                    <div class="price-wrap h5">
                                        <span class="price-new">$115</span><del
        class="price-old">$180</del>
                                    </div><!-- price-wrap.// -->
                                </div><!-- bottom-wrap.// -->
                            </figure>
        </div><!-- col // -->
        <div class="col-sm-12 col-md-6 col-xl-4">
                            <figure class="card card-product zoom">
                                <div class="img-wrap"><img src="images/item-2.jpg" class="rounded p-
        1"></div>
                                <figcaption class="info-wrap">
                                    <h4 class="title">Modern fashion</h4>
                                    <p class="desc">Some small description goes here</p>
                                    <div class="rating-wrap">
                                        <div class="label-rating">132 reviews</div>
                                        <div class="label-rating">154 orders </div>
                                        <br/>
                                        <small class="text-muted">&amp;#9733; &amp;#9733;
        &amp;#9733; &amp;#9733; &amp;#9734;</small>
                                    </div><!-- rating-wrap.// -->
                                </figcaption>
                                <div class="bottom-wrap">
                                    <a href="" class="btn btn-sm btn-primary float-right add-
        to-cart show-toast" data-product_id="3"  data-pimage="images/item-2.jpg" data-
        name="Modern fashion" data-price="680">
                                        <i class="fas fa-shopping-cart"></i> Add to cart
                                    </a>
                                    <div class="price-wrap h5">
                                        <span class="price-new">$680</span><del
        class="price-old">$980</del>
                                    </div><!-- price-wrap.// -->
                                </div><!-- bottom-wrap.// -->
                            </figure>
        </div><!-- col // -->
    </div><!-- row.// -->

    <div class="row p-2">
        <p class="text-info font-weight-bold">
            Easy to add to a new or existing project. Requires Bootstrap 4 Css & Js, Jquery, and
        Fontawesome. Just add the resources <code class="text-danger">*/style.css</code> and
        <code class="text-danger">*/script.js</code> to get going.
        </p>
        <p class="text-info font-weight-bold">
            Customize the layout to your own preference. All you need is to provide the
        <kbd>Add to Cart</kbd> buttton with the <code class="text-danger">data-
        product_id</code>, <code class="text-danger">data-pimage</code>
            <code class="text-danger">data-name</code>, and <code class="text-
        danger">data-price</code> attributes.
        </p>
        <p class="text-info font-weight-bold">
            The cart form data is generated with the function <code class="text-
        danger">displayCart()</code> in the <code class="text-danger">script.js</code> code.
        You can modify it to your own preference.
        </p>
```

```
</div>


</div>
<!--container.//-->
<!-- Modal -->
<div class="modal fade" id="cart" tabindex="-1" role="dialog" aria-
    labelledby="exampleModalLabel" aria-hidden="true">
<div class="modal-dialog modal-lg" role="document">
<div class="modal-content">
        <form action="index.html" method="get" class="cart-form">
        <div class="modal-header">
                <h5 class="modal-title" id="exampleModalLabel"><i class="fas fa-
    shopping-cart text-black"></i> Shopping Cart</h5>
                <button type="button" class="close" data-dismiss="modal" aria-
    label="Close">
                <span aria-hidden="true">&times;</span>
                </button>
        </div>
        <div class="modal-body">
                <table class="show-cart table">

                </table>
                <div class="font-weight-bold">Total price: $<span class="total-cart text-
    danger"></span></div>
        </div>
        <div class="modal-footer">
                <button type="button" class="btn btn-danger" data-
    dismiss="modal">Close</button>
                <button type="submit" class="btn btn-success">Checkout</button>
        </div>
        </form>
</div>
</div>
</div>
<!-- partial -->
<script src='assets/js/jquery.min.js'></script>
<script  src="assets/js/script.js"></script>

<!-- Shopping Cart Add Toast -->
<div id="product-toast">
<div id="img">
<i class="fas fa-shopping-cart"></i>
    Cart
</div>
<div id="desc"><span id="item-name"></span> added to cart</div>
</div>

</body>
</html>
```

script.js

```
// ************************************************
// Shopping Cart API

// ************************************************
```

```
var shoppingCart = (function() {
 // =============================
 // Private methods and propeties
 // =============================
 cart = [];

 // Constructor
 function Item(product_id, name, price, pimage, count) {
  "use strict";
  this.product_id = product_id;
  this.name = name;
  this.price = price;
  this.pimage = pimage;
  this.count = count;
 }

 // Save cart
 function saveCart() {
  "use strict";
  sessionStorage.setItem('shoppingCart', JSON.stringify(cart));
 }

 // Load cart
 function loadCart() {
  "use strict";
  cart = JSON.parse(sessionStorage.getItem('shoppingCart'));
 }
 if (sessionStorage.getItem("shoppingCart") != null) {
  loadCart();
 }


 // =============================
 // Public methods and propeties
 // =============================
 var obj = {};

 // Add to cart
 obj.addItemToCart = function(product_id, name, price, pimage, count) {
  "use strict";
  for(var item in cart) {
    if(cart[item].product_id === product_id) {
      cart[item].count ++;
      saveCart();
      return;
    }
  }
  var item = new Item(product_id, name, price, pimage, count);
  cart.push(item);
  saveCart();
 }
 // Set count from item
 obj.setCountForItem = function(product_id, count) {
  "use strict";
  for(var i in cart) {
    if (cart[i].product_id === product_id) {
      cart[i].count = count;
      break;
```

```
    }
  }
};
// Remove item from cart
obj.removeItemFromCart = function(product_id) {
  "use strict";
  for(var item in cart) {
    if(cart[item].product_id === product_id) {
      cart[item].count --;
      if(cart[item].count === 0) {
        cart.splice(item, 1);
      }
      break;
    }
  }
  saveCart();
}

// Remove all items from cart
obj.removeItemFromCartAll = function(product_id) {
  "use strict";
  for(var item in cart) {
    if(cart[item].product_id === product_id) {
      cart.splice(item, 1);
      break;
    }
  }
  saveCart();
}

// Clear cart
obj.clearCart = function() {
  "use strict";
  cart = [];
  saveCart();
}

// Count cart
obj.totalCount = function() {
  "use strict";
  var totalCount = 0;
  for(var item in cart) {
    totalCount += cart[item].count;
  }
  return totalCount;
}

// Total cart
obj.totalCart = function() {
  "use strict";
  var totalCart = 0;
  for(var item in cart) {
    totalCart += cart[item].price * cart[item].count;
  }
  return Number(totalCart.toFixed(2));
}

// List cart
```

```
  obj.listCart = function() {
    var cartCopy = [];
    for(i in cart) {
      item = cart[i];
      itemCopy = {};
      for(p in item) {
        itemCopy[p] = item[p];
      }
      itemCopy.total = Number(item.price * item.count).toFixed(2);
      cartCopy.push(itemCopy)
    }
    return cartCopy;
  }

  // cart : Array
  // Item : Object/Class
  // addItemToCart : Function
  // removeItemFromCart : Function
  // removeItemFromCartAll : Function
  // clearCart : Function
  // countCart : Function
  // totalCart : Function
  // listCart : Function
  // saveCart : Function
  // loadCart : Function
  return obj;
})();


// ****************************************
// Triggers / Events
// ****************************************
// Add item
$('.add-to-cart').on("click", function(event){
  "use strict";
  event.preventDefault();
  var product_id = $(this).data('product_id');
  var name = $(this).data('name');
  var price = Number($(this).data('price'));
  var pimage = $(this).data('pimage');
  shoppingCart.addItemToCart(product_id, name, price, pimage, 1);
  displayCart();
});


// Clear items
$('.clear-cart').on("click",function() {
  "use strict";
  shoppingCart.clearCart();
  displayCart();
});

function displayCart() {
  "use strict";
  var cartArray = shoppingCart.listCart();
  var output = "";
  var total_order = 0;
  for(var i in cartArray) {
```

```
    total_order++;
    output += "<div class='col-12'>"
        + "<div class='row justify-content-center'>"
        + "<div class='col-5'>"
        + "<div class='row'>"
        + "<div class='col-9 text-left'>"
        + "<img src='" + cartArray[i].pimage + "' class='rounded-circle' width='50' height='50' />"
        + "<span class='text-info pl-3 font-weight-bold'>" + cartArray[i].name + "</span>"
        + "</div>"
        + "<div class='col-3'>"
        + "<span class='text-dark pl-2 mr-1'>(" + cartArray[i].price + ")</span>"
        + "</div>"
        + "</div>"
        + "</div>"
        + "<div class='col-4'>"
        + "<div class='input-group'><button class='minus-item input-group-addon btn btn-primary'
data-product_id=" + cartArray[i].product_id + ">-</button>"
        + "<input type='number' class='item-count form-control' data-product_id='" +
cartArray[i].product_id + "' value='" + cartArray[i].count + "'>"
        + "<button class='plus-item btn btn-primary input-group-addon' data-product_id=" +
cartArray[i].product_id + ">+</button></div>"
        + "</div>"
        + "<div class='col-3'>"
        + "<button class='delete-item btn btn-danger' data-product_id=" + cartArray[i].product_id +
">X</button>"
        + "<span class='text-dart pl-2'><kbd>" + cartArray[i].total + "</kbd></span>"
        + "</div>"
        + "</div>"
        + "</div>"
        + "<div class='row' id='hedden-fields'><input type = 'hidden' class='sr-only'
name='ProductID[]' id='ProductName' value='" + cartArray[i].product_id + "'>"
        + "<input type = 'hidden' class='sr-only' name='ProductName[]' id='ProductName' value='" +
cartArray[i].name + "'>"
        + "<input type = 'hidden' class='sr-only' name='ProductPrice[]' id='ProductPrice' value='" +
cartArray[i].price + "'>"
        + "<input type = 'hidden' class='sr-only' name='ProductQuantity[]' id='ProductQuantity'
value='" + cartArray[i].count + "'>"
        + "<input type = 'hidden' class='sr-only' name='TotalOrderedItems' id='TotalOrderedItems'
value='" + total_order + "'></div>"
        + "<hr/>" ;
  }

  $('.show-cart').html(output);
  $('.total-cart').html(shoppingCart.totalCart());
  $('.total-count').html(shoppingCart.totalCount());
}


// Delete item button

$('.show-cart').on("click", ".delete-item", function(event) {
  "use strict";
  var name = $(this).data('name');
  var product_id = $(this).data('product_id');
  shoppingCart.removeItemFromCartAll(product_id);
  displayCart();
})
```

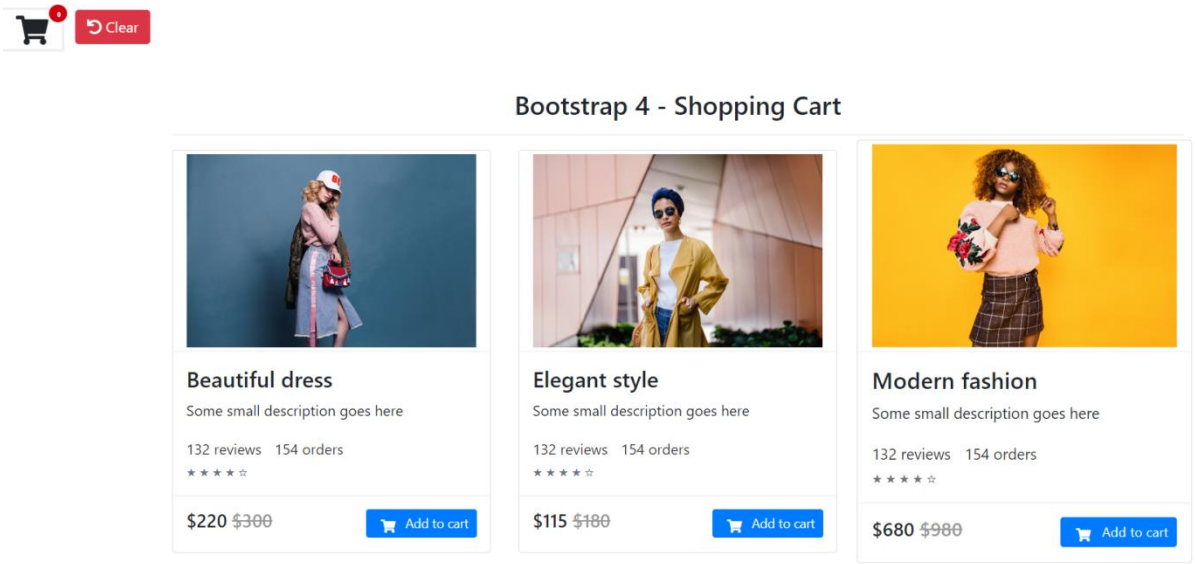
```
// -1
$('.show-cart').on("click", ".minus-item", function(event) {
  "use strict";
  var name = $(this).data('name');
  var product_id = $(this).data('product_id');
  shoppingCart.removeItemFromCart(product_id);
  displayCart();
})
// +1
$('.show-cart').on("click", ".plus-item", function(event) {
  "use strict";
  var name = $(this).data('name');
  var product_id = $(this).data('product_id');
  shoppingCart.addItemToCart(product_id);
  displayCart();
})

// Item count input
$('.show-cart').on("change", ".item-count", function(event) {
  "use strict";
  var name = $(this).data('name');
  var count = Number($(this).val());
  var product_id = $(this).data('product_id');
  shoppingCart.setCountForItem(product_id, count);
  displayCart();
});

displayCart();


// ==============================
// Cart toast display
// ==============================
$('.show-toast').on("click",function(){
  "use strict";
  const x = document.getElementById("product-toast");
  var item_name = $(this).data('name');
  $("#item-name").text(item_name);
  x.className = "show";
  setTimeout(function(){ x.className = x.className.replace("show", ""); }, 3000);
});



body {
  padding-top: 80px;
}

/* product cart design */
.show-cart li {
  display: flex;
}
.card {
  margin-bottom: 20px;
}
.card-img-top {
  width: 200px;
```

```
  height: 200px;
  align-self: center;
}

.card-product .img-wrap {
    border-radius: 3px 3px 0 0;
    overflow: hidden;
    position: relative;
    height: 220px;
    text-align: center;
}
.card-product .img-wrap img {
    max-height: 100%;
    max-width: 100%;
    object-fit: cover;
}
.card-product .info-wrap {
    overflow: hidden;
    padding: 15px;
    border-top: 1px solid #eee;
}
.card-product .bottom-wrap {
    padding: 15px;
    border-top: 1px solid #eee;
}

.label-rating { margin-right:10px;
    color: #333;
    display: inline-block;
    vertical-align: middle;
}

.card-product .price-old {
    color: #999;
}

.fa-2x-cart{
    font-size:24px
}

/* shopping cart counter */
.header{background:rgb(0, 178, 255);color:#fff;}
#lblCartCount {
    font-size: 12px;
    background: #ff0000;
    color: #fff;
    padding: 0 5px;
    vertical-align: top;
    margin-left: -10px;
}
.badge {
  padding-left: 9px;
  padding-right: 9px;
  -webkit-border-radius: 9px;
  -moz-border-radius: 9px;
  border-radius: 9px;
}
```

```
.label-warning[href],
.badge-warning[href] {
  background-color: #c67605;
}

/* shopping cart counter style */
i.fa-shopping-cart {
    width:2.2em;
    text-align:center;
    vertical-align:middle;
}

.fa-shopping-cart {
    margin-top: 0.2em;
    position: relative;
}

.badge {
    font-size: .25em;
    display: block;
    position: absolute;
    top: -.70em;
    right: -.70em;
    width: 2.5em;
    height: 2.4em;
    line-height: 2em;
    border-radius: 50%;
    text-align: center;

    color: #fff;
    background: rgba(207, 0, 15, 1);
}


/* product zoom on hover */
.zoom, .zoom-sm {
transition: transform .2s;
}

.zoom:hover {
transform: scale(1.05);
}

.zoom-sm:hover {
    transform: scale(1.03);
}


/* shopping cart toast */
#product-toast {
  visibility: hidden;
  max-width: 50px;
  height: 50px;
  margin: auto;
  background-color: #333;
  color: #fff;
  text-align: center;
  border-radius: 2px;
```

```
  position: fixed;
  z-index: 1;
  left: 0;right:0;
  bottom: 30px;
  font-size: 17px;
  white-space: nowrap;
}
#product-toast #img{
  width: 70px;
  height: 50px;

  float: left;

  padding-top: 16px;
  padding-bottom: 16px;

  box-sizing: border-box;


  background-color: #111;
  color: #fff;
}
#product-toast #desc{


  color: #fff;

  padding: 16px;

  overflow: hidden;
  white-space: nowrap;
}

#product-toast.show {
  visibility: visible;
  -webkit-animation: fadein 0.5s, expand 0.5s 0.5s,stay 3s 1s, shrink 0.5s 2s, fadeout 0.5s 2.5s;
  animation: fadein 0.5s, expand 0.5s 0.5s,stay 3s 1s, shrink 0.5s 4s, fadeout 0.5s 4.5s;
}

@-webkit-keyframes fadein {
  from {bottom: 0; opacity: 0;}
  to {bottom: 30px; opacity: 1;}
}

@keyframes fadein {
  from {bottom: 0; opacity: 0;}
  to {bottom: 30px; opacity: 1;}
}

@-webkit-keyframes expand {
  from {min-width: 50px}
  to {min-width: 350px}
}

@keyframes expand {
  from {min-width: 50px}
  to {min-width: 350px}
```

```
}
@-webkit-keyframes stay {
  from {min-width: 350px}
  to {min-width: 350px}
}

@keyframes stay {
  from {min-width: 350px}
  to {min-width: 350px}
}
@-webkit-keyframes shrink {
  from {min-width: 350px;}
  to {min-width: 50px;}
}

@keyframes shrink {
  from {min-width: 350px;}
  to {min-width: 50px;}
}

@-webkit-keyframes fadeout {
  from {bottom: 30px; opacity: 1;}
  to {bottom: 60px; opacity: 0;}
}

@keyframes fadeout {
  from {bottom: 30px; opacity: 1;}
  to {bottom: 60px; opacity: 0;}
}
```

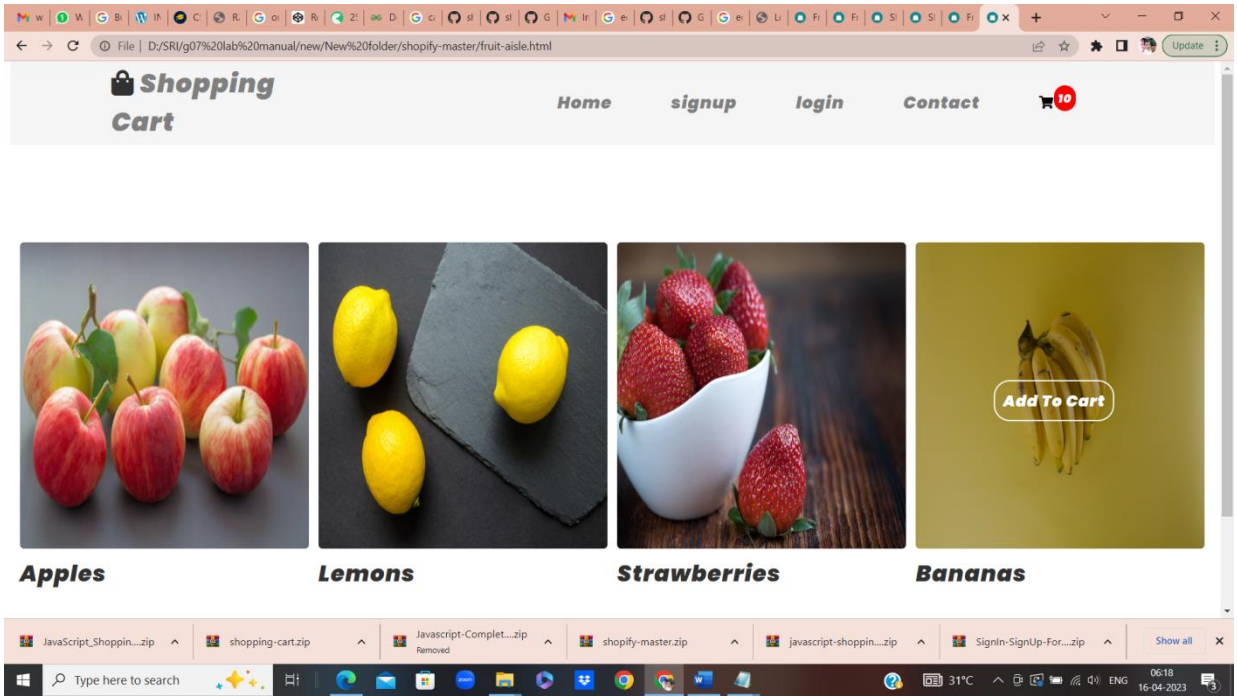


3. Use JavaScript for doing client – side validation of the pages implemented in experiment 1 andexperiment 2.

```
const top_items_count = document.querySelector('.items-count'),
    bottom_items_count = document.querySelector('.bottom-items-count'),
    exit = document.querySelector('.exit'),
```

```
    bucket = document.querySelector('.cart').style;

var items = document.querySelectorAll('.items');

top_items_count.innerHTML = count;
bottom_items_count.innerHTML = count;

function openBucket() {
    bucket.visibility = "visible";
    bucket.opacity = "1";
    bucket.zIndex = "9";
    bucket.transition = "all 0.5s";
}

exit.addEventListener('click', () => {
    bucket.visibility = "hidden";
    bucket.opacity = "0";
    bucket.zIndex = "-9";
    bucket.transition = "all 0.5s";
});

var food_cart = [];

() => {

    if (localStorage.food_cart) {
        food_cart = JSON.parse(localStorage.food_cart);
        showCart();
    }

}

var qty=1;

for (i = 0; i <= items.length - 1; i++) {
    var count=0;
    items[i].onclick = e => {
        count=count+1
        var itemName = e.target.dataset.item;
        var price = e.target.dataset.price;
        addToCart(itemName, price, qty);
        top_items_count.innerHTML = count;
        bottom_items_count.innerHTML = count;
    }
}

function addToCart(itemName, price,qty) {

    for (var i in food_cart) {
        if (food_cart[i].Product == itemName) {
            food_cart[i].Qty += qty;
            showCart();
            saveCart();
            return;
        }
    }

    var itemArray = {
```

```
        Product: itemName,
        Price: price,
        Qty: qty
    }

    food_cart.push(itemArray)
    saveCart();
    showCart();
}

function saveCart() {
    if (window.localStorage) {
        localStorage.food_cart = JSON.stringify(food_cart);
    }
}

function deleteItem(index) {
    food_cart.splice(index, 1);
    showCart();
    saveCart();
}

function showCart() {
    if (food_cart.length == 0) {
        var _ul = document.querySelector('#ul');
        _ul.innerHTML = "";
        return;
    }

    var _ul = document.querySelector('#ul');
    _ul.innerHTML = "";
    for (var i in food_cart) {
        var item = food_cart[i];
        var li = document.createElement("li")
        var row = `<span>${item.Product}</span><span onclick='deleteItem(" + i + ")'><i class='fas
fa-trash'></i></span><span>${item.Qty}</span><span>${item.Qty * item.Price}</span>`;
        li.innerHTML += row;
        var _ul = document.querySelector('#ul');
        _ul.appendChild(li);
    }
}
```



4. **Explore the features of ES6 like arrow functions, callbacks, promises, async/await. Implementan application for reading the weather information from openweathermap.org and display the information in the form of a graph on the web page.**

Arrow functions are introduced in ES6, which provides you a more accurate way to write the functions in JavaScript. They allow us to write smaller function syntax. Arrow functions make your code more readable and structured.

```
1. const functionName = (arg1, arg2, ?..) => {
2.     //body of the function
3. }
```

There are three parts to an Arrow Function or Lambda Function:

- **Parameters:** Any function may optionally have the parameters.
- **Fat arrow notation/lambda notation:** It is the notation for the **arrow (=>).**
- **Statements:** It represents the instruction set of the function.

```
// function expression

var myfun1 = function show() {
 console.log("It is a Function Expression");
}

// Anonymous function

var myfun2 = function () {
   console.log("It is an Anonymous Function");
  }
```

```
//Arrow function

var myfun3 = () => {
   console.log("It is an Arrow Function");
  };

myfun1();
myfun2();
myfun3();
```

Output

```
It is a Function Expression
It is an Anonymous Function
It is an Arrow Function
```

ES6 Promises

A Promise represents something that is eventually fulfilled. A Promise can either be rejected or resolved based on the operation outcome.

ES6 Promise is the easiest way to work with asynchronous programming in JavaScript. Asynchronous programming includes the running of processes individually from the main thread and notifies the main thread when it gets complete. Prior to the Promises, **Callbacks** were used to perform asynchronous programming.

Callback

A Callback is a way to handle the function execution after the completion of the execution of another function.

A Callback would be helpful in working with events. In Callback, a function can be passed as a parameter to another function.

Syntax

```
const Promise = new Promise((resolve,reject) => {....});
```

Example

```
let Promise = new Promise((resolve, reject)=>{
    let a = 3;
    if(a==3){
        resolve('Success');
    }
    else{
        reject('Failed');
```

```
    }
})
Promise.then((message)=>{
    console.log("It is then block. The message is: ?+ message)
}).catch((message)=>{
console.log("It is Catch block. The message is: ?+ message)
})
```

Output

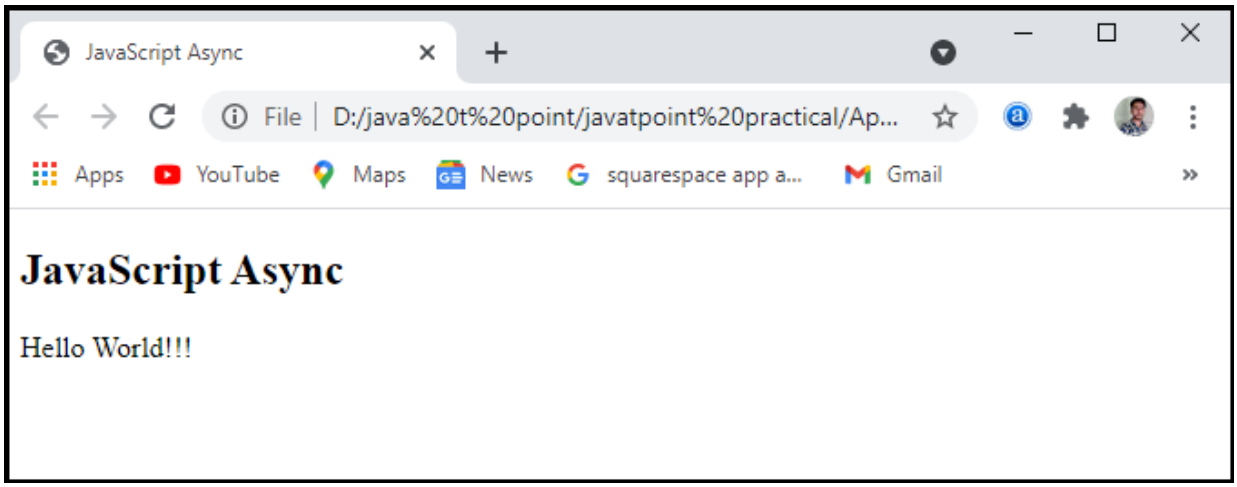
```
It is then block. The message is: Success
```

JavaScript Async/Await

JavaScript is always synchronous and single-threaded that provides the event loops. The event loops enable us to queue up an activity. This activity will not happen until the loops become available after the program that queued the action has completed the execution. However, our program contains a large number of functionalities, which causes our code to be asynchronous. The Async/Await functionality is one of them. Async/Await is an extension of promises that we get as language support.

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
<title>JavaScript Async</title>
</head>
<body>
    <h2>JavaScript Async</h2>
    <p id="main"></p>
<script>
function myDisplayer(some) {
  document.getElementById("main").innerHTML = some;
}
async function myfirstFunction() {
    return "Hello World!!!";
    }
myfirstFunction().then(
  function(value) {myDisplayer(value);},
  function(error) {myDisplayer(error);}
);
</script>
</body>
</html>
```

Output: After executing the above code, we will get the output shown below in the screenshot.



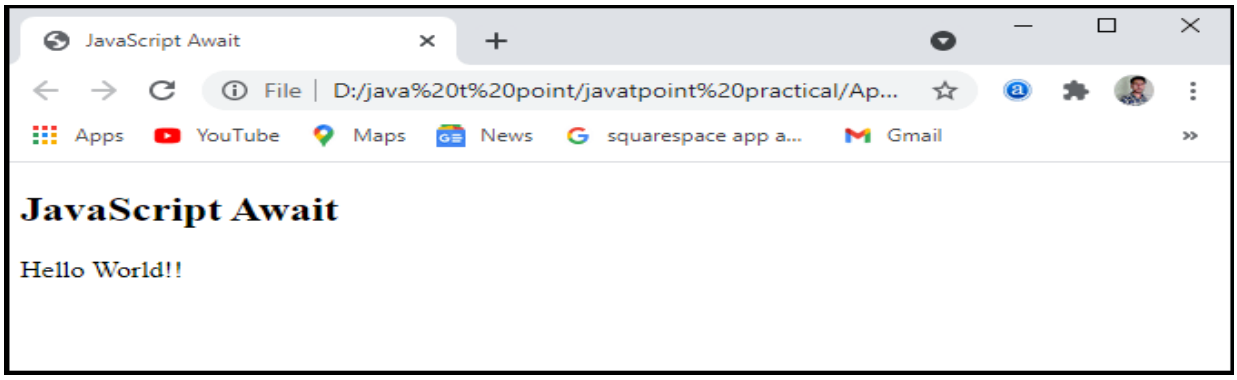
JavaScript Await

JavaScript Await function is used to wait for the promise. It could only be used inside the async block. It instructs the code to wait until the promise returns a response. It only delays the async block. Await is a simple command that instructs JavaScript to wait for an asynchronous action to complete before continuing with the feature. It's similar to a **"pause until done"** keyword. The await keyword is used to retrieve a value from a function where we will usually be used the **then()** function. Instead of calling after the asynchronous function, we'd use await to allocate a variable to the result and then use the result in the code as we will in the synchronous code.

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
<title>JavaScript Await</title>
</head>
<body>
    <h2>JavaScript Await</h2>
    <p id="main"></p>
<script>
async function myDisplay() {
  let myPromise = new Promise(function(myResolve, myReject) {
    myResolve("Hello World!!");
  });
  document.getElementById("main").innerHTML = await myPromise;
}
myDisplay();
</script>
</body>
```

20. **</html>**

Output: After executing this code, we will get the output as shown below in the screenshot:



Implementan application for reading the weather information from openweathermap.org and display the information in the form of a graph on the web page.

MainPage.html

```
<!DOCTYPE html>
<html>
<head>
<title>Weather Report</title>
<link rel="icon" href="favicon.png">
<link rel="stylesheet" href="PageStyle.css">
<link href=
'https://fonts.googleapis.com/css?family=Delius Swash Caps'
  rel='stylesheet'>
</head>
<body>
<p id="data" class="styleIt"></p>
<script src="JSPage.js"></script>
</body>
</html>
```

PageStyle.css

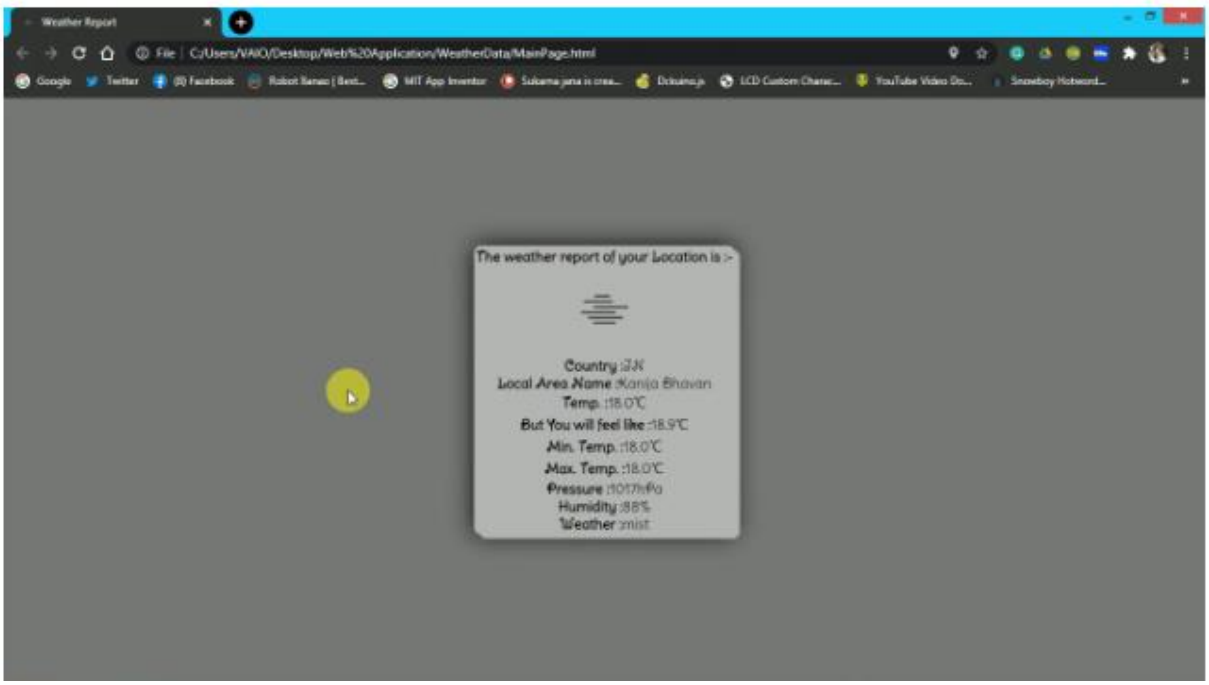
```
p.styleIt{
                background-color: rgb(182, 182, 182);
                border: 2px solid rgb(182, 182, 182);
                border-radius: 8px;
                text-align: center;
                box-shadow: 6px 5px 2px rgb(182, 182, 182), 0 0 25px rgb(0, 0, 0), 0 0
            5px rgb(182, 182, 182);
                font-family: 'Delius Swash Caps';
            }
            body{
                background:rgb(120, 120, 120);
                margin: 0;
```

```
            position: absolute;
            top: 50%;
            left: 50%;
            margin-right: -50%;
            transform: translate(-50%, -50%)
        }
```

JSPage.js

```
var data =
document.getElementById("dat
a");
                                var Latitude;
                                var Longitude;
                                var key = "------Put Your Own Key-----";
                                var url =
                                "http://api.openweathermap.org/data/2.5/weather?";
                                function getLocation(){
                                    if(navigator.geolocation){

                                navigator.geolocation.getCurrentPosition(showPosition);
                                    }
                                    else{
                                        data_of_Lat_Lon.innerHTML="Geolocation is not
                                supported by this browser. SORRY!";
                                    }
                                }
                                function showPosition(position){
                                    Latitude = position.coords.latitude;
                                    Longitude = position.coords.longitude;
                                    getData(Latitude,Longitude);
                                }
                                function getData(Lat,Lon){
                                    const readyToSent =
                                (url+"lat="+Lat+"&lon="+Lon+"&appid="+key);
                                    fetch(readyToSent)
                                    .then(response=>response.json())
                                    .then(data=>{
                                        console.log(data);
                                        fetchData(data)
                                    })
                                }
                                function fetchData(data){
                                    const icon =
                                "http://openweathermap.org/img/wn/"+data.weather[0].icon+"
                                @2x.png"
                                    document.getElementById("data").innerHTML =
```

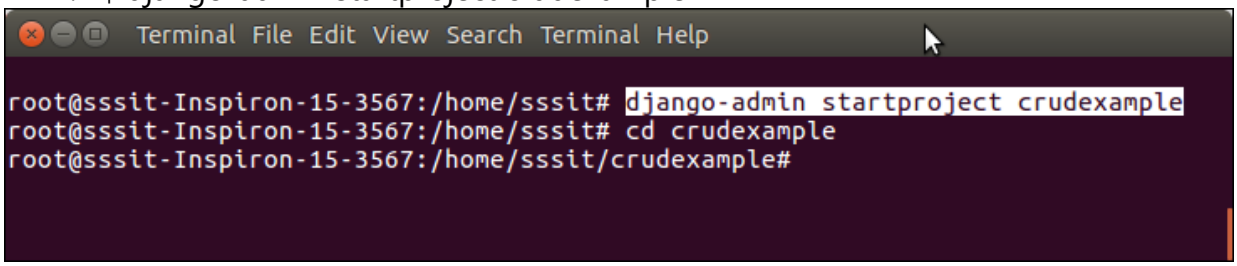
```
        "<b>The weather report of your Location is :-
</b><br>"+
        "<img src="+icon+"><br>"+
        "<b>Country :</b>"+data.sys.country+
        "<br><b>Local Area Name :</b>"+data.name+
        "<br><b>Temp. :</b>"+parseFloat((data.main.temp -
273.15)).toFixed(1)+"&#8451;"+
        "<br><b>But You will feel like
:</b>"+parseFloat((data.main.feels_like -
273.15)).toFixed(1)+"&#8451;"+
        "<br><b>Min. Temp.
:</b>"+parseFloat((data.main.temp_min -
273.15)).toFixed(1)+"&#8451;"+
        "<br><b>Max. Temp.
:</b>"+parseFloat((data.main.temp_max -
273.15)).toFixed(1)+"&#8451;"+
        "<br><b>Pressure :</b>"+data.main.pressure+"hPa"+
        "<br><b>Humidity :</b>"+data.main.humidity+"%"+
        "<br><b>Weather
:</b>"+data.weather[0].description+
        "<br>"
}
getLocation();
showPosition();
getData();
```



To create a Django application that performs CRUD operations, follow the following steps.

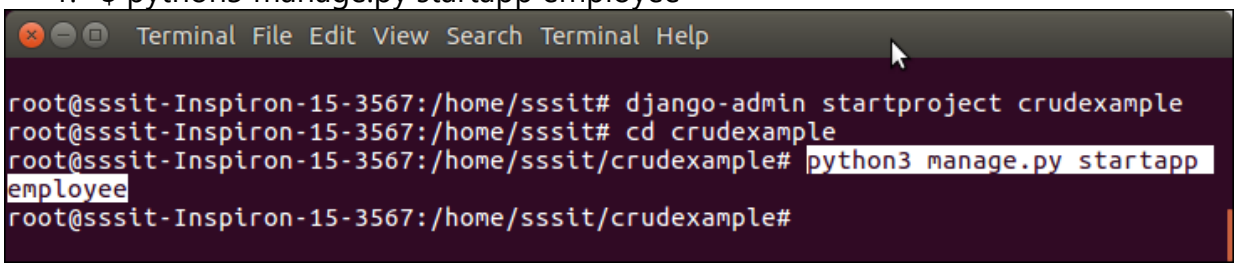
1. Create a Project

1. $ django-admin startproject crudexample



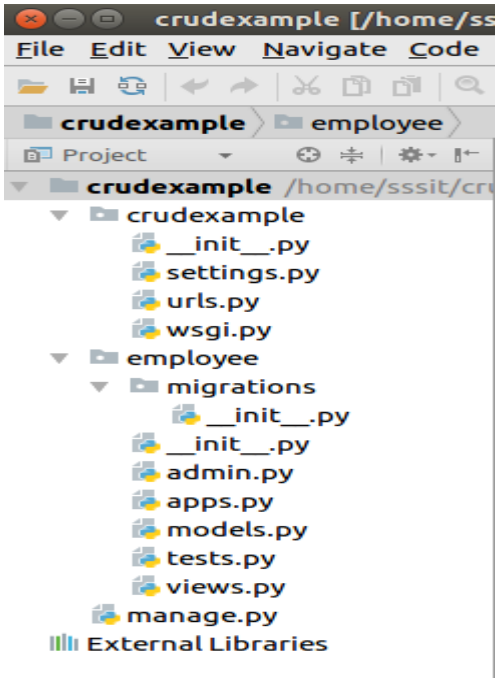
2. Create an App

1. $ python3 manage.py startapp employee



3. Project Structure

nitially, our project looks like this:



4. Database Setup

Create a database **djangodb** in mysql, and configure into the **settings.py** file of django project. See the example.

// settings.py

```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.mysql',
        'NAME': 'djangodb',
        'USER':'root',
        'PASSWORD':'mysql',
        'HOST':'localhost',
        'PORT':'3306'
    }
}
```

5. Create a Model

Put the following code into **models.py** file.

// models.py

```
from django.db import models
class Employee(models.Model):
    eid = models.CharField(max_length=20)
    ename = models.CharField(max_length=100)
    eemail = models.EmailField()
    econtact = models.CharField(max_length=15)
    class Meta:
        db_table = "employee"
```

6. Create a ModelForm

// forms.py

```
from django import forms
from employee.models import Employee
class EmployeeForm(forms.ModelForm):
    class Meta:
        model = Employee
        fields = "__all__"
```

7. Create View Functions

// views.py

```
from django.shortcuts import render, redirect
from employee.forms import EmployeeForm
from employee.models import Employee
# Create your views here.
def emp(request):
    if request.method == "POST":
        form = EmployeeForm(request.POST)
        if form.is_valid():
            try:
                form.save()
                return redirect('/show')
            except:
                pass
    else:
        form = EmployeeForm()
    return render(request,'index.html',{'form':form})
def show(request):
    employees = Employee.objects.all()
    return render(request,"show.html",{'employees':employees})
def edit(request, id):
    employee = Employee.objects.get(id=id)
    return render(request,'edit.html', {'employee':employee})
def update(request, id):
    employee = Employee.objects.get(id=id)
    form = EmployeeForm(request.POST, instance = employee)
    if form.is_valid():
        form.save()
        return redirect("/show")
    return render(request, 'edit.html', {'employee': employee})
def destroy(request, id):
    employee = Employee.objects.get(id=id)
    employee.delete()
    return redirect("/show")
```

8. Provide Routing

Provide URL patterns to map with views function.

// urls.py

```
from django.contrib import admin
from django.urls import path
from employee import views
urlpatterns = [
    path('admin/', admin.site.urls),
    path('emp', views.emp),
    path('show',views.show),
    path('edit/<int:id>', views.edit),
    path('update/<int:id>', views.update),
    path('delete/<int:id>', views.destroy),
]
```

9. Organize Templates

Create a **templates** folder inside the **employee** app and create three (index, edit, show) html files inside the directory. The code for each is given below

// index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Index</title>
    {% load staticfiles %}
    <link rel="stylesheet" href="{% static 'css/style.css' %}"/>
</head>
<body>
<form method="POST" class="post-form" action="/emp">
        {% csrf_token %}
    <div class="container">
<br>
    <div class="form-group row">
    <label class="col-sm-1 col-form-label"></label>
    <div class="col-sm-4">
    <h3>Enter Details</h3>
    </div>
  </div>
    <div class="form-group row">
    <label class="col-sm-2 col-form-label">Employee Id:</label>
    <div class="col-sm-4">
```

```
      {{ form.eid }}
    </div>
  </div>
  <div class="form-group row">
    <label class="col-sm-2 col-form-label">Employee Name:</label>
    <div class="col-sm-4">
      {{ form.ename }}
    </div>
  </div>
    <div class="form-group row">
    <label class="col-sm-2 col-form-label">Employee Email:</label>
    <div class="col-sm-4">
      {{ form.eemail }}
    </div>
  </div>
    <div class="form-group row">
    <label class="col-sm-2 col-form-label">Employee Contact:</label>
    <div class="col-sm-4">
      {{ form.econtact }}
    </div>
  </div>
    <div class="form-group row">
    <label class="col-sm-1 col-form-label"></label>
    <div class="col-sm-4">
    <button type="submit" class="btn btn-primary">Submit</button>
    </div>
  </div>
    </div>
</form>
</body>
</html>
```

// show.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Employee Records</title>
    {% load staticfiles %}
```

```
    <link rel="stylesheet" href="{% static 'css/style.css' %}"/>
</head>
<body>
<table class="table table-striped table-bordered table-sm">
    <thead class="thead-dark">
    <tr>
        <th>Employee ID</th>
        <th>Employee Name</th>
        <th>Employee Email</th>
        <th>Employee Contact</th>
        <th>Actions</th>
    </tr>
    </thead>
    <tbody>
{% for employee in employees %}
    <tr>
        <td>{{ employee.eid }}</td>
        <td>{{ employee.ename }}</td>
        <td>{{ employee.eemail }}</td>
        <td>{{ employee.econtact }}</td>
        <td>
            <a href="/edit/{{ employee.id }}"><span class="glyphicon glyphicon-pencil" >Edit</span></a>
            <a href="/delete/{{ employee.id }}">Delete</a>
        </td>
    </tr>
{% endfor %}
    </tbody>
</table>
<br>
<br>
<center><a href="/emp" class="btn btn-primary">Add New Record</a></center>
</body>
</html>
```

// edit.html

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

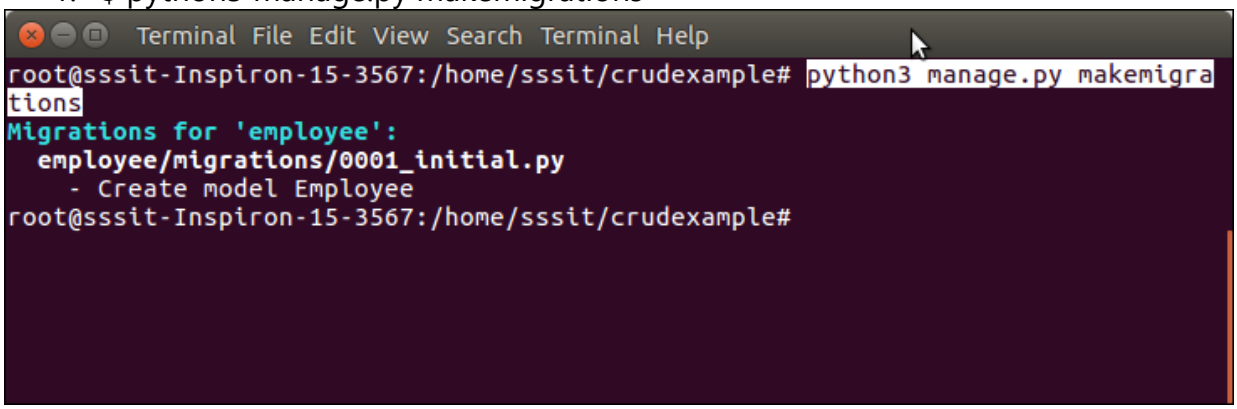
```
<head>
    <meta charset="UTF-8">
    <title>Index</title>
    {% load staticfiles %}
    <link rel="stylesheet" href="{% static 'css/style.css' %}"/>
</head>
<body>
<form method="POST" class="post-form" action="/update/{{employee.id}}">
        {% csrf_token %}
    <div class="container">
<br>
    <div class="form-group row">
    <label class="col-sm-1 col-form-label"></label>
    <div class="col-sm-4">
    <h3>Update Details</h3>
    </div>
  </div>
    <div class="form-group row">
    <label class="col-sm-2 col-form-label">Employee Id:</label>
    <div class="col-sm-4">
        <input type="text" name="eid" id="id_eid" required maxlength="20" value="{{ employee.eid }}"/>
    </div>
  </div>
  <div class="form-group row">
    <label class="col-sm-2 col-form-label">Employee Name:</label>
    <div class="col-sm-4">
        <input type="text" name="ename" id="id_ename" required maxlength="100" value="{{ employee.ename }}" />
    </div>
  </div>
    <div class="form-group row">
    <label class="col-sm-2 col-form-label">Employee Email:</label>
    <div class="col-sm-4">
        <input type="email" name="eemail" id="id_eemail" required maxlength="254" value="{{ employee.eemail }}" />
    </div>
  </div>
```

```
38.    <div class="form-group row">
39.    <label class="col-sm-2 col-form-label">Employee Contact:</label>
40.    <div class="col-sm-4">
41.        <input type="text" name="econtact" id="id_econtact" required maxlen
       gth="15" value="{{ employee.econtact }}" />
42.    </div>
43.  </div>
44.    <div class="form-group row">
45.    <label class="col-sm-1 col-form-label"></label>
46.    <div class="col-sm-4">
47.    <button type="submit" class="btn btn-success">Update</button>
48.    </div>
49.  </div>
50.    </div>
51. </form>
52. </body>
53. </html>
```

12. Create Migrations

Create migrations for the created model employee, use the following command.

```
1.  $ python3 manage.py makemigrations
```



After migrations, execute one more command to reflect the migration into the database. But before it, mention name of app (employee) in INSTALLED_APPS of settings.py file.

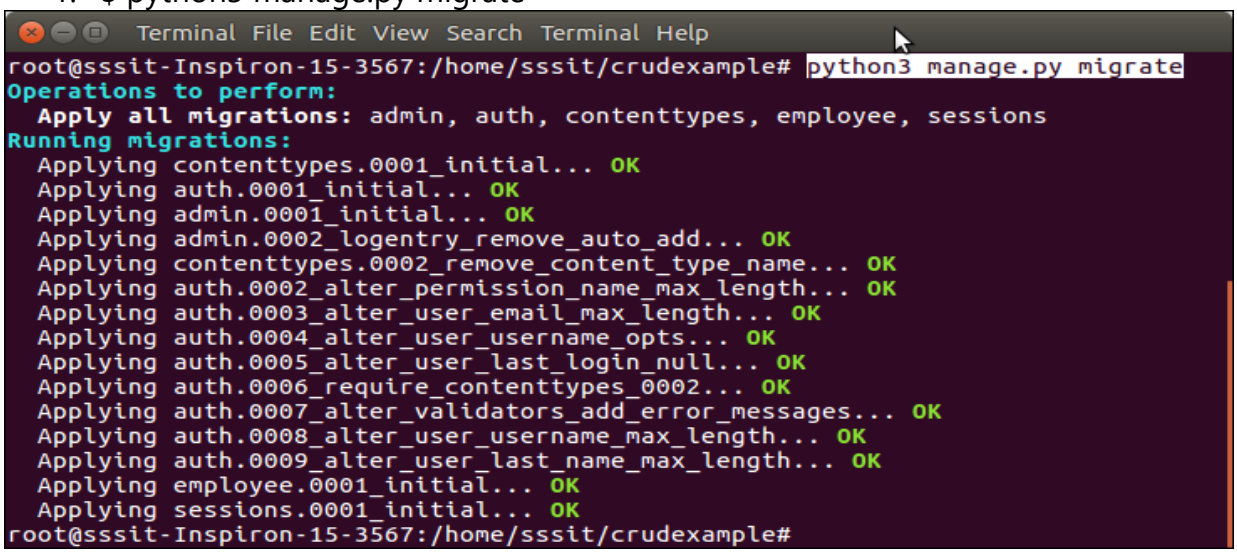
// settings.py

```
1.  INSTALLED_APPS = [
2.      'django.contrib.admin',
3.      'django.contrib.auth',
4.      'django.contrib.contenttypes',
```

```
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'employee'
]
```

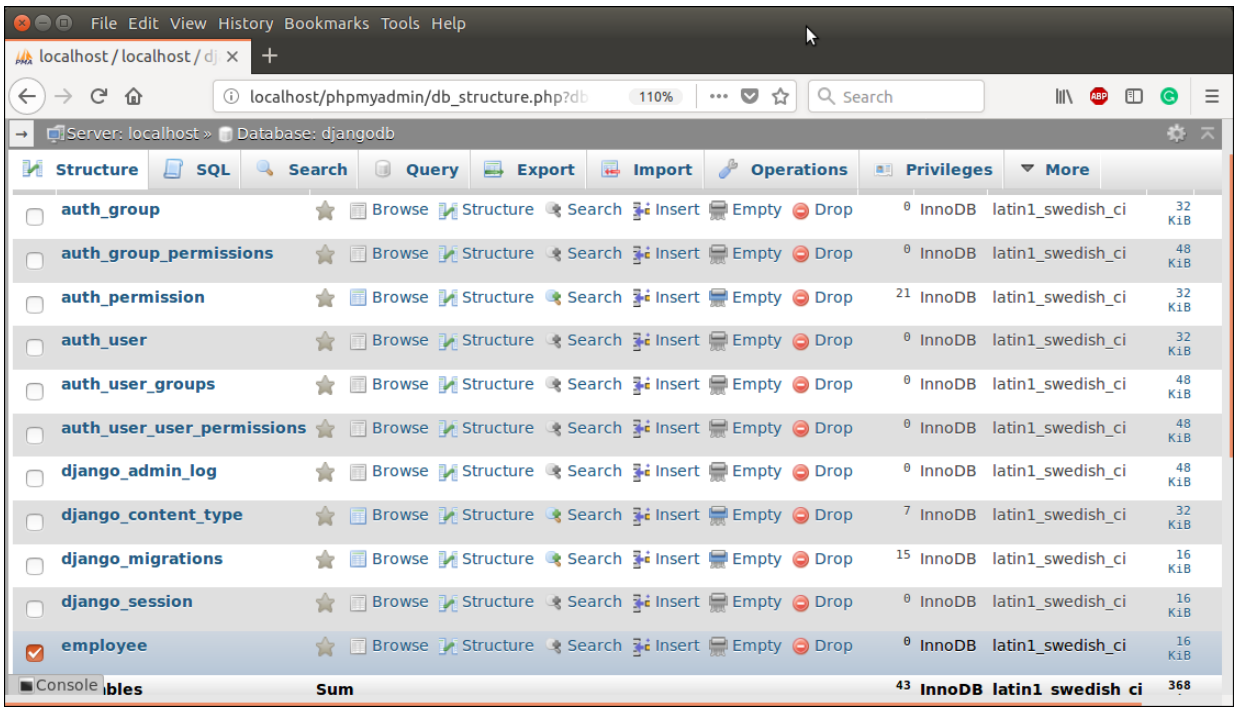
Run the command to migrate the migrations.

1. $ python3 manage.py migrate



Now, our application has successfully connected and created tables in database. It creates 10 default tables for handling project (session, authentication etc) and one table of our model that we created.

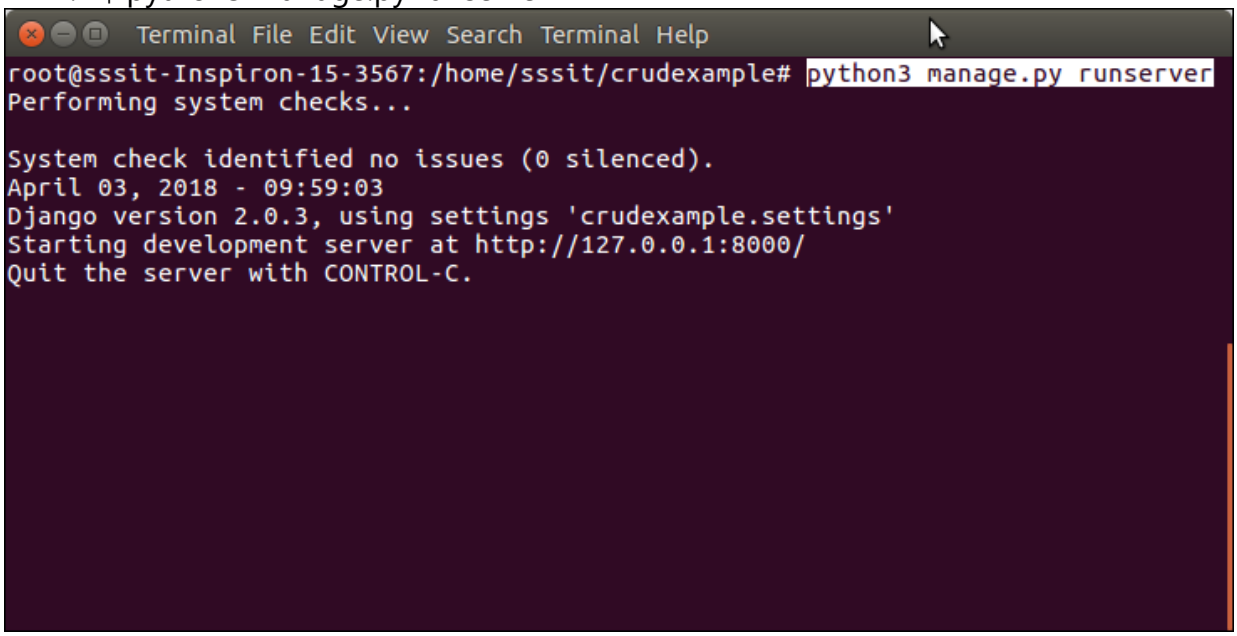
See list of tables created after migrate command.



Run Server

To run server use the following command.

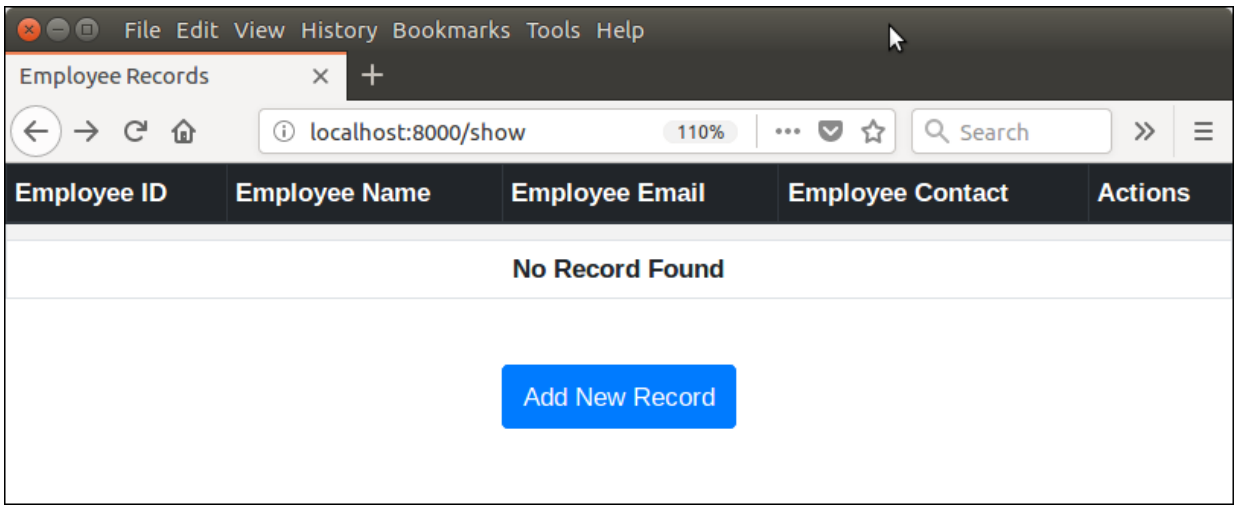
1. $ python3 manage.py runserver



Access to the Browser

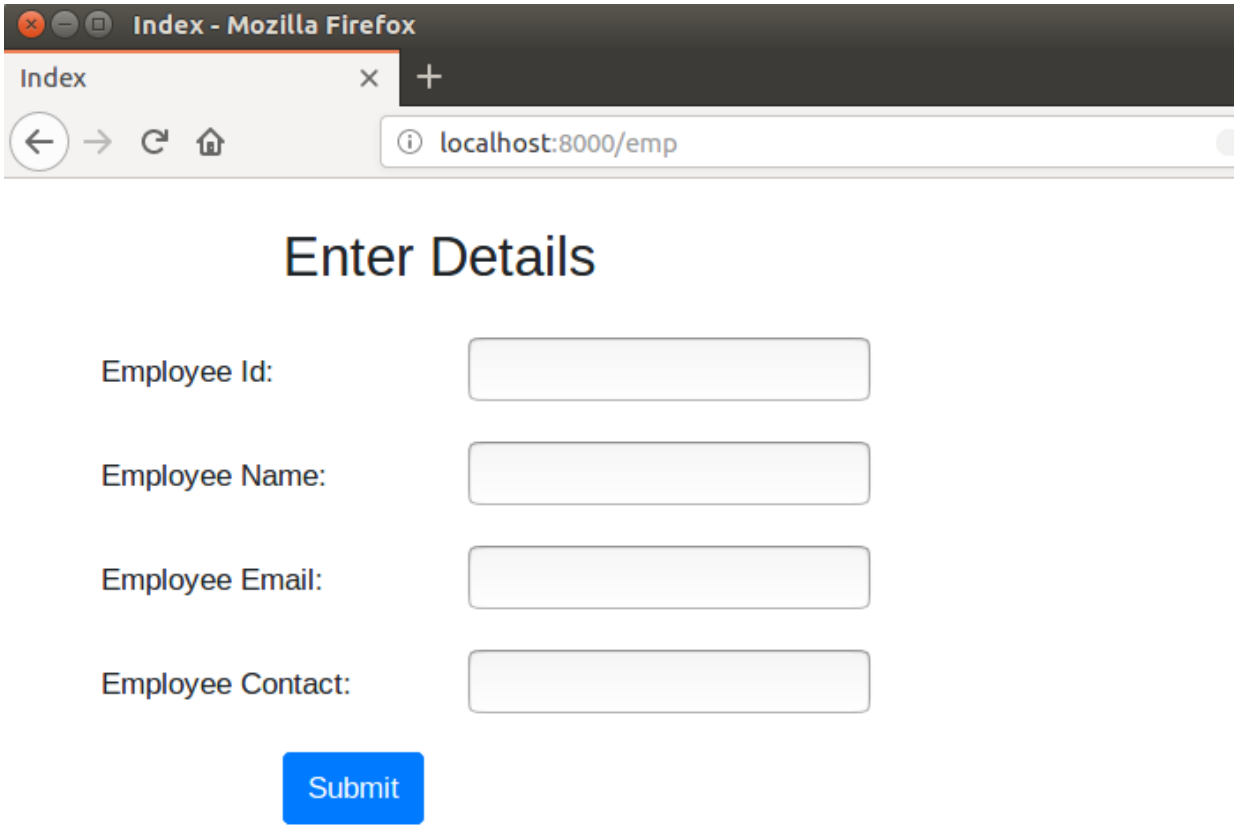
Access the application by entering **localhost:8000/show**, it will show all the available employee records.

Initially, there is no record. So, it shows no record message.

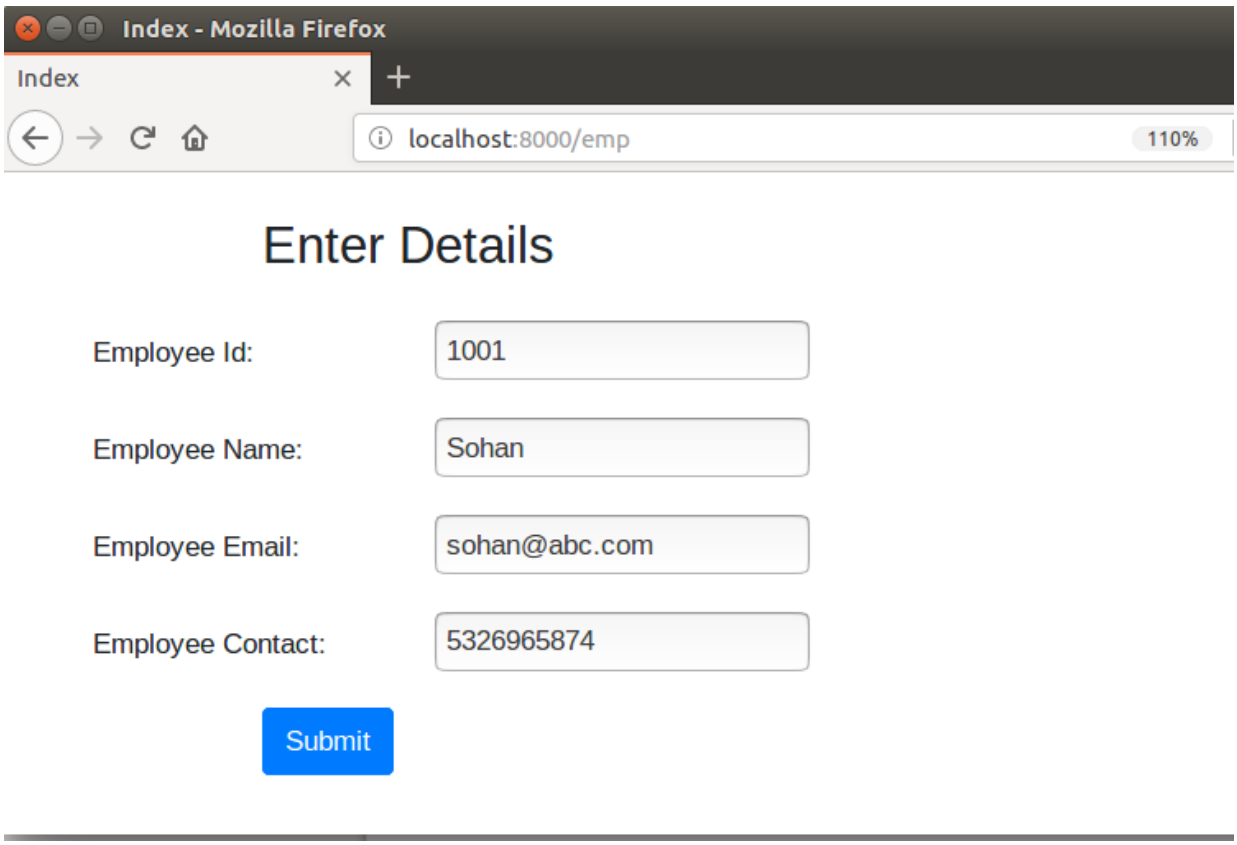


Adding Record

Click on the **Add New Record** button and fill the details. See the example.



Filling the details.



6.Create an xml for the bookstore. Validate the same using both DTD and XSD

< booksinfor.dtd >

```
<!ELEMENT books (book+)>
<!ELEMENT book (title,author,ISBN,publisher,edition,price)>
<!ELEMENT title (#PCDATA)>
<!ELEMENT author (#PCDATA)>
<!ELEMENT ISBN (#PCDATA)>
<!ELEMENT publisher (#PCDATA)>
<!ELEMENT edition (#PCDATA)>
<!ELEMENT price (#PCDATA)>
```

< week5.xml >

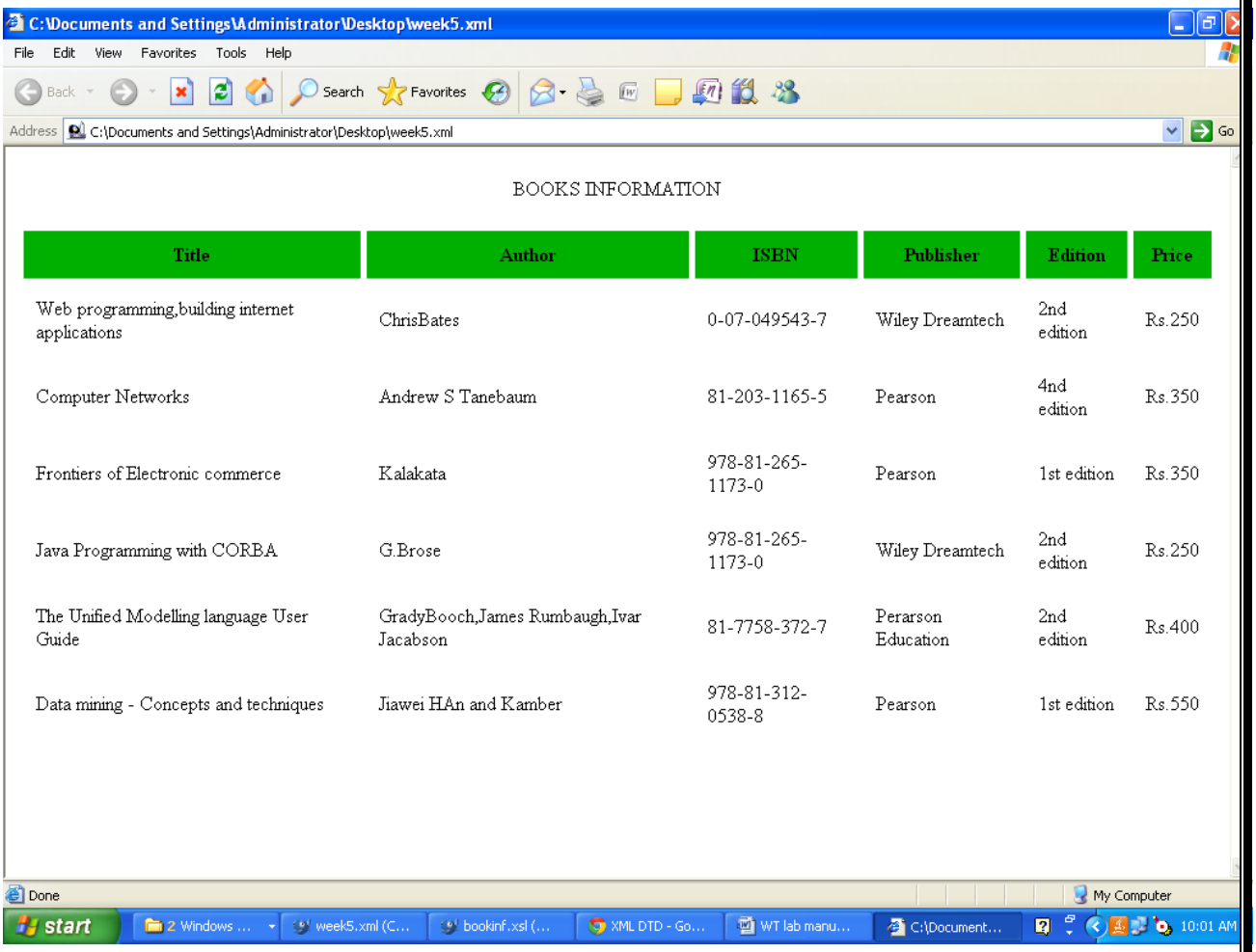
```
<?xml version="1.0"?>
<!DOCTYPE books SYSTEM "booksinfor.dtd">
<?xml:stylesheet type="text/xsl" href="bookinf.xsl"?>
<books>
<book>
<title>Web programming,building internet applications</title>
<author> ChrisBates</author>
<ISBN>0-07-049543-7</ISBN>
<publisher>Wiley Dreamtech</publisher>
<edition>2nd edition</edition>
<price>Rs.250</price>
</book>
<book>
<title>Computer Networks</title>
<author> Andrew S Tanebaum</author>
<ISBN>81-203-1165-5</ISBN>
<publisher>Pearson</publisher>
<edition>4nd edition</edition>
<price>Rs.350</price>
</book>
<book>
<title>Frontiers of Electronic commerce</title>
<author> Kalakata</author>
<ISBN>978-81-265-1173-0</ISBN>
<publisher>Pearson</publisher>
<edition>1st edition</edition>
<price>Rs.350</price>
</book>
<book>
<title>Java Programming with CORBA</title>
<author> G.Brose</author>
<ISBN>978-81-265-1173-0</ISBN>
<publisher>Wiley Dreamtech</publisher>
<edition>2nd edition</edition>
<price>Rs.250</price>
</book>
<book>
<title>The Unified Modelling language User Guide</title>
<author>GradyBooch,James Rumbaugh,Ivar Jacabson</author>
<ISBN>81-7758-372-7</ISBN>
<publisher>Perarson Education</publisher>
```

```
<edition>2nd edition</edition>
<price>Rs.400</price>
</book>
<book>
<title>Data mining - Concepts and techniques </title>
<author> Jiawei HAn and Kamber</author>
<ISBN>978-81-312-0538-8</ISBN>
<publisher>Pearson</publisher>
<edition>1st edition</edition>
<price>Rs.550</price>
</book>
</books>
```

< bookinf.xsl >

```
<?xml version="1.0"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
version="1.0">
<xsl:output method="html"/>
<xsl:template match="/">
<html>
<body>
<table align="center" cellspacing="5" cellpadding="10">
<caption>BOOKS INFORMATION</caption>
<tr bgcolor="light brown ">
<th>Title</th>
<th>Author</th>
<th>ISBN</th>
<th>Publisher</th>
<th>Edition</th>
<th>Price</th>
</tr>
<xsl:for-each select="books/book">
<tr>
<td><xsl:value-of select="title"/></td>
<td><xsl:value-of select="author"/></td>
<td><xsl:value-of select="ISBN"/></td>
<td><xsl:value-of select="publisher"/></td>
<td><xsl:value-of select="edition"/></td>
<td><xsl:value-of select="price"/></td>
</tr>
</xsl:for-each>
</table>
</body>
</html>
</xsl:template>
</xsl:stylesheet>
```

Output:



7. **Design a controller with servlet that provides the interaction with application developed inexperiment 1 and the database created in experiment 5.**

Mysql database

Procedure:
Reg.html:

```
<html>
<head>
<title> validation </title>
</head>
<body bgcolor="magenta">
<form action="Register.jsp" method="post">
<h1 align="center">REGISTRATION FORM</h1>
<table border="0">
<tr>
<td>Name:</td>
<td><input type="text" name="t1" min length="6"></td>
</tr>
<tr>
```

```
<td>Password:</td>
<td><input type="password" name="t2"></td>
</tr>
<tr>
<td>Phone number:</td>
<td><input type="text" name="t3"></td>
</tr>
<tr>
<td>E-mail id:</td>
<td><input type="text" name="t4"></td>
</tr>
<tr>
<td><input type="submit" value="submit"></td>
<td><input type="reset" value="cancel"></td>
</tr>
</table>
</form>
</body>
</html>
```

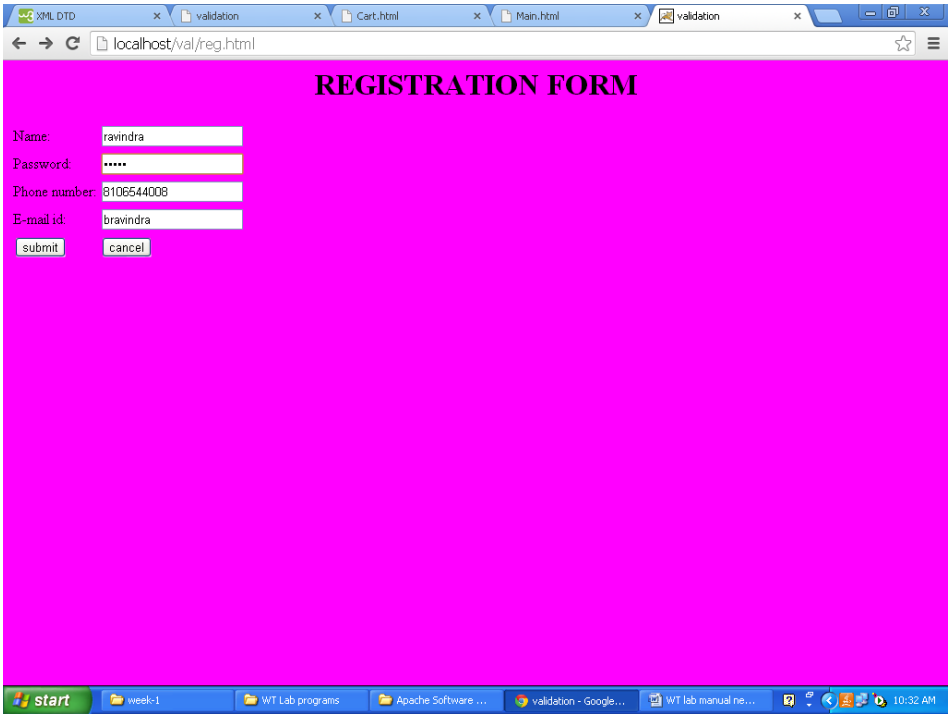
Register.jsp:

```
<%@ page language="java" import="java.sql.*,javax.servlet.*" %>
<html>
<form >
<%
String v1,v2,v3,v4,str;
v1=request.getParameter("t1");
v2=request.getParameter("t2");
v3=request.getParameter("t3");
v4=request.getParameter("t4");
try{
Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
Connection cn=DriverManager.getConnection("jdbc:odbc:oradsn","system","ravindra");
Statement st=cn.createStatement();
st.executeUpdate("insert into register values('"+v1+"','"+v2+"','"+v3+"','"+v4+"')");
if(cn!=null)
{
%>
</form>
Registration Successful
<%
}
else
{
out.println("Registration failed");
}
st.close();
cn.close();
}catch(Exception e) { out.println(" Registration failed");
%>
<P><a href = "reg.html" target =f2 ><B> Back<B></a>
<%
}
%>
</body></html>
```

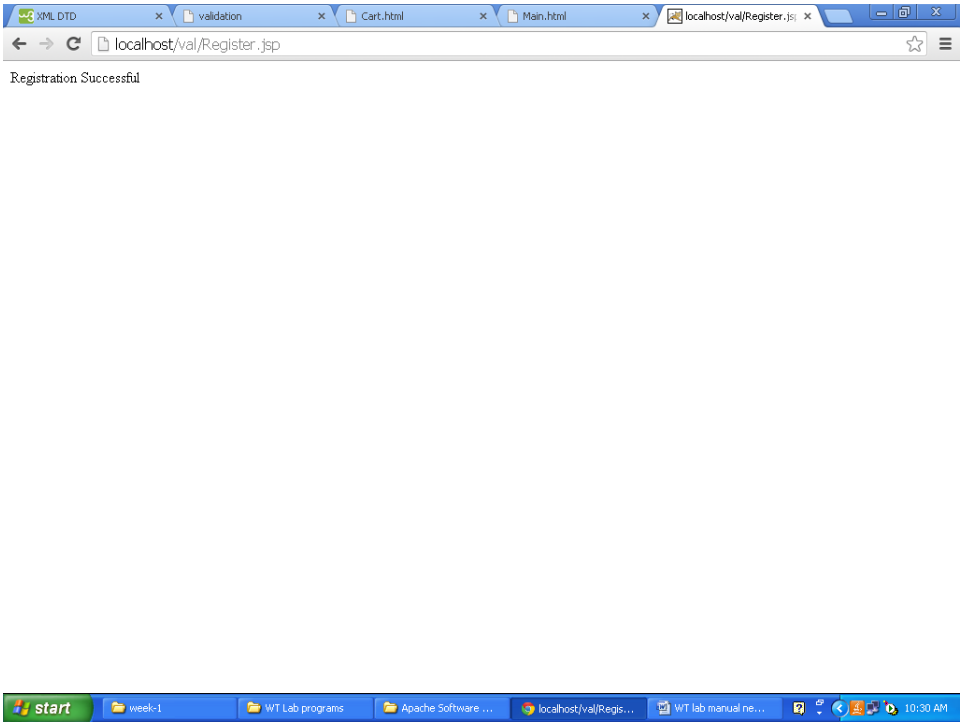
Execution:

Create a table with name register with name (varchar2 (10)), password (varchar2 (10)), Phone(number (10)) ,Email-ID (varchar2(10)).
Create the Data source name.
Start->settings->control panel->Administrative Tools->Data Sources.
Under SystemDSN add Microsoft ODBC for Oracle.
Set Data Source name to oradsn.

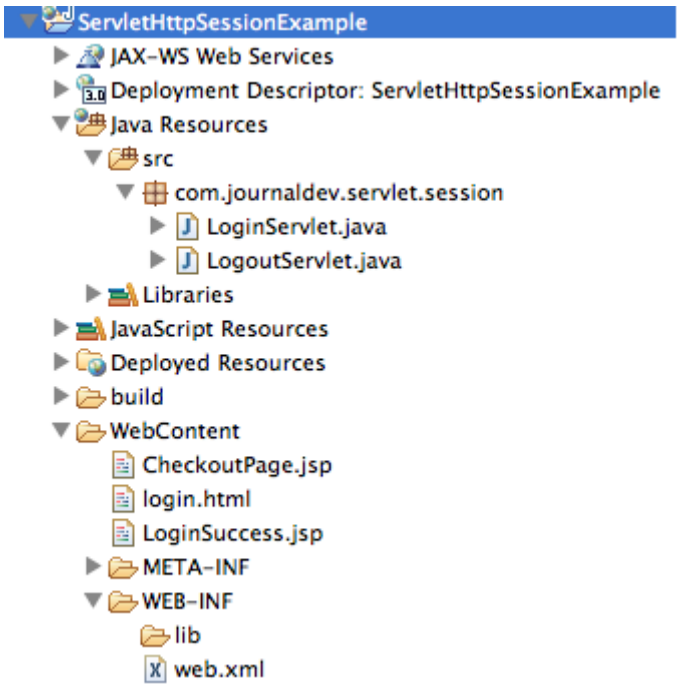
Output:1



Output:2



8. Maintaining the transactional history of any user is very important. Explore the various sessiontracking mechanism (Cookies, HTTP Session)




```
package com.journaldev.servlet.session;

import java.io.IOException;
import java.io.PrintWriter;

import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.Cookie;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;

/**
 * Servlet implementation class LoginServlet
 */
@WebServlet("/LoginServlet")
public class LoginServlet extends HttpServlet {
	private static final long serialVersionUID = 1L;
	private final String userID = "admin";
	private final String password = "password";

	protected void doPost(HttpServletRequest request,
			HttpServletResponse response) throws ServletException,
IOException {

		// get request parameters for userID and password
		String user = request.getParameter("user");
		String pwd = request.getParameter("pwd");

		if(userID.equals(user) && password.equals(pwd)){
			HttpSession session = request.getSession();
			session.setAttribute("user", "Pankaj");
			//setting session to expiry in 30 mins
```

```
                        session.setMaxInactiveInterval(30*60);
                        Cookie userName = new Cookie("user", user);
                        userName.setMaxAge(30*60);
                        response.addCookie(userName);
                        response.sendRedirect("LoginSuccess.jsp");
                }else{
                        RequestDispatcher rd =
getServletContext().getRequestDispatcher("/login.html");
                        PrintWriter out= response.getWriter();
                        out.println("<font color=red>Either user name or password is
wrong.</font>");
                        rd.include(request, response);
                }

        }

}
```

LoginSuccess.jsp code is given below.

```
<%@ page language="java" contentType="text/html; charset=US-ASCII"
   pageEncoding="US-ASCII"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"https://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=US-ASCII">
<title>Login Success Page</title>
</head>
<body>
<%
//allow access only if session exists
String user = null;
if(session.getAttribute("user") == null){
        response.sendRedirect("login.html");
}else user = (String) session.getAttribute("user");
String userName = null;
String sessionID = null;
Cookie[] cookies = request.getCookies();
if(cookies !=null){
for(Cookie cookie : cookies){
        if(cookie.getName().equals("user")) userName = cookie.getValue();
        if(cookie.getName().equals("JSESSIONID")) sessionID = cookie.getValue();
}
}
%>
<h3>Hi <%=userName %>, Login successful. Your Session ID=<%=sessionID %></h3>
<br>
User=<%=user %>
<br>
<a href="CheckoutPage.jsp">Checkout Page</a>
<form action="LogoutServlet" method="post">
<input type="submit" value="Logout" >
</form>
</body>
```

```
</html>
```

When a JSP resource is used, container automatically creates a session for it, so we can't check if session is null to make sure if user has come through login page, so we are using session attribute to validate request. CheckoutPage.jsp is another page and it's code is given below.

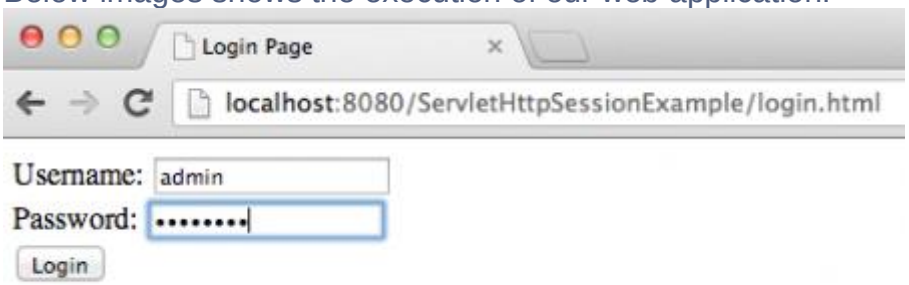
```
<%@ page language="java" contentType="text/html; charset=US-ASCII"
   pageEncoding="US-ASCII"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"https://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=US-ASCII">
<title>Login Success Page</title>
</head>
<body>
<%
//allow access only if session exists
if(session.getAttribute("user") == null){
        response.sendRedirect("login.html");
}
String userName = null;
String sessionID = null;
Cookie[] cookies = request.getCookies();
if(cookies !=null){
for(Cookie cookie : cookies){
        if(cookie.getName().equals("user")) userName = cookie.getValue();
}
}
%>
<h3>Hi <%=userName %>, do the checkout.</h3>
<br>
<form action="LogoutServlet" method="post">
<input type="submit" value="Logout" >
</form>
</body>
</html>
```

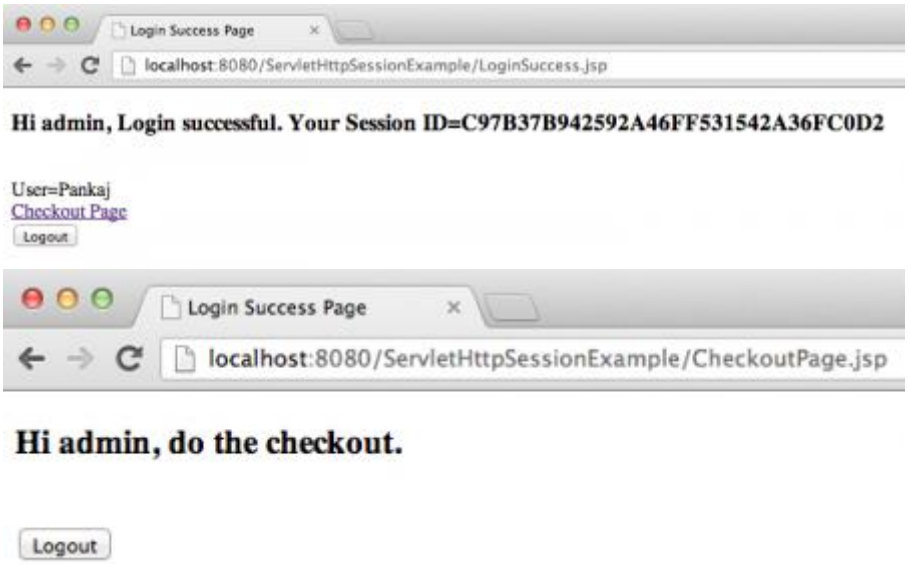
Our LogoutServlet code is given below.

```
package com.journaldev.servlet.session;

import java.io.IOException;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.Cookie;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;

/**
 * Servlet implementation class LogoutServlet
 */
@WebServlet("/LogoutServlet")
public class LogoutServlet extends HttpServlet {
```

```
        private static final long serialVersionUID = 1L;

    protected void doPost(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
        response.setContentType("text/html");
        Cookie[] cookies = request.getCookies();
        if(cookies != null){
        for(Cookie cookie : cookies){
                if(cookie.getName().equals("JSESSIONID")){
                        System.out.println("JSESSIONID="+cookie.getValue());
                        break;
                }
        }
        }
        //invalidate the session if exists
        HttpSession session = request.getSession(false);
        System.out.println("User="+session.getAttribute("user"));
        if(session != null){
                session.invalidate();
        }
        response.sendRedirect("login.html");
    }

}
```

1. Below images shows the execution of our web application.





9. **Create a custom server using http module and explore the other modules of Node JS like OS,path, event.**

The http.createServer() method includes request and response parameters which is supplied by Node.js. The request object can be used to get information about the current HTTP request e.g., url, request header, and data. The response object can be used to send a response for a current HTTP request.

The following example demonstrates handling HTTP request and response in Node.js.

```
var http = require('http'); // Import Node.js core module

var server = http.createServer(function (req, res) {   //create web
server
if (req.url == '/') { //check the URL of the current request

// set response header
        res.writeHead(200, { 'Content-Type': 'text/html' });

// set response content
        res.write('<html><body><p>This            is            home
Page.</p></body></html>');
        res.end();

    }
elseif (req.url == "/student") {

        res.writeHead(200, { 'Content-Type': 'text/html' });
        res.write('<html><body><p>This          is          student
Page.</p></body></html>');
        res.end();
```

```
    }
elseif (req.url == "/admin") {

        res.writeHead(200, { 'Content-Type': 'text/html' });
        res.write('<html><body><p>This is admin
Page.</p></body></html>');
        res.end();

    }
else
        res.end('Invalid Request!');

});

server.listen(5000); //6 - listen for any incoming requests

console.log('Node.js web server at port 5000 is running..')
```

Now, run the above web server as shown below.

C:\> node server.js

Node.js web server at port 5000 is running..

To test it, you can use the command-line program curl, which most Mac and Linux machines have pre-installed.

curl -i http://localhost:5000

You should see the following response.

HTTP/1.1 200 OK

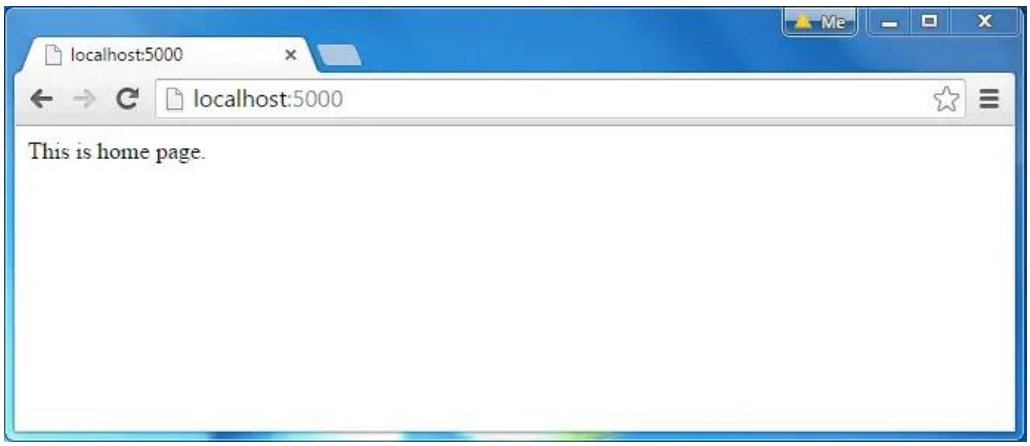
Content-Type: text/plain

Date: Tue, 8 Sep 2015 03:05:08 GMT

Connection: keep-alive

This is home page.

For Windows users, point your browser to *http://localhost:5000* and see the following result.



Node.js Web Server Response

10. Develop an express web application that can interact with REST API to perform 14 CRUDoperations on student data. (Use Postman)

#1) Installations.

#2) Create new Node.js Project with Express.js

#3) CRUD Operations and HTTP methods.

#4) Testing API with Postman.

Installation Required

- Node.js Or NPM (Node Package Manager)
- VS Code (Optional) — A code editor.

To check whether the Node.js is already installed on your computer, open your terminal or CMD and run `node -v` command. If you see your Node.js version means it's installed.

Otherwise go to these links and install:
→ Click here to download and install Node.js (You should choose

LTS version).
→ Click here to download VS Code

Express application generator:

To quickly create an application skeleton. You may use this application generator tool (`express-generator`). The application generator uses `npx` command (available in Node.js newer versions).
→ Click here to know more about Express application generator.

We will not use `express-generator` in this article, instead we will create everything by ourself to avoid extra files generated from generator and to understand in depth.

#2) Create New Project (using Node.js with Express.js)

Create a new folder (at the place you want to keep your project).
Name that folder: `node-ex-api`

Create two files inside `node-ex-api` folder:
→ `package.json` file.
→ `server.js` file.

Open up and update your `node-ex-api/package.json` file with below code:

```
{
    "name": "node-ex-api",
    "version": "1.0.0",
    "scripts": {
        "start": "node server.js"
    },
    "dependencies": {
        "express": "^4.17.1"
    }
}
```

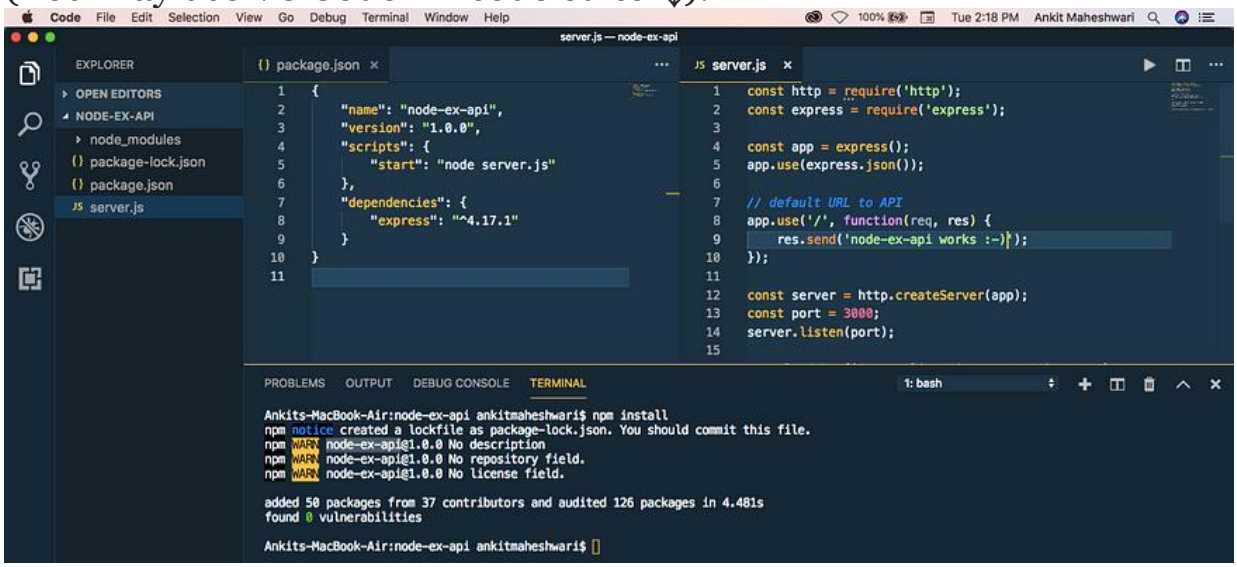
Open up and update your `node-ex-api/server.js` file with below code:

```
const http = require('http');
const express = require('express');

const app = express();
app.use(express.json());

// default URL to API
app.use('/', function(req, res) {
    res.send('node-ex-api works :-)');
});

const server = http.createServer(app);
const port = 3000;
server.listen(port);

console.debug('Server listening on port ' + port);
```

After creating above two files, open your terminal in the `"node-ex-api"` folder and run this command:

```
npm install
```

This command ↑ will install the dependencies defined in `"package.json"` file.
(You may use VS Code - A code editor↓).



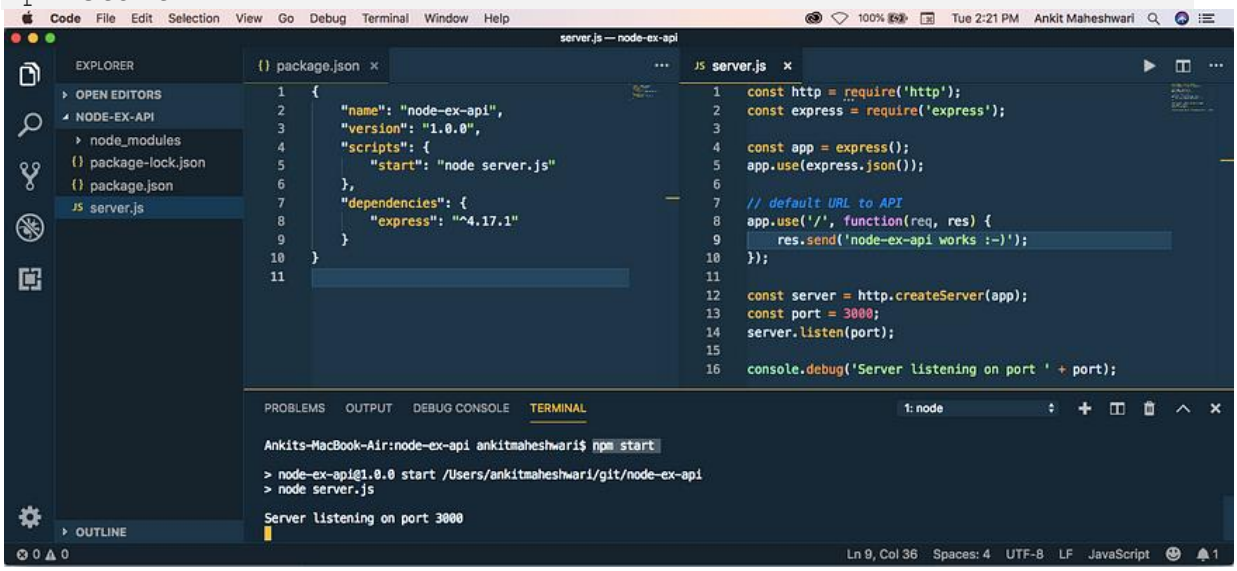
After dependency installation this will create `"node_modules"` folder at the root of the `"node-ex-api"` folder.

Run Project

We have just created a Node-Express Project 😍 Let's start a server.

To start a server run this command:

```
npm start
```



To test this API — Open your web browser and enter this URL → localhost:3000

The localhost:3000 is calling default Server Path. Which returns simple string "node-ex-api works :-)"

CRUD Operations and HTTP methods.

Yet we have a list of items (localhost:3000/items). Next task is to manage this list. To do so we need to have CRUD operations **C**reate, **R**ead, **U**pdate and **D** elete over the list of items.

For **C**reate we will create new end-point `router.post('/', .......)` → localhost:3000/items (With post request)

For **R**ead we already have two end-points:
→ localhost:3000/items (Returns all objects)
→ localhost:3000/items/1 (Returns single object of id=1)

For **U**pdate we will create new end-point `router.put('/', .......)`
→ localhost:3000/items (With put request)

For **D**elete we will create new end-point `router.delete('/', .......)`
→ localhost:3000/items (With delete request)

Now, Open up and update your `node-ex-api/routes/items.js` file with below code: (HTTP methods ↓↓)

Important explanation about above ↑ code:

→ We worked on static array of JSON objects named: `data`.
→ All HTTP methods GET, POST, PUT and DELETE are just manipulating this JSON array.
→ Feel free to use any database instead of using local static array. The database code can be written inside these HTTP methods to do operations like GET, POST, PUT and DELETE.

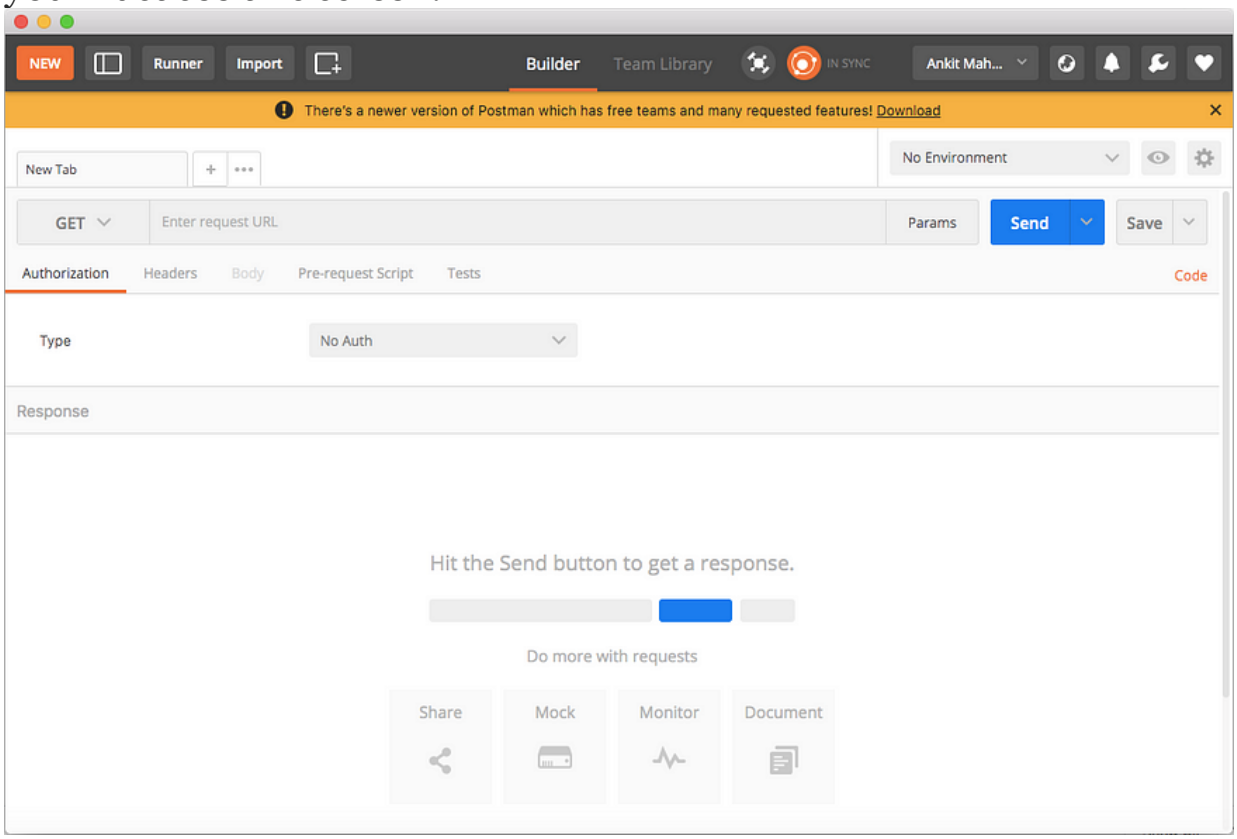
Testing API with Postman.

We have tested the GET methods of our API in our web browser (check #4) and seen responses. But we can't test POST, PUT and DELETE http methods in web browser. To test those methods we use Postman or you may use another http utility to call APIs.

Here we use Postman. So before start click here and install Postman.

Now, run `npm start` if your server is not running.

After Postman installation skip start window (if comes), then login with Google or email/password (whichever you prefer) and finally you must see this screen:



11.**For the above application create authorized end points using JWT (JSON Web Token).**

Create API endpoints

Our default endpoint returns string — we see that above. Now we'll create another API endpoint, that is another URL which returns some useful data.

Before proceed, let's do some more exercise:

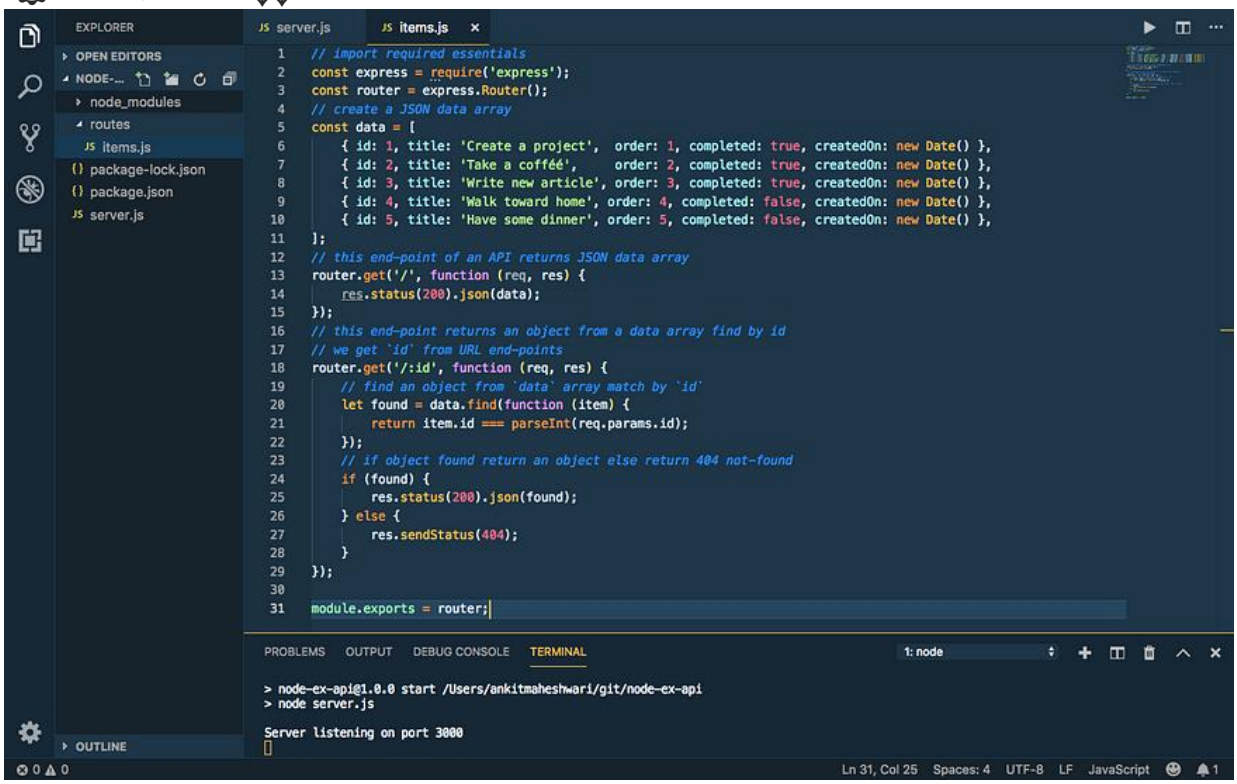
→ Create a folder inside the project root with name `"routes"`.

→ Then create a file inside this `"routes"` folder with name `"items.js"`.

Next, Open up and update your `node-ex-api/routes/items.js` file with below code:

```
// import required essentials
const express = require('express');
// create new router
const router = express.Router();

// create a JSON data array
let data = [
    { id: 1, title: 'Create a project',  order: 1, completed:
true, createdOn: new Date() },
    { id: 2, title: 'Take a cofféé',     order: 2, completed:
true, createdOn: new Date() },
    { id: 3, title: 'Write new article', order: 3, completed:
true, createdOn: new Date() },
    { id: 4, title: 'Walk toward home', order: 4, completed:
false, createdOn: new Date() },
    { id: 5, title: 'Have some dinner', order: 5, completed:
false, createdOn: new Date() },
];

// this end-point of an API returns JSON data array
router.get('/', function (req, res) {
    res.status(200).json(data);
});

// this end-point returns an object from a data array find by
id
// we get `id` from URL end-points
router.get('/:id', function (req, res) {
    // find an object from `data` array match by `id`
    let found = data.find(function (item) {
        return item.id === parseInt(req.params.id);
    });
    // if object found return an object else return 404 not-
found
    if (found) {
        res.status(200).json(found);
    } else {
        res.sendStatus(404);
    }
});

module.exports = router;
```

See below ↓↓



Register API endpoints

Let's register it in the `"server.js"` file to make use of new endpoints.

Do not forget to install CORS.
Open your terminal in the `"node-ex-api"` folder and run this command:

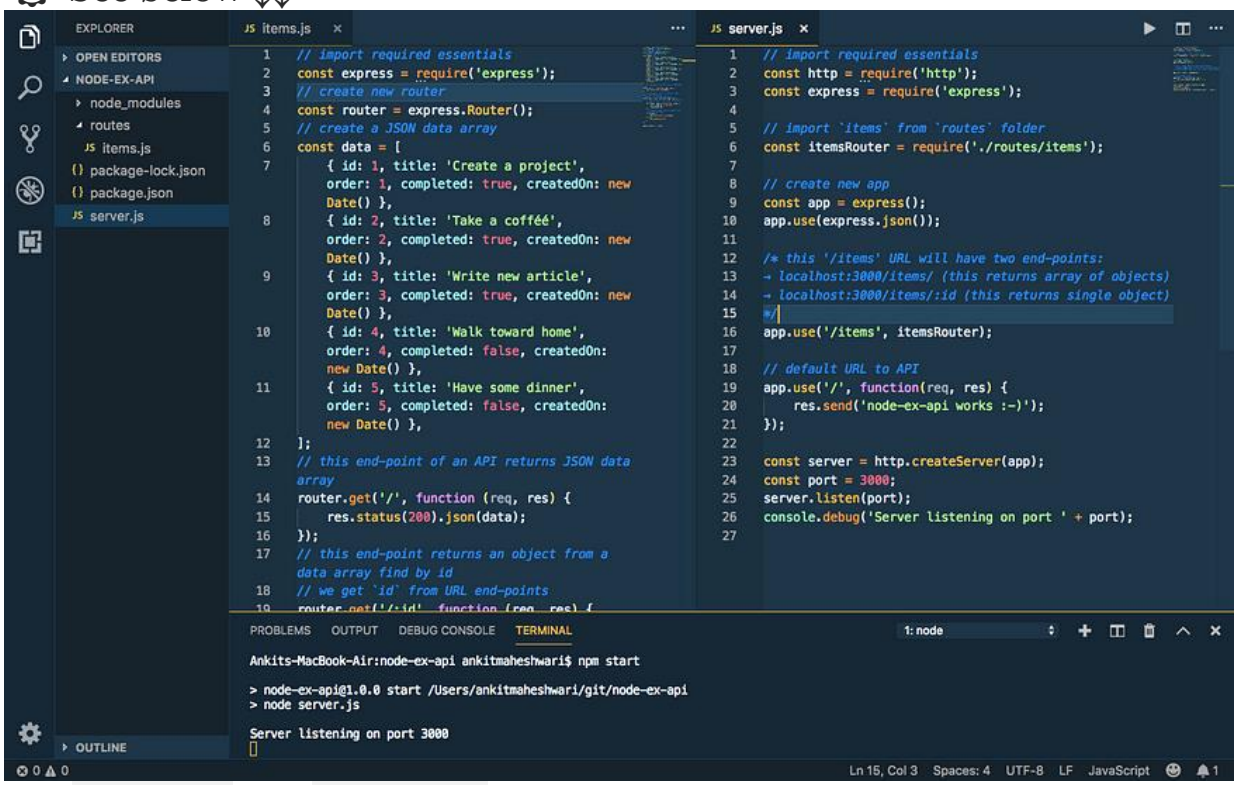
```
npm install cors
```

Now, open up your `node-ex-api/server.js` file and modify with below code:

```
// import required essentials
const http = require('http');
const express = require('express');
var cors = require('cors');// import `items` from `routes`
folder
const itemsRouter = require('./routes/items');

// create new app
const app = express();
app.use(express.json());
```

```
// use it before all route definitions
// allowing below URL to access these APIs end-points
// you can replace this URL(http://localhost:8100) with your
// application URL from where you are calling these APIs
app.use(cors({origin: 'http://localhost:8100'}));

/* this '/items' URL will have two end-points:
→ localhost:3000/items/ (this returns array of objects)
→ localhost:3000/items/:id (this returns single object)
*/
app.use('/items', itemsRouter);

// default URL to API
app.use('/', function(req, res) {
    res.send('node-ex-api works :-)');
});

const server = http.createServer(app);
const port = 3000;
server.listen(port);
console.debug('Server listening on port ' + port);
```

See below ↓↓



The `"items.js"` and `"server.js"` file will look like this ↑

Run API endpoints

Now, run `npm start` if your server is not running. This time we have three end-points:

→ localhost:3000 (Default)

→ localhost:3000/items (Returns all objects)

→ localhost:3000/items/1 (Returns single object of id=1)

🧐 See browser below ↓

Click image 👆 If you can't see this.

12.Create a react application for the student management system having registration, login, contact, about pages and implement routing to navigate through these pages.

Create the project folder containing two sub-folders named client and server.

```
mkdir auth-system
cd auth-system
mkdir client server
```

Navigate into the server folder and create a package.json file.

```
cd server & npm init -y
```

Install Express.js, CORS, and Nodemon.

```
npm install express cors nodemon
```

Express.js is a fast, minimalist framework that provides several features for building web applications in Node.js. CORS is a Node.js package that allows communication between different

domains. Nodemon is a Node.js tool that automatically restarts the server after detecting file changes.

Create an index.js file - the entry point to the web server.

```
touch index.js
```

Set up a simple Node.js server as below:

```
constexpress=require("express");
constcors=require("cors");
constapp=express();
constPORT=4000;

app.use(express.urlencoded({extended:true}));
app.use(express.json());
app.use(cors());

app.get("/api",(req,res)=>{
res.json({message:"Hello world"});
});

app.listen(PORT,()=>{
console.log(`Server listening on ${PORT}`);
});
```

Building the app user interface

In this section, we'll build the user interface for the application allowing users to register and sign in to an application. Users can create an account, log in, and perform 2FA via SMS before they are authorised to view the dashboard.

Create a new React.js project within the client folder.

```
cd client
npx create-react-app ./
```

Delete the redundant files such as the logo and the test files from the React app, and update the App.js file to display Hello World as below.

```
functionApp(){
return(
<div>
<p>Hello World!</p>
</div>
);
```

```
}
exportdefaultApp;
```

Navigate into the src/index.css file and copy the code below. It contains all the CSS required for styling this project.

```
@importurl("https://fonts.googleapis.com/css2?family=Space+Grotesk:wght@300;400;500;600;700&display=swap");
*{
box-sizing:border-box;
margin:0;
padding:0;
font-family:"Space Grotesk",sans-serif;
}
input{
height:45px;
padding:10px15px;
margin-bottom:15px;
}
button{
width:200px;
outline:none;
border:none;
padding:15px;
cursor:pointer;
font-size:16px;
}
.login__container,
.signup__container,
.verify,
.dashboard{
width:100%;
min-height:100vh;
padding:50px70px;
display:flex;
flex-direction:column;
align-items:center;
justify-content:center;
}
.login__form,
.verify__form,
.signup__form{
width:100%;
display:flex;
flex-direction:column;
}
.loginBtn,
.signupBtn,
.codeBtn{
background-color:green;
color:#fff;
margin-bottom:15px;
```

```
}
.signOutBtn{
background-color:#c21010;
color:#fff;
}
.link{
cursor:pointer;
color:rgb(39,147,39);
}
.code{
width:50%;
text-align:center;
}
.verify__form{
align-items:center;
}

@mediascreenand(max-width:800px){
.login__container,
.signup__container,
.verify{
padding:30px;
}
}
```

Install [React Router] - a JavaScript library that enables us to navigate between pages in a React application.

```
npm install react-router-dom
```

Create a components folder within the React app containing the Signup.js, Login.js, PhoneVerify.js and Dashboard.js files.

```
mkdir components
cd components
touch Signup.js Login.js PhoneVerify.js Dashboard.js
```

Update the App.js file to render the newly created components on different routes via React Router.

```
import{BrowserRouter,Route,Routes}from"react-router-dom";
importLoginfrom"./components/Login";
importSignupfrom"./components/Signup";
importDashboardfrom"./components/Dashboard";
importPhoneVerifyfrom"./components/PhoneVerify";

functionApp(){
return(
<BrowserRouter>
<Routes>
```

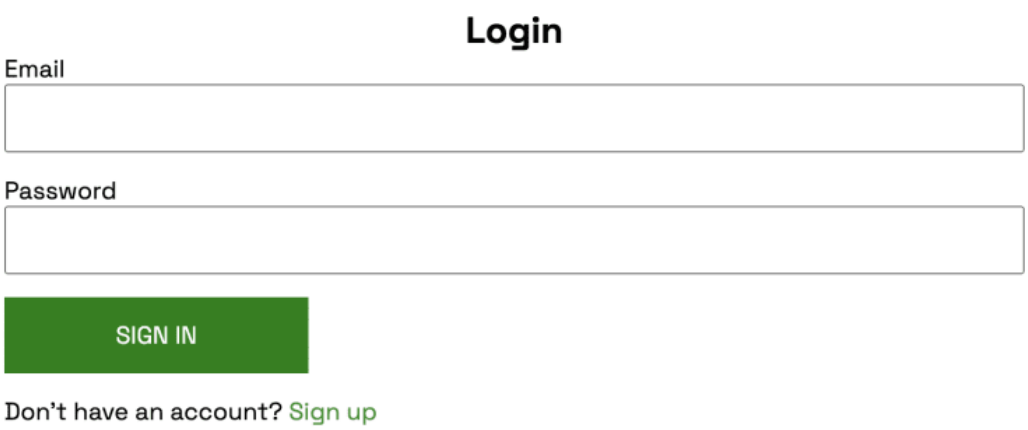
```
<Routepath='/'element={<Login/>}/>
<Routepath='/register'element={<Signup/>}/>
<Routepath='/dashboard'element={<Dashboard/>}/>
<Routepath='phone/verify'element={<PhoneVerify/>}/>
</Routes>
</BrowserRouter>
);
}

exportdefaultApp;
```

The Login page

Copy the code below into the Login.js file. It accepts the email and password from the user.

```
importReact,{useState}from"react";
import{useNavigate}from"react-router-dom";

constLogin=()=>{
const[email,setEmail]=useState("");
const[password,setPassword]=useState("");
constnavigate=useNavigate();

consthandleSubmit=(e)=>{
e.preventDefault();
console.log({email,password});
setPassword("");
setEmail("");
};

constgotoSignUpPage=()=>navigate("/register");

return(
<divclassName='login__container'>
<h2>Login </h2>
<formclassName='login__form'onSubmit={handleSubmit}>
<labelhtmlFor='email'>Email</label>
<input
type='text'
id='email'
name='email'
value={email}
required
onChange={(e)=>setEmail(e.target.value)}
/>
<labelhtmlFor='password'>Password</label>
<input
type='password'
name='password'
id='password'
minLength={8}
```

```
required
value={password}
onChange={(e)=>setPassword(e.target.value)}
/>
<buttonclassName='loginBtn'>SIGN IN</button>
<p>
                Don't have an account?{""}
<spanclassName='link'onClick={gotoSignUpPage}>
                  Sign up
</span>
</p>
</form>
</div>
);
};

exportdefaultLogin;
```



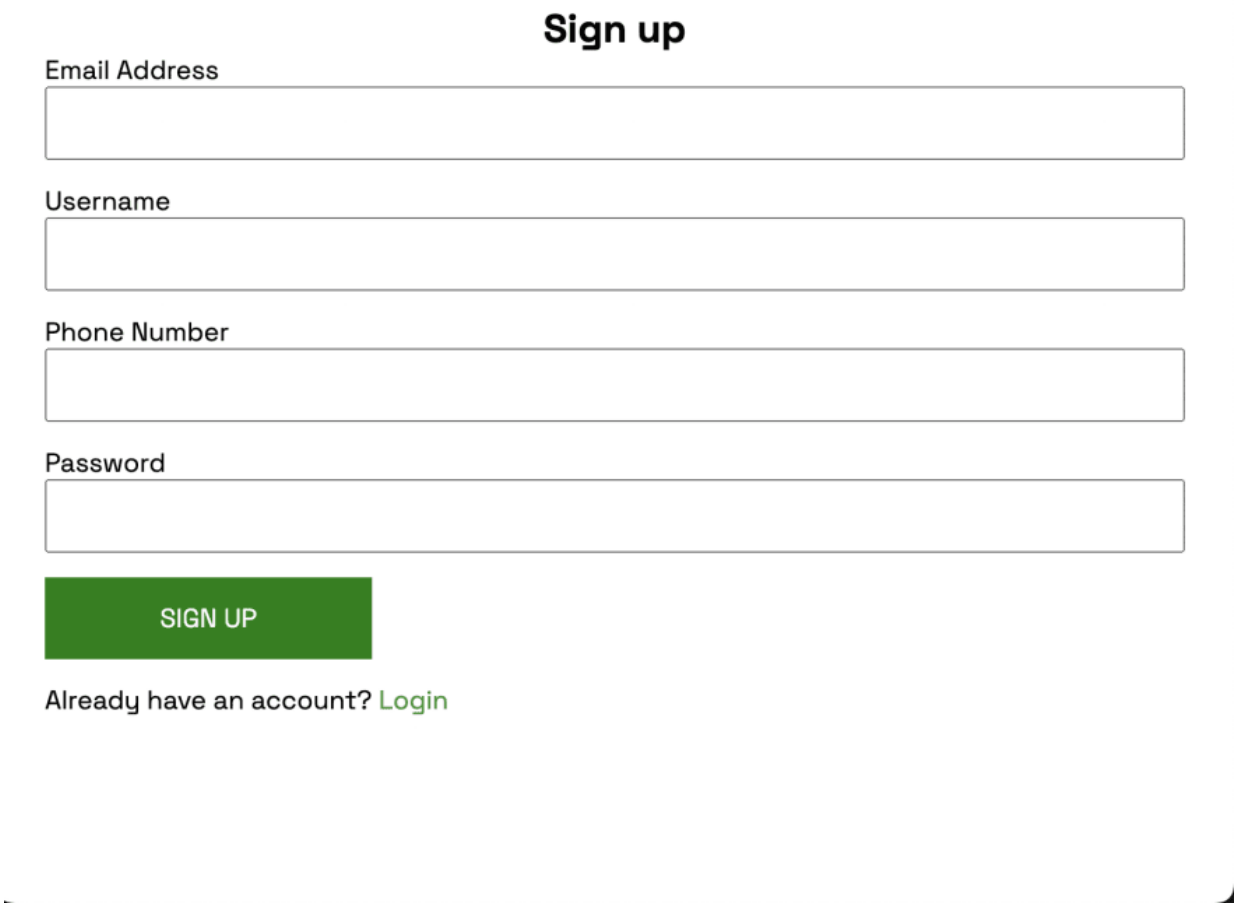
The Sign up page

Copy the code below into the Signup.js file. It accepts the email, username, telephone, and password from the user.

```
importReact,{useState}from"react";
import{useNavigate}from"react-router-dom";

constSignup=()=>{
const[email,setEmail]=useState("");
const[username,setUsername]=useState("");
const[tel,setTel]=useState("");
const[password,setPassword]=useState("");
```

```
constnavigate=useNavigate();

consthandleSubmit=(e)=>{
e.preventDefault();
console.log({email,username,tel,password});
setEmail("");
setTel("");
setUsername("");
setPassword("");
};
constgotoLoginPage=()=>navigate("/");

return(
<divclassName='signup__container'>
<h2>Sign up </h2>
<formclassName='signup__form'onSubmit={handleSubmit}>
<labelhtmlFor='email'>Email Address</label>
<input
type='email'
name='email'
id='email'
value={email}
required
onChange={(e)=>setEmail(e.target.value)}
/>
<labelhtmlFor='username'>Username</label>
<input
type='text'
id='username'
name='username'
value={username}
required
onChange={(e)=>setUsername(e.target.value)}
/>
<labelhtmlFor='tel'>Phone Number</label>
<input
type='tel'
name='tel'
id='tel'
value={tel}
required
onChange={(e)=>setTel(e.target.value)}
/>
<labelhtmlFor='tel'>Password</label>
<input
type='password'
name='password'
id='password'
minLength={8}
required
value={password}
onChange={(e)=>setPassword(e.target.value)}
```

```
/>
<buttonclassName='signupBtn'>SIGN UP</button>
<p>
            Already have an account?{""}
<spanclassName='link'onClick={gotoLoginPage}>
              Login
</span>
</p>
</form>
</div>
);
};

exportdefaultSignup;
```



13.Create a service in react that fetches the weather information from openweathermap.org and the display the current and historical weather information using graphical representation usingchart.js

```
//PAGE LOAD//
//*** GET SAVED ITEMS FROM LOCAL STORAGE ***//
let savedSearchArray =
JSON.parse(localStorage.getItem('savedSearchArray'));
```

```
if (!savedSearchArray) {
    savedSearchArray = [
        'Columbus',
        'Bend',
        'Los Angeles',
        'San Francisco',
        'Savannah',
        'New York',
        'Nashville'
    ];
}

// Define Global Date Variables
// Moment.js
let time = document.querySelector('#time');
time.textContent = moment().format('h:mm a');
let today = document.querySelector('#today');
today.textContent = moment().format('MMM D');

let tomorrow = document.querySelector('#date_1');
tomorrow.textContent = moment().add(1, 'days').format('ddd');
let tomorrow2 = document.querySelector('#date_2');
tomorrow2.textContent = moment().add(2, 'days').format('ddd');
let tomorrow3 = document.querySelector('#date_3');
tomorrow3.textContent = moment().add(3, 'days').format('ddd');
let tomorrow4 = document.querySelector('#date_4');
tomorrow4.textContent = moment().add(4, 'days').format('ddd');
let tomorrow5 = document.querySelector('#date_5');
tomorrow5.textContent = moment().add(5, 'days').format('ddd');


// Define variables for getCurrent function
const searchBtnEl = document.querySelector('#search-btn');
const searchInputEl = document.querySelector('#search-input');

// Define getCurrent function to access current weather conditions
function getCurrent(searchInputEl) {

    // Current Weather Data API call
    const currentWeatherUrl =
'https://api.openweathermap.org/data/2.5/weather?q=' + searchInputEl +
'&units=imperial&appid=c6a9bf78cf3b504fe7e8382ca53765c4';

    fetch(currentWeatherUrl).then(function (response) {
        if (response.ok) {
```

```
            response.json().then(function (data) {

                console.log(data)

                // Append data from API to DOM
                const cityName = document.querySelector('#city-name');
                cityName.textContent = data.name;

                const icon = document.querySelector('#current-icon');
                const iconCall = data.weather[0].icon;
                icon.setAttribute('src', 'http://openweathermap.org/img/wn/'
+iconCall+ '.png');

                const temp = document.querySelector('#current-temp');
                temp.innerHTML = Math.round(data.main.temp) + `&#186;`;

                const description = document.querySelector('#desc');
                description.textContent = data.weather[0].main;

                const humidity = document.querySelector('#current-humid');
                humidity.innerHTML = 'Humidity:  ' + data.main.humidity +
` &#37;`;

                const windSpeed = document.querySelector('#current_wind');
                windSpeed.textContent = 'Wind Speed: ' + data.wind.speed + '
mph';

                //call function to render 5 day forecast in this function for
access to latitude and longitude from city search in current weather
function
                getForecast(data.coord.lat, data.coord.lon);

                //access uv index from onecall used for 5 day forecast
            })
        }
    })
};

//define function to get 5 day forecast //access lat and lon from previous
function
//use one call API instead of 5 day forecast //better accessibility to needed
information
function getForecast(latitude, longitude) {
    const forecastUrl =
'https://api.openweathermap.org/data/2.5/onecall?lat=' + latitude +
'&lon=' + longitude +
```

```
'&units=imperial&exclude=hourly,minutely&appid=c6a9bf78cf3b504fe7
e8382ca53765c4';

   fetch(forecastUrl).then(function (response) {
      if (response.ok) {
         response.json().then(function (data) {

            console.log(data)

            //append uv data to current conditions div
            const uvIndex = document.querySelector('#current_uv');
            uvIndex.textContent = 'UV Index: ' + data.current.uvi;

            // Render icons for each day in forecast
            // Note: Must use <img> tag in html to use .setAttribute method
to set the img src
            const icon1 = document.querySelector('#icon1');
            const icon1Call = data.daily[1].weather[0].icon;
            icon1.setAttribute('src', 'http://openweathermap.org/img/wn/'
+icon1Call+ '@2x.png');

            const icon2 = document.querySelector('#icon2');
            const icon2Call = data.daily[2].weather[0].icon;
            icon2.setAttribute('src', 'http://openweathermap.org/img/wn/'
+icon2Call+ '@2x.png');

            const icon3 = document.querySelector('#icon3');
            const icon3Call = data.daily[3].weather[0].icon;
            icon3.setAttribute('src', 'http://openweathermap.org/img/wn/'
+icon3Call+ '@2x.png');

            const icon4 = document.querySelector('#icon4');
            const icon4Call = data.daily[4].weather[0].icon;
            icon4.setAttribute('src', 'http://openweathermap.org/img/wn/'
+icon4Call+ '@2x.png');

            const icon5 = document.querySelector('#icon5');
            const icon5Call = data.daily[5].weather[0].icon;
            icon5.setAttribute('src', 'http://openweathermap.org/img/wn/'
+icon5Call+ '@2x.png');



            //description forecast
            //TODO: for loop
            const desc1 = document.querySelector('#desc_1');
            desc1.innerHTML = (data.daily[1].weather[0].main);
```

```
            const desc2 = document.querySelector('#desc_2');
            desc2.innerHTML = (data.daily[2].weather[0].main);
            const desc3 = document.querySelector('#desc_3');
            desc3.innerHTML = (data.daily[3].weather[0].main);
            const desc4 = document.querySelector('#desc_4');
            desc4.innerHTML = (data.daily[4].weather[0].main);
            const desc5 = document.querySelector('#desc_5');
            desc5.innerHTML = (data.daily[5].weather[0].main);

            //temp forecast
            //TODO for loop
            // for (i = 0; i < forecast.length; i++)
            const temp1 = document.querySelector('#temp_1');
            temp1.innerHTML = Math.round(data.daily[1].temp.day) +
`&#186;`;
            const temp2 = document.querySelector('#temp_2');
            temp2.innerHTML = Math.round(data.daily[2].temp.day) +
`&#186;`;
            const temp3 = document.querySelector('#temp_3');
            temp3.innerHTML = Math.round(data.daily[3].temp.day) +
`&#186;`;
            const temp4 = document.querySelector('#temp_4');
            temp4.innerHTML = Math.round(data.daily[4].temp.day) +
`&#186;`;
            const temp5 = document.querySelector('#temp_5');
            temp5.innerHTML = Math.round(data.daily[5].temp.day) +
`&#186;`;

            //humidity forecast
            //TODO for loop
            // for (i = 0; i < forecast.length; i++)
            const humid1 = document.querySelector('#humid_1');
            humid1.innerHTML = (data.daily[1].humidity) + `&#37;`;
            const humid2 = document.querySelector('#humid_2');
            humid2.innerHTML = (data.daily[2].humidity) + `&#37;`;
            const humid3 = document.querySelector('#humid_3');
            humid3.innerHTML = (data.daily[3].humidity) + `&#37;`;
            const humid4 = document.querySelector('#humid_4');
            humid4.innerHTML = (data.daily[4].humidity) + `&#37;`;
            const humid5 = document.querySelector('#humid_5');
            humid5.innerHTML = (data.daily[5].humidity) + `&#37;`;

        })
    }
  })
};
```

```
//select each search history list item
const searchHistCon = document.querySelector('#search-history-list')
const searchedItemEl = searchHistCon.querySelectorAll('li.search-
history-item');

console.log(searchedItemEl);

//define function to render search history
function renderSearchHistory() {
    //clear previous selection before appending

    for (i = 0; i < savedSearchArray.length; i++) {

        const newSearchedItem = document.createElement('li');
        newSearchedItem.classList.add('search-history-item');
        newSearchedItem.textContent = savedSearchArray[i];

        newSearchedItem.addEventListener('click', function (event) {
            let newUserInput = event.target.innerText;
            getCurrent(newUserInput);

        })

        if (newSearchedItem.textContent !== "") {
            searchHistCon.prepend(newSearchedItem);
        }


    }
}

//add event listener to search button
searchBtnEl.addEventListener('click', function (event) {
    event.preventDefault();

    //use event to access user input
    console.log(event);
    let userInput= searchInputEl.value;

    getCurrent(userInput);

    //TODO: if statement to check edge cases of misspelled cities, etc
```

```
    savedSearchArray.push(userInput);
    //call function to render search history within this function for access to
needed variables
    renderSearchHistory();

    localStorage.setItem('savedSearchArray',
JSON.stringify(savedSearchArray));
});


const searchHistoryMenuLi = document.querySelector('#search-hist-
menu-li');

searchHistoryMenuLi.addEventListener('click', renderSearchHistory());

getCurrent('Columbus');
```

Whether.html

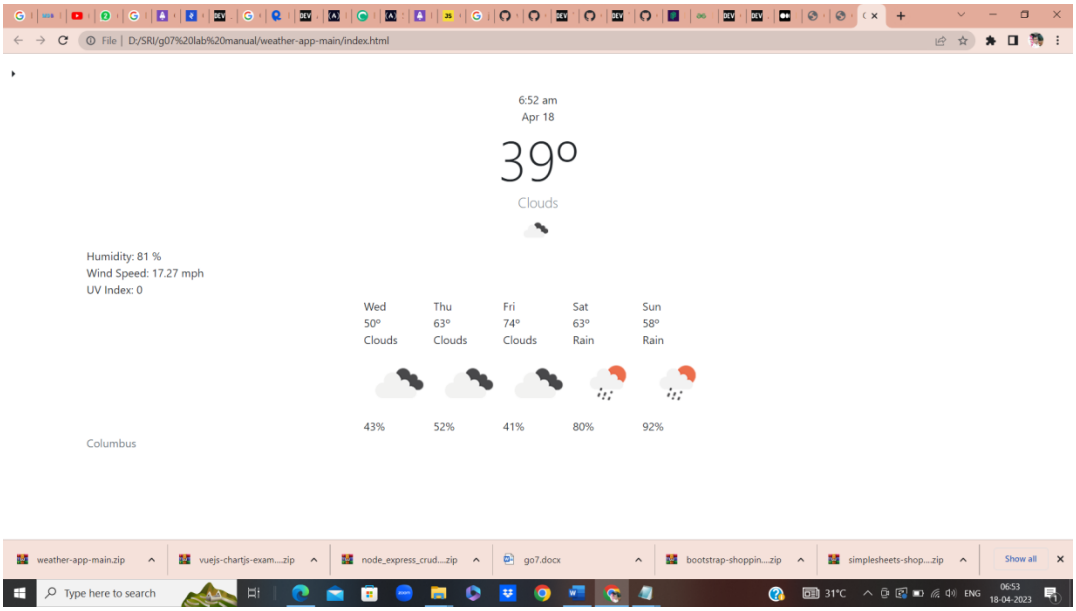
```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-
scale=1.0">
<meta http-equiv="X-UA-Compatible" content="ie=edge" />
<link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min
.css" rel="stylesheet" integrity="sha384-
EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65Vohh
puuCOmLASjC" crossorigin="anonymous">
<link rel="stylesheet"
href="https://use.fontawesome.com/releases/v5.8.1/css/all.css"
        integrity="sha384-
50oBUHEmvpQ+1lW4y57PTFmhCaXp0ML5d60M1M7uH2+nqUivzIeb
hndOJK28anvf" crossorigin="anonymous" />
<link rel="preconnect" href="https://fonts.googleapis.com">
<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
<link
href="https://fonts.googleapis.com/css2?family=Josefin+Sans:ital,wght@
0,100;0,200;0,300;0,400;0,500;1,100;1,200;1,300;1,400&family=Montse
rrat:ital,wght@0,100;0,200;0,300;0,400;0,500;0,600;0,700;0,800;0,900;1,
100;1,200;1,300&family=Noto+Sans:ital,wght@0,100;0,200;0,300;0,400
;0,500;0,600;0,700;0,800;0,900;1,100;1,200;1,300&family=Poppins:ital,
wght@0,100;0,200;0,300;0,400;0,500;0,600;0,700;0,800;0,900;1,100;1,2
00;1,300&family=Roboto:ital,wght@0,100;0,300;0,400;0,500;0,700;0,90
```

```
0;1,100;1,300;1,400&family=Rubik+80s+Fade&family=Source+Sans+Pr
o:wght@300&display=swap" rel="stylesheet">
<link rel="stylesheet" href="assets/css/reset.css" />
<link rel="stylesheet" href="assets/css/style.css" />
<link rel="stylesheet" href="assets/css/responsive.css" />


<title>Current Weather</title>
</head>
<body>
<div class="navbar">
<!-- Dropdown menu -->
<nav class="btn-group dropend">
<button type="button" class="btn dropdown-toggle" data-bs-
toggle="dropdown" aria-expanded="false"></button>
<ul class="dropdown-menu" id="popout-menu">

<li class="nav-item dropdown-item" id="search-box-txt">Search
<form class="active search-form " aria-current="page" href="#">
<div class="input-group">
<input type="text" class="input" id="search-input">
<div class="input-group-div" id="search-btn-group">
<button class="btn btn-default" id="search-btn" type="submit">
<i class="fas fa-search" id="search-icon"></i>
</button>
</div>
</div>
</form>
</li>

<li id="search-hist-menu-li" class="nav-item dropdown-item"> History
<ul id="search-history-list">
<!--append search history in script-->
</ul>
</li>

<!-- <li class="nav-item dropdown-item"> Starred
<ul id="favorites-list"> -->
<!--append search history in script-->
<!-- </ul>
</li> -->
</ul>
</nav>
</div>
```

```
<main class="d-flex flex-column gy-2">

<!-- Current Weather Display -->
<div class="container" id="current-conditions">
<div class="d-flex flex-column align-items-center">
<div class="" id="time"></div>
<div class="" id="today"></div>

<div class="display-1" id="current-temp"></div>
<div class="current-icon-wrapper">
<div class="lead text-muted" id="desc"></div>
<img class="display-1" id="current-icon"/>
</div>
</div>

<div class="hidden">
<div class="" id="current-humid"></div>
<div class="" id="current_wind">Wind Speed: </div>
<div class="" id="current_uv">UV Index: </div>
</div>
</div>

<!-- Forecast Card Container -->
<div class="forecast-content-wrapper container d-flex flex-row justify-
content-center" id="week-forecast">

<div class="forecast-card">

<div class="day" id="date_1"></div>
<div class="temperature" id="temp_1"></div>
<div class="description" id="desc_1"></div>
<img class="weather-icon" alt="weather-icon" id="icon1"></img>

<div class="humidity" id="humid_1"></div>

</div>

<div class="forecast-card">

<div class="day" id="date_2"></div>
<div class="temperature" id="temp_2"></div>

<div class="description" id="desc_2"></div>
<img class="weather-icon" alt="weather-icon" id="icon2"></img>
<div class="humidity" id="humid_2"></div>
```

```
</div>

<div class="forecast-card">

<div class="day" id="date_3"></div>
<div class="temperature" id="temp_3"></div>

<div class="description" id="desc_3"></div>
<img class="weather-icon" alt="weather-icon" id="icon3"></img>
<div class="humidity" id="humid_3"></div>

</div>

<div class="forecast-card">

<div class="day" id="date_4"></div>
<div class="temperature" id="temp_4"></div>

<div class="description" id="desc_4"></div>
<img class="weather-icon" alt="weather-icon" id="icon4"></img>
<div class="humidity" id="humid_4"></div>

</div>

<div class="forecast-card">

<div class="day" id="date_5"></div>
<div class="temperature" id="temp_5"></div>

<div class="description" id="desc_5"></div>
<img class="weather-icon" alt="weather-icon" id="icon5"></img>
<div class="humidity" id="humid_5"></div>

</div>
</div>
<div class="text-muted container" id="city-name">Search a City</div>
</main>


<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle.min.js" integrity="sha384-
MrcW6ZMFYlzcLA8Nl+NtUVF0sA7MsXsP1UyJoMp4YLEuNSfAP+JcXn/tWtIaxVXM" crossorigin="anonymous"></script>
<script
src="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.2.1/jquery.min.js"></
```

```
script>
<script
src="https://cdnjs.cloudflare.com/ajax/libs/moment.js/2.24.0/moment.min
.js"></script>
<script src="assets/script.js"></script>
</body>
</html>
```



14.Create a TODO application in react with necessary components and deploy it into github.

```
<!DOCTYPE
html>
            <html lang="en">
            <head>
            <meta charset="utf-8" />
            <link rel="icon" href="%PUBLIC_URL%/favicon.ico" />
            <meta name="viewport" content="width=device-width, initial-scale=1" />
            <meta name="theme-color" content="#000000" />
            <meta
                  name="description"
                  content="Web site created using create-react-app"
                />
            <link rel="apple-touch-icon" href="%PUBLIC_URL%/logo192.png" />
            <!--
                  manifest.json provides metadata used when your web app is installed on
            a
                  user's mobile device or desktop. See
            https://developers.google.com/web/fundamentals/web-app-manifest/
```

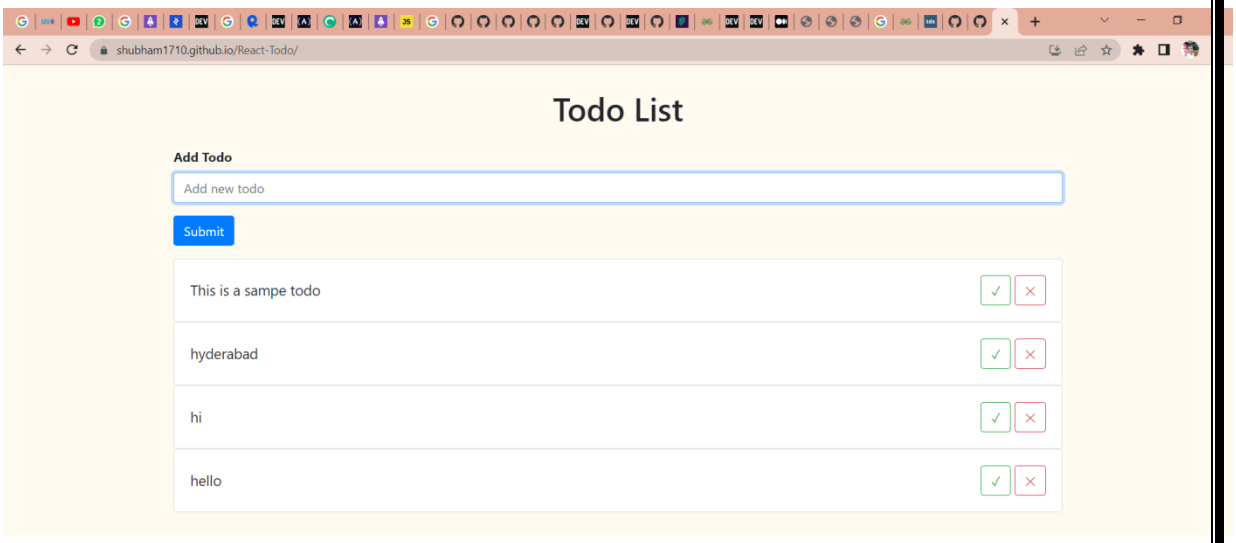
```
    -->
<link rel="manifest" href="%PUBLIC_URL%/manifest.json" />
<!--
      Notice the use of %PUBLIC_URL% in the tags above.
      It will be replaced with the URL of the `public` folder during the
build.
      Only files inside the `public` folder can be referenced from the HTML.
      Unlike "/favicon.ico" or "favicon.ico", "%PUBLIC_URL%/favicon.ico" will
      work correctly both with client-side routing and a non-root public URL.
      Learn how to configure a non-root public URL by running `npm run
build`.
    -->
<title>Todo List</title>
</head>
<body>
<noscript>You need to enable JavaScript to run this app.</noscript>
<div id="root"></div>
<!--
      This HTML file is a template.
      If you open it directly in the browser, you will see an empty page.
      You can add webfonts, meta tags, or analytics to this file.
      The build step will place the bundled scripts into the <body> tag.
      To begin the development, run `npm start` or `yarn start`.
      To create a production bundle, use `npm run build` or `yarn build`.
    -->
</body>
</html>
```

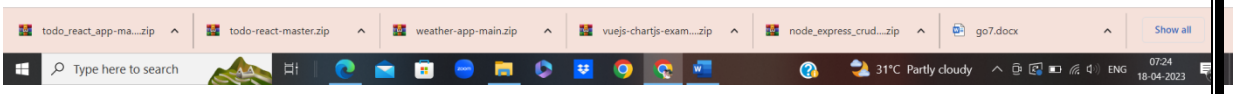
App.js file:-

```
importReactfrom"react";
                        import"./App.css";
                        import { Button, Card, Form } from'react-bootstrap';
                        import'bootstrap/dist/css/bootstrap.min.css';


                        functionTodo({ todo, index, markTodo, removeTodo }) {
                        return (
                        <div
                        className="todo"
                        >
                        <spanstyle={{ textDecoration: todo.isDone ? "line-through" : ""
                        }}>{todo.text}</span>
                        <div>
                        <Buttonvariant="outline-success"onClick={()
                        =>markTodo(index)}>✓</Button>{' '}
                        <Buttonvariant="outline-danger"onClick={()
```

```
=>removeTodo(index)}>X</Button>
</div>
</div>
  );
}

functionFormTodo({ addTodo }) {
const [value, setValue] =React.useState("");

consthandleSubmit= e => {
    e.preventDefault();
if (!value) return;
addTodo(value);
    setValue("");
  };

return (
<FormonSubmit={handleSubmit}>
<Form.Group>
<Form.Label><b>Add Todo</b></Form.Label>
<Form.Controltype="text"className="input"value={value}
onChange={e => setValue(e.target.value)} placeholder="Add new
todo"/>
</Form.Group>
<Buttonvariant="primary mb-3"type="submit">
      Submit
</Button>
</Form>
  );
}

functionApp() {
const [todos, setTodos] =React.useState([
    {
text: "This is a sampe todo",
isDone: false
    }
  ]);

constaddTodo= text => {
const newTodos = [...todos, { text }];
    setTodos(newTodos);
  };

constmarkTodo= index => {
```

```
const newTodos = [...todos];
    newTodos[index].isDone=true;
    setTodos(newTodos);
  };

constremoveTodo= index => {
const newTodos = [...todos];
    newTodos.splice(index, 1);
    setTodos(newTodos);
  };

return (
<divclassName="app">
<divclassName="container">
<h1className="text-center mb-4">Todo List</h1>
<FormTodoaddTodo={addTodo} />
<div>
          {todos.map((todo, index) => (
<Card>
<Card.Body>
<Todo
key={index}
index={index}
todo={todo}
markTodo={markTodo}
removeTodo={removeTodo}
/>
</Card.Body>
</Card>
          ))}
</div>
</div>
</div>
  );
}
exportdefaultApp;
```


shubham1710.github.io/React-Todo/
Todo List
Add Todo
Add new todo
Submit
This is a sampe todo
hyderabad
hi
hello


todo_react_app-ma....zip
todo-react-master.zip
weather-app-main.zip
vuejs-chartjs-exam....zip
node_express_crud....zip
go7.docx
Show all
Type here to search
31°C Partly cloudy
ENG
07:24
18-04-2023