



MARRI LAXMAN REDDY **INSTITUTE OF TECHNOLOGY AND MANAGEMENT**

(AN AUTONOMOUS INSTITUTION)

(Approved by AICTE, New Delhi & Affiliated to JNTUH, Hyderabad)

Accredited by NAAC with 'A' Grade & Recognized Under Section 2(f) & 12(B) of the UGC act, 1956

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

VISION & MISSION OF THE INSTITUTE

Vision of the Institute

To be a globally recognized institution that fosters innovation, excellence, and leadership in education, research, and technology development, empowering students to create sustainable solutions for the advancement of society.

Mission of the Institute

- To foster a transformative learning environment that empowers students to excel in engineering, innovation, and leadership.
- To produce skilled, ethical, and socially responsible engineers who contribute to sustainable technological advancements and address global challenges.
- To shape future leaders through cutting-edge research, industry collaboration, and community engagement.



MARRI LAXMAN REDDY INSTITUTE OF TECHNOLOGY AND MANAGEMENT

(AN AUTONOMOUS INSTITUTION)

(Approved by AICTE, New Delhi & Affiliated to JNTUH, Hyderabad)

Accredited by NAAC with 'A' Grade & Recognized Under Section 2(f) & 12(B) of the UGC act, 1956

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

DEPARTMENT VISION & MISSION

Vision:

To empower the students to be technologically adept, innovative, self-motivated and responsible global citizen possessing human values and contribute significantly towards high quality technical education by harmonizing innovation with sustainability.

Mission:

- To offer high-quality education in the computing fields by providing an environment where the knowledge is gained and applied to participate in research, for both students and faculty.
- To develop the problem solving skills in the students to be ready to deal with cutting edge technologies of the industry.
- To make the students and faculty excel in their professional fields by inculcating the communication skills, leadership skills, team building skills with the organization of various co-curricular and extra-curricular programmes.
- To provide the students with theoretical and applied knowledge, and adopt an education approach that promotes lifelong learning and ethical growth.



MARRI LAXMAN REDDY
INSTITUTE OF TECHNOLOGY AND MANAGEMENT

(AN AUTONOMOUS INSTITUTION)

(Approved by AICTE, New Delhi & Affiliated to JNTUH, Hyderabad)

Accredited by NAAC with 'A' Grade & Recognized Under Section 2(f) & 12(B) of the UGC act, 1956

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

CERTIFICATE

This is to certify that this manual is a **bonafide record of practical work** carried out in **NODE JS /REACT JS/ DJANGO Lab** for the B.Tech Computer Science and Engineering (CSE) III Semester Programme during the academic year **2026–2027**.

This manual has been prepared by the **Department of Computer Science and Engineering (CSE)** with our own efforts and to the best of our knowledge.

Signature of Lab Faculty Signature of HOD



MARRI LAXMAN REDDY **INSTITUTE OF TECHNOLOGY AND MANAGEMENT**

(AN AUTONOMOUS INSTITUTION)

(Approved by AICTE, New Delhi & Affiliated to JNTUH, Hyderabad)

Accredited by NAAC with 'A' Grade & Recognized Under Section 2(f) & 12(B) of the UGC act, 1956

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING **PREFACE**

This lab manual entitled “**NODE JS/REACT JS/DJANGO Lab**” is intended for the use of II B.Tech I Semester Computer Science and Engineering (CSE) students of **Marri Laxman Reddy Institute of Technology and Management, Dundigal, Hyderabad**.

The main objective of the Software Engineering Lab is to provide hands-on experience with intelligent computing techniques such as neural networks, fuzzy logic, and genetic algorithms. Students learn to solve complex, real-world problems that do not have exact mathematical models.

It enhances skills in Artificial Intelligence and Machine Learning through practical implementation and experimentation. The lab prepares learners for advanced research and industry applications in adaptive and intelligent systems.



MARRI LAXMAN REDDY **INSTITUTE OF TECHNOLOGY AND MANAGEMENT**

(AN AUTONOMOUS INSTITUTION)

(Approved by AICTE, New Delhi & Affiliated to JNTUH, Hyderabad)

Accredited by NAAC with 'A' Grade & Recognized Under Section 2(f) & 12(B) of the UGC act, 1956

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

GENERAL INSTRUCTIONS

1. Students are instructed to come to the Software Engineering laboratory on time. Latecomers are not entertained in the lab.
2. Students should be punctual. Experiments will not be repeated for latecomers.
3. Students must come prepared with the experiments to be performed.
4. Students must display their identity cards before entering the lab.
5. Mobile phones are strictly not allowed inside the lab.
6. Any damage or loss of system components (keyboard, mouse, etc.) is the responsibility of the student, and a penalty will be imposed.
7. Students must update their records and observation books session-wise. The observation book should be signed by the faculty before leaving the lab.
8. Lab records must be submitted in the next lab session for correction.
9. Students should not move around unnecessarily during the lab session.
10. In case of emergency, students must obtain written permission from the concerned faculty.
11. Faculty members may suspend any student from the lab session on disciplinary grounds.
12. Students must not copy outputs from others and should write their own results.



MARRI LAXMAN REDDY **INSTITUTE OF TECHNOLOGY AND MANAGEMENT**

(AN AUTONOMOUS INSTITUTION)

(Approved by AICTE, New Delhi & Affiliated to JNTUH, Hyderabad)

Accredited by NAAC with 'A' Grade & Recognized Under Section 2(f) & 12(B) of the UGC act, 1956

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING **SAFETY MEASURES**

To ensure the safe and efficient use of the Computer Science and Engineering (AI & ML) laboratory, all students must strictly follow the safety guidelines given below:

1. General Conduct

- Maintain silence and discipline during lab sessions.
- Do not bring food, drinks, or chewing gum into the lab.
- Use lab resources responsibly and follow all instructions given by the instructor or lab assistant.

2. Electrical Safety

- Do not touch electrical switches, sockets, or plugs with wet hands.
- Avoid overloading power sockets with unauthorized devices.
- Immediately report any loose connections, sparks, or unusual noises.

3. Computer and Equipment Handling

- Handle computers, keyboards, mouse, and peripherals with care.
- Do not open or tamper with hardware components.
- Use only the assigned system; do not switch without permission.

4. Software and Data Safety

- Use only authorized software installed by the lab administrator.
- Do not install or modify any software without approval.
- Save your work frequently and maintain backups.

5. Cyber Security and Network Usage

- Keep login credentials confidential.
- Do not access restricted websites or servers.
- Avoid activities such as hacking, gaming, or using pirated content.

6. Emergency Preparedness

- Be aware of emergency exits, fire extinguishers, and first aid kits.
- In case of emergencies, remain calm and inform the instructor immediately.
- Follow evacuation procedures properly.

7. Post-Lab Procedures

- Log out and shut down the system properly.
- Keep the workstation clean and organized.
- Return borrowed materials to their place.

8. Hygiene and Cleanliness

- Wash or sanitize hands before and after using systems.
- Do not place unnecessary items on the workstation.
- Report spills or cleanliness issues to lab staff.

COURSE CONTENT - LIST OF EXPERIMENTS

Module	List of Experiments	No. of Lectures
1	Exercise-1 Build a responsive web application for shopping cart with registration, login, catalog and Cart pages using CSS3 features, flex and grid	
2	Exercise-2 Make the above web application responsive web application using Bootstrap framework.	
3	Exercise-3 Use JavaScript for doing client – side validation of the pages implemented in experiment 1 and experiment 2.	
4	Exercise-4 Explore the features of ES6 like arrow functions, callbacks, promises, a sync /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	Exercise-5 Develop a java stand alone application that connects with the database (Oracle / my Sql) and perform the CRUD operation on the database tables.	
6	Exercise-6 Create an xml for the bookstore. Validate the same using both DTD and XSD.	
7	Exercise-7 Design a controller with servlet that provides the interaction with application developed in experiment 1 and the database created in experiment 5.	
8	Exercise-8 Maintaining the transactional history of any user is very important. Explore the various session tracking mechanism (Cookies, HTTP Session)	
9	Exercise-9 Create a custom server using http module and explore the other modules of Node JS like OS, path, event.	
10	Exercise-10 Develop an express web application that can interact with REST API to perform CRUD operations on student data. (Use Postman)	
11	Exercise-11 For the above application create authorized end points using JWT (JSON Web Token).	
12	Exercise-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	Exercise-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 using chart.js	
14	Exercise-14 Create a TODO application in react with necessary components and deploy it into github.	

TEXTBOOKS:

1. Web Technologies, Uttam K Roy, Oxford University Press
2. The Complete Reference PHP — Steven Holzner, Tata McGraw-Hill

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, 2nd Edition, 2008.
3. Vasan Subramanian, Pro MERN Stack, Full Stack Web App Development with Mongo, Express, React, and Node, 2nd Edition, A Press.

14. ELECTRONIC RESOURCES

Resource Type	Title/Description	Link
Online Courses	Introduction to Web Development Coursera	https://www.google.com/aclk?sa=l&ai=DChcSEwiYkbPKiZSMAxWjzRYFHQI9CLQYABAAGgJ0bA&co=1&gclid=Cj0KCQjw s-S-BhD2ARIsALssG0Y0SLVO-41taR2bo7Ljj_uTiIQMsQSdWiHifRsBKmC4Vuyyg_S1MaAjDBEALw_wcB&sig=AD64_0b8yQaNbAHIO5nQYScIQcDJXj-1w&q&adurl&ved=2ahUKEwjk4K3KiZSM_AxUP2TOHHYdbL_0Q0Qx6BAGbEAE
	Web Technology - Online Courses Swayam 2	https://onlinecourses.swayam2.ac.in/nu24_cs09/preview
E-Books	Web Technologies:HTML, JAVASCRIPT, PHP, JAVA, JSP, ASP.NET, XML and Ajax, Black Book	https://www.wileyindia.com/web-technologies-html-javascript-php-java-jsp-asp-net-xml-and-ajax-black-book.html
	Web Design and Development: Concepts, Methodologies, Tools, and Applications	https://www.igi-global.com/gateway/book/125530
Research Journals	Web of Technology: Multidimensional Research Journal	https://webofjournals.com/index.php/4
	A Brief Study on Web Technology	https://ijrcst.org/DOC/33-a-brief-study-on-web-technology.pdf
	Introduction to Web Technologies - Part 1 Web Technologies ... YouTube · Naresh i Technologies 14 Oct 2016	https://www.youtube.com/watch?v=Jsbx B217QGY

YouTube Channels	"What Are Web Technologies? Unlocking the Secrets Behind Web Development!"	https://www.youtube.com/watch?v=5BUkS9qBwoo
-------------------------	--	---

COURSE PLAN

S. No	Lecture no	Topic	COs	Text book/ Chapter
1	LH1	WEEK-1: Build a responsive web application for shopping cart with registration, login, catalog and Cart pages using CSS3 features, flex and grid.	CO1	T1
2	LH2	WEEK-2: Make the above web application responsive web application using Bootstrap framework.	CO 1	T1
3	LH3	WEEK-3: Use JavaScript for doing client – side validation of the pages implemented in experiment 1 and experiment 2.	CO 1	T1
4	LH4	WEEK-4: Explore the features of ES6 like arrow functions, callbacks, promises, a sync /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	CO 1	T1
5	LH5	WEEK-5: Develop a java stand alone application that connects with the database (Oracle / my Sql) and perform the CRUD operation on the database tables.	CO 2	T1
6	LH6	WEEK-6: Create an xml for the bookstore. Validate the same using both DTD and XSD.	CO 2	T1
7	LH7	WEEK-7 Design a controller with servlet that provides the interaction with application developed in experiment 1 and the database created in experiment 5.	CO 3	T1
8	LH8	WEEK-8 : Maintaining the transactional history of any user is very important. Explore the various session tracking mechanism (Cookies, HTTP Session)	CO 3	T1
		CIE - I		
9	LH9	WEEK-9: Create a custom server using http module and explore the other modules of Node JS like OS, path, event.	CO 4	T1
10	LH10	WEEK-10: Develop an express web application that can interact with REST API to perform CRUD operations on student data. (Use Postman)	CO 4	T1
11	LH11	WEEK-11: For the above application create authorized end points using JWT (JSON Web Token).	CO 4	T1
12	LH12	WEEK-12: Create a react application for the student management system having registration, login, contact, about pages and implement routing to navigate through these pages.	CO 5	T1

13	LH13	WEEK-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 using chart.js	CO 5	T1
14	LH14	WEEK-14: Create a TODO application in react with necessary components and deploy it into github.	CO 5	T1
		CIE-II		

PROGRAM OUTCOMES & PROGRAM SPECIFIC OUTCOMES

PO NO	NBA Statement / Vital Features		
	Graduate Attributes	Program Outcomes	No. of key competencies
PO1	Engineering knowledge	Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.	3
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.	10
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.	10
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.	11
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.	1
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	5
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.	3
PO8	Ethics	Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.	3
PO9	Individual and team work	Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.	12

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.	5
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.	12
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	8

PO NO	NBA Statement / Vital Features	
	Program Specific Outcomes	No. of key competencies
PSO1	Applications of Computing: Ability to use knowledge in various domains to provide solution to new ideas and innovations.	2
PSO2	Programming Skills: Identify required data structures, design suitable algorithms, develop and maintain software for real world problems.	3
PSO3	Make use of computational and experimental knowledge for creating innovative career paths, to be an entrepreneur and desire for higher studies.	2



MARRI LAXMAN REDDY INSTITUTE OF TECHNOLOGY AND MANAGEMENT

(AN AUTONOMOUS INSTITUTION)

(Approved by AICTE, New Delhi & Affiliated to JNTUH, Hyderabad)

Accredited by NAAC with 'A' Grade & Recognized Under Section 2(f) & 12(B) of the UGC act, 1956

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

PROGRAM EDUCATIONAL OBJECTIVES (PEOs)

PO NO	NBA Statement / Vital Features
	Program Specific Outcomes
PEO1	Professional Competence
	Graduates will possess strong theoretical and practical knowledge in Artificial Intelligence and Machine Learning, enabling them to solve complex real-world problems, pursue higher education, or excel in professional careers.
PEO2	Innovation and Research Orientation
	Graduates will engage in innovative practices, cutting-edge research, and contribute to the advancement of AI and ML technologies through collaboration with industry and academia.
PEO3	Leadership and Lifelong Learning
	Graduates will exhibit leadership qualities, effective communication, and teamwork skills, and will continuously upgrade their knowledge to adapt to evolving technological landscapes.
PEO4	Entrepreneurial and Community Engagement
	Graduates will leverage entrepreneurial skills and a sense of civic responsibility to create AI-driven solutions that benefit local and global communities.



MARRI LAXMAN REDDY
INSTITUTE OF TECHNOLOGY AND MANAGEMENT
(AN AUTONOMOUS INSTITUTION)

(Approved by AICTE, New Delhi & Affiliated to JNTUH, Hyderabad)
Accredited by NAAC With 'A' Grade & Recognized Under Section 2(f) & 12(B) of the UGC act, 1956

B.Tech. II Year I Sem.

L T P C

0 0 2 1

Prerequisites: Object Oriented Programming through Java, HTML Basics.

Course Objectives:

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

Course Outcomes: After Completion of the Course, Students Should be able to:

1. Design responsive web applications using HTML5, CSS3, Flexbox, Grid, Bootstrap, and client-side scripting techniques.
2. Apply JavaScript and ES6 features including validation logic, asynchronous programming, API consumption, and graphical data representation.
3. Develop server-side applications using Java, Servlets, Node.js, Express framework, and database connectivity with CRUD functionality.
4. Implement session management, authentication mechanisms, secure RESTful services, and authorization using JWT for web applications.

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 open weather map.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 / my Sql) 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 open weather map.org and the display the current and historical weather information using graphical representation using chart.js
14. Create a TODO application in react with necessary components and deploy it into github.

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. Vasam Subramanian, Pro MERN Stack, Full Stack Web App Development with Mongo, Express, React, and Node, 2nd Edition, A Press.

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="/assets/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/XRVoUu3IX4sGWtiC3MPpFnJvZNq71
    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" />
<iclass="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 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
        q7IVWDCa8LSqgS/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"/>
      <iclass="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-around;
    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-around;
  }
```

```
  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);
}
```

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-around;
    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-around;
  }
  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-around;
  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-around;
  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("../assets/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;
  }

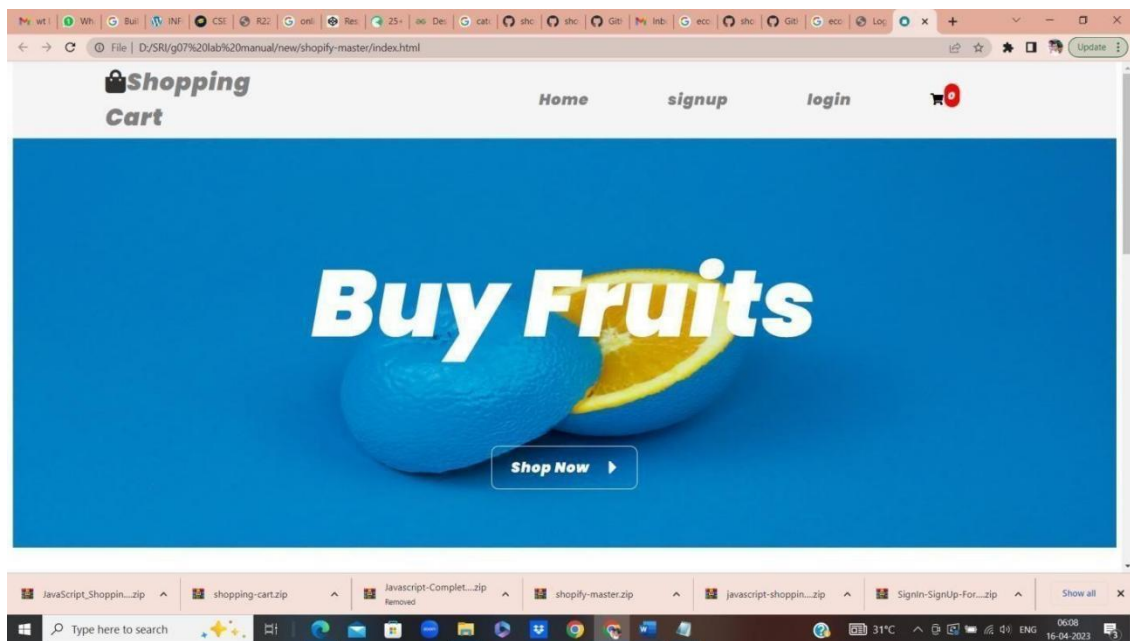
  section > main {
    grid-template-columns: repeat(2, 35vw);
    grid-template-rows: repeat(2, 55vh);
  }
}
```

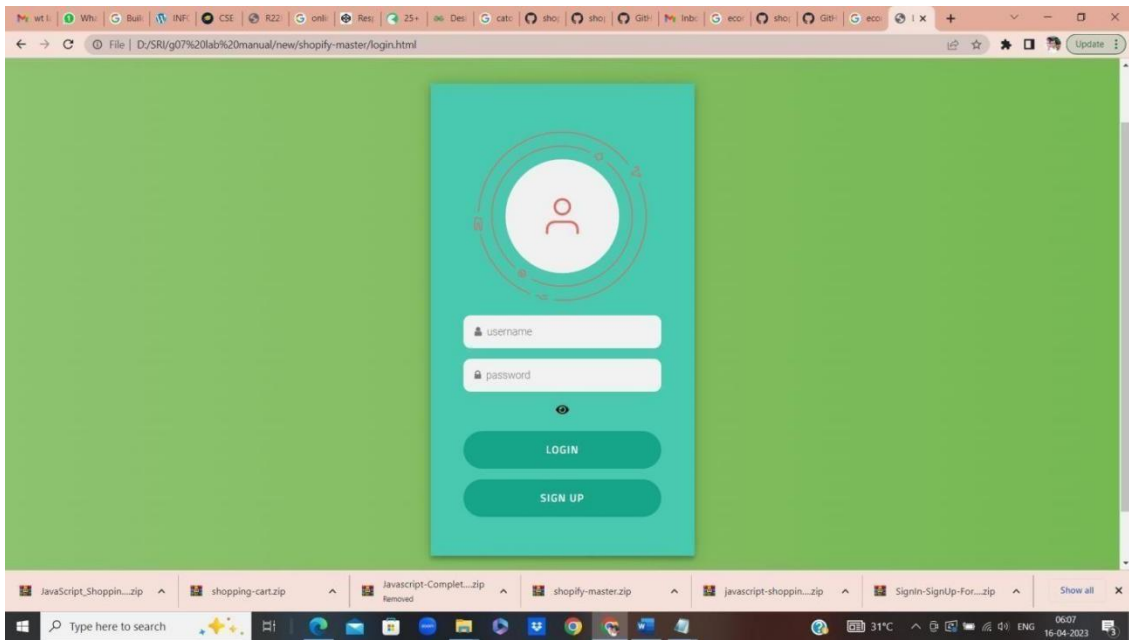
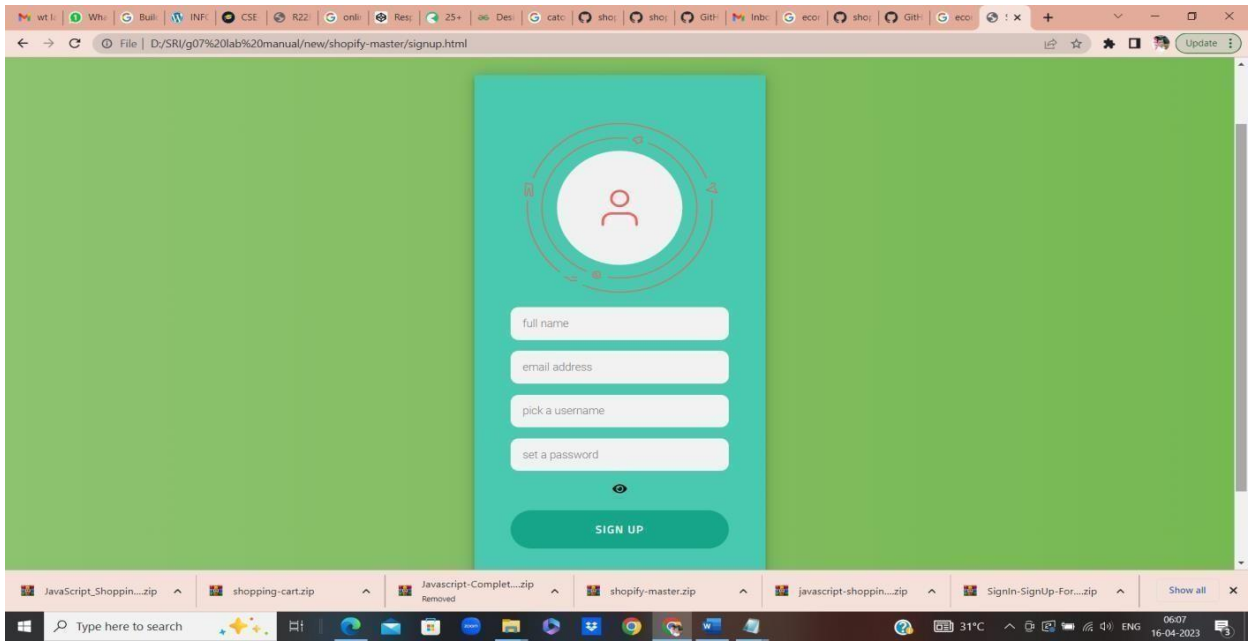
```

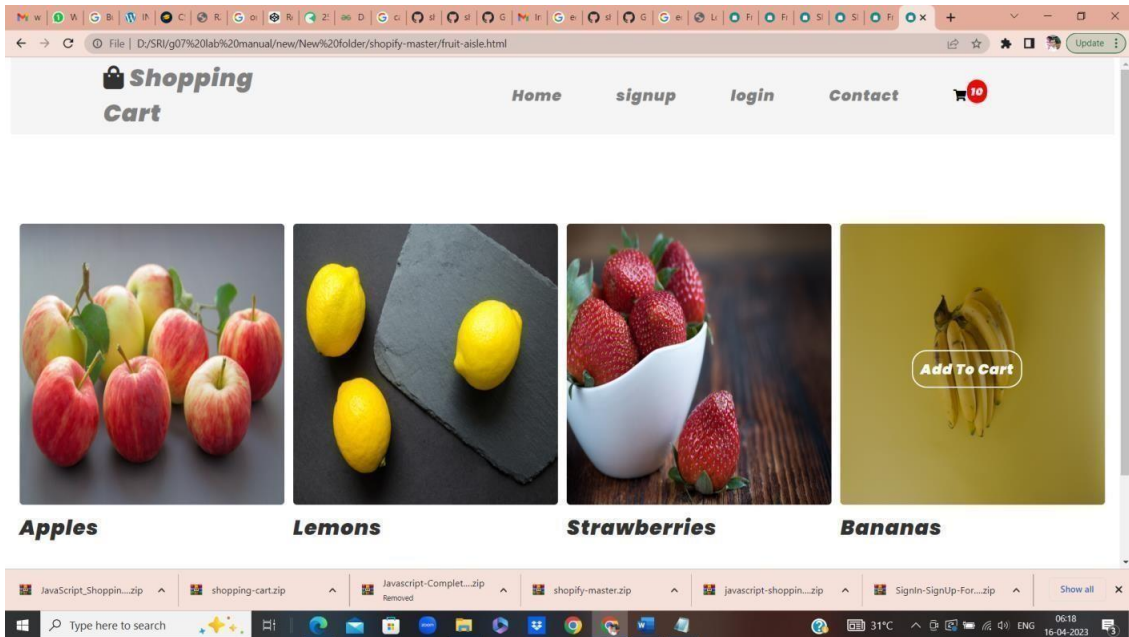
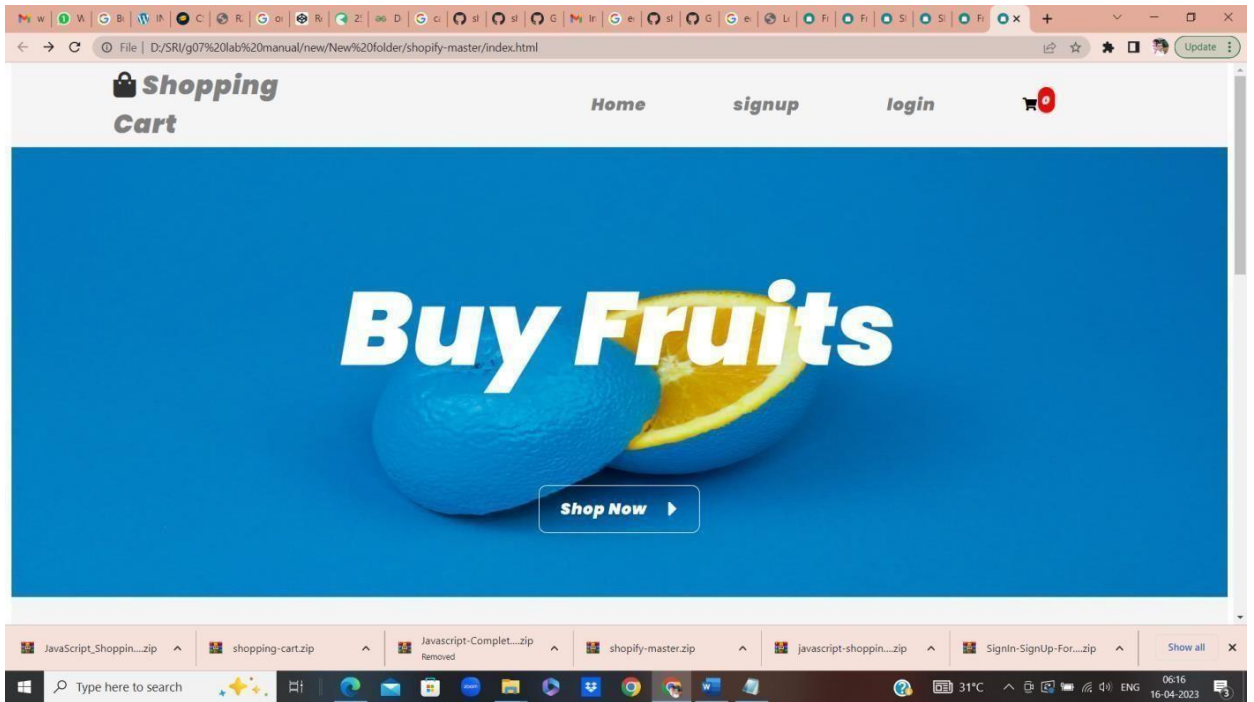
}
.item-1::before,
.item-2::before,
.item-3::before,
.item-4::before {
padding: 10px 15px;
left: 20%;
top: 43%;
}
.item-1::after,
.item-2::after,
.item-3::after,
.item-4::after {
bottom: -18%;
font-size: 1.5rem;
}
}

```

OUTPUT:







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>
</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"></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"></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"></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">&#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="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.// -->
        </div>
    </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
<code class="text-danger">Add to Cart</code> button 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>
    </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>
</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 properties
    // =====
    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 properties
    // =====
    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 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-dark 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;}
}

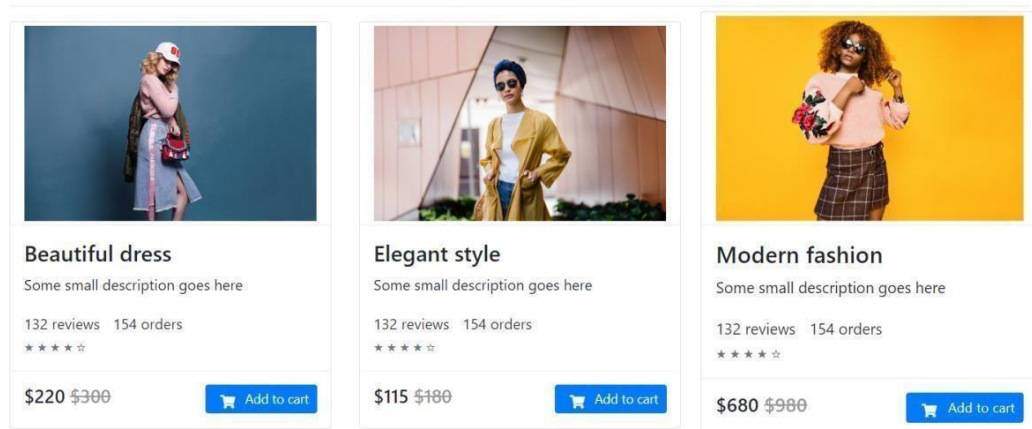
@-webkit-keyframes fadeout {
  from {bottom: 30px; opacity: 1;}
  to {bottom: 60px; opacity: 0;}
}

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

```



Bootstrap 4 - Shopping Cart



3. Use JavaScript for doing client – side validation of the pages implemented in experiment 1 and experiment 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();
}

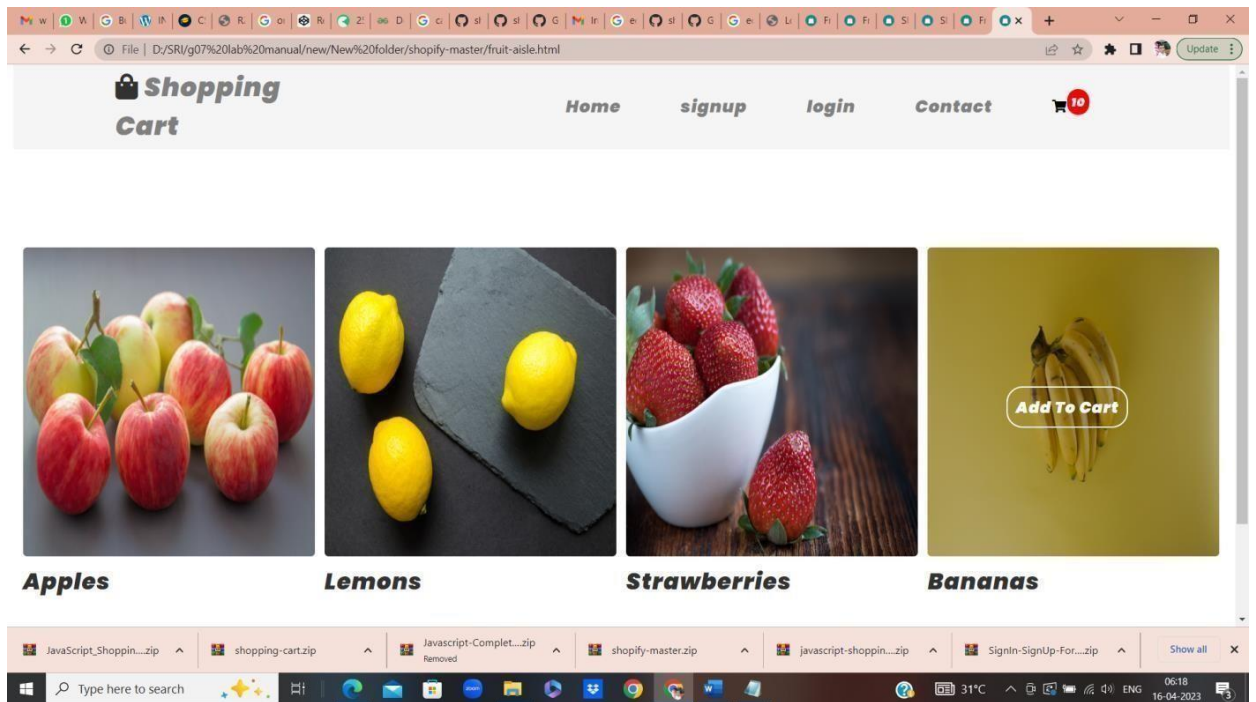
function saveCart() {
    if (window.localStorage) {
        localStorage.food_cart = JSON.stringify(food_cart);
    }
}

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 = `${item.Product}</span><span onclick='deleteItem(" + i + ")'><i class='fas
fa-trash'></i></span><span>${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. 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.

Arrow functions are introduced in [ES6](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Functions/Arrow_functions), which provides you a more accurate way to write the [functions in JavaScript](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Functions/Arrow_functions). 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

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

Example

1. let Promise = **new** Promise((resolve, reject)=>{
2. let a = 3;
3. **if**(a==3){
4. resolve('Success');
5. }
6. **else**{
7. reject('Failed');

```

8.     }
9. })
10. Promise.then((message)=>{
11.     console.log("It is then block. The message is: ?+ message)
12. }).catch((message)=>{
13.     console.log("It is Catch block. The message is: ?+ message)
14. })

```

Output

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

JavaScript Async/Await

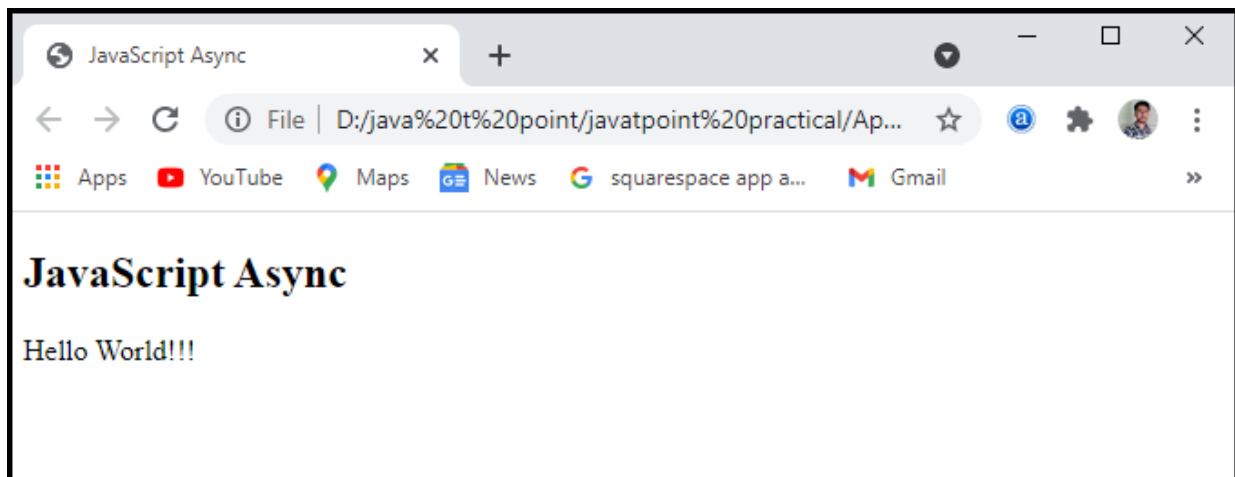
JavaScript is always synchronous and single-threaded that provides the event loops. The event loops enable us to queue up an activity. This activity will not happen until the loops become available after the program that queued the action has completed the execution. However, our program contains a large 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.

```

1. <!DOCTYPE html>
2. <html>
3. <head>
4.     <meta charset="utf-8">
5. <title>JavaScript Async</title>
6. </head>
7. <body>
8.     <h2>JavaScript Async</h2>
9.     <p id="main"></p>
10. <script>
11. function myDisplayer(some) {
12.     document.getElementById("main").innerHTML = some;
13. }
14. async function myfirstFunction() {
15.     return "Hello World!!!";
16. }
17. myfirstFunction().then(
18.     function(value) { myDisplayer(value);},
19.     function(error) { myDisplayer(error);}
20. );
21. </script>
22. </body>
23. </html>

```

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



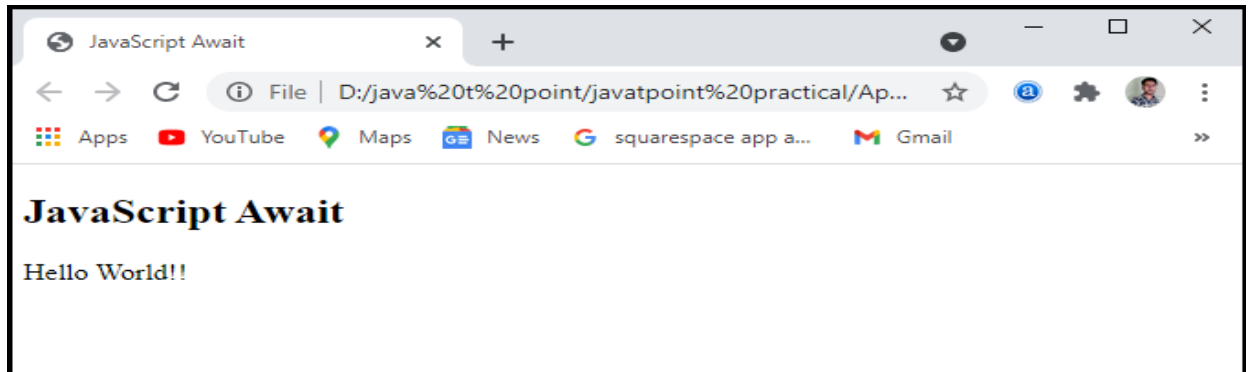
JavaScript Await

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.

1. `<!DOCTYPE html>`
2. `<html>`
3. `<head>`
4. `<meta charset="utf-8">`
5. `<title>JavaScript Await</title>`
6. `</head>`
7. `<body>`
8. `<h2>JavaScript Await</h2>`
9. `<p id="main"></p>`
10. `<script>`
11. `async function myDisplay() {`
12. `let myPromise = new Promise(function(myResolve, myReject) {`
13. `myResolve("Hello World!!!");`
14. `});`
15. `document.getElementById("main").innerHTML = await myPromise;`
16. `}`
17. `myDisplay();`
18. `</script>`
19. `</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 readyToSend =
(url+"lat="+Lat+"&lon="+Lon+"&appid="+key);
    fetch(readyToSend)
    .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>" +
        "<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();

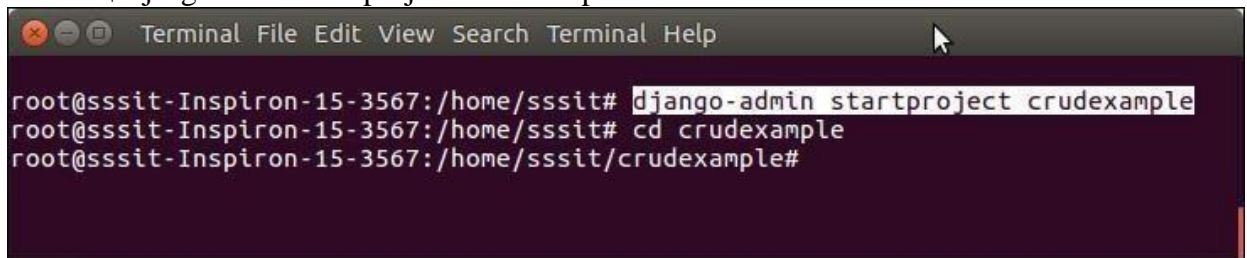
```



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

1. Create a Project

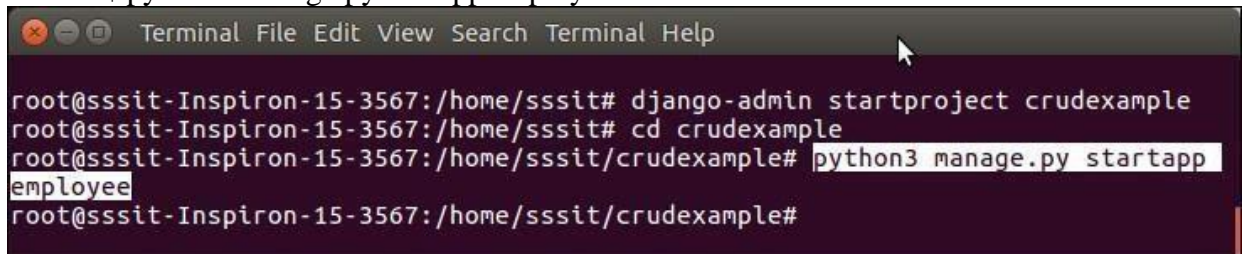
1. `$ django-admin startproject crudexample`



```
Terminal File Edit View Search Terminal Help
root@sssit-Inspiron-15-3567:/home/sssit# django-admin startproject crudexample
root@sssit-Inspiron-15-3567:/home/sssit# cd crudexample
root@sssit-Inspiron-15-3567:/home/sssit/crudexample#
```

2. Create an App

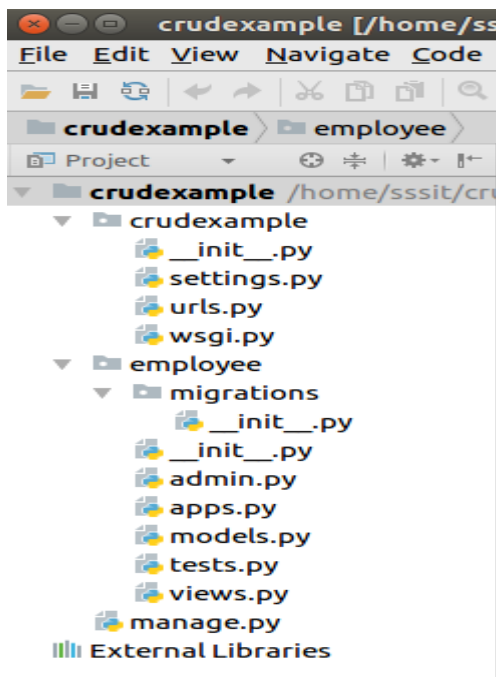
1. `$ python3 manage.py startapp employee`



```
Terminal File Edit View Search Terminal Help
root@sssit-Inspiron-15-3567:/home/sssit# django-admin startproject crudexample
root@sssit-Inspiron-15-3567:/home/sssit# cd crudexample
root@sssit-Inspiron-15-3567:/home/sssit/crudexample# python3 manage.py startapp
employee
root@sssit-Inspiron-15-3567:/home/sssit/crudexample#
```

3. Project Structure

Initially, 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
```

```
1. DATABASES = {
2.     'default': {
3.         'ENGINE': 'django.db.backends.mysql',
4.         'NAME': 'djangodb',
5.         'USER': 'root',
6.         'PASSWORD': 'mysql',
7.         'HOST': 'localhost',
8.         'PORT': '3306'
9.     }
10. }
```

5. Create a Model

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

```
// models.py
```

```
1. from django.db import models
2. class Employee(models.Model):
3.     eid = models.CharField(max_length=20)
4.     ename = models.CharField(max_length=100)
5.     email = models.EmailField()
6.     econtact = models.CharField(max_length=15)
7.     class Meta:
8.         db_table = "employee"
```

6. Create a ModelForm

```
// forms.py
```

```
1. from django import forms
2. from employee.models import Employee
3. class EmployeeForm(forms.ModelForm):
4.     class Meta:
5.         model = Employee
6.         fields = "__all__"
```

7. Create View Functions

```
// views.py
```

```
1. from django.shortcuts import render, redirect
2. from employee.forms import EmployeeForm
3. from employee.models import Employee
4. # Create your views here.
5. def emp(request):
6.     if request.method == "POST":
7.         form = EmployeeForm(request.POST)
8.         if form.is_valid():
9.             try:
10.                form.save()
11.                return redirect('/show')
12.            except:
13.                pass
14.     else:
15.         form = EmployeeForm()
16.     return render(request, 'index.html', {'form':form})
17. def show(request):
18.     employees = Employee.objects.all()
19.     return render(request, "show.html", {'employees':employees})
20. def edit(request, id):
21.     employee = Employee.objects.get(id=id)
22.     return render(request, 'edit.html', {'employee':employee})
23. def update(request, id):
24.     employee = Employee.objects.get(id=id)
25.     form = EmployeeForm(request.POST, instance = employee)
26.     if form.is_valid():
27.         form.save()
28.         return redirect("/show")
29.     return render(request, 'edit.html', {'employee': employee})
30. def destroy(request, id):
31.     employee = Employee.objects.get(id=id)
32.     employee.delete()
33.     return redirect("/show")
```

8. Provide Routing

Provide URL patterns to map with views function.

```
// urls.py
```

1. from django.contrib **import** admin
2. from django.urls **import** path
3. from employee **import** views
4. urlpatterns = [
 5. path('admin/', admin.site.urls),
 6. path('emp', views.emp),
 7. path('show', views.show),
 8. path('edit/<int:id>', views.edit),
 9. path('update/<int:id>', views.update),
 10. path('delete/<int:id>', views.destroy),
11.]

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

1. <!DOCTYPE html>
2. <html lang="en">
3. <head>
4. <meta charset="UTF-8">
5. <title>Index</title>
6. {% load staticfiles %}
7. <link rel="stylesheet" href="{% static 'css/style.css' %}"/>
8. </head>
9. <body>
10. <form method="POST" class="post-form" action="/emp">
11. <input type="hidden" value="{% csrf_token %}" />
12. <div class="container">
13.

14. <div class="form-group row">
15. <label class="col-sm-1 col-form-label"></label>
16. <div class="col-sm-4">
17. <h3>Enter Details</h3>
18. </div>
19. </div>
20. <div class="form-group row">
21. <label class="col-sm-2 col-form-label">Employee Id:</label>
22. <div class="col-sm-4">

```
23.    {{ form.eid }}
24.  </div>
25. </div>
26. <div class="form-group row">
27.  <label class="col-sm-2 col-form-label">Employee Name:</label>
28.  <div class="col-sm-4">
29.    {{ form.ename }}
30.  </div>
31. </div>
32. <div class="form-group row">
33.  <label class="col-sm-2 col-form-label">Employee Email:</label>
34.  <div class="col-sm-4">
35.    {{ form.eemail }}
36.  </div>
37. </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.    {{ form.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-primary">Submit</button>
48.  </div>
49. </div>
50. </div>
51. </form>
52. </body>
53. </html>
```

// show.html

```
1. <!DOCTYPE html>
2. <html lang="en">
3. <head>
4.   <meta charset="UTF-8">
5.   <title>Employee Records</title>
6.   {% load staticfiles %}
```

```

7. <link rel="stylesheet" href="{% static 'css/style.css' %}"/>
8. </head>
9. <body>
10. <table class="table table-striped table-bordered table-sm">
11. <thead class="thead-dark">
12. <tr>
13. <th>Employee ID</th>
14. <th>Employee Name</th>
15. <th>Employee Email</th>
16. <th>Employee Contact</th>
17. <th>Actions</th>
18. </tr>
19. </thead>
20. <tbody>
21. {% for employee in employees %}
22. <tr>
23. <td>{{ employee.id }}</td>
24. <td>{{ employee.ename }}</td>
25. <td>{{ employee.email }}</td>
26. <td>{{ employee.econtact }}</td>
27. <td>
28. <a href="/edit/{{ employee.id }}"><span class="glyphicon glyphicon-
pencil">Edit</span></a>
29. <a href="/delete/{{ employee.id }}">Delete</a>
30. </td>
31. </tr>
32. {% endfor %}
33. </tbody>
34. </table>
35. <br>
36. <br>
37. <center><a href="/emp" class="btn btn-
primary">Add New Record</a></center>
38. </body>
39. </html>

```

// edit.html

```

1. <!DOCTYPE html>
2. <html lang="en">

```

```
3. <head>
4.   <meta charset="UTF-8">
5.   <title>Index</title>
6.   { % load staticfiles % }
7.   <link rel="stylesheet" href="{ % static 'css/style.css' % }"/>
8. </head>
9. <body>
10. <form method="POST" class="post- form"
    action="/update/{{ employee.id }}">
11.   { % csrf_token % }
12.   <div class="container">
13. <br>
14.   <div class="form-group row">
15.     <label class="col-sm-1 col-form-label"></label>
16.     <div class="col-sm-4">
17.       <h3>Update Details</h3>
18.     </div>
19.   </div>
20.   <div class="form-group row">
21.     <label class="col-sm-2 col-form-label">Employee Id:</label>
22.     <div class="col-sm-4">
23.       <input type="text" name="eid" id="id_eid" required maxlength="20" va
        lue="{{ employee.eid }}" />
24.     </div>
25.   </div>
26.   <div class="form-group row">
27.     <label class="col-sm-2 col-form-label">Employee Name:</label>
28.     <div class="col-sm-4">
29.       <input type="text" name="ename" id="id_ename" required maxlength
        ="100" value="{{ employee.ename }}" />
30.     </div>
31.   </div>
32.   <div class="form-group row">
33.     <label class="col-sm-2 col-form-label">Employee Email:</label>
34.     <div class="col-sm-4">
35.       <input type="email" name="email" id="id_email" required maxlength
        ="254" value="{{ employee.email }}" />
36.     </div>
37.   </div>
```

```

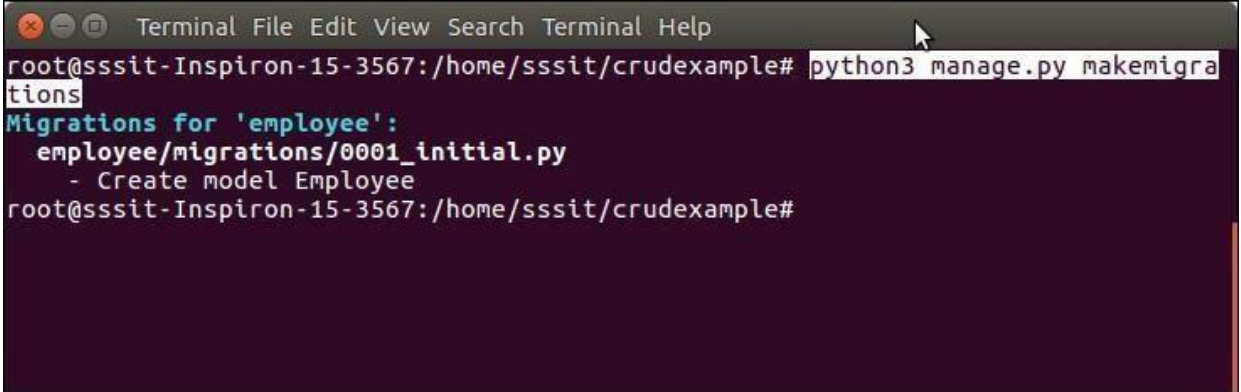
38. <div class="form-group row">
39. <label class="col-sm-2 col-form-label">Employee Contact:</label>
40. <div class="col-sm-4">
41.     <input type="text" name="econtact" id="id_econtact" required maxlen
      gth="15" value="{{ employee.econtact }}" />
42. </div>
43. </div>
44. <div class="form-group 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`



```

Terminal File Edit View Search Terminal Help
root@sssit-Inspiron-15-3567:/home/sssit/crudexample# python3 manage.py makemigrations
Migrations for 'employee':
  employee/migrations/0001_initial.py
  - Create model Employee
root@sssit-Inspiron-15-3567:/home/sssit/crudexample#

```

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

```

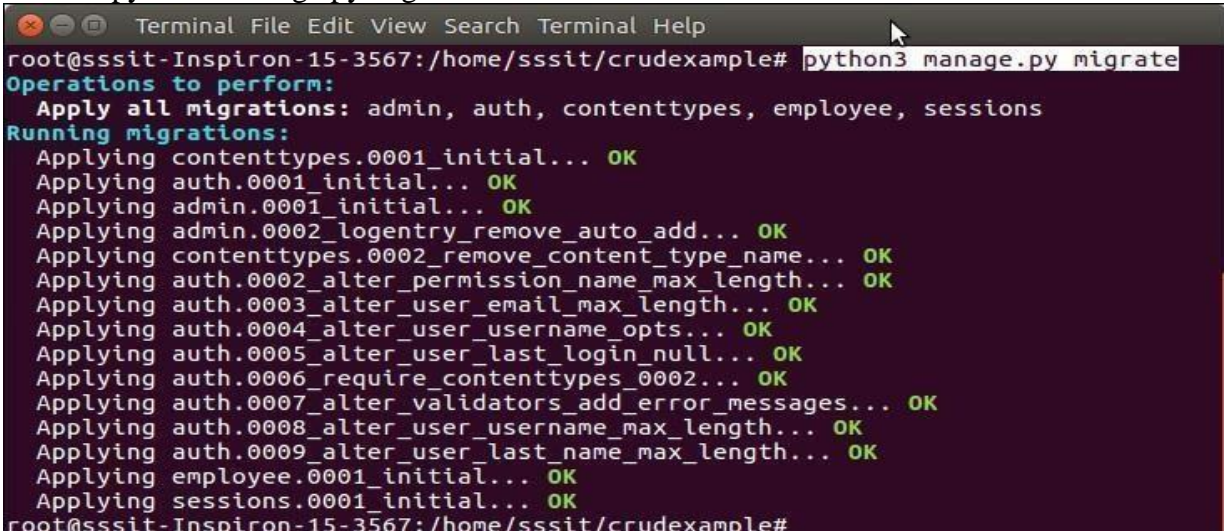
1. INSTALLED_APPS = [
2.     'django.contrib.admin',
3.     'django.contrib.auth',
4.     'django.contrib.contenttypes',

```

5. 'django.contrib.sessions',
6. 'django.contrib.messages',
7. 'django.contrib.staticfiles',
8. 'employee'
9.]

Run the command to migrate the migrations.

1. \$ python3 manage.py migrate



```
Terminal File Edit View Search Terminal Help
root@sssit-Inspiron-15-3567:/home/sssit/crudexample# python3 manage.py migrate
Operations to perform:
Apply all migrations: admin, auth, contenttypes, employee, sessions
Running migrations:
Applying contenttypes.0001_initial... OK
Applying auth.0001_initial... OK
Applying admin.0001_initial... OK
Applying admin.0002_logentry_remove_auto_add... OK
Applying contenttypes.0002_remove_content_type_name... OK
Applying auth.0002_alter_permission_name_max_length... OK
Applying auth.0003_alter_user_email_max_length... OK
Applying auth.0004_alter_user_username_opts... OK
Applying auth.0005_alter_user_last_login_null... OK
Applying auth.0006_require_contenttypes_0002... OK
Applying auth.0007_alter_validators_add_error_messages... OK
Applying auth.0008_alter_user_username_max_length... OK
Applying auth.0009_alter_user_last_name_max_length... OK
Applying employee.0001_initial... OK
Applying sessions.0001_initial... OK
root@sssit-Inspiron-15-3567:/home/sssit/crudexample#
```

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.

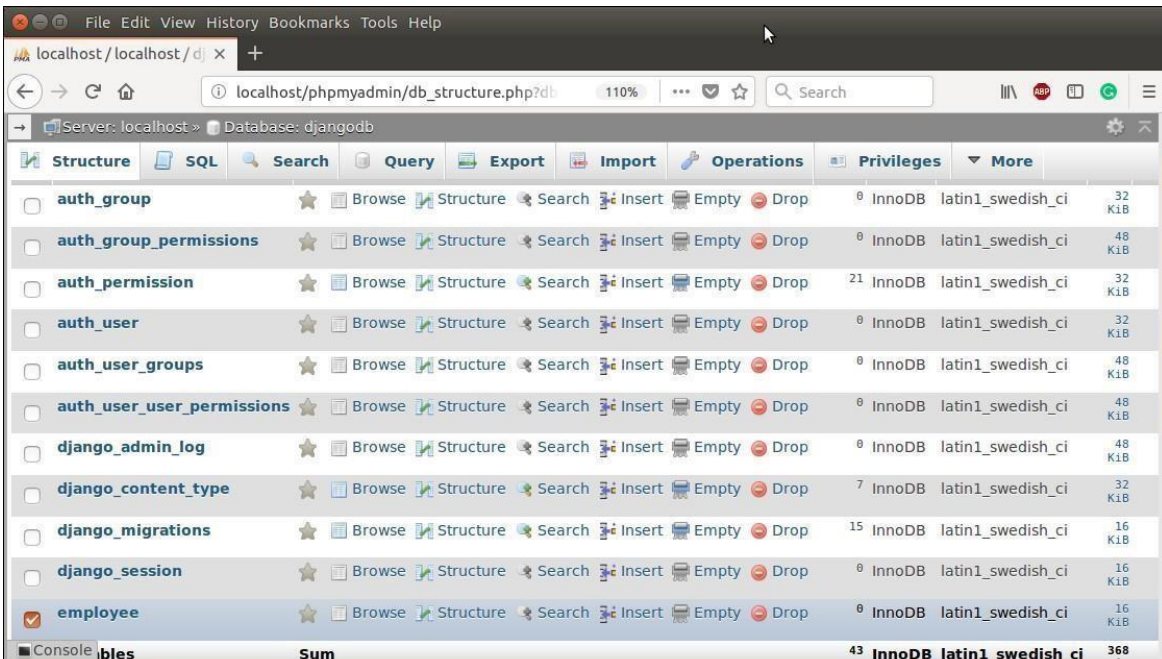
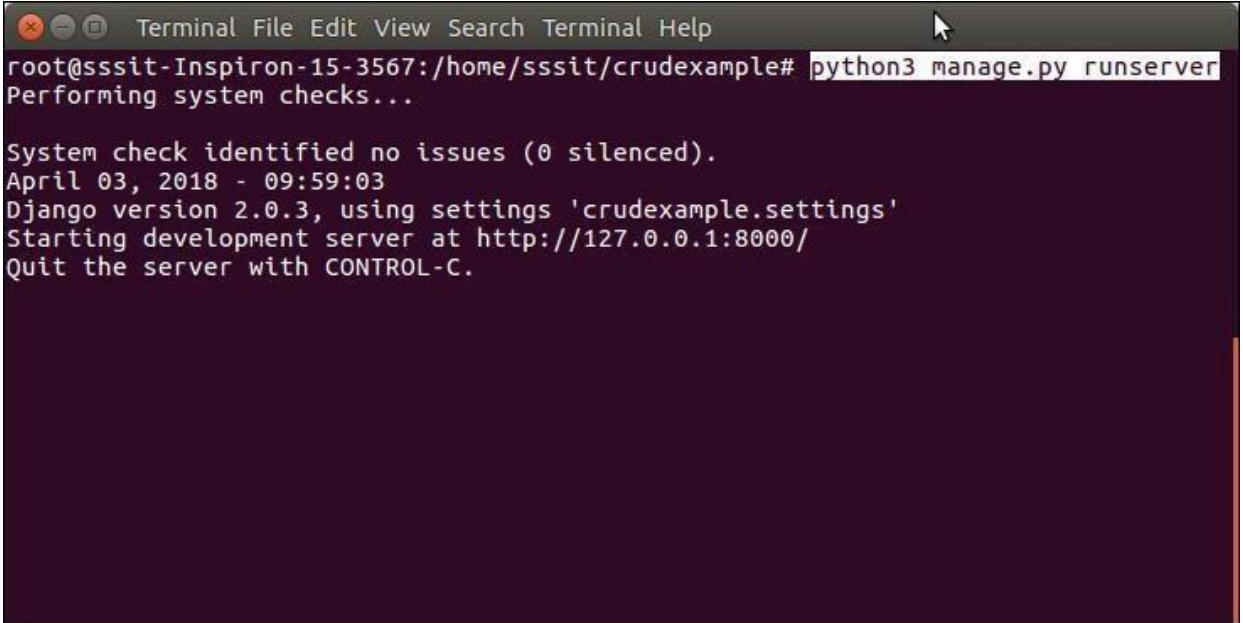


Table Name	Engine	Character Set	Collation	Row Count	Size
auth_group	InnoDB	latin1_swedish_ci		0	32 KiB
auth_group_permissions	InnoDB	latin1_swedish_ci		0	48 KiB
auth_permission	InnoDB	latin1_swedish_ci		21	32 KiB
auth_user	InnoDB	latin1_swedish_ci		0	32 KiB
auth_user_groups	InnoDB	latin1_swedish_ci		0	48 KiB
auth_user_user_permissions	InnoDB	latin1_swedish_ci		0	48 KiB
django_admin_log	InnoDB	latin1_swedish_ci		0	48 KiB
django_content_type	InnoDB	latin1_swedish_ci		7	32 KiB
django_migrations	InnoDB	latin1_swedish_ci		15	16 KiB
django_session	InnoDB	latin1_swedish_ci		0	16 KiB
employee	InnoDB	latin1_swedish_ci		0	16 KiB
Sum	43 InnoDB	latin1_swedish_ci			368

Run Server

To run server use the following command.

1. `$ python3 manage.py runserver`



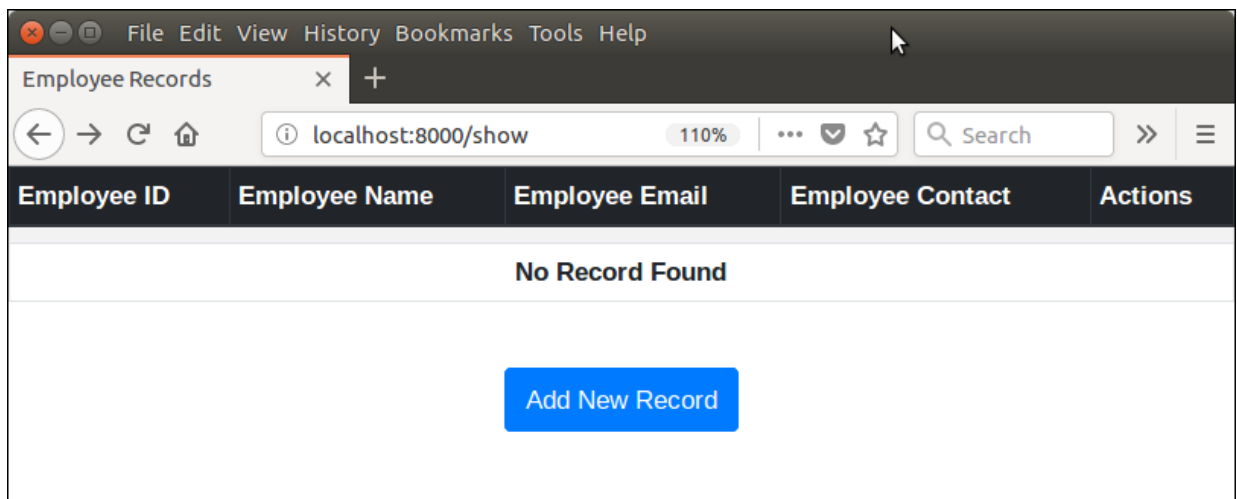
```
Terminal File Edit View Search Terminal Help
root@sssit-Inspiron-15-3567:/home/sssit/crudexample# python3 manage.py runserver
Performing system checks...

System check identified no issues (0 silenced).
April 03, 2018 - 09:59:03
Django version 2.0.3, using settings 'crudexample.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CONTROL-C.
```

Access to the Browser

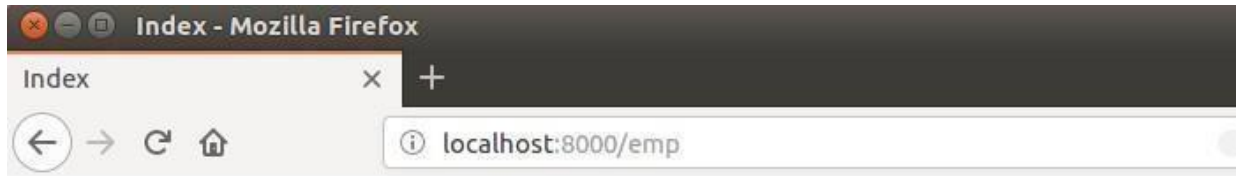
Access the application by entering **localhost:8000/show**, it will show all the available employee records.

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.



Enter Details

Employee Id:

Employee Name:

Employee Email:

Employee Contact:

Submit

Filling the details.



Enter Details

Employee Id:

1001

Employee Name:

Sohan

Employee Email:

sohan@abc.com

Employee Contact:

5326965874

Submit

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 Jacobson</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:

The screenshot shows a web browser window with the address bar containing 'C:\Documents and Settings\Administrator\Desktop\week5.xml'. The page content is titled 'BOOKS INFORMATION' and displays a table with the following data:

Title	Author	ISBN	Publisher	Edition	Price
Web programming,building internet applications	ChrisBates	0-07-049543-7	Wiley Dreamech	2nd edition	Rs.250
Computer Networks	Andrew S Tanebaum	81-203-1165-5	Pearson	4nd edition	Rs.350
Frontiers of Electronic commerce	Kalakata	978-81-265-1173-0	Pearson	1st edition	Rs.350
Java Programming with CORBA	G.Brose	978-81-265-1173-0	Wiley Dreamech	2nd edition	Rs.250
The Unified Modelling language User Guide	GradyBooch,James Rumbaugh,Ivar Jacobson	81-7758-372-7	Perarson Education	2nd edition	Rs.400
Data mining - Concepts and techniques	Jiawei HAn and Kamber	978-81-312-0538-8	Pearson	1st edition	Rs.550

7. Design a controller with servlet that provides the interaction with application developed in experiment 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)).

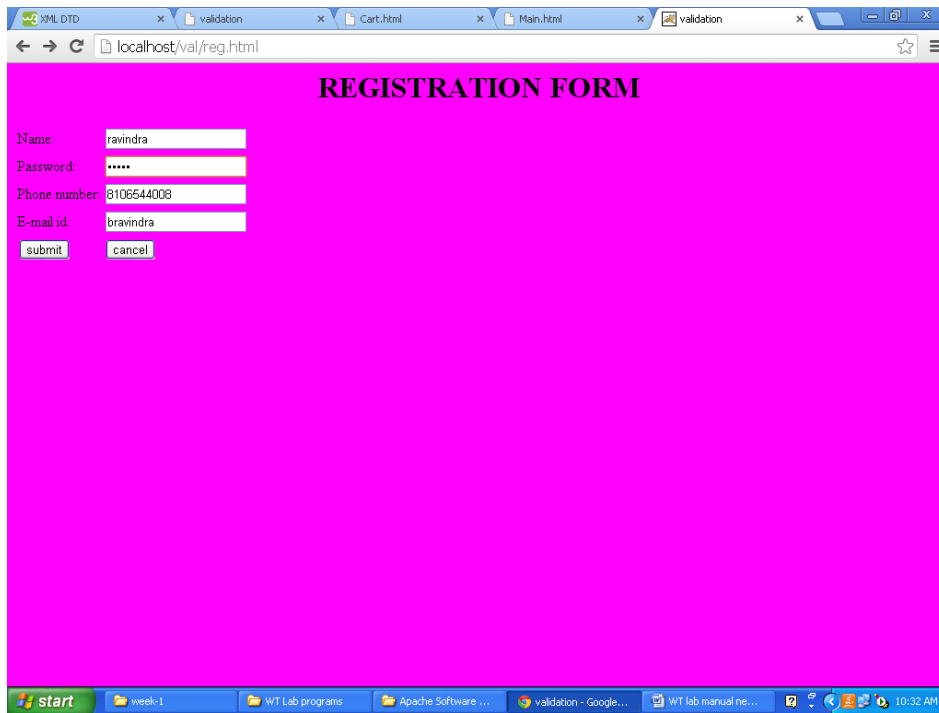
Create the Data source name.

Start->settings->control panel->Administrative Tools->Data Sources.

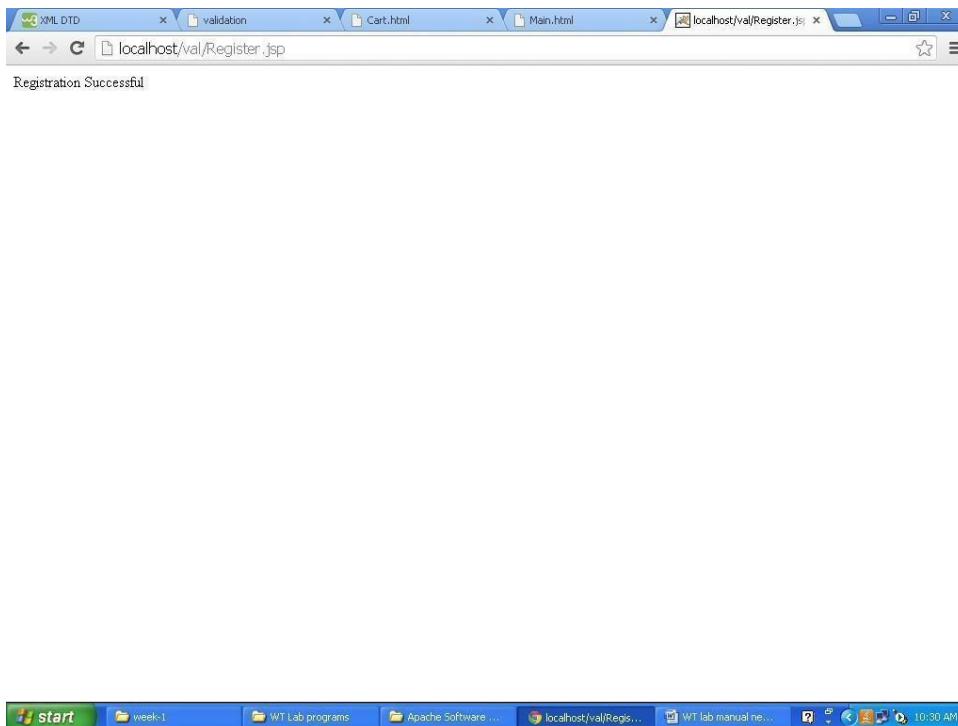
Under SystemDSN add Microsoft ODBC for Oracle.

Set Data Source name to oradsn.

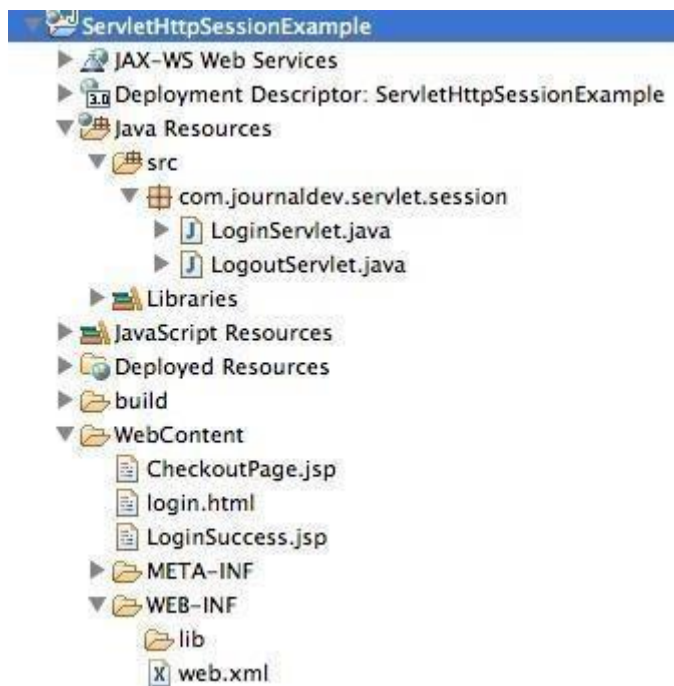
Output:1



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 {
```

```

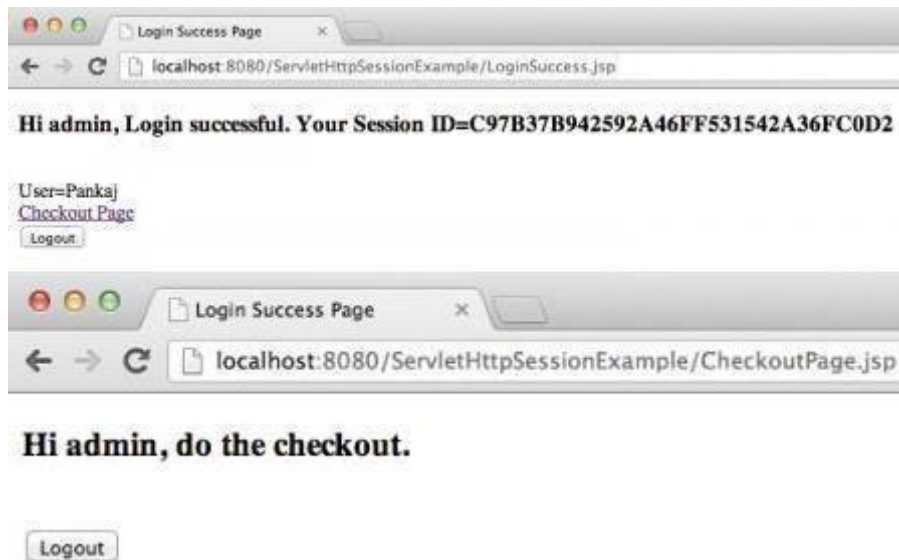
private static final long serialVersionUID = 1L;

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
```

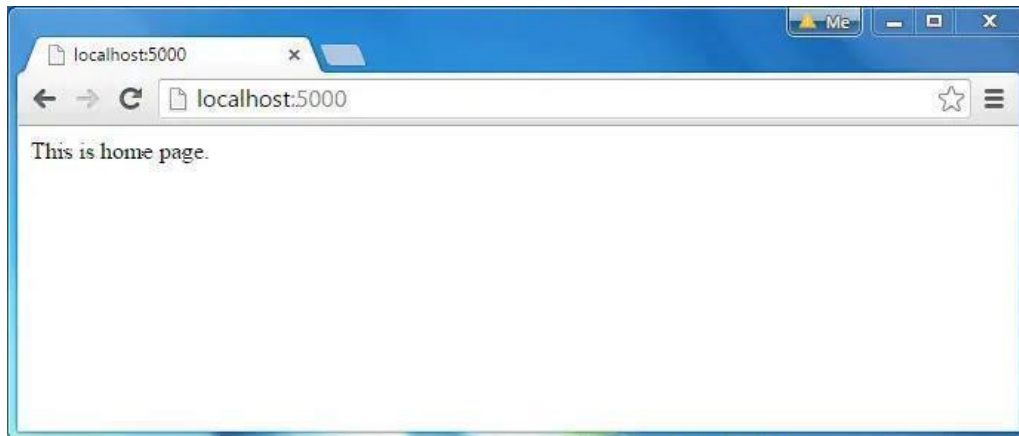
```
Content-Type: text/plain
```

```
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 CRUD operations 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);

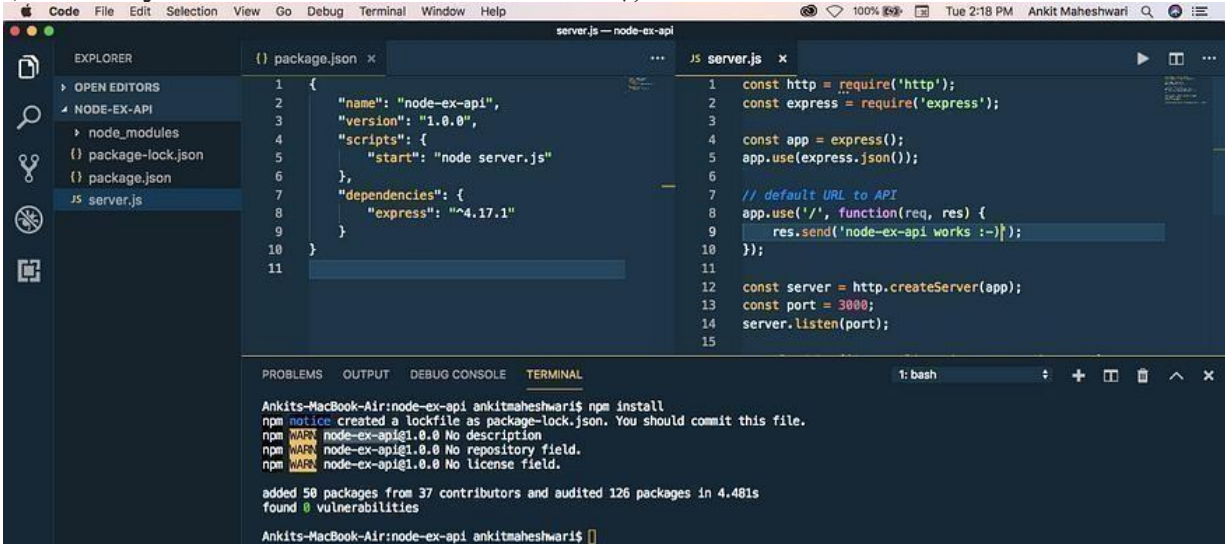
console.debug('Server listening on port ' + port);
```

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 ↓).



The screenshot shows the VS Code interface with two files open: `package.json` and `server.js`. The `package.json` file contains the following content:

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

The `server.js` file contains the code shown in the previous block. The terminal at the bottom shows the output of the `npm install` command:

```
Ankits-MacBook-Air:node-ex-api ankitmaheshwari$ npm install
npm notice created a lockfile as package-lock.json. You should commit this file.
npm WARN node-ex-api@1.0.0 No description
npm WARN node-ex-api@1.0.0 No repository field.
npm WARN node-ex-api@1.0.0 No license field.

added 50 packages from 37 contributors and audited 126 packages in 4.481s
found 0 vulnerabilities

Ankits-MacBook-Air:node-ex-api ankitmaheshwari$
```

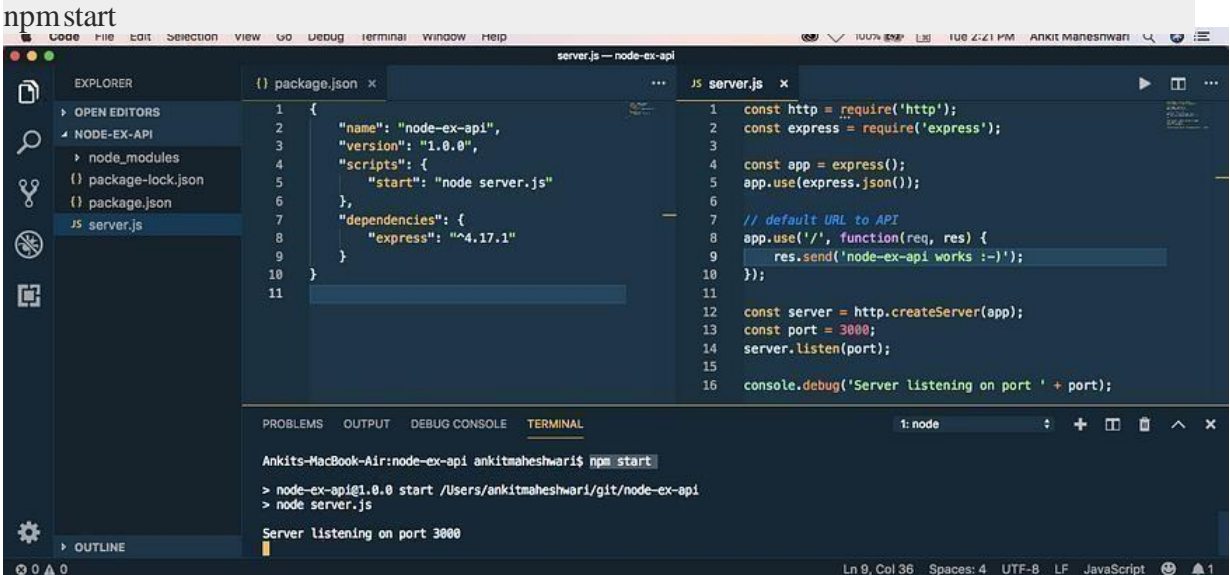
After dependency installation this will create `"node_modules"` folder at the root of the `"node-ex-api"` folder.

Run Project

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

To start a server run this command:

```
npm start
```



The screenshot shows a VS Code editor with the following content:

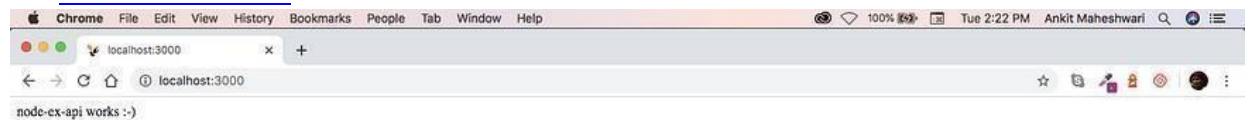
```
package.json
1 {
2   "name": "node-ex-api",
3   "version": "1.0.0",
4   "scripts": {
5     "start": "node server.js"
6   },
7   "dependencies": {
8     "express": "^4.17.1"
9   }
10 }
11
```

```
server.js
1 const http = require('http');
2 const express = require('express');
3
4 const app = express();
5 app.use(express.json());
6
7 // default URL to API
8 app.use('/', function(req, res) {
9   res.send('node-ex-api works :-)');
10 });
11
12 const server = http.createServer(app);
13 const port = 3000;
14 server.listen(port);
15
16 console.debug('Server listening on port ' + port);
```

```
Terminal
Ankits-MacBook-Air:node-ex-api ankitmaheshwari$ npm start
> node-ex-api@1.0.0 start /Users/ankitmaheshwari/git/node-ex-api
> node server.js
Server listening on port 3000
```

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**ppdate 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 **Read** we already have two end-points:

→ localhost:3000/items (Returns all objects)

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

For **Update** we will create new end-point `router.put('/',.....)`

→ localhost:3000/items (With put request)

For **Delete** 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.

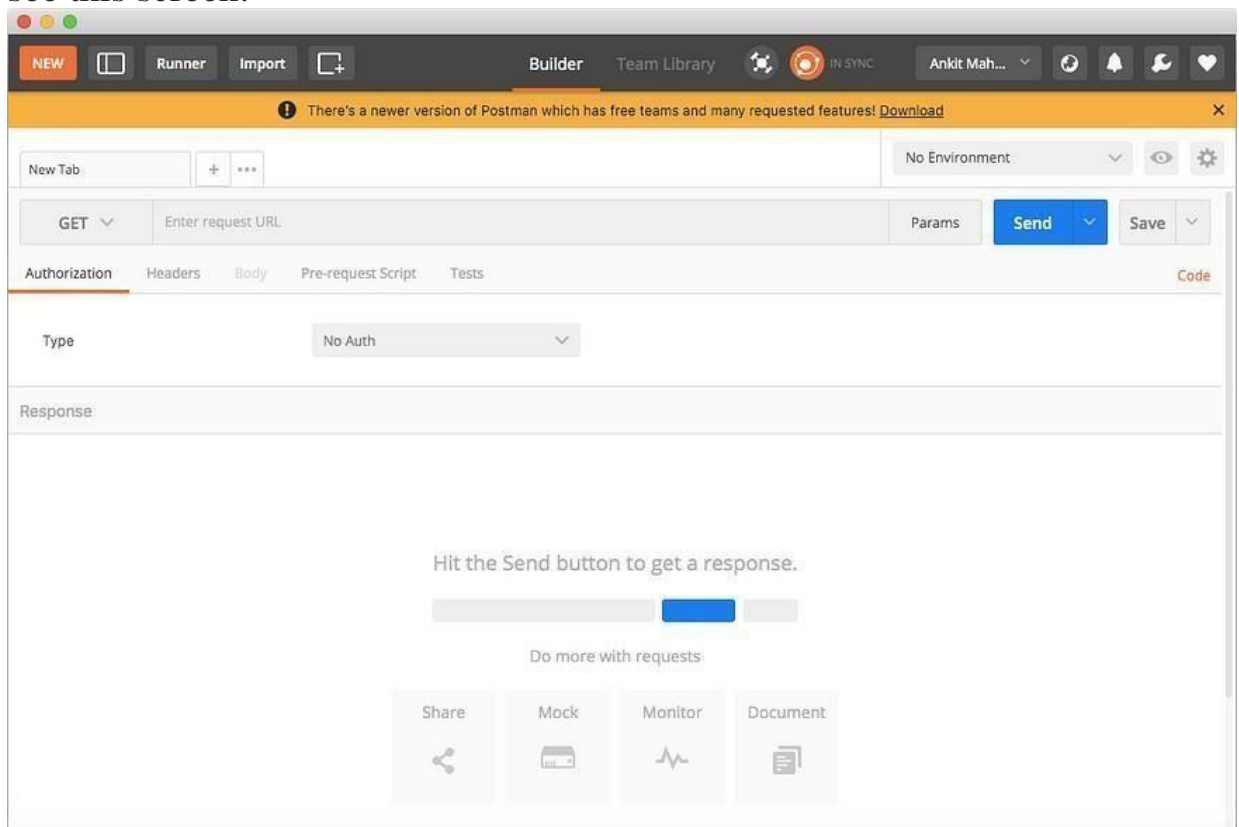
Testing API with [Postman](#).

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();

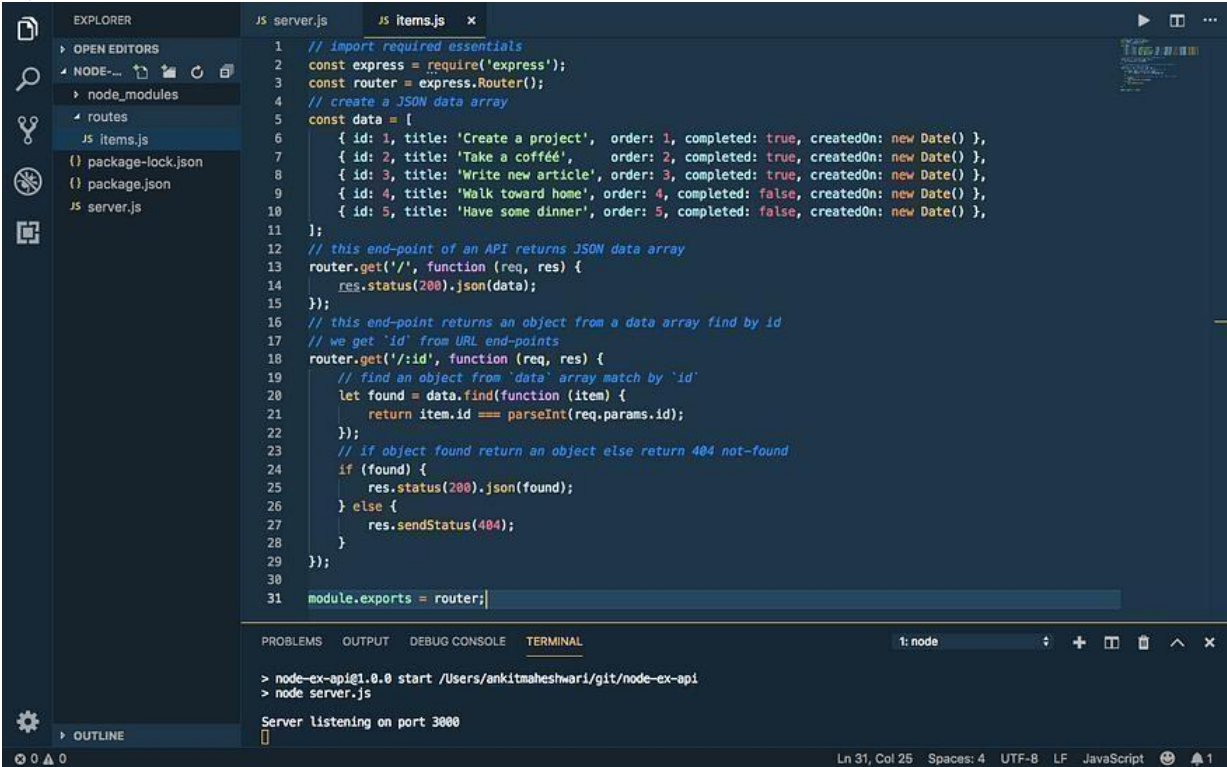
// create a JSON data array let data = [
  { id: 1, title: 'Create a project', order: 1, completed: true, createdOn: new Date() },
  { id: 2, title: 'Take a coffee', order: 2, completed: true,
createdOn: new Date() },
  { id: 3, title: 'Write new article', order: 3, completed: true, createdOn: new Date() },
  { 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 ↓↓



```
1 // import required essentials
2 const express = require('express');
3 const router = express.Router();
4 // create a JSON data array
5 const data = [
6   { id: 1, title: 'Create a project', order: 1, completed: true, createdOn: new Date() },
7   { id: 2, title: 'Take a coffée', order: 2, completed: true, createdOn: new Date() },
8   { id: 3, title: 'Write new article', order: 3, completed: true, createdOn: new Date() },
9   { id: 4, title: 'Walk toward home', order: 4, completed: false, createdOn: new Date() },
10  { id: 5, title: 'Have some dinner', order: 5, completed: false, createdOn: new Date() },
11 ];
12 // this end-point of an API returns JSON data array
13 // we get 'id' from URL end-points
14 router.get('/', function (req, res) {
15   res.status(200).json(data);
16 });
17 // this end-point returns an object from a data array find by id
18 // we get 'id' from URL end-points
19 router.get('/:id', function (req, res) {
20   // find an object from 'data' array match by 'id'
21   let found = data.find(function (item) {
22     return item.id === parseInt(req.params.id);
23   });
24   // if object found return an object else return 404 not-found
25   if (found) {
26     res.status(200).json(found);
27   } else {
28     res.sendStatus(404);
29   }
30 });
31 module.exports = router;
```

```
> node-ex-api@1.0.0 start /Users/ankitmaheshwari/git/node-ex-api
> node server.js
Server listening on port 3000
```

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 ↓↓

```

// items.js
1 // import required essentials
2 const express = require('express');
3 // create new router
4 const router = express.Router();
5 // create a JSON data array
6 const data = [
7   { id: 1, title: 'Create a project',
8     order: 1, completed: true, createdOn: new
9     Date() },
10  { id: 2, title: 'Take a coff   ',
11    order: 2, completed: true, createdOn: new
12    Date() },
13  { id: 3, title: 'Write new article',
14    order: 3, completed: true, createdOn: new
15    Date() },
16  { id: 4, title: 'Walk toward home',
17    order: 4, completed: false, createdOn:
18    new Date() },
19  { id: 5, title: 'Have some dinner',
20    order: 5, completed: false, createdOn:
21    new Date() },
22 ];
23 // this end-point of an API returns JSON data
24 // array
25 router.get('/', function (req, res) {
26   res.status(200).json(data);
27 });
28 // this end-point returns an object from a
29 // data array find by id
30 // we get 'id' from URL end-points
31 router.get('/:id', function (req, res) {
32   // ...
33 });

```

```

// server.js
1 // import required essentials
2 const http = require('http');
3 const express = require('express');
4
5 // import 'items' from 'routes' folder
6 const itemsRouter = require('./routes/items');
7
8 // create new app
9 const app = express();
10 app.use(express.json());
11
12 /* this '/items' URL will have two end-points:
13 → localhost:3000/items/ (this returns array of objects)
14 → localhost:3000/items/:id (this returns single object)
15 */
16 app.use('/items', itemsRouter);
17
18 // default URL to API
19 app.use('/', function(req, res) {
20   res.send('node-ex-api works :-)');
21 });
22
23 const server = http.createServer(app);
24 const port = 3000;
25 server.listen(port);
26 console.debug('Server listening on port ' + port);
27

```

```

Terminal
1: node
Ankits-MacBook-Air:node-ex-api ankitmaheshwari$ npm start
> node-ex-api@1.0.0 start /Users/ankitmaheshwari/git/node-ex-api
> node server.js
Server listening on port 3000

```

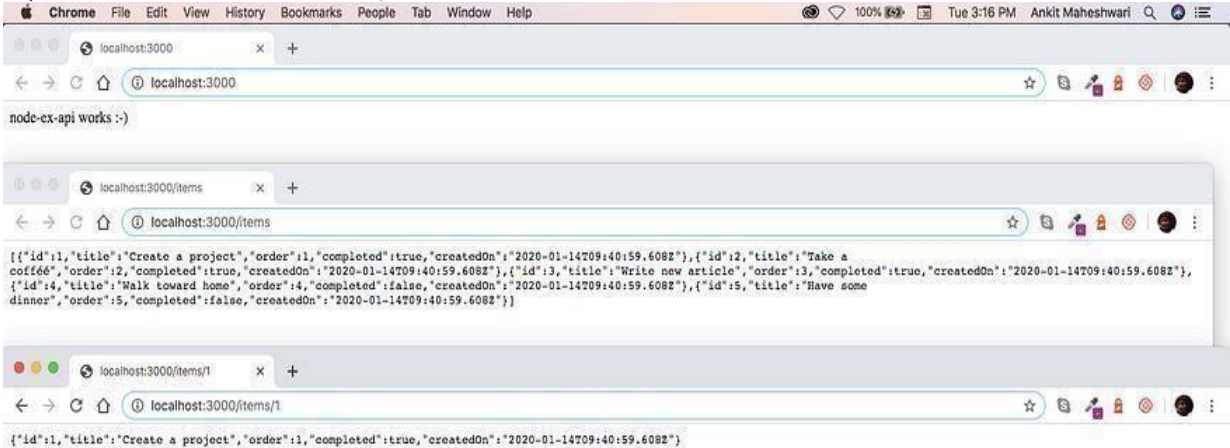
The "items.js" and "server.js" file will look like this ↑

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](https://expressjs.com/) is a fast, minimalist framework that provides several features for building web applications in Node.js. [CORS](https://github.com/expressjs/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:

```
const express = require("express");
const cors = require("cors");
const app = express();
const PORT = 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.

```
function App() {
  return (
    <div>
      <p>Hello World!</p>
    </div>
  );
}
```

```
}  
export default App;
```

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

```
@import url("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: 10px 15px;  
  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: 50px 70px;  
  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>

```

```

<Route path="/" element={ <Login/> } />
<Route path="/register" element={ <Signup/> } />
<Route path="/dashboard" element={ <Dashboard/> } />
<Route path="/phone/verify" element={ <PhoneVerify/> } />
</Routes>
</BrowserRouter>
);
}

```

export default App;

The Login page

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

```

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

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

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

  const gotoSignUpPage = () => navigate("/register");

  return (
    <div className="login _container">
      <h2>Login </h2>
      <form className="login_form" onSubmit={handleSubmit}>
        <label htmlFor="email">Email</label>
        <input type="text"
          id="email"
          name="email"
          value={email}
          required
          onChange={(e) => setEmail(e.target.value)}
        />
        <label htmlFor="password">Password</label>
        <input
          type="password"
          name="password"
          id="password"
          minLength={8}

```

```

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

export default Login;

```

Login

Email

Password

SIGN IN

Don't have an account? [Sign up](#)

The Sign up page

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

```

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

const Signup = () => {
  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;

```

Sign up

Email Address

Username

Phone Number

Password

SIGN UP

Already have an account? [Login](#)

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 using chart.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) + `°C`;

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

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

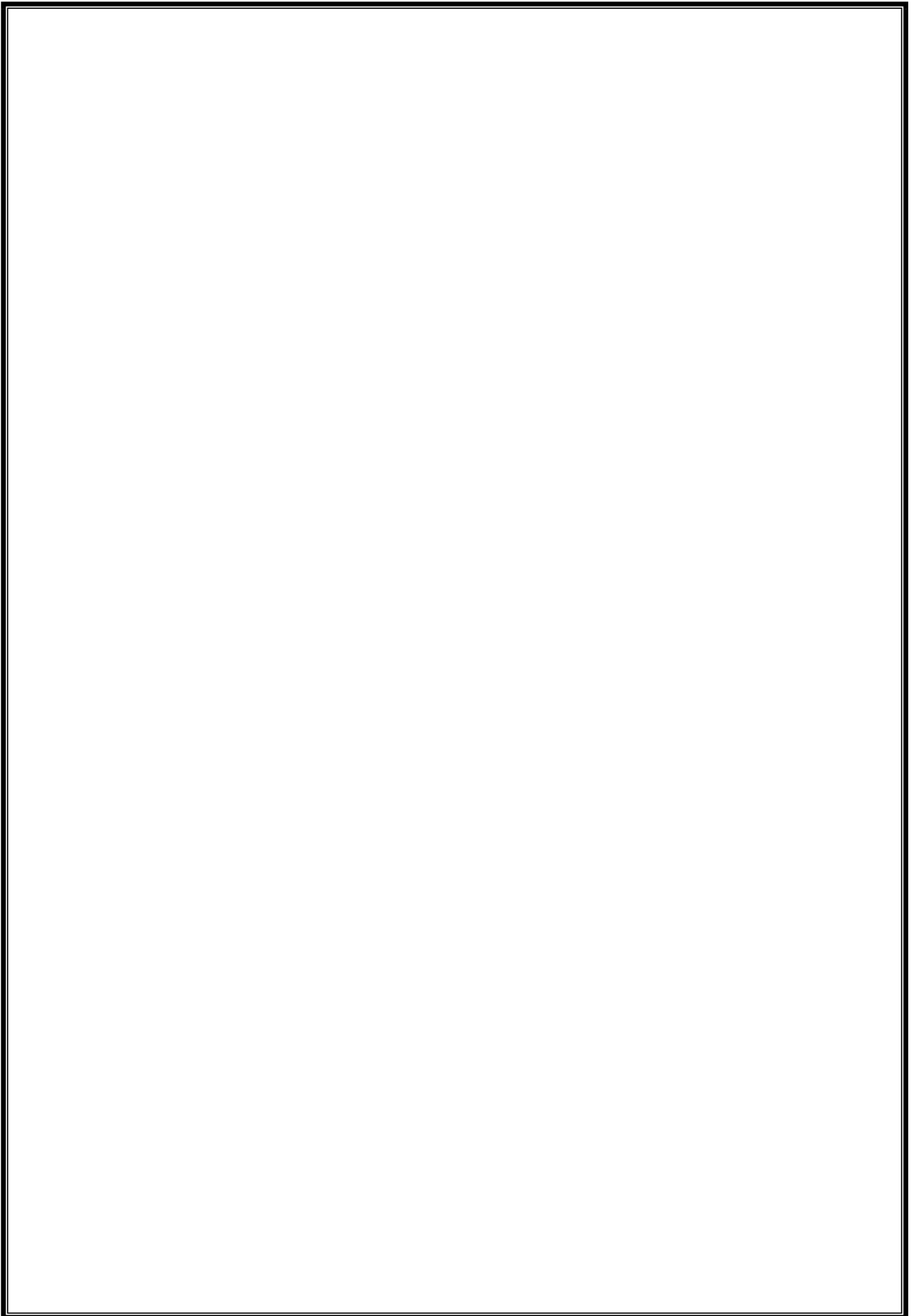
    const windSpeed = document.querySelector('#current_wind');
    windSpeed.textContent = 'Wind Speed: ' + data.wind.speed + '
mph';

    //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/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFsp3yD65Vohh
puuCOMLASjC" crossorigin="anonymous">
<link rel="stylesheet"
href="https://use.fontawesome.com/releases/v5.8.1/css/all.css"
integrity="sha384-
50oBUHEmvpQ+11W4y57PTFmhCaXp0ML5d60M1M7uH2+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>

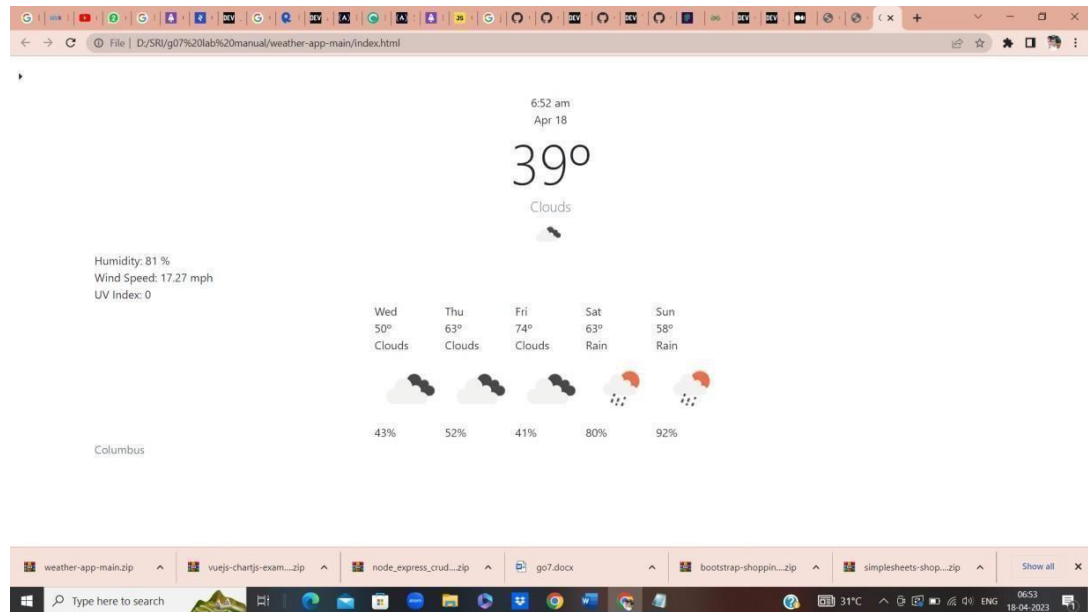
</div>
</div>
<div class="text-muted container" id="city-name">Search a City</div>
</main>

<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle.min.js" integrity="sha384-MrcW6ZMFYlzcLA8NI+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:-

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

function Todo({ todo, index, markTodo, removeTodo }) {
  return (
    <div
      className="todo"
    >
    <span style={{ 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);
}
```

```

    };

    const markTodo = index => {
    const newTodos = [...todos];
      newTodos[index].isDone = true;
      setTodos(newTodos);
    };

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

    return (
    <div className="app">
    <div className="container">
    <h1 className="text-center mb-4">Todo
    List</h1>
    <FormTodo addTodo={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>
    );
  }
  export default App;

```

