MRCET CAMPUS

MALLA REDDY COLLEGE OF ENGINEERING & TECHNOLOGY

(AUTONOMOUS INSTITUTION - UGC, GOVT. OF INDIA)

Affiliated to JNTUH; Approved by AICTE, NBA-Tier 1 & NAAC with A-GRADE | ISO 9001:2015

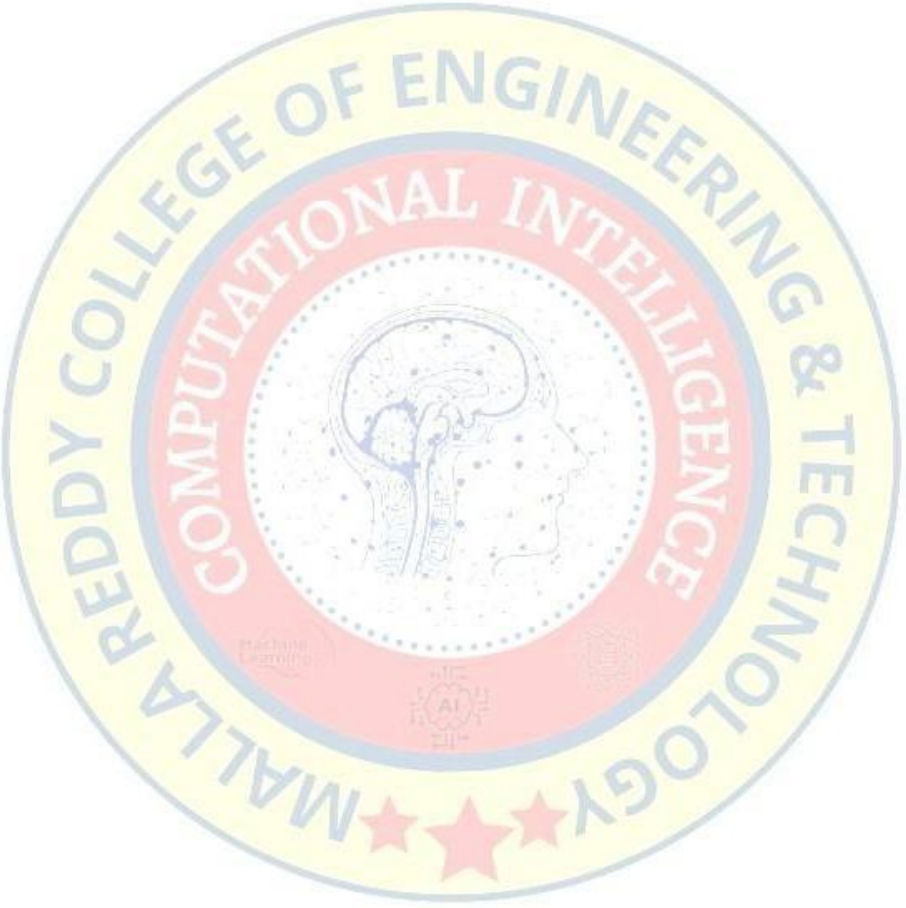
Maisammaguda, Dhulapally, Komaplly, Secunderabad - 500100, Telangana State, India

LABORATORY MANUAL & RECORD

Name:.....

Roll No:.....Branch:.....

Year:.....Sem:.....


MALLA REDDY COLLEGE OF ENGINEERING & TECHNOLOGY
COMPUTATIONAL INTELLIGENCE

MALLA REDDY COLLEGE OF ENGINEERING & TECHNOLOGY

(AUTONOMOUS INSTITUTION - UGC, GOVT. OF INDIA)

Affiliated to JNTUH; Approved by AICTE, NBA-Tier 1 & NAAC with A-GRADE | ISO 9001:2015

Maisammaguda, Dhulapally, Komaplly, Secunderabad - 500100, Telangana State, India

Certificate

Certified that this is the Bonafide Record of the Work Done by

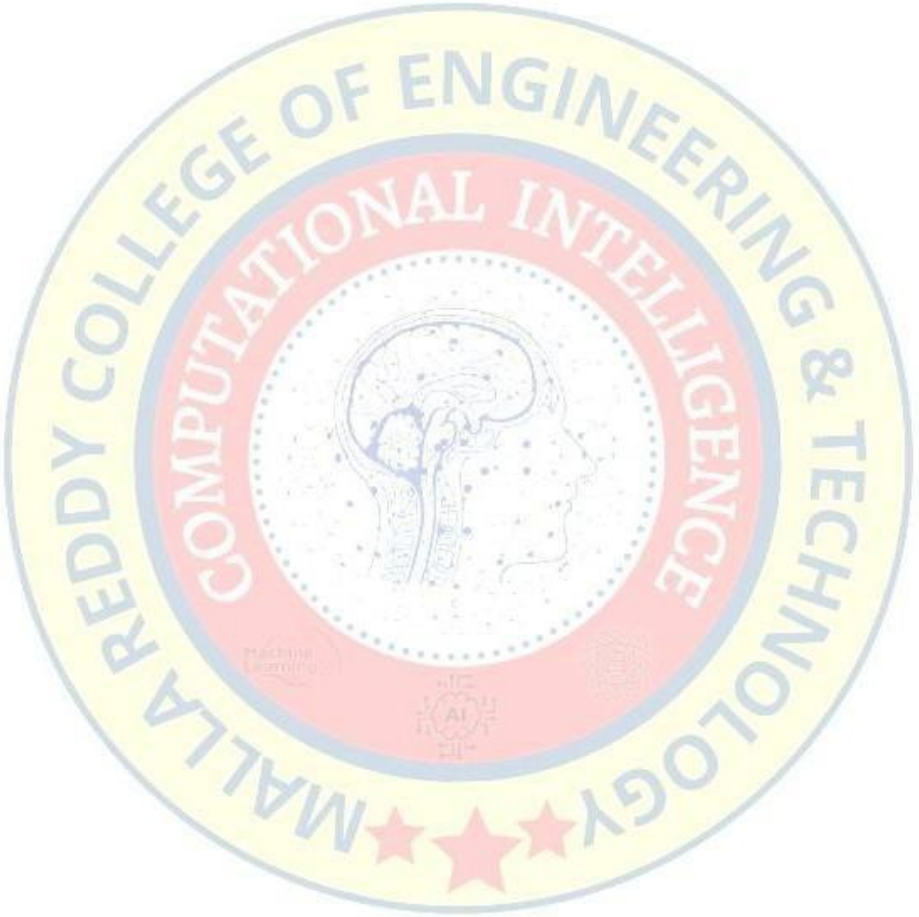
Mr./Ms.....RoII.No.....of

B.Tech.....year..... Semester for Academic year.....

in.....Laboratory.

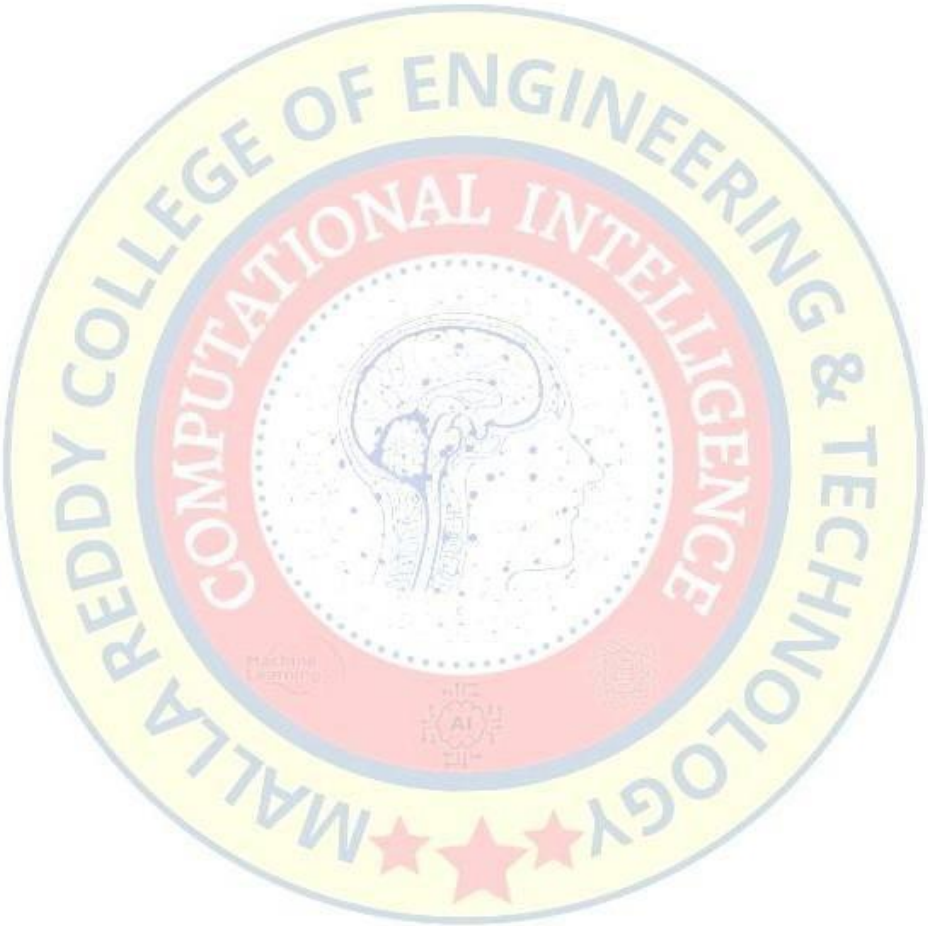
Date: Faculty Incharge HOD

Internal Examiner External Examiner


MALLA REDDY COLLEGE OF ENGINEERING & TECHNOLOGY
COMPUTATIONAL INTELLIGENCE

INDEX

S.No	Date	Name of the Activity/Experiment	Grade/ Marks	Faculty Signature


MALLA REDDY COLLEGE OF ENGINEERING & TECHNOLOGY
COMPUTATIONAL INTELLIGENCE

FULL STACK DEVELOPMENT

LAB MANUAL

(R20A0589)

B.TECH

(III YEAR – II SEM)
(2023-24)

DEPARTMENT OF COMPUTATIONAL INTELLIGENCE
(CSE-AIML, AIML & AIDS)

MALLA REDDY COLLEGE OF ENGINEERING & TECHNOLOGY
(Autonomous Institution – UGC, Govt. of India)

Recognized under 2(f) and 12 (B) of UGC ACT 1956

(Affiliated to JNTUH, Hyderabad, Approved by AICTE - Accredited by NBA & NAAC – 'A' Grade - ISO 9001:2015 Certified)

Maisammaguda, Dhulapally (Post Via. Hakimpet), Secunderabad – 500100, Telangana State, India

PROGRAMME EDUCATIONAL OBJECTIVES (PEOs)

PEO1 – PROFESSIONALISM &CITIZENSHIP

To create and sustain a community of learning in which students acquire knowledge and learn to apply it professionally with due consideration for ethical, ecological and economic issues.

PEO2 – TECHNICAL ACCOMPLISHMENTS

To provide knowledge based services to satisfy the needs of society and the industry by providing hands on experience in various technologies in core field.

PEO3 – INVENTION, INNOVATION AND CREATIVITY

To make the students to design, experiment, analyze, interpret in the core field with the help of other multi-disciplinary concepts wherever applicable.

PEO4 – PROFESSIONAL ETHICS

To educate the students to disseminate research findings with good soft skills and become a successful entrepreneur.

PEO5 – HUMAN RESOURCE DEVELOPMENT

To graduate the students in building national capabilities in technology, education and research.

PROGRAM OUTCOMES (POs)

Engineering Graduates should possess the following:

1. **Engineering knowledge**: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **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.
3. **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.
4. **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.
5. **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.
6. **The engineer and society**: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilitiesrelevant to the professional engineering practice.
7. **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.
8. **Ethics**: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. **Individual and team work**: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. **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.
11. **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 multi disciplinary environments.
12. **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.

MALLA REDDY COLLEGE OF ENGINEERING & TECHNOLOGY

Maisammaguda, Dhulapally Post, Via Hakimpet, Secunderabad – 500100

DEPARTMENT OF COMPUTATIONAL INTELLIGENCE
(CSE-AIML, AIML & AIDS)

GENERAL LABORATORY INSTRUCTIONS

1. Students are advised to come to the laboratory at least 5 minutes before (to the starting time), those who come after 5 minutes will not be allowed into the lab.
2. Plan your task properly much before to the commencement, come prepared to the lab with the synopsis / program / experiment details.
3. Student should enter into the laboratory with:

a. Laboratory observation notes with all the details (Problem statement, Aim, Algorithm, Procedure, Program, Expected Output, etc.,) filled in for the lab session.

b. Laboratory Record updated up to the last session experiments and other utensils (if any) needed in the lab.

c. Proper Dress code and Identity card.

4. Sign in the laboratory login register, write the TIME-IN, and occupy the computer system allotted to you by the faculty.
5. Execute your task in the laboratory, and record the results / output in the lab observation note book, and get certified by the concerned faculty.
6. All the students should be polite and cooperative with the laboratory staff, must maintain the discipline and decency in the laboratory.
7. Computer labs are established with sophisticated and high end branded systems, which should be utilized properly.
8. Students / Faculty must keep their mobile phones in SWITCHED OFF mode during the lab sessions. Misuse of the equipment, misbehaviors with the staff and systems etc., will attract severe punishment.
9. Students must take the permission of the faculty in case of any urgency to go out ; if anybody found loitering outside the lab / class without permission during working hours will be treated seriously and punished appropriately.
10. Students should LOG OFF/ SHUT DOWN the computer system before he/she leaves the lab after completing the task (experiment) in all aspects. He/she must ensure the system / seat is kept properly

HEAD OF THE DEPARTMENT **PRINCIPAL**

INDEX

S.NO	LIST OF PROGRAMS	PAGE NO.
WEEK-1	Write a program to create a simple webpage using HTML.	
WEEK-2	Write a program to create a website using HTML CSS and JavaScript	
WEEK-3	Write a program to build a Chat module using HTML CSS and JavaScript	
WEEK-4	Write a program to create a simple calculator Application using React JS	
WEEK-5	Write a program to create a voting application using React JS	
WEEK-6	Write a program to create and Build a Password Strength Check using Jquery	
WEEK-7	Write a program to create and Build a star rating system using Jquery	
WEEK-8	Create a Simple Login form using React JS	
WEEK-9	Create a blog using React JS	
WEEK-10	Create a project on Grocery delivery application	
WEEK-11	Connecting our TODO React js Project with Firebase	

Week-1

1. Write a program to create a simple webpage using HTML.

Program Code:

```
<!DOCTYPE html>
<html>
   <head>
      <title>My web page</title>
   </head>
   <body>
      <h1>Hello, world!</h1>
      <p>This is my first web page.</p>
      <p>It contains a
          <strong>main heading</strong> and <em> paragraph </em>.
      </p>
   </body>
</html>
```

Output:

EXERCISE PROGRAM:

Week-2

2. Write a program to create a website using HTML CSS and JavaScript?

Program Code:

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

<head>
<meta charset="UTF-8">
<meta http-equiv="X-UA-Compatible" content="IE=edge">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Simple HTML HomePage</title>
<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/5.15.3/css/all.min.css">
<style>@import url("https://fonts.googleapis.com/css2?family=Sriracha&display=swap");

body {
  margin: 0;
  box-sizing: border-box;
}

/* CSS for header */
.header {
  display: flex;
  justify-content: space-between;
  align-items: center;
  background-color: #f5f5f5;
}

.header .logo {
  font-size: 25px;
  font-family: "Sriracha", cursive;
color: #000;
  text-decoration: none;
  margin-left: 30px;
}

.nav-items {
  display: flex;
  justify-content: space-around;
  align-items: center;
  background-color: #f5f5f5;
  margin-right: 20px;
}

.nav-items a {
  text-decoration: none;
color: #000;
  padding: 35px 20px;
}

/* CSS for main element */
.intro {
```

```
  display: flex;
  flex-direction: column;

  justify-content: center;
  align-items: center;

  width: 100%;
  height: 520px;
  background: linear-gradient(to bottom, rgba(0, 0, 0, 0.5) 0%, rgba(0, 0, 0, 0.5) 100%),
url("https://images.unsplash.com/photo-1587620962725-abab7fe55159?ixlib=rb-
1.2.1&ixid=MnwxMjA3fDB8MHxwaG90by1wYWdlfHx8fGVufDB8fHx8&auto=format&fit=crop&w=1031&q=80
");
  background-size: cover;
  background-position: center;
  background-repeat: no-repeat;
}

.intro h1 {
  font-family: sans-serif;
  font-size: 60px;
color: #fff;
  font-weight: bold;
  text-transform: uppercase;
  margin: 0;
}

.intro p {
  font-size: 20px;
color: #d1d1d1;
  text-transform: uppercase;
  margin: 20px 0;
}

.intro button {
  background-color: #5edaf0;
color: #000;
  padding: 10px 25px;
  border: none;
  border-radius: 5px;
  font-size: 20px;
  font-weight: bold;
  cursor: pointer;
  box-shadow: 0px 0px 20px rgba(255, 255, 255, 0.4)
}

.achievements {
  display: flex;
  justify-content: space-around;
  align-items: center;
  padding: 40px 80px;
}

.achievements .work {
  display: flex;
```

```
  flex-direction: column;
  justify-content: center;
  align-items: center;
  padding: 0 40px;

}
.achievements .work i {

  width: fit-content;
  font-size: 50px;

color: #333333;
  border-radius: 50%;
  border: 2px solid #333333;
  padding: 12px;
}

.achievements .work .work-heading {
  font-size: 20px;
color: #333333;
  text-transform: uppercase;
  margin: 10px 0;
}

.achievements .work .work-text {
  font-size: 15px;
color: #585858;
  margin: 10px 0;
}

.about-me {
  display: flex;
  justify-content: center;
  align-items: center;
  padding: 40px 80px;
  border-top: 2px solid #eeeeee;
}

.about-me img {
  width: 500px;
  max-width: 100%;
  height: auto;
  border-radius: 10px;
}

.about-me-text h2 {
font-size: 30px;
color: #333333;
  text-transform: uppercase;
  margin: 0;
}

.about-me-text p {
font-size: 15px;
color: #585858;
```

```
  margin: 10px 0;
}

/* CSS for footer */
.footer {


  display: flex;
  justify-content: space-between;
  align-items: center;
  background-color: #302f49;

  padding: 40px 80px;
}


.footer .copy {
color: #fff;
}

.bottom-links {
  display: flex;
  justify-content: space-around;
  align-items: center;
  padding: 40px 0;
}

.bottom-links .links {
  display: flex;
  flex-direction: column;
  justify-content: center;
  align-items: center;
  padding: 0 40px;
}

.bottom-links .links span {
  font-size: 20px;
color: #fff;
  text-transform: uppercase;
  margin: 10px 0;
}

.bottom-links .links a {
text-decoration: none;
color: #a1a1a1;
padding: 10px 20px;
}</style></head>

<body>
<header class="header">
<a href="#" class="logo">Developer</a>
<nav class="nav-items">
<a href="#">Home</a>
<a href="#">About</a>
<a href="#">Contact</a>
```

```
</nav>
</header>
<main>
<div class="intro">
<h1>A Web Developer</h1>
<p>I am a web developer and I love to create websites.</p>
<button>Learn More</button>
</div>
<div class="achievements">
<div class="work">
<i class="fas fa-atom"></i>
<p class="work-heading">Projects</p>
<p class="work-text">I have worked on many projects and I am very proud of them. I am a very good developer and I am always looking for new projects.</p>
</div>
<div class="work">
<i class="fas fa-skiing"></i>
<p class="work-heading">Skills</p>
<p class="work-text">I have a lot of skills and I am very good at them. I am very good at programming and I am always looking for new skills.</p>
</div>
<div class="work">
<i class="fas fa-ethernet"></i>
<p class="work-heading">Network</p>
<p class="work-text">I have a lot of network skills and I am very good at them. I am very good at networking and I am always looking for new network skills.</p>
</div>
</div>
<div class="about-me">
<div class="about-me-text">
<h2>About Me</h2>
<p>I am a web developer and I love to create websites. I am a very good developer and I am always looking for new projects. I am a very good developer and I am always looking for new projects.</p>
</div>
<imgsrc="" alt="me">
</div>
</main>
<footer class="footer">
<div class="copy">© 2022 Developer</div>
<div class="bottom-links">
<div class="links">
<span>More Info</span>
<a href="#">Home</a>
<a href="#">About</a>
<a href="#">Contact</a>
</div>
<div class="links">
<span>Social Links</span>
<a href="#"><i class="fab fa-facebook"></i></a>
<a href="#"><i class="fab fa-twitter"></i></a>
<a href="#"><i class="fab fa-instagram"></i></a>
</div>
</div>
</footer>
<script></script></body>
</html>
```

Output:

EXERCISE PROGRAM:

Week-3

3. Write a program to build a Chat module using HTML CSS and JavaScript

Program code

HTML Code For Chatbot

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8" />
<meta http-equiv="X-UA-Compatible" content="IE=edge" />
<meta name="viewport" content="width=device-width, initial-scale=1.0" />
<title>Chatbot Javascript</title>
<link rel="stylesheet" href="style.css" />
</head>
<body>
<div class="glass">
<h1>Ask Your Question??</h1>
<h2>Yeah,ask Some Question</h2>
<div class="input">
<input
type="text"
id="userBox"
onkeydown="if(event.keyCode == 13){ talk()}"
placeholder="Type your Question"
/>
</div>
<p id="chatLog">Answer Appering Hear</p>
</div>
<script src="app.js"></script>
</body>
</html>
```

Adding the structure for Chatbot

1. Using the div tag, we will create a container for our chatbot.
2. Inside the div tag, using the h1> tag, we will add the heading of our chatbot, and then using the input tag with type text, we will create an input column for the user so that they can ask questions to the chatbot.
3. Then we will create a section using the div tag in which the answer will be displayed.
4. There is all the HTML code for the Chatbot. Now, you can see an output without CSS and JavaScript. then we write Css and JavaScript for the Chatbot.

CSS Code For Chatbot

```
* {
margin: 0;
padding: 0;
box-sizing: border-box;
}
body {
width: 100vw;
height: 100vh;
font-family: sans-serif;
```

```
padding: 10em 10em;
background: url(./bg.jpg);
opacity: 0.5;
background-position: center;
background-repeat: no-repeat;
background-position: 100% 20%;
background-size: cover;
display: flex;
align-items: center;
justify-content: space-between;
}
.glass {
width: 500px;
height: 400px;
background-color: rgba(255, 255, 255, 0.1);
padding: 50px;
color: #000;
border-radius: 9px;
backdrop-filter: blur(50px);
border: 2px solid transparent;
background-clip: padding-box;
box-shadow: 10px 10px10pxrgba(45, 55, 68, 0.3);
line-height: 1.5;
transform: translatey(-5%);
transition: transform 0.5s;
}
.glass-1 {
width: 500px;
height: 400px;
background-color: rgba(255, 255, 255, 0.1);
padding: 50px;
color: rgb(122, 82, 82);
border-radius: 9px;
backdrop-filter: blur(50px);
border: 2px solid transparent;
background-clip: padding-box;
box-shadow: 10px 10px10pxrgba(45, 55, 68, 0.3);
line-height: 1.5;
transform: translatey(-5%);
transition: transform 0.5s;
font-size: 1.7rem;
}
.glass h1 {
font-size: 1.5rem;
text-align: center;
}
.glass h2 {
font-size: 1rem;
margin-top: 20px;
}
.input {
width: 100%;
height: 70px;
overflow: hidden;
margin-top: 40px;
}
```

```
.input input {
width: 100%;
height: 70px;
border: none;
padding-left: 30px;
padding-top: 0;
outline: none;
font-size: 1.5rem;
border-radius: 20px;
}
.glass p {
font-size: 1.6rem;
margin-top: 30px;
}
```

Javascript Code For Chatbot

```
function talk(){
var know = {
"Who are you" : "Hello, Codewith_random here ",
"How are you" : "Good :)",
"What can i do for you" : "Please Give us A Follow & Like.",
"Your followers" : "I have my family of 5000 members, i don't have follower ,have supportive Famiy ",
"ok" : "Thank You So Much ",
"Bye" : "Okay! Will meet soon.."
};
var user = document.getElementById('userBox').value;
document.getElementById('chatLog').innerHTML = user + "<br>";
if (user in know) {
document.getElementById('chatLog').innerHTML = know[user] + "<br>";
}else{
document.getElementById('chatLog').innerHTML = "Sorry,I didn't understand <br>";
}
}
```

Output:

EXERCISE PROGRAM:

Week-4

4. Write a program to create a simple calculator Application using React JS

Program Code:

```
class App extends Component {
constructor() {
super()
this.state = { operations: [] }
}
.....
}
render() {
   return (
<div className="App">
<Display data={this.state.operations} />
<Buttons>
<Button onClick={this.handleClick} label="C" value="clear" />
<Button onClick={this.handleClick} label="7" value="7" />
<Button onClick={this.handleClick} label="4" value="4" />
<Button onClick={this.handleClick} label="1" value="1" />
<Button onClick={this.handleClick} label="0" value="0" /><Button onClick={this.handleClick} label="/" value="/" />
<Button onClick={this.handleClick} label="8" value="8" />
<Button onClick={this.handleClick} label="5" value="5" />
<Button onClick={this.handleClick} label="2" value="2" />
<Button onClick={this.handleClick} label="." value="." /><Button onClick={this.handleClick} label="x" value="*" />
<Button onClick={this.handleClick} label="9" value="9" />
<Button onClick={this.handleClick} label="6" value="6" />
<Button onClick={this.handleClick} label="3" value="3" />
<Button label="" value="null" /><Button onClick={this.handleClick} label="-" value="-" />
<Button onClick={this.handleClick} label="+" size="2" value="+" />
<Button onClick={this.handleClick} label="=" size="2" value="equal" />
</Buttons>
</div>
   )
class Buttons extends Component {
render() {
return <div className="Buttons"> {this.props.children} </div>
}
}class Button extends Component {
  render() {
    return (
<div
onClick={this.props.onClick}
className="Button"
      data-size={this.props.size}
      data-value={this.props.value}
>
      {this.props.label}
</div>
    )
  }
}
class Display extends Component {
```

```
  render() {
    const string = this.props.data.join('')
    return <div className="Display"> {string} </div>
  }
}

handleClick = e => {
    const value = e.target.getAttribute('data-value')
    switch (value) {
      case 'clear':
this.setState({
          operations: [],
        })
        break
      case 'equal':
this.calculateOperations()
        break
      default:
        const newOperations = update(this.state.operations, {
          $push: [value],
        })
this.setState({
          operations: newOperations,
        })
        break
    }

calculateOperations = () => {
    let result = this.state.operations.join('')
    if (result) {
      result = math.eval(result)
      result = math.format(result, { precision: 14 })
      result = String(result)
this.setState({
        operations: [result],
      })
    }
  }
```

Output:

Exercise Program

Week-5

5. Write a program to create a voting application using React JS

Program Code:

Simple Vote Application

To create the example project for this example, open command prompt, navigate to a convenient location, and run the command as shown below :

create-react-app example3

Now just replace the placeholder content of App.js and App.css with given below content :
src\App.js

```
import React,{Component} from 'react';
import './App.css';
class App extends Component{
constructor(props){
super(props);
this.state = {
languages : [

{name: "Php", votes: 0},

{name: "Python", votes: 0},
{name: "Go", votes: 0},
{name: "Java", votes: 0}
]
}
}
vote (i) {
let newLanguages = [...this.state.languages];
newLanguages[i].votes++;
function swap(array, i, j) {
var temp = array[i];
array[i] = array[j];
array[j] = temp;
}
this.setState({languages: newLanguages});
        }
render(){
return(
<>
<h1>Vote Your Language!</h1>
<div className="languages">
{
this.state.languages.map((lang, i) =>
<div key={i} className="language">
<div className="voteCount">
{lang.votes}
</div>
```

```
<div className="languageName">
{lang.name}
</div>
<button onClick={this.vote.bind(this, i)}>Click Here</button>
</div>
)
}
</div>
</>
);
}
}
export default App;
```

In React we need to create a constructor function for our classes that extend React classes. We are passing an argument into super(props) called "props" this will also be available in the Component constructor class. The super() function will make this available in the constructor method.
src\App.css

```
*{
  margin: 0;
  padding: 0;
}

body {
  text-align: center;
  color: #222;

  font-size: 24px;
  font-family: sans-serif;
}

h1 {
  margin: 30px;
}

.languages {
  height: 400px;
  width: 400px;
  margin: 10px auto;
  display: flex;
  flex-direction: column;
}

.language {
  flex: 1;
  display: flex;
  justify-content: space-between;
  align-items: center;
  padding: 10px 40px;
  background-color: blanchedalmond;
  border: 1px solid #222;
  margin: 2px;

}
```

```
.voteCount {
  border-radius: 50%;
  display: flex;

  justify-content: center;
  align-items: center;
}

.language button {
  color: green;
  background-color: #0000;
  border: none;
  font-size: 30px;
  outline: none;
  cursor: pointer;
}
```

Output:

EXERCISE PROGRAM:

Week-6

6. Write a program to create and Build a Password Strength Check using Jquery.

Program Code:

HTML File – passwordscheck.html

```
<!DOCTYPE HTML>
<html>
<head>
<title>Password strength checker in jQuery</title>
<link href='http://fonts.googleapis.com/css?family=Source+Sans+Pro|Open+Sans+Condensed:300|Raleway'
rel='stylesheet' type='text/css'>
<script  src="http://ajax.googleapis.com/ajax/libs/jquery/1.7.1/jquery.min.js"></script><!--  jQuery  Library-->
<link rel="stylesheet" href="css/passwordscheck.css" /><!-- Include Your CSS file here-->
<script src="js/passwordscheck.js"></script><!-- Include Your jQUery file here-->
</head>
<body>
<div id="container">
<div id="header">
<h2>Password Strength Checking with jQuery</h2>
<hr>
</div>
<div id="content">
<form id="register">
<label for="password"><b>Password : </b></label>
<input name="password" id="password" type="password" placeholder="Type Your Password here"/>  
<span id="result"></span>
</form>
</div>
</div>
</body>
</html>
```

jQuery File – passwordscheck.js

```
$(document).ready(function()  {
$('#password').keyup(function()  {
$('#result').html(checkStrength($('#password').val()))
})
function checkStrength(password) {
var strength = 0
if (password.length < 6) {
$('#result').removeClass()
$('#result').addClass('short')
return 'Too short'
}
if (password.length > 7) strength += 1
// If password contains both lower and uppercase characters, increase strength value.
if (password.match(/([a-z].*[A-Z])|([A-Z].*[a-z])/)) strength += 1
// If it has numbers and characters, increase strength value.
```

```
if (password.match(/([a-zA-Z])/) && password.match(/([0-9])/)) strength += 1
// If it has one special character, increase strength value.
if (password.match(/([!,%,&,@,#,$,^,*,?,_,~])/)) strength += 1
// If it has two special characters, increase strength value.
if (password.match(/(.*[!,%,&,@,#,$,^,*,?,_,~].*[!,%,&,@,#,$,^,*,?,_,~])/)) strength += 1
// Calculated strength value, we can return messages
// If value is less than 2

if (strength < 2) {
$('#result').removeClass()
$('#result').addClass('weak')
return 'Weak'
} else if (strength == 2) {
$('#result').removeClass()
$('#result').addClass('good')
return 'Good'
} else {
$('#result').removeClass()
$('#result').addClass('strong')
return 'Strong'
}
}
});
```

CSS File – passwordscheck.css

```
body{
margin: 0;
padding: 0;
font-family: 'Raleway', sans-serif;
font-size: 15px;
line-height: 1.5;
}
#container {
width: 535px;
background: #ffffff;
padding: 20px;
margin: 90px auto;
border-radius: 5px;
height: 150px;
border: 2px solid gray;
}
#header {
text-align: center;
background-color: #FEFFED;
border-radius: 5px;
margin: -39px -20px 10px -20px;
}
h2{
padding-top: 10px;
}
#content {
margin-left: 57px;
margin-top: 40px;
}
```

```
#register label{
margin-right:5px;
}

#register input {
padding: 5px 14px;
border: 1px solid #d5d9da;
box-shadow: 0 0 9px #0E34F5 inset;
width: 272px;
font-size: 1em;
height: 25px;
}

#register .short{
font-weight:bold;
color:#FF0000;
font-size:larger;
}
#register .weak{
font-weight:bold;
color:orange;
font-size:larger;
}
#register .good{
font-weight:bold;
color:#2D98F3;
font-size:larger;
}
#register .strong{
font-weight:bold;
color:  limegreen;
font-size:larger;
}
```

Output:

Exercise Program:

Week-7

7. Write a program to create and Build a star rating system using Jquery.

Program Code:

```
<!DOCTYPE html>
<html lang = "en">
<head>
   <meta charset = "UTF-8">
   <meta name = "viewport" content="width=device-width, initial-scale=1.0">
   <link rel = "stylesheet" href = "https://cdnjs.cloudflare.com/ajax/libs/twitter-bootstrap/4.4.1/css/bootstrap.min.css">
   <script src = "https://cdnjs.cloudflare.com/ajax/libs/jquery/3.4.1/jquery.js"> </script>
   <script  src  =  "https://cdnjs.cloudflare.com/ajax/libs/twitter-bootstrap/4.4.1/js/bootstrap.min.js">  </script>
   <link rel = "stylesheet" href = "https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">
   <script src = "https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
   <title> jQuery simple star rating example </title>
   <style>
   body {
      background-color: aquamarine;
      margin : 0px;
   }
   .fa-star {
      font-size : 50px;
      align-content: center;
   }
   .container {
      height: 100px;
      width: 600px;
      margin: auto;
   }
   </style>
</head>
<body>
   <div class = "container">
      <h2 style="margin-top: 50px;">jQuery simple star rating example</h2>
     <div class = "con">
      <h3 style = "margin-top : 80px; color: green;">Rate our product :-</h3>
      <i class = "fa fa-star" aria-hidden = "true" id = "st1"></i>
     <i class = "fa fa-star" aria-hidden = "true" id = "st2"></i>
     <i class = "fa fa-star" aria-hidden = "true" id = "st3"></i>
     <i class = "fa fa-star" aria-hidden = "true" id = "st4"></i>
     <i class = "fa fa-star" aria-hidden = "true" id = "st5"></i>
     </div>
   </div>
   <script>
      $(document).ready(function()  {
        $("#st1").click(function()  {
           $(".fa-star").css("color",  "black");
           $("#st1").css("color",  "yellow");

        });
```

```
        $("#st2").click(function() {


            $(".fa-star").css("color", "black");
            $("#st1, #st2").css("color", "yellow");

        });
        $("#st3").click(function() {
            $(".fa-star").css("color", "black")

            $("#st1, #st2, #st3").css("color", "yellow");

        });
        $("#st4").click(function() {
            $(".fa-star").css("color", "black");
            $("#st1, #st2, #st3, #st4").css("color", "yellow");

        });
        $("#st5").click(function() {
            $(".fa-star").css("color", "black");
            $("#st1, #st2, #st3, #st4, #st5").css("color", "yellow");

        });
       });
    </script>
</body>
</html>
```

Output:

Exercise Program

Week-8

8: Create a Simple Login form using React JS

Program Code:

```
Import react{user Reducer ,user Effect}from ‘React’;
import { createStyles, makeStyles, Theme } from '@material-ui/core/styles';
import TextField from '@material-ui/core/TextField';
import Card from '@material-ui/core/Card';
import CardContent from '@material-ui/core/CardContent';
import CardActions from '@material-ui/core/CardActions';
import CardHeader from '@material-ui/core/CardHeader';
import Button from '@material-ui/core/Button';
const useStyles = makeStyles((theme: Theme) =>
  createStyles({
    container: {
      display: 'flex',
      flexWrap: 'wrap',
      width: 400,
      margin: `${theme.spacing(0)} auto`
    },
    loginBtn: {
      marginTop: theme.spacing(2),
      flexGrow: 1
    },
    header: {
      textAlign: 'center',
      background: '#212121',
      color: '#fff'
    },
    card: {
      marginTop: theme.spacing(10)
    }
```

```
  })
);
//state type
type State = {
  username: string
  password:  string
  isButtonDisabled: boolean
  helperText: string
  isError: boolean
};
const initialState:State = {
  username: '',
  password: '',
  isButtonDisabled: true,
  helperText: '',
  isError: false
};
type Action = { type: 'setUsername', payload: string }
  | { type: 'setPassword', payload: string }
  | { type: 'setIsButtonDisabled', payload: boolean }
  | { type: 'loginSuccess', payload: string }
  | { type: 'loginFailed', payload: string }
  | { type: 'setIsError', payload: boolean };
const reducer = (state: State, action: Action): State => {
  switch (action.type) {
    case 'setUsername':
      return {
        ...state,
        username: action.payload
      };
    case 'setPassword':
```

```
      return {
        ...state,
        password: action.payload
      };
    case 'setIsButtonDisabled':
      return {
        ...state,
        isButtonDisabled: action.payload
      };
    case 'loginSuccess':
      return {
        ...state,
        helperText: action.payload,
        isError: false
      };
    case 'loginFailed':
      return {
        ...state,
        helperText: action.payload,
        isError: true
      };
    case 'setIsError':
      return {
        ...state,
        isError: action.payload
      };
  }
}
const Login = () => {
  const classes = useStyles();
  const [state, dispatch] = useReducer(reducer, initialState);
```

```
  useEffect(() => {
    if (state.username.trim() && state.password.trim()) {
     dispatch({
       type: 'setIsButtonDisabled',
       payload: false
     });
    } else {
     dispatch({
       type: 'setIsButtonDisabled',
       payload: true
     });
    }
  }, [state.username, state.password]);
const handleLogin = () => {
    if (state.username === 'abc@email.com' && state.password === 'password') {
     dispatch({
       type: 'loginSuccess',
       payload: 'Login Successfully'
     });
    } else {
     dispatch({
       type: 'loginFailed',
       payload: 'Incorrect username or password'
     });
    }
  };
const handleKeyPress = (event: React.KeyboardEvent) => {
    if (event.keyCode === 13 || event.which === 13) {
    state.isButtonDisabled || handleLogin();
    }
  };
```

```
const handleUsernameChange: React.ChangeEventHandler<HTMLInputElement> =
    (event) => {
      dispatch({
        type: 'setUsername',
        payload: event.target.value
      });
    };
const handlePasswordChange: React.ChangeEventHandler<HTMLInputElement> =
    (event) => {
      dispatch({
        type: 'setPassword',
        payload: event.target.value
      });
    }
  return (
    <form className={classes.container} noValidate autoComplete="off">
      <Card className={classes.card}>
        <CardHeader className={classes.header} title="Login App" />
        <CardContent>
          <div>
            <TextField
              error={state.isError}
              fullWidth
              id="username"
              type="email"
              label="Username"
              placeholder="Username"
              margin="normal"
              onChange={handleUsernameChange}
              onKeyPress={handleKeyPress}
            />
```

```
            <TextField
          error={state.isError}
            fullWidth
            id="password"
            type="password"
            label="Password"
            placeholder="Password"
            margin="normal"
            helperText={state.helperText}
            onChange={handlePasswordChange}
            onKeyPress={handleKeyPress}
           />
          </div>
        </CardContent>
        <CardActions>
          <Button
            variant="contained"
            size="large"
            color="secondary"
            className={classes.loginBtn}
            onClick={handleLogin}
            disabled={state.isButtonDisabled}>
            Login
          </Button>
        </CardActions>
      </Card>
    </form>
  );
}
export default Login;
```

Output:

Exercise Program

Week-9

9. Create a blog in React js

Program Code

App.js

```
import React, { useState } from "react";
import ReactDOM from "react-dom";

import "./styles.css";

function App() {
  // React States
  const [errorMessages, setErrorMessages] = useState({});
  const [isSubmitted, setIsSubmitted] = useState(false);

  // User Login info
  const database = [
    {
      username: "user1",
      password: "pass1"
    },
    {
      username: "user2",
      password: "pass2"
    }
  ];

  const errors = {
    uname: "invalid username",
    pass: "invalid password"
  };

  const handleSubmit = (event) => {
    //Prevent page reload
    event.preventDefault();

    var { uname, pass } = document.forms[0];

    // Find user login info
    const userData = database.find((user) => user.username === uname.value);

    // Compare user info
    if (userData) {
      if (userData.password !== pass.value) {
        // Invalid password
        setErrorMessages({ name: "pass", message: errors.pass });
      } else {
        setIsSubmitted(true);
      }
```

```
    } else {
      // Username not found

      setErrorMessages({ name: "uname", message: errors.uname });
    }
  };

  // Generate JSX code for error message
  const renderErrorMessage = (name) =>
  name === errorMessages.name && (
      <div className="error">{errorMessages.message}</div>
    );

  // JSX code for login form
  const renderForm = (
    <div className="form">
      <form onSubmit={handleSubmit}>
        <div  className="input-container">
          <label>Username </label>
          <input type="text" name="uname" required />
          {renderErrorMessage("uname")}
        </div>
        <div  className="input-container">
          <label>Password </label>
          <input type="password" name="pass" required />
          {renderErrorMessage("pass")}
        </div>
        <div  className="button-container">
          <input type="submit" />
        </div>
      </form>
    </div>
  );

  return (
    <div className="app">
      <div  className="login-form">
        <div className="title">Sign In</div>
        {isSubmitted ? <div>User is successfully logged in</div> : renderForm}
      </div>
    </div>
  );
}

export default App;
```

styles.css

```
.app {
  font-family: sans-serif;
  display: flex;
  align-items: center;
  justify-content: center;
  flex-direction: column;
```

```
  gap: 20px;
  height: 100vh;
  font-family: Cambria, Cochin, Georgia, Times, "Times New Roman", serif;

  background-color: #f8f9fd;
}


input[type="text"],
input[type="password"] {
height: 25px;
  border: 1px solid rgba(0, 0, 0, 0.2);
}

input[type="submit"] {
  margin-top: 10px;
  cursor: pointer;
  font-size: 15px;
  background: #01d28e;
  border: 1px solid #01d28e;
  color: #fff;
  padding: 10px 20px;
}

input[type="submit"]:hover {
  background: #6cf0c2;
}

.button-container {
  display: flex;
  justify-content: center;
}

.login-form {
  background-color: white;
  padding: 2rem;
  box-shadow: 0 4px 8px 0 rgba(0, 0, 0, 0.2), 0 6px 20px 0 rgba(0, 0, 0, 0.19);
}

.list-container {
  display: flex;
}

.error {
  color: red;
  font-size: 12px;
}

.title {
  font-size: 25px;
  margin-bottom: 20px;
}

.input-container {
```

```
  display: flex;
  flex-direction: column;
  gap: 8px;
  margin: 10px;
}
```

Output:

Exercise Program

Week-10

10. Create a project on Grocery delivery application

Assume this project is for a huge online departmental store. Assume that they have a myriad of grocery items at their godown. All items must be listed on the website, along with their quantities and prices.Users must be able to sign up and purchase groceries. The system should present him with delivery slot options, and the user must be able to choose his preferred slot. Users must then be taken to the payment page where he makes the payment with his favorite method. This week will have many pages like Header,footer,categories and app.jsx

File Structure:

App.jsx

```
import "./index.css"
import "./App.css"
import products from "./assets/products.json"
import Product from "./components/Product";

export default function App() {
  return (
<div className={"container"}>
<main className={"main"}>
<h1>
        E-Commerce in React and SnipCart
</h1>

<div className={"grid"}>
        {
products.map((product, i) =><Product {...product} key={i}/>)
        }
</div>
</main>
<div
        id="snipcart"
        data-api-
key="NWMwZWNkZGMtZjU2ZS00YzM3LWFlZjYtMmM5Zjk0MWViZDcxNjM3Njg0OTY0ODg5NTk4MTM3"
hidden
>
</div>
</div>
  );
}
```

Components/Product/index.js

```
import "./index.css";

export default function Product(props) {
const {id, imageUrl, name, description, price} = props

    return (
<div
        key={id}
```

```
className={"product"}
>
<img


src={imageUrl}

  alt={`Image of ${name}`}
className={"image-product"}
          />
<h3>{name}</h3>
<p>{description}</p>
<span>${price}</span>
<div>
<button
className="snipcart-add-item"
                data-item-id={id}
                data-item-image={imageUrl}
                data-item-name={name}
                data-item-url="/"
                data-item-price={price}
>
                Add to Cart
</button>
</div>
</div>
   );
}
```

Assets/products.json

```
[
  {
    "id": "t-shirt",
    "name": "Fruits",
    "price": 35.0,
    "imageUrl":"https://www.lalpathlabs.com/blog/wpcontent/uploads/2019/01/Fruits-and-Vegetables.jpg",
    "description": "A Basket of fruits",
    "url": "/api/products/halfmoon"
  },
  {
    "id": "wallet",
    "name": "Vegitables",
    "price": 20.0,
    "imageUrl":"https://img.freepik.com/free-photo/bottom-view-fruits-vegetables-radish-cherry-tomatoes-persimmon-
tomatoes-kiwi-cucumber-apples-red-cabbage-parsley-quince-aubergines-blue-table_140725-146174.jpg",
    "description": "A Basket of Veges",
    "url": "/api/products/wallet"
  },
  {
    "id": "cup",
    "name": "Milk",
    "price": 5.0,
```

```
    "imageUrl":"https://encrypted-
tbn0.gstatic.com/images?q=tbn:ANd9GcSeujHMy6OLRZHTpsrUMVLsHyio1mZiZI4fMQ&usqp=CAU",
    "description": "Healthy Milk",
    "url": "/api/products/veiltail"
  }
]
```


Output

Exercise Program

Week-11

Week-11. Connecting our TODO React js Project with Firebase

We all can create applications but in realtime when we are building an application we have to store the user data some ware now a days best way to store is Firebase which can be integrated in react app
In this week we will learn how to connect our application to firebase

File Structure:

After creating the project make sure to install firebase dependencies:

Install it using npm install firebase

-Now we have mainly 3 pages
1.firebase.js
2.App.js
3.Todo.js

-.In firebase.js we will establish connection to our app and firebase
-In Todo,js we will write the code
And we will import it in to the App.js file

firebase.js

```
import firebase from 'firebase/compat/app';
import 'firebase/compat/auth';
import 'firebase/compat/firestore';

constfirebaseApp = firebase.initializeApp({
apiKey: "",
authDomain: "",
projectId: ",
storageBucket: ",
messagingSenderId: "",
appId: ":,
measurementId: ""
});

constdb = firebaseApp.firestore();

export default db;
```

Todo.js

```
import { ListItem, List, ListItemAvatar, ListItemText, Button, Modal, makeStyles } from '@material-ui/core'
import './Todo.css';
import React, { useState } from 'react';
import db from './firebase'

function Todo(props) {
const [open, setOpen] = useState(false);
```

```
const [input, setInput] = useState(props.todo.todo);

consthandleOpen = () => {
setOpen(true)
   };

constupdateTodo = () => {
      // update to do with the new input text

db.collection('todos').doc(props.todo.id).set({
todo: input
      }, { merge: true })

setOpen(false);
   }

   return (
<>
<div
            open={open}
onClose={e =>setOpen(false)}
>
<div >
<h1>I am a model</h1>
<input placeholder={props.todo.todo} value={input} onChange={event =>setInput(event.target.value)} />
<button onClick={updateTodo}>Update Todo</button>
</div>
</div>
<ulclassName='todo_list'>
<li>
<li primary={props.todo.todo} secondary='Dummy deadline ⏰' />
</li>
<button onClick={e =>setOpen(true)}>Edit</button>
<button  onClick={event  =>db.collection('todos').doc(props.todo.id).delete()}> ❌ DELETE  ME</button>
</ul>
</>
   )
}

 export default Todo
```

Now the last file App.js

```
import React, { useEffect, useState } from 'react';
import './App.css';
import Todo from './Todo';
import db from './firebase'
import firebase from 'firebase/compat/app';
import 'firebase/compat/auth';
import 'firebase/compat/firestore';
function App() {


const [todos, setTodos] = useState([]);
const [input, setInput] = useState('');
```

```
// when the upload, we need to listen to the database and fetch new todos as they get added/remove
useEffect(() => {
  // This code here... fires when the app.js lodes
db.collection('todos').orderBy('timestamp', 'desc').onSnapshot(snapshot => {
    // console.log(snapshot.docs.map(doc =>doc.data()));
setTodos(snapshot.docs.map(doc => ({id: doc.id, todo: doc.data().todo})))
  })
}, []);

constaddTodo = (event) => {
    // this will fire off when we click the button
event.preventDefault(); //will stop the refresh

db.collection('todos').add({
todo: input,
      timestamp: firebase.firestore.FieldValue.serverTimestamp()
    })

setTodos([...todos, input]);
setInput(' ');  // clear up the input after clicking todo
    console.log(todos)
  }

  return (
<div className="App">
<h1>Build A TODO App 🚀!</h1>

<form>

<form>
<span>✅ Write a Todo</span>
<input value={input} onChange={event =>setInput(event.target.value)} />
</form>


<button disabled={!input} type='submit' onClick={addTodo} variant="contained" color="primary">Add
Todo</button>
</form>

<ul>
        {todos.map(todo => (
<Todotodo={todo}/>
          // <li>{todo}</li>
        ))}

<li></li>
</ul>

</div>
  );
}

export default App;
```

Output:

Exercise Program