DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

FULL STACK DEVELOPMENT

[R20A0589]

LABORATORY MANUAL

B. TECH CSE

(III YEAR–IISEM)

R20 REGULATION

(2023-24)

Name :____________________________________

Roll no:____________________________________

Section:____________________________________

Year :____________________________________

MALLAREDDYCOLLEGEOFENGINEERING&TECHNOLOGY

(Autonomous Institution– UGC, Govt. of India)
Recognizedunder2(f)and12(B) ofUGCACT1956
(Affiliated to JNTUH, Hyderabad, Approved by AICTE - Accredited by NBA &NAAC–'A' Grade-ISO9001:2015 Certified)
Maisammaguda, Dhulapally (Post Via. Hakimpet), Secunderabad –500100, Telangana, India

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Vision

To acknowledge quality education and instill high patterns of discipline making the students technologically superior and ethically strong which involves the improvement in the quality of life in human race.

Mission

- ❖ To achieve and impart holistic technical education using the best of infrastructure, outstanding technical and teaching expertise to establish the students in to competent and confident engineers.
- ❖ Evolving the center of excellence through creative and innovative teaching learning practices for promoting academic achievement to produce internationally accepted competitive and world class professionals.

PROGRAMME EDUCATIONAL OBJECTIVES (PEOs)

PEO1-ANALYTICALSKILLS

- To facilitate the graduates with the ability to visualize, gather information, articulate, analyze, solve complex problems, and make decisions. These are essential to address the challenges of complex and computation intensive problems increasing their productivity.

PEO2-TECHNICALSKILLS

- Tofacilitatethegraduateswiththetechnicalskillsthatpreparethemforimmediateemployment and pursue certification providing a deeper understanding of the technology in advanced areas of computer science and related fields, thus encouraging to pursue higher education and research based on their interest.

PEO3- SOFTSKILLS

- To facilitate the graduates with the soft skills that include fulfilling the mission, setting goals, showing self confidence by communicating effectively, having a positive attitude, get involved in team-work, being a leader, managing their career and their life.

PEO4-PROFESSIONALETHICS

- To facilitate the graduates with the knowledge of professional and ethical responsibilities by paying attention to grooming, being conservative with style, following dress codes, safety codes, and adapting them to technological advancements.

PROGRAM SPECIFIC OUTCOMES (PSOs)

After the completion of the course, B.Tech Computer Science and Engineering, the graduates will have the following Program Specific Outcomes:

1. Fundamentals and critical knowledge of the Computer System: - Able to Understand the working principles of the computer System and its components, Apply the knowledge to build, asses, and analyze the software and hardware aspects of it.

2. The comprehensive and Applicative knowledge of Software Development: Comprehensive skills of Programming Languages, Software process models, methodologies, and able to plan, develop, test, analyze, and manage the software and hardware intensive systems in heterogeneous platforms individually or working in teams.

3. Applications of Computing Domain & Research: Able to use the professional, managerial, interdisciplinary skill set, and domain specific tools in development processes, identify their search gaps, and provide innovative solutions to them.

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 anddesignsystemcomponentsorprocessesthatmeetthespecifiedneedswithappropriateconsideration 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 responsibilities relevant 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 multidisciplinary 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.

FULL STACK DEVELOPMENT LABORATORY

1. Course Objectives:

1. This course will enable the students:
2. Usage of various front and backend Tools
3. They can understand and create applications on their own
4. Demonstrate and Designing of Websites can be carried out.
5. Develop web based application using suitable client side and server side code.
6. Implement web based application using effective database access.

2. Course outcomes:

Students will be able to understand

1. Usage of various front and back end Tools
2. They can understand and create applications on their own
3. Demonstrate and Designing of Websites can be carried out.
4. Develop web based application using suitable client side and server side code.
5. Implement web based application using effective database access.

3. Introduction about lab

Minimum System requirements:

Processors: Intel Atom® processor or Intel® Core™ i3 processor.
Disk space: 1 GB.
Operating systems: Windows* 7 or later, mac OS, and Linux.
Python* versions: 2.7.X, 3.6.X., 3.8.X

4. Guidelines to students

A. Standard operating procedure

a) Explanation on today’s experiment by the concerned faculty using PPT covering the following aspects:

1) Name of the experiment
2) Aim

b) Writing the python programs by the students

c) Commands for executing programs

Writing of the experiment in the Observation Book

The students will write the today’s experiment in the Observation book as per the following format:

a) Name of the experiment
b) Aim
c) Writing the program
d) Viva-Voce Questions and Answers
e) Errors observed (if any) during compilation/execution
f) Signature of the Faculty

B. Guide Lines to Students in Lab

Disciplinary to be maintained by the students in the Lab

- Students are required to carry their lab observation book and record book with completed experiments while entering the lab.·
- Students must use the equipment with care. Any damage is caused student is punishable·
- Students are not allowed to use their cell phones/pen drives/ CDs in labs. ·
- Students need to be maintain proper dress code along with ID Card·
- Students are supposed to occupy the computers allotted to them and are not supposed to talk or make noise in the lab. Students, after completion of each experiment they need to be updated in observation notes and same to be updated in the record. ·
- Lab records need to be submitted after completion of experiment and get it corrected with the concerned lab faculty.·
- If a student is absent for any lab, they need to be completed the same experiment in the free time before attending next lab.

Instructions to maintain the record

- Before start of the first lab they have to buy the record and bring the record to the lab.·
- Regularly (Weekly) update the record after completion of the experiment and get it corrected with concerned lab in-charge for continuous evaluation.
- In case the record is lost inform the same day to the faculty in charge and get the new record within 2 days the record has to be submitted and get it corrected by the faculty.·
- If record is not submitted in time or record is not written properly, the evaluation marks (5M) will be deducted.

C. 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.

INDEX

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

INDEX

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

```
<!DOCTYPE html>
<html>
<body>

<h1>Heading 1</h1>
<h2>Heading 2</h2>
<h3>Heading 3</h3>
<h4>Heading 4</h4>
<h5>Heading 5</h5>
<h6>Heading 6</h6>



<img src="pic_trulli.jpg" alt="Italian Trulli">

<p>This is a paragraph.</p>
<p>This is another paragraph.</p>



<table>
  <tr>
    <th>Company</th>
    <th>Contact</th>
    <th>Country</th>
  </tr>
  <tr>
    <td>Alfreds Futterkiste</td>
    <td>Maria Anders</td>
    <td>Germany</td>
  </tr>
  <tr>
    <td>Centro comercial Moctezuma</td>
    <td>Francisco Chang</td>
    <td>Mexico</td>
  </tr>
</table>



</body>
</html>
```

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

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

<head>
    <meta charset="UTF-8">
    <title>Collecting Data</title>
    <script src=
"https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js">
    </script>

    <link rel="stylesheet" href=
"https://cdn.jsdelivr.net/npm/bootstrap@4.5.3/dist/css/bootstrap.min.css"
        integrity=
"sha384-TX8t27EcRE3e/ihU7zmQxVncDAy5uIKz4rEkgIXeMed4M0jlfIDPvg6uqKI2xXr2"
        crossorigin="anonymous">
</head>

<body class="container" style="margin-top: 50px;
    width: 50%; height:auto;">

    <h2 class="text-primary" style=
        "margin-left: 15px; margin-bottom: 10px">
        Hey There,Help Us In Collecting Data
    </h2>

    <form class="container" id="contactForm">
        <div class="card">
            <div class="card-body">
                <div class="form-group">
                    <label for="exampleFormControlInput1">
                        Enter Your Name
                    </label>

                    <input type="text" class="form-control"
                    id="name" placeholder="Enter your name">
                </div>

                <div class="form-group">
                    <label for="exampleFormControlInput1">
                        Email address
                    </label>

                    <input type="email" class="form-control"
                    id="email" placeholder="name@example.com">
                </div>
            </div>
            <button type="submit" class="btn btn-primary"
                style="margin-left: 15px; margin-top: 10px">
                Submit
            </button>
        </div>
    </form>
    <script src=
"https://www.gstatic.com/firebasejs/3.7.4/firebase.js">
    </script>

    <script>
        var firebaseConfig = {
            apiKey: "Use Your Api Key Here",
            authDomain: "Use Your authDomain Here",
            databaseURL: "Use Your databaseURL Here",
            projectId: "Use Your projectId Here",
            storageBucket: "Use Your storageBucket Here",
            messagingSenderId: "Use Your messagingSenderId Here",
            appId: "Use Your appId Here"
        };

        firebase.initializeApp(firebaseConfig);

        var messagesRef = firebase.database()
            .ref('Collected Data');

        document.getElementById('contactForm')
            .addEventListener('submit', submitForm);

        function submitForm(e) {
            e.preventDefault();

            // Get values
            var name = getInputVal('name');
            var email = getInputVal('email');

            saveMessage(name, email);
            document.getElementById('contactForm').reset();
        }

        // Function to get get form values
        function getInputVal(id) {
            return document.getElementById(id).value;
        }

        // Save message to firebase
        function saveMessage(name, email) {
            var newMessageRef = messagesRef.push();
            newMessageRef.set({
                name: name,
                email: email,
            });
        }
    </script>
</body>

</html>
```

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

```
<div class="container">
  <img src="/w3images/bandmember.jpg" alt="Avatar">
  <p>Hello. How are you today?</p>
  <span class="time-right">11:00</span>
</div>

<div class="container darker">
  <img src="/w3images/avatar_g2.jpg" alt="Avatar" class="right">
  <p>Hey! I'm fine. Thanks for asking!</p>
  <span class="time-left">11:01</span>
</div>

<div class="container">
  <img src="/w3images/bandmember.jpg" alt="Avatar">
  <p>Sweet! So, what do you wanna do today?</p>
  <span class="time-right">11:02</span>
</div>

<div class="container darker">
  <img src="/w3images/avatar_g2.jpg" alt="Avatar" class="right">
  <p>Nah, I dunno. Play soccer.. or learn more coding perhaps?</p>
  <span class="time-left">11:05</span>
</div>
/* Chat containers */
.container {
  border: 2px solid #dedede;
  background-color: #f1f1f1;
  border-radius: 5px;
  padding: 10px;
  margin: 10px 0;
}

/* Darker chat container */
.darker {
  border-color: #ccc;
  background-color: #ddd;
}

/* Clear floats */
.container::after {
  content: "";
  clear: both;
  display: table;
}

/* Style images */
.container img {
  float: left;
  max-width: 60px;
  width: 100%;
  margin-right: 20px;
  border-radius: 50%;
}

/* Style the right image */
.container img.right {
  float: right;
  margin-left: 20px;
  margin-right:0;
}

/* Style time text */
.time-right {
  float: right;
  color: #aaa;
}

/* Style time text */
.time-left {
  float: left;
  color: #999;
}
```

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

```
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],
      })
    }
  }
```

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

```
CREATE
OR REPLACE VIEW "public"."poll_results" AS
SELECT
  poll.id AS poll_id,
  o.option_id,
  count(*) AS votes
FROM
  (
    (
      SELECT
        vote.option_id,
        option.poll_id,
        option.text
      FROM
        (
          vote
          LEFT JOIN option ON ((option.id = vote.option_id))
        )
    ) o
    LEFT JOIN poll ON ((poll.id = o.poll_id))
  )
GROUP BY
  poll.question,
  o.option_id,
  poll.id;
CREATE
OR REPLACE VIEW "public"."online_users" AS
SELECT
  count(*) AS count
FROM
  "user"
WHERE
  (
    "user".last_seen_at > (now() - '00:00:15' :: interval)
  );
import { ApolloClient, HttpLink, InMemoryCache, split } from "@apollo/client";
import { GraphQLWsLink } from '@apollo/client/link/subscriptions';
import { createClient } from "graphql-ws";
import { getMainDefinition } from "@apollo/client/utilities";
const GRAPHQL_ENDPOINT = "realtime-poll-example.hasura.app";

const scheme = (proto) =>
  window.location.protocol === "https:" ? `${proto}s` : proto;

const wsURI = `${scheme("ws")}://${GRAPHQL_ENDPOINT}/v1/graphql`;
const httpURL = `${scheme("https")}://${GRAPHQL_ENDPOINT}/v1/graphql`;
const splitter = ({ query }) => {
  const { kind, operation } = getMainDefinition(query) || {};
  const isSubscription =
    kind === "OperationDefinition" && operation === "subscription";
  return isSubscription;
};
const cache = new InMemoryCache();
const options = { reconnect: true };

const wsLink = new GraphQLWsLink(createClient({ url: wsURI, connectionParams: { options }
}));
const httpLink = new HttpLink({ uri: httpURL });
const link = split(splitter, wsLink, httpLink);
const client = new ApolloClient({ link, cache });
```

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

```
<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">
<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css">
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
<script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"></script>
<script src="CheckPasswordStrength.js"></script>
<link href="CheckPasswordStrength.css" rel="stylesheet" />
```

```
$(document).ready(function () {
    $('#txtPassword').keyup(function () {
        $('#strengthMessage').html(checkStrength($('#txtPassword').val()))
    })
    function checkStrength(password) {
        var strength = 0
        if (password.length < 6) {
            $('#strengthMessage').removeClass()
            $('#strengthMessage').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) {
            $('#strengthMessage').removeClass()
            $('#strengthMessage').addClass('Weak')
            return 'Weak'
        } else if (strength == 2) {
            $('#strengthMessage').removeClass()
            $('#strengthMessage').addClass('Good')
            return 'Good'
        } else {
            $('#strengthMessage').removeClass()
            $('#strengthMessage').addClass('Strong')
            return 'Strong'
        }
    }
});
.Short {
    width: 100%;
    background-color: #dc3545;
    margin-top: 5px;
    height: 3px;
    color: #dc3545;
    font-weight: 500;
    font-size: 12px;
}
.Weak {
    width: 100%;
    background-color: #ffc107;
    margin-top: 5px;
    height: 3px;
    color: #ffc107;
    font-weight: 500;
    font-size: 12px;
}
.Good {
    width: 100%;
    background-color: #28a745;
    margin-top: 5px;
    height: 3px;
    color: #28a745;
    font-weight: 500;
    font-size: 12px;
}
.Strong {
    width: 100%;
    background-color: #d39e00;
    margin-top: 5px;
    height: 3px;
    color: #d39e00;
    font-weight: 500;
    font-size: 12px;
}
```

```
<body>
    <form id="form1" runat="server">
        <div class="container py-3">
            <h4 class="text-center text-
uppercase">How to check password strength in jquery</h4>
            <div class="row">
                <div class="col-md-12">
                    <div class="row">
                        <div class="col-md-6 mx-auto">
                            <div class="card border-secondary">
                                <div class="card-header">
                                    <h3 class="mb-0 my-
2">Sign Up</h3>
                                </div>
                                <div class="card-body">
                                    <div class="form-group">
                                        <label>Name</label>
                                        <div class="input-group">
                                            <div class="input-group-
prepend">
                                                <span class="input-
group-text"><i class="fa fa-user"></i></span>
                                            </div>
                                            <asp:TextBox ID="txtFirst
Name" runat="server" CssClass="form-control"></asp:TextBox>
                                        </div>
                                    </div>
                                    <div class="form-group">
                                        <label>Phone Number</label>
                                        <div class="input-group">
                                            <div class="input-group-
prepend">
                                                <span class="input-
group-text"><i class="fa fa-phone"></i></span>
                                            </div>
                                            <asp:TextBox ID="txtPhone
Number" runat="server" CssClass="form-control"></asp:TextBox>
                                        </div>
                                    </div>
                                    <div class="form-group">
                                        <label>Email</label>
                                        <div class="input-group">
                                            <div class="input-group-prepend">
                                                <span class="input-group-text"><i class="fa fa-envelope"></i></span>
                                            </div>
                                            <asp:TextBox ID="txtEmail" runat="server" CssClass="form-control"></asp:TextBox>
                                        </div>
                                    </div>
                                    <div class="form-group">
                                        <label>Password</label>
                                        <div class="input-group">
                                            <div class="input-group-prepend">
                                                <span class="input-group-text"><i class="fa fa-lock"></i></span>
                                            </div>
                                            <asp:TextBox ID="txtPassword" runat="server" TextMode="Password" CssClass="form-control"></asp:TextBox>
                                        </div>
                                        <div id="strengthMessage"></div>
                                    </div>
                                    <div class="form-group">
                                        <label>Confirm Password</label>
                                        <div class="input-group">
                                            <div class="input-group-prepend">
                                                <span class="input-group-text"><i class="fa fa-lock"></i></span>
                                            </div>
                                            <asp:TextBox ID="txtConfirmPassword" runat="server" TextMode="Password" CssClass="form-control"></asp:TextBox>
                                        </div>
                                    </div>
                                    <div class="form-group">
                                        <button type="submit" class="btn btn-success float-right rounded-0">Register</button>
                                    </div>
                                </div>
                            </div>
```

```
                        </div>
                    </div>
                </div>
            </div>
        </div>
    </form>
</body>
```

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

```
$(document).ready(function() {
 $("#st1").click(function() {
    $(".fa-star").css("color", "black");
    $("#st1").css("color", "yellow");

 });
          <!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;
```

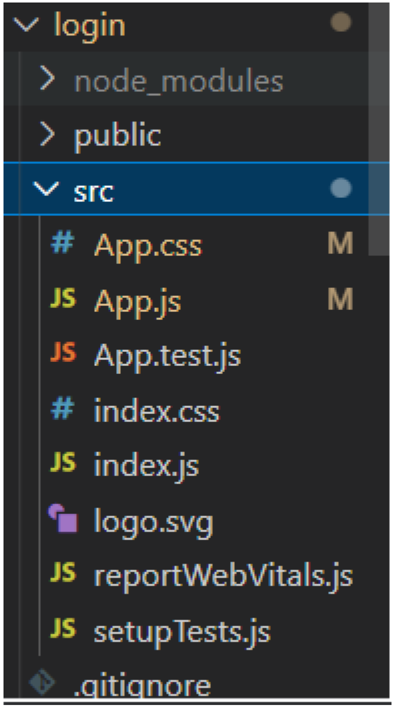
```
        align-content: center;
    }
    .container {
        height: 100px;
        width: 600px;
        margin: auto;
    }
    </style>
</head>
<body>
    <div class = "container">
        <h2 style="margin-top: 50px;">jQuery simple star rating example</h2>
       <div class = "con">
        <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>
```

Week-8. Create a Simple Login form using React js

Now we are creating a login form which is a very important in any application or website as you know if open any website or application you will get a message to login if you click that you will be redirected to login page in this week we will be creating login page

File Structure:-



```
login
  node_modules
  public
  src
    App.css            M
    App.js             M
    App.test.js
    index.css
    index.js
    logo.svg
    reportWebVitals.js
    setupTests.js
  .gitignore
```

--**App.js**-------------------------------------

```
import './App.css';

import React, { useState } from "react";

import ReactDOM from "react-dom";



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;
```

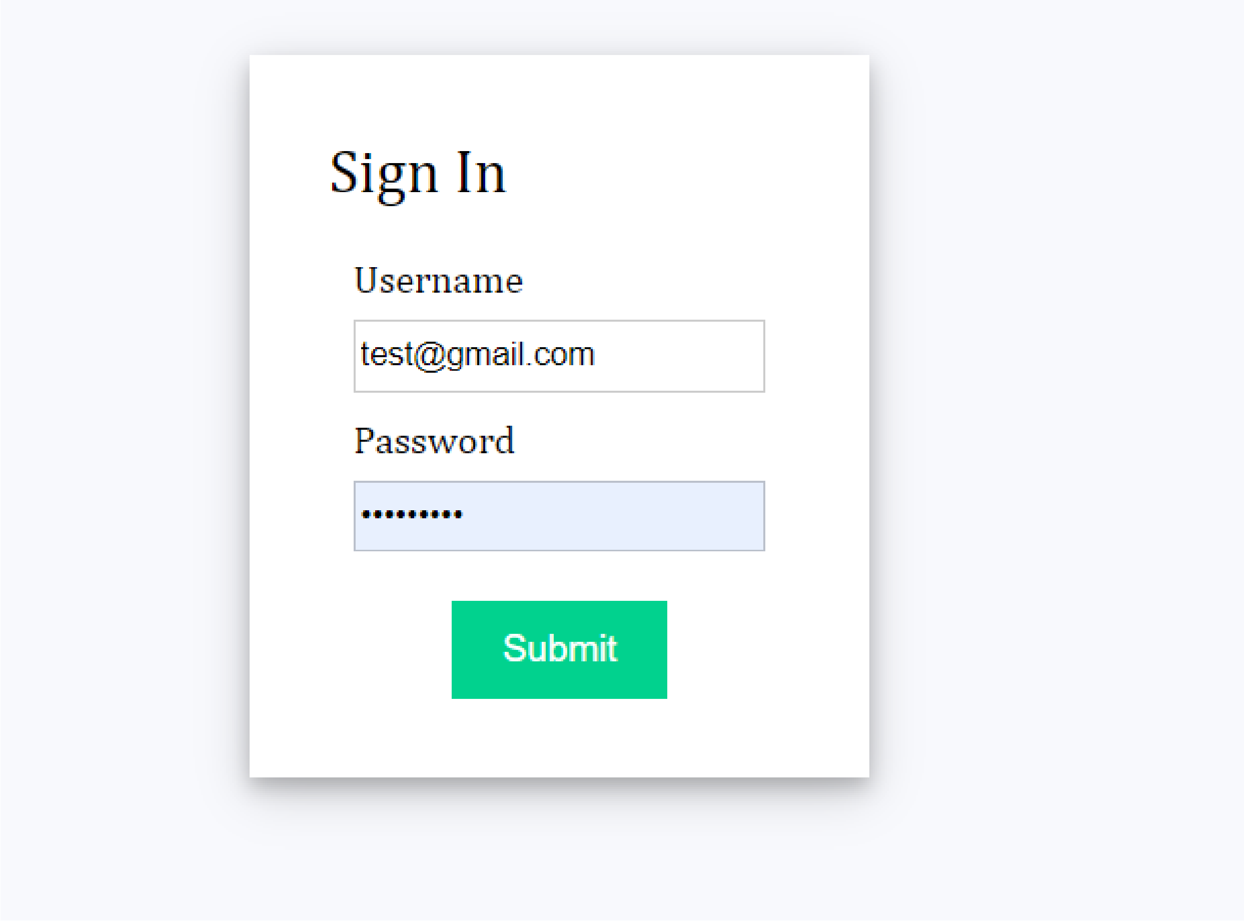
----------------Now Create a App.css file in same folder----------------

App.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;
}
```



Username

test@gmail.com

Password

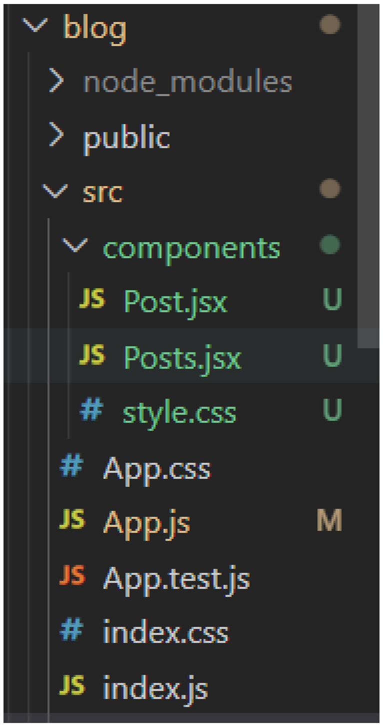
••••••••••

Week-9. Create a blog in React js

In this week we are going to create a blog website using react js

We have mainly 3 pages
1.App.js
2.Post.js
3.Posts.js

This is the Structure of the project



```
blog
  node_modules
  public
  src
    components
      JS Post.jsx      U
      JS Posts.jsx     U
      #  style.css     U
    #  App.css
    JS App.js          M
    JS App.test.js
    #  index.css
    JS index.js
```

1. App.js

```
import logo from './logo.svg';
import './App.css';
import Posts from "./components/Posts";
function App() {
  return (
    <div className="main-container">
      <h1 className="main-heading">
        Blog App using React Js
      </h1>
      <Posts />
    </div>
  );
}

export default App;
```

2. Posts.js

```
import React from "react";
import "./style.css";

import Post from "./Post";

const Posts = () => {
  const blogPosts = [
    {
      title: "JAVASCRIPT",
      body: `JavaScript is the world most popular
      lightweight, interpreted compiled programming
      language. It is also known as scripting
      language for web pages. It is well-known for
      the development of web pages, many non-browser
      environments also use it. JavaScript can be
      used for Client-side developments as well as
      Server-side developments`,
      author: "Nishant Singh ",
      imgUrl:
        "https://media.geeksforgeeks.org/img-practice/banner/diving-into-excel-thumbnail.png",
    },
    {
      title: "Data Structure ",
      body: `There are many real-life examples of
      a stack. Consider an example of plates stacked
      over one another in the canteen. The plate
      which is at the top is the first one to be
      removed, i.e. the plate which has been placed
      at the bottommost position remains in the
      stack for the longest period of time. So, it
      can be simply seen to follow LIFO(Last In
      First Out)/FILO(First In Last Out) order.`,
      author: "Suresh Kr",
      imgUrl:
        "https://media.geeksforgeeks.org/img-practice/banner/coa-gate-2022-thumbnail.png",
    },
    {
      title: "Algorithm",
      body: `The word Algorithm means “a process
      or set of rules to be followed in calculations
      or other problem-solving operations”. Therefore
      Algorithm refers to a set of rules/instructions
      that step-by-step define how a work is to be
      executed upon in order to get the expected
      results. `,
      author: "Monu Kr",
      imgUrl:
        "https://media.geeksforgeeks.org/img-practice/banner/google-test-series-thumbnail.png",
    },
    {
      title: "Computer Network",
      body: `An interconnection of multiple devices,
      also known as hosts, that are connected using
      multiple paths for the purpose of sending/
      receiving data media. Computer networks can
      also include multiple devices/mediums which
      help in the communication between two different
      devices; these are known as Network devices
      and include things such as routers, switches,
      hubs, and bridges. `,
      author: "Sonu Kr",
      imgUrl:
        "https://media.geeksforgeeks.org/img-practice/banner/cp-maths-java-thumbnail.png",
    },
  ];

  return (
    <div className="posts-container">
      {blogPosts.map((post, index) => (
        <Post key={index} index={index} post={post} />
      ))}
    </div>
  );
};

export default Posts;
```

3.Post.js

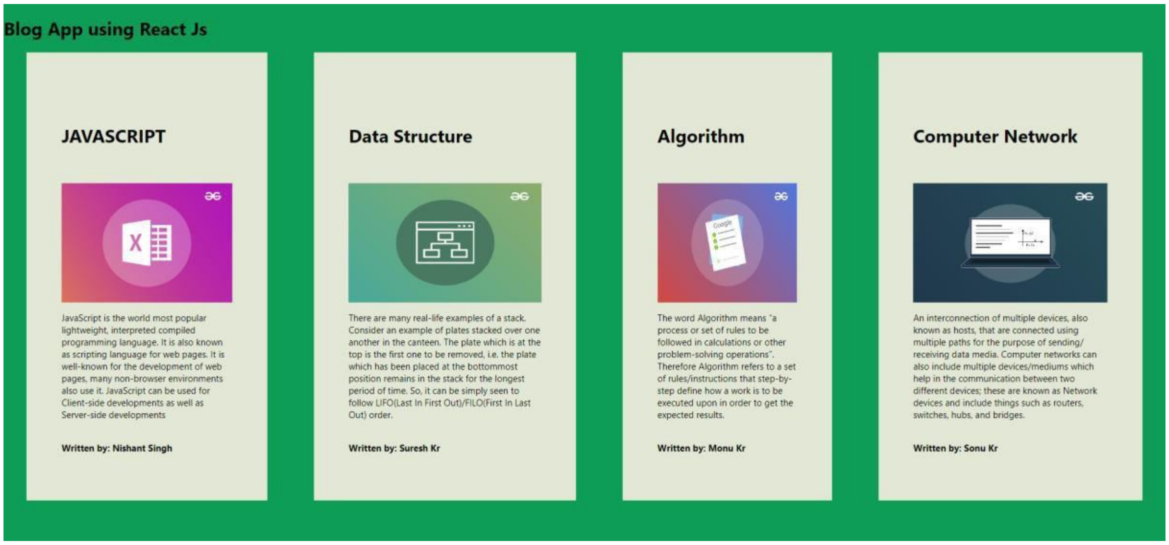
```
import React from "react";
import "./style.css";
const Post = ({ post: { title, body,
imgUrl, author }, index }) => {
return (
    <div className="post-container">
      <h1 className="heading">{title}</h1>
      <img className="image" src={imgUrl} alt="post" />
      <p>{body}</p>
      <div className="info">
        <h4>Written by: {author}</h4>
      </div>
    </div>
  );
};

export default Post;
```

Now we will style the project

Style.css in componenets folder

```
body {
   background-color: #0e9d57;
}
.posts-container {
   display: flex;
   justify-content: center;
   align-items: center;
}
.post-container {
   background: #e2e8d5;
   display: flex;
   flex-direction: column;
   padding: 3%;
   margin: 0 2%;
   height: 40%;
}
.heading {
   height: 126px;
   text-align: center;
   display: flex;
   align-items: center;
}
.image {
   width: 100%;
   height: 210px;
}
```

OUTPUT



Blog App using React Js

JAVASCRIPT

JavaScript is the world most popular lightweight, interpreted compiled programming language. It is also known as scripting language for web pages. It is well-known for the development of web pages, many non-browser environments also use it. JavaScript can be used for Client-side developments as well as Server-side developments

Written by: Nishant Singh

Data Structure

There are many real-life examples of a stack. Consider an example of plates stacked over one another in the canteen. The plate which is at the top is the first one to be removed, i.e. the plate which has been placed at the bottommost position remains in the stack for the longest period of time. So, it can be simply seen to follow LIFO(Last In First Out)/FILO(First In Last Out) order.

Written by: Suresh Kr

Algorithm

The word Algorithm means "a process or set of rules to be followed in calculations or other problem-solving operations". Therefore Algorithm refers to a set of rules/instructions that step-by-step define how a work is to be executed upon in order to get the expected results.

Written by: Monu Kr

Computer Network

An interconnection of multiple devices, also known as hosts, that are connected using multiple paths for the purpose of sending/ receiving data media. Computer networks can also include multiple devices/mediums which help in the communication between two different devices; these are known as Network devices and include things such as routers, switches, hubs, and bridges.

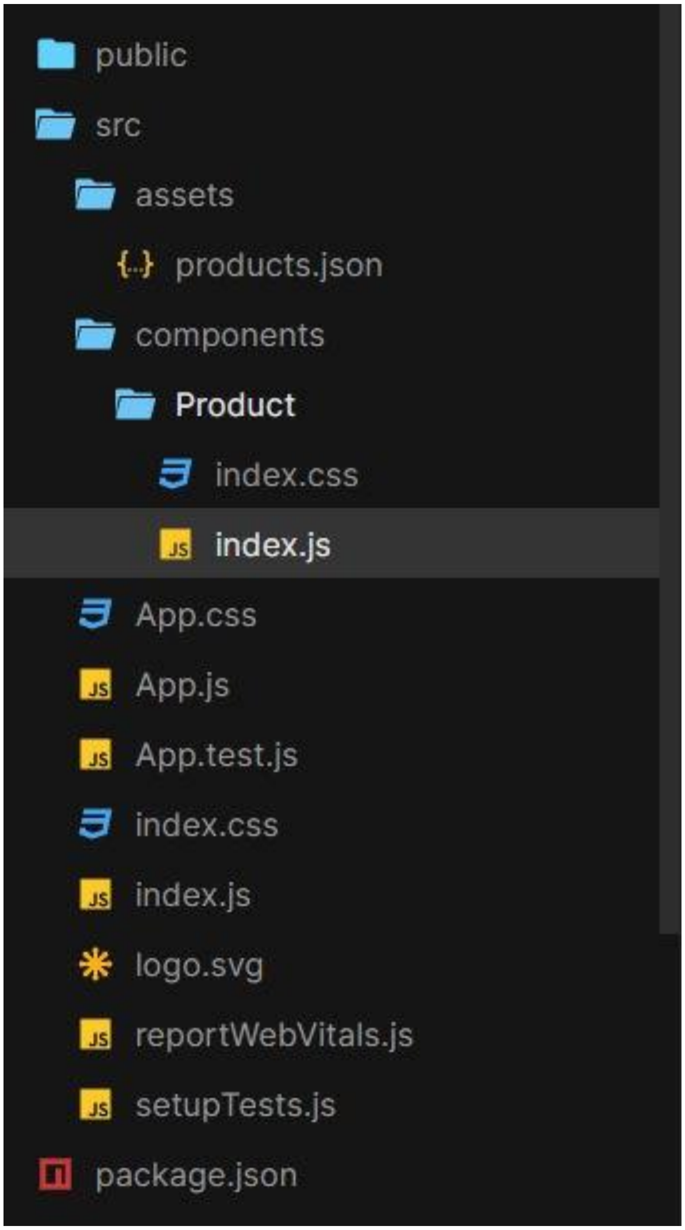
Written by: Sonu Kr

Week-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:



```
public
src
  assets
    products.json
  components
    Product
      index.css
      index.js
  App.css
  App.js
  App.test.js
  index.css
  index.js
  logo.svg
  reportWebVitals.js
  setupTests.js
package.json
```

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="NWMwZWNkZGMtZjU2ZS00YzM3LWFlZjYtMmM5Zjk0MWViZDcxNjM3Njg0OTY
0ODg5NTk4MTM3" hidden
        >
        </div>
      </div>
  );
}
```

Components/Product/index.js

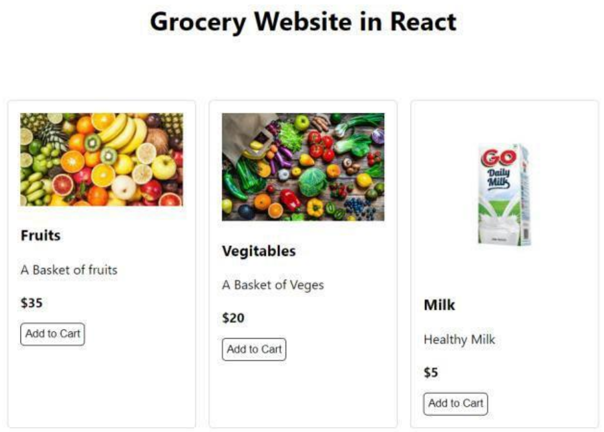
```
import "./index.css";

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/wp-content/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&us
qp=CAU",
    "description": "Healthy Milk",
    "url": "/api/products/veiltail"
  }
```

]

Output



Grocery Website in React

Fruits

A Basket of fruits

$35

Vegitables

A Basket of Veges

$20

Milk

Healthy Milk

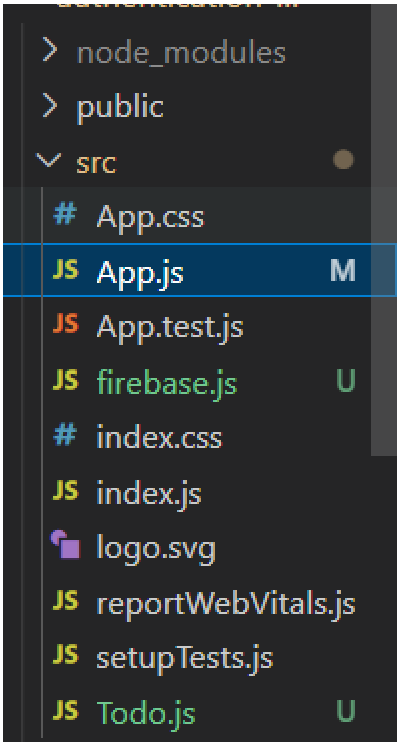
$5

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:



```
> node_modules
> public
v src
  # App.css
  JS App.js            M
  JS App.test.js
  JS firebase.js       U
  # index.css
  JS index.js
  logo.svg
  JS reportWebVitals.js
  JS setupTests.js
  JS Todo.js           U
```

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';

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

const db = firebaseApp.firestore();

export default db;
```

Note Replace the highlighted code with your firebase connection compoenents

You can get you own keys from firebase account for more details Take the Reference of below video

https://www.youtube.com/watch?v=ad6IavyAHsQ

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);
```

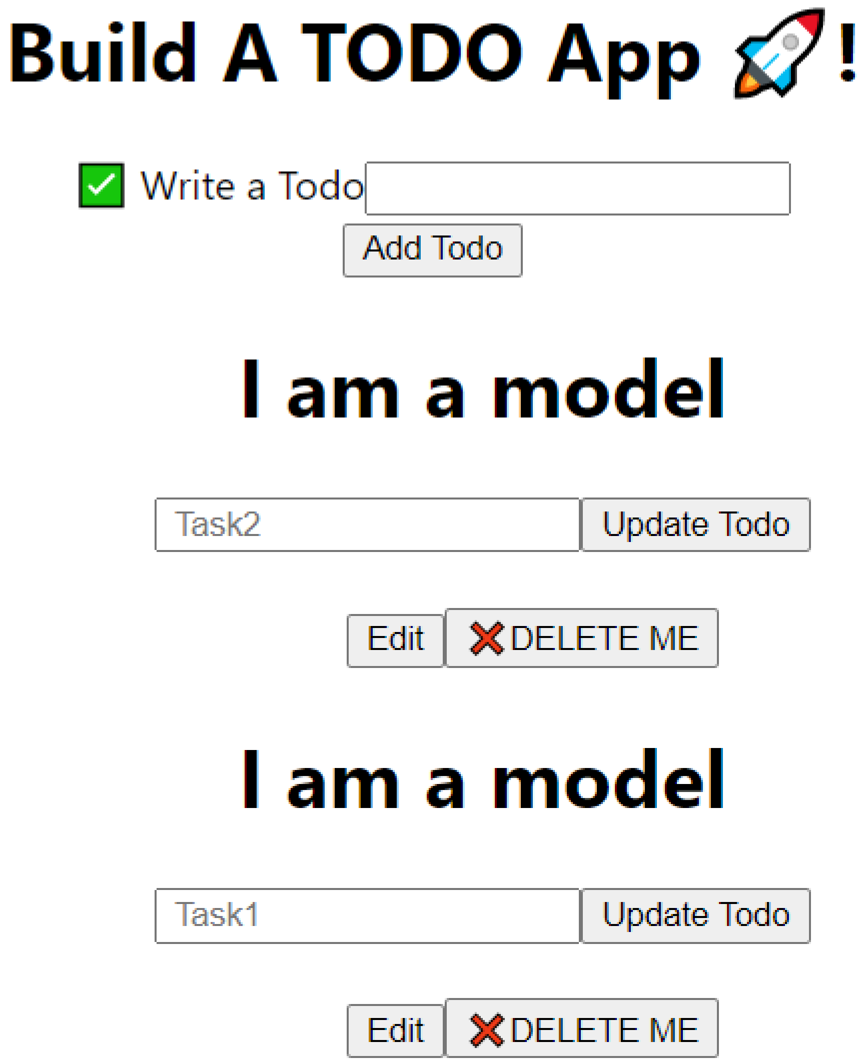
```
  const [input, setInput] = useState(props.todo.todo);

  const handleOpen = () => {
    setOpen(true)
  };

  const updateTodo = () => {
    // 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>
      <ul className='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})))
  })
}, []);

  const addTodo = (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 => (
          <Todo todo={todo}/>
          // <li>{todo}</li>
        ))}

        <li></li>
      </ul>

    </div>
  );
}

export default App;
```

OUTPUT



Build A TODO App 🚀!

✅ Write a Todo

I am a model

Task2

I am a model

Task1