ANNA UNIVERSITY

THANTHAI PERIYAR

GOVERNMENT INSTITUTE OF TECHNOLOGY

VELLORE-632 002

MASTER OF COMPUTER APPLICATIONS

MC4212– FULL STACK WEB DEVELOPMENT

Name: ______________________________

Reg. No: ______________________________

THANTHAI PERIYAR GOVERNMENT INSTITUTE OF TECHNOLOGY

VELLORE-632 002

MASTER OF COMPUTER APPLICATIONS

MC4212 –FULL STACK WEB DEVELOPMENT LABORATORY

2023 – 2025

Certified that this is a bonafide record of work done by ..……………………………………………………………………………………….. with Reg. no . . . . in this department during the academic year of 2023 – 2024.

Staff Incharge **Head of the Department**

Date:

Submitted for M.C.A Degree Practical Examination (II Semester) held on

……...................................... at TPGIT Bagayam, Vellore – 2.

Internal Examiner **External Examiner**

INDEX

01.FORM VALIDATION USING JAVASCRIPT

Program:

Formvalidation.html

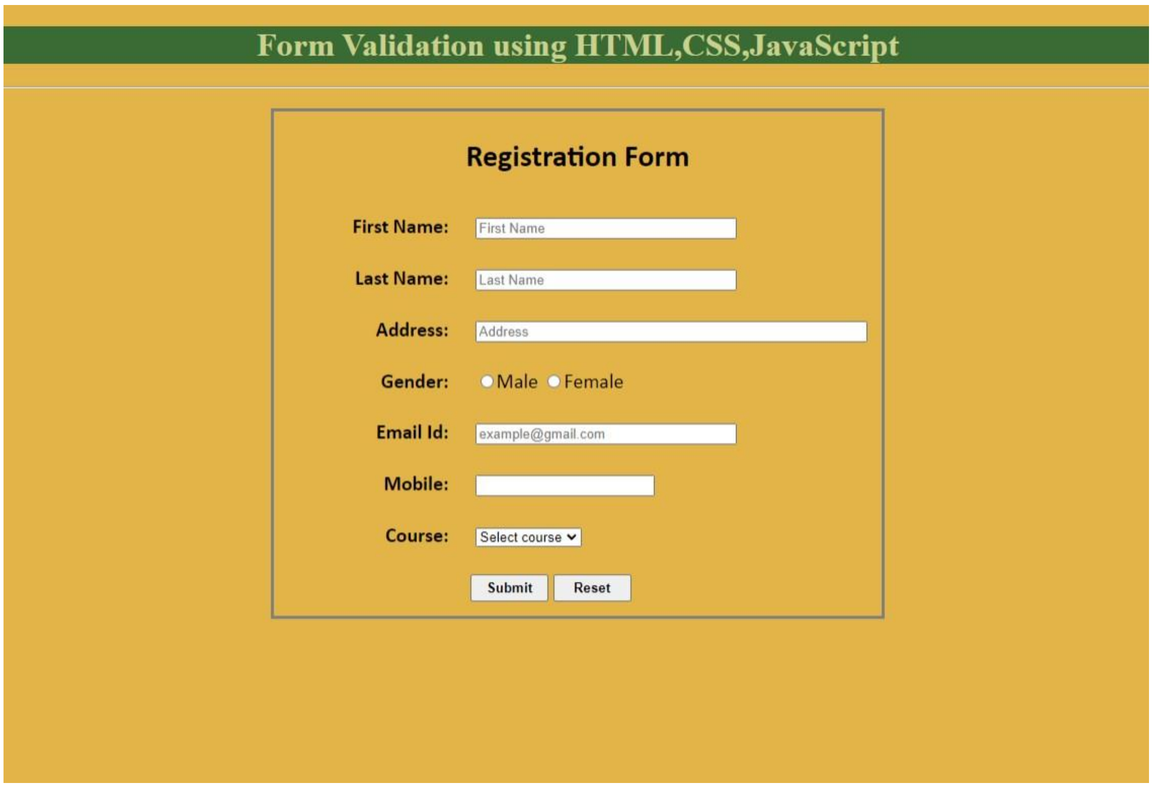
```
<!DOCTYPE html>
<html>
<head>
  <title>Registration Form</title>
  <style>
  h1{
        font-family: Times;
           background-color: #3A6B35;
        color:#cbd18F ;
     }
        body{
        font-family: Calibri;
           background-color: #E3B448;
     }
     input[type="text]{
        width: 250px;
     }
     input[type="submit"], input[type="reset"] {
        width: 77px;
        height: 27px;
           color: Black;
           font-weight: bold;
        position: relative;left: 180px;
     }
     form{
        text-align: center;
        font-family: Calibri;
        font-size: 20px;
        border: 3px solid grey;
        width: 600px;
        margin: 20px auto;
    }
    td {
        padding: 12px;
    }
    td:first-child {
        text-align: right;
        font-weight: bold;
    }
    td:last-child {
        text-align: left;

</style>
<script>
    function validate() {
        var fname = document.reg_form.fname;
        var lname = document.reg_form.lname;
        var address = document.reg_form.address;
        var gender = document.reg_form.gender;
        var email = document.reg_form.email;
        var mobile = document.reg_form.mobile;
        var course = document.reg_form.course;

        if (fname.value.length <= 0) {
            alert("Name is required");
            fname.focus();
            return false;
        }
if (lname.value.length <= 0) {
    alert("Last Name is required");
    lname.focus();
    return false;
}
if (address.value.length <= 0) {
    alert("Address is required");
    address.focus();
    return false;
}
if (gender.value.length <= 0) {
    alert("Gender is required");
    gender.focus();
    return false;
}
if (email.value.length <= 0) {
    alert("Email Id is required");
    email.focus();
    return false;
}
var val = mobile.value
            } else {
             alert("Invalid number; must be ten digits")
            mobile.focus()
             return false
            }
if (course.value == "select course") {
    alert("Course is required, Select any course");
    course.focus();
    return false;
}
```

```
            return false;
          }
      </script>
  </head>
  <body>
      <center><h1>Form Validation using HTML,CSS,JavaScript</h1></center>
      <hr>
      <form method="" action="" name="reg_form" onsubmit="return validate()">
          <h2>Registration Form</h2>
          <table>
              <tr>
                  <td><label>First Name: </label></td>
                  <td>
                      <input type="text" name="fname" placeholder="First Name">
                  </td>
              </tr>
              <tr>
                  <td><label>Last Name: </label></td>
                  <td>
                      <input type="text" name="lname" placeholder="Last Name">
                  </td>
              </tr>
              <tr>
                  <td><label>Address: </label></td>
                  <td>
                      <input type="textarea" size="50" name="address" placeholder="Address">
                  </td>
              </tr>
              <tr>
                  <td><label>Gender: </label></td>
                  <td>
```

```
        <input type="radio" name="gender" value="male">Male
        <input type="radio" name="gender" value="femele">Female
    </td>
</tr>
<tr>
    <td><label>Email Id: </label></td>
    <td>
        <input type="text" name="email" placeholder="example@gmail.com">
    </td>
</tr>
<tr>
    <td><label>Mobile: </label></td>
    <td>
        <input type="number" name="mobile">
    </td>
</tr>
<tr>
<td><label>Course: </label></td>
    <td>
        <select name="course">
            <option value="select course">Select course</option>
            <option value="HTML">HTML</option>
            <option value="CSS">CSS</option>
            <option value="JavaScript">JAVASCRIPT</option>
            <option value="Java">JAVA</option>
        </select>
    </td>
</tr>
<tr>
    <td>
        <input type="submit" name="submit" value="Submit">
```

```
                <input type="reset" name="reset" value="Reset">
            </td>
        </tr>
    </table>
  </form>
</body>
</html>
```

Output:



Form Validation using HTML,CSS,JavaScript

Registration Form

First Name:	First Name
Last Name:	Last Name
Address:	Address
Gender:	○Male ○Female
Email Id:	example@gmail.com
Mobile:	
Course:	Select course

02.GET DATA USING FETCH API FROM AN OPEN-SOURCE ENDPOINT & DISPLAY THE CONTENTS IN THE FORM OF A CARD.

index.html

```
<!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>fake store api</title>
        <link rel="stylesheet" href="style.css">
    </head>
    <body>
        <h1 class="heading">Fetch fake store Api</h1>
        <div id="cards">
        </div>
        <script src="script.js"></script>
    </body>
</html>
```

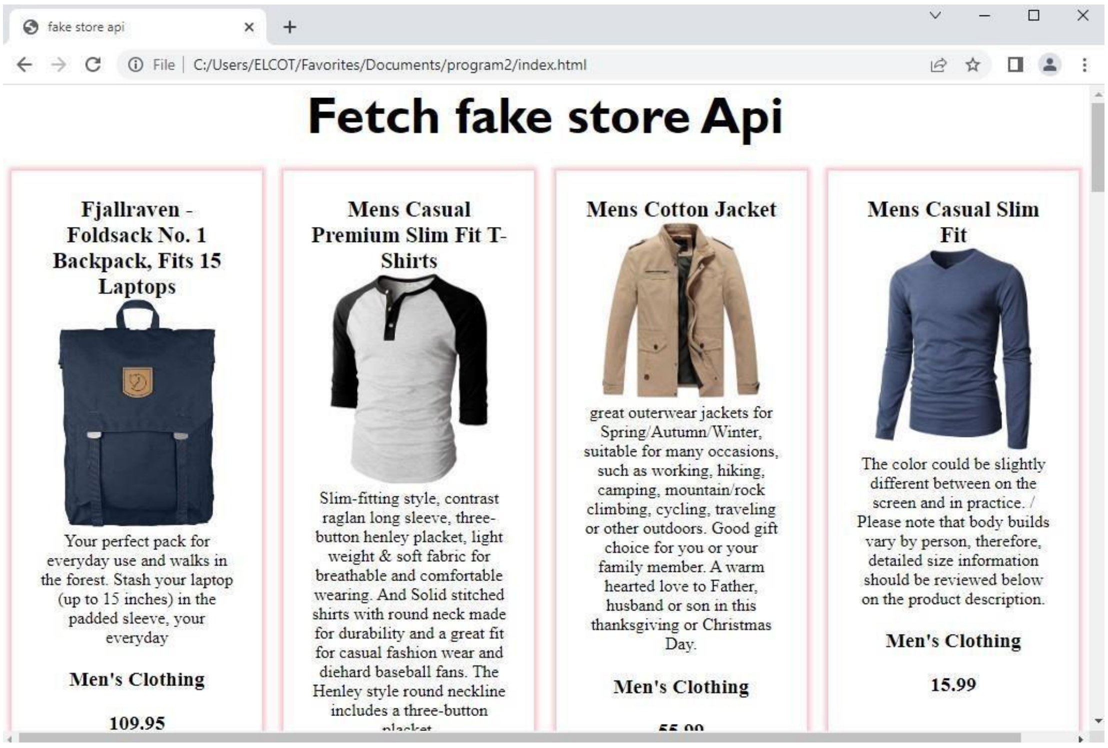
style.css

```
*{
    padding: 0;
    margin: 0;
    box-sizing: border-box;
}
.heading{
    text-align: center;
    font-size: 3rem;
    margin-bottom: 1.5rem;
    font-family: 'Gill Sans', 'Gill Sans MT', Calibri, 'Trebuchet MS', sans-serif;
}
#cards{
    display: flex;
    justify-content: space-around;
    flex-wrap: wrap;
}
.card{
    width: 23%;
    box-shadow: 0 0 4px 3px pink;
    text-align: center;
    padding: 1.5em;
    margin-bottom: 2em;
}
.images{
    width: 80%;
}
.title{
    font-size: 1.3rem;
}
.category,.price{
    font-weight: bold;
    text-transform: capitalize ;
    margin: 1em;
    font-size: 1.2em;
}
```

script.js

```
fetch('https://fakestoreapi.com/products').then((data)=>{
    return data.json();
}).then((completedata)=>{
    let data1="";
    completedata.map((values)=>{
        data1+=` <div class="card">
        <h1 class="title">${values.title}</h1>
        <img src=${values.image} alt="img" class="images">
        <p>${values.description}</p>
        <p class="category">${values.category}</p>
        <p class="price">${values.price}</p>
    </div>`;
    });
    document.getElementById("cards").innerHTML=data1;

}).catch((err)=>{
    console.log(err);
})
```

Output:



Fetch fake store Api

Fjallraven - Foldsack No. 1 Backpack, Fits 15 Laptops

Your perfect pack for everyday use and walks in the forest. Stash your laptop (up to 15 inches) in the padded sleeve, your everyday

Men's Clothing

109.95

Mens Casual Premium Slim Fit T-Shirts

Slim-fitting style, contrast raglan long sleeve, three-button henley placket, light weight & soft fabric for breathable and comfortable wearing. And Solid stitched shirts with round neck made for durability and a great fit for casual fashion wear and diehard baseball fans. The Henley style round neckline includes a three-button placket.

Mens Cotton Jacket

great outerwear jackets for Spring/Autumn/Winter, suitable for many occasions, such as working, hiking, camping, mountain/rock climbing, cycling, traveling or other outdoors. Good gift choice for you or your family member. A warm hearted love to Father, husband or son in this thanksgiving or Christmas Day.

Men's Clothing

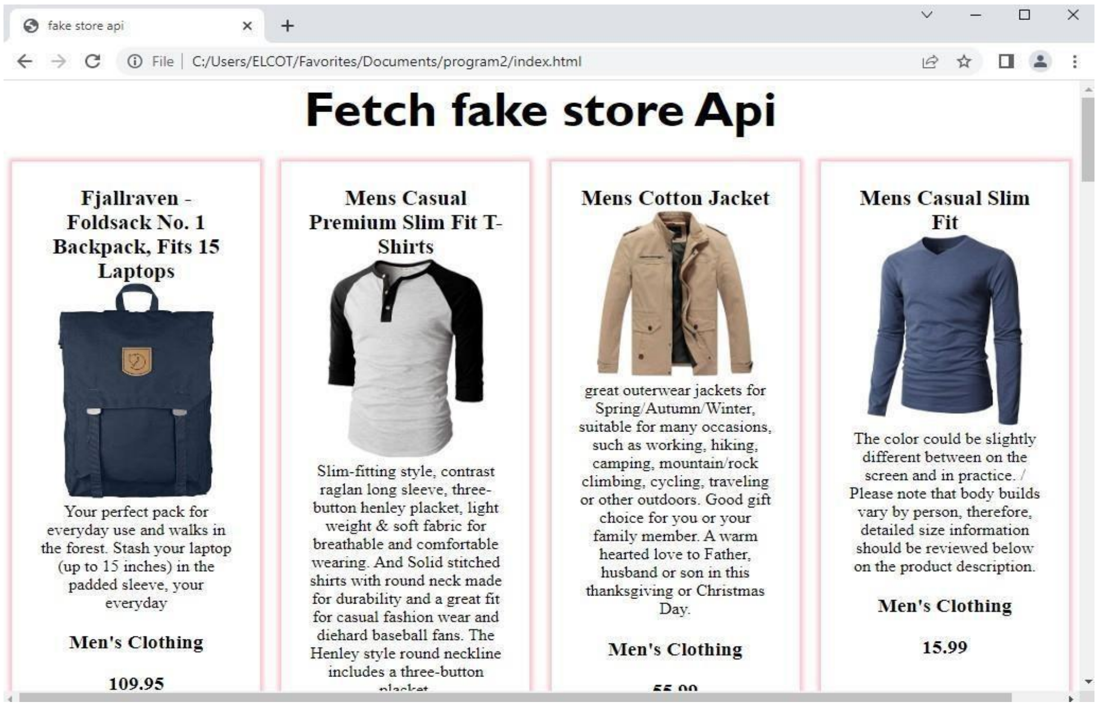
55.99

Mens Casual Slim Fit

The color could be slightly different between on the screen and in practice. / Please note that body builds vary by person, therefore, detailed size information should be reviewed below on the product description.

Men's Clothing

15.99


fake store api
C:/Users/ELCOT/Favorites/Documents/program2/index.html

Fetch fake store Api

Fjallraven - Foldsack No. 1 Backpack, Fits 15 Laptops

Your perfect pack for everyday use and walks in the forest. Stash your laptop (up to 15 inches) in the padded sleeve, your everyday

Men's Clothing

109.95

Mens Casual Premium Slim Fit T-Shirts

Slim-fitting style, contrast raglan long sleeve, three-button henley placket, light weight & soft fabric for breathable and comfortable wearing. And Solid stitched shirts with round neck made for durability and a great fit for casual fashion wear and diehard baseball fans. The Henley style round neckline includes a three-button placket.

Mens Cotton Jacket

great outerwear jackets for Spring/Autumn/Winter, suitable for many occasions, such as working, hiking, camping, mountain/rock climbing, cycling, traveling or other outdoors. Good gift choice for you or your family member. A warm hearted love to Father, husband or son in this thanksgiving or Christmas Day.

Men's Clothing

55.99

Mens Casual Slim Fit

The color could be slightly different between on the screen and in practice. / Please note that body builds vary by person, therefore, detailed size information should be reviewed below on the product description.

Men's Clothing

15.99

03.CREATE A NODEJS SERVER THAT SERVES STATIC HTML AND CSS FILES TO THE USER WITHOUT USING EXPRESS.

Create a server

index.js:

```
var http = require('http');
var fs = require('fs');
var path = require('path');

http.createServer(function (request, response) {
    console.log('request ', request.url);

    var filePath = '.' + request.url;
    if (filePath == './') {
        filePath = './index.html';
    }

    var extname = String(path.extname(filePath)).toLowerCase();
    var mimeTypes = {
        '.html': 'text/html',
        '.js': 'text/javascript',
        '.css': 'text/css',
        '.json': 'application/json',
        '.png': 'image/png',
        '.jpg': 'image/jpg',
        '.gif': 'image/gif',
        '.svg': 'image/svg+xml',
        '.wav': 'audio/wav',
        '.mp4': 'video/mp4',
        '.woff': 'application/font-woff',
        '.ttf': 'application/font-ttf',
        '.eot': 'application/vnd.ms-fontobject',
        '.otf': 'application/font-otf',
        '.wasm': 'application/wasm'
    };

    var contentType = mimeTypes[extname] || 'application/octet-stream';

    fs.readFile(filePath, function(error, content)
        { if (error) {
            if(error.code == 'ENOENT') {
                fs.readFile('./404.html', function(error, content)
                {
                    response.writeHead(404, { 'Content-Type':
                    'text/html' }); response.end(content, 'utf-8');
                });
            }
            else {
                response.writeHead(500);
                response.end('Sorry, check with the site admin for error: '+error.code+'
            ..\n'); }
        }
        else {
            response.writeHead(200, { 'Content-Type':
            contentType }); response.end(content, 'utf-8');
        }
    });

}).listen(8080);
console.log('Server running at http://localhost:8080/');
```

index.html:

```
<!DOCTYPE html>
```

```
<html>
<head>
<title>Ex.No.3</title>
<link rel="stylesheet" href="main.css">
<script src="main.js"></script>
</head>
<body>

<h2>NodeJS server that serves static HTML and CSS files to the user without using Express</h2>
<p id="demo">This is HTML page along with CSS.</p>
<input type="button" value="Click Here" onClick="myFunction()" />
</body>
</html>
```

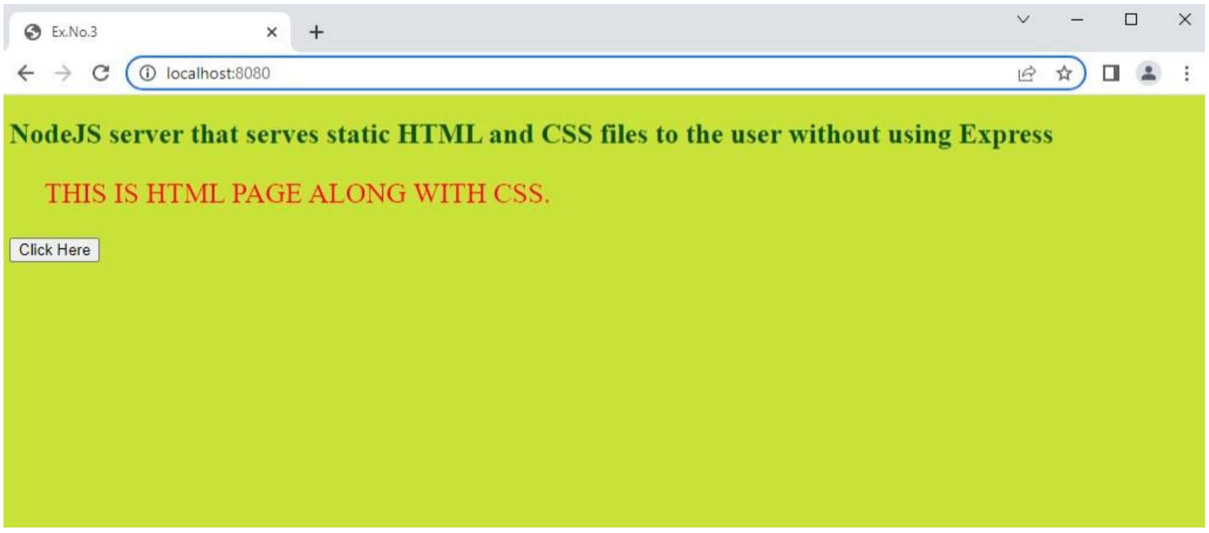
Main.css:

```
body{
  color: #085308;
  background-color: #cae33b;
}

p {
  color: red;
  text-indent: 30px;
  text-transform: uppercase;
  font-size: 24px;
}
```

main.js:

```
function myFunction() {
    document.getElementById("demo").innerHTML = "Now the JavaScript code is running."; }
```

Output:

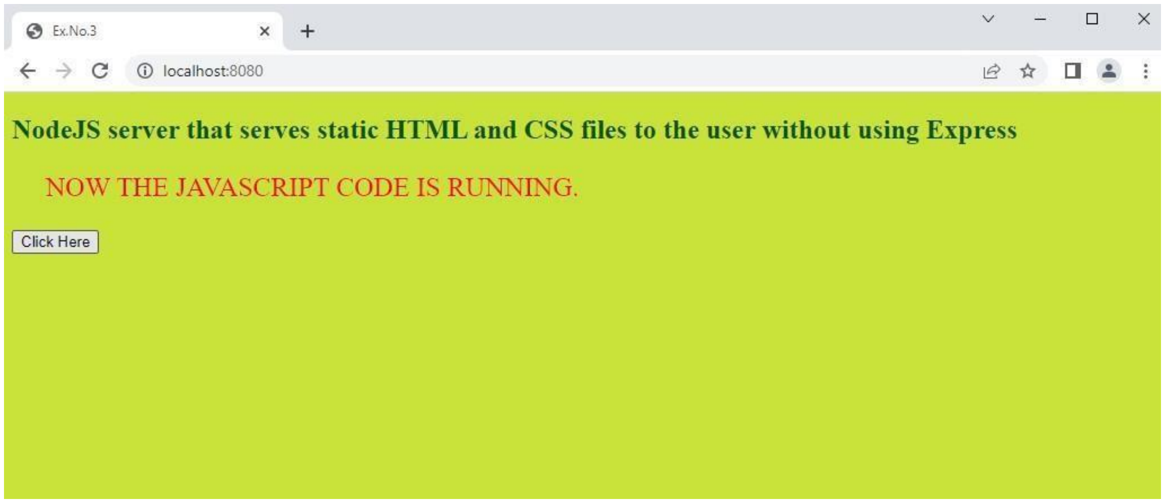


Ex.No.3

localhost:8080

NodeJS server that serves static HTML and CSS files to the user without using Express

THIS IS HTML PAGE ALONG WITH CSS.



Ex.No.3

localhost:8080

NodeJS server that serves static HTML and CSS files to the user without using Express

NOW THE JAVASCRIPT CODE IS RUNNING.

04.USE HANDLE BARS AND EXPRESS IN NODE.JS

Step 1: Open IntelliJ IDEA Ultimate JetBrains: Developer Tools
Step 2: Select New Project and then select Express from Generators, give name of the project and select Handlebars from the View Engine and Click on Create button.

Layout.hbs:
```
<!DOCTYPE html>
<html>
  <head>
    <title>{{title}}</title>
     <script
          src="https://code.jquery.com/jquery-3.2.1.js"
          integrity="sha256-DZAnKJ/6XZ9si04Hgrsxu/8s717jcIzLy3oi35EouyE="
crossorigin="anonymous"></script>
         <!-- Latest compiled and minified CSS & JS -->
         <link rel="stylesheet" media="screen"
         href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">
         <script
         src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"></script>
         <link rel='stylesheet' href='/stylesheets/style.css' />
    </head>
    <body>
          {{{body}}}
    </body>
     </html>
```

Index.hbs:
```
<div class="jumbotron">
  <div class="container">
     <h1>Welcome to Node.js with Handlebars</h1>
```

```
      <p>This demonstrates a form with Node.js</p>
    </div>
  </div>
  <div class="container">
    <div class="row">
      <div class="col-xs-12 col-sm-12 col-md-12 col-lg-12">
          <form action="/" method="post" role="form" enctype="application/x-www-form-
urlencoded">
             <legend>Enter Registration Information Here</legend>
             <div class="form-group">
             <label for="first_name">First Name</label>
              <input type="text" class="form-control" name="first_name" id="first_name"
placeholder="First _Name">
             </div>
             <div class="form-group">
             <label for="last_name">Last Name</label>
               <input type="text" class="form-control" name="last_name" id="first_name"
placeholder="Last _Name">
             </div>
              <div class="form-group">
              <label for="email">Email Address</label>
              <input type="text" class="form-control" name="email" id="email" placeholder="Email
_Address">
              </div>
              <div class="form-group">
              <label for="pw">Password</label>
              <input type="password" class="form-control" name="pw" id="pw">
    </div>
  <div class="form-group">
              <label for="pw_confirm">Confirm Password</label>
              <input type="password" class="form-control" name="pw_confirm" id="pw_confirm">
          </div>
         <button type="submit" class="btn btn-primary">Submit</button>
      </form>
    </div>
  </div>
</div>
```

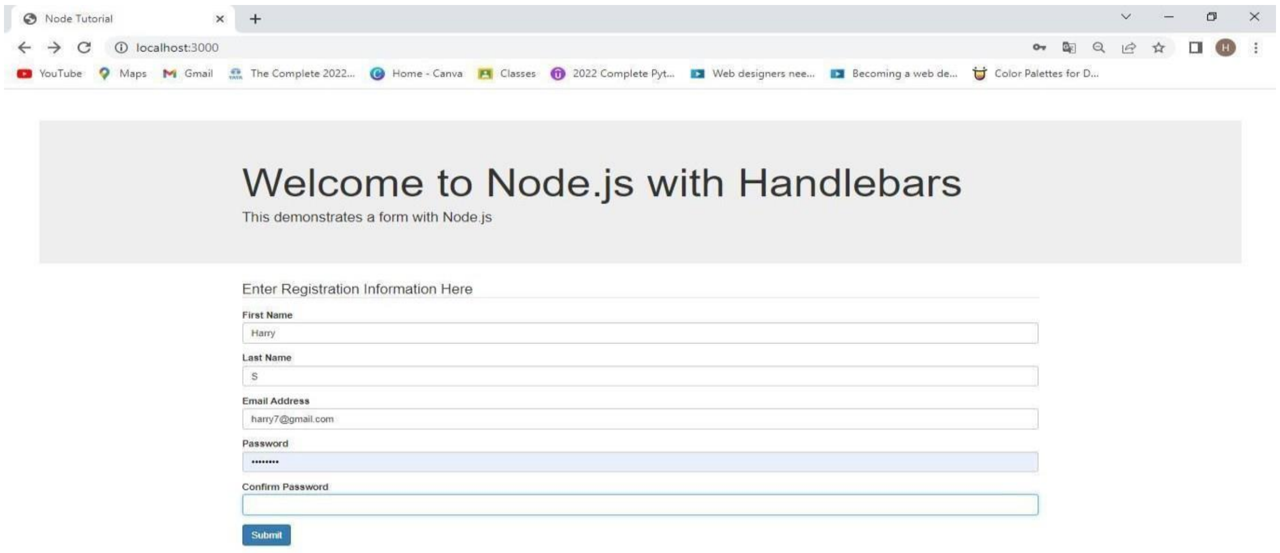
Index.js:

```
var express = require('express'); var router = express.Router();
router.get('/', function(req, res, next) { res.render('index', { title: 'Node Tutorial' });
});
router.post('/', function(req, res){ var body = req.body;
  var res_body = {
first_name: body.first_name, last_name: body.last_name, email: body.email

});
};
res.render('welcome', res_body);
module.exports = router;
```

Welcome.hbs:

```
 <div class="jumbotron">
    <div class="container">
       <h1>Hello {{first_name}} {{last_name}}!</h1>
        <p>You are registered with an email address of {{email}}</p>
    <p>Well done!</p>
  </div>
</div>
```

Output:


Node Tutorial

localhost:3000

Welcome to Node.js with Handlebars

This demonstrates a form with Node.js

Enter Registration Information Here

First Name
Harry

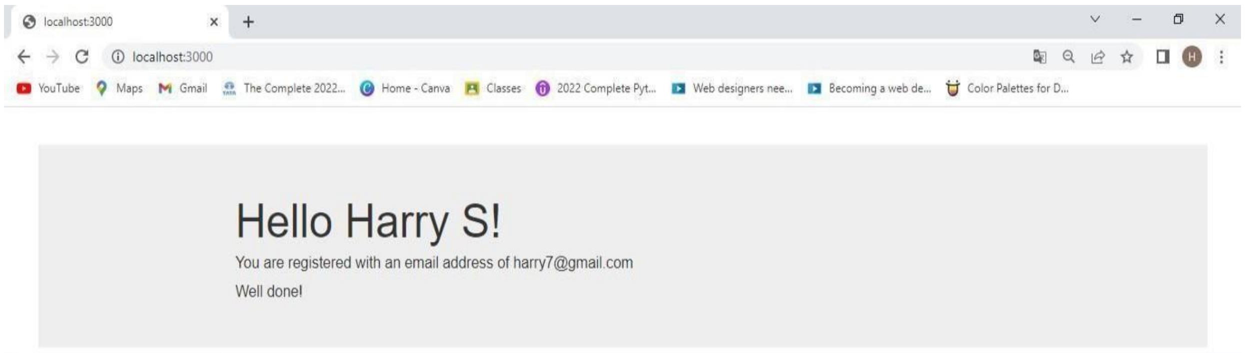
Last Name
S

Email Address
harry7@gmail.com

Password
••••••••

Confirm Password

Submit


localhost:3000

localhost:3000

Hello Harry S!

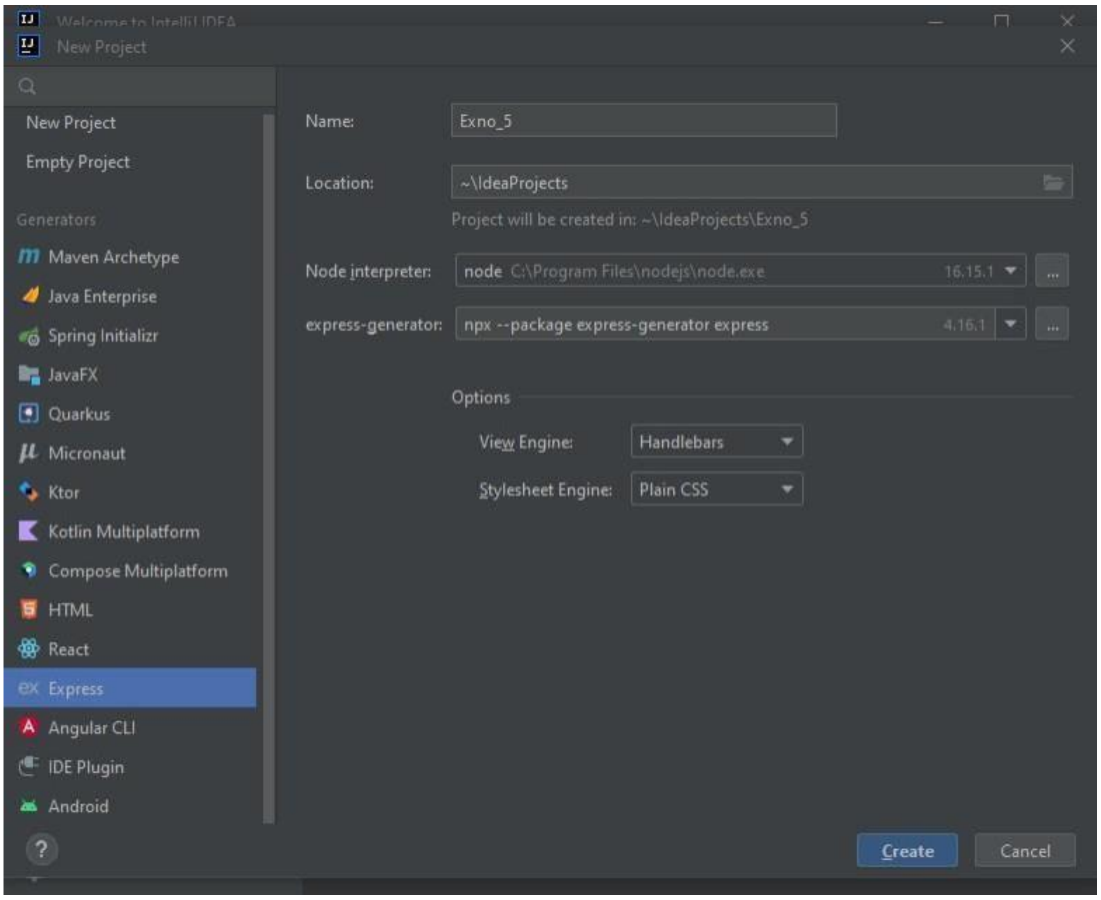
You are registered with an email address of harry7@gmail.com

Well done!

05.CRUD USING MONGODB AND NODEJS

Step 1: Open IntelliJ IDEA Ultimate JetBrains: Developer Tools

Step 2: Select New Project and then select Express from Generators, give name of the project and select Handlebars from the View Engine and Click on Create button.


Name: Exno_5
Location: ~\IdeaProjects
Project will be created in: ~\IdeaProjects\Exno_5
Node interpreter: node C:\Program Files\nodejs\node.exe 16.15.1
express-generator: npx --package express-generator express 4.16.1
Options
View Engine: Handlebars
Stylesheet Engine: Plain CSS

Step 3: Enter the following command in the Terminal to install MongoDB.

npm install --save mongodb

index.hbs:

```
<h1>MONGODB - EXERCISE</h1>
<section class="insert">
   <h3>Insert Data</h3>
   <form action="/insert" method="post">
```

```
            <div class="input">
                <label for="name">Student Name</label>
                <input type="text" id="name" name="name">
            </div>
            <div class="input">
                <label for="year">Year of Study</label>
                <input type="text" id="year" name="year">
            </div>
            <div class="input">
                <label for="gender">Gender</label>
                <input type="text" id="gender" name="gender">
            </div>
            <button type="submit">INSERT</button>
        </form>
    </section>
    <section class="get">
        <h3>Get Data</h3>
        <a href="/get-data">LOAD DATA</a>
        <div>
            {{# each items }}
                <article class="item">
                    <div>Student Name: {{ this.name }}</div>
                    <div>Year of  Study: {{ this.year }}</div>
                    <div>Gender: {{ this.gender }}</div>
                    <div>ID: {{ this._id }}</div>
                </article>
            {{/each}}
        </div>
    </section>
    <section class="update">
        <h3>Update Data</h3>
```

```
  <form action="/update" method="post">
      <div class="input">
          <label for="id">ID</label>
          <input type="text" id="id" name="id">
      </div>
      <div class="input">
          <label for="name">Student Name</label>
          <input type="text" id="name" name="name">
      </div>
      <div class="input">
          <label for="year">Year of Study</label>
          <input type="text" id="year" name="year">
      </div>
      <div class="input">
          <label for="gender">Gender</label>
          <input type="text" id="gender" name="gender">
      </div>
      <button type="submit">UPDATE</button>
  </form>
</section>
<section class="delete">
  <h3>Delete Data</h3>
  <form action="/delete" method="post">
      <div class="input">
          <label for="id">ID</label>
          <input type="text" id="id" name="id">
      </div>
      <button type="submit">DELETE</button>
  </form>
</section>
```

style.css:

```css
body {
  padding: 50px;
  font: 14px "Lucida Grande", Helvetica, Arial, sans-serif;
}

a {
  color: #00B7FF;
}

section {
  float: left;
  background: #bdc3c7;
  padding: 10px;
  margin: 30px;
  width: 300px;
  box-shadow: 3px 3px 1px #34495e;
  min-height: 400px;
}

section:first-of-type {
  margin-left: 0;
}

section:last-of-type {
  margin-right: 0;
}
section h3 {
  border-bottom: 1px solid black;
  padding-bottom: 5px;
}
```

```
.insert {
  background: #2ecc71;
 }

 .get {
  background: #ecf0f1;
 }

 .update {
  background: #3498db;
 }

 .delete {
  background: #e74c3c;
 }

 .input label {
  display: block;
  font-weight: bold;
  padding: 2px 0;
 }

 input,
 button {
  font: inherit;
 }

 button {
  margin-top:
  10px; border:
  none;
  box-shadow: 1px 1px 1px #34495e;
  border-radius: 0;
  background: #ecf0f1;
  cursor: pointer;
}

button:hover {
  background: #bdc3c7;
}

.item {
  margin: 10px 0;
  padding: 5px;
  background: #95a5a6;
  border: 1px solid black;
}
```

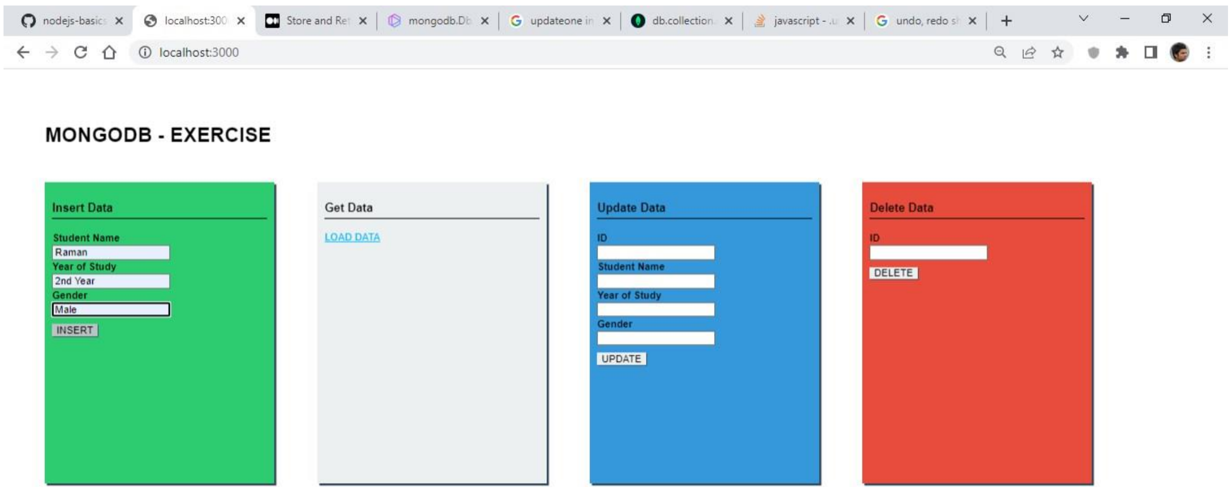
index.js:

```
const express = require('express');
const router = express.Router();
const objectId = require('mongodb').ObjectId;
const assert = require('assert');
const {MongoClient} = require("mongodb");

let url = 'mongodb://localhost:27017/test';

router.get('/', function(req, res) {
  res.render('index');
});
```

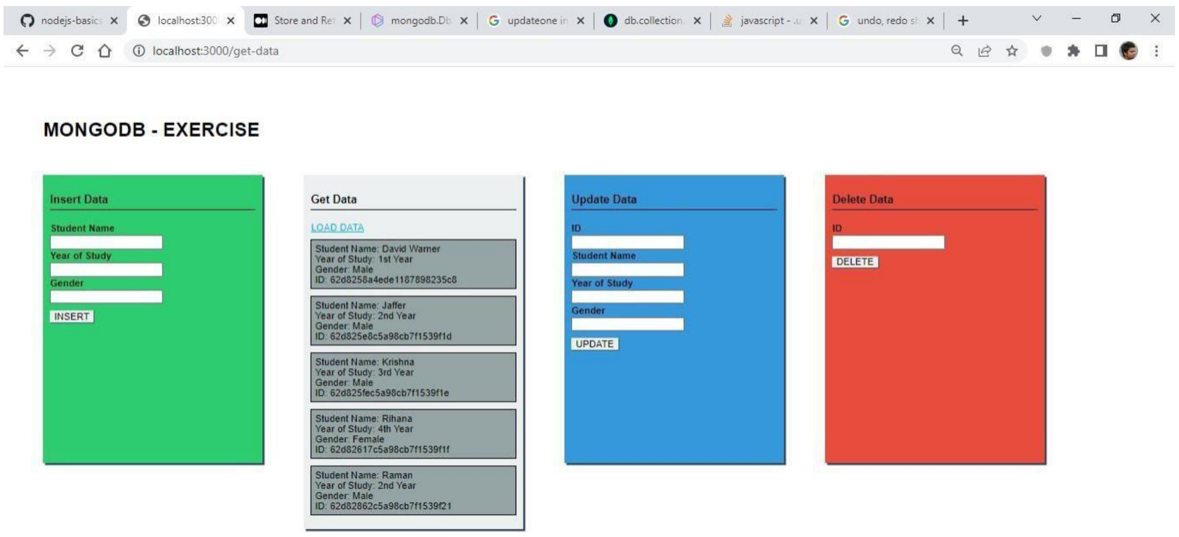
```
router.get('/get-data', function(req, res) {
  let resultArray = [];
  MongoClient.connect(url, function(err, db) {
    assert.equal(null, err);
    const dbo = db.db("test");
    let cursor = dbo.collection('user-data').find();
    cursor.forEach(function(doc, err) {
    assert.equal(null, err);
    resultArray.push(doc);
    }, function() {
      res.render('index', {items: resultArray});
    });
  });
});

router.post('/insert', function(req, res) {
  let item = {
    name: req.body.name,
    year: req.body.year,
    gender: req.body.gender
  };

  MongoClient.connect(url, function(err, db) {
    assert.equal(null, err);
    const dbo = db.db("test");
    dbo.collection('user-data').insertOne(item, function(err) {
      assert.equal(null, err);
      console.log('Item inserted');
    });
  });
  res.redirect('/');
});
router.post('/update', function(req) {
  let item = {
    name: req.body.name,
    year: req.body.year,
    gender: req.body.gender
  };
  let id = req.body.id;
  MongoClient.connect(url, function(err, db) {
  assert.equal(null, err);
    const dbo = db.db("test");
    dbo.collection('user-data').updateOne({"_id": objectId(id)}, {$set: item}, function(err) {
      assert.equal(null, err);
      console.log('Item updated');
    });
  });
});
router.post('/delete', function(req) {
  let id = req.body.id;
  MongoClient.connect(url, function(err, db) {
    assert.equal(null, err);
    const dbo = db.db("test");
    dbo.collection('user-data').deleteOne({"_id":objectId(id)}, function(err) {
      assert.equal(null, err);
      console.log('Item deleted');
    });
  });
});

module.exports = router;
```

Output:

Insert Data:

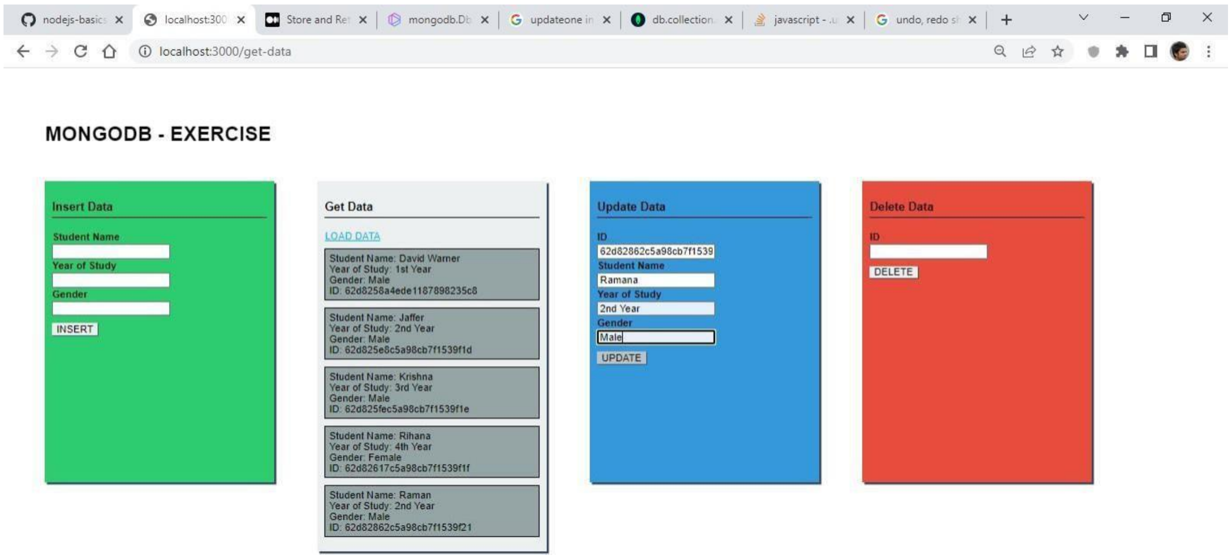


Read Data:



```
Student Name: David Warner
Year of Study: 1st Year
Gender: Male
ID: 62d8258a4ede1187898235c8

Student Name: Jaffer
Year of Study: 2nd Year
Gender: Male
ID: 62d825e8c5a98cb7f1539f1d

Student Name: Krishna
Year of Study: 3rd Year
Gender: Male
ID: 62d825fec5a98cb7f1539f1e

Student Name: Rihana
Year of Study: 4th Year
Gender: Female
ID: 62d82617c5a98cb7f1539f1f

Student Name: Raman
Year of Study: 2nd Year
Gender: Male
ID: 62d82862c5a98cb7f1539f21
```

Update Data:



localhost:3000/get-data

MONGODB - EXERCISE

Insert Data

Student Name

Year of Study

Gender

INSERT

Get Data

LOAD DATA

```
Student Name: David Warner
Year of Study: 1st Year
Gender: Male
ID: 62d8258a4ede1187898235c8

Student Name: Jaffer
Year of Study: 2nd Year
Gender: Male
ID: 62d825e8c5a98cb7f1539f1d

Student Name: Krishna
Year of Study: 3rd Year
Gender: Male
ID: 62d825fec5a98cb7f1539f1e

Student Name: Rihana
Year of Study: 4th Year
Gender: Female
ID: 62d82617c5a98cb7f1539f1f

Student Name: Raman
Year of Study: 2nd Year
Gender: Male
ID: 62d82862c5a98cb7f1539f21
```

Update Data

ID: 62d82862c5a98cb7f1539

Student Name: Ramana

Year of Study: 2nd Year

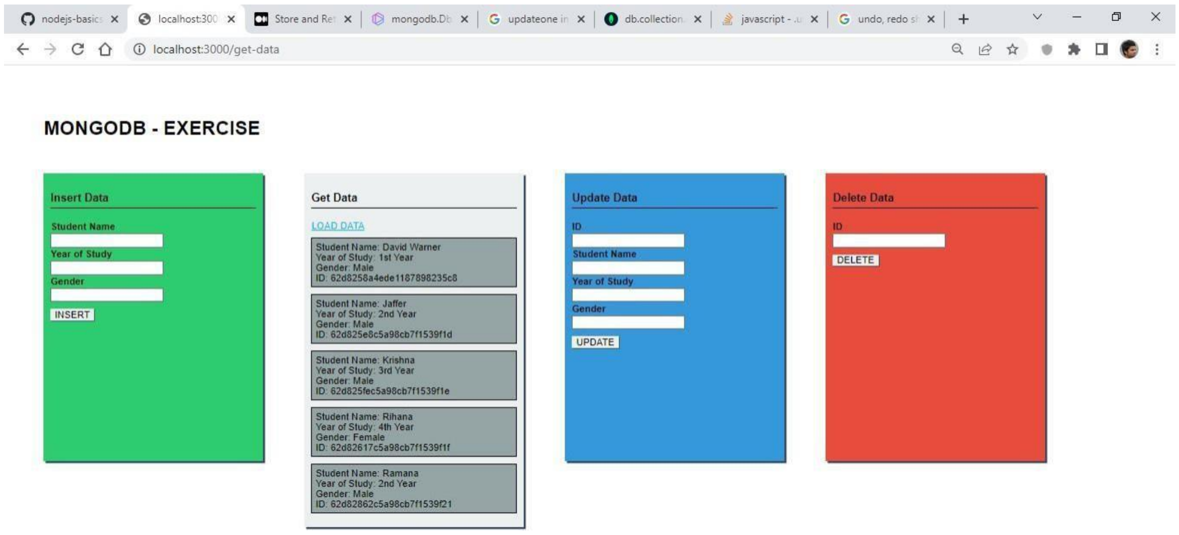
Gender: Male

UPDATE

Delete Data

ID

DELETE



localhost:3000/get-data

MONGODB - EXERCISE

Insert Data

Student Name

Year of Study

Gender

INSERT

Get Data

LOAD DATA

```
Student Name: David Warner
Year of Study: 1st Year
Gender: Male
ID: 62d8258a4ede1187898235c8

Student Name: Jaffer
Year of Study: 2nd Year
Gender: Male
ID: 62d825e8c5a98cb7f1539f1d

Student Name: Krishna
Year of Study: 3rd Year
Gender: Male
ID: 62d825fec5a98cb7f1539f1e

Student Name: Rihana
Year of Study: 4th Year
Gender: Female
ID: 62d82617c5a98cb7f1539f1f

Student Name: Ramana
Year of Study: 2nd Year
Gender: Male
ID: 62d82862c5a98cb7f1539f21
```

Update Data

ID

Student Name

Year of Study

Gender

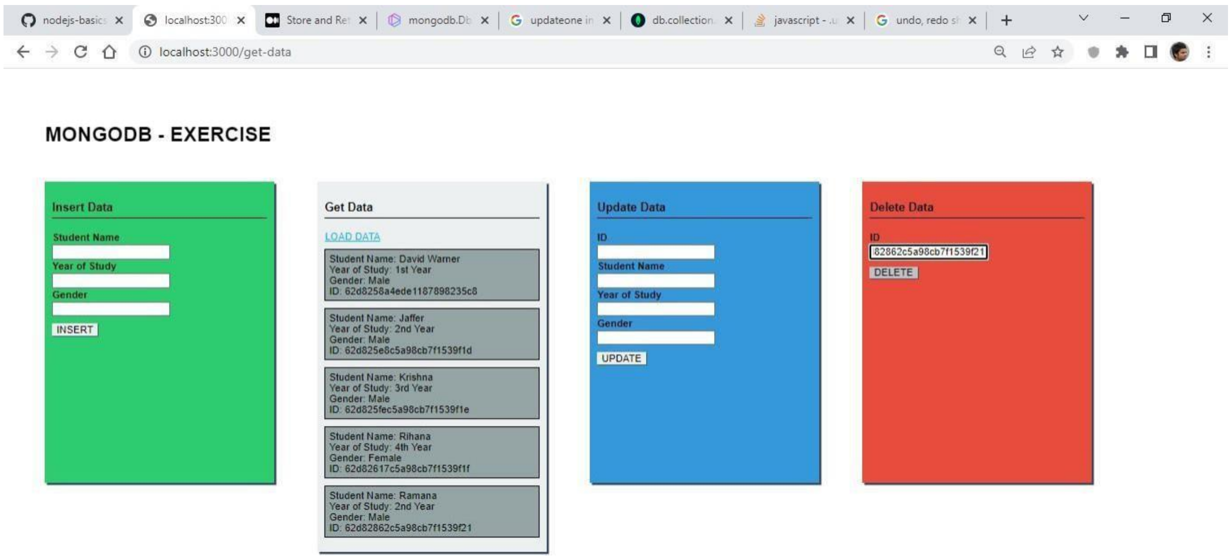
UPDATE

Delete Data

ID

DELETE

Delete Data:



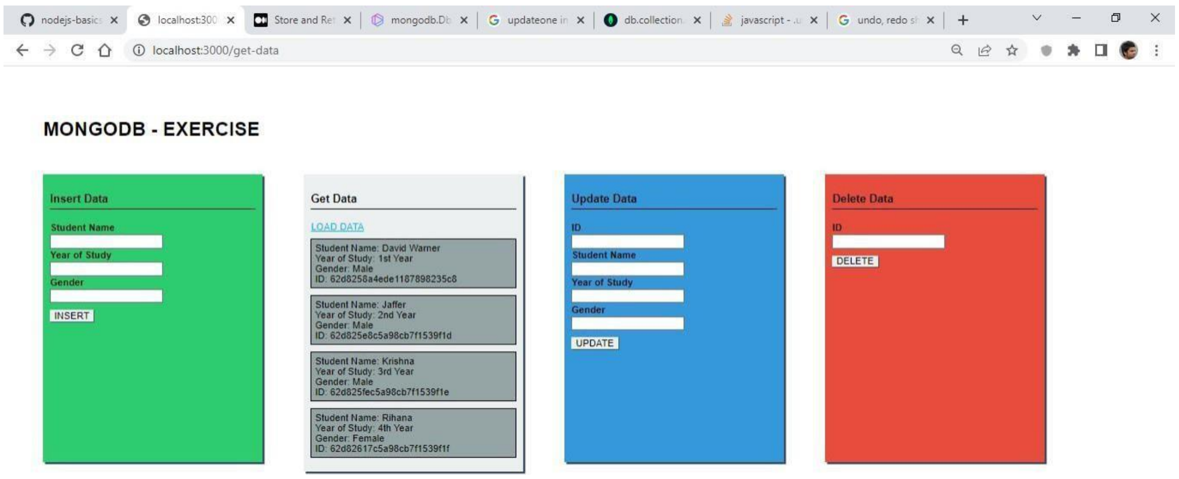
Student Name: David Warner
Year of Study: 1st Year
Gender: Male
ID: 62d8258a4ede1187898235c8

Student Name: Jaffer
Year of Study: 2nd Year
Gender: Male
ID: 62d825e8c5a98cb7f1539f1d

Student Name: Krishna
Year of Study: 3rd Year
Gender: Male
ID: 62d825fec5a98cb7f1539f1e

Student Name: Rihana
Year of Study: 4th Year
Gender: Female
ID: 62d82617c5a98cb7f1539f1f

Student Name: Ramana
Year of Study: 2nd Year
Gender: Male
ID: 62d82862c5a98cb7f1539f21



Student Name: David Warner
Year of Study: 1st Year
Gender: Male
ID: 62d8258a4ede1187898235c8

Student Name: Jaffer
Year of Study: 2nd Year
Gender: Male
ID: 62d825e8c5a98cb7f1539f1d

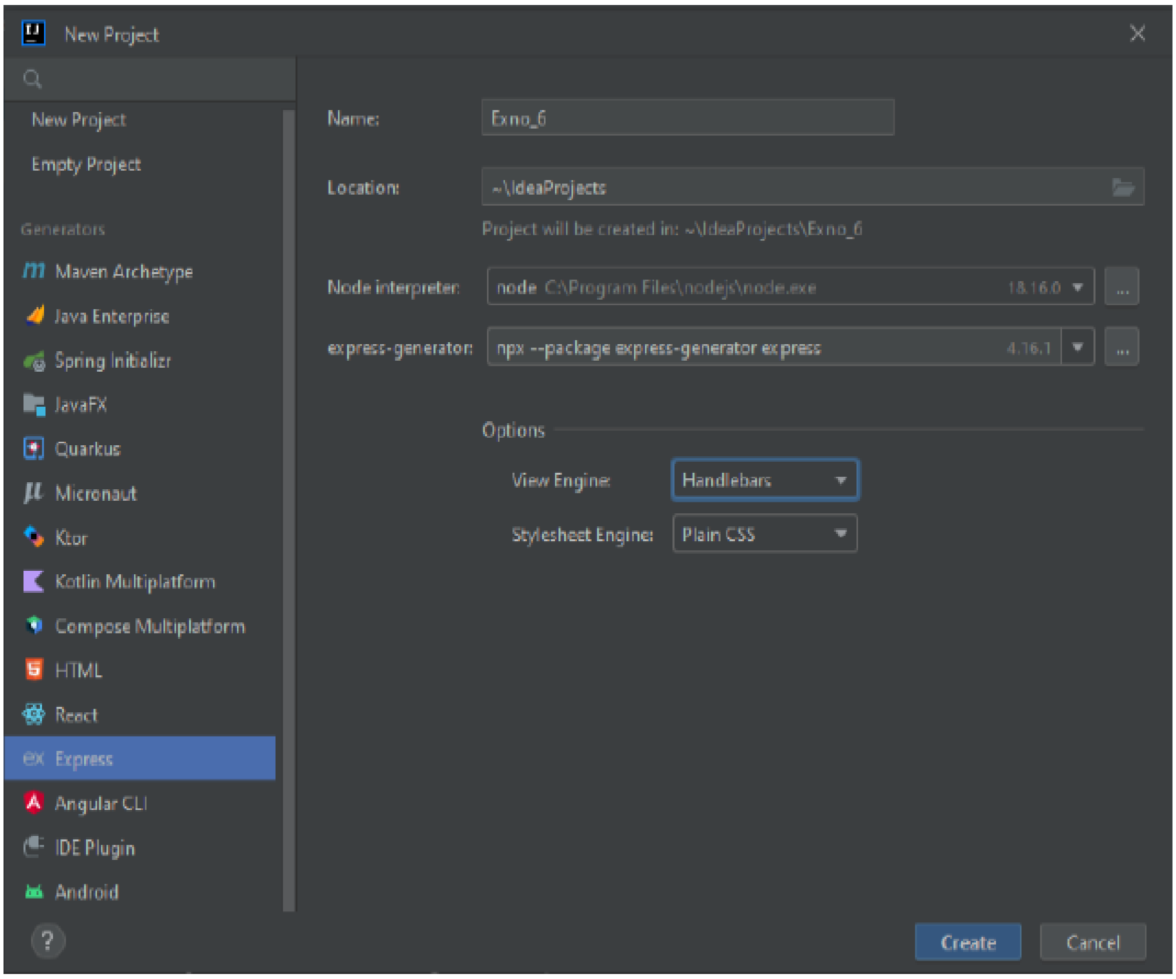
Student Name: Krishna
Year of Study: 3rd Year
Gender: Male
ID: 62d825fec5a98cb7f1539f1e

Student Name: Rihana
Year of Study: 4th Year
Gender: Female
ID: 62d82617c5a98cb7f1539f1f

06.CRUD USING MYSQL AND NODEJS

Step 1: Open IntelliJ IDEA Ultimate JetBrains: Developer Tools

Step 2: Select New Project and then select Express from Generators, give name of the project and select Handlebars from the View Engine and Click on Create button.



Name: Exno_6

Location: ~\IdeaProjects

Project will be created in: ~\IdeaProjects\Exno_6

Node interpreter: node C:\Program Files\nodejs\node.exe 18.16.0

express-generator: npx --package express-generator express 4.16.1

Options

View Engine: Handlebars

Stylesheet Engine: Plain CSS

Step 3: Enter the following command in the Terminal to install MongoDB.

npm install --save mysql

index.hbs:

```
<h1>MYSQL CRUD -
EXERCISE</h1> <section
class="insert">
<h3>Insert Data</h3>
```

```
<form action="/insert"
method="post"> <div class="input">
<label for="name">Student Name</label>
<input type="text" id="name"
name="name"> </div>
<div class="input">
<label for="regno">Register No</label>
<input type="text" id="regno"
name="regno"> </div>
<div class="input">
<label for="gender">Gender</label>
<input type="text" id="gender"
name="gender"> </div>
<button
type="submit">INSERT</button>
</form>

</section>
<section
class="get">
<h3>Get Data</h3>
<a href="/get-data">LOAD
DATA</a> <div>
{{#each items }}
<h4>Student Name: {{ this.name }}</h4>
<h4>Registration Number: {{ this.regno
}}</h4> <h4>Gender: {{ this.gender }}</h4>
<br>
{{/each}}
</div>
</section>

<section
class="update">
<h3>Update Data</h3>
<form action="/update"
method="post"> <div class="input">
<label for="name">Student Name</label>
<input type="text" id="name"
name="name"> </div>
<div class="input">
<label for="regno">Registration No</label>
<input type="text" id="regno"
name="regno"> </div>
<div class="input">
<label for="gender">Gender</label>
<input type="text" id="gender"
name="gender"> </div>
<button
type="submit">UPDATE</button>
</form>
</section>
```

```
 <section
 class="delete">
 <h3>Delete Data</h3>
 <form action="/delete"
 method="post"> <div class="input">
 <label for="regno">Registration No</label>
 <input type="text" id="regno"
 name="regno"> </div>
 <button
 type="submit">DELETE</button>
 </form>
 </section>
```

style.css:

```
body {
padding:
50px;
font: 14px "Lucida Grande", Helvetica, Arial, sans-
serif; }

a {
color:
#00B7FF; }

section {
float: left;
background:
#bdc3c7; padding:
10px; margin: 30px;
width: 300px;
box-shadow: 3px 3px 1px
#34495e; min-height: 400px;
}

section:first-of-type
{ margin-left: 0;
}

section:last-of-type
{ margin-right: 0;
}

section h3 {
border-bottom: 1px solid
black; padding-bottom: 5px;
}

.insert {
background:
orangered; }
```

```
.get {
background:
white; }

.update {
background:green
; }

.delete {
background:
blue; }

.input label {
display: block;
font-weight:
bold; padding:
2px 0;
}

input,
button {
font: inherit;
}

button {
margin-top:
10px; border:
none;
box-shadow: 1px 1px 1px
#34495e; border-radius: 0;
background:
#ecf0f1; cursor:
pointer;
}

button:hover {
background:
#bdc3c7; }

.item {
margin: 10px 0;
padding: 5px;
background: #95a5a6;
border: 1px solid
black; }
```

index.js:

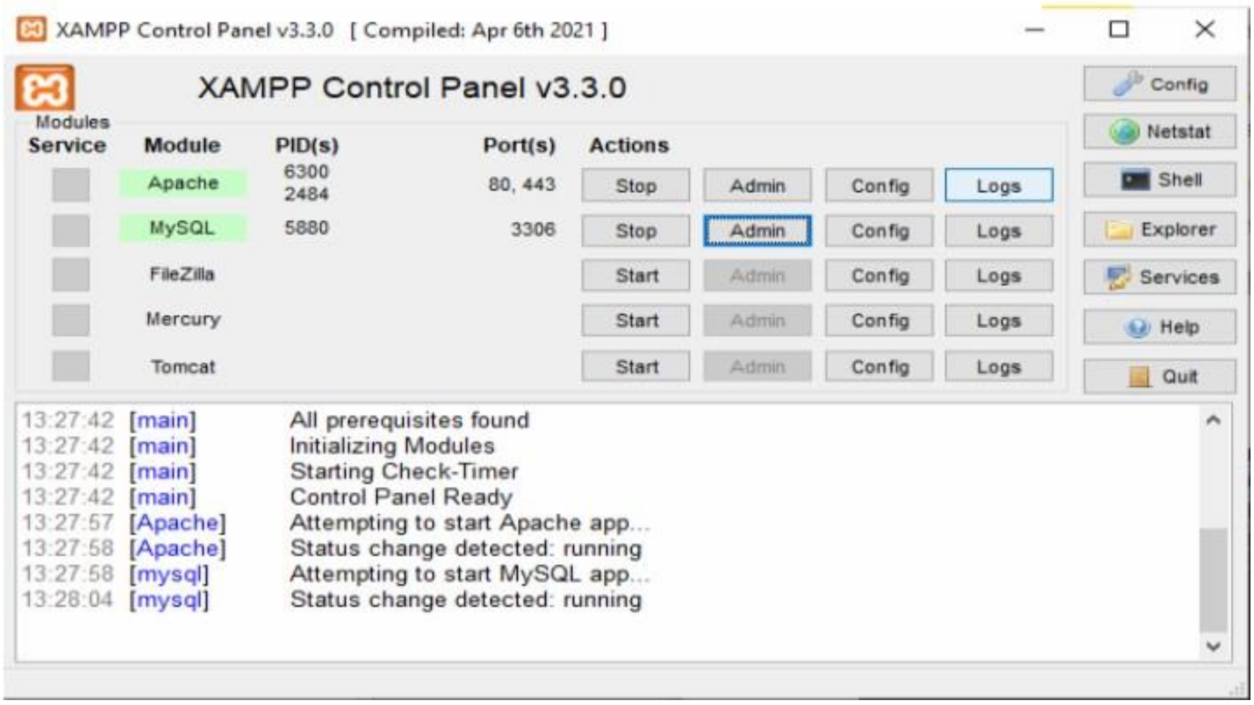
```
connection.connect((err) =>
{ if (err) {
console.error('Error connecting to MySQL database:',
err); }
```

```
else {
console.log('Connected to MySQL
database'); }
});


router.get('/', (req, res) =>
{ res.render('index');
});
router.get('/get-data', (req, res) => {
const query = 'SELECT * FROM student';

connection.query(query, (err, results) =>
{ if (err) {
console.error('Error retrieving data from MySQL:',
err); res.render('index', { items: [] });
} else {
res.render('index', { items: results
}); }
});
});

router.post('/insert', (req, res) => {
const { name, regno, gender } = req.body;
const query = 'INSERT INTO student (name, regno, gender) VALUES (?, ?,
?)'; connection.query(query, [name, regno, gender], (err, result) => {
if (err) {
console.error('Error inserting data into MySQL:',
err); }
res.redirect('/')
; });
});

router.post('/update', (req, res) => {
const { name,regno, gender } = req.body;
const query = 'UPDATE student SET name = ?,gender = ? WHERE
regno=?'; connection.query(query, [name,gender,regno], (err, result) => {
if (err) {
console.error('Error updating data in MySQL:',
err); }
res.redirect('/')
; });
});

router.post('/delete', (req, res) =>
{ const { regno } = req.body;
const query = 'DELETE FROM student WHERE regno =
?'; connection.query(query, [regno], (err, result) => {
```

```
if (err) {
console.error('Error deleting data from MySQL:',
err); }
res.redirect('/')
; });
});
module.exports = router;
```

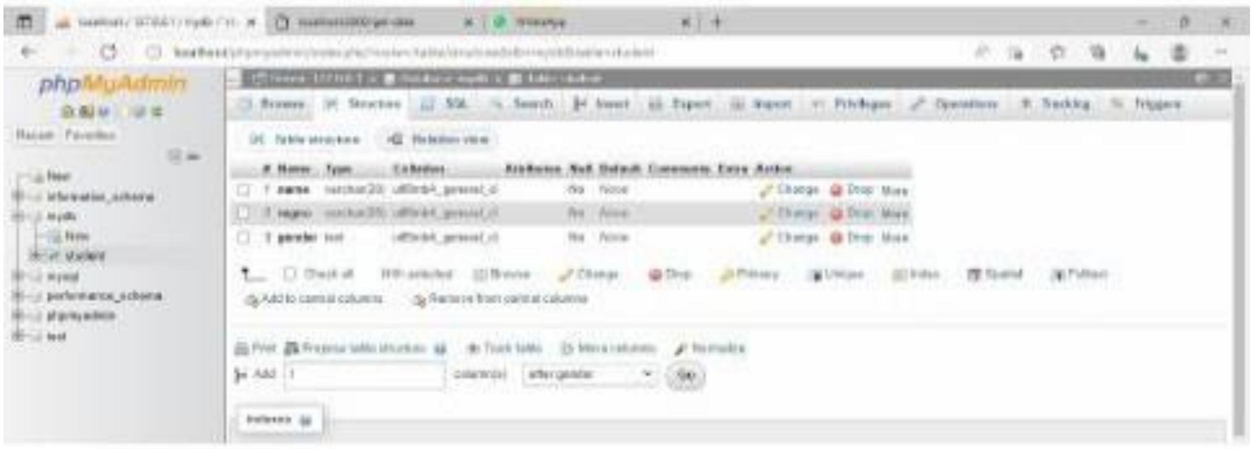
Step 4: Run mysql server ,create new database call mydb and create table called student with name, regno,gender.

Step 5: Save the program and run index.js file using npm start or run button.


XAMPP Control Panel v3.3.0 [Compiled: Apr 6th 2021]
XAMPP Control Panel v3.3.0
Config
Netstat
Shell
Explorer
Services
Help
Quit

Modules Service	Module	PID(s)	Port(s)	Actions			
	Apache	6300 2484	80, 443	Stop	Admin	Config	Logs
	MySQL	5880	3306	Stop	Admin	Config	Logs
	FileZilla			Start	Admin	Config	Logs
	Mercury			Start	Admin	Config	Logs
	Tomcat			Start	Admin	Config	Logs

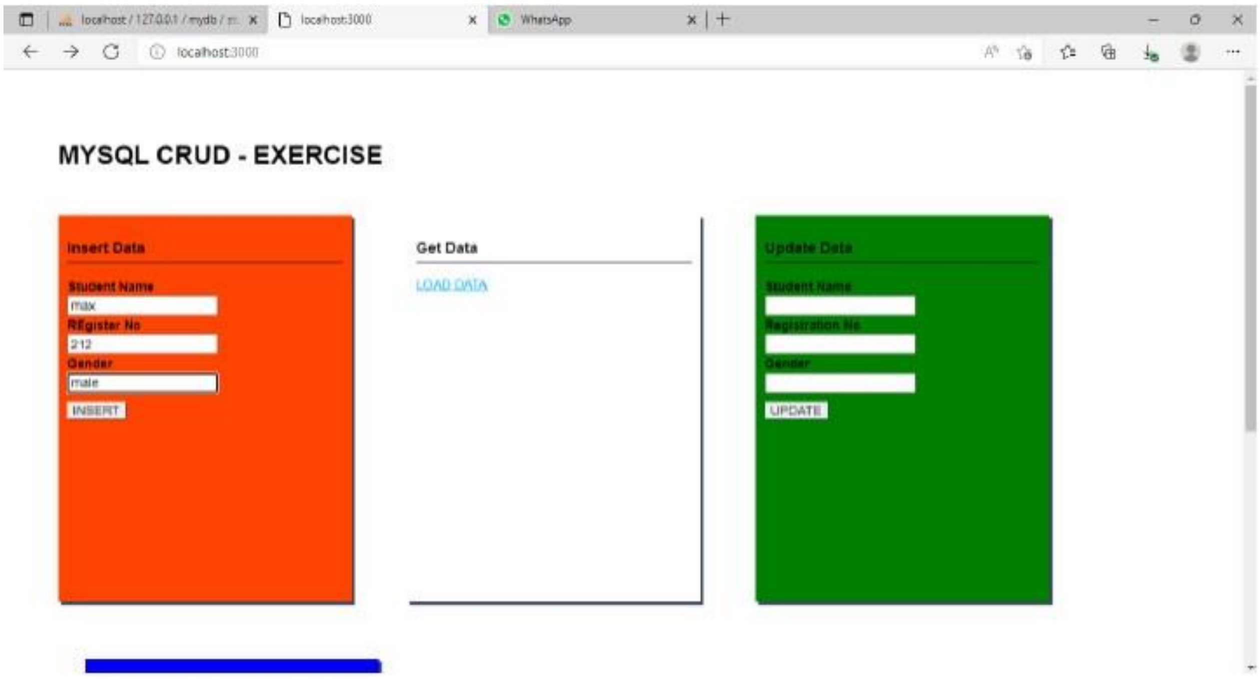
```
13:27:42  [main]     All prerequisites found
13:27:42  [main]     Initializing Modules
13:27:42  [main]     Starting Check-Timer
13:27:42  [main]     Control Panel Ready
13:27:57  [Apache]   Attempting to start Apache app...
13:27:58  [Apache]   Status change detected: running
13:27:58  [mysql]    Attempting to start MySQL app...
13:28:04  [mysql]    Status change detected: running
```



#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	name	varchar(20)	utf8mb4_general_ci		No	None			Change Drop More
2	regno	varchar(20)	utf8mb4_general_ci		No	None			Change Drop More
3	gender	text	utf8mb4_general_ci		No	None			Change Drop More

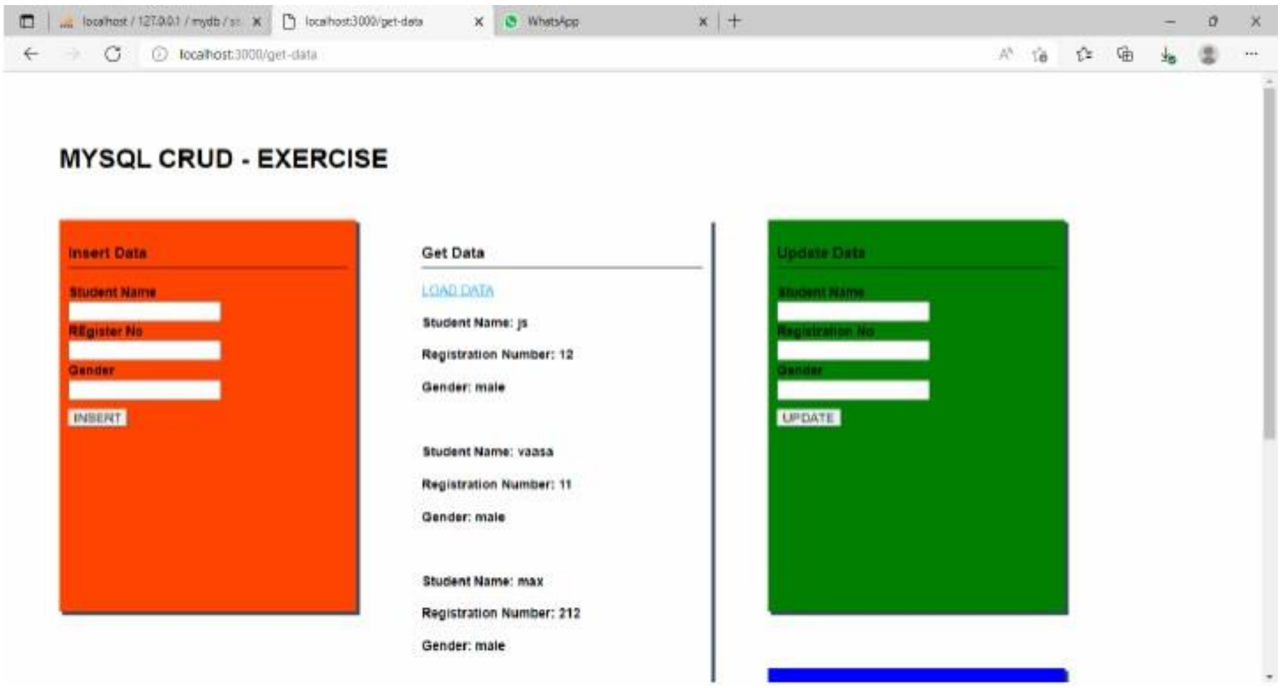
Output:

Insert Data:



MYSQL CRUD - EXERCISE

Read Data:



MYSQL CRUD - EXERCISE

Student Name: js

Registration Number: 12

Gender: male

Student Name: vaasa

Registration Number: 11

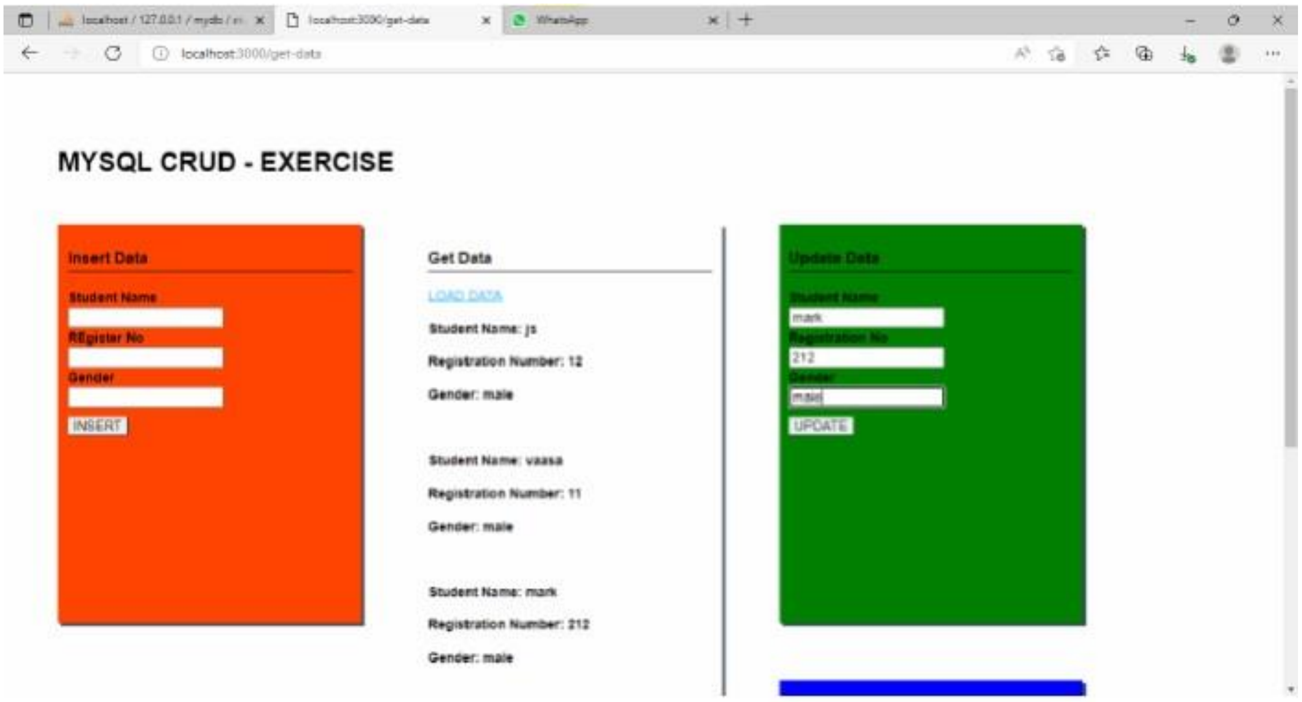
Gender: male

Student Name: max

Registration Number: 212

Gender: male

Update data:


localhost:3000/get-data

MYSQL CRUD - EXERCISE

Insert Data

Student Name

REgister No

Gender

INSERT

Get Data

LOAD DATA

Student Name: js

Registration Number: 12

Gender: male

Student Name: vaasa

Registration Number: 11

Gender: male

Student Name: mark

Registration Number: 212

Gender: male

Update Data

Student Name

mark

Registration No

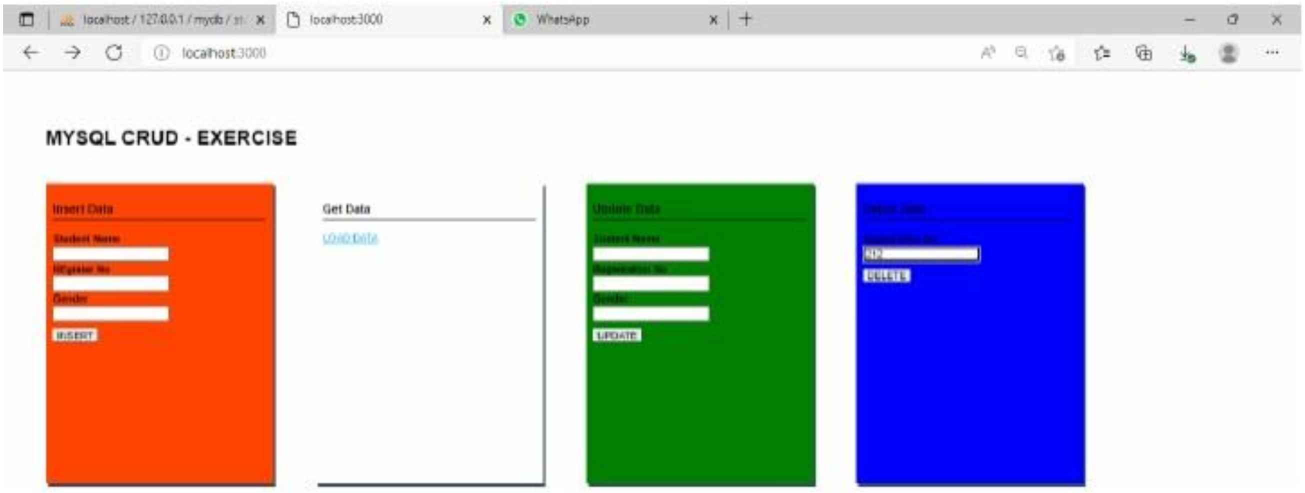
212

Gender

male

UPDATE

Delete Data:


localhost:3000

MYSQL CRUD - EXERCISE

Insert Data

Student Name

REgister No

Gender

INSERT

Get Data

LOAD DATA

Update Data

Student Name

Registration No

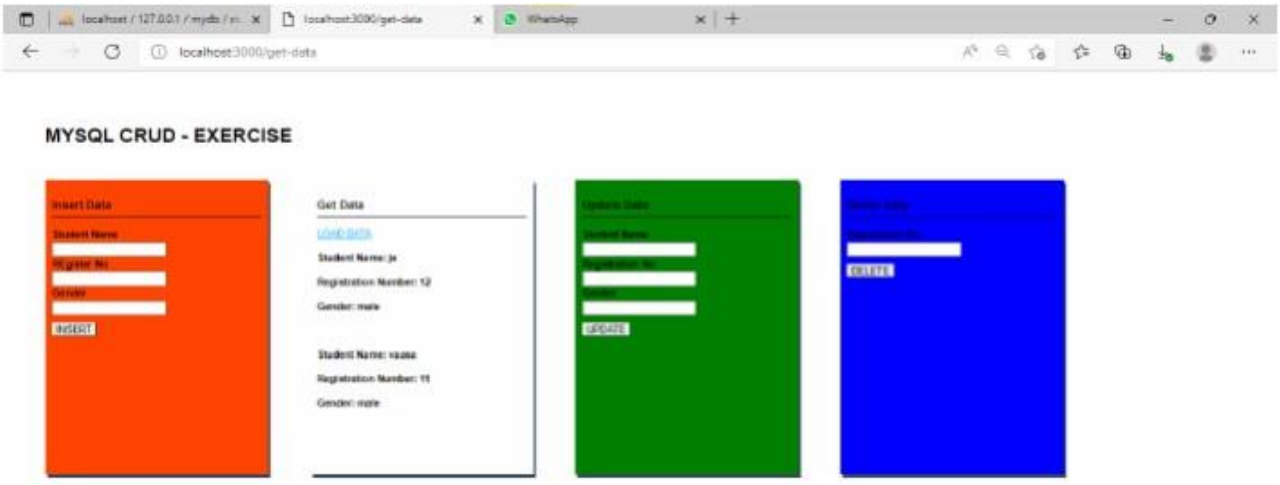
Gender

UPDATE

[illegible]

212

DELETE



MYSQL CRUD - EXERCISE

Student Name: ja

Registration Number: 12

Gender: male

Student Name: vaasa

Registration Number: 11

Gender: male

07.CREATE A COUNTER USING REACTJS

Initial Setup : The npx is a CLI tool used to install and manage dependencies in the npm registry. NPX comes pre-bundled with npm 5.2+,else we can install it using the following command:

npm i -g npx // -g flag indicates global installation

Creating Ract Application :

Step 1: Create a React application using the following command :

npx create-react-app-counter

Step :2 After creating your project folder i.e.,counter , move to it using the following command :

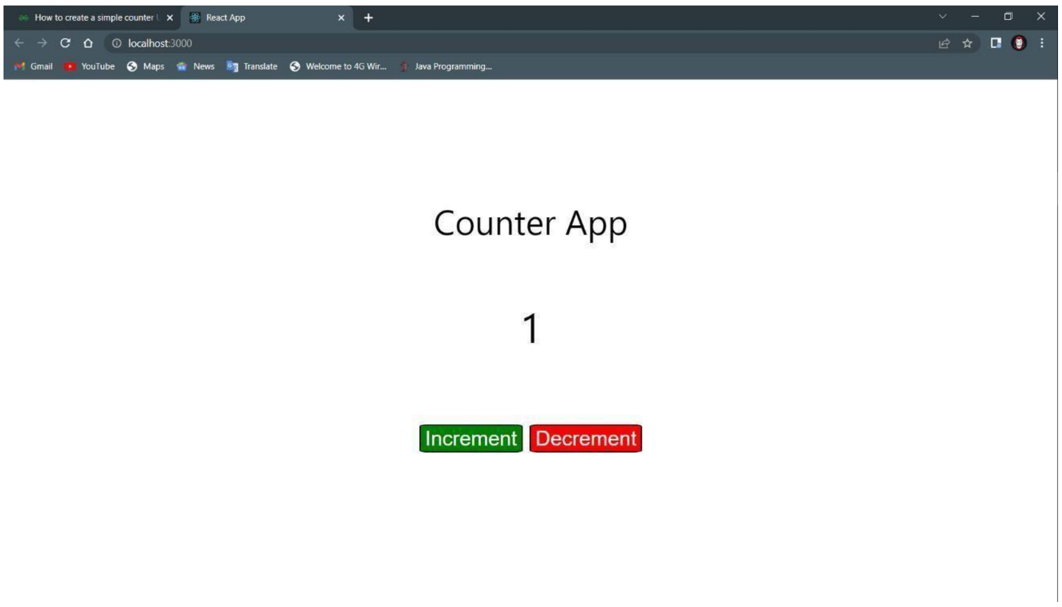
cd counter

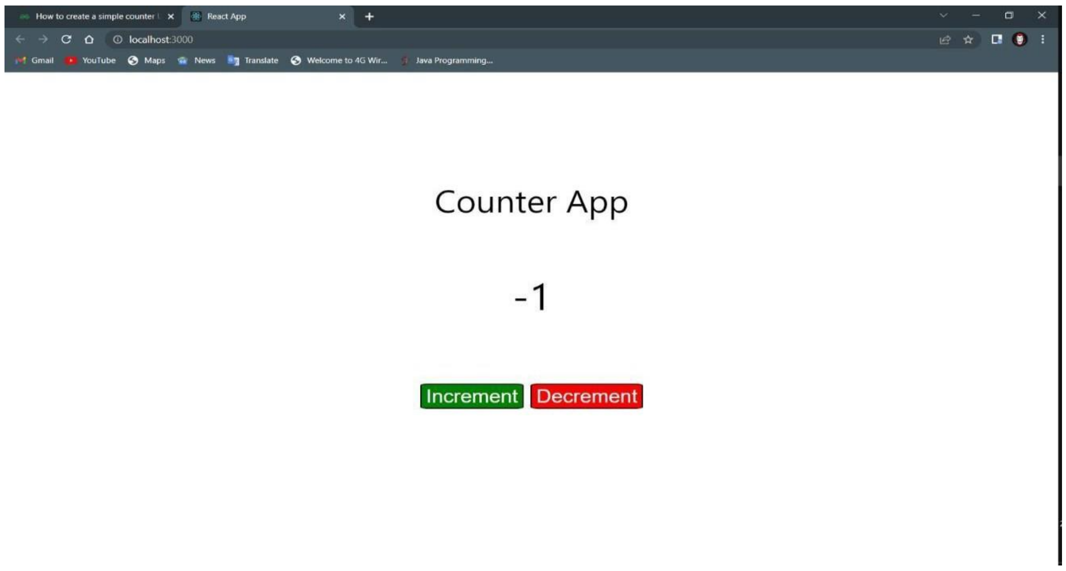
App.js:

```
import React. { useState } from "react";
import "./App.cs";
const App = () => {
  const [counter.setCounter] = useState(0)
  const handleClick1 = () => {
    setCounter(counter + 1)
  }
  const handleClick2 = () => {
    setCounter(counter - 1)
  }
return (
  <div style={{
    display: 'flex' ,
    flexDirection: 'column' ,
    alignItems: 'center' ,
    justiftContent: 'center' ,
    fontSize: '300%' ,
    position: 'absolute' ,
    width: '100%' ,
  height: '100%' ,
  top: '-15%' ,
}}>
  Counter App
  <div style={{
    fontSize: '120%' ,
    position: 'relative' ,
    top: '10vh' ,
  }}>
    {counter}
  </div>
  <div className="buttons">
    <button style={{
      fontSize: '60%' ,
      position: 'relative' ,
      top: '20%' ,
      marginRight: '5px' ,
      backgroundColor: 'green' ,
      borderRadius: '8%' ,
      color: 'white' ,
    }}
      onClick={handleClick1}>Increment</button>
    <button style={{
      fontSize: '60%' ,
      position: 'relative' ,
      top: '20%' ,
      marginLight: '5px' ,
      backgroundColor: 'red' ,
      borderRadius: '8%' ,
      color: 'white' ,
    }}
```

```
            onClick={handleClick2}>Decrement</button>
        </div>
     </div>
   }
 }
export default App
```

Output:


How to create a simple counter
React App
localhost:3000
Gmail
YouTube
Maps
News
Translate
Welcome to 4G Wir...
Java Programming...
Counter App
1
Increment
Decrement


How to create a simple counter
React App
localhost:3000
Gmail
YouTube
Maps
News
Translate
Welcome to 4G Wir...
Java Programming...
Counter App
-1
Increment
Decrement

08.TODO APPLICATION USING REACTJS

Step 1: Create a React application

- npm: npx create-react-app todo-list

Step 2: cd into todo-list and run npm start

The project should now be served on localhost:3000

App.js

```
import React, { useState } from 'react';
import data from "./data.json";
import Header from "./Header";
import ToDoList from "./ToDoList";
import ToDoForm from './ToDoForm';

function App() {

 const [ toDoList, setToDoList ] = useState(data);

 const handleToggle = (id) => {
  let mapped = toDoList.map(task => {
   return task.id === Number(id) ? { ...task, complete: !task.complete } : { ...task};
  });
  setToDoList(mapped);
 }

 const handleFilter = () => {
  let filtered = toDoList.filter(task => {
   return !task.complete;
  });
  setToDoList(filtered);

  }
  const addTask = (userInput ) => {
    let copy = [...toDoList];
    copy = [...copy, { id: toDoList.length + 1, task: userInput, complete: false }];
    setToDoList(copy);
  }

  return (
    <div className="App">
      <Header />
      <ToDoList toDoList={toDoList} handleToggle={handleToggle}
handleFilter={handleFilter}/>
      <ToDoForm addTask={addTask}/>
    </div>
  );
}

export default App;
```

Header.js

```
import React from 'react';
const Header = () => {
    return (
        <header>
            <h1>To Do List</h1>
        </header>
    );
};
export default Header;
```

data.json

```
[{
    "id": 1,
    "task": "Give dog a bath",
    "complete": true
  }, {
    "id": 2,
    "task": "Do laundry",
    "complete": true
  }, {
    "id": 3,
    "task": "Vacuum floor",
    "complete": false
  }, {
    "id": 4,
    "task": "Feed cat",
    "complete": true
  }, {
    "id": 5,
    "task": "Change light bulbs",
    "complete": false
  }, {
    "id": 6,
    "task": "Go to Store",
    "complete": true
  }, {
    "id": 7,
    "task": "Fill gas tank",
    "complete": true
  }, {
    "id": 8,
  "task": "Change linens",
  "complete": false
}, {
  "id": 9,
  "task": "Rake leaves",
  "complete": true
}, {
  "id": 10,
  "task": "Bake Cookies",
  "complete": false
}, {
  "id": 11,
  "task": "Take nap",
  "complete": true
}, {
  "id": 12,
  "task": "Read book",
  "complete": true
}, {
  "id": 13,
  "task": "Exercise",
  "complete": false
}, {
  "id": 14,
  "task": "Give dog a bath",
  "complete": false
}, {
  "id": 15,
  "task": "Do laundry",
  "complete": false
}, {
  "id": 16,
  "task": "Vacuum floor",
  "complete": false
 }, {
  "id": 17,
  "task": "Feed cat",
  "complete": true
 }, {
  "id": 18,
  "task": "Change light bulbs",
  "complete": false
 }, {
  "id": 19,
  "task": "Go to Store",
  "complete": false
 }, {
  "id": 20,
  "task": "Fill gas tank",
  "complete": false
 }]
```

ToDoList.js

```
import React from 'react';
import ToDo from './ToDo';

const ToDoList = ({toDoList, handleToggle, handleFilter}) => {
  return (
    <div>
      {toDoList.map(todo => {
        return (
          <ToDo todo={todo} handleToggle={handleToggle} handleFilter={handleFilter}/>
```

```
            )
          })}
          <button style={{margin: '20px'}} onClick={handleFilter}>Clear Completed</button>
      </div>
   );
};
export default ToDoList;
```

ToDo.js

```
import React from 'react';
const ToDo = ({todo, handleToggle}) => {

   const handleClick = (e) => {
      e.preventDefault()
      handleToggle(e.currentTarget.id)
   }

   return (
      <div id={todo.id} key={todo.id + todo.task} name="todo" value={todo.id}
onClick={handleClick} className={todo.complete ? "todo strike" : "todo"}>
          {todo.task}
      </div>
   );
};
export default ToDo;
```

ToDoForm.js

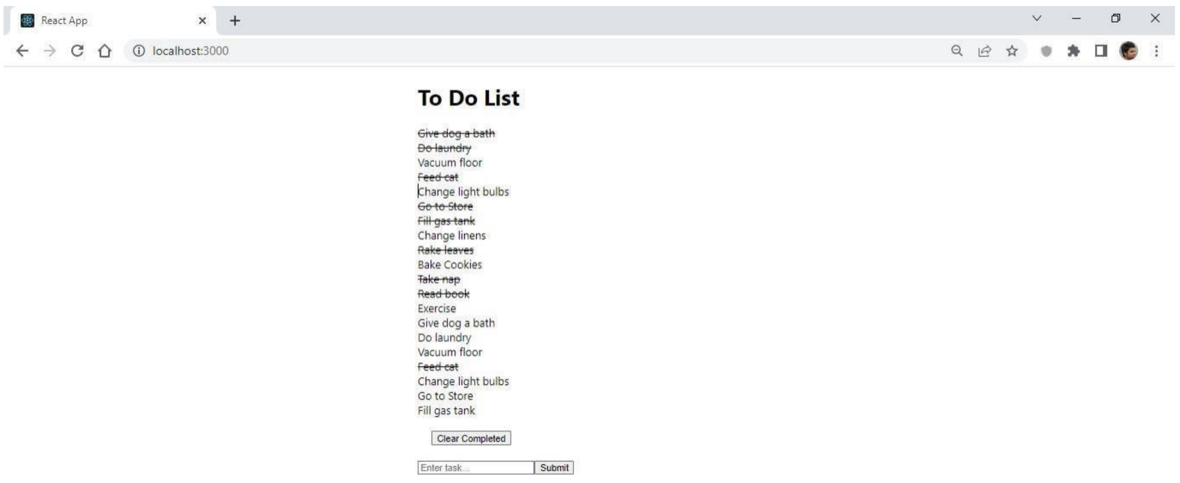
```
import React, { useState } from 'react';
const ToDoForm = ({ addTask }) => {

   const [ userInput, setUserInput ] = useState('');

   const handleChange = (e) => {

   setUserInput(e.currentTarget.value)

   }
```

```
  const handleSubmit = (e) => {
    e.preventDefault();
    addTask(userInput);
    setUserInput("");
  }
  return (
    <form onSubmit={handleSubmit}>
      <input value={userInput} type="text" onChange={handleChange} placeholder="Enter task..."/>
      <button>Submit</button>
    </form>
  );
};
export default ToDoForm;
```

index.css

```
body {
 max-width: 500px;
 margin: auto;
 font-family: -apple-system, BlinkMacSystemFont, 'Segoe UI', 'Roboto', 'Oxygen',
  'Ubuntu', 'Cantarell', 'Fira Sans', 'Droid Sans', 'Helvetica Neue',
  sans-serif;
 -webkit-font-smoothing: antialiased;
 -moz-osx-font-smoothing: grayscale;
}
code {
font-family: source-code-pro,Menlo, Monaco,Consolas, 'CourierNew',
monospace;
}
.todo {
cursor: pointer;
}
.strike { text-decoration: line-through; }
```

Output:



React App

localhost:3000

```
To Do List

~~Give dog a bath~~
~~Do laundry~~
Vacuum floor
~~Feed cat~~
Change light bulbs
~~Go to Store~~
~~Fill gas tank~~
Change linens
~~Rake leaves~~
Bake Cookies
~~Take nap~~
~~Read book~~
Exercise
Give dog a bath
Do laundry
Vacuum floor
~~Feed cat~~
Change light bulbs
Go to Store
Fill gas tank

[Clear Completed]

[Enter task...] [Submit]
```

09.SIGNUP AND LOGIN SYSTEM WITH NODE.JS, EXPRESS, AND MYSQL

Step1: Create a new directory called nodesignup

Step2: Go to terminal in visual studio code

- Run the command: npm init - it will prompt us to enter a package name, enter: login.
- When it prompts to enter the entry point, enter login.js

Step3: Install below commands using terminal

- Express - Install with command: npm install express --save.
- Express Sessions - Install with command: npm install express-session --save.
- MySQL for Node.js - Install with command: npm install mysql --save.

Step4: Create required folder and files

```
\-- nodesignup
    |-- signup.html
    |-- login.html
    |-- login.js
   \-- static
         |-- style.css
```

Step5: In mysql,

1. CREATE DATABASE nodelogin
2. CREATE TABLE IF NOT EXISTS `accounts` (

```
   `id` int(11) NOT NULL AUTO_INCREMENT,
   `username` varchar(50) NOT NULL,
   `password` varchar(255) NOT NULL,
   `email` varchar(100) NOT NULL,
   PRIMARY KEY (`id`)
    ) AUTO_INCREMENT=2 DEFAULT CHARSET=utf8;
```

signup.html

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width,minimum-scale=1">
    <title>Signup</title>
    <link href="/style.css" rel="stylesheet" type="text/css">
  </head>
  <body>
    <div class="login">
      <h1>Signup</h1>
      <form action="/auth" method="post">
        <label for="username">USERNAME:
        </label>
        <input type="text" name="username" placeholder="Username" id="username"
required><br>
        <label for="email">EMAIL:
        </label>
        <input type="text" name="email" placeholder="eMail" id="email" required><br>
        <label for="password">PASSWORD:
        </label>
        <input type="password" name="password" placeholder="Password" id="password"
required><br>
        <label for="confirmpassword">CONFIRMPASSWORD:
        </label>
        <input type="password" name="confirmpassword" placeholder="confirmpassword"
id="confirmpassword" required><br>
```

```
        <input type="submit" value="Signup">
      </form>
    </div>
  </body>
</html>
```

login.html

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width,minimum-scale=1">
    <title>Login</title>
    <link href="/style.css" rel="stylesheet" type="text/css">
  </head>
  <body>
    <div class="login">
      <h1>Login</h1>
      <form action="/auth1" method="post">
        <label for="username">USERNAME:
        </label>
        <input type="text" name="username" placeholder="Username" id="username"
required>
        <label for="password">PASSWORD:
        </label>
        <input type="password" name="password" placeholder="Password" id="password"
required>
        <input type="submit" value="Login">
      </form>
    </div>
    </body>
</html>
```

style.css

```
 * {
    box-sizing: border-box;
    font-family: -apple-system, BlinkMacSystemFont, "segoe ui", roboto, oxygen, ubuntu,
 cantarell,     "fira sans", "droid sans", "helvetica neue", Arial, sans-serif;
    font-size: 16px;
 }
 body {
    background-color: #435165;
 }
 .login {
    width: 400px;
    background-color: #ffffff;
    box-shadow: 0 0 9px 0 rgba(0, 0, 0, 0.3);
    margin: 100px auto;
 }
 .login h1 {
    text-align: center;
    color: #5b6574;
    font-size: 24px;
    padding: 20px 0 20px 0;
    border-bottom: 1px solid #dee0e4;
 }
 .login form {
    display: flex;
    flex-wrap: wrap;
    justify-content: center;
    padding-top: 20px
}
```

```
.login form label {
display: flex;
    justify-content: center;
    align-items: center;
    width: 150px;
    height: 50px;
    flex-direction: row;
    background-color: #3274d6;
    color: #ffffff;
 }
 .login form input[type="password"], .login form input[type="text"] {
    width: 250px;
    height: 50px;
    border: 1px solid #dee0e4;
    margin-bottom: 20px;
    flex-direction: row;
    padding: 0 15px;
 }
 .login form input[type="submit"] {
    width: 100%;
    padding: 15px;
    margin-top: 20px;
    background-color: #3274d6;
    border: 0;
    cursor: pointer;
    font-weight: bold;
    color: #ffffff;
    transition: background-color 0.2s;
 }
 .login form input[type="submit"]:hover {
 background-color: #2868c7;
 transition: background-color 0.2s;
 }
```

login.js

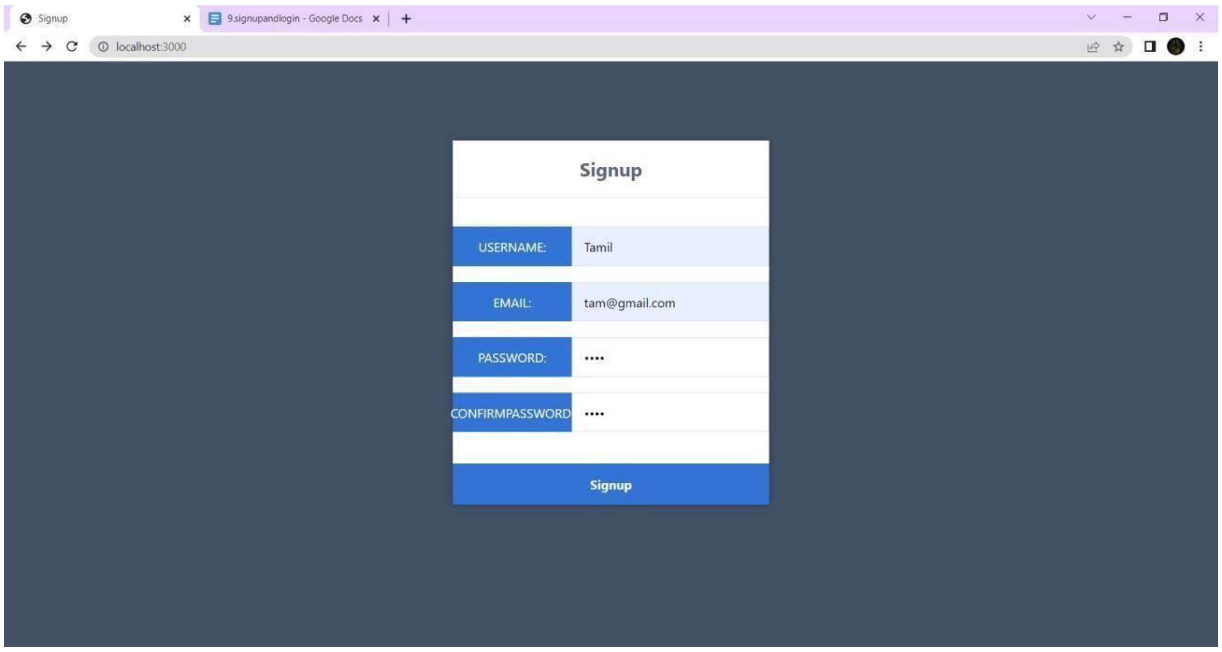
```
const mysql = require('mysql');
const express = require('express');
const session = require('express-session');
const path = require('path');

const connection = mysql.createConnection({
   host     : 'localhost',
   user     : 'root',
   password : 'root',
   database : 'nodelogin'
});

const app = express();

app.use(session({
   secret: 'secret',
   resave: true,
   saveUninitialized: true
}));
app.use(express.json());
app.use(express.urlencoded({ extended: true }));
app.use(express.static(path.join(_dirname, 'static')));

app.get('/', function(request, response) {
   response.sendFile(path.join(_dirname + '/signup.html'));
});
app.get('/login', function(request, response) {

         response.sendFile(path.join(_dirname+ '/login.html'));

});
```

```
app.post('/auth', function(request, response) {
   let username = request.body.username;
   let email = request.body.email;
   let password = request.body.password;
   let confirmpassword = request.body.confirmpassword;
   if (username && email && password && confirmpassword) {
   if (password===confirmpassword){
      connection.query('insert        into        accounts(username,password,email)values(?,?,?)',
[username,password,email], function(error, results, fields) {
         if (error) throw error;
         else if(!error) {
            request.session.signup = true;
            request.session.username = username;
            response.redirect('/login');
         } else {
            response.send('Incorrect details');
         }
         response.end();
      });
   }
}
else {
      response.send('Please correct details');
      response.end();
   }
});

app.post('/auth1', function(request, response) {
let username = request.body.username;
let password = request.body.password;
```

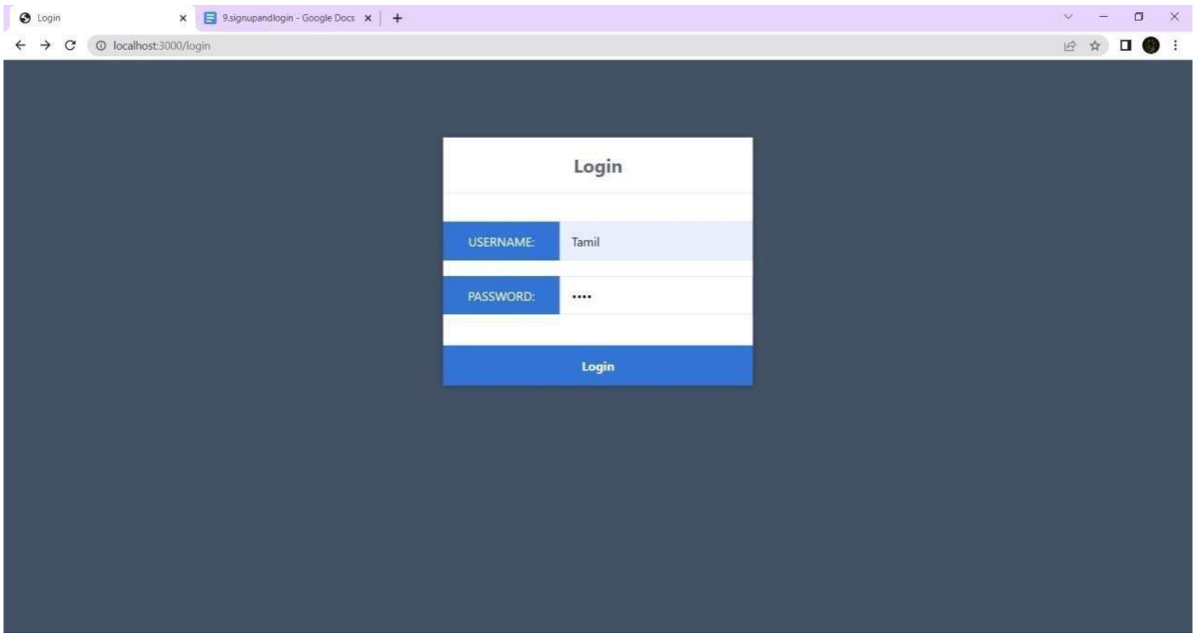
```
    if (username && password) {
        connection.query('SELECT * FROM accounts WHERE username = ? AND password = ?',
[username, password], function(error, results, fields) {
            if (error) throw error;
            if (results.length > 0) {
                request.session.loggedin = true;
                request.session.username = username;
                response.redirect('/home');
            } else {
                response.send('Incorrect Username and/or Password!');
            }
            response.end();
        });
    } else {
        response.send('Please enter Username and Password!');
        response.end();
    }
});

app.get('/home', function(request, response) {
    if (request.session.loggedin) {
        response.send('Welcome ' + request.session.username + '!!!');
    } else {
        response.send('Please login to view this page!');
    }
    response.end();
});

app.listen(3000);
```

Output:

Signup Page:



Login Page:



10.A DOCKER CONTAINER THAT WILL DEPLOY A NODEJS PING SERVER USING NODEJS IMAGE

Step1: https://docs.docker.com/desktop/install/windows-install/

Download and install Docker Desktop Application.

Step2: Create a new folder in VSCode, In VSCode terminal run npm in it -y and install Express with npm install -S express.

Step3: Create an index.js file.

Index.js

```
const express = require('express');
const app = express();
app.get('/plug', (req,res) => {
  res.send('pong');
});
app.listen(3000,  () => {
  console.log('listening on port 3000');
});
```

Step4:

In terminal, run your app with node index.js and go to chrome,check localhost:3000/ ping you able to see pong as a output.

Step5: Go to VSCode Extension and install Docker, then create a Dockerfile.

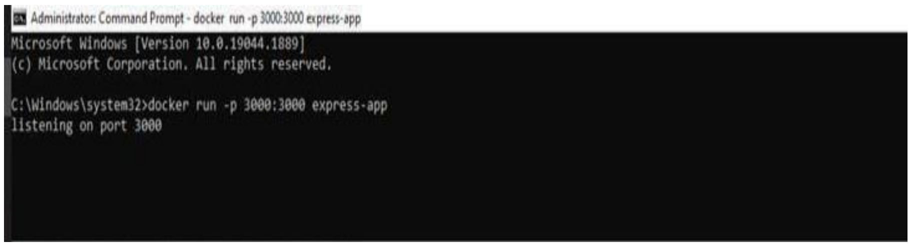
Dockerfile

```
FROM node:16-alpine
ADD .  ./
RUN npm install -ci
CMD ["node", "index.js"]
```

Step6: Go to CMD click run as administrator.

Step7: Enter a command docker build -t express-appis to assign a tag to our image.

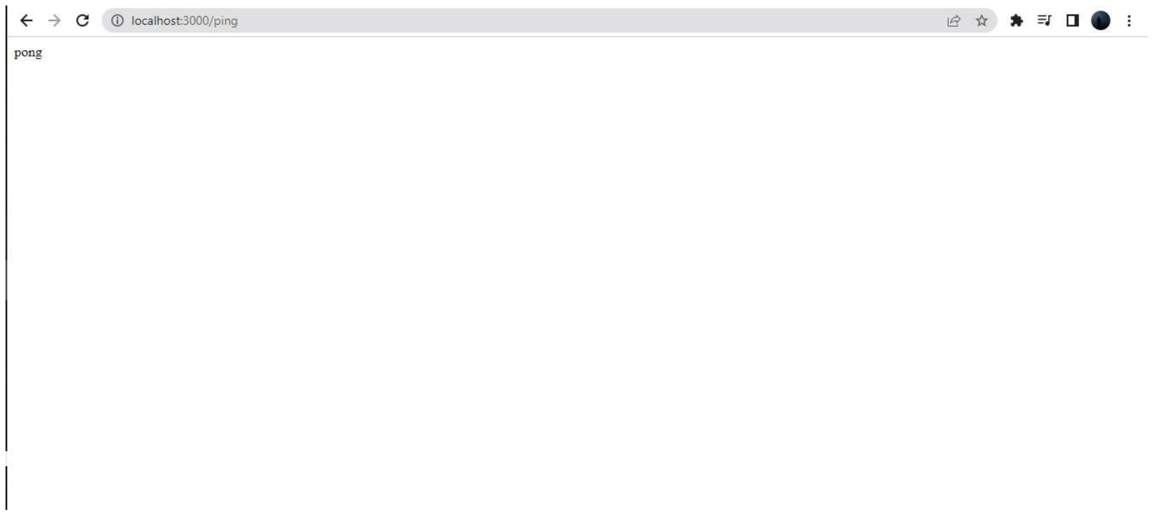
Step8: Run with command docker run -p 3000:3000 express-app



```
Administrator: Command Prompt - docker run -p 3000:3000 express-app
Microsoft Windows [Version 10.0.19044.1889]
(c) Microsoft Corporation. All rights reserved.

C:\Windows\system32>docker run -p 3000:3000 express-app
listening on port 3000
```

Step 9: Go to google chrome type localhost:3000/ping.

Output:



```
pong
```

11.DESIGN A WEB PAGE USING INLINE, INTERNAL AND EXTERNAL CSS

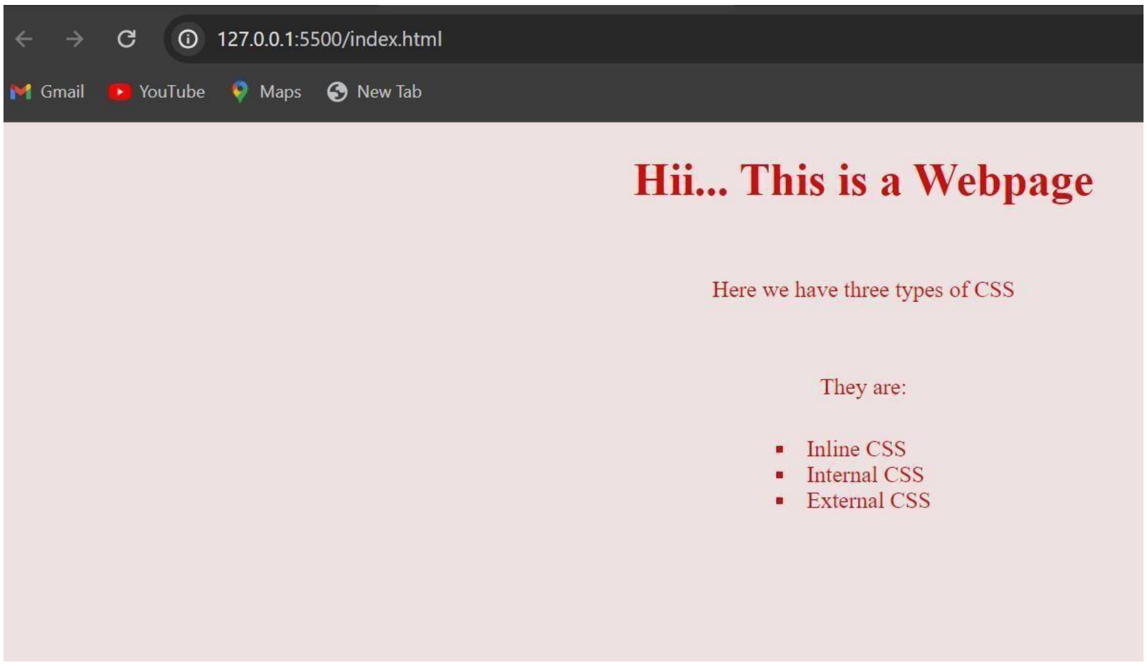
index.html:

```
<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <title>BASIC CSS</title>

  <link rel="stylesheet" href="style.css">

  <style>

    body{

      background-color: rgba(0, 0, 0); color:

      white;

    }

  </style>

   </head>

   <body>

  <h1 style="text-align: center;"> Hii... This is a Webpage </h1>

  <p>Here we have three types of
  CSS</p> <p>They are:</p>

  <ul class="list">

    <li>Inline CSS</li>

    <li>Internal CSS</li>

    <li>External CSS</li>

      </ul>

   </body>

   </html>
```

style.css:

```
 p{
    text-align: center;
    margin-top: 50px;
 }

 ul{
    margin-top: 25px;
    margin-left: 500px;
    list-style-position: inside;
    list-style-type: square;
 }
```

Output:



Hii... This is a Webpage

Here we have three types of CSS

They are:

- Inline CSS
- Internal CSS
- External CSS

12.CREATE WEBPAGE THAT USING AJAX AND PERFORM JQUERY OPERATIONS

index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>AJAX AND JQUERY</title>
</head>
<body>
   <h1>Hii... In this webpage we are performing AJAX and JQUERY operations</h1>
   <button id="loadContent">Load Content</button>
   <div id="loading"></div>
   <script  src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
   <script src="script.js"></script>
</body>
</html>
```

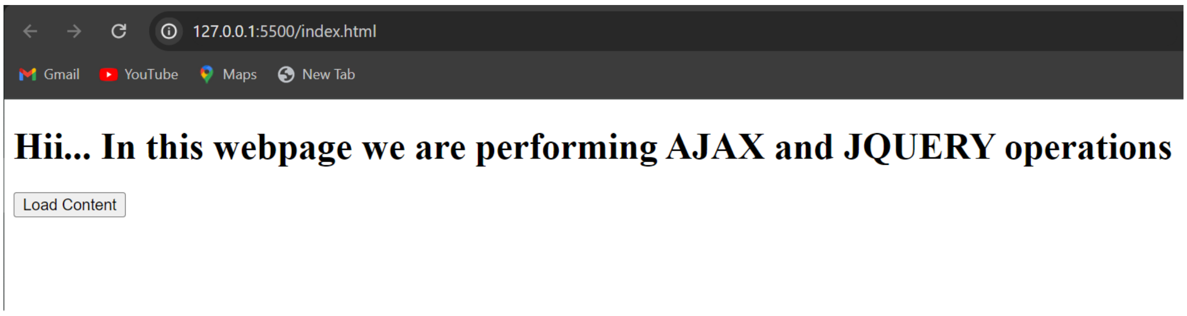
script.js:

```
$(document).ready(function(){
   $("#loadContent").click(function(){
      $.ajax({
         url:'content.html',
         method:'GET',
         success:function(response){
            $('#loading').html(response)
            $('#loading').find('p').css('color','green');
            $('#loading').append('<p>New content updated</p>')
            },error : function(xhr,status,error){
                $('#loading').html('<p>Error loading content. Please try again.</p>');
            }
        })
    })
})
```

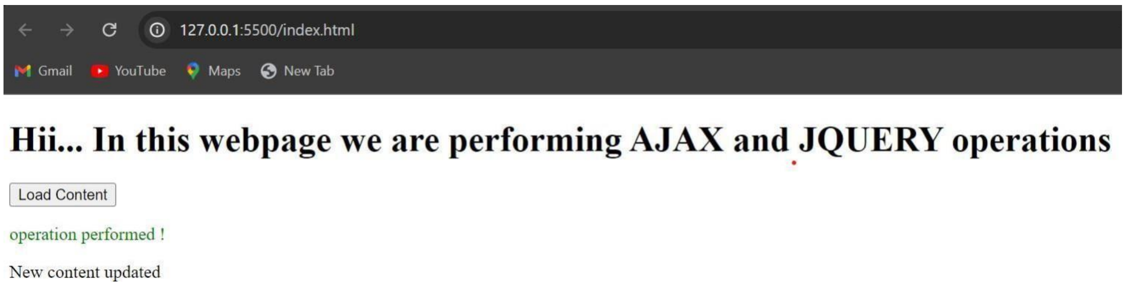
content.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>LOADED CONTENT</title>
</head>
<body>
    <p>operation performed !</p>
</body>
</html>
```

Output:



Hii... In this webpage we are performing AJAX and JQUERY operations



Hii... In this webpage we are performing AJAX and JQUERY operations

```
operation performed !

New content updated
```

13.DESIGN A WEBPAGE HAVING INPUT TAGS WITH SPELL CHECK AND EDITABLE TEXT

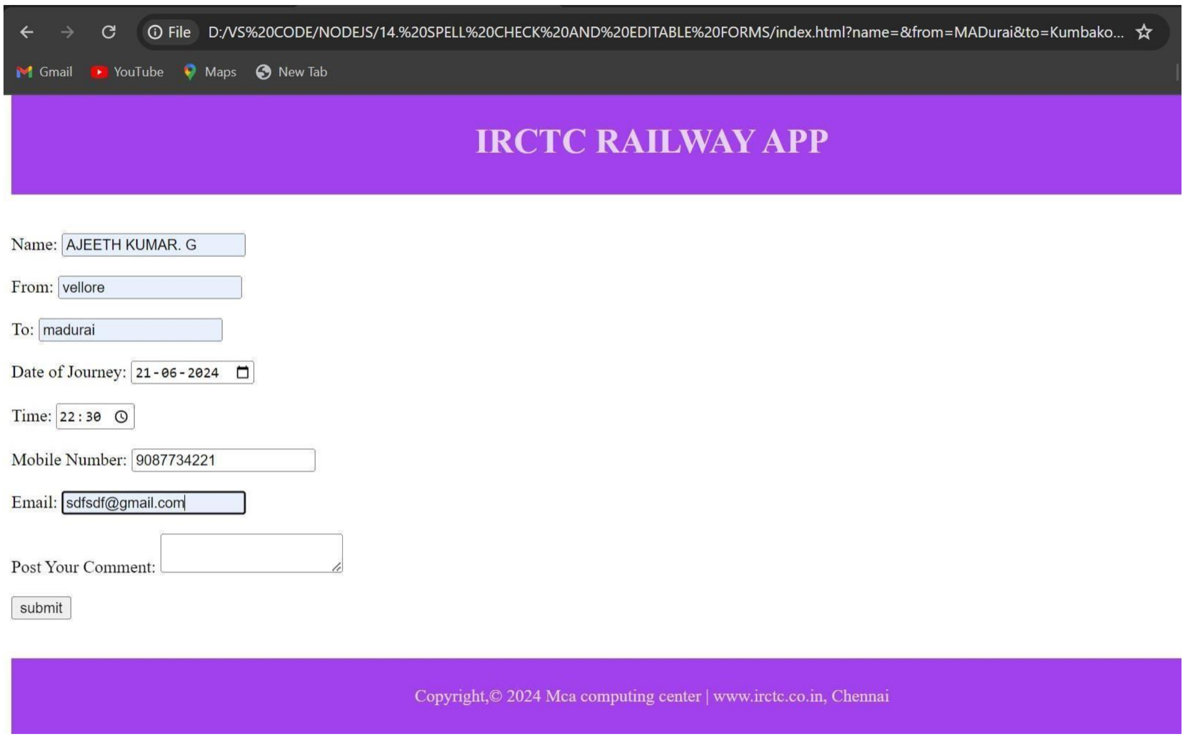
index.html:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>SPELL  CHECK AND EDITABLE TEXT</title>
    <style>
        header,footer{
            background-color: rgb(160, 65, 237);
            color:  rgb(230, 210, 247); ;
            text-align: center;
            padding: 10px;
        }
    </style>
</head>
<body>
    <header>
        <h1>IRCTC RAILWAY APP</h1>
    </header>
    <br><br>
        <section>
            <form action="">
                <label for="name">Name:</label>
                <input type="text" id="name" name="name" required>
                <br><br>
                <label for="from">From:</label>
```

```
          <input type="text" id="from" name="from" class="fromInput" required>


          <br><br>
          <label for="to">To:</label>
          <input type="text" id="to" name="to" class="toInput" required>


          <br><br>
          <label for="date">Date of Journey:</label>
          <input type="date" id="date" name="date" required>
          <br><br>
          <label for="time">Time:</label>
          <input type="time" id="time" name="time" required>
          <br><br>
          <label for="phn">Mobile Number:</label>
          <input type="number" id="phn" name="phn" required>
          <br><br>
          <label for="email">Email:</label>
          <input type="email" id="email" name="email" >
          <br><br>
          <label for="message">Post Your Comment: </label>
          <textarea name="message" id="message" rows="5" cols="50"></textarea>
          <br><br>
          <input type="submit" value="submit" onclick="checkJourney()">
           </form>
         </section>
    <br>
    <br>
 <footer>
    <p>Copyright,&COPY 2024 Mca computing center | www.irctc.co.in, Chennai</p>
 </footer>
```

```
   <script>

            function checkJourney(){
               const location = ['tambaram','chennai egmore','katpadi','vellore','arakonam','coimbatore',
            'chengalpattu','madurai','jolarpettai','salem','erode','tiruppur','villupuram',
            'tiruchirapalli','dindigul','perambur','thanjavur','tirunelveli','nagerkoil','tiruvallur',
            'avadi','perungulathur','mayiladuturai','hosur','nagapattinam','kumbakonam']

               let fromData = document.querySelector('.fromInput').value let
               toData = document.querySelector('.toInput').value

               let val1 = fromData.toLowerCase() let
               val2 = toData.toLowerCase()
               console.log(val1,val2)

               if(location.includes(val1) && location.includes(val2) && val1 !== val2){ let
                  start = val1.toUpperCase()
                  let end = val2.toUpperCase()
                  window.alert(`Your journey start at ${start} and ends at ${end}
was successfully recorded`)
               }
               else{
                  window.alert(`Please enter the place correctly`)
  }
            }
   </script>
</body>
</html>
```

Output:



D:/VS%20CODE/NODEJS/14.%20SPELL%20CHECK%20AND%20EDITABLE%20FORMS/index.html?name=&from=MADurai&to=Kumbako...

IRCTC RAILWAY APP

Name: AJEETH KUMAR. G

From: vellore

To: madurai

Date of Journey: 21-06-2024

Time: 22:30

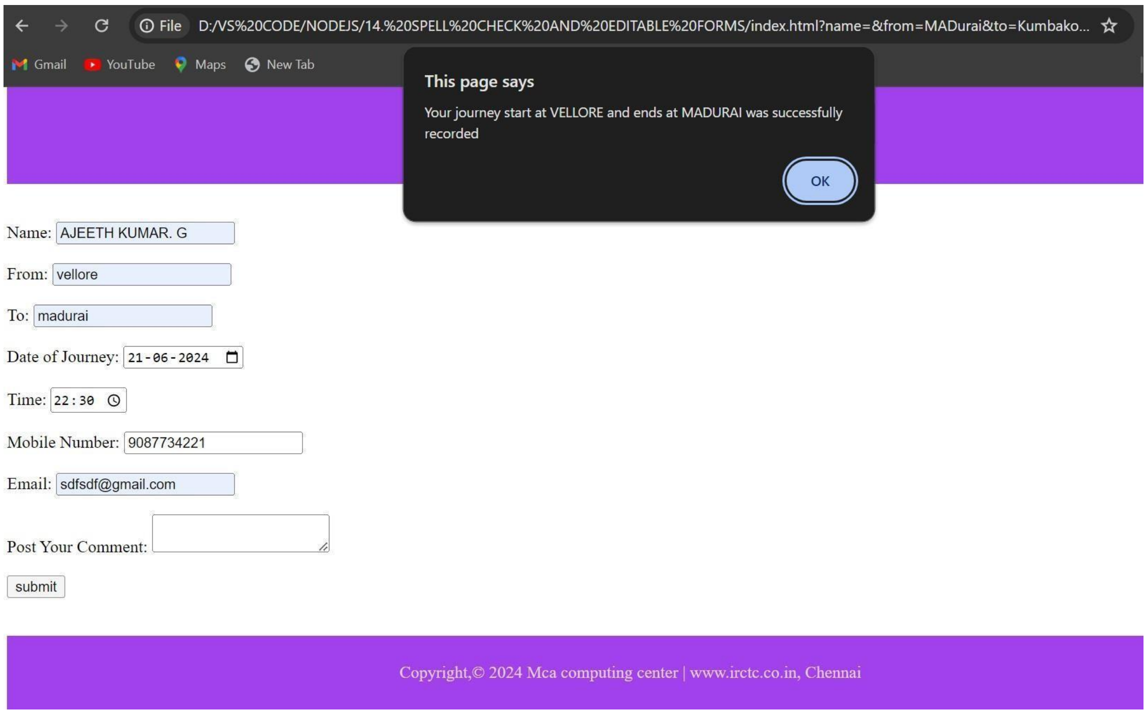
Mobile Number: 9087734221

Email: sdfsdf@gmail.com

Post Your Comment:

submit

Copyright,© 2024 Mca computing center | www.irctc.co.in, Chennai



D:/VS%20CODE/NODEJS/14.%20SPELL%20CHECK%20AND%20EDITABLE%20FORMS/index.html?name=&from=MADurai&to=Kumbako...

This page says

Your journey start at VELLORE and ends at MADURAI was successfully recorded

OK

Name: AJEETH KUMAR. G

From: vellore

To: madurai

Date of Journey: 21-06-2024

Time: 22:30

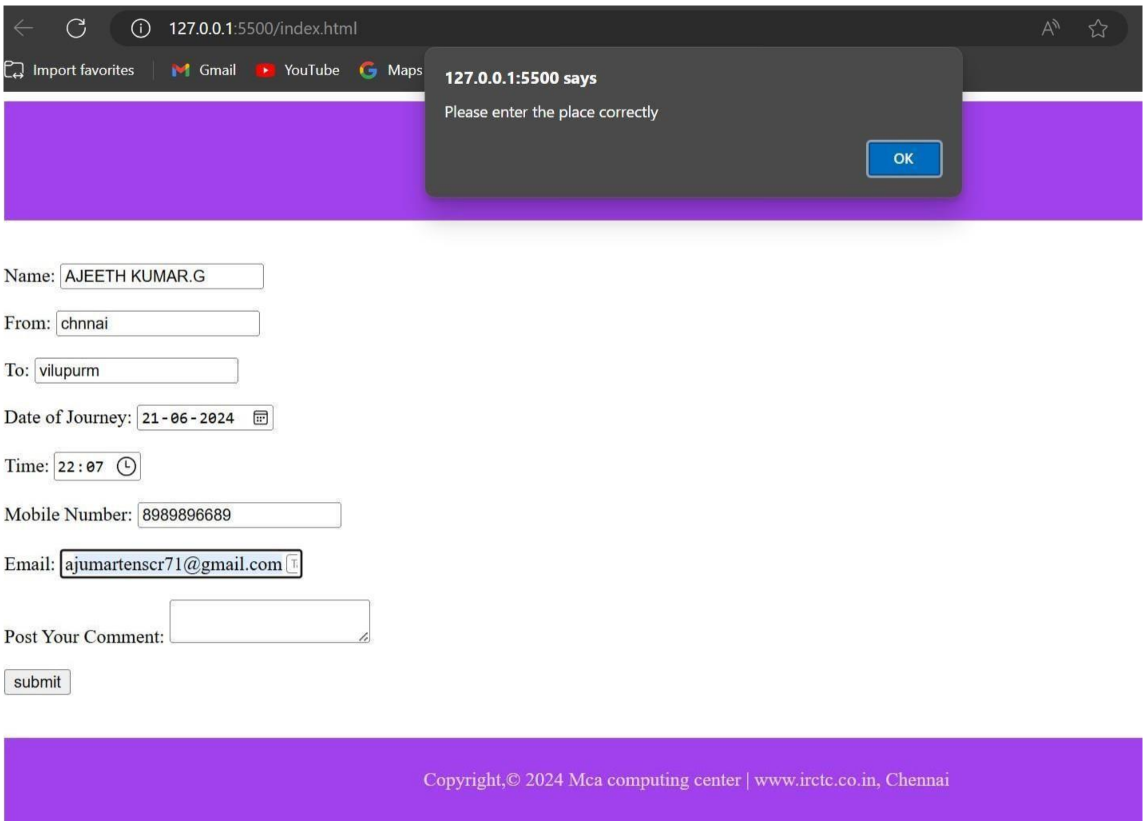
Mobile Number: 9087734221

Email: sdfsdf@gmail.com

Post Your Comment:

submit

Copyright,© 2024 Mca computing center | www.irctc.co.in, Chennai


127.0.0.1:5500/index.html

127.0.0.1:5500 says

Please enter the place correctly

OK

Name: AJEETH KUMAR.G

From: chnnai

To: vilupurm

Date of Journey: 21-06-2024

Time: 22:07

Mobile Number: 8989896689

Email: ajumartenscr71@gmail.com

Post Your Comment:

submit

Copyright,© 2024 Mca computing center | www.irctc.co.in, Chennai

14.BUILD A SIMPLE CALCULATOR USING REACT JS

App.js

```
import './App.css';
function App() {
  const [value,setValue] = useState('');
  function handleClick(e){
  setValue(value.concat(e.target.name))
  }
  function clear(){
    setValue("");
  }
  function handleDelete(){
    setValue(value.slice(0,-1))
  }
  function calculate(){
    try{
      setValue((eval(value)).toString())
    }
    catch{
      setValue("Error!")
    }
  }
  return (
   <div className="container">
    <div className="calculator">
      <form action="">
        <div className='display'>
          <input type="text" value={value} />
          </div>
```

```
<div>
  <input type="button" onClick={clear} name="AC" value="AC" />
  <input type="button" onClick={handleDelete} name="DE" value="DE" />
  <input type="button" name="." onClick={handleClick}  value="."/>
  <input type="button" name="/" onClick={handleClick} value="/" />
</div>

<div>
  <input type="button" name="7" value="7" onClick={handleClick}/>
  <input type="button" name="8" value="8" onClick={handleClick} />
  <input type="button" name="9" value="9" onClick={handleClick}/>
  <input type="button" name="*" value="*" onClick={handleClick}/>
</div>

<div>
  <input type="button" name="4" value="4" onClick={handleClick}/>
  <input type="button" name="5" value="5" onClick={handleClick}/>
  <input type="button" name="6" value="6" onClick={handleClick}/>
  <input type="button" name="+" value="+" onClick={handleClick}/>
</div>

<div>
  <input type="button" name="1" value="1" onClick={handleClick}/>
  <input type="button" name="2" value="2" onClick={handleClick}/>
  <input type="button" name="3" value="3" onClick={handleClick}/>
  <input type="button" name="-" value="-" onClick={handleClick}/>
</div>
```

```
        <div>
          <input type="button" name='00' value="00" />
          <input type="button" name='0' value="0" />
          <input type="button" name='=' value="=" className='equal' onClick={calculate} />
        </div>
      </form>
    </div>
   </div>
  );
}

export default App;
```

App.css

```
.container{ width:
  100%;
  height:100vh;
  display: flex;
  align-items: center;
  justify-content: center;
  background: linear-gradient(140deg,rgb(25, 25, 138),rgb(93, 93, 236));
}
.calculator{
  padding: 20px;
  border-radius: 10px;
  background-color: white;
}
```

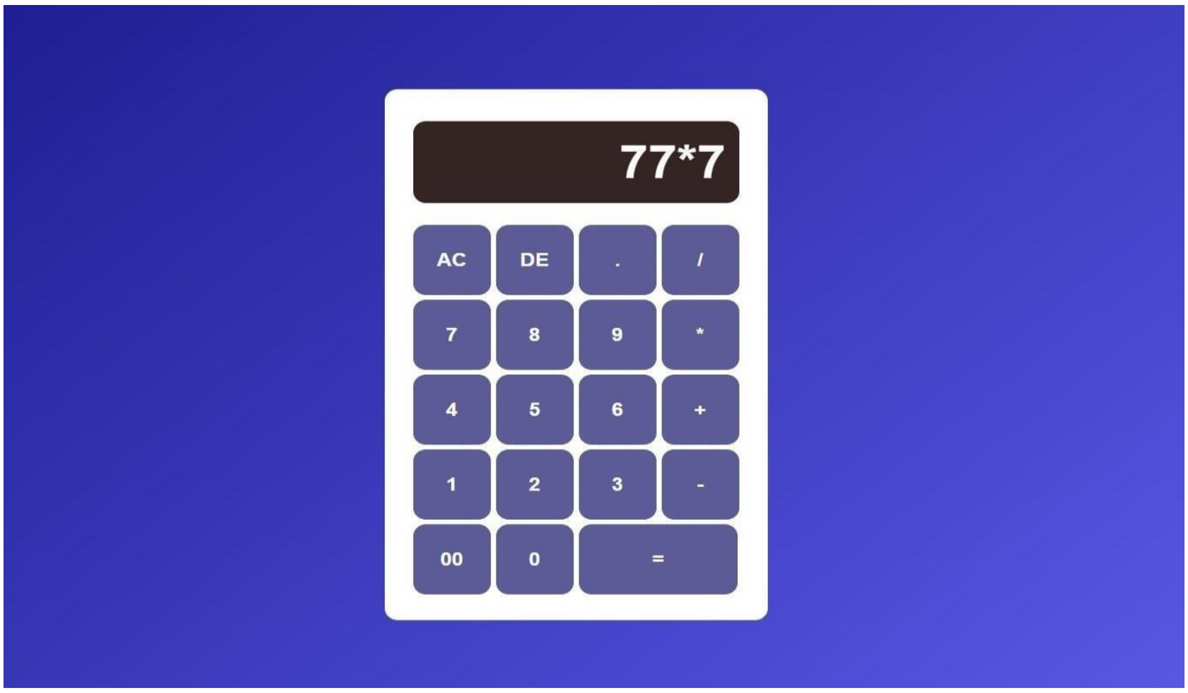
```
 form input{
    border: none;
   outline: 0;
    width: 60px;
   height:60px;
    font-size: 16px;
   background-color: rgb(91,91,151);
   margin: 2px;
   border-radius: 10px;
   color: white;
   font-weight: bold;
   cursor: pointer;
  }
 form input[type="button"]:hover{
   background-color: rgb(89, 6, 167);
  }
 form .display{
   display: flex;
   justify-content: flex-end;
   margin: 5px 0px 15px 0px;
  }
 form .display input{
   text-align: right;
   flex:1;
   font-size: 40px;
   padding: 5px 10px;
   background-color: rgb(52, 36, 36);
}
```

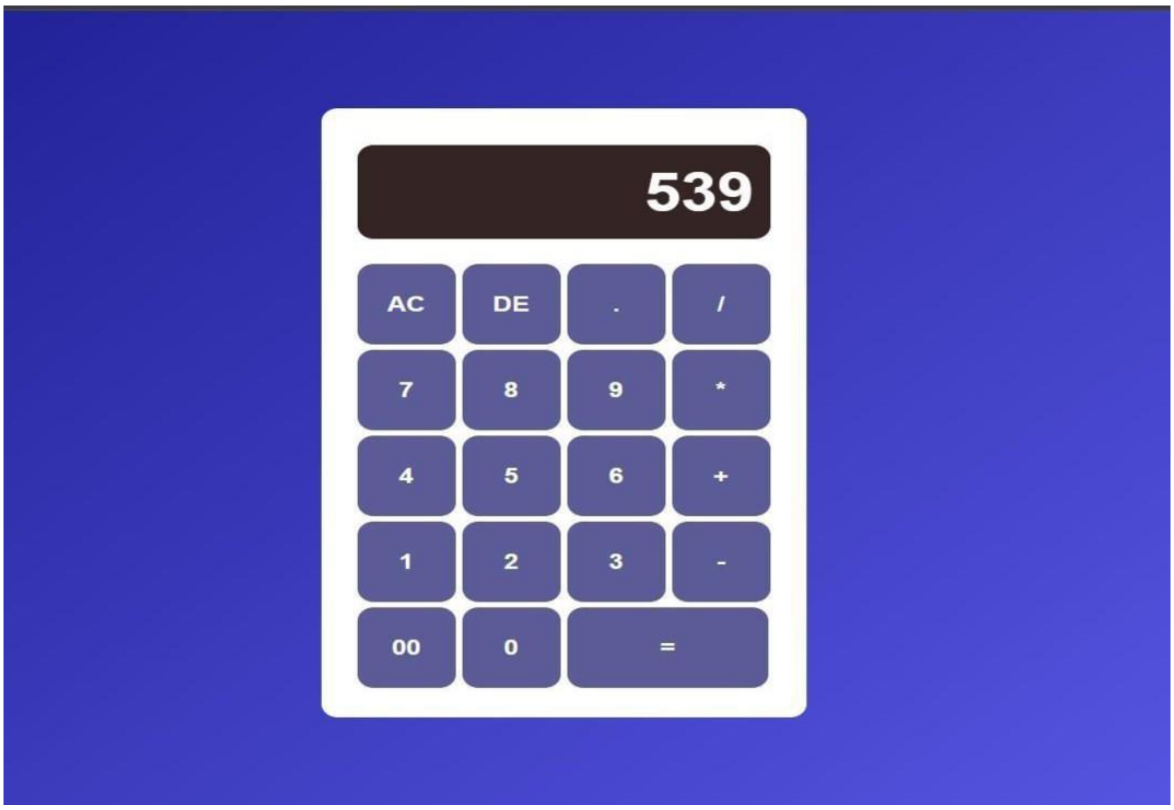
```
form input.equal{
  width: 123px;
}
```

Index.js

```
import React from 'react';
import ReactDOM from 'react-dom/client';
import App from './App';

const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(
  <React.StrictMode>
    <App />
  </React.StrictMode>
);
```

Output:


77*7
AC
DE
.
/
7
8
9
*
4
5
6
+
1
2
3
-
00
0
=


539
AC
DE
.
/
7
8
9
*
4
5
6
+
1
2
3
-
00
0
=