Experiment No: 9

Program:

StateExp.js

```
import React, {Component} from "react";

class StateExp extends Component{
state={
   name: "State",
   clg: "XIE",
   city: "Mahim"
}
render(){
   return(
      <>
      <h2>This is Example of <span>{this.state.name} </span> in React</h2>
      <h3>Location: {this.state.clg} , {this.state.city}</h3>
      </>
   )
}
}
export default StateExp;
```

PropsExp.js

```
import React, {Component} from "react";
class PropsExp extends Component{
state={
   name: this.props.name,
   clg: "XIE",
   city: "Mahim"
}
render(){
   return(
      <>
      <h2>This is Example of <span>{this.state.name} </span> in React</h2>
      <h3>Location: {this.state.clg} , {this.state.city}</h3>
      </>
   )
}
}
export default PropsExp;
```

App.js

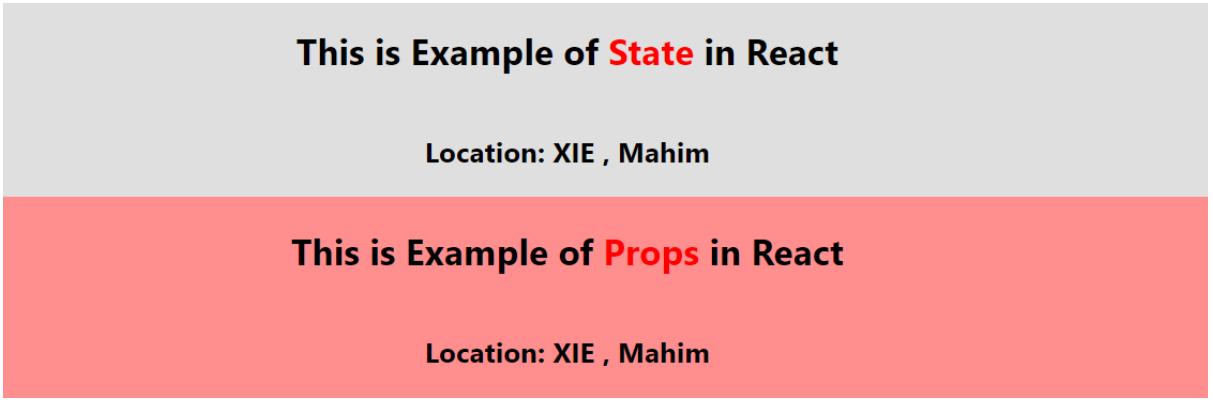
```
import PropsExp from "./components/PropsExp";
import StateExp from "./components/StateExp";
```

```
function App() {

  return (
    <div className="App">
      <div className="pt1">
        <StateExp/>
      </div>
      <div className="pt2">
        <PropsExp name="Props" />
      </div>

    </div>
  );
}

export default App;
```

App.css

```
.pt1{
  background-color: #dfdfdf;
  display: flex;
  flex-direction: column;
  align-items: center;
  justify-content: center;
}
.pt2{
  background-color: #ff8e8e;
  display: flex;
  flex-direction: column;
  align-items: center;
  justify-content: center;
}
```

Output:



Experiment 12

Opening REPL Terminal:

```
C:\Users\Mrunal>node
Welcome to Node.js v16.17.0.
Type ".help" for more information.
```

Arithmetic Operators:

```
> 10 + 15
25
> var a = 24
undefined
> var b = 17
undefined
> a - b
7
> a * b
408
> a / b
1.411764705882353
> a % b
7
```

Logical Operators:

```
> var x = true
undefined
> var y = false
undefined
> x && y
false
> x || y
true
> !x
false
```

Concatenation of two strings:

```
> "hello " + "Experiment " + 12
'hello Experiment 12'
```

Using the Variable to obtain the last value:

```
> a + b
41
> var sum = _
undefined
> console.log(sum)
41
```

To print the table of a given number:

```
> .editor
// Entering editor mode (Ctrl+D to finish, Ctrl+C to cancel)
var a = 7
for(i = 1; i <= 10; i++){
 console.log(a," x ",i," = ", a*i)
 }
7  x  1  =  7
7  x  2  =  14
7  x  3  =  21
7  x  4  =  28
7  x  5  =  35
7  x  6  =  42
7  x  7  =  49
7  x  8  =  56
7  x  9  =  63
7  x  10  =  70
```

Even Series:

```
> .editor
// Entering editor mode (Ctrl+D to finish, Ctrl+C to cancel)
for(i = 0; i <=25; i++){
 if(i%2 == 0){
  console.log(i)}
  }
0
2
4
6
8
10
12
14
16
18
20
22
24
```

Odd Series:

```
> .editor
// Entering editor mode (Ctrl+D to finish, Ctrl+C to cancel)
for(i = 0; i <= 25 ; i++){
 if(i % 2 != 0 ){
  console.log(i)}
  }
1
3
5
7
9
11
13
15
17
19
21
23
25
```

Fibonacci Series:

```
> .editor
// Entering editor mode (Ctrl+D to finish, Ctrl+C to cancel)
var t1 = 0
var t2 = 1
console.log(t1)
console.log(t2)
for(i = 2; i < 10; i ++){
 let t3 = t1 + t2
 t1 = t2
 t2 = t3
 console.log(t3)}

0
1
1
2
3
5
8
13
21
34
```

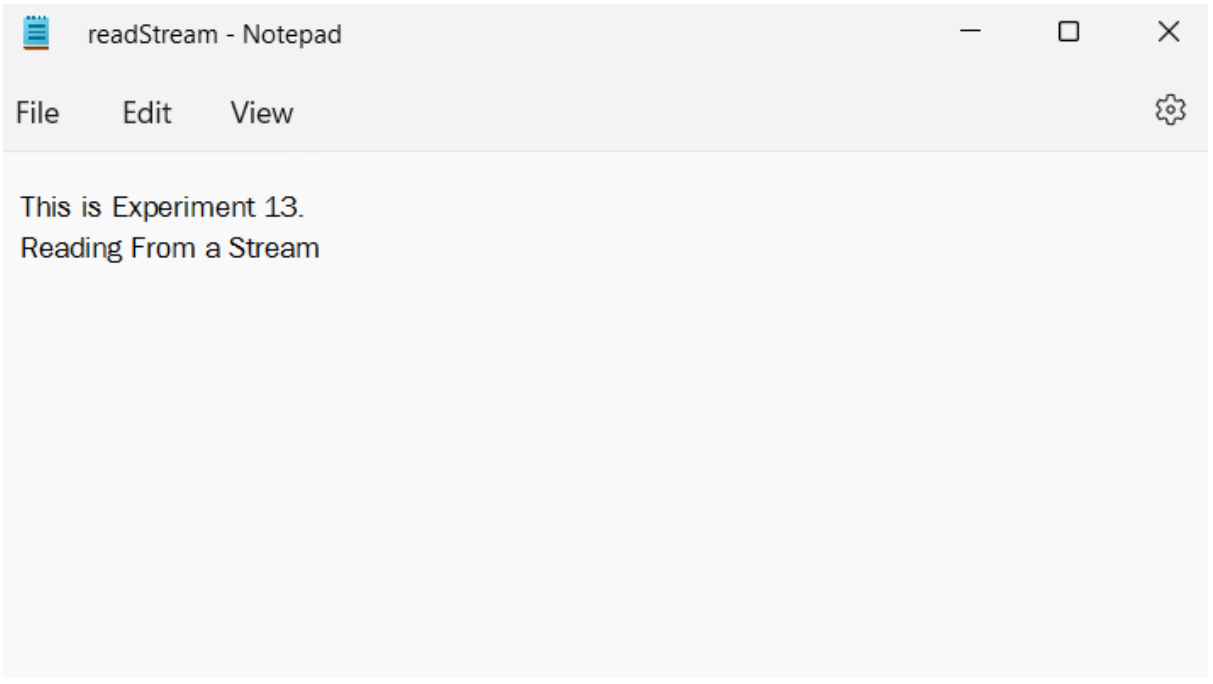
Experiment 13

Reading From Stream.

Program:

```
var fs = require("fs");
var data = “”;
var readerStream = fs.createReadStream('input.txt');
readerStream.setEncoding('UTF8');
readerStream.on('data', function(chunk) {
   data += chunk;
});
readerStream.on('end',function() {
   console.log(data);
});
readerStream.on('error', function(err) {
   console.log(err.stack);
});
console.log("Program Ended");
```

Output:




```
C:\Users\Mrunal\OneDrive\Desktop\Self>node -v
v16.17.0

C:\Users\Mrunal\OneDrive\Desktop\Self>node main.js
Program Ended
This is Experiment 13.
Reading From a Stream
```

Writing to a Stream:

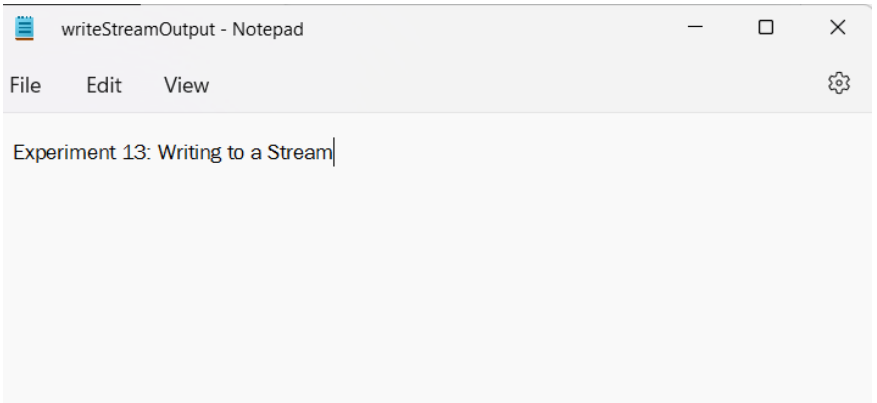
Program:

```
var fs = require("fs");
var data = 'Experiment 13: Writing to a Stream';

var writerStream = fs.createWriteStream('writeStreamOutput.txt');

writerStream.write(data,'UTF8');

writerStream.end();

writerStream.on('finish', function() {
   console.log("Write completed.");
});

writerStream.on('error', function(err) {
   console.log(err.stack);
});

console.log("Program Ended");
```

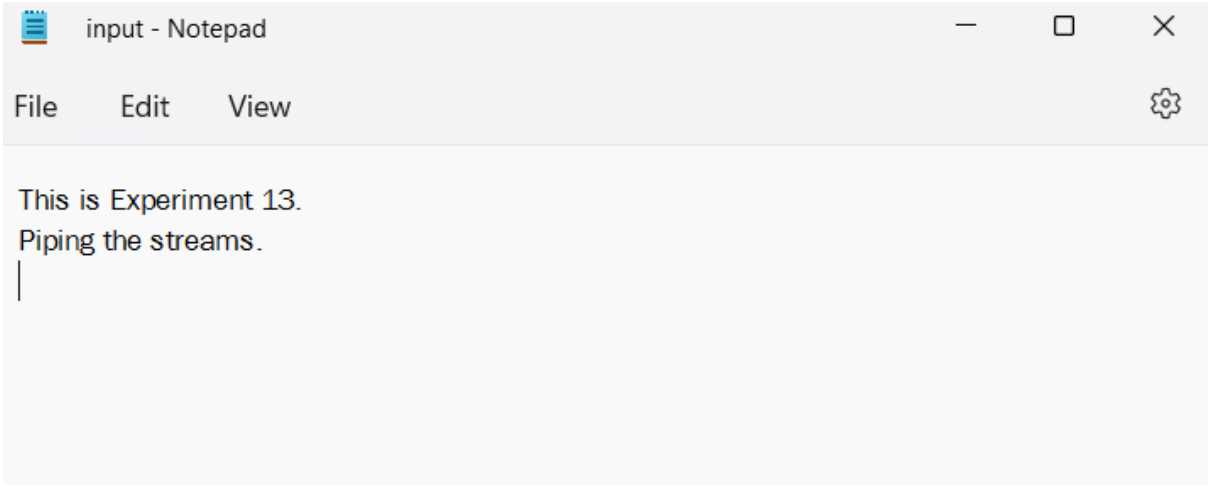
Output:

```
C:\Users\Mrunal\OneDrive\Desktop\Self>node main.js
Program Ended
Write completed.
```

fileOutput:



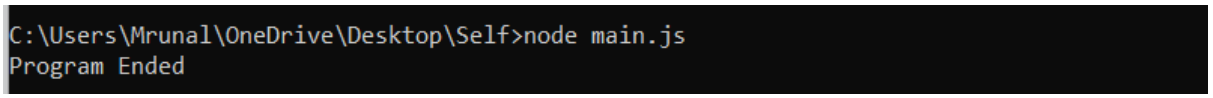
Piping the Streams:

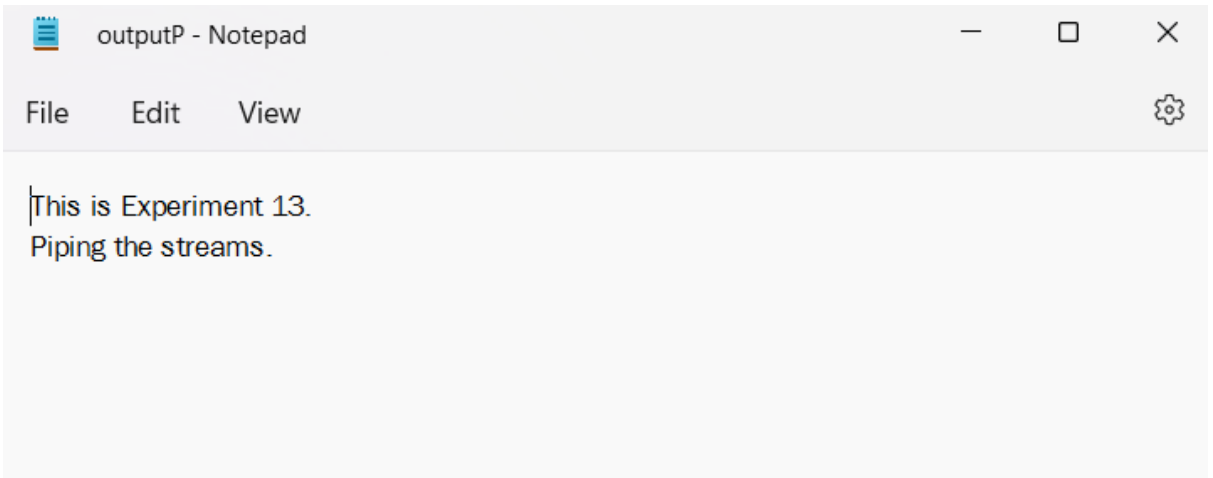


Program:

```
var fs = require("fs");

var readerStream = fs.createReadStream('input.txt');

var writerStream = fs.createWriteStream('outputP.txt');


readerStream.pipe(writerStream);

console.log("Program Ended");
```

Output:



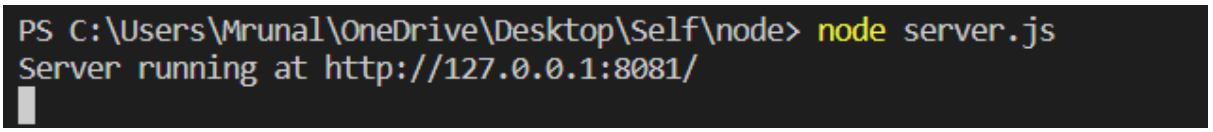


Experiment 14:

1.Creating a Web Server using Node
Server.js
Program:

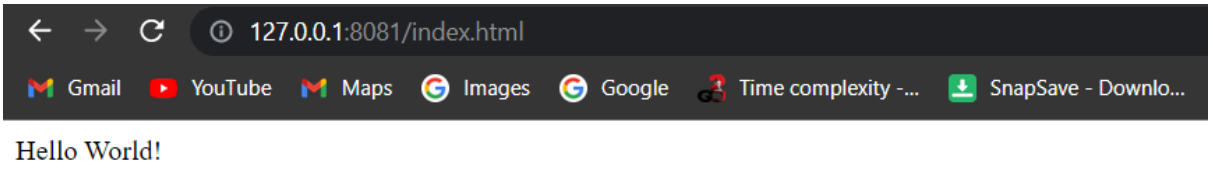
```
var http = require('http');
var fs = require('fs');
var url = require('url');

http.createServer( function (request, response) {

   var pathname = url.parse(request.url).pathname;

   console.log("Request for " + pathname + " received.");

   fs.readFile(pathname.substr(1), function (err, data) {
      if (err) {
         console.log(err);

         response.writeHead(404, {'Content-Type': 'text/html'});
      } else {

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

         response.write(data.toString());
      }

      response.end();
   });
}).listen(8081);

console.log('Server running at http://127.0.0.1:8081/');
```

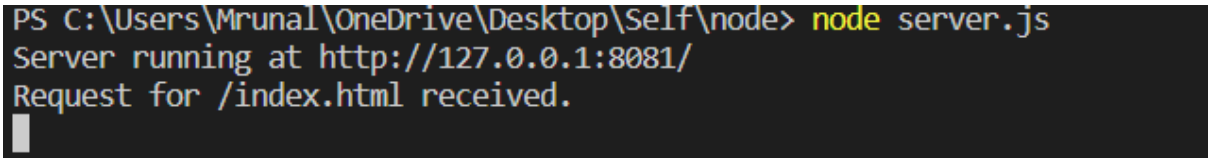
Output:



2.Make a request to Node.js server:

Output:





Index.html
Program:

```
<html>
  <head>
    <title>HTML Page</title>
  </head>

  <body>
    Hello World!
  </body>
</html>
```

3.Creating Web client using Node:

Client.js

Program:

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

var options = {
  host: 'localhost',
  port: '8081',
  path: '/index.html'
};

var callback = function(response) {

  var body = '';
  response.on('data', function(data) {
    body += data;
```

```
  });

  response.on('end', function() {
    console.log(body);
  });
}

var req = http.request(options, callback);
req.end();
```

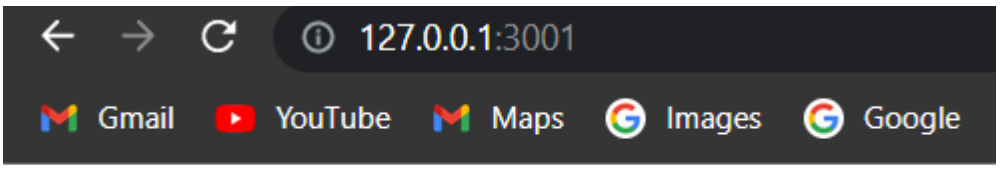
Output:

```
PS C:\Users\Mrunal\OneDrive\Desktop\Self\node> node client.js
<html>
   <head>
      <title>HTML Page</title>
   </head>

   <body>
      Hello World!
   </body>
</html>
```

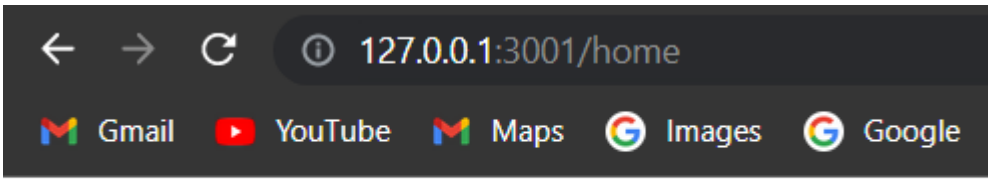
```
PS C:\Users\Mrunal\OneDrive\Desktop\Self\node> node server.js
Server running at http://127.0.0.1:8081/
Request for /index.html received.
Request for /index.html received.
```

Experiment No: 15

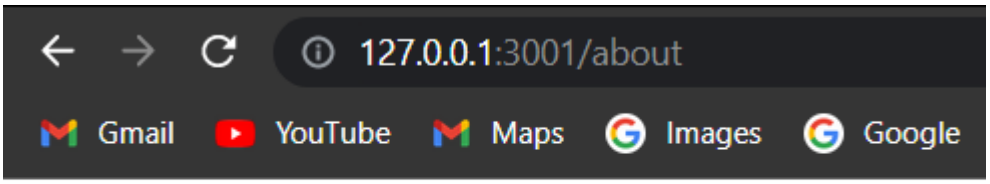
Output:



Landing Page



Home Page



About Page

```
PS C:\Users\Mrunal\OneDrive\Desktop\Self\node> node index.js
Example app listening on port 3001!
A new request received at 1666945240170 at HomePage
A new request received at 1666945244929 at AboutPage
```

Experiment No:10

Program:
App.js

```
import { useState, useEffect } from "react";
import './App.css';

function App() {
  const initialValues = { username: "", email: "", password: "" };
  const [formValues, setFormValues] = useState(initialValues);
  const [formErrors, setFormErrors] = useState({});
  const [isSubmit, setIsSubmit] = useState(false);

  const handleChange = (e) => {
    const { name, value } = e.target;
    setFormValues({ ...formValues, [name]: value });
  };

  const handleSubmit = (e) => {
    e.preventDefault();
    setFormErrors(validate(formValues));
    setIsSubmit(true);
  };

  useEffect(() => {
    console.log(formErrors);
    if (Object.keys(formErrors).length === 0 && isSubmit) {
      console.log(formValues);
    }
  }, [formErrors]);
  const validate = (values) => {
    const errors = {};
    const regex = /^[^\s@]+@[^\s@]+\.[^\s@]{2,}$/i;
    if (!values.username) {
      errors.username = "Username is required!";
    }
    if (!values.email) {
      errors.email = "Email is required!";
    } else if (!regex.test(values.email)) {
      errors.email = "This is not a valid email format!";
    }
    if (!values.password) {
      errors.password = "Password is required";
    } else if (values.password.length < 4) {
      errors.password = "Password must be more than 4 characters";
    } else if (values.password.length > 10) {
      errors.password = "Password cannot exceed more than 10 characters";
    }
    return errors;
```

```
};
return (
  <div className="App">
    <div className="container">
      {Object.keys(formErrors).length === 0 && isSubmit ? (
        <div className="msgSuc">Signed in successfully</div>
      ) : (
        ""
      )}

      <form onSubmit={handleSubmit}>
        <h1>Login Form</h1>
        <div className="ui-divider"></div>
        <div className="ui-form">
          <div className="field">
            <label>Username</label>
            <input
              type="text"
              name="username"
              placeholder="Username"
              value={formValues.username}
              onChange={handleChange}

            />
          </div>
          <p>{formErrors.username}</p>
          <div className="field">
            <label>Email</label>
            <input
              type="email"
              name="email"
              placeholder="Email"
              value={formValues.email}
              onChange={handleChange}
            />
          </div>
          <p>{formErrors.email}</p>
          <div className="field">
            <label>Password</label>
            <input
              type="password"
              name="password"
              placeholder="Password"
              value={formValues.password}
              onChange={handleChange}
            />
          </div>
          <p>{formErrors.password}</p>
          <button className="button">Submit</button>
          <button className="button2">Clear</button>
        </div>
      </form>
```

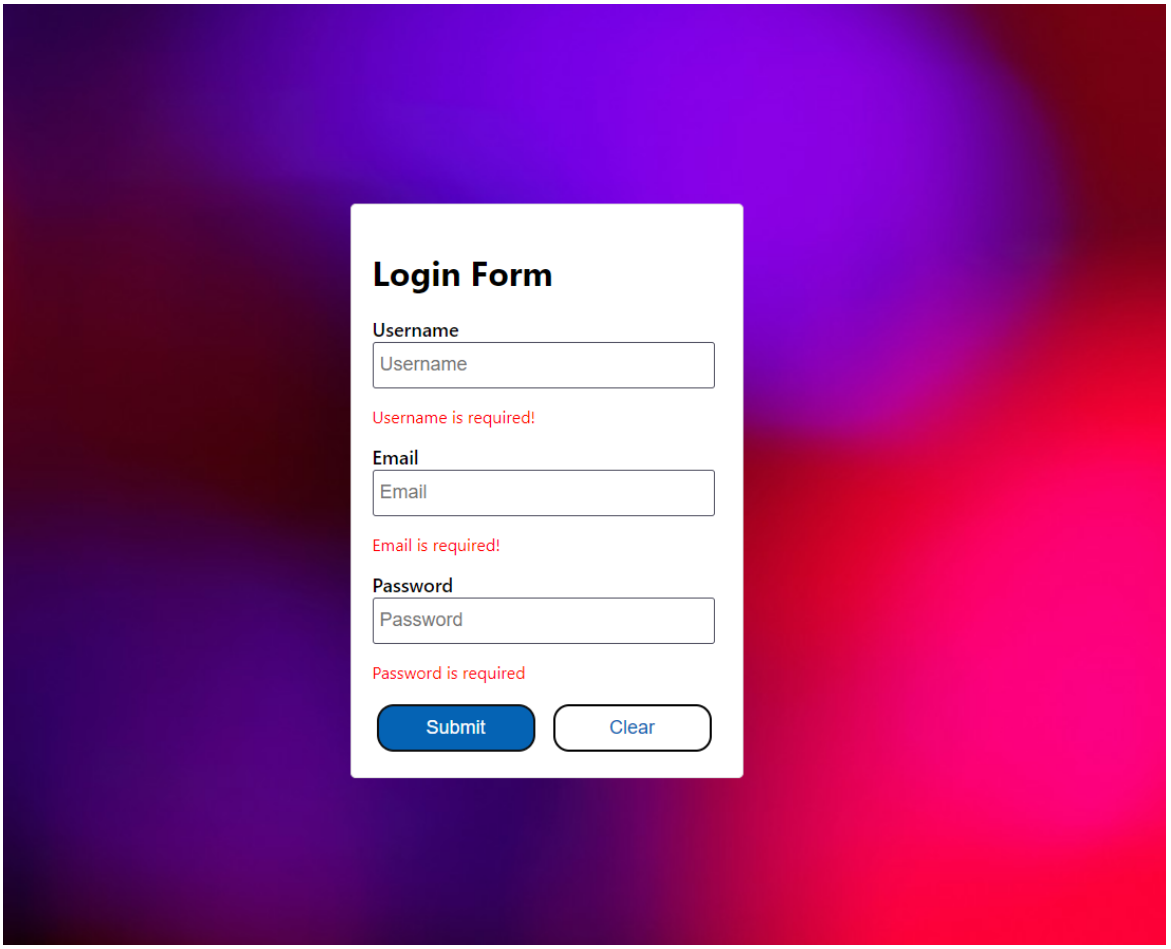
```
      </div>

    </div>
  );
}

export default App;
```

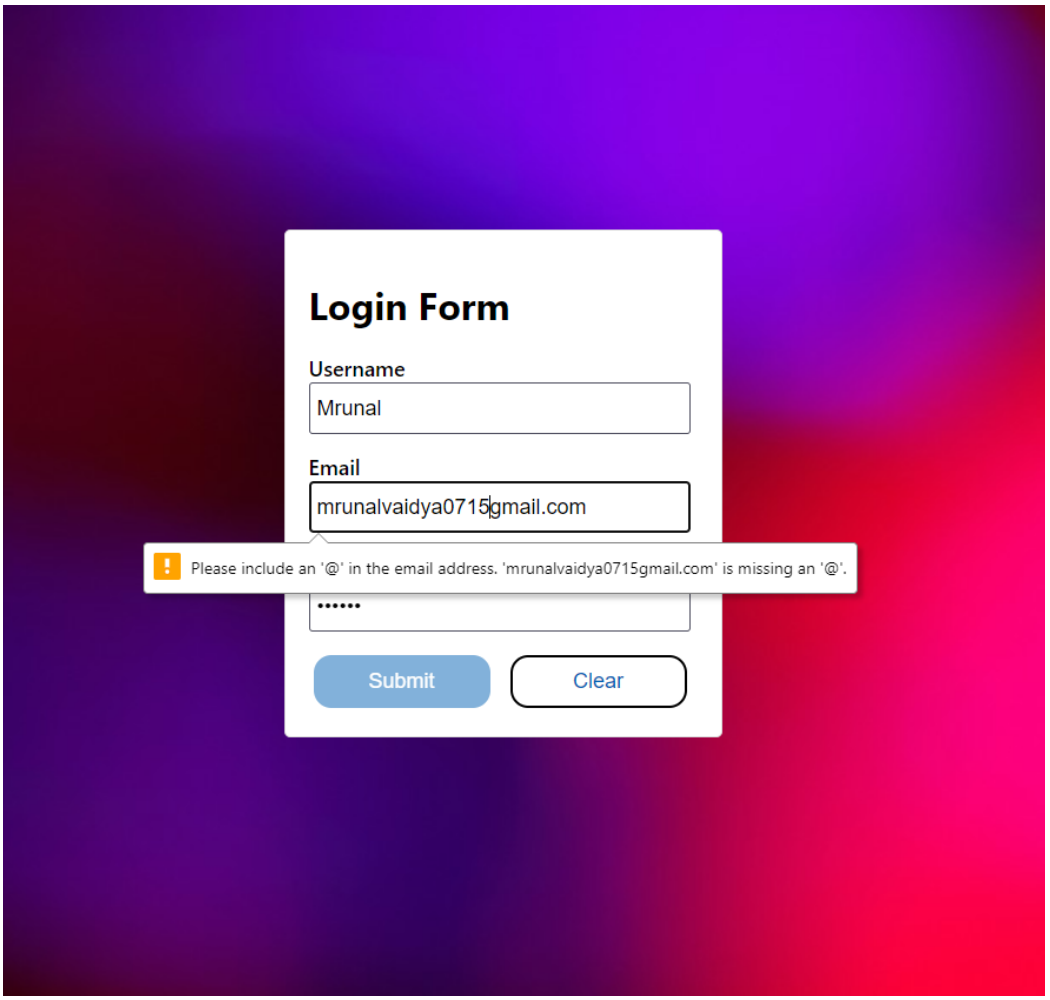
App.css

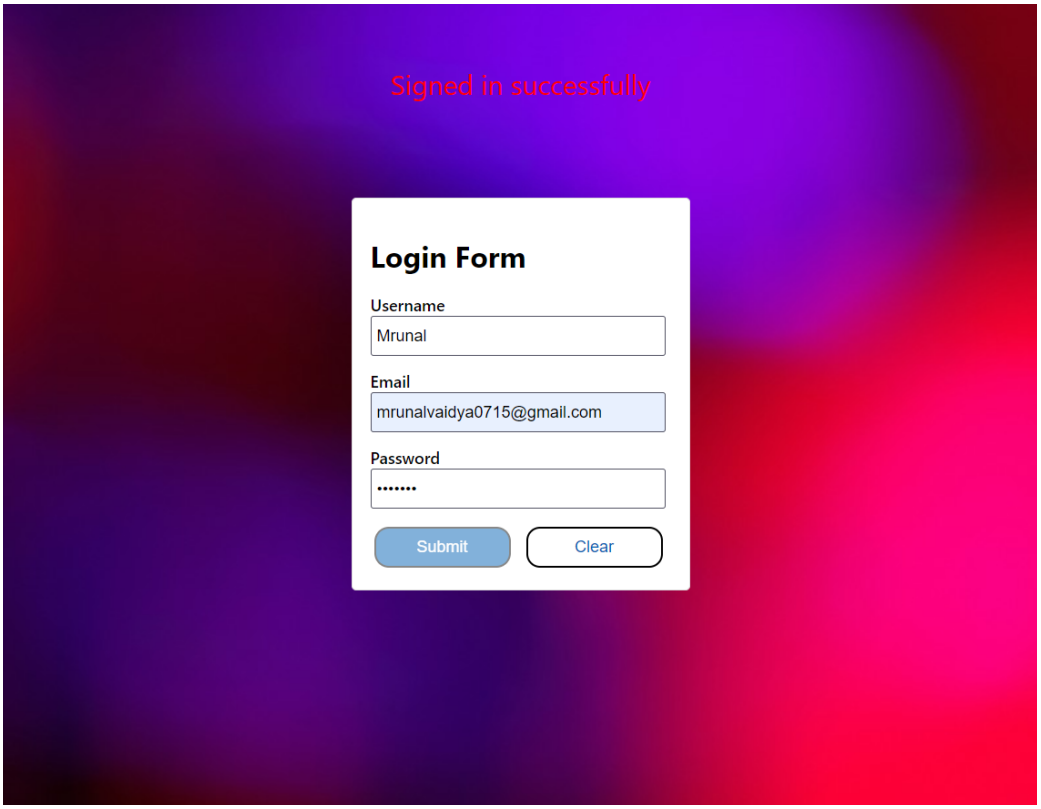
```
body {

  background-repeat: no-repeat;
  background-size: auto;
  background-size: 1980px 1080px;
  background-image:
url("https://www.hdwallpapers.in/download/blur_abstract_lights_4k_hd_abstract-1366x768.jpg");}
.container {
  padding: 15px;
  display: flex;
  flex-direction: column;
  justify-content: center;
  align-items: center;
  height: 100vh;
  max-width: 550px;
  width: 100%;
  margin: auto;}
.container > form {
  width: 60%;
  border: 1px solid #dfdfdf;
  padding: 20px;
  border-radius: 5px;
  background: #fff;}
pre {
  width: 70%;
  color: #fff;
  font-size: larger;}
.button2 {
  background: #ffffff;
  margin-left: 13px;
  border: 2px solid black;
  padding: 5px;
  width: 150px;
  border-radius: 15px;
  align-items: center;
  color: #0563b4;
  font-size: 18px;}
button {
  margin: 4px;
  background: #0563b4;
  border: 2px solid #0563b4;;
  padding: 5px;
  width: 150px;
```

```
  border-radius: 15px;
  color: white;
  font-size: 18px;
  align-items: center;
  height: 45px;}

button:hover{
  opacity: 0.5;}
button:active {
  opacity: 0.2;}
p {
  color: red;}
.field{
  font-size: large;
  font-weight: 500;
  margin-right: 250px;}
.field > input{
  width: 310px;
  height: 30px;
  font-size: 18px;
  border-radius: 3px;
  border: 1px solid rgb(82, 82, 99);
  padding: 6px;}
.msgSuc{
  color: red;
  font-size: 30px;
  position: absolute;
  top: 100px;}
```

Output:


Login Form
Username
Username
Username is required!
Email
Email
Email is required!
Password
Password
Password is required
Submit
Clear


Login Form
Username
Mrunal
Email
mrunalvaidya0715gmail.com
Please include an '@' in the email address. 'mrunalvaidya0715gmail.com' is missing an '@'.
Submit
Clear


Signed in successfully
Login Form
Username
Mrunal
Email
mrunalvaidya0715@gmail.com
Password
Submit
Clear

Experiment No:4

Program:
Exp4.html

```
<!DOCTYPE html>
<head>
   <title>Exp4</title>
   <link rel="stylesheet" href="Exp4.css">
   <script defer src="Exp4.js"></script>
</head>
<body>
   <div class="container">
      <form id="form" action="/">
         <h1>Registration Form</h1>
         <div class="input-control">
            <label for="username">Username<span class="sp">*</span></label>


            <input id="username" name="username" type="text">
            <div class="error"></div>
         </div>
         <div class="input-control">
            <label for="email">Email<span class="sp">*</span></label>
            <input id="email" name="email" type="text">
            <div class="error"></div>
         </div>
         <div class="input-control">
            <label for="password">Password<span class="sp">*</span></label>
            <input id="password"name="password" type="password">
            <div class="error"></div>
         </div>
         <div class="input-control">
            <label for="password2">Password again<span class="sp">*</span></label>
            <input id="password2"name="password2" type="password">
            <div class="error"></div>
         </div>
         <button type="submit">Sign Up</button>
      </form>
   </div>
</body>
</html>
```

Exp4.css

```
body {
   background-repeat: no-repeat;
  background-size: auto;
  background-size: 1980px 1080px;
```

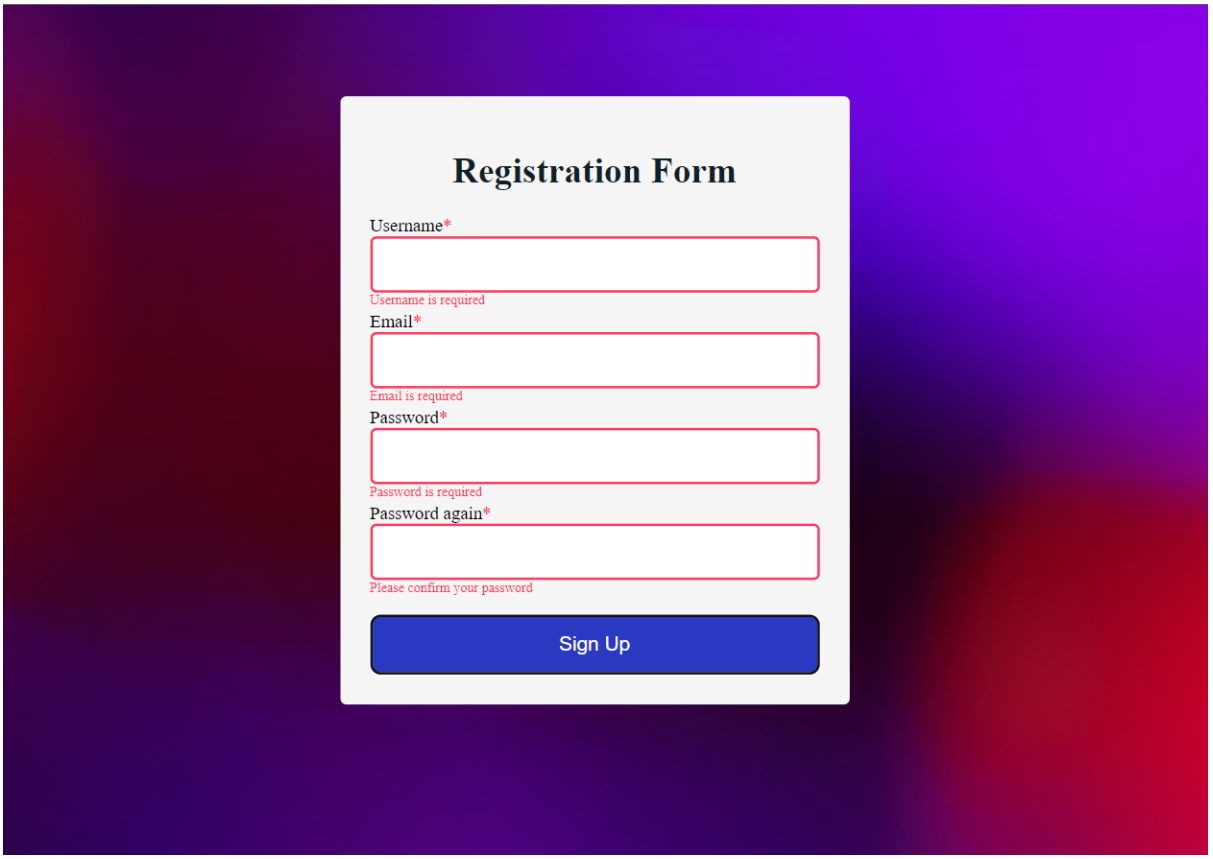
```
  background-image:
url("https://www.hdwallpapers.in/download/blur_abstract_lights_4k_hd_abstract-1366x768.jpg");}
#form {
    width: 300px;
    margin: 20vh auto 0 auto;
    padding: 20px;
    background-color: whitesmoke;
    border-radius: 4px;
    font-size: 12px;}
.sp{
    color: red;}
#form h1 {
    color: #0f2027;
    text-align: center;}
#form button {
    padding: 10px;
    margin-top: 10px;
    width: 100%;
    color: white;
    background-color: rgb(41, 57, 194);
    border: 2px solid;
    border-radius: 6px;
    transition: background-color ease-in 100ms;}
#form button:hover {
    padding: 10px;
    margin-top: 10px;
    width: 100%;
    color: rgb(41, 57, 194);
    background-color: white;
    border: 2px solid;
    border-radius: 6px;
    border-color:rgb(41, 57, 194) ;}
.input-control {
    display: flex;
    flex-direction: column;}
.input-control input {
    border: 2px solid #f0f0f0;
            border-radius: 4px;
            display: block;
            font-size: 12px;
            padding: 10px;
            width: 100%;}
.input-control input:focus {
    outline: 0;}
.input-control.success input {
    border-color: #09c372;}
.input-control.error input {
    border-color: #ff3860;}
.input-control .error {
    color: #ff3860;
    font-size: 9px;
    height: 13px;}
```

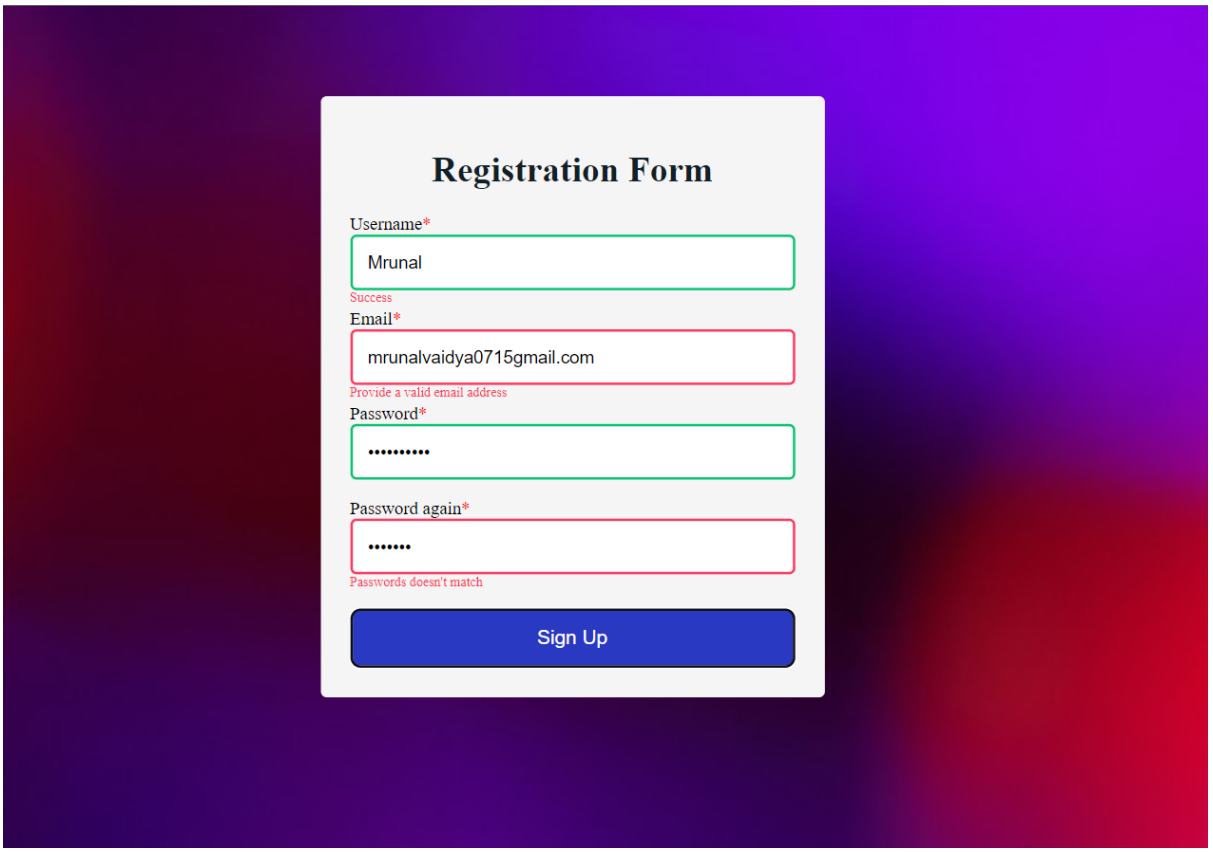
Exp4.js

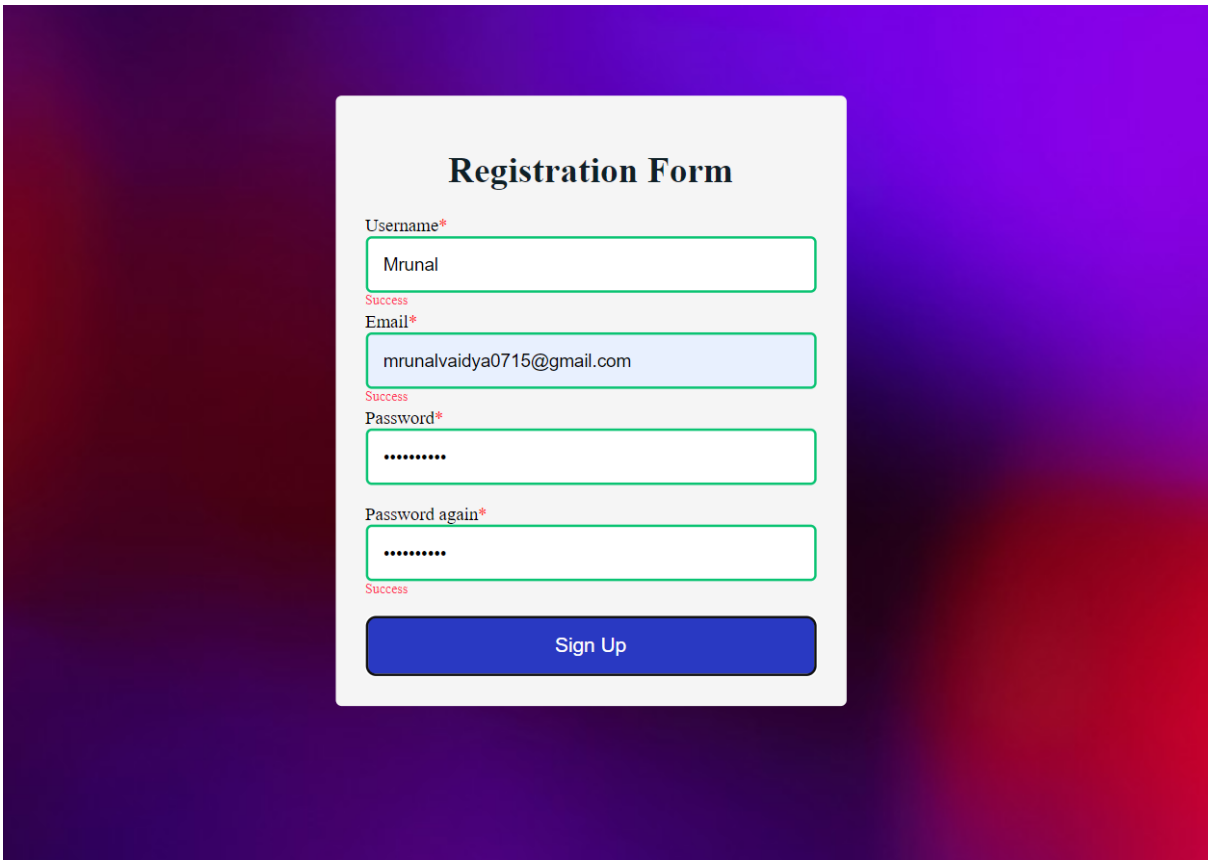
```
const form = document.getElementById('form');
const username = document.getElementById('username');
const email = document.getElementById('email');
const password = document.getElementById('password');
const password2 = document.getElementById('password2');
form.addEventListener('submit', e => {
    e.preventDefault();
    validateInputs();
});
const setError = (element, message) => {
    const inputControl = element.parentElement;
    const errorDisplay = inputControl.querySelector('.error');
    errorDisplay.innerText = message;
    inputControl.classList.add('error');
    inputControl.classList.remove('success')
}
const setSuccess = (element,success_msg) => {
    const inputControl = element.parentElement;
    const errorDisplay = inputControl.querySelector('.error');
    errorDisplay.innerText = success_msg;
    inputControl.classList.add('success');
    inputControl.classList.remove('error');
};
const isValidEmail = email => {
    const re =
/^(([^<>()[\]\\.,;:\s@"]+(\.[^<>()[\]\\.,;:\s@"]+)*)|(".+"))@((\[[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\])|(([
a-zA-Z\-0-9]+\.)+[a-zA-Z]{2,}))$/;
    return re.test(String(email).toLowerCase());
}
const validateInputs = () => {
    const usernameValue = username.value.trim();
    const emailValue = email.value.trim();
    const passwordValue = password.value.trim();
    const password2Value = password2.value.trim();
    if(usernameValue === '') {
        setError(username, 'Username is required');
    }
    else if(usernameValue.length < 5){
        setError(username, 'Username must be at least 5 characters');

    } else {
        setSuccess(username,"Success");
    }
    if(emailValue === '') {
        setError(email, 'Email is required');
    } else if (!isValidEmail(emailValue)) {
        setError(email, 'Provide a valid email address');
    } else {
        setSuccess(email,"Success");
```

```
    }
    if(passwordValue === '') {
        setError(password, 'Password is required');
    } else if (passwordValue.length < 8 ) {
        setError(password, 'Password must be at least 8 character.')
    } else {
        setSuccess(password,"");
    }

    if(password2Value === '') {
        setError(password2, 'Please confirm your password');
    } else if (password2Value !== passwordValue) {
        setError(password2, "Passwords doesn't match");
    } else {
        setSuccess(password2,"Success");
    }
};
```

Output:




Registration Form
Username*
Mrunal
Success
Email*
mrunalvaidya0715gmail.com
Provide a valid email address
Password*
Password again*
Passwords doesn't match
Sign Up


Registration Form
Username*
Mrunal
Success
Email*
mrunalvaidya0715@gmail.com
Success
Password*
Password again*
Success
Sign Up