



ElectroGate

SOFTWARE TESTING REPORT

Title: Electronics Home Service Solutions

Github link: <https://github.com/ankitiiitian/ElectroGate-Home-Service-Solutions->

Overview:

The purpose of this web application is to connect client and service care through a virtual platform without being present physically.

Our motive is to provide home service solutions to electronics problem 24*7 when clients can't reach out physically, especially in current pandemic scenarios.

This platform is to make our lives more fulfilling to solve our electronics problem. It enables users to find any service professional related to electronics problem.

Testing:

Unit test:

1. Registration as Professional:

In firebase we should test the labels for all the fields, because even though we haven't explicitly specified most of them, we have a design that says what these values should be. If we don't test the values, then we don't know that the field labels have their intended values.

Snippet: Two codes there, one for first.test.js which have sample input and second for testCollectionTriggers.js



ElectroGate

The screenshot shows two instances of Visual Studio Code side-by-side. Both instances have the same project structure:

- EXPLORER**: Shows files like `index.js`, `setupTests.js`, `package.json`, `first.test.js`, `testCollectionTriggers.js`, `.eslintrc.js`, `.gitignore`, `index.js`, `jest.config.js`, `jsonconfig.json`, `package-lock.json`, `package.json`, `server.js`, and `src` (containing `functions`, `node_modules`, and `_tests_`).
- PROBLEMS**: Shows no errors.
- OUTPUT**: Shows no output.
- DEBUG CONSOLE**: Shows no output.
- TERMINAL**: Shows the command `node`.

Code Content (first.test.js):

```
client > functions > src > _tests_ > first.test.js > test("Expect to find a copy in 'Copies' Collection") callback
1 const firebase = require('firebase/testing')
2 const admin = require('firebase-admin');
3 const { ref } = require('firebase-functions/v1/database');
4
5 const projectId = "electrosolve-521ce"
6 process.env.GCLOUD_PROJECT = projectId
7 process.env.FIRESTORE_EMULATOR_HOST = 'localhost:8080';
8 let app = admin.initializeApp({projectId})
9 let db = firebase.firestore(app)
10
11 beforeAll(async ()=>
12   await firebase.clearFirestoreData([projectId]));
13 )
14
15 //When Document written to "/testcollection/{DocumentId}" , trigger function to copy it to "/Copies/{DocumentId}"
16 test("Expect to find a copy in 'Copies' Collection", async ()=>
17   const testDoc = db.collection('TestCollection').doc()
18   const doc = {
19     name: "Sameer",
20     age: 21,
21     city: "Riyadh"
22   }
23
24   const ref = db.collection('TestCollection').doc()
25   await ref.set(testDoc)
26
27   const copyId = ref.id
28
29   const copyRef = db.collection('copies').doc(copyId)
30
31   const copyDoc = await copyRef.get()
32
33   expect(copyDoc.data()).toStrictEqual(testDoc)
34 ))
```

Code Content (testCollectionTriggers.js):

```
client > functions > src > _tests_ > testCollectionTriggers.js > test("Expect to find a copy in 'Copies' Collection") onCreate
1 const functions = require('firebase-functions');
2 const admin = require('firebase-admin');
3
4 admin.initializeApp();
5 const db = admin.firestore();
6
7 exports.onCreate = functions.firestore.document('TestCollection/{docId}').onCreate(async (snapshot)=>{
8   const data = snapshot.data()
9
10  const docId = snapshot.id
11  const copyRef = db.collection('copies').doc(docId)
12
13  await copyRef.set(data)
14 })
```

Output: This one is output of firebase Cloud Firestore where we can store all data for register as professional.

The screenshot shows the Firebase Emulator Suite interface with the following details:

- EXPLORER**: Shows files like `index.js`, `first.test.js`, `testCollectionTriggers.js`, and `index.js` under `functions`, and `src` (containing `wrap-ansi`, `wrappy`, `write`, `write-file-atomic`, `ws`, `xdg-basedir`, `xml-name-validator`, `xmllchars`, `xmlhttprequest`, `xtend`, `y18n`, `yallist`, `yargs`, `yargs-parser`, and `src/_tests_`).
- FIREBASE-EMULATORS-T...**: Shows `wrap-ansi`, `wrappy`, `write`, `write-file-atomic`, `ws`, `xdg-basedir`, `xml-name-validator`, `xmllchars`, `xmlhttprequest`, `xtend`, `y18n`, `yallist`, `yargs`, and `yargs-parser`.
- New Tab**: Shows the URL `localhost:4000/firestore/TestCollection/JtH1C8d96gXdz2fRt9lQ`.
- Firebase Emulator Suite**: Shows the `TestCollection` database structure with a single document `JtH1C8d96gXdz2fRt9lQ` containing fields: `city: "Riyadh"`, `name: "Sameer"`, and `age: 21`.



Black-Box Testing:

Black Box Testing is a software testing method in which the functionalities of software applications are tested without having knowledge of internal code structure, implementation details and internal paths. Black Box Testing mainly focuses on input and output of software applications and it is entirely based on software requirements and specifications. It is also known as Behavioural Testing.

- Two test accounts were created to test the flow of entire web application.

From Users-End:

1.Sign-Up:

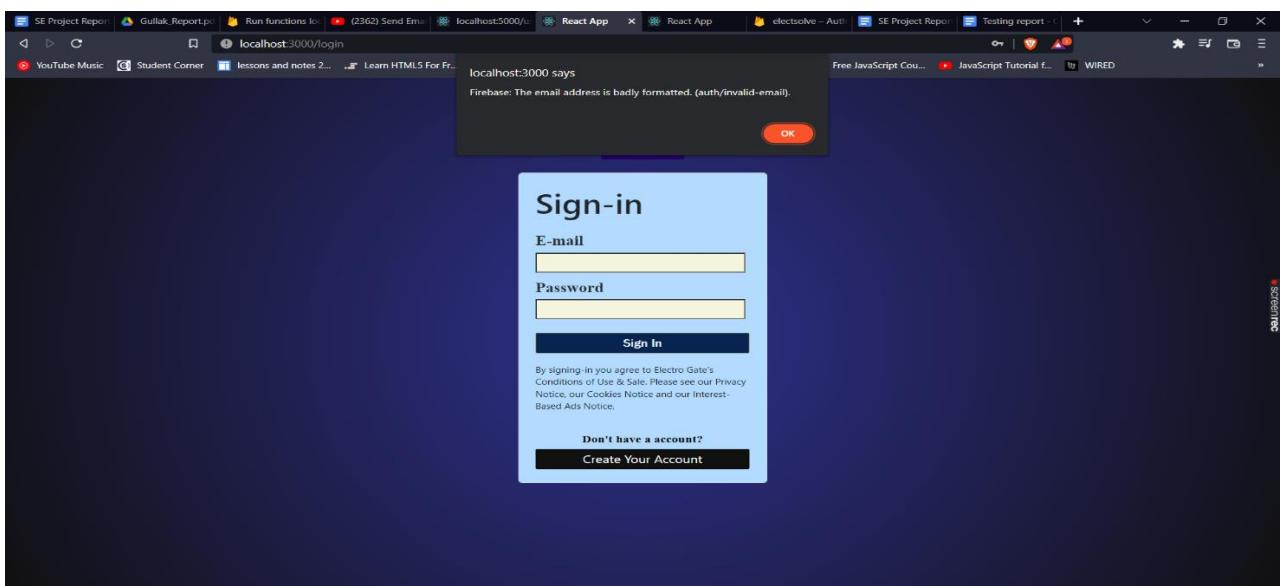
Test-Cases:

1.1 When all field left empty and press “Create new Account”.

Expectations: It will show error message.

Result: Passed.

Screen Shots/Logs:





ElectroGate

1.2 When all field filled correctly and press “Create new Account”.

Expectations: It will open Home-page and show your email in place of Hello Guest.

Result: Passed.

Screen Shots/Logs:

The screenshot shows two browser windows. The top window displays the ElectroGate website with a dark blue header featuring the logo and navigation links. The bottom window shows the Firebase Authentication console under the project 'electslove'. The 'Users' tab is selected, showing a table of registered users with columns for Identifier, Providers, Created, Signed In, and User UID. The table lists ten users, all of whom are signed in at the time the screenshot was taken.

Identifier	Providers	Created	Signed In	User UID
amit.k@iitg.ac.in	Email	Apr 15, 2022	Apr 15, 2022	z0Q8fa9PPVeF4yNgfI2IIJAQ7i1
worldtraveler345@gmail.c...	Email	Apr 8, 2022	Apr 8, 2022	NeWuVGRrU0WZFgo0ryQaaTmIW...
june@gmail.com	Email	Mar 31, 2022	Mar 31, 2022	WImLh6Q8WTeVgvJ1spH107FVjh1
usha@gmail.com	Email	Mar 31, 2022	Mar 31, 2022	ehfamhjLe6UT81uWEPkul2DmA...
ask@gmail.com	Email	Mar 30, 2022	Mar 31, 2022	I17GSghR2Rbcf4IXi8qHVQehGfp2
majorankits136@gmail.com	Email	Mar 28, 2022	Mar 28, 2022	e1NwEUR2voNlsqoYlqane8HOY0p1
anima1@gmail.com	Email	Mar 22, 2022	Mar 25, 2022	LC8e9agmbvUiazICPRzaQGqhnyd2
anima@gmail.com	Email	Mar 22, 2022	Mar 22, 2022	4bUM3fPzuZWgPhHSDbELh8mw4...
an@gmail.com	Email	Mar 22, 2022	Mar 22, 2022	PmGupeAxcEdnsNwlb4qYlgMAktx1
anu@gmail.com	Email	Mar 22, 2022	Mar 22, 2022	PRAPUxG5hdTBKA9f5TzNICZV0...

1.3 When already register user using sign up.

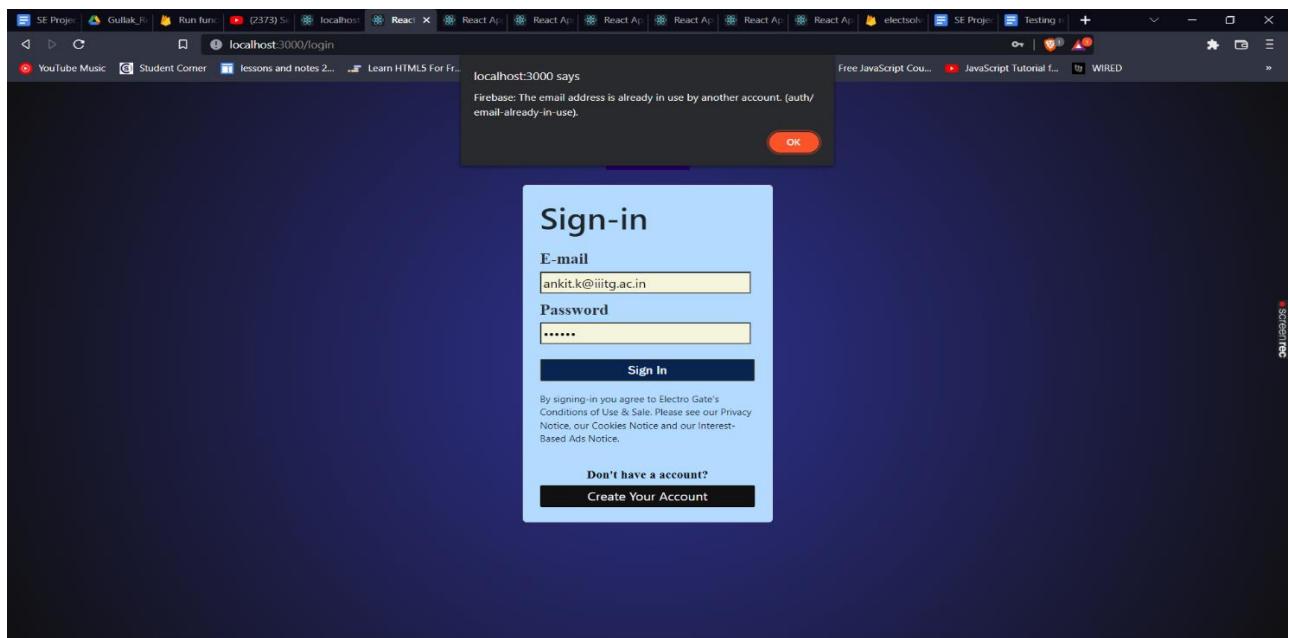
Expectations: It will show error message.

Result: Passed.

Screen Shots/Logs:



ElectroGate



2.Sign-In:

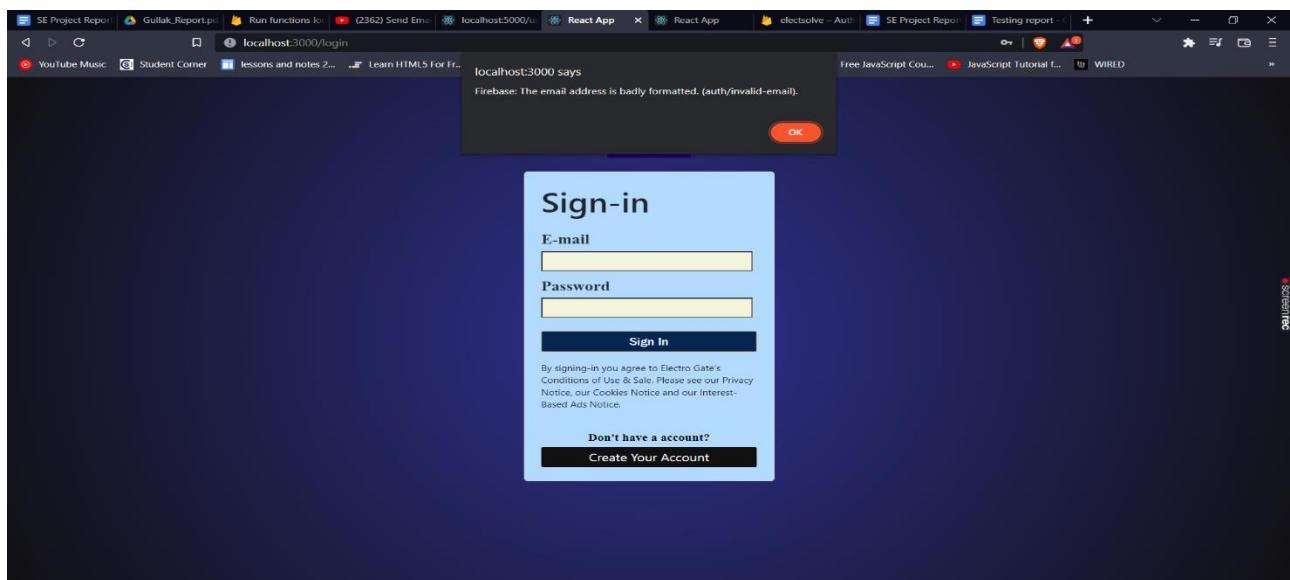
Test-Cases:

2.1 When all field left empty and press “Sign-in”.

Expectations: It will show error message.

Result: Passed.

Screen Shots/Logs:





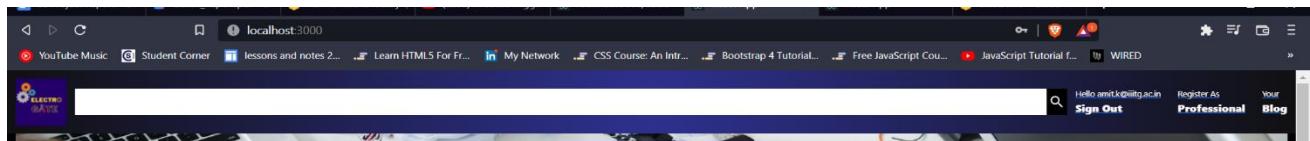
ElectroGate

2.2 When all field filled correctly and press “Sign-in”.

Expectations: It will open Home-page and show your email in place of Hello Guest.

Result: Passed.

Screen Shots/Logs:

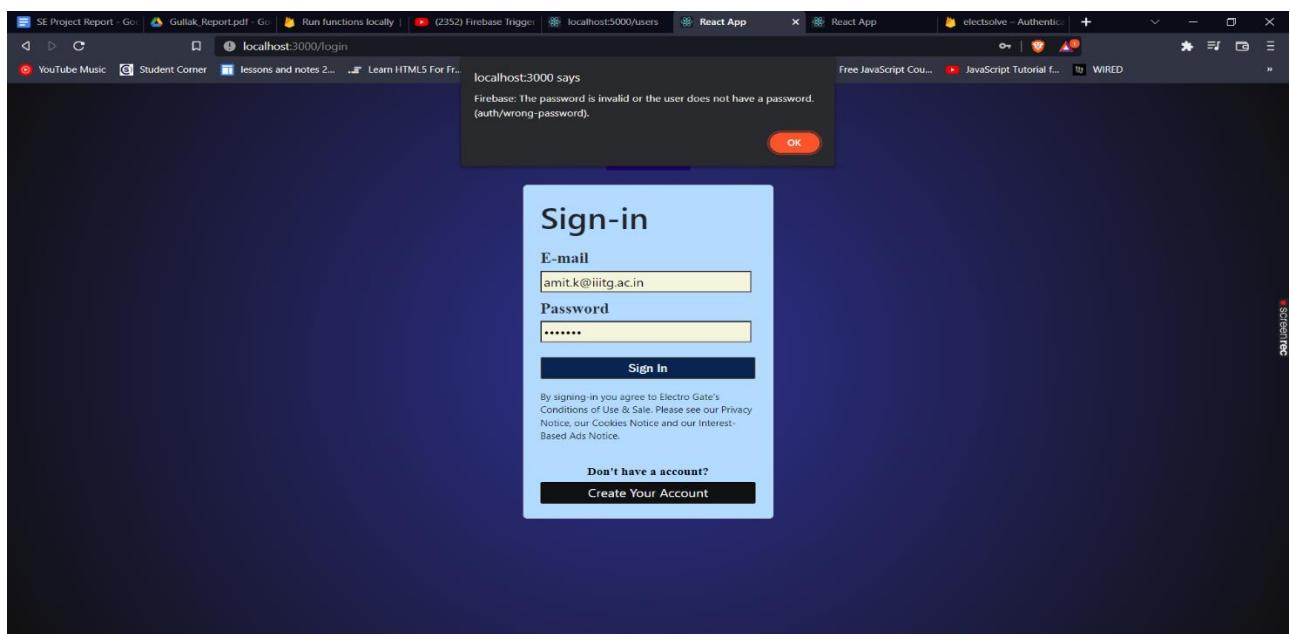


2.3 When you fill password wrong and press “Sign-in”.

Expectations: It will show error message.

Result: Passed.

Screen Shots/Logs:



2.4 When you fill new user Email and press “Sign-in”.

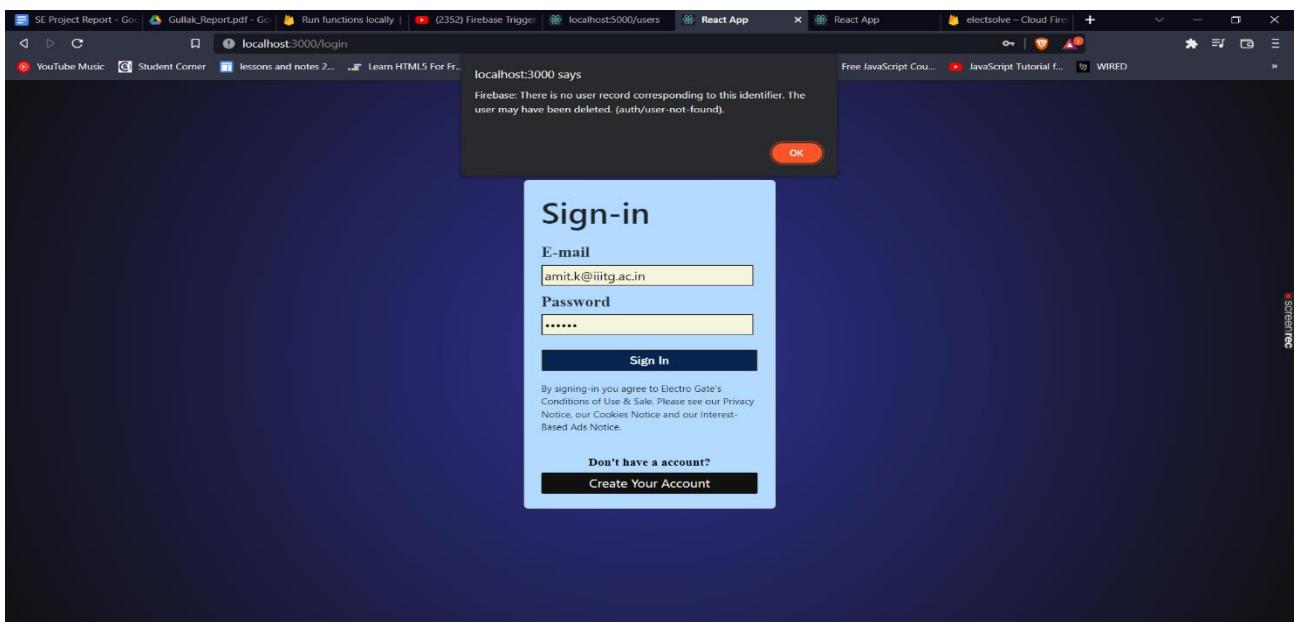
Expectations: It will show error message.



ElectroGate

Result: Passed.

Screen Shots/Logs:



3. Register as Professional:

Test-Cases:

3.1 When you fill all field and then press “Submit”.

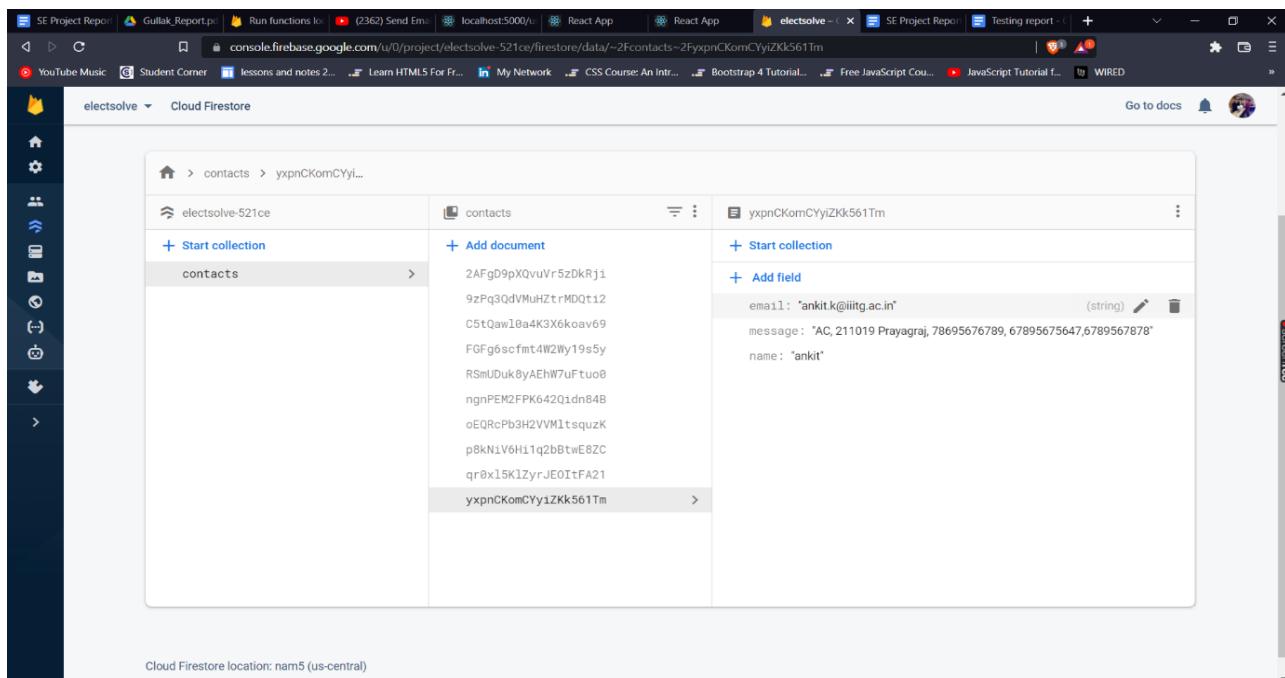
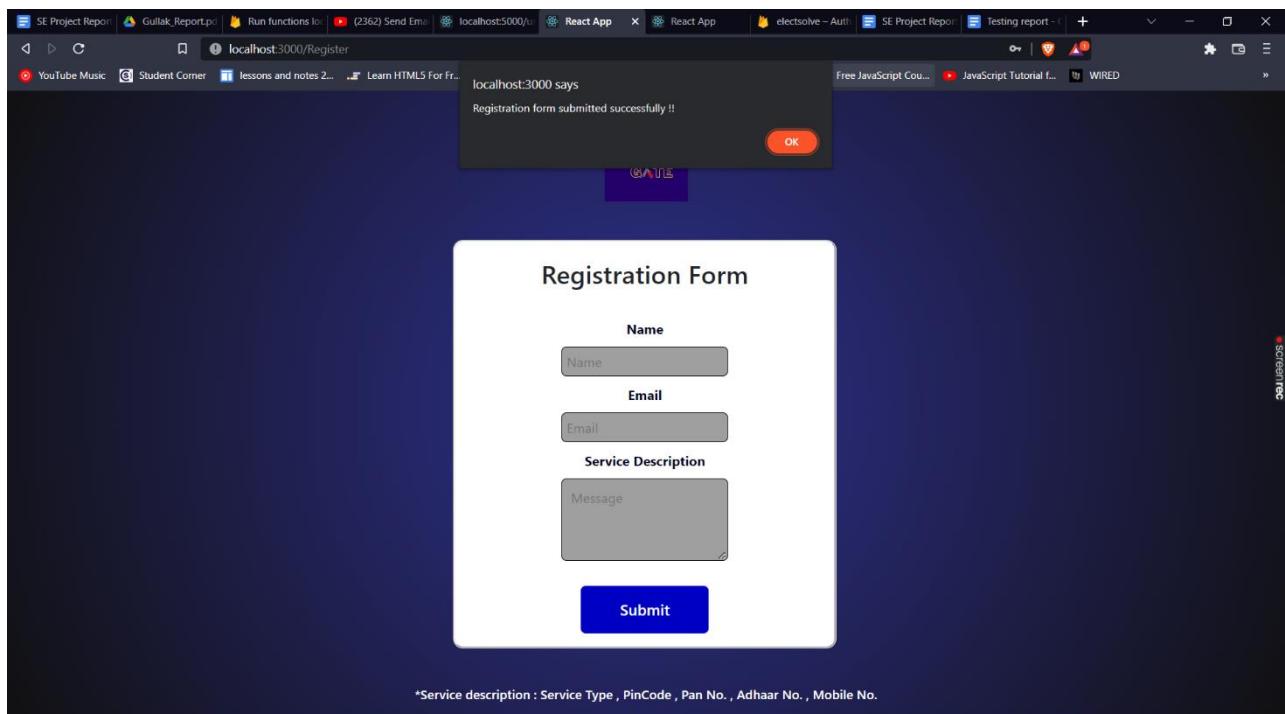
Expectations: It will show successful submitted message and store information in Database.

Result: Passed.

Screen Shots/Logs:



ElectroGate



3.2 When you do not fill all fields and then press "Submit".

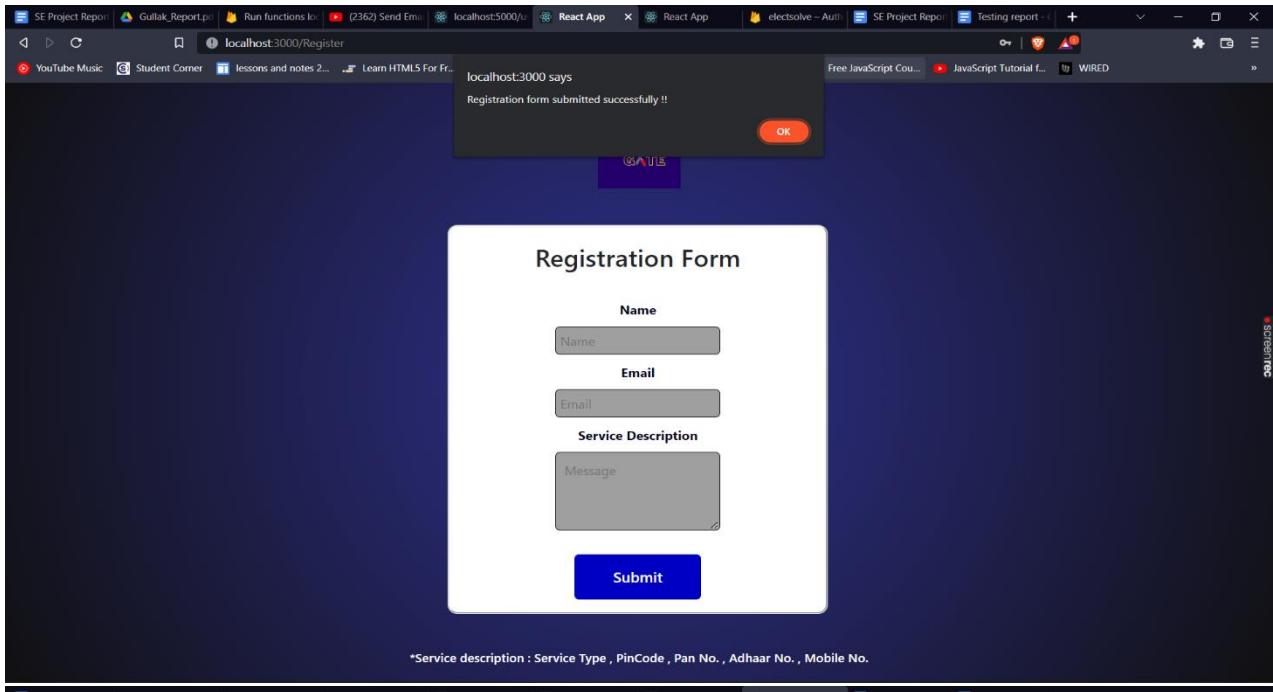
Expectations: It will show successful submitted message and store information in Database.

Result: Passed.

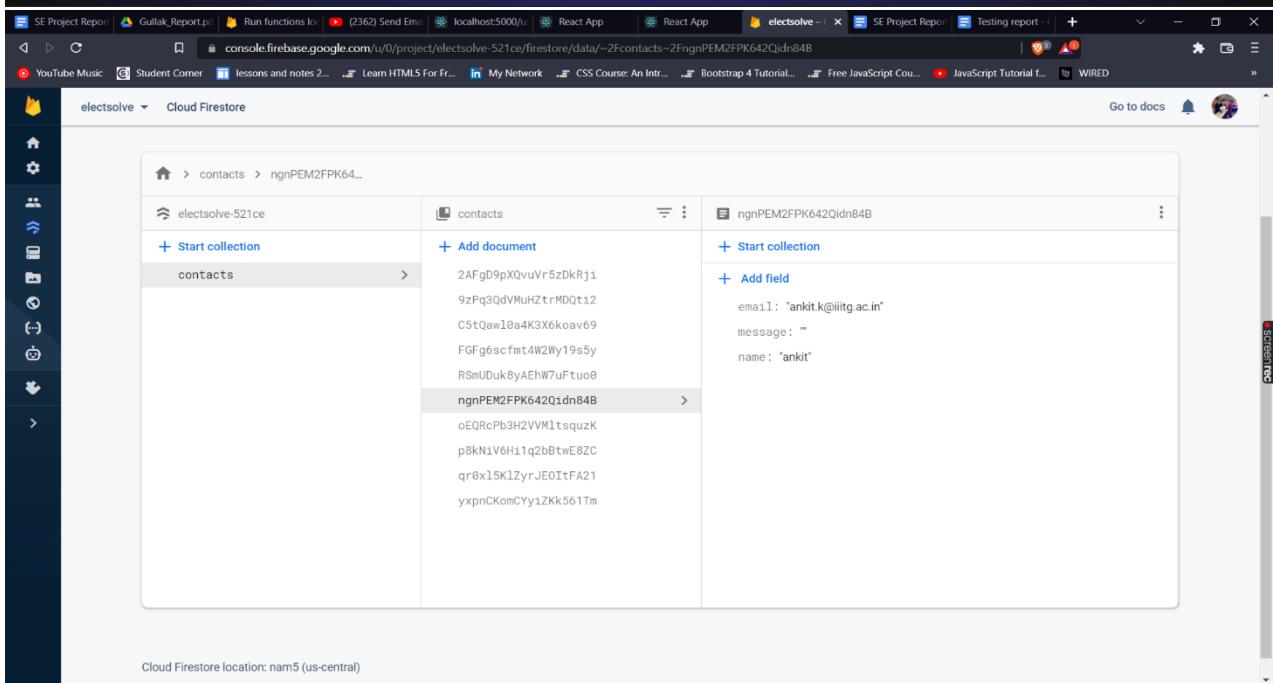


ElectroGate

Screen Shots/Logs:



*Service description : Service Type , PinCode , Pan No. , Adhaar No. , Mobile No.



3.3 When you do not fill any of field and then press “Submit”.

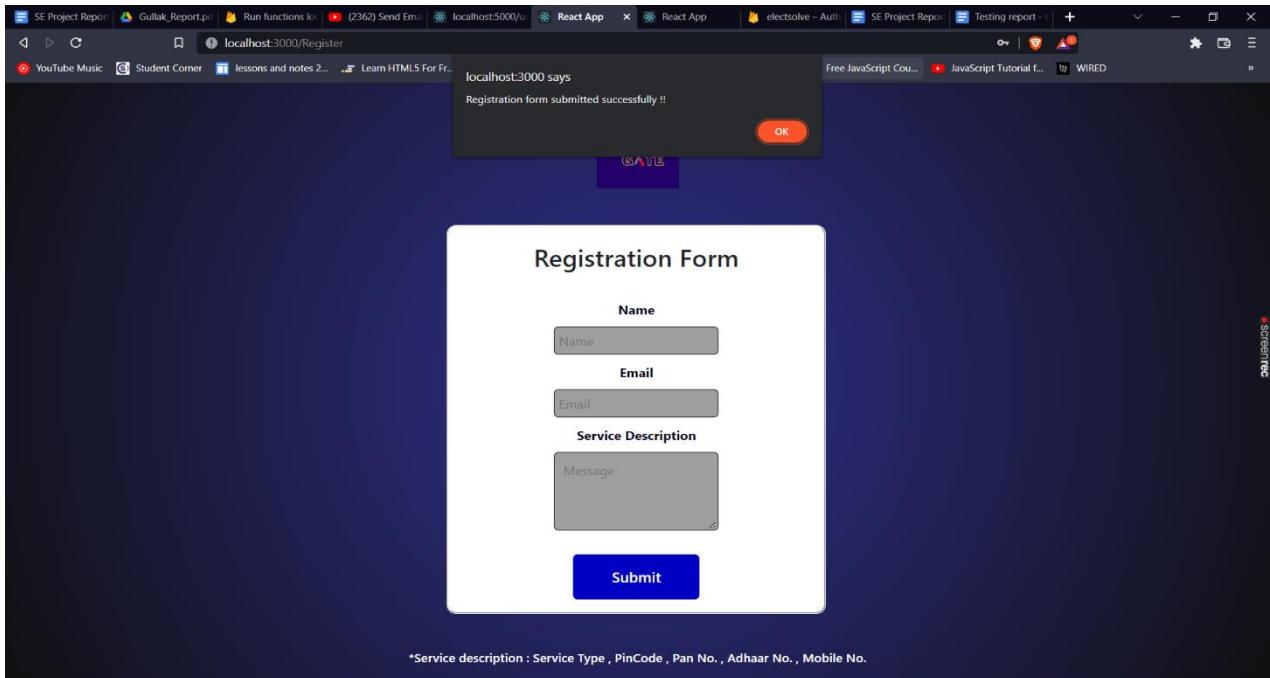
Expectations: It will show successful submitted message and store information in Database.

Result: Passed.

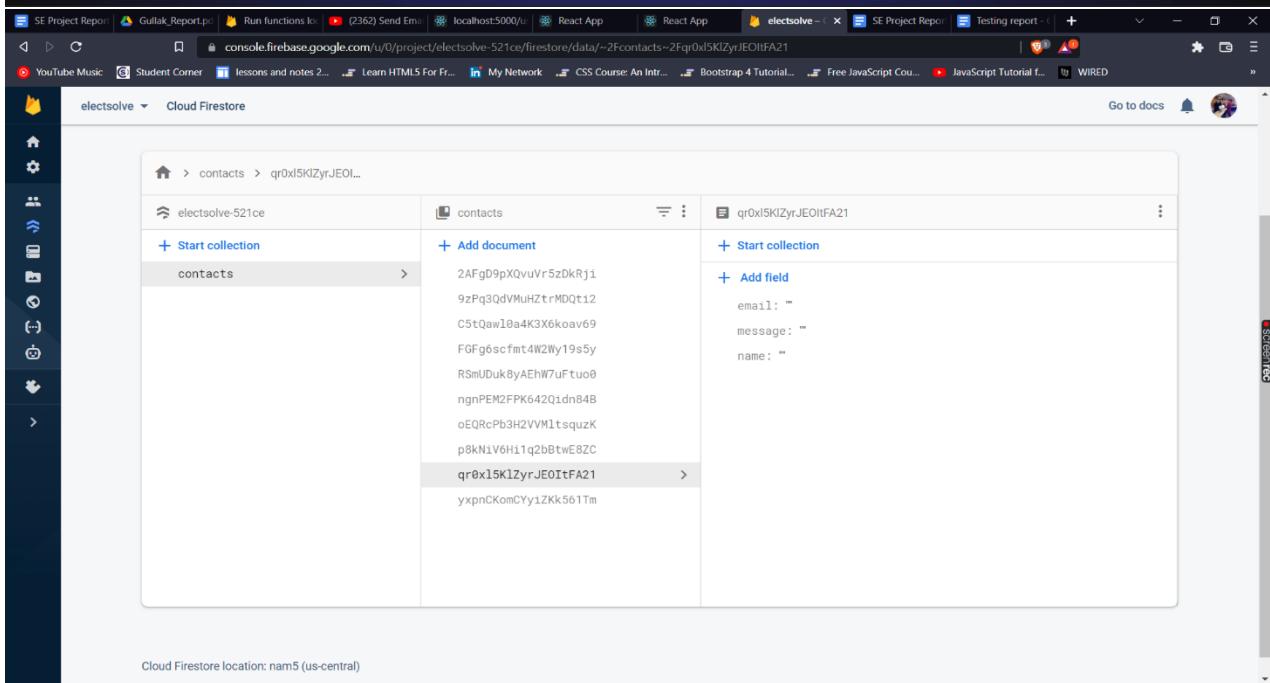


ElectroGate

Screen Shots/Logs:



*Service description : Service Type , PinCode , Pan No. , Adhaar No. , Mobile No.



4. Insurance-Form:

Test-Cases:

4.1 When you fill all field and then press “Send”.



ElectroGate

Expectations: It will show successful submitted message and email will send on admin Gmail.

Result: Passed.

Screen Shots/Logs:

The screenshot shows a browser window with multiple tabs open. A modal dialog box in the center says "localhost:3000 says Insurance form submitted successfully !!". Below it is the "Insurance Form" page, which has fields for Name (Ankit Kumar), Email (ankit.k@iitg.ac.in), and Appliance's Detail (AC, Jhusi Prayagraj, 678956784567, 7896785674). A "Send" button is at the bottom. At the bottom of the page, there is a note: "*Appliances Description : Type , Address , Adhaar No. , Mobile No."

The screenshot shows an open Gmail inbox. A new message from "Ankit Kumar" is highlighted. The message is from "majoranks136@gmail.com" and was sent at 2:27 PM (0 minutes ago). The message content is: "Hello Admin, You got a new message from Ankit Kumar: Customer Details: AC, Jhusi Prayagraj, 678956784567, 7896785674". Below the message, it says "Email sent via EmailJS.com". There are "Reply" and "Forward" buttons at the bottom of the message preview.

4.2 When you do not fill all fields and then press "Send".



ElectroGate

Expectations: It will show successful submitted message and email will send on admin Gmail.

Result: Passed.

Screen Shots/Logs:

The screenshot shows a browser window with a title bar containing multiple tabs. A modal dialog box is displayed in the center, stating "localhost:3000 says Insurance form submitted successfully !!". Below this, the "Insurance Form" page is visible, featuring fields for Name (Ankit Kumar), Email, and Appliance's Detail (AC, Jhusi Prayagraj, 678956784567, 7896785674). A blue "Send" button is at the bottom. The browser's address bar shows the URL "localhost:3000/Ins_form".

The screenshot shows a Gmail inbox with 5,982 messages. A new message from "Ankit Kumar" is selected, with the subject "New message from Ankit Kumar" and the recipient "majorankits136@gmail.com". The message body contains a greeting and a "Customer Details" section with the text "AC, Jhusi Prayagraj, 678956784567, 7896785674". The message was sent at 2:28 PM (0 minutes ago). At the bottom of the message view, there are "Reply" and "Forward" buttons, along with a note about the email being sent via EmailJS.com.

4.3 When you do not fill any of field and then press “Send”.



ElectroGate

Expectations: It will show successful submitted message and email will send on admin Gmail.

Result: Passed.

Screen Shots/Logs:

New message from [Inbox](#)

majorankits136@gmail.com
to me ▾
2:28 PM (0 minutes ago)

Hello Admin,
You got a new message from :
Customer Details:

Email sent via [EmailJS.com](#)

[Reply](#) [Forward](#)

Meet

Hangouts

Ankit +

No recent chats
Start a new one

localhost:3000 says
Insurance form submitted successfully !!

OK

Insurance Form

Name

Email

Appliance's Detail

Send

*Appliances Description : Type , Address , Adhaar No. , Mobile No.



ElectroGate

5. Order-Form:

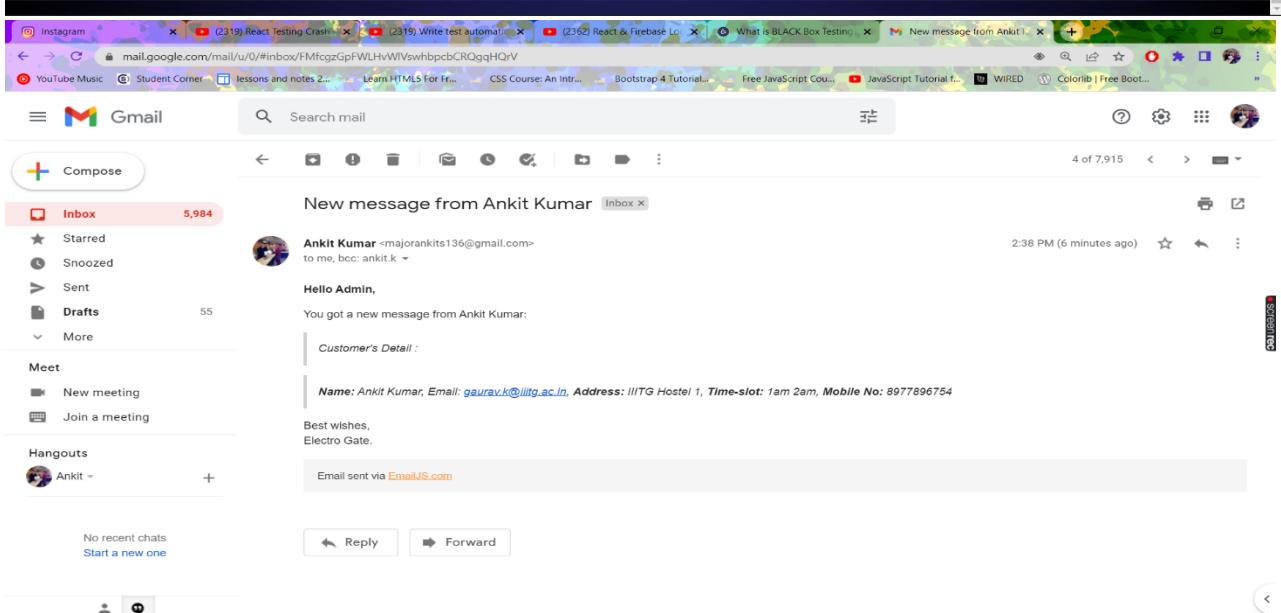
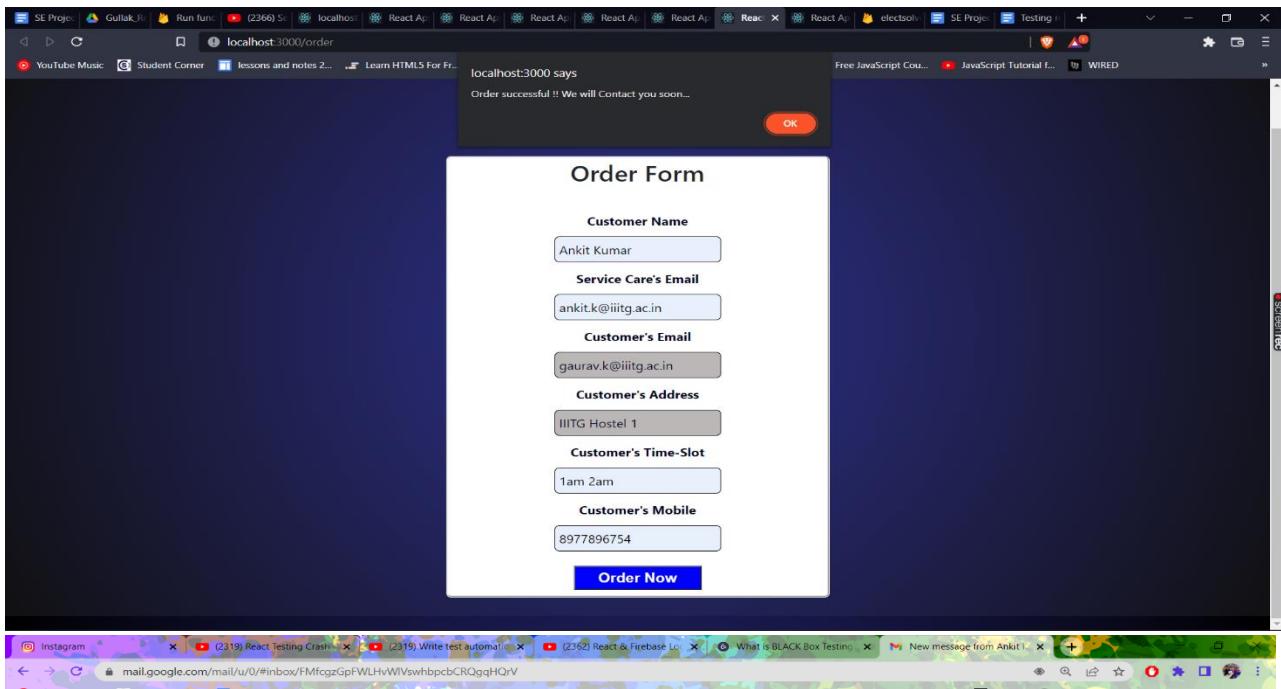
Test-Cases:

5.1 When you fill all field and then press “Order”.

Expectations: It will show successful submitted message and email will send on admin Gmail as well as Service Care.

Result: Passed.

Screen Shots/Logs:





ElectroGate

The screenshot shows a Gmail inbox with 2,665 messages. A new message from 'Ankit Kumar' is highlighted. The message content is as follows:

New message from Ankit Kumar [External] Inbox

Ankit Kumar
to majorankits136, bcc: me 14:38 (7 minutes ago)

Hello Admin,

You got a new message from Ankit Kumar:

Customer's Detail :

Name: Ankit Kumar, Email: gaurav.k@iitg.ac.in, Address: IITG Hostel 1, Time-slot: 1am 2am, Mobile No: 8977896754

Best wishes,
Electro Gate.

Email sent via [EmailJS.com](#)

Buttons at the bottom: Reply, Reply to all, Forward.

5.2 When you do not fill all fields (must fill Service care email) and then press “Order”.

Expectations: It will show successful submitted message and email will send on admin Gmail as well as Service Care.

Result: Passed.

Screen Shots/Logs:

The screenshot shows a browser window with multiple tabs open. A modal dialog box is displayed in the center, titled 'Order Form'. The form contains the following fields:

- Customer Name: Ankit Kumar
- Service Care's Email: ankit.k@iitg.ac.in
- Customer's Email: gaurav.k@iitg.ac.in
- Customer's Address: (empty input field)
- Customer's Time-Slot: (empty input field)
- Customer's Mobile: 8977896754

Below the form is a blue button labeled 'Order Now'.

A separate message box at the top says: 'localhost:3000 says Order successful !! We will Contact you soon...' with an 'OK' button.



ElectroGate

The screenshot shows a Gmail inbox with 5,982 messages. A new message from Ankit Kumar (majorankits136@gmail.com) is highlighted. The message content is as follows:

Ankit Kumar <majorankits136@gmail.com>
to me, bcc: ankit.k
Hello Admin,
You got a new message from Ankit Kumar:
Customer's Detail :
Name: Ankit Kumar, **Email:** gaurav.k@iitg.ac.in, **Address:** , **Time-slot:** , **Mobile No:** 8977896754
Best wishes,
Electro Gate.

Email sent via [EmailJS.com](#)

At the bottom, there are buttons for Reply, Reply to all, and Forward.

The second screenshot shows the same inbox, but the message from Ankit Kumar now has a yellow 'External' label next to the recipient name. The message content is identical to the first screenshot.

5.3 When you do not fill all fields (also not fill Service care email) and then press “Order”.

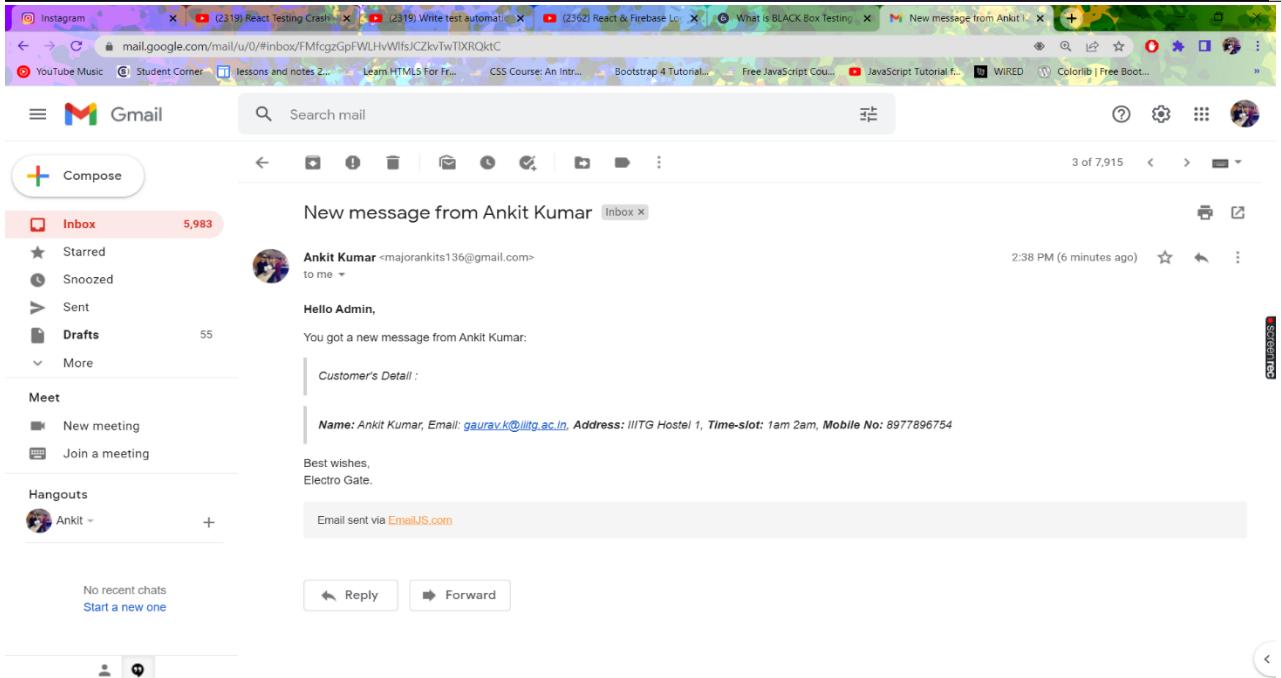
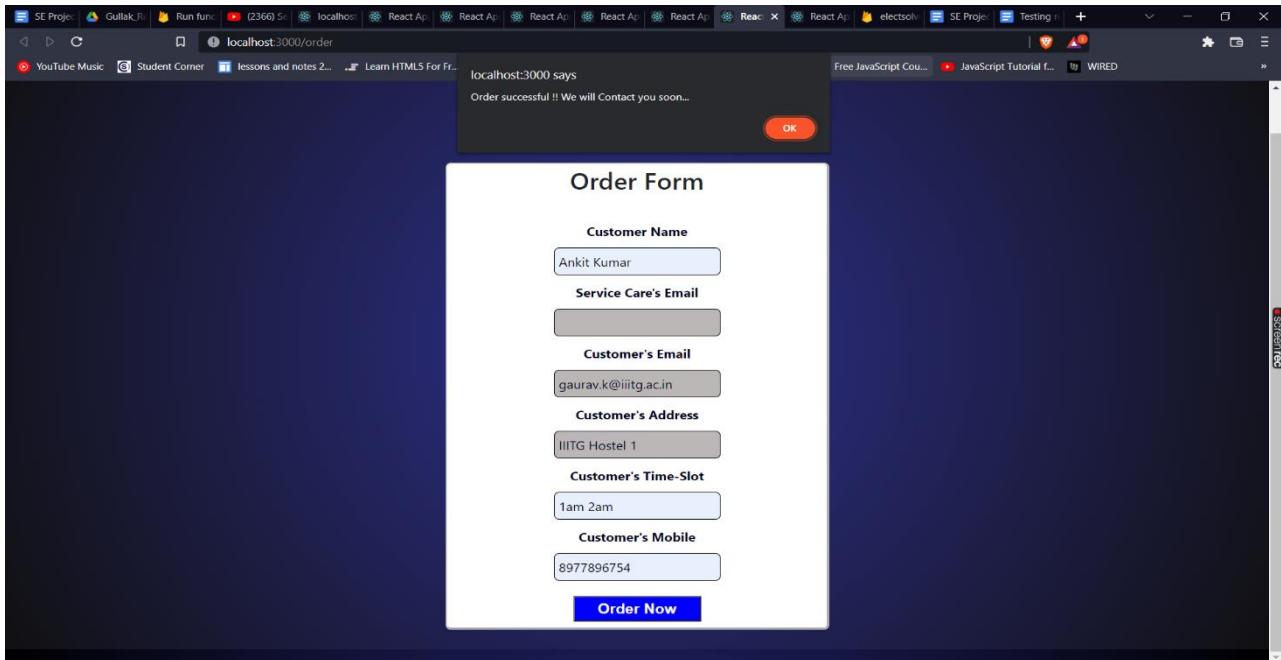
Expectations: It will show successful submitted message and email will send on only admin Gmail not to Service Care.

Result: Passed.



ElectroGate

Screen Shots/Logs:



5.4 When you do not fill any of field and then press “Order”.

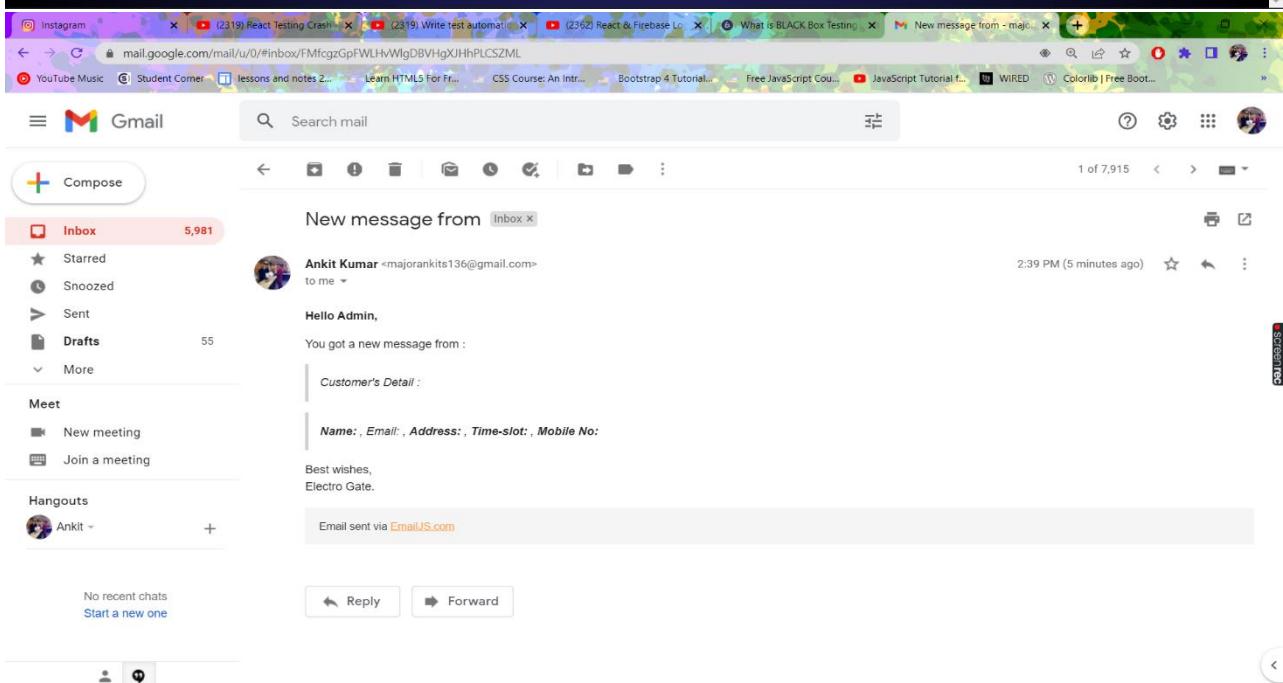
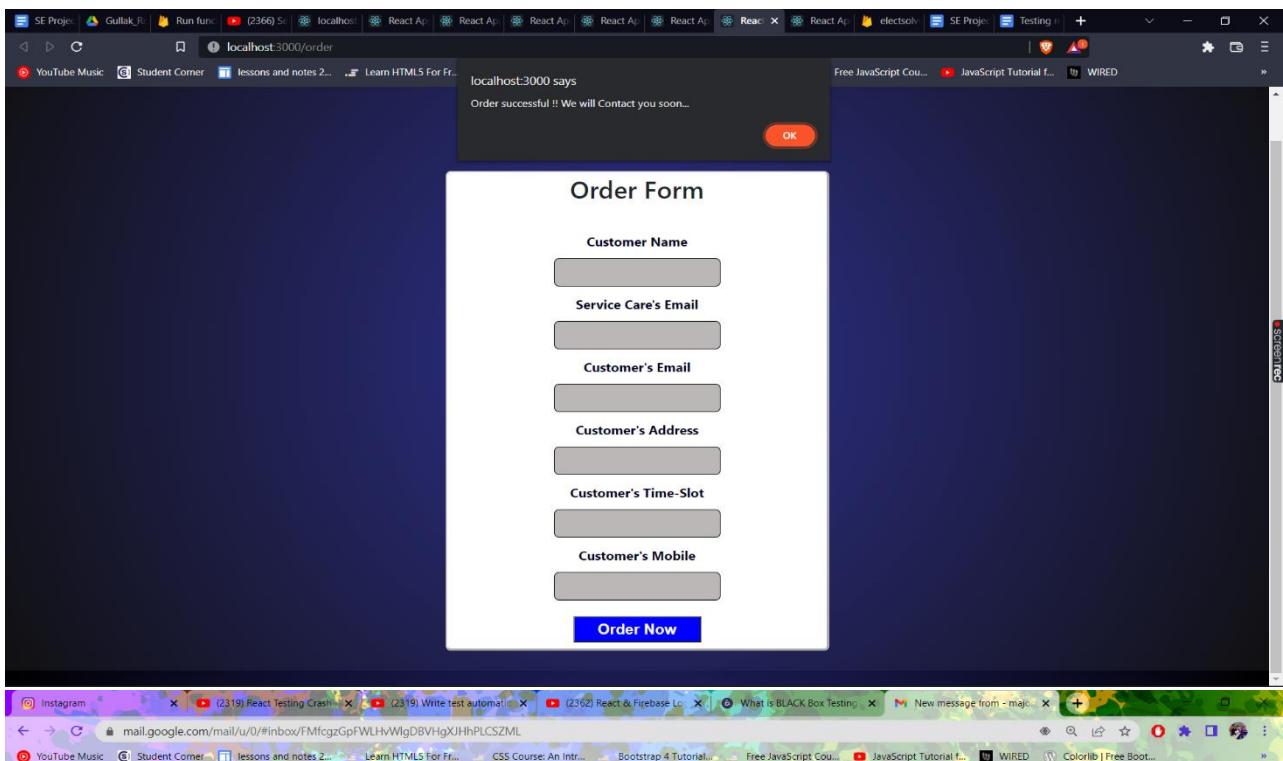
Expectations: It will show successful submitted message and email will send on only admin Gmail not to Service Care.

Result: Passed.

Screen Shots/Logs:



ElectroGate



6. Video-Call:

Test-Cases:



ElectroGate

6.1 When you fill right credential sent by service care and then press "Call" button.

Expectations: It will show ringing message on service care side.

Result: Passed.

Screen Shots/Logs:

Snippets for Video-Calling Part:

```
client > src > JS VideoCall.js > ...
1 import React, { useEffect, useRef, useState } from "react"
2 import Button from "@material-ui/core/Button"
3 import IconButton from "@material-ui/core/IconButton"
4 import TextField from "@material-ui/core/TextField"
5 import AssignmentIcon from "@material-ui/icons/Assignment"
6 import PhoneIcon from "@material-ui/icons/Phone"
7 import { CopyToClipboard } from "react-copy-to-clipboard"
8 import Peer from "simple-peer"
9 import io from "socket.io-client"
10 import "./VideoCall.css"
11
12 const socket = io.connect("http://localhost:5001")
13
14 function VideoCall() {
15     const [ me, setMe ] = useState("")
16     const [ stream, setStream ] = useState()
17     const [ receivingCall, setReceivingCall ] = useState(false)
18     const [ caller, setCaller ] = useState("")
19     const [ callersignal, setCallersignal ] = useState()
20     const [ callAccepted, setCallAccepted ] = useState(false)
21     const [ idfocall, setIdfocall ] = useState("")
22     const [ calledName, setCalledName ] = useState("")
23     const [ name, setName ] = useState("")
24     myVideo = useRef()
25     userVideo = useRef()
26     connectionRef = useRef()
27
28     useEffect(() => {
29         navigator.mediaDevices.getUserMedia({ video: true, audio: true }).then((stream) => {
30             setStream(stream)
31             myVideo.current.srcObject = stream
32         })
33     }
34     socket.on("me", (id) => {
35         setMe(id)
36     })
37 }
```



```
const express = require("express")
const http = require("http")
const app = express()
const server = http.createServer(app)
const io = require("socket.io")(server, {
  cors: {
    origin: "http://localhost:3000",
    methods: [ "GET", "POST" ]
  }
})

io.on("connection", (socket) => {
  socket.emit("me", socket.id)

  socket.on("disconnect", () => {
    socket.broadcast.emit("callEnded")
  })

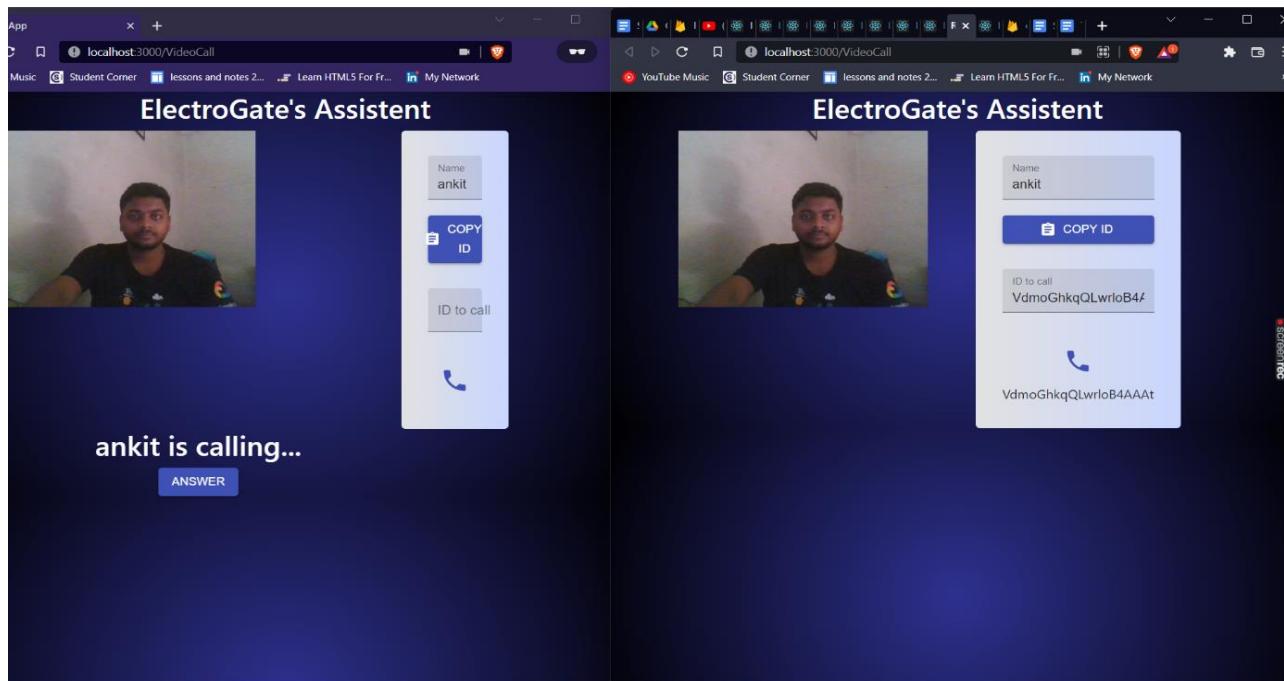
  socket.on("callUser", (data) => {
    io.to(data.userToCall).emit("callUser", { signal: data.signalData, from: data.from, name: data.name })
  })

  socket.on("answerCall", (data) => {
    io.to(data.to).emit("callAccepted", data.signal)
  })
})

server.listen(5001, () => console.log("server is running on port 5001"))
```



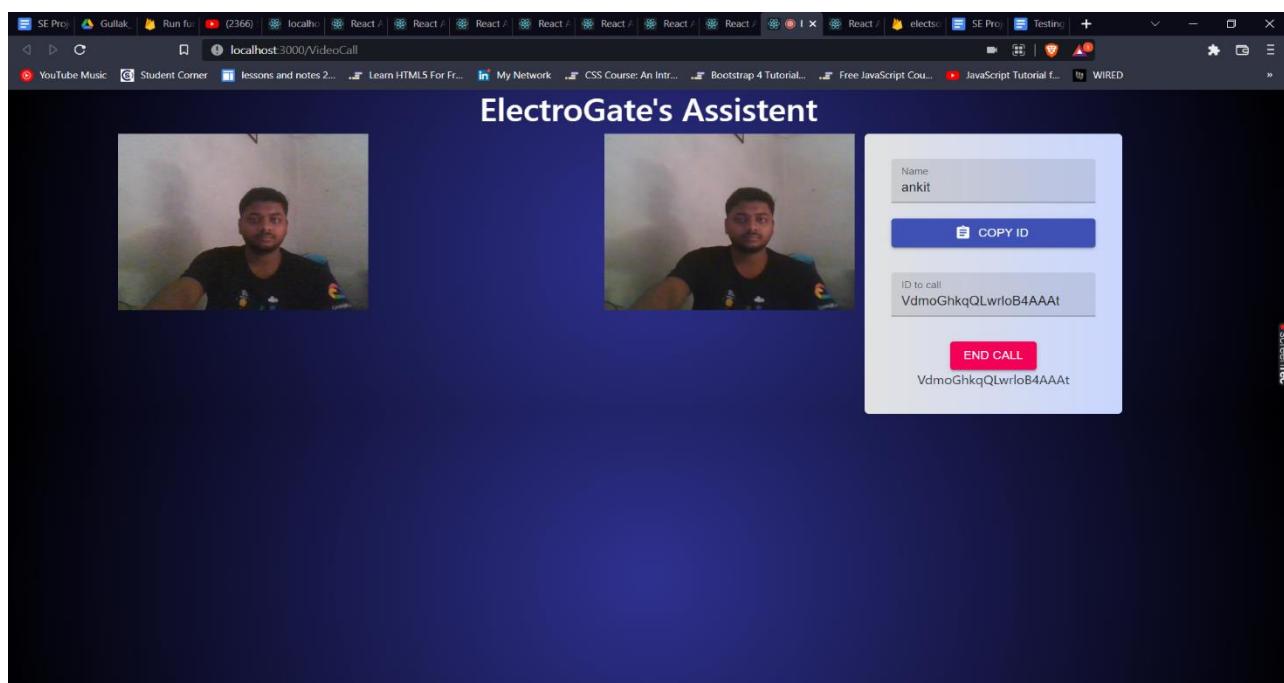
ElectroGate



6.2 When your service care pick-up call by press "Answer" button.

Expectations: The call will connect both sides.

Screen Shots/Logs:





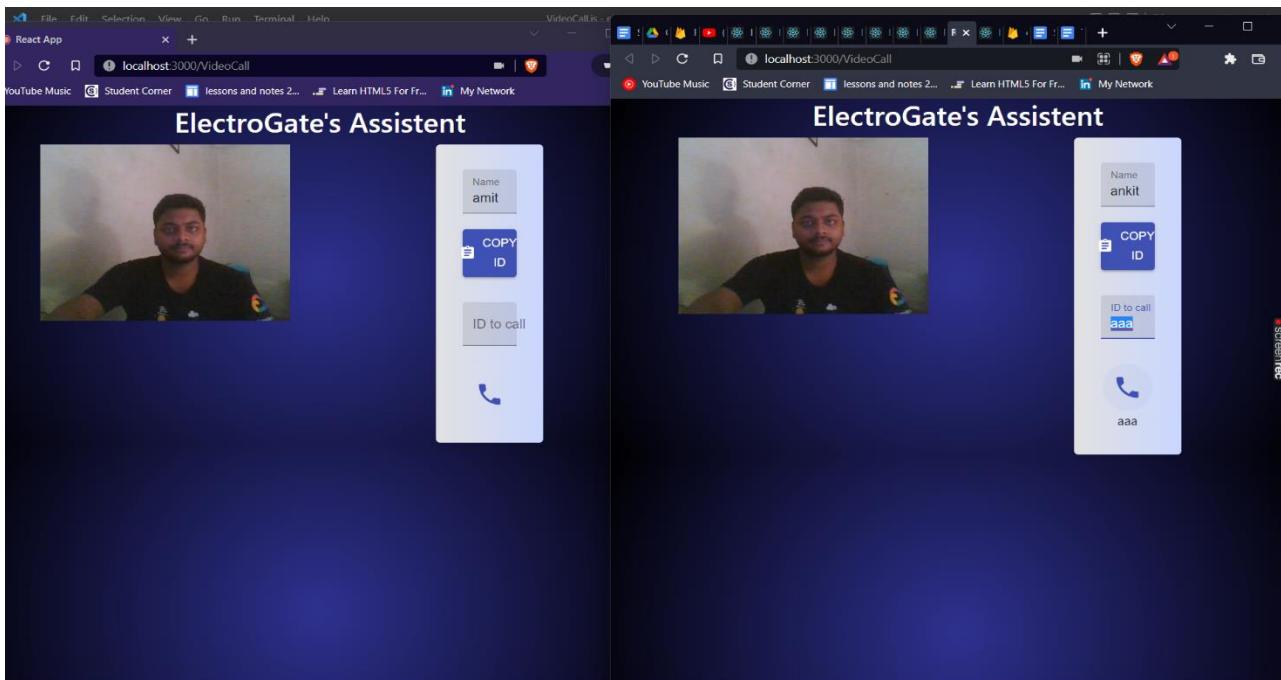
ElectroGate

6.3 When you do not fill right credential sent by service care and then press “Call” button.

Expectations: It will never show ringing message on service care side.

Result: Passed.

Screen Shots/Logs:



7. Feedback:

Test-Cases:

7.1 When Customer simply write their review and press on “Write” button.

Expectations: It will show on Feedback page.

Result: Passed.

Screen Shots/Logs:



ElectroGate

The screenshot shows a web browser window with the URL `localhost:3000/Comments`. The page title is "ElectroGate's Customer Feedback". There is a placeholder text "Write comment" at the top. Below it, there are four customer reviews:

- Customer 4/15/2022**: hii wow experience. [Reply](#) [Edit](#) [Delete](#)
- Customer 3/30/2022**: Great Experience. [Reply](#)
- Customer 3/30/2022**: Flawless connectivity. [Reply](#)
- Customer 3/30/2022**: On Time Service. [Reply](#)

7.2 When Customer Reply on others review and press on “Reply” button.

Expectations: It will show below to replied message.

Result: Passed.

Screen Shots/Logs:

The screenshot shows the same web browser window as before. A reply message has been added to the third review:

Customer 3/30/2022: Flawless connectivity.
This is not so. [Reply](#)

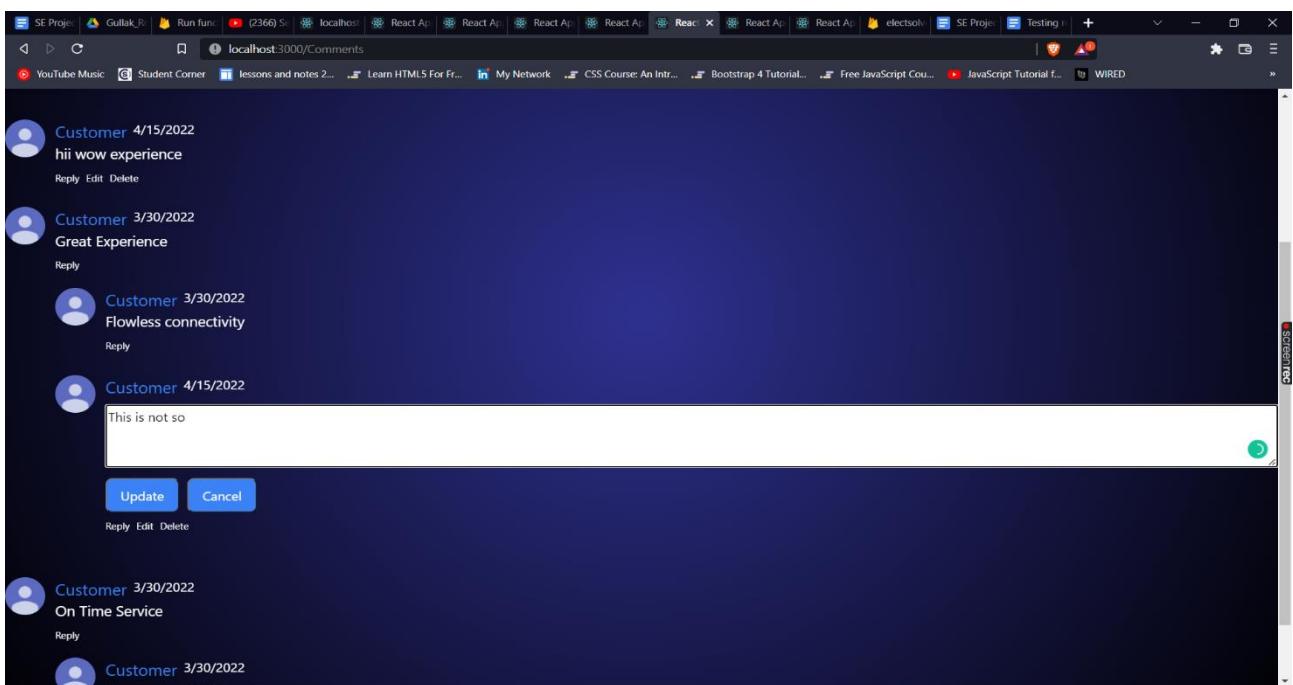


7.3 When Customer Edit their own review and press on “Edit” button.

Expectations: It will show updated one in place of previous review.

Result: Passed.

Screen Shots/Logs:



7.4 When Customer Delete their own review and press “Delete” button.

Expectations: It will show warning message for confirmation.

Result: Passed.

Screen Shots/Logs:



ElectroGate

The screenshot shows a list of customer comments on a dark-themed website. The comments are as follows:

- Customer 4/15/2022: hii wow experience. Options: Reply, Edit, Delete.
- Customer 3/30/2022: Great Experience. Options: Reply.
- Customer 3/30/2022: Flawless connectivity. Options: Reply.
- Customer 4/15/2022: This is not so good. Options: Reply, Edit, Delete.
- Customer 3/30/2022: On Time Service. Options: Reply.
- Customer 3/30/2022: it's amazing 24*7 service. Options: Reply.

A modal dialog box is overlaid on the page, asking "Are you sure you want to remove comment?". It has two buttons: "OK" and "Cancel".

From Admin-End:

1. Fake Rest API (Service-Care Database):

Test-Cases:

1.1 Admin can modify Service-Care Database

Expectations: It will show changed database.

Result: Passed.

Screen Shots/Logs:

Snippets: There is two snippets one for Database (db.json) and second for function which connect Database with all .js files where we can access Database.



ElectroGate

The screenshot shows two instances of Visual Studio Code side-by-side. Both instances have the title bar "electslove - Visual Studio Code".

Top Instance (db.json tab):

- Explorer sidebar: Shows files like Refrigerator.js, Register.css, Register.js, reportWebVitals.js, setupTests.js, Speaker.js, StateProvider.js, StringED.js, Student_App.css, Student_App.js, Team.css, Team.js, Television.js, theme.js, Vacum.js, Video_form.css, Video_form.js, Video_page.js, Video_call.css, VideoCall.js, Warranty.js, Washing.js, WaterPure.js, Who.css, Who.js, .firebaserc, .gitignore, db.json, firebase.json, firestore.indexes.json, firestore.rules, package-lock.json.
- Editor pane: Displays the db.json file content. It defines a "users" array with five entries, each containing fields: name, email, phone, service, address, pincode, id, and status. The first entry is for "Ram" with id 1 and status "Active". The second entry is for "Shant Singh" with id 2 and status "Active". The third entry is for "Tuta Kumar" with id 3 and status "Active". The fourth entry is for "Hari Ram" with id 4 and status "Active".
- Terminal pane: Shows the message "server is running on port 5001".

Bottom Instance (AC.js tab):

- Explorer sidebar: Shows files like client, functions, node_modules, .eslintrc.js, .gitignore, index.js, jest.config.js, jsonconfig.json, package-lock.json, package.json, public, src, comments, components, containers, css, AC.css, AC.js.
- Editor pane: Displays the AC.js file content. It includes imports for useState, useEffect, and axios. The code handles user sorting and filtering, making API calls to "localhost:5000/users" to fetch and update data based on search, sort, and filter parameters.
- Terminal pane: Shows the message "server is running on port 5001".

2. Client Database:

Test-Cases:

2.1 Admin can modify Client Database.



ElectroGate

Expectations: It will show changed database.

Result: Passed.

Screen Shots/Logs:

Snippets: Client Database(data.js)

The screenshot shows the Visual Studio Code interface with the file 'data.js' open. The code defines an array of objects representing testimonies. Each object has properties: 'image', 'name', and 'testimony'. The names correspond to the six avatars shown in the accompanying image. The 'testimony' property contains a quote from each customer.

```
client > src > components > data.js > data > e testmony
1  export const data = [
2    {
3      image: "/avatars/avatar-1.png",
4      name: "Anima",
5      testimony:
6        "Awesome experience and service available at 24*7 this is amazing.",
7    },
8    {
9      image: "/avatars/avatar-2.png",
10     name: "Amit",
11     testimony:
12       "I insured my appliances from ElectroGate and actually they has best policy.",
13    },
14    {
15      image: "/avatars/avatar-3.png",
16      name: "Kishika",
17      testimony:
18        "I my glad to connect with ElectroGate that served me best customer experience.",
19    },
20    {
21      image: "/avatars/avatar-4.png",
22      name: "Suman",
23      testimony:
24        "I am from UP and I used this service during pandemic and I found it's very hassel.",
25    },
26    {
27      image: "/avatars/avatar-5.png",
28      name: "Rohit",
29      testimony:
30        "I belong to Haryana and this area haven't good service cares and now we all got best service care at 24*7. ",
31    },
32    {
33      image: "/avatars/avatar-6.png",
34      name: "Sanjay",
35      testimony:
36        "less Panework... best insurance policy.. experienced Service care and fast service these all are in one place. "
]
```

3. Blogging:

Test-Cases:

3.1 Admin can post new Blogs on Website.

Expectations: It will show new Blog on website.

Result: Passed.

Screen Shots/Logs:

Snippets: Blog.js which have Blog details



ElectroGate

The screenshot shows a Visual Studio Code interface with the following details:

- File Explorer:** Shows a tree view of the project structure under "ELECTRSOLVE". Files listed include icons, data.js, playstation.jpg, testimonials.css, Testimonials.js, containers.css, AC.css, AC.js, ankit's pic.jpeg, AntiBill.css, AntiBill.js, api.js, App.css, App.js, App.test.js, apit.jpeg,AuthProvider.js, Blog.css, Blog.js (which is selected), Blog.test.js, Bulb.js, CCTV.js, Clock.js, ComputerAcc.js, Decor.css, Decor.js, DeskLamp.js, Dishwasher.js, Electro.png, Fan.js, and firebase.js.
- Code Editor:** The main editor area displays the content of "Blog.js". The code includes imports for "client", "tCollectionTriggers.js", "jsonconfig.json", "jest.config.js", "package.json", "functions", "firebase.json", "db.json", and "Blog.js". The code itself is a React component structure with various components like Card, CardContent, Typography, Box, Grid, and CardActions.
- Terminal:** The bottom right corner shows the terminal output: "server is running on port 5001".
- Bottom Status Bar:** Shows file status (master*), line count (Ln 81, Col 60), character count (Spaces: 2, UTF-8, CRLF), and icons for Go Live, Prettier, and Node.js.