Name:- Shinde Smit Manohar
Roll no:- 52
Class:- TE Comp(B)

Project:- Business Contact Manager

Code:-

```
const express = require('express');
const mongoose = require('mongoose');
const cors = require('cors');

const app = express();
app.use(cors());
app.use(express.json());

mongoose.connect('mongodb+srv://JOBList:2005@cluster0.zlw9n.mongodb.net/con
t actManager', {
 useNewUrlParser: true,
 useUnifiedTopology: true,
});

const contactSchema = new mongoose.Schema({
 name: String,
 email: String,
 phone: String,
 company: String,
 notes: String,
});

const Contact = mongoose.model('Contact', contactSchema);

// Create
app.post('/api/contacts', async (req, res) => {
 const contact = new Contact(req.body);
 await contact.save();
 res.send(contact);
});

// Read
app.get('/api/contacts', async (req, res) => {
 const contacts = await Contact.find();
 res.send(contacts);
});

// Update
app.put('/api/contacts/:id', async (req, res) => {
 const contact = await Contact.findByIdAndUpdate(req.params.id, req.body, { new:
```

```
true });
 res.send(contact);
});

// Delete
app.delete('/api/contacts/:id', async (req, res) => {
 await Contact.findByIdAndDelete(req.params.id);
 res.send({ success: true });
});

app.listen(5000, () => {
 console.log('Server started on port 5000');
});
```

Frontend
App.js

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

function App() {
 const [contacts, setContacts] = useState([]);
 const [form, setForm] = useState({ name: '', email: '', phone: '', company: '', notes: ''
});
 const [editId, setEditId] = useState(null);

 const fetchContacts = async () => {
        const res = await axios.get('http://localhost:5000/api/contacts');
        setContacts(res.data);
 };

 useEffect(() => {
        fetchContacts();
 }, []);

 const handleSubmit = async (e) => {
        e.preventDefault();
        if (editId) {
 await axios.put(`http://localhost:5000/api/contacts/${editId}`, form); } else
        {
 await axios.post('http://localhost:5000/api/contacts', form); }
        setForm({ name: '', email: '', phone: '', company: '', notes: '' });
        setEditId(null);
        fetchContacts();
 };

 const handleEdit = (contact) => {
```

```
          setForm(contact);
          setEditId(contact._id);
 };

 const handleDelete = async (id) => {
          await axios.delete(`http://localhost:5000/api/contacts/${id}`);
          fetchContacts();
 };

 return (
          <>
 <div className="container">
 <h1>�� Business Contact Manager</h1>
 <form onSubmit={handleSubmit}>
 <input value={form.name} onChange={e => setForm({ ...form, name:
e.target.value })} placeholder="Name" required />
 <input value={form.email} onChange={e => setForm({ ...form, email:
e.target.value })} placeholder="Email" />
 <input value={form.phone} onChange={e => setForm({ ...form, phone:
e.target.value })} placeholder="Phone" />
 <input value={form.company} onChange={e => setForm({ ...form, company:
e.target.value })} placeholder="Company" />
 <textarea value={form.notes} onChange={e => setForm({ ...form, notes:
e.target.value })} placeholder="Notes"></textarea>
 <button type="submit">{editId ? 'Update Contact' : 'Add Contact'}</button>
</form>
 </div>

 <div className="cards-container">
 {contacts.map((contact) => (
 <div className="contact-card" key={contact._id}>
 <strong>{contact.name}</strong> <span>({contact.company})</span>
<p>�� {contact.phone}</p>
 <p>✉️ {contact.email}</p>
 {contact.notes && <p>�� {contact.notes}</p>}
 <div>
 <button onClick={() => handleEdit(contact)}>Edit</button>  <button
onClick={() => handleDelete(contact._id)}>Delete</button>  </div>
 </div>
 ))}
 </div>
          </>
 );
}

export default App;
```

indes.js

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

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

App.css

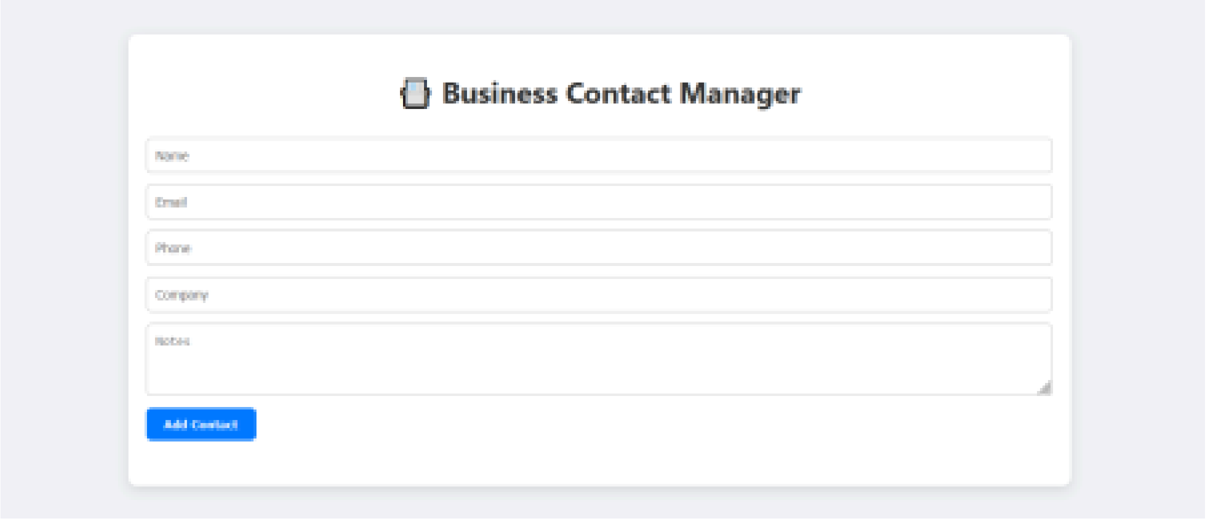
```
body {
 font-family: 'Segoe UI', sans-serif;
 background-color: #f0f2f5;
 margin: 0;
 padding: 0;
 }

 /* Container for Form */
 .container {
 max-width: 1000px;
 margin: 40px auto;
 padding: 20px;
 background: #ffffff;
 border-radius: 10px;
 box-shadow: 0 4px 12px rgba(0, 0, 0, 0.1);
 }

 h1 {
 text-align: center;
 margin-bottom: 30px;
 color: #333;
 }

 /* Form Styling */
 form {
 display: flex;
 flex-direction: column;
 gap: 12px;
 margin-bottom: 30px;
 }

 form input,
 form textarea {
 width: 100%;
 padding: 10px;
 border: 1px solid #ccc;
 border-radius: 6px;
 font-size: 14px;
 box-sizing: border-box;
 }

 form textarea {
 resize: vertical;
 min-height: 80px;
 }

 form button {
 align-self: flex-start;
 background-color: #007bff;
 color: white;
 border: none;
 padding: 10px 20px;
 border-radius: 6px;
 cursor: pointer;
 font-weight: bold;
 transition: background-color 0.3s ease;
}

 form button:hover {
 background-color: #0056b3;
 }

 /* Grid layout for contact cards */
.cards-container {
 max-width: 1000px;
 margin: 0 auto 40px auto;
 display: grid;
 grid-template-columns: repeat(auto-fill, minmax(280px, 1fr));
gap: 20px;
 padding: 0 20px;
 }

 /* Individual contact card */
 .contact-card {
 background: #fafafa;
 border: 2px solid #ccc;
 border-radius: 10px;
 padding: 20px;
 box-shadow: 1px 3px 6px rgba(0, 0, 0, 0.05);  transition:
transform 0.2s ease, box-shadow 0.2s ease;  }

 .contact-card:hover {
 transform: translateY(-4px);
 box-shadow: 2px 6px 12px rgba(0, 0, 0, 0.1);
}

 .contact-card strong {
 font-size: 18px;
 color: #333;
 }

 .contact-card span {
 color: #555;
 font-style: italic;
 }

 .contact-card p {
 margin: 8px 0;
 font-size: 14px;
 color: #444;
 }

 .contact-card div {
 margin-top: 12px;
 }

 /* Buttons inside contact card */
 .contact-card button {
 margin-right: 10px;
 padding: 6px 12px;
 font-size: 14px;
 border: none;
 border-radius: 4px;
 cursor: pointer;
 }

 .contact-card button:first-child {
 background-color: #ffc107;
 color: #000;
 }

 .contact-card button:last-child {
 background-color: #dc3545;
 color: white;
 }

 .contact-card button:first-child:hover {
 background-color: #e0a800;
 }
```

```
.contact-card button:last-child:hover {
background-color: #c82333;
}
```

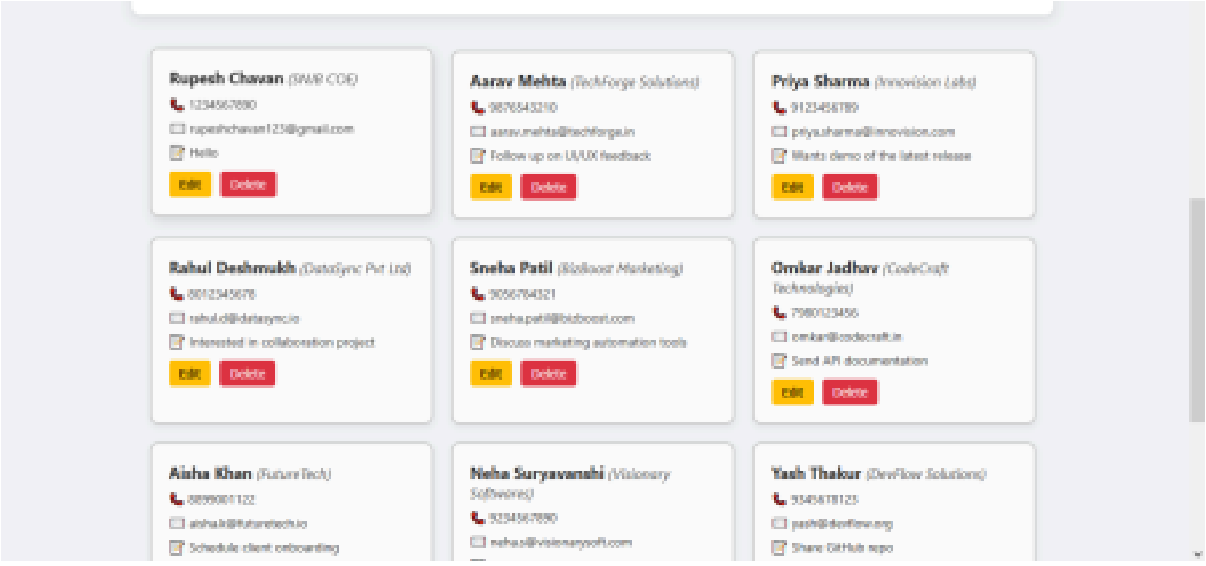
Index.html

```
<!DOCTYPE html>
<html lang="en">
 <head>
        <meta charset="utf-8" />
        <link rel="icon" href="%PUBLIC_URL%/favicon.ico" />
        <meta name="viewport" content="width=device-width, initial-scale=1" />
        <meta name="theme-color" content="#000000" />
        <title>Business Contact Manager</title>
 </head>
 <body>
        <noscript>You need to enable JavaScript to run this app.</noscript>
        <div id="root"></div>
 </body>
</html>
```

OutPut



Business Contact Manager



Rupesh Chavan *(SNJB COE)*

☎ 1234567890

✉ rupeshchavan123@gmail.com

📝 Hello

Aarav Mehta *(TechForge Solutions)*

☎ 9876543210

✉ aarav.mehta@techforge.in

📝 Follow up on UI/UX feedback

Priya Sharma *(Innovision Labs)*

☎ 9123456789

✉ priya.sharma@innovision.com

📝 Wants demo of the latest release

Rahul Deshmukh *(DataSync Pvt Ltd)*

☎ 8012345678

✉ rahul.d@datasync.io

📝 Interested in collaboration project

Sneha Patil *(BizBoost Marketing)*

☎ 9056784321

✉ sneha.patil@bizboost.com

📝 Discuss marketing automation tools

Omkar Jadhav *(CodeCraft Technologies)*

☎ 7980123456

✉ omkar@codecraft.in

📝 Send API documentation

Aisha Khan *(FutureTech)*

☎ 8899001122

✉ aisha.k@futuretech.io

📝 Schedule client onboarding

Neha Suryavanshi *(Visionary Softwares)*

☎ 9234567890

✉ neha.s@visionarysoft.com

Yash Thakur *(DevFlow Solutions)*

☎ 9345678123

✉ yash@devflow.org

📝 Share GitHub repo