5. Develop a Contact Manager Backend Application using Node.js, Express.js and MongoDB.

app.js

```
const express = require("express");
const dotenv = require("dotenv").config();
const connectDB = require('./config/dbConnect');
connectDB();

const app = express();

const PORT = 5000;
app.use(express.json());

const contactRoutes = require('./routes/contactRoutes');

app.use("/api/contacts",contactRoutes);

app.listen(PORT,()=>{
    console.log("SERVER IS RUNNING AT PORT 5000");

});
```

routes/contactRoutes.js

```
const express = require("express");
const router = express.Router();
const {getContact,createContact,updateContact,deleteContact} =
require('../controllers/contactController');


router.get('/:id',getContact);
router.post('/',createContact);
router.put('/:id',updateContact)
router.delete('/:id',deleteContact)

module.exports = router;  //export
```

config/dbConnection.js

```
const mongoose = require('mongoose');
const dotenv = require('dotenv').config();

const connectDB = async()=>{
    try{
        const connect = await mongoose.connect(process.env.CONNECTION_STRING);
        console.log("database connected");
```

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

    }
}

module.exports = connectDB;
```

model/contactModel.js

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

const contactSchema = mongoose.Schema({
    name:{
        type:String,
        required:[true,"Please add the contact name"]
    },
    email:{
        type:String,
        required:[true,"Please add the contact email"]
    },
    phone:{
        type:String,
        required:[true,"Please add the contact number"]

    }
});

module.exports = mongoose.model("Contact",contactSchema);
```

controllers/contactController.js

```
const Contact = require("../model/contactModel");

const getContact = async (req, res) => {

    try {
        const contact = await Contact.findById(req.params.id);
        if(!contact){
            res.status(400).json({ message: "contact not found" });
        }
        res.status(200).json(contact);
        }
    catch (error) {
        console.log(error)
```

```
    }
}
const createContact = async (req, res) => {
    try {
        const { name, email, phone } = req.body;
        if (!name || !email || !phone) {
            res.status(400).json({ message: "All fields are required" });
        }
        const contact = await Contact.create({
            name,
            email,
            phone
        });
        res.status(201).json({ message: "A new contact is created" });

       }
    catch (error) {
        console.log(error)

    }
}
const updateContact = async (req, res) => {
    try {
        const contact = await Contact.findById(req.params.id);
        if(!contact){
        res.status(400).json({ message: "contact not found" });
      }
        // Use findByIdAndUpdate to update the document
        const updatedContact = await Contact.findByIdAndUpdate(
            req.params.id, // ID of the document to update
            req.body,      // Data to update with (sent from the client)
            { new: true }  // Option: returns the newly updated document
        );

        // Return the updated contact object
        res.status(200).json(updatedContact);

    }
    catch (error) {
        console.log(error)
        res.status(500).json({ message: "Server error during update" });
    }
}
const deleteContact = async (req, res) => {
    try {
        const contact = await Contact.findById(req.params.id);
        if(!contact){
            res.status(404).json({ message: "Contact not found" });
```

```
            return;
        }

        // Use findByIdAndDelete (or contact.deleteOne()) to remove the
document
        await Contact.findByIdAndDelete(req.params.id);

        res.status(200).json(contact); // Respond with the deleted object or a
success message

    }
    catch (error) {
        console.log(error)
        res.status(500).json({ message: "Server error during deletion" });
    }
}
module.exports = {

    getContact,
    createContact,
    updateContact,
    deleteContact
}
```

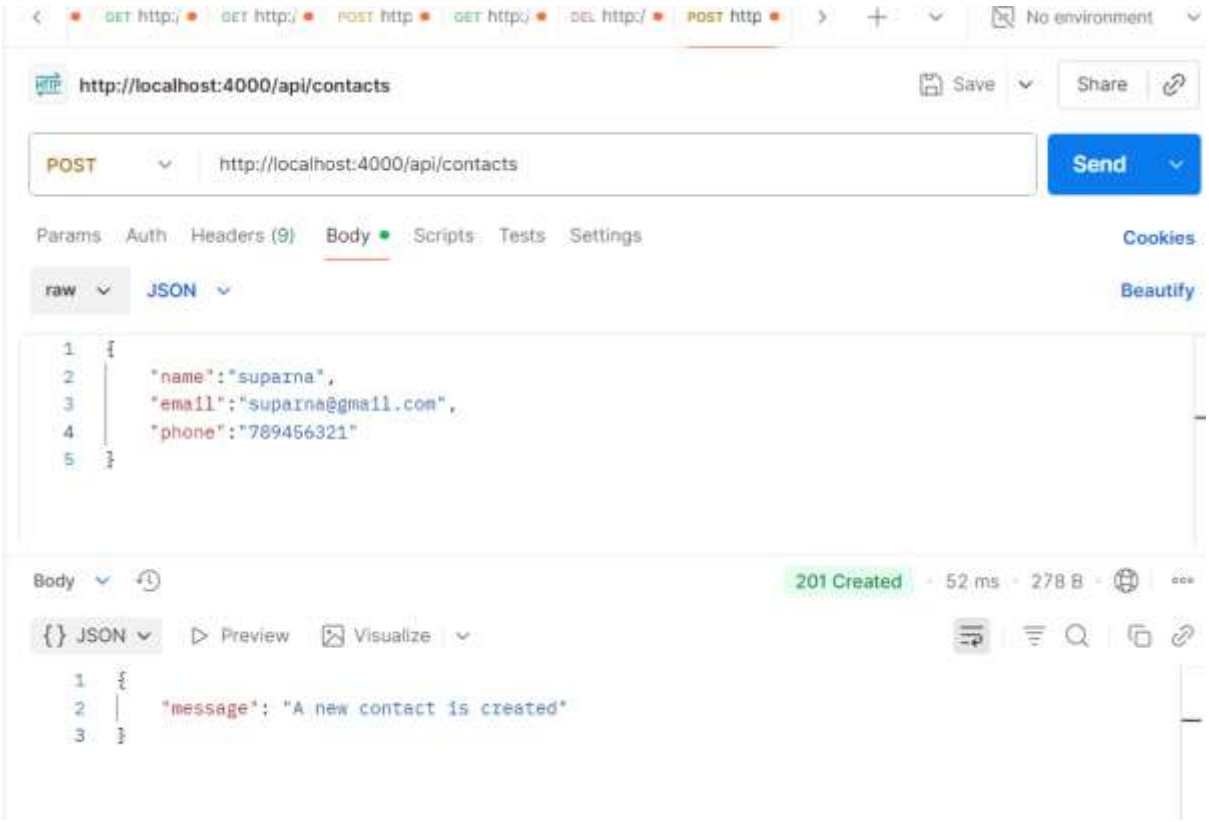
.env

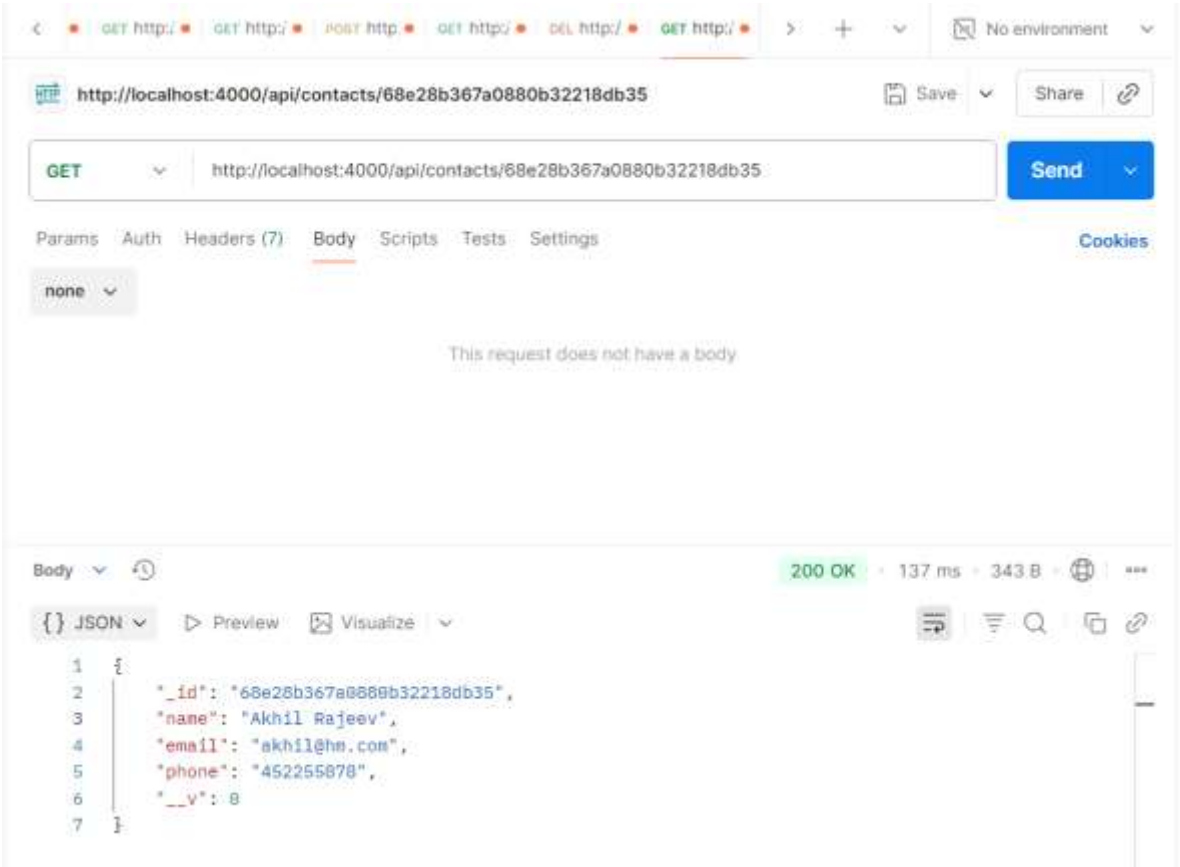
```
CONNECTION_STRING = "ATTTACH YOUR-MONGODB-CONNECTION STRING HERE"
```

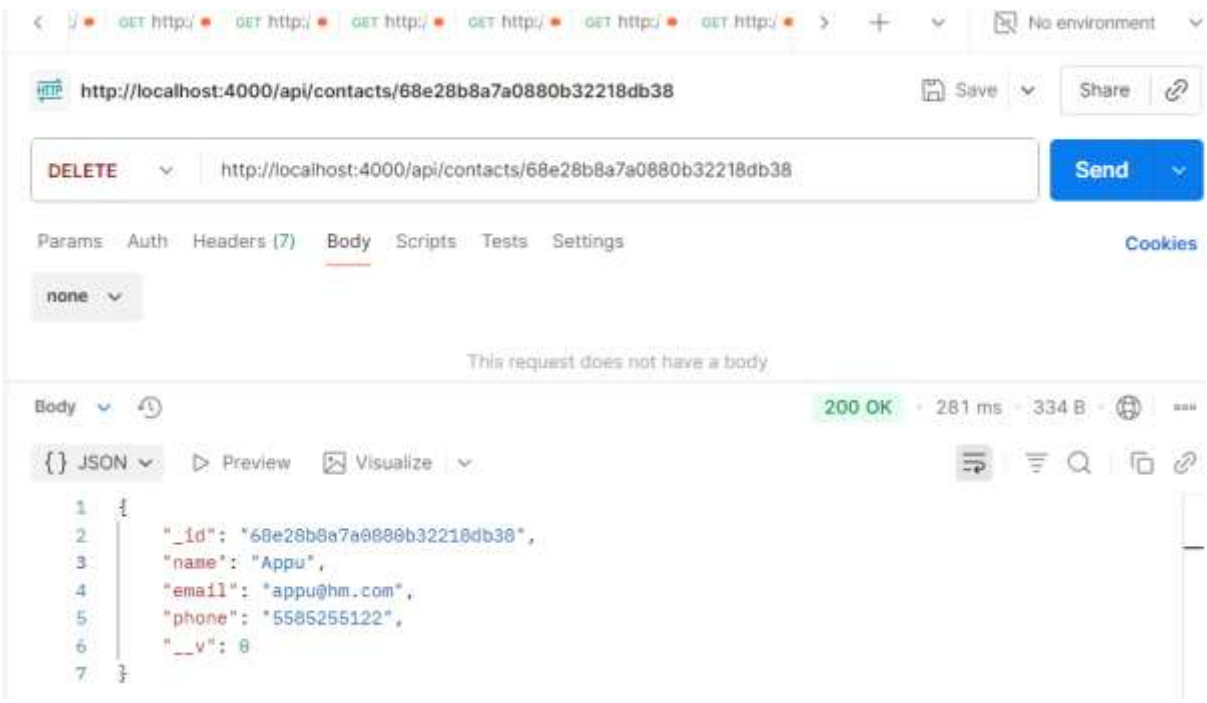
.gitignore

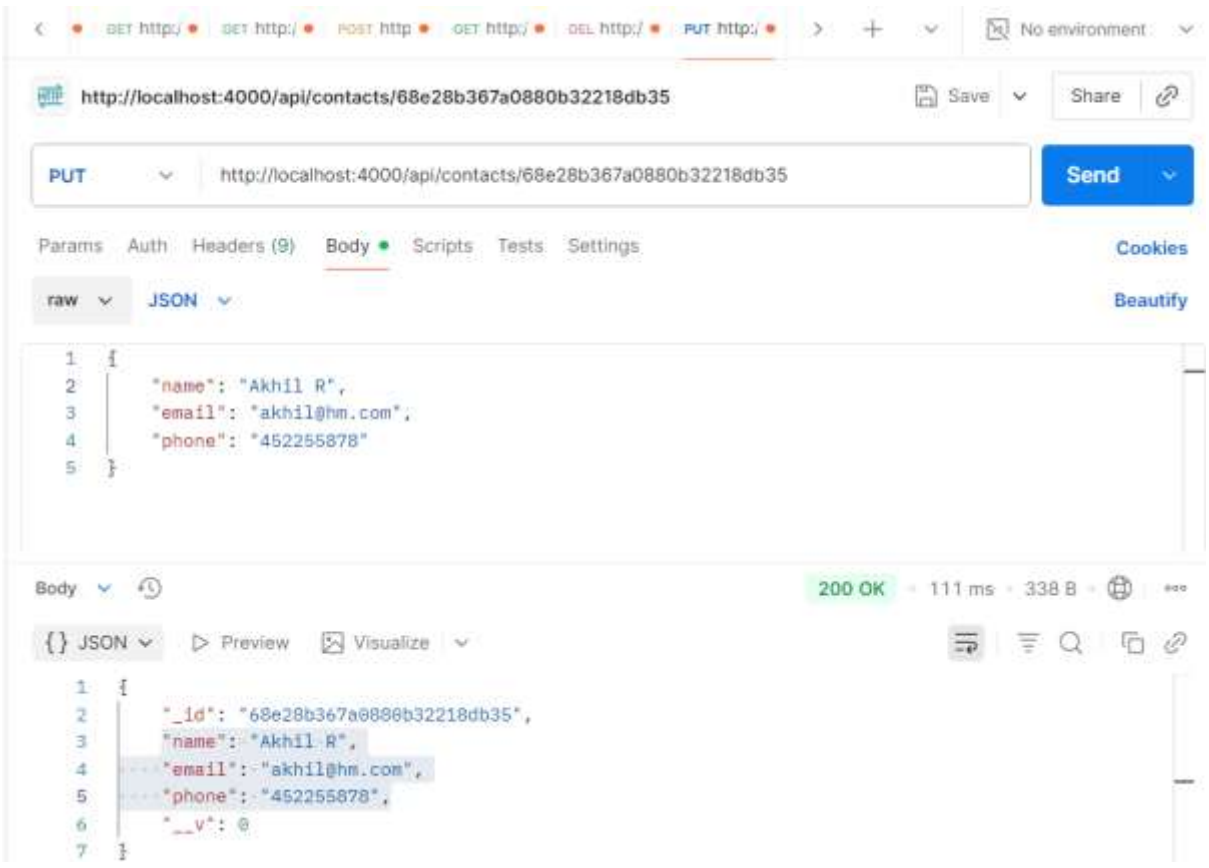
```
.env
/node_modules
```

Output:


http://localhost:4000/api/contacts
Save
Share
POST
http://localhost:4000/api/contacts
Send
Params
Auth
Headers (9)
Body
Scripts
Tests
Settings
Cookies
raw
JSON
Beautify
{
"name":"suparna",
"email":"suparna@gmail.com",
"phone":"789456321"
}
Body
201 Created
52 ms
278 B
{} JSON
Preview
Visualize
{
"message": "A new contact is created"
}


http://localhost:4000/api/contacts/68e28b367a0880b32218db35
Save
Share
GET
http://localhost:4000/api/contacts/68e28b367a0880b32218db35
Send
Params
Auth
Headers (7)
Body
Scripts
Tests
Settings
Cookies
none
This request does not have a body
Body
200 OK
137 ms
343 B
{} JSON
Preview
Visualize
{
"_id": "68e28b367a0880b32218db35",
"name": "Akhil Rajeev",
"email": "akhil@hm.com",
"phone": "452255078",
"__v": 0
}


No environment
http://localhost:4000/api/contacts/68e28b8a7a0880b32218db38
Save
Share
DELETE
http://localhost:4000/api/contacts/68e28b8a7a0880b32218db38
Send
Params
Auth
Headers (7)
Body
Scripts
Tests
Settings
Cookies
none
This request does not have a body
Body
200 OK
281 ms
334 B
JSON
Preview
Visualize
{
"_id": "68e28b8a7a0880b32218db38",
"name": "Appu",
"email": "appu@hm.com",
"phone": "5585255122",
"__v": 0
}


No environment
http://localhost:4000/api/contacts/68e28b367a0880b32218db35
Save
Share
PUT
http://localhost:4000/api/contacts/68e28b367a0880b32218db35
Send
Params
Auth
Headers (9)
Body
Scripts
Tests
Settings
Cookies
raw
JSON
Beautify
{
"name": "Akhil R",
"email": "akhil@hm.com",
"phone": "452255878"
}
Body
200 OK
111 ms
338 B
JSON
Preview
Visualize
{
"_id": "68e28b367a0880b32218db35",
"name": "Akhil R",
"email": "akhil@hm.com",
"phone": "452255878",
"__v": 0
}