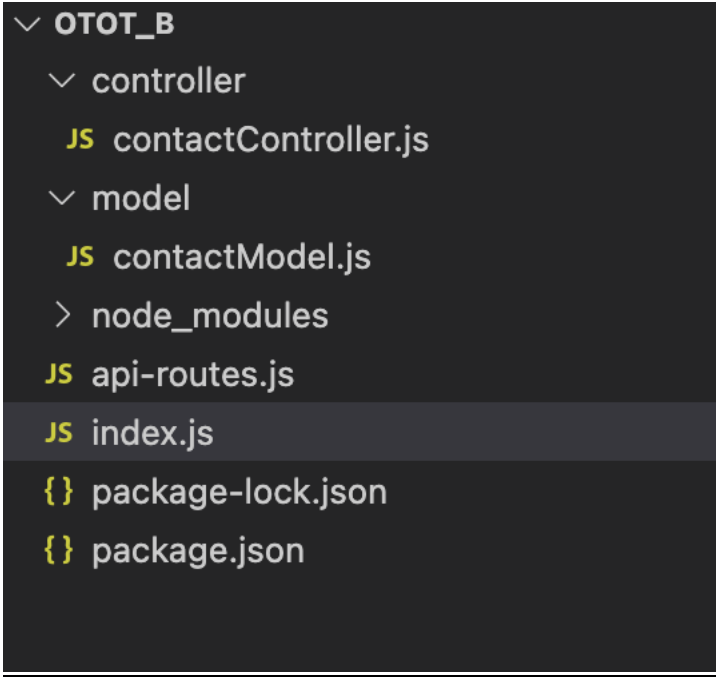
Name: Justin Yip Jeng Teng Matric No.: A0222764E
Github link: https://github.com/justinyjt/OTOT-B

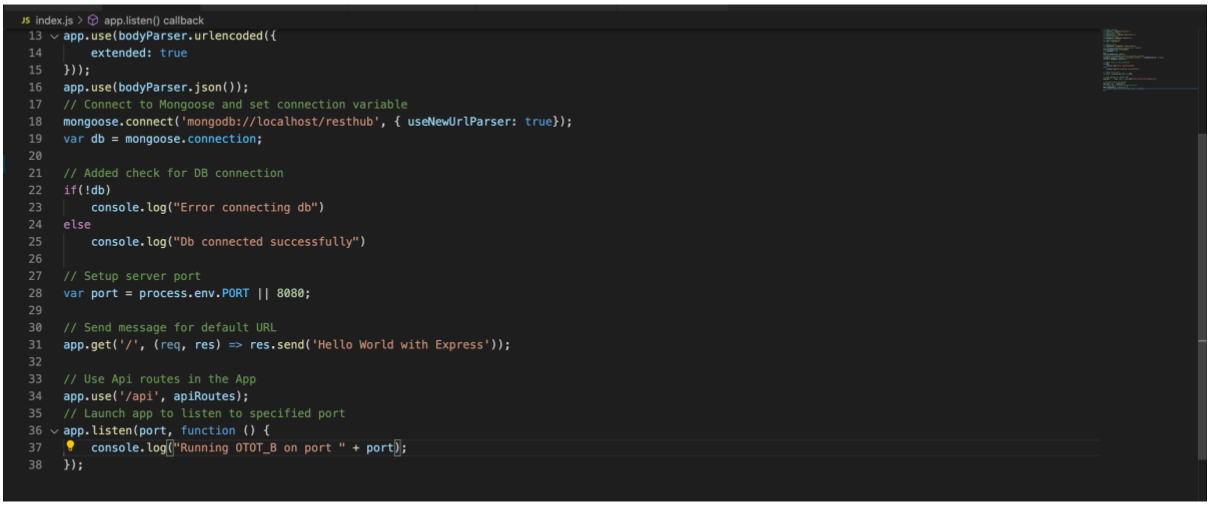
<u>B1:</u>

<u>Screenshots of directory structure and programs:</u>



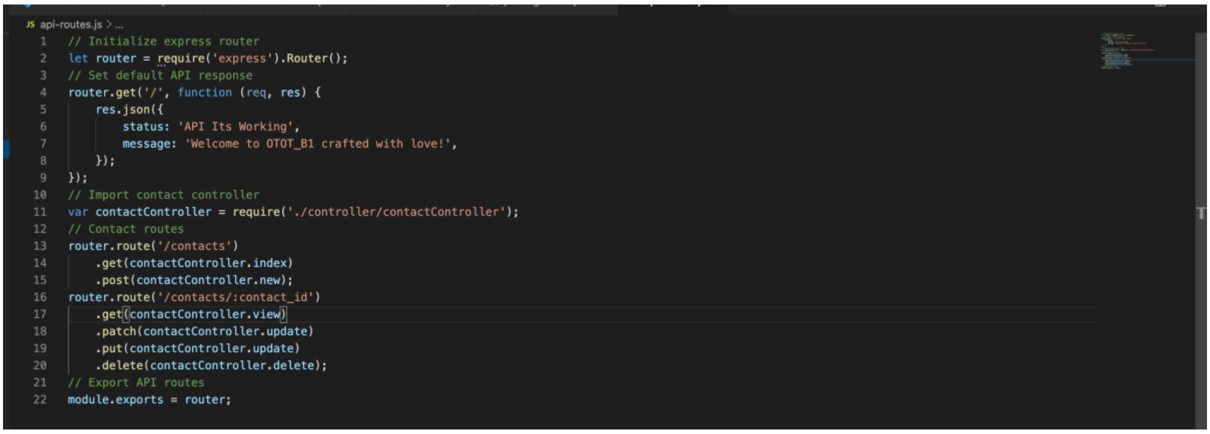
```
OTOT_B
  controller
    contactController.js
  model
    contactModel.js
  node_modules
  api-routes.js
  index.js
  package-lock.json
  package.json
```

Directory structure



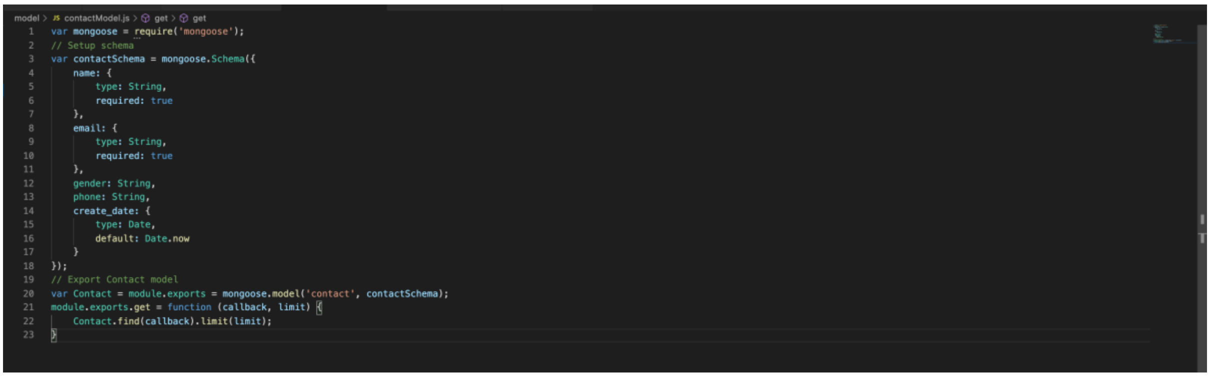
```
app.use(bodyParser.urlencoded({
    extended: true
}));
app.use(bodyParser.json());
// Connect to Mongoose and set connection variable
mongoose.connect('mongodb://localhost/resthub', { useNewUrlParser: true});
var db = mongoose.connection;

// Added check for DB connection
if(!db)
    console.log("Error connecting db")
else
    console.log("Db connected successfully")

// Setup server port
var port = process.env.PORT || 8080;

// Send message for default URL
app.get('/', (req, res) => res.send('Hello World with Express'));

// Use Api routes in the App
app.use('/api', apiRoutes);
// Launch app to listen to specified port
app.listen(port, function () {
    console.log("Running OTOT_B on port " + port);
});
```

index.js



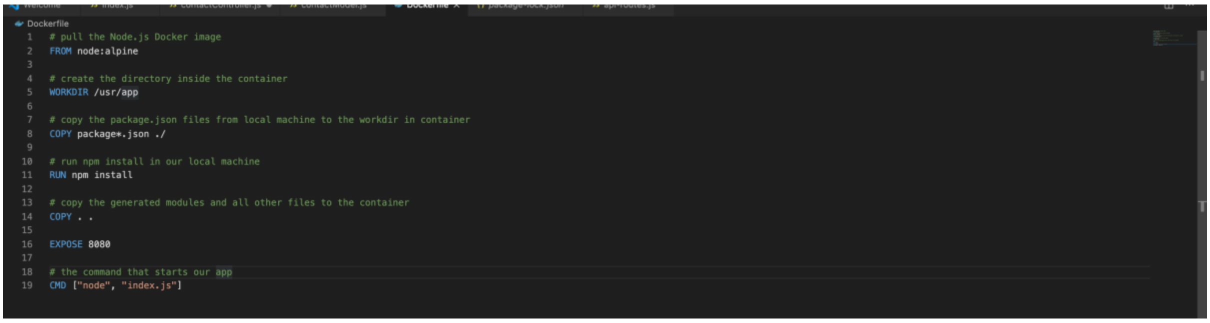
```javascript
// Initialize express router
let router = require('express').Router();
// Set default API response
router.get('/', function (req, res) {
    res.json({
        status: 'API Its Working',
        message: 'Welcome to OTOT_B1 crafted with love!',
    });
});
// Import contact controller
var contactController = require('./controller/contactController');
// Contact routes
router.route('/contacts')
    .get(contactController.index)
    .post(contactController.new);
router.route('/contacts/:contact_id')
    .get(contactController.view)
    .patch(contactController.update)
    .put(contactController.update)
    .delete(contactController.delete);
// Export API routes
module.exports = router;
```

api-routes.js



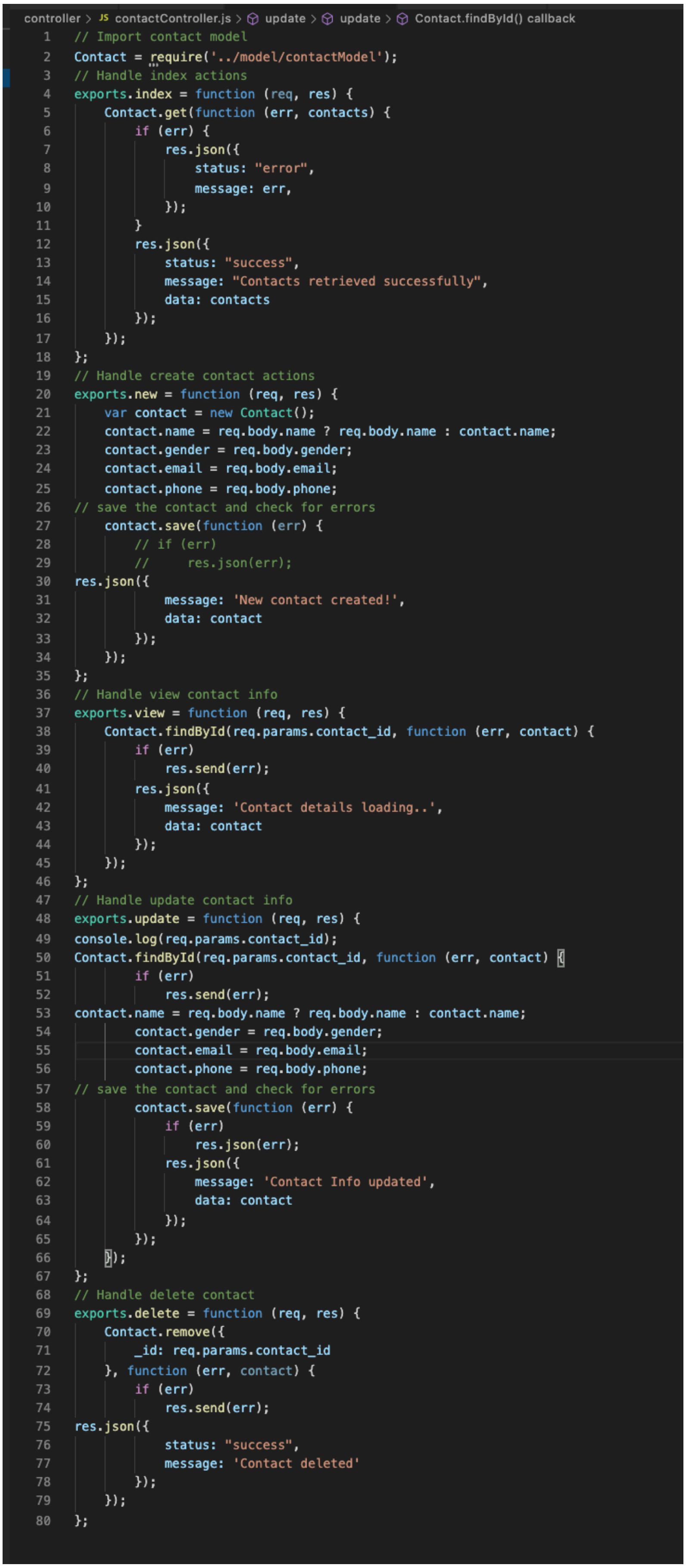
```javascript
var mongoose = require('mongoose');
// Setup schema
var contactSchema = mongoose.Schema({
    name: {
        type: String,
        required: true
    },
    email: {
        type: String,
        required: true
    },
    gender: String,
    phone: String,
    create_date: {
        type: Date,
        default: Date.now
    }
});
// Export Contact model
var Contact = module.exports = mongoose.model('contact', contactSchema);
module.exports.get = function (callback, limit) {
    Contact.find(callback).limit(limit);
}
```

contactModel.js



```dockerfile
# pull the Node.js Docker image
FROM node:alpine

# create the directory inside the container
WORKDIR /usr/app

# copy the package.json files from local machine to the workdir in container
COPY package*.json ./

# run npm install in our local machine
RUN npm install

# copy the generated modules and all other files to the container
COPY . .

EXPOSE 8080

# the command that starts our app
CMD ["node", "index.js"]
```

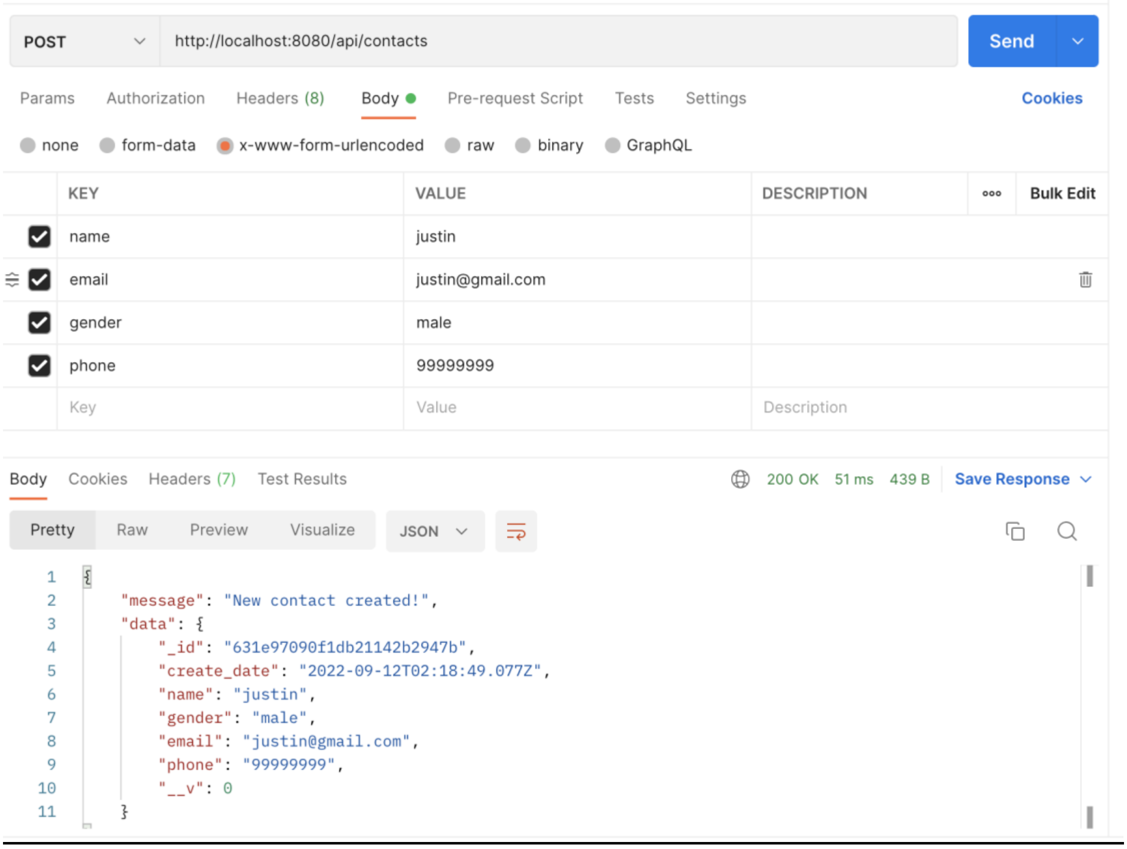
Dockerfile


controller > contactController.js > update > update > Contact.findById() callback

```
// Import contact model
Contact = require('../model/contactModel');
// Handle index actions
exports.index = function (req, res) {
    Contact.get(function (err, contacts) {
        if (err) {
            res.json({
                status: "error",
                message: err,
            });
        }
        res.json({
            status: "success",
            message: "Contacts retrieved successfully",
            data: contacts
        });
    });
};
// Handle create contact actions
exports.new = function (req, res) {
    var contact = new Contact();
    contact.name = req.body.name ? req.body.name : contact.name;
    contact.gender = req.body.gender;
    contact.email = req.body.email;
    contact.phone = req.body.phone;
// save the contact and check for errors
    contact.save(function (err) {
        // if (err)
        //     res.json(err);
res.json({
            message: 'New contact created!',
            data: contact
        });
    });
};
// Handle view contact info
exports.view = function (req, res) {
    Contact.findById(req.params.contact_id, function (err, contact) {
        if (err)
            res.send(err);
        res.json({
            message: 'Contact details loading..',
            data: contact
        });
    });
};
// Handle update contact info
exports.update = function (req, res) {
console.log(req.params.contact_id);
Contact.findById(req.params.contact_id, function (err, contact) {
        if (err)
            res.send(err);
contact.name = req.body.name ? req.body.name : contact.name;
        contact.gender = req.body.gender;
        contact.email = req.body.email;
        contact.phone = req.body.phone;
// save the contact and check for errors
        contact.save(function (err) {
            if (err)
                res.json(err);
            res.json({
                message: 'Contact Info updated',
                data: contact
            });
        });
    });
};
// Handle delete contact
exports.delete = function (req, res) {
    Contact.remove({
        _id: req.params.contact_id
    }, function (err, contact) {
        if (err)
            res.send(err);
res.json({
            status: "success",
            message: 'Contact deleted'
        });
    });
};
```

contactController.js

Successful API calls on localhost via postman:



POST http://localhost:8080/api/contacts

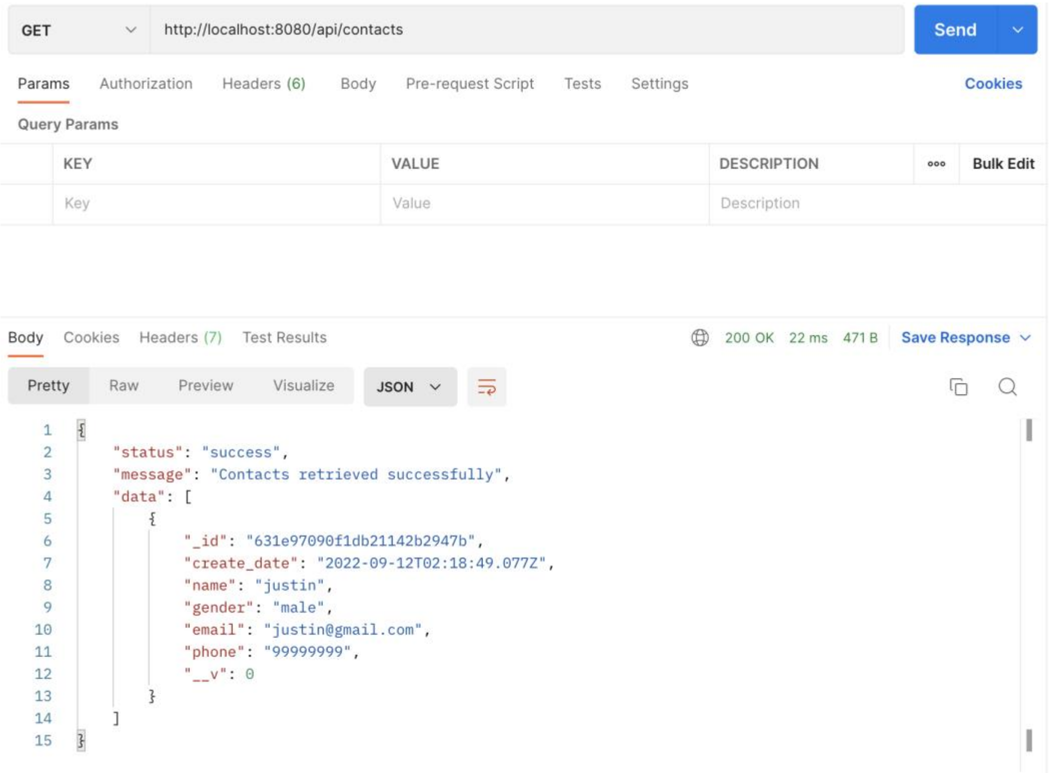
Body: ● x-www-form-urlencoded

	KEY	VALUE	DESCRIPTION
☑	name	justin	
☑	email	justin@gmail.com	
☑	gender	male	
☑	phone	99999999	
	Key	Value	Description

Body — 200 OK 51 ms 439 B — JSON

```
{
    "message": "New contact created!",
    "data": {
        "_id": "631e97090f1db21142b2947b",
        "create_date": "2022-09-12T02:18:49.077Z",
        "name": "justin",
        "gender": "male",
        "email": "justin@gmail.com",
        "phone": "99999999",
        "__v": 0
    }
```

POST request to insert contact



GET http://localhost:8080/api/contacts

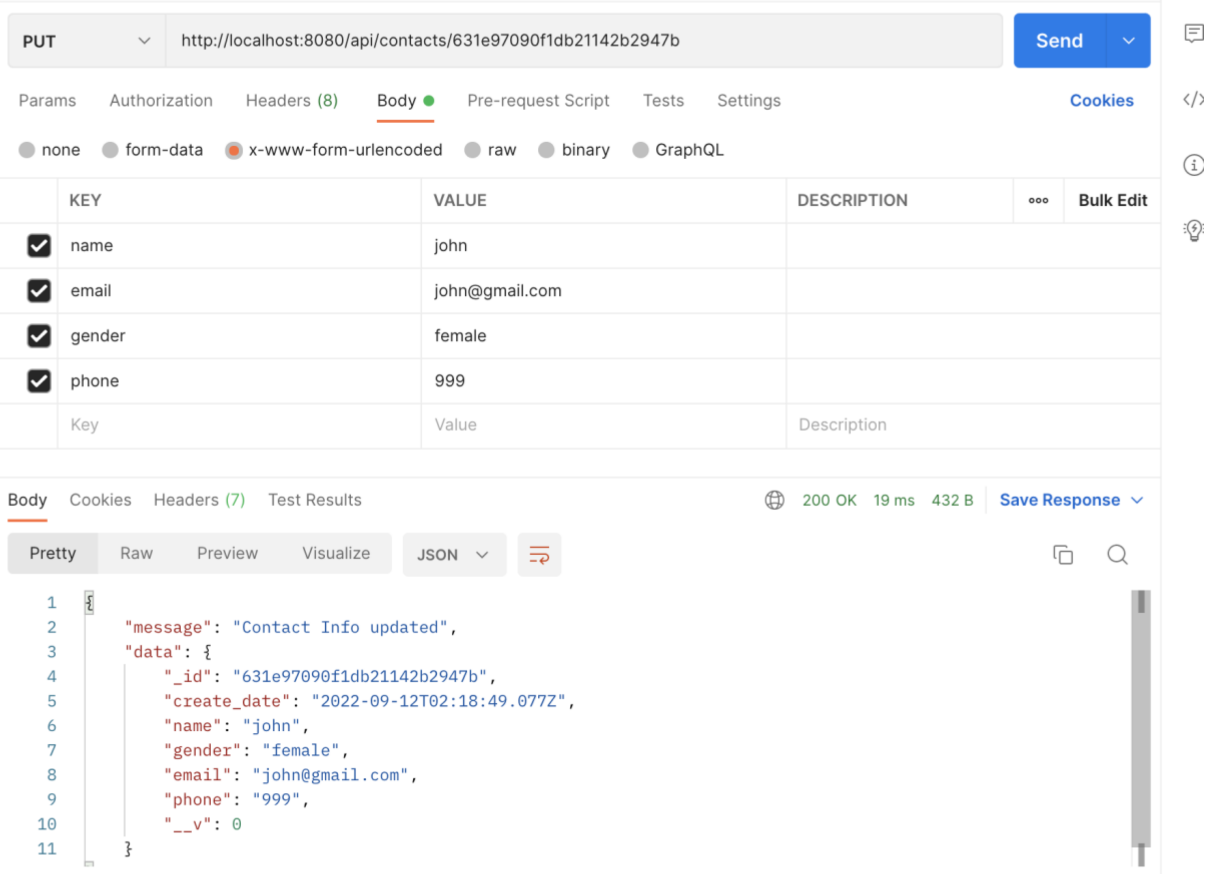
Query Params

KEY	VALUE	DESCRIPTION
Key	Value	Description

Body — 200 OK 22 ms 471 B — JSON

```
{
    "status": "success",
    "message": "Contacts retrieved successfully",
    "data": [
        {
            "_id": "631e97090f1db21142b2947b",
            "create_date": "2022-09-12T02:18:49.077Z",
            "name": "justin",
            "gender": "male",
            "email": "justin@gmail.com",
            "phone": "99999999",
            "__v": 0
        }
    ]
}
```

GET request to retrieve contacts



PUT http://localhost:8080/api/contacts/631e97090f1db21142b2947b

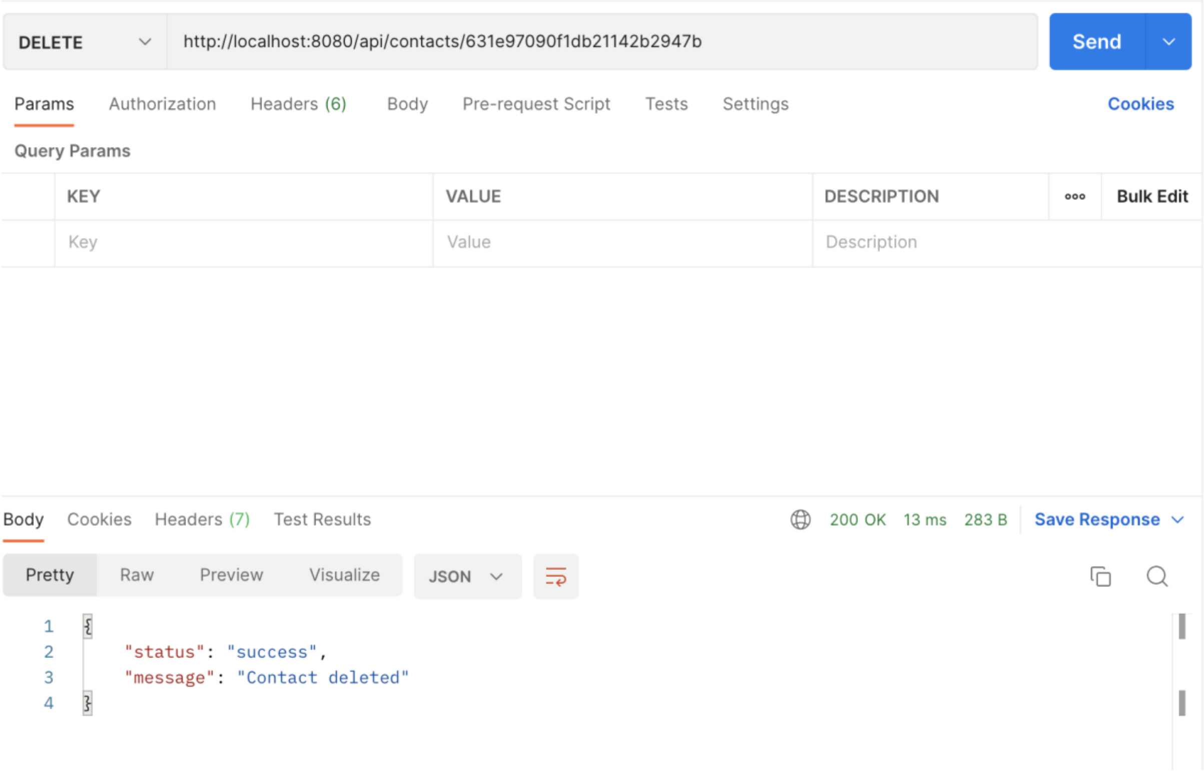
Body: x-www-form-urlencoded

	KEY	VALUE	DESCRIPTION
☑	name	john	
☑	email	john@gmail.com	
☑	gender	female	
☑	phone	999	
	Key	Value	Description

200 OK 19 ms 432 B

```
{
    "message": "Contact Info updated",
    "data": {
        "_id": "631e97090f1db21142b2947b",
        "create_date": "2022-09-12T02:18:49.077Z",
        "name": "john",
        "gender": "female",
        "email": "john@gmail.com",
        "phone": "999",
        "__v": 0
    }
```

PUT call to update contact info



DELETE http://localhost:8080/api/contacts/631e97090f1db21142b2947b

Query Params

KEY	VALUE	DESCRIPTION
Key	Value	Description

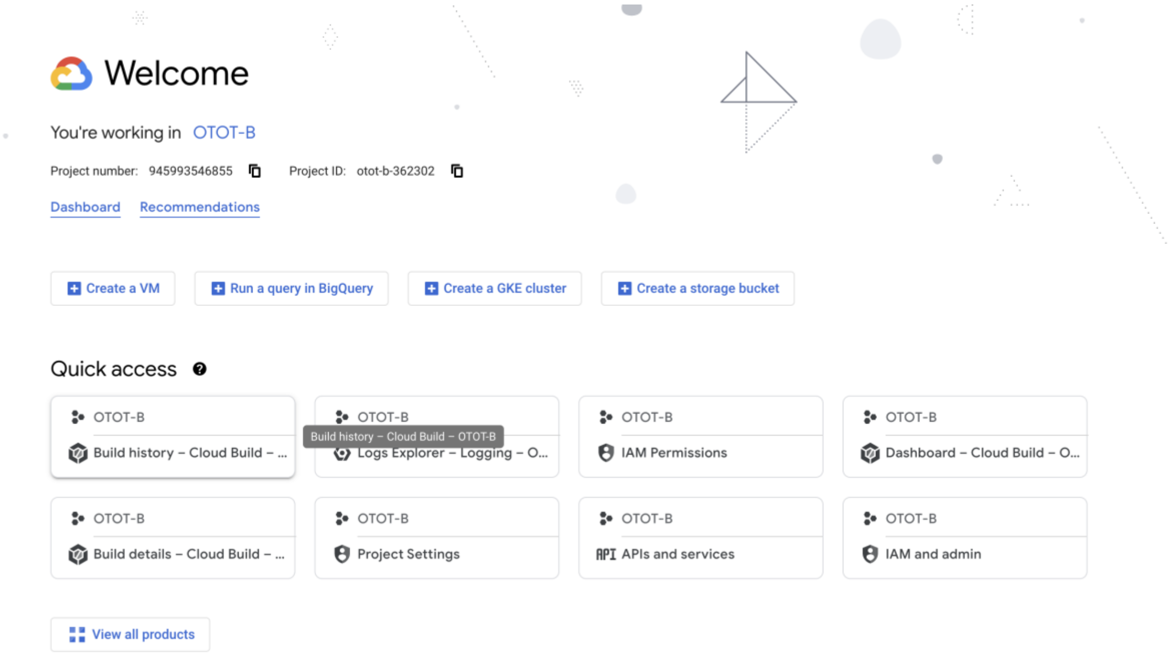
200 OK 13 ms 283 B

```
{
    "status": "success",
    "message": "Contact deleted"
}
```

DELETE call to delete contact from database

<u>Deployment to GCP:</u>

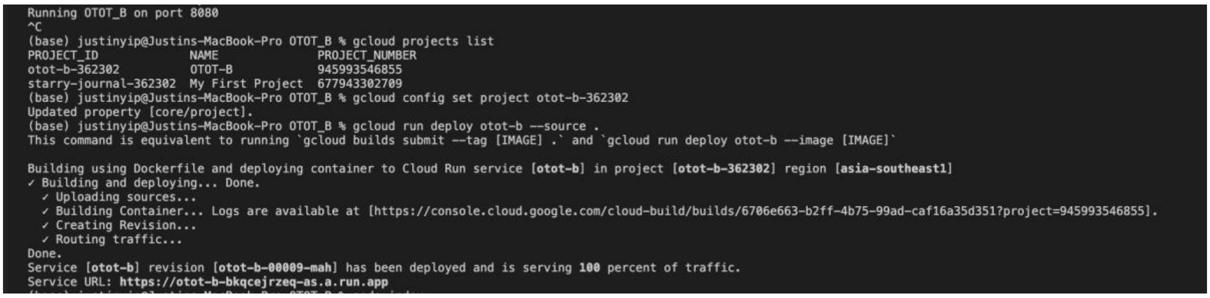
1. Run the following commands:
   - `gcloud projects list` to get project name
   - `gcloud set project <project name>
   - `gcloud config set run/region asia-southeast1`
   - `gcloud run deploy <service-name> --source .`



Welcome

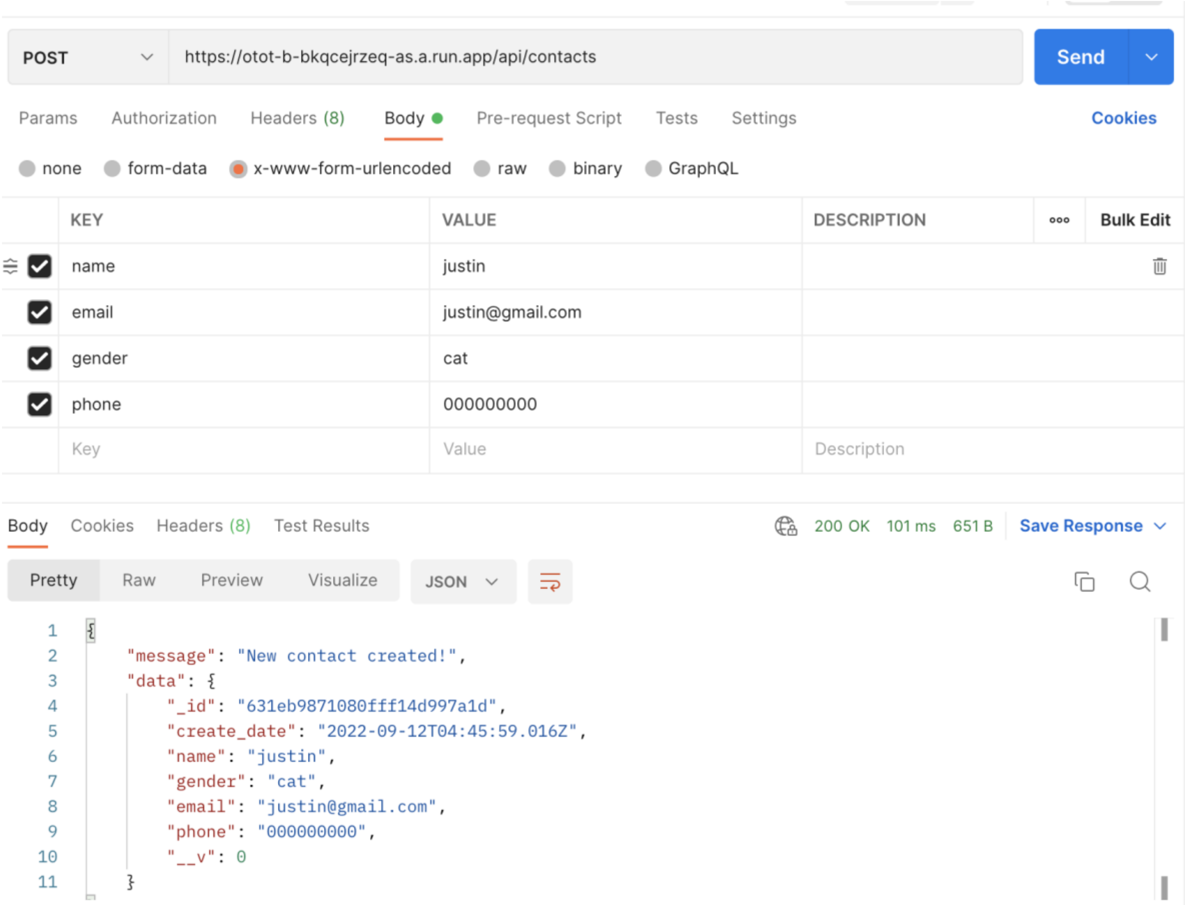
You're working in OTOT-B

Project number: 945993546855 Project ID: otot-b-362302



```
Running OTOT_B on port 8080
^C
(base) justinyip@Justins-MacBook-Pro OTOT_B % gcloud projects list
PROJECT_ID            NAME              PROJECT_NUMBER
otot-b-362302         OTOT-B            945993546855
starry-journal-362302  My First Project  677943302709
(base) justinyip@Justins-MacBook-Pro OTOT_B % gcloud config set project otot-b-362302
Updated property [core/project].
(base) justinyip@Justins-MacBook-Pro OTOT_B % gcloud run deploy otot-b --source .
This command is equivalent to running `gcloud builds submit --tag [IMAGE] .` and `gcloud run deploy otot-b --image [IMAGE]`

Building using Dockerfile and deploying container to Cloud Run service [otot-b] in project [otot-b-362302] region [asia-southeast1]
✓ Building and deploying... Done.
  ✓ Uploading sources...
  ✓ Building Container... Logs are available at [https://console.cloud.google.com/cloud-build/builds/6706e663-b2ff-4b75-99ad-caf16a35d351?project=945993546855].
  ✓ Creating Revision...
  ✓ Routing traffic...
Done.
Service [otot-b] revision [otot-b-00009-mah] has been deployed and is serving 100 percent of traffic.
Service URL: https://otot-b-bkqcejrzeq-as.a.run.app
```

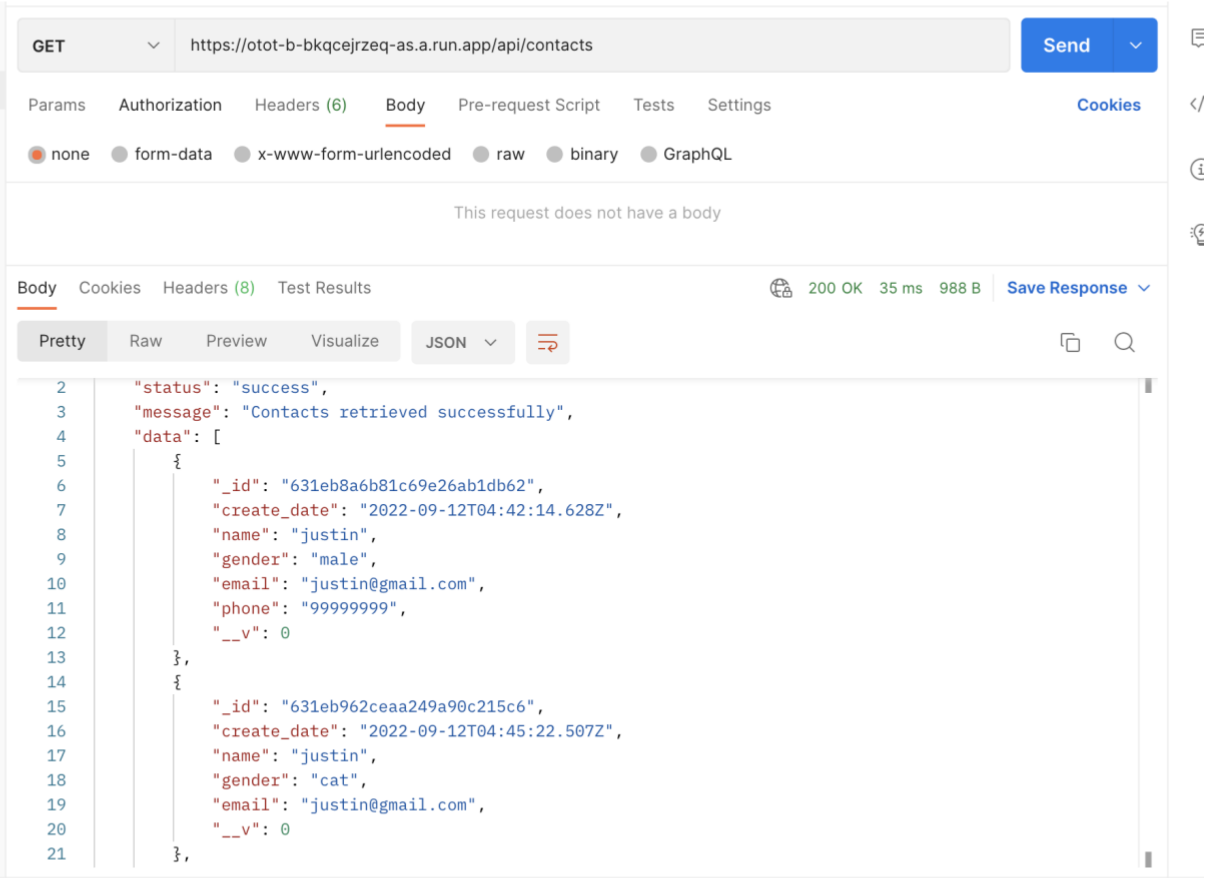
Successful API calls on deployment via postman:



	KEY	VALUE	DESCRIPTION
☑	name	justin	
☑	email	justin@gmail.com	
☑	gender	cat	
☑	phone	000000000	
	Key	Value	Description

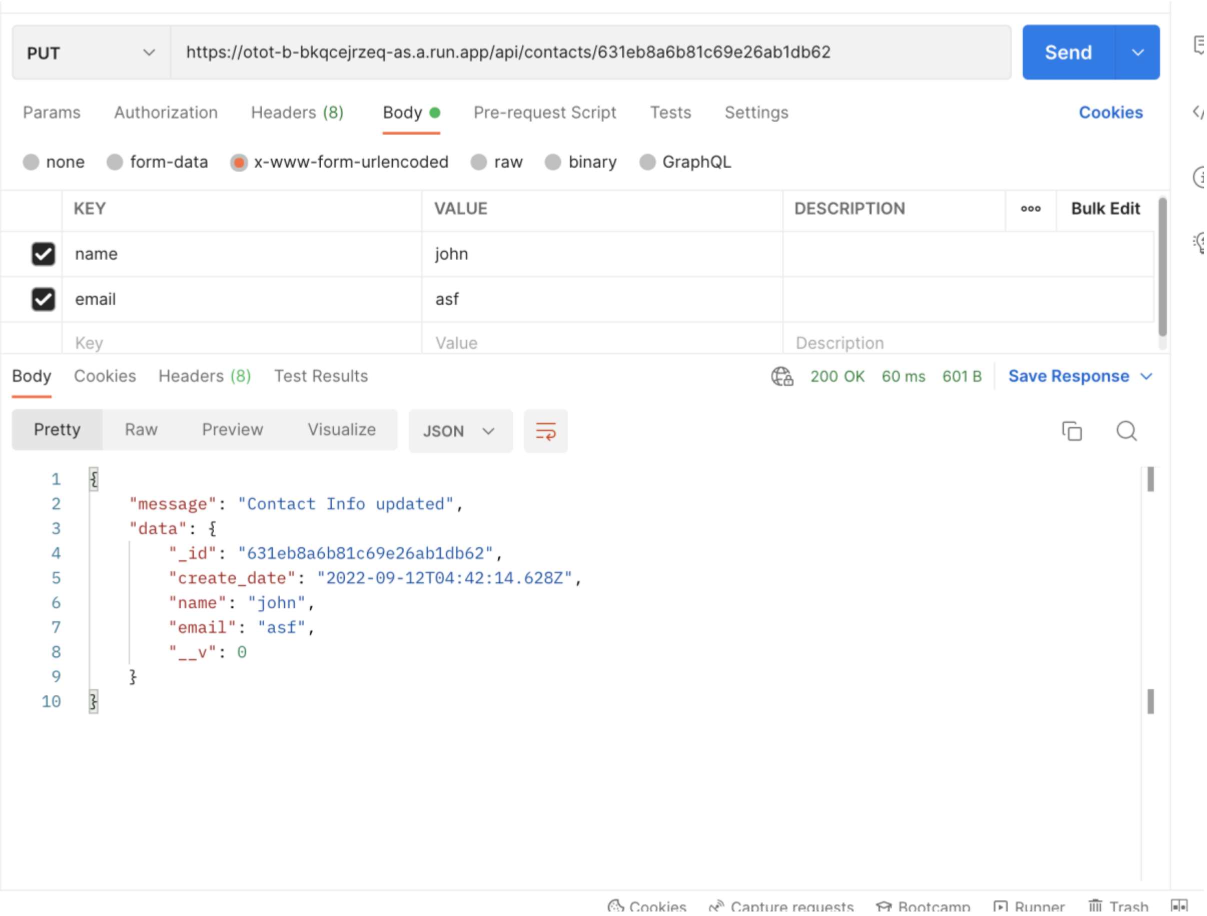
```
{
    "message": "New contact created!",
    "data": {
        "_id": "631eb9871080fff14d997a1d",
        "create_date": "2022-09-12T04:45:59.016Z",
        "name": "justin",
        "gender": "cat",
        "email": "justin@gmail.com",
        "phone": "000000000",
        "__v": 0
    }
```

POST



```
    "status": "success",
    "message": "Contacts retrieved successfully",
    "data": [
        {
            "_id": "631eb8a6b81c69e26ab1db62",
            "create_date": "2022-09-12T04:42:14.628Z",
            "name": "justin",
            "gender": "male",
            "email": "justin@gmail.com",
            "phone": "99999999",
            "__v": 0
        },
        {
            "_id": "631eb962ceaa249a90c215c6",
            "create_date": "2022-09-12T04:45:22.507Z",
            "name": "justin",
            "gender": "cat",
            "email": "justin@gmail.com",
            "__v": 0
        },
```

GET



PUT https://otot-b-bkqcejrzeq-as.a.run.app/api/contacts/631eb8a6b81c69e26ab1db62

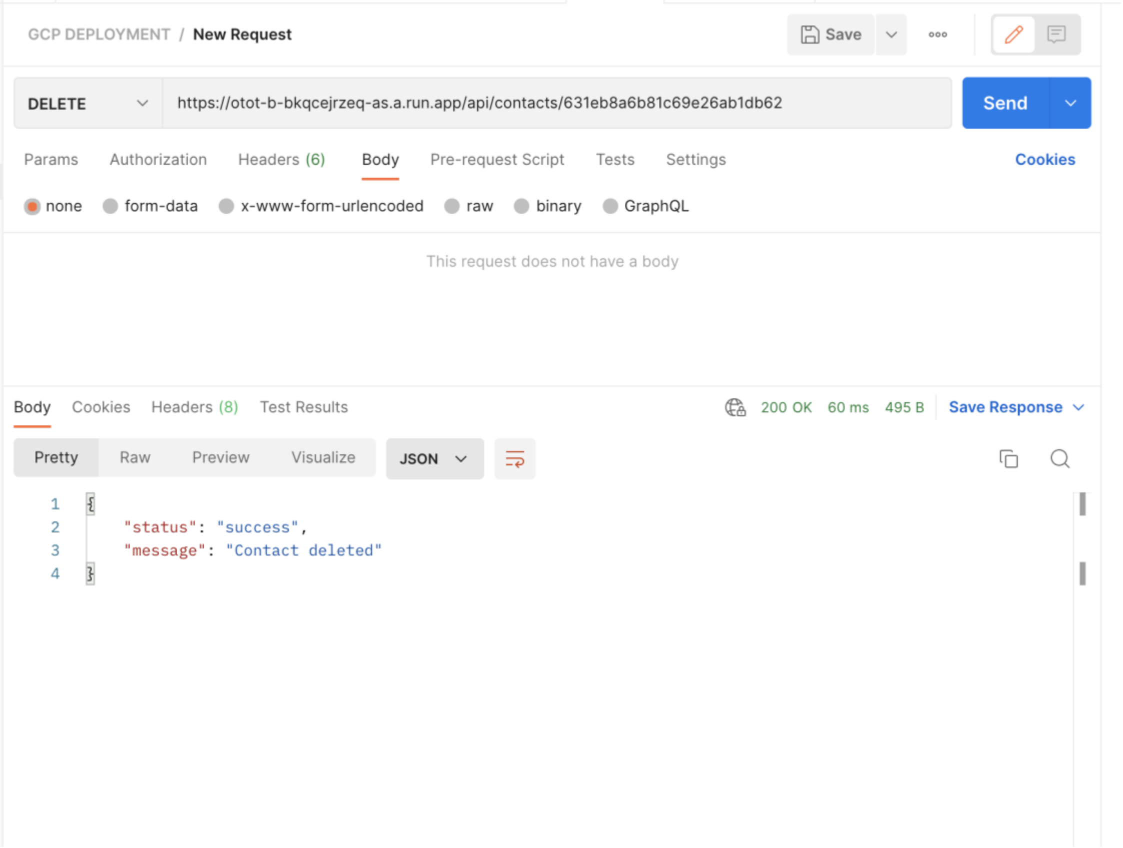
○ none ○ form-data ● x-www-form-urlencoded ○ raw ○ binary ○ GraphQL

	KEY	VALUE	DESCRIPTION
☑	name	john	
☑	email	asf	
	Key	Value	Description

Body — 200 OK 60 ms 601 B — JSON

```
{
    "message": "Contact Info updated",
    "data": {
        "_id": "631eb8a6b81c69e26ab1db62",
        "create_date": "2022-09-12T04:42:14.628Z",
        "name": "john",
        "email": "asf",
        "__v": 0
    }
}
```

PUT



GCP DEPLOYMENT / **New Request**

DELETE https://otot-b-bkqcejrzeq-as.a.run.app/api/contacts/631eb8a6b81c69e26ab1db62

● none ○ form-data ○ x-www-form-urlencoded ○ raw ○ binary ○ GraphQL

This request does not have a body

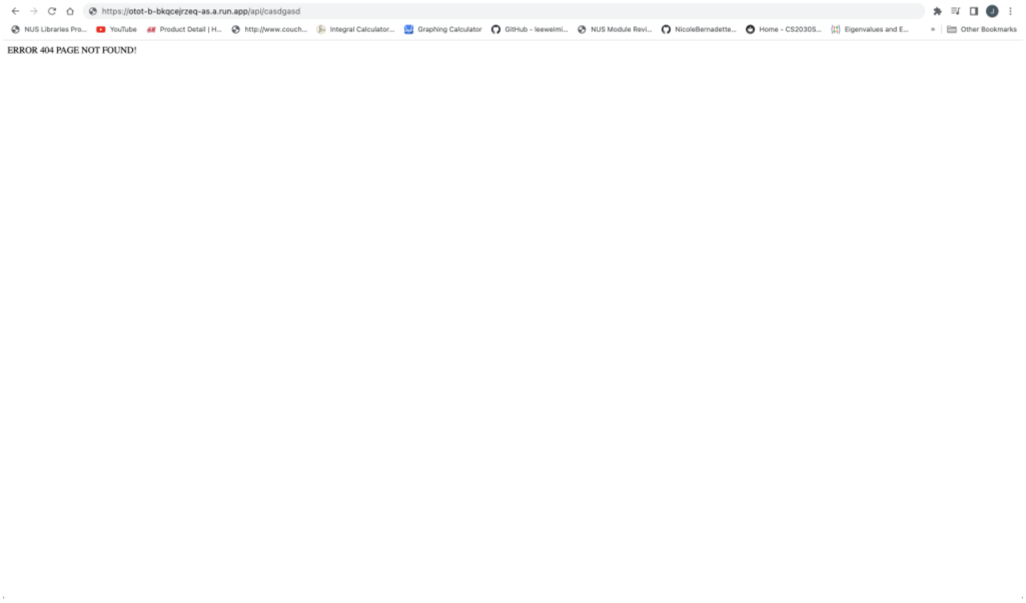
Body — 200 OK 60 ms 495 B — JSON

```
{
    "status": "success",
    "message": "Contact deleted"
}
```

DELETE

Common edge cases and error-resiliency:

1.

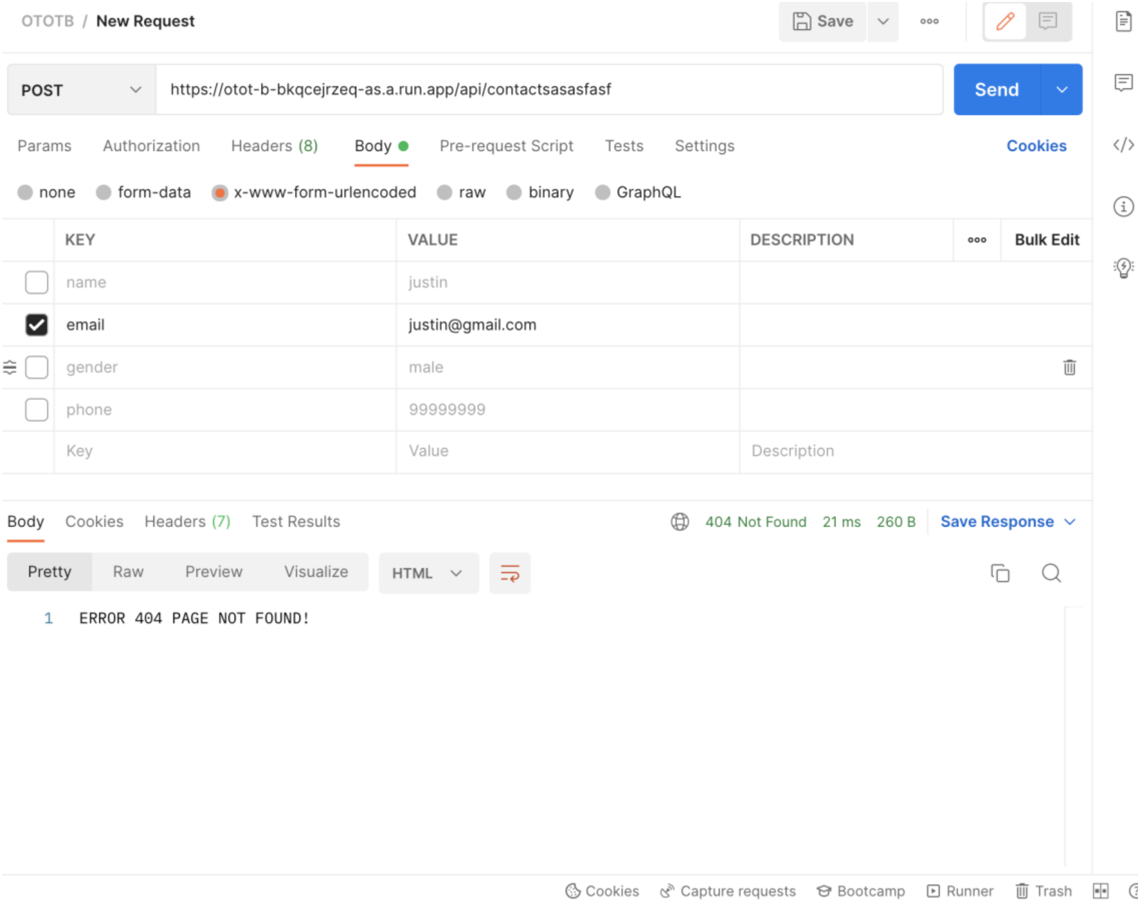


https://otot-b-bkqcejrzeq-as.a.run.app/api/casdgasd

ERROR 404 PAGE NOT FOUND!

Return error 404 when user tries to access (GET) an invalid url.

2.



POST https://otot-b-bkqcejrzeq-as.a.run.app/api/contactsasasfasf

○ none ○ form-data ● x-www-form-urlencoded ○ raw ○ binary ○ GraphQL

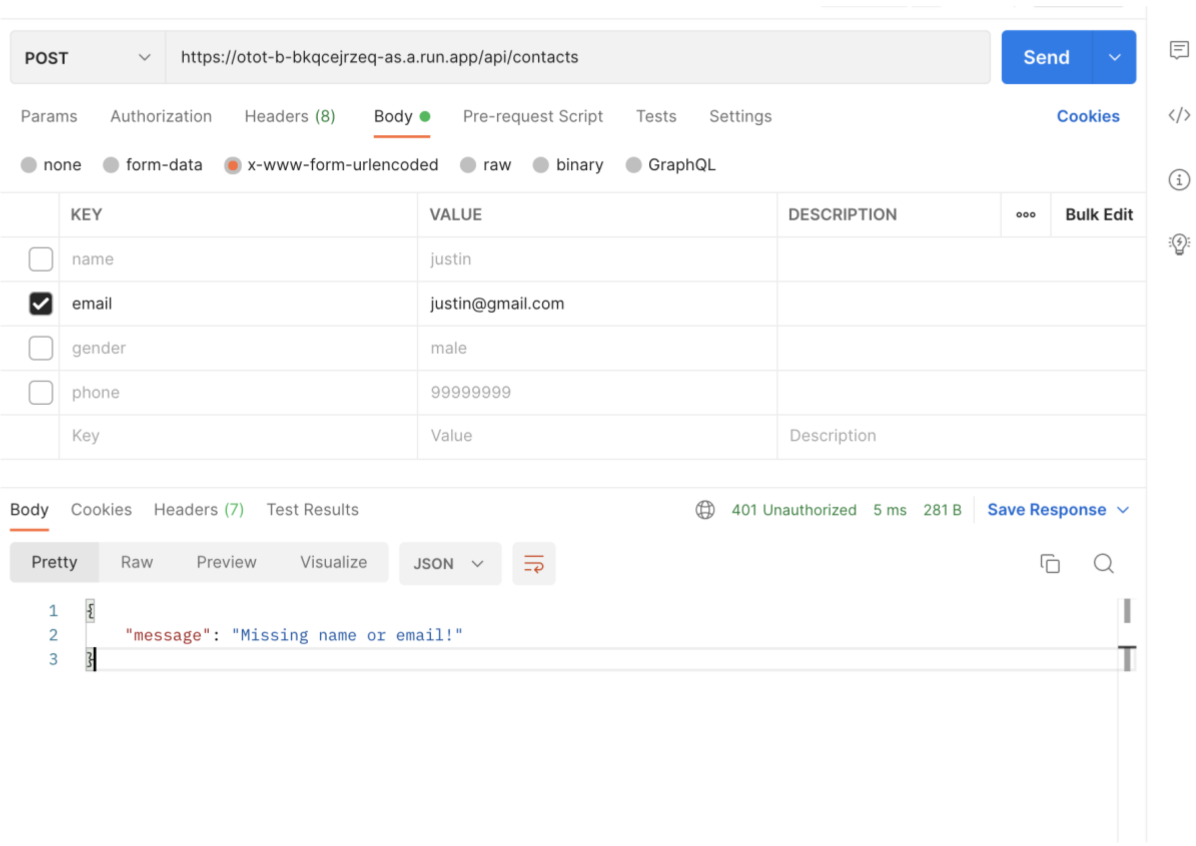
	KEY	VALUE	DESCRIPTION
□	name	justin	
☑	email	justin@gmail.com	
□	gender	male	
□	phone	99999999	
	Key	Value	Description

404 Not Found 21 ms 260 B

```
ERROR 404 PAGE NOT FOUND!
```

Return error 404 when user tries to POST to invalid url

3.



POST https://otot-b-bkqcejrzeq-as.a.run.app/api/contacts

○ none ○ form-data ● x-www-form-urlencoded ○ raw ○ binary ○ GraphQL

	KEY	VALUE	DESCRIPTION
□	name	justin	
☑	email	justin@gmail.com	
□	gender	male	
□	phone	99999999	
	Key	Value	Description

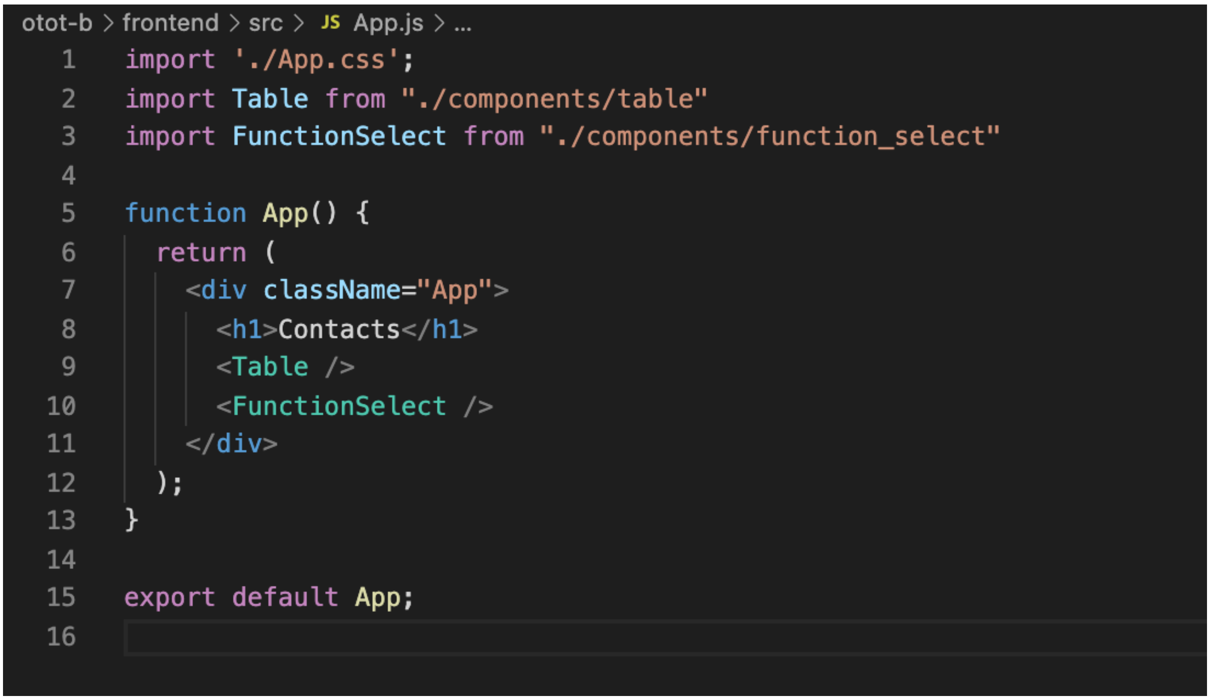
Body — 401 Unauthorized 5 ms 281 B — JSON

```
{
    "message": "Missing name or email!"
}
```

Return error message when user tries to insert contact without required name/email fields.

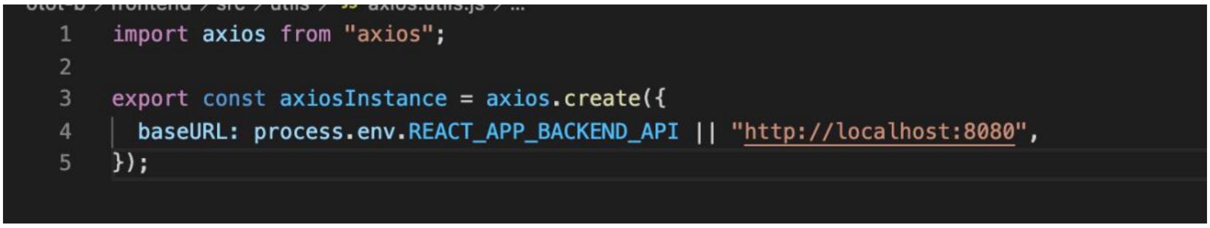
B3:

1. Run `*npm I create-react-app*` then run `*npx create-react-app otot-b*`
2. Edit and add the following js files



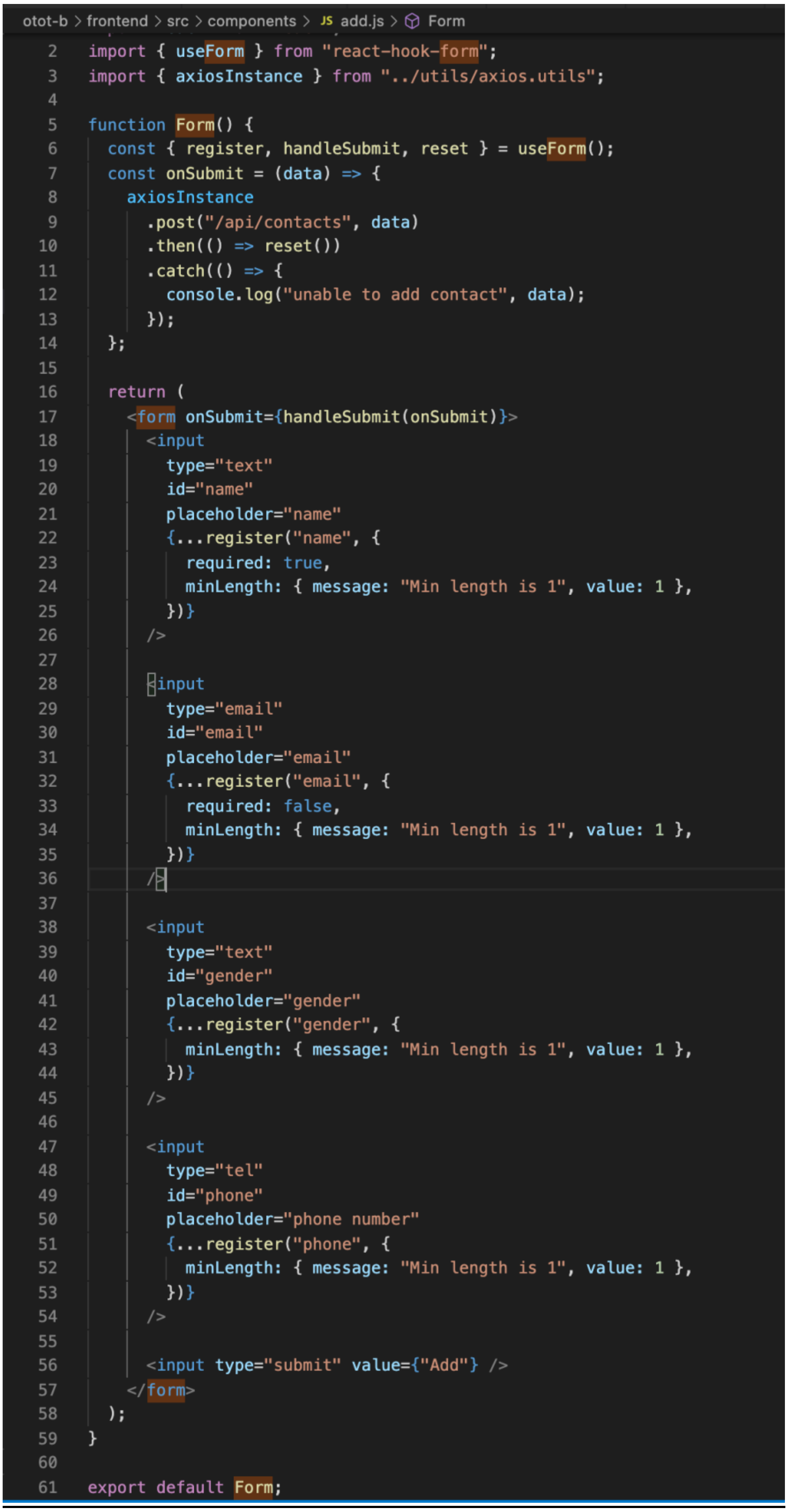
```
import './App.css';
import Table from "./components/table"
import FunctionSelect from "./components/function_select"

function App() {
  return (
    <div className="App">
      <h1>Contacts</h1>
      <Table />
      <FunctionSelect />
    </div>
  );
}

export default App;

```

App.js



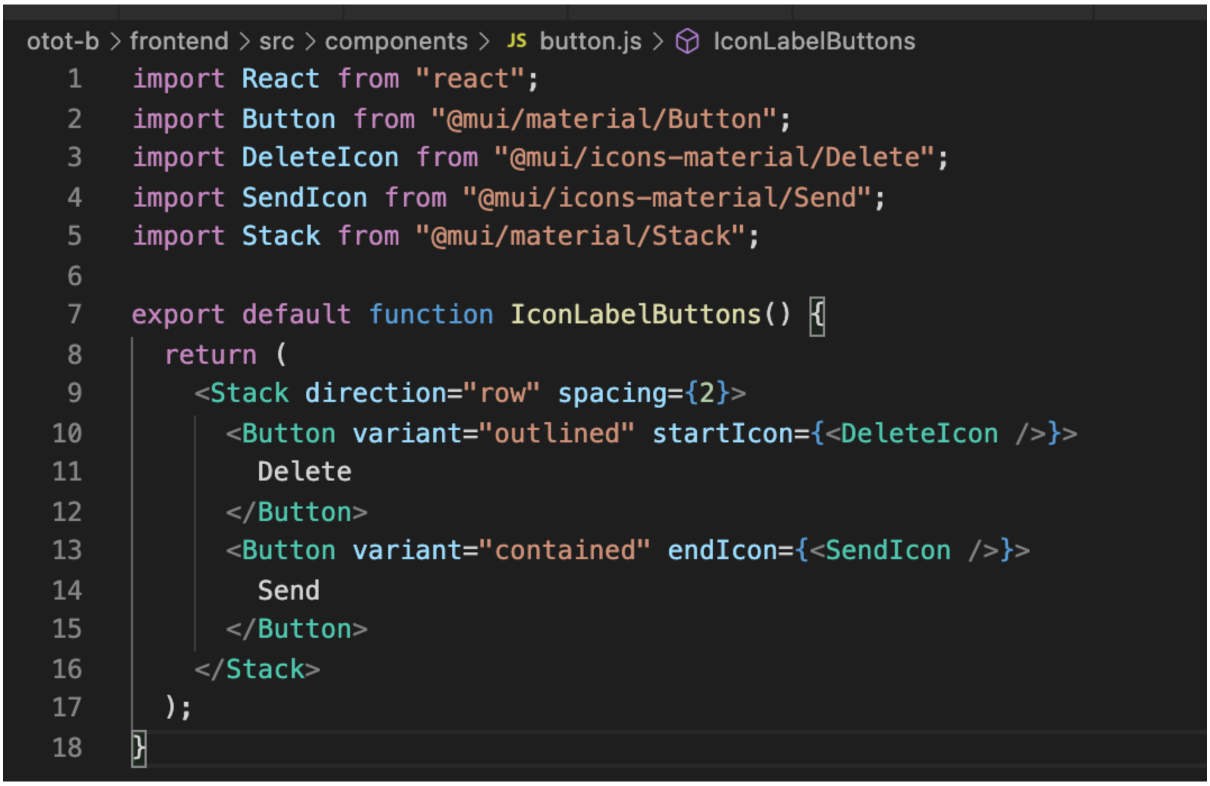
```
import axios from "axios";

export const axiosInstance = axios.create({
  baseURL: process.env.REACT_APP_BACKEND_API || "http://localhost:8080",
});
```

utils/axios.utils.js



```
import { useForm } from "react-hook-form";
import { axiosInstance } from "../utils/axios.utils";

function Form() {
  const { register, handleSubmit, reset } = useForm();
  const onSubmit = (data) => {
    axiosInstance
      .post("/api/contacts", data)
      .then(() => reset())
      .catch(() => {
        console.log("unable to add contact", data);
      });
  };

  return (
    <form onSubmit={handleSubmit(onSubmit)}>
      <input
        type="text"
        id="name"
        placeholder="name"
        {...register("name", {
          required: true,
          minLength: { message: "Min length is 1", value: 1 },
        })}
      />

      <input
        type="email"
        id="email"
        placeholder="email"
        {...register("email", {
          required: false,
          minLength: { message: "Min length is 1", value: 1 },
        })}
      />

      <input
        type="text"
        id="gender"
        placeholder="gender"
        {...register("gender", {
          minLength: { message: "Min length is 1", value: 1 },
        })}
      />

      <input
        type="tel"
        id="phone"
        placeholder="phone number"
        {...register("phone", {
          minLength: { message: "Min length is 1", value: 1 },
        })}
      />

      <input type="submit" value={"Add"} />
    </form>
  );
}

export default Form;
```

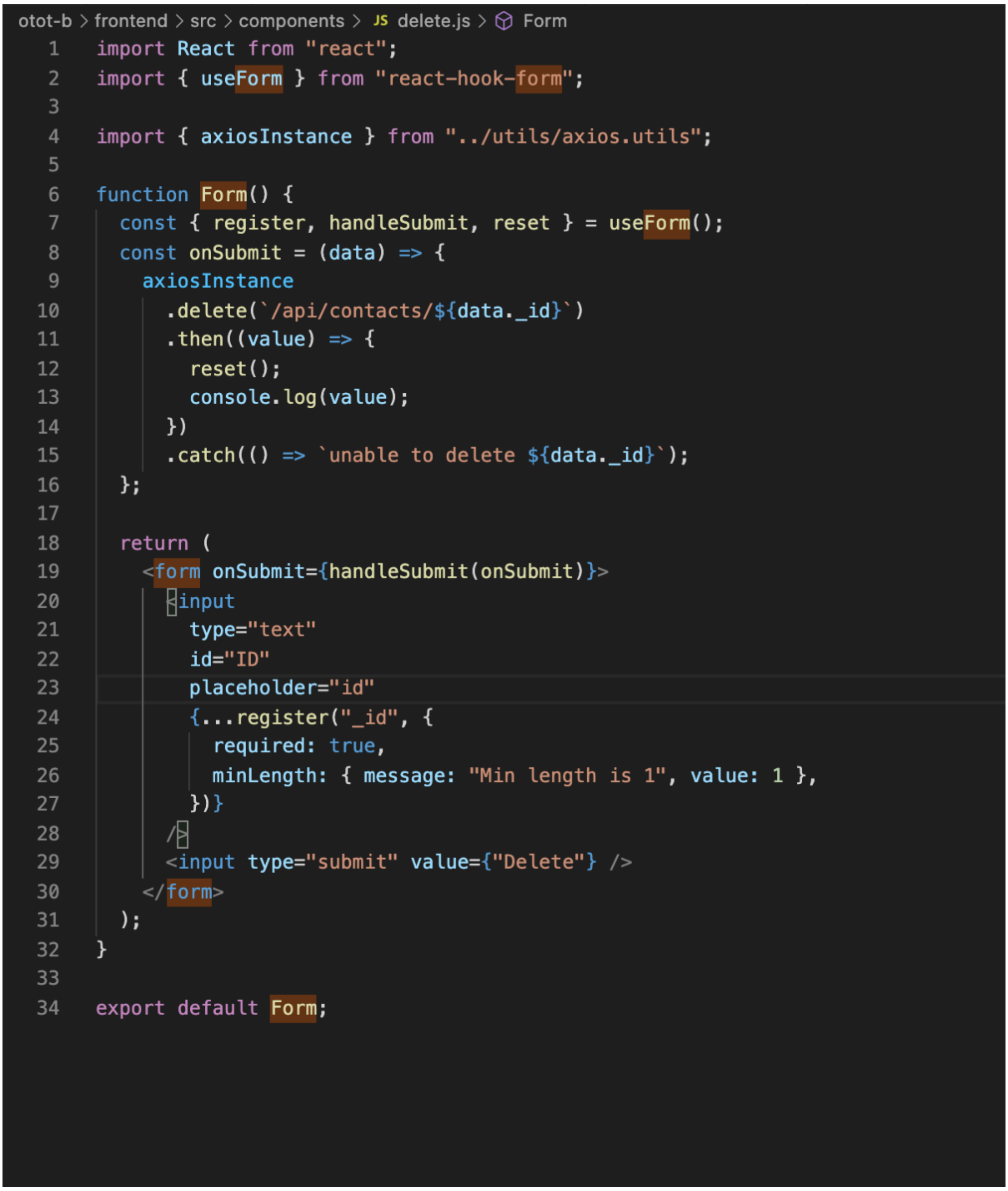
components/add.js



otot-b > frontend > src > components > JS button.js > IconLabelButtons

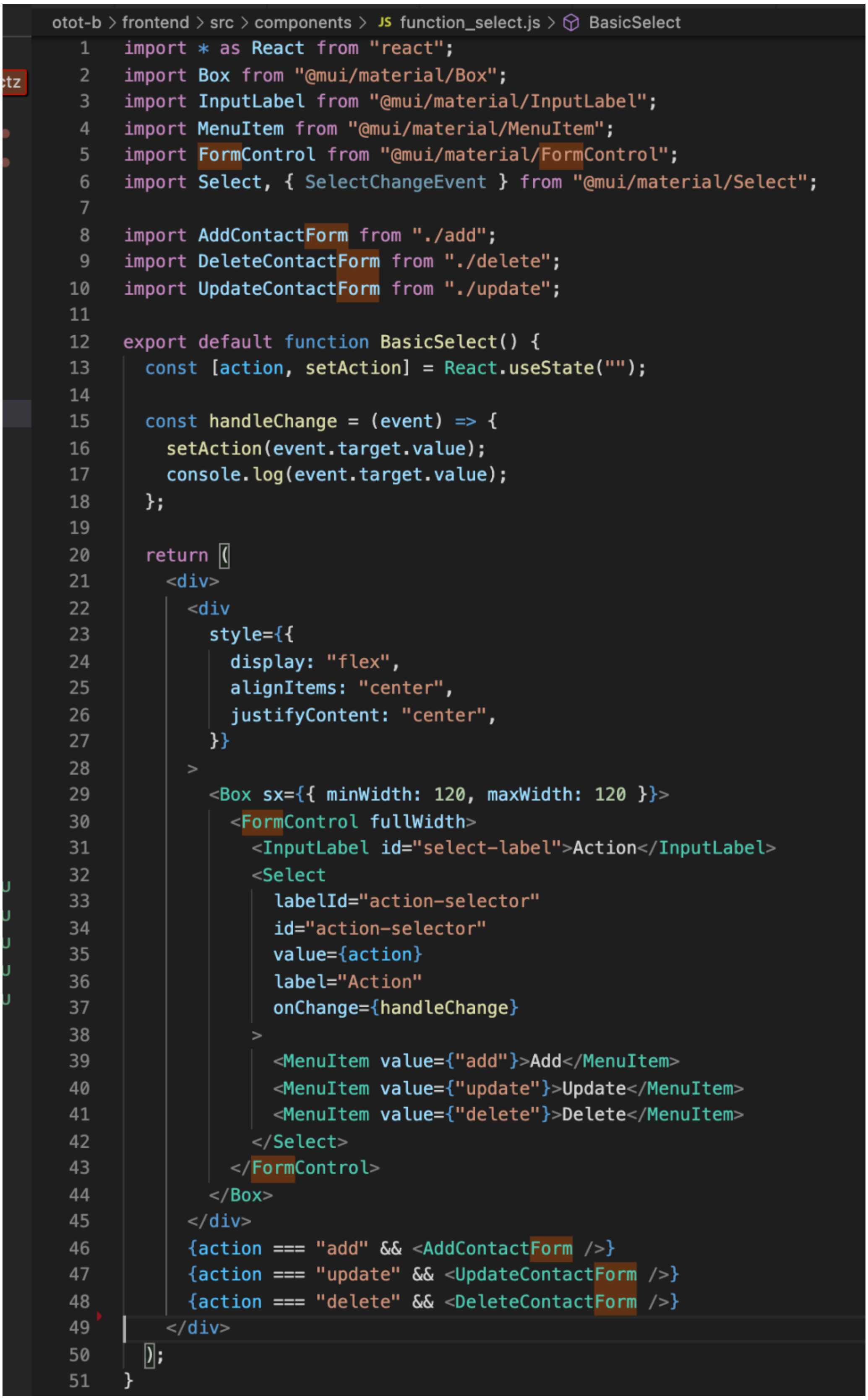
```
import React from "react";
import Button from "@mui/material/Button";
import DeleteIcon from "@mui/icons-material/Delete";
import SendIcon from "@mui/icons-material/Send";
import Stack from "@mui/material/Stack";

export default function IconLabelButtons() {
  return (
    <Stack direction="row" spacing={2}>
      <Button variant="outlined" startIcon={<DeleteIcon />}>
        Delete
      </Button>
      <Button variant="contained" endIcon={<SendIcon />}>
        Send
      </Button>
    </Stack>
  );
}
```

components/button.js



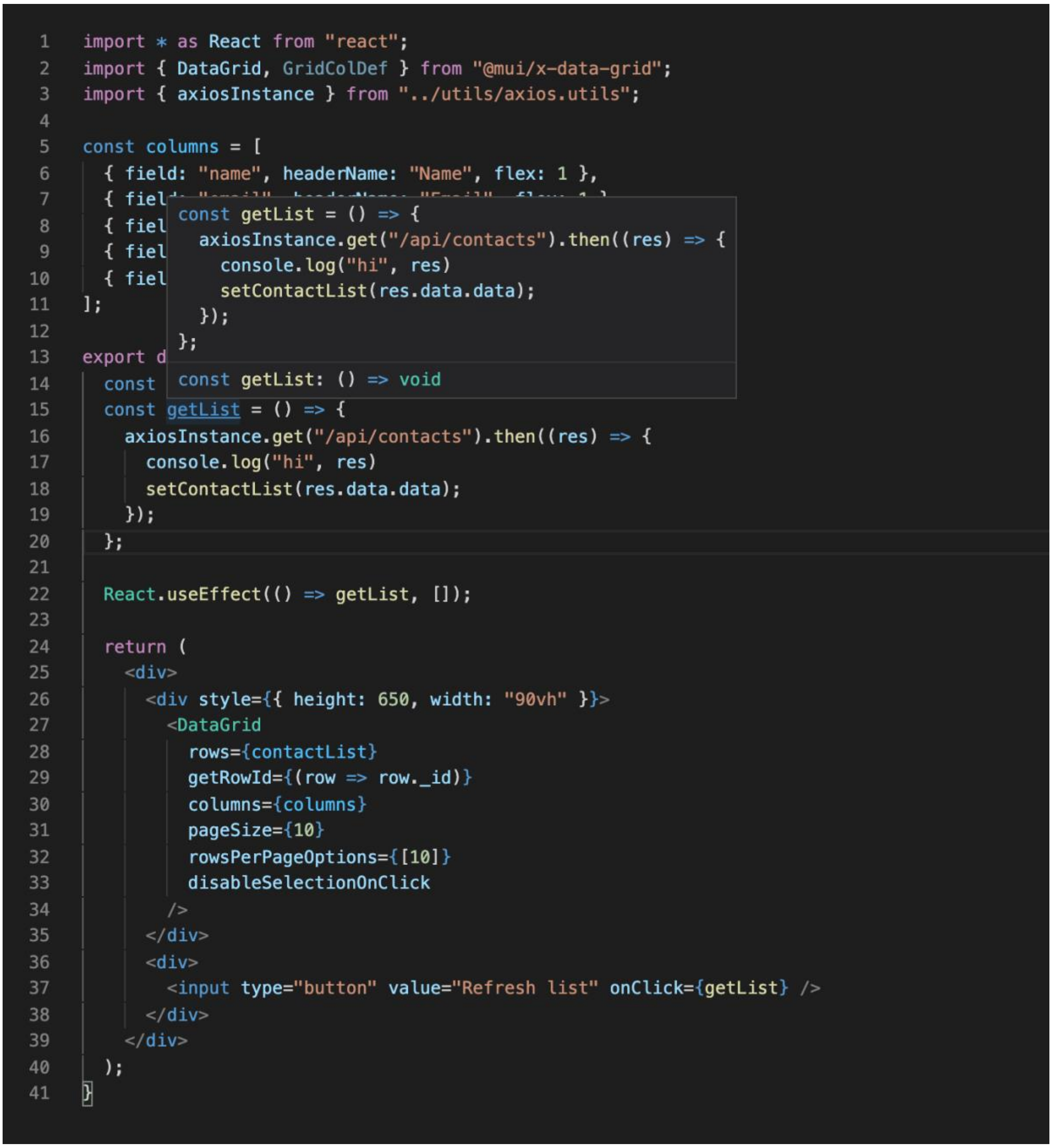
```
import React from "react";
import { useForm } from "react-hook-form";

import { axiosInstance } from "../utils/axios.utils";

function Form() {
  const { register, handleSubmit, reset } = useForm();
  const onSubmit = (data) => {
    axiosInstance
      .delete(`/api/contacts/${data._id}`)
      .then((value) => {
        reset();
        console.log(value);
      })
      .catch(() => `unable to delete ${data._id}`);
  };

  return (
    <form onSubmit={handleSubmit(onSubmit)}>
      <input
        type="text"
        id="ID"
        placeholder="id"
        {...register("_id", {
          required: true,
          minLength: { message: "Min length is 1", value: 1 },
        })}
      />
      <input type="submit" value={"Delete"} />
    </form>
  );
}

export default Form;
```

components/delete.js


otot-b > frontend > src > components > JS function_select.js > BasicSelect

```
import * as React from "react";
import Box from "@mui/material/Box";
import InputLabel from "@mui/material/InputLabel";
import MenuItem from "@mui/material/MenuItem";
import FormControl from "@mui/material/FormControl";
import Select, { SelectChangeEvent } from "@mui/material/Select";

import AddContactForm from "./add";
import DeleteContactForm from "./delete";
import UpdateContactForm from "./update";

export default function BasicSelect() {
  const [action, setAction] = React.useState("");

  const handleChange = (event) => {
    setAction(event.target.value);
    console.log(event.target.value);
  };

  return (
    <div>
      <div
        style={{
          display: "flex",
          alignItems: "center",
          justifyContent: "center",
        }}
      >
        <Box sx={{ minWidth: 120, maxWidth: 120 }}>
          <FormControl fullWidth>
            <InputLabel id="select-label">Action</InputLabel>
            <Select
              labelId="action-selector"
              id="action-selector"
              value={action}
              label="Action"
              onChange={handleChange}
            >
              <MenuItem value={"add"}>Add</MenuItem>
              <MenuItem value={"update"}>Update</MenuItem>
              <MenuItem value={"delete"}>Delete</MenuItem>
            </Select>
          </FormControl>
        </Box>
      </div>
      {action === "add" && <AddContactForm />}
      {action === "update" && <UpdateContactForm />}
      {action === "delete" && <DeleteContactForm />}
    </div>
  );
}
```

components/function_select.js

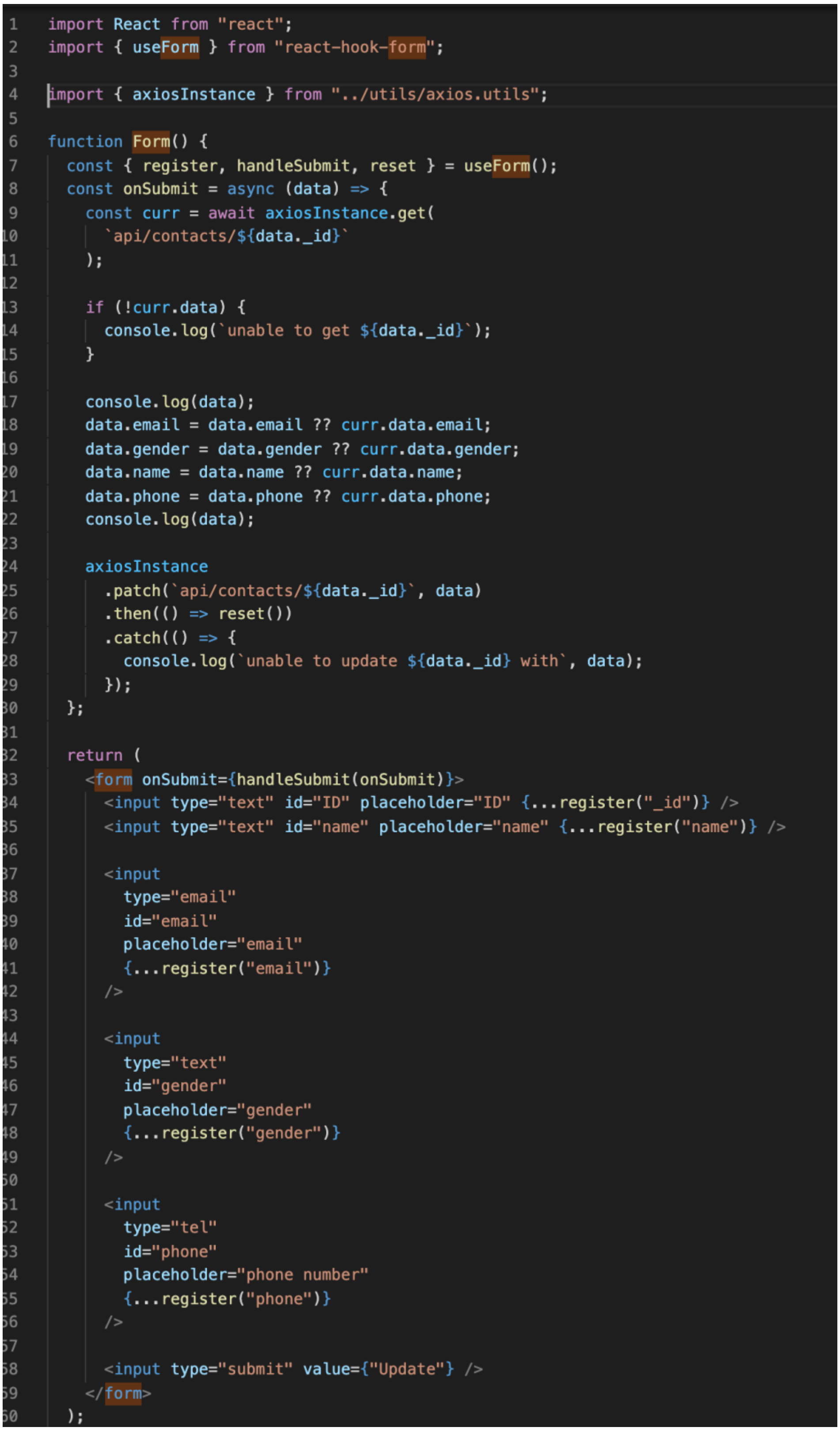


```
import * as React from "react";
import { DataGrid, GridColDef } from "@mui/x-data-grid";
import { axiosInstance } from "../utils/axios.utils";

const columns = [
  { field: "name", headerName: "Name", flex: 1 },
  { fiel
  { fiel
  { fiel
  { fiel
];

export d
  const
  const getList = () => {
    axiosInstance.get("/api/contacts").then((res) => {
      console.log("hi", res)
      setContactList(res.data.data);
    });
  };

  React.useEffect(() => getList, []);

  return (
    <div>
      <div style={{ height: 650, width: "90vh" }}>
        <DataGrid
          rows={contactList}
          getRowId={(row => row._id)}
          columns={columns}
          pageSize={10}
          rowsPerPageOptions={[10]}
          disableSelectionOnClick
        />
      </div>
      <div>
        <input type="button" value="Refresh list" onClick={getList} />
      </div>
    </div>
  );
}
```

```
const getList = () => {
  axiosInstance.get("/api/contacts").then((res) => {
    console.log("hi", res)
    setContactList(res.data.data);
  });
};
```

```
const getList: () => void
```

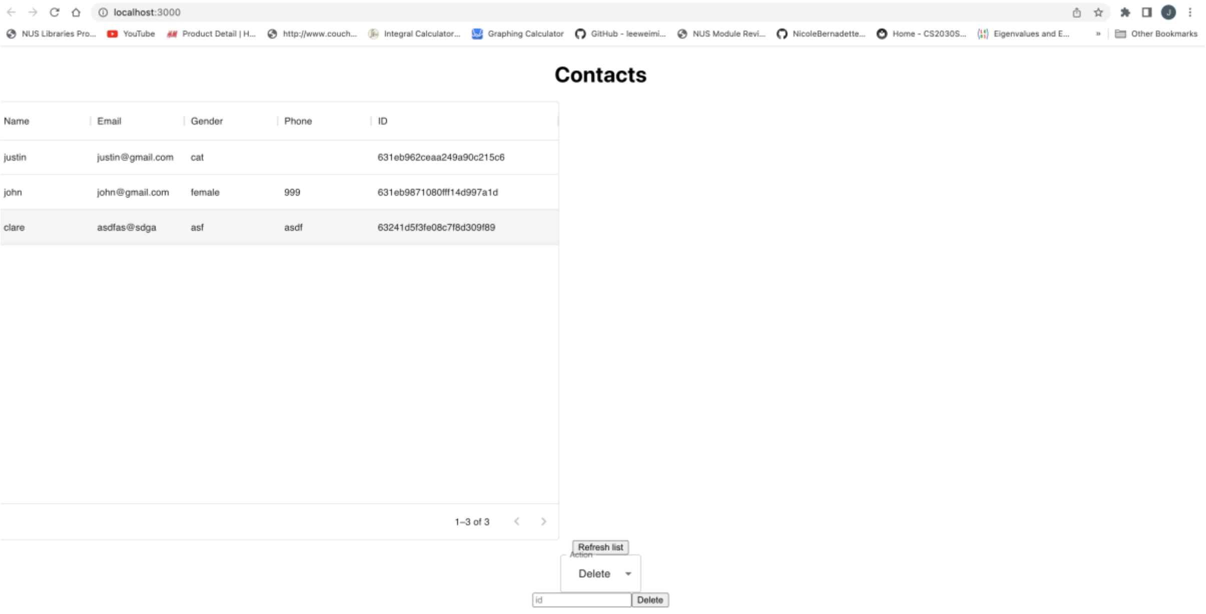
components/table.js



```
import React from "react";
import { useForm } from "react-hook-form";

import { axiosInstance } from "../utils/axios.utils";

function Form() {
  const { register, handleSubmit, reset } = useForm();
  const onSubmit = async (data) => {
    const curr = await axiosInstance.get(
      `api/contacts/${data._id}`
    );

    if (!curr.data) {
      console.log(`unable to get ${data._id}`);
    }

    console.log(data);
    data.email = data.email ?? curr.data.email;
    data.gender = data.gender ?? curr.data.gender;
    data.name = data.name ?? curr.data.name;
    data.phone = data.phone ?? curr.data.phone;
    console.log(data);

    axiosInstance
      .patch(`api/contacts/${data._id}`, data)
      .then(() => reset())
      .catch(() => {
        console.log(`unable to update ${data._id} with`, data);
      });
  };

  return (
    <form onSubmit={handleSubmit(onSubmit)}>
      <input type="text" id="ID" placeholder="ID" {...register("_id")} />
      <input type="text" id="name" placeholder="name" {...register("name")} />

      <input
        type="email"
        id="email"
        placeholder="email"
        {...register("email")}
      />

      <input
        type="text"
        id="gender"
        placeholder="gender"
        {...register("gender")}
      />

      <input
        type="tel"
        id="phone"
        placeholder="phone number"
        {...register("phone")}
      />

      <input type="submit" value={"Update"} />
    </form>
  );
```

components/update.js

3. Start up the backend and frontend with npm start.
4. This page should be visible:

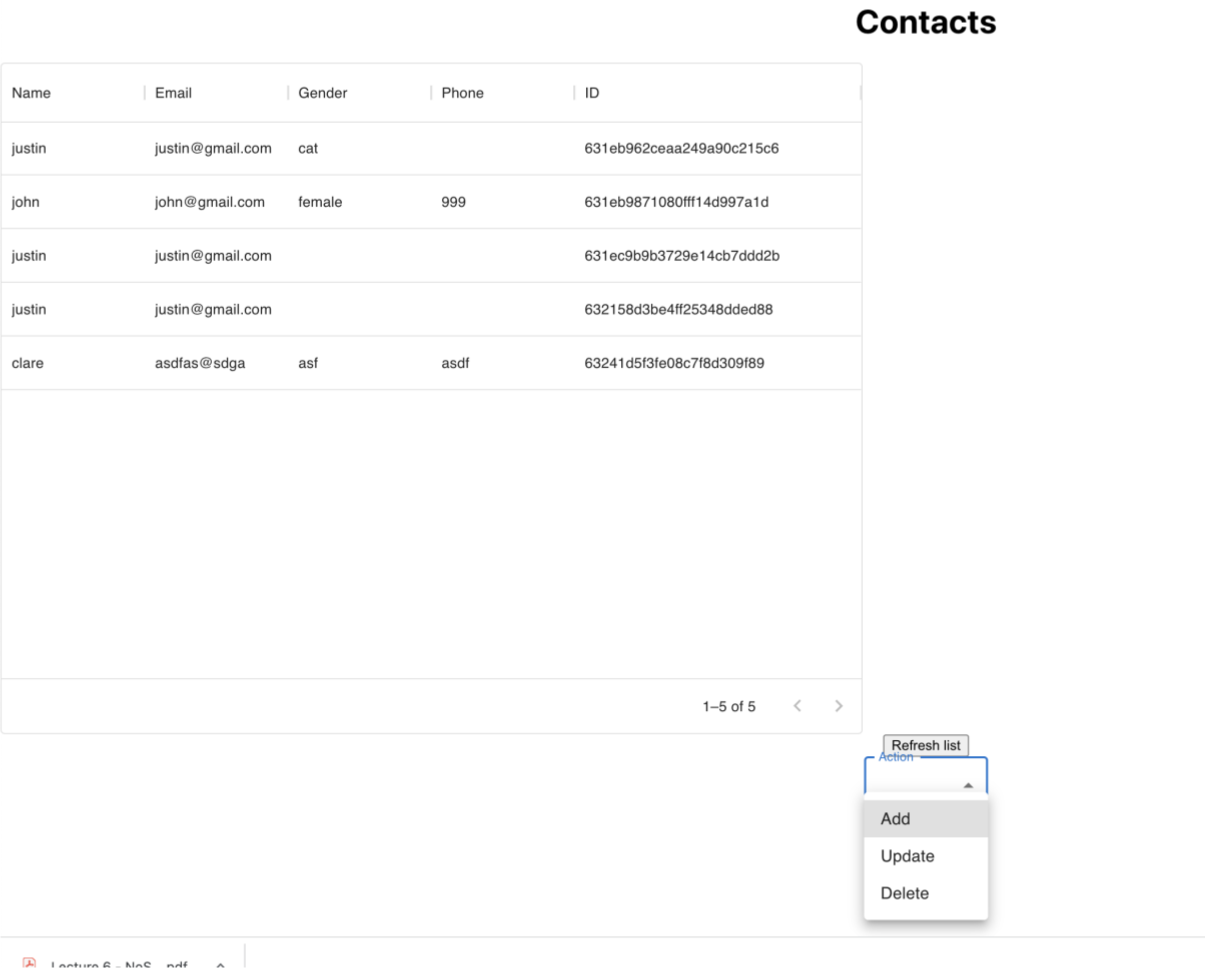


Contacts

Name	Email	Gender	Phone	ID
justin	justin@gmail.com	cat		631eb962ceaa249a90c215c6
john	john@gmail.com	female	999	631eb9871080fff14d997a1d
clare	asdfas@sdga	asf	asdf	63241d5f3fe08c7f8d309f89

1–3 of 3

5. Choose an action to perform with the dropdown menu:

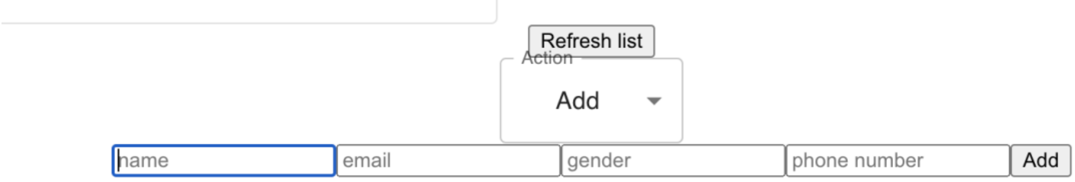


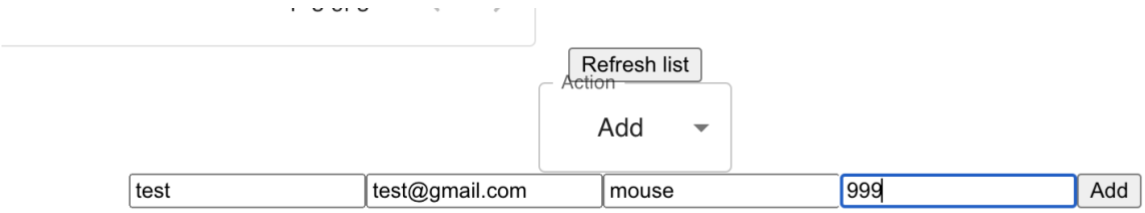
Contacts

Name	Email	Gender	Phone	ID
justin	justin@gmail.com	cat		631eb962ceaa249a90c215c6
john	john@gmail.com	female	999	631eb9871080fff14d997a1d
justin	justin@gmail.com			631ec9b9b3729e14cb7ddd2b
justin	justin@gmail.com			632158d3be4ff25348dded88
clare	asdfas@sdga	asf	asdf	63241d5f3fe08c7f8d309f89

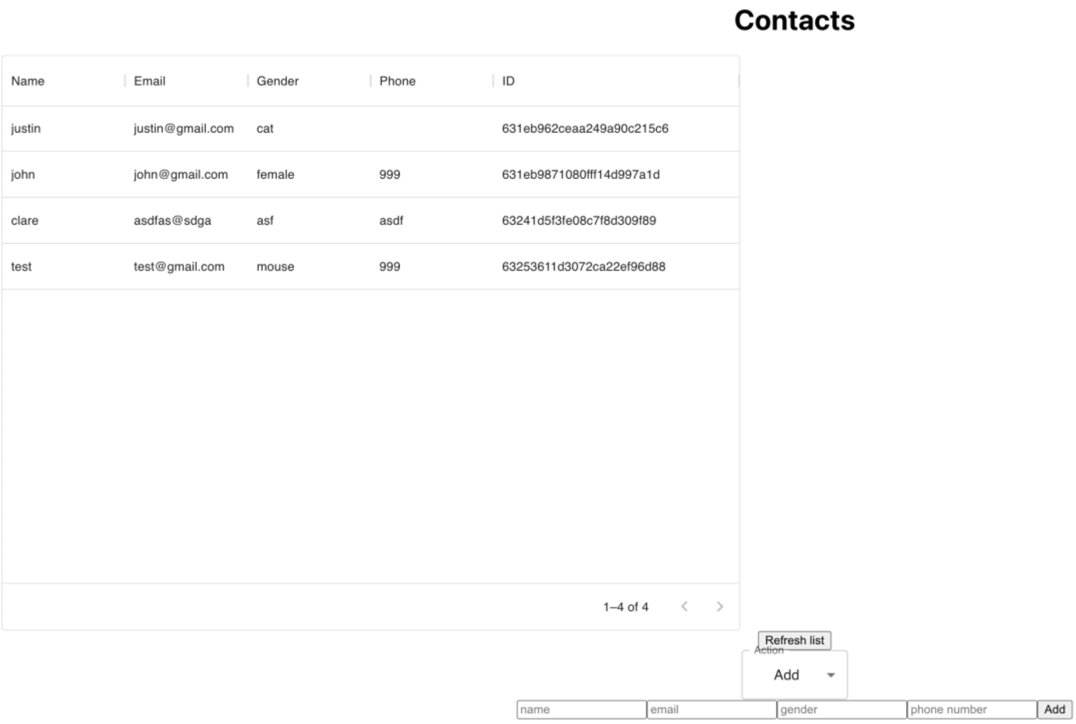
1–5 of 5

6. This is how to add (need to click on refresh):





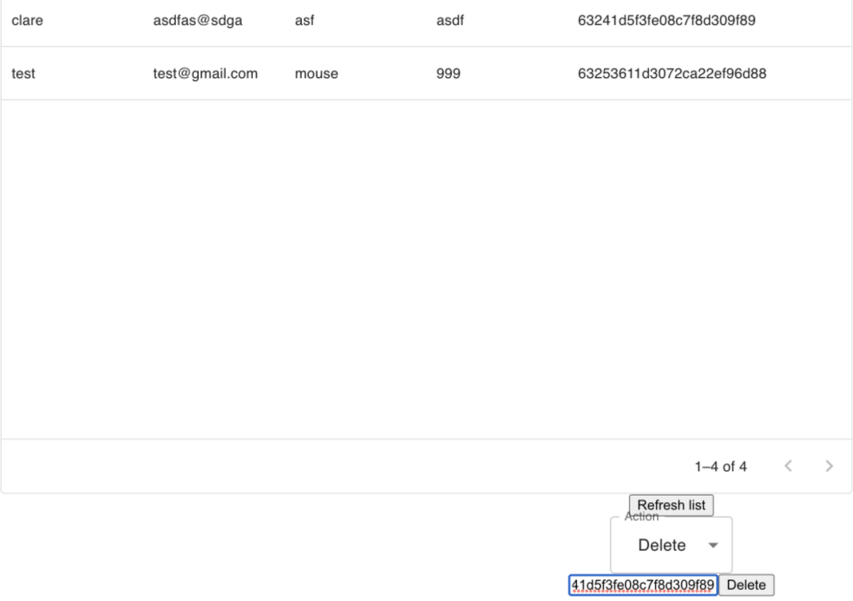
test | test@gmail.com | mouse | 999



Contacts

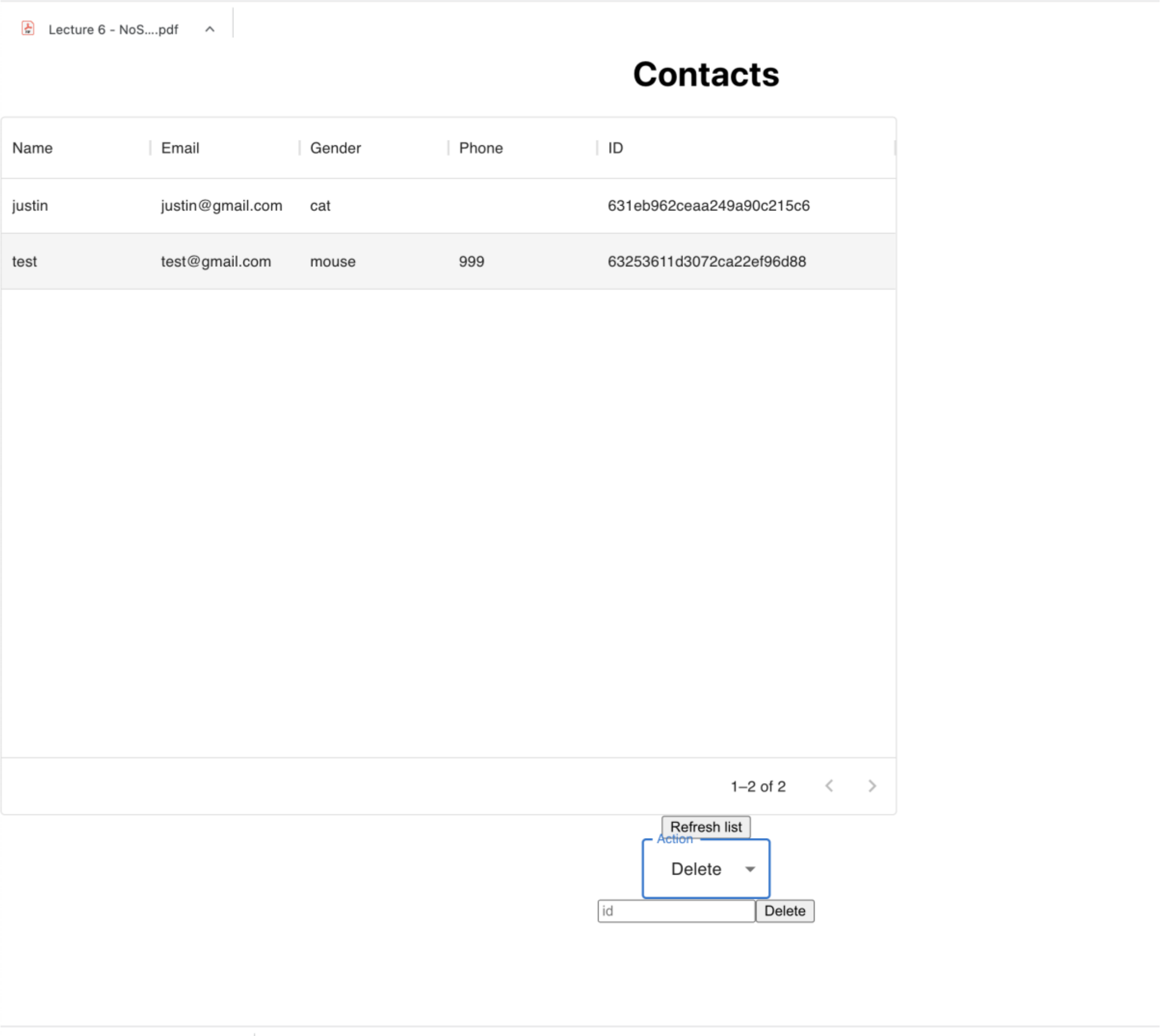
Name	Email	Gender	Phone	ID
justin	justin@gmail.com	cat		631eb962ceaa249a90c215c6
john	john@gmail.com	female	999	631eb9871080fff14d997a1d
clare	asdfas@sdga	asf	asdf	63241d5f3fe08c7f8d309f89
test	test@gmail.com	mouse	999	63253611d3072ca22ef96d88

7. This is how to delete (need to click refresh):



clare	asdfas@sdga	asf	asdf	63241d5f3fe08c7f8d309f89
test	test@gmail.com	mouse	999	63253611d3072ca22ef96d88

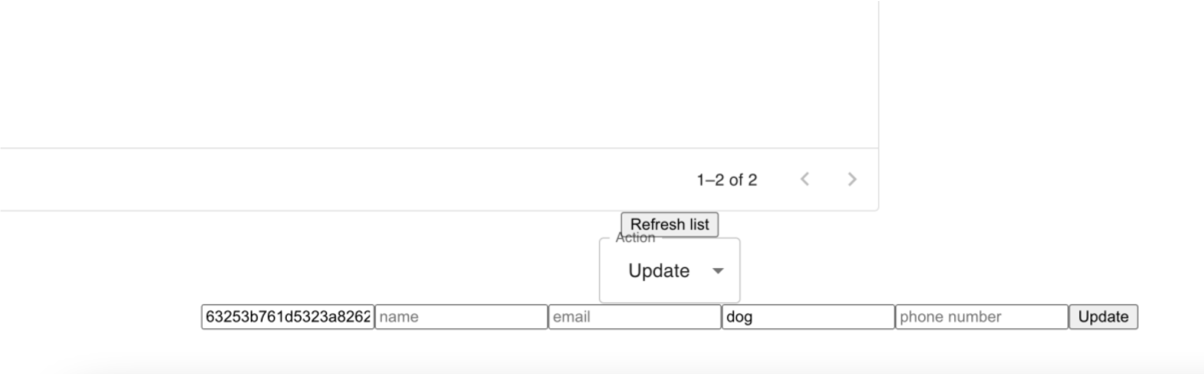
41d5f3fe08c7f8d309f89



Contacts

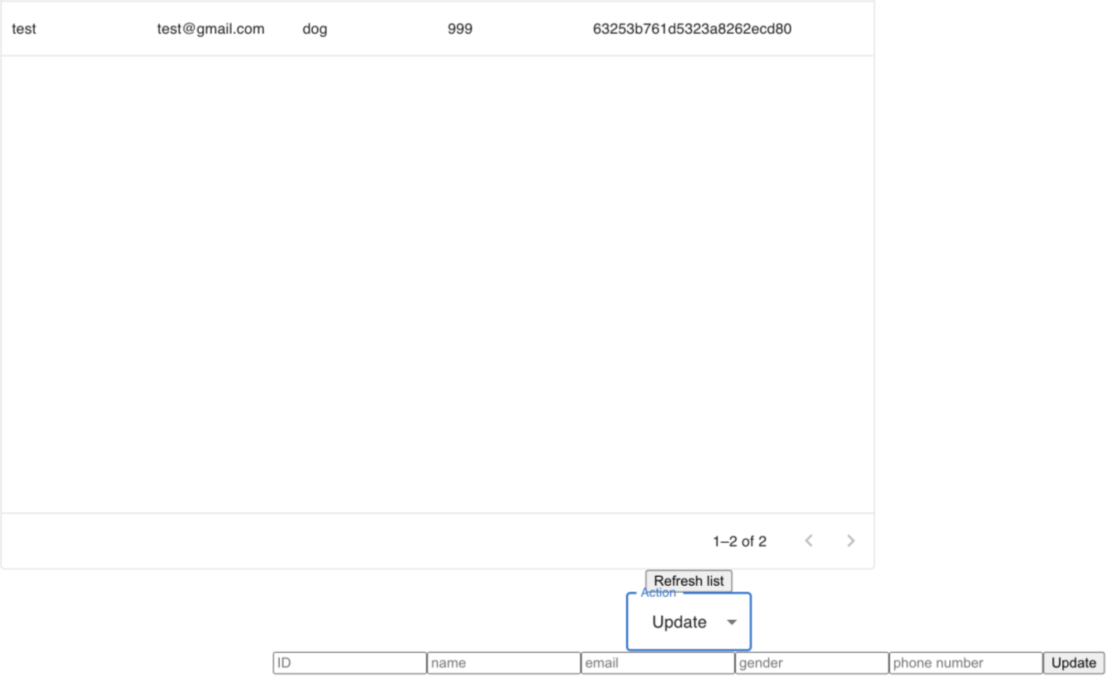
Name	Email	Gender	Phone	ID
justin	justin@gmail.com	cat		631eb962ceaa249a90c215c6
test	test@gmail.com	mouse	999	63253611d3072ca22ef96d88

8. This is how to update:



1–2 of 2

63253b761d5323a8262 | name | email | dog | phone number | Update



test test@gmail.com dog 999 63253b761d5323a8262ecd80

1–2 of 2