

## DATAGROKR Assignment

The app was made with the help of react JS acting as the frontend supporting all the UI components and the API calls to the server were handled with the help of node JS and to store the data on cloud MongoDB is used.

### Information Form

First Name\*

Last Name\*

Mobile number\*

Street Address

City

State

Post Code

Account Number

Yearly Salary

Email\*

Storage Type 

Cloud Server

Add

## FRONTEND[REACT JS]

### TECH USED:

- 1.ReactJS and its hooks
- 2.Axios

## COMPONENTS

### 1.ContactDetails[Form element to get contact details]

The form input element inputs the data with the help of controlled components. The form takes the personal details of user and validate it if the data is invalid a error message will be rendered on immediate right of input field and it will only go away when user types validate data after getting all the data it passes it to its child component for saving either on cloud or local storage as per user choice.

## Information Form

First Name\* Ashish

Last Name\* Roushan

Mobile number\* 6376093465

Street Address P 28 Nehru Colony Ratan

City Jodhpur

State Rajasthan

Post Code 342011

Account Number 75259563456

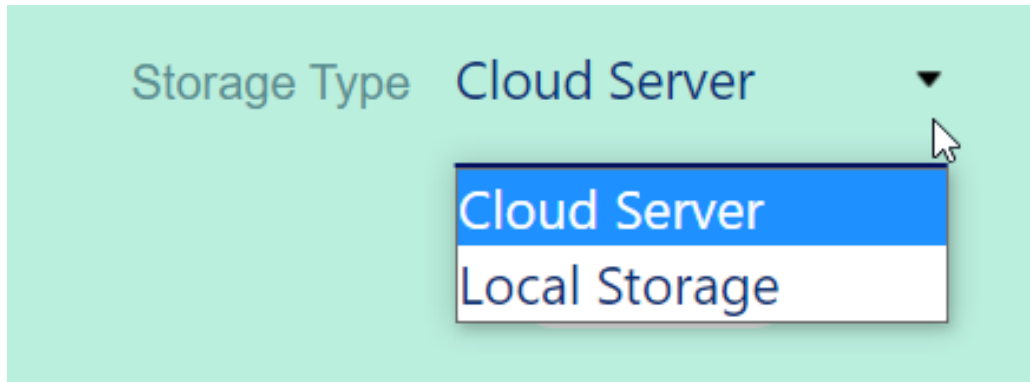
Yearly Salary 800000

Email\* asd@asd.dsa

Storage Type Local Storage ▼

Add

2.SavingData[Stores data either on cloud server or local storage]



This Component gets the value of user details , storage type and id via props. After getting the required data it will be posted to backend server through axios. The useEffect hook is used to call the backend only when id is changed. ID will change whenever ADD button is clicked.

## BACKEND[NODE JS]

### TECH/DEPENDENCIES USED

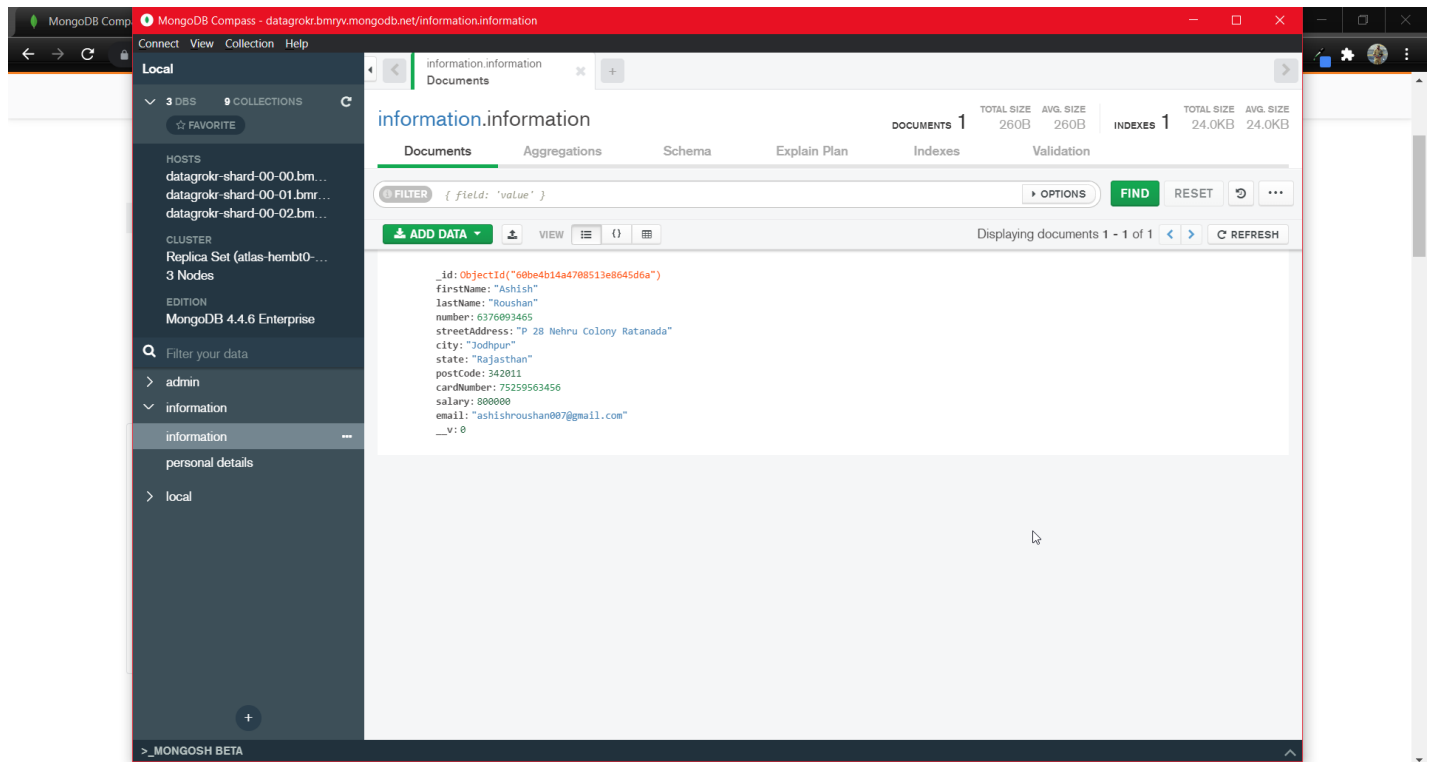
- 1.Node JS
- 2.Express
- 3.Cors
- 4.mongoose
- 5.nodemon
- 6.dotenv

## COMPONENTS

1.index[Connects to database]

The express module is installed in the backend and an express app is created. The app is used to handle the POST request that the axios request made from the frontend react app. It takes the incoming object and stores it in the different variables. There are two routes /dbs which stores data on server and /local which writes data in local file if it gets request at /dbs then all values will be passed down to Information component in form of object and sends the response the data is inserted if there was any error it will log the error in console if /local is called it writes the data in

local storage by using fs in JSON format. MongoDB connection is also established in this component.



Data stored in local file named as contactDetails.json



## 2.Information[mongoose schema]

This component stores the data which in the variables which was going to store on cloud with their datatype if there is any mismatched in datatype or any required field is missing it shows an error.

## TEST CASES

### 1. Invalid email

Yearly Salary  Email\*  invalid

Storage Type

This screenshot shows a form with a light green background and a purple border. The 'Yearly Salary' field contains '800000'. The 'Email\*' field contains 'asd' and is marked as 'invalid' in red text. The 'Storage Type' dropdown menu is set to 'Cloud Server'. The 'Add' button is highlighted with a mouse cursor.

Yearly Salary  Email\*  invalid

Storage Type

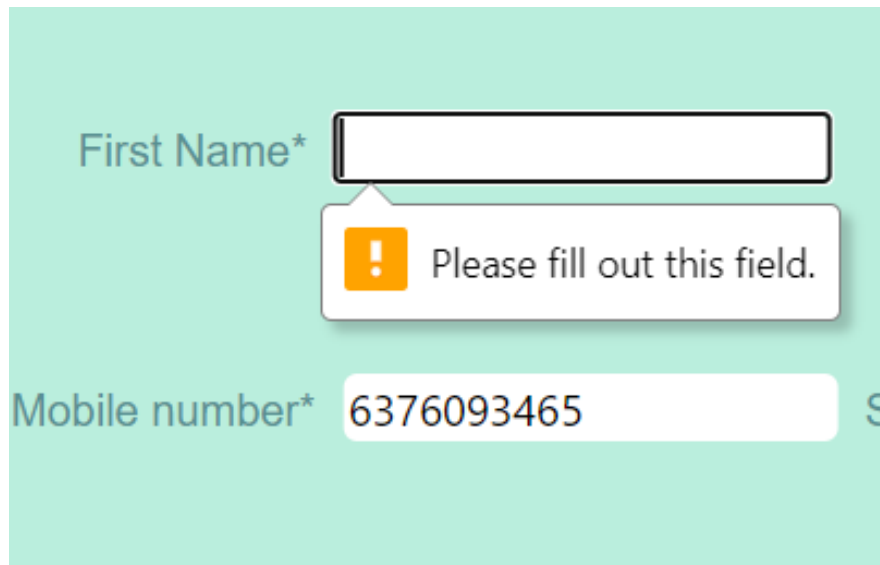
This screenshot shows the same form as the first one, but the 'Email\*' field now contains 'asd@asd' and is still marked as 'invalid' in red text. The 'Add' button is still highlighted with a mouse cursor.

Yearly Salary  Email\*

Storage Type

This screenshot shows the same form, but the 'Email\*' field now contains 'asd@asd.dsa' and is no longer marked as 'invalid'. The 'Add' button is no longer highlighted.

## 2.Empty field



A screenshot of a form on a light green background. The form has two input fields. The first field is labeled "First Name\*" and is empty. A white error message box with a yellow exclamation mark icon is positioned below it, containing the text "Please fill out this field." The second field is labeled "Mobile number\*" and contains the value "6376093465".

## HOW TO RUN CODE

### Client folder

- 1.npm i //it installs required node\_modules
- 2.npm install axios validator

### Server folder

- 1.npm i
- 2.npm install nodemon mongoose express dotenv cors
- 3.make sure to add type:module in package.JSON if not present there.