**This Documentation elaborates about setting up the frontend and the backend.**

*Make sure that node.js is installed in your system.*

**FRONTEND:**

It is an REACT.js application, first we need to install all the required modules to start the application.

Inside the folder “react-frontend”, open terminal in the same location and run the command

*“ npm i ”* or *“ npm install ”*.

After the node modules gets downloaded the use the command *“ npm start “* to start the server and *“ ctrl+c “* to end the server.

Don’t start it now, we need backend also to run the entire application.

**BACKEND:**

This is an express.js application. Similar to the above steps we need to install all the required node modules. Simple run *“ npm I “* which should install all the required modules from the package file.

We are not yet ready to run the Application yet.

Now we need to configure the *“ keys.js “* file.

**In the AWS section:**

*//keys for AWS*

*const aws={*

*region: "",*

*accessKeyId:"",*

*secretAccessKey: ""*

*};*

*const dbUsers="";*

*const dbFiles="";*

*const dbSharedFile="";*

*const bucketName="";*

It should be looking like this now. First, we need *region, accesskeyId* and *secretAccessKey* of your AWS account with full access to dynamodb and S3 storage.

Specify the region where you are hosting the database and S3.

Once you it is all done populate them in the respective places.

After that we need to create three databases in the DynamoDB to maintain the list of users in the platform, files uploaded by the users and also the files shared across the users in the platform.

Create the three databases with partition key as *“ id ”* for all three databases. Once it is done.

Fill the names of those databases in the *“ dbUsers “,* *“ dbFiles “* and *“ dbSharedFiles “.*

Then, create an bucket in the S3 and copy that S3 bucket name and paste in the *“ bucketName ”.*

Check once that all the field are populated correctly.

**In the AZURE section:**

*//keys for AZURE*

*const STORAGE\_ACCOUNT\_NAME = ' ';*

*const ACCOUNT\_ACCESS\_KEY = ” ”;*

*const containerName = ' ';*

Here we need to create an storage account in azure, copy that name and paste it in *the “ STORAGE\_ACCOUNT\_NAME “.*

Now navigate into the created storage account, find Access Keys section and copy the *“ key “* and paste it in the *“ ACCOUNT\_ACCESS\_KEY “.*

Finally navigate to the containers tab in the same tool window and create a new container. Copy and paste the name of that container to *“ containerName “.*

Save the file.

Now almost all the basic configurations are over to run the application.

**Advanced Configurations (\*Advanced according to me)**

**Frontend:**

1. This modification allows you to set the frontend to allow what are all the type of files you need to allowed to be stored sent through the API.

(backend supports all sorts of files upto my knowledge)

Navigate into *“react-frontend->src->layouts->dashboard->dash.js”*

In the 10th line you should encounter this line

*const fileFormats=[".csv",".pdm",".pdf",".jpg",".png",".xls",".xlsx",".ppt",".pptx",".txt"];*

these are all the files which can be uploaded from the frontend, if any other format of file is to be uploaded it can be added to the list of the array.

**Backend:**

1. Setting the limit or the maximum size of the files that the API can handle.

Navigate to the *“server->app.js”* file, here

You must be seeing the below lines at 17&18 respectively.

*app.use(express.json({ extended: false, limit: '50mb' }));*

*app.use(express.urlencoded({ limit: '50mb', extended: false, parameterLimit: 50000 }));*

here you can set the limit for the maximum size of the file that the API can handle.

1. Allowing the server to respond to requests from specific IP addresses alone.

In the same file line no:20

*// ------------------------configuration of cors----------------------*

*var corsOptions = {*

*origin: '\*',*

*optionsSuccessStatus: 200*

*};*

The origin property can be set to the list of IP address or the IP where the frontend is hosted, it is set to default as *“ \* “* to listen to all the requests across all the IP.

1. Changing the port number of the server to run on the local machine for testing.

At the end of the same file,

*app.listen(process.env.PORT || 3001, function(){*

*console.log("Server started on port 3001");*

*});*

Here the listening port is set to *“ 3001 ”,* this can be changed to some other services are running on the same port or you just needed to change it.

(If the port number is changed then it should be update across the following places in the frontend such that the application wont crash”

* *“react-frontend->src->layouts->dashboard->dash.js”* in this file the lines needs to be changed are 67, 81, 95, 117, 144, 175, 205, 232 and 261.
* *“react-frontend->src->layouts->login->login.js”* in this file the lines needs to be changed is 36.
* *“react-frontend->src->layouts->signup->signup”* in this file the lines needs to be changed is 52.