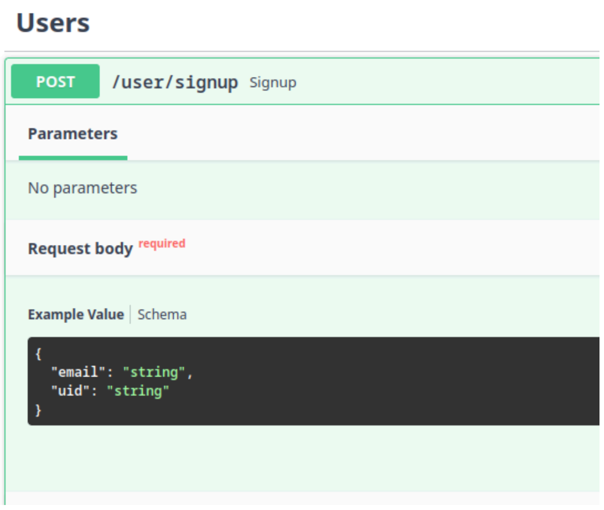
RECOVER FrontEnd intergration instructions

- The link to the backed and backend doc is `http://ec2-18-220-202-10.us-east-2.compute.amazonaws.com/docs`

Login-signup

Sign-up

- A sample template has been provided for this named `login.html`
- The below API is used to store the email id and the uid which we get from firebase


Users

POST /user/signup Signup

Parameters

No parameters

Request body required

Example Value | Schema

```
{
  "email": "string",
  "uid": "string"
}
```

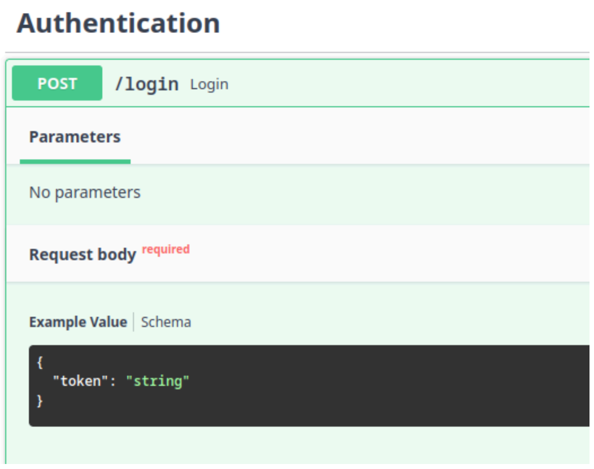
- Firebase details are given below

```
const firebaseConfig = {
apiKey: "AIzaSyCjLvkZ1r_MFuu9DlgETzJXdEAQrkucrWw",
authDomain: "recover-trial.firebaseapp.com",
projectId: "recover-trial",
storageBucket: "recover-trial.firebasestorage.app",
messagingSenderId: "1017206253975",
appId: "1:1017206253975:web:561d0c1d5774173b67bcd2"
};
```

- The code to send verify email is also present in the html template (which has been commented out)

Login

- The login is done using firebase and the token is sent to the backend via the below API


Authentication

POST /login Login

Parameters

No parameters

Request body required

Example Value | Schema

```
{
  "token": "string"
}
```

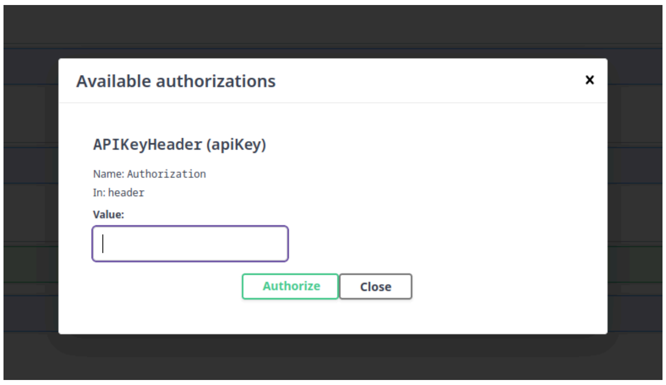
- User should only be allowed to login if his email is verified.
- A response like the one given below will be served by the backend

```
{'access_token':
'eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiJzdXBlcnVzZ
XJAZ21haWwuY29tIiwiZXhwIjoxNzQ0NTYxMTE0fQ.gkV-RqFWJ-
_MHPve_XLTQhfkvLI_Mqo12ACg2R-TYsU', 'token_type': 'bearer',
'uid': '4D4XK4VMwRZaNZjBrboK3sU5x7q2', 'is_info': False,
'is_verified': False}
```

- `access_token` is the JWT token which should be passed via the header during each response like below

```
headers: {
'Accept': 'application/json',
'Authorization':
'eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiJzdXBlcnVzZ
XJAZ21haWwuY29tIiwiZXhwIjoxNzQ0NTYxMTE0fQ.gkV-RqFWJ-
_MHPve_XLTQhfkvLI_Mqo12ACg2R-TYsU'
    }
```

- The access token can be used with the API Doc to execute cells in the API Doc itself.
- For that you will have to paste the access token in the part where it says "Authorize"


Available authorizations

APIKeyHeader (apiKey)

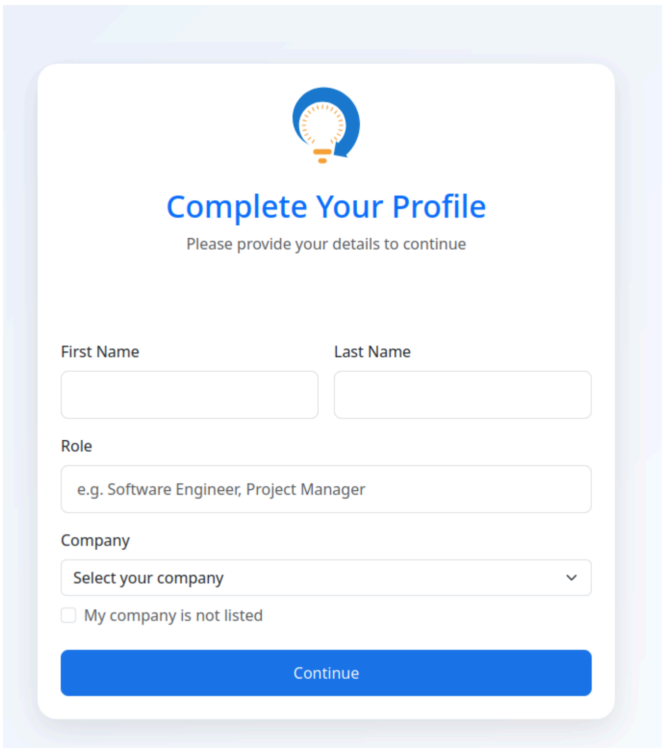
Name: Authorization

In: header

Value:

Get details page

- `is_info` states whether the user's info is present. If its false then the user should be redirected to the below page



Complete Your Profile

Please provide your details to continue

First Name

Last Name

Role

e.g. Software Engineer, Project Manager

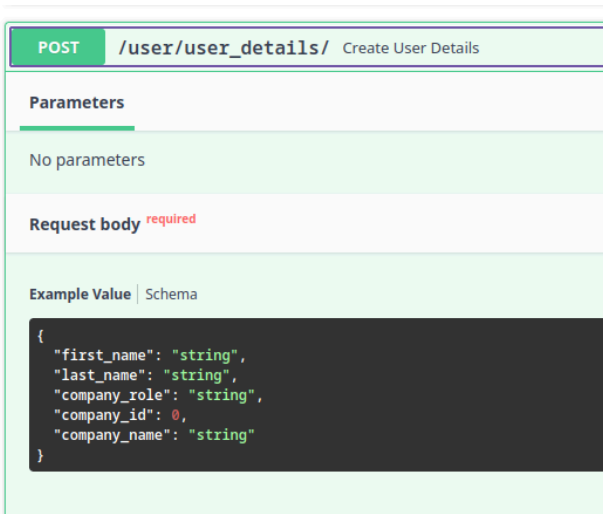
Company

Select your company

My company is not listed

Continue

- A sample of this page is present in the repo called `get_details.html`
- The form is submitted into the below response



POST /user/user_details/ Create User Details

Parameters

No parameters

Request body required

Example Value | Schema

```
{
  "first_name": "string",
  "last_name": "string",
  "company_role": "string",
  "company_id": 0,
  "company_name": "string"
}
```

- The list of companies in the dropdown can be obtained using `/company/list`. The name of the company should be shown in the list and the id is used in the post request.

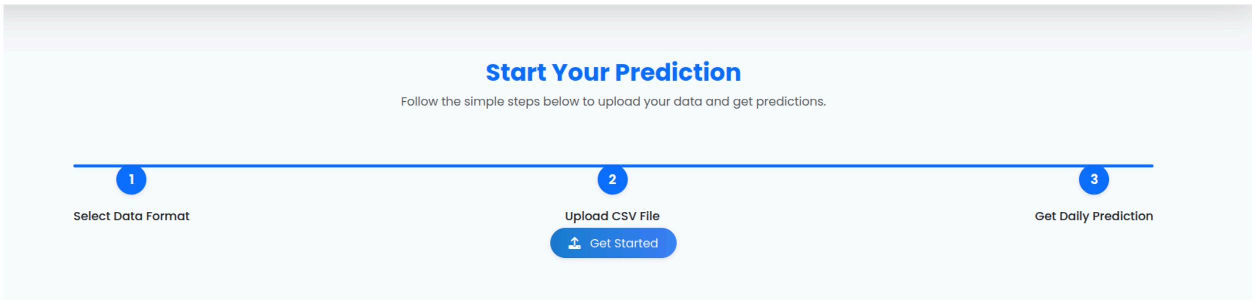
- The request accepts first_name, last_name, company_role and company_id.
- If the user selects "My company is not listed", a text box will appear and the request should contain company_name instead of company_id
- `company_id` and `company_name` should NOT be present simultaneously in the request

Is verified page

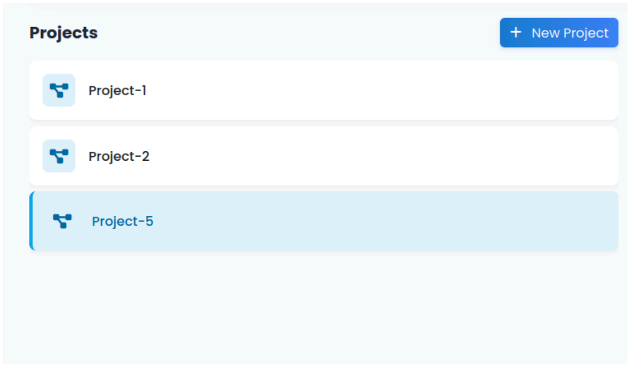
- If in the login request `is_info` is `true`, It should be checked whether `is_verified` is true.
- If `is_verified` is false then the user should be redirected to a page which says. - "Your account is pending verification. Our team will review your details and notify you once your account is approved. If you have any questions, please contact support."
- This page isn't present in the front end template.
- A user can be verified using `/user/verify/{id}` . There is no need for any frontend integration for this API.

Main dashboards

- If both `is_info` and `is_verified` were true while logging in the user should be redirected to the dashboard.
- The template corresponding to this is `dashboard.html`
- After signing up the list of projects can be retrieved by `/projects/list`
- If the list is empty then a similar page to the below should be shown. (Please fix the css while integrating)

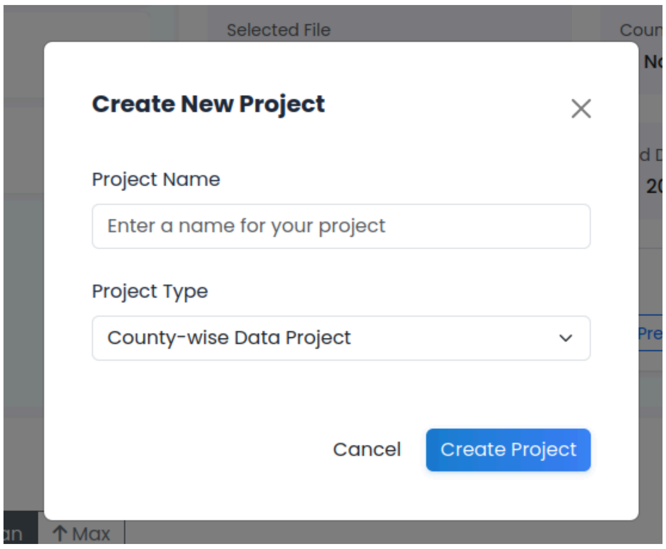


- If there are projects present then it should be listed as below.



(The later section of the document details what happens when a project is selected.)

Creating a new project


Create New Project

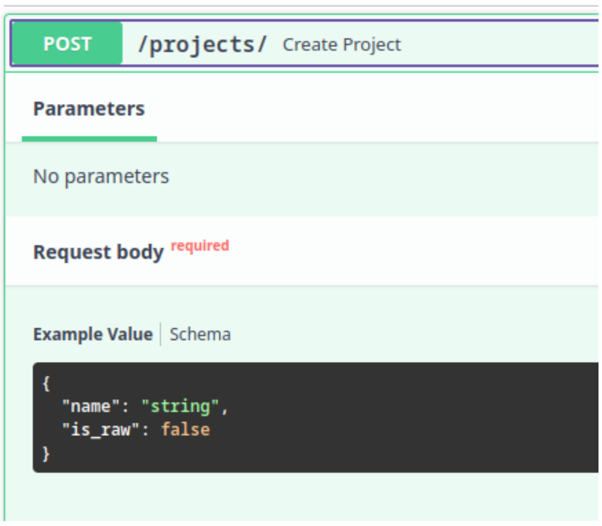
Project Name

Enter a name for your project

Project Type

County-wise Data Project

- The below api is used to create the project when clicking the "create project" button.


POST /projects/ Create Project

Parameters

No parameters

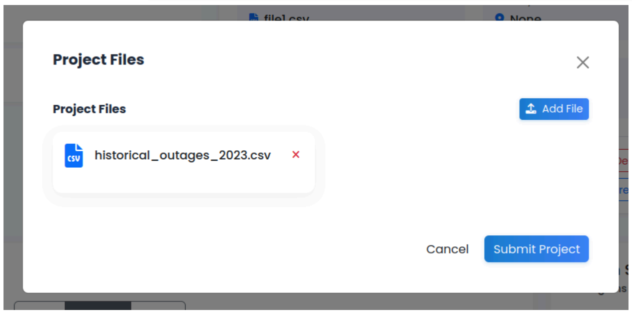
Request body required

Example Value | Schema

```
{
  "name": "string",
  "is_raw": false
}
```

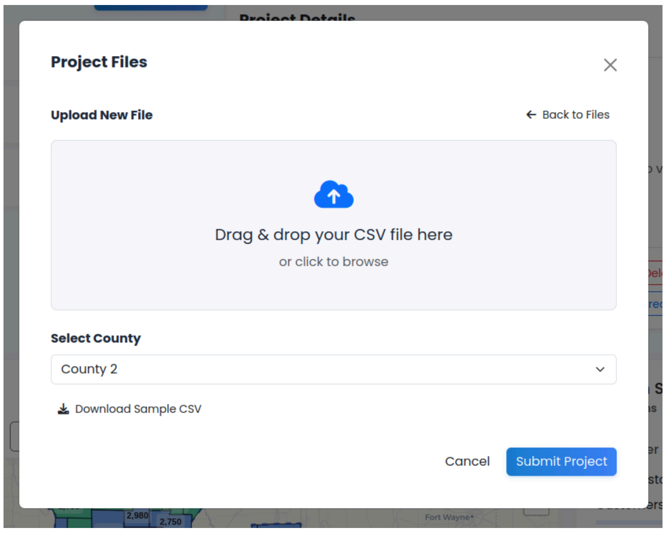
- The files used in the project and be of 2 types raw file and county wise file
- If the user selects "Raw OMS Data Project" the `is_raw` should be set to false. Otherwise it should be true

- Past uploaded files are shown in the next page which can be obtained by doing a get request at `/files/list/{project_id}`

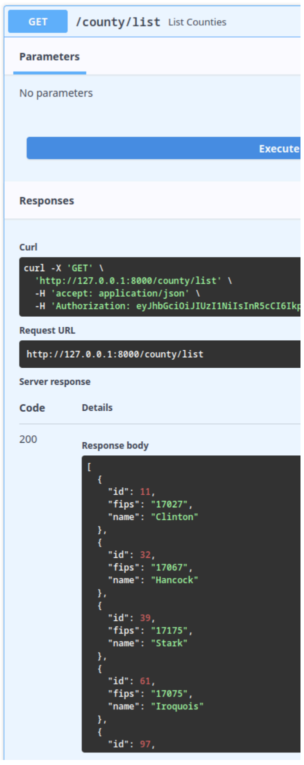


- On clicking the 'x' mark the file should be deleted by making a delete request at `/files/list/{project_id}`
- If no files are found the same should be displayed

- The below page is used to upload files.



- You can get the list of counties using the below API



No parameters

Curl

```
curl -X 'GET' \
  'http://127.0.0.1:8000/county/list' \
  -H 'accept: application/json' \
  -H 'Authorization: eyJhbGciOiJIUzI1NiIsInR5cCI6Ikp
```

Request URL

```
http://127.0.0.1:8000/county/list
```

Server response

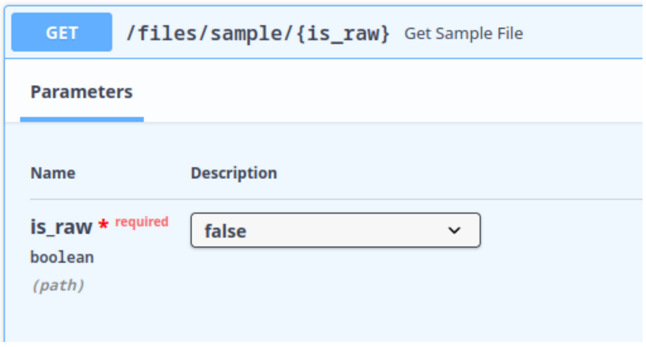
Code | Details

200

Response body

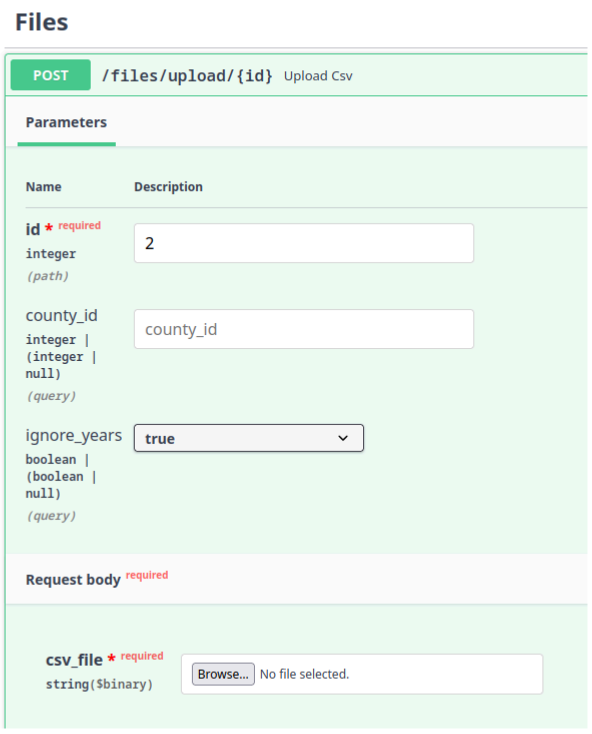
```
[
  {
    "id": 11,
    "fips": "17027",
    "name": "Clinton"
  },
  {
    "id": 32,
    "fips": "17067",
    "name": "Hancock"
  },
  {
    "id": 39,
    "fips": "17175",
    "name": "Stark"
  },
  {
    "id": 61,
    "fips": "17075",
    "name": "Iroquois"
  },
  {
    "id": 97,
```

- The below API should be used while downloading sample. The values of `is_raw` will be same as the one used while initialising the project.



Name	Description
is_raw * required boolean *(path)*	false

- A few things to take into consideration while uploading data
  - If the project type is raw (if `is_raw` was true) then the county id field must be empty.
  - If the project type is raw (if `is_raw` was true) then the only one file can be uploaded.
  - A sample raw data is present in `sample data/sample_raw_data.csv`
  - A sample county data is present in `sample data/sample_county_data.csv`
  - `ignore_years` should be `false` by default except in the below mentioned case.
  - While uploading if an error occurs (404,400 etc..) then the error should be shown as a pop up. But if the error code is 422. Then the pop up should say "Data is older than 2 years, this might effect the accuracy of the model. Do you want to continue" and the user should be given two option `yes` or `no`.
  - If the user selects `yes` then do the same request again with `ignore_years` to true otherwise close the dialogue box.



Files

POST /files/upload/{id} Upload Csv

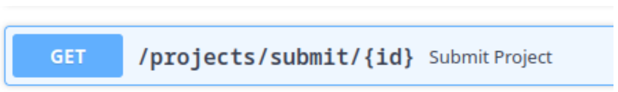
Parameters

Name	Description
id * required integer (path)	2
county_id integer | (integer | null) (query)	county_id
ignore_years boolean | (boolean | null) (query)	true

Request body required

csv_file * required
string($binary)
Browse... No file selected.

- When the submit button is pressed the following endpoint should be used

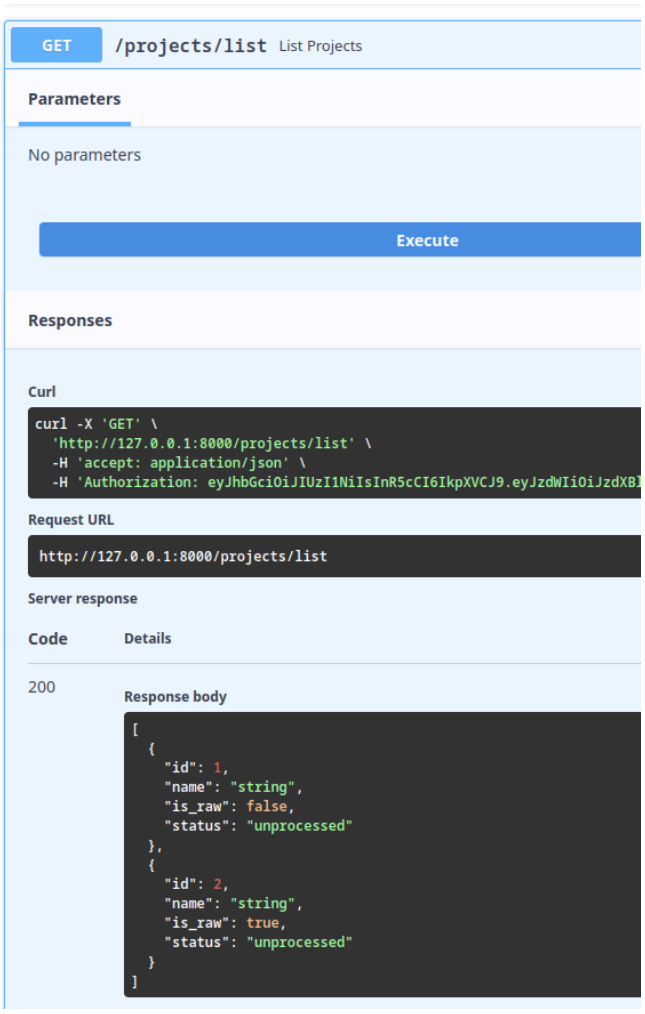


GET /projects/submit/{id} Submit Project

- This will start a background task in the backed.

Listing projects

- The list of projects currently present can be obtained by the following API.



GET `/projects/list` List Projects

Parameters

No parameters

Responses

Curl

```
curl -X 'GET' \
  'http://127.0.0.1:8000/projects/list' \
  -H 'accept: application/json' \
  -H 'Authorization: eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiJzdXBl
```

Request URL

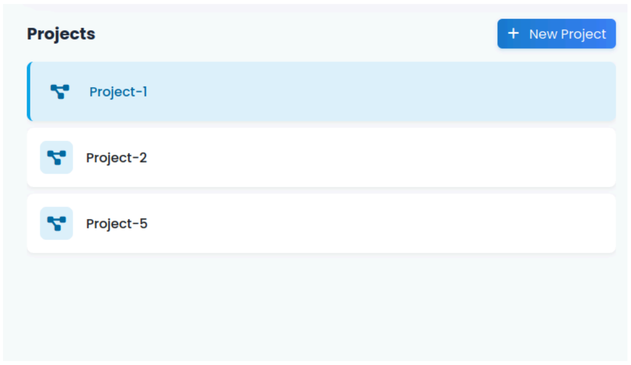
```
http://127.0.0.1:8000/projects/list
```

Server response

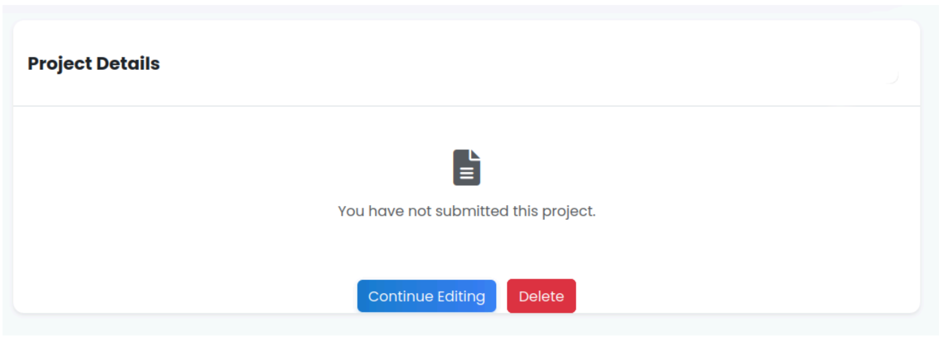
Code	Details
200	Response body

```
[
  {
    "id": 1,
    "name": "string",
    "is_raw": false,
    "status": "unprocessed"
  },
  {
    "id": 2,
    "name": "string",
    "is_raw": true,
    "status": "unprocessed"
  }
]
```

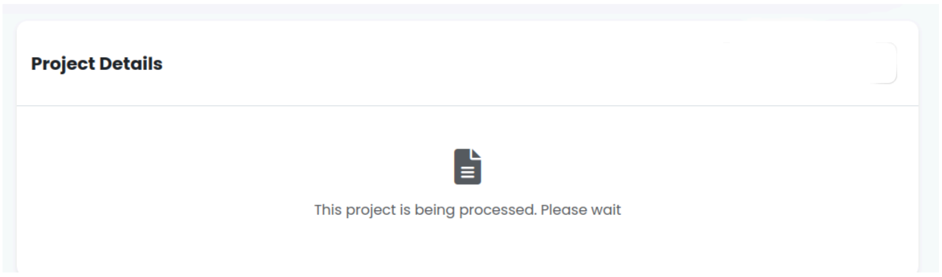
- It should be listed as below



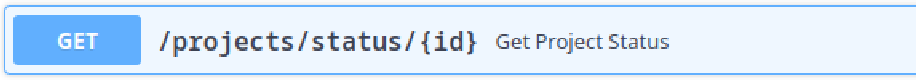
- If a project is selected and its `status` is `unprocessed`, show the below dialogue box under project details



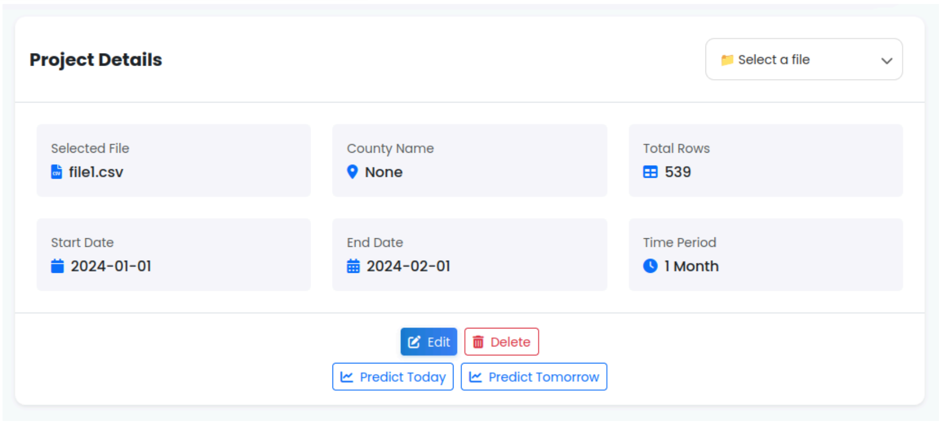
- If a project is selected and its `status` is `processing`, show the below dialogue box under project details



- If the status is `processing` and that project is selected, keep checking the status by pinging the back end by using the below api until the `status` becomes `processed`



- If the project status is `processed`, project details should be loaded.
- The files in the select file list can be obtained by. `/files/list/{project_id}`

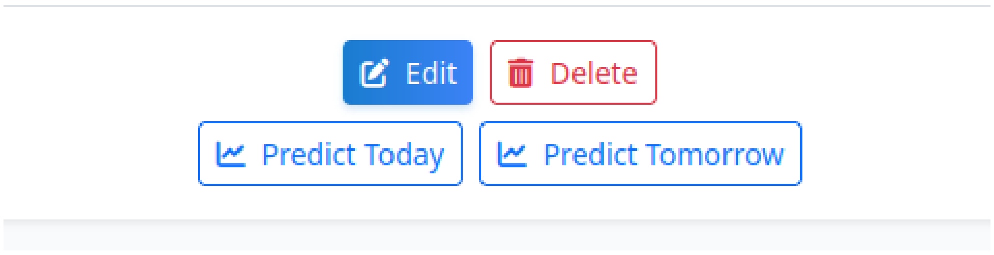


Selected File file1.csv	County Name None	Total Rows 539
Start Date 2024-01-01	End Date 2024-02-01	Time Period 1 Month

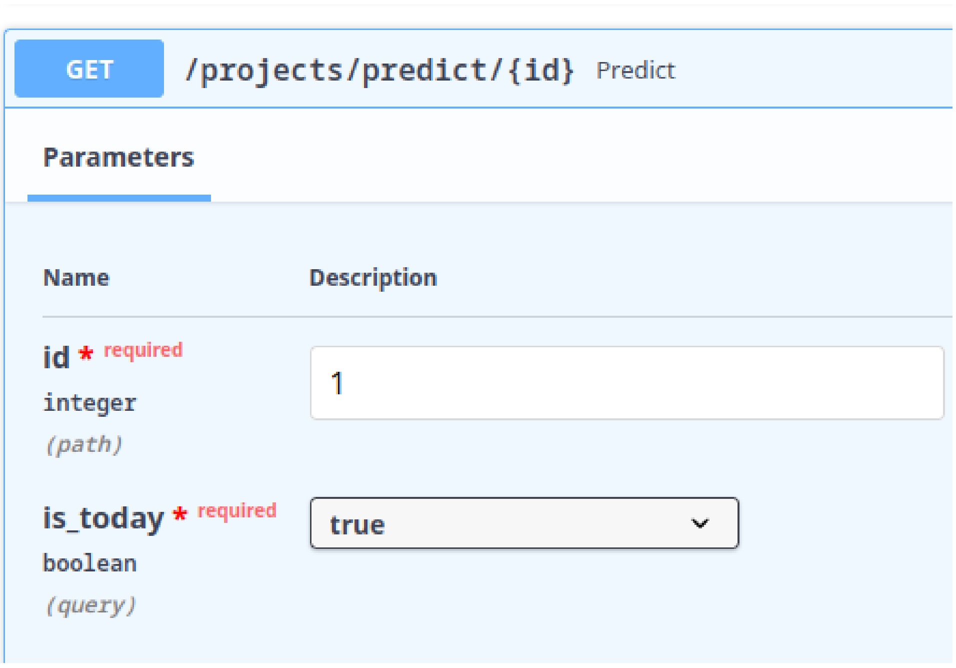
- When a file is selected, the details regarding the file can be obtained by `/files/details/{file_id}`
- The delete endpoint can be triggered by `/projects/delete/{id}`
- If you select the edit button, the same pop up which was used to upload the files and submit should be displayed.

Making and displaying predictions

- IMPORTANT NOTE - In the sample `html` geoid has been shown. These should be replaced with county name.
- The user has two options while making predictions. One to get the prediction of current day or for tomorrow.



- Prediction can be made using the following endpoint
- if `is_today` is set to `true` then today's prediction will be made (or else tomorrow's prediction will be made)



/projects/predict/{id} Predict

Name	Description
id * required integer (path)	1
is_today * required boolean (query)	true

Displaying the map

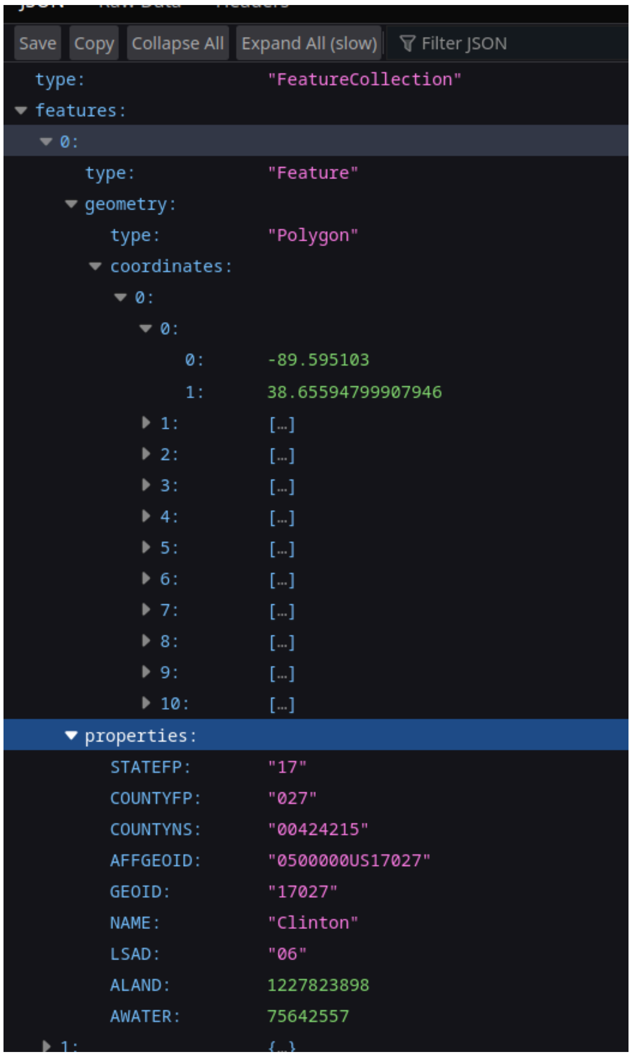
- Two maps should be displayed after prediction.

Displaying the first map

- The geojson of the first map can be obtained using `/company/geojson?file_type=company`
- This will outline the companies region

Displaying the second map

- The second map will show the predictions and can be obtained from `/company/geojson?file_type=county`
- A sample geojson is given below

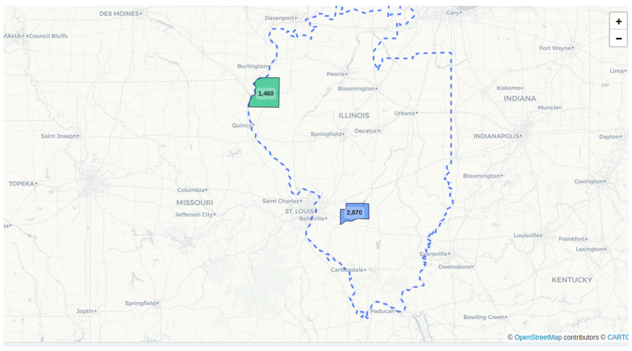


```
  type:                  "FeatureCollection"
▾ features:
  ▾ 0:
      type:              "Feature"
    ▾ geometry:
        type:            "Polygon"
      ▾ coordinates:
        ▾ 0:
          ▾ 0:
              0:         -89.595103
              1:         38.65594799907946
          ▸ 1:           […]
          ▸ 2:           […]
          ▸ 3:           […]
          ▸ 4:           […]
          ▸ 5:           […]
          ▸ 6:           […]
          ▸ 7:           […]
          ▸ 8:           […]
          ▸ 9:           […]
          ▸ 10:          […]
    ▾ properties:
        STATEFP:         "17"
        COUNTYFP:        "027"
        COUNTYNS:        "00424215"
        AFFGEOID:        "0500000US17027"
        GEOID:           "17027"
        NAME:            "Clinton"
        LSAD:            "06"
        ALAND:           1227823898
        AWATER:          75642557
  ▸ 1:                   {…}
```

- And a sample prediction will look like

```
{
  "17027": {
    "outage": {
      "min": 15,
      "mean": 24,
      "max": 38
    },
    "customers": {
      "min": 1711,
      "mean": 2917,
      "max": 5128
    }
  },
  "17067": {
    "outage": {
      "min": 13,
      "mean": 16,
      "max": 21
    },
    "customers": {
      "min": 1000,
      "mean": 1750,
      "max": 2471
    }
  }
}
```

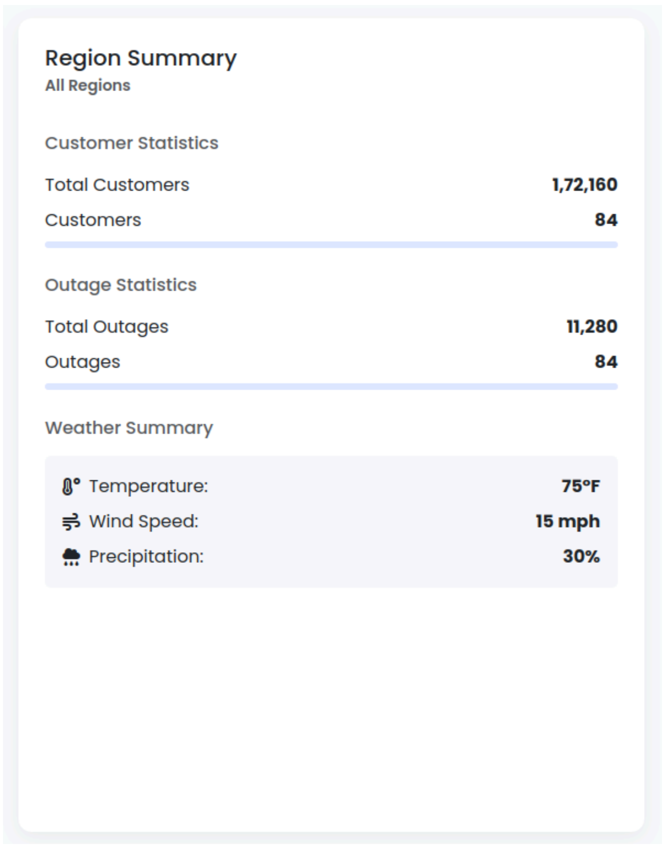
- The id given in the prediction should be matched with the id given in the geojson and only those regions should be displayed whose data is available.
- Eg - In the above case predictions for the regions `17027` and `17067` should be shown and the map will look like



- The user should be able to switch b/w min, max and mean for customers and outages.

Displaying summaries

- If a user selects a region on the map the stats on the right should be updated to show the customers and outages of that region



Region Summary
All Regions

Customer Statistics

Total Customers	1,72,160
Customers	84

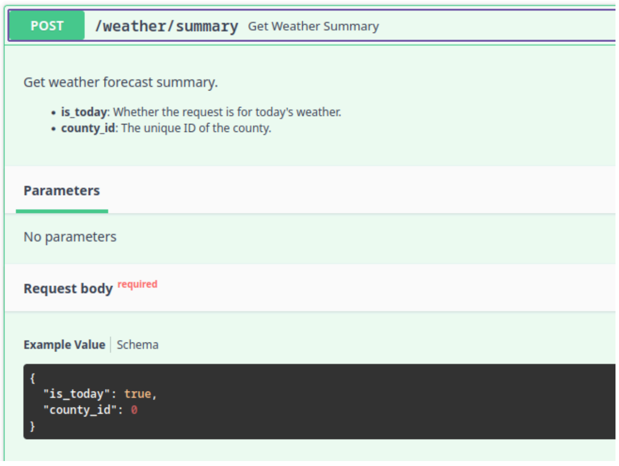
Outage Statistics

Total Outages	11,280
Outages	84

Weather Summary

Temperature:	75°F
Wind Speed:	15 mph
Precipitation:	30%

- The weather summary of that region can be obtained by.



POST /weather/summary Get Weather Summary

Get weather forecast summary.

- **is_today**: Whether the request is for today's weather.
- **county_id**: The unique ID of the county.

Parameters

No parameters

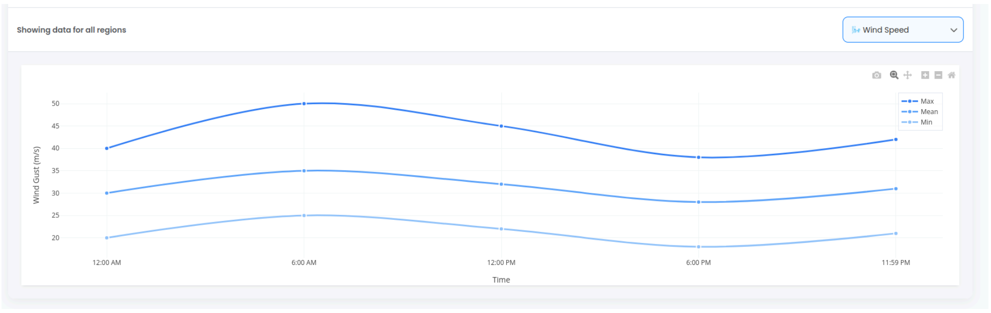
Request body required

Example Value | Schema

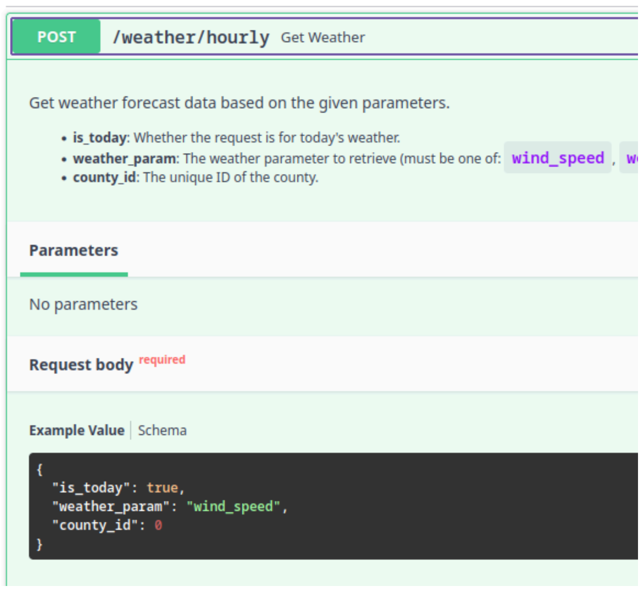
```
{
  "is_today": true,
  "county_id": 0
}
```

- Here the value of `is_today` is same as the one used during prediction.
- Note that in this request `county_id` should be used and not `geoid`. The information of the name, county_id and geoid of a region can be obtained from `/county/list` endpoint

- The following values should be displayed over here
  "min_temperature"
  "max_temperature"
  "max_wind_gusts"
  "avg_wind_spd"
  "precipitation"
  "avg_wind_dir"
  "precipitation_probability_max"
  "weather_code"
- If a user selects a county the corresponding weather graph should also be updated
- The user has a drop-down to select weather parameters and has the following options - `wind_speed` , `weather_code` , `wind_gusts` , `temperature` , `precipitation` , `wind_direction`
- `wind_gust` should be the default option



- The data for the same can be obtained by the below API


POST /weather/hourly Get Weather

Get weather forecast data based on the given parameters.

- **is_today**: Whether the request is for today's weather.
- **weather_param**: The weather parameter to retrieve (must be one of: `wind_speed` , `w
- **county_id**: The unique ID of the county.

Parameters

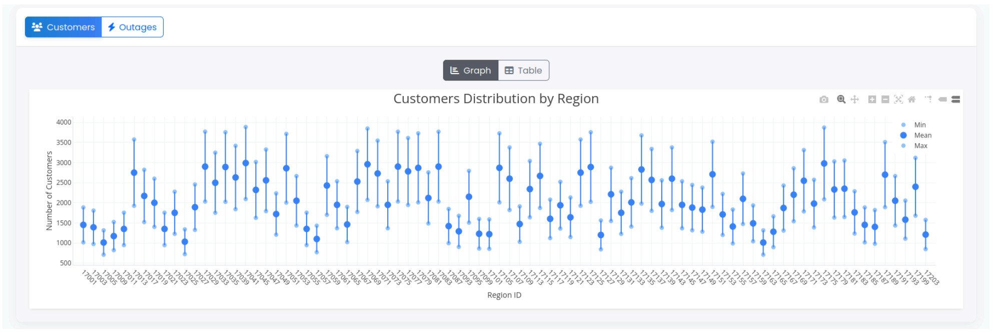
No parameters

Request body required

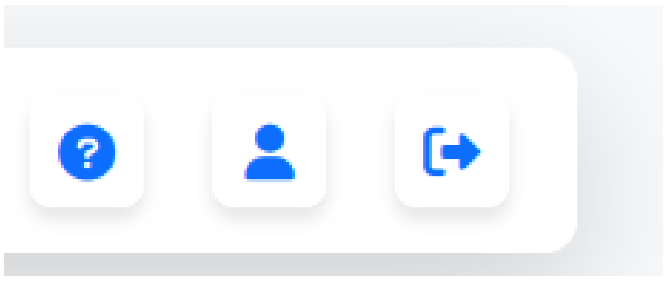
Example Value | Schema

```
{
  "is_today": true,
  "weather_param": "wind_speed",
  "county_id": 0
}
```

- Here `is_today` is same as the one used during prediction
- Note that in the `html` there are three graph. In the actual product only one graph should be shown at a time.
- This is the graphical/tabular representation of the predictions



- Top bar icons



- If you click on the help icon a model should pop up which will show instructions as a [scrible]() embedded
- If the profile icon is selected the user's details should be displayed using `/my_details`