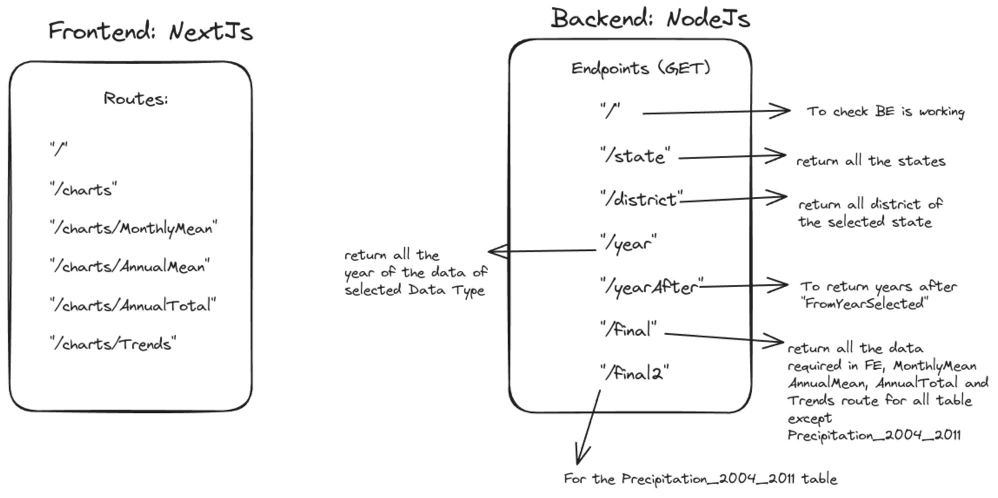
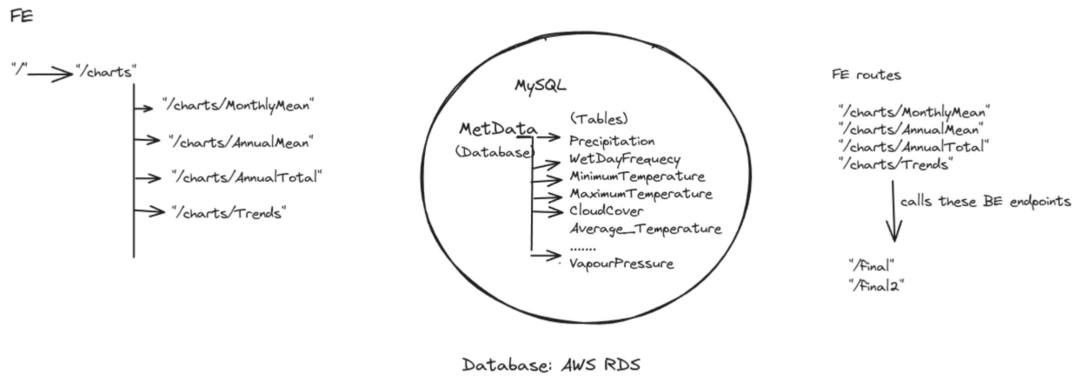
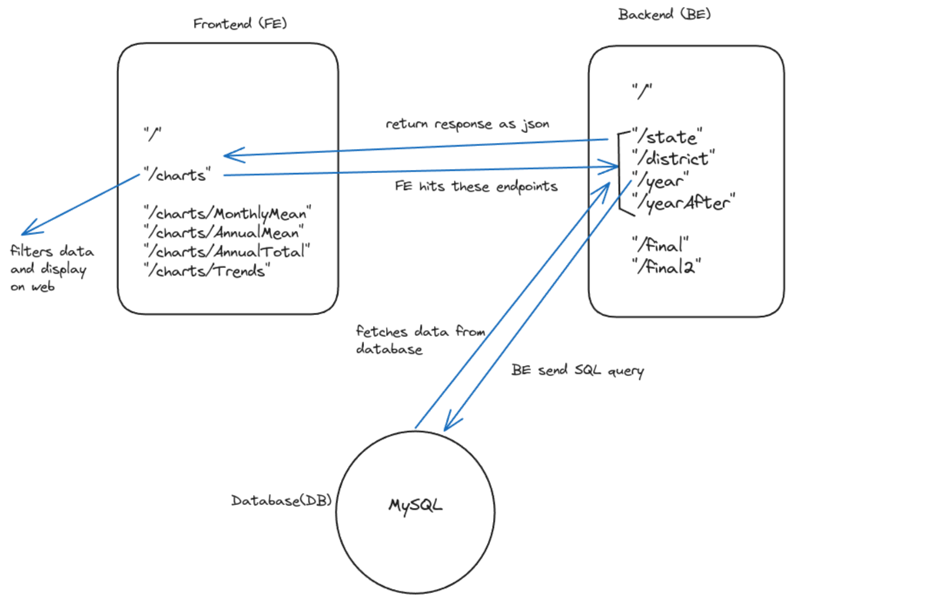
Architecture



Frontend: NextJs
Routes:
"/"
"/charts"
"/charts/MonthlyMean"
"/charts/AnnualMean"
"/charts/AnnualTotal"
"/charts/Trends"
Backend: NodeJs
Endpoints (GET)
"/"
To check BE is working
"/state"
return all the states
"/district"
return all district of the selected state
"/year"
return all the year of the data of selected Data Type
"/yearAfter"
To return years after "FromYearSelected"
"/final"
return all the data required in FE, MonthlyMean AnnualMean, AnnualTotal and Trends route for all table except Precipitation_2004_2011
"/final2"
For the Precipitation_2004_2011 table



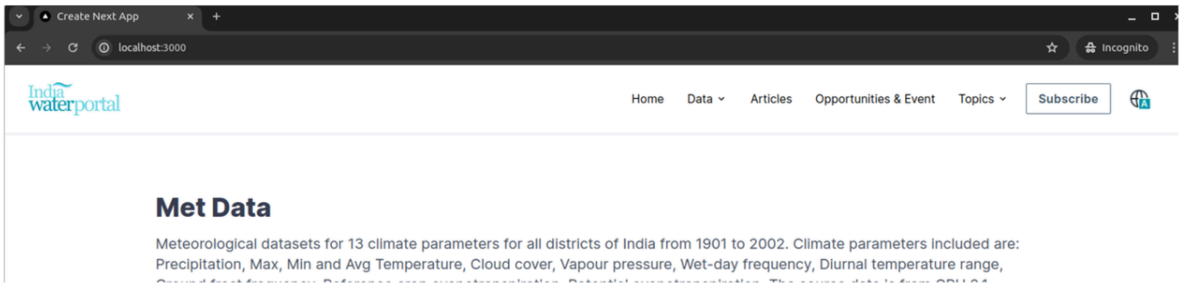


UI / UX Design

- **Navbar :** as Met Data is a product of India Water Portal , navbar will have same theme as of the India Water Portal website (assuming it as an feature of https://www.indiawaterportal.org/)

Desktop View

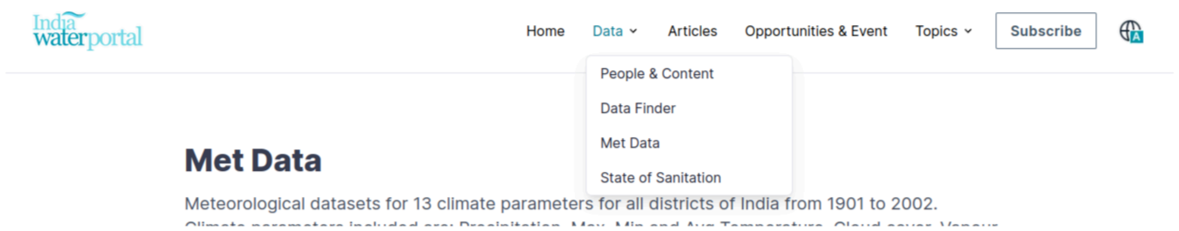
Desktop View of the Navbar



Met Data

Meteorological datasets for 13 climate parameters for all districts of India from 1901 to 2002. Climate parameters included are: Precipitation, Max, Min and Avg Temperature, Cloud cover, Vapour pressure, Wet-day frequency, Diurnal temperature range,

Dropdown



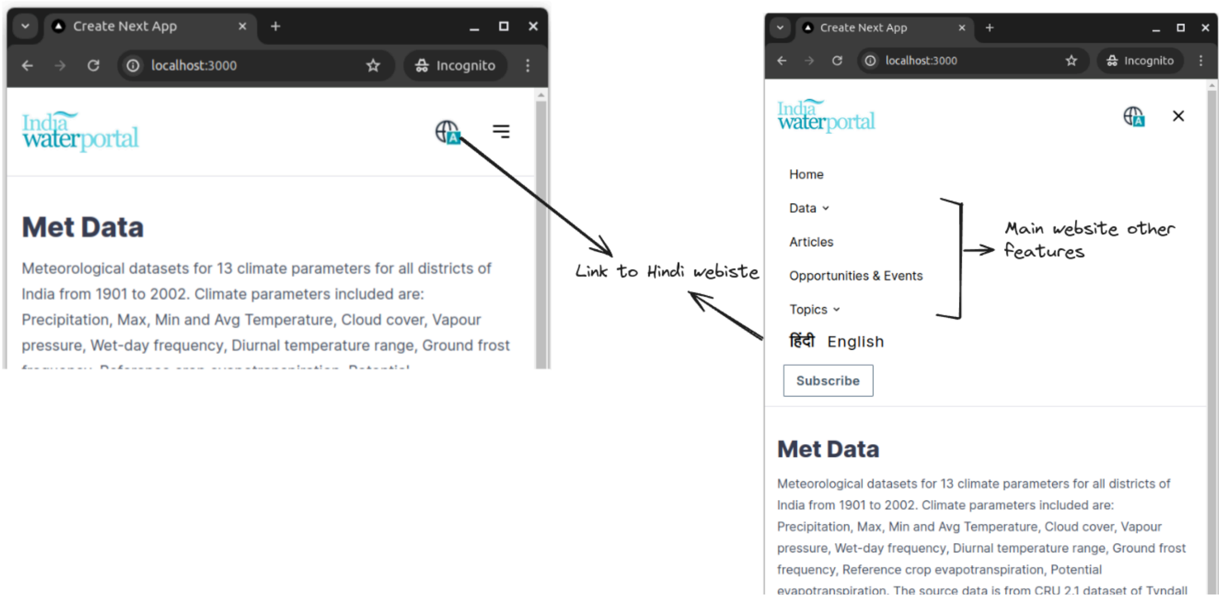
Met Data

Meteorological datasets for 13 climate parameters for all districts of India from 1901 to 2002.

Mobile View

Mobile View of the Navbar

Dropdown



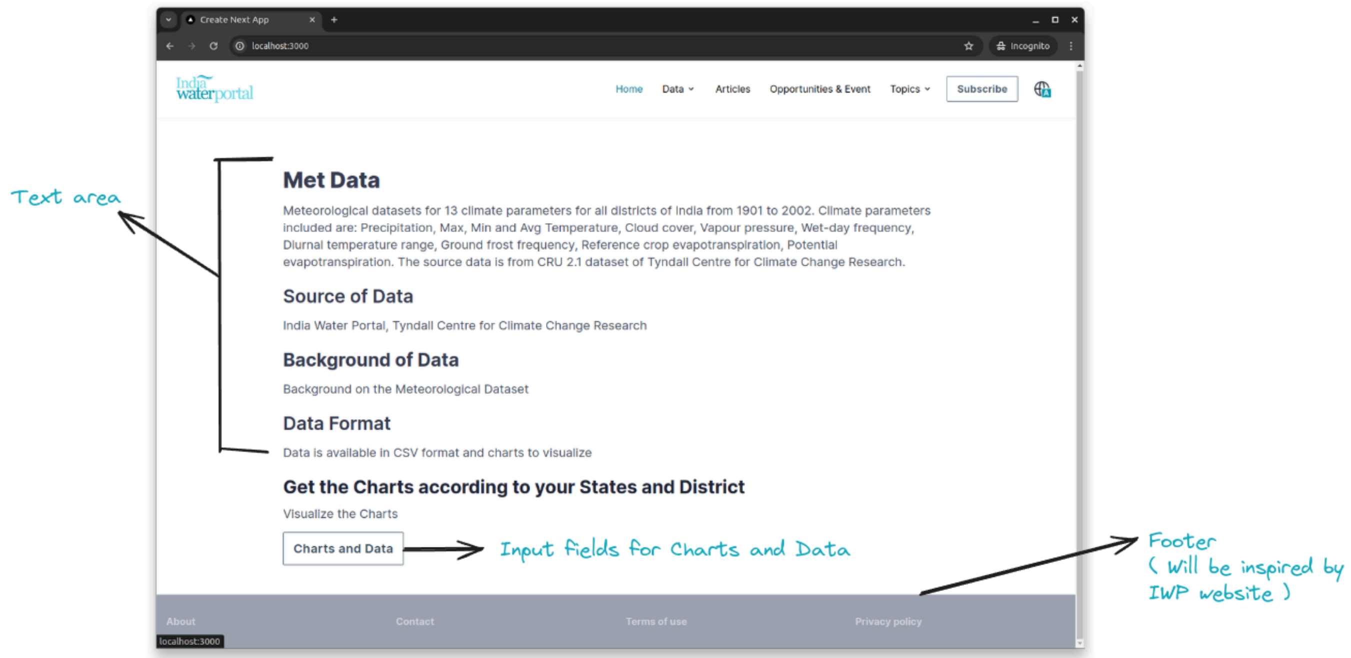
Met Data

Meteorological datasets for 13 climate parameters for all districts of India from 1901 to 2002. Climate parameters included are: Precipitation, Max, Min and Avg Temperature, Cloud cover, Vapour pressure, Wet-day frequency, Diurnal temperature range, Ground frost

Met Data

Meteorological datasets for 13 climate parameters for all districts of India from 1901 to 2002. Climate parameters included are: Precipitation, Max, Min and Avg Temperature, Cloud cover, Vapour pressure, Wet-day frequency, Diurnal temperature range, Ground frost frequency, Reference crop evapotranspiration, Potential evapotranspiration. The source data is from CRU 2.1 dataset of Tyndall

- **Homepage :** route: "/"



Met Data

Meteorological datasets for 13 climate parameters for all districts of India from 1901 to 2002. Climate parameters included are: Precipitation, Max, Min and Avg Temperature, Cloud cover, Vapour pressure, Wet-day frequency, Diurnal temperature range, Ground frost frequency, Reference crop evapotranspiration, Potential evapotranspiration. The source data is from CRU 2.1 dataset of Tyndall Centre for Climate Change Research.

Source of Data

India Water Portal, Tyndall Centre for Climate Change Research

Background of Data

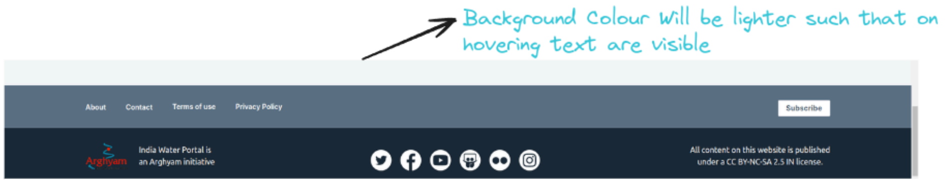
Background on the Meteorological Dataset

Data Format

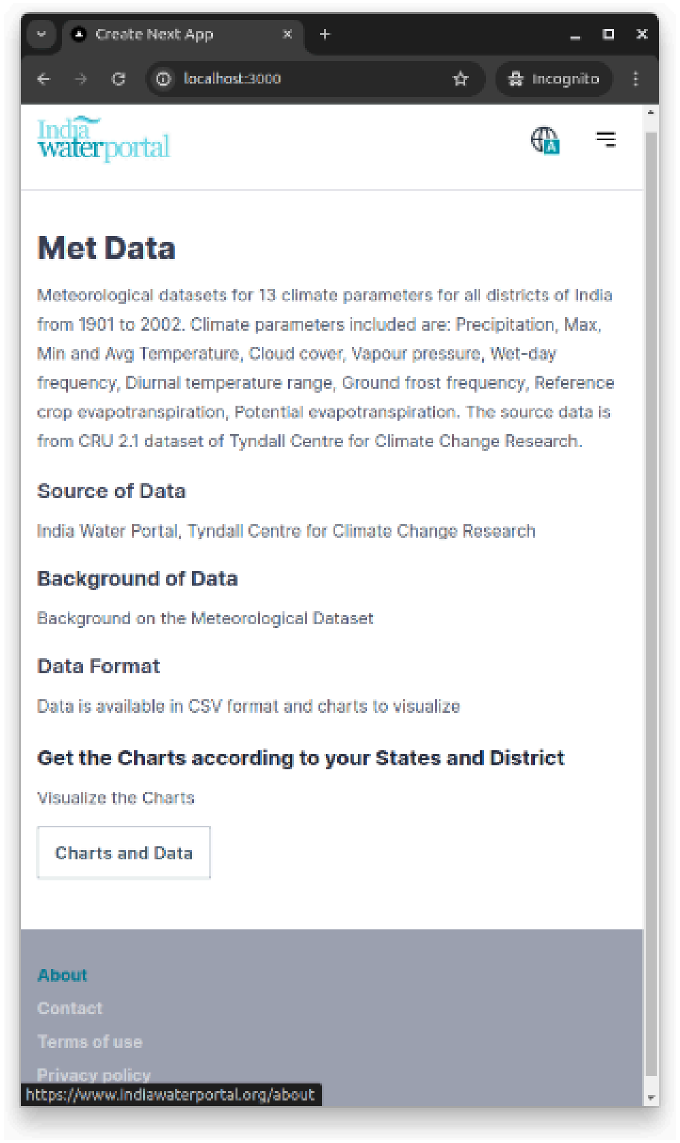
Data is available in CSV format and charts to visualize

Get the Charts according to your States and District

Visualize the Charts


Background Colour will be lighter such that on hovering text are visible
About
Contact
Terms of use
Privacy Policy
Subscribe
Arghyam
India Water Portal is an Arghyam initiative
All content on this website is published under a CC BY-NC-SA 2.5 IN license.

Mobile View



Met Data

Meteorological datasets for 13 climate parameters for all districts of India from 1901 to 2002. Climate parameters included are: Precipitation, Max, Min and Avg Temperature, Cloud cover, Vapour pressure, Wet-day frequency, Diurnal temperature range, Ground frost frequency, Reference crop evapotranspiration, Potential evapotranspiration. The source data is from CRU 2.1 dataset of Tyndall Centre for Climate Change Research.

Source of Data

India Water Portal, Tyndall Centre for Climate Change Research

Background of Data

Background on the Meteorological Dataset

Data Format

Data is available in CSV format and charts to visualize

Get the Charts according to your States and District

Visualize the Charts

Charts and Data

About
Contact
Terms of use
Privacy policy

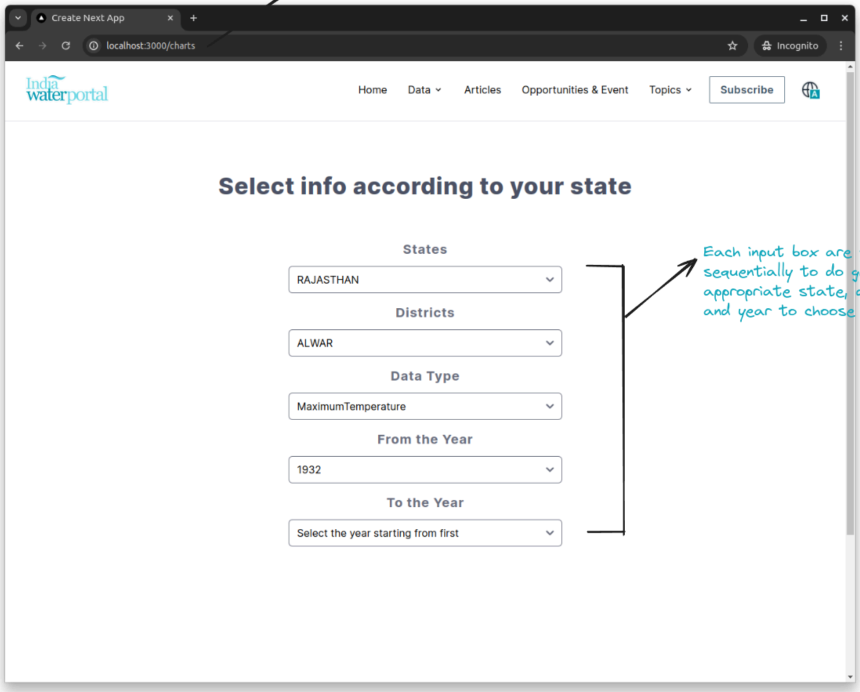
https://www.indiawaterportal.org/about

- **Trends and Charts** : route: “/charts”

Before Choosing all the options from the Input boxes

Charts and Trend page: Before Selecting To the Year

/charts route



Select info according to your state

States: RAJASTHAN

Districts: ALWAR

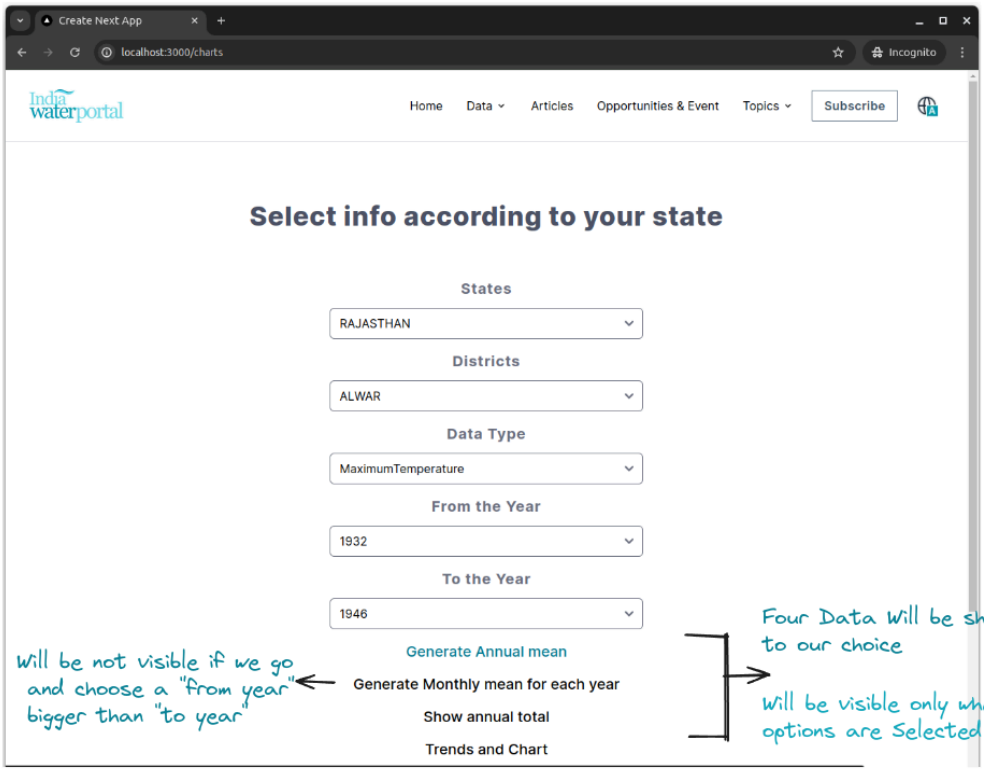
Data Type: MaximumTemperature

From the Year: 1932

To the Year: Select the year starting from first

Each input box are selected sequentially to do get the appropriate state, district and year to choose

After choosing all the Input options



Select info according to your state

States: RAJASTHAN

Districts: ALWAR

Data Type: MaximumTemperature

From the Year: 1932

To the Year: 1946

Generate Annual mean

Generate Monthly mean for each year

Show annual total

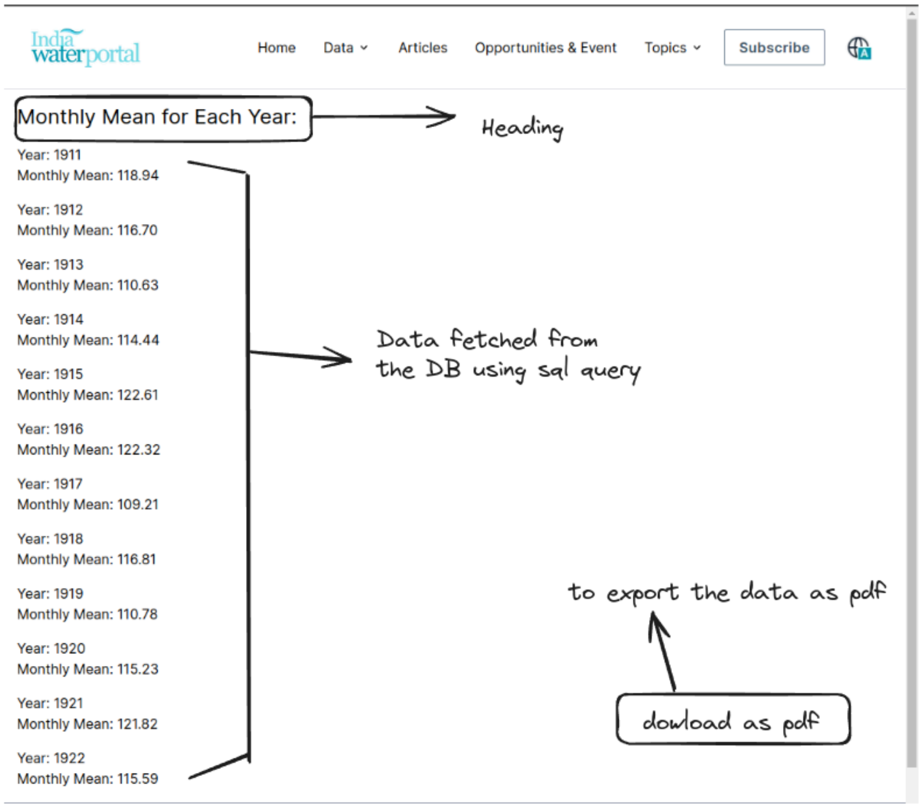
Trends and Chart

Will be not visible if we go and choose a "from year" bigger than "to year"

Four Data Will be shown acc. to our choice

Will be visible only when all the options are Selected

- **Monthly Mean:** route: “/charts/MonthlyMean”



Monthly Mean for Each Year:

Year: 1911
Monthly Mean: 118.94

Year: 1912
Monthly Mean: 116.70

Year: 1913
Monthly Mean: 110.63

Year: 1914
Monthly Mean: 114.44

Year: 1915
Monthly Mean: 122.61

Year: 1916
Monthly Mean: 122.32

Year: 1917
Monthly Mean: 109.21

Year: 1918
Monthly Mean: 116.81

Year: 1919
Monthly Mean: 110.78

Year: 1920
Monthly Mean: 115.23

Year: 1921
Monthly Mean: 121.82

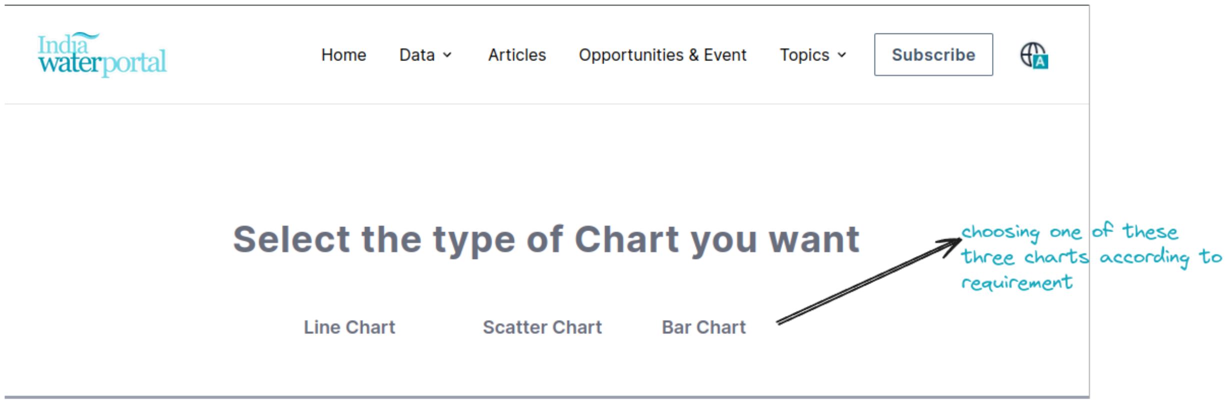
Year: 1922
Monthly Mean: 115.59

Annual Mean and Annual Total have similar design

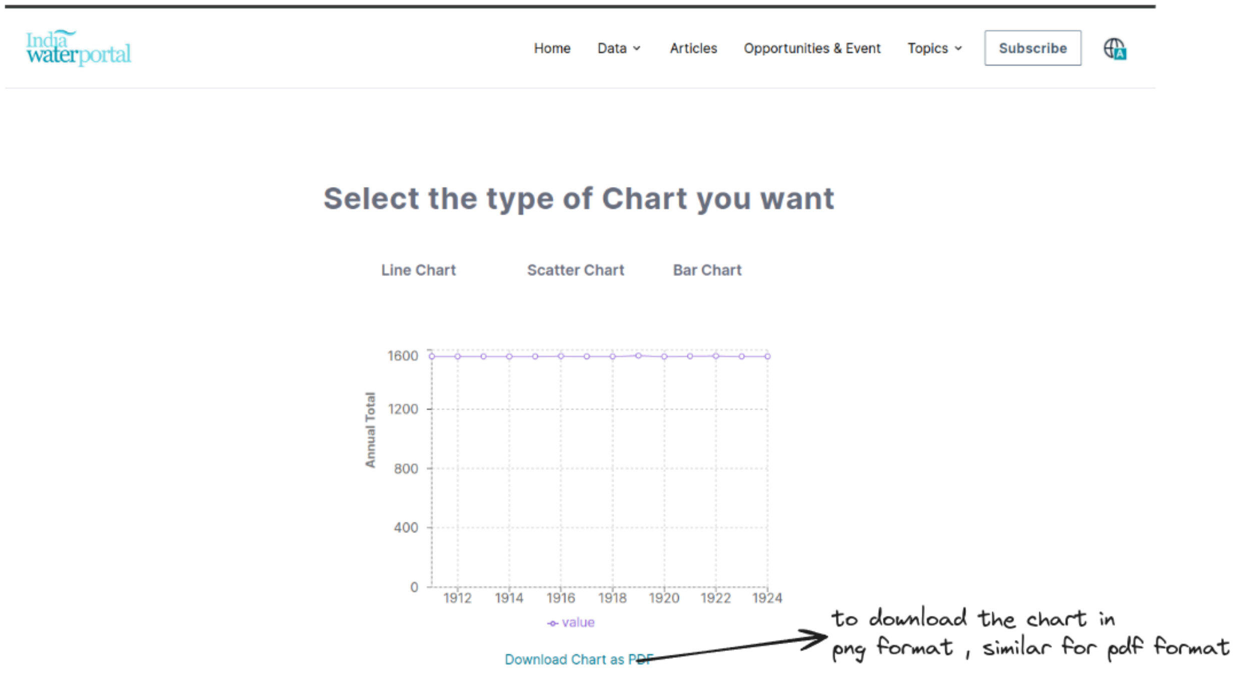
- **Charts:** route: “/charts/Trends”

In charts page we will allow three type of charts : Line chart, Bar chart and Scatter chart

Before choosing one of the three charts the page will be like



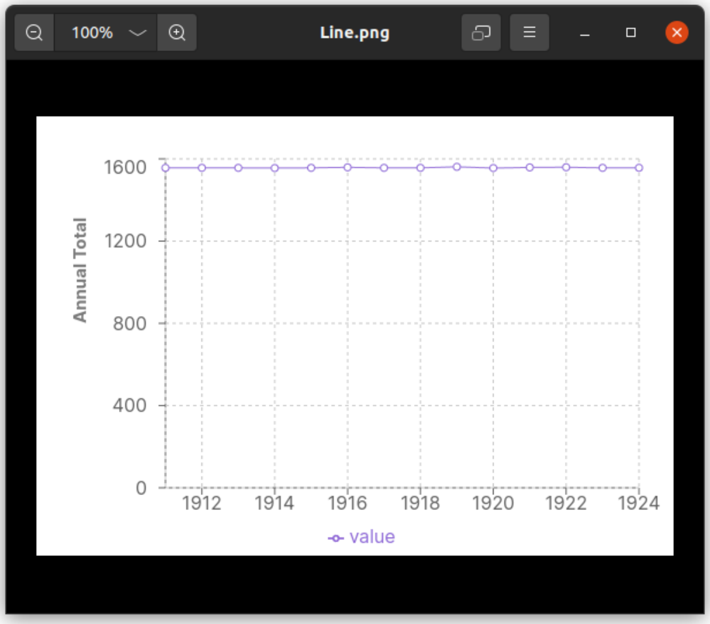
After choosing the type of chart



Select the type of Chart you want

to download the chart in png format , similar for pdf format

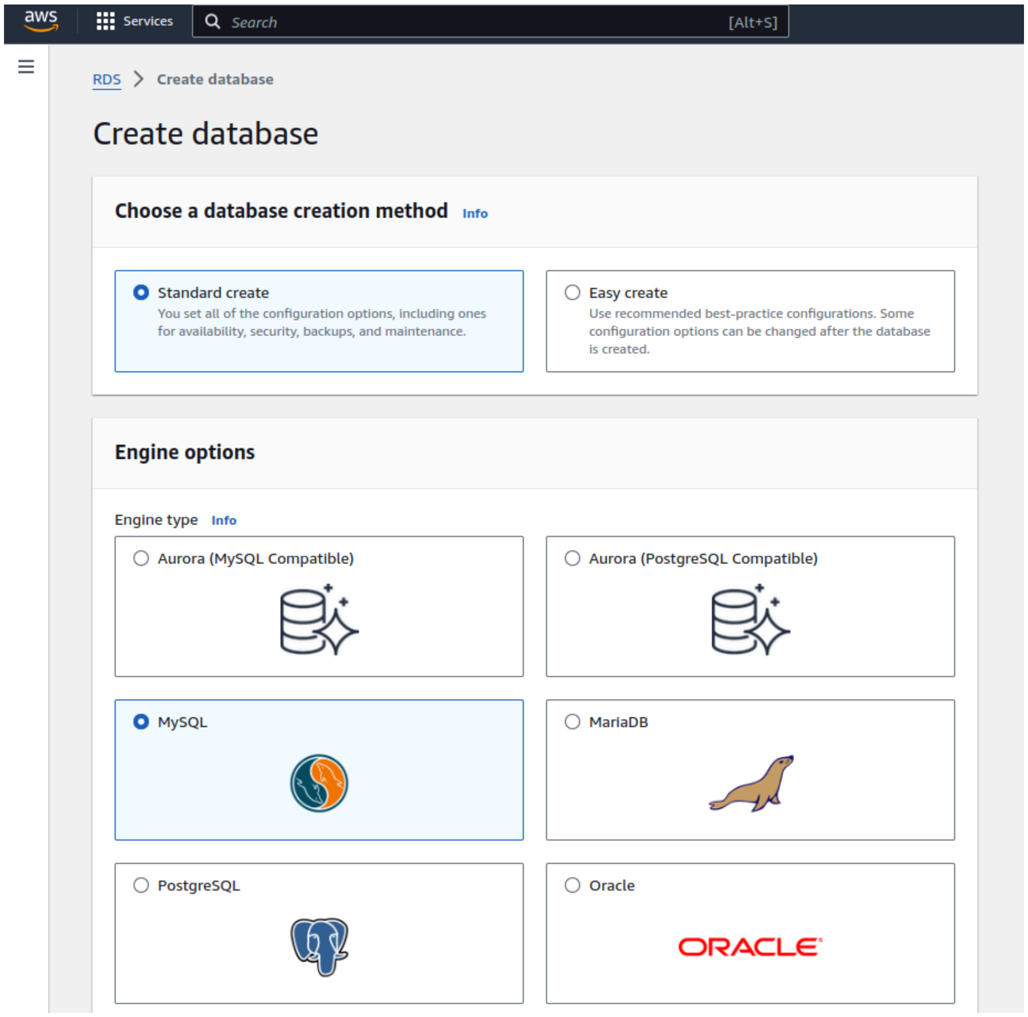
img download



Project Implementation

We will first create an AWS RDS mySql Database

https://aws.amazon.com/rds/



Create database

Choose a database creation method Info

[✓ Standard create] You set all of the configuration options, including ones for availability, security, backups, and maintenance.

Easy create: Use recommended best-practice configurations. Some configuration options can be changed after the database is created.

Engine options

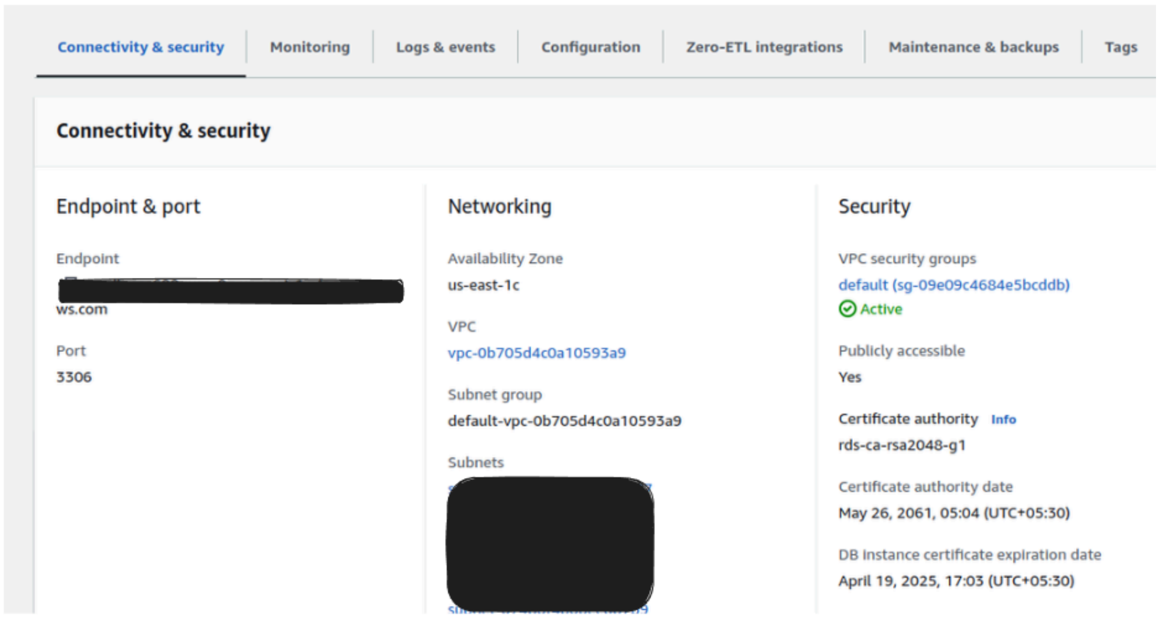
Engine type Info

Aurora (MySQL Compatible) · Aurora (PostgreSQL Compatible) · [✓ MySQL] · MariaDB · PostgreSQL · Oracle

Remember the username and password with which the Database was created.

Select on the Database once it’s status is available

Go to Connectivity and Security and store the Endpoint & Port it will be used to connect to DB


Connectivity & security | Monitoring | Logs & events | Configuration | Zero-ETL integrations | Maintenance & backups | Tags

Connectivity & security

Endpoint & port

Endpoint
[redacted]
ws.com

Port
3306

Networking

Availability Zone
us-east-1c

VPC
vpc-0b705d4c0a10593a9

Subnet group
default-vpc-0b705d4c0a10593a9

Subnets
[redacted]

Security

VPC security groups
default (sg-09e09c4684e5bcddb)
Active

Publicly accessible
Yes

Certificate authority Info
rds-ca-rsa2048-g1

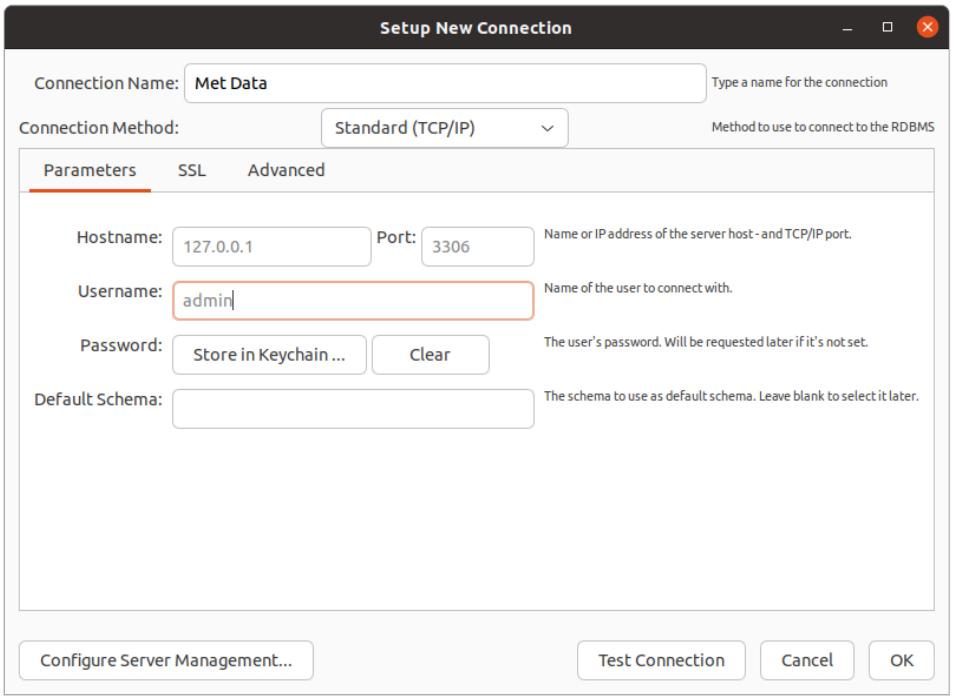
Certificate authority date
May 26, 2061, 05:04 (UTC+05:30)

DB instance certificate expiration date
April 19, 2025, 17:03 (UTC+05:30)

We will use MySQL Workbench as a visualization tool for our database.

Lets connect our AWS RDS database to MySql workbench , We will use it to define our database and table schema and import Csv files in the defined schema.
Csv file for our Project:
https://drive.google.com/drive/folders/1ewF5fNSFsg3v_Z7BZ_oNtCUQORK2J3uo

Establishing MySQL connection in MySQL Workbench


Setup New Connection

Connection Name: Met Data — Type a name for the connection

Connection Method: Standard (TCP/IP) — Method to use to connect to the RDBMS

Parameters | SSL | Advanced

Hostname: 127.0.0.1 Port: 3306 — Name or IP address of the server host - and TCP/IP port.

Username: admin — Name of the user to connect with.

Password: Store in Keychain ... Clear — The user's password. Will be requested later if it's not set.

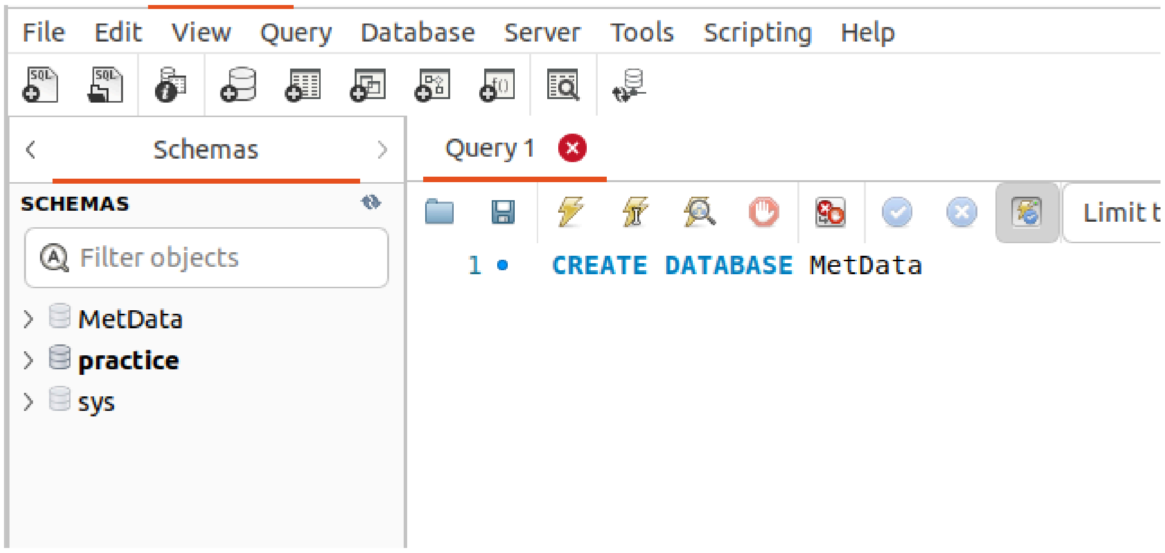
Default Schema: — The schema to use as default schema. Leave blank to select it later.

Configure Server Management... Test Connection Cancel OK

Open the this Connection in Workbench

Creating database called MetData for our Project inside it we will store our Tables

```
CREATE DATABASE MetData;
```



```
CREATE DATABASE MetData
```

Now we will Define Table Schema one by one for all 12 csv files.

- Average Temperature

- Cloud Cover
- Diurnal Temperature
- Ground Frost Frequency
- Maximum Temperature
- Minimum Temperature
- Potential Evapotranspiration
- Precipitation
- Precipitation_2004_2011
- Reference Crop Evapotranspirati
- Vapour Pressure
- Wet Day Frequency

Let see an example of a Precipitation table

```
CREATE TABLE MetData.Precipitation (
    year_val INT NOT NULL,
    stateid INT NOT NULL,
    state_dist_key INT NOT NULL,
    State TEXT NOT NULL,
    sep DOUBLE NOT NULL,
    oct DOUBLE NOT NULL,
    nov DOUBLE NOT NULL,
    may DOUBLE NOT NULL,
    mar DOUBLE NOT NULL,
    jun DOUBLE NOT NULL,
    jul DOUBLE NOT NULL,
    jan DOUBLE NOT NULL,
    feb DOUBLE NOT NULL,
    districtid INT NOT NULL,
    District TEXT NOT NULL,
    dec DOUBLE NOT NULL,
    aug DOUBLE NOT NULL,
    apr DOUBLE NOT NULL
);
```

All the table except Precipitation_2004_2011 will have same schema

Let import data from Precipitation.csv to our Precipitation table

```
LOAD DATA LOCAL INFILE '/home/ntc/Desktop/C4GT_Projects/Met
Data/Precipitation.csv'
INTO TABLE practice.Precipitation
FIELDS TERMINATED BY ',' ENCLOSED BY '"'
LINES TERMINATED BY '\n'
IGNORE 1 ROWS; -- Skip the header row if it exists
```

Lets see the Schema for Precipitation_2004_2011

```
CREATE TABLE MetData.Precipitation (
   state TEXT NOT NULL,
   district TEXT NOT NULL,
   year_val INT NOT NULL,
   jan DOUBLE NOT NULL,
   feb DOUBLE NOT NULL,
   mar DOUBLE NOT NULL,
   apr DOUBLE NOT NULL,
   may DOUBLE NOT NULL,
   jun DOUBLE NOT NULL,
   jul DOUBLE NOT NULL,
   aug DOUBLE NOT NULL,
   sep DOUBLE NOT NULL,
   oct DOUBLE NOT NULL,
   nov DOUBLE NOT NULL,
   dec DOUBLE NOT NULL,
   AnnualTotal DOUBLE NOT NULL
);
```

After importing the csv file we

Hence we will implement the Data cleaning from the csv files

Some of examples need to be

- Truncate extra column from the tables
- Adding **State** column in Potential Evapotranspiration.csv
- Maintaining a common schema for all the 12 tables except the Precipitation_2004_2011 table

Our database is setup , now we will connect the database to backend

Backend

npm init -y

npm install express mysql nodemon cors

npm install @types/mysql @types/node

npm install -g typescript

npx tsc –init

Change tsconfig to this

```
{
 "compilerOptions": {
   "target": "es2016",
   "module": "commonjs",
   "rootDir": "./src",
   "outDir": "./dist",
   "esModuleInterop": true,
   "forceConsistentCasingInFileNames": true,
   "strict": true,
   "skipLibCheck": true
 }
}
```

Make a .gitignore file

```
node_modules
dist
```

Make a **src** folder inside it form a file called index.ts and a folder called types , this folder will store the types required in the project

Add script to package. json

```
"scripts": {
   "test": "echo \"Error: no test specified\" && exit 1",
   "build": "tsc -b",
   "start": "nodemon dist/index.js"
 },
```

Define type inside types/index.ts

We need to define types for two type one for Precipitation_2004_2011 and one for rest of the tables.

```

export interface TablesTypes {
 stateid: number;
 districtid: number;
 State: string;
 state_dist_key: string;
 District: string;
 year_val: number;
 jan: number;
 feb: number;
 mar: number;
 apr: number;
 may: number;
 jun: number;
 jul: number;
 aug: number;
 sep: number;
 oct: number;
 nov: number;
 dec: number;
}

export interface Precipitation_2004_2011 {
 State: "text";
 District: "text";
 year_val: "int";
 January: "double";
 February: "double";
 March: "double";
 April: "double";
 May: "double";
 June: "double";
 July: "double";
 August: "double";
 September: "double";
 October: "double";
 November: "double";
 December: "double";
 AnnualTotal: "double";
}
```

Importing the required packages/ and setting cors in src/index.ts

```
const express = require('express')
import { Request,Response } from "express"
const mysql = require('mysql')
const cors = require('cors')
import { WeatherData } from "./types"
import { Precipitation_2004_2011 } from "./types"
const app = express()
app.use(cors())
```

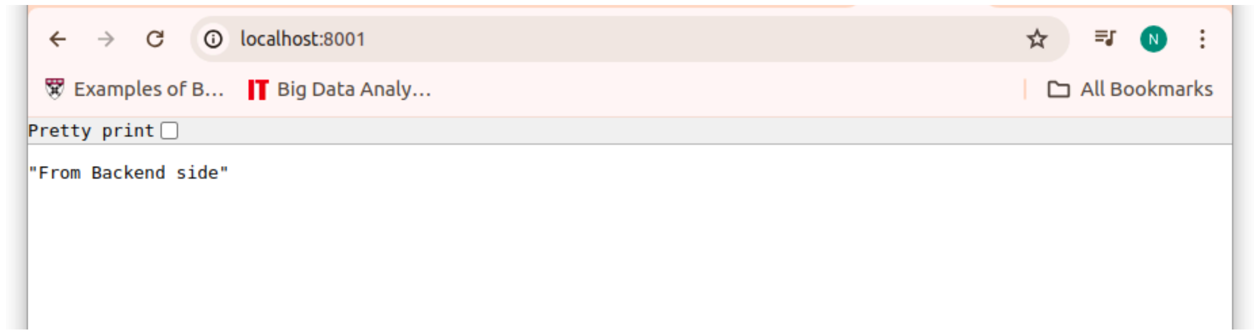
Making .env file

```
DB_URL="mydb.aacsa35sd.us-west-1.rds.amazonaws.com"
DB_User="admin"
DB_password="12345678"
DB_database="practice"
# Ask for the DB_URL
```

Checking backend running or not

```
const express = require('express')
import { Request,Response } from "express"
```

```
const mysql = require('mysql')
const cors = require('cors')
import { TableTypes } from "./types"
import { Precipitation_2004_2011 } from "./types"
const app = express()
app.use(cors())

app.get('/',(req:Request,res:Response)=>{
   return res.json("From Backend side")
})

app.listen(8001,()=>{
   console.log("listening")
})
```



```
"From Backend side"
```

Connecting to the Database

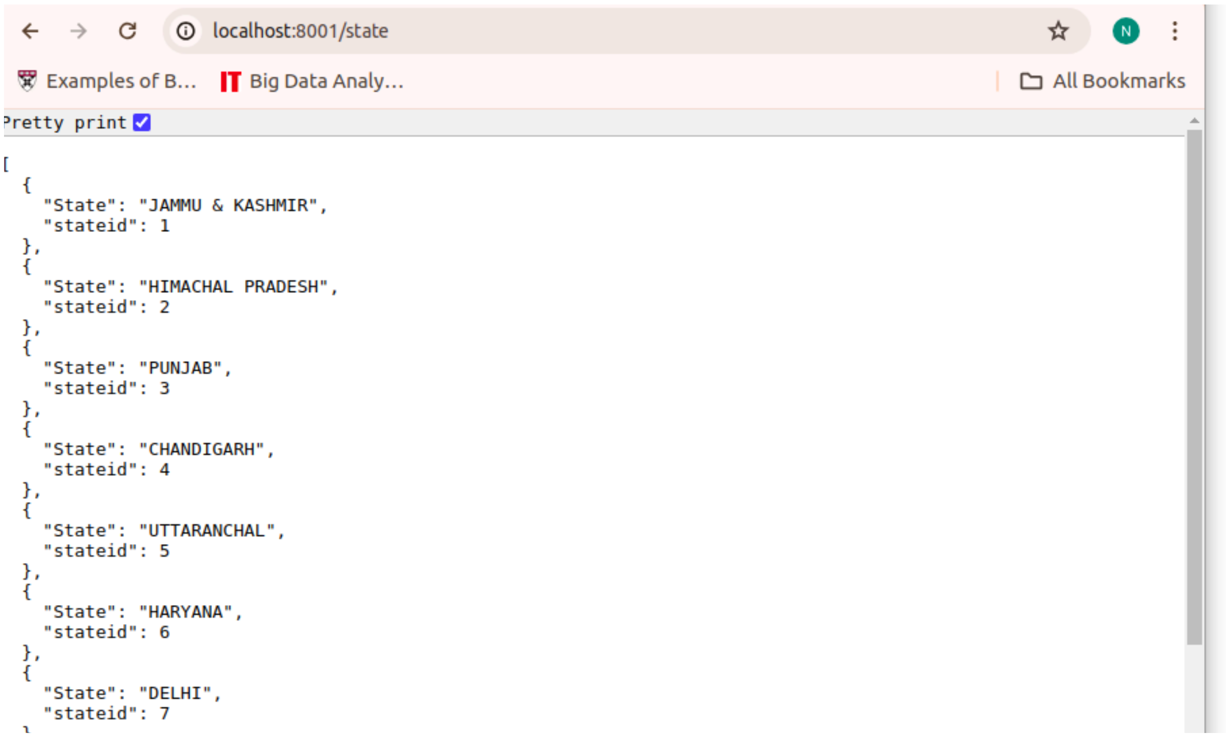
npm install dotenv

```
require('dotenv').config();
```

```
const db = mysql.createConnection({
   host:process.env.DB_URL,
   user:process.env.DB_User,
   password:process.env.DB_password,
   database:process.env.DB_database,
})
```

Testing DB connection with a **"/state"** endpoint

```
app.get('/state',(req:Request,res:Response)=>{
   const sql = "SELECT DISTINCT State,stateid FROM
practice.WetDayFrequency ORDER BY stateid ASC;"
   db.query(sql,(err:Error|null,data:TableTypes[]) => {
       if(err) return res.json(err);
       return res.json(data)
   })
})
```



localhost:8001/state

Examples of B... Big Data Analy... All Bookmarks

Pretty print ✓

```
[
  {
    "State": "JAMMU & KASHMIR",
    "stateid": 1
  },
  {
    "State": "HIMACHAL PRADESH",
    "stateid": 2
  },
  {
    "State": "PUNJAB",
    "stateid": 3
  },
  {
    "State": "CHANDIGARH",
    "stateid": 4
  },
  {
    "State": "UTTARANCHAL",
    "stateid": 5
  },
  {
    "State": "HARYANA",
    "stateid": 6
  },
  {
    "State": "DELHI",
    "stateid": 7
```

Add .github/workflows/build.yml for continous integration and running test whenever there is any pull request to the master branch

```
name: Build Succeds on PR

on:
   pull_request:
       branches:
           - master


jobs:
   build:
       name: Build the project
       runs-on: ubuntu-latest
       steps:
           - uses: actions/checkout/@v3
           - name: Use Node.js
             uses: actions/setup-node@v3
             with:
               node-version: '20'

           - name: Install Dependencies
             run: npm install

           - name: Run Build
             run: npm run build

           - name: Run Test
             run: npm test --if-present
```

We will now add some sample test and then create a pull request to check whether CI is working or not

npm install –save-dev jest

npm install --save-dev

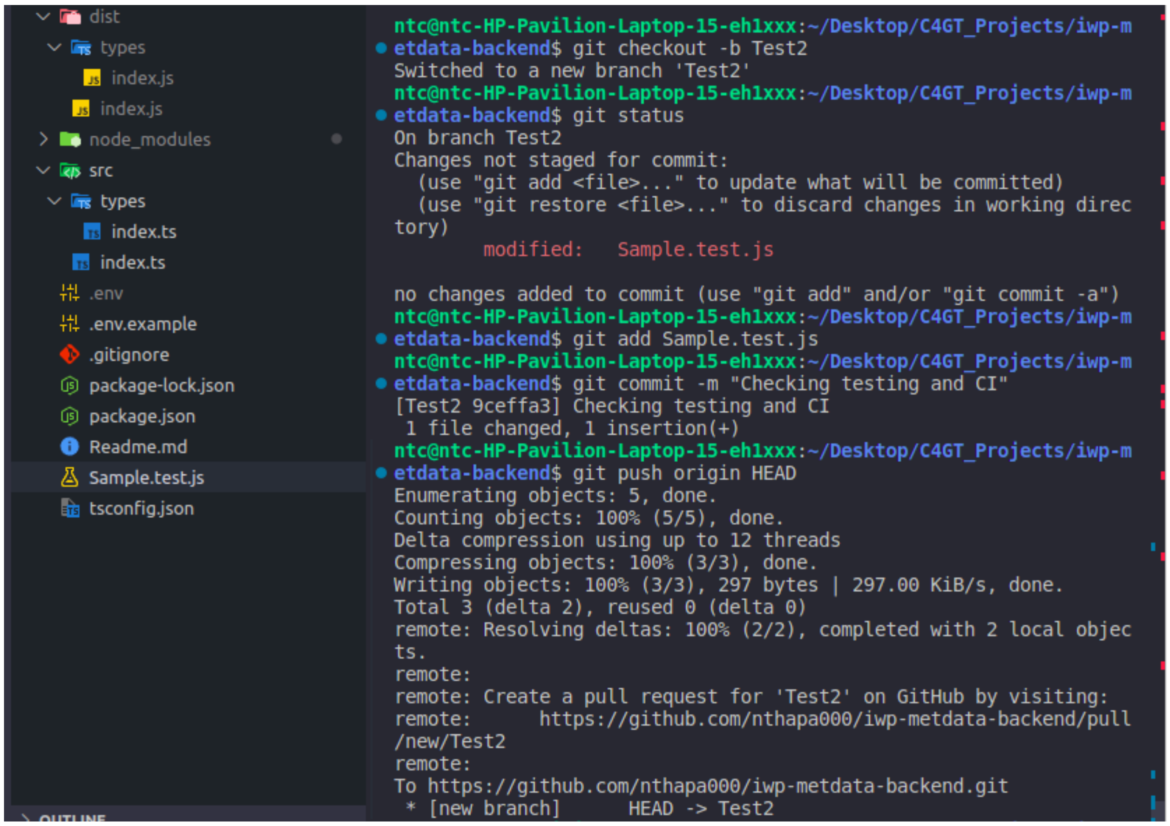
npm install supertest

Change package.json Add a script

“test”:”jest”

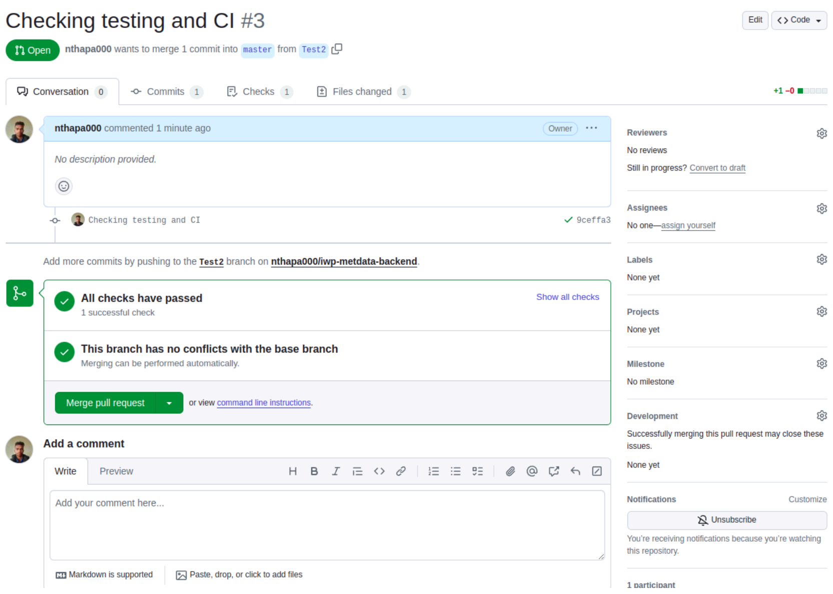
Adding sample test

```
function add(a, b) {
   return a + b;
}

describe('add function', () => {
   it('should return the sum of two numbers', () => {
       const result = add(2, 3);
       expect(result).toBe(5);
   });
});
```



```
ntc@ntc-HP-Pavilion-Laptop-15-eh1xxx:~/Desktop/C4GT_Projects/iwp-m
etdata-backend$ git checkout -b Test2
Switched to a new branch 'Test2'
ntc@ntc-HP-Pavilion-Laptop-15-eh1xxx:~/Desktop/C4GT_Projects/iwp-m
etdata-backend$ git status
On branch Test2
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working direc
tory)
        modified:   Sample.test.js

no changes added to commit (use "git add" and/or "git commit -a")
ntc@ntc-HP-Pavilion-Laptop-15-eh1xxx:~/Desktop/C4GT_Projects/iwp-m
etdata-backend$ git add Sample.test.js
ntc@ntc-HP-Pavilion-Laptop-15-eh1xxx:~/Desktop/C4GT_Projects/iwp-m
etdata-backend$ git commit -m "Checking testing and CI"
[Test2 9ceffa3] Checking testing and CI
 1 file changed, 1 insertion(+)
ntc@ntc-HP-Pavilion-Laptop-15-eh1xxx:~/Desktop/C4GT_Projects/iwp-m
etdata-backend$ git push origin HEAD
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 12 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 297 bytes | 297.00 KiB/s, done.
Total 3 (delta 2), reused 0 (delta 0)
remote: Resolving deltas: 100% (2/2), completed with 2 local objec
ts.
remote:
remote: Create a pull request for 'Test2' on GitHub by visiting:
remote:      https://github.com/nthapa000/iwp-metdata-backend/pull
/new/Test2
remote:
To https://github.com/nthapa000/iwp-metdata-backend.git
 * [new branch]      HEAD -> Test2
```

Creating a pull request



Checking testing and CI #3

nthapa000 wants to merge 1 commit into master from Test2

nthapa000 commented 1 minute ago

No description provided.

Checking testing and CI 9ceffa3

Add more commits by pushing to the Test2 branch on nthapa000/iwp-metdata-backend.

All checks have passed
1 successful check

This branch has no conflicts with the base branch
Merging can be performed automatically.

or view command line instructions.

Add a comment

Add your comment here...

Markdown is supported | Paste, drop, or click to add files

Reviewers
No reviews
Still in progress? Convert to draft

Assignees
No one—assign yourself

Labels
None yet

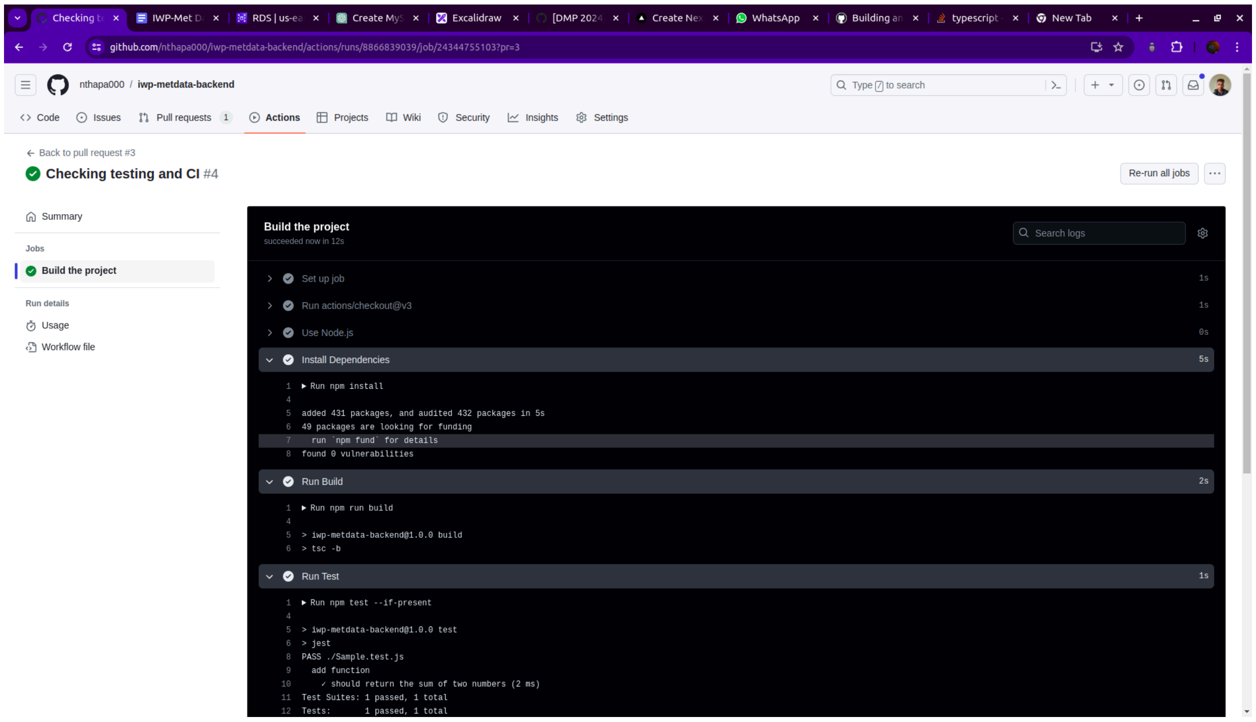
Projects
None yet

Milestone
No milestone

Development
Successfully merging this pull request may close these issues.
None yet

Notifications
You're receiving notifications because you're watching this repository.

1 participant



Build the project
succeeded now in 12s

```
Set up job                         1s
Run actions/checkout@v3            1s
Use Node.js                        0s
Install Dependencies               5s
▸ Run npm install

added 431 packages, and audited 432 packages in 5s
49 packages are looking for funding
  run `npm fund` for details
found 0 vulnerabilities
Run Build                          2s
▸ Run npm run build

> iwp-metdata-backend@1.0.0 build
> tsc -b
Run Test                           1s
▸ Run npm test --if-present

> iwp-metdata-backend@1.0.0 test
> jest
PASS ./Sample.test.js
  add function
    ✓ should return the sum of two numbers (2 ms)
Test Suites: 1 passed, 1 total
Tests:       1 passed, 1 total
```

Adding rest of endpoints

“/district” endpoint: return all the district of all states

```
app.get('/distict', (req:Request,res:Response) => {
   const stateid = req.query.Stateid; // Extract stateid from query 
parameters
   const sql = `SELECT DISTINCT Distict,districtid FROM 
practice.WetDayFrequency WHERE stateid = ${stateid} order by districtid;`;
   db.query(sql, stateid, (err:Error|null,data:TableTypes[]) => {
       if (err) return res.json(err);
       return res.json(data);
   });
});
```

“/year” endpoint: return all the year of the data of selected Data Type

```
app.get('/year', (req:Request,res:Response) => {
   const tablename = req.query.tablename; // Extract stateid from query 
parameters
```

```
   const sql = `SELECT DISTINCT year_val FROM ${tablename} order by 
year_val`;
   if(tablename=="Precipitation_2004_2011"){
       db.query(sql, tablename, (err:Error|null, 
data:Precipitation_2004_2011[]) => {
           if (err) return res.json(err);
           return res.json(data);
       });
   }else{
       db.query(sql, tablename, (err:Error|null, data:TableTypes[]) => {
           if (err) return res.json(err);
           return res.json(data);
       });
   }
});
```

"/yearAfter" endpoint: To return years after "FromYearSelected"

```
app.get('/yearAfter', (req:Request,res:Response) => {
   const tablename = req.query.tablename; // Extract stateid from query 
parameters
   const selectedFromYear = req.query.selectedFromYear;
   const sql =`SELECT DISTINCT year_val FROM ${tablename} WHERE year_val 
>= ${selectedFromYear} ORDER BY year_val`;

  if(tablename=="Precipitation_2004_2011"){
       db.query(sql, tablename, (err:Error|null, 
data:Precipitation_2004_2011[]) => {
           if (err) return res.json(err);
           return res.json(data);
       });
   }else{
       db.query(sql, tablename, (err:Error|null, data:TableTypes[]) => {
           if (err) return res.json(err);
           return res.json(data);
       });
   }
});
```

“/final” endpoint: return all the data required in FE, MonthlyMean AnnualMean, AnnualTotal and Trends route for all table except Precipitation_2004_2011

```
app.get('/final', (req:Request,res:Response) => {
   const tablename = req.query.tablename; // Extract stateid from query
parameters
   const selectedFromYear = req.query.selectedFromYear;
   const selectedToYear = req.query.selectedToYear;
   const districtid = req.query.districtid;
   const stateid = req.query.stateid;
   const sql =`SELECT *
   FROM ${tablename}
   WHERE year_val BETWEEN ${selectedFromYear} AND ${selectedToYear}
   AND stateid = ${stateid}
   AND districtid = ${districtid}`;

  if(tablename=="Precipitation_2004_2011"){
       db.query(sql, tablename, (err:Error|null,
data:Precipitation_2004_2011[]) => {
           if (err) return res.json(err);
           return res.json(data);
       });
   }else{
       db.query(sql, tablename, (err:Error|null, data:TableTypes[]) => {
           if (err) return res.json(err);
           return res.json(data);
       });
   }
});
```

“final2” endpoint : this endpoint for Precipitation_2004_2011 table

```
app.get('/final2', (req:Request,res:Response) => {
   const tablename = req.query.tablename; // Extract stateid from query
parameters
   const selectedFromYear = req.query.selectedFromYear;
   const selectedToYear = req.query.selectedToYear;
   const district = req.query.District;
   const state = req.query.State;
   const sql =`SELECT *
   FROM ${tablename}
   WHERE year_val BETWEEN ${selectedFromYear} AND ${selectedToYear}
   AND State = ${state}
   AND District = ${district}`;

       db.query(sql, tablename, (err:Error|null,
data:Precipitation_2004_2011[]) => {
           if (err) return res.json(err);
           return res.json(data);

   })
});
```

Frontend

npx create-next-app@latest

Select typescript, eslint, app router and tailwind

Adding **.github/workflow/build.yml** to ensure that each new feature does get build and doesn't break the application when merge with master branch

```
name: Build Succeds on PR

on:
   pull_request:
       branches:
           - master

jobs:
   build:
       name: Build the project
       runs-on: ubuntu-latest
       steps:
           - uses: actions/checkout/@v3
           - name: Use Node.js
```

```yaml
            uses: actions/setup-node@v3
            with:
              node-version: '20'

          - name: Install Dependencies
            run: npm install

          - name: Run Build
            run: npm run build
```

Later we will add test also such that a feature will be merge to master branch only when it doesn’t violate the test

Go to app/layout.tsx add metadata and import two component name Header and Footer which will me common in all pages

```tsx
import type { Metadata } from "next";
import { Inter } from "next/font/google";
import "./globals.css";
import Header from "@/components/shared/Header";
import Footer from "@/components/shared/Footer";

const inter = Inter({ subsets: ["latin"] });

export const metadata: Metadata = {
 title: "India Water Portal - Met Data For Everyone",
 description: "Visualize and Analyze the data collected over 100 years for
more than 12 meterological parameters",
};

export default function RootLayout({
 children,
}: Readonly<{
 children: React.ReactNode;
}>) {
 return (
   <html lang="en">
     <body className={inter.className}>
     <Header />
       {children}
```

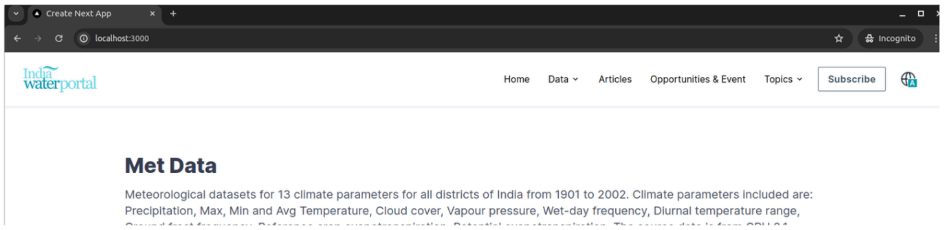
```
    <Footer />
    </body>
  </html>
 );
}
```

Header.tsx

We can take reference from the UI/ UX design mentioned above

We will first add the Desktop View along with the Dropdown

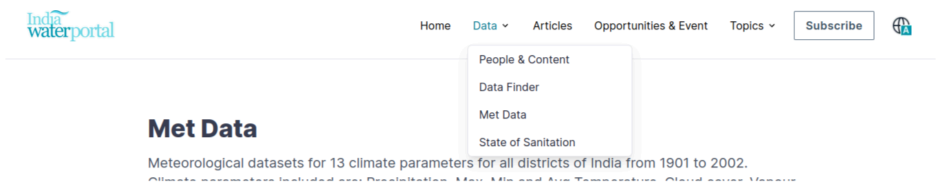
Desktop View of the Navbar



Met Data

Meteorological datasets for 13 climate parameters for all districts of India from 1901 to 2002. Climate parameters included are: Precipitation, Max, Min and Avg Temperature, Cloud cover, Vapour pressure, Wet-day frequency, Diurnal temperature range,

Dropdown



Met Data

Meteorological datasets for 13 climate parameters for all districts of India from 1901 to 2002.

```
"use client";
import React, { useState } from "react";
import LogoImage from "../utils/LogoImage";
import DataDropDownDesktop from "../utils/DataDropDownDesktop";
import TopicsDesktop from "../utils/TopicsDesktop";
import Button from "../utils/Button";
import Language from "../utils/Language";
import MobileDropdown from "../utils/MobileDropdown";
import Link from "next/link";
import Image from "next/image";

const Header = () => {
 const [isClick, setIsClick] = useState(false);
 const toggleNavbar = () => {
   setIsClick(!isClick);
 };
 return (
   <nav className="border-b">
     <div className="max-w-9xl m-4 mx-auto px-4 sm:px-6 lg:px-8">
       <div className="flex items-center justify-between h-16">
         <LogoImage />
         <div className="hidden lg:block">
           <div className="ml-4 flex items-center space-x-4">
             <a href="/" className="arghyam p-2">
               Home
             </a>
             <DataDropDownDesktop />
             <a href="/" className="arghyam p-2">
               Articles
             </a>
             <a href="/" className="arghyam p-2">
               Opportunities & Event
             </a>
             <TopicsDesktop />
             <Button label="Subscribe" />
             <Language />
           </div>
         </div>
         <div className="lg:hidden flex items-center justify-end
space-x-4">
           <Link href="https://hindi.indiawaterportal.org/"
className="p-2">
             <Image src={"/global.png"} width={30} height={20} alt="IWF"
/>
           </Link>{" "}
           <div>
             <button
               className="inline-flex items-center justify-center p-2
rounded-md"
               onClick={toggleNavbar}
             >
               {isClick ? (
                 <svg
                   className="h-6 w-6"
                   xmlns="http://www.w3.org/2000/svg"
                   fill="none"
                   viewBox="0 0 24 24"
                   stroke="currentColor"
                 >
                   <path
                     strokeLinecap="round"
                     strokeLinejoin="round"
                     strokeWidth={2}
                     d="M6 18L18 6M6 6l12 12"
                   />
                 </svg>
               ) : (
                 <svg
                   className="h-6 w-6"
                   xmlns="http://www.w3.org/2000/svg"
                   fill="none"
                   viewBox="0 0 24 24"
                   stroke="currentColor"
                 >
                   <path
                     strokeLinecap="round"
                     strokeLinejoin="round"
                     strokeWidth={2}
                     d="M4 6h16M4 12h16m-7 6h7"
                   />
                 </svg>
               )}
             </button>
           </div>
         </div>
       </div>
     </div>
     {isClick && <MobileDropdown />}
   </nav>
 );
};

export default Header;
```

Lets see the individual Components one by one

LogoImage

```
import Image from "next/image";
import React from "react";

const LogoImage = () => {
 return (
   <div className="flex items-center">
   <div className="flex-shrink-0">
     <a href="/">
       <Image src={"/IWF.png"} width={128} height={40} alt="IWF" />
     </a>
   </div>
 </div>
 );
};

export default LogoImage;
```

DataDropDownDesktop

```
"use client";
import React, { useState } from "react";

const DataDropDownDesktop = () => {
 const [isOpen, setIsOpen] = useState(false);
 const toggleSubMenu = () => {
   setIsOpen(!isOpen);
 };
 const closeSubMenu = () => {
   setIsOpen(false);
 };

 return (
   <div className="relative">
     <a
```

```
       className="hover:text-arghyam p-2 flex items-center"
       onMouseEnter={toggleSubMenu}
       onMouseLeave={closeSubMenu}
     >
       Data
       <svg
         className="w-4 h-4 ml-1 fill-current text-gray-500"
         xmlns="http://www.w3.org/2000/svg"
         viewBox="0 0 24 24"
       >
         <path d="M16.59 8.59L12 13.17 7.41 8.59 6 10l6 6 6-6z" />
       </svg>
     </a>
     {isOpen && (
       <div className="absolute z-10 left-0 mt-2 w-60 bg-white border
border-gray-200 rounded-md shadow-lg dark:bg-gray-800
dark:border-gray-600">
         <a
           href="/"
           className="block px-4 justify-center h-full  py-2 text-gray-800
hover:bg-gray-100 dark:hover:bg-gray-700"
         >
           People & Content
         </a>
         <a
           href="/"
           className="block px-4 py-2 text-gray-800 hover:bg-gray-100
dark:hover:bg-gray-700"
         >
           Data Finder
         </a>
         <a
           href="/"
           className="block px-4 py-2 text-gray-800 hover:bg-gray-100
dark:hover:bg-gray-700"
         >
           Met Data
         </a>
         <a
           href="/"
           className="block px-4 py-2 text-gray-800 hover:bg-gray-100
dark:hover:bg-gray-700"
         >
           State of Sanitation
         </a>
       </div>
     )}
   </div>
 );
};

export default DataDropDownDesktop;
```

TopicsDesktop

```
"use client";
import React, { useState } from "react";

const TopicsDesktop = () => {
 const [isTopicOpen, setIsTopicOpen] = useState(false);
 const toggleSubMenuTopic = () => {
   setIsTopicOpen(!isTopicOpen);
 };
 const closeSubMenuTopic = () => {
   setIsTopicOpen(false);
 };
 return (
   <div className="relative ">
     <a
       className="hover:text-arghyam p-2  flex items-center"
       onMouseEnter={toggleSubMenuTopic}
       onMouseLeave={closeSubMenuTopic}
     >
       Topics
       <svg
         className="w-4 h-4 ml-1 fill-current text-gray-500"
         xmlns="http://www.w3.org/2000/svg"
         viewBox="0 0 24 24"
       >
```

```
         <path d="M16.59 8.59L12 13.17 7.41 8.59 6 10l6 6 6-6z" />
       </svg>
     </a>
     {isTopicOpen && (
       <div className="absolute z-10  left-0  mt-2 w-60 bg-white border
border-gray-200 rounded-md shadow-lg dark:bg-gray-800
dark:border-gray-600">
         <a
           href="/"
           className="block px-4 py-2 text-gray-800 hover:bg-gray-100
dark:hover:bg-gray-700"
         >
           Solid Waste
         </a>
         <a
           href="/"
           className="block px-4 py-2 text-gray-800 hover:bg-gray-100
dark:hover:bg-gray-700"
         >
           Rainwater Harvesting
         </a>
         <a
           href="/"
           className="block px-4 py-2 text-gray-800 hover:bg-gray-100
dark:hover:bg-gray-700"
         >
           Rural Sanitation
         </a>
         <a
           href="/"
           className="block px-4 py-2 text-gray-800 hover:bg-gray-100
dark:hover:bg-gray-700"
         >
           Agriculture
         </a>
         <a
           href="/"
           className="block px-4 py-2 text-gray-800 hover:bg-gray-100
dark:hover:bg-gray-700"
         >
```

```
           View all Topics
         </a>
       </div>
     )}
   </div>
 );
};

export default TopicsDesktop;
```

Button

```
import React from "react";

interface ButtonProps {
 label: string;
}

const Button: React.FC<ButtonProps> = ({ label }) => {
 return (
   <button
     className="bg-transparent font-semibold hover:bg-[#3e5463]
text-[#3e5463] hover:text-white py-2 px-4 border border-black
hover:border-transparent rounded"
     style={{
       border: "1px solid #5b7282",
       borderRadius: "2px",
       letterSpacing: ".5px",
     }}
   >
     {label}
   </button>
 );
};

export default Button;
```

Language

```
import Image from 'next/image'
import Link from 'next/link'
import React from 'react'

const Language = () => {
 return (
   <Link href="/" className="p-2">
   <Image src={"/global.png"} width={30} height={20} alt="IWF" />
 </Link>
 )
}

export default Language
```

MobileDropdown

```
"use client"
import Link from "next/link";
import React, { useState } from "react";

const MobileDropdown = () => {
 const [isOpen, setIsOpen] = useState(false);
 const [isTopicOpen, setIsTopicOpen] = useState(false);
 const toggleSubMenu = () => {
   setIsOpen(!isOpen);
 };
 const toggleSubMenuTopic = () => {
   setIsTopicOpen(!isTopicOpen);
 };
 return (
       <div className="lg:hidden">
         <div className="px-6 pt-2 pb-3 space-y-1 sm:px-8 ">
           <a href="/" className="hover:text-arghyam p-2 block">
             Home
           </a>
           <div className="relative">
             <a
               className="hover:text-arghyam p-2 flex items-center"
```

```
                  onClick={toggleSubMenu}
                >
                  Data
                  <svg
                    className="w-4 h-4 ml-1 fill-current  text-gray-500"
                    xmlns="http://www.w3.org/2000/svg"
                    viewBox="0 0 24 24"
                  >
                    <path d="M16.59 8.59L12 13.17 7.41 8.59 6 10l6 6 6-6z" />
                  </svg>
                </a>
                {isOpen && (
                  <div className="left-0 mt-2 w-60 bg-white  ">
                    <a
                      href="/"
                      className="block px-4 py-2 text-gray-800
hover:text-arghyam"
                    >
                      People & Content
                    </a>
                    <a
                      href="/"
                      className="block px-4 py-2 text-gray-800
hover:text-arghyam"
                    >
                      Data Finder
                    </a>
                    <a
                      href="/"
                      className="block px-4 py-2 text-gray-800
hover:text-arghyam"
                    >
                      Met Data
                    </a>
                    <a
                      href="/"
                      className="block px-4 py-2 text-gray-800
hover:text-arghyam"
                    >
                      State of Sanitation
```

```
                 </a>
               </div>
             )}
           </div>
           <a href="/" className="hover:text-arghyam p-2 block">
             Articles
           </a>
           <a href="/" className="hover:text-arghyam p-2 block">
             Opportunities & Events
           </a>
           <div className="relative">
             <a
               className="hover:text-arghyam p-2 flex items-center"
               onClick={toggleSubMenuTopic}
             >
               Topics
               <svg
                 className="w-4 h-4 ml-1 fill-current text-gray-500"
                 xmlns="http://www.w3.org/2000/svg"
                 viewBox="0 0 24 24"
               >
                 <path d="M16.59 8.59L12 13.17 7.41 8.59 6 10l6 6 6-6z" />
               </svg>
             </a>

             {isTopicOpen && (
               <div className=" left-0 mt-2 w-60 bg-white
dark:bg-gray-800 dark:border-gray-600">
                 <a
                   href="/"
                   className="block px-4 py-2 text-gray-800
hover:text-arghyam"
                 >
                   Solid Waste
                 </a>
                 <a
                   href="/"
                   className="block px-4 py-2 text-gray-800
hover:text-arghyam"
                 >
```

```
                   Rainwater Harvesting
                 </a>
                 <a
                   href="/"
                   className="block px-4 py-2 text-gray-800
hover:text-arghyam"
                 >
                   Rural Sanitation
                 </a>
                 <a
                   href="/"
                   className="block px-4 py-2 text-gray-800
hover:text-arghyam"
                 >
                   Agriculture
                 </a>
                 <a
                   href="/"
                   className="block px-4 py-2 text-gray-800
hover:text-arghyam"
                 >
                   View all Topics
                 </a>
               </div>
             )}
             <div className="p-2">
               <span className="tracking-wider text-xl font-semibold">
                 <Link
href="https://hindi.indiawaterportal.org/">हिंदी</Link>
               </span>
               <span className="m-4 tracking-wider text-xl  ">
                 <a href="/">English</a>
               </span>
             </div>
             <button
               className="bg-transparent  font-semibold hover:bg-[#3e5463]
text-[#3e5463] hover:text-white py-2 px-4 border border-black mt-2
hover:border-transparent rounded"
               style={{
                 border: "1px solid #5b7282",
                 borderRadius: "2px",
                 letterSpacing: ".5px",
               }}
             >
               Subscribe
             </button>
           </div>
         </div>
       </div>
 );
};

export default MobileDropdown;
```

Footer: this is sample footer

```
import Image from "next/image";
import Link from "next/link";
import React, { useState } from "react";
const Footer = () => {
 return (
   <footer className="bg-gray-400 text-gray-300 py-8">
   <div className="flex flex-col md:flex-row  ">
     <div className="ml-4  w-full  ">
       <Link href="https://www.indiawaterportal.org/about"
className="text-md font-bold mb-2 hover:text-arghyam ">
         About
       </Link>
     </div>
     <div className="ml-4 mt-2 w-full ">
       <Link href="https://www.indiawaterportal.org/contact-us"
className="text-md font-bold mb-2 hover:text-arghyam ">
       Contact
       </Link>
     </div>
     <div className="ml-4 mt-2 w-full ">
       <Link href="https://www.indiawaterportal.org/static-page/terms-use"
className="text-md font-bold mb-2 hover:text-arghyam">
       Terms of use
       </Link>
     </div>
     <div className="ml-4 mt-2 w-full">
       <Link
href="https://www.indiawaterportal.org/static-page/privacy-policy"
className="text-md font-bold mb-2 hover:text-arghyam">
      Privacy policy
       </Link>
     </div>
   </div>
 </footer>
 );
}

export default Footer;
```

Lets work on the main body

```
import Image from "next/image";
import Link from "next/link";

export default function Home() {
 return (
   <main className="mb-10">
     <div className="justify-center items-center m-4 mt-10 lg:mt-20">
       <div className=" text-left sm:px-16 xl:px-48">
         <h1 className="mb-4  text-3xl font-extrabold leading-none tracking-normal
text-gray-700 md:text-4xl lg:text-4xl dark:text-white">
           Met Data
         </h1>
         <p className="mb-6 text-left font-normal text-gray-600 lg:text-xl
dark:text-gray-400 leading-7">
           Meteorological datasets for 13 climate parameters for all districts
           of India from 1901 to 2002. Climate parameters included are:
           Precipitation, Max, Min and Avg Temperature, Cloud cover, Vapour
           pressure, Wet-day frequency, Diurnal temperature range, Ground frost
           frequency, Reference crop evapotranspiration, Potential
           evapotranspiration. The source data is from CRU 2.1 dataset of
           Tyndall Centre for Climate Change Research.
         </p>
       </div>
       <div className=" text-left sm:px-16 xl:px-48">
```

```
         <h1 className="mb-4  text-xl font-semibold leading-none tracking-normal
text-gray-700 md:text-2xl lg:text-3xl dark:text-white">
           Source of Data
         </h1>
         <p className="mb-6 text-left font-normal text-gray-600 lg:text-xl
dark:text-gray-400">
           India Water Portal, Tyndall Centre for Climate Change Research
         </p>
       </div>
       <div className=" text-left sm:px-16 xl:px-48">
         <h1 className="mb-4  text-xl font-semibold leading-none tracking-normal
text-gray-700 md:text-2xl lg:text-3xl dark:text-white">
         Background of Data
         </h1>
         <p className="mb-6 text-left font-normal text-gray-600 lg:text-xl
dark:text-gray-400">
         <Link

href="https://www.indiawaterportal.org/articles/background-meteorological-datasets
"
             className="hover:text-arghyam"
           >
             Background on the Meteorological Dataset
           </Link>
         </p>
       </div>
       <div className=" text-left sm:px-16 xl:px-48">
         <h1 className="mb-4  text-xl font-semibold leading-none tracking-normal
text-gray-700 md:text-2xl lg:text-3xl dark:text-white">
           Data Format
         </h1>
         <p className="mb-6 text-left font-normal text-gray-600 lg:text-xl
dark:text-gray-400">
           Data is available in CSV format and charts to visualize
         </p>
       </div>
       <div className=" text-left sm:px-16 xl:px-48">
         <h1 className="mb-3  text-xl font-semibold leading-7  tracking-normal
text-gray-800 md:text-2xl lg:text-3xl dark:text-white">
         Get the Charts according to your States and District
         </h1>
         <div className="mb-6 text-left font-normal text-gray-600 lg:text-xl
dark:text-gray-400">
         <Link
```

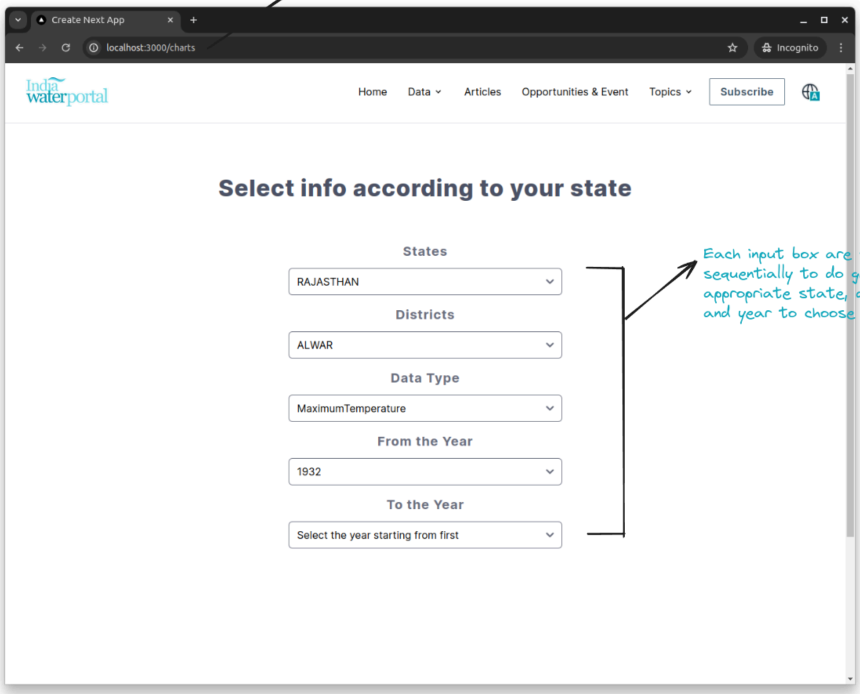
```
             href="/charts"
             className="arghyam"
           >
            Visualize the Charts
            <div className="py-2"><button
              className="bg-transparent  font-semibold hover:bg-[#3e5463]
text-[#3e5463] hover:text-white py-3 px-4 border border-black mt-2
hover:border-transparent rounded"
              style={{
                border: "1px solid #5b7282",
                borderRadius: "2px",
                letterSpacing: ".5px",
              }}
             >
              Charts and Data
             </button>
             </div>
           </Link>
         </div>
       </div>
     </div>
   </main>
 );
}
```

Make a folder in app directory called charts , with two folder layout.tsx and page.tsx

Page.tsx

Charts and Trend page: Before Selecting To the Year

/charts route



Create Next App

localhost:3000/charts

Home | Data | Articles | Opportunities & Event | Topics | Subscribe

Select info according to your state

States: RAJASTHAN

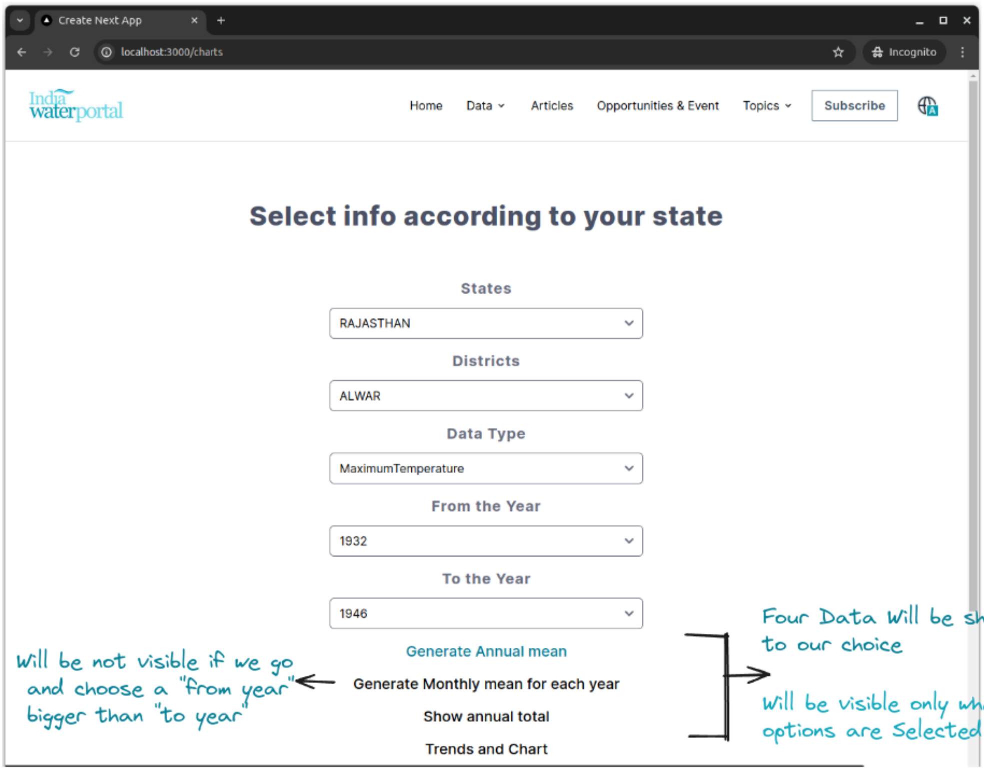
Districts: ALWAR

Data Type: MaximumTemperature

From the Year: 1932

To the Year: Select the year starting from first

Each input box are selected sequentially to do get the appropriate state, district and year to choose



Create Next App

localhost:3000/charts

Home | Data | Articles | Opportunities & Event | Topics | Subscribe

Select info according to your state

States: RAJASTHAN

Districts: ALWAR

Data Type: MaximumTemperature

From the Year: 1932

To the Year: 1946

Generate Annual mean

Generate Monthly mean for each year

Show annual total

Trends and Chart

Will be not visible if we go and choose a "from year" bigger than "to year"

Four Data Will be shown acc. to our choice

Will be visible only when all the options are Selected

```
"use client";

import React from "react";
import { useEffect, useState } from "react";
import { State, District, Year } from "@/types"
import Link from "next/link";
import { parameters } from "@/data";

const Page = () => {
 const [state, setState] = useState([]);
 const [selectedState, setSelectedState] = useState("");
 const [selectedDataType, setSelectedDataType] = useState("");
 const [selectedDistrictType, setSelectedDistrictType] = useState("");
 const [selectedFromYear, setSelectedFromYear] = useState("");
 const [selectedToYear, setSelectedToYear] = useState("");
 const [districts, setDistricts] = useState([]);
 const [fromYear, setfromYear] = useState([]);
 const [toYear, setToYear] = useState([]);

 useEffect(() => {
   fetch(`http://localhost:8001/state`)
     .then((res) => res.json())
     .then((data) => setState(data))
     .catch((err) => console.log(err));
 }, []);

 const handleStateSelectChange = (event: any) => {
   const selectedValue = event.target.value;
   console.log(selectedValue);
   setSelectedState(selectedValue);
   fetch(`http://localhost:8001/distict?Stateid=${selectedValue}`)
     .then((res) => res.json())
     .then((data) => setDistricts(data))
     .catch((err) => console.log(err));
 };

 const handleDistrictSelectChange = (event: any) => {
   const selectedDistrictValue = event.target.value;
   console.log(selectedDistrictValue);
   setSelectedDistrictType(selectedDistrictValue);
 };

 const handleDataSelectChange = (event: any) => {
   const selectedDataValue = event.target.value;
   console.log(selectedDataValue);
   setSelectedDataType(selectedDataValue);
   fetch(`http://localhost:8001/year?tablename=${selectedDataValue}`)
     .then((res) => res.json())
     .then((data) => {
       setfromYear(data);
       setToYear(data);
     })
     .catch((err) => console.log(err));
 };

 const handleFromSelectChangeYear = (event: any) => {
   const selectedFromYearValue = event.target.value;
   setSelectedFromYear(selectedFromYearValue);
   fetch(

`http://localhost:8001/yearAfter?tablename=${selectedDataType}&selectedFromYear=${
selectedFromYearValue}`
   )
     .then((res) => res.json())
     .then((data) =>{
       setToYear(data)
       setSelectedToYear("");
     })
     .catch((err) => console.log(err));
 };

 const handleToYearChange = (event: any) => {
   const selectedToYearValue = event.target.value;
   setSelectedToYear(selectedToYearValue);
 };

 return (
   <main>
     <div className="h-full">
       <div className=" justify-center items-center h-screen m-4 mt-10 lg:mt-20">
         <div className=" text-center sm:px-16 xl:px-48 mb-10 md:mb-16">
           <h1 className="mb-4  text-3xl font-extrabold leading-none tracking-wide
text-gray-600 md:text-4xl lg:text-4xl dark:text-white">
             Select info according to your state
           </h1>
         </div>
         <div className="mt-4 text-center">
           <div className="mt-4">
```

```
             <h1 className="text-gray-500 font-bold m-3 md:text-xl
tracking-wider">
               States
             </h1>
             <select
               value={selectedState}
               onChange={handleStateSelectChange}
               className="w-8/12 md:w-5/12 lg:w-4/12 bg-gray-50 border
border-gray-300 text-gray-900 text-sm rounded-lg  p-3"
             >
               <option>Select a state</option>
               {state.map((item: State) => (
                 <option key={item.stateid} value={item.stateid}>
                   {item.State}
                 </option>
               ))}
             </select>
           </div>
           <div className="mt-4">
             <h1 className="text-gray-500 font-bold m-3 tracking-wider md:text-xl
">
               Districts
             </h1>
             <select
               value={selectedDistrictType}
               onChange={handleDistrictSelectChange}
               className="w-8/12 md:w-5/12 lg:w-4/12 bg-gray-50 border
border-gray-300 text-gray-900 text-sm rounded-lg  p-3"
             >
               <option value="">
                 Select a district from the selected state
               </option>
               {districts.map((district: District) => (
                 <option key={district.districtid} value={district.districtid}>
                   {district.Distict}
                 </option>
               ))}
             </select>
           </div>
           <div className="mt-4">
             <h1 className="text-gray-500 font-bold tracking-wider m-3
md:text-xl">
               Data Type
             </h1>
```

```
            <select
              value={selectedDataType}
              onChange={handleDataSelectChange}
              className="w-8/12 md:w-5/12 lg:w-4/12 bg-gray-50 border
border-gray-300 text-gray-900 text-sm rounded-lg  p-3"
            >
              <option>Select the Data type</option>
              {parameters.map((parameter, index) => (
                <option key={index} value={parameter}>
                  {parameter}
                </option>
              ))}
            </select>
          </div>
          <div className="mt-4">
            <h1 className="text-gray-500 font-bold tracking-wider m-3 md:text-xl
">
              From the Year
            </h1>
            <select
              value={selectedFromYear}
              onChange={handleFromSelectChangeYear}
              className="w-8/12 md:w-5/12 lg:w-4/12 bg-gray-50 border
border-gray-300 text-gray-800 text-sm rounded-lg  p-3"
            >
              <option value="">Select the year starting from</option>
              {fromYear.map((year: Year) => (
                <option key={year.year_val} value={year.year_val}>
                  {year.year_val}
                </option>
              ))}
            </select>
          </div>
          <div className="mt-4">
            <h1 className="text-gray-500 font-bold m-3 tracking-wider md:text-xl
">
              To the Year
            </h1>
            <select
              className="w-8/12 md:w-5/12 lg:w-4/12 bg-gray-50 border
border-gray-300 text-gray-900 text-sm rounded-lg  p-3"
              onChange={handleToYearChange}
            >
              <option value="">Select the year starting from first</option>
```

```
               {toYear.map((year: Year) => (
                 <option key={year.year_val} value={year.year_val}>
                   {year.year_val}
                 </option>
               ))}
             </select>
           </div>
         </div>
         {selectedToYear && (
           <div className="flex justify-center mt-6 sm:px-16 xl:px-48 mb-10
md:mb-16">
             <div className="mb-4 w-8/12  leading-none lg:text-4xl
dark:text-white">
               <div className="">
                 <Link
                   href={{
                     pathname:'/charts/AnnualMean',
                     query:{
                       selectedState,
                       selectedDistrictType,
                       selectedDataType,
                       selectedFromYear,
                       selectedToYear
                     }
                   }}
                   className=" text-xl font-medium "
                 >
                   <p className="m-4 text-center arghyam">
                     Generate Annual mean
                   </p>
                 </Link>
                 <Link
                   href={{
                     pathname:"/charts/MonthlyMean",
                     query:{
                       selectedState,
                       selectedDistrictType,
                       selectedDataType,
                       selectedFromYear,
                       selectedToYear
                     }
                   }}
                   className=" text-xl font-medium "
                 >
```

```
          <p className="m-4 text-center arghyam">
            Generate Monthly mean for each year
          </p>
        </Link>
      </div>
      <div className="">
        <Link
          href={{
            pathname:"/charts/AnnualTotal",
            query:{
              selectedState,
              selectedDistrictType,
              selectedDataType,
              selectedFromYear,
              selectedToYear
            }
          }}
          className="text-xl font-medium "
        >
          <p className="m-4 text-center arghyam">
            Show annual total
          </p>
        </Link>
        <Link
          href={{
            pathname:"/charts/Trends",
            query:{
              selectedState,
              selectedDistrictType,
              selectedDataType,
              selectedFromYear,
              selectedToYear
            }
          }}
          className="text-xl font-medium "
        >
          <p className="m-4 text-center arghyam">
            Trends and Chart
          </p>
        </Link>
      </div>
    </div>
  </div>
)}
```

```
      </div>
    </div>
  </main>
 );
};

export default Page;
```

Now make four folder for four different option which we saw in charts Homepage

- AnnualMean
- MonthlyMean
- AnnualTotal
- Trends

Now Lets add some testing

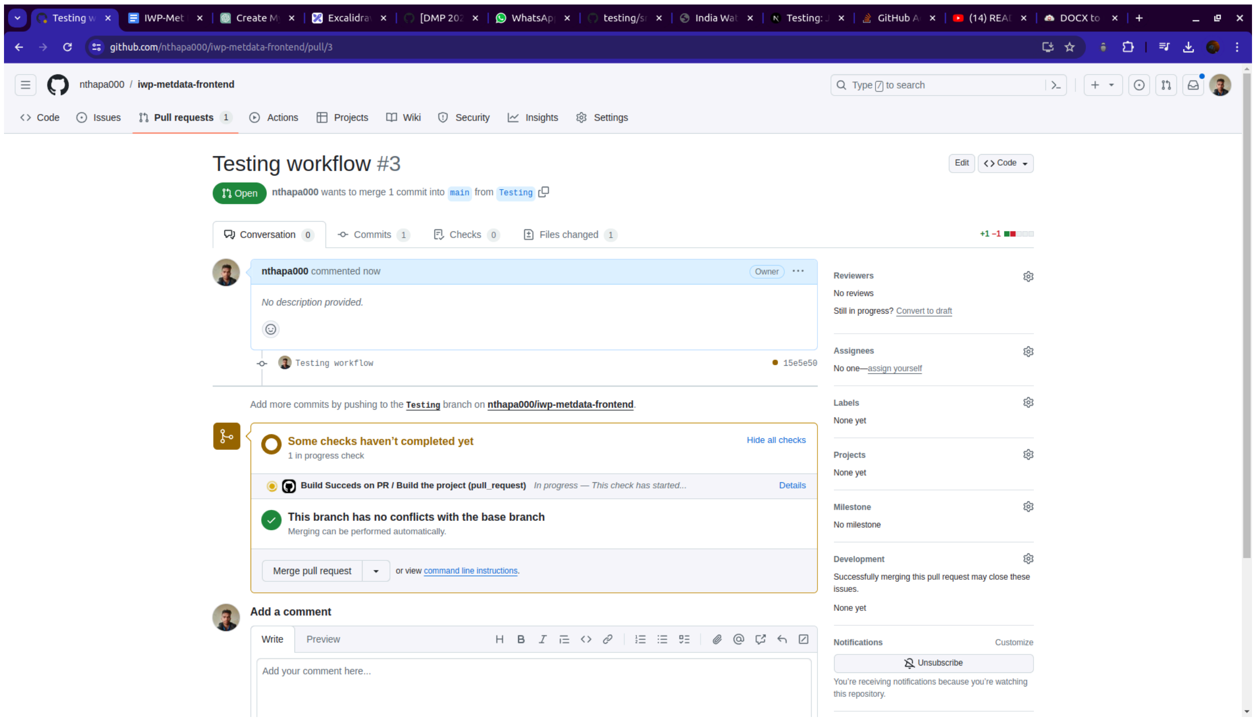
```
npm install -D jest jest-environment-jsdom @testing-library/react
@testing-library/jest-dom
```

https://nextjs.org/docs/app/building-your-application/testing/jest

annualMean.test.ts

```
import '@testing-library/jest-dom';
import { WeatherData } from "@/types";
import { calculateAnnualMean } from '@/app/charts/AnnualMean/utils';

describe('calculateAnnualMean', () => {
 it('calculates the correct annual mean from multiple years', () => {
   const mockData: WeatherData[] = [
     { stateid: 1, districtid: 1, State: 'State1', state_dist_key: 's1d1',
Distict: 'District1', year_val: 2000, jan: 10, feb: 20, mar: 30, apr: 40,
may: 50, jun: 60, jul: 70, aug: 80, sep: 90, oct: 100, nov: 110, dec: 120
},
     { stateid: 2, districtid: 2, State: 'State2', state_dist_key: 's2d2',
Distict: 'District2', year_val: 2001, jan: 5, feb: 10, mar: 15, apr: 20,
may: 25, jun: 30, jul: 35, aug: 40, sep: 45, oct: 50, nov: 55, dec: 60 }
   ];

   const result = calculateAnnualMean(mockData);
   expect(result).toEqual(585); // Expected mean ((780 + 390) / 2)
 });

 it('handles an empty data array', () => {
   const mockData: WeatherData[] = [];
   const result = calculateAnnualMean(mockData);
   expect(result).toBeNaN(); // Should return NaN since division by zero
is not defined
 });

 it('calculates the correct annual mean for a single year', () => {
   const mockData: WeatherData[] = [
     { stateid: 1, districtid: 1, State: 'State1', state_dist_key: 's1d1',
Distict: 'District1', year_val: 2000, jan: 10, feb: 20, mar: 30, apr: 40,
may: 50, jun: 60, jul: 70, aug: 80, sep: 90, oct: 100, nov: 110, dec: 120
}
   ];

   const result = calculateAnnualMean(mockData);
   expect(result).toEqual(780); // Only one year's data, mean should be
the sum of months
 });
});
```

Workflow test on PR


github.com/nthapa000/iwp-metdata-frontend/pull/3

nthapa000 / iwp-metdata-frontend

Code · Issues · Pull requests 1 · Actions · Projects · Wiki · Security · Insights · Settings

Testing workflow #3

Open · nthapa000 wants to merge 1 commit into `main` from `Testing`

Conversation 0 · Commits 1 · Checks 0 · Files changed 1 · +1 −1

nthapa000 commented now — Owner

No description provided.

`Testing workflow` — `15e5e50`

Add more commits by pushing to the **Testing** branch on **nthapa000/iwp-metdata-frontend**.

Some checks haven't completed yet
1 in progress check — Hide all checks

Build Succeds on PR / Build the project (pull_request) *In progress — This check has started...* Details

This branch has no conflicts with the base branch
Merging can be performed automatically.

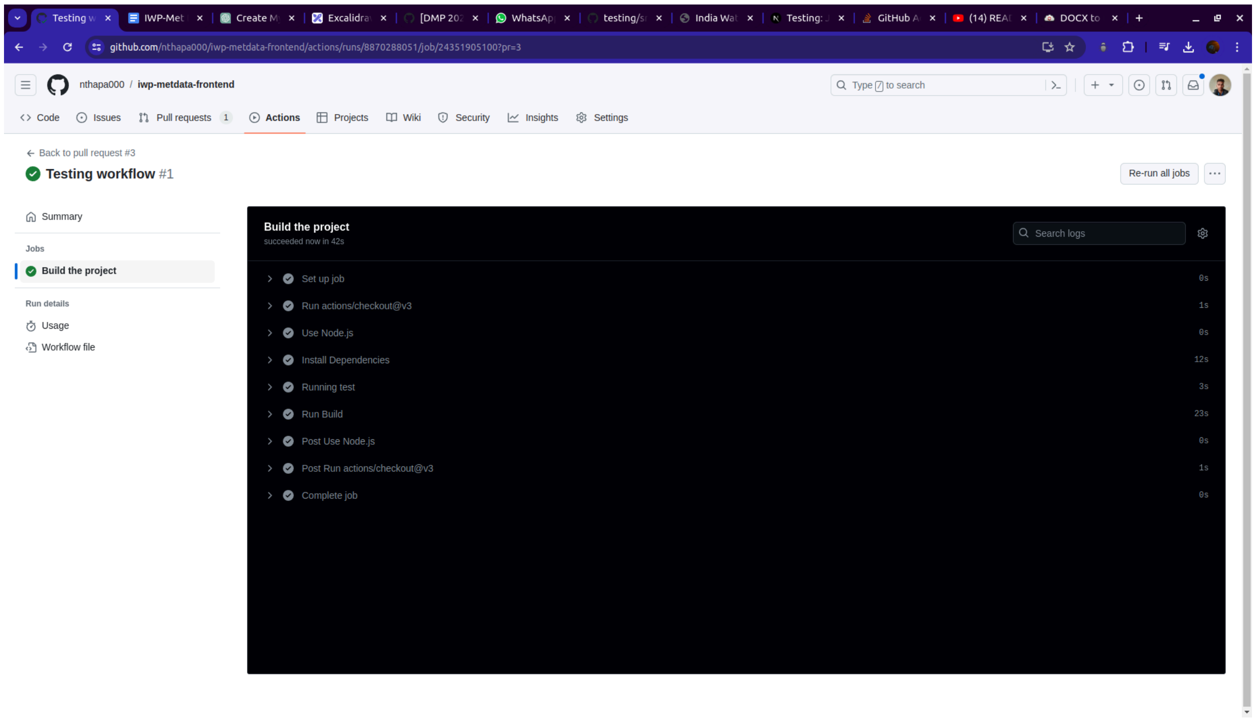
Merge pull request · or view command line instructions.

Add a comment

Write · Preview

Add your comment here...

Reviewers: No reviews. Still in progress? Convert to draft
Assignees: No one—assign yourself
Labels: None yet
Projects: None yet
Milestone: No milestone
Development: Successfully merging this pull request may close these issues. None yet
Notifications: Customize · Unsubscribe. You're receiving notifications because you're watching this repository.


github.com/nthapa000/iwp-metdata-frontend/actions/runs/8870288051/job/24351905100?pr=3

nthapa000 / iwp-metdata-frontend

Code · Issues · Pull requests 1 · Actions · Projects · Wiki · Security · Insights · Settings

← Back to pull request #3

Testing workflow #1 — Re-run all jobs

Summary
Jobs: Build the project
Run details: Usage · Workflow file

Build the project
succeeded now in 42s

```
Set up job                      0s
Run actions/checkout@v3         1s
Use Node.js                     0s
Install Dependencies           12s
Running test                    3s
Run Build                      23s
Post Use Node.js                0s
Post Run actions/checkout@v3    1s
Complete job                    0s
```