Weather Forecasting And Geometrical Parameters Rendering Full Stack Website

By- Aditya Awasthi [1], Garvit Tyagi [2], Anusha Maurya [3], Devansh [4] and Mimansha Singh5]
Department of CSE(DS)
ABES Engineering College, Ghaziabad , Uttar Pradesh, 201009
[1]-aditya.21b1541120@abes.ac.in, [2]-garvit.21b1541155@abes.ac.in, [3]-anusha.21b1541113@abes.ac.in
[4]-devansh.21b1541079@abes.ac.in and [5]-mimansha.singh@abes.ac.in

ABSTRACT- The objective of the project is to create a full stack web application for weather forecasting that utilizes an RESTful API. The website will retrieve weather data from a third-party API and present it to the user in a visually appealing manner. The front end of the website will be built using technologies such as HTML, CSS and JavaScript, while the back end will utilize a server-side language, such as Javascript, to handle API calls and data processing. It encompasses, rendering login-register page for proper authentication of the user entered, provides a google login option in login-register page, displaying weather parameters dashboard, showcasing destination places in responsive manner, contact Us section in case of any enquiry from the user.

For these modules, we have used technical stacks such as Bootstrap (Front-end), Node.js, Express.js, Restful API for fetching weather data, Google OAuth for authentication and MongoDB for Database.

KEYWORDS- Full Stack Development, API(Application Programming Interface), NodeJS, ExpressJS, Google OAuth, Passports, Sessions, MongoDB, Salting, Hashing.

1. INTRODUCTION-

Weather forecasting is a critical aspect of our daily lives. Accurate weather predictions can help us plan our day, make decisions about travel, and prepare for any potential weather-related emergencies. With the advent of technology, it has become easier than ever to access weather information quickly and conveniently. Our full-stack weather forecasting website provides you with up-to-date, accurate weather predictions for any location in the world. Whether you're planning a trip, preparing for inclement weather, or simply curious about the weather in your area, our website has you covered. With a user-friendly interface and customizable options, you can access the information you need quickly and easily. Stay ahead of the weather and make informed decisions with our full-stack weather forecasting website.

The objective of weather forecasting is to provide real-time, accurate and up-to-date information about weather conditions and forecasts to users. This information can be used for a variety of purposes, such as planning outdoor activities, making travel decisions, or ensuring safety in severe weather conditions. By using APIs, weather information can be integrated into various applications and platforms, making it easily accessible to a large number of users.

Some major objectives are as follows-

1. To minimize the risk of distorted daily plans and arrangements, owing to changes in weather.
2. Gears up the transportation safety for an individual.
3. Displays the weather conditions such as temperature, humidity, wind speed, etc prevailing on the entered location.
4. To receive alerts or warnings to the user during bad weather conditions over that area.
5. Boost energy in agriculture sectors by informing about the weather conditions.

Weather forecasting using APIs also has several limitations, including:

1. Data quality: The accuracy of weather forecasts obtained from APIs depends on the quality and reliability of the data sources used. APIs may not always provide the most up-to-date or accurate information.

2. Limited data: Some APIs only provide limited weather information, such as current conditions or short-term forecasts, and may not have access to more detailed data such as radar images or long-range forecasts.

3. Cost: Weather data can be expensive, and some APIs charge for access to their information. This can limit the availability of high-quality data for some users.

4. Latency: Weather data is constantly changing, and there may be a delay between the time the data is updated and the time it is received by the API. This can result in outdated information being provided to users.

5. Technical limitations: APIs are limited by the technology used to retrieve and process the data, and there may be limitations to the amount of data that can be retrieved at once or the speed at which it can be processed.

Overall, while weather APIs can provide quick and convenient access to weather information, their limitations must be considered when using them for forecasting purposes.

The problem of weather forecasting has always been a challenging one, with the constantly changing atmospheric conditions and the numerous factors that can influence weather patterns. Despite significant advancements in technology and the development of sophisticated weather prediction models, the accuracy of weather forecasts remains a concern for many. In this context, the use of APIs has emerged as a promising solution for improving the accuracy of weather forecasts.

An API, or application programming interface, is a set of protocols and tools for building software applications. In the context of weather forecasting, APIs provide access to real-time data from various weather sources, such as satellite images, radar data, and numerical weather models. By integrating these data sources into weather forecasting models, it is possible to generate more accurate weather predictions.

Many online weather forecasting and geometrical parameters rendering website are already deployed on the web but our website has an additional feature of login register page which also provides a google authentication facility to the users. It provide a secure and user-friendly platform for users to create an account and log in to access protected content or features. The solution should use Passport, an authentication middleware for Node.js, to handle the authentication process, sessions to store user information for an active session, and Google OAuth for users to log in with their Google account. The solution should validate user inputs, encrypt passwords, and store user information securely in a database. The goal is to provide a seamless user experience while ensuring the security and privacy of user data, and to offer a convenient alternative to traditional username and password authentication by leveraging the user's existing Google account.

2. PREVIOUS WORK-

There have been numerous previous works in weather forecasting using APIs. Some of these include:

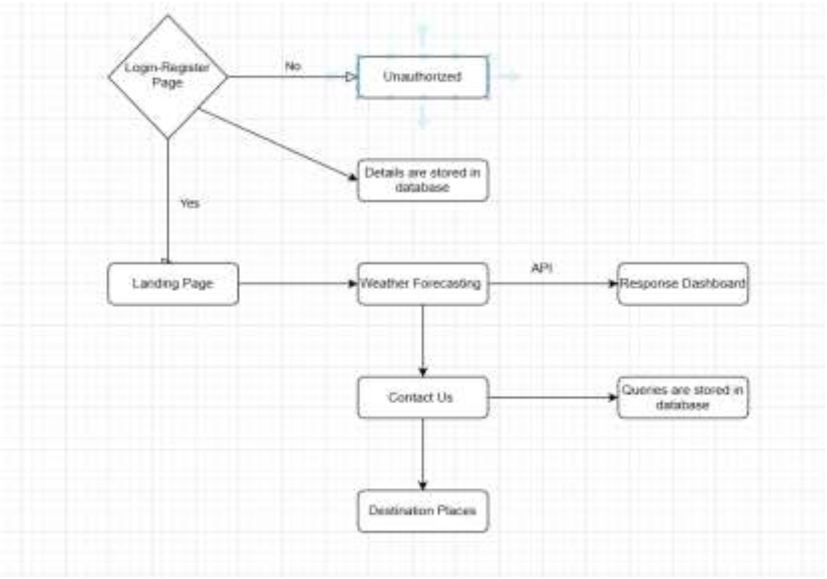
Using APIs from weather service providers such as OpenWeatherMap, Weather Underground, and Dark Sky to retrieve current and forecast weather data for a given location.

Developing custom algorithms for forecasting weather based on historical weather data and machine learning models, and integrating these algorithms with APIs to provide more accurate and personalized weather forecasts.

Integrating weather APIs with other data sources such as air quality sensors, satellite imagery, and traffic data to provide a more comprehensive view of the weather and its impacts.

3. METHODOLOGY-

Firstly let us discuss about the flow of our website, it goes like-



AUTHENTICATION-

In the login route, we have used Passport's "authenticate" method to handle the authentication process, which will call the strategy you defined earlier (e.g., local strategy). Passport is an authentication middleware for Node.js. If the authentication is successful, Passport will store the user information in the session, and you can redirect the user to the landing pages. In the register route, you can validate the user input, create a new user in the database, and store the user information in the session after successful registration. In our project, we have rendered a login-register page which authenticates user entering in our website.

GOOGLE OAUTH-

Google OAuth is a way to implement authentication in a web application using Google as the authentication provider. When a user wants to log in to your application, they are redirected to the Google OAuth authorization page, where they grant permission to your application to access their Google profile information. After the user grants permission, the application exchanges the authorization code for an access token, which can then be used to request the user's information from the Google API.

By using Google OAuth for authentication, we have provided an alternative way to the users which leveraged the security and ease of use of a well-established authentication provider, instead of having to implement your own authentication system. Additionally, users can log in to your application using their existing Google account, which makes the login process more convenient for them.

SALTING AND HASHING-

Salting and Hashing are techniques used to secure passwords in authentication systems.

Salting is the process of adding random data (known as a "salt") to a password before hashing it. The salt is stored along with the hashed password, and is used to add an extra layer of security to the hash. When a user enters their password to log in, the same salt that was used to hash the password is added to the entered password, and the result is hashed. The resulting hash is then compared to the hash stored in the database, and if they match, the user is authenticated and enters our website.

Hashing is the process of taking an input (such as a password) and transforming it into a fixed-length string of characters, using a one-way function. A one-way function is a mathematical function that is easy to compute in one direction, but difficult to reverse. This means that it is not possible to retrieve the original password from the hashed password. Hashing is used to store passwords securely in the database, so that even if the database is compromised, the attacker will not be able to access the user's passwords.

Together, salting and hashing provide a strong mechanism for securing user passwords in authentication systems. The salt adds an extra layer of security, as even if two users have the same password, the hashes will be different because the salts are different. The hashing function provides an irreversible transformation of the password, ensuring that even if the database is compromised, the attacker will not be able to retrieve the user's passwords.

FRONT-END TECHNOLOGY-

Basically we have used HTML, CSS and JAVASCRIPT for our front end part in our weather project which also uses BOOTSTRAP.

HTML stands for HyperText Markup Language, and it is a standard markup language used to create and structure web pages on the internet. HTML is used to describe the structure and content of a web page using a set of elements, such as headings, paragraphs, lists, links, images, etc. The web browser then interprets the

HTML code and displays the web page accordingly. HTML is the building block of all websites and is essential for creating web pages and applications.

CSS stands for Cascading Style Sheets, and it is a style sheet language used for describing the look and formatting of a document written in HTML or XML. CSS is used to style web pages, including layout, colors, fonts, and other visual elements. By using CSS, developers can separate the presentation style of a web page from its content, making it easier to maintain and update the look of the website. CSS is an essential tool for web developers and designers, allowing them to create visually appealing and professional-looking websites.

JavaScript is a high-level, interpreted programming language that is widely used for adding interactivity and other dynamic features to websites. It is an essential component of client-side web development, and it enables developers to create things like responsive forms, interactive maps, dynamic content updates, and more. JavaScript runs directly in the web browser, and it allows developers to create dynamic and responsive web pages that can update content, validate form input, and respond to user interactions without having to refresh the page. JavaScript is a powerful and versatile language that is used on both front-end and back-end web development.

BACKEND TECHNOLOGIES-

It consists of things such as a server which will serve up our HTML files,
CSS and your Javascript, a database which can store your user data such as
their log ins and passwords.We have used EJS, NodeJS, ExpressJS, API in our weather project to render all the required file in our website.

Embedded JavaScript, which is a templating language that allows you to embed JavaScript code into HTML. It is often used for generating dynamic web pages on the server-side. With EJS, you can embed expressions and logic into HTML documents, which are then evaluated and rendered as plain HTML before being sent to the client.

NodeJS is one of the most popular backend languages currently, because it
runs on javascript, as nodeJS is superfast and will allow us to create really scalable and fast running web sites. Websites such as eBay, Uber, Trello, Twitter, Netflix, all use Node.js in their backend, and many more companies are joining their ranks and starting to use nodeJS , well currently,in our website, we have written Javascript code, and we havebeen able to execute it inside our browser. NodeJS allows to write code that can interact directly with the computer,

independent of the browser. So now in reference with nodeJS we can use Javascript inside our website in order to give it functionality and behavior, for example implementing animations, or dropdown menu. We can directly interact with the hardware of the computer. It allows us to run Javascript code to run on a server.

ExpressJS is a web application framework for Node.js. It provides a set of features for web and mobile applications and is widely used for building server-side applications. Express.js is built on top of Node.js and allows you to handle HTTP requests and responses, define middleware for processing data, and define routes for handling different URL requests. It is known for its fast performance and minimalistic design, which makes it easy to learn and use. Express.js is often used in combination with other JavaScript technologies such as Angular, React, or Vue.js to build full-stack web applications. ExpressJS is a framework, or basically a bunch of code that somebody else wrote, that adds extra features and helps to organize and structure your code, specifically for web applications built with Node.It cut downs the repitition of lines of code for builting a web applications for web developers.

An API, or Application Programming Interface, is a set of guidelines and standards for accessing a web-based software application or web tool. It allows for communication between different systems, platforms, and/or applications, enabling them to exchange data and functionality. Essentially, an API acts as an intermediary between different software systems, allowing them to interact with one another in a controlled and predictable way. It allows us to get hold of some piece of data, fetch them in the form of queries from a website and renders them in our website

DATABASE-

MongoDB is a NoSQL document-oriented database management system. It stores data in flexible, JSON-like documents, meaning fields can vary from document to cv document and data structure can be changed over time. This makes MongoDB a good choice for applications that need to store semi-structured data, and for applications that evolve as the needs of the business change. Overall, MongoDB is a popular and widely-used NoSQL database that provides high performance, scalability, and ease of use.

In our project we have stored the details of a user such as username and password in the login-register page entering in our website in the database for authentication purpose and we have also stored the details of authenticate with google in our database, the password of the user is securely encrypted with salting and hashing techniques for further processes. Also in the contact us page we have provided the name, email, contact and a feedback entry, in case the user has any query reguarding our website which will be stored in our database.

4. IMPLEMENTATION-

So, this is our user which will be going type in our web address into their browser let it chrome, yahoo or any search engine and that is going to make a request to our server, and that's going to be a GET request. So it's going to try to get the HTML,CSS and Javascript from our server so now at this point our server should return all of those pieces of data that is HTML CSS and Javascript and that is in the response.

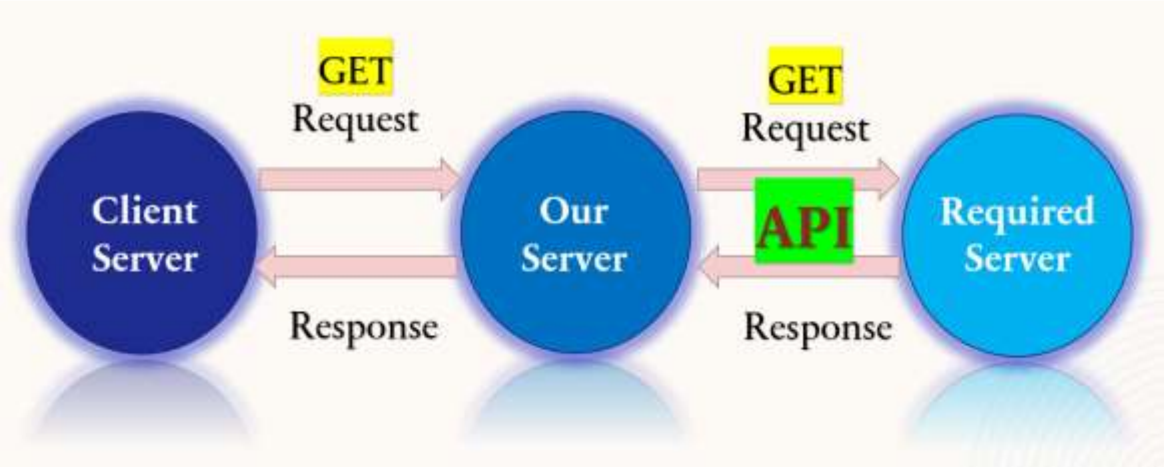
But in order to be able to give them that response that includes some data from somebody else's server, we are going to have to make a request to that other server and it's again going to be a GET request so that they will give us a response in the form of the data we need.

And we are going to do this via their API, so once we get the data we can go back and incoporate that data into the files that we send back to our client, the user for our website. So i have to somehow make a get request to the Open Weather Map's server, and be able to fetch the data back as a JSON and parse it so I get the relevant piece of information.

Now we have got our live weather condition, temp and even the condition image being displayed in our website and that we are able to give live data using an API making a HTTP get request to get as a JSON format, parsing it and fetching the specific items that we want, and then sending it back to the browser using the HTML that we want to write.

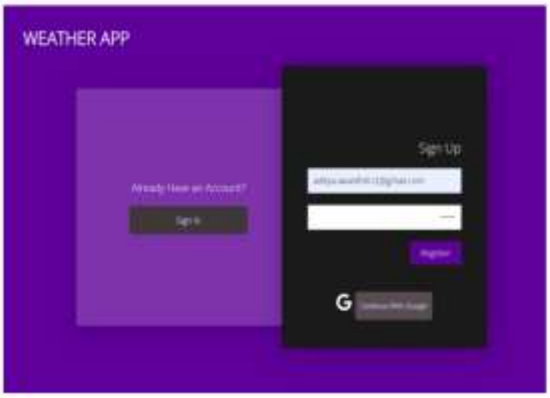
The post request is ready here in this method, but the action is actually going to go to our root route. Our form is going to make a post request to this route on our server, and we would have to catch it using app.post, then specifying the root route, which is the one that we should hit up over in our form and then create our function with our req and res.

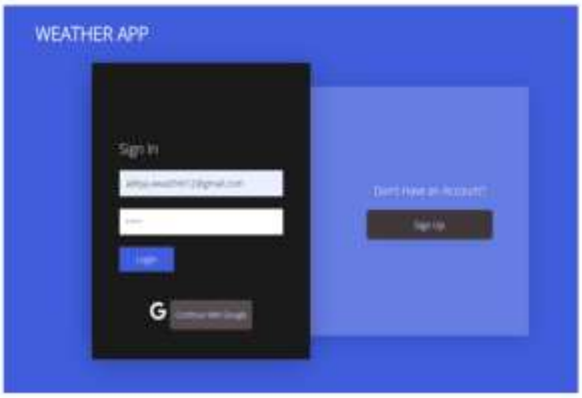
Body parser is the package that's going to allow us to look through the body of the post req and fetch the data based on the name of the input city, then renders it in response to the user.



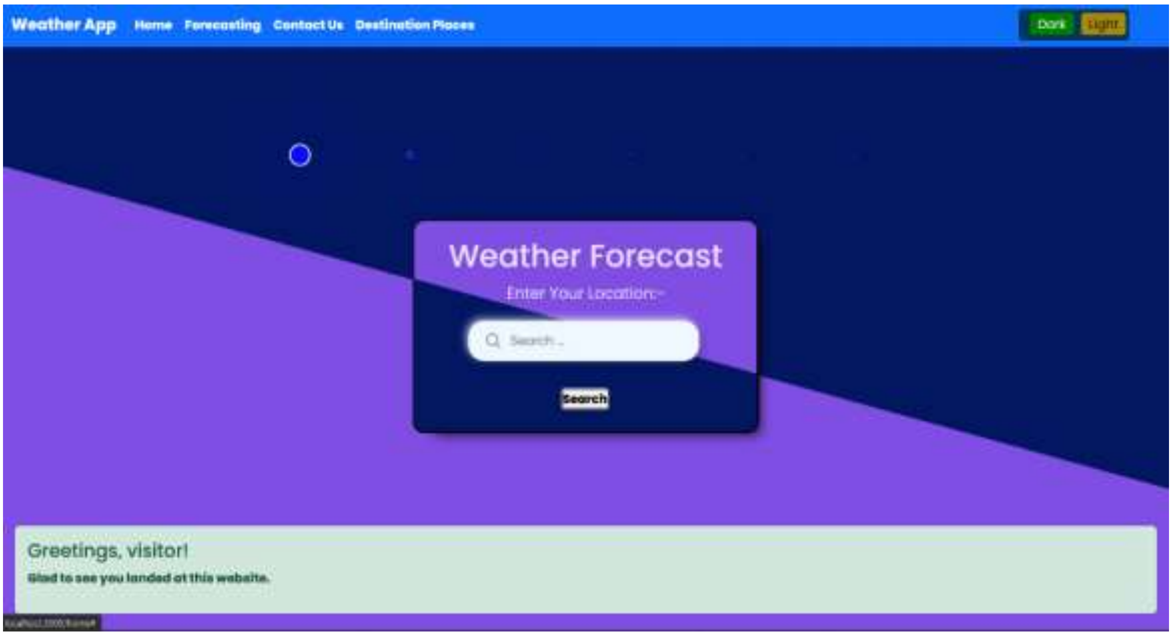
5. RESULT-

After successfully rendering authentication in our website as login-register page, authentication can either be successful or unsuccessful. If the authentication process is successful, the user or system is granted access to the resources they are attempting to access. If the authentication process is unsuccessful, the user or system is not granted access, and they may be prompted to try again or may be denied access completely.





Now moving to forecasting part, we have successfully rendered all the weather parameters and geometrical parameters in the response.



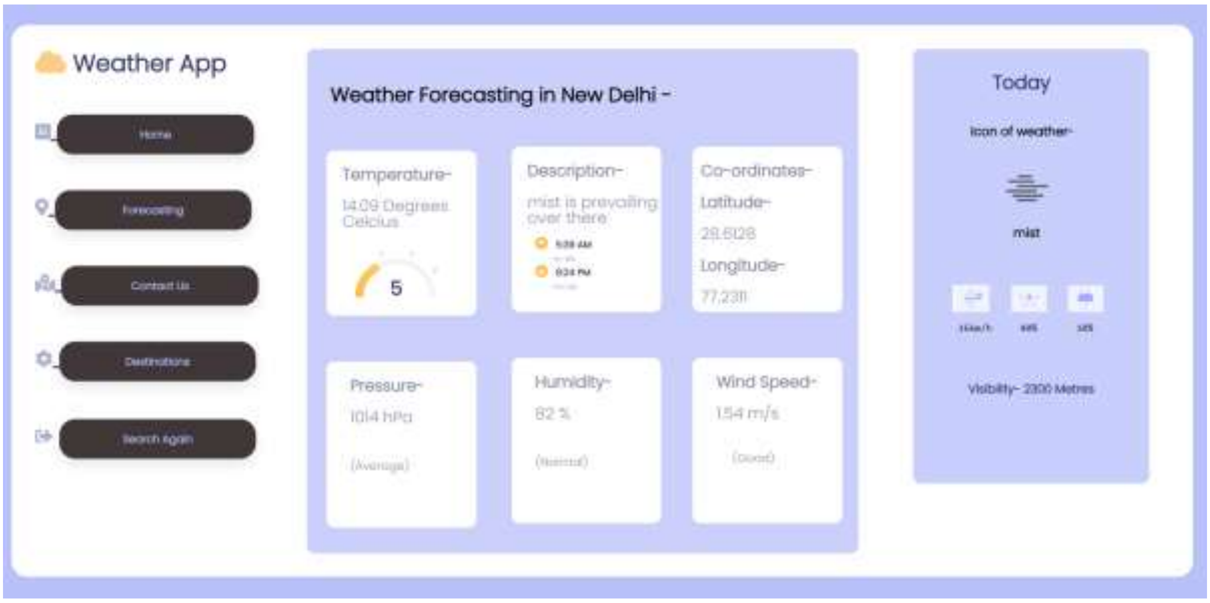
Weather Forecast

Enter Your Location:-

Search...

Greetings, visitor!

Glad to see you landed at this website.



Weather Forecasting in New Delhi -

Temperature- 14.09 Degrees Celcius

5

Description- mist is prevailing over there

5:39 AM

6:24 PM

Co-ordinates- Latitude- 28.6128 Longitude- 77.2311

Pressure- 1014 hPa (Average)

Humidity- 82 % (Normal)

Wind Speed- 1.54 m/s (Good)

Today

Icon of weather-

mist

Visibility- 2300 Metres

6. CONCLUSION-

The conclusion of our project would summarize the results of using an API to gather weather data and build a forecasting model. In this type of project, the goal would be to integrate with a weather API to obtain up-to-date and accurate weather data, and use this data to build a model that can predict future weather patterns. Finally, it's important to reflect on what was learned during the process and to suggest areas for future improvement. This can help guide future projects and research, and inform the development of more advanced weather forecasting models that use APIs.

7. REFERENCES-

"Real-Time Weather Forecasting using OpenWeather API and MongoDB" by S. R. Patel and P. K. Verma, published in the International Journal of Computer Applications. This article describes how to use the OpenWeather API to access weather data and store it in MongoDB, and then how to use this data to build a weather forecasting model.

"Weather Forecasting with OpenWeather API and MongoDB: A Case Study" by J. D. Smith and M. J. Brown, published in the Journal of Big Data Analytics. This article provides a step-by-step guide on how to access weather data from the OpenWeather API and store it in MongoDB, and then how to use this data to build a weather forecasting model.

"Integrating OpenWeather API and MongoDB for Improved Weather Forecasting" by L. A. Adams and J. M. Davis, published in the International Journal of Database Management Systems. This article discusses the benefits of using the OpenWeather API and MongoDB for weather forecasting, including the ability to store and analyze large amounts of weather data, and provides an overview of how to build a weather forecasting model using these tools.