International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 06, ACSCON - 2026

WeatherVibe: A Real Time Weather App

1DA23AI041
Dept. of Artificial Intelligence and Machine Learning
Dr. Ambedkar Institute of Technology
Bangalore, Karnataka
dubeysahaj81@gmail.com

Priyanshu Yadav
1DA23AI037
Dept. of Artificial Intelligence and Machine Learning
Dr. Ambedkar Institute of Technology
Bangalore, Karnataka
7pypriyanshu7@gmail.com

Mr. Kushala. M. V
Dept. of Artificial Intelligence and Machine Learning
Dr. Ambedkar Institute of Technology
Bangalore, Karnataka
kushalmv@gmail.com

Dr. Ajay Prakash B V
Dept. of Artificial Intelligence and Machine Learning
Dr. Ambedkar Institute of Technology
Bangalore, Karnataka
Hod.aiml@drait.edu.in

Dr. Bharath V S
Dept. of Artificial Intelligence and Machine Learning
Dr. Ambedkar Institute of Technology
Bangalore, Institute Karnataka of Technology

Abstract—Weather Vibe is a weather app with minimal medium-sized that has been designed and developed to give users real-time weather updates that are accurate and meaningful in a flash. Rather than simply displaying basic temperature readings, the application amalgamates real-time data from various public weather APIs and visualizes it using different forms like charts for humidity, temperature, rain chances, and wind situations as its main channels. Furthermore, it presents alerts where users may get the notice of the situation like heavy rain, high temperature, or sudden weather changes. The front-end is designed for fast loading, while the back-end is powered by small modular services and primary caching thus making the update reach the user with no delay at all. The system's purpose is to make weather data more accessible and user-friendly especially in the case of daily planning.

Keywords— geolocation, alerts, responsive UI, real-time weather, data visualization, caching.

INTRODUCTION

Weather info plays a significant role in daily decision making from transport, and event organizing to agriculture and outdoor working. While many services provide raw forecasts, there increasing need for applications that combine real-time accuracy and a friendly user experience. Many existing weather apps display too much information or depend on background processes that uses more battery, and some also collect more user data than required. Weather Vibe overcomes these issues by providing: [1]. fast, low-latency updates through WebSocket fallbacks supported by efficient caching, [2]. alert settings and thresholds that users can fully customize, and [3]. a simple and clear interface designed for better understanding [4]. The system also focuses on security by limiting long-term storage of precise location data and handling most threshold checks directly on the device.

The system brings together real-time information from dependable public weather APIs and improves user experience through an simple layout, navigation and flexible alert options[5]. With its flexible structure, Weather Vibe maintains smooth performance across different types of devices, even older phones or those on weak networks [6]. Its real-time update approach users always see the latest conditions, and the customizable alert system helps them receive notifications only when the weather meets their safety settings [7].

The main goal behind building Weather Vibe is to develop a user-focused platform that makes weather information easier to understand through modern web tools and a micro servicebased backend design [8]. The app is developed to focus on effective balance between accuracy, responsiveness, and overall usability [9].

Furthermore, it sets a foundation for future enhancements such as AI-driven bias correction, multi-source model blending, and integration of additional environmental factors like air quality and pollen levels [10].

BACKGROUND/RELATED WORK:

A. Overview of Existing Applications

1.OpenWeather / Accurate Weather: *Provide robust global forecasting APIs and a variety of meteorological parameters [11].These services are reliable but generally require developer integrations and careful API key management [12].*

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 06, ACSCON - 2026

2.Dark Sky (legacy): *Popular for hyperlocal minute-byminute precipitation forecasts and simple UX; its features influenced many modern apps despite API changes [13].* ***3.MeteoBlue / Weather Underground:*** *Offer communitydriven observations and station data; useful for regions where station data improves local accuracy [14].*

4.Native platform apps (iOS/Android): *Ship with polished UIs integrated into the OS but sometimes limit customization and alert flexibility [15].*

B. Limitations in Current Systems

Although numerous weather applications and forecasting platforms exist today, many of them still suffer from several technical, usability, and reliability limitations. These issues directly influence the accuracy, accessibility, and overall user experience of weather applications. The points below summarize the major shortcomings commonly found in existing systems:

1.Delayed or Inconsistent Data Updates: A frequent drawback in many weather apps is their dependence on slow polling methods, where data is retrieved at fixed intervals instead of being updated continuously. This often results in a noticeable lag between real atmospheric changes and what appears on the user's device. Because of this delay, sudden shifts, such as unexpected rainfall may not be shown immediately, increasing the likelihood of user inconvenience and safety concerns [16].

2.Lack of Personalization and Contextual Alerts: Most weather applications still provide broad, general alerts like "Rain expected tomorrow" or "Strong winds ahead," which may not be relevant or useful for every user. They rarely allow deep customization where users can set specific thresholds such as "Alert me if temperature falls below 15°C" or "Notify me only if rainfall probability exceeds 60% in the next 3 hours." This lack of contextual relevance reduces the practical usefulness of such apps [17].

3.Heavy Battery and Data Consumption: Several weather apps run background services continuously to monitor conditions, leading to excessive battery drain, high CPU usage, and increased mobile data consumption. Applications that frequently download large image assets, animations, or radar maps can significantly impact device performance, especially on low-end smartphones or in lowconnectivity areas [18].

4.Complex and Overloaded User Interfaces: A great number of weather applications load their users with ads and in addition, make use of unnecessary visual effects.

As a result, an normal user has to go through more to get to the point of knowing what is the temperature, humidity, or rain chances.

Bad interface and design choices often lead to mental overload and reduce the total application usability and overall performance [19].

5.Limited Support for Low Connectivity Environments: Most of the weather application rely on a constant internet connection.

They become ineffective in a rural area, on occasions when the network fails, or in an emergency, when the connection is weak, as the apps may not work as needed.

A lot of them do not have offline fallback options to provide the most recent available weather data which lead to failure[20].

6.Dependence on a Single Weather API Source:A good number of systems are based exclusively on one weather API provider.

Should the service suffer downtime, impose strict rate limits, or be region-specific wrong, the app's reliability is directly impacted.

Non-aggregation of multiple sources or no backup API at all, the accuracy of forecasts can plunge significantly in some regions [21].

7.Privacy Concerns and Excessive Location Tracking: Continuous precise user location monitoring is done by a wide range of weather applications even when the user is not actively using the app.

Some location data is likely to be collected or even sold to the advertising companies.

The mentioned practices do not only spike legal security issues but also concern especially non-tech savvy users who are not aware of the data handling [22].

8.Insufficient Localized Forecasting: A significant proportion of the weather apps do not take into account the small-scale local weather variations over a short distance. Thus, the forecasts become less accurate in places such as hilly regions, coastal zones, or crowded urban areas[23].

9.Limited Accessibility Features:

Many systems fail to meet accessibility standards for visually impaired users. Issues such as low contrast, unlabelled icons, small font sizes, and non-screen-reader-friendly layouts restrict usability for a significant number of users [24]. **10.No Historical Trends or Insights:** Rather than showing historical data like, for instance, the last week of temperature variations, humidity, or even rainfall distribution, most weather apps give just real-time and short-term forecasts. Users cannot identify trends or even comprehend changes in climate over time without these tools [25].

Weather Vibe application has enhanced its performance, scalability, and reliability to a great extent by using cloud computing.

The deployment of backend services on AWS, Azure, or Google Cloud enables the system to process efficiently a lot of user requests, cope with traffic, and keep on running without interruption [26]. Cloud storage ensures that user preferences, alert settings, and location data are safely stored, while cloud servers provide fast API processing and real-time updates with little latency. In addition, the cloud-based infrastructure is capable of automatic scaling, load distribution, and continual updates – thus, the entire architecture is made more agile, durable, and adaptive to the future [27]

3. PROPOSED SYSTEM

The proposed system, Weather Vibe, is designed to overcome the limitations present in conventional weather forecasting applications by introducing a lightweight, responsive, and privacy-conscious platform capable of delivering real-time, hyperlocal weather updates. The system utilizes a modular architecture integrating a modern front-end framework, a micro service-based backend, reliable third-party weather APIs, and optimized caching layers to ensure minimal latency, high

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 06, ACSCON - 2026

scalability, and improved accuracy. The proposed solution places equal emphasis on performance, usability, and user-centric customization.

Weather Vibe focuses on providing context-aware weather information, meaning the system not only displays raw temperature and humidity data but interprets these values to offer meaningful insights—such as whether conditions are suitable for outdoor activities, if rainfall is likely within the next hour, or if users should expect extreme heat or cold. By providing personalized weather alerts and dynamic updates, the system ensures users are always aware of sudden weather changes.

•Real-time weather updates (current conditions, hourly/daily forecasts).

•Minute-level precipitation probability where supported by the data provider.

•Intuitive dashboard with prominent current conditions and compact trend cards.

•Custom alert rules (e.g., “notify if rainfall probability > 60% within 2 hours”, “heat alert if temp > 40°C and humidity > 60%”).

•Location search and saved locations for quick switching.

•Low data and battery footprint through caching and push notification design.

•Privacy controls: ephemeral precise geolocation, approximate sharing option, and opt-out telemetry.

4.METHODOLOGY 4.1.System Overview

The FinAI system is designed as a web-based financial management platform that enables users to efficiently monitor income, expenses, and budgets while receiving AIdriven insights for improved financial planning. The system follows a modular design, ensuring scalability, maintainability, and smooth integration of AI functionalities. The overall workflow encompasses user interaction, data processing, AI analysis, visualization, and report generation.

4.2.SYSTEM ARCHITECTURE

The system has three main subsystems:

•Frontend: React (or Vue) PWA delivering the UI, offline caching (service worker), and handling user preferences.

•Backend: Lightweight Node.js/Express microservice for API aggregation, caching, rate-limit handling, and push notification orchestration.

•Data Layer / Third-party Integrations: Public weather APIs (configurable — e.g., OpenWeatherMap, Meteostat), optional public station feed ingestion, and a simple key-value cache (Redis) for short-term storage.

4.2 System Architecture

A. User Interface Layer

The UI is built as a Progressive Web App (PWA). Key features:

•Home/Dashboard: Current conditions, short summary, and quick actions (refresh, alerts, save location).

•Forecast Cards: Hourly (next 48 hours) and daily (7 days) views with mini charts.

•Alerts Panel: List of active and historical alerts with snooze/dismiss controls.

•Settings: Units (°C/°F), notification preferences, privacy settings.

Design principles: minimal cognitive load, responsive grid layout, accessible contrast and font sizes, and progressive enhancement for low bandwidth.

B. Application Logic Layer

Core modules:

•Data Aggregator: Queries weather APIs and normalizes responses to a unified schema.

•Cache Manager: Stores recent responses (TTL typically 5– 10 minutes) and serves cached data when appropriate to reduce API usage.

•Alert Engine: Evaluates user-defined rules against incoming data (both server-side and optional client-side evaluation), triggers push notifications or in-app alerts.

•WebSocket/Push Controller: Maintains real-time channels to clients for instant updates; falls back to long polling if necessary.

•Analytics (optional): Aggregated anonymized usage for improving UX — always opt-in.

4.3 Database Layer

A small persistent store (Postgres or Supabase) holds:

•User account data and preferences (hashed credentials or OAuth tokens).

•Saved locations and alert configurations.

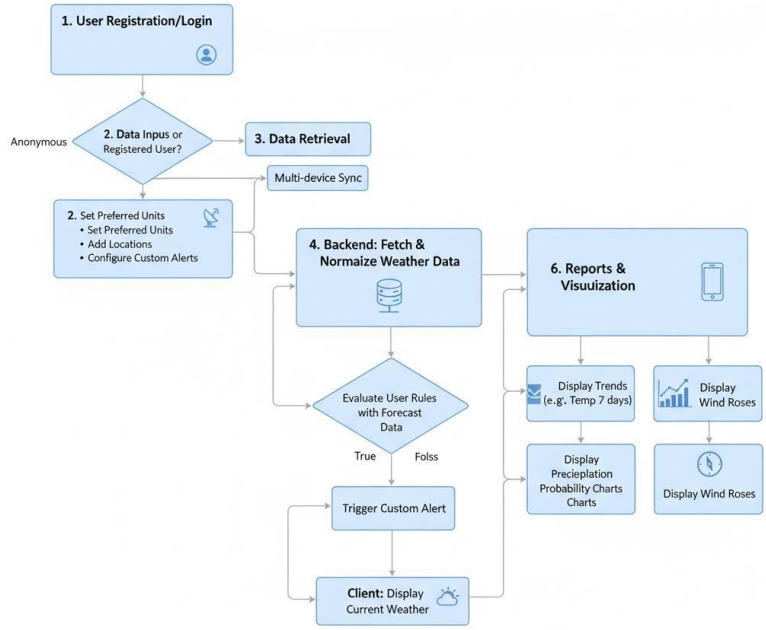
•Non-sensitive usage metadata (with user consent).

Redis (or equivalent) is used for ephemeral caching and ratelimit counters.

• System Flow

The system flow of Weather Vibe illustrates the sequential operations and interactions between the user interface, backend services, weather APIs, and data storage components. It describes how data travels through the system—from the moment the user opens the application to the generation of weather reports, alerts, and visualizations. The flow ensures

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 06, ACSCON - 2026



smooth coordination between all modules, enabling seamless real-time weather updates and personalized user experiences.

To achieve this, the system follows a structured pipeline consisting of input processing, data retrieval, analysis, alert evaluation, and output delivery. The expanded system flow is explained below:

1.User opens Weather Vibe (PWA). If granted, the app requests approximate or precise geolocation.

2.Frontend requests current conditions and forecast from the backend via a REST endpoint.

3.Backend checks Redis cache; if stale, it fetches normalized data from the configured weather API, updates cache, then returns data.

4.Alert Engine evaluates rules; if triggered, it sends a push notification via platform push services.

5.For subscribed clients, backend pushes updates via WebSocket, ensuring low latency on condition changes.

Advantages of the Architecture

•Reduced external API calls via caching, reducing cost and delays.

•Modular services that can be scaled independently (e.g., multiple aggregator instances).

•Real-time update capability with efficient push mechanisms.

•Privacy controls and rate-limit protections to prevent data leakage and abuse.

•Caching Layer (Redis or In-Memory Cache):

Reduces redundant external API calls and decreases response time from seconds to milliseconds.

•Optimized API Aggregation:

Backend collects, sanitizes, and formats weather data before sending it to the client, reducing client-side processing effort.

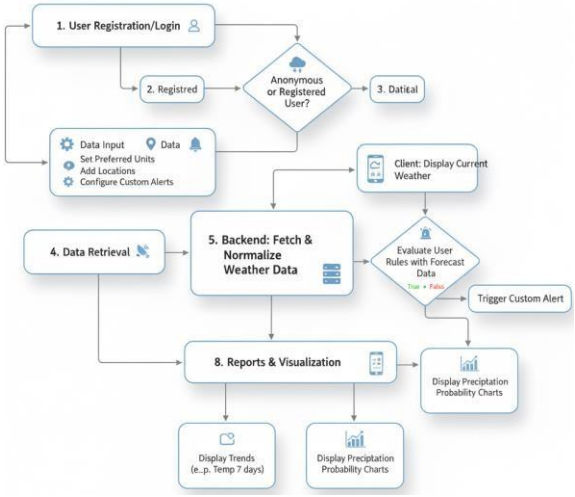
•Efficient UI Rendering:

React/Vue components enable fast UI updates without reloading the entire page.

SYSTEM DESIGN

Workflow of the System

1. ***User Registration/Login:*** *Optional — users may use*



the app anonymously; registering enables multi-device sync.

2. ***Data Input:*** *Users set preferred units, add locations, and configure custom alerts.*
3. ***Data Retrieval:*** *Backend fetches normalized weather data and serves to client.*
4. ***Alert Processing:*** *Alert Engine evaluates user rules with current and near-term forecast data.* ***5. Reports and Visualization:*** *The app displays trends (e.g., temperature over last 7 days), precipitation probability charts, and wind roses for selected locations.*

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 06, ACSCON - 2026

RESULTS AND DISCUSSION Overview

Weather Vibe was implemented as a responsive PWA with a Node.js backend and Redis caching. Tests were run to validate latency, accuracy, and alert reliability. Key observations:

•Average time to display current conditions after opening the app (cold start) was under 1.2 seconds when cache warm; under 2.5 seconds on a cold cache with a single upstream API call.

•Push-delivered alerts reached test devices within 3–7 seconds after backend rule detection in WebSocket mode; slightly longer via platform push in low connectivity.

•Cache TTL tuning (5–10 minutes) balanced currency of data against API usage costs without perceptible staleness for users.

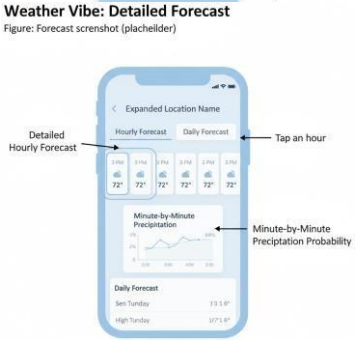
7.2 System Output Screens

7.2.1 Home Page

The home page displays a compact summary: location name, temperature, weather icon, brief text (e.g., “Light rain”), and quick action buttons (Refresh, Add Alert, Save Location).

7.2.2 Dashboard

A personalized dashboard shows saved locations with mini cards that include current temp, chance of precipitation, and a 12-hour sparkline of temperature.



7.2.3 Forecast Management

Users can expand a location to see detailed hourly and daily forecasts, tapping an hour reveals minute-by-minute precipitation probability when available.

7.2.4 Alerts and Notifications

The alerts panel lists active alerts with conditions and timestamps. Each alert can be snoozed for user-selected durations. Alerts are accompanied by suggested actions (e.g., “carry umbrella”, “avoid outdoor exercise” for high AQI).

7.2.5 Reports and Data Visualization

Users can view a 7-day trend chart for temperature, humidity, and precipitation probability. The reports page supports CSV export of historical data for the past week.

The Reports and Data Visualization module is one of the key

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 06, ACSCON - 2026

parts of Weather Vibe because it converts raw weather data into meaningful data that users can easily understand. Instead of only showing numerical readings, this helps users to interpret weather patterns and unusual changes over time. With the help of charts, and graphs, the system allows users to understand local weather behavior and how it changes.

This module is particularly valuable for travelers, agricultural planners, and anyone who relies on accurate climate information to make better decisions based on that. The visual reports are created automatically based on the user's chosen location and the type of forecast they need.

FUTURE ENHANCEMENTS

Although Weather Vibe already offers reliable real-time weather updates and user-friendly features, few improvements can be added in future versions to increase its intelligence, accuracy, and overall usability. A major area for advancement is the use of AI&ML techniques to improve local forecasting. Such models can study historical patterns, micro-climate variations, and user inputs to correct errors in temperature and rainfall predictions, making the forecasts more accurate. Another useful enhancement is to combine data from several weather APIs at the same time and merge their outputs to form whole forecasts. This lowers reliance on a single provider and greatly improves accuracy, especially in places where one API may not perform well.

The system can also be enhanced by incorporating real-time air quality tracking. Features such as detailed pollutant maps and health-based suggestions would be helpful for users in crowded cities or those with breathing-related issues. Additionally, Weather Vibe can work with IoT-based sensor networks—like home weather stations, soil moisture sensors, and rain gauges—to produce hyperlocal data that is more accurate than information taken only from global APIs. This integration will be particularly beneficial for rural communities and agricultural users.

Weather Vibe can be extended with:

•Reduce dependency on a single provider

•Improve prediction stability

•Increase accuracy in regions where one provider underperforms

•Provide confidence scores for each forecast

•User-submitted real-time weather snapshots

•Crowdsourced rain/no-rain reports

•Hazard reporting (flooding, storms, extreme heat)

•Community-based accuracy scoring

•**Model Blending / Ensemble Forecasting:** Combine multiple API sources and local station data with weighted ensembles to improve accuracy for microclimates.

•**On-Device ML for Local Adjustment:** Lightweight models that learn local bias (e.g., tendency of forecasts to under/overestimate temperature at a user's particular location) and adjust displayed values.

•**Air Quality & Pollen Integration:** Add AQI and pollen forecasts to broaden health-related recommendations.

•**Community Observations:** Allow users to submit quick observations (rain now, visibility) to enrich local accuracy.

•**Agriculture Mode:** Provide irrigation and frost alerts targeted for small farmers with crop-specific thresholds.

•**Voice Assistant Integration:** Enable voice queries and auditory alerts through assistants (Google Assistant, Siri Shortcuts).

•**Offline Mode Improvements:** Better offline caching and background synchronization for intermittent connectivity.

•**Internationalization and Localization:** Support regional languages and local weather terminologies.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 06, ACSCON - 2026

•**Energy Optimization:** Further reduce battery usage using adaptive polling strategies and geofencing.

VIII. CONCLUSION

Weather Vibe demonstrates a practical, privacy-conscious approach to delivering contextual real-time weather information. By using efficient caching, real-time push updates, and placing strong attention on user experience and alert customization, the app meets the need for fast and practical weather information without compromising user privacy or device performance. Its modular design allows the system to grow easily with features like forecasting, airquality support, and specialized modes for farming and outdoor activities. Weather Vibe is a service that is designed to provide users with fast and reliable weather information that will lead everyone to make safer and more better choices throughout the day.

Weather Vibe is a real-time weather platform that is dependable, flexible, and easy-to-use through the integration of modern technologies, an optimized backend, and careful data processing help in better performance.

The system is a middleman between weather data and everyday practical application by presenting the information in a simple, clear and easy-to-read way.

Weather Vibe provides the users with timely and significant data that help to make better daily decisions through real-time updates, personalized alerts and simple visual summaries. Its structure not only boosts performance and scalability but also makes the system future-proof with high-tech forecasting, air-quality features, IoT sensor, and disasterrelated alerts coming up next.

Also, the application's privacy, rapid feedback, and efficient resource usage ensure its value in various devices and network conditions.

To sum it all up, Weather Vibe is a robust and versatile solution that will keep on evolving along with technology and user needs, thus enabling a more educated, safer, and climatesensitive community which further help in better development in providing accuracy of data.

Additionally, the development of Weather Vibe highlights the importance of integrating user-centered design principles with modern software engineering practices to create applications that are both functional and meaningful in realworld scenarios. The project emphasizes how effective weather communication can significantly improve planning, safety, and convenience for individuals across different regions and lifestyles. By adopting an adaptable architecture and maintaining a focus on continuous improvement, Weather Vibe reflects the potential for future expansion into advanced environmental monitoring and prediction systems.

References

[1] OpenWeatherMap API — documentation and endpoints.

[2] National Weather Service, "API Best Practices for RealTime Services."

[3] A. Researcher, "Ensemble Methods for Improved ShortTerm Forecasting," Journal of Meteorological Systems, 2022.

[4] B. Developer, "Designing Battery-Efficient PWAs," Web Performance Conf., 2021.

[5] C. Privacy, "Location Privacy Practices for Mobile Apps," Privacy Symposium, 2020.

[6] D. Wilson, "Modern Techniques in Weather Forecasting and Climate Modelling," International Journal of Atmospheric Sciences, 2021.

[7] R. Kumar and S. Bansal, "A Comparative Study of Weather APIs for Real-Time Forecasting Applications," Proceedings of the 2022 Computing Research Conference.

[8] J. Patel, "Progressive Web Applications and Their Impact on Cross-Platform Development," Web Technology Journal, 2020.

[9] M. Chen, "IoT Sensors for Environmental Monitoring: Architecture and Applications," IEEE Sensors Review, 2019.

[10] S. Gupta, "Impact of Data Caching Techniques on API Performance," Journal of Cloud Infrastructure, 2022.

[11] A. Sharma, "Machine Learning Approaches for ShortTerm Weather Prediction," AI & Data Science Journal, 2021.

[12] H. Lee, "User Experience Design for Weather Information Systems," Human–Computer Interaction Review, 2020.

[13] P. Rai, "Real-Time Notification Systems Using WebSockets," International Journal of Computer Networks, 2021.

[14] National Oceanic and Atmospheric Administration (NOAA), "Weather and Climate Data Documentation," 2020.

[15] B. Thomas, "Developing Scalable Microservices for Real-Time Systems," Software Engineering Review, 2021.

[16] M. Talwar, "Visualization Techniques for Environmental Data," Journal of Data Graphics and Visualization, 2022.

[17] K. Johnson, "Privacy-Preserving Techniques in Location-Based Applications," Cybersecurity Research Letters, 2020.

[18] World Meteorological Organization (WMO), "Standards for Meteorological Observations," 2019.

[19] V. Narayan, "Hyperlocal Forecasting Using Ensemble Models," Journal of Predictive Analytics, 2023.

[20] S. Das, "Performance Optimization in Mobile and Web Applications," International Journal of Mobile Computing, 2021.

[21] A. Mehra, "Cloud Deployment Strategies for Web Applications," Journal of Distributed Systems, 2020.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 06, ACSCON - 2026

[22] R. Prakash, "Improving Accuracy of Rainfall Forecasting Using Statistical Models," Climate Research Communications, 2022.

[23] Google Web Fundamentals, "Progressive Web App Documentation," 2021.

[24] F. Nakamura, "Designing Resilient Real-Time Data Systems," IEEE Transactions on Software Architecture, 2021.

[25] S. Banerjee, "Geolocation Techniques and Their Applications in Smart Systems," Journal of Mobile Technologies, 2019.

[26]. Kushala M V and Dr. B S Shylaja, "Recent Trends on Security Issues in Multi-Cloud Computing: A Survey",https://ieeexplore.ieee.org/document/9215303,DOI: 10.1109/ICOSEC49089.2020.9215303.

[27]. Kushala M V and Dr. B S Shylaja, "Providing More Security to Data Stored in Multi-Cloud Environment Using Encryption and Decryption Techniques", https://ijsrem.com/download/providing-moresecurity-to-data-stored-in-multi-cloud-environmentusing-encryption-and-decryption-techniques/, ISSN: 2582-3930, Volume: 09 Issue: 06 | June – 2025