



Design and Evaluation of a Real-Time Web-Based Agroclimatological Information System for Central Java

Rizky Dicko Bagus Hendratno^{1*}, Adi Widiatmoko Wastumirad¹, Dodo Gunawan¹

¹ Instrumentation Study Program, College of Meteorology, Climatology, and Geophysics (STMKG), Tangerang City, Indonesia.

DOI: <https://doi.org/10.29303/goescienceed.v7i4.3303>

Article Info:

Received : August, 27 2026
Revised : September, 15 2026
Accepted : September, 27 2026
Published : October, 13 2026

Correspondence:

Rizky Dicko Bagus Hendratno

Phone:

Abstract: Timely agroclimatological information is required to translate high-frequency weather observations into decisions on irrigation, water availability, and seasonal planning. This study designs and evaluates a web-based information system that integrates real-time data from 22 Automatic Weather Stations (AWS) in Central Java. The system was developed iteratively using Agile Scrum and an architecture comprising MQTT, Node-RED, MySQL, REST application programming interfaces, and a map-based web interface. One-minute JSON observations are validated, routed, and stored by Node-RED as they arrive, enabling real-time weather presentation. Daily processing calculates 24-hour rainfall, FAO-56 reference evapotranspiration (ET_o), water balance, and consecutive dry days. A dekadal prediction service generates a rolling one-year rainfall sequence used to determine seasonal onset, while the website displays D+1, D+2, and D+3 rainfall cards as a concise near-term summary. The interface organizes information into Weather Information, Water Balance, and Season Prediction menus. API verification returned HTTP 200 and consistent data for all 22 station tables; all planned functional, integration, boundary, and error-handling scenarios passed. A ten-item user evaluation involving 15 respondents produced 673 of 750 points (89.73%) and a mean score of 4.49/5. The platform demonstrates that an end-to-end web architecture can convert heterogeneous station data into location-specific operational information.

Keywords: Agroclimatological Information System; Automatic Weather Station; MQTT; Web GIS.

Citation: Hendratno, R. D. B., Wastumirad, A. W., & Gunawan, D. (2026). Design and Evaluation of a Real-Time Web-Based Agroclimatological Information System for Central Java. *Jurnal Pendidikan, Sains, Geologi, Dan Geofisika (GeoScienceEd Journal)*, 7(4), 6136–6145. <https://doi.org/10.29303/goescienceed.v7i4.3303>

Introduction

Agricultural decisions depend not only on the availability of meteorological observations but also on whether those observations are converted into understandable, location-specific information. Central Java has diverse terrain, strong rainfall variability, and a large agricultural sector. Raw sensor feeds alone do not tell a user whether recent rainfall offsets atmospheric water demand, how long a dry spell has persisted, or whether the next seasonal transition is approaching. An information system is therefore needed to connect real-time observation, agroclimatological calculation, spatial

presentation, and predictive products in one operational path.

Automatic Weather Stations (AWS) can measure rainfall, air temperature, relative humidity, pressure, wind, and solar radiation reliably and with minimal human intervention (World Meteorological Organization [WMO], 2021). In operational telemetry architectures, MQTT provides a lightweight publish-subscribe protocol that is well-matched to sensor networks with intermittent connectivity (Banks et al., 2019). Node-RED is widely used to orchestrate ingestion workflows, validate sensor payloads, and persist

Email: dickohendratno.08@gmail.com

observations to relational storage (Rattanapoka et al., 2019; Chen et al., 2024; OpenJS Foundation, 2026).

Despite these technological advances, existing literature reveals significant research gaps from a web system design and end-to-end operational integration perspective. First, prior IoT-based weather monitoring platforms (Rattanapoka et al., 2019; Chen et al., 2024) have primarily focused on hardware telemetry, device online/offline status, and raw observation dashboards, without extending their software architecture to compute and deliver secondary agroclimatological indicators such as reference evapotranspiration (ET_0), water balance, and consecutive dry days (HTH) required for farming operations. Second, while agricultural water-balance assessments have been conducted in Indonesia (Asnawi & Hutauruk, 2022; Thahir et al., 2024), they typically rely on offline calculations using historical or modelled data, leaving an absence of web-based platforms capable of dynamically evaluating and mapping daily water surplus/deficit conditions in near-real time directly from AWS data feeds.

Third, dekadal rainfall prediction studies for seasonal onset (Devi et al., 2022) have largely functioned as standalone analytical modeling experiments, lacking an operational web delivery pipeline that continuously transforms predictive models into user-friendly interface components, such as near-term D+1–D+3 cards and spatial seasonal-onset maps. Finally, no integrated web platform has synthesized these multi-cadence services—one-minute sensor telemetry, daily agroclimatological calculation, and rolling dekadal seasonal predictions—into a unified, fault-tolerant web GIS interface covering multiple stations at a provincial scale. Developing an operational web platform capable of bridging these multi-cadence services therefore represents an unaddressed system-design and operational-delivery challenge.

Agroclimatological interpretation adds a second processing layer. Reference evapotranspiration (ET_0) estimates atmospheric evaporative demand, water balance compares rainfall input with ET_0 , and Hari Tanpa Hujan (HTH; consecutive dry days) describes the persistence of rainfall absence (Murmayani & Darwis, 2024). The FAO-56 Penman–Monteith method remains a standard physical basis for ET_0 calculation (Allen et al., 1998), while recent Indonesian studies illustrate the continuing relevance of meteorological water-balance information for water management (Asnawi & Hutauruk, 2022; Thahir et al., 2024). These indicators are more actionable when presented together with current weather and map context.

This article directly addresses these research gaps by designing, implementing, and evaluating an operational real-time web-based agroclimatological information system for 22 AWS locations in Central Java.

Its objectives are to: (1) integrate 22 AWS feeds using MQTT and Node-RED for real-time storage; (2) automate daily ET_0 , water balance, and HTH processing; (3) implement a rolling dekadal prediction service that exposes D+1–D+3 rainfall cards and seasonal onset; and (4) provide a responsive web interface tested for functional correctness, resilience, and user acceptance.

Method

Iterative Development and Requirements

The system was developed using Agile Scrum because acquisition, calculation, storage, prediction, and interface components had to be integrated incrementally. Four implementation sprints were used: (1) AWS integration and database design; (2) agroclimatological calculation; (3) development and evaluation of rainfall-prediction services; and (4) system integration and web-interface development. This iterative decomposition follows Scrum's incremental delivery model and its use in website-based information-system development (Schwaber & Sutherland, 2020; Nova et al., 2022).

Functional requirements covered real-time acquisition and weather information from 22 AWS locations, seven-day historical queries, daily agroclimatological indicators, a one-year dekadal rainfall sequence for seasonal-onset determination, D+1–D+3 rainfall cards for near-term display, interactive location selection, and clear responses to missing or invalid data. Non-functional priorities were modularity, readable spatial presentation, protection against excessive historical queries, and graceful behaviour when the database, API, or station feed was unavailable. These priorities correspond to product-quality concerns such as functional suitability, reliability, performance efficiency, usability, security, and maintainability (International Organization for Standardization & International Electrotechnical Commission [ISO/IEC], 2023).

Data Acquisition and Service Architecture

Each AWS publishes a JSON observation every minute through an MQTT broker. Node-RED subscribes to the relevant topic, extracts station identity, coordinates, observation time, rainfall, temperature, humidity, pressure, wind speed and direction, and solar radiation, and performs basic format and validity checks. Every valid observation is routed and inserted into MySQL as it arrives. The API reads the latest stored record and exposes it as standards-based JSON over HTTP (Bray, 2017; Fielding et al., 2022). Consequently, acquisition continues independently of a user opening the website, while the interface can refresh from the latest persistent observation and present weather information in real time.

Three logical MySQL databases separate operational concerns. The data_aws database contains 22 station tables with a common 19-column structure. The perhitungan_agroklimatologi_db database contains 22 tables with station identity, location, date, 24-hour rainfall, HTH, water balance, and evapotranspiration fields. The prediksi_musim database stores the combined historical rainfall series, the rolling one-year prediction sequence, ZOM-level seasonal-onset results, and generated map data in four principal tables. The first

three predicted dekads are also delivered to the website as D+1, D+2, and D+3 cards. This separation makes provenance clearer and prevents derived products from being confused with raw or aggregated observations. MySQL was selected as a structured persistent store that can be queried consistently by the application layer (Wahyudi et al., 2022; Oracle, 2026). The overall system architecture is shown in Figure 1, while Table 1 summarizes the system layers, update cycles, and user-facing outputs.

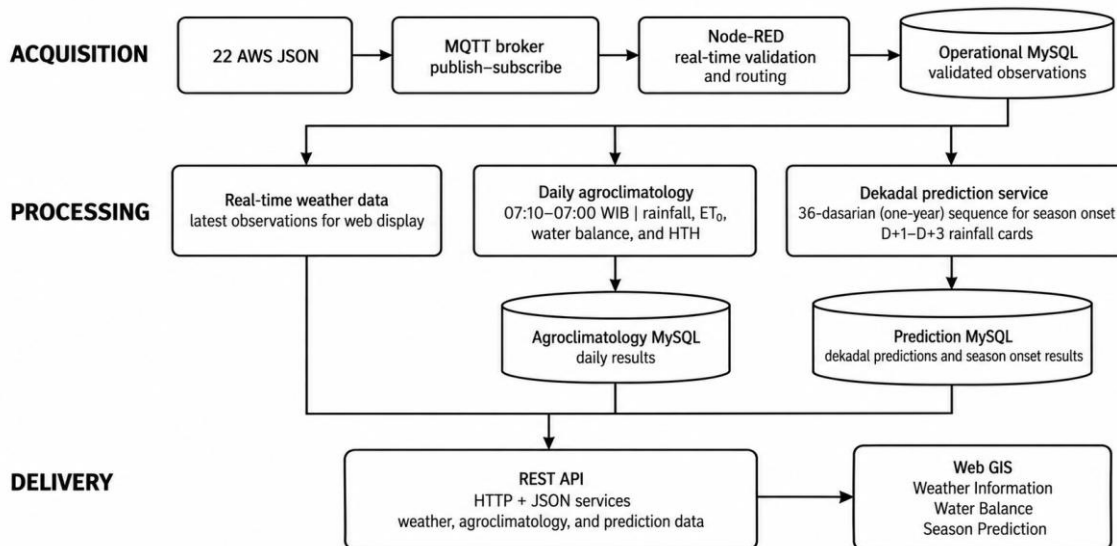


Figure 1. Operational architecture linking observation, derived agroclimatology, prediction services, and web presentation

Table 1. System layers, update cycles, and user-facing outputs

Layer	Primary components	Update cycle	Output
Observation	22 AWS, JSON, MQTT broker	1 minute	Raw meteorological messages
Integration	Node-RED extraction, validation, and routing	Real time	Validated station records
Operational storage	MySQL data_aws	On observation arrival	Real-time weather observations
Agroclimatology	Daily aggregation and FAO-56 calculation	Daily at 07:10 WIB	Rainfall, ET ₀ , water balance, HTH
Prediction	Station model and ZOM onset processor	Every dekad	One-year sequence for onset; D+1-D+3 web cards
Service	REST API	On request	JSON for map, cards, tables, and charts
Presentation	Responsive map-based website	User request / refresh	Weather, Water Balance, Season Prediction menus

Agroclimatological Processing

The daily agroclimatological period runs from 07:10 WIB on day to 07:00 WIB on day + 1 so that all

variables refer to the same observation day. The processor calculates mean temperature, humidity, pressure, wind speed, wind direction, and radiation,

identifies maximum and minimum temperature, and obtains 24-hour rainfall from the 07:00 observation or the maximum valid value in the same period when that record is absent. Wind measured at 10 m is converted to 2 m and solar radiation from W/m^2 to $MJ/m^2/day$ before the FAO-56 terms are evaluated. ET_o is calculated when all required parameters are available; a missing required parameter produces NULL rather than a fabricated value.

Water balance is calculated as rainfall minus ET_o . A negative value indicates deficit, while a value of zero or more indicates adequate or surplus input. HTH increases by one when daily rainfall is below 1 mm and resets to zero when rainfall reaches 1 mm. Results are written with an insert-or-update rule keyed by date, preventing duplicates during recalculation. On the map, red indicates deficit, green indicates surplus conditions with rainfall below 50 mm/day, and blue identifies heavy rainfall of at least 50 mm/day. The interface supplements these colours with numeric values and context-sensitive recommendations.

Interface and Information Design

The Weather Information menu uses an interactive map as the entry point. Selecting a marker exposes station name, location, last-update time, and

principal variables; a detail view adds real-time condition cards and an observation-history table for a user-selected period of no more than seven days. The Water Balance menu shows the same spatial distribution but changes marker meaning to deficit, surplus, or heavy-rain status. Its detail view combines HTH, water balance, ET_o , rainfall, a plain-language condition narrative, and actions such as adjusting irrigation, using mulch, conserving water, or checking drainage.

The Season Prediction menu presents dekadal rainfall and seasonal status based on updated BMKG Seasonal Zones (ZOM) (Badan Meteorologi, Klimatologi, dan Geofisika [BMKG], 2022). The prediction service generates a rolling one-year dekadal rainfall sequence, which is scanned to determine the next seasonal onset. Rainy-season onset is defined as the first of three consecutive dekads with rainfall ≥ 50 mm per dekad, while dry-season onset is the first of three consecutive dekads with rainfall < 50 mm per dekad.

The website displays D+1, D+2, and D+3 rainfall cards for near-term information, together with the current season, estimated next-season onset, and a ZOM-based onset map. Interactive mapping supports spatial navigation (Leaflet, 2023; Wahyuni, 2025). Figures 2–4 show the main information displayed after a location is selected.

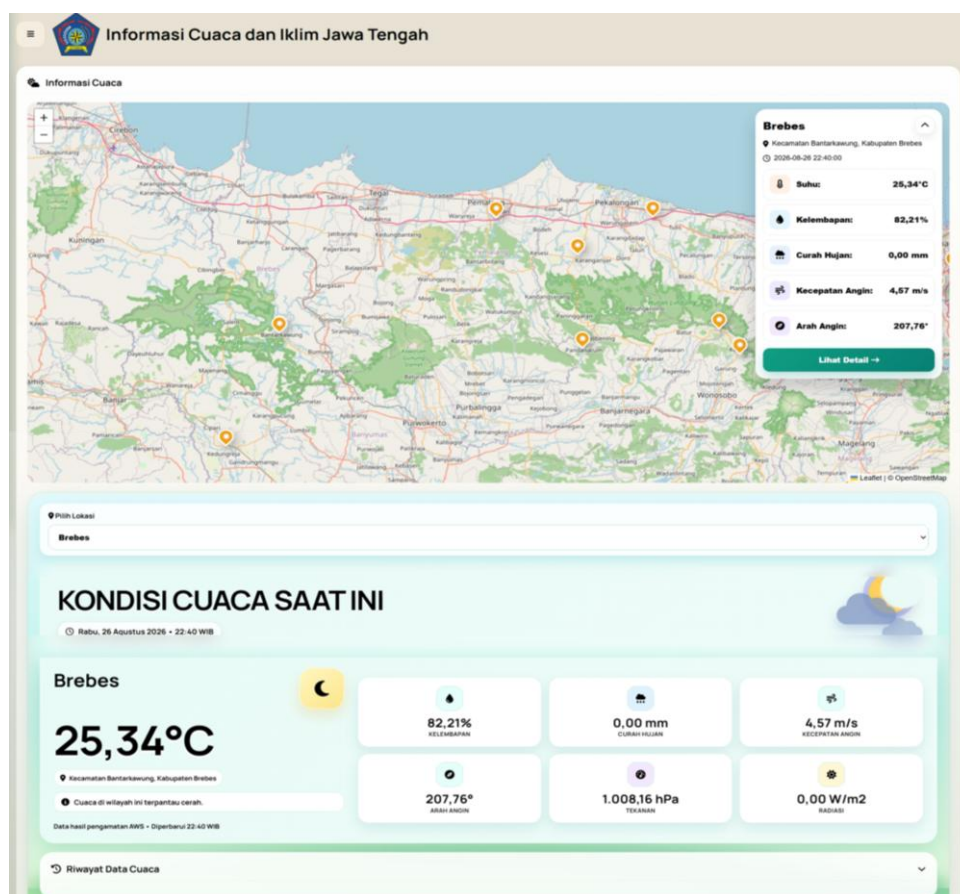


Figure 2. Weather Information menu showing real-time variables for the selected station

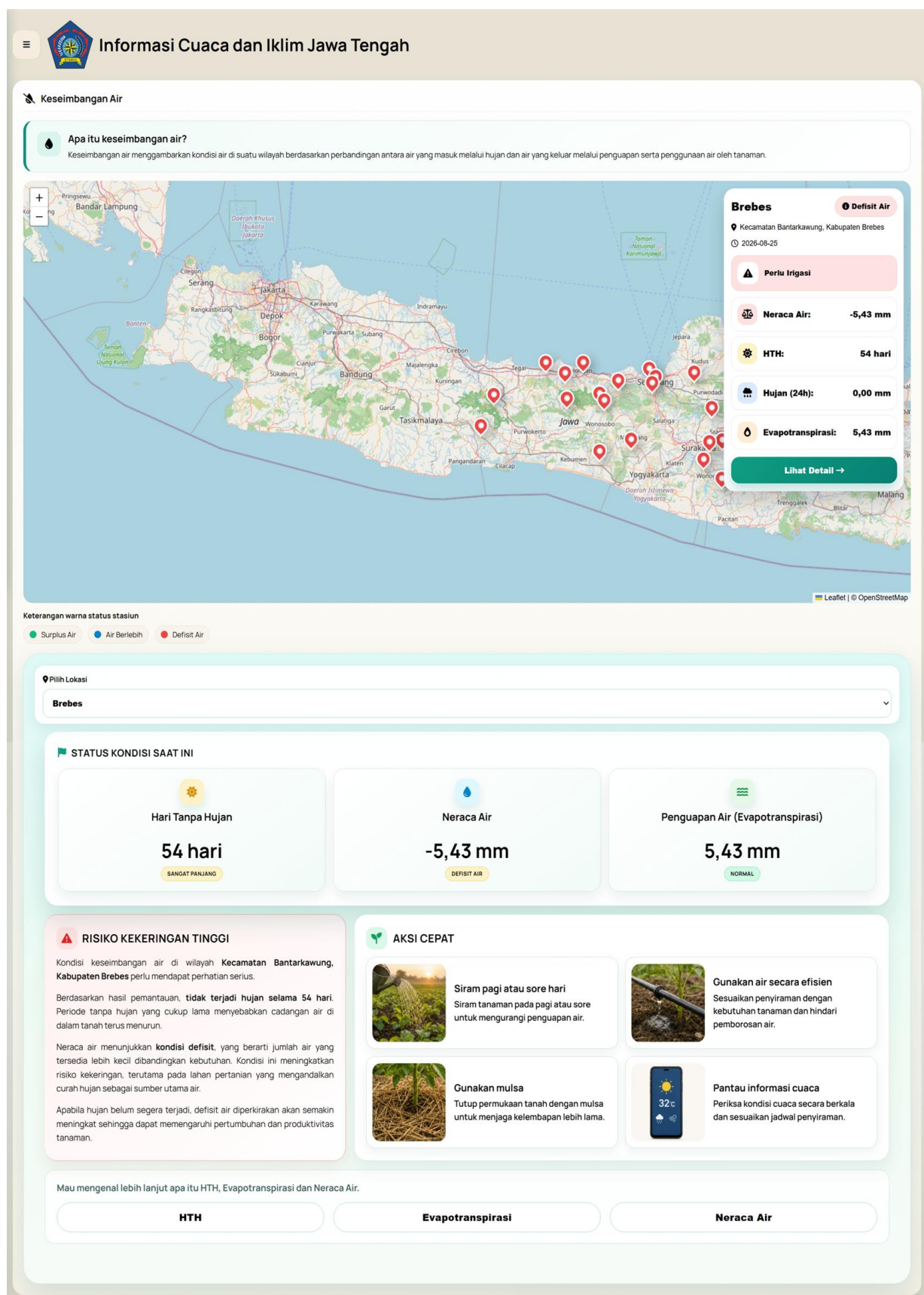


Figure 3. Water Balance menu showing HTH, water balance, evapotranspiration, condition interpretation, and recommended actions

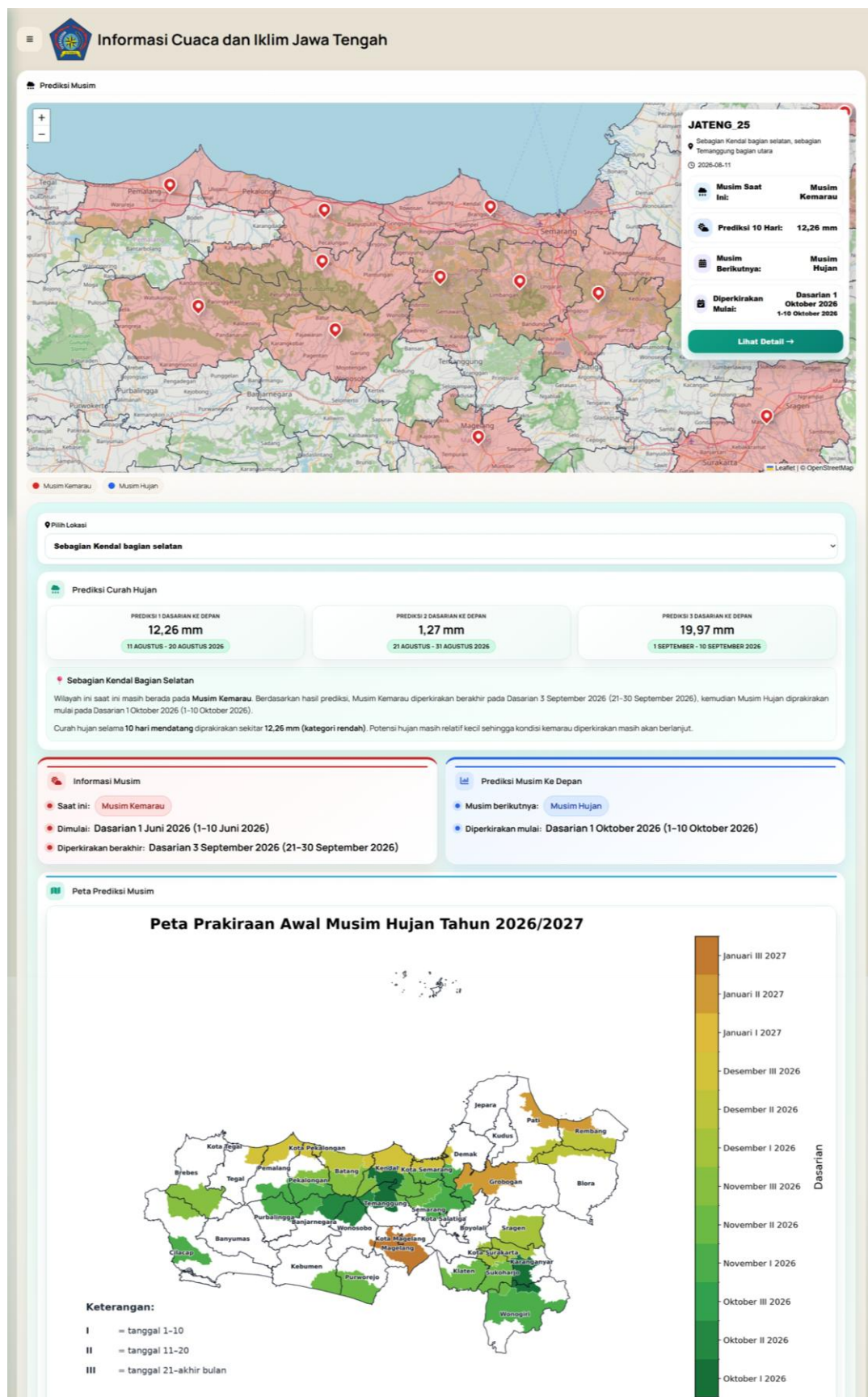


Figure 4. Season Prediction menu showing D+1-D+3 rainfall cards and seasonal-onset information derived from the one-year prediction sequence

Evaluation Design

Evaluation combined data-path verification, black-box functional tests, failure and boundary scenarios, and a user questionnaire. API verification compared the latest MySQL record with the returned JSON; this is consistent with the need to test response status, payload structure, data correctness, and exceptional behaviour in REST services (Ehsan et al., 2022). Functional tests covered page loading, menu changes, station markers, weather details, agroclimatological indicators, D+1-D+3 rainfall cards, one-year seasonal-onset information, database reads, and navigation without a full page reload. Failure tests disconnected the database/API, stopped an AWS feed, supplied empty and anomalous parameters, and requested more than seven days of history.

User acceptance was evaluated through a structured questionnaire consisting of respondent demographics, ten closed Likert-scale statements (ranging from 1 = strongly disagree to 5 = strongly agree), and one open-ended question for qualitative feedback. Respondents were selected using purposive sampling to reflect diverse potential user groups of an agroclimatological web portal, comprising 15 participants from student, private-sector employee, and entrepreneurial backgrounds. This sample size conforms to established usability engineering guidelines by Nielsen (1994), which establish that 5–15 participants are sufficient to uncover more than 80% of usability and interaction problems in software evaluations. Inclusion criteria required participants to: (1) be experienced in operating web browsers on desktop and mobile devices; (2) complete interactive exploration across all three functional menus (Weather Information, Water Balance, and Season Prediction); and (3) complete the evaluation questionnaire comprehensively. While formative in

nature under ISO 9241-11 usability criteria (International Organization for Standardization [ISO], 2018), this group provided balanced perspective across varying technical competencies.

Result and Discussion

Operational Integration and Functional Behaviour

The MQTT-Node-RED-MySQL path successfully handled all 22 AWS station tables. The API returned HTTP 200 and JSON containing station identity, coordinates, observation time, and meteorological parameters consistent with the latest stored records. The website then rendered these data in maps, cards, and tables. Daily processing also created persistent records for rainfall, ET_0 , water balance, and HTH, demonstrating that the platform could transform operational observations into derived information instead of merely relaying sensor values. All planned functional and integration scenarios passed. The three menus loaded their corresponding data, navigation changed the active view without corrupting another menu, and historical retrieval respected the location and date choices. The Season Prediction menu separated the display horizon from the analytical horizon: D+1-D+3 values were rendered as near-term cards, whereas the onset date and seasonal status were obtained from the complete one-year dekadal sequence. The use of separate data, calculation, and prediction stores made the API contracts easier to trace. More importantly, the interface preserved provenance: current weather derives from real-time AWS observations processed by Node-RED, water-balance information from the daily processor, and season information from the dekadal prediction service. The results of the functional, boundary, and failure-handling verification are summarized in Table 2.

Table 2. Summary of functional, boundary, and failure-handling verification

Test group	Scenarios	Observed response	Status
API integration	Read 22 AWS tables and compare latest records	HTTP 200; identity, time, coordinates, and parameters matched stored data	Passed
Core interface	Open site and access three menus	Map, navigation, weather, water balance, and prediction views displayed	Passed
Data functions	Read agroclimatology, history, D+1-D+3 cards, and one-year onset data	Correct database products were returned and rendered	Passed
API/database outage	Disconnect service before page load	Warning and retry control shown; base map or last available data retained	Passed
AWS offline	Stop MQTT delivery from a station	User informed that the latest station observation was unavailable	Passed
Missing/anomalous value	Empty rainfall; pressure = 99.99 hPa	Dash displayed; value not changed to zero or used in calculations	Passed
Query boundary	Request history longer than seven days	Request blocked before server processing; user asked to revise dates	Passed

Graceful Failure and Data Integrity

The four failure scenarios were handled without a total interface failure. If the API or database was unavailable, the website displayed a connection warning, provided a retry action, and retained the base map or last available data. If a station stopped publishing, the user was told that the latest observation had not arrived. Empty and out-of-range sensor values were represented with a dash and excluded from calculations that depended on the affected parameters. This behaviour is preferable to substituting zero, which would be a valid rainfall value and could falsely reset or extend an agroclimatological condition.

The seven-day limit on detailed observation history is a simple but effective boundary control. It prevents an excessively large browser request and communicates the constraint before server processing. These tests establish basic functional resilience, but they do not substitute for production tests of concurrent users, sustained message load, database growth, authentication, authorization, encrypted transport, backup recovery, or malicious input.

User Acceptance

The questionnaire produced 673 of 750 possible points, equivalent to 89.73%, with an overall mean of

4.49/5. Across 150 closed-item responses, 81 (54.00%) were 'strongly agree', 61 (40.67%) 'agree', and eight (5.33%) neutral; no respondent selected disagreement. The highest item mean was 4.67 for the usefulness of warnings and recommended actions in the Water Balance menu. The lowest was 4.33 for the statement that the website was fit to support weather, water-balance, and season information.

Open responses requested additional activity recommendations, dark mode, adjustable text size, multilingual and voice support, improved visual polish, refinement of a five-day forecast view, and clearer presentation of seasonal predictions. The one-year dekadal prediction horizon had been included since the initial system implementation to determine seasonal onset; therefore, the feedback concerned how this existing information was communicated to users rather than the addition of a new forecast horizon. The website presents D+1-D+3 rainfall cards as a concise near-term summary while retaining the complete one-year sequence for seasonal-onset analysis. The remaining suggestions identify accessibility and forecast communication as priorities for subsequent sprints. The mean score for each questionnaire item is presented in Figure 5.

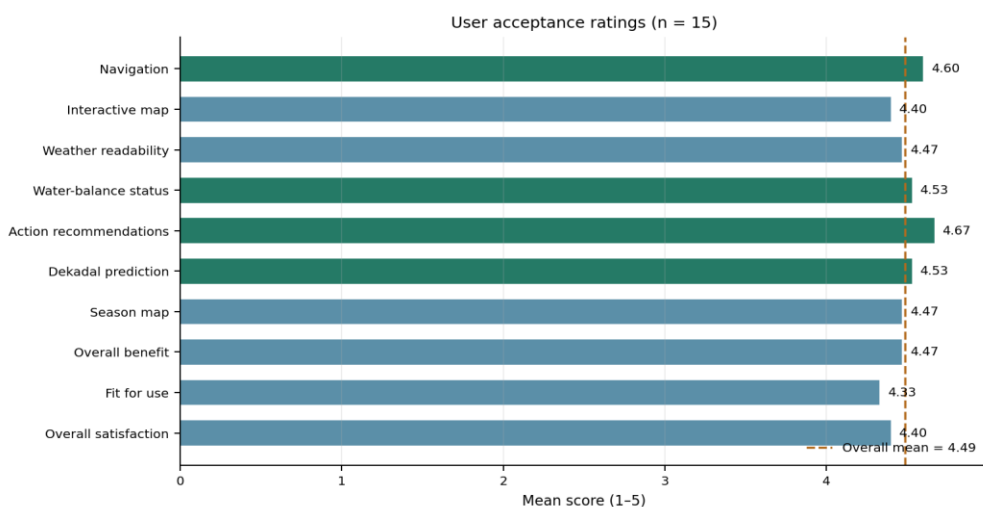


Figure 5. Mean rating for each closed questionnaire item; the dashed line shows the overall mean

Implications and Limitations

The main system contribution is the integration of four service rhythms: one-minute AWS messages, real-time Node-RED processing and weather presentation, daily agroclimatological assessment, and a one-year seasonal outlook updated every dekad. Keeping these products in one interface reduces the fragmentation that occurs when users must consult separate raw-data, climatology, and forecast tools. The distinction between the one-year analytical horizon and the D+1-D+3

presentation cards supports two information needs: seasonal-onset planning and rapid reading of the nearest rainfall outlook. Spatial navigation identifies the station context, while cards, tables, narratives, and recommendations deliver the information required for interpretation. The colour encoding is reinforced with text and numerical values, reducing reliance on colour alone.

The evaluation remains limited in scope. Implementation and testing were conducted on a local

server, so response time, availability, and resource use under production traffic were not measured. The 15-person convenience sample is too small to represent farmers, extension officers, forecasters, and water managers across Central Java, and task-completion time was not recorded. A production study should recruit role-specific participants, use scenario-based tasks, assess accessibility, and measure effectiveness, efficiency, and satisfaction (ISO, 2018). Before public operational release, verification should also cover authentication, authorization, input validation, audit logging, data retention, backup recovery, and service monitoring using an established web-security baseline such as OWASP ASVS, with encrypted transport based on TLS 1.3 (OWASP Foundation, 2025; Rescorla, 2018).

Conclusion

A real-time web-based agroclimatological information system was successfully implemented for 22 Central Java AWS locations using MQTT, Node-RED, MySQL, REST services, and a map-based interface. The system integrates weather observations processed in real time by Node-RED with daily rainfall, ET_0 , water balance, HTH, and season-prediction information in three coherent menus. Seasonal onset is determined from a rolling one-year dekadal rainfall sequence, while D+1, D+2, and D+3 cards provide a concise view of the nearest forecast periods on the website. All planned functional, integration, boundary, and failure-handling scenarios passed, while the initial user evaluation reached 89.73% and 4.49/5. These results support the feasibility of the architecture as an information-delivery platform. The next development stage should prioritize production performance, security, broader role-based usability testing, accessibility, and transparent communication of forecast uncertainty.

Acknowledgements

The authors thank Sekolah Tinggi Meteorologi Klimatologi dan Geofisika for academic support and the Meteorology, Climatology, and Geophysics Agency for access to the AWS and climatological data used in the system.

References

- Allen, R. G., Pereira, L. S., Raes, D., & Smith, M. (1998). Crop evapotranspiration: Guidelines for computing crop water requirements (FAO Irrigation and Drainage Paper No. 56). Rome, Italy: Food and Agriculture Organization.
- Asnawi, M., & Hutauruk, R. C. H. (2022). Perhitungan neraca air meteorologi di Waduk Jatiluhur berdasarkan data curah hujan dan penguapan. *Buletin GAW Bariri*, 3(2), 16–24. doi: <https://doi.org/10.31172/bgb.v3i2.71>
- Badan Meteorologi, Klimatologi, dan Geofisika. (2022). *Pemutakhiran Zona Musim Indonesia Periode 1991–2020*. Jakarta, Indonesia: BMKG.
- Banks, A., Briggs, E., Borgendale, K., & Gupta, R. (Eds.). (2019). MQTT Version 5.0. OASIS Standard. Retrieved from <https://docs.oasis-open.org/mqtt/mqtt/v5.0/os/mqtt-v5.0-os.html>
- Bray, T. (2017). The JavaScript Object Notation (JSON) Data Interchange Format (RFC 8259). RFC Editor. doi: <https://doi.org/10.17487/RFC8259>
- Chen, C., Wu, S., Huang, B., Huang, C., & Yang, C.-F. (2024). Web-based Internet of Things environmental and lighting control and monitoring using Node-RED, MQTT and Modbus communications within an embedded Linux platform. *Internet of Things*, 27, Article 101305. doi: <https://doi.org/10.1016/j.iot.2024.101305>
- Devi, N. M. M. C., Bayupati, I. P. A., & Wirdiani, N. K. A. (2022). Prediksi curah hujan dasarian dengan metode Vanilla RNN dan LSTM untuk menentukan awal musim hujan dan kemarau. *Jurnal Edukasi dan Penelitian Informatika (JEPIN)*, 8(3), 405–411. doi: <https://doi.org/10.26418/jp.v8i3.56606>
- Ehsan, A., Abuhaliqa, M. A. M., Catal, C., & Mishra, D. (2022). RESTful API testing methodologies: Rationale, challenges, and solution directions. *Applied Sciences*, 12(9), Article 4369. doi: <https://doi.org/10.3390/app12094369>
- Fielding, R., Nottingham, M., & Reschke, J. (2022). HTTP semantics (RFC 9110). RFC Editor. doi: <https://doi.org/10.17487/RFC9110>
- International Organization for Standardization. (2018). *Ergonomics of human-system interaction – Part 11: Usability: Definitions and concepts* (ISO Standard No. 9241-11:2018). Geneva, Switzerland: ISO.
- International Organization for Standardization, & International Electrotechnical Commission. (2023). *Systems and software engineering – Systems and software Quality Requirements and Evaluation (SQuARE) – Product quality model* (ISO/IEC Standard No. 25010:2023). Geneva, Switzerland: ISO.
- Leaflet. (2023). Leaflet 1.9.4 API reference. Retrieved from <https://leafletjs.com/reference.html>
- Murmayani, & Darwis. (2024). *Agroklimatologi*. Bandung, Indonesia: Widina Media Utama.
- Nielsen, J. (1994). Estimating the number of subjects needed for a thinking aloud test. *International Journal of Human-Computer Studies*, 41(3), 385–397. doi: <https://doi.org/10.1006/ijhc.1994.1065>

- Nova, S. H., Widodo, A. P., & Warsito, B. (2022). Analisis metode Agile pada pengembangan sistem informasi berbasis website: Systematic literature review. *Techno.Com*, 21(1), 139–148. doi: <https://doi.org/10.33633/tc.v21i1.5659>
- OpenJS Foundation. (2026). Node-RED documentation. Retrieved from <https://nodered.org/docs/>
- Oracle. (2026). MySQL 8.4 reference manual. Retrieved from <https://dev.mysql.com/doc/refman/8.4/en/>
- OWASP Foundation. (2025). OWASP Application Security Verification Standard 5.0.0. Retrieved from <https://owasp.org/www-project-application-security-verification-standard/>
- Rattanaopoka, C., Chanthakit, S., Chimchai, A., & Sookkeaw, A. (2019). An MQTT-based IoT cloud platform with flow design by Node-RED. In 2019 Research, Invention, and Innovation Congress (RI2C) (pp. 1–6). IEEE. doi: <https://doi.org/10.1109/RI2C48728.2019.8999942>
- Rescorla, E. (2018). The Transport Layer Security (TLS) Protocol Version 1.3 (RFC 8446). RFC Editor. doi: <https://doi.org/10.17487/RFC8446>
- Schwaber, K., & Sutherland, J. (2020). The Scrum guide. Retrieved from <https://scrumguides.org/docs/scrumguide/v2020/2020-Scrum-Guide-US.pdf>
- Thahir, A. P. L., Pratiwi, E. P. A., & Jayadi, R. (2024). Evaluasi neraca air untuk Daerah Irigasi Payaman, Kabupaten Gunung Kidul, Provinsi DIY. In *Simposium Nasional Teknologi Infrastruktur* (pp. 1–6). Yogyakarta, Indonesia: Universitas Gadjah Mada.
- Wahyudi, J., Asbari, M., Sasono, I., Pramono, T., & Novitasari, D. (2022). Database management education in MySQL. *Edumaspul: Jurnal Pendidikan*, 6(2), 2413–2417. doi: <https://doi.org/10.33487/edumaspul.v6i2.4570>
- Wahyuni, R. (2025). Implementasi Sistem Informasi Geografis untuk pemetaan dan pengelolaan sumber daya alam di kawasan pertanian. *Journal of Human and Education*, 5(1), 545–550. doi: <https://doi.org/10.31004/jh.v5i1.2227>
- World Meteorological Organization. (2021). Guide to instruments and methods of observation (WMO-No. 8, 2021/2018 ed.). Geneva, Switzerland: WMO.