Literature, Data, and Technology Review

Project Title: Dengue Fever Weekly Risk Alert System (Thailand)
Team: Group 8

PART 1: LITERATURE REVIEW

1.1 Introduction

Dengue fever remains one of the most critical vector-borne diseases in Southeast Asia, with Thailand experiencing recurring outbreaks that strain healthcare systems and communities. The importance of this research lies in developing an early warning system that can predict dengue risk at the district and provincial level, enabling proactive intervention measures. By integrating climate data, behavioral signals, and historical disease patterns, our project addresses a critical gap in public health preparedness.

A comprehensive review of existing literature is necessary to understand the established relationships between environmental factors and dengue transmission, evaluate proven forecasting methodologies, and identify best practices from similar warning systems. This review will ground our approach in evidence-based research while highlighting how our project contributes novel value through its integration of multiple data streams and focus on actionable, district-level alerts for Thailand.

1.2 Thematic Organization of Literature

Theme 1: Climate and Weather Drivers of Dengue Transmission

Climate variables particularly temperature, rainfall, and humidity have been consistently identified as primary drivers of dengue transmission dynamics. Soneja et al. (2021) provide a comprehensive review demonstrating that temperature affects mosquito development rates, survival, and biting frequency, while also influencing viral replication within the mosquito vector. Their analysis shows that optimal transmission occurs within specific temperature ranges (20-30°C), with both extremely high and low temperatures reducing transmission efficiency.

Sugeno et al. (2023) examined environmental factors and dengue incidence in Lao PDR, a country with similar tropical climate patterns to Thailand. Their study revealed that moderate rainfall increases dengue risk by creating breeding sites, but excessive rainfall can flush out larvae, creating a non-linear relationship. Additionally, they found that humidity levels above 60% significantly correlate with increased mosquito survival rates. These findings are particularly relevant for our Thai context, as both countries share similar monsoon patterns and mosquito ecology.

The methodology employed by Sugeno et al. (2023) used multivariable regression models with temporal lag structures, finding that weather variables from 1-4 weeks prior showed the strongest associations with dengue cases. This supports our approach of incorporating lagged weather features in our predictive model.

Theme 2: Large-Scale Climate Signals and ENSO Effects

Beyond local weather patterns, large-scale climate phenomena significantly influence dengue dynamics across regions. Tian et al. (2025) recently published findings in Nature Communications demonstrating that El Niño-Southern Oscillation (ENSO) creates region-wide temperature anomalies that modulate dengue risk. Their analysis of multi-decade data from Asia and the Americas revealed that El Niño events correlate with increased dengue incidence due to warmer-than-average temperatures that enhance vector competence and expand the geographic range of suitable mosquito habitats.

Johansson et al. (2009) provided foundational work on this topic through their PLOS Medicine study, which examined multiyear climate variability and dengue transmission. They demonstrated that ENSO indicators could serve as seasonal leading indicators for dengue risk, with El Niño years showing 2-3 times higher dengue incidence in affected regions. Importantly, they noted that ENSO effects vary by geography, with some regions experiencing stronger correlations than others.

These studies highlight the importance of incorporating ENSO indices as potential features in our model, particularly for seasonal forecasting. While our primary focus is week-scale predictions, understanding ENSO phases could improve our model's ability to anticipate high-risk seasons.

Theme 3: Spatiotemporal Dynamics and Regional Synchronization

Understanding how dengue spreads across geographic areas is crucial for effective warning systems. Van Panhuis et al. (2015) conducted a landmark study published in Proceedings of the Royal Society B, analyzing region-wide synchrony and traveling waves of dengue across Thailand over multiple decades. Their research revealed that Thai provinces exhibit spatiotemporal synchronization of dengue outbreaks, with traveling waves moving across the country at predictable speeds.

This synchronization has important implications for our modeling approach. The authors demonstrated that province-level models must account for temporal dependencies and spatial autocorrelation to achieve accurate predictions. They found that outbreaks in neighboring provinces often occur within 2-4 weeks of each other, suggesting that spatial features (neighboring province cases) could improve forecast accuracy.

Their work strongly supports our decision to build province- and district-granular models with time-aware validation strategies. The existence of traveling waves also suggests that a successful early warning in one district could trigger enhanced surveillance in neighboring areas.

Theme 4: Digital Behavior Signals as Early Indicators

The integration of internet search behavior into disease surveillance represents an innovative approach to early warning systems. Li et al. (2022) demonstrated the value of combining Google Trends data with environmental variables for dengue forecasting. Their study, which integrated Google Earth Engine, artificial intelligence, and Google Trends (GT), showed that search activity for dengue-related terms often precedes clinical case reporting by 1-2 weeks. This temporal advantage makes search data particularly valuable for early warning purposes.

Puengpreeda et al. (2020) conducted Thailand-specific research published in Engineering Journal, developing a weekly forecasting model for dengue hemorrhagic fever using Google Trends and meteorological data. They tested both Thai keywords ("ไข้เลือดออก," "ยุงลาย") and English terms ("dengue"), finding that Thai-language searches showed stronger correlations with subsequent cases in urban areas. Their model achieved significant improvements in forecast accuracy when combining search data with weather variables compared to weather data alone.

The Thai-focused nature of Puengpreeda et al.'s (2020) work provides direct evidence for our approach. They demonstrated that search behavior correlates with dengue awareness and care-seeking behavior, which typically increases as community incidence rises. However, they also noted limitations in rural areas with lower internet penetration, suggesting our model should potentially weight search signals differently based on urban/rural classification.

1.3 Synthesis and Gaps in Existing Research

Comparing these studies reveals several commonalities. First, all research confirms that weather variables are fundamental predictors of dengue risk, with consensus on the importance of temperature, rainfall, and humidity. Second, there is growing recognition that multi-source data integration combining environmental, climatic, and behavioral signals improves forecast performance beyond single-source approaches.

However, important differences exist in methodological approaches. While Soneja et al. (2021) and Sugeno et al. (2023) employed traditional statistical regression methods, Li et al. (2022) and Puengpreeda et al. (2020) incorporated machine learning techniques including random forests and gradient boosting. The latter approaches generally demonstrated superior predictive performance, supporting our selection of tree-based ensemble methods (XGBoost, LightGBM) as primary modeling techniques.

A critical gap identified across this literature is the limited focus on actionable, operational warning systems. Most studies focus on retrospective analysis or model development without addressing implementation challenges, user interface design, or integration with public health workflows. Our project specifically addresses this gap by designing a dashboard-based system with clear risk classifications, top-K district rankings, and explainable predictions suitable for non-technical public health staff.

Additionally, while Puengpreeda et al. (2020) conducted Thailand-specific research, their model operated at the national or regional level. Our district-level granularity represents a significant advancement, as local health officers require more geographically precise information for targeted interventions like larval control and community education campaigns.

1.4 Conclusion

The reviewed literature establishes a strong foundation for our dengue early warning system. Research consistently demonstrates that weather variables, large-scale climate signals, and digital behavior patterns provide predictive power for dengue risk. The evidence of spatiotemporal synchronization across Thai provinces supports our province/district-level modeling approach with time-aware validation.

Our project contributes to the existing body of knowledge in three key ways. First, we provide district-level granularity that enables more targeted public health responses than existing regional models. Second, we integrate multiple data streams (satellite weather data, Google Trends, and epidemiological records) in a unified framework specifically optimized for Thailand. Third, we emphasize interpretability and operational deployment through our dashboard design and SHAP-based explanations, bridging the gap between academic forecasting models and practical public health tools.

By building on established methodologies while addressing identified gaps in granularity, integration, and operationalization, our system will provide Thai public health authorities with a practical tool for early dengue intervention, ultimately contributing to reduced disease burden and improved health outcomes aligned with SDG 3.

1.5 Reference

- **Cummings, D. A. T.,** et al. (2004). *Travelling waves in the occurrence of dengue haemorrhagic fever in Thailand.* Nature, 427, 344–347.
- **Johansson, M. A.,** et al. (2009). *An emerging disease due to a decline in human immunity? Multiyear climate variability and dengue.* PLoS Medicine, 6(12): e1000168.
- **van Panhuis, W. G.,** et al. (2015). *Region-wide synchrony and traveling waves of dengue across Southeast Asia.* Proc. R. Soc. B, 282: 20150591.
- **Tian, H.,** et al. (2025). *ENSO modulation of global dengue risk* (Nature Communications). Use to ground your ENSO mechanism narrative (check exact title/details when finalizing).
- **WHO.** (2024/2025). *Dengue – Key facts* (factsheet for concise epidemiology/definitions).

PART 2: DATA RESEARCH

2.1 Introduction

This data research submission outlines our comprehensive data collection and preparation strategy for the Dengue Fever Weekly Risk Alert System. The research questions we aim to address are:

(1) Can we accurately predict district-level dengue risk 1-2 weeks in advance?
(2) Which combination of weather, climate, and behavioral signals provides the strongest predictive power?
(3) How do these relationships vary across different Thai provinces and seasons?

A thorough exploration of data is necessary because dengue prediction requires integrating heterogeneous data sources with different spatial and temporal resolutions, quality levels, and access mechanisms. Understanding data characteristics, limitations, and preprocessing requirements is essential for building a robust and reliable early warning system.

2.2 Data Sources and Description

2.2.1 Weather and Climate Data

Source: ERA5 Reanalysis (European Centre for Medium-Range Weather Forecasts), GPM (Global Precipitation Measurement), and TRMM (Tropical Rainfall Measuring Mission)

Data Format: NetCDF and GeoTIFF files containing gridded climate variables

Data Size: Approximately 3-10 years of daily observations across Thailand, resulting in ~50-150 GB of raw satellite data

Variables Collected:

- Daily precipitation (mm)
- Mean, minimum, and maximum temperature (°C)
- Relative humidity (%)
- Additional variables: wind speed, solar radiation (for potential NDVI calculations)

Spatial Resolution: ERA5 provides ~31 km resolution; GPM provides ~11 km resolution. Data will be aggregated to district or provincial boundaries using spatial averaging.

Temporal Coverage: 2014-2024 (10 years), aligning with Thailand's transition to digital disease surveillance

Rationale: We selected ERA5 as our primary source because it provides consistent, quality-controlled reanalysis data with global coverage and no missing values. GPM supplements this with higher-resolution precipitation data for Thailand's monsoon patterns. These satellite-based sources are chosen over ground station data due to complete geographic coverage across all Thai districts, including remote areas with sparse weather stations.

Access Method: Open access through Copernicus Climate Data Store (ERA5) and NASA GES DISC (GPM). Data can be programmatically downloaded via APIs.

2.2.2 Google Trends Search Data

Source: Google Trends API (pytrends library)

Data Format: CSV files with weekly relative search volume indices (0-100 scale)

Data Size: ~500-1,000 rows per keyword (weekly observations × number of provinces), totaling ~5-10 MB

Keywords Monitored:

- Thai: "ไข้เลือดออก" (dengue fever), "ยุงลาย" (Aedes mosquito), "ป้องกันไข้เลือดออก" (dengue prevention)
- English: "dengue", "dengue fever", "dengue symptoms"

Spatial Resolution: Provincial level (Google Trends does not provide sub-provincial data for Thailand)

Temporal Coverage: 2014-2024, weekly data aligned to ISO week standards

Rationale: Following Puengpreeda et al. (2020), we selected keywords that showed the strongest correlations with Thai dengue cases. Thai-language keywords are prioritized as they better capture local search behavior. Search data provides a near-real-time behavioral signal that often precedes official case reporting by 1-2 weeks, as people search for symptoms before seeking medical care.

Limitations: Provincial-level resolution means we cannot distinguish district-level search patterns within provinces. Search data may be less reliable for rural provinces with lower internet penetration rates. We will address this by creating an urban/rural indicator variable to potentially downweight search signals in rural areas.

Access Method: Programmatic access via unofficial pytrends Python library, which queries Google Trends. Data is free but rate-limited.

2.2.3 Historical Dengue Case Data

Source: Thailand Ministry of Public Health (MOPH) Report 506 surveillance system and Digital Disease Surveillance (DDS) API (post-2024)

Data Format: CSV/Excel files with weekly disease reports

Data Size: 3-10 years of weekly case counts across 77 provinces, approximately 40,000-100,000 rows

Variables:

- Week of report (ISO week format)
- Province/district code
- Number of dengue fever cases
- Number of dengue hemorrhagic fever (DHF) cases
- Hospitalizations and deaths (where available)
- Age groups and demographic information (if accessible)

Spatial Resolution: Provincial level confirmed available; district-level data availability to be verified during data collection phase

Temporal Coverage: 2014-2024 target range; actual availability depends on public data access

Rationale: This is our primary outcome variable. Weekly resolution aligns with public health decision cycles and intervention planning timelines. The 2024 transition to DDS API may provide improved data quality and timeliness for recent years.

Limitations: Historical data may have reporting delays, with rural areas potentially showing longer lag times between case occurrence and official reporting. Data completeness may vary by province. Case definitions may have changed over time (dengue fever vs. dengue hemorrhagic fever classifications).

Access Method: We will pursue multiple access strategies:

1. Public data from MOPH Bureau of Epidemiology website
2. Academic data sharing agreements if available
3. WHO WPRO dengue surveillance data as a supplement
4. DDS API access if publicly available

If district-level data proves inaccessible, we will proceed with provincial-level modeling and note this as a limitation.

2.3 Data Integration and Preprocessing Strategy

Our preprocessing pipeline will transform raw data from multiple sources into a unified analytical dataset following these steps:

Step 1: Temporal Alignment

- Convert all data sources to ISO week format (YYYY-WW)
- Aggregate daily weather data to weekly summaries (mean temperature, total precipitation, mean humidity)

- Align all datasets to Sunday-Saturday week definitions for consistency

Step 2: Spatial Alignment

- Map gridded satellite data to district/province boundaries using weighted spatial averaging
- Create lookup tables linking province codes across different data sources (MOPH codes, Google Trends region codes, geographic boundaries)

Step 3: Feature Engineering

- **Lag features:** Create 1-4 week lagged values for weather variables (rain_lag1, rain_lag2, etc.) based on mosquito lifecycle timing
- **Rolling statistics:** 2-week and 4-week rolling means and sums for weather variables
- **Wet spell indicators:** Count consecutive days with >1mm rainfall per week
- **Seasonality features:** Week of year (1-52), month, season indicators
- **ENSO index:** Download and integrate monthly Oceanic Niño Index (ONI) values as a climate signal
- **Temporal features:** Holidays, school terms (when disease surveillance may change)
- **Spatial features:** Neighboring province case counts (for spatial autocorrelation)

Step 4: Missing Data Handling

- ERA5 reanalysis has no missing values by design
- Google Trends: interpolate short gaps (1-2 weeks) using linear interpolation; longer gaps flagged for exclusion or imputation with province-specific historical averages
- Case data: investigate missingness patterns; exclude provinces/weeks with unreliable data from training

Step 5: Normalization and Scaling

- Standardize continuous features (z-score normalization) separately for each province to account for geographic variation
- Min-max scaling for bounded features like search indices

Step 6: Target Variable Creation

- Create binary high-risk labels based on multiple definitions:
  - **Percentile approach:** Top 20% of historical case counts for each province
  - **Threshold approach:** Exceeds 3-year moving average by >50%
  - **Epidemic curve approach:** Cases above endemic channel upper threshold
- We will test all three definitions and select the most operationally useful

Step 7: Train/Validation Split

- Time-based split with walk-forward validation (no random shuffling to prevent temporal leakage)
- Initial training: 2014-2020, validation: 2021-2022, test: 2023-2024
- Province-aware splits ensuring no data leakage between neighboring areas

2.4 Initial Data Exploration and Insights

While full data collection is ongoing, preliminary exploration of publicly available data reveals several important patterns:

Seasonality: Thailand exhibits strong seasonal dengue patterns with peaks typically occurring during and immediately after the rainy season (June-August), with a smaller peak in the cool-dry season (November-January). This seasonality varies by region, with southern provinces showing less pronounced seasonality due to year-round rainfall.

Geographic Variation: Bangkok and surrounding central provinces historically report the highest case numbers, but per-capita rates are often higher in northern and northeastern provinces. This suggests our model should learn province-specific patterns rather than assuming uniform national relationships.

Search Behavior Patterns: Preliminary Google Trends analysis shows that dengue-related search spikes correlate with major outbreak years (2013, 2019) and show geographic variation consistent with case patterns. Thai-language searches show higher volumes than English searches, confirming our keyword selection.

Data Quality Considerations: Satellite weather data shows excellent coverage and consistency. Case reporting completeness appears high for provincial-level data but may be more variable at district level. Google Trends data occasionally shows unexpected zeros or spikes that may require outlier detection and handling.

Correlation Insights: Initial correlation analysis suggests:

- Temperature shows positive correlation with cases up to ~30°C
- Rainfall shows complex non-linear relationship (positive up to moderate levels, then decreasing)
- Humidity consistently shows positive correlation with dengue incidence
- Lagged variables (1-4 weeks prior) show stronger correlations than same-week variables, supporting our lag feature strategy

2.5 Conclusion

Our data research establishes a comprehensive multi-source data foundation for the dengue early warning system. By integrating satellite-derived weather data (ERA5, GPM), behavioral signals (Google Trends), and epidemiological surveillance (MOPH reports), we can capture the environmental, behavioral, and disease dynamics necessary for accurate prediction.

Key findings from this data research phase include: (1) all required data sources are accessible through open or public channels, reducing implementation barriers; (2) temporal and spatial alignment procedures are clearly defined and feasible; (3) feature engineering strategy is grounded in dengue ecology literature; and (4) data quality appears sufficient for machine learning, though some limitations exist at district level.

The importance of this data research to our project is foundational successful integration and preprocessing of these heterogeneous data sources directly determines model performance and operational feasibility. Our next steps involve completing data download, implementing the preprocessing pipeline, and conducting comprehensive exploratory data analysis to validate initial insights and refine feature engineering strategies.

2.6 Reference

Thailand surveillance

- **Bureau of Epidemiology, MOPH (Thailand).** *Report 506 weekly notifiable diseases* (historical + current reporting framework).
  WHO Western Pacific/SEARO dashboards or country pages (optional for cross-check).

Climate reanalysis (ERA5)

- **Hersbach, H.,** et al. (2020). *The ERA5 global reanalysis.* QJRMS, 146(730), 1999–2049.
- **Copernicus Climate Data Store.** *ERA5: monthly means/single levels – product documentation and CDS API examples.*

Precipitation (GPM IMERG / TRMM TMPA)

- **NASA GES DISC.** *GPM IMERG V07 data access & product guide.*
- **Huffman, G. J.,** et al. (2007/2010). *TRMM Multi-satellite Precipitation Analysis (TMPA).* (methods/product docs).

ENSO indices

- **NOAA CPC.** *Oceanic Niño Index (ONI): definition and dataset.*

Digital exhaust

- **Google.** *Trends data methodology (normalization, 0–100 scaling, sampling).*
- **pytrends (GitHub).** Unofficial Python client used for automated pulls (document limitations/terms).

Thai administrative boundaries

- **GADM v4.1.** Thailand ADM0–ADM2 shapefiles.
- **HDX COD-AB (OCHA).** Thailand admin levels (geoBoundaries/Kontur alternatives)

PART 3: TECHNOLOGY REVIEW

3.1 Introduction

This technology review examines the computational tools, libraries, and platforms required to build, deploy, and maintain our Dengue Fever Weekly Risk Alert System. Given our project's requirements for geospatial data processing, machine learning, and web-based visualization, a careful technology review ensures we select tools that balance functionality, ease of use, performance, and sustainability.

The importance of this review lies in making informed architectural decisions early in the project lifecycle, avoiding costly technology pivots later. This review is particularly relevant to our goal of creating an operational system that public health staff can use, not just an academic prototype.

3.2 Core Technology Stack Overview

Our technology stack is organized into four layers: data acquisition and processing, machine learning and modeling, explainability and interpretation, and user interface and deployment.

Layer 1: Data Acquisition and Processing Technologies

Python 3.9+ serves as our primary programming language for all data science workflows. Python's extensive ecosystem, readability, and widespread adoption in health analytics make it ideal for collaborative development.

Key Libraries for Data Processing:

Pandas (v2.0+): The fundamental data manipulation library for our tabular data processing. Pandas excels at time series operations, merging heterogeneous data sources, and handling missing data all critical for our preprocessing pipeline. We selected Pandas over alternatives like Polars because of its maturity, extensive documentation, and team familiarity, accepting the performance tradeoff for development speed.

GeoPandas (v0.12+): Essential for processing spatial data and mapping satellite grids to administrative boundaries. GeoPandas integrates Shapely for geometric operations and enables us to aggregate ERA5 grid cells to province/district polygons. Alternative spatial libraries like ArcPy require expensive licensing, making GeoPandas the clear choice for open-source development.

Xarray (v2023.1+): Purpose-built for multi-dimensional array data like satellite NetCDF files. Xarray provides labeled dimensions (time, latitude, longitude) that make working with climate data intuitive and less error-prone than raw NumPy arrays. We chose Xarray specifically for its integration with climate data standards and ability to efficiently chunk large satellite datasets.

Climate Data Extraction:

cdsapi (Climate Data Store API): Official Python interface for downloading ERA5 reanalysis data from Copernicus. This library handles authentication, batch requests, and automatic retries. We selected this over manual data downloads because it enables reproducible, automated data updates.

h5netcdf / netCDF4: Libraries for reading NetCDF file formats common in satellite data. These integrate seamlessly with Xarray for efficient loading of large climate datasets.

Google Trends Access:

pytrends (v4.9+): Unofficial Python API for Google Trends. While not officially supported by Google, pytrends is well-maintained and widely used in epidemiological surveillance research. We acknowledge the risk of API changes but accept this given the lack of an official programmatic interface. We will implement error handling and fallback mechanisms.

3.3 Machine Learning and Modeling Technologies

Scikit-learn (v1.3+): The foundational machine learning library providing preprocessing utilities (StandardScaler, train_test_split), baseline models (Logistic Regression, Random Forest), and evaluation metrics (ROC-AUC, precision-recall). Scikit-learn's consistent API across algorithms simplifies experimentation.

XGBoost (v2.0+): Our primary modeling framework. XGBoost (Extreme Gradient Boosting) is specifically chosen for tabular data problems because it:

- Handles missing values natively
- Provides built-in feature importance measures
- Offers excellent performance on structured data
- Supports custom objectives for imbalanced classification
- Integrates with SHAP for explainability

XGBoost has proven highly effective in epidemiological forecasting competitions and is production-tested in healthcare applications worldwide.

LightGBM (v4.0+): Microsoft's gradient boosting framework as an alternative to XGBoost. LightGBM offers faster training on large datasets through histogram-based learning. We include both libraries to compare performance and select the best-performing model for our specific data characteristics.

Comparison: XGBoost vs. LightGBM vs. Random Forest

Feature	**XGBoost**	**LightGBM**	**Random Forest (sklearn)**
Training Speed	Medium	Fast	Slow on large data
Prediction Speed	Fast	Fast	Medium
Memory Usage	Medium	Low	High
Handling Missing Data	Native	Native	Requires imputation
Hyperparameter Tuning	Complex	Complex	Simpler
Interpretability	Good (with SHAP)	Good (with SHAP)	Good (native importance)

Based on this comparison, we prioritize XGBoost for its balance of performance and robustness, with LightGBM as a backup if training time becomes prohibitive.

Optuna (v3.3+): An automatic hyperparameter optimization framework. Optuna uses Bayesian optimization to efficiently search hyperparameter spaces, significantly reducing tuning time compared to grid search. We selected Optuna over Hyperopt because of its more intuitive API and better visualization tools.

Imbalanced-learn (v0.11+): Specialized library for handling class imbalance, a common challenge in disease prediction where high-risk weeks are rare. Provides SMOTE (Synthetic Minority Over-sampling Technique) and under-sampling strategies. While XGBoost has built-in class weighting, imbalanced-learn offers more sophisticated resampling techniques we may explore.

3.4 Explainability and Interpretation Technologies

SHAP (SHapley Additive exPlanations) (v0.42+): Critical for our requirement to provide "why this area is high-risk" explanations. SHAP provides:

- Model-agnostic explanations based on game theory
- Both global feature importance and local instance explanations
- Integration with XGBoost/LightGBM for fast computation
- Visualization tools (waterfall plots, force plots, summary plots)

SHAP is superior to simple feature importance because it accounts for feature interactions and provides consistent, theoretically grounded explanations. We specifically need SHAP for our dashboard's "top 3 drivers" requirement.

Alternatives Considered:

- **LIME (Local Interpretable Model-agnostic Explanations):** More computationally expensive and less stable than SHAP for tree models
- **Native feature importance:** Simpler but doesn't account for feature correlations

SHAP is the clear choice for production explainable AI in our context.

3.5 Visualization and Dashboard Technologies

Plotly (v5.15+): Interactive visualization library for our web dashboard. Plotly enables:

- Interactive choropleth maps showing district-level risk
- Responsive charts that work on mobile devices
- Hover tooltips for detailed information
- Time slider for exploring historical predictions

We chose Plotly over alternatives (Matplotlib, Bokeh, Folium) because it provides the best balance of interactivity, ease of use, and professional appearance suitable for stakeholders.

Dash (v2.12+): Web application framework built on Plotly. Dash enables us to build interactive dashboards entirely in Python without requiring JavaScript expertise. Key features:

- Component-based architecture for maintainability
- Callback system for interactivity
- Easy deployment options
- Built-in responsiveness

Alternative Considered:

- **Streamlit:** Simpler and faster for prototyping but less flexible for complex layouts
- **Flask + custom JavaScript:** More flexible but requires full-stack development expertise
- **Tableau/Power BI:** Proprietary tools that lack customization and automation capabilities

Dash is selected because it provides the right level of customization while keeping development entirely in Python, reducing complexity for our team and future maintainers.

Folium (v0.14+): Python library for creating Leaflet.js maps. While Plotly handles most visualization needs, Folium is included as a backup for specific geospatial visualizations if Plotly's mapping capabilities prove insufficient.

3.6 Development and Deployment Technologies

Jupyter Notebook / JupyterLab (v4.0+): Primary environment for exploratory data analysis, model development, and documentation. Notebooks enable literate programming that combines code, visualizations, and explanations ideal for collaboration and knowledge transfer.

Git / GitHub: Version control and collaboration platform. Essential for team development, code review, and maintaining project history. We will use GitHub for repository hosting and project management features (issues, pull requests).

Docker (Optional for Deployment): Containerization platform for packaging the dashboard and its dependencies. Docker ensures consistent deployment across different environments. We mark this as optional for the MVP but recommended for production deployment to health department servers.

Deployment Options:

1. **Local deployment:** Dash app running on a local server, suitable for initial testing
2. **Cloud deployment - Heroku/Render:** Free tiers available for hosting Dash apps, easy deployment via GitHub integration
3. **Cloud deployment - AWS/GCP:** More scalable but complex; overkill for MVP
4. **On-premises server:** Deployment to MOPH servers if access granted

We will start with option 2 (Heroku/Render) for MVP demonstration, with option 3 considered if usage scales or if MOPH requests internal hosting.

Scheduled Execution:

GitHub Actions / Cron Jobs: For automated weekly data updates and model retraining. GitHub Actions provides free CI/CD for scheduled data pipelines, while traditional cron jobs can run on any Linux server. We will implement both options depending on final deployment environment.

3.7 Data Storage Technologies

CSV / Parquet Files: For structured tabular data. Parquet is preferred for larger datasets due to columnar storage efficiency and compression. CSV remains useful for human-readable intermediate outputs and smaller files.

SQLite (v3.40+): Lightweight embedded database for storing processed features and predictions. SQLite is ideal for our scale because:

- No separate server setup required
- File-based storage integrates easily with Git LFS
- Sufficient performance for provincial-level weekly data (~50K rows)
- Built-in time series capabilities

Alternative Considered:

- **PostgreSQL:** More powerful but requires separate server management, overkill for our data volume
- **MongoDB:** NoSQL database unsuitable for our structured time series data

SQLite strikes the right balance between simplicity and capability for our MVP.

3.8 Testing and Quality Assurance Technologies

Pytest (v7.4+): Python testing framework for unit tests and integration tests. We will implement tests for:

- Data preprocessing functions
- Feature engineering correctness
- Model loading and prediction pipeline
- API/dashboard functionality

Great Expectations (v0.17+): Data quality validation framework. Essential for production pipelines to catch data quality issues before they affect predictions. We can define expectations like "precipitation values must be non-negative" or "weekly case counts must be integers" that are automatically validated.

3.9 Use Cases and Examples

Similar Projects Using Our Technology Stack:

1. **COVID-19 Dashboards (Johns Hopkins, WHO):** Deployed using similar Plotly Dash architecture for real-time disease tracking. Demonstrates scalability and reliability of Dash for epidemiological dashboards serving millions of users.

2. **Weather Prediction Services:** NOAA and meteorological agencies use XGBoost/LightGBM for short-term weather forecasting, showing these models' capability for time-dependent predictions similar to our use case.

3. **Google Flu Trends Research:** Multiple academic replications have used pytrends + scikit-learn for disease surveillance, validating our approach of combining search behavior with epidemiological data.

4. **Malaria Early Warning Systems (Kenya, Tanzania):** Used Python-based ML pipelines with SHAP explainability for operationalizing vector-borne disease predictions, directly paralleling our application.

These case studies demonstrate that our selected technology stack is production-proven for health surveillance systems at various scales.

3.10 Identified Gaps and Customization Needs

Gap 1: Real-time Data Ingestion Current libraries assume batch processing. For true real-time alerts, we may need to implement:

- Custom API wrappers for DDS system once available
- Data update monitoring and automatic pipeline triggering
- Notification system integration (LINE API for Thai users)

Gap 2: Model Drift Monitoring While we have tools for training and deployment, we lack built-in model performance monitoring over time. We should implement:

- Automated prediction vs. actual case comparison
- Alert system for model performance degradation
- Periodic retraining triggers

Gap 3: Multi-language Support All selected technologies default to English. For MOPH staff usability:

- Dashboard text should be Thai-English bilingual
- This requires custom i18n (internationalization) implementation in Dash
- Documentation must be provided in Thai

Gap 4: Mobile Optimization While Dash is responsive, field health workers may access the system via mobile devices with limited bandwidth. We may need:

- Progressive Web App (PWA) capabilities
- Offline mode for viewing recent predictions
- Lighter visualization options for low-bandwidth connections

3.11 Conclusion

Our technology stack centered on Python, XGBoost/LightGBM for modeling, SHAP for explainability, and Dash for deployment provides a robust, maintainable, and operationally viable foundation for the Dengue Fever Weekly Risk Alert System.

Key takeaways from this technology review:

1. All selected technologies are open-source, reducing cost barriers for long-term maintenance
2. The stack emphasizes Python-centric tools, minimizing language complexity
3. Each technology is chosen based on proven use in similar health surveillance applications

4. The architecture supports both MVP development and future scaling

The importance of these technology choices extends beyond technical implementation. By selecting accessible, well-documented tools with strong community support, we ensure that our system can be maintained and enhanced by future developers, including Thai public health informatics teams.

Our technology selection directly benefits the project by enabling:

- Rapid prototype development (critical for the capstone timeline)
- Model explainability that public health staff can trust and act upon
- Deployment flexibility to accommodate different hosting environments
- Maintainability for long-term operational use beyond our capstone period

This technology foundation positions our project to deliver not just an academic exercise, but a practical tool that can meaningfully contribute to dengue prevention efforts in Thailand.

3.12 References

Wrangling & multidimensional climate data

- **pandas docs.** *Time series & resampling.*
- **xarray docs.** *Dataset/DataArray + NetCDF I/O.*
- **cdsapi docs.** ERA5 programmatic download examples.
- **netCDF4 / h5netcdf docs** (as needed).

Geospatial & interactive maps

- **GeoPandas docs** (choropleths & mapping).
- **Folium docs** (GeoJSON/Choropleth).
- **Plotly docs** (choropleth / Maplibre migration).

Modeling & explainability

- **Chen, T., & Guestrin, C.** (2016). *XGBoost: A Scalable Tree Boosting System.* **KDD’16.**
- **Ke, G.,** et al. (2017). *LightGBM: A Highly Efficient GBDT.* NeurIPS’17
  Lundberg, S. M., & Lee, S-I. (2017). *A Unified Approach to Interpreting Model Predictions (SHAP).* NeurIPS’17.

Hyperparameter search & class imbalance

- **Akiba, T.,** et al. (2019). *Optuna: A Next-generation Hyperparameter Optimization Framework.* KDD’19.