**CHAPTER THREE**

**METHODOLOGY**

**3.1 Chapter Overview**

This chapter outlines the systematic procedures and tools employed to achieve the objectives of the study. It provides a comprehensive description of the proposed simulation environment, data sources, forecasting models, evaluation criteria, and implementation workflow. Given the infrastructural challenges in developing urban settings such as Nigeria, the methodology emphasizes low-cost simulation, synthetic data modeling, and comparative analysis of forecasting techniques. The chosen approach enables replication, scalability, and practical deployment in resource-constrained environments.

**3.2 Research Design**

This study adopts a **simulation-based experimental research design** as a methodological strategy to evaluate the feasibility, performance, and scalability of IoT-enabled energy monitoring and forecasting systems within the context of Nigerian urban environments. Recognizing the financial, logistical, and infrastructural limitations associated with deploying physical IoT hardware at scale—especially in resource-constrained settings—this approach prioritizes the creation of a virtual simulation environment that replicates the core components and behavioral dynamics of a real-world energy monitoring infrastructure. The simulation-based design offers several methodological advantages. First, it provides **controlled experimental conditions**, allowing the researcher to manipulate key environmental variables—such as scheduled blackouts, fluctuating power supply, communication interruptions, and sensor failures—without the risk or cost of physical damage or field disruption. These controlled conditions enable rigorous testing of system resilience and adaptability, which are critical in designing solutions for unstable infrastructure settings.

Second, the design facilitates the **comparative evaluation of forecasting algorithms**, specifically Facebook Prophet and LSTM neural networks, under identical operational scenarios. By standardizing the simulation environment, the study ensures that variations in model performance are attributable to the algorithms themselves rather than to external inconsistencies. This methodological consistency enhances the reliability and validity of the performance metrics generated from the analysis. Third, the simulation platform supports **iterative development and validation**, whereby system components—such as data acquisition modules, forecasting engines, and decision-making logic—can be tuned, modified, and retested repeatedly until optimal configurations are achieved. This iterative refinement is crucial for identifying potential implementation bottlenecks and performance trade-offs prior to committing to physical deployment. It also aligns with agile system development methodologies that emphasize continuous improvement through feedback-driven experimentation. Ultimately, the simulation-based research design serves as both a cost-effective and methodologically robust means of advancing energy innovation in developing regions. It allows for the testing of scalable solutions in a risk-free environment, fosters reproducibility, and lays a strong foundation for future hardware implementation guided by data-driven insights.

**3.3 Simulation Environment and Tools**

The simulation environment for this study was meticulously designed to emulate the structure, behavior, and constraints of a real-world urban energy monitoring system, with particular sensitivity to the infrastructural conditions of Nigerian cities. The setup consists of several integrated modules that collectively reproduce the operational workflow of an IoT-based energy management system—ranging from data generation and transmission to forecasting and decision-making. At the core of the simulation setup is a **virtual sensor network**, which models the deployment of energy consumption sensors across multiple urban households or buildings. These sensors are programmed to generate synthetic energy consumption data, derived from a combination of actual usage patterns (where available) and statistically modeled consumption profiles based on contextual factors such as time of day, seasonality, and user behavior. Importantly, the simulation incorporates **realistic environmental disruptions**, including unscheduled blackouts, sensor dropouts, communication delays, and packet loss—conditions that are common in developing-world urban settings but often overlooked in traditional simulations.

The second component of the setup is the **data transmission and aggregation layer**, which simulates how data from distributed sensors would be transmitted to a central server or edge node for analysis. This layer includes simulated communication protocols and latency models to evaluate how different network conditions—such as bandwidth limitations or temporary outages—affect data integrity and system responsiveness. This module helps examine the resilience of the system under varying degrees of connectivity, which is crucial for deployment in areas with inconsistent internet infrastructure. The third module focuses on the **forecasting engine**, which is built to support the execution and comparison of two primary models: Prophet and Long Short-Term Memory (LSTM) neural networks. Both models are trained on the synthetic dataset generated by the simulation and are tasked with predicting short- to medium-term energy consumption across simulated households. The engine is designed to assess the predictive accuracy, computational efficiency, and fault tolerance of each model under identical conditions, thereby enabling a fair and comprehensive performance comparison.

In addition, the simulation setup integrates a **system control and evaluation dashboard**, which visualizes energy flows, model predictions, error rates, and system responses in real time. This dashboard enables the researcher to monitor system behavior, identify anomalies, and iteratively adjust simulation parameters for improved accuracy and realism. The inclusion of feedback loops also allows for **scenario-based testing**, wherein different grid conditions, user profiles, and load types can be simulated to evaluate how the system would behave under varying stress conditions. Lastly, the simulation environment is underpinned by a **transparent configuration layer**, which documents all underlying assumptions, parameter values, and model settings. This documentation ensures that the simulation can be reproduced or extended by other researchers, contributing to the overall transparency, credibility, and academic value of the work. Overall, the simulation setup serves not only as a testing ground for forecasting models and IoT architectures but also as a proof-of-concept for how energy monitoring systems can be designed with the specific challenges of developing urban environments in mind. It bridges the gap between theoretical modeling and real-world implementation by providing a flexible, cost-effective, and context-aware research platform.

3.3.1 Simulation Tools

To emulate and analyze the performance of the proposed IoT-based energy monitoring and forecasting system, several software tools and libraries were employed, each serving a specific layer within the system architecture. These tools were selected based on their ability to replicate real-world scenarios such as network constraints, data dynamics, and forecasting tasks in a controlled environment.

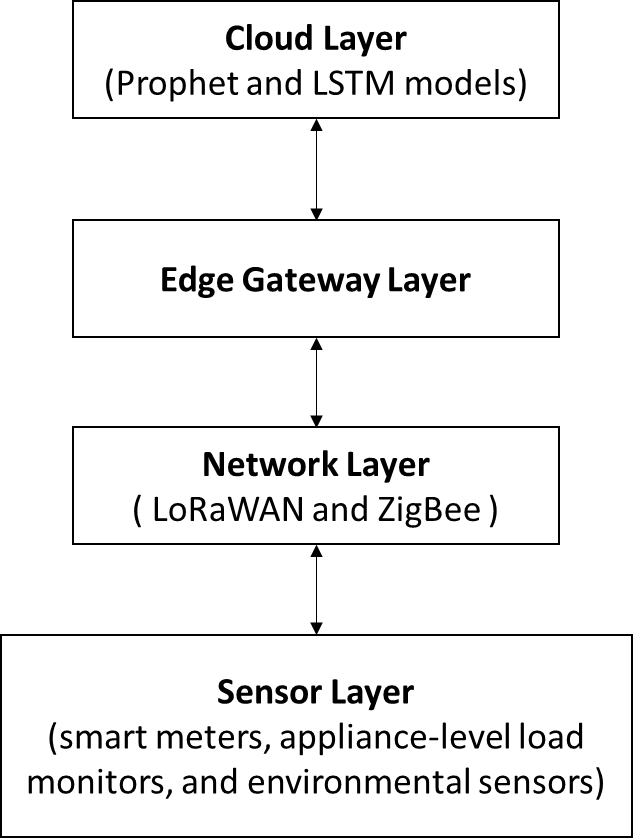
| **Tool** | **Purpose** |
| --- | --- |
| iFogSim (Gupta et al., 2016) | A simulation toolkit designed for modeling fog and edge computing environments. It was used to replicate the hierarchical flow of IoT-generated data from the sensor layer to cloud-based forecasting modules. It enables detailed configuration of fog nodes, data delays, task scheduling, and load balancing mechanisms. |
| IoTNetSim (Salama et al., 2019) | A discrete-event simulator used to emulate network-level operations such as data transmission latency, packet loss, routing failures, and throughput constraints across various protocols (e.g., ZigBee, LoRaWAN). This was essential in evaluating network reliability and delay-sensitive behavior of IoT transmissions. |
| Python (with Prophet, TensorFlow, Keras) | Python served as the main programming environment for implementing and training the forecasting models. Prophet was used for its time series capabilities, while TensorFlow and Keras were leveraged for building and optimizing the LSTM model. These libraries allowed for model tuning, accuracy comparison, and visualization of predictions. |
| Excel, NumPy, Pandas | These tools were used for data preprocessing and trend analysis. Excel was useful for initial data inspection, while NumPy and Pandas enabled efficient handling of large time series datasets, including outlier detection, normalization, and resampling for uniform intervals. Together, they ensured clean, structured inputs for the forecasting engines. |

These tools were integrated to simulate a realistic smart energy environment that includes network-level variability, edge processing delays, and the behavior of forecasting models under multiple load and failure scenarios. To ensure functional integration across the various simulation layers, these tools were orchestrated in a sequential pipeline. First, IoTNetSim simulated the behavior of the network layer, capturing data transmission characteristics such as latency and packet loss across LoRaWAN and ZigBee channels. This simulated sensor data was then routed through iFogSim, which modeled edge processing tasks such as task scheduling, data buffering, and latency optimization at intermediary fog nodes. Finally, the structured outputs were ingested by Python-based forecasting models (Prophet and LSTM) for prediction, evaluation, and visualization. This layered interaction not only preserved system modularity but also enhanced the realism of the simulated environment by capturing how delays, noise, and failure events propagate through each component of the IoT architecture

3.3.2 System Architecture Design

The simulated system was architected to replicate a functional IoT-driven energy monitoring framework, reflecting the operational flow from sensor deployment to cloud-based analytics. It is structured into four primary layers:

* Sensor Layer: This represents the lowest tier of the architecture, consisting of virtual nodes mimicking smart meters, appliance-level load monitors, and environmental sensors (e.g., temperature, light intensity). These nodes generate synthetic or real-time energy consumption data that feeds into the network layer. Their behavior includes periodic sampling, event-triggered reporting, and failure simulation (e.g., missing values due to outages).
* Network Layer: At this layer, communication protocols such as LoRaWAN and ZigBee are modeled to capture the behavior of low-power, low-bandwidth IoT networks commonly deployed in urban smart grid projects. IoTNetSim was used to simulate packet transmission delays, data loss, link failures, and varying bandwidth conditions—providing a realistic sense of data availability and transmission lag across devices.
* Edge Gateway Layer: Acting as an intermediary between the sensor nodes and the cloud analytics layer, the edge gateway simulates preprocessing tasks such as data aggregation, cleaning, temporal buffering, and fault filtering. This design ensures that the cloud layer receives structured inputs while reducing network congestion and latency—a key principle in edge computing for smart infrastructure.
* Cloud Layer: This top-level component hosts the forecasting engines—Prophet and LSTM models—running on virtualized Python environments. It includes functionality for historical data storage, periodic forecasting, model retraining, and report generation. The cloud also logs performance metrics (e.g., MAE, RMSE, MAPE), simulating a real-world scenario where smart city dashboards visualize prediction accuracy and system health.

**

*Figure 3.1: A schematic representation of this architecture*

This modular, multi-tiered design enables high-fidelity testing of the proposed solution under diverse urban operating conditions—ranging from intermittent connectivity to fluctuating energy patterns—without requiring a physical deployment. It also offers a platform for future expansion, such as incorporating demand-response modules or integrating renewable energy **data streams.**

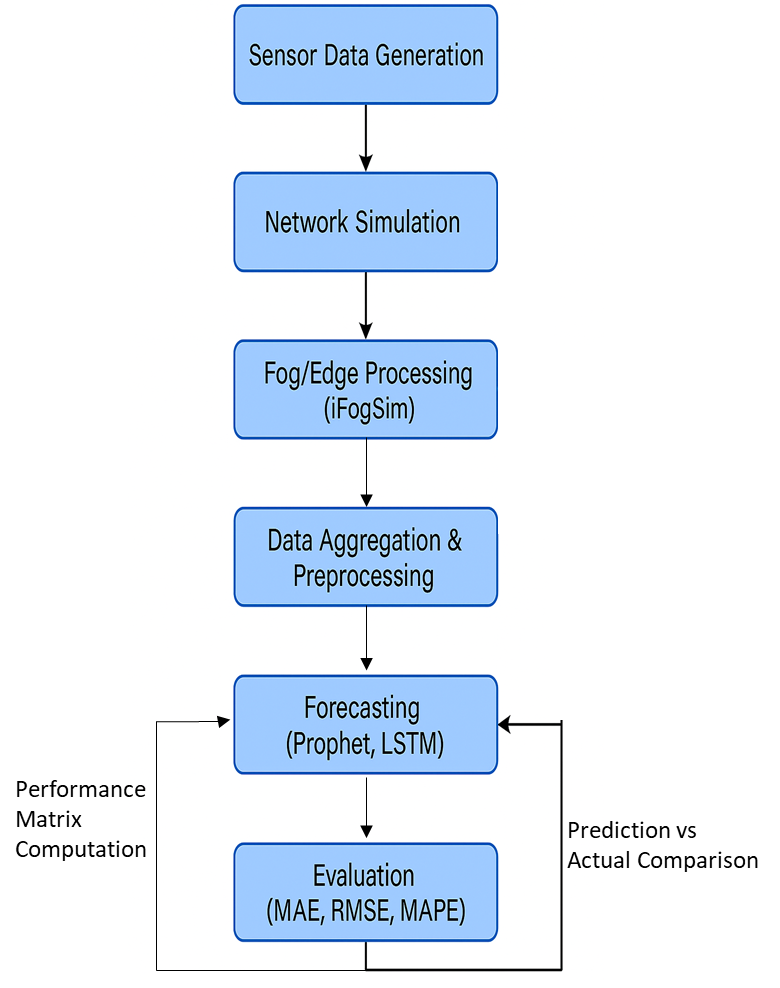
****

Figure 3.2: Workflow of Simulation Tools and Forecasting Pipeline

This diagram (Figure 3.2) illustrates the end-to-end workflow of the simulated IoT-enabled energy forecasting system used in this study. The process begins with **Sensor Data Generation**, where virtual sensor nodes produce synthetic energy consumption data based on behavioral patterns and blackout scenarios. The data then passes through **Network Simulation**, where communication protocols (e.g., LoRaWAN, ZigBee) are emulated to introduce latency, packet loss, and dropouts reflective of real-world transmission conditions in urban Nigerian contexts. The **Fog/Edge Processing** module, modeled via iFogSim, temporarily stores and preprocesses the data at intermediary nodes to reduce latency and bandwidth usage. Subsequently, the data enters the **Data Aggregation & Preprocessing** block, where it is cleaned, resampled, and structured using tools like Pandas and NumPy to prepare for forecasting. Two machine learning models—**Facebook Prophet** and **LSTM (via TensorFlow/Keras)**—receive the cleaned time series as input within the **Forecasting** layer. The output predictions are evaluated in the **Evaluation** module using statistical performance metrics such as MAE, RMSE, and MAPE. Feedback loops are established between the evaluation and forecasting layers, enabling iterative tuning and retraining. Collectively, the diagram captures how different system components and tools interact to support robust, context-aware energy consumption forecasting.

**3.4 Data Collection and Preparation**

The fidelity of any machine learning–based forecasting model is highly dependent on the quality, resolution, and representativeness of the input data it receives. In the context of urban energy systems—especially within infrastructurally constrained environments like many Nigerian cities—reliable and granular consumption data is often limited or entirely unavailable. As a result, this study adopted a hybrid data strategy, combining real-world empirical datasets with meticulously constructed synthetic time series to model the electricity demand landscape in a realistic yet flexible manner. The use of **real-world data** introduces empirical grounding into the simulation framework, ensuring that the trends, seasonal variations, and anomalies captured by forecasting models are not merely theoretical constructs but grounded in observable behavior. Real datasets were sourced from publicly accessible repositories such as the UCI Machine Learning Repository and institutional audits, including the Lagos State Electricity Board’s (LSEB) energy assessment of Magodo district. These records provided concrete data points such as daily fuel consumption, generator capacities, and residential energy loads—critical for shaping the foundational trends in the simulation environment. Supplementary figures were extracted from national grid dashboards and load distribution reports released by organizations like the Transmission Company of Nigeria and Lagos State’s energy portal. These sources offered structured insight into hourly consumption patterns, frequency of blackouts, and peak demand behavior under normal grid operations.

However, gaps remained due to the sporadic availability and coarse granularity of Nigerian consumption datasets, particularly in the residential and micro-utility segments. To bridge this gap, the study incorporated **synthetic data generation** techniques that allowed for higher resolution modeling of energy use at the appliance and household level. These synthetic sequences were constructed using statistically informed assumptions derived from academic literature, field interviews, and government blackout schedules. For instance, patterned 12-hour ON/OFF blackout simulations were used to mimic common load-shedding practices in urban neighborhoods. Additionally, appliance-level consumption behavior—such as lighting usage in the evenings or refrigerator cycling loads—was modeled using empirical duty cycles and usage probabilities outlined in Nigerian-specific energy research (e.g., Akara et al., 2025). To reflect the unpredictability and behavioral diversity of real consumers, Gaussian noise and random spikes were also introduced into selected time windows, thereby allowing the forecasting models to learn under conditions of signal distortion and data irregularity.

Each synthetic dataset was transparently documented using parameter files, allowing reproducibility and external scrutiny. Metadata included details such as seed values, blackout frequency, appliance start/stop probabilities, and noise intensity levels. This transparency ensured not only academic rigor but also that the models could be retrained, extended, or adapted by future researchers using the same or modified input logic. In summary, the combination of empirical and synthetic data in this study allowed for both realism and modeling flexibility. It ensured that the forecasting models were trained on a variety of signal types—structured, erratic, and partially missing—much like what would be encountered in a real smart city deployment. By calibrating synthetic data against real-world observations, the study achieved a data preparation strategy that is both technically robust and contextually relevant to urban African energy systems.

**3.4.1 Real-World Data Sources**

To ground the simulation in actual urban energy dynamics, authentic datasets were sourced from two key channels:

* **Open-access repositories**: Publicly available datasets such as the *UCI Smart Grid Stability Dataset* were utilized to train baseline models. This dataset, containing thousands of instances of grid voltage, frequency, and load-level metrics, enabled preliminary model calibration and benchmarking.
* **Localized energy audit reports**: Empirical data from the *Lagos State Electricity Board (LSEB)* energy audit—particularly the 2013 Magodo case study—was integrated into the simulation design. Key features included:
  + Total daily generator fuel consumption (57,648 L)
  + Daily CO₂ emissions (217,442 lbs)
  + Tariff estimates (₦48.05/kWh average)
  + Willingness of end-users to switch to independent power plants (78%)

These metrics were adapted into regional load curves and emission profiles to localize the simulation environment for Nigerian urban contexts.

* **Public grid dashboards and government publications**: Hourly and daily consumption records, grid performance summaries, and energy balance reports were retrieved from federal energy platforms and dashboards (e.g., Transmission Company of Nigeria reports, Lagos Energy Portal). These sources helped model baseline demand cycles and blackout intervals.

All real-world data points were cross-validated where possible to ensure internal consistency and contextual alignment.

**3.4.2 Synthetic Data Generation**

Given gaps in high-resolution, real-time sensor data across Nigerian cities, the study employed synthetic data generation techniques to simulate diverse consumption patterns and system behaviors. These included:

* **Patterned blackout schedules**: Based on documented load-shedding practices in Nigeria, structured blackout sequences (e.g., 12 hours ON / 12 hours OFF) were programmed to simulate grid unreliability and evaluate model robustness under data discontinuities.
* **Appliance-level usage modeling**: Using empirical load signatures from literature (Akara et al., 2025) and structured interviews with urban residents, daily appliance consumption profiles were constructed. Devices such as lighting units, refrigerators, ceiling fans, and water pumps were assigned usage probabilities and hourly duty cycles to mirror real consumption behavior.
* **Random noise and anomaly injection**: Gaussian noise and time-based anomalies were injected into the synthetic dataset to replicate sensor variability, load spikes, and behavioral randomness—thus training the models to account for real-world irregularities.

All synthetic datasets were transparently documented, including code-based seed parameters, probability distributions, and scenario templates, to ensure full reproducibility of experimental conditions. This combination of empirical grounding and synthetic flexibility provided a robust data foundation for modeling, testing, and comparing the performance of the proposed energy forecasting frameworks.

Synthetic consumption sequences were generated at an hourly granularity to match the resolution of most national energy dashboards and to reflect realistic user behavior cycles. Each simulated household profile included a 24-hour load curve, ensuring that the models were exposed to fluctuations across time-of-day and daily cycles. This temporal resolution enabled the forecasting engines to learn finer-grained patterns of appliance usage and blackout impacts, particularly during high-demand periods such as evenings or weekends. By maintaining a consistent time-step across real and synthetic data, the forecasting pipeline was kept structurally coherent and computationally tractable.

**3.5 Forecasting Models Implementation**

To achieve reliable and context-sensitive predictions of short- to medium-term energy consumption in urban settings, this study employs two complementary forecasting models: the Prophet model and the Long Short-Term Memory (LSTM) neural network. Both models are implemented within a Python-based analytics environment and are selected for their proven effectiveness in time series analysis. While Prophet offers simplicity and interpretability for seasonally structured data, LSTM provides the capacity to model complex, non-linear temporal dependencies often found in high-variability consumption patterns. Together, these models form the core of the simulation’s predictive intelligence layer and are subjected to a comparative performance evaluation using identical datasets and runtime conditions.

**3.5.1 Prophet Model**

The Prophet model, developed by Facebook's Core Data Science team, is utilized in this study due to its adaptability to time series datasets that exhibit multiple seasonalities, holidays, and trend shifts. Implemented via the open-source Prophet library in Python, the model is structured as an additive regression framework where time series data is decomposed into trend, seasonality, and holiday components. This study capitalized on Prophet’s capability to automatically detect daily and weekly seasonality patterns within the energy consumption data—particularly useful given the observed cyclical usage of electricity in Nigerian urban households. Additionally, the model’s in-built robustness to missing values and outliers made it well-suited to simulated datasets that include blackout periods or incomplete reporting. To further contextualize the predictions, the Prophet configuration incorporated Nigerian national and religious holidays, such as Independence Day and Eid-al-Fitr, which are known to influence residential energy use patterns. By assigning these holiday dates within the forecasting script, the model was able to account for potential consumption spikes or dips associated with public festivities. The model was trained on 80% of the available data and tested on the remaining 20%, with forecasts generated over a 24-hour horizon. Its performance was benchmarked using MAE, RMSE, and MAPE to assess its predictive accuracy relative to the LSTM model.

**3.5.2 Long Short-Term Memory (LSTM) Neural Network**

The Long Short-Term Memory (LSTM) neural network, a variant of Recurrent Neural Networks (RNNs), was implemented using the TensorFlow and Keras libraries. LSTM is particularly effective for modeling sequential dependencies and temporal correlations, making it an ideal candidate for energy forecasting where past consumption patterns often influence future usage. In the present study, the LSTM architecture consisted of a single LSTM layer with 64 hidden units, followed by a fully connected dense output layer. The model received input in the form of sliding time windows, each containing a 24-hour lookback period of historical energy consumption values to predict the subsequent hour. The network was trained using the Adam optimization algorithm, which offers adaptive learning rate capabilities and efficient gradient computation, thereby accelerating convergence. Root Mean Square Error (RMSE) was employed as the primary loss function during training, aligning model optimization with the evaluation metric used in post-training comparisons. To enhance generalization and prevent overfitting, dropout regularization was applied to the LSTM layer, and early stopping criteria were incorporated to terminate training when validation loss ceased to improve. Training was performed on the same 80/20 split as the Prophet model to ensure comparability. The LSTM's ability to adapt to noise, handle nonlinear trends, and internalize long-term dependencies provided a performance benchmark for assessing the potential of deep learning in context-aware energy demand forecasting within Nigerian urban settings.

**3.6 Model Evaluation Metrics**

To assess the predictive effectiveness of the forecasting models developed in this study, a set of statistical evaluation metrics was employed. These metrics serve as quantitative instruments for analyzing the degree of alignment between the forecasted energy consumption values and their actual counterparts. By measuring different aspects of prediction error—such as absolute deviation, variance sensitivity, and percentage discrepancy—the selected metrics offer a robust framework for model comparison and validation within the simulation environment:

1. **Mean Absolute Error (MAE)**

Among these, the **Mean Absolute Error (MAE)** serves as a primary benchmark. The Mean Absolute Error (MAE) quantifies the average absolute difference between the predicted values () and the actual observations (), without taking into account the direction of the error. It is mathematically expressed as:

A lower MAE indicates better model accuracy. It tells you, on average, how much your predictions deviate from the actual values in absolute terms (e.g., kilowatt-hours). MAE is straightforward and easy to interpret, making it a reliable baseline metric for evaluating model precision.

1. **Root Mean Square Error (RMSE)**

The **Root Mean Square Error (RMSE)** is a widely adopted evaluation metric in regression and forecasting tasks, particularly where precision and responsiveness to large deviations are critical. RMSE measures the square root of the average of the squared differences between the predicted values () and the actual observed values (), and is mathematically expressed as:

RMSE =

Unlike Mean Absolute Error (MAE), which treats all errors equally, RMSE applies a quadratic penalty to larger errors by squaring each residual before averaging. This means that even a few large mispredictions—such as forecasting far too low during a load surge or missing a blackout rebound—can significantly increase the RMSE value. This sensitivity makes RMSE especially useful in energy forecasting scenarios where sudden changes in consumption patterns can occur due to blackouts, fuel shortages, or seasonal events. It provides a more pronounced signal when the forecasting model performs poorly under unstable conditions, thus acting as a robust diagnostic for volatility responsiveness. In essence, a lower RMSE indicates not only general predictive accuracy but also the model’s ability to handle consumption outliers effectively. For this reason, RMSE is often preferred in high-stakes forecasting domains—such as electricity and grid demand—where occasional large errors can lead to significant system inefficiencies or economic costs.

1. **Mean Absolute Percentage Error (MAPE)**

The **Mean Absolute Percentage Error (MAPE)** is a widely used metric that quantifies forecast accuracy by expressing prediction error as a percentage of the actual observed values. It provides an intuitive and scale-independent evaluation of model performance, making it especially useful for comparing forecasting results across datasets with differing magnitudes of energy demand. MAPE calculates the average of the absolute percentage differences between predicted values () and actual values () over all forecast points, using the formula:

MAPE =

In practical terms, MAPE answers the question: “On average, by what percentage did the model’s predictions deviate from the actual values?” For instance, a MAPE score of 10% implies that the predicted energy usage was off by 10 percent, on average, over the course of the forecast horizon. This makes MAPE highly interpretable for both technical and non-technical audiences, as stakeholders can readily understand the proportional error without needing to contextualize units like kilowatt-hours. A key strength of MAPE is its **scale invariance**—it enables meaningful comparisons between forecasting models applied to households with vastly different energy loads, or between urban zones with different usage intensities. However, the metric is not without limitations. It can become inflated or even undefined when the actual values approach zero, as the denominator in its formula becomes negligible or null. This sensitivity is especially pertinent in low-demand periods (e.g., during prolonged outages or overnight load troughs), where even small absolute errors can yield disproportionately large percentage deviations. Despite this caveat, MAPE remains a valuable metric in the context of this study, offering an accessible and standardized way to gauge model reliability across varied urban energy scenarios.

1. **Training Time**

The **Training Time** metric represents the computational duration required for a forecasting model to complete its learning phase on a given dataset. It is typically measured in seconds or minutes, depending on the complexity of the algorithm, the volume of training data, and the hardware specifications of the system executing the task. In this study, training time was recorded for both the Prophet and LSTM models to evaluate the efficiency of each algorithm in terms of resource usage and deployment feasibility. Training time holds particular significance in **resource-constrained environments**, where access to high-performance computing infrastructure may be limited, and energy efficiency is a non-trivial concern. In such settings—common across many sub-Saharan urban areas—models that require less computational power and time are inherently more scalable and sustainable, especially when frequent retraining is necessary to account for evolving consumption behavior, updated sensor streams, or infrastructural disruptions. While accuracy metrics such as MAE, RMSE, and MAPE are central to evaluating predictive fidelity, training time adds a **pragmatic dimension** by revealing the operational cost of deploying each model in real-world applications.

Furthermore, lower training times improve responsiveness in adaptive systems, where on-demand retraining or periodic updates are essential. For instance, in a smart energy dashboard that responds to fluctuating grid conditions or dynamic user loads, fast model retraining enables near real-time forecasting updates. As such, this metric allows researchers and system designers to assess the **trade-off between computational efficiency and predictive power**, informing model selection not only on the basis of accuracy but also on system performance, energy consumption, and turnaround time.

**Table 1.0: Summar Model Evaluation Metrics**

| **Metric** | **Description** |
| --- | --- |
| **MAE (Mean Absolute Error)** | Measures average error magnitude |
| **RMSE (Root Mean Square Error)** | Penalizes large deviations; captures variance |
| **MAPE (Mean Absolute Percentage Error)** | Measures accuracy in percentage terms |
| **Training Time** | For comparing efficiency (especially important in resource-scarce contexts) |

Collectively, these four metrics—MAE, RMSE, MAPE, and Training Time—provide a comprehensive framework for evaluating model performance not only in terms of predictive accuracy, but also in terms of robustness to variability, interpretability across stakeholder groups, and practical deploy ability in resource-constrained computing environments. Their combined use ensures that model selection is informed by both statistical rigor and operational viability.

**3.7 Justification of Methodological Choices**

To ensure methodological rigor and contextual applicability, the study adopts a simulation-based framework that minimizes infrastructural constraints while maximizing replicability and scalability. The integration of dual-model forecasting—leveraging Prophet’s transparency and LSTM’s computational depth—facilitates a nuanced performance comparison across differing algorithmic paradigms. By combining real-world energy data with synthetically generated consumption profiles, the approach balances empirical realism with experimental flexibility, allowing for structured variation in input conditions. Furthermore, the inclusion of contextual scenario modeling reinforces the ecological validity of the system by embedding behavioral dynamics and infrastructural disruptions commonly observed in Nigerian urban energy landscapes. Collectively, these methodological choices position the research to deliver both technically sound results and locally relevant insights.

Table 2.0: Table of Methodological Choices

| **Component** | **Rationale** |
| --- | --- |
| **Simulation-Based Design** | Low-cost, replicable, avoids hardware constraints |
| **Dual-Model Forecasting** | Enables comparative insight: Prophet’s simplicity vs. LSTM’s power |
| **Hybrid Data (Real + Synthetic)** | Balances realism with control |
| **Contextual Scenario Modeling** | Embeds behavioral and infrastructural realism into simulation logic |

**3.8 Limitations of the Methodology**

* **Limited access to high-resolution consumption data** in Nigerian cities required reliance on synthetic models
* **Simulation assumptions** (e.g., sensor dropout rates) may not capture every real-world nuance
* **LSTM training requires high computation**—training time may be prohibitive on low-end systems
* **No physical validation** of system performance due to lack of hardware deployment in this phase

**3.9 Chapter Summary**

This chapter has detailed the methodological framework used to simulate, forecast, and evaluate energy consumption in an urban IoT-based infrastructure setting. By adopting a simulation-driven design, the research circumvents the limitations of hardware deployment while maintaining contextual realism through hybrid data modeling. The integration of both the Prophet model and LSTM neural networks allows for comparative insights into accuracy, interpretability, and computational efficiency. Additionally, the use of real-world audit data from Lagos State—combined with synthetic load profiles—enhances the study’s local relevance and transferability. The next chapter will present the system implementation, analyze simulation results, and evaluate model performance based on the metrics outlined herein.