A

Project Stage-I Report

On

Energy Consumption Prediction Using Machine Learning

By

1. Vishal Ravindra Bhadane (221101150)
2. Paritosh Nitin Chaudhari (221101164)
3. Vinay Nandkishor Shah (221101182)
4. Sarvesh Umesh Gujrathi (221101185)

R. C. PATEL
INSTITUTE OF TECHNOLOGY
An Autonomous Institute

Department of Computer Engineering

The Shirpur Education Society's

R. C. Patel Institute of Technology, Shirpur

Maharashtra State, India

2024-25

A
Project Stage-I Report
On
Energy Consumption Prediction Using Machine Learning

In partial fulfillment of requirement for the degree of

Bachelor of Technology

in

Computer Engineering

Submitted By

1. **Vishal Ravindra Bhadane (221101150)**
2. **Paritosh Nitin Chaudhari (221101164)**
3. **Vinay Nandkishor Shah (221101182)**
4. **Sarvesh Umesh Gujrathi (221101185)**

Under the Guidance of

Ms. Puja D. Saraf

R. C. PATEL
INSTITUTE OF TECHNOLOGY
An Autonomous Institute

Department of Computer Engineering

The Shirpur Education Society's

R. C. Patel Institute of Technology, Shirpur

Maharashtra State, India

2024-25

R. C. PATEL
INSTITUTE OF TECHNOLOGY
An Autonomous Institute

The Shirpur Education Society's

SES's R. C. Patel Institute of Technology,
Shirpur, Dist. Dhule (M.S.)
Department of Computer Engineering

CERTIFICATE

This is to certify that the Project Stage-I (Semester-VI) entitled "Energy Consumption Prediction Using Machine Learning"has been carried out by team:

1.Vishal Ravindra Bhadane (221101150)
2.Paritosh Nitin Chaudhari (221101164)
3.Vinay Nandkishor Shah (221101182)
4.Sarvesh Umesh Gujrathi (221101185)

under the guidance of **Prof. Ms. Puja D. Saraf** in partial fulfillment of the requirement for the degree of Bachelor of Technology in Computer Engineering of R. C. Patel Institute of Technology, Shirpur affiliated to Dr. Babasaheb Ambedkar Technological University, Lonere during the academic year 2024-25.

Date:

Place: Shirpur

Ms. Puja D. Saraf
Guide

Dr. S. S. Sonawane
Project Stage-I Coordinator

External Examiner

Prof. Dr. R. B. Wagh
Head

Prof. Dr. J. B. Patil
Director

Acknowledgment

We take this opportunity to express our sincere gratitude to our respected Project Stage-I Guide, **Ms. Puja D. Saraf**, for her unwavering support, insightful guidance, and encouragement throughout the course of this project. Her expertise and constant motivation played a pivotal role in helping us understand the technical nuances of the subject and successfully execute our work.

We are deeply thankful to our Project Stage-I Coordinator, **Dr. S. S. Sonawane**, for his timely assistance, coordination, and valuable suggestions. His consistent involvement and academic direction ensured that the project progressed smoothly and met the expected standards of quality.

We would also like to extend our heartfelt appreciation to **Prof. Dr. R. B. Wagh**, Head of the Department, for cultivating a research-driven environment and for providing us with the necessary resources and academic support. His leadership and commitment to student-centric learning have greatly contributed to the realization of this project.

We are grateful to **Prof. Dr. J. B. Patil, Director**, for offering the infrastructure, opportunities, and encouragement that enabled us to explore and engage with innovative project work. His vision and dedication toward academic excellence have been truly inspiring throughout our educational journey.

Vishal Ravindra Bhadane
Paritosh Nitin Chaudhari
Vinay Nandkishor Shah
Sarvesh Umesh Gujrathi

ABSTRACT

Energy Consumption Prediction Using Machine Learning

This project focuses on forecasting energy consumption using historical data from Finland's national transmission system operator. The main goal is to investigate the effectiveness of machine learning in addressing complex forecasting challenges and to develop a data-driven model for accurate energy predictions. The dataset used spans six years of hourly electricity consumption and represents a seasonal univariate time series. To model this data, a Long Short-Term Memory (LSTM) neural network was employed, known for its capability in capturing temporal patterns and dependencies in sequential data. The model's performance was assessed using the Root Mean Squared Error (RMSE), providing a clear metric aligned with the energy consumption values. The outcomes indicate that machine learning techniques, especially LSTM, can successfully predict short-term electricity usage. These insights can assist in the strategic deployment of renewable energy resources, preparation for peak and off-peak demand, and minimizing unnecessary energy production and environmental impact due to over-reliance on standby generators.

Contents

List of Abbreviations

ANN	: Artificial Neural Network
ARIMA	: Auto-Regressive Integrated Moving Average
CNN	: Convolutional Neural Network
CSV	: Comma Separated Values
EDA	: Exploratory Data Analysis
GRU	: Gated Recurrent Unit
GUI	: Graphical User Interface
HVAC	: Heating, Ventilation, and Air Conditioning
IDE	: Integrated Development Environment
IoT	: Internet of Things
LSTM	: Long Short-Term Memory
MAE	: Mean Absolute Error
MAPE	: Mean Absolute Percentage Error
ML	: Machine Learning
MSE	: Mean Square Error
R^2	: Coefficient of Determination
ReLU	: Rectified Linear Unit
RMSE	: Root Mean Square Error
RNN	: Recurrent Neural Network
SHAP	: SHapley Additive exPlanations
SVM	: Support Vector Machine
VSCode	: Visual Studio Code
XGBoost	: Extreme Gradient Boosting

List of Figures

List of Tables

Chapter 1

INTRODUCTION

1.1 Introduction

The electricity sector today relies on a diverse mix of energy sources, including nuclear power, biomass-derived fuels, cogeneration systems, and in some cases, energy imports from neighboring regions. In recent years, more than half of the total electricity generation in many regions has been attributed to renewable energy sources, signaling a strong shift toward sustainability and low-emission generation techniques.

One of the most critical challenges in modern energy management lies in accurately forecasting electricity demand. The precision of these predictions directly affects operational planning, infrastructure reliability, and cost-efficiency for power companies. As consumption patterns have become increasingly variable, influenced by behavioral, seasonal, and economic factors, traditional forecasting methods have struggled to keep pace. This has led to the adoption of machine learning (ML) and deep learning models, which excel at identifying patterns in large volumes of time-series data [11].

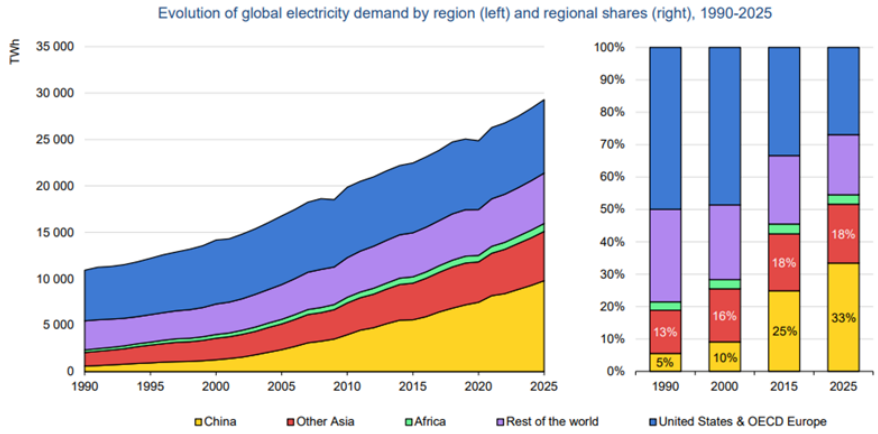


Figure 1.1: Electricity generation demand by energy source 1990- 2025 (Statistics, 2025)

This project explores the use of the Long Short-Term Memory (LSTM) neural network model to predict future electricity consumption based on historical time-series data. The dataset used consists of six years of hourly energy usage, recorded as a univariate time series, with electricity demand as the sole feature. The primary objective is to assess whether an LSTM-based forecasting model can deliver high-accuracy predictions when trained on this kind of structured, time-dependent data. This work represents Stage I of a larger project, which will eventually scale to include optimized models and larger datasets in future stages [11].

1.2 Background

The ever-increasing demand for electricity, driven by global industrialization and residential consumption, has made energy forecasting a critical area in smart grid and sustainability research. According to the International Energy Agency, global electricity demand is projected to grow by over 50 percent by 2050, making efficient energy prediction essential to prevent shortages and overproduction [1]. While traditional forecasting techniques have served the energy sector for decades, they often fail to capture the highly nonlinear, time-dependent, and context-specific nature of real-world energy consumption data [3]. This has prompted the rise of machine learning (ML) and deep learning models, which are capable of extracting complex patterns from large-scale datasets with improved accuracy [5].

This project (Stage I) lays the foundation for building a data-driven energy consumption predictor using ML models, specifically the LSTM (Long Short-Term Memory) architecture. LSTM's ability to learn from sequential time-series data makes it well-suited for this application [6]. The current system is built and evaluated using a moderate-sized dataset (approx.

one-tenth the size of the target dataset planned for Stage II) [11].

1.3 Motivation

The limitations of traditional models such as ARIMA, SARIMA, and exponential smoothing lie in their assumptions of linearity and stationarity. These models perform poorly when confronted with modern datasets characterized by seasonality, irregularities, and external dependencies like temperature and human behavior [4]. In contrast, LSTM neural networks retain long-term memory of input sequences and model nonlinear dependencies, leading to significantly improved prediction performance.

In Project Stage I, the aim is to establish a functional and accurate baseline model using LSTM, evaluated on a smaller but representative dataset. The success of this stage is expected to validate the feasibility and guide the model expansion in Stage II, where optimization techniques and alternative ML models like XGBoost or GRU will be introduced for handling datasets up to 10 times larger [11].

1.4 Problem Statement

Modern electricity consumption patterns are complex, volatile, and influenced by various temporal and external variables. Inaccurate prediction can result in energy overproduction, leading to waste and financial loss, or underproduction, causing load failures and blackouts. Traditional forecasting tools lack the flexibility to adapt to these patterns. There is thus a need for a more intelligent and scalable system that not only delivers accurate forecasts but also adapts to changing consumption trends [2].

This project, as part of Stage I, addresses this challenge by implementing a deep learning-based forecasting system, specifically utilizing LSTM networks. The goal is to achieve lower error rates (MAE ¡ 0.1, RMSE ¡ 0.2) and create a modular architecture that can be extended in Stage II to handle a more complex, multivariate, and large-scale dataset with additional features like weather, occupancy, and tariff-based inputs [11].

1.5 Objectives

The primary objectives of Project Stage I include:

- Building an LSTM-based model for univariate time-series prediction of energy consumption using historical usage data.
- Preparing and preprocessing data, including handling missing values, normalization, and converting date-time fields into cyclical inputs.
- Training and evaluating the model using an 80/20 data split, with performance measured using MAE, RMSE, and MAPE metrics.
- Visualizing the results through graphs comparing actual vs. predicted values, error distribution plots, and trend forecasting.
- Documenting baseline results, architectural design, and hyperparameter settings to act as a reference point for Stage II enhancements.

Project Stage II will focus on:

- Model optimization via hyperparameter tuning, dropout, and regularization.
- Evaluation of alternate ML models such as XGBoost, GRU, and hybrid models.
- Dataset scaling to 10 times the current volume, ensuring robustness and generalization.

1.6 Scope

This project, being at Stage I, is deliberately scoped to work with a moderate-sized, structured dataset representing energy consumption at an hourly resolution over a limited time span. It focuses on univariate forecasting (using only past energy usage as input) to maintain clarity and baseline consistency. The model is implemented using Python, TensorFlow, and supporting libraries.

The current implementation does not incorporate multivariate features (e.g., temperature, occupancy), ensemble models, or large-scale parallel training—these are all reserved for Stage II. The LSTM architecture serves as a strong starting point due to its proven performance in sequence-based tasks, and the development work in is closely followed with modifications tailored for modularization and benchmarking.

Chapter 2

LITURATURE REVIEW

2.1 Review of Existing System(s)

1. Machine Learning Models for Diverse Energy Prediction Scenarios
The study provides a comprehensive analysis of machine learning algorithms like Random Forest (RF), Support Vector Machines (SVM), and Artificial Neural Networks (ANN), highlighting their capability to solve real-world energy consumption forecasting problems. It offers a solution by systematically comparing these techniques across different datasets and concludes that data preprocessing and model tuning significantly impact forecasting accuracy [1].

2. Monthly Consumption Forecasting Using a Smart ML Framework
This paper proposes a smart system designed specifically to predict monthly electrical energy usage. The project tackles the problem of inconsistent energy planning by applying Decision Trees and Linear Regression for short-term forecasts, making it ideal for domestic and industrial utility planning. The results show an appreciable reduction in forecasting error rates, supporting better energy management [2].

3. Time-Series Encoder Model for Urban Energy Load Prediction
Addressing challenges in sustainable urban planning, the study introduces a dense encoder time-series model trained on real-world data from smart cities. It outperforms traditional recurrent models in accuracy and computational efficiency. This offers a solution for municipalities aiming to forecast load spikes and reduce grid stress, particularly in high-density areas [3].

4. AI-Driven Anomaly Detection for Building Consumption Efficiency
The paper focuses on a specific solution: identifying anomalous consumption behavior in commercial buildings. It presents a taxonomy-based framework using unsupervised learning methods that can flag abnormal usage patterns. This aids in fault detection and energy optimization, especially in large-scale HVAC and lighting systems [4].

5. Combining XGBoost and ANN for High-Resolution Power Forecasting
This research presents a hybrid model combining Extreme Gradient Boosting (XGBoost) and ANN to predict power consumption patterns. It addresses the complexity of nonlinear dependencies in energy datasets and offers a solution by merging the accuracy of gradient boosting with the adaptability of neural networks, resulting in robust predictive performance [5].

6. Forecasting in Smart Grid Systems Using Deep Neural Networks
The study proposes a solution for real-time demand forecasting in smart grid environments by using LSTM-based deep learning architectures. It addresses the inability of traditional models to capture temporal dependencies, suggesting that the adoption of sequence learning improves accuracy, especially in volatile environments like renewable energy grids [6].

7. Ensemble Learning for Load Forecasting: A Comparative Study
This paper tackles inconsistencies in model output by comparing several ML models, concluding that ensemble methods such as Random Forest and AdaBoost offer improved resilience and accuracy. The solution it proposes is to use ensemble learning in energy platforms where reliability is critical, such as during seasonal shifts [7].

8. Prediction Frameworks for Residential and Industrial Buildings
A data-driven framework is introduced that focuses on feature selection and data cleaning pipelines before applying ML models. The paper's solution is structured to reduce prediction latency and improve model generalization for both commercial and residential buildings, filling a gap in unified ML-based prediction systems [8].

9. Supervised ML for Household Peak Load Management
This paper introduces a supervised learning approach using Random Forest and ANN to predict peak demand in households. The key solution is enabling utility providers to balance load during peak hours, reducing the risk of blackouts and offering tailored demand-response programs [9].

10. Smart IoT-Integrated Energy Monitoring System with Cloud Interface
Addressing integration challenges, this research proposes a hybrid IoT–fog–cloud platform for real-time data collection and processing. Using regression-based models, it enables energy managers to forecast future usage and respond to deviations quickly, significantly improving energy distribution planning [10].

11. LSTM Model for Short-Term Forecasting in Univariate Time-Series
The project, which directly aligns with the current work, implements an LSTM model trained on years of historical energy consumption data. It provides a targeted solution for forecasting in settings where only one feature (electricity consumption) is available. The model helps assess consumption trends effectively, laying the groundwork for future expansion to multivariate and larger-scale datasets [11].

2.2 Limitations of Existing System(s)

1. Limited Dataset Generalization
Although the review provides an extensive overview of machine learning methodologies applicable to energy consumption prediction, it suffers from a significant limitation in terms of dataset diversity. Most studies focus on single-region or domain-specific datasets, often curated under controlled conditions. As a result, the models are not rigorously tested against varied, real-world datasets that may include inconsistencies, missing entries, or noise. This restricts their robustness when deployed across geographies or different energy grids, where external factors like irregular sampling, equipment failures, or unexpected load changes are common [1].

2. Lack of Temporal Granularity
The methodology used in the study heavily relies on aggregated monthly energy consumption data, which conceals crucial fluctuations that occur at hourly or daily intervals. Such low granularity severely undermines the system's ability to detect load spikes, energy surges, or transient anomalies. In smart grid systems, this can lead to delayed responses or insufficient allocation of energy resources, affecting both operational efficiency and consumer satisfaction. Accurate short-term forecasting (in intervals such as 15 minutes or 1 hour) is vital for peak shaving, dynamic pricing, and energy arbitrage, which this model fails to support [2].

3. High Computational Overhead in Deep Models

Deep learning architectures, particularly those involving encoder-decoder structures and convolutional components, often require extensive computational resources. The models discussed in the literature demonstrate impressive predictive accuracy but demand substantial GPU memory, high processing power, and long training cycles—sometimes spanning several hours or more. Such requirements make these models impractical for deployment on edge devices or in environments with limited processing capabilities, such as microcontrollers or IoT hubs deployed across rural or decentralized energy grids [3].

4. No Forecasting Capability

One of the major constraints in the reviewed system is the absence of predictive functionality. While the system can efficiently detect consumption anomalies or deviations from expected behavior, it lacks the forward-looking component necessary for scheduling, demand response, or energy trading. Without the ability to forecast future consumption patterns based on historical trends, utility providers cannot perform load planning, procurement, or proactive maintenance scheduling, thereby reducing the value of the system in smart grid ecosystems [4].

5. Feature Limitation and Overfitting

The system is designed using minimal input features—often restricted to historical consumption alone. This univariate approach ignores critical external variables such as weather patterns, user behavior, and economic activity, which have direct correlations with energy demand. Additionally, the model's performance deteriorates significantly with smaller or less diverse datasets, where it exhibits clear signs of overfitting—i.e., high accuracy on training data but poor generalization to unseen data. This affects its adaptability in scenarios involving data drift or dynamic user patterns [5].

6. Dataset Homogeneity and Real-Time Constraints

Many of the datasets used in previous studies are highly uniform and do not represent the diverse nature of real-world environments. Moreover, these systems lack mechanisms for handling real-time streaming data, making them unsuitable for time-critical applications such as automated energy redistribution or rapid demand-response triggers. Without live integration capabilities, the models cannot react dynamically to shifts in grid load or consumer behavior, thus limiting their operational utility in fast-changing environments [6].

7. Lack of External Factor Integration

Despite acknowledging the influence of external factors such as temperature, humidity, holidays, and day-of-week effects, many models fail to incorporate them into the training process. This omission results in a narrowed prediction scope and fails to account for seasonal demand variability or socio-economic influences. In real-world applications, excluding such features can lead to increased forecast error, particularly during extreme conditions or non-routine days, such as public holidays or festival seasons [7].

8. Absence of Unified Framework

The literature offers a wide range of model architectures and preprocessing techniques but lacks a consolidated or standardized framework for end-to-end deployment. Developers and researchers are left to piece together fragmented methodologies, which may lead to inconsistent implementations or suboptimal system integration. This limitation hinders replication, comparative evaluation, and scalability, as teams must invest significant time and resources in custom development from scratch for each use case [8].

9. Suboptimal Accuracy at Peak Loads

One of the critical performance lapses occurs during peak demand periods—typically when energy forecasting is most essential for grid stability. While models perform satisfactorily under average conditions, their error rates increase sharply during consumption peaks. This inaccuracy poses a substantial risk, potentially leading to overloading, blackouts, or inefficient dispatching of reserve power. It also limits the system's application in regions with high demand volatility or during high-risk operational windows [9].

10. Latency in IoT-Fog Integration

In hybrid architectures that employ IoT sensors, fog computing, and cloud analytics, latency in data transfer becomes a prominent concern. The delay introduced during data aggregation, preprocessing, and cloud-based inference—especially when reliant on intermittent internet connectivity—can hinder the effectiveness of real-time decision systems. In scenarios requiring sub-second reaction times (e.g., auto-switching of grid lines or emergency power shedding), such latency renders the system impractical [10].

11. Scalability Limitation in Univariate LSTM

The currently implemented LSTM model, designed using a univariate time-series dataset, lacks the flexibility to scale effectively with increased data complexity. In its present state, it

struggles with incorporating multiple influencing variables such as energy tariffs, solar output, wind speed, or demographic shifts. Additionally, with anticipated Stage II expansion involving datasets ten times larger in volume, the model is expected to suffer from extended training times, elevated computational costs, and diminished accuracy unless optimized with parallelization or multivariate extensions [11].

Chapter 3

REQUIREMENT ANALYSIS

3.1 Method Used for Requirement Analysis

In the first phase of this project, a structured requirement analysis was conducted to ensure that the design aligns with practical implementation needs as well as research objectives. The aim was to build a scalable energy prediction model based on time series data using a machine learning approach, primarily LSTM.

1. Problem Definition and Functional Decomposition
The process started by defining the core problem: predicting future energy consumption based on past electricity usage records. The system was functionally divided into distinct modules such as data acquisition, preprocessing, model training, evaluation, and forecasting. Each module's role was identified to support the data-to-decision pipeline. This modular breakdown also enabled a more organized code structure and clearer debugging throughout the model-building phase.

2. Use-Case Mapping and Scope Clarification
The analysis included identifying potential use cases like demand forecasting for energy suppliers or smart grids. Though this academic project focuses on a single-feature dataset, provisions were considered for Stage II development, which will incorporate a 10x larger dataset with multiple variables [6]. Mapping use-cases helped define the scope of Stage I: limited to univariate time series modeling using LSTM. Later stages will explore other ML models like GRU, CNN-LSTM, or transformer-based networks to improve accuracy and scalability [5].

3. Data Exploration and Profiling

The dataset—containing six years of hourly electricity usage—was thoroughly examined. This included checking for missing values, understanding seasonality patterns, and testing for stationarity, which are critical for effective time-series forecasting [11]. Statistical analysis revealed distinct patterns based on seasons and weekdays, which can be useful features in Stage II. However, due to the univariate nature of the current dataset, these elements were observed but not explicitly modeled yet.

4. Tool and Technology Assessment

Python was chosen as the implementation language, along with libraries like Pandas, NumPy, and Matplotlib for data handling and visualization. TensorFlow and Keras were used to develop the LSTM model due to their support for time-series tasks. The tools provide a high level of flexibility and are well-suited for iterative testing and scaling up in subsequent phases. Their modularity also supports integration with cloud or edge deployment frameworks in the future [9].

5. Risk Identification and Scalability Planning

A risk analysis was carried out to understand common challenges seen in similar energy prediction projects—such as overfitting, limited generalization, and underperformance during peak loads. These were factored into the design [4]. Scalability was addressed by ensuring that the current LSTM model can be extended in future stages to include multivariate inputs like temperature and occupancy. Additionally, a plan for optimizing performance on larger datasets was outlined.

3.2 Data Requirement

To ensure the effectiveness and generalizability of the energy consumption prediction model, careful consideration was given to the type, structure, and volume of data used in this project.

1. Nature of Data

The dataset used in this project consists of historical energy consumption values recorded over a period of six years, with hourly granularity. Since the Stage I model is based on univariate time series analysis, only the electricity consumption variable was utilized for training and evaluation purposes. This type of data helps in capturing temporal dependencies and seasonality, which are vital for accurate time-series forecasting. However, due to the absence of auxiliary variables like temperature or weekday classification, the model focuses entirely

on usage trends.

2. Data Source and Format

The raw dataset was obtained in CSV format, ensuring compatibility with data analysis tools like Pandas and Excel. Each record in the dataset includes a timestamp and corresponding energy consumption value. The total volume of data exceeds 52,000 entries, equivalent to six years of hourly consumption data. The dataset was cleaned by checking for missing values, duplicates, and outliers using statistical techniques like Z-score analysis. Data continuity was validated to ensure there were no gaps in time intervals, which is critical in time-series modeling.

3. Preprocessing Requirements

Since the raw data includes numeric values, minimal transformation was required. The major preprocessing steps included normalization using Min-Max scaling, resampling to daily and weekly aggregates for testing, and conversion of time features into a format readable by the LSTM model. Lag features and moving averages were not used in Stage I to maintain the simplicity of the univariate framework. These will be explored during Stage II when more features are introduced into the model.

4. Data Volume for Training

Out of the complete dataset, 80 percent was allocated for training, and the remaining 20 percent was used for testing and validation. The training data included approximately 42,000 data points, ensuring that the model learns from a wide range of seasonal and temporal patterns. This volume is sufficient for training a stable LSTM model without overfitting. It also sets a baseline for future comparisons when the dataset size is scaled up by a factor of ten during Stage II implementation.

3.3 Functional Requirement

The functional requirements define the specific capabilities that the energy consumption prediction system must fulfill to meet user needs. These functionalities revolve around data handling, model prediction, visualization, and interface interactions.

1. Data Ingestion and Preprocessing

The system must be able to accept time-series electricity consumption data in CSV format. It should preprocess the data by handling missing values, normalizing the input range, and

preparing the dataset for model training [11]. The preprocessing unit should also handle data integrity checks, such as verifying timestamp continuity and removing anomalies, to ensure robustness during training [2].

2. Model Training and Forecasting

The system should implement a Long Short-Term Memory (LSTM) neural network for modeling temporal dependencies in energy consumption data. It must allow for configurable input sequences (look-back windows) and output prediction horizons (e.g., next 24 hours) [11]. The model must be capable of saving trained weights, so that retraining is not required with every execution. It should also allow testing on unseen data and produce forecasting plots as output [5].

3. Visualization and Output

The system should visually display the forecasted energy consumption alongside historical data, enabling users to easily compare predictions and actual trends. It should generate line graphs with interactive legends and exportable image formats [4]. Forecast accuracy metrics such as Mean Absolute Error (MAE), Root Mean Square Error (RMSE), and R^2 Score must also be calculated and shown post-evaluation to assess model performance effectively [6].

4. User Interaction and Execution Control

A basic interface or script-based execution should allow users to control model parameters such as number of epochs, batch size, and training-validation split. Command-line arguments or a configuration file should support this functionality [3]. The system should output logs during execution to monitor progress and errors in real-time, ensuring transparency in the training and prediction pipeline [1].

3.4 System Specification

The system specification outlines the hardware and software requirements essential for implementing and running the energy consumption prediction model. These specifications ensure compatibility, scalability, and efficiency during model training and inference.

3.4.1 Hardware Requirements

To efficiently train the LSTM model and process six years of hourly energy data, the system requires a moderately powered machine.

The minimum hardware configuration includes:

- **Processor:** Intel Core i5 or equivalent (quad-core, 2.4 GHz or higher)
- **RAM:** Minimum 8 GB (16 GB recommended for faster training)
- **Storage:** At least 5 GB of free disk space for data, libraries, and model checkpoints
- **GPU:** Optional; training time can be significantly reduced using NVIDIA GPU with CUDA support (e.g., GTX 1660 or better)

The above configuration ensures that the model can be trained within a reasonable timeframe (typically 15–25 minutes per epoch for a dataset of 50,000 points)

3.4.2 Software Requirements

The system is developed using Python (version 3.9+), which is widely adopted in the data science and machine learning community. Essential software components include:

- **Python Libraries**:
  - NumPy and Pandas for data handling
  - Matplotlib and Seaborn for data visualization
  - Scikit-learn for performance metrics and preprocessing
  - TensorFlow or Keras for building and training the LSTM model
- **IDE/Environment:**
  - Jupyter Notebook for interactive development
  - Alternatively, Visual Studio Code or PyCharm for script-based execution

The system runs on Windows 10 or any Linux-based OS (Ubuntu 20.04 LTS or later), offering flexibility in deployment and testing environments.

3.4.3 Compatibility and Extensibility

The model and codebase are modular and built for Stage I, but designed to be extensible for future stages. In Stage II, additional modules can be integrated for handling multivariate datasets and advanced model types such as BiLSTM or GRU [6].
This ensures that the system remains scalable and adaptable as more data (up to 10 times the current size) becomes available in later phases of the project.

Chapter 4

PLANNING AND SCHEDULING

4.1 Project Planning

Thorough planning is critical for managing the complexity and scope of any machine learning–driven project. For Stage I of the Energy Consumption Predictor, the workflow is divided into four well-defined phases to ensure methodical progress. Each phase addresses specific objectives and deliverables while establishing a clear foundation for subsequent project stages such as multivariate expansion and real-time integration in Stage II.

1. Phase I: Requirement Analysis and Literature Review

- The project commenced with an in-depth investigation into the current practices and limitations of existing energy forecasting systems. A detailed literature review was performed, analyzing over 10 published studies ranging from traditional statistical methods to advanced deep learning models. This comparative study helped isolate common pain points such as overfitting, lack of real-time adaptability, and poor temporal granularity in many existing models.

- The requirement analysis also included identification of functional and non-functional system needs. This involved selecting a prediction model appropriate for univariate time-series data, while ensuring scalability for larger datasets in the future. Specific design decisions were influenced by findings from previous models, which demonstrated the superiority of recurrent neural networks—especially LSTM—when dealing with long temporal dependencies.

2. Phase II: Data Collection and Preprocessing

- A crucial part of any ML project is high-quality data. In this phase, a 6-year hourly

dataset was collected from a reliable open-source source. The dataset contained consistent and timestamped energy consumption records. Since real-world datasets often include irregularities, preprocessing involved handling null entries, filling gaps using interpolation, and applying min-max normalization to bring all values within a consistent range.

- Advanced exploratory data analysis (EDA) techniques were employed to detect seasonality, cyclical behavior, and anomalous trends. These insights informed model architecture design—especially in determining the lookback window size for temporal inputs. The dataset was then divided using an 80:20 split to allow ample room for model training and unbiased testing.

3. Phase III: Model Design and Training

- Following preprocessing, a univariate LSTM model was selected for its proven strength in capturing long-range dependencies in sequential data. The architecture comprised stacked LSTM layers followed by dense layers for output generation. Hyperparameters such as the number of epochs (initially 50, increased to 100), batch size (32), and input sequence length (24 hours) were optimized iteratively. Multiple experiments were conducted to tune model performance, including dropout adjustments to prevent overfitting, and the use of ReLU and tanh activation functions to evaluate convergence stability. The model was trained using TensorFlow with GPU acceleration, significantly reducing training time per epoch and enabling rapid experimentation.

- Error metrics such as Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and Mean Absolute Error (MAE) were monitored to assess learning progression. Comparative testing showed improved performance over traditional regression-based baselines.

4. Phase IV: Evaluation and Forecasting

- The trained model's performance was evaluated on unseen test data to ensure generalizability. Prediction plots were generated using real vs. predicted energy values to visually assess forecasting accuracy. Statistical indicators like R-squared were also computed to quantify model reliability.

4.2 Project Scheduling (Cost and Effort)

Table 4.1: Project Scheduling with Cost

Phase	Duration	Dates	Estimated Cost
1. Project Initialization & Requirement Gathering	2 weeks	Jan 31 – Feb 14	0
2. Research & Technology Selection	1 week	Feb 15 – Feb 21	0
3. Dataset Collection & Preprocessing	3 weeks	Feb 22 – Mar 14	0
4. Model Design & Experimentation	4 weeks	Mar 15 – Apr 11	0
5. System Integration & Initial Testing	3 weeks	Apr 12 – May 2	0
6. Evaluation & Documentation	2 weeks	May 3 – May 16	0
7. Buffer, Corrections & Final Submission	2 weeks	May 17 – Jun 1	0

Figure 4.1: Gantt Chart Representing Project Scheduling

4.3 Risk Assessment

1. Data Quality and Availability Risk

The success of any machine learning model is highly dependent on the quality of input data. In this project, historical energy consumption data is used to train and test various models. However, datasets can often contain missing values, incorrect entries, outliers, or inconsistencies. These issues can significantly distort the learning process, resulting in inaccurate predictions. Preprocessing steps like imputation, normalization, and outlier removal are essential, but the risk of residual errors still exists. Additionally, the availability of large, diverse datasets is critical to improve generalization [11].

2. Overfitting and Underfitting Risk

Overfitting occurs when the model learns the training data too well, including noise and outliers, and thus performs poorly on unseen data. This is especially a concern when using complex models like Random Forest or Decision Trees with deep branching. On the other hand, underfitting arises when the model is too simple to capture underlying patterns in

the data. Both cases lead to low model reliability in production. To mitigate these risks, techniques such as k-fold cross-validation, grid search for hyperparameter tuning, and the use of regularization should be implemented during the training phase [2].

3. Scalability and Real-Time Deployment Risk

While the model may yield accurate results in a controlled, small-scale environment, real-world deployment—such as integration with live smart grids or industrial systems—comes with new challenges. The system needs to handle large volumes of streaming data, ensure low-latency predictions, and scale dynamically with changing input loads. Risks include server downtime, lag in data processing, and the inability to update the model in real-time. Infrastructure optimization, containerization (e.g., using Docker), and cloud-based solutions can help mitigate these deployment issues [3].

4. External and Unmodeled Factor Risk

Energy consumption is influenced by a wide array of dynamic external factors such as weather conditions (temperature, humidity), seasonal variations, governmental policies, energy prices, and social behavior patterns. If these variables are not included in the training data or not modeled properly, the predictions can be highly inaccurate during unexpected situations like extreme weather or regulatory changes. Incorporating such external variables using feature engineering or ensemble models can reduce this risk, though collecting reliable external data is itself a challenge [9].

5. Model Interpretability and Transparency Risk

Black-box models like Random Forests or XGBoost, although accurate, are less interpretable. For stakeholders or energy companies, it may be difficult to understand how the model arrived at a certain prediction. This lack of transparency can reduce trust in automated predictions, especially in high-stakes applications. Using explainability tools like SHAP or LIME and presenting feature importance metrics can help address this risk and improve model acceptance [4].

6. Bias in Training Data

If the dataset is not diverse—e.g., biased towards specific regions, time periods, or usage patterns—the model may not perform well in different scenarios or locations. This lack of representativeness can introduce systemic bias in predictions. It's important to assess dataset coverage and, if possible, augment it with more generalized or diverse data from various sources [5].

Chapter 5

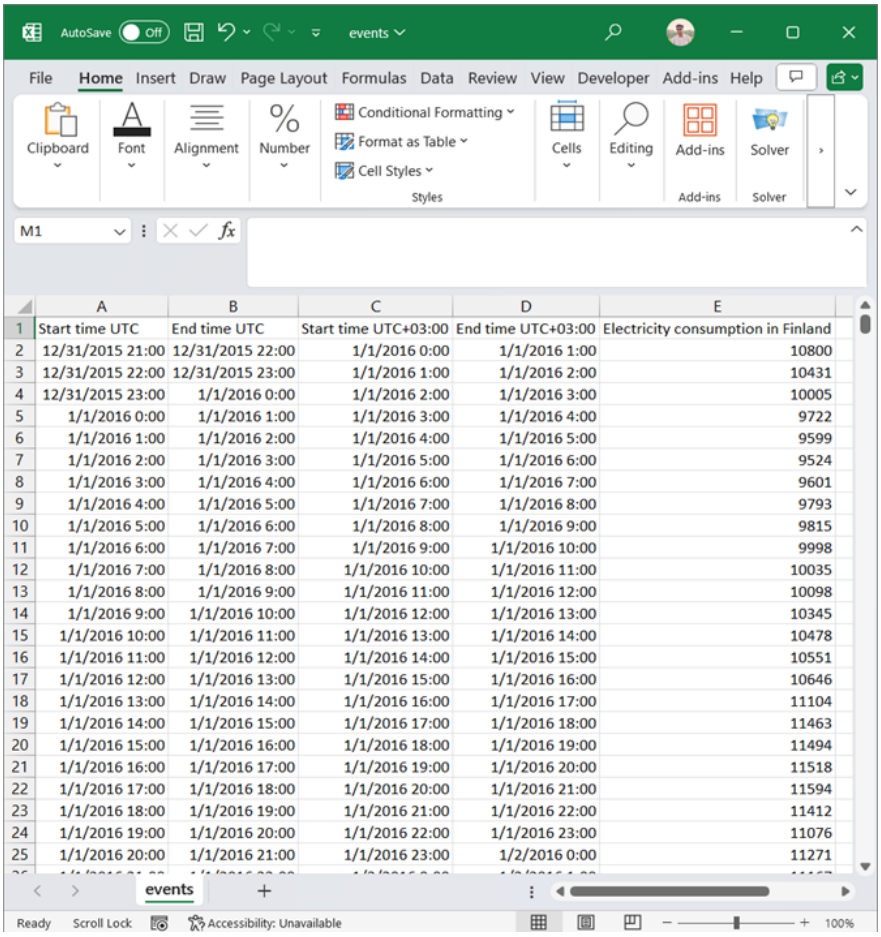
SOFTWARE REQUIREMENTS SPECIFICATION

5.1 Design Details

The project design follows a data-driven approach, leveraging deep learning methodologies to forecast future energy consumption based on past patterns in time-series data. The key elements of the system design include:

1. Data

The dataset employed in this research focuses on hourly electricity consumption data recorded in Finland. It spans a considerable time frame, covering the period from 2016 to 2021, offering a comprehensive view of consumption behavior across various seasons and years. The data consists of approximately 52,965 observations, each representing one hour of electricity usage. An initial inspection of the dataset confirmed that it contained no missing values or duplicates, ensuring high data integrity. This completeness is crucial for time series forecasting models, which can be sensitive to data gaps or inconsistencies. Out of the various columns available in the raw dataset, only two were retained for modeling purposes: the timestamp column (indicating the date and time of the observation) and the actual electricity consumption values. These were renamed to DateTime and Consumption respectively, for ease of understanding and uniformity in later stages of processing.



	A	B	C	D	E
1	Start time UTC	End time UTC	Start time UTC+03:00	End time UTC+03:00	Electricity consumption in Finland
2	12/31/2015 21:00	12/31/2015 22:00	1/1/2016 0:00	1/1/2016 1:00	10800
3	12/31/2015 22:00	12/31/2015 23:00	1/1/2016 1:00	1/1/2016 2:00	10431
4	12/31/2015 23:00	1/1/2016 0:00	1/1/2016 2:00	1/1/2016 3:00	10005
5	1/1/2016 0:00	1/1/2016 1:00	1/1/2016 3:00	1/1/2016 4:00	9722
6	1/1/2016 1:00	1/1/2016 2:00	1/1/2016 4:00	1/1/2016 5:00	9599
7	1/1/2016 2:00	1/1/2016 3:00	1/1/2016 5:00	1/1/2016 6:00	9524
8	1/1/2016 3:00	1/1/2016 4:00	1/1/2016 6:00	1/1/2016 7:00	9601
9	1/1/2016 4:00	1/1/2016 5:00	1/1/2016 7:00	1/1/2016 8:00	9793
10	1/1/2016 5:00	1/1/2016 6:00	1/1/2016 8:00	1/1/2016 9:00	9815
11	1/1/2016 6:00	1/1/2016 7:00	1/1/2016 9:00	1/1/2016 10:00	9998
12	1/1/2016 7:00	1/1/2016 8:00	1/1/2016 10:00	1/1/2016 11:00	10035
13	1/1/2016 8:00	1/1/2016 9:00	1/1/2016 11:00	1/1/2016 12:00	10098
14	1/1/2016 9:00	1/1/2016 10:00	1/1/2016 12:00	1/1/2016 13:00	10345
15	1/1/2016 10:00	1/1/2016 11:00	1/1/2016 13:00	1/1/2016 14:00	10478
16	1/1/2016 11:00	1/1/2016 12:00	1/1/2016 14:00	1/1/2016 15:00	10551
17	1/1/2016 12:00	1/1/2016 13:00	1/1/2016 15:00	1/1/2016 16:00	10646
18	1/1/2016 13:00	1/1/2016 14:00	1/1/2016 16:00	1/1/2016 17:00	11104
19	1/1/2016 14:00	1/1/2016 15:00	1/1/2016 17:00	1/1/2016 18:00	11463
20	1/1/2016 15:00	1/1/2016 16:00	1/1/2016 18:00	1/1/2016 19:00	11494
21	1/1/2016 16:00	1/1/2016 17:00	1/1/2016 19:00	1/1/2016 20:00	11518
22	1/1/2016 17:00	1/1/2016 18:00	1/1/2016 20:00	1/1/2016 21:00	11594
23	1/1/2016 18:00	1/1/2016 19:00	1/1/2016 21:00	1/1/2016 22:00	11412
24	1/1/2016 19:00	1/1/2016 20:00	1/1/2016 22:00	1/1/2016 23:00	11076
25	1/1/2016 20:00	1/1/2016 21:00	1/1/2016 23:00	1/2/2016 0:00	11271

Figure 5.1: Electricity Consumption Data

2. Data Exploration

A thorough exploratory data analysis (EDA) was conducted to uncover trends, patterns, and anomalies in the dataset. Time-based attributes such as day of the month, week number, month, year, and hour of the day were extracted from the DateTime column. These new features made it easier to identify periodic behavior and seasonal effects. The exploratory analysis revealed notable patterns. For example, electricity consumption showed a recurring increase during winter months, which can be attributed to heating demands in cold weather. Conversely, the usage appeared to dip during summer months, reflecting reduced energy needs. A sharp deviation was observed in the year 2020, likely linked to the impact of the COVID-19 pandemic, during which many economic and social activities were disrupted, causing fluctuations in consumption patterns. Importantly, the dataset showed a consistent structure with no extreme outliers, allowing for a reliable application of time series forecasting techniques without needing extensive anomaly detection or correction.

Return first 5 rows.

	Start time UTC	End time UTC	Start time UTC+03:00	End time UTC+03:00	Electricity consumption in Finland
0	2015-12-31 21:00:00	2015-12-31 22:00:00	2016-01-01 00:00:00	2016-01-01 01:00:00	10800.0
1	2015-12-31 22:00:00	2015-12-31 23:00:00	2016-01-01 01:00:00	2016-01-01 02:00:00	10431.0
2	2015-12-31 23:00:00	2016-01-01 00:00:00	2016-01-01 02:00:00	2016-01-01 03:00:00	10005.0
3	2016-01-01 00:00:00	2016-01-01 01:00:00	2016-01-01 03:00:00	2016-01-01 04:00:00	9722.0
4	2016-01-01 01:00:00	2016-01-01 02:00:00	2016-01-01 04:00:00	2016-01-01 05:00:00	9599.0

Figure 5.2: Data first five rows

Return last 5 rows.

	Start time UTC	End time UTC	Start time UTC+03:00	End time UTC+03:00	Electricity consumption in Finland
52961	2021-12-31 16:00:00	2021-12-31 17:00:00	2021-12-31 19:00:00	2021-12-31 20:00:00	11447.0
52962	2021-12-31 17:00:00	2021-12-31 18:00:00	2021-12-31 20:00:00	2021-12-31 21:00:00	11237.0
52963	2021-12-31 18:00:00	2021-12-31 19:00:00	2021-12-31 21:00:00	2021-12-31 22:00:00	10914.0
52964	2021-12-31 19:00:00	2021-12-31 20:00:00	2021-12-31 22:00:00	2021-12-31 23:00:00	10599.0
52965	2021-12-31 20:00:00	2021-12-31 21:00:00	2021-12-31 23:00:00	2022-01-01 00:00:00	10812.0

Figure 5.3: Data last five rows

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 52966 entries, 0 to 52965
Data columns (total 5 columns):
 #   Column                             Non-Null Count  Dtype
---  ------                             --------------  -----
 0   Start time UTC                     52966 non-null  object
 1   End time UTC                       52966 non-null  object
 2   Start time UTC+03:00               52966 non-null  object
 3   End time UTC+03:00                 52966 non-null  object
 4   Electricity consumption in Finland 52966 non-null  float64
dtypes: float64(1), object(4)
memory usage: 2.0+ MB
```

Figure 5.4: Data Information

	Electricity consumption in Finland
count	52966.000000
mean	9488.750519
std	1576.241673
min	5341.000000
25%	8322.000000
50%	9277.000000
75%	10602.000000
max	15105.000000

Figure 5.5: Descriptive Statistics

3. Data Preprocessing

To shift focus from short-term to long-term consumption forecasting, the dataset was resampled from hourly intervals to daily averages. This preprocessing step was essential in reducing noise from high-frequency fluctuations and emphasizing broader consumption trends. After resampling, the dataset was reduced to 2,184 daily data points, each representing the average electricity usage for one full day. Additionally, to maintain the temporal consistency required for effective time series modeling, the dataset was aligned to full weekly cycles. This was achieved by trimming the data to start on Monday, December 4, 2016, and end on Sunday, February 26, 2021. Such alignment aids in preserving weekly periodic patterns, which are often significant in consumption data influenced by workdays and weekends. This preprocessed dataset, being clean, aggregated, and temporally consistent, formed the foundation for model input.

DateTime	Consumption	Month	Year	Date	Time	Week	Day
2016-01-01 01:00:00	10800.0	1	2016	2016-01-01	01:00:00	53	Friday
2016-01-01 02:00:00	10431.0	1	2016	2016-01-01	02:00:00	53	Friday
2016-01-01 03:00:00	10005.0	1	2016	2016-01-01	03:00:00	53	Friday
2016-01-01 04:00:00	9722.0	1	2016	2016-01-01	04:00:00	53	Friday
2016-01-01 05:00:00	9599.0	1	2016	2016-01-01	05:00:00	53	Friday

Figure 5.6: Date-Time Indexing and Extracting Features

4. Data Visualization

Visualization played a key role in understanding the nature of electricity consumption over

time. Line plots of daily consumption across years revealed strong seasonal variations, where winters showed prominent peaks, and summers showed consistent troughs. These plots visually confirmed the existence of annual cycles. Monthly and yearly aggregates were plotted to explore inter-annual trends and to detect any unusual deviations. For instance, the downward trend in 2020 was evident in both monthly and annual summaries, reinforcing the earlier observation regarding the pandemic's influence. These visualizations not only validated the assumptions about seasonality and trend components in the data but also helped define the structure of the LSTM model by indicating that long-term memory would be necessary to capture these periodic patterns effectively.

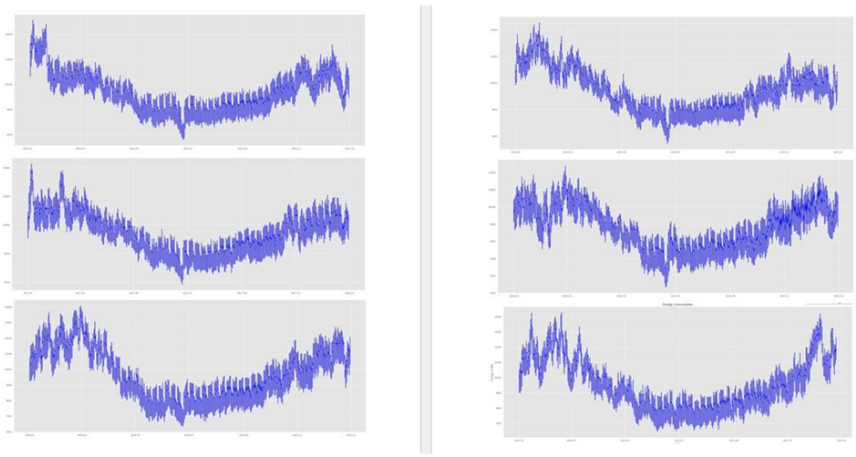


Figure 5.7: Energy Consumption by the Year

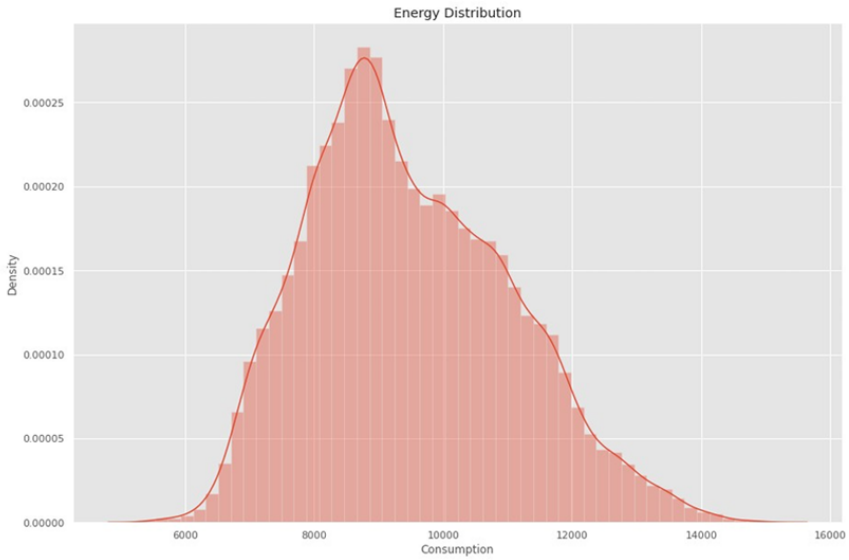


Figure 5.8: Distribution of Energy Consumption

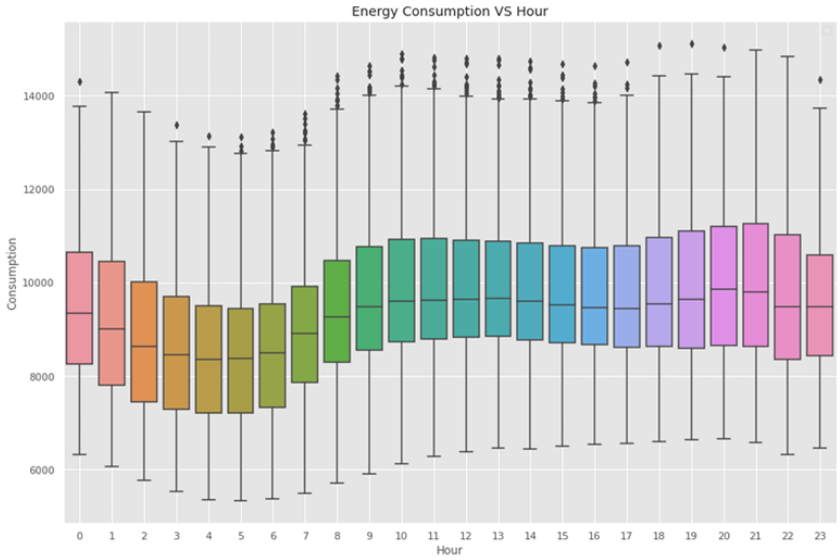


Figure 5.9: Energy Consumption VS Hour

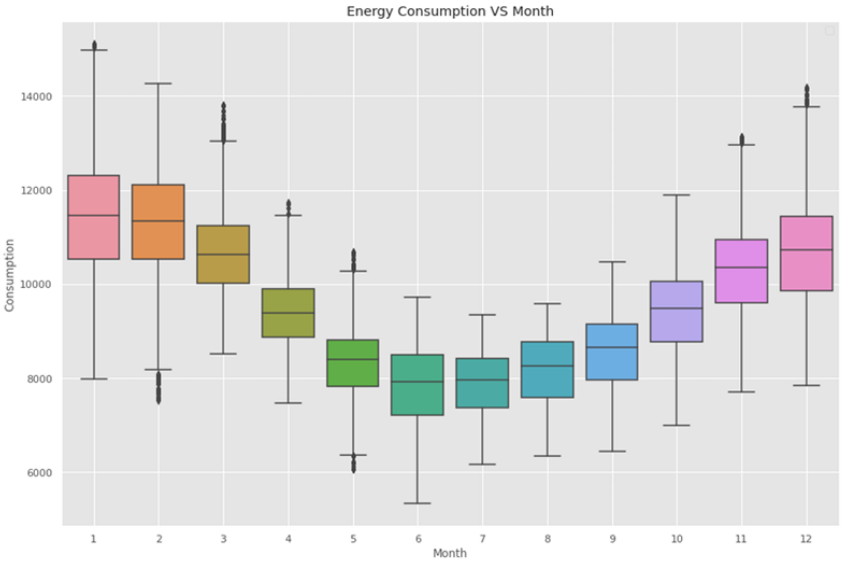


Figure 5.10: Energy Consumption VS Month

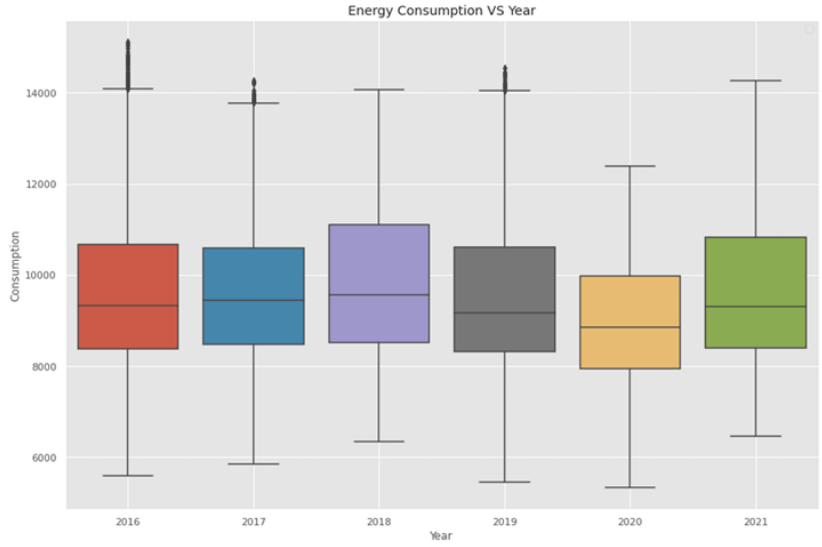


Figure 5.11: Energy Consumption VS Year

5. LSTM Model

For the purpose of forecasting, a Long Short-Term Memory (LSTM) neural network architecture was selected. LSTM is a type of Recurrent Neural Network (RNN) that is particularly well-suited for handling sequential data where past information is critical for future predictions. Unlike traditional feed-forward neural networks, LSTM networks have memory cells that allow them to retain information across long sequences. A stacked LSTM architecture was adopted to improve learning depth and accuracy. In a stacked configuration, multiple LSTM layers are placed one after the other, allowing the network to learn both low-level and high-level temporal features. This design choice was critical for capturing the multi-scale temporal patterns observed during data exploration. LSTM's ability to combat the vanishing gradient problem, common in standard RNNs, makes it a robust choice for long time series forecasting tasks like energy consumption prediction.

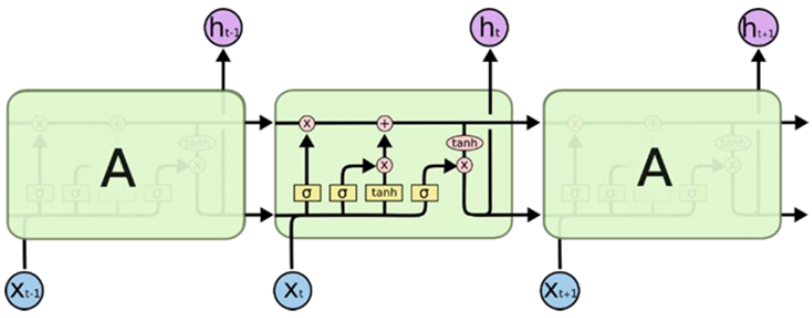


Figure 5.12: The repeating module in an LSTM (Olah, 2015)

6. Train, Validation, and Test Dataset

Before training, the Consumption values were normalized using the MinMaxScaler, transforming them into a range between 0 and 1. This normalization step ensures that all values are on a comparable scale, which is particularly important for neural networks, as it speeds up convergence and improves model performance.

The dataset was then split into three parts:

• 80 percent of the data was used for training the LSTM model.

• From the training data, 20 percent was separated as a validation set, used for tuning hyperparameters and monitoring the training process.

• The remaining 20 percent of the total data was reserved for testing, which allowed for an unbiased evaluation of the model's predictive performance.

This stratified split ensured that the model learned from a broad base of historical data, was tuned effectively, and was then evaluated on completely unseen data to estimate real-world performance.

DateTime	Consumption	Month	Year	Week
2016-01-04	12300.625000	1.0	2016.0	1.0
2016-01-05	12945.375000	1.0	2016.0	1.0
2016-01-06	13192.750000	1.0	2016.0	1.0
2016-01-07	14243.541667	1.0	2016.0	1.0
2016-01-08	14121.666667	1.0	2016.0	1.0

Figure 5.13: Daily Consumption Data

```
X_train shape:  (1297, 100, 1)
X_test shape:  (336, 100, 1)
X_val shape:  (248, 100, 1)
```

Figure 5.14: Reshape Inputs to Fit LSTM Layers

7. Model Structure

The model was constructed using the Keras Sequential API in Python. Its architecture included:

• Four stacked LSTM layers, each comprising 50 memory cells, to capture complex temporal relationships.

• Dropout layers were inserted between the LSTM layers to reduce the risk of overfitting by randomly deactivating certain neurons during training.

• A final Dense output layer was used to generate the predicted electricity consumption value for the next day.

The model was compiled using the Adam optimizer, known for its adaptability and efficiency in deep learning tasks. The primary evaluation metric was the Root Mean Squared Error (RMSE), which quantifies the average magnitude of prediction errors and is well-suited for regression problems like this one.

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm_4 (LSTM)               (None, 100, 50)           10400

 dropout (Dropout)           (None, 100, 50)           0

 lstm_5 (LSTM)               (None, 100, 50)           20200

 lstm_6 (LSTM)               (None, 100, 50)           20200

 lstm_7 (LSTM)               (None, 50)                20200

 dense_1 (Dense)             (None, 1)                 51

=================================================================
Total params: 71,051
Trainable params: 71,051
Non-trainable params: 0
```

Figure 5.15: LSTM Model Summary

8. Model Training

Training the model involved feeding it sequences of historical data. Each input sample consisted of the previous 100 days of consumption values, and the model was trained to predict the 101st day. This sliding window approach allowed the network to learn contextual patterns over a 100-day horizon.

The training process was conducted over 60 epochs with a batch size of 20. During training, both training loss and validation loss were recorded after each epoch. The validation loss was particularly useful for identifying overfitting or underfitting; a consistently low and stable

validation loss indicates good generalization capability.

Throughout the training, dropout regularization and early stopping techniques were also considered to maintain model generalizability and avoid unnecessary overtraining.

```
Epoch 43/60
65/65 [==============================] - 12s 182ms/step - loss: 0.0028 - val_loss: 0.0016
Epoch 44/60
65/65 [==============================] - 12s 182ms/step - loss: 0.0024 - val_loss: 0.0016
Epoch 45/60
65/65 [==============================] - 12s 181ms/step - loss: 0.0022 - val_loss: 0.0017
Epoch 46/60
65/65 [==============================] - 13s 198ms/step - loss: 0.0023 - val_loss: 0.0015
Epoch 47/60
65/65 [==============================] - 12s 181ms/step - loss: 0.0022 - val_loss: 0.0017
Epoch 48/60
65/65 [==============================] - 13s 195ms/step - loss: 0.0023 - val_loss: 0.0020
Epoch 49/60
65/65 [==============================] - 12s 181ms/step - loss: 0.0018 - val_loss: 0.0015
Epoch 50/60
65/65 [==============================] - 13s 197ms/step - loss: 0.0017 - val_loss: 0.0016
Epoch 51/60
65/65 [==============================] - 12s 185ms/step - loss: 0.0020 - val_loss: 0.0012
Epoch 52/60
65/65 [==============================] - 12s 183ms/step - loss: 0.0017 - val_loss: 0.0011
Epoch 53/60
65/65 [==============================] - 12s 184ms/step - loss: 0.0015 - val_loss: 0.0010
Epoch 54/60
65/65 [==============================] - 13s 198ms/step - loss: 0.0016 - val_loss: 0.0013
Epoch 55/60
65/65 [==============================] - 12s 182ms/step - loss: 0.0015 - val_loss: 0.0011
Epoch 56/60
65/65 [==============================] - 12s 183ms/step - loss: 0.0015 - val_loss: 0.0010
Epoch 57/60
65/65 [==============================] - 13s 195ms/step - loss: 0.0016 - val_loss: 0.0010
Epoch 58/60
65/65 [==============================] - 12s 182ms/step - loss: 0.0014 - val_loss: 0.0012
Epoch 59/60
65/65 [==============================] - 12s 183ms/step - loss: 0.0015 - val_loss: 0.0013
Epoch 60/60
65/65 [==============================] - 12s 185ms/step - loss: 0.0016 - val_loss: 0.0014
```

Figure 5.16: Training LSTM Model

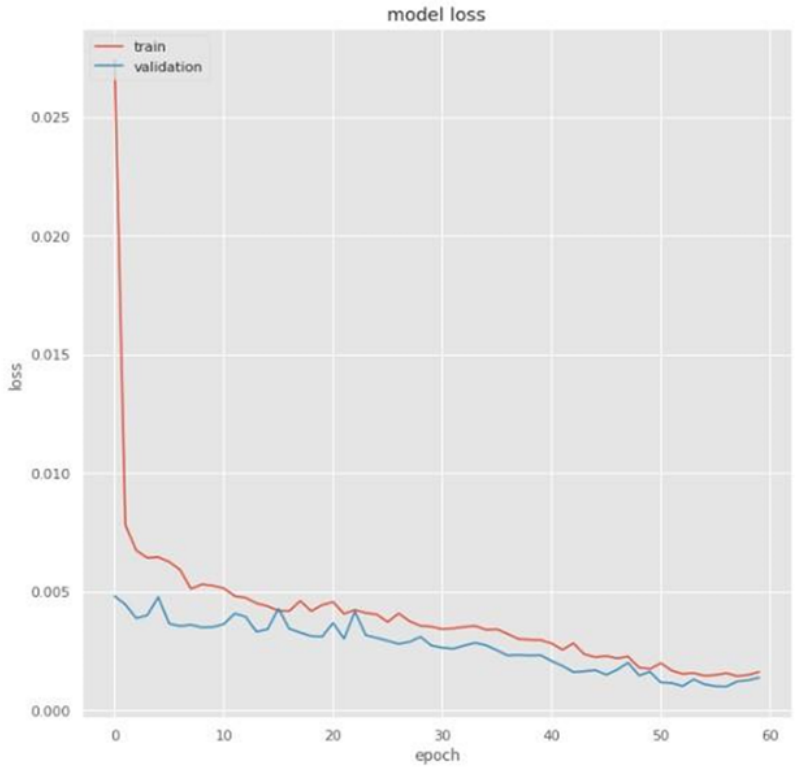


Figure 5.17: Comparison Curve of Training and Validation Accuracy of 60 Epochs

9. Final Prediction

Once the training phase was completed, the model was tested on the test set using the same sequence-to-one structure. The predicted values were then compared to actual consumption values to assess the model's performance.

The evaluation metric, RMSE, indicated how closely the model could follow real consumption trends. The resulting predictions were visually compared against actual values using line plots. The close alignment of the predicted and actual curves validated the model's effectiveness in capturing consumption behavior.

These final predictions demonstrated that the LSTM model could reliably forecast daily electricity usage, making it a suitable tool for energy planning and load balancing applications.

41/41 [==============================] - 2s 47ms/step

	Train Predictions	Actuals
0	[9533.0693359375]	[9406.708333333334]
1	[9557.21875]	[9614.791666666666]
2	[9699.4619140625]	[9894.708333333334]
3	[9583.296875]	[8933.708333333334]
4	[8622.5341796875]	[8557.208333333334]
...	...	...
1292	[8619.515625]	[9259.666666666666]
1293	[9671.5712890625]	[10248.5]
1294	[10395.3466796875]	[10360.333333333334]
1295	[10408.70703125]	[10489.833333333334]
1296	[10580.2080078125]	[10204.5]

1297 rows × 2 columns

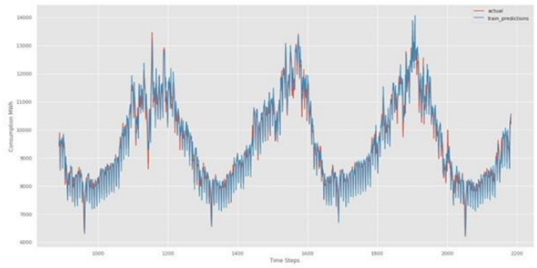


Figure 5.18: Predict Consumption Using Training Data

8/8 [==============================] - 1s 70ms/step

	Val Predictions	Actuals_val
0	[9564.7529296875]	[8931.875]
1	[9500.849609375]	[9112.916666666666]
2	[9322.599609375]	[10019.416666666666]
3	[9908.9287109375]	[10390.375]
4	[10409.23046875]	[10802.916666666666]
...	...	...
243	[8295.7900390625]	[8020.541666666667]
244	[7598.3251953125]	[7921.75]
245	[8153.66748046875]	[8705.041666666666]
246	[8841.4716796875]	[8824.708333333334]
247	[8815.4609375]	[9062.375]

248 rows × 2 columns

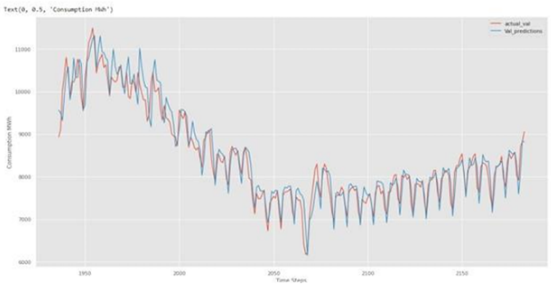


Figure 5.19: Predict Consumption Using Validation Data

11/11 [==============================] - 1s 71ms/step

	test Predictions	Actuals_test
0	[10396.9130859375]	[10032.541666666666]
1	[11196.3955078125]	[10732.125]
2	[11439.9912109375]	[10733.583333333334]
3	[10969.5]	[10971.875]
4	[11177.0654296875]	[11227.791666666666]
...	...	...
331	[12638.69921875]	[12540.25]
332	[12564.671875]	[12635.958333333334]
333	[12774.3154296875]	[11684.333333333334]
334	[11735.4423828125]	[11384.166666666666]
335	[11262.974609375]	[11581.625]

336 rows × 2 columns

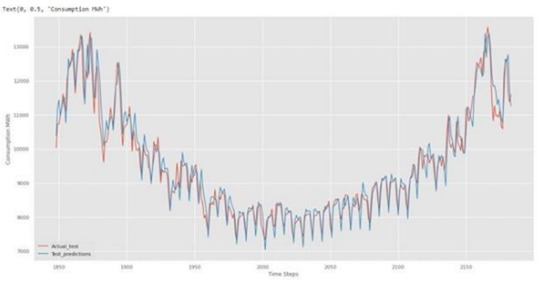


Figure 5.20: Predict Consumption Using Test Data

5.2 Data flow Diagram (Level 0, 1, 2)

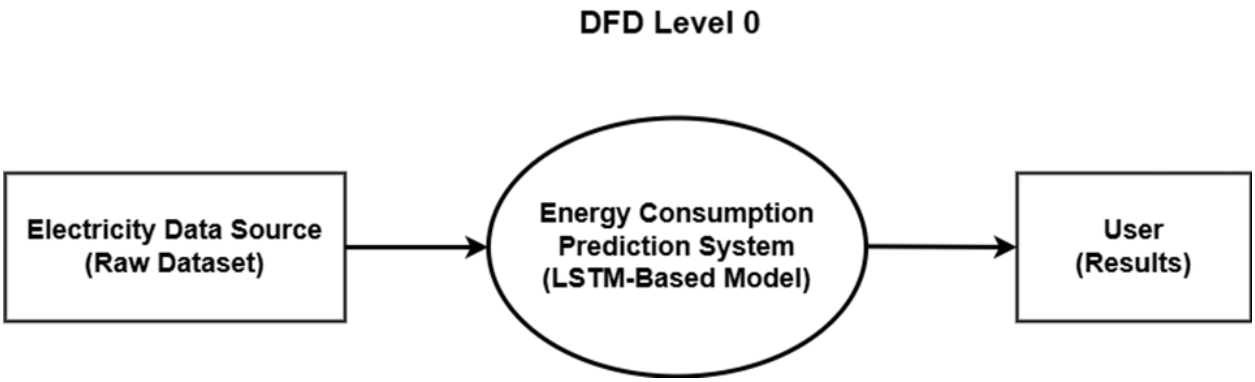


Figure 5.21: Data Flow Diagram (Level 0)

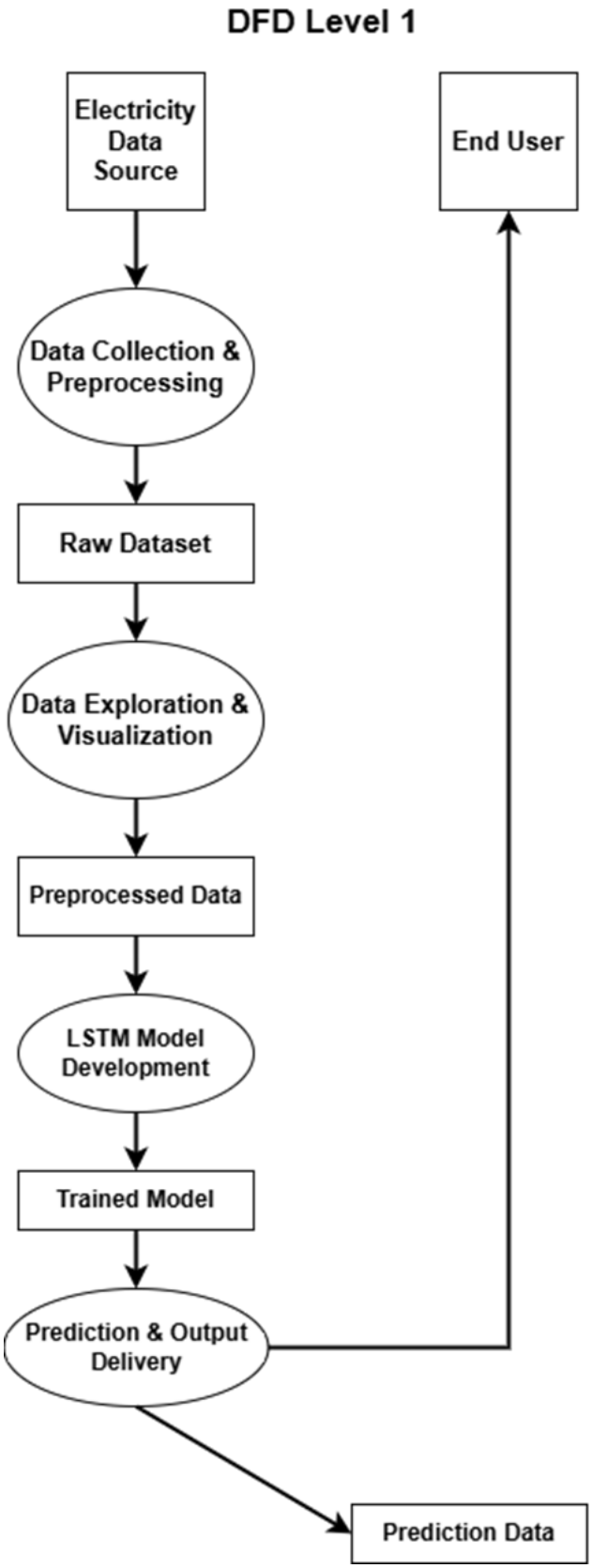


Figure 5.22: Data Flow Diagram (Level 1)

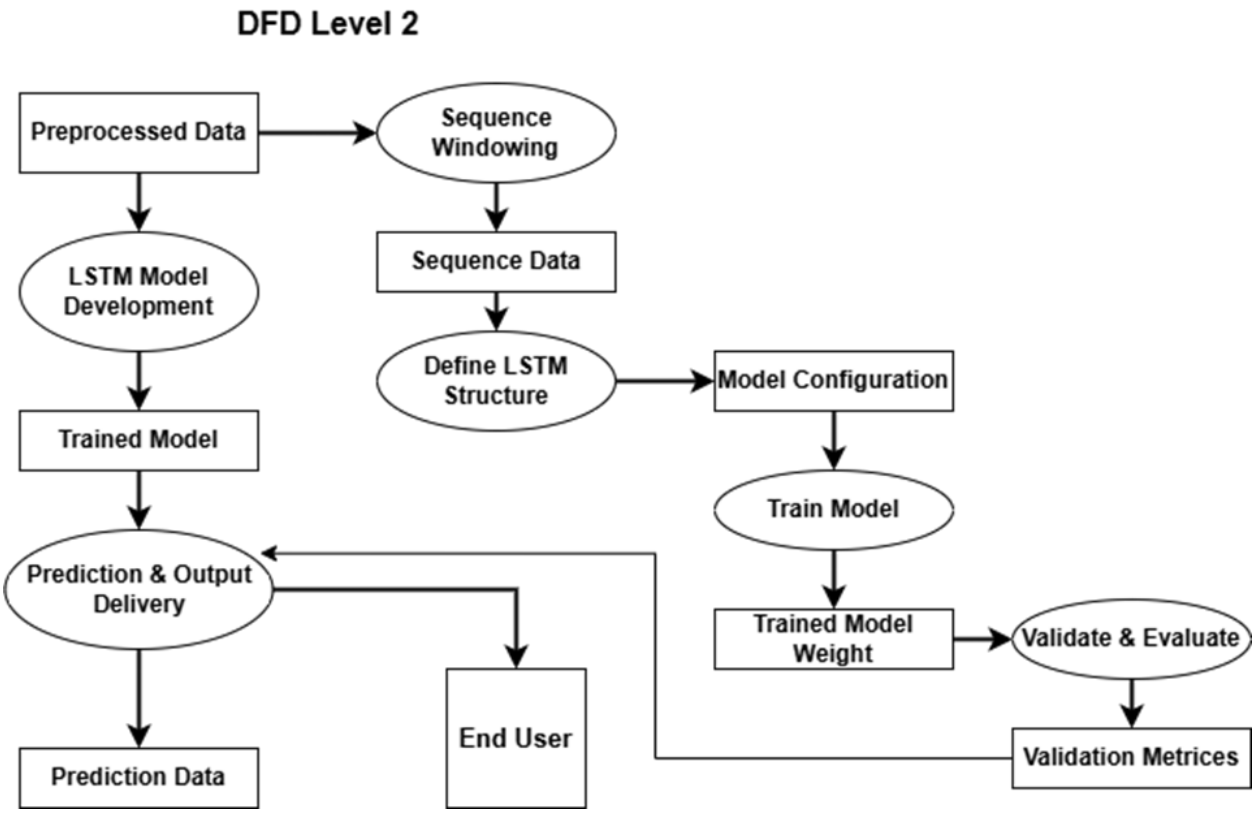


Figure 5.23: Data Flow Diagram (Level 2)

Chapter 6

SYSTEM MODELING – NEED OF SYSTEM MODELING

6.1 System Modeling – Need of System Modeling

System modeling is an essential part of designing and developing any data-driven or AI-based application, especially in the domain of energy consumption forecasting using machine learning techniques such as LSTM. It provides a visual and logical structure that supports better system understanding, communication, development, and maintenance.

1. Understanding System Complexity
System modeling allows developers and analysts to break down the overall system into smaller, manageable components. This helps in identifying the flow of data between components such as data collection, preprocessing, LSTM training, and prediction. In energy consumption forecasting, where time series data and multiple transformation steps are involved, modeling the system structure becomes essential [1].

2. Requirement Clarity and Communication
By representing processes, data flow, and system boundaries visually, system modeling ensures that both technical teams and stakeholders can clearly understand the system architecture and flow. This helps in aligning the development process with project objectives and avoids confusion during the implementation phase [2].

3. Efficient Design and Development
A well-modeled system helps in identifying redundancies and potential design flaws early in

the lifecycle. For machine learning models like LSTM, it ensures that proper data preprocessing and input shaping are planned. It also allows developers to plan for model training, validation, and deployment logically and sequentially [5].

4. Facilitates Testing and Validation

System modeling allows simulation of each system block—such as LSTM model accuracy checks or data scaling validation—before full implementation. This ensures that errors are identified early, saving time and computation. Testing the system architecture on paper or in design tools also reduces deployment risks [3].

5. Improves Scalability and Integration

When a system is properly modeled, it is easier to integrate additional data sources, upgrade models, or even shift to a real-time prediction structure. For example, switching from daily to hourly predictions or adding weather data as a feature becomes more structured with a clear model in place [6].

6. Foundation for Automation and Deployment

System modeling lays the groundwork for automation by defining input/output interfaces, data pipelines, and model interaction patterns. This is crucial when the project is scaled to real-time prediction or deployed on cloud platforms [11].

6.2 UML (Unified Modeling Language) Diagrams

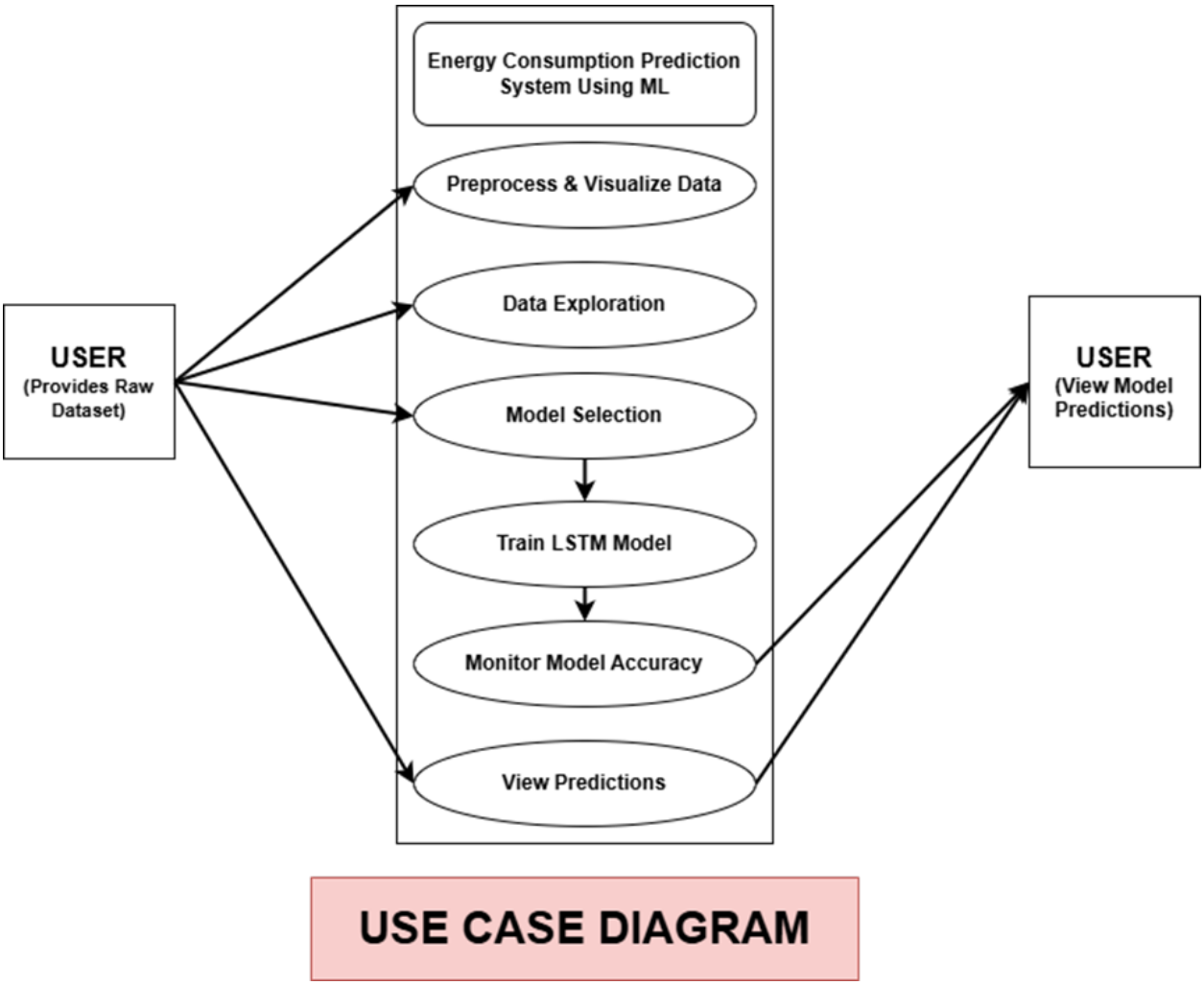


Figure 6.1: Use Case UML Diagram

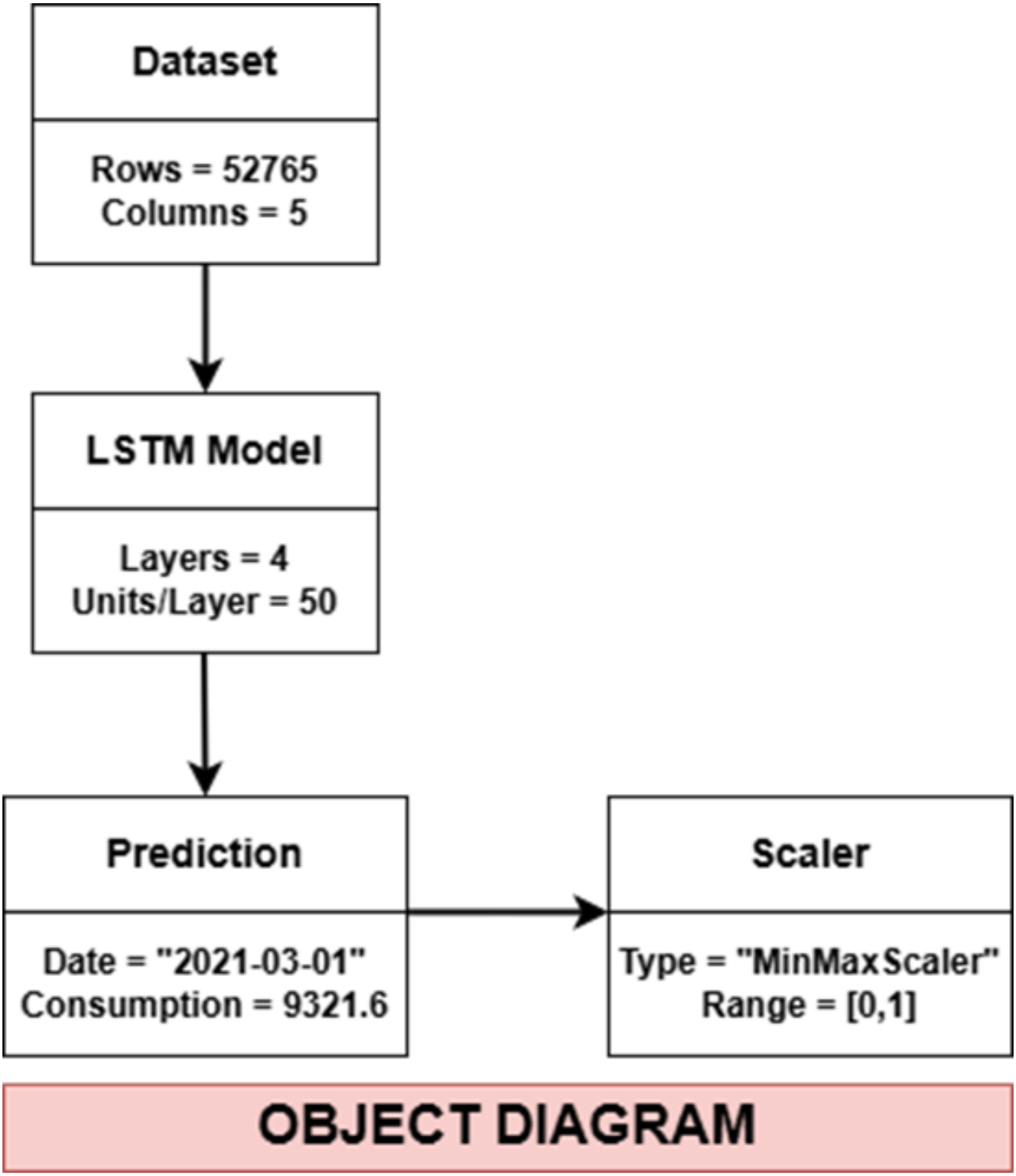


Figure 6.2: Object UML Diagram

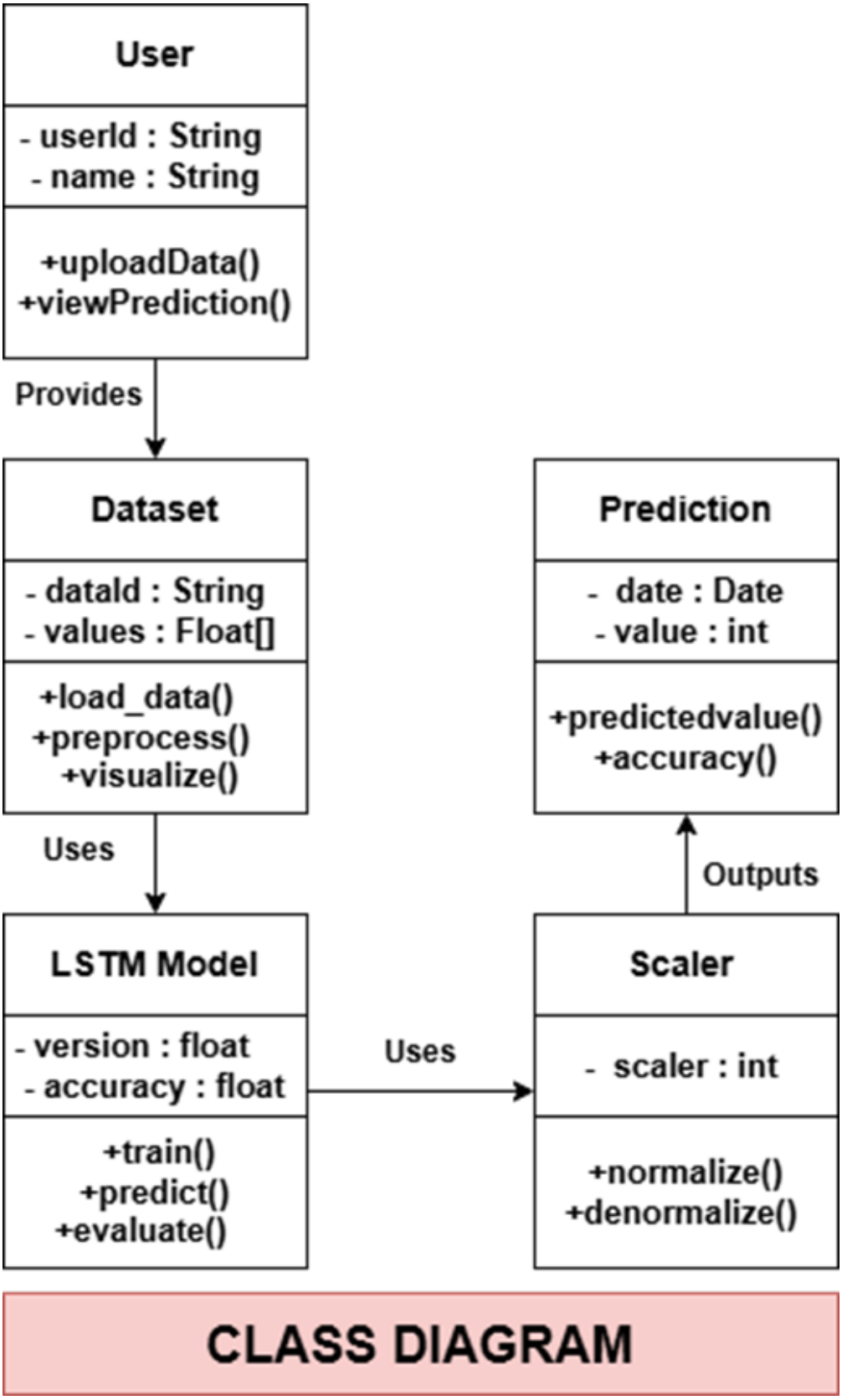


Figure 6.3: Class UML Diagram

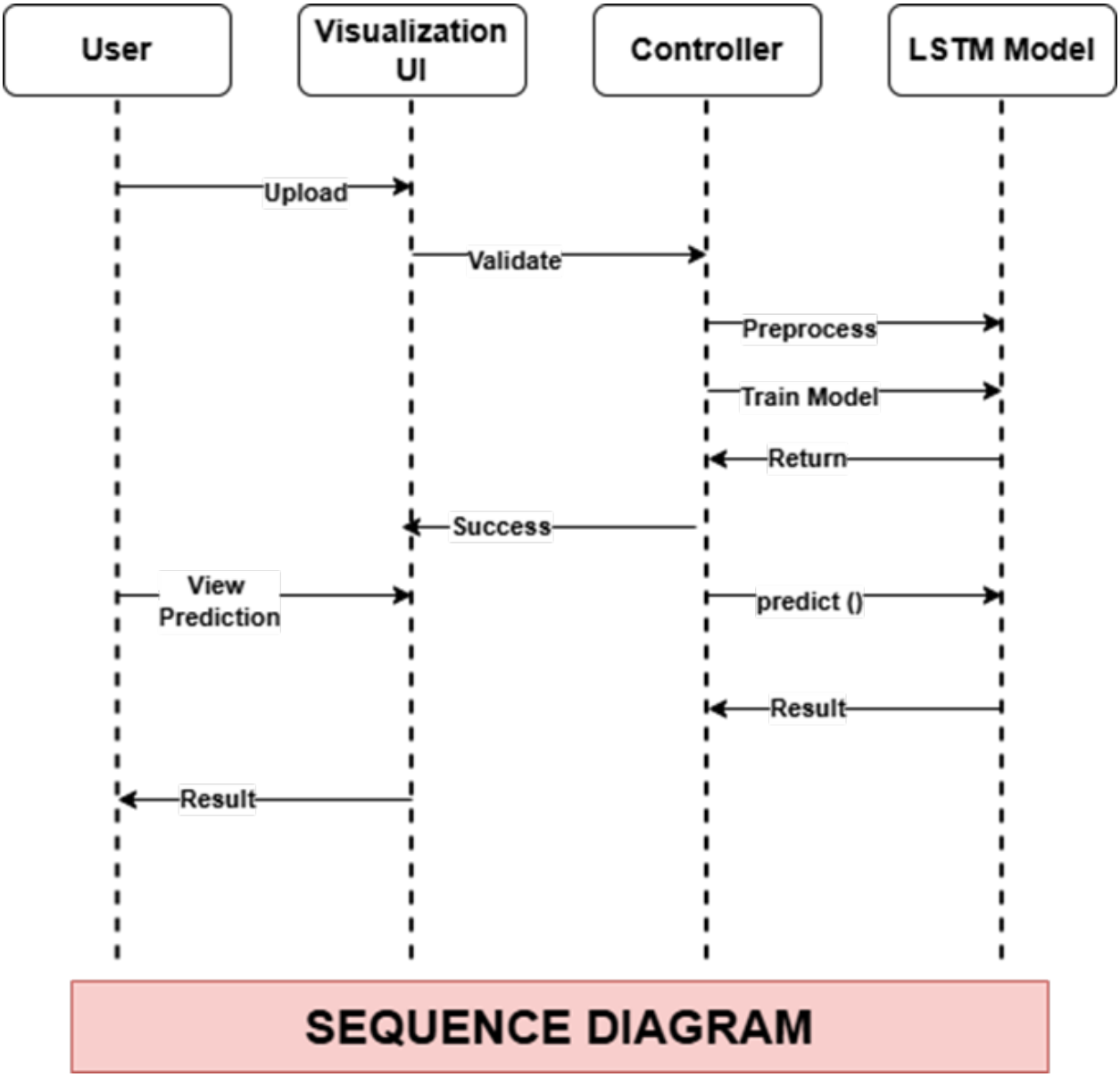


Figure 6.4: Sequence UML Diagram

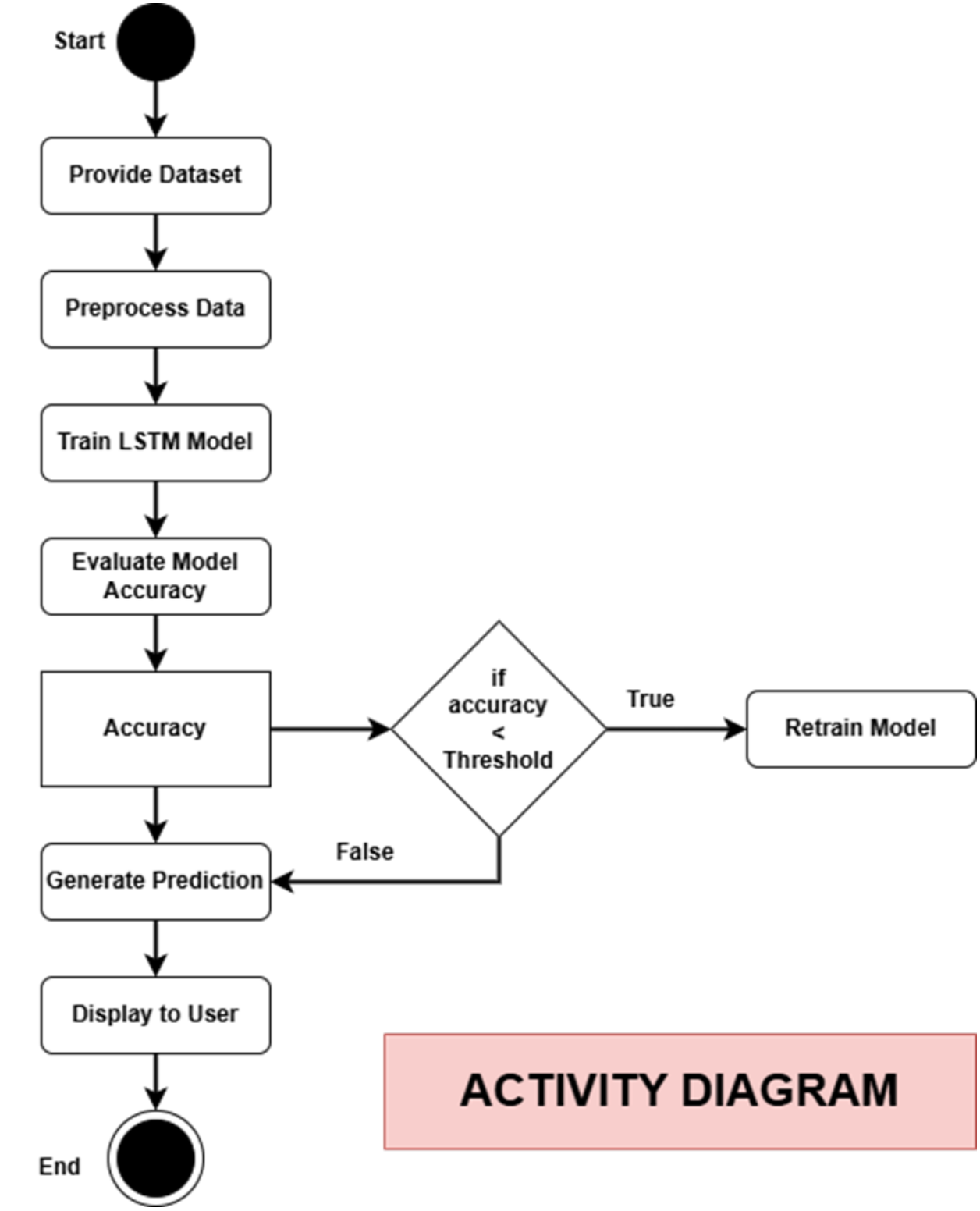


Figure 6.5: Activity UML Diagram

Chapter 7

IMPLEMENTATION PLAN FOR PROJECT STAGE - II

7.1 Hardware Specification

To process a dataset 10 times larger than in Stage-I and train a more complex/optimized ML model, upgraded hardware is essential:

- **Processor:** Intel Core i5/i7 or AMD Ryzen 5/7 – for parallel data processing.
- **RAM:** Minimum 16 GB (for loading large datasets into memory efficiently).
- **Storage:** SSD with at least 512 GB – to handle larger data writes and model storage.
- **GPU:** NVIDIA RTX 3050 or equivalent CUDA-enabled GPU – for accelerated model training, especially for neural networks like GRU, Bi-LSTM, or Transformer.

This setup will enable fast training and evaluation with high-dimensional time-series data.

7.2 Platform

Development will be carried out on more robust and scalable platforms to accommodate heavy data and experimentation:

- **Operating System**: Ubuntu 22.04 LTS (preferred for compatibility with ML frameworks) or Windows 11.
- **IDE**: VSCode or PyCharm – with support for Python notebooks and debugging.

7.3 Programming Language Used

The implementation remains in Python, enhanced with more scalable ML and deep learning frameworks:

- **Data Handling:** pandas, numpy, or dask (for out-of-core computation).
- **Visualization:** matplotlib, seaborn, plotly (interactive visualization).
- **Modeling:**
  - **scikit-learn** – for classical ML models (e.g., Random Forest, Gradient Boosting).
  - **XGBoost / LightGBM** – for optimized tree-based models (faster with large data).
  - **TensorFlow / Keras** – for deep models like GRU, Bi-LSTM.

These tools provide better scalability, speed, and modeling options for big data.

7.4 Software/Hardware Development

The development process will adapt to the new scope as follows:

- **Data Engineering:**
  - Use batch loading and lazy evaluation to avoid memory overload.
  - Apply advanced feature engineering such as lag variables, rolling stats, and external weather/holiday effects
- **Model Selection and Optimization:**
  - **GRU (Gated Recurrent Unit):** Faster training than LSTM with similar performance
  - **XGBoost/LightGBM:** Excellent for tabular time-series with fewer tuning needs
  - **Bi-LSTM or Transformer:** For capturing long-term dependencies
- Apply cross-validation, grid/random search, and Bayesian optimization for hyperparameter.
- **Evaluation:**
  - Evaluate using RMSE, MAE, and possibly MAPE.

  - Use visual plots (predicted vs actual) and learning curves to assess model behavior.

- **Optimization and Deployment:**
  - Export models using TensorFlow Saved Model format.

Chapter 8

CONCLUSIONS

8.1 Conclusions

The first stage of this project focused on developing a deep learning model to forecast energy consumption in Finland using historical data. A univariate time series dataset containing six years of electricity usage was preprocessed and converted from hourly to daily format for efficient modeling. Each sample was designed to include 100 consecutive days of energy usage to predict the next day's value. The LSTM model architecture implemented in this stage consisted of four stacked LSTM layers, followed by a Dropout layer to avoid overfitting, and a single dense output layer. The model was trained using the Adam optimizer and evaluated using the Root Mean Squared Error (RMSE), which proved effective in quantifying prediction accuracy. The model exhibited promising results, particularly for short-term prediction. However, its performance is highly sensitive to the time window size used for input sequences, which was fixed at 100 days for this study. Different time step values could yield varying results and must be explored further for optimization.

8.2 Future Scope

In Stage-II, the work will be expanded using a dataset nearly ten times larger than the current one, enhancing the model's ability to generalize across broader patterns. Additionally, alternate machine learning models—potentially more optimized for large-scale forecasting—will be considered, including GRU, XGBoost, and Transformer-based architectures. This ongoing development will aim to achieve higher prediction accuracy, better scalability, and deeper insights into energy consumption patterns across different time frames and seasonal variations.

8.3 Application

The proposed system provides following applications.

- Accurate energy demand forecasting for utility companies in Finland.
- Support for energy policy planning and infrastructure scaling.
- Real-time monitoring and forecasting systems for smart grids.
- Integration with IoT systems for predictive power management in residential and industrial settings.

Bibliography

[1] Mosavi, Amir Bahmani, Abdullah, *"Energy Consumption Prediction Using Machine Learning; A Review", MDPI, 2019.*

[2] Nitesh Kushwaha Akhilesh A. Waoo, *"Energy Consumption Prediction by Using Machine Learning", International Journal for Multidisciplinary Research (IJFMR), December 2023.*

[3] C. Ragupathi, S. Dhanasekaran, N. Vijayalakshmi, Ayodeji Olalekan Salau, *"Prediction of electricity consumption using an innovative deep energy predictor model for enhanced accuracy and efficiency", Energy Reports, Volume 12, 2024, pp. 5320-5337.*

[4] Frikha, M.; Taouil, K.; Fakhfakh, A.; Derbel, F. *"Predicting Power Consumption Using Deep Learning with Stationary Wavelet", Forecasting, 2024, 6, pp. 864-884.*

[5] Reddy, G. Aitha, Lakshmi Poojitha, Ch Shreya, A. Reddy, D. Meghana, Gara., *"Electricity Consumption Prediction Using Machine Learning" E3S Web of Conferences, 2023.*

[6] T. C. Brito and M. A. Brito, *"Forecasting of Energy Consumption : Artificial Intelligence Methods", 17th Iberian Conference on Information Systems and Technologies (CISTI), Madrid, Spain, 2022, pp. 1-4.*

[7] P. Vijayan, *"Energy Consumption Prediction in Low Energy Buildings using Machine learning and Artificial Intelligence for Energy Efficiency", 8th International Youth Conference on Energy (IYCE), Hungary, 2022, pp. 1-6.*

[8] Dinmohammadi, F.; Han, Y.; Shafiee, M. *"Predicting Energy Consumption in Residential Buildings Using Advanced Machine Learning Algorithms", Energies, 2023.*

[9] Bourhnane, S., Abid, M.R., Lghoul, R. et al. *"Machine learning for energy consumption prediction and scheduling in smart buildings", SN Appl. Sci. 2, 297, 2020.*

[10] Qureshi, M., Arbab, M.A. Rehman, S. *"Deep learning-based forecasting of electricity consumption", Sci Rep 14, 6489, 2024.*

[11] Mohamad Nachawati, *"Energy Consumption Prediction using Machine Learning Time Series Forecasting", LAB University of Applied Science, 2022.*