German International University in Berlin

Faculty of Media Engineering and Technology

Simulator for Controlling Irrigation in Smart Farms

Bachelor Thesis

Author: Youssef Magdy Abdelrahman

Supervisors: Prof. Dr. Mohamed KHALGUI

Submission Date: 23 June, 2025

German International University in Berlin

Faculty of Media Engineering and Technology

Simulator for Controlling Irrigation in Smart Farms

Bachelor Thesis

Author: Youssef Magdy Abdelrahman

Supervisors: Prof. Dr. Mohamed KHALGUI

Submission Date: 23 June, 2025

This is to certify that:

(i) the thesis comprises only my original work toward the Bachelor Degree

(ii) due acknowledgement has been made in the text to all other material used

---

Youssef Magdy Abdelrahman
23 June, 2025

Acknowledgments

Firstly and Lastly, thanks to God

Abstract

In the face of increasing water scarcity and the growing global demand for food, efficient water resource management in agriculture has become more critical than ever. Evapotranspiration (ET), representing the combined loss of water through evaporation and plant transpiration, is a fundamental factor in determining crop water requirements. Accurate prediction of ET, a key determinant of crop water requirements, forms the core of this system. While traditional ET estimation methods present challenges in scalability and accessibility, recent advancements in Machine Learning (ML) offer promising avenues for enhanced precision. This project investigates the application of various ML models to predict ET using a 15-year historical weather dataset obtained from a weather Application Programming Interface (API). It includes parameters such as average temperature, humidity, wind speed, precipitation, and sunlight hours. By introducing a software simulator that is designed to support smart irrigation planning by leveraging historical and future weather data. The outcome is a user-friendly, data-driven decision support tool that enables farmers to enhance crop yield while minimizing water waste. This project demonstrates the potential of integrating machine learning with environmental data to promote sustainable and intelligent agricultural practices and contribute to the development of a practical tool for efficient agricultural water management by providing users with data-driven irrigation recommendations.

Contents

Chapter 1

Introduction

1.1 Water Scarcity and Importance of Irrigation

Water is an indispensable resource for global agriculture, underpinning food security and economic stability. The Food and Agriculture Organization of the United Nations (FAO) highlights the sector's significant water footprint, reporting that agriculture accounts for 70% of global freshwater withdrawals [1]. With projections indicating a doubling of water demand for irrigated food production by 2050, the efficient management of this vital resource becomes paramount, particularly in the face of increasing water scarcity and the exacerbating effects of climate change [2]. While advancements in irrigation practices aim to mitigate this pressure, the need for more precise and adaptive water management strategies remains critical.

1.2 Limitations of Traditional Methods

Within this landscape, water management stands out as a crucial area for AI intervention. Traditional irrigation methods often struggle with accurately matching water supply to crop demand, leading to inefficiencies such as over-irrigation and under-irrigation. Over-irrigation results in water wastage, increased energy costs, and potential environmental pollution through the leaching of fertilizers and pesticides. Conversely, under-irrigation can stress crops, reduce yields, and contribute to soil salinization, particularly in arid and semi-arid regions [3].

1.3 AI/ML in Agriculture

The advent of Artificial Intelligence (AI), and specifically Machine Learning (ML), is transforming numerous domains, and agriculture is no exception [3]. AI offers powerful tools to address the complex challenges of modern farming, including the optimization of resource utilization, enhancement of crop yields, and the promotion of sustainable practices. As illustrated in [4], AI applications in agriculture span various critical areas, encompassing water management, soil health optimization, livestock monitoring, and the improvement of crop management practices (Fig. 1, adapted from [4]).

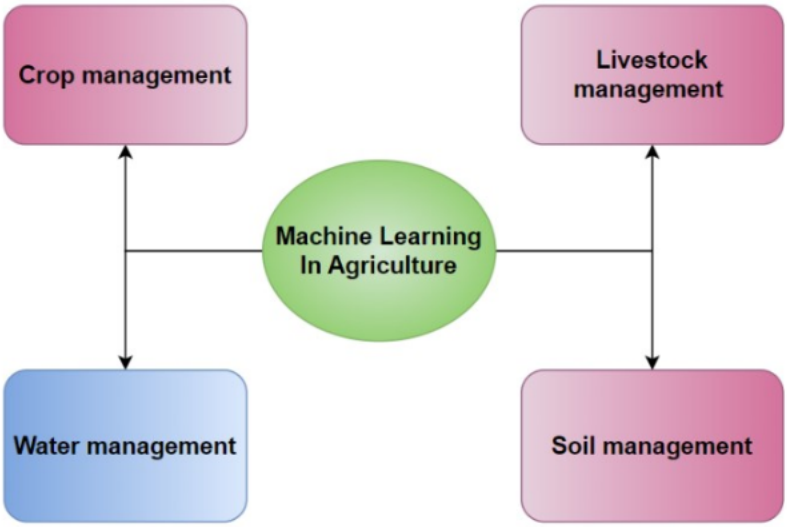


Figure 1.1: Categories of AI Applications in Agriculture

1.4 ML for ET Prediction and Smart Irrigation

Recent advancements in machine learning offer promising alternatives to enhance the accuracy and efficiency of ET estimation and the management of irrigation systems. ML algorithms can learn complex relationships from historical data, including meteorological variables, soil characteristics, and even remote sensing information, to predict ET with increasing precision. Furthermore, ML techniques are being integrated into smart irrigation systems to automate irrigation scheduling based on real-time data and predicted future conditions, leading to significant improvements in water use efficiency [5].

1.5 Project-Specific Details

This Bachelor's project focuses on the development of a software simulator for a smart irrigation system, with a core emphasis on leveraging machine learning for accurate ET prediction. By utilizing a 15-year historical weather dataset obtained from a weather API, various ML models were trained and rigorously evaluated for their ability to predict daily ET. The superior performance of the SVM model in this evaluation forms the foundation of the simulator's ET prediction capability.

1.6 Project Aims and Contributions

This project contributes to the growing body of research on AI-driven solutions for sustainable agriculture, specifically addressing the critical challenge of efficient water management. By providing a practical and user-friendly simulation tool, this work aims to empower users with the information needed to optimize their irrigation practices, conserve water resources, and potentially enhance crop yields. The subsequent chapters of

this thesis will delve into the background of smart irrigation and ET, the methodology employed for data collection and model development, the results of the model evaluation, the architecture and functionality of the developed simulator, and finally, the conclusions and potential future directions of this research.

Chapter 2

Background

2.1 Technical Context

2.1.1 Evapotranspiration (ET)

Evapotranspiration (ET) is a critical parameter in irrigation management, representing the sum of evaporation from the soil surface and transpiration from plants. It is the combined process of water evaporation from the soil and transpiration from plants. Accurate estimation of ET is crucial for effective irrigation scheduling, ensuring crops receive the necessary water without wastage. Traditional methods for estimating ET include the Penman-Monteith equation and the Hargreaves-Samani method, which rely on meteorological data such as temperature, humidity, wind speed, and solar radiation. The Food and Agriculture Organization (FAO) provides guidelines for computing crop water requirements based on ET, emphasizing its significance in agricultural water management [6].

Recent advancements have introduced machine learning techniques to estimate ET more efficiently. For instance, ensemble Kalman filters and maximum likelihood estimation methods have been employed to infer spatially varying ET rates from soil moisture measurements, enhancing the precision of irrigation systems [7].

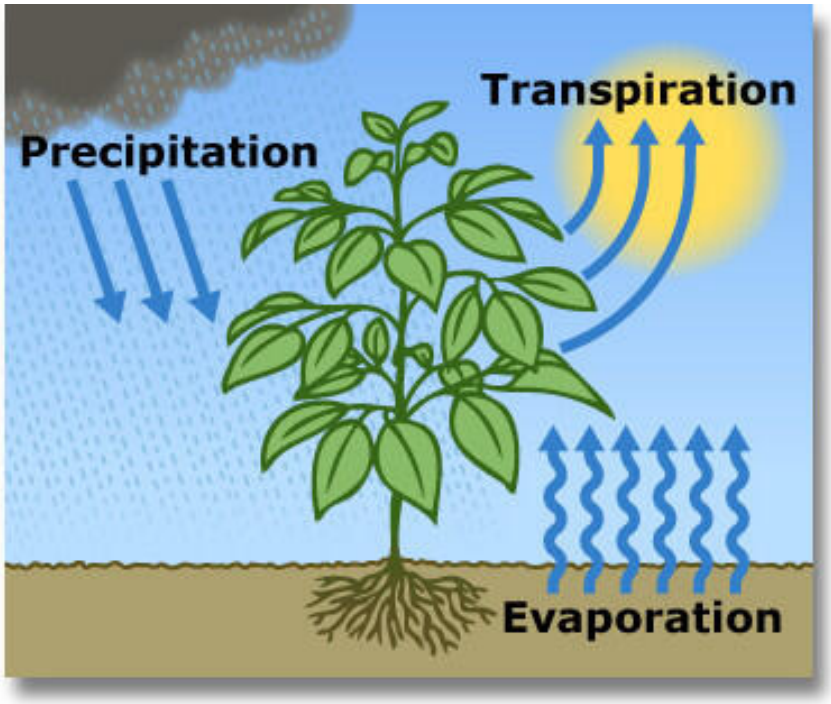


Figure 2.1: Evapotranspiration Process

2.1.2 Smart Irrigation Systems

Smart irrigation systems integrate sensors, weather data, and automated control mechanisms to optimize water usage in agriculture. These systems aim to provide water to crops precisely when and where it's needed, reducing waste and improving yield. The integration of Internet of Things (IoT) devices allows for real-time monitoring and control, enabling adaptive responses to changing environmental conditions.

Machine learning models, such as decision trees and support vector machines, have been applied to classify irrigation statuses accurately, contributing to the development of intelligent irrigation scheduling systems.

2.2 Relevant Technologies

2.2.1 Weather APIs

Access to accurate and historical weather data is vital for predicting ET and scheduling irrigation. Weather Application Programming Interfaces (APIs) provide future, real-time, and historical weather data, which are crucial for estimating ET and making irrigation decisions. In smart irrigation systems, weather APIs provide information on parameters such as temperature, humidity, wind speed, solar radiation, and precipitation, enabling dynamic adjustment of irrigation schedules based on current and forecast weather conditions. Utilizing such data enables the development of predictive models that can forecast irrigation needs based on weather patterns.

2.2.2 Sensors and Internet of Things Devices

Sensors play a pivotal role in smart irrigation by collecting data on soil moisture, temperature, humidity, and other environmental factors. These sensors provide real-time feedback, allowing the system to respond promptly to changing conditions and maintain optimal soil moisture levels for crop growth [8]. IoT devices facilitate the collection and transmission of this data, allowing for automated decision-making in irrigation systems. The integration of these technologies leads to more responsive and efficient water management practices.

2.2.3 Machine Learning Techniques

Machine Learning (ML) techniques have been increasingly adopted in smart irrigation to predict ET and optimize water usage. Models such as Extreme Gradient Boosting, Random Forests, Support Vector Machines, and Neural Networks have demonstrated effectiveness in analyzing complex datasets to forecast irrigation requirements accurately.

For instance, ML models have been utilized to estimate ET by analyzing weather parameters, leading to more precise irrigation scheduling and water conservation. In addition to, systems utilizing embedded technologies and real-time data transmission have demonstrated improved irrigation management and support for sustainable agriculture [9]. These systems serve as benchmarks and inspiration for developing advanced irrigation simulators.

2.2.4 Existing Systems

Several smart irrigation systems have been developed, incorporating IoT devices, sensors, and ML algorithms to optimize water usage. For instance, systems utilizing embedded technologies and real-time data transmission have demonstrated improved irrigation management and support for sustainable agriculture. These systems integrate various technologies to improve water management.

- **IoT-Based Systems:** These systems utilize sensors and IoT devices to monitor soil moisture and environmental conditions, enabling automated irrigation based on real-time data.
- **ML-Driven Systems:** Incorporating machine learning algorithms allows for predictive analysis of irrigation needs, improving the efficiency and effectiveness of water usage [10].
- **Weather Forecast Integration:** Some systems integrate weather forecasting to anticipate irrigation requirements, adjusting schedules proactively to conserve water and maintain crop health.

These systems serve as a foundation for developing advanced irrigation simulators that can predict ET and optimize water distribution based on various parameters.

2.2.5 Terminology

- **Evapotranspiration (ET):** The combined process of evaporation from the soil and transpiration from plants.
- **Crop Evapotranspiration (ETc):** The crop-specific evapotranspiration, calculated by multiplying ET with a Crop Coefficient (Kc).
- **Reference Evapotranspiration (ETo):** A standard measure of ET from a reference surface, typically grass, used as a baseline for calculating ETc.
- **Crop Coefficient (Kc):** A factor that adjusts ET to reflect the water use of specific crops.
- **Internet of Things (IoT):** A network of interconnected devices that collect and exchange data.
- **Machine Learning (ML):** A subset of artificial intelligence that enables systems to learn from data and make predictions.
- **Application Programming Interface (API):** A set of protocols that allow different software applications to communicate and share data.
- **Smart Irrigation:** An automated irrigation approach that uses data from sensors, weather forecasts, and ML models to optimize water usage.

Chapter 3

Methodology

3.1 Data Collection

In this research, historical daily weather data was collected to develop and validate the proposed smart irrigation system. The data was obtained from the WorldWeatherOnline API, covering a continuous period of 15 years (from 2010 to 2024), resulting in a total of 5471 daily records.

Each record contains key meteorological parameters necessary for the estimation of evapotranspiration (ET), specifically:

- **Average Temperature (avgtemp)** [°C]
- **Maximum Temperature (maxtemp)** [°C]
- **Minimum Temperature (mintemp)** [°C]
- **Average Wind Speed (avgwindspeed)** [km/h]
- **Average Humidity (avghumidity)** [%]
- **Total Precipitation (totalprecip)** [mm]
- **Sunshine Hours (sunHour)** [hours]

An excerpt of the collected dataset is presented in Table 3.1 to illustrate the structure of the data:

Table 3.1: Sample of the collected weather data and calculated ET

avgtemp	maxtemp	mintemp	avgwindspeed	avghumidity	totalprecip	sunHour	ET
23.5	28.0	18.2	10.5	60	0.5	8.5	2.35
25.0	30.0	20.1	8.0	65	0.0	9.2	2.70
21.7	26.3	17.5	12.2	58	1.0	7.8	2.20
26.1	31.4	20.6	7.5	63	0.2	9.8	2.90
24.3	29.0	19.3	9.7	70	0.8	8.1	2.50
22.8	27.1	18.0	11.0	66	1.3	7.5	2.15
27.5	32.2	22.8	6.3	60	0.0	10.0	3.00
20.9	25.5	16.8	13.5	55	0.6	6.9	2.10
25.8	30.6	21.0	9.0	68	0.4	8.6	2.65
23.0	27.7	18.5	10.1	62	0.7	8.0	2.40

To calculate the daily reference evapotranspiration (ET_0), the well-established **Penman-Monteith** equation was used. This method, recommended by the Food and Agriculture Organization (FAO-56), combines principles of energy balance and aerodynamic transport. It considers solar radiation, temperature, humidity, and wind speed to accurately estimate the evapotranspiration rate. The equation is given as:

$$ET_0 = \frac{0.408\Delta(R_n - G) + \gamma \frac{900}{T+273} u_2 (e_s - e_a)}{\Delta + \gamma(1 + 0.34u_2)} \tag{3.1}$$

where:

- ET_0 is the reference evapotranspiration (mm/day),
- Δ is the slope of the vapor pressure curve (kPa/°C),
- R_n is the net radiation at the crop surface (MJ/m^2/day),
- G is the soil heat flux density (MJ/m^2/day),
- γ is the psychrometric constant (kPa/°C),
- T is the mean daily air temperature at 2 meters height (°C),
- u_2 is the wind speed at 2 meters height (m/s),
- e_s is the saturation vapor pressure (kPa),
- e_a is the actual vapor pressure (kPa).

Given that some parameters such as net radiation and vapor pressures were not directly available, they were estimated using empirical equations based on the available weather features following FAO-56 guidelines.

A graphical representation was also plotted to visualize the behavior of ET_0 over time. As shown in Figure 4.5, the seasonal trend is clearly observed, where ET_0 increases during the summer months due to higher temperatures and solar radiation, and decreases during winter when temperatures and sunshine hours are lower.

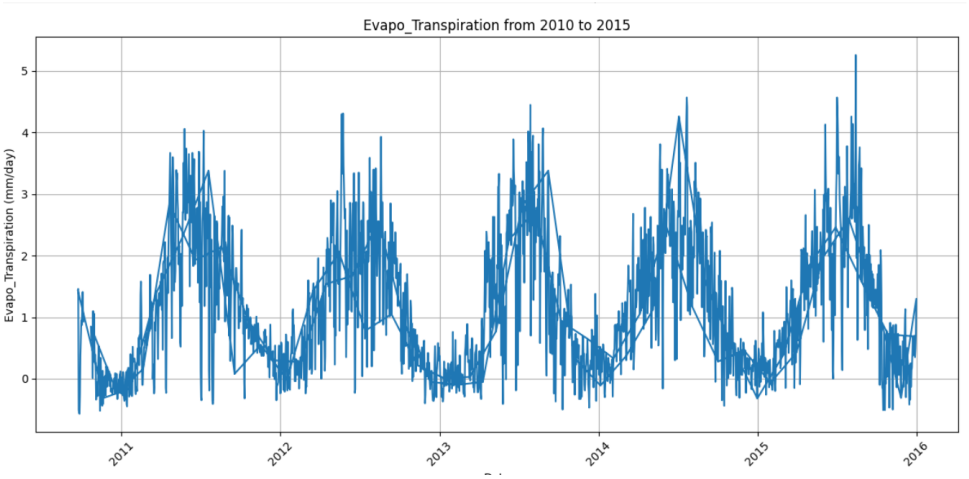


Figure 3.1: Trend of daily ET_0 from 2010 to 2015 showing seasonal variations

This collected and pre-processed dataset served as the foundation for the development of machine learning models to predict ET_0 and optimize the irrigation system.

3.2 System Design

The software simulator was designed to estimate irrigation requirements based on forecasted weather conditions. It:

- Accepts user inputs (date, location, crop type, soil type, area, moisture levels)
- Fetches future weather data
- Predicts ET using a machine learning model
- Computes crop evapotranspiration (ETc)
- Calculates required water based on soil moisture and rainfall

3.3 User Inputs

- Date
- Location (City)

- Crop Type: Olive, Date, Orange
- Soil Type: Loamy, Sandy, Clay
- Time Season: Initial, Mid-season, Late-season
- Area (m^2)
- Initial Soil Moisture Level (L/m^2)
- Maximum Soil Moisture Level (L/m^2)

3.4 ET and ETc Calculation

- ET is predicted using the trained models.
- ETc (Crop ET): $ETc = Kc \times ET$

Crop	**Initial** K_c	**Mid-season** K_c	**Late-season** K_c
Rice	1.05 – 1.20	1.20 – 1.35	0.90 – 1.05
Olives	0.30 – 0.50	0.60 – 0.75	0.55 – 0.65
Date Palms	0.90 – 1.00	0.95 – 1.10	0.85 – 0.90
Oranges (Citrus)	0.65 – 0.75	0.85 – 0.95	0.70 – 0.80

Table 3.2: Crop coefficient (K_c) values for different growth stages

- Effective Rain: Effective Rain = totalprecip_mm × Infiltration Efficiency

Soil Type	**Infiltration Efficiency**
Loamy	50% – 80%
Sandy	75% – 95%
Clay	30% – 50%

Table 3.3: Soil types and their corresponding infiltration efficiency ranges

- Required Water:

  Water = (Max Moisture Level−(Initial Moisture Level+Rain Contribution−ETc))×Area

3.5 ML Algorithms Overview

Algorithm	Strengths	Limitations	Applications
XGBoost	High accuracy, handles missing values, supports regularization to reduce overfitting.	Requires careful hyperparameter tuning, can be complex and time-consuming to optimize.	Regression and classification tasks, ET prediction, tabular data modeling.
CNN	Excellent at extracting features from spatial or grid-like data such as images.	Requires large labeled datasets and is not inherently suited for sequential data like time series.	Pattern recognition, image processing, occasionally used in ET prediction via weather grid maps.
ANN	Learns complex nonlinear patterns, general-purpose and adaptable.	Sensitive to initialization and architecture choices, often acts like a black box.	Basic modeling, regression/classification, part of ensemble deep learning pipelines.
DT	Simple to interpret and visualize, performs well on small datasets.	Easily overfits, especially on noisy data, and may not generalize well.	Rule-based systems, decision logic modeling, ET estimation as baseline.
RF	Reduces overfitting by averaging multiple trees, handles missing and categorical data well.	Interpretability is lower than a single tree, may be slow with many trees.	Ensemble forecasting, regression for irrigation models, ET prediction.
PROPHET	Captures seasonality, holidays, and trends, easy to use with few parameters.	Struggles with very noisy data or abrupt shifts, assumes additive seasonality.	Business time series forecasting, crop season patterns.
LSTM	Retains long-term dependencies in sequence data, ideal for time-dependent patterns.	Training is computationally intensive, prone to overfitting, needs large datasets.	ET time series forecasting, rainfall and temperature sequence modeling.
DNN	Flexible architecture, models complex nonlinearities, general-purpose learning system.	Requires a lot of data and compute, needs careful tuning and may overfit.	Time-series regression, ET estimation, general prediction systems.
SARIMAX	Captures seasonal trends and includes external features (exogenous variables).	Assumes linearity, requires manual tuning of seasonal parameters.	Time-series forecasting with known external influences (e.g., rainfall, temperature).

3.6 Evaluation Methods

Machine Learning models were evaluated using three main statistical metrics: RMSE, MAE, and R^2. This chapter focuses on explaining the system methodology, while detailed performance comparisons between models are presented in Chapter 5.

Metric	**Description**
Root Mean Square Error (RMSE)	RMSE measures the square root of the average of squared differences between predicted and actual values. It heavily penalizes large errors, making it sensitive to outliers. A lower RMSE indicates better model performance.
Mean Absolute Error (MAE)	MAE calculates the average of the absolute differences between predicted and actual values. It treats all errors equally, providing a clear interpretation of the average magnitude of errors without considering their direction.
Coefficient of Determination (R^2 Score)	R^2 represents the proportion of the variance in the dependent variable that is predictable from the independent variables. An R^2 value of 1 indicates perfect prediction, while a value close to 0 indicates that the model fails to capture the variability of the target variable.

Table 3.4: Evaluation metrics used to assess Machine Learning model performance

Chapter 4

Implementation

This chapter explains the technical steps taken to implement the Smart Irrigation System Simulator. The system combines Machine Learning models, a web-based User Interface (UI), and a database to provide daily evapotranspiration (ET) predictions for irrigation management.

4.1 Technology Stack

The technologies used for building the system include:

- **Backend Development:** Python, FastAPI
- **Frontend Development:** Angular
- **Machine Learning:** TensorFlow, Scikit-learn, Prophet, XGBoost
- **Data Analysis and Visualization:** Pandas, Matplotlib
- **Database:** MongoDB
- **Cloud Development:** Google Colab for ML model training
- **Deployment:** (to be added)

4.2 System Architecture

The system is composed of three major components: the Frontend, the Backend, and the Database. A block diagram of the system architecture is shown below:

Frontend (Angular) collects user inputs and sends them via API requests to the backend (FastAPI server). The backend processes the requests, loads the selected Machine Learning model from MongoDB, performs ET prediction, and sends back the results. All user inputs and model outputs are stored in the MongoDB database.

4.3 Machine Learning Models Development

The Machine Learning models were developed separately using Google Colab. The process included:

- Collecting and preprocessing 15 years of historical weather data.
- Engineering features relevant to evapotranspiration prediction.
- Training and validating several models: XGBoost, DNN, SVM, DT, RF, and SARIMAX.
- Saving the trained models for later use by the backend server.

Specifically for DNN Model

- Training & Validation Loss Over 100 Epochs

```
# Set random seed
tf.random.set_seed(42)

# Create a model (same as model_8)
model_9 = tf.keras.Sequential([
  tf.keras.layers.Dense(64, activation='relu'),
    tf.keras.layers.Dense(64, activation='relu'),
  tf.keras.layers.Dense(1)
])

# Compile the model
model_9.compile(
    loss=tf.keras.losses.mae,
    optimizer='adam',
    metrics=['mse']
)

# Create a learning rate callback
lr_scheduler = tf.keras.callbacks.LearningRateScheduler(lambda epoch:
↪  1e-4 * 10**(epoch/20))

# Fit the model (passing lr_scheduler callback)
history_9 = model_9.fit(X_train,
                        y_train,
                        epochs=100,
                        callbacks=[lr_scheduler], validation_split=0.1)

```

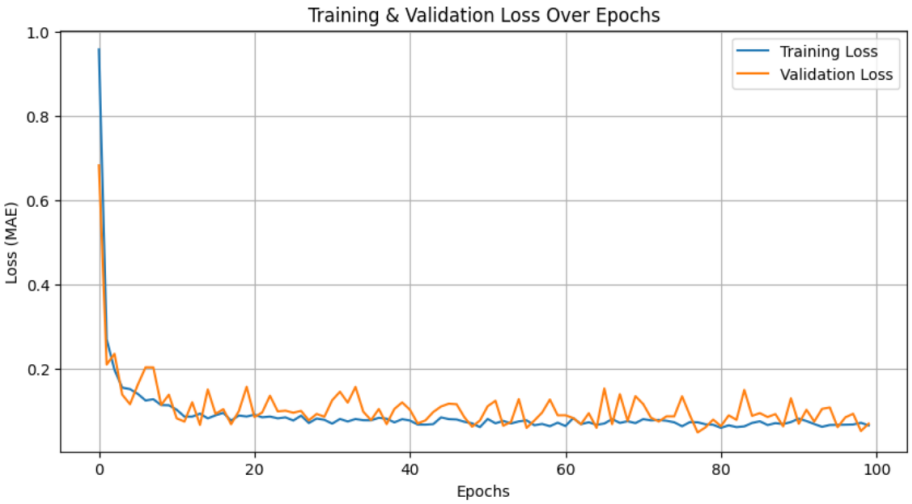


Figure 4.1: Training & Validation Loss over Epochs

- Loss Curves

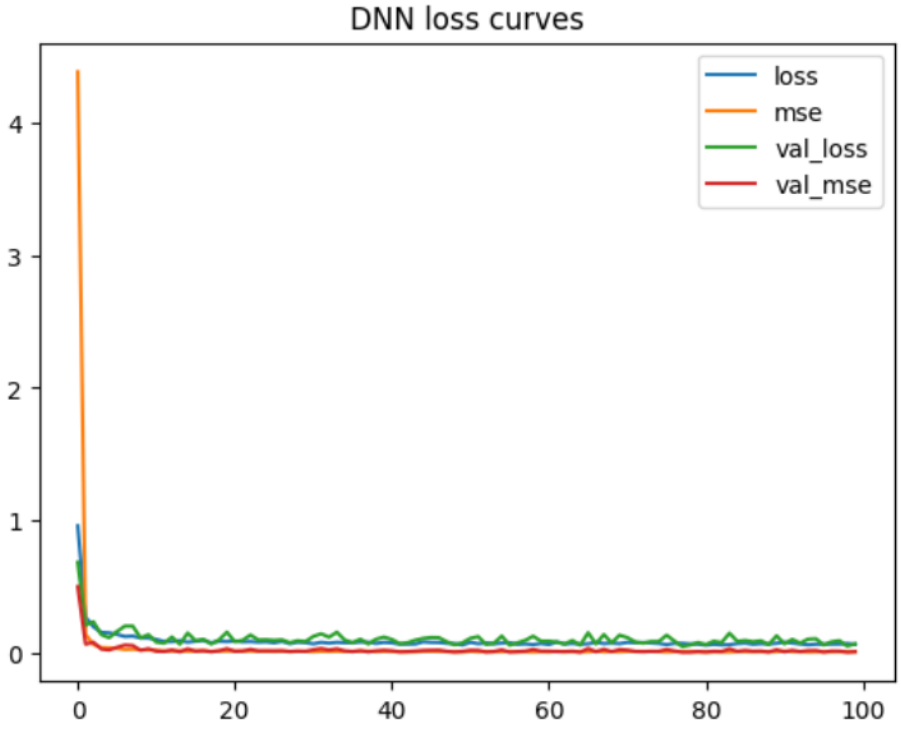


Figure 4.2: Loss Curves

- Decide for the **Best** `Learning Rate`

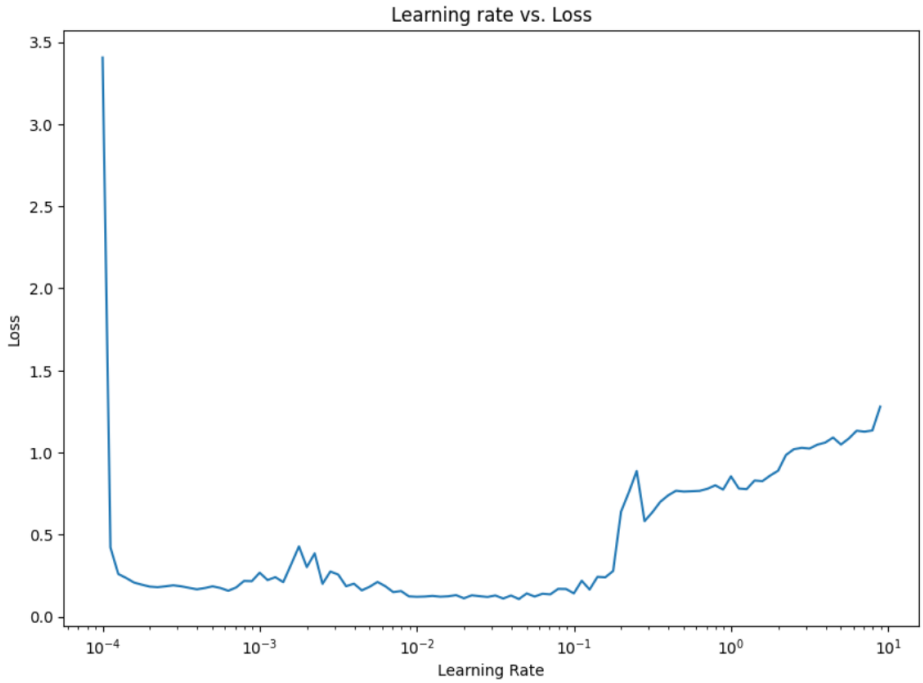


Figure 4.3: Best Learning Rate with least Loss

Each model was evaluated using RMSE, MAE, and R^2 metrics. The DNN model was selected as the most reliable based on performance.

4.4 Backend Development

The backend is built with FastAPI, designed to handle:

- Receiving user input via HTTP requests.
- Loading the corresponding Machine Learning model from MongoDB.
- Predicting the ET value based on inputs.
- Calculating the Correctness Percentage between predicted ET and actual ET.
- Sending back results to the frontend.
- Storing inputs and outputs in the MongoDB database for history tracking.

Key FastAPI endpoints include:

- `GET / getAllModels`
- `POST / load / model_id`
- `POST / predict / model_id`

```python
@app.get("/getAllModels")
def get_all_models():
    return {
        "models": [
            "dnn",
            "dt",
            "rf",
            "xgboost",
            "sarimax",
            "svm"
        ]
    }

@app.post("/load/{model_id}")
def load_model(model_id: str):
    model_id = model_id.lower()
    if model_id == "dnn":
        dnn_service.load()
        return {"model": "DNN model loaded"}
    elif model_id == "dt":
        dt_service.load()
        return {"model": "DT model loaded"}
    elif model_id == "rf":
        rf_service.load()
        return {"model": "RF model loaded"}
    elif model_id == "xgboost":
        xgboost_service.load()
        return {"model": "XGBoost model loaded"}
    elif model_id == "sarimax":
        sarimax_service.load()
        return {"model": "SARIMAX model loaded"}
    elif model_id == "svm":
        svm_service.load()
        return {"model": "SVM model loaded"}
    else:
        return {"error": f"Model '{model_id}' is not supported"}


@app.post("/predict/{model_id}")
def predict(model_id: str, data: PredictionInput):
    model_id = model_id.lower()

    if model_id == "dnn":
        x = dnn_service.predict(data)
        if(x is None):
            return {"amount": None}
        return {"amount": float(x)}
    elif model_id == "dt":
        x = dt_service.predict(data)
        return {"amount": float(x)}
    elif model_id == "rf":
```

```
52          x = rf_service.predict(data)
53
54          return {"amount": float(x)}
55      elif model_id == "sarimax":
56
57          x = sarimax_service.predict(data)
58          return {"amount": float(x)}
59      elif model_id == "svm":
60          return {"amount": svm_service.predict(data)}
61      elif model_id == "xgboost":
62          x = xgboost_service.predict(data)
63          if(x is None):
64              return {"amount": None}
65          return {"amount": float(x)}
66      elif model_id == "actual":
67          return {"amount": float(ActualResultService().compute(data))}
```

4.5 Frontend Development

The frontend is a simple web application developed using Angular. It contains:

- Input fields to collect:
  - Date
  - Location
  - Crop Type
  - Soil Type
  - Time Season (Initial, Mid-season, Late-season)
  - Area (m^2)
  - Initial Soil Moisture Level (L/m^2)
  - Maximum Soil Moisture Level (L/m^2)
- Six buttons, each corresponding to a different ML model:
  - XGBoost, DNN, SVM, DT, RF, SARIMAX
- One button to calculate the actual ET value.
- Output fields to display:
  - Predicted ET by each model
  - Correctness percentage under each model

The Angular service handles communication with the FastAPI backend through HTTP requests.

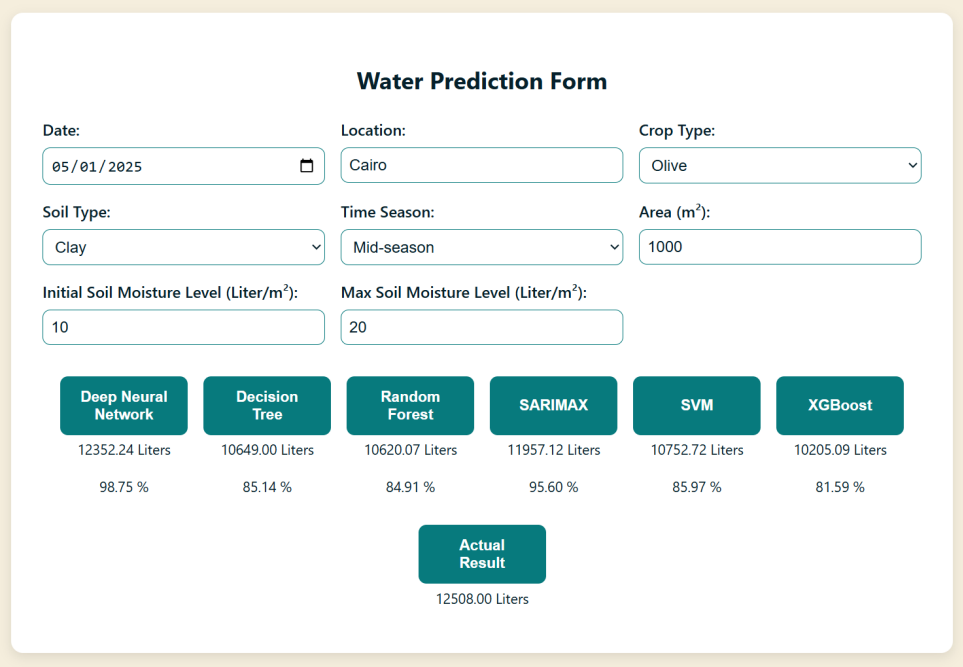


Figure 4.4: Website of the Machine Learning models

4.6 Database Management

MongoDB is used as the primary database system. It stores:

- Serialized ML models (stored as binary objects).

The database collections are structured as follows:

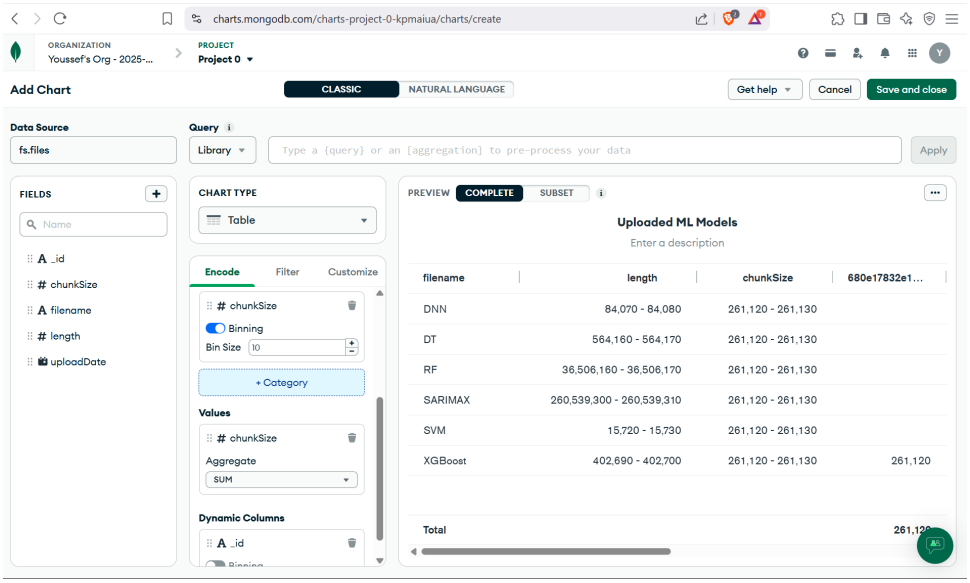


Figure 4.5: Database Structure in MongoDB

4.7 Model Integration and API Interaction

To estimate evapotranspiration (ET), the system integrates predictive machine learning models with weather data obtained via RESTful API interactions. The workflow involves both historical and forecasted weather conditions and follows a sequence of coordinated steps between the frontend, backend, and model services.

4.7.1 API Utilization for Weather Data

The system leverages the `WorldWeatherOnline` API for two distinct purposes:

- **Historical Weather Data Collection**: Using the endpoint `http://api.worldweatheronline.` the system retrieves daily weather data spanning the past 15 years. This dataset includes key features such as `avgtemp_c`, `maxtemp_c`, `mintemp_c`, `avgwind_kph`, `avghumidity`, `totalprecip_mm`, and `sunHour`. These variables are crucial for training the models to estimate ET accurately.
- **Weather Forecasting**: For real-time ET predictions, the system fetches short-term weather forecasts using the endpoint `http://api.worldweatheronline.com/premium/v1/w` This allows the models to predict ET for upcoming days based on expected conditions, enhancing the application's utility for agricultural planning.

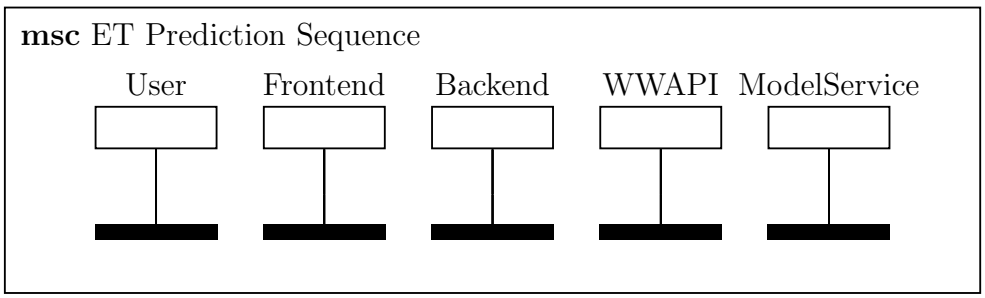
4.7.2 ET Prediction Pipeline

Once a model is selected by the user through the frontend interface, the following sequence of operations is triggered:

1. The frontend collects user inputs and sends a request to the backend.
2. The backend identifies and loads the corresponding model (e.g., DNN, XGBoost, RF) from a MongoDB database where pre-trained models are stored.
3. The model processes the input data and predicts the ET value.
4. The backend compares the prediction to the actual ET value (if available), calculating a correctness percentage using the formula:

$$\text{Accuracy } (\%) = \begin{cases} \frac{\text{Predicted}}{\text{Actual}} \times 100, & \text{if Predicted} < \text{Actual} \\ \frac{\text{Actual}}{\text{Predicted}} \times 100, & \text{otherwise} \end{cases}$$

5. The prediction and its associated accuracy are returned to the frontend for display.



4.8 Testing and Debugging

Throughout development, various tests were performed:

- Backend unit testing using `Pytest`.

```
@pytest.fixture
def dummy_model()
dummy_sarimax_model()
dummy_test_case(),
test_get_water_predicted(dummy_model, dummy_test_case)
test_get_sarimax_water_predicted(dummy_sarimax_model, dummy_test_case)
```

- API testing using `Postman`.

Chapter 5

Results

This chapter presents the experimental results obtained from the development and evaluation of the Smart Irrigation System Simulator. The simulator was designed to predict daily evapotranspiration (ET) values using historical and future weather data, and to determine the irrigation volume required based on crop, soil type, and area specifications. The results highlight the performance of various machine learning and deep learning models used for ET prediction, as well as visual representations and evaluation metrics.

5.1 Simulator Overview

The developed simulator accepts user inputs including date, location (city), crop type (Olive, Date, Orange), soil type (Loamy, Sandy, Clay), area, and moisture levels (minimum and maximum per m^2). It then performs the following steps:

1. Fetches weather conditions from the WorldWeatherOnline API using either:
   - Historical weather endpoint: `past-weather.ashx`
   - Forecast weather endpoint: `weather.ashx`
2. Predicts the ET using the selected model.
3. Calculates ETc based on crop coefficient.
4. Computes the required irrigation water using the formula:

   $$\text{Irrigation Volume} = (\text{Max Moisture Level} - (\text{Initial Moisture} + \text{Rain Contribution} - ET_c)) \times \text{Area}$$

5.2 Dataset Summary

The dataset comprises 15 years of daily weather records, including:

{`avgtemp_c`, `maxtemp_c`, `mintemp_c`, `avgwind_kph`, `avghumidity`, `totalprecip_mm`, `sunHour`}

These features were used to compute the ground-truth ET values and to train multiple models.

5.3 Model Evaluation Metrics

The following models were trained and evaluated using standard regression metrics: Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), R^2 Score, and percentage correctness. The table below summarizes the results.

Model	**MAE**	**RMSE**	**R^2 Score**	**Accuracy (%)**
XGBoost	0.058	0.080	0.9950	84.25
CNN	-	0.1401	0.9853	71.78
ANN	-	0.1951	0.9715	37.36
Decision Tree	-	0.1907	0.9727	79.65
Random Forest	-	0.1113	0.9907	88.19
Prophet	0.734	0.934	0.298	-317.09
LSTM	0.495	0.694	0.612	54.38
SVM	-	0.0801	0.9952	78.85
SARIMAX	0.255	0.364	0.893	-
DNN	0.0661	-	0.9942	-

Table 5.1: Model performance metrics on test dataset

5.4 Model Results and Graphical Analysis

For each model, a graphical representation of predicted vs actual ET values is shown. These figures visualize the performance and alignment of model predictions over time.

5.4.1 XGBoost Results

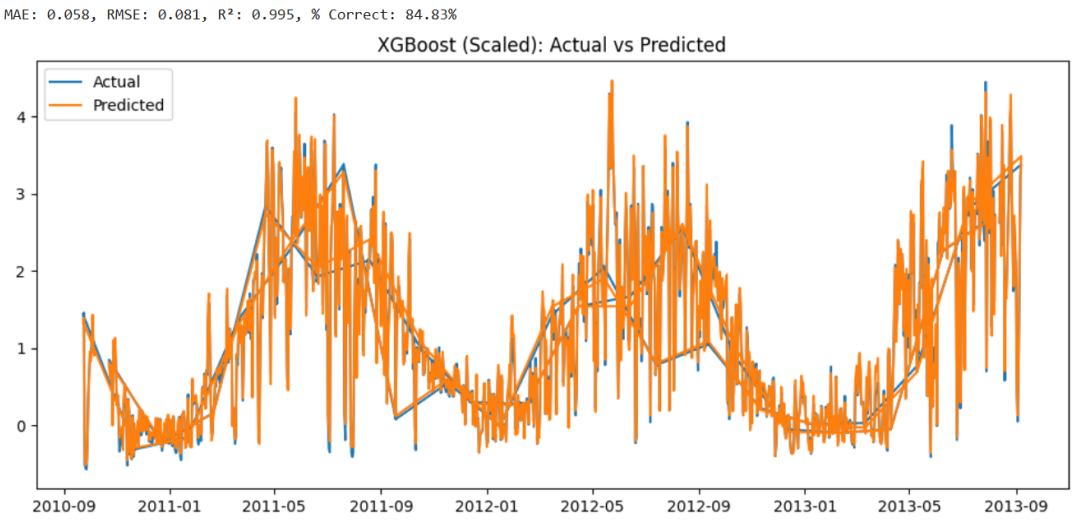


Figure 5.1: XGBoost - Predicted vs Actual ET

5.4.2 CNN Results

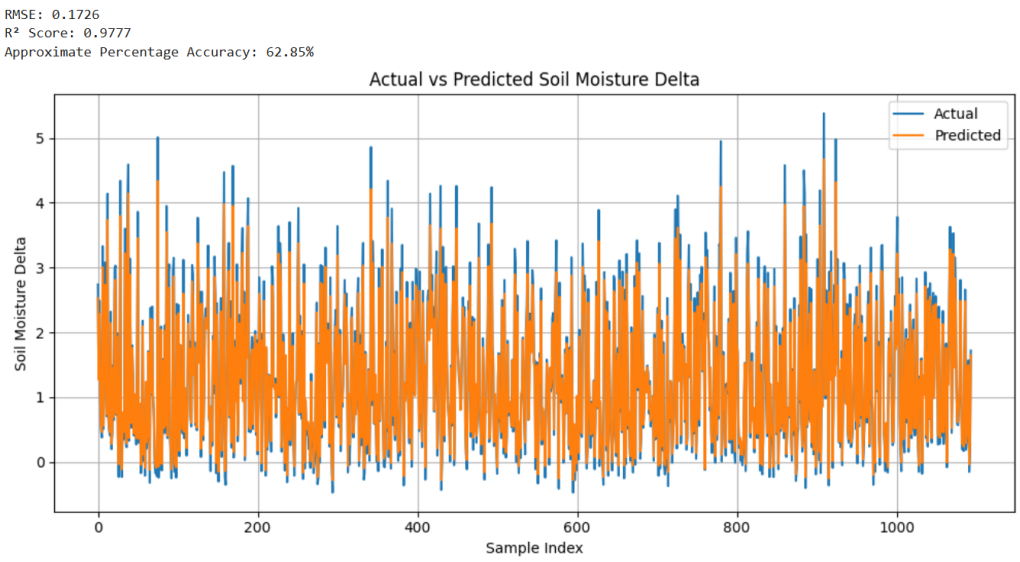


Figure 5.2: CNN - Predicted vs Actual ET

5.4.3 ANN Results

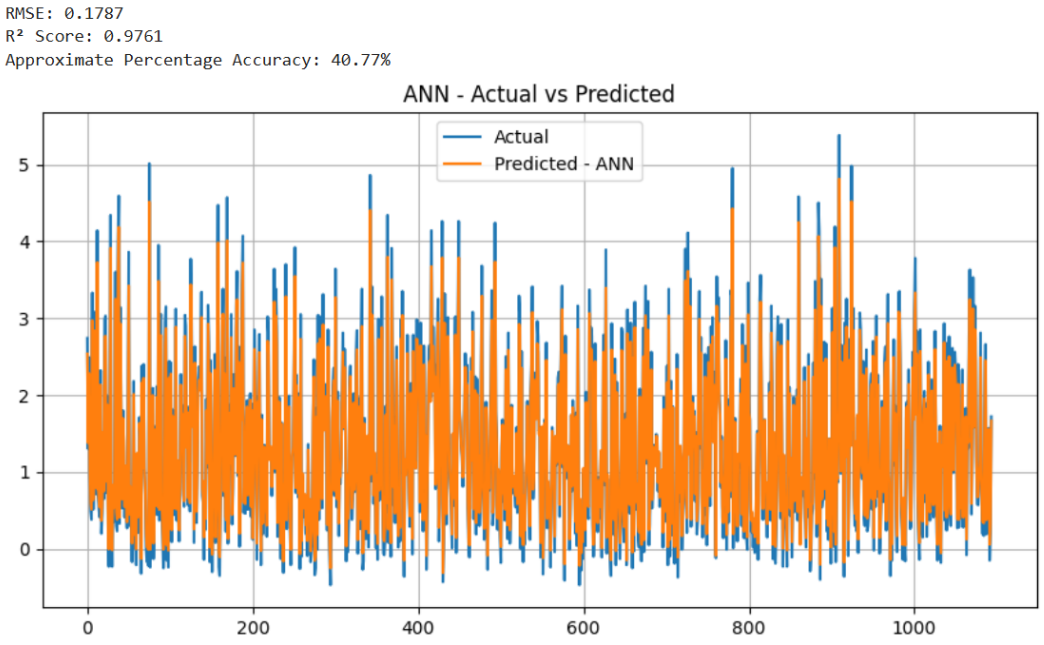


Figure 5.3: ANN - Predicted vs Actual ET

5.4.4 Decision Tree Results

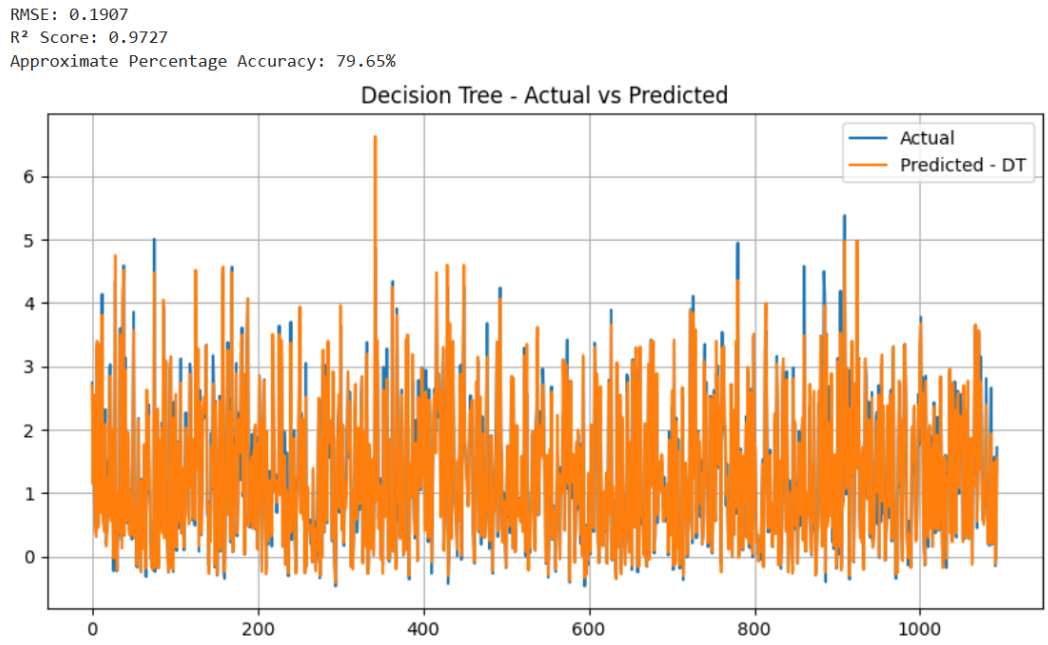


Figure 5.4: Decision Tree - Predicted vs Actual ET

5.4.5 Random Forest Results

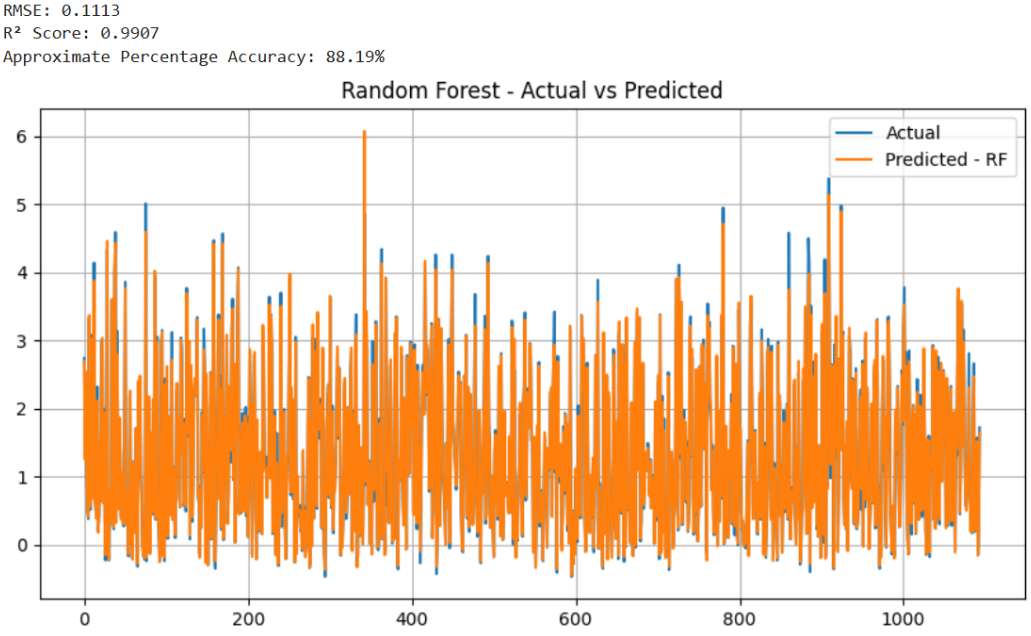


Figure 5.5: Random Forest - Predicted vs Actual ET

5.4.6 Prophet Results

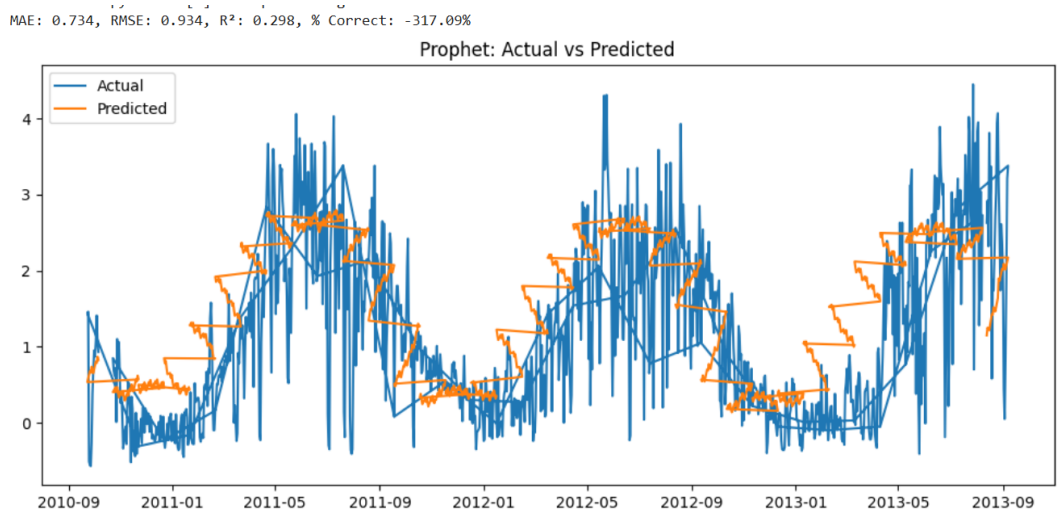


Figure 5.6: Prophet - Predicted vs Actual ET

5.4.7 LSTM Results

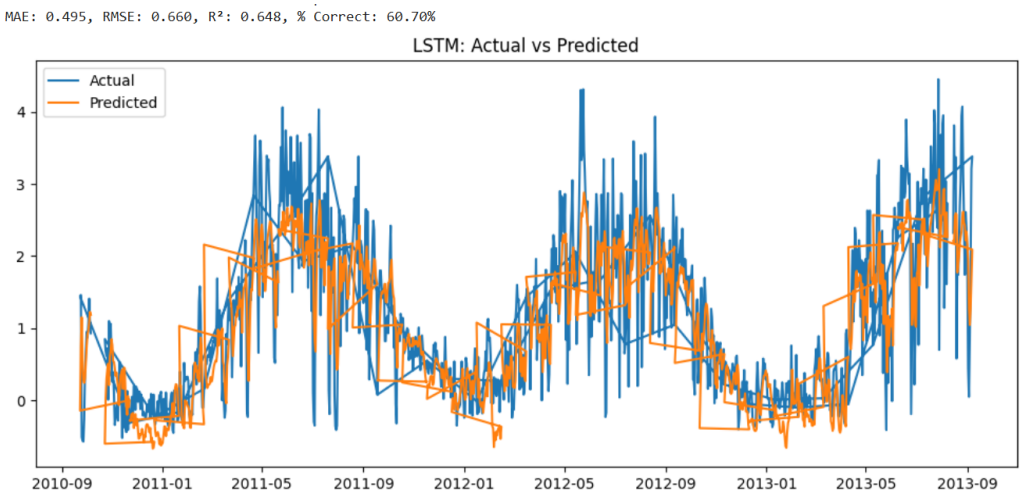


Figure 5.7: LSTM - Predicted vs Actual ET

5.4.8 SVM Results

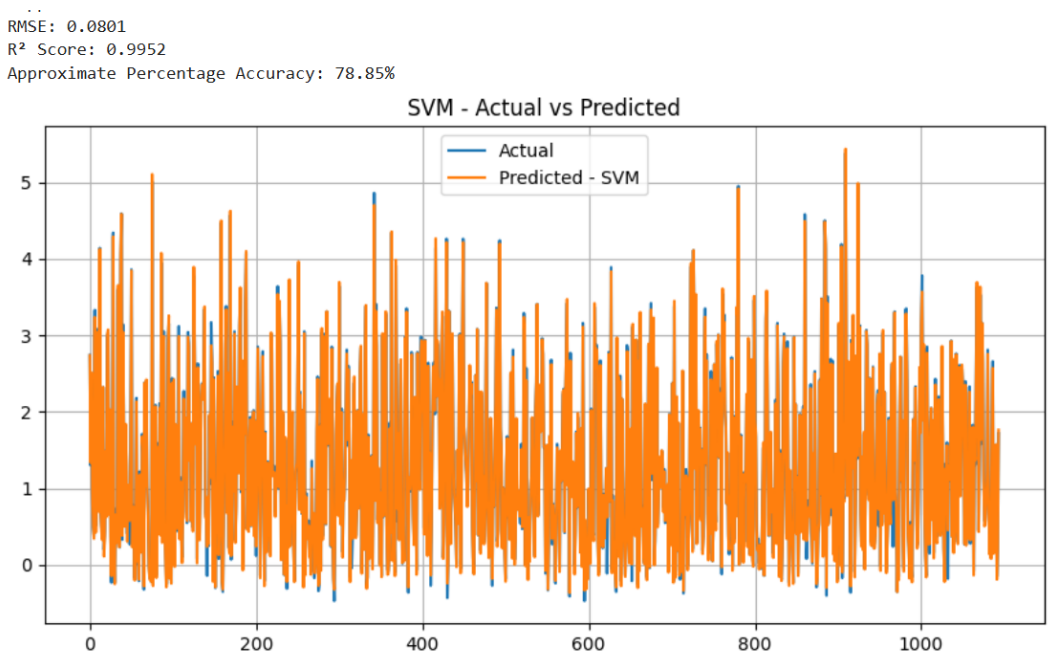


Figure 5.8: SVM - Predicted vs Actual ET

5.4.9 SARIMAX Results

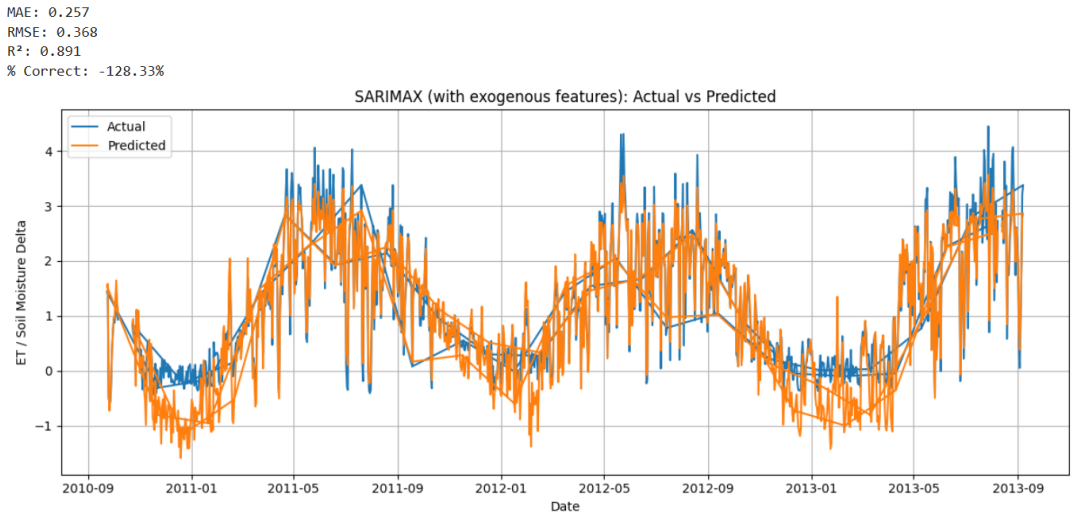


Figure 5.9: SARIMAX - Predicted vs Actual ET

5.4.10 DNN Results (Best Model)

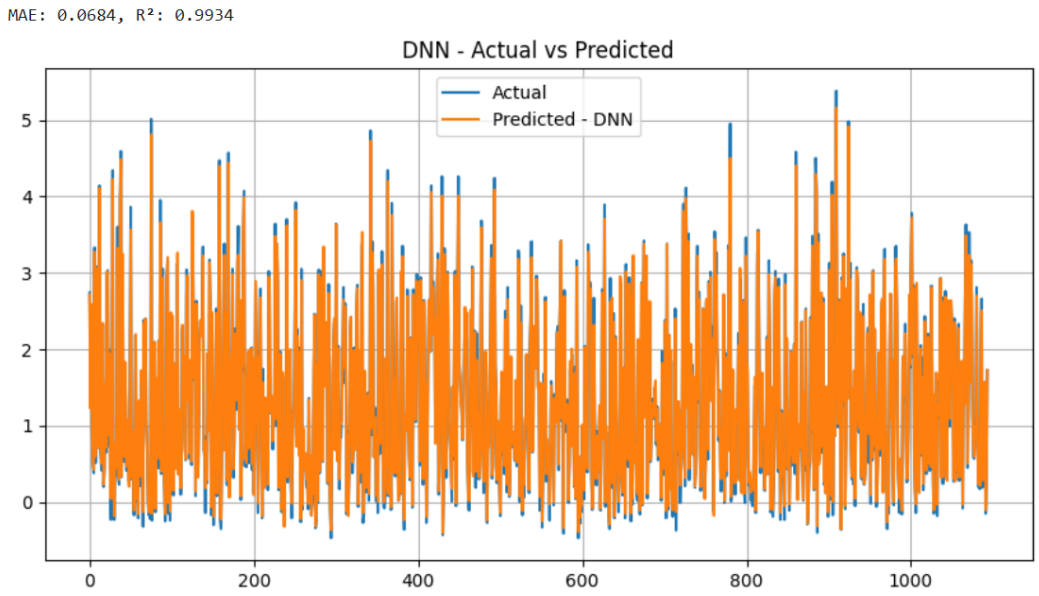


Figure 5.10: DNN - Predicted vs Actual ET

5.5 User Interface and Interaction

The simulator's user interface was developed to be intuitive and functional. Screenshots of key features are included below. Website link

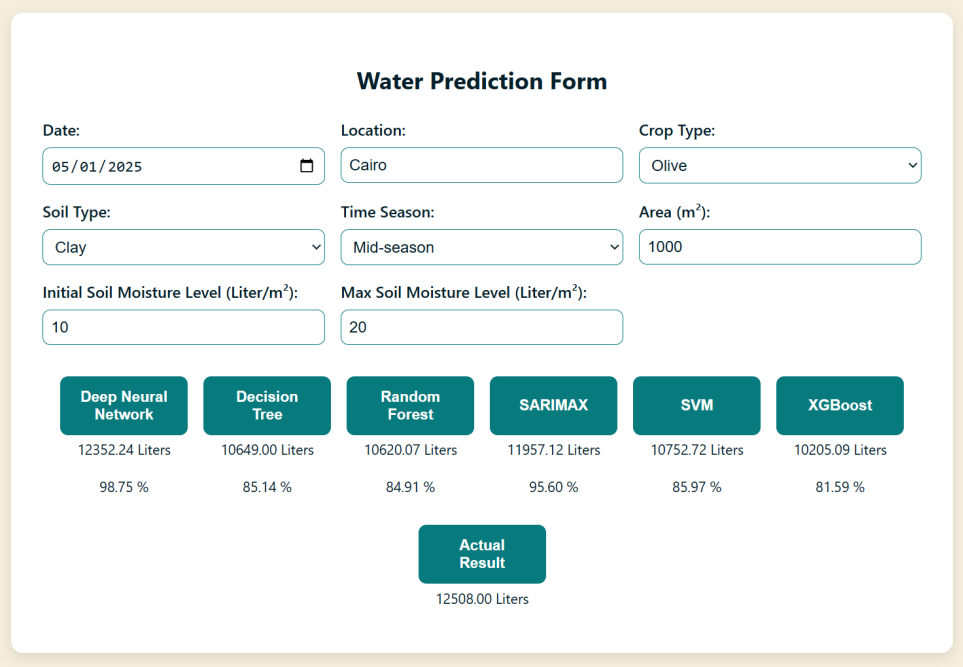


Figure 5.11: User Input Interface

5.6 Conclusion

Among all tested models, the Deep Neural Network (DNN) demonstrated the highest R^2 score and consistent accuracy, making it the optimal choice for ET prediction. This model was integrated into the live system to support real-time irrigation guidance based on future weather forecasts.

Chapter 6

Conclusion

This thesis presented the design and development of a software simulator for a smart irrigation system based on predictive modeling of evapotranspiration (ET). The simulator utilizes weather data retrieved from the WorldWeatherOnline API, covering both historical conditions over the past 15 years and forecast data for upcoming days. Key meteorological variables—such as temperature, wind speed, humidity, precipitation, and sunlight duration—were used to calculate daily ET values, forming a rich dataset for model training and evaluation.

Multiple machine learning and deep learning models were implemented and compared, including XGBoost, CNN, ANN, Decision Tree, Random Forest, PROPHET, LSTM, SVM, SARIMAX, and DNN. Each model was assessed using standard regression metrics: MAE, RMSE, R^2 score, and prediction accuracy. Among all, the Deep Neural Network (DNN) achieved superior performance with an R^2 score of 0.9942 and the lowest RMSE and MAE values, demonstrating its capability to accurately predict ET from weather features.

The selected DNN model was integrated into the system backend, enabling real-time ET prediction based on user-specified parameters such as city, date, crop type, soil type, land area, and moisture thresholds. The simulator calculates crop evapotranspiration (ETc) using appropriate crop coefficients and estimates the irrigation volume required. The system also provides users with comparative visualizations and accuracy charts for each model, enhancing transparency and user trust.

In summary, the simulator successfully combines environmental data, machine learning techniques, and domain-specific irrigation logic to offer an intelligent, scalable solution for water-efficient agriculture. It lays the groundwork for future enhancements and potential deployment in real-world agricultural decision-making environments.

Chapter 7

Future Work

While the current implementation of the smart irrigation simulator has demonstrated promising results in ET prediction and irrigation planning, there are several areas where the system can be expanded or improved:

- **Real-time Sensor Integration:** Incorporating live data from on-field sensors (e.g., soil moisture, temperature, humidity) would enhance prediction accuracy and allow for more dynamic, automated control of irrigation systems.
- **Model Optimization and Ensemble Learning:** Combining predictions from multiple models through ensemble techniques may further improve reliability, particularly under abnormal weather conditions or data sparsity.
- **Crop-Specific Enhancements:** Extending support for a broader variety of crops with distinct evapotranspiration coefficients and growing periods would make the system more versatile.
- **Localized Calibration:** Adapting the system to regional agricultural practices and soil profiles through localized datasets and fine-tuning would improve its applicability across different geographic zones.
- **User Interface and Experience:** Enhancing the frontend with GIS-based field mapping and personalized dashboards could improve usability, especially for farmers or agronomists with limited technical background.
- **Irrigation Scheduling and Control:** Integrating the simulator with smart irrigation hardware (e.g., IoT-enabled valves and pumps) could support end-to-end automation from prediction to water delivery.
- **API Optimization and Data Sources:** Using additional weather APIs or switching to more flexible, open-source weather data sources may reduce dependency on quota-limited services and expand data coverage.

These future directions aim to improve the simulator's accuracy, scalability, and real-world usability, transforming it into a comprehensive decision-support tool for smart agriculture.

Appendix

Appendix A

Lists

ET	Evapotranspiration
AI	Artificial Intelligence
ML	Machine Learning
CNN	Convolutional Neural Network
ANN	Artificial Neural Network
RF	Random Forest
DT	Decision Tree
SVM	Support Vector Machine
LSTM	Long Short-Term Memory
RMSE	Root Mean Square Error
XGBoost	Extreme Gradient Boosting
API	Application Programming Interface
IoT	Internet of Things
Kc	Crop Coefficient
ETc	Crop Evapotranspiration
ETo	Reference Evapotranspiration
DNN	Deep Neural Network
RMSE	Root Mean Square Error
MAE	Mean Absolute Error
R^2	Coefficient of Determination
SARIMAX	Seasonal AutoRegressive Integrated Moving Average with eXogenous regressors)

List of Figures

Bibliography

[1] C. Ingrao, R. Strippoli, G. Lagioia, and D. Huisingh. Life cycle thinking and smart irrigation: A critical review toward sustainable agriculture. *Heliyon*, 9:e18507, 2023. doi:10.1016/j.heliyon.2023.e18507.

[2] A. A. Ahmed, S. Sayed, A. Abdoulhalik, S. Moutari, and L. Oyedele. A comprehensive review of smart irrigation systems and technologies for sustainable agriculture. *Journal of Cleaner Production*, 441:140715, 2024. doi:10.1016/j.jclepro.2024.140715.

[3] Ritesh Singh, Parvinder Kaur, Rajneesh Singh, and Gurpreet Sethi. Iot and machine learning enabled smart irrigation system. *International Journal of Engineering and Advanced Technology (IJEAT)*, 10(3):44–49, 2021. URL: https://www.researchgate.net/publication/349042572_IoT_and_Machine_Learning_enabled_Smart_Irrigation_System.

[4] D. K. Singh, R. Sobti, A. Jain, P. K. Malik, and D. Le. A survey of communication technologies for smart agriculture: Insights and future directions. *IET Communications*, 16:604, 2022. doi:10.1049/cmu2.12352.

[5] Ritesh Singh, Parvinder Kaur, Rajneesh Singh, and Gurpreet Sethi. Evaluating the machine learning based efficacy of decision tree and support vector machines in smart irrigation systems for precise irrigation status classification for optimizing water management. *ResearchGate*, 2024. URL: https://www.researchgate.net/publication/379857409_Evaluating_the_machine_learning_based_Efficacy_of_Decision_Tree_and_Support_Vector_Machines_in_Smart_Irrigation_Systems_for_Precise_Irrigation_Status_Classification_for_Optimizing_Water_Management_in_.

[6] Ritesh Singh, Parvinder Kaur, Rajneesh Singh, and Gurpreet Sethi. A review of evapotranspiration estimation methods and its application in water resource management. *Journal of Water and Climate Change*, 16(2):249–268, 2024. URL: https://iwaponline.com/jwcc/article/16/2/249/106353/A-review-of-evapotranspiration-estimation-methods, doi:10.2166/wcc.2023.302.

[7] Weiyu Li and Daniel M. Tartakovsky. Fast and accurate estimation of evapotranspiration for smart agriculture. *Water Resources Research*, 59(4):e2023WR034535, 2025. URL: https://agupubs.onlinelibrary.wiley.com/doi/10.1029/2023WR034535, doi:10.1029/2023WR034535.

[8] Y. Lu, X. Zhang, Y. Wang, and J. Li. Irrigation scheduling based on soil moisture prediction using a hybrid machine learning model. *Agricultural Water Management*, 250:106835, 2021. URL: https://www.sciencedirect.com/science/article/pii/S0378377421006016, doi:10.1016/j.agwat.2021.106835.

[9] Weiyu Li and Daniel M. Tartakovsky. Fast and accurate estimation of evapotranspiration for smart agriculture. *Water Resources Research*, 59(4):e2023WR034535, 2025. URL: https://agupubs.onlinelibrary.wiley.com/doi/10.1029/2023WR034535, doi:10.1029/2023WR034535.

[10] Weiyu Li and Daniel M. Tartakovsky. Fast and accurate estimation of evapotranspiration for smart agriculture. *Water Resources Research*, 59(4):e2023WR034535, 2025. URL: https://agupubs.onlinelibrary.wiley.com/doi/10.1029/2023WR034535, doi:10.1029/2023WR034535.