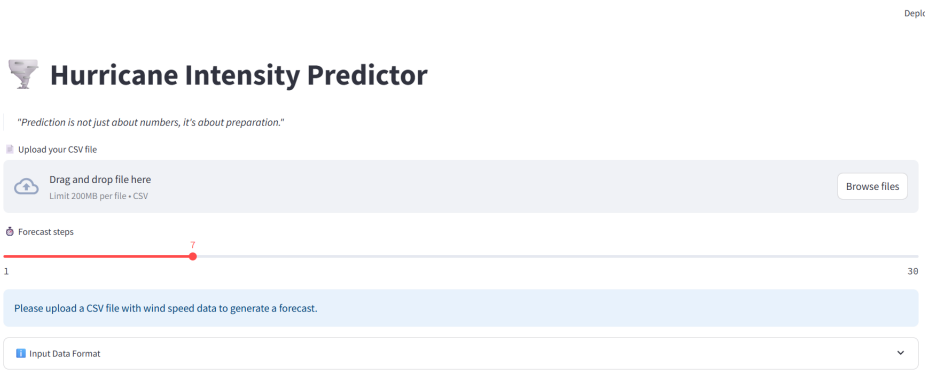
University of Applied Sciences
HOCHSCHULE EMDEN·LEER

Manual

Hurricane Intensity Prediction Using Time Series Algorithms



Autor:	Kreetika Mohanta
Matriculation-Nr.:	7026001
Autor:	Daanyaal Parvaize
Matriculation-Nr.:	7026795
Autor:	Keerti Belmane
Matriculation-Nr.:	7026791
Course of study:	Master's in Business Intelligence and Data Analytics
First Examiner:	Prof. Dr. Elmar Wings
Submission Date:	June 29, 2025

Hochschule Emden/Leer
Fachbereich Technik · Abteilung Maschinenbau
Constantiaplatz 4 · 26723 Emden
http://www.hs-emden-leer.de

Contents

List of Figures

List of Tables

Acronyms

NOAA National Oceanic and Atmospheric Administration

LSTM Long Short-Term Memory

ARIMA AutoRegressive Integrated Moving Average

ML Machine Learning

AI Artificial Intelligence

1 Introduction

1.1 Project Overview

This project uses the NOAA Atlantic Hurricane Dataset as the main source of training data to build predictive models for hurricane intensity forecasting. Leveraging time series modeling techniques such as ARIMA and LSTM, the project aims to provide accurate forecasts based on historical hurricane data. The implementation is designed for ease of use and can be run entirely on your local machine, with Streamlit serving as the cloud-based interface for visualization and interaction.

1.2 Dataset Specification

- **Name**: NOAA Atlantic Hurricane Dataset (HURDAT2)
- **Size**: Approximately 1.4 MB (varies with updates; covers 1851–present)
- **Format**: Comma-delimited text file (CSV-like structure)
- **Content**:
  - Storm metadata (name, year, identifier)
  - Six-hourly records of each storm, including:
    - Date and UTC time
    - Storm status (e.g., Tropical Depression, Storm, Hurricane, etc.)
    - Latitude and longitude
    - Maximum sustained wind speed (knots)
    - Minimum central pressure (millibars)
    - Wind radii for 34, 50, and 64 knot winds in each quadrant
    - Radius of maximum wind (recent years)
    - Special markers for landfall and intensity peaks
- **Temporal Resolution**: Six-hourly intervals (00, 06, 12, 18 UTC), with additional records for landfall or intensity changes
- **Coverage**: North Atlantic basin, from 1851 to present (updated annually)
- **Source**: Maintained by the National Hurricane Center (NHC), NOAA

- **Note**: This dataset is used exclusively for "ONLY" training the prediction models.

1.3 Model Specification

- **ARIMA (AutoRegressive Integrated Moving Average)**:
  - Suitable for univariate time series forecasting.
  - Handles seasonality and trends.
- **LSTM (Long Short-Term Memory Networks)**:
  - A type of recurrent neural network (RNN) for sequence prediction problems.
  - Effective for capturing long-term dependencies in time series data.

1.4 Software and Tools

- **Local Machine**: Primary environment for running the project.
- **Python Version**: 3.9 (specifically required for running TensorFlow)
- **Libraries Used**:
  - `pandas` for efficient data manipulation and handling of tabular data.
  - `numpy` for high-performance numerical computations and array operations.
  - `matplotlib` for static data visualization and plotting.
  - `scikit-learn` for data preprocessing, feature scaling, and evaluation of machine learning models.
  - `tensorflow` for building, training, and evaluating LSTM-based deep learning models.
  - `keras` as a high-level API for simplifying deep learning model development with TensorFlow backend.
  - `statsmodels` (v0.13.5) for time series modeling using ARIMA and statistical analysis.
  - `streamlit` for developing an interactive and user-friendly web-based interface.
  - `plotly` for interactive and dynamic visualizations.
  - `joblib` for model serialization and efficient object persistence.
  - `requests` for making HTTP requests, useful for external data integration.

1.5 Hardware and System Requirements

This project is lightweight and runs smoothly on basic modern hardware. Here's what you need to set it up and run it comfortably—even at home or school.

- **Computer or Laptop:** Any standard PC or laptop, preferably less than 7 years old.
- **Mouse and Keyboard:** For navigating and typing.
- **Monitor/Screen:** Minimum resolution 1024×768.
- **Processor:** At least a dual-core CPU (Intel or AMD).
- **RAM:** Minimum 4 GB (8 GB recommended).
- **Disk Space:** Approximately 20–100 MB needed for dataset, project files, and temporary files.

- **Operating System:** Windows 10/11, macOS, or Linux.
- **Internet Access:** Required for downloading dependencies and running the Streamlit app (e.g., connect to a WiFi like `HomeWiFi123`).
- **Browser:** Chrome, Firefox, Edge, or Safari to view results.
- **Python:** Version 3.7 or above. Python 3.8 or 3.9 is ideal for compatibility.
- **Development Tools:** Jupyter Notebook or JupyterLab, and Streamlit installed.
- **Optional – GPU:** Helps speed up model training, but not mandatory.

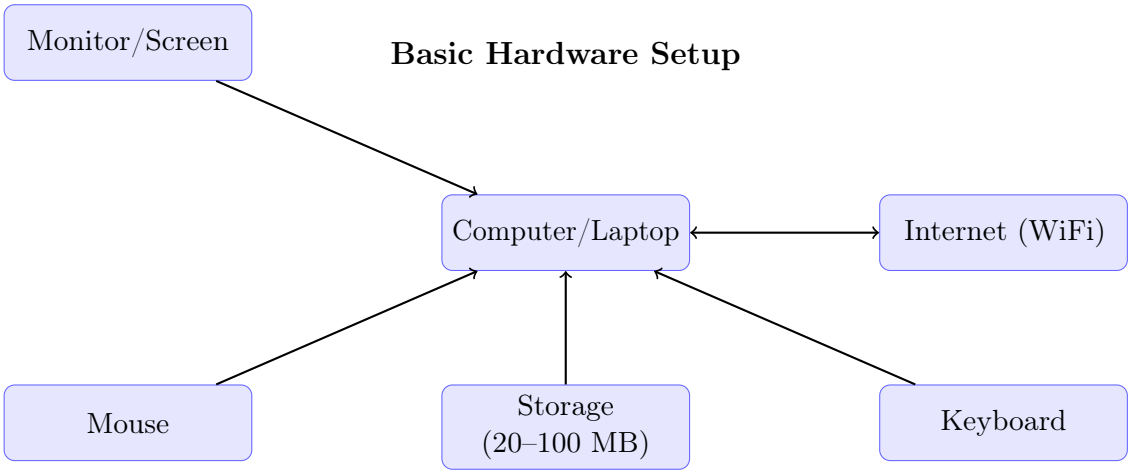


Figure 1.1: Hardware setup needed for running the hurricane prediction project.

2 Installation

2.1 System Requirements

This project is built to be run easily on local computers. It doesn't need advanced hardware or cloud services. If your computer is from the last 5–7 years, it's most likely ready!

- **Computer or Laptop:** Any modern machine (Windows, macOS, or Linux).
- **Mouse and Keyboard:** Required for user interaction and navigation.
- **Monitor/Screen:** At least 1024×768 resolution.
- **Processor:** Dual-core or better (Intel/AMD).
- **RAM:** 4 GB minimum (8 GB+ recommended).
- **Disk Space:** 20–100 MB for project files and data.
- **Internet Access:** Required for installing packages and running the Streamlit app.
- **Python Version:** Python 3.7 or higher (prefer 3.8 or 3.9).
- **Operating System:** Windows 10/11, macOS, Linux.
- **Development Tools:** Jupyter Notebook, JupyterLab, or Streamlit.
- **Web Browser:** Chrome, Firefox, Edge, Safari.
- **Optional – GPU Support:** Accelerates LSTM training but not required.

2.2 Project Specifications

- **Training Dataset:** NOAA Atlantic Hurricane Dataset (about 1.36 MB).
- **User Data:** You can upload your own CSV data with hurricane wind speeds.
- **Forecast Models:** ARIMA and LSTM (already trained and ready to use).
- **Programming Language:** Python 3.x (preferably 3.8 or 3.9).

- **Storage:** Very little local space needed. Most work happens in memory.
- **User Interface:** Easy-to-use app using Streamlit.
- **Outputs:** Beautiful graphs and simple tables showing forecast results.
- **Compatibility Note:** TensorFlow doesn't work well with Python 3.10 or newer. Stick to 3.8 or 3.9!

3 First Steps

Once your environment is set up, follow these steps to start using the dataset and models:

3.1 Setting Up the Project Locally

Download or clone the project repository to your local machine.

1. If using a GitHub repository, open a terminal, navigate to your desired directory, and run:

   ```
   git clone https://github.com/your-repository-link.git
   ```

2. If you do not use Git, you can manually download the project files as a ZIP archive from GitHub and extract them to your preferred folder.

3.2 Installing Dependencies

Before running the application, ensure you have Python (version 3.9 or higher recommended) and pip installed. Then, install the required libraries:

```
pip install -r requirements.txt
```

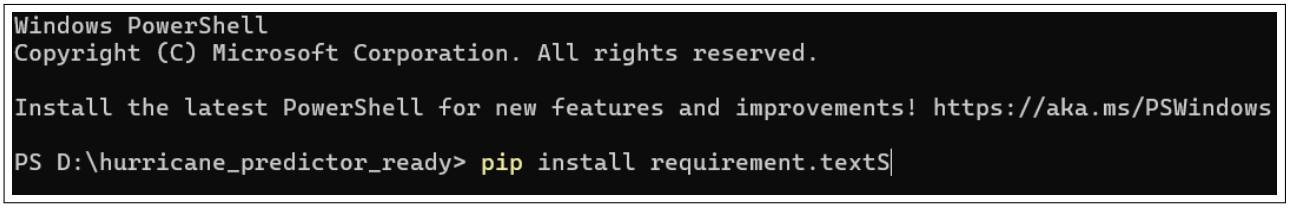


Figure 3.1: Installing the Required Python libraries

3.3 Running the Application

To launch the application, use the following command in your terminal from the project directory:

```
streamlit run app.py
```

This will open the application in your default web browser, where you can upload your data and generate predictions.

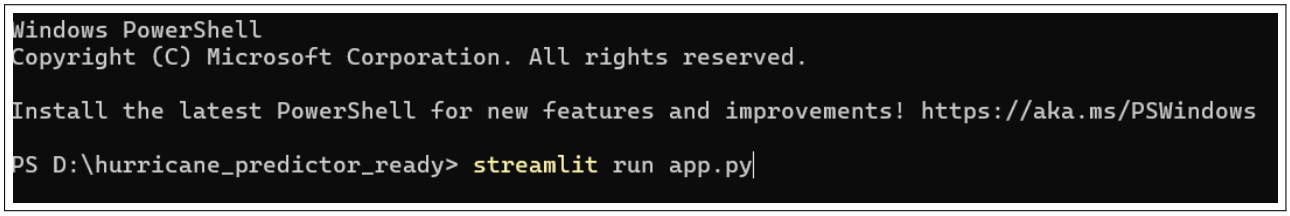


Figure 3.2: Streamlit Command

3.4 Running the Preprocessing and Prediction Models

Once you have installed the necessary libraries and set up the project, you can use the application to generate hurricane intensity predictions with the pre-trained ARIMA and LSTM models. Follow these steps:

Step 1: Launch the Application

Open a terminal, navigate to the project directory, and run the following command to start the Streamlit app:

```
streamlit run app.py
```

This will open the application in your default web browser.

Step 2: Upload Your Data

On the web interface, you will see an option to upload your hurricane dataset.

Ensure your data is formatted according to the required structure (date, time, location, wind speed, pressure, etc.).

Upload your file using the provided upload button.

Figure 3.3: Uploading the File

Step 3: Generate Predictions

Once your data is uploaded, the application will automatically use the pre-trained ARIMA and LSTM models to generate predictions.

The models have been pre-trained by the developer using optimal parameters and are stored in the `models/` folder. You do not need to set or adjust any parameters.

Step 4: View and Download Results

The predicted results will be displayed on the web interface.

You will also have the option to download the predictions as a CSV file for further analysis.

Notes

No Training Required: You do not need to train the models or set any parameters. All model training and parameter selection have been handled by the developer.

Pre-trained Models: The application uses models that have already been trained and saved. Your uploaded data is only used for generating predictions.

Model Files: The ARIMA and LSTM model files are stored in the `models/` directory and are loaded automatically by the application.

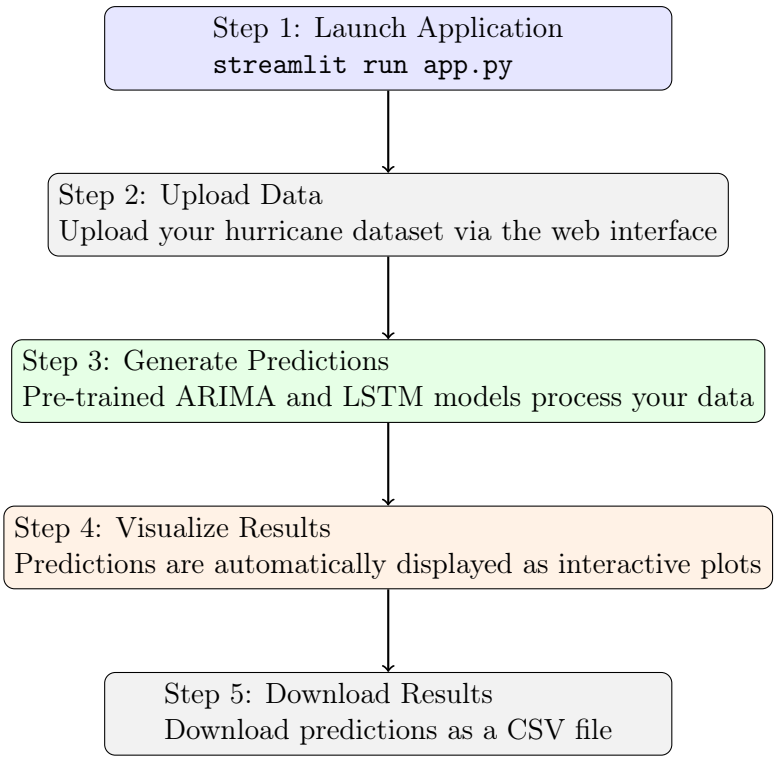


Figure 3.4: Workflow for Running Preprocessing, Prediction, and Visualization

3.5 Visualizing the Predictions

After you upload your data and predictions are generated, the results are automatically visualized within the Streamlit application. You will see interactive plots for both ARIMA and LSTM model predictions directly on the web interface.

No additional steps or code are required to display these visualizations. The app handles all plotting internally using Matplotlib and Streamlit's `st.pyplot()` function.

If you wish to save or further analyze the plots, you can use the download options provided in the app or take screenshots as needed.

4 Setup

This section guides you through preparing your environment and project files before running the Hurricane Intensity Prediction System. Please follow each step carefully.

Project Folder Structure

Your main project folder should be named `hurricane_predictor_ready` and contain the following files and folders:

```
hurricane_predictor_ready/
|
+-- app.py
+-- developer.py
+-- models_train.py
+-- utils.py
+-- requirements.txt
+-- models/         # Folder for storing trained model files
+-- data/           # Folder for datasets
```

Initial Steps

1. **Clone or Download the Repository:**
   Download the project from the official source or clone it using:

   ```
   git clone https://github.com/yourusername/hurricane_predictor_ready.git
   ```

2. **Create and Activate a Virtual Environment:**
   Open your terminal or command prompt, navigate to the project folder, and run:

   ```
   python3 -m venv venv
   ```

   Activate the environment:

   - On Linux/Mac:

     ```
     source venv/bin/activate
     ```

- On Windows:

  ```
  venv\Scripts\activate
  ```

3. **Install Required Libraries:**
   With the virtual environment activated, install dependencies:

   ```
   pip install -r requirements.txt
   ```

4. **Check Model Files:**
   Ensure the `models/` folder contains the pre-trained ARIMA and LSTM model files. If not, contact the developer or use `developer.py` to train and save the models.

5. **Prepare Your Data:**
   Place your hurricane dataset (CSV format) in the `data/` folder, or have it ready for upload via the app interface. The file should include at least `datetime` and `wind_speed` columns.

6. **Test the Setup:**
   Run the following command:

   ```
   streamlit run app.py
   ```

   The application should launch in your web browser without errors.

Setup Workflow Diagram

Tip: If you encounter any issues during setup, refer to the **Troubleshooting** chapter for solutions.

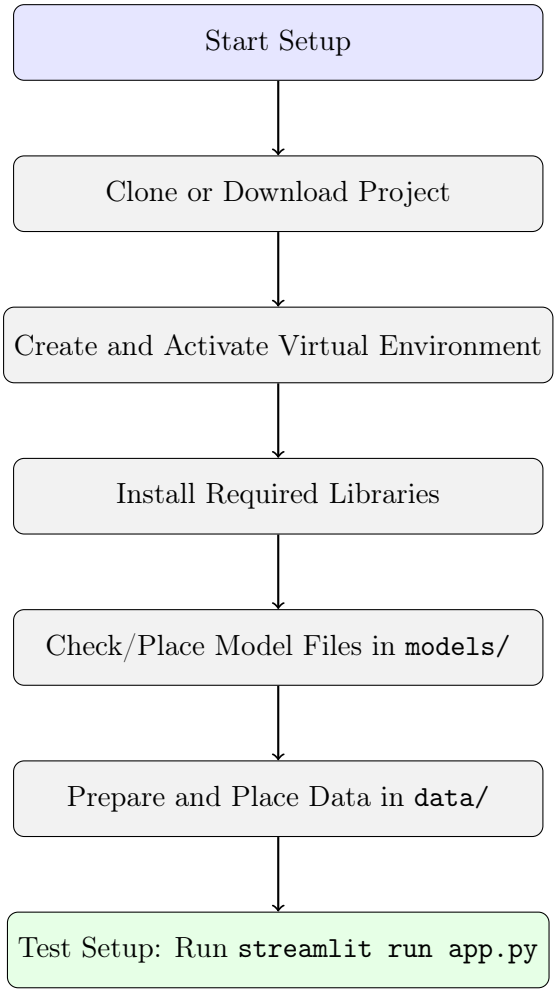


Figure 4.1: Setup Workflow for Hurricane Intensity Prediction System

5 Main Function

In this section, we describe the key functionalities and workflow of the Hurricane Intensity Prediction System from both the developer and user perspectives.

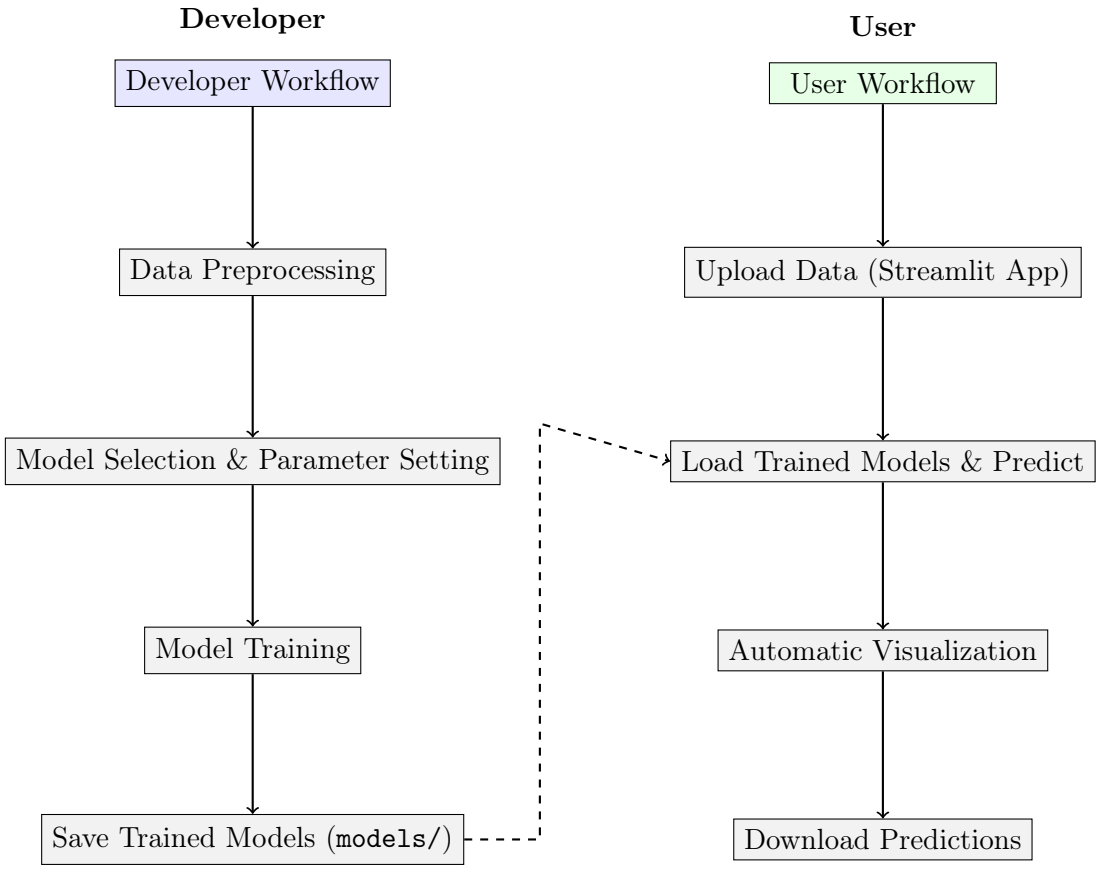
The main functions of the system are summarized in Table 5.1.

Functionality	**Description**
Data Preprocessing	Load and clean the NOAA Atlantic Hurricane Dataset or user-provided data to ensure compatibility with the models.
Model Selection	Developer selects and configures ARIMA and LSTM models for hurricane intensity prediction.
Training (Developer)	Models are trained on historical hurricane data using optimal parameters set by the developer. Trained models are saved in the `models/` directory.
Prediction (User)	User uploads new hurricane data via the Streamlit app. The system loads the pretrained models and generates predictions automatically.
Visualization	The application automatically visualizes prediction results for the user within the web interface.
Evaluation (Developer)	Developer can compare predicted values with actual values to assess and improve model performance.

Table 5.1: Main Functions of the Hurricane Intensity Prediction System

Workflow Overview

The following diagram illustrates the workflow for both developers and users:



Legend:
Developer: Responsible for data preprocessing, model selection, training, and saving models.
User: Interacts with the application via Streamlit, uploads data, and receives predictions and visualizations using pre-trained models.

6 Functions

This chapter explains what each main file and component in the Hurricane Intensity Prediction System does. Understanding the role of each file will help you know how the system works and how to use it effectively.

File Responsibilities

File/Folder	Description
`app.py`	The main user interface. Launches the Streamlit web app, lets users upload their data, loads pre-trained models, generates predictions, and visualizes the results.
`developer.py`	Used by the developer to set model parameters, select the best model (ARIMA or LSTM), and initiate model training and saving.
`models_train.py`	Contains the logic for training ARIMA and LSTM models using the NOAA dataset and developer-specified parameters. Called by `developer.py`.
`util.py`	Provides utility functions for data preprocessing, saving/loading models, and managing files.
`models/`	Directory where all trained model files (ARIMA/LSTM) are stored for later use by the application.
`requirements.txt`	Lists all Python dependencies needed to run the project.
`data/`	Directory for storing datasets used for training or prediction.

Table 6.1: Main Files and Their Functions

Functional Workflow

The figure below illustrates the flow of data and control between the main components of the system, from both the developer and user perspectives.
Legend:
Developer: Prepares and trains models, which are saved for user predictions.
User: Uploads data, receives predictions and visualizations using pre-trained models.

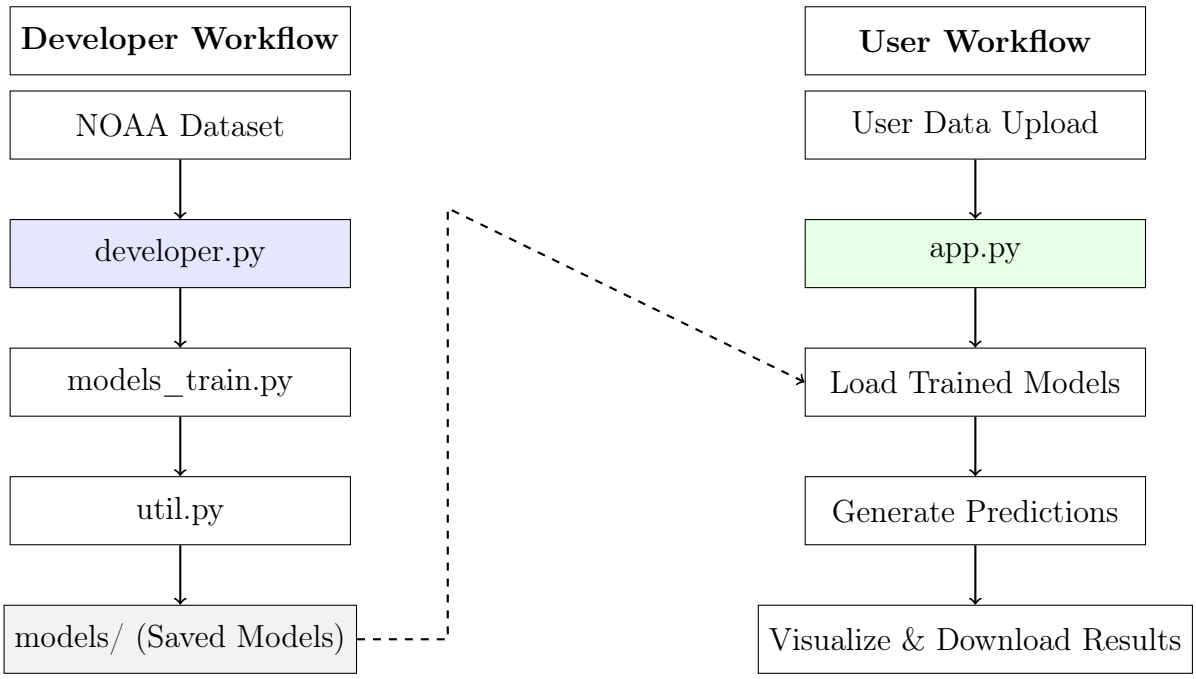


Figure 6.1: Functionality Overview and File Interactions

User Interface Sketch

The figure below provides a schematic overview of the main elements in the user interface of the Hurricane Intensity Prediction App. This helps users understand what to expect when running the application.

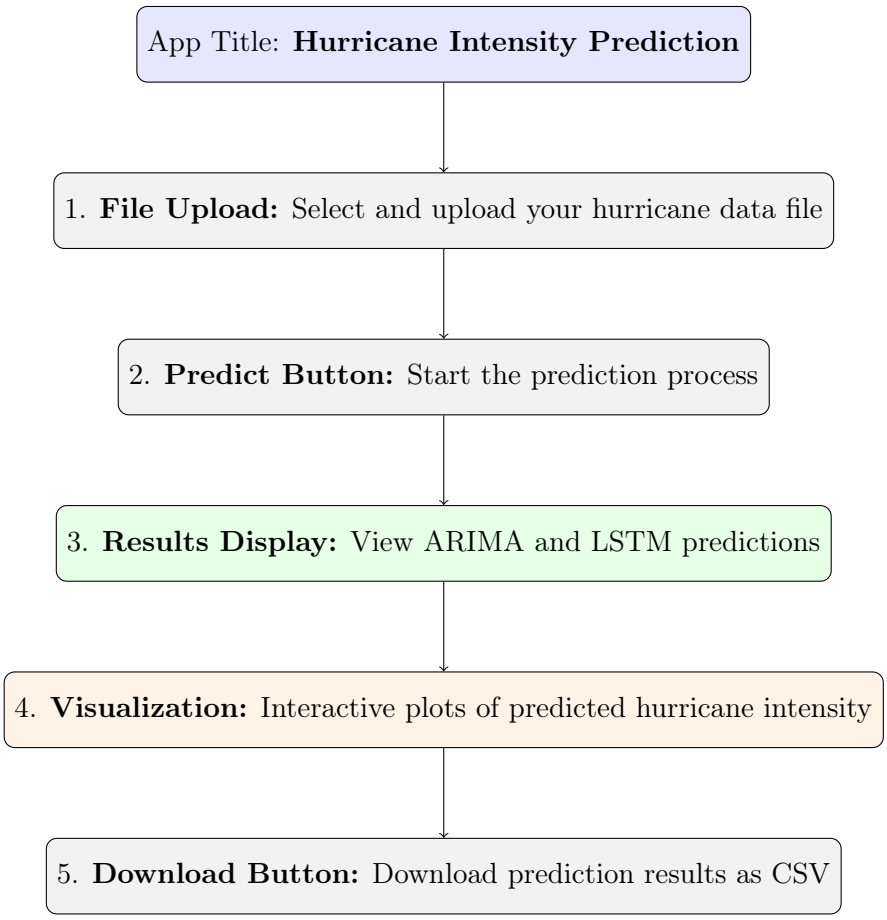


Figure 6.2: Sketch of the User Interface for the Hurricane Intensity Prediction App

7 Safety Guidelines

When using the Hurricane Intensity Prediction System, please follow these safety guidelines to ensure responsible and secure use of the application and its predictions:

- **Validate Your Data:** Always check that your uploaded hurricane data is accurate, complete, and properly formatted before using it for predictions. Poor data quality can lead to unreliable results and misleading forecasts.
- **Understand Model Limitations:** The predictions provided by this system are based on statistical and machine learning models trained on historical data. These models have inherent limitations and uncertainties. Do not rely solely on these predictions for making critical decisions, especially those affecting safety, property, or emergency planning.
- **Monitor Model Performance:** Regularly review prediction outputs for unexpected results or errors. If you notice significant discrepancies or unusual forecasts, consult additional trusted sources or contact the system administrator.
- **Use Official Forecasts for Safety Decisions:** For real-world hurricane preparedness, always refer to official sources such as the National Hurricane Center (NHC), NOAA, or local authorities for the latest warnings, advisories, and emergency instructions. This system is intended as a supplementary tool for research, analysis, and educational purposes.
- **Protect Sensitive Data:** Ensure that any datasets you use or generate, as well as trained model files, are stored securely and are not accessible to unauthorized individuals. Follow your organization's data privacy and security policies.
- **Stay Informed About Hurricane Risks:** Keep up to date with the latest hurricane forecasts and preparedness guidelines from trusted agencies. Enable emergency alerts on your devices and have a plan for responding to severe weather events.
- **Do Not Overinterpret Model Outputs:** Remember that all hurricanes are unique, and model predictions may not capture rapid changes or rare events. Use the system's outputs as one of several sources in your decision-making process.

8 Maintenance

This chapter provides maintenance guidelines for both users (customers) and developers to ensure the Hurricane Intensity Prediction System remains reliable and effective.

For Customers

As a user, you can help keep the application running smoothly by following these steps:

- **Keep Your Data Organized:** Use accurate, well-formatted data files for predictions. Store your datasets securely and back them up regularly.
- **Update the Application:** If a new version or updated models are provided by the developer, follow the instructions to update your files.
- **Monitor Results:** Regularly review prediction outputs. If you notice unusual results, double-check your input data and consult the troubleshooting section.
- **Report Issues:** If you encounter technical problems or errors, report them to the developer or support team with as much detail as possible.
- **Follow Security Practices:** Do not share sensitive data or model files with unauthorized individuals. Store your files in secure locations.
- **Consult the Manual:** Refer to the user manual for troubleshooting tips, FAQs, and guidance on using the application.

For Developers

Developers are responsible for the technical upkeep and improvements of the system:

- **Update Dependencies:** Regularly update the `requirements.txt` file and test the application after any changes to dependencies.
- **Retrain and Update Models:** When new hurricane data is available, retrain the ARIMA and LSTM models and update the `models/` directory for users.
- **Maintain Data Quality:** Validate, clean, and archive datasets. Use version control for datasets and code.

- **Refactor and Document Code:** Keep the code modular, remove obsolete code, and update documentation to reflect any changes.
- **Ensure Security:** Protect sensitive data and model files. Use environment variables for credentials and restrict access as needed.
- **Test Thoroughly:** Maintain automated tests and regularly test the application on various platforms to ensure stability.
- **Support Users:** Respond to user reports, update the user manual as needed, and provide clear instructions for updates or new features.

Note: Customers should only perform maintenance steps described in the "For Customers" section. Technical maintenance and updates should be handled by the developer or system administrator.

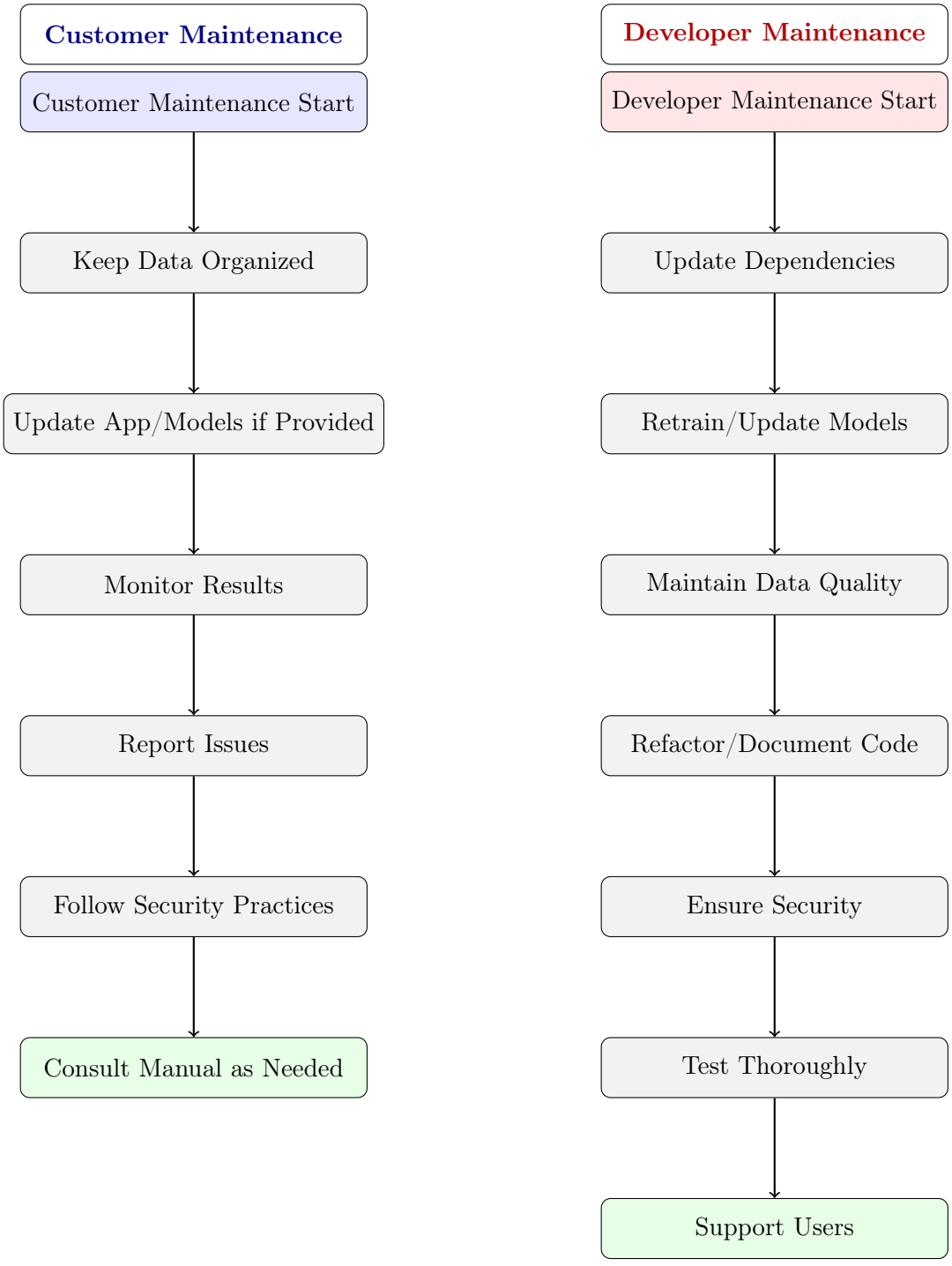


Figure 8.1: Parallel Maintenance Workflows for Customers and Developers

9 Troubleshooting

If you encounter issues while using the Hurricane Intensity Prediction System, follow the steps below to identify and resolve common problems. This approach will help ensure smooth operation and reliable predictions.

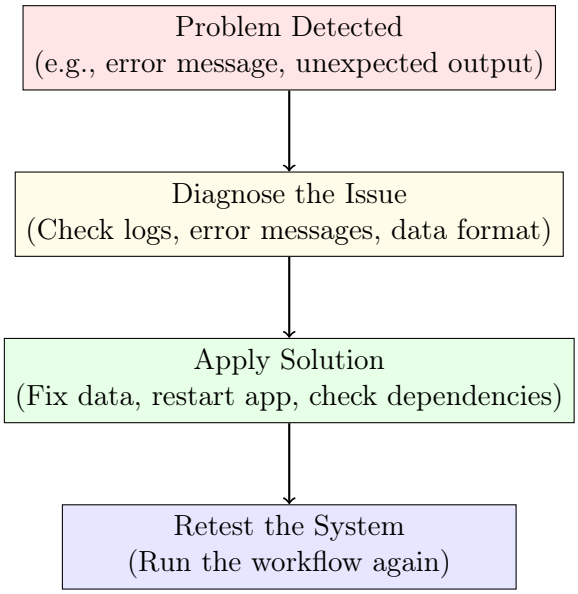


Figure 9.1: Troubleshooting Workflow for the Hurricane Intensity Prediction System

Common Issues and Solutions

- **Application Fails to Start:**
  Ensure all required Python packages are installed. Run `pip install -r requirements.txt` in your terminal.

- **File Upload Not Working:**
  Verify that your data file is in the correct format (e.g., CSV with columns for date, time, location, wind speed, pressure). Check for missing or malformed values.

- **No Predictions Generated:**
  Confirm that the `models/` directory contains the pre-trained ARIMA and LSTM model files. If missing, contact the developer to retrain and save the models.

- **Incorrect or Unexpected Predictions:**
  Double-check your input data for accuracy and completeness. Outliers or missing values can affect model performance.

- **Visualization Not Displayed:**
  Make sure your browser supports interactive plots and that there are no errors in the Streamlit app logs.

- **Permission or Access Errors:**
  Ensure you have the necessary permissions to read/write files in the project directory, especially for uploading data and downloading results.

- **Other Errors:**
  Review any error messages shown in the app or terminal. Consult the FAQ or contact support if the problem persists.

Tip: If you encounter persistent issues, restarting the Streamlit app and re-uploading your data can often resolve temporary glitches.

Note: For critical or unresolved problems, please contact the system administrator or developer for further assistance.

10 Help

This chapter offers guidance and resources if you encounter any difficulties while using the Hurricane Intensity Prediction System.

Quick Checklist

Before seeking additional help, please ensure you have completed the following:

- **Installed all required dependencies.**
  Refer to the **Installation** chapter to confirm all necessary Python packages are present.
- **Followed the initial setup procedures.**
  Review the **First Steps** and **Setup** chapters to make sure you have properly configured your environment and project files.
- **Understood the project workflow.**
  Refer to the **Workflow Diagram** and relevant sections to familiarize yourself with how data flows through the system.

Troubleshooting

If you experience technical issues such as errors during execution, missing files, or unexpected results, please consult the **Troubleshooting** chapter. It provides step-by-step solutions for common problems, including data formatting, model loading, and application errors.

Frequently Asked Questions (FAQ)

- **Q: The application does not start. What should I do?**
  A: Check that you are using a compatible Python version and that all required libraries are installed. Verify that you are running the correct command (`streamlit run app.py`) from the project directory.
- **Q: My predictions seem inaccurate or unusual. Why?**
  A: Ensure your uploaded data is complete and correctly formatted. Double-check that the models in the `models/` folder are up-to-date and have been trained with appropriate parameters.

- **Q: I cannot upload my data file. What could be wrong?**
  A: Make sure your file is in the supported format (e.g., CSV) and follows the required structure (date, time, location, wind speed, pressure, etc.). Check for missing or invalid entries.
- **Q: The results are not visualized or the download button does not work.**
  A: Refresh your browser and ensure you are using a supported web browser. If the issue persists, restart the Streamlit app.

Further Assistance

If the above steps do not resolve your issue:

- Reinstall the required Python packages and dependencies.
- Restart the application and carefully follow the user manual steps.
- Consult online resources or tutorials for working with ARIMA, LSTM, or data preprocessing in Python.
- Contact the developer for advanced support.

Important Reminder

This Project is designed for research and educational purposes. Always validate your data and review the **Safety Guidelines** chapter before using model outputs for any important decisions.