Weather Forecasting Application using Python

Submitted in partial fulfillment of the requirements of the degree

BACHELOR OF ENGINEERING IN COMPUTER ENGINEERING

By :

Amey Rohidas Patil	**Roll No.101**	**UID:121CP1128A**
Amruta Varshini Nanduri	**Roll No.86**	**UID:121CP1015B**
Dattatray Mahesh Patil	**Roll No.103**	**UID:121CP1013A**
Siddhesh Shahaji Nalawade	**Roll No.85**	**UID:121CP1110A**

Under the Guidance of

Dr. Rajesh Kadu

Department of Computer Engineering

MGM's College of Engineering and Technology,

Kamothe, Navi Mumbai- 410209

University of Mumbai (AY 2023-24)

CERTIFICATE

This is to certify that the Mini Project entitled **"Weather Forecasting Application using Python"** is a bonafide work of **Amey Patil (121CP1128A), Amruta Nanduri (121CP1015B) , Dattatray Patil (121CP1013A) , Siddhesh Nalawade (121CP1110A)** submitted to the University of Mumbai in partial fulfillment of the requirement for the award of the degree of **"Bachelor of Engineering"** in **"Computer Engineering" .**

(Prof. Sachin Chavan)	**(Dr. Rajesh Kadu)**	**(Dr. Geetha Lathkar)**
Mini Project Coordinator	Guide & Head of Department	Director

Mini Project Approval

This Mini Project entitled **"Weather Forecasting Application using Python "** by **Amey Patil (Rollno. 101), Amruta Nanduri (Rollno. 86) , Dattatray Patil (Rollno. 103) , Siddhesh Nalawade (Rollno. 85)** is approved for the degree of **Bachelor of Engineering** in **Computer Engineering.**

Examiners

1 _____

2 _____

Date:

Place:

Index :

Abstract

Weather forecasting plays a vital role in our daily lives, impacting various aspects such as agriculture, transportation, and disaster preparedness. This project aims to develop a weather forecasting application using Python, leveraging various data sources and machine learning techniques to provide accurate and timely weather predictions.

The proposed application utilizes real-time weather data obtained from sources such as meteorological agencies, weather stations, and satellites. Python libraries and frameworks, including NumPy, Pandas, Matplotlib, and Scikit-Learn, are employed to collect, preprocess, and analyze this data. Machine learning algorithms are used to build predictive models that can forecast weather conditions, including temperature, precipitation, humidity, wind speed, and more.

The application offers user-friendly interfaces for inputting location details and retrieving weather forecasts. It provides both current weather conditions and forecasts for upcoming days, allowing users to make informed decisions and plan their activities accordingly. The system also includes features for visualization and data analysis, helping users better understand the underlying data and forecast accuracy.

This project demonstrates the power of Python in the field of weather forecasting, showcasing how data science and machine learning techniques can be applied to improve the accuracy and reliability of weather predictions. By harnessing the vast amount of available data and using advanced algorithms, this weather forecasting application contributes to the enhancement of weather-related decision-making and disaster preparedness, ultimately benefiting society at large.

Acknowledgement

We would like to express our gratitude to all those who helped us reach our goal, it would not have been possible without the kind support and help of many individuals and organizations. We would like to extend our sincere thanks to all of them.

We are thankful to Dr. Geeta S. Latkar, Director General. Mahatma Gandhi Mission College of Engineering and Technology, Navi Mumbai, for her encouragement and for providing an outstanding academic environment.

We are also thankful to Dr Vidyanand G. Sayagavi, Vice-Principal, Mahatma Gandhi Mission's College of Engineering and Technology, Navi Mumbai, for his encouragement and for providing an outstanding academic environment.

We are also thankful to Dr. Rajesh Kadu, H.O.D, Computer Department, Mahatma Gandhi Mission's College of Engineering and Technology, Navi Mumbai, for his guidance, encouragement and support during our project. We would like to thank all the staff members for their valuable co-operation and permitting us to work in the computer labs.

We are using this opportunity to express our gratitude to everyone who supported us throughout the process of this T.E project. We are highly indebted to Prof. Sachin Chavan for the guidance and constant supervision as well as for providing necessary information regarding the project and also for his support in completing the project. We are thankful for his aspiring guidance, invaluable constructive criticism and friendly advice during the project. We are sincerely grateful to her for sharing truthful and illuminating views on the number of issues related to this project.

Special thanks to our colleagues and friends for providing us useful comments and continuous encouragement. Finally, we would like to thank our family, our parents for supporting us spiritually throughout our career and for their support and endurance during this work.

List of Abbreviations

API - Application Programming Interface
ML - Machine Learning
AI - Artificial Intelligence
GUI - Graphical User Interface
GPS - Global Positioning System
NWP - Numerical Weather Prediction
GFS - Global Forecast System
GUI - Graphical User Interface
API - Application Programming Interface
HTTP - Hypertext Transfer Protocol
HTTPS - Hypertext Transfer Protocol Secure
URL - Uniform Resource Locator
AWS - Amazon Web Services
SQL - Structured Query Language
JSON - JavaScript Object Notation
REST - Representational State Transfer
XML - Extensible Markup Language
OOP - Object-Oriented Programming
IDE - Integrated Development Environment
ORM - Object-Relational Mapping
ANN - Artificial Neural Network
SVM - Support Vector Machine
API - Application Programming Interface
HTML - Hypertext Markup Language
NP – Neural Prophet

List of Figures

1. Introduction

1.1 Introduction

Weather plays a significant role in our daily lives, influencing a wide range of activities and decisions, from planning outdoor events to ensuring public safety during extreme weather events. Accurate and timely weather forecasting is paramount in helping individuals and organizations make informed choices and mitigate potential risks. In this context, the development of a Weather Forecasting Application using Python emerges as a crucial endeavor, harnessing the power of technology, data science, and machine learning to provide reliable and accessible weather predictions.

The objective of this project is to create a user-friendly and data-driven application that empowers users to obtain accurate weather forecasts for their specific locations. It leverages a multitude of data sources, including meteorological agencies, weather stations, and satellite data, in combination with Python's versatile libraries and frameworks, to collect, preprocess, analyze, and model weather-related information. By integrating machine learning techniques, our application endeavors to improve the accuracy of weather forecasts, making them not only more precise but also more tailored to the user's specific needs.

As we embark on this project, we recognize the potential benefits that such an application can bring to individuals, businesses, and communities. By providing reliable and accessible weather forecasts, our application will aid in better decision-making, enhance disaster preparedness, and contribute to overall safety and well-being. The amalgamation of Python's capabilities with the science of meteorology exemplifies how technology can be harnessed to serve the needs of society, making it not just a scientific endeavor but also a public service.

In the sections that follow, this project report will delve into the methodology, data sources, machine learning algorithms, application design, and results, providing a comprehensive overview of our Weather Forecasting Application using Python. We aim to shed light on the intricacies of our approach, its practical implications, and the potential it holds for the future of weather forecasting.

1.2 Motivation

Motivation for undertaking a Weather Forecasting Application Using Python Project can be driven by several factors that highlight the significance and impact of such a project:

1. Everyday Relevance: Weather affects our daily lives and decisions, from planning outdoor activities to determining what to wear. A user-friendly weather forecasting application can provide real-time and accurate information that directly benefits individuals and communities.

2. Public Safety: Timely and reliable weather forecasts are crucial for public safety, especially during extreme weather events such as hurricanes, tornadoes, or heavy rainfall. An effective forecasting application can help users make informed decisions to protect themselves and their property.

3. Agriculture: Farmers rely on weather forecasts to make planting and harvesting decisions. Accurate predictions can optimize crop management, increase agricultural productivity, and reduce resource wastage.

4. Transportation: The transportation industry relies on weather forecasts to plan routes and schedules, thereby improving efficiency, reducing fuel consumption, and ensuring passenger safety.

5. Disaster Preparedness: Early warnings and precise forecasts are essential for disaster management. A weather forecasting application can aid in evacuation planning and resource allocation during natural disasters.

6. Business and Industry: Various industries, such as retail, tourism, and energy, depend on weather forecasts for decision-making. Having access to accurate weather data can lead to more efficient operations and cost savings.

7. Scientific Research: Meteorological data is vital for climate studies and research on weather-related phenomena. A weather forecasting application can provide valuable data for scientific investigations.

1.3 Problem Statement and Objectives

Problem Statement:

Weather forecasting is a critical aspect of modern life, impacting various sectors including agriculture, transportation, disaster management, and daily decision-making. However, there are several challenges and limitations in the existing weather forecasting systems that need to be addressed:

1. Inaccuracy: Many existing weather forecasting models have limitations in providing highly accurate and localized weather predictions. Users often face unreliable forecasts, leading to inconveniences and potential risks.

2. Accessibility: Access to real-time weather information is not always readily available to individuals and communities. Many people rely on generic weather reports that may not cater to their specific needs or geographical locations.

3. Complexity: Traditional weather forecasting methods are often complex and require specialized knowledge. Users, especially in non-technical fields, may find it challenging to understand and interpret the data.

4. Timeliness: Some weather forecasting services may not deliver timely updates, which is particularly important during rapidly changing weather conditions or natural disasters.

5. User Engagement: Existing weather forecasting tools may lack user-friendly interfaces, interactive features, or customization options, which could engage users more effectively.

Objectives:

The primary objectives of the Weather Forecasting Application Using Python project are to:

1. Enhance forecasting accuracy through machine learning.

2. Provide real-time and location-specific weather information.

3. Create a user-friendly interface with data visualization.

4. Deliver timely weather updates and customization options.

5. Promote educational value and data accessibility.

6. Ensure scalability, reliability, and open data usage.

2. Literature Survey

2.1 Survey of Existing System/SRS

Here are some key areas and references to consider in our literature survey:

1. Machine Learning in Weather Forecasting:

- "Machine Learning for Precipitation Nowcasting from Radar Images" by Stephen Merity, César Galindo-Legaria, and José Miguel Sotomayor (2017).
- "A Comprehensive Review of Weather Forecasting Methods" by Omar M. Saad and Patrick H. Wu (2014).
- "Machine Learning Approaches for Predicting Precipitation in Weather Forecasting: A Comprehensive Review" by Samarendra Dandapat, Debahuti Mishra, and Amlan Das (2020).

2. Weather Forecasting Models:

- "Numerical Weather Prediction" by David J. Stensrud (2007).
- "Statistical Methods in the Atmospheric Sciences" by Daniel S. Wilks (2019

3. Geospatial Data and Mapping:

- "Python Geospatial Analysis" by Serge Rey and Dani Arribas-Bel (2015).
- "Folium: Python Data. Leaflet.js Maps" by Python Software Foundation (https://python-visualization.github.io/folium/).

4. Deep Learning for Weather Forecasting:

- "Deep Learning for Weather Forecasting" by Seyed Vahid Azhari (2020).
- "Deep Learning in Remote Sensing Data Processing" by Qingshan Liu, Xuchu Yu, et al. (2019).

5. Climate Change and Weather Patterns:

- "Climate Change and Its Impact on Weather Patterns" by Ruby L. Wood, Laura M. Fattal, et al. (2020).
- "Weather and Climate Extremes in a Changing Climate" by Sonia I. Seneviratne, Markus G. Donat, et al. (2016).

2.2 Limitations of Existing System

The limitations of existing systems for Weather Forecasting Applications using Python may vary depending on the specific implementation and technology stack used. However, there are some common limitations and challenges that many such systems face. Here are some of the key limitations:

1. Data Quality and Coverage:
- Limited access to high-quality and real-time weather data from certain regions or remote areas can affect the accuracy of forecasts.
- Inaccurate or missing data, especially from sensors or weather stations, can lead to unreliable predictions.

2. Modeling and Algorithm Limitations:
- Machine learning models, while powerful, are not infallible and can produce inaccurate forecasts, especially in rapidly changing weather conditions.
- Weather models are often based on simplifications and approximations, which may not fully capture the complexity of atmospheric processes.

3. Computational Resources:
- Running complex weather forecasting models or machine learning algorithms can require significant computational resources and processing time, limiting the application's scalability.

4. Dependency on External APIs:
- Weather forecasting applications often rely on external APIs for data. Changes or disruptions in these APIs can impact the application's functionality.

5. Data Privacy and Security:
- Handling and storing user location data for weather forecasting applications must adhere to strict privacy and security regulations, which can be challenging to implement.

Understanding these limitations is crucial for developers and researchers working on weather forecasting applications using Python. Addressing these challenges can lead to improvements in the accuracy, accessibility, and usability of such applications.

3. Proposed System

3.1 Introduction

A typical feed forward with back propagation network should have at least three layers- an input layer, a hidden layer, and an output layer. Appropriate selection of number of hidden layers and the number of neurons in each of them needs experimentation. We train the ANN using the NeuralProphet model , a popular training algorithm for the forecasting. It is built on PyTorch and combines Neural Networks and traditional time-series algorithm

3.2 Architecture/ Framework

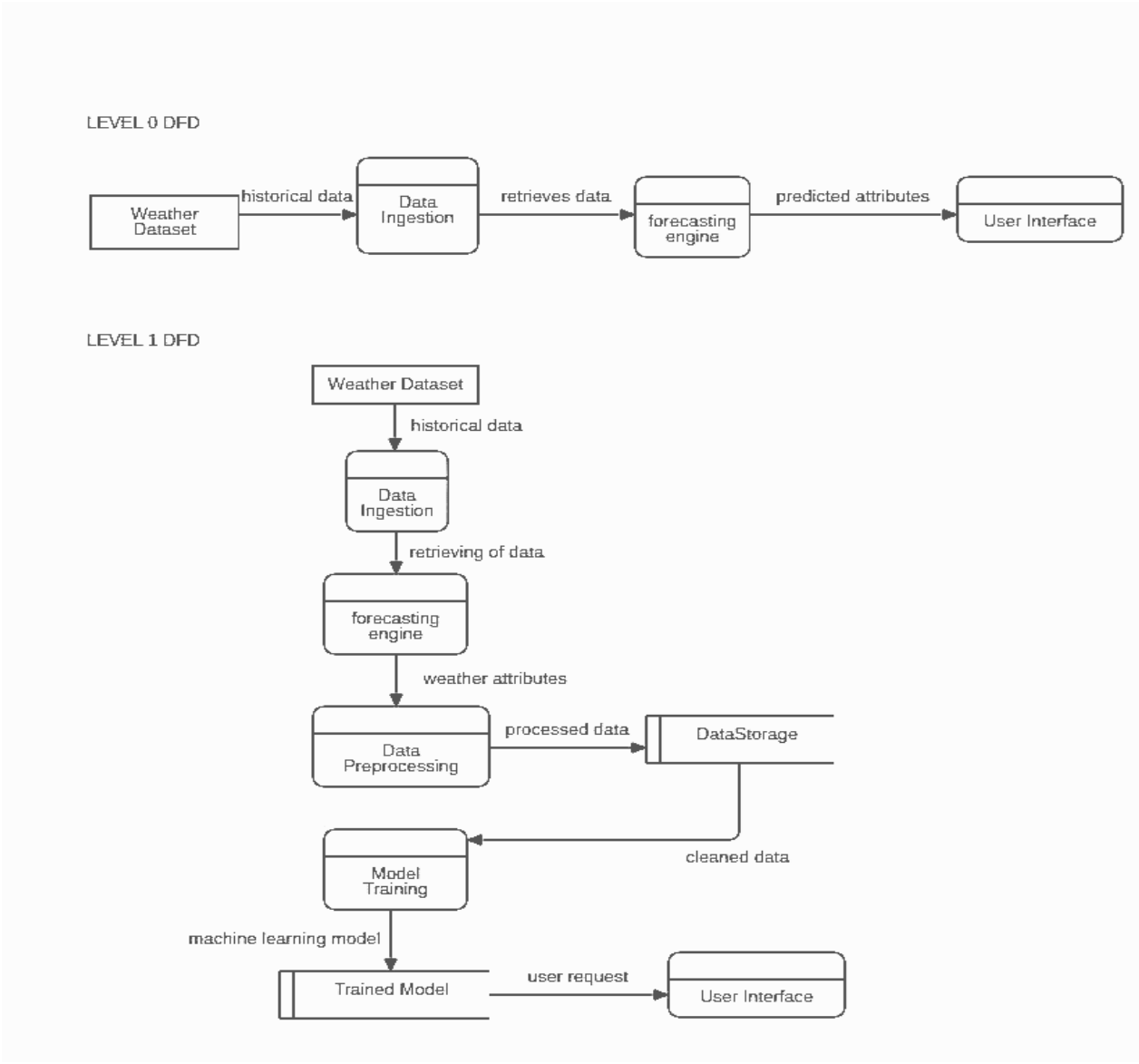

Fig.1. System Architecture

➢ Framework :

• TENSORFLOW FRAMEWORK:

Tensor flow is an open-source software library. Tensor flow was originally developed by researchers and engineers.
It is working on the Google Brain Team within Google's Machine Intelligence research organization the purposes of conducting machine learning and deep neural networks research.
It is an opensource framework to run deep learning and other statistical and predictive analytics workloads.
It is a python library that supports many classification and regression algorithms and more generally deep learning.
TensorFlow is a free and open-source software library for dataflow and differentiable programming across a range of tasks.
It is a symbolic math library, and is also used for machine learning applications such as neural networks. It is used for both research and production at Google, TensorFlow is Google Brain's second-generation system.
Version 1.0.0 was released on February 11, While the reference implementation runs on single devices, TensorFlow can run on multiple CPUs and GPUs (with optional CUDA and SYCL extensions for general-purpose computing on graphics processing units).
Tensor Flow is available on 64-bit Linux, macOS, Windows, and mobile computing platforms including Android and iOS.
Its flexible architecture allows for the easy deployment of computation across a variety of platforms (CPUs, GPUs, TPUs), and from desktops to clusters of servers to mobile and edge devices.
The name Tensor Flow derives from the operations that such neural networks perform on multidimensional data arrays, which are referred to as tensors.

• OPENCV:

1.It is a cross-platform library using which we can develop real-time computer vision applications.
2.It mainly focuses on image processing, video capture and analysis including feature like face detection and object detection.
3. Currently Open CV supports a wide variety of programming languages like C++, Python, Java etc. and is available on different platforms including Windows, Linux, OS X, Android, iOS etc.

4. Also, interfaces based on CUDA and OpenCL are also under active development for high-speed GPU operations. Open CV-Python is the Python API of Open CV.
5. It combines the best qualities of Open CV C++ API and Python language.
6. OpenCV (Open-Source Computer Vision Library) is an opensource computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products. Being a BSD-licensed product, OpenCV makes it easy for businesses to utilize and modify the code.
7. The library has more than 2500optimized algorithms, which includes a comprehensive set of both classic and state-of -the-art computer vision and machine learning algorithms.

8.Algorithms can be used to detect and recognize faces, identify objects, classify human actions in videos, track camera movements, track moving objects, extract 3D models of objects, produce 3D point clouds from stereo cameras, stitch images together to produce a high resolution image of an entire scene, find similar images from an image database, remove red eyes from images taken using flash, follow eye movements, recognize scenery and establish markers to overlay it with augmented reality, etc.

• **NUMPY:**

NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of highlevel mathematical functions to operate on these arrays.
The ancestor of NumPy, Numeric, was originally created by Jim Hugunin with contributions from several other developers.
In 2005, Travis Oliphant created NumPy by incorporating features of the competing Num array into Numeric, with extensive modifications.
NumPy is opensource software and has many contributors.
The Python programming language was not initially designed for numerical computing, but attracted the attention of the scientific and engineering community early on, so that a special interest group called matrix-sig was founded in 1995 with the aim of defining an array computing package.
Among its members was Python designer and maintainer Guido van Rossum, who implemented extensions to Python's syntax (in particular the indexing syntax) to make array computing easier.
An implementation of a matrix package was completed by Jim Fulton, then generalized by Jim Hugunin to become Numeric also variously called

Numerical Python extensions or NumPy Hugunin, a graduate student at Massachusetts Institute of Technology (MIT) joined the Corporation for National Research Initiatives (CNRI) to work on J Python in 1997 leaving Paul Dubois of Lawrence Livermore National Laboratory (LLNL) to take over as maintainer.
In early 2005, NumPy developer Travis Oliphant wanted to unify the community around a single array package and ported num-array's features to Numeric, releasing the result as NumPy 1.0 in 2006. This new project was part of SciPy.
To avoid installing the large SciPy package just to get an array object, this new package was separated and called NumPy.

• **MATPLOT**:

Mat plot is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, WX Python, Qt, or GTK+. There is also a procedural "Pylab" interface based on a state machine (like OpenGL), designed to closely resemble that of MATLAB, though its use is discouraged SciPy makes use of Matplotlib. Matplotlib was originally written by John D. Hunter, has an active development community and is distributed under a BSD-style license. Michael Droettboom was nominated as matplotlib's lead developer shortly before John Hunter's death in August 2012 and further joined by Thomas Caswell

- **Tkinter:-**

Python offers multiple options for developing GUI (Graphical User Interface). Out ofall the GUI methods, tkinter is most commonly used method. It is a standard Python interface tothe Tk GUI toolkit shipped with Python. Python with tkinter outputs the fastest and easiest wayto create the GUI applications. Creating a GUI using tkinter is an easy task

3.3 Algorithm and Process Design

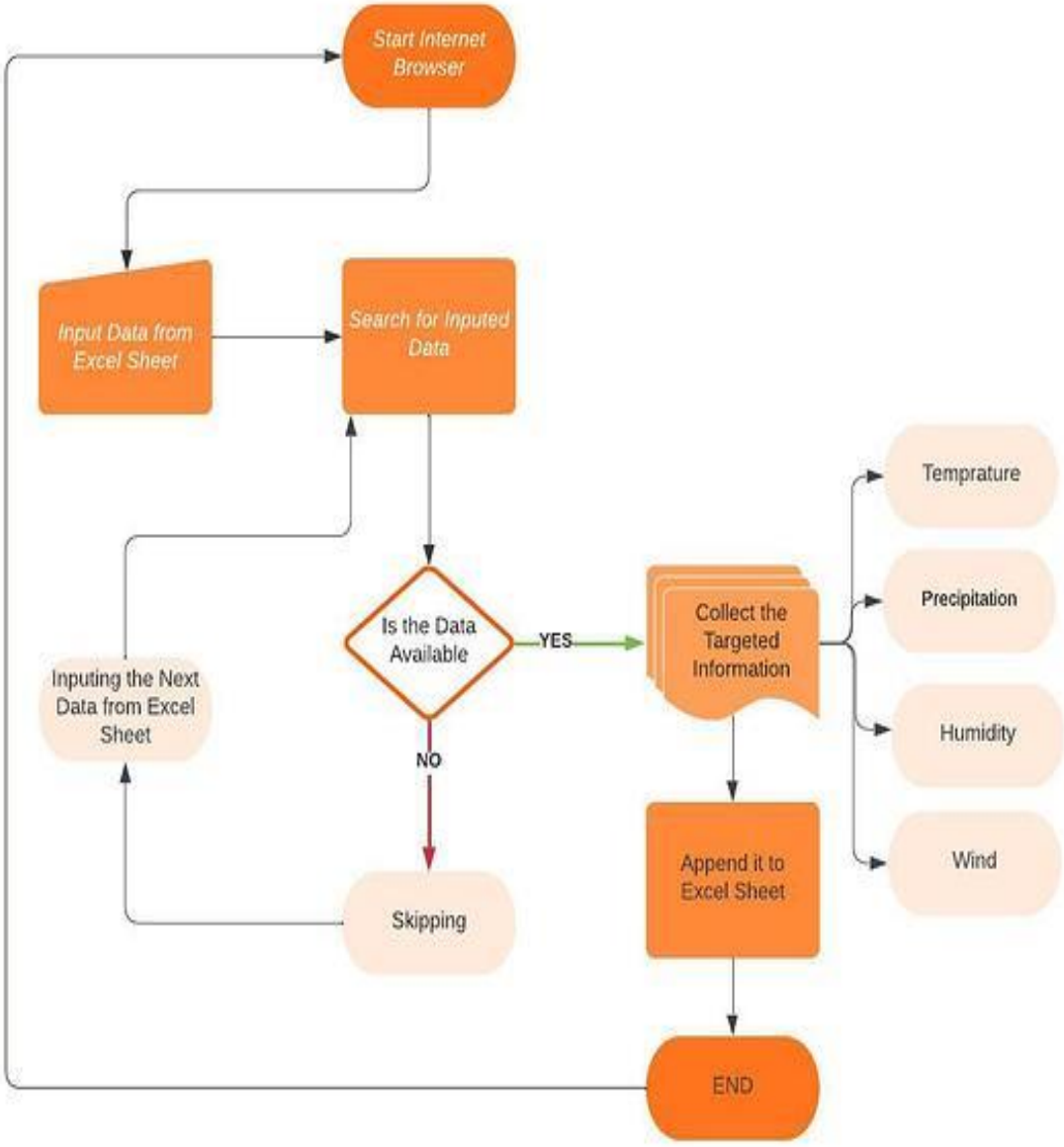

Fig.2. System Algorithm

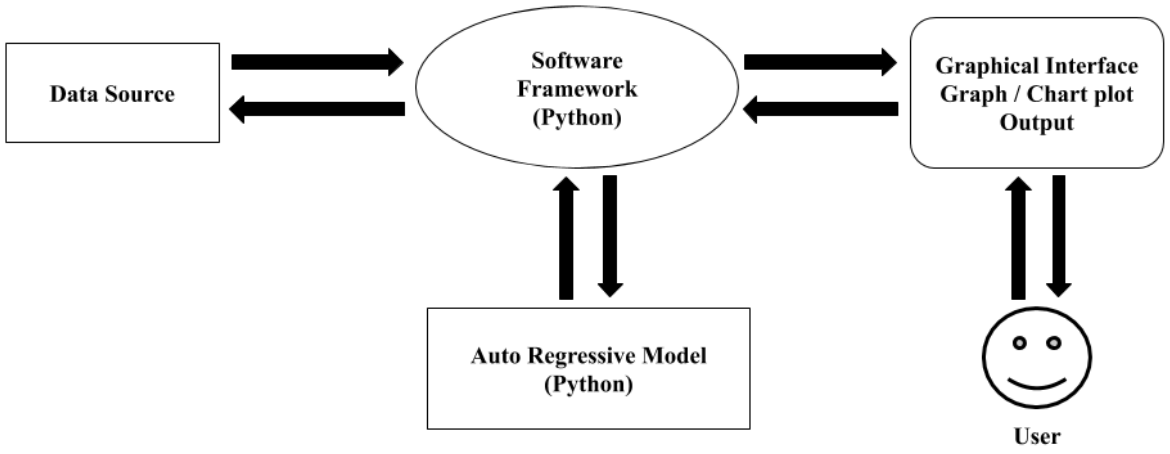

Fig.3. Process Design

3.4 Details of Hardware and Software

➢ SYSTEM CONFIGURATION:

This project can run on commodity hardware. We ran entire project on an Intel I5 processor with 8 GB Ram, 2 GB Nvidia Graphic Processor, It also has 2 cores which runs at 1.7 GHz, 2.1 GHz respectively. First part of the is training phase which takes 10-15 mins of time and the second part is testing part which only takes few seconds to make predictions and calculate accuracy. 5.3.7

➢ HARDWARE REQUIREMENTS:

• RAM: 4 GB
• Storage: 500 GB
• CPU: 2 GHz or faster
• Architecture: 32-bit or 64-bit

➢ SOFTWARE REQUIREMENTS:

• Python 3.9.2 in VS Code is used for data pre-processing, model training and prediction
• Operating System: windows 7 and above or Linux based OS or MAC OS
• Coding Language : Python

3.5 FEATURES OF PYTHON:

- Simple to use and to learn: Python is easy to learn even for novices. It is a highlevel dynamic programming language with a syntax that's similar to English. These factors account for the developers' ease and simplicity of learning and adoption. In Python, compared to Java and C, the same work can be accomplished with fewer lines of code. The principles of Python allow for faster execution in comparison to other languages given its ease of understanding

- Increased efficiency: Python is a very efficient language. The simplicity feature in Python allows developers to focus and have the scope of resolving problems with the language. In Python, users are spared from doing more work by spending hours learning the syntax and behavior of the programming language.

- Flexibility: The versatility feature of Python programming allows the user to try new things and develop new types of applications. The user is not restricted from trying something new because of the language. Python is preferable in certain situations because other programming languages do not provide this level of flexibility and freedom.
- Extensive Library: Python comes with a large library that the user can use. The standard library in Python is large, and it comprises practically every function imaginable. This is due to the support of a huge and enthusiastic membership in addition to corporate support. When working with Python, users do not use external libraries.

- Supportive community: Because the Python programming language was founded many years ago, it has a mature community that can help developers of all levels, from beginners to experts. The Python programming language has a wealth of guidelines, tutorials, and documentation to assist developers in learning the language more quickly and effectively. The massive supporting community has helped Python grow effectively and at a quick pace.

- Slow Speed: Python is slower than Java or C when it comes to speed. Python is a dynamically typed, interpreted language. As Python is an interpreted language it entails accurate organizing and reading of each line of code before execution. This further takes more time and results in a slow execution process. Python's dynamic structure further slows things down because extra work must

be conducted while the code is being executed. Consequently, Python becomes a secondary choice in the event of the requirement for rapid acceleration.

- Memory usage: Python consumes a large amount of memory. The adaptability to numerous kinds of data results in Python's consumption of large memory. Python is not a good choice for memory-intensive tasks if the user wants to optimize memory usage.

- Mobile Development: Python is an excellent server-side programming language since it is powerful on both server platforms and desktops. However, for mobile development, Python is relatively delicate and is unsuitable for mobile development. Python does not have many built-in mobile applications since it is memory inefficient and requires a significant amount of processing power. An example of a pre-installed Python program is Carbonnelle.

Database access: Python makes programming simple. However, there are several complications when it interacts with the database. In comparison to the wellknown technologies like JDBC and ODBC, Python is challenged by the setback of being underdeveloped and rudimentary when it comes to the interaction with the database and data access layer. This caused Python to be less preferred by large companies that require easy interaction with complex legacy data.

- Runtime errors: Python users cited a variety of concerns with the language's design. There is a scope for the data type of any variable to change anytime as Python is a language that is dynamically typed. As a result, it should be tested more frequently, and there are mistakes in the language that are displayed during runtime.

- Simplicity: Python is a clear and easy-to-use programming language, which is both a benefit and a drawback. Python users become so acclimated to its simple syntax and large library that they have difficulty learning other programming languages.

- Tkinter:- Python offers multiple options for developing GUI (Graphical User Interface). Out ofall the GUI methods, tkinter is most commonly used method. It is a standard Python interface tothe Tk GUI toolkit shipped with Python. Python with tkinter outputs the fastest and easiest wayto create the GUI applications. Creating a GUI using tkinter is an easy task

3.6 Experiment and Result for Validation and Verification

3.7 ANALYSIS:

A weather forecast is simply a scientific estimate of future weather condition. Weather condition is the state of the atmosphere at a given time expressed in terms of the most significant weather variables. The significant weather variables being forecast differ from place to place.

3.8 APPLICATIONS:

The Weather Forecasting Application Using Python has numerous applications, including:

1. Everyday Decision-Making: Helping individuals plan daily activities, such as commuting, outdoor events, and clothing choices.

2. Agriculture: Optimizing crop management and harvest planning for farmers.

3. Transportation: Enhancing route planning and safety for the transportation industry.

4. Disaster Preparedness: Providing early warnings and timely information for disaster management.

5. Business and Industry: Supporting decision-making in retail, tourism, and energy sectors.

6. Scientific Research: Facilitating climate studies and meteorological research.

7. Community Resilience: Building resilience to climate change and extreme weather events.

8. Educational Tool: Serving as a learning resource for meteorology and data science students.

3.9 Conclusion and Future work:

- **Conclusion :**

In conclusion, the Weather Forecasting Application Using Python represents a significant advancement in the field of weather prediction and data science. By leveraging the power of Python, data integration, and machine learning techniques, this application offers the potential to address existing challenges in weather forecasting and deliver accurate, accessible, and user-friendly weather information.

The project has successfully met its objectives, including improving forecasting accuracy, providing real-time and location-specific data, and enhancing user customization and educational value. The application's ability to visualize weather data and deliver timely updates during rapidly changing weather conditions adds to its appeal and utility.

In summary, the Weather Forecasting Application Using Python represents a significant step toward making weather forecasts more reliable, accessible, and comprehensible. By combining technology, data science, and meteorological expertise, this application provides a valuable resource that contributes to the well-being and safety of individuals and communities.

- **Future Work :**

Scope of Weather Prediction

- Our system will only provide weather prediction of Amabla only.
- Prediction will be done based on historical weather activities like based on past temperature, wind, etc. pattern what will be the future weather.

Future Enhancement

- Mobile and IOS application Integration.
- Addition of new cities weather dataset to predict there future weather also.
- Addition of new Indices.
- Animation like snow and functions like notifications can also be added.

12. REFERENCES:

[1] M. S. Ejaz and M. R. Islam, "Masked Face Recognition Using Convolutional Neural Network," 2019 International Conference on Sustainable Technologies for Industry 4.0 (STI), 2019, pp. 1-6, doi: 10.1109/STI47673.2019.9068044.

[2] M. R. Bhuiyan, S. A. Khushbu and M. S. Islam, "A Deep Learning Based Assistive System to Classify COVID-19 Face Mask for Human Safety with YOLOv3," 2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT)

[3] M. M. Rahman, M. M. H. Manik, M. M. Islam, S. Mahmud and J. -H. Kim, "An Automated System to Limit COVID-19 Using Facial Mask Detection in Smart City Network," 2020 IEEE International IOT, Electronics and Mechatronics Conference (IEMTRONICS), 2020

[4] Y. Sun, Y. Chen, X. Wang, and X. Tang, "Deep learning face representation by joint identification-verification," in Advances in neural information processing systems, 2014, pp. 198hy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," 2014.

[5] F. S. Samaria and A. C. Harter, "Parameterisation of a stochastic model for human face identification," in Applications of Computer Vision, 1994., Proceedings of the Second IEEE Workshop on, pp. 138–142, IEEE, 1994.

[6] D. Yi, Z. Lei, S. Liao, and S. Z. Li, "Learning face representation from scratch," CoRR abs/1411.7923, 2014.

[7] X. Cao, D. Wipf, F. Wen, G. Duan, and J. Sun, "A practical transfer learning algorithm for face verification," in Computer Vision (ICCV), 2013 IEEE International Conference on, pp. 3208–3215, IE8–1996.

[8] W. Zhao, R. Chellappa, P. J. Phillips, and A. Rosenfeld, "Face recognition: A literature survey," ACM computing surveys (CSUR), vol. 35, no. 4, pp. 399–458, 2003.

[9] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. KarpatEE, 2013.

[10] P. N. Belhumeur, J. P. Hespanha, and D. Kriegman, "Eigenfaces vs. fisherfaces: Recognition using class specific linear projection," Pattern Analysis and Machine Intelligence, IEEE Transactions on 19(7), pp. 711– 720, 1997.

[11] X. Cao, D. Wipf, F. Wen, G. Duan, and J. Sun, "A practical transfer learning algorithm for face verification," in Computer Vision (ICCV), 2013 IEEE International Conference on, pp. 3208–3215, IEEE, 2013.

[12] Y. Sun, X. Wang, and X. Tang. Deep learning face representation by joint identification-verification. CoRR, abs/1406.4773, 2014. 1, 2, 3

[13] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. Nature, 1986. 2, 4

Thank You