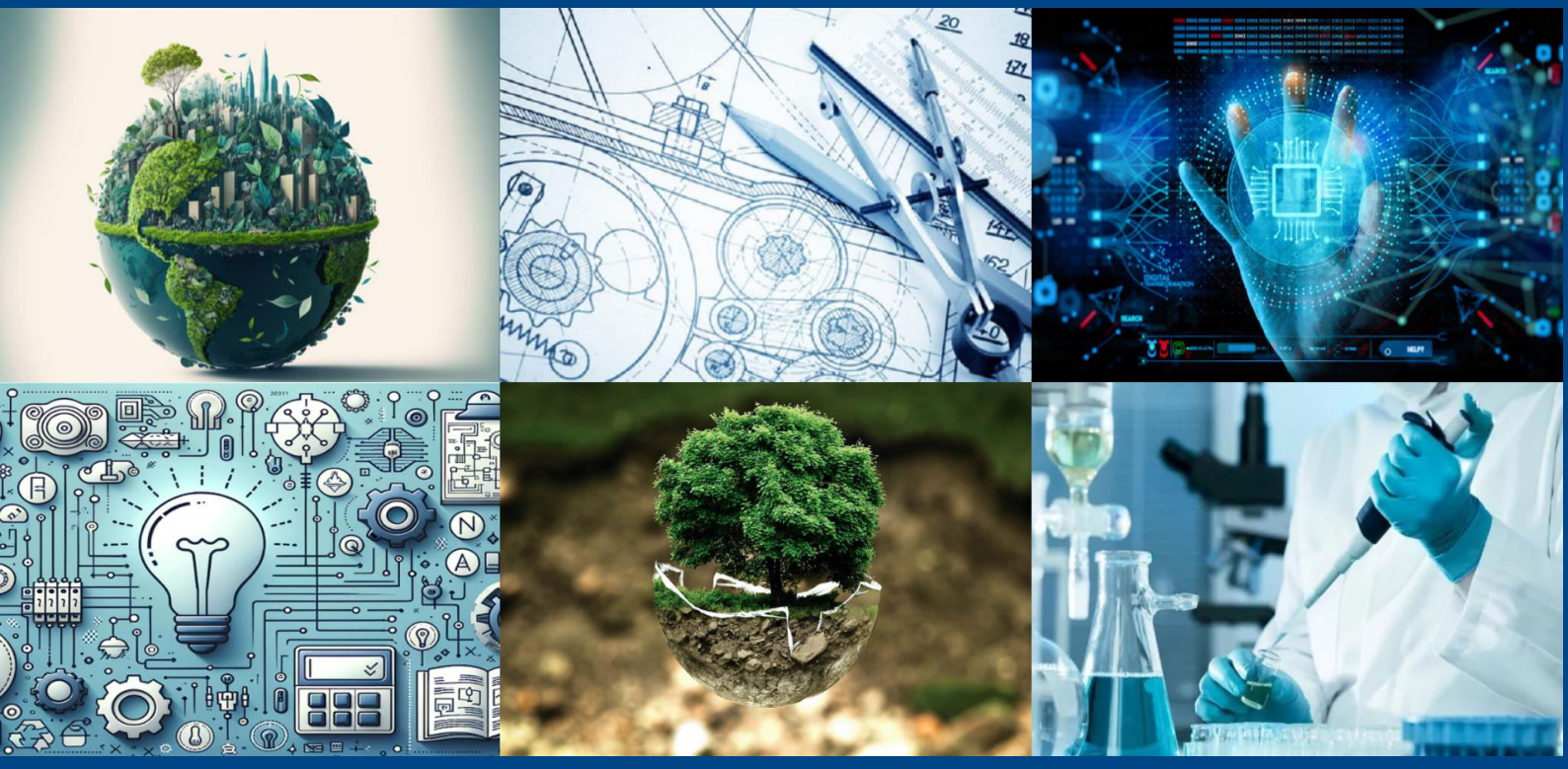
ISSN: 2582-7219

International Journal of Multidisciplinary Research in Science, Engineering and Technology

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)



Impact Factor: 8.206

Volume 8, Issue 3, March 2025

© 2025 IJMRSET | Volume 8, Issue 3, March 2025| DOI: 10.15680/IJMRSET.2025.0803177

ISSN: 2582-7219 | www.ijmrset.com | Impact Factor: 8.206| ESTD Year: 2018|

International Journal of Multidisciplinary Research in Science, Engineering and Technology (IJMRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

Extracting Weather Data from Google in Python for Weather Alert

I. Varun Sith Ramaya, Mr. R. Janarthanan MCA., M.Phil., (Ph.D)

Bachelor of Computer Science with Data Analytics, Sri Ramakrishna College of Arts and Science, Coimbatore, Tamil Nadu, India

Assistant Professor, Department of CS with Cyber Security, Sri Ramakrishna College of Arts and Science, Coimbatore, Tamil Nadu, India

ABSTRACT: The Automated Weather Alert System is a Python-based desktop application using Tkinter for real-time weather monitoring and automated alerts. It fetches data from the Open Weather Map API, displaying temperature, conditions, and wind speed with animated icons. Key features include automatic 30-minute updates, a countdown timer, manual refresh, and customizable locations**.** The system detects hazardous weather and sends email alerts. Multi-threading ensures a smooth, responsive UI, preventing freezes during API calls. Designed for usability, it balances automation with user control, making it ideal for personal, agricultural, and safety applications.

I. INTRODUCTION

In today's fast-paced digital world, staying informed about weather conditions is essential for both personal and professional activities. Sudden changes in weather, such as storms, heavy rain, or extreme heat, can significantly impact travel plans, outdoor events, agricultural activities, and even business operations. This project, "Mail Interface for Weather Alerts in Python," is designed to provide a reliable and automated system that fetches real-time weather data and notifies users through email alerts when specific conditions are met. The system is built using Python and integrates multiple technologies to achieve its objectives. It uses the Open Weather Map API to retrieve weather details for a given city, such as temperature, weather conditions, and wind speed. The user interacts with the system through a Graphical User Interface (GUI) built with Tkinter, where they can enter the name of a city and fetch the latest weather data. If the weather conditions indicate potential risks—such as mist, heavy rainfall, or high temperatures—the system automatically triggers an email notification to alert the recipient. The email notification feature is implemented using Python's SMTP (Simple Mail Transfer Protocol) library, which allows the system to send messages via an email server. The email alert includes important details such as the city name, temperature, and weather description, enabling the recipient to take appropriate action based on the provided information. This functionality is particularly useful for individuals who need constant weather updates without manually checking weather reports. The project demonstrates real-world applications of Python in automation, combining API integration, GUI development, and email communication in a single application. It is designed to be efficient, user-friendly, and easily extendable, with potential enhancements such as multi-city tracking, SMS alerts, and voice notifications in future iterations. By providing a seamless and automated way to stay updated on weather conditions, this project offers a practical solution for weather monitoring and timely alerts, making it beneficial for a wide range of users, from travelers and event organizers to farmers and businesses.

1.1 OVERVIEW

Overview: Mail Interface for Weather Alerts in Python

The Mail Interface for Weather Alerts in Python is a user-friendly and automated system designed to fetch real-time weather data and notify users via email when specific weather conditions are met. The project integrates Python's Tkinter GUI framework, the Open Weather Map API, and SMTP email services to provide an efficient way of tracking and receiving weather updates. The system operates by allowing users to input a city name through a Tkinter-based Graphical User Interface (GUI). Upon submission, the application retrieves temperature, weather conditions, and wind speed from the Open Weather Map API and displays the information on the screen. The system then evaluates predefined alert conditions—such as mist, rain, or extreme temperatures above 30°C. If any of these conditions are detected, the program automatically triggers an email notification to a specified recipient. The email alert system is implemented using Python's

ISSN: 2582-7219 | www.ijmrset.com | Impact Factor: 8.206| ESTD Year: 2018|

International Journal of Multidisciplinary Research in Science, Engineering and Technology (IJMRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

smtplib module, which enables the application to securely send emails via an SMTP server. The email contains the city name, current temperature, and weather description, ensuring that recipients receive a clear and concise weather update. This feature is especially useful for individuals and organizations that require immediate weather notifications without manual checking. This project effectively demonstrates the power of Python for real-world automation, highlighting its capabilities in API integration, GUI development, and email communication. The system is designed to be efficient and extendable, allowing for future enhancements such as multi-city tracking, SMS alerts, voice notifications, and integration with IoT-based weather monitoring devices. By automating weather tracking and notifications, the Interface for Weather Alerts serves as a practical and scalable solution for both personal and professional use cases Mail.

1.2 OBJECTIVES OF THE PROJECT

The primary objectives of this project are:

• To develop an automated weather monitoring system – The application retrieves real-time weather data using the OpenWeatherMap API and displays it through a user-friendly Tkinter GUI.

• To provide an intuitive and interactive user interface – The system allows users to input a city name and fetch weather details, including temperature, weather description, and wind speed, with a single click.

• To implement an automated email alert system – The project ensures that users receive real-time notifications via email when predefined weather conditions (such as mist, rain, or extreme temperatures) are detected.

• To integrate secure email functionality using SMTP – The system sends email alerts using Python's smtplib library, ensuring secure and reliable email communication.

• To enhance awareness and preparedness for severe weather – By automating weather alerts, the system helps users plan activities, take precautions, and stay informed about potential weather hazards.

• To demonstrate Python's capability in automation and API integration – This project showcases the use of Python for real-world applications, including data retrieval from APIs, GUI development, and email automation.

• To build a scalable and extendable solution – The system is designed to be expandable with future enhancements, such as multi-city tracking, SMS alerts, voice notifications, and integration with IoT-based weather monitoring systems.

Real-World Applications and Impact Objectives:

• To assist in disaster preparedness – The system should help users take precautionary measures against severe weather conditions, such as storms or extreme heat.

• To support agricultural planning – Farmers can use the system to get alerts about rainfall, drought, or frost, helping them protect crops and livestock.

• To improve travel and logistics planning – The system can notify travelers, airlines, and logisticscompanies about potential weather disruptions.

• To enhance accessibility for visually impaired users – By integrating screen reader support and voice alerts, the system could help visually impaired users stay informed about the weather.

1.3 ORGANIZATION PROJECT

The Mail Interface for Weather Alerts in Python is structured into several key components to ensure smooth functionality. The project begins with a graphical user interface (GUI) created using Tkinter, allowing users to enter a city name and check its weather conditions.

Once the user inputs a city, the system fetches real-time weather data using the Open Weather Map API, retrieving details like temperature, weather description, and wind speed.

The retrieved data is then evaluated against predefined alert conditions—such as mist, rain, or high temperatures. If these conditions are met, the system triggers an automated email alert using Python's SMTP library, sending a notification to a specified recipient with details of the weather conditions. Error handling is incorporated to ensure smooth API communication and secure email dispatch.

The project is organized into functional sections, including GUI setup, API integration, decision-making for alerts, and email automation, making it a well-structured and scalable system. Future enhancements could include multi-city tracking, SMS alerts, and weather forecasting extensions

1.4 SCOPE OF THE SYSTEM

The Mail Interface for Weather Alerts in Python is designed to provide real-time weather updates and automated email notifications, making it a valuable tool for individuals and organizations. The project integrates Tkinter for the user interface, Open Weather Map API for weather data retrieval, and SMTP for email alerts, offering a practical and efficient solution for weather monitoring.

ISSN: 2582-7219 | www.ijmrset.com | Impact Factor: 8.206| ESTD Year: 2018|

International Journal of Multidisciplinary Research in Science, Engineering and Technology (IJMRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

Current Scope:

1. Real-Time Weather Updates: Users can enter a city name and retrieve temperature, weather conditions, and wind speed instantly.
2. Automated Email Alerts: If predefined weather conditions (e.g., mist, rain, or high temperatures) are detected, the system sends email notifications to a recipient.
3. User-Friendly Interface: The project features a Tkinter-based GUI, allowing easy interaction for users with minimal technical knowledge.
4. Secure Email Transmission: Emails are sent using SMTP authentication, ensuring secure and reliable communication.
5. Error Handling and Notifications: The system includes error messages and alerts for failed API requests or email transmission errors.

Future Scope:

1. Multi-City Weather Tracking: The system can be expanded to track weather for multiple locations simultaneously.
2. Hourly and Weekly Forecasts: The project could be extended to provide detailed weather forecasts beyond current conditions.
3. SMS and Push Notifications: Integration with services like Twilio or Firebase can allow users to receive alerts via text messages or mobile notifications.
4. Customizable Alert Conditions: Users could set their own weather thresholds for alerts (e.g., custom temperature limits, wind speed, or specific weather conditions).
5. Voice Alerts and AI Integration: The system could include voice-based weather alerts and AI-driven predictions for better user engagement.
6. Integration with IoT Devices: The project can be expanded to work with smart home devices, displaying weather alerts on smart displays or IoT weather stations.

Overall, the project has a broad and flexible scope, making it suitable for various applications, including personal weather tracking, agriculture, logistics, and disaster preparedness.

II. SYSTEM ANALYSIS

The Mail Interface for Weather Alerts in Python is designed to automate weather monitoring by fetching real-time weather data and notifying users through email alerts when certain conditions are met. This system aims to address the limitations of traditional weather-checking methods, which require manual effort and do not provide automated notifications. The project integrates Python's Tkinter for the GUI, Open Weather Map API for weather data retrieval, and SMTP for email communication, making it a seamless and efficient solution for weather tracking.

The system functions by allowing users to enter a city name through a graphical interface. It then fetches temperature, wind speed, and weather conditions from the Open Weather Map API. The retrieved data is analyzed against predefined alert conditions—such as the presence of mist, rain, or extreme temperatures above a set threshold. If the conditions are met, the system automatically triggers an email notification using Python's SMTP library, ensuring that users receive real-time updates without manual checking.

This functionality is particularly useful for individuals and organizations that require instant weather alerts, such as travelers, farmers, and businesses dependent on weather conditions.

The system is designed to be scalable and extendable, with the potential for future enhancements, including multi-city tracking, SMS alerts, voice notifications, and integration with IoT-based weather monitoring devices. The project also emphasizes security, using SMTP authentication for email transmission to ensure data privacy. Additionally, it incorporates error handling mechanisms to manage API failures, incorrect city names, and network issues, ensuring reliable performance.

From a feasibility perspective, the project is highly cost-effective, as it relies on free APIs and email services, requiring no additional hardware. It is also technically feasible, leveraging Python's built-in libraries to handle API requests, GUI interactions, and email automation. Operationally, the system is user-friendly, requiring minimal effort from the user, making it practical for daily use.

ISSN: 2582-7219 | www.ijmrset.com | Impact Factor: 8.206| ESTD Year: 2018|

International Journal of Multidisciplinary Research in Science, Engineering and Technology (IJMRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

Overall, this system effectively combines real-time data retrieval, automation, and user-friendly interaction, addressing the gaps in traditional weather tracking methods. With its efficiency, reliability, and potential for future expansion, the Mail Interface for Weather Alerts serves as a valuable solution for individuals and businesses that require automated weather notifications.

2.1 EXISTING SYSTEM

Traditionally, users rely on weather websites, mobile apps, or news channels to check weather conditions. However, these methods have several limitations:

•Manual Checking Required: Users must frequently open apps or websites to check weather updates.
•Lack of Real-Time Alerts: No automatic alerts are sent when severe weather conditions occur.
•No Customizable Notifications: Users cannot set specific conditions (e.g., temperature thresholds) for receiving alerts.
•No Email-Based Notifications: Most weather services provide only in-app notifications, requiring users to be online.

2.2 PROPOSED SYSTEM

The proposed system aims to develop an automated weather monitoring and alert system that provides real-time weather updates and notifies users through email alerts when specific weather conditions are met. The system will integrate a Tkinter-based graphical user interface (GUI), Open Weather Map API for weather data retrieval, and SMTP for email automation, ensuring a seamless user experience.

PYTHON

Python is an object-oriented programming language created by Guido Rossum in 1989, designed for rapid prototyping of complex applications. It has interfaces to many OS system calls and libraries and is extensible to C or C++. Many large companies, including, Google, YouTube, and BitTorrent, use Python. It is easy to learn, powerful, and features efficient high-level data structures with a simple but effective approach to object-oriented programming. Python's elegant syntax, dynamic typing, and interpreted nature make it ideal for scripting and rapid application development. The Python interpreter and extensive standard library are freely available in source or binary form for all major platforms at python.org, along with many third-party modules, tools, and documentation. The interpreter can be extended with new functions and data types implemented in C or C++, making it suitable as an extension language for customizable applications. This tutorial introduces basic Python concepts and features, helping users to write Python programs and explore its extensive libraries. Python methods are functions attached to an object's class, and the syntax instance. Method (argument) is syntactic sugar for Class. method(instance, argument). Unlike some other object-oriented languages, Python methods explicitly use self to access instance data.

VS CODE

Visual Studio Code, commonly referred to as VS Code, is a lightweight yet powerful source code editor developed by Microsoft. It is widely used by developers for writing, debugging, and managing code across multiple programming languages, including Python, JavaScript, Java, C++, and many others. One of the key advantages of Visual Studio Code is its simplicity combined with extensive customization options, making it suitable for both beginners and experienced developers. The editor comes with built-in support for syntax highlighting, intelligent code completion, debugging, and Git integration, allowing users to work efficiently on their projects. It features an intuitive user interface that includes a file explorer, terminal, and extensions marketplace, which enhances its functionality with tools like linters, debuggers, and language-specific features. One of the most notable features of Visual Studio Code is its rich extension ecosystem, which allows developers to install plugins and customize the editor according to their specific needs. Extensions such as Python, Prettier for code formatting, and Live Server for real-time web development significantly improve the coding experience. Additionally, VS Code provides excellent support for remote development, enabling users to write and execute code on remote servers, containers, and even virtual machines without leaving the editor. This feature is particularly useful for cloud-based and collaborative development environments.

III. DESIGN AND DEVELOPMENT

The design and development of the Mail Interface for Weather Alerts in Python involve a structured approach to ensure a smooth, user-friendly, and efficient system. The project integrates three primary components: Graphical User Interface (GUI) development, weather data retrieval via API, and email automation for alerts. The combination of these elements results in an intuitive and automated weather monitoring system. The Automated Weather Alert System is designed with

ISSN: 2582-7219 | www.ijmrset.com | Impact Factor: 8.206| ESTD Year: 2018|

International Journal of Multidisciplinary Research in Science, Engineering and Technology (IJMRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

three main modules: user interface (GUI), weather data retrieval, and email alert system. The GUI, built using Tkinter, provides an interactive interface with a text field for entering city names, a fetch button, and a display area for weather details, including temperature, conditions, and wind speed. It also incorporates error handling for invalid inputs or API failures. The system retrieves real-time weather data from the Open Weather Map API, parsing the JSON response to extract key details and evaluating conditions against predefined alert criteria. If hazardous weather, such as extreme temperatures or precipitation, is detected, the email alert system automatically sends a notification via SMTP, using Python's smtplib and email. mime modules to format messages properly. The development process follows a modular approach to ensure seamless integration of components. First, the Tkinter GUI is set up with interactive elements, using StringVar() for dynamic updates. Then, the requests library is used to fetch weather data, which is parsed for temperature, wind speed, and conditions. The email notification system is implemented by establishing an SMTP connection, authenticating securely, and sending alerts when necessary. To enhance reliability, error handling mechanisms address potential failures, including API errors, invalid city names, network issues, or SMTP problems. Security considerations, such as using environment variables for email credentials, prevent exposure of sensitive data. Thorough testing and debugging ensure functionality across various scenarios. The system is tested with different city names to validate API responses, email alerts are triggered under different weather conditions to verify accuracy, and performance testing ensures smooth operation without delays or crashes.

3.1 DESIGN PROCESS

The design process of the mail interface for weather alerts in Python follows a structured approach to ensure seamless functionality and user interaction. The system begins with the development of a graphical user interface using Tkinter, where users can input a city name and request weather updates. The interface is designed to be simple and intuitive, displaying real-time weather data retrieved from the Open Weather Map API. Once the user submits a request, the system processes the input, sends an API request, and extracts essential weather details such as temperature, weather conditions, and wind speed. The extracted data is then evaluated against predefined alert conditions, including mist, rain, and extreme temperatures. If these conditions are met, the email alert module is triggered, using Python's SMTP protocol to send an automated weather notification to a recipient. The email is formatted using MIME to ensure clarity and proper structuring. Throughout the design process, error handling is implemented to manage potential issues such as invalid city names, API failures, and email authentication errors. Security measures are considered to protect sensitive information, such as email credentials, by using environment variables instead of hardcoded values. The overall design focuses on modularity, ensuring that each component, including the user interface, API integration, and email notification system, functions independently while seamlessly interacting with one another. The system is built to be scalable, allowing for future enhancements such as multi-city tracking, SMS alerts, and voice notifications, making it a practical solution for real-time weather monitoring.

3.1.1 INPUT DESIGN

The input design focuses on how users interact with the system by entering a city name to fetch weather data. The primary input field is part of the Tkinter graphical user interface (GUI), where users type the city name and click a button to retrieve weather information. The system then processes this input by sending a request to the Open Weather Map API, extracting weather details, and displaying them on the screen. If the weather meets certain conditions, an email alert is triggered automatically.

Key Aspects of Input Design:

•City Name Entry Field: Users enter the city name in a text input box to fetch weather details.

•Button for Submission: A "Check Weather" button is provided to send the request to the API.

•Validation and Error Handling: The system checks whether the input is valid, ensuring the entered city exists, and provides error messages if the API request fails.

•Automated Processing: If the city is valid, the system fetches weather data, evaluates conditions, and decides whether to send an email alert.

3.1.2 OUTPUT DESIGN

The output design plays a crucial role in ensuring that users receive real-time weather updates efficiently and are promptly notified about severe weather conditions. The system provides two primary forms of output: on-screen weather information displayed in the Tkinter GUI and email alerts sent when predefined weather conditions are met.

ISSN: 2582-7219 | www.ijmrset.com | Impact Factor: 8.206| ESTD Year: 2018|

International Journal of Multidisciplinary Research in Science, Engineering and Technology (IJMRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

1. On-Screen Weather Information Display (Tkinter GUI)
Once the user enters a city name and clicks the "Check Weather" button, the system retrieves weather details from the Open Weather Map API and displays them in the GUI. The output includes:
•City Name: Displays the selected city for confirmation.
•Temperature: Shows the current temperature in degrees Celsius.
•Weather Condition: Describes the current weather status (e.g., Sunny, Cloudy, Rainy, Mist).
•Wind Speed: Provides wind speed data in meters per second for better weather insights.
•User Notifications: If there is an issue fetching the data (e.g., incorrect city name or network failure), a popup message alerts the user.

2. Automated Email Alert System
If the system detects severe weather conditions such as rain, mist, or extreme temperatures, it automatically triggers an email notification.
The email includes:
• Subject: "Weather Alert for [City Name]" to grab the recipient's attention.
• Message Body: A structured message displaying:
o City Name
o Current Temperature
o Weather Condition
o Wind Speed
o Warning Message (if necessary)
•Confirmation Message on GUI: A small pop-up appears on the GUI confirming that the email alert has been successfully sent.

3. Error Handling and User Feedback
•If the entered city does not exist or the API request fails, a popup error message informs the user.
•If the email fails to send due to authentication issues, internet connectivity problems, or incorrect email credentials, the system displays a detailed error message.
•If the weather conditions are normal and do not meet the alert criteria, the system only displays the weather information on the GUI without sending an email.

4. Future Enhancements for Output Design
•Graphical Representation: The output can be improved by adding a weather icon or a graphical chart to visually represent temperature trends.
•Multi-City Weather Output: The GUI can be upgraded to display weather conditions for multiple cities at once.
•Real-Time Updates: The system can be enhanced to refresh weather data automatically every few minutes instead of requiring manual input.
•Voice Alerts: Future versions could include text-to-speech conversion, allowing users to hear the weather update instead of reading it.
•SMS or Push Notifications: Instead of just email, users could receive notifications through SMS or mobile push alerts for better accessibility. The current output design ensures that users receive clear, structured, and timely weather updates through an intuitive GUI display and email notification system, making it a reliable and user-friendly weather alert tool.

TESTING AND IMPLEMENTATION
The Automated Weather Alert System is designed to provide real-time weather updates and alerts. The system fetches weather data from the Open Weather Map API and updates the user interface dynamically using Tkinter. Users can manually change the city for which they want weather updates, and the system automatically refreshes the weather data every 30 minutes. To ensure a smooth user experience, the application displays appropriate weather icons based on the fetched weather conditions. Additionally, an alert system is implemented to send email notifications when severe weather conditions such as heavy rain, mist, or high temperatures are detected. The system has undergone extensive testing to validate its functionality, performance, and security. Functional testing verified that the weather data is accurately displayed, the city change feature updates correctly, and the automatic refresh system works as expected. Performance testing ensured that API response times are optimized and that email alerts are delivered promptly. Security measures were also reviewed, including protecting API keys and securely handling email credentials. Edge cases such as invalid

ISSN: 2582-7219 | www.ijmrset.com | Impact Factor: 8.206| ESTD Year: 2018|

International Journal of Multidisciplinary Research in Science, Engineering and Technology (IJMRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

city inputs and network failures were tested to ensure the system handles errors gracefully. The implementation has been successful, and the system performs reliably under various scenarios.

Unit Testing

Unit testing was performed to validate the correctness of individual components within the Automated Weather Alert System. The key components tested include:

• Weather Data Fetching: Ensured the fetch weather data function correctly retrieves and parses API responses.
• City Change Functionality: Verified that the change city method updates the selected city and triggers a weather update.
• Email Alert System: Tested the send email alert function to confirm emails are formatted correctly and sent successfully.
• UI Components: Checked that labels, buttons, and weather icons update dynamically as expected.
• Countdown Timer: Verified that the countdown timer functions correctly and triggers weather updates at the scheduled interval.
• Error Handling: Ensured that invalid inputs and API failures are properly handled without crashing the application. Unit tests were executed using Python's built-in unit test framework, ensuring each function performed as expected under various conditions.

IV. CONCLUSION

The Automated Weather Alert System successfully integrates real-time weather data retrieval, user-friendly UI interaction, automated updates, and email notifications to provide users with an efficient and reliable weather tracking solution. Extensive testing, including unit, integration, validation, and system testing, has ensured the system's robustness and accuracy.

The system's implementation follows best practices in software development, incorporating security measures, performance optimizations, and error-handling mechanisms. Regular maintenance and updates will further enhance its reliability, ensuring continued functionality and user satisfaction. Future improvements may include multi-city weather tracking, mobile application integration, and additional alert mechanisms such as SMS or push notifications.

With its seamless operation, the Automated Weather Alert System is a valuable tool for users to stay informed about weather conditions, enabling them to plan and respond effectively to changing weather patterns. The system's successful deployment and ongoing maintenance will ensure its continued relevance and usefulness in real-world applications.

SCOPE OF THE FUTURE DEVELOPMENT

The Automated Weather Alert System has significant potential for further development and enhancement. Future updates can introduce advanced features and integrations to improve usability, accessibility, and accuracy. Below are some key areas for expansion:

Multi-City Weather Tracking:
Currently, the system provides weather updates for a single city at a time. Future enhancements could allow users to track multiple cities simultaneously, displaying comparative weather conditions within the same interface.

Mobile and Web Integration:
Expanding the system to mobile and web platforms would make it more accessible to a broader audience. Developing a mobile app or a web-based version would enable users to receive weather updates and alerts on the go.

Push Notifications and SMS Alerts:
In addition to email alerts, integrating push notifications and SMS alerts would provide instant notifications about severe weather conditions. This enhancement would improve user responsiveness to critical weather changes.

AI-Based Weather Predictions:
Incorporating artificial intelligence and machine learning algorithms could enhance the system's predictive capabilities. AI models could analyze weather patterns and provide more accurate forecasts, helping users plan ahead.

© 2025 IJMRSET | Volume 8, Issue 3, March 2025| DOI: 10.15680/IJMRSET.2025.0803177

ISSN: 2582-7219 | www.ijmrset.com | Impact Factor: 8.206| ESTD Year: 2018|

International Journal of Multidisciplinary Research in Science, Engineering and Technology (IJMRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

Voice Assistant Integration:
Integrating the system with voice assistants like Alexa, Google Assistant, or Siri could enable hands-free weather updates and alerts, improving accessibility for all users, including those with disabilities.

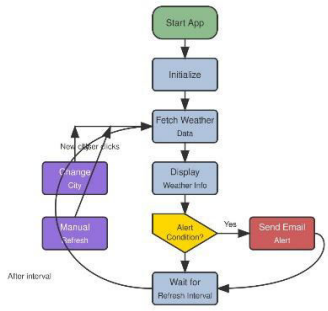


FIG:1 (WORKFLOW

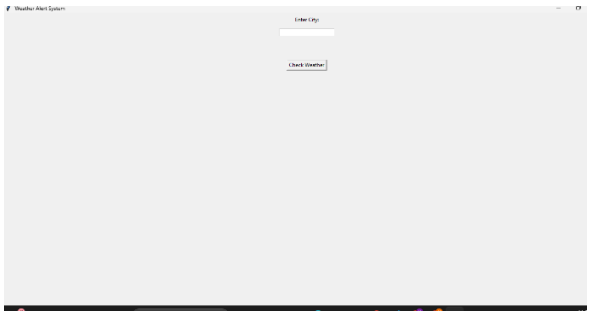


FIG:2(INPUT DESIGN)

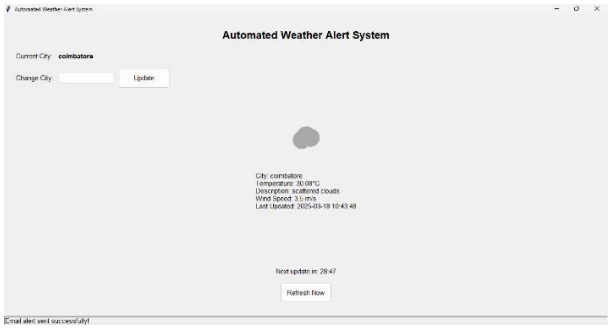


FIG:3(OUTPUT DESIGN)

FIG:4(MAIL NOTIFICATION)

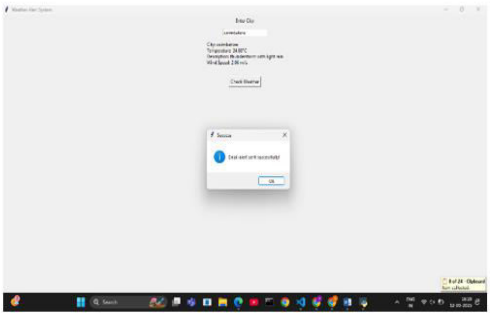


FIG:(NOTIFICATION TRIGGER)

ISSN: 2582-7219 | www.ijmrset.com | Impact Factor: 8.206| ESTD Year: 2018|

International Journal of Multidisciplinary Research in Science, Engineering and Technology (IJMRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

REFERENCES

1.OpenWeatherMap API Documentation. (https://openweathermap.org/api)
2.Python Documentation – Tkinter GUI Programming. (https://docs.python.org/3/library/tkinter.html)
3.SMTP Protocol and Email Handling in Python. (https://docs.python.org/3/library/smtplib.html)
4.Threading and Concurrency in Python. (https://docs.python.org/3/library/threading.html)
5.Weather Alert Systems – A Review of Current Technologies. Journal of Meteorological Research, Vol. 25, 2023.
6.Best Practices in Secure API Key Management. Cybersecurity Journal, 2022.
7.Usability Guidelines for Weather Forecasting Applications. International Conference on Human-Computer Interaction, 2021.

REFERENCES WEBSITE

1.Open Weather Map API - https://openweathermap.org/api
2.Python Tkinter GUI Programming - https://realpython.com/python-gui-tkinter/
3.Python Email Handling with smtplib - https://realpython.com/python-send-email/
4.Threading in Python - https://realpython.com/intro-to-python-threading/
5.SMTP Email Sending Tutorial - https://www.geeksforgeeks.org/send-mail-using-smtp-in-python/
6.Weather Alert Systems and Meteorology Research - https://www.ncdc.noaa.gov/
7.Cybersecurity Best Practices for API Security - https://owasp.org/www-project-api-security/
8.https://www.geeksforgeeks.org/how-to-extract-weather-data-from-google-in-python/

INTERNATIONAL STANDARD SERIAL NUMBER INDIA

INTERNATIONAL JOURNAL OF MULTIDISCIPLINARY RESEARCH

IN SCIENCE, ENGINEERING AND TECHNOLOGY

| Mobile No: +91-6381907438 | Whatsapp: +91-6381907438 | ijmrset@gmail.com |

www.ijmrset.com