

**Nikita H**

 Nikita-H

 Nikita-Ho7

 [nikita.h.0745@gmail.com](mailto:nikita.h.0745@gmail.com)

# Real-Time Weather Monitoring Dashboard with Alerts

# Project Overview

The **Real-Time Weather Monitoring Dashboard with Alerts** is a full-stack application designed to collect, display, and monitor weather data in real-time with additional alert system to alert the user(e.g., high temperature, strong winds). It features dynamic visualizations, user authentication, alert mechanisms, and seamless integration with cloud infrastructure. This project showcases the practical use of cloud computing, third-party APIs, and data analytics for a responsive and scalable weather monitoring solution.

## Key Features

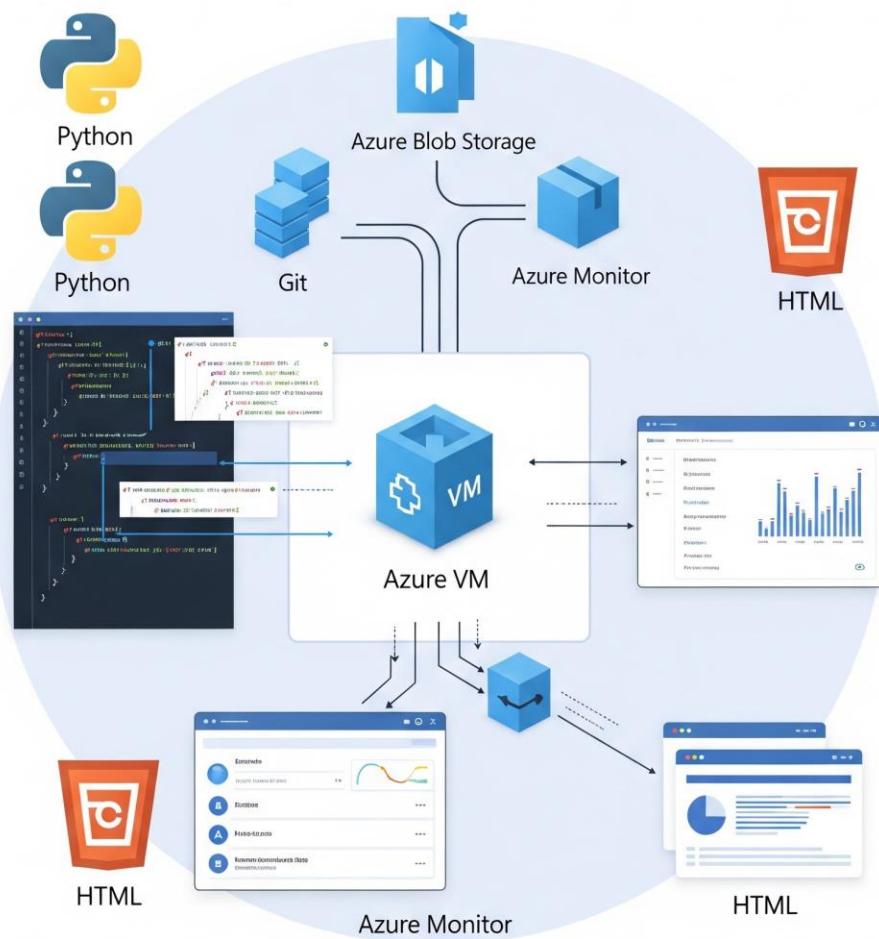
- **Real-time Weather Data:** Fetches live data using the OpenWeatherMap API.
- **User Authentication:** Secure login to access dashboard features.
- **Data Visualization:** Displays weather metrics like temperature, humidity, and wind speed using interactive charts.
- **Alert System:** Automatically triggers alerts when extreme or abnormal weather conditions are detected.
- **Cloud Deployment:** Hosted on a Microsoft Azure Virtual Machine with Azure Monitor enabled for performance tracking and logging.

## Technologies Used

- **Frontend:** HTML, CSS, JavaScript.
- **Backend:** Python (Flask).
- **APIs:** OpenWeatherMap.
- **Cloud:** Microsoft Azure (Virtual Machine, Azure Monitor, Blob Storage).

## Modules and Functionality

1. **Data Collection:** Periodically fetches live weather data via OpenWeatherMap API.
2. **Dashboard UI:** Charts dynamically update using Chart.js to reflect real-time changes.
3. **User Authentication:** Basic login system ensures secure access to the dashboard.
4. **Alerts:** Triggers visual and/or email alerts for values that exceed defined thresholds (e.g., high temperature, strong winds).
5. **Deployment:** Flask app is deployed on an Azure Virtual Machine with system monitoring via Azure Monitor.



# Project Timeline

Phase	Task	Duration
Phase 1	API Research and Planning	2 days
Phase 2	Frontend UI Development	3 days
Phase 3	Flask Backend Setup	2 days
Phase 4	Azure VM + Monitor Setup	2 days
Phase 5	Testing and Final Report	1 day

## Challenges and Learnings

- Learned cloud deployment using Microsoft Azure.
- Faced API rate limit challenges and solved using caching.
- Understood Azure Monitor's diagnostic features.
- Improved frontend chart integration with asynchronous fetch.

The screenshot shows a GitHub repository page for 'Weather-Monitoring'. The repository is public and has 5 commits. The commit list includes updates to README.md, .pycache\_, app, templates, .env, .env.example, .gitignore, README.md, azure\_blob.py, main.py, and requirements.txt. The repository has 0 stars, 0 forks, and 0 packages published. It also has 2 contributors: Harshad03-GN and Harshad-1207, and 5 deployments. A preview of the dashboard is shown at the bottom.

**Weather-Monitoring** (Public)

main · 1 Branch · 0 Tags

Go to file Add file · Code

Harshad03-GN	Update README.md	2bc453d · 20 hours ago	5 Commits
__pycache__	ioefh	yesterday	
app	Initial commit	2 days ago	
templates	Initial commit	2 days ago	
.env	Initial commit	2 days ago	
.env.example	init	2 days ago	
.gitignore	init	2 days ago	
README.md	Update README.md	20 hours ago	
azure_blob.py	ioefh	yesterday	
main.py	Initial commit	2 days ago	
requirements.txt	Initial commit	2 days ago	

**About**

Weather-Monitoring

- Readme
- Activity
- 0 stars
- 0 watching
- 0 forks

**Releases**

No releases published

[Create a new release](#)

**Packages**

No packages published

[Publish your first package](#)

**Contributors** 2

Harshad03-GN

Harshad-1207

**Deployments** 5

github-pages 20 hours ago

**Real-Time Weather Monitoring Dashboard with Alerts**

## ← Microsoft Azure

### weather-monitor

Overview

Running

Start

Restart

Stop

Connect

Delete

Resource group : weather-monitor-rg

Location

Ubuntu

Size : Central India

DNS

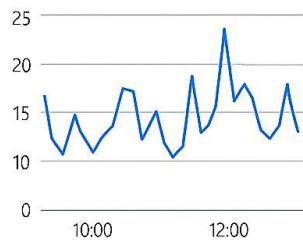
52.183.88.99

Public IP address : 52.183.88.99

Subscription

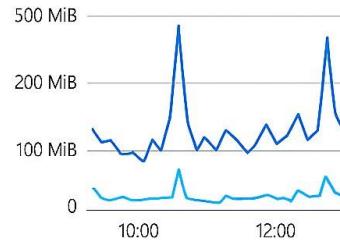
cf0336ce-8ff3-4f80-90df-a1231f22  
4be8

CPU utilization (%)



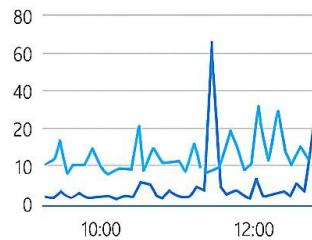
CPU utilization (%)  
15

Network (bytes)



In  
560vb

Disk operations/sec



Read  
0  
Write  
00

Microsoft Azure

Home > Monitor

Monitor | Logs

Logs Metrics Alerts >

Temperature Over Time

A line chart showing temperature over a 12-hour period. The Y-axis ranges from 10 to 25 in increments of 5. The X-axis shows times from 04:00 to 12:00. The temperature starts at ~12°C at 04:00, rises steadily to a peak of ~22°C at 10:00, and then gradually declines to ~18°C by 12:00.

Time	Temperature (°C)
04:00	12
06:00	15
08:00	18
10:00	22
12:00	18

Alert Count

A bar chart showing the count of alerts over a four-hour period. The X-axis shows times from 07:00 to 11:00. Two alerts are recorded: one at 09:00 with a count of 1, and another at 10:00 with a count of 2.

Time	Alert Count
09:00	1
10:00	2

Humidity (%)

61

Learn more about Log Analytics

Quickly retrieve, consolidate, and analyze all data collected into Azure Monitoring Logs. Save your queries for future use, pin query results to Azure Dashboards, and

Humidity Over Time

A line chart showing humidity over a 12-hour period. The Y-axis ranges from 40 to 80 in increments of 20. The X-axis shows times from 04:00 to 12:00. The humidity starts at ~80% at 04:00 and gradually declines to ~40% by 12:00.

Time	Humidity (%)
04:00	80
06:00	60
08:00	50
10:00	45
12:00	40

Query Language

Log Analytics uses a version of the Kusto Query language (KQL) that is suitable for both simple and advanced log queries using functionality such as aggregations, joins, and smart analytics.

## Conclusion

This project provided a complete development and deployment experience in real-world cloud and DevOps practices. It showcased how weather data can be used in meaningful, user-friendly ways through real-time monitoring, dynamic visualization, and automated alert mechanisms. The alert system was a critical component, designed to detect abnormal weather conditions—such as extreme temperature, high humidity, or strong winds—and immediately notify users through visual warnings on the dashboard. This proactive approach ensured timely awareness and demonstrated how real-time data processing can enhance decision-making and safety in practical applications.

## Acknowledgment

I would like to acknowledge that this cloud-based application project was conceptualized, developed, and deployed independently. The experience allowed me to explore various technologies, apply cloud computing principles, and gain hands-on practice with real-world development and DevOps workflows.

Connect:  [Nikita-Ho7](#)