



# PANDORA

INFRA MONITERING TOOL

PURANJOT SINGH

TANISH KACKRIA

PRANAV MALHOTRA

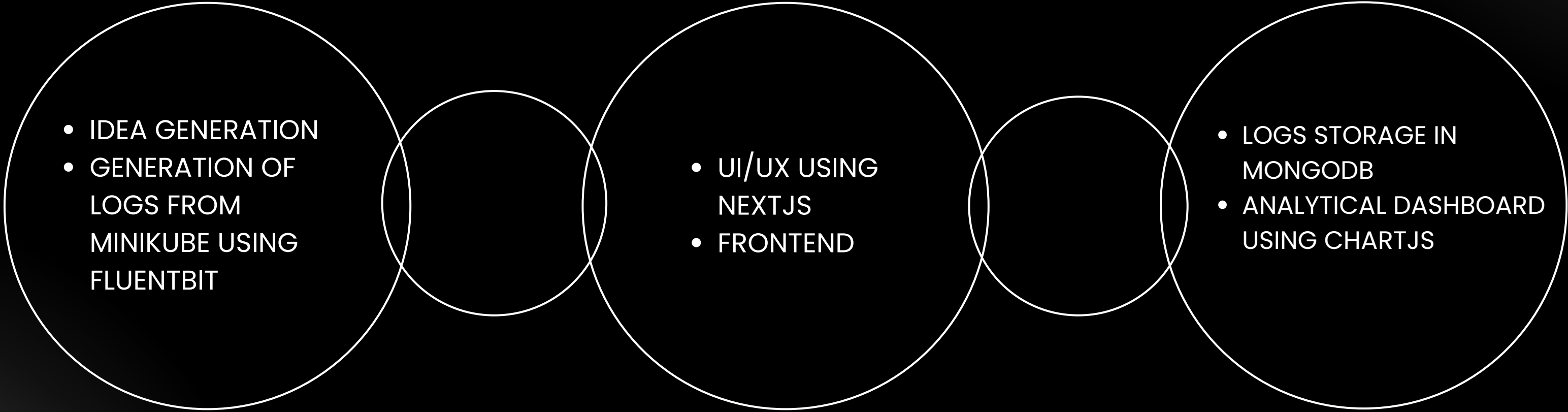
DEVANSH AGGARWAL

SIDDHARTH CHAUHAN

# PROBLEM DESCRIPTION

Team Pandora successfully developed an innovative cloud observability platform as part of the STGI Hackathon. The project focused on building a robust solution to monitor applications deployed in a Kubernetes cluster, specifically targeting two databases—PostgreSQL and MySQL.

# ROADMAP

- 
- IDEA GENERATION
  - GENERATION OF LOGS FROM MINIKUBE USING FLUENTBIT

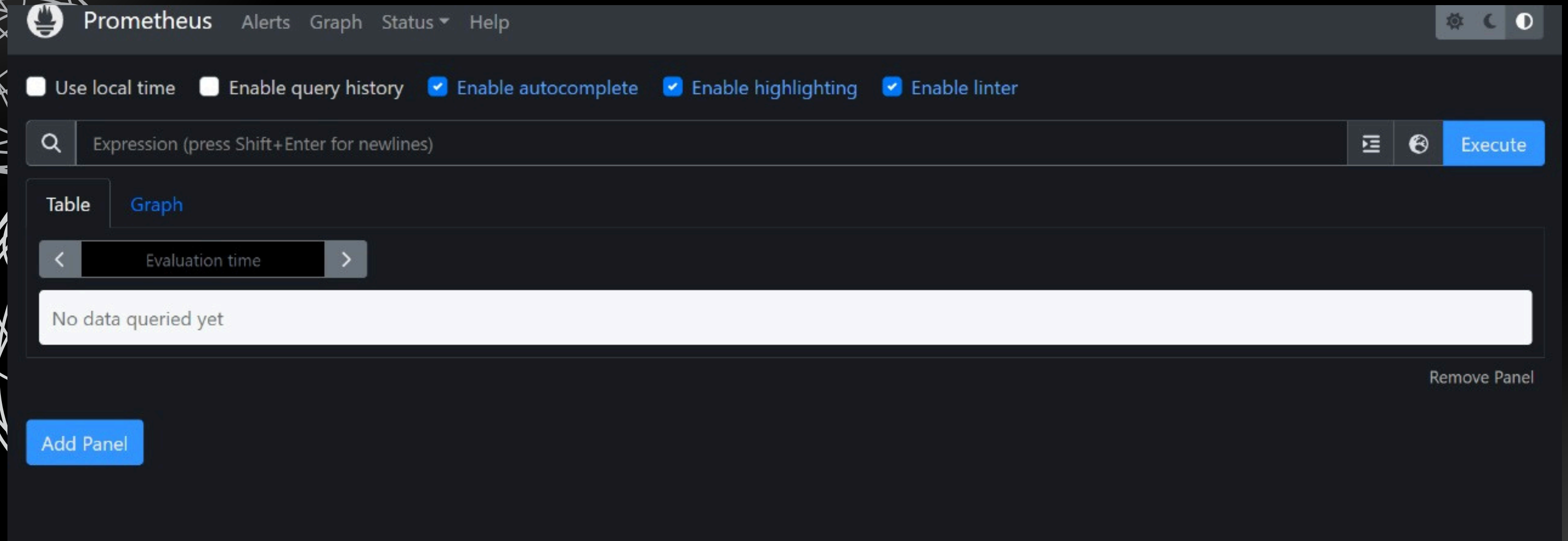
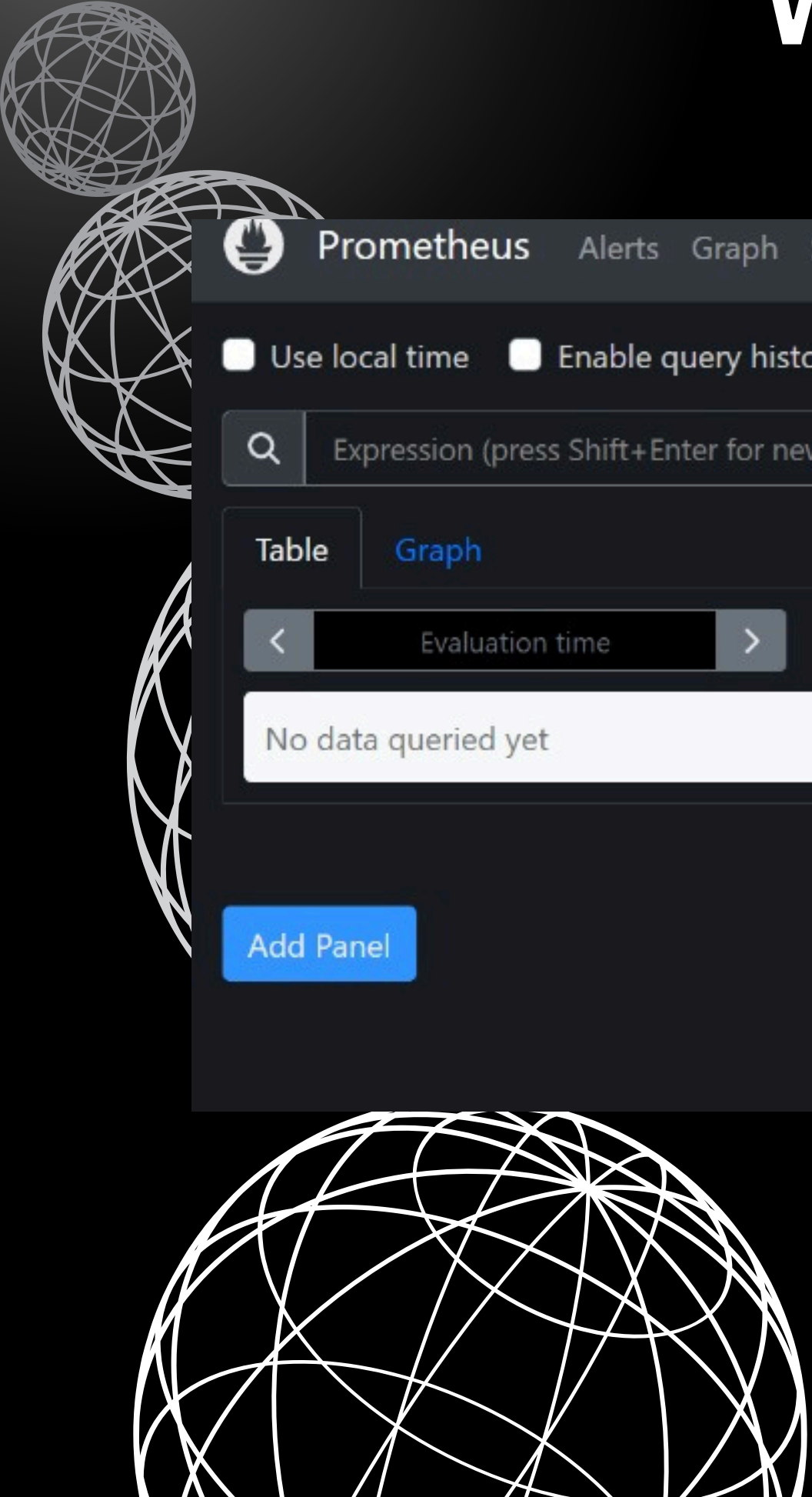
- UI/UX USING NEXTJS
- FRONTEND

- LOGS STORAGE IN MONGODB
- ANALYTICAL DASHBOARD USING CHARTJS

# LOGS FROM FLUENTBIT

```
ail.0 > output=es.0 (out_id=0)
PS E:\STGI\pandora\deployment-env> kubectl logs -l "app.kubernetes.io/name=fluent-bit" -n kube-system
[2024/09/28 16:16:33] [ warn] [net] getaddrinfo(host='elasticsearch-master', err=4): Domain name not found
[2024/09/28 16:16:33] [ warn] [engine] failed to flush chunk '1-1727540103.679228328.flb', retry in 116 seconds: task_id=178, input=t
ail.0 > output=es.0 (out_id=0)
[2024/09/28 16:16:34] [ warn] [net] getaddrinfo(host='elasticsearch-master', err=4): Domain name not found
[2024/09/28 16:16:34] [ warn] [engine] failed to flush chunk '1-1727540039.494379526.flb', retry in 256 seconds: task_id=48, input=ta
il.0 > output=es.0 (out_id=0)
[2024/09/28 16:16:34] [ warn] [net] getaddrinfo(host='elasticsearch-master', err=4): Domain name not found
[2024/09/28 16:16:34] [ warn] [net] getaddrinfo(host='elasticsearch-master', err=4): Domain name not found
[2024/09/28 16:16:34] [ warn] [net] getaddrinfo(host='elasticsearch-master', err=4): Domain name not found
[2024/09/28 16:16:34] [ warn] [engine] failed to flush chunk '1-1727540026.560811858.flb', retry in 132 seconds: task_id=21, input=ta
il.0 > output=es.0 (out_id=0)
[2024/09/28 16:16:34] [ warn] [engine] failed to flush chunk '1-1727540077.640485819.flb', retry in 113 seconds: task_id=126, input=t
ail.0 > output=es.0 (out_id=0)
[2024/09/28 16:16:34] [ warn] [engine] failed to flush chunk '1-1727540099.595493610.flb', retry in 110 seconds: task_id=169, input=t
ail.0 > output=es.0 (out_id=0)
PS E:\STGI\pandora\deployment-env> |
```

# WORKING PROMETHEUS



The screenshot shows the Prometheus web interface with a dark theme. The top navigation bar includes the Prometheus logo, 'Prometheus', 'Alerts', 'Graph', 'Status', and 'Help'. On the right of the bar are icons for settings, a moon (dark mode), and a sun (light mode). Below the navigation bar, there are five toggle switches: 'Use local time' (disabled), 'Enable query history' (disabled), 'Enable autocomplete' (enabled), 'Enable highlighting' (enabled), and 'Enable linter' (enabled). A search bar contains the placeholder text 'Expression (press Shift+Enter for newlines)'. To the right of the search bar are icons for a menu and a globe, followed by a blue 'Execute' button. Below the search bar, there are two tabs: 'Table' and 'Graph', with 'Graph' being the active tab. Under the 'Graph' tab, there is a time range selector showing '<' and '>' buttons with 'Evaluation time' in the center. Below this is a large white rectangular area with the text 'No data queried yet'. In the bottom right corner of the panel area is a 'Remove Panel' link. At the bottom left of the interface is a blue 'Add Panel' button.

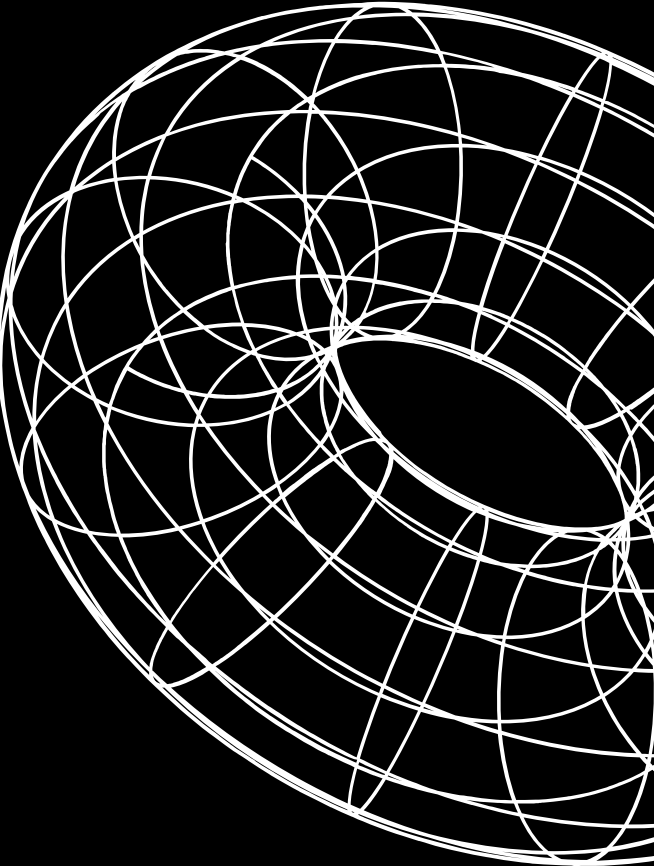




# FEATURES

## Real-Time Alerting System

```
1 import pandas as pd
2 import numpy as np
3 import matplotlib.pyplot as plt
4
5 # Step 1: Generate sample log data simulating errors and traffic
6 np.random.seed(42) # For reproducibility
7 dates = pd.date_range(start='2023-09-01', periods=100, freq='D')
8
9 # Simulate normal traffic data and error counts
10 traffic_data = np.random.poisson(lam=200, size=len(dates)) # Normal traffic
11 error_data = np.random.poisson(lam=5, size=len(dates)) # Base errors
12
13 # Introduce some anomalies in traffic and errors
14 error_data[10] += 25 # Spike in errors
15 error_data[50] += 15 # Another spike
16 traffic_data[20] += 1000 # Spike in traffic
17 traffic_data[80] += 800 # Another spike
18
19 # Create DataFrame
20 df = pd.DataFrame({
21     'date': dates,
22     'errors': error_data,
23     'traffic': traffic_data,
24 })
25
26 df.set_index('date', inplace=True)
27
28 # Step 2: Calculate rolling mean and standard deviation for errors and traffic
29 window_size = 7 # 7-day rolling window
30 df['error_rolling_mean'] = df['errors'].rolling(window=window_size).mean()
31 df['error_rolling_std'] = df['errors'].rolling(window=window_size).std()
32
33 df['traffic_rolling_mean'] = df['traffic'].rolling(window=window_size).mean()
34 df['traffic_rolling_std'] = df['traffic'].rolling(window=window_size).std()
35
36 # Step 3: Define anomalies for errors and traffic
37 threshold = 2 # Number of standard deviations
38 df['error_anomaly'] = (df['errors'] > df['error_rolling_mean'] + threshold * df['error_rolling_std'])
39 df['traffic_anomaly'] = (df['traffic'] > df['traffic_rolling_mean'] + threshold * df['traffic_rolling_std'])
40
```



# Log Anomaly Detection

```
1  import smtplib
2  from email.mime.text import MIMEText
3  from email.mime.multipart import MIMEMultipart
4
5  # Replace with your email server details and login credentials
6  smtp_server = "smtp.gmail.com"
7  smtp_port = 587
8  sender_email = "your_email@gmail.com"
9  sender_password = "your_password"
10
11 # Function to send email alert
12 def send_email_alert(subject, body, recipient_email):
13     # Create the email headers and content
14     msg = MIMEMultipart()
15     msg['From'] = sender_email
16     msg['To'] = recipient_email
17     msg['Subject'] = subject
18     msg.attach(MIMEText(body, 'plain'))
19
20     # Setup the SMTP server
21     server = smtplib.SMTP(smtp_server, smtp_port)
22     server.starttls() # Secure the connection
23     server.login(sender_email, sender_password) # Login to your email
24
25     # Send the email
26     server.sendmail(sender_email, recipient_email, msg.as_string())
27     server.quit()
28
29     print(f"Email alert sent to {recipient_email}")
30
31 # Example usage: Send an email when threshold is exceeded
32 def check_thresholds_and_alert_email(metrics, recipient_email):
33     alerts = []
34
35     if metrics['cpu_usage'] > 85:
36         alerts.append("High CPU Usage Alert!")
37     if metrics['memory_usage'] > 80:
38         alerts.append("High Memory Usage Alert!")
39     if metrics['error_rate'] > 5:
40         alerts.append("High Error Rate Alert!")
```

# OUR SIGNUP PAGE

## Welcome to Pandoras

Sign up to start your journey

Username

Email

Password

Sign Up

Already a member? [Sign in](#)



# OUR LANDING PAGE

## INFRA MONITORING TOOL

TEAM PANDORA SUCCESSFULLY DEVELOPED AN INNOVATIVE CLOUD OBSERVABILITY PLATFORM AS PART OF THE STGI HACKATHON. THE PROJECT FOCUSED ON BUILDING A ROBUST SOLUTION TO MONITOR APPLICATIONS DEPLOYED IN A KUBERNETES CLUSTER, SPECIFICALLY TARGETING TWO DATABASES—POSTGRESQL AND MYSQL.

## MEET OUR TEAM:

- 1) PURANJOT SINGH
- 2) TANISH KACKRIA
- 3) PRANAV MALHOTRA
- 4) DEVANSH AGGARWAL
- 5) SIDDHARTH CHAUHAN

About our Project ▾

The background is a dark gradient with intricate white line art. On the left, a series of concentric, wavy lines form a shape reminiscent of a stylized leaf or a wing, extending from the bottom towards the center. On the right, another set of similar wavy lines curves upwards from the top right corner towards the center. The overall effect is a sense of fluid motion and organic form.

**THANK YOU**