Detailed Report on Clickstream Data Pipeline Implementation

Hayat Sikandar Dahraj

01/12/2024

Abstract

This report outlines the implementation of a clickstream data pipeline using Apache logs and AWS CloudWatch. The pipeline was designed to capture detailed visitor interactions on a café website, process them into structured JSON format, and use AWS CloudWatch for log collection, storage, and analytics. It also includes the integration of geolocation data and the creation of interactive visualizations for better business decision-making.

1 Introduction

The objective of this project was to create a sophisticated data pipeline for tracking customer interactions on a café website. Initially, Apache web server logs were in a basic text format. Our goal was to enhance these logs by converting them into a structured JSON format to facilitate modern data analysis using AWS CloudWatch. This change was aimed at improving the understanding of customer behavior and providing the café owner with valuable insights into website traffic.

2 Apache Web Server Configuration

2.1 Log Format Modification

The first step was modifying the Apache configuration file `httpd.conf`. By default, Apache records logs in the Common Log Format (CLF). We changed this format to JSON, which is more suitable for structured data analysis.

2.2 Log Directory Setup

To organize our logs, we created specific directories for access and error logs using the following commands:

```
sudo mkdir -p /var/log/www/access
sudo mkdir -p /var/log/www/error
```

This step helped ensure that logs were separated into distinct folders for better organization.

2.3 Real-Time Log Monitoring

To monitor logs in real-time, we used the following command:

```
sudo tail -f /var/log/www/access/access_log
```

This command allowed us to view live log entries as visitors interacted with the website.

3 CloudWatch Setup

3.1 CloudWatch Agent Installation

We installed and configured the AWS CloudWatch agent to collect logs and send them to CloudWatch for further analysis. The installation was completed using the following commands:

```
sudo yum update -y
sudo yum install -y amazon-cloudwatch-agent
```

3.2 CloudWatch Agent Configuration

The agent was configured with a specific settings file that we downloaded and placed in the appropriate directory:

```
wget https://aws-tc-largeobjects.s3.us-west-2.amazonaws.com/CUR-TF-200-
sudo mv config.json /opt/aws/amazon-cloudwatch-agent/bin/
```

3.3 Log Group Configuration

Once the agent was configured, we ensured that the logs were properly organized into two distinct log groups within CloudWatch: `apache/access` and `apache/error`. These groups were set with retention policies to store logs for 180 days.

4 Simulated Data and Testing

We tested the system by replacing the live Apache logs with simulated data. This allowed us to validate the data pipeline's functionality. We used commands such as:

```
cat samplelogs/access_log.log | head
python -m json.tool
```

These commands helped verify that the logs were in the correct JSON format and were ready for analysis.

5 Geolocation Data Integration

5.1 Geolocation Log File Setup

We successfully integrated geolocation data into our log files. This new data provided geographical information such as the country, region, city, and coordinates for each visitor interaction. After verifying that the log file contained the necessary geolocation information, we proceeded to replace the existing access log with this new version.

5.2 Geolocation Data Testing

To finalize the process, we deleted the old log group in CloudWatch and restarted the CloudWatch agent to start sending the geolocation-enabled logs to CloudWatch. This step ensured that our geographic analysis was based on the most up-to-date data.

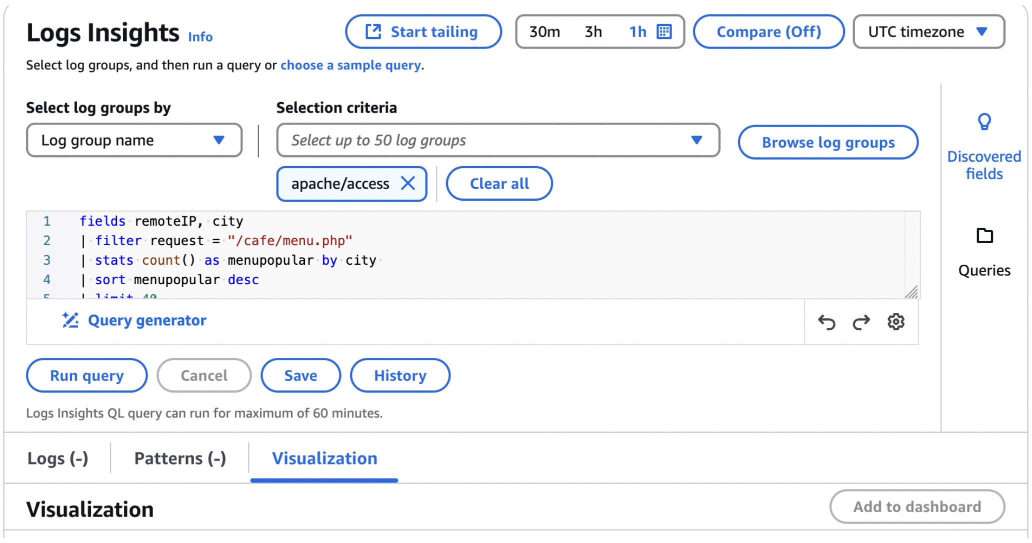
6 Data Visualization in CloudWatch

6.1 Creating Visualizations

We created several visualizations in CloudWatch to track customer behavior. The first visualization was a pie chart displaying the number of menu views by city. The query used for this chart is shown below:

```
fields remoteIP, city
| filter request = "/cafe/menu.php"
| stats count() as menupopular by city
| sort menupopular desc
| limit 10
```

This query filtered the logs for menu page visits and displayed the top 10 cities with the most visits.



Select log groups, and then run a query or choose a sample query.

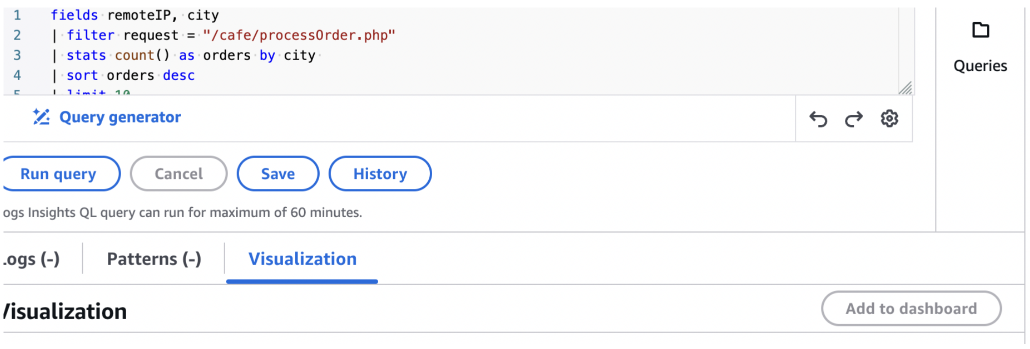
```
fields remoteIP, city
| filter request = "/cafe/menu.php"
| stats count() as menupopular by city
| sort menupopular desc
| limit 40
```

Logs Insights QL query can run for maximum of 60 minutes.

Figure 1: SQL Command for Menu Visitors Query

6.2 Additional Visualizations

We also created visualizations to show regional behavior patterns. The second pie chart, for instance, focused on regions with the highest purchase activities.



```
fields remoteIP, city
| filter request = "/cafe/processOrder.php"
| stats count() as orders by city
| sort orders desc
| limit 10
```

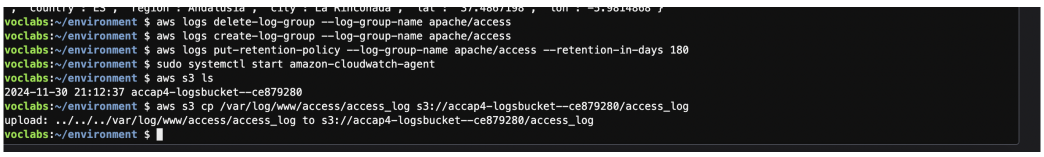
ogs Insights QL query can run for maximum of 60 minutes.

Figure 2: SQL Command for Regional Purchases Query

7 S3 Integration for Data Storage

Once the visualizations were set up, we transferred the log files to an S3 bucket for long-term storage. The transfer was done using the following command:

```
aws s3 cp /var/log/www/access/access_log s3://accap4-logsbucket--ce879
```

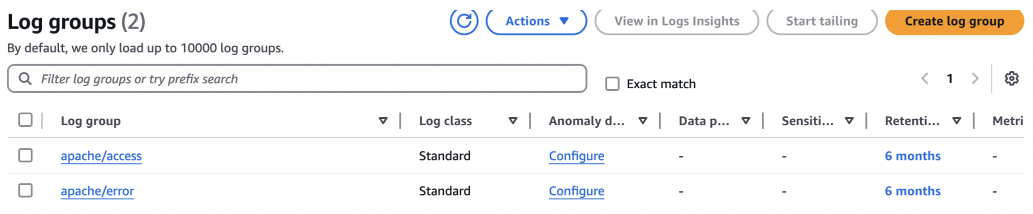


```
, "country":"ES", "region":"Andalusia", "city":"La Rinconada", "lat": "37.4867198", "lon":"-5.9814868"}
voclabs:~/environment $ aws logs delete-log-group --log-group-name apache/access
voclabs:~/environment $ aws logs create-log-group --log-group-name apache/access
voclabs:~/environment $ aws logs put-retention-policy --log-group-name apache/access --retention-in-days 180
voclabs:~/environment $ sudo systemctl start amazon-cloudwatch-agent
voclabs:~/environment $ aws s3 ls
2024-11-30 21:12:37 accap4-logsbucket--ce879280
voclabs:~/environment $ aws s3 cp /var/log/www/access/access_log s3://accap4-logsbucket--ce879280/access_log
upload: ../../../var/log/www/access/access_log to s3://accap4-logsbucket--ce879280/access_log
voclabs:~/environment $
```

Figure 3: Amazon S3 Commands for Log Transfer

8 CloudWatch Logs Overview

We reviewed the CloudWatch log groups to understand the structure and verify data flow. The log groups were set for both access and error logs.

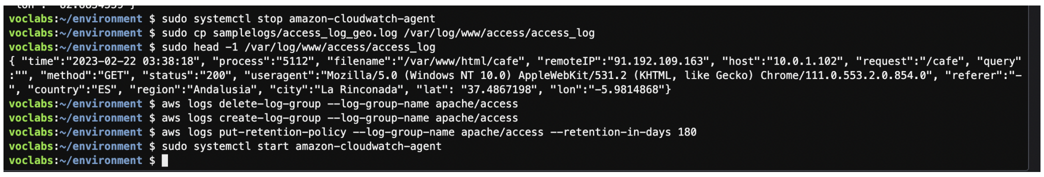

Log groups (2)
Actions
View in Logs Insights
Start tailing
Create log group
By default, we only load up to 10000 log groups.
Filter log groups or try prefix search
Exact match
1

Log group	Log class	Anomaly d...	Data p...	Sensiti...	Retenti...	Metri
apache/access	Standard	Configure	-	-	6 months	-
apache/error	Standard	Configure	-	-	6 months	-

Figure 4: CloudWatch Log Groups Overview

9 Geolocation Data in Logs

The integration of geolocation data was confirmed through the log entries.



```
voclabs:~/environment $ sudo systemctl stop amazon-cloudwatch-agent
voclabs:~/environment $ sudo cp samplelogs/access_log_geo.log /var/log/www/access/access_log
voclabs:~/environment $ sudo head -1 /var/log/www/access/access_log
{ "time":"2023-02-22 03:38:18", "process":"5112", "filename":"/var/www/html/cafe", "remoteIP":"91.192.109.163", "host":"10.0.1.102", "request":"/cafe", "query"
:"", "method":"GET", "status":"200", "useragent":"Mozilla/5.0 (Windows NT 10.0) AppleWebKit/531.2 (KHTML, like Gecko) Chrome/111.0.553.2.0.854.0", "referer":"-
", "country":"ES", "region":"Andalusia", "city":"La Rinconada", "lat": "37.4867198", "lon":"-5.9814868"}
voclabs:~/environment $ aws logs delete-log-group --log-group-name apache/access
voclabs:~/environment $ aws logs create-log-group --log-group-name apache/access
voclabs:~/environment $ aws logs put-retention-policy --log-group-name apache/access --retention-in-days 180
voclabs:~/environment $ sudo systemctl start amazon-cloudwatch-agent
voclabs:~/environment $
```

Figure 5: Preview of Access Logs with Geolocation Data

10 Final Configuration and System Setup

We finalized the configuration and saved all settings to preserve the system state, including the CloudWatch dashboard, log groups, and S3 configurations.

11 City and Region Analysis

We created visualizations to analyze the cities with the most visitors and the regions making the most purchases.

12 Screenshots of CloudWatch Configuration

The final configuration details and screenshots are provided to ensure all settings were saved and accurate.

13 Conclusion

In conclusion, this project successfully transformed Apache web logs into structured JSON format for analysis, integrated geolocation data, and set up CloudWatch for log collection, visualization, and long-term storage. The data pipeline now provides valuable insights into customer behavior, helping the café owner make informed decisions about the business.

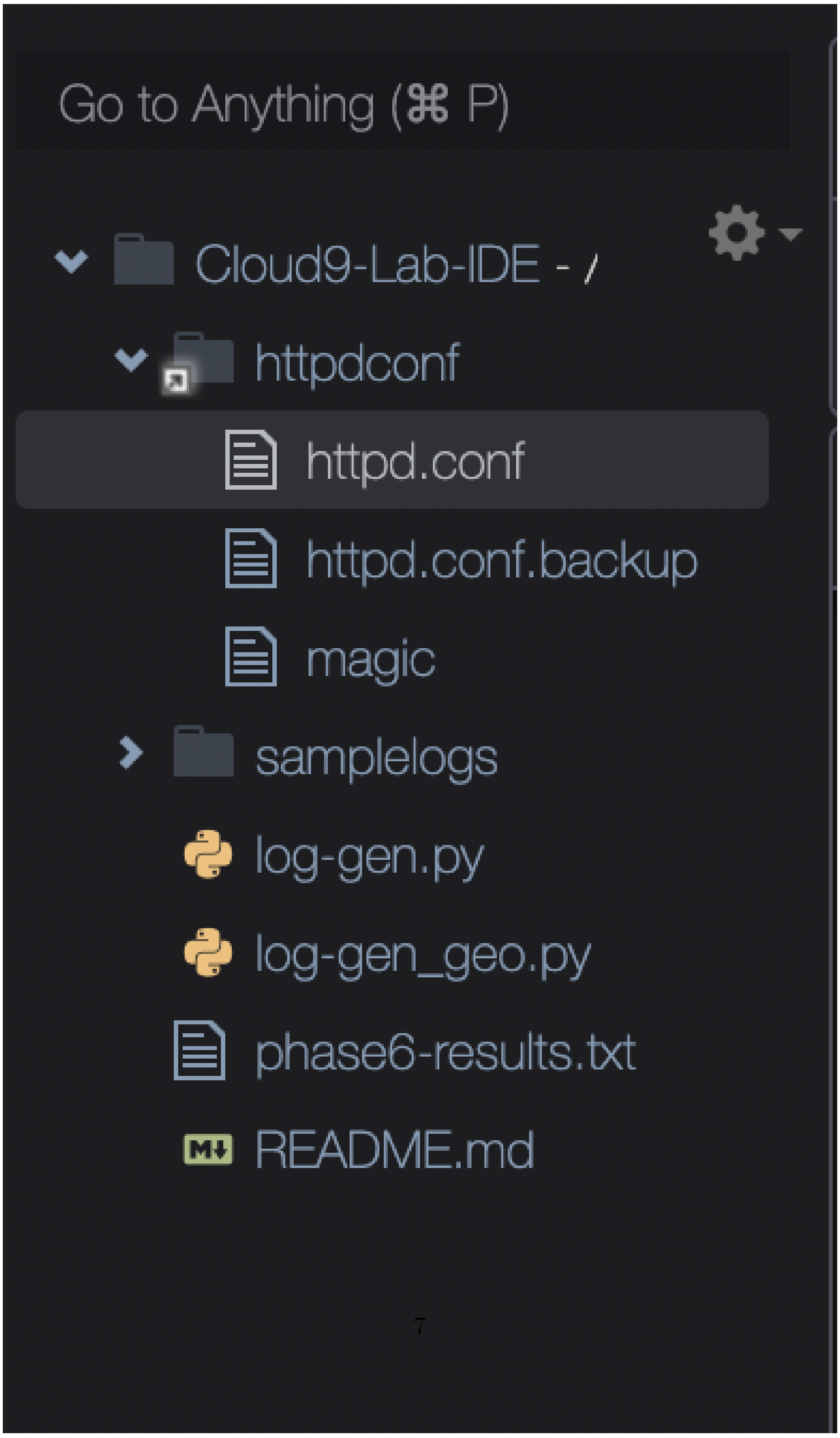


Figure 6: Cloud9 Directory Overview

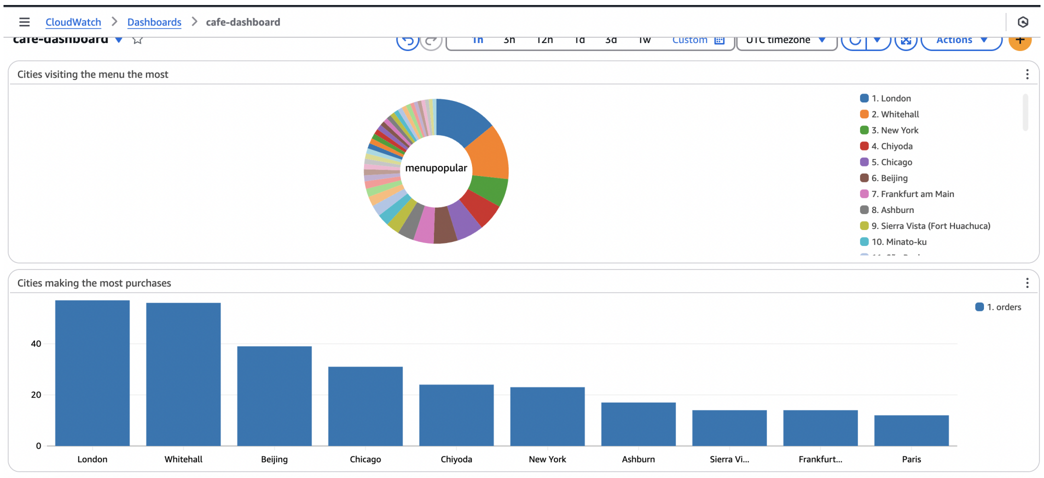


Figure 7: Cities with Most Visits and Purchases

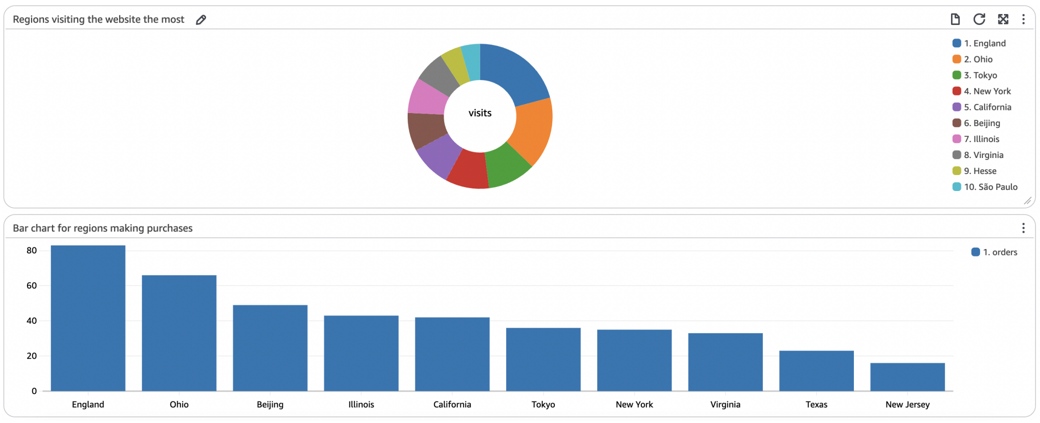


Figure 8: Regions with Most Visits and Purchases

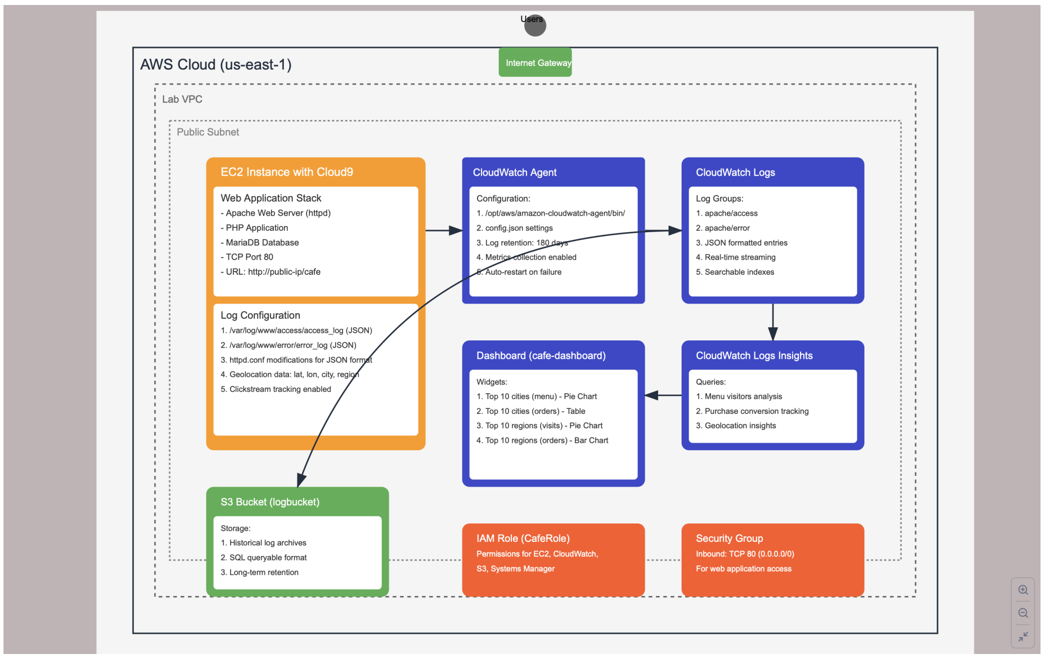

Users
AWS Cloud (us-east-1)
Internet Gateway
Lab VPC
Public Subnet
EC2 Instance with Cloud9
Web Application Stack
- Apache Web Server (httpd)
- PHP Application
- MariaDB Database
- TCP Port 80
- URL: http://public-ip/cafe
Log Configuration
1. /var/log/www/access/access_log (JSON)
2. /var/log/www/error/error_log (JSON)
3. httpd.conf modifications for JSON format
4. Geolocation data: lat, lon, city, region
5. Clickstream tracking enabled
CloudWatch Agent
Configuration:
1. /opt/aws/amazon-cloudwatch-agent/bin/
2. config.json settings
3. Log retention: 180 days
4. Metrics collection enabled
5. Auto-restart on failure
CloudWatch Logs
Log Groups:
1. apache/access
2. apache/error
3. JSON formatted entries
4. Real-time streaming
5. Searchable indexes
Dashboard (cafe-dashboard)
Widgets:
1. Top 10 cities (menu) - Pie Chart
2. Top 10 cities (orders) - Table
3. Top 10 regions (visits) - Pie Chart
4. Top 10 regions (orders) - Bar Chart
CloudWatch Logs Insights
Queries:
1. Menu visitors analysis
2. Purchase conversion tracking
3. Geolocation insights
S3 Bucket (logbucket)
Storage:
1. Historical log archives
2. SQL queryable format
3. Long-term retention
IAM Role (CafeRole)
Permissions for EC2, CloudWatch,
S3, Systems Manager
Security Group
Inbound: TCP 80 (0.0.0.0/0)
For web application access

Figure 9: CloudWatch Dashboard Configuration Screenshot

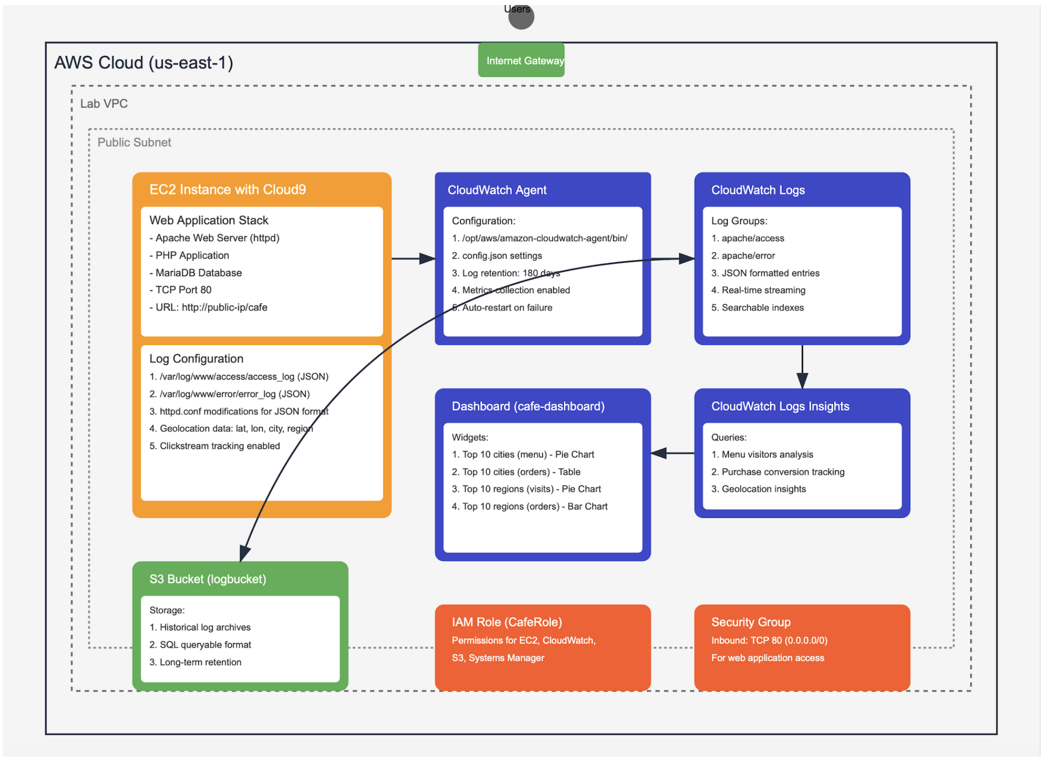
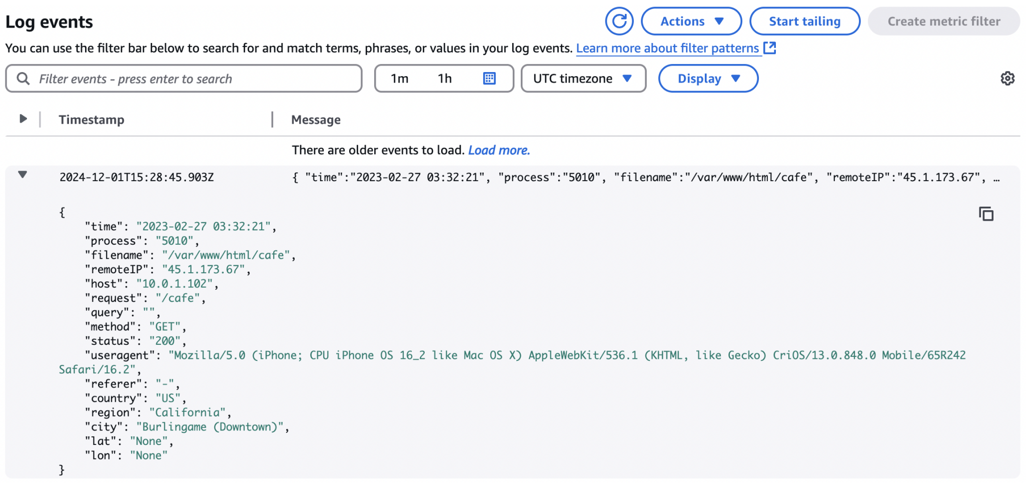

Users
AWS Cloud (us-east-1)
Internet Gateway
Lab VPC
Public Subnet
EC2 Instance with Cloud9
Web Application Stack
- Apache Web Server (httpd)
- PHP Application
- MariaDB Database
- TCP Port 80
- URL: http://public-ip/cafe
Log Configuration
1. /var/log/www/access/access_log (JSON)
2. /var/log/www/error/error_log (JSON)
3. httpd.conf modifications for JSON format
4. Geolocation data: lat, lon, city, region
5. Clickstream tracking enabled
CloudWatch Agent
Configuration:
1. /opt/aws/amazon-cloudwatch-agent/bin/
2. config.json settings
3. Log retention: 180 days
4. Metrics collection enabled
5. Auto-restart on failure
CloudWatch Logs
Log Groups:
1. apache/access
2. apache/error
3. JSON formatted entries
4. Real-time streaming
5. Searchable indexes
Dashboard (cafe-dashboard)
Widgets:
1. Top 10 cities (menu) - Pie Chart
2. Top 10 cities (orders) - Table
3. Top 10 regions (visits) - Pie Chart
4. Top 10 regions (orders) - Bar Chart
CloudWatch Logs Insights
Queries:
1. Menu visitors analysis
2. Purchase conversion tracking
3. Geolocation insights
S3 Bucket (logbucket)
Storage:
1. Historical log archives
2. SQL queryable format
3. Long-term retention
IAM Role (CafeRole)
Permissions for EC2, CloudWatch,
S3, Systems Manager
Security Group
Inbound: TCP 80 (0.0.0.0/0)
For web application access

Figure 10: Final CloudWatch Settings Screenshot



You can use the filter bar below to search for and match terms, phrases, or values in your log events. Learn more about filter patterns

```
Timestamp                    Message
                             There are older events to load. Load more.
2024-12-01T15:28:45.903Z     { "time":"2023-02-27 03:32:21", "process":"5010", "filename":"/var/www/html/cafe", "remoteIP":"45.1.173.67", …

{
    "time": "2023-02-27 03:32:21",
    "process": "5010",
    "filename": "/var/www/html/cafe",
    "remoteIP": "45.1.173.67",
    "host": "10.0.1.102",
    "request": "/cafe",
    "query": "",
    "method": "GET",
    "status": "200",
    "useragent": "Mozilla/5.0 (iPhone; CPU iPhone OS 16_2 like Mac OS X) AppleWebKit/536.1 (KHTML, like Gecko) CriOS/13.0.848.0 Mobile/65R242
Safari/16.2",
    "referer": "-",
    "country": "US",
    "region": "California",
    "city": "Burlingame (Downtown)",
    "lat": "None",
    "lon": "None"
}
```

Figure 11: Details of Access Logs in CloudWatch