**CSCI 5411 Milestone 2 Project Report**

**Name:** Vaibhavkumar Patel  
**Banner ID:** B01006432  
**Project Title:** Smart Facility Environmental Monitoring Dashboard

1. **Final Architecture and System Design**

**A diagram of a computer

AI-generated content may be incorrect.**

The Smart Facility Environmental Monitoring Dashboard is a cloud-based IoT solution built on AWS to monitor temperature, humidity, CO₂, and AQI data. Simulated sensor data is ingested using MQTT, processed in real time, stored in both raw and structured formats, and visualized via a Grafana dashboard. The system is designed for secure, scalable, and cost-effective operation using services allowed in the AWS Learner Lab.

**2. Service Selection and Configuration**

|  |  |  |
| --- | --- | --- |
| **AWS Service** | **Function** | **Justification** |
| AWS IoT Core | MQTT-based IoT data ingestion | Provides native MQTT broker and device message routing. Enables secure, scalable communication between IoT simulator and backend infrastructure. |
| AWS Lambda | Data transformation, routing | Event-driven compute service perfect for processing real-time IoT streams without provisioning infrastructure. Ensures elasticity and low cost. |
| Amazon S3 | Archive raw sensor data | Durable object storage with lifecycle management. Ideal for storing time-series sensor payloads with auto-deletion to optimize costs. |
| Amazon DynamoDB | Store processed data | Fully managed NoSQL service supporting real-time reads/writes for dashboard use. On-demand capacity mode scales with incoming data rate. |
| Amazon SNS | Alert developers | Lightweight pub/sub service used to notify admins of system faults or data threshold breaches. Ensures proactive intervention. |
| Amazon EC2 (Grafana) | Dashboard visualization | Used to self-host Grafana in absence of managed services. Visualizes DynamoDB metrics for monitoring facility conditions. |
| Amazon CloudWatch | Logging, alarms, metrics | Enables end-to-end observability by collecting Lambda logs, defining alarms, and tracking custom system metrics. |
| Amazon VPC/Subnets | Network isolation | Ensures all services operate within isolated environments, supporting secure routing and inter-service communication across private subnets. |
| VPC Endpoints | Private AWS service access | Allows Lambda and EC2 to access S3 and DynamoDB securely over AWS backbone without traversing the public internet. |

Removed:

1. AWS kinesis : Not available in AWS free tier account.
2. GRAFANA: Integrating the Grafana based custom dashboard with the one click automation file using the terraform was more time consuming. For this simple use case, I replaced it with the CloudWatch based dashboard for the insights from the data.

**3. Requirements Fulfillment**

**Functional:**

Simulate IoT sensors producing temperature, humidity, AQI, CO₂.

A screenshot of a computer

AI-generated content may be incorrect.

• Ingest sensor data via MQTT through AWS IoT Core.

A screenshot of a computer

AI-generated content may be incorrect.

• Use AWS Lambda to process, enrich, and route data.

A screenshot of a computer

AI-generated content may be incorrect.

Store raw sensor data in S3 with a 34-hour lifecycle deletion policy.

A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.

Store processed sensor data in DynamoDB for efficient dashboard querying.

A screenshot of a computer

AI-generated content may be incorrect.

• Generate SNS alerts on critical AWS resource events.

A screenshot of a computer

AI-generated content may be incorrect.

Visualize trends and real-time conditions

A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a phone

AI-generated content may be incorrect.

Monitor application health and data pipeline via AWS CloudWatch dashboards

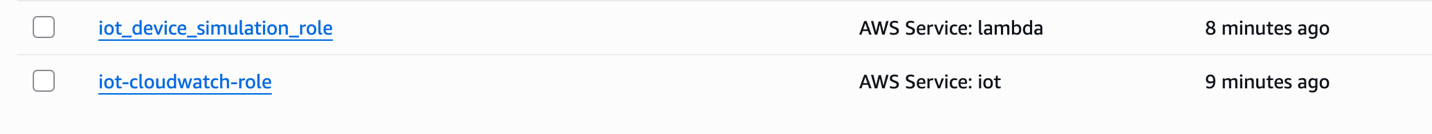
and alarms.

A screenshot of a computer

AI-generated content may be incorrect.

**Non-Functional:**

* **Security**: IAM roles, VPC isolation, Policies with least privileges



* **Scalability**: Serverless design, managed services.
* **Performance**: Low-latency dashboard (<2s).
* **Reliability**: CloudWatch alarms, Central logging for the logs, SNS based alerts for the critical failures in the system.
* **Cost**: on-demand usage, Event-driven architecture.
* **Disaster Recovery**: Stateless processing + Terraform redeployment.

**4. Monitoring and Logging**

* Configured the CloudWatch based logs from the lambda functions.
* Also, exported custom critical matrices to get the pin point in the system failure.
* Crafted a System health dashboard to see the events at one central place
* Alarms notify via SNS on errors or abnormal conditions.
* Centralized logs improve observability and traceability.

**5. Security Controls**

* IAM roles per service, least privilege applied.
* TLS used for MQTT and HTTPS traffic.
* S3 and DynamoDB encrypted at rest.
* Grafana IP-restricted; future support for Cognito considered.

**6. Cost Optimization**

| **Component** | **Monthly Est.** | **Strategy** |
| --- | --- | --- |
| EC2 (Grafana) | $10–15 | t3.small, minimal uptime |
| Lambda | <$1 | Serverless, under free tier |
| S3 | <$0.25 | Lifecycle deletes after 34 hours |
| DynamoDB | <$2 | On-demand pricing |
| Monitoring | <$2 | Controlled metrics, minimal logs |
| **Total** | **~$20** | Well under Learner Lab budget |

**7. Lessons Learned & Future Scope**

**Challenges:**

* Learner Lab restrictions on IAM/Cognito.
* Persistent storage configuration for Grafana.

**Future Improvements:**

* Use Cognito for secure dashboard access.
* Integrate Amazon Timestream for long-term trends.
* ML-based anomaly detection with SageMaker or Lookout.

**8. Infrastructure as Code (Terraform)**

* IaC used to provision all components.
* Modules defined for IoT, Lambda, S3, DynamoDB, EC2, VPC.
* One-step deployment via terraform apply.
* Screenshots and TF files provided in appendix.

**9. AWS Well-Architected Framework Alignment**

**Operational Excellence:**  
The architecture promotes operational visibility through centralized logging using CloudWatch and event-based alerts using SNS. Lambda functions are stateless and monitored, enabling easy rollback and issue tracking. Terraform ensures consistent and repeatable deployments, allowing operational improvements and recovery to be automated.

**Security:**  
The architecture implements the principle of least privilege using IAM roles for all AWS services. All services are deployed within a VPC with appropriate subnets and security groups. TLS encryption is enforced for MQTT traffic and HTTPS endpoints. S3 and DynamoDB apply encryption at rest by default. VPC Endpoints ensure secure service-to-service communication without traversing the public internet.

**Reliability:**  
The architecture ensures high availability by deploying components across multiple availability zones within a VPC. Lambda’s event-driven model with retry logic and decoupling through IoT Core increases system resilience. CloudWatch Alarms detect anomalies and notify developers instantly. Persistent storage in DynamoDB and S3 ensures no data loss in case of downstream failures.

**Performance Efficiency:**  
The system uses AWS managed services like Lambda, DynamoDB, and IoT Core to auto-scale based on workload. Data is processed within sub-second latency, and performance metrics are monitored through CloudWatch. DynamoDB supports real-time, low-latency querying for dashboard use, and unnecessary computation is avoided via lightweight Lambda functions.

**Cost Optimization:**  
Serverless components like Lambda and DynamoDB on-demand pricing help reduce idle resource costs. S3 lifecycle rules delete raw data after 34 hours to control storage costs. EC2 usage is minimal and only chosen due to Learner Lab restrictions. Logging and metrics collection is scoped to essential fields to avoid cost inflation.

**Sustainability:**  
The architecture emphasizes minimal infrastructure footprint. Serverless components reduce resource overprovisioning. Lifecycle management in S3 and on-demand DynamoDB ensures unused data is removed efficiently. EC2 usage is tightly controlled, and future enhancements include replacing it with managed Grafana for reduced energy consumption.

**Conclusion:**  
This Milestone 2 submission demonstrates a production-grade AWS architecture for IoT-based smart facility monitoring. The system is secure, scalable, observable, and cost-effective, adhering to the principles of the AWS Well-Architected Framework while satisfying both project and real-world application needs.