Software Construction and Development

Department of computer and software engineering

Submitted To: Dr Zunnurain Hussain

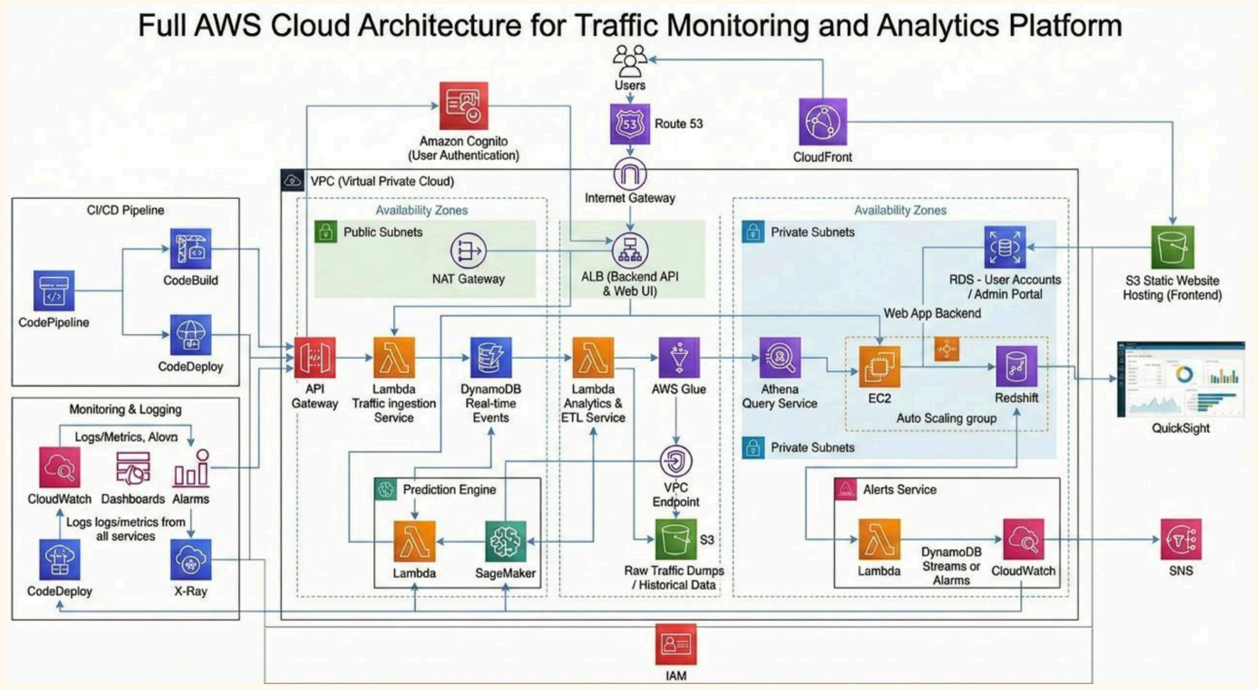
Submitted by:

Name: Abdullah Jabbar	**Roll no**: Bsse23073
Name: Hira Ijaz	**Roll no**: Bsse23103

Project:

Smart City Traffic Prediction & Alert System

Architecture Diagram



1. Introduction

The **Smart City Traffic Prediction & Alert System** is designed to collect live traffic data, perform real-time analytics, generate predictions using machine learning, and provide alerts to users and administrators.
This document explains the complete AWS cloud architecture used in this project, as shown in the submitted diagram. It covers data flow, components, security, scalability, and monitoring.

2. High-Level Architecture Overview

The system uses a **serverless and scalable AWS cloud infrastructure** to support:

- Traffic data ingestion
- Real-time analytics
- Traffic prediction using ML
- Alerts and notifications
- Admin portal + user dashboards
- CI/CD automation
- Full monitoring and logging
- High availability through multi-Availability Zone setup

The architecture is divided into:

1. **Frontend Layer (User Access)**
2. **Backend Layer (API + Web App Backend)**
3. **Data Ingestion & Processing Layer**

4. **Machine Learning & Prediction Layer**
5. **Storage & Query Layer**
6. **Alerts & Notification Layer**
7. **Monitoring & CI/CD Layer**
8. **Security Layer (IAM, VPC, Cognito)**

3. Architecture Components (Detailed Description)

3.1 User Access Layer

3.1.1 Amazon Route 53

- Handles domain routing.
- Directs users to the CloudFront distribution.

3.1.2 Amazon CloudFront

- Global CDN for delivering frontend UI with low latency.
- Serves frontend hosted on **S3 Static Website Hosting**.

3.1.3 Amazon Cognito

- Handles **user authentication** (Admin + Normal Users).
- Generates secure tokens for API access.

4. VPC Environment (Core of System)

Everything runs inside a secured **Virtual Private Cloud (VPC)** with **public and private subnets** across multiple availability zones.

4.1 Public Subnets

Used for:

- NAT Gateway
- Application Load Balancer (ALB)

4.2 Private Subnets

Used for secured backend operations:

- EC2 Auto Scaling Group
- RDS Database
- ETL Services
- Lambda Functions
- DynamoDB
- Redshift

4.3 Internet Gateway

Allows public communication for approved resources.

4.4 NAT Gateway

Allows private-subnet resources to access the internet securely without exposing them publicly.

5. Backend Layer

5.1 Application Load Balancer (ALB)

- Routes authenticated traffic to backend APIs and Web UI services.
- Works with Cognito for secure access.

5.2 API Gateway

- Public entry for IoT devices / sensors / mobile apps to send live traffic data.
- Connected to a Lambda Function.

5.3 EC2 Auto Scaling Group

- Runs Web App Backend (Admin Portal, User Dashboard APIs).
- Auto scaling ensures high availability during traffic spikes.

5.4 Amazon RDS

- Stores user accounts, admin data, and system configuration.
- Secured inside private subnets.

6. Data Ingestion & Processing Layer

6.1 Lambda – Traffic Ingestion Service

- Receives live traffic data from sensors/devices through API Gateway.
- Performs initial validation and formatting.

6.2 DynamoDB (Real-Time Events Storage)

- Stores processed live events with high read/write performance.
- Triggers Streams for further processing.

6.3 Lambda – Analytics & ETL Service

Performs:

- Cleaning
- Transformation
- Aggregation of traffic data
- Sends processed data to S3 Bucket for historical storage.

6.4 S3 (Raw Traffic Dumps / Historical Data)

- Stores sensor raw data
- Stores ETL-processed data
- Integrated with VPC Endpoint for secure internal operations.

7. Machine Learning & Prediction Layer

7.1 AWS Glue

- Performs ETL pipelines for preparing datasets for ML.
- Extracts data from S3 and DynamoDB.

7.2 Amazon Athena

- SQL-style query service for analyzing stored traffic data.
- Used by analysts and system admins.

7.3 Amazon SageMaker

- Builds, trains, and deploys **traffic prediction ML models**.
- Model takes input from processed traffic analytics.

7.4 Lambda – Prediction Engine

Uses deployed ML model to generate:

- Traffic predictions
- Congestion probability
- Estimated travel times
- Sends results to DynamoDB and Alerts service.

8. Storage & Analytics Layer

8.1 DynamoDB

- Used for fast real-time storage for events and predictions.

8.2 Amazon Redshift

- Data warehouse for large-scale traffic analytics.
- Connects with QuickSight for dashboards.

8.3 Amazon QuickSight

Visualizes:

- Traffic trends
- Predictions
- Alert history
- Admin management reports

9. Alerts & Notification Layer

9.1 Lambda – Alerts Service

- Automatically detects abnormal traffic patterns.
- Processes DynamoDB Streams or CloudWatch alarms.

9.2 SNS (Simple Notification Service)

Sends:

- SMS alerts
- Push notifications
- Email notices

9.3 CloudWatch Alarms

Alerts admins for:

- API failures
- High traffic load
- System issues

10. Monitoring & Logging Layer

10.1 Amazon CloudWatch

- Centralized logs, metrics, dashboards.

Monitors:

- Lambda performance
- API calls
- EC2 health
- RDS usage
- System errors

10.2 AWS X-Ray

- Tracing for API and backend performance.

- Deep insight into system bottlenecks.

11. CI/CD Pipeline

11.1 CodePipeline

- Automates the entire deployment process.

11.2 CodeBuild

- Builds the backend application, Lambda code, and frontend assets.

11.3 CodeDeploy

Deploys the application to:

- EC2 Auto Scaling Group
- Lambda Functions
- S3 Website Hosting

This ensures seamless, zero-downtime deployments.

12. Security Layer

12.1 IAM (Identity & Access Management)

- Controls permissions for all AWS services.
- Ensures least-privilege access principle.

12.2 Cognito

- User identity and access control.

12.3 VPC Security Groups

- Protect backend servers and DB from public access.
- Only ALB and NAT have controlled internet access.

13. Data Flow Summary

1. Users access the system via Route 53 → CloudFront → S3 frontend.
2. Cognito handles login.
3. Traffic data collected via API Gateway → Lambda → DynamoDB.
4. ETL jobs prepare data → S3 → Glue → Athena.
5. ML prediction pipeline uses SageMaker models.
6. Alerts generated using Lambda, CloudWatch, and SNS.
7. Admin backend runs on EC2 in private subnets
8. QuickSight dashboards use Redshift for analytics.

14. Conclusion

This architecture provides a **scalable, secure, and reliable AWS infrastructure** for the Smart City Traffic Prediction & Alert System.
It supports:

- Real-time data ingestion
- Traffic prediction with ML
- System alert
- High-availability web interface
- Automated deployments

- Full monitoring and logs

This document includes **all major technical details**, matches the architecture diagram.