 **Architecture Overview**

The system is built entirely with **AWS serverless components** and is designed to:

* Ingest and process events
* Enrich events when necessary
* Route notifications via **WebSocket**, **Push**, or **Email**
* Monitor and log all activity

## ****Components & Data Flow****

### 1. ****Event Source (App/API)****

* This is where the event originates—could be a user action, system trigger, or external API.
* Sends events to **EventBridge**.

### 2. ****Amazon EventBridge****

* Acts as the central event bus.
* Decouples event producers from consumers.
* Routes incoming events to a Lambda function for enrichment.

### 3. ****Lambda: Enrichment****

* **Validates** whether the event requires enrichment.
* If yes, it queries:
  + **External APIs**
  + **Databases**
* Adds missing details (e.g., user profile, metadata).

### 4. ****Lambda: Notification Router****

* Takes the (enriched or raw) event and:
  + Determines target channels (WebSocket, push, email)
  + Sends to each accordingly

## ****Notification Channels****

### 5. ****WebSocket Notifications****

* Uses **API Gateway WebSocket** to maintain bi-directional connections with clients.
* **DynamoDB** stores active connection IDs.
* Lambda sends real-time messages to the appropriate connections.

### 6. ****Mobile Push Notifications****

* Uses **SNS (Simple Notification Service)**.
* Supports Android and iOS platforms.
* Devices are subscribed to SNS topics.

### 7. ****Email Notifications****

* Uses **SES (Simple Email Service)**.
* Lambda handles:
  + Dynamic content generation
  + Multilingual support
  + Placeholder substitution (e.g., user name, date)

### 8. ****CloudWatch Logging & Monitoring****

* Logs all Lambda executions, errors, and metrics.
* Supports alarms and debugging.