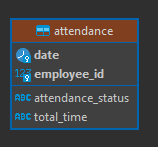
* Logical Architecture:

For the logical architecture, we'll have the following components:

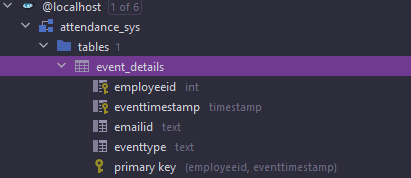
* + attendance-service-graphql:
    - Responsible for calling the attendance-system-service
    - Exposes Graphql endpoints to fetch details of an employee from RDBMS via service call.
    - The endpoint will be accessible by api-gateway
* attendance-system-service:
  + - Responsible for fetching records from MySQL DB
    - The endpoint will also be accessible by api-gateway.
* attendance-event-store:
  + - Responsible for mimicking HTTP POST Swipe In/Out calls
    - The endpoint will also be accessible by api-gateway.
  + attendance-calculation-service:
    - Reads swipe events from Cassandra.
    - Performs EOD calculation for all the employee swipe records
    - Publishes EOD calculation results to a Kafka topic.
* Attendance-consumer-service:
  + - Consumes EOD calculation events from Kafka.
    - Persists attendance records in the MySQL database for reading purposes.
* attendance-email-service:
  + - Consumes employee absent events from Kafka.
    - Sends mails to the employee about absent notification after EOD calculation
  + Attendance-system-api-gateway:
    - Responsible for Security- applies JWT Authentication with Google API
    - Routing: to route to load balanced service instance
    - Logging: to see which path is being called
  + attendance-system-cloud-config-server
    - Responsible for maintain centralized configuration properties of microservices
    - Provides server and client-side support for externalized configuration in a distributed system using spring-cloud-config
  + attendance-system-naming-server
    - It is a centralized registry that maintains a list of all the registered services and their instances.
    - All the services are Clients which register themselves on this server
    - Helps in load balancing act as mid-tier.
* Kafka:
  + - Message broker used for event-driven communication between services.
* Cassandra:
  + - NoSQL database used for storing swipe events.
* MySQL:
  + - Relational database used for storing attendance records.
  + Docker:
    - For containerisation of each services, kafka and databases
* Azure Services**: (Initially: Apache Kafka for event streaming instead of Azure Event hubs, MySQL instead of Azure SQL Database, Cassandra instead of Azure Cosmos DB, Kubernetes instead of AKS, Docker containers for DB)**
* Azure Event Hubs:
  + Used for event streaming and message processing.
  + Provides a Kafka-compatible interface for publishing and consuming events.
* Azure SQL Database:
  + Stores employee information such as employee ID, name, and department.
  + Accessed by the Attendance Service for querying employee data.
* Azure Cosmos DB:
  + Used as the event store for storing swipe in/out events.
  + Provides scalability and fault tolerance for event storage.
* API Gateway:
  + Utilize Azure API Management for managing APIs, enforcing security policies, and monitoring API usage
* Monitoring:
  + Utilize Azure Monitor for logging, monitoring, and alerting on system performance and health.
* CI/CD Pipeline:
  + Utilize Azure DevOps for continuous integration and continuous deployment pipelines for automated build, test, and deployment processes.

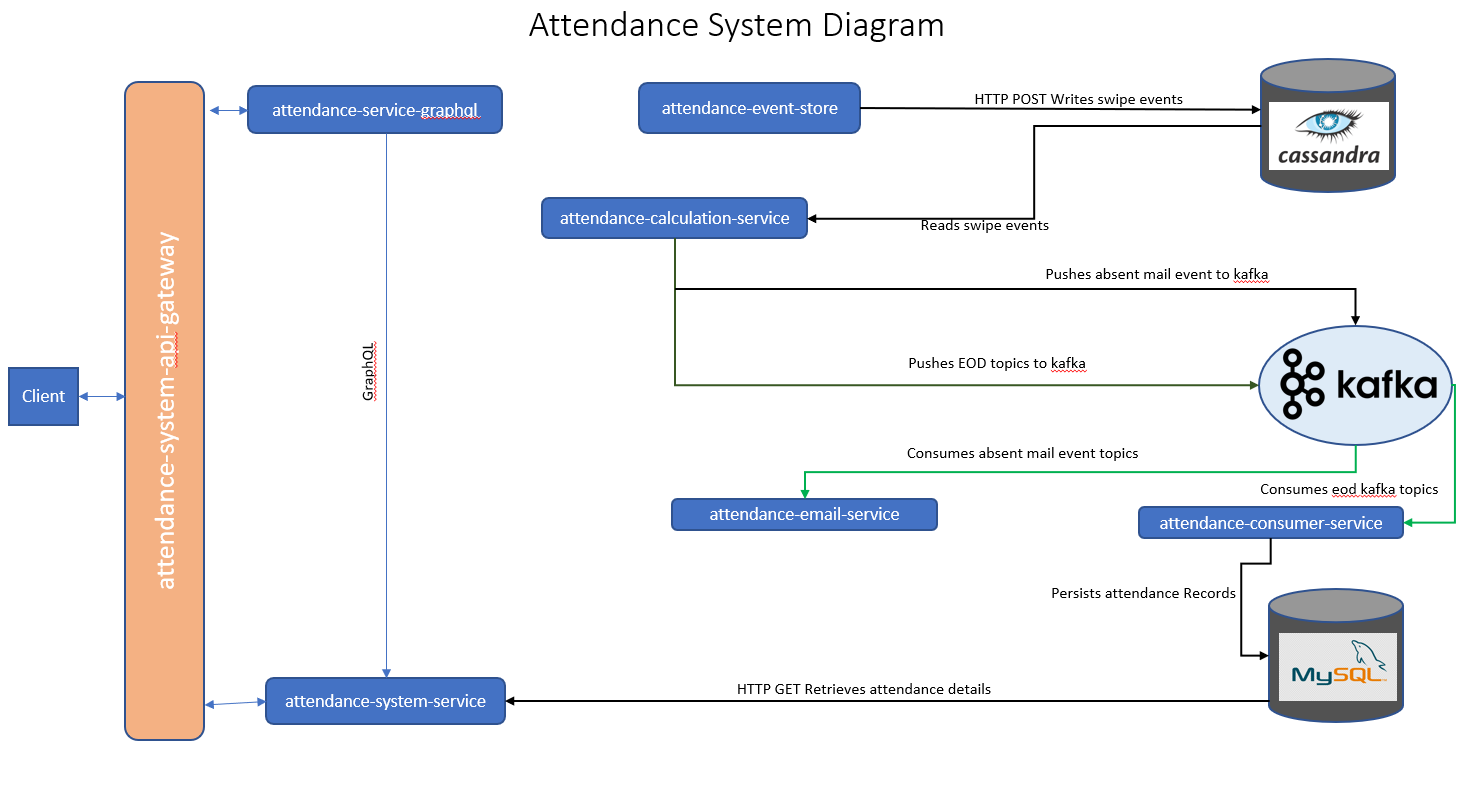
For the physical architecture, we'll deploy these components on Azure cloud using Azure Kubernetes Service (AKS) for container orchestration.

* Data Stores:
* MySQL:

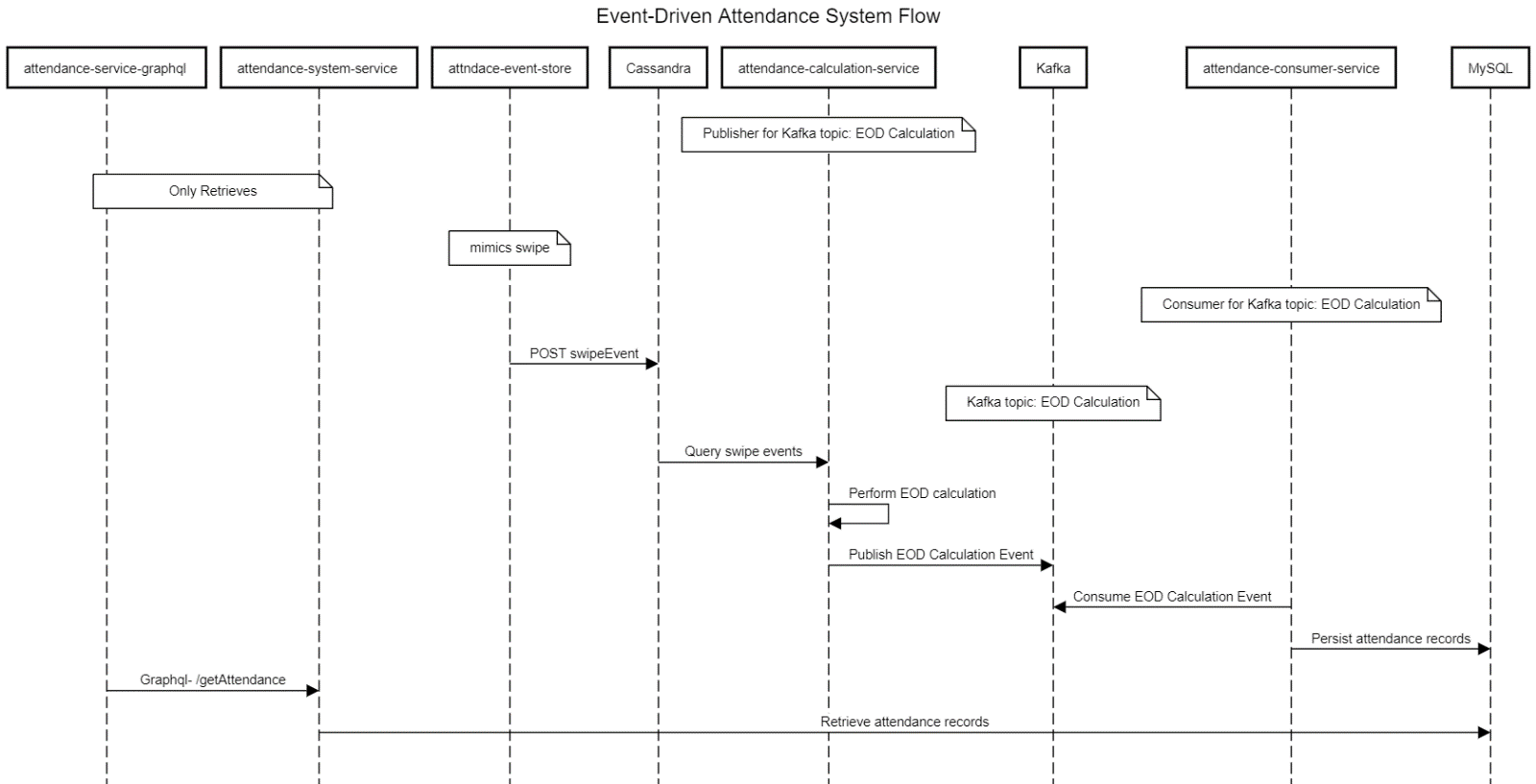


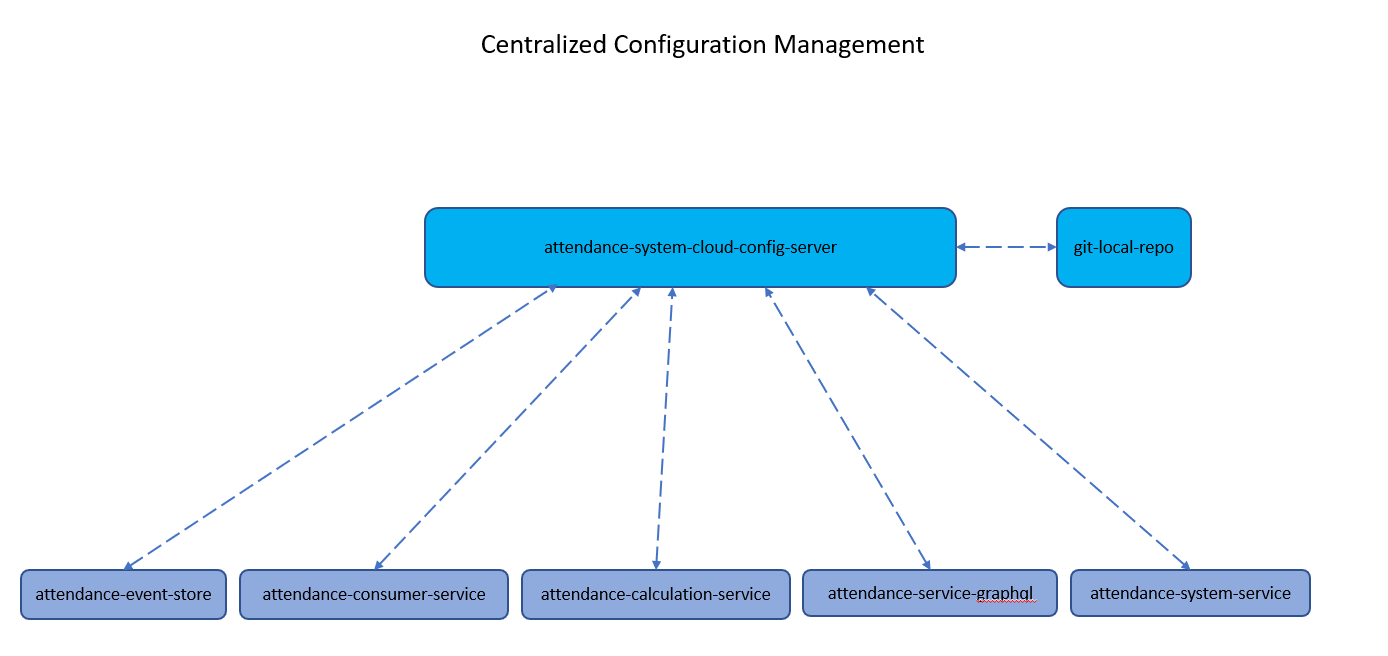
* Event Service (Cassandra):

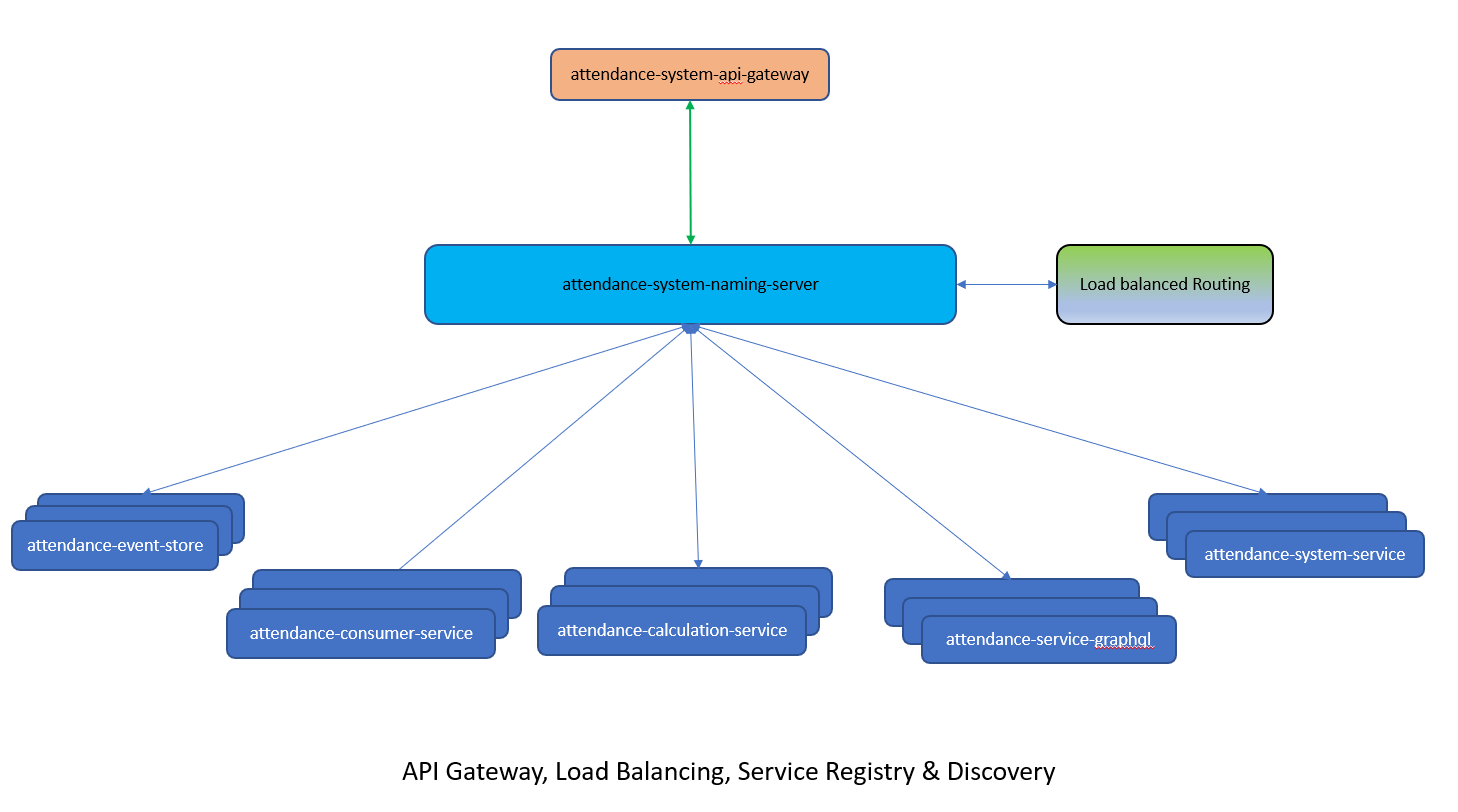




* Sequence Diagram:







DDL:

**Cassandra:**

*create table attendance\_sys.event\_details*

*(*

*employeeid int,*

*eventtimestamp timestamp,*

*emailid text,*

*eventtype text,*

*primary key (employeeid, eventtimestamp)*

*)*

*with caching = {'keys': 'ALL', 'rows\_per\_partition': 'NONE'}*

*and compaction = {'class': 'org.apache.cassandra.db.compaction.SizeTieredCompactionStrategy', 'max\_threshold': '32', 'min\_threshold': '4'}*

*and compression = {'chunk\_length\_in\_kb': '16', 'class': 'org.apache.cassandra.io.compress.LZ4Compressor'}*

*and dclocal\_read\_repair\_chance = 0.0*

*and speculative\_retry = '99p';*

**MySQL:**  
*create table attendance*

*(*

*date date not null,*

*employee\_id int not null,*

*attendance\_status varchar(255) null,*

*total\_time varchar(255) null,*

*primary key (date, employee\_id)*

*);*