* Logical Architecture:

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

* Microservices:
  + Attendance REST API:
    - Responsible for handling HTTP requests from employees.
    - Provides endpoints for employees to swipe in/out and retrieve attendance records.
    - Interacts directly with the Attendance Calculation Service to perform EOD calculation and retrieve attendance records.
* Event Service:
  + - Responsible for receiving swipe events from employees.
    - Publishes swipe events to a Kafka topic.
* Attendance Calculation Service (Microservice):
  + - Consumes swipe events from Kafka.
    - Performs EOD calculation based on the received events.
    - Publishes EOD calculation results to a Kafka topic.
* Attendance Read Store (Microservice):
  + - Consumes EOD calculation events from Kafka.
    - Persists attendance records in the MySQL database for reading purposes.
* 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.

* Detailed Design:

1. Attendance REST API (Microservice):
   * Exposes REST endpoints for swiping in/out and retrieving attendance records.
   * Validates and processes incoming requests.
   * Communicates directly with the Attendance Calculation Service to perform EOD calculation and retrieve attendance records.
   * API Contracts:
     1. GET /attendance/total-hours/{employeeId}: Retrieves total hours worked by an employee.
     2. POST /events/swipe: to mimic swipe-in/swipe-out
2. Event Service (Microservice):
   * Receives swipe events from employees.
   * Publishes received swipe events to a Kafka topic.
   * Ensures fault tolerance and scalability of event publishing.
   * Event Structures:
     + 1. Produces: SwipeEvent: { employeeId, timestamp, eventType }
3. Attendance Calculation Service (Microservice):
   * Subscribes to the Swipe Events Kafka topic and consumes swipe events.
   * Processes received events to perform EOD calculation for attendance.
   * Publishes calculated EOD results to a Kafka topic for further processing.
   * Employee attendance formula to get status for a particular day:
     1. Total hours for a date less than 4 hours – Absent
     2. Greater than 4 hours but less than 8 –Half day
     3. Greater than 8 hours – Present
   * Event Structures:
     1. Produces: EODCalculation: {employeeId, totalHours}
     2. Consume: SwipeEvent: { employeeId, timestamp, eventType }
4. Attendance Read Store (Microservice):
   * Listens for EOD Calculation Events from Kafka and consumes them.
   * Persists the calculated attendance records in the MySQL database.
   * Implements mechanisms for data consistency and reliability.
   * Event Structures:
     1. Consumes: EODCalculation: {employeeId, totalHours}
5. Kafka:
   * Configured with topics for Swipe Events and EOD Calculation Events.
   * Provides durable message storage and event distribution to subscribers.
6. Data Stores:

* MySQL:
* Tables:

1. Employee: { employeeId (PK), name, department }
2. AttendanceRecord: { employeeId (FK), date, status }

* Event Service (Cassandra):
* Tables:

1. SwipeEvents: { employeeId (PK), eventTimestamp (CK), eventType, PRIMARY KEY (employeeId, eventTimestamp) }

* Sequence Diagram:



