SWE In Class Activity 4

Members:

George, Brahma, Srikanth, Tisha, Manaswini

Question 1

Functional requirements

1. Users should be able to view venue availability and receive recommendations for available venues and able to book the one , of his/her choice
2. Users should be able to view parking lot details: location, number of spaces available, and able to reserve an available slot and pay for the parking
3. Users should be able to view weather updates and receive dynamic updates on indoor and outdoor venue availability.
4. Admin should have the ability to view and modify the event schedule,parking places

Non functional requirements

1. Scalability - the system should be scalable to support up to 1000 requests/second to handle the high traffic during the event. And upto overall 35K visitors for the event.
2. Reliability - the system should be reliable, ensuring that visitors receive up-to-date information and administrators get accurate data.
3. Security - the system should provide secure access to data, ensuring the privacy of visitors' personal information.
4. Power efficiency - since sensors may run out of power, the system should make use of intelligent mechanisms to save power.

Stakeholders

1. Visitors
2. IOT device managers
3. Venue and parking admins
4. App developers

Concerns

1. Network unavailability at the venues
2. More than expected visitors
3. Last minute changes in the schedule and/or venue
4. Malfunction of IOT devices
5. App severe crash
6. User unawareness

Question 2

Subsystems

1) User Management
    a) Login0
    b) Signup
    c) User details
2) Admin

    User +

    a) Adding deleting users
    b) Updating Venues/parkings details
3) Intelligence
    a) Suggested venues
    b) Expected crowd
    c) Possible availability of venue/parking
4) IOT
    a) Status of the device
    b) Status of the parking

Question 3

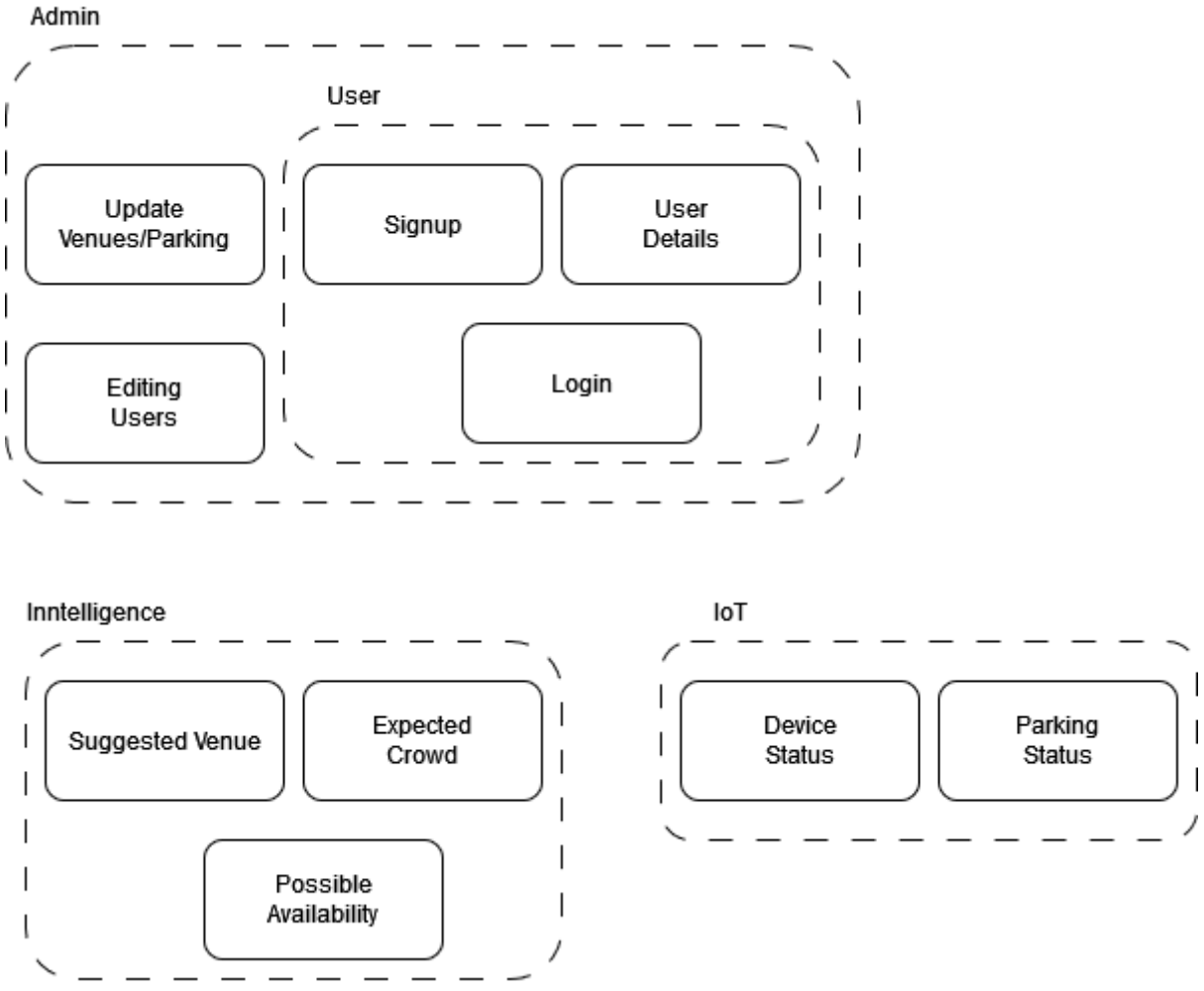
Key Quality attributes

1) App latency
    a) We can have multiple servers and in nearby locations to decrease latency
2) Accuracy

a) The accuracy of the information to be presented to the user can be maintained by having refresh rates at nearly 1 ms for payment systems around 10ms for venus and parking lot availability
b) Taking into consideration the IOT device health before finalizing the status of the parking slot availability

3) App availability
   a) The max strength of concurrent users can be considered more than [generally twice] the actual estimates and planning servers
   b) Having a fall back server with the most latest info as new as 1 sec ago

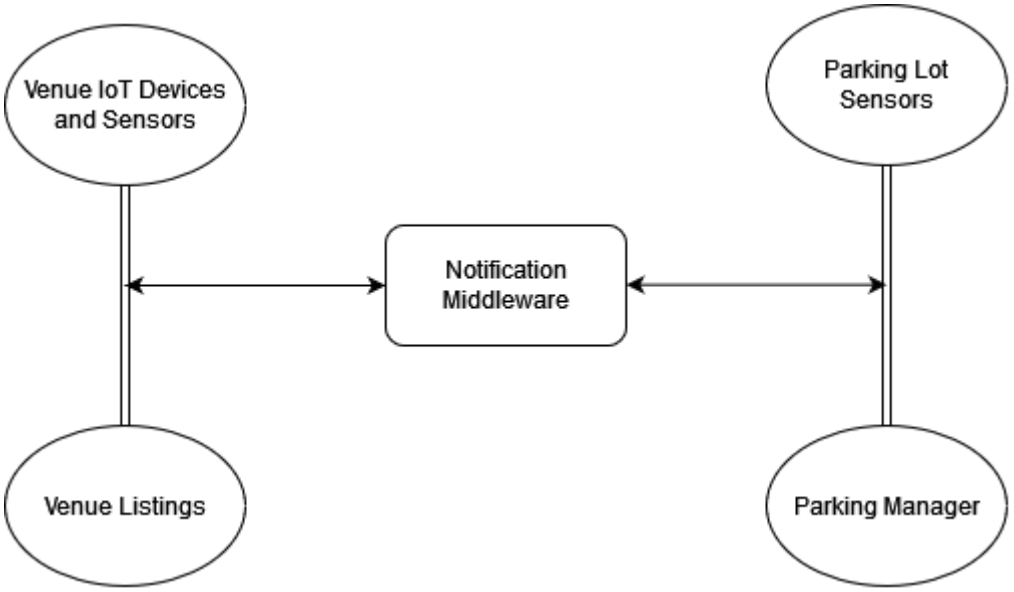
Question 4



Question 5

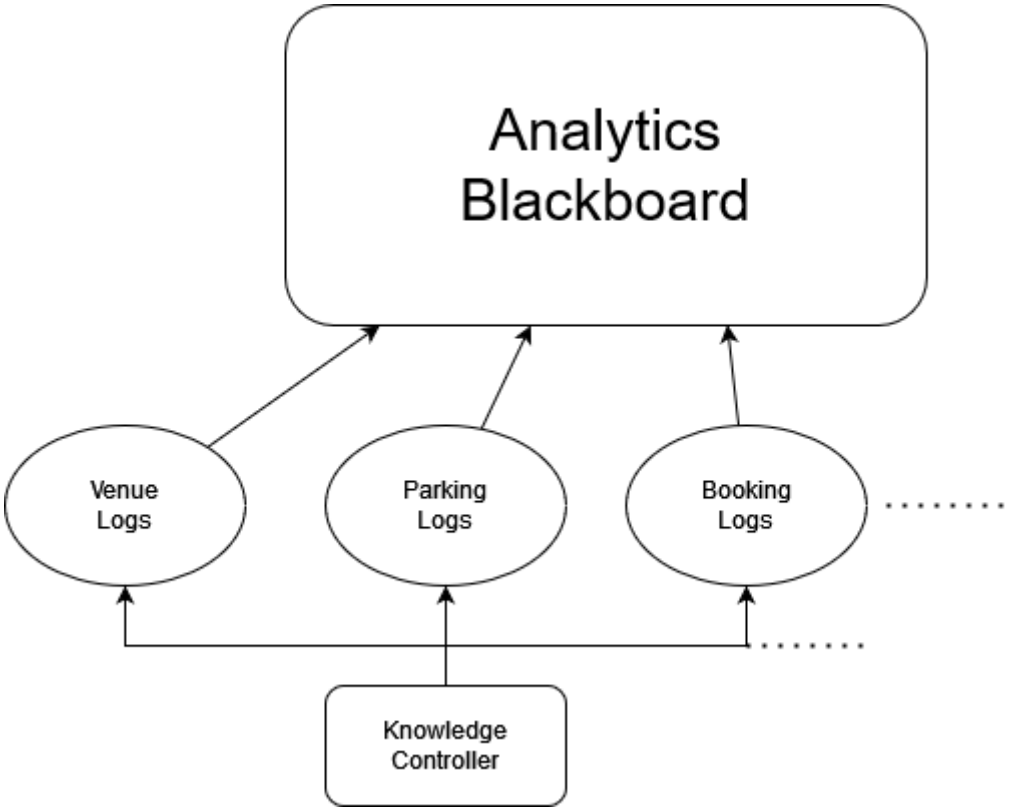
Publish-Subscribe

The IoT subsystem contains a stream of mostly passive data. We only need to be aware of when, say, cars enter and leave parking lots. This makes it a perfect candidate for the publish-subscribe pattern.



Blackboard

The analytics engine handles a lot of data from many sources. Knowledge comes in from the venue system, parking system, booking system etc. When a request comes in for the analytics, we can use the blackboard pattern to get data from various subsystems.



Question 6

a. List the contexts:

1. User Management
2. Parking Management
3. Venue Management
4. Booking Management
5. Billing/ Payment
6. Weather Forecast
7. Recommendation
8. Visualization and Analytics

b. Hidden modules:

1. User authentication and authorization
2. Data storage and retrieval
3. Logging and error handling

Shared modules:

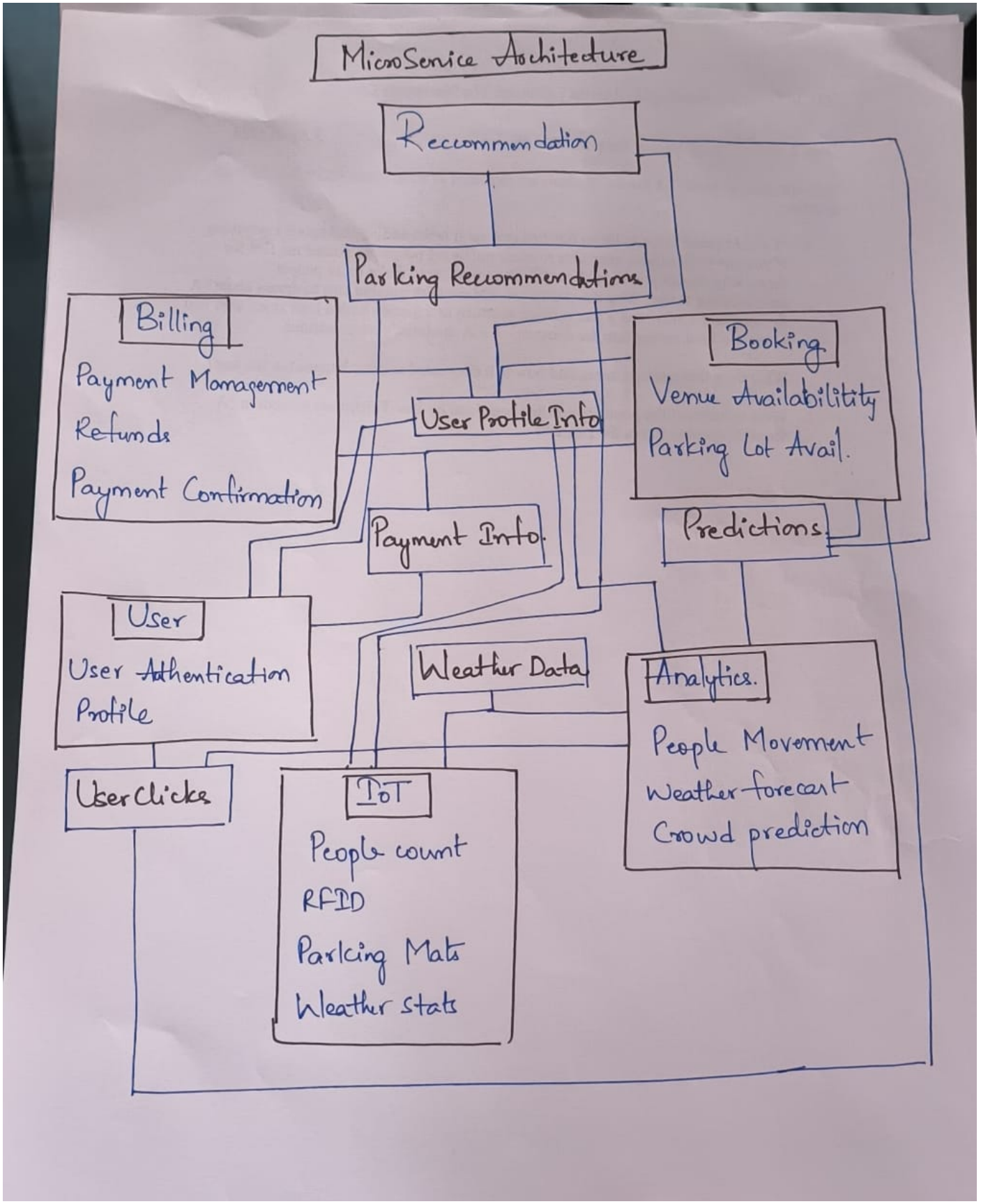
1. Notification service
2. Sensor data processing service

c. Microservices and corresponding databases:

1. User Service - Authentication and authorization platform (AuthO)
2. Parking Service - NoSQL DB (Mongo)
3. Venue Service - Relations DB (Oracle/ MySQL/ PostgreSQL for consistency in data retrieval)
4. Booking Service - Relations DB (Oracle/ MySQL for consistency or transaction processing)
5. Weather Service
6. Recommendation Service -
7. Visualization and Analytics Service - Elasticsearch for huge data lake.

i. The microservices will communicate using HTTP protocol over RESTful APIs. For North-South traffic, the system will use load balancers to distribute the incoming traffic to the respective microservices.

ii. The data exchange format will be JSON. This will be light weight and compatible to all the latest standards of RESTful APIs. Each microservice will accept and return JSON payloads to allow easy integration with other services.


MicroService Architecture
Recommendation
Parking Recommendations
Billing
Payment Management
Refunds
Payment Confirmation
User Profile Info
Booking
Venue Availability
Parking Lot Avail.
Payment Info.
Predictions
User
User Authentication
Profile
Weather Data
Analytics.
People Movement
Weather forecast
Crowd prediction
User Clicks
IoT
People count
RFID
Parking Mats
Weather stats