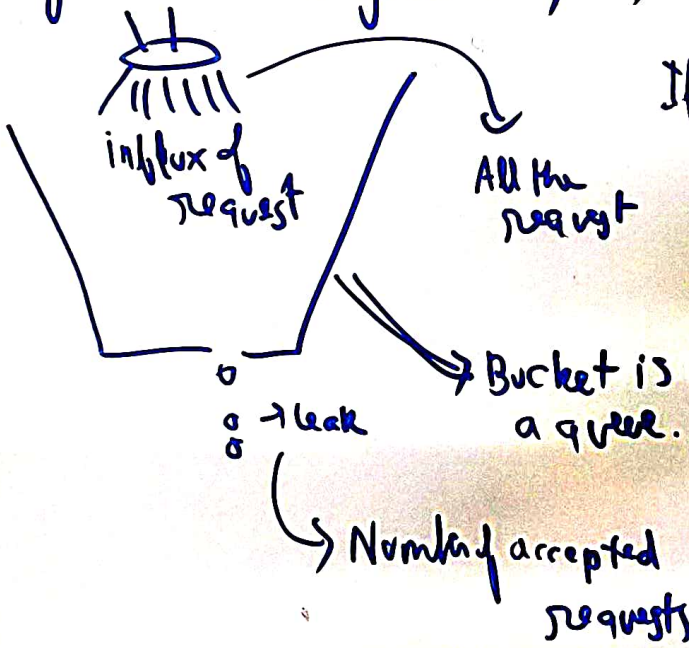
(Rate Limiter)

(Low Level Design)

(Leaky Bucket Algorithm) →



If Bucket is FULL then Requests start getting rejected.

ConcurrentLinkedQueue. → Bucket.

Background thread → Polling from the queue at a constant rate.

Multiple threads ⇒ pushing messages to the thread.

② Sliding Window Log.

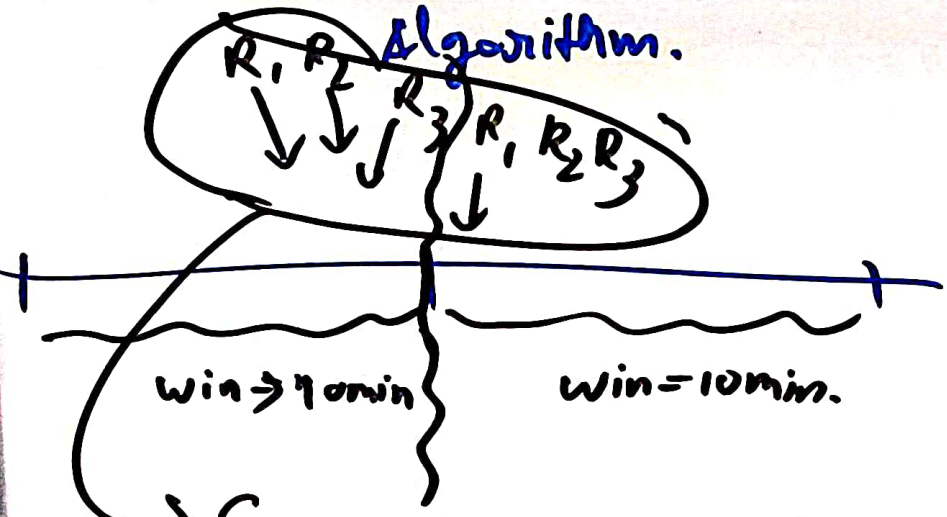
In this algo we are basically Logging all the accepted requests, in a queue.

↳ WindowSize Eg.:

→ when a new request comes we call an

```
allowRequest( ){
long now = System.currentTimeInMillis();
while(|q.peek - now| > windowSize){
    q.poll(). → Basic Sliding window
}
if(q.size() < cap){
    q.offer(now);
}
else
    Reject;
}
}
```

Fixed Window Counter Algorithm.



cap = 3
window = 10 min.

→ Separate windows, but bursts happening due to uneven splits.

```
windowStart;
windowSize;
AtomicInteger requestCount;
cap.
```

```
allowRequest(){
now = System.currentTime()
if (windowStart - now < windowSize()){
    if (requestCount < cap){
        requestCount.incrementAndGet();
    }
    else
        reject}
}
else{
    windowStart = now();
    requestCount.set(1);
}
```

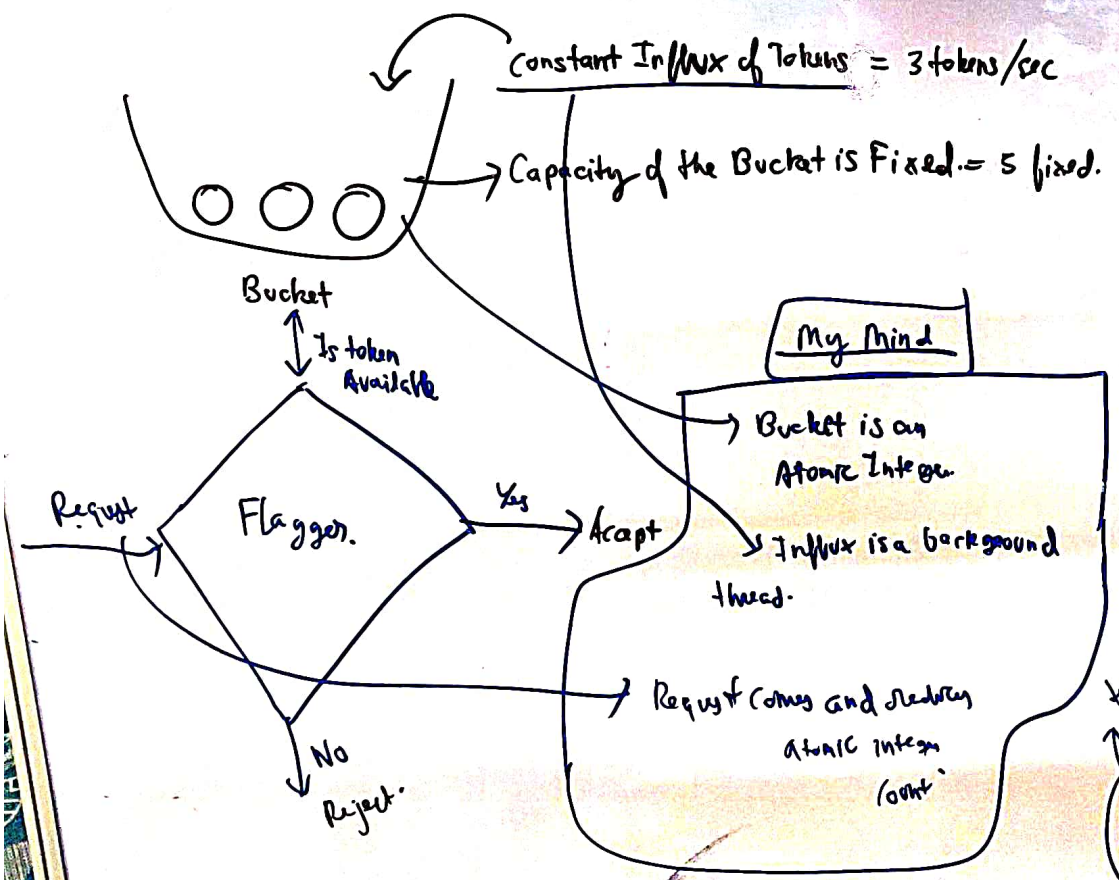
Sliding Window Counter.

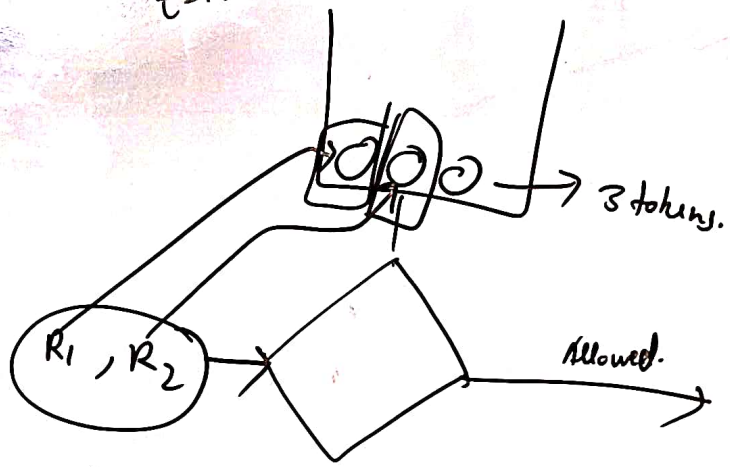
In this algorithm, we are trying to solve the memory issue of Sliding window Log.

(Rate Limiter.)

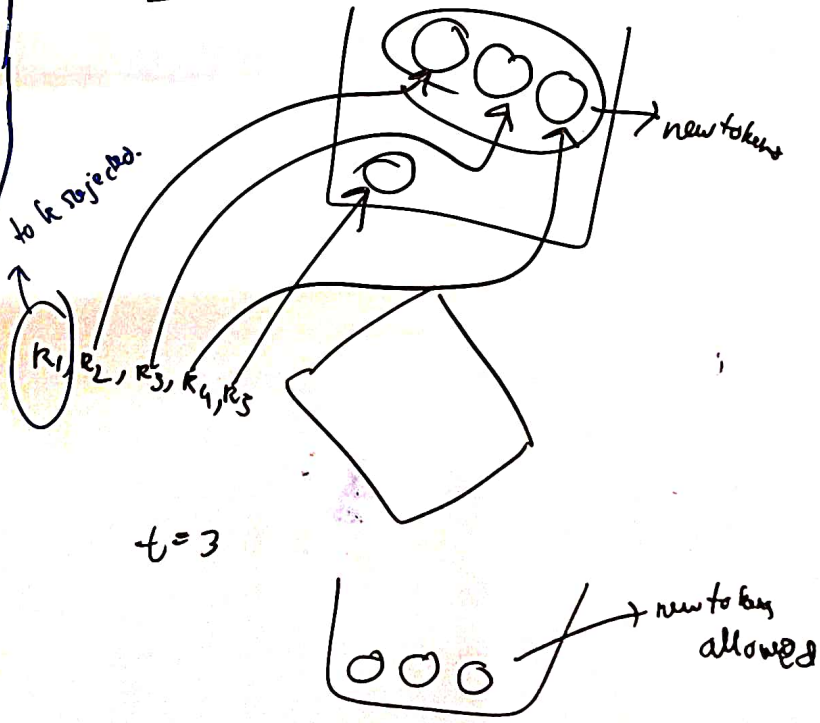
(Low Level Design)

Token Bucket Algorithm of Rate Limiting





t=2.



t=3