Objects:

```
Allow {
  boolean permitted
  string reason
}

User {
  string user_id
  string country_code
  boolean premiem_member
}

Request {
  string type
  string service
  int priority
  RequestBody rb
}

FallBackAlgorithm {
   THROTTLE_NEW_REQUESTS,
   DUMP_OLDEST_REQUEST,
   SPLIT_QUEUE
}
```

```
Capacity {

  int maximum_requests

  TimeDuration seconds/minutes/hours

  int maximum_requests_per_client

  FallBack fallback_algorithm

}
```

Oracle API Contract:

EmptyResponse register(Service, IP_address, Map<API, Capacity>, Capacity total_service_capacity)

Allow shouldProcess(User, Request)

You can also implement the rate limiting strategy as a state machine. Keep in mind that the more complex a rate limiter gets, the harder it is to:

1. Maintain
2. Predict behaviour
3. Stay performant