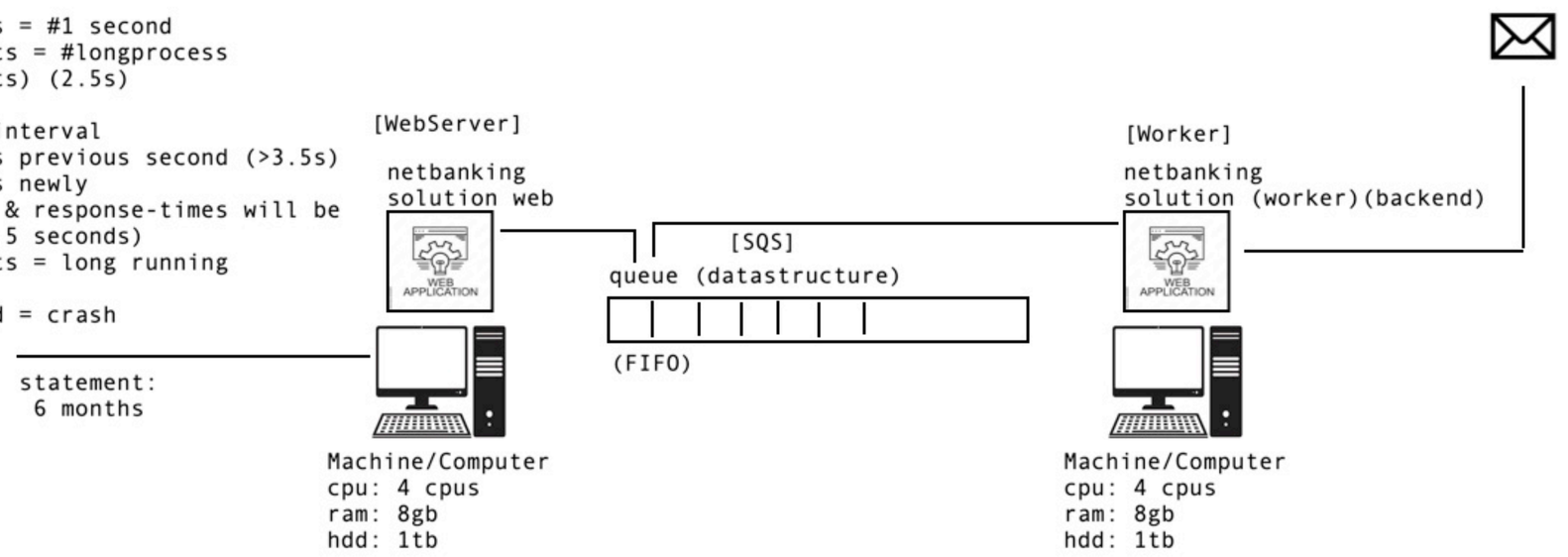
digicouser (Web Application)
digicoworker (Worker Application)

requestTypes:
1. open new account
2. account details
3. balance
4. funds transfer
5. bill payments
6. statements
etc

#500 requests = #1 second
|=20% requests = #longprocess
(100 requests) (2.5s)

#2nd second interval
#100 requests previous second (>3.5s)
#500 requests newly
(performance & response-times will be degraded) (1.5 seconds)
#100 requests = long running

#10 th second = crash



The Web Applications are by nature should take short-interval of time in processing the request and dispatching the response back to the user, which means all the requests of an web application are short-lived and should not take more amount of time in processing upon receiving the request.

standard average response time for an web application is between: 200 milliseconds to 1 second.
by taking the average response time into consideration, let us assume the above application with the machine capacity we have created should handle: 500 requests per second.

But in the above Web Application, there are few types of requests that are by nature/design takes more amount of time in performing the operation and computing the output, for dispatching the response.
1. generating statements
2. bill payment to third-parties (electricity bill, gas bill, dth/mobile recharges)

because of long-running requests are handled in sychronous manner, if more requests are being received towards these requestTypes or usecases, even though end-user traffic is constant and stable and has not exceeded the expected/designed capacity, still the application will run into performance and poor response times problems and eventually leads to the crash at some point of time.
at this point of time, it looks like we need to scale-out the application to handle the poor performance, but if we scale-out the application, the cost of running the application increases even though the in-flow of the users/business is same, that puts the business into losses.
so instead of scaling-out the application, the application should be re-designed:
1. identify the requestTypes and separate from which requests takes short-interval of time and which takes longer-time.
2. now design the requestTypes which seems to be taking long-time in processing and make them to be asynchronous.