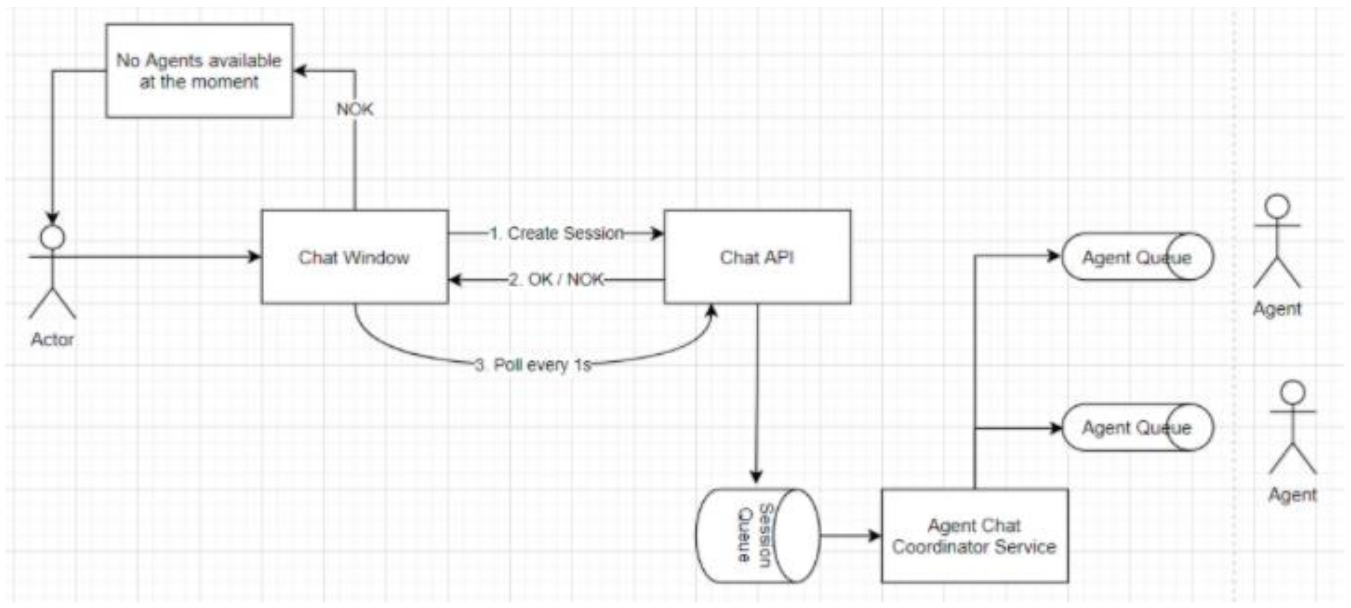
Calamatta Cuschieri
YOUR PARTNER IN FINANCIAL SERVICES

Back End Test – December 2020



A user will initiate a support request.
An API endpoint will be available that creates and queues a chat session
Once a chat session is created, it is put in an FIFO queue and monitored.
Once the session queue is full, unless it's during office hours and an overflow is available. The chat is refused.
Same rules applies for overflow; once full, the chat is refused.

Once the chat window receives OK as a response it will start polling every 1s

A monitor will watch the queue and mark a session inactive once it has not received 3 poll requests.

Once a chat request enters the queue a service will pick and assign the chat to the next available agent. The following rules applies:

Agents are on a 3 shift basis 8hrs each.
When a shift is over, the agent must finish his current chats, but will not be assigned new chats.
Capacity is the number of agents available, multiplied by their seniority and rounded down.

The maximum queue length allowed is the capacity of the team multiplied by 1.5
Once the maximum queue is reached and if during office hours, an over flow team kicks in. This adds extra people who are not normally their job to work as such. All persons assigned to overflow team are considered with the efficiency of a junior.

Consider the maximum concurrency (how many chats each agent can handle at the same time) is 10.
This is multiplied by the efficiency, which is pegged to the seniority of the agent as per below.

Seniority Multipliers:

- Junior: 0.4
- Mid-Level: 0.6
- Senior: 0.8
- Team Lead: 0.5

So example: A team of 2 mid-levels and a junior will have a capacity of
(2 x 10 x 0.6) + (1 x 10 x 0.4) = 16 concurrent chats capacity with a queue size of 24.

Teams available:
Team A: 1x team lead, 2x mid-level, 1x junior
Team B: 1x senior, 1x mid-level, 2x junior
Team C: 2x mid-level (night shift team)
Overflow team: x6 considered Junior.

Assigning Chats:
Chats are assigned in a round robin fashion, preferring to assign the junior first, then mid, then senior etc.
This ensures that the higher seniority are more available to assist the lower

e.g.

A team of 2 people: 1 snr(cap 8), 1 jnr (cap 4).
5 chats arrive. 4 of which would be assigned to the jnr and 1 to the senior
Team of 2jnr 1mid
6 chats arrive. 3 each to the jnr, non to the mid.

Guidelines:

- *Focus to be primarily on the back-end side of things, so it's ok to completely skip the front-end side of things. We would like to see fully working back-end code please.*

- *In terms of tech stack on should use: C# (.net framework or .net core), APIs, services, sql (or couchbase), rabbitmq (or kafka)*