**Preguntas de parcial**

Parcial 1:

- concepto de microservicios

- principal ventaja de microservicios por la que uno querría pasarse de monolito (*hablé de cuánto me va a salir (esfuerzo) pasarme a microservicios y porqué me pasaría. Le hable por temas de escalabilidad*.)

- porqué microservicios se lleva bien con cloud?

- que le hable de docker

- cuál es la ventaja que te da docker para un contexto que está en expansión?

- para desarrollar puntualmente / como ambiente de desarrollo, ¿qué te suma docker?

- en tu primera semana de que te sirve tener todo dockerizado?

Service Mesh

- contar como concepto qué es.

- qué ventajas me daría tener la infraestructura de esa forma con service mesh (*le dije que le podés agregar seguridad con el hecho de que los proxies están en todas las instancias, así que después me preguntó qué cosa más le agregaría además de seguridad, y le respondi que monitoreo, todo lo que tiene que ver con métricas*)

- Si yo no tengo service mesh, y quiero por ej agregar monitoreo, agregar seguridad, ¿qué tengo que hacer? ¿cómo lo hago?

Me dio un tema a elegir y yo le hablé de Sticky sessions y sessions replications

- y ahí donde se guarda la session, en qué servidor? ¿en cuál de los 3 servidores?

- para qué es sticky sessions, para que es el mecanismo, ¿qué me da?

Mecanismo para manejar la sesion en un contexto de clusters, apps donde hay más de un servidor.

- para qué sirve guardar/manejar la sesión?

Parcial 2:

- NOSQL: caracteristicas tecnicas, que tiene NOSQL en comparacion con SQL (*aca queria que dijera que en SQL no podia dividir la data*), me fue haciendo preguntas entre medio de si yo podria tener replicada la data en NOSQL y que ventaja tenía.

- Me dio a elegir entre microservicios y docker, elegí microservicios y me tiro un caso en el que un tipo tenía un equipo de 5 personas y se quería migrar a microservicios y que le explique ventajas y desventajas que tendría, y me pregunto si podía tener un monolito corriendo en varios servidores (*primero le dije que no porque pensé que se refería a dividir el monolito, pero se refería a distintas instancias*)

- Me tiro una en que me pidió que le justifique a un PM porque migrar de REST con JSON a RPC con proto buffers, aca hice agua como si fuera una cloaca (le fui tirando características de uno y el otro que recordaba) y lo que quería era que dijera que como proto buffer usa binary encoding pasar datos por el caño es más ligero.

- Como me fue mal en la anterior me dijo que errores veria si me dice que una API es restful y no lo es, le fui contestando, pero medio me fue ayudando, era decir que los recursos tenian que ser sustantivos en lugar de verbos, buen uso de los verbos REST y que se usen bien los codigos.

Parcial 3:

SRE:

¿Qué es?

¿Si te estas pasando del error budget y tenes q explicarle al gerente de marketing porque tenes que parar de agregar features como lo harías?

Front:

Como le venderias a algún gerente porque deberías parar de agregar features para lograr que sea responsive

Infra tradicional

Sticky sessions y session replication: dame un ejemplo en el q usarias cada una (osea en la q una es mejor q la otra). (*Replication vale la pena que si se te cae el server perdes la sesion y le tire facebook q es seguro q se va a morir uno Y me dijo q no pasa mucho ahi y si tenia un ej donde fuera crucial q no pasara eso y le tire en la bolsa q el min q tardas en crear la nueva sesion en otro server te puede costar millones. Sticky es mejor cuando queres q el usuario caiga siempre en el mismo server por alguna razon*).

Parcial 4:

* Explicar tipos de NOSQL, que tan flexibles son y demás.
  + *key value (solo tenes una key para busqueda, nada flexible)*
  + *Wide Column (La sort key te permite un poco mas de flexibilidad, y al estar ordenado te permite busquedas de rango)*
  + *Columnar (Ordenado y aparte agrupado por columna, mas flexible)*
  + *Documentos (Permite muchos indices, altamente flexible)*
  + *Grafos (el mas flexible, permite hacer busquedas de relaciones de relaciones)*
* infra tradicional: me pregunto cuando usarias no cloud (*Bancos, gobiernos, empresas muy grandes*) y despues cuando si cloud (*Basicamente dije las ventajas de cloud, mas que cuando si*)
* Me dio a explicar porque que tu sitio sea user responsive es bueno (*Muchos dispositivos con muchos navegadores posibles, si no sitio no anda en algun navegador, podes perder clientes*)

Parcial 5:

Me preguntó primero por monitoreo, lo de la última clase. Qué es un memory leak, thread leak, qué causó el leak en el ejemplo en clase, cómo se dieron cuenta, qué herramienta usaron

Después Docker, ¿Cómo le explicaría a un PM qué Docker vale la pena? (*Le dije que es un entorno aislado que ya tiene todas las dependencias. Qué ayudaría a migrar y escalar, por ejemplo si migrás a cloud y después no se me ocurrió otra cosa. Me dijo que estuve medio floja con eso*.)

Luego no sql (*Le nombré todos los tipos de bases de datos nosql y le expliqué qué era CAP y sus limitaciones*.)

Microservicios, ¿Qué pros y contras tiene pasarse a microservicios? *(Ahí dije la cantidad de deploys aumenta, pero no sontan grandes comparador con un monolito. Permite usar distintos tipos de tecnologías, aunque puede ser unsa desventaja porque si se usan cosas muy raras, luego no hay quién mantenga. Es más complicado a nivel management)*

Final:

- cloud: me puso un ej en que tenia los servidores en casa y queria pasarse a cloud, que tenia que tener en cuenta

- microservicios: que variables tengo que tener en cuenta si paso de monolito a microservicios (*le dije el tamaño del equipo y la robustes de la solucion esto ultimo me dijo que no es tan asi*) y que ventajas y desventajas a tener en cuenta

- nosql: que es el teorema CAP y que le nombre las BDs que hay