Sziasztok! Köszöntök mindenkit eme előadások én Buga Péter vagyok és Maliosz Markosz / Simon Csaba volt a konzulens, sajnos ők most nincsenek itt de megbeszéltem velük hogy írnak véleményt.

Szóval, mi is volt a feladatom? Az volt hogy egy streaming szolgáltatást alakítsak ki Kubernetes környezetben microszolgáltatás architektúrával. Ez a teljes rendszert magába foglalja mind a streaming megoldástól a felhasználó kezelésen át a kliens megírásáig.

Egy picit szeretnék beszélni a streaming szolgáltatás jelentőségéről hogy kontextusba helyezzem a feladatot. Napjainkban minden nap emberek milliói használnak valamilyen féle streaming rendszert, legyen ez szórakozásra, mint a sportközvetítések oktatásra mint a COVID idején illetve mindennaposak hogy a különbüző vállalatok online tartanak megbeszéléseket, tájékoztatókat

Következőnek az informatikai kihívásokról szeretnék beszélni. Egy a streaming rendszer felépítése komplexxé tud válni mivel működésükhoz elég sok funkciónak kell eggyütt dolghoznia. Ilyen például a kép és hang szimkronizációja, az éppen elő streamek kezelése, a nézettség monitorozása.

Az emberi szokásoknak napi rutinjának megfelelően nagyon hullámzó az erőforrásigénye egy ilyen rendszernek, érthetően valószínűleg több ember fog valamilyen elő adást nézni hétköznap este vagy a hétvégén mint mondjuk munkaidőben.

Ezt a kihívást megoldja ha valamilyen felhőre alapozunk, hiszen nem kell megvenni a fizikai hardware-t és akkor kihasználatlanul fut az esetek egy részében hanem épp annyi erőforrást igénylünk dinamikusan amennyire szükségünk van. Ez az üzemeltetés árát is csökkenti.

Ezt a felhő alapú megoldás igényt nagyon jól támogatja a kubernetes, mivel egyrészt konténerekkel dolgozik ami segít a abban hogy bármilyen környezetben tudjon futni a szolgáltatás, illetve az erőforrás menedzselést is támogatja hiszen ha kell akkor tud új konténereket (pod-okat) létrehozni ezeknek a rendeklezésre állását monitorozni és egy service alá szervezni őket automatikusan,

Következőnek az elméleti háttért szeretném kicsit bemutatni, kezdeném a szolgáltatás magjával a stream közvetítéssel, két protokollt használ a server az egyiket a bejövő stream fogadására a másikat pedig a lejátszó felé történő adatátvitelre. Amikor elkezdődik a tream akkor a streaming server-en létrejön egy .m3u8 kitejesztésű file a stream azonosítújával majd ebbe a fileba belekerülnek a kissebb szegmensek elérhetősége. Amikor egy player szeretné lejátszani a videót akkor előszőr elkérni a m3u8-at majd ebből elkezdi lekérni az egyes szegmenseket.

Most egy picit a micoservice architektúráról és szeretnék beszélni. A lényege hogy kissebb részekre bontjuk a monolitikus alkalmazást, ezzel könnyítve a fejlesztést és az így kapott lazán csatolt részeket egyszerűen lecserélhetjük ha kellene. Ezt az architektúrát nagyon egyszerűen lehet implementálni kubernetes segítségével, annyi a dolgunk hogy minden rész modulhoz külön service-t és deployment-et készítünk és a többit mint a skálázást és a modulok közti koomunikáció forgalomirányítását egyszerűen kezelhetjük a kubernetes által adott lehetőségekkel.

Rá is térnék az implementációra. A félév elején tervezett funkciók a streaming service alapjait fedik le mint maga a stream indítása, nézése, az ektív streamek közötti böngészés és pár alap közösségi funkciót is mint a követés és a nézőszámlálás

Következzen az architektóra, a képen látható hogy milyen komponensekre bontottam a szolgáltatást és ezeket milyen logikai rétegekbe lehet szervezni. Kezdjük az elejétől a stream indításától, amikor valaki elindít egy streamet az az RTMP-service-be fog befutni, ez a service felel a hitelesítésért a stream feljegyzéséért az adatbázisba majd a lejátszó kiszolgálásáért is. Következzenek az adatbázisok. Ebből kettő is található egy POSTGRESQL relációs adatbázis három táblával, a felhasználókkal, a streamekkel és a követés kapcsolatokkal, és egy Redis kulcs-erték adatbázis amit cache ként használ a szolgáltatás. A következő réteg az adat réteg, itt három service helyezkedik el az egyik a user-eket kezeli az adatbázisban a másik a streameket és a harmadik pedig az magát az adatbázis migrációkat. A követező réteg az a service réteg ahol az applikáció buisness logic-ja található. Itt vannak megvaéósítva a fő funkciók mint például a hitelesítő komponens, a Streamek kezeléséért felelős logika, a nézőszámlálás, és a követő funkcióü is. A következő réteg pedig a kliens, ez egy Angular alapú weboldal ahol a felhasználók elérik az alkalmazás fukcióit. Végül pedig a clusterben fut még egy ngnix alapú ingress controller ami azért felelős hogy a clusterbe bejövő kéréseket a megfelelő service-hez irányítsa.

De hát hogy is tud stream-et indítani a felhasználó, először belép a weboldalon majd a stream settings-nél generál magának egy stream kulcsot. Ezt beíírja a streaming softwareébe pl OBS és elindítja a streamet (DEMO if time allows)