

flag=asdasdads  
flag=noththis  
wargame  
wargame  
wargamewargamewargamewargamewargamewargame

## elkészítés

1. Egy 5 l-es befőttes üveget jól elmosunk.
  2. Az uborkához felteszünk forrni 4 l vizet. Literenként 1 púpozott evőkanál sót szórunk bele.  
"Akkor jó a lé, ha egy fokkal sósabb, mint ahogy jó lenne." Ha felforrt, hagyjuk langyosra hűlni. Azért kell 4L mert ha nagyon napos az idő, a párolgás miatt sok vizet kell utánatölteni!
  3. Az uborka se kicsi, se ne nagy legyen. Az uborkákat jól megmossuk, a külső szúrószőrököt kézzel vagy (konyhában rendszeresített) körömkefével (szivaccsal) ledörzsöljük. Két végét levágjuk, mert attól kesernyés lenne. Hosszában 2-4 cikkre vágjuk, hogy a lé átjárja. Felénk úgy szokás bevágni, hogy a két végén marad kb 1-1 cm, először az egyik végéről, majd a másik végéről 90°-kal elforgatva.
  4. Megpucoljuk a fokhagymát, megmossuk a kápot.
  5. A kápot felét az üveg aljára tesszük, mellé a fokhagyma felét. Jó szorosan elkezdjük az üveget telepapolni a bevagdosott uborkával. Szépen megfogalmazta ezt Zilahy Ágnes: "uborkával rakjuk tele az üveget, csinosan sorba igazítva", "nyakhajlásig". A tetejére tesszük a kápot másik felét és a fokhagyma maradékát. Felöntjük a sós lével, hogy az ellepje az uborkákat, és rátesszük a kenyér szeletet a tetejére. Maradék levet pótláshoz félretesszük.
  6. Az üveget egy tányéra tesszük (a kifolyó uborkalé ne csúfitson össze semmit), és a tetejére is rakunk egy tányért. 2-3 nap múlva már kész is. A lé mindenkorban tetejéig legyen, naponta pótoljuk, mert ha kilóg a kenyér a léből, akkor penészedni fog, ha meg az uborka lóg ki, akkor kifehéredik és kiszárad. Ha nagyon meleg helyre tesszük, akkor az uborka nagyon megpuhul.
  7. Kóstolás után, ha jónak ítéljük, akkor az uborkát kiszedjük egy tálba, a levét leszűrve ráontjük, és hűtőben tároljuk.
  8. Ha télen is szeretnénk belőle enni, akkor egy csírátlanított, tiszta befőttes üvegbe tesszük az uborkát, a levét ráontjük. Kidunsztoljuk, mint ahogy a befőttet szokás. Tartósítószer sem kell bele.
- elkészítettem: 30 alkalommal

# Mi az a microservice? Újraírtuk az Admin.SCH-t

Szerző: [Schulcz Ferenc](#)

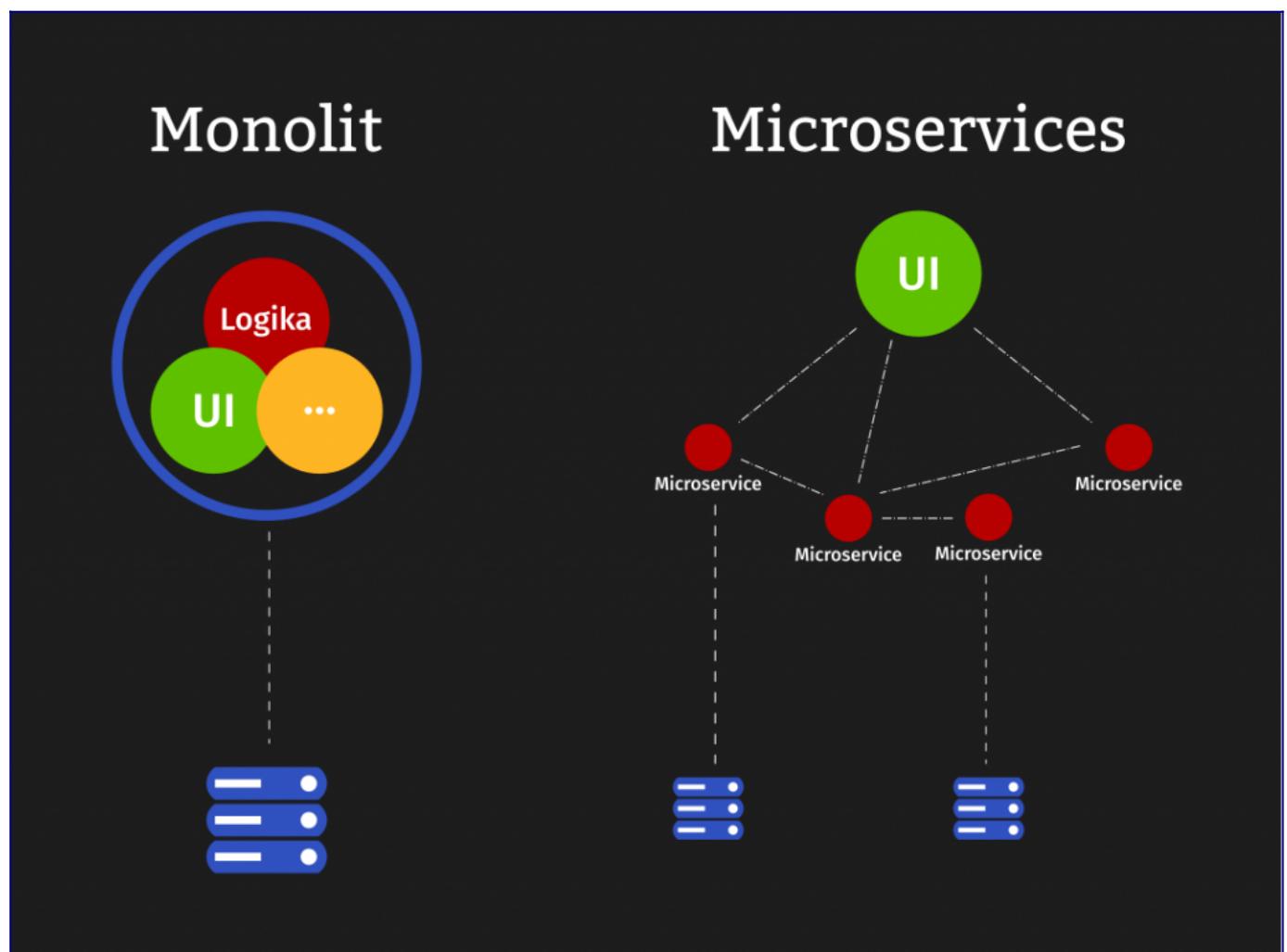
Az [Admin.SCH](#) az az oldal, ahol a kollégisták kezelhetnek mindenet, ami a Házban belüli internevezésekkel kapcsolatos: IP címet regisztrálhatnak, újabb eszközöket adhatnak hozzá, vagy bejelenthetik a szobai access point-jukat. Ez egy olyan weboldal, amit minden kollégista használt már,

és életbevágó szolgáltatásokat tudnak itt beállítani – mondhatjuk tehát, hogy kritikus rendszerről van szó.

Az oldalt kiszolgáló backend a közelmúltban elkezdett elavulni, ráadásul a dokumentáció problémái és architekturális sajátosságai miatt nem tudtuk ésszerűen továbbfejleszteni. Nem volt hát más választásunk, mint hogy újraírjuk.

A korábbi Admin.SCH egy monolit webalkalmazás volt, ami azt jelenti, hogy a szoftver összes funkciója (IP cím regisztráció, Active Directory integráció, stb.) egyetlen “csomag” részét képezi. Ez a hagyományos megközelítés, egy programozás házinál is ezt használjuk: hiába bontjuk a megoldásunkat ezernyi .c és .h fájlra vagy osztályra, a végén mégis egyetlen bináris, azaz egy monolit készül belőle. Egy ilyen alkalmazást könnyebb fejleszteni, könnyebb tesztelni, hiszen csak egyetlen programot kell elindítani és ellenőrizni, és könnyebb deploy-olni, hiszen csak egyetlen dolgot kell felmásolni egy szerverre, és márás éles lehet a szolgáltatás.

Ennek viszont ára is van: a projekt növekvő kódbazisával az egyre átláthatatlanabb lehet, minden frissítéskor az egészet kell újratelepíteni, és egy hiba akár az egész szolgáltatást is térdre kényszerítheti.



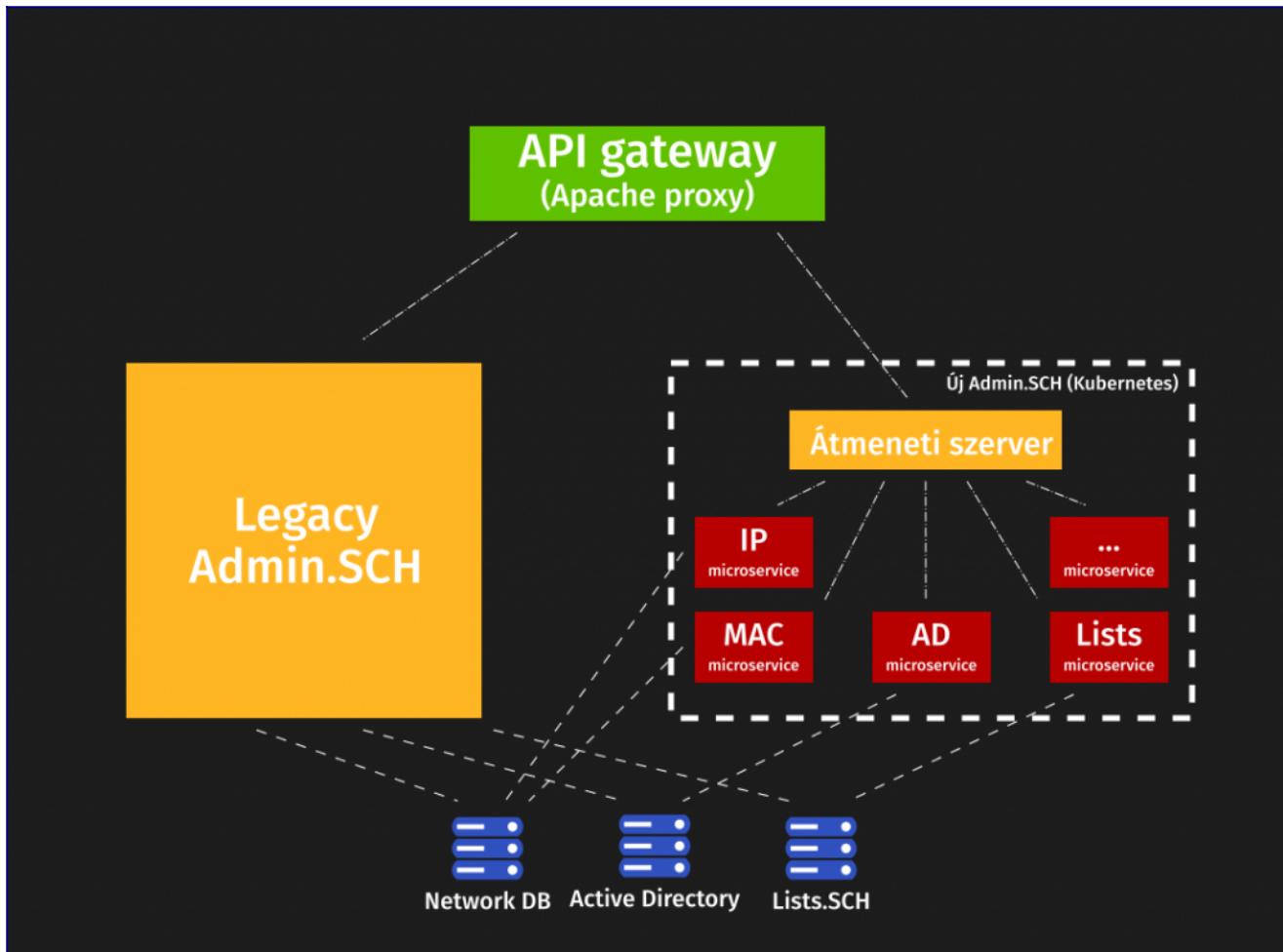
Monolit rendszer vs. microservice architektúra

Úgy döntöttünk, hogy az új rendszert már microservice-ek segítségével építjük fel: nem egy nagy szolgáltatást készítünk, hanem több kicsit. Ezeknek a kicsi szoftvereknek aztán kicsi, meghatározott felelőssége is lehet – van például, ami az IP-k regisztrációját teszi lehetővé, van, amivel a saját AP-kat lehet bejelenteni, és így tovább. Ezzel nagyrészt megoldottuk az átláthatatlan kód problémáját, mert apró, önálló programokkal dolgozunk, amik mindenkorán könnyen áttekinthetők maradnak. Lehetővé vált az is, hogy a komponensek külön fejlődjenek, ami különösen előnyös egy olyan projektnél, amin sok egyetemista dolgozik, mert mindenki a saját ütemében haladhat. A későbbi bővítés is egyszerű, hiszen az új szolgáltatások hozzáadásához nem kell belenyúlni az eddigi kódba. További előny, hogy a KSZK egyéb, a hálózat üzemeltetéséhez kapcsolódó, már létező szolgáltatásait is bele lehet vonni az Admin.SCH-ba.

A microservice architektúra viszont több hátránya is jár. Egyszerre a szolgáltatások közötti kommunikáció miatt előfordul, hogy ha az egyik szolgáltatás interfésze megváltozik, akkor több másik szolgáltatásba is át kell vezetni a változtatásokat. Ezen a problémán segít az API verziózás és az API dokumentációs eszközök használata (mint például az OpenAPI/Swagger). Az alkalmazás futtatását jelentősen bonyolíthatja, hogy sok kisebb szolgáltatásból áll, szóval szükségünk volt valamilyen rendszerre, ami a microservice-eket menedzseli.

A microservice-eket konténerekbe szerveztük, és a KSZK Kubernetes klaszterébe telepítettük. Az ebben futó Docker konténerek könnyedén példányosítható, leállítható és mozgatható process-ek, amikben egy-egy microservice fut. Megtalálható bennük minden, a futtatáshoz kellő dependencia, könnyen futathatók, és a microservice-ek szeparációja is megoldott. A Kubernetes pedig a konténerek menedzselését, és a közöttük lévő kommunikációt biztosítja, azaz orkesztrálja őket. A konténerek Docker image-ekből készülnek, amiket [git.sch.bme.hu](https://git.sch.bme.hu) szolgáltatásba épített Continuous Integration automatikusan előállít.

Az újraírás első ütemének célja az volt, hogy a kollégisták által használt funkciók kerüljenek át az új rendszerbe. Ez a cél meg is valósult, így jelenleg a két rendszer párhuzamosan fut. A legtöbb kollégista csak az új rendszerrel fog találkozni, ami teljesen különálló a régitől, viszont a felület át lett emelve, ezért csak kisebb átalakításokra volt most rajta lehetőség. A következő félévekben át szeretnénk alakítani a felületet és új funkciókkal bővíteni az új rendszert.



A régi és az új rendszer együttétele

Ha valamilyen hibát vagy szokatlan működést tapasztaltok az oldal használata közben, akkor adjatok fel ticketet a [support.sch.bme.hu](https://support.sch.bme.hu) weboldalon!