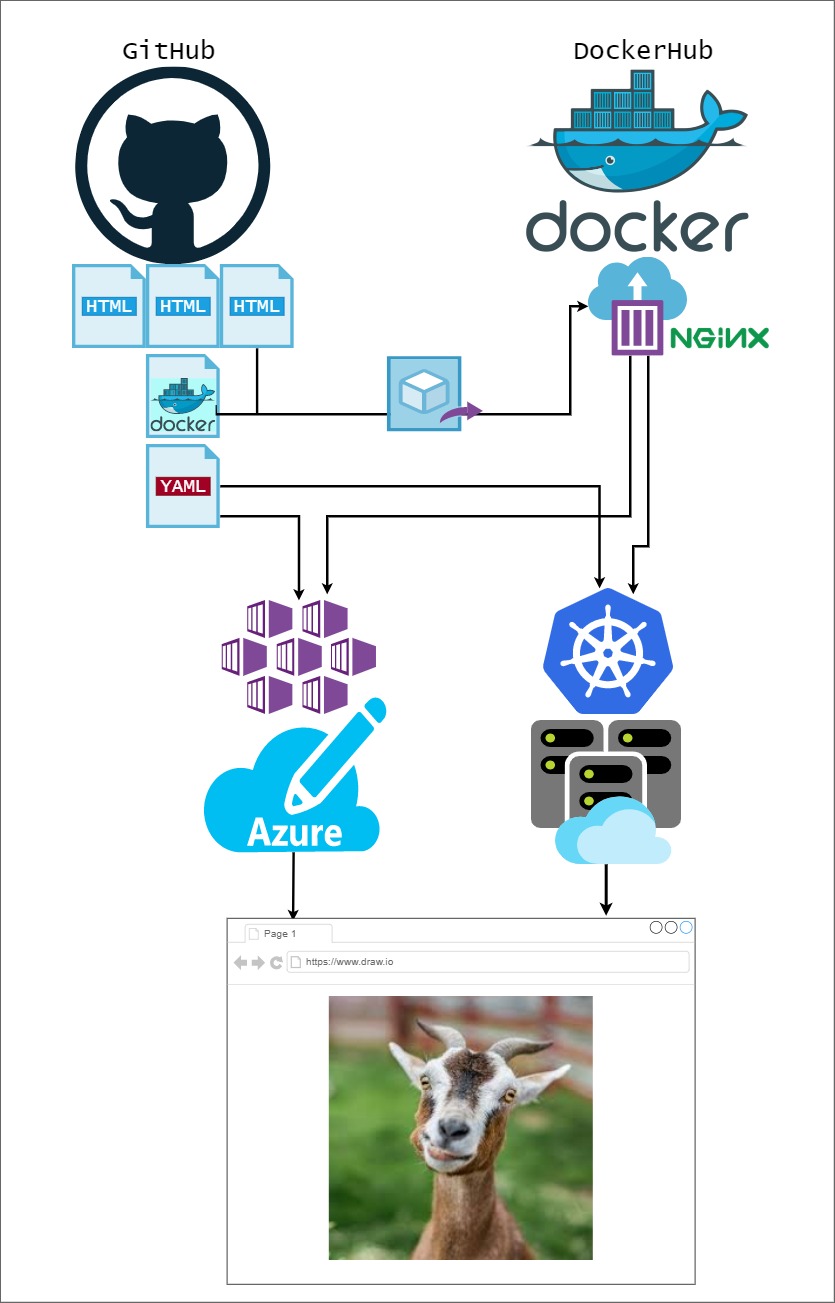
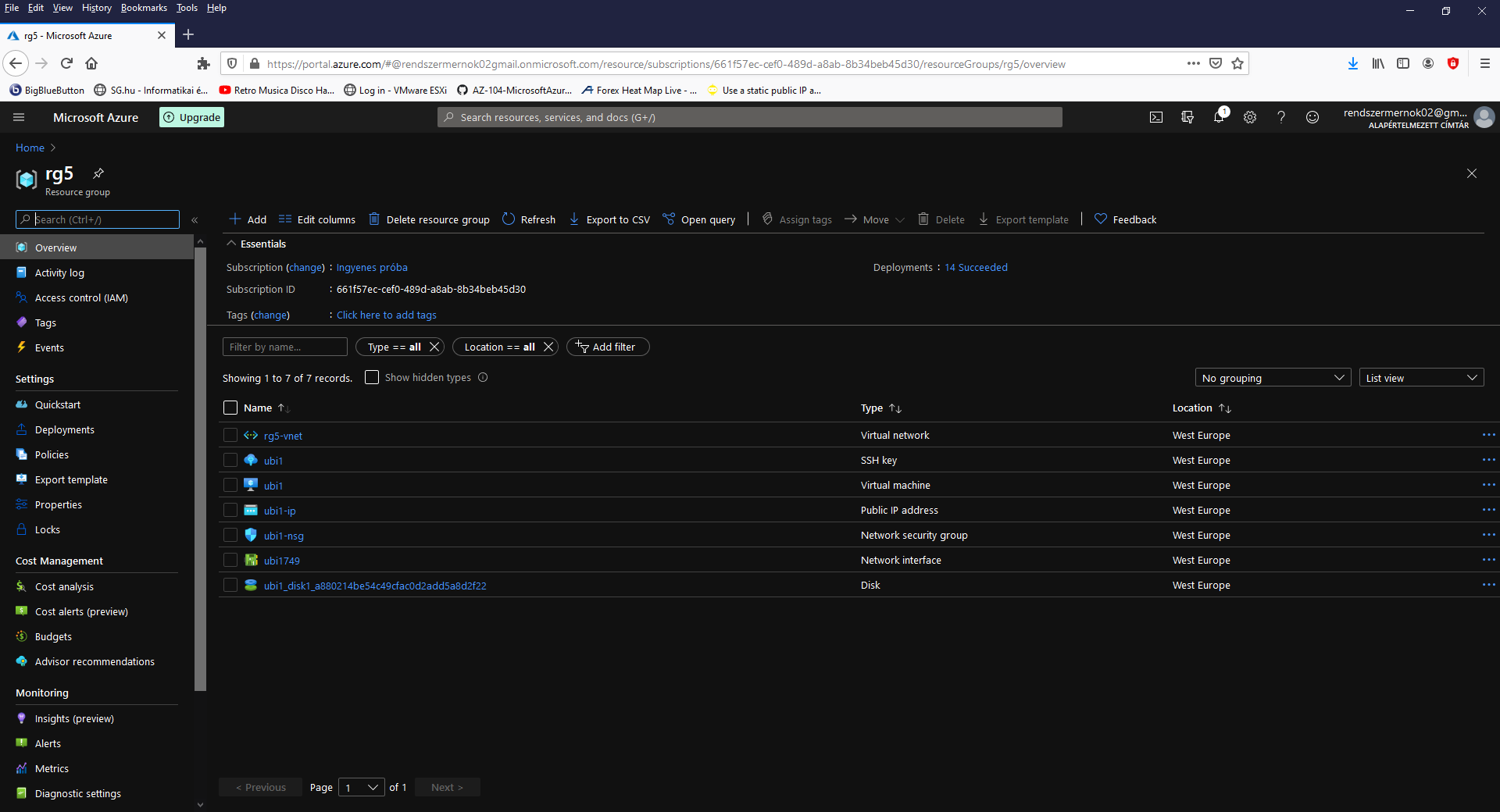
# Webserver futtatása Azure Kubernetes-ben

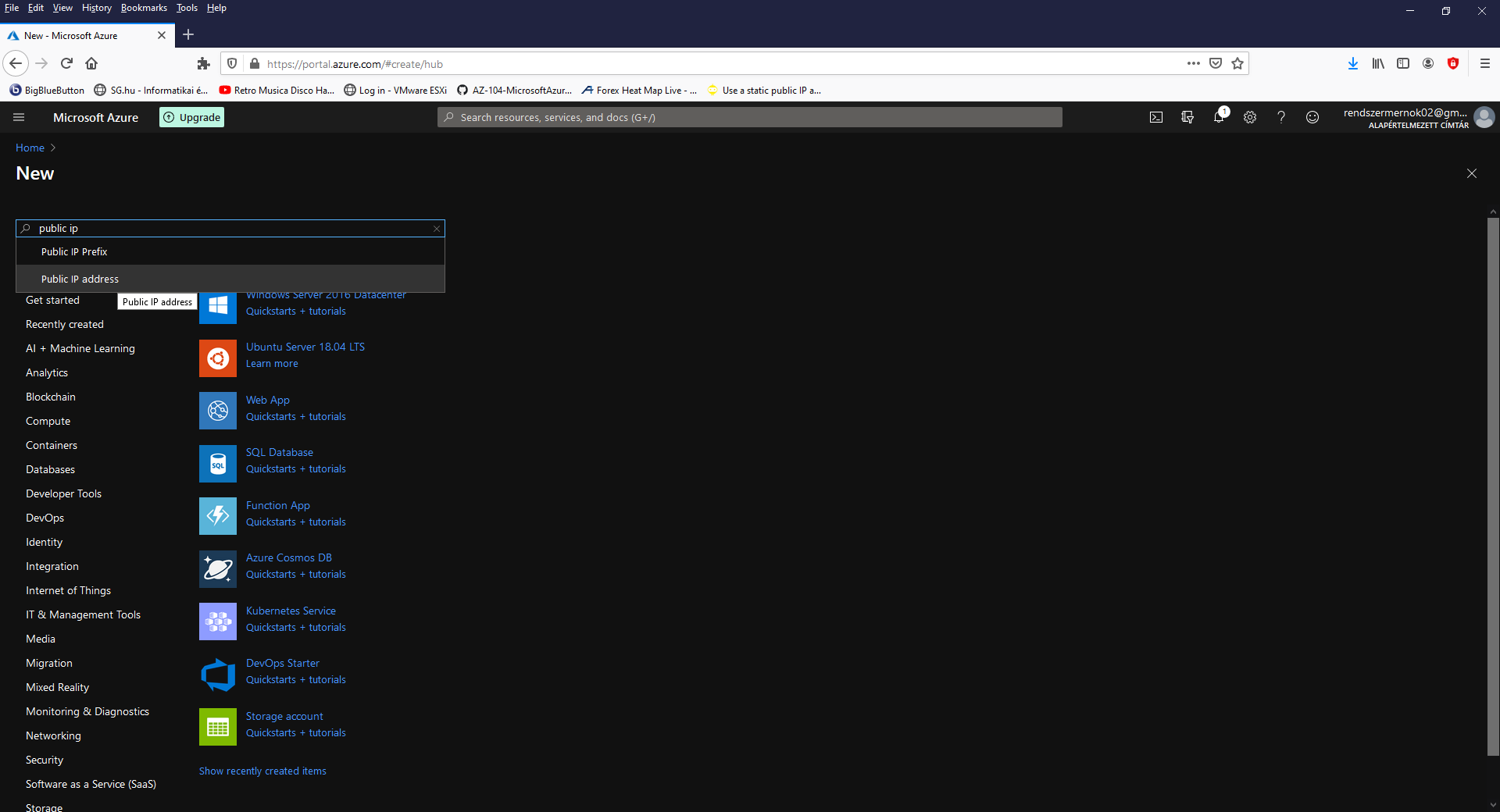


## Publikus IP cím létrehozása

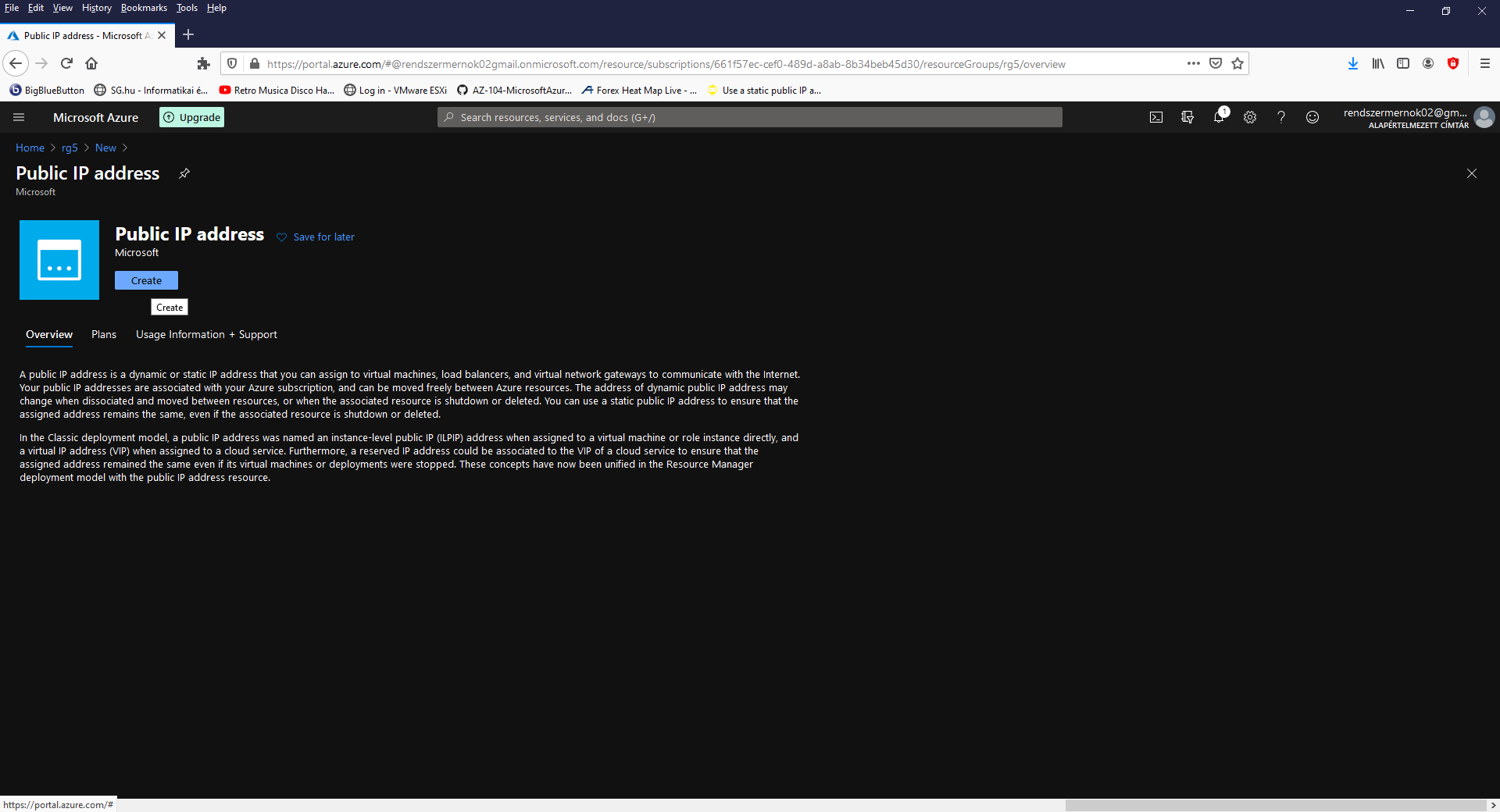
Van egy rg5 nevű „resource group”-unk, vagy ha nincs, létrehozunk egyet.



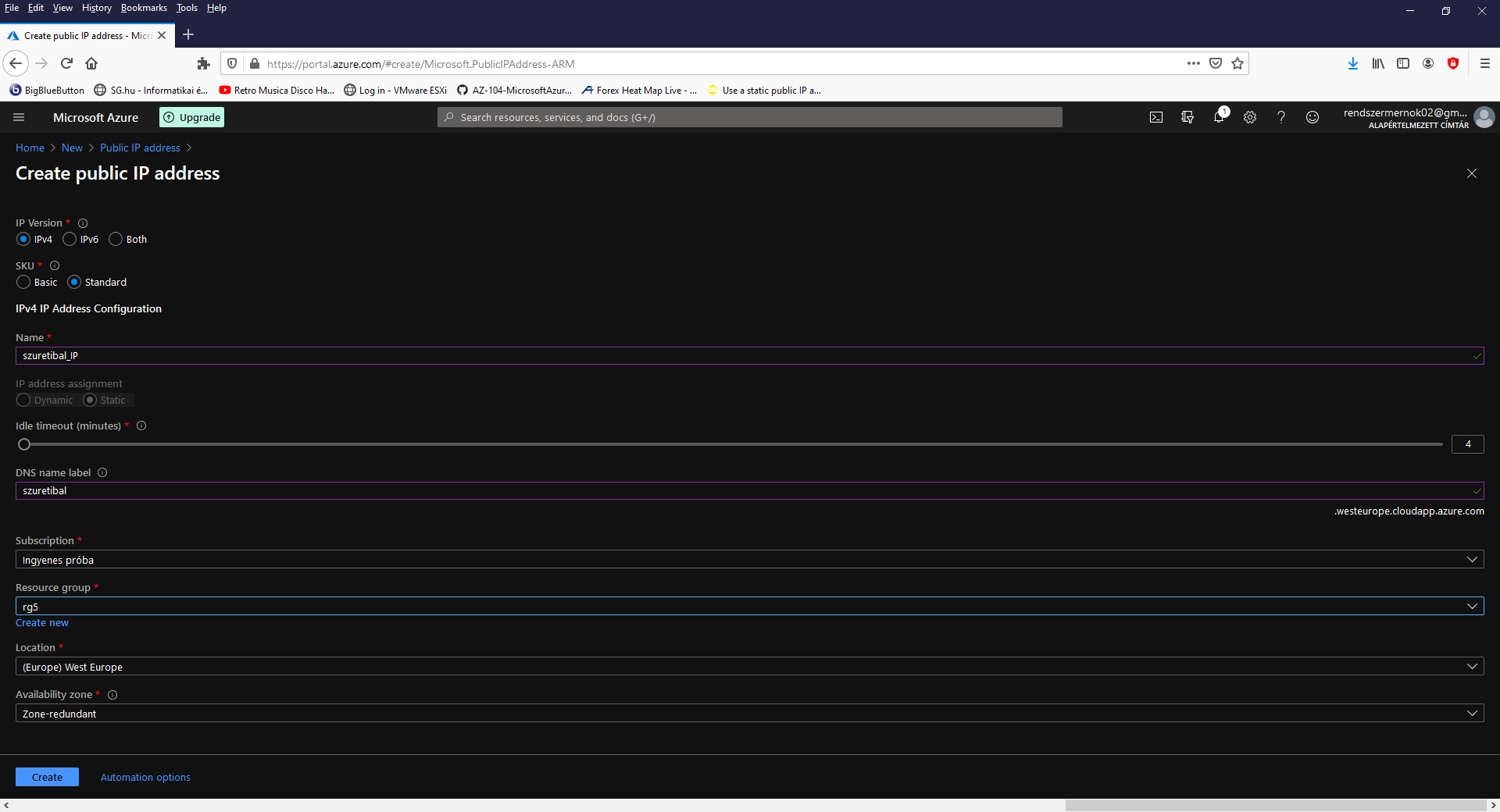
Az új erőforrásoknál kikeressük a „Public IP address”-t



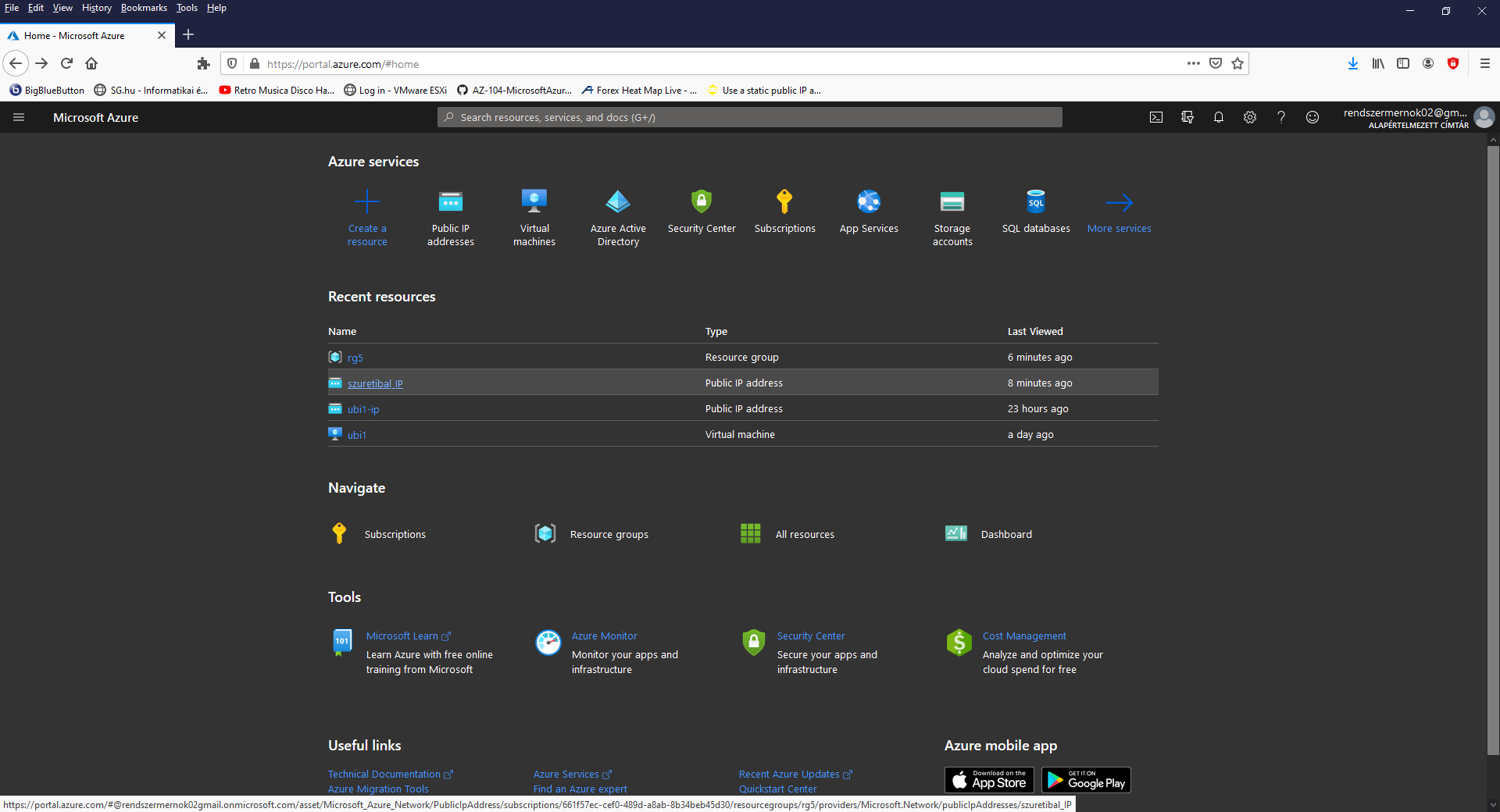
Majd megnyomjuk a „Create” gombot



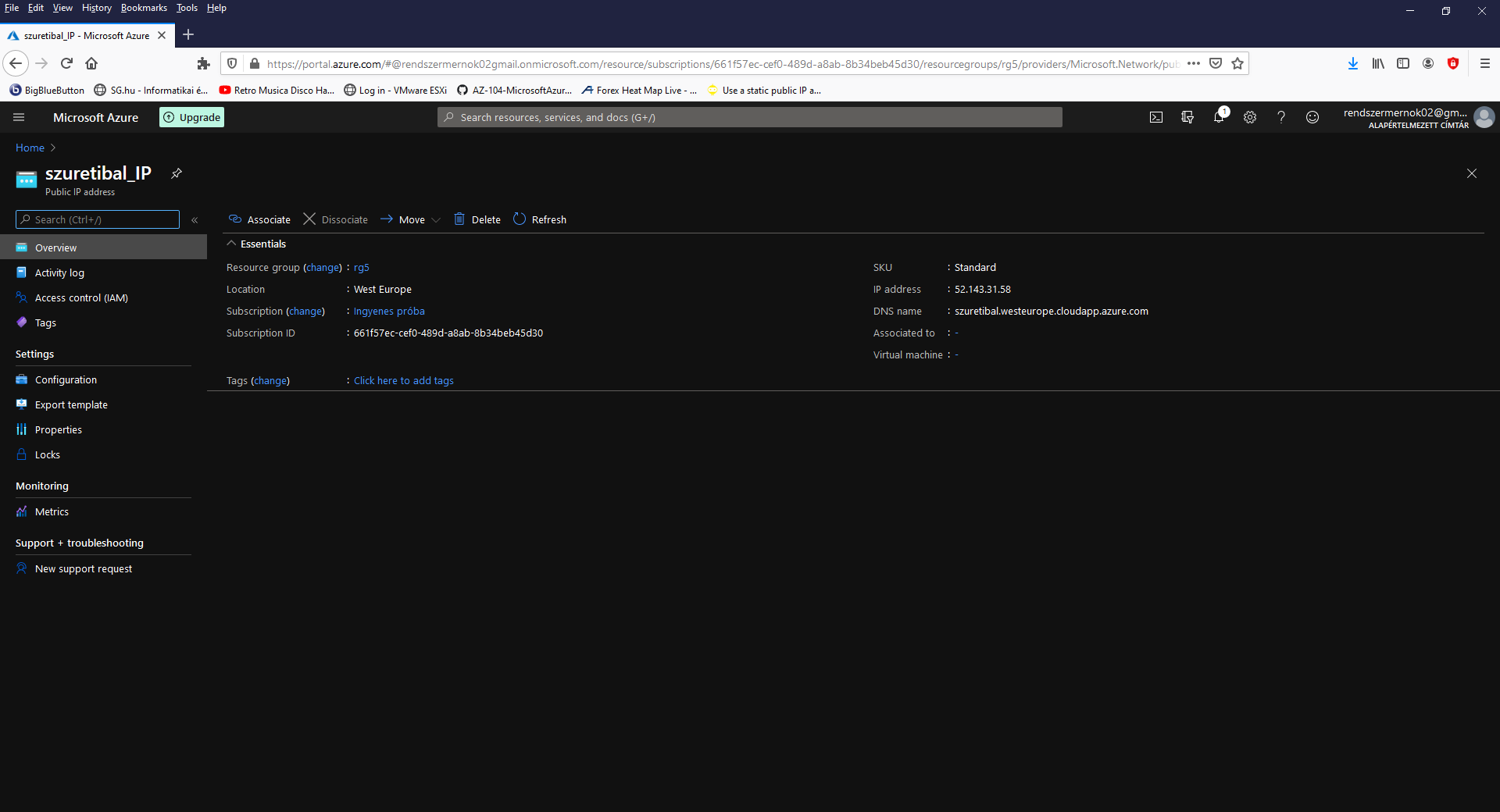
Beírjuk a szükséges adatokat (SKU -> Standard!!!).



Létrejött az IP címünk. Nézzük meg a „Go to resource” gombra, vagy az IP cím előzőleg megadott nevére klikkelve!

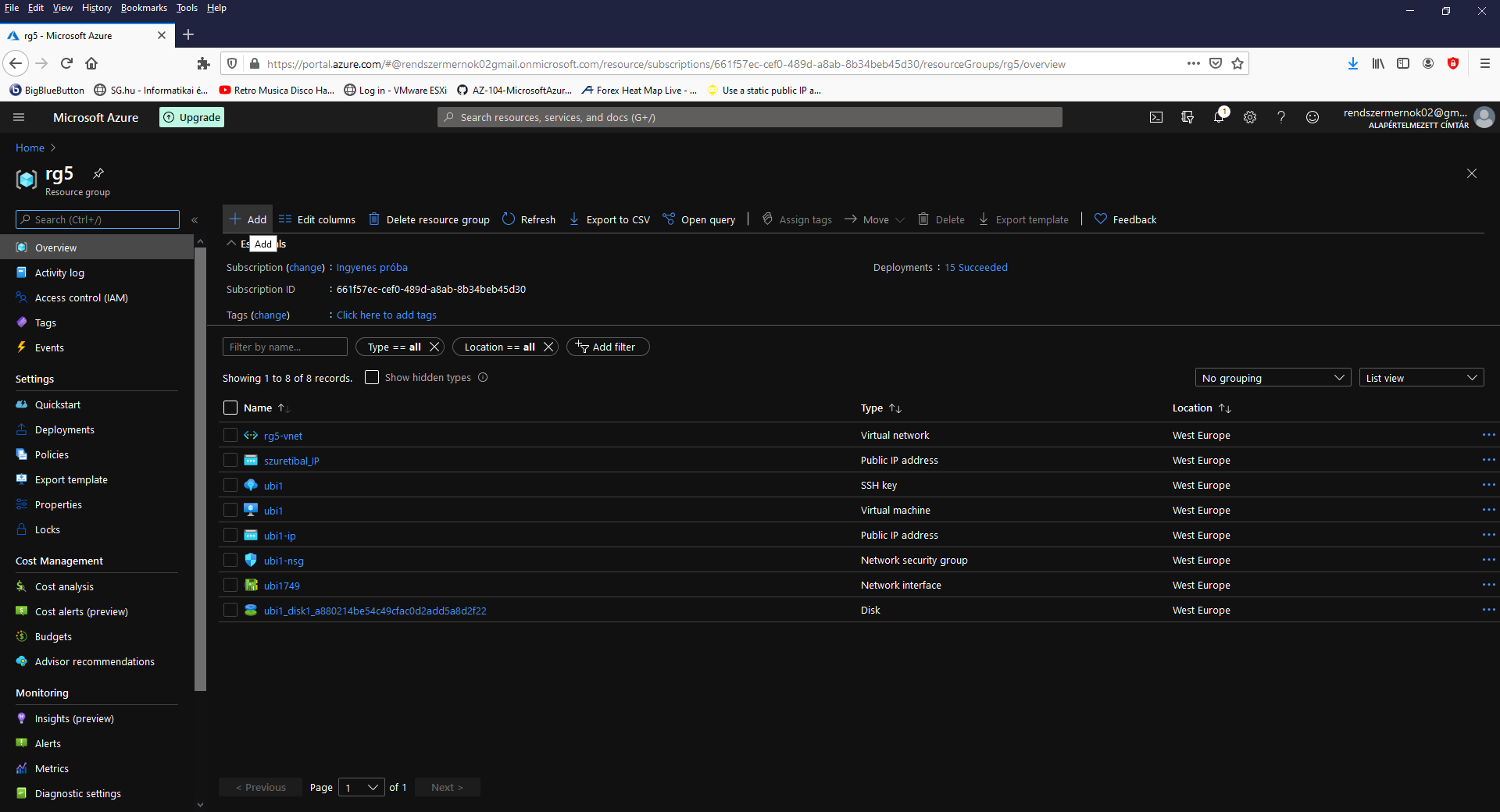


Kaptunk egy publikus IP címet, és hozzátartozó domain nevet, ha kértünk.

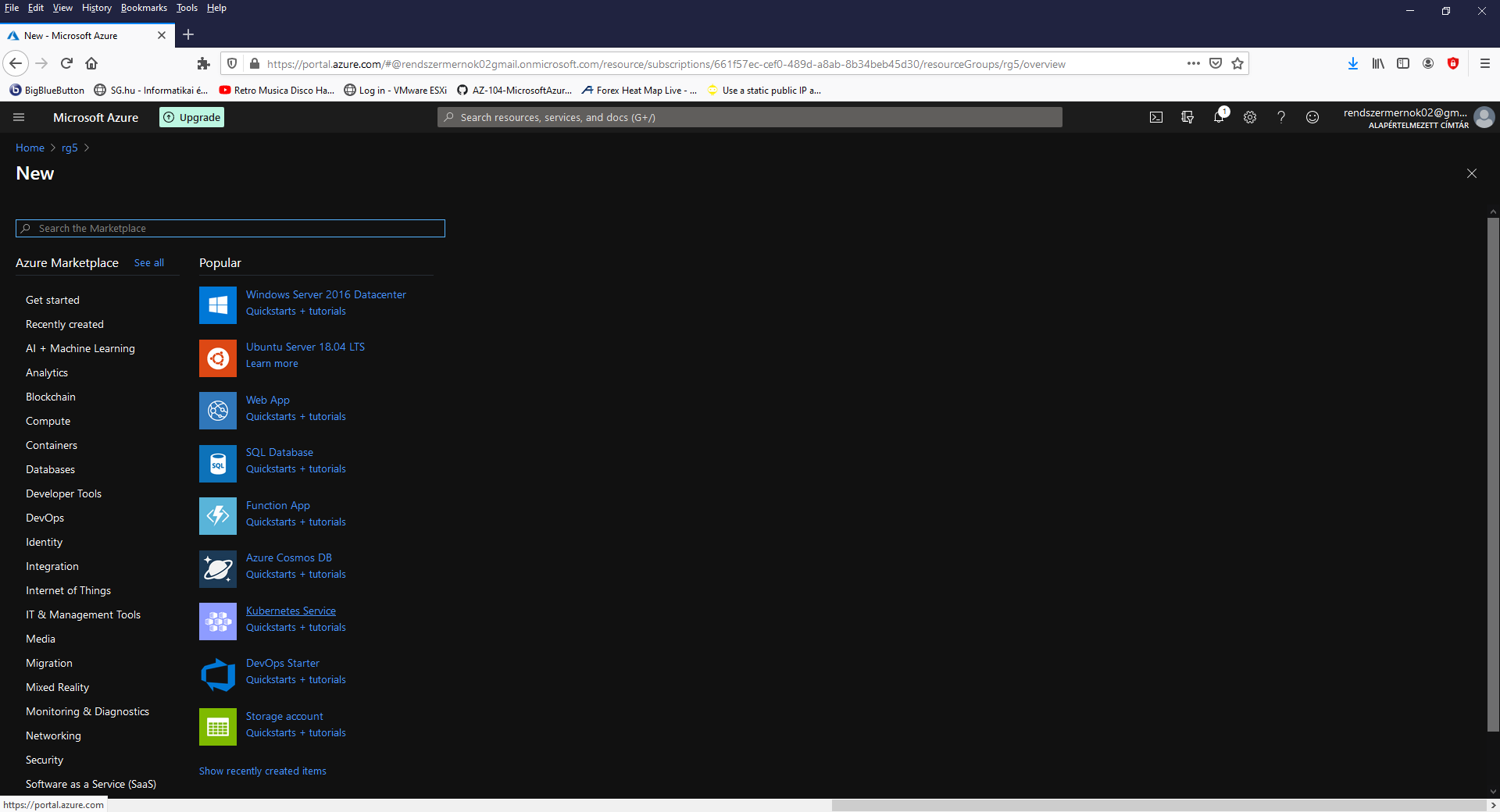


## Kubernetes service indítása

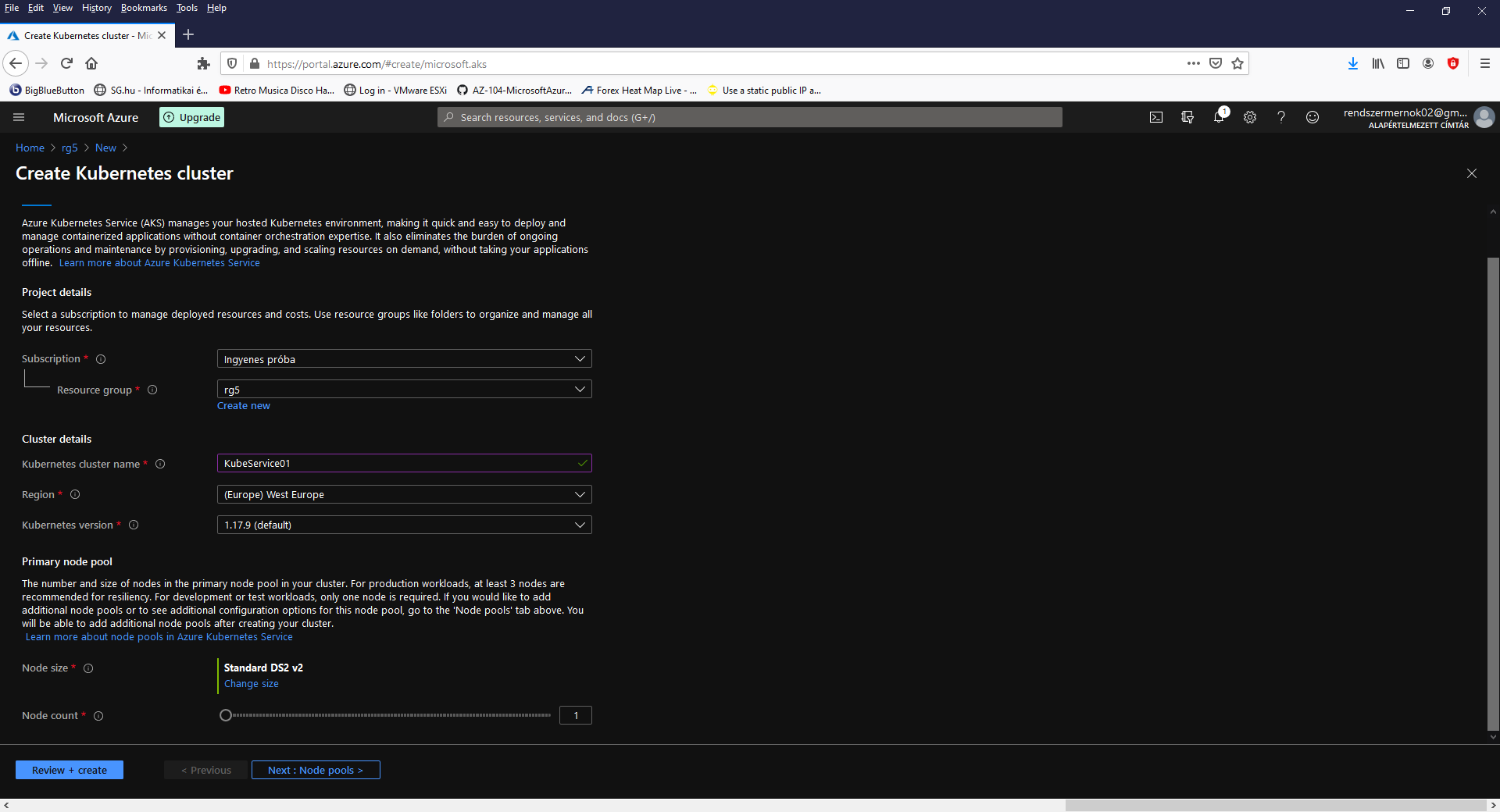
Lépjünk be a kiválasztott resource group-ba és nyomjunk az „Add” gombra!



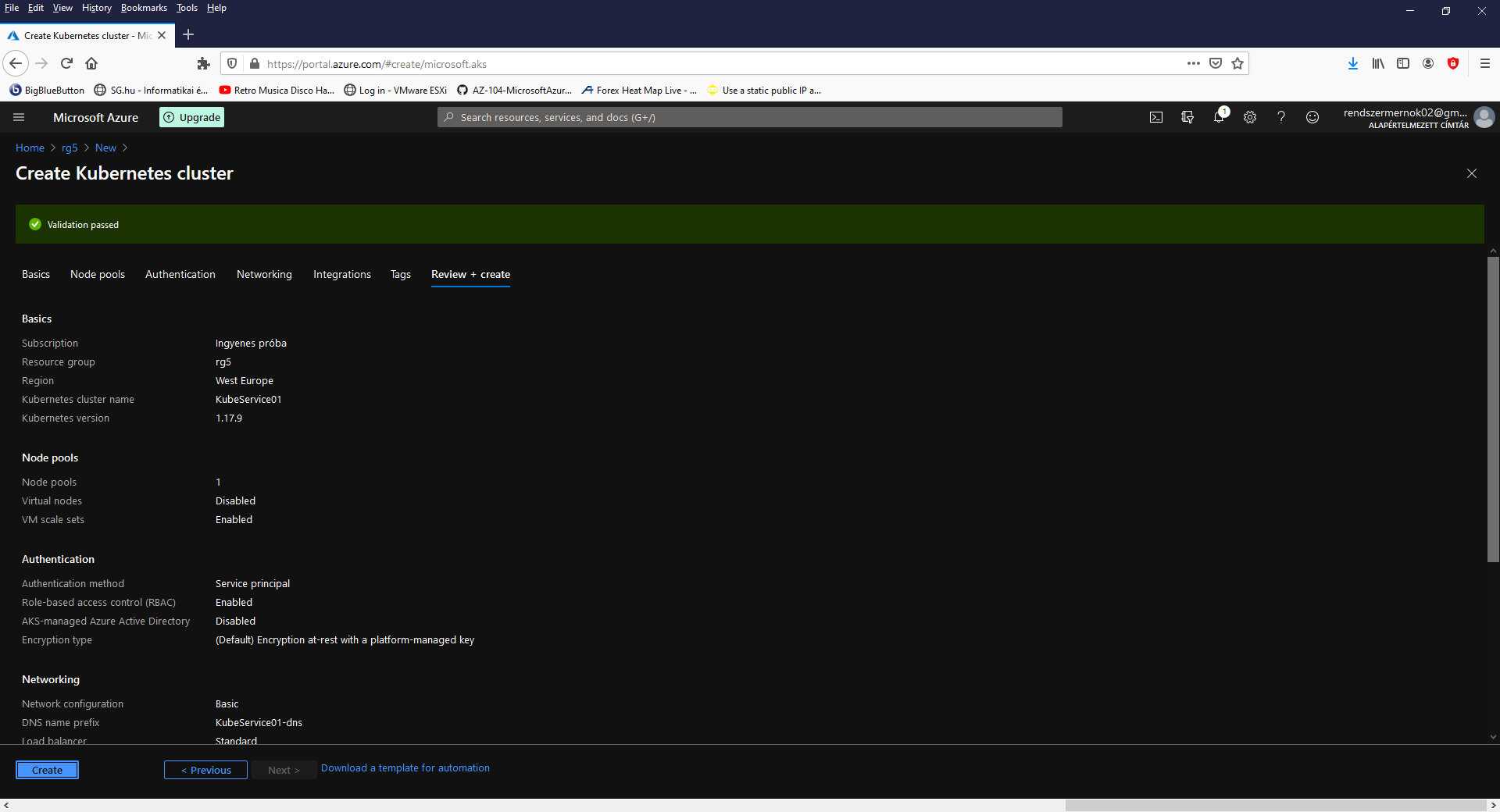
Válasszuk a Kubernetice Service-t!



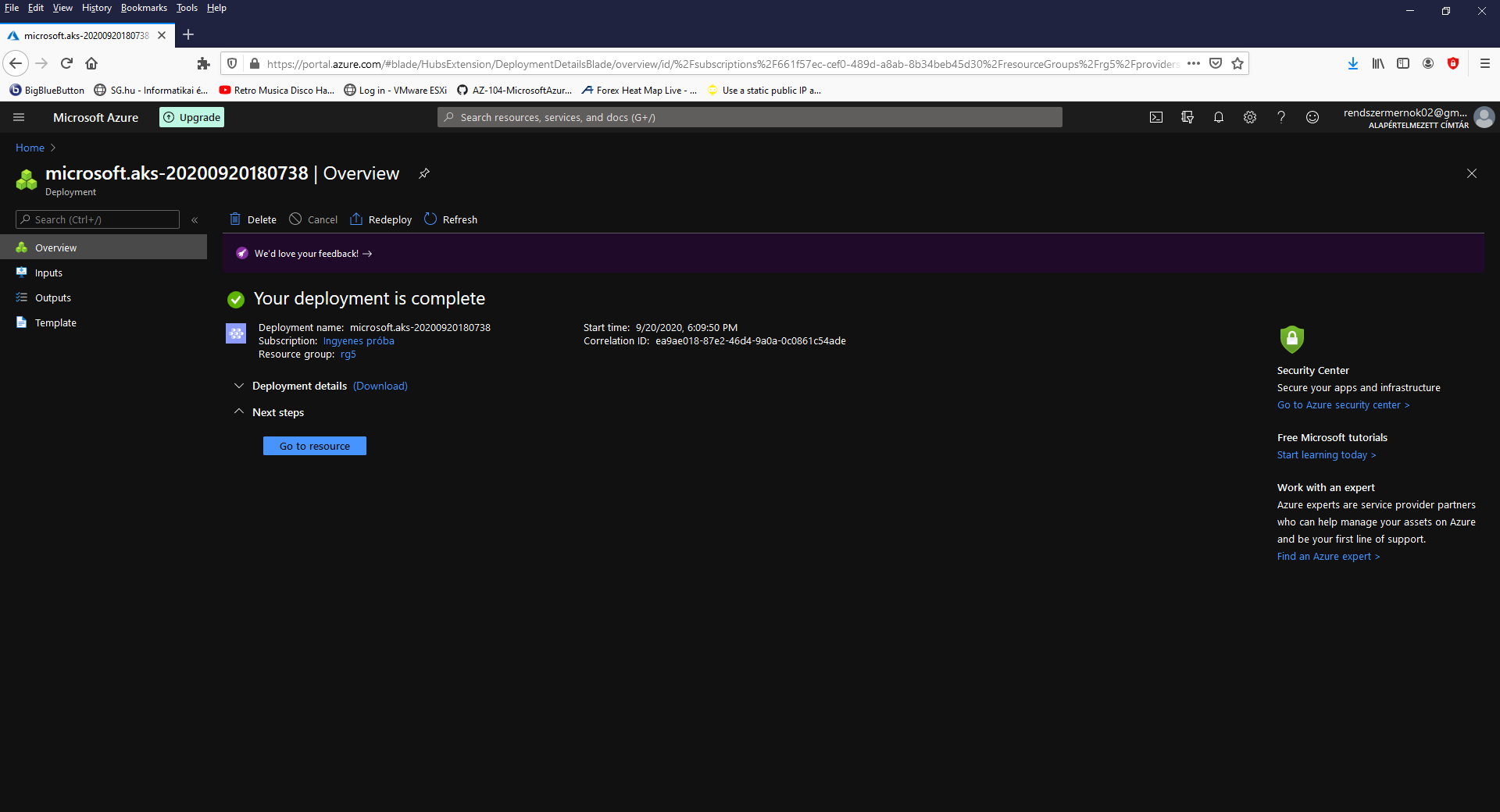
Írjuk be a kért paramétereket, majd alul nyomjunk a „Review + Create”-re!



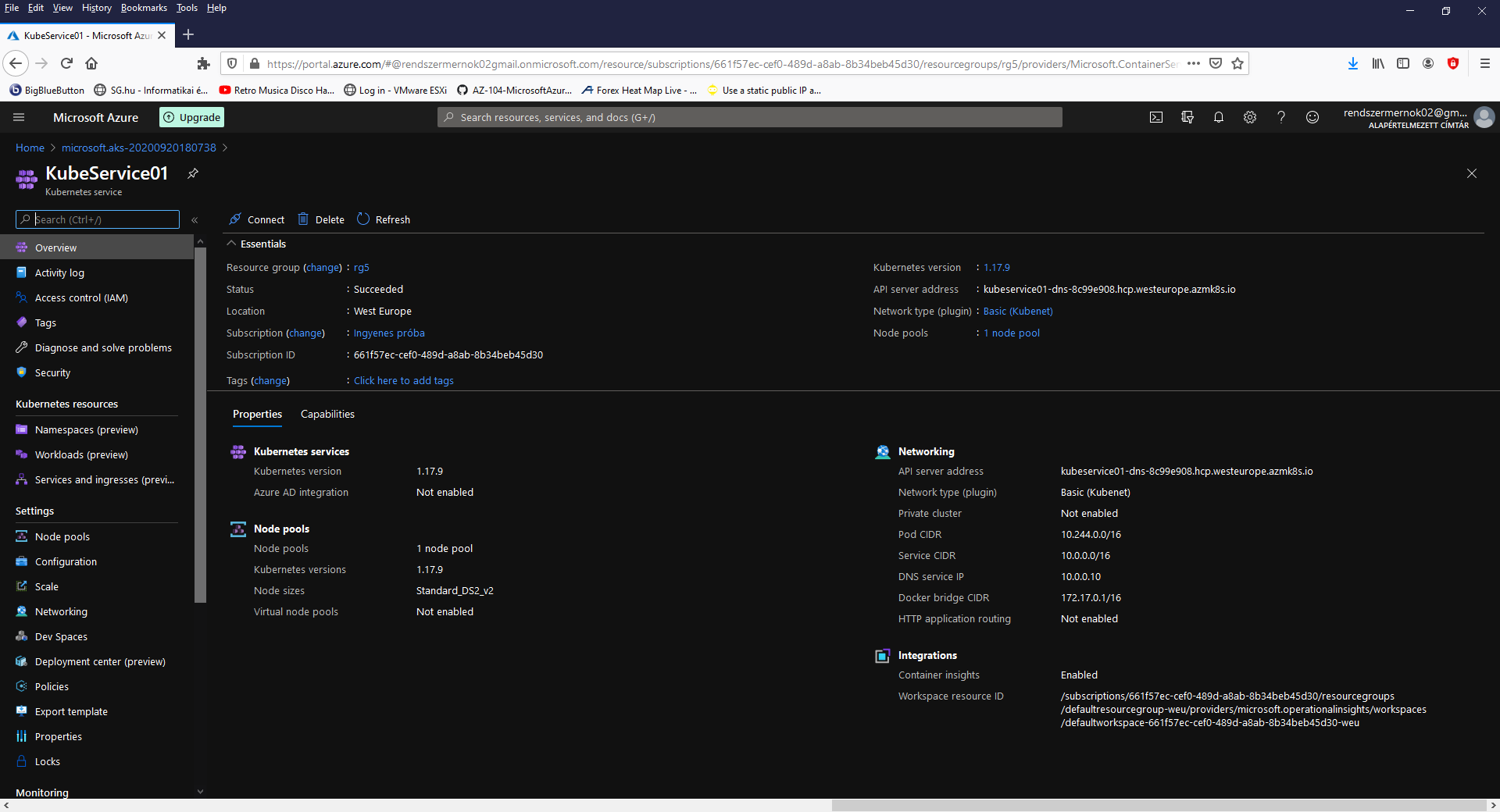
Validálás után létrehozhatjuk a klasztert -> „Create”. Pár perc alatt elindul a Kubernetes klaszterünk.



„Go to resource”-al lépjünk be a klaszterbe!



A Kubernetes klaszter használatra kész.



## Deployment indítása

Az alábbi konfigurációs fájlt töltsük le és írjuk át, ha szükséges:

<https://github.com/rendszermernok02/szuretibal/blob/master/TestHtml01.yaml>

apiVersion: apps/v1

kind: Deployment

metadata:

name: testhtml-deployment

labels:

app: testhtml

spec:

replicas: 2

selector:

matchLabels:

app: testhtml

template:

metadata:

labels:

app: testhtml

spec:

containers:

- name: testhtml

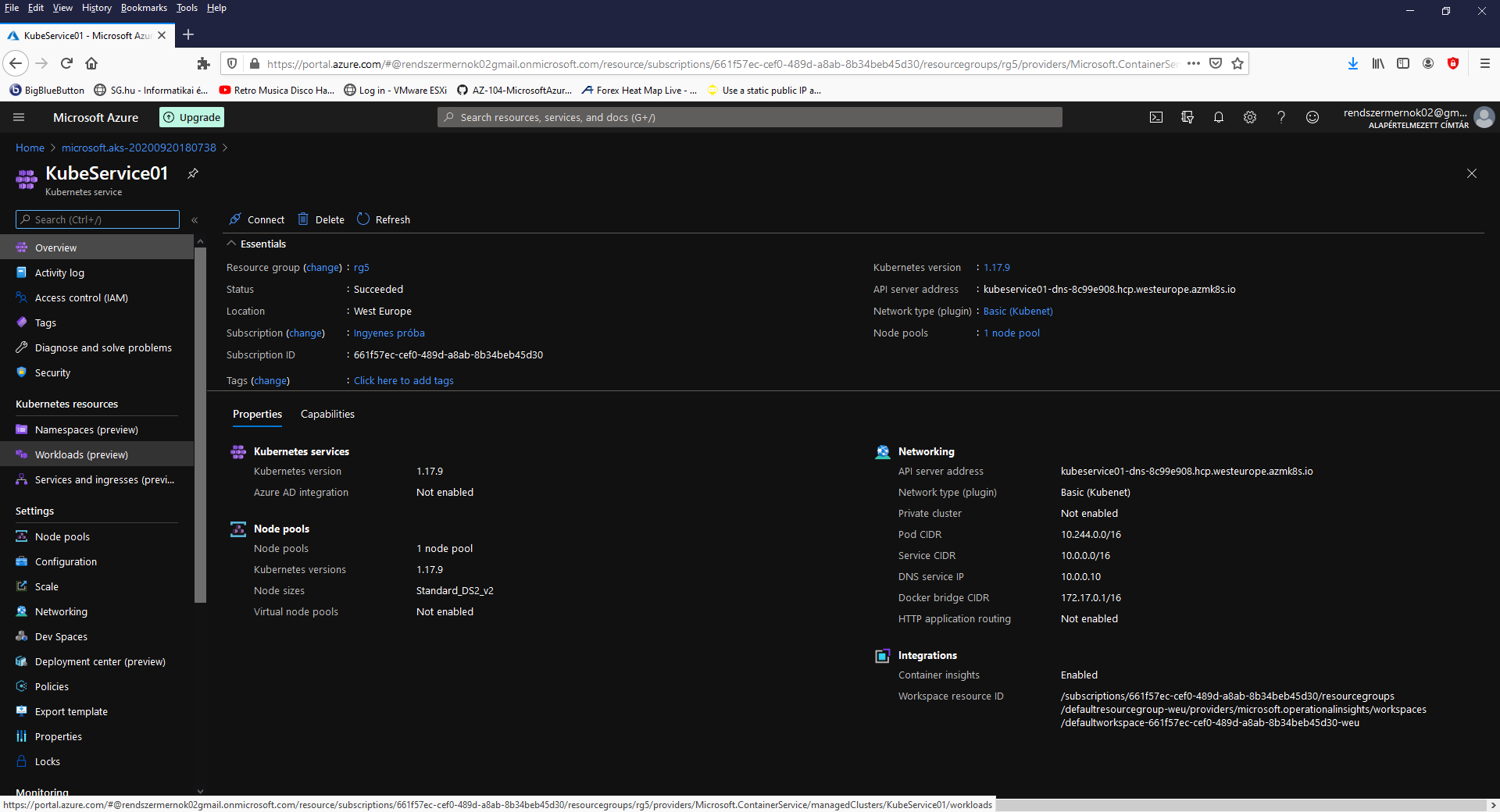
image: rendszermernok02/szuretibal

ports:

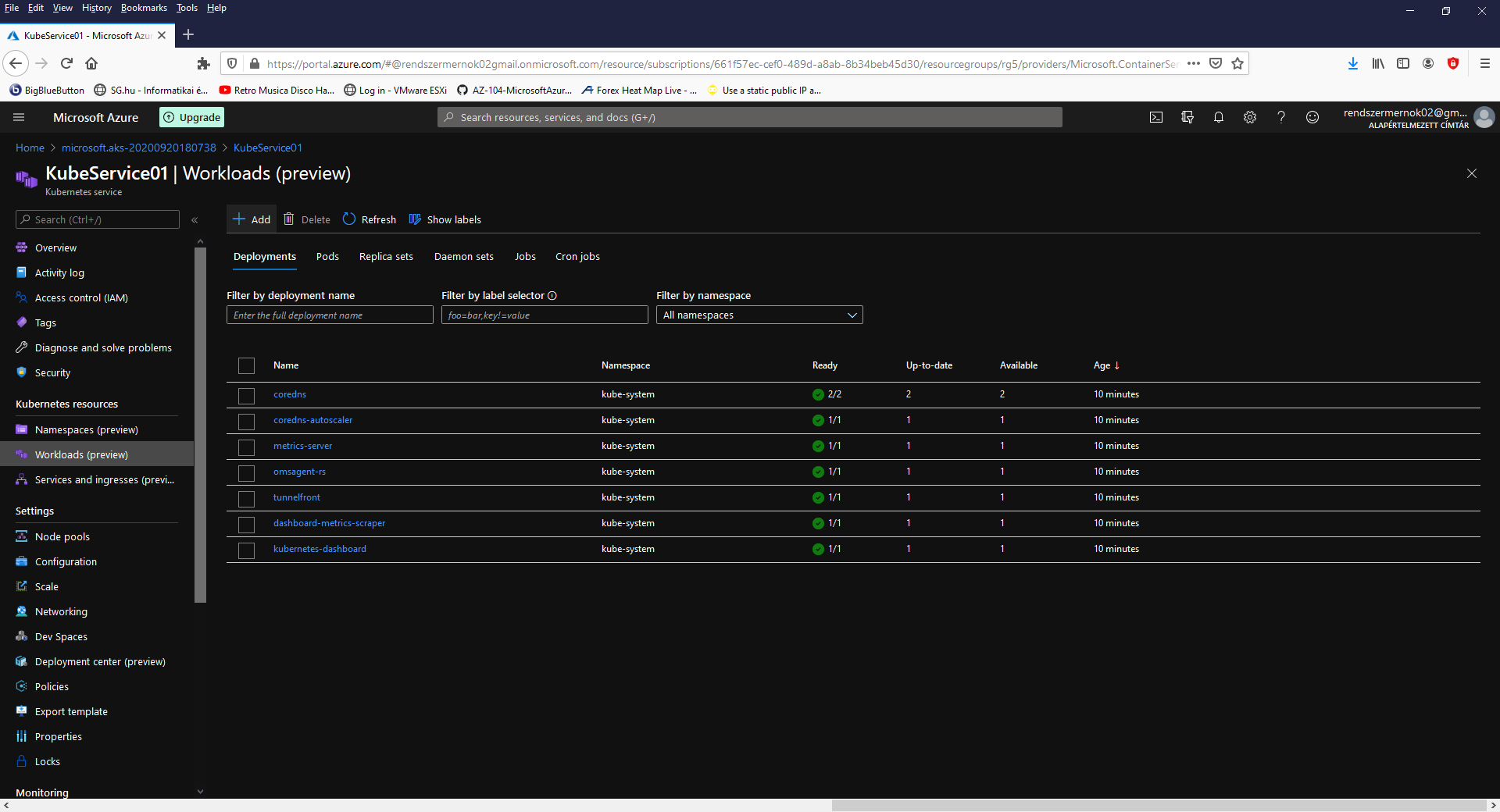
- containerPort: 80

A DockerHub-ról letölti a „rendszermernok02/szuretibal” Nginx alapú konténert, majd 2 pod-ot índít belőle („replicas: 2”).

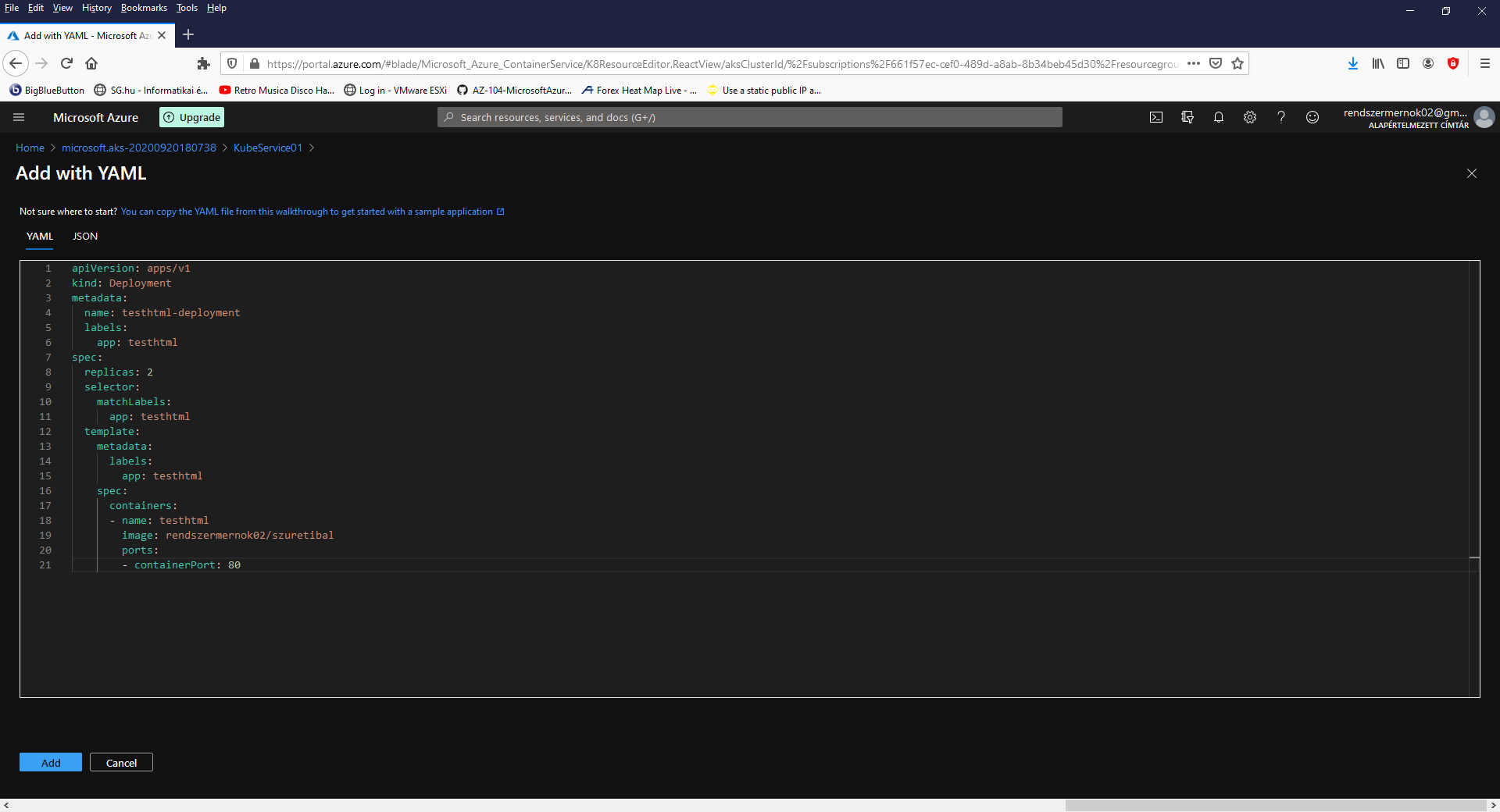
Lépjünk a „workloads” menübe!



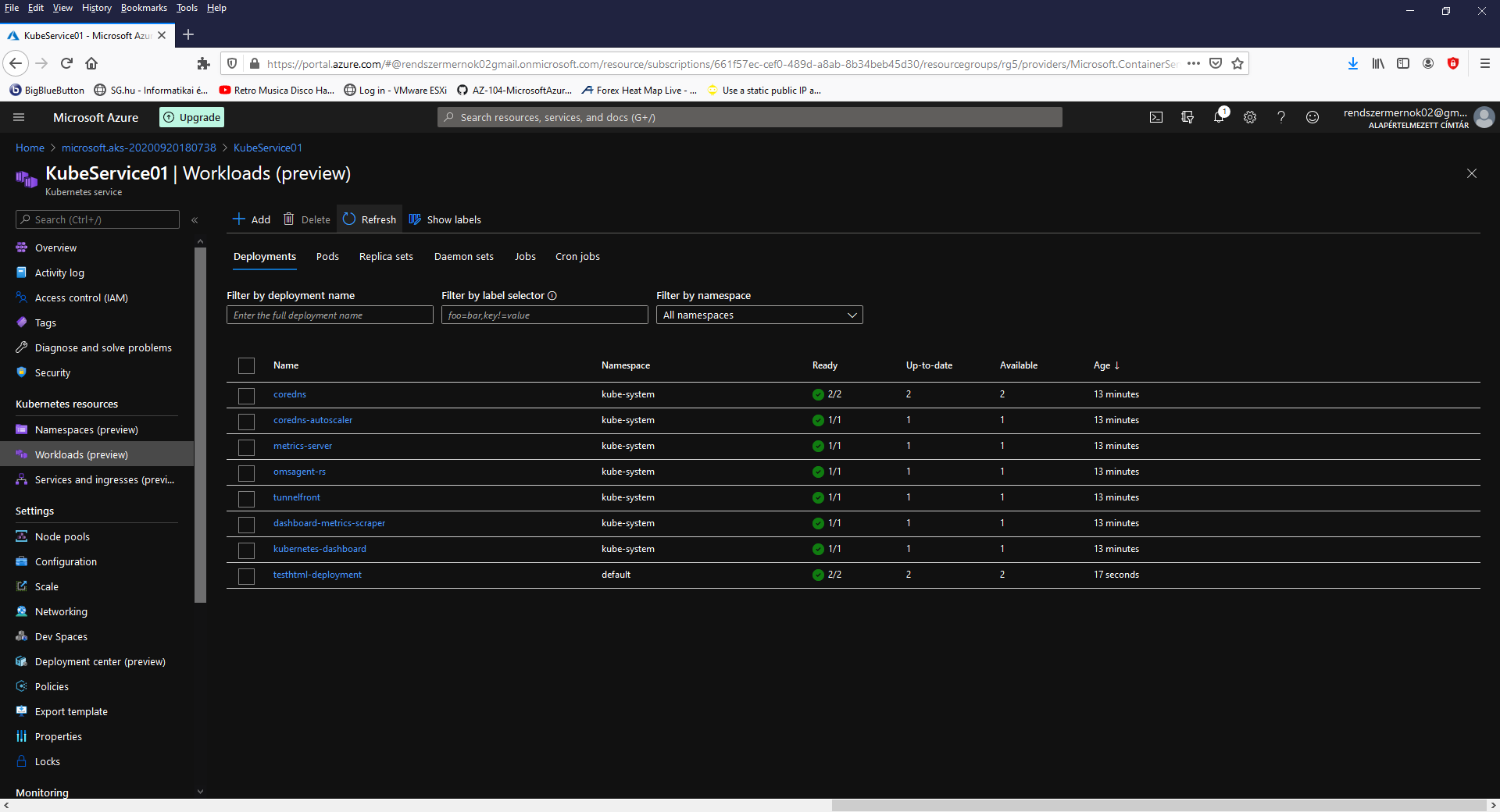
Az „Add” megnyomása után írjuk a be a fenti „yaml” fájlt!



Az „Add” gombra klikkelve a yaml fájlban leírt pod-ok elindulnak a klaszterben.



1-2 perc és elindul:



## Load balancer service hozzárendelésével publikus IP cím és domain név biztosítása az alkalmazásnak.

Először be kell állítani a megfelelő jogosultságokat a Kubernetes klaszter és a publikus IP cím között.

Ehhez le kell futtatni a „Cloud shell”-ben az alábbiakat:

RESOURCE\_GROUP=rg5

AKS\_CLUSTER=KubeService01

az aks get-credentials --resource-group $RESOURCE\_GROUP --name $AKS\_CLUSTER

# Enter your details below.

PIP\_RESOURCE\_GROUP=rg5

AKS\_RESOURCE\_GROUP=rg5

AKS\_CLUSTER\_NAME='KubeService01'

# Do not change anything below this line

CLIENT\_ID=$(az aks show --resource-group $AKS\_RESOURCE\_GROUP --name $AKS\_CLUSTER\_NAME --query "servicePrincipalProfile.clientId" --output tsv)

SUB\_ID=$(az account show --query "id" --output tsv)

az role assignment create\

    --assignee $CLIENT\_ID \

    --role "Network Contributor" \

    --scope /subscriptions/$SUB\_ID/resourceGroups/$PIP\_RESOURCE\_GROUP

<https://github.com/rendszermernok02/szuretibal/blob/master/TestHtml01.sh>

Ezután töltsük le a Github-ról és igény szerint írjuk át a load balancer szolgáltatás definíciós fájlját. Az x.x.x.x helyére írjuk be a korábban kapott publikus IP címet.

<https://github.com/rendszermernok02/szuretibal/blob/master/LoadBalancer.yaml>

apiVersion: v1

kind: Service

metadata:

annotations:

service.beta.kubernetes.io/azure-load-balancer-resource-group: rg5

name: testhtml

spec:

loadBalancerIP: x.x.x.x

type: LoadBalancer

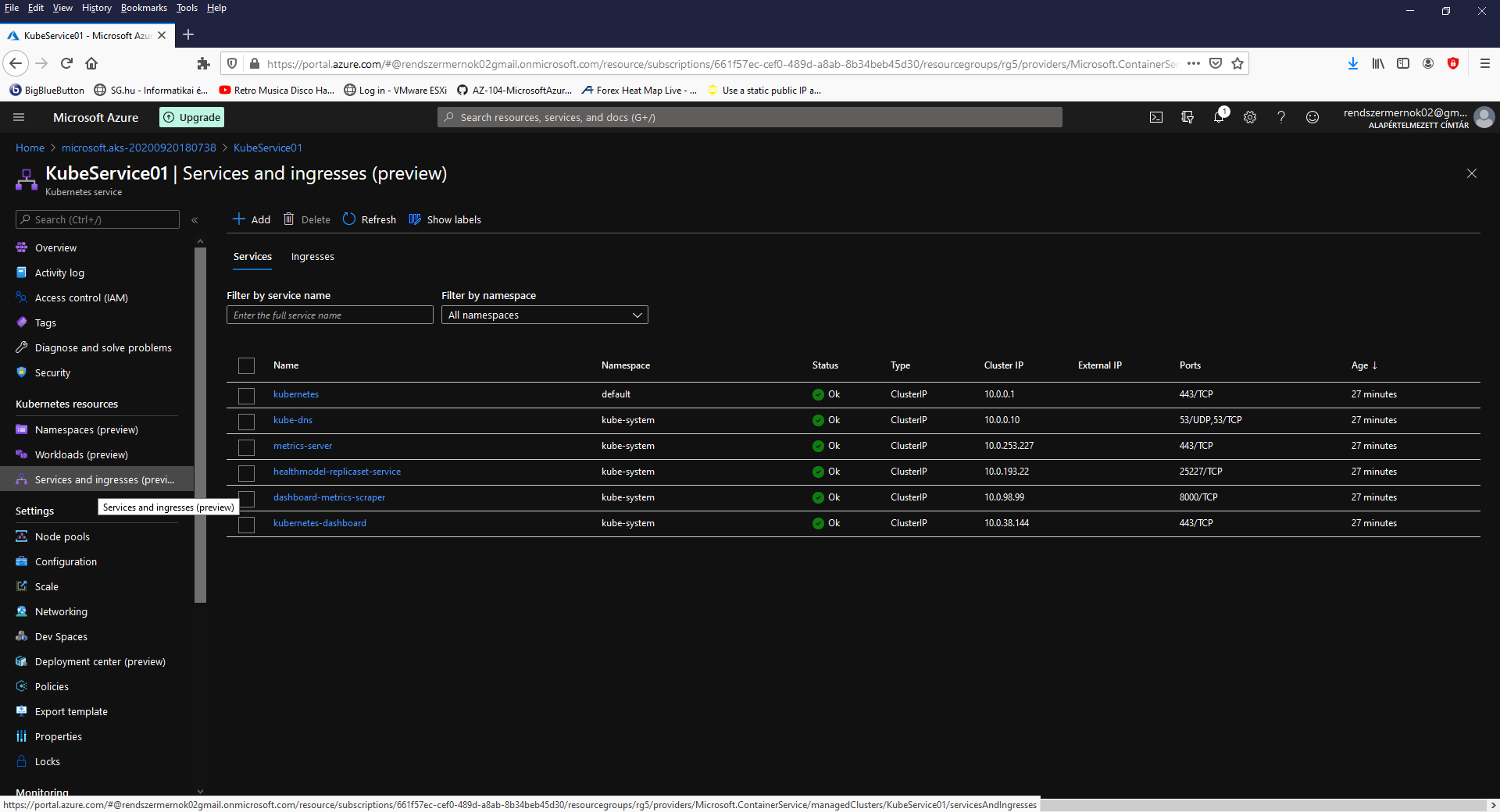
ports:

- port: 80

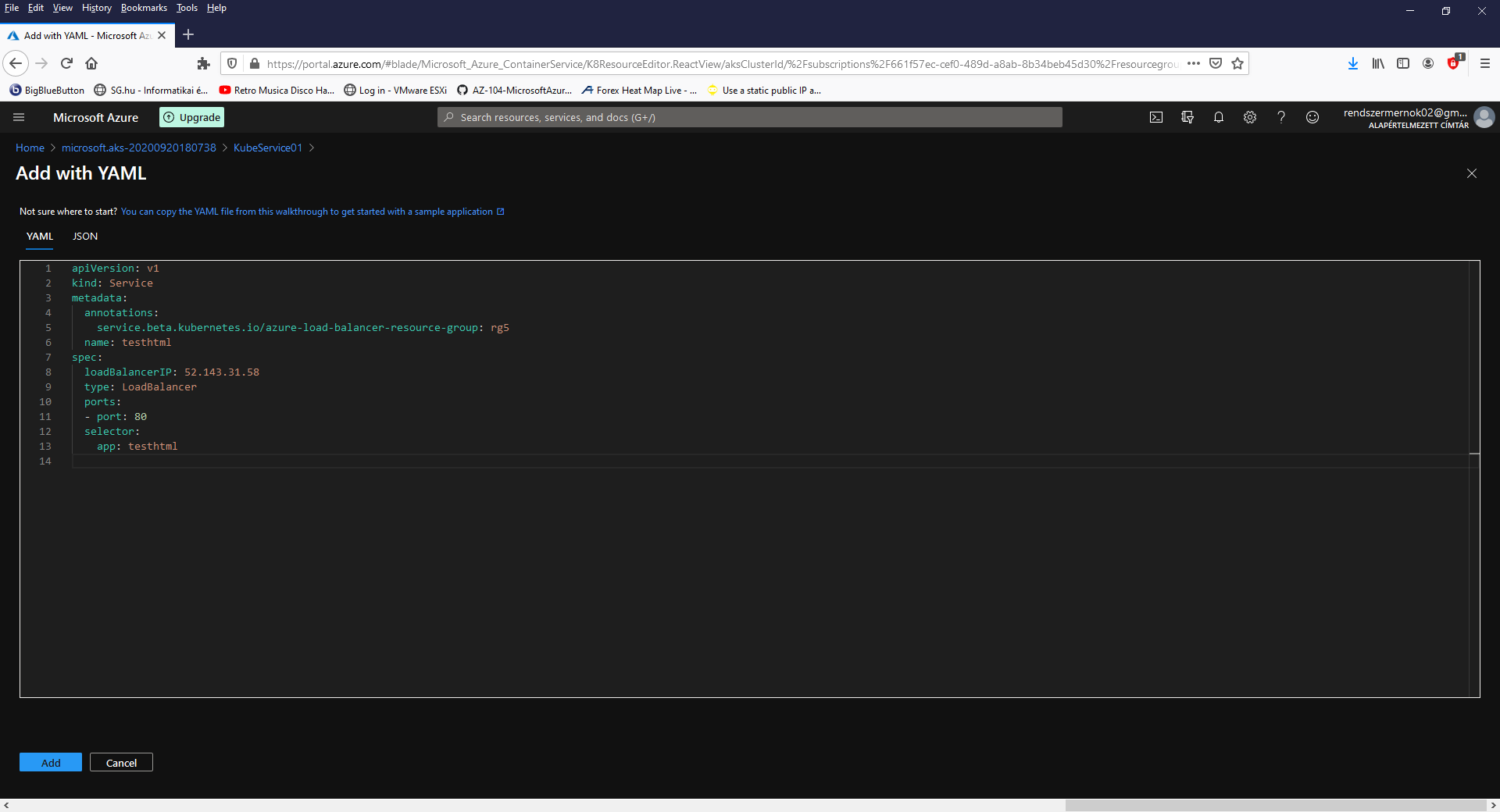
selector:

app: testhtml

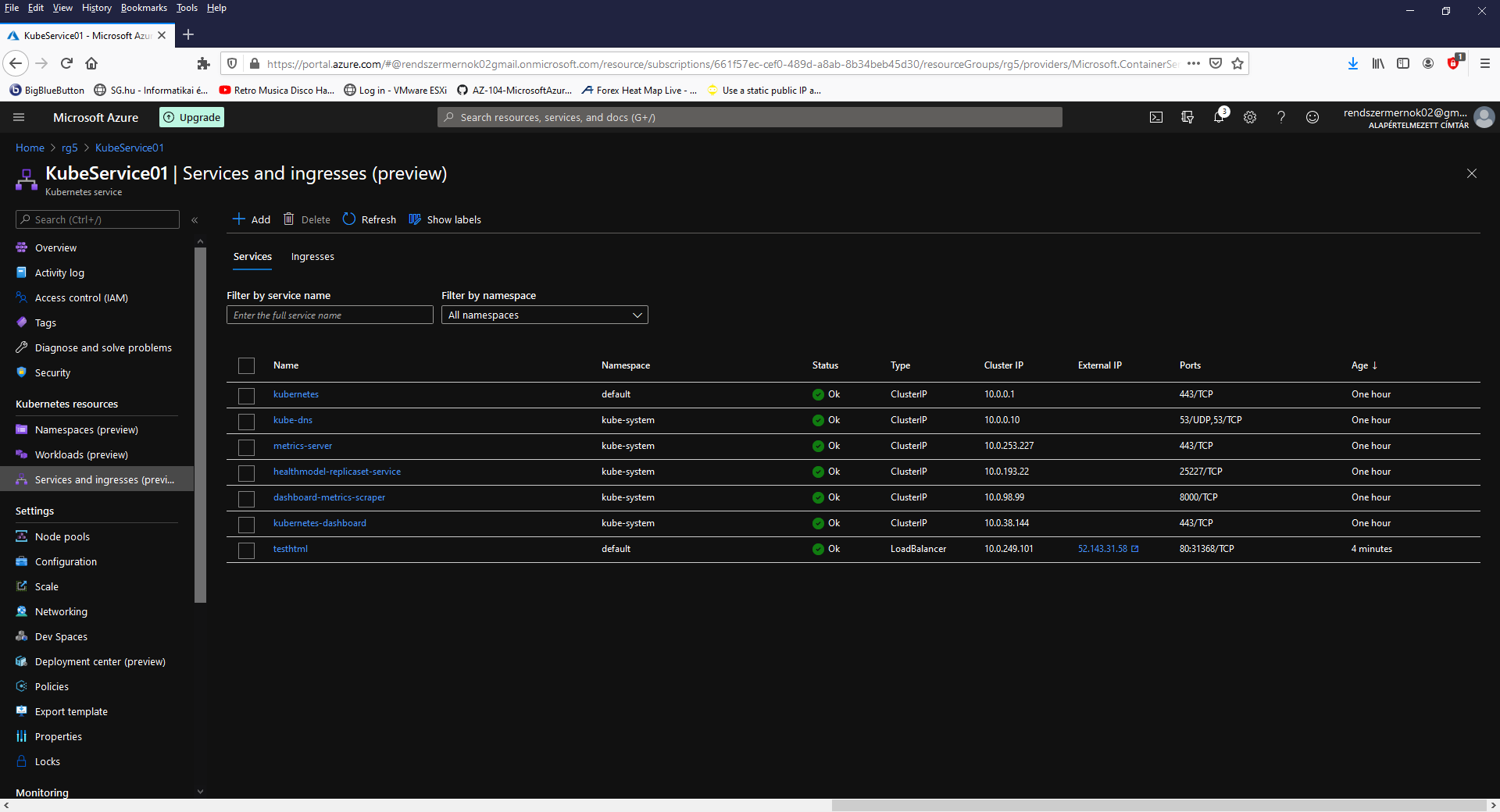
A Kubernetes service-en belül a bal oldali menüben válasszuk a „Services and ingresses” menüt, majd az „Add” gombot.



Másoljuk be a definíciós fájlt, majd nyumjuk meg az „Add”-ot.



Hamarosan látható, hogy elindult a szolgáltatás és így a weblap a megadott IP címen vagy domain néven már elérhető.



<http://szuretibal.westeurope.cloudapp.azure.com>