

---

Національний університет “Києво-Могилянська Академія”

*Диплом*

*Розробка веб-застосування на мікросервісній архітектурі та порівняння  
Microsoft Azure та AWS для його розгортання*

Виконав:

**Ахмадов Олексій Сергійович**

Науковий керівник:

**Гречко Анастасія Валеріївна**

---

**НАУКМА 2025**

# Актуальність

**Поширення хмарних технологій**

**Зростання популярності мікросервісної архітектури**

**Microsoft Azure та AWS є лідерами серед хмарних провайдерів із широким набором сервісів**

<b>Провайдер</b>	<b>Частка ринку</b>
<b>AWS</b>	<b>30%</b>
<b>Azure</b>	<b>21%</b>
<b>Google Cloud</b>	<b>12%</b>

# Мета та задачі

*Мета: Розробити адаптивну менторську платформу з мікросервісною архітектурою та реалізувати її деплой у Microsoft Azure та AWS для порівняння часу, вартості, легкості та зручності використання цих сервісів та опанування навичок роботи з цими платформами*



## *Проектування*

Визначення архітектури, вибір технологій



## *Розробка*

Написання коду, реалізація бізнес-логіки



## *Деплой Azure*

Налаштування Azure AKS, реєстру ACR, бази, secrets



## *Деплой AWS*

Налаштування AWS: EKS, ECR, IAM, EC2



## *Порівняння*

Порівняння Azure і AWS за критеріями: вартість, швидкість деплою, масштабованість, моніторинг

# Ринок менторських платформ



*ADPList — глобальний  
волонтерський маркетплейс*



*MentorCruise — платна  
підписка від \$50/міс*



*GrowthMentor — бізнес-фокус,  
\$60–120/міс*

# Ключові порівняння

Платформа	\$ / міс	Локалізація	Кастомізація
ADPList	0*	EN	—
MentorCruise	50-150	EN	—
GrowthMentor	60-120	EN	—
MentorMatch	0	UA / EN	+

# Функціональні вимоги до системи

1

*Реєстрація та авторизація через JWT, підтримка ролей: MENTEE / MENTOR / ADMIN*

2

*Профілі менторів і менті з фільтрацією за навичками, компаніями, рейтингом*

3

*Бронювання сесій: запит, підтвердження, скасування, завершення*

4

*Система відгуків і рейтингів для обох сторін*

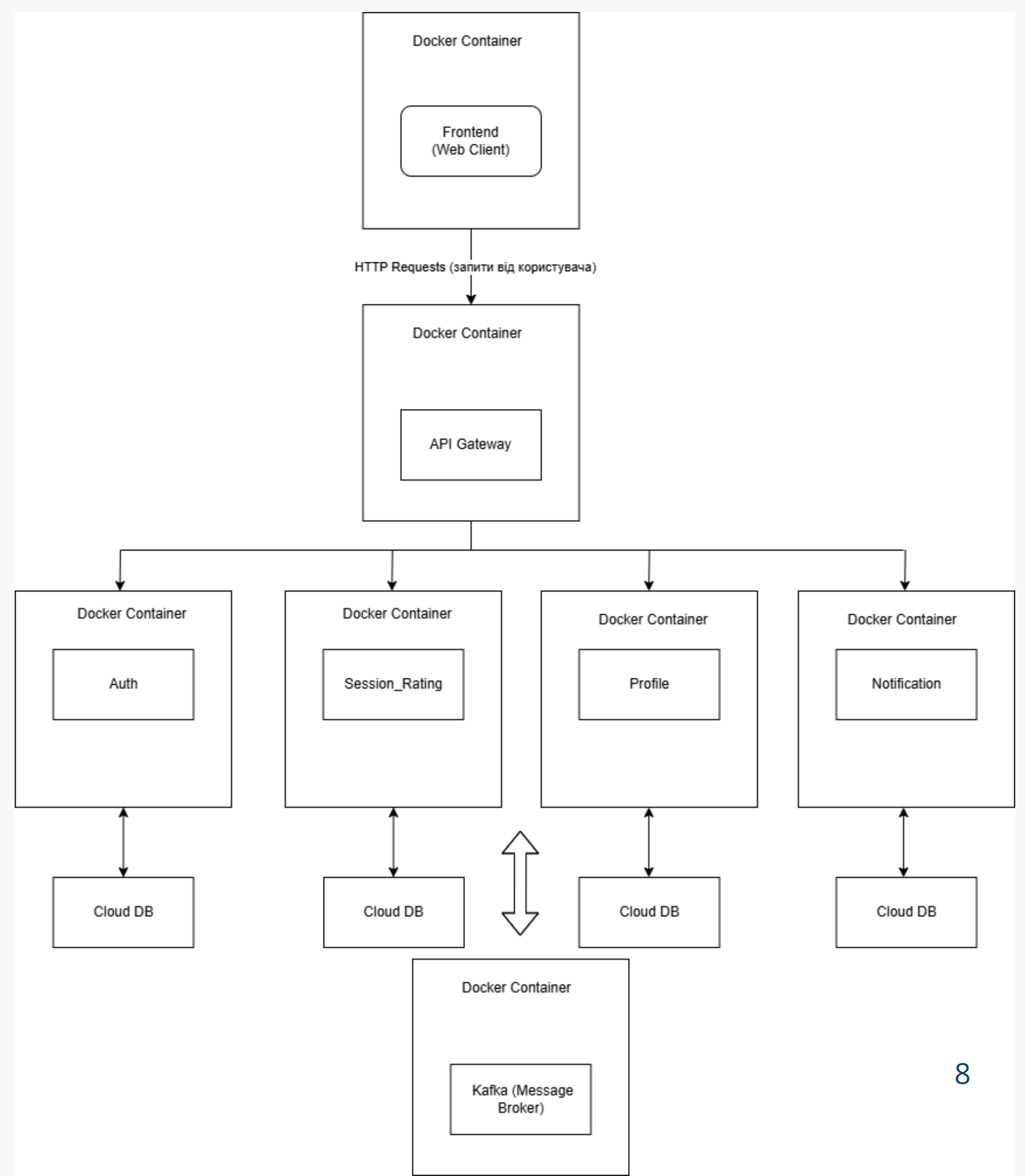
5

*Сповіщення подій у системі*

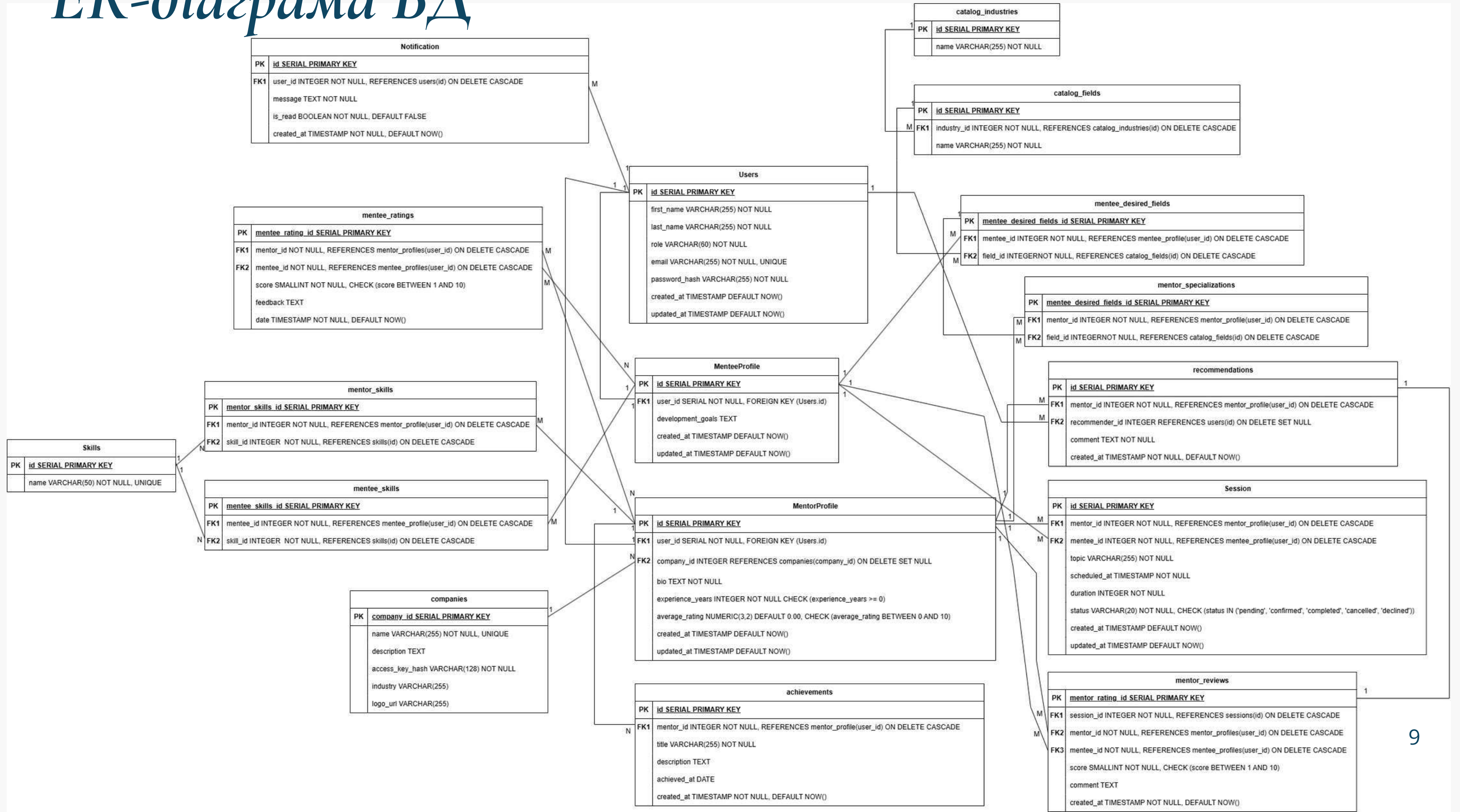
# Обрані технології розробки

<b>Категорія</b>	<b>Обрано</b>
<b>БД</b>	Azure Database for PostgreSQL, Amazon RDS for PostgreSQL
<b>Події</b>	Kafka
<b>Оркестрація</b>	Kubernetes (AKS для Azure, EKS для AWS)
<b>Моніторинг</b>	Prometheus + Grafana

# Загальна архітектура



# ER-діаграма БД



# Інтерфейс

MentorMatch

Головна

Ментори

Сесії

Зареєструватися

Логін

користувача

## MentorMatch



Стань ментором або знайди наставника як менті

Приєднатися зараз

Запросити демо

# Інтерфейс

## користувача

MentorMatch

Головна

Ментори

Сесії



A

Фільтр



**Jane Doe**

Експерт з Data Science

10 років досвіду

Забронювати



**John Smith**

Спеціаліст з штучного інтелекту

7 років досвіду

Забронювати



**John Clark**

Full-stack розробник

5 років досвіду

Забронювати



**Michael Brown**

Експерт з хмарних обчислень

8 років досвіду

Забронювати



**Alber Lee**

Спеціаліст з кібербезпеки

6 років досвіду

Забронювати



**Sara Kim**

Архітектор програмного забезпечення

12 років досвіду

Забронювати

# Створення AKS та

## Microsoft

```
az aks create \
  --resource-group mentormatch-rg \
  --name mm-cluster \
  --node-count 2 \
  --node-vm-size Standard_B2s \
  --generate-ssh-keys
```

*Azure створює кластер та  
нод-групу за замовчуванням  
автоматично. Менше  
контролю, але швидкий  
старт.*

## EKS

```
aws eks create-cluster \
  --name "mentormatch-eks" \
  --region "eu-north-1" \
  --kubernetes-version "1.32" \
  --role-arn "arn:aws:iam::476864926250:role/AmazonEKSAutoClusterRole" \
  --resources-vpc-config subnetIds=subnet-0e74b2149aeaec979, \
  subnet-08962afe0b5881d70,subnet-0b7fc881f09e2fb49,endpointPublicAccess=true

aws eks create-nodegroup \
  --cluster-name "mentormatch-eks" \
  --nodegroup-name "worker-v1" \
  --node-role "arn:aws:iam::476864926250:role/AmazonEKSNodeRole_main" \
  --subnets subnet-0e74b2149aeaec979 subnet-08962afe0b5881d70 subnet-0b7fc881f09e2fb49 \
  --scaling-config minSize=1,maxSize=2,desiredSize=1 \
  --instance-types t3.medium \
  --disk-size 20 \
  --region "eu-north-1"
```

*У AWS конфігурація кластера і нод-групи розділена.  
Більше контролю — складніше налаштування.*

## Amazon AWS

Microsoft

```

apiVersion: v1
kind: Service
metadata:
  name: api-gateway
spec:
  type: LoadBalancer
  selector:
    app: api-gateway
  ports:
    - name: http
      port: 80
      targetPort: 80

```

Azure

# Load Balancer та

# Ingress

Amazon AWS

```

1  apiVersion: networking.k8s.io/v1
2  kind: Ingress
3  metadata:
4    name: api-gateway-ingress
5  annotations:
6    alb.ingress.kubernetes.io/scheme: internet-facing
7    alb.ingress.kubernetes.io/target-type: ip
8    alb.ingress.kubernetes.io/listen-ports: '[{"HTTP": 80}]'
9    alb.ingress.kubernetes.io/backend-protocol: HTTP
10   kubernetes.io/ingress.class: alb

```

Ingress з ALB та анотаціями (створює AWS Application Load Balancer)

Kubernetes Service муну LoadBalancer (створює Azure Load Balancer)

Load balancers (1/1)

Elastic Load Balancing scales your load balancer capacity automatically in response to changes in incoming traffic.

Filter load balancers:

Name	DNS name	State	VPC ID	Availability Zones	Type
k8s-default-apigatew-2...	k8s-default-apigatew-2519...	Active	vpc-00379c9c7d8343d5b	3 Availability Zones	application

```

~ $ kubectl get ingress
NAME                CLASS    HOSTS    ADDRESS                                                                 PORTS
api-gateway-ingress <none>  *       k8s-default-apigatew-25193b2fe1-1428406214.eu-north-1.elb.amazonaws.com  80
~ $

```

ALB у AWS Console, створений через Ingress Controller

kubernetes Load balancer

Move Delete Refresh Give feedback

Essentials

Resource group (move): [mc-mentormatch-rg-mm-cluster-west-europe](#)

Location: [West Europe](#)

Subscription (move): [Azure subscription 1](#)

Subscription ID: 5023f3de-1461-4a7d-89f9-a7b1286c848f

SKU: Standard

Tags (edit): aks-managed-cluster-name: mm-cluster aks-managed-cluster-rg: mentormatch-rg

See more

api-gateway	default	Ok	LoadBalancer	10.0.246.107	9.163.141.7	80:30205/TCP
-------------	---------	----	--------------	--------------	-------------	--------------

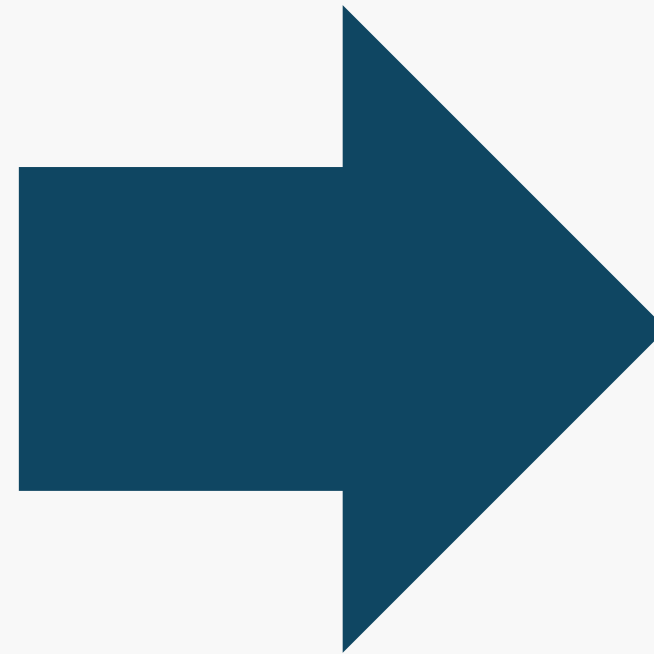
Load Balancer в Azure Portal та автоматичне створення зовнішнього порта

# Secret

*.ENV*

```
1 # Django settings
2 SECRET_KEY=your_secret_key_here
3
4 # PostgreSQL AWS connecting
5 DB_NAME=your_db_name
6 DB_USER=your_db_user
7 DB_PASSWORD=your_db_password
8 DB_HOST=your_db_host_url
9 DB_PORT=5432
10
11 ADMIN_EMAIL=admin@example.com
12 ADMIN_FIRST_NAME=Admin
13 ADMIN_LAST_NAME=User
14 ADMIN_PASSWORD=your_admin_password
15
```

*S*

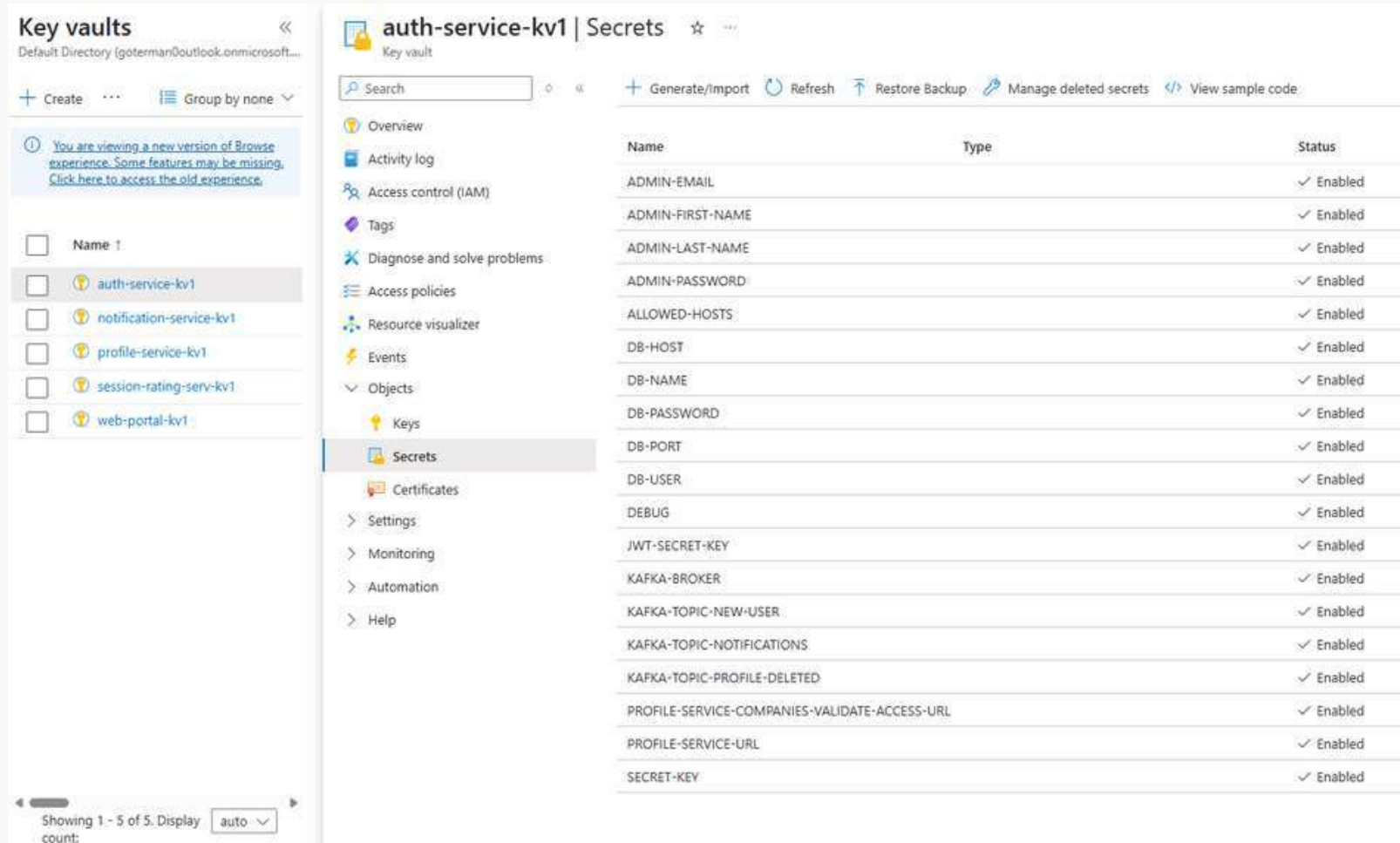


*service-*

```
apiVersion: v1
kind: Secret
metadata:
  name: auth-service-env
type: Opaque
stringData:
  # Django settings
  SECRET_KEY: "your_secret_key_here"
  # PostgreSQL AWS connecting
  DB_NAME: "your_db_name"
  DB_USER: "your_db_user"
  DB_PASSWORD: "your_db_password"
  DB_HOST: "your_db_host_url"
  DB_PORT: "5432"
  # Admin user
  ADMIN_EMAIL: "admin@example.com"
  ADMIN_FIRST_NAME: "Admin"
  ADMIN_LAST_NAME: "User"
  ADMIN_PASSWORD: "your_admin_password"
```

# Зберігання секретів у хмарі та інтеграція з Kubernetes через CSI

## Microsoft Azure - Key Vault



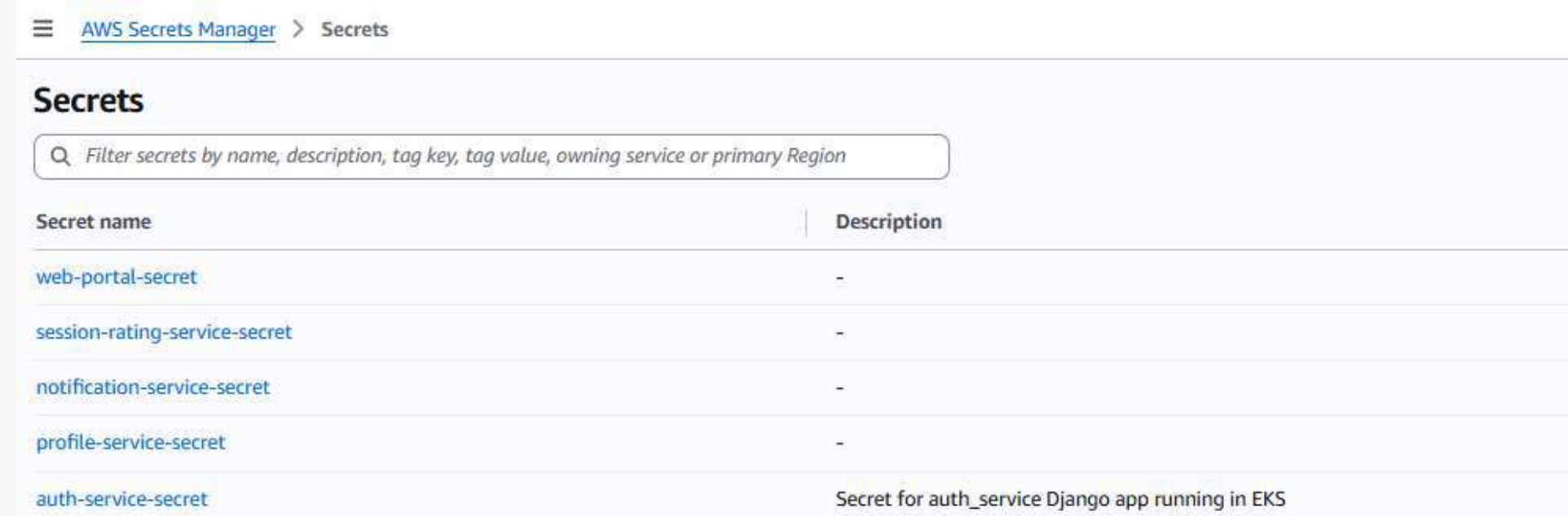
Key Vault зі списком секретів для кожного мікросервісу

```
1. Встановлення CSI драйвера:  
helm upgrade --install csi-secrets-store secrets-store-csi-driver/secrets-store-csi-driver \\  
  --namespace kube-system  
2. Встановлення Azure provider:  
kubectl apply -f https://raw.githubusercontent.com/Azure/secrets-store-csi-driver-provider-azure/master/deployment/secretproviderclasspodstatus-crd.yaml  
kubectl apply -f https://raw.githubusercontent.com/Azure/secrets-store-csi-driver-provider-azure/master/deployment/provider-azure-installer.yaml  
3. Створення SP (Service Principal) з доступом до Key Vault:  
az ad sp create-for-rbac \  
  --name "k8s-keyvault-access" \  
  --role "Key Vault Secrets User" \  
  --scopes /subscriptions/.../vaults/auth-service-kv1
```

Налаштування інтеграції з Azure Key Vault

## Driver

## Amazon AWS - Secrets Manager



Секрети для кожного мікросервісу зберігаються в AWS Secrets

```
helm repo add secrets-store-csi-driver https://kubernetes-sigs.github.io/secrets-store-csi-driver/charts  
helm repo update  
  
aws eks update-kubeconfig --name mentormatch-eks --region eu-north-1  
  
helm upgrade --install csi-secrets-store secrets-store-csi-driver/secrets-store-csi-driver \  
  --namespace kube-system  
kubectl get pods -n kube-system | grep csi-secrets-store  
  
kubectl apply -f \  
https://raw.githubusercontent.com/aws/secrets-store-csi-driver-provider-aws/main/deployment/aws-provider-install  
kubectl apply -f csi-driver-secret-rbac.yaml
```

Команди для встановлення AWS Secrets Store CSI

Driver

# Хмарні бази даних для мікросервісів

## Microsoft

**Azure Database for PostgreSQL flexible servers** ...  
Default Directory

+ Create Manage view Refresh Export to CSV Open query Assign tags

You are viewing a new version of Browse experience. Some features may be missing. Click here to access the old experience.

Filter for any field... Subscription equals all Resource Group equals all Location equals all Add filter

Name ↑	Status	Version
authdbserver	Ready	15.12
notificationdbserver	Ready	15.12
profiledbserver	Ready	15.12
sessionrating	Ready	15.12

## Azure Database for PostgreSQL Flexible Server

```
# PostgreSQL Azure connecting
DB_NAME=postgres
DB_USER=[REDACTED]
DB_PASSWORD=[REDACTED]
DB_HOST=authdbserver.postgres.database.azure.com
DB_PORT=5432
```

## Підключення до Azure PostgreSQL

## Amazon AWS

Aurora and RDS > Databases

Databases (4)

Filter by databases

DB identifier	Status	Role	Engine	Region ...	Size
auth-db	Available	Instance	PostgreSQL	eu-north-1c	db.t4g.micro
notification-db	Available	Instance	PostgreSQL	eu-north-1b	db.t4g.micro
profile-db	Available	Instance	PostgreSQL	eu-north-1c	db.t4g.micro
sessionrating-db	Available	Instance	PostgreSQL	eu-north-1c	db.t4g.micro

## Amazon RDS for PostgreSQL

Secret key

SECRET_KEY	[REDACTED]
JWT_SECRET_KEY	[REDACTED]
DEBUG	True
[REDACTED]	[REDACTED]
[REDACTED]	[REDACTED]
DB_USER	[REDACTED]
DB_PASSWORD	[REDACTED]
DB_HOST	auth-db.cbq62m4e81ns.eu-north-1.rds.amazonaws.com
DB_PORT	5432

## Підключення до AWS RDS

```
az login

echo "==== [2/7] Create resource group, ACR, AKS ====="
az group create --name mentormatch-rg --location westeurope

az acr create --resource-group mentormatch-rg --name mmregistry --sku Basic

az aks create \
  --resource-group mentormatch-rg \
  --name mm-cluster \
  --node-count 2 \
  --node-vm-size Standard_B2s \
  --generate-ssh-keys

az aks update --name mm-cluster --resource-group mentormatch-rg --attach-acr mmregistry

echo "==== [3/7] Get AKS credentials ====="
az aks get-credentials --resource-group mentormatch-rg --name mm-cluster --overwrite-existing

# === CSI Driver install (Kube System Namespace, as required) ===
echo "==== [4/7] Install Azure Key Vault CSI Driver ====="
helm repo add secrets-store-csi-driver https://kubernetes-sigs.github.io/secrets-store-csi-driver/charts
helm repo update

helm upgrade --install csi-secrets-store secrets-store-csi-driver/secrets-store-csi-driver \
  --namespace kube-system

kubectl apply -f https://raw.githubusercontent.com/Azure/secrets-store-csi-driver-provider-azure/master/deployment/secret-provider.yaml
kubectl apply -f https://raw.githubusercontent.com/Azure/secrets-store-csi-driver-provider-azure/master/deployment/provider.yaml

# === OIDC/Federation: This is important for production (no static secrets!) ===
echo "==== [5/7] Setup AKS OIDC and federated identity ====="
OIDC_ENABLED=$(az aks show --resource-group mentormatch-rg --name mm-cluster --query "oidcIssuerProfile.enabled" -o tsv)
if [[ "$OIDC_ENABLED" != "true" ]]; then
  az aks update --resource-group mentormatch-rg --name mm-cluster --enable-oidc-issuer
fi

OIDC_URL=$(az aks show --resource-group mentormatch-rg --name mm-cluster --query "oidcIssuerProfile.issuerUrl" -o tsv)
echo "OIDC Issuer URL: $OIDC_URL"
```

Фрагмент Bash-скрипт для деплою в Microsoft Azure

```
eksctl create iamserviceaccount \
  --region eu-north-1 \
  --cluster mentormatch-eks \
  --namespace kube-system \
  --name aws-load-balancer-controller \
  --attach-policy-arn arn:aws:iam::476864926250:policy/AWSLoadBalancerControllerIAMPolicy \
  --approve \
  --override-existing-serviceaccounts

# 8. Install AWS Load Balancer Controller with Helm
echo "[Step 8] Install AWS Load Balancer Controller"
VPC_ID=$(aws eks describe-cluster \
  --name mentormatch-eks \
  --region eu-north-1 \
  --query "cluster.resourcesVpcConfig.vpcId" \
  --output text)

helm uninstall aws-load-balancer-controller -n kube-system || true

helm install aws-load-balancer-controller eks/aws-load-balancer-controller \
  -n kube-system \
  --set clusterName=mentormatch-eks \
  --set serviceAccount.create=false \
  --set serviceAccount.name=aws-load-balancer-controller \
  --set region=eu-north-1 \
  --set vpcId=$VPC_ID

kubectl get pods -n kube-system -l app.kubernetes.io/name=aws-load-balancer-controller

# 9. Install Zookeeper and Kafka with persistent storage
echo "[Step 9] Install Zookeeper and Kafka with persistent storage"
helm install zookeeper bitnami/zookeeper \
  --set replicaCount=1 \
  --set persistence.enabled=true \
  --set persistence.storageClass=gp2 \
  --set persistence.size=2Gi

helm install kafka bitnami/kafka \
  --set replicaCount=1 \
  --set zookeeper.enabled=false \
  --set externalZookeeper.servers=zookeeper.default.svc.cluster.local:2181 \
```

Фрагмент Bash-скрипт для деплою в AWS

# Azure vs

Характеристики	<i>Azure</i> <b>AWS</b>	AWS
Безкоштовний пакет послуг	200\$	\$100 - навчальний заклад є учасником AWS Educate \$30-\$75 - не є учасником AWS Educate
Панель управління	Azure Portal — легко зрозумілий для початківця	AWS Console гнучкіша, але складна. Для всього потрібні налаштування політик та ролей
Підтримка Kubernetes	AKS має простішу інтеграцію з ACR та автоматичне оновлення kubeconfig	EKS дає більше контролю, але потребує ручного налаштування (OIDC, IAM, LoadBalancer)
Балансування навантаження	Під капотом	Повне ручне налаштування

# Висновки

- Проаналізовано існуючі системи-аналоги та сформульовано вимоги до створюваної системи
- Розроблено повноцінне веб-застосування із мікросервісною архітектурою, з можливостями локалізації та кастомізації. Проект має потенціал до масштабування як внутрішній інструмент для компаній, освітніх закладів та спільнот.
- Використано сучасні інструменти, які застосовуються у продакшн-середовищах: Kafka, Docker, Kubernetes, Prometheus, Grafana, хмарні бази даних та системи зберігання секретів.
- Порівняно розгортання системи на хмарних платформах Azure та AWS



# *Питання*

