

Задание 1: настройка Azure Cloud Shell
Задание 2: Скачивание стартовых файлов
Задание 3: Создание ресурсной группы
Задание 4: Создание SSH ключа
Задание 5: Разворачивание ARM шаблона
Задание 6: Создание GitHub репозитория
Задание 7: Подключение к build агенту
Задание 8: Настройка build агента
Задание 9: Клонирование репозитория на Build агент
Задание 10: Запуск стартового приложения
Задание 11: Сборка Docker контейнеров
Задание 12: Запуск docker контейнеров
Задание 13: Конфигурация переменных окружения
Задание 14: Отправка образов в Azure Container Registry
Задание 15: Настройка CI Pipeline для отправки образов

- Задание 1. Миграция MongoDB в Cosmos DB с помощью службы миграции базы данных Azure.
Задача 1. Включение поставщика ресурсов Microsoft.DataMigration
Задача 2: предоставление службы миграции базы данных Azure
Задача 3. Перенести данные в Azure Cosmos DB.
- Задание 2. Разверните решение в службе Azure Kubernetes.
Задача 1. Туннель в кластер службы Azure Kubernetes.
Задача 2. Разверните службу с помощью портала Azure.
Задача 3. Разверните сервис с помощью kubectl
Задача 4. Обзор Azure Monitor для контейнеров.
- Задание 3: масштабируйте приложение и тестируйте высокую доступность
Задача 1. Увеличение количества экземпляров службы с портала Azure.
Задача 2: устранение сбоев при инициализации реплик.
Задача 3: перезапустить контейнеры и протестировать НА
Задача 4. Настройка автомасштабирования Cosmos DB.
Задача 5. Тестирование автомасштабирования Cosmos DB.
- Задание 4: Работа со службами и маршрутизация трафика приложений.
Задача 1. Обновите внешнюю службу для поддержки динамического обнаружения с помощью балансировщика нагрузки.
Задача 2. Отрегулируйте ограничения ЦП, чтобы улучшить масштаб
Задача 3: выполнить скользящее обновление
Задача 4: настроить Kubernetes Ingress

Аннотация

Эта практическая лабораторная работа предназначена для того, чтобы провести вас через процесс создания и развертывания образов Docker на платформе Kubernetes, размещенной на Azure Kubernetes Services (AKS), а также научиться работать с динамическим обнаружением сервисов, масштабированием сервисов и высокой доступностью.

По завершении этой лабораторной работы вы сможете создавать и развертывать контейнерные приложения в службе Azure Kubernetes и выполнять стандартные процедуры DevOps.

Обзор

Fabrikam Medical Conferences (FabMedical) предоставляет услуги веб-сайта конференций, адаптированные для медицинского сообщества. Они реорганизуют код своего приложения на основе node.js, чтобы его можно было запускать как приложение Docker, и хотят реализовать POC, который поможет им ознакомиться с процессом разработки, жизненным циклом развертывания и критическими аспектами хостинга. среда. Они будут развертывать свои приложения в службе Azure Kubernetes и хотят узнать, как развертывать контейнеры с динамической балансировкой нагрузки, обнаруживать контейнеры и масштабировать их по запросу.

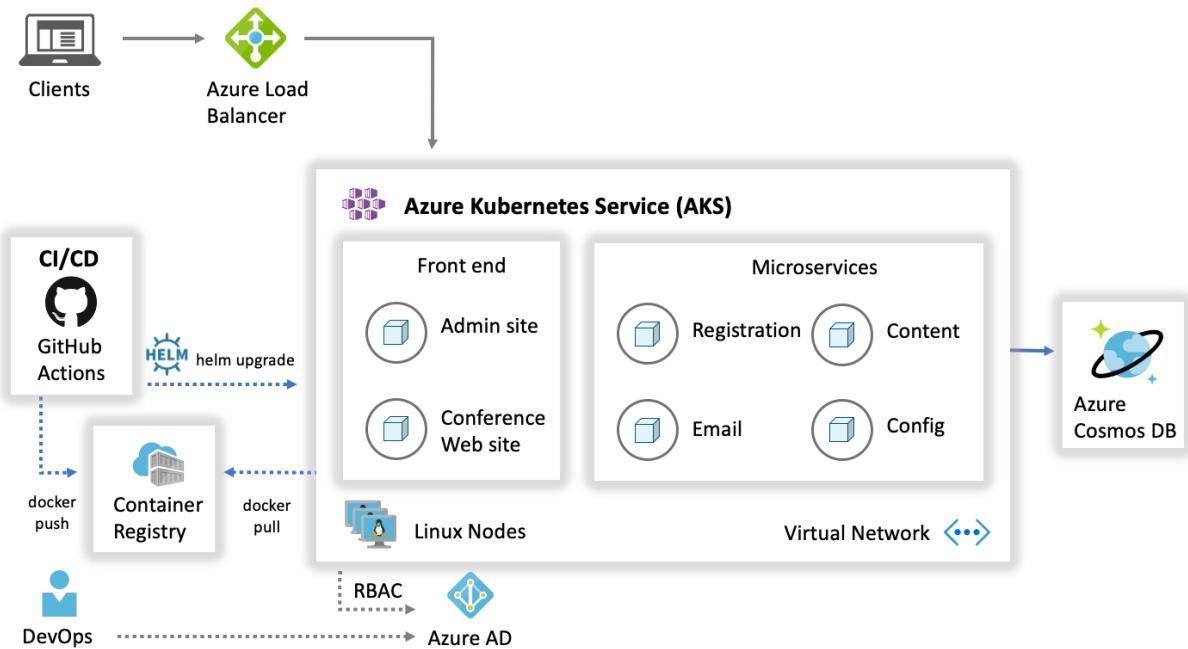
В этой практической лабораторной работе вы поможете заполнить этот POC с подмножеством кодовой базы приложения. Вы создадите агент сборки на основе Linux и кластер службы Azure Kubernetes для запуска развернутых приложений. Вы поможете им завершить настройку Docker для своего приложения, протестировать локально, отправить в репозиторий образов, развернуть в кластере и протестировать балансировку нагрузки и масштабирование.

Важно: для большинства ресурсов Azure требуются уникальные имена. На протяжении этих шагов вы будете видеть слово «СУФФИКС» как часть имен ресурсов. Вы должны заменить это уникальным дескриптором (например, префиксом электронной почты вашей учетной записи Microsoft), чтобы обеспечить уникальные имена для ресурсов..

Архитектура решения

Ниже представлена схема архитектуры решения, которую вы создадите. Пожалуйста, изучите это внимательно, чтобы понять решение в целом, работая над различными компонентами.

Решение будет использовать службу Azure Kubernetes (AKS), что означает, что топология кластера контейнеров подготавливается в соответствии с количеством запрошенных узлов. Предлагаемые контейнеры, развернутые в кластере, показаны ниже с Cosmos DB в качестве управляемой службы:



Each tenant will have the following containers:

- **Conference Web site:** Приложение SPA, которое будет использовать параметры конфигурации для обработки пользовательских стилей для клиента.
- **Admin Web site:** Приложение SPA, которое владельцы конференций используют для управления деталями конфигурации конференции, управления регистрацией участников, управления кампаниями и общения с участниками.
- **Registration service:** API, который обрабатывает все действия по регистрации, создавая новые регистрации на конференции с соответствующим выбором пакетов и соответствующей стоимостью.
- **Email service:** API, который обрабатывает уведомления по электронной почте для участников конференции во время регистрации или когда владельцы конференции решают привлечь участников через свой сайт администратора.
- **Config service:** API-интерфейс, который обрабатывает такие параметры конфигурации конференции, как даты, места, таблицы цен, специальные предложения, обратный отсчет и т. д.
- **Content service:** API, который обрабатывает контент для конференции, такой как докладчики, сессии, семинары и спонсоры.

Задание 1: Миграция MongoDB в Cosmos DB с помощью службы миграции базы данных Azure

На этом этапе у вас есть веб-приложения и приложения API, запущенные в экземпляре Docker (VM - Build Agent). Следующим шагом является перенос данных базы данных MongoDB в Azure Cosmos DB. В этом упражнении будет использоваться служба миграции базы данных Azure для переноса данных из базы данных MongoDB в Azure Cosmos DB.

Задача 1. Включение поставщика ресурсов Microsoft.DataMigration

В этой задаче вы включите использование службы миграции базы данных Azure в своей подписке Azure, зарегистрировав поставщика ресурсов Microsoft.DataMigration.

1. Откройте Azure Cloud Shell.
2. Выполните следующую команду Azure CLI, чтобы зарегистрировать поставщика ресурсов Microsoft.DataMigration в своей подписке Azure:

```
az provider register --namespace Microsoft.DataMigration
```

Задача 2. Подготовка службы миграции базы данных Azure

В этой задаче вы развернете экземпляр службы миграции базы данных Azure, который будет использоваться для миграции данных из MongoDB в Cosmos DB.

1. На портале Azure выберите + Создать ресурс.
2. Найдите на торговой площадке Службу миграции базы данных Azure и выберите ее.
3. Выберите "Создать".

The screenshot shows the Microsoft Azure portal interface. At the top, there is a navigation bar with a search bar labeled 'Search resources, services, and docs (G+/-)'. Below the navigation bar, the breadcrumb trail shows 'Home > New > Marketplace >'. The main title is 'Azure Database Migration Service' by Microsoft. There is a 'Create' button. Below the title, there is a brief description: 'Azure Database Migration Service (DMS) is designed to streamline the process of migrating on-premises databases to the cloud.' At the bottom of the page, there are three tabs: 'Overview' (which is underlined), 'Plans', and 'Usage Information + Support'.

На вкладке «Основные» панели «Создание службы миграции» введите следующие значения:

- Группа ресурсов: выберите группу ресурсов, созданную с помощью этой лабораторной работы.
- Название службы миграции: введите имя, например fabmedical [SUFFIX].
- Расположение: выберите регион Azure, используемый для группы ресурсов.

Create Migration Service

Basics Networking Tags Review + create

Azure Database Migration Service is designed to streamline the process of migrating on-premises databases to Azure.
[Learn more.](#)

Project details

Select the subscription to manage deployed resources and constants. Use resource groups as you would folders, to organize and manage all of your resources.

Subscription * ⓘ

Windows Azure - MVP Sponsorship

Resource group * ⓘ

fabmedical-cp

[Create new](#)

Instance details

Migration service name * ⓘ

fabmedicalcp

Location * ⓘ

East US

Service mode * ⓘ

Azure Hybrid (Preview)

Pricing tier *

Standard

1 vCores

[Configure tier](#)

4. Выберите Далее: Сеть >>.
5. На вкладке «Сеть» выберите виртуальную сеть в группе ресурсов fabmedical-[SUFFIX]..

Create Migration Service

Basics **Networking** Tags Review + create

Select an existing virtual network or create a new one.

i Select from a list of existing virtual networks. Click on the links to see more details about the selected virtual network.
[Learn more.](#)

Search to filter items...				
↑↓	Name	↑↓	Resource group	↑↓
			Gateways	↑↓
<input checked="" type="checkbox"/>	fabmedical-cp-vnet/default	fabmedical-cp	Network without gateway	
<input type="checkbox"/>	aks-vnet-40307078/aks-s...	MC_fabmedical-cp_fabm...	Network without gateway	

i Create a new virtual network by entering a name below. This will create a basic VNET that can connect to source servers with public facing IPs. You can then take additional steps to upgrade this network and increase your connectivity options.
[Learn more.](#)

Virtual network name

Enter new network name

6. Выберите Обзор + создать.
7. Выберите «Создать», чтобы создать экземпляр службы миграции базы данных Azure.

Предоставление услуги может занять 5-10 минут.

Задача 3. Перенести данные в Azure Cosmos DB.

В этой задаче вы создадите проект миграции в службе миграции базы данных Azure, а затем перенесете данные из MongoDB в Azure Cosmos DB.

1. На портале Azure перейдите к виртуальной машине агента сборки и скопируйте частный IP-адрес (2). Вставьте содержимое в текстовый редактор по вашему выбору (например, Блокнот в Windows, пользователи macOS могут использоватьTextEdit) для использования в будущем.

The screenshot shows the Azure portal interface for a virtual machine named 'fabmedical-dy'. The left sidebar has a 'Overview' tab selected, indicated by a red box and the number '1'. The main content area displays 'Essentials' information and 'Properties' tabs. In the 'Properties' section, under 'Virtual machine', the 'Networking' tab is selected, showing network details. A red box labeled '2' highlights the 'Private IP address' field, which is listed as '172.16.0.4'.

2. На портале Azure перейдите к ранее подготовленной службе миграции базы данных Azure.
3. В колонке «Служба миграции базы данных Azure» выберите + Новый проект миграции на панели «Обзор».
4. На панели «Новый проект миграции» введите следующие значения, затем выберите «Создать и запустить действие»:
 - Название проекта: fabmedical
 - Тип исходного сервера: MongoDB
 - Тип целевого сервера: CosmosDB (MongoDB API)
 - Выберите тип деятельности: offline data migration.

New migration project

Project name

 ✓

Source server type *

 ▼

Target server type *

 ▼

*Choose type of activity >

Offline data migration



To successfully migrate from MongoDB, please:

Create a Cosmos DB account with support of MongoDB API

Примечание. Выбран тип действия «Автономная миграция данных», поскольку вы будете выполнять однократную миграцию с MongoDB на Cosmos DB. Кроме того, данные в базе данных не будут обновляться во время миграции. В производственном сценарии вам нужно будет выбрать тип деятельности проекта миграции, который наилучшим образом соответствует требованиям вашего решения.

2. На панели мастера автономной миграции MongoDB в базу данных Azure для CosmosDB введите следующие значения для вкладки Выбор источника.:
 - Mode: **Standard mode**
 - Source server name: Enter the Private IP Address of the Build Agent VM used in this lab.
 - Server port: 27017
 - Require SSL: Unchecked

Примечание. Оставьте поля «Имя пользователя» и «Пароль» пустыми, поскольку для экземпляра MongoDB на виртуальной машине с агентом сборки для этой лабораторной работы не включена проверка подлинности. Служба миграции базы данных Azure подключена к той же виртуальной сети, что и виртуальная машина агента сборки, поэтому она может обмениваться данными в виртуальной сети напрямую с виртуальной машиной, не открывая службу MongoDB в Интернете. В производственных сценариях вы всегда должны включать аутентификацию на MongoDB.

MongoDB to Azure Database for CosmosDB Offline Migration Wizard

Select source Select target Database setting Collection setting Migration summary

Mode Standard mode

Source server name * 172.16.0.4

Server port 27017

User Name Enter domain\user name

Password Enter password

Connection properties
 Require SSL

i DMS requires **TLS 1.2 security protocol** enabled to establish an encrypted connection to the source MongoDB instance.
Follow these steps to enable TLS support: [TLS 1.2 support for MongoDB](#)

Or, enable TLS 1.0/1.1 from service configuration.

3. Выберите Далее: Выбрать цель >.
4. На панели «Выбрать цель» выберите следующие значения.:
 - Mode: **Select Cosmos DB target**
 - Subscription: Select the Azure subscription you're using for this lab.

- Выберите имя Cosmos DB: выберите экземпляр fabmedical-[SUFFIX] Cosmos DB..

MongoDB to Azure Database for CosmosDB Offline Migration Wizard

Select source **Select target** Database setting Collection setting Migration summary

Mode	Select Cosmos DB target
Subscription	
Select Cosmos DB name:	fabmedical-[SUFFIX]
Connection string *	<pre>mongodb://fabmedical- cp:HjJTu1ZK8CjX2g2j3vZ2QHBBuWBVlvGEDKz11AZIrzBPc1owQ2D1QzSvu dCfvd4ChD1TXmBiAS5HCu7Q61IVFA==@undefined:10255?/ ssl=true&replicaSet=globaldb</pre>

Обратите внимание, что строка подключения автоматически заполнится ключом для вашего экземпляра Azure Cosmos DB.

5. Измените строку подключения, заменив @undefined: на @ fabmedical-[SUFFIX].documents.azure.com: так, чтобы имя DNS соответствовало экземпляру Azure Cosmos DB. Обязательно замените [СУФФИКС].

Connection string *

```
mongodb://fabmedical-
cp:HjJTu1ZK8CjX2g2j3vZ2QHBBuWBVlvGEDKz11AZIrzBPc1owQ2D1QzSvu
dCfvd4ChD1TXmBiAS5HCu7Q61IVFA==@fabmedical-
cp.documents.azure.com:10255/?ssl=true&replicaSet=globaldb
```

6. Выберите **Next: Database setting >>**.
7. На вкладке «Параметры базы данных» выберите исходную базу данных contentdb, чтобы эта база данных из MongoDB была перенесена в Azure Cosmos DB..

MongoDB to Azure Database for CosmosDB Offline Migration Wizard

Select source Select target **Database setting** Collection setting Migration summary

Search to filter items... All

1 item(s) ← prev Page 1 of 1 next →

Source Database	Target Database	Throughput (RU/s)	Clean up collections
<input checked="" type="checkbox"/> contentdb	Create: contentdb	Blank for default or RU...	<input checked="" type="checkbox"/> Clean up collections

8. Выберите Далее: Настройка коллекции >>.
9. На вкладке «Параметры коллекции» разверните базу данных contentdb и убедитесь, что для миграции выбраны коллекции sessions и speakers. Кроме того, обновите пропускную способность (RU / s) до 400 для обеих коллекций.

MongoDB to Azure Database for CosmosDB Offline Migration Wizard

Select source Select target Database setting **Collection setting** Migration summary

Mongo collection settings for each database

^ contentdb 2 of 2

Search to filter items...
2 item(s) ← prev Page 1 of 1 next →

Name	Target collection	Throughput (RU/s)	Shard key	Unique
<input checked="" type="checkbox"/> sessions	Create: sessions	RU: 400	_id	<input type="checkbox"/>
<input checked="" type="checkbox"/> speakers	Create: speakers	RU: 400	_id	<input type="checkbox"/>

10. Выберите Далее: Сводка миграции >>.
11. На вкладке «Сводка миграции» введите MigrateData в поле «Имя действия», затем выберите «Начать миграцию», чтобы начать миграцию данных MongoDB в Azure Cosmos DB..

MongoDB to Azure Database for CosmosDB Offline Migration Wizard

Select source Select target Database setting Collection setting **Migration summary**

Activity name MigrateData ✓

Target server name
fabmedical-cp.documents.azure.com

Target server version
3.2.0

Source server name
172.16.0.4

Source server version
4.0.21

Database(s) to migrate
1 of 1
 Boost RUs during migration

12. Будет показан статус миграционной активности. Перенос займет всего несколько секунд. Выберите «Обновить», чтобы перезагрузить статус и убедиться, что он показывает статус «Завершено».

MigrateData ×

fabmedical2

Delete migration Stop migration Refresh Retry

Essentials

Status	: Complete	Duration	: 00:00:10
Throughput (bytes/s)	: Complete: 15.23 KB of 15.23 KB	Throughput (documents/s)	: Complete: 8 of 8
Total bytes	: 15.23 KB	Total documents	: 8

Search to filter items...

Name	↑↓ Status	↑↓ Errors	↑↓ Documents processed	↑↓ Bytes copied	↑↓ Duration	↑↓
contentdb	Complete		8/8	15.23 KB/15.23 KB	00:00:10	...

13. Чтобы убедиться, что данные были перенесены, перейдите к учетной записи Cosmos DB на портале Azure, затем выберите проводник данных. Вы увидите коллекции докладчиков и сессий, перечисленные в базе данных contentdb, и сможете изучить документы внутри.

The screenshot shows the Azure Data Explorer interface. On the left, there's a sidebar with various options like Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Quick start, Notifications, and Data Explorer (which is selected). Below that is a 'Settings' section with items such as Connection String, Features, Mongo Preview Features, Replicate data globally, Default consistency, Backup & Restore, Firewall and virtual networks, Private Endpoint Connections, Data Migration, and Preview Features. The main area shows 'COLLECTIONS' with 'contentdb' expanded to show 'sessions'. Under 'sessions', 'Documents' is selected, and a list of documents is shown with IDs like 54b... and 54b3e... The right side has a 'Documents' panel with a text input for queries and a code editor displaying a JSON document. The JSON document contains fields like '_id', 'speakerNames', 'speakers', 'trackNames', 'tracks', '_v', 'abstract', 'date', 'endTime', 'eventName', 'hidden', 'roomID', 'roomName', 'sessionId', 'sessioncode', 'startTime', 'timeSlot', and 'title'.

Задание 2. Развёртывание решения в службе Azure Kubernetes.

В этом задании вы подключитесь к кластеру службы Azure Kubernetes, который вы создали перед практической лабораторной работой, и развернете приложение Docker в кластере с помощью Kubernetes.

Задача 1: туннель в кластер службы Azure Kubernetes

В этой задаче вы соберете необходимую информацию о кластере службы Azure Kubernetes для подключения к кластеру и выполните команды для подключения к панели управления Kubernetes из облачной оболочки.

Примечание. Следующие задачи должны выполняться в облачной оболочке, а не на машине сборки, поэтому отключитесь от машины сборки, если она все еще подключена. Verify that you are connected to the correct subscription with the following command to show your default subscription:

```
az account show
```

- Если вы не подключены к правильной подписке, укажите свои подписки, а затем установите подписку по ее идентификатору с помощью следующих команд (аналогично тому, что вы делали в облачной оболочке перед лабораторной работой):

```
az account list
az account set --subscription {id}
```

- Настройте kubectl для подключения к кластеру Kubernetes:

```
az aks get-credentials -a --name fabmedical-SUFFIX --resource-group fabmedical-SUFFIX
```

- Проверьте правильность конфигурации, выполнив простую команду kubectl для создания списка узлов.:

```
kubectl get nodes
```

@Azure:~\$ kubectl get nodes				
NAME	STATUS	ROLES	AGE	VERSION
aks-agentpool-40307078-vmss00000	Ready	agent	11d	v1.17.11
aks-agentpool-40307078-vmss00001	Ready	agent	11d	v1.17.11

Задача 2. Развёртывание службы с помощью портала Azure.

В этой задаче вы развернете приложение API в кластере службы Azure Kubernetes с помощью портала Azure.

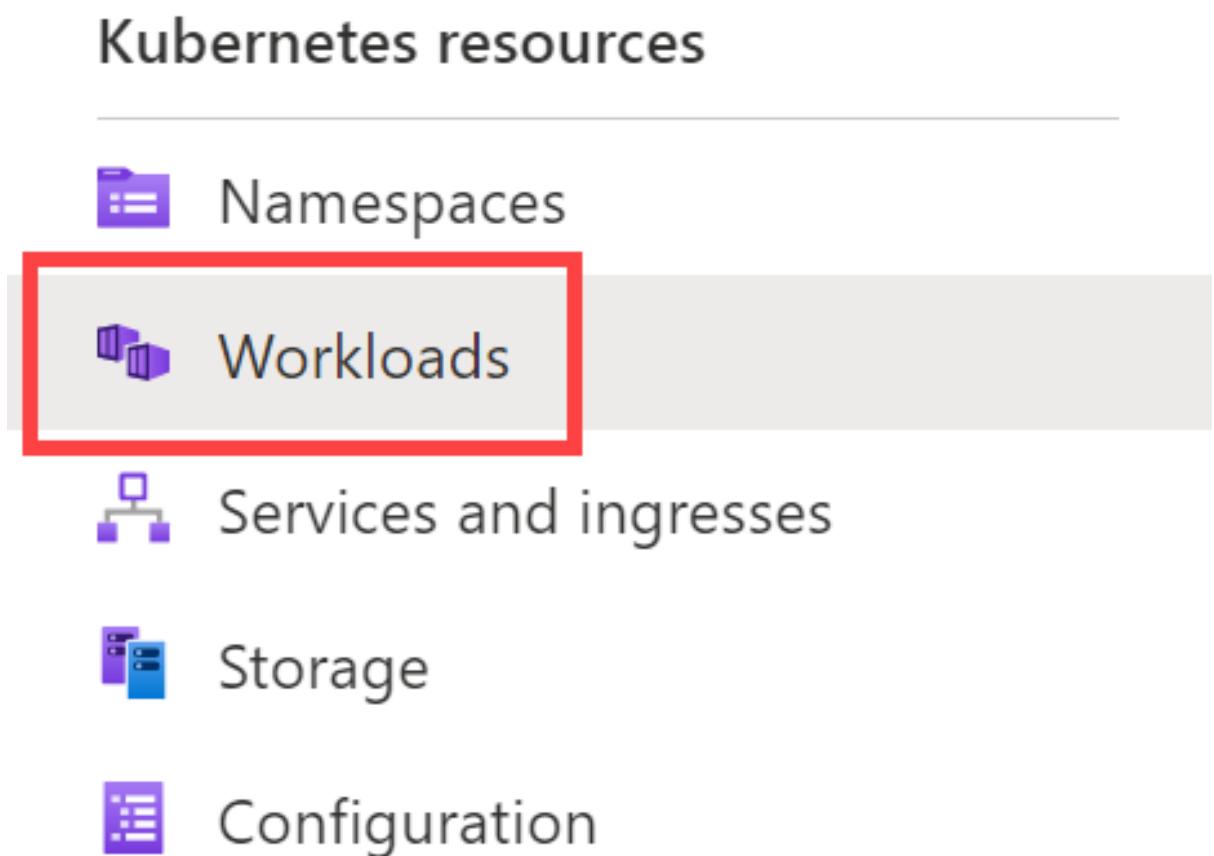
- Сначала нам нужно определить Сервис для нашего API, чтобы приложение было доступно в кластере. В колонке AKS на портале Azure выберите Services and ingresses, а на вкладке Службы выберите + Добавить.

The screenshot shows the Azure portal interface for managing Kubernetes services. At the top, it displays the service name "fabmedical-sjw | Services and ingresses" and indicates it's a "Kubernetes service". Below this, there's a navigation bar with a search bar, an "Add" button (which is highlighted with a red box), and other standard UI elements like "Delete", "Refresh", and "Show labels". On the left, there's a sidebar with links for "Activity log", "Access control (IAM)", "Tags", "Diagnose and solve problems", and "Security". Under "Kubernetes resources", the "Services and ingresses" link is also highlighted with a red box. The main content area has tabs for "Services" and "Ingresses", with "Services" currently selected. It includes a "Filter by service name" input field and a "Name" column with a list of existing services: "kubernetes", "healthmodel-replicaset-service", "kube-dns", and "metrics-server".

- На экране «Добавить с YAML» вставьте следующий YAML и выберите «Добавить».

```
3. apiVersion: v1
4. kind: Service
5. metadata:
6.   labels:
7.     app: api
8.   name: api
9. spec:
10.   ports:
11.     - name: api-traffic
12.       port: 3001
13.       protocol: TCP
14.       targetPort: 3001
15.   selector:
16.     app: api
17.   sessionAffinity: None
    type: ClusterIP
```

18. Теперь выберите Workloads в разделе ресурсов Kubernetes на левой панели навигации.



19. В представлении «Рабочие нагрузки», выбрав «Развертывания» (по умолчанию), нажмите «+ Добавить».

fabmedical-sjw | Workloads

Kubernetes service

Search (Ctrl+/) <<

+ Add Delete Refresh

-  Overview
-  Activity log
-  Access control (IAM)
-  Tags
-  Diagnose and solve problems
-  Security

Kubernetes resources

-  Namespaces
-  Workloads
-  Services and ingresses
-  Storage

Deployments Pods Replica sets

Filter by deployment name

Enter the full deployment name

<input type="checkbox"/>	Name
<input type="checkbox"/>	coredns-autoscaler
<input type="checkbox"/>	coredns
<input type="checkbox"/>	metrics-server
<input type="checkbox"/>	omsagent-rs
<input type="checkbox"/>	tunnefront

20. На загружающемся экране «Добавить с YAML» вставьте следующий YAML и обновите заполнитель [LOGINSERVER], указав имя экземпляра ACR.

```
21. apiVersion: apps/v1
22. kind: Deployment
23. metadata:
24.   labels:
25.     app: api
26.   name: api
27. spec:
28.   replicas: 1
29.   selector:
30.     matchLabels:
31.       app: api
32.   strategy:
33.     rollingUpdate:
34.       maxSurge: 1
35.       maxUnavailable: 1
36.     type: RollingUpdate
37.   template:
38.     metadata:
39.       labels:
40.         app: api
41.         name: api
42.     spec:
```

```

43.      containers:
44.        - name: api
45.          image: [LOGINSERVER].azurecr.io/content-api
46.          imagePullPolicy: Always
47.          livenessProbe:
48.            httpGet:
49.              path: /
50.              port: 3001
51.            initialDelaySeconds: 30
52.            periodSeconds: 20
53.            timeoutSeconds: 10
54.            failureThreshold: 3
55.          ports:
56.            - containerPort: 3001
57.              hostPort: 3001
58.              protocol: TCP
59.          resources:
60.            requests:
61.              cpu: 1
62.              memory: 128Mi
63.          securityContext:
64.            privileged: false
65.            terminationMessagePath: /dev/termination-log
66.            terminationMessagePolicy: File
67.            dnsPolicy: ClusterFirst
68.            restartPolicy: Always
69.            schedulerName: default-scheduler
70.            securityContext: {}
    terminationGracePeriodSeconds: 30

```

71. Выберите Добавить, чтобы начать развертывание. Это может занять несколько минут, после чего вы увидите список развертывания.

	Name	Namespace	Ready
<input type="checkbox"/>	coredns-autoscaler	kube-system	✓ 1/1
<input type="checkbox"/>	coredns	kube-system	✓ 2/2
<input type="checkbox"/>	metrics-server	kube-system	✓ 1/1
<input type="checkbox"/>	omsagent-rs	kube-system	✓ 1/1
<input type="checkbox"/>	tunneelfront	kube-system	✓ 1/1
<input checked="" type="checkbox"/>	api	default	! 0/1

72. Выберите развертывание api, чтобы открыть Deployment, выберите Live logs, а затем Pod из раскрывающегося списка. Через несколько секунд должны появиться живые журналы.

Примечание: если журналы не отображаются, возможно, модуля больше не существует. Вы можете использовать представление в Log Analytics для просмотра журналов истории независимо от модуля.

73. Если вы прокрутите журнал, вы увидите, что это указывает на то, что приложение content-api снова дает сбой, потому что не может найти MongoDB api для связи. Вы решите эту проблему, подключившись к Cosmos DB.

74. На портале Azure перейдите к своей группе ресурсов и найдите свою Cosmos DB. Выберите ресурс Cosmos DB, чтобы просмотреть подробности.

Filter by name...			All types	All locations	N
9 items <input checked="" type="checkbox"/> Show hidden types <small>?</small>					
NAME	TYPE	LOCATION			
fabmedical-sol	Virtual machine	East US			
fabmedical-sol	Kubernetes service	East US			
fabmedical-sol_0SDISK_1_50214e494c7040500de55c1b43a9***	Disk	East US			
fabmedical-sol2	Azure Cosmos DB account	East US			
fabmedicalsol3	Container registry	East US			
fabmedical-sol637	Network interface	East US			
fabmedical-sol-ip	Public IP address	East US			
fabmedical-sol-nsg	Network security group	East US			
fabmedical-sol-vnet	Virtual network	East US			

75. В разделе «Быстрый запуск» выберите вкладку Node.js и скопируйте строку подключения к Node.js 3.0.

Congratulations! Your Azure Cosmos DB account with MongoDB API is ready.

Now, let's connect your existing MongoDB app to it:

Choose a platform

- .NET
- Node.js**
- MongoDB Shell
- Java
- Python
- Others

1 Using the Node.js 2.2 driver, connect your existing MongoDB app

You can use your existing MongoDB Node.js 2.2 driver to work with Azure Cosmos DB. Make sure to enable SSL. Here is an example

```
var mongoClient = require("mongodb").MongoClient;
mongoClient.connect("mongodb://fabmedical-sol2:2nb7uXNPKeg5NT73Vo400sjVThUTxElpggqkr5J7guvVJB5B1DsLBCdtir1LUJVzthJq...");
```

PRIMARY CONNECTION STRING

```
mongodb://fabmedical-sol2:2nb7uXNPKeg5NT73Vo400sjVThUTxElpggqkr5J7guvVJB5B1DsLBCdtir1LUJVzthJq...
```

For more details on configuring Node.js driver to use SSL, follow [this article](#).

Questions? [Contact us](#)

Using the Node.js 3.0 driver, connect your existing MongoDB app

You can use your existing MongoDB Node.js 3.0 driver to work with Azure Cosmos DB. Make sure to enable SSL. Here is an example

```
var mongoClient = require("mongodb").MongoClient;
mongoClient.connect("mongodb://fabmedical-sol2:2nb7uXNPKeg5NT73Vo400sjVThUTxElpggqkr5J7guvVJB5B1DsLBCdtir1LUJVzthJq...");
```

PRIMARY CONNECTION STRING

```
mongodb://fabmedical-sol2:2nb7uXNPKeg5NT73Vo400sjVThUTxElpggqkr5J7guvVJB5B1DsLBCdtir1LUJVzthJq...
```

For more details on configuring Node.js driver to use SSL, follow [this article](#).

Questions? [Contact us](#)

76. Измените скопированную строку подключения, добавив базу данных contentdb к URL-адресу вместе с replicaSet globaldb. Результатирующая строка подключения должна выглядеть как в примере ниже. Обратите внимание, что вам может потребоваться изменить URL-адрес конечной точки.

Примечание. Имя пользователя и пароль отредактированы для краткости.

```
mongodb://<USERNAME>:<PASSWORD>@fabmedical-<SUFFIX>.documents.azure.com:10255/contentdb?ssl=true&replicaSet=globaldb
```

Примечание. В этой лабораторной работе использовалась версия API Cosmos DB 3.2. Однако, если ваша скопированная строка подключения имеет суффикс конечной точки .mongo.cosmos.azure.com, не стесняйтесь использовать его, поскольку он будет ссылаться на API 3.6. Лаборатория совместима с обеими версиями. Вот как строка подключения, показанная выше, будет отображаться с API версии 3.6:

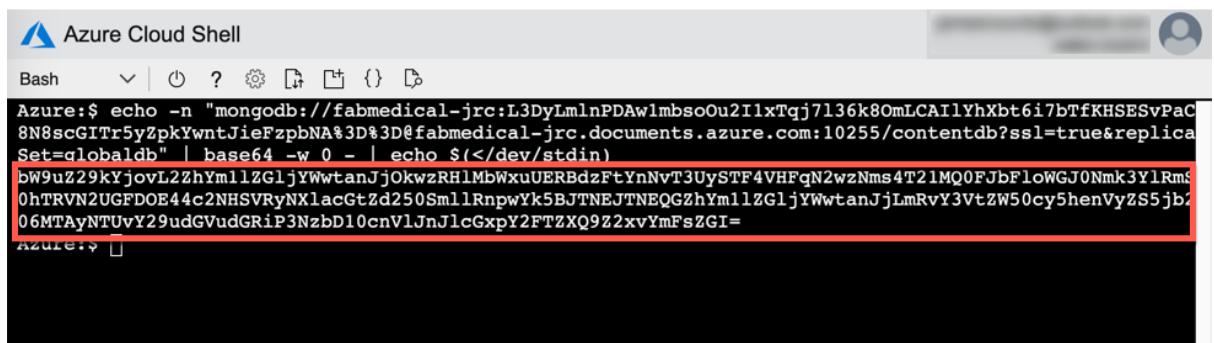
```
mongodb://<USERNAME>:<PASSWORD>@fabmedical-<SUFFIX>.mongo.cosmos.azure.com:10255/contentdb?ssl=true&replicaSet=globaldb
```

77. Вы настроите секрет Kubernetes для хранения строки подключения и настроите приложение content-api для доступа к секрету. Во-первых, вы должны base64

закодировать секретное значение. Откройте окно Azure Cloud Shell и используйте следующую команду для кодирования строки подключения, а затем скопируйте выходные данные.

Примечание: двойные кавычки, окружающие строку подключения, необходимы для успешного создания требуемого вывода.

```
echo -n "[CONNECTION STRING VALUE]" | base64 -w 0 - | echo  
$(</dev/stdin)
```



Azure Cloud Shell interface showing a terminal window. The command `echo -n "..." | base64 -w 0 - | echo \$(</dev/stdin)"` is entered and executed, displaying a long base64 encoded string. The output is highlighted with a red rectangle.

78. Вернитесь в колонку AKS на портале Azure и выберите «Конфигурация» в разделе ресурсов Kubernetes. Выберите Секреты и нажмите + Добавить.
79. На экране «Добавить с YAML» вставьте следующий YAML и замените заполнитель на закодированную строку подключения из буфера обмена и выберите «Добавить». Обратите внимание, что YAML чувствителен к расположению, поэтому вы должны убедиться, что отступы правильные при вводе или вставке.
80. apiVersion: v1
81. kind: Secret
82. metadata:
83. name: cosmosdb
84. type: Opaque
85. data:
db: <base64 encoded value>

Add with YAML

Not sure where to start? You can copy the YAML:

[YAML](#) [JSON](#)

```
1  apiVersion: v1
2  kind: Secret
3  metadata:
4    name: cosmosdb
5  type: Opaque
6  data:
7    db: bW9uZ29kYjovL2ZhYm11
c3NsPXRydWUmcmVwbG1jYVNldD1n
```

86. Отсортируйте список секретов по имени, и теперь вы должны увидеть свой новый секрет.

The screenshot shows the Kubernetes Secrets list interface. At the top, there are tabs for 'Config maps' and 'Secrets', with 'Secrets' being the active tab. Below the tabs are three filter inputs: 'Filter by secret name' containing 'cosmosdb', 'Filter by label selector' containing 'foo=bar,key!=value', and 'Filter by namespace' set to 'All namespaces'. The main table lists one secret: 'cosmosdb'. The table columns are 'Name', 'Namespace', 'Type', 'Data', and 'Age'. The 'cosmosdb' row has a checkbox in the first column, a link to its details in the second column, and values 'default', 'Opaque', '1', and 'One minute' respectively in the other columns.

Name	Namespace	Type	Data	Age
cosmosdb	default	Opaque	1	One minute

87. Просмотрите сведения о секрете `cosmosdb`, выбрав его в списке.

The screenshot shows the Azure portal interface for a secret named 'cosmosdb'. The top navigation bar includes a search bar ('Search (Ctrl+)/') and a refresh button. On the left, there are three tabs: 'Overview' (selected), 'YAML', and 'Events'. To the right, the secret details are shown: Namespace 'default', Labels (empty), and a 'See more' link. Below this, the 'Data' tab is selected, showing the value 'db' and its corresponding hex representation 'bW9uZ29kYjovL2ZhYm1lZGljYWw'.

88. Затем загрузите конфигурацию api, используя следующую команду в окне Azure Cloud Shell.:

```
kubectl get -o=yaml deployment api > api.deployment.yml
```

89. Отредактируйте загруженный файл с помощью редактора кода облачной оболочки:

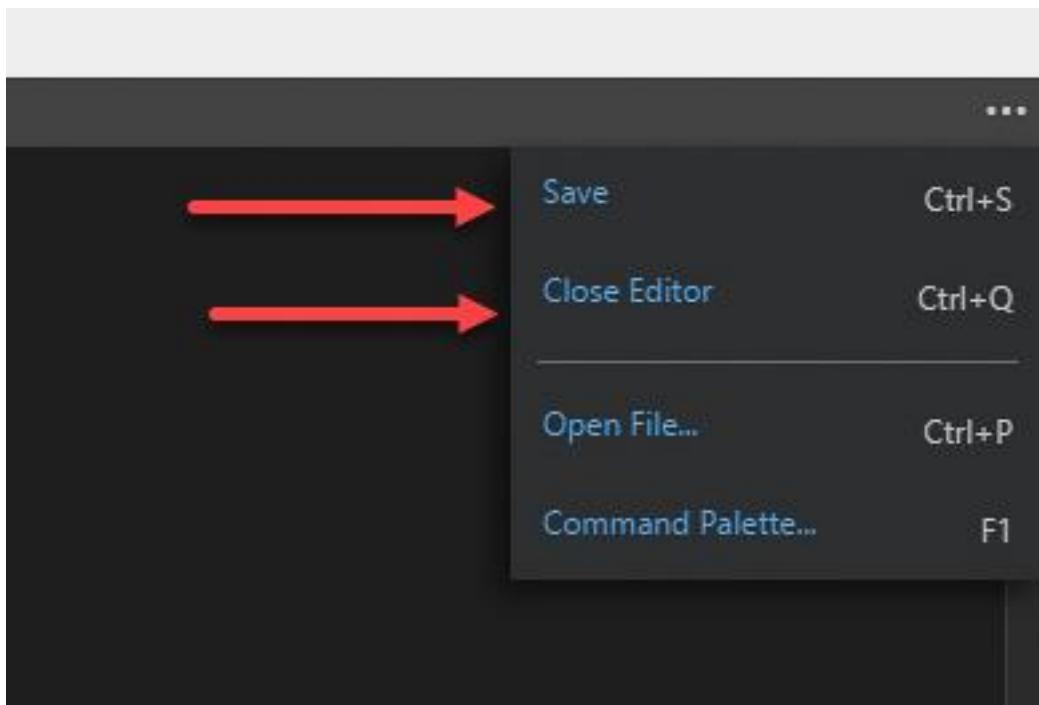
```
code api.deployment.yml
```

Добавьте следующую конфигурацию среды в спецификацию контейнера под свойством изображения:

```
env:
- name: MONGODB_CONNECTION
  valueFrom:
    secretKeyRef:
      name: cosmosdb
      key: db
```

```
20      |   rollingUpdate:
21      |   |   maxSurge: 25%
22      |   |   maxUnavailable: 25%
23      |   |   type: RollingUpdate
24      |   template:
25      |   |   metadata:
26      |   |   |   creationTimestamp: null
27      |   |   |   labels:
28      |   |   |   |   k8s-app: api
29      |   |   |   |   name: api
30      |   |   spec:
31      |   |   |   containers:
32      |   |   |   |   - image: fabmedicalco.azurecr.io/content-api
33      |   |   |   |   env:
34      |   |   |   |   |   - name: MONGODB_CONNECTION
35      |   |   |   |   |   |   valueFrom:
36      |   |   |   |   |   |   |   secretKeyRef:
37      |   |   |   |   |   |   |   |   name: cosmosdb
38      |   |   |   |   |   |   |   |   key: db
39      |   |   |   |   imagePullPolicy: Always
40      |   |   |   |   name: api
41      |   |   resources:
42      |   |   |   requests:
43      |   |   |   |   cpu: 125m
44      |   |   |   |   memory: 128Mi
45      |   |   securityContext:
46      |   |   |   privileged: false
47      |   |   |   terminationMessagePath: /dev/termination-log
48      |   |   |   terminationMessagePolicy: File
49      |   |   |   dnsPolicy: ClusterFirst
```

90. Сохраните изменения и закройте редактор.



91. Обновите развертывание API, используя kubectl для развертывания API.
92.

```
kubectl delete deployment api
kubectl create -f api.deployment.yml
```
93. На портале Azure вернитесь к живым журналам (см. Шаг 5). Последний журнал должен отображаться как подключенный к MongoDB.

2/16/2021, 4:22:03 PM api-84c6489554-wjwgn dcba0da02a2cd0919df8779bae6912f75af7c... Connected to MongoDB

Задача 3. Разверните сервис с помощью kubectl

В этой задаче разверните веб-службу с помощью kubectl.

1. Откройте новую консоль Azure Cloud Shell.
2. Создайте текстовый файл с именем web.deployment.yml с помощью редактора Azure Cloud Shell..

```
code web.deployment.yml
```

3. Скопируйте и вставьте следующий текст в редактор:

```
apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    app: web
  name: web
spec:
```

```

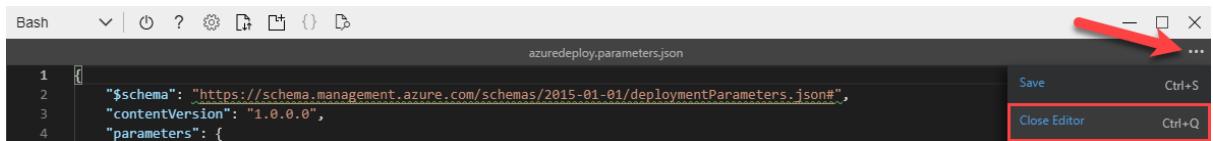
replicas: 1
selector:
  matchLabels:
    app: web
strategy:
  rollingUpdate:
    maxSurge: 1
    maxUnavailable: 1
    type: RollingUpdate
template:
  metadata:
    labels:
      app: web
      name: web
  spec:
    containers:
      - image: [LOGINSERVER].azurecr.io/content-web
        env:
          - name: CONTENT_API_URL
            value: http://api:3001
        livenessProbe:
          httpGet:
            path: /
            port: 3000
          initialDelaySeconds: 30
          periodSeconds: 20
          timeoutSeconds: 10
          failureThreshold: 3
        imagePullPolicy: Always
        name: web
        ports:
          - containerPort: 3000
            hostPort: 80
            protocol: TCP
        resources:
          requests:
            cpu: 1000m
            memory: 128Mi
        securityContext:
          privileged: false
      terminationMessagePath: /dev/termination-log
      terminationMessagePolicy: File
      dnsPolicy: ClusterFirst
      restartPolicy: Always
      schedulerName: default-scheduler
      securityContext: {}
      terminationGracePeriodSeconds: 30

```

4. Обновите запись [LOGINSERVER], чтобы она соответствовала имени вашего сервера входа в ACR.
5. Нажмите кнопку ... и выберите Сохранить..



6. Снова нажмите кнопку... и выберите «Закрыть редактор»..



7. Создайте текстовый файл с именем web.service.yml с помощью редактора Azure Cloud Shell.

```
code web.service.yml
```

8. Скопируйте и вставьте следующий текст в редактор:

```
apiVersion: v1
kind: Service
metadata:
  labels:
    app: web
    name: web
spec:
  ports:
    - name: web-traffic
      port: 80
      protocol: TCP
      targetPort: 3000
  selector:
    app: web
  sessionAffinity: None
  type: LoadBalancer
```

9. Сохраните изменения и закройте редактор.
10. Введите следующую команду, чтобы развернуть приложение, описанное в файлах YAML. Вы получите сообщение о том, что kubectl создал веб-развертывание и веб-службу.

```
kubectl create --save-config=true -f web.deployment.yml -f
web.service.yml
```



11. Вернитесь к колонке AKS на портале Azure. В меню навигации в разделе «Ресурсы Kubernetes» выберите представление «Службы и входящие данные». У вас должна быть возможность получить доступ к веб-сайту через внешнюю конечную точку.

Name	Namespace	Status	Type	Cluster IP	External IP
kubernetes	default	Ok	ClusterIP	10.0.0.1	
healthmodel-replicaset-ser...	kube-system	Ok	ClusterIP	10.0.222.237	
kube-dns	kube-system	Ok	ClusterIP	10.0.0.10	
metrics-server	kube-system	Ok	ClusterIP	10.0.48.50	
web	default	Ok	LoadBalancer	10.0.38.118	13.77.137.14

12. В верхней части навигации выберите ссылки на докладчиков и сеансы.

Задача 4. Обзор Azure Monitor для контейнеров

В этой задаче вы получите доступ и просмотрите различные журналы и панели мониторинга, доступные в Azure Monitor для контейнеров.

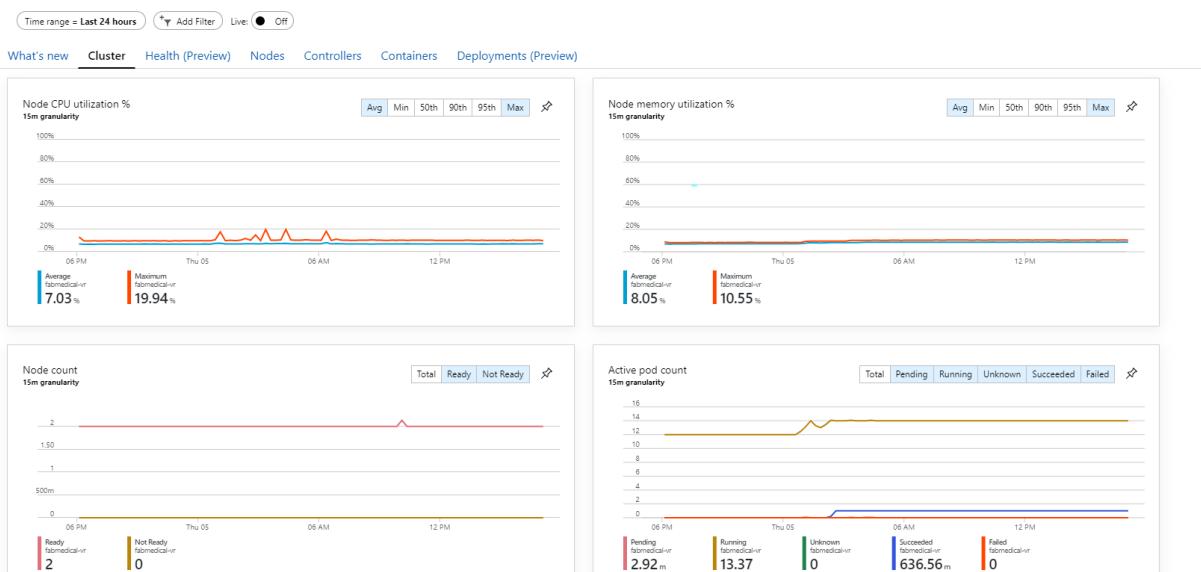
- На портале Azure выберите созданную вами группу ресурсов с именем fabmedical-SUFFIX, а затем выберите ресурс Azure службы Kubernetes.

NAME	TYPE	LOCATION
fabmedical-nsg	Network security group	East US
fabmedical	Container registry	East US
fabmedical-	Virtual machine	East US
fabmedical-	Kubernetes service	East US

2. В колонке "Мониторинг" выберите Insights.



3. Просмотрите различные доступные информационные панели и более подробно изучите различные метрики и журналы, доступные для кластера, узлов, контроллеров и развернутых контейнеров.



4. Чтобы просмотреть информационные панели «Контейнеры» и просмотреть более подробную информацию о каждом контейнере, выберите вкладку «Контейнеры».

NAME	STATUS	95TH %	POD	NODE	RESTARTS	UPTIME	TREND 95TH % (1 BAR = 1H)
omsagent	Ok	9%	14 mc	omsagent-rr-67569...	aks-agentpool-216...	0	23 hours
omsagent	Ok	6%	8 mc	omsagent-fq2jm	aks-agentpool-216...	0	23 hours
omsagent	Ok	5%	8 mc	omsagent-c25w9	aks-agentpool-216...	0	22 hours
tunnel-front	Ok	5%	94 mc	tunnelfront-867765...	aks-agentpool-216...	0	23 hours
heapster-nanny	Ok	1%	0.7 mc	heapster-5fc7766c9...	aks-agentpool-216...	0	23 hours
heapster	Ok	1%	0.9 mc	heapster-5fc7766c9...	aks-agentpool-216...	0	23 hours
main	Ok	0.5%	0.5 mc	kubernetes-dashbo...	aks-agentpool-216...	0	23 hours
kubedns	Ok	0.3%	6 mc	kube-dns-v20-864b...	aks-agentpool-216...	0	23 hours
kubedns	Ok	0.3%	5 mc	kube-dns-v20-864b...	aks-agentpool-216...	0	23 hours
kube-proxy	Ok	0.2%	5 mc	kube-proxy-mcblc	aks-agentpool-216...	0	23 hours
kube-proxy	Ok	0.2%	4 mc	kube-proxy-cz6bq	aks-agentpool-216...	0	23 hours
healthz	Ok	0.2%	4 mc	kube-dns-v20-864b...	aks-agentpool-216...	0	23 hours
healthz	Ok	0.2%	4 mc	kube-dns-v20-864b...	aks-agentpool-216...	0	23 hours
sidecar	Ok	0.1%	2 mc	kube-dns-v20-864b...	aks-agentpool-216...	0	23 hours
sidecar	Ok	0.1%	2 mc	kube-dns-v20-864b...	aks-agentpool-216...	0	23 hours

5. Теперь выполните фильтрацию по имени контейнера и найдите веб-контейнеры. Вы увидите все контейнеры, созданные в кластере Kubernetes, с именами модулей.

NAME	STATUS	95TH %	POD	NODE	RESTARTS	UPTIME	TREND 95TH % (1 BAR = 1H)
web	Ok	0%	0.7 mc	web-6dbc6d4d56-s...	aks-agentpool-216...	0	12 hours
web	Done	-	-	web-c7fd8c9d-7lwz	aks-agentpool-216...	0	?
web	Unk	-	-	web-69b854dd8d-ft...	aks-agentpool-216...	0	?
web	Unk	-	-	web-5b6ddccf-p27cx	aks-agentpool-216...	0	?
web	Unk	-	-	web-5bbfd59965-m...	aks-agentpool-216...	0	?

6. По умолчанию будет выбрана метрика использования ЦП, отображающая всю информацию о ЦП для выбранного контейнера. Чтобы переключиться на другую метрику, откройте раскрывающийся список метрик и выберите другую метрику.

NAME	Metric	POD	NODE	RESTARTS
web	CPU Usage (millicores)	web-6dbc6d4d56-s...	aks-agentpool-216...	0
web	CPU Usage (millicores)	web-c7fd8c9d-7lwz	aks-agentpool-216...	0
web	Memory working set	web-69b854dd8d-ft...	aks-agentpool-216...	0
web	Memory Rss	web-5b6ddccf-p27cx	aks-agentpool-216...	0
web	Memory Rss	web-5bbfd59965-m...	aks-agentpool-216...	0

7. После выбора любого модуля вся информация, относящаяся к выбранной метрике, будет отображаться на правой панели, и это будет иметь место при выборе любой другой метрики, детали будут отображаться на правой панели для выбранного пода.

»	 web
	Container
	View live data (preview)
	View in analytics ▼
Container Name	web
Container ID	e4033082d73d26f1da6b341d5f1307ff1b09654ef78a973fc48f 11f3fa4320d3
Container Status	running
Container Status Reason	-
Image	content-web
Image Tag	latest
Container Creation Time Stamp	12/5/2019, 4:18:30 AM
Start Time	12/5/2019, 4:18:30 AM
Finish Time	-
CPU Limit	1900 mc
CPU Request	1000 mc
Memory Limit	4.45 GB
Memory Request	128 MB
Last reported	16 secs ago

▷ Environment Variables

8. Чтобы отобразить журналы для любого контейнера, просто выберите его и просмотрите правую панель, и вы найдете опцию «Просмотр журналов контейнера», в которой будут перечислены все журналы для этого конкретного контейнера.

```

let startTime = datetime('2019-12-04T17:00:00.000Z');
let endTime = datetime('2019-12-05T17:08:46.532Z');
let ContainerIdList = kubeNodeInventory
    .where TimeGenerated >= startTime and TimeGenerated < endTime
    .where ContainerName == '392cfe6a-1716-11ea-b91c-1a158b045cd5/web'
    .select ContainerId
    .distinct ContainerID;
ContainerLog
    where TimeGenerated >= startTime and TimeGenerated < endTime
    where ContainerId in (ContainerIdList)
    project LogEntrySource, LogEntry, TimeGenerated, Computer, Image, Name, ContainerID
Completed
  
```

TimeGenerated [UTC]	LogEntrySource	LogEntry	Computer	Image	Name	ContainerID
12/5/2019, 4:18:30.652 AM	stdout	Running	aks-agentpool-21681325-0		e4033082d73d26f1da6b341d5f1307ff09054ef78a973fc48ff1f3fa4320d3	

9. Для каждой записи журнала вы можете отобразить дополнительную информацию, развернув запись в журнале, чтобы просмотреть приведенные ниже сведения.

```

let startTime = datetime('2019-12-04T17:00:00.000Z');
let endTime = datetime('2019-12-05T17:08:46.532Z');
let ContainerIdList = kubeNodeInventory
    .where TimeGenerated >= startTime and TimeGenerated < endTime
    .where ContainerName == '392cfe6a-1716-11ea-b91c-1a158b045cd5/web'
    .select ContainerId
    .distinct ContainerID;
ContainerLog
    where TimeGenerated >= startTime and TimeGenerated < endTime
    where ContainerId in (ContainerIdList)
    project LogEntrySource, LogEntry, TimeGenerated, Computer, Image, Name, ContainerID
Completed
  
```

TimeGenerated [UTC]	LogEntrySource	LogEntry	Computer	Image	Name	ContainerID
12/5/2019, 4:18:30.652 AM	stdout	Running	aks-agentpool-21681325-0	e4033082d73d26f1da6b341d5f1307ff09054ef78a973fc48ff1f3fa4320d3		
	LogEntrySource	stdout				
	LogEntry	Running				
	TimeGenerated [UTC]	2019-12-05T04:18:30.652Z				
	Computer	aks-agentpool-21681325-0				
	ContainerID	e4033082d73d26f1da6b341d5f1307ff09054ef78a973fc48ff1f3fa4320d3				

Упражнение 3: масштабируйте приложение и тестируйте высокую доступность

На этом этапе вы развернули один экземпляр контейнеров веб-служб и служб API. В этом упражнении вы увеличите количество экземпляров контейнера для веб-службы и масштабируете интерфейсную часть существующего кластера.

Задача 1. Увеличение количества экземпляров службы на портале Azure.

В этой задаче вы увеличите количество экземпляров для развертывания API в колонке AKS Azure Portal. Во время развертывания вы увидите изменение статуса.

1. В колонке AKS на портале Azure выберите «Рабочие нагрузки», а затем выберите развертывание API.
2. Выберите YAML в загружающемся окне и прокрутите вниз, пока не найдете реплики. Измените количество реплик на 2, а затем выберите Просмотр + сохранение. При появлении запроса установите флажок Подтвердить изменение манифеста и выберите Сохранить.

The screenshot shows the Azure AKS Deployment YAML editor. On the left, there's a sidebar with icons for Overview, Events, Insights, Live logs, and Changelogs. The 'YAML' icon is highlighted with a red box. On the right, the YAML code for a deployment is displayed. The 'replicas' field is highlighted with a red box and has the value '2' over it. Below the code, there are 'Review + save' and 'Discard' buttons, with 'Review + save' also highlighted with a red box.

```
130      'f:replicas': {}
131      'f:updatedReplicas': {}
132 spec:
133   replicas: 2
134   selector:
135     matchLabels:
136       app: api
137   template:
138     metadata:
139       name: api
140       creationTimestamp: null
141     labels:
142       app: api
143   spec:
144     containers:
145       - name: api
```

Примечание: Если развертывание завершается быстро, вы можете не увидеть состояния ожидания развертывания на портале, как описано в следующих шагах.

3. В представлении «Набор реплик» для API вы увидите, что он развертывается, и что есть один работоспособный и один ожидающий экземпляры.

fabmedical-cnr | Workloads

Replica sets

Name	Namespace	Ready	Current	Age	Images
coredns-autoscaler-5b6cbd75d7	kube-system	✓ 1/1	1	19 hours	mcr.microsoft.com/oss/kubern...
coredns-b94d8b788	kube-system	✓ 2/2	2	19 hours	mcr.microsoft.com/oss/kubern...
metrics-server-77c8679d7d	kube-system	✓ 1/1	1	19 hours	mcr.microsoft.com/oss/kubern...
omsagent-rs-6ccc6ddf9	kube-system	✓ 1/1	1	19 hours	mcr.microsoft.com/azuremonitor...
tunelfront-7d9f7947fc	kube-system	✓ 1/1	1	19 hours	mcr.microsoft.com/aks/hcp/hcp-...
api-5fd454b7d5	default	⚠ 1/2	2	13 hours	fabmedicalcn.azurecr.io/content...
web-944f69f45	default	✓ 1/1	1	12 hours	fabmedicalcn.azurecr.io/content...

- В меню навигации выберите Рабочие нагрузки. Обратите внимание, что при развертывании арі отображается предупреждение и отображается количество модулей 1 из 2 (показано как 1/2).

fabmedical-sjw | Workloads

Deployments

Name	Namespace	Ready
coredns-autoscaler	kube-system	✓ 1/1
coredns	kube-system	✓ 2/2
metrics-server	kube-system	✓ 1/1
omsagent-rs	kube-system	✓ 1/1
tunelfront	kube-system	✓ 1/1
api	default	⚠ 1/2
web	default	✓ 1/1

Примечание. Если вы получаете сообщение об ошибке, указывающее, что ЦП недостаточно, это нормально. Мы увидим, как с этим справиться, в следующей задаче (подсказка: вы можете использовать параметр Insights на портале AKS Azure, чтобы просмотреть статус узла и просмотреть журналы событий Kubernetes).

На этом этапе мы представляем обзор состояния окружающей среды:

- Одно развертывание и один набор реплик являются работоспособными для веб-службы.
 - Развертывание арі и набор реплик находятся в состоянии предупреждения.
 - Два модуля исправны в пространстве имен "по умолчанию".
- Откройте веб-приложение Contoso Neuro Conference. Приложение должно работать без ошибок при переходе на страницы докладчиков и сеансов.

- Перейдите на страницу /stats. Вы увидите информацию о среде хостинга, включая:
 - webTaskId: идентификатор задачи для экземпляра веб-службы.
 - taskId: идентификатор задачи для экземпляра службы API.
 - hostName: идентификатор имени хоста для экземпляра службы API.
 - pid: идентификатор процесса для экземпляра службы API.
 - mem: некоторые индикаторы памяти, возвращаемые экземпляром службы API.
 - counters: счетчики для самой службы, возвращенные экземпляром службы API.
 - uptime: время безотказной работы службы API.

Задача 2: устранение сбоев при инициализации реплик

В этой задаче вы устраниете неисправные реплики API. Эти сбои происходят из-за неспособности кластеров удовлетворить запрошенные ресурсы.

1. В колонке AKS на портале Azure выберите «Рабочие нагрузки», а затем выберите развертывание API. Выберите элемент навигации YAML.
2. Прокрутите экран YAML вниз и измените следующие элементы:
 - Измените порты и удалите hostPort. Два модуля не могут подключаться к одному и тому же порту хоста.
 - ports:
 - - containerPort: 3001
 - protocol: TCP
 - Измените ЦП и установите его на 100 м. ЦП разделен между всеми модулями на узле.
 - resources:
 - requests:
 - cpu: 100m
 - memory: 128Mi
3. Выберите «Обзор + сохранение», при появлении запроса подтвердите изменения и нажмите «Сохранить».

```

apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    app: api
  spec:
    containers:
      - name: web
        image: fabmedicalsjw.azurecr.io/content-api
        ports:
          - containerPort: 3001
            protocol: TCP
        env:
          - name: MONGODB_CONNECTION
            valueFrom:
              secretKeyRef:
                name: cosmosdb
                key: db
        resources:
          requests:
            cpu: 100m
            memory: 128Mi
        livenessProbe:
          httpGet:
            path: /
            port: 3001
            scheme: HTTP
            initialDelaySeconds: 30
            timeoutSeconds: 10
            periodSeconds: 20
            successThreshold: 1
            failureThreshold: 3
            terminationMessagePath: /dev/termination-log

```

- 4.
5. Вернитесь к главному представлению рабочих нагрузок на портале AKS Azure, и теперь вы увидите, что развертывание исправно с двумя работающими модулями.

<input type="checkbox"/>	Name	Namespace	Ready	Up-to-date	Available	Age
<input type="checkbox"/>	coredns	kube-system	✓ 2/2	2	2	19 hours
<input type="checkbox"/>	coredns-autoscaler	kube-system	✓ 1/1	1	1	19 hours
<input type="checkbox"/>	metrics-server	kube-system	✓ 1/1	1	1	19 hours
<input type="checkbox"/>	omsagent-1s	kube-system	✓ 1/1	1	1	19 hours
<input type="checkbox"/>	tunneffront	kube-system	✓ 1/1	1	1	19 hours
<input checked="" type="checkbox"/>	api	default	✓ 2/2	2	2	14 hours
<input type="checkbox"/>	web	default	✓ 1/1	1	1	12 hours

Задача 3: перезапустить контейнеры и протестировать НА

В этой задаче вы перезапустите контейнеры и убедитесь, что перезапуск не влияет на работающую службу.

1. Откройте образец веб-приложения и перейдите на страницу «Статистика», как показано.

The screenshot shows a web application interface. At the top left is the logo 'CONTOSO NEURO'. To the right are three navigation links: 'Speakers', 'Sessions', and 'Stats'. Below the header is a large, dark banner image. Underneath the banner, the word 'Stats' is displayed in a large, bold, white font. Below 'Stats' is a table with the following data:

webTaskId	1
taskId	18
hostName	api-f4948fb86-bj9cv
pid	18
mem	{ "rss": 68993024, "heapTotal": 18808832, "heapUsed": 16981256, "external": 19266663, "arrayBuffers": 18288142 }
counters	{ "stats": 1, "speakers": 4, "sessions": 4 }
uptime	535.373722288

2. В колонке AKS на портале Azure откройте развертывание арі и увеличьте необходимое количество реплик до 4.

The screenshot shows the Azure API Management deployment interface. On the left, there's a sidebar with icons for Overview, YAML, Events, Insights, Live logs, and Changelogs. The 'YAML' icon is selected. At the top, there's a search bar and a refresh button. Below the search bar, there are tabs for 'YAML' and 'JSON', with 'YAML' being the active tab. The main area displays the YAML configuration for a service instance. A red box highlights the 'replicas: 4' line in the 'spec' section.

```

    117
    118
    119
    120
    121
    122
    123
    124
    125
    126
    127
    128
    129
    130
    131
    132 spec:
    133   replicas: 4
    134   selector:

```

- Через несколько секунд вы обнаружите, что развертывание API теперь успешно выполняет 4 реплики.
- Вернитесь на вкладку браузера с загруженной страницей статистики веб-приложения. Обновляйте страницу снова и снова. Вы не увидите никаких ошибок, но вы будете видеть, что имя хоста api периодически меняется между четырьмя экземплярами api pod. Идентификатор задачи и pid также могут меняться между четырьмя экземплярами api pod.

The screenshot shows two data cards from the Azure application insights stats page. Both cards have identical fields: webTaskId (18), taskId (18), and uptime (50570.163). The first card has a hostName value of 'api-2547917202-glwz7'. The second card has a hostName value of 'api-2547917202-51gjc'. Both hostName values are highlighted with red boxes.

webTaskId	18
taskId	18
hostName	api-2547917202-glwz7
pid	18
mem	{"rss":36429824,"heapTotal":19582720,"heapUsed":13544928,"external":232284}
counters	{"stats":6,"speakers":2,"sessions":1}
uptime	50570.163

webTaskId	18
taskId	18
hostName	api-2547917202-51gjc
pid	18
mem	{"rss":36376576,"heapTotal":19582720,"heapUsed":13479320,"external":237370}
counters	{"stats":5,"speakers":3,"sessions":1}
uptime	3012.69

- После обновления достаточного количества раз, чтобы увидеть, что значение hostName изменяется, а служба остается работоспособной, вы можете открыть представление Replica Sets для API на портале Azure.
- В этом представлении вы можете увидеть, что значение hostName, показанное на странице статистики веб-приложения, соответствует именам модулей для запущенных модулей.

Search (Ctrl+ /) <> Refresh

Overview

YAML

Events

Namespace: default

Labels: app: api, pod-template-hash: 64fb98c7c8

Selector: app=api, pod-template-hash=64fb98c7c8

See more

Pods

Delete Show labels

<input type="checkbox"/>	Name	Ready	Status
<input type="checkbox"/>	api-64fb98c7c8-8r5ks	✓ 1/1	Running
<input type="checkbox"/>	api-64fb98c7c8-245b2	✓ 1/1	Running
<input type="checkbox"/>	api-64fb98c7c8-5v2rc	✓ 1/1	Running
<input type="checkbox"/>	api-64fb98c7c8-j9ppl	✓ 1/1	Running

7. Выберите случайным образом два модуля и нажмите «Удалить». Выберите Подтвердить удаление и снова нажмите Удалить.

Pods

Delete Show labels

<input type="checkbox"/>	Name	Ready	Status
<input checked="" type="checkbox"/>	api-64fb98c7c8-8r5ks	✓ 1/1	Running
<input checked="" type="checkbox"/>	api-64fb98c7c8-245b2	✓ 1/1	Running
<input type="checkbox"/>	api-64fb98c7c8-5v2rc	✓ 1/1	Running
<input type="checkbox"/>	api-64fb98c7c8-j9ppl	✓ 1/1	Running

8. Kubernetes запустит новые поды, чтобы обеспечить необходимое количество реплик. В зависимости от вашего представления вы можете увидеть, что старые экземпляры завершаются, а новые экземпляры создаются.

	Name	Ready	Status
<input type="checkbox"/>	api-64fb98c7c8-8r5ks	✓ 1/1	Terminating
<input checked="" type="checkbox"/>	api-64fb98c7c8-245b2	✓ 1/1	Terminating
<input type="checkbox"/>	api-64fb98c7c8-5v2rc	✓ 1/1	Running
<input type="checkbox"/>	api-64fb98c7c8-j9ppl	✓ 1/1	Running
<input type="checkbox"/>	api-64fb98c7c8-fntxk	⚠ 0/1	ContainerCreating
<input type="checkbox"/>	api-64fb98c7c8-zlmn4	⚠ 0/1	ContainerCreating

9. Вернитесь к развертыванию API и уменьшите масштаб до 1 реплики. См. Шаг 2 выше, чтобы узнать, как это сделать, если вы не уверены.
10. Вернитесь на страницу статистики примера веб-сайта в браузере и обновите, пока Kubernetes сокращает количество подов. Вы заметите, что отображается только одно имя хоста API, хотя вы все еще можете видеть несколько запущенных модулей в представлении набора реплик API. Несмотря на то, что запущено несколько модулей, Kubernetes больше не будет отправлять трафик в модули, которые он выбрал для завершения. Через несколько секунд в представлении набора реплик API отобразится только один модуль.

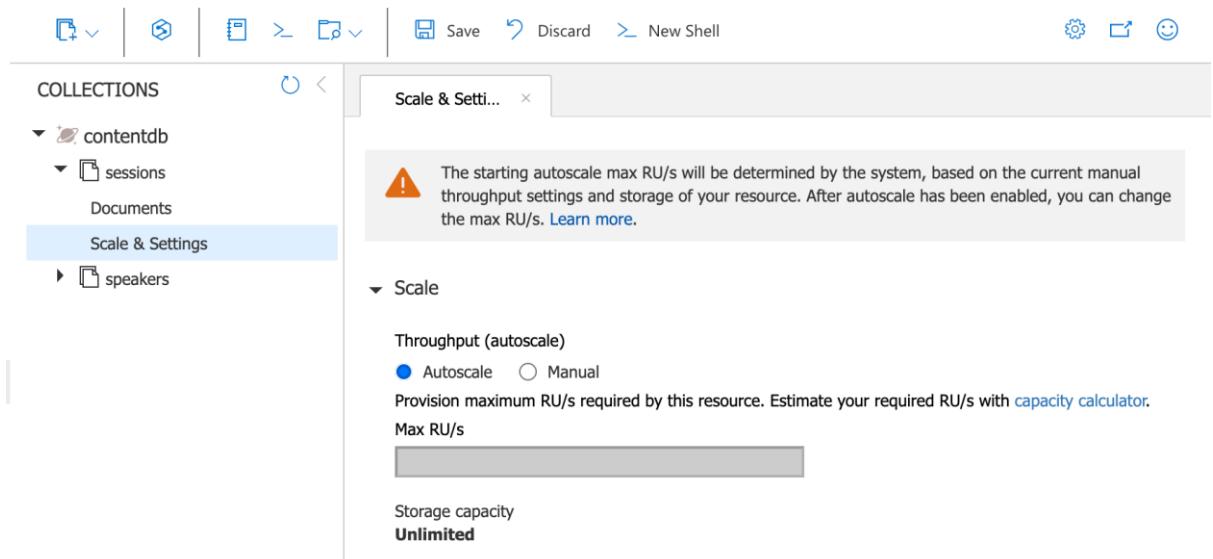
	Name	Ready	Status
<input type="checkbox"/>	api-64fb98c7c8-5v2rc	⚠ 0/1	Terminating
<input type="checkbox"/>	api-64fb98c7c8-j9ppl	⚠ 0/1	Terminating
<input type="checkbox"/>	api-64fb98c7c8-fntxk	✓ 1/1	Running
<input type="checkbox"/>	api-64fb98c7c8-zlmn4	⚠ 0/1	Terminating

Задача 4. Настройка автомасштабирования Cosmos DB.

В этой задаче вы настроите автомасштабирование в Azure Cosmos DB.

1. На портале Azure перейдите к учетной записи fabmedical-[SUFFIX] Azure Cosmos DB.
2. Выберите **Data Explorer**.

3. В проводнике данных разверните базу данных contentdb, затем разверните коллекцию сеансов.
4. В коллекции сеансов выберите Scale & Settings.
5. В разделе Scale & Settings выберите Autoscale для параметра в разделе.



6. Выполните ту же задачу, чтобы включить автоматическое масштабирование пропускной способности для коллекции speakers.

Задача 5. Тестирование автомасштабирования Cosmos DB

В этой задаче вы запустите сценарий тестирования производительности, который проверит функцию автомасштабирования в Azure Cosmos DB, чтобы вы могли увидеть, что теперь масштабирование будет превышать 400 RU/s.

1. На портале Azure перейдите к учетной записи fabmedical-[SUFFIX] Cosmos DB.
2. В разделе «Настройки» выберите «Строка подключения».
3. На панели строки подключения скопируйте значения HOST, USERNAME и PRIMARY PASSWORD. Сохраните их для использования позже.

The screenshot shows the Azure Cosmos DB Connection String page for the fabmedical-cp account. The 'Read-write Keys' tab is selected. The highlighted area contains the following connection string fields:

- HOST: fabmedical-cp.documents.azure.com
- PORT: 10255
- USERNAME: fabmedical-cp
- PRIMARY PASSWORD: HjJTu1ZK8CjX2g2j3vZ2QHBBuWVlvGEDKz11AZlrzBPc1owQ2D1QzSvudCfvd4ChD1TXmBiAS5HCu7Q61IVFA==

Примечание. В своей учетной записи Cosmos DB вы можете увидеть, что конечная точка хоста использует .mongo.cosmos.azure.com, который предназначен для Mongo DB версии 3.6. Показанная здесь конечная точка - .documents.azure.com, которая предназначена для Mongo DB версии 3.2. Вы можете использовать любую конечную точку для целей этой Задачи.

- Откройте Azure Cloud Shell и подключитесь по SSH к виртуальной машине агента сборки.
- На виртуальной машине агента сборки перейдите в каталог ~/Fabmedical.

```
cd ~/Fabmedical
```

- Выполните следующую команду, чтобы открыть сценарий perftest.sh для редактирования в Vim.
- В верхней части скрипта perftest.sh объявлено несколько переменных. Измените переменные хоста, имени пользователя и пароля, установив их значения в соответствующие значения строки подключения Cosmos DB, которые были скопированы ранее.

```
Bash
host="fabmedical-cp.documents.azure.com"
username="fabmedical-cp"
password="HjJTu1ZK8CjX2g2j3vZ2QHBBuWVlvGEDKz11AZlrzBPc1owQ2D1QzSvudCfvd4ChD1TXmBiAS5HCu7Q61IVFA=="
dbname="contentdb"
port="10255"

for i in {1..250}; do
    mongo $host:$port/$dbname \
        -u $username -p $password \
```

- Сохраните файл и выйдите из Vim.

9. Выполните следующую команду, чтобы выполнить сценарий `perftest.sh` для запуска небольшого нагрузочного теста для Cosmos DB. Этот скрипт будет использовать RU в Cosmos DB, вставляя множество документов в контейнер Sessions.

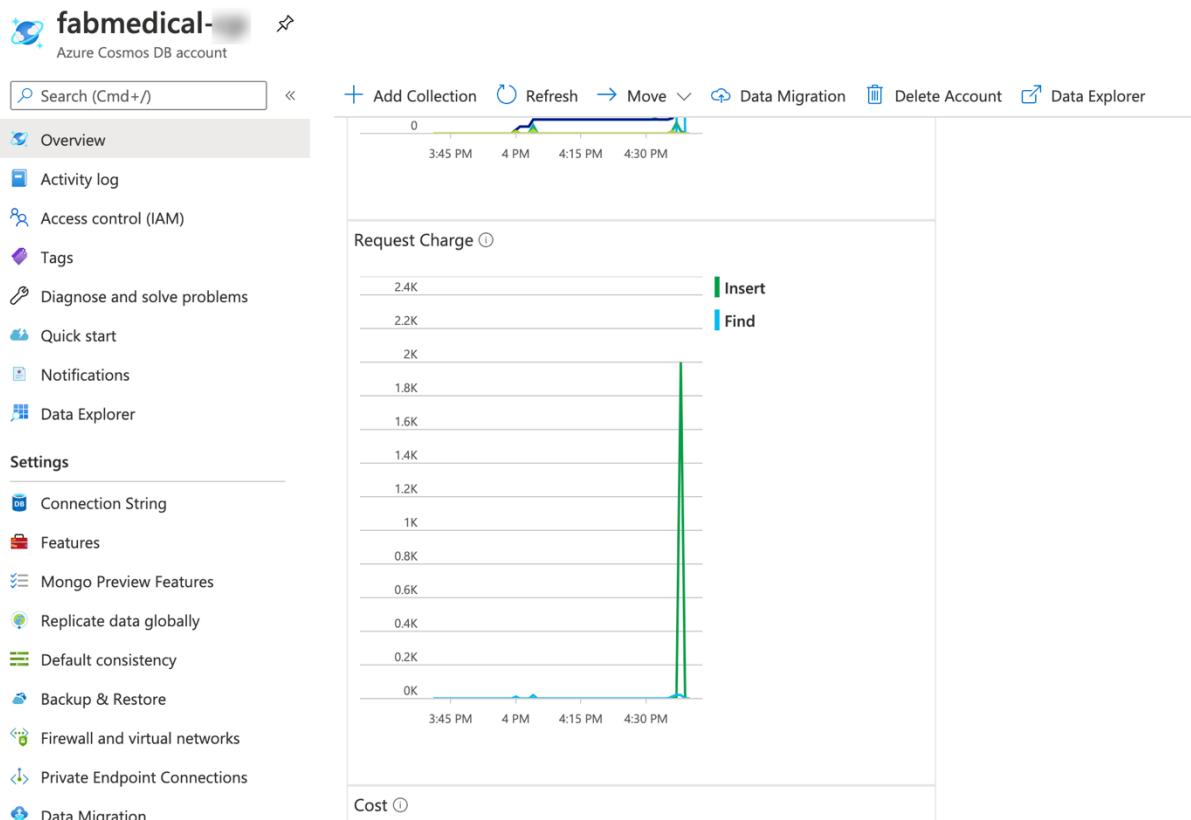
```
bash ./perftest.sh
```

Примечание. Скрипт завершится через минуту.

10. После завершения сценария вернитесь к учетной записи Cosmos DB на портале Azure.
11. После завершения сценария вернитесь к учетной записи Cosmos DB на портале Azure.

Примечание. Для отображения активности в коллекции Cosmos DB в журнале активности может потребоваться 2–5 минут. Подождите пару минут, а затем обновите панель, если недавняя плата за запрос не отображается прямо сейчас.

12. Обратите внимание, что плата за запрос теперь показывает, что в учетной записи Cosmos DB была активность, которая превысила предел в 400 единиц в секунду, который был ранее установлен до включения автомасштабирования.



Упражнение 4: Работа со службами и маршрутизация трафика приложений

В предыдущем задании мы ввели ограничение на свойства масштабирования службы. В этом упражнении вы настроите развертывания арі для создания модулей, которые используют динамическое сопоставление портов, чтобы устранить ограничение ресурсов порта во время масштабных операций.

Службы Kubernetes могут обнаруживать порты, назначенные каждому модулю, что позволяет запускать несколько экземпляров модуля на одном узле агента - что невозможно при настройке определенного статического порта (например, 3001 для службы API).

Задача 1. Обновите внешнюю службу для поддержки динамического обнаружения с помощью балансировщика нагрузки.

В этой задаче вы обновите веб-службу, чтобы она поддерживала динамическое обнаружение через балансировщик нагрузки Azure.

1. В меню ресурсов AKS Kubernetes выберите «Развертывания» в разделе «Рабочие нагрузки». Из списка выберите веб-развертывание.
2. Выберите YAML, затем перейдите на вкладку JSON.
3. Сначала найдите узел реплик и обновите необходимое количество до 4.
4. Затем перейдите к спецификации веб-контейнеров, как показано на снимке экрана. Удалите запись hostPort для сопоставления портов веб-контейнера.

```
168     ]
169   },
170 },
171 "spec": {
172   "replicas": 4,
173   "selector": {
174     "matchLabels": {
175       "app": "web"
176     }
177   },
178   "template": {
179     "metadata": {
180       "name": "web",
181       "creationTimestamp": null,
182       "labels": {
183         "app": "web"
184       }
185     },
186     "spec": {
187       "containers": [
188         {
189           "name": "web",
190           "image": "fabmedicalcnr.azurecr.io/content-web",
191           "ports": [
192             {
193               "containerPort": 3000,
194               "protocol": "TCP"
195             }
196           ],
197           "env": [
198             {
199               "name": "CONTENTWEBPORT",
200               "value": "3000"
201             }
202           ]
203         }
204       ]
205     }
206   }
207 }
```

[Review + save](#) [Discard](#)

5. Выберите «Обзор + сохранение», затем подтвердите изменение и нажмите «Сохранить».
6. Проверьте состояние горизонтального масштабирования, обновив представление веб-развертывания. В меню навигации выберите «Модули» в разделе «Рабочие нагрузки». Выберите веб-модули. В этом представлении вы должны увидеть ошибку, подобную показанной на следующем снимке экрана.

Pods Replica sets

Delete Show labels

<input type="checkbox"/>	Name	Ready	Status
<input type="checkbox"/>	web-57d76bd7d4-6cljq	✓ 1/1	Running
<input type="checkbox"/>	web-57d76bd7d4-dxx8z	⚠ 0/1	Pending
<input type="checkbox"/>	web-57d76bd7d4-qfsmg	⚠ 0/1	Pending
<input type="checkbox"/>	web-57d76bd7d4-zs4mr	✓ 1/1	Running

Подобно развертыванию API, веб-развертывание использовало фиксированный hostPort, и ваши возможности масштабирования были ограничены количеством доступных узлов агента. Однако после решения этой проблемы для веб-службы путем удаления параметра hostPort веб-развертывание по-прежнему не может масштабироваться за пределы двух модулей из-за ограничений ЦП. Разворачивание требует больше ЦП, чем требуется веб-приложению, поэтому вы исправите это ограничение в следующей задаче.

Задача 2. Отрегулируйте ограничения ЦП, чтобы улучшить масштабирование

В этой задаче вы измените требования к ЦП для веб-службы, чтобы ее можно было масштабировать до большего количества экземпляров.

- Снова откройте представление JSON для веб-развертывания, а затем найдите требования к ресурсам ЦП для веб-контейнера. Измените это значение на 125 м.

```

230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
  "protocol": "TCP"
},
"env": [
{
  "name": "CONTENT_API_URL",
  "value": "http://api:3001"
},
],
"resources": {
  "requests": {
    "cpu": "125m",
    "memory": "128Mi"
  }
},

```

- Выберите «Обзор + сохранение», подтвердите изменение, а затем выберите «Сохранить», чтобы обновить развертывание.
- В меню навигации выберите «Наборы реплик» в разделе «Рабочие нагрузки». В списке Replica Sets представления выберите набор веб-реплик.
- По завершении обновления развертывания четыре веб-модуля должны отображаться в рабочем состоянии.

Pods Replica sets

Delete Show labels

<input type="checkbox"/>	Name	Ready	Status
<input type="checkbox"/>	web-57d76bd7d4-6cljq	✓ 1/1	Terminating
<input type="checkbox"/>	web-57d76bd7d4-zs4mr	⚠ 0/1	Terminating
<input type="checkbox"/>	web-b7b564445-kklhb	✓ 1/1	Running
<input type="checkbox"/>	web-b7b564445-l76rj	✓ 1/1	Running
<input type="checkbox"/>	web-b7b564445-5zl44	✓ 1/1	Running
<input type="checkbox"/>	web-b7b564445-j9kgv	✓ 1/1	Running

Задача 3. Добавьте поддержку Application Insights

В этой задаче вы отредактируете исходный код веб-приложения, чтобы добавить Application Insights и обновить образ Docker, используемый при развертывании. Затем вы выполните скользящее обновление, чтобы продемонстрировать, как развернуть изменение кода.

- Выполните эту команду в Azure Cloud Shell, чтобы получить ключ инструментария для ресурса Content-Web Application Insights:

```
az resource show -g fabmedical-[SUFFIX] -n content-web --resource-type "Microsoft.Insights/components" --query properties.InstrumentationKey -o tsv
```

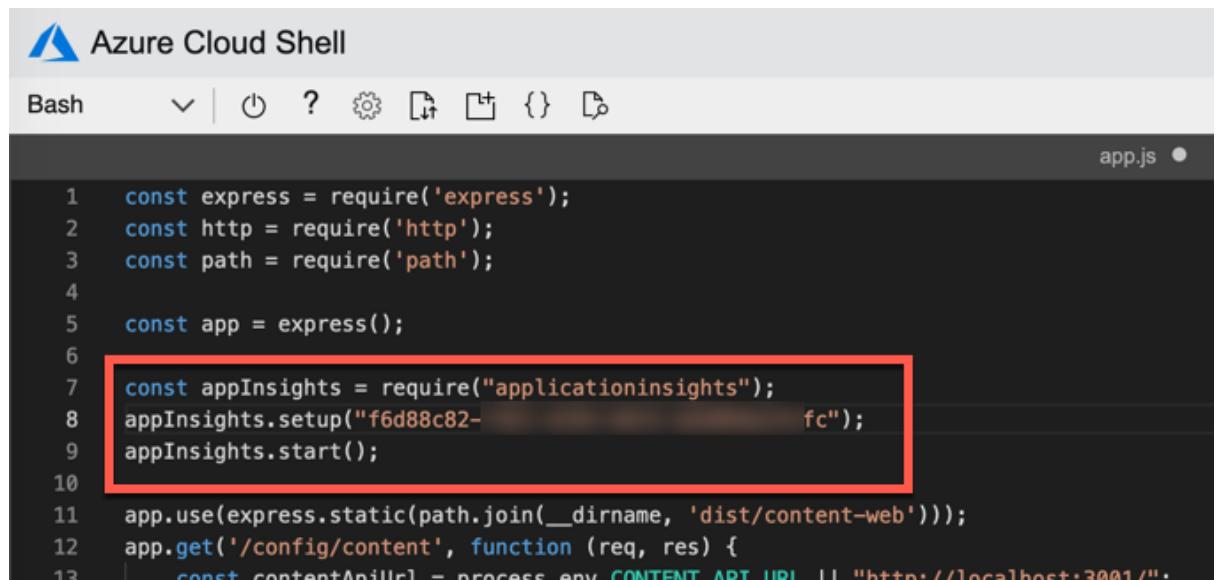
Скопируйте это значение. Вы будете использовать это позже.

Примечание: если у вас пустой результат, убедитесь, что введенная вами команда относится к нужному ресурсу.

- На виртуальной машине обновите файлы репозитория fabmedical, извлекая последние изменения из репозитория git:
- cd ~/fabmedical/content-web
git pull
- Установите поддержку Application Insights.

```
npm install applicationinsights --save
```

5. Отредактируйте файл app.js с помощью Vim или Visual Studio Code и добавьте следующие строки сразу после создания экземпляра express в строке 6:
6. const appInsights = require("applicationinsights");
7. appInsights.setup("[YOUR APPINSIGHTS KEY]");
appInsights.start();



```
Bash      v | ⏻ ? ⚙️ ⌂ ⌃ ⌄ {} ⌁ app.js ●

1  const express = require('express');
2  const http = require('http');
3  const path = require('path');
4
5  const app = express();
6
7  const appInsights = require("applicationinsights");
8  appInsights.setup("f6d88c82-fc");  
  appInsights.start();
9
10 app.use(express.static(path.join(__dirname, 'dist/content-web')));
11 app.get('/config/content', function (req, res) {
12   const contentAppliance = process.env.CONTENT_APPLIANCE || "http://localhost:3001/";
13 }
```

8. Сохраните изменения и закройте редактор.
9. Отправьте эти изменения в свой репозиторий, чтобы GitHub Actions CI построил и развернул новый образ контейнера.
10. git add .
11. git commit -m "Added Application Insights"
git push
12. Посетите действие content-web для вашего репозитория GitHub Fabmedical и посмотрите, как новый образ развертывается в вашем кластере Kubernetes.
13. Пока выполняется это обновление, верните портал Azure в браузере.
14. В меню навигации выберите «Наборы реплик» в разделе «Рабочие нагрузки». В этом представлении вы увидите новый набор реплик для Интернета, который может все еще находиться в процессе развертывания (как показано ниже) или уже полностью развернут.

15. Во время развертывания вы можете перейти к веб-приложению и посетить страницу статистики по адресу `/stats`. Обновляйте страницу по мере выполнения непрерывного обновления. Обратите внимание, что служба работает нормально, а задачи продолжают балансировать нагрузку.

<code>webTaskId</code>	1
<code>taskId</code>	17
<code>hostName</code>	api-f8ddd5667-5svf6
<code>pid</code>	17
<code>mem</code>	{ "rss": 70266880, "heapTotal": 18546688, "heapUsed": 16571136, "external": 19309316, "arrayBuffers": 18330755 }
<code>counters</code>	{ "stats": 6, "speakers": 0, "sessions": 0 }
<code>uptime</code>	7920.812241335

Задача 4: настроить Kubernetes Ingress

В этой задаче вы настроите Kubernetes Ingress с использованием прокси-сервера nginx, чтобы воспользоваться преимуществами маршрутизации на основе адресов и работой с TLS.

1. В Azure Cloud Shell выполните следующую команду, чтобы добавить стабильный репозиторий Helm nginx:

```
helm repo add ingress-nginx https://kubernetes.github.io/ingress-nginx
```

2. Обновите список пакетов Helm.

```
helm repo update
```

Примечание. Если вы получили сообщение «Репозитории не найдены». ошибка, затем выполните следующую команду. Это вернет официальный "стабильный" репозиторий Helm.

```
helm repo add stable https://charts.helm.sh/stable
```

3. Создайте пространство имен в Kubernetes для установки ресурсов Ingress.

```
kubectl create namespace ingress-demo
```

4. Установите ресурс контроллера Ingress для обработки входящих запросов по мере их поступления. Контроллер Ingress получит собственный общедоступный IP-адрес на балансировщике нагрузки Azure и сможет обрабатывать запросы для нескольких служб через порты 80 и 443.

```
5. helm install nginx-ingress ingress-nginx/ingress-inginx \
6. --namespace ingress-demo \
7. --set controller.replicaCount=2 \
8. --set controller.nodeSelector."beta\.kubernetes\.io/os"=linux \
9. --set defaultBackend.nodeSelector."beta\.kubernetes\.io/os"=linux \
--set
controller.admissionWebhooks.patch.nodeSelector."beta\.kubernetes\.io
/os"=linux
```

10. На портале Azure в разделе «Службы и входящие данные» скопируйте IP-адрес для внешнего IP-адреса службы nginx-ingress-RANDOM-nginx-ingress.

Name	Namespace	Status	Type	Cluster IP	External IP
nginx-ingress-ingress-nginx...	ingress-demo	Ok	LoadBalancer	10.0.56.75	13.66.208.254
nginx-ingress-ingress-nginx...	ingress-demo	Ok	ClusterIP	10.0.153.239	

Примечание. Обновление может занять несколько минут. Кроме того, вы можете найти IP-адрес с помощью следующей команды в Azure Cloud Shell.

```
kubectl get svc --namespace ingress-demo
```

@Azure:~\$ kubectl get svc --namespace kube-system	NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)	AGE
healthmodel-replicaset-service	ClusterIP	10.0.154.232	<none>	25227/TCP		12h
ingress-controller-nginx-ingress-controller	LoadBalancer	10.0.26.40	51.144.226.102	80:31716/TCP,443:30115/TCP		8m
ingress-controller-nginx-ingress-default-backend	ClusterIP	10.0.231.84	<none>	80/TCP		8m
kube-dns	ClusterIP	10.0.0.10	<none>	53/UDP,53/TCP		12h
kubernetes-dashboard	ClusterIP	10.0.198.13	<none>	80/TCP		12h
metrics-server	ClusterIP	10.0.196.10	<none>	443/TCP		12h

11. Откройте колонку «Группы ресурсов» портала Azure и найдите группу ресурсов, которая была автоматически создана для размещения пулов узлов для

AKS. Он будет иметь формат именования MC_fabmedical-[SUFFIX]_fabmedical-[SUFFIX]_[REGION].

12. В Azure Cloud Shell создайте сценарий для обновления общедоступного DNS-имени для входящего внешнего IP-адреса.

```
code update-ip.sh
```

Вставьте следующее как содержимое. Обязательно замените в скрипте следующие заполнители:

- [INGRESS PUBLIC IP]: Замените это IP-адресом, скопированным с шага 5.
- [AKS NODEPOOL RESOURCE GROUP]: Замените именем группы ресурсов, скопированным с шага 6.
- [SUFFIX]: Замените его тем же значением SUFFIX, которое использовалось ранее в этой лабораторной работе.

```
#!/bin/bash

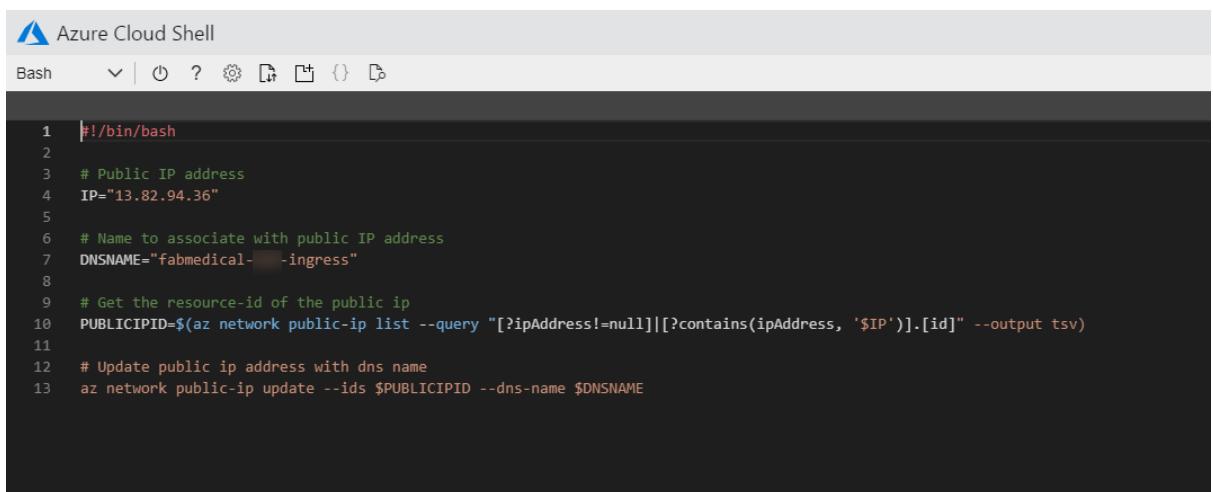
# Public IP address
IP="[INGRESS PUBLIC IP]"

# Resource Group that contains AKS Node Pool
KUBERNETES_NODE_RG="[AKS NODEPOOL RESOURCE GROUP]"

# Name to associate with public IP address
DNSNAME="fabmedical-[SUFFIX]-ingress"

# Get the resource-id of the public ip
PUBLICIPID=$(az network public-ip list --resource-group $KUBERNETES_NODE_RG --query "[?ipAddress!=null]|[?contains(ipAddress, '$IP')].[id]" --output tsv)

# Update public ip address with dns name
az network public-ip update --ids $PUBLICIPID --dns-name $DNSNAME
```



The screenshot shows the Azure Cloud Shell interface with the title "Azure Cloud Shell". Below the title is a toolbar with icons for Bash, Command History, Help, and others. The main area is a dark terminal window displaying the contents of the update-ip.sh script. The script uses numbered comments (1 through 13) to indicate where to replace placeholder values. The script starts with a shebang line, defines a variable for the public IP, specifies the resource group for the AKS node pool, defines a DNS name, and then uses the az command to get the resource ID of the public IP and update it with the specified DNS name.

```
1  #!/bin/bash
2
3  # Public IP address
4  IP="13.82.94.36"
5
6  # Name to associate with public IP address
7  DNSNAME="fabmedical-[SUFFIX]-ingress"
8
9  # Get the resource-id of the public ip
10 PUBLICIPID=$(az network public-ip list --resource-group $KUBERNETES_NODE_RG --query "[?ipAddress!=null]|[?contains(ipAddress, '$IP')].[id]" --output tsv)
11
12 # Update public ip address with dns name
13 az network public-ip update --ids $PUBLICIPID --dns-name $DNSNAME
```

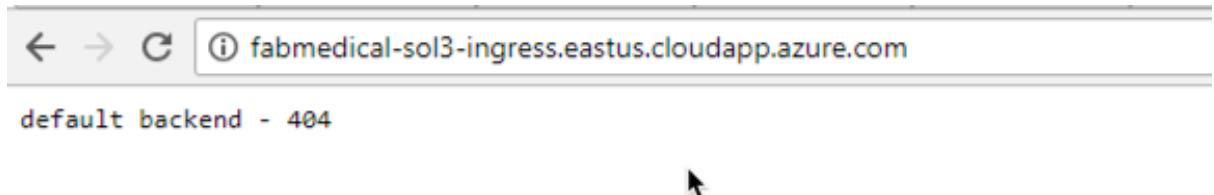
13. Сохраните изменения и закройте редактор.
14. Запускаем скрипт обновления.

```
bash ./update-ip.sh
```

15. Проверьте обновление IP, посетив URL-адрес в своем браузере.

Примечание. В настоящее время получение сообщения 404 является нормальным.

```
http://fabmedical-[SUFFIX]-ingress.[AZURE-REGION].cloudapp.azure.com/
```



16. Используйте helm для установки cert-manager, инструмента, который может автоматически предоставлять сертификаты SSL с letsencrypt.org.

17. kubectl create namespace cert-manager

18.

19. kubectl label namespace cert-manager cert-manager.io/disable-validation=true

20.

```
kubectl apply --validate=false -f https://github.com/jetstack/cert-manager/releases/download/v1.0.1/cert-manager.yaml
```

21. Диспетчеру сертификатов потребуется специальный ресурс ClusterIssuer для обработки запросов SSL-сертификатов.

```
code clusterissuer.yml
```

Следующая конфигурация ресурсов должна работать как есть:

```
apiVersion: cert-manager.io/v1
kind: ClusterIssuer
metadata:
  name: letsencrypt-prod
spec:
  acme:
    # The ACME server URL
    server: https://acme-v02.api.letsencrypt.org/directory
    # Email address used for ACME registration
    email: user@fabmedical.com
    # Name of a secret used to store the ACME account private key
    privateKeySecretRef:
      name: letsencrypt-prod
    # Enable HTTP01 validations
```

```
solvers:
- http01:
  ingress:
    class: nginx
```

22. Сохраните изменения и закройте редактор.

23. Создайте эмитент с помощью kubectl.

```
kubectl create --save-config=true -f clusterissuer.yml
```

24. Теперь вы можете создать объект сертификата.

Примечание:

Cert-manager, возможно, уже создал для вас объект сертификата с помощью ingress-shim.

Чтобы убедиться, что сертификат был успешно создан, используйте команду kubectl describe certificate tls-secret.

Если сертификат уже доступен, перейдите к шагу 16.

```
code certificate.yml
```

Используйте следующее в качестве содержимого и обновите [SUFFIX] и [AZURE-REGION], чтобы они соответствовали вашему входному DNS-имени.

```
apiVersion: cert-manager.io/v1
kind: Certificate
metadata:
  name: tls-secret
spec:
  secretName: tls-secret
  dnsNames:
  - fabmedical-[SUFFIX]-ingress.[AZURE-REGION].cloudapp.azure.com
  issuerRef:
    name: letsencrypt-prod
    kind: ClusterIssuer
```

25. Сохраните изменения и закройте редактор.

26. Создайте сертификат с помощью kubectl.

```
kubectl create --save-config=true -f certificate.yml
```

Примечание. Чтобы проверить статус выдачи сертификата, используйте команду kubectl describe certificate tls-secret и найдите выходные данные событий, аналогичные приведенным ниже:

Type	Reason	Age	From	Message
-----	-----	-----	-----	-----
Normal	Generated	38s	cert-manager	Generated new private key
Normal	GenerateSelfSigned	38s	cert-manager	Generated temporary self signed certificate

```
Normal  OrderCreated      38s   cert-manager  Created Order
resource "tls-secret-3254248695"
Normal  OrderComplete     12s   cert-manager  Order "tls-
secret-3254248695" completed successfully
Normal  CertIssued       12s   cert-manager  Certificate
issued successfully
```

Прежде чем tls-secret станет доступным, может пройти от 5 до 30 минут. Это связано с задержкой, связанной с предоставлением сертификата TLS от letsencrypt.

27. Теперь вы можете создать входящий ресурс для контент-приложений.

```
code content.ingress.yml
```

Используйте следующее в качестве содержимого и обновите [SUFFIX] и [AZURE-REGION], чтобы они соответствовали вашему входному DNS-имени:

```
apiVersion: networking.k8s.io/v1beta1
kind: Ingress
metadata:
  name: content-ingress
  annotations:
    kubernetes.io/ingress.class: nginx
    nginx.ingress.kubernetes.io/rewrite-target: /$1
    nginx.ingress.kubernetes.io/use-regex: "true"
    nginx.ingress.kubernetes.io/ssl-redirect: "false"
    cert-manager.io/cluster-issuer: letsencrypt-prod
spec:
  tls:
  - hosts:
    - fabmedical-[SUFFIX]-ingress.[AZURE-REGION].cloudapp.azure.com
      secretName: tls-secret
    rules:
    - host: fabmedical-[SUFFIX]-ingress.[AZURE-REGION].cloudapp.azure.com
      http:
        paths:
        - path: /(.*)
          backend:
            serviceName: web
            servicePort: 80
        - path: /content-api/(.*)
          backend:
            serviceName: api
            servicePort: 3001
```

28. Сохраните изменения и закройте редактор.

29. Создайте вход с помощью kubectl.

```
kubectl create --save-config=true -f content.ingress.yml
```

30. Обновите конечную точку входа в браузере. Вы должны иметь возможность посещать страницы докладчиков и сессий и видеть весь контент.

31. Посетите API напрямую, перейдя в /content-api/sessions на входящей конечной точке.

```
{
  "_id": "54b321f979cfa6002dd73477",
  "speakerNames": [
    "Theresa Zesiewicz",
    "Kevin Allison",
    "Israt Jahan",
    "Jessica Shaw",
    "F. Reed Murtagh",
    "Tracy Jones",
    "Clifton Gooch",
    "Jason Salemi",
    "Matthew B. Klein",
    "Guy Miller",
    "Kelly Sullivan"
  ],
  "speakers": [],
  "trackNames": [
    "Visual Studio/Azure intersection",
    "ASP.NET / HTML5 intersection"
  ],
  "tracks": [
    "124",
    "125"
  ]
}
```

32. Протестируйте завершение TLS, снова посетив обе службы, используя https.

Примечание. Прежде чем сайт SSL станет доступным, может пройти от 5 до 30 минут. Это связано с задержкой, связанной с предоставлением сертификата TLS от letsencrypt.