Dokumentation M120

By Nando Sterki + Jori Villiger

Inhaltsverzeichnis

[Create service principal: 2](#_Toc129597739)

[Logs in einer Pipeline: 3](#_Toc129597740)

[.gitlab-ci.yml 4](#_Toc129597741)

[Tipps & Tricks: 4](#_Toc129597742)

[Vorlage für Namenskonzept beim Erstellen von bsw. Apps 7](#_Toc129597743)

[RBAC 8](#_Toc129597744)

[Basuc features 8](#_Toc129597745)

[Lis role assignments 9](#_Toc129597746)

[List role assignments for a resource group 9](#_Toc129597747)

[Grant access 9](#_Toc129597748)

[Remove access 9](#_Toc129597749)

[View activity logs 10](#_Toc129597750)

# Create service principal:

az ad sp create-for-rbac --scopes /subscriptions/<RESOURCE\_GROUP\_SUBSCRIPTION\_ID> --role "Owner"

Creating 'Owner' role assignment under scope '/subscriptions/965aa8a5-9a39-4f68-8908-06e3c4d82d85'

The output includes credentials that you must protect. Be sure that you do not include these credentials in your code or check the credentials into your source control. For more information, see https://aka.ms/azadsp-cli

#Username, password, someStringThatIsNeeded

#Username

eac396b0-c31e-4c92-af4c-5e2c19d132ba

#Tenant

02b7e332-b310-470b-8eef-729bfcf99667

#Password

Cr.8Q~NCijtNw5EpSwa\_SPy7IQ0~uNxESXOXjdl6

### Connect to server (Task 1)

az mysql flexible-server connect   
-n <server-name>  
-u <db-admin-user-name>  
--interactive

Create App (simpler)

az spring create \

--resource-group <resource-group-name> \

--name <app-service-name>

### Create App (Task 2)

az spring app create \

--resource-group <resource-group-name> \

--service <app-service-name> \

--name <app-name> \

--runtime-version Java\_17 \

--assign-endpoint true

Es wird eine Url ausgegeben, auf welcher später die Applikation verfügbar sein wird. Diese sollte folgender URL ähneln:

[https://studle2-studletheapp.azuremicroservices.io](https://studle2-studletheapp.azuremicroservices.io¨)

sobald die App läuft, kann unter der angepassten Url die App aufgerufen werden:

<generierte-url>/swagger-ui/index.html

### Get set up on server (Task 3)

az spring connection create mysql-flexible \

-g <resource-group-name> \

--tg <resource-group-name> \

--connection <connection\_name> \

--service <app-service-name> \

--app <app-name> \

--server <server-name> \

--database <database-name> \

--client-type springBoot \

--secret name=<db-admin-user-name> secret=<db-admin-password>

### Set everything up (Task 4)

az mysql flexible-server parameter set \

--resource-group <resource-group-name> \

--server-name <mysql-server-name> \

--name require\_secure\_transport \

--value OFF

### Deploy App (jar File) (Task 5)

az spring app deploy \

--resource-group <resource-group-name> \

--service <app-service-name> \

--name <app-name> \

--artifact-path <path-to-jar-file>.jar

### Delete App (when youre done)

az spring app delete --name <app-name> \  
 -g <resource-group-name> \  
 -s <app-service-name>

az spring delete --name <app-service-name> -g <resource-group-name>

### View logs:

az spring app logs -g <resource-group-name> -s <app-service-name> -n <app-name>

# Logs in einer Pipeline:

CI/CD > Jobs > Status:passed

# .gitlab-ci.yml

default:

image: alpine:3

job1:

stage: build

script:

- echo "Executed on all branches"

job2:

stage: build

script:

- echo "Executed only on the production branch"

rules:

- if: $CI\_MERGE\_REQUEST\_SOURCE\_BRANCH\_NAME == /main/

when: on\_success

- if: $CI\_MERGE\_REQUEST\_SOURCE\_BRANCH\_NAME == /test1/

when: never

job3:

stage: build

script:

- echo "Executed on the branches main and staging"

rules:

- if: $CI\_MERGE\_REQUEST\_SOURCE\_BRANCH\_NAME == /main/

when: on\_success

- if: $CI\_MERGE\_REQUEST\_SOURCE\_BRANCH\_NAME == /staging/

when: on\_success

- if: $CI\_MERGE\_REQUEST\_SOURCE\_BRANCH\_NAME == /test1/

when: never

## Tipps & Tricks:

* Keine Tabs erlaubt => 2 Leerschläge == 1 Tab
* $CI\_MERGE\_REQUEST\_SOURCE\_BRANCH\_NAME == aktueller Branchname
* When-Attribute :
  + on\_success

default, wenn alle Jobs erfolgreich durchlaufen

* + manual

nur wenn Job manuell getriggert wird

* + always

immer, Status vom Job egal (auch wenn er failed)

* + on\_failure

nur wenn min. ein Job failt

* + delayed

verzögert um eine angegebene Zeit

* + never

niemals

* equals (==)
* not equals (!=)
* and (&&)
* or (||)

Das funktioniert nur mit absolut leeren Reps:

cd existing\_folder

git init --initial-branch=main

git remote add origin PFAD\_ZU\_DEINEM\_GITLAB\_REPO

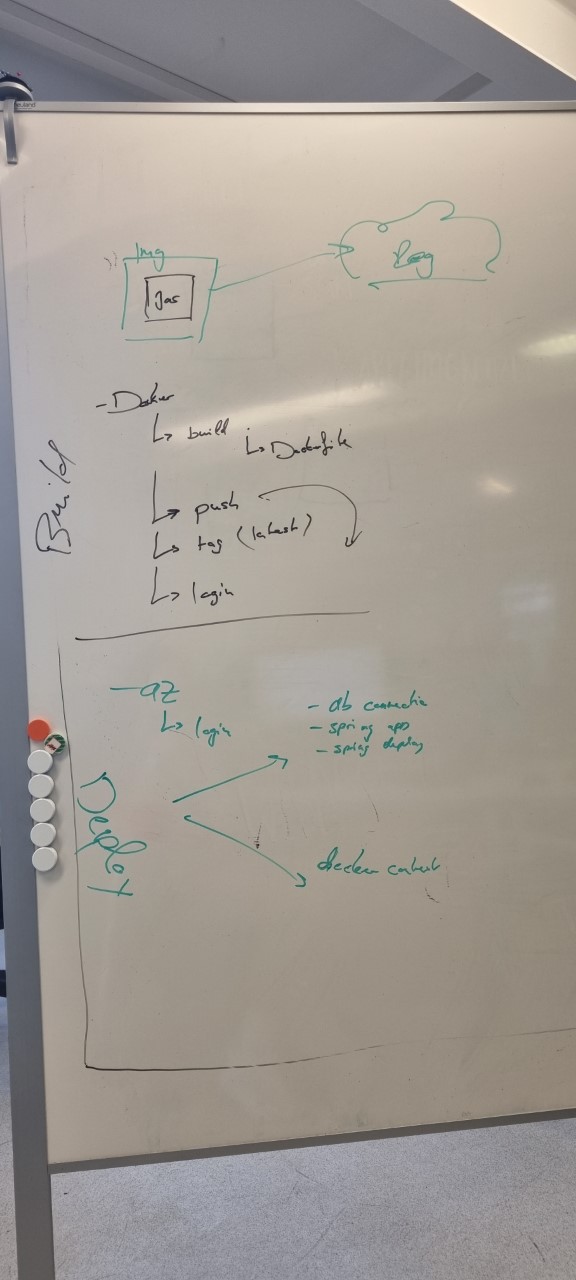
git add .

git commit -m "Initial commit"

git push -u origin main

für möglichst fehlerfreies Ergebnis:

die remote branch sollte im GUI noch nicht existieren (nur bei first commits)



Azure Functions Core Tool installieren:

wget -q [https://packages.microsoft.com/config/ubuntu/20.04/packages-microsoft-prod.deb sudo dpkg -i packages-microsoft-prod.deb](https://packages.microsoft.com/config/ubuntu/20.04/packages-microsoft-prod.deb%20sudo%20dpkg%20-i%20packages-microsoft-prod.deb)

# Vorlage für Namenskonzept beim Erstellen von bsw. Apps

<resource-group-name> Resource group-Name

<app-service-name> App-Service Name (Spring Applikation)

<app-name> App Name

<path-to-jar-file> Jar-File

<connection\_name> Connection-/Netzwerk-Name

<server-name> Server-Name

<database-name> Datenbank-Name

<db-admin-user-name> DB-Admin

<db-admin-password> DB-Admin-Password

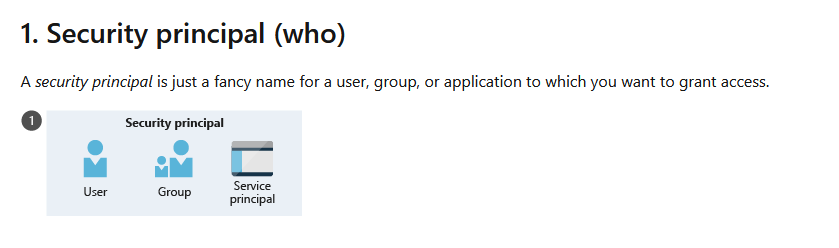
<mysql-server-name> MySQL-Name (Erstellt in Task 1)

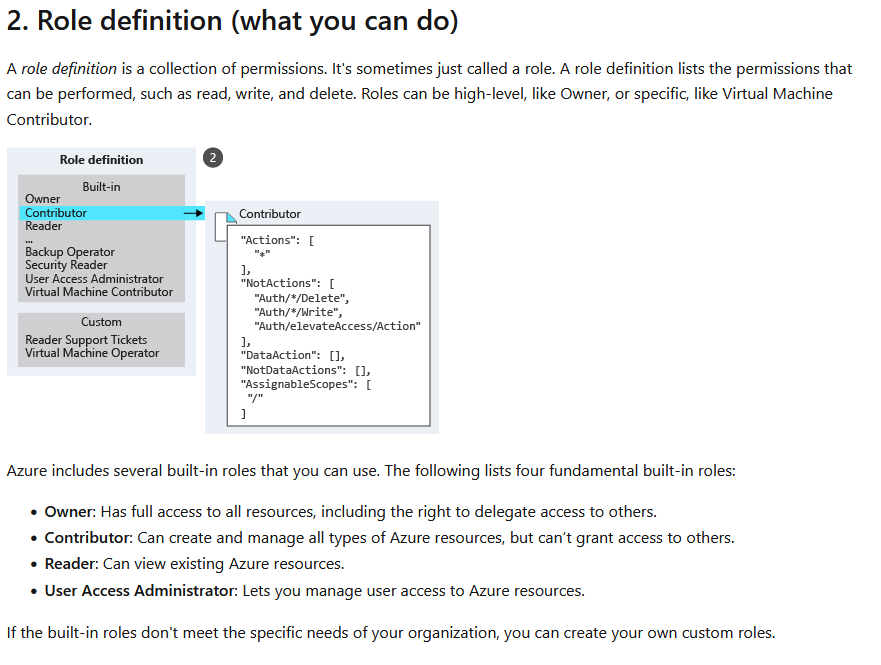
# RBAC

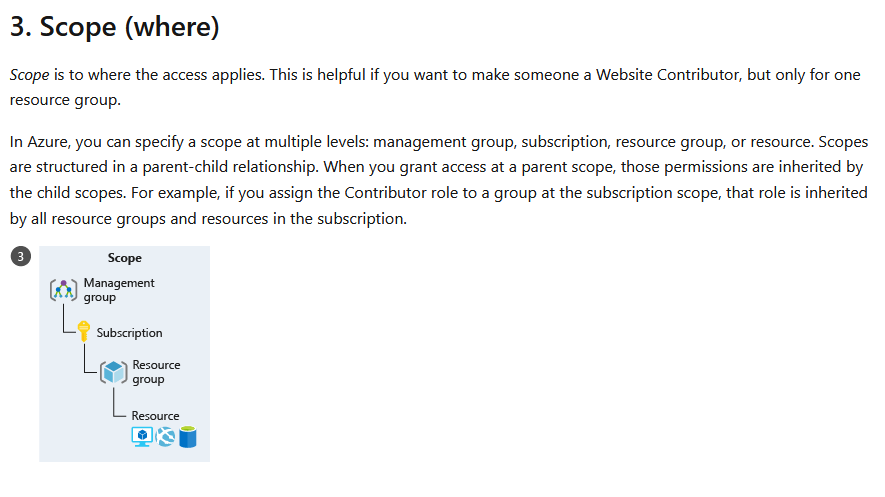
Azure role-based access control (Azure RBAC) is an authorization system that provides fine-grained access management of resources.

## Basuc features

* Allow one user to manage virtual machines in a subscription and another user to manage virtual networks
* Allow a database administrator group to manage SQL databases in a subscription
* Allow a user to manage all resources in a resource group, such as virtual machines, websites, and subnets
* Allow an application to access all resources in a resource group







## Lis role assignments

Profile -> (…) -> My permissions

## List role assignments for a resource group

Resource groups -> example-group -> Access control (IAM)-> role assignments

# Grant access

Resource group -> example-group -> Access control (IAM) -> Role assignments Add -> Add role assignment

# Remove access

Select user -> Remove -> Yes

# View activity logs

All services -> Activity log -> Timespan filter, operation filter