title

Subtitle

Approval for use

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| delaware |  | Name | Signature | Date |
| Setup by: |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
| Approved by: |  |  |  |  |
|  |  |  |  |  |

Distribution

|  |  |
| --- | --- |
| (e.g. name customer) | (name contact person) |
|  |  |
|  |  |
|  |  |

Data classification

|  |  |  |
| --- | --- | --- |
| Choose a classification of confidentiality | | |
|  | **Public** | An information asset is classified as public if the information is purposely aimed to be disclosed to the public and has no business impact for delaware if that information is disclosed. |
|  | **Internal Use** | An information asset is classified as internal use if the disclosure of the information to unauthorized individuals could only cause limited or no business impact to delaware. |
|  | **Confidential** | An information asset is classified confidential if the disclosure of the information to unauthorized individuals could cause a moderate business impact to delaware. |
|  | **Strictly Confidential** | An information asset is classified strictly confidential if the disclosure of the information to unauthorized individuals could cause a severe business impact to delaware. |

Change history

|  |  |  |
| --- | --- | --- |
| Version | Date | Description |
|  |  |  |
|  |  |  |
|  |  |  |

Approval for use

|  |  |  |  |
| --- | --- | --- | --- |
| delaware | Name | Signature | Date |
| Setup by: |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
| Approved by: |  |  |  |
|  |  |  |  |
|  |  |  |  |

Distribution

|  |  |
| --- | --- |
| (e.g. name customer) | (name contact person) |
|  |  |
|  |  |
|  |  |
|  |  |

Data classification

|  |  |  |
| --- | --- | --- |
| Choose a classification of confidentiality | | |
|  | **Public** | An information asset is classified as public if the information is purposely aimed to be disclosed to the public and has no business impact for delaware if that information is disclosed. |
|  | **Internal Use** | An information asset is classified as internal use if the disclosure of the information to unauthorized individuals could only cause limited or no business impact to delaware. |
|  | **Confidential** | An information asset is classified confidential if the disclosure of the information to unauthorized individuals could cause a moderate business impact to delaware. |
|  | **Strictly Confidential** | An information asset is classified strictly confidential if the disclosure of the information to unauthorized individuals could cause a severe business impact to delaware. |

Change history

|  |  |  |
| --- | --- | --- |
| Version | Date | Description |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

Approval for use

|  |  |  |  |
| --- | --- | --- | --- |
| delaware | Name | Signature | Date |
| Setup by: |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
| Approved by: |  |  |  |
|  |  |  |  |
|  |  |  |  |

Distribution

|  |  |
| --- | --- |
| (e.g. name customer) | (name contact person) |
|  |  |
|  |  |
|  |  |
|  |  |

Data classification

|  |  |  |
| --- | --- | --- |
| Choose a classification of confidentiality | | |
|  | **Public** | An information asset is classified as public if the information is purposely aimed to be disclosed to the public and has no business impact for delaware if that information is disclosed. |
|  | **Internal Use** | An information asset is classified as internal use if the disclosure of the information to unauthorized individuals could only cause limited or no business impact to delaware. |
|  | **Confidential** | An information asset is classified confidential if the disclosure of the information to unauthorized individuals could cause a moderate business impact to delaware. |
|  | **Strictly Confidential** | An information asset is classified strictly confidential if the disclosure of the information to unauthorized individuals could cause a severe business impact to delaware. |

Change history

|  |  |  |
| --- | --- | --- |
| Version | Date | Description |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

Contents

[1 Azure services 4](#_Toc1449429301)

[1.1 Resource group 5](#_Toc74822882)

[1.2 Static web app 5](#_Toc595587992)

[1.3 SQL Database + server 5](#_Toc779006760)

[1.4 Storage account 6](#_Toc150101577)

[1.5 App service 6](#_Toc873962186)

[1.6 Registration 7](#_Toc1047049829)

[2 Microsoft graph 11](#_Toc817313589)

[2.1 Extensions 11](#_Toc1202087188)

[3 Repository 12](#_Toc598908518)

[3.1 Info 12](#_Toc173325785)

[3.2 Frontend 14](#_Toc965867156)

[3.3 Backend API 15](#_Toc214551449)

[3.4 Backend web jobs 15](#_Toc1009876988)

# Azure services

Go to <https://portal.azure.com/#home>

## Resource group

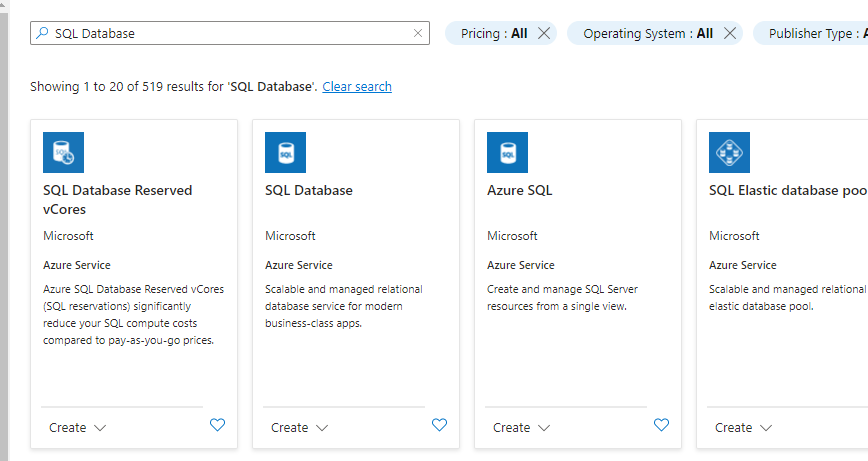
Create a resource group and call it [company]-OOOScheduler-[developer]-[resource] in our case we make it “dw-OOOScheduler-devjj-rg” here we add all the other resources.

Then inside the resource group add the following:

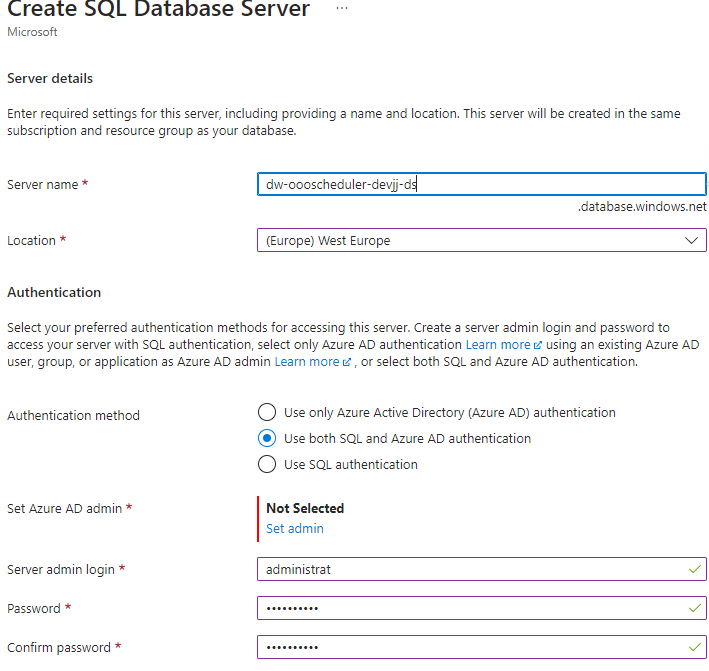
## Static web app

Create a Static web app inside the Resource group and name it “dw-OOOScheduler-devjj-swa”

## SQL Database + server



Name the data base “dw-OOOScheduler-devjj-db”

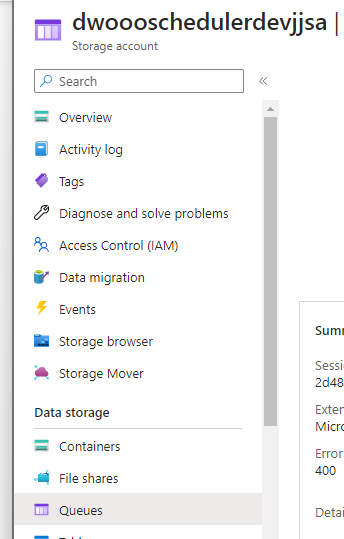


Set an admin and a login. The login is the most important, this is used to connect the apps to the database

In the data base select a proper payment plan.

## Storage account

Create a storage account then go to the storage. Then go to the queue tab and click create queue. Name this queue ”auto-reply-queue“

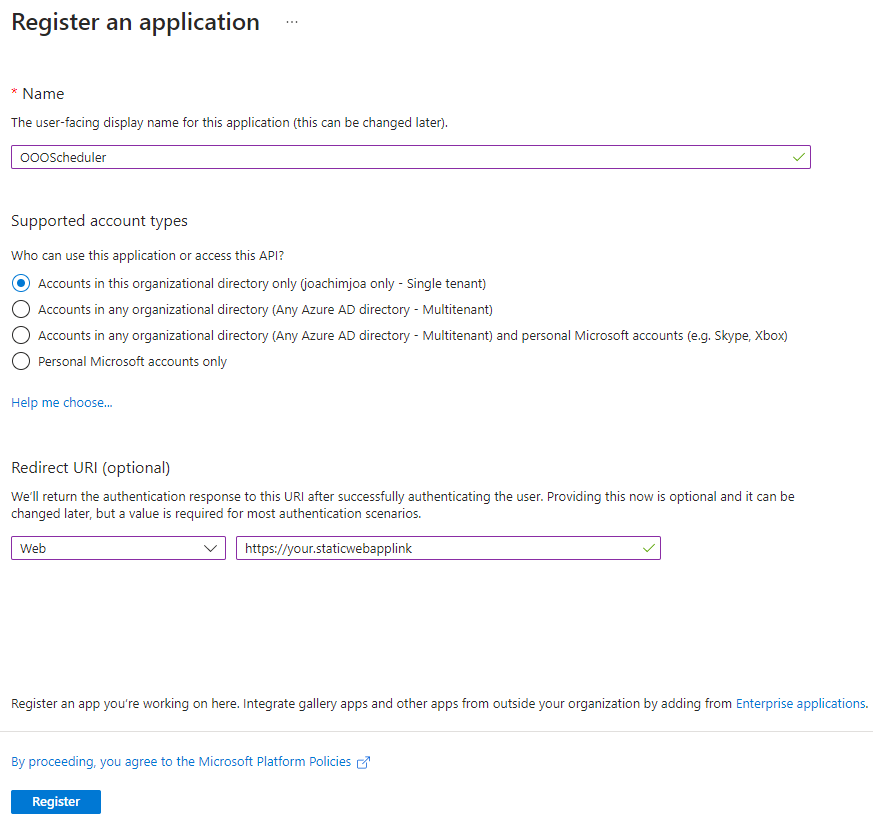


## App service

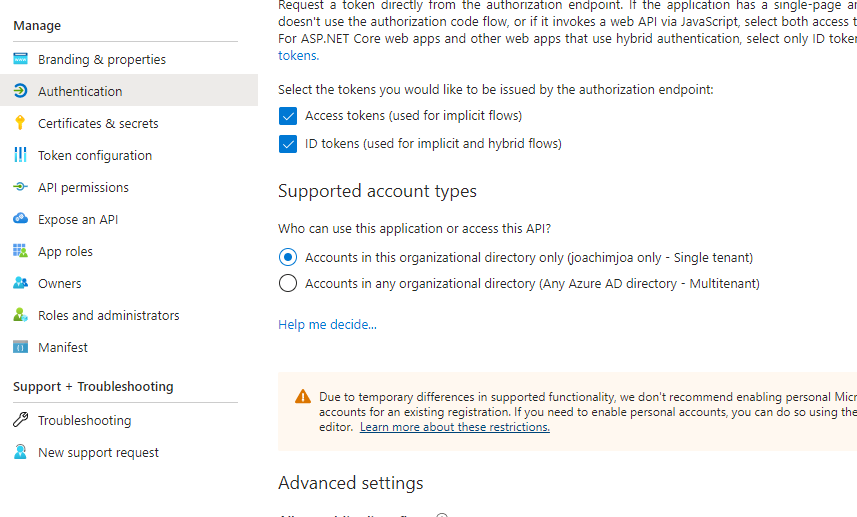
Create an app service with the name “dw-OOOScheduler-devjj-as”

## Registration

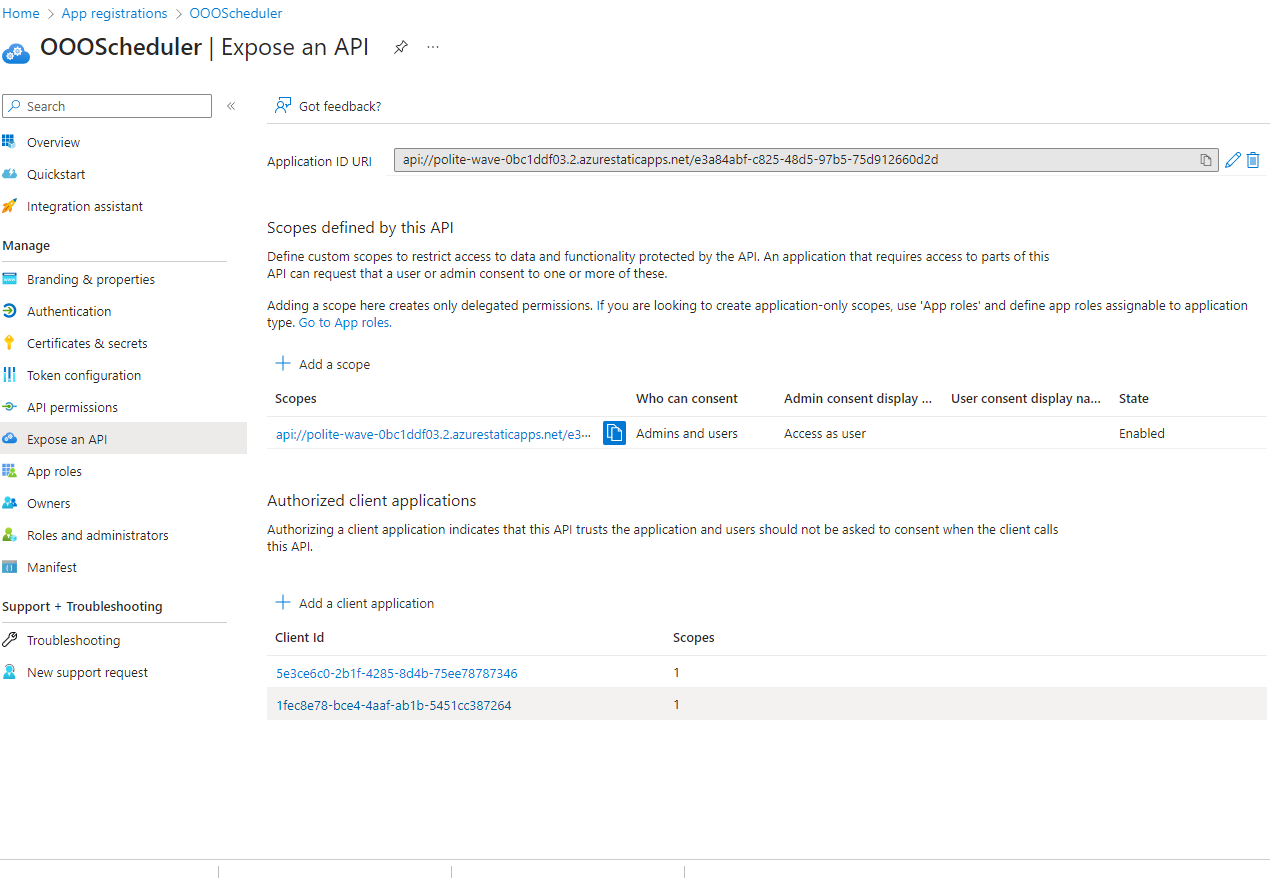
Go to the [app registration tab on the azure portal](https://portal.azure.com/#view/Microsoft_AAD_RegisteredApps/ApplicationsListBlade) and add a new registration

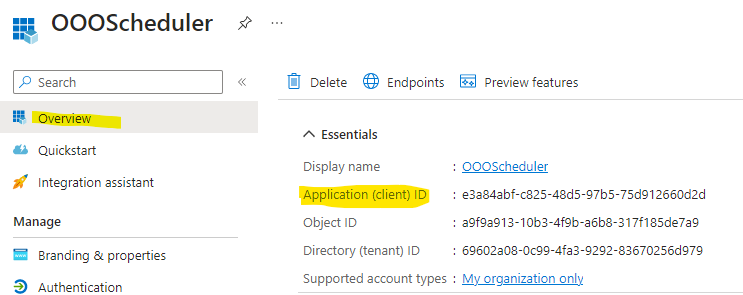


Go to manage -> Authentication and make sure the settings are as shown below:

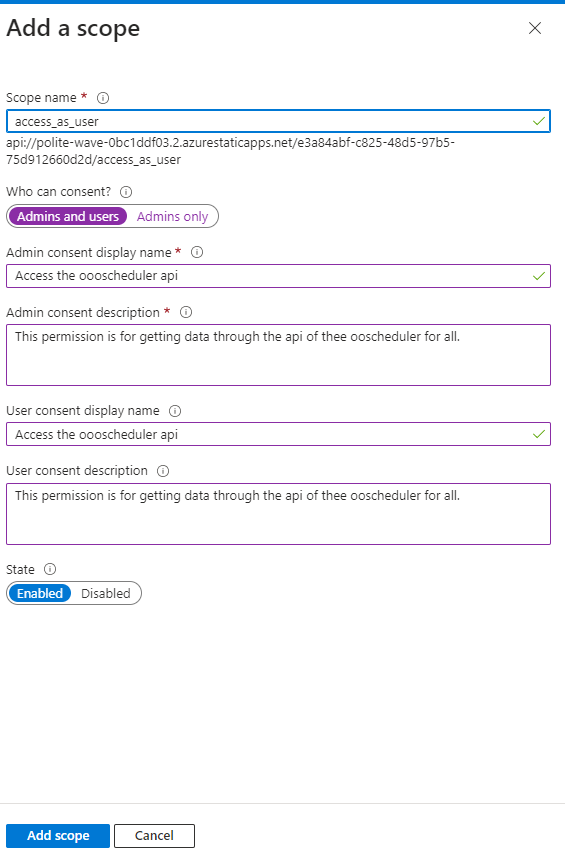


Now go to the “expose an API” tab and edit the application ID URI to “api://{static web app link}/{application Id}”





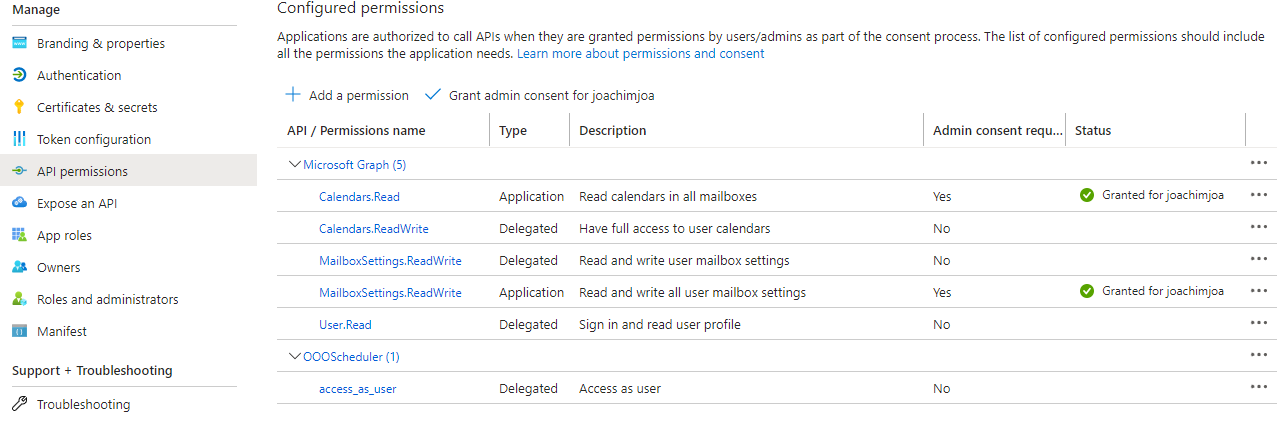
Make sure to add the access\_as\_user scope



Here also add the client id’s

* “5e3ce6c0-2b1f-4285-8d4b-75ee78787346” the ID use by the Microsoft Teams web application
* “1fec8e78-bce4-4aaf-ab1b-5451cc387264” the ID used by the Microsoft Teams desktop and mobile application

Finally add the permissions required



If you want the users to give consent themselves, I suggest adding the application permissions first then grant admin consent and then add the rests because this option grants permission for all permissions

# Microsoft graph

## Extensions

Go to <https://developer.microsoft.com/en-us/graph/graph-explorer> and click the “get” method. Then select post. And fill in the url to <https://graph.microsoft.com/v1.0/schemaExtensions> and use the following data as the body. Don’t forget to fill in the client id from the registration portal.

{

    "id": "oooschedulerMessage",

    "description": "Message definition for events. these will be set to the users Automatic reply message",

    "name": "messageExtention",

    "targetTypes": [

        "Event"

    ],

    "owner": {client id},

    "properties": [

        {

            "name": "messageId",

            "type": "String"

        },

        {

            "name": "message",

            "type": "String"

        },

        {

            "name": "messageType",

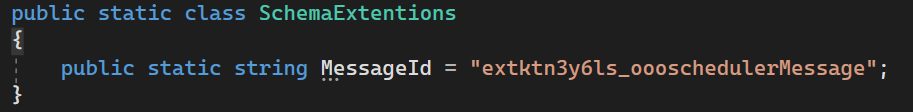
            "type": "String"

        }

    ]

}

Send the request and look at the response. Make sure it has a respond code of 200. Look at the response ID and copy it. Go to the Global class and paste it in the SchemaExtensions class as the MessageId. It should look like this where the “extktn3y6ls” will be different



Then also add this inside the prod.env file:

VITE\_MESSAGE\_EXTENTION\_ID="extktn3y6ls\_oooschedulerMessage"

# Repository

## Info

Make sure you have git installed on your computer. Then execute the command `git clone https://dlwr-dlwr.visualstudio.com/DLWR.Internship%20Engage/\_git/Internship%20-%20Joachim%20Jacobs` this will install the repository on your system in this folder you will need to make some changes

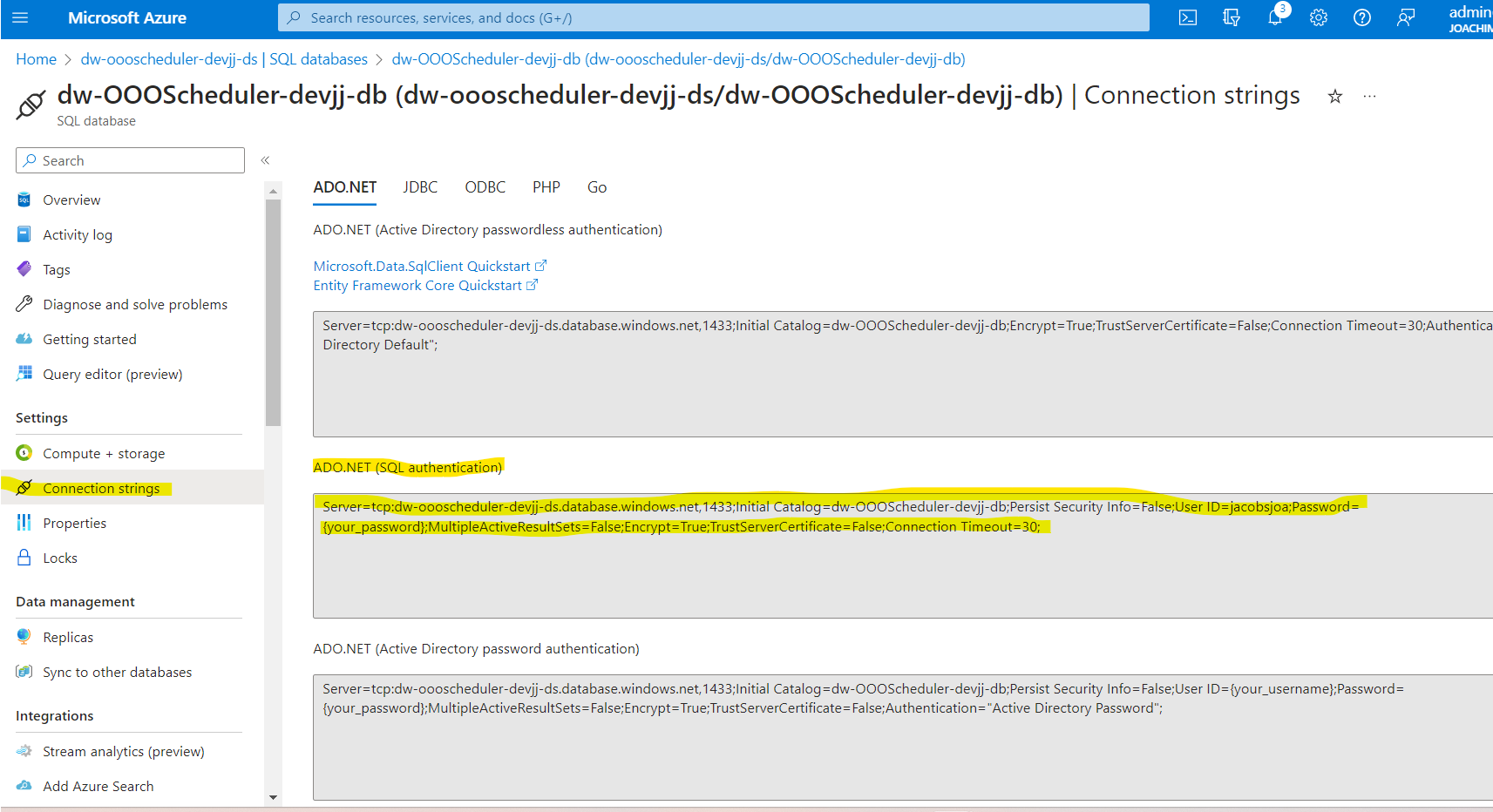
Here is some extra information about where to find certain variables you’ll need.

### Database connection string

Go to the db resource on the azure portal then go to the settings-> connection strings

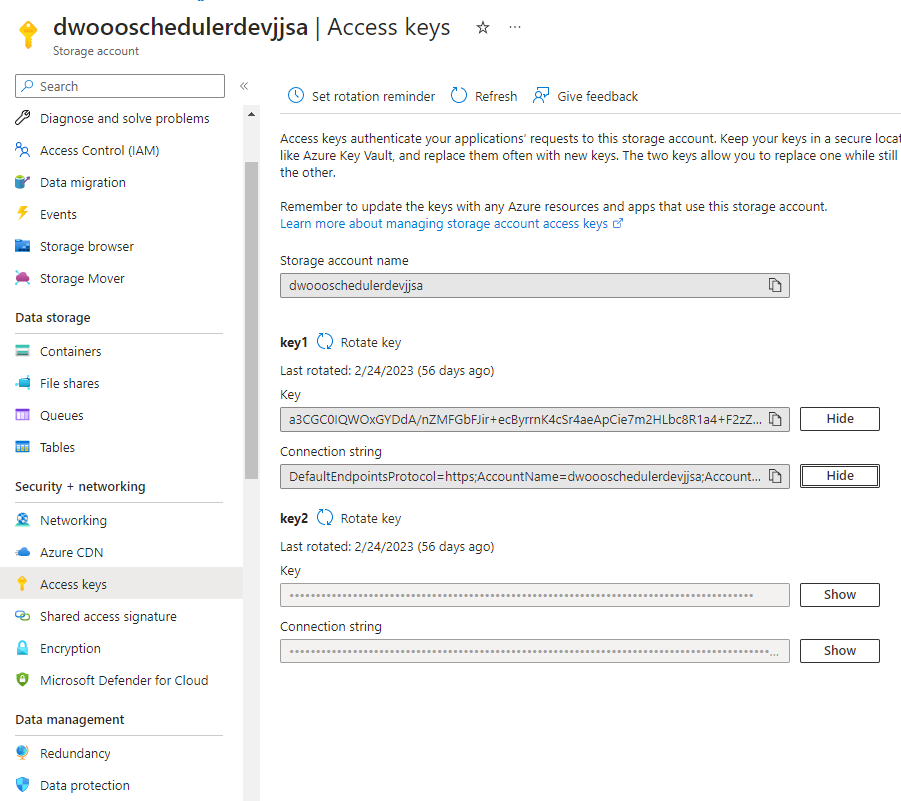
Then in the ADO.NET part copy the second string (ADO.NET SQL authentication)

**Don’t forget to change the password to the password you set**



### Storage account connection string

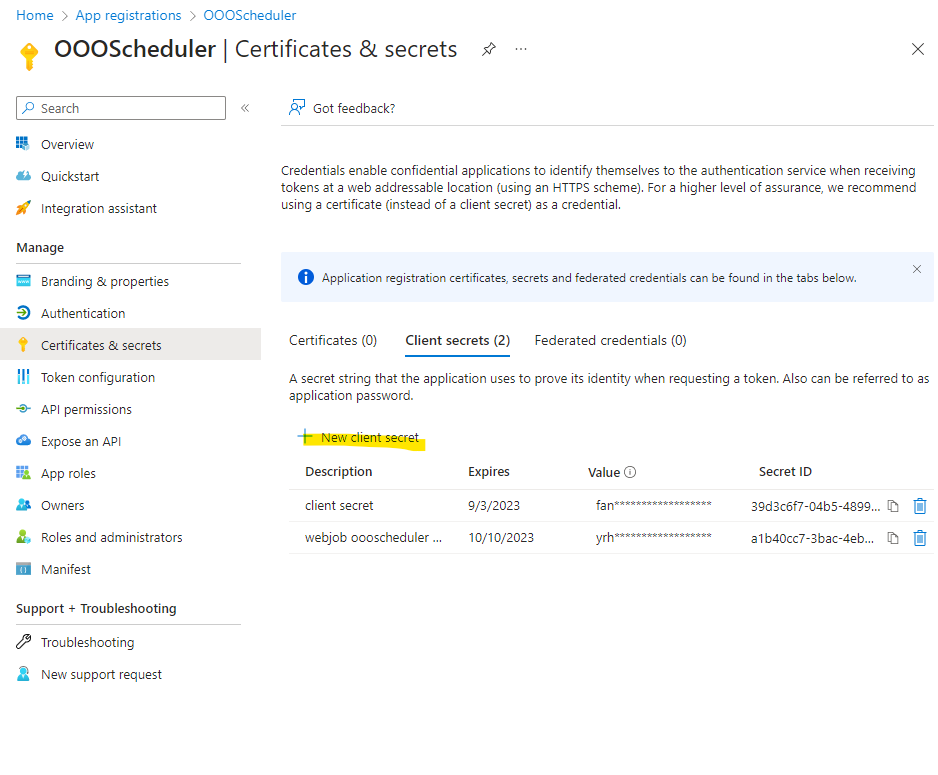
Go to the storage account in the resource group and then go to security+networking -> access keys



Click on show connection string and paste it in the corresponding value in the appsettings.json files

### Client secret

To get the client secret you need to create a new one. In the app registration go to the certificates and secrets. Then click the new client secret button.



Enter the description which is either the webjob secret or the api secret (client secret)

When created copy the value and paste this in the corresponding value in the appsettings see web jobs and API.

**You cannot copy the value after leaving the page so make sure you copy it and paste it before leaving the page.**

## Frontend

In the frontend folder there is a “.env.production” file. In that file we need to set some variables so the react app can function. Replace the data inside the {}

VITE\_API\_URI="{the link of your app service (this is the api)}"

VITE\_REDIRECT\_URI="{the link for your static web app}"

VITE\_CLIENT\_ID="{client id from app registration overview}"

VITE\_TENANT\_ID="{tenant id from app registration overview}"

VITE\_SCOPES="{Go to the expose an API tab and copy the scope access\_as\_user scopes (it should look like: api://polite-wave-0bc1ddf03.2.azurestaticapps.net/{clientId}/access\_as\_user)}"

VITE\_MESSAGE\_EXTENTION\_ID="{see graph -> extensions}"

## Backend API

{

  "AzureWebJobsStorage": "{storage account connection string (look at info)}",

  "AzureAd": {

    "scopes": "access\_as\_user",

    "Instance": "https://login.microsoftonline.com/",

    "TenantId": “{tenant id from app registration overview}",

    "ClientId": "{client id from app registration overview}",

    "ClientSecret": "{Client secrets}"

  },

  "MsGraph": {

    "BaseUrl": "https://graph.microsoft.com/v1.0/",

    "Scopes": "User.Read Calendars.ReadWrite MailboxSettings.ReadWrite"

  },

  "Logging": {

    "LogLevel": {

      "Default": "Trace",

      "Microsoft.AspNetCore": "Warning"

    }

  },

  "AllowedHosts": "\*"

}

## Backend web jobs

{

    "TenantId": “{tenant id from app registration overview}",

    "ClientId": “{client id from app registration overview}",

    "ClientSecret": "{(look at info)}}"

  "dlwrConnectionString": "{database connection string (look at info)}”

  "AzureWebJobsStorage": "{storage account connection string (look at info)}

  "exclude": [

    "\*\*/bin",

    "\*\*/bower\_components",

    "\*\*/jspm\_packages",

    "\*\*/node\_modules",

    "\*\*/obj",

    "\*\*/platforms"

  ]

}

# Teams’ config

## Manifest

{

    "$schema": "https://developer.microsoft.com/en-us/json-schemas/teams/v1.16/MicrosoftTeams.schema.json",

    "version": "1.0.2",

    "manifestVersion": "1.16",

    "id": "87bf41ad-73a5-489b-a682-9f5bb9cbf91b",

    "packageName": "com.package.name",

    "name": {

        "short": "Out-of-office scheduler",

        "full": "Out of office scheduler - delaware"

    },

    "developer": {

        "name": "delaware",

        "mpnId": "",

        "websiteUrl": "https://polite-wave-0bc1ddf03.2.azurestaticapps.net/web",

        "privacyUrl": "https://polite-wave-0bc1ddf03.2.azurestaticapps.net/pvc",

        "termsOfUseUrl": "https://polite-wave-0bc1ddf03.2.azurestaticapps.net/tos"

    },

    "description": {

        "short": "Schedule time for when you are out of office and let it turn on the auto reply",

        "full": "Schedule time for when you are out of office. It will turn on the automatic reply function of outlook for when you're not in the office. Schedule once and forget about it"

    },

    "icons": {

        "outline": "outline.png",

        "color": "color.png"

    },

    "accentColor": "#FFFFFF",

    "staticTabs": [

        {

            "entityId": "72bfe4e9-8d3f-47ee-8a2a-5bc1bf516f91",

            "name": "OOOScheduler",

            "contentUrl": "{your static web app link}",

            "scopes": [

                "personal"

            ],

            "context": [

                "personalTab"

            ]

        },

        {

            "entityId": "about",

            "scopes": [

                "personal"

            ]

        }

    ],

    "validDomains": [],

    "webApplicationInfo": {

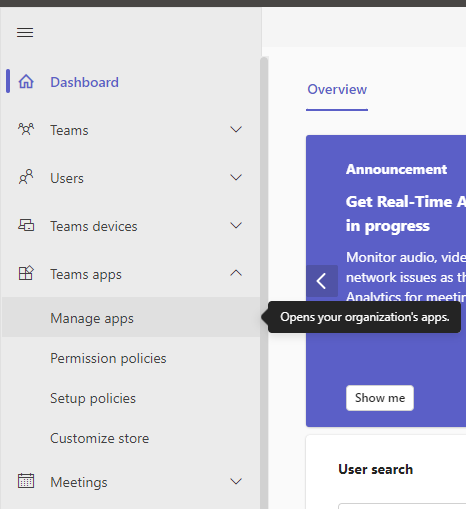
        "id": "{client id}",

        "resource": "{api://{static web app link}/{application Id}(can be found in the registration section}"

    }

}

## Add to tenant

Go to <https://admin.teams.microsoft.com/dashboard>. And go to Teams apps -> manage apps

Before moving on make a .zip file from the documents inside the teams folder .Here find the upload buttonang

