Lesson3 - Create Secret

In Lesson2 we deployed MSSQL in Kuberneties with **password in a plain text** which is **insecure**. The source files were checked-in to github source control so everyone can see our database password.

The SA password is placed in **mssql-deployment.yml**:

```
resources:
  limits:
    memory: 3Gi
    cpu: 1
ports:
- containerPort: 1433
env:
- name: ACCEPT_EULA
  value: "Y"
- name: SA_PASSWORD
  value: "MyDemoPwd2021!"
- name: MSSQL_AGENT_ENABLED
  value: "true"
```

The connection string is placed in **appsettings.json** with password in plain text as well:

```
    "Microsoft": "Warning"
  }
},
"AllowedHosts": "*",
"MyKey": "MW3: MyKey value from appsettings.json",

"ConnectionStrings": {
  "ConnectionString": "server=localhost;Initial Catalog=EmployeeDB;Persist Security Info=False;User ID=sa;Password=MyDemoPwd2021!;MultipleActive
}
```

Solution - Create Kuberneties Secret

In order to use Kuberneties Secret for MSSQL we need to delete the namespace **employee** so all components that were deployed within the namespace will be deleted as well. This is because the secret must be installed in the namespace before deploying the MSSQL.

kubectl delete ns employee - delete the entire employee namespace.

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL

PS C:\employee\employeemanagement\Employees> kubectl delete ns employee
namespace "employee" deleted
PS C:\employee\employeemanagement\Employees> kubectl get all -n employee
No resources found in employee namespace.
PS C:\employee\employeemanagement\Employees>
```

kubectl get all -n employee – verify that the namespace was deleted.

Run the commands below to create employee namespace:

kubectl create namespace employee

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL

PS C:\Kuberneties\EmployeesManagement> kubectl create namespace employee
namespace/employee created
PS C:\Kuberneties\EmployeesManagement>
```

Run the command below to create secret with two keys: **ConnectionString** and **SA_PASSWORD:**

kubectl create secret generic mssql-secret --namespace=employee --from-literal='ConnectionString="server=mssql-service;Initial Catalog=EmployeeDB;Persist Security Info=False;User ID=sa;Password=MyDemoPwd2021!;MultipleActiveResultSets=true"' --from-literal='SA_PASSWORD=MyDemoPwd2021!'

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL                    1: powershell

PS C:\Kuberneties\EmployeesManagement> kubectl create secret generic mssql-secret --namespace=employee --from-literal='ConnectionString="server=mssq
l-service;Initial Catalog=EmployeeDB;Persist Security Info=False;User ID=sa;Password=1qaz!QAZ;MultipleActiveResultSets=true"' --from-literal='SA_PAS
SWORD=1qaz!QAZ'
secret/mssql-secret created
PS C:\Kuberneties\EmployeesManagement>
```

kubectl get secret mssql-secret -n employee

```
PS C:\Kuberneties\EmployeesManagement> kubectl get secret mssql-secret -n employee
NAME           TYPE     DATA   AGE
mssql-secret   string   1      3m1s
PS C:\Kuberneties\EmployeesManagement>
```

kubectl describe secret mssql-secret -n employee

```
PS C:\Kuberneties\EmployeesManagement> kubectl describe secret mssql-secret -n employee
Name:         mssql-secret
Namespace:    employee
Labels:       <none>
Annotations:  <none>

Type:  Opaque

Data
====
ConnectionString:  134 bytes
SA_PASSWORD:       8 bytes
PS C:\Kuberneties\EmployeesManagement>
```

kubectl get secret mssql-secret -n employee –oyaml

```
PS C:\Kuberneties\EmployeesManagement> kubectl get secret mssql-secret -n employee -oyaml
apiVersion: v1
data:
  ConnectionString: c2VydmVyPW1zc3FsLXNlcnZpY2U7SW5pdGlhbCBDYXRhbG9nPUVtcGxveWVlREI7UGVyc2lzdCB
MXFheiFRQVo7TXVsdGlwbGVBY3RpdmVSZXN1bHRTZXRzPXRydWU=
  SA_PASSWORD: MXFheiFRQVo=
kind: Secret
metadata:
```

We can see in the screenshot above that both keys are encrypted.

cd .\manifests

kubectl apply -f .\mssql-deploy-with-secret.yml (we deploy different YML file for the secret)

```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL

PS C:\Kuberneties\EmployeesManagement\manifests> kubectl apply -f .\mssql-deploy-with-secret.yml
deployment.apps/mssql-deployment created
service/mssql-service created
PS C:\Kuberneties\EmployeesManagement\manifests>
```

kubectl get all -n employee

```
PS C:\Kuberneties\EmployeesManagement\manifests> kubectl apply -f .\mssql-deploy-with-secret.yml
deployment.apps/mssql-deployment created
service/mssql-service created
PS C:\Kuberneties\EmployeesManagement\manifests> kubectl get all -n employee
NAME                                    READY   STATUS    RESTARTS   AGE
pod/mssql-deployment-6bfb754db5-sn9zg   1/1     Running   0          91s

NAME                    TYPE           CLUSTER-IP      EXTERNAL-IP   PORT(S)          AGE
service/mssql-service   LoadBalancer   10.104.160.19   localhost     1433:31375/TCP   91s
PS C:\Kuberneties\EmployeesManagement\manifests> kubectl get all -n employee
NAME                                    READY   STATUS    RESTARTS   AGE
pod/mssql-deployment-6bfb754db5-sn9zg   1/1     Running   0          103s
```

The different between **mssql-deploy-with-secret.yml** that we are using in this lesson and **mssql-deployment.yml** that we used in lesson-1 is that in this case we are getting encrypted password from Kuberneties secret named **mssql-secret** and not from plain text.

```yaml
env:
- name: ACCEPT_EULA
  value: "Y"
- name: SA_PASSWORD
  valueFrom:
    secretKeyRef:
      name: mssql-secret
      key: SA_PASSWORD
- name: MSSQL_AGENT_ENABLED
  value: "true"
```

```yaml
env:
- name: ACCEPT_EULA
  value: "Y"
- name: SA_PASSWORD
  value: "MyDemoPwd2021!"
- name: MSSQL_AGENT_ENABLED
  value: "true"
```

cd ..

cd Employees

dotnet ef database update - creates **EmployeeDB** database in our MSSQL server with initial data.

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL

PS C:\employee\employeemanagement\manifests> cd ..
PS C:\employee\employeemanagement> cd .\Employees\
PS C:\employee\employeemanagement\Employees> dotnet ef database update
Build started...
Build succeeded.
Done.
PS C:\employee\employeemanagement\Employees>
```

Now the connection string in **appsettings.json** will be overridden by Kuberneties secret.

```
  "ConnectionStrings": {
    "ConnectionString": "From Kuberneties secret and Windows Environment Variable"
  }
}
```

dotnet build & **dotnet run**

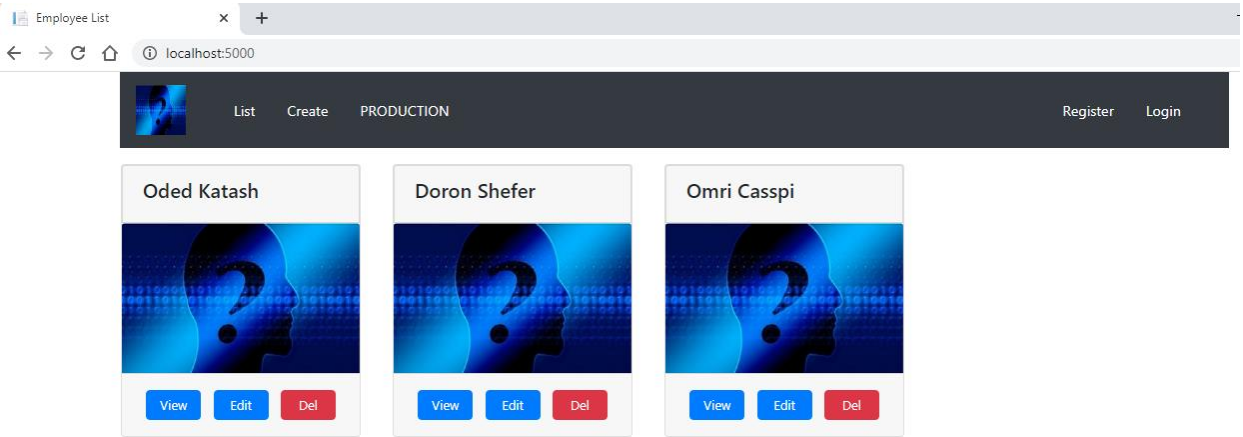
```
Try the new cross-platform PowerShell https://aka.ms/pscore6
PS C:\employee> cd .\employeemanagement\
PS C:\employee\employeemanagement> cd .\Employees\
PS C:\employee\employeemanagement\Employees> dotnet build
Microsoft (R) Build Engine version 16.7.2+b60ddb6f4 for .NET
Copyright (C) Microsoft Corporation. All rights reserved.

  Determining projects to restore...
  Restored C:\employee\employeemanagement\Employees\Employees.csproj (in 1.56 sec).
  Employees -> C:\employee\employeemanagement\Employees\bin\Debug\netcoreapp3.1\Employees.dll
  Employees -> C:\employee\employeemanagement\Employees\bin\Debug\netcoreapp3.1\Employees.Views.dll

Build succeeded.
    0 Warning(s)
    0 Error(s)

Time Elapsed 00:00:30.76
PS C:\employee\employeemanagement\Employees> dotnet run
```

Open Google Chrome or any Other browser and enter the URL: http://localhost:5000

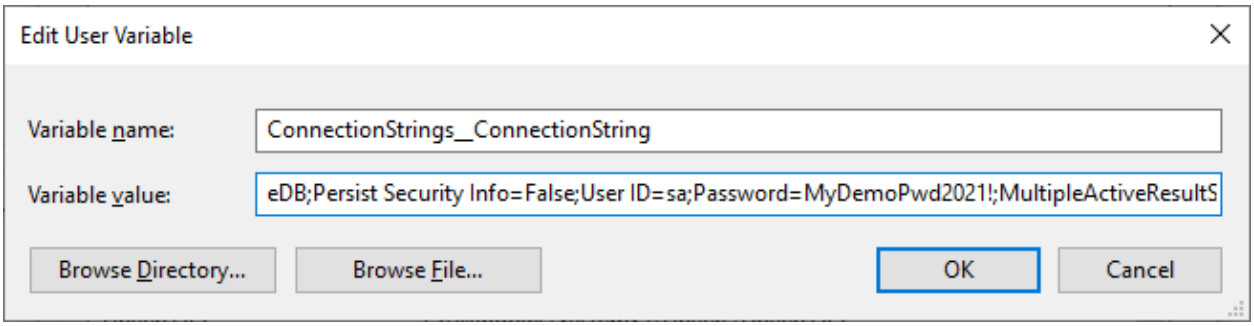


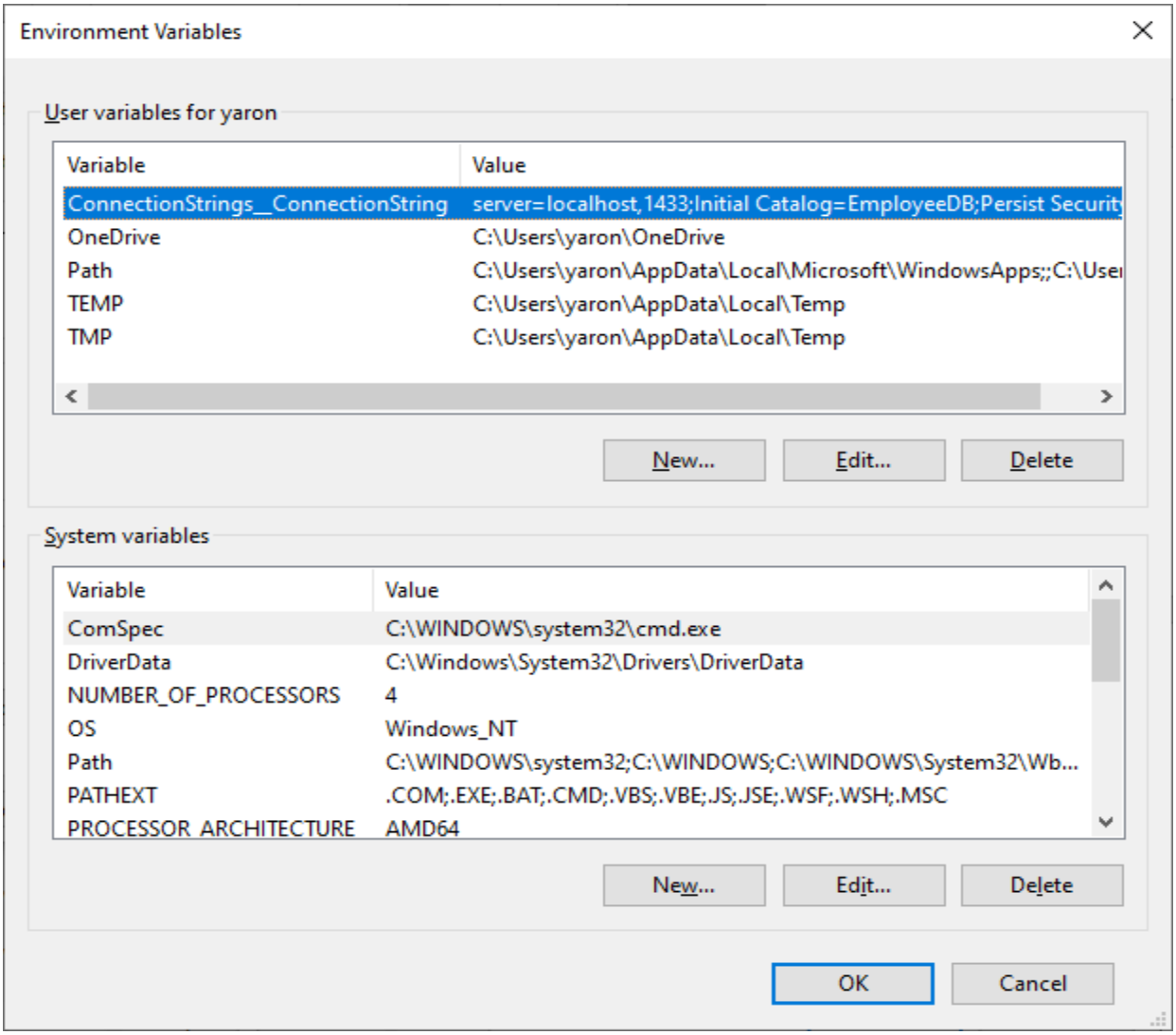
Use **CTRL+C** in the terminal to exit the application

Done!

Other option for developers

Create **Windows Environment Variable** on the developer's PC.





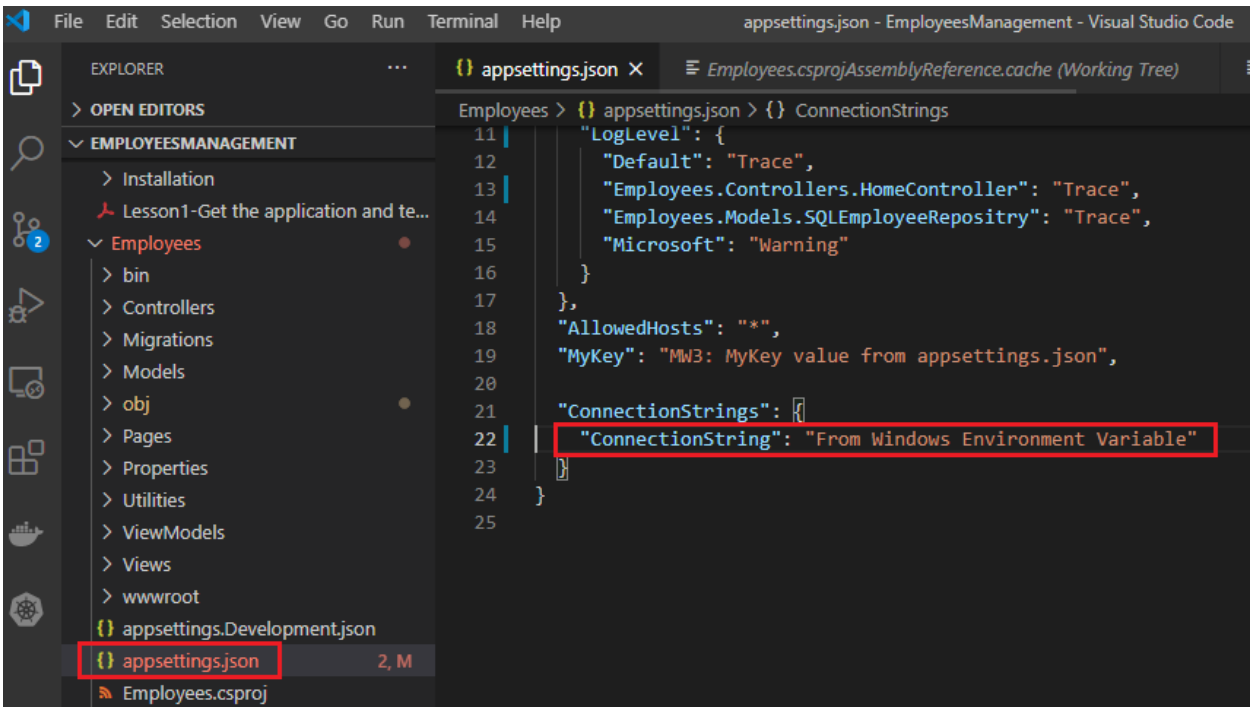
Use the key and the value below:

Variable name: ConnectionStrings__ConnectionString

Variable value: server=localhost,1433;Initial Catalog=EmployeeDB;Persist Security Info=False;User ID=sa;Password=MyDemoPwd2021!;MultipleActiveResultSets=true

*When done you need to close VS code and open it again so the environment variable will affect the application.

In **appsettings.json,** remove the connection string and write something else, the connection string in the environment variable will override the connection string in the appsettings.json**.**



Run a test and verify that the connection string is working from the environment variable:

dotnet build & **dotnet run**

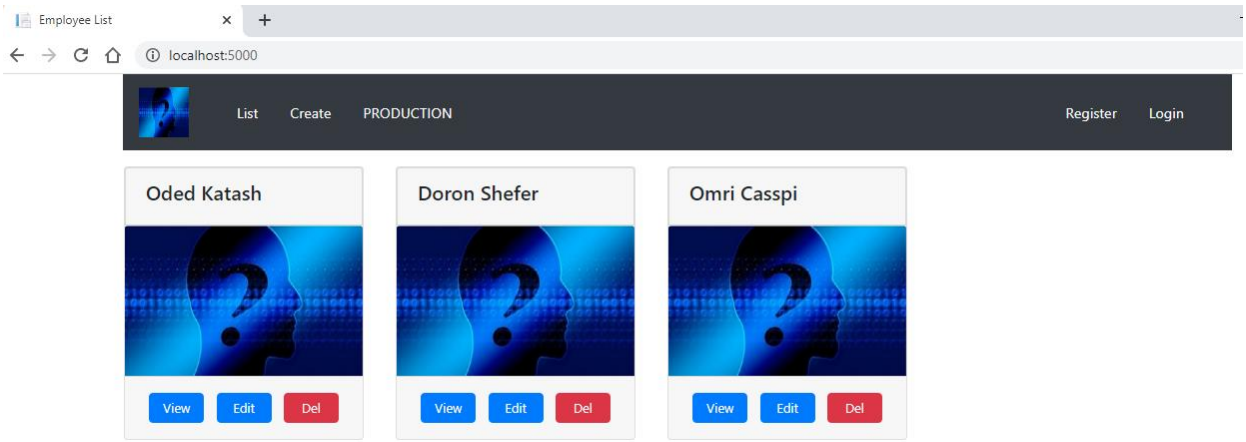
```
Try the new cross-platform PowerShell https://aka.ms/pscore6
PS C:\employee> cd .\employeemanagement\
PS C:\employee\employeemanagement> cd .\Employees\
PS C:\employee\employeemanagement\Employees> dotnet build
Microsoft (R) Build Engine version 16.7.2+b60ddb6f4 for .NET
Copyright (C) Microsoft Corporation. All rights reserved.

  Determining projects to restore...
  Restored C:\employee\employeemanagement\Employees\Employees.csproj (in 1.56 sec).
  Employees -> C:\employee\employeemanagement\Employees\bin\Debug\netcoreapp3.1\Employees.dll
  Employees -> C:\employee\employeemanagement\Employees\bin\Debug\netcoreapp3.1\Employees.Views.dll

Build succeeded.
    0 Warning(s)
    0 Error(s)

Time Elapsed 00:00:30.76
PS C:\employee\employeemanagement\Employees> dotnet run
```

Open Google Chrome or any Other browser and enter the URL: http://localhost:5000



Use **CTRL+C** in the terminal to exit the application

Done!