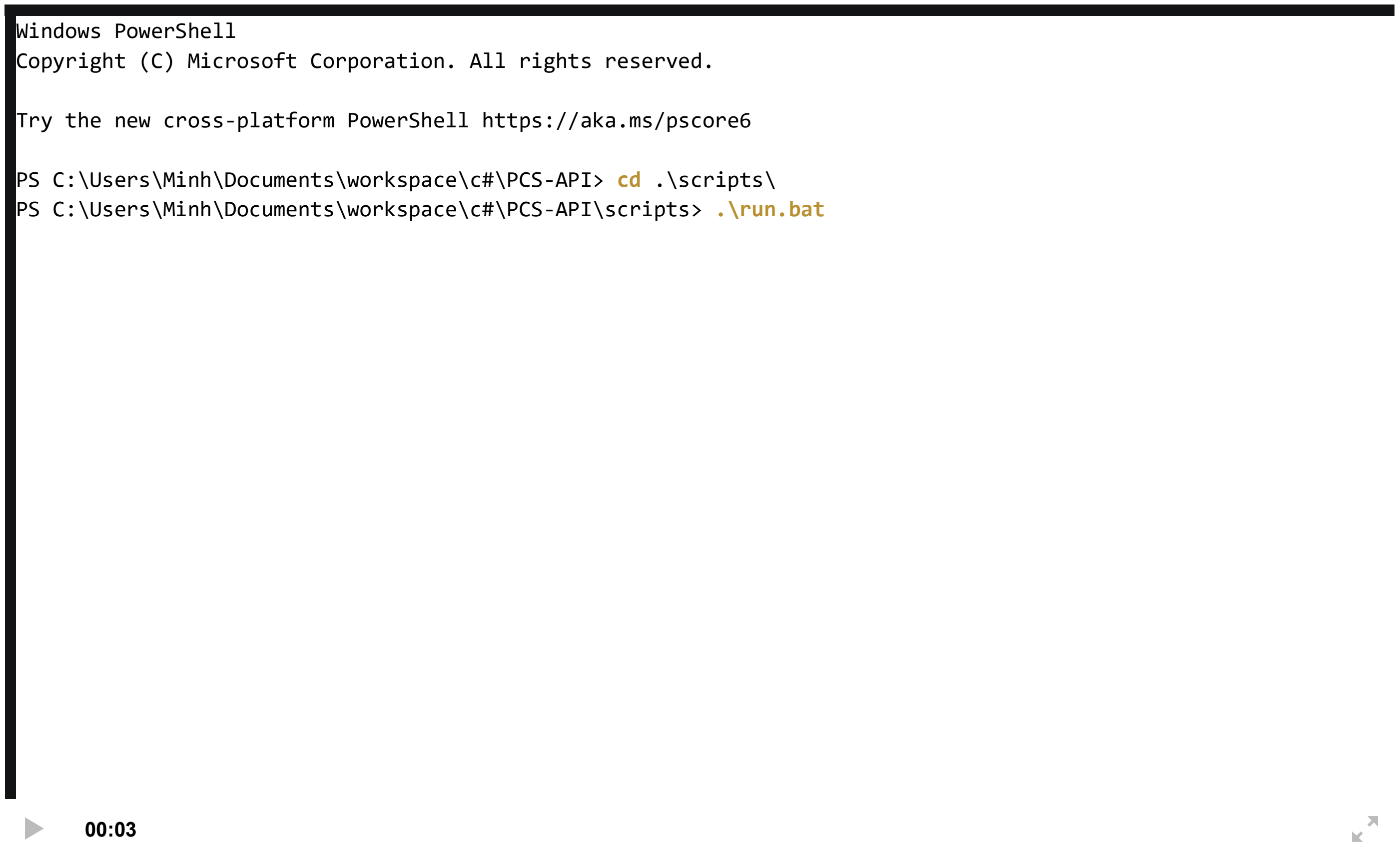
Demo dockerized development environment

pdf



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\Minh\Documents\workspace\c#\PCS-API> cd .\scripts\
PS C:\Users\Minh\Documents\workspace\c#\PCS-API\scripts> .\run.bat
```

Table of Contents

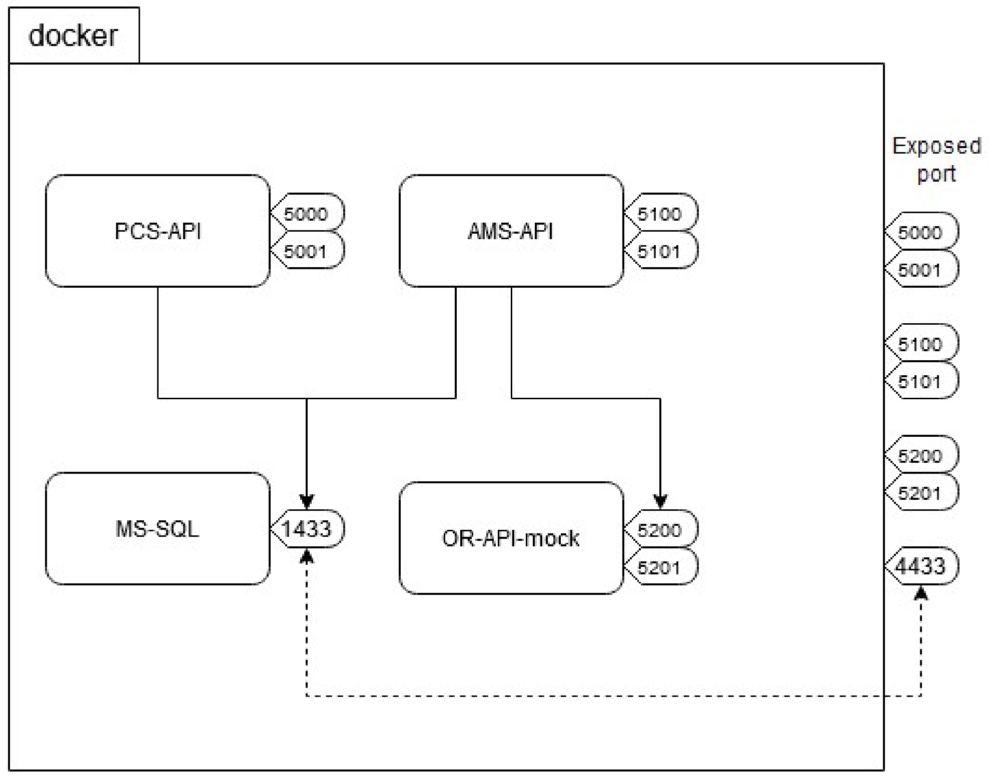
Quickstart

back

1. clone OR-API-mock repo
2. run build.bat script in cloned OR-API-mock folder
3. clone PCS-API repo
4. run script/clean.bat to remove and clean old containers
5. run script/run.bat to build and rundified to run on simple cli tool without visual studio

Overall architecture of the services

back



MSSQL - SQL database

Mounting volume

- .\services\sqlserver -> /opt/script/init_mssql
- .\src\Rosen.Data\Resource\SqlScripts -> /opt/script/sql

Port mapping

- 1433 (internal) -> 4433 (external)

Expressing connection between services

back

By default Compose sets up a single network for your app.

Each container for a service joins the default network and is both reachable by other containers on that network, and discoverable by them at a hostname identical to the container name.

Example appsettings:

```
"ConnectionStrings": {
  "Portal": "Server=mssql,1433\\Catalog=ROSEN_PCS;Database=ROSEN_PCS;User Id=SA;Password=
  "AMS": "Server=mssql,1433\\Catalog=ROSEN_AMS;Database=ROSEN_AMS;User Id=SA;Password=You
},
"ApiUri": {
  "Portal": "",
  "ReportingAgent": "http://or_api_mock:5200/"
},
"ReportingAgent": {
  "tokenEndpoint": "http://or_api_mock:5200/assets/token"
}
```

Other tidbit

- SQL connection string
- Create a https dev-cert

back

SQL connection string:

mssql is the name of the MSSQL service's name defined in the docker-compose.yml

1433 is the port configured for the MSSQL service in the docker-compose.override.yml

```
"ConnectionStrings": {
  "Portal": "Server=mssql,1433\\Catalog=ROSEN_PCS;Database=ROSEN_PCS;User Id=SA;Passwor
  "AMS": "Server=mssql,1433\\Catalog=ROSEN_AMS;Database=ROSEN_AMS;User Id=SA;Password=Y
}
```

Create a dev-certs using dev-certs tool and use it for the API service's https

generate a dev-cert

```
dotnet dev-certs https --clean
dotnet dev-certs https -ep %APPDATA%\ASP.NET\Https\aspnetapp.pfx -p "password"
dotnet dev-certs https --trust
```

DEMO Time!

QA Time!

The End