

# **Przetwarzanie w chmurach obliczeniowych**

Projekt zaliczeniowy

Patryk Kuszneruk, IV rok Informatyki, spec. ISI  
Wydział Nauk Ścisłych i Technicznych  
Uniwersytet Śląski

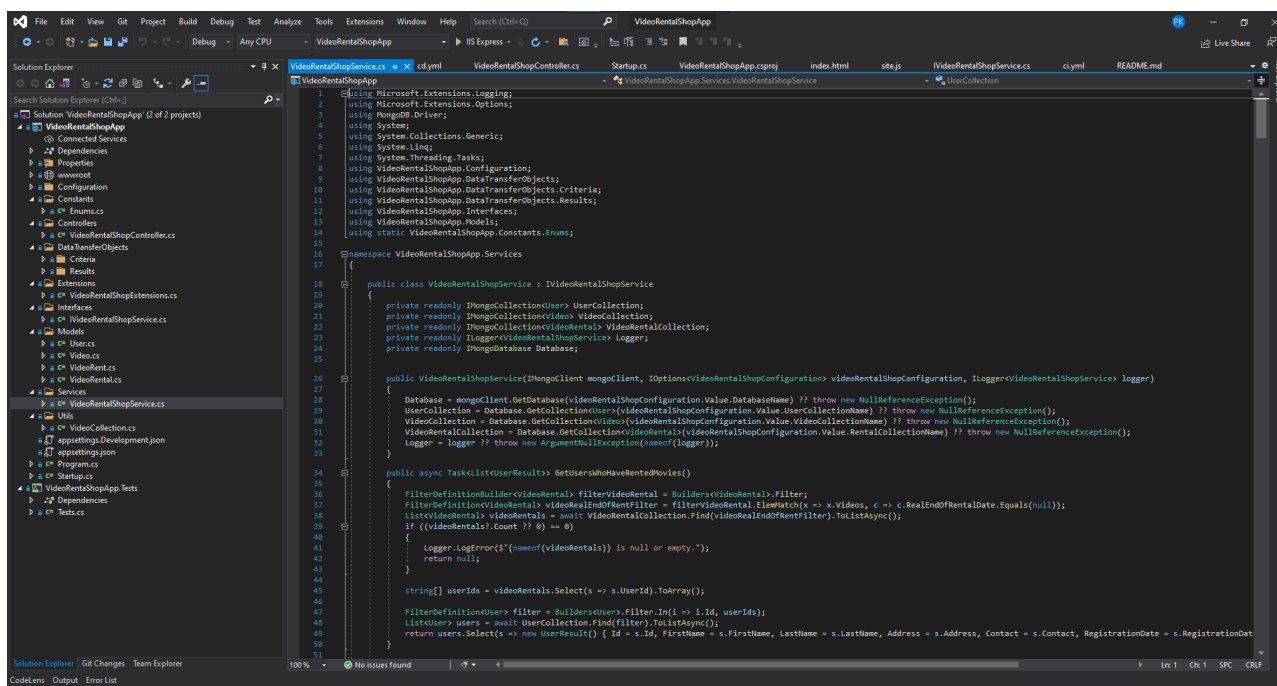
## 1. Opis

Przedmiotem dokumentacji jest aplikacja webowa ze skonfigurowanym potokiem CI/CD na platformie GitHub z wykorzystaniem GitHub Actions, która została wdrożona na chmurze Azure. Aplikacja jest typu CRUD, korzysta z nierelacyjnej bazy danych MongoDB, która została utworzona na Azure z wykorzystaniem Azure Cosmos DB for MongoDB account. Zawiera ona skonfigurowane testy automatyczne oraz korzysta z protokołu HTTPS.

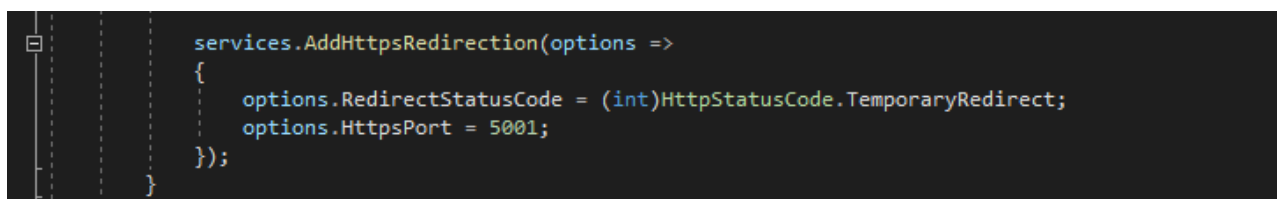
## 2. Przebieg stworzenia i wdrożenia aplikacji

### 2.1. Aplikacja

#### Kod źródłowy



## Przekierowanie na HTTPS



## Proste testy automatyczne

```
namespace VideoRentaShopApp.Tests
{
    0 references | pkusznruk, 10 hours ago | 1 author, 1 change
    public class Tests
    {
        [Fact]
        0 references | pkusznruk, 10 hours ago | 1 author, 1 change
        public async Task GetVideosAsync_NotEmpty_True()
        {
            using WebApplicationFactory<VideoRentalShopApp.Startup> app = new WebApplicationFactory<VideoRentalShopApp.Startup>();
            using System.Net.Http.HttpClient client = app.CreateClient();

            System.Net.Http.HttpResponseMessage response = await client.GetAsync("/VideoRentalShop/GetVideos");

            response.Content.Should().NotBeNull();
        }

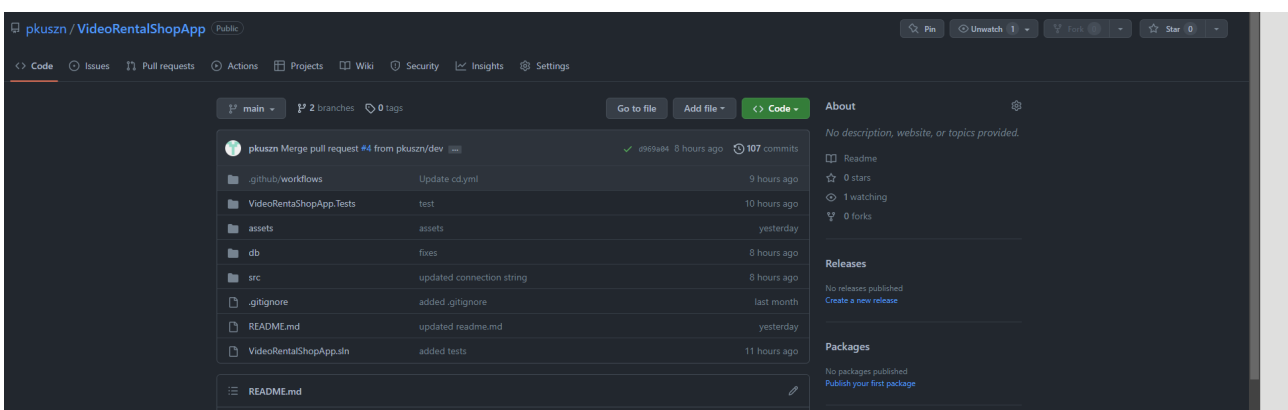
        [Fact]
        0 references | 0 changes | 0 authors, 0 changes
        public async Task GetUsersAsync_NotEmpty_True()
        {
            using WebApplicationFactory<VideoRentalShopApp.Startup> app = new WebApplicationFactory<VideoRentalShopApp.Startup>();
            using System.Net.Http.HttpClient client = app.CreateClient();

            System.Net.Http.HttpResponseMessage response = await client.GetAsync("/VideoRentalShop/GetUsers");

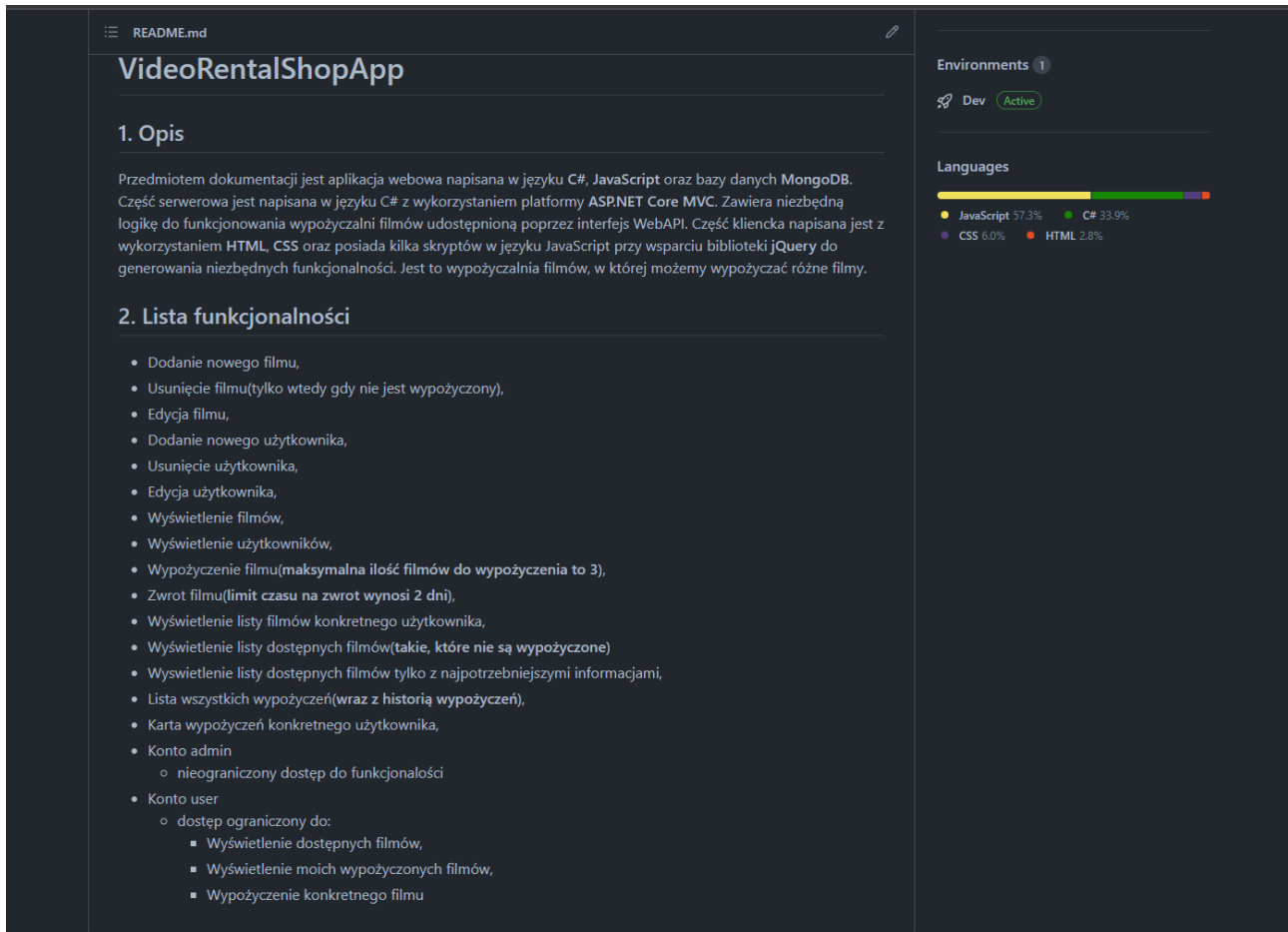
            response.Content.Should().NotBeNull();
        }
    }
}
```

## 2.2 Repozytorium

Aplikacja jest śledzona przy wykorzystaniu repozytorium Git, wysyłana jest na zdalny serwer GitHub



Posiada plik README.md z opisem aplikacji



**VideoRentalShopApp**

## 1. Opis

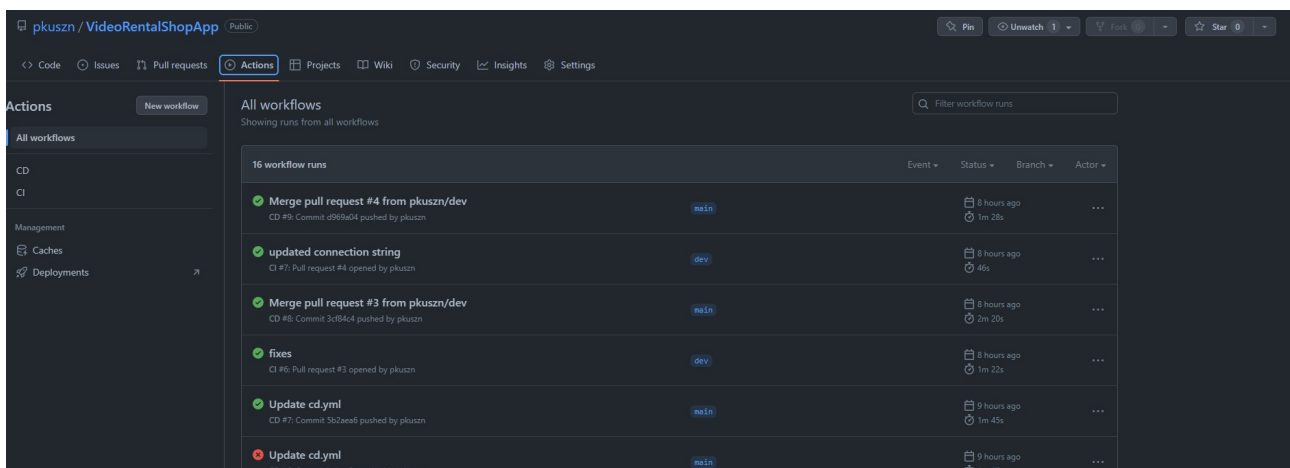
Przedmiotem dokumentacji jest aplikacja webowa napisana w języku C#, JavaScript oraz bazy danych MongoDB. Część serwerowa jest napisana w języku C# z wykorzystaniem platformy ASP.NET Core MVC. Zawiera niezbędną logikę do funkcjonowania wypożyczalni filmów udostępnioną poprzez interfejs WebAPI. Część kliencka napisana jest z wykorzystaniem HTML, CSS oraz posiada kilka skryptów w języku JavaScript przy wsparciu biblioteki jQuery do generowania niezbędnych funkcjonalności. Jest to wypożyczalnia filmów, w której możemy wypożyczać różne filmy.

## 2. Lista funkcjonalności

- Dodanie nowego filmu,
- Usunięcie filmu(tylko wtedy gdy nie jest wypożyczony),
- Edycja filmu,
- Dodanie nowego użytkownika,
- Usunięcie użytkownika,
- Edycja użytkownika,
- Wyświetlenie filmów,
- Wyświetlenie użytkowników,
- Wypożyczenie filmu(maksymalna ilość filmów do wypożyczenia to 3),
- Zwrot filmu(limit czasu na zwrot wynosi 2 dni),
- Wyświetlenie listy filmów konkretnego użytkownika,
- Wyświetlenie listy dostępnych filmów(takie, które nie są wypożyczone)
- Wyswietlenie listy dostępnych filmów tylko z najpotrzebniejszymi informacjami,
- Lista wszystkich wypożyczeń(wraz z historią wypożyczeń),
- Karta wypożyczeń konkretnego użytkownika,
- Konto admin
  - nieograniczony dostęp do funkcjonalności
- Konto user
  - dostęp ograniczony do:
    - Wyświetlenie dostępnych filmów,
    - Wyświetlenie moich wypożyczonych filmów,
    - Wypożyczenie konkretnego filmu

## 2.3 Potok CI/CD

Stworzony potok wykorzystuje GitHub Actions



pkuszn / VideoRentalShopApp

Actions

All workflows

Showing runs from all workflows

Event	Status	Branch	Actor
Merge pull request #4 from pkuszn/dev	Success	main	pkuszn
updated connection string	Success	dev	pkuszn
Merge pull request #3 from pkuszn/dev	Success	main	pkuszn
fixes	Success	dev	pkuszn
Update cd.yml	Success	main	pkuszn
Update cd.yml	Failure	main	pkuszn

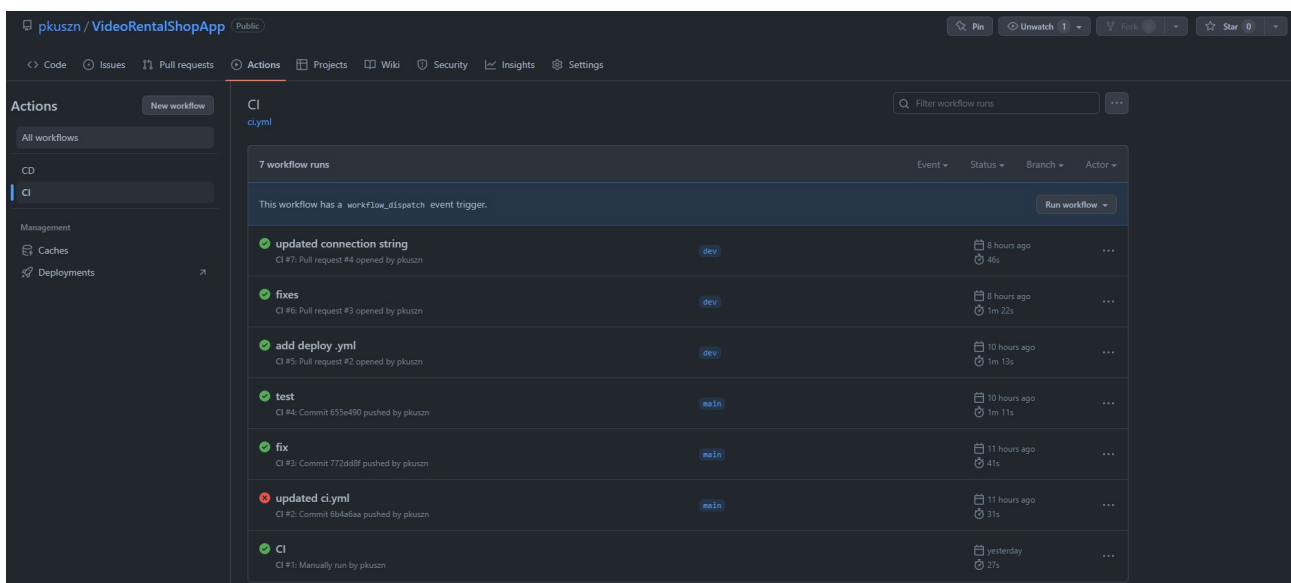
## 2.3.1 Continuous Integration

Do stworzenia potoku CI stworzony został plik z rozszerzeniem .yaml

```
github > workflows > ! ci.yml
1  name: CI
2  on:
3    pull_request:
4      branches: [ main ]
5    workflow_dispatch:
6
7  jobs:
8    build:
9      runs-on: ubuntu-latest
10     steps:
11       - uses: actions/checkout@v2
12       - name: Setup .NET Core SDK
13         uses: actions/setup-dotnet@v1.9.0
14         with:
15           dotnet-version: 5.0.x
16       - name: Restore dependencies
17         run: dotnet restore
18       - name: Build
19         run: dotnet build --no-restore
20       - name: Test
21         run: dotnet test
22
```

Kroki.

- instalacja .NET SDK,
- przywrócenie wymaganych zależności,
- zbudowanie solucji,
- uruchomienie testów



Potok jest uruchamiany w trakcie każdego pull requestu.

### 2.3.2 Continuous Delivery

Do stworzenia potoku CD stworzony został plik .yml

```
! cd.yml x {} appsettings.json <> index.html {} video.json 9+ ! ci.yml
.github > workflows > ! cd.yml
1  name: CD
2  on:
3    push:
4      branches: [ main ]
5
6  jobs:
7    build:
8      runs-on: ubuntu-latest
9      steps:
10       - uses: actions/checkout@v2
11       - name: Setup .NET Core SDK
12         uses: actions/setup-dotnet@v1.9.0
13         with:
14           dotnet-version: 5.0.x
15       - name: Restore dependencies
16         run: dotnet restore
17       - name: Build
18         run: dotnet build --no-restore
19       - name: Test
20         run: dotnet test
21       - name: Publish
22         run: dotnet publish ./src/VideoRentalShopApp.csproj -c Release -o ${ env.DOTNET_ROOT }}/api
23       - name: upload artifact
24         uses: actions/upload-artifact@v2.2.4
25         with:
26           name: api-artifact
27           path: ${ env.DOTNET_ROOT }}/api
28
29    deploy-dev:
30      runs-on: ubuntu-latest
31      needs: build
32      environment:
33        name: 'Dev'
34        url: ${ steps.deploy-to-azure.outputs.webapp-url }
35      steps:
36       - name: Download a Build Artifact
37         uses: actions/download-artifact@v2.0.10
38         with:
39           name: api-artifact
40       - name: Azure WebApp deploy
41         id: deploy-to-azure
42         uses: Azure/webapps-deploy@v2
43         with:
44           app-name: 'VideoRentalShopApp'
45           publish-profile: ${ secrets.PUBLISH_PROFILE_DEV }
46
47
```

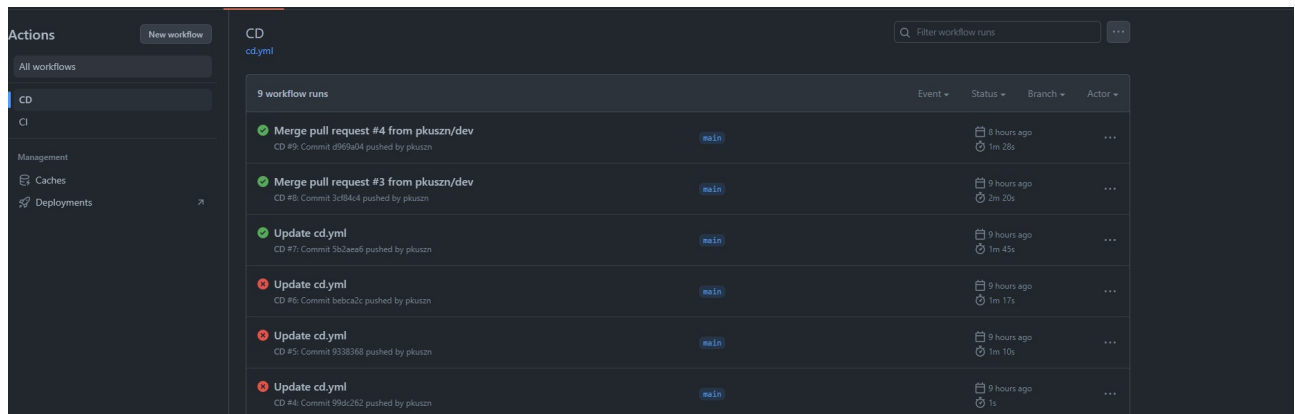
kroki build)

- a) w.w jak w CI,
- b) opublikowanie
- c) upload artefaktu przy użyciu skryptu

kroki deploy-dev)

- a) do wdrożenia aplikacji musi zostać wykonany job 'build',
- b) download artefaktu przy użyciu skryptu,
- c) wdrożenie na chmurze Azure

Potok CD uruchamiany jest przy każdym wypchnięciu zmian na branch ‘main’

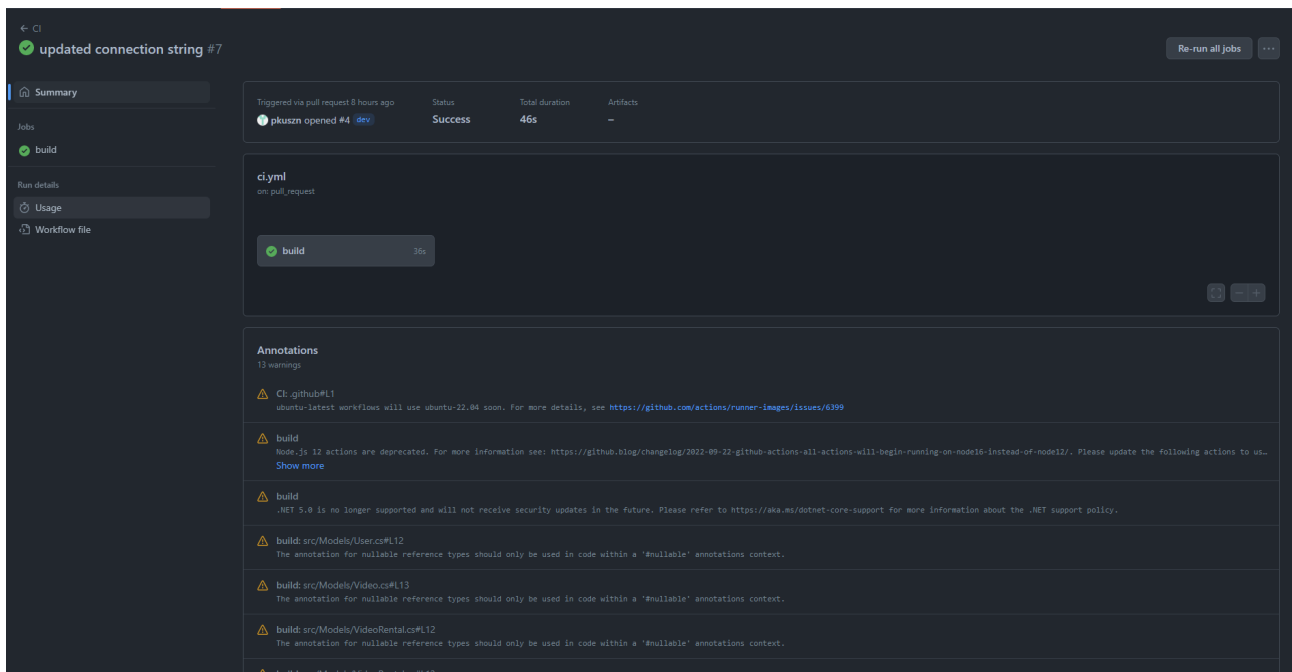


The screenshot shows the GitHub Actions interface for a workflow named 'CD'. The left sidebar contains a navigation menu with 'Actions', 'All workflows', 'CD', 'CI', 'Management', 'Caches', and 'Deployments'. The main area displays a table of 9 workflow runs. The first two runs are successful (green checkmarks) and represent merge pull requests. The remaining seven runs are failed (red X marks) and represent updates to the 'cd.yml' file. All runs are on the 'main' branch and occurred 9 hours ago.

Event	Status	Branch	Actor
Merge pull request #4 from pkuszn/dev	Success	main	pkuszn
Merge pull request #3 from pkuszn/dev	Success	main	pkuszn
Update cd.yml	Failure	main	pkuszn
Update cd.yml	Failure	main	pkuszn
Update cd.yml	Failure	main	pkuszn
Update cd.yml	Failure	main	pkuszn
Update cd.yml	Failure	main	pkuszn
Update cd.yml	Failure	main	pkuszn
Update cd.yml	Failure	main	pkuszn

## 2.3.3 Screeny

## Wizualizacja CI



The screenshot shows the details of a specific workflow run titled 'updated connection string #7'. The left sidebar includes 'Summary', 'Jobs', 'Run details', 'Usage', and 'Workflow file'. The main area shows the run was triggered via a pull request, has a status of 'Success', and a total duration of 46s. Below this, a job named 'build' is shown with a duration of 36s. The 'Annotations' section lists 13 warnings, including deprecated actions and nullable reference types.

Summary: updated connection string #7

Triggered via pull request 8 hours ago

Status: Success

Total duration: 46s

Artifacts: -

Jobs: build

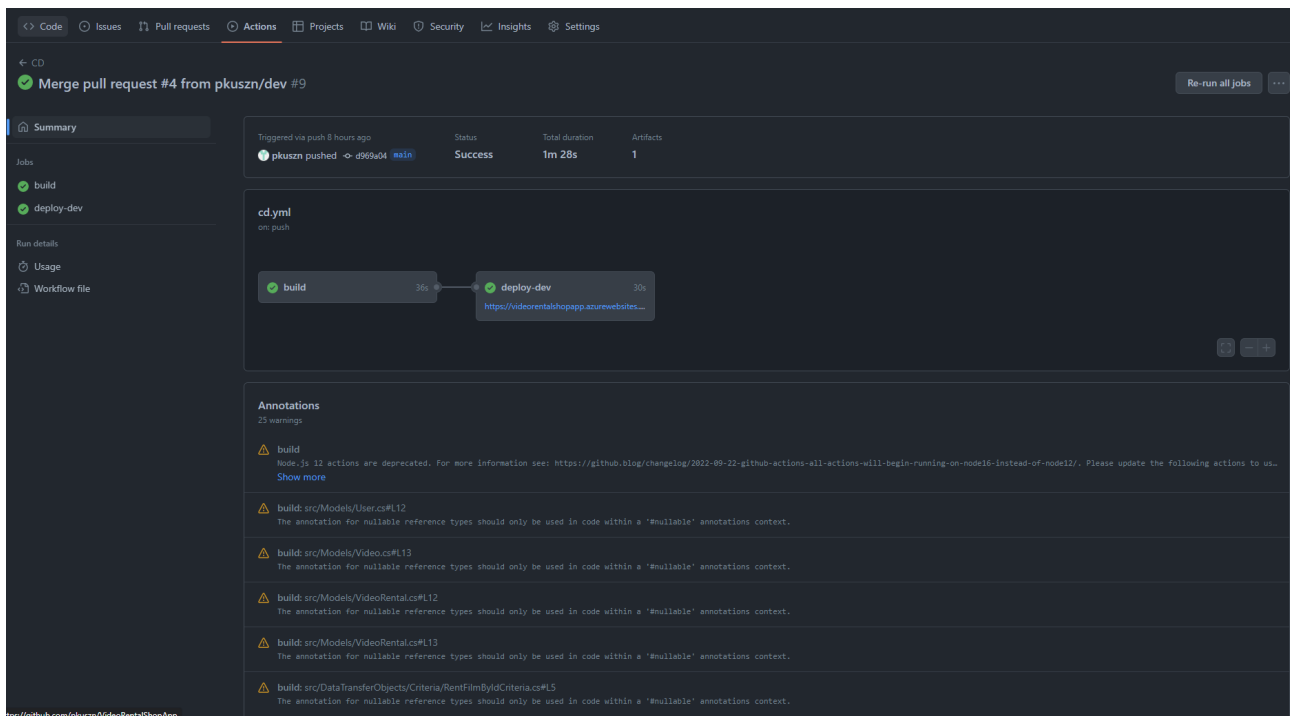
Run details: Usage

Workflow file

Annotations: 13 warnings

- Cl: github#11: ubuntu-latest workflows will use ubuntu-22.04 soon. For more details, see <https://github.com/actions/runner-images/issues/6399>
- build: Node.js 12 actions are deprecated. For more information see: <https://github.blog/changelog/2022-09-22-github-actions-all-actions-will-begin-running-on-node16-instead-of-node12/>. Please update the following actions to us. [Show more](#)
- build: .NET 5.0 is no longer supported and will not receive security updates in the future. Please refer to <https://aka.ms/dotnet-core-support> for more information about the .NET support policy.
- build: src/Models/User.cs#L12: The annotation for nullable reference types should only be used in code within a '#nullable' annotations context.
- build: src/Models/Video.cs#L13: The annotation for nullable reference types should only be used in code within a '#nullable' annotations context.
- build: src/Models/VideoRental.cs#L12: The annotation for nullable reference types should only be used in code within a '#nullable' annotations context.
- build: src/Models/VideoRental.cs#L13: The annotation for nullable reference types should only be used in code within a '#nullable' annotations context.

## Wizualizacja CD



## 2.4 Wdrożenie na chmurze Azure

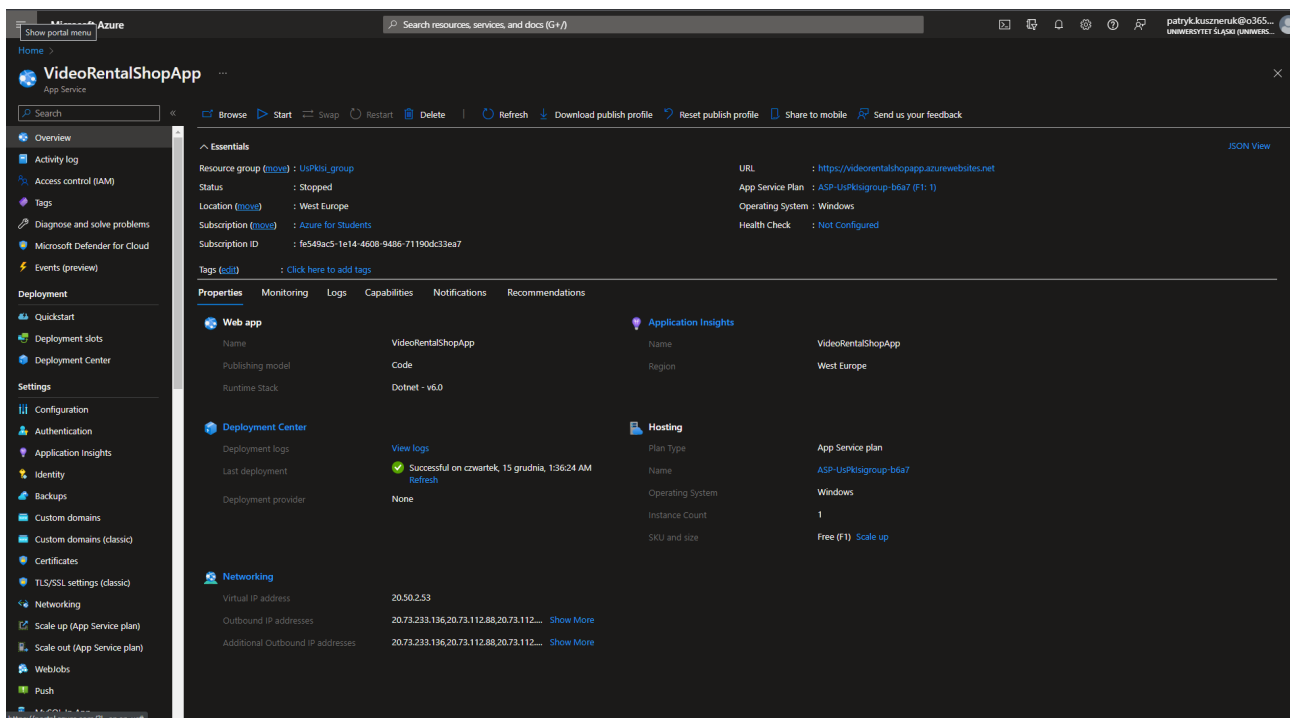
Do wdrożenia na chmurze potrzebujemy utworzyć

- App Service,
- Resource Group,
- Azure Cosmos DB for MongoDB account

### 2.4.1 App Service

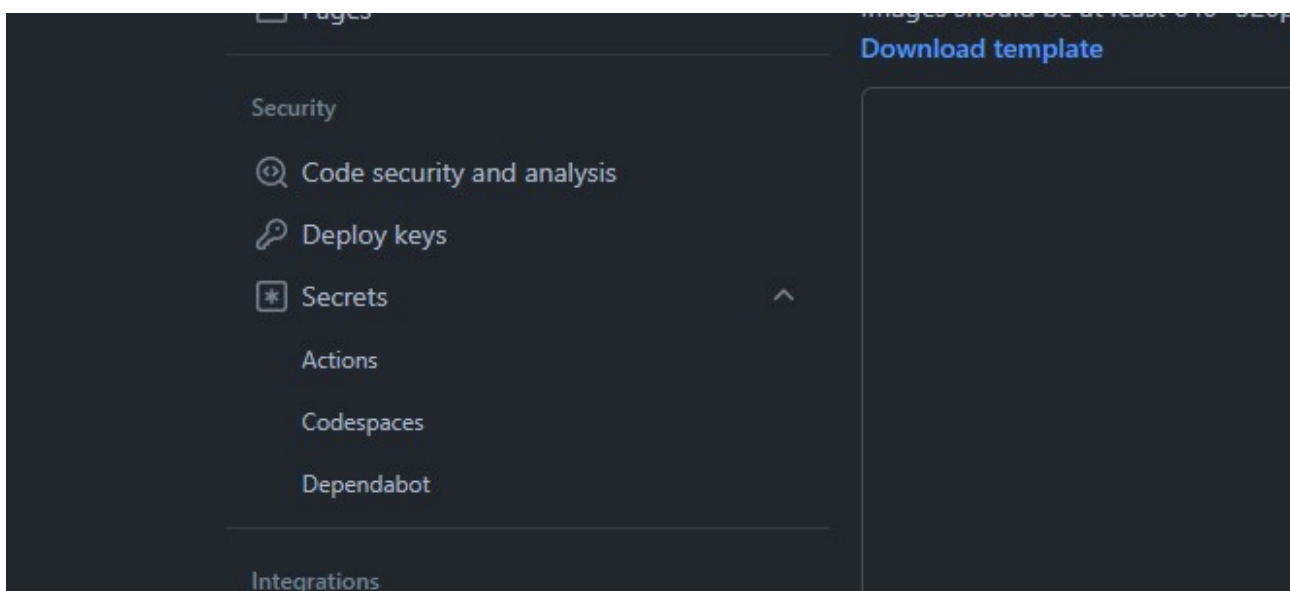
W celu wdrożenia aplikacji na chmurę Azure musimy stworzyć usługę App Service. Poniżej screenshot z panelu głównego



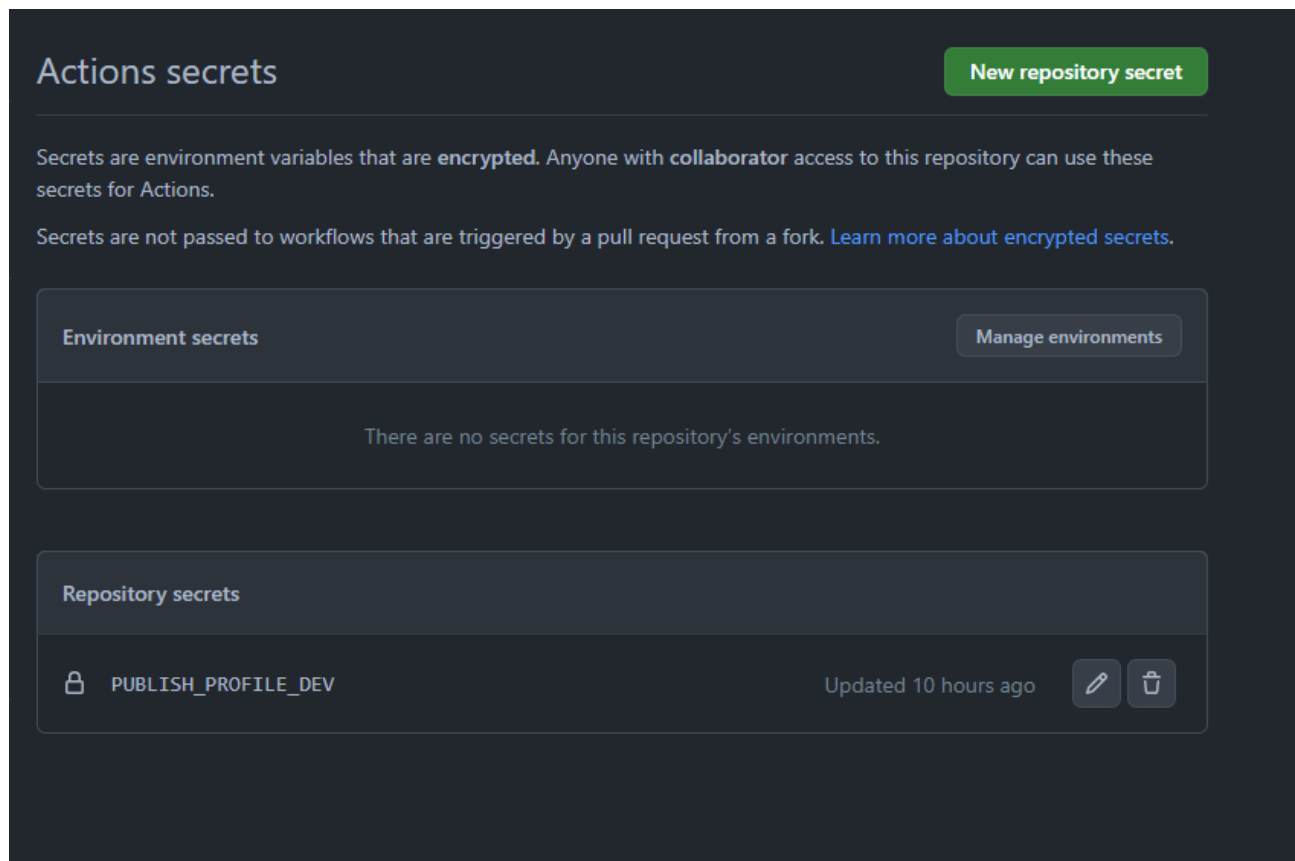


Aby wdrożyć aplikację musimy pobrać Publish Profile i ustawić w pliku .yml. W tym celu klikamy Download Publish Profile i kopiujemy zawartość pliku do GitHub

Na platformie GitHub przechodzimy w ustawienia projektu i przechodzimy do zakładki Secrets

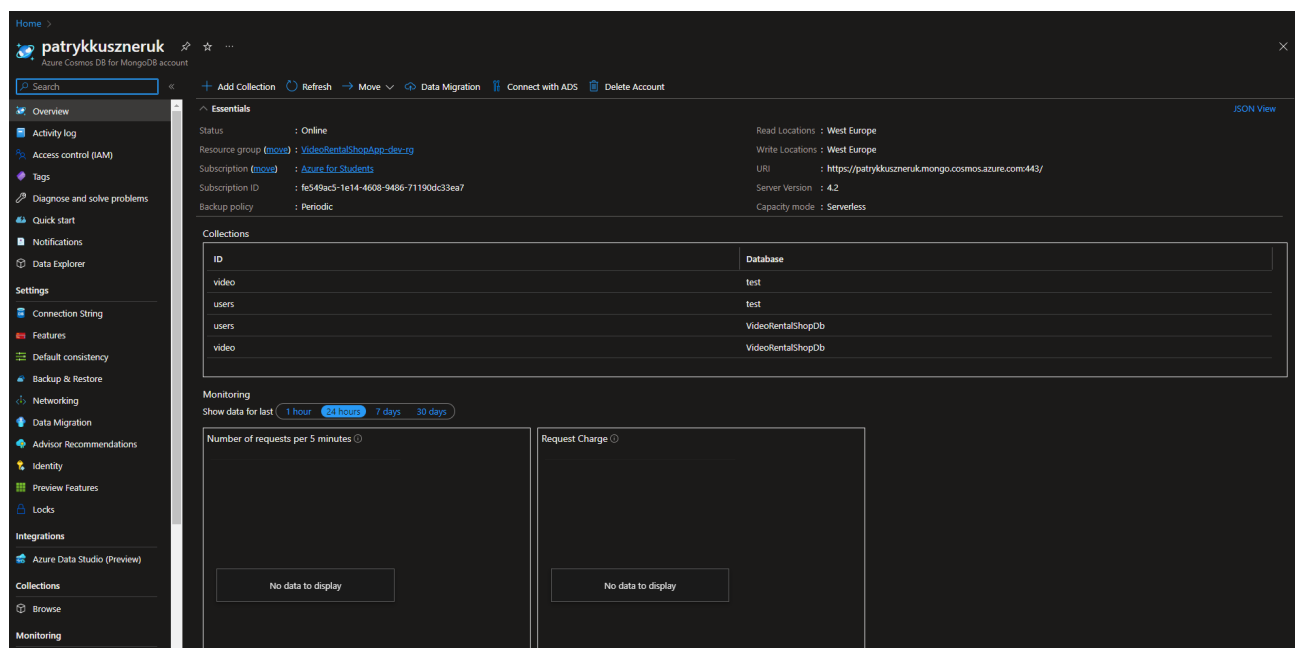


Tworzymy nowy sekret i wklejamy zawartość pobranego pliku



## 2.4.2 Azure Cosmos DB for mongoDB account

Do poprawnego działania aplikacji wymagana jest baza danych MongoDB. Azure umożliwia stworzenie instancji mongoDB



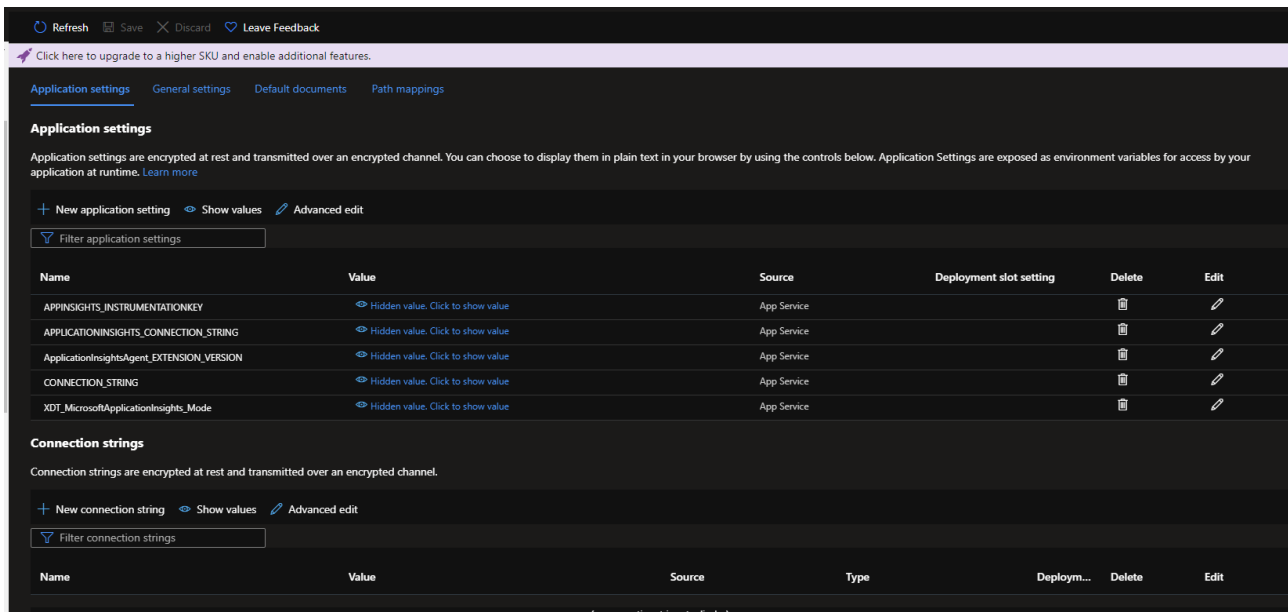
# Zapełnienie bazy kolekcją danych

The screenshot shows the Azure Cosmos DB Data Explorer interface. The left sidebar contains navigation options like Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Quick start, Notifications, Data Explorer, Settings, Connection String, Features, Default consistency, Backup & Restore, Networking, Data Migration, Advisor Recommendations, Identity, Preview Features, Integrations, Azure Data Studio (Preview), Collections, Browse, Monitoring, Insights, and Alerts. The main pane is titled 'patrykkusznerek | Data Explorer' and shows the 'MONGODB API' view. The 'DATA' section lists collections: 'test' (with sub-collections 'users' and 'video') and 'VideoRentalShopDb' (with sub-collections 'users' and 'video'). The 'NOTEBOOKS' section is currently not available. The right pane shows the MongoDB shell output, displaying a list of movies with their details, including title, genre, director, runtime, score, and description.

# Kopiuujemy Primary Connection String z zakładki Connection String

The screenshot shows the Azure Cosmos DB 'Connection String' page. The left sidebar is the same as the previous image. The main pane is titled 'patrykkusznerek | Connection String' and contains a 'Get started faster with driver specific connection information with our quick start.' link. Below this, there are sections for 'Read-write Keys' and 'Read-only Keys'. The 'Read-write Keys' section shows the 'PRIMARY CONNECTION STRING' and the 'SECONDARY CONNECTION STRING'. The 'PRIMARY CONNECTION STRING' is highlighted, and its value is copied to the clipboard. The 'SECONDARY CONNECTION STRING' is also shown. The 'Read-only Keys' section shows the 'SECONDARY CONNECTION STRING' and the 'PRIMARY CONNECTION STRING'. The 'PRIMARY CONNECTION STRING' is highlighted, and its value is copied to the clipboard. The 'SECONDARY CONNECTION STRING' is also shown. The bottom of the page states: 'Azure Cosmos DB has strict security requirements and standards. Azure Cosmos DB accounts require authentication and secure communication via SSL.'

Przechodzimy do App Service i klikamy w zakładkę Configuration. Klikamy ‘new application setting’. Nazywamy zmienną „CONNECTION\_STRING” i wklejamy skopiowaną wartość.



The screenshot shows the Azure App Service Configuration page. At the top, there are tabs for 'Application settings', 'General settings', 'Default documents', and 'Path mappings'. The 'Application settings' tab is active. Below the tabs, there's a section for 'Application settings' with a description and a 'Learn more' link. There are buttons for '+ New application setting', 'Show values', and 'Advanced edit'. A filter box is present. Below this is a table of application settings:

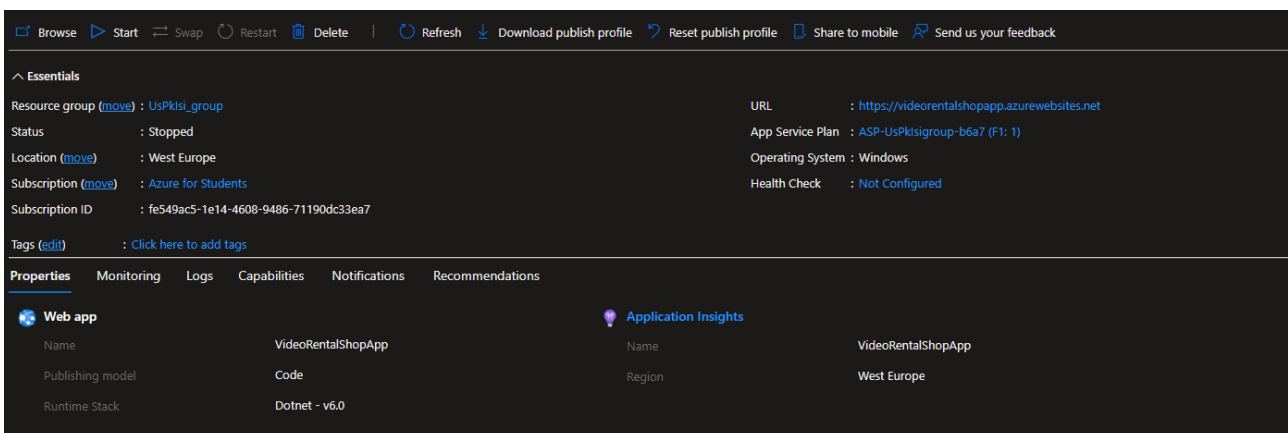
Name	Value	Source	Deployment slot setting	Delete	Edit
APPSIGHTS_INSTRUMENTATIONKEY	Hidden value. Click to show value	App Service			
APPLICATIONINSIGHTS_CONNECTION_STRING	Hidden value. Click to show value	App Service			
ApplicationInsightsAgent_EXTENSION_VERSION	Hidden value. Click to show value	App Service			
CONNECTION_STRING	Hidden value. Click to show value	App Service			
XDT_MicrosoftApplicationInsights_Mode	Hidden value. Click to show value	App Service			

Below the table is a section for 'Connection strings' with a description. There are buttons for '+ New connection string', 'Show values', and 'Advanced edit'. A filter box is present. Below this is a table of connection strings:

Name	Value	Source	Type	Deploym...	Delete	Edit
------	-------	--------	------	------------	--------	------

At the bottom, there's a note: '(for connection strings to display)'. The interface is dark-themed.

### 3. Uruchomienie aplikacji i prezentacja



The screenshot shows the Azure App Service Overview page. At the top, there are buttons for 'Browse', 'Start', 'Swap', 'Restart', 'Delete', 'Refresh', 'Download publish profile', 'Reset publish profile', 'Share to mobile', and 'Send us your feedback'. Below these buttons is the 'Essentials' section with the following information:

- Resource group (move): UsPkisi\_group
- Status: Stopped
- Location (move): West Europe
- Subscription (move): Azure for Students
- Subscription ID: fe549ac5-1e14-4608-9486-71190dc33ea7
- Tags (edit): Click here to add tags
- URL: https://videorentalshopapp.azurewebsites.net
- App Service Plan: ASP-UsPkisigroup-b6a7 (F1: 1)
- Operating System: Windows
- Health Check: Not Configured

Below the Essentials section is the 'Properties' section with tabs for 'Properties', 'Monitoring', 'Logs', 'Capabilities', 'Notifications', and 'Recommendations'. The 'Properties' tab is active. It shows two sections: 'Web app' and 'Application Insights'.

Web app		Application Insights	
Name	VideoRentalShopApp	Name	VideoRentalShopApp
Publishing model	Code	Region	West Europe
Runtime Stack	Dotnet - v6.0		

Aby uruchomić aplikację klikamy Start.

