

# Laboratorium CI/CD

---

Filip Maciąg, 21.11.2023

## Wymagania

- Konto na GitHubie
- Konto Azure + subskrypcja z aktywnymi środkami
- Visual Studio 2019/2022
- Dostęp do Azure DevOps

## Scenariusz

### Przygotowanie repozytorium

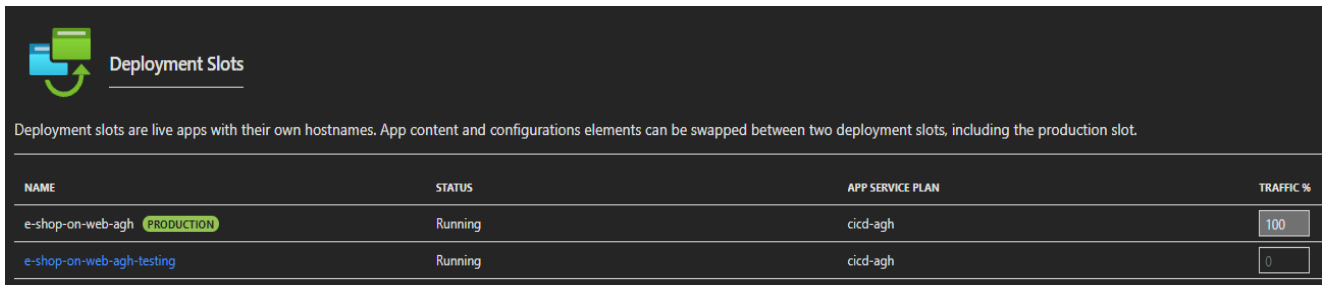
Zajęcia rozpoczniemy od wykonania forku repozytorium z przykładową aplikacją webową, która posłuży jako baza do tworzenia pipeline'ów dla CI/CD. W tym celu:

1. Zaloguj się na swoje konto GitHub
2. Wykonaj fork repozytorium: <https://github.com/filipmaciag/eShopOnWeb>
3. Można zapoznać się z tym jak wygląda aplikacja, uruchamiając projekt *Web* z poziomu Visual Studio

## Stworzenie bazowej infrastruktury w Azure

Kolejnym krokiem jest stworzenie bazowej infrastruktury dla wdrożenia aplikacji na platformie Azure. Aby tego dokonać:

4. Zaloguj się na swoje konto w *Azure Portal*
5. Stwórz *AppService* dla aplikacji webowej (analogiczna procedura wykonywana była na laboratoriach z Web App):
  - a. Na głównej stronie Azure Portal wybierz *Create a Resource*
    - i. Z przedstawionej listy możliwych produktów wybierz *Web App*
    - ii. Utwórz nową resource grupę wybierając *Create New* pod rozwijaną listą z resource grupami. Do nazwy resource grupy dodaj suffix
    - iii. Wpisz dowolną nazwę aplikacji z suffixem
    - iv. Publish: *Code*
    - v. Runtime stack: *.NET 6*
    - vi. Operating System: *Windows*
    - vii. Region: *Poland Central*
    - viii. Utwórz nowy app service plan wybierając *Create New* pod rozwijaną listą z planami. Nazwa planu dowolna. Sku and size: *Production -> Standard S1* (**płatny plan, więc po zajęciach należy wyłączyć *AppService***)
    - ix. W zakładce Monitoring -> Enable Application Insights: *No*
    - x. Kliknij *Review + Create*
6. Począć na zakończenie tworzenia aplikacji i następnie kliknąć w *Go to resource*
7. W oknie *AppService*'u wybrać z menu po lewej stronie *Deployment slots*
8. Wybrać *Add Slot*, nazwać nowy slot *testing* oraz potwierdzić przyciskiem *Add*



NAME	STATUS	APP SERVICE PLAN	TRAFFIC %
e-shop-on-web-agh <b>PRODUCTION</b>	Running	cicd-agh	100
e-shop-on-web-agh-testing	Running	cicd-agh	0

Subscription \* ⓘ Filip Maciąg - VS Subscription

Resource Group \* ⓘ cicdlab-test  
Create new

**Instance Details**

Name \* ci-cd-agh-lab-test ✓  
.azurewebsites.net

Publish \* ☒ Code ☐ Docker Container ☐ Static Web App

Runtime stack \* .NET 6 (LTS)

Operating System \* ☐ Linux ☒ Windows

Region \* Poland Central

Not finding your App Service Plan? Try a different region or select your App Service Environment.

**Pricing plans**

App Service plan pricing tier determines the location, features, cost and compute resources associated with your app.  
[Learn more](#)

Windows Plan (Poland Central) \* ⓘ cicdlabplann (S1)  
Create new

Pricing plan **Standard S1** (100 total ACU, 1.75 GB memory, 1 vCPU)

**Zone redundancy**

An App Service plan can be deployed as a zone redundant service in the regions that support it. This is a deployment time only decision. You can't make an App Service plan zone redundant after it has been deployed [Learn more](#)

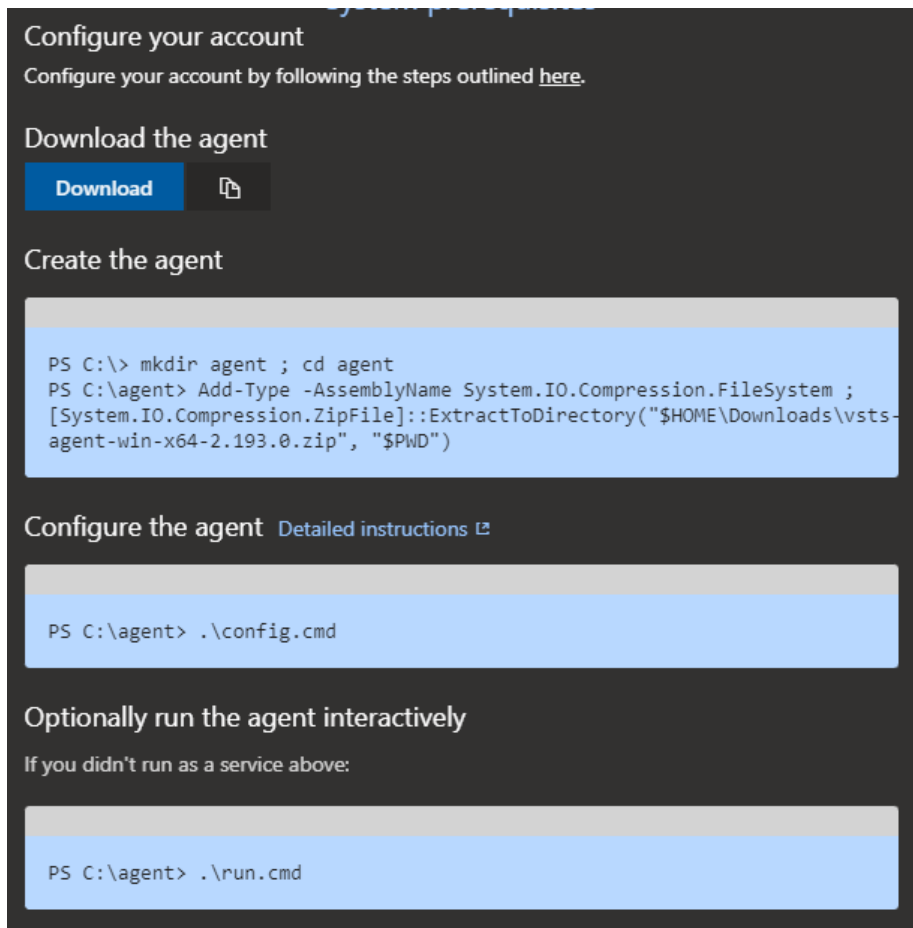
Zone redundancy ☐ Enabled: Your App Service plan and the apps in it will be zone redundant. The minimum App Service plan instance count will be three.  
☒ Disabled: Your App Service Plan and the apps in it will not be zone redundant. The minimum App Service plan instance count will be one.

## Konfiguracja Self-Hosted Agenta

Microsoft z powodów finansowych wyłączył możliwość korzystania z hostowanych przez nich maszyn w celu uruchamiania pipeline'ów. Z tego powodu na potrzeby zajęć zaadaptujemy własne maszyny do bycia tzw. „Self-Hosted Agents” i to na nich uruchamiane będą tworzone pipeline'y. Proces ten został dokładnie opisany w dokumentacji Microsoftu (<https://docs.microsoft.com/en-us/azure/devops/pipelines/agents/v2-windows?view=azure-devops>).

1. Zaloguj się do Azure DevOps i przejdź do swojej organizacji, jeśli automatycznie nie zostałeś tam przekierowany (<http://dev.azure.com/>)
2. W widoku organizacji w prawym górnym rogu wybierz *New Project*
  - a. Wpisz dowolną nazwę projektu
  - b. Resztę ustawień zostaw jako default
3. Po utworzeniu projektu w lewym dolnym rogu wybierz *Project Settings*

4. W oknie projektu wybierz zakładkę *Agent Pools*
5. Wybierz *Default*
6. W prawym górnym rogu (pod avatarą użytkownika) wybierz *New agent*
7. W wyświetlonym modalu kliknij przycisk *Download*
8. Po pobraniu się pliku *.zip* wykonaj na swojej maszynie z poziomu konsoli bądź Powershella komendy z sekcji *Create the agent* (nie zamykaj konsoli po wykonaniu poleceń)



9. Stwórz *Personal Access Token* w Azure Devops:
  - a. Kliknij w ikonę *User Settings* znajdującą się obok avataru użytkownika i z rozwijanego menu wybierz *Personal Access Tokens*
  - b. Kliknij przycisk *New Token*
  - c. Uzupełnij formularz tworzenia tokenu zgodnie z przykładem i wybierz *Create*

d. Skopiuj utworzony token i zapisz go w dowolnym miejscu (na potrzeby zajęć może to być plik tekstowy). Standardowo takie tokeny należy przechowywać w odpowiednio zabezpieczonym miejscu

10. Wróć do konsoli i wykonaj komendę z sekcji *Configure the agent* przekazując parametry według przykładu:

- a. Enter server URL:  
[https://dev.azure.com/<<NAZWA\\_TWOJEJ\\_ORGANIZACJI\\_AZURE\\_DEVOPS>>](https://dev.azure.com/<<NAZWA_TWOJEJ_ORGANIZACJI_AZURE_DEVOPS>>) (można skopiować z url Azure DevOps)
  - a. Enter authentication type: wciśnij *ENTER*
  - b. Enter Personal Access Token: wklej skopiowany access token
  - c. Enter agent pool: wciśnij *ENTER*
  - d. Enter agent name: wciśnij *ENTER*
  - e. Enter work folder: wciśnij *ENTER*
  - f. Enter run agent as service: Y (po zajęciach można wyłączyć autostart agenta z poziomu Usług Windows)
  - g. Enter enable SERVICE\_SID\_TYPE\_UNRESTRICTED for agent service: wciśnij *ENTER*
  - h. Enter User account to use for the service: wciśnij *ENTER*
  - i. Enter whether to prevent service starting immediately after configuration is finished?: wciśnij *ENTER*
11. Wróć do Azure DevOps i przenawiguj się do puli *Default* z punktu 5.
12. W zakładce *Agent* powinien być widoczny stworzony agent (maszyna użytkownika)

## Tworzenie build pipeline (CI)

Na tym etapie jesteśmy gotowi do stworzenia pipeline służącego do budowania przykładowej aplikacji oraz publikowania artefaktów powstałych w wyniku tego procesu. W tym celu:

1. W Azure DevOps przejdź do projektu stworzonego w poprzedniej części
2. Z menu po lewej stronie wybierz *Pipelines*, a następnie kliknij przycisk *Create Pipeline*
3. Wybierz *Use the classic editor* (opcja znajduje się pod listą systemów kontroli wersji)
4. Jako źródło kodu wybierz *GitHub*
5. Wybierz opcję *Authorize using OAuth*
6. Po autoryzacji wybierz trzy kropki obok pola *Repository* i z listy wybierz sforkowane na początku zajęć repozytorium z przykładową aplikacją
7. Wybierz *Continue*
8. Z listy template'ów wybierz *ASP.NET Core*

9. Kliknij w napis *Pipeline* pod którym wyświetlony powinien być czerwony wykrzyknik
10. Ustaw opcję "Project(s) to test" zgodnie z przykładem:

Name <sup>\*</sup>

Build pipeline

Agent pool ⓘ | Pool information | Manage ↗

Default

Parameters ⓘ | 🔗 Unlink all

Project(s) to restore and build 🔗

\*\*/\*.csproj

Project(s) to test 🔗

\*\*/\*tests/UnitTests.csproj  
\*\*/\*tests/IntegrationTests.csproj

*\*\*/\*tests/UnitTests.csproj*

*\*\*/\*tests/IntegrationTests.csproj*

11. Przejdź do zakładki *Triggers* i zaznacz checkbox *Enable continuous integration*
12. Zapisz i jednocześnie uruchom pipeline klikając przycisk *Save & queue*, a następnie potwierdź wybór klikając *Save and run*
13. Pipeline powinien zostać uruchomiony. Przejdź do następnego kroku, a pipeline będzie działał w tle.

## Tworzenie release pipelineu (CD)

Po stworzeniu build pipelineu jesteśmy gotowi do stworzenia pipelineu odpowiadającego za releasowanie aplikacji.

1. Z menu po lewej stronie wybierz *Releases*, a następnie kliknij *New Pipeline*
2. W wyświetlonym modalu wybierz opcję *Empty Job* znajdującą po lewej stronie pola wyszukiwania
3. Stage Name -> *Run Functional Tests*
4. W sekcji *Artifacts* wybierz *Add an artifact*
5. Z listy *Source* wybierz utworzony uprzednio build pipeline i potwierdź klikając *Add*
6. W kroku *Run Functional Tests* kliknij w link *1 job, 0 task*
7. Kliknij znak + w polu *Agent Job*
8. Z listy opcji wybierz *Visual Studio Test* i potwierdź klikając *Add*
9. Kliknij w krok dodany krok znajdujący się pod polem *Agent job*

Display name \*

Run Functional Tests

Test selection ^

Select tests using \* ⓘ

Test assemblies

Test files \* ⓘ

\*\*/\*tests/FunctionalTests.csproj

Search folder \* ⓘ

\$(System.DefaultWorkingDirectory)

Test results folder ⓘ

\$(Agent.TempDirectory)\TestResults

Test filter criteria ⓘ

☐ Run only impacted tests ⓘ

☐ Test mix contains UI tests ⓘ

10. Uzupełnij pola *Display name* oraz *Test files* zgodnie z przykładem, a resztę pól pozostaw bez zmian:

*\*\*/\*tests/FunctionalTests.csproj*

11. Kliknij w pole *Agent Job* i ustaw *Agent pool* na *default* pulę stworzoną na początku zajęć
12. Zapisz powstały krok pipeline, klikając *Save* (prawy górny róg)
13. Wróć do zakładki *Pipeline*
14. Wybierz znak + przy napisie nazwie pola *Stages*, a następnie kliknij *New Stage*
15. Z listy z opcjami wybierz *Azure App Service Deployment* i potwierdź wybierając *Apply*
16. Stage Name -> *Deploy to Test Environment*
17. Wybierz *1 job*, *1 task* pod stworzonym właśnie krokiem
18. Azure Subscription -> wybierz z listy swoją subskrypcję Azure, na której stworzone zostały aplikacje na początku zajęć
19. Kliknij przycisk *Authorize* i jeśli pojawi się okno logowania to zaloguj się na swoje konto Azure (sprawdź, czy przeglądarka ma odblokowane popupy)
20. Po zakończeniu autoryzacji w polu *App service name* wybierz aplikację stworzoną na początku zajęć
21. Kliknij w pole *Run on agent* znajdujące się nad stworzonym krokiem
22. Kliknij w pole *Agent Job* i ustaw *Agent pool* na *default* pulę stworzoną na początku zajęć
23. Kliknij w stage *Deploy Azure App Service*
24. Zaznacz checkbox przy *Deploy to Slot or App Service Environment* i ustaw slot na *testing*
25. Kliknij znak + znajdujący się w polu *Run on agent*

26. Z listy wybierz opcję *Azure App Service Settings* i potwierdź klikając *Add*
27. Wybierz *Azure subscription* oraz *App service name* takie same, jak w krokach 18. oraz 20.
28. Jeśli *Resource group* nie uzupełni się automatycznie, wybierz z listy resource grupę, w której znajduje się *App Service*
29. Slot -> *testing*
30. W polu *App settings* umieść następujący tekst: [{

"name": "ASPNETCORE\_ENVIRONMENT",

"value": "Development",

"slotSetting": true

}]

31. Kliknij w task *Deploy Azure App Service*
32. Kliknij 3 kropki obok pola *Package or folder* i przejdź do końca hierarchii folderów, a następnie wybierz plik *Web.zip* (plik dostępny będzie jedynie w przypadku, jeśli uruchomiony w poprzednim punkcie build pipeline z sukcesem zakończył swoje działanie). Potwierdź klikając *OK*
33. Kliknij przycisk *Save* w prawym górnym rogu
34. Przejdź ponownie do zakładki "*Pipeline*"
35. Rozwiń listę przy przycisku "+ *Add*" i wybierz opcję *Clone Stage*
36. Nazwij utworzony klon *Deploy to Production Environment*. Następnie zmodyfikuj utworzony klon wykonując następujące kroki:
  - a. Przejdź do zadań danego stage'a klikając link *1 job, 2 tasks*
  - b. Zmień parameter Slot w krokach *Deploy Azure App Service* oraz *Azure App Service Settings* na *production* (jesli nie będzie dostępny z listy, należy wpisać go ręcznie w pole)
  - c. W kroku *Azure App Service Settings...* w polu tekstowym *App settings* zmień wartość parametru *value* na ***Production***

## Uruchomienie Release Pipelinu

Jeśli tak, można przejść do procedury wdrożenia wyprodukowanego przez build pipeline artefaktu. Aby tego dokonać:

1. Z menu po lewej stronie wybierz *Releases*
2. W prawym górnym rogu kliknij przycisk *Create release*
3. Potwierdź wybierając *Create*
4. Kliknij w menu po lewej stronie w *Releases*. Nowy release powinien zostać utworzony i być widoczny. Kliknij w jego nazwę.
5. Obserwuj progres i wykonywanie kolejnych kroków
6. Jeśli wszystkie kroki zakończą się sukcesem, przejdź do *Azure Portal* i otwórz utworzony na potrzeby zajęć *App Service*. Domyślnie otworzony zostaje slot *production*
7. W zakładce *Configuration* sprawdź, że zostały dodane ustawienia odpowiednie wartości zmiennej *ASPNETCORE\_ENVIRONMENT*



8. Przejdź do aplikacji, klikając jej URL w zakładce *Overview*. W stopce aplikacji powinna znajdować się informacja, w jakim środowisku obecnie działa.
9. Wróć do widoku *App Service* i przejdź do zakładki *Deployment Slots*
10. Kliknij w slot *...testing*
11. Przejdź kroki 7-8 dla tego slotu. W stopce strony powinna znajdować się inna informacja, niż dla slotu *production*. URL aplikacji również powinien być inny.

**UWAGA:** Pamiętaj o wyłączeniu stworzonego *App Service* po zajęciach, ponieważ utworzony on został w oparciu o płatny plan i przy jakimkolwiek wykorzystaniu będzie generował koszty.

## Bonus (opcjonalnie dla chętnych)

- Zepsucie dowolnego testu w projekcie *UnitTests* i sprawdzenie, czy build pipeline “zapali się na czerwono”
- Zepsucie dowolnego testu w projekcie *FunctionalTests* i upewnienie się, że w takim przypadku release pipeline nie pozwoli nam na wdrożenie aplikacji
- Można pobrać repozytorium bądź z poziomu GitHuba spróbować zmienić coś w wyglądzie strony (np. napis w stopce) i ponownie zbudować i wdrożyć aplikację przy pomocy Release Pipeline.
- Konfiguracja Continuous Deploymentu (na zajęciach konfigurowaliśmy tzw. Continuous Delivery). Przy takiej konfiguracji automatycznie po uruchomieniu build pipeline uruchomi się release pipeline i aplikacja zostanie wdrożona. Możliwe jest to do ustawienia z poziomu konfiguracji Release Pipeline klikając ikonę błyskawicy w sekcji *Artifacts*.