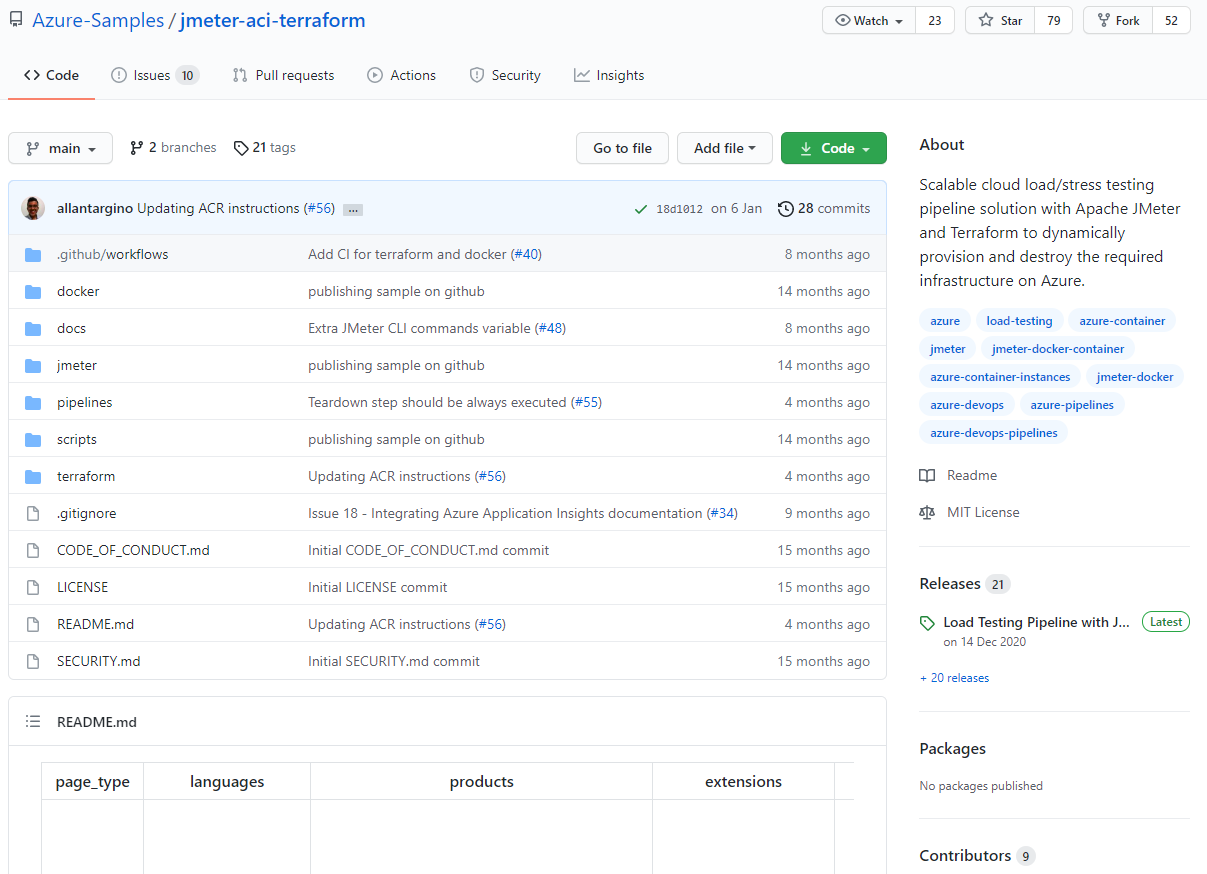
Do przeprowadzenie testów został użyty program JMeter w połączeniu z infrastrukturą Terraform pozwalającej dynamicznie tworzyć oraz usuwać potrzebną infrastrukture na platformie Azure DevOps kontrolowaną za pomocą Azure Pipeline.

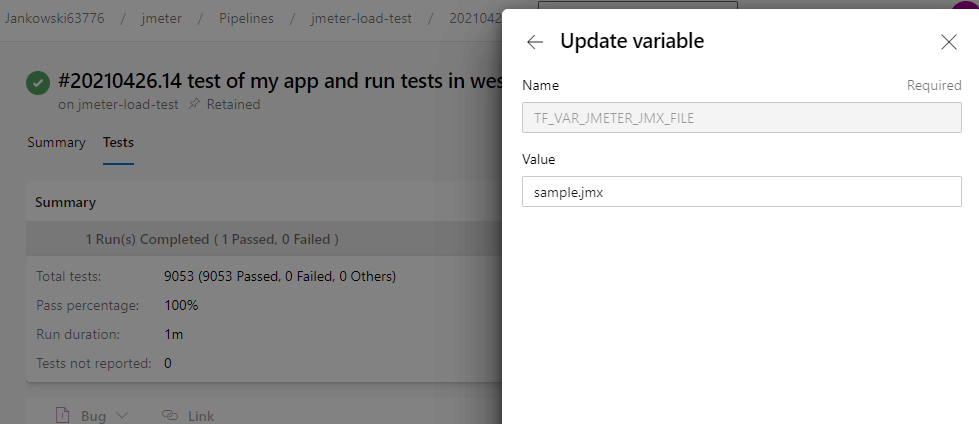
Do automatyzacji stawiania infrastruktury wykorzystałem przykładowe rozwiązanie z repozytorium jmeter-aci-terraform, które sforkowałem do swojego GitHuba I lekko skustomizowałem na potrzeby projektu. Automatyzacja ta:

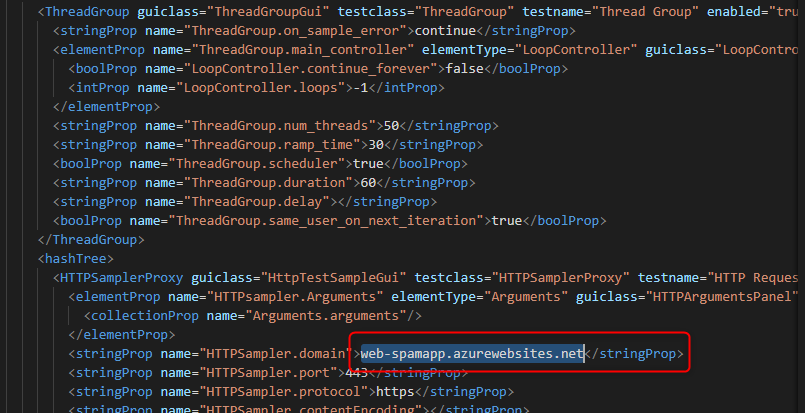
Opiera się o kontener zawierający JMeter

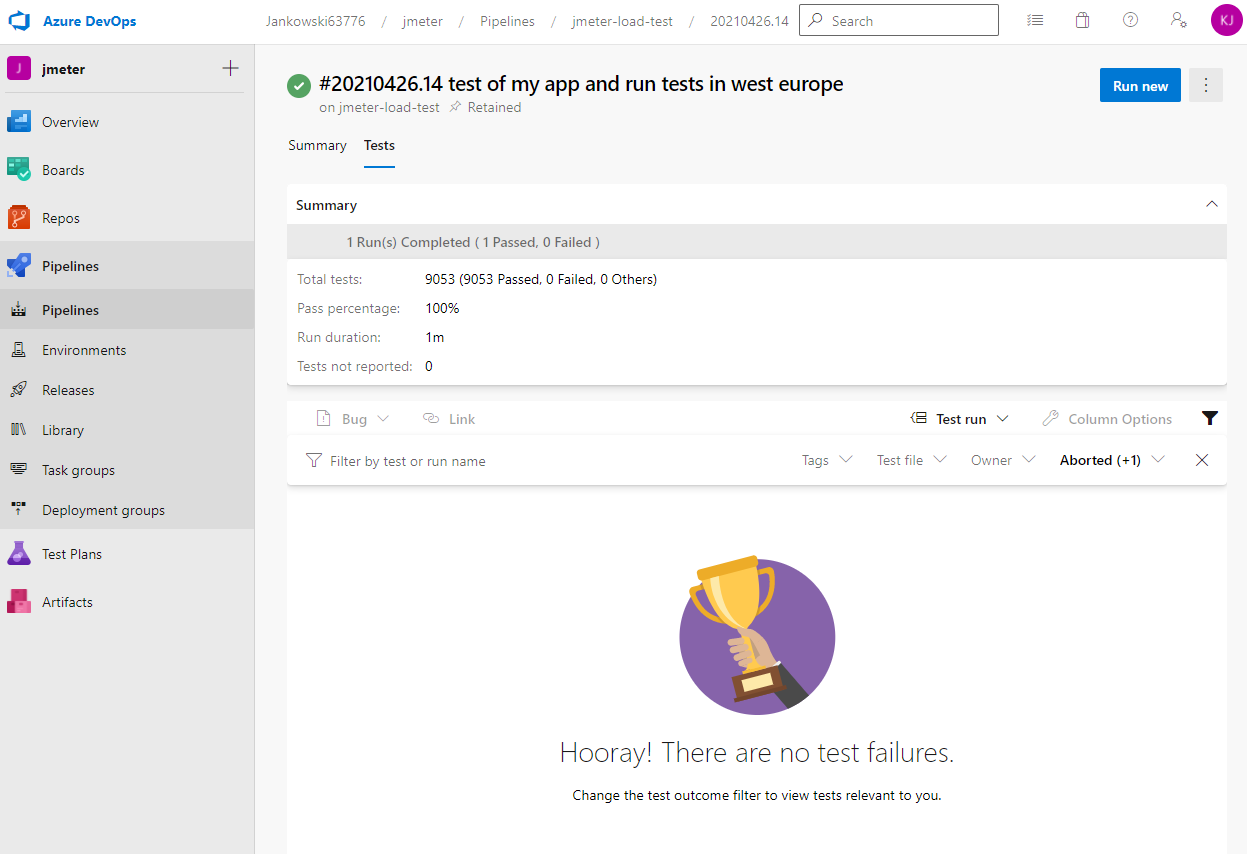
W Pipelinie Azure DevOps stawiane są Terraformem zasoby potrzebne do jego uruchomienia

Wyniki pracy JMeter są przekazywane do Pipelinea jako artefakt (reszta opracowania jest o niego oparta)

Wstępnym krokiem jest uruchomienie drugiego Pipeline do zbudowania w/w kontenera

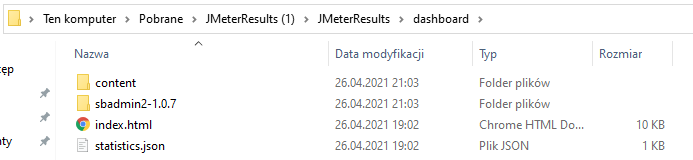
Podstawowym parametrem umożliwiającym wykorzystanie tej automatyzacji jest zmienna Pipeline’a pozwalająca na podanie nazwy pliku z testem JMeter, który będzie wykonany w danym przebiegu.

Plik sample.jmx zostal dostosowany aby testować obciążenie strony głównej naszego projektu Aplikacji Spamerskiej <https://web-spamapp.azurewebsites.net/>



W ciągu minuty zostało przeprowadzone 9053 testy, które zostały w 100% zdane.

Generator dashboardu jest rozszerzeniem program JMeter. Jego zadaniem jest odczytywanie I procesowanie plików CSV aby wygenerować plik HTML zawierający grafy z danymi z przeprowadzonych testów.



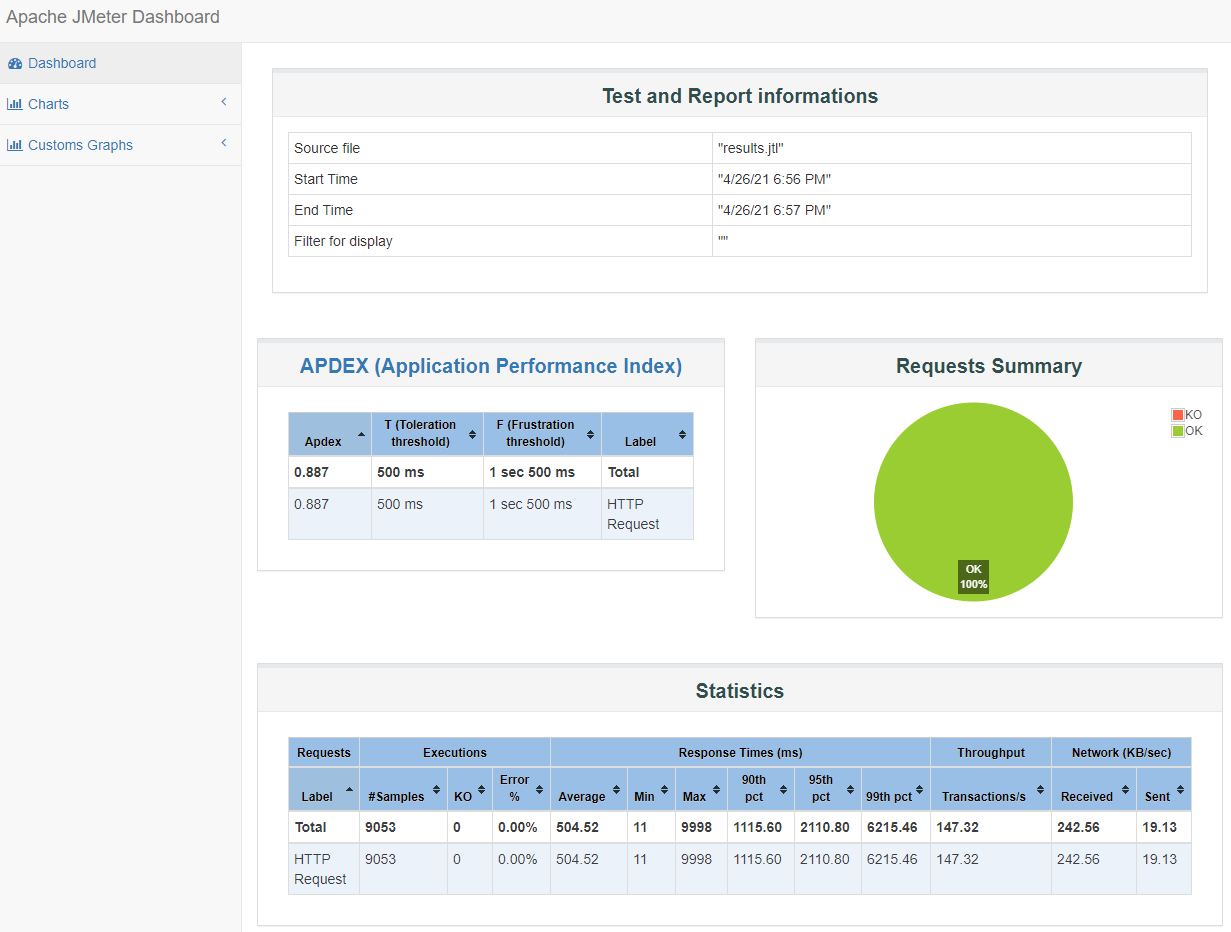
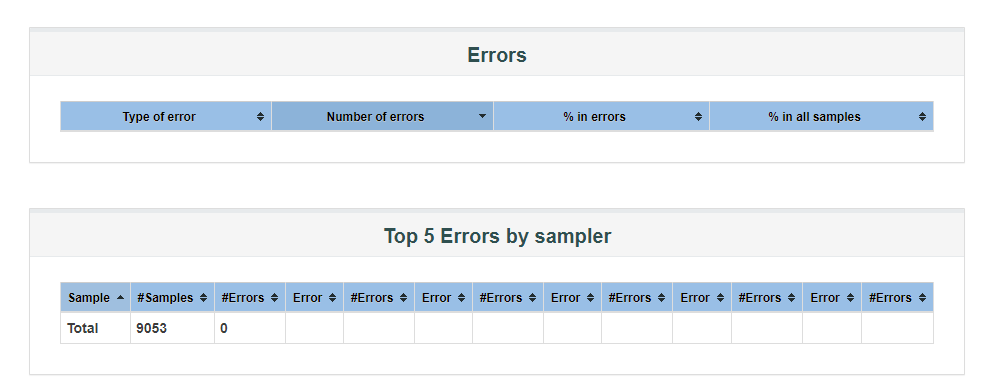


Tabela APDEX (Application Performance Index) porównuje wyniki testu ze standardem APDEX, który służy jako zunifikowany sposób na porównywanie wydajności aplikacji.

Graf Requests Summary pokazuje procentowy poziom udanych i nie udanych requestów.

Tabela Statistics zawiera wszystkie podsumowanie wszystkich metrics per transaction.

  
Tabela Errors podaje podsumowanie wszystkich errorów w proporcji do wszystkich requestów.