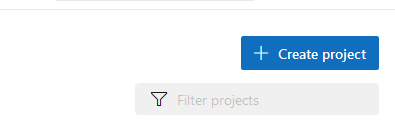
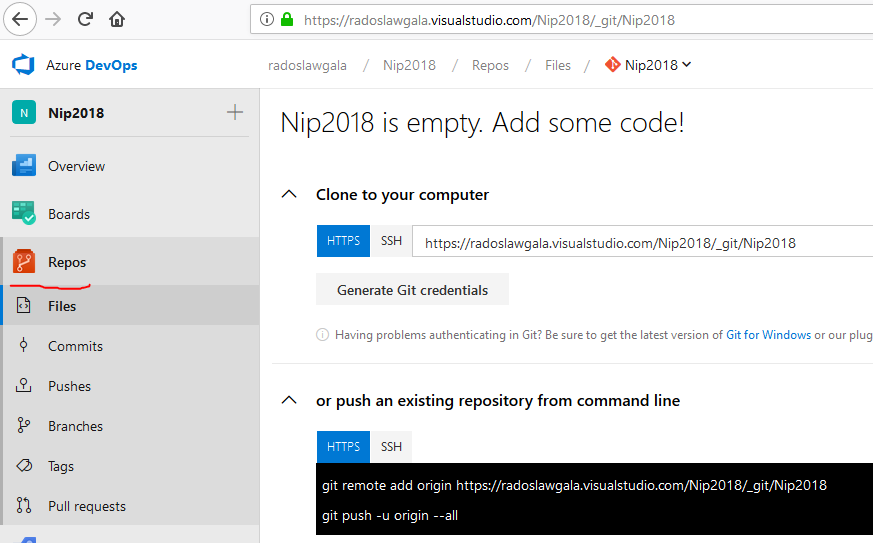
**SETUP**

1. Otwieramy <https://<projectname>.visualstudio.com>
2. Create New Project : 
3. Otwieramy link repozutoria w ramach projektu:



1. Zakładamy dwa repozytoria dla fron i backend – menu „New Repository”
2. Push naszych repozytoriów z komend (pod warunkiem że są częścią lokalnego repozytorium):
   1. git push -u origin –all ## sciagamy wszystkie branche z origin
   2. git remote remove origin ## usuwamy oryginalnego origin
   3. git remote add origin <adres nowego repozytorium w AzureDevOps>
   4. git push -u origin –all
3. Wykorzystując linka <https://docs.microsoft.com/en-us/azure/devops/pipelines/agents/v2-windows?view=vsts> postawić Agenta na lokalnym komputerze.
   1. Przed połączeniem Agenta należy stworzyć Pool pod linkiem “https://<account\_name> .visualstudio.com/<project\_name>/\_settings/agentqueues” .
   2. Generujemy token z odpowiednimi uprawnieniami
   3. Uruchamiamy konfigurację za pomocą .\config
   4. Uruchamiamy Agenta za pomocą polecenia .\run.cmd . Nie należy definiować seriwsu
4. Projekt FronEnd:
   1. Dodajemy 1 komponent za pomocą komendy “ng generate component TestComp”
   2. Dodajemy w pliku TestComp.component.spec.ts zamiast describe > fdescribie (priorytet w puszczaniu testów)

**Zadanie:**

* dla obu projektów stworzyć builda wraz z unit testami
* ustawić CI dla branch master – uruchomienie builda dla każdego commita dla danego branchabuild zwraca error jeżeli nie przejdzie chociaż jeden test.
* Opublikowany jest coverage z testów.
* Artefakty buildu są opublikowane w VSTS.

**Hints:**

* Nawigacja w ramach builda za pomocą zmiennych <https://docs.microsoft.com/en-us/azure/devops/pipelines/build/variables?view=vsts> Czterynajważniejsze:
  + Build.SourcesDirectory
  + Build.StagingDirectory
  + Build.BinariesDirectory
  + System.Debug
* Uruchomienie testów angula robimy za pomocą komendy “ng test --progress false --code-coverage --single-run=true --sourcemaps=false --reporters=junit”
* npm install karma-junit-reporter --save-dev
* Dodać do karma.conf.js plugin require('karma-junit-reporter')
* Dodać do karma.conf.js

coverageIstanbulReporter: {

reports: ['html', 'cobertura'],

fixWebpackSourcePaths: true,

skipFilesWithNoCoverage: false,

'report-config': {

html: {

subdir: 'html'

}

},

},

* Wyniki Testow pojawią się w pliku XML, musimy je załadować do VSTS za pomocą tasku PublishTestResults
* Pomocne “stepy” w VSTS:
  + “.Net Core”
  + “Npm”
  + “Publish Code Coverage Results”
  + “Publish Test Results”
  + “Publish Build Artifacts”
  + “Visual Studio Build”
  + “Visual Studio Test”
  + “Nuget Restore”