TESTY MACIERZOWE W AZURE DEVOPS

OSKAR DUDYCZ

KILKA SŁÓW O MNIE

MARTEN

~12 LAT DOŚWIADCZENIA

OSKAR_AT_NET

SZKOLA-EVENT-SOURCING.PL

KILKA SŁÓW O MNIE

Jenkins

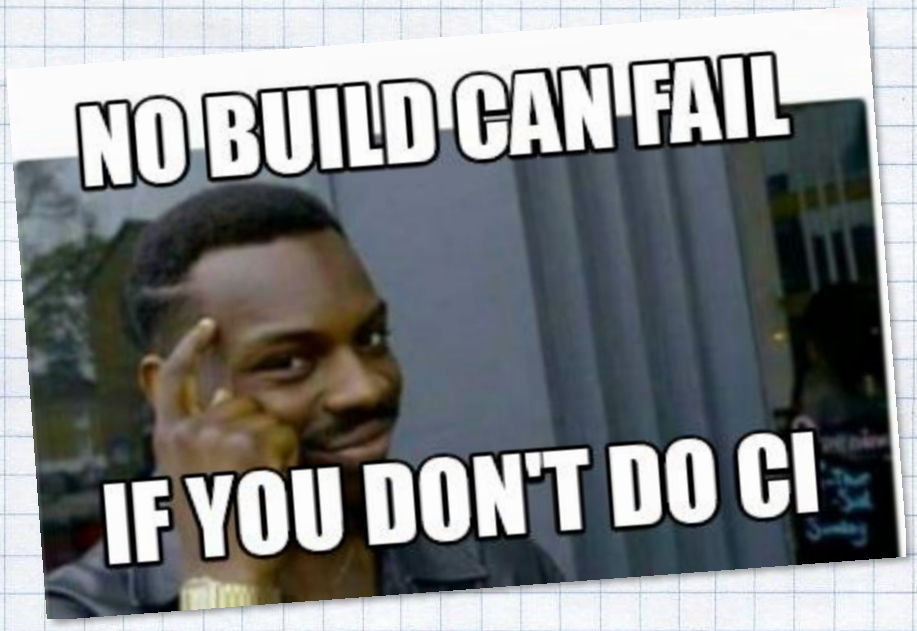
Team Foundation Server

Azure Pipelines

AppVeyor

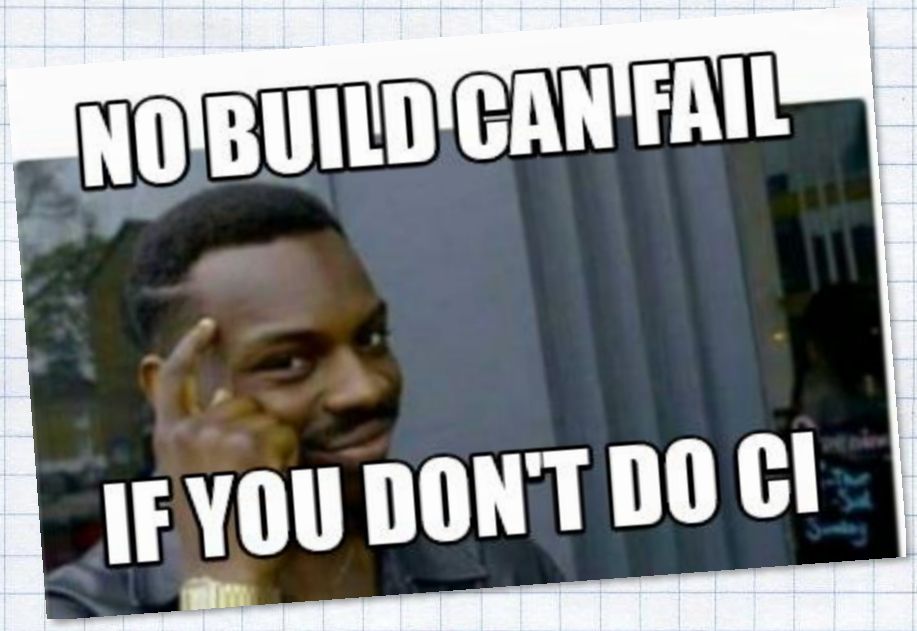
Bamboo

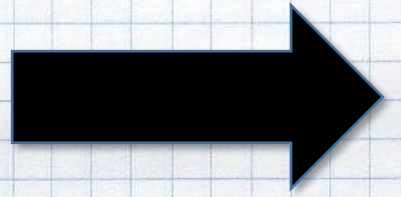
POCZĄTKI



DEVELOPMENT

POCZĄTKI





DEVELOPMENT

WDROŻENIE

BIBLIOTEKI OPEN SOURCE

1. WIELOPLATFORMOWE
2. PROSTA KONFIGURACJA ŚRODOWISKA PROGRAMISTYCZNEGO
3. BRAK ZNAJOMOŚCI WSZYSTKICH SCENARIUSZY OSTATECZNEGO UŻYCIA
4. ZRÓŻNICOWANI KONTRYBUTORZY
5. ASYNCHRONICZNA PRACA

MATRIX TESTS

MATRIX TESTS

MATRIX TESTS

Result table

Home \ Away	ARS	AST	BOU	CHE	CRY	EVE	LEI	LIV	MCI	MUN	NEW	NOR	SOU	STK	SUN	SWA	TOT	WAT	WBA	WHU
Arsenal		4–0	2–0	0–1	1–1	2–1	2–1	0–0	2–1	3–0	1–0	1–0	0–0	2–0	3–1	1–2	1–1	4–0	2–0	0–2
Aston Villa	0–2		1–2	0–4	1–0	1–3	1–1	0–6	0–0	0–1	0–0	2–0	2–4	0–1	2–2	1–2	0–2	2–3	0–1	1–1
AFC Bournemouth	0–2	0–1		1–4	0–0	3–3	1–1	1–2	0–4	2–1	0–1	3–0	2–0	1–3	2–0	3–2	1–5	1–1	1–1	1–3
Chelsea	2–0	2–0	0–1		1–2	3–3	1–1	1–3	0–3	1–1	5–1	1–0	1–3	1–1	3–1	2–2	2–2	2–2	2–2	2–2
Crystal Palace	1–2	2–1	1–2	0–3		0–0	0–1	1–2	0–1	0–0	5–1	1–0	1–0	2–1	0–1	0–0	1–3	1–2	2–0	1–3
Everton	0–2	4–0	2–1	3–1	1–1		2–3	1–1	0–2	0–3	3–0	3–0	1–1	3–4	6–2	1–2	1–1	2–2	0–1	2–3
Leicester City	2–5	3–2	0–0	2–1	1–0	3–1		2–0	0–0	1–1	1–0	1–0	1–0	3–0	4–2	4–0	1–1	2–1	2–2	2–2
Liverpool	3–3	3–2	1–0	1–1	1–2	4–0	1–0		3–0	0–1	2–2	1–1	1–1	4–1	2–2	1–0	1–1	2–0	2–2	0–3
Manchester City	2–2	4–0	5–1	3–0	4–0	0–0	1–3	1–4		0–1	6–1	2–1	3–1	4–0	4–1	2–1	1–2	2–0	2–1	1–2
Manchester United	3–2	1–0	3–1	0–0	2–0	1–0	1–1	3–1	0–0		0–0	1–2	0–1	3–0	3–0	2–1	1–0	1–0	2–0	0–0
Newcastle United	0–1	1–1	1–3	2–2	1–0	0–1	0–3	2–0	1–1	3–3		6–2	2–2	0–0	1–1	3–0	5–1	1–2	1–0	2–1
Norwich City	1–1	2–0	3–1	1–2	1–3	1–1	1–2	4–5	0–0	0–1	3–2		1–0	1–1	0–3	1–0	0–3	4–2	0–1	2–2
Southampton	4–0	1–1	2–0	1–2	4–1	0–3	2–2	3–2	4–2	2–3	3–1	3–0		0–1	1–1	3–1	0–2	2–0	3–0	1–0
Stoke City	0–0	2–1	2–1	1–0	1–2	0–3	2–2	0–1	2–0	2–0	1–0	3–1	1–2		1–1	2–2	0–4	0–2	0–1	2–1
Sunderland	0–0	3–1	1–1	3–2	2–2	3–0	0–2	0–1	0–1	2–1	3–0	1–3	0–1	2–0		1–1	0–1	0–1	0–0	2–2
Swansea City	0–3	1–0	2–2	1–0	1–1	0–0	0–3	3–1	1–1	2–1	2–0	1–0	0–1	0–1	2–4		2–2	1–0	1–0	0–0
Tottenham Hotspur	2–2	3–1	3–0	0–0	1–0	0–0	0–1	0–0	4–1	3–0	1–2	3–0	1–2	2–2	4–1	2–1		1–0	1–1	4–1
Watford	0–3	3–2	0–0	0–0	0–1	1–1	0–1	3–0	1–2	1–2	2–1	2–0	0–0	1–2	2–2	1–0	1–2		0–0	2–0
West Bromwich Albion	2–1	0–0	1–2	2–3	3–2	2–3	2–3	1–1	0–3	1–0	1–0	0–1	0–0	2–1	1–0	1–1	1–1	0–1		0–3
West Ham United	3–3	2–0	3–4	2–1	2–2	1–1	1–2	2–0	2–2	3–2	2–0	2–2	2–1	0–0	1–0	1–4	1–0	3–1	1–1	

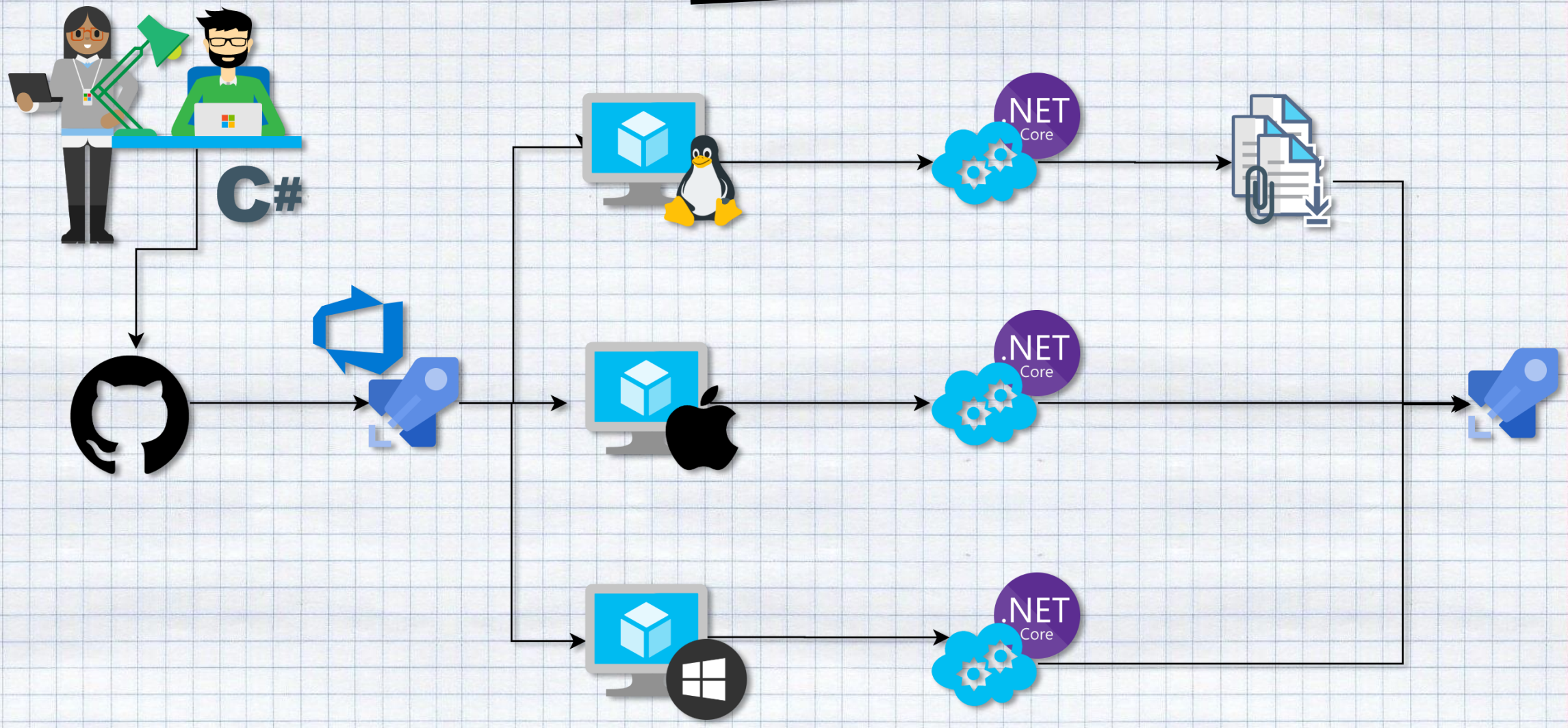
Updated to games played on 17 May 2016.

Source: Barclays Premier League football scores & results

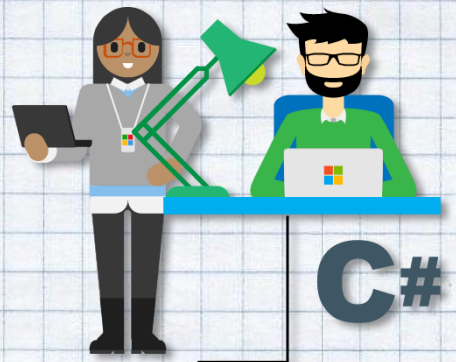
1 ^ The home team is listed in the left-hand column.

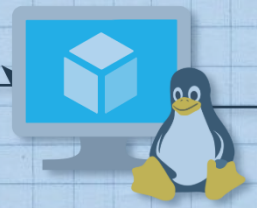
Colours: Blue = home team win; Yellow = draw; Red = away team win.

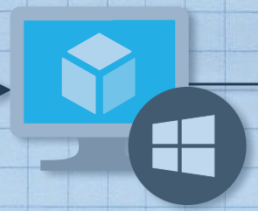
BUILD

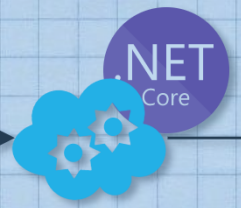


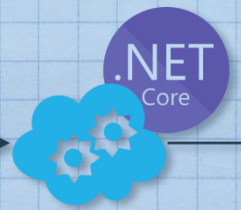
BUILD

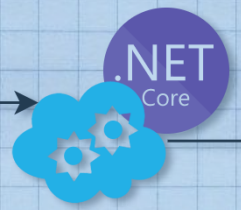


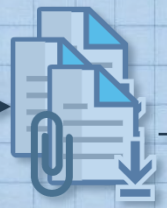




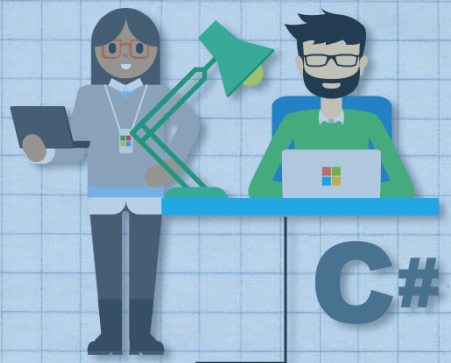


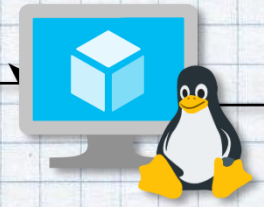


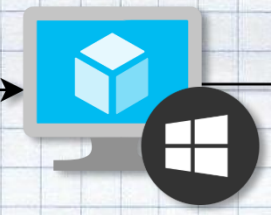


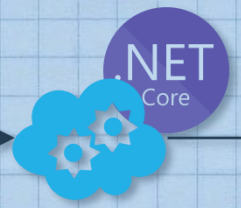


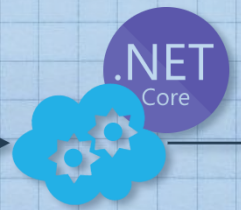
BUILD

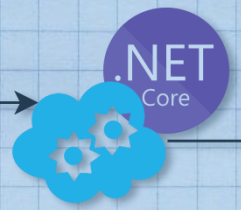


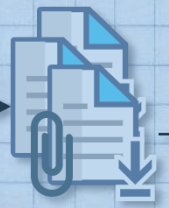


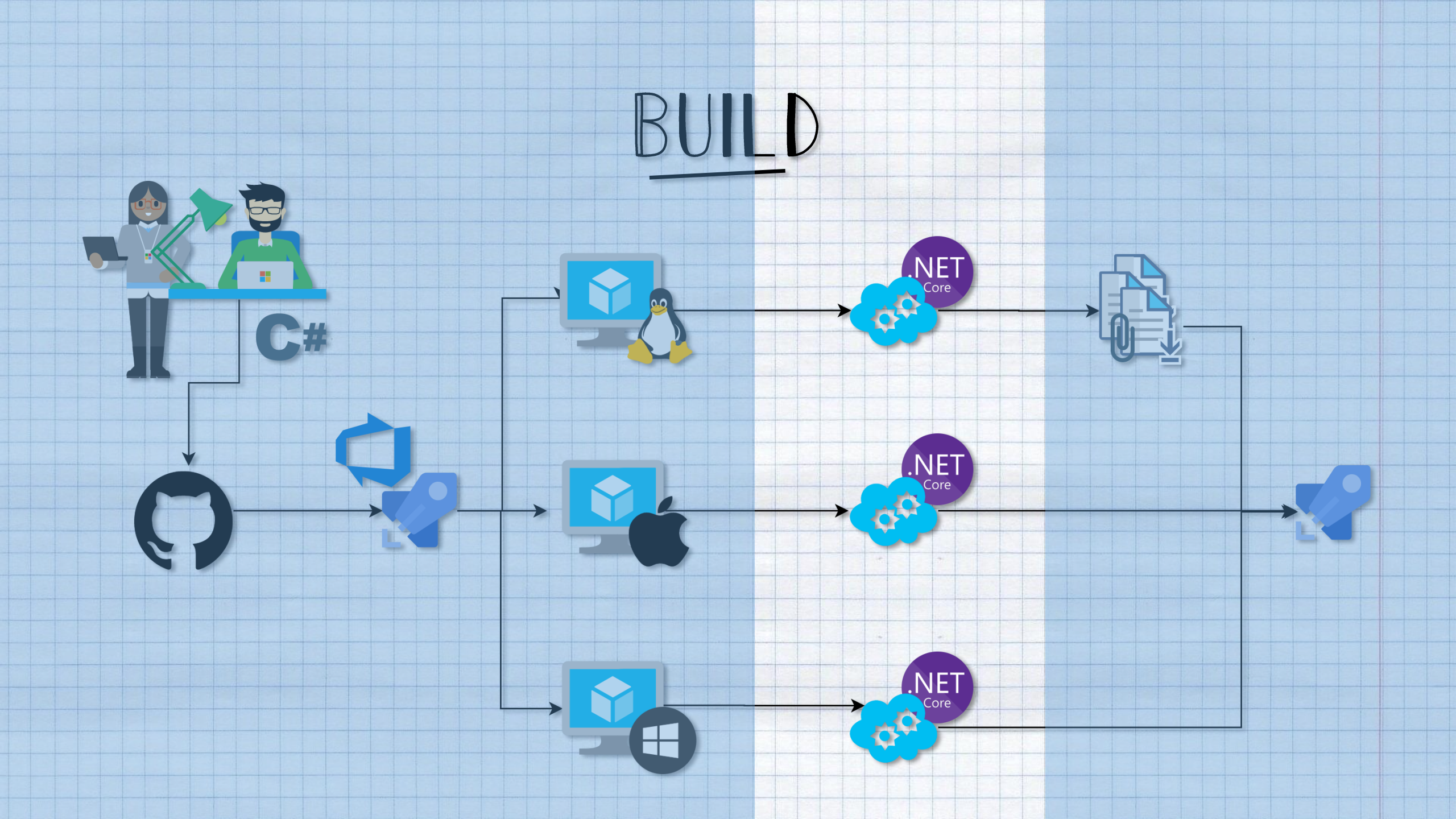




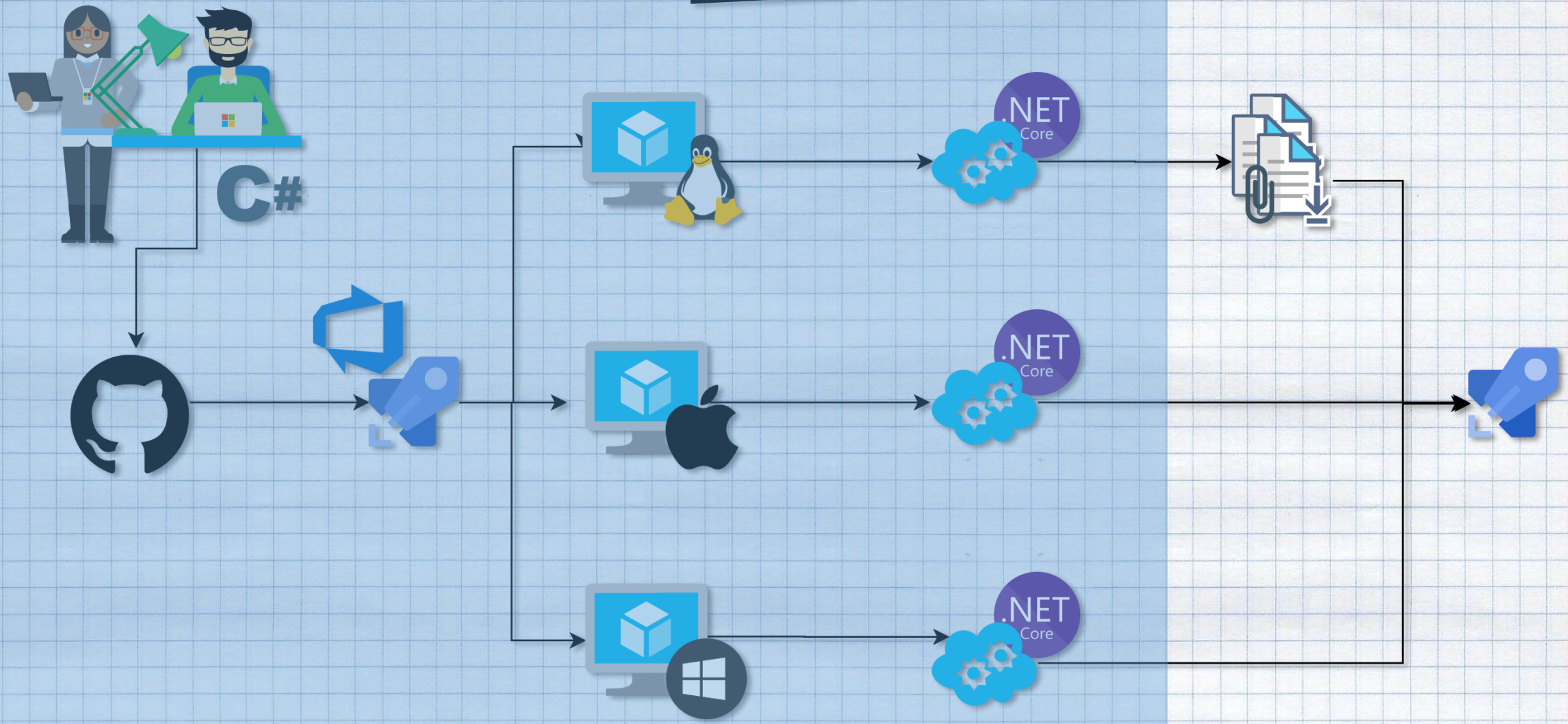




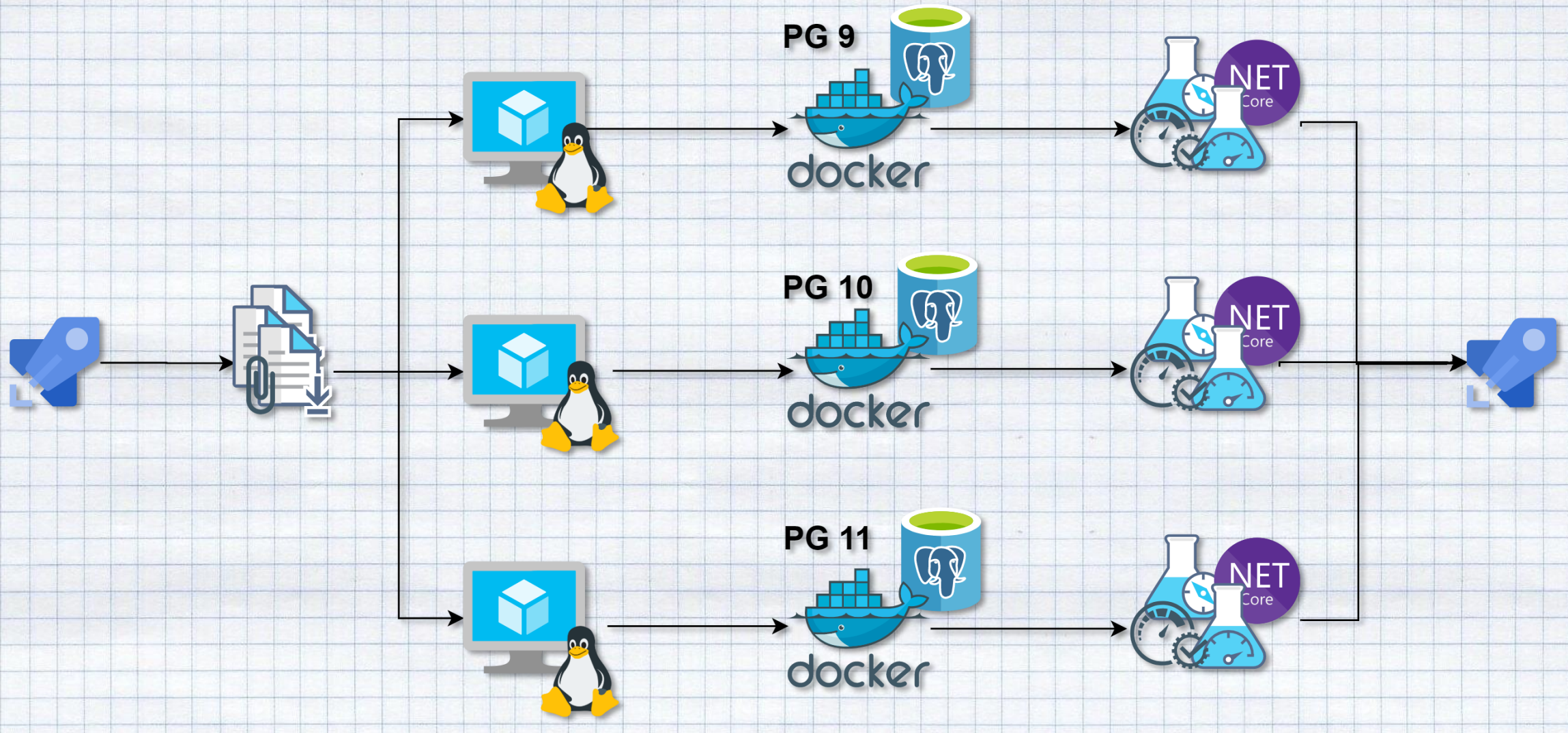


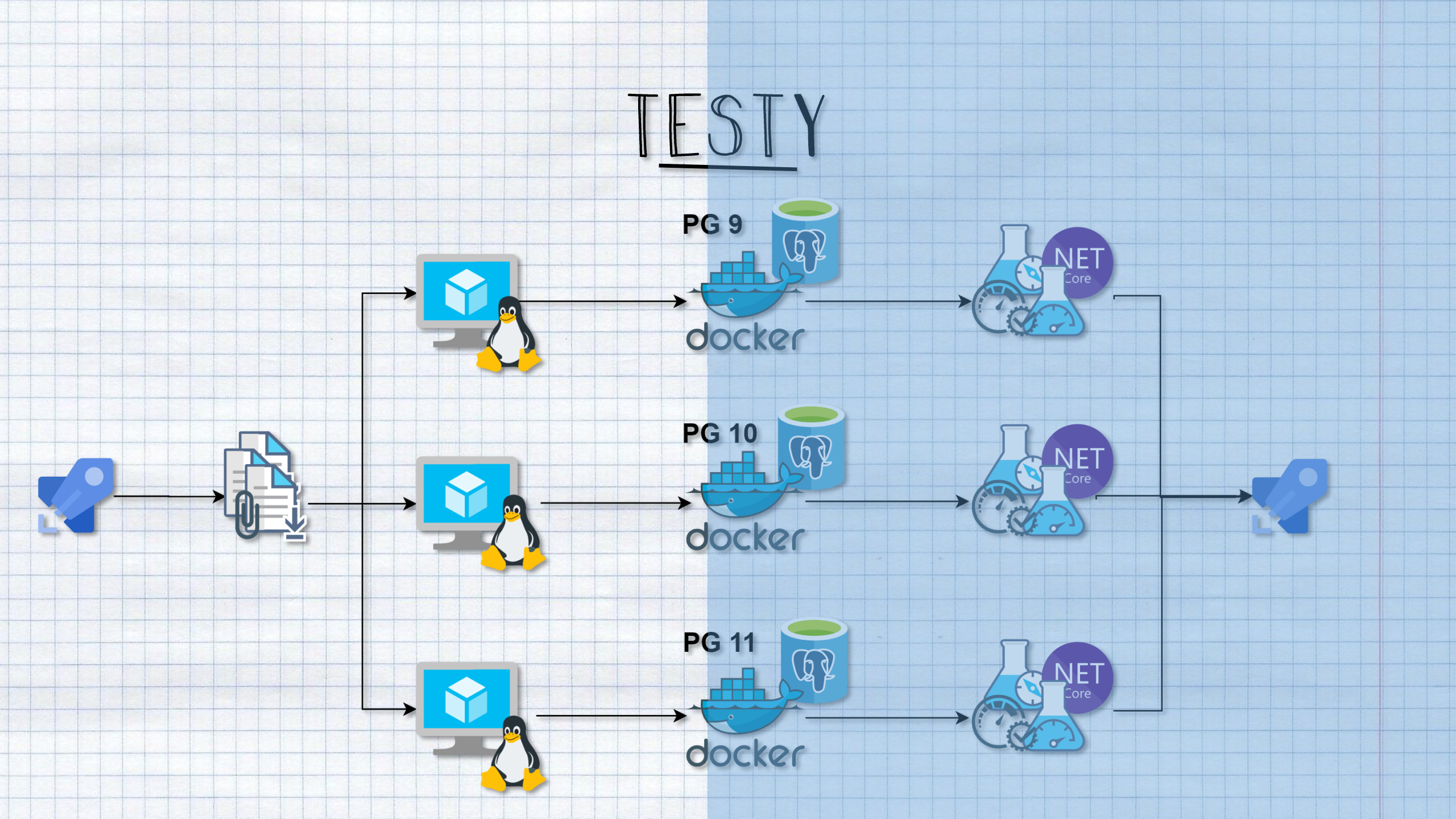

BUILD
C#
.NET
Core
.NET
Core
.NET
Core

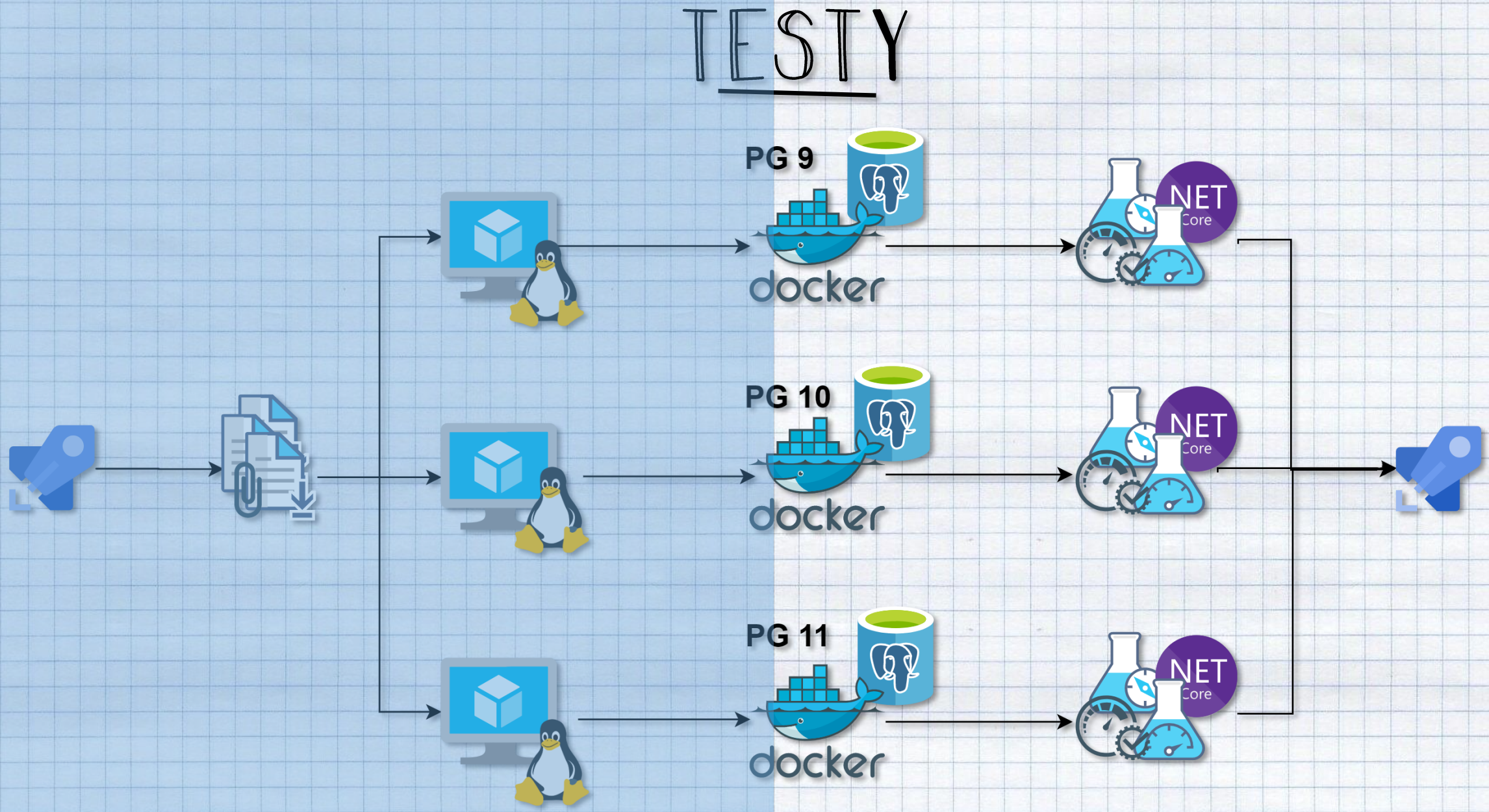
BUILD



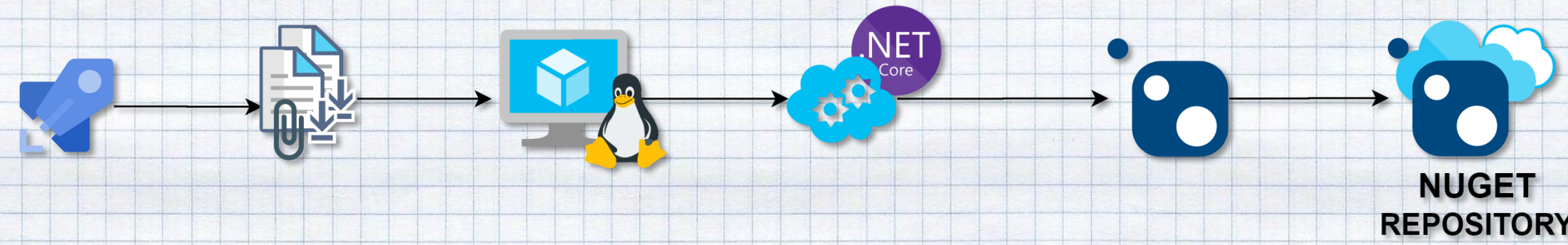
TESTY


PG 9
docker
PG 10
docker
PG 11
docker
NET
Core
NET
Core
NET
Core

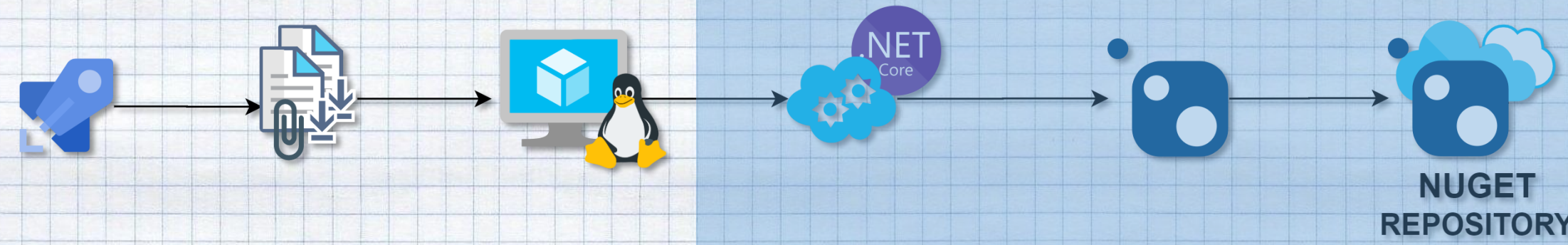

TESTY
PG 9
docker
PG 10
docker
PG 11
docker
NET
Core
NET
Core
NET
Core


TESTY
PG 9
docker
PG 10
docker
PG 11
docker
NET
Core
NET
Core
NET
Core

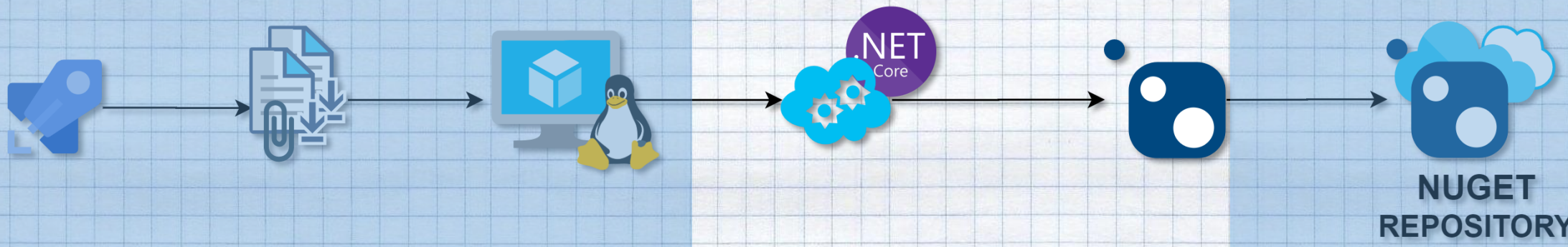
PUBLISH



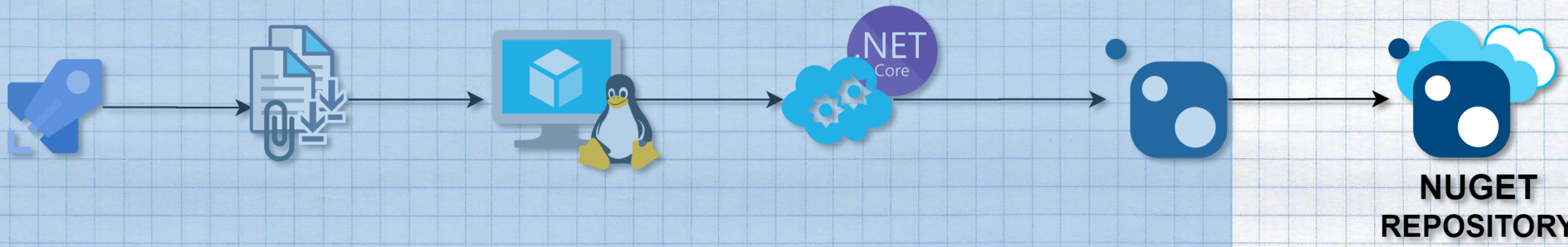
PUBLISH



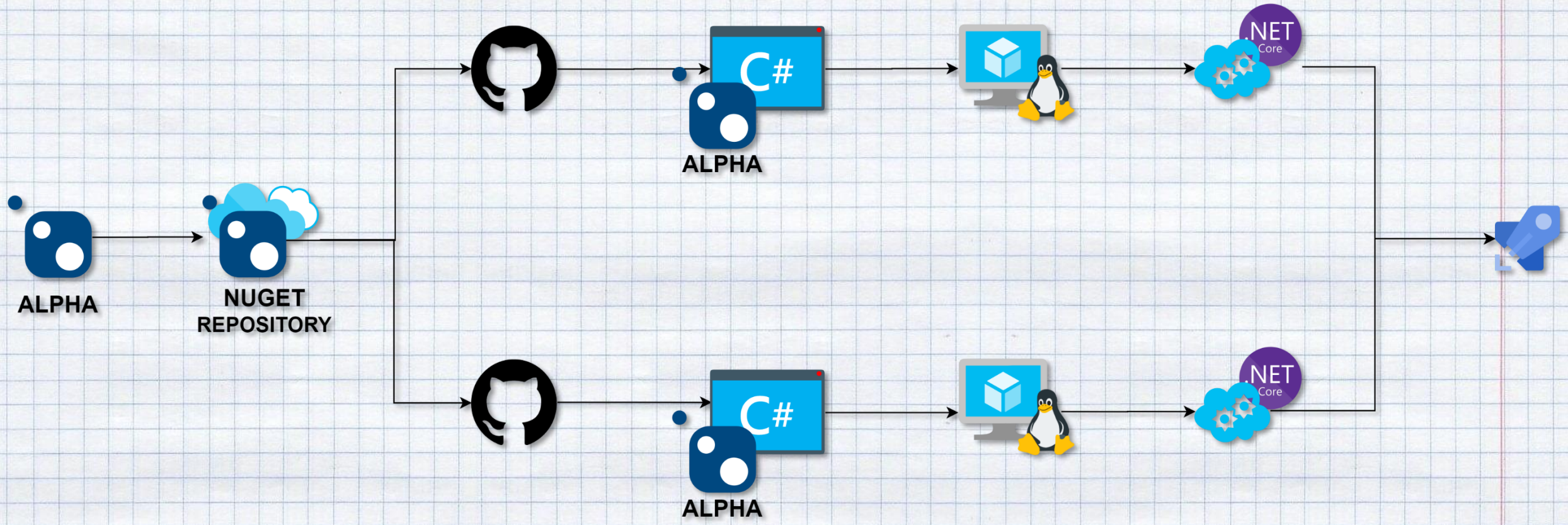
PUBLISH



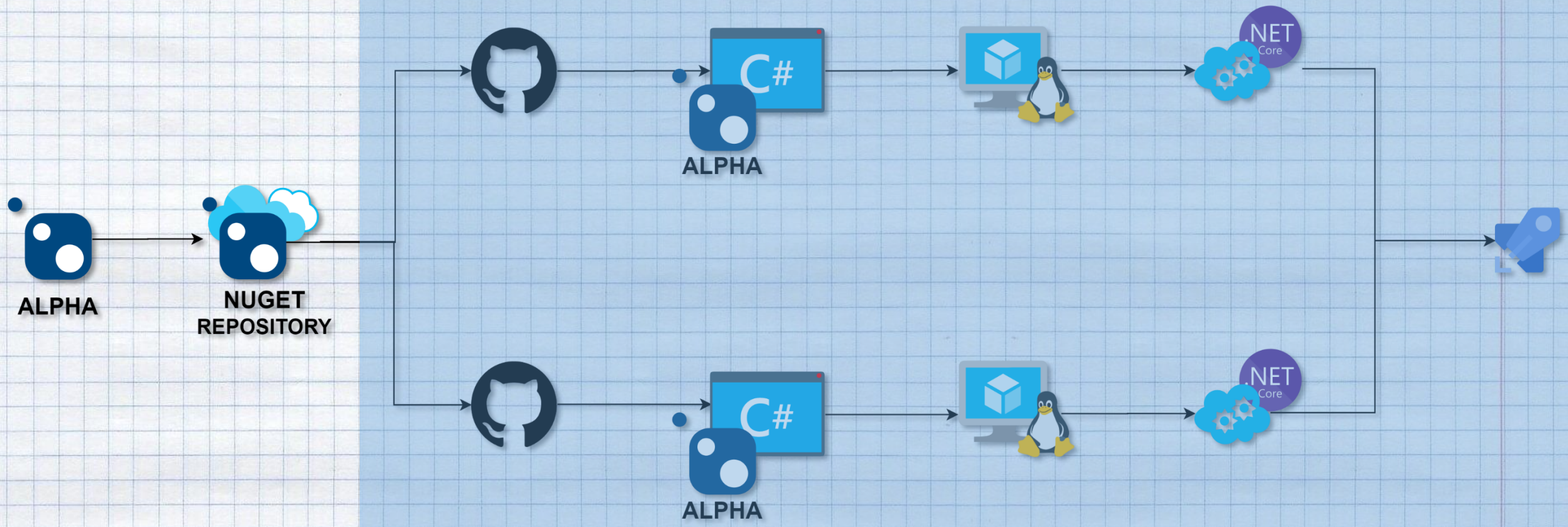
PUBLISH



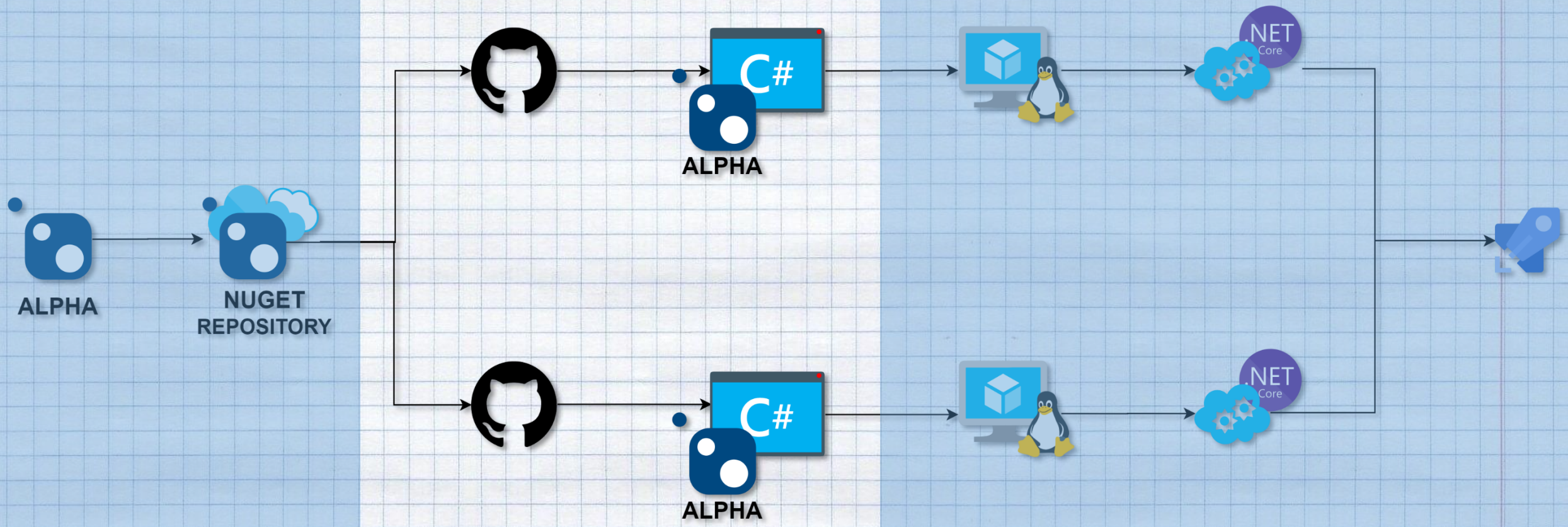
TESTY KONTRAKTÓW



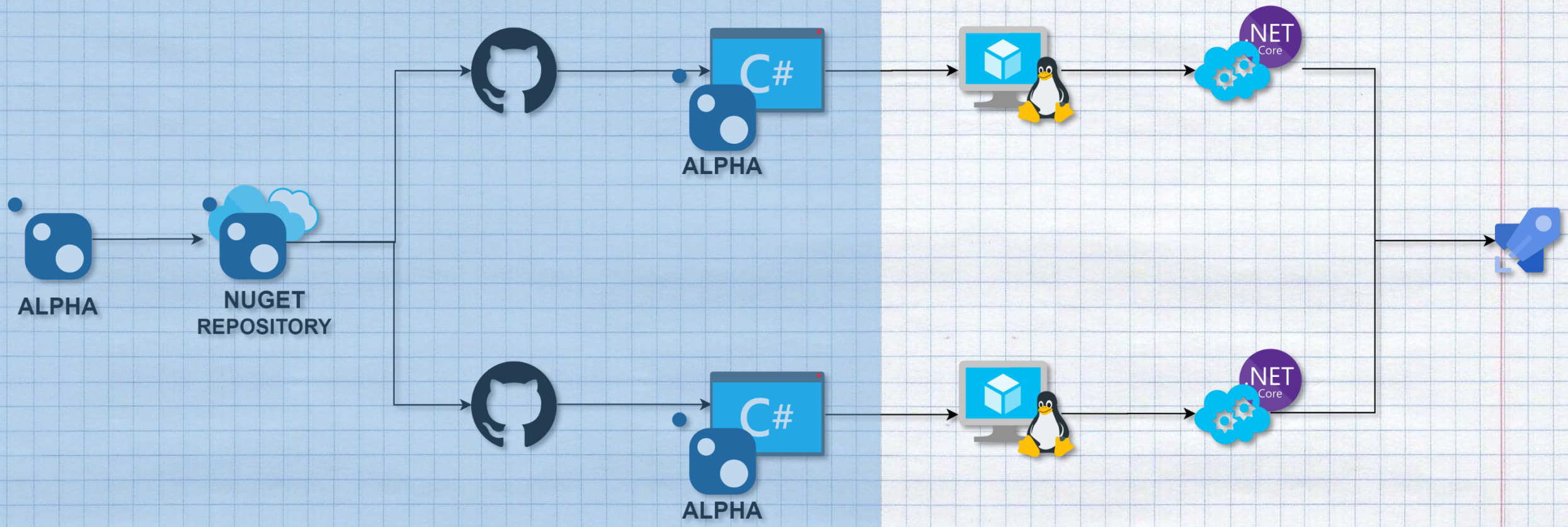
TESTY KONTRAKTÓW



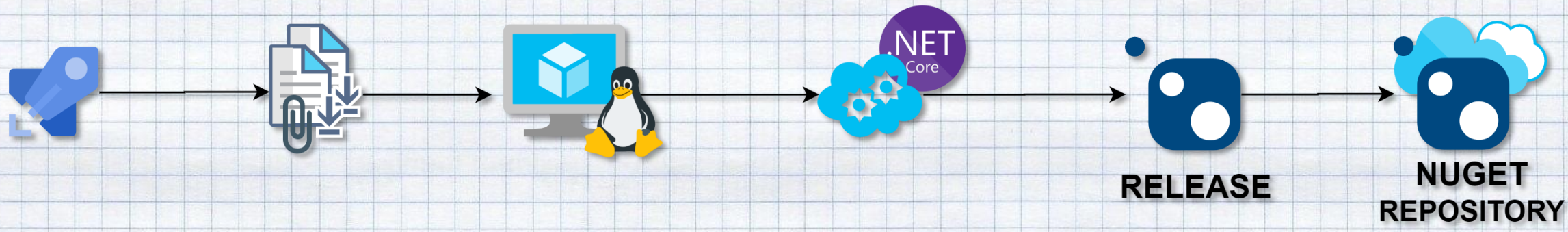
TESTY KONTRAKTÓW



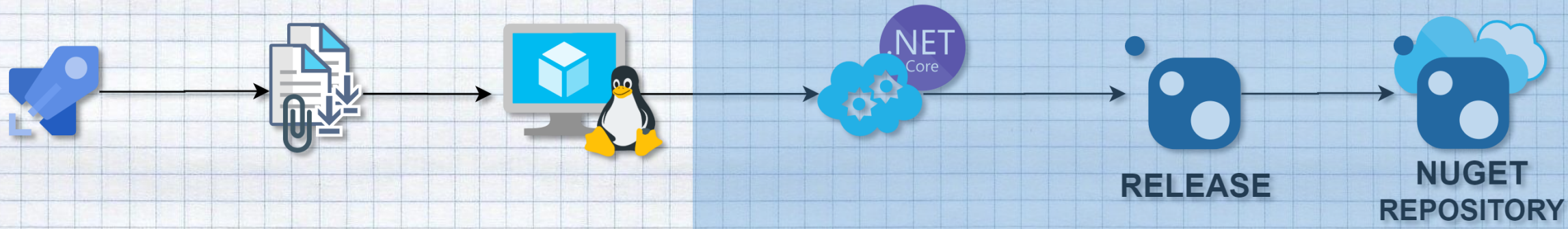
TESTY KONTRAKTÓW



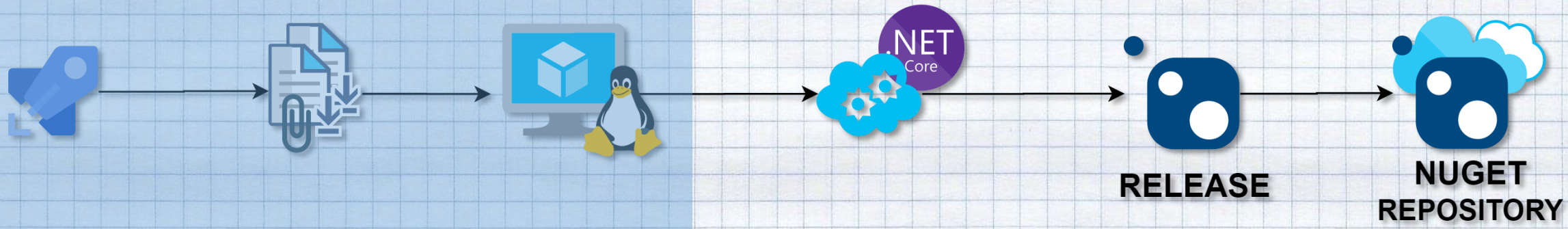
RELEASE



RELEASE



RELEASE



PRZYKŁAD

AZURE-PIPELINES.YAML

```yaml
##################################################
#   Triggers
##################################################
trigger:
  batch: true
  branches:
    include:
      - master

  paths:
    include:
    - Library/*
pr:
  branches:
    include:
      - master

  paths:
    include:
    - Library/*

##################################################
#   Resources
##################################################
resources:
  containers:
    - container: pg11
      image: ionx/postgres-plv8:11.1
      ports:
        - 5432:5432
```

Document 1/1 › pr:

AZURE-PIPELINES.YAML

```yaml
22  ######################################################
23  #   Resources
24  ######################################################
25  resources:
26    containers:
27      - container: pg11
28        image: ionx/postgres-plv8:11.1
29        ports:
30          - 5432:5432
31      - container: pg10
32        image: ionx/postgres-plv8:10.6
33        ports:
34          - 5432:5432
35      - container: pg9.6
36        image: mysticmind/postgres-plv8:9.6-1.4
37        ports:
38          - 5432:5432
39
40  ######################################################
41  #   Variables
42  ######################################################
43  variables:
44    # .NET build variables
45    dotnet_core_version: 3.1.x
46    # Postgres variables
47    pg_db: postgres_net_test
48    testing_database: "Host=localhost;Port=5432;Database=postgres_net_test;Username=postgres;Password=Password12!"
49    solution_path: Library/Postgres.NET.sln
50    # NuGet variables
51    major: 1
```

AZURE-PIPELINES.YAML

```yaml
##################################################
#   Variables
##################################################
variables:
  # .NET build variables
  dotnet_core_version: 3.1.x
  # Postgres variables
  pg_db: postgres_net_test
  testing_database: "Host=localhost;Port=5432;Database=postgres_net_test;Username=postgres;Password=Password12!"
  solution_path: Library/Postgres.NET.sln
  # NuGet variables
  major: 1
  minor: 0
  patch: $[counter(variables['minor'], 0)] #this will reset when we bump minor
  NugetVersion: $(major).$(minor).$(patch)
  AlphaNugetVersion: $(major).$(minor).$(patch)-alpha.1



##################################################
#   JOBS
##################################################
jobs:
  ##################################################
  #   JOB 1: Build Source Code
  ##################################################
  - template: jobs/job_1_build_with_multiple_os.yml

  ##################################################
  #   JOB 2: Test with different PG versions
```

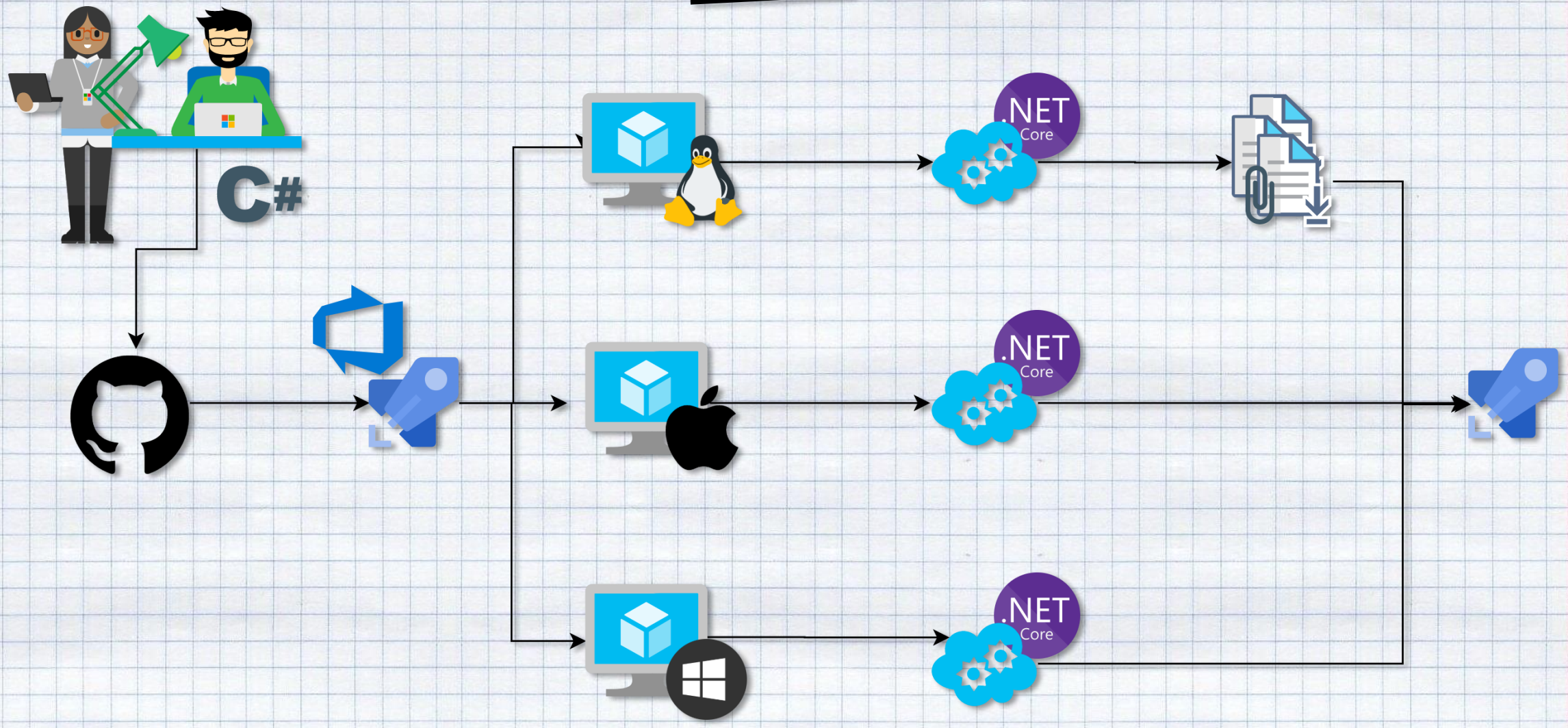
AZURE-PIPELINES.YAML

```yaml
##################################################
#   JOBS
##################################################
jobs:
  ##################################################
  #   JOB 1: Build Source Code
  ##################################################
  - template: jobs/job_1_build_with_multiple_os.yml

  ##################################################
  #   JOB 2: Test with different PG versions
  ##################################################
  - template: jobs/job_2_test_with_different_postgres_versions.yml

  ##################################################
  #   JOB 3: Publish Pre-release NuGet
  ##################################################
  - template: jobs/job_publish_nuget.yml
    parameters:
      name: Publish_Prerelease_NuGet
      dependsOn: Test
      nugetVersionEnv: 'AlphaNugetVersion'

  ##################################################
  #   JOB 4: Wait for NuGet to be published
  ##################################################
  - job: Wait
    dependsOn: Publish_Prerelease_NuGet
    pool: server
    steps:
```

AZURE-PIPELINES.YAML

```yaml
    ########################################################
    #   JOB 4: Wait for NuGet to be published
    ########################################################
    - job: Wait
      dependsOn: Publish_Prerelease_NuGet
      pool: server
      steps:
        - task: Delay@1
          inputs:
            delayForMinutes: '3'

    ########################################################
    #   JOB 5: Perform contract tests to verify
    #          if there are no breaking change
    ########################################################
    - template: jobs/job_contract_tests_with_client_apps.yml

    ########################################################
    #   JOB 6: Publish NuGet
    ########################################################
    - template: jobs/job_publish_nuget.yml
      parameters:
        name: Publish_NuGet
        dependsOn: Contract_tests
        nugetVersionEnv: 'NugetVersion'
        condition: and(succeeded(), eq(variables['Build.SourceBranch'], 'refs/heads/master'))
```

BUILD

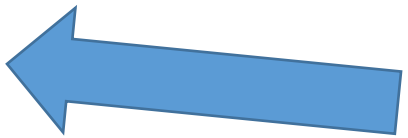


SCHEMAUPDATER.CS

```
1   using ...
4
5   namespace Postgres.NET
6   {
        1 usage    Oskar Dudycz
7       public class SchemaUpdater: IDisposable
8       {
9           private readonly NpgsqlConnection databaseConnection;
10
            1 usage    Oskar Dudycz
11          public SchemaUpdater(NpgsqlConnection databaseConnection)
12          {
13              this.databaseConnection = databaseConnection;
14          }
15
            1 usage    Oskar Dudycz
16          public int CreateTable(string tableName, string idColumnName)
17          {
18              return databaseConnection.Execute(
19                  sql: $"CREATE TABLE {tableName} ( {idColumnName} serial PRIMARY KEY );");
20          }
21
            1 usage    Oskar Dudycz
22          public void Dispose()
23          {
24              databaseConnection.Dispose();
25          }
26      }
27  }
28
```

AZURE-PIPELINES.YAML

```yaml
##################################################
#   Variables
##################################################
variables:
  # .NET build variables
  dotnet_core_version: 3.1.x
  # Postgres variables
  pg_db: postgres_net_test
  testing_database: "Host=localhost;Port=5432;Database=postgres_net_test;Username=postgres;Password=Password12!"
  solution_path: Library/Postgres.NET.sln
  # NuGet variables
  major: 1
  minor: 0
  patch: $[counter(variables['minor'], 0)] #this will reset when we bump minor
  NugetVersion: $(major).$(minor).$(patch)
  AlphaNugetVersion: $(major).$(minor).$(patch)-alpha.1



##################################################
#   JOBS
##################################################
jobs:
  ##################################################
  #   JOB 1: Build Source Code
  ##################################################
  - template: jobs/job_1_build_with_multiple_os.yml

  ##################################################
  #   JOB 2: Test with different PG versions
```



JOB_1_BUILD_WITH_MULTIPLE_OS.YAML

```yaml
parameters:

  - name: buildConfig
    default: Release

jobs:
  ################################################
  #   JOB 1: Build Source Code
  ################################################
  - job: Build
    strategy:
      ################################################
      #  Matrix build for:
      #  Linux, MacOS, Windows Server
      ################################################
      matrix:
        linux:
          imageName: 'ubuntu-16.04'
        mac:
          imageName: 'macos-10.14'
        windows:
          imageName: 'vs2017-win2016'
    pool:
      ################################################
      # Set VM image for the build
      ################################################
      vmImage: $(imageName)
    steps:
      ################################################
      # Step 1: Checkout
```

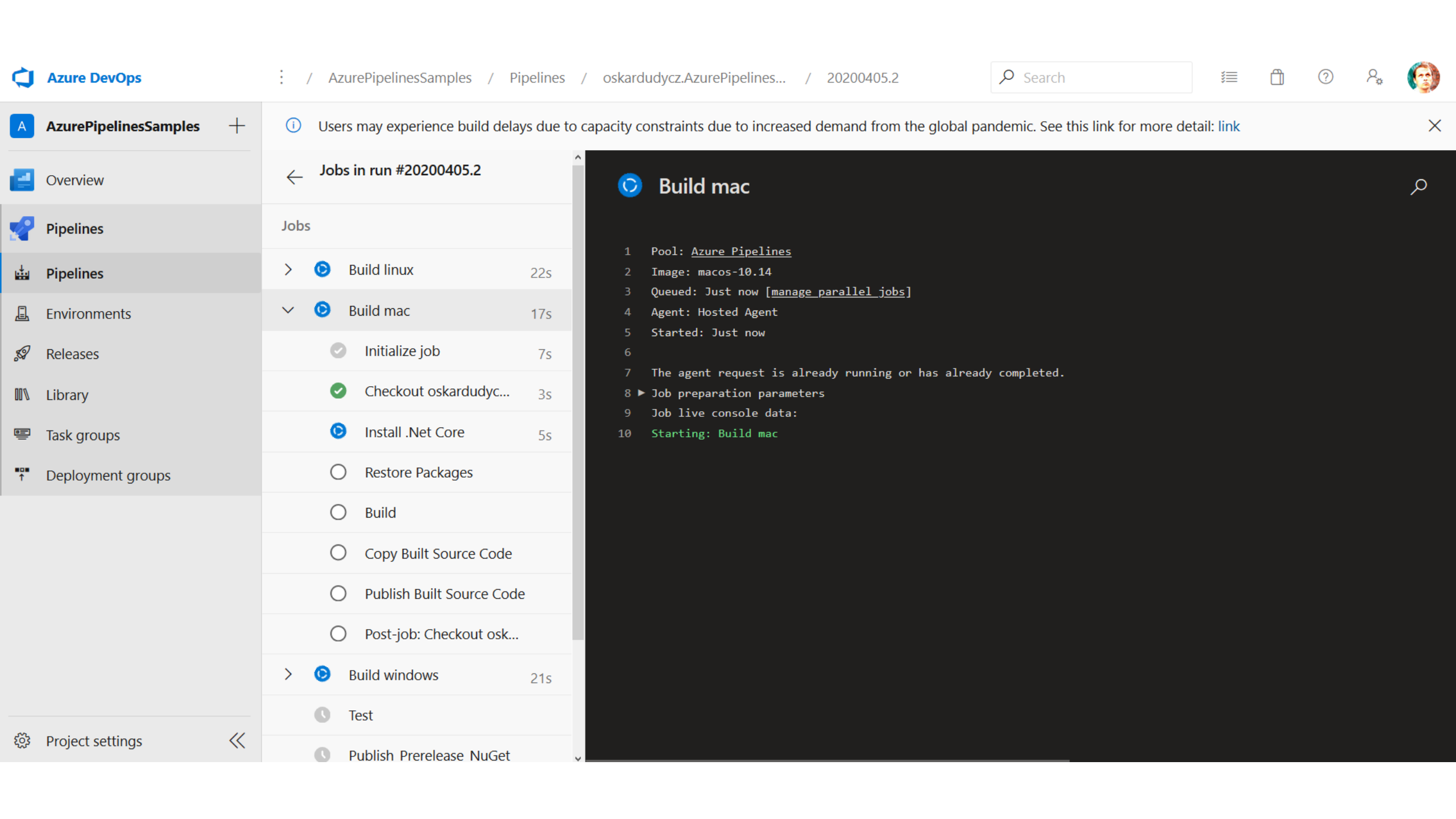
JOB_1_BUILD_WITH_MULTIPLE_OS.YAML

```yaml
steps:
  ##############################################
  # Step 1: Checkout
  ##############################################
  - checkout: self
  ##############################################
  # Step 2: Install .NET Core
  ##############################################
  - task: UseDotNet@2
    displayName: Install .Net Core
    inputs:
      version: $(dotnet_core_version)
      performMultiLevelLookup: true

  ##############################################
  # Step 2: Restore packages
  ##############################################
  - task: DotNetCoreCLI@2
    displayName: Restore Packages
    inputs:
      command: restore
      projects: $(solution_path)

  ##############################################
  # Step 3: Build
  ##############################################
  - task: DotNetCoreCLI@2
    displayName: Build
    inputs:
      command: build
```

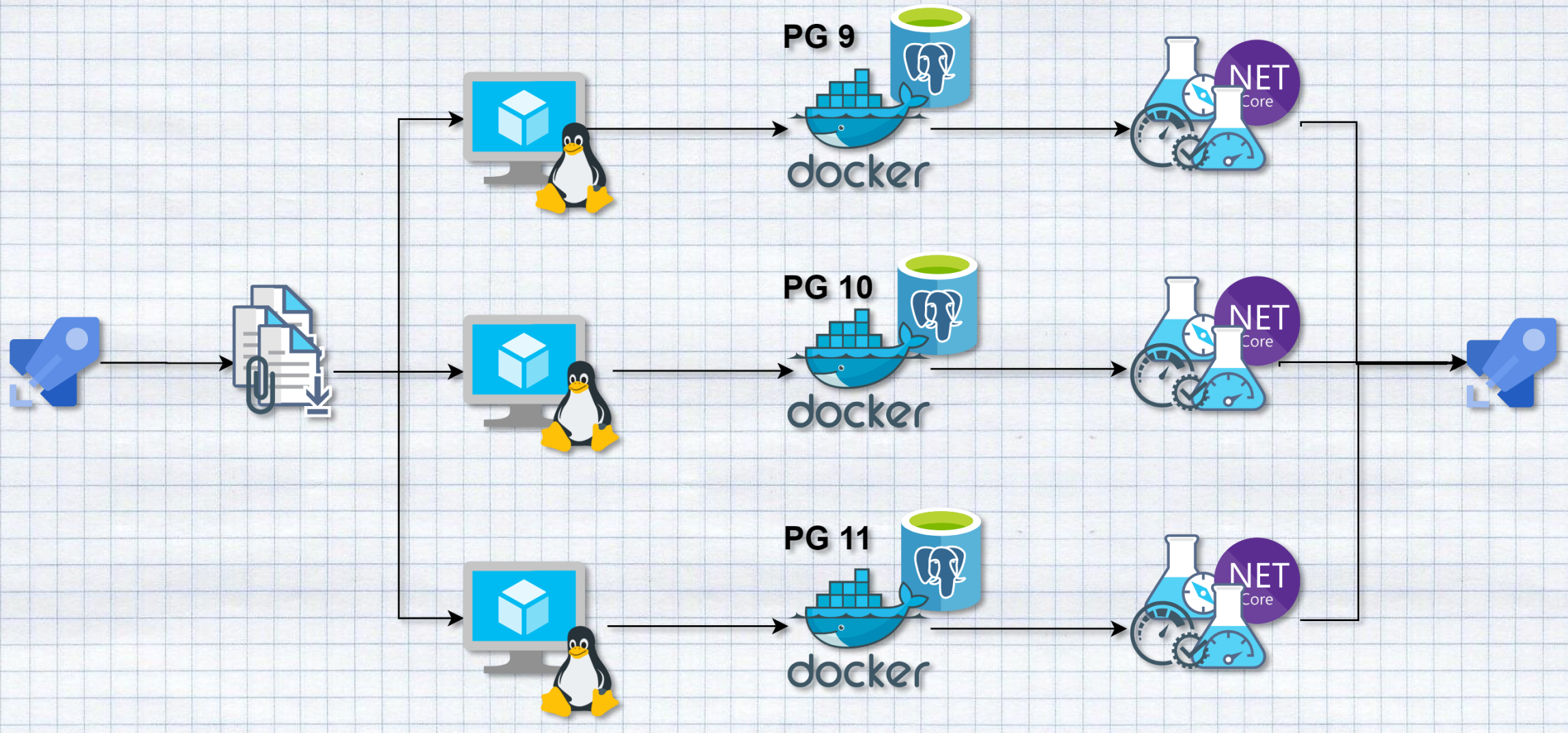
JOB_1_BUILD_WITH_MULTIPLE_OS.YAML

```yaml
##############################################
# Step 3: Build
##############################################
- task: DotNetCoreCLI@2
  displayName: Build
  inputs:
    command: build
    projects: $(solution_path)
    arguments: '--configuration ${{ parameters.buildConfig }} --no-restore'

##############################################
# Step 4: Copy built source code
#         to build artifacts
##############################################
- task: CopyFiles@2
  displayName: Copy Built Source Code
  condition: eq( variables['Agent.OS'], 'Linux' )
  inputs:
    contents:
      **
      !.git/**/*
    targetFolder: $(Build.ArtifactStagingDirectory)

##############################################
# Step 4: Publish build artifacts
##############################################
- task: PublishBuildArtifacts@1
  displayName: Publish Built Source Code
  condition: eq( variables['Agent.OS'], 'Linux' )
  inputs:
```

```
- task: DotNetCoreCLI@2
  displayName: Build
  inputs:
    command: build
    projects: $(solution_path)
    arguments: '--configuration ${{ parameters.buildConfig }} --no-restore'

################################################
# Step 4: Copy built source code
#         to build artifacts
################################################
- task: CopyFiles@2
  displayName: Copy Built Source Code
  condition: eq( variables['Agent.OS'], 'Linux' )
  inputs:
    contents:
      **
      !.git/**/*
    targetFolder: $(Build.ArtifactStagingDirectory)

################################################
# Step 4: Publish build artifacts
################################################
- task: PublishBuildArtifacts@1
  displayName: Publish Built Source Code
  condition: eq( variables['Agent.OS'], 'Linux' )
  inputs:
    pathToPublish: $(Build.ArtifactStagingDirectory)
    artifactName: BuiltSourceCode
```


Azure DevOps
AzurePipelinesSamples / Pipelines / oskardudycz.AzurePipelines... / 20200405.2
Search
AzurePipelinesSamples
Users may experience build delays due to capacity constraints due to increased demand from the global pandemic. See this link for more detail: link
Overview
Pipelines
Pipelines
Environments
Releases
Library
Task groups
Deployment groups
Project settings
Jobs in run #20200405.2
Jobs
Build linux 22s
Build mac 17s
Initialize job 7s
Checkout oskardudyc... 3s
Install .Net Core 5s
Restore Packages
Build
Copy Built Source Code
Publish Built Source Code
Post-job: Checkout osk...
Build windows 21s
Test
Publish Prerelease NuGet
Build mac
1 Pool: Azure Pipelines
2 Image: macos-10.14
3 Queued: Just now [manage parallel jobs]
4 Agent: Hosted Agent
5 Started: Just now
6
7 The agent request is already running or has already completed.
8 ▶ Job preparation parameters
9 Job live console data:
10 Starting: Build mac

TESTY

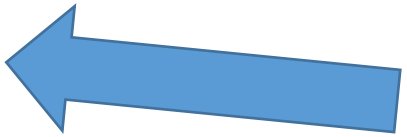

PG 9
docker
PG 10
docker
PG 11
docker
NET
Core
NET
Core
NET
Core

SchemaUpdaterTests.cs

```
1   using ...
4
5   namespace Postgres.NET.Tests
6   {
        Oskar Dudycz
7       public class SchemaUpdaterTests
8       {
9           private static readonly string ConnectionString
10              = Environment.GetEnvironmentVariable("testing_database")
11                ?? "Host=localhost;Port=5432;Database=postgres;Username=postgres;password=postgres";
12
13          [Fact]
            Oskar Dudycz
14          public void CreateTable_ShouldSucceed_ForNonEmptyTableName()
15          {
16              using var schemaUpdater = new SchemaUpdater(new NpgsqlConnection(ConnectionString));
17
18              Assert.NotEqual( expected: 0,  actual: schemaUpdater.CreateTable("testTable",  idColumnName: "id"));
19          }
20      }
21  }
22
```

AZURE-PIPELINES.YAML

```yaml
59  ##################################################
60  #   JOBS
61  ##################################################
62  jobs:
63    ##################################################
64    #   JOB 1: Build Source Code
65    ##################################################
66    - template: jobs/job_1_build_with_multiple_os.yml
67
68    ##################################################
69    #   JOB 2: Test with different PG versions
70    ##################################################
71    - template: jobs/job_2_test_with_different_postgres_versions.yml
72
73    ##################################################
74    #   JOB 3: Publish Pre-release NuGet
75    ##################################################
76    - template: jobs/job_publish_nuget.yml
77      parameters:
78        name: Publish_Prerelease_NuGet
79        dependsOn: Test
80        nugetVersionEnv: 'AlphaNugetVersion'
81
82    ##################################################
83    #   JOB 4: Wait for NuGet to be published
84    ##################################################
85    - job: Wait
86      dependsOn: Publish_Prerelease_NuGet
87      pool: server
88      steps:
```



JOB_2_TEST_WITH_DIFFERENT_POSTGRES_VERSIONS.YAML

```yaml
parameters:

  - name: buildConfig
    default: Release

jobs:
  ##################################################
  #   JOB 2: Test with different PG versions
  ##################################################
  - job: Test
    dependsOn: Build
    pool:
      ################################################
      # Use Linux VM image
      ################################################
      vmImage: 'ubuntu-16.04'
    strategy:
      ################################################
      #  Matrix build for
      #  Different PG Versions
      ################################################
      matrix:
        pg9.6:
          postgresService: pg9.6
        pg10:
          postgresService: pg10
        pg11:
          postgresService: pg11
    services:
      ################################################
```

JOB_2_TEST_WITH_DIFFERENT_POSTGRES_VERSIONS.YAML

```yaml
services:
  ################################################
  #  Start Postgres Docker image
  ################################################
  postgres: $[ variables['postgresService'] ]
steps:
  ################################################
  # Step 0: No checkout as we're getting
  #         built code from artifacts
  ################################################
  - checkout: none

  ################################################
  # Step 1: Get built source code
  #         from build artifacts
  ################################################
  - task: DownloadBuildArtifacts@0
    displayName: 'Download Built Source Code'
    inputs:
      artifactName: BuiltSourceCode
      downloadPath: $(Build.SourcesDirectory)

  ################################################
  # Step 2: Setup Postgres database
  #         for tests
  ################################################
  - script: |
      PG_CONTAINER_NAME=$(docker ps --filter expose=5432/tcp --format {{.Names}})
      docker exec $PG_CONTAINER_NAME psql -U postgres -c "create database $(pg_db);"
    displayName: Create test database
```

JOB_2_TEST_WITH_DIFFERENT_POSTGRES_VERSIONS.YAML

```yaml
##############################################
# Step 2: Setup Postgres database
#         for tests
##############################################
- script: |
    PG_CONTAINER_NAME=$(docker ps --filter expose=5432/tcp --format {{.Names}})
    docker exec $PG_CONTAINER_NAME psql -U postgres -c "create database $(pg_db);"
  displayName: Create test database

##############################################
# Step 3: Install .NET Core
##############################################
- task: UseDotNet@2
  displayName: Install .Net Core
  inputs:
    version: $(dotnet_core_version)
    performMultiLevelLookup: true

##############################################
# Step 4: Run tests
##############################################
- task: DotNetCoreCLI@2
  displayName: Test
  inputs:
    command: test
    nobuild: true
    projects: '$(Build.SourcesDirectory)/BuiltSourceCode/$(solution_path)'
    arguments: '--configuration ${{ parameters.buildConfig }}'

```

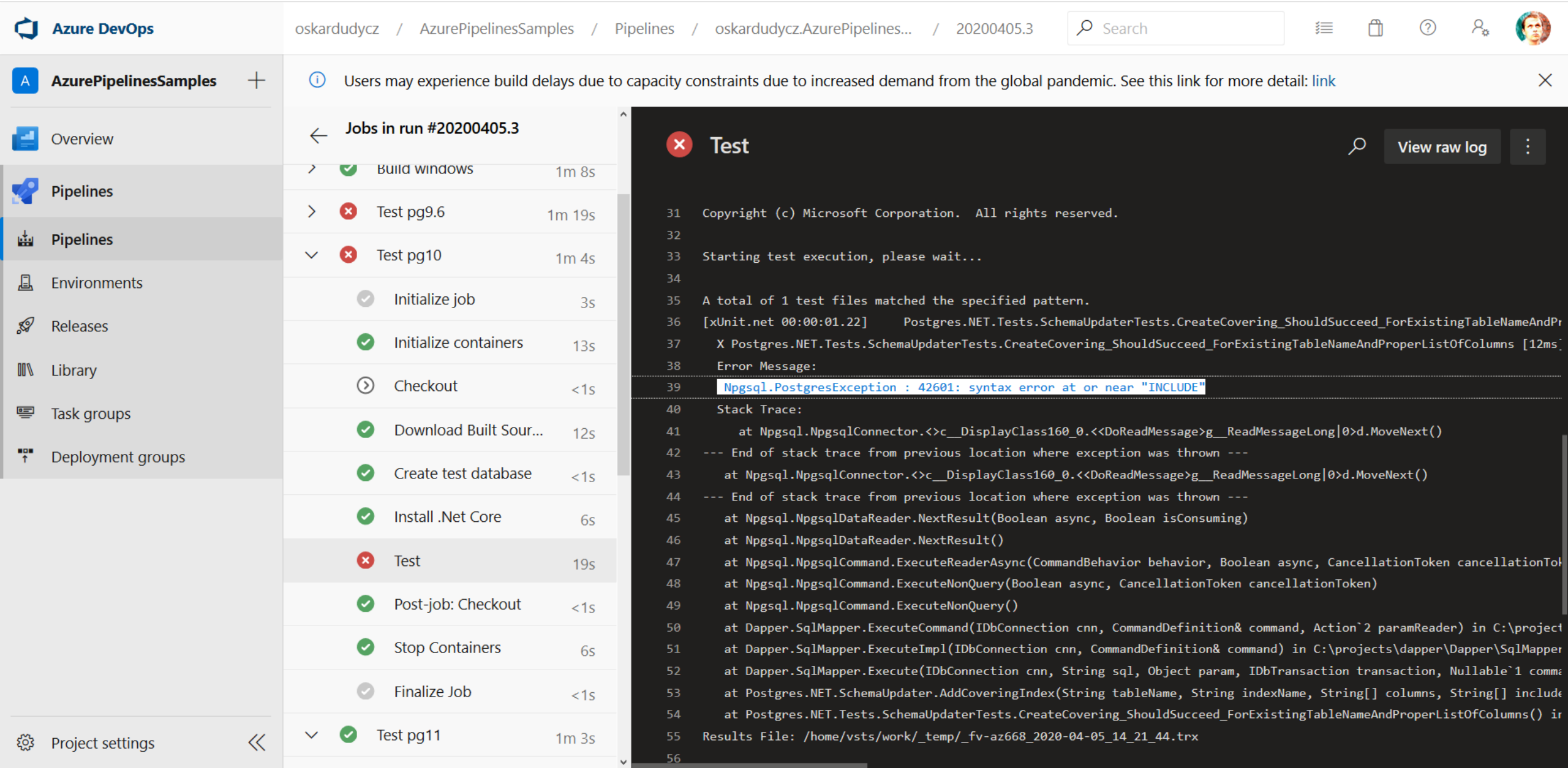
SCHEMAUPDATER.CS

```
2 usages  Oskar Dudycz
18  public int CreateTable(string tableName, params string [] columns)
19  {
20      return databaseConnection.Execute(
21          sql: $@"CREATE TABLE {tableName} (
22              { Zip( firstColumn: "id serial PRIMARY KEY", columns) }
23          );");
24  }
25
1 usage  Oskar Dudycz
26  public int AddCoveringIndex(string tableName, string indexName, string[] columns, string [] include)
27  {
28      return databaseConnection.Execute(
29          sql: $"CREATE INDEX {indexName} ON {tableName}({Zip(columns)}) INCLUDE ({Zip(include)});");
30  }
31
1 usage  Oskar Dudycz
32  private string Zip(string firstColumn, string [] items)
33  {
34      return Zip( items: new []{ firstColumn }.Union(items ?? Array.Empty<string>()));
35  }
36
3 usages  Oskar Dudycz
37  private string Zip(IEnumerable<string> items)
38  {
39      return items?.Count() > 0 ?
40          string.Join( separator: ',', items)
41          : string.Empty;
42  }
```

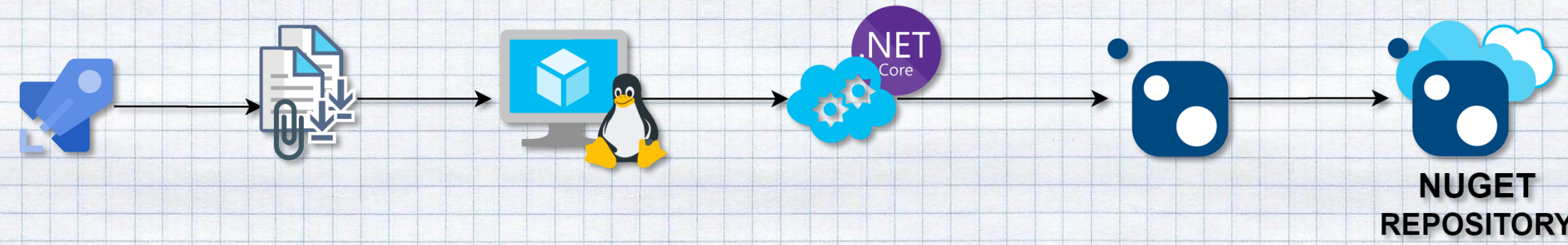
SCHEMAUPDATERTESTS.CS

```
13
14      [Fact]
        Oskar Dudycz
15      public void CreateTable_ShouldSucceed_ForNonEmptyTableName()
16      {
17          using var schemaUpdater = new SchemaUpdater(new NpgsqlConnection(ConnectionString));
18
19          Assert.NotEqual( expected: 0,  actual: schemaUpdater.CreateTable("testTable"));
20      }
21
22      [Fact]
        Oskar Dudycz
23      public void CreateCovering_ShouldSucceed_ForExistingTableNameAndProperListOfColumns()
24      {
25          using var schemaUpdater = new SchemaUpdater(new NpgsqlConnection(ConnectionString));
26
27          var tableName = "testTableCoveringIndex";
28          var columnNames = new [] { "firstColumn", "secondColumn", "thirdColumn" };
29          var columns :string[]  = columnNames.Select(column :string  => $"{column} int").ToArray();
30          var include = new [] { "thirdColumn" };
31
32          schemaUpdater.CreateTable(tableName, columns);
33
34          Assert.NotEqual( expected: 0,  actual: schemaUpdater.AddCoveringIndex(tableName,  indexName: $"{tableName}_covering_idx",  colum
35      }
36  }
37  }
38
```

Tests › SchemaUpdaterTests

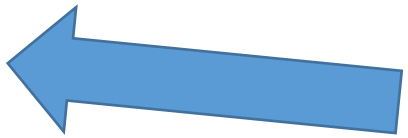

Azure DevOps
oskardudycz / AzurePipelinesSamples / Pipelines / oskardudycz.AzurePipelines... / 20200405.3
Search
AzurePipelinesSamples
Overview
Pipelines
Pipelines
Environments
Releases
Library
Task groups
Deployment groups
Project settings
Users may experience build delays due to capacity constraints due to increased demand from the global pandemic. See this link for more detail: link
Jobs in run #20200405.3
Build windows 1m 8s
Test pg9.6 1m 19s
Test pg10 1m 4s
Initialize job 3s
Initialize containers 13s
Checkout <1s
Download Built Sour... 12s
Create test database <1s
Install .Net Core 6s
Test 19s
Post-job: Checkout <1s
Stop Containers 6s
Finalize Job <1s
Test pg11 1m 3s
Test
View raw log
31 Copyright (c) Microsoft Corporation. All rights reserved.
32
33 Starting test execution, please wait...
34
35 A total of 1 test files matched the specified pattern.
36 [xUnit.net 00:00:01.22] Postgres.NET.Tests.SchemaUpdaterTests.CreateCovering_ShouldSucceed_ForExistingTableNameAndPr
37 X Postgres.NET.Tests.SchemaUpdaterTests.CreateCovering_ShouldSucceed_ForExistingTableNameAndProperListOfColumns [12ms
38 Error Message:
39 Npgsql.PostgresException : 42601: syntax error at or near "INCLUDE"
40 Stack Trace:
41 at Npgsql.NpgsqlConnector.<>c__DisplayClass160_0.<<DoReadMessage>g__ReadMessageLong|0>d.MoveNext()
42 --- End of stack trace from previous location where exception was thrown ---
43 at Npgsql.NpgsqlConnector.<>c__DisplayClass160_0.<<DoReadMessage>g__ReadMessageLong|0>d.MoveNext()
44 --- End of stack trace from previous location where exception was thrown ---
45 at Npgsql.NpgsqlDataReader.NextResult(Boolean async, Boolean isConsuming)
46 at Npgsql.NpgsqlDataReader.NextResult()
47 at Npgsql.NpgsqlCommand.ExecuteReaderAsync(CommandBehavior behavior, Boolean async, CancellationToken cancellationTok
48 at Npgsql.NpgsqlCommand.ExecuteNonQuery(Boolean async, CancellationToken cancellationToken)
49 at Npgsql.NpgsqlCommand.ExecuteNonQuery()
50 at Dapper.SqlMapper.ExecuteCommand(IDbConnection cnn, CommandDefinition& command, Action`2 paramReader) in C:\project
51 at Dapper.SqlMapper.ExecuteImpl(IDbConnection cnn, CommandDefinition& command) in C:\projects\dapper\Dapper\SqlMapper
52 at Dapper.SqlMapper.Execute(IDbConnection cnn, String sql, Object param, IDbTransaction transaction, Nullable`1 comma
53 at Postgres.NET.SchemaUpdater.AddCoveringIndex(String tableName, String indexName, String[] columns, String[] include
54 at Postgres.NET.Tests.SchemaUpdaterTests.CreateCovering_ShouldSucceed_ForExistingTableNameAndProperListOfColumns() ir
55 Results File: /home/vsts/work/_temp/_fv-az668_2020-04-05_14_21_44.trx
56

PUBLISH



AZURE-PIPELINES.YAML

```yaml
#####################################################
#   JOBS
#####################################################
jobs:
  #####################################################
  #   JOB 1: Build Source Code
  #####################################################
  - template: jobs/job_1_build_with_multiple_os.yml

  #####################################################
  #   JOB 2: Test with different PG versions
  #####################################################
  - template: jobs/job_2_test_with_different_postgres_versions.yml

  #####################################################
  #   JOB 3: Publish Pre-release NuGet
  #####################################################
  - template: jobs/job_publish_nuget.yml
    parameters:
      name: Publish_Prerelease_NuGet
      dependsOn: Test
      nugetVersionEnv: 'AlphaNugetVersion'

  #####################################################
  #   JOB 4: Wait for NuGet to be published
  #####################################################
  - job: Wait
    dependsOn: Publish_Prerelease_NuGet
    pool: server
    steps:
```



Document 1/1 › pr:

JOB_PUBLISH_NUGET.YAML

```yaml
parameters:

  - name: name
    type: string
    default: 'Publish_NuGet'

  - name: dependsOn
    default: any

  - name: nugetVersionEnv
    type: string

  - name: condition
    default: succeeded()

  - name: buildConfig
    default: Release

jobs:
  ##################################################
  #   JOB: Publish NuGet
  ##################################################
  - job: ${{ parameters.name }}
    dependsOn: ${{ parameters.dependsOn }}
    condition: ${{ parameters.condition }}
    pool:
      vmImage: 'ubuntu-16.04'

    steps:
      ##################################################
```

Document 1/1 › parameters: › Item 5/5 › name: › buildConfig

JOB_PUBLISH_NUGET.YAML

```yaml
jobs:
  ################################################
  #   JOB: Publish NuGet
  ################################################
  - job: ${{ parameters.name }}
    dependsOn: ${{ parameters.dependsOn }}
    condition: ${{ parameters.condition }}
    pool:
      vmImage: 'ubuntu-16.04'

    steps:
      ################################################
      # Step 0: No checkout as we're getting
      #         built code from artifacts
      ################################################
      - checkout: none

      ################################################
      # Step 1: Get built source code
      #         from build artifacts
      ################################################
      - task: DownloadBuildArtifacts@0
        displayName: 'Download Built Source Code'
        inputs:
          artifactName: BuiltSourceCode
          downloadPath: $(Build.SourcesDirectory)

      ################################################
      # Step 2: Create alpha NuGet package
```

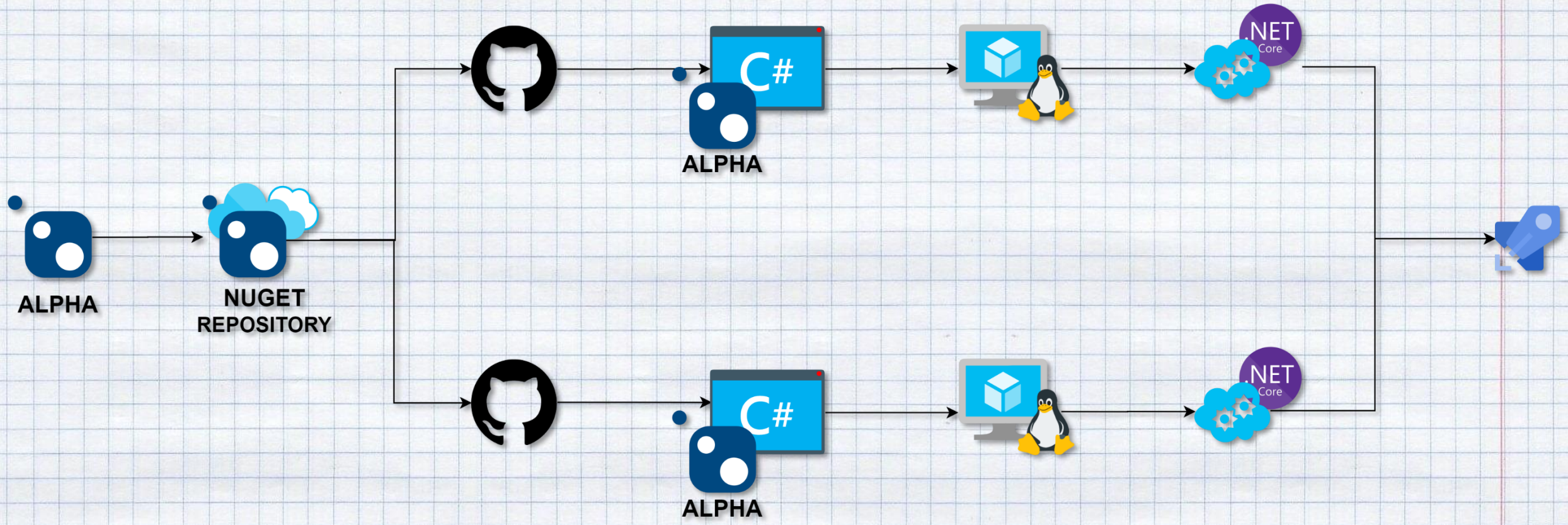
JOB_PUBLISH_NUGET.YAML

```yaml
steps:
  ################################################
  # Step 0: No checkout as we're getting
  #         built code from artifacts
  ################################################
  - checkout: none

  ################################################
  # Step 1: Get built source code
  #         from build artifacts
  ################################################
  - task: DownloadBuildArtifacts@0
    displayName: 'Download Built Source Code'
    inputs:
      artifactName: BuiltSourceCode
      downloadPath: $(Build.SourcesDirectory)

  ################################################
  # Step 2: Create alpha NuGet package
  ################################################
  - task: DotNetCoreCLI@2
    displayName: Pack
    inputs:
      command: pack
      projects: '$(Build.SourcesDirectory)/BuiltSourceCode/$(solution_path)'
      arguments: '--configuration ${{ parameters.buildConfig }}'
      packDestination: '$(Build.ArtifactStagingDirectory)'
      versioningScheme: 'byEnvVar'
      versionEnvVar: ${{ parameters.nugetVersionEnv }}

```

JOB_PUBLISH_NUGET.YAML

```yaml
- task: DownloadBuildArtifacts@0
  displayName: 'Download Built Source Code'
  inputs:
    artifactName: BuiltSourceCode
    downloadPath: $(Build.SourcesDirectory)

##############################################
# Step 2: Create alpha NuGet package
##############################################
- task: DotNetCoreCLI@2
  displayName: Pack
  inputs:
    command: pack
    projects: '$(Build.SourcesDirectory)/BuiltSourceCode/$(solution_path)'
    arguments: '--configuration ${{ parameters.buildConfig }}'
    packDestination: '$(Build.ArtifactStagingDirectory)'
    versioningScheme: 'byEnvVar'
    versionEnvVar: ${{ parameters.nugetVersionEnv }}

##############################################
# Step 3: Publish alpha NuGet package
##############################################
- task: NuGetCommand@2
  displayName: Publish packages to NuGet
  inputs:
    command: 'push'
    nuGetFeedType: 'external'
    packagesToPush: '$(Build.ArtifactStagingDirectory)/**/*.nupkg'
    publishFeedCredentials: 'NuGet'

```

TESTY KONTRAKTÓW

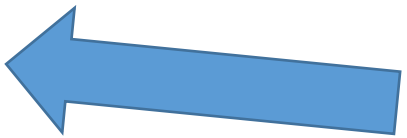


AZURE-PIPELINES.YAML

```yaml
######################################################
#   JOB 4: Wait for NuGet to be published
######################################################
- job: Wait
  dependsOn: Publish_Prerelease_NuGet
  pool: server
  steps:
    - task: Delay@1
      inputs:
        delayForMinutes: '3'

######################################################
#   JOB 5: Perform contract tests to verify
#          if there are no breaking change
######################################################
- template: jobs/job_contract_tests_with_client_apps.yml

######################################################
#   JOB 6: Publish NuGet
######################################################
- template: jobs/job_publish_nuget.yml
  parameters:
    name: Publish_NuGet
    dependsOn: Contract_tests
    nugetVersionEnv: 'NugetVersion'
    condition: and(succeeded(), eq(variables['Build.SourceBranch'], 'refs/heads/master'))
```

AZURE-PIPELINES.YAML

```yaml
########################################################
#   JOB 4: Wait for NuGet to be published
########################################################
- job: Wait
  dependsOn: Publish_Prerelease_NuGet
  pool: server
  steps:
    - task: Delay@1
      inputs:
        delayForMinutes: '3'

########################################################
#   JOB 5: Perform contract tests to verify
#          if there are no breaking change
########################################################
- template: jobs/job_contract_tests_with_client_apps.yml

########################################################
#   JOB 6: Publish NuGet
########################################################
- template: jobs/job_publish_nuget.yml
  parameters:
    name: Publish_NuGet
    dependsOn: Contract_tests
    nugetVersionEnv: 'NugetVersion'
    condition: and(succeeded(), eq(variables['Build.SourceBranch'], 'refs/heads/master'))
```



JOB_CONTRACT_TESTS_WITH_CLIENT_APPS.YAML

```yaml
parameters:

  - name: buildConfig
    default: Release

jobs:
  - job: Contract_tests
    dependsOn: Wait
    strategy:
      ################################################
      #  Matrix build for two client projects
      #  that are using our NuGet package
      ################################################
      matrix:
        client1:
          repositoryUrl: 'https://github.com/oskardudycz/AzurePipelinesSamples'
          clientProjectPath: 'Library/Clients/Client1/Client1/Client1.csproj'
          clientSolutionPath: 'Library/Clients/Client1/Client1.sln'
        client2:
          repositoryUrl: 'https://github.com/oskardudycz/AzurePipelinesSamples'
          clientProjectPath: 'Library/Clients/Client2/Client2/Client2.csproj'
          clientSolutionPath: 'Library/Clients/Client2/Client2.sln'
    pool:
      ################################################
      # Use Linux VM image
      ################################################
      vmImage: 'ubuntu-16.04'
    steps:
      ################################################
      # Step 0: No checkout as we're getting
```

JOB_CONTRACT_TESTS_WITH_CLIENT_APPS.YAML

```yaml
steps:
  ################################################
  # Step 0: No checkout as we're getting
  #         other repositories via checkout scrpt
  ################################################
  - checkout: none
  ################################################
  # Step 1: Clone client project repository
  ################################################
  - script: |
     git clone $(repositoryUrl) .
  ################################################
  # Step 2: Install .NET Core
  ################################################
  - task: UseDotNet@2
    displayName: Install .Net Core
    inputs:
      version: $(dotnet_core_version)
      performMultiLevelLookup: true

  ################################################
  # Step 3: Update library NuGet to alpha version
  #         published in previous steps
  ################################################
  - script: |
     dotnet add $(Build.SourcesDirectory)/$(clientProjectPath) package Postgres.NET -v $(AlphaNugetVersion)

  ################################################
  # Step 4: Restore packages to
  ################################################
```

JOB_CONTRACT_TESTS_WITH_CLIENT_APPS.YAML

```yaml
47
48  ##############################################
49  # Step 3: Update library NuGet to alpha version
50  #         published in previous steps
51  ##############################################
52  - script: |
53    dotnet add $(Build.SourcesDirectory)/$(clientProjectPath) package Postgres.NET -v $(AlphaNugetVersion)
54
55  ##############################################
56  # Step 4: Restore packages to
57  ##############################################
58  - task: DotNetCoreCLI@2
59    displayName: Restore Packages
60    inputs:
61      command: restore
62      projects: $(clientSolutionPath)
63
64  ##############################################
65  # Step 5: Build client library
66  ##############################################
67  - task: DotNetCoreCLI@2
68    displayName: Build
69    inputs:
70      command: build
71      projects: $(clientSolutionPath)
72      arguments: '--configuration ${{ parameters.buildConfig }} --no-restore'
73
```

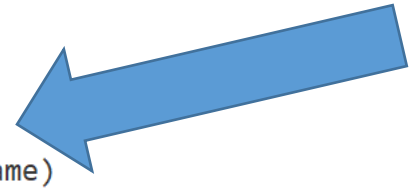
TESTPOSTGRESUSAGE.CS

```
1   using ...
3
4   namespace Client1
5   {
        Oskar Dudycz
6       public class TestPostgresUsage
7       {
8           private readonly SchemaUpdater schemaUpdater;
9
            Oskar Dudycz
10          public TestPostgresUsage(string connectionString)
11          {
12              schemaUpdater = new SchemaUpdater(new NpgsqlConnection(connectionString));
13          }
            Oskar Dudycz
14          public int Use()
15          {
16              return schemaUpdater.CreateTable("test");
17          }
18      }
19  }
20
```

Client1

SCHEMAUPDATER.CS

```
namespace Postgres.NET
{
    1 usage    Oskar Dudycz
    public class SchemaUpdater: IDisposable
    {
        private readonly NpgsqlConnection databaseConnection;

        1 usage    Oskar Dudycz
        public SchemaUpdater(NpgsqlConnection databaseConnection)
        {
            this.databaseConnection = databaseConnection;
        }

        1 usage    Oskar Dudycz
        public int CreateTable(string tableName, string idColumnName)
        {
            return databaseConnection.Execute(
                sql: $"CREATE TABLE {tableName} ( {idColumnName} serial PRIMARY KEY );");
        }

        1 usage    Oskar Dudycz
        public void Dispose()
        {
            databaseConnection.Dispose();
        }
    }
}
```



NET

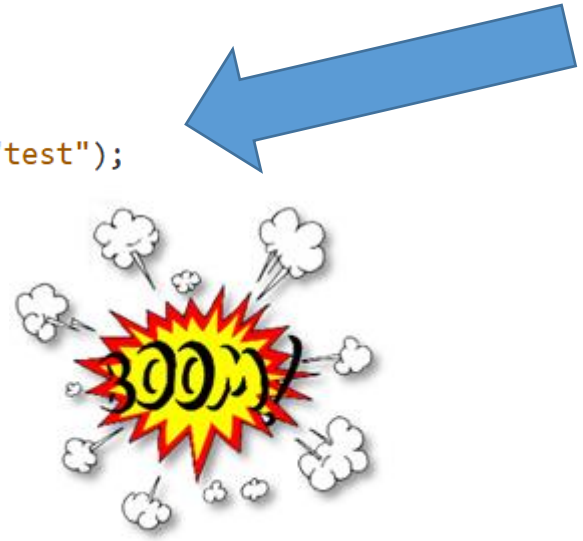
SCHEMAUPDATERTESTS.CS

```
1   using ...
4
5   namespace Postgres.NET.Tests
6   {
        Oskar Dudycz
7       public class SchemaUpdaterTests
8       {
9           private static readonly string ConnectionString
10              = Environment.GetEnvironmentVariable("testing_database")
11                ?? "Host=localhost;Port=5432;Database=postgres;Username=postgres;password=postgres";
12
13          [Fact]
            Oskar Dudycz
14          public void CreateTable_ShouldSucceed_ForNonEmptyTableName()
15          {
16              using var schemaUpdater = new SchemaUpdater(new NpgsqlConnection(ConnectionString));
17
18              Assert.NotEqual( expected: 0,  actual: schemaUpdater.CreateTable("testTable",  idColumnName: "id"));
19          }
20      }
21  }
22
```

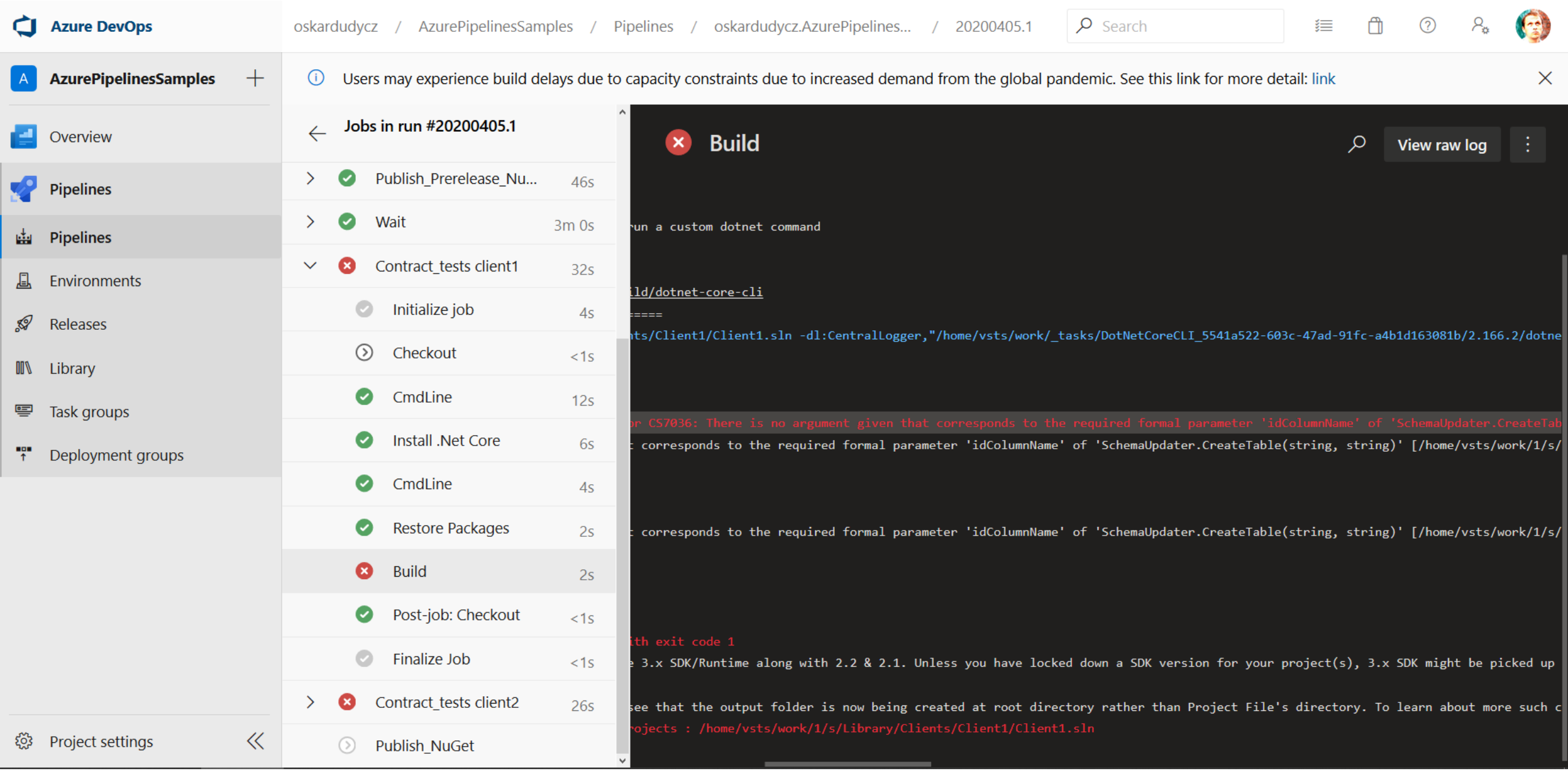
Tests

TESTPOSTGRESUSAGE.CS

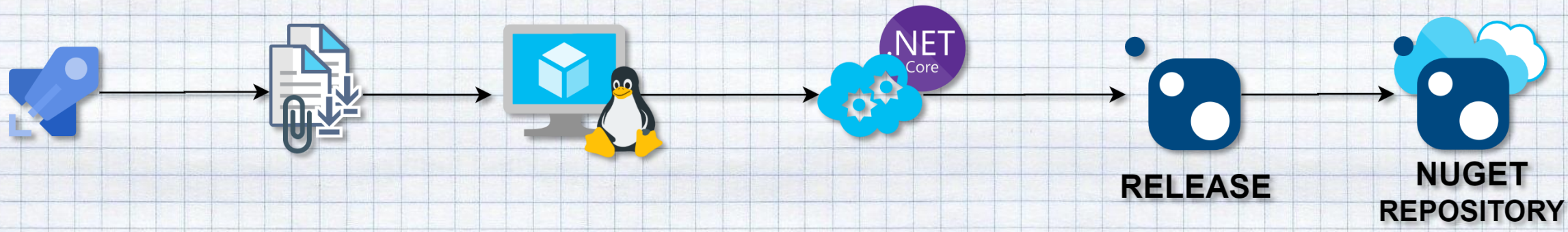
```
1   using ...
3
4   namespace Client1
5   {
        Oskar Dudycz
6       public class TestPostgresUsage
7       {
8           private readonly SchemaUpdater schemaUpdater;
9
            Oskar Dudycz
10          public TestPostgresUsage(string connectionString)
11          {
12              schemaUpdater = new SchemaUpdater(new NpgsqlConnection(connectionString));
13          }
            Oskar Dudycz
14          public int Use()
15          {
16              return schemaUpdater.CreateTable("test");
17          }
18      }
19  }
20
```



Client1

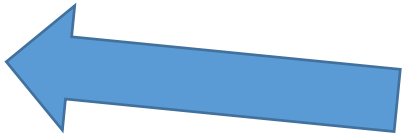

Azure DevOps
oskardudycz / AzurePipelinesSamples / Pipelines / oskardudycz.AzurePipelines... / 20200405.1
Search
AzurePipelinesSamples
Users may experience build delays due to capacity constraints due to increased demand from the global pandemic. See this link for more detail: link
Overview
Pipelines
Pipelines
Environments
Releases
Library
Task groups
Deployment groups
Project settings
Jobs in run #20200405.1
Publish_Prerelease_Nu... 46s
Wait 3m 0s
Contract_tests client1 32s
Initialize job 4s
Checkout <1s
CmdLine 12s
Install .Net Core 6s
CmdLine 4s
Restore Packages 2s
Build 2s
Post-job: Checkout <1s
Finalize Job <1s
Contract_tests client2 26s
Publish_NuGet
Build
View raw log
un a custom dotnet command
ld/dotnet-core-cli
=====
nts/Client1/Client1.sln -dl:CentralLogger,"/home/vsts/work/_tasks/DotNetCoreCLI_5541a522-603c-47ad-91fc-a4b1d163081b/2.166.2/dotne
r CS7036: There is no argument given that corresponds to the required formal parameter 'idColumnName' of 'SchemaUpdater.CreateTab
corresponds to the required formal parameter 'idColumnName' of 'SchemaUpdater.CreateTable(string, string)' [/home/vsts/work/1/s/
corresponds to the required formal parameter 'idColumnName' of 'SchemaUpdater.CreateTable(string, string)' [/home/vsts/work/1/s/
ith exit code 1
3.x SDK/Runtime along with 2.2 & 2.1. Unless you have locked down a SDK version for your project(s), 3.x SDK might be picked up
see that the output folder is now being created at root directory rather than Project File's directory. To learn about more such c
ojects : /home/vsts/work/1/s/Library/Clients/Client1/Client1.sln

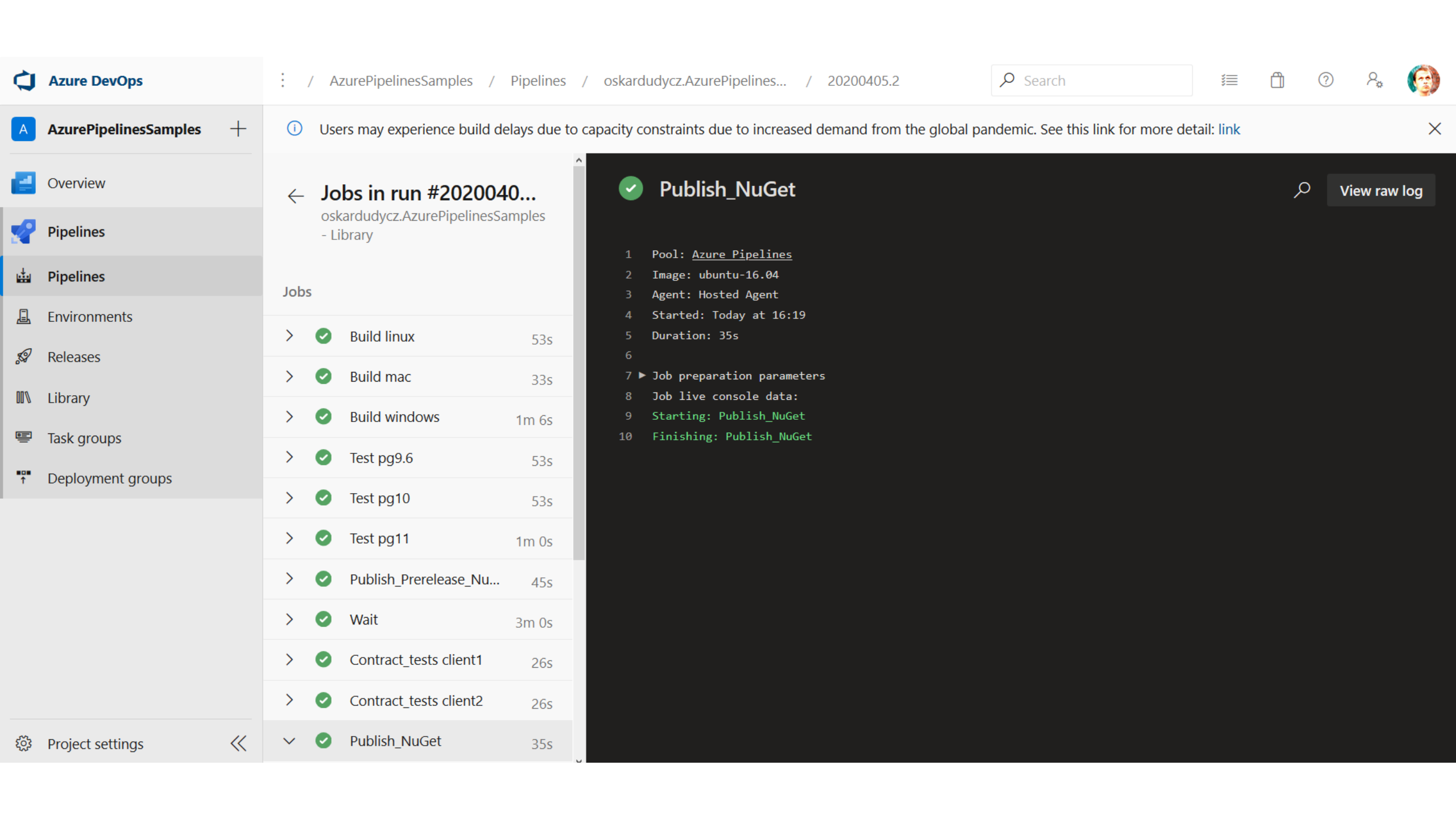
RELEASE



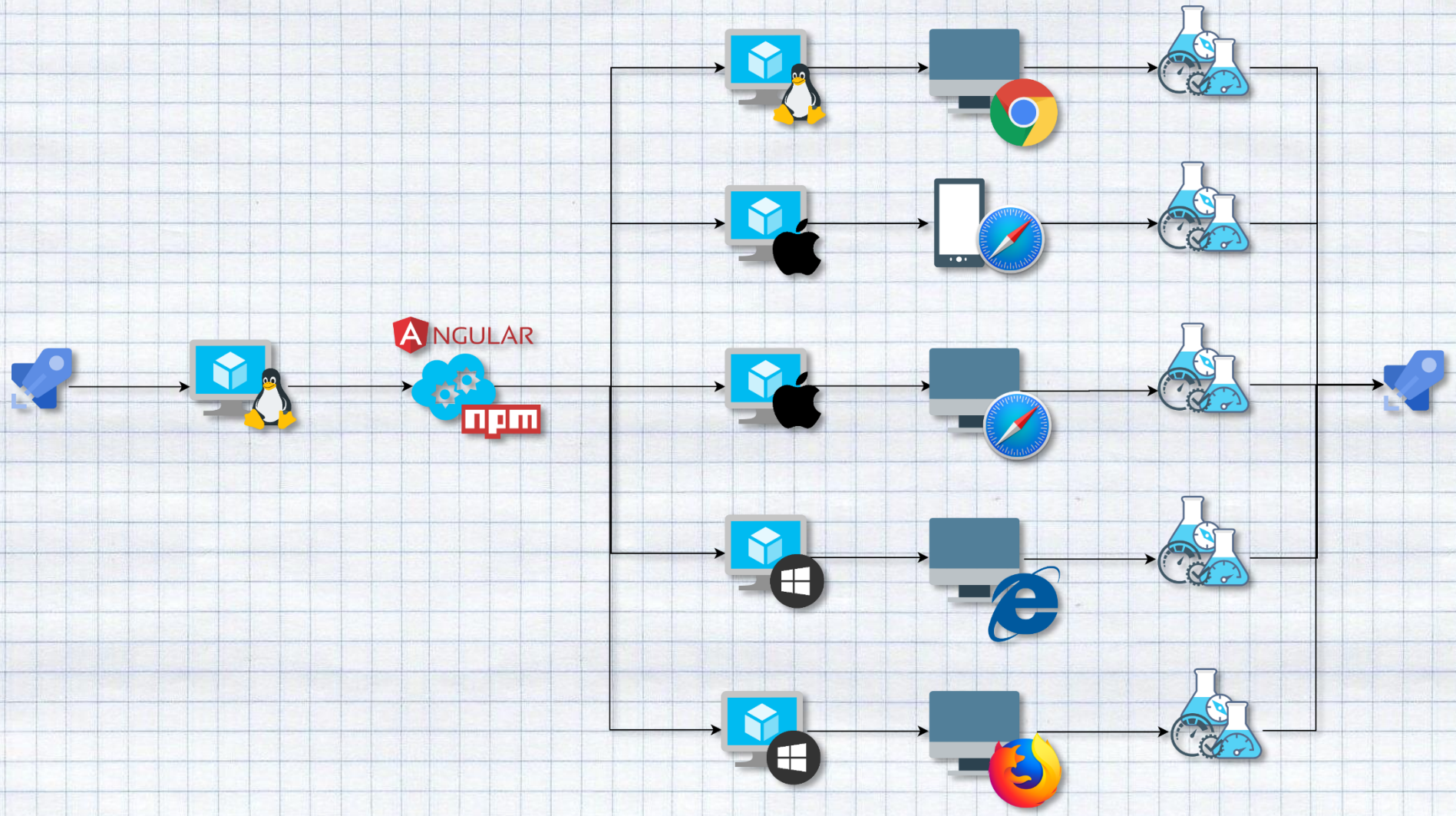
AZURE-PIPELINES.YAML

```yaml
######################################################
#   JOB 4: Wait for NuGet to be published
######################################################
- job: Wait
  dependsOn: Publish_Prerelease_NuGet
  pool: server
  steps:
    - task: Delay@1
      inputs:
        delayForMinutes: '3'

######################################################
#   JOB 5: Perform contract tests to verify
#          if there are no breaking change
######################################################
- template: jobs/job_contract_tests_with_client_apps.yml

######################################################
#   JOB 6: Publish NuGet
######################################################
- template: jobs/job_publish_nuget.yml
  parameters:
    name: Publish_NuGet
    dependsOn: Contract_tests
    nugetVersionEnv: 'NugetVersion'
    condition: and(succeeded(), eq(variables['Build.SourceBranch'], 'refs/heads/master'))
```

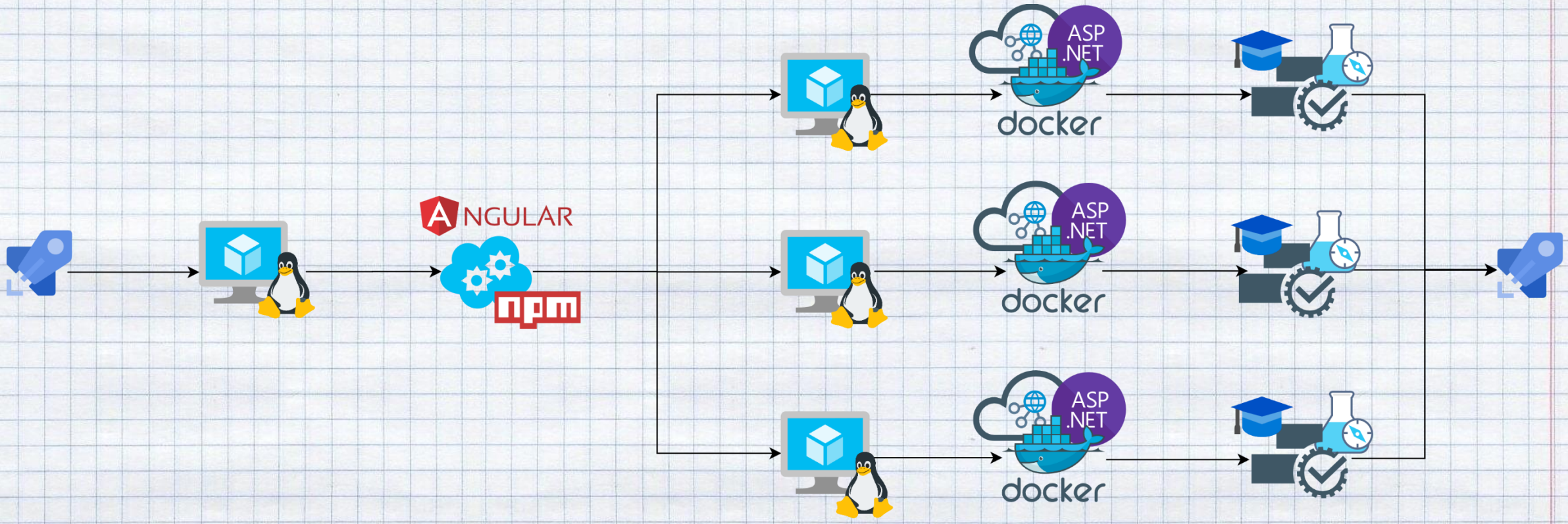



Azure DevOps
AzurePipelinesSamples / Pipelines / oskardudycz.AzurePipelines... / 20200405.2
Search
AzurePipelinesSamples
Users may experience build delays due to capacity constraints due to increased demand from the global pandemic. See this link for more detail: link
Overview
Pipelines
Pipelines
Environments
Releases
Library
Task groups
Deployment groups
Project settings
Jobs in run #2020040...
oskardudycz.AzurePipelinesSamples - Library
Jobs
Build linux 53s
Build mac 33s
Build windows 1m 6s
Test pg9.6 53s
Test pg10 53s
Test pg11 1m 0s
Publish_Prerelease_Nu... 45s
Wait 3m 0s
Contract_tests client1 26s
Contract_tests client2 26s
Publish_NuGet 35s
Publish_NuGet
View raw log
1 Pool: Azure Pipelines
2 Image: ubuntu-16.04
3 Agent: Hosted Agent
4 Started: Today at 16:19
5 Duration: 35s
6
7 ▶ Job preparation parameters
8 Job live console data:
9 Starting: Publish_NuGet
10 Finishing: Publish_NuGet

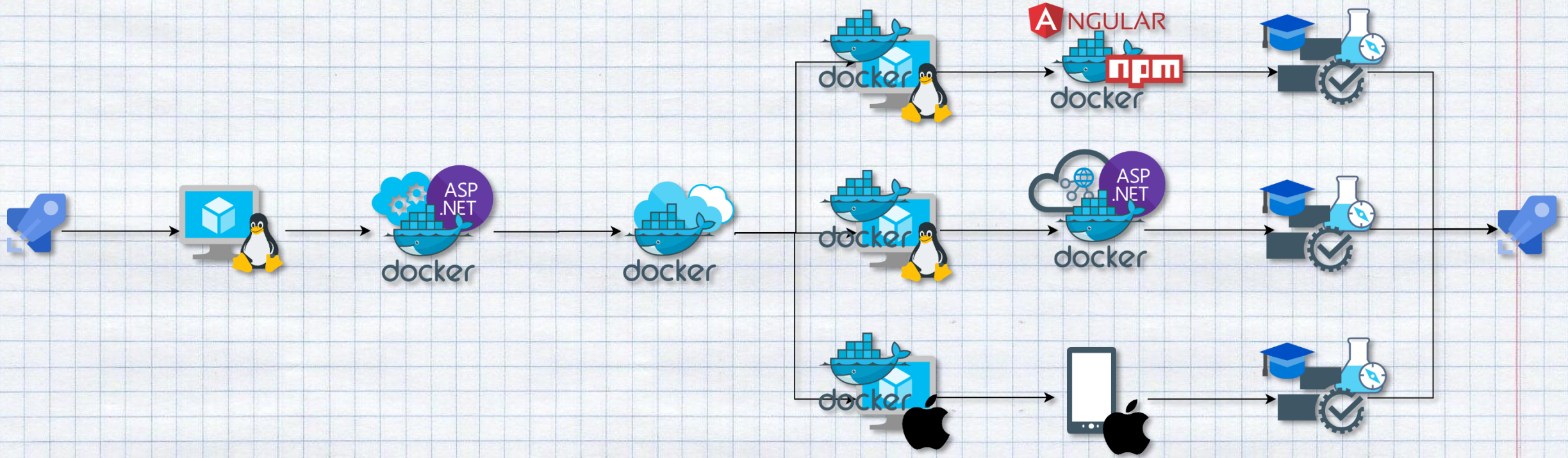
TESTY SELENIUM SPA



TESTY END-TO-END SPA



TESTY END-TO-END API



PODSUMOWANIE

PODSUMOWANIE

TESTY MACIERZOWE UŁATWIAJA URUCHAMIANIE TEGO SAMEGO ZESTAWU TESTÓW DLA ROŻNYCH ŚRODOWISK I KONFIGURACJI

PODSUMOWANIE

PROCESOWANIE MACIERZOWE POZWALA DZIĘKI ZRÓWNOLEGLENIU PRZYŚPIESZYĆ PIPELINE

PODSUMOWANIE

PROCESOWANIE MACIERZOWE UPRASZCZA, SKRACA I UŁATWIA ZARZĄDZANIE PIPELINE

PODSUMOWANIE

PROCESOWANIE MACIERZOWE ULATWIA ZARZĄDZANIE PROCESEM (TECHNICZNYM I BIZNESOWYM), JEGO STABILNOŚCIĄ I PRZEWIDYWALNOŚCIĄ

CO DALEJ?

PRZYKŁADY I MATERIAŁY NA:
HTTPS://GITHUB.COM/OSKARDUDYCZ/AZUREPIPELINESSAMPLES

EMAIL Z PYTANIAMI, UWAGAMI, KRYTYKA:
CONTACT@OSKAR.DUDYCZ.PL

SKLEJENIE ŻÓŁWIKA PO PREZENTACJI! :)

NAPISZ FEEDBACK NA:
CONTACT@OSKAR-DUDYCZ.PL

DZIĘKUJĘ WSZYSTKIM ZA UWAGĘ! :)