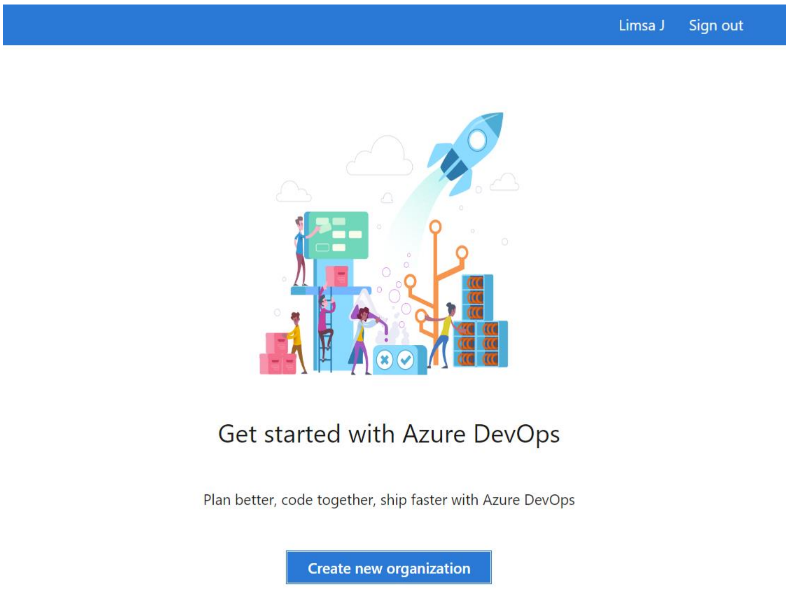
Ex No. 6 Create a Maven build pipeline in Azure

1. **Sign in to Azure:**

   Go to Azure DevOps and create an organization.

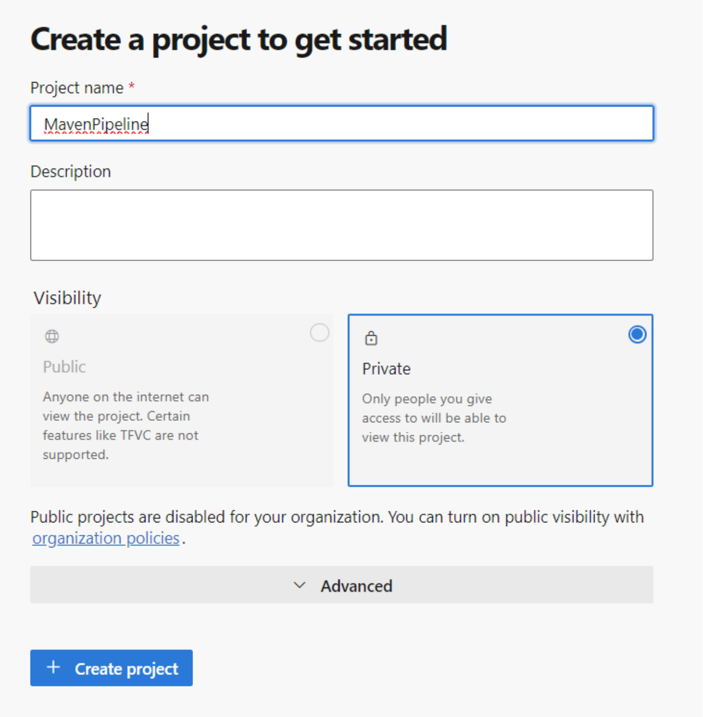


Get started with Azure DevOps

Plan better, code together, ship faster with Azure DevOps

2. **Create a New Project:**

   Create a new one by clicking on "New project" on the Azure DevOps dashboard.



Create a project to get started

Project name *: MavenPipeline

Description:

Visibility

○ Public
Anyone on the internet can view the project. Certain features like TFVC are not supported.

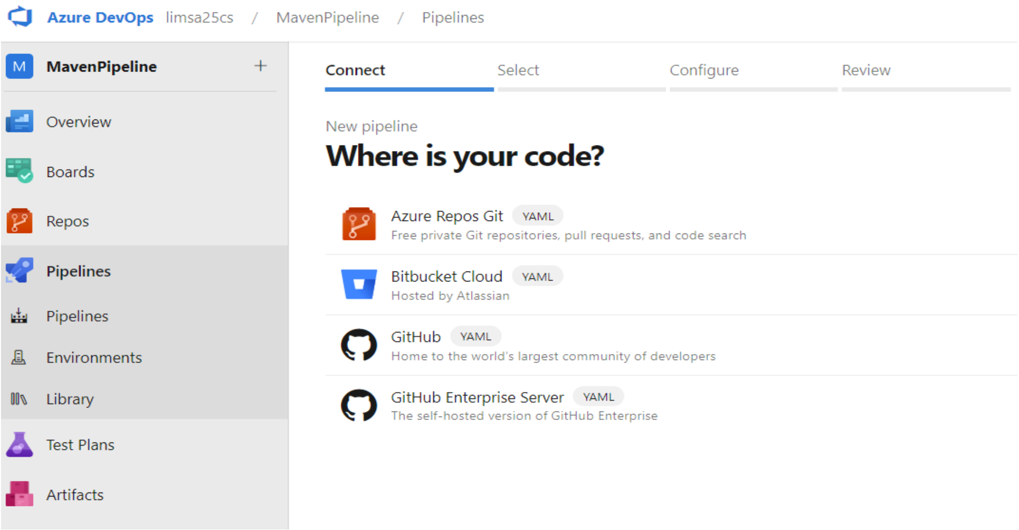
● Private
Only people you give access to will be able to view this project.

Public projects are disabled for your organization. You can turn on public visibility with organization policies.

3. **Create Azure Pipeline:**

   Navigate to Pipelines > Pipelines in the Azure DevOps dashboard.

   Click on "New pipeline" to create a new pipeline.

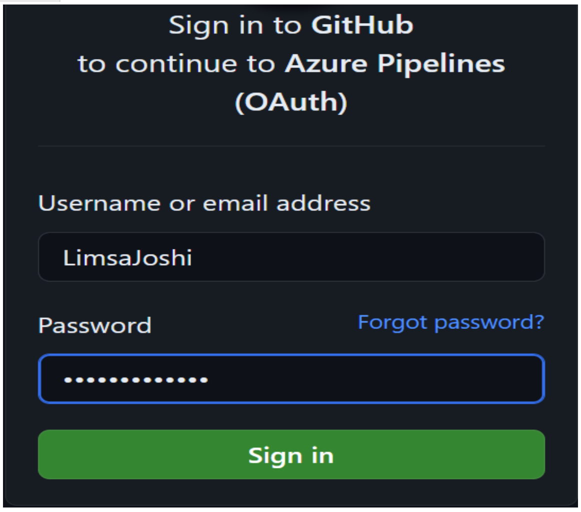


Azure DevOps limsa25cs / MavenPipeline / Pipelines

New pipeline

Where is your code?

- Azure Repos Git YAML — Free private Git repositories, pull requests, and code search
- Bitbucket Cloud YAML — Hosted by Atlassian
- GitHub YAML — Home to the world's largest community of developers
- GitHub Enterprise Server YAML — The self-hosted version of GitHub Enterprise



Sign in to GitHub
to continue to Azure Pipelines
(OAuth)

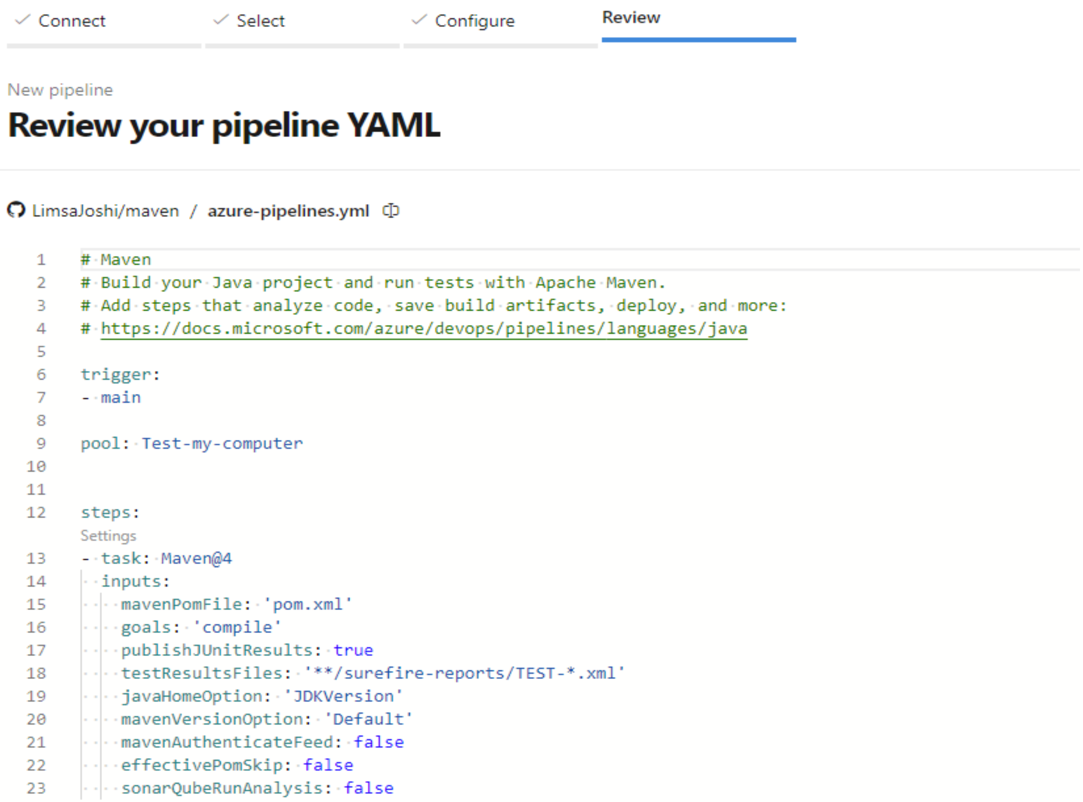
Username or email address: LimsaJoshi

Password: ••••••••••••••

Configure Azure DevOps CI pipeline:

- Use “New pipeleine” in “Pipilenes -> Builds” in Azure DevOps project.
- Connect to GitHub and select your repo. You may want to install Azure DevOps GitHub app and grant access to your GitHub repo.
- Select “Maven” in “Configure your pipeline” step.
- Create new pipeline and push `azure-pipelines.yml` to your repo to keep it alogn with the source code and run build.
- As the result you will get:
  1. CI pipeline configued in Azure DevOps
  2. `azure-pipelines.yml` config in your repo

Define the Maven goals and options, such as clean, compile, package, test, etc., based on your project's needs.
Ensure you have the necessary plugins and dependencies configured in your pom.xml file.



New pipeline

Review your pipeline YAML

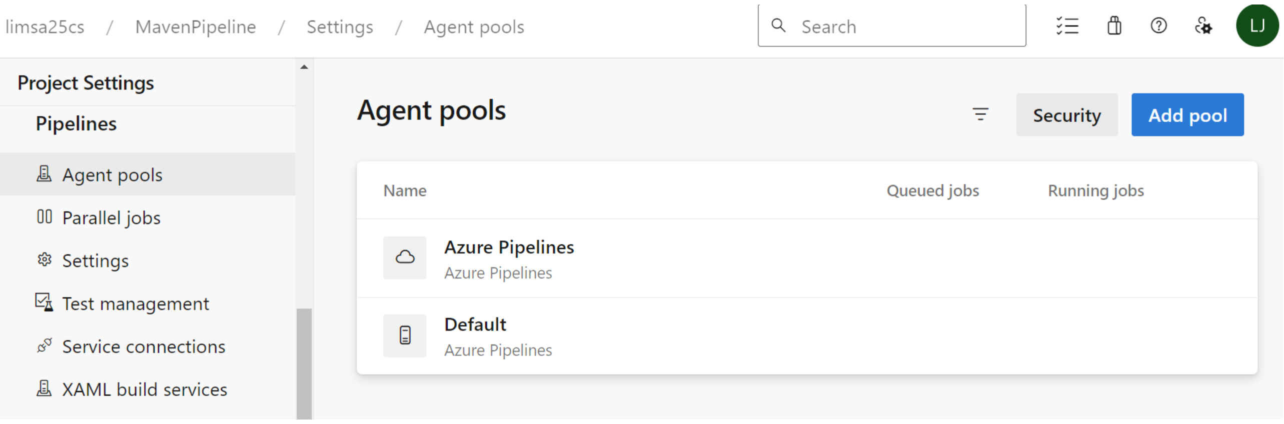
LimsaJoshi/maven / azure-pipelines.yml

```yaml
# Maven
# Build your Java project and run tests with Apache Maven.
# Add steps that analyze code, save build artifacts, deploy, and more:
# https://docs.microsoft.com/azure/devops/pipelines/languages/java

trigger:
- main

pool: Test-my-computer


steps:
- task: Maven@4
  inputs:
    mavenPomFile: 'pom.xml'
    goals: 'compile'
    publishJUnitResults: true
    testResultsFiles: '**/surefire-reports/TEST-*.xml'
    javaHomeOption: 'JDKVersion'
    mavenVersionOption: 'Default'
    mavenAuthenticateFeed: false
    effectivePomSkip: false
    sonarQubeRunAnalysis: false
```

4. **Save and Run Pipeline:**
   Once configured the pipeline, save the changes to the YAML file.
   Optionally, you can run a manual build to test the pipeline's functionality and verify that the Maven build process executes successfully.
5. **Review Build Results:**
   After the pipeline runs, review the build results, logs, and any errors or warnings.
   Azure DevOps provides detailed reports and insights into the build process, including test results if you've configured testing tasks in your pipeline.
6. **Add Additional Tasks (Optional):**
   Depending on your project's requirements, add additional tasks to the pipeline, such as deploying artifacts to Azure services, running integration tests, or generating build artifacts.
7. **Configure Deployment (Optional):**
   If the pipeline includes deployment tasks, configure the deployment targets and settings accordingly. Deploy Maven artifacts to Azure App Service, Azure Kubernetes Service (AKS), Azure Functions, or any other Azure resources.

Steps to create a Self-Hosted Agent

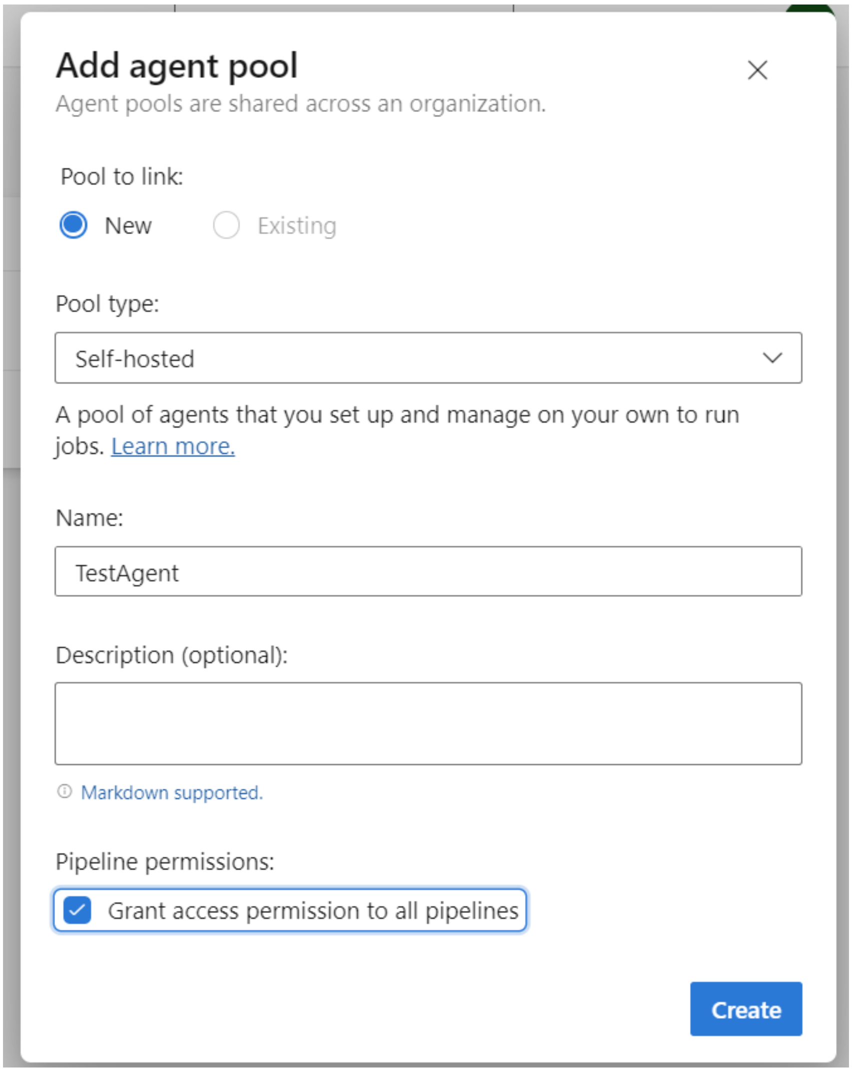
1. Open your web browser and log in to your Azure DevOps account.
2. Navigate to your Azure DevOps project and click on **Project settings** in the left side of the page.
3. Click on **Agent Pools** under Pipelines and click on **Add pool**.



Agent pools

Name	Queued jobs	Running jobs
Azure Pipelines Azure Pipelines		
Default Azure Pipelines		

1. Select **Pool-type** as **Self-hosted**, give it a name, grant access to all pipellines and click on **Create.**



Add agent pool

Agent pools are shared across an organization.

Pool to link:

● New ○ Existing

Pool type:

Self-hosted

A pool of agents that you set up and manage on your own to run jobs. Learn more.

Name:

TestAgent

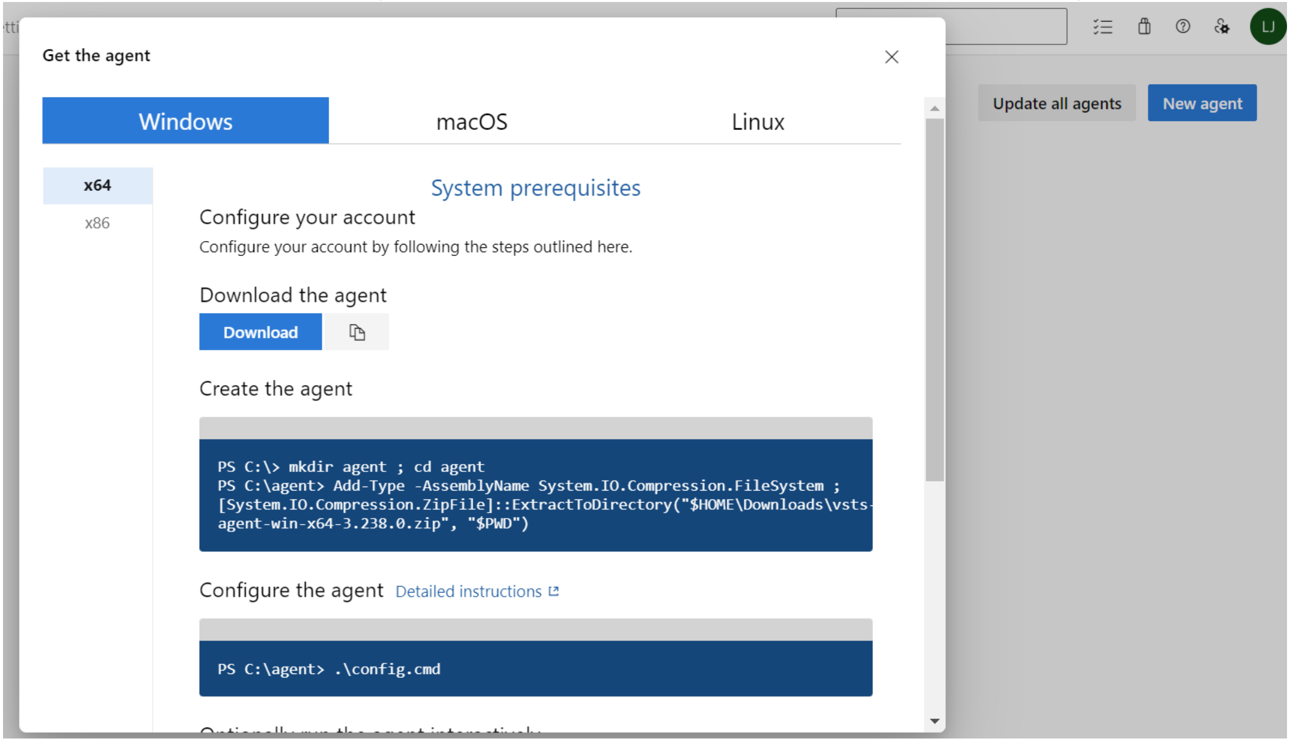
Description (optional):

Markdown supported.

Pipeline permissions:

☑ Grant access permission to all pipelines

2. Click on the created agent → Click on **New agent**. It will show the below screen. Just follow the steps to create the agent. Click on the **Download** button to download the agent.



System prerequisites

Configure your account

Configure your account by following the steps outlined here.

Download the agent

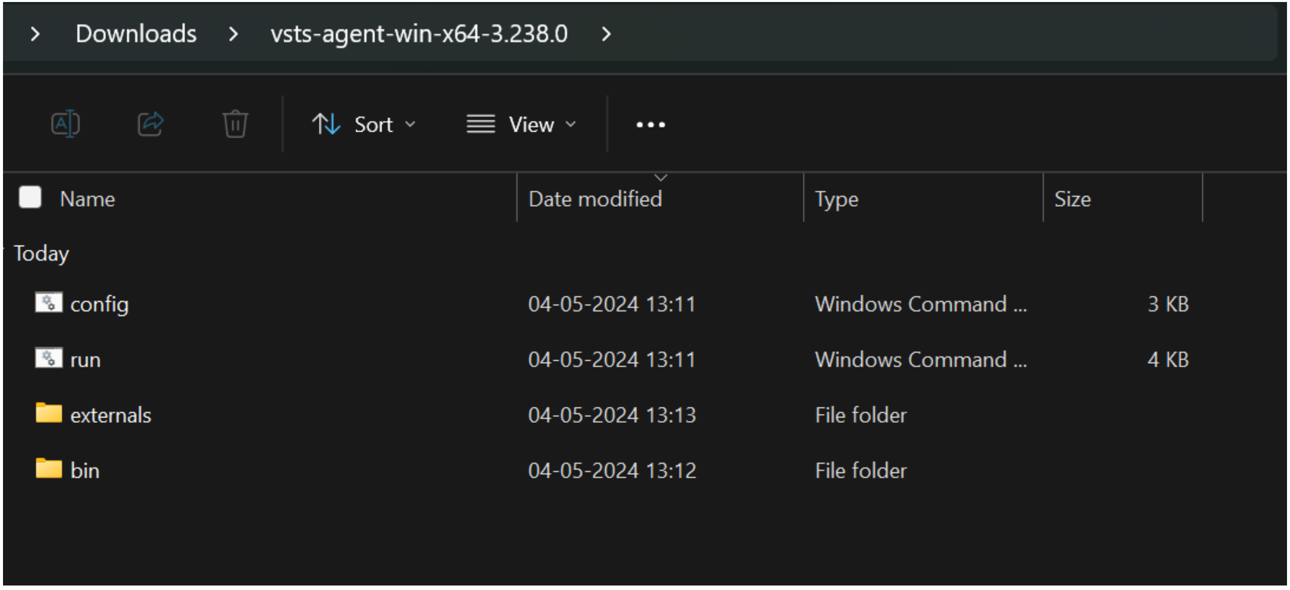
Create the agent

```
PS C:\> mkdir agent ; cd agent
PS C:\agent> Add-Type -AssemblyName System.IO.Compression.FileSystem ;
[System.IO.Compression.ZipFile]::ExtractToDirectory("$HOME\Downloads\vsts-
agent-win-x64-3.238.0.zip", "$PWD")
```

Configure the agent

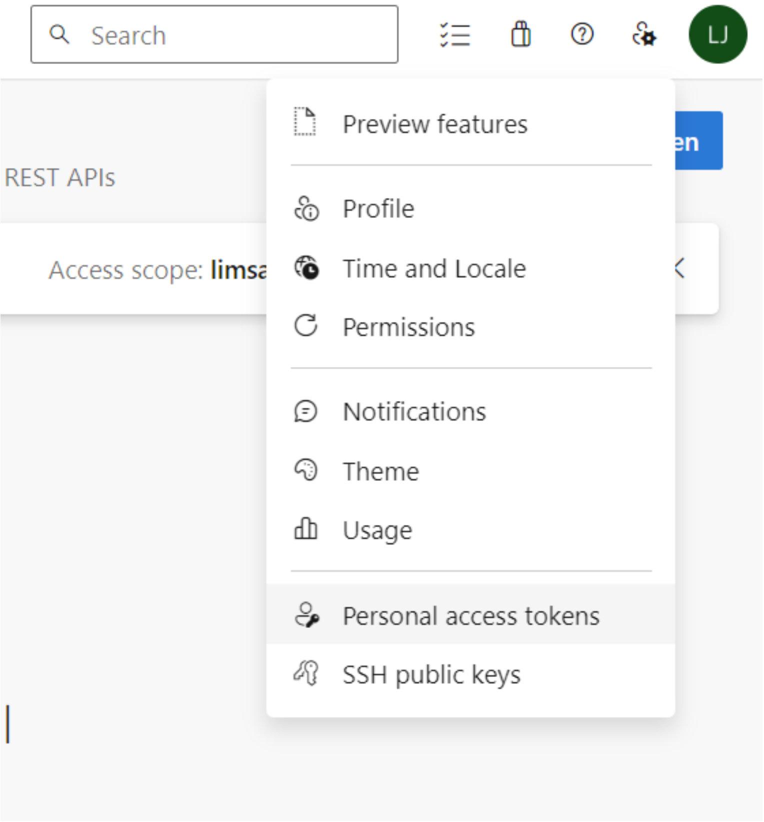
```
PS C:\agent> .\config.cmd
```

3. Extract the downloaded agent package to a directory on your system.

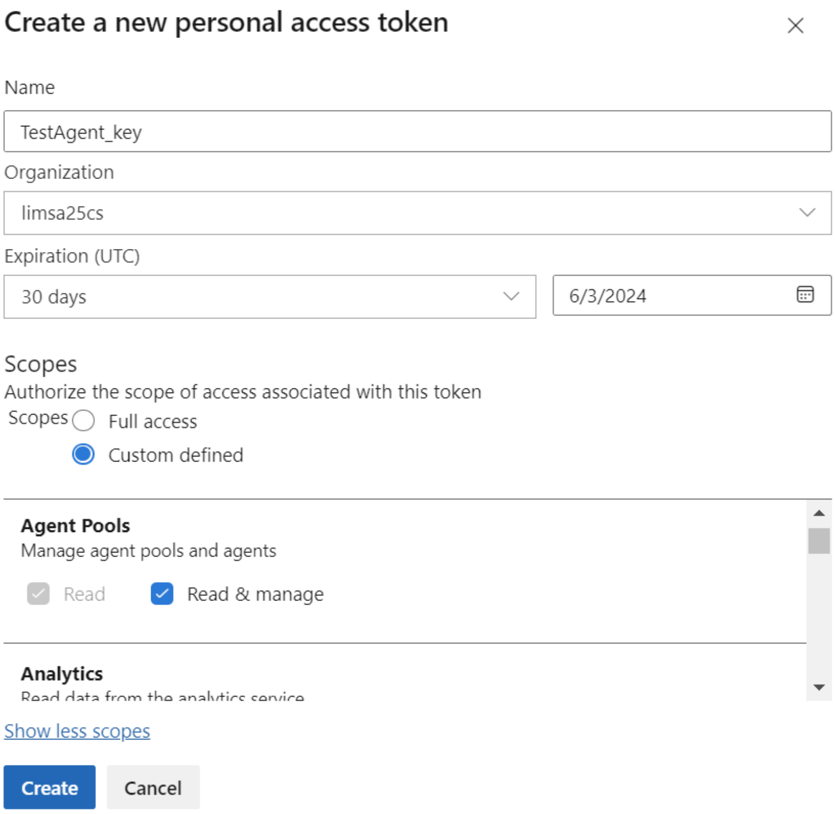


Name	Date modified	Type	Size
config	04-05-2024 13:11	Windows Command ...	3 KB
run	04-05-2024 13:11	Windows Command ...	4 KB
externals	04-05-2024 13:13	File folder	
bin	04-05-2024 13:12	File folder	

4. Run the configuration script. The configuration script will prompt for your Azure Organization account URL and a personal access token (PAT).
5. To generate a PAT, go to Azure DevOps account, click on the small icon on the left side of the profile picture in the top-right corner, and select **Personal access tokens** from the dropdown menu.



6. Generate a new token with appropriate permissions for the agent.



Create a new personal access token

Name

TestAgent_key

Organization

limsa25cs

Expiration (UTC)

30 days 6/3/2024

Scopes

Authorize the scope of access associated with this token

Scopes ○ Full access

● Custom defined

Agent Pools

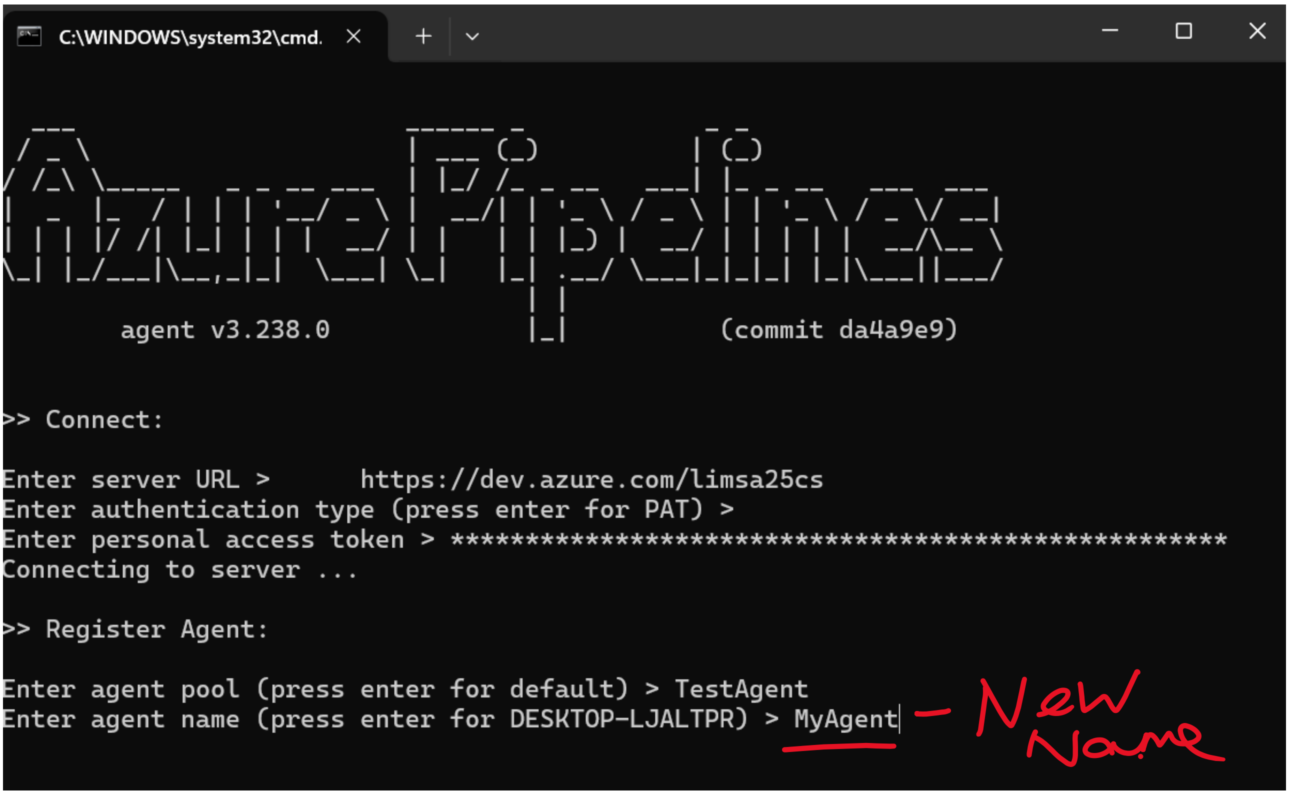
Manage agent pools and agents

☑ Read ☑ Read & manage

Analytics

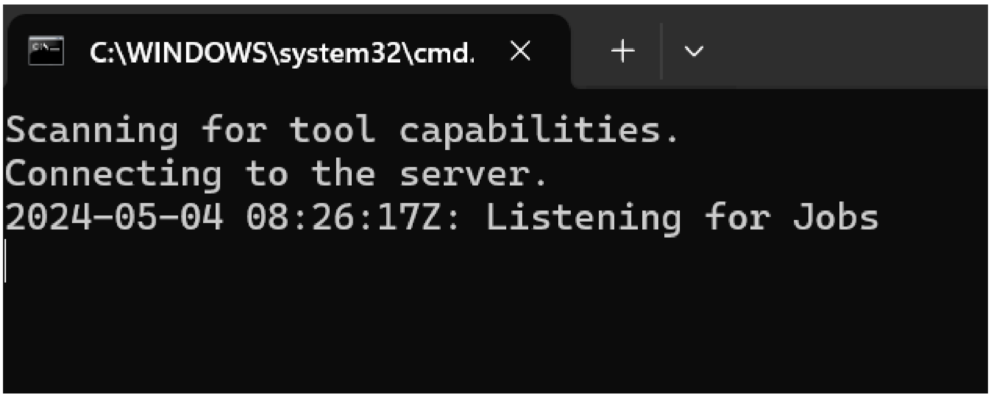
Read data from the analytics service

7. Run the configuration file “config.cmd”. Enter the Organization URL, PAT created in the previous step, **Agent pool name created in step 4** and **enter a new name for the agent or press enter for taking the system name as default.**



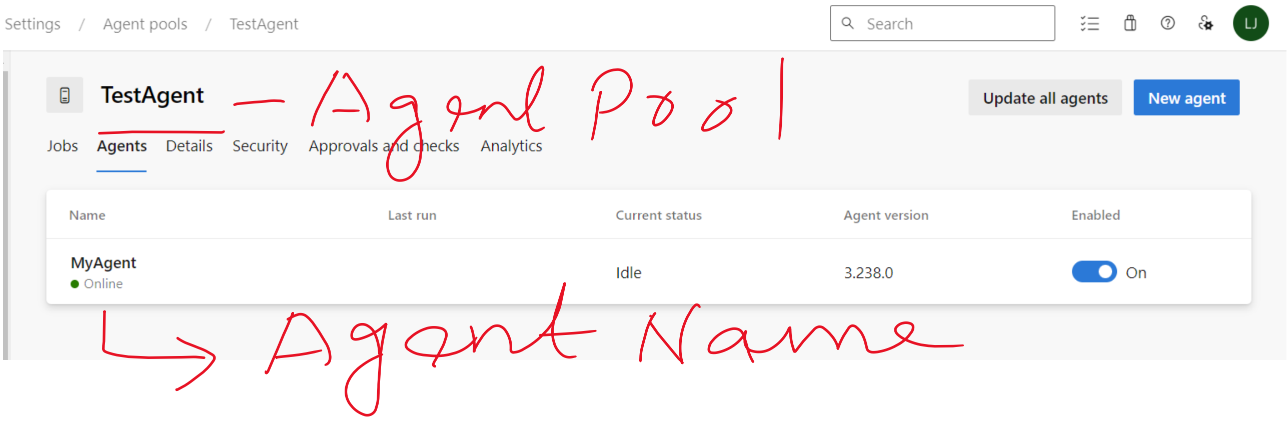
```
  ___                      ______ _            _ _
 / _ \                     | ___ (_)          | (_)
/ /_\ \_____   _ _ __ ___  | |_/ /_ _ __   ___| |_ _ __   ___  ___
|  _  |_  / | | | '__/ _ \ |  __/| | '_ \ / _ \ | | '_ \ / _ \/ __|
| | | |/ /| |_| | | |  __/ | |   | | |_) |  __/ | | | | |  __/\__ \
\_| |_/___|\__,_|_|  \___| \_|   |_| .__/ \___|_|_|_| |_|\___||___/
                                   | |
       agent v3.238.0              |_|          (commit da4a9e9)


>> Connect:

Enter server URL >      https://dev.azure.com/limsa25cs
Enter authentication type (press enter for PAT) >
Enter personal access token > ****************************************************
Connecting to server ...

>> Register Agent:

Enter agent pool (press enter for default) > TestAgent
Enter agent name (press enter for DESKTOP-LJALTPR) > MyAgent
```

8. Execute “run.cmd” to run the agent



```
Scanning for tool capabilities.
Connecting to the server.
2024-05-04 08:26:17Z: Listening for Jobs
```

9. After completing the configuration, the agent will register itself with the specified agent pool in Azure DevOps. Now, go back to agent pool → Select the created Agent Pool (TestAgent), you will see the agent (MyAgent) online.

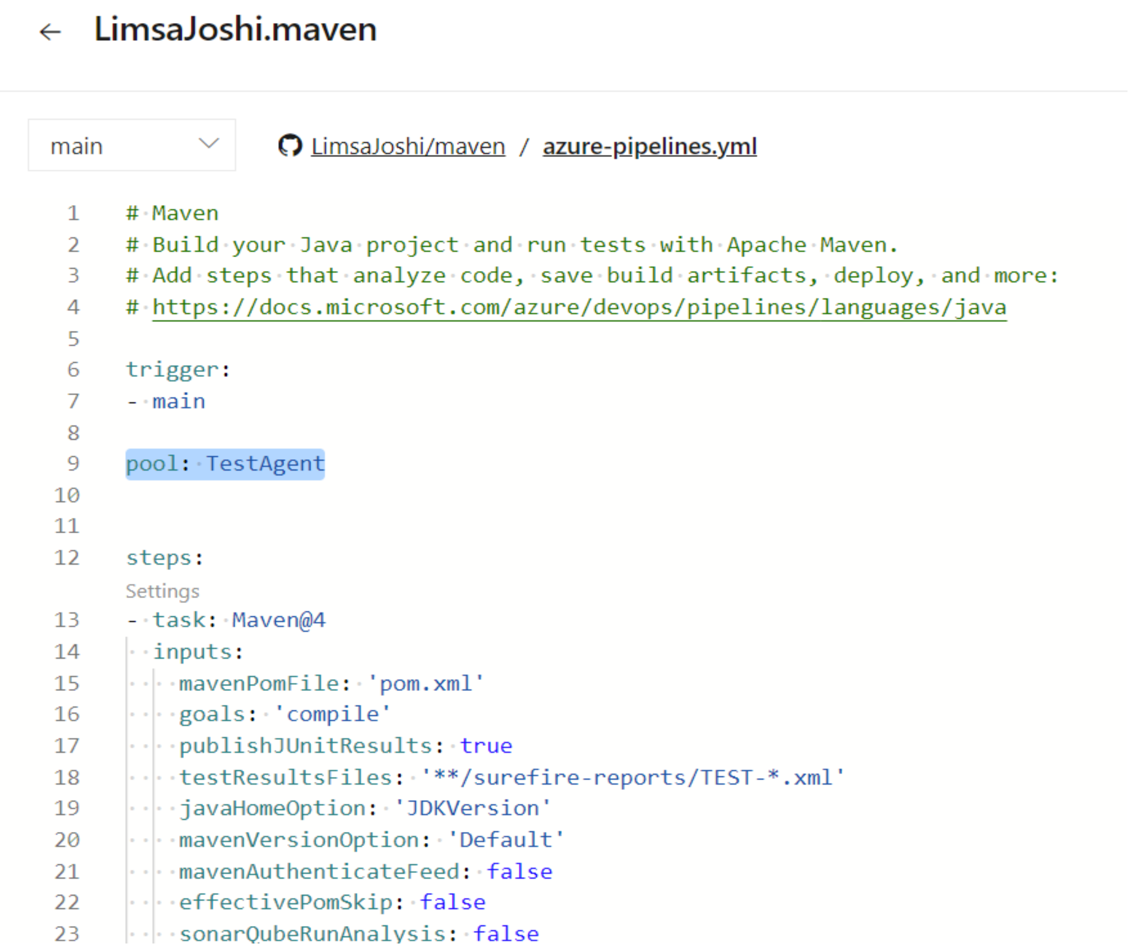


— Agent Pool

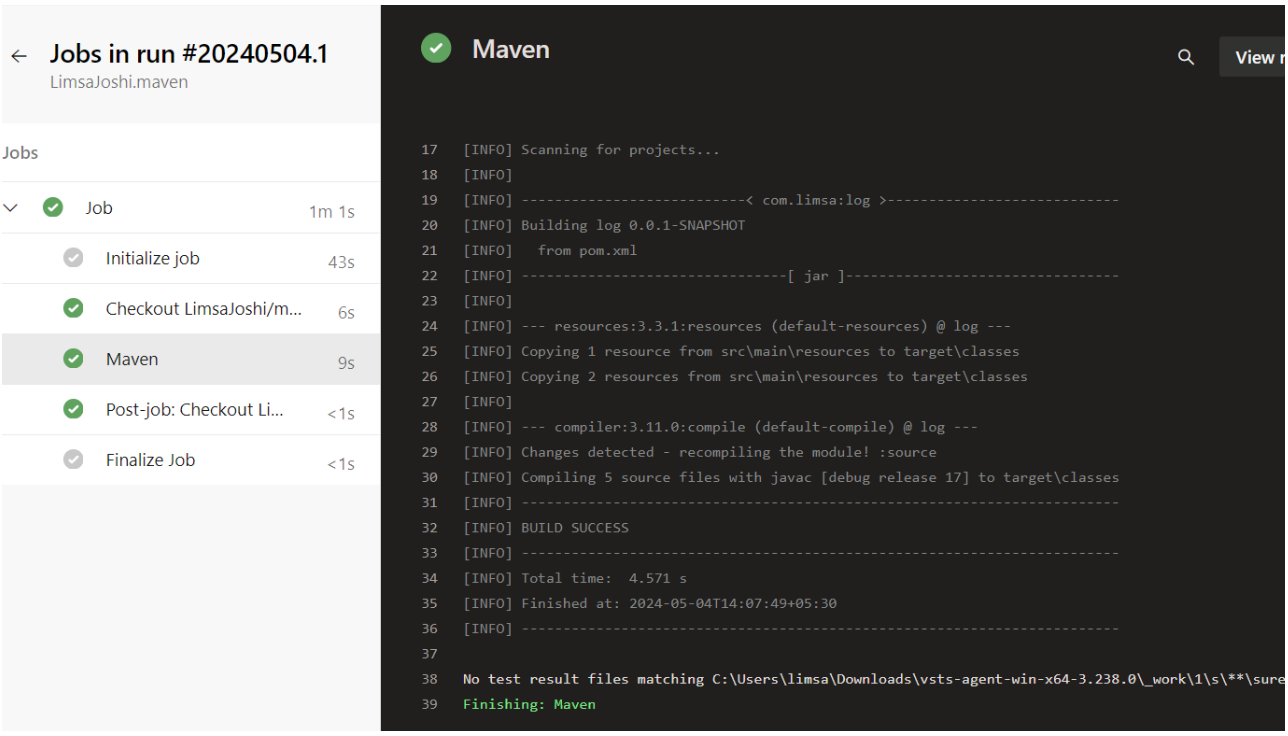
Name	Last run	Current status	Agent version	Enabled
MyAgent ● Online		Idle	3.238.0	On

↳ Agent Name

10. Now use the self-hosted agent in the pipeline



```yaml
# Maven
# Build your Java project and run tests with Apache Maven.
# Add steps that analyze code, save build artifacts, deploy, and more:
# https://docs.microsoft.com/azure/devops/pipelines/languages/java

trigger:
- main

pool: TestAgent


steps:
- task: Maven@4
  inputs:
    mavenPomFile: 'pom.xml'
    goals: 'compile'
    publishJUnitResults: true
    testResultsFiles: '**/surefire-reports/TEST-*.xml'
    javaHomeOption: 'JDKVersion'
    mavenVersionOption: 'Default'
    mavenAuthenticateFeed: false
    effectivePomSkip: false
    sonarQubeRunAnalysis: false
```



Jobs in run #20240504.1

LimsaJoshi.maven

Jobs

Job	1m 1s
Initialize job	43s
Checkout LimsaJoshi/m...	6s
Maven	9s
Post-job: Checkout Li...	<1s
Finalize Job	<1s

Maven

```
[INFO] Scanning for projects...
[INFO]
[INFO] -------------------------< com.limsa:log >--------------------------
[INFO] Building log 0.0.1-SNAPSHOT
[INFO]   from pom.xml
[INFO] --------------------------------[ jar ]---------------------------------
[INFO]
[INFO] --- resources:3.3.1:resources (default-resources) @ log ---
[INFO] Copying 1 resource from src\main\resources to target\classes
[INFO] Copying 2 resources from src\main\resources to target\classes
[INFO]
[INFO] --- compiler:3.11.0:compile (default-compile) @ log ---
[INFO] Changes detected - recompiling the module! :source
[INFO] Compiling 5 source files with javac [debug release 17] to target\classes
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  4.571 s
[INFO] Finished at: 2024-05-04T14:07:49+05:30
[INFO] ------------------------------------------------------------------------

No test result files matching C:\Users\limsa\Downloads\vsts-agent-win-x64-3.238.0\_work\1\s\**\sure
Finishing: Maven
```

Result: