1 Docker-Installation Instructions for PNYX

1.1 Introduction

The following document describes the required steps to install a running Docker environment for PNYX on:

- Windows 11
- Mac
- Linux (Debian / Ubuntu)

The following steps are required in principle to make it work:

1. Clone GIT PNYX-Archive https://github.com/NeaBouli/pnyx/tree/development/Frontend
2. Install Docker Desktop
3. Install .NET 6.0 SDK
4. Register development certs
5. Run PNYX Docker containers

1.2 Windows Installation

The following steps describe the installation of PNYX Docker on Windows

1.2.1 Clone GIT PNYX-Archive

You need to install the GIT Console which can be downloaded here:

https://git-scm.com/download/win

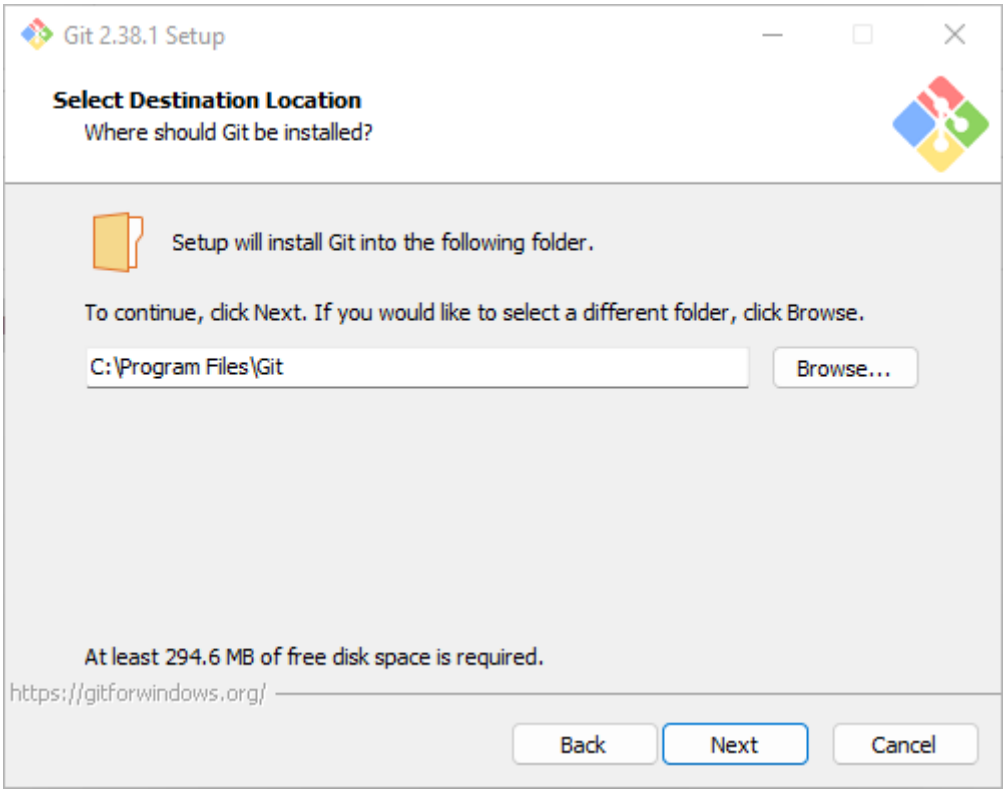
Download the relevant version of the GIT for your system (should be 64-bit in most cases) and install it.

For better editing capabilities you should install Notepad++ before doing the GIT installation:

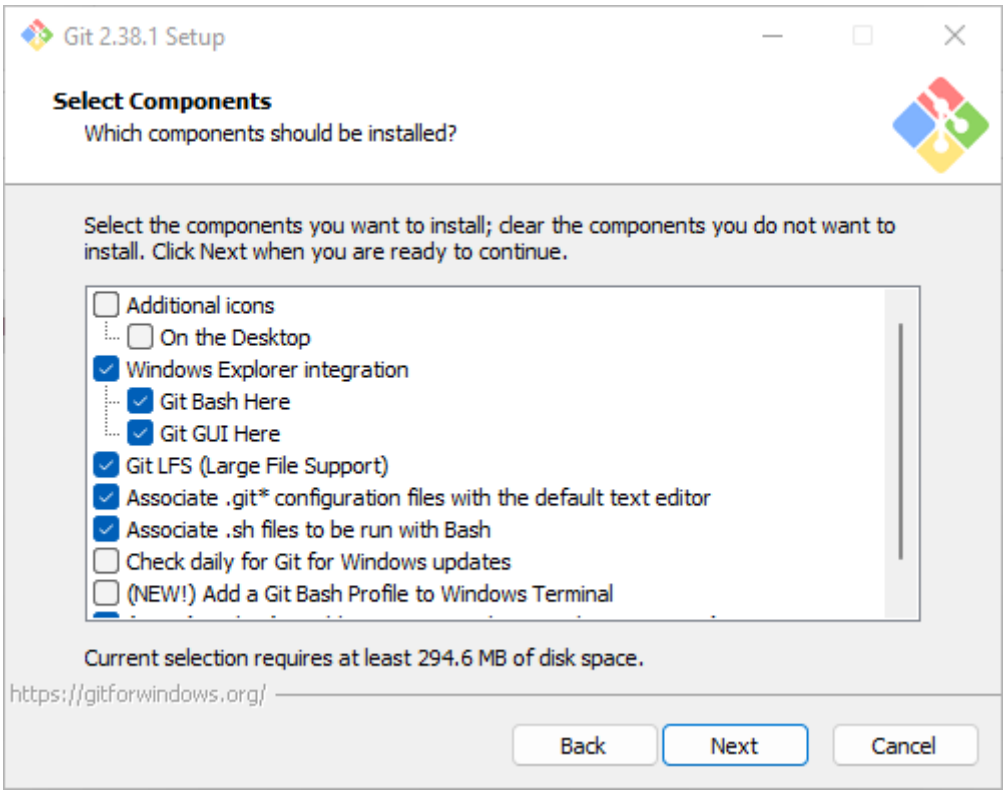
https://notepad-plus-plus.org/downloads/

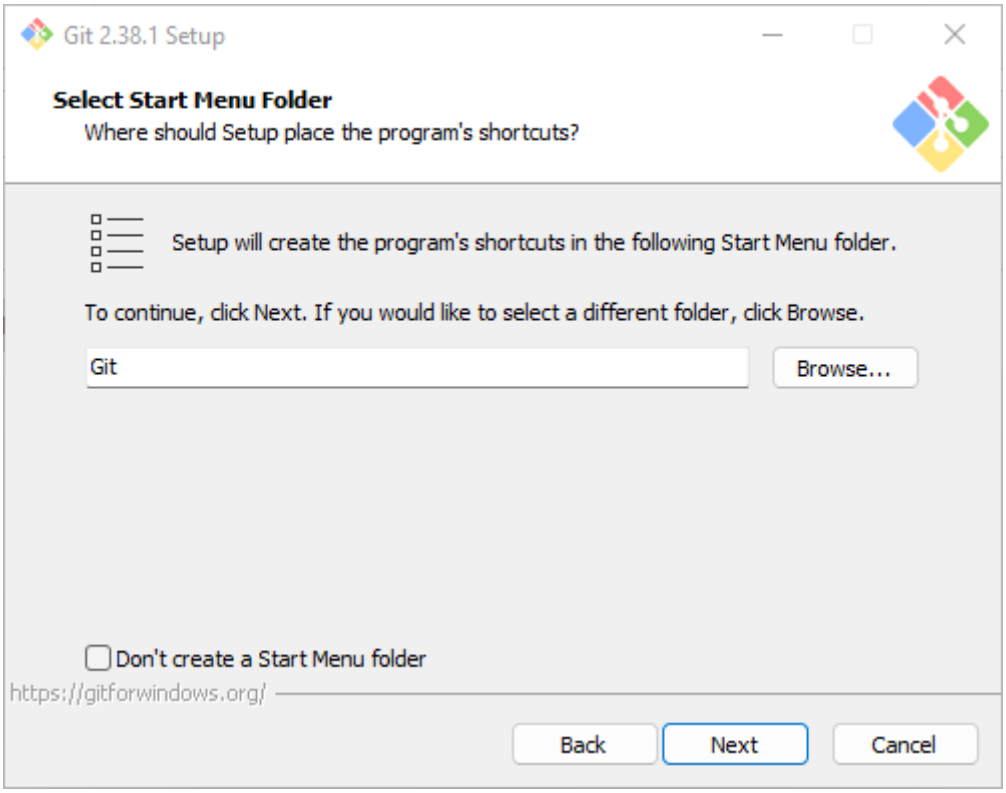
Git installation is done as follows:

Note: You might install WSL and Ubuntu on Windows in order to make GIT work.

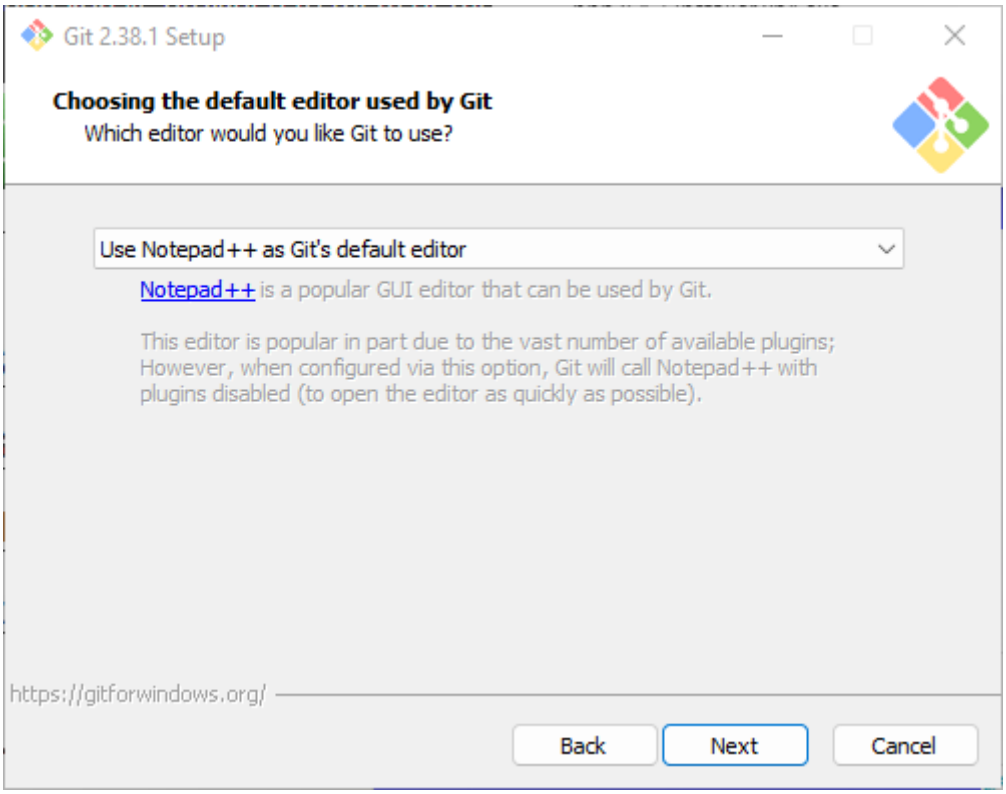


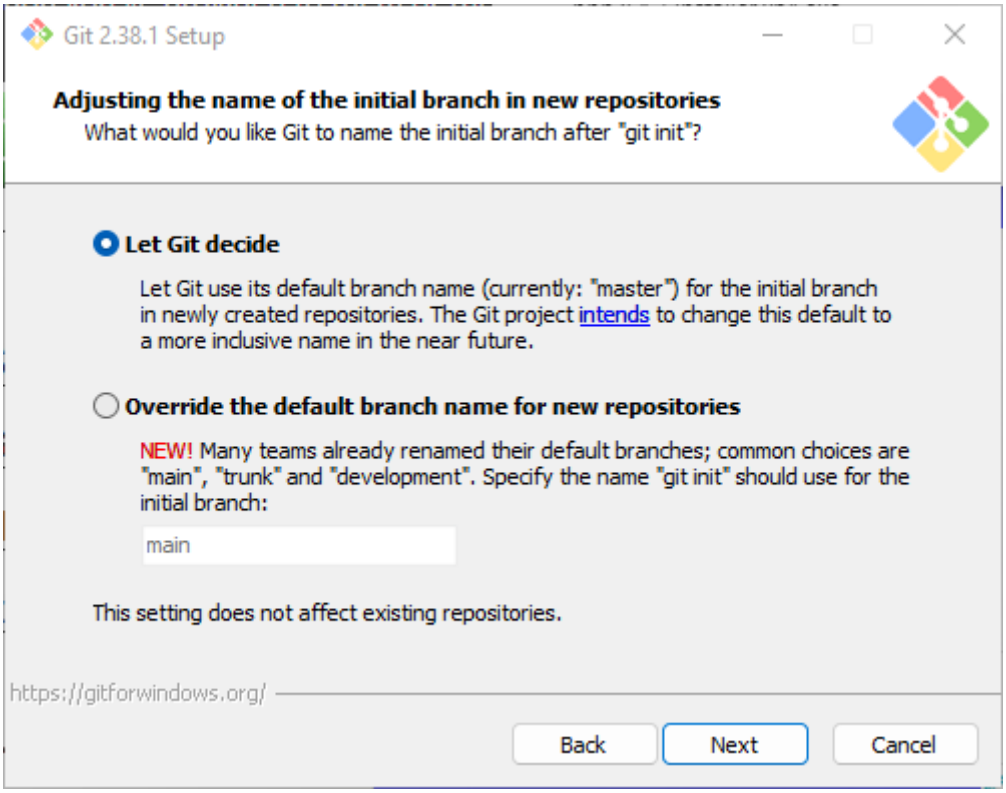
You could leave defaults on the next screen:

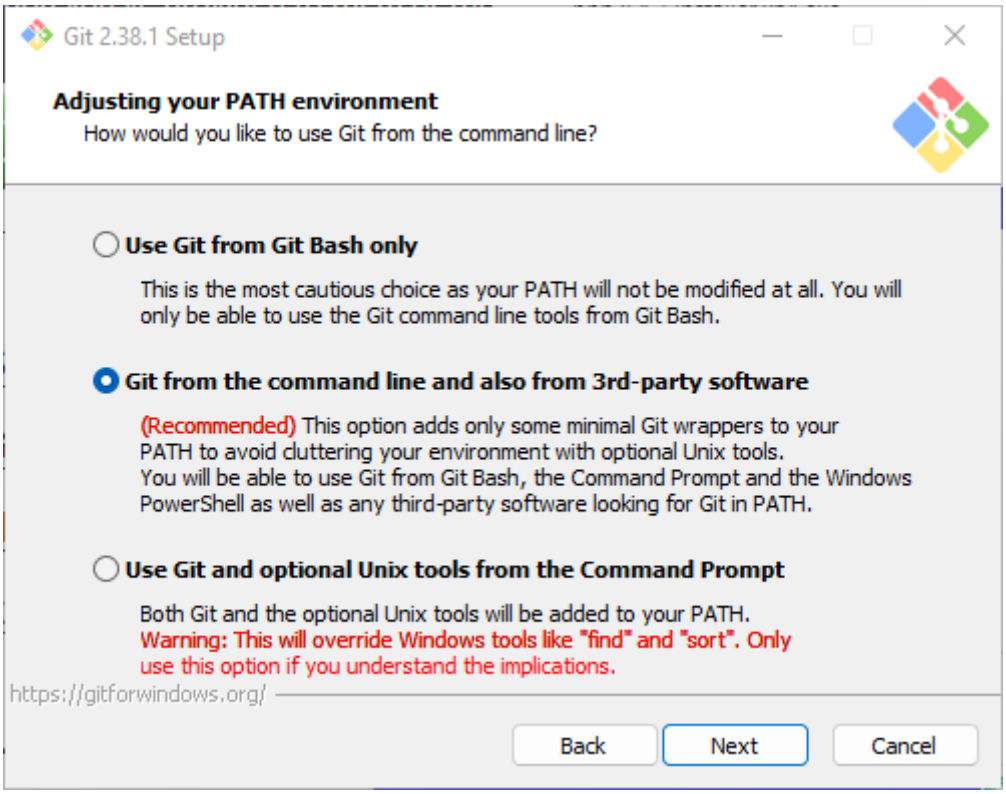


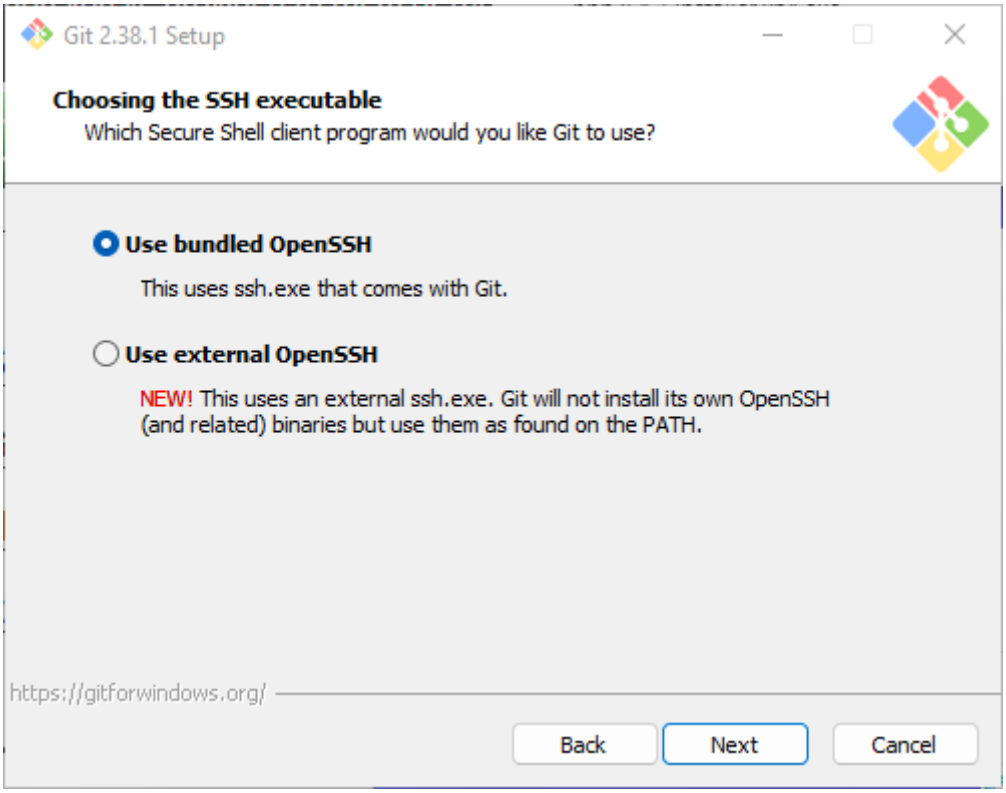


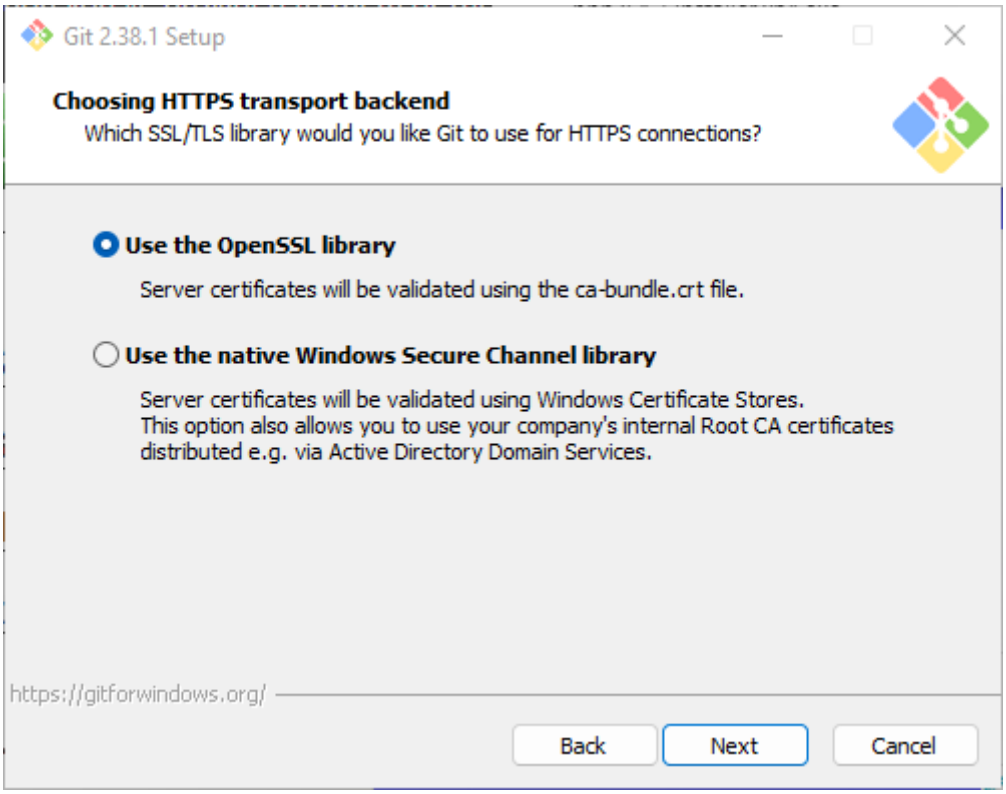
You better should install Notepad++ and use it as GIT's default editor:

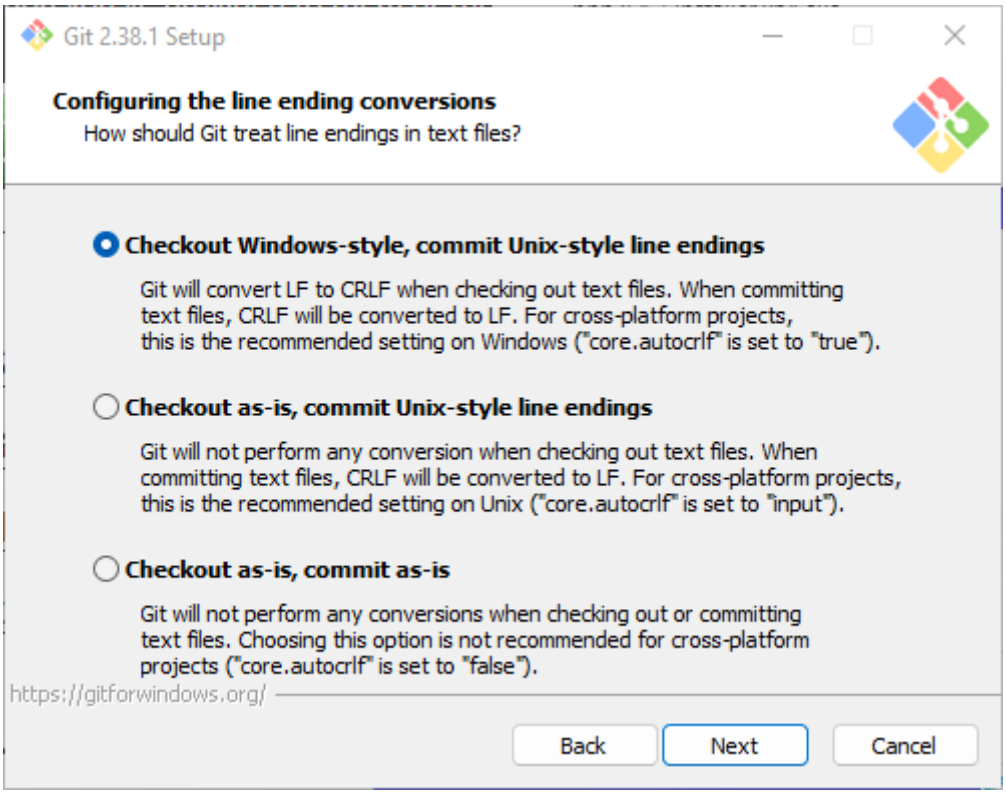


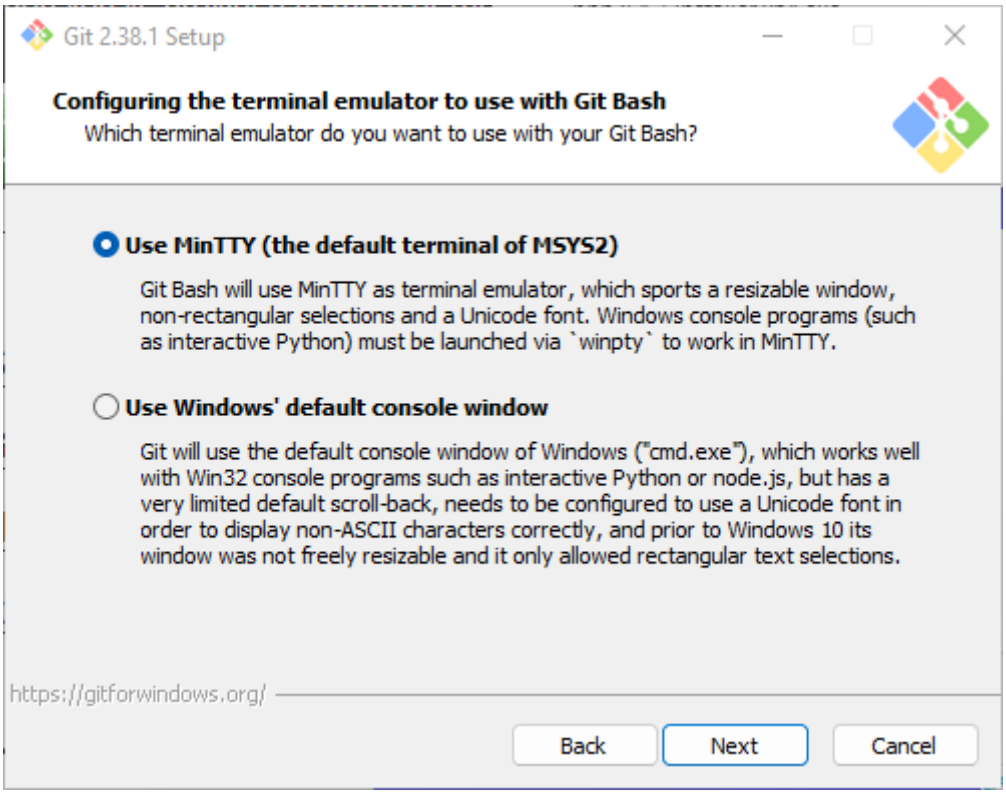


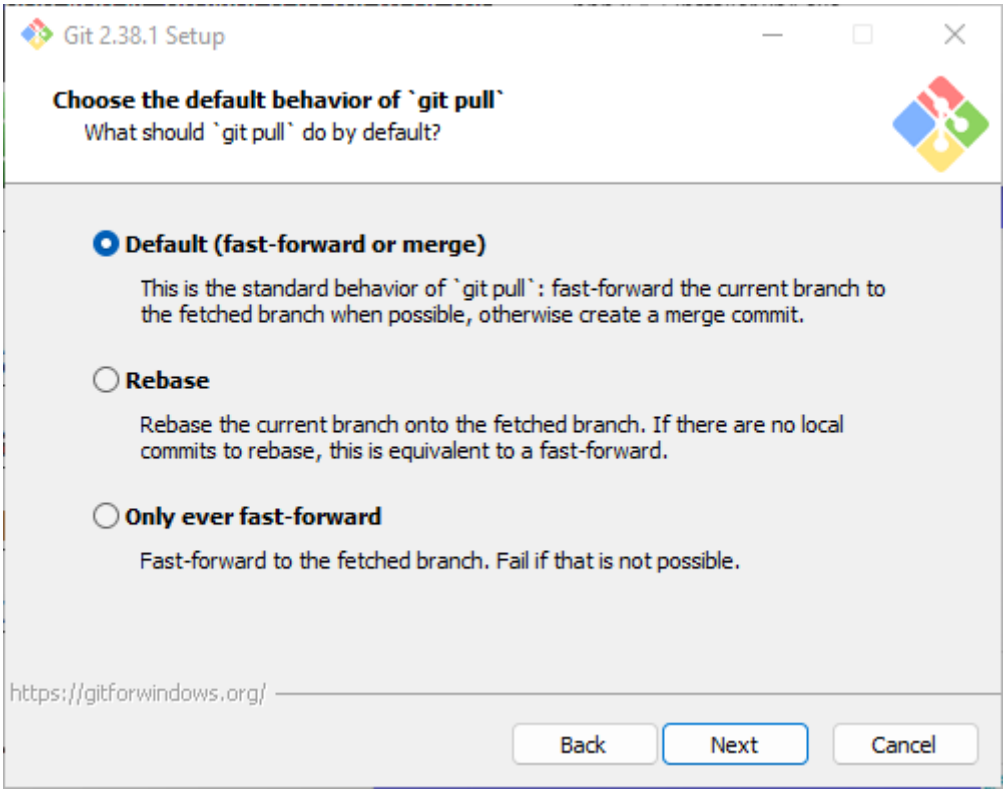

Git 2.38.1 Setup
Adjusting your PATH environment
How would you like to use Git from the command line?
Use Git from Git Bash only
This is the most cautious choice as your PATH will not be modified at all. You will only be able to use the Git command line tools from Git Bash.
Git from the command line and also from 3rd-party software
(Recommended) This option adds only some minimal Git wrappers to your PATH to avoid cluttering your environment with optional Unix tools.
You will be able to use Git from Git Bash, the Command Prompt and the Windows PowerShell as well as any third-party software looking for Git in PATH.
Use Git and optional Unix tools from the Command Prompt
Both Git and the optional Unix tools will be added to your PATH.
Warning: This will override Windows tools like "find" and "sort". Only use this option if you understand the implications.
https://gitforwindows.org/
Back
Next
Cancel

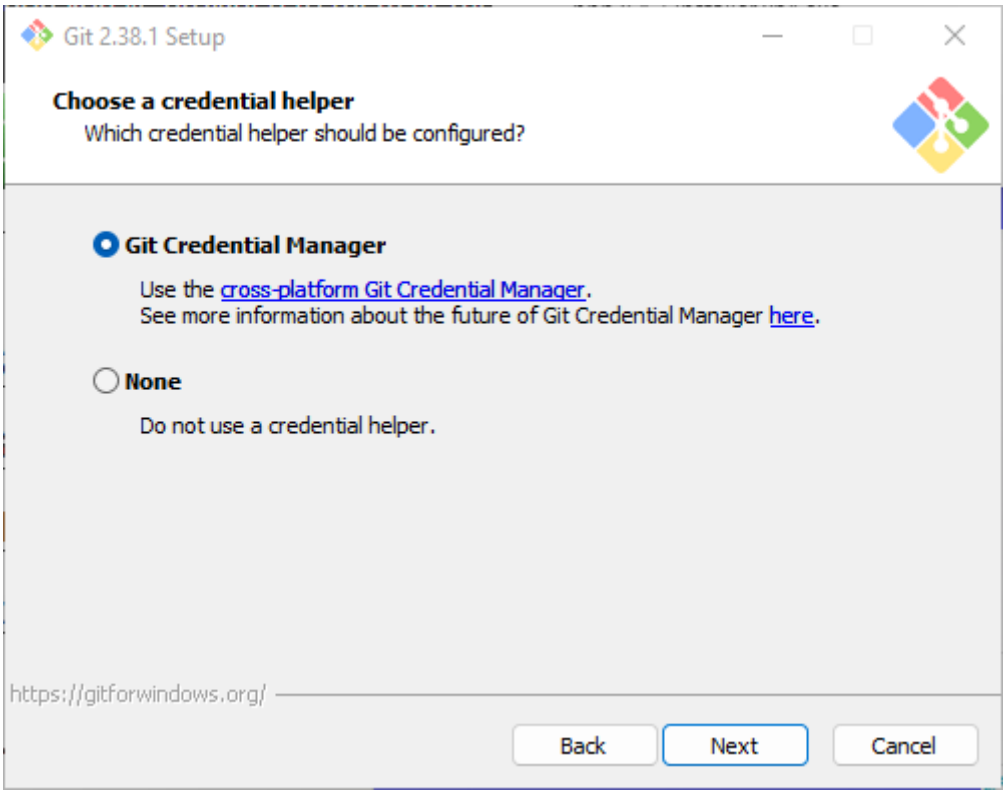

Git 2.38.1 Setup
Choosing the SSH executable
Which Secure Shell client program would you like Git to use?
Use bundled OpenSSH
This uses ssh.exe that comes with Git.
Use external OpenSSH
NEW! This uses an external ssh.exe. Git will not install its own OpenSSH (and related) binaries but use them as found on the PATH.
https://gitforwindows.org/
Back
Next
Cancel

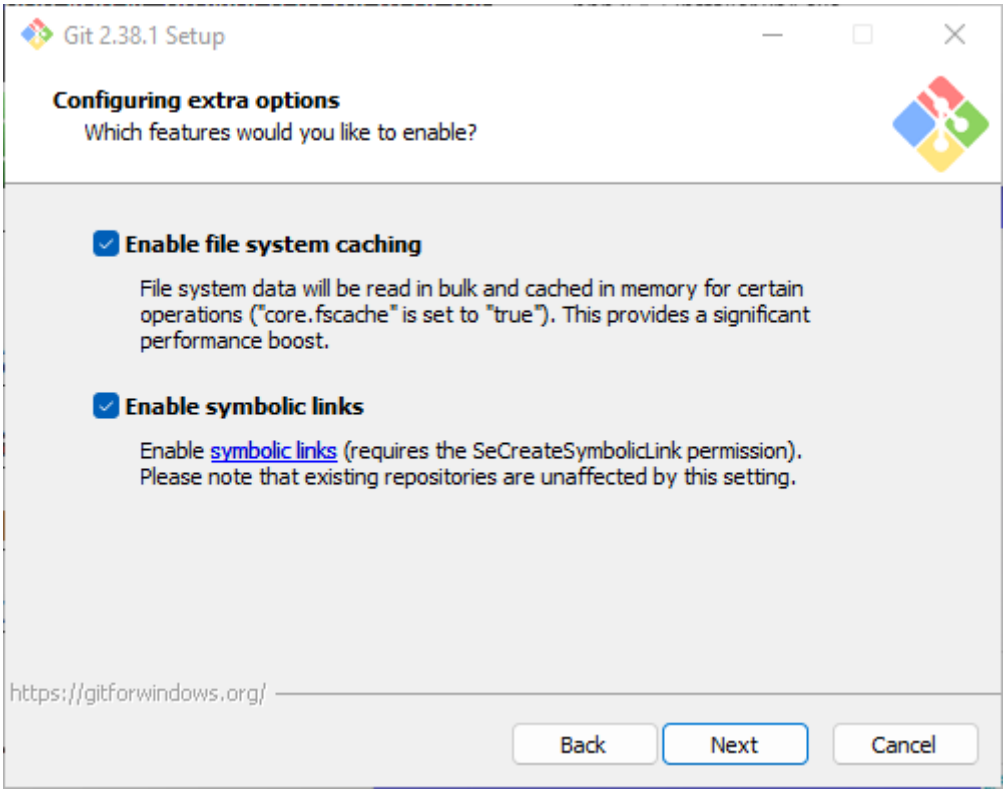

Git 2.38.1 Setup
Choosing HTTPS transport backend
Which SSL/TLS library would you like Git to use for HTTPS connections?
Use the OpenSSL library
Server certificates will be validated using the ca-bundle.crt file.
Use the native Windows Secure Channel library
Server certificates will be validated using Windows Certificate Stores.
This option also allows you to use your company's internal Root CA certificates
distributed e.g. via Active Directory Domain Services.
https://gitforwindows.org/
Back
Next
Cancel

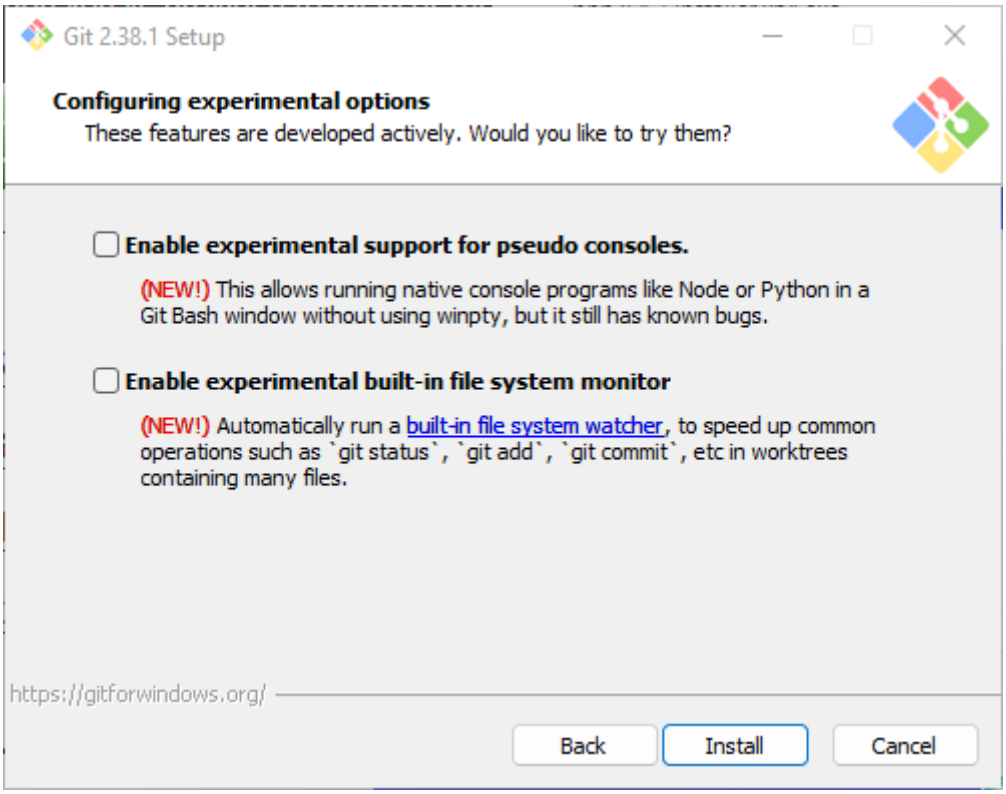

Git 2.38.1 Setup
Configuring the line ending conversions
How should Git treat line endings in text files?
Checkout Windows-style, commit Unix-style line endings
Git will convert LF to CRLF when checking out text files. When committing
text files, CRLF will be converted to LF. For cross-platform projects,
this is the recommended setting on Windows ("core.autocrlf" is set to "true").
Checkout as-is, commit Unix-style line endings
Git will not perform any conversion when checking out text files. When
committing text files, CRLF will be converted to LF. For cross-platform projects,
this is the recommended setting on Unix ("core.autocrlf" is set to "input").
Checkout as-is, commit as-is
Git will not perform any conversions when checking out or committing
text files. Choosing this option is not recommended for cross-platform
projects ("core.autocrlf" is set to "false").
https://gitforwindows.org/
Back
Next
Cancel


Git 2.38.1 Setup
Configuring the terminal emulator to use with Git Bash
Which terminal emulator do you want to use with your Git Bash?
Use MinTTY (the default terminal of MSYS2)
Git Bash will use MinTTY as terminal emulator, which sports a resizable window, non-rectangular selections and a Unicode font. Windows console programs (such as interactive Python) must be launched via `winpty` to work in MinTTY.
Use Windows' default console window
Git will use the default console window of Windows ("cmd.exe"), which works well with Win32 console programs such as interactive Python or node.js, but has a very limited default scroll-back, needs to be configured to use a Unicode font in order to display non-ASCII characters correctly, and prior to Windows 10 its window was not freely resizable and it only allowed rectangular text selections.
https://gitforwindows.org/
Back
Next
Cancel

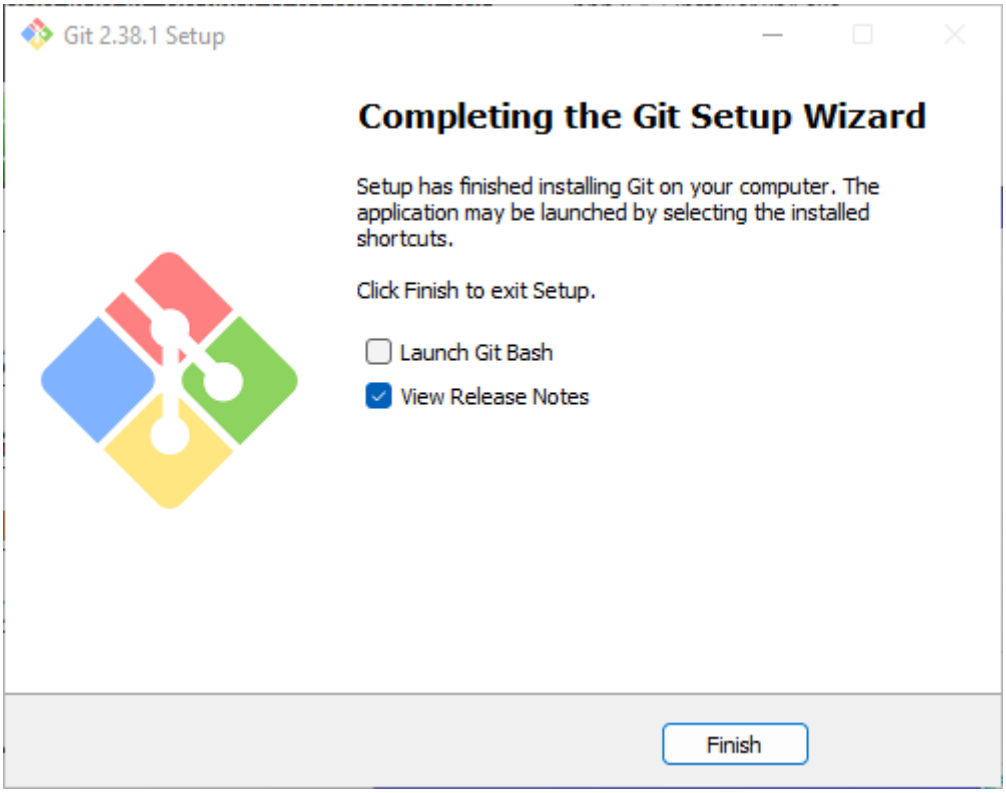

Git 2.38.1 Setup
Choose the default behavior of `git pull`
What should `git pull` do by default?
Default (fast-forward or merge)
This is the standard behavior of `git pull`: fast-forward the current branch to the fetched branch when possible, otherwise create a merge commit.
Rebase
Rebase the current branch onto the fetched branch. If there are no local commits to rebase, this is equivalent to a fast-forward.
Only ever fast-forward
Fast-forward to the fetched branch. Fail if that is not possible.
https://gitforwindows.org/
Back
Next
Cancel


Git 2.38.1 Setup
Choose a credential helper
Which credential helper should be configured?
Git Credential Manager
Use the cross-platform Git Credential Manager.
See more information about the future of Git Credential Manager here.
None
Do not use a credential helper.
https://gitforwindows.org/
Back
Next
Cancel


Git 2.38.1 Setup
Configuring extra options
Which features would you like to enable?
Enable file system caching
File system data will be read in bulk and cached in memory for certain operations ("core.fscache" is set to "true"). This provides a significant performance boost.
Enable symbolic links
Enable symbolic links (requires the SeCreateSymbolicLink permission).
Please note that existing repositories are unaffected by this setting.
https://gitforwindows.org/
Back
Next
Cancel

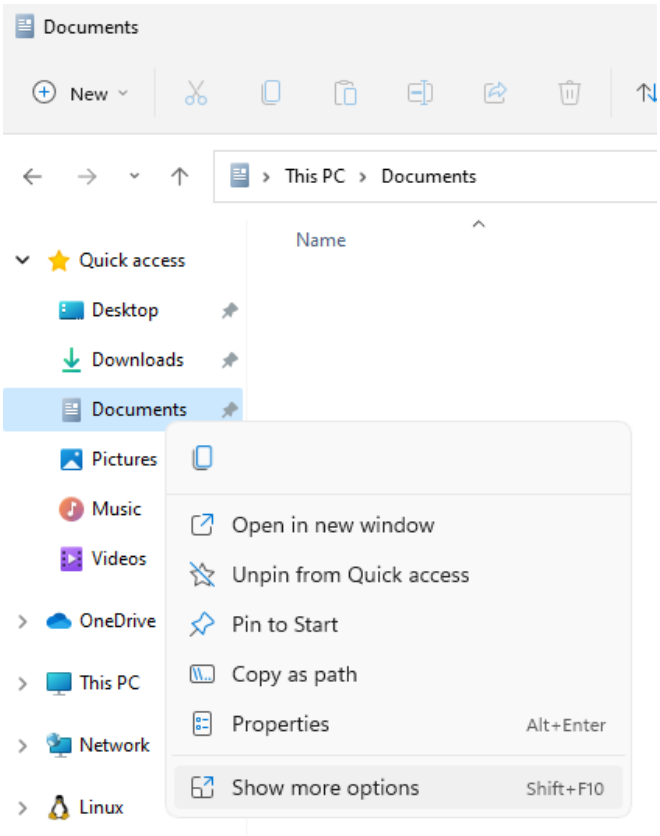


GIT will now be installed and the installation will finish:

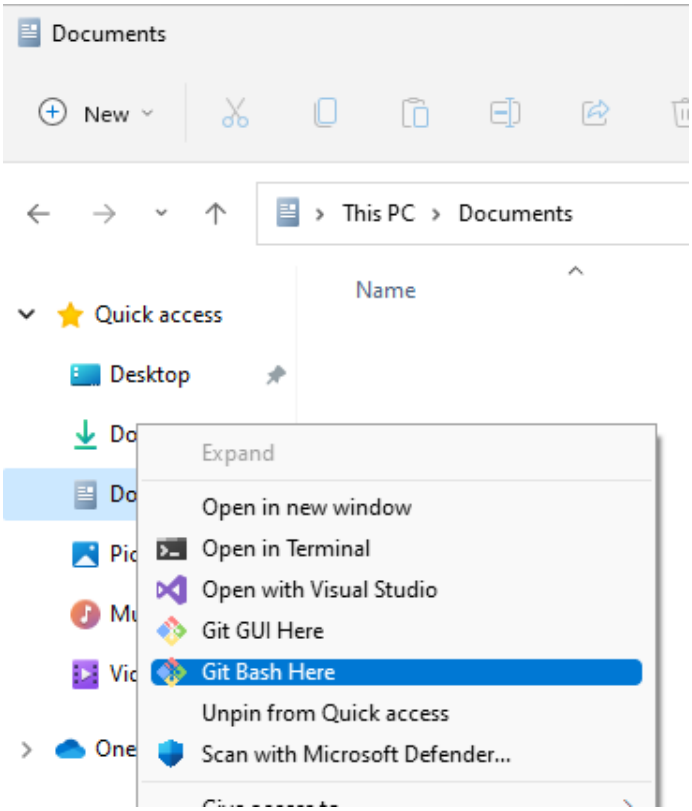


Do the following to clone PNYX from GIT:

Choose “Show more options” on the Documents context menu



Select then “Git Bash Here”



Enter the following:

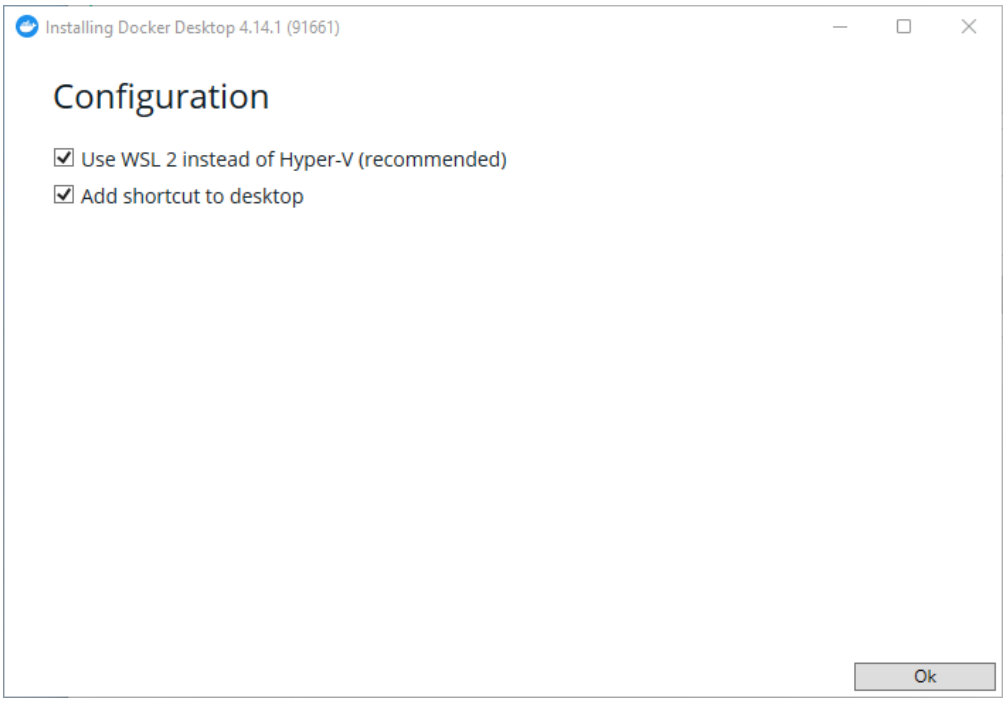
```
git clone https://github.com/NeaBouli/pnyx
cd pnyx
git switch development
```

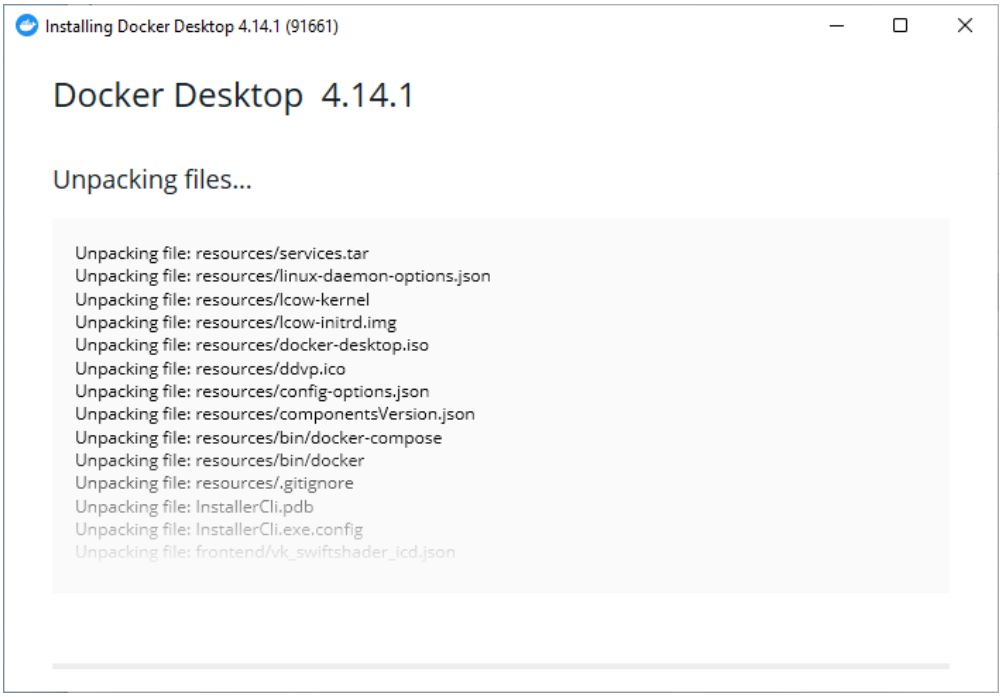
1.2.2 Install Docker Desktop

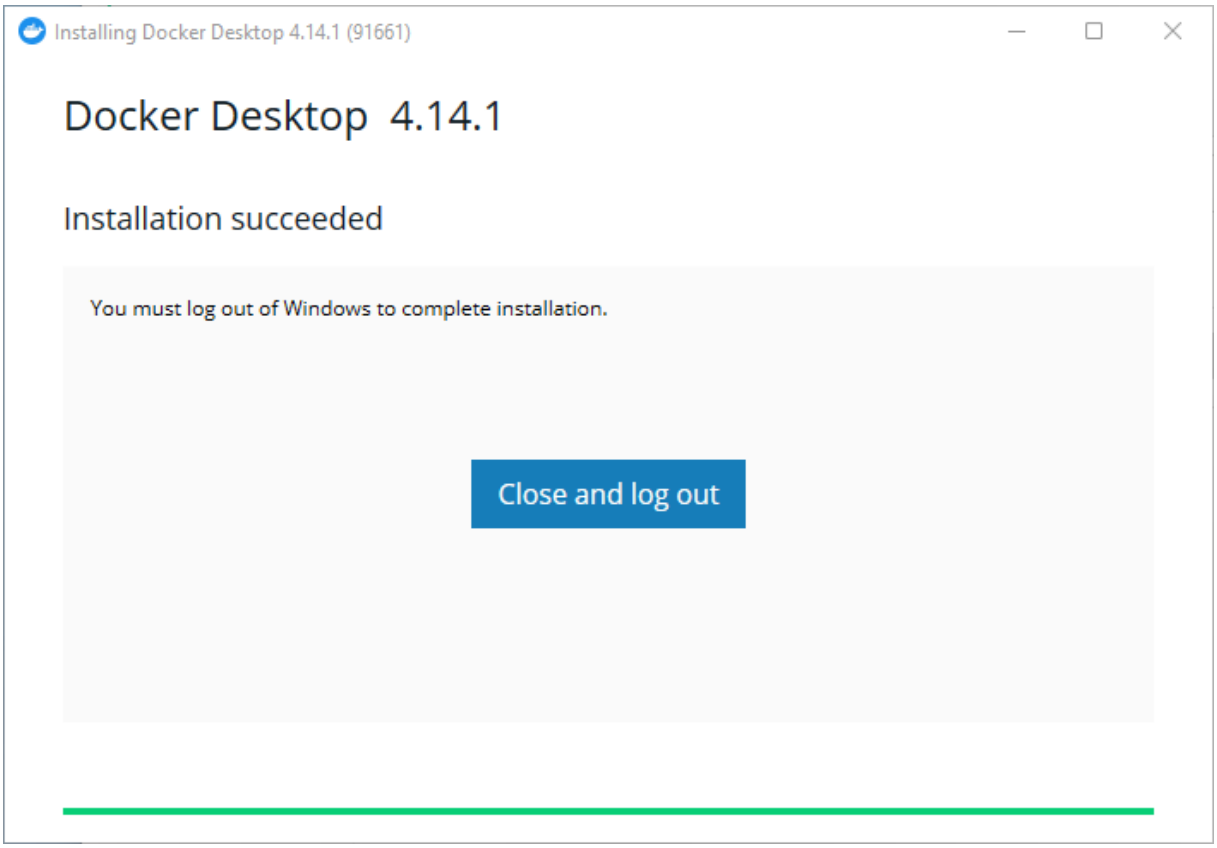
Download and Install Docker Desktop in the next step

https://www.docker.com/products/docker-desktop/

Installation works as follows:





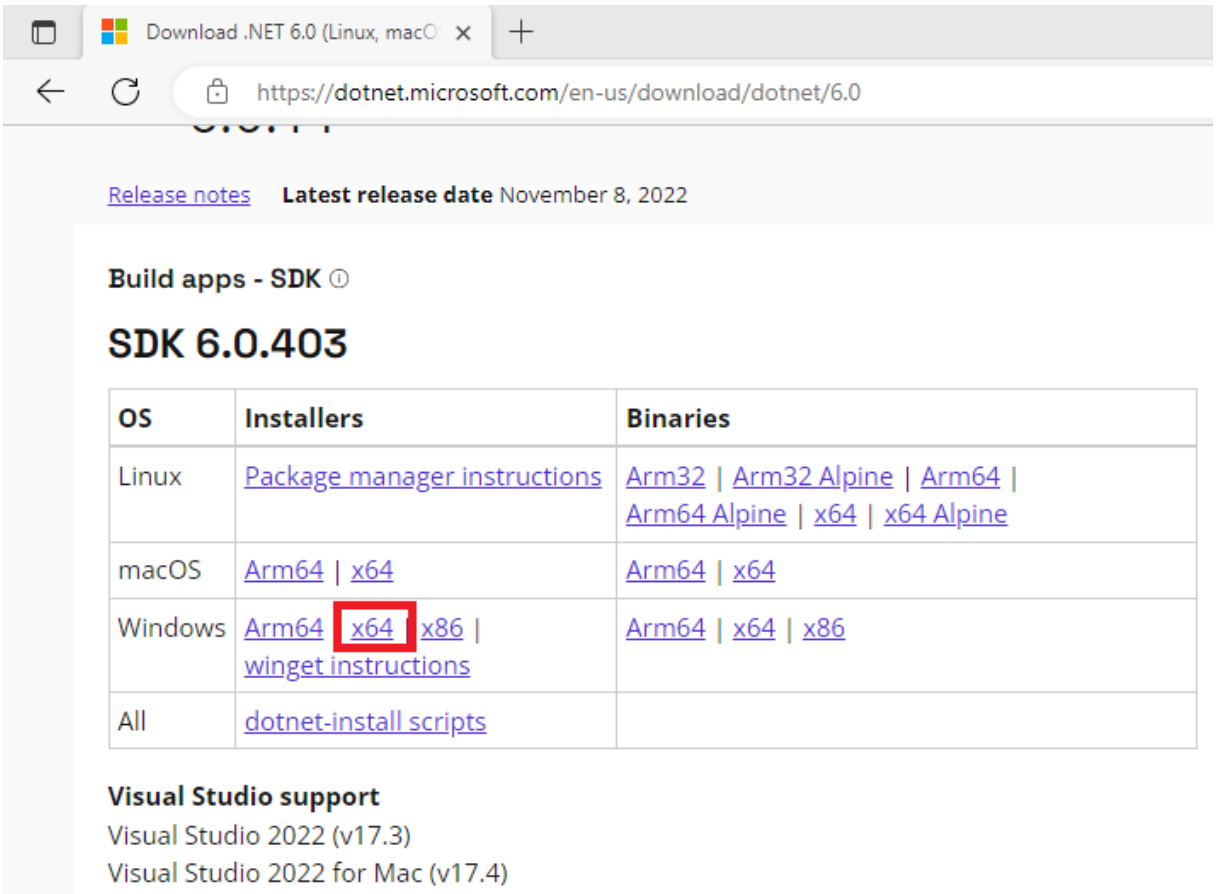

Installing Docker Desktop 4.14.1 (91661)
Docker Desktop 4.14.1
Installation succeeded
You must log out of Windows to complete installation.
Close and log out

1.2.3 Install .NET 6.0 SDK

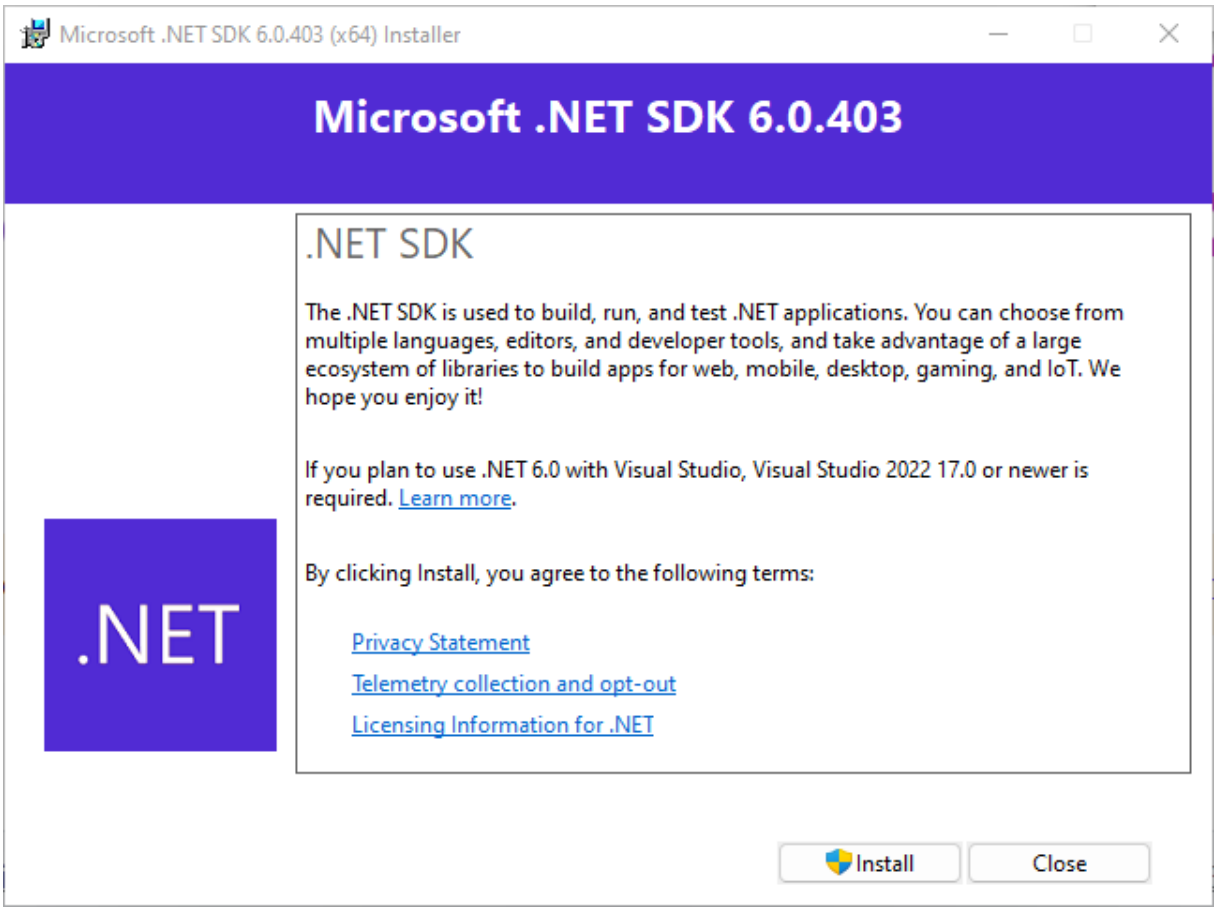
Now download the .NET 6.0 SDK

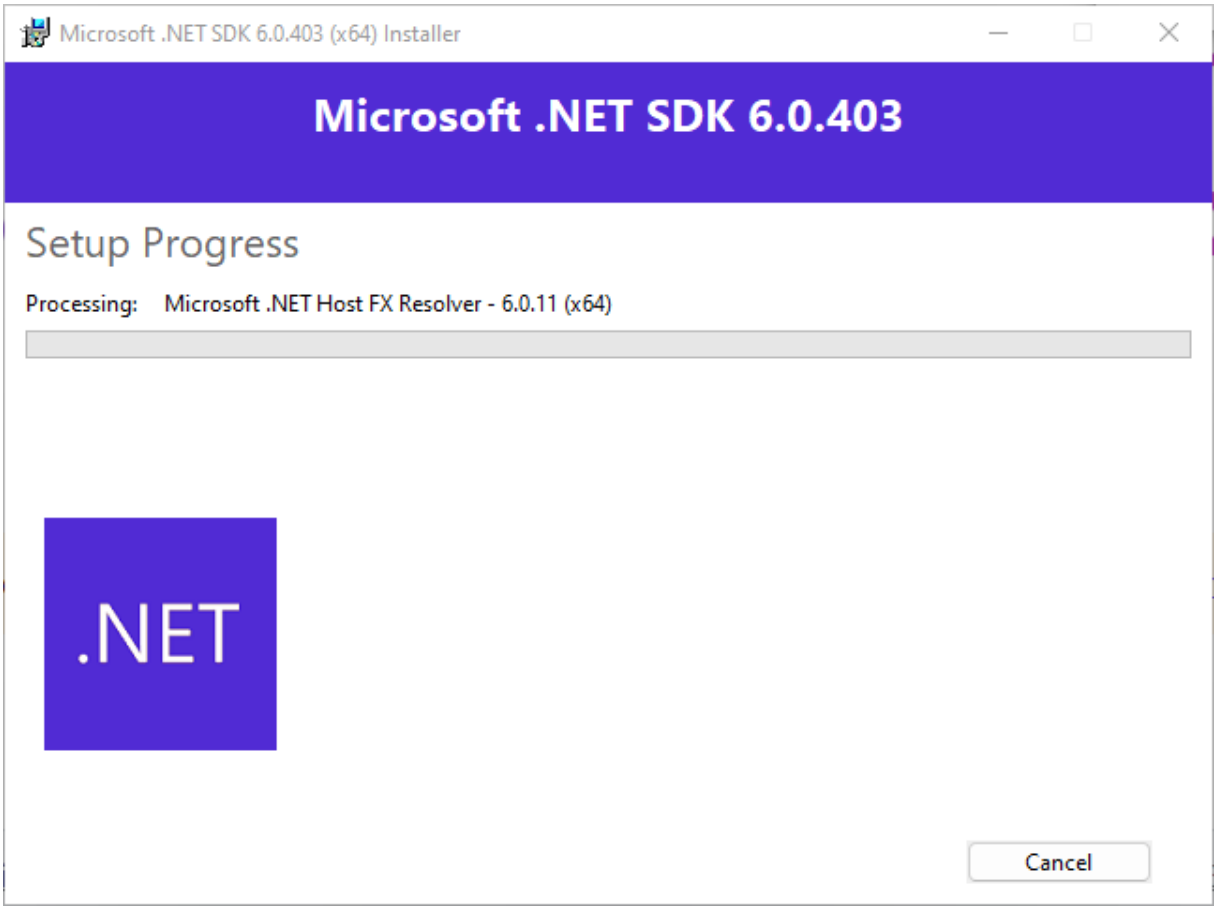
https://dotnet.microsoft.com/en-us/download/dotnet/6.0

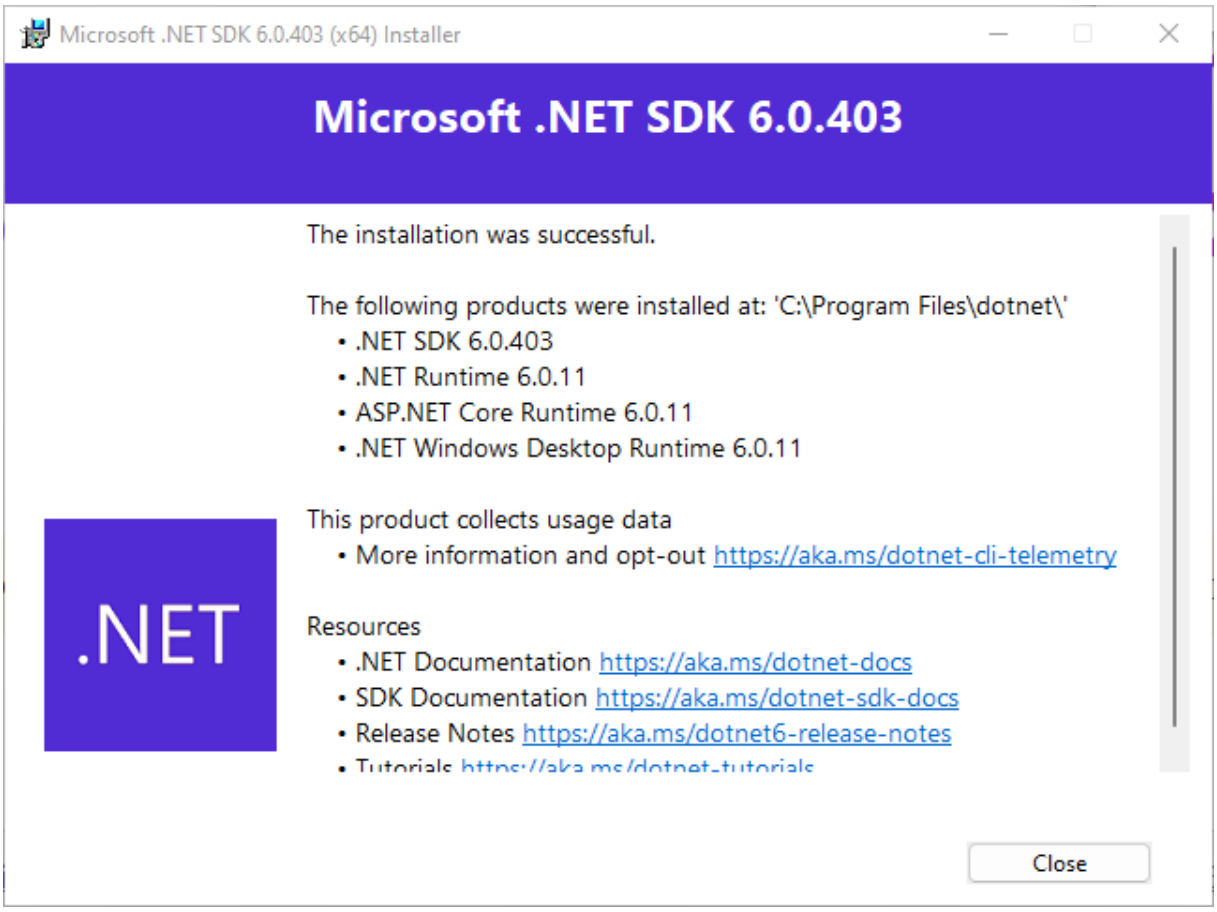
You should download and install the Windows x64 SDK



Installation works as follows:





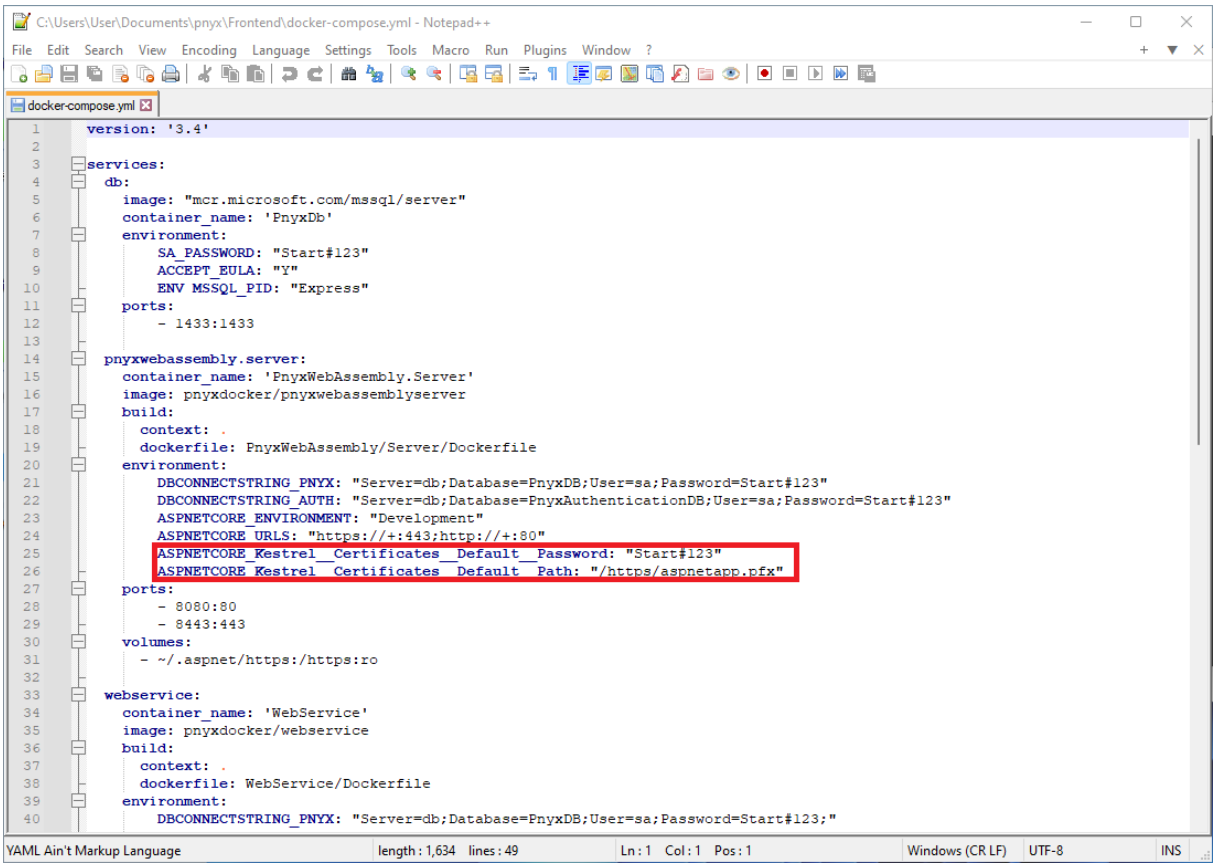

Microsoft .NET SDK 6.0.403 (x64) Installer
Microsoft .NET SDK 6.0.403
The installation was successful.
The following products were installed at: 'C:\Program Files\dotnet\'
• .NET SDK 6.0.403
• .NET Runtime 6.0.11
• ASP.NET Core Runtime 6.0.11
• .NET Windows Desktop Runtime 6.0.11
.NET
This product collects usage data
• More information and opt-out https://aka.ms/dotnet-cli-telemetry
Resources
• .NET Documentation https://aka.ms/dotnet-docs
• SDK Documentation https://aka.ms/dotnet-sdk-docs
• Release Notes https://aka.ms/dotnet6-release-notes
Close

1.2.4 Register https cert

In the next step self-signed https-certs must be registered as described here:

https://learn.microsoft.com/de-de/aspnet/core/security/docker-compose-https?view=aspnetcore-6.0

Create a new https cert with password as defined in the pnyx docker compose file:



Open a MS-DOS console as administrator within the pnyx frontend folder (C:\Users\<user>\Documents\pnyx\Frontend) and enter the following:

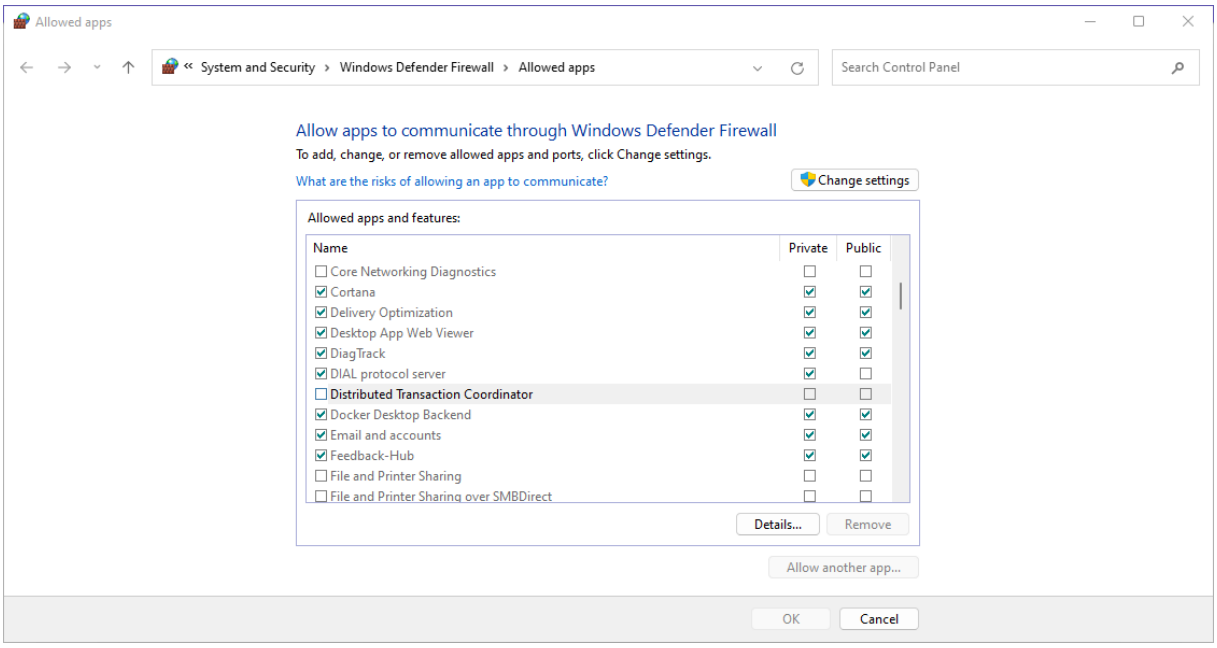
```
dotnet dev-certs https -ep %USERPROFILE%\.aspnet\https\aspnetapp.pfx -p Start#123
```

```
dotnet dev-certs https –trust
```

After this you can start docker with:

```
docker-compose up
```

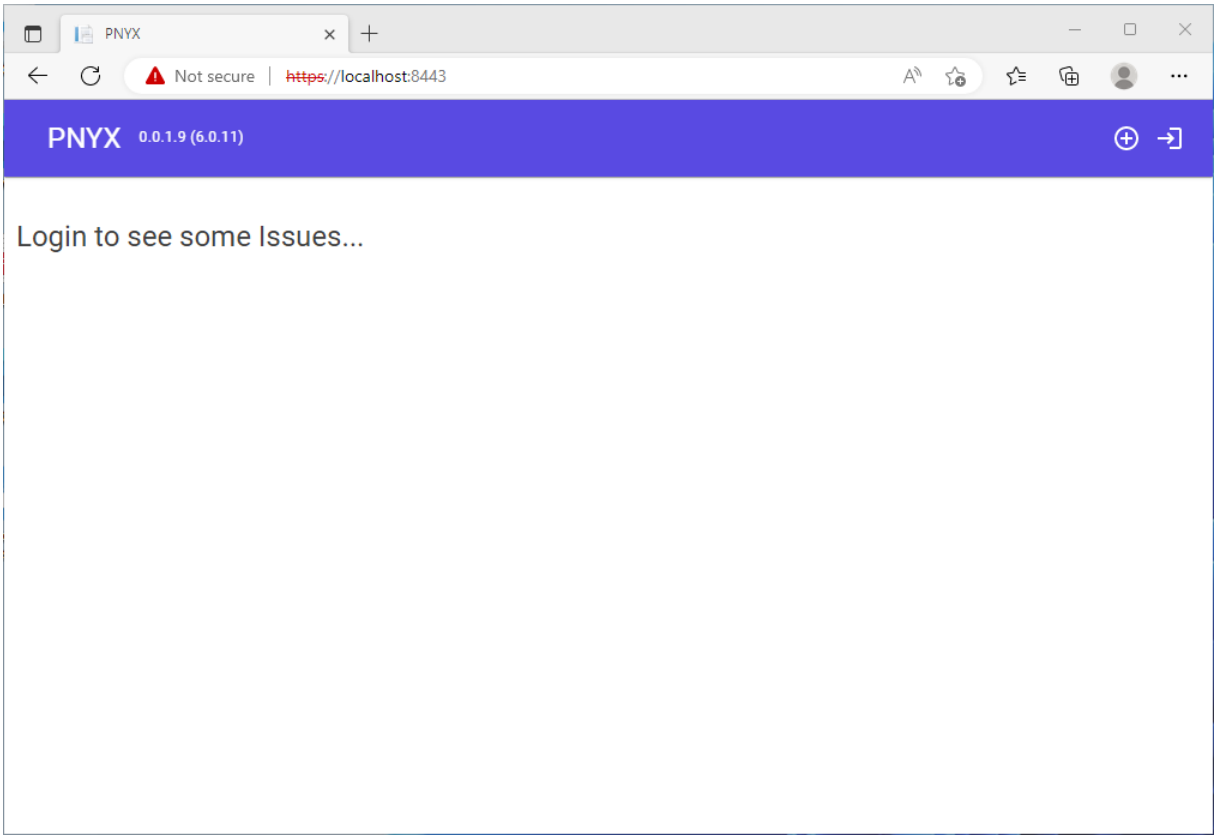
If your computer asks you to allow docker access over the firewall you have to allow it.



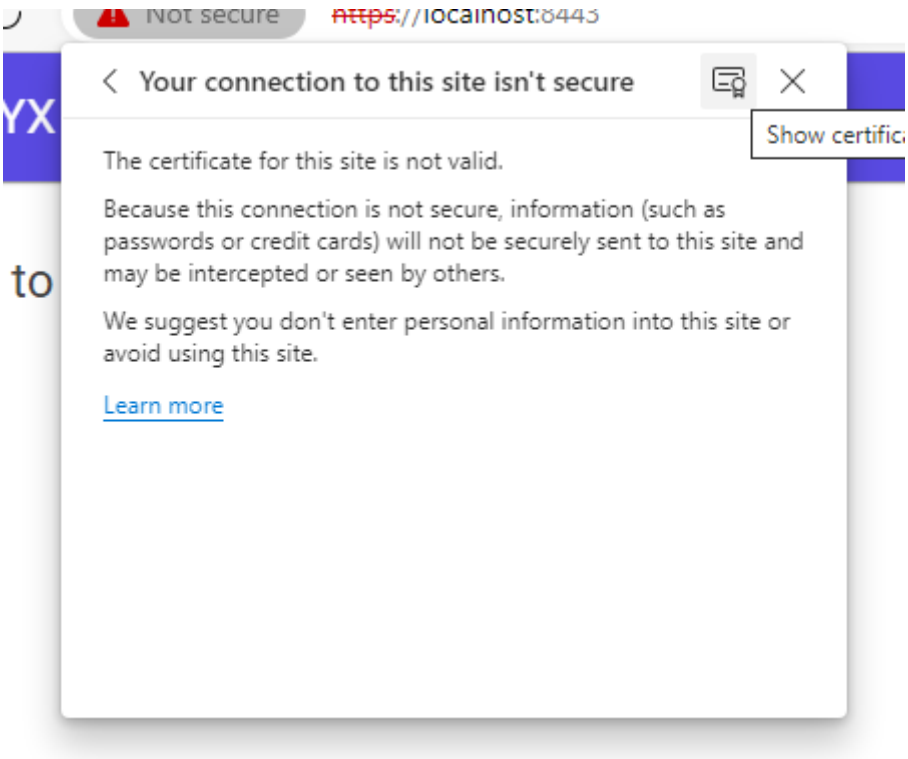
There might be DB updgrades – however docker should come up.

If PNYXWebAssembly.Server is not started you should start it manually after all docker tasks are finished.

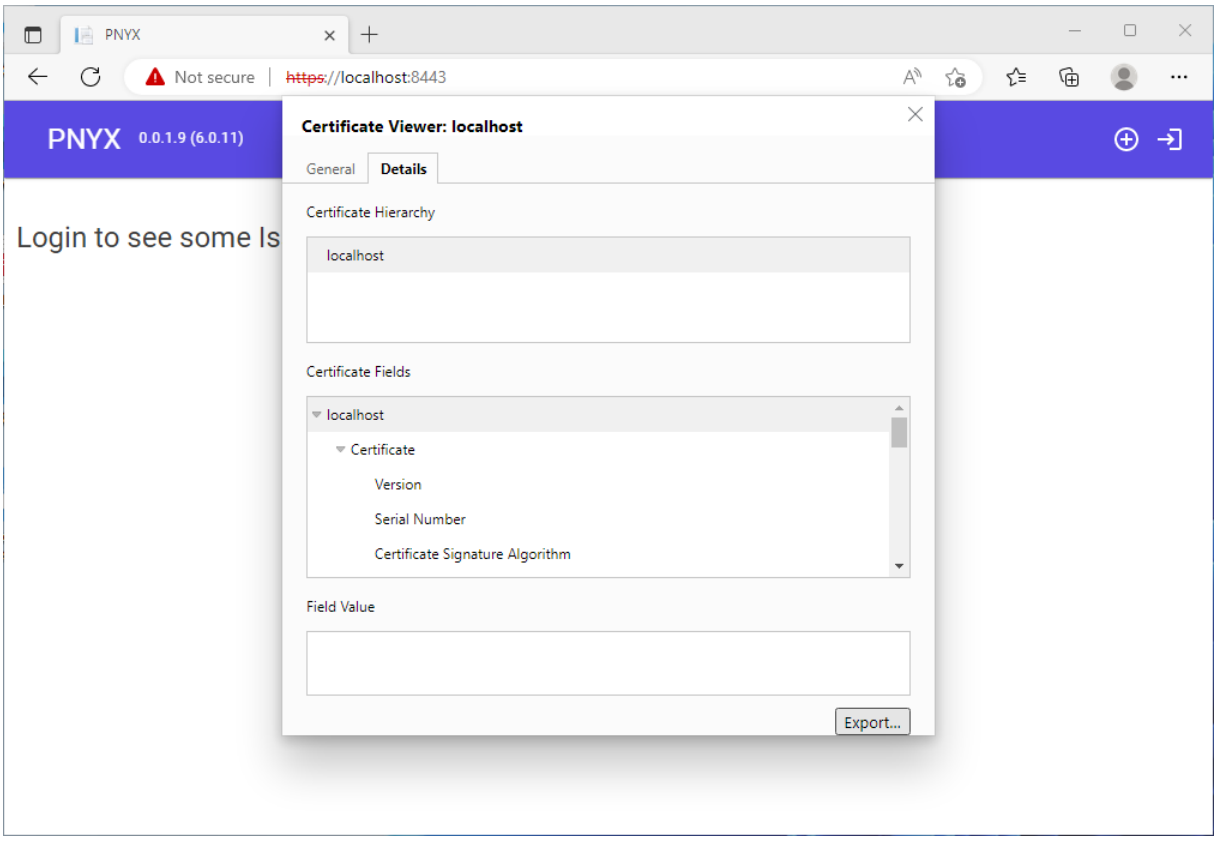
Now you should be able to open PNYX in the web browser as follows:

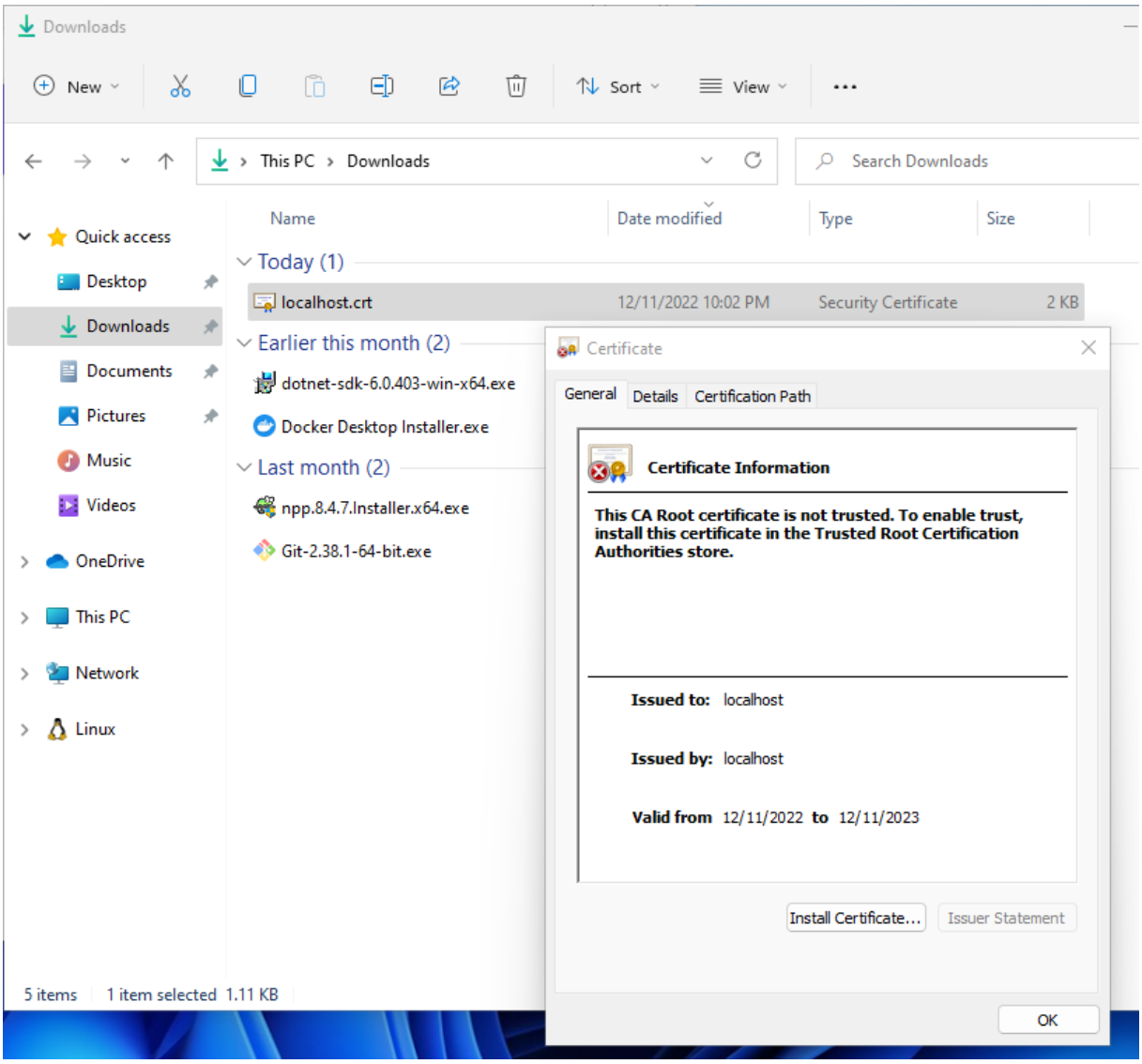


You should click onto the certificate and download it:

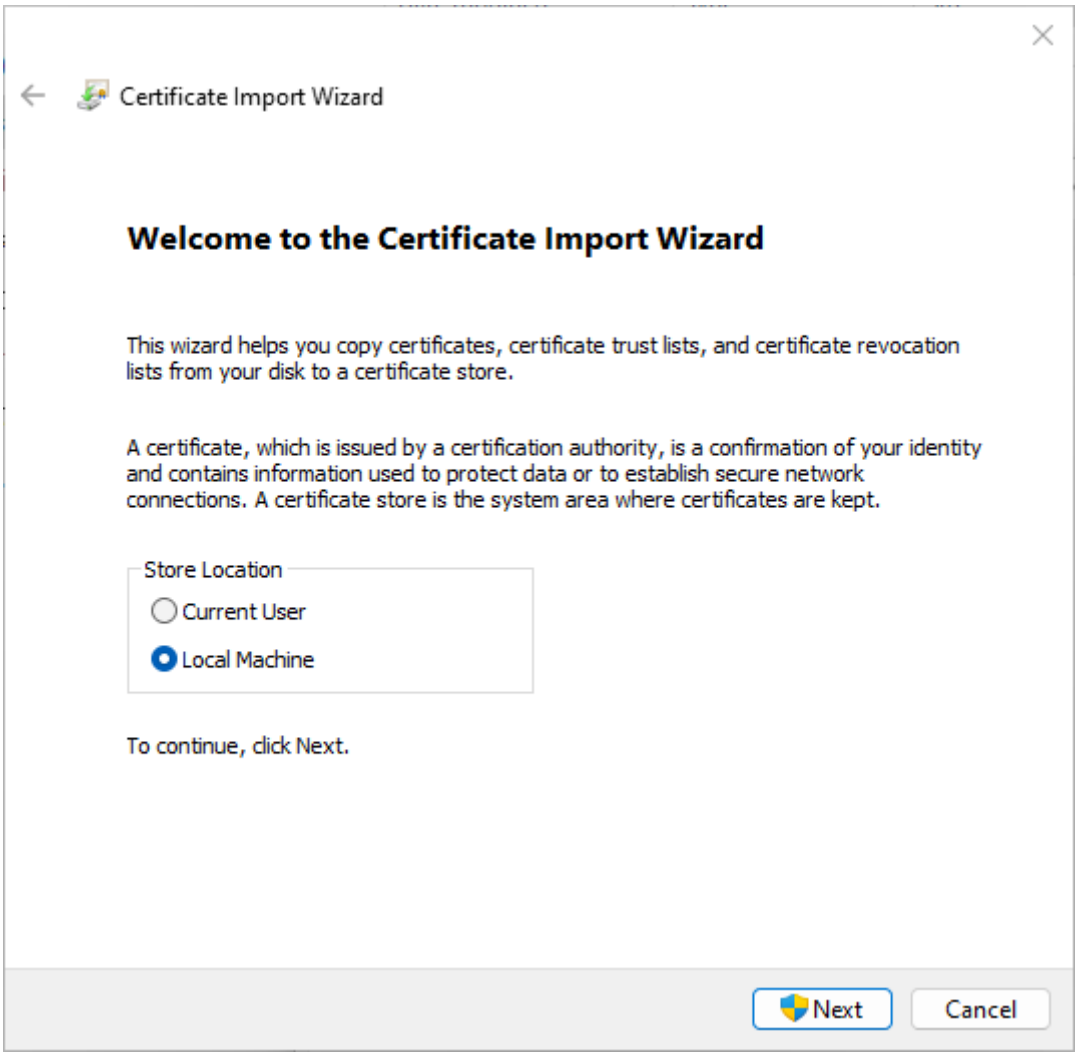


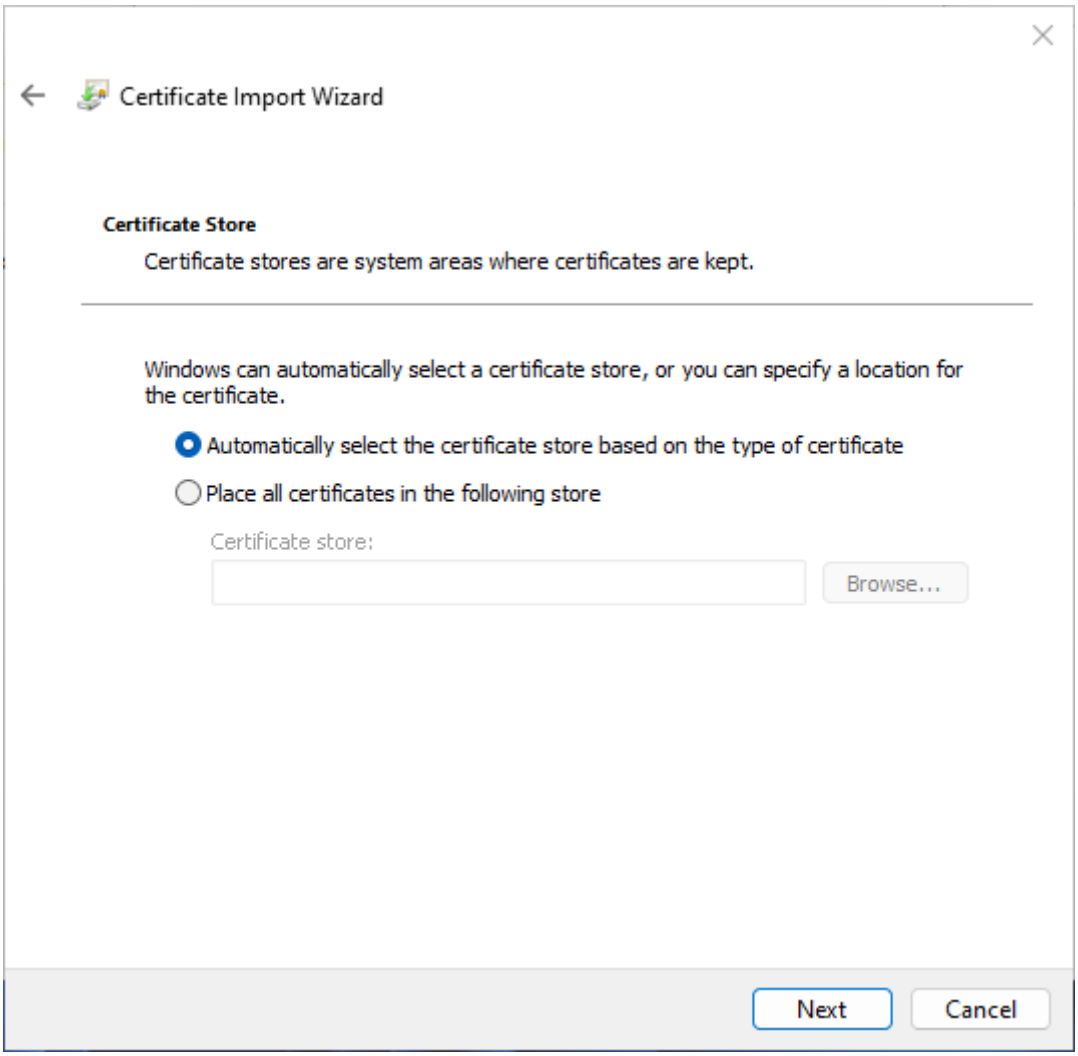
Click export to export the certificate





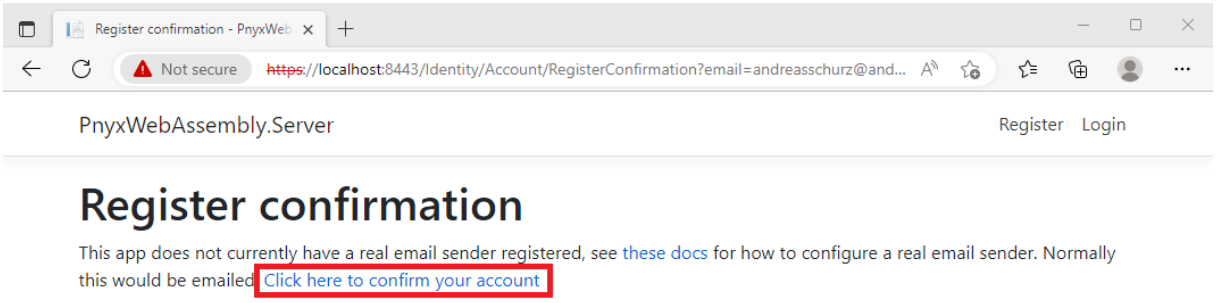
Double click the downloaded crt file and click “Install Certificate”


Certificate Import Wizard
Welcome to the Certificate Import Wizard
This wizard helps you copy certificates, certificate trust lists, and certificate revocation lists from your disk to a certificate store.
A certificate, which is issued by a certification authority, is a confirmation of your identity and contains information used to protect data or to establish secure network connections. A certificate store is the system area where certificates are kept.
Store Location
Current User
Local Machine
To continue, click Next.
Next
Cancel



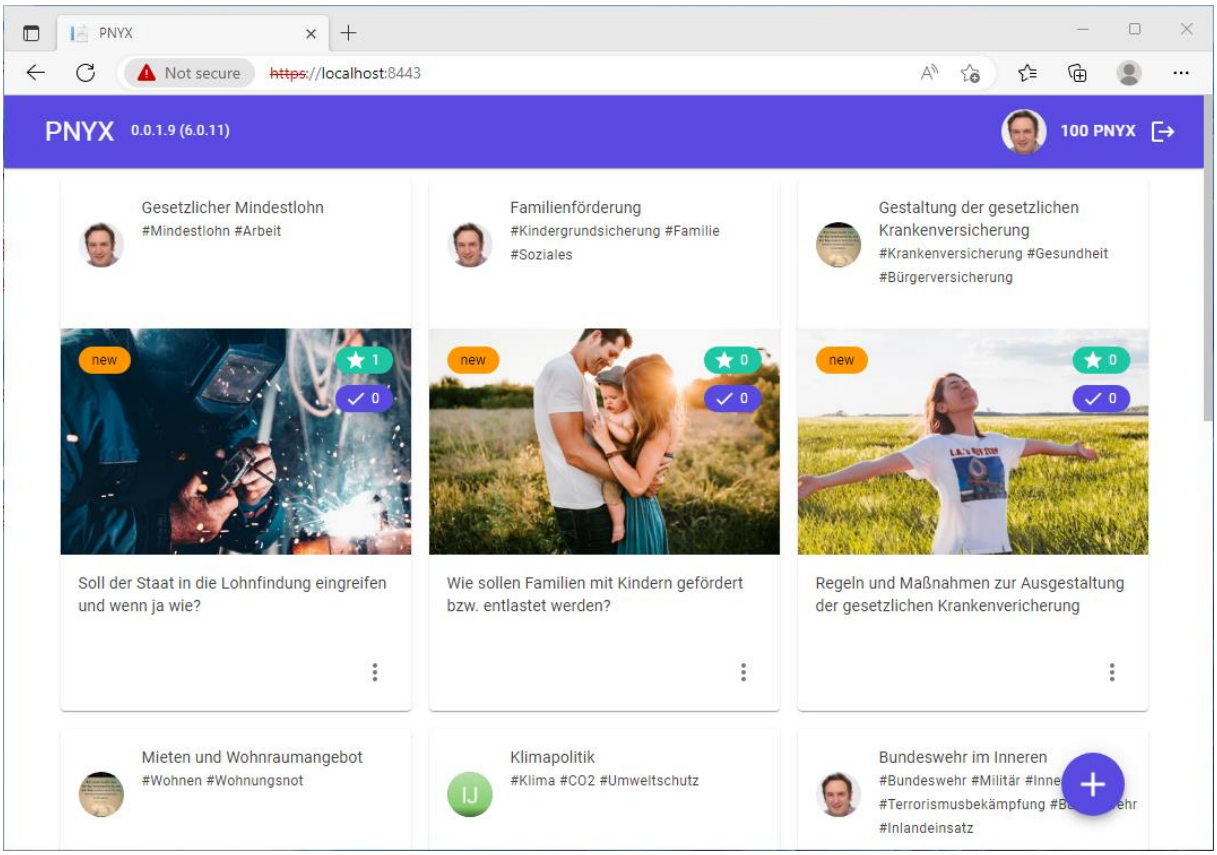
Now the PNYX on localhost is still not secure but will be loaded without a warning.

For better results for the test you should register the following user: andreasschurz@andreasschurz.de by using the + button on the upper right corner.

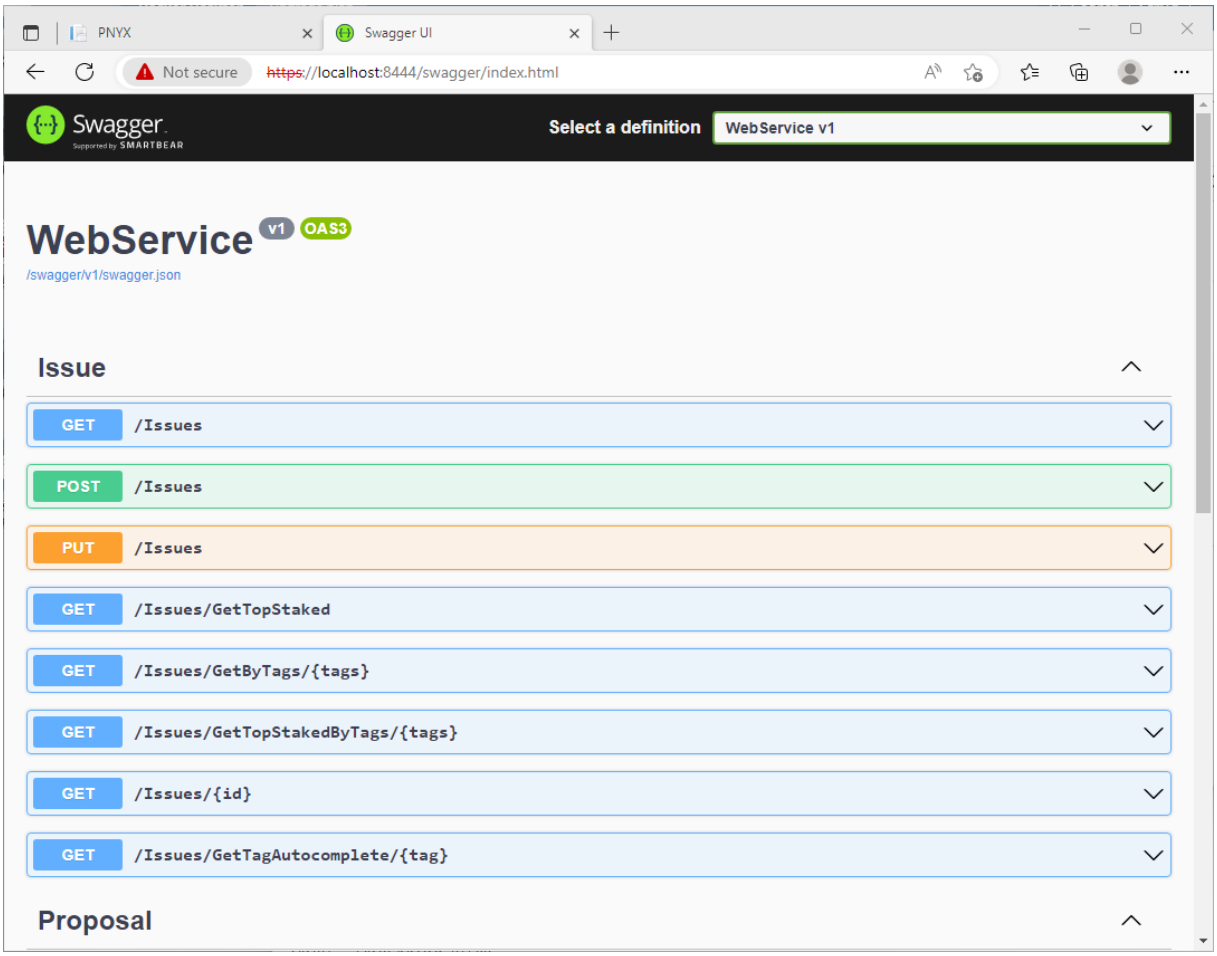


Click on the link above to register your test user

After this you can login using the link on the “Register Email” page and the PNXY demo website comes up.



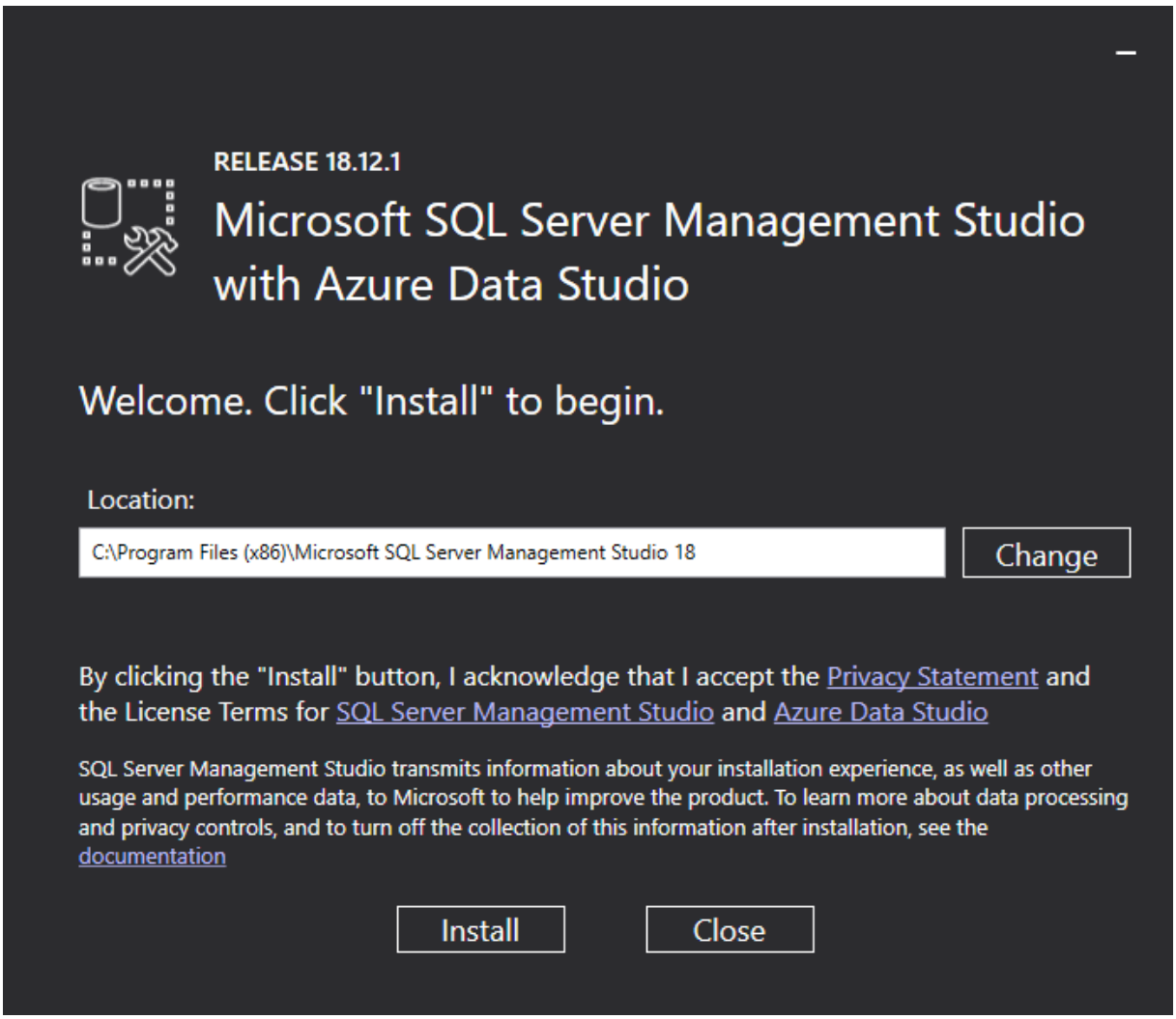
You can also test the swagger access to the webservice and models:



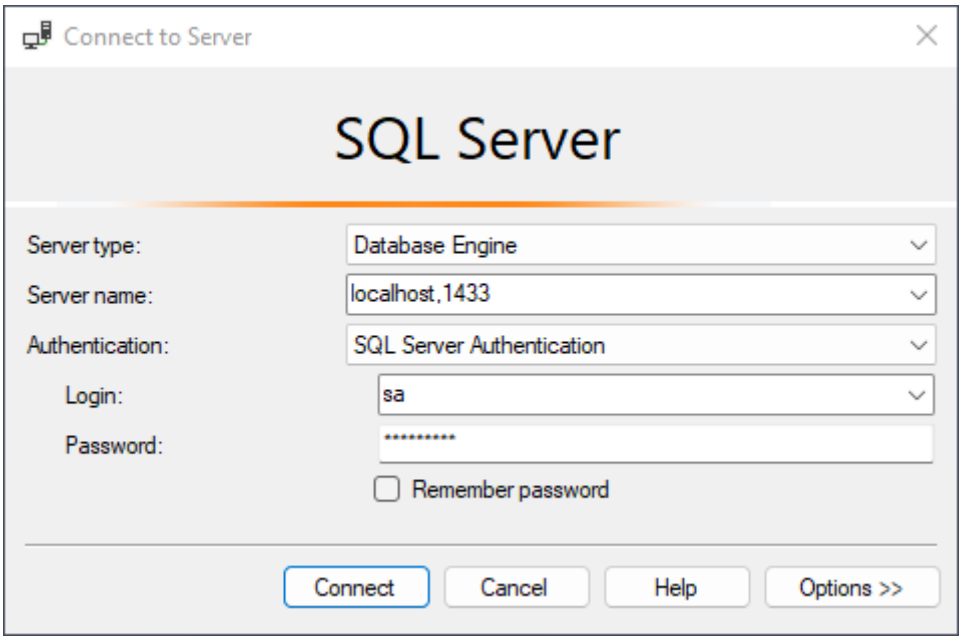
1.2.5 Database Management

You can download and install https://aka.ms/ssmsfullsetup if you want to access the database.

Installation works as follows

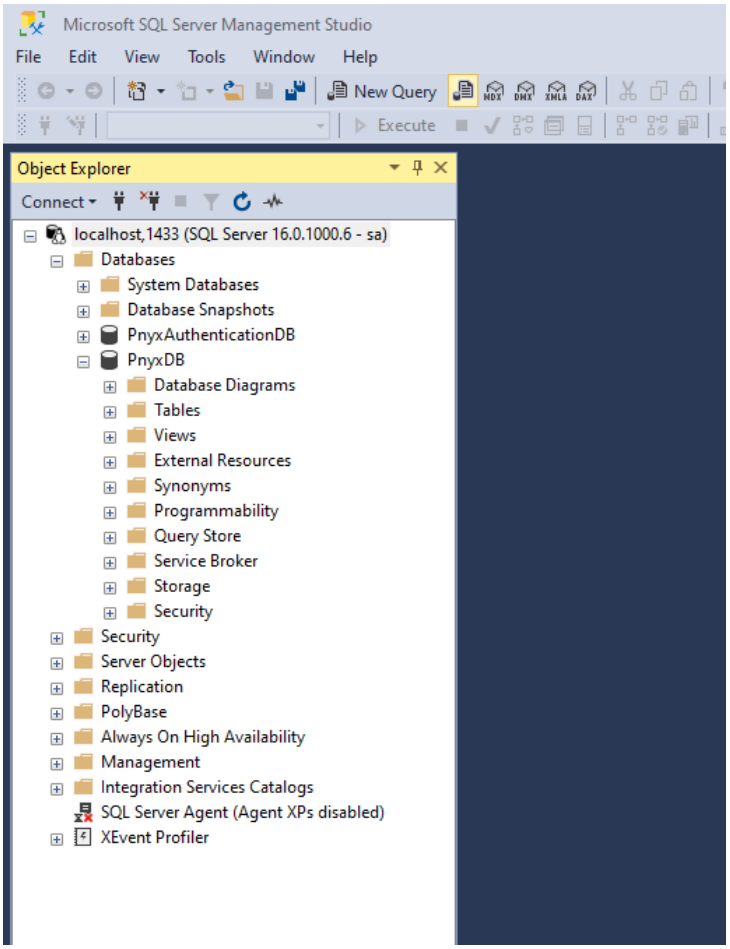


Start “SQL Server Management Studio” after installation and connect to your localhost on port 1433 as follows:



The default password for the sa user is “Start#123”

Now DB administration tasks can be performed:



1.3 Linux Installation on Debian

1.3.1 Add regular user to suders

If you have installed Debian please make sure that your regular user, e. g. “debian” has sudo permissions to ensure that do the following

```
su root
nano /etc/sudoers
```

Add your user to the file as shown below

Example: `debian ALL=(ALL:ALL) ALL`

This allows then to use the sudo command with your user later.