Propel

Reach your servers

**Configuration Management**

Contents

[Development Phases 4](#_Toc160200265)

[First time deployment 5](#_Toc160200266)

[Node.js 5](#_Toc160200267)

[Node Version Manager 5](#_Toc160200268)

[MongoDB 6](#_Toc160200269)

[Additional tools and steps 10](#_Toc160200270)

[ActiveDirectory Module 10](#_Toc160200271)

[AWS Tools for Windows Powershell 10](#_Toc160200272)

[PostgreSQL ODBC driver 11](#_Toc160200273)

[Preparing development environment 13](#_Toc160200274)

[Cloning the repo in your local 13](#_Toc160200275)

[Propel project scaffolding 14](#_Toc160200276)

[Running Propel Tests 15](#_Toc160200277)

[Running Propel in your Dev environment 16](#_Toc160200278)

[Run Propel frontend inside an Electron app in your DEV environment 16](#_Toc160200279)

[Deploying Propel 19](#_Toc160200280)

Figures

[Figure 1 Keep this unchecked: "Tools for Native Modules" 5](file:///C:\DATA\Dev\Propel\propel\doc\Propel%20-%20Configuration%20Management.docx#_Toc160200281)

[Figure 2 - Installing latest Node.js version 6](file:///C:\DATA\Dev\Propel\propel\doc\Propel%20-%20Configuration%20Management.docx#_Toc160200282)

[Figure 3 - Using an installing Node.js version 6](file:///C:\DATA\Dev\Propel\propel\doc\Propel%20-%20Configuration%20Management.docx#_Toc160200283)

[Figure 2 - Checking Node.js versions 7](file:///C:\DATA\Dev\Propel\propel\doc\Propel%20-%20Configuration%20Management.docx#_Toc160200284)

[Figure 3 - Choose complete setup type 7](file:///C:\DATA\Dev\Propel\propel\doc\Propel%20-%20Configuration%20Management.docx#_Toc160200285)

[Figure 4 - Mongo DB site 7](file:///C:\DATA\Dev\Propel\propel\doc\Propel%20-%20Configuration%20Management.docx#_Toc160200286)

[Figure 5 - Installing Mongo DB as a service 7](file:///C:\DATA\Dev\Propel\propel\doc\Propel%20-%20Configuration%20Management.docx#_Toc160200287)

[Figure 6 - Modified Mongod.cfg file with alternative folders. 8](file:///C:\DATA\Dev\Propel\propel\doc\Propel%20-%20Configuration%20Management.docx#_Toc160200288)

[Figure 7 - Granting full access to the service account. 8](file:///C:\DATA\Dev\Propel\propel\doc\Propel%20-%20Configuration%20Management.docx#_Toc160200289)

[Figure 8 Script to add the Mongo DB path to the PATH system environment variable 8](file:///C:\DATA\Dev\Propel\propel\doc\Propel%20-%20Configuration%20Management.docx#_Toc160200290)

[Figure 9 - Getting the Mongo engine version 9](file:///C:\DATA\Dev\Propel\propel\doc\Propel%20-%20Configuration%20Management.docx#_Toc160200291)

[Figure 10 - Creating the new DBA user 9](file:///C:\DATA\Dev\Propel\propel\doc\Propel%20-%20Configuration%20Management.docx#_Toc160200292)

[Figure 11 - Adding the Authorization security feature. 9](file:///C:\DATA\Dev\Propel\propel\doc\Propel%20-%20Configuration%20Management.docx#_Toc160200293)

[Figure 12 - Enabling auth for all MongoDB databases. 9](file:///C:\DATA\Dev\Propel\propel\doc\Propel%20-%20Configuration%20Management.docx#_Toc160200294)

[Figure 13 - Creating the new DBA user 9](file:///C:\DATA\Dev\Propel\propel\doc\Propel%20-%20Configuration%20Management.docx#_Toc160200295)

[Figure 12 - Installing the PowerShell ActiveDirectory module. 10](file:///C:\DATA\Dev\Propel\propel\doc\Propel%20-%20Configuration%20Management.docx#_Toc160200296)

[Figure 13 - Installing AWS Tools for PowerShell 11](file:///C:\DATA\Dev\Propel\propel\doc\Propel%20-%20Configuration%20Management.docx#_Toc160200297)

[Figure 14 - Setup ODBC drivers in Control panel 11](file:///C:\DATA\Dev\Propel\propel\doc\Propel%20-%20Configuration%20Management.docx#_Toc160200298)

[Figure 15 - Installed ODBC drivers. 12](file:///C:\DATA\Dev\Propel\propel\doc\Propel%20-%20Configuration%20Management.docx#_Toc160200299)

[Figure 16 - Getting the clone URL from Git repo 13](file:///C:\DATA\Dev\Propel\propel\doc\Propel%20-%20Configuration%20Management.docx#_Toc160200300)

[Figure 17 - Cloning the Repository in VS Code 14](file:///C:\DATA\Dev\Propel\propel\doc\Propel%20-%20Configuration%20Management.docx#_Toc160200301)

[Figure 18 - Propel Project folder structure 14](file:///C:\DATA\Dev\Propel\propel\doc\Propel%20-%20Configuration%20Management.docx#_Toc160200302)

[Figure 19 - Propel API and Propel web tests execution. 15](file:///C:\DATA\Dev\Propel\propel\doc\Propel%20-%20Configuration%20Management.docx#_Toc160200303)

[Figure 20 - Options to run the API and the web frontend from the tasks menu. 16](file:///C:\DATA\Dev\Propel\propel\doc\Propel%20-%20Configuration%20Management.docx#_Toc160200304)

[Figure 21 - Task "Build PRODUCTION (Electron Shell Only)" finished run. 17](file:///C:\DATA\Dev\Propel\propel\doc\Propel%20-%20Configuration%20Management.docx#_Toc160200305)

[Figure 22- Starting Propel in an unpackaged Electron app. 17](file:///C:\DATA\Dev\Propel\propel\doc\Propel%20-%20Configuration%20Management.docx#_Toc160200306)

[Figure 23 - Propel running as a desktop app. 18](file:///C:\DATA\Dev\Propel\propel\doc\Propel%20-%20Configuration%20Management.docx#_Toc160200307)

[Figure 24 -Dist folder 19](file:///C:\DATA\Dev\Propel\propel\doc\Propel%20-%20Configuration%20Management.docx#_Toc160200308)

[Figure 25 - Installer file 19](file:///C:\DATA\Dev\Propel\propel\doc\Propel%20-%20Configuration%20Management.docx#_Toc160200309)

# Development Phases

Two phases planned so far:

**Phase I**: The tool will be accessible only from a specific server via Remote Desktop technology, reducing any security risk to the minimum.

**Phase II**: The tool will be accessible from the internet adding security to both APP and API.

**This document will focus on the configuration management aspect of Phase I only**.

# First time deployment

This topic is going to cover everything required to deploy the solution for the first time. We will take care of all the project dependencies that we need to install and configure.

## Node.js

Is **strongly recommended** to use a Node.js version manager to deal with different versions that you can require of node.js. if you would like to use it, please move forward to the Node Version Manager next chapter.

Otherwise: Navigate to Node.js site and download the current LTS, (Long Term Support), version. Then install using all by defect options, except for the “Tools for native modules check that need to be checked manually as you can see in Figure 2.

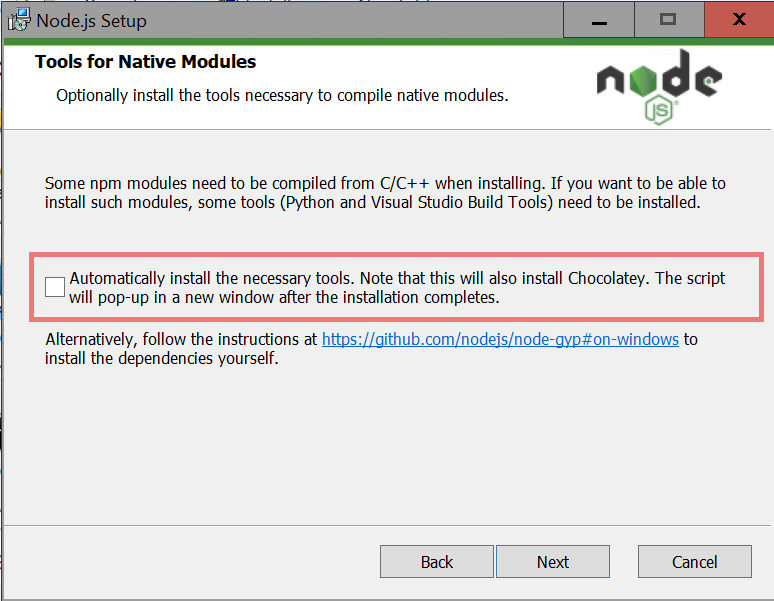


Figure 1 Keep this unchecked: "Tools for Native Modules"

After Node is installed, you can check its version and also the one for Npm, (Node Package Manager), by running the command “node --version” in a PowerShell console.

## Node Version Manager

I will describe here how to install and work with [nvm-windows](https://github.com/coreybutler/nvm-windows), but there are also other Node version managers you can use. This information is extracted from the readme in nvm-windows repository and [this article](https://learn.microsoft.com/en-us/windows/dev-environment/javascript/nodejs-on-windows) from Microsoft Learn.

**If you already have installed Node.js**: You need to start by uninstalling, it. You must do that by following these steps:

1st – In a console run “npm cache clean --force”, ([npm cache help here](https://docs.npmjs.com/cli/v7/commands/npm-cache))

2nd - Uninstall from Programs & Features with the uninstaller.

3rd – Verify there is no running Node.Js processes. If that’s the case kill them.

4th – Look for the following folders. If they exist delete them with all his content:

* *C:\Program Files (x86)\Nodejs*
* *C:\Program Files\Nodejs*
* *%appdata%\npm*
* *%appdata%\npm-cache*
* *%appdata%\Local\Temp\npm-\**
* *C:\Users\{your\_name\_here}\.npmrc*
* *C:\Users\{your\_name\_here}\npmrc*

5th – Restart your computer.

**If you don’t have Node.js installed or you already uninstalled**: You can navigate to the [nvm-windows Releases page](https://github.com/coreybutler/nvm-windows/releases) and download the latest *nvm-setup.zip* file.





Unzip the installer and proceed with the installation, as soon is done, you can open an elevated PowerShell console and type “nvm ls” to list the installed Node.JS versions. Of course, there is none now.

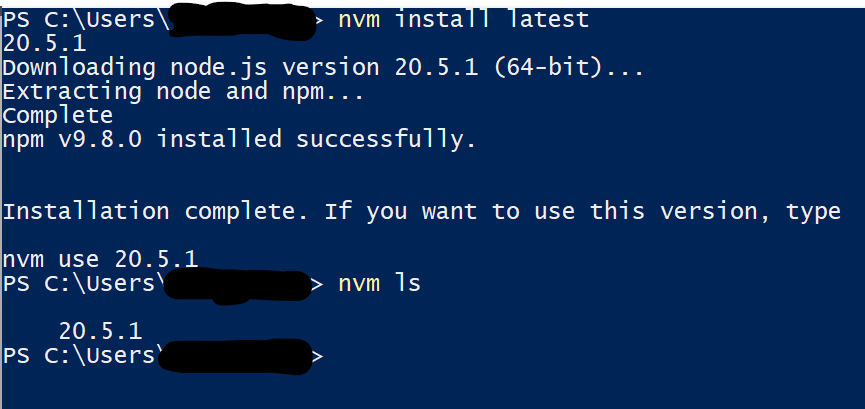


Figure 2 - Installing latest Node.js version

But you can install the latest Node.js version as depicted in Figure 2. Also, you can install a specific Node.js version, please check the product documentation for more details.

But the most important feature , of course, is the ability to switch between installed versions, for that we have the “use” command that you can see in Figure 3

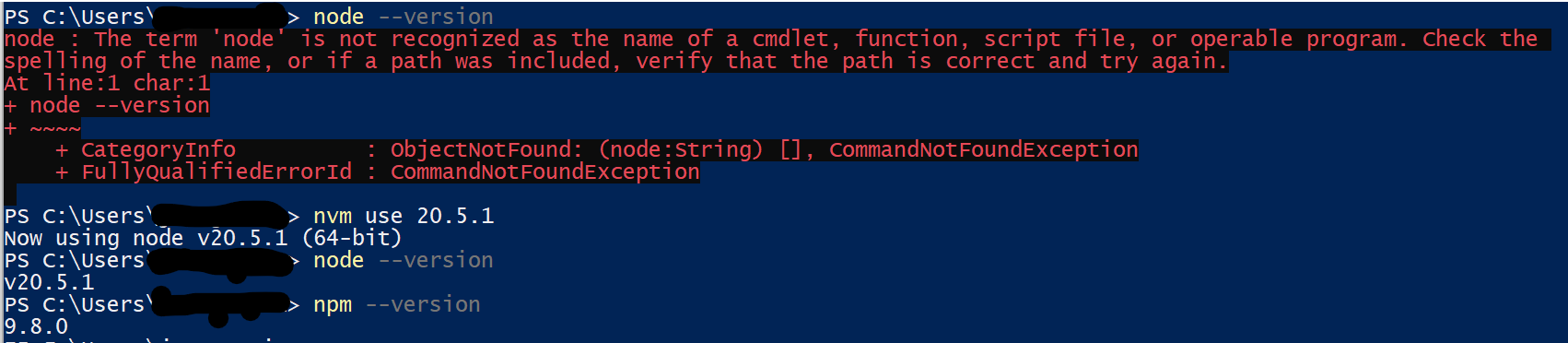


Figure 3 - Using an installing Node.js version

## MongoDB

Download the Mongo DB community server edition and install it with the default values. Choose the version based on the support of the OS. From time-to-time Mongo is dropping the support for different Server versions.

Choose the most up to date version that support your target OS. When the installation prompt for a setup type be sure to select the “Complete” option as you can see in Figure 4.

Next step is to configure MongoDB to run as a service.

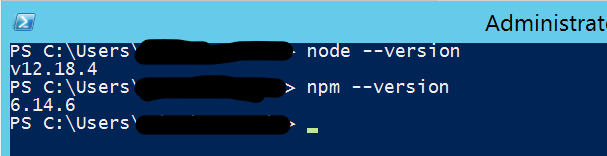


Figure 2 - Checking Node.js versions

Last step in the installer is prompting about to install Mongo DB compass, uncheck the option. You can install later if needed. Also, optionally you can try [Robo 3T](https://robomongo.org/), (a.k.a. RoboMongo), is a light Mongo DB GUI.

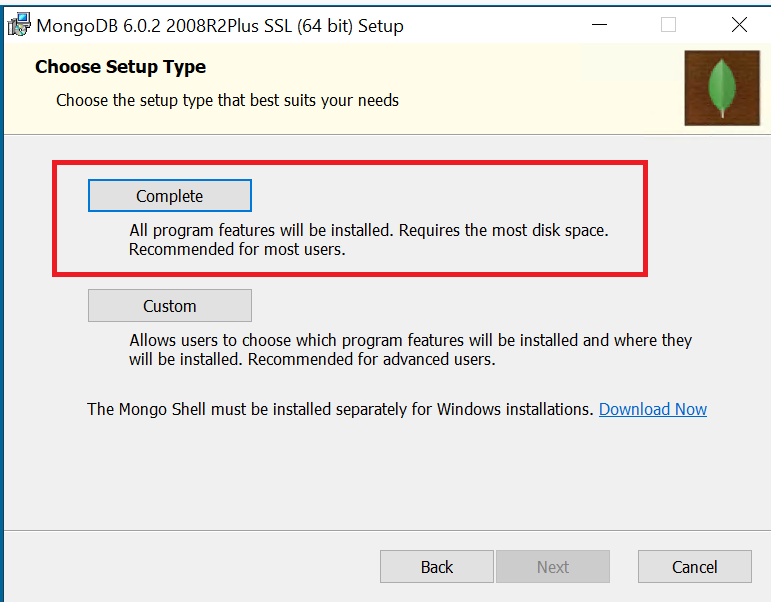


Figure 3 - Choose complete setup type

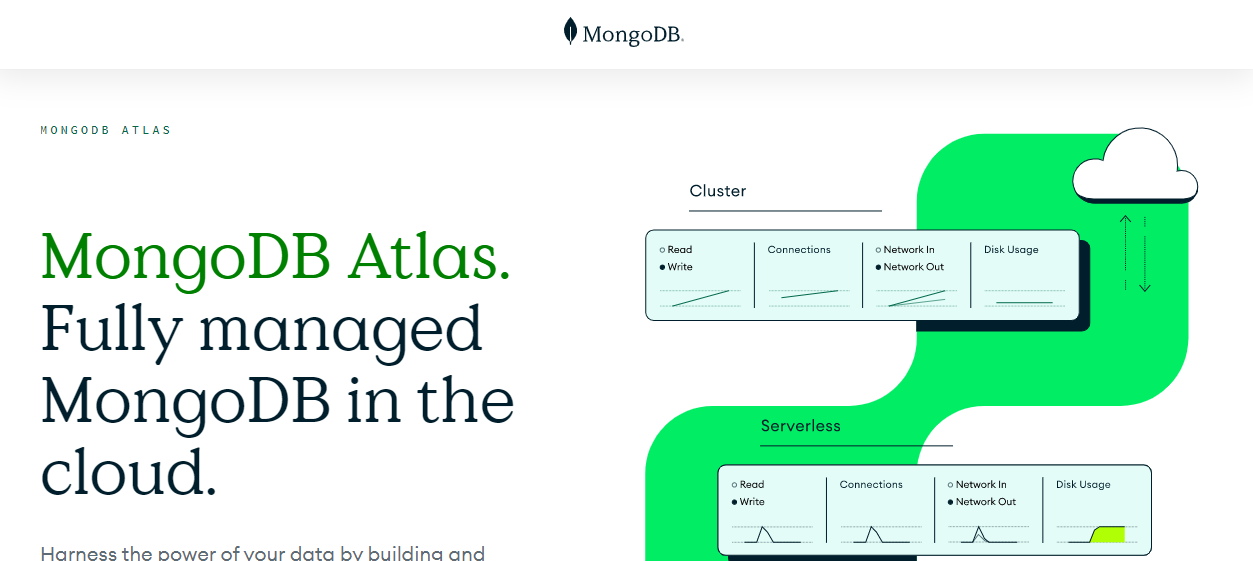


Figure 4 - Mongo DB site

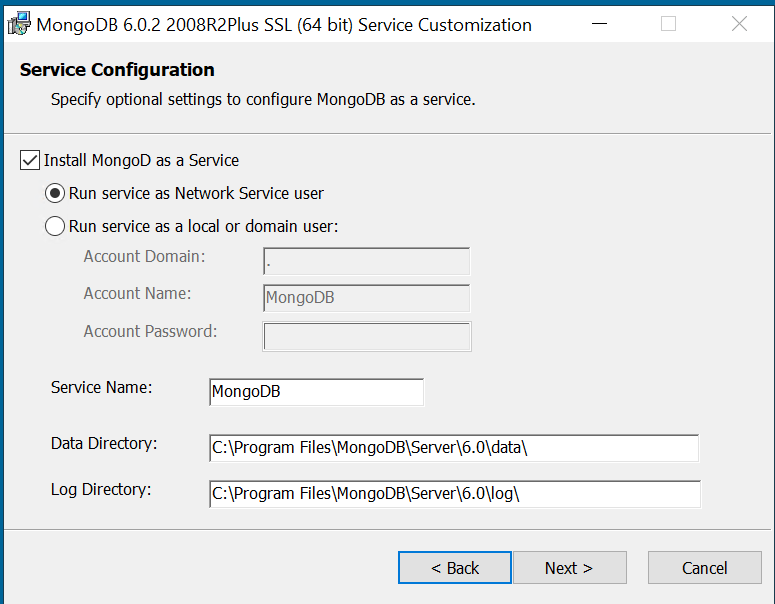


Figure 5 - Installing Mongo DB as a service

If after the installation you want to change the folders the engine is going to use for data and logging, you must proceed in this way:

**1st -** Stop the “MongoDB” service.

**2nd** - Open the folder *C:\Program Files\MongoDB\Server\4.0\bin* with a file explorer instance.

**3rd** - Make a backup of current “mongod.cfg” file.

**4th** - Update the mongod.cfg file by modifying the “storage.dbPath” and “systemLog.path” values as detailed in Figure 6.

**5th** - Grant full access to the local built in account “NETWORK SERVICE” over those folders,(Figure 7).

**6th** - And as last step, start the service.

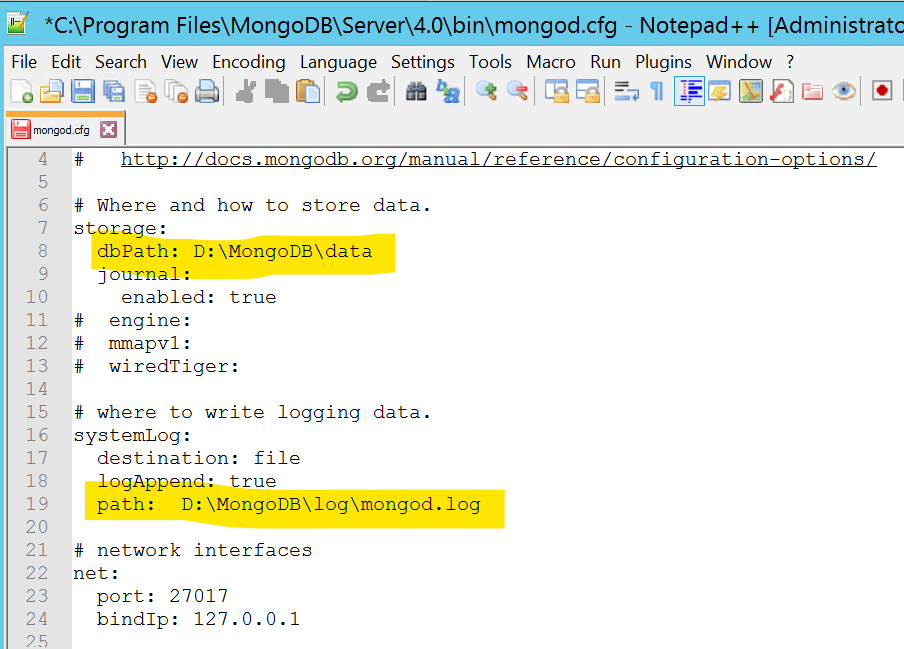


Figure 6 - Modified Mongod.cfg file with alternative folders.

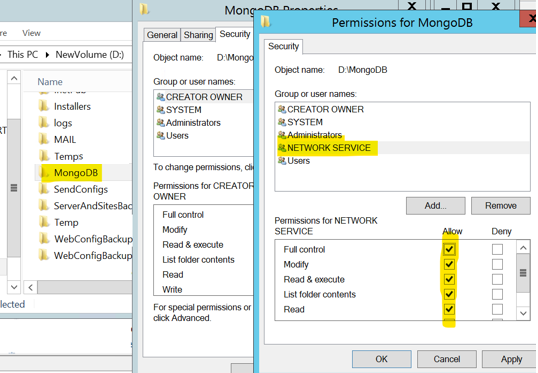


Figure 7 - Granting full access to the service account.

Also, you maybe would like to install the [Mongo DB tools](https://www.mongodb.com/try/download/database-tools). These are helper commands to for example make backups, extract or import data into the db, etc.

To avoid later that the Propel installer failed, there is one more single step required. This is to add Mongo DB runtime folder to the global path. You can do it by running the following script.

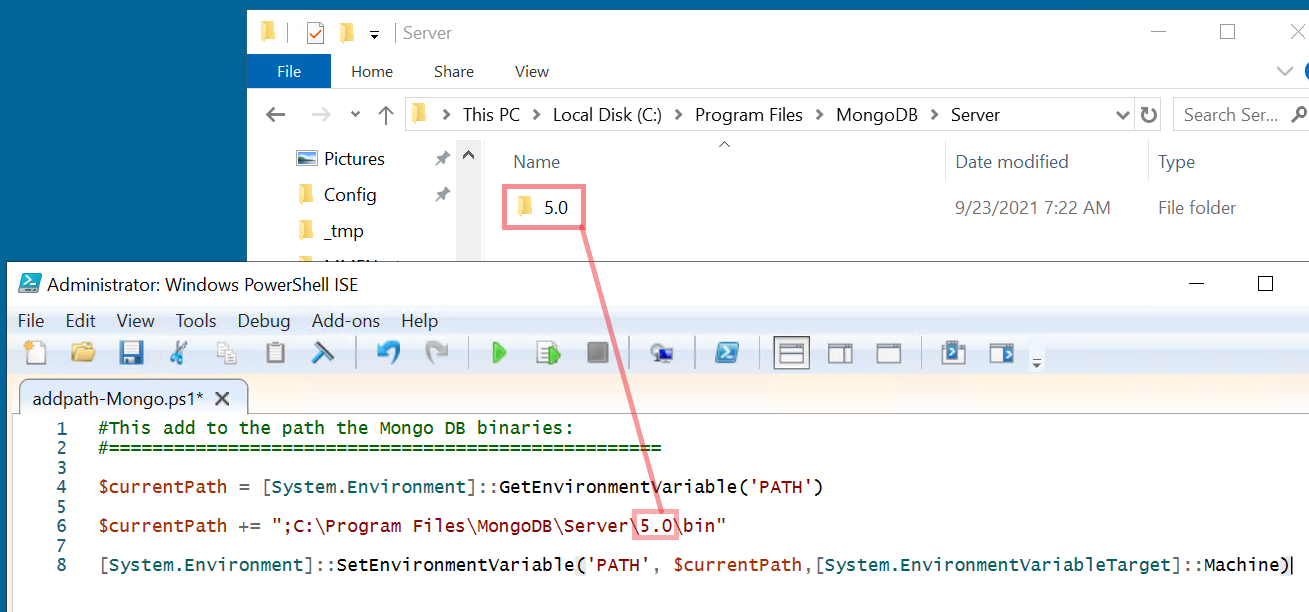


Figure 8 Script to add the Mongo DB path to the PATH system environment variable

Now you can run the command you can see in Figure 9 from any folder to verify the path was set correctly. Be aware **this require to restart the computer to apply the changes**.

Next step is to secure the database. To do that, you can use the following script from Propel v2.0 repository:

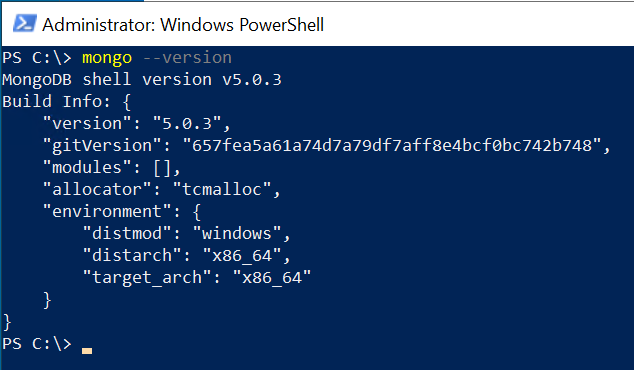


Figure 9 - Getting the Mongo engine version

*.\distrib\cutover\create-admin-user.js*

Yu can copy in the target machine and run it as you can see in Figure 10. This script is going to create a user named “*DBA”* in the Admin database that will have administrator privileges in any database.

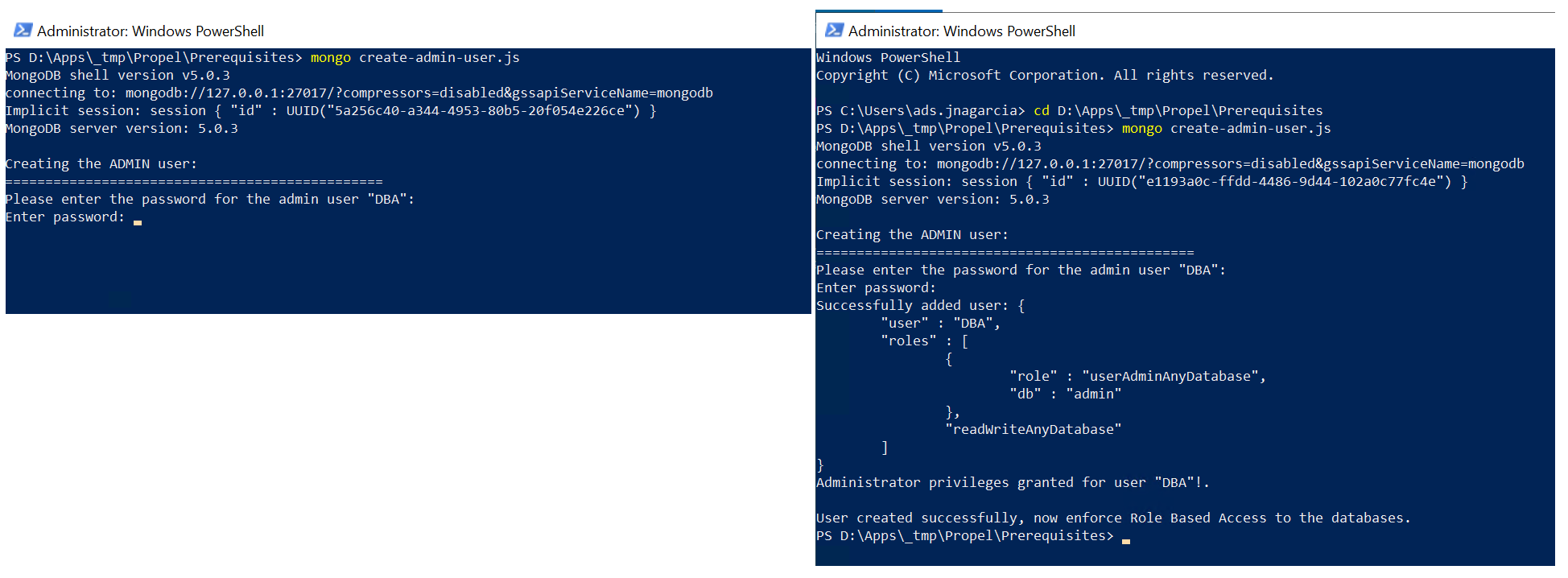


Figure 10 - Creating the new DBA user

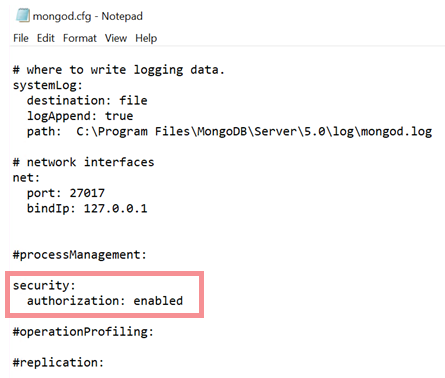


Figure 11 - Adding the Authorization security feature.

After doing that our next steps is enforce user authentication in the MongoDB engine.

So, now we need to stop the MongoDB service, locate the mongod config file located at *C:\Program Files\MongoDB\Server\{Here the server version folder}\bin\mongod.cfg* and edit it to enable the authorization security feature as depicted in Figure 11.

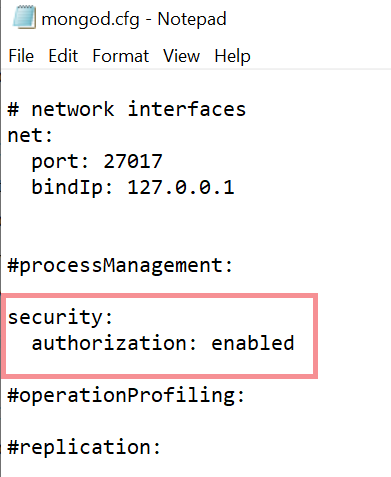


Figure 12 - Enabling auth for all MongoDB databases.

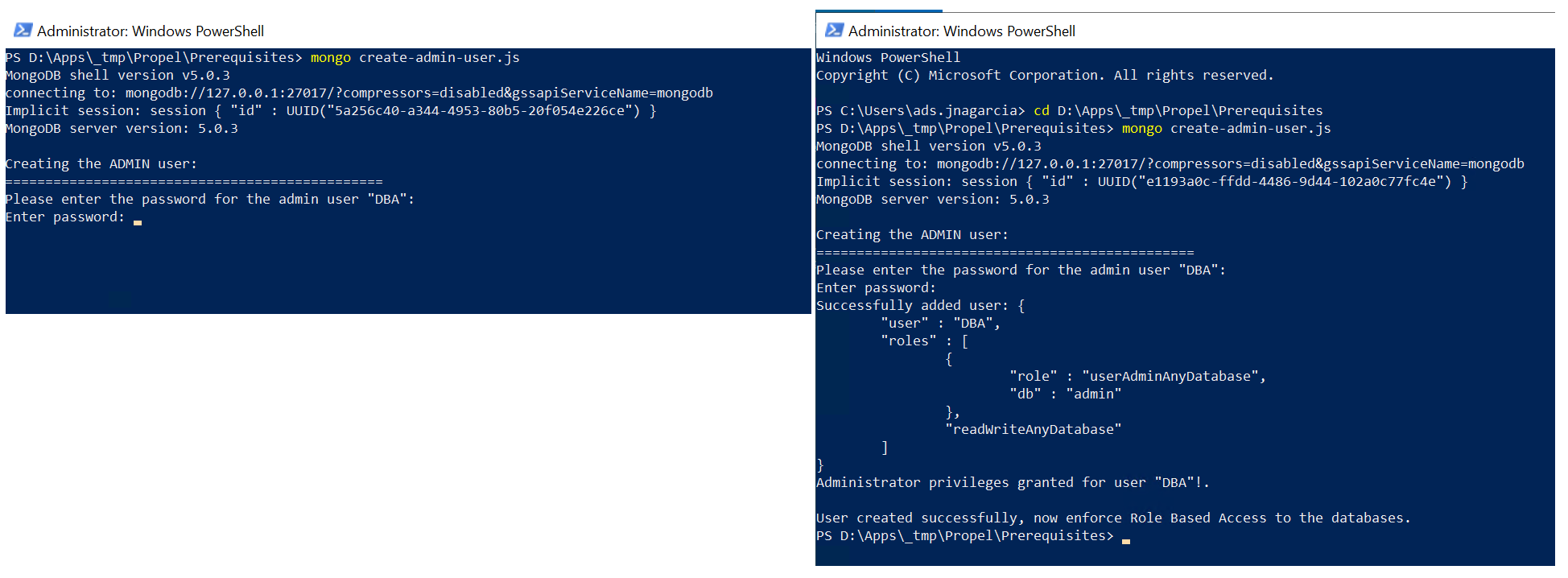


Figure 13 - Creating the new DBA user

After that we can restart the Mongo DB Service.

# Additional tools and steps

Depending on the kind of scripts you are going to execute, maybe you will need to take care of installing and configuring come of the following PowerShell tools and modules.

## ActiveDirectory Module

In order to be able to run LDAP Queries to Microsoft Active Directory, you will need installed the “*ActiveDirectory*” PowerShell module. To do this, you will need to open the Server Manager and add the feature as depicted in Figure 12

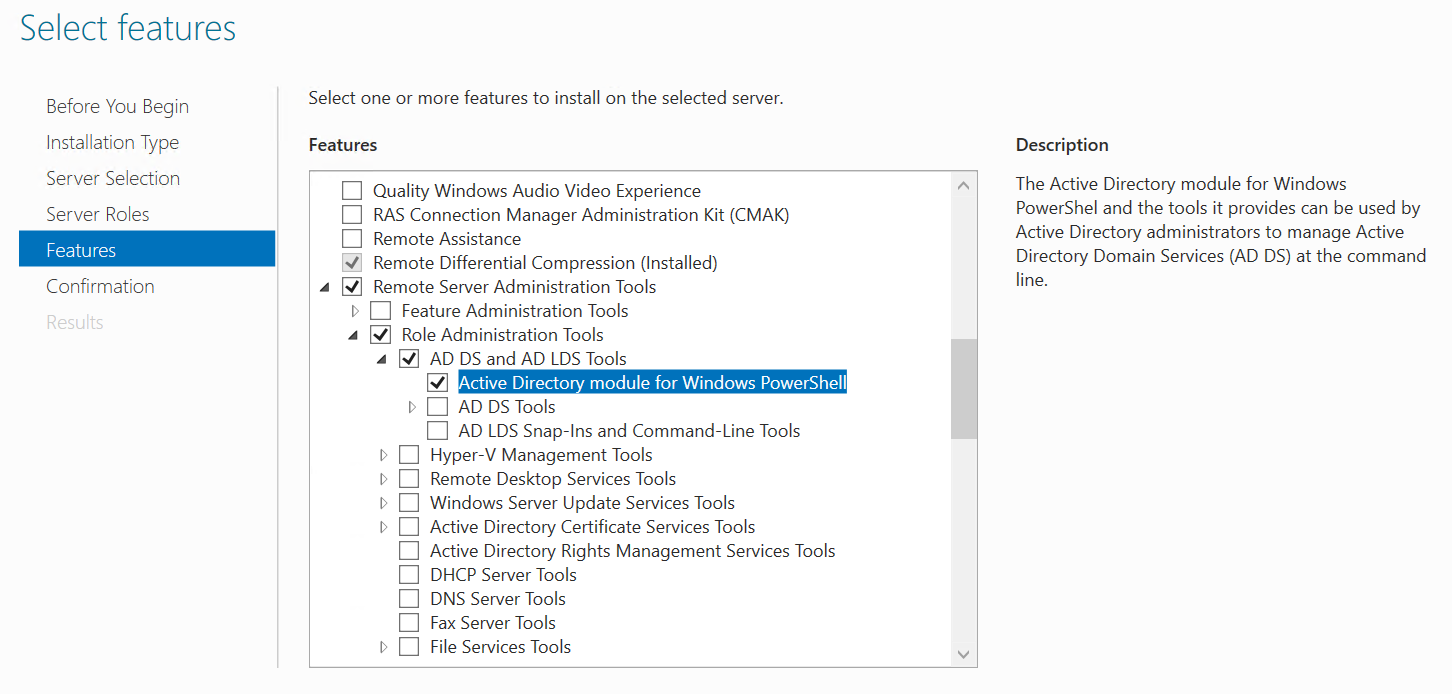


Figure 12 - Installing the PowerShell ActiveDirectory module.

## AWS Tools for Windows Powershell

In the case you have this dependency for your scripts the way to install it is by opening an elevated PowerShell console session and execute this:

Install-Module -Name AWSPowerShell -Scope AllUsers

You can see more details in Figure 13.

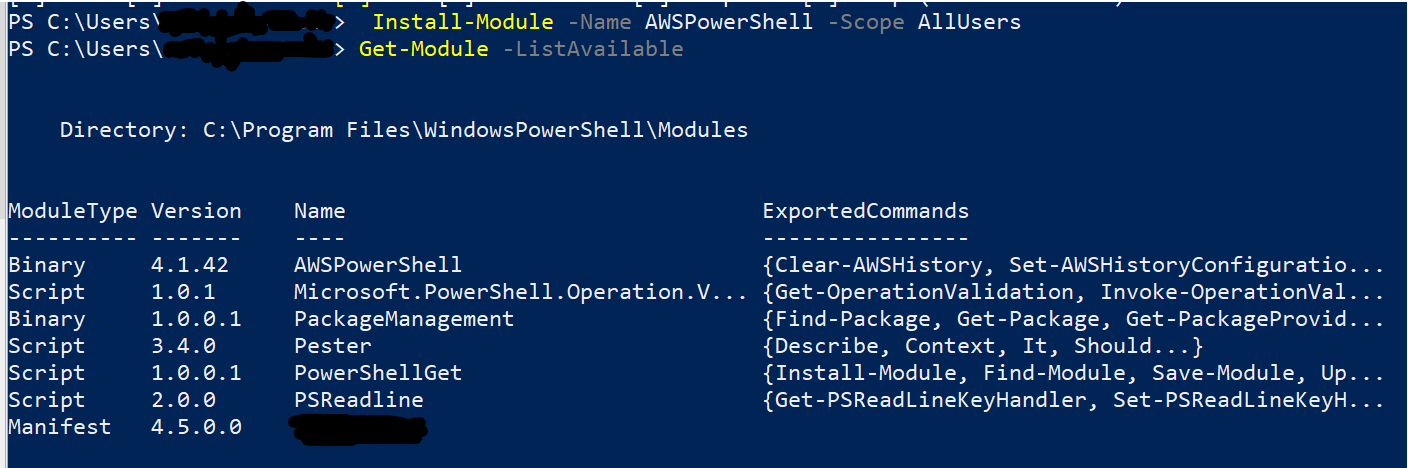


Figure 13 - Installing AWS Tools for PowerShell

For more details about this visit the [AWS Tools for Powershell](https://docs.aws.amazon.com/powershell/latest/userguide/pstools-getting-started.html) site.

## PostgreSQL ODBC driver

You can download the official PostgreSQL ODBC driver for windows from [here](https://www.postgresql.org/ftp/odbc/versions/msi/). Download and unzip the *psqlodbc\_x64.msi* file to install it.

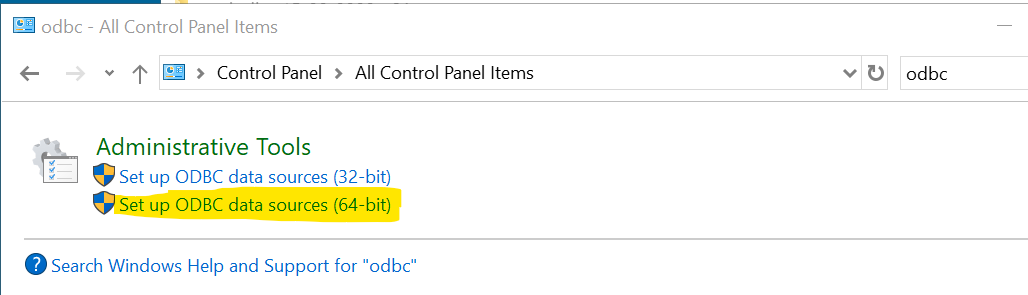


Figure 14 - Setup ODBC drivers in Control panel

Once installed you can see in the ODBC drivers list the Unicode X64 driver.

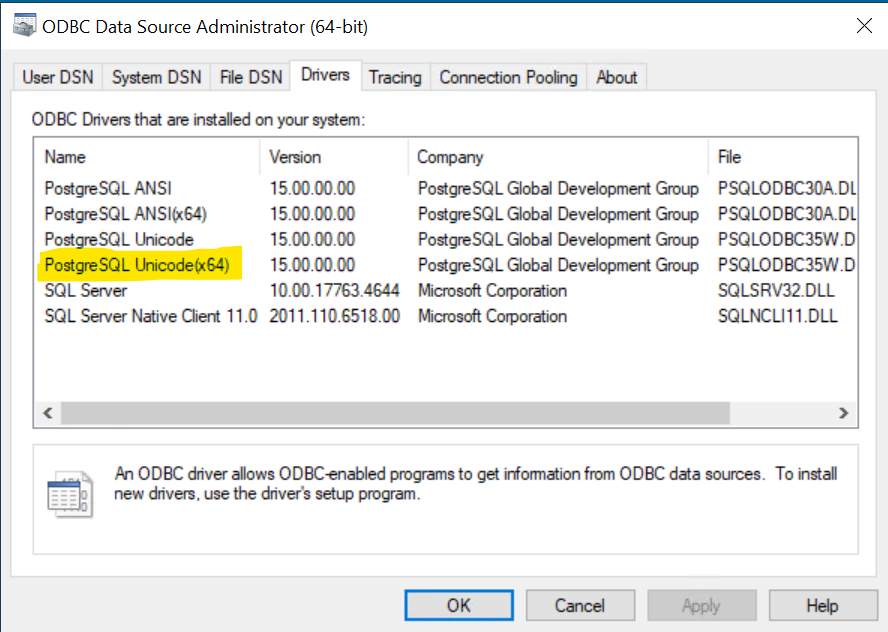


Figure 15 - Installed ODBC drivers.

# Preparing development environment

This section is intended to guide you in the requirements to setup a development environment to be able to clone/download Propel code from our GIT repository and also be able to build, run test, etc.

## Cloning the repo in your local

You will need some sort of IDE to follow the steps in this section. In this document [Visual Studio Code](https://code.visualstudio.com/) is the selected IDE, but you can use whatever you want.

First navigate to our [GIT repository](https://github.com/Elmosoftware/propel-api) and get the clone URL

Graphical user interface, text, application, email

Description automatically generated

Figure 16 - Getting the clone URL from Git repo

Let’s open now the parent folder we want to use for our repository in a PowerShell console and run the following command:

**git clone {Here the url} {Here the folder name to create}**

You can see the run of this command in Figure 17.

Also if is a new Git installation, recall you will need to [setup your user name and email](https://git-scm.com/book/en/v2/Getting-Started-First-Time-Git-Setup) before to be able to commit.

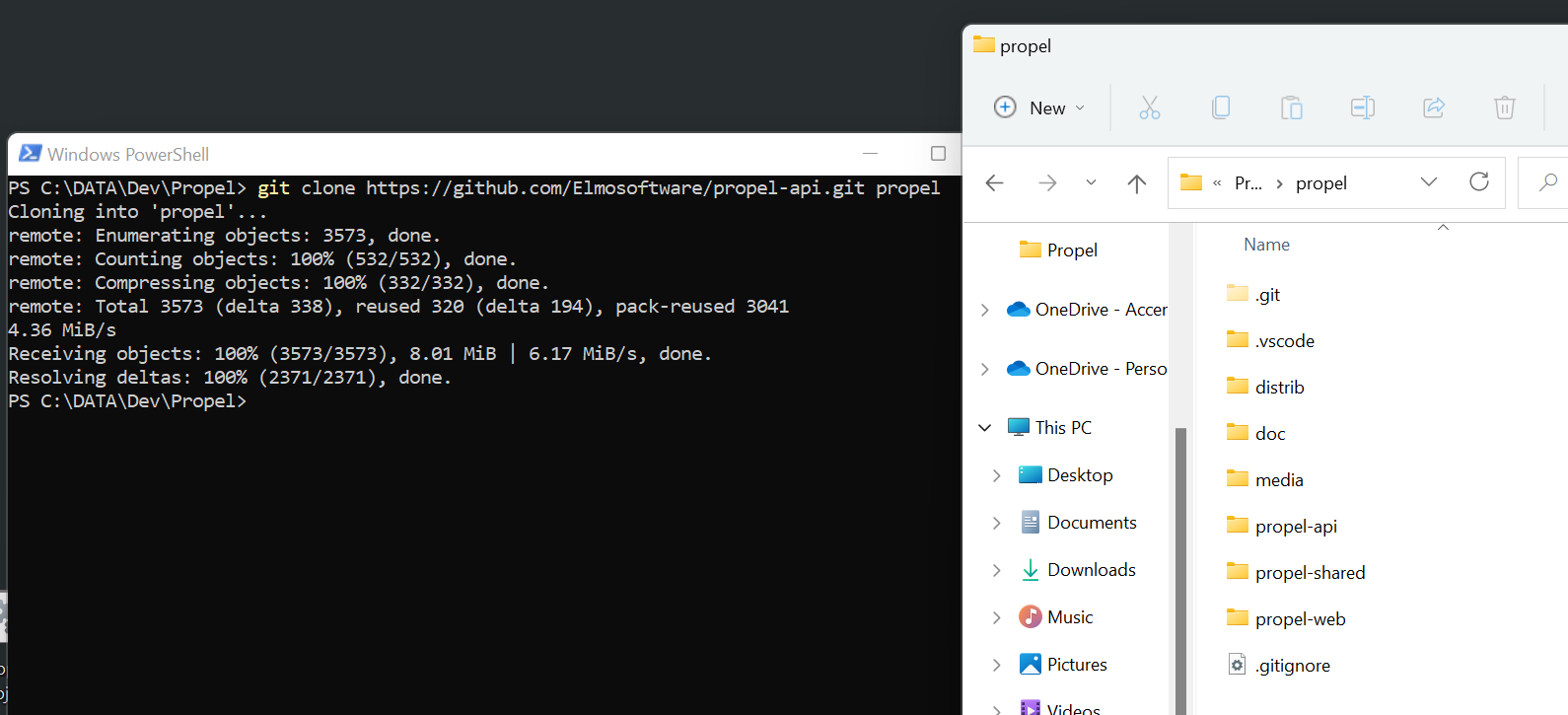


Figure 17 - Cloning the Repository in VS Code

## Propel project scaffolding

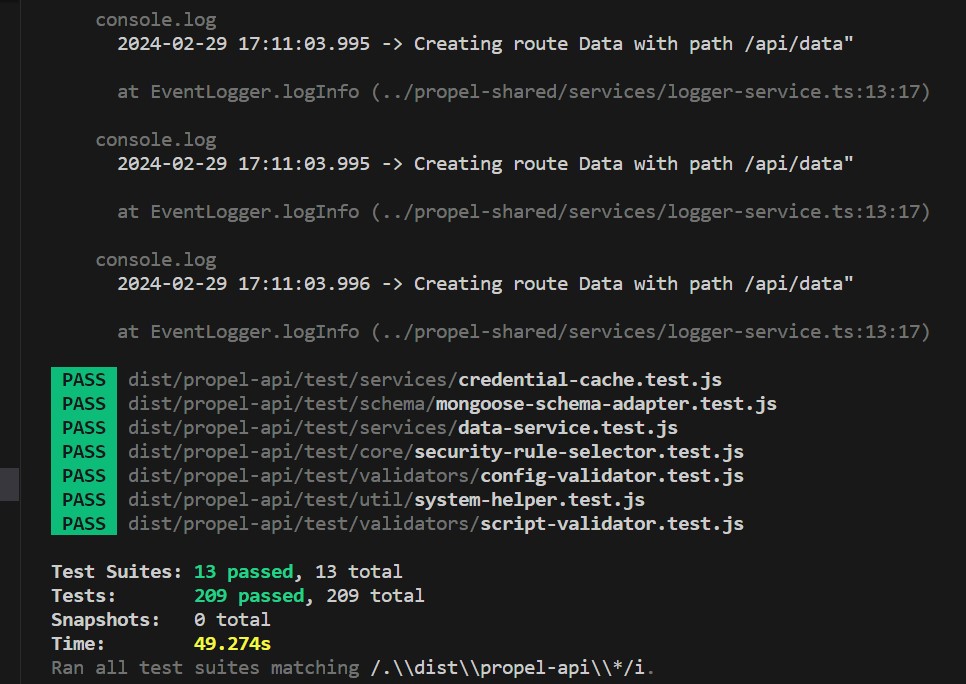
After the cloning process you will see in your destination folder the following structure of the project. Following is some description of each folder:

|  |  |
| --- | --- |
| **Folder** | **Description** |
| **.vscode** | Visual studio code tasks and configurations |
| **distrib** | Distribution folder. The resultant files of a build will be located here. |
| **doc** | Propel document folder |
| **media** | Imagery, icons, etc. |
| **propel-api** | Propel API files. This is the service running in background. |
| **propel-shared** | Shared project containing classes and utility files. |
| **propel-shell** | This contains everything required to package Propel inside an Electron app. |
| **propel-web** | Propel Frontend. |

Figure 18 - Propel Project folder structure

## Running Propel Tests

In you select the menu option “*Run Task*”[[1]](#footnote-1) is VS Code you will be able to select the Propel Test. There is one for API, Shared, Shell and Web sub folders. They will run separately inside the terminal window, except for the Web test that run inside a Web browser.



A computer screen with lights

Description automatically generated

Figure 19 - Propel API and Propel web tests execution.

## Running Propel in your Dev environment

After cloning the project you are able to compile and run propel in your dev environment. In order to do this you must perform 2 separate steps

1. **Run the API**: Again, clicking in the menu option “*Run Task*” of VS Code you can find a task named “Run API” this task is going to transpile the code to Typescript and start the API.
2. **Run the Frontend local server**: Now, if you click in the task named “Run Web”

As soon both task are completed you will be able to open a browser and navigate to <http://localhost:8080> and start Propel in a web browser.

A screenshot of a computer program

Description automatically generated

Figure 20 - Options to run the API and the web frontend from the tasks menu.

**Just a note**: Any change you do in a file inside the *\propel-web* folder is going to cause immediate compilation of the app and you will see the changes in the browser.

But this is not the case for the API. If you make any changes to the *\propel-api* folder you will need to kill the service process and run again the task to see reflected those changes.

## Run Propel frontend inside an Electron app in your DEV environment

There are some specific testing situations where you can desire to run Propel as a desktop app instead of running from a Browser. Reasons for this could be to test specific accessibility/UI features or test the simplified login feature that is only present when running propel inside an Electron app[[2]](#footnote-2).

To do this, you will need to follow the same procedure already described to run the API, but for the Frontend, we need to execute the following steps:

**1st** – Execute the task “*Build PRODUCTION (Electron Shell Only)*”: This task is going to build for production the Propel frontend, and then build the electron app that is going to package Propel as a desktop app.

**2nd** – In a terminal window change your working dir to *\propel-shell* and run in the command line:

**npm run startElectron**

This is going to start Propel as an Electron desktop app.

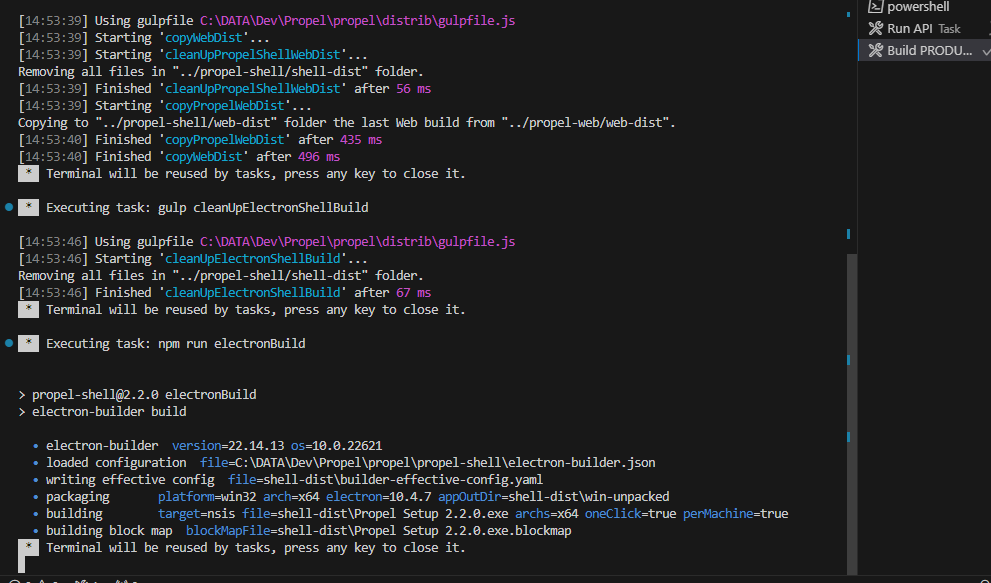


Figure 21 - Task "Build PRODUCTION (Electron Shell Only)" finished run.

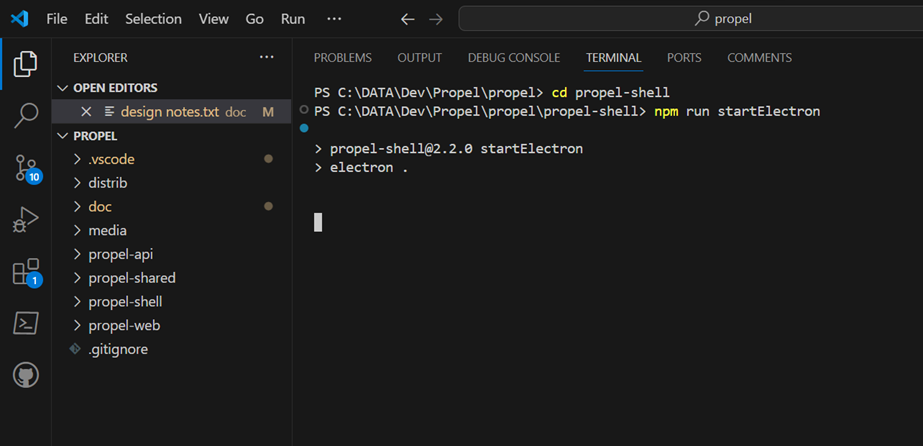


Figure 22- Starting Propel in an unpackaged Electron app.

A screenshot of a computer

Description automatically generated

Figure 23 - Propel running as a desktop app.

## Deploying Propel

In VS Code click on the “Run Task” menu option of the “Terminal” menu and select the “Build Production” task. After it runs you will have in *.\Distrib\dist* folder the new build. Next step is to compress the folder content and copy them to the destination server in a temporal location. **If it’s a first-time deployment, ensure the destination folder *C:\Propel* exists**, otherwise please create it. That’s the folder where Propel is being installed.

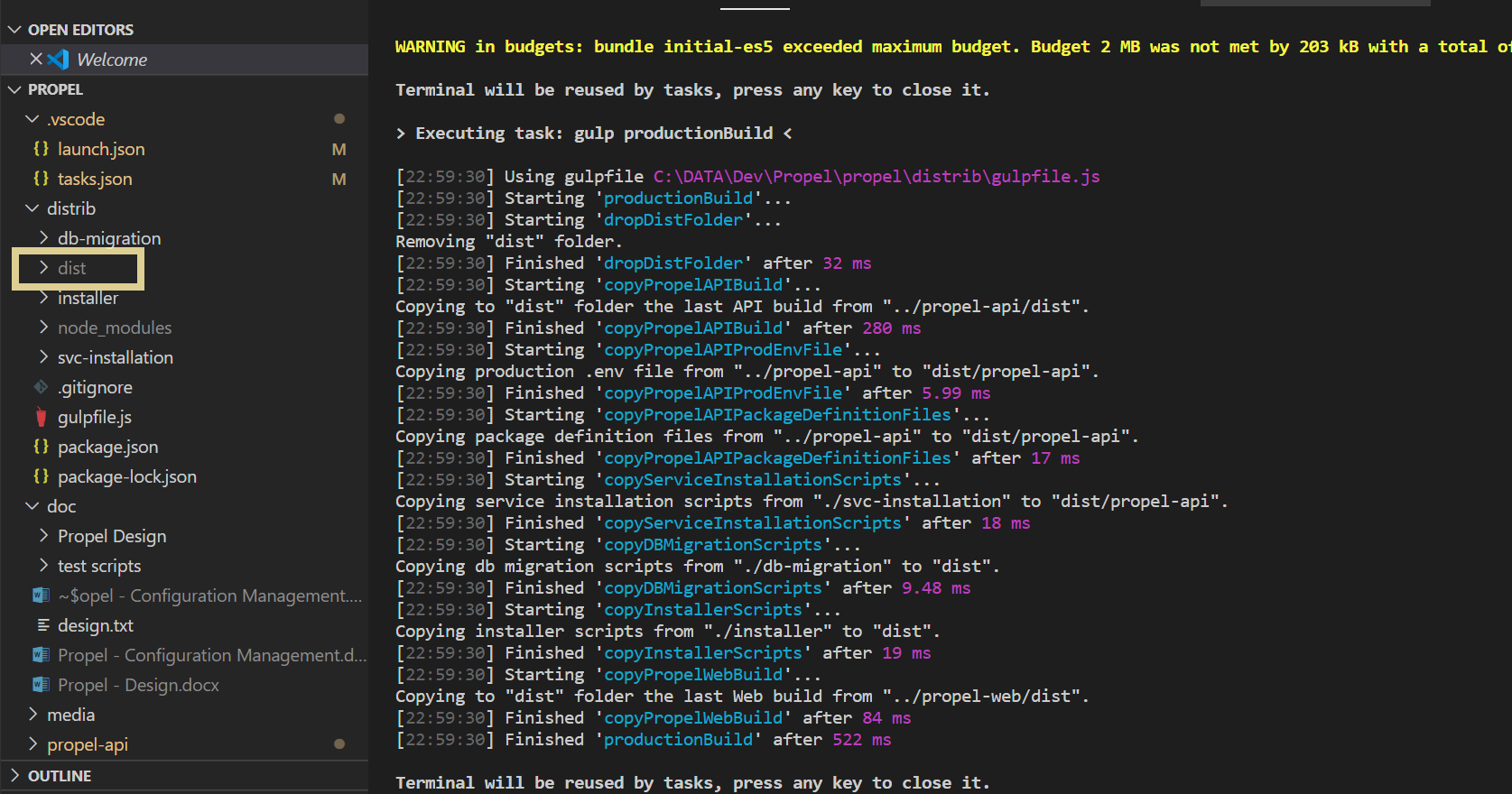


Figure 24 -Dist folder

In the temporal installer path, please locate the file “install.ps1” and run it with PowerShell.

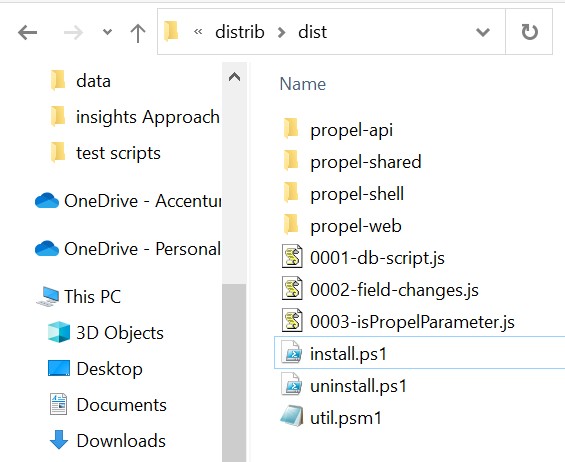


Figure 25 - Installer file

The installer will take care of:

Stop the Propel service if it is running, install the new version and restart the service.

It will also apply any required database migration scripts. And will allows you to configure a specific account to use for impersonation during remote script execution, (otherwise remote scripts will run with the credentials set to the service).

## Tagging the Release

To tag our release please proceed in this way:

• Create the tag with the following command:

**git tag -a v2.2.0 -m "v2.2.0"**

• You can see the list of created tags with the command:

**git tag**

• Next step is to push it to the remote:

**git push origin v2.2.0**

1. [Tasks in VS Code](https://code.visualstudio.com/docs/editor/tasks). [↑](#footnote-ref-1)
2. [Electron.JS](https://www.electronjs.org/) [↑](#footnote-ref-2)