Egret Installation Guide

## Software Recommendations

I highly suggest that Egret be installed on a computer running Windows 10 or Windows Server with at least 100gb of hard drive space and 8gb of memory, but more is wise. Although the software requirements are not burdensome, and Egret can easily be installed on less-than-cutting-edge systems, avoiding a complete reinstall has value, so it is worth using quality hardware from the beginning. A hard drive failure in the middle of the day, for example, could grind application usage to a halt and prove a heavy burden to restore.

Be sure that the computer is placed in a secure location and cannot easily be turned off by accident. A modest backup power source is recommended. Also, if you will be using Windows 10, as opposed to Windows Server, you may wish to research various “hardening” methods for reducing any potential security threats. This should be a computer dedicated to the application and database – this should not be a YouTube viewing station :-) . Do not install miscellaneous programs, and consider modifying the firewall to block all unnecessary ports.

Egret has not been tested to modern security standards, and will not currently be using SSL over a local intranet. This can be configured, but it would require processes to be followed on each accessing machine to allow for secure communication. Nonetheless, modern security features have been implemented such as hashed passwords and authorization/authentication features. Some form of virus protection is recommended, but the default Windows Defender may be enough.

Make sure that your router is not running in “promiscuous mode” or communications between browsers and the server where Egret is installed may be susceptible to interception and modification.

## PostgreSQL

PostgreSQL is the database that Egret uses to store its data.

Install

1. Go to [https://www.postgresql.org](https://www.postgresql.org/)
2. The website may change, but go to Downloads, Installers for Windows.
3. In this document, I will be using version 12.0, but it will likely be safe to install newer versions. Install the 64-bit version installer, which will appear as Windows x86-64.
4. Run the installer. The installer may also install some C++ binaries. This is fine.
5. Use the default installation directory. It should be something like C:\Program Files\PostgreSQL\12
6. The password for test systems is typically postgres, but you may want to change this to make the system more secure. If you do change it, make sure you properly store the new password! There will be no way to get into the database later without this.
7. Most other defaults for the installation should be fine.

# PGAdmin

PGAdmin is a popular user interface program for interacting with the PostgreSQL database. It runs in a web browser, but please note it is not accessed over the internet. PostgreSQL already comes with a command line utility for interacting with your databases, but you will probably not want to deal with that unless you are an experienced user, so the installation of PGAdmin is recommended.

Install

1. Go to [https://pgadmin.org](https://pgadmin.org/)
2. Go to Downloads
3. Select the download for Windows
4. The latest version should be fine, but for this document I will be installing pgAdmin 4 v4.13
5. Look for and select the x86.exe, and run the installer when prompted
6. The default installation options should be fine
7. You will need to set a master password for the PGAdmin. This can be the same as the password for the PostgreSQL database.

You will need to make the icon for PGAdmin easily accessible:

1. Hit the search key, or click the start button
2. Search for PGAdmin 4
3. Right click the search result for PGAdmin 4 and select “Open File Location”
4. Right click the program in the new window and select “Send To… Desktop”
5. Run this file to open PGAdmin. Enter the password when prompted
6. By drilling down into the Servers on the left of the screen, you will find a database called “postgres”. This is the default database. Later, when Egret is installed, you will see a new database here called “egret”

# Visual Studio Community Edition

Visual Studio is an Integrated Development Environment (IDE). Normally, I would avoid having you install this yourself, but it is critical for setting the database up initially and will also be used for upgrades and potential rollbacks. Please note that Visual Studio Community Edition is free but requires a user account to register after a 30-day “trial” period. Sorry, there’s no good way around this.

Install

1. The website changes all the time. Just search Google for Visual Studio 2019, and select the official Microsoft site that appears first
2. Mouse over “Download Visual Studio” and select “Community 2019”. Don’t select the other versions unless you want to pay for them
3. Run the installer.
4. You will be asked which features you would like to install with Visual Studio. If the options exist, select “ASP.NET and Web Development”, and “.NET Core Cross-Platform Development”. Note that “.NET Cross-Platform Development” may be toward the bottom of the screen when you scroll down. For the optional tools, feel free to stick with whatever is auto-selected.
5. Note that this is about 9gb of download. The full installation could take awhile!
6. Restart the computer when prompted
7. Search for Visual Studio
8. In the search results screen, right click Visual Studio 2019 and select Open File Location
9. Right click the Visual Studio 2019 shortcut and select Send To > Desktop

Optional:

If you would like to perform development on a local repository, you may need the Web Compiler extension for Visual Studio. This will be used for compiling SASS files into CSS. Go to Extensions > Manage Extension, and search Online for “Web Compiler” by Mads Kristensen. Install this and restart Visual Studio. If you do not plan to perform manual development, this may be skipped. Be warned, manual development may be overridden by source code pulls from the Egret repository, so only developers should install this extension.

# Internet Information Services (IIS)

IIS is a web server that comes pre-installed on all versions of Windows, it just needs to be activated to use. IIS is how the application will serve up the web pages to the computers on your network.

Install

1. Hit Start or the Windows key and search for Programs and Features
2. On the Programs and Features window, select “Turn Windows features on or off” on the left side of the window
3. Scroll down to Internet Information Services and select the box. Don’t worry about additional options
4. Wait for the installation to complete, then close the window
5. Search for IIS
6. In the search screen, right click IIS and select Open File Location
7. Right click the shortcut in the window and select Send To > Desktop
8. We will configure IIS later

# Git

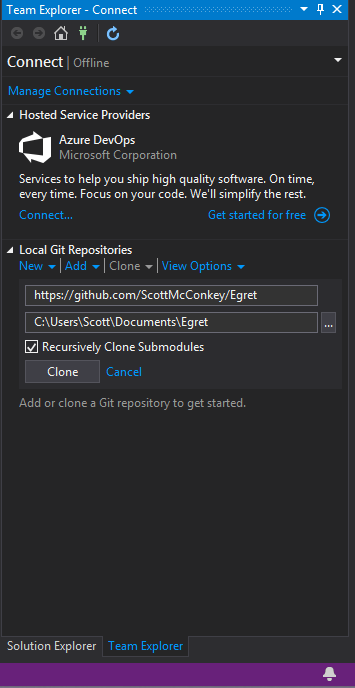
Git is a Version Control client that will be used for checking out the code files and any new update files. Please reference the Versioning\_Guide.docx file for more information.

Install

1. Go to https://git-scm.com/
2. Go to Downloads
3. The site should detect you are using Windows. Select the Latest source Release, often located in the teal image of a computer screen. The version should not matter
4. Run the installer
5. Use any default text editor. I would recommend Notepad++, but you will need to install this separately (it may be found online and downloaded for free)
6. The other defaults should be fine

# Putting it All Together

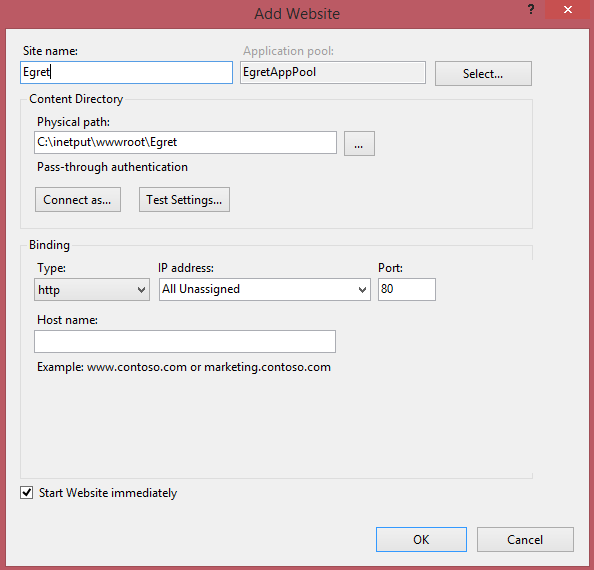
We now have all the base programs required to get Egret running! It may seem like a lot, but this is actually fairly basic for most websites. The beauty of being a website user is that you never have to see these details behind the scenes, but the story is far more involved on the other side of things. I’ve tried to keep this as simple as possible, but several tools are still necessary.

1. In the Solution Explorer window, in the appsettings.json file, find the “Connection Strings” section. Note the Password is set to the default postgres. If you used a different password to install the database, remove the default password and type your chosen password there.
2. Open Visual Studio and go the Team Explorer. If you do not see it, go to View > Team Explorer
3. Under the Local Git Repositories section from the Connect screen, select the Clone dropdown and enter https://github.com/ScottMcConkey/Egret in the space that says, “Enter the URL of a Git repo….”
4. For the default path, change it to something like C:\Users\<default user>\Documents\Egret and click Clone. The cloning process may take a few minutes.
5. Check the file location in Windows Explorer. It may be worth dragging this folder into the Quick Access section of Windows Explorer.
6. With the Egret solution opened, opened the Package Manager Console. If you do not see it, go to View > Other windows > Package Manager Console.
7. In the package manager Console, type dotnet restore. This downloads all the packages that that Egret uses internally. This may also take a few minutes.
8. Go to Build > Build Solution. If this fails, try running the dotnet restore again.
9. Once the project has successfully built, go to the Package Manager Console, type update-database and hit enter.
10. If you go back into PGAdmin, you should notice the new database “Egret”.
11. In Visual Studio, go to Debug > Start Without Debugging
12. Login with the default bob@example.com user with password secret123
13. Go to Admin > Users, and create your user. Be sure to store your password securely for future use.
14. One access group – Administrator – has been automatically included in the system for you. You may need to configure additional Access Groups from Admin > Access Groups. On your user, go to Configure and check the checkbox for Administrator and Save.
15. At this point, logout and log back in with your new user. Ensure that you can access the Admin pages. At this point, you will want to inactivate the Bob default user by editing him and un-checking the checkbox for Active. Note that if you do this before you have your own user saved and added to the Administrator access group, you will not be able to get into the Admin portions of the system without running a special database command. I can write this if you need it, but it’s better to simply make sure you have fully completed this step first.

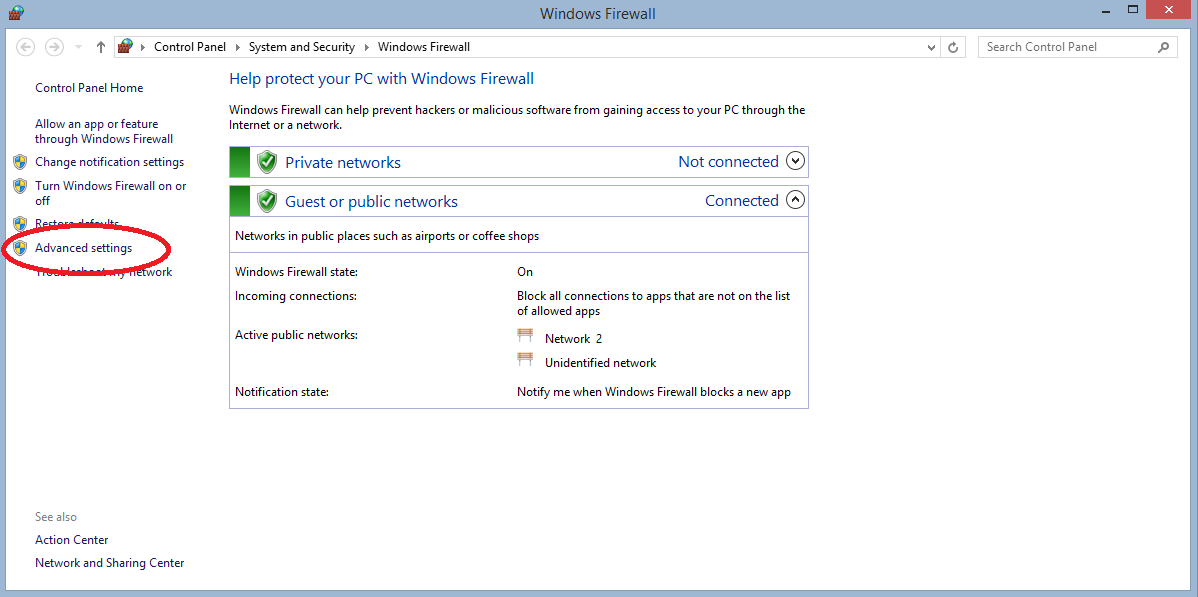
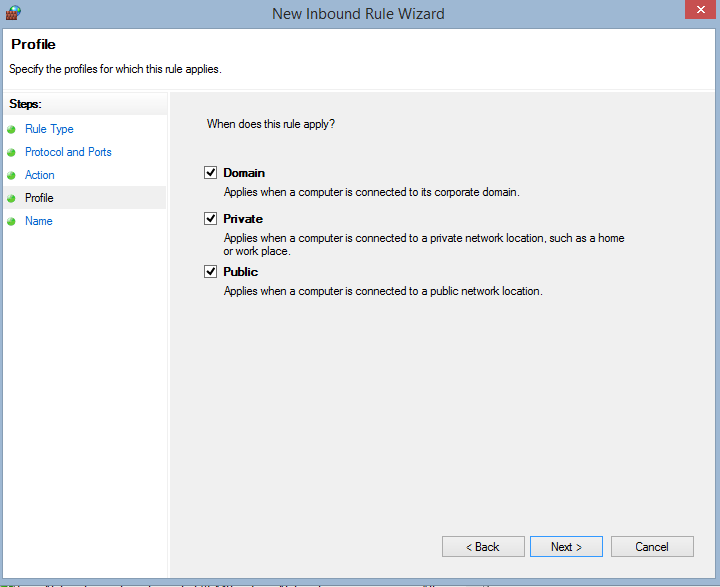
Congratulations! You should be able to run Egret from Visual Studio by going to Debug > Start Without Debugging. Please note that Internet Explorer 11 will NOT work with Egret, but any modern browser such as Firefox, Chrome, or Edge from the past few years should. If you encounter any issues with this, please let me know.

This is enough to begin testing Egret out locally, but this is not enough to serve Egret pages up over the network; this requires configuring IIS to do so. This will probably be the most involved aspect of setting Egret up.

# Configuring IIS

1. In Visual Studio, go to Build > Publish. The Publish options should already be filled in. I have experienced issue with with concerning the security on the target file path. I have not been able to reproduce this, but if you encounter issues, let me know.
2. For the publish target, select “IIS, FTP, etc” and go to Next
3. Go to the web, and search for “Download .NET Core 3.1” and find the appropriate Microsoft page. On the downloads page, you may see a table with “OS”, “Installers”, and “Binaries” at the top. For the Windows OS, pick the Hosting Bundle Installer for 3.1.x (most recent) and install it.
4. Go into IIS
5. Go into App Pool and create a new App Pool called “EgretAppPool”. The default options should be fine.
6. Right click on the Default Site and delete it.
7. Right click on Sites and create a new Website. Set the site name to Egret, change the app pool to be EgretAppPool, and specify a physical path of C:\inetput\wwwroot\Egret
8. Click OK
9. Go to a new browser tab and type localhost in the URL and hit enter
10. You should now be directed to the Egret login page.
11. When you configure your network router, you will want to use a network address of “Egret” and map this to the ip address of the computer that is hosting Egret. This should probably be a static IP address, but you are free to network to your heart’s content if you know what you’re doing. The requests to this computer will be calling this page, and that is how users can login from other computers on the network. How to do this will depend upon the router you are using.

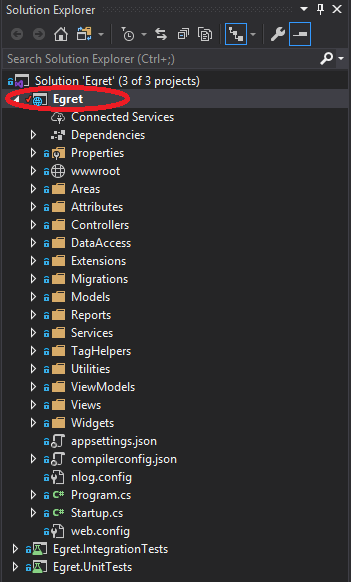
Great, now we can access Egret from IIS. However, we still can’t access Egret from outside of our server. To do this we need to add a Firewall rule that allows IIS to serve receive network requests.

1. Hit start or the Windows key and search for Windows Firewall. Click on the “Windows Firewall” option that should appear.
2. Go to “Advanced Settings” on the left hand side of the screen
3. Right click on “Inbound Rules” and select “New Rule”.
4. In the “New Inbound Rule Wizard” that is brought up, select a type of “Port” and click Next
5. Leave the TCP option selected, and “Specific Local Ports” enter the number 80 and click Next
6. Leave “Allow the connection” selected and click Next
7. Feel free to leave all three options of “Domain”, “Private”, and “Public” selected. I’m hesitant to state these should be selected, but Windows has a weird way of organizing so-called Public and Private networks, and I wasn’t able to access the site without Public selected. You may need to audit your network setup to differentiate the security between those computers that can access the Egret server and those that can simply connect to wifi. Egret requires a login to access data, however, so this may not be a concern. Click Next
8. Give the rule a name of Egret Website Access, or really anything you want that is memorable.

(HTTP connections are made on port 80 by default, so by opening port 80, we are allowing outside computers to send requests to IIS on the server, which then sends pages from the website back)

Optional: If you’d like to get fancy, you can enable ping on the server by adding a New Rule in the Inbound Rules for Custom > Action and calling it something like “Egret Ping”. Pinging the website from the command line on on outside computer is a fast way to test whether the server is accessible or not. Also, configuring your server for Remote Connection is a great way to access the server without logging into it physically, but that is beyond the scope of this guide.

**The last step!:** If you do not wish to receive nasty code puke in the event Egret experiences an error, you will want to modify the Solution.

1. In Visual Studio, right click on the Egret project in the Solution Explorer and select Properties.
2. In Properties, select the Debug vertical tab.
3. On the Debug tab, look at Environment Variables. You will notice a value that says ASPNETCORE\_ENVIRONMENT with a value of Development. Change this from Development to Production and save.
4. Getting this setting to stay the same across updates could be difficult, I haven’t figure out yet if Visual Studio does that automatically or if every updates replaces it with Development. At any rate, if it is on Development and you get a nasty error, you can always send that error to me and I can look into it, but in general, this can be acquired from logs, and it’s better that users see a nice clean Error 404 or 500 page instead.