
Jp Project Documentation

Release 1.0.0

Bruno Brito

Aug 01, 2021

1	Admin UI	3
2	Login page	5
3	Consent page	7
4	Profile	9
5	Demo online	11
5.1	Free	11



The main goal of JP Project is to be a Management Ecosystem for IdentityServer4 and ASP.NET Identity.

Helping Startup's and Organization to Speed Up Microservices Environment. Providing tools for an OAuth2 Server and User Management.

Built with IdentityServer4. An OpenID Connect and OAuth 2.0 framework for ASP.NET Core.

Features of SSO:

- Register users
- Recover password flow
- MFA
- Federation Gateway (Login by Google, Facebook.. etc)
- Argon2 password hashing
- CSP Headers
- Event monitoring (For compliance scenarios)

Admin UI is an administrative panel where it's possible to manage the OAuth2 Server, Users and Roles.

From OAuth2 panel it's possible to manage:

- *Clients*
- *Identity Resources*
- *Api Resources*
- *Persisted Grants*

From Identity panel it's possible to manage *Users* and *Roles*

It's open source and free. From community to community.

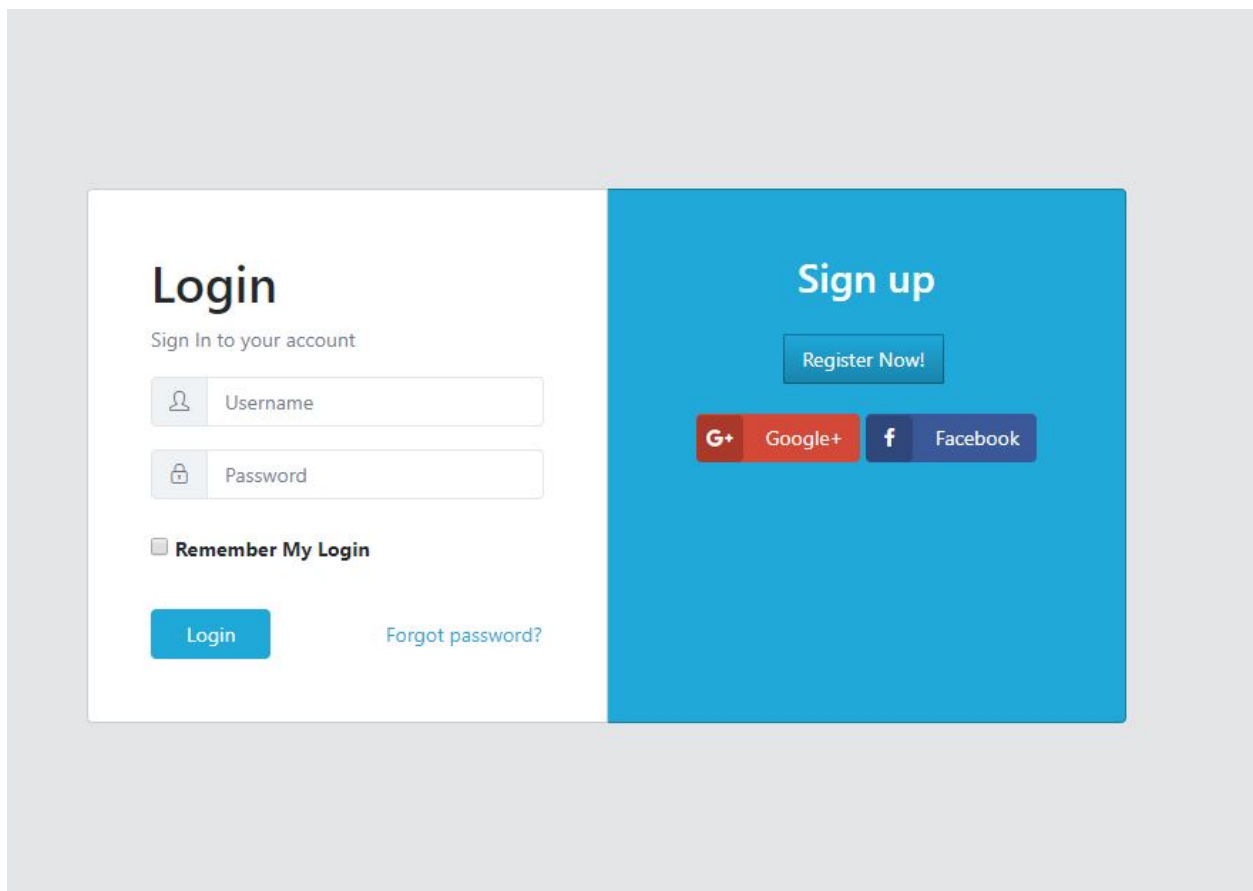
Screenshots

CHAPTER 1

Admin UI

CHAPTER 2




Login page



CHAPTER 3

Consent page

Authorize Application

   **identity**

User Management UI is requesting your permission

Uncheck the permissions you do not wish to grant!

Personal information

<input checked="" type="checkbox"/>	Your user identifier (required)
<input checked="" type="checkbox"/>	User profile ⓘ Your user profile information (first name, last name, etc.)
<input checked="" type="checkbox"/>	Your email address ⓘ
<input checked="" type="checkbox"/>	picture

CHAPTER 4

Profile

Now we are online! [See it in action](#)

Below an intro video!

Do you love it? give us a Star!

Wanna contribute? Feel free to do that!

But remember to check the [Contributing Section](#)

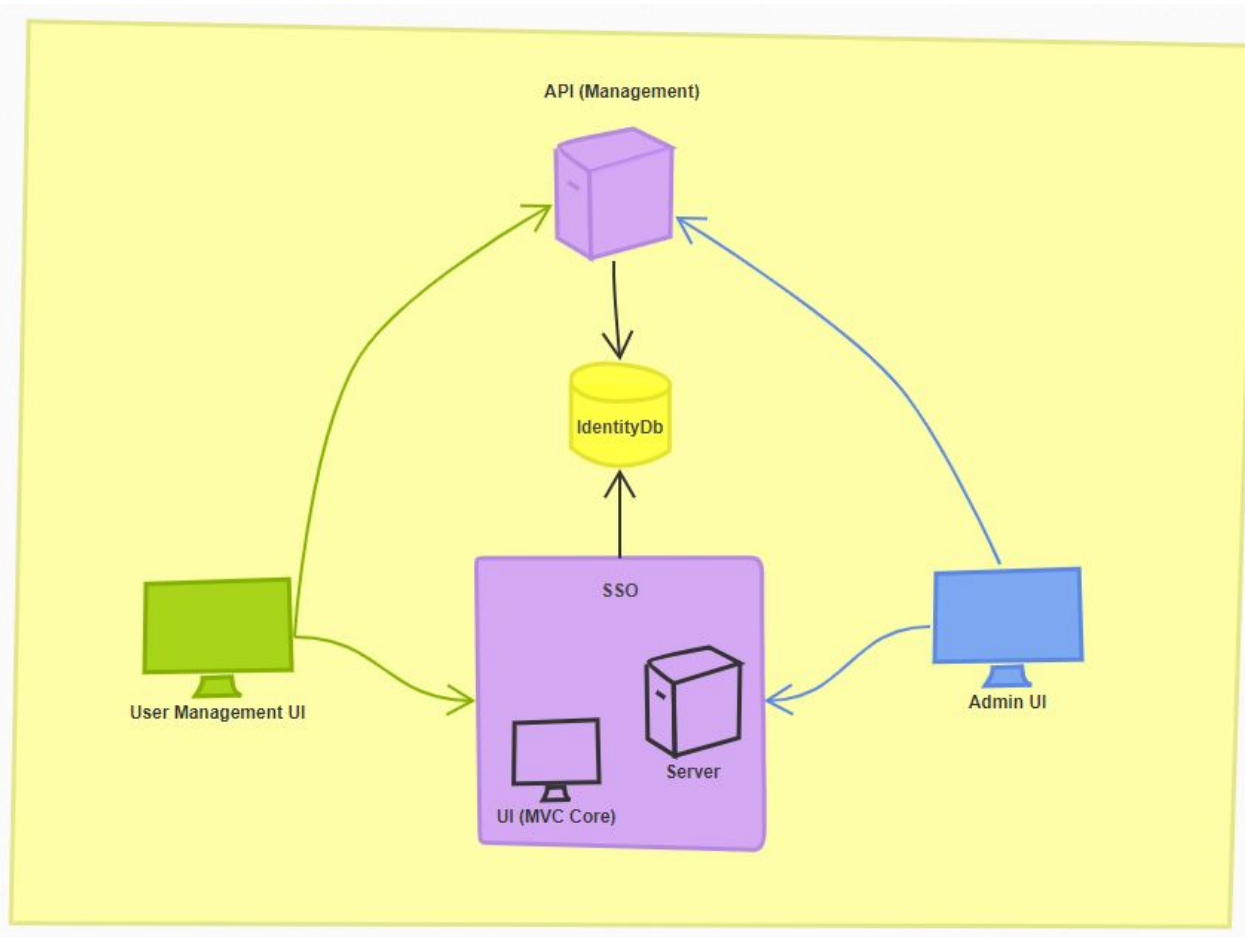
Take a special place in our heart!

5.1 Free

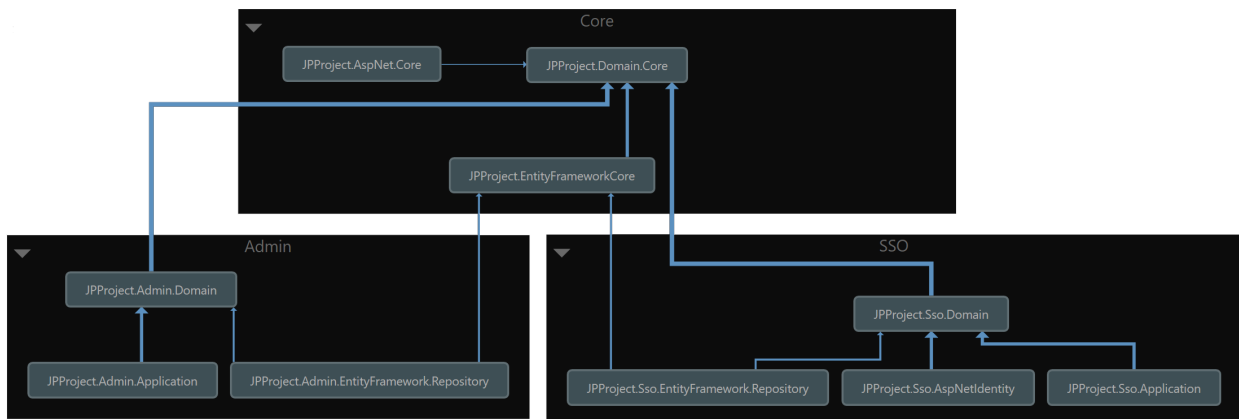
If you need help building or running your Jp Project platform There are several ways we can help you out.

5.1.1 Overview

An overview about current development



Dependency Diagram



SSO

It's responsible for Authenticate users. Check it's credentials and emit Tokens for Applications. Authentication is needed when an application needs to know the identity of the current user. Typically these applications manage data on behalf of that user and need to make sure that this user can only access the data for which he is allowed.

Click [here](#) to check more details.

User Management UI

A SPA application responsible for create and manager users. Send reset links. E-mail validation. Profile validation, and so on.

Management API

API to serve UI's.

Admin UI

It's the Admin user interface to manage identity Server 4 and ASP.NET Identity. It also provide another SSO features, like configure Blob Storage, E-mail settings. Event viewer.

5.1.2 Architecture

This project was developed aiming the best practices.

Equinox Project

The ASP.NET Core Architecture was based on [Equinox Project](#).

Angular 8

Admin UI and User management was built with Angular 8.

Technologies

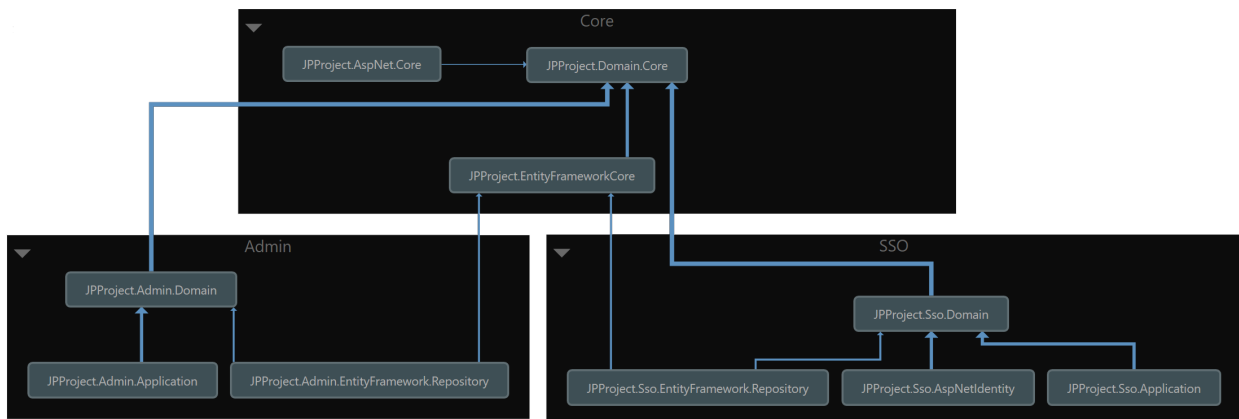
Check below how it was developed.

Written in ASP.NET Core and Angular 8.

- Angular 8
- Rich UI interface
- ASP.NET Core 3.1
- ASP.NET MVC Core
- ASP.NET WebApi Core
- ASP.NET Identity Core
- Argon2 Password Hashing
- MySql Ready
- Sql Ready
- Postgree Ready
- SQLite Ready
- Entity Framework Core
- .NET Core Native DI

- AutoMapper
- FluentValidator
- MediatR
- Swagger UI
- High customizable
- Translation for 7 different languages

Architecture



- Architecture with responsibility separation concerns, SOLID and Clean Code
- Domain Driven Design (Layers and Domain Model Pattern)
- Domain Events
- Domain Notification
- CQRS (Immediate Consistency)
- Event Sourcing
- Unit of Work
- Repository and Generic Repository

5.1.3 Data Protection Keys

SSO has a default implementation for ASP.NET Core Data Protection Keys at database level. Using dataprotection at database level we leverage configuration to run through a Web Farm:

- **Reliability/availability** – When one or more nodes fail, the load balancer can route requests to other functioning nodes to continue processing requests.
- **Capacity/performance** – Multiple nodes can process more requests than a single server. The load balancer balances the workload by distributing requests to the nodes.
- **Scalability** – When more or less capacity is required, the number of active nodes can be increased or decreased to match the workload. Web farm platform technologies, such as Azure App Service, can automatically add or remove nodes at the request of the system administrator or automatically without human intervention.

- **Maintainability** – Nodes of a web farm can rely on a set of shared services, which results in easier system management. For example, the nodes of a web farm can rely upon a single database server and a common network location for static resources, such as images and downloadable files.

It's ready for load balance scenarios and to run under multiple containers as well.

5.1.4 Key Material

The Cryptography Keys (JWKS) are stored within Database and auto refresh it every 90 days. It uses ECDSA using P-256 and SHA-256 (ES256) by default.

Signing key rollover

While you can only use one signing key at a time, IdentityServer4 provide mechanisms to publish more than one validation key to the discovery document. This is useful for key rollover. So every 90 days a new Key is auto published following NIST SP 800-107 Rev. 1 best practices.

Algorithm

It uses Elliptic Curves through ECDSA using P-256 and SHA-256 as default, following RFC 7518 best practices.

Database Store

To manage keys, SSO use Jwks.Manager component. It provide many Algorithm, you can change it at *Startup.cs*

5.1.5 Contributing



We are open to community contributions.

First, Read this: [Being a good open source citizen](#). There are a couple of guidelines you should follow so we can handle this without too much effort.

How to contribute?

Looking to contribute something to Jp Project? **Here's how you can help.**

Contributing to Jp Project

Please take a moment to review this document in order to make the contribution process easy and effective for everyone involved.

Following these guidelines helps to communicate that you respect the time of the developers managing and developing this open source project. In return, we will be lovely persons that respect in addressing your issue or assessing patches and features.

The easiest way to contribute is to open an issue and start a discussion.

Using the issue tracker

The [issue tracker](#) is the preferred channel for [bug reports](#bug-reports), [features requests](#feature-requests) and [submitting pull requests](#pull-requests), but please respect the following restrictions:

- Please **do not** use the issue tracker for personal support requests.
- Please **do not** post comments consisting solely of “+1” or “:thumbsup:”. Use GitHub’s “reactions” feature instead.

Bug reports

A bug is a **demonstrable problem** that is caused by the code in the repository. Good bug reports are extremely helpful!

Guidelines for bug reports:

0. **Validate and lint your code** — to ensure your problem isn’t caused by a simple error in your own code.
1. **Use the GitHub issue search** — check if the issue has already been reported.
2. **Check if the issue has been fixed** — try to reproduce it using the latest *master* or development branch in the repository.

A good bug report shouldn’t leave others needing to chase you up for more information. Please try to be as detailed as possible in your report. What is your environment? What steps will reproduce the issue? Did you check the logs? All these details will help people to fix any potential bugs.

Example:

Short and descriptive example bug report title

A summary of the issue and the OS environment in which it occurs. If suitable, include the steps required to reproduce the bug.

1. This is the first step
2. This is the second step
3. Further steps, etc.

Any other information you want to share that is relevant to the issue being reported. This might include the lines of code that you have identified as causing the bug, and potential solutions (and your opinions on their merits).

Feature requests

Feature requests are welcome. Before opening a feature request, please take a moment to find out whether your idea fits with the scope and aims of the project. It’s up to *you* to make a strong case to convince the project’s developers of the merits of this feature. Please provide as much detail and context as possible.

Pull requests

Issue First Before even writing the first line of code raise an issue and get buy in on your proposal from the maintainers. There's several reasons for this, people might already be working on the issue, the issue might not be an issue or by design, but mainly just letting the community know your working on something, it gets "assigned" to you and you get implementation detail feedback early. All to reduce chance of redoing work or getting your contribution rejected.

Good pull requests—patches, improvements, new features—are a fantastic help. They should remain focused in scope and avoid containing unrelated commits.

Please ask first before embarking on any significant pull request (e.g. implementing features, refactoring code, porting to a different language), otherwise you risk spending a lot of time working on something that the project's developers might not want to merge into the project.

Adhering to the following process is the best way to get your work included in the project:

1. Fork the project, clone your fork, and configure the remotes:

```
# Clone your fork of the repo into the current directory
git clone https://github.com/<your-username>/JP-Project.git
# Navigate to the newly cloned directory
cd free-bootstrap-admin-template
# Assign the original repo to a remote called "upstream"
git remote add upstream https://github.com/brunohbrito/JP-Project.git
```

2. If you cloned a while ago, get the latest changes from upstream:

```
git checkout master
git pull upstream master
```

3. Create a new topic branch (off the main project development branch) to contain your feature, change, or fix:

```
git checkout -b <topic-branch-name>
```

4. Commit your changes in logical chunks. Please adhere to these [git commit message guidelines](#) or your code is unlikely to be merged into the main project. Use Git's [interactive rebase](#) feature to tidy up your commits before making them public.

5. Locally merge (or rebase) the upstream development branch into your topic branch:

```
git pull [--rebase] upstream master
```

6. Push your topic branch up to your fork:

```
git push origin <topic-branch-name>
```

7. [Open a Pull Request](#) with a clear title and description against the *master* branch.

IMPORTANT: By submitting a patch, you agree to allow the project owners to license your work under the terms of the [MIT License](#).

Platform

Backend of JpProject is built against ASP.NET Core and runs on .NET Framework 4.6.1 (and higher) and .NET Core 2.1 (and higher).

The Frontend SPA is built against Angular 6 and runs on Node and Angular Cli 6.

General feedback and discussions?

Please start a discussion on the [issue tracker](#).

5.1.6 Contributor Covenant Code of Conduct

Our Pledge

In the interest of fostering an open and welcoming environment, we as contributors and maintainers pledge to making participation in our project and our community a harassment-free experience for everyone, regardless of age, body size, disability, ethnicity, sex characteristics, gender identity and expression, level of experience, education, socio-economic status, nationality, personal appearance, race, religion, or sexual identity and orientation.

Our Standards

Examples of behavior that contributes to creating a positive environment include:

- Using welcoming and inclusive language
- Being respectful of differing viewpoints and experiences
- Gracefully accepting constructive criticism
- Focusing on what is best for the community
- Showing empathy towards other community members

Examples of unacceptable behavior by participants include:

- The use of sexualized language or imagery and unwelcome sexual attention or advances
- Trolling, insulting/derogatory comments, and personal or political attacks
- Public or private harassment
- Publishing others' private information, such as a physical or electronic address, without explicit permission
- Other conduct which could reasonably be considered inappropriate in a professional setting

Our Responsibilities

Project maintainers are responsible for clarifying the standards of acceptable behavior and are expected to take appropriate and fair corrective action in response to any instances of unacceptable behavior.

Project maintainers have the right and responsibility to remove, edit, or reject comments, commits, code, wiki edits, issues, and other contributions that are not aligned to this Code of Conduct, or to ban temporarily or permanently any contributor for other behaviors that they deem inappropriate, threatening, offensive, or harmful.

Scope

This Code of Conduct applies within all project spaces, and it also applies when an individual is representing the project or its community in public spaces. Examples of representing a project or community include using an official project e-mail address, posting via an official social media account, or acting as an appointed representative at an online or offline event. Representation of a project may be further defined and clarified by project maintainers.

Enforcement

Instances of abusive, harassing, or otherwise unacceptable behavior may be reported by contacting the project team at bhdebrito@gmail.com. All complaints will be reviewed and investigated and will result in a response that is deemed necessary and appropriate to the circumstances. The project team is obligated to maintain confidentiality with regard to the reporter of an incident. Further details of specific enforcement policies may be posted separately.

Project maintainers who do not follow or enforce the Code of Conduct in good faith may face temporary or permanent repercussions as determined by other members of the project's leadership.

Attribution

This Code of Conduct is adapted from the [Contributor Covenant][homepage], version 1.4, available at <https://www.contributor-covenant.org/version/1/4/code-of-conduct.html>

[homepage](#)

5.1.7 Contributors

Below our lovely Contributors!

5.1.8 Stress Tests Results

When running in production scenarios. Some endpoints has more requests than other. There are two that has a heavy workload in stressed scenarios.

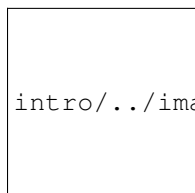
- *connect/token*
- *connect/introspect*

Why?

- Usually developers don't carry about generate a new token at each request to some API.
- There a huge amount of stuff that donn't use Token introspect endpoint, and when it is used, don't care about send it to server to validate at each request.

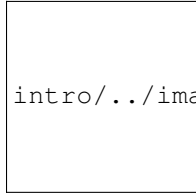
In fact the most used component for token introspection for ASP.NET users is *AccessTokenValidation*. It has a default cache of 5 minutes. But when are using another component for node.js, Springboot (java). it doesn't has cache capabilities. So the stress test was base in this case.

Generate token



```
intro/../../images/stress/connecttoken.png
```

Token introspect



5.1.9 First Use

In this section you will learn the basics to Get Ready!

Pre-requisites

To build solution you need to certify about these components first

- .NET Core 3.1
- node 8
- npm 5
- Angular CLI 8

Build Files

After `clone` the initial state of project is:

- Use MySql Server LocalDb
- Use Jwks.Manager as Key Material
- No migration.

Using build.bat

Open folder *build* and execute build.bat. The file will install Nuget and NPM dependencies. Then compile and run.

5.1.10 Docker

Now you can run through a docker!

Unfortunately you need to change hosts for it. Because Authority URL. I can't do anything to face it. It's security feature from OAuth2 to keep same Authority name for each Token.

Warning:

- Demo Database.
- This environment will not persist data once the containers stop running and is only suitable for basic testing.

Important: You must need to update hosts file in order to successful run.

Tutorial

If you do not already have a working IdentityServer installation up and running, or just want to demo AdminUI, then this walkthrough is for you.

The demo Docker Compose file will run Docker containers for Admin UI, plus containers test IdentityServer installation & database. It will get you up and running quickly with a full demo test environment.

This walk-through will assume you already have Docker installed on your machine.

The latest Docker compose file can be downloaded [here](#).

1 - Download

Download or clone project.

2 - Update HOST

Before you begin, add a map within your hosts file from your local IP address to “jpproject”.

As follows: 127.0.0.1 jpproject

To do so, go to project folder > build(folder) and execute update-hosts.bat as Administrator. Or follow the steps:

- **On Windows, your hosts file is usually found at C:\Windows\System32\Drivers\etc\hosts**
 - Add 127.0.0.1 jpproject at the end of file
- **On Linux, your hosts file is usually found at /etc/hosts**
 - Add 127.0.0.1 jpproject at the end of file

3 - Run docker-compose

Open terminal (cmd) from project folder and type:

Listing 1: CMD

```
docker-compose up
```

BE HAPPY!

Authority vs Issuer name

Authority name differs from Issuer name inside the docker ambient. The Authority and Issuer url must be the same across the environment. Look at the image:



If Authority differs from Issure the authorization will fail. There are some ways to do that:

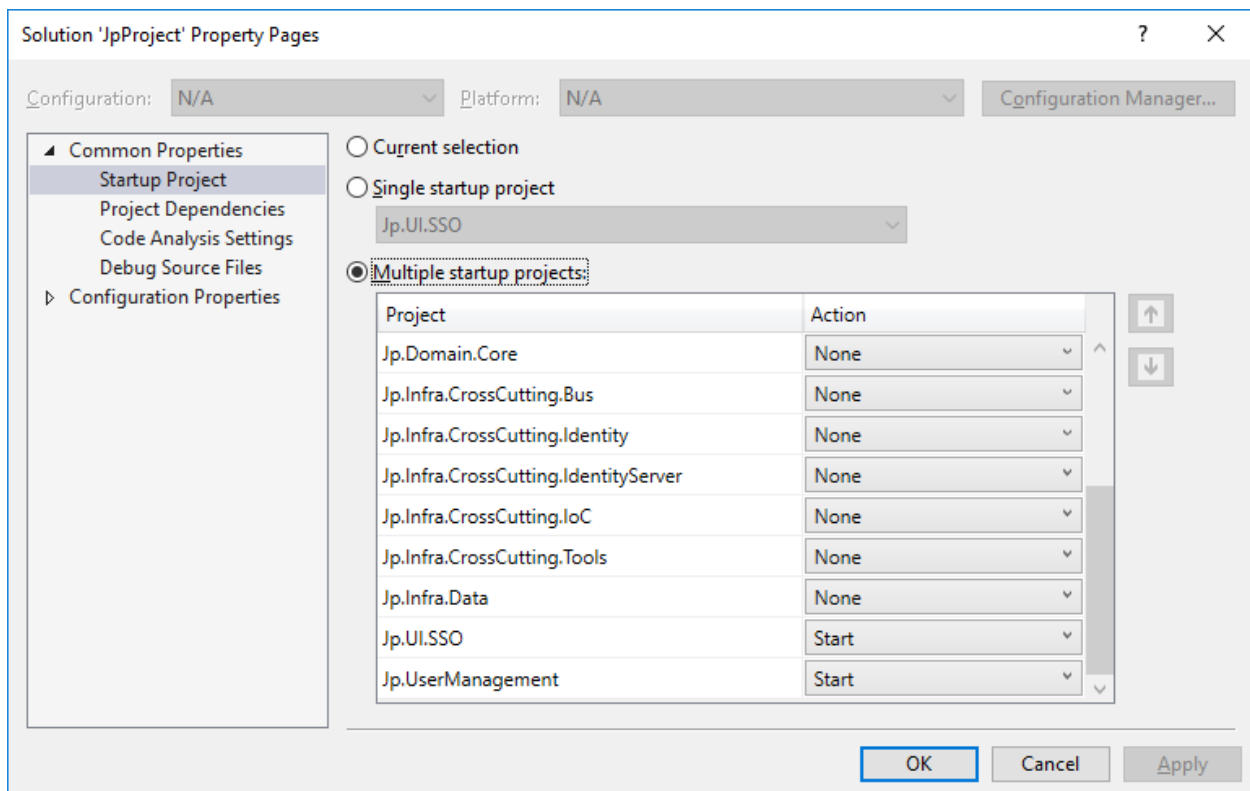
- Same DNS. Host machine and docker ambient in the same network. So it's possible to access them with the hostname, because they wil have same DNS.
- Change the Api hosts file. Change localhost with the same ip address of SSO.
- Change ID4 validation to ignore Issuer. It's not recomended, but [RFC6749](#) do not have specific rules about it.

5.1.11 VS and VSCode

The default way to Start the project.

SSO and API

To load project open src/JpProject.sln with Visual Studio. Now you need to set Multiple Startup Projects.



Run the project

Admin UI

Open VSCode then go to File > Open Folder > Locate srcFrontendJp.AdminUI.

Open Terminal CTRL + '. Type:

- npm install
- ng serve

Wait and open Browser at <http://localhost:4300>

User Management

Open VSCode then go to File > Open Folder > Locate srcFrontendJp.UserManagement.

Open Terminal CTRL + '. Type:

- npm install
- ng serve

Wait and open Browser at <http://localhost:4200>

Expected

After a success build and run, 3 consoles app's must be open:

5.1.12 Ambient variables

Using Ambient var you can minimize efforts to change configuration after publish.

SSO Variables

Table 1: Ambient Variables

Variable	Default	Docker	Expected in prod	Description
ASPNETCORE_ENVIRONMENT	Development	Development	Production	For more info access the default docs
ASPNETCORE_URLS	https://+:443;http://+:5000	http://+:5000	https://+:443;http://+:80	Set the ports for Https and Http. For more info docs
APPINSIGHTS_INSTRUMENTATIONKEY			Something like 762FAF25-9480-4AF7-8821-06875ED9266C	To create an Application Insights on Azure go to docs
CERTIFICATE_TYPE	File	Auto	File	Can be Temporary / File / Auto / Environment
USER_MANAGEMENT_URI	http://localhost:4200	http://localhost:4200		Url of User Management UI after published
IS4_ADMIN_UI	http://localhost:4300	http://localhost:4300		The path Url of Admin UI
RESOURCE_SERVER_URI	http://localhost:5002	http://localhost:5003		The path Url of Management API
CUSTOM_CONNECTION_STRING		DATABASE_CONNECTION at docker.compose		Database Connection String Specially made for Azure App Service
DATABASE_TYPE	SqlServer, Postgre, Sqlite	MySql		Which database will be used.

User Management API Variables

Table 2: Ambient Variables

Variable	Default	Docker	Expected in prod	Description
ASPNETCORE_ENVIRONMENT	Development	Development	Production	For more info access the default docs
ASPNETCORE_URLS	https://+:443;http://+:5003	http://+:5003	https://+:443;http://+:80	Set the ports for Https and Http. For more info docs
APPINSIGHTS_INSTRUMENTATIONKEY			Something like 762FAF25-9480-4AF7-8821-06875ED9266C	To create an Application Insights on Azure go to docs
AUTHORITY	https://localhost:5001	http://jpproject:5000		Authority URL
CUSTOM_CONNECTION_STRING		DATABASE_CONNECTION at docker.compose		Database Connection String Specially made for Azure App Service
DATABASE_TYPE	SqlServer or	MySql		Which database will be used.

5.1.13 App Settings

Here you can find detailed explanation about App Settings

There are Ambient Variables, such as Database Connection, that are equivalent in App Settings. In case that both of them are set. The project will respect the Ambient Variables first.

SSO Settings

Connection

The project use only one Database and all projects use the same Connection String name “SSOConnection”

CertificateOptions

There are two options to provide CertificateOptions.

- **File - You need to provide**
 - FileName: Entire path of certificate
 - FilePassword: Password of certificate
- **Store** - You need to provide KeyStoreIssuer. The certificate is search by X509FindType.FindByIssuerName. (Good option for Azure SSL)
- **Environment** - Will look into Env Var ASPNETCORE_Kestrel__Certificates__Default__Path and ASPNETCORE_Kestrel__Certificates__Default__Password to locate the certificate and it pass.
- **Temporary** - It will create an auto signed Certificate. Use only on dev’s environment.

EmailConfiguration

E-mail settings to provide the capabilities to send e-mail after a new user and reset link.

ExternalLogin

With these settings the SSO can provide External login.

To take token settings from Google, get [docs here](#)

To take from Facebook, see [docs here](#)

ApplicationSettings

These settings will be overridden by Environment Variables, if set.

Table 3: Ambient Variables

Variable	Value	Description
Authority	https://localhost:5000	The path Url of SSO
UserManagementUrl	http://localhost:4200	The path Url of User Management UI after published
IS4AdminUi	http://localhost:4300	The path Url of Admin UI
ResourceServerUrl	https://localhost:5003	The path Url of Management API
CUSTOMCONNSTR_<YOURDATABASE_CONNECTION>	YOURDATABASE_CONNECTION STRING>	Database Connection String
DatabaseType	MySql or SqlServer	Which database will be used.

User Management API

Connection

The project use only one Database and all projects use the same Connection String name “**SSOConnection**”

Storage

If user send his profile picture, then the project will upload it to Azure Blob container, so these options must be set.

To get more info how to take Key, see [docs here](#)

EmailConfiguration

E-mail settings to provide the capabilities to send e-mail after a new user and reset link.

ApplicationSettings

Need to provide the **DatabaseType**.

This settings will be override by Environment Variables if set.

User Management UI

environment files

To change settings from UI, you need to open environment file. Located at src/environment.

Table 4: UI Variables

Variable	Description
GoogleClientId	Same from ApplicationSettings, the UI use this to take user info while is registering himself.
FacebookClientId	Same from ApplicationSettings, the UI use this to take user info while is registering himself.
ResourceServer	Base Url from User Management API
IssuerUri	Base Url from SSO
RequireHttps	True if IssuerUri has a HTTPS.
Uri	Final URL of App.

Admin UI

environment files

To change settings from UI, you need to open environment file. Located at src/environment.

Table 5: UI Variables

Variable	Description
ResourceServer	Base Url from User Management API
IssuerUri	Base Url from SSO
RequireHttps	True if IssuerUri has a HTTPS.
Uri	Final URL of App.

5.1.14 Database Type

By default the project come with MySQL, to change is very simple. You just need to change the appsettings.json.

The file appsettings.json of **Jp.UI.SSO** and **Jp.UserManagement** has a DatabaseType property. Change it to MySQL, SqlServer, Postgre, Sqlite

Warning: Both of them must point to the same config Database. You just need to config **Connection String** at Environment or at appsettings.json. For Environment see [docs here](#)

SQL on Docker

Don't have the SQL Server on you local machine? Use it from Docker

```
docker run -e 'ACCEPT_EULA=Y' -e 'SA_PASSWORD=@Password1' -e 'MSSQL_PID=Express' -p_
↪1433:1433 -d microsoft/mssql-server-linux:latest
```

As simple as that!

5.1.15 Serilog

To log the project uses Serilog. There are several kinds of Configuration, you can see them at [Serilog homepage](#)

The file Program.cs of **Jp.UI.SSO** and **Jp.UserManagement** has the configuration of logs.

```
public static void Main(string[] args)
{
    ...
    Log.Logger = new LoggerConfiguration()
        .MinimumLevel.Debug()
        .MinimumLevel.Override("Microsoft", LogEventLevel.Warning)
        .MinimumLevel.Override("System", LogEventLevel.Warning)
        .MinimumLevel.Override("Microsoft.AspNetCore.Authentication", LogEventLevel.
↪Information)
        .Enrich.FromLogContext()
        .WriteTo.ApplicationInsightsEvents(Environment.GetEnvironmentVariable(
↪"APPINSIGHTS_INSTRUMENTATIONKEY"))
```

(continues on next page)

(continued from previous page)

```

        .WriteTo.File(@"jpProject_sso_log.txt")
        .WriteTo.Console(outputTemplate: "[{Timestamp:HH:mm:ss} {Level}]
↪{SourceContext}{NewLine}{Message:l}{NewLine}{Exception}{NewLine}", theme: ↪
↪AnsiConsoleTheme.Literate)
        .CreateLogger();
        ...
    }

    public static IWebHostBuilder CreateWebHostBuilder(string[] args) =>
        WebHost.CreateDefaultBuilder(args)
            .UseApplicationInsights(Environment.GetEnvironmentVariable("APPINSIGHTS_
↪INSTRUMENTATIONKEY"))
            .ConfigureLogging(builder =>
                {
                    builder.ClearProviders();
                    builder.AddSerilog();
                })
            .UseStartup<Startup>();

```

5.1.16 Application Insights

The project has the basis config of Application Insights. You can change it at `Program.cs`. There is a Component of Serilog too.

```

public static void Main(string[] args)
{
    ...
    // Serilog plugin
    .WriteTo.ApplicationInsightsEvents(Environment.GetEnvironmentVariable(
↪"APPINSIGHTS_INSTRUMENTATIONKEY"))
    ...
}

public static IWebHostBuilder CreateWebHostBuilder(string[] args) =>
    WebHost.CreateDefaultBuilder(args)
        // default Application Insights config
        .UseApplicationInsights(Environment.GetEnvironmentVariable("APPINSIGHTS_
↪INSTRUMENTATIONKEY"))
    ...

```

5.1.17 Certificate

Self-signed Certificate

To generate certificates, there is a Script `key\cert.ps1` that generate Self Signed Certificates. You can use it to generate certificate and user against docker, or test in your local.

Open up a PowerShell terminal and run the script. It'll ask you for a certificate name, password, and then save it for you.

Change Config file

If a certificate is generated, to use it, change the appsettings.json. See [docs](#) for more info