# App Features

|  |  |
| --- | --- |
| 1. Registration & Login with ASP.Net Identity 2. View list of members currently online 3. Like members and list 4. View members who liked them 5. Upload photo 6. Update member profile 7. Messaging (live chat) system to message member’s real time 8. Pagination 9. Caching | **What we will be using?**   * Entity Framework * HTML5 * Bootstrap * CSS * TypeScript * C# * Sqlite (DB) |

# Command Documents

* [000 Commands Angular.txt](000%20Commands%20Angular.txt)
* [000 Commands DotNet.txt](000%20Commands%20DotNet.txt)
* [000 Commands Node.txt](000%20Commands%20Node.txt)
* [000 Commands VS Code Short Cuts.txt](000%20Commands%20VS%20Code%20Short%20Cuts.txt)

# Installations

## VisualStudio Code Install

We’ll use VS Code as the code editor: <https://code.visualstudio.com/>

### Making VS Code IDE Better

1. AutoSave: Go to File and select “AutoSave” to automatically save our changes
2. Go to File > Preferences > Settings
   1. Type Font and change
      1. Main font size
      2. Scroll down and change for Console and Terminal as well
   2. Type exclude and add following to hide BIN and OBJ folders
      1. \*\*/bin
      2. \*\*/obj
   3. Type folders, go to Explorer:Compat Folder and unselect
   4. Type bracket,
      1. make sure that bracket pair colorization is checked
      2. select true for guides: bracket pairs
   5. Type auto close
      1. Auto closing brackets: always
      2. Auto closing comments: always
      3. Auto closing quotes: always

## .Net 8 Install

1. Got to dotnet.microsoft.com/download
2. Click download
3. Select your OS Windows/Linux/macOS/Docker
4. Install SDK x64/x86

## Install Postman

<https://www.postman.com/downloads/>

No need to create an account to use Postman. Towards the bottom there is skip link to skip login

### Collections Used

Import [Postman\_collection.json](../Postman_collection.json) to see the collections

## Node Install

### Install NVM

* <https://github.com/coreybutler/nvm-windows#readme>
* <https://www.freecodecamp.org/news/node-version-manager-nvm-install-guide/>

>nvm -v



|  |  |
| --- | --- |
| Install Node >nvm install latest  >nvm install 18.17.0 | Using node >nvm use 20.11.1  >node -v [18.17.0]  >npm --version [9.6.7] |

# Setting WebApi Project

Create a base folder to house the projects and files

## Creating Solution

|  |  |
| --- | --- |
| >dotnet new sln  Solution name the same as the container folder [Do not use] | > dotnet new sln --name MySocialConnectWebApi  Solution name with custom name [Use] |

## Creating MSC.WebApi Project

|  |  |
| --- | --- |
| >dotnet new webapi -o MSC.WebApi  Create a new project with name MCS.WebApi < dotnet 8 | >dotnet new webapi -n MSC.WebApi –use-controllers  Create a new project with name MCS.WebApi = dotnet 8 |

### Adding MSC.WebApi to Solution

>dotnet sln add MSC.WebApi

## Creating MSC.Core Project

>dotnet new classlib -o MSC.Core

### Adding MSC.Core to Solution

>dotnet sln add MSC.Core

## Reference MSC.Core in MSC.WebApi

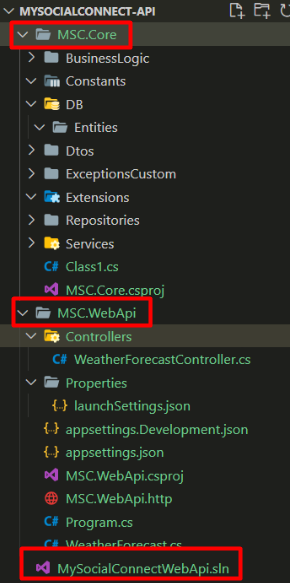
>cd MSC.WebApi

>dotnet add reference ../MSC.Core/MSc.Core.csproj

## Viewing all projects in the solution

>dotnet sln list

## Project Structure



## Updating the ApplicationURL

Open Properties > launchSettings.json

Update Profiles http and https,

* launchBrowser: false
* http applicationUrl: http://localhost:5000;https://localhost:5001

|  |  |
| --- | --- |
| From | To |
| {    "$schema": "http://json.schemastore.org/launchsettings.json",    "iisSettings": {      "windowsAuthentication": false,      "anonymousAuthentication": true,      "iisExpress": {        "applicationUrl": "http://localhost:11672",        "sslPort": 44324      }    },    "profiles": {      "http": {        "commandName": "Project",        "dotnetRunMessages": true,        "launchBrowser": true,        "launchUrl": "swagger",        "applicationUrl": "http://localhost:5013",        "environmentVariables": {          "ASPNETCORE\_ENVIRONMENT": "Development"        }      },      "https": {        "commandName": "Project",        "dotnetRunMessages": true,        "launchBrowser": true,        "launchUrl": "swagger",        "applicationUrl": "https://localhost:7164;http://localhost:5013",        "environmentVariables": {          "ASPNETCORE\_ENVIRONMENT": "Development"        }      },      "IIS Express": {        "commandName": "IISExpress",        "launchBrowser": true,        "launchUrl": "swagger",        "environmentVariables": {          "ASPNETCORE\_ENVIRONMENT": "Development"        }      }    }  } | {    "$schema": "http://json.schemastore.org/launchsettings.json",    "profiles": {      "http": {        "commandName": "Project",        "dotnetRunMessages": true,        "launchBrowser": false,        "launchUrl": "swagger",        "applicationUrl": "http://localhost:5000;https://localhost:5001",        "environmentVariables": {          "ASPNETCORE\_ENVIRONMENT": "Development"        }      }    }  } |

## Running .Net API Project

|  |  |
| --- | --- |
| * Open command prompt and navigate to “[basePath]/MySocialConnect-API/MSC.WebApi” * Then execute “dotnet run” or “dotnet watch run” commands. * Once running successfully then pick the url from the command prompt | Building...  For information about trusting the ASP.NET Core developer certificate, see https://aka.ms/aspnet/https-trust-dev-cert.  info: Microsoft.Hosting.Lifetime[14]  Now listening on: http://localhost:5000  info: Microsoft.Hosting.Lifetime[14]  Now listening on: https://localhost:5001  info: Microsoft.Hosting.Lifetime[0]  Application started. Press Ctrl+C to shut down.  info: Microsoft.Hosting.Lifetime[0]  Hosting environment: Development |

Then go to <http://localhost:5013/swagger/> or <https://localhost:5013/swagger>

Swagger will display. Expand the GET method end point under WeatherForecast, click Try it out and then click Execute. You should see result here.

## Trusting Dev Certificate

If you see warning then trust the certificate. If error shows for the certificate then first run clean command

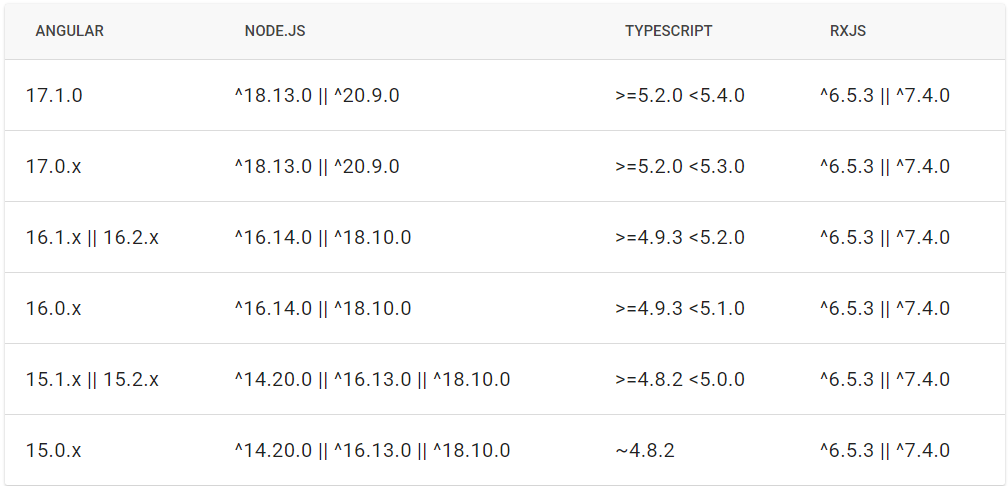
|  |  |
| --- | --- |
| >dotnet dev-certs https --clean  >dotnet dev-certs https --trust  Trusting the HTTPS development certificate was requested. A confirmation prompt will be displayed if the certificate was not previously trusted. Click yes on the prompt to trust the certificate.  Successfully trusted the existing HTTPS certificate. |  |

# Setting up Angular Project

## Node and Angular Version Compatibility

Visit this link to check compatibility: <https://angular.io/guide/versions>

Node version used is 20.11.1



## Angular Install V16

<https://angular.io/guide/setup-local>

Installing it globally. Nodejs must be installed first, v18.17.0 is being used

* npm uninstall -g @angular/cli
* npm cache clean --force
* npm install -g @angular/cli@16

>ng version

Angular CLI: 16.2.12

Node: 18.17.0

Package Manager: npm 9.6.7

OS: win32 x64

Angular:

...

Package Version

------------------------------------------------------

@angular-devkit/architect 0.1602.12 (cli-only)

@angular-devkit/core 16.2.12 (cli-only)

@angular-devkit/schematics 16.2.12 (cli-only)

@schematics/angular 16.2.12 (cli-only)

## Setting MySocialConnect-SPA Project

> ng new MySocialConnect-SPA

? Would you like to add Angular routing? Yes

? Which stylesheet format would you like to use? CSS

## Adding HTTPS – Certificate – MKCERT

Install chocolately: <https://chocolatey.org/install#individual>

Create a directory “ssl” in the SPA project

Naviagte to the ssl folder in command prompt [admin mode]

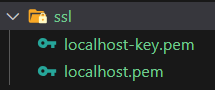
Wil be installing from this in ssl : <https://github.com/FiloSottile/mkcert>

>choco install mkcert

>mkcert -install

>mkcert localhost

This will create two files in the ssl folder



Go to angular.json > architect > serve and add the options for ssl

"options": {

            "ssl": true,

            "sslCert": "./ssl/localhost.pem",

            "sslKey": "./ssl/localhost-key.pem"

          },

## Run Angular App

>ng serve

|  |  |
| --- | --- |
| **Without ssl**  Build at: 2024-02-16T06:07:14.726Z - Hash: 43cae753a7878de5 - Time: 42808ms  \*\* Angular Live Development Server is listening on localhost:4200, open your browser on http://localhost:4200/ \*\* | **With ssl**  Build at: 2024-02-16T07:41:57.083Z - Hash: 43cbcd1aff4d61d0 - Time: 13665ms  \*\* Angular Live Development Server is listening on localhost:4200, open your browser on https://localhost:4200/ \*\* |

## Adding Different Components

### ngx-bootstrap

[Documentation](https://valor-software.com/ngx-bootstrap/#/documentation) | [Compatibility](https://valor-software.com/ngx-bootstrap/#/documentation) | [Installation](https://valor-software.com/ngx-bootstrap/#/documentation)

|  |  |
| --- | --- |
| > ng add ngx-bootstrap [USE]  > npm install ngx-bootstrap --save [Don’t use]  BrowserAnimationModule is gets added to the app.module as well | Check angular.json and make sure that paths are correct in styles:  "styles": [  "./node\_modules/ngx-bootstrap/datepicker/bs-datepicker.css",  "./node\_modules/bootstrap/dist/css/bootstrap.min.css",  "src/styles.css"  ] |

### font-awesome

|  |  |
| --- | --- |
| > npm install font-awesome | Angular.json  Add following before styles.css  "./node\_modules/font-awesome/css/font-awesome.min.css" |

### Dropdown component ngx-bootstrap dropdown component

> ng add ngx-bootstrap --component dropdowns

### ngx-toastr

|  |  |
| --- | --- |
| <https://github.com/scttcper/ngx-toastr>  version 17.0.2 is supports angular 16  > npm install ngx-toastr@17 --save   * Angular animation is needed but it is installed by angular. Check package.json for details * Open angular.json and add the style to styles array 🡺 * Finally add the BrowserAnimationModule and ToastrModule.forRoot() to the omports array in app.module.ts * First used on [004 Project5](004%20Project5%20-Ng-routing-toast-route-guard-shared-module-misc-components.docx) | "styles": [   "./node\_modules/ngx-bootstrap/datepicker/bs-datepicker.css",   "./node\_modules/bootstrap/dist/css/bootstrap.min.css",   "./node\_modules/font-awesome/css/font-awesome.min.css",  "./node\_modules/ngx-toastr/toastr.css",  "src/styles.css"              ] |

### Bootswatch for theming

<https://bootswatch.com/united/>

> npm install bootswatch

Add the theme unit css to angular.json

"styles": [

              "./node\_modules/ngx-bootstrap/datepicker/bs-datepicker.css",

              "./node\_modules/bootstrap/dist/css/bootstrap.min.css",

              "./node\_modules/bootswatch/dist/spacelab/bootstrap.min.css",

              "./node\_modules/font-awesome/css/font-awesome.min.css",

              "./node\_modules/ngx-toastr/toastr.css",

              "src/styles.css"

            ]

### ngx-gallery

* <https://www.npmjs.com/package/ng-gallery>
* <https://github.com/murhafsousli/ngx-gallery>
* Home page: <https://ngx-gallery.netlify.app/#/>
* Getting started: <https://ngx-gallery.netlify.app/#/getting-started/gallery>

>npm i ng-gallery @angular/cdk

If this gives an error then install a specific version

>npm i ng-gallery @angular/cdk@16.2.0

ng-gallery is stand alone component so it cannot be used inside a non stand alone component.

Check [007 Project8 -Ng-interceptor-membercards-photogallery.docx](007%20Project8%20-Ng-interceptor-membercards-photogallery.docx) for details

### ngx-spinner

* <https://napster2210.github.io/ngx-spinner/>
* <https://github.com/Napster2210/ngx-spinner>

For the full setup including the service and interceptor check: [008 Project9 -Ng-edit-templateform-guard-candeactivate-spinner-intercepter-caching.docx](008%20Project9%20-Ng-edit-templateform-guard-candeactivate-spinner-intercepter-caching.docx)

> npm install ngx-spinner

Or if an issues happens then install per the ng version of the app

>npm install ngx-spinner@16 --legacy-peer-deps

#### Angular.json

And then add the style for the spinner you want to use to angular.json styles array

./node\_modules/ngx-spinner/animations/line-scale-party.css

#### /core/modules/shared.module.ts or the app.modules.ts

Add it to the imports and exports array

NgxSpinnerModule.forRoot({type: 'line-scale-party'})

### ng2-file-upload

<https://valor-software.com/ng2-file-upload/>

npm install ng2-file-upload

#### /core/modules/shared.module.ts or the app.modules.ts

Add it to the imports and exports array

FileUploadModule

### ngx-timeago

<https://www.npmjs.com/package/ngx-timeago>

> npm install ngx-timeago

And add it to the shared module or app module

# Extensions and Packages

## C# Related Extensions

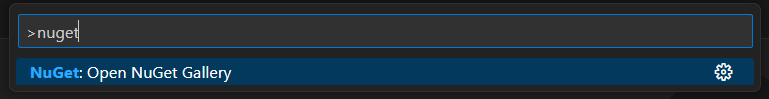
1. C# Dev Kit Microsoft – this will also install
   1. .NET Install Tool – Microsoft
   2. C# base language support – Microsoft
   3. IntekkiCode for C# Dev Kit
2. Material Icon Theme Philip Kief
3. NuGet Gallery from pcislo
4. SQLITE by alexcvzz
   1. SHIFT+CTRL+P then type sqlite and select and open the database. Applicable with/after Site-02-Entity-Cors.

## C# Packages

### Important

Must add Nuget Gallery extension above first

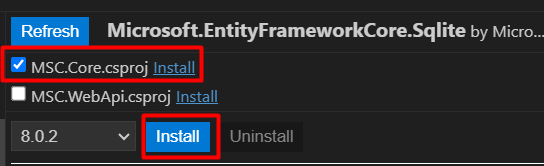
### To Add Packages



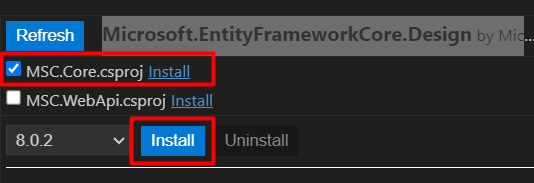
### Add EntityFramework

Search for “Microsoft.entityframeworkcore” and then install following

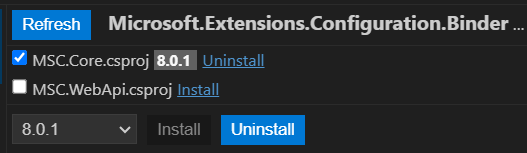
#### Microsoft.EntityFrameworkCore.Sqlite by Microsoft



#### Microsoft.EntityFrameworkCore.Design by Microsoft

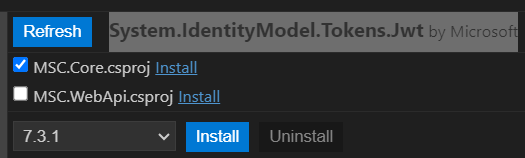


### Microsoft.Extensions.Configuration.Binder



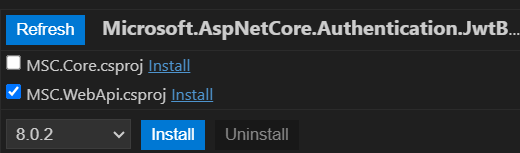
### NugetPackages for JWT Token

#### System.IdentityModel.Tokens.Jwt by Microsoft



#### Microsoft.AstNetCore.Authentication.JwtBearer by Microsoft

Install in both projects



### Add dotnet-ef

Search for “nuget dotnet-ef” online and then pick per the dotnet version installed. It will be installed globally

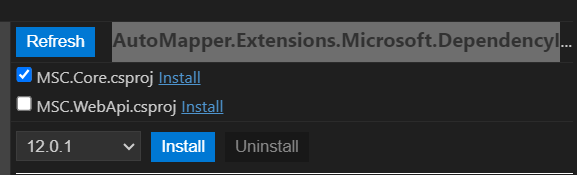
<https://www.nuget.org/packages/dotnet-ef/>

>dotnet tool install --global dotnet-ef --version 8.0.2

### AutoMapper

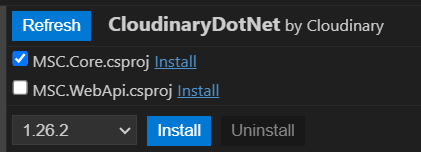
Open Nuget and search for automapper

Select AutoMapper.Extensions.Microsoft.DependencyInjection by Jimmy Bogard : AutoMapper extensions for ASP.NET Core



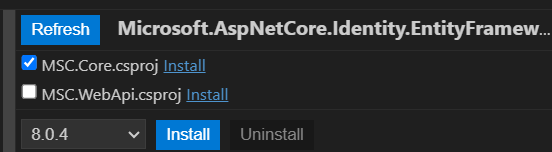
### Cloudinary

CloudinaryDotNet by Cloudinary



## Packages after Site-15 Identity Role Management

### Microsoft.AspNetCore.Identity.EntityFrameworkCore



## Angular Related Extensions

1. Angular Language Service – Angular

# Migrations and Database

## Commands

>dotnet ef migrations add InitialCreate -o DbFile/Migrations

>dotnet ef migrations remove

>dotnet ef database update

## Migrations

* dotnet ef migrations add InitialCreate -o DbFile/Migrations
  + any time a change a made is to the entities run the above command, just change the InitialCreate part and then run update database command from below.
* dotnet ef migrations remove

## Database Update

After migrations update the database

* dotnet ef database update

## Completely remove all migrations and start all over again

* dotnet ef database update 0
* dotnet ef migrations remove

## Drop database and recreate

* dotnet ef database drop
* dotnet ef database update

## Users File

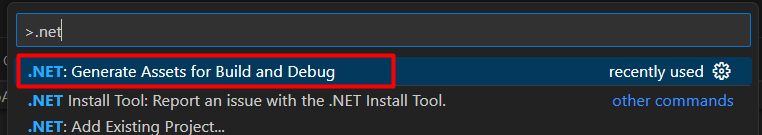
Check the users file in the root: [UserIds And Passwords.txt](../UserIds%20And%20Passwords.txt)

# Debugging API

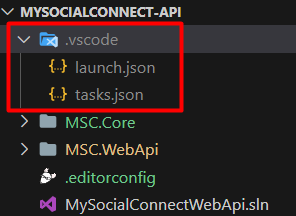
Open the solution project

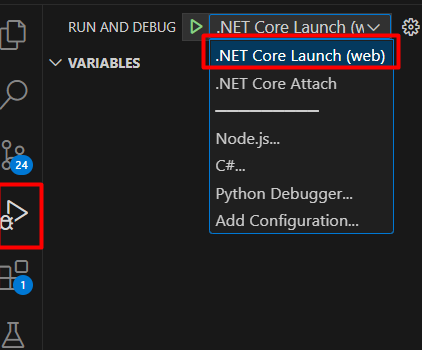
CTRL+SHIFT+P

Type .net and select “Generate assets for build and debug”



This will create





Alternatively run

dotnet watch –no-hot-reload

select .NET Core Attach

Search for your project and select it.