Creating a .NET Core 3.1 Web Application

# Configuration

This document describes the process to create a .NET Core website in Visual Studio with multiple projects. The projects will include:

* a website client using Typescript with modules and JQuery
* a web API
* a common area for services and models
* a test suite

I’m creating a solution called EntMgr.

# Create EntMgr Project

Follow these [instructions](https://docs.microsoft.com/en-us/visualstudio/javascript/tutorial-aspnet-with-typescript?view=vs-2019). By the end, you’ll have a Typescript-based web project that includes JQuery.

# Add Gulp

In tsconfig.json, remove the “outDir” line.

In the second half of these [instructions](https://www.typescriptlang.org/docs/handbook/asp-net-core.html), they discuss updating the “devDependencies” section of the package.json file. Start here and follow along. Stop when you get to the section “Write a HTML page”.

# Add Modules

Set up modules by changing compilerOptions in tsconfig.json to this:

"compilerOptions": {

"esModuleInterop": true,

"forceConsistentCasingInFileNames": true,

"module": "ES6",

"moduleResolution": "Node",

"noEmitOnError": true,

"noImplicitAny": true,

"removeComments": false,

"sourceMap": true,

"strict": false,

"target": "ES6"

},

Eventually you’ll want to set strict to true, but the boilerplate code causes an error if you do it now.

In \_Layout.cshtml, change the lines that add the Typescript files to this:

<script type="module" src="~/js/library.js"></script>

<script type="module" src="~/js/app.js"></script>

By now, you should have two Typescript files: app.ts and library.ts. In library.ts, add “export” in front of the var jqtest and remove the last line that calls jqtest.showMsg().

In app.ts, import jqtest:

import { jqtest } from "./library.js"

Now modify the code so that it uses jqtest in app.ts. See my code for an example. I also removed the button from Index.cshtml.

Older Attempt That Didn’t Work

# Configuration

This document describes the process to create a .NET Core website in Visual Studio with multiple projects. The projects will include:

* a website client using Typescript
* a web API
* a common area for services and models
* a test suite

I’m creating a solution called EntMgr.

# Create Solution

Create an empty solution, starting in the parent folder, using the command line:

# dotnet new sln -o EntMgr

# Create Typescript Website

Open Visual Studio and open the solution.

Create a new project in the solution (right-click the solution, Add > New Project).

Follow these [instructions](https://www.typescriptlang.org/docs/handbook/asp-net-core.html) to fill in the Typescript project, making sure you create the project in the ParentFolder/EntMgr folder.

To add jquery, follow these [instructions](https://www.c-sharpcorner.com/article/typescript-with-jquery/).

# Create API

Create a new project in the solution (right-click the solution, Add > New Project).

Choose ASP.NET Core Web Application template using C#.

Name it Api, set the location in ParentFolder/EntMgr.

Select API as the project template. Ensure it’s configured for HTTPS.

# TODO\_CAB: describe data store and test project creation

# Set Multiple Start Projects

Right-click solution, select Properties.

Common Properties > Startup Project: Select multiple startup projects.

Set Action to “Start” for both projects. I don’t know if order matters. Click OK.

# TODO\_CAB: Describe Setting Project Dependencies

# Configure CORS

For the website to call the API, we need to add a little code. Otherwise you’ll get an error.

Open the API’s Startup.cs.

Add the following to the beginning of ConfigureServices():

services.AddCors(options =>

{

options.AddPolicy("AllowSpecificOrigin",

builder => builder.AllowAnyOrigin().AllowAnyHeader().AllowAnyMethod());

});

Add the following to Configure() (not sure if it matters where in the method you add it):

# Modify Sample Code

To prove that the website can call the API, modify Api/Controllers/WeatherForecastController.cs. Change the values in Summaries to anything else.

In Web/