**Как сделать SPA на реакте в одном веб-сервере**

Создаем проект для WebApi.

**Конфигурируем реакт.**

Переходим в каталог WebApi и там вводим команду ;

npx create-react-app <имя приложения> --use-npm --template typescript

Это создает заготовку react-приложения с уже настроенным dev-сервером и подключенным typescript -> в каталоге WebApi создастся соответствующая папка.

Подключаем в проект нугет пакет Microsoft.TypeScript.MSBuild.

Если запустить проект командой npm start - будет предупреждение что-то вроде - "...onaftersetupmiddleware option is deprecated ..." Поэтому замещаем в папке D:\Projects\IdentityReact\IdentityReact\ClientApp\node\_modules\react-scripts\config файл webpackDevServer.config.js, его замещаем сохраненным файлом.

Ставим нужные пакеты, удаляем ненужные (например, web-vitals).

Проверяем – переходим в проект реакта, запускаем npm start, должно открыться приложение реакта.

Создаем в каталоге приложения реакта файл конфигурации реакта - .env В этом файле вводим параметр конфигурации билда –

BUILD\_PATH=../wwwroot

Выполняем команду – npm run build

По этой команде в каталоге выше – корневом каталоге WebApi создастся папка wwwroot с бандлом всех файлов.

**Конфигурируем WebApi**

Создаем контроллер FallbackController, в нем единственный метод:

return PhysicalFile(Path.Combine(Directory.GetCurrentDirectory(), "wwwroot", "index.html"), "text/HTML");

Класс контроллера помечаем как - [AllowAnonymous] для доступа.

ВАЖНО – в конфиге WebApi добавить доступ к статичным файлам, иначе не запуститься реакт:

app.UseDefaultFiles();

app.UseStaticFiles();

В appsettings – конфигурируем порты, сертификаты.

В итоге получаем WebApi, который работает со статичными файлами и который при обращении по корневому адресу – передает управление на папку wwwroot, в которой находится бандл реакта.

Если нам нужно запускать уже скомпилированный exe, а не из проекта командой dotnet run - копируем папку wwwroot в папку bin - где находится запускаемый файл.