

Ejercicio 1

The screenshot shows the Swagger UI for 'WebApplication2 v1' at the URL 'https://localhost:7044/swagger/index.html'. The 'WeatherForecast' endpoint is selected, showing a GET method. The response body is displayed as a JSON array of weather forecast objects. The response headers are also visible.

WeatherForecast

GET /weatherforecast

Schemas

WeatherForecast >

Response body

```
{
  "date": "2023-09-08T12:51:57.4285124-03:00",
  "temperature": 45,
  "temperaturef": 112,
  "summary": "Hild"
},
{
  "date": "2023-09-09T12:51:57.4289899-03:00",
  "temperature": -5,
  "temperaturef": 23,
  "summary": "Bracing"
},
{
  "date": "2023-09-10T12:51:57.4289927-03:00",
  "temperature": 1,
  "temperaturef": 34,
  "summary": "Bracing"
},
{
  "date": "2023-09-11T12:51:57.428993-03:00",
  "temperature": 71,
  "temperaturef": 160,
  "summary": "Rally"
},
{
  "date": "2023-09-12T12:51:57.4289932-03:00",
  "temperature": 81,
  "temperaturef": 177,
  "summary": "Cool"
}
```

Response headers

```
content-type: application/json; charset=utf-8
date: Thu, 07 Sep 2023 15:51:56 GMT
server: Kestrel
```

```
$ dotnet add package Newtonsoft.Json
Determinando los proyectos que se van a restaurar...
Writing C:\Users\Usuario\AppData\Local\Temp\tmpD37C.tmp
info : X.509 certificate chain validation will use the default trust store selected by .NET for code signing.
info : X.509 certificate chain validation will use the default trust store selected by .NET for timestamping.
info : Agregando PackageReference para el paquete "Newtonsoft.Json" al proyecto "C:\Users\Usuario\source\repos\WebApplication2\WebApplication2.csproj".
info : GET https://api.nuget.org/v3/registration5-gz-semver2/newtonsoft.json/index.json
info : OK https://api.nuget.org/v3/registration5-gz-semver2/newtonsoft.json/index.json 475 ms
info : Restaurando paquetes para C:\Users\Usuario\source\repos\WebApplication2\WebApplication2.csproj...
info : GET https://api.nuget.org/v3-flatcontainer/newtonsoft.json/index.json
info : OK https://api.nuget.org/v3-flatcontainer/newtonsoft.json/index.json 804 ms
info : GET https://api.nuget.org/v3-flatcontainer/newtonsoft.json/13.0.3/newtonsoft.json.13.0.3.nupkg
info : OK https://api.nuget.org/v3-flatcontainer/newtonsoft.json/13.0.3/newtonsoft.json.13.0.3.nupkg 31 ms
info : Se instaló Newtonsoft.Json 13.0.3 de https://api.nuget.org/v3/index.json con el hash de contenido Hrc58Xd100IP9zeV+0Z848QwPa0Cr9P3bDEZguI+gkLcBKA0xix/tLEAAHC+UvDNPv4a2d1810ReHMOagPa+zQ==.
info : El paquete "Newtonsoft.Json" es compatible con todos los marcos de trabajo especificados del proyecto "C:\Users\Usuario\source\repos\WebApplication2\WebApplication2.csproj".
info : Se agregó PackageReference para la versión "13.0.3" del paquete "Newtonsoft.Json" al archivo "C:\Users\Usuario\source\repos\WebApplication2\WebApplication2.csproj".
info : Generación de archivo MSBuild C:\Users\Usuario\source\repos\WebApplication2\obj\WebApplication2.csproj.nuget.g.props.
info : Generación de archivo MSBuild C:\Users\Usuario\source\repos\WebApplication2\obj\WebApplication2.csproj.nuget.g.targets.
info : Escribiendo el archivo de recursos en el disco. Ruta de acceso: C:\Users\Usuario\source\repos\WebApplication2\obj\project.assets.json
log : Se ha restaurado C:\Users\Usuario\source\repos\WebApplication2\WebApplication2.csproj (en 3.46 sec).
```

```
<Project Sdk="Microsoft.NET.Sdk.Web">

  <PropertyGroup>
    <TargetFramework>net6.0</TargetFramework>
    <Nullable>enable</Nullable>
    <ImplicitUsings>enable</ImplicitUsings>
  </PropertyGroup>

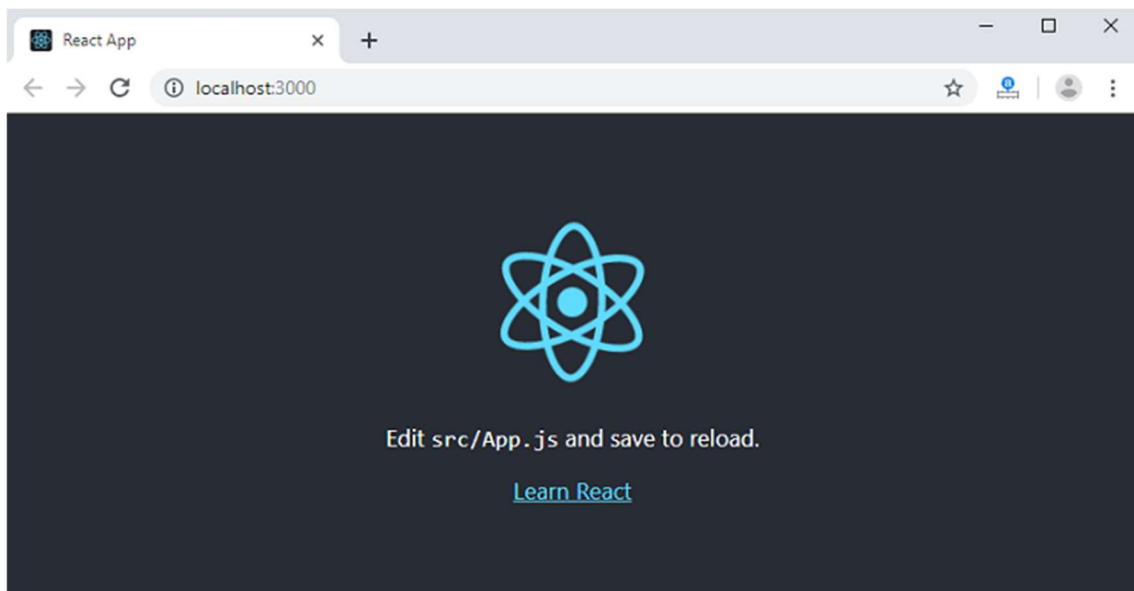
  <ItemGroup>
    <PackageReference Include="Newtonsoft.Json" Version="13.0.3" />
    <PackageReference Include="Swashbuckle.AspNetCore" Version="6.5.0" />
  </ItemGroup>

</Project>
```

Crear App react

```
npx create-react-app my-app
```

```
cd my-app
npm start
```



- Hacer una lista de herramientas de build (una o varias) para distintos lenguajes, por ejemplo (Rust -> cargo)

1. **Python:** Python utiliza herramientas como **setuptools** y **pip** para administrar paquetes y construir aplicaciones.
2. **Java:** Para Java, una de las herramientas más conocidas es **Maven**, que gestiona dependencias y construcción de proyectos.
3. **C:** En C, **Make** es una herramienta común para la construcción de programas, aunque existen alternativas modernas como **CMake**.
4. **JavaScript:** Para el ecosistema JavaScript, **npm** (Node Package Manager) y **yarn** se utilizan para gestionar dependencias y construir proyectos.
5. **C++:** Además de CMake, **Bazel** es una herramienta popular para compilar proyectos en C++.
6. **C#:** Microsoft proporciona **MSBuild** como herramienta de construcción para proyectos .NET.
7. **PHP:** **Composer** es una herramienta comúnmente utilizada para gestionar dependencias en proyectos PHP.
8. **Swift:** En el mundo de Swift, **Swift Package Manager** se utiliza para construir y administrar dependencias.
9. **Ruby:** En Ruby, **RubyGems** y **Bundler** son herramientas para gestionar dependencias y construcción de proyectos.
10. **Go:** El lenguaje Go viene con su propia herramienta de construcción llamada "**go build**", que facilita la construcción de programas en Go.