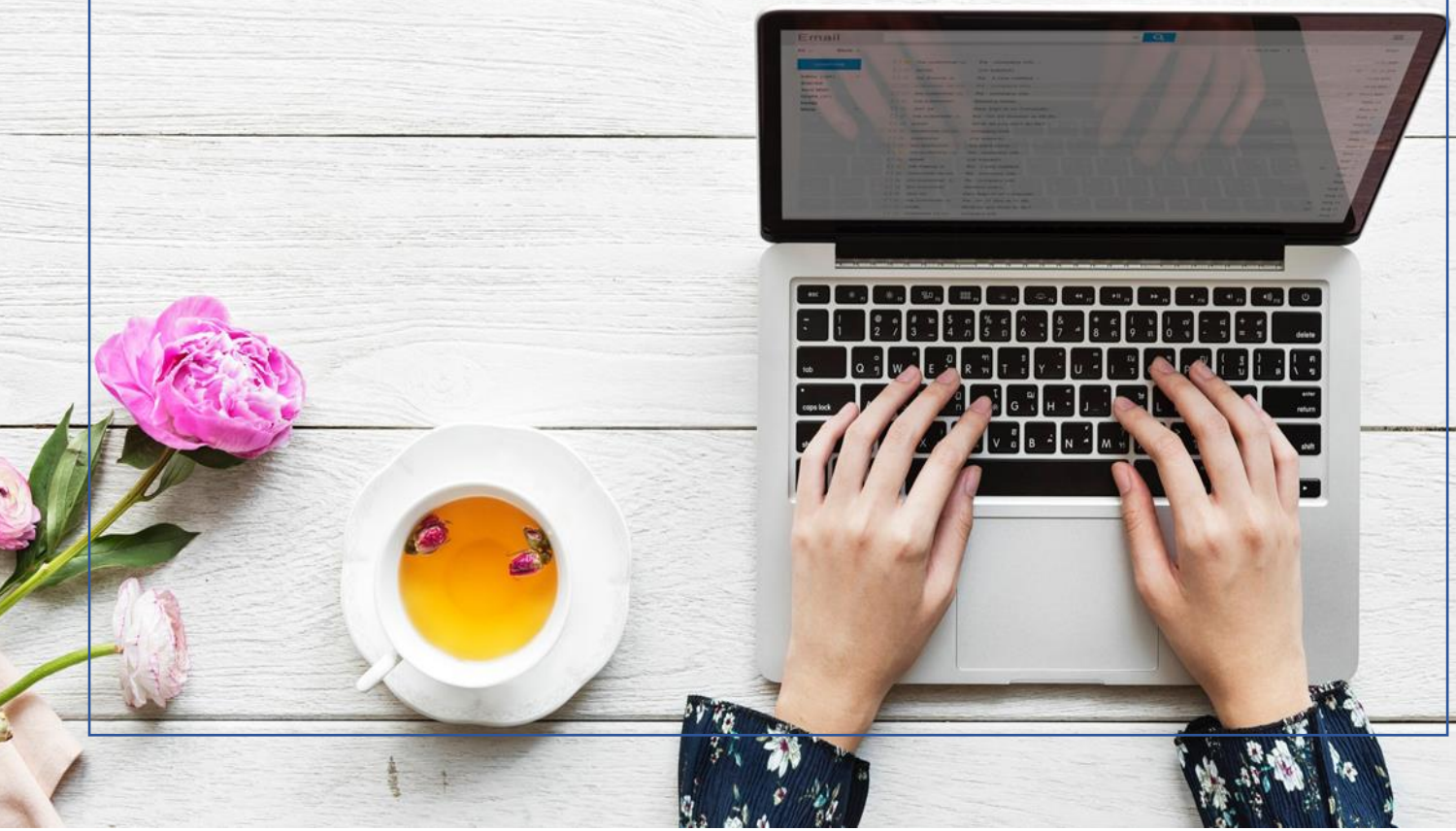
Laura Feü

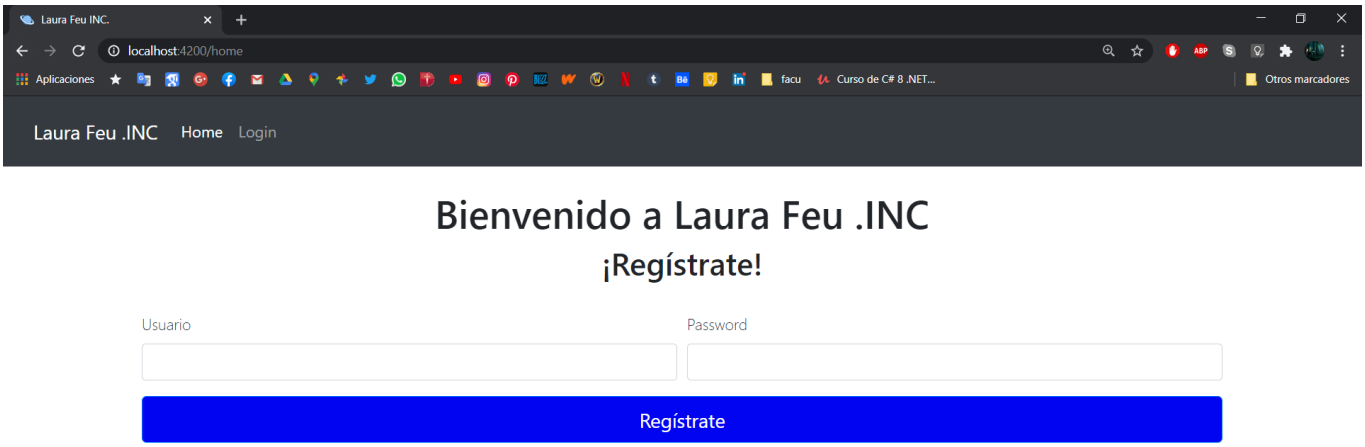
Dicsys

Implementación de una WebAPI con .NET y Angular



INTRODUCCIÓN

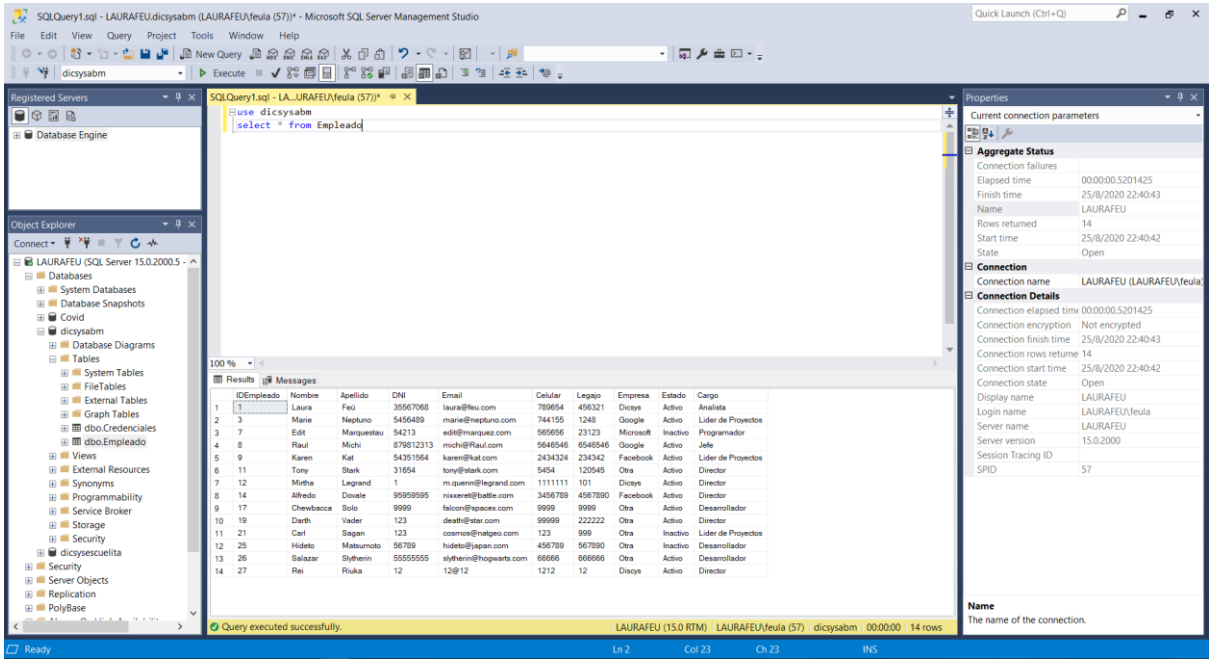
A continuación se presentan los pasos que se llevaron a cabo para crear una ABM-WebApi Utilizando las tecnologías .NET y Angular. Se describe el proceso, los programas utilizados y los problemas surgidos a medida que se iba desarrollando el proyecto así mismo como sus soluciones.



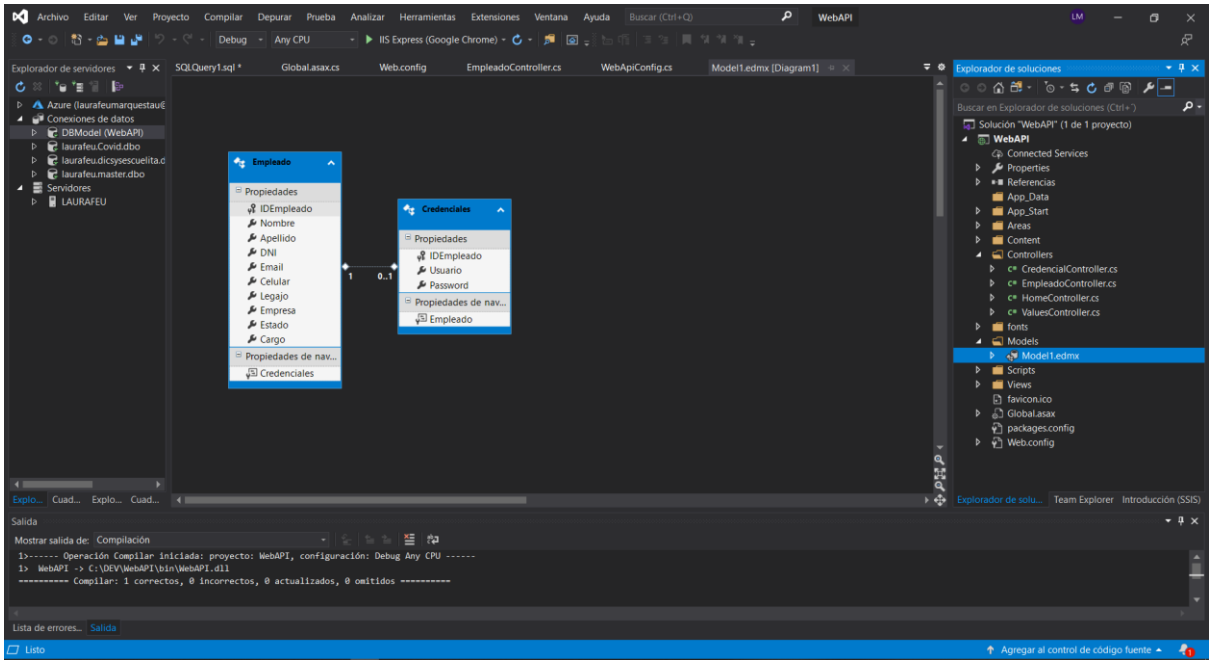
DESCRIPCION DEL MODELO

Utilicé DatabaseFirst como modo de trabajo con el objetivo de que sea el Entity Framework el que se encargue de crear las clases necesarias para trabajar con los datos.

La base de datos fue creada desde en Microsoft SQL Server, donde se crearon dos tablas, una donde contiene todos los datos de los empleados y otra donde contiene los Usuarios y Contraseñas de los mismos.

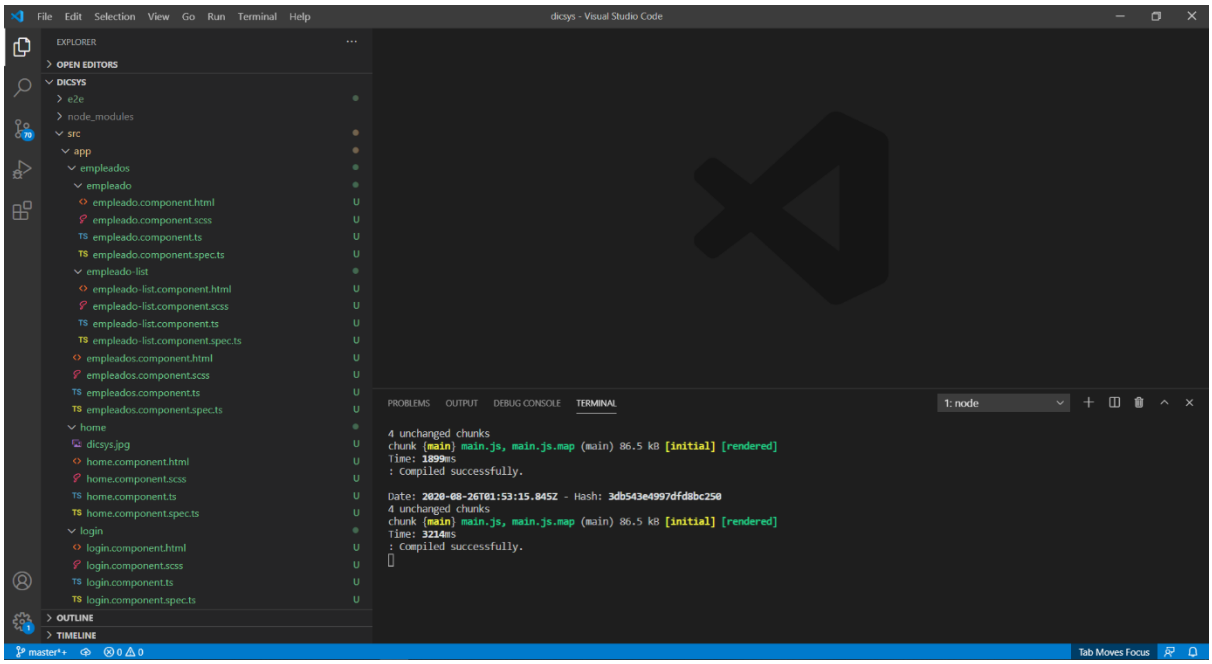


Una vez realizada la base de datos, en Visual Studio creamos un proyecto nuevo WebApi en el Entity Framework. Una vez generado el proyecto desde la carpeta Model conectamos a nuestra base de datos, como se puede ver a continuación:



Para proceder a realizar la parte del FrontEnd decidí utilizar Angular. Para esto realicé los siguientes pasos:

1. Se instaló Visual Studio Code
2. Se instaló Node js.
3. Se instaló Angular 10.0.11 a través de la Terminal del VS Code.
4. Se crea el proyecto en Angular con el comando “ng new dicsys”



Algunos de los comandos que me sirvieron para la creación de mi proyecto fueron:

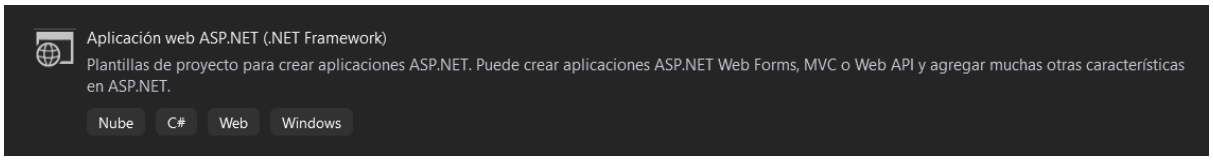
ng g cl empleado --type=model → Para crear modelos

ng g s empleado → Para crear servicios

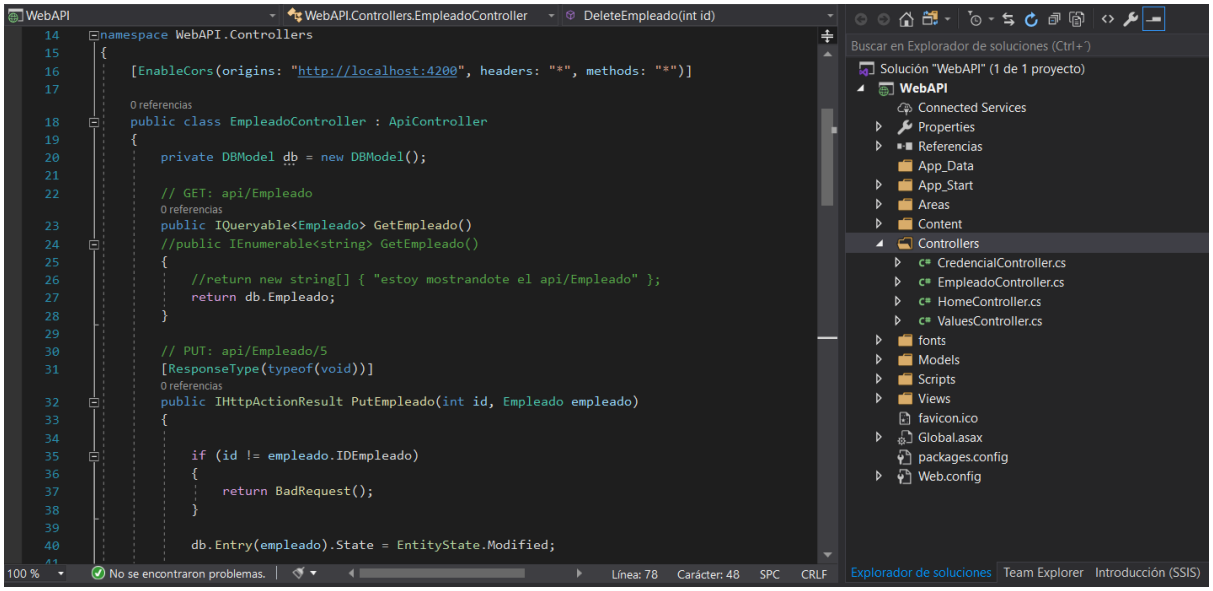
ng g c empleado-list → Para crear componentes

BACK-END

Como había mencionado previamente, mi backend lo generé con .NET desde Microsoft Visual Studio, para esto generé un nuevo proyecto del tipo “Aplicación WEB ASP.NET (NET Framework) y lo ubiqué en la ruta C:\DEV



Luego procedí a crear el modelo y finalmente los controladores que me permiten conectarme con el front-end.



FRONT-END

El front-end se realizó en Angular, siguiendo la siguiente estructura:

```
■ src
+------- ■ app
|         +------ ■ home
|         |        |-- home.component.ts|.html|.scss
|         |       ■ login
|         |        |-- login.component.ts|.html|.scss
|         |       ■ empleados
|         |        |-- empleados.component.ts|.html|.scss
|         |        + -- ■ empleado
|         |              |-- empleado.component.ts|.html|.scss
|         |             ■ empleado-list
|         |              |-- empleado-list.component.ts|.html|.scss
|         |       ■ shared
|         |        |-- empleado.service.ts
|         |        |-- empleado.model.ts
|         |        |-- login.service.ts
|         |        |-- login.model.ts
|         |
|         --app.module.ts
|
|-- index.html
```

PROBLEMAS DETECTADOS Y SUS SOLUCIONES

1. **Problema: CORS**
   Solución: Agregar en el Web.Config lo siguiente después de <system.webServer>

```
<httpProtocol>
        <customHeaders>
                <add name="Access-Control-Allow-Origin" value="*" />
                <add name="Access-Control-Allow-Headers" value="Content-Type" />
                <add name="Access-Control-Allow-Methods" value="GET, POST, PUT, DELETE, OPTIONS" />
        </customHeaders>
</httpProtocol>
```

   Luego seguí obteniendo errores de CORS y me fijé en varios foros (stackoverflow) donde algunos aconsejaban sacar las siguientes líneas:

```
                <add name="Access-Control-Allow-Origin" value="*" />
                <add name="Access-Control-Allow-Headers" value="Content-Type" />
```

   El Web.Config me quedó así entonces después del <system.webServer>

```
<httpProtocol>
        <customHeaders>
                <add name="Access-Control-Allow-Methods" value="GET, POST, PUT, DELETE, OPTIONS" />
        </customHeaders>
</httpProtocol>
```

2. **Problema:“**el-tipo-objectcontent1-no-pudo-serializar-el-cuerpo-de-respuesta-para-el-tipo-de-contenido
   Solución: En el Global.asax para evitar referencias circulares en la base

```
//Evito las referencias circulares al trabajar con Entity FrameWork
GlobalConfiguration.Configuration.Formatters.JsonFormatter.SerializerSettings.ReferenceLoopHandling =
Newtonsoft.Json.ReferenceLoopHandling.Serialize;
GlobalConfiguration.Configuration.Formatters.JsonFormatter.SerializerSettings.PreserveReferencesHandling =
Newtonsoft.Json.PreserveReferencesHandling.Objects;

//Elimino que el sistema devuelva en XML, sólo trabajaremos con JSON
GlobalConfiguration.Configuration.Formatters.Remove(GlobalConfiguration.Configuration.Formatters.XmlFormatter);
```

   Referencia: http://www.resuelvetusproblemas.com/el-tipo-objectcontent1-no-pudo-serializar-el-cuerpo-de-respuesta-para-el-tipo-de-contenido/

3. **Problema:** Error: The 'Access-Control-Allow-Origin' header contains multiple values 'http://localhost:4200, *', but only one is allowed.

   Solución: https://stackoverflow.com/questions/22343384/the-access-control-allow-origin-header-contains-multiple-values

   Elimino linea de Web.Config

```
<add name="Access-Control-Allow-Origin" value="*" />
```

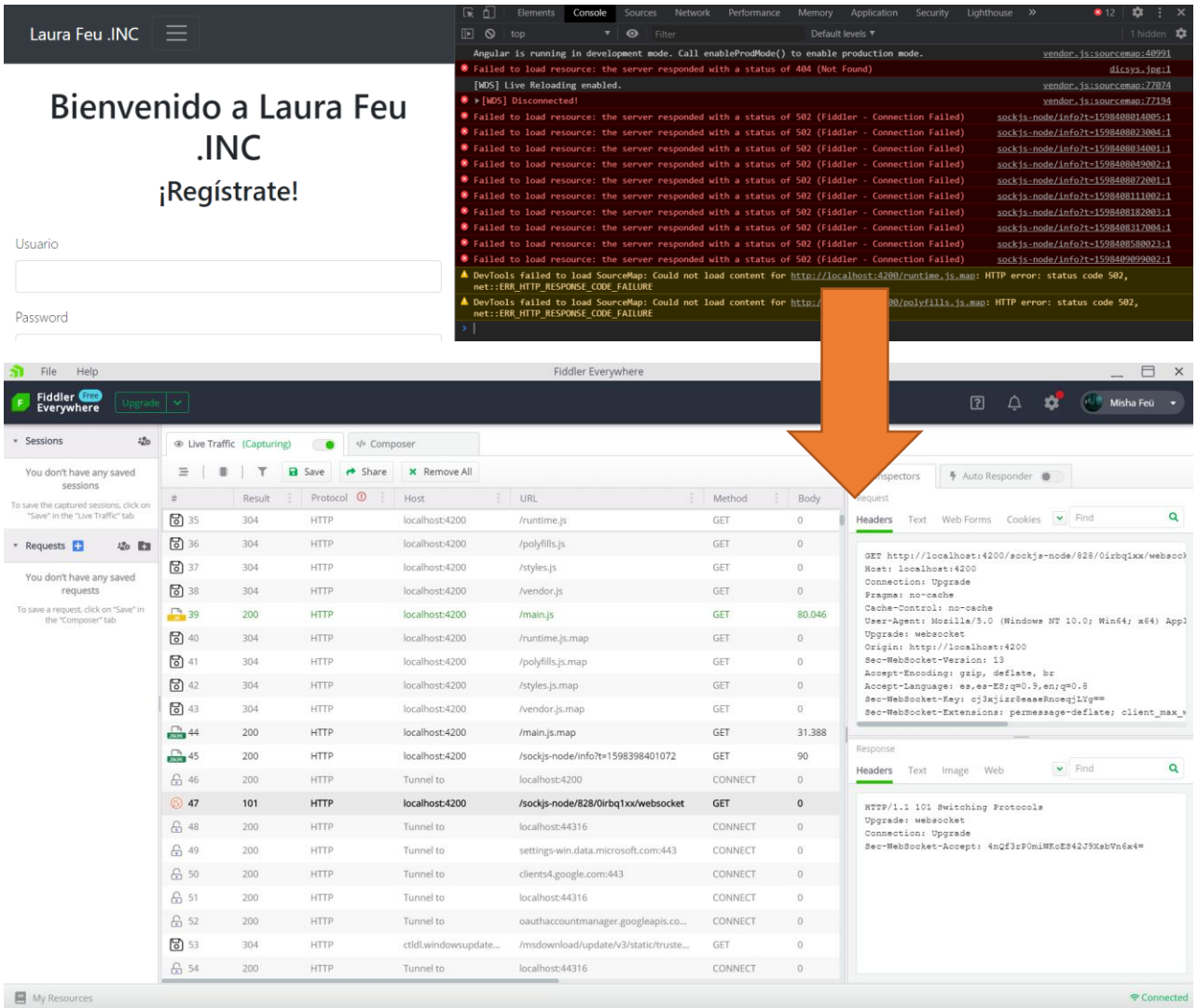
4. Problema: "Error converting value {null} to type 'System.Int32'. Path 'IDEmpleado', line 1, position 18."

Solución/es: Cuando quiero agregar un nuevo registro "empleado" luego de haber agregado funciones de modificación y borrado, obtengo el error indicado arriba

Si saco del empleado.component.html la línea: <input type="hidden" name="IDEmpleado" #IDEmpleado="ngModel" [(ngModel)]="service.formData.IDEmpleado">

Si funciona el agregado, pero no modifica (si borra) por la lógica que tengo en el typescript

Notas: en la mayoría de casos de error los debuggeaba fijándome en la consola del explorador y con ayuda de Fiddler Everywhere que me ayudaba a ubicar en qué parte estaba el error y podía ir a buscar en google algunas soluciones.



CONCLUSIÓN

Fue una gran experiencia crear esta WEBApi, aunque aún tiene cosas por mejorar, por ejemplo, me gustaría agregarle en el front-end una plantilla para hacerla visualmente más atractiva probar varias opciones extras que ofrece Bootstrap y además de arreglar los errores que aún quedaron por resolver.

Me llevo un gran aprendizaje de esta experiencia.

Gracias.