Informe

Miguel Alejandro Rodríguez Hernández

Para la implementación del proyecto Cine+ se me fue asignado la tarea de trabajar en el **backend** de la aplicación, específicamente en el área de la base de datos. Confeccionando los modelos y ofreciendo una forma limpia de acceso a los datos que están en el contexto de la base de datos.

La primera tarea que debía cumplirse para comenzar con el proyecto era la creación de un contexto con una base de datos **SQLite**. La forma en la que logré hacer esto posible fue siguiendo el ejemplo de libro **Adam Freeman - Pro ASP.NET Core 3_ Develop Cloud-Ready Web Applications Using MVC, Blazor, and Razor Pages (2020, Apress)** que se encontraba en **EVEA**, en este libro el autor se enfrentaba al mismo objetivo con la única diferencia de que él iba a trabajar con **SQLServer**. El proceso comenzó con la creación de un archivo **.cs** llamado **CineDbContext** donde programé un clase con el mismo nombre la cual hereda de **IdentityDbContext** una clase propia de **ASPNET** utilizada para declarar contextos de base de datos que también apoya los servicios de registro e inicio de sesión. Luego añadí el servicio del contexto al proyecto en el archivo **Startup.cs** y por último instalé el nuget necesario para trabajar con **SQLite**.

Siguiendo en mi misión tocaba realizar un **modelo entidad relacional extendido (MERX)** siguiendo las indicaciones del informe de la orientación del proyecto. Durante el proceso de confección fui discutiendo con mi equipo si les parecía correcto los pasos que estaba dando pues todos debíamos tener conocimiento de las entidades con las que luego trabajaríamos.

Lo próximo que realicé fue la implementación de los modelos en los **.cs** que fui creando en la carpeta **Models** para almacenar cada entidad con sus atributos. Modelos como **Movie**, **Show** o **Ticket**. Para lograr esto me fue necesario leer la documentación de **ASPNET** que ofrece **Microsoft** en **Internet**. La documentación me fue muy útil a la hora de aprender como se representaban las relaciones entre las entidades (**Uno a cero o uno**, **uno a muchos** y **muchos a muchos**). Luego añadí al contexto de la base de datos una propiedad **DbSet** por cada entidad para decirle al proyecto que confeccionara las tablas de esas entidades en la base de datos. Finalmente solo quedaba realizar la migración, lo cual hice siguiendo otro ejemplo del libro.

A medida que el trabajo en el proyecto progresaba se hizo necesario empezar a acceder a la base de datos. Gracias a un vídeo en **Youtube** del canal **kudvenkat** pude aprender como podía implementar una interfaz de repositorios para conseguir una forma limpia de trabajar con una entidad del contexto gracias al **Dependency Injection**. Los repositorios son uno de modelos con métodos que consideramos necesarios para trabajar y luego un repositorio de base de datos que implementaba el de algún modelo, así cumplíamos con el uso de interfaces del que habla el principio **SOLID**. Algunos repositorios de los modelos son genéricos para evitar la repetición de cogido innecesario.

También fue necesario que pusiera en **Starup.cs** los servicios de tipo **AddScoped** por cada repositorio de base de datos con su respectivo repositorio de modelo que fuéramos a usar.