UNIVERSIDAD NACIONAL DE HURLINGHAM

CARRERA: TECNICATURA UNIVERSITARIA DE INFORMÁTICA

Construcción de Interfaces de Usuario (CIU) y Estrategia de Persistencia (EP)

AÑO: 2019

Informe de desarrollo de API

(Frontend y Backend)

Profesores a cargo:

Gabriel Molina (CIU)

Pablo Marcelli (EP)

Andrés Atencio (CIU y EP)

**Alumnos Integrantes:**

Nicolás de la Fuente

Eduardo Campagno

Adrián Fazio

1. Objetivos:

El objetivo global fue el de crear una API llamada Cual curso.

Debido a que la API es muy grande, la misma fue dividida entre los equipos creados por los docentes, en las siguientes partes:

- Carrera

- Aula

- Docente

- Cursada

- Estudiante

- Materia

Cada equipo se dedicó a crear la API de dos partes de ella. En el caso de nuestro grupo enfocamos el trabajo en la parte de **Docente** y **Cursada**.

Nuestro objetivo particular fue el de crear una API que realizara un sistema de Alta, Baja y Modificación (**ABM**) de Docentes.

Por problemas de tiempo no se llegó a realizar la parte de la API en donde lo que se tenía que generar era un **ABM** de las cursadas.

1. Alcances de la aplicación

El alcance de la **API** fue la creación de pantalla de Alta, Baja y Modificación de los Docentes y Cursadas, como también del lado del backend se creó una base de datos en mariaDB para que persistan los datos de la misma.

1. Para el Backend

Para poder desarrollar la API utilizamos

* 1. **Docker** (para poder levantar cada uno de los servicios abajo mencionados sin necesidad de instalarlos en la PC)
  2. **Mariadb** como Base de Datos (**BD**)
  3. **Sequelize** como **ORM** para poder conectarnos entre nuestra **API** y la **BD**
  4. **Postman** para realizar las pruebas de manejo de datos (Alta, Baja, Modificación y Solicitud de los mismos) utilizando los métodos http **POST**, **GET**, **PUT** y **DELETE**

Las entidades que creamos fueron, docentes, cursadas y docentecursadas.

Para la creación de las entidades en la Base de datos utilizamos migraciones con sequelize. Lo hicimos por línea de comandos siguiendo estos pasos:

**Generamos el modelo para la migración**.

sudo npx sequelize-cli model:generate --name docente --attributes nombre:string, apellido:string,dni:integer

sudo npx sequelize-cli model:generate --name cursada --attributes nombre:string

sudo npx sequelize-cli model:generate --name docentecursada --attributes docenteId:integer,cursadaId:integer

**Configuramos el archivo de conexión a la base de datos:**

myapp/config/config.json

**Creamos la bd**

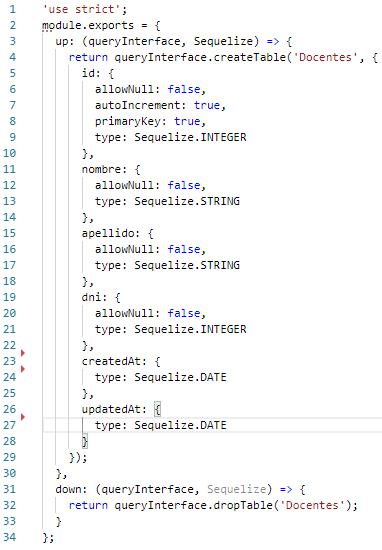
Manualmente por línea de comandos usando el **CREATE DATABASE**.

**Corrimos la migración**

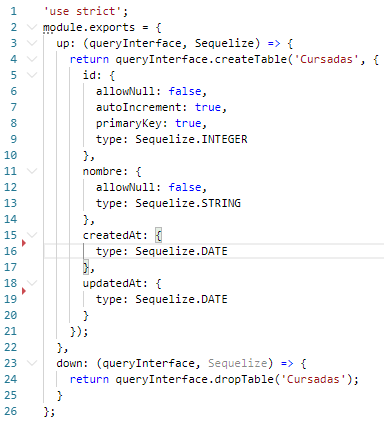
sudo npx sequelize db:migrate

Así quedaron las migraciones:

Migración Docentes:



Migración Cursadas:

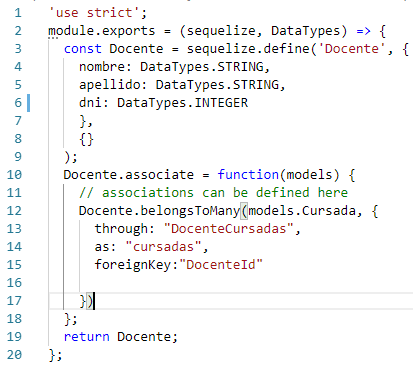


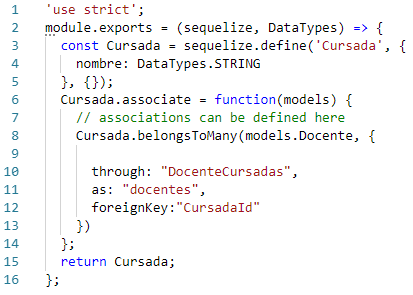
Migración DocenteCursadas:



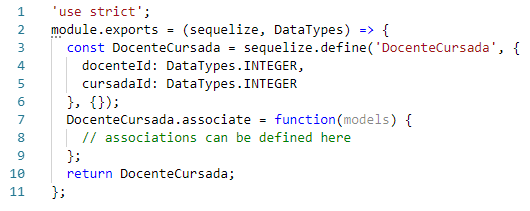
También tuvimos que crear las relaciones de las mismas, para ello utilizamos este código:

Entidad Docentes:

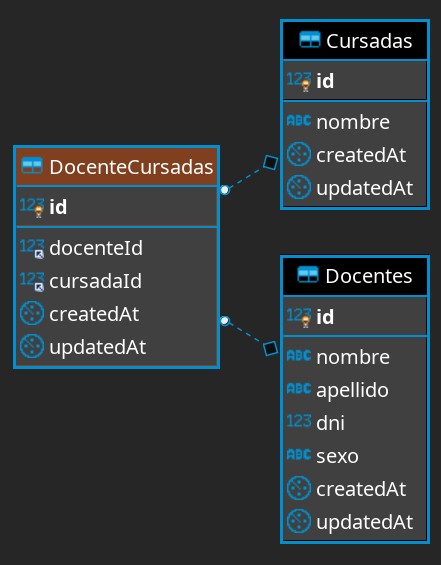


Entidad Cursadas:

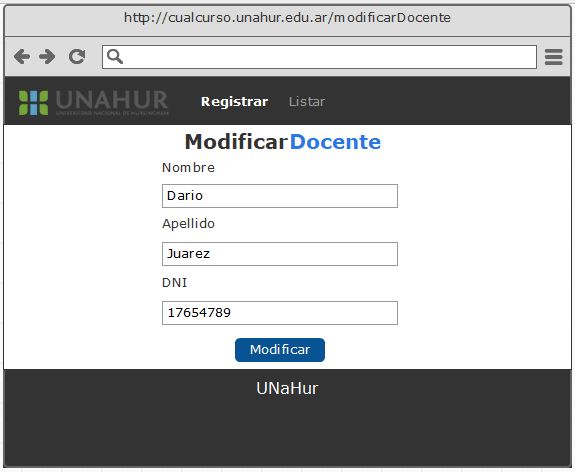
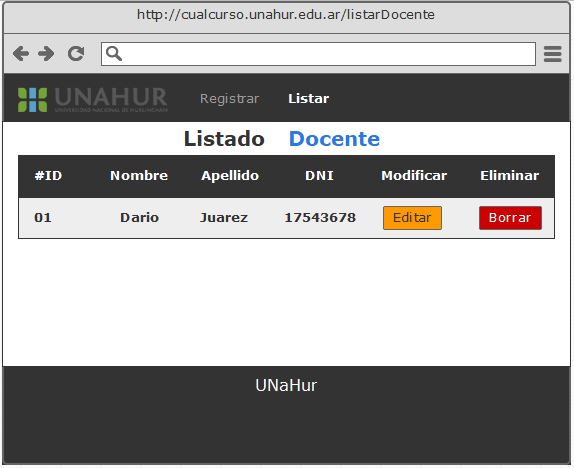
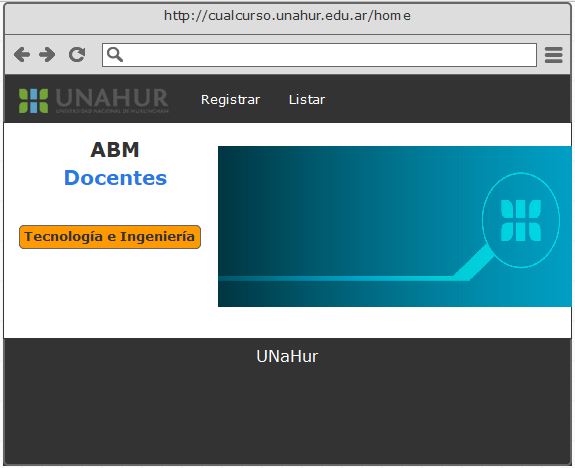
Y la entidad que las relaciona para salvar la relación muchos a muchos es la de DocenteCursadas:



El diagrama UML quedó de la siguiente forma:



1. Para el Frontend
   1. Creamos los siguientes Mockups para la parte de web:

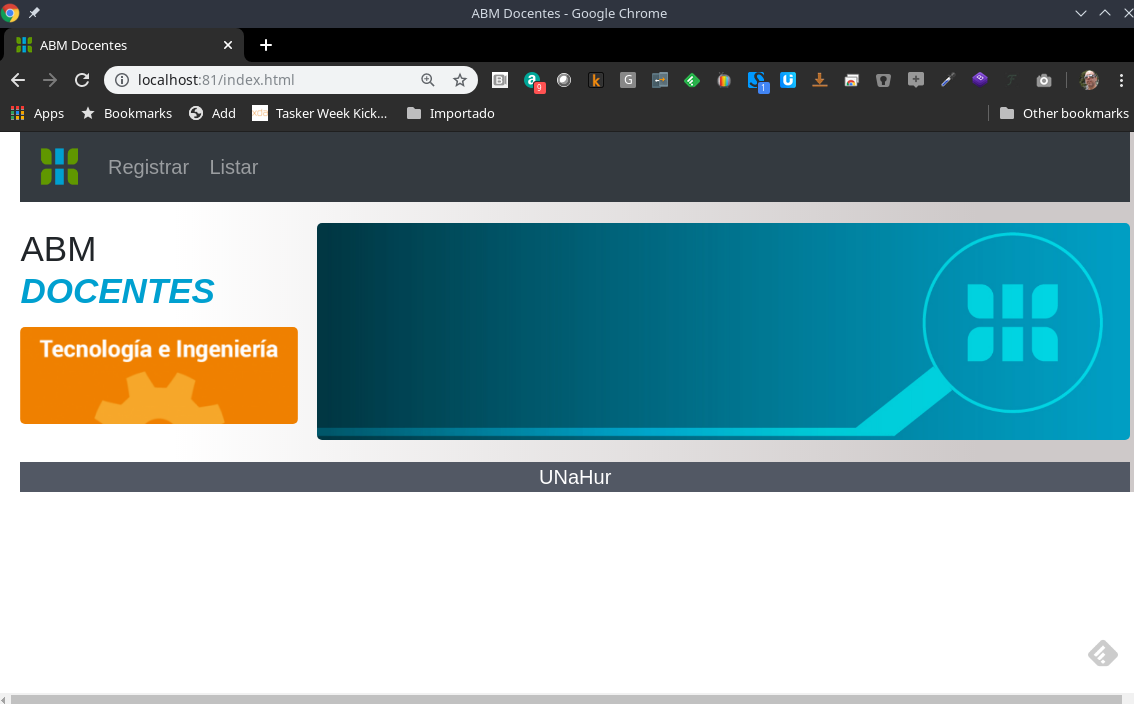


Y para la parte de Mobile:

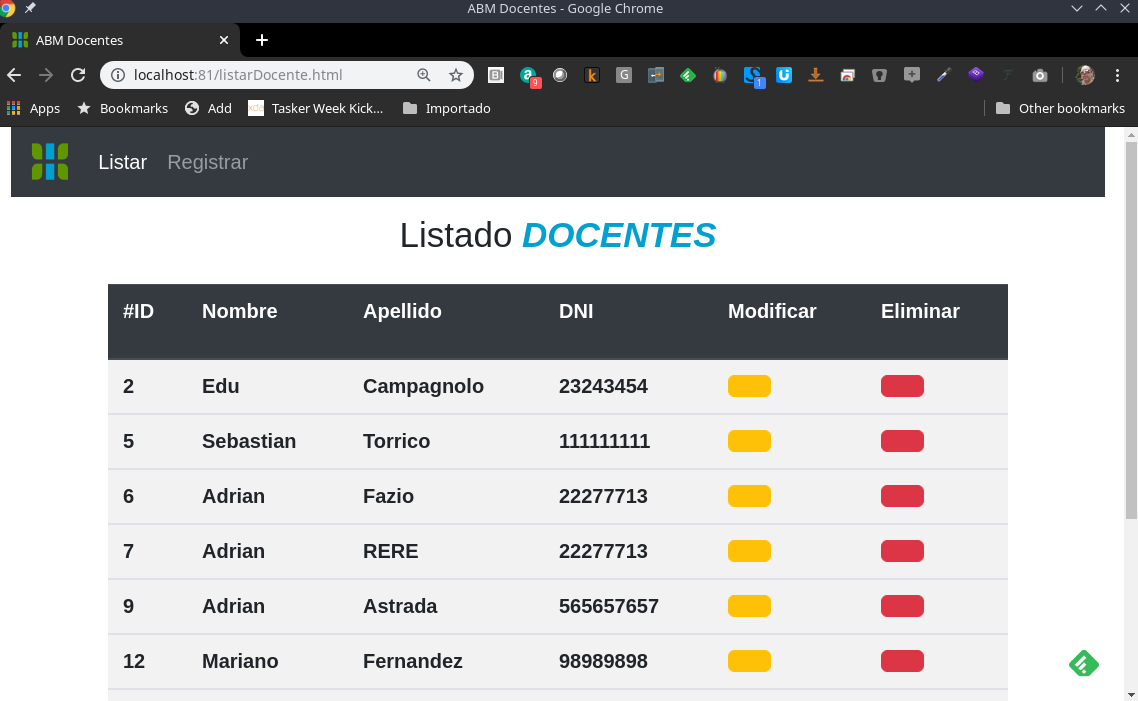


**Capturas de las pantallas implementadas**

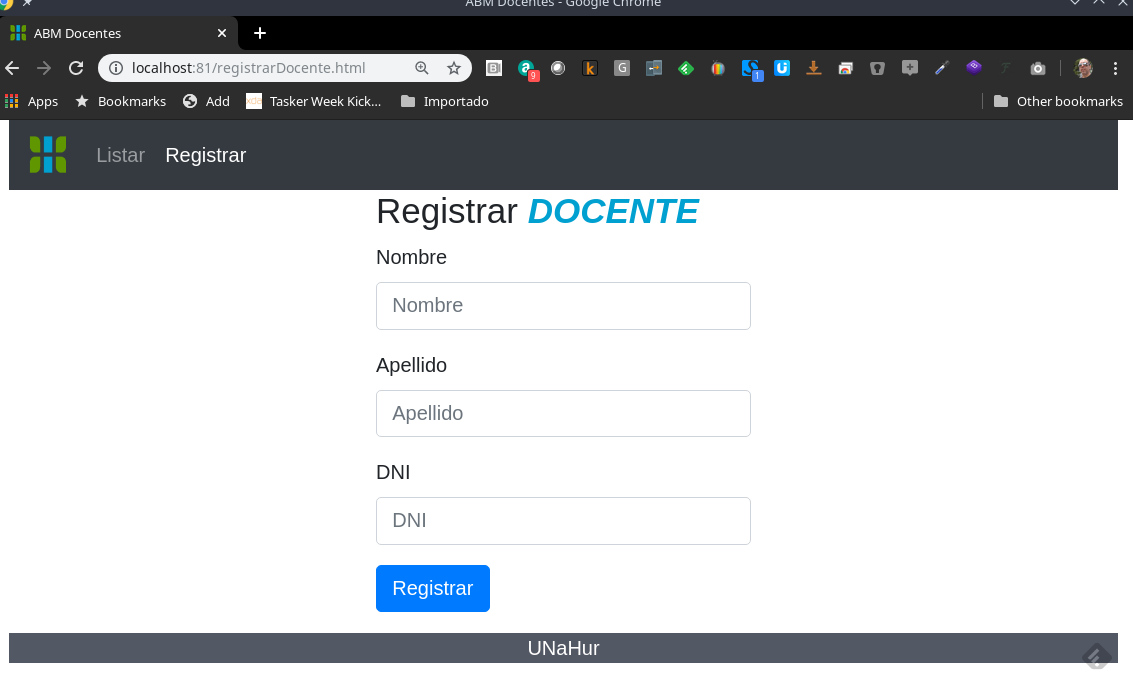
Pantalla inicial:



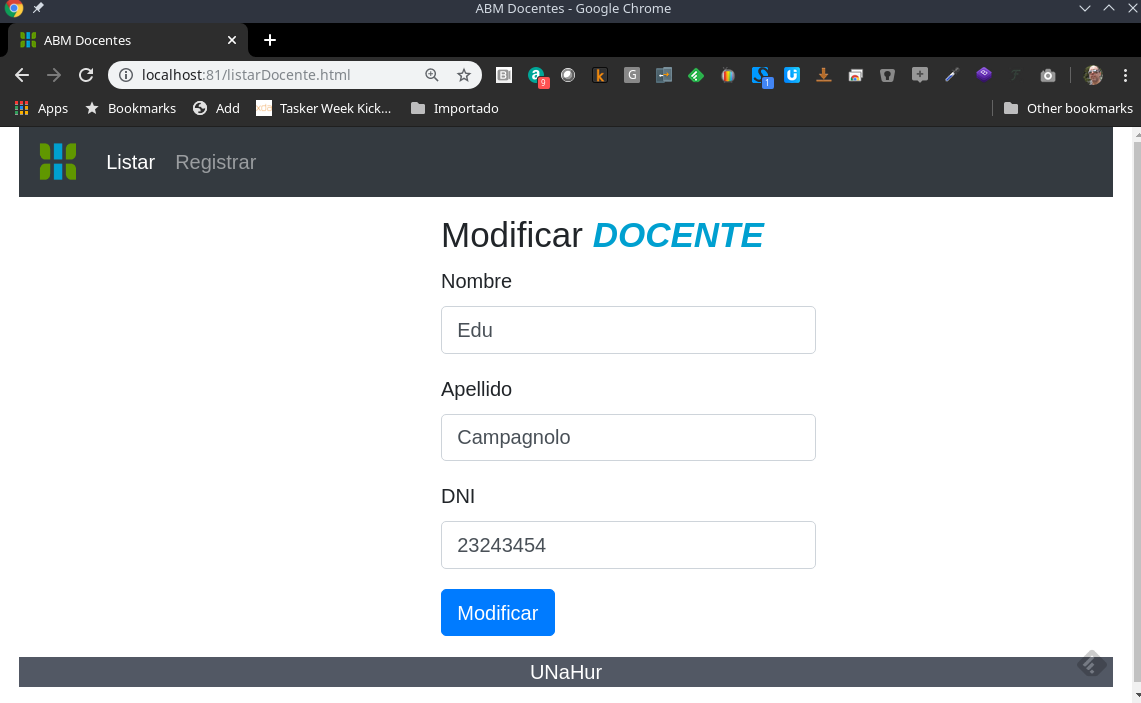
Pantalla Listar Docente:



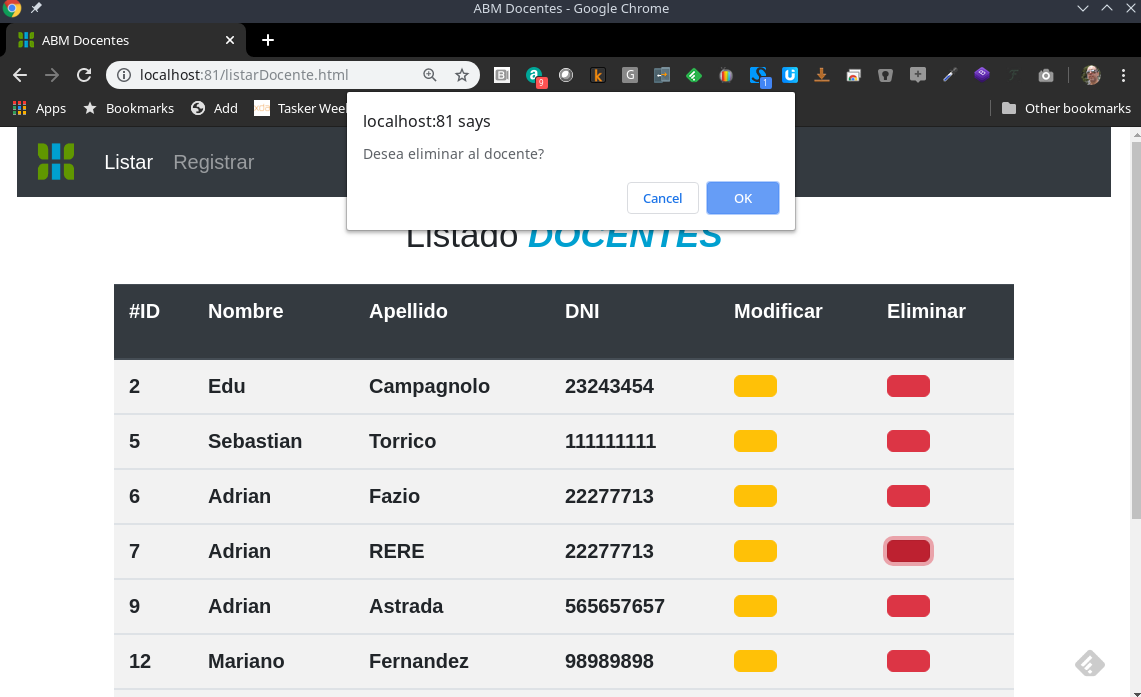
Registrar Docente:



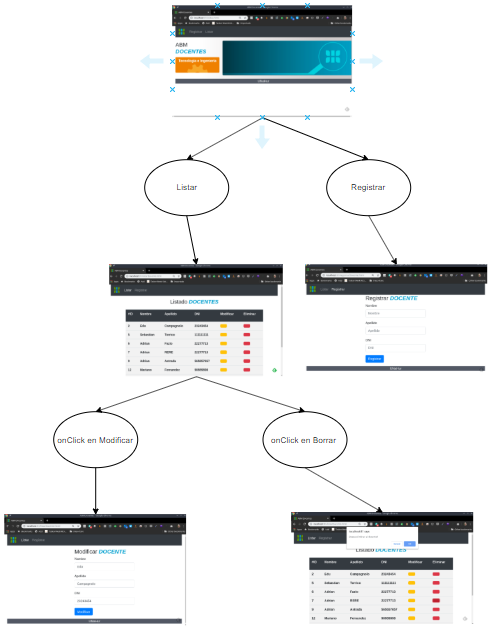
Modificar Docente: (haciendo click sobre el cuadrado amarillo en el registro a modificar en la pantalla de Listado de Docentes aparece la pantalla de Modificar con los datos a modificar)



Pantalla Eliminar Docente: (seleccionado el cuadrado rojo sobre el registro a eliminar y pide confirmación sobre dicha acción)



* 1. Como es la interacción de las pantallas



1. Utilizamos Github para el desarrollo de la API, el mismo está ubicado en:

<https://github.com/unahur/cual-curso#prob%C3%A1-la-api>

Posicionándose en el branch de grupo-docente.

El desarrollo se basó en HTML, JavaScript para darle funcionalidad y CSS para darle estilos a las pantallas.

Después se realizó una versión en React para poder tener una “Single Page Application” (**SPA**)

1. Conclusiones:
   1. **Dificultades**:

Al principio tuvimos dificultades para poder utilizar las aplicaciones con docker ya que se nos dificultó la instalación del soft y sus dependencias.

También en la parte de JavaScript y React tuvimos que investigar mucho sin una base como para empezar y fue algo que nos demoró mucho en entender cómo se hacía o lo que hacía.

1. **Objetivos alcanzados:**

Pudimos alcanzar los objetivos de crear la aplicación de **ABM** de Docentes, pero por tiempo no pudimos con la parte de **ABM** de Cursada.

1. Posibles mejoras

Lo ideal sería que puedan interactuar entre todas la **APIs** creadas por los demás grupos de la cursada así poder tener una aplicación completa de Inscripciones para la UNAHUR.