

Instituto Tecnológico de Costa Rica

Área Académica de Ingeniería en Computadores

Bases de Datos

Grupo 2

Proyecto I: Documentación Técnica

Profesor: Luis Diego Noguera Mena

Estudiantes:

- José Antonio Espinoza Chaves 2019083698
- Isaac Mauricio Herrera Monge 2019160455
- Juan Daniel Rodríguez Montero 2020426163
- Michael Valverde Navarro 2020044189

II Semestre 2022

### **Modelo conceptual con notación de Chen:**

[https://miro.com/app/board/uXjVPPAozk8=/?share\\_link\\_id=332147222541](https://miro.com/app/board/uXjVPPAozk8=/?share_link_id=332147222541)


### **Modelo relacional:**

[https://miro.com/app/board/uXjVPOZ\\_718=/?share\\_link\\_id=467049342205](https://miro.com/app/board/uXjVPOZ_718=/?share_link_id=467049342205)

### **Descripción de las estructuras de datos desarrolladas:**

#### **Tabla Cita:**

Para la tabla de cita, se utilizaron los siguientes atributos: cliente (int), placaVehiculo (string), sucursal(string), idLavado(int) el cual es una foreign key de Lavado, factura(int) el cual es el número de factura y es un FK proveniente de Factura. Finalmente, está numeroCita, el cual es de tipo int y es la primary key de la Tabla.

	Column Name	Data Type	Allow Nulls
	cliente	int	<input type="checkbox"/>
	placaVehiculo	varchar(50)	<input type="checkbox"/>
	sucursal	varchar(50)	<input type="checkbox"/>
	idLavado	int	<input type="checkbox"/>
	factura	int	<input type="checkbox"/>
	numeroCita	int	<input type="checkbox"/>


#### **Tabla Citas por Cliente:**

Para esta tabla se manejaron solo dos atributos, los cuales son las PK de Cita y de Cliente, de manera que se utiliza como tabla intermedia. Los atributos son numeroCita(int) y cedula\_cliente(int). Se creó para poder guardar todas las citas de un cliente, pues pueden tener más de una.

Column Name	Data Type	Allow Nulls
numeroCita	int	<input type="checkbox"/>
cedula_cliente	int	<input type="checkbox"/>


#### **Tabla Cliente:**

Para esta tabla se utilizaron los siguientes atributos. Primero la cedula(int), el cual es la llave principal, nombreCompleto(string), puntos(int), contrasena(string), correo(string), usuario(string) y dirección(string). Al no tener ningún valor multivaluado no fue necesario poner una tabla intermedia.

	Column Name	Data Type	Allow Nulls
	cedula	int	<input type="checkbox"/>
	nombreCompleto	varchar(50)	<input type="checkbox"/>
	puntos	int	<input type="checkbox"/>
	contrasena	varchar(50)	<input type="checkbox"/>
	correo	varchar(50)	<input type="checkbox"/>
	usuario	varchar(50)	<input type="checkbox"/>
	direccion	varchar(100)	<input type="checkbox"/>


#### Tabla Factura:

Para esta tabla se eligieron los siguientes atributos. El primero es ID\_Factura(int), el cuál es la llave primaria de la relación. También se tiene servicio (int), monto (int), iva(int) y cliente(int), el cual es la llave foránea proveniente de Cliente.cedula.

	Column Name	Data Type	Allow Nulls
	ID_Factura	int	<input type="checkbox"/>
	servicio	int	<input checked="" type="checkbox"/>
	monto	int	<input checked="" type="checkbox"/>
	iva	int	<input checked="" type="checkbox"/>
	cliente	int	<input checked="" type="checkbox"/>


#### Tabla Lavado:

Para esta tabla se utilizan los atributos como personalRequerido (int) como foreign key, precio (int), costo (int), nombre (varchar) sin ser null, duración (int) puntuacionLealtad (int) y id\_lavado (int) que corresponde a ser primary key.

	Column Name	Data Type	Allow Nulls
	personalRequerido	int	<input checked="" type="checkbox"/>
	precio	int	<input checked="" type="checkbox"/>
	costo	int	<input checked="" type="checkbox"/>
	nombre	varchar(50)	<input type="checkbox"/>
	duracion	int	<input checked="" type="checkbox"/>
	puntuacionLealtad	int	<input checked="" type="checkbox"/>
	id_Lavado	int	<input type="checkbox"/>

#### Tabla Producto:

Se usan como atributo nombre (varchar) como primary key, marca (varchar) y costo (int).

	Column Name	Data Type	Allow Nulls
	nombre	varchar(50)	<input type="checkbox"/>
	marca	varchar(50)	<input checked="" type="checkbox"/>
	costo	int	<input checked="" type="checkbox"/>

#### Tabla Producto por Factura:

Esta tabla se compone por dos atributos que corresponden a ID\_Factura (int) que no permite valores null y que sirve como foreign key, nombre\_producto (varchar) que no permite valores null y que funciona como foreign key.

	Column Name	Data Type	Allow Nulls
	ID_Factura	int	<input type="checkbox"/>
	nombre_producto	varchar(50)	<input type="checkbox"/>

#### Tabla Producto por Proveedor:

Esta tabla tiene cedula\_proveedor (int, sin ser null) y nombre\_producto (varchar, sin ser null) ambos sirven como foreign key.

	Column Name	Data Type	Allow Nulls
	cedula_proveedor	int	<input type="checkbox"/>
	nombre_producto	varchar(50)	<input type="checkbox"/>


#### Tabla Productos por Lavado:

Como atributos utilizados se tienen nombre\_producto (varchar) no permite valores null, id\_lavado (int) sin permitir valores null, ambos atributos mencionados corresponden a foreign key.

	Column Name	Data Type	Allow Nulls
	nombre_producto	varchar(50)	<input type="checkbox"/>
	id_lavado	int	<input type="checkbox"/>


#### Tabla Proveedor:

Los atributos utilizados por la tabla proveedor son nombre (varchar), cedula\_jurídica (int) que no puede ser null y constituye la primary key de esta tabla, dirección (varchar), correo (varchar) y contacto (varchar).

	Column Name	Data Type	Allow Nulls
	nombre	varchar(50)	<input type="checkbox"/>
	cedulaJuridica	int	<input type="checkbox"/>
	direccion	varchar(50)	<input type="checkbox"/>
	correo	varchar(50)	<input type="checkbox"/>
	contacto	varchar(50)	<input type="checkbox"/>

#### Tabla Snack:

En la tabla snack se utilizan los atributos ID\_Snack (int) y que es el primary key de esta tabla, nombre (varchar) inventario (int) y precio (int).

	Column Name	Data Type	Allow Nulls
	ID_Snack	int	<input type="checkbox"/>
	nombre	varchar(50)	<input type="checkbox"/>
	inventario	int	<input type="checkbox"/>
	precio	int	<input type="checkbox"/>

#### Tabla Snacks por Factura:

Se contemplan dos atributos para esta tabla los cuales son snack\_id (int) sin ser null y IDFactura (int) sin ser null, ambos constituyen ser foreign keys.

	Column Name	Data Type	Allow Nulls
	snack_id	int	<input type="checkbox"/>
	IDFactura	int	<input type="checkbox"/>

Tabla Sucursal: Contiene dos atributos, uno cliente(varchar) que es foreign key y el nombre de la sucursal nombre\_sucursal(varchar), fechaApertura(date), fechaInicioGerente(date) Que referencia a cuándo inició el gerente a regir en la sucursal, gerente(int) que corresponde a la cedula del

trabajador que gerencia en la sucursal, teléfono(int) que es el teléfono de sucursal y la ubicación; provincia(varchar), cantón(varchar) y distrito(varchar).


	Column Name	Data Type	Allow Nulls
	nombre	varchar(50)	<input type="checkbox"/>
	fechaInicioGerente	date	<input type="checkbox"/>
	fechaApertura	date	<input type="checkbox"/>
	gerente	int	<input type="checkbox"/>
	telefono	int	<input type="checkbox"/>
	provincia	varchar(50)	<input type="checkbox"/>
	canton	varchar(50)	<input type="checkbox"/>
	distrito	varchar(50)	<input type="checkbox"/>

Tabla Teléfonos por Cliente: Los atributos utilizados en la tabla telefonos por cliente son cliente(varchar) y telefono(int), es la tabla utilizada para almacenar varios teléfonos y relacionarlo a un cliente.

	Column Name	Data Type	Allow Nulls
	cliente	int	<input type="checkbox"/>
	telefono	int	<input type="checkbox"/>

Tabla Trabajador: Contiene los siguientes atributos: cedula(int), nombre(varchar), apellidos(varchar), contrasena(varchar), edad(int), fechaNacimiento(date), fechaIngreso(date), tipoPago(varchar), rol(varchar), isGerente(int), donde la cedula es primary key, pues es el identificador que lo separa.


	Column Name	Data Type	Allow Nulls
	cedula	int	<input type="checkbox"/>
	nombre	varchar(50)	<input type="checkbox"/>
	apellidos	varchar(50)	<input type="checkbox"/>
	contrasena	varchar(50)	<input type="checkbox"/>
	edad	int	<input type="checkbox"/>
	fechaNacimiento	date	<input type="checkbox"/>
	fechaIngreso	date	<input type="checkbox"/>
	tipoPago	varchar(50)	<input type="checkbox"/>
	rol	varchar(50)	<input type="checkbox"/>
	isGerente	bit	<input type="checkbox"/>

Tabla Trabajador por Sucursal: Los atributos utilizados en la tabla trabajador por sucursal son cedula\_trabajador(int) y nombre\_sucursal(varchar), es la tabla intermedia de la relación 1-N entre la Sucursal el Trabajador, ambos atributos son foreign key.

Column Name	Data Type	Allow Nulls
cedula_trabajador	int	<input type="checkbox"/>
nombre_sucursal	varchar(50)	<input type="checkbox"/>

Se puede observar el modelo final de SQL Server en el siguiente [enlace](#).

## Problemas conocidos:

Algunas direcciones de la API no están implementadas por lo que limita la utilización del servicio REST y a la aplicación web, esto ocurre pues no se lograron completar las queries debidas para consultar a la base de datos y obtener la información deseada.

Con respecto al alojamiento del servicio Web API en IIS Express no fue posible pues se presentaron problemas de instalación con el software de IIS Hosting .Net Core, esto pues el servicio REST fue desarrollado en .NET Core y con el IDE de Rider, actualmente Rider no brinda soporte lo suficientemente satisfactorio para hacer el deployment con ISS Express,. Al investigar en la documentación oficial de Microsoft se encontró que se puede hacer deployment con el servicio IIS pero el documento applicationhost.config no contaba con los atributos correctos pues el software IIS Hosting .Net Core no fue capaz de instalarlos.

En la aplicación móvil, no se logró que la base local y la principal se actualizarán mediante sync process. Esto pues se requería de una librería que daba muchos problemas en la parte del API. Otro error por parte de la aplicación es que a veces, a la hora de ingresar citas en la tabla local, tira un error de puntero a vacío. Después de investigar, se encontró que es un problema de la librería y que una solución era cambiar el nombre de la base de datos para cada ejecución de la aplicación, sin embargo, esta solución no siempre servía. Por último, otro problema conocido es que las actualizaciones de la información personal se hacen de manera local, por lo que no pasan por la base local.

## Problemas encontrados:

Se encontró un problema en la aplicación móvil, basado en que la conexión entre las interfaces y el API se hacían mediante localhost. Sin embargo, a la hora de probar la conexión se caía múltiples veces. Se encontró que esto ocurría porque el emulador de Android o un celular pegado a la computadora tiene un localhost diferente al de la computadora. Por lo que impedía que se conectara debido a una incongruencia de la dirección del “endpoint”. Para esto, se utilizó un programa llamado “ngrok”, el cual toma el localhost y un puerto y lo aloja de manera temporal en el internet, ofreciendo una IP nueva equivalente al localhost de la computadora, dejando así que se pudiera conectar el celular al localhost de la computadora. Este programa se puede encontrar en el siguiente [enlace](#).

Al realizar la solicitud POST a la base de datos, se encontraba la tabla y la columna respectiva, sin embargo, no se podía referenciar con el nombre asociado.

```
Exception data:
Severity: ERROR
SqlState: 42703
MessageText: no existe la columna «no_promocion»
Hint: Hay una columna llamada «no_promocion» en la tabla «promocion», pero no puede ser referenciada desde esta parte de la consulta.
Position: 177
File: parse_relation.c
Line: 3599
Routine: errorMissingColumn
```

El problema se presentó por un error en el INSERT, cuando se insertaban los valores, el nombre de que se pasaba de parámetro en VALUES era distinto al nombre de los atributos del modelo

Fue solucionado utilizando el nombre de los atributos del modelo

```
export interface SingleAppointmentI {
  cliente: number,
  placaVehiculo: string,
  sucursal: string,
  tipoLavado: string,
  responsable: number,
  factura: string,
  numeroCita: string
}
```

Problema al ejecutar las solicitudes

Cuando se ejecutaron POST o GETS iniciales se presentó el siguiente error:

```
Error: response status is 500
Response body
System.InvalidOperationException: Unable to resolve service for type 'TECAir_API.Database.Interface.IReservacion' while attempting to activate 'TECAir_API.Controllers.ReservacionController'.
at Microsoft.Extensions.DependencyInjection.ActivatorUtilities.GetService(IServiceProvider sp, Type type, Type requiredBy, Boolean isDefaultParameterRequired)
at lambda_method(Closure , IServiceProvider , Object[] )
at Microsoft.AspNetCore.Mvc.Controllers.ControllerActivatorProvider.<CreateActivator>_b__0(ControllerContext controllerContext)
at Microsoft.AspNetCore.Mvc.Controllers.ControllerFactoryProvider.<CreateControllerFactory>_b__0(ControllerContext controllerContext)
at Microsoft.AspNetCore.Mvc.Infrastructure.ControllerActionInvoker.Next(State& next, Scope& scope, Object& state, Boolean& isCompleted)
at Microsoft.AspNetCore.Mvc.Infrastructure.ControllerActionInvoker.InvokeInnerFilterAsync()
--- End of stack trace from previous location ---
```

La operación no era válida porque no se había agregado el archivo de interfaz a los servicios en el archivo Program.cs, el problema se solucionó agregando los archivos de interfaz para cada solicitud con AddScope en Program.cs.



## **Conclusiones**

La arquitectura es de un nivel superior, permite resolver el problema planteado y cumplir los objetivos establecidos en el proyecto, implementando servicios de aplicación Web con tecnologías actuales como Angular, conexión por medio de un REST API desarrollado como un WEB API en ASP.net core, bases de datos de gran calidad y fáciles de utilizar como SQL Server. Desarrollo de aplicaciones móviles en Xamarin, uno de los sistemas más flexibles para realizar desarrollo móvil.

De acuerdo con lo anterior, cabe mencionar que esta arquitectura y tecnologías son muy utilizadas en la industria de desarrollo de software tanto a nivel nacional como internacional, por lo que se adquirieron habilidades técnicas que se podrán utilizar en el ámbito profesional en el futuro.

## **Recomendaciones**

- Mantener todas las branches del repositorio actualizadas para evitar conflictos de los archivos.
- Evitar subir archivos que generan los IDEs que contienen rutas locales de la computadora de cada desarrollador para evitar conflictos en los archivos.
- Crear las tablas de la base de datos en orden en el script de base de datos puede evitar errores de creación, ya que la base de datos no puede crear primero una tabla que posee una llave foránea a otra tabla que se crea después en el script.
- Es importante consultar la documentación de los lenguajes y tecnologías utilizadas en el desarrollo, ya que entre una versión y otra puede haber cambios en la estructura y forma de estos. Por ejemplo, en el presente proyecto, .NET 6 ya no cuenta con el archivo Startup.cs sino que cambió por Program.cs y cambió la estructura de este archivo y la forma de añadir algunos elementos como los servicios.

## **Bibliografía**

ASP.NET Core Dependency Injection error: Unable to resolve service for type while attempting to activate. (2016, 30 noviembre). Stack Overflow. <https://stackoverflow.com/questions/40900414/asp-net-core-dependency-injection-error-unable-to-resolve-service-for-type-whil>

Calingasan, A. (2021, 13 abril). .Net 5 Web API With Repository Pattern Docker and PostgreSQL. Alexcodetuts. <https://alexcodetuts.com/2021/04/10/net-5-web-api-with-repository-pattern-docker-and-postgresql/>

DB Browser for SQLite. (2021, 25 julio). DB4S. <https://sqlitebrowser.org>

FlightConnections. (2022, 25 abril). Vuelos directos desde San José (SJO). <https://www.flightconnections.com/es/vuelos-desde-san-jos%C3%A9-sjo>

God, P. [Patrick God]. (2021, 23 noviembre). CRUD with a .NET 6 Web API & Entity Framework Core Full Course [Vídeo]. YouTube. [https://www.youtube.com/watch?v=Fbf\\_ua2t6v4](https://www.youtube.com/watch?v=Fbf_ua2t6v4)

Munonye, K. (2021, 5 junio). How to Create REST API in .Net Using C# and Visual Studio. Kindson The Genius. <https://www.kindsonthegenius.com/how-to-create-rest-api-in-net-using-c-and-visual-studio/>

Munonye, K. [Kindson The Tech Pro]. (2021, 9 junio). Build your First REST API in Net with C# in Visual Studio - Step by Step [Vídeo]. YouTube. <https://www.youtube.com/watch?v=SDHlLmkdd3s>

Robles, M. [Códigos de Programación]. (2021, 10 marzo). Crear base de datos en Android Studio (SQLite) [Vídeo]. YouTube. <https://www.youtube.com/watch?v=iWQIXjQ8ucA>