Contenido

Planteamiento del problema

Nos han contratado para elaborar una API basada en REST para un establecimiento de renta de vehículos.

Debemos gestionar información sobre los carros de los que dispone el establecimiento para rentar, los clientes, los contratos de renta, y además permitir registrar rentas y retornos de los vehículos.

Requerimientos de datos

Los requerimientos de datos son los siguientes:

Carro

Atributo	**Descripción**	**Ejemplo**	**Observaciones**
Marca	Marca del carro	Renault	
Modelo	Modelo del carro	Captur	
Año	Año del carro	2017	
Color	Color del carro	Azul Celeste	
Tipo	Tipo del carro	Camioneta	Los tipos disponibles son: • Sedan • Hatchback • SUV • Camioneta
Placa	Placa del carro	ABC123	Compuesta de tres letras y tres números
Cantidad de pasajeros	Cantidad de pasajeros máxima	5	
Kilometraje	Kilometraje del carro	1350	
Aire acondicionado	Indica si el vehículo cuenta o no con aire acondicionado	Si	
Valor por día (en USD)	Valor de la renta diaria	30	
Contrato	Número del contrato asociado actualmente al carro si este se encuentra rentado	2345	Numero de 4 dígitos. No puede iniciar con 0.

Cliente

Atributo	**Descripción**	**Ejemplo**	**Observaciones**
Nombre	Primer nombre del cliente	Juan	
Apellido	Prime apellido del cliente	González	
Fecha de nacimiento		1985-07-12	

Número de identificación		173971089	Numero de 9 dígitos. No puede iniciar con 0.
Fecha de vencimiento del documento de identificación		2025-03-15	
Número telefónico	Numero de contacto del cliente	(605) 456-10-90	El formato debe ser (NNN) NNN-NN-NN

Contrato

Los Contratos vinculan a una **persona** con un **carro**.

Atributo	**Descripción**	**Ejemplo**	**Observaciones**
Número de contrato	Número del contrato de renta	2345	Numero de 4 dígitos. No puede iniciar con 0.
Fecha de inicio	Fecha de renta del carro	2023-02-02	
Duración	Días pactados en el contrato	15	
Valor del alquiler por día (en USD)	Valor del carro por día al momento del registro del contrato	30	
Valor total del contrato (en USD)	Valor total que debe cancelar el cliente	450	Es igual al valor del vehículo por el número de días del contrato
Valor cancelado (en USD)	Valor que ha pagado el cliente hasta los momentos	315	
Valor restante (en USD)	Valor que aún tiene que pagar el cliente al momento de entregar el carro	135	
Días en mora	Número de días transcurridos desde el día en el cual el cliente ha debido entregar el carro y hoy	3	
Saldo en mora	Valor extra que el cliente debe cancelar por los días de mora	180	Se calcula como el número de días de mora multiplicado por 2 veces el valor de renta diaria del carro.

Saldo total	Suma de valor restante más el saldo en mora.	315	

Además, de los conceptos inherentes al problema, necesitaremos también almacenar información de autenticación de los usuarios que harán uso de la API.

Atributo	**Descripción**	**Ejemplo**	**Observaciones**
Login	Nombre de usuario de la persona utilizando la API	mdiaz	
Password	Contraseña de la persona utilizando la API	Bcg987k	
Fecha de expiración	Fecha más allá de la cual las credenciales ya no son válidas	2023-03-02	

Diseño general de la solución

Base de datos

Primeramente, construiremos una base de datos en MySQL para almacenar la información.

Como mínimo necesitaremos tablas para:

- Carros
- Clientes
- Contratos
- Logins

¡Importante!:

1. Es posible (y muy probable) que necesitemos tablas adicionales para modelar las relaciones entre nuestras entidades.
2. Para efectos de pruebas, al momento de arrancar nuestra aplicación ya debemos contar con datos en todas las tablas.
3. En el caso de la tabla de logins, debemos tener al menos un login activo y otro inactivo (expirado) para pruebas.

Especificación de la API

Gestión de carros

Crear Carro

Descripción	Crea un nuevo carro
Método	POST
Path	/carro

Parámetros:

Nombre	Localización	Obligatorio	Tipo de dato	Descripción	Ejemplo
login	Header	Si	String	Login para autenticarse	gdiaz
pass	Header	Si	String	Password para autenticarse	12345
marca	Body	Si	String	Se debe validar que sea alguna de las marcas disponibles. Ver endpoint de marcas.	Renault
modelo	Body	Si	String		Captur
año	Body	Si	String		2017
color	Body	Si	String		Azul Celeste
tipo	Body	Si	String	Se debe validar que sea alguno de los tipos disponibles: • Sedan • Hatchback • SUV • Camioneta Ver endpoint de tipos de carro. El usuario puede ingresar el valor utilizando cualquier combinación de mayúsculas y minúsculas, pero en la base de datos almacenaremos los valores como se muestran arriba.	Camioneta
placa	Body	Si	String	Se debe validar que esté compuesta de tres letras seguidas de tres números. El usuario puede ingresar las letras en mayúsculas o minúsculas, pero en la base de datos las almacenaremos en mayúsculas. Se debe validar también que la placa no exista previamente en el sistema.	ABC123

cantidadDePasajeros	Body	Si	String	Debe ser mayor a 1.	5
kilometraje	Body	Si	String	Debe ser mayor o igual a 0.	1350
aire	Body	Si	String	Valores disponibles: • Si • No En mayúsculas o minúsculas. Se debe validar.	Si
rentaDiaria	Body	Si	Numérico	Debe ser mayor a 0.	30

Cuerpo de la respuesta: El carro creado.

Códigos de respuesta:

Código de respuesta	**Descripción**
201 Created	Carro creado.
401 Unauthorized	Usuario y/o password erróneos o credenciales expiradas.
400 Bad request	Parámetros faltantes o erróneos. El mensaje debe ser apropiado. Ejm.: “Falta el parámetro modelo” o “El tipo no es válido. Los valores aceptados son: Sedan, Hatchback, SUV y Camioneta” o “La placa ABC123 ya existe”

Actualizar carro

Descripción	Actualiza un carro ya existente en el sistema. Solo actualiza los valores enviados.
Método	PATCH
Path	/carro/{placa}

Parámetros:

Nombre	**Localización**	**Obligatorio**	**Tipo de dato**	**Descripción**	**Ejemplo**
login	Header	Si	String	Login para autenticarse	gdiaz
pass	Header	Si	String	Password para autenticarse	12345
placa	Path	Si	String	Se debe validar que esté compuesta de tres letras seguidas de tres números. El usuario puede ingresar las letras en mayúsculas o minúsculas.	ABC123

marca	Body	No	String	Se debe validar que sea alguna de las marcas disponibles. Ver endpoint de marcas.	Renault
modelo	Body	No	String		Captur
año	Body	No	String		2017
color	Body	No	String		Azul Celeste
tipo	Body	No	String	Se debe validar que sea alguno de los tipos disponibles: • Sedan • Hatchback • SUV • Camioneta Ver endpoint de tipos de carro. El usuario puede ingresar el valor utilizando cualquier combinación de mayúsculas y minúsculas, pero en la base de datos almacenaremos los valores como se muestran arriba.	Camioneta
placa	Body	No	String	Se debe validar que esté compuesta de tres letras seguidas de tres números. El usuario puede ingresar las letras en mayúsculas o minúsculas, pero en la base de datos las almacenaremos en mayúsculas. Se debe validar también que la placa no exista previamente en el sistema.	ABC123
cantidadDePasajeros	Body	No	String	Debe ser mayor a 1.	5
kilometraje	Body	No	String	Debe ser mayor o igual al kilometraje anterior.	1350
aire	Body	No	String	Valores disponibles: • Si	Si

				• No En mayúsculas o minúsculas. Se debe validar.	
rentaDiaria	Body	No	Numérico	Debe ser mayor a 0.	30

Cuerpo de la respuesta: El carro actualizado.

Códigos de respuesta:

Código de respuesta	**Descripción**
200 OK	Carro actualizado.
401 Unauthorized	Usuario y/o password erróneos o credenciales expiradas.
404 Not found	Si el carro no existe.
400 Bad request	Parámetros faltantes o erróneos. El mensaje debe ser apropiado. Ejm.: "Falta el parámetro modelo" o "El tipo no es válido. Los valores aceptados son: Sedan, Hatchback, SUV y Camioneta" o "La placa ABC123 ya existe"

Buscar un carro

Descripción	Obtiene la información de un carro. Si el carro actualmente se encuentra rentado debe incluir un campo adicional llamado *contrato* que indique el número del contrato de renta.
Método	GET
Path	/carro/{placa}

Parámetros:

Nombre	**Localización**	**Obligatorio**	**Tipo de dato**	**Descripción**	**Ejemplo**
login	Header	Si	String	Login para autenticarse	gdiaz
pass	Header	Si	String	Password para autenticarse	12345
placa	Path	Si	String	Se debe validar que esté compuesta de tres letras seguidas de tres números. El usuario puede ingresar las letras en mayúsculas o minúsculas.	ABC123

Cuerpo de la respuesta: El carro buscado, si existe.

Códigos de respuesta:

Código de respuesta	Descripción
200 OK	Si el carro existe.
401 Unauthorized	Usuario y/o password erróneos o credenciales expiradas.
404 Not found	Si el carro no existe.
400 Bad request	Parámetros faltantes o erróneos. El mensaje debe ser apropiado. Ejm.: "El formato de la placa es errado"

Buscar varios carros

Descripción	Obtiene la información de varios carros. Si un carro actualmente se encuentra rentado debe incluir un campo adicional llamado *contrato* que indique el número del contrato de renta.
Método	GET
Path	/carro

Parámetros:

Nombre	Localización	Obligatorio	Tipo de dato	Descripción	Ejemplo
login	Header	Si	String	Login para autenticarse	gdiaz
pass	Header	Si	String	Password para autenticarse	12345
tipo	QueryString	No	String	Se debe validar que sea alguno de los tipos disponibles: • Sedan • Hatchback • SUV • Camioneta Ver endpoint de tipos de carro. El usuario puede ingresar el valor utilizando cualquier combinación de mayúsculas y minúsculas, pero en la base de datos almacenaremos los valores como se muestran arriba.	Camioneta
cantidadDePasajeros	QueryString	No	String	Cantidad mínima de pasajeros deseada.	2

aire	QueryString	No	String	Valores disponibles: • Si • No En mayúsculas o minúsculas. Se debe validar.	Si
rentaDiaria	QueryString	No	Numérico	Renta mínima deseada.	30
soloDisponibles	QueryString	No	String	Valores disponibles: • Si • No En mayúsculas o minúsculas. Se debe validar. Si este parámetro no viene en el QueryString, por defecto su valor será Si.	No

Cuerpo de la respuesta: Lista de carros obtenidos.

Códigos de respuesta:

Código de respuesta	**Descripción**
200 OK	Si todo fue bien.
401 Unauthorized	Usuario y/o password erróneos o credenciales expiradas.
404 Not found	Si ningún carro cumple con los criterios de búsqueda.
400 Bad request	Parámetros faltantes o erróneos. El mensaje debe ser apropiado. Ejm.: "El valor del parámetro soloDisponibles no es válido. Las opciones son Si o No."

Eliminar un carro

Descripción	Elimina un carro. Solo se pueden eliminar carros que actualmente no tengan y que nunca hayan tenido un contrato asociado.
Método	DELETE
Path	/carro/{placa}

Parámetros:

Nombre	**Localización**	**Obligatorio**	**Tipo de dato**	**Descripción**	**Ejemplo**
login	Header	Si	String	Login para autenticarse	gdiaz
pass	Header	Si	String	Password para autenticarse	12345

placa	Path	Si	String	Se debe validar que esté compuesta de tres letras seguidas de tres números. El usuario puede ingresar las letras en mayúsculas o minúsculas.	ABC123

Cuerpo de la respuesta: Un mensaje indicando que el carro fue eliminado.

Códigos de respuesta:

Código de respuesta	**Descripción**
200 OK	Si el carro fue eliminado.
401 Unauthorized	Usuario y/o password erróneos o credenciales expiradas.
404 Not found	Si el carro no existe.
400 Bad request	Parámetros faltantes o erróneos. El mensaje debe ser apropiado. Ejm.: "El formato de la placa es errado"

Gestión de clientes

Crear Cliente

Descripción	Crea un nuevo cliente
Método	POST
Path	/cliente

Parámetros:

Nombre	**Localización**	**Obligatorio**	**Tipo de dato**	**Descripción**	**Ejemplo**
login	Header	Si	String	Login para autenticarse	gdiaz
pass	Header	Si	String	Password para autenticarse	12345
nombre	Body	Si	String		Juan
apellido	Body	Si	String		González
fechaDeNacimiento	Body	Si	String	Se debe validar el formato. Se debe validar que el cliente tenga más de 18 años, de lo contrario se debe rechazar.	1985-07-12
id	Body	Si	Numérico	Se debe validar que sea un valor	137490456

				numérico de 9 dígitos. No puede comenzar con 0.	
expiracionId	Body	Si	String	Se debe validar el formato. Se debe validar que la fecha sea posterior a la fecha actual. De lo contrario se debe rechazar.	2025-01-15
telefono	Body	Si	String	El formato debe ser (NNN) NNN-NN-NN. Se debe validar.	(605) 342-23-12

Cuerpo de la respuesta: El cliente creado.

Códigos de respuesta:

Código de respuesta	**Descripción**
201 Created	Cliente creado.
401 Unauthorized	Usuario y/o password erróneos o credenciales expiradas.
400 Bad request	Parámetros faltantes o erróneos. El mensaje debe ser apropiado. Ejm.: “Falta el parámetro nombre” o “El formato de fecha no es válido. El formato esperado es YYYY-MM-DD” o “El id 137490456 ya existe”

Actualizar cliente

Descripción	Actualiza un cliente ya existente en el sistema. Solo actualiza los valores enviados.
Método	PATCH
Path	/cliente/{id}

Parámetros:

Nombre	**Localización**	**Obligatorio**	**Tipo de dato**	**Descripción**	**Ejemplo**
login	Header	Si	String	Login para autenticarse	gdiaz
pass	Header	Si	String	Password para autenticarse	12345
id	Path	Si	Numérico	Se debe validar que sea un valor numérico de 9 dígitos.	137490456

				No puede comenzar con 0.	
nombre	Body	No	String		Juan
apellido	Body	No	String		González
fechaDeNacimiento	Body	No	String	Se debe validar el formato. Se debe validar que el cliente tenga más de 18 años, de lo contrario se debe rechazar.	1985-07-12
expiracionId	Body	No	String	Se debe validar el formato. Se debe validar que la fecha sea posterior a la fecha actual. De lo contrario se debe rechazar.	2025-01-15
telefono	Body	No	String	El formato debe ser (NNN) NNN-NN-NN. Se debe validar.	(605) 342-23-12

Cuerpo de la respuesta: El cliente actualizado.

Códigos de respuesta:

Código de respuesta	**Descripción**
200 OK	Cliente actualizado.
401 Unauthorized	Usuario y/o password erróneos o credenciales expiradas.
404 Not found	Si el cliente no existe.
400 Bad request	Parámetros faltantes o erróneos. El mensaje debe ser apropiado. Ejm.: “El formato de fecha no es válido. El formato esperado es YYYY-MM-DD” o “El documento de identidad ya ha expirado” o “El cliente es menor de 18 años”

Buscar un cliente

Descripción	Obtiene la información de un cliente.
Método	GET
Path	/cliente/{id}

Nombre	**Localización**	**Obligatorio**	**Tipo de dato**	**Descripción**	**Ejemplo**
login	Header	Si	String	Login para autenticarse	gdiaz

pass	Header	Si	String	Password para autenticarse	12345
id	Path	Si	Numérico	Se debe validar que sea un valor numérico de 9 dígitos. No puede comenzar con 0.	137490456

Cuerpo de la respuesta: El cliente buscado.

Códigos de respuesta:

Código de respuesta	Descripción
200 OK	Si el cliente existe.
401 Unauthorized	Usuario y/o password erróneos o credenciales expiradas.
404 Not found	Si el cliente no existe.
400 Bad request	Parámetros faltantes o erróneos. El mensaje debe ser apropiado. Ejm.: "El formato de id no es válido"

Listar todos los clientes

Descripción	Obtiene la información de todos los clientes.
Método	GET
Path	/cliente

Parámetros:

Nombre	Localización	Obligatorio	Tipo de dato	Descripción	Ejemplo
login	Header	Si	String	Login para autenticarse	gdiaz
pass	Header	Si	String	Password para autenticarse	12345

Cuerpo de la respuesta: Lista de clientes obtenidos.

Códigos de respuesta:

Código de respuesta	Descripción
200 OK	Si todo fue bien.
401 Unauthorized	Usuario y/o password erróneos o credenciales expiradas.
404 Not found	Si no existe ningún cliente.
400 Bad request	Parámetros faltantes o erróneos. El mensaje debe ser apropiado. Ejm.: "Falta el header login"

Eliminar un cliente

Descripción	Elimina un cliente. Solo se pueden eliminar clientes que actualmente no tengan y que nunca hayan tenido un contrato asociado.
Método	DELETE
Path	/cliente/{id}

Parámetros:

Nombre	**Localización**	**Obligatorio**	**Tipo de dato**	**Descripción**	**Ejemplo**
login	Header	Si	String	Login para autenticarse	gdiaz
pass	Header	Si	String	Password para autenticarse	12345
id	Path	Si	Numérico	Se debe validar que sea un valor numérico de 9 dígitos. No puede comenzar con 0.	137490456

Cuerpo de la respuesta: Un mensaje indicando que el cliente fue eliminado.

Códigos de respuesta:

Código de respuesta	**Descripción**
200 OK	Si el cliente fue eliminado.
401 Unauthorized	Usuario y/o password erróneos o credenciales expiradas.
404 Not found	Si el cliente no existe.
400 Bad request	Parámetros faltantes o erróneos. El mensaje debe ser apropiado. Ejm.: “El formato de id no es válido”

Gestión de contratos

Rentar un carro (crear contrato)

Descripción	Crea un contrato de renta. Este endpoint además debe marcar de alguna forma el carro en el sistema para que no se encuentre disponible para renta hasta que el carro sea devuelto.
Método	POST
Path	/contrato

Parámetros:

Nombre	**Localización**	**Obligatorio**	**Tipo de dato**	**Descripción**	**Ejemplo**
login	Header	Si	String	Login para autenticarse	gdiaz

pass	Header	Si	String	Password para autenticarse	12345
placa	Body	Si	String	Se debe validar que esté compuesta de tres letras seguidas de tres números. El usuario puede ingresar las letras en mayúsculas o minúsculas. Se debe validar también que la placa exista en el sistema.	ABC123
clienteId	Path	Si	Numérico	Se debe validar que sea un valor numérico de 9 dígitos. No puede comenzar con 0. Se debe validar que el cliente exista.	137490456
fechaDeInicio	Body	Si	String	Se debe validar el formato.	2025-01-15
dias	Body	Si	Numérico	Se debe validar que sea mayor a 1.	
valorCancelado	Body	Si	Numérico	Se debe validar que sea mayor o igual al 75% del valor del contrato.	

Cuerpo de la respuesta: Información del contrato creado. Debe contener la siguiente información:

- Numero de contrato
- Placa del carro
- Id del cliente
- Fecha de inicio
- Duración
- Valor del alquiler por día
- Valor total del contrato
- Valor cancelado
- Valor restante
- Días en mora
- Saldo en mora
- Saldo total

Códigos de respuesta:

Código de respuesta	Descripción
201 Created	Contrato creado.
401 Unauthorized	Usuario y/o password erróneos o credenciales expiradas.
404 Not found	Si no existe el cliente. Si no existe el carro.
400 Bad request	Parámetros faltantes o erróneos. El mensaje debe ser apropiado. Ejm.: "El valor cancelado debe ser mayor o igual al 75% del valor total del contrato. Valor total del contrato: 450 $. 75%: 337.5 $"

Buscar un contrato

Descripción	Obtiene la información de un contrato. **¡Importante!:** A diferencia del endpoint anterior que solo obtiene la placa del carro y el id del cliente, este debe contener el campo *carro* cuyo valor será todo el objeto carro y el campo *cliente* cuyo valor será todo el objeto cliente.
Método	GET
Path	/contrato/{numero}

Nombre	Localización	Obligatorio	Tipo de dato	Descripción	Ejemplo
login	Header	Si	String	Login para autenticarse	gdiaz
pass	Header	Si	String	Password para autenticarse	12345
numero	Path	Si	Numérico	Se debe validar que sea un valor numérico de 4 dígitos. No puede comenzar con 0.	3426

Cuerpo de la respuesta: El contrato buscado incluyendo toda la información del cliente y del carro.

Códigos de respuesta:

Código de respuesta	Descripción
200 OK	Si el contrato existe.
401 Unauthorized	Usuario y/o password erróneos o credenciales expiradas.
404 Not found	Si el contrato no existe.
400 Bad request	Parámetros faltantes o erróneos. El mensaje debe ser apropiado. Ejm.: "El formato del número de contrato no es válido"

Listar todos los contratos

Descripción	Obtiene la información de todos los contratos.

	Para cada contrato obtiene la misma información que obtenemos del endpoint **Rentar un carro**.
Método	GET
Path	/contrato

Parámetros:

Nombre	Localización	Obligatorio	Tipo de dato	Descripción	Ejemplo
login	Header	Si	String	Login para autenticarse	gdiaz
pass	Header	Si	String	Password para autenticarse	12345

Cuerpo de la respuesta: Lista de contratos obtenidos.

Códigos de respuesta:

Código de respuesta	Descripción
200 OK	Si todo fue bien.
401 Unauthorized	Usuario y/o password erróneos o credenciales expiradas.
404 Not found	Si no existe ningún contrato con el número indicado.
400 Bad request	Parámetros faltantes o erróneos. El mensaje debe ser apropiado. Ejm.: "Falta el header login"

Devolver un carro (cerrar un contrato)

Descripción	Cierra un contrato de renta. Este endpoint además debe marcar de alguna forma el carro en el sistema para que vuelva a estar disponible para renta.
Método	PATCH
Path	/contrato/{numero}

Parámetros:

Nombre	Localización	Obligatorio	Tipo de dato	Descripción	Ejemplo
login	Header	Si	String	Login para autenticarse	gdiaz
pass	Header	Si	String	Password para autenticarse	12345
numero	Path	Si	Numérico	Se debe validar que sea un valor numérico de 4 dígitos. No puede comenzar con 0.	3426

valorCancelado	Body	Si	Numérico	Se debe validar que sea el monto exacto del saldo total.	

Cuerpo de la respuesta: Un mensaje indicando que el contrato fue cerrado y el saldo final del contrato (que debe ser cero).

Códigos de respuesta:

Código de respuesta	Descripción
200 OK	Contrato cerrado.
401 Unauthorized	Usuario y/o password erróneos o credenciales expiradas.
404 Not found	Si no existe el contrato.
400 Bad request	Parámetros faltantes o erróneos. El mensaje debe ser apropiado. Ejm.: "El valor cancelado debe ser igual al saldo total que es 315 $"

Endpoints Auxiliares

Listar los tipos de carro

Descripción	Obtiene una lista de todos los valores que se pueden usar como tipo de carro
Método	GET
Path	/tipoDeCarro

Nombre	Localización	Obligatorio	Tipo de dato	Descripción	Ejemplo
login	Header	Si	String	Login para autenticarse	gdiaz
pass	Header	Si	String	Password para autenticarse	12345

Cuerpo de la respuesta: Lista de todos los tipos de carro.

Códigos de respuesta:

Código de respuesta	Descripción
200 OK	Si todo va bien
401 Unauthorized	Usuario y/o password erróneos o credenciales expiradas.
400 Bad request	Parámetros faltantes o erróneos. El mensaje debe ser apropiado. Ejm.: "Falta el header login"

Listar las marcas de carro

Descripción	Obtiene una lista de todos los valores que se pueden usar como la marca del carro
Método	GET

Path	/marcaDeCarro

Nombre	**Localización**	**Obligatorio**	**Tipo de dato**	**Descripción**	**Ejemplo**
login	Header	Si	String	Login para autenticarse	gdiaz
pass	Header	Si	String	Password para autenticarse	12345

Cuerpo de la respuesta: Lista de todas las marcas de carro.

Códigos de respuesta:

Código de respuesta	**Descripción**
200 OK	Si todo va bien
401 Unauthorized	Usuario y/o password erróneos o credenciales expiradas.
400 Bad request	Parámetros faltantes o erróneos. El mensaje debe ser apropiado. Ejm.: “Falta el header login”

Fecha y formato de la entrega

- El proyecto tiene un tiempo máximo de dos semanas para su desarrollo.
- La fecha de entrega máxima (sin prorroga) es el viernes 17 de marzo hasta el final del día.
- La entrega se realizará a través de GitHub. Solo deben hacerme llegar el link del repositorio.
- Dentro del proyecto debe haber un script para crear y poblar la base de datos.
- Dentro del proyecto debe haber una colección de Postman con todos los requests necesarios para probar cada uno de los endpoints desarrollados.
- Durante las dos semanas planificadas para la elaboración del proyecto se atenderán consultas *en relación al enunciado* a través de los canales de comunicación previamente establecidos a lo largo de la evolución del curso.
- De ser necesaria, el viernes 10 de marzo podemos tener una sesión en la hora habitual de clase para discutir posibles dificultades técnicas que puedan tener para completar el proyecto.

Criterios de evaluación

Del resultado de la evaluación del presente proyecto, dependerá la obtención del certificado de aprobación del curso. En ese sentido, a continuación se describen los puntos clave a evaluar.

A nivel de usuario

- Es imperativo que la base de datos se cree sin errores utilizando el script provisto por Uds. dentro del proyecto.
- Es imperativo que el proyecto compile y arranque sin errores.

- Se evaluará que todos los endpoints funcionen como fueron descritos en el presente documento, incluyendo autenticación, validaciones de campos y otras reglas de negocio.

A nivel de base de datos

- Se evaluará la estructura de la base de datos: tablas, columnas y tipos de datos usados, asi como restricciones sobre los tipos de datos.
- Se evaluará el correcto uso de claves primarias y foráneas.

A nivel del código fuente

Se evaluará:

- La correcta división de la aplicación en al menos las tres capas fundamentales: controlador, servicio y persistencia.
- El correcto uso de todas las anotaciones vistas en clase y que apliquen a su solución del problema.
- El correcto manejo de las dependencias y su inyección, incluyendo la aplicación del principio de inversión.
- El correcto uso de loggers y logging en archivos.
- El correcto manejo de excepciones.

A nivel de versionamiento

- Se evaluará la correcta aplicación de Git-Flow para el desarrollo del proyecto.
- Se evaluará la inclusión/exclusión de los archivos versionados.