Instituto Tecnológico de Costa Rica

Tarea Corta 1

José Morales Vargas, carné 2019024270
Alejandro Soto Chacón, carné 2019008164
Ignacio Vargas Campos, carné 2019053776
José Retana Corrales, carné 2020144743

Área Académica de

Ingeniería en Computadores

Bases de Datos

(CE3101)

Profesor Marco Rivera Meneses

Semestre I 2022

Índice

Modelo Conceptual

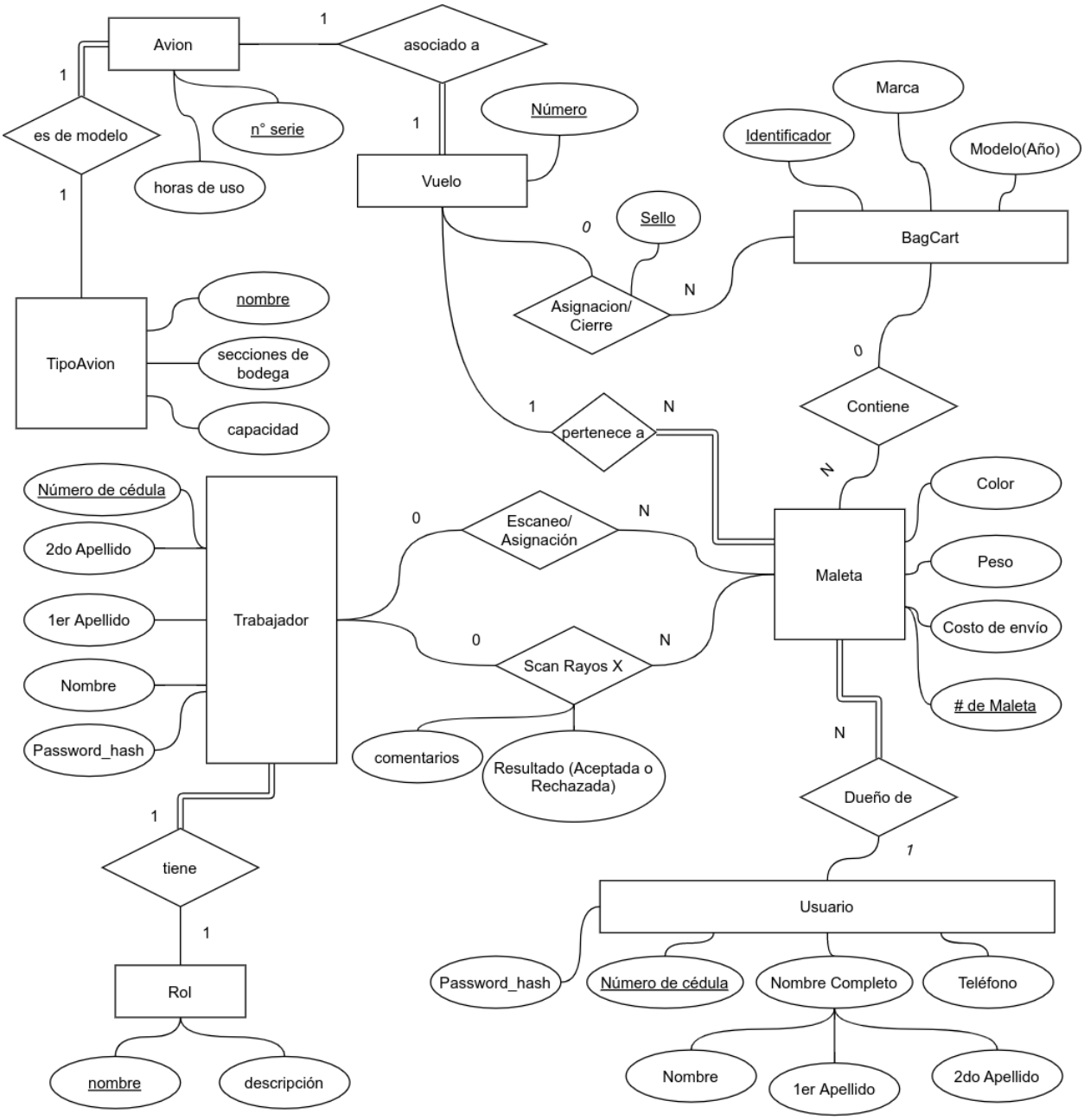

Figura 1: Diagrama Entidad Relación

Modelo Relacional

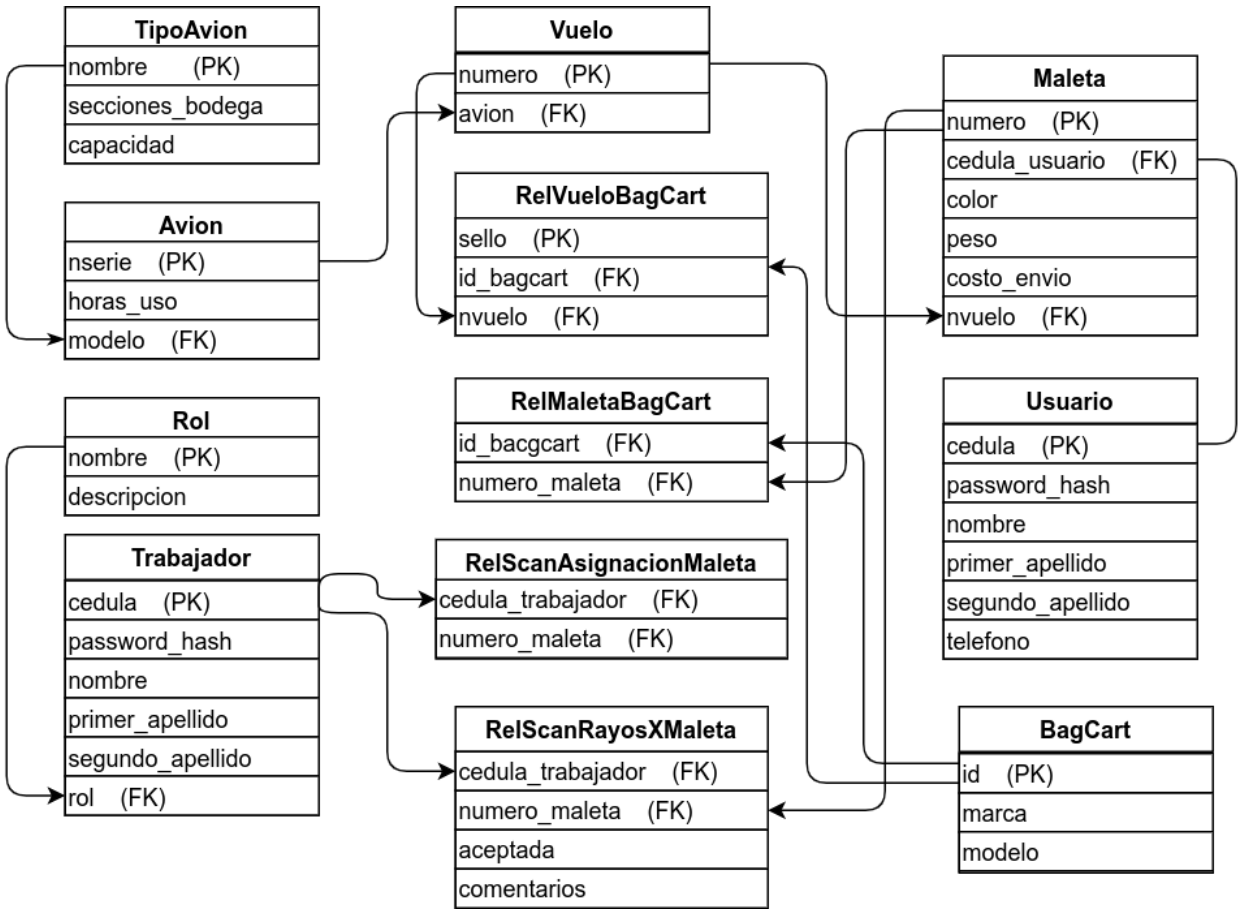

Figura 2: Diagrama Modelo Relacional

Descripción de estructuras de datos

A continuación se describen las estructuras de datos implementadas para representar a las diferentes entidades y relaciones.

`TipoAvion`

Descripción: Representa a un tipo de avion en específico.

Atributos:

- `nombre`: PK. Nombre del tipo de avion.

- `capacidad`: Indica la cantidad de maletas que puede almacenar un tipo de avión.
- `secciones de bodega`: Se refiere a los espacios destinados a las maletas del avión.

`Avion`

Descripción: Representa a un avión en específico

Atributos:

- `nserie`: PK. Número de serie del avión. Funciona como el identificador de un avión en específico.
- `horas_uso`: Indica las horas de vuelo del avión.
- `modelo`: FK. Indica el tipo de avion que es.

`Vuelo`

Descripción: Contiene los datos de un vuelo en particular.

Atributos:

- `numero`: PK. Es el identificador del vuelo el cual se asigna en la creación del mismo.
- `avion`: FK. Referencia al número de serie del avión utilizado para el vuelo.

`Rol`

Descripción: Se refiere a un tipo de rol que puede ser adoptado por uno de los trabajadores.

Atributos:

- `nombre`: PK. Nombre por el cual se conoce el rol. Este es único.
- `descripción`: Descripción de las funciones del rol.

`Trabajador`

Descripción: Representa a uno de los empleados que interactúan con el sistema.

Atributos:

- `cedula`: PK. Identificación del trabajador.

- password_hash: hash de la contraseña del usuario. Se utiliza para verificar que la autenticación del usuario de sistema sea la correcta.
- nombre: Contiene el nombre (sin apellidos) del trabajador.
- primer_apellido: Contiene el primer apellido del trabajador.
- segundo_apellido: Contiene el segundo apellido del trabajador.
- rol: FK. Contiene el nombre del rol del trabajador.

Usuario

Descripción: Representa a un usuario del servicio de aerolínea.

Atributos:

- cedula: PK. Identificación del usuario.
- password_hash: hash de la contraseña del usuario. Se utiliza para verificar que la autenticación del usuario de sistema sea la correcta.
- nombre: Contiene el nombre (sin apellidos) del usuario.
- primer_apellido: Contiene el primer apellido del usuario.
- segundo_apellido: Contiene el segundo apellido del usuario.
- telefono: Número de teléfono para contacto del usuario.

Maleta

Descripción: Contiene los datos necesarios para representar una maleta.

Atributos:

- numero: PK. Identifica la maleta. El número se asigna en el registro de la maleta.
- cedula_usuario: FK. Referencia al usuario del servicio que es dueño de la maleta.
- color: Color de la maleta que se utiliza como hint a la hora de buscar la misma.
- peso: Peso en kilos de la maleta
- costo_envío: Almacena el costo total de envío de la maleta

`BagCart`

Descripción: Contiene la información que referencia a un bagcart, el cual es el medio de transporte utilizado para las maletas desde que las mismas se registran hasta que se abordan al avión.

Atributos:

- `id`: PK. Identifica a un bagcart en específico. Se asigna al crear el bagcart.
- `marca`: Indica el vendor del bagcart.
- `modelo`: Indica el tipo de bagcart.

`RelScanRayosXMaleta`

Descripción: Representa un registro de escaneo de rayos x de una maleta, el cuál es un requisito previo a abordar la maleta a un bagcart.

Atributos:

- `cedula_trabajador`: FK. Referencia al trabajador que realizó el escaneo de la maleta.
- `numero_maleta`: FK. Referencia a la maleta escaneada.
- `aceptada`: Indica si el proceso de escaneado resultó en la maleta siendo aceptada. Si es de valor "falso", significa que la maleta ha sido rechazada.
- `comentarios`: Observaciones del trabajador respecto al proceso de escaneo.

`RelScanAsignacionMaleta`

Descripción: Representa un registro que indica que una maleta ha sido abordada dentro de un avión.

Atributos:

- `cedula_trabajador`: FK. Referencia al trabajador que escanea y asigna una maleta a un avión.
- `numero_maleta`: FK. Referencia a la maleta abordada al avión.

`RelMaletaBagCart`

Descripción: Representa un registro de asignación de una maleta a un bagcart en específico.

Atributos:

- `numero_maleta`: FK. Referencia a la maleta asignada al bagcart.
- `id_bagcart`: FK. Referencia al bagcart al que ha sido asignada la maleta.

`RelVueloBagCart`

Descripción: Registro de la asignación de un bagcart cerrado a un vuelo. A la hora de cerrar un bagcart se le asigna un sello.

Atributos:

- `id_vuelo`: FK. Referencia el número del vuelo al que se ha asignado un bagcart.
- `id_bagcart`: FK. Referencia al bagcart que ha sido asignado a un vuelo.
- `sello`: PK. Valor de sello alfanumérico que se coloca en el bagcart una vez cerrado y asignado.

Descripción de la arquitectura desarrollada

La arquitectura del sistema consiste de tres aplicaciones que interactúan por medio de una conexión de red.

En el caso del servidor, el mismo consiste de un computador con el sistema operativo Windows 10 corriendo una instancia de IIS en la cual se ejecuta una REST API. En dicho servidor, los datos se almacenan en un archivo "Data.json" el cual es el que administra el REST API. Se puede interactuar con el servidor por medio de una conexión de red direccionada a un puerto en específico - dígase por ejemplo www.misitio.com:5000.

La aplicación web y la aplicación móvil comparten una arquitectura de aplicación similar, la diferencia es el sistema que usan por debajo. En el caso de la aplicación web, se trata de una aplicación de angular que se ejecuta en el navegador, mientras que para la aplicación móvil el entorno de ejecución es el sistema operativo Android.

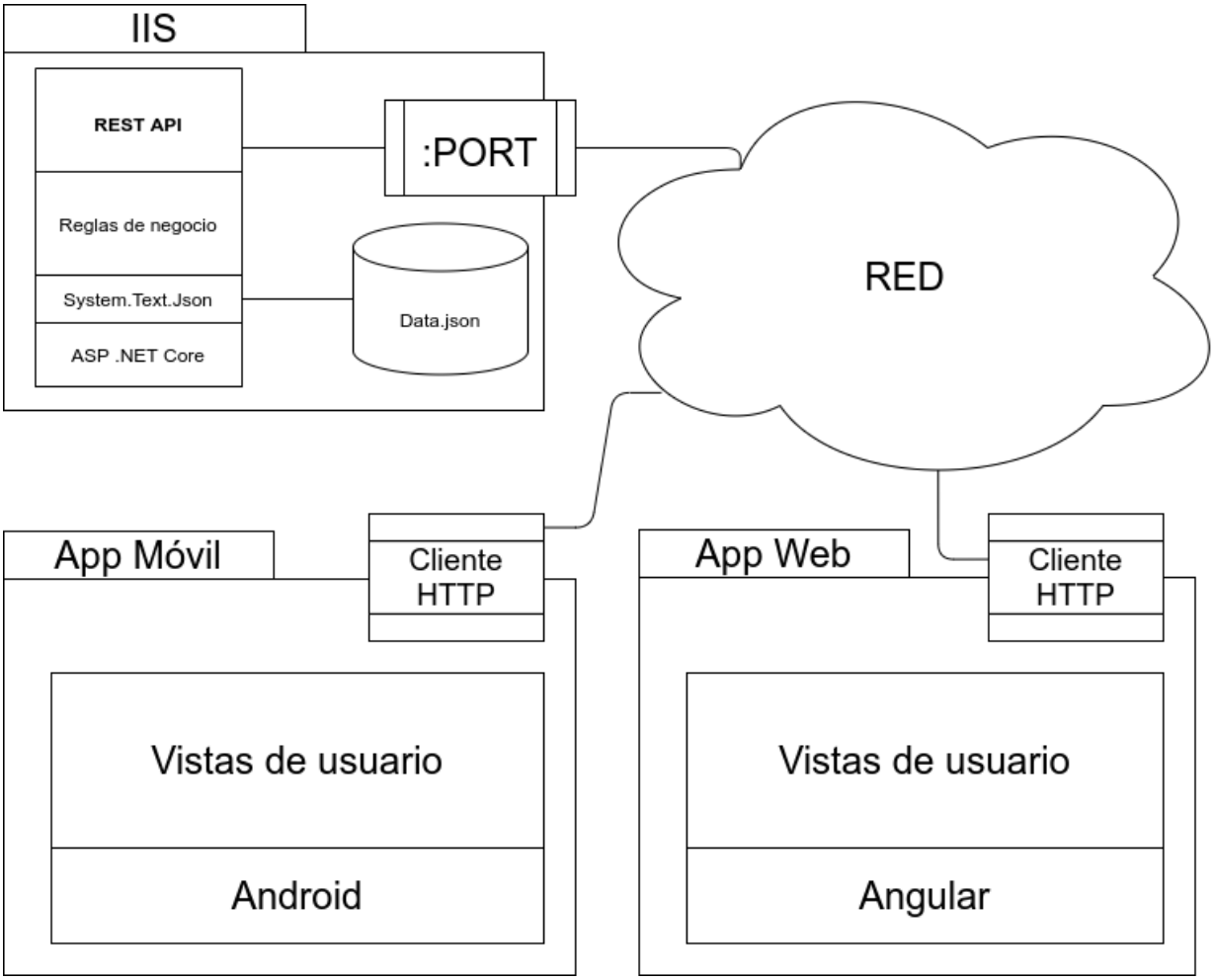

Figura 3: Diagrama de la arquitectura del sistema

Problemas conocidos

No se identifican.

Plan de proyecto

Plan de actividades

El plan de actividades se adjunta como un documento aparte en los anexos.

Minutas

Reunión No.001

Fecha:	25/02/22		**Hora de Inicio:**	19:00
Tema:	Coordinación de equipo de trabajo		**Hora de fin:**	21:00
Miembros:	José Morales	Jose Antonio Retana	Alejandro Soto	Ignacio Vargas
Objetivos alcanzados:	- Se coordinan aspectos iniciales de metodología de trabajo. - Se crea plan de trabajo. - Se dividen roles de equipo. - Se crean documentos y repositorios de colaboración grupal.			

Reunión No.002

Fecha:	3/03/22		**Hora de Inicio:**	18:00
Tema:	Chequeo y correcciones de diagramas de TC2		**Hora de fin:**	19:45
Miembros:	José Morales	Jose Antonio Retana	Alejandro Soto	Ignacio Vargas
Objetivos alcanzados:	- Se revisaron los diagramas realizados para la tarea corta 2. - Se discutieron ciertos aspectos respecto a los diagramas presentados. - Se realizaron las correcciones necesarias en los diagramas.			

Reunión No.003

Fecha:	7/03/22		**Hora de Inicio:**	18:00
Tema:	Definición de UX y UI		**Hora de fin:**	20:00
Miembros:	Ignacio Vargas	Jose Antonio Retana		
Objetivos alcanzados:	- Se definieron los aspectos estéticos y de experiencia de usuario que serán implementadas en las aplicaciones web y móvil.			

Reunión No.004

Fecha:	17/03/22		**Hora de Inicio:**	18:00
Tema:	Comprobación de funcionamiento de sistemas		**Hora de fin:**	19:45
Miembros:	José Morales	Jose Antonio Retana	Alejandro Soto	Ignacio Vargas
Objetivos alcanzados:	- Se comprobó el funcionamiento de los distintos sistemas implementados, de forma que se pudiese comprobar que está en el estado de completitud estipulado.			

Control de versiones

Para la el trabajo en equipo se hizo uso de un repositorio en GitHub el cual se encuentra en el siguiente link:

https://github.com/itcr3442/CE3101-Tabas

Conclusiones y recomendaciones

Conclusiones

- Se implementó existosamente un servicio de API/REST en el lenguaje de programación C# haciendo uso de las herramientas ofrecidas por .NET Core.

- Se creó una app móvil para el sistema operativo android en el lenguaje de programación Kotlin.
- Se desarrolló una aplicación web de varias vistas utilizando el framework de Angular.
- Para servicios REST minimalistas, el flujo de trabajo con ASP.NET Core y las herramientas de OpenApi/Swagger facilitan una implementación rápida y eficiente, sin pérdida en desempeño o en mantenibilidad.
- De querer tener un servicio web corriendo en una máquina de Windows y que el mismo sea accesible desde computadoras exteriores, IIS es una herramienta que facilita este proceso.

Recomendaciones

- Se recomiendo utilizar el template de "webapi minimal" sobre el template normal, a menos de que exista una justificación de diseño que haga de la webapi usando el esquema de controladores una mejor opción.
- Si se desea usar Windows como sistema operativo de servidor, se recomeinda utilizar las herramientas de IIS.
- Para un flujo de desarrollo más similar al estándar de la industria, se recomiendo considerar Linux como plataforma para el hosting de servicios web.
- Se recomiendo el uso de herramientas de control de versiones pues facilitan fuertemente la colaboración en grupo.

Diagramas de clases

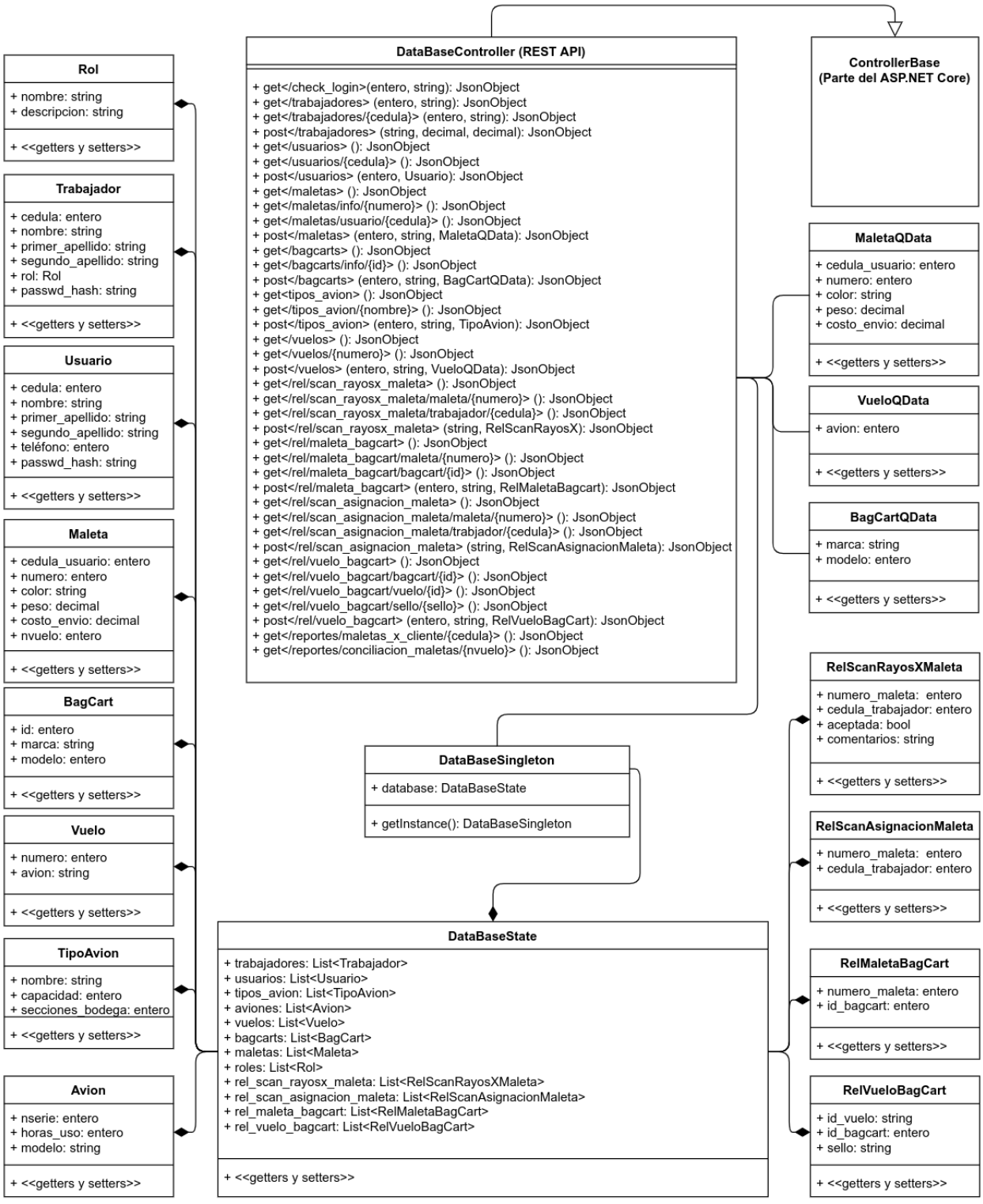

Figura 4: Diagrama de clases de la REST API.

Para la REST API el diseño es relativamente simple. La mayor parte de todas las clases señaladas son las que suelen ser descritas con el nombre de "DataClasses", es decir, clases cuyo único objetivo es ser representaciones de datos usualmente con el objetivo de facilitar el intercambio de información con otras aplicaciones. En el caso de este proyecto, estas clases son utilizadas para almacenar el estado de la base de datos y hacer su actualización relativamente simple por medio de utilidades de C# que permiten serializarlas a JSON fácilmente.

Respecto al uso de OOP, debido a la simplicidad del diseño solo era realmente necesario aplicar un patrón que permitiese regular el acceso al estado de la base de datos. Esto se hace mediante el uso de un patrón Singleton que se asegura de que solo exista un estado de base de datos en toda la aplicación, y al mismo tiempo se encarga de evitar condiciones de carrera entre los que necesitan acceder a dicha base de datos.

Para llegar al diseño de la REST API fuer necesario primero un proceso de mapeo. La documentación de este proceso de adjunta en anexos bajo el nombre de "Tarea Corta 3".

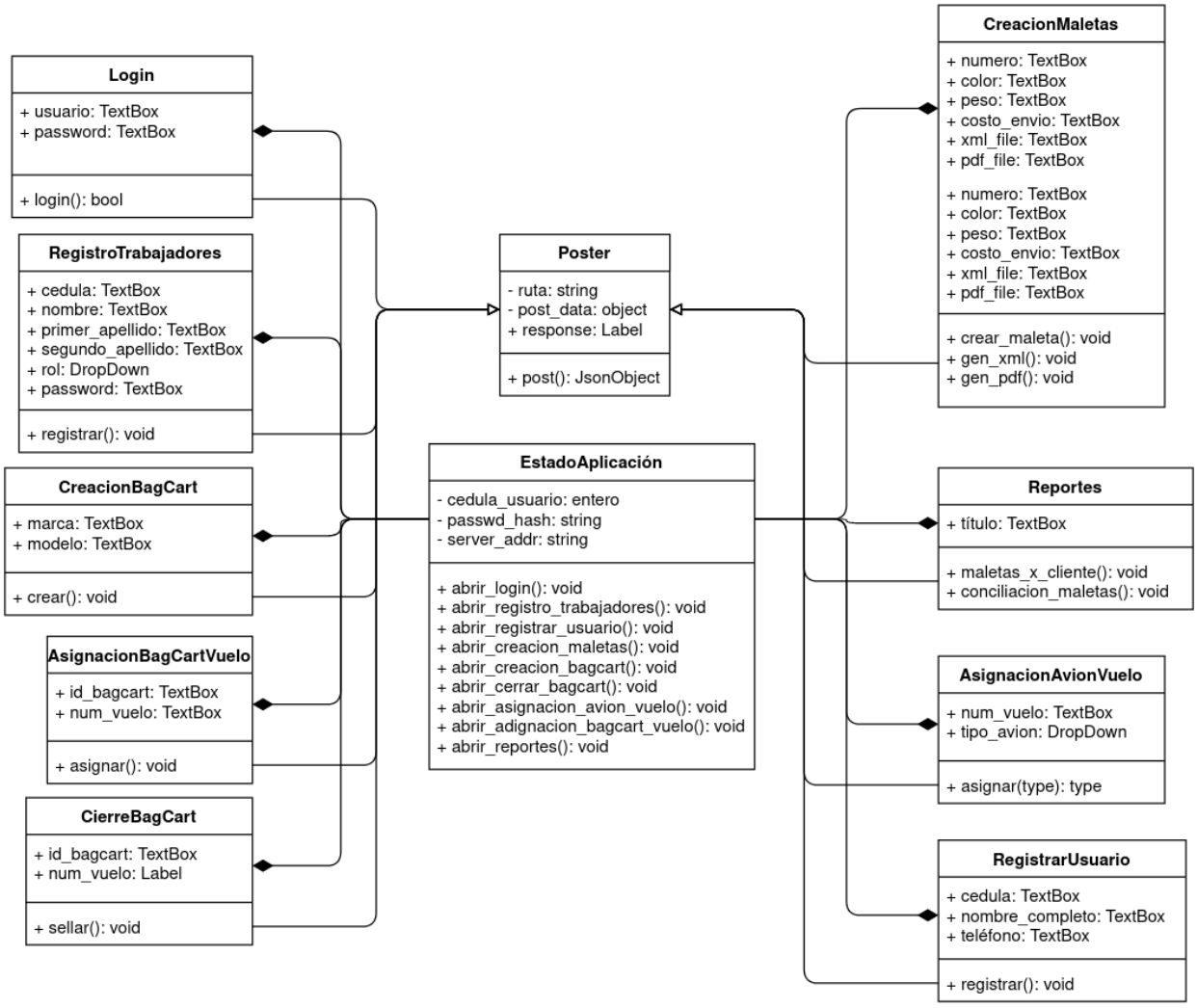

Figura 5: Diagrama de clases de la App Web.

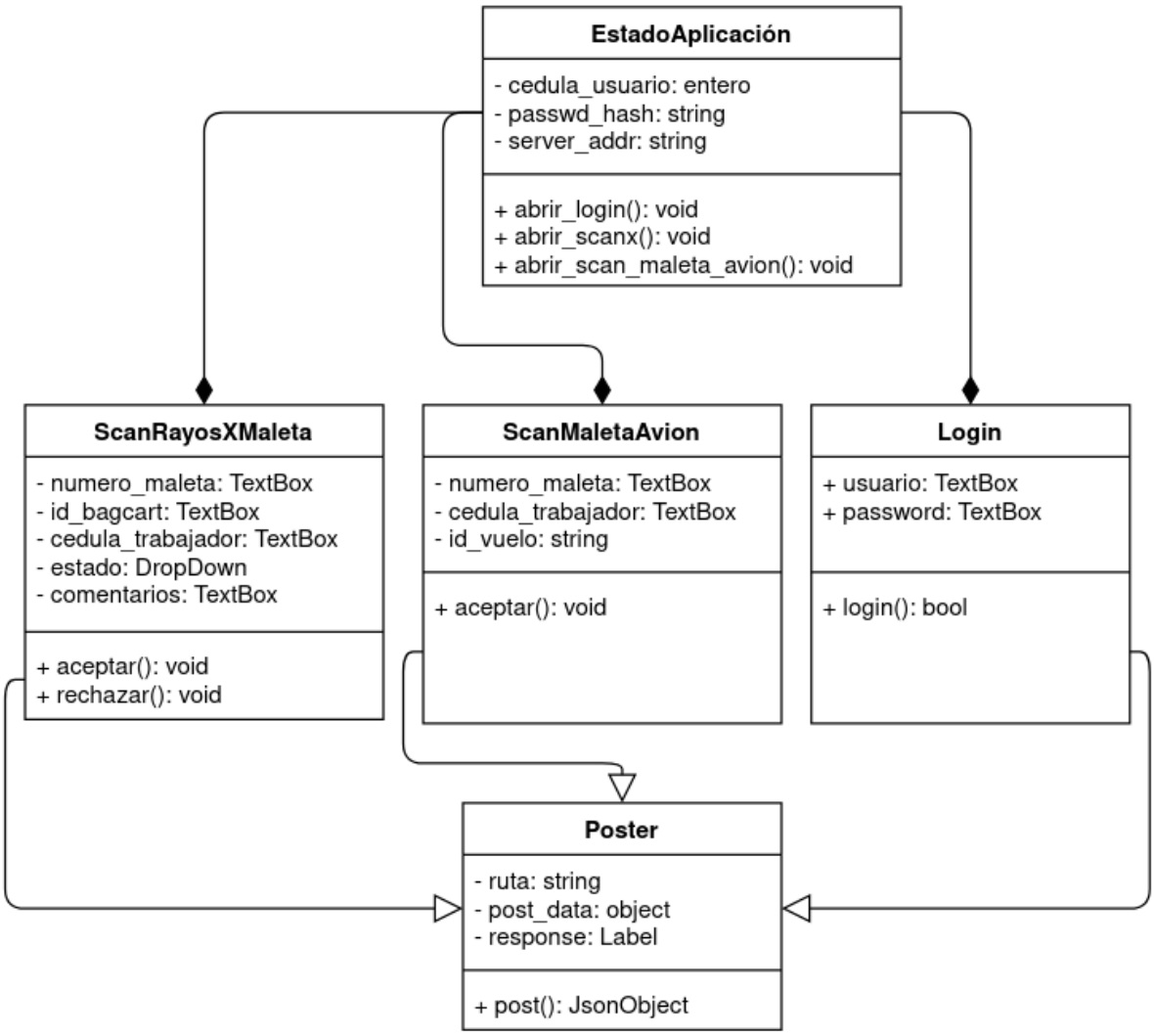

Figura 6: Diagrama de clases de la App Móvil

Los diseños de los frontends son un poco menos específicos debido a que las tecnologías utilizadas como angular no necesariamente piensan en términos de "clases", sino que se basan en la idea de componentes. Igualmente se buscó ilustrar la funcionalidad del diseño utilizando diagramas de clases que describen la estructura general de estas aplicaciones.

Sobre el diseño de la REST API

Como se puede notar, esta aplicación es relativamente simple a nivel de arquitectura. Se aplicó el patrón "Singleton" de manera que solo exista un controlador del archivo de bases de datos, de forma que se puedan evitar condiciones de carrera. La gran parte de las clases son "DataClasses" y se utilizan principalmente para la serialización y deserialización de datos.

Bibliografía

[1] Microsoft, «How to serialize and deserialize JSON using C# - .NET». feb. 2022, [En línea]. Disponible en: https://docs.microsoft.com/en-us/dotnet/standard/serialization/system-text-json-how-to?pivots=dotnet-6-0.

[2] Microsoft, «Tutorial: Create a minimal web API with ASP.NET Core». feb. 2022, [En línea]. Disponible en: https://docs.microsoft.com/en-us/aspnet/core/tutorials/min-web-api?view=aspnetcore-6.0&tabs=visual-studio.

[3] Microsoft, «Create a Web Site». may 2020, [En línea]. Disponible en: https://docs.microsoft.com/en-us/iis/get-started/getting-started-with-iis/create-a-web-site.

[4] Google, «Getting started with Angular». sep. 2021, [En línea]. Disponible en: https://angular.io/start.

[5] Google, «Getting started with Angular». 2022, [En línea]. Disponible en: https://angular.io/guide/deployment.

[6] R. Elmasri y S. Navathe, *Fundamentals of Database Systems*, 7.ª ed. Pearson, 2016.

[7] «npm: bcrypt». feb. 2021, [En línea]. Disponible en: https://www.npmjs.com/package/bcrypt.

Anexos

Instituto Tecnológico de Costa Rica

Plan de Actividades Tarea Corta 1

José Morales Vargas, carné 2019024270
Alejandro Soto Chacón, carné 2019008164
Ignacio Vargas Campos, carné 2019053776
José Retana Corrales, carné 2020144743

Área Académica de

Ingeniería en Computadores

Bases de Datos

(CE3101)

Profesor Marco Rivera Meneses

Semestre I 2022

Actividad	Responsable	Duración	Fecha de entrega
Reunión inicial para coordinación de equipo de trabajo	José Morales Alejandro Soto José Retana Ignacio Vargas	1:30:00	2022-02-25
Desarrollo de plan de trabajo	José Morales Alejandro Soto José Retana Ignacio Vargas	1:45:00	2022-02-26
Creación de repositorios y documentos colaborativos	José Morales Alejandro Soto José Retana Ignacio Vargas	0:30:00	2022-02-27
División de roles principales de cada miembro	José Morales Alejandro Soto José Retana Ignacio Vargas	0:30:00	2022-02-26
Elaboración del modelo conceptual de la base de datos (Entidad-Relación y Clases)	José Morales	2:30:00	2022-03-02
Definición del modelo de base de datos	Soto	2:00:00	2022-03-02
Investigar sobre sistema de generación de reportes	José Retana	1:30:00	2022-03-02
Definición de sistema de generación de reportes a utilizar	José Retana	0:15:00	2022-03-05
Investigar sobre uso de IIS	José Retana	2:00:00	2022-03-06
Investigar sobre el uso de Angular y Bootstrap	Ignacio Vargas	2:00:00	2022-03-06
Investigar sobre lo necesario para creación de REST APIs en C#	José Morales Alejandro Soto	1:30:00	2022-03-06
Creación de los proyectos base para cada entregable	José Morales Alejandro Soto José Retana Ignacio Vargas	1:15:00	2022-03-06
Investigar acerca del esquema de facturación digital del Ministerio de Hacienda de la República de Costa Rica	José Morales Alejandro Soto José Retana Ignacio Vargas	2:30:00	2022-03-07
Investigar acerca de la interacción entre aplicaciones C# e IIS	José Morales José Retana	2:15:00	2022-03-07
Reunión para definir UX y UI	José Retana Ignacio Vargas	1:30:00	2022-03-07
Planeamiento de desarrollo de aplicación móvil	Ignacio Vargas	2:00:00	2022-03-07
Primer avance de resumen ejecutivo	José Morales Alejandro Soto	1:30:00	2022-03-09
Implementación inicial de la REST API	José Morales Ignacio Vargas	2:45:00	2022-03-10
Implementación inicial de la interfaz gráfica de la aplicación web	José Retana Ignacio Vargas	2:45:00	2022-03-11
Implementación inicial de la interfaz gráfica de la aplicación móvil	José Retana Ignacio Vargas	4:15:00	2022-03-11
Sistema de usuarios, registro e inicio de sesión	José Morales Alejandro Soto	2:30:00	2022-03-11

Implementación de primitivos de la base de datos (XML o JSON)	Alejandro Soto	4:45:00	2022-03-11
Maletas y relaciones maletas-aviones-vuelos	Alejandro Soto	2:15:00	2022-03-11
Pruebas iniciales de montaje sobre IIS	Alejandro Soto José Retana	2:00:00	2022-03-11
Generación de reportes (PDF)	José Retana	2:15:00	2022-03-12
Manual de usuario	Ignacio Vargas	2:15:00	2022-03-13
Documentación interna de frontend	José Morales Alejandro Soto José Retana	1:30:00	2022-03-14
Documentación interna de backend	Alejandro Soto José Retana Ignacio Vargas	2:15:00	2022-03-15
Segundo avance de resumen ejecutivo	José Retana Ignacio Vargas	1:15:00	2022-03-16
Fase de pruebas y mejoras finales	José Morales Alejandro Soto José Retana Ignacio Vargas	3:30:00	2022-03-16
Entrega Final	José Morales Alejandro Soto José Retana Ignacio Vargas		2022-03-18

Instituto Tecnológico de Costa Rica

Tarea Corta 3

José Morales Vargas, carné 2019024270
Alejandro Soto Chacón, carné 2019008164
Ignacio Vargas Campos, carné 2019053776
José Retana Corrales, carné 2020144743

Área Académica de

Ingeniería en Computadores

Bases de Datos

(CE3101)

Profesor Marco Rivera Meneses

Semestre I 2022

Índice

Mapeo de Diagrama ER a Modelo Relacional

Diagrama ER

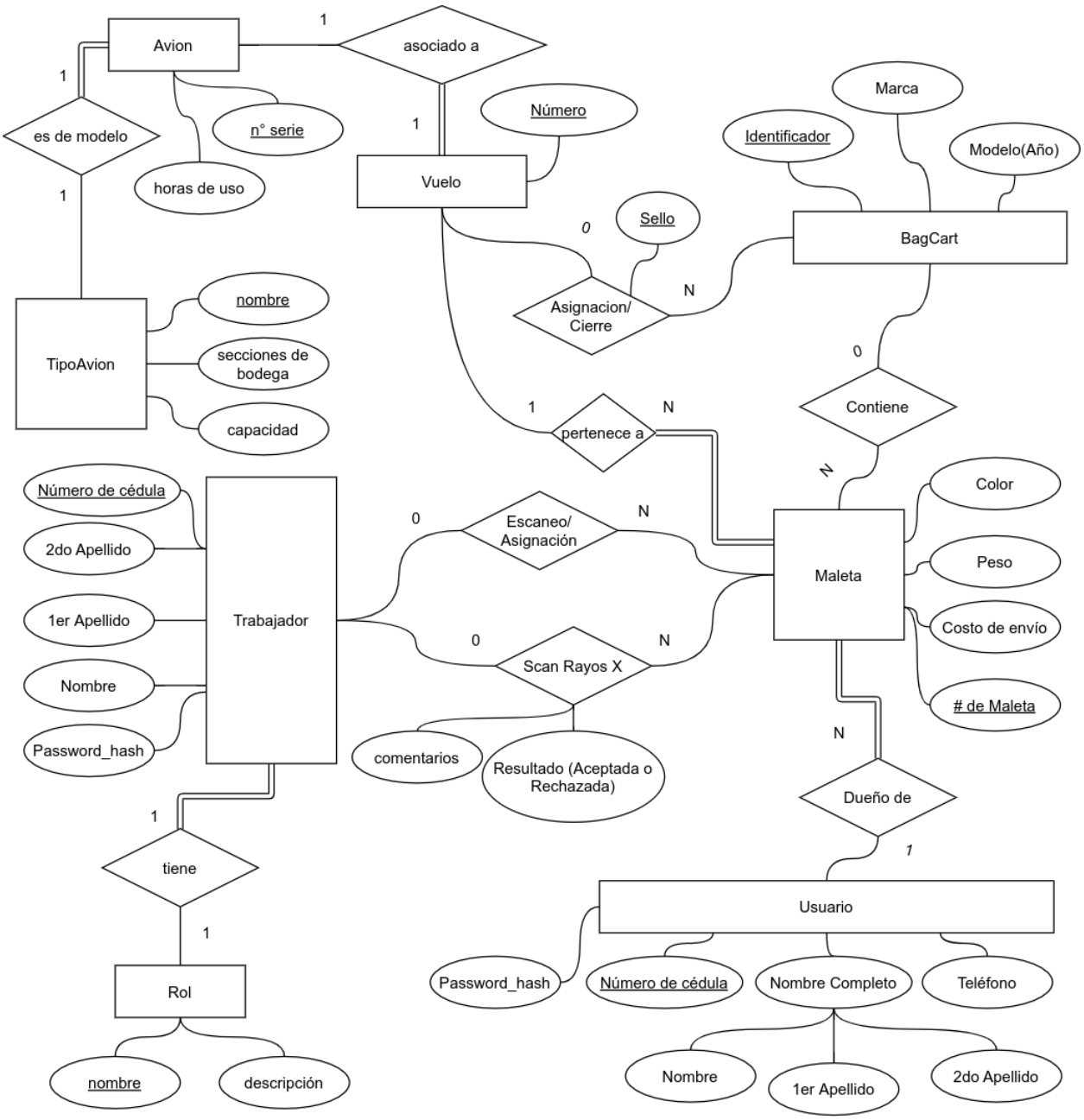

Figura 1: Diagrama Entidad Relación

Mapeo de entidad TipoAvion

Esta entidad es una entidad fuerte con solo atributos simples, por lo cual se mapea a una relación con los mismos atributos que tiene como llave primaria el atributo `nombre`.

Cuadro 1: Relación `TipoAvion`

Campo	Tipo de Dato
nombre(PK)	string
secciones_bodega	entero
capacidad	entero

Mapeo de entidad Avion y relación "es de modelo"

La entidad `Avion` es una entidad fuerte con atributos simples. Inicialmente se mapea como una relación con los mismos atributos de la entidad, con el atributo numero de serie(`nserie`) como la llave primaria.

Posteriormente, se identifica que `TipoAvion` y `Avion` se encuentran en una relación binaria 1:N, por lo que se agrega un atributo `modelo` en la relación `Avion` que referencia como llave foránea al atributo `nombre` de `TipoAvion`.

Cuadro 2: Relación `Avion`

Campo	Tipo de Dato
nserie(PK)	entero
horas_uso	entero
modelo(FK)	string

Mapeo de entidad Vuelo y relación "asociado a"

`Vuelo` es una entidad fuerte de un solo atributo simple. Inicialmente se mapea como una relación con el atributo `numero` como llave primaria de la relación.

Se nota que las entidades `Vuelo` y `Avion` se encuentran en una relación binaria N:1 (Un vuelo solo se asocia a un avión, pero a través del tiempo un avión se asocia a varios vuelos). Se decide entonces agregar un atributo `avion` como llave foránea que referencia a un número de serie de un avión en la relación `Avion`

Cuadro 3: Relación `Vuelo`

Campo	Tipo de Dato
numero(PK)	entero
avion(FK)	entero

Mapeo de entidad Rol

`Rol` es una entidad fuente con dos atributos simples. Se mapea como una relación con los mismos atributos y se toma `nombre` como la llave primaria.

Cuadro 4: Relación `Rol`

Campo	Tipo de Dato
nombre(PK)	string
descripcion	string

Mapeo de entidad Trabajador y relación "tiene" con Rol

`Trabajador` es una entidad fuerte con varios atributos simples. Inicialmente se mapea a una relación con estos mismos atributos y se toma `cedula` como la llave primaria.

`Trabajador` y `Rol` se relacionan por medio de la relación "tiene" la cual es de 1:N (un trabajador puede tener un solo rol, pero un rol no es exclusivo para un trabajador). Esta relación se mapeo agregando un atributo `rol` que funciona como llave foránea que referencia el atributo `nombre` de `Rol`.

Cuadro 5: Relación `Trabajador`

Campo	Tipo de Dato
cedula(PK)	entero
password_hash	string
nombre	string
primer_apellido	string
segundo_apellido	string
rol	string

Mapeo de entidad Usuario

Usuario es un entidad fuerte con varios atributos simples y uno complejo (nombre completo) el cual a la hora de realizar el mapeo se descompone en atributos simples `nombre`, `primer_apellido`, `segundo_apellido`. Una vez descompuesto nombre completo, se mapea la entidad a una relación del mismo nombre y con los mismos atributos. Se utiliza el atributo `cédula` como llave primaria.

Cuadro 6: Relación `Usuario`

Campo	Tipo de Dato
cedula(PK)	entero
password_hash	string
nombre	string
primer_apellido	string
segundo_apellido	string
telefono	entero

Mapeo de entidad Maleta y relaciones "Dueño de" y "Asignada a"

`Maleta` es una entidad fuerte con solo atributos simples. Inicialmente se mapea a una relación con mismo nombre y atributos en la cual `numero` funciona como la llave primaria.

Se nota que existen dos relaciones concernientes a la maleta, el mapeo para cada una es el siguiente:

- "Dueño de" (`Maleta-Usuario`): Es una relación N:1, por lo que se agrega un atributo `cedula_usuario` como llave foránea que referencia el atributo `cedula` de la relación `Usuario`.
- "Asignada a" (`Maleta-Vuelo`): Es una relación N:1, de igual manera se decide agregar un atributo `nvuelo` como llave foránea que referencia al atributo `numero` de la relación 'vuelo.

Cuadro 7: Relación `Maleta`

Campo	Tipo de Dato
numero(PK)	entero
cedula_usuario(FK)	entero
color	entero
peso	flotante
costo_envio	flotante
nvuelo(FK)	entero

Mapeo de entidad BagCart

`BagCart` es una entidad fuerte con varios atributos simples. Se mapea a una relación con estos mismos atributos y se utiliza el atributo `id` como llave primaria.

Cuadro 8: Relación `BagCart`

Campo	Tipo de Dato
id(PK)	entero
marca	string
modelo	entero

Mapeo de relación Trabajador-Maleta "Scan Rayos X"

Esta relación es de tipo 0:N (una maleta puede no haber sido escaneada en rayos x todavía, y un trabajador puede haber escaneado varias maletas), por lo cual se mapea a una referencia cruzada que contiene los atributos de la relación. En esta referencia cruzada, `cedula_trabajador` y `numero_maleta` son llaves foráneas que representan al atributo `cedula` de la relación `Trabajador` y el atributo `numero` de la relación Maleta.

Cuadro 9: Relación `RelScanRayosXMaleta`

Campo	Tipo de Dato
cedula_trabajador(FK)	entero
numero_maleta(FK)	entero
aceptada	bool
comentarios	string

Mapeo de relación Trabajador-Maleta "Escaneo/Asignación"

De igual manera al caso de `Scan Rayos X` Esta relación es de tipo 0:N (Una maleta puede no estar ya escaneada y asignada a un avión, pero un trabajador puede procesar varias maletas), por lo cual se mapea a una referencia cruzada que contiene los atributos de la relación. En esta referencia cruzada, `cedula_trabajador` y `numero_maleta` son llaves foráneas que representan al atributo `cedula` de la relación `Trabajador` y el atributo `numero` de la relación Maleta.

Cuadro 10: Relación `RelScanAsignacionMaleta`

Campo	Tipo de Dato
cedula_trabajador(FK)	entero
numero_maleta(FK)	entero

Mapeo de relación Maleta-Bagcart "Contiene"

Esta relación es de tipo 0:N (un bagcart puede contener varias o ninguna maleta, y puede ser que una maleta no esté contenida en un bagcart), por lo que también se mapea como una referencia cruzada. El atributo `id_bagcart` funciona como llave foránea del atributo `id` de la relación `Bagcart`. El atributo `numero_maleta` funciona como llave foránea del atributo "

Cuadro 11: Relación `RelMaletaBagCart`

Campo	Tipo de Dato
id_bagcart(FK)	entero
numero_maleta(FK)	entero

Mapeo de relación "Asignación/Cierre"

Esta relación es de cardinalidad 0:N (Un bagcart puede no estar cerrado, un vuelo puede tener asignados varios bagcarts) y tiene un atributo llave `sello`. Esto se mapea a una relación de referencia cruzada en la que `id_bagcart` y `nvuelo` funcionan como llaves foráneas que referencian al atributo `id` de la relación `BagCart` y `numero` de la relación `Vuelo` respectivamente. El atributo `sello` se mapea como un atributo de la relación que funciona como llave primaria.

Cuadro 12: Relación `RelVueloBagCart`

Campo	Tipo de Dato
sello(PK)	string
id_bagcart(FK)	entero
nvuelo(FK)	entero

Diagrama modelo relacional resultante

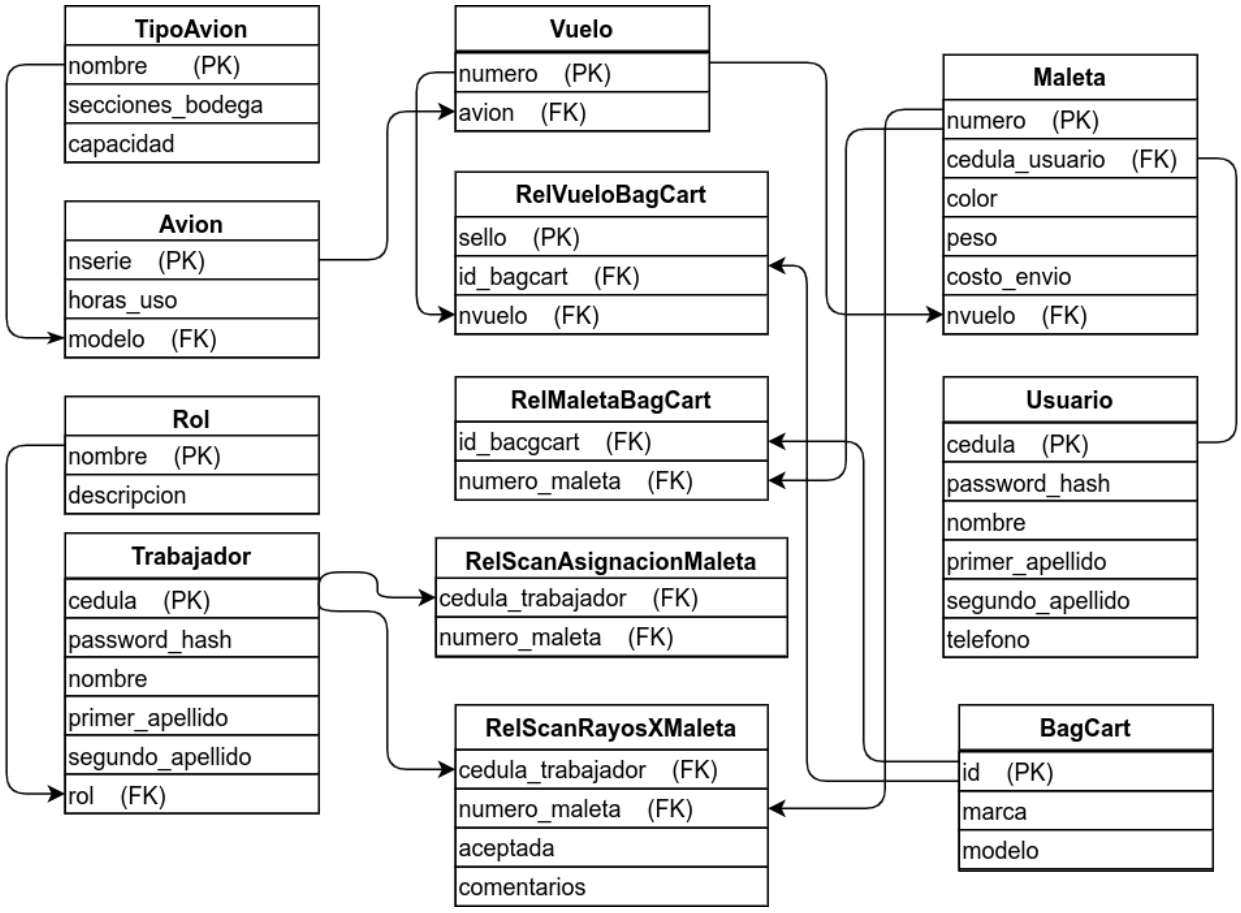

Figura 2: Diagrama modelo relacional

Diagrama Arquitectura

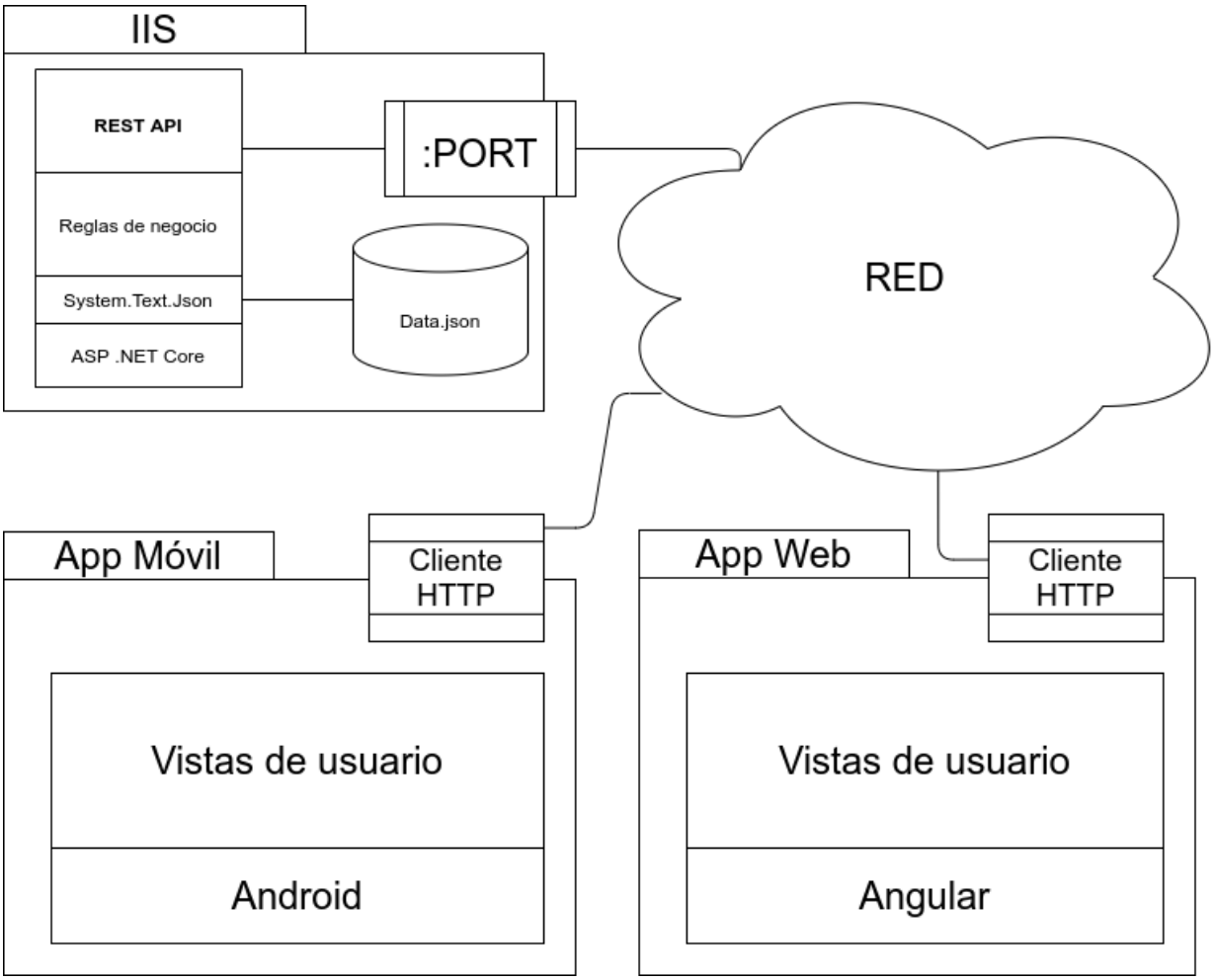

Figura 3: Diagrama de Arquitectura