TEC | Tecnológico de Costa Rica

Área académica de Ingeniería en Computadores

Bases de Datos – CE3101

TAREA CORTA #1
Documentación

Estudiantes:

Josué Santamaría Ramírez

Johnny Zaet Agüero Sandí

José Fabián Mendoza Mata

Esteban Madrigal Sandoval

Profesor:

Marco Rivera Meneses

IS - 2022

ÍNDICE

1. Modelo Conceptual

Basado en las características que se proponen para las entidades según la notación de Chen y tomando en consideración el contexto de la presente tarea, se propuso el siguiente modelo conceptual, en el cual se consideraron ciertos valores como entidades y estos tienen además, sus atributos y relaciones respectivos.

Entidad: *Usuario*

Atributo(s):

- *Nombre*
- *Teléfono*
- *Contraseña*
- *Cédula*

Relación(es):

- *Tiene maleta*

Entidad: *Trabajador*

Atributo(s):

- *Nombre*
- *Apellidos*
- *Cédula*
- *Contraseña*
- *Rol*

Relación(es):

- *Crea usuario*
- *Crea maleta*
- *Escanea maleta*
- *Crea bagcart*
- *Asigna maleta a bagcart*

Entidad: *Maleta*

Atributo(s):

- *Número*
- *Usuario*
- *Tipo*
- *Costo*
- *Color*
- *Estado*

Relación(es):

- *N/A*

Entidad: *Bagcart*

Atributo(s):

- *Identificador*
- *Marca (atributo multivaluado)*
- *Cantidad de maletas*
- *Modelo*

Relación(es):

- *Se asigna un sello de seguridad*
- *Se asigna a un vuelo*

Entidad: *Vuelo*

Atributo(s):

- *Número*
- *Total de maletas*

Relación(es):

- *Tiene un avión*

Entidad: *Avión*

Atributo(s):

- *Tipo*
- *Cantidad de maletas*
- *Capacidad*

Relación(es):

- *N/A*

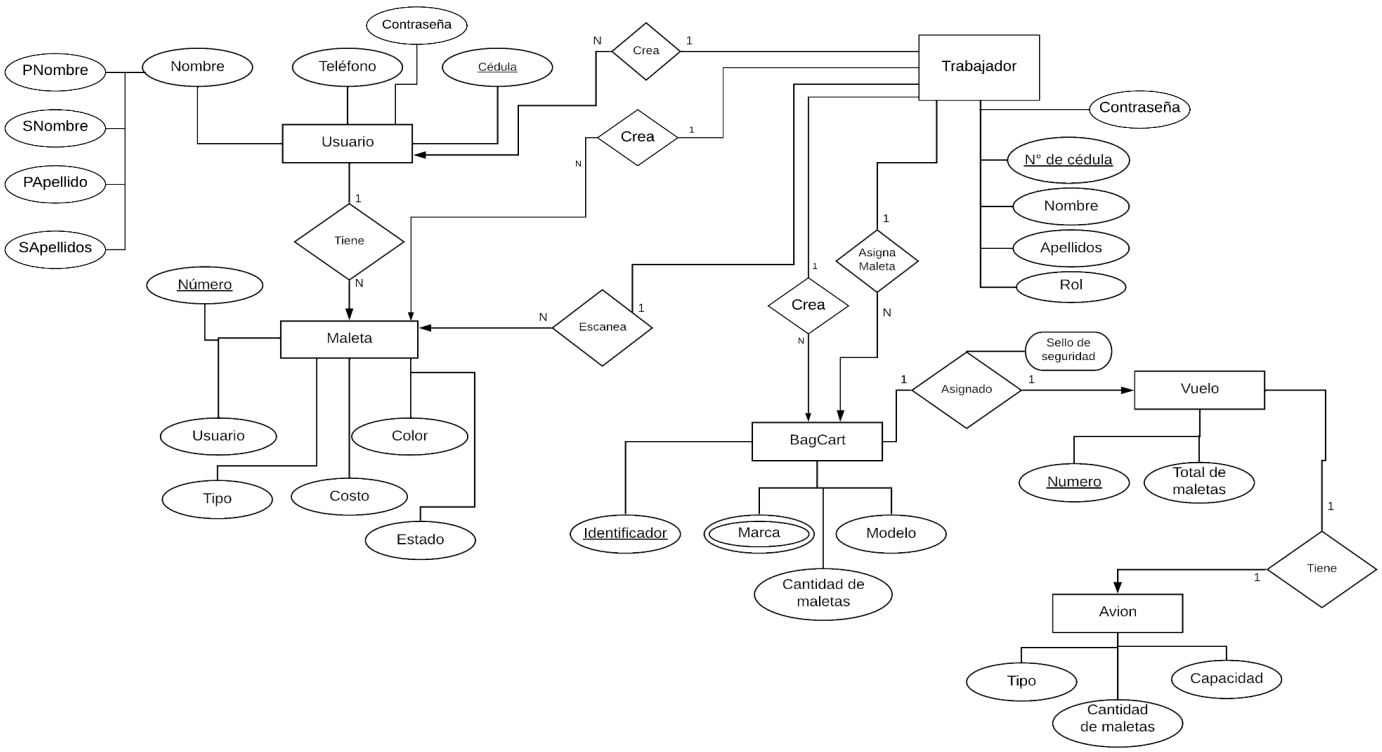

Figura 1. Diagrama Modelo Conceptual

2. Modelo Relacional

Para realizar el modelo conceptual, se realizó primeramente el mapeo de las entidades fuertes, en el cual se obtuvieron 6 entidades fuertes y cada una de estas tiene a su vez, su llave principal (PK), obteniendo lo siguiente:

Tabla 1. Entidades y sus llaves primarias

Entidad	Llave Principal (PK)
Trabajador	Cédula
Usuario	Cédula
Maleta	NúmeroMaleta
BagCart	Identificador
Vuelo	NúmeroVuelo
Avión	Tipo

Posteriormente, al no tener entidades débiles, se procedió con el mapeo de las asociaciones binarias 1:1, en las cuales se tomó en cuenta cual era la entidad que no podía permitir atributos nulos, esto para que el valor del atributo de la relación no se viese afectado por un valor nulo.

De este tipo de relación se tenían 2, las cuales eran que un bagcart tiene asignado un vuelo y que un vuelo tiene asignado un avión. Esto se puede notar como llave foránea (FK) en la tabla de BagCart y Vuelo respectivamente.

Ahora bien, con respecto al mapeo de las asociaciones binarias de tipo 1:N, se identificaron 5 relaciones de este tipo, las cuales se detallan a continuación junto con las del tipo 1:1.

Tabla 2. Tipos de relaciones entre entidades

Relación	Tipo
Un Bagcart tiene asignado un Vuelo	1:1
Un Vuelo tiene asignado un Avión	1:1
Trabajador crea un Usuario	1:N
Trabajador crea una Maleta	1:N
Trabajador escanea una Maleta	1:N
Trabajador crea un BagCart	1:N
Trabajador asigna una Maleta a un BagCart	1:N

Las relaciones anteriores se denotaron con llaves foráneas en las tablas respectivas de cada entidad involucrada en la relación. Basado en lo anterior, se obtuvo el siguiente esquema de tabla, producto del mapeo realizado.

Figura 2. Mapeo de Modelo Conceptual a Modelo Relacional

Basado en lo expuesto anteriormente, se procedió a pasar de las tablas anteriores a las tablas del modelo relacional, para lo cual cada una de las relaciones se representó mediante referencia cruzada, usando así una relación extra para representar dichas asociaciones.

De esto, se obtuvo el siguiente modelo relacional:

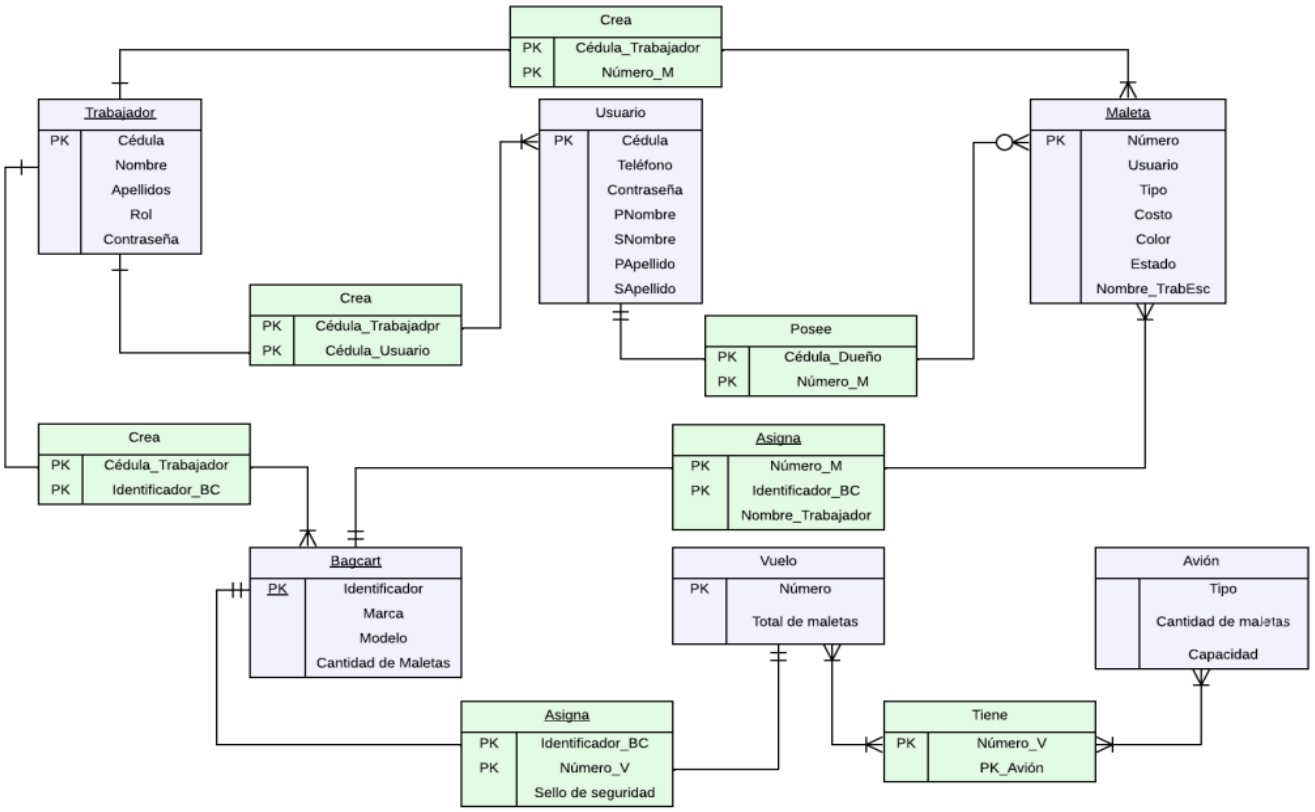

Figura 3. Modelo Relacional

3. Descripción de las estructuras de datos desarrolladas

Con respecto a las estructuras de datos desarrolladas para modelar las entidades a usar, las cuales son Trabajador, Usuario, Maleta, Bagcart, Vuelo y Avión, estas se implementaron como clases, haciendo uso del paradigma de programación orientada a objetos (POO), dado que esto era lo más adecuado para este proyecto y además, facilitaba mucho el manejo, acceso y manipulación de las entidades.

Para el caso de la clase Trabajador, sus atributos se definieron de la siguiente manera:

- Cédula => id : argumento es un “número”
- Contraseña => password : argumento es un “string”
- Nombre => name : argumento es un “string”
- Apellidos => lastname : argumento es un “string”
- Rol => rol : argumento es un “string”

Lo anterior se puede apreciar de una mejor manera en la siguiente imagen del código implementado.

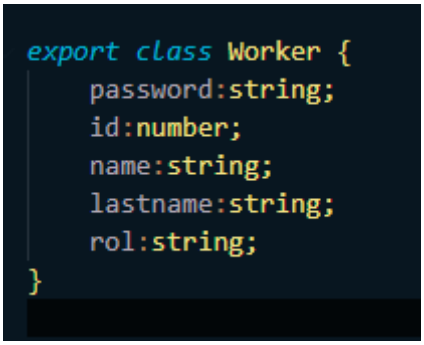

Figura 4. Implementación de clase Trabajador

Para el caso de la clase Usuario, sus atributos se definieron de la siguiente manera:

- Cédula => id : argumento es un “número”
- Teléfono => telephone : argumento es un “número”
- Nombre => name : argumento es un “string”
- Contraseña => password : argumento es un “string”

Lo anterior se puede apreciar en la siguiente imagen.

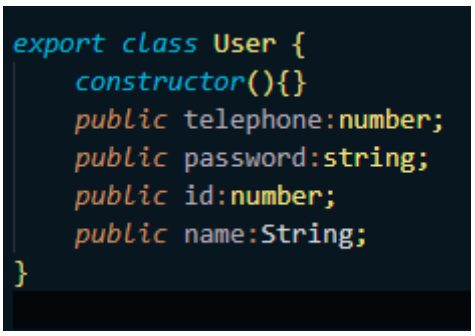

Figura 5. Implementación de clase Usuario

Para el caso de la clase Maleta, sus atributos se definieron de la siguiente manera:

- Identificador => id : argumento es un “número”
- Usuario => usuario : argumento es de tipo “User”
- Costo => cost : argumento es un “número”
- Estado => state : argumento es un “boolean”
- Color => color : argumento es un “string”
- Tipo => type : argumento es un “string”

Lo anterior se puede apreciar en la siguiente imagen.

Figura 6. Implementación de clase Maleta

Para el caso de la clase BagCart, sus atributos se definieron de la siguiente manera:

- Identificador => id : argumento es un “número”
- Modelo => model : argumento es un “string”
- Maletas => bags : argumento es un “número”
- Marca => brand : argumento es un “string”
- Sello => stamp : argumento es un “string”

Lo anterior se puede apreciar en la siguiente imagen.

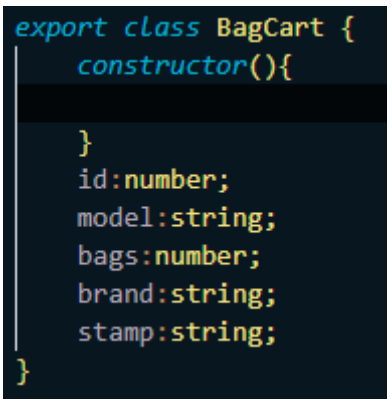

Figura 7. Implementación de clase BagCart

Para el caso de la clase Vuelo, sus atributos se definieron de la siguiente manera:

- Identificador => id : argumento es un “número”
- Maletas => bags : argumento es un “número”
- Avión => plane : argumento es de tipo “Plane”

Lo anterior se puede apreciar en la siguiente imagen.

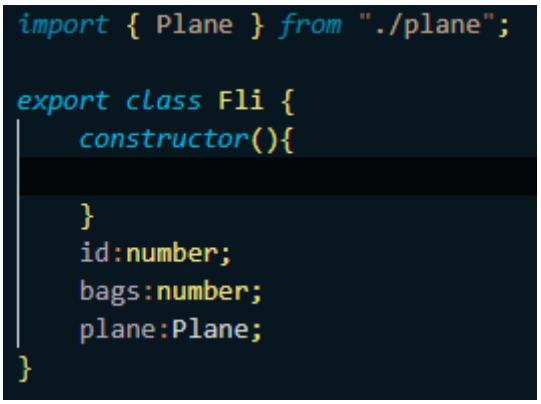

Figura 8. Implementación de clase Vuelo

Para el caso de la clase Avión, sus atributos se definieron de la siguiente manera:

- Tipo => type : argumento es un “string”
- Capacidad => capacity : argumento es un “número”
- Maletas => bags : argumento es un “número”

Lo anterior se puede apreciar en la siguiente imagen.

```
export class Plane {
    type:string;
    capacity:number;
    bags:number;
}
```

Figura 9. Implementación de clase Avión

Con respecto a las estructuras de datos desarrolladas en la API, estas se modelaron de manera similar a las desarrolladas en el frontend, para así evitar ambigüedades y que se hiciese la comunicación de una manera más efectiva. Esto se puede corroborar con la siguiente imagen.

```
1   using System;
2   using System.Collections.Generic;
3   using System.Linq;
4   using System.Threading.Tasks;
5
6   namespace Core
7   {
8       public class User
9       {
10          public int telephone { get; set; }
11          public string password { get; set; }
12          public int id { get; set; }
13          public string name { get; set; }
14      }
15  }
16
```

Figura 10. Implementación de clase Usuario en API

4. Descripción de la arquitectura desarrollada

Con base a la especificación obtenida para la presente tarea, se optó por definir la arquitectura de la aplicación de la siguiente manera:

- Dado que se debe de tener una interfaz gráfica para los usuarios, otra interfaz gráfica similar para los administradores y una aplicación móvil, se optó por denotar los anteriores con color verde, el cual significa la interacción de un usuario o administrador con la aplicación.
- Como los trabajadores o administradores tienen acceso total a la aplicación, se modeló esto de forma tal que dichas personas tienen acceso a la interfaz de administrador, la cual se utiliza para la asignación de vuelos, bagcarts y comunicación con el Ministerio de Hacienda, así como también tienen acceso a la aplicación móvil para controlar las acciones mencionadas anteriormente de una mejor forma.
- Los usuarios por su parte, solamente tienen acceso a la interfaz gráfica de usuario la cual les permite interactuar con la aplicación y escoger entre las opciones disponibles la que mejor se adecue a sus preferencias.
- Posteriormente, estas interfaces se comunican con una base de datos mediante un servidor, el cual se encuentra desplegado en Internet Information Services (IIS) y es de esta manera que se interactúa y se obtiene información de dicha base de datos.
- La base de datos mencionada es utilizada para la comprobación de inicio de sesión, comprobación de asignación de maletas, acceso al registro de bagcarts y reportes PDF generados producto de las interacciones de los usuarios con la aplicación, los cuales se denotan de color amarillo.

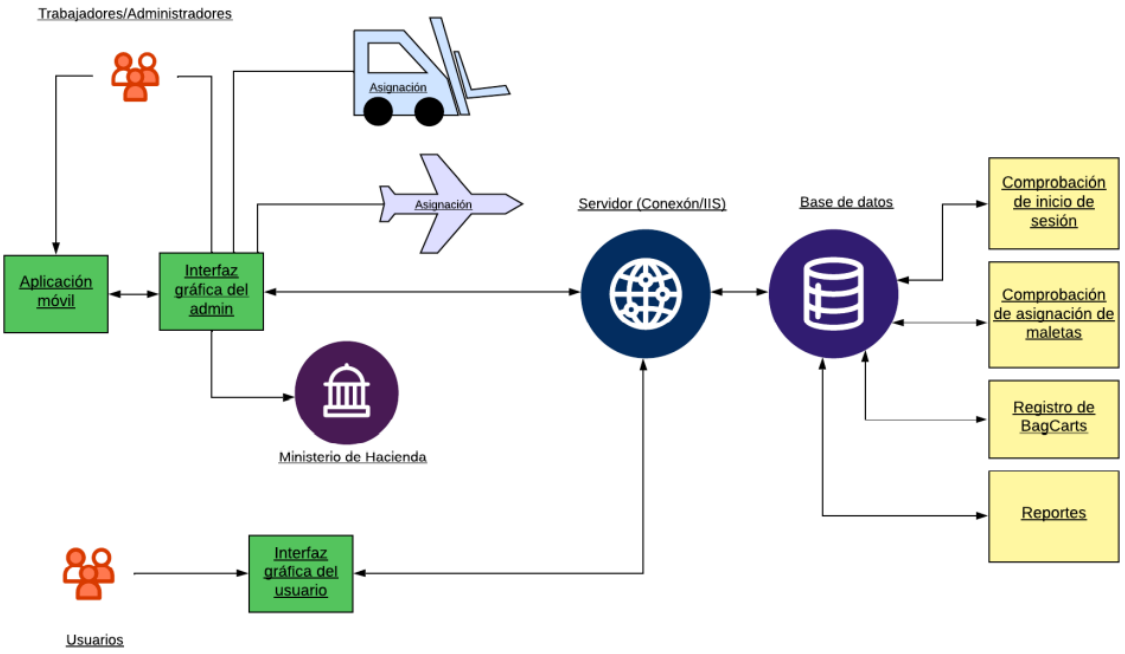

Figura 11. Diagrama de arquitectura de la aplicación

5. Diagrama de clases

Para el diagrama de clases de la aplicación desarrollada se tomaron en cuenta ciertos aspectos, entre los cuales se pueden mencionar:

- La especificación propia de la aplicación.
- El uso y funcionalidad que representa cada entidad tomando como base el modelo relacional.
- El modelado de la arquitectura de la aplicación.

Lo anterior se puede representar de la siguiente manera:

Clase: *Usuario*

Atributo(s)*:*

- *NombreCompleto*
- *Teléfono*
- *Cédula*

Función(es):

- *ObtenerUsuario*

Clase: *Trabajador*

Atributo(s)*:*

- *Cédula*
- *Nombre*
- *Apellidos*
- *Rol*

Función(es):

- *CrearUsuario*
- *CrearMaleta*
- *EscanearMaleta*
- *AsignarMaleta*
- *AsignarMaletaAvión*

Clase: *Maleta*

Atributo(s)*:*

- *Usuario*
- *Color*
- *Peso*
- *Costo*
- *NúmeroDeMaleta*
- *Estado*

Función(es):

- *ObtenerMaleta*
- *RechazarMaleta*

Clase: *BagCart*

Atributo(s)*:*

- *ID*
- *Marca*
- *Modelo*
- *CantidadDeMaletas*

Función(es):

- *CierreBagCart*

Clase: *Vuelo*

Atributo(s)*:*

- *Número*
- *TotalDeMaletas*

Función(es):

- *ObtenerAvión*
- *AsignarBagCart*

Clase: *Avión*

Atributo(s)*:*

- *Tipo*
- *CantidadDeMaletas*
- *Capacidad*

Función(es):

- *EstablecerVuelo*

Clase: *Reportes*

Atributo(s)*:*

- *N/A*

Función(es):

- *CreaciónFactura*
- *ReportePDF*

Con base en lo anterior, se obtuvo el siguiente diagrama de arquitectura de la aplicación desarrollada:

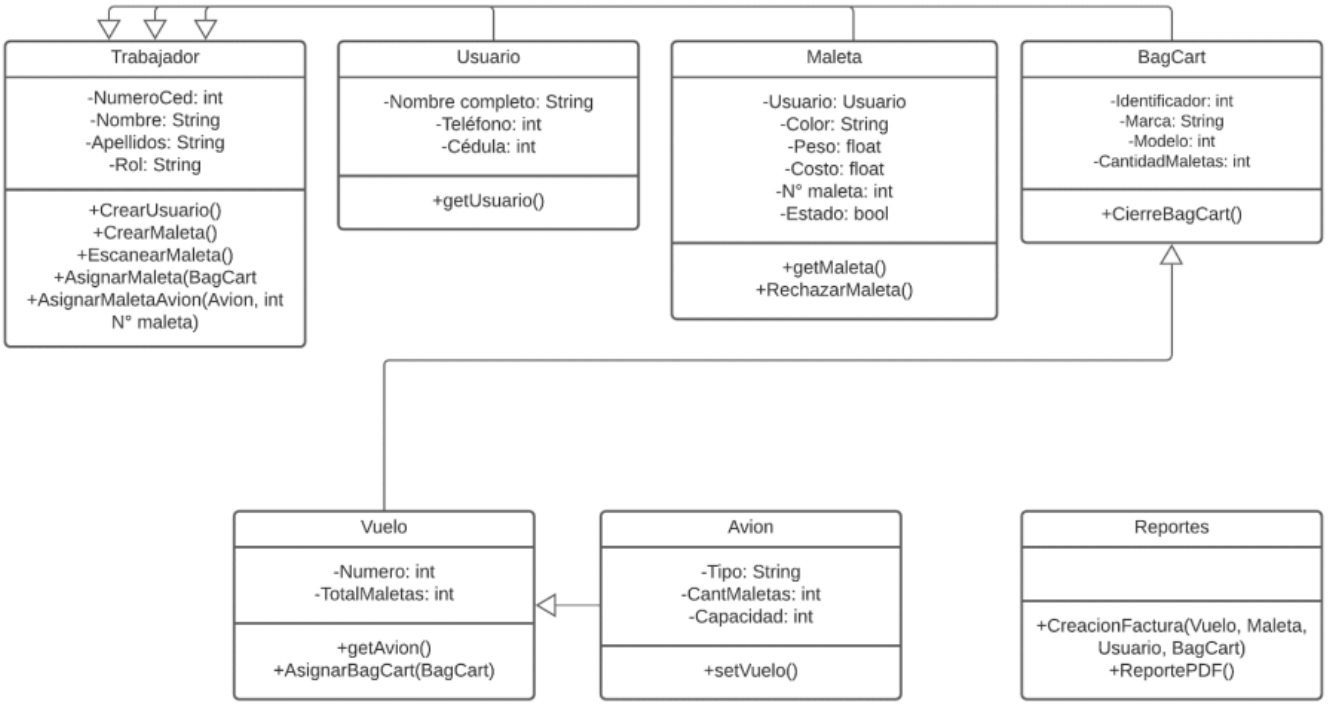

Figura 12. Diagrama de clases de la aplicación

6. Problemas conocidos

Como problemas conocidos se tiene lo siguiente:

Tabla 3. Problemas conocidos de la aplicación

Problema	Descripción
Asignación de un BagCart a un Vuelo	Esta funcionalidad no se pudo implementar debido a que hacía falta una funcionalidad para que esto funcionara de la forma que se esperaba.
Reportes PDF	Para los reportes PDF, estos no se pudieron generar correctamente, debido a que se requería de otras funcionalidades para que estos pudieran desplegar la información correcta.
Comando POST	Este comando se encuentra implementado y es recibido por la aplicación pero sin embargo, este genera problemas para realizar cambios en la base de datos

7. Documentación de evidencia del trabajo en equipo

Como evidencia del trabajo realizado en equipo, se decidió crear un proyecto en *Jira Software* para así poder tener un mejor control de las actividades pendientes y realizadas. Dicho proyecto se puede encontrar en la siguiente página web:

https://ceengineer.atlassian.net/jira/software/projects/TC1/boards/1/backlog

A manera de resumen de las actividades planeadas, el responsable de estas y su fecha de entrega, se realizó la siguiente tabla (Plan de proyecto) con la finalidad de tener una idea general sobre las funcionalidades de la tarea y su respectivo avance.

Tabla 4. Resumen de tareas pendientes y completas del proyecto

Clave	Resumen	Actualizada	Creador	Estado	Padre	Responsable	Categoría	Fecha de entrega
TC1-11	Login	18/mar/2022	Josué Santamaría Ramírez	Tareas listas	TC1-7	Esteban Madrigal	Terminado	**4/3/2022**
TC1-12	Employee registration	18/mar/2022	Josué Santamaría Ramírez	Tareas listas	TC1-7	Josué Santamaría Ramírez	Terminado	**4/3/2022**
TC1-13	Creation of suitcases	18/mar/2022	Josué Santamaría Ramírez	Tareas listas	TC1-7	Jose Mendoza	Terminado	**4/3/2022**
TC1-14	Assigning a BagCart to a flight	28/feb/2022	Josué Santamaría Ramírez	Tareas por hacer	TC1-7	Johnny Zaet Aguero Sandi	Por hacer	**4/3/2022**
TC1-15	PDF reports	28/feb/2022	Josué Santamaría Ramírez	Tareas por hacer	TC1-7	Esteban Madrigal	Por hacer	**10/3/2022**
TC1-16	Login	18/mar/2022	Josué Santamaría Ramírez	Tareas listas	TC1-8	Josué Santamaría Ramírez	Terminado	**10/3/2022**

TC1-17	Assign/Scan a Suitcase to a BagCart	18/mar/2022	Josué Santamaría Ramírez	En proceso	TC1-8	Esteban Madrigal	En proceso	**10/3/2022**
TC1-18	Assignment of suitcases to an aircraft	18/mar/2022	Josué Santamaría Ramírez	En proceso	TC1-8	Johnny Zaet Aguero Sandi	En proceso	**10/3/2022**
TC1-19	JSON or XML Database	18/mar/2022	Josué Santamaría Ramírez	Tareas listas	TC1-9	Jose Mendoza	Terminado	**14/3/2022**
TC1-20	Deploy on IIS	18/mar/2022	Josué Santamaría Ramírez	Tareas listas	TC1-9	Johnny Zaet Aguero Sandi	Terminado	**14/3/2022**
TC1-21	Post creation	18/mar/2022	Josué Santamaría Ramírez	En proceso	TC1-9	Jose Mendoza	En proceso	**14/3/2022**
TC1-22	Get creation	18/mar/2022	Josué Santamaría Ramírez	Tareas listas	TC1-9	Josué Santamaría Ramírez	Terminado	**14/3/2022**
TC1-23	Get multivalue creation	18/mar/2022	Josué Santamaría Ramírez	Tareas listas	TC1-9	Esteban Madrigal	Terminado	**14/3/2022**
TC1-24	Document the source code	18/mar/2022	Josué Santamaría Ramírez	Tareas listas	TC1-10	Johnny Zaet Aguero Sandi	Terminado	**14/3/2022**
TC1-25	Conceptual model	18/mar/2022	Josué Santamaría Ramírez	Tareas listas	TC1-10	Jose Mendoza	Terminado	**14/3/2022**

TC1-26	Relational model	18/mar/2022	Josué Santamaría Ramírez	Tareas listas	TC1-10	Josué Santamaría Ramírez	Terminado	**18/3/2022**
TC1-27	Description of the data structures developed	18/mar/2022	Josué Santamaría Ramírez	Tareas listas	TC1-10	Esteban Madrigal	Terminado	**18/3/2022**
TC1-28	Description of the developed architecture	18/mar/2022	Josué Santamaría Ramírez	Tareas listas	TC1-10	Johnny Zaet Aguero Sandi	Terminado	**18/3/2022**
TC1-29	Known problems	18/mar/2022	Josué Santamaría Ramírez	Tareas listas	TC1-10	Jose Mendoza	Terminado	**18/3/2022**
TC1-30	Class diagram and an explanation	18/mar/2022	Josué Santamaría Ramírez	Tareas listas	TC1-10	Esteban Madrigal	Terminado	**18/3/2022**
TC1-31	User manual	18/mar/2022	Josué Santamaría Ramírez	Tareas listas	TC1-10	Josué Santamaría Ramírez	Terminado	**18/3/2022**

Minutas de sesiones de trabajo

Sesión #1 (Feb 28)

Tema: Realización del plan de trabajo.

Descripción: Se realizó una sesión sincrónica mediante la plataforma Discord en la que se tomaron decisiones acerca del futuro del proyecto, como el lenguaje a utilizar, además de la asignación de roles y la descomposición del proyecto por partes para así asignar a distintos integrantes y así conseguir un avance porcentual.

Resultados: Se asignaron las tareas por hacer y roles específicos de cada integrante del grupo junto con la fecha esperada de finalización.

Sesión #2 (Mar 3)

Tema: Inicio de desarrollo de funcionalidades del plan de trabajo.

Descripción: Se realizó una sesión sincrónica mediante la plataforma Discord en la que se tomaron las respectivas decisiones acerca de cómo se podían desarrollar las funcionalidades de la aplicación web (login, portal de administrador y usuario, creación y registro de maletas, asignación de BagCarts a ciertos vuelos, etc) y así conseguir un avance porcentual.

Resultados: Se discutieron las posibles formas de desarrollar el proyecto y se empezó a trabajar en las tareas por hacer para poder cumplir con su fecha esperada de finalización.

Sesión #3 (Mar 9)

Tema: Comunicación de la aplicación con el servidor.

Descripción: Se realizó una sesión sincrónica mediante la plataforma Discord en la que se decidió como se podía implementar el servidor de la aplicación y su posterior despliegue en IIS.

Resultados: Se decidió como crear el servidor y de la aplicación y se comenzó con su desarrollo.

Sesión #4 (Mar 14)

Tema: Actualización del plan de trabajo y revisión de funcionalidades pendientes.

Descripción: Se realizó una sesión sincrónica mediante la plataforma Discord en la que se tomaron decisiones acerca de las funcionalidades pendientes del proyecto, como se iban a implementar y utilizar, además de la asignación de roles y la descomposición de estas funcionalidades del proyecto por partes para así asignar a distintos integrantes y así conseguir un avance porcentual.

Resultados: Se asignaron las tareas pendientes por hacer y roles específicos de cada integrante del grupo junto con la fecha esperada de finalización.

Sesión #5 (Mar 15)

Tema: Comunicación de la aplicación web y móvil con servidor.

Descripción: Se realizó una sesión sincrónica mediante la plataforma Discord en la que se decidió como se podía implementar la aplicación móvil y su comunicación con el servidor en IIS.

Resultados: Se decidió como crear la comunicación de la aplicación web y móvil del proyecto y se comenzó con su desarrollo.

Sesión #6 (Mar 16)

Tema: Comunicación de la aplicación con la API y su despliegue en IIS.

Descripción: Se realizó una sesión sincrónica mediante la plataforma Discord en la que se continuó con el desarrollo de la API requerida para el proyecto y su comunicación con el servidor en IIS.

Resultados: Se avanzó en el desarrollo y despliegue de la API en IIS.

Sesión #7 (Mar 17)

Tema: Comunicación de la aplicación con la API.

Descripción: Se realizó una sesión sincrónica mediante la plataforma Discord en la que se continuó con el desarrollo de la API requerida para el proyecto y su comunicación con la aplicación.

Resultados: Se continuó con el desarrollo y comunicación de la API con la aplicación web.

Evidencia de uso de un manejador de código

Como evidencia del uso de un manejador de código para el desarrollo y control del proyecto, se hizo uso de GitHub como plataforma preferida para esto.

El repositorio del proyecto se puede encontrar en la siguiente dirección web:

https://github.com/Santamix728/ITCR.BASESDEDATOS.TareaCorta/

8. Conclusiones y recomendaciones del proyecto

Con respecto a la elaboración de una aplicación web, se debe de tomar en cuenta que esta va a hacer uso de un servidor para la comunicación entre el usuario y la aplicación, así como también se hace uso de una base de datos, en la cual se almacena toda la información requerida para que dicha aplicación funcione adecuadamente. Es por esto que resulta importante que tanto la aplicación así como el servidor y la base de datos estén correctamente implementados, para evitar errores y que estos lleguen a causar que la página web no realice las funciones que se esperan de esta.

Por otra parte, la comunicación que se realiza en una aplicación mediante el uso de un servidor es de suma importancia, dado que los mensajes que este reciba serán tramitados de una forma específica y sino llegan a ser mensajes esperados por este, se produciría un comportamiento inesperado tanto de la aplicación como del servidor, afectando directamente al usuario final.

Respecto a las aplicaciones web móviles, estas permiten tener un fácil acceso a la aplicación web general mediante el uso de un dispositivo móvil, lo cual hace que la aplicación pueda ser accedida en cualquier momento y desde cualquier dispositivo, permitiendo así una disponibilidad las 24 horas del día, los 7 días de la semana y los 365 días del año.

Como recomendación, cuando se desee realizar una aplicación web completamente desde cero, es importante resaltar que existen ciertas librerías o frameworks que facilitan mucho la creación de la aplicación web, no solo desde la parte del servidor que se vaya a utilizar sino también del diseño y desarrollo del frontend y del backend de la aplicación, por lo que este trabajo se puede realizar de una manera automatizada, rápida y ágil sin desperdiciar tiempo en hacer esto desde cero.

Siguiendo con lo mencionado anteriormente, resulta sumamente importante el hecho de que para este proyecto se modeló la base de datos como un archivo JSON, lo cual no resulta del todo correcto porque para esto existen motores de bases de datos, los cuales se encargan de realizar todo lo relacionado al almacenamiento, consulta y actualización de información, por lo que a largo plazo, a gran escala y siendo estos motores de bases de datos lo más utilizado actualmente, es bastante viable crear y utilizar la base de datos de esta manera.

9. Bibliografía

[1]. Angular. (2022). *Introduction to the Angular Docs.* Disponible en: https://angular.io/docs

[2]. Agrimbau, T. (2016). *Developing Mobile Web Applications: When, Why, and How.* Disponible en: https://www.toptal.com/android/developing-mobile-web-apps-when-why-and-how

[3]. Microsoft. (2021). *Let's Learn .NET: Web APIs.* Disponible en: https://docs.microsoft.com/en-us/shows/lets-learn-dotnet/Web-APIs

[4]. Ministerio de Hacienda. (2022). *Anexos y estructuras.* Disponible en: https://www.hacienda.go.cr/ATV/ComprobanteElectronico/frmAnexosyEstructuras.aspx#

[5]. Oracle. (2022). *How to Write Doc Comments for the Javadoc Tool.* Disponible en: http://www.oracle.com/technetwork/articles/java/index-137868.html

[6]. TutorialsTeacher. (2022). *Create Web API project*. Disponible en: https://www.tutorialsteacher.com/webapi/create-web-api-project

[7]. WrapBootstrap. (2022). *Bootstrap Themes & Templates.* Disponible en: https://wrapbootstrap.com/