INSTITUTO TECNOLÓGICO COLONIA CTC

Obligatorio 2: Algoritmos y Estructuras de Datos

Entregado como requisito para la obtención del título de Analista Programador

Tutor: Carlos Rodríguez
Bargas Maicol,
Pescetto Angelo
Año: 2022

Declaración de auditoría

Nosotros, Maicol Bargas y Angelo Pescetto, Podemos asegurar que:

• La obra fue producida en su totalidad mientras cursaba la materia Diseño y Desarrollo de Aplicaciones.

• Hemos tenido en cuenta las clases dictadas por el Prof. Carlos Rodríguez quién además nos proporcionó material.

• Cuando consultamos el trabajo publicado por otros, lo hemos atribuido con claridad.

• Cuando citamos obras de otros, indicamos las fuentes. Con excepción de estas citas, la obra es enteramente nuestra.

• Ninguna parte de este trabajo ha sido publicada previamente a su entrega.

Índice

Letra del problema

Obligatorio 2 DDA

Fecha: desde el 03 de noviembre al 30 de noviembre de 2022, a las 23:59.

Una empresa de turismo quiere manejar los planes para ofertar a sus clientes y luego poder asociar esos planes con los clientes que los adquieran.

Hay clientes estándar y clientes premium. Los clientes premium son aquellos que ya han adquirido 3 planes de viajes o más. Para ellos, el costo del cuarto plan en adelante tiene un 20% de descuento.

EL SISTEMA DEBE PERMITIR:

1) Mostrar un dashboard administrativo en ambiente Web o Mobile (sencillo) que permita trabajar con planes de viaje y clientes.

2) Dar de alta, eliminar y modificar planes de viaje. Cada uno de ellos tiene un solo **destino** (máx. 20 dígitos alfanuméricos), fecha, **modalidad** (solamente pueden ser aérea, marítima o terrestre), **precio** en USD, **carrusel de fotos** (opcional).

2) dar de alta, eliminar y modificar clientes.

Los clientes tienen **CI** (máx. 8 dígitos, mín. 7 dígitos, sin puntos ni guiones), **nombre** (máx. 30 dígitos alfanuméricos), **apellido** (máx. 30 dígitos alfanuméricos), **email** (máx. 30 dígitos alfanuméricos), **planes comprados** (si no tiene ninguno se deberá indicar con un mensaje que "no tiene planes comprados").

3) contratar o borrarse de un plan (o más) de viaje.

4) Listar los planes (viajes) de un cliente mostrando todos los atributos de cada viaje.

5) Listar el primer viaje que tendrá un cliente después de una fecha especificada.

6) Controlar que no se permita ingresar viajes (planes) con fechas anteriores a la fecha actual.

7) Controles de errores para ingreso de datos.

PASOS A SEGUIR:

1) Analizar la realidad planteada y presentar una solución en Java.

2) Realizar el diagrama de clases con sus métodos y atributos.

3) Realizar el código que una vez ejecutado permita implementar la solución al problema.

A ENTREGAR:

1. Documentación en formato PDF que contenga:
    a. Portada del trabajo con los nombres de los integrantes del equipo.
    b. Letra del problema.
    c. Descripción del análisis y la solución.
    d. Alcances obtenidos a consideración del equipo.

e. Diagrama de clases.
f. Código impreso de las clases.

2. El código de las clases implementadas:
   a. en el Back-End, en un archivo rar llamado *backendObligatorio2-ApellidoDeAlumno1-ApellidoDeAlumno2*
   b. en el Front-End, en un archivo rar llamado *frontendObligatorio2-ApellidoDeAlumno1ApellidoDeAlumno2*
3. La base de datos (preferentemente en MySQL v 5.7.40 o SQL Server v 2012).

Descripción y análisis de la solución

En esta ocasión, se nos pidió desarrollar una aplicación en Spring para la plataforma Java. El proyecto consta de 2 clases, Cliente (ci, nombre, apellido, email) Plan (id, destino, fecha, modalidad, precio) y entre ellas se crea una relación Cliente_Viaje a través de sus identificadores.
El sistema nos permite dar de alta, eliminar y modificar todos los datos de las clases anteriormente nombradas, además de enlazarlas entre sí.
Con esta aplicación levantamos un api rest en el puerto http://localhost:8080. Esta api es la que conecta el front-end con nuestra base de datos la cual persiste toda la información que manejamos. Dicha base de datos será del motor MySQL.
Para la interfaz gráfica nos decidimos por la librería de React Js, la cual nos permite crear una interfaz atractiva al usuario y muy práctica de utilizar en cuanto al manejo de datos dentro de la página web.
Para enviar la información de la página web hacia el back-end utilizamos con la función Fetch a través de los métodos POST y GET.

La interfaz para el usuario es algo sencilla, posee una barra de navegación principal la cual nos permite manejarnos dentro de la aplicación, dentro de ella tenemos las opciones de Inicio, Clientes y Planes.
El inicio simplemente es la pantalla principal donde no encontramos ninguna funcionalidad en especial. Luego tanto en la página de Clientes como en la de Planes encontramos todos los datos referidos a estas entidades. En ambas tenemos una lista con todos los datos de cada uno, y diferentes botones que nos permiten dar de alta, modificar o dar de baja ya sea a Planes como a Clientes. En la página de Clientes encontramos la opción llamada Asignar un Plan, esta nos redirige a otro apartado en el cual nosotros podremos seleccionar que plan será contratado por el cliente que elegimos anteriormente.

Alcances obtenidos

Tras finalizar el proyecto analizando el resultado obtenido valoramos que no hemos concretado todos los objetivos como deseábamos, las principales funcionalidades de la aplicación realizan lo necesario de una forma correcta, el manejo por completo de los datos tanto de Clientes como de los Planes se ejecutan de manera correcta, además de una manera muy intuitiva para el usuario.
Luego encontramos detalles dentro de la interfaz las cuales podrían tener mejora pero se decidió darle prioridad a otros aspectos.
El mayor detalle que podemos encontrar es el manejo de errores, el cual creemos que no es tan correcto como nos planteamos en un principio.

Diagrama de clases

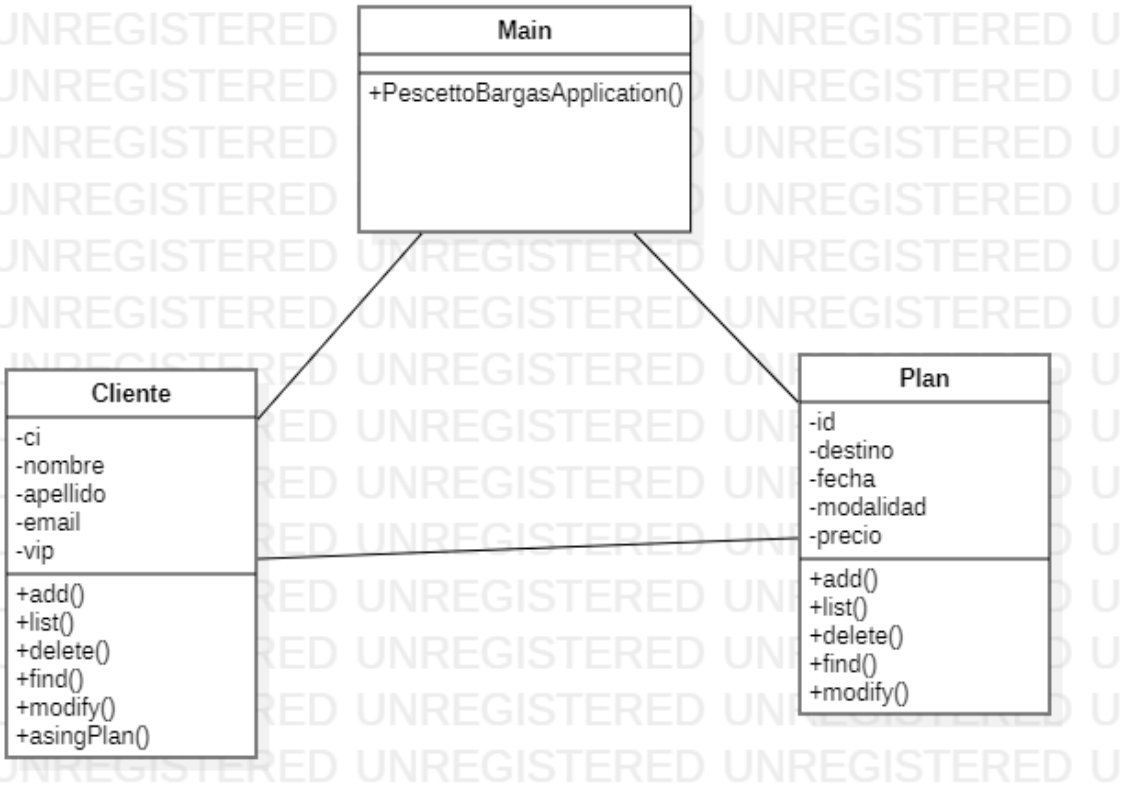

Imagen 1: Diagrama de Clases