



# Turismo Local

Las imágenes vistas del programa son una prueba de diseño, mas no el producto final

# Objetivos Generales y Específicos

Universidad  
Industrial de  
Santander



## Objetivos generales:

- Desarrollar un programa capaz de crear planes turísticos personalizados para los usuarios.
- Facilitar la organización y planificación de viajes turísticos de manera eficiente.
- Mejorar la experiencia del usuario al planificar y realizar viajes turísticos.


## Objetivos específicos:

- Recolectar información sobre destinos turísticos y servicios disponibles en El Socorro.
- Analizar los intereses y preferencias de los usuarios para crear planes turísticos personalizados.
- Permitir la selección y reserva de servicios turísticos como vuelos, alojamiento, transportes y actividades en destino.
- Generar un itinerario detallado de viaje con horarios, direcciones y contactos útiles.
- Incluir herramientas de seguimiento y actualización en tiempo real para ajustar los planes turísticos en caso de cambios inesperados.





# 1.

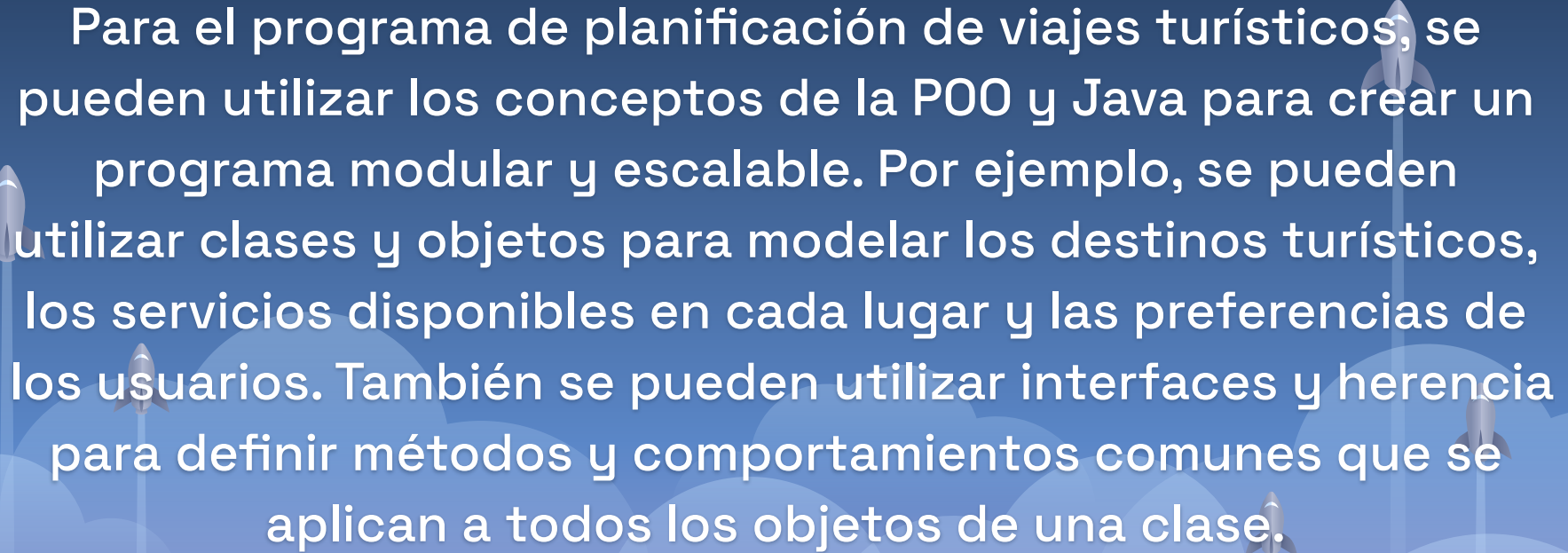
# POO Y JAVA



La Programación Orientada a Objetos (POO) es un paradigma de programación que se basa en la idea de que todo en el mundo real es un objeto. Los objetos tienen atributos (datos) y métodos (comportamientos), y se comunican entre sí para lograr un objetivo común. En la POO, se modela el mundo real en términos de objetos, y se utilizan estos objetos para crear programas.



Java es un lenguaje de programación orientado a objetos que se utiliza ampliamente en el desarrollo de aplicaciones empresariales, juegos, aplicaciones web y móviles, entre otros. Java es un lenguaje de programación de alto nivel, que se ejecuta en una máquina virtual (JVM) que garantiza la portabilidad del código. Además, Java es un lenguaje seguro, ya que cuenta con un sistema de seguridad incorporado y es resistente a los errores de memoria.



Para el programa de planificación de viajes turísticos, se pueden utilizar los conceptos de la POO y Java para crear un programa modular y escalable. Por ejemplo, se pueden utilizar clases y objetos para modelar los destinos turísticos, los servicios disponibles en cada lugar y las preferencias de los usuarios. También se pueden utilizar interfaces y herencia para definir métodos y comportamientos comunes que se aplican a todos los objetos de una clase.

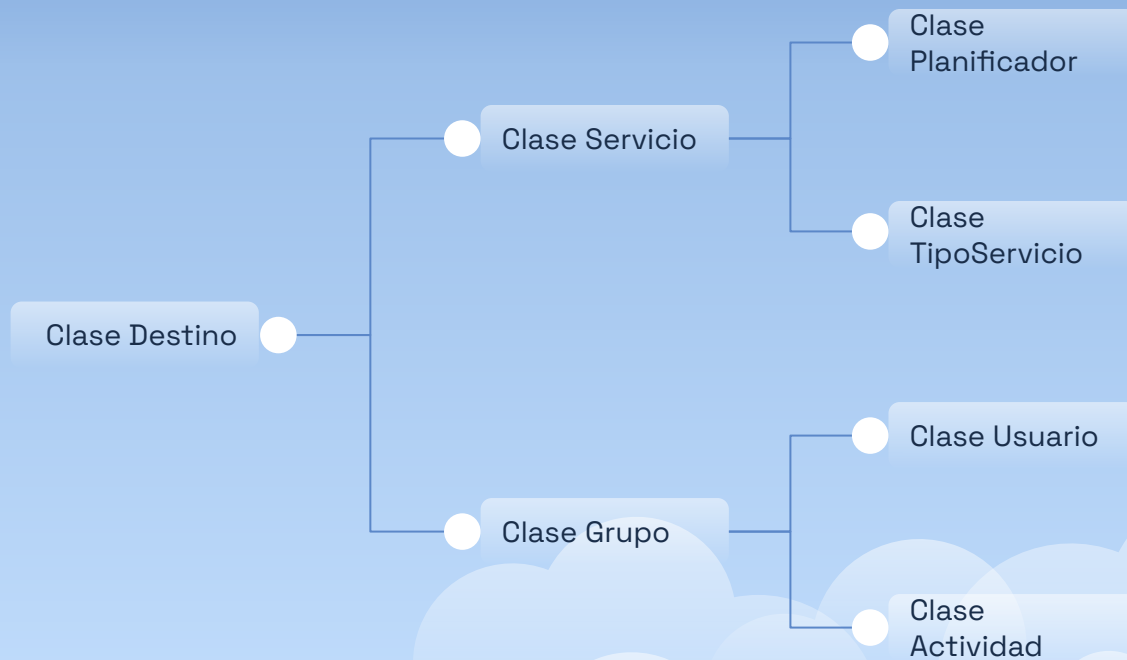


# ¿Por qué?

Un programa de planificación de viajes turísticos puede ayudar a hacer que la planificación y organización de un viaje sea más fácil y accesible para todos los interesados en el turismo. Los usuarios podrían ingresar sus preferencias de viaje, y el programa utilizará esa información para crear planes turísticos completos y detallados que se adapten a las necesidades y gustos de cada viajero.



# Esquema de clases



# Ejemplo Login de la interfaz



The image shows a web browser window with the title "Acceso al Sistema". The background is blue. On the left, there are two input fields: "Usuario:" with the text "jcarlosad7" and "Password:" with masked characters "\*\*\*\*\*". To the right of these fields is an illustration of a person with a padlock, symbolizing security. At the bottom, there are two buttons: "Ingresar" (with a login icon) and "Salir" (with a home icon). A small, disabled button is also visible to the right of the "Salir" button.

(Se usará una pequeña base de datos con MySQL y XAMPP)

# Ejemplo de la interfaz general

Registro de Reservas

Habitación:

401

Cliente:

Juan Carlos Arcila

Trabajador:

Ana Díaz

Tipo Reserva:

Alquiler

Fecha Reserva:

02/06/2014

Fecha Ingreso:

03/06/2014

Fecha Salida:

04/06/2014

Costo:

150.00

Estado Reserva:

Pagada

Nuevo

Editar

Cancelar

Listado de Reservas

Listado de Habitaciones

Buscar

Buscar

Número	Piso	Descripción	Caracterist...	Precio	Estado	Tipo habit...
401	3	Habitación	Habitación	120.00	Disponible	Individual
213	2	Habitación	Habitación	25.00	Disponible	Individual

Total Registros 2

Sali...	Costo	Estado
6-04	150.00	Pagada

Total Registros 1



# Librería Java JDBC

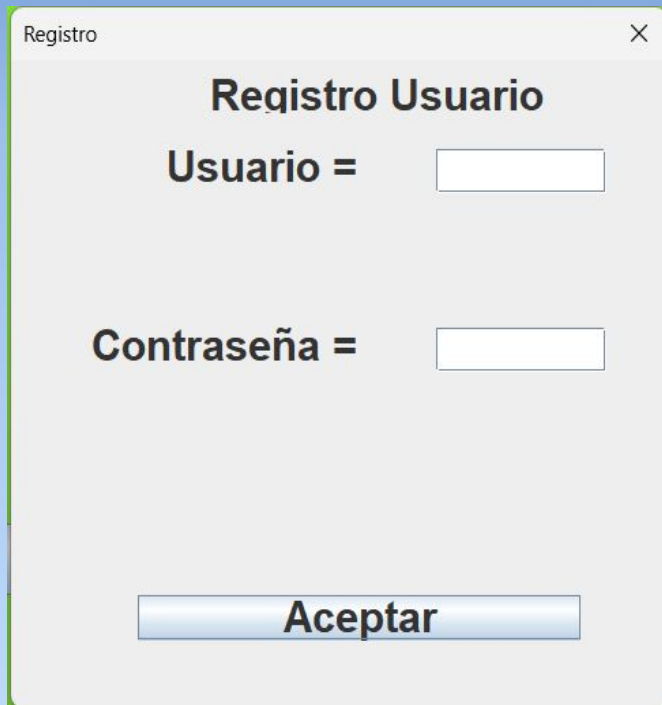
La librería Java JDBC nos permite realizar cualquier tipo de operación sobre una base de datos, ya sean consultas, inserciones, borrados, etc. Una de estas operaciones que podemos hacer es crear una base de datos en Java.



En resumen, la POO y Java pueden ser utilizados para crear un programa de planificación de viajes turísticos que sea modular, escalable y seguro. Estos conceptos permiten la creación de objetos, la definición de métodos y comportamientos comunes, y la utilización de librerías útiles para facilitar el desarrollo del programa.

En pocas palabras, el programa de planificación de viajes turísticos se convierte en una herramienta valiosa para los viajeros que buscan una experiencia de viaje personalizada y sin complicaciones, lo que hace que sea una necesidad importante para aquellos que desean disfrutar de sus viajes al máximo.

# Funcionamiento del registro de usuarios



Registro

**Registro Usuario**

Usuario =

Contraseña =

**Aceptar**

- Para el registro de usuario decidimos usar “FILEREADER”, para leer un archivo “.txt”, que es el encargado de contener cada usuario con su respectiva contraseña

```
public boolean verificarLogin(String usuario, String Contraseña)
{
    boolean resultado = false;
    FileReader reader = null;
    BufferedReader bufferedReader = null;
    try
    {
        File archivo = new File(pathname:"Login.txt");
        reader = new FileReader(archivo);
        bufferedReader = new BufferedReader(reader);
        String linea;
        while ((linea = bufferedReader.readLine()) != null)
        {
            String[] credenciales = linea.split(regex:"");
            if (credenciales[0].equals(usuario) && credenciales[1].equals(Contraseña))
            {
                resultado = true;
                break;
            }
        }
    }
}
```

Otro problema con el que nos encontramos al momento de usar un archivo de texto para almacenar usuarios y contraseña, es el que usar para poder editar este archivo de texto con los datos que se ingresen en el login y que luego la librería anteriormente mencionada pueda leer de manera correcta estos datos ya sobreescritos en el archivo de texto.

```
public boolean registrarLogin(String usuario, String Contraseña)
{
    boolean resultado = false;
    FileWriter writer = null;
    try {
        File archivo = new File(pathname:"Login.txt");
        writer = new FileWriter(archivo, append:true);
        writer.write(usuario + ":" + Contraseña + "\n");
        resultado = true;
    } catch (IOException e) {
        e.printStackTrace();
    } finally {
        try {
            if (writer != null) {
                writer.close();
            }
        } catch (IOException ex) {
            ex.printStackTrace();
        }
    }
    return resultado;
}
```

Para resolver esto usamos una librería familiar a la ya mencionada "FILEREADER" y es que la librería que usamos es "FILEWRITER" esta es la encargada de según lo que se ingrese en la pestaña de registro, edita y organiza secuencialmente los dos string ingresados para que sea entendible para el lector del archivo de texto

Una vez dentro del programa se solicitará al usuario lo básico a tener en cuenta para poder formar una guía de turismo equilibrada de acuerdo a los gustos y preferencias del usuario en cuestión.

**Datos cliente**

Cual es tu presupuesto?	<input type="text" value="0"/>	Dime un gusto tuyo	<input type="text" value="sisa"/>
Cuando llegaste?	<input type="text" value="01/01/2023"/>	Cuando te vas?	<input type="text" value="01/11/2023"/>

Visto en código sería algo como esto:

```
public PanelOperaciones()
{
    setLayout(null);
    setBackground(Color.decode(nmi:"#ebd234"));

    bCrear = new JButton(text:"Crear");
    bCrear.setFont(new Font(name:"Arial", BOLD, size:12));
    bCrear.setBounds(x:80, y:20, width:90, height:20);
    bCrear.setBackground(Color.decode(nmi:"#21eb5d"));
    add(bCrear);
    bCrear.setActionCommand(actionCommand:"crear");

    bBorrar = new JButton(text:"Borrar");
    bBorrar.setFont(new Font(name:"Arial", BOLD, size:12));
    bBorrar.setBounds(x:80, y:100, width:90, height:20);
    bBorrar.setBackground(Color.decode(nmi:"#eb2e21"));
    add(bBorrar);
    bBorrar.setActionCommand(actionCommand:"borrar");

    bCerrarSesion = new JButton(text:"Cerrar Sesión");
    bCerrarSesion.setFont(new Font(name:"Arial", BOLD, size:12));
    bCerrarSesion.setBounds(x:210, y:100, width:140, height:20);
    bCerrarSesion.setBackground(Color.decode(nmi:"#eb2e21"));
    add(bCerrarSesion);
    bCerrarSesion.setActionCommand(actionCommand:"cerrarSesion");

    bDialogoTabla = new JButton(text:"Tabla Eventos");
    bDialogoTabla.setFont(new Font(name:"Arial", BOLD, size:12));
    bDialogoTabla.setBounds(x:210, y:20, width:140, height:20);
    bDialogoTabla.setBackground(Color.decode(nmi:"#21eb5d"));
    add(bDialogoTabla);
    bDialogoTabla.setActionCommand(actionCommand:"dialogoTabla");

    TitledBorder borde = BorderFactory.createTitledBorder(title:"Botones");
    borde.setTitleColor(Color.decode(nmi:"#7542f5"));
    setBorder(borde);
}
```