

## LISTADO HUESPEDES

### Clase ListadoHuespedes:

- Es una ventana de tipo JFrame, donde se muestra una tabla con los datos de los huéspedes.
- Se utiliza para listar, agregar, editar y eliminar registros de huéspedes.

### Atributos:

- modeloTabla: Un DefaultTableModel que maneja los datos de la tabla (de tipo JTable).
- huespedDAO: Un objeto de la clase HuespedDAO que interactúa con la base de datos para obtener y modificar los datos de los huéspedes.

### Método cargarDatos:

- Este método carga los datos de los huéspedes en la tabla desde la base de datos a través del objeto huespedDAO.
- Limpia la tabla y luego agrega cada fila con los datos de los huéspedes recuperados.

### Método refrescarDatos:

- Llama a cargarDatos para recargar y actualizar los datos mostrados en la tabla.

### Botón "Atrás" (btnAtras):

- Cuando se presiona este botón, se cierra la ventana actual y se abre la ventana principal del menú (MenuPrincipal).

### Botón "Editar" (btnEditar):

- Permite seleccionar un huésped de la tabla y, al hacer clic en "Editar", abre un formulario de edición (FormularioHotel), pasando el ID del huésped seleccionado para realizar la actualización.

### Botón "Agregar" (btnAgregar):

- Abre un formulario (FormularioHotel) para agregar un nuevo huésped, donde no se pasa ID (es cero) ya que es una nueva entrada.

### Botón "Eliminar" (btnEliminar):

- Elimina el huésped seleccionado de la tabla y la base de datos, mediante el uso del método `huespedDAO.delete()`.
- Después de eliminar, recarga los datos en la tabla.

**Interacción con la base de datos:**

- La clase `HuespedDAO` es responsable de manejar las operaciones de base de datos (obtener todos los huéspedes, eliminar uno, etc.).

**Método main:**

- Inicializa la interfaz gráfica y hace visible la ventana de `ListadoHuespedes`.

## FORMULARIO HOTEL

### Clase FormularioHotel:

- Extiende JDialog.
- Esta clase es responsable de manejar la inserción, actualización y eliminación de los datos de los huéspedes.

### Atributos:

- huesped: Objeto de la clase Huesped que se utilizará para representar y manipular los datos de un huésped.
- Mode: String que indica el modo de la operación (puede ser "INS" para insertar, "UPD" para actualizar, o "DLT" para eliminar).
- id\_huesped: ID del huésped que se está editando o eliminando.
- parent: Referencia a la ventana ListadoHuespedes, que es la ventana principal donde se listan los huéspedes.
- dao: Objeto HuespedDAO, responsable de las operaciones de base de datos (insertar, actualizar, eliminar).

### Constructor FormularioHotel:

- Inicializa la ventana del formulario, recibiendo el modo (Mode) y el id\_huesped que se editará.
- Si el modo es "UPD" o "DLT", obtiene los datos del huésped de la base de datos y los muestra en los campos de texto del formulario.
- Según el modo, se cambia el texto del botón de operación a "Insertar", "Modificar" o "Borrar".

### Método validar():

- Verifica si el campo txtNombre está vacío y muestra un mensaje de error si es el caso. Devuelve false si no está lleno.

### Método initComponents():

- Este método genera los componentes visuales de la interfaz gráfica (como los campos de texto y botones). No debe ser modificado.

### Métodos de eventos de componentes:

- txtIDActionPerformed, txtNombreActionPerformed, txtFechaActionPerformed: Son métodos generados automáticamente para manejar eventos en los campos de texto, pero no contienen lógica en este caso.

- **btnOperacionActionPerformed**: Este es el manejador de eventos para el botón de operación (Insertar/Actualizar/Eliminar). Dependiendo del modo (Mode), realiza lo siguiente:
  - **Modo "INS"**: Valida los datos y luego inserta un nuevo huésped en la base de datos.
  - **Modo "UPD"**: Valida los datos y luego actualiza el huésped existente con los nuevos valores.
  - **Modo "DLT"**: Aunque la lógica para eliminar no está completa en este código, sería el paso donde se eliminaría un huésped.
- **Validaciones**:
  - El teléfono solo puede contener números (se valida con una expresión regular).
  - La fecha se valida con el formato yyyy-MM-dd.

#### **Método btnCancelarActionPerformed:**

- Este método simplemente cierra la ventana del formulario cuando se presiona el botón "Cancelar".

#### **Interacción con la base de datos:**

- Se utiliza la clase HuespedDAO para realizar las operaciones de base de datos, como insertar (dao.insert(newHuesped)), actualizar (dao.update(huesped)), etc.

#### **Método main:**

- Inicializa la aplicación con la ventana principal MenuPrincipal. Este método no se usa directamente en el formulario, pero es una manera de iniciar la aplicación.

## RESERVAS

### Funcionalidades Clave:

- **Visualización de Reservas:** Muestra las reservas existentes en una tabla JTable, utilizando un DefaultTableModel para cargar y actualizar los datos.
- **Llenado de JComboBox:** Se rellenan los combos de selección (cbxHuesped y cbxHabitacion) con los datos de la base de datos, específicamente los id\_huesped y id\_habitacion de la tabla reservas.
- **Acciones de los Botones:**
  - o **Agregar:** Abre un formulario para agregar una nueva reserva (modal).
  - o **Eliminar:** Elimina la reserva seleccionada en la tabla.
  - o **Editar:** Permite editar la reserva seleccionada.
  - o **Atrás:** Regresa al menú principal.

### Explicación de los Métodos Principales:

- **cargarDatos():** Carga las reservas desde la base de datos y las muestra en la tabla.
- **llenarCBHuesped() y llenarCBHabitacion():** Se encargan de llenar los combos de selección con los IDs de los huéspedes y las habitaciones, respectivamente, extraídos de la base de datos.
- **btnAgregarActionPerformed():** Abre un formulario para crear una nueva reserva.
- **btnEditarActionPerformed():** Permite editar una reserva existente. Si no se selecciona ninguna fila, muestra un mensaje de error.
- **btnEliminarActionPerformed():** Elimina una reserva seleccionada, y luego actualiza la vista.

## FORMULARIO RESERVAS

La clase FormularioReservas extiende de JDialog y permite a los usuarios realizar operaciones de **insertar**, **actualizar** o **eliminar** una reserva. La información de la reserva incluye:

- **Fecha de entrada y Fecha de salida.**
- **Estado** de la reserva (confirmada o cancelada).
- **Huésped y Habitación** asociados a la reserva.

### Llenado de los JComboBox:

- Los comboboxes cbxHuesped y cbxHabitacion se llenan con los id\_huesped e id\_habitacion de las reservas existentes en la base de datos. Esto es realizado por los métodos llenarCBHuesped y llenarCBHabitacion.

### Inicialización de los datos del formulario:

- Si el modo es "UPD" (actualizar) o "DLT" (eliminar), los campos del formulario se rellenan con los datos de la reserva seleccionada. Esto se hace recuperando la información de la base de datos mediante ReservaDAO.

### Operación de Insertar, Actualizar y Eliminar:

- Cuando se hace clic en el botón btnOperacion, se realiza la operación correspondiente dependiendo del modo (INS, UPD o DLT).
- Se convierten las fechas seleccionadas a java.sql.Date para su almacenamiento en la base de datos.
- Dependiendo de la operación (insertar, actualizar o eliminar), el objeto Reserva se guarda, modifica o elimina utilizando el ReservaDAO.

### Botones:

- El botón btnOperacion cambia de texto dependiendo de si estamos en modo insertar (INS), actualizar (UPD) o eliminar (DLT).
- El botón btnCancelar cierra el formulario sin realizar cambios.

### Validación y formato de fechas:

- Las fechas de entrada y salida se manejan mediante un JDateChooser de la librería toedter. Estas fechas se convierten a formato yyyy-MM-dd antes de ser utilizadas en las operaciones con la base de datos.

## MENU PRINCIPAL

- **MenuPrincipal():** Este es el constructor de la clase. Llama al método `initComponents()`, que inicializa los componentes gráficos (botones, etiquetas, paneles) de la ventana.
- **initComponents():** Este método genera el diseño de la interfaz gráfica de la aplicación. Configura los componentes como los botones, etiquetas y paneles y asigna eventos a los botones, como las acciones a realizar cuando se hace clic en ellos.
- **btnSalirActionPerformed(ActionEvent evt):** Este método es ejecutado cuando el usuario hace clic en el botón "Salir". El método termina la aplicación cerrando la ventana principal (`System.exit(0)`), lo que termina el programa y también cierra la ventana del menú.
- **btnHuespedesActionPerformed(ActionEvent evt):** Este método es ejecutado cuando el usuario hace clic en el botón "Huespedes". Abre una nueva ventana llamada `ListadoHuespedes`, que podría mostrar la lista de los huéspedes, y cierra la ventana actual del menú principal (`this.dispose()`).
- **btnReservasActionPerformed(ActionEvent evt):** Este método es ejecutado cuando el usuario hace clic en el botón "Reservas". Abre una nueva ventana llamada `Reservas`, que probablemente gestiona las reservas de los huéspedes, y cierra la ventana actual del menú principal.
- **main(String[] args):** Este es el punto de entrada principal de la aplicación. Ejecuta el método `run()`, que inicializa y muestra la ventana del `MenuPrincipal`.

## LOGIN

Funcionalidades Clave:

- **Autenticación de Usuario:** Permite a los usuarios iniciar sesión en la aplicación utilizando su nombre de usuario y contraseña.
- **Validación de Credenciales:** Verifica si el nombre de usuario y la contraseña proporcionados coinciden con los almacenados en la base de datos.
- **\*\*Acciones del Botón:**
  - **Login:** Realiza la validación de las credenciales y, si son correctas, permite acceder al sistema.

Explicación de los Métodos Principales:

- **jButton1ActionPerformed ():** Este método se ejecuta cuando el usuario hace clic en el botón de "Login". Valida si los campos de usuario y contraseña están completos y, si es así, consulta a la base de datos para verificar si las credenciales son correctas. Si el login es exitoso, se redirige al usuario al menú principal de la aplicación. Si no, se muestra un mensaje de error.
- **main():** Este es el método principal que arranca la aplicación y muestra la ventana de login cuando se ejecuta el programa.



## Formulario de Habitaciones

### Funcionalidades Clave:

#### - Formulario de Inserción, Actualización y Eliminación:

Este formulario permite gestionar las habitaciones en el sistema, con tres operaciones básicas: insertar, actualizar o eliminar.

#### - Validación de Datos:

Antes de insertar o actualizar los datos de una habitación, se verifica que el nombre de la habitación no esté vacío. Si lo está, se muestra un mensaje de error.

#### - Acciones de los Botones:

- Insertar: Si el formulario está en modo de inserción ("INS"), permite agregar una nueva habitación.

- Modificar: Si el formulario está en modo de actualización ("UPD"), permite modificar los datos de la habitación seleccionada.

- Eliminar: Si el formulario está en modo de eliminación ("DLT"), permite eliminar la habitación seleccionada.

- Cancelar: Cierra el formulario sin realizar ninguna acción.

### Explicación de los Métodos Principales:

#### - FormularioHabitacion()

Este es el constructor del formulario. Recibe los parámetros necesarios como el modo de operación ("INS", "UPD" o "DLT") y el `id\_huesped` de la habitación que se va a modificar o eliminar. Dependiendo del modo, llena los campos del formulario con los datos de la habitación seleccionada. Si el modo es de inserción, los campos estarán vacíos para ingresar nuevos datos. Además, el botón de operación cambiará según el modo: Insertar, Modificar o Borrar.

#### - validar()

Este método verifica que el campo `txtNombre` (el nombre de la habitación) no esté vacío. Si está vacío, muestra un mensaje de error y devuelve `false`,

impidiendo que se continúe con la operación. Si el campo es válido, devuelve `true`.

#### **- btnOperacionActionPerformed()**

Este método se ejecuta cuando el usuario hace clic en el botón de operación (Insertar, Modificar o Borrar).

- Si el modo es "INS" (Insertar), crea una nueva instancia de la clase `Habitación` y le asigna los valores ingresados en los campos del formulario. Luego, usa el DAO para insertar la nueva habitación en la base de datos. Si la operación es exitosa, actualiza la vista en el formulario principal y cierra el formulario.

- Si el modo es "UPD" (Actualizar), actualiza los datos de la habitación seleccionada con los valores ingresados y realiza la operación de actualización en la base de datos. Después, actualiza la vista y cierra el formulario.

#### **- btnCancelarActionPerformed()**

Este método se ejecuta cuando el usuario hace clic en el botón "Cancelar". Cierra el formulario sin realizar ninguna operación.

### **Otros Detalles Importantes:**

#### **- Componentes de la Interfaz Gráfica:**

El formulario tiene campos de texto para ingresar los datos de la habitación: `txtID` (ID de la habitación), `txtNombre` (tipo de la habitación), `txtEmail` (precio de la habitación), `txtTelefono` (capacidad de la habitación), y `txtFecha` (estado de la habitación). El botón "Operación" realiza la acción seleccionada (Insertar, Modificar o Eliminar), y el botón "Cancelar" cierra el formulario sin guardar cambios.

#### **- Manejo del Modo del Formulario:**

Dependiendo del modo de operación ("INS", "UPD" o "DLT"), se modifica el texto del botón de operación para indicar al usuario qué acción se va a realizar.

## Listado habitaciones

Este es el código de una ventana de tipo JFrame que muestra un listado de habitaciones en una tabla. Permite realizar operaciones de Agregar, Editar y Eliminar habitaciones. También incluye un botón para regresar al menú principal.

### Descripción de Funcionalidades y Componentes

#### 1. Interfaz Gráfica:

- Tabla de Habitaciones ( `tablaHabitacion` ): Muestra un listado de habitaciones con las siguientes columnas: "ID", "Tipo", "Precio", "Capacidad", y "Estado".
- Botones:
  - Agregar: Abre un formulario para agregar una nueva habitación.
  - Editar: Abre un formulario para editar los detalles de una habitación seleccionada.
  - Eliminar: Elimina la habitación seleccionada de la lista.
  - Atrás: Regresa al menú principal.

#### 2. Estructura de la Clase:

- La clase extiende ` javax.swing.JFrame ` y contiene una tabla que se llena con los datos de las habitaciones desde la base de datos.
- Utiliza un modelo de tabla ( ` DefaultTableModel ` ) para gestionar y mostrar las habitaciones.

#### 3. Métodos Clave:

- ` cargarDatos() `
- Este método se encarga de obtener todas las habitaciones de la base de datos mediante el ` HabitacionDAO ` (Data Access Object) y llenar la tabla con la información.

- Se limpia la tabla antes de agregar los nuevos datos (``modeloTabla.setRowCount(0)``).
- `refrescarDatos()`:
  - Este método se llama después de realizar una operación de Agregar, Editar o Eliminar para actualizar la tabla.

#### **- Eventos de los Botones:**

- `btnEditarActionPerformed()`: Si el usuario selecciona una fila de la tabla y hace clic en "Editar", se obtiene el ``id_huesped`` de la fila seleccionada, y se abre el formulario de ``FormularioHabitacion`` en modo de Actualización (UPD).
- `btnAgregarActionPerformed()`: Abre el formulario de `FormularioHabitacion` en modo de **\*\*Inserción (INS)\*\*** para agregar una nueva habitación.
- `btnAtrasActionPerformed()`: Cierra la ventana actual y regresa al menú principal.
- `btnEliminarActionPerformed()`: Si el usuario selecciona una fila y hace clic en "Eliminar", se obtiene el ``id_huesped`` de la habitación seleccionada y se elimina mediante el `HabitacionDAO`. Luego, refresca la tabla.

#### **4. DAO (Data Access Object) - HabitacionDAO:**

- `getAll()`: Este método en el DAO recupera todas las habitaciones de la base de datos.
- `delete(int id_huesped)`: Elimina la habitación cuyo ``id_huesped`` es pasado como parámetro. Si la operación tiene éxito, la tabla se actualiza.

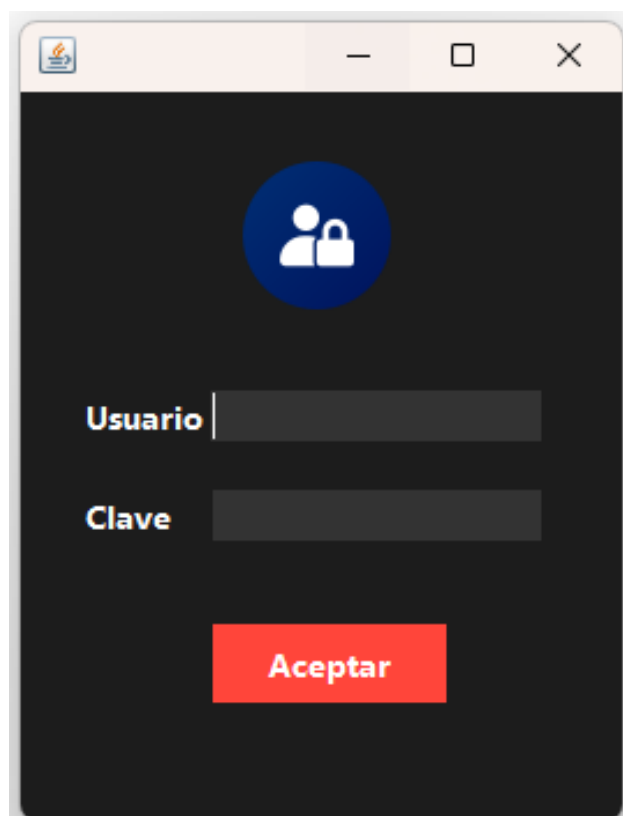
#### **5. Diseño de la Interfaz:**

- El diseño incluye un panel con la tabla, un título y los botones de acción. La tabla ocupa la mayor parte del espacio, mientras que los botones se encuentran en la parte inferior.
- `btnAtras` está en la parte superior derecha, permitiendo regresar al menú principal, y los otros botones (`btnAgregar`, `btnEditar`, `btnEliminar`) están alineados horizontalmente debajo de la tabla.

### **Comportamiento del Programa:**

- Al iniciar la aplicación, se carga el listado de habitaciones desde la base de datos y se muestra en la tabla.
- Los usuarios pueden agregar una nueva habitación con el botón Agregar, que abre un formulario de inserción. Pueden editar una habitación seleccionada con Editarlo eliminarla con Eliminar.
- Después de realizar una operación (agregar, editar, eliminar), la tabla se actualiza automáticamente para reflejar los cambios.


## ANEXOS



A login form with a dark background. At the top center is a blue circular icon containing a white silhouette of a person with a padlock. Below this are two input fields: the first is labeled 'Usuario' and the second is labeled 'Clave'. At the bottom center is a red button with the text 'Aceptar'.



A main menu with a dark background. At the top left is a red circular icon with a white house symbol. To its right is the text 'Menú Principal'. Below this are four red buttons stacked vertically: the first has a white person icon and the text 'Huespedes'; the second has a white house icon with a plus sign and the text 'Habitaciones'; the third has a white calendar icon and the text 'Reservas'; and the fourth is a grey button with a white door icon and the text 'Salir'.



## Huespedes

Atrás

ID	Nombre	E-mail	Teléfono	Fecha registro
1	Pedro García	pedroqarcia@e...	98856325	2025-02-13
2	Laura Ruiz	lauraruiz@email...	94452368	2025-02-13
3	Andrés Martínez	andresmartinez...	89956345	2025-02-13
4	Elena Pérez T	elenaperez@em...	95426355	2025-02-13
5	Carlos Gómez	carlos@email.com	99783261	2025-02-13
11	Anthony Padilla	anthony@gmail....	96623589	2025-04-03

Agregar
Editar
Eliminar



## Huespedes

ID:


Nombre:

E-mail:

Teléfono:

Fecha registro:

Insertar
Cancelar



## Huespedes

ID:

Nombre:

E-mail:


Teléfono:

Fecha registro:

Modificar
Cancelar




## Habitaciones


Atrás

ID	Tipo	Precio	Capacidad	Estado
1	Doble	850.0	2	disponible
2	Suite	950.0	4	ocupada
3	Doble	1000.0	1	mantenimiento
4	Suite	550.0	4	disponible
5	Doble	900.0	2	ocupada
7	Sencilla	800.0	1	disponible

Agregar
Editar
Eliminar



## Habitación

ID:

Tipo

Precio

Capacidad

Estado

Insertar
Cancelar



## Habitación

ID:

Tipo


Precio

Capacidad


Estado

Modificar
Cancelar





## Reservas


**Atrás**

Id Reserva	Id Huesped	Id Habitación	Fecha entrada	Fecha salida	Estado
1	4	1	2025-06-01	2025-06-05	confirmada
2	5	4	2025-03-02	2025-03-12	confirmada
3	1	4	2025-08-20	2025-08-25	confirmada
4	2	5	2025-09-05	2025-09-10	cancelada
5	3	2	2025-10-01	2025-10-05	cancelada
6	1	2	2025-03-01	2025-03-05	confirmada
13	1	5	2025-04-03	2025-04-10	confirmada
14	1	1	2025-04-08	2025-04-10	confirmada

Agregar

Editar

Eliminar



## Reservas

ID Reserva:

Fecha de entrada:  

Fecha de salida:  

Estado:  

Huesped:  

Habitación:  

Insertar

Cancelar



## Reservas

ID Reserva:

Fecha de entrada:  

Fecha de salida:  

Estado:  

Huesped:  

Habitación:  

Modificar

Cancelar