

## FORMATO DE INFORME DE PRÁCTICA DE LABORATORIO / TALLERES / CENTROS DE SIMULACIÓN – PARA ESTUDIANTES

**CARRERA:** Computación

**ASIGNATURA:** Programación Aplicada

**NRO. PRÁCTICA:**

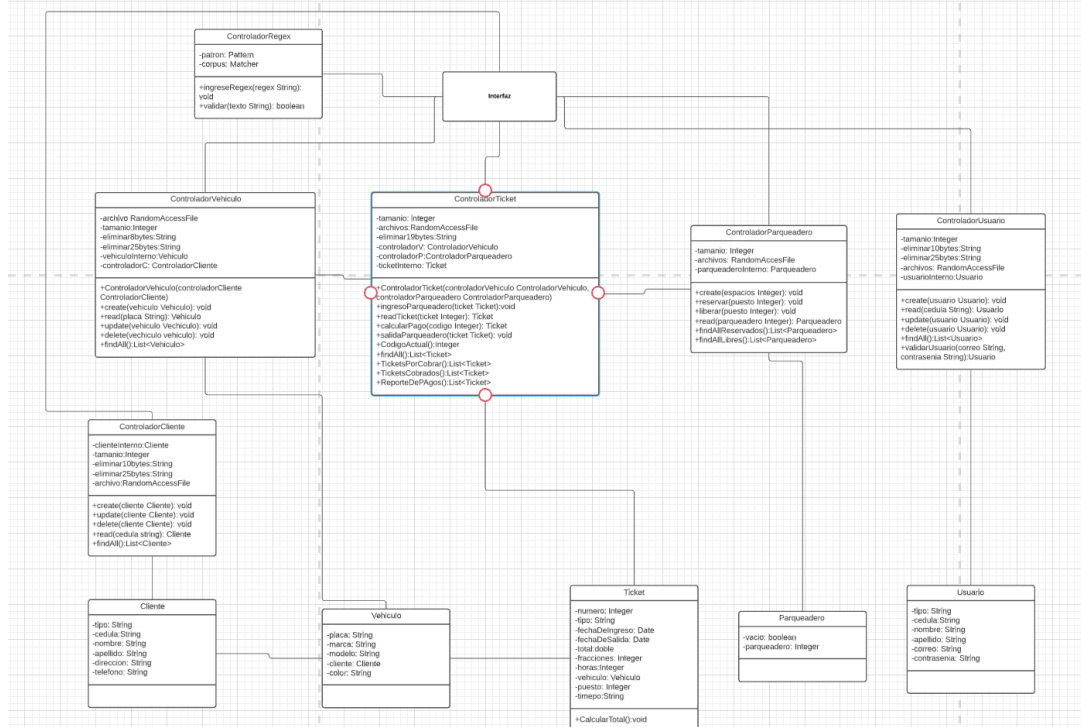
1.1

**TÍTULO PRÁCTICA:** Proyecto Integrador Interciclo

### OBJETIVO ALCANZADO:

Reforzar los conocimientos adquiridos en clase sobre la programación aplicada (POO, Interfaz grafica, etc.) en un contexto real.

### ACTIVIDADES DESARROLLADAS




1. Se pidió desarrollar una aplicación que utilice métodos y patrones enseñados en el interciclo. A continuación, se describirá las clases y métodos utilizados para el programa de parqueadero que se pidió desarrollar. Primeramente, se desarrollaron tres paquetes para nuestra aplicación, utilizando la arquitectura de MVC.

- Ec.edu.ups.modelo
- Ec.edu.ups.controlador
- Ec.edu.ups.vista

2. Ec.edu.ups.modelo

En este paquete se crearon todos los modelos que interactuaran con los controladores.

- Cliente
- Parqueadero

	<b>VICERRECTORADO DOCENTE</b>	<b>Código:</b> GUIA-PRL-001
	CONSEJO ACADÉMICO	<b>Aprobación:</b> 2016/04/06
<b>Formato:</b> Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

- Ticket
- Usuario
- Vehículo
- Singleton

### 3. Singleton

### 4. Clase Cliente

En este modelo se crearon diversos atributos de un cliente al momento de crear un ticket.

Atributos:

- private String tipo;
- private String cedula;
- private String nombre;
- private String apellido;
- private String direccion;
- private String telefono;

En esta clase se crearon dos constructores. El primer constructor tiene instanciado todos los atributos de la clase Cliente, y el segundo constructor es un constructor vacío. De la misma manera se crearon todos sus métodos getters y setters para todos sus atributos. Por ultimo, se sobre escribieron los métodos hashCode y equals, para permitir dar un valor único al atributo de cedula.

### 5. Clase Parqueadero

Esta clase se creo con la intención de simular el espacio de un lote en un parqueadero. En el cual solamente tiene dos atributos.

Atributos:

- private boolean vacio;
- private int parqueadero;


En esta clase se crearon dos constructores. El primer constructor tiene instanciado todos los atributos de la clase Parqueadero, y el segundo constructor es un constructor vacío. De la misma manera se crearon todos sus métodos getters y setters para todos sus atributos. Por ultimo, se sobre escribieron los métodos hashCode y equals, para permitir dar un valor único al numero del parqueadero.

### 6. Clase Ticket

Esta clase tiene la intención de simular un ticket en el mundo real. Y tiene varias relaciones con muchas clases ya que un ticket en la vida real tiene varios datos en si.

Atributos:

- private int numero;
- private String tipo;
- private Date fechaDeIngreso;
- private Date fechaDeSalida;
- private double total;
- private int fracciones;

	<b>VICERRECTORADO DOCENTE</b>	<b>Código:</b> GUIA-PRL-001
	CONSEJO ACADÉMICO	<b>Aprobación:</b> 2016/04/06
<b>Formato:</b> Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

- private int minutos;
- private Vehiculo vehiculo;
- private int puesto;
- private String tiempo;

En esta clase se crearon dos constructores. El primer constructor tiene instanciado todos los atributos con excepción de dos atributos de la clase Ticket, y el segundo constructor es un constructor vacío. De la misma manera se crearon todos sus métodos getters y setters para todos sus atributos. Por ultimo, se sobre escribieron los métodos hashCode y equals, para permitir dar un valor único al numero del ticket.

Métodos:

De la misma manera se crearon varios métodos apartes en la misma clase para calcular el valor a pagar de los servicios prestados.

- **CalcularTotal:** Este método tiene un objeto de tipo string en su parámetro. Lo que realiza este método es calcular el valor a pagar por los servicios prestados. Para calcular el precio a pagar primero recibe la fecha de ingreso y la fecha de salida. Después calcula cuantas horas ha estado el usuario si es que cumple con el horario establecido el usuario pagara \$0,60 por la hora, \$3,00 por día, \$18,00 por semana y \$68,00 por mes. En caso de que se haya expirado el tiempo de servicio del cliente se le multara con el 30% por cada hora excedido.

## 7. Clase Usuario

Esta Clase simula la autoridad que va a tener el Usuario al poder acceder a diferentes atributos que los clientes no van a poder acceder.

Atributos:

- private String tipo;
- private String cedula;
- private String nombre;
- private String apellido;
- private String correo;
- private String contrasenia;


En esta clase se crearon dos constructores. El primer constructor tiene instanciado todos los atributos de la clase Usuario, y el segundo constructor es un constructor vacío. De la misma manera se crearon todos sus métodos getters y setters para todos sus atributos. Por ultimo, se sobre escribieron los métodos hashCode y equals, para permitir dar un valor único a la cedula del usuario.

## 8. Clase Vehículo

Esta clase simula un auto en la vida real. En el caso de un parqueadero se debe tener un registro de los autos que ingresan en caso de algún reclamo o problema.

Atributos:

- private String placa;
- private String marca;
- private String modelo;

	<b>VICERRECTORADO DOCENTE</b>	<b>Código:</b> GUIA-PRL-001
	CONSEJO ACADÉMICO	<b>Aprobación:</b> 2016/04/06
<b>Formato:</b> Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

- private String color;
- private Cliente cliente;

En esta clase se crearon dos constructores. El primer constructor tiene instanciado todos los atributos de la clase vehículo, y el segundo constructor es un constructor vacío. De la misma manera se crearon todos sus métodos getters y setters para todos sus atributos. Por ultimo, se sobre escribieron los métodos hashCode y equals, para permitir dar un valor único a la placa del vehículo.

#### 9. Paquete Ec.edu.ups.controlador

Este paquete contiene todos los controladores que se utilizaran para las interfaces.

- ControladorCliente
- ControladorParquadero
- ControladorRegex
- ControladorTicket
- ControladorUsuario
- ControladorVehiculo

#### 10. ControladorCliente

Este controlador maneja todos los métodos para un cliente. El tamaño calculado de bytes por cada cliente creado es de 117 bytes.


Atributos:

- private Cliente clienteInterno;
- private int tamaño;
- private String eliminar10bytes;
- private String eliminar25bytes;
- private RandomAccessFile archivo;

En el constructor de esta clase se inicializa el archivo en donde se guardarán los datos, y se inicializa algunos atributos de la clase.

Métodos:

- **Créate:** Este método recibe un objeto de tipo cliente. La tarea que realiza este método es crear un nuevo cliente.
- **Update:** Este método recibe un objeto de tipo cliente. La tarea que realiza este método es actualizar los datos de un cliente. La manera en la que realiza esta tarea es por medio de la cedula, es decir compara la cedula del cliente en los parámetros con la cedula de los clientes en la basa de datos.
- **Delete:** Este método recibe un objeto de tipo cliente. La tarea que realiza este método es eliminar los datos de un cliente. La manera en la que realiza esta tarea es por medio de la cedula, es decir compara la cedula del cliente en los parámetros con la cedula de los clientes en la basa de datos.
- **Read:** Este método recibe un objeto de tipo string. La tarea que realiza este método es la búsqueda de un cliente por medio de la cedula. Una vez que hay encontrado al cliente retorna ese cliente al sistema para poder visualizarlo.

	<b>VICERRECTORADO DOCENTE</b>	<b>Código:</b> GUIA-PRL-001
	CONSEJO ACADÉMICO	<b>Aprobación:</b> 2016/04/06
<b>Formato:</b> Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

- findAll: Este método la única tarea que realiza esa permitir la visualización de todos los clientes que puedan existir en la base de datos.

### 11. ControladorParqueadero

Este controlador maneja todos los métodos para un lote. El tamaño calculado de bytes por cada lote creado es de 5 bytes.

Atributos:

- private int tamaño;
- private RandomAccessFile archivos;
- private Parqueadero parqueaderoInterno;

En el constructor de esta clase se inicializa el archivo en donde se guardarán los datos, y se inicializa algunos atributos de la clase.

Métodos:

- Crear: Este método recibe un objeto de tipo entero. La tarea que realiza este método es crear un nuevo lote.
- reservar: Este método recibe un objeto de tipo entero. La tarea que realiza este método es reservar un parqueadero.
- liberar: Este método recibe un objeto de tipo entero. La tarea que realiza este método es liberar un parqueadero.
- Read: Este método recibe un objeto de tipo entero. La tarea que realiza este método es la búsqueda de un parqueadero por medio del número de lote. Una vez que hay encontrado el parqueadero retorna ese parqueadero al sistema para poder visualizarlo.
- findAllReservados: Este método la única tarea que realiza esa permitir la visualización de todos los parqueaderos que estén reservados.
- findAllLibres: Este método la única tarea que realiza esa permitir la visualización de todos los parqueaderos que no estén reservados.


### 12. ControladorTicket

Este controlador maneja todos los métodos para un ticket. El tamaño calculado de bytes por cada ticket creado es de 90 bytes.

Atributos:

- private int tamaño;
- private RandomAccessFile archivos;
- private String eliminar19bytes;
- private ControladorVehiculo controladorV;
- private ControladorParqueadero controladorP;
- private Ticket TicketInterno;

El constructor de esta clase recibe dos constructores en su parámetro también se inicializa el archivo en donde se guardarán los datos, y se inicializa algunos atributos de la clase.

	<b>VICERRECTORADO DOCENTE</b>	<b>Código:</b> GUIA-PRL-001
	CONSEJO ACADÉMICO	<b>Aprobación:</b> 2016/04/06
<b>Formato:</b> Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

#### Métodos:

- **ingresarParqueadero:** Este método recibe un ticket en su parámetro. Este método realiza la tarea de crear un ticket. A la misma vez que se crea el ticket también llama al controlador parqueadero para que reserve el parqueadero que tiene el ticket.
- **readTicket:** Este método recibe un objeto de tipo entero en su parámetro. Este método realiza la tarea de buscar un ticket por medio de su numero asignado. Una vez que haya encontrado el ticket con el mismo numero lo retorna para visualizarlo o sino retornara null.
- **cacluarPago:** este método recibe un objeto de tipo entero y un objeto de tipo date en su parámetro. Este método tiene la tarea de calcular el valor a pagar por los servicios prestado del parqueadero.
- **salidaParqueadero:** este método recibe un objeto de tipo ticket en su parámetro. Este método realiza la tarea de actualizar los datos del ticket. De esa manera ahora el ticket se encontrará en una base de datos donde ya se han cobrado los tickets.
- **codigoActual:** Este método lo único que hace es retornar el numero del ultimo ticket registrado.
- **findAll:** Este método retorna todo el listado de tickets que existen en la basa de datos.
- **TicketsPorCobrar:** Este método retorna todo el listado de tickets que faltan por cobrar que existen en la basa de datos.
- **TicketsCobrados:** Este método retorna todo el listado de tickets cobrados que existen en la basa de datos.

### 13. ControladorUsuario

Este controlador maneja todos los métodos para un usuario. El tamaño calculado de bytes por cada usuario creado es de 117 bytes.


#### Atributos:

- private int tamano;
- private String eliminar10bytes;
- private String eliminar25bytes;
- private RandomAccessFile archivos;
- private Usuario usuarioInterno;

El constructor inicializa el archivo en donde se guardarán los datos, y se inicializa algunos atributos de la clase.

#### Métodos:

- **Créate:** Este método recibe un objeto de tipo usuario en su parámetro. Este método realiza la tarea de crear un nuevo usuario en la base de datos.
- **Read:** Este método recibe un objeto de tipo string en su parámetro. Este método realiza la tarea de la búsqueda de un usuario por medio de su cedula. Si es que lo encuentra retorna ese usuario para visualizarlo caso contrario retorna un nulo.
- **Update:** Este método recibe un objeto de tipo usuario en su parámetro. Este método realiza la tarea de actualizar un usuario. Primero busca al usuario por medio de la cedula en la base de datos y luego proceda a actualizar los datos.

	<b>VICERRECTORADO DOCENTE</b>	<b>Código:</b> GUIA-PRL-001
	CONSEJO ACADÉMICO	<b>Aprobación:</b> 2016/04/06
<b>Formato:</b> Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

- Delete: Este método recibe un objeto de tipo usuario en su parámetro. Este método realiza la tarea de eliminar un usuario y lo hace mediante la búsqueda de cedula. Cuando lo encuentra elimina todos los datos con espacios vacíos.
- findAll: Este método realiza la tarea de retornar todos los usuarios dentro de la base de datos.
- validarUsuario: Este método realiza la tarea de validar si es que el usuario existe caso contrario no le permitirá acceder a los atributos de el administrador.

#### 14. ControladorRegex

Este controlador se utilizará para la validación de varios campos dentro del programa.

Atributos:

- private Pattern patron;
- private Matcher corpus;

Esta clase tiene creado todos sus métodos getters y setters para todos sus atributos.

Métodos:

- ingreseRegex: Este método recibe en su parámetro un objeto de tipo string. La tarea que realiza este método es la creación de un patrón por medio de una expresión de java.
- validar: Este método recibe en su parámetro un objeto de tipo string. La tarea que realiza este método es validar el texto ingresado. Si es que cumple con el patrón establecido devuelve true caso contrario retornara false.

#### 15. Controlador Vehículo

Este controlador maneja todos los métodos para un vehículo. El tamaño calculado de bytes por cada vehículo creado es de 93 bytes.


Atributos:

- private RandomAccessFile archivo;
- private int tamaño;
- private String eliminar8bytes;
- private String eliminar25bytes;
- private Vehiculo vehiculoInterno;
- private ControladorCliente controladorC;

El constructor inicializa el archivo en donde se guardarán los datos, y se inicializa algunos atributos de la clase.

Métodos:

- Créate: Este método recibe un objeto de tipo vehículo en su parámetro. Este método realiza la tarea de crear un nuevo vehículo con los datos recibidos
- Update: Este método recibe un objeto de tipo string en su parámetro. Este método actualiza los datos de un vehículo por medio de su placa. Es decir, realiza una búsqueda por medio de la placa y cuando lo encuentra reemplaza los datos con los nuevos datos recibidos.

	<b>VICERRECTORADO DOCENTE</b>	<b>Código:</b> GUIA-PRL-001
	CONSEJO ACADÉMICO	<b>Aprobación:</b> 2016/04/06
<b>Formato:</b> Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

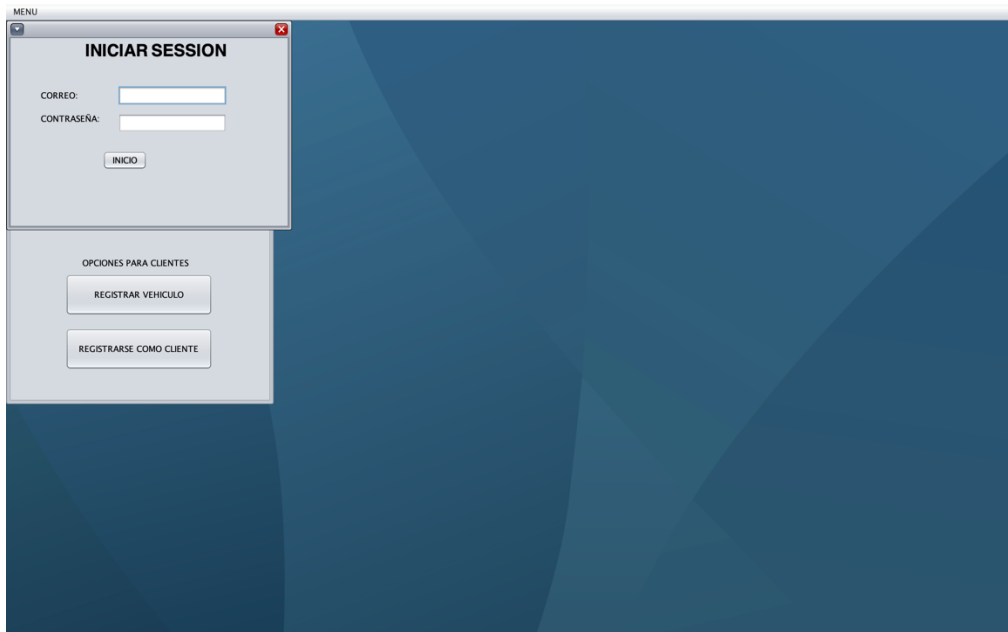
- Read: Este método recibe un objeto de tipo vehículo en su parámetro. Este método realiza la tarea de buscar un vehículo por medio de la placa. Una vez encontrada lo retorna para poder visualizarla.
- Delete: Este método recibe un objeto de tipo vehículo en su parámetro. Este método elimina los datos de un vehículo por medio de su placa. Es decir, realiza una búsqueda por medio de la placa y cuando lo encuentra elimina los datos de ese vehículo.
- findAll: Este método solamente realiza la tarea de poder visualizar todos los vehículos registrados en la base de datos.

## 16. Ec.edu.ups.vista

Este paquete contiene todas las interfaces en las cuales el usuario interactuará.

- IniciarSession
- Inicio
- MenuPrincipal
- RegistrarAdministrador
- RegistrarCliente
- RegistrarVehiculo
- ReservarParqueadero
- Tipo
- VentanaAdministrador


## 17. IniciarSession

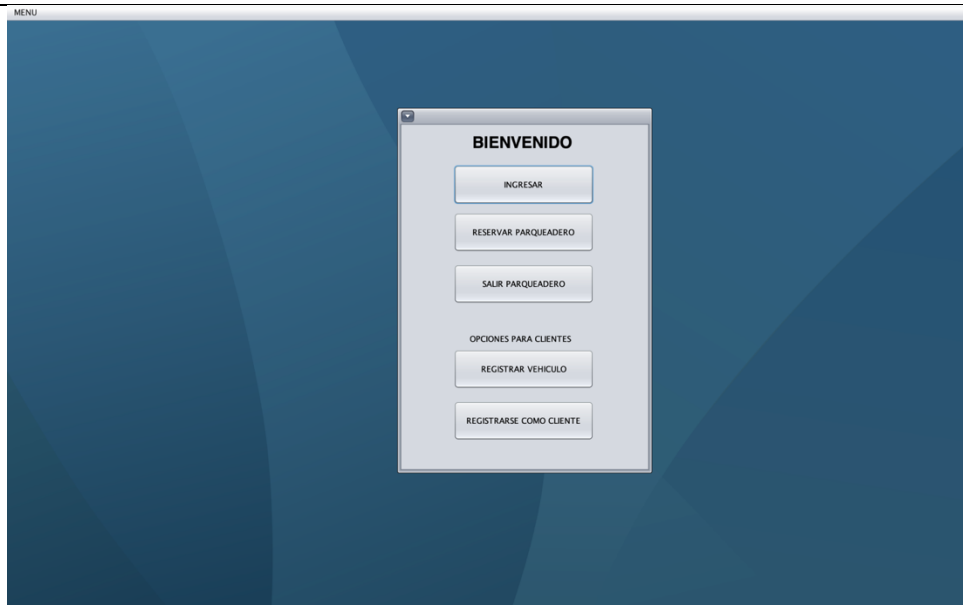


Esta ventana solamente pueden ingresar los administrados, caso contrario los clientes no podrán visualizar la ventana de los administradores

## 18. Inicio



	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		




Esta ventana es el menú principal. En esta ventana se pueden ver que existen diversas opciones, en las cuales el usuario puede elegir el tipo de servicio que desea del parqueadero.

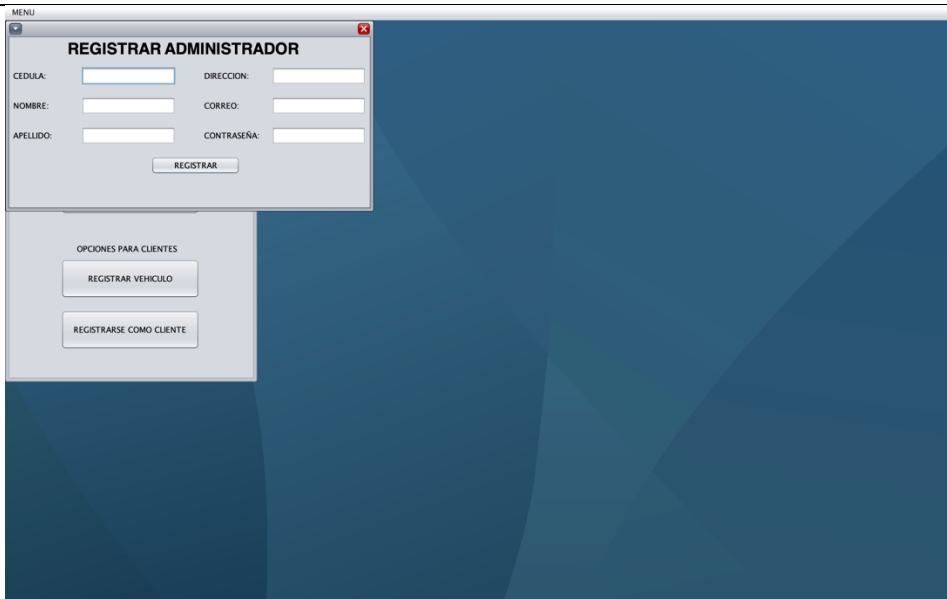
## 19. MenuPrincipal



Este es el frame que va a contener todas las ventanas.

## 20. RegistrarAdministrador

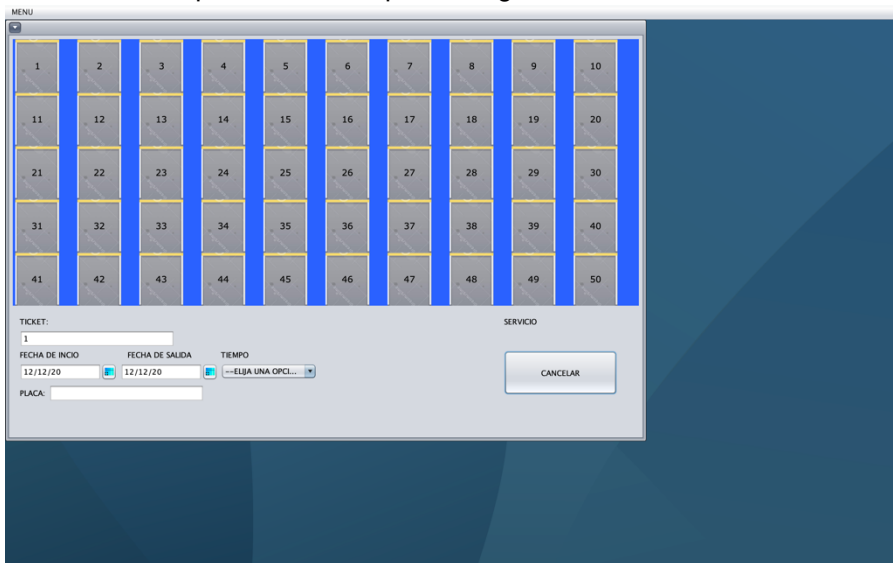
	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		




Esta ventana permite que se pueda registrar el administrador. Al ingresar el correo y la cedula se ha creado validaciones para que el administrador ingrese los datos correctamente. Una vez corregido las validaciones se crea un administrador.

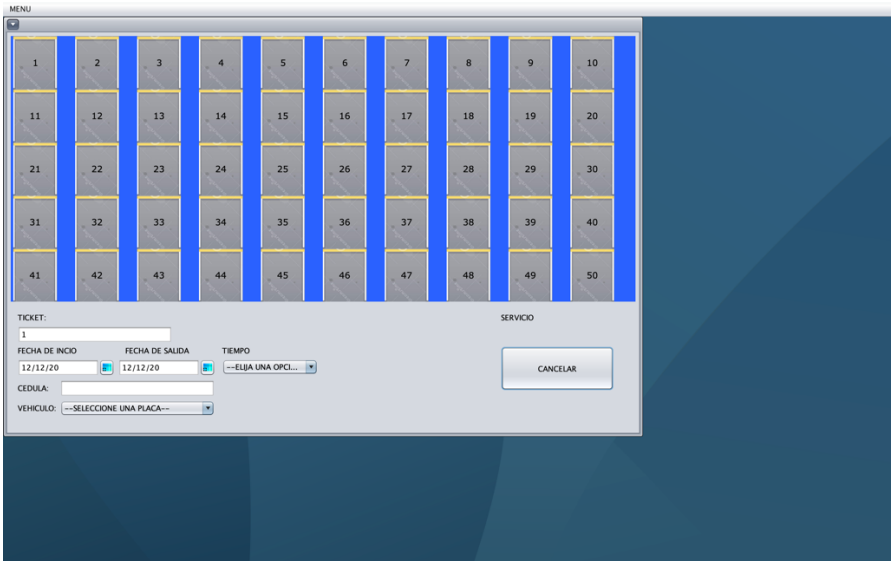
## 21. ReservarParqueadero

Existen cuatro tipos de ventana para el registrado de vehículos

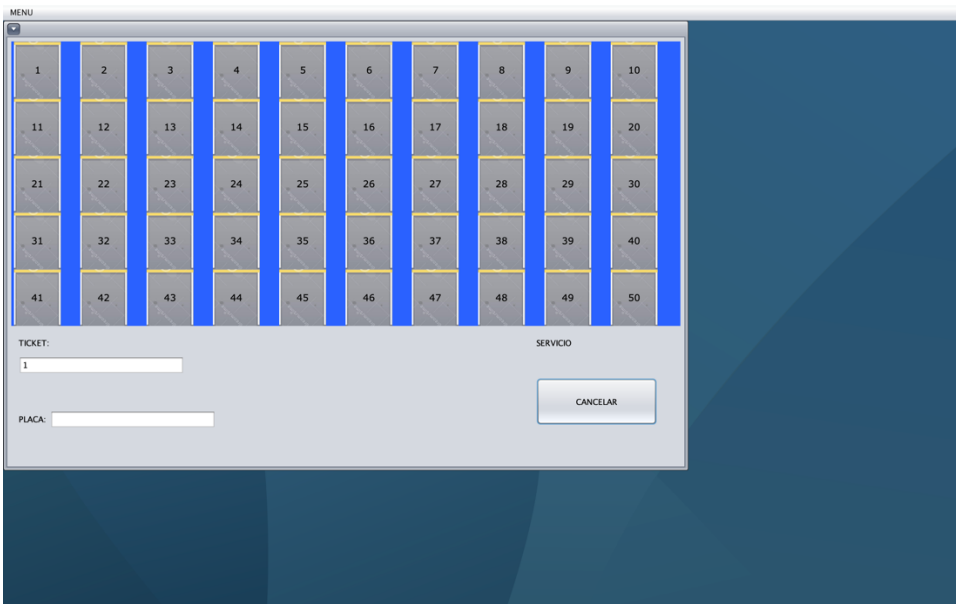


Esta ventana es la opción de reservado de parqueadero con datos finales. El usuario al elegir la opción de reservado y consumidor final el programa visualizara todos los atributos para que el cliente pueda reservar el auto por un cierto tiempo y por ultimo el programa visualizara la caja de texto de placa. De esta manera al crear el ticket para el parqueadero se registrará solamente el numero de ticket, el numero del parqueadero y la placa del vehículo.


	<b>VICERRECTORADO DOCENTE</b>	<b>Código:</b> GUIA-PRL-001
	CONSEJO ACADÉMICO	<b>Aprobación:</b> 2016/04/06
<b>Formato:</b> Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

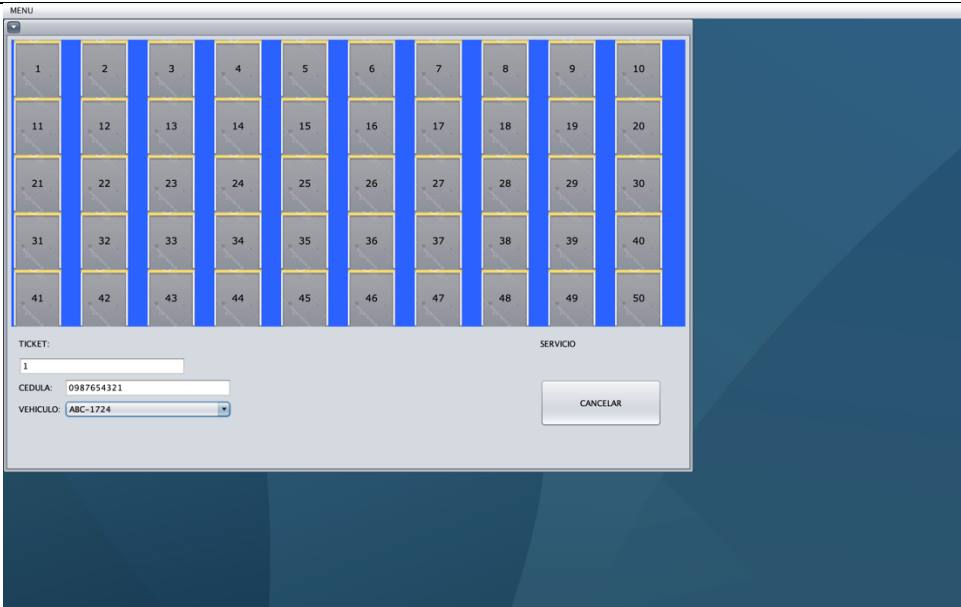


Esta ventana es la opción de reservado de parqueadero con datos. El usuario al elegir la opción de reservado y cliente el programa visualizara todos los atributos para que el cliente pueda reservar el auto por un cierto tiempo con los datos del usuario. De esta manera al crear el ticket para el parqueadero se registrará todos los atributos del ticket, el numero del parqueadero y los datos del cliente.



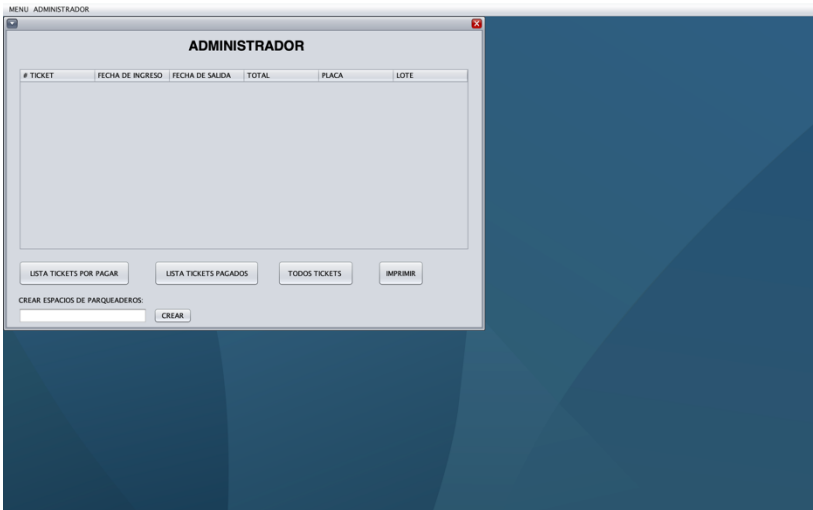
Esta ventana es la opción de ingreso de parqueadero por hora como consumidor final. El usuario al elegir la opción ingreso por hora el programa visualizara solo el área de ingreso de placa. De esta manera al crear el ticket para el parqueadero se registrará todos los atributos del ticket, el numero del parqueadero y la placa de vehículo.

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		




Esta ventana es la opción de ingreso de parqueadero por hora como cliente. El usuario al elegir la opción ingreso por hora el programa visualizara la caja de texto “cedula” para que el cliente pueda ingresar su cedula y el programa pueda cargar sus datos. De esta manera al crear el ticket para el parqueadero se registrará todos los atributos del ticket, el numero del parqueadero y los datos del cliente.

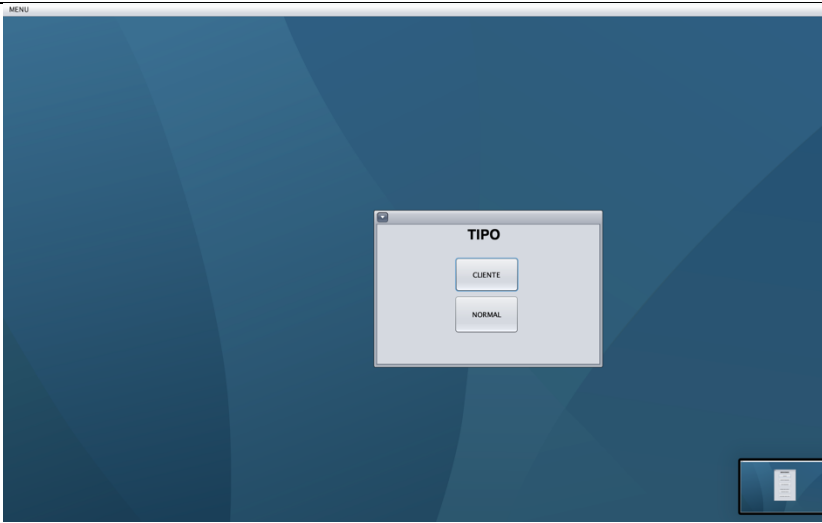
## 22. VentanaAdministrador



Esta ventana visualiza todas las tareas que un administrador puede acceder. El administrador puede ver los tickets cobrados, los tickets que faltan por cobrar y puede visualizar todos los tickets. De la misma manera el administrador puede ingresar el numero de parqueaderos que van a existir dentro del sistema.

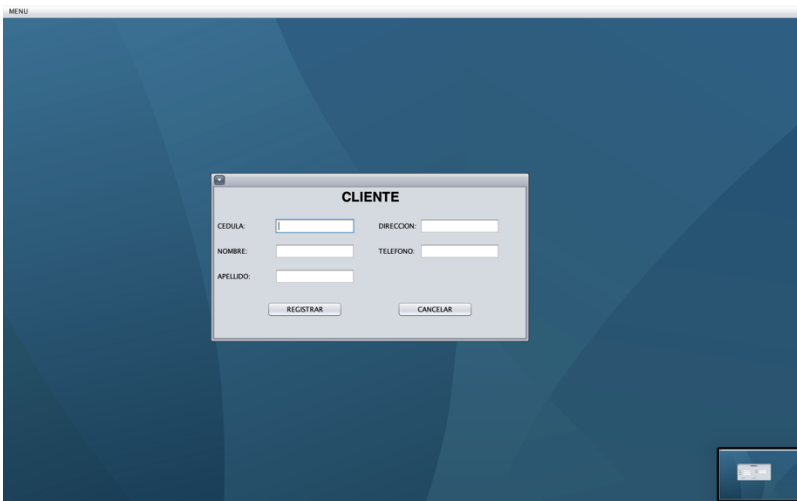
## 23. Tipo

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		



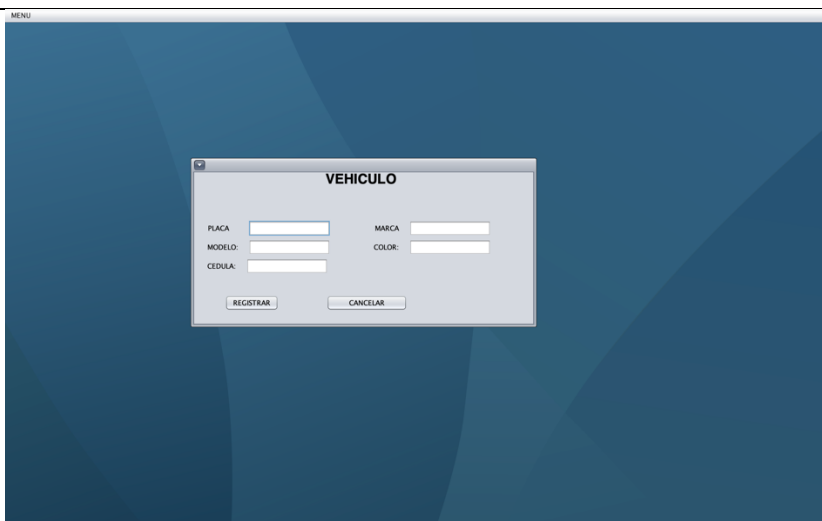
Esta ventana es un intermediario entre la ventana de inicio y ReservarParqueadero. Este permite que el cliente pueda ingresar como un cliente registrado o como un consumidor final.

## 24. RegistrarCliente



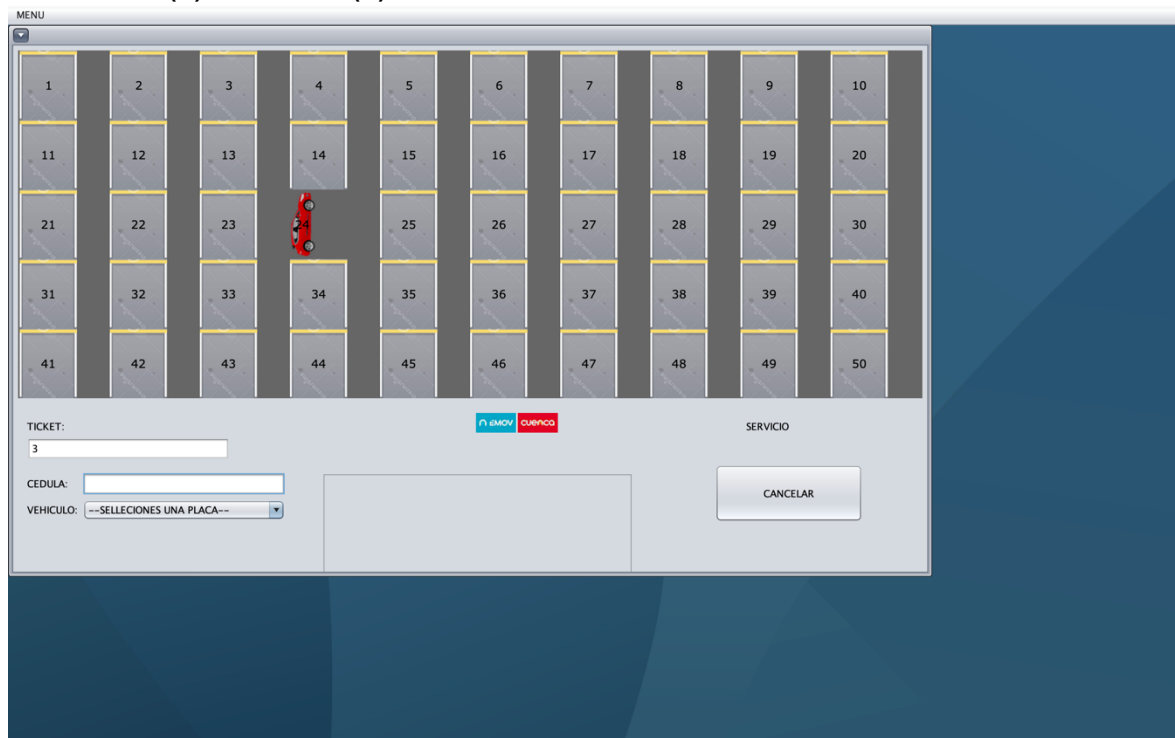
Esta ventana permite que el cliente pueda registrarse dentro del sistema si es que desea, caso contrario solamente presiona el botón de “cancelar” y el programa le envía al menú principal.


## 25. RegistrarVehiculo



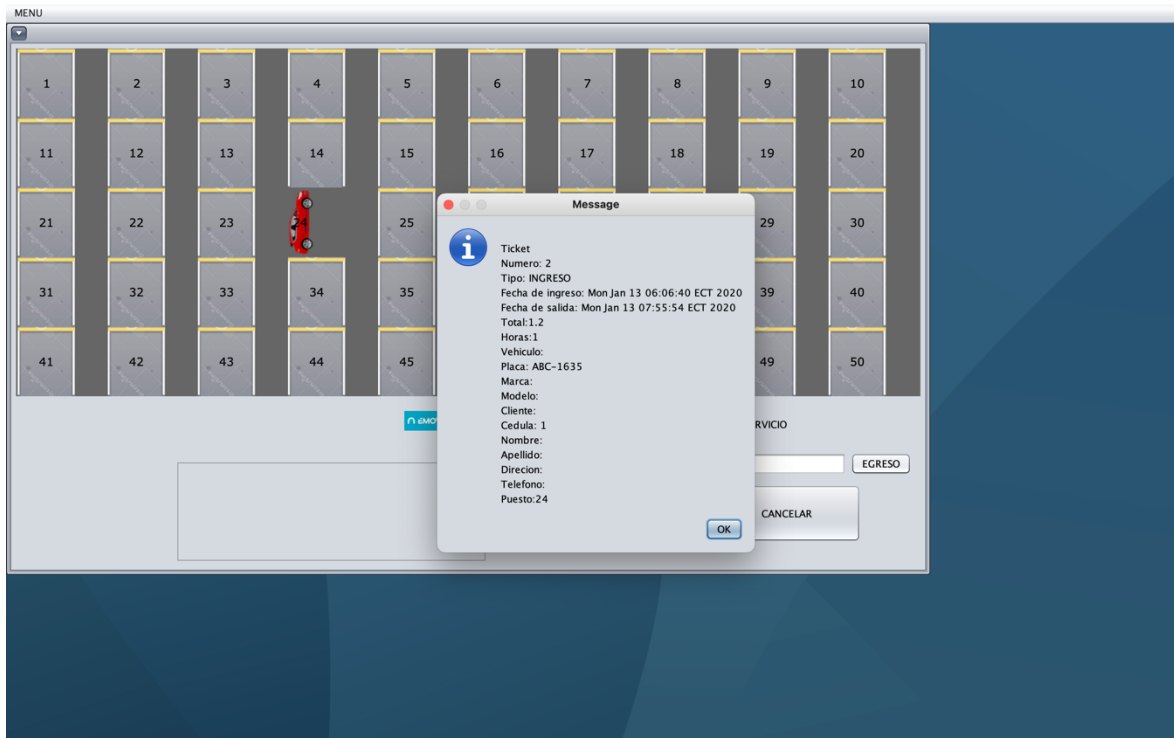
Esta ventana permite que solamente un cliente registrado pueda registrar su vehículo ya que esto significaría que el cliente ingresa al parqueadero frecuentemente. De la misma manera si el usuario desea cancelar la acción solamente se debe presionar el botón de cancelado y el programa le envía al inicio.

### RESULTADO(S) OBTENIDO(S):



	<b>VICERRECTORADO DOCENTE</b>	<b>Código:</b> GUIA-PRL-001
	CONSEJO ACADÉMICO	<b>Aprobación:</b> 2016/04/06
<b>Formato:</b> Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

Esta ventana representa la reservación de un parqueadero. Como se puede observar al reservar un parqueadero la imagen de un parqueadero cambia a un vehículo. Si es que algún usuario ingresa en el área ocupada una ventana de error aparecerá diciendo “ESPACIO OCUPADO” y le permitirá al usuario elegir otro puesto disponible.



El cliente al salir del parqueadero una ventana con la factura aparecerá. En ella se puede ver el costo que tiene que pagar por los servicios prestados.

**CONCLUSIONES:** En conclusión, se debe tener un entendimiento de como expresar y utilizar expresiones de java. Los métodos regex son de inmensa utilidad para validar campos de texto al momento de ingresar datos. Por ultimo se debe tener entendimiento de patrones de diseño que puedan facilitar la arquitectura de un programa ya que estos patrones han tenido éxito anteriormente.

#### RECOMENDACIONES:

**Nombre de estudiante:** Denys Dutan

**Firma de estudiante:** 