

**Sistema Integral**

**Manual Técnico del sistema**

Jahaziel Aarón Aguilera Castillo

Departamento de Tecnologías de la Información

Destinos Parhikuni S.A. de C.V.

2022-2023

# Indice

[Indice 2](#_Toc30152)

[Introducción 3](#_Toc25614)

[Glosario 4](#_Toc231)

[Base de datos 5](#_Toc25741)

[Tablas 5](#_Toc30585)

[Eventos 6](#_Toc32712)

[Funciones 6](#_Toc29587)

[Procedimientos 7](#_Toc16958)

[Triggers 7](#_Toc10119)

[Cliente de Windows 8](#_Toc21889)

[Laravel / PHP 9](#_Toc32000)

# Introducción

En este manual tiene el propósito de ayudar a los trabajadores de DTI a dar mantenimiento al Sistema Integral y apoyar para que el resto de empleados pueda hacer uso del mismo para realizar sus actividades de servicios, venta, almacén, taller, etc. Estos módulos se irán implementando con el tiempo, especificando tanto la parte técnica como su funcionamiento para el usuario final.

El Sistema Integral (SI) es el nuevo sistema de venta, creado como alternativa a SUC, llamado *integral* buscando tener todo concentrado en un sólo sistema y no distintos que pueden tener discrepancias en la información, problemas de comunicación, estar limitados a pedir acceso para hacer actualizaciones y lo que se les ocurra. Surge a partir de un corte de servicio en 2022 y como una necesidad que quienes no conocen de informática ignoran cuánto tiempo y personas se necesitan para registrar toda esta información, aquí sólo se necesita un servidor y algunos segundos.

# Glosario

* **DTI** (Departamento de Tecnologías de la Información): el área encargada de los sistemas, computadoras, programación, bases de datos, servidores, etc.
* **Chófer**: persona cuyo trabajo es conducir, estando al servicio particular de alguien. Parhíkuni no tiene chóferes, tiene conductores.
* **Cliente**: persona que compra nuestros productos o servicios.
* **Conductor**: persona cuyo trabajo es conducir los autobuses.
* **Corrida**: el viaje programado de un autobús.
* **Corrida disponible**: así le llamamos a las corridas ya existentes dentro del sistema, las cuales se toman en cuenta para la venta de boletos.
* **Corrida programada**: así le llamamos a la planeación donde se indica cómo, a partir de estas se generarán las Corridas Disponibles. Especificamos horario, clase de servicio, días de la semana, etc.
* **Franquicia**: Boleto que se le concede sin costo a un empleado.
* **Itinerario**: ruta que sigue una corrida, donde se indican los lugares por donde se pasa. El itinerario se considera el recorrido que se realiza en la corrida y se constituye de Tramos.
* **Oficina**: se refiere a las instalaciones con las que cuenta Destinos Parhíkuni, ya sean oficinas administrativas, puntos de venta o terminales de autobuses.
* **Pasajero**: persona que con su boleto comprado viaja en los autobuses.
* **Persona**: individuo. Se especifica para evitar confundir con usuario, pasajero o similares, lo cual no necesariamente es.
* **Promoción**: Descuentos que se lanzan al público.
* **Servicio** (clases de servicio): las distintas marcas que se utilizan para diferenciar los beneficios que obtendrá el cliente al comprar su boleto: el modelo del autobús, si hay servicio de venta a bordo, si es o no un viaje con escalas, si el autobus es de ruta o rentado, si el autobús cuenta con baño de hombres y mujeres, etc.
* **Servicios** (área): es el área operativa encargada de coordinas las corridas de los autobuses y a los conductores.
* **Terminal**: se puede referir así a un equipo de cómputo utilizado para acceder a nuestro sistema. No confundir con terminal de autobuses.
* **Tramo**: recorrido de un punto A a un punto B. Los tramos conforman itinerarios.

# Base de datos

Se optó por usar MySql

## Tablas

* **Tablas de Laravel**
  + failed\_jobs
  + migrations
  + model\_has\_permissions
  + model\_has\_roles
  + password\_resets
  + permissions
  + personal\_access\_tokens
  + role\_has\_permissions
  + roles
  + Users
* **Tablas propias**
  + autobuses
  + boletos\_vendidos\_estados
  + boletoscancelados
  + boletosdisponibles
  + boletosvendidos
  + boletosvendidos\_descuentos
  + boletosvendidos\_factorpaqueteria
  + boletosvendidos\_promociones
  + boletosvendidos\_tipopaquetes
  + cliente\_windows
  + clientes
  + conductores
  + corridas\_conductores
  + corridas\_disponibles\_historial
  + corridas\_estados
  + corridas\_versiones
  + corridasdisponibles
  + corridasprogramadas
  + descuentos
  + disponibilidad
  + disponibilidadasientos
  + distribucionasientos
  + factorpaqueteria
  + familiares
  + formapagosubtipo
  + formaspago
  + itinerario
  + oficinas
  + origenesdestinos
  + periodos\_vacacionales
  + personas
  + personas\_estados
  + promociones
  + registropasopuntos
  + sesiones
  + tarifastramos
  + tarjetasboletinadas
  + Terminales
  + tipopaquetes
  + Tipopasajero
  + Tiposboletos
  + Tiposervicio
  + Tipospersona
  + Tipoventa
  + Tramos
  + Venta
  + Ventascliente

## Eventos

**crear\_corridasDia**

Este evento se ejecuta cada día y se encarga de consultar las corridas programadas para crear las corridas disponibles. Su funcionamiento se limita a llamar la función corridasPorDia con los parámetros (fechaInicio, posteriores)

**liberar\_apartados**

Se ejecuta cada minuto en busca de los asientos que hayan quedado apartados por más de 15 minutos y liberarlos. Elimina los registros de la tabla disponibilidadasientos que tengan aEstado=’a’

## Funciones

**apartar\_asiento**

* Parámetros:
  + IN\_nCorrida: corridasdisponibles.nNumero
  + IN\_nOrigen: oficinas.nNumero
  + IN\_nDestino: oficinas.nNumero
  + IN\_nAsientos: numero de asiento 1 a X
  + IN\_user: users.nNumero, quién lo está apartando
* Funcionamiento: verifica que el asiento se encuentre disponible según los parámetros dados y de ser posible los aparta en la tabla disponibilidadasientos
* *Retorno INT*: los ids insertados en disponibilidadasientos Ej. “01,02,03,04” o el texto “NO INSERTADO”

**corridasPorDia**

* Parámetros
  + IN\_fIni: fecha de inicio, a partir de qué fecha se empieza a trabajar
  + IN\_dias: a partir de la fecha de inicio cuantos días van a ser procesados
* Funcionamiento: Consulta las corridas programadas y crea las corridas disponibles llamando a la función crear\_corridas\_disponibles, a partir de la fecha de inicio hasta x días. Básicamente es sólo un ciclo.
* Retorno INT: total de registros insertados 0 → X

**crear\_corridas\_disponibles**

* Parámetros
  + IN\_fecha: fecha, un sólo día
* Funcionamiento: Con la fecha dada se buscan las corridas programadas que no tienen sus corridas disponibles y se crearán las últimas.
* Retorno INT: la cantidad de corridas creadas

**crear\_disponibilidad**

* Parámetros
  + IN\_corrida\_dis: corridasdisponibles.nNumero
* Funcionamiento: Toma la corrida dada para insertar en la tabla disponibilidad, la disponibilidad de horarios para esa corrida, es decir, la fecha y hora en que esa corrida estará disponible para cada uno de sus tramos, es importante para el control de las ventas, mostrar horarios y tener menos registros en la tabla de corridasdisponibles.
* Retorno SMALLINT: retorna la cantidad de registros insertados.

**estadoAsientos**

* Parámetros:
  + IN\_cordis: nNumero de corridasdisponibles
  + IN\_origen: nNumero de la tabla oficinas
  + IN\_destino: nNumero de la tabla oficinas
  + IN\_nAsiento: Número de asiento 1-X
  + IN\_estado: estado del asiento, para indicar el estado de un asiento en la tabla relacional disponibilidadasientos, en la columna aEstadoAsiento
    - D = Disponible
    - A = Apartado
    - V = Vendido
    - B = Bloqueado
* Funcionamiento:
* Retorno (text): ids de las inserciones separados por comas.

**insert\_pasajero**

* Parámetros
  + IN\_nVenta
  + IN\_nCorrida
  + IN\_fSalida
  + IN\_hSalida
  + IN\_nOrigen
  + IN\_nDestino
  + IN\_aTipoPasajero
  + IN\_aPasajero
  + IN\_nAsiento
  + IN\_aTipoVenta
  + IN\_nMontoBase
  + IN\_nMontoDescuento
  + IN\_nIva
  + IN\_aEstado
  + IN\_nTerminal
  + IN\_nPromocion
  + IN\_nDescuento
* Funcionamiento: Inserta en la tabla de boletosvendidos, guarda si se aplicó una promoción en boletosvendidos\_promociones
* Retorno TEXT: id del boleto vendido boletosvendidos.nNumero

**exceptionTest** (no se usa)

**refrescar\_asientos** (no se usa): sirve para extender el tiempo de reserva de x asiento(s)

## Procedimientos

**view\_iti\_tra\_update**

* Parámetros (ningún parámetro)
* Funcionamiento: se manda a llamar desde el código después de insertar, editar o eliminar un tramo. Sirve como ayuda para calcular los tiempos de llegada.
* Retorno (ningún retorno)

## Triggers

No hay .\_.

***Diccionario de datos***

Ver diccionario de datos adjunto como hoja de cálculo.

# Cliente de Windows

El cliente de windows consiste en un navegador web basado en WebView2. De esta manera logramos algunas funciones que los navegadores web no nos ofrecen.

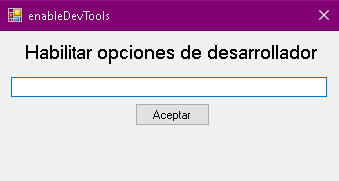
**Seguridad**

Exceptuando a administradores, el acceso al sistema se deberá realizar a travéz de su propio programa cliente, hecho en C#: Utilizando la librería *libc.hwid* se crea un Identificador del equipo (HarwareID / HwID) que se guarda en la base de datos [*BD.terminales.hwid]*

Añadimos más seguridad mediante la inserción de campos ocultos en formularios, con la función *ExecuteScriptAsync()****,*** antes de ser enviados que se validarán en el servidor web.

Se han bloqueador las herramientas de desarrollador y para iniciarlas hace falta ingresar la contraseña de administrador, con esto evitamos que se vea el código y se pueda intervenir. Para hacer uso de estas herramientas navegamos al menú Herramientas > Dev Tools e ingresamos la contraseña



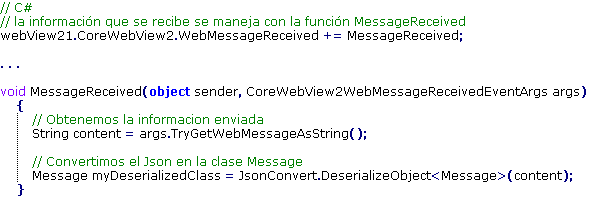


Se deshabilitó el menú contextual y el poder abrir otras ventanas para evitar que desde ahí se habran las opciones de desarrollador, también el auto completado y almacenamiento de contraseñas, para evitar el mal uso de estos datos

**Comunicación**

También contamos con las siguientes don funciones que nos permiten comunicar nuestro código Javascript con el navegador, con esto logramos otras funcionalidades como leer archivos (para procesar pagos) o mandar archivos directamente a la impresora.

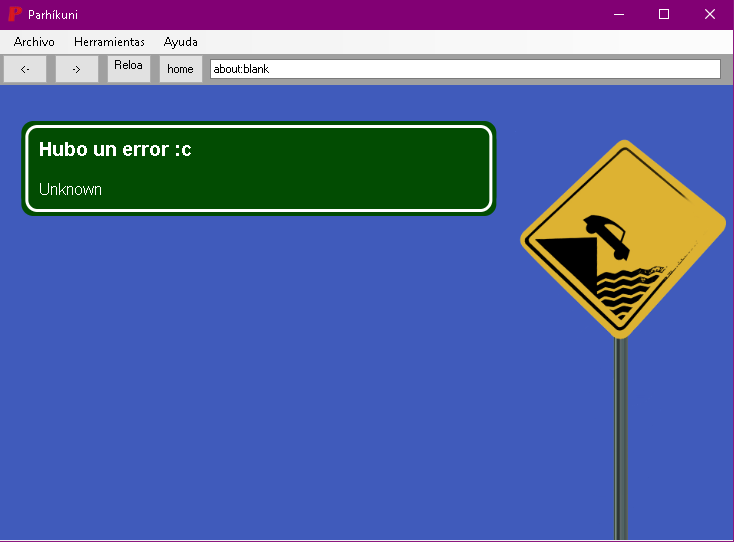




**Actualización automática (pendiente)**

Funcionalidad pendiente.

**Pantalla de error personalizada**



# Laravel / PHP

Se optó por desarrollar el sistema utilizando el framework Laravel por ser PHP un lenguaje bastante extendido, con una amplia documentación, por ser aceptado en todos los servidores web sin ningún costo adicional, por ser fácil realizar cambios y pruebas en partes específicas de los procesos.

Para instalar este software hay que clonar el repositorio en <https://github.com/anon34mx/venta2022>