

UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA

SEDE VIÑA DEL MAR – JOSÉ MIGUEL CARRERA

SISTEMA WEB DE RESERVA DEL HOSTAL VIÑA BELLA.

Trabajo de Titulación para optar al Título
de Técnico Universitario en
INFORMÁTICA.

Alumnos:

Mario Hoare Silva

Alexsander Zúñiga Cordero

Profesor Guía:

Carlos Alten López

RESUMEN

KEYWORDS: Sistema - PHP - Laravel - Hostal - JQuery

El objetivo de este sistema es facilitar la gestión de las reservas, al igual que mejorar el manejo de los anuncios de éstas y poder extraer información y tenerla disponible oportunamente para mejorar la toma de decisiones dentro del Hostal Viña Bella tanto para el administrador de esta empresa como para los empleados.

Para poder crear una solución que satisficiera estos requerimientos tuvimos que escoger una alternativa web, en la cual utilizamos PHP y el *framework* Laravel por el lado del servidor, mientras que por el lado del cliente utilizamos JavaScript y *jquery*. Para la base de datos se optó por MySQL para poder crear este motor de reservas web.

Este texto se organiza en capítulos cuyos contenidos comprenden los siguientes temas:

Capítulo 1. Descripción de la situación actual y breve historia del Hostal Viña Bella. Se detallan los problemas detectados y se describen en términos generales las funciones que se planean construir para solucionarlos.

La situación actual es que ningún proceso está automatizado y tiene un apoyo débil de herramientas digitales para solucionar problemas de administración, que en este caso sería el uso de planillas excel, lo que puede conducir a inconsistencias de información y se puede afirmar que actualmente no satisface las necesidades de información de la empresa.

La solución a los problemas antes mencionados es construir un software que apoye la gestión administrativa de las habitaciones del hostel para evitar inconsistencias, y gracias a este se podrá reunir datos y generar informes que satisfagan las necesidades de información de la empresa.

Capítulo 2. Descripción del entorno de implementación. Se describen el *software* y *hardware* disponible en el hostel y el que tendrá que ser adquirido, además del que se utilizará para el desarrollo del sistema y la descripción de las tablas de la base de datos de este.

El software necesario para poder sostener los requerimientos tomados, en general, apunta a utilizar un servidor en donde se desplegará la aplicación web, este servidor estará alojado por un proveedor de VPS. Para poder utilizar esta aplicación bastará con que los usuarios tengan instalado un navegador.

El hardware necesario por la aplicación está administrado por el proveedor de VPS, el de los usuarios pueden ser cualquier computadora que pueda mantener funcionando de forma óptima un navegador web. De todas maneras, se describe con mayor detalle el que actualmente está disponible en el hostel.

El modelo de datos es descrito con mayor detalle en este capítulo, en donde se describe cada tabla con sus campos y para qué serán utilizados, además de sus restricciones, largos y tipos.

Capítulo 3. Descripción de la arquitectura del sistema. Se presentan diagramas y se describen los programas del sistema, para dar una mayor claridad al funcionamiento del sistema y la interacción que tiene que tener el usuario con él.

Se utilizan distintos tipos de diagramas para poder mostrar cómo está formado este sistema. Entre ellos está el diagrama de menús, que muestra cómo está organizado por categoría de operaciones. Otro sería el modular, que muestra cuáles son los distintos módulos del sistema y como están conformados, siendo los más importantes “clientes”, “habitaciones”, “reservas” y “pago”.

Los diagramas de bloques junto a las vistas describen los requerimientos para que puedan funcionar los distintos programas del sistema, y cuál es su lógica de ejecución y qué distintos resultados pueden arrojar.

ÍNDICE

INTRODUCCIÓN.....	1
CAPÍTULO 1: ASPECTOS RELEVANTES DEL DISEÑO LÓGICO.....	3
1.1. DESCRIPCIÓN DE LA ORGANIZACIÓN	5
1.2. DESCRIPCIÓN DE LA SITUACIÓN ACTUAL.....	5
1.3. PROBLEMAS DETECTADOS.....	7
1.4. DESCRIPCIÓN DEL SISTEMA PORPUESTO.....	8
1.4.1. Objetivos Generales del Sistema Propuesto.....	8
1.4.2. Objetivos Específicos del Sistema Propuesto.....	8
1.4.3. Funcionalidades del Sistema.....	8
1.4.4. Entradas y salidas.....	9
1.4.5. Condicionantes de Diseño.....	16
1.4.6. Modelo de Datos.....	18
CAPÍTULO 2: MEDIO AMBIENTE COMPUTACIONAL Y DESCRIPCIÓN DE ARCHIVOS.....	19
2.1. RECURSOS COMPUTACIONALES.....	21
2.1.1. Configuración del hardware.....	21
2.1.2. Software Utilizado.....	23
2.2. DESCRIPCIÓN DE ARCHIVOS.....	25
2.2.1. Tabla “clientes”	25
2.2.2. Tabla “usuario”	26
2.2.3. Tabla “ciudad”	27
2.2.4. Tabla “pais”	28
2.2.5. Tabla “cancelacion”	28
2.2.6. Tabla “devolucion”	29
2.2.7. Tabla “reservacion”	30
2.2.8. Tabla “pago”	31

2.2.9. Tabla “moneda”	32
2.2.10. Tabla “valor_moneda”	33
2.2.11. Tabla “días_reservada”	34
2.2.12. Tabla “habitacion”	34
2.2.13. Tabla “tipo_habitacion”	35
2.2.14. Tabla “bloqueo”	36
2.2.15. Tabla “anuncio”	37
2.2.16. Tabla “usuario_ota”	38
2.2.17. Tabla “ota”	39
CAPÍTULO 3: DESCRIPCIÓN DE PROGRAMAS.....	41
3.1. DIAGRAMAS DE MENÚS.....	43
3.2. DIAGRAMA MODULAR.....	47
3.3. PROGRAMAS DEL SISTEMA.....	49
3.4. DESCRIPCIÓN DE PROGRAMAS.....	53
3.4.1. Inicio de Sesión.....	53
3.4.2. Modificar Habitación.....	54
3.4.3. Registrar Reservación.....	55
3.4.4. Asignar Habitación.....	60
3.4.5. Lista de Clientes Hospedados.....	62
3.4.6. Eliminar Pago.....	63
3.4.7. Consultar Clientes.....	65
3.4.8. Inicio.....	67
CONCLUSIÓN.....	69
BIBLIOGRAFÍA.....	71
ANEXO A: CODIGOS FUENTES.....	73

ÍNDICE DE TABLAS

Tabla 2.1 Especificaciones técnicas hardware servidor.....	21
Tabla- 2.2 Especificaciones técnicas hardware cliente Notebook Asus.....	22
Tabla- 2.3 Especificaciones técnicas hardware cliente Notebook Sony Vaio	23
Tabla- 2.4 Descripción de campos Tabla “cliente”	25
Tabla- 2.5 Descripción de campos Tabla “usuario”	26
Tabla- 2.6 Descripción de campos Tabla “ciudad”	27
Tabla- 2.7 Descripción de campos Tabla “pais”	28
Tabla- 2.8 Descripción de campos Tabla “cancelacion”	29
Tabla- 2.9 Descripción de campos Tabla “devolucion”	29
Tabla- 2.10 Descripción de campos Tabla “reservacion”	30
Tabla- 2.11 Descripción de campos Tabla “pago”	32
Tabla- 2.12 Descripción de campos Tabla “moneda”	33
Tabla- 2.13 Descripción de campos Tabla “valor_moneda”	33
Tabla- 2.14 Descripción de campos Tabla “Dias_reservada”	34
Tabla- 2.15 Descripción de campos Tabla “habitacion”	35
Tabla- 2.16 Descripción de campos Tabla “tipo_habitacion”	36
Tabla- 2.17 Descripción de campos Tabla “bloqueo”	37
Tabla- 2.18 Descripción de campos Tabla “anuncio”	38
Tabla- 2.19 Descripción de campos Tabla “usuario_ota”	39
Tabla- 2.20 Descripción de campos Tabla “OTA”	40
Tabla- 3.1 Programas del sistema.....	49

ÍNDICE DE FIGURAS

Figura 1.1 Agenda de reservaciones	6
Figura 1.2 Planilla Excel	6
Figura 1.3 Modelo de datos.....	18
Figura 3.1 Pantalla de inicio de sesión.....	53
Figura 3.2 Pantalla de error en inicio de sesión.....	54
Figura 3.3 Pantalla de formulario modificar habitación.....	55
Figura 3.4 Pantalla de mensaje satisfactorio de habitación modificada.....	55
Figura 3.5 Pantalla de mensaje error de fecha vacía al registrar reservación.....	57
Figura 3.6 Pantalla de registrar reservación.....	58
Figura 3.7 Pantalla de confirmar reservación	59
Figura 3.8 Pantalla de mensaje satisfactorio al registrar reservación.....	59
Figura 3.9 Pantalla de mensaje error al seleccionar fechas de reservación.....	60
Figura 3.10 Pantalla de mensaje error al asignar habitación.....	60
Figura 3.11 Pantalla de asignar habitación.....	61
Figura 3.12 Pantalla de gestor lista de clientes hospedados.....	63
Figura 3.13 Pantalla de tabla de lista de clientes hospedados.....	63
Figura 3.14 Pantalla confirmación eliminar pago.....	64
Figura 3.15 Pantalla eliminar pago.....	65
Figura 3.16 Pantalla consultar cliente tabla con datos.....	66
Figura 3.17 Pantalla consultar clientes tabla vacía.....	66
Figura 3.18 Pantalla de inicio.....	67

ÍNDICE DE DIAGRAMAS

Diagrama 3.1 Diagrama de menú administrador.....	43
Diagrama 3.2 Diagrama de menú recepcionista.....	45
Diagrama 3.3 Diagrama modular.....	47
Diagrama 3.4 Diagrama de bloques Inicio de sesión.....	53
Diagrama 3.5 Diagrama de bloques modificar habitación.....	54
Diagrama 3.6 Diagrama de bloques registrar reservación.....	56
Diagrama 3.7: Diagrama de bloques asignar habitación.....	61
Diagrama 3.8: Diagrama de bloques lista de clientes hospedados.....	62
Diagrama 3.9 Diagrama de bloques Eliminar pago.....	64
Diagrama 3.10 Diagrama de bloques Consultar clientes.....	65
Diagrama 3.11 Diagrama de bloques pantalla de inicio.....	67

INTRODUCCIÓN

Hostal Viña Bella es un hostel de ambiente familiar con giro en el hospedaje temporal a través del arriendo de diferentes tipos de habitaciones, está dirigido a captar principalmente viajeros que vienen a vacacionar a Viña del Mar.

Actualmente no cuenta con un medio que ayude a satisfacer completamente sus necesidades operacionales y de información, ya que tienen que recurrir al uso de planillas en Excel para poder administrar las reservas, tanto de su registro como de su asignación correspondiente a una habitación. Además, el administrar sus anuncios en distintas páginas web se vuelve una tarea tediosa.

Para resolver los problemas antes mencionados, se construirá el sistema web de reservas, el cual contará con todos los módulos necesarios para poder manipular las reservas y administrar los anuncios en las demás páginas web.

Este sistema se construirá utilizando el *framework* Laravel y JavaScript, con el fin de poder hacer funcional este sistema de forma web y modularizar lo más posible el sistema. Al separarlo en distintas capas y módulos con responsabilidades específicas su construcción y mantención será más fácil, además se utilizará el motor de base de datos MySQL para mantener la integridad de los datos.

CAPÍTULO 1: ASPECTOS RELEVANTES DEL DISEÑO LÓGICO

1. ASPECTOS RELEVANTES DEL DISEÑO LÓGICO

1.1. DESCRIPCIÓN DE LA ORGANIZACIÓN

Hostal Viña Bella es un hostel de ambiente familiar con giro en el hospedaje temporal a través del arriendo de diferentes tipos de habitaciones, las cuales se acomodan a las necesidades de los viajeros.

Su actual dueña y administradora es doña Karen Beckdorf, la cual trabaja junto a dos empleadas, las que ayudan en el cuidado y limpieza de las habitaciones, así como también en la gestión de las reservas.

El Hostal lleva funcionando cerca de 4 años bajo el mismo nombre, éste se encuentra ubicado en la calle 5 ½ Poniente n°55 en la ciudad de Viña del Mar, región de Valparaíso.

Durante su trayectoria el hostel ha tenido otros tres dueños. La actual dueña está al mando desde octubre de 2017. En estos momentos lo está sometiendo a un proceso de actualización para mantener el ambiente del hostel lo más cómodo posible para los huéspedes.

Actualmente, el Hostal cuenta con un total de 12 habitaciones, las cuales son habitaciones dobles con baño privado, habitaciones triples con baño privado, habitaciones dobles con baño privado externo, habitaciones cuádruples con baño privado y habitaciones dobles estándar. Estas habitaciones se encuentran divididas en dos secciones, denominadas la sección de habitaciones y la sección de departamento. Las habitaciones que se encuentran en la sección de departamento cuentan con refrigeradores para la comodidad del huésped.

1.2. DESCRIPCIÓN DE LA SITUACIÓN ACTUAL

El hostel maneja las reservas a través de una planilla en Excel donde se registran los datos de los clientes que realizan una reserva de habitación. Los datos son obtenidos directamente por la dueña del hostel via mail o red social, logrando que la reservación se ingrese directamente en la planilla, sin embargo, cuando la reservación es recibida por las recepcionistas, estas anotan en una agenda la petición para luego ser traspasada a la planilla por la dueña.



Figura 1.1 Agenda de reservaciones.

Fuente: Hostal Viña Bella.

Día	Noche del	Departamentos			
		1	2	3	4
Domingo	4/1/2018	Roberto Pacheco	Beatriz Gomez	Alfonso Bravo	Daiana Vergara
Lunes	4/2/2018	Roberto Pacheco	Beatriz Gomez		Alexis Davila
Martes	4/3/2018	Pablo Abarca Araya	Beatriz Gomez	Ivan Castillo	Alexis Davila
Miércoles	4/4/2018	Pablo Abarca Araya	Fernanda Bravo	Ivan Castillo	Alexis Davila
Jueves	4/5/2018	Pablo Abarca Araya	Fernanda Bravo	Ivan Castillo	Alexis Davila
Sábado	4/6/2018	David Monardez	Jose Ortega	Sara Yadegari	German
	4/7/2018	Oscar Gaete	Alejandro Palma	Constanza Salamanca	Sara Yadegari

Figura 1.2 Planilla Excel.

Fuente: Hostal Viña Bella.

Los datos que se piden son el nombre, fecha de reservación, fecha y hora de llegada (*check-in*), fecha y hora de salida (*check-out*), cantidad de personas, tipo habitación, número de noches a hospedarse y método de pago. Las reservaciones se realizan a través de contacto telefónico con el hostel, a través de Facebook o *e-mail*. El cliente debe pagar con anticipación el 50% del total del valor para confirmar la reserva, el cual se puede depositar en una cuenta bancaria. El hostel también maneja las reservas a través de Booking.com, una página web de viajes y turismo. En dicha plataforma los clientes reservan las habitaciones y pagan vía tarjeta de crédito. Además de eso, también se puede alquilar habitaciones sin previa reservación dependiendo de la disponibilidad del hostel. En estos momentos el hostel solo recibe pagos en dólares y pesos chilenos.

Actualmente, el hostel no posee ningún sistema informático, por lo cual no cuentan con un medio seguro y eficaz para poder documentar las reservaciones que se realizan. Lo anterior conlleva a un gran gasto de recursos y tiempo por parte de la dueña, que pasa hasta un máximo de 5 horas diarias manejando las reservaciones que se realizan. Además, la dueña asigna las habitaciones dependiendo de los requerimientos de las reservas entrantes, por lo que puede cambiar la ocupación de una habitación reservada a otro cliente dependiendo de la necesidad. Todo este trabajo se vuelve tedioso y difícil de llevar a cabo a través de una planilla.

Tampoco cuenta con la posibilidad de poder generar informes a través de los archivos de las reservas, ya que no cuenta con un medio para poder procesar la gran cantidad de información con la que cuenta.

1.3. PROBLEMAS DETECTADOS

- Uno de los problemas es la falta de control en las reservaciones de las habitaciones, recibidas por distintos medios en el hostel.
- Gran gasto de recursos y tiempo en el manejo de las reservas, ya que no están sincronizados los métodos al momento de generar una reserva o cancelarla, produciendo inconsistencia antes de llegar a la planilla Excel.
- Tiempos de consultas o de búsqueda de datos extensas debido a que se cuenta con una gran cantidad de información guardada.
- Todas las reservas quedan registradas en una planilla Excel, lo cual genera que éstas sean extensas, difíciles de manejar y con alta redundancia.
- Incapacidad de poder generar informes o estadísticas de los datos de las reservas para ayudar en la toma de decisiones dentro del hostel.
- Problemas al publicitarse a través de una gran cantidad de *Online Travel Agencies* (en adelante OTAs).
- No se tiene acceso oportuno a la información de las reservas en cualquier momento y lugar como requiere el administrador.

1.4. DESCRIPCIÓN DEL SISTEMA PROPUESTO

1.4.1. Objetivo Generales del Sistema Propuesto

Brindar al hostel Viña Bella una aplicación web práctica para poder gestionar los datos de las reservas del hostel de manera efectiva, publicitarse a través de distintas OTAs y aprovechar los datos registrados en el sistema de reservas para generar informes que puedan ayudar a la toma de decisiones.

1.4.2. Objetivos Específicos del Sistema Propuesto

- a. Construir e implementar una aplicación web que administre las reservas, ésta registrará las operaciones que solicitan los clientes a los operadores del sistema, como reservar una habitación, cancelar una reservación, extender o disminuir el tiempo de hospedaje de un cliente, entre otras operaciones.
- b. Generar diferentes tipos de informes en base a los datos de las reservas y estancia de los clientes para la toma de decisiones.
- c. Disminuir la cantidad de horas invertidas para manejar la información de las reservas entrantes y facilitar el manejo de las reservas a las recepcionistas.

1.4.3. Funcionalidades del Sistema:

Gestor de reservas: Permitirá ingresar, consultar, cancelar o modificar la reserva realizada por un cliente. También contempla los posibles cambios de las reservas, como el periodo de estancia del huésped o las habitaciones en específico a utilizar.

Gestor de cliente: Permitirá ingresar, consultar, modificar y eliminar un cliente al momento de realizar la reserva.

Gestor de recepción: Permitirá registrar, consultar, y eliminar el *check-in* y el *check-out* de un cliente.

Gestor de informes: Permitirá generar distintos tipos de informes a través de los datos recolectados mediante el sistema, como los datos de los clientes (RUT, nombre, dirección, número de veces en las que se ha hospedado), de las reservas (fecha de reservación, valor

por noche, total a pagar) y de las habitaciones (habitaciones disponibles, habitaciones ocupadas), entre otro tipo de datos.

Gestor de pago: Permitirá registrar, consultar, eliminar o modificar un pago realizado por un cliente el cual está ligado a una reserva, así como también registrar las devoluciones de dinero causadas por cancelar o por acortar el tiempo de estancia acordado en la reserva, o al contrario, registrar los cargos extras por extender su estadía.

Gestor de divisa: Registrará, modificará, eliminará y permitirá consultar las divisas que manejará el sistema. También estará encargado de registrar el valor en pesos chilenos al momento del pago, además de mantener actualizado el valor de las divisas dentro del sistema.

Gestor de habitaciones: Permitirá registrar los datos de las habitaciones (su capacidad, costo, código de la habitación, entre otros), modificar, consultar, eliminar o cambiar el estado de disponibilidad de las habitaciones.

Gestor de usuarios: Permitirá agregar, consultar, modificar o eliminar lógicamente a usuarios del sistema, los cuales están encargados de la administración de las reservas.

Diagrama interactivo de asignación de habitaciones: Representación gráfica que mostrará dentro de un rango de fechas la disponibilidad de las habitaciones y permitirá a los usuarios asignar las reservaciones a una habitación.

Channel manager: Módulo encargado de administrar, generar, modificar y eliminar los múltiples anuncios publicados en las distintas OTAs, su tarea será replicar el mismo anuncio en diferentes OTAs.

1.4.4-Entradas y Salidas

1.4.4.1-Entradas:

-Datos del cliente (Núm. documento cliente, tipo documento, nombre, teléfono, correo, dirección).

-Datos del pago (Código de pago, tipo de pago, monto total a pagar, estado de pago, monto de abono, tipo de moneda, fecha pago).

- Datos de la habitación (Código habitación, capacidad de la habitación, tipo de baño, cantidad de baños, precio habitación por noche, estado de la habitación).
- Datos del usuario (código de usuario, nombre, apellido, nombre de usuario, contraseña, nivel de acceso, RUT, teléfono, correo, dirección).
- Datos de reserva (Código de reserva, núm. documento cliente, código de pago, código de habitaciones, fecha de entrada, fecha de salida, fecha de reservación).
- Datos de moneda (Valor de compra [en \$CLP), valor de venta, fecha actual de la moneda).
- Datos de las OTAs (código de la página, URL, nombre de las páginas, versión API).
- Importar las reservas realizadas antes del sistema desde un archivo Excel hacia la base de datos.

1.4.4.2-Salidas:

- Lista de clientes:

Información: {Código, Nombre, Apellido materno, Apellido Paterno, Número de teléfono, País, Ciudad, Dirección, Correo electrónico, Fecha de última Estancia, Última habitación reservada, Código última reservación, Número de veces hospedado, Estancia promedio, Tarifa promedio cancelada, Número de reservas canceladas, Número de extensiones, Número de disminuciones}

Descripción: Se mostrará en pantalla una lista detallada de todos los clientes. Es posible que se pueda especificar un rango de fechas, si se hace, se limitará a mostrar solo los clientes que están hospedados dentro de ese rango, si no se especifica será global. También se dará la posibilidad de elegir los campos a desplegar, en caso contrario se mostrará una lista estándar, que se podrá ordenar según el criterio correspondiente al campo elegido, (si el campo es numérico: de mayor a menor o al revés, si no de la A a la Z o viceversa, o si es fecha según antigüedad) además de poder aplicar distintos filtros para buscar dentro de la lista.

-Lista de clientes hospedados:

Información: {Código, Nombre, Apellido materno, Apellido Paterno, Número de teléfono, País, Ciudad, Dirección, Correo electrónico, Check in, Fecha de fin de reserva, Código de habitación, Tipo de habitación}

Descripción: Se mostrará en pantalla una lista detallada de todos los clientes hospedados en un rango de fechas, si no se usa el rango, muestra los de la fecha actual. Se podrá ordenar según el criterio correspondiente al campo, además de poder aplicar distintos filtros para buscar dentro de la lista.

-Informe de clientes:

Información: {Fecha inicio, Fecha fin, Número de clientes, Clientes extranjeros, Clientes nacionales, Estancia promedio, Tarifa promedio, Estancia promedio nacional, Tarifa promedio nacional, Estancia promedio internacional, Tarifa promedio internacional, Número de *no show* (*El cliente no se presenta*), Numero de *no show* nacional, Número de *no show* internacional}

Descripción: Se mostrará en pantalla una lista que detalla los datos de los clientes correspondientes a un periodo de tiempo que puede ser semanal, mensual o anual, dentro de un rango de fechas. Se podrá ordenar según el criterio correspondiente al campo, además de poder aplicar distintos filtros para buscar dentro de la lista.

- Lista de habitaciones:

Información: {Código, Tipo de habitación, Capacidad, Número de baños, Tipo de baño, Precio por noche, Código ultimo cliente, Nombre último cliente, Última fecha de uso, Estancia promedio, Días ocupada, Días bloqueada, Días disponible, Número de Reservas}

Descripción: Se mostrará en pantalla una lista detallada de todas las habitaciones. Es posible que se pueda especificar un rango de fechas, si no se especifica será global, también se dará la posibilidad de elegir los campos a desplegar, si no, se mostrará una lista estándar. Se

podrá ordenar según el criterio correspondiente al campo elegido, además de poder aplicar distintos filtros para buscar dentro de la lista.

- Lista de disponibilidad de habitaciones:

Información: {Código, Tipo de habitación, Capacidad, Número de baños, Tipo de baño, Precio por noche, Fecha inicio bloqueo, Fecha fin bloqueo, Estado habitación}

Descripción: Se mostrará en pantalla una lista detallada de todas las habitaciones disponibles o no disponibles según elija el usuario, dentro de un rango de fechas específico. Se podrá ordenar según el criterio correspondiente al campo elegido, además de poder aplicar distintos filtros para buscar dentro de la lista.

-Lista de tipo de habitaciones:

Información: {Código, Nombre de tipo de habitación, Capacidad, Número de baños, Tipo de baño, Precio por noche, Cantidad de habitaciones, Estancia promedio}

Descripción: Se mostrará en pantalla una lista detallada de todos los tipos de habitaciones. Se podrá ordenar según el criterio correspondiente al campo elegido, además de poder aplicar distintos filtros para buscar dentro de la lista.

-Informe de habitaciones:

Información: {Fecha inicio, Fecha fin, Tipo de habitación, Estancia promedio, Tarifa promedio, Dinero recaudado, Número de reservaciones, Días ocupadas promedio, Días bloqueadas promedio, Días disponibles promedio}

Descripción: Se mostrará en pantalla una lista que detalla los datos de las habitaciones agrupadas por su tipo correspondiente a un periodo de tiempo que puede ser semanal, mensual o anual, dentro de un rango de fechas. Se podrá ordenar según el criterio correspondiente al campo, además de poder aplicar distintos filtros para buscar dentro de la lista.

-Lista de reservaciones:

Información: {Código, Fecha reservación, Fecha inicio, Fecha fin, Días de estancia, {Código habitación, Tipo de habitación, Precio por noche}, Total a pagar, Estado de reserva, Código cliente, Nombre Cliente}

Descripción: Se mostrará en pantalla una lista detallada de todas las reservas. Es posible que se pueda especificar un rango de fechas, si no se especifica será global, también será posible que se filtre por estado de la reserva. Además, se dará la posibilidad de elegir los campos a desplegar, si no, mostrará una lista estándar. Se podrá ordenar según el criterio correspondiente al campo elegido, además de poder aplicar distintos filtros para buscar dentro de la lista.

-Informe de reservas:

Información: {Fecha inicio, Fecha fin, Número de reservas, Número de reservas canceladas, Número de reservas extendidas, Número de reservas acortadas, Número de no show}

Descripción: Se mostrará en pantalla una lista que detalla los datos de las reservaciones correspondientes a un periodo de tiempo que puede ser semanal, mensual o anual, dentro de un rango de fechas. Se podrá ordenar según el criterio correspondiente al campo, además de poder aplicar distintos filtros para buscar dentro de la lista.

-Lista de pagos:

Información: {Código, Código reservación, Tipo de habitación, Precio por noche, Medio de pago, Fecha de pago, Divisa, Monto, Código cliente, Nombre cliente}

Descripción: Se mostrará en pantalla una lista detallada de todos los pagos dentro un rango de fecha específico se podrá ordenar según el criterio correspondiente al campo elegido, además de poder aplicar distintos filtros para buscar dentro de la lista.

-Lista de pagos pendientes:

Información: {Código reservación, Fecha inicio estadía, Fecha fin estadía, Divisa, Monto, Código cliente, Nombre cliente}

Descripción: Se mostrará en pantalla una lista detallada de todos los pagos pendientes. Se podrá ordenar según el criterio correspondiente al campo elegido, además de poder aplicar distintos filtros para buscar dentro de la lista.

-Lista de Devoluciones:

Información: {Código, Código reservación, Medio de pago, Fecha de devolución, Divisa, Monto, Código cliente, Nombre cliente}

Descripción: Se mostrará en pantalla una lista detallada de todas las devoluciones dentro un rango de fechas específico. Se podrá ordenar según el criterio correspondiente al campo elegido, además de poder aplicar distintos filtros para buscar dentro de la lista.

-Informe de pagos:

Información: {Fecha inicio, Fecha fin, Recaudación total hospedaje, Recaudación multas cancelación, Dinero devuelto, Recaudación por multas no show (el cliente no se presenta), Pérdidas estimadas por cancelación, Pérdidas estimadas por no show}

Descripción: Se mostrará en pantalla una lista que mostrará los datos de los movimientos de dinero relacionados con las reservas, correspondientes a un periodo de tiempo que puede ser semanal, mensual o anual, dentro de un rango de fecha. Se podrá ordenar según el criterio correspondiente al campo, además de poder aplicar distintos filtros para buscar dentro de la lista.

-Lista de cancelaciones:

Información: {Código, Código reservación, Fecha cancelación, Código devolución Medio de pago, Fecha de devolución, Divisa, Monto, Código cliente, Nombre cliente}

Descripción: Se mostrará en pantalla una lista detallada de todas las cancelaciones dentro un rango de fecha específico se podrá ordenar según el criterio correspondiente al campo elegido, además de poder aplicar distintos filtros para buscar dentro de la lista.

-Lista de divisas:

Información: {Código, Nombre divisa}

Descripción: Se mostrará en pantalla una lista detallada de todas las divisas. Se podrá ordenar según el criterio correspondiente al campo elegido, además de poder aplicar distintos filtros para buscar dentro de la lista.

-Lista de valor de divisa:

Información: {Fecha, Valor}

Descripción: Se mostrará en pantalla una lista detallada de todos los valores de la divisa dentro un rango de fecha específico. Se podrá ordenar según el criterio correspondiente al campo elegido, además de poder aplicar distintos filtros para buscar dentro de la lista.

-Lista de usuarios:

Información: {Código, Nombre, Apellido materno, Apellido Paterno, Número de teléfono, Correo electrónico, Ciudad, Dirección, Identificación, Contraseña, Nivel de acceso}

Descripción: Se mostrará en pantalla una lista detallada de todos los usuarios. Se podrá ordenar según el criterio correspondiente al campo elegido, además de poder aplicar distintos filtros para buscar dentro de la lista.

-Lista de actividad de usuarios:

Información: {Fecha, Código usuario, Nombre, acceso, código operación, Nombre operación}

Descripción: Se mostrará en pantalla una lista detallada de toda la actividad de los usuarios (registrar, cancelar, modificar las reservas, las devoluciones de dinero y los pagos recibidos) dentro un rango de fechas específico. Se podrá ordenar según el criterio correspondiente al campo elegido, además de poder aplicar distintos filtros para buscar dentro de la lista.

-Lista de *Check-in* del día:

Información: {Código cliente, Nombre, Apellido paterno, Apellido materno, Número de teléfono, Código de reservación, Monto por pagar, {Código de habitación, tipo de habitación}}

Descripción: Se mostrará en pantalla una lista detallada de los clientes que tienen que llegar dentro del día. Se podrá ordenar según el criterio correspondiente al campo elegido, además de poder aplicar distintos filtros para buscar dentro de la lista.

-Lista de Check out del día:

Información: {Código cliente, Nombre, Apellido paterno, Apellido materno, Número de teléfono, Código de reservación, Monto por pagar, {Código de habitación, tipo de habitación}}

Descripción: Se mostrará en pantalla una lista detallada de los clientes que tienen que irse dentro del día. Se podrá ordenar según el criterio correspondiente al campo elegido, además de poder aplicar distintos filtros para buscar dentro de la lista.

-Lista de anuncios:

Información: {Código OTA, url anuncio, {Ruta imagen habitación}, Precio por noche, Tipo de habitación, Dirección Hostal, Nombre hostal, Teléfono hostal, Correo hostal}

Descripción: Se mostrará en pantalla una lista detallada de los anuncios publicados en las distintas OTAs. Se podrá ordenar según el criterio correspondiente al campo elegido, además de poder aplicar distintos filtros para buscar dentro de la lista.

-Calendario interactivo: mostrará la disponibilidad de una habitación específica dentro de un mes.

-Notificador: Permite que los usuarios reciban notificaciones dentro de la aplicación sobre las reservas realizadas dentro de la misma aplicación, a través de diferentes OTAs y en el caso de las que han sido ingresadas manualmente, se les notificará a los usuarios que se les debe asignar una habitación a la reserva.

-Exportar el listado generado por el sistema a sus archivos Excel respectivos.

1.4.5 Condicionantes de Diseño

El sistema para satisfacer el requerimiento de acceso a la información relativa a las reservas en cualquier momento y lugar requerirá de un servidor web. Éste será alojado en algún proveedor de hosting, debido a que es más acorde a las necesidades y presupuesto del hostel que un servidor físico.

Se decidió optar por una aplicación web y no de escritorio a causa de que es más simple el desarrollo de la interfaz de usuario para dispositivos móviles.

En esta fase de desarrollo del sistema es apresurado escoger las herramientas para su desarrollo, pero se está evaluando usar PHP con su *framework* Laravel para el *back-end*, y para el *front-end* usaremos javascript, más HTML y CSS con su *framework* Bootstrap, La base de datos será MySQL y un servidor apache, todo esto dentro de una arquitectura de un servidor LAMP.

1.4.6 Modelo de Datos

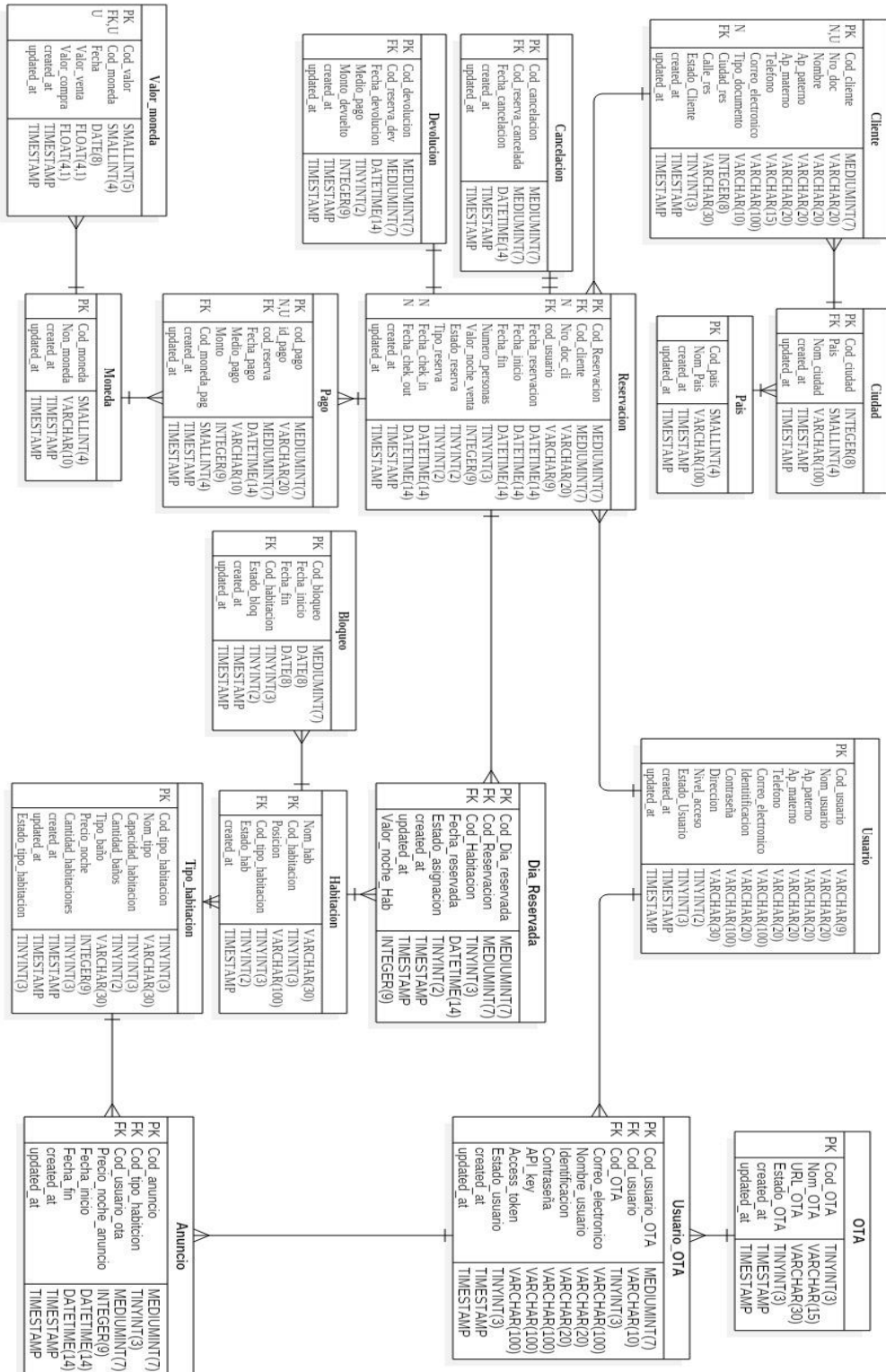


Figura 1.3 Modelo de datos.

Fuente: Elaboración propia.

CAPÍTULO 2: MEDIO AMBIENTE COMPUTACIONAL Y DESCRIPCIÓN DE ARCHIVOS

2. MEDIO AMBIENTE COMPUTACIONAL Y DESCRIPCIÓN DE ARCHIVOS

2.1 RECURSOS COMPUTACIONALES

El recurso computacional empleado en el sistema será bastante simple, ya que solo necesitará un *web browser*, considerando que este es un sistema basado en el manejo de reservas de un hostel vía sitio web, sin embargo, el servidor principal será el que se llevará todo el peso al guardar este sitio y la información que ahí se maneje.

2.1.1 Configuración del *Hardware*

2.1.1.1 Servidor

El *hardware* empleado en el sistema se basó en los requerimientos mínimos que se necesitan para que el servidor funcione correctamente, incluso bajo las condiciones de elevada concurrencia con los otros sistemas como el *channel manager* y los usuarios.

Este servidor será un recurso virtual el cual se contratará a través del servicio de *hosting* de DigitalOcean, el cual ofrece distintos tipos de servicios de máquinas virtuales dependiendo de la necesidad. El servicio que se eligió fue el siguiente:

Tabla- 2.1 Especificaciones técnicas *hardware* servidor.

Tipo procesador	Longitud de palabra: 64 bits.
	Tipo: Virtual CPU
	Cantidad de núcleos: 2.
Almacenamiento primario	Tipo: RAM.
	Capacidad: 4 GB.
Almacenamiento secundario	Tipo: SSD Disk
	Capacidad: 80 GB
Límite de Transferencia de datos mensual	4 TB

Fuente: Elaboración propia.

2.1.1.2. Cliente y Empresa

El *hardware* del cliente necesariamente tiene que ser capaz de poder ejecutar sin problemas el navegador Google Chrome en su última versión tanto en sus variantes de escritorio y móvil, además debe tener una conexión de internet estable para poder utilizar de forma óptima el sistema.

Las siguientes tablas describen el *hardware* que utilizan en el hostel y servirán como computador cliente del sistema.

Tabla- 2.2 Especificaciones técnicas *hardware* cliente Notebook Asus.

Tipo procesador	Marca: Intel Celeron N315
	Velocidad: 1.6 Ghz
Almacenamiento primario	Marca: Kingston
	Tipo: RAM
	Velocidad: 1866 MHz
	Capacidad: 4GB
Almacenamiento secundario	Marca: Western digital
	Tipo: HDD
	Velocidad: 5.550 rpm
	Capacidad: 500GB
Monitor:	Genérico 15,6 pulgadas
Mouse:	Genérico (Touchpad)
Teclado:	Genérico

Fuente: Elaboración propia.

Tabla- 2.3 Especificaciones técnicas *hardware* cliente Notebook Sony Vaio.

Tipo procesador	Marca: AMD E-350
	Velocidad: 1.6 Ghz
Almacenamiento primario	Marca: Kingston
	Tipo: RAM
	Velocidad:1.066 Mhz
	Capacidad: 2GB
Almacenamiento secundario	Marca: Toshiba
	Tipo: HDD
	Velocidad:5.400rpm
	Capacidad: 320 GBs
Monitor:	Genérico 11.6 Pulgadas
Mouse:	Genérico (Touchpad)
Teclado:	Genérico

Fuente: Elaboración propia.

2.1.2 Software Utilizado

2.1.2.1 Servidor

Sistema Operativo: GNU Linux, distribución Centos 7.x.

Esta distribución es apta para poder correr un servidor durante las veinticuatro horas del día, los siete días de la semana, es robusto y se tiene un mayor dominio de este sistema operativo, además de que cuenta con repositorios para facilitar la instalación y configuración del *software* necesario para el desarrollo del sistema y su despliegue. A continuación, se describen características técnicas del sistema operativo:

Tipo de Administración de Memoria: Segmentación.

- Concurrencia: Permite tiempo compartido y tareas *batch*. Además, es multiusuario.
- sistema de Archivos: XFS
- Utilitarios:
 - XAMPP:
 - Apache
 - MARIADB 5.5
 - PHP 7.x
 - Perl
 - Librerías propias del sistema

2.1.2.2 Administrador y Usuario

A. Sistema Operativo: Windows 10 Home Edition.

Sistema operativo fácil de usar, y más cercano y conocido por el usuario común que GNU Linux. Posee una robustez “relativamente aceptable” dentro de los sistemas Windows, además de ser seguro para el usuario. A continuación, se describen características técnicas del sistema operativo:

- Tipo de administración de memoria: Paginación
- Concurrencia: Permite tareas *batch* multitarea y no es multiusuario.
- Sistema de Archivos: NTFS.
- Utilitarios:
 - Google Chrome v67.0.33

2.1.2.3 Lenguajes de Programación

- PHP 7

Es un lenguaje de programación de propósito general que se utiliza por el lado del servidor, en un principio se utilizó para el desarrollo web de contenido dinámico, ya que se puede incorporar directamente en el documento HTML, que luego es interpretado por un módulo del servidor el cual genera el documento final.

-jQuery

jQuery es una biblioteca multiplataforma de JavaScript, creada inicialmente por John Resig, que permite simplificar la manera de interactuar con los documentos HTML, manipular el

árbol DOM, manejar eventos, desarrollar animaciones y agregar interacción con la técnica AJAX a páginas web.

2.2 DESCRIPCIÓN DE ARCHIVOS

2.2.1 Tabla “clientes”

Nombre Lógico: Cliente.

Nombre Físico: cliente.

Descripción: Tabla que contiene todos los clientes que se han hospedado en el hostel, con sus respectivos datos personales.

Clave(s) Primaria(s): cod_cliente.

Clave(s) Foránea(s): ciudad_res hace referencia a tabla ciudad y su campo cod_ciudad.

Longitud de Registro: 253 caracteres.

Campos:

Tabla- 2.4 Descripción de campos Tabla “cliente”.

Nombre	Tipo	Longitud	Descripción
cod_cliente	MEDIUMINT	7	Correlativo que se le asignará al cliente para identificarlo dentro del sistema
nro_doc	VARCHAR	20	RUT o número de pasaporte del cliente.
nombre	VARCHAR	20	Nombres del cliente.
ap_paterno	VARCHAR	20	Apellido Paterno del cliente.
ap_materno	VARCHAR	20	Apellido Materno del cliente.
telefono	VARCHAR	15	Teléfono de contacto del cliente.
correo_electronico	VARCHAR	100	Correo electrónico de contacto del cliente.
tipo_documento	VARCHAR	10	Tipo de documento (Cédula nacional o Pasaporte).

Nombre	Tipo	Longitud	Descripción
calle_res	VARCHAR	30	Calle de residencia del cliente.
ciudad_res	INTEGER	8	Código de la Ciudad de residencia del cliente.
estado_cliente	TINYINT	3	Valor que identificará si el cliente se encuentra eliminado de de manera lógica en el sistema (0:"vigente", 1:"eliminada") , y futuros estados que puedan ser añadidos.

Fuente: Elaboración propia.

2.2.2 Tabla "usuario"

Nombre Lógico: Usuario.

Nombre Físico: usuario.

Descripción: Tabla que contiene todos los usuarios que maneja el sistema, con sus respectivos datos personales.

Clave(s) Primaria(s): cod_usuario.

Longitud de Registro: 345 caracteres.

Campos:

Tabla- 2.5 Descripción de campos Tabla "usuario".

Nombre	Tipo	Longitud	Descripción
cod_usuario	VARCHAR	9	RUT del usuario que identificará al usuario en el sistema
nom_usuario	VARCHAR	20	Nombre del usuario
ap_paterno	VARCHAR	20	Apellido paterno del usuario
ap_materno	VARCHAR	20	Apellido materno del usuario
telefono	VARCHAR	20	Teléfono de contacto del usuario

Nombre	Tipo	Longitud	Descripción
correo_electronico	VARCHAR	100	Correo electrónico de contacto del usuario
identificacion	VARCHAR	20	Identificación del usuario que le permitirá ingresar al sistema.
contraseña	VARCHAR	100	Contraseña del usuario que validara su ingreso al sistema
direccion	VARCHAR	30	Dirección de residencia del usuario
nivel_acceso	TINYINT	3	Nivel de acceso que tendrá el usuario dentro del sistema (0: "Administrador", 1:"Recepcionista")
estado_usuario	TINYINT	3	Valor que permitirá identificar si el usuario se encuentra eliminado de manera lógica en el sistema(0:"vigente", 1:"eliminada") y futuros estados.

Fuente: Elaboración propia.

2.2.3 Tabla "ciudad"

Nombre Lógico: Ciudad.

Nombre Físico: ciudad.

Descripción: Tabla que contiene las ciudades dentro del sistema, con sus respectivos datos.

Clave(s) Primaria(s): cod_ciudad.

Clave(s) Foránea(s): pais hace referencia a la tabla pais y su campo cod_pais.

Longitud de Registro: 112 caracteres.

Campos:

Tabla- 2.6 Descripción de campos Tabla "ciudad".

Nombre	Tipo	Longitud	Descripción
cod_ciudad	INTEGER	8	Código de ciudad que la identificará dentro del sistema

Nombre	Tipo	Longitud	Descripción
pais	SMALLINT	4	Código del país que lo identificará dentro del sistema
nom_ciudad	VARCHAR	100	Nombre de la ciudad

Fuente: Elaboración propia.

2.2.4 Tabla “pais”

Nombre Lógico: Pais.

Nombre Físico: pais.

Descripción: Tabla que contiene los países dentro del sistema, con sus respectivos datos.

Clave(s) Primaria(s): cod_pais.

Longitud de Registro: 104 caracteres.

Campos:

Tabla- 2.7 Descripción de campos Tabla “pais”.

Nombre	Tipo	Longitud	Descripción
cod_pais	SMALLINT	4	Código del país que lo identificará dentro del sistema
nom_pais	VARCHAR	100	Nombre del país

Fuente: Elaboración propia.

2.2.5 Tabla “cancelacion”

Nombre Lógico: Cancelacion.

Nombre Físico: cancelacion.

Descripción: Tabla que registrará las cancelaciones ligadas a una reserva realizadas por el sistema

Clave(s) Primaria(s): cod_cancelacion.

Clave(s) Foránea(s): cod_reserva_cancelada hace referencia a tabla reserva y su campo cod_reserva.

Longitud de Registro: 28 caracteres.

Campos:

Tabla- 2.8 Descripción de campos Tabla “cancelacion”.

Nombre	Tipo	Longitud	Descripción
cod_cancelacion	MEDIUMINT	7	Código de cancelación que lo identificará dentro del sistema
cod_reseva_cancelada	MEDIUMINT	7	Código de la reserva cancelada
fecha_cancelacion	DATETIME	14	Fecha y hora en la cual se realizó la cancelación

Fuente: Elaboración propia.

2.2.6 Tabla “devolucion”

Nombre Lógico: Devolucion.

Nombre Físico: devolucion.

Descripción: Tabla que registrará las devoluciones de dinero ligadas a una reserva realizadas por el sistema

Clave(s) Primaria(s): cod_devolucion.

Clave(s) Foránea(s): cod_reserva_dev hace referencia a tabla reserva y su campo cod_reserva.

Longitud de Registro: 47 caracteres.

Campos:

Tabla- 2.9 Descripción de campos Tabla “devolucion”.

Nombre	Tipo	Longitud	Descripción
cod_devolucion	MEDIUMINT	7	Código de devolución con el cual se identificará en el sistema.
cod_reserva_dev	MEDIUMINT	7	Código de reserva sobre la cual se realiza la devolución de dinero.

Nombre	Tipo	Longitud	Descripción
fecha	DATETIME	14	Fecha y hora en la cual se realizó la devolución.
medio_pago	VARCHAR	10	Medio de pago por el cual se devuelve el dinero.
monto_devuelto	INTEGER	9	Monto total devuelto.

Fuente: Elaboración propia.

2.2.7 Tabla “reservacion”

Nombre Lógico: Reservacion.

Nombre Físico: reservacion.

Descripción: Tabla que registrará las reservaciones realizadas por los usuarios en el sistema.

Clave(s) Primaria(s): cod_reservacion.

Clave(s) Foránea(s): cod_cliente hace referencia a tabla cliente y su campo cod_cliente.

Clave(s) Foránea(s): cod_usuario hace referencia a la tabla usuario y su campo cod_usuario.

Longitud de Registro: 115 caracteres.

Campos:

Tabla- 2.10 Descripción de campos Tabla “reservacion”.

Nombre	Tipo	Longitud	Descripción
cod_reservacion	MEDIUMINT	7	Código de la reservación que lo identificará en el sistema.
cod_cliente	MEDIUMINT	7	Código del cliente que lo identificará en el sistema y lo ligará a una reservación.
nro_doc_Cli	VARCHAR	20	Número de documento del cliente (RUT o pasaporte).
cod_usuario	VARCHAR	9	Código del usuario que lo identificará en el sistema y lo ligará a una reserva.

Nombre	Tipo	Longitud	Descripción
fecha_reservacion	DATETIME	14	Fecha y hora en la que se realiza la reservación.
fecha_inicio	DATETIME	14	Fecha en la que comienza la reservación.
fecha_fin	DATETIME	14	Fecha en la que termina la reservación.
numero_personas	TINYINT	3	Valor que permitirá identificar el número de personas que se hospedarán.
valor_noche_venta	INTEGER	9	Valor de la habitación por noche.
estado_reserva	TINYINT	2	Estado de la reserva (00:“Confirmada” o 01:“Pendiente” o 02:“Cancelada” o 03:“Finalizada” o 04:“Extendida” o 05:“ <i>No show</i> ” o 06:“No finalizada”).
tipo_reserva	TINYINT	2	Tipo de la reserva (00:“Directa” o 01:“OTA” o 02:“Email” o 03:“Teléfono”).
fecha_check_in	DATETIME	14	Fecha y hora en la que el cliente llega al Hostal.
fecha_check_out	DATETIME	14	Fecha y hora en la que el cliente se va del Hostal.

Fuente: Elaboración propia.

2.2.8 Tabla “pago”

Nombre Lógico: Pago.

Nombre Físico: pago.

Descripción: Tabla que registra los pagos realizados en el sistema ligados a una reservación

Clave(s) Primaria(s): cod_pago.

Clave(s) Foránea(s): cod_moneda_pago hace referencia a tabla moneda y su campo cod_moneda.

Clave(s) Foránea(s): cod_reserva hace referencia a tabla reserva y su campo cod_reserva.

Longitud de Registro: 56 caracteres.

Campos:

Tabla- 2.11 Descripción de campos Tabla “pago”.

Nombre	Tipo	Longitud	Descripción
cod_pago	MEDIUMINT	7	Código correlativo que identifica al pago.
id_pago	VARCHAR	20	Código del pago que lo identificará en el sistema si fue por deposito o tarjeta de crédito.
cod_reserva	MEDIUMINT	7	Código de la reserva que lo identificará en el sistema y lo ligará a un pago.
fecha_pago	DATETIME	14	Fecha y hora en la que se realizó el pago.
medio_pago	TINYINT	2	Medio por el cual se realizó el pago (Transferencia, deposito, efectivo).
monto	INTEGER	9	Monto de dinero del pago.
cod_moneda_pago	SMALLINT	4	Código de la moneda con la cual se realizó el pago.

Fuente: Elaboración propia.

2.2.9 Tabla “moneda”

Nombre Lógico: Moneda.

Nombre Físico: moneda.

Descripción: Tabla que registra las monedas que manejará el sistema.

Clave(s) Primaria(s): cod_moneda.

Longitud de Registro: 14 caracteres.

Campos:

Tabla- 2.12 Descripción de campos Tabla “moneda”.

Nombre	Tipo	Longitud	Descripción
cod_moneda	SMALLINT	4	Código de la moneda que lo identificará en el sistema.
nom_moneda	VARCHAR	10	Nombre de la moneda.

Fuente: Elaboración propia.

2.2.10 Tabla “valor_moneda”

Nombre Lógico: Valor moneda.

Nombre Físico: valor_moneda.

Descripción: Tabla que registra el valor de compra y venta actual de la moneda.

Clave(s) Primaria(s): cod_valor.

Clave(s) Foránea(s): cod_moneda hace referencia a tabla moneda y su campo cod_moneda.

Longitud de Registro: 20 caracteres.

Campos:

Tabla- 2.13 Descripción de campos Tabla “valor_moneda”.

Nombre	Tipo	Longitud	Descripción
cod_valor	SMALLINT	5	Código del valor de la moneda que lo identificará en el sistema.
cod_moneda	SMALLINT	4	Código de la moneda que lo identificará en el sistema.
fecha	DATE	8	Fecha del valor de la moneda.
valor_compra	FLOAT	4(4,1)	Valor de compra actual de la moneda.

valor_venta	FLOAT	4(4,1)	Valor de venta actual de la moneda.
-------------	-------	--------	-------------------------------------

Fuente: Elaboración propia.

2.2.11 Tabla “dias reservada”

Nombre Lógico: Días reservada.

Nombre Físico: dias_reservada.

Descripción: Tabla que registra una habitación reservada a través del tiempo.

Clave(s) Primaria(s): cod_dia_reservada.

Clave(s) Foránea(s): cod_reservacion hace referencia a tabla reserva y su campo cod_reserva.

Clave(s) Foránea(s): cod_habitacion hace referencia a tabla habitacion y su campo cod_habitacion.

Longitud de Registro: 33 caracteres.

Campos:

Tabla- 2.14 Descripción de campos Tabla “Dias_reservada”.

Nombre	Tipo	Longitud	Descripción
cod_dia_reservada	MEDIUMINT	7	Código incremental para identificar la habitación reservada a través del tiempo en el sistema.
cod_reservada	MEDIUMINT	7	Código de la reserva que la identifica en el sistema y la liga a una habitación a través del tiempo.
cod_habitacion	TINYINT	3	Código de la habitación que lo identificará en el sistema y lo liga a una reserva a través del tiempo.
fecha_reservada	DATETIME	14	Fecha y hora de la habitación reservada.
estado_asignacion	TINYINT	2	Estado de asignación de este registro (0:“vigente”, 1:“eliminada”) y futuros estados.

Fuente: Elaboración propia.

2.2.12 Tabla “habitacion”

Nombre Lógico: Habitación.

Nombre Físico: habitacion.

Descripción: Tabla que registra cada habitación en el sistema.

Clave(s) Primaria(s): cod_habitacion.

Clave(s) Foránea(s): cod_tipo_habitacion hace referencia a tabla tipo_habitacion y su campo cod_tipo_habitacion.

Longitud de Registro: 48 caracteres.

Campos:

Tabla- 2.15 Descripción de campos Tabla “habitacion”.

Nombre	Tipo	Longitud	Descripción
cod_habitacion	TINYINT	3	Código de la habitación que la identificará en el sistema.
nom_hab	VARCHAR	30	Nombre con el cual se registrará la habitación.
posicion	VARCHAR	10	Posición en la que se encuentra físicamente la habitación dentro del hostel.
cod_tipo_habitacion	TINYINT	3	Código del tipo de la habitación que lo identificará en el sistema.
estado_hab	TINYINT	2	Valor que identificará si la habitación se encuentra eliminada de manera lógica en el sistema, se podrían añadir más estados en el futuro.

Fuente: Elaboración propia.

2.2.13 Tabla “tipo_habitacion”

Nombre Lógico: Tipo habitación.

Nombre Físico: tipo_habitacion.

Descripción: Tabla que registra los tipos de habitación en el sistema.

Clave(s) Primaria(s): cod_tipo_habitacion.

Longitud de Registro: 53 caracteres.

Campos:

Tabla- 2.16 Descripción de campos Tabla “tipo_habitacion”.

Nombre	Tipo	Longitud	Descripción
cod_tipo_habitacion	TINYINT	3	Código del tipo de habitación que lo identificará en el sistema.
nom_tipo	VARCHAR	15	Nombre del tipo de la habitación.
capacidad_habitacion	TINYINT	3	Capacidad de personas de la habitación.
cantidad_baños	TINYINT	2	Cantidad de baños de la habitación.
tipo_baño	VARCHAR	15	Tipo de baño de la habitación.
precio_noche	INTEGER	9	Precio por noche de la habitación.
cantidad_habitaciones	TINYINT	3	Cantidad de habitaciones por tipo de habitación.
estado_tipo_habitacion	TINYINT	3	Valor que identificara si el tipo de habitación se encuentra eliminado de manera lógica en el sistema(0:“vigente”, 1:“eliminada”) y futuros estados.

Fuente: Elaboración propia.

2.2.14 Tabla “bloqueo”

Nombre Lógico: Bloqueo.

Nombre Físico: bloqueo.

Descripción: Tabla que registra las habitaciones bloqueadas a través del tiempo.

Clave(s) Primaria(s): cod_bloqueo.

Clave(s) Foránea(s): `cod_habitacion` hace referencia a tabla habitación y su campo `cod_habitacion`.

Longitud de Registro: 28 caracteres.

Campos:

Tabla- 2.17 Descripción de campos Tabla “bloqueo”.

Nombre	Tipo	Longitud	Descripción
<code>Cod_bloqueo</code>	MEDIUMINT	7	Código incremental que identifica el bloqueo de una habitación a través del tiempo en el sistema.
<code>fecha_inicio</code>	DATE	8	Fecha inicio de cuando se bloqueó la habitación.
<code>fecha_fin</code>	DATE	8	Fecha de fin del bloqueo de la habitación.
<code>cod_habitacion</code>	TINYINT	3	Código de la habitación a bloquear.
<code>estado_bloq</code>	TINYINT	2	Valor que identificará si el bloqueo se encuentra eliminado de manera lógica(0:“vigente”, 1:“eliminada”) y futuros estados.

Fuente: Elaboración propia.

2.2.15 Tabla “anuncio”

Nombre Lógico: Anuncio.

Nombre Físico: anuncio.

Descripción: Tabla que registra los anuncios publicados en las distintas otas.

Clave(s) Primaria(s): `cod_anuncio`.

Clave(s) Foránea(s): `cod_tipo_habitacion` hace referencia a tabla tipo_habitacion y su campo `cod_tipo_habitacion`.

Clave(s) Foránea(s): `cod_usuario_ota` hace referencia a tabla usuario_ota y su campo `cod_usuario_ota`.

Longitud de Registro: 55 caracteres.

Campos:

Tabla- 2.18 Descripción de campos Tabla “anuncio”.

Nombre	Tipo	Longitud	Descripción
Cod_anuncio	MEDIUMINT	7	Código incremental que identifica el anuncio de las habitaciones de un tipo en un periodo de tiempo.
Cod_tipo_habitacion	TINYINT	3	Código del tipo de habitaciones publicitadas.
Cod_usuario_ota	MEDIUMINT	7	Código del usuario que realizó la publicación en una OTA.
Precio_noche_anuncio	INTEGER	9	Precio referencial por noche publicado en el anuncio
Fecha_inicio	DATETIME	14	Fecha inicio de vigencia del anuncio.
Fecha_fin	DATETIME	14	Fecha fin de vigencia del anuncio.

Fuente: Elaboración propia.

2.2.16 Tabla “usuario_ota”

Nombre Lógico: Usuario OTA.

Nombre Físico: usuario_ota.

Descripción: Tabla que registra la información de los usuarios de las distintas OTA's.

Clave(s) Primaria(s): cod_usuario_ota.

Clave(s) Foránea(s): cod_usuario hace referencia a tabla usuario y su campo cod_usuario.

Clave(s) Foránea(s): cod_ota hace referencia a la tabla ota y su campo cod_ota.

Longitud de Registro: 473 caracteres.

Campos:

Tabla- 2.19 Descripción de campos Tabla “usuario_ota”.

Nombre	Tipo	Longitud	Descripción
cod_usuario_ota	MEDIUMINT	7	Código del usuario registrado en una OTA.
cod_usuario	VARCHAR	20	Código del usuario del sistema dueño de la cuenta del usuario_ota.
cod_ota	TINYINT	3	Código de la OTA donde está registrado el usuario.
correo_electronico	VARCHAR	100	Correo ligado al registro de la cuenta.
nombre_usuario	VARCHAR	20	Nombre de usuario registrado en la ota.
identificacion	VARCHAR	20	Credencial requerida para el acceso a la cuenta.
contraseña	VARCHAR	100	Contraseña del usuario requerida para el acceso a la cuenta.
api_key	VARCHAR	100	Llave de acceso requerida para el uso de los endpoints de la OTA
access_token	VARCHAR	100	Token de autenticación del usuario para uso de peticiones PUT y POST en la OTA
estado_usuario	TINYINT	3	Valor que identificará si el usuario ota se encuentra eliminado de manera lógica (0:“vigente”, 1:“eliminada”) y futuros estados.

Fuente: Elaboración propia.

2.2.17 Tabla “ota”

Nombre Lógico: OTA.

Nombre Físico: ota.

Descripción: Tabla que registra las OTAs utilizadas en el sistema.

Clave(s) Primaria(s): cod_ota.

Longitud de Registro: 58 caracteres.

Campos:

Tabla- 2.20 Descripción de campos Tabla “OTA”.

Nombre	Tipo	Longitud	Descripción
cod_ota	TINYINT	3	Código incremental que identifica a una OTA en el sistema.
nom_ota	VARCHAR	15	Nombre de la OTA.
url_ota	VARCHAR	30	Dirección web de la página principal de la OTA
estado_ota	TINYINT	3	Valor que identificara si la OTA se encuentra eliminada de manera lógica (0:”vigente”, 1:”eliminada”) y futuros estados.

Fuente: Elaboración propia.

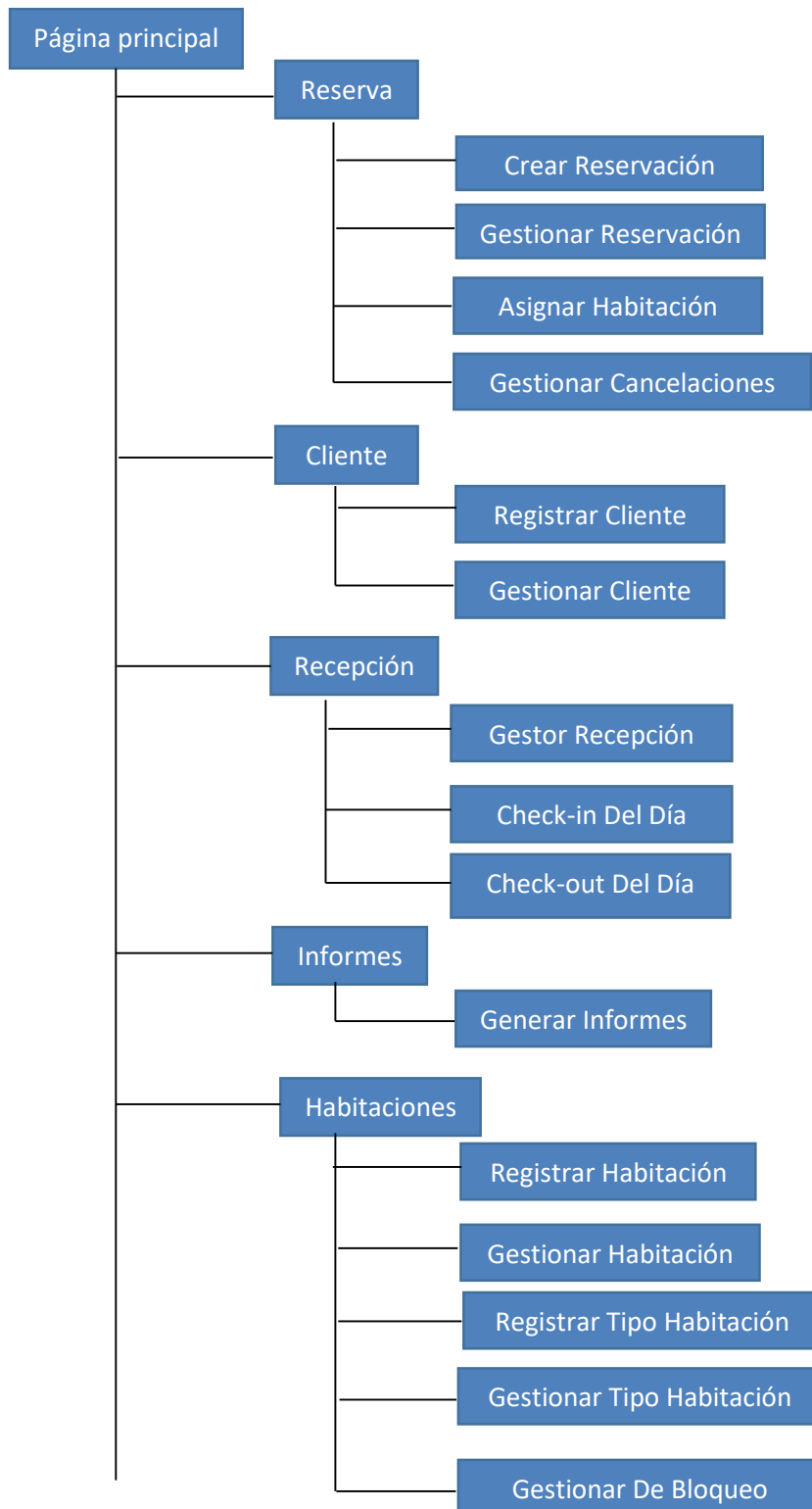
CAPÍTULO 3: DESCRIPCIÓN DE PROGRAMAS

3. DESCRIPCIÓN DE PROGRAMAS

3.1 DIAGRAMAS DE MENÚS

A continuación, se mostrará en un diagrama las opciones que se despliegan en el menú del programa dependiendo del perfil del usuario.

Usuario: Administrador



El diagrama que se encuentra en esta página es la continuación del anterior.

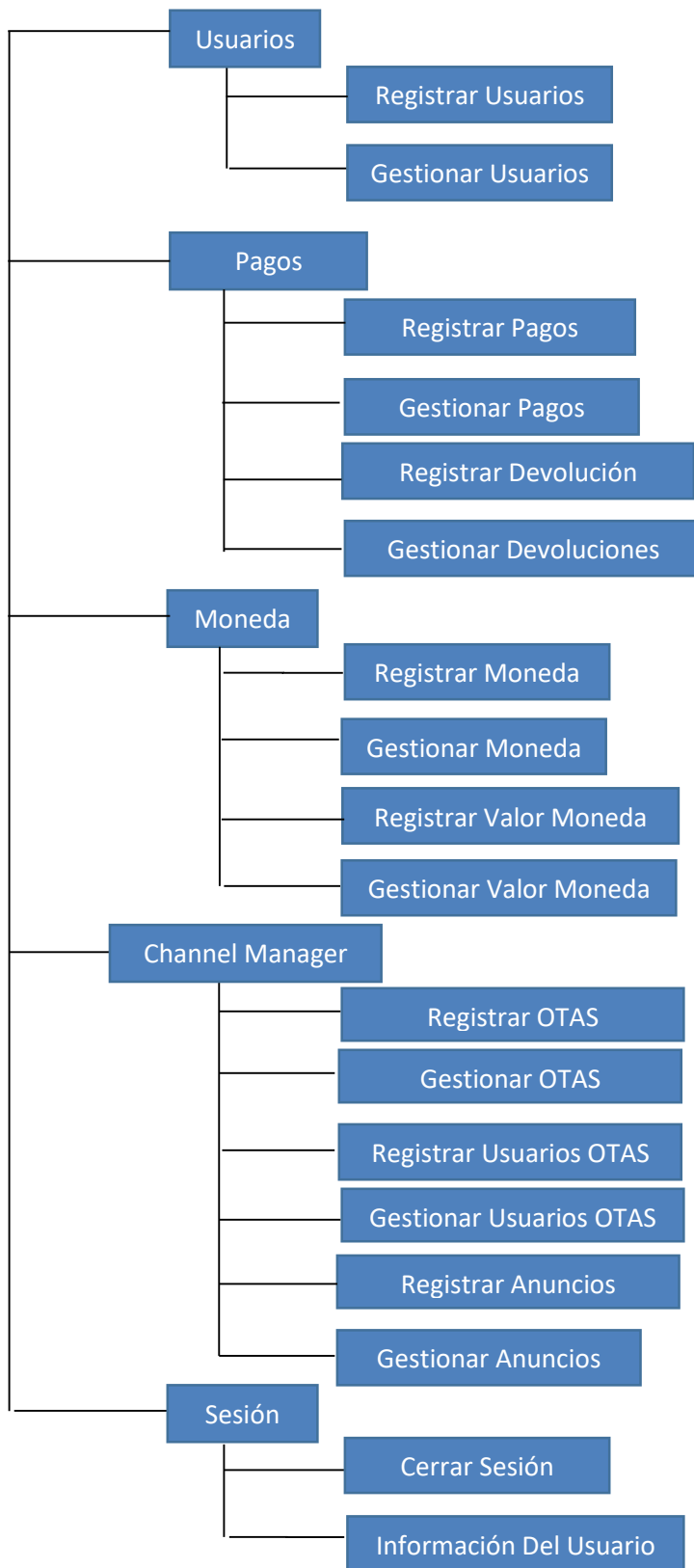
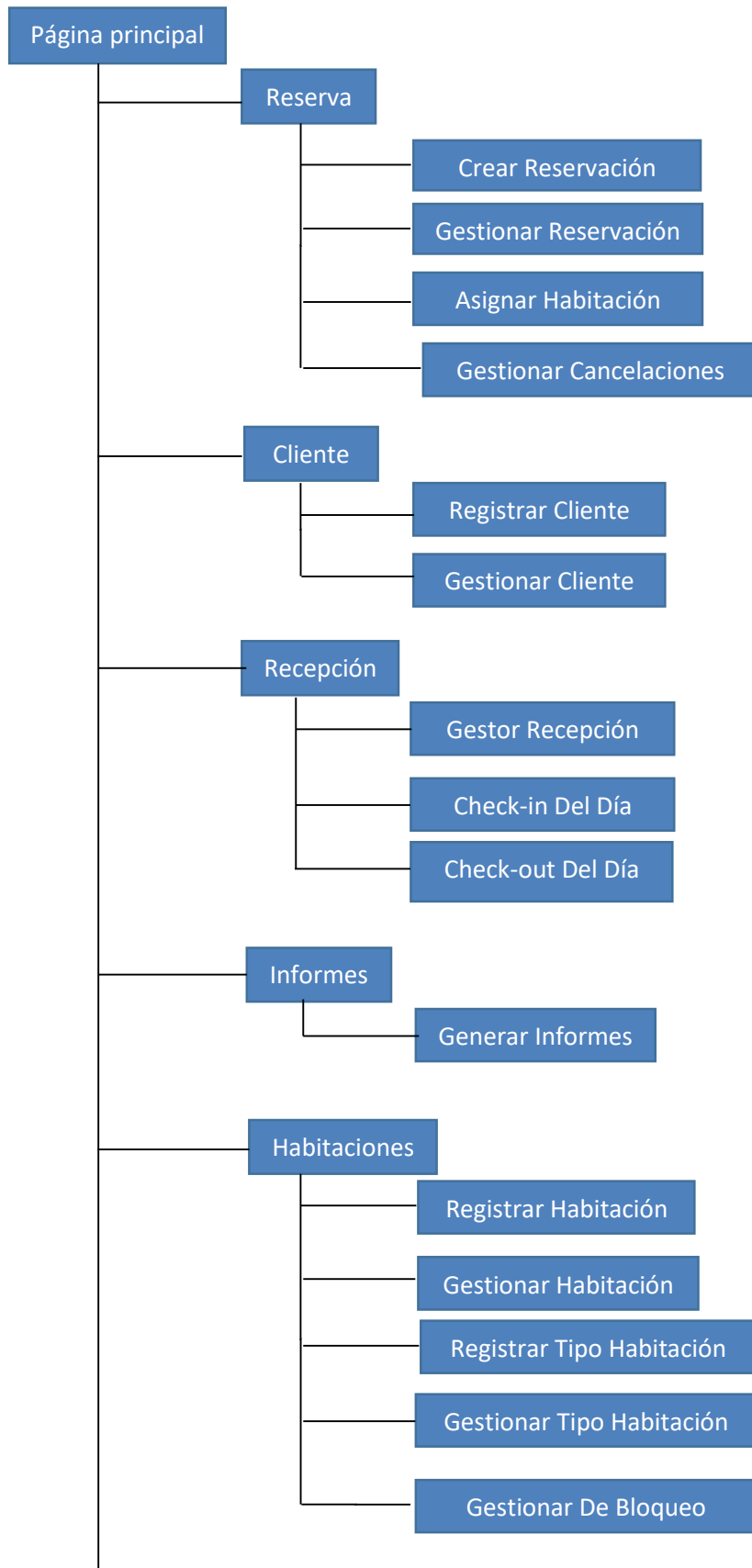


Diagrama 3.1 Diagrama de menú administrador.

Fuente: Elaboración propia.

Usuario: Recepcionista



El diagrama que se encuentra en esta página es la continuación del anterior.

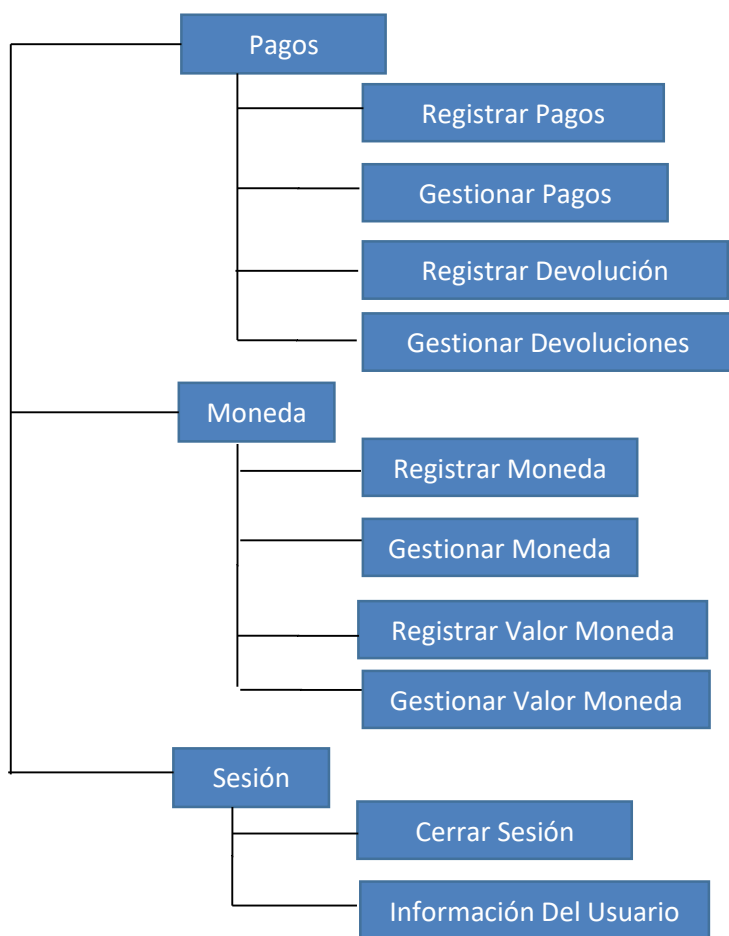


Diagrama 3.2 Diagrama de menú recepcionista.

Fuente: Elaboración propia.

3.2 DIAGRAMA MODULAR

A continuación, se mostrará el diagrama modular del sistema desarrollado.



Diagrama 3.3 Diagrama modular.

Fuente: Elaboración propia.

El diagrama que se encuentra en esta página es una continuación del anterior.



3.3. PROGRAMAS DEL SISTEMA

A continuación, se nombrarán los programas del sistema. Los programas con el símbolo * en un costado serán descritos con más detalles más adelante.

Tabla 3.1 Programas del sistema.

Nombre del programa	Objetivo
Hostal/login*	Permite el ingreso de un usuario al sistema
Hostal/home*	Página principal del sistema, en la cual se desplegará la última reservación realizada.
Hostal/clientes/create	Registra a un nuevo cliente en el sistema.
Hostal/clientes/index*	Consulta los clientes registrados en el sistema a través de distintos parámetros.
Hostal/clientes/{id}/edit	Modifica los datos de un cliente en específico.
Hostal/clientes/{id}/destroy	Elimina a un cliente de manera lógica o física del sistema.
Hostal/moneda/create	Registra una nueva moneda en el sistema
Hostal/moneda/index	Consulta las distintas monedas que se encuentran registradas en el sistema.
Hostal/moneda/{id}/edit	Modifica los datos de una moneda.
Hostal/moneda/{id}/destroy	Elimina de manera lógica o física una moneda registrada en el sistema.
Hostal/valorMoneda/create	Registra el valor de una moneda en una fecha específica en el sistema.
Hostal/valorMoneda/index	Consulta los distintos valores asignados a una o varias monedas.
Hostal/valorMoneda/{id}/edit	Modifica el precio y fecha del valor de una moneda en el sistema.
Hostal/valorMoneda/{id}/destroy	Elimina de manera física o lógica el valor de una moneda en el sistema.
Hostal/pago/create	Registra un pago asociado a una reservación en el sistema.
Hostal/pago/index	Consulta los pagos asociados a una reservación registrada en el sistema a través de distintos parámetros.
Hostal/pago/{id}/edit	Modifica los datos de un pago realizado en el sistema.
Hostal/pago/{id}/destroy*	Elimina de manera lógica o física un pago en el sistema.
Hostal/devolucion/create	Registra una devolución de dinero asociada a una reservación en el sistema.
Hostal/devolucion/index	Consulta las devoluciones registradas en el sistema.
Hostal/devolucion/{id}/edit	Modifica los datos de una devolución en el sistema.
Hostal/devolucion/{id}/destroy	Elimina una devolución del sistema.
Hostal/usuarios/create	Registra un nuevo usuario en el sistema.
Hostal/usuarios/index	Consulta a los usuarios registrados en el sistema.

Nombre del programa	Objetivo
Hostal/usuarios/{id}/edit	Modifica los datos de los usuarios en el sistema.
Hostal/usuarios/{id}/destroy	Elimina de manera física o lógica un usuario en el sistema.
Hostal/tipoHabitacion/create	Registra un nuevo tipo de habitación en el sistema.
Hostal/tipoHabitacion/index	Consulta los distintos tipos de habitación registrados en el sistema.
Hostal/tipoHabitacion/{id}/edit	Modifica los datos de un tipo de habitación en el sistema.
Hostal/tipoHabitacion/{id}/destroy	Elimina de manera física o lógica un tipo de habitación en el sistema.
Hostal/habitacion/create	Registra una nueva habitación en el sistema.
Hostal/habitacion/index	Consulta las habitaciones registradas en el sistema.
Hostal/habitacion/{id}/edit*	Modifica los datos de una habitación en el sistema.
Hostal/habitacion/{id}/destroy	Elimina de manera lógica o física una habitación del sistema.
Hostal/bloquear/{id}/create	Registra el bloqueo de una habitación en un rango de fechas en el sistema.
Hostal/bloquear/index	Consulta los bloqueos realizados a las habitaciones.
Hostal/bloquear/{id}/edit	Modifica la fecha en la que se realizará el bloqueo de una habitación.
Hostal/bloquear/{id}/destroy	Elimina de manera lógica o física un bloqueo en el sistema.
Hostal/recepcion/index	Registra los <i>check-in</i> y <i>check-out</i> que se deben realizar ese día, además de poder realizar los <i>no-shows</i> de las reservaciones pendientes.
Hostal/recepcion/index_in	Permite consultar los <i>check-in</i> realizados en el día, así como eliminarlos.
Hostal/recepcion/index_out	Permite consultar los <i>check-out</i> realizados en el día, así como eliminarlos.
Hostal/reservacion/create*	Registra una nueva reservación en el sistema
Hostal/reservacion/index	Consulta las reservaciones realizadas en el sistema.
Hostal/reservacion/{id}/edit	Modifica el rango de estadía y las habitaciones utilizadas en una reservación.
Hostal/reservacion/{id}/destroy	Elimina de manera física o lógica una reservación del sistema.
Hostal/cancelar/{id}/create	Realiza la cancelación de una reservación.
Hostal/cancelar/index	Consulta las cancelaciones realizadas.
Hostal/cancelar/{id}/destroy	Elimina de manera física y lógica las cancelaciones realizadas.
Hostal/asignar/index*	Permite cambiar la ocupación de las habitaciones.
Hostal/ota/create	Registra una nueva OTA en el sistema.
Hostal/ota/index	Consulta las OTAs registradas en el sistema.

Nombre del programa	Objetivo
Hostal/ota/{id}/edit	Modifica los datos de una OTA en el sistema.
Hostal/ota/{id}/destroy	Elimina de manera física o lógica una OTA del sistema.
Hostal/usuario_ota/create	Registra un nuevo usuario para una OTA.
Hostal/usuario_ota/index	Consulta los usuarios de las OTAs registrados en el sistema.
Hostal/usuario_ota/{id}/edit	Modifica los datos de un usuario.
Hostal/usuario_ota/{id}/destroy	Elimina de manera lógica o física un usuario de una OTA del sistema.
Hostal/anuncio/create	Registra el anuncio de una habitación que se desplegará en una OTA.
Hostal/anuncio/index	Consulta los anuncios registrados en el sistema.
Hostal/anuncio/{id}/edit	Modifica los datos de un anuncio en el sistema.
Hostal/anuncio/{id}/destroy	Elimina de manera lógica o física un anuncio en el sistema.
Hostal/logout	Permite cerrar la sesión del usuario conectado.
Hostal/gestolformes	Permite elegir el tipo de informe o lista que se desea generar.
Hostal/listas/clientes_index	Obtendrá los parámetros necesarios para generar una lista de los clientes registrados en el sistema.
Hostal/listas/clientesHospedados_index*	Obtendrá los parámetros necesarios para generar una lista de los clientes hospedados en un rango de fechas.
Hostal/listas/habitaciones_index	Obtendrá los parámetros necesarios para generar una lista con las habitaciones registradas en el sistema.
Hostal/listas/habitacionesDisponibles_index	Obtendrá los parámetros necesarios para generar una lista con la disponibilidad de las habitaciones en una fecha en específico.
Hostal/listas/tipoHabitaciones_index	Obtendrá los parámetros necesarios para generar una lista con los tipos de habitaciones registrados en el sistema.
Hostal/listas/reservaciones_index	Obtendrá los parámetros necesarios para generar una lista con las reservaciones realizadas en una fecha en específico.
Hostal/listas/pagos_index	Obtendrá los parámetros necesarios para generar una lista con los pagos realizados en fechas específicas.
Hostal/listas/pagosPendientes_index	Obtendrá los parámetros necesarios para generar una lista con los pagos pendientes.
Hostal/listas/devoluciones_index	Obtendrá los parámetros necesarios para generar una lista con las devoluciones realizadas en fechas específicas.
Hostal/listas/cancelaciones_index	Obtendrá los parámetros necesarios para generar una lista con las cancelaciones realizadas en fechas específicas.
Hostal/listas/divisas_index	Obtendrá los parámetros necesarios para generar una lista con las monedas registradas en

	el sistema.
Nombre del programa	Objetivo
Hostal/listas/valorDivisa_index	Obtendrá los parámetros necesarios para generar una lista con los valores de las monedas en fechas específicas.
Hostal/listas/actividadUsuario_index	Obtendrá los parámetros necesarios para generar una lista con la actividad de los usuarios en fechas específicas.
Hostal/listas/checkIn_index	Obtendrá los parámetros necesarios para generar una lista con los <i>check-in</i> realizados en un día en específico.
Hostal/listas/checkOut_index	Obtendrá los parámetros necesarios para generar una lista con los <i>check-out</i> realizados en un día en específico.
Hostal/listas/anuncios_index	Obtendrá los parámetros necesarios para generar una lista con los anuncios registrados en el sistema.
Hostal/informes/reservas_index	Obtendrá los parámetros necesarios para generar un informe detallado con la información de las reservaciones realizadas en un rango de fechas y organizadas de manera semanal, mensual o anual.
Hostal/informes/habitaciones_index	Obtendrá los parámetros necesarios para generar un informe detallado con la información de las habitaciones y sus estadías en un rango de fechas y organizadas de manera semanal, mensual o anual.
Hostal/informes/clientes_index	Obtendrá los parámetros necesarios para generar un informe detallado con la información de los clientes y sus estadías en un rango de fechas y organizadas de manera semanal, mensual o anual.
Hostal/informes/pago_index	Obtendrá los parámetros necesarios para generar un informe detallado con la información de los pagos realizados en un rango de fechas y organizadas de manera semanal, mensual o anual.

Fuente: Elaboración propia.

3.4 DESCRIPCIÓN DE PROGRAMAS

3.4.1 Inicio de Sesión.

Nombre: Hostal/reservacion/create*

Objetivo: Controlar el acceso al sistema por parte de los usuarios.

Diagrama de bloques

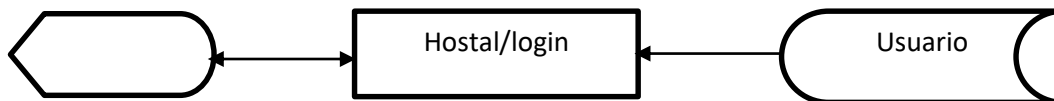


Diagrama 3.4 Diagrama de bloques Inicio de sesión.

Fuente: Elaboración propia.

Reglas de proceso

Se muestra una pantalla para el ingreso de datos del usuario, éstos son validados que existan en tabla Usuario.

SI (Valores existen) ENTONCES

Hace el ingreso y se redirige al menú principal

SINO Mostrar mensaje “Usuario o contraseña ingresadas incorrectas”.

Capturas de pantalla

Figura 3.1 Pantalla de inicio de sesión.

Hostal Viña Bella

Usuario o contraseña ingresadas incorrectas

Figura 3.2 Pantalla de error en inicio de sesión.

3.4.2 Modificar Habitación.

Nombre: [Hostal/habitación/{id}/edit](#)

Objetivo: Modifica los datos de las habitaciones registradas en el sistema.

Diagrama de bloques

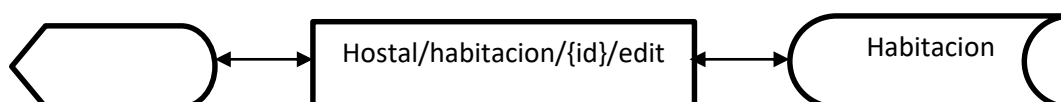


Diagrama 3.5 Diagrama de bloques modificar habitación.

Fuente: Elaboración propia.

Reglas de proceso

En la pantalla del gestor de habitaciones, se selecciona la habitación a modificar y se da click en el botón modificar, esto dirigirá a un formulario en donde se desplegarán los datos ya existentes de la habitación seleccionada para poder modificarlos. Al confirmar la modificación de los datos, los datos ya existentes de la habitación serán modificados por los nuevos datos ingresados.

Capturas de pantalla

Motor de reserva
Hostal Viña Bella

Inicio Reservaciones ▾ Clientes ▾ Recepción ▾ Informes Habitaciones ▾ Usuarios ▾ Pagos ▾ Monedas ▾ Channel Manager ▾ Sesión ▾

Ingrese datos de la habitación a modificar

Nombre:	Posición:	modificar
Departamento 8	primer piso primera puerta a la izquierda	
Campo obligatorio	Campo obligatorio	

Figura 3.3 Pantalla de formulario modificar habitación.

Fuente: Elaboración propia.

Motor de reserva
Hostal Viña Bella

Inicio Reservaciones ▾ Clientes ▾ Recepción ▾ Informes Habitaciones ▾ Usuarios ▾ Pagos ▾ Monedas ▾ Channel Manager ▾ Sesión ▾

Se ha modificado la habitacion de forma exitosa

Ingrese datos a consultar

Código: 	Nombre: 	Capacidad: 	Cantidad de baños:
Tipo habitación: 	Tipo de baño: 	Valor por noche: 	Consultar

Figura 3.4 Pantalla de mensaje satisfactorio de habitación modificada.

Fuente: Elaboración propia.

3.4.3 Registrar Reservación.

Nombre: Hostal/reservacion/create

Objetivo: Registrar una nueva reservación en el sistema, y la ocupación de las habitaciones entre las fechas que dura la reservación.

Diagrama de bloques

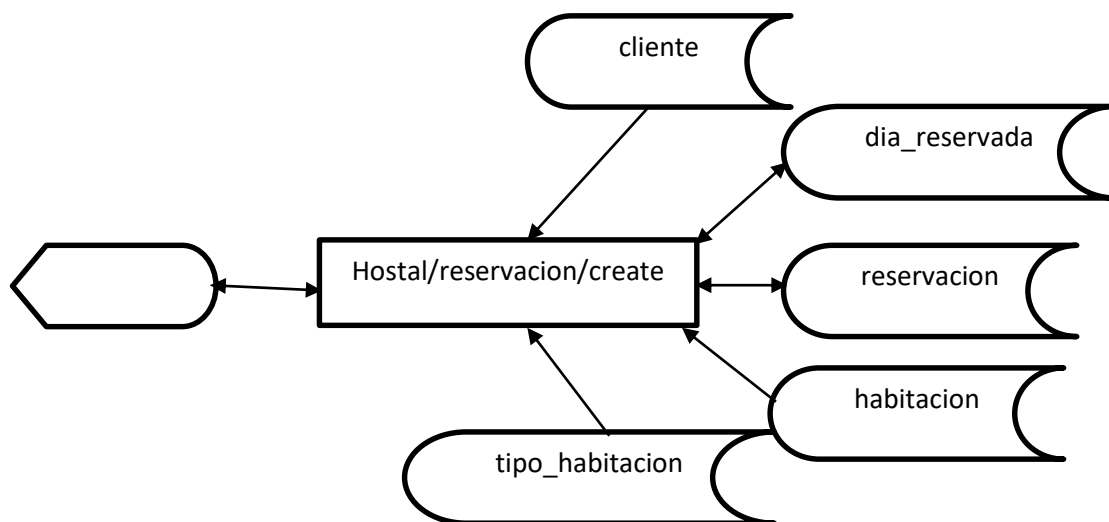


Diagrama 3.6 Diagrama de bloques registrar reservación.

Fuente: Elaboración propia.

Reglas de proceso

En la pantalla de menú, al presionar la opción “Reservaciones” se despliega un submenú donde se selecciona “Registrar reservación”, se dirige a la página con el formulario para registrar la reservación.

Se ingresan los datos del cliente junto a la fecha para la cual se desea registrar la reservación y se realizan las siguientes validaciones:

SI (Fecha es menor a la fecha actual actual) ENTONCES

Mostrar mensaje “La fecha de reservación no puede ser menor a la actual”

SINO SI (Fecha está vacío) ENTONCES

Mostrar mensaje “Debe ingresar una fecha”

SINO

Se despliega una tabla con los clientes que concuerdan con los datos ingresados, al seleccionar uno, se despliega un selector en el cual mostrará las opciones para el tipo de habitación a la cual se le quiere asignar la reservación.

Al seleccionar un tipo de habitación, se despliega un calendario ubicado en las fechas seleccionadas con las habitaciones que posee este tipo de habitación y las reservaciones que se les han asignado.

Al hacer click en la fila de una habitación, se realizarán las siguientes validaciones:

Si (La habitación ya posee reservaciones en fechas similares) ENTONCES

Mostrar mensaje “Error, la habitación ya se encuentra reservada para estos días”.

SINO

Se asigna la reservación en el calendario y se despliega una tabla con los datos de la habitación asignada y su valor por noche, al dar click en el botón confirmar se dirige a una vista que desplegará una tabla con los datos recopilados para confirmar la reservación.

Si (Se da click en el botón cancelar) ENTONCES

Se cancela la reservación y se dirige a la pantalla de registrar reservación.

SINO

Se registra la reservación de manera exitosa y se dirige a la pantalla de registrar reservación con el mensaje “Se ha registrado la reservación de manera exitosa”.

Capturas de pantalla

The screenshot shows a web application interface. At the top, a dark header bar contains a navigation menu with items: Inicio, Reservaciones, Clientes, Recepción, Informes, Habitaciones, Usuarios, Pagos, Monedas, Channel Manager, and Sesión. Below the header, a light gray banner displays the title 'Registrar reservación'. The main content area is a form with several input fields and a button. The fields are arranged in two rows. The first row contains: 'Código cliente:' (text input), 'Tipo documento:' (dropdown menu), 'Número de documento:' (text input), and 'Nombre:' (text input). The second row contains: 'Apellido paterno:' (text input), 'Rango de fecha de reservación:' (text input), 'Cantidad de personas' (text input with the value '1'), and 'Tipo reservacion:' (dropdown menu with 'Presencial' selected). A blue button labeled 'Consultar' is located at the bottom left of the form. An error message box is overlaid on the top right of the form, displaying the text '127.0.0.1:8000 dice' and 'Debe ingresar una fecha', with an 'Aceptar' button.

Figura 3.5 Pantalla de mensaje error de fecha vacía al registrar reservación.

Fuente: Elaboración propia.

Motor de reserva

Hostal Viña Bella

Inicio
Reservaciones
Clientes
Recepción
Informes
Habitaciones
Usuarios
Pagos
Monedas
Channel Manager
Sesión

Registrar reservación

Código cliente:

1

Tipo documento:

Carnet

Número de documento:

194695616

Nombre:

Alex

Apellido paterno:

Zuñiga

Rango de fecha de reservación:

2019-01-05 a 2019-01-08

Cantidad de personas

1

Tipo reservación:

Presencial

Consultar

Seleccionar

Buscar

Código	Tipo de documento	Nro° documento	Nombre	Apellido paterno	Apellido materno	Telefono	Correo electronico	Pais	Ciudad
1	carnet	194695616	Alex	Zuñiga	Cordero	+569822147	alex69@gmail.com	Albania	Wangjiang

Código

Tipo de documento

Nro° documento

Nombre

Apellido paterno

Apellido materno

Telefono

Correo electronico

Pais

Ciudad

Mostrando registros del 1 al 1 de un total de 1 registros

Anterior

1

Siguiente

Tipo Habitación:

habitacion cuadruple

Valor por Noche:

35000

<

>

Hoy

enero 2019

Hoy

<

>

Habitaciones	ma 1	mi 2	ju 3	vi 4	sá 5	do 6	lu 7	ma 8	mi 9	ju 10	vi 11	sá 12	do 13	lu 14	ma 15	mi 16	ju 17	vi 18	sá 19	do 20	lu 21	ma 22	mi 23	ju 24	vi 25
Departamento 8	Reservación 43																								
Departamento 2			Reservación 39	Nueva Reservación						Reservación 43															

Eliminar

Buscar

Tipo Habitación	Habitación	Precio
habitacion cuadruple	Departamento 2	35000

Tipo Habitación

Habitación

Precio

Mostrando registros del 1 al 1 de un total de 1 registros

Anterior

1

Siguiente

Precio total:

35000

confirmar

Figura 3.6 Pantalla de registrar reservación.

Fuente: Elaboración propia.

Hostal Viña Bella			
Inicio Reservaciones ▾ Clientes ▾ Recepción ▾ Informes Habitaciones ▾ Usuarios ▾ Pagos ▾ Monedas ▾ Channel Manager ▾ Sesión ▾			
Confirmar reservación			
Nombre cliente completo	Alex Zuñiga Cordero		
Tipo documento	carnet		
Nro* documento	194695616		
País residencia	Albania		
Ciudad residencia	Wangjiang		
Número de contacto	+569822147		
Código de reserva	51		
Fecha inicio reserva	2019-01-05		
Fecha fin reserva	2019-01-08		
Cantidad de personas	1		
#	Habitación	Tipo habitación	Costo por noche
2	Departamento 2	habitacion cuadruple	35000
Total a pagar	105000		
Abono mínimo	52500	Confirmar Cancelar	

Figura 3.7 Pantalla de confirmar reservación.

Fuente: Elaboración propia.

Motor de reserva			
Hostal Viña Bella			
Inicio Reservaciones ▾ Clientes ▾ Recepción ▾ Informes Habitaciones ▾ Usuarios ▾ Pagos ▾ Monedas ▾ Channel Manager ▾ Sesión ▾			
Se ha registrado la reserva de forma exitosa			
Registrar reservación			
Código cliente:	Tipo documento:	Número de documento:	Nombre:
Apellido paterno:	Rango de fecha de reservación:	Cantidad de personas	Tipo reservación:
		1	Presencial ▾
Consultar			

Figura 3.8 Pantalla de mensaje satisfactorio al registrar reservación.

Fuente: Elaboración propia.

127.0.0.1:8000 dice
La fecha ingresada no puede ser menor a la actual

Acceptar

Inicio Reservaciones Clientes Recepción Informes Habitaciones Usuarios Pagos Monedas Channel Manager Sesión

Registrar reservación

Código cliente:

Tipo documento:

Número de documento:

Nombre:

Apellido paterno:

Rango de fecha de reservación:

Cantidad de personas:

Tipo reservación:

Consultar

Figura 3.9 Pantalla de mensaje error al seleccionar fechas de reservación.

Fuente: Elaboración propia.

127.0.0.1:8000 dice
Error, la habitación ya se encuentra reservada para esas fechas

Acceptar

Inicio Reservaciones Clientes Recepción Informes Habitaciones Usuarios Pagos Monedas Channel Manager Sesión

Registrar reservación

Código cliente:

Tipo documento:

Número de documento:

Nombre:

Apellido paterno:

Rango de fecha de reservación:

Cantidad de personas:

Tipo reservación:

Consultar

Tipo Habitación:

Valor por Noche:

< > Hoy

enero 2019

Hoy < >

Habitaciones	ma 1	mi 2	ju 3	vi 4	sá 5	do 6	lu 7	ma 8	mi 9	ju 10	vi 11	sá 12	do 13	lu 14	ma 15	mi 16	ju 17	vi 18	sá 19	do 20	lu 21	ma 22	mi 23	ju 24	vi 25
Departamento 8	Reservación 49																								
Departamento 2			Reservación 50							Reservación 43															

Figura 3.10 Pantalla de mensaje error al asignar habitación.

Fuente: Elaboración propia.

3.4.4 Asignar Habitación.

Nombre: Hostal/asignar/index

Objetivo: Permite cambiar la ocupación de las habitaciones reservadas en un rango de fechas.

Diagrama de bloques

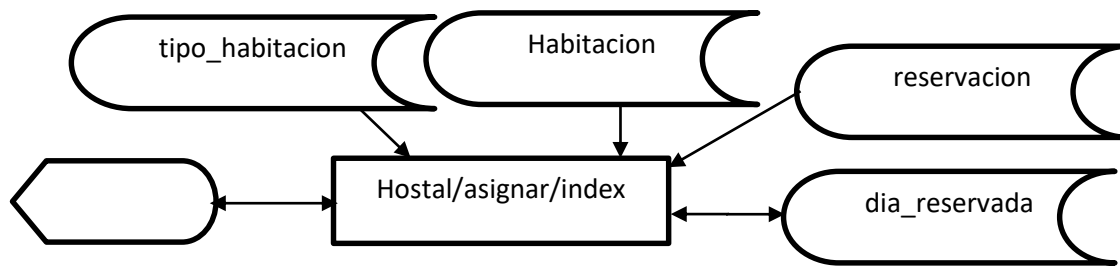


Diagrama 3.7: Diagrama de bloques asignar habitación.

Fuente: Elaboración propia.

Reglas de proceso

En la pantalla de menú, al presionar la opción “Reservaciones” se despliega un submenú donde se selecciona “Asignar habitación”, se dirige a la página con el formulario con un selector con los tipos de habitaciones para asignar habitación.

Se selecciona un tipo de habitación y se despliega un calendario con las habitaciones que posee este tipo de habitación, al arrastrar una reservación y cambiarla de habitación se realizan validaciones de disponibilidad:

SI (La habitación posee reservaciones en un rango de fechas similares) ENTONCES

Se cancela la modificación de la reservación y se regresa a su posición inicial

SINO

Se registra de manera exitosa en el sistema el cambio de habitación.

Capturas de pantalla

Motor de reserva
Hostal Viña Bella

Inicio Reservaciones Clientes Recepción Informes Habitaciones Usuarios Pagos Monedas Channel Manager Sesión

Asignar Habitación

Tipo Habitación:
habitacion cuadruple

< > Hoy

enero 2019

Habitaciones	ma 1	mi 2	ju 3	vi 4	sá 5	do 6	lu 7	ma 8	mi 9	ju 10	vi 11	sá 12	do 13	lu 14	ma 15	mi 16	ju 17	vi 18	sá 19	do 20	lu 21	ma 22	mi 23	ju 24	vi 25
Departamento 8	Reservación 49																								
Departamento 2			Reservación 50							Reservación 43															

Figura 3.11 Pantalla de asignar habitación.

Fuente: Elaboración propia.

3.4.5 Lista de Clientes Hospedados.

Nombre: Hostal/listas/clientesHospedados_index

Objetivo: Obtendrá los datos necesarios para generar una lista con los clientes hospedados en un rango de fechas determinado.

Diagrama de bloques

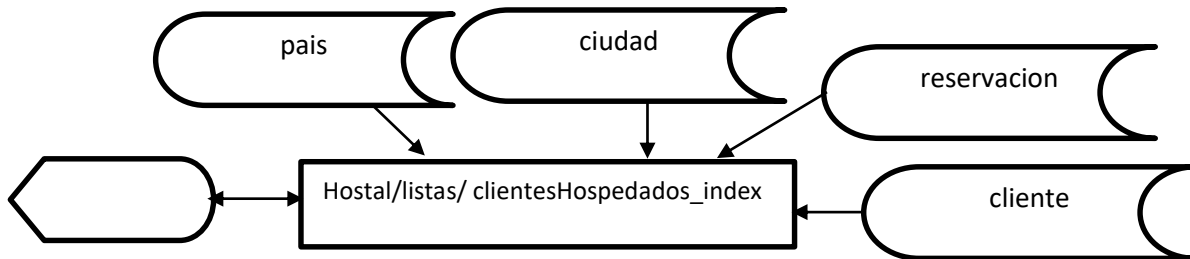


Diagrama 3.8: Diagrama de bloques lista de clientes hospedados.

Fuente: Elaboración propia.

Reglas de proceso

Desde la pantalla de gestor de informes, se selecciona la lista de clientes hospedados y se dirigirá a un formulario en donde se podrá seleccionar los campos que se desean desplegar en la tabla, junto con la fecha en la que se hospedaron.

SI (Fecha contiene un rango de fecha) ENTONCES

Se recopilarán los datos de los clientes que se encuentran hospedados en el rango de fechas seleccionados.

SINO

Se recopilarán los datos de los clientes que se encuentran actualmente hospedados.

Al terminar la recopilación de datos se dirige a la pantalla que contiene la tabla con los resultados obtenidos.

Capturas de pantalla

Motor de reserva
Hostal Viña Bella

Inicio Reservas Clientes Recepción Informes Habitaciones Usuarios Pagos Monedas Channel Manager Sesión

Lista de clientes Hospedados

fecha clientes hospedado:

☒ Código cliente
 ☒ Nombre
 ☒ Apellido paterno
☒ Apellido materno
 ☒ Número de teléfono
 ☒ País
☒ Ciudad
 ☒ Dirección
 ☒ Correo electrónico
☒ Fecha inicio de reserva
 ☒ Fecha de fin de reserva

[Generar Informes](#) [Cancelar](#)

Figura 3.12 Pantalla de gestor lista de clientes hospedados.

Fuente: Elaboración propia.

Motor de reserva
Hostal Viña Bella

Inicio Reservas Clientes Recepción Informes Habitaciones Usuarios Pagos Monedas Channel Manager Sesión

[Exportar a excel](#)

Buscar:

Código cliente	Nombre	Apellido paterno	Apellido materno	Teléfono	País	Ciudad	Dirección	Correo electrónico	Fecha inicio reserva	Fecha fin reserva
1	Alex	Zuñiga	Cordero	+569822147	Albania	Wangjiang	Jose uribe	alex69@gmail.com	2018-12-26 00:00:00	2018-12-31 00:00:00
2	enzo	Hoare	Maria	+56982215929	Pakistan	Tan-Tan	jefe	buuufffff@gmail.com	2018-12-24 00:00:00	2018-12-26 00:00:00
4	Mario	Hoare	Silva	+569822147	Albania	Wangjiang	asdasd	mHoare@gmail.comn	2018-12-23 00:00:00	2018-12-26 00:00:00

Mostrando registros del 1 al 3 de un total de 3 registros

Anterior **1** Siguiente

Figura 3.13 Pantalla de tabla de lista de clientes hospedados.

Fuente: Elaboración propia.

3.4.6 Eliminar Pago

Nombre: Hostal/pago/{id}/destroy

Objetivo: Eliminar un pago asociado a una reserva.

Diagrama de bloques

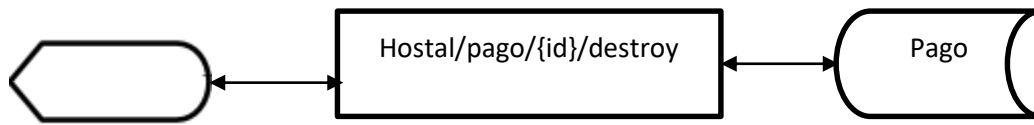


Diagrama 3.9 Diagrama de bloques Eliminar pago.

Fuente: Elaboración propia.

Reglas de proceso

Se ingresa por el gestor de pago y, luego de hacer una búsqueda, si hay resultados se procede a seleccionar un pago para luego presionar el botón eliminar y despliega un modal donde le pide la confirmación de la acción.

SI (Confirma) ENTONCES

Se elimina el pago de la base de datos y de la tabla que visualiza el usuario

SINO

se mantiene el pago en la base de datos y en la tabla que visualiza el usuario.

Capturas de pantalla

Código	Código reserva	Nombre	Apellido paterno	Correo electrónico	Medio de pago	Moneda	Monto	fecha
3	1	marcelo	salas	msilva@gmail.com	Transferencia	CLP	1	07-12-2018
4	1	marcelo	salas	msilva@gmail.com	Transferencia	CLP	100000	27-12-2018

Mostrando registros del 1 al 2 de un total de 2 registros 1 row selected

Anterior 1 Siguiente

Figura 3.14 Pantalla confirmación eliminar pago.

Fuente: Elaboración propia.

Motor de reserva
Hostal Viña Bella

Inicio Reservas Clientes Recepción Informes Habitaciones Usuarios Pagos Monedas Channel Manager Notificaciones Sesión

Ingrese datos a consultar

Código: Código de cliente: Tipo de documento: Número de documento:

Código de reserva: Nombre: Apellido paterno: Tipo de Habitación:

Medio de pago: Buscar por rango de fecha:

Modificar Eliminar Exportar a excel

Buscar:

Código	Código reserva	Nombre	Apellido paterno	Correo electrónico	Medio de pago	Moneda	Monto	fecha
4	1	marcelo	salas	msilva@gmail.com	Transferencia	CLP	100000	27-12-2018

Mostrando registros del 1 al 1 de un total de 1 registros

Figura 3.15 Pantalla eliminar pago.

Fuente: Elaboración propia.

3.4.7 Consultar Clientes

Nombre: Hostal/clientes/index

Objetivo: Ver los datos de los clientes registrados en el sistema.

Diagrama de bloques



Diagrama 3.10 Diagrama de bloques Consultar clientes.

Fuente: Elaboración propia.

Reglas de proceso

Si se quiere buscar por algún parámetro, se rellenan los campos del formulario según se necesite, si no, se dejan en blanco los campos para mostrar de forma paginada todos los clientes del sistema, en ambos casos se presiona el botón consultar para ejecutar la consulta.

SI (Los valores coinciden con registros de la base de datos) ENTONCES

Se envían los clientes que cumplen con los parámetros de búsqueda y se muestran en una tabla de forma paginada.

SINO

se envía una lista sin clientes y se muestra una tabla vacía que indica que no hay resultados.

Capturas de pantalla

Motor de reserva
Hostal Viña Bella

Inicie Reservas Clientes Recepción Informes Habitaciones Usuarios Pagos Monedas Channel Manager Notificaciones Sesión

Ingrese datos a consultar

Código: Tipo de documento: Número de documento: Nombre:

Apellido paterno: Apellido materno: Correo electrónico: País de origen:

Ciudad de origen: Tipo de Habitación: Habitaciones: Buscar por rango de fecha:

[Consultar](#)

[Modificar](#) [Eliminar](#) [Exportar a excel](#)

Buscar:

Código	Tipo de documento	Nro° documento	Nombre	Apellido paterno	Apellido materno	Telefono	Correo electronico	País	Ciudad
1	carne	185443358	marcelo	salas	silva	9983442111	malva@gmail.com	Chile	quique
2	pasaporte	123456789	ricardo	solis	arjona	9983442111	malva@gmail.com	Chile	quique

Mostrando registros del 1 al 2 de un total de 2 registros

[Anterior](#) [1](#) [Siguiente](#)

Figura 3.16 Pantalla consultar cliente tabla con datos.

Fuente: Elaboración propia.

Motor de reserva
Hostal Viña Bella

Inicie Reservas Clientes Recepción Informes Habitaciones Usuarios Pagos Monedas Channel Manager Notificaciones Sesión

Ingrese datos a consultar

Código: Tipo de documento: Número de documento: Nombre:

Apellido paterno: Apellido materno: Correo electrónico: País de origen:

Ciudad de origen: Tipo de Habitación: Habitaciones: Buscar por rango de fecha:

[Consultar](#)

[Modificar](#) [Eliminar](#) [Exportar a excel](#)

Buscar:

Código	Tipo de documento	Nro° documento	Nombre	Apellido paterno	Apellido materno	Telefono	Correo electronico	País	Ciudad
Ningún dato disponible en esta tabla									
Código	Tipo de documento	Nro° documento	Nombre	Apellido paterno	Apellido materno	Telefono	Correo electronico	País	Ciudad

Mostrando registros del 0 al 0 de un total de 0 registros

[Anterior](#) [1](#) [Siguiente](#)

Figura 3.17 Pantalla consultar clientes tabla vacía.

Fuente: Elaboración propia.

3.4.8 Inicio

Nombre: Hostal/home

Objetivo: Pantalla principal del sistema en la cual se desplegará la última reservación realizada en el sistema y el usuario el cual la realizo.

Diagrama de bloques

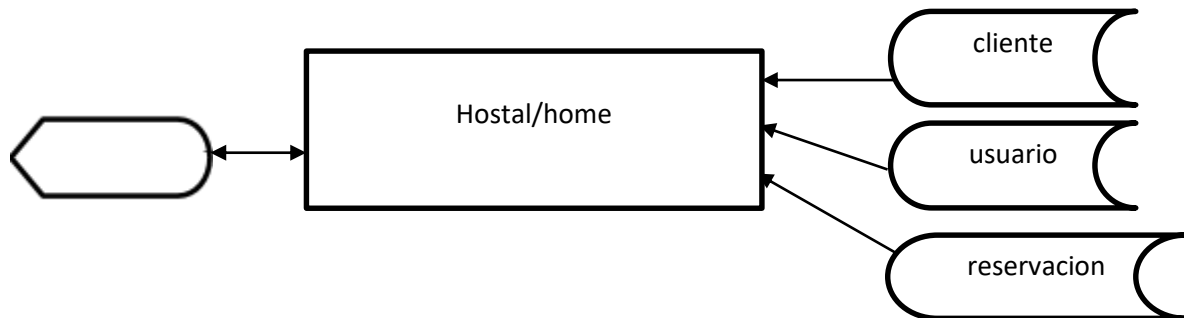


Diagrama 3.11 Diagrama de bloques pantalla de inicio.

Fuente: Elaboración propia.

Reglas de proceso

Al iniciar sesión de manera satisfactoria o dar click en la opción de inicio del menú, se dirigirá a la pantalla de inicio, la cual mostrará un formulario en donde se desplegará la última reservación realizada en el sistema.

Capturas de pantalla



Figura 3.18 Pantalla de inicio.

Fuente: Elaboración propia.

CONCLUSIÓN

El sistema como se ha descrito es una posible solución para el hostel Viña Bella ante su problema sobre el manejo de sus clientes, habitaciones, reservaciones y los pagos de estas.

Con la solución que hemos planteado y realizado, se puede decir que se ha abarcado gran parte de las dificultades con las que se atraviesa el hostel a la hora de realizar reservaciones, facilitando en gran medida el manejo de estas al momento de asignarles una habitación. Aunque si bien el sistema desarrollado a la fecha no contempla la sincronización de las reservaciones realizadas a través de las OTAs con el stock de las habitaciones que se mantiene en el sistema, sí permite registrar los anuncios publicados en estas páginas web.

La experiencia adquirida durante todos estos aproximados 9 meses de trabajo, en los cuales hemos tenido que investigar el funcionamiento del hostel junto a la ayuda de la dueña, como también aprender a utilizar el *framework* Laravel, nos ha ayudado para comprender en cierto modo a los que nos enfrentaremos al momento de iniciar nuestra vida laboral como profesionales.

Después de todo el tiempo que se ha invertido a lo largo de este proyecto, nos hemos dado cuenta de que para afrontar el desarrollo de un sistema de esta envergadura es necesaria una buena comunicación y cohesión entre los miembros de un equipo, para así poder trabajar de la manera óptima.

A futuro se planea mejorar el diseño de las interfaces de usuarios y de los mensajes de errores, para que estas le brinden una mejor experiencia al usuario al momento de utilizar el sistema desarrollado.

Actualmente el sistema no ha sido desplegado, ya que la dueña del hostel utiliza un sistema pagado para poder manejar la información de las reservas entrantes.

BIBLIOGRAFÍA

- Otwell, T. (2018). *Laravel - The PHP Framework For Web Artisans*. [online] Laravel.com. Disponible en: <https://laravel.com/>.
- Aravena Soto, E. and Guerrero Guzmán, R. (2008). *Sistema de venta por internet para todo tipo de empresa*. Viña del mar: USM.
- Shaw, A. (2018). *FullCalendar - JavaScript Event Calendar*. [online] Fullcalendar.io. Disponible en: <https://fullcalendar.io/>.
- Datatables.net. (2017). *DataTables | Table plug-in for jQuery*. [online] Disponible en: <https://datatables.net/>.
- Flatpickr.js.org. (2018). *flatpickr*. [online] Disponible en: <https://flatpickr.js.org/>.

ANEXO A: CODIGOS FUENTES

A continuación, se dejarán incluidos los códigos fuentes de los programas mencionados en la sección 3.4 “Descripción de programas” en el capítulo 3.

Inicio de sesión

Login.blade.php

```
<html lang="en">
<head>
  <!-- Required meta tags -->
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
  <!-- Bootstrap CSS -->
  <link rel="stylesheet"
ref=https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css
integrity="sha384-
MCw98/SFnGE8fJT3GXwEOngsV7Zt27NXFoaoApmYm81iuXoPkFOJwJ8ERdknLPMO"
crossorigin="anonymous">
  <title>Login</title>
</head>
<body>
  <section class="container-fluid ">
    <header class="bg-dark text-white row justify-content-lg-center">
      <h1 class="col-lg-3 col-sm-12 col-md-6 ">Hostal Viña Bella</h1>
    </header>
    <div class="alert alert-danger">Usuario o contraseña ingresada incorrectas</div>
    <div class="row justify-content-lg-center pt-5">
      <div id="login" class="col-lg-3 col-sm-12 col-md-6">
        <div class="card ">
          <div class="card-header">
            <h5>Login</h5>
          </div>
          <div class="card-body">
            <form method="POST" action="{{ route('login') }}">
              {{ csrf_field() }}
              <div class="form-group {{ $errors->has('identificacion') ? 'has-error' : '' }}">
                <label for="exampleInputEmail1">Usuario</label>
                <input type="text"
                  class="form-control"
                  id="identificacion"
                  name="identificacion"
                  value="{{ old('identificacion') }}"
                  required
```

```

        placeholder="Ingrese su usuario">
        {!! $errors->first('identificacion', '<span class="help-block">:message</span>') !!}
    </div>
    <div class="form-group {{ $errors->has('contraseña') ? 'has-error' : '' }}">
        <label for="exampleInputPassword1">Contraseña</label>
        <input type="password" class="form-control" id="contraseña" name="contraseña"
required placeholder="ingrese su Contraseña">
        {!! $errors->first('contraseña', '<span class="help-block">:message</span>') !!}
    </div>
    <button type="submit" class="btn btn-primary">Acceder</button>
</form>
</div>
</div>
</div>
</div>
</section>
<!-- Optional JavaScript -->
<!-- jQuery first, then Popper.js, then Bootstrap JS -->
<script src="https://code.jquery.com/jquery-3.3.1.slim.min.js" integrity="sha384-
q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo"
crossorigin="anonymous"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.3/umd/popper.min.js"
integrity="sha384-ZMP7rVo3mIykV+2+9J3UJ46jBkOWLaUAdn689aCwoqBBIjSnjAK/I8WvCWPIpm49"
crossorigin="anonymous"></script>
<script src="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/js/bootstrap.min.js"
integrity="sha384-ChfqquxuZUCnJSK3+MXmPNIyE6ZbWh2IMqE241rYiqJxyMiZ6OW/JmZQ5stwEULTy"
crossorigin="anonymous"></script>
</body>
</html>

```

LoginController.php

```

<?php
namespace App\Http\Controllers\Auth;
use App\Http\Controllers\Controller;
use Illuminate\Foundation\Auth\AuthenticatesUsers;
use Auth;

class LoginController extends Controller
{
    public function __construct(){
        $this->middleware('guest', ['only' => 'showLoginForm']);
    }

    public function username()
    {

```

```

        return 'identificacion';
    }
    public function showLoginForm(){
        return view('login');
    }
    public function login(){
        $credentials = $this->validate(request(), [
            'identificacion' => 'string',
            'contraseña' => 'string'
        ]);
        if(Auth::attempt(['identificacion' => $credentials["identificacion"], 'password' =>
$credentials["contraseña"]])){
            return redirect()->route('Home');
        }
        return back()->withErrors(['identificacion' => 'Este usuario no figura en el sistema'])
            ->withinputs(request(['identificacion']));
    }
}

```

Modificar habitación

HabitacionController.php

```

<?php
namespace App\Http\Controllers;
use Illuminate\Http\Request;
use App\Habitacion;
use App\Tipo_habitacion;
use App\Dia_reservada;

class HabitacionController extends Controller{
    public function edit($id)
    {
        $habitacion = Habitacion::find($id);
        return view('admin.Habitacion.modificar')->with('habitacion', $habitacion);
    }
    /**
     * Update the specified resource in storage.
     *
     * @param \Illuminate\Http\Request $request
     * @param int $id
     * @return \Illuminate\Http\Response
     */
    public function update(Request $request, $id)
    {

```

```

$habitacion = Habitacion::find($id);
$habitacion->fill($request->all());
$habitacion->save();
flash("Se ha modificado la habitacion de forma exitosa")->success();
return redirect()->route('Habitacion.index');
}
}

```

modificar.blade.php

```

@extends('admin.template.main')
@section('title', 'Modificar Habitación')
@section('content')
    @if(count($errors) > 0)
        <div class="alert alert-danger" role="alert">
            @foreach($errors->all() as $error)
                <li>{{ $error }}</li>
            @endforeach
        </div>
    @endif
    <div class="card bg-dark text-light">
        <div class="card-header ">
            <h4>Ingrese datos de la habitación a modificar</h4>
        </div>
        <div class="card-body">
            {!! Form::open(['route' => ['Habitacion.update', $habitacion], 'method' => 'POST']) !!}
            @method('PUT')
            <div class="row">
                <div class="col-lg-3 col-md-6 col-sm-12">
                    <div class="form-group">
                        {!! Form::label('nombre', 'Nombre:') !!}
                        {!! Form::text('nom_hab', $habitacion->nom_hab, ['class'=>'form-control is-
invalid', 'id' => 'nom_hab', 'maxlength' => "30", 'required']) !!}
                        <div class="invalid-feedback">
                            Campo obligatorio
                        </div>
                    </div>
                </div>
                <div class="col-lg-3 col-md-6 col-sm-12">
                    <div class="form-group">
                        {!! Form::label('calle_res', 'Posicion:') !!}
                        {!! Form::text('posicion', $habitacion->posicion, ['class'=>'form-control is-invalid', 'maxlength'
=> "100", 'required' ]) !!}
                        <div class="invalid-feedback">
                            Campo obligatorio
                        </div>
                    </div>
                </div>
            </div>
        </div>
    </div>

```

```

        </div>
    </div>
    <div class="col-lg-3 col-md-6 col-sm-12">
        <div class="form-group">
            <br>
            {!! Form::submit('modificar', ['class' => 'btn btn-primary']) !!}
        </div>
    </div>
</div>
{!! Form::close() !!}
</div>
</div>
@endsection

```

Registrar reservación

RegistrarReservacion.blade.php

```

@extends('admin.template.main')
@section('title', 'Crear reservacion')
@section('page-css-style')
    <link rel="stylesheet" type="text/css" href="https://cdn.datatables.net/v/bs4/jszip-2.5.0/dt-
    1.10.18/b-1.5.2/b-html5-1.5.2/cr-1.5.0/r-2.2.2/sl-1.2.6/datatables.min.css"/>
    <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/flatpickr/dist/flatpickr.min.css">
    <link rel="stylesheet" href="{{asset('plugin/fullcalendar/css/fullcalendar.min.css')}}"/>
    <link rel="stylesheet" href="{{asset('plugin/fullcalendar/css/scheduler.min.css')}}"/>
@endsection
@if(count($errors) > 0)
    <div class="alert alert-danger" role="alert">
        @foreach($errors->all() as $error)
            <li>{{ $error }}</li>
        @endforeach
    </div>
@endif
<div class="card bg-dark text-light">
    <div class="card-header">
        <h4>Registrar reservación</h4>
    </div>
    <div class="card-body">
        <form>
            <div class="row">
                <div class="col-lg-3 col-md-6 col-sm-12">
                    <div class="form-group">
                        <label for="codigo">Código cliente:</label>

```

```

        <input id="codigo" type="text" class="form-control" name="codigo"/>
    </div>
</div>
<div class="col-lg-3 col-md-6 col-sm-12">
    <div class="form-group">
        <label for="tipo_documento">Tipo documento:</label>
        <select id="tipo_documento" class="form-control" name="tipo_documento">
            <option></option>
            <option value="carnet">Carnet</option>
            <option value="pasaporte">Pasaporte</option>
        </select>
    </div>
</div>

<div class="col-lg-3 col-md-6 col-sm-12">
    <div class="form-group">
        <label for="nro_doc">Número de documento:</label>
        <input id="nro_doc" type="text" class="form-control" name="nro_doc"/>
    </div>
</div>
<div class="col-lg-3 col-md-6 col-sm-12">
    <div class="form-group">
        <label for="nombre">Nombre:</label>
        <input id="nombre" type="text" class="form-control" name="nombre"/>
    </div>
</div>
<div class="col-lg-3 col-md-6 col-sm-12">
    <div class="form-group">
        <label for="ap_paterno">Apellido paterno:</label>
        <input id="ap_paterno" type="text" class="form-control" name="ap_paterno"/>
    </div>
</div>
<div class="col-lg-3 col-md-6 col-sm-12">
    <div class="form-group">
        <label for="fecha-fin">Rango de fecha de reservación:</label>
        <input id="fecha-range" type="text" class="form-control required" name="fecha"
required/>
    </div>
</div>
<div class="col-lg-3 col-md-6 col-sm-12">
    <div class="form-group">
        <label for="cant_per">Cantidad de personas</label>
        <input id="cant_per" type="number" class="form-control" name="cant_per" value="1"
min="1"/>

```

```

    </div>
  </div>

  <div class="col-lg-3 col-md-6 col-sm-12">
    <div class="form-group">
      <label class="" id="tipo" for="tipo">Tipo reservacion:</label>
      <select id="tipo_reservacion" class="form-control " name="tipo_reserva">
        <option value="0">Presencial</option>
        <option value="1">Fecebook</option>
        <option value="2">Telefono</option>
        <option value="3">WhatsApp</option>
      </select>
    </div>
  </div>
  <div class="col-lg-3 col-md-6 col-sm-12">
    <br>
    <button type="button" class="btn btn-primary" id="consultar_btn"
name="consultar_btn">Consultar</button>
  </div>
  <div id="tabla"></div>
</div>
</form>
</div>
</div>
<div class="card ">
  <div class="card-header ">
  </div>
  <div class="card-body">
    <Form>
      <div class="row">
        <div class="col-lg-3 col-md-6 col-sm-12">
          <div class="form-group">
            <label class="" id="cod_tipo" for="tipo_habitacion">Tipo Habitacion:</label>
            <select id="cod_tipo_habitacion" class="form-control " name="cod_tipo_habitacion">
              <option></option>
              @foreach($tipos as $tipo)
                <option value="{{ $tipo->cod_tipo_habitacion }}">{{ $tipo->nom_tipo}}</option>
              @endforeach
            </select>
          </div>
        </div>
        <div class="col-lg-3 col-md-6 col-sm-12">
          <div class="form-group">
            <label class="" id="valor" for="fecha-fin">Valor por Noche:</label>

```

```

        <input id ="valor_noche" type="text" class="form-control " name="valor_noche"
required/>
    </div>
</div>
</div>
</form>
<form>
<div id='calendario'></div>
</form>
<form>
<div id='tabla2'></div>
<div class="col-lg-3 col-md-6 col-sm-12">
<div class="form-group">
<label class="" id="precio" for="fecha-fin">Precio total:</label>
<input id ="precio_total" type="text" class="form-control " name="precio_total"
required/>
</div>
</div>
</form>
{!! Form::open(['route' => 'admin.reservacion.confirmar', 'method' => 'POST']) !!}
{!! Form::hidden('events', null, ['class'=>'form-control', 'id' => 'events']) !!}
{!! Form::hidden('cod_cliente', null, ['class'=>'form-control', 'id' => 'cod_cliente']) !!}
{!! Form::hidden('cant_personas', null, ['class'=>'form-control', 'id' => 'cant_personas']) !!}
{!! Form::hidden('costo_noche', null, ['class'=>'form-control', 'id' => 'costo_noche']) !!}
{!! Form::hidden('to', null, ['class'=>'form-control', 'id' => 'to']) !!}
{!! Form::hidden('from', null, ['class'=>'form-control', 'id' => 'from']) !!}
{!! Form::hidden('diff', null, ['class'=>'form-control', 'id' => 'diff']) !!}
{!! Form::hidden('token', 0, ['class'=>'form-control', 'id' => 'diff']) !!}
{!! Form::hidden('tipo_reserva', null, ['class'=>'form-control', 'id' => 'tipo_reserva']) !!}
{!! Form::submit('confirmar' ,['class' => 'btn btn-primary', 'id' => 'confirmar']) !!}
{!! Form::close() !!}
</div>
</div>
@endsection
@section('page-js-script')
<script src="{{asset('plugin/fullcalendar/js/jquery.min.js')}}" ></script>
<script src="{{asset('plugin/fullcalendar/js/moment.min.js')}}" ></script>
<script src="{{asset('plugin/fullcalendar/js/fullcalendar.min.js')}}" ></script>
<script src="{{asset('plugin/fullcalendar/js/scheduler.min.js')}}" ></script>
<script src="{{asset('plugin/fullcalendar/js/es.js')}}" ></script>
<script src="http://momentjs.com/downloads/moment.min.js"></script>
<script type="text/javascript" src="https://cdn.datatables.net/v/bs4/jzip-2.5.0/dt-1.10.18/b-1.5.2/b-html5-1.5.2/cr-1.5.0/r-2.2.2/sl-1.2.6/datatables.min.js"></script>
<script src="https://cdn.jsdelivr.net/npm/flatpickr"></script>

```

```
<script src="https://npmcdn.com/flatpickr/dist/l10n/es.js"></script>
<script type="text/javascript" src="{{asset('js/fullcalendar.js')}}" ></script>
<script type="text/javascript" src="{{asset('js/flatpicker.js')}}"></script>
@endsection
```

flatpicker.js

```
window.rango = {};
window.init = function () {
    flatpickr.localize(flatpickr.l10ns.es);
    window.rango= flatpickr("#fecha-range",{
        dateFormat: "Y-m-d",
        mode : 'range'
    });
}
$("document").ready(function() {
    window.init();
})
```

fullcalendar.js

```
window.calendar = {};
window.tablaClientes = {};
window.tablaReserva = {}
window.rango = {};
window.event = [];
window.list = {};
window.fecha = function(fechaD){
    var fechaE = new Date(fechaD);
    var fechaF = fechaE.getFullYear()+"-";
    if(fechaE.getMonth() == 0){
        fechaF = fechaF + "1-";
    }else if(fechaE.getMonth() == 1){
        fechaF = fechaF + "2-";
    }else if(fechaE.getMonth() == 2){
        fechaF = fechaF + "3-";
    }else if(fechaE.getMonth() == 3){
        fechaF = fechaF + "4-";
    }else if(fechaE.getMonth() == 4){
        fechaF = fechaF + "5-";
    }else if(fechaE.getMonth() == 5){
        fechaF = fechaF + "6-";
    }else if(fechaE.getMonth() == 6){
        fechaF = fechaF + "7-";
    }else if(fechaE.getMonth() == 7){
        fechaF = fechaF + "8-";
    }
```

```

}else if(fechaE.getMonth() == 8){
    fechaF = fechaF + "9-";
}else if(fechaE.getMonth() == 9){
    fechaF = fechaF + "10-";
}else if(fechaE.getMonth() == 10){
    fechaF = fechaF + "11-";
}else if(fechaE.getMonth() == 11){
    fechaF = fechaF + "12-";
}
fechaF = fechaF + fechaE.getDate();
return fechaF;
}
//Crear tabla consulta clientes
window.crearTabla = function( data){
    $('#tabla').empty()
    $('#tabla').append('<div class = "card"><div class="card-header"></div><div class="card-
body"><table id = "tbl_cli" class="table"></table></div></div>');
    columnas = '<tr><th>Codigo</th><th>Tipo de documento</th><th>Nro°
documento</th><th>Nombre</th><th>Apellido paterno</th><th>Apellido
materno</th><th>Telefono</th><th>Correo
electronico</th><th>Pais</th><th>Ciudad</th></tr>';
    thead = '<thead>'+columnas+'</thead>';
    tfoot = '<tfoot>'+columnas+'</tfoot>';
    $('#tbl_cli').append(thead+tfoot);
    window.tablaClientes = $('#tbl_cli').DataTable( {
        dom: 'Bfrtip',
        data: data,
        buttons: [
            {
                text : 'Seleccionar',
                attr : {id : 'seleccionar',
                    disabled : true
                }
            }
        ]
    },
    columns: [
        { data: 'cod_cliente' },
        { data: 'tipo_documento' },
        { data: 'nro_doc' },
        { data: 'nombre' },
        { data: 'ap_paterno' },
        { data: 'ap_materno' },
        { data: 'telefono' },
        { data: 'correo_electronico' },

```

```

        { data: 'ciudad.pais.nom_pais'},
        { data: 'ciudad.nom_ciudad'},
    ],
    select: {
        style: 'single'
    },
    responsive: true,
    "language": {
        "sProcessing": "Procesando...",
        "sLengthMenu": "Mostrar _MENU_ registros",
        "sZeroRecords": "No se encontraron resultados",
        "sEmptyTable": "Ningún dato disponible en esta tabla",
        "sInfo": "Mostrando registros del _START_ al _END_ de un total de _TOTAL_ registros",
        "sInfoEmpty": "Mostrando registros del 0 al 0 de un total de 0 registros",
        "sInfoFiltered": "(filtrado de un total de _MAX_ registros)",
        "sInfoPostFix": "",
        "sSearch": "Buscar:",
        "sUrl": "",
        "sInfoThousands": ",",
        "sLoadingRecords": "Cargando...",
        "oPaginate": {
            "sFirst": "Primero",
            "sLast": "Último",
            "sNext": "Siguiente",
            "sPrevious": "Anterior"
        },
    },
    "oAria": {
        "sSortAscending": ": Activar para ordenar la columna de manera ascendente",
        "sSortDescending": ": Activar para ordenar la columna de manera descendente"
    }
}
});

```

```

window.tablaClientes.on( 'select', function ( e, dt, type, indexes ) {
    if ( type === 'row' ) {
        var data = window.tablaClientes.rows( indexes ).data();
        $('#seleccionar').attr('disabled',false);
        $('#seleccionar').attr('onclick',function(e, dt, type, indexes){
            $('#nombre').val(data[0].nombre);
            $('#codigo').val(data[0].cod_cliente);
            $('#ap_paterno').val(data[0].ap_paterno);
            $('#nro_doc').val(data[0].nro_doc);
            $('#tipo_documento').val(data[0].tipo_documento);
        });
    }
});

```

```

$('#seleccionar').on('click',function(e){
    $('#cod_tipo').show();
    $('#cod_tipo_habitacion').show();
    $('#valor').show();
    $('#valor_noche').show();
    $('#tabla').empty()

    $('#nombre').prop('disabled', true);
    $('#codigo').prop('disabled', true);
    $('#ap_paterno').prop('disabled', true);
    $('#nro_doc').prop('disabled', true);
    $('#tipo_documento').prop('disabled', true);
});
}
});
window.tablaClientes.on( 'deselect', function ( e, dt, type, indexes ) {
    if ( type === 'row' ) {
        $('#seleccionar').attr('disabled',true);
    }
});
}
//Crea tabla detalle de habitaciones
window.crearTabla2 = function( data){
    $('#tabla2').empty()
    $('#tabla2').append('<div class = "card"><div class="card-header"></div><div class="card-
body"><table id ="tbl_res"class="table"></table></div></div>');
    columnas = '<tr><th>Tipo Habitacion</th><th>Habitacion</th><th>Precio</th></tr>';
    thead = '<thead>'+columnas+'</thead>';
    tfoot = '<tfoot>'+columnas+'</tfoot>';
    $('#tbl_res').append(thead+tfoot);
    window.tablaReserva = $('#tbl_res').DataTable( {
        dom: 'Bfrtip',
        data: data,
        buttons: [
            {
                text : 'Eliminar',
                attr : {id : 'eliminar',
                    disabled : true
                }
            }
        ],
        columns: [
            { data: 'nom_tipo' },
            { data: 'nom_hab' },

```

```

    { data: 'valor_noche' },
  ],
  select: {
    style: 'single'
  },
  responsive: true,
  "language": {
    "sProcessing": "Procesando...",
    "sLengthMenu": "Mostrar _MENU_ registros",
    "sZeroRecords": "No se encontraron resultados",
    "sEmptyTable": "Ningún dato disponible en esta tabla",
    "sInfo": "Mostrando registros del _START_ al _END_ de un total de _TOTAL_ registros",
    "sInfoEmpty": "Mostrando registros del 0 al 0 de un total de 0 registros",
    "sInfoFiltered": "(filtrado de un total de _MAX_ registros)",
    "sInfoPostFix": "",
    "sSearch": "Buscar:",
    "sUrl": "",
    "sInfoThousands": ",",
    "sLoadingRecords": "Cargando...",
    "oPagate": {
      "sFirst": "Primero",
      "sLast": "Último",
      "sNext": "Siguiente",
      "sPrevious": "Anterior"
    },
    "oAria": {
      "sSortAscending": ": Activar para ordenar la columna de manera ascendente",
      "sSortDescending": ": Activar para ordenar la columna de manera descendente"
    }
  }
});

window.tablaReserva.on( 'select', function ( e, dt, type, indexes ) {
  if ( type === 'row' ) {
    var data = window.tablaReserva.rows( indexes ).data();
    $('#eliminar').attr('disabled',false);
    $("#eliminar").on('click', function(){
      for(var i=0; window.event.length >= i ;i++){
        if(window.event[i] == data[0]){
          $("#calendario").fullCalendar( 'removeEvents', window.event[i] );
          window.event.splice(i, 1)
          if(window.event.length == 0){
            $("#fecha-range").prop('disabled', false);
            $("#confirmar").hide();
          }
        }
      }
    });
  }
});

```

```

    }
  }
  var cod_tipo_habitacion = $("#cod_tipo_habitacion").val()
  $.getJSON(      "http://127.0.0.1:8000/api/reservaEvent?",      {"cod_tipo_habitacion":
cod_tipo_habitacion })
  .done(function( data, textStatus, jqXHR ) {
    if (data) {
      window.list = data;
      $('#valor_noche').val(data.valor);
      $('#calendario').fullCalendar('destroy');
      var total=0;
      $.each(window.event,function(pos, elemento){
        total = total + parseInt(elemento.valor_noche);
      })
      $('#precio_total').val(total);
      window.crearCalendar(data);
    }
  })
  window.tablaReserva.rows( { selected: true } ).remove().draw();
});
});
window.tablaReserva.on( 'deselect', function ( e, dt, type, indexes ) {
  if ( type === 'row' ) {
    $('#eliminar').attr('disabled',true);
  }
});
}
//Crear Calendario
window.crearCalendar = function( data){
  var evento = window.event
  ,rango = window.rango.selectedDates
  , from = ""
  if (Array.isArray(rango) && rango.length) {
    from = flatpickr.formatDate(rango[0], "Y-m-d");
  }

  $("#calendario").fullCalendar({
    header: {
      left: 'prev,next today',
      center: 'title',
    },
    displayEventTime: false,
    defaultDate: moment(from),

```

```

lazyFetching : true,
//funcion para desplegar los eventos al cambiar la vista del mes
viewRender: function(view){
    $('#calendario').fullCalendar( 'removeEvents');
    $('#calendario').fullCalendar( 'addEventSource', window.list.events );
    $('#calendario').fullCalendar( 'addEventSource', window.event );
    $('#calendario').fullCalendar( 'refetchEvents' );
},
//Funcion para agregar evento al calendario
dayClick: function(date, jsEvent, view, resourceObj ) {
    var rango = window.rango.selectedDates
    , to = ""
    , from = ""
    if (Array.isArray(rango) && rango.length) {
        from = flatpickr.formatDate(rango[0], "Y-m-d");
        if (rango.length > 1) {
            to = flatpickr.formatDate(rango[1], "Y-m-d");
        }
    }
    var NuevoEvento= {
        resourceId: resourceObj.id,
        id: 'A' + resourceObj.id ,
        title: 'Nueva Reservacion',
        start: from,
        end: to + 'T23:59:59',
        nom_tipo: resourceObj.nombre_tipo,
        nom_hab: resourceObj.nom_hab,
        color: '20eb09',
        overlap: false,
        editable: true,
        durationEditable: false,
        valor_noche: $('#valor_noche').val(),
        cod_tipo_habitacion : $('#cod_tipo_habitacion').val()
    }
    var cont=0;
    resp = false;
    var fechaF = window.fecha(NuevoEvento.end);
    var fechaS;
    window.list.events.forEach(function(value){
        if(((NuevoEvento.start <= value.start && NuevoEvento.end >= value.end)
        || (NuevoEvento.start >= value.start && NuevoEvento.end <= value.end)
        || ((NuevoEvento.start >= value.start && NuevoEvento.end >= value.end)
        &&(NuevoEvento.start <= value.end && NuevoEvento.end >= value.star))

```

```

    || ((NuevoEvento.start <= value.start && NuevoEvento.end <= value.end) && (NuevoEvento.end
>= value.start)))
    && NuevoEvento.resourceId == value.resourceId && cont === 0){
        fechaS = window.fecha(value.end);
        if((fechaF == value.start && NuevoEvento.start <= value.start) || (NuevoEvento.start == fechaS
&& NuevoEvento.end >= value.end)){
            resp = false;
        }else{
            cont++;
            alert('Error, la habitación ya se encuentra reservada para esas fechas');
            resp = true;
        }
    }
})
cont=0;
window.event.forEach(function(value){
    if(((NuevoEvento.start <= value.start && NuevoEvento.end >= value.end &&
NuevoEvento.resourceId == value.resourceId)) && cont === 0){
        cont++;
        resp = true;
        alert('Error, la habitación ya se encuentra reservada para esas fechas');
    }
})
if(!resp){
    window.event.push(NuevoEvento);
    $('#calendario').fullCalendar('renderEvent', NuevoEvento);
    $('#fecha-range').prop('disabled', true);
    window.crearTabla2(window.event);
    $('#precio_total').show();
    $('#precio').show();
    $('#confirmar').show();
    var total=0;
    $.each(window.event,function(pos, elemento){
        total = total + parseInt(elemento.valor_noche);
    })
    $('#precio_total').val(total);
}
},
eventDrop: function ( evento , delta , revertFunc , jsEvent , ui , vista ) {
    if (delta >= 1 || delta <= -1){
        revertFunc();
    }else{
        var cont=0;
        resp = false;

```

```

var fechaF = window.fecha(evento.end._i);
var fechaS;
window.list.events.forEach(function(value){
  if(((evento.start._i <= value.start && evento.end._i >= value.end)
  || (evento.start._i >= value.start && evento.end._i <= value.end)
  || ((evento.start._i >= value.start && evento.end._i >= value.end) && (evento.start._i <=
value.end && evento.end._i >= value.start))
  || ((evento.start._i <= value.start && evento.end._i <= value.end) && (evento.end._i >=
value.start)))
    && evento.resourceId == value.resourceId && cont === 0) && evento.id != value.id){
    fechaS = window.fecha(value.end);
    if((fechaF == value.start && evento.start._i <= value.start) || (evento.end._i == fechaS &&
evento.end._i >= value.end)){
      resp = false;
    }else{
      cont++;
      resp = true;
    }
  }
})
cont=0;
window.event.forEach(function(value){
  if(((evento.start._i <= value.start && evento.end._i >= value.end && evento.resourceId ==
value.resourceId)) && cont === 0){
    cont++;
    resp = true;
  }
})
if(resp){
  revertFunc();
}else{
  // Esto es lo que ya tengo antes de realizar cambios
  window.event.forEach(function(value){
    if(value.id == evento.id){
      value.resourceId = evento.resourceId;
    }
  });
  var cod_habitacion = evento.resourceId
    , cod_reservacion = evento.id ;
  $.getJSON( "http://127.0.0.1:8000/api/reservasave?", {"cod_habitacion": cod_habitacion,
    "cod_reservacion": cod_reservacion})
}
}
},

```

```

schedulerLicenseKey: 'GPL-My-Project-Is-Open-Source',
defaultView: 'timelineMonth',
resourceLabelText: 'Habitaciones',
resources: window.list.resource,
eventSources: [{
  events: window.list.events,
},
{
  events: window.event,
}],
resourceAreaWidth: '10%',
height: 'auto',
eventOverlap: false,
// allow "more" link when too many events
});
}
//Consultar datos clientes para iniciar tabla
$("document").ready(function() {
  $('#cod_tipo').hide();
  $('#precio_total').hide();
  $('#cod_tipo_habitacion').hide();
  $('#valor').hide();
  $('#valor_noche').hide();
  $('#consultar2_btn').hide();
  $('#precio').hide();
  $('#confirmar').hide();
  $("body").on("click", "#consultar_btn", function() {
    var rango = window.rango.selectedDates
    , from = ""
    , hoy = new Date().toJSON().slice(0,10);
    if (Array.isArray(rango) && rango.length) {
      from = flatpickr.formatDate(rango[0], "Y-m-d");
    }
    if($('#fecha-range').val() == ""){
      alert("Debe ingresar una fecha");
    }else{
      if(from < hoy){
        alert("La fecha ingresada no puede ser menor a la actual");
        from = "";
      }else{
        var nombre = $("#nombre").val()
        , codigo = $("#codigo").val()
        , ap_paterno = $("#ap_paterno").val()
        , nro_doc = $("#nro_doc").val()

```

```

        , tipo_documento = $("#tipo_documento").val()
    ;
    $.getJSON( "http://127.0.0.1:8000/api/clientes?", {"cod_cliente": codigo,
        "tipo_documento": tipo_documento,
        "nombre" : nombre,
        "ap_paterno" : ap_paterno,
        "nro_doc": nro_doc})
    .done(function( data, textStatus, jqXHR ) {
        if (data) {
            window.crearTabla(data);
        }
    })
    .fail(function(jqXHR, textStatus, errorThrown) {
        console.log( "error" );
    })
    }
}
})

//Consultar tipo habitacion para crear fullcalendar
$("#body").on('change','#cod_tipo_habitacion',function () {
    var cod_tipo_habitacion = $("#cod_tipo_habitacion").val()
    $.getJSON( "http://127.0.0.1:8000/api/reservaEvent?", {"cod_tipo_habitacion":
cod_tipo_habitacion})
    .done(function( data, textStatus, jqXHR ) {
        if (data) {
            window.list = data;
            $('#valor_noche').val(data.valor);
            $('#calendario').fullCalendar('destroy');
            window.crearCalendar(data);
        }
    })
    .fail(function(jqXHR, textStatus, errorThrown) {
        console.log( "error" );
    })
    })

//Confirmar Habitacion
$("#body").on("click", "#confirmar", function() {
    var evento = window.event
    ,rango = window.rango.selectedDates
    , to = ""
    , from = ""
    if (Array.isArray(rango) && rango.length) {

```

```

    from = flatpickr.formatDate(rango[0], "Y-m-d");
    if (rango.length > 1) {
        to = flatpickr.formatDate(rango[1], "Y-m-d");
    }
}
var fecha1 = moment(from);
var fecha2 = moment(to);
var diff = fecha2.diff(fecha1, 'days')

$('#events').val(JSON.stringify(window.event));
$('#cod_cliente').val($("#codigo").val());
$('#costo_noche').val($("#precio_total").val());
$('#cant_personas').val($("#cant_per").val());
$('#tipo_reserva').val($("#tipo_reservacion").val());
$('#to').val(to);
$('#from').val(from);
$('#diff').val(diff);
})
});

```

ReservacionController.php

```

<?php
namespace App\Http\Controllers;
use Illuminate\Http\Request;
use App\Habitacion;
use App\Bloqueo;
use App\Reservacion;
use App\Dia_reservada;
use App\Tipo_habitacion;
use App\Cliente;
use Carbon\Carbon;
use App\Ciudad;
use App\Pais;
use Illuminate\Support\Facades\DB;
class ReservacionController extends Controller
{
    public function create()
    {
        $tipos = Tipo_habitacion::orderBy('cod_tipo_habitacion', 'ASC')->get();
        return view('admin.Reservacion.RegistrarReservacion')->with('tipos', $tipos);
    }
    public function store(Request $request)
    {
        if($request->token == 0){

```

```

$eventos = Json_decode($request['habs'], true);
$reserva = new Reservacion($request->all());
$reserva->fecha_reservacion = date('Y-m-d');
$reserva->cod_usuario = auth()->user()->cod_usuario;
$reserva->tipo_reserva = $request->tipo_reserva;
$reserva->save();
foreach($eventos as $value){
    $this->reservada($value, $reserva->cod_reservacion, $reserva->fecha_inicio, $reserva-
>fecha_fin);
}
flash("Se ha registrado la reserva de forma exitosa")->success();
return redirect()->route('reservacion.create');
}else{
    $eventos = Json_decode($request['habs'], true);
    $reserv = Reservacion::find($request->cod_reser);
    $reserv->estado_reserva = 4;
    $reserv->fecha_check_out = $request->fecha_inicio;
    $reserv->save();
    $reserva = new Reservacion($request->all());
    $reserva->fecha_reservacion = date('Y-m-d');
    $reserva->fecha_check_in = $request->fecha_inicio;
    $reserva->estado_reserva = 2;
    $reserva->cod_usuario = auth()->user()->cod_usuario;
    $reserva->tipo_reserva = 0;
    $reserva->save();
    foreach($eventos as $value){
        $this->reservada($value, $reserva->cod_reservacion, $reserva->fecha_inicio, $reserva-
>fecha_fin);
    }

    flash("Se ha extendido la reserva de forma exitosa")->success();
    return redirect()->route('reservacion.index');
}
}

public function reservada($data, $id, $fecha, $fecha_fin){
    while($fecha <= $fecha_fin){
        $fecha = new Carbon($fecha);
        $this->dia($id, $data['cod_habitacion'], $data['valor_noche'], $fecha);
        $fecha = $fecha->addDay();
        $fecha = $fecha->toDateString();
    }
}

public function dia($id_reserva, $id_habitacion, $valor_noche, $fecha){

```

```

$dias_reservada = new Dia_reservada();
$dias_reservada->cod_reservacion = $id_reserva;
$dias_reservada->cod_habitacion = $id_habitacion;
$dias_reservada->fecha_reservada = $fecha;
$dias_reservada->valor_hab_noche = $valor_noche;
$dias_reservada->save();
}

public function confirmar(Request $request){
    $eventos = Json_decode($request['events']);
    $cliente = Cliente::find($request['cod_cliente']);
    $ciudad = Ciudad::find($cliente->ciudad_res);
    $pais = Pais::find($ciudad->pais);
    $habitacion = new Habitacion;
    $valor_noche = $request['costo_noche'];
    $cantidad_personas = $request['cant_personas'];
    $fecha_inicio = $request['from'];
    $fecha_fin = $request['to'];
    $tipo_reserva = $request['tipo_reserva'];
    if($request['diff'] == 0){
        $diff = 1;
    }else{
        $diff = $request['diff'];
    }
    $i=0;
    $precio= array();
    foreach($eventos as $value){
        $habitacion = $habitacion->orWhere('cod_habitacion', $value->resourceId);
        $precio[$i]['resourceId'] = $value->resourceId;
        $precio[$i]['valor_noche'] = $value->valor_noche;
        $i = $i + 1;
    }
    $habitacion = $habitacion->join('Tipo_habitacion', 'Tipo_habitacion.cod_tipo_habitacion',
'Habitacion.cod_tipo_habitacion')
->get();
    foreach($habitacion as $value){
        foreach($precio as $value2){
            if($value2['resourceId'] == $value->cod_habitacion){
                $value->valor_noche = $value2['valor_noche'];
            }
        }
    }
    $reservacion= Reservacion::all()->pluck('cod_reservacion')->last();
    $reserv = Reservacion::all()->pluck('cod_reservacion')->last();
    $reservacion = $reservacion +1;
}

```

```

    return view('admin.Reservacion.ConfirmarReservacion')->with('cliente', $cliente)->with('ciudad'
,$ciudad)
        ->with('pais', $pais)->with('habitacion', $habitacion)->with('reservacion', $reservacion)
        ->with('valor_noche', $valor_noche)->with('cantidad_personas', $cantidad_personas)
        ->with('fecha_inicio', $fecha_inicio)->with('fecha_fin', $fecha_fin)->with('diff', $diff)
        ->with('precio', $precio)->with('token', $request->token)->with('reserv', $reserv)
        ->with('tipo_reserva', $tipo_reserva);
    }
    public function reservasave(Request $request){
        $cod = explode('$', $request->cod_reservacion);
        $reserva = Dia_reservada::where('cod_reservacion', $cod[0])->get();
        foreach ($reserva as $value) {
            $value->cod_habitacion = $request->cod_habitacion;
            $value->save();
        }
        return response()->json(true);
    }
    public function reservaEvent(Request $request){
        $bloqueo = new habitacion;
        $tipo_hab = new habitacion;
        $reserva = new habitacion;
        $reserva2 = new habitacion;
        $param = $request->all();
        $wResp = array();
        $resource = array();
        $events =array();
        $t = Tipo_habitacion::find($request->cod_tipo_habitacion);
        $bloqueo = $bloqueo->where('cod_tipo_habitacion', $param['cod_tipo_habitacion']);
        $bloqueo = $bloqueo->join('Bloqueo', 'Bloqueo.cod_habitacion', 'Habitacion.cod_habitacion')
            ->where('Bloqueo.fecha_fin', '>=', Date('Y-m-d'));
        $bloqueo = $bloqueo->get();
        $tipo_hab
            =
            $tipo_hab->where('Habitacion.cod_tipo_habitacion',
$param['cod_tipo_habitacion'])
            ->join('Tipo_habitacion',
                'Tipo_habitacion.cod_tipo_habitacion',
'Habitacion.cod_tipo_habitacion')->get();
        $reserva = $reserva->where('cod_tipo_habitacion', $param['cod_tipo_habitacion']);
        $reserva
            =
            $reserva->join('Dia_reservada',
                'Dia_reservada.cod_habitacion',
'Habitacion.cod_habitacion')
            ->join('Reservacion','Dia_reservada.cod_reservacion',
                'Reservacion.cod_reservacion')->where('Reservacion.fecha_fin', '>=', Date('Y-m-d'))
            ->where('Reservacion.estado_reserva', '<>', 5);

        if(isset($param["cod_reservacion"])){
            $reserva = $reserva->where('Reservacion.cod_reservacion', '<>', $param["cod_reservacion"]);
        }
    }

```

```

    }
    $reserva = $reserva->get();
    foreach ($tipo_hab as $value) {
        array_push($resource, array('id'=>$value['cod_habitacion'],
            'title'=>$value['nom_hab'],
            'nombre_tipo'=>$value['nom_tipo'],
            'nom_hab'=>$value['nom_hab']));
    }
    foreach ($bloqueo as $value) {
        array_push($events, array('resourceId'=>$value['cod_habitacion'],
            'id'=> 'b'. $value['cod_bloqueo'],
            'title'=>'Bloqueo '. $value['cod_bloqueo'],
            'start'=>Carbon::parse($value['fecha_inicio'])->format("Y-m-d"),
            'end'=>Carbon::parse($value['fecha_fin'])->format("Y-m-d").'T23:59:59',
            'color'=>'fbff0f',
            'editable'=>false,
            'overlap'=> true));
    }
    $p = "";
    $p1 = "";
    foreach ($reserva as $value) {
        if($value['cod_habitacion']!= $p || $value['cod_reservacion'] != $p1){
            array_push($events, array('resourceId'=>$value['cod_habitacion'],
                'id'=> $value['cod_reservacion']. " $" . $value['cod_habitacion'],
                'title'=>'Reservacion '. $value['cod_reservacion'],
                'start'=>Carbon::parse($value['fecha_inicio'])->format("Y-m-d"),
                'end'=>Carbon::parse($value['fecha_fin'])->format("Y-m-d").'T23:59:59',
                'color'=>'e00016',
                'editable'=>true,
                'overlap'=> true,
                'durationEditable'=> false,));
            $p=$value['cod_habitacion'];
            $p1=$value['cod_reservacion'];
        }
    }
    $wResp = array('resource'=>$resource,
        'events' =>$events,
        'valor' => $t->precio_noche);
    return response()->json($wResp);
}
}

```

ConfirmarReservacion.blade.php

@extends('admin.template.main')

```

@section('title', 'Crear Habitacion')
@section('page-css-style')
@section('content')
<div class="card bg-light text-dark">
  <div class="card-header ">
    <h4>Confirmar reservación</h4>
  </div>
  {!! Form::open(['route' => 'reservacion.store', 'method' => 'POST']) !!}
  <div class="card-body ">
    <div class="row">
      <table class="table table-bordered">
        <tbody>
          <tr>
            <td>Nombre cliente completo</td>
            <td>{{$cliente->nombre}} {{$cliente->ap_paterno}} {{$cliente->ap_materno}}</td>
            {!! Form::hidden('cod_cliente', $cliente->cod_cliente, ['class'=>'form-control']) !!}
            {!! Form::hidden('tipo_reserva', $tipo_reserva, ['class'=>'form-control']) !!}
          </tr>
          <tr>
            <td>Tipo documento</td>
            <td>{{$cliente->tipo_documento}}</td>
          </tr>
          <tr>
            <td>Nroª documento</td>
            <td>{{$cliente->nro_doc}}</td>
            {!! Form::hidden('nro_doc_cli', $cliente->nro_doc, ['class'=>'form-control']) !!}
            {!! Form::hidden('token', $token, ['class'=>'form-control']) !!}
          </tr>
          <tr>
            <td>País residencia</td>
            <td>{{$pais->nom_pais}}</td>
          </tr>
          <tr>
            <td>Ciudad residencia</td>
            <td>{{$ciudad->nom_ciudad}}</td>
          </tr>
          <tr>
            <td>Número de contacto</td>
            <td>{{$cliente->telefono}}</td>
          </tr>
          <tr>
            <td>Código de reserva</td>
            <td>{{$reservacion}}</td>
            {!! Form::hidden('cod_reser', $reserv, ['class'=>'form-control']) !!}
          </tr>
        </tbody>
      </table>
    </div>
  </div>

```

```

</tr>
<tr>
  <td>Fecha inicio reserva</td>
  <td>{{ $fecha_inicio }}</td>
  {!! Form::hidden('fecha_inicio', $fecha_inicio, ['class'=>'form-control']) !!}
</tr>
<tr>
  <td>Fecha fin reserva</td>
  <td>{{ $fecha_fin }}</td>
  {!! Form::hidden('fecha_fin', $fecha_fin, ['class'=>'form-control']) !!}
</tr>
<tr>
  <td>Cantidad de personas</td>
  <td>{{ $cantidad_personas }}</td>
  {!! Form::hidden('numero_personas', $cantidad_personas, ['class'=>'form-control']) !!}
  {!! Form::hidden('habs', $habitacion, ['class'=>'form-control']) !!}
</tr>
</tbody>
</table>
<table class="table table-striped table-white">
  <thead>
    <th>#</th>
    <th>Habitación</th>
    <th>Tipo habitación</th>
    <th>Costo por noche</th>
  </thead>
  <tbody>
    @foreach($habitacion as $h)
      <tr>
        <td>{{ $h->cod_habitacion }}</td>
        <td>{{ $h->nom_hab }}</td>
        <td>{{ $h->nom_tipo }}</td>
        <td>{{ $h->valor_noche }}</td>
      </tr>
    @endforeach
  </tbody>
</table>
<div class="col-lg-3 col-md-6 col-sm-12">
  <div class="label">Total a pagar</div>
  <div class="label">Abono mínimo</div>
</div>
<div class="col-lg-3 col-md-6 col-sm-12">
  {!! Form::hidden('valor_noche_venta', $valor_noche, ['class'=>'form-control']) !!}
  <div class="label">{{ $valor_noche * $diff }}</div>

```

```
<div class="label">{{($valor_noche * $diff)/2}}</div>
</div>
<div class="col-lg-6 col-md-6 col-sm-12">
  {!! Form::submit('Confirmar', ['class' => 'btn btn-primary']) !!}
  <a class="button btn btn-danger" href="{{ route('reservacion.create') }}">Cancelar</a>
</div>
</div>
</div>
{!! Form::close() !!}
</div>
@endsection
@section('page-js-script')
@endsection
```

Asignar habitaciónAsignarHabitacion.blade.php

```

@extends('admin.template.main')
@section('title', 'Asignar habitación')
@section('page-css-style')
    <link rel="stylesheet" href="{{asset('plugin/fullcalendar/css/fullcalendar.min.css')}}"/>
    <link rel="stylesheet" href="{{asset('plugin/fullcalendar/css/scheduler.min.css')}}"/>
@endsection
<div class="card ">
    <div class="card-header ">
        Asignar Habitación
    </div>
    <div class="card-body">
        <Form>
            <div class="row">
                <div class="col-lg-3 col-md-6 col-sm-12">
                    <div class="form-group">
                        <label class="" id="cod_tipo" for="tipo_habitacion">Tipo Habitacion:</label>
                        <select id ="cod_tipo_habitacion" class="form-control " name="cod_tipo_habitacion">
                            <option></option>

                            @foreach($tipos as $tipo)
                                <option value="{{ $tipo->cod_tipo_habitacion }}">{{ $tipo->nom_tipo}}</option>
                            @endforeach
                        </select>
                    </div>
                </div>
            </div>
        </form>
        <form>
            <div id='calendario'></div>
        </form>
    </div>
</div>
@endsection
@section('page-js-script')
    <script src="{{asset('plugin/fullcalendar/js/jquery.min.js')}}"></script>
    <script src="{{asset('plugin/fullcalendar/js/moment.min.js')}}"></script>
    <script src="{{asset('plugin/fullcalendar/js/fullcalendar.min.js')}}"></script>
    <script src="{{asset('plugin/fullcalendar/js/scheduler.min.js')}}"></script>
    <script src="{{asset('plugin/fullcalendar/js/es.js')}}"></script>
    <script type="text/javascript" src="{{asset('js/asignar.js')}}"></script>
@endsection

```

asignar.js

```

window.calendar = {};
window.list = {};
window.fecha = function(fechaD){
  var fechaE = new Date(fechaD);
  var fechaF = fechaE.getFullYear()+"-";
  if(fechaE.getMonth() == 0){
    fechaF = fechaF + "1-";
  }else if(fechaE.getMonth() == 1){
    fechaF = fechaF + "2-";
  }else if(fechaE.getMonth() == 2){
    fechaF = fechaF + "3-";
  }else if(fechaE.getMonth() == 3){
    fechaF = fechaF + "4-";
  }else if(fechaE.getMonth() == 4){
    fechaF = fechaF + "5-";
  }else if(fechaE.getMonth() == 5){
    fechaF = fechaF + "6-";
  }else if(fechaE.getMonth() == 6){
    fechaF = fechaF + "7-";
  }else if(fechaE.getMonth() == 7){
    fechaF = fechaF + "8-";
  }else if(fechaE.getMonth() == 8){
    fechaF = fechaF + "9-";
  }else if(fechaE.getMonth() == 9){
    fechaF = fechaF + "10-";
  }else if(fechaE.getMonth() == 10){
    fechaF = fechaF + "11-";
  }else if(fechaE.getMonth() == 11){
    fechaF = fechaF + "12-";
  }
  fechaF = fechaF + fechaE.getDate();
  return fechaF;
}
//Crear Calendario
window.crearCalendar = function( data){
  var from = new Date().toJSON().slice(0,10);
  $("#calendario").fullCalendar({
    header: {
      left: 'prev,next today',
      center: 'title',
    },
    displayEventTime: false,
  });
}

```

```

defaultDate: moment(from),
lazyFetching : true,
//funcion para desplegar los eventos al cambiar la vista del mes
viewRender: function(view){
  $('#calendario').fullCalendar( 'removeEvents');
  $('#calendario').fullCalendar( 'addEventSource', window.list.events );
  $('#calendario').fullCalendar( 'refetchEvents' );
},
eventDrop: function ( evento , delta , revertFunc , jsEvent , ui , vista ) {
  if (delta >= 1 || delta <= -1){
    revertFunc();
  }else{
    var cont=0;
    resp = false;
    var fechaF = window.fecha(evento.end._i);
    var fechaS;
    window.list.events.forEach(function(value){
      if(((evento.start._i <= value.start && evento.end._i >= value.end)
      || (evento.start._i >= value.start && evento.end._i <= value.end)
      || ((evento.start._i >= value.start && evento.end._i >= value.end) &&(evento.start._i <=
value.end && evento.end._i >= value.start))
      || ((evento.start._i <= value.start && evento.end._i <= value.end) && (evento.end._i >=
value.start)))
        && evento.resourceId == value.resourceId && cont === 0) && evento.id != value.id){
          fechaS = window.fecha(value.end);
          if((fechaF == value.start && evento.start._i <= value.start) || (evento.end._i == fechaS &&
evento.end._i >= value.end)){
            resp = false;
          }else{
            cont++;
            resp = true;
          }
        }
      })
    if(resp){
      revertFunc();
    }else{
      var cod_habitacion = evento.resourceId
      , cod_reservacion = evento.id
      ;
      $.getJSON( "http://127.0.0.1:8000/api/reservasave?", {"cod_habitacion": cod_habitacion,
"cod_reservacion": cod_reservacion})
    }
  }
}

```

```

},
schedulerLicenseKey: 'GPL-My-Project-Is-Open-Source',
defaultView: 'timelineMonth',
resourceLabelText: 'Habitaciones',
resources: window.list.resource,
eventSources: [{
  events: window.list.events,
}],
resourceAreaWidth: '10%',
height: 'auto',
eventOverlap: false,
// allow "more" link when too many events
});
}
//Consultar datos clientes para iniciar tabla
$("document").ready(function() {
  //Consultar tipo habitacion para crear fullcalendar
  $("#body").on('change', '#cod_tipo_habitacion', function () {
    var cod_tipo_habitacion = $("#cod_tipo_habitacion").val()
    $.getJSON(
      "http://127.0.0.1:8000/api/reservaEvent?",
      {"cod_tipo_habitacion":
cod_tipo_habitacion})
    .done(function( data, textStatus, jqXHR ) {
      if (data) {
        window.list = data;
        $('#calendario').fullCalendar('destroy');
        window.crearCalendar(data);
      }
    })
    .fail(function(jqXHR, textStatus, errorThrown) {
      console.log( "error" );
    })
  })
});

```


Lista de clientes hospedadosindex.blade.php

```

@extends('admin.template.main')
@section('title', 'Home')
@section('page-css-style')
    <link rel="stylesheet"
        href="https://cdn.jsdelivr.net/npm/flatpickr/dist/flatpickr.min.css">
@endsection
@section('content')
    <div class="card bg-dark text-light">
        <div class="card-header ">
            <h4>Lista de clientes Hospedados</h4>
        </div>
        <div class="card-body">
            {!! Form::open(['route' => 'listas.clienteshospedados_show', 'method' => 'GET']) !!}
                <div class="row">
                    <div class="col-lg-3 col-md-6 col-sm-12">
                        <div class="form-group">
                            <label for="id_pago">fecha clientes hospedado:</label>
                            <input id="fecha-range" type="text" class="form-control" name="fecha" />
                        </div>
                    </div>
                    <div class="input-group mb-3">
                        <div class="col-lg-4 col-md-6 col-sm-12">
                            <div class="form-group">
                                {!!Form::checkbox('cod_cliente', null ,true)!!}
                                <label for="checkbox1">Código cliente</label>
                            </div>
                        </div>
                        <div class="col-lg-4 col-md-6 col-sm-12">
                            <div class="form-group">
                                {!!Form::checkbox('nombre', null ,true)!!}
                                <label for="check">Nombre</label>
                            </div>
                        </div>
                        <div class="col-lg-4 col-md-6 col-sm-12">
                            <div class="form-group">
                                {!!Form::checkbox('ap_paterno', null ,true)!!}
                                <label for="check">Apellido paterno</label>
                            </div>
                        </div>
                    </div>
                    <div class="input-group mb-3">
                        <div class="col-lg-4 col-md-6 col-sm-12">

```

```

<div class="form-group">
  {!!Form::checkbox('ap_materno', null ,true)!!}
  <label for="check">Apellido materno</label>
</div>
</div>
<div class="col-lg-4 col-md-6 col-sm-12">
  <div class="form-group">
    {!!Form::checkbox('telefono', null ,true)!!}
    <label for="check">Número de teléfono</label>
  </div>
</div>
<div class="col-lg-4 col-md-6 col-sm-12">
  <div class="form-group">
    {!!Form::checkbox('pais', null ,true)!!}
    <label for="check">País</label>
  </div>
</div>
</div>
<div class="input-group mb-3">
  <div class="col-lg-4 col-md-6 col-sm-12">
    <div class="form-group">
      {!!Form::checkbox('ciudad', null ,true)!!}
      <label for="check">Ciudad</label>
    </div>
  </div>
  <div class="col-lg-4 col-md-6 col-sm-12">
    <div class="form-group">
      {!!Form::checkbox('direccion', null ,true)!!}
      <label for="check">Dirección</label>
    </div>
  </div>
  <div class="col-lg-4 col-md-6 col-sm-12">
    <div class="form-group">
      {!!Form::checkbox('correo_electronico', null ,true)!!}
      <label for="check">Correo electrónico</label>
    </div>
  </div>
</div>
<div class="input-group mb-3">
  <div class="col-lg-4 col-md-6 col-sm-12">
    <div class="form-group">
      {!!Form::checkbox('check_in', null ,true)!!}
      <label for="check_in">Fecha inicio de reserva</label>
    </div>
  </div>
</div>

```

```

    </div>
    <div class="col-lg-4 col-md-6 col-sm-12">
        <div class="form-group">
            {!!Form::checkbox('fecha_fin', null ,true)!!}
            <label for="fecha_fin">Fecha de fin de reserva</label>
        </div>
    </div>
</div>
<div class="col-lg-6 col-md-6 col-sm-12">
    {!! Form::submit('Generar Informes', ['class' => 'btn btn-primary']) !!}
    <a class="button btn btn-danger" href="/gestorInformes">Cancelar</a>
</div>
{!!Form::close()!!}
</div>
</div>
@endsection
@section('page-js-script')
    <script src="https://cdn.jsdelivr.net/npm/flatpickr"></script>
    <script src="https://npmcdn.com/flatpickr/dist/l10n/es.js"></script>
    <script type="text/javascript" src="{{asset('js/flatpicker.js')}}"></script>
@endsection

```

show.blade.php

```

@extends('admin.template.main')
@section('title', 'Lista de Clientes Hospedados')
@section('page-css-style')
    <link rel="stylesheet" type="text/css" href="https://cdn.datatables.net/v/bs4/jszip-2.5.0/dt-
1.10.18/b-1.5.2/b-html5-1.5.2/cr-1.5.0/r-2.2.2/sl-1.2.6/datatables.min.css"/>
@endsection
@section('content')
    <table id="tabla" class="table table-striped">
        <thead>
            @if(isset($request->cod_cliente))
                <th>Codigo cliente</th>
            @endif
            @if(isset($request->nombre))
                <th>Nombre</th>
            @endif
            @if(isset($request->ap_paterno))
                <th>Apellido paterno</th>
            @endif
            @if(isset($request->ap_materno))
                <th>Apellido materno</th>
            @endif
        </thead>
    </table>

```

```

@endif
@if(isset($request->telefono))
    <th>Teléfono</th>
@endif
@if(isset($request->pais))
    <th>País</th>
@endif
@if(isset($request->ciudad))
    <th>Ciudad</th>
@endif
@if(isset($request->direccion))
    <th>Dirección</th>
@endif
@if(isset($request->correo_electronico))
    <th>Correo electronico</th>
@endif
@if(isset($request->check_in))
    <th>Fecha inicio reserva</th>
@endif
@if(isset($request->fecha_fin))
    <th>Fecha fin reserva</th>
@endif
</thead>
<tbody>
    @foreach($resp as $value)
        <tr>
            @if(isset($value['cod_cliente']))
                <td>{{ $value['cod_cliente'] }}</td>
            @endif
            @if(isset($value['nombre']))
                <td>{{ $value['nombre'] }}</td>
            @endif
            @if(isset($value['ap_paterno']))
                <td>{{ $value['ap_paterno'] }}</td>
            @endif
            @if(isset($value['ap_materno']))
                <td>{{ $value['ap_materno'] }}</td>
            @endif
            @if(isset($value['telefono']))
                <td>{{ $value['telefono'] }}</td>
            @endif
            @if(isset($value['nom_pais']))
                <td>{{ $value['nom_pais'] }}</td>
            @endif
        </tr>
    @endforeach
</tbody>

```

```

        @if(isset($value['nom_ciudad']))
            <td>{{ $value['nom_ciudad'] }}</td>
        @endif
        @if(isset($value['calle_res']))
            <td>{{ $value['calle_res'] }}</td>
        @endif
        @if(isset($value['correo_electronico']))
            <td>{{ $value['correo_electronico'] }}</td>
        @endif
        @if(isset($value['fecha_inicio']))
            <td>{{ $value['fecha_inicio'] }}</td>
        @endif
        @if(isset($value['fecha_fin']))
            <td>{{ $value['fecha_fin'] }}</td>
        @endif
    </tr>
@endforeach
</tbody>
</table>
@endsection
@section('page-js-script')
    <script type="text/javascript" src="https://cdn.datatables.net/v/bs4/jszip-2.5.0/dt-1.10.18/b-
1.5.2/b-html5-1.5.2/cr-1.5.0/r-2.2.2/sl-1.2.6/datatables.min.js"></script>
<script>
    $("document").ready(function() {
        $("#tabla").DataTable({
            dom: 'Bfrtip',
            buttons: [
                {
                    extend: 'excelHtml5',
                    text: 'Exportar a excel',
                    exportOptions: {
                        modifier: {
                            page: 'current'
                        }
                    }
                }
            ],
            select: {
                style: 'single'
            },
            responsive: true,
            "language": {
                "sProcessing": "Procesando...",

```

```

        "sLengthMenu": "Mostrar _MENU_ registros",
        "sZeroRecords": "No se encontraron resultados",
        "sEmptyTable": "Ningún dato disponible en esta tabla",
        "sInfo": "Mostrando registros del _START_ al _END_ de un total de _TOTAL_
registros",
        "sInfoEmpty": "Mostrando registros del 0 al 0 de un total de 0 registros",
        "sInfoFiltered": "(filtrado de un total de _MAX_ registros)",
        "sInfoPostFix": "",
        "sSearch": "Buscar:",
        "sUrl": "",
        "sInfoThousands": ",",
        "sLoadingRecords": "Cargando...",
        "oPaginate": {
            "sFirst": "Primero",
            "sLast": "Último",
            "sNext": "Siguiente",
            "sPrevious": "Anterior"
        },
        "oAria": {
            "sSortAscending": ": Activar para ordenar la columna de manera ascendente",
            "sSortDescending": ": Activar para ordenar la columna de manera descendente"
        }
    }
});
})
</script>
@endsection

```

ListasController.php

```

<?php
namespace App\Http\Controllers;
use Illuminate\Http\Request;
use App\Cliente;
use App\Tipo_habitacion;
use App\Pais;
use App\Habitacion;
use App\Bloqueo;
use App\Reservacion;
use App\Dia_reservada;
use Carbon\Carbon;
use App\Ciudad;
use Illuminate\Support\Facades\DB;

class ListasController extends Controller
{

```

```

public function clientesHospedados_index(){
    return view('admin.listas.clientesHospedados.index');
}

public function clienteshospedados_show(Request $request){
    $cliente = cliente::join('Reservacion', 'Cliente.cod_cliente', 'Reservacion.cod_cliente');
    if(isset($request->fecha)){
        $request->fecha = str_replace("a", "", $request->fecha);
        $request->fecha = str_replace(" ", "", $request->fecha);
        $arra1 = str_split($request->fecha, 10 );
        $cliente = $cliente->where('fecha_inicio','>=' , $arra1[0])->where('fecha_fin', '<=', $arra1[0])
            ->orWhere('fecha_inicio','<=' , $arra1[1])->where('fecha_fin', '>=', $arra1[1]);
    }else{
        $cliente = $cliente->where('fecha_inicio','>=' ,date('Y-m-d'))->where('fecha_fin', '<=', date('Y-m-
d'))
            ->orWhere('fecha_inicio','<=' ,date('Y-m-d'))->where('fecha_fin', '>=', date('Y-m-d'));
    }
    if(isset($request->cod_cliente)){
        $cliente = $cliente->addSelect('Cliente.cod_cliente');
    }
    if(isset($request->nombre)){
        $cliente = $cliente->addSelect('Cliente.nombre');
    }
    if(isset($request->ap_paterno)){
        $cliente = $cliente->addSelect('Cliente.ap_paterno');
    }
    if(isset($request->ap_materno)){
        $cliente = $cliente->addSelect('Cliente.ap_materno');
    }
    if(isset($request->telefono)){
        $cliente = $cliente->addSelect('Cliente.telefono');
    }
    if(isset($request->ciudad)){
        $cliente = $cliente->join('Ciudad', 'Ciudad.cod_ciudad', 'Cliente.ciudad_res')->
        addSelect('Ciudad.nom_ciudad');
    }
    if(isset($request->pais)){
        $cliente = $cliente->join('Pais', 'Ciudad.pais', 'Pais.cod_pais')->addSelect('Pais.nom_pais');
    }
    if(isset($request->direccion)){
        $cliente = $cliente->addSelect('Cliente.calle_res');
    }
    if(isset($request->correo_electronico)){
        $cliente = $cliente->addSelect('Cliente.correo_electronico');
    }
}

```

```
if(isset($request->check_in)){
    $cliente = $cliente->addSelect('Reservacion.fecha_inicio');
}
if(isset($request->fecha_fin)){
    $cliente = $cliente->addSelect('Reservacion.fecha_fin');
}
$cliente = $cliente->get();

return view('admin.listas.clienteshospedados.show')->with('resp', $cliente)->with('request',
$request);
}
}
```


Eliminar PagoPagoController.php

```
<?php
namespace App\Http\Controllers;
use App\Moneda;
use App\Pago;
use App\Valor_moneda;
use App\Reservacion;
use App\Tipo_habitacion;
use Carbon\Carbon;
use Illuminate\Http\Request;
class PagoController extends Controller
{
    public function destroy($cod_pago)
    {
        Pago::destroy($cod_pago);
    }
}
```


Consultar clientesgestor.blade.php

@extends('admin.template.main')

@section('title', 'Gestor Clientes')

@section('page-css-style')

```
<link rel="stylesheet" type="text/css" href="https://cdn.datatables.net/v/bs4/jszip-2.5.0/dt-1.10.18/b-1.5.2/b-html5-1.5.2/cr-1.5.0/r-2.2.2/sl-1.2.6/datatables.min.css"/>
```

```
<link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/flatpickr/dist/flatpickr.min.css">
```

@endsection

@section('content')

```
<div class="card bg-dark text-light">
```

```
<div class="card-header ">
```

```
<h4>Ingresa datos a consultar</h4>
```

```
</div>
```

```
<div class="card-body">
```

```
<form >
```

```
<div class="row">
```

```
<div class="col-lg-3 col-md-6 col-sm-12">
```

```
<div class="form-group">
```

```
<label for="codigo">Codigo:</label>
```

```
<input id="codigo" type="text" class="form-control" name="codigo"/>
```

```
</div>
```

```
</div>
```

```
<div class="col-lg-3 col-md-6 col-sm-12">
```

```
<div class="form-group">
```

```
<label for="tipo_documento">Tipo de documento:</label>
```

```
<select id="tipo_documento" class="form-control" name="tipo_documento">
```

```
<option value=""></option>
```

```
<option value="carnet">carnet</option>
```

```
<option value="pasaporte">pasaporte</option>
```

```
</select>
```

```
</div>
```

```
</div>
```

```
<div class="col-lg-3 col-md-6 col-sm-12">
```

```
<div class="form-group">
```

```
<label for="nro_doc">Numero de documento:</label>
```

```
<input id="nro_doc" type="text" class="form-control" name="nro_doc"/>
```

```
</div>
```

```
</div>
```

```
<div class="col-lg-3 col-md-6 col-sm-12">
```

```
<div class="form-group">
```

```
<label for="nombre">Nombre:</label>
```

```
<input id="nombre" type="text" class="form-control" name="nombre"/>
```

```
</div>
```

```

</div>
<div class="col-lg-3 col-md-6 col-sm-12">
  <div class="form-group">
    <label for="ap_paterno">Apellido paterno:</label>
    <input id="ap_paterno" type="text" class="form-control" name="ap_paterno"/>
  </div>
</div>
<div class="col-lg-3 col-md-6 col-sm-12">
  <div class="form-group">
    <label for="ap_materno">Apellido materno:</label>
    <input id="ap_materno" type="text" class="form-control" name="ap_materno"/>
  </div>
</div>
<div class="col-lg-3 col-md-6 col-sm-12">
  <div class="form-group">
    <label for="correo_electronico">Correo electronico:</label>
    <input id="correo_electronico" type="text" class="form-control"
name="correo_electronico"/>
  </div>
</div>
<div class="col-lg-3 col-md-6 col-sm-12">
  <div class="form-group">
    <label for="pais">Pais de origen:</label>
    <select id="pais" class="form-control" name="pais">
      <option value=""></option>
      @foreach($paises as $pais)
        <option value="{{ $pais->nom_pais }}">{{ $pais->nom_pais }}</option>
      @endforeach
    </select>
  </div>
</div>
<div class="col-lg-3 col-md-6 col-sm-12">
  <div class="form-group">
    <label for="ciudad">Ciudad de origen:</label>
    <select id="ciudad" class="form-control" name="ciudad"/>
    </select>
  </div>
</div>
<div class="col-lg-3 col-md-6 col-sm-12">
  <div class="form-group">
    <label for="tipoHabitacion">Tipo de Habitacion</label>
    <select id="cod_tipo_habitacion" class="form-control" name="cod_tipo_habitacion">
      <option value=""></option>
      @foreach($tipos as $tipo)

```

```

        <option value="{{ $tipo->cod_tipo_habitacion }}">{{ $tipo->nom_tipo }}</option>
    @endforeach
</select>
</div>
</div>
<div class="col-lg-3 col-md-6 col-sm-12">
    <div class="form-group">
        <label for="habitacion">Habitacion:</label>
        <select id="habitacion" type="text" class="form-control" name="habitacion"/>
    </select>
    </div>
</div>
<div class="col-lg-3 col-md-6 col-sm-12">
    <div class="form-group">
        <label for="">Buscar por rango de fecha:</label>
        <input id="fecha-range" type="text" class="form-control" name="fecha-ini"/>
    </div>
</div>
<div class="col-lg-3 col-md-6 col-sm-12">
    <button type="button" class="btn btn-primary" id="consultar_btn"
name="consultar_btn">Consultar</button>
    </div>
</div>
</form>
</div>
</div>
<div id="tabla">
</div>
@endsection
@section('page-js-script')
    <script type="text/javascript" src="https://cdn.datatables.net/v/bs4/jszip-2.5.0/dt-1.10.18/b-
1.5.2/b-html5-1.5.2/cr-1.5.0/r-2.2.2/sl-1.2.6/datatables.min.js"></script>
    <script src="https://cdn.jsdelivr.net/npm/flatpickr"></script>
    <script src="https://npmcdn.com/flatpickr/dist/l10n/es.js"></script>
    <script type="text/javascript" src="{{ asset('js/clientesmain.js') }}"></script>
@endsection

```

clientesmain.js

```

window.tablaClientes = {};
window.rango = {};
window.eliminar = function() {
    var id = window.tablaClientes.rows( { selected: true } ).data()[0].cod_cliente;
    if (confirm('Seguro que quiere eliminar el cliente?')) {
        $.get( "Clientes/"+id+"/destroy");
    }
}

```

```

    window.tablaClientes.rows( { selected: true } ).remove().draw();
  }
}

window.crearTabla = function( data){
  $('#tabla').empty()
  $('#tabla').append('<div class = "card"><div class="card-header"></div><div class="card-
body"><table id ="tbl_cli"class="table"></table></div></div>');
  columnas = '<tr><th>Codigo</th><th>Tipo de documento</th><th>Nro°
documento</th><th>Nombre</th><th>Apellido paterno</th><th>Apellido
materno</th><th>Telefono</th><th>Correo electronico</th><th>Pais</th><th>Ciudad</th></tr>';
  thead = '<thead>'+columnas+'</thead>';
  tfoot = '<tfoot>'+columnas+'</tfoot>';
  $('#tbl_cli').append(thead+tfoot);

  window.tablaClientes = $('#tbl_cli').DataTable( {
    dom: 'Bfrtip',
    data: data,
    buttons: [
      {
        text : 'Modificar',
        attr : {id : 'modificar',
          disabled : true
        }
      },
      {
        text : 'Eliminar',
        attr : {id : 'eliminar',
          onclick:'eliminar()',
          disabled : true
        }
      }
    ],
    {
      extend: 'excelHtml5',
      text: 'Exportar a excel',
      exportOptions: {
        modifier: {
          page: 'current'
        }
      }
    }
  ],
  columns: [
    { data: 'cod_cliente' },

```

```

    { data: 'tipo_documento' },
    { data: 'nro_doc' },
    { data: 'nombre' },
    { data: 'ap_paterno' },
    { data: 'ap_materno' },
    { data: 'telefono' },
    { data: 'correo_electronico' },
    { data: 'ciudad.pais.nom_pais' },
    { data: 'ciudad.nom_ciudad' },
  ],
  select: {
    style: 'single'
  },
  responsive: true,
  "language": {
    "sProcessing": "Procesando...",
    "sLengthMenu": "Mostrar _MENU_ registros",
    "sZeroRecords": "No se encontraron resultados",
    "sEmptyTable": "Ningún dato disponible en esta tabla",
    "sInfo": "Mostrando registros del _START_ al _END_ de un total de _TOTAL_ registros",
    "sInfoEmpty": "Mostrando registros del 0 al 0 de un total de 0 registros",
    "sInfoFiltered": "(filtrado de un total de _MAX_ registros)",
    "sInfoPostFix": "",
    "sSearch": "Buscar:",
    "sUrl": "",
    "sInfoThousands": ",",
    "sLoadingRecords": "Cargando...",
    "oPagate": {
      "sFirst": "Primero",
      "sLast": "Último",
      "sNext": "Siguiente",
      "sPrevious": "Anterior"
    },
    "oAria": {
      "sSortAscending": ": Activar para ordenar la columna de manera ascendente",
      "sSortDescending": ": Activar para ordenar la columna de manera descendente"
    }
  }
}
});

window.tablaClientes.on( 'select', function ( e, dt, type, indexes ) {
  if ( type === 'row' ) {
    var data = window.tablaClientes.rows( indexes ).data().pluck( 'cod_cliente' );
    $('#modificar').attr('disabled',false);
    $('#eliminar').attr('disabled',false);
  }
});

```

```

        $('#modificar').attr('onclick',"window.location.href='Clientes/'+data[0]+"/edit'");
    }
});
window.tablaClientes.on( 'deselect', function ( e, dt, type, indexes ) {
    if ( type === 'row' ) {
        $('#modificar').attr('disabled',true);
        $('#eliminar').attr('disabled',true);
    }
});
}
window.init = function () {
    flatpickr.localize(flatpickr.l10ns.es);
    window.rango= flatpickr("#fecha-range",{
        dateFormat: "d-m-Y",
        mode : 'range'
    });
    $("body").on('change','#pais',function () {
        if($("#pais").val()){
            $('#ciudad').empty();
            $.getJSON( "http://127.0.0.1:8000/api/ciudades?",{
                pais: $("#pais").val()
            })
            .done(function( data, textStatus, jqXHR ) {
                $("#ciudad").append('<option value=""></option>')
                for (var i = 0; i < data.length; i++) {
                    $("#ciudad").append('<option
value='+data[i].nom_ciudad+'>'+data[i].nom_ciudad+'</option>')
                }
            })
            .fail(function(jqXHR, textStatus, errorThrown) {
                console.log( "error" );
            })
        }else {
            $('#ciudad').empty();
        }
    })
    $("body").on('change','#cod_tipo_habitacion',function () {
        if($("#cod_tipo_habitacion").val()){
            $('#habitacion').empty();
            $.getJSON( "http://127.0.0.1:8000/api/habitaciones?",{
                cod_tipo_habitacion: $("#cod_tipo_habitacion").val()
            })
            .done(function( data, textStatus, jqXHR ) {
                $("#habitacion").append('<option value=""></option>')

```

```

        for (var i = 0; i < data.length; i++) {
            $('#habitacion').append('<option
value='+data[i].cod_habitacion+'>'+data[i].nom_hab+'</option>')
        }
    })
    .fail(function(jqXHR, textStatus, errorThrown) {
        console.log( "error" );
    })
    }else {
        $('#habitacion').empty();
    }
}
}
$("document").ready(function() {
    window.init();
    $("body").on("click", "#consultar_btn", function() {
        var nombre = $("#nombre").val()
        , codigo = $("#codigo").val()
        , ap_paterno = $("#ap_paterno").val()
        , ap_materno = $("#ap_materno").val()
        , nro_doc = $("#nro_doc").val()
        , tipo_documento = $("#tipo_documento").val()
        , correo_electronico = $("#correo_electronico").val()
        , pais = $("#pais").val()
        , ciudad = $("#ciudad").val()
        , habitacion = $("#habitacion").val()
        , cod_tipo_habitacion = $("#cod_tipo_habitacion").val()
        , rango = window.rango.selectedDates
        , to = ""
        , from = ""
        ;
        if (Array.isArray(rango) && rango.length) {
            from = flatpickr.formatDate(rango[0], "Y-m-d h:i:S");
            if (rango.length > 1) {
                to = flatpickr.formatDate(rango[1], "Y-m-d h:i:S");
            }
        }
    }
    $.getJSON( "http://127.0.0.1:8000/api/clientes?", { "cod_cliente": codigo,
        "tipo_documento": tipo_documento,
        "nombre" : nombre,
        "ap_paterno" : ap_paterno,
        "ap_materno" : ap_materno ,
        "correo_electronico": correo_electronico,
        "nro_doc": nro_doc,

```

```

        "pais":pais,
        "ciudad":ciudad,
        "habitacion": habitacion,
        "cod_tipo_habitacion": cod_tipo_habitacion,
        "from": from,
        "to":to} )
    .done(function( data, textStatus, jqXHR ) {
        if (data) {
            window.crearTabla(data);
        }
    })
    .fail(function(jqXHR, textStatus, errorThrown) {
        console.log( "error" );
    })
    })
})

```

ClienteController.php

```

<?php
namespace App\Http\Controllers;
use Illuminate\Http\Request;
use App\Cliente;
use App\Pais;
use App\Ciudad;
use App\Reserva;
use App\Dia_reservada;
use App\Tipo_habitacion;
use App\Reservacion;
use laracast\flash\flash;
use app\config\database;
use App\Http\Requests\ClienteRequest;
use Freshwork\ChileanBundle\Rut;

class ClienteController extends Controller
{
    public function index()
    {
        $paises = Pais::orderBy('cod_pais', 'ASC')->get();
        $tipos = Tipo_habitacion::orderBy('cod_tipo_habitacion', 'ASC')->get();
        return view('admin.Cliente.gestor')->with('paises', $paises)
            ->with('tipos', $tipos)
        ;
    }
}

```

```
}
```

```
public function search(Request $request)
```

```
{
```

```
    $resp = new Cliente;
```

```
    $wResp = array();
```

```
    $param = $request->all();
```

```
    $ciudad = false;
```

```
    $go = false;
```

```
    if (isset($param['tipo_documento']) ) {
```

```
        $go = true;
```

```
        $resp = $resp->where('tipo_documento',$param['tipo_documento']);
```

```
    }
```

```
    if (isset($param['nro_doc']) ) {
```

```
        $go = true;
```

```
        $resp = $resp->where('nro_doc',$param['nro_doc']);
```

```
    }
```

```
    if (isset($param['nombre']) ) {
```

```
        $go = true;
```

```
        $resp = $resp->where('Nombre',$param['nombre']);
```

```
    }
```

```
    if (isset($param['ap_paterno']) ) {
```

```
        $go = true;
```

```
        $resp = $resp->where('Ap_paterno',$param['ap_paterno']);
```

```
    }
```

```
    if (isset($param['ap_materno']) ) {
```

```
        $go = true;
```

```
        $resp = $resp->where('ap_materno',$param['ap_materno']);
```

```
    }
```

```
    if (isset($param['cod_cliente']) ) {
```

```
        $go = true;
```

```
        $resp = $resp->where('Cliente.cod_cliente',$param['cod_cliente']);
```

```
    }
```

```
    if (isset($param['correo_electronico']) ) {
```

```
        $go = true;
```

```
        $resp = $resp->where('Correo_electronico',$param['correo_electronico']);
```

```
    }
```

```
    if (isset($param['from']) || isset($param['to']) ||
```

```
        isset($param['habitacion']) || isset($param['cod_tipo_habitacion']) ){
```

```
        $go = true;
```

```
        $resp = $resp->join('Reservacion','Reservacion.cod_cliente',
                            'Cliente.cod_cliente')
```

```

->join('Dia_reservada','Reservacion.cod_reservacion',
      'Dia_reservada.cod_reservacion')
->select('Cliente.cod_cliente',
        'tipo_documento',
        'nro_doc',
        'nombre',
        'telefono',
        'ap_paterno',
        'ap_materno',
        'telefono',
        'correo_electronico',
        'cod_ciudad',
        'nom_ciudad',
        'cod_pais',
        'nom_pais',
        'calle_res'
        )
->distinct() ;
}
if (isset($param['from']) && isset($param['to']) ) {
    $go = true;
    $resp = $resp->where('fecha_reservada', '>=', $param['from'])
        ->where('fecha_reservada', '<=', $param['to'])
        ;
}elseif(isset($param['from'])){
    $go = true;
    $resp = $resp->where('fecha_reservada', '>=', $param['from']);
}
if (isset($param['habitacion']) && isset($param['cod_tipo_habitacion']) ) {
    $go = true;
    $resp = $resp->join('Habitacion','Habitacion.cod_habitacion',
        'Dia_reservada.cod_habitacion')
        ->where('Habitacion.cod_habitacion', $param['habitacion'])
        ;
}elseif (isset($param['cod_tipo_habitacion'])) {
    $go = true;
    $resp = $resp->join('Habitacion','Habitacion.cod_habitacion',
        'Dia_reservada.cod_habitacion')
        ->where('Habitacion.cod_tipo_habitacion',
            $param['cod_tipo_habitacion'])
        ;
}
if (isset($param['ciudad']) && isset($param['pais']) ){
    $go = true;

```

```

    $p = Pais::where( 'Nom_pais',$param['pais'])->first();
    $c = Ciudad::where( 'nom_ciudad',$param['ciudad']);
    $c = $c->where( 'pais', $p['cod_pais'])->first();
    $resp = $resp->where('Ciudad_res',$c['cod_ciudad']);

}elseif (isset($param['pais'])) {
    $go = true;
    $ciudad = true;
    $p = Pais::where( 'Nom_pais',$param['pais'])->first();
    $resp = $resp->join('ciudad','Ciudad_res','ciudad.cod_ciudad')
        ->where('pais',$p['cod_pais']);
}

if ($go && $ciudad) {
    $resp = $resp ->join('pais','pais','cod_pais')
        ->where('estado_cliente',0);
    $resp = $resp->get();
}elseif (!$ciudad) {
    $resp = $resp ->join('ciudad', 'ciudad_res','ciudad.cod_ciudad')
        ->join('pais','ciudad.pais','cod_pais')
        ->where('estado_cliente',0)
        ;

    $resp = $resp->get();
}
if($go || !$ciudad){
    foreach ($resp as $value) {
        array_push($wResp, array('cod_cliente' => $value['cod_cliente'],
            'tipo_documento' => $value['tipo_documento'],
            'nro_doc' => $value['nro_doc'],
            'nombre' => $value['nombre'],
            'ap_paterno' => $value['ap_paterno'],
            'ap_materno' => $value['ap_materno'],
            'telefono' =>$value['telefono'],
            'correo_electronico' => $value['correo_electronico'],
            'ciudad' =>array(
                'cod_ciudad'=>$value['cod_ciudad'],
                'nom_ciudad'=>$value['nom_ciudad'],
                'pais'=>array('cod_pais' => $value['cod_pais'],
                    'nom_pais'=>$value['nom_pais']
                )
            ),
            'calle'=> $value['calle_res']
        ));
    }
}

```

```

    }
}
return response()->json($wResp);
}
}

```

CiudadController.php

```

public function search(Request $request)
{
    $resp = new Ciudad;
    $wResp = array();
    $param = $request->all();
    if (isset($param['cod_ciudad']) ) {
        $resp = $resp->where('cod_ciudad',$param['cod_ciudad']);
    }
    if (isset($param['nom_ciudad']) ) {
        $resp = $resp->where('nom_ciudad',$param['nom_ciudad']);
    }
    if (isset($param['pais']) ) {
        $p = Pais::where('nom_pais',$param['pais'])->first();
        $resp = $resp->where('pais',$p['cod_pais']);
    }
    $resp = $resp->join('pais','pais','cod_pais');
    $resp = $resp->get();
    foreach ($resp as $value) {
        array_push($wResp, array('cod_ciudad'=>$value['cod_ciudad'],
                                'nom_ciudad'=>$value['nom_ciudad'],
                                'pais'=>array('cod_pais' => $value['cod_pais'],
                                                'nom_pais'=>$value['nom_pais']
                                ));
    }
    return response()->json($wResp);
}

```

HabitacionController.php

```

public function search(Request $request)
{
    $resp = new Habitacion;
    $wResp = array();
    $param = $request->all();
    if (isset($param['cod_habitacion']) ) {
        $resp = $resp->where('cod_habitacion',$param['cod_habitacion']);
    }
    if (isset($param['nom_hab']) ) {
        $resp = $resp->where('nom_hab',$param['nom_hab']);
    }
}

```

```

if (isset($param['cod_tipo_habitacion']) ) {
    $t = Tipo_habitacion::find($param['cod_tipo_habitacion'])->first();
    $resp = $resp->where('habitacion.cod_tipo_habitacion',$param['cod_tipo_habitacion']);
}
$resp=$resp->join('tipo_habitacion','tipo_habitacion.cod_tipo_habitacion',
'habitacion.cod_tipo_habitacion');
if (isset($param['capacidad_habitacion']) ) {
    $resp = $resp->where('capacidad_habitacion',$param['capacidad_habitacion']);
}
if (isset($param['cantidad_baños']) ) {
    $resp = $resp->where('cantidad_baños',$param['cantidad_baños']);
}
if (isset($param['precio_noche']) ) {
    $resp = $resp->where('precio_noche',$param['precio_noche']);
}
if (isset($param['tipo_baño']) ) {
    $resp = $resp->where('tipo_baño',$param['tipo_baño']);
}
$resp = $resp->where('estado_hab', '<>', 1);
$resp = $resp->get();
foreach ($resp as $value) {
    array_push($wResp, array('cod_habitacion'=>$value['cod_habitacion'],
        'nom_hab'=>$value['nom_hab'],
        'posicion'=>$value['posicion'],
        'estado_hab'=>$value['estado_hab'],
        'tipo_habitacion'=>array('cod_tipo_habitacion' => $value['cod_tipo_habitacion'],
            'nom_tipo'=>$value['nom_tipo'],
            'capacidad_habitacion'=>$value['capacidad_habitacion'],
            'cantidad_baños'=>$value['cantidad_baños'],
            'tipo_baño'=>$value['tipo_baño'],
            'precio_noche'=>$value['precio_noche'],
            ));
}
return response()->json($wResp);
}

```