ESCUELA POLITÉCNICA NACIONAL

FACULTAD DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA

DESARROLLO DE UN SISTEMA PROTOTIPO PARA LA GESTIÓN DE RESERVAS EN LÍNEA Y REGISTRO DE ENTRADA-SALIDA DE CLIENTES DE UN HOTEL

TRABAJO DE TITULACIÓN PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN ELECTRÓNICA Y REDES DE INFORMACIÓN

ROBERTO DAVID POZO ANDRADE
roberto.pozo@epn.edu.ec

DIRECTOR: MSc. FRANKLIN LEONEL SÁNCHEZ CATOTA
franklin.sanchez@epn.edu.ec

CODIRECTOR: MSc. GABRIEL ROBERTO LÓPEZ FONSECA
gabriel.lopez@epn.edu.ec

Quito, agosto 2018

AVAL

Certificamos que el presente trabajo fue desarrollado por Roberto David Pozo Andrade bajo nuestra supervisión.

MSc. FRANKLIN LEONEL SÁNCHEZ CATOTA
DIRECTOR DEL TRABAJO DE TITULACIÓN

MSc. GABRIEL ROBERTO LÓPEZ FONSECA
CODIRECTOR DEL TRABAJO DE TITULACIÓN

DECLARACIÓN DE AUTORÍA

Yo, Roberto David Pozo Andrade, declaro bajo juramento que el trabajo aquí descrito es de mi autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.

A través de la presente declaración cedo mis derechos de propiedad intelectual correspondientes a este trabajo, a la Escuela Politécnica Nacional, según lo establecido por la Ley de Propiedad Intelectual, por su Reglamento y por la normatividad institucional vigente.

ROBERTO DAVID POZO ANDRADE

DEDICATORIA

A mis padres

AGRADECIMIENTOS

Agradezco a mis padres, hermanos, amigos y compañeros por las risas de cada día, que nunca me han faltado.

ÍNDICE DE CONTENIDO

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

ÍNDICE DE SEGMENTOS DE CÓDIGO

RESUMEN

En el Ecuador existe un gran porcentaje de empresas dedicadas al turismo, entre ellas se encuentra el Hotel Playa Cristal, y al ser un sector importante de la economía es necesario potenciarlo mediante el uso de tecnologías de la información, es por ello que el presente trabajo de titulación se enfoca en la creación de un sistema que permita realizar la reserva de habitaciones en línea y el registro de entrada-salida de los clientes del hotel.

Dicho sistema se lo realizó en el lenguaje ASP.NET y C#, para los principales procesos, y otros lenguajes de programación, como JavaScript, que permiten el correcto funcionamiento del sistema. Una vez finalizado el desarrollo se implementó el sistema en un ambiente *cloud computing*, específicamente en *Amazon Web Services*.

Para la creación del sistema se realizaron varios procesos, estos comprenden desde el estudio teórico de los lenguajes de programación utilizados hasta la etapa de pruebas del prototipo implementado. En cada uno de los procesos se utilizaron varias herramientas de *software*, facilitando de esta manera la realización correcta de dicho proceso.

Una vez implementado el sistema y realizadas las pruebas correspondientes, se pudo verificar que las funciones cumplen con los requerimientos establecidos por el *Hotel Playa Cristal*. Además, se pudo comprobar el potencial que poseen los lenguajes de programación utilizados en cada uno de los aspectos en que se los utiliza.

Palabras clave: Hotel, ASP.NET, Cloud Computing, C#.

ABSTRACT

In Ecuador there is a big percentage of enterprises that are dedicated to tourism, one of them is the hotel Playa Cristal, and being an important part of the economy it is necessary to enhance it with all the technologies that are available nowadays, that's why the present project is focused in the creation of a system that allows the booking of rooms on-line and also the check-in and check-out in the hotel.

The system was developed in the language ASP.NET and C#, for the main purposes, and other programming languages, like JavaScript, which allows the right work of the system. Once the development was over the system is implemented on a cloud computing environment, specifically Amazon Web Services.

When the system was being developed many process were realized, these processes involve from the theoretical study of the programming languages to the test realized in the implemented system. In every process many software tools were used, making easier the right realization of the mentioned process.

Once the system was implemented and the test were performed it was verified that all the functions fulfil the requirements established by the Hotel Playa Cristal. Also, it could be verified the potential that the programming languages used have in every aspect in which they were used.

Keywords**:** Hotel, ASP.NET, Cloud Computing, C#.

1. INTRODUCCIÓN

El presente trabajo de titulación corresponde a la creación de un sistema, que permita realizar la gestión de reservas en línea de habitaciones y el registro de entrada-salida de clientes del hotel Playa Cristal.

El primer capítulo describe el estudio realizado sobre las tecnologías utilizadas en el presente trabajo de titulación, entre las cuales se encuentra HTML, ASP.NET, C#, SQL *y* los servicios de *Amazon Web Services* como EC2, RDS y S3.

El segundo capítulo describe la etapa de diseño, codificación e implementación del sistema. La etapa de diseño comprende desde el análisis de requerimientos realizado a una persona, encargada del hotel Playa Cristal, hasta la creación de los diagramas de clases del sistema.

En la etapa de codificación se describe la creación de la base de datos, incluyendo los procedimientos almacenados y la creación de todo el código en ASP.NET y C#, finalmente en la etapa de implementación se describe la creación de los servicios en *Amazon Web Services* y el despliegue de la aplicación en dichos servicios.

El tercer capítulo corresponde a los resultados de las pruebas de funcionamiento e integración realizadas al sistema, así como su análisis. Y el cuarto capítulo corresponde a las conclusiones y recomendaciones.

1.1 OBJETIVOS

El objetivo del presente trabajo de titulación es el de desarrollar un sistema prototipo, que permita realizar la gestión de reservas en línea y el registro de entrada-salida de los clientes del hotel Playa Cristal, implementándolo en *Amazon Web Services*.

Los objetivos específicos son:

- Estudiar las diferentes herramientas que servirán para el desarrollo e implementación del sistema.
- Diseñar cada uno de los componentes del sistema prototipo propuesto.
- Implementar cada uno de los componentes diseñados.

1.2 ALCANCE

El trabajo de titulación presentado propone el desarrollo de un sistema que tiene dos propósitos principales, el primero es realizar la gestión de reservas de habitaciones en línea y el segundo es la realización del registro de entrada-salida de los clientes del hotel Playa Cristal.

El sistema completo se lo realizará en el modelo de desarrollo web unificado ASP.NET*,* con el lenguaje de programación orientada a objetos C#, para el almacenamiento de los datos se utilizará SQL Server, todo esto implementado en *Amazon Web Services Free Tier*.

En la etapa de desarrollo se utilizará Microsoft *Visual Studio 2017* y SQL Server *2012 Express*, una vez desarrollado, se creará una instancia Windows *en Amazon Elastic Compute Cloud*, el cual es un "servicio web que proporciona capacidad informática en la nube segura y de tamaño modificable [1]", de características *t2.micro*.

Los archivos se alojarán en Amazon *Simple Storage Service,* el cual "permite recopilar, almacenar y analizar datos de forma cómoda y sencilla, independientemente de su formato y a escala masiva [2]". Posee hasta 5 GB de almacenamiento incluidos en AWS *Free Tier.*

También se creará una base de datos *SQL Server* en Amazon *Relational Database Service*, el cual "es un servicio administrado de bases de datos [3]", cuya capacidad *Free Tier* permite hasta 20 GB de tamaño. El esquema de implementación del sistema en AWS se lo muestra en la Figura 1.1. El sistema estará dividido en cinco módulos como se describen a continuación:

- El primero corresponde al sistema de reserva de habitaciones.
- El segundo corresponde a un módulo de gestión administrativa, que permitirá la creación, lectura y actualización de usuarios del sistema, habitaciones del hotel y clientes.
- El tercer módulo permitirá gestionar toda la información de las reservas de habitaciones realizadas en el primer módulo por parte de los clientes, esto implica la creación, modificación o cancelación de una reserva.
- En el cuarto módulo se controla el proceso de registro de entrada-salida de los clientes del hotel, además en este módulo se podrá generar una factura

electrónica con los datos del registro.

- El quinto módulo corresponde a la generación de reportes, por períodos, con la información almacenada en la base de datos.

Los módulos dos, tres y cuatro son para el uso del personal encargado de la administración del hotel, por lo tanto, requieren de autenticación.

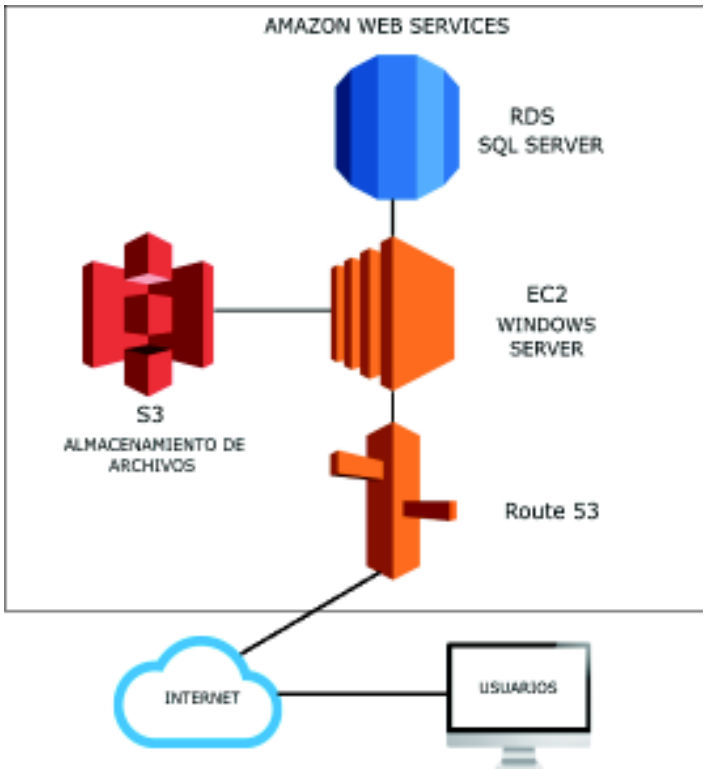

Figura 1.1: Implementación del sistema en AWS.

1.2.1 Contenido de cada uno de los módulos

- **Primer módulo**
  - Formulario web 1: En este formulario se podrá realizar las reservas de habitaciones, mediante el ingreso de datos que se definan en el proceso de análisis de requerimientos.

Para los siguientes módulos se creará un formulario web de inicio de sesión mediante usuario y contraseña, que permite acceder a una página maestra, que contiene los enlaces de cada uno de los formularios web administrativos, dependiendo del tipo de usuario que sea. El usuario podrá acceder a todos los formularios, si es administrador, o solo a ciertos formularios si es usuario normal.

- **Segundo módulo**
    - Formulario web 2: Permitirá crear, leer y actualizar usuarios del sistema.
    - Formulario web 3: Permitirá crear, leer y actualizar las habitaciones que existen en el hotel
    - Formulario web 4: Permitirá crear, leer y actualizar los datos de clientes que se han registrado en el hotel
    - Formulario web 5: Este formulario permitirá modificar el nombre de usuario y contraseña del usuario que se encuentra conectado
    - Formulario web 6: Permitirá visualizar que usuarios se encuentran conectados en ese instante y desconectarlos si se desea, este formulario es emergente.
- **Tercer módulo**
    - Formulario web 7: En este formulario se podrá realizar la confirmación de una reserva, para ello se visualizará que habitaciones se encuentran disponibles y las reservas realizadas en el primer módulo, asignando a un cliente dicha habitación, adicionalmente se enviará un correo de confirmación de reserva al usuario.
    - Formulario web 8: Permitirá visualizar las reservas pendientes, seleccionar una reserva y modificarla o cancelarla y además permitirá crear una nueva reserva de una habitación.
- **Cuarto módulo**
    - Formulario web 9: Este formulario permite el registro de ingreso de un cliente con todos los datos que se definan en el proceso de análisis de requerimientos.
    - Formulario web 10: Permitirá el registro de la salida de un cliente, se asignará la fecha y hora de salida del hotel, luego se realizará el cálculo de tiempo de permanencia en el hotel, generando con estos datos una factura electrónica que se enviará al correo del cliente (cabe aclarar que dicha factura no poseerá validez fiscal ni estará vinculada a un sistema contable externo ya que es un prototipo).
- **Quinto módulo**
    - Formulario web 11**:** En este formulario se podrá crear un reporte de todas las reservas de habitaciones generadas en un determinado período de tiempo.
    - Formulario web 12: En este formulario se podrá generar un reporte de los registros de ingreso-salida de los clientes, en un período de tiempo determinado.

1.3 MARCO TEÓRICO

1.3.1 Aplicaciones Web

Como se menciona en [4] una aplicación web es un programa de computación que mediante el uso de tecnologías web realiza una tarea en Internet, existen millones de personas que hacen uso de este servicio a diario, mediante páginas web que facilitan el intercambio de cualquier tipo de información entre los diferentes usuarios, en general son simples y cumplen con una tarea específica, pero también existen páginas web más complejas que permiten realizar varias tareas.

La primera aplicación web que se publicó consistía en simplemente archivos informativos acerca de la *World Wide Web* disponible en [5]. Entre los diferentes tipos de aplicaciones web que existen hoy en día se puede diferenciar principalmente dos tipos de aplicaciones: dinámicas y estáticas.

a. Aplicaciones Web Estáticas

Como se menciona en [6] las aplicaciones web del tipo estáticas son aquellas que muestran el mismo contenido cada vez que se realiza una solicitud por parte del usuario hacia el servidor, como se muestra en la Figura 1.2, solo cambian cuando el administrador web actualiza el contenido reemplazando los archivos en el servidor web. El principal uso que se tiene para este tipo de aplicaciones es brindar información en general.

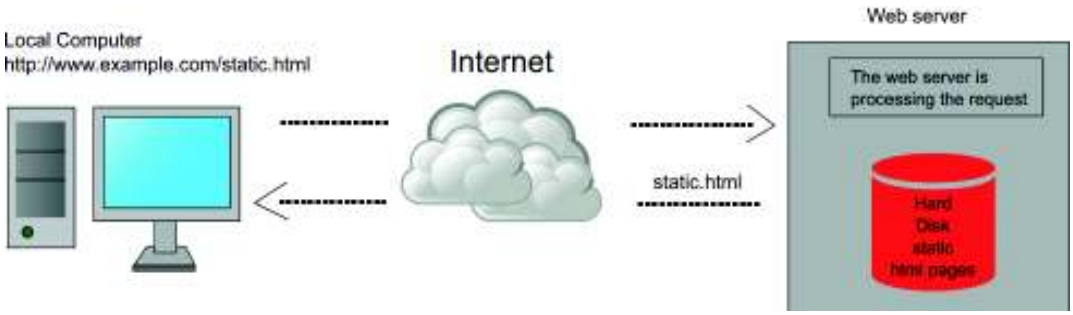

Figura 1.2: Funcionamiento página web estática [7]

b. Aplicaciones Web Dinámicas

Como se menciona en [8] una página dinámica es aquella que provee de contenido personalizado para el usuario basado en los resultados de una búsqueda o una solicitud. Se utilizan generalmente lenguajes del lado del cliente como JavaScript o en el lado

servidor con lenguajes como PHP o ASP para mostrar resultados personalizados. Otra característica de este tipo de aplicaciones web es que generalmente están conectadas a una base de datos que provee la información necesaria.

Existen varios gestores de páginas web dinámicas como WordPress el cual según [9] es un sistema de administración de contenido o CMS, que provee de más de un cuarto de los nuevos sitios web. Hoy en día los CMS más conocidos son: WordPress, Joomla y Drupal, cada uno ofrece diferentes características con el mismo propósito de administrar la información y datos que poseen las páginas web. El funcionamiento de dichos CMS se encuentra esquematizado en la Figura 1.3.

Figura 1.3: Funcionamiento de una CMS [10]

1.3.2 Modelo de Objeto del Documento [11]

Existen varias tecnologías web que permiten el desarrollo de aplicaciones, como el Modelo de Objeto del Documento que permite, entre otras cosas, identificar a los elementos que existen en una aplicación web.

En la especificación [11] del *World Wide Web Consortium* se describe que el modelo de objeto del documento, denominado DOM, es una interfaz de programación de aplicaciones, denominadas API, para archivos HTML y XML. Este es el que define que estructura lógica poseen y en qué forma se accede y se manipula los documentos.

“Como una especificación W3C, un objetivo importante del DOM es proveer una interfaz de programación estándar que se puede usar en una vasta variedad de ambientes y aplicaciones [11]”. Por lo que se ha considerado como la base para el desarrollo de todas las aplicaciones actuales, ya que puede utilizarse en cualquier lenguaje de programación.

1.3.3 HTML

Un lenguaje de marcado es un sistema para realizar anotaciones en un documento, de tal manera que sea distinguible del texto en su sintaxis. Existen varios lenguajes de marcado entre los cuales se encuentran HTML y XML.

El *HyperText Markup Language* o HTML "es un lenguaje abstracto que las aplicaciones pueden representar potencialmente en la memoria de muchas maneras posibles, y que puede ser transmitido utilizando cualquier número de posibles sintaxis concretas [12]". Existen dos tipos de sintaxis concretas definidas en [12], la primera es la sintaxis HTML y la segunda es la sintaxis XML.

Todos los elementos de HTML poseen etiquetas las cuales poseen una estructura específica detallada en [12], estas etiquetas son de inicio y de fin. Las etiquetas de inicio consisten en las siguientes partes, en el siguiente orden:

1. El carácter "<"
2. El nombre de la etiqueta del elemento
3. Opcional, uno o más atributos, cada uno de los cuales debe estar precedido por uno o más caracteres de espacio
4. Opcional, uno o más caracteres de espacio
5. Opcional, un carácter "/", que solo puede estar presente si el elemento es vacío.
6. El carácter ">"

Las etiquetas de fin consisten en las siguientes partes, en el siguiente orden:

1. El carácter "<"
2. El carácter "/"
3. El nombre de la etiqueta del elemento
4. Opcional, uno o más caracteres de espacio
5. El carácter ">"

De esta manera se definen todos los elementos HTML, permitiendo al desarrollador identificar las características que ofrecen dichos elementos en la construcción de una aplicación web. También existen elementos que pueden contener elementos anidados, en dicho caso los dos elementos deben poseer sus respectivas etiquetas de inicio y de fin en el orden correcto.

a. HTML5

Según [13] HTML5, última versión de HTML representa dos significados diferentes:

- Es una versión de HTML que posee nuevos elementos, comportamientos y atributos.
- Las tecnologías que utiliza permiten a las aplicaciones web ser más diversos y poseer un mayor alcance.

Esta versión de HTML fue diseñada para que pueda ser utilizado por cualquier desarrollador de *Open Web*, mejorando características como la semántica, la conectividad, operación sin conexión a Internet, multimedia, etc.

Dentro de la semántica se agregaron nuevas secciones, que permiten describir la funcionalidad de los elementos que se encuentren dentro de ellas. En cuanto a multimedia se agregaron los elementos *<audio>* y *<video>,* de esta manera se puede prescindir de otros programas externos que se encargaban del contenido multimedia, además se incluye la tecnología *WebRTC,* que permite conectarse en tiempo real directamente desde el navegador.

1.3.4 CSS

Las Hojas de Estilo en Cascada, cuyas siglas son CSS, "son un mecanismo simple para añadir estilos a documentos Web [14]". *CSS* cumple la función de dar un estilo a cualquier documento ya sea HTML o XML, separando de esta manera el contenido de la presentación.

Mediante reglas definidas en un documento CSS se puede modificar cualquier característica visual que posea un elemento, para poder indicar que elemento se requiere modificar se utilizan selectores y para indicar que característica es la que se va a modificar se utiliza la declaración.

En el Segmento de Código 1.1 se puede observar el selector: h1 y la declaración entre llaves definiendo la característica de color igual a rojo. Luego de definir los selectores y declaraciones correspondientes a todos los elementos que deseemos modificar se tiene que indicar a que documento se va a vincular el documento CSS.

```
1                                h1 { color: red ; }
```

Segmento de Código 1.1: Ejemplo de CSS

La manera más directa de realizar la vinculación entre el documento HTML y el documento CSS es incrustándolo dentro de la cabecera del documento HTML como se puede observar en el Segmento de Código 1.2. La razón para ubicarlo en la cabecera es porque así los CSS serán reconocidos antes de los demás elementos de la página.

```
<html>
    <head>
      <title>Ejemplo CSS</title>
      <!--En style se ubica el código CSS-->
        <style>
        <!--Se aplica el estilo a body-
        -> body {
          <!--Características de los elementos-->
          font-family:georgia,"times new roman",
          serif; color: blue;
          padding - left : 11 em;
          background-color:#000000
          ;}
        </style></head>
    <body>
      <!--Header 1 con el texto de ejemplo-->
      <h1>Letra de ejemplo</h1>
    </body></html>
```

Segmento de Código 1.2: Ejemplo de inserción de *CSS* en un documento *HTML*

a. Bootstrap

Hoy en día, al momento de diseñar una página web, se debe considerar la adaptabilidad que poseen dichas páginas a los diferentes tamaños de pantalla que existen, principalmente a las pantallas de los teléfonos inteligentes, utilizados a diario para acceder a contenido en Internet. Es por ello que existe *Bootstrap* el cual "es un kit de herramientas de código abierto para desarrollar con HTML, CSS y JS [15]".

Bootstrap se basa en un sistema de cuadrícula que permite organizar los elementos de una página web en 12 columnas, las cuales se adaptan al tamaño del explorador como se muestra en la Tabla 1.1, de esta manera cuando se cambia el tamaño de la pantalla los elementos se ordenaran según las reglas definidas por el desarrollador de la página.

Tabla 1.1: Sistema de cuadrícula de *Bootstrap* [16]

	Small <576px	**Medium** ≥576px	**Large** ≥768px	**Extra large** ≥992px	**Extra small** ≥1200px
Ancho máximo del contenedor	Ninguno (auto)	540px	720px	960px	1140px
Prefijo de clase	.col-	.col-sm-	.col-md-	.col-lg-	.col-xl-
# de columnas	12				
Ancho del canal	30px (15px en cada lado de una columna)				
Anidable	Sí				
Ordenamiento de columna	Sí				

Bootstrap contiene varias clases con declaraciones para elementos que requieran de características específicas. En el Segmento de Código 1.3 se puede observar un ejemplo, disponible en [17], de una página web simple con las clases de *Bootstrap*; en este ejemplo en específico se utiliza la clase *Jumbotron*, la cual se utiliza para títulos y cabeceras de las páginas web y la clase *container* la cual se utiliza para contener elementos comunes de las páginas web.

```
<html>
  <head>
      <title>Ejemplo</title>
      <!--Se importa los estilos CSS de Bootstrap-->
      <link rel="stylesheet" href="https://bootstrap.com/bootstrap
      /4.1.1/css/bootstrap.min.css">
      <!--Se importa la librería jQuery necesaria para
       bootstrap -->
      <script src="https://ajax.googleapis.com/ajax/libs/jquery
      /3.2.1/jquery.min.js"></script>
      <!--Se importa el código JavaScript de Bootstrap -->
      <script src="https://bootstrap.com/bootstrap/4.1.1/js
      /bootstrap.min.js"></script>
  </head>
  <body>
     <!--División que tendrá la clase bootstrap de ejemplo-->
     <div class="Jumbotron">
     <h1>Ejemplo de uso</h1>
```

Segmento de Código 1.3 (Parte 1 de 2): Ejemplo de uso de *Bootstrap*

```
19            <p>Este párrafo contiene la clase Jumbotron-Bootstrap</p>
20            </div>
21            <!--División que tendrá la clase container-->
22      <div class="container">
23      <p>Texto de ejemplo.</p>
24      <p>Texto de ejemplo.</p>
25      </div></body></html>
```

Segmento de Código 1.3 (Parte 2 de 2): Ejemplo de uso de *Bootstrap*

b. Skeleton

Skeleton es un *framework* similar a *Bootstrap,* pero mucho más simple y con menores características que *Bootstrap*. Según [18] se utiliza *Skeleton* en el desarrollo de proyectos pequeños. Se basa en un sistema de cuadrícula de 12 columnas que al igual que *Bootstrap* permite ordenar los elementos dependiendo del tamaño del navegador.

1.3.5 JavaScript

"JavaScript es un lenguaje ligero e interpretado orientado a objetos con funciones de primera clase, más conocido como el lenguaje de *script* para páginas web [19]". Este lenguaje de programación se utiliza en varios entornos de desarrollo web, como *node.js,* ya que es un lenguaje de programación con características de multiparadigma, que a su vez es dinámico.

JavaScript se basa en el estándar *ECMAScript*, dicho estándar en su versión 5.1 es soportado por todos los exploradores web modernos desde el año 2012, y los exploradores antiguos soportan por lo menos la versión 3, por lo que cualquier navegador puede interpretar JavaScript.

El modo de trabajo de JavaScript se basa en el acceso al documento y sus elementos mediante DOM. En el Segmento de Código 1.4 se observa la manera de acceso a un documento nuevo, la creación de un nuevo elemento y luego el acceso a dicho elemento mediante su identificador.

a. jQuery

"jQuery es solo una librería de JavaScript [20]" que permite utilizar todo el potencial que posee JavaScript de una manera más sencilla mediante el uso de las funciones

específicas de esta librería.

```
<Script> <!--Dentro va el código JavaScript-->
<!--Se crea un nuevo documento-->
  var documento =
  document.implementation.createHTMLDocument("New Document");
    <!--Se crea el elemento p-->
    var p = documento.createElement("p")
    <!--Se asigna un id al elemento p-->
    p.id = "demo";
    <!--Se obtiene el elemento por Id-->
      var elemento = documento.getElementById("demo");
</ Script>
```

Segmento de Código 1.4: Ejemplo de uso de JavaScript

Para poder acceder a dichas funciones se utiliza el símbolo “$” el cual es un alias para jQuery. En el Segmento de Código 1.5 se puede observar una página HTML básica que incluye la librería jQuery y un *script* que permite acceder al evento de *click* en un elemento; una vez lanzado el evento *click* sobre el elemento indicado se muestra un mensaje en el navegador.

```
<html>
  <head><title>Ejemplo</title></head>
  <body>
      <!--Aquí se incluye el archivo js de jQuery-->
      <script src="jquery.js"></script>
      <script>
        $( document ).ready(function() {
        <!--Mediante el selector CSS se cambian las características
          de todos los elementos que posean ese selector-->
        $( "ejemplo de selector" ).click(function( event ) {
           <!--Se mostrará una alerta con un mensaje-->
            alert( "Gracias por la visita!" );
          }); });
      </script></body></html>
```

Segmento de Código 1.5: Ejemplo de *jQuery*

jQuery y CSS son compatibles entre sí, por lo que todos los selectores de CSS se pueden aplicar en jQuery para acceder a los elementos que los posean. Mediante la sentencia que se observa en el Segmento de Código 1.6 se puede acceder a todas las

características del elemento que contenga el selector, por lo que se podrá modificarlo de una manera simple; esto es útil al momento de querer modificar un grupo de elementos de manera masiva en vez de tener que modificar cada elemento individualmente.

```
1                                    $("SelectorCSS")
```

Segmento de Código 1.6: Ejemplo de selector de *CSS*

a.1. jQuery UI

"jQuery UI es un conjunto de interacciones, efectos, *widgets* y temas de la interfaz de usuario creados sobre la biblioteca JavaScript de jQuery [21]", que permite a los desarrolladores de aplicaciones web añadir características que mejoren la experiencia del usuario.

Al utilizar dichas aplicaciones que sin utilizar jQuery UI serían mucho más complicadas de lograr en JavaScript, en la Figura 1.4 se observa un ejemplo de un campo de texto con *autocomplete* el cual permite realizar búsquedas rápidas desde una lista de lenguajes de programación.

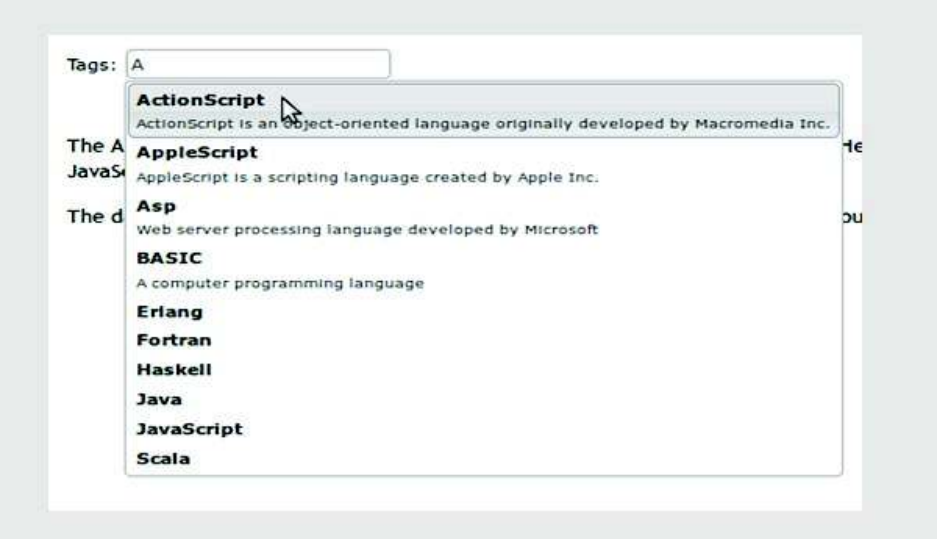

Figura 1.4: Ejemplo de funcionamiento de jQuery UI [21]

1.3.6 ASP.NET

Según [22] Microsoft *Active Server Pages*, denominadas ASP, son páginas que corresponden a un ambiente de codificación que actúa en el lado del servidor y sirven para crear y ejecutar aplicaciones de servidor web. ASP combina documentos HTML,

lenguajes de *script* y componentes COM para su funcionamiento.

"ASP.NET es un *framework* web gratuito que permite construir sitios web por medio de HTML, CSS y JavaScript. También se puede crear APIs web y utilizar tecnologías de tiempo real como Web Sockets [23]". ASP.NET ofrece tres tipos de *frameworks* los cuales son: Formularios Web, *ASP.NET* Modelo Vista Controlador y *ASP.NET* Páginas Web. Cada *framework* apunta al desarrollo de diferentes tipos de aplicaciones, que posean diferentes características.

En el desarrollo del presente trabajo de titulación se utiliza ASP.NET *Web Forms* el cual permite construir sitios dinámicos de una manera rápida, incluyendo controles que encapsulan el marcado HTML.

1.3.7 C#

Como se define en [24], C# es un lenguaje orientado a objetos que se desarrolló por la empresa Microsoft y que se basa en el lenguaje C. El diseño contemporáneo de software se basa en componentes de paquetes de funcionalidad de la forma autocontenidos y autodescriptivos.

Otra característica es el manejo de excepciones para la detección y recuperación de errores que puedan surgir en la aplicación. El diseño *type-safe* de C# obliga a que las variables antes de leerlas sean inicializadas, también a que no se pueda indexar arreglos más allá de sus límites o que no se pueda realizar conversiones sin revisarlas previamente. En el Segmento de Código 1.7 se puede visualizar un ejemplo de código C#, el cual realiza una impresión en consola del mensaje *Hola mundo!*.

```
1     using System; // Se importa System
2     // Inicio de la clase Ejemplo
3     class Ejemplo{
4     //Método principal de la clase
5           static void Main (){
6       //Código para imprimir en consola
7           Console.WriteLine("Hola mundo!");
8           }}
```

Segmento de Código 1.7: Ejemplo de código C#

1.3.8 Bases de Datos

En el ámbito del software “una base de datos es una entidad en la cual se pueden almacenar datos de manera estructurada, con la menor redundancia posible [25]”. Existen varios *DataBase Management System*, cuyas siglas son DBMS, que permiten administrar bases de datos usando un conjunto de servicios.

En el presente trabajo de titulación se utiliza el DBMS SQL Server, perteneciente a Microsoft el cual posee varios tipos de distribuciones descritas en [26]. La distribución denominada SQL Express permite su uso gratuito con propósitos de aprendizaje y construcción de aplicaciones de escritorio pequeñas, es por ello que posee limitaciones en sus características.

a. SQL Server

SQL Server es un *Operational Database Management System*, cuyas siglas son ODBMS, desarrollado por Microsoft y que posee varias distribuciones del tipo comercial y gratuita especificadas en [26].

Como se menciona en [27] a partir de la versión SQL Server 2005 se incorpora el componente *Common Language Runtime* de Microsoft .NET *Framework* el cual posee varias características como procedimientos almacenados agregados definidos por el usuario, entre otros. Como se menciona en [27] los lenguajes incluidos con SQL Server son:

- Transact-SQL
- Multidimensional Expressions
- Data Mining Extensions

Se define en [28] que todas las aplicaciones que se comunican con una instancia de SQL Server lo hacen mediante el envío de declaraciones *Transact-SQL* al servidor, esto sin importar cuál es la interfaz que utiliza el usuario, por lo tanto, *Transact-SQL* es esencial para el uso de SQL Server.

1.3.9 JSON

Según [29] la *Notación de Objetos de* JavaScript, denominado JSON, es un formato ligero para realizar el intercambio de datos. Esta notación es simple de leer y escribir para las personas, e igual de simple para las computadoras interpretarlo y generarlo.

Se basa en un subconjunto de la tercera edición del estándar *ECMA-262* de JavaScript y su principal característica es su independencia de cualquier lenguaje que se use y se compone de dos estructuras:

- Un conjunto de pares de nombre/dato
- Una lista ordenada de datos

Al ser universal, todos los lenguajes de programación lo soportan, por lo que el intercambio de datos puede realizase sin problemas, entre cualquiera de los lenguajes mencionados. En [29] se especifica que un objeto es una colección desordenada de pares *(nombre/valor)*, el cual inicia con una llave de apertura y finaliza con una llave de cierre, dentro de las cuales se ingresa los pares (*nombre:valor)* separados por coma. En el Segmento de Código 1.8 se puede observar un ejemplo de un objeto JSON.

```
1                <!--Ejemplo de datos JSON nombre/valor-->
2                  {nombre1: valor1, nombre2: valor2}
```

Segmento de Código 1.8: Ejemplo de un objeto JSON

1.3.10 Cloud Computing

Como se define en [30] *cloud computing* corresponde a la entrega de potencia informática bajo demanda, la cual puede incluir bases de datos, almacenamiento de aplicaciones y cualquier otro recurso de tecnologías de la información por medio de Internet.

Existen varios servicios, como almacenamiento de datos o capacidad de procesamiento, ofrecidos por distintas empresas, entre las cuales se encuentra *Amazon Web Services*, cuyas siglas son AWS, dichos servicios se pueden clasificar en tres diferentes tipos: *Software Como Servicio* o SaaS, *Plataforma Como Servicio* o PaaS e *Infraestructura Como Servicio* o IaaS.

a. AWS

En [31] se define a AWS como una plataforma de servicios de nube que proporciona infraestructura bajo demanda con la modalidad de pago por consumo. Posee más de 50 servicios enfocados en diferentes áreas de las tecnologías de la información como por ejemplo almacenamiento de datos, bases de datos, almacenamiento de aplicaciones, entre otros.

En el presente trabajo de titulación se hace uso de tres servicios de AWS en específico: EC2, S3 y RDS, además de otros servicios que brindan soporte a los servicios principales como *ElasticBeanstalk*.

a.1. EC2

Elastic Compute Cloud, cuyas siglas son EC2, es un "entorno virtual de cómputo que permite utilizar interfaces de servicios web para lanzar instancias con distintos sistemas operativos [1]", trabaja con imágenes de máquina de Amazon denominadas *AMI*.

Existen varios tipos de sistemas operativos que ofrece EC2 entre los cuales se encuentra Windows Server 2016 utilizado en el presente trabajo de titulación. Para cada instancia se posee características de *hardware* diferentes y las características mínimas que puede poseer una instancia son una vCPU, una memoria de 0.5 GB y almacenamiento *EBS* de 30 GB denominada t2.nano.

Una vez creada una instancia, ésta funciona como un servidor cualquiera con las características de *hardware* especificadas, si es necesario aumentar las características de *hardware* se lo puede realizar sin ningún problema ya que es un servicio escalable hacia arriba o hacia abajo.

a.2. S3

"Amazon S3 es un servicio de almacenamiento de objetos creado para almacenar y recuperar cualquier volumen de datos desde cualquier ubicación como: sitios web y aplicaciones móviles, aplicaciones corporativas y datos de sensores o dispositivos IoT [2]".

S3 permite almacenar los objetos en contenedores denominados *Buckets* desde los cuales se puede acceder mediante *FTP* o directamente desde la consola de administración de *Amazon Web Services*. S3 ofrece una durabilidad de casi el 100 % y es utilizado por varias aplicaciones a nivel mundial, una de las más conocidas es Netflix[1]. La seguridad que ofrece S3 es la posibilidad de cifrar los datos en los procesos de transporte y almacenamiento, mediante el algoritmo *AES-256* o con una *KMS*.

[1] Netflix, Inc. es una empresa comercial estadounidense de entretenimiento que proporciona mediante tarifa plana mensual un *streaming* de contenido multimedia bajo demanda por Internet.

a.3. RDS

El *Servicio de Base de Datos Relacional*, denominado RDS, es un servicio que permite configurar, utilizar y escalar una base de datos relacional. “Proporciona seis motores de bases de datos familiares entre los que elegir, incluidos Amazon Aurora, PostgreSQL, MySQL, MariaDB, Oracle y Microsoft SQL Server [3]” y la administración de la base de datos se la realiza mediante la consola de AWS.

2. METODOLOGÍA

En este capítulo se detalla todos los procesos del diseño, desarrollo e implementación de los módulos que conforman el prototipo para la gestión de reservas en línea y registro de entrada-salida de clientes de un hotel.

2.1 INTRODUCCIÓN

La etapa de diseño del sistema comprende cinco procesos:

1. Análisis de requerimientos
2. Creación de los diagramas de casos de uso
3. Creación del diagrama relacional
4. Creación del diagrama de clases
5. Diseño visual del sistema

La etapa de desarrollo del sistema comprende dos procesos:

1. Creación de una base de datos local
2. Codificación del sistema

La etapa de implementación del sistema comprende dos procesos:

1. Creación de una instancia SQL Server 2012 en RDS
2. Creación de un servidor web mediante *ElasticBeanstalk*

2.2 ANÁLISIS DE REQUERIMIENTOS

Para la realización del análisis de requerimientos se consultó [32] y se realizó una entrevista con la Licenciada Irina Galarraga, gerente general del hotel Playa Cristal, dicha entrevista se encuentra disponible en el ANEXO I y se la realizó de manera presencial, para ello se escribieron trece preguntas enfocadas en cinco aspectos específicos del hotel:

1. La estructura física que posee el hotel
2. Proceso de reservas de habitaciones
3. Proceso de registro de los clientes
4. Costo de las habitaciones
5. Cantidad de personas que intervienen en los procesos de reservas y registro

De la entrevista realizada se obtienen los datos que corresponden a las habitaciones, reservas de habitaciones, empleados del hotel y parte de los datos necesarios para el registro de entrada-salida de los clientes ya que los demás datos se encuentran definidos en [32].

De las habitaciones se describe su capacidad y sus características. En cuanto a las reservas se describe qué datos son necesarios para la realización de estas, de los empleados del hotel se describe que datos pueden manejar y de los registros se describe que datos se solicitan.

Primero se tomó en cuenta que el sistema debía realizar dos procesos principales:

- Reservas de habitaciones en línea
- Registro de entrada-salida de clientes

Luego se tomó en cuenta que existen dos tipos de personas que iban a utilizar el sistema:

- Clientes
- Usuarios

Se define como un cliente a aquella persona que va a hacer uso de los servicios que ofrece un hotel y se define como un usuario a la persona que se encuentra encargada de realizar una o varias tareas como empleado del hotel, por lo que se crea el proceso de *control de acceso*.

Debido a que el sistema almacenara datos personales de clientes, empleados y habitaciones es necesario que se pueda visualizar y modificar dichos datos, por lo que se decide la creación de un proceso de administración del sistema. Luego se tiene la necesidad de poder mostrar la información de las dos tareas principales que posee el sistema, por lo que *generar reportes* conforma el último proceso que se realizará.

De esta manera se definen los procesos principales que debe poder realizar el sistema, dichos procesos principales conforman los *módulos*.

El primer proceso que se debe tomar en cuenta es el de control de acceso al sistema por lo que es necesario conocer quienes tendrían acceso a cada uno de los módulos; en la Tabla 2.1 se puede observar una descripción de los procesos que realiza cada módulo y se describirán a detalle.

Tabla 2.1: Módulos que comprenden el sistema

Módulo	Descripción
Administración	Permite realizar la lectura, creación, modificación o eliminación de los datos correspondientes a los clientes, usuarios o habitaciones.
Reservas	Comprende todos los procesos desde la creación de una posible reserva, la confirmación de una reserva y su actualización o eliminación.
Registros	Comprende todos los procesos correspondientes al registro de entrada-salida de los clientes del hotel, el registro de los huéspedes y la creación de un comprobante de pago.
Reportes	Permite la generación de reportes de las reservas confirmadas y los registros de entrada-salida del hotel en un periodo determinado.

2.2.1 Módulo de reserva de habitaciones en línea

Se define a una reserva como la acción de solicitar el alquiler de una habitación, en un período de tiempo en días, con un precio establecido el momento de realizar la reserva, impidiendo de esta manera que otra persona alquile dicha habitación en ese período de tiempo.

Para realizar una reserva son necesarios ciertos datos específicos del cliente los cuales se definieron en la entrevista.

Datos obligatorios del cliente para realizar una reserva:

- Nombre
- Correo
- Teléfono
- Tipo de habitación deseada
- Fecha de ingreso deseada
- Fecha de salida deseada
- Cantidad de personas adultas
- Cantidad de personas menores a 12 años

Datos Opcionales del cliente para realizar una reserva:

- Teléfono secundario

- Comentario acerca de las necesidades del cliente

También se definió que la cantidad máxima de habitaciones, que puede reservar en línea un cliente, es de cuatro y si se desea reservar más habitaciones se lo realizará mediante contacto directo con una persona encargada del hotel.

En la Figura 2.1 se puede observar el diagrama de actividad del módulo de reservas, en donde se especifica los procesos que se debe seguir para poder realizar una posible reserva, confirmar una posible reserva y, de ser el caso, modificar o eliminar una reserva.

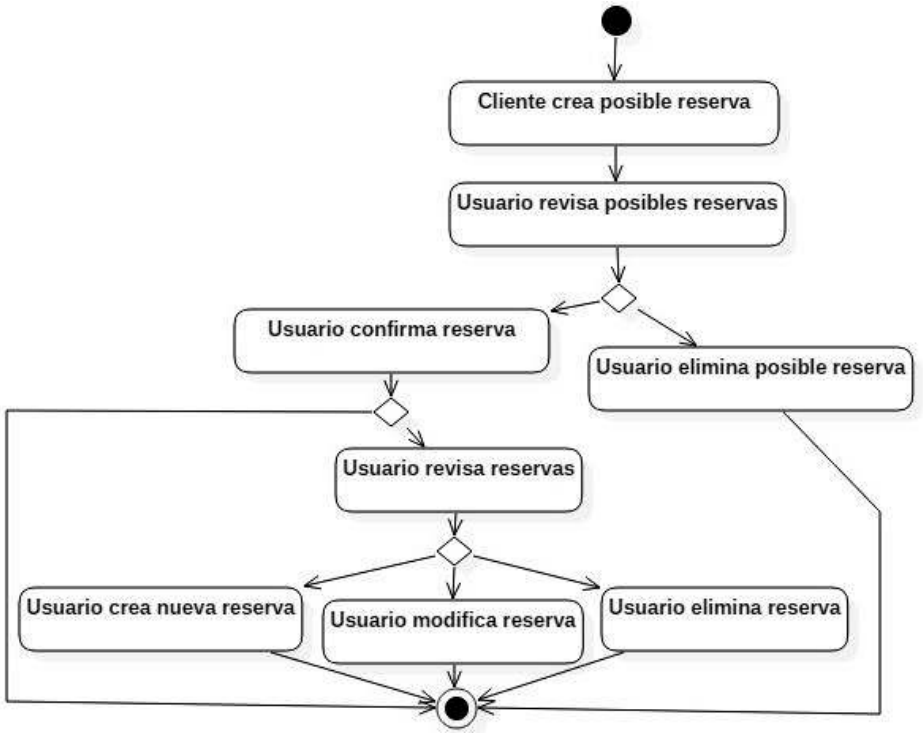

Figura 2.1: Diagrama de actividad del módulo de reservas

Las reservas las puede realizar cualquier persona por lo que el formulario de reservas del cliente no posee un control de acceso. Una vez realizada la reserva por parte del cliente es necesario que una persona encargada del hotel pueda visualizar la información del cliente y confirmar las reservas realizadas; debido a que dichas reservas contienen información personal de los clientes estos formularios si poseen un control de acceso.

Una vez conocidos los pasos y los datos necesarios para realizar una reserva, se definió que, para el proceso de reservas de habitaciones por parte de los clientes, se tendría el formulario *Reservas.aspx*. Hay que tomar en cuenta que, al ser de libre acceso, pueden

generarse varias reservas falsas o reservas que nunca se confirmen, por lo tanto, una reserva realizada en este formulario se considera una *posible reserva*.

Para los demás formularios del módulo de reservas se posee un control de acceso. Se decidió la creación del formulario *Confirmar_Reservas.aspx,* en el cual una *posible reserva* pasa a ser una *reserva confirmada,* o simplemente reserva y se informa al cliente de la confirmación de la reserva mediante el envío de un correo electrónico.

Una vez creadas las reservas, se debe tener la posibilidad de crear una nueva reserva, realizar cambios a una reserva existente, o incluso cancelar una reserva, por lo que se decide la creación del formulario *Modificar_Reservas.aspx* el cual realizaría las siguientes tareas:

- Crear una reserva nueva
- Visualizar datos de una reserva existente
- Modificar los datos de una reserva existente
- Eliminar una reserva existente

2.2.2 Módulo de registro de entrada-salida de clientes

En este módulo se realiza el proceso de registro de entrada-salida de los clientes del hotel por parte de los usuarios. El diagrama de actividad correspondiente al módulo de registros se lo puede observar en la Figura 2.2. En el diagrama se define los pasos para realizar el proceso de registro de entrada de un cliente, en caso de que sea necesario modificar un registro de entrada, realizar el registro de salida y finalmente crear un comprobante de pago con los datos del registro de entrada-salida, para su posterior envío al correo electrónico del cliente.

En [32], Artículo 6.- “Derechos y obligaciones de los establecimientos de alojamiento turístico”, se señala lo siguiente respecto al registro de los clientes en un hotel:

- Art 6.- literal s) Exigir información al huésped, incluyendo la presentación de documentos de identidad de todas las personas que ingresen al establecimiento;
- Art 6.- literal v) Llevar un registro diario y proporcionar a la Autoridad Nacional de Turismo y a las autoridades que así lo requieran, información sobre el perfil del huésped donde se incluya al menos nombre, edad, nacionalidad, género, número de identificación, tiempo de estadía y otros que se determinen;

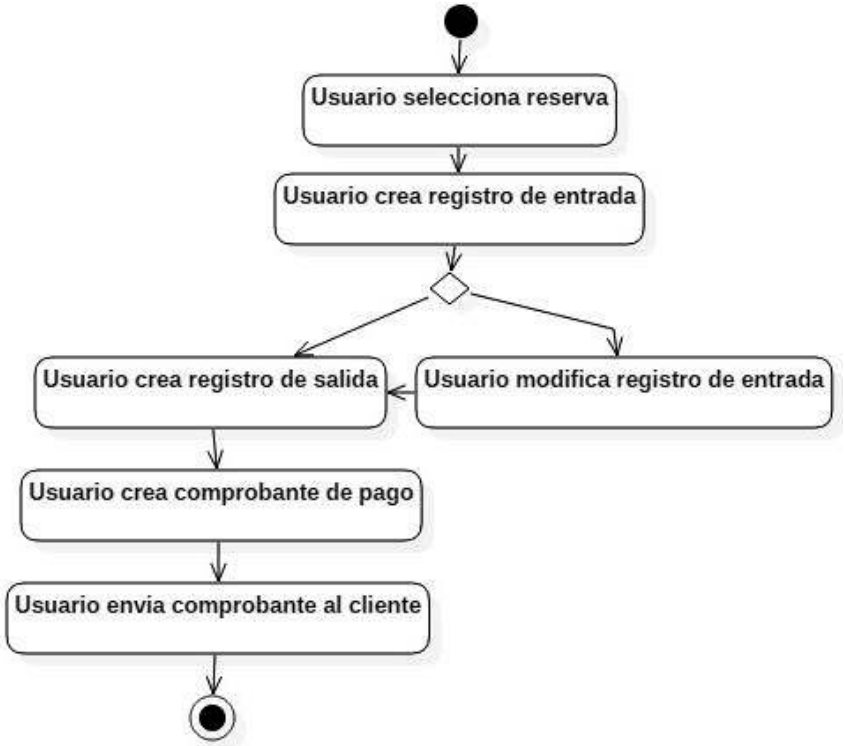

Figura 2.2: Diagrama de actividad del módulo de registros

Por lo tanto, como se especifica en el literal v, se determinaron otros datos a solicitar sobre el perfil del huésped del hotel, dichos datos se los definió en la entrevista que se encuentra en el ANEXO I. Se muestra a continuación una lista de los datos necesarios para la realización de un registro de huéspedes.

- Nombre
- Número de identificación
- Edad
- Nacionalidad
- Género
- Correo electrónico
- Número de teléfono 1 y un número de teléfono opcional
- Dirección o ubicación

Luego de definir qué datos son los necesarios para la realización de un registro, se decidió separar las tareas de registro de entrada-salida en cuatro diferentes formularios. El primer formulario `Registro_Entrada.aspx` permite realizar el registro de los huéspedes de una habitación y asignar el registro a un cliente. El segundo formulario *Registro_Salida.aspx* permite registrar la salida del cliente y calcular el tiempo de estadía en el hotel.

El tercer formulario `Modificar_Registro.aspx` permite modificar los registros de entrada, así como los huéspedes que se ingresaron en dicho registro.

Y el cuarto formulario `Comprobante.aspx` permite realizar la creación de un comprobante electrónico con los datos de un registro de entrada-salida, adjuntar elementos extras al comprobante y realizar el envío del comprobante al correo electrónico del cliente. Para la tarea de adjuntar elementos extras a un comprobante se definieron tres datos necesarios acerca del elemento extra:

- Descripción
- Cantidad
- Costo unitario

2.2.3 Módulo de administración del sistema

En el módulo de administración se podrá leer, crear y actualizar diferentes datos que se encuentren almacenados en la base de datos. Debido a que este módulo permitirá acceder a datos de todo el sistema, solo deberá permitir el acceso por parte de usuarios del hotel que estén encargados de la administración.

Luego de definir qué tareas específicas realizan cada uno de los formularios, se define que el módulo de administración podrá modificar datos correspondientes a los usuarios, las habitaciones, los clientes y el usuario que se encuentra conectado en ese momento. Cada uno de estos conjuntos de datos posee un formulario en específico para su administración: `Admin_Usuarios.aspx`, `Admin_Habitaciones.aspx` y `Admin_Clientes.aspx`. Para los datos del usuario que se encuentra conectado el formulario es `Mi_Perfil.aspx`.

a. Administración de usuarios

Los usuarios se definen como las personas que poseen acceso al sistema, excluyendo el formulario *Reservas.aspx* al cual puede ingresar cualquier persona.

El diagrama de actividad del formulario de administración de usuarios se encuentra en la Figura 2.3, en donde se puede observar el proceso a seguir para crear o modificar los datos de un usuario del sistema.

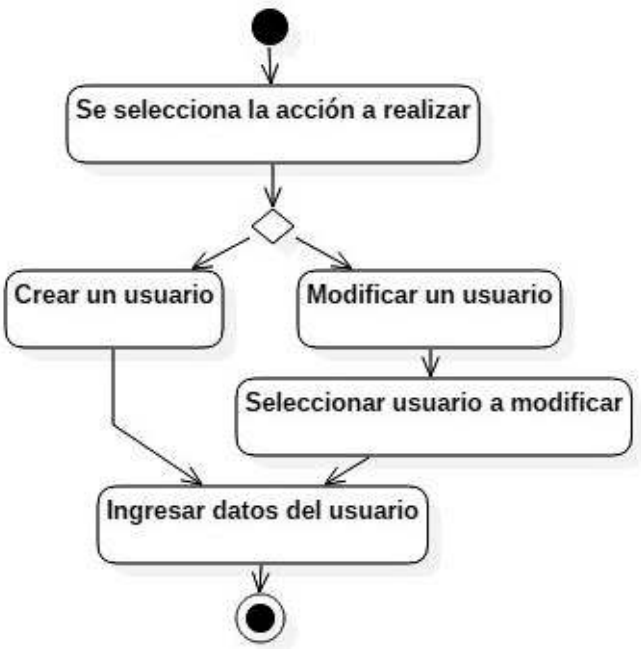

Figura 2.3: Diagrama de actividad de administración de usuarios

Se definieron tres tipos de usuarios, cada uno con acceso a diferentes formularios, en la Tabla 2.2 se observa los tres usuarios con sus respectivos accesos, en donde la X determina que el usuario posee acceso al formulario correspondiente.

Tabla 2.2: Control de acceso a los formularios del sistema

Formulario	Administrador	Supervisor	Normal
Admin_Usuarios	X		
Admin_Habitaciones	X		
Admin_Clientes	X	X	X
Mi_Perfil	X	X	X
Confirmar_Reservas	X	X	X
Modificar_Reservas	X	X	X
Registro_Entrada	X	X	X
Registro_Salida	X	X	X
Modificar_Registro	X		
Comprobante	X	X	X
Reporte_Reservas	X	X	
Reporte_Registros	X	X	

Como se puede observar en la Tabla 2.2, el formulario `Admin_Usuarios.aspx` solo puede ser accedido por los administradores, ya que en dicho formulario se puede leer, crear y modificar cualquier tipo de usuario, además se añadió el formulario emergente `Usuarios_Conectados.aspx`, el cual permite visualizar a los usuarios que se encuentran conectados y desconectarlos si se desea, para poder prevenir acceso no deseado en caso de dejar abierta una sesión.

b. Administración de habitaciones

Una parte esencial que se debe almacenar en la base de datos del sistema es la correspondiente a los datos de las habitaciones existentes, ya que mediante estos datos se conocerá las características de:

- Cantidad total de habitaciones
- Tipo de habitaciones
- Capacidad de las habitaciones
- Disponibilidad de la habitación

Por lo tanto, en el formulario `Admin_Habitaciones.aspx` se podrá leer, modificar y crear estos datos, como se puede observar en el diagrama de actividad de la Figura 2.4.

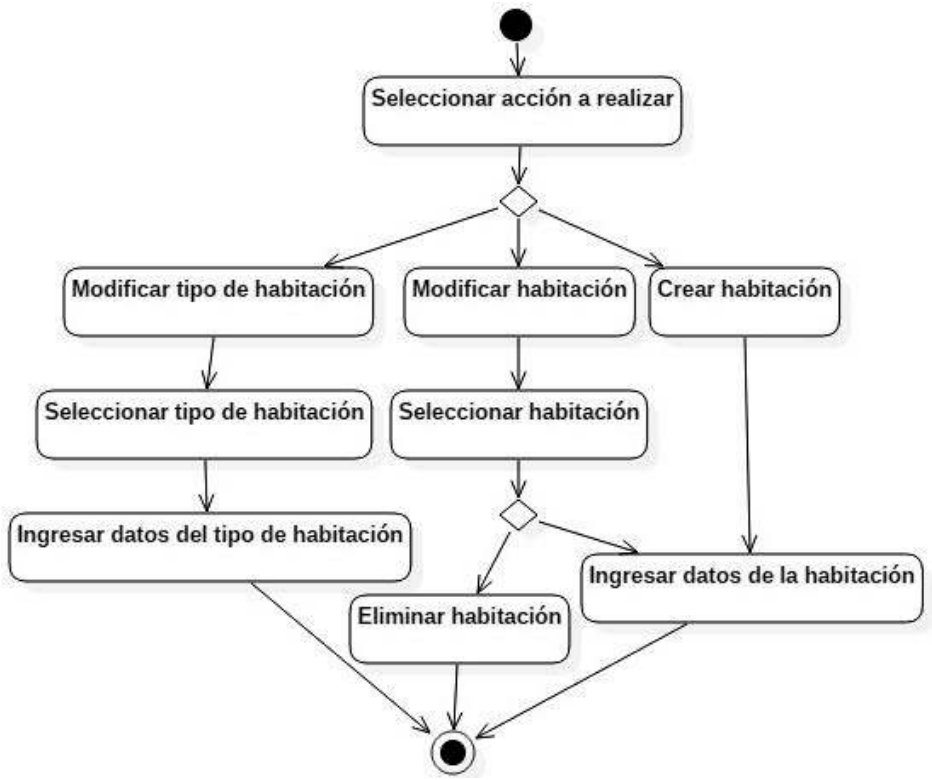

Figura 2.4: Diagrama de actividad de administración de habitaciones

Para conocer las características que se desea administrar de las habitaciones, se realiza una descripción de la estructura que posee el hotel Playa Cristal. El hotel tiene dos tipos de habitaciones del tipo *suite* minimalista, la *suite* familiar y la *suite* matrimonial.

Existen veintiocho habitaciones en total, de las cuales cuatro son del tipo *suite* matrimonial, con una capacidad para dos personas, y doce del tipo *suite* familiar. Las *suites* familiares se dividen en dos habitaciones independientes, cada una con un baño, la *suite* familiar alta, que posee capacidad para dos personas y la *suite* familiar baja con capacidad para tres personas.

Dentro de cada una de las habitaciones existe televisión por cable, aire acondicionado e Internet y una o dos camas dependiendo del tipo de habitación, las camas *Queens*, o matrimoniales, cuyo tamaño es de dos plazas y media y las camas individuales cuyo tamaño es de una plaza y media.

En el hotel actualmente se identifica a cada una de las habitaciones mediante un código específico, para las *suites* matrimoniales se utiliza la letra “M” seguida del número de habitación desde el 1 hasta el 4, por ejemplo: M-01 identifica a la habitación matrimonial uno.

Para las *suites* familiares se utiliza la letra “A” para identificar a la parte alta de la habitación y la letra “B” para identificar a la parte baja de la habitación seguido del número de habitación desde el 1 hasta el 12, por ejemplo, A-05 identifica a la *suite* familiar alta cinco.

Tomando en cuenta la estructura actual que posee el hotel se decidió almacenar los siguientes datos sobre las habitaciones:

- Tipo de habitación
- Código de la habitación
- Cantidad de camas matrimoniales
- Cantidad de camas individuales
- Cantidad de baños
- Extras que posee la habitación

c. Administración de clientes

En la Figura 2.5 se puede observar el diagrama de actividad correspondiente a la

administración de los clientes, en donde se describen los procesos para la creación o modificación de los datos correspondientes a clientes del hotel.

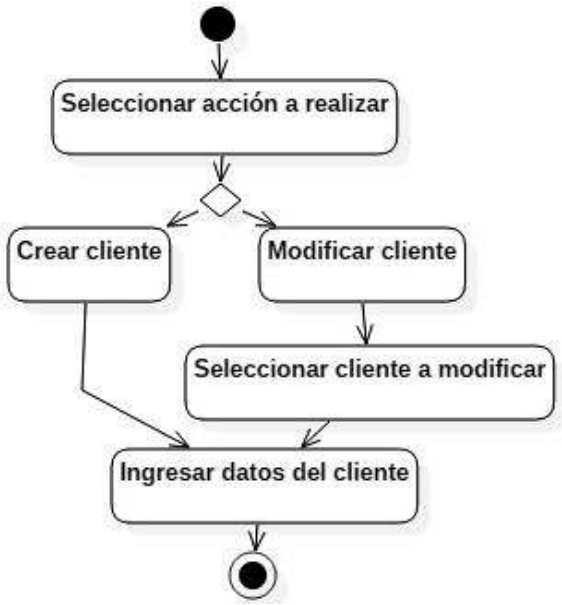

Figura 2.5: Diagrama de actividad de administración de clientes

La información correspondiente a los clientes se almacena en la base de datos del sistema. Para ello se tiene en cuenta los datos definidos en el módulo de reservas y el módulo de registro que son los siguientes:

- Nombre
- Número de identificación
- Correo electrónico
- Teléfono 1
- Teléfono 2
- Dirección
- Edad
- Nacionalidad
- Género

d. Administración de *Mi Perfil*

Los datos que se administran en el formulario *Mi_Perfil.aspx* son los mismos que los de *Admin_Usuarios.aspx* correspondientes al usuario que se encuentra conectado en ese momento, con la diferencia que no se podrá cambiar el tipo de usuario. En la Figura 2.6 se puede observar el diagrama de actividad correspondiente a la administración de *Mi Perfil.*

Figura 2.6: Diagrama de actividad de administración de *Mi Perfil*

2.2.4 Módulo de generación de reportes

En este módulo se presentará un informe dentro de un periodo determinado de los datos almacenados en los dos procesos principales que posee el sistema, el diagrama de actividad correspondiente se lo puede observar en la Figura 2.7, en el cual se describen los procesos para la visualización del reporte deseado ya sea de reservas o de registros.

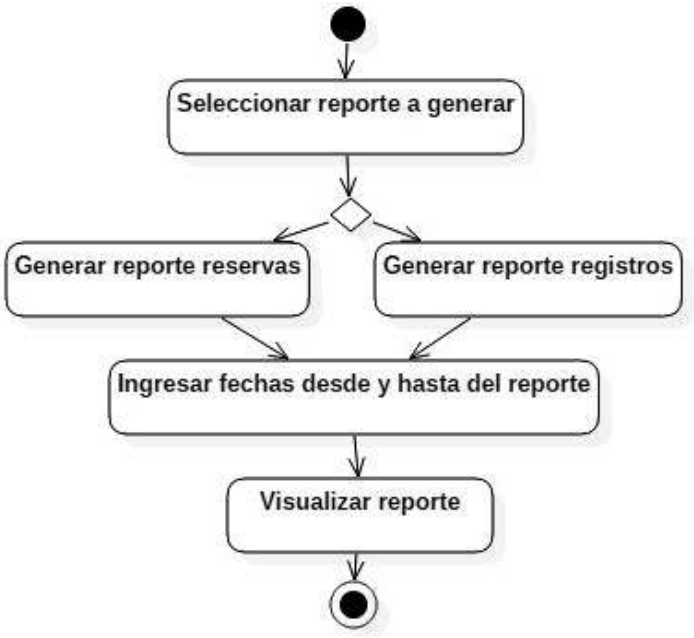

Figura 2.7: Diagrama de actividad del módulo de reportes

Para la generación de los reportes se debe definir las fechas desde y hasta, entre las cuales se muestran los datos de reserva de habitaciones y de registros de entrada-salida de clientes. A continuación, se especifican qué datos son los necesarios para

mostrar en dichos reportes; para el reporte de reserva de habitaciones los datos que se necesitan son:

- Fechas de ingreso y salida de la reserva
- Cantidad de días que se hospedará
- La habitación reservada
- Número de identificación de la persona que reserva
- Nombre de la persona que reserva
- Cantidad de adultos que se hospedarán
- Cantidad de menores a 12 años que se hospedarán

Para el reporte de registros de entrada-salida se necesitan los siguientes datos:

- Fechas de ingreso y salida del hotel
- Cantidad de días hospedados
- Habitación en la cual se hospedó el cliente
- Número de identificación del cliente
- Nombre del cliente

2.3 DIAGRAMAS DE CASOS DE USO

Una vez que se definieron los formularios que conforman cada uno de los módulos, al igual que sus funciones y con qué datos trabajarían, se procedió a la creación de los diagramas de casos de uso.

2.3.1 Reserva de habitaciones (cliente)

El diagrama de casos de uso correspondiente a la reserva de habitaciones por parte del cliente se encuentra en la Figura 2.8, en dicho diagrama se pueden observar las tres acciones que puede realizar un cliente, entre las cuales la principal es la reserva de una habitación y si se desea se puede realizar un comentario incluido en la reserva.

2.3.2 Administración

En la Figura 2.9 se puede observar los tres tipos de actores correspondientes a los tres tipos de usuarios que posee el sistema, cada uno puede realizar diferentes acciones en el módulo de administración de acuerdo con lo definido en la etapa de análisis de requerimientos.

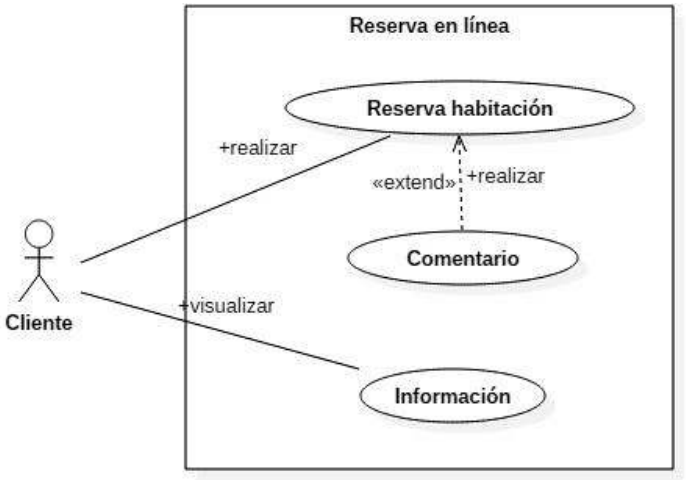

Figura 2.8: Diagrama de casos de uso de reserva de habitaciones por parte del cliente

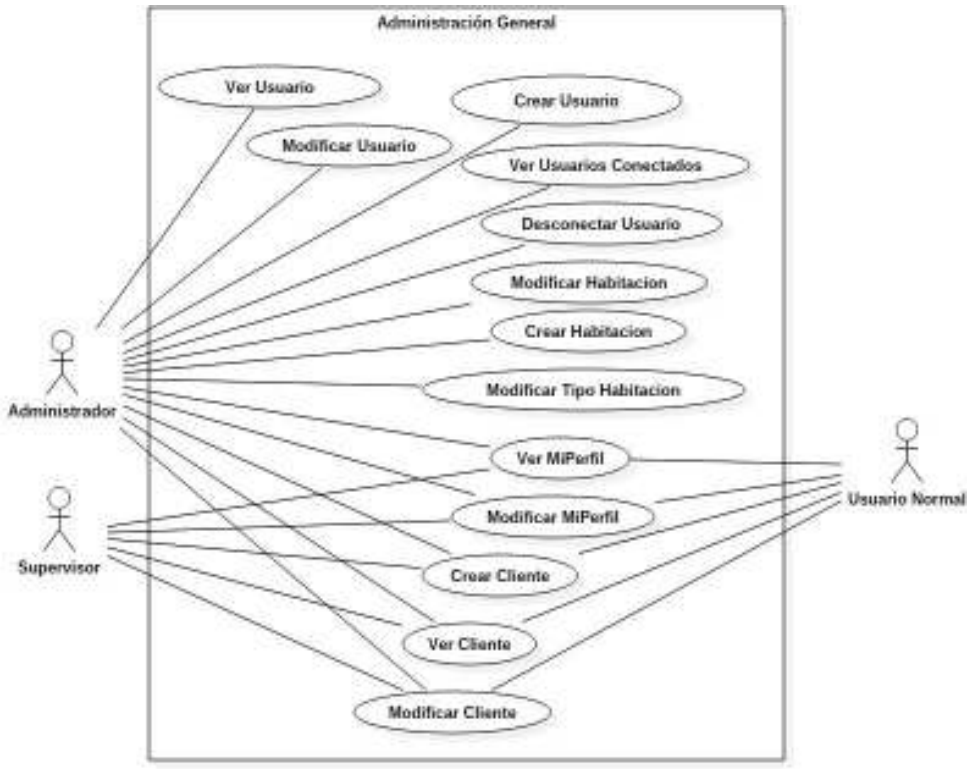

Figura 2.9: Diagrama de casos de uso de administración

El administrador podrá acceder al formulario `Admin_Usuarios.aspx` para visualizar los datos de los usuarios, modificar los datos de un usuario, crear un nuevo usuario, visualizar los usuarios conectados y si desea desconectar un usuario en específico. Para las dos últimas acciones se decidió la creación de otro formulario emergente denominado `Usuarios_Conectados.aspx`.

El administrador también podrá acceder al formulario de `Admin_Habitacion.aspx` para modificar una habitación, crear una habitación y modificar el tipo de habitación. Para los procesos de crear una habitación y modificar el tipo de habitación se decidió la creación de un formulario emergente para cada proceso, denominados `Nueva_Habitacion.aspx` y `Modificar_Tipo_H.aspx`.

Los siguientes formularios poseen acceso por parte de los tres actores. El formulario denominado `Mi_Perfil.aspx` permite ver datos del usuario conectado actualmente y modificar sus datos y el formulario `Admin_Cliente.aspx` permite visualizar datos de los clientes registrados, crear un nuevo cliente y modificar los datos de un cliente existente.

2.3.3 Reservas

El diagrama de casos de uso correspondiente a las reservas se lo puede observar en la Figura 2.10 en el cual todos los actores pueden realizar las mismas tareas.

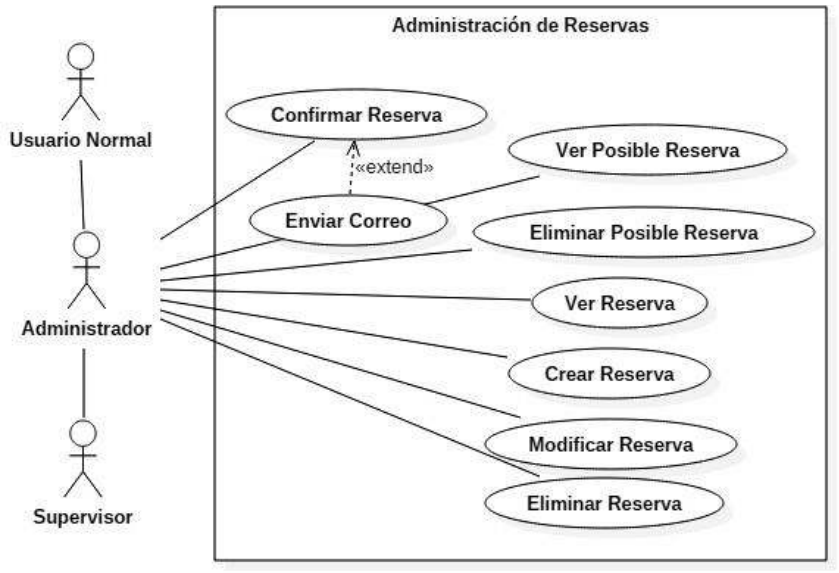

Figura 2.10: Diagrama de casos de uso de reservas

El formulario `Confirmar_Reserva.aspx` permitirá realizar las siguientes tareas de visualizar una posible reserva, eliminar una posible reserva, realizar la confirmación de una reserva y enviar un correo electrónico de al cliente con los datos de la reserva confirmada.

En el formulario `Modificar_Reservas.aspx` se podrá visualizar las reservas confirmadas, modificar una reserva, crear una nueva reserva y eliminar una reserva.

2.3.4 Registros

En el formulario `Registro_Entrada.aspx` se podrá crear un registro de entrada y crear huéspedes pertenecientes a dicho registro, luego, en el formulario `Registro_Salida.aspx`, se podrá crear un registro de salida correspondiente a un registro de entrada.

Para poder crear un comprobante se accederá al formulario `Comprobante.aspx` y además permitirá la creación de extras que se incluirán en el comprobante de pago. Existe la posibilidad de realizar una modificación a un registro de entrada por parte de un administrador del sistema en el formulario `Modificar_Registro.aspx`.

En la Figura 2.11 se describen todas las acciones que puede realizar un usuario y los tres tipos de usuarios pueden realizar dichas acciones.

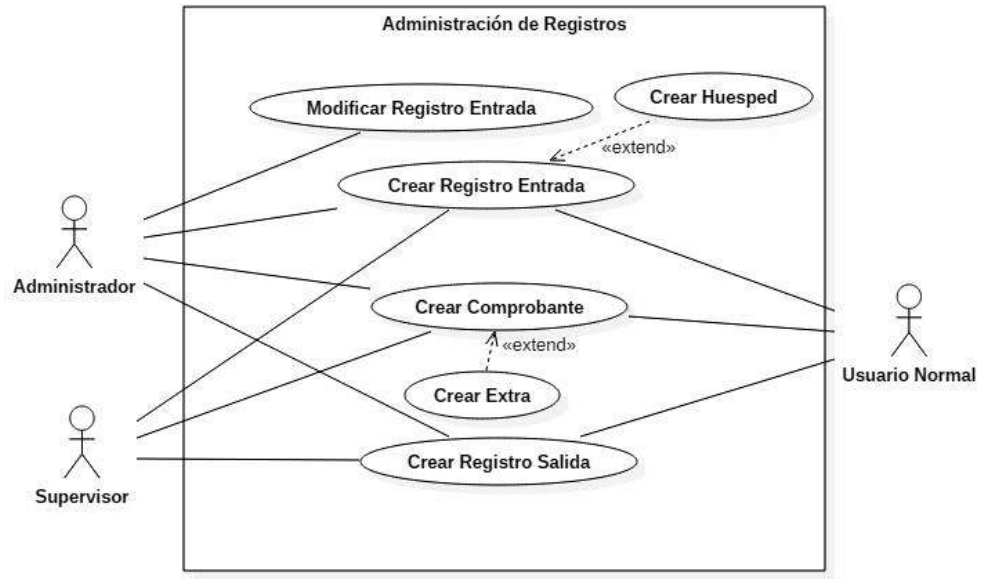

Figura 2.11: Diagrama de casos de uso de registros

2.3.5 Reportes

En la Figura 2.12 se puede observar que solo dos actores intervienen en este módulo, estos actores pueden realizar las acciones de generar un reporte para las reservas y para los registros, limitando los reportes a un período de tiempo. Para el reporte de

reservas se accederá al formulario `Reporte_Reservas.aspx` y para el reporte de los registros se accederá al formulario `Reporte_Registros.aspx`.

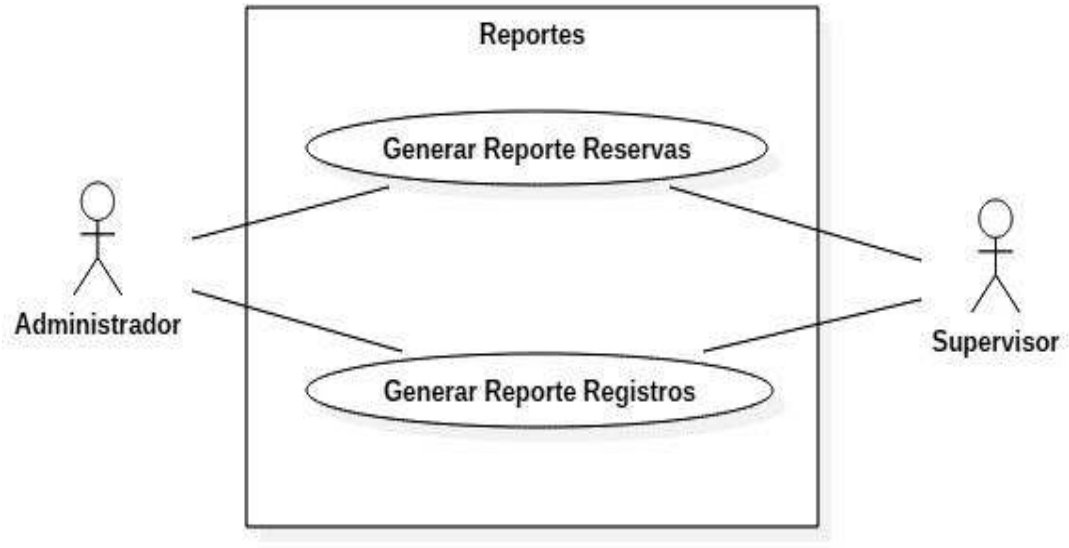

Figura 2.12: Diagrama de casos de uso de reportes

2.4 DIAGRAMA RELACIONAL

En la Figura 2.13 se puede observar el diagrama relacional que se utilizó para la creación de la base de datos del sistema. Se definen once tablas de las cuales la tabla de `Posible_Reserva` no posee relación con ninguna otra, ya que se almacenan datos que no han sido confirmados todavía.

Cada una de las tablas posee un identificador auto-incremental de tipo numérico, que comienza desde el valor de uno, de esta manera se podrá identificar de una manera única a cada elemento de la tabla. A continuación, se describen todas las tablas y los datos que almacenan.

2.4.1 Tabla Usuario

Los datos correspondientes a todos los usuarios se almacenarán en la tabla Usuario. Para almacenar la clave del usuario se analizaron dos opciones:

- Almacenar un *hash* de la clave
- Almacenar la clave mediante un algoritmo de cifrado

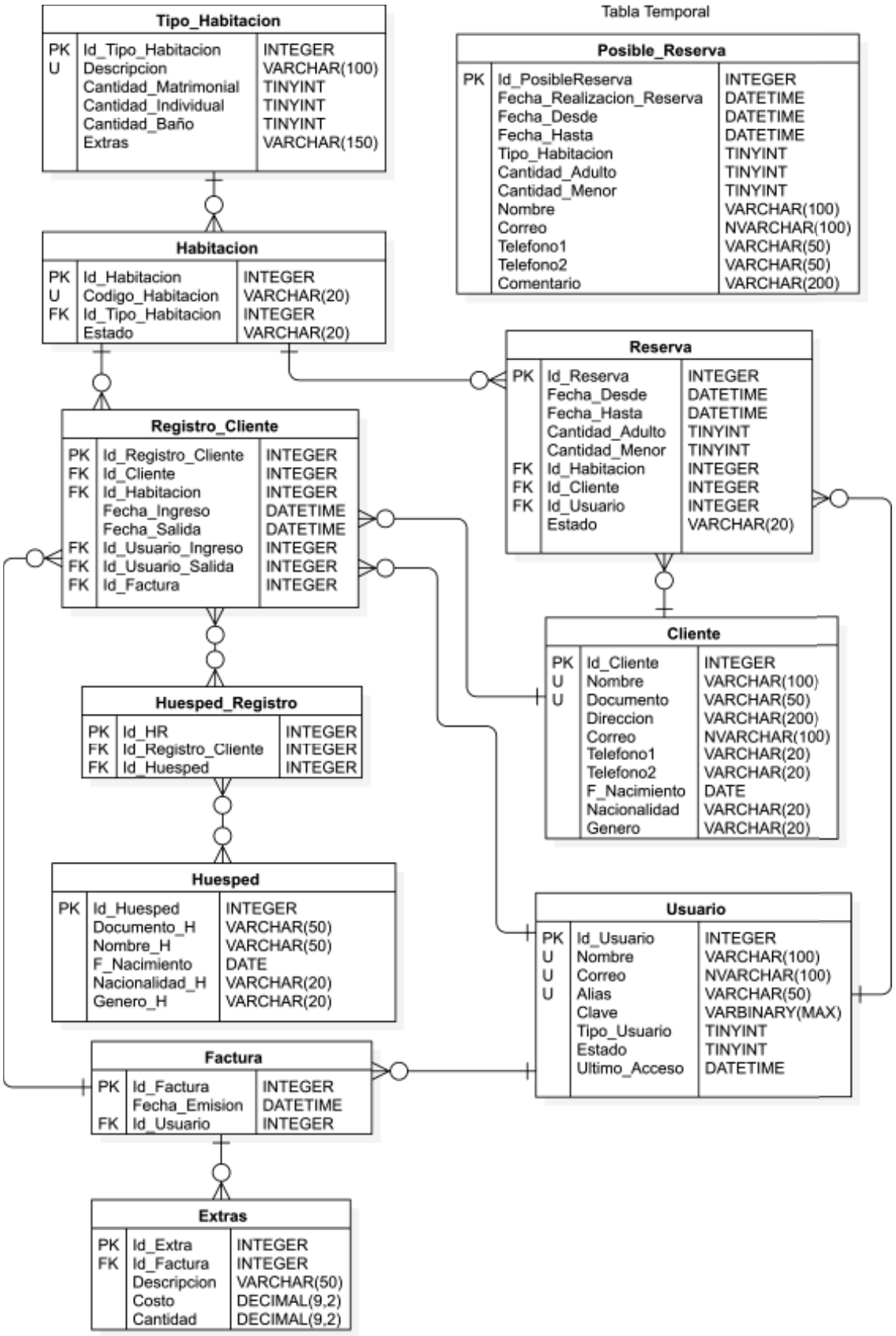

Figura 2.13: Diagrama relacional del sistema

Se decidió el uso del algoritmo de cifrado asimétrico *RSA_2048*, el cuál utiliza una clave pública, ya que de esta manera sería posible recuperar la clave del usuario en caso de pérdida, también se decidió que el presente trabajo de titulación no tomaría en cuenta ni la seguridad del sistema ni el tamaño de la base de datos, por lo que al momento de la autenticación del usuario la clave llega en texto plano al sistema para su comparación, lo cual supone un riesgo de seguridad.

Los datos que se almacenarán se pueden observar en la Figura 2.13. Los datos que permiten la identificación en el sistema son alias y clave y el tipo de usuario permite al sistema identificar si el usuario es Administrador, Supervisor o Usuario Normal.

El campo de estado permite conocer si el usuario se encuentra conectado o desconectado en ese instante y el campo de último acceso almacena la fecha en la cual el usuario ingresó por última vez al sistema.

2.4.2 Tabla Cliente

En esta tabla se almacenará todos los datos definidos en la etapa de análisis de requerimientos correspondientes a un cliente como se observa en la Figura 2.13. El nombre almacena nombre y apellido del cliente en el mismo campo, el documento puede ser la cédula de identidad del cliente o un pasaporte, por lo que se lo almacena a manera de texto y no de número, ya que los pasaportes pueden contener letras y símbolos.

La dirección, correo, telefono1, telefono2, nacionalidad y género almacenan datos personales del cliente y para el almacenamiento de la edad se decidió que era mejor almacenar la fecha de nacimiento del cliente para poder conocer su edad en futuras ocasiones que se hospede en el hotel.

Como se muestra en la Figura 2.13 un cliente puede tener varias reservas de habitaciones y varios registros de entrada-salida a su nombre por lo que decidió prevenir la duplicidad de la información añadiendo un atributo de único al nombre y documento del cliente.

2.4.3 Tabla Tipo_Habitacion

En esta tabla se almacena los datos de los diferentes tipos de habitaciones que posee el hotel, estos datos definen la cantidad de camas, baños y extras de las habitaciones

como se observa en la Figura 2.13. Por el momento en el hotel existen tres tipos de habitaciones definidos en el análisis de requerimientos.

2.4.4 Tabla Habitacion

En la tabla `Habitacion` se ingresan los datos de todas las habitaciones existentes en el hotel, como se observa en la Figura 2.13, con su correspondiente código (el cual debe ser único) y tipo de habitación. Además, se almacena el estado de la habitación, el cual permite conocer si se encuentra activa, en mantenimiento o clausurada.

2.4.5 Tabla Posible_Reserva

La tabla `Posible_Reserva` almacena los datos de una posible reserva realizada por los clientes, de esta tabla se pasa la información a la tabla Reserva y Cliente, una vez se haya confirmado dicha información por parte de una persona encargada del hotel.

Los datos que almacena `Posible_Reserva` se pueden observar en la Figura 2.13, en donde la fecha de realización de la reserva corresponde a la fecha en la cual el cliente solicitó la reserva en la página web, las fechas desde y hasta corresponden a las fechas deseadas de ingreso y salida al hotel.

El tipo de habitación permite especificar qué tipo de habitación desea el cliente, la cantidad de adultos y menores indican cuantos huéspedes va a tener la habitación deseada; esta cantidad depende de la capacidad de la habitación. El nombre, correo, telefono1 y telefono2 son datos proporcionados por el cliente y el comentario permite agregar detalles de la reserva al cliente.

2.4.6 Tabla Reserva

La tabla de reservas es la que almacena los datos de una reserva confirmada por parte de una persona encargada del hotel, los datos que almacena se pueden observar en la Figura 2.13.

Al igual que en la tabla de posible reserva, se crea un registro por cada habitación reservada, en donde la fecha desde indica la fecha de ingreso por parte del cliente, la fecha hasta indica la fecha de salida del cliente del hotel, la cantidad de adultos y menores indica cuántas personas se hospedarán en la habitación asignada.

Los siguientes campos corresponden a claves foráneas en diferentes tablas cuya relación se encuentra definida en la Figura 2.13, el `Id_Habitacion` indica que habitación se encuentra reservada, el `Id_Cliente` indica a nombre de quien se encuentra la reserva y el `Id_Usuario` indica que usuario fue el que realizó la reserva.

El campo de estado permite conocer si la reserva se encuentra confirmada, modificada o cancelada.

2.4.7 Tabla Registro_Cliente

En esta tabla se almacenan los registros de entrada-salida de un cliente del hotel, y por cada habitación en la cual se registra un cliente, se almacena un registro de entrada y salida.

Los siguientes campos corresponden a claves foráneas de diferentes tablas, su relación se define en la Figura 2.13. El `Id_Cliente` indica a nombre de quien se realiza el registro de entrada-salida de la habitación, el `Id_Habitacion` indica en que habitación se hospeda el cliente y los huéspedes que lo acompañan, el `Id_Usuario_Ingreso` indica que usuario fue el que realizó el registro de entrada del cliente, el `Id_Usuario_Salida` indica que usuario fue el que realizó el registro de salida del cliente y el `Id_Factura`, si es que existe, indica el comprobante de pago que se generó para dicho registro.

2.4.8 Tabla Huesped

En esta tabla se almacenan los datos correspondientes a cada huésped que tiene el hotel, como se puede observar en la Figura 2.13, los campos de documento, nombre, nacionalidad y genero almacenan datos personales del huésped. Se decidió almacenar la fecha de nacimiento del huésped para poder conocer la edad al momento de realizar el registro de entrada-salida en el hotel.

2.4.9 Tabla Huesped_Registro

Esta tabla relaciona el registro de entrada-salida con los huéspedes alojados en la habitación correspondiente a dicho registro, se la creó para evitar redundancia en los datos.

2.4.10 Tabla Factura

En la tabla factura se almacena solo la fecha de emisión de la factura y el usuario que la creó, ya que permite relacionarla con el registro de entrada-salida correspondiente y los extras que se crearon para dicho comprobante.

2.4.11 Tabla Extras

La tabla de extras almacena datos de los elementos extra que se ingresaron en un comprobante de pago los cuales son la descripción, el costo y la cantidad del elemento. Se decidió crear esta tabla a pesar de conocer que existirán datos redundantes ya que el sistema no se enfoca en llevar un inventario de productos o servicios que provee el hotel aparte del servicio de alojamiento.

2.5 DIAGRAMA DE CLASES

El diagrama de clases del sistema se encuentra dividido en cinco partes que incluyen a la conexión con la base de datos, el módulo de administración general, el módulo de reservas de habitaciones, el módulo de registros de entrada-salida y el módulo de reportes y se lo puede encontrar en el ANEXO II.

2.5.1 Base de datos

El diagrama de clases correspondiente a la base de datos comprende la clase `dbExternaDataContext`, la cual es una clase del tipo `LINQtoSQL` y contiene todos los procedimientos almacenados necesarios para el funcionamiento del sistema, dichos procedimientos almacenados se tratan como métodos como se puede observar en la Figura 2.14.

Todos los métodos son públicos, ya que son necesarios en todos los formularios del sistema. Se encuentran definidos los parámetros de entrada de cada procedimiento almacenado, y el resultado es del tipo `ISingleResult` perteneciente a LINQ. En las clases restantes se crea una instancia de esta clase, para poder acceder a los métodos y por lo tanto a los procedimientos almacenados.

2.5.2 Administración

Para poder realizar el control de acceso se utiliza el diagrama de clases de `Login` el

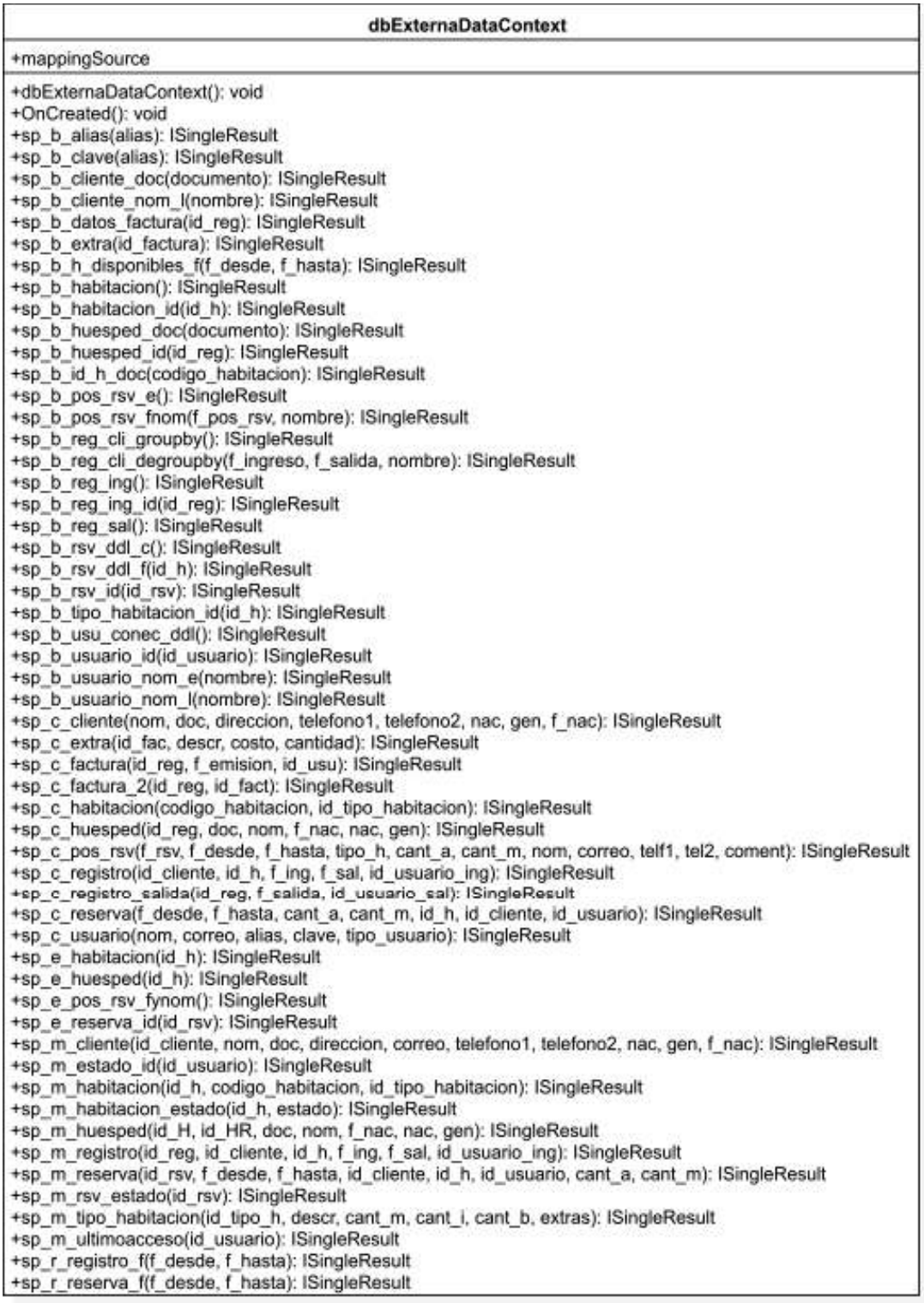

Figura 2.14: Diagrama de clases de la conexión con la base de datos

cual contiene los elementos necesarios para realizar dicho proceso como se puede observar en el diagrama de clases que se encuentra en la Figura 2.15.

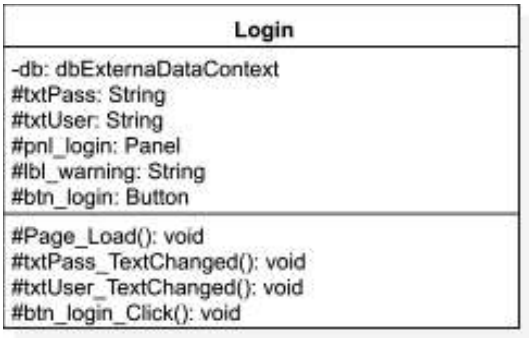

Figura 2.15: Diagrama de clases de *Login*

Una vez ingresado al sistema se utiliza la clase `MasterPage`, la cual se puede observar en la Figura 2.16, para poder acceder a cada uno de los formularios restantes. Además, se introduce una variable del tipo *integer* denominada `id`, la cual se encuentra en todas las demás clases, y permite almacenar la identificación que tiene el usuario conectado en la base de datos y mediante el uso de los métodos `MenuAdmin`, `MenuEncargado` y `MenuUsuario` otorgar los permisos de acceso a los formularios que corresponda al usuario conectado.

Figura 2.16: Diagrama de clases de la página maestra

Una vez se ha otorgado los permisos de acceso a los formularios se procede a almacenar el identificador del usuario conectado en cada una de las clases con la variable `id`.

La clase de administración de clientes y la clase administración de usuarios contienen todos los elementos y variables necesarias para poder crear, leer y actualizar los clientes y usuarios respectivamente en la base de datos.

La clase de administración de habitaciones contiene todos los elementos y variables necesarias para poder modificar una habitación. Para el proceso de creación de una habitación se utiliza la clase `Nueva_Habitacion` y para realizar la modificación del tipo de habitación se utiliza la clase `Modificar_Tipo_H`.

La clase `Mi_Perfil` se utiliza para modificar los datos del usuario que se encuentra utilizando el sistema en ese instante por lo que posee los elementos y variables necesarios para mostrar los datos correspondientes al usuario conectado y modificarlos en la base de datos y por último la clase `Usuarios_Conectados` contiene todos los elementos y variables necesarias para poder visualizar los usuarios conectados en ese instante y cambiar su estado a desconectado en la base de datos.

2.5.3 Reservas de Habitaciones

Para los procesos que involucren a las reservas de una habitación en un hotel, ya sea por parte del cliente o por parte de un usuario del hotel, se utilizan las clases `Reservas`, `Confirmar_Reservas` y `Modificar_Reservas` indicadas en el diagrama que se encuentra en el ANEXO II.

Para permitir que un cliente realice una posible reserva de una habitación, se utiliza la clase Reserva, la cual contiene todos los elementos y variables necesarias para la creación de una posible reserva en la base de datos, y el envío de un correo electrónico al cliente con los datos correspondientes a la posible reserva creada.

La clase `Confirmar_Reservas` contiene todos los elementos y variables necesarias para poder visualizar una posible reserva realizada por un cliente, crear una reserva en la base de datos y realizar el envío de un correo electrónico al cliente con los datos de la reserva creada, y por último la clase `Modificar_Reservas` contiene todos los elementos y variables necesarias para leer, crear, actualizar y eliminar una reserva en la base de datos.

En `Confirmar_Reservas` existen varios métodos, uno de los cuales es

`LlenarCampos();` dicho método se encarga de obtener los datos de una posible reserva seleccionada, mediante el uso de los procedimientos almacenados que se encuentran en `dbExternaDataContext`, para luego llenar los campos correspondientes. Los métodos que conforman el sistema dependen de cada proceso que realiza cada módulo.

2.5.4 Registro de Entrada-Salida

Para los procesos que involucran a los registros de entrada, los registros de salida o el comprobante de pago electrónico se utilizan las clases `Registro_Entrada`, `Registro_Salida`, `Modificar_Registro` y `Comprobante` indicados en el diagrama que se encuentra en el ANEXO II.

La clase `Registro_Entrada` contiene todos los elementos y variables necesarias para realizar la creación de un registro de entrada en la base de datos, esto incluye la creación o actualización de los datos correspondientes a los huéspedes de dicho registro de entrada.

La clase `Modificar_Registro` contiene todos los elementos y variables necesarias para leer y actualizar un registro de entrada, al igual que la lectura, creación y actualización de todos los huéspedes asociados al registro de entrada seleccionado.

La clase `Registro_Salida` contiene todos los elementos y variables necesarias para seleccionar un registro de entrada, y realizar la creación de un registro de salida en la base de datos.

La clase `Comprobante` contiene todos los elementos y variables necesarias para visualizar los registros de entrada-salida completados, y crear un comprobante de pago en la base de datos. Una vez creado el comprobante, se puede crear un documento PDF con los datos de dicho comprobante y enviarlo al cliente.

2.5.5 Reportes

El diagrama de clases que se muestra en la Figura 2.17 contiene todos los elementos y variables necesarias para la creación de un reporte de los registros de entrada-salida y de las reservas de habitaciones realizadas, limitadas por una fecha desde y una fecha hasta.

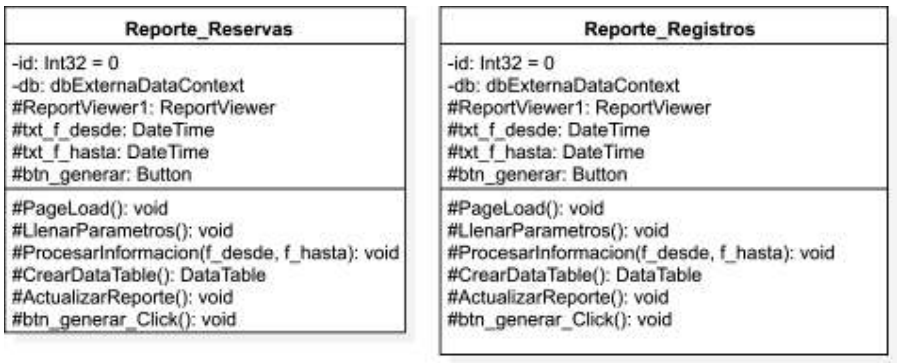

Figura 2.17: Diagrama de clases del módulo de reportes

2.6 DISEÑO VISUAL

Para la realización del diseño visual del sistema se utilizaron *mockups* que permiten evaluar el diseño realizado y de esta manera identificar y corregir posibles errores que se puedan presentar en el momento de implementar el sistema.

Se divide el diseño visual del sistema en tres partes:

- Reservas de habitaciones por parte del cliente
- Control de acceso al sistema
- Administración del sistema

Para nombrar cada uno de los elementos que contendrían los formularios se decidió utilizar contracciones que identifiquen a cada elemento por su tipo según la Tabla 2.3.

Tabla 2.3: Contracciones de los elementos ASP

Elemento	Contracción
Button	btn
Input	inp
Textbox	txt
Dropdownlist	ddl
Panel	pnl
Radiobuttonlist	rbtn
Label	lbl
Imagebutton	ibtn

De la misma manera se utilizan contracciones para hacer referencia a tipos específicos

de variables. Para separar cada parte del nombre del elemento o variable se utiliza el carácter guion bajo _. En la Tabla 2.4 se definen las contracciones utilizadas para los datos:

Tabla 2.4: Contracciones de los datos del sistema

Dato	Contracción
Fecha	f
Adulto	a
Menor	m
Habitación	h
Huésped	H
Cantidad	cant
Base de datos	db
Identificador	id
Contraseña	pwd
Modificar	mod
Nueva habitación	nh
Cama individual	cam_i
Cama matrimonial	cam_m
Baño	ba
Código	cod
Cliente	cli
Registro	reg
Buscar	b
Extra	e
Descripción	descr

2.6.1 Formulario Reservas.aspx

Como se especificó en el análisis de requerimientos, este formulario puede ser accedido por cualquier persona y contiene información de las habitaciones que posee el hotel. Para la creación de la sección de reservas se tomó en cuenta los datos necesarios, definidos en el análisis de requerimientos, y su diseño se puede observar en la Figura 2.18.

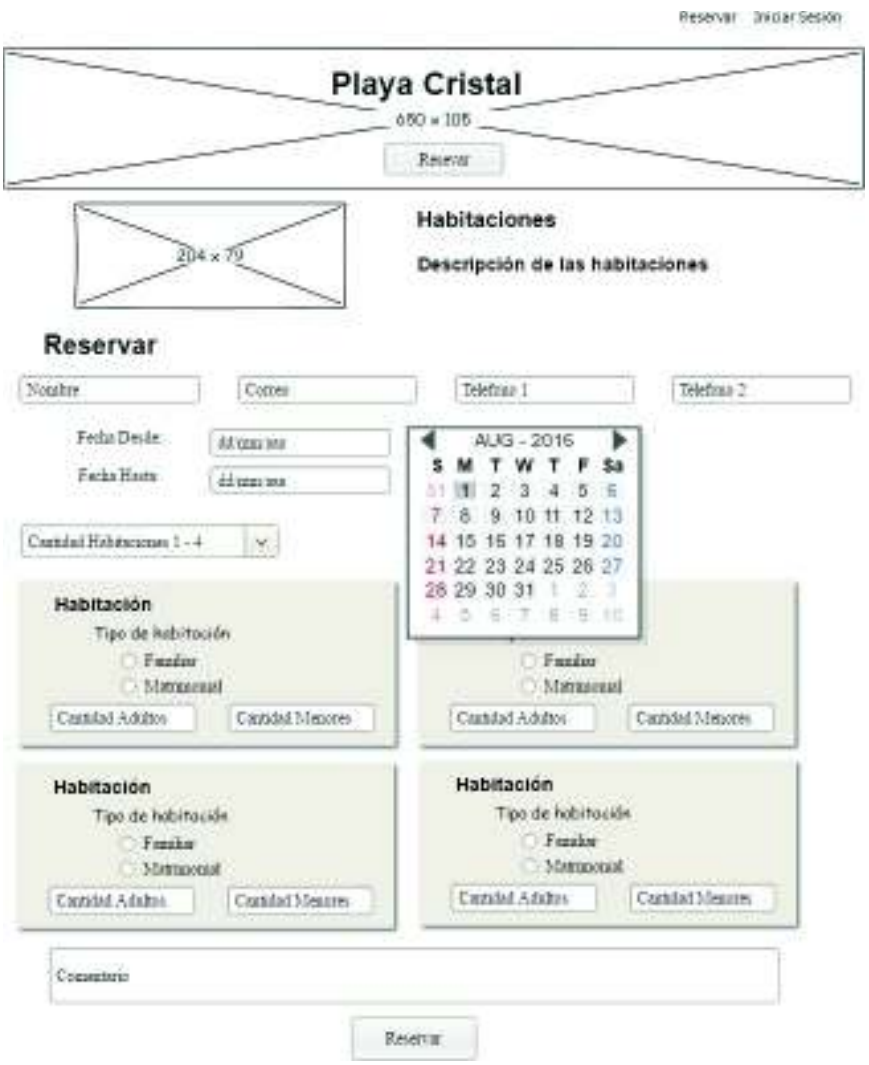

Figura 2.18: Diseño del formulario de reservas

2.6.2 Diseño Login.aspx

Se puede acceder a este formulario mediante el enlace que se encuentra en la página principal del sistema, llamado `Iniciar Sesión`, cabe indicar que, al ser un prototipo de sistema, dicho enlace solo se encuentra visible por motivos de permitir acceder al sistema como usuario de una manera rápida y sencilla, aunque se podría encontrar otra forma de acceder a `Login.aspx`, sin que sea visible para todas las personas que accedan a `Reservas.aspx`.

Este formulario permite ingresar el usuario y contraseña para acceder al sistema, por lo que solo requiere de dos campos y un botón, el diseño visual se lo realizó mediante la utilización de una plantilla con licencia *Creative Commons* de [33]. En la Figura 2.19 se puede observar el diseño con los elementos necesarios para realizar el ingreso al

sistema.

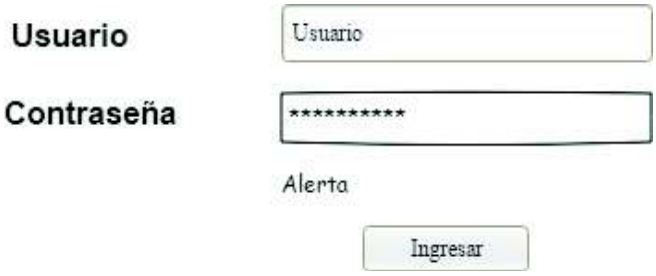

Figura 2.19: Diseño del formulario de *Login*

2.6.3 Página Maestra

"Las páginas maestras de ASP.NET permiten crear un diseño coherente para las páginas de la aplicación. Puede definir el aspecto, el diseño y el comportamiento estándar que desea que tengan todas las páginas (o un grupo de páginas) de la aplicación en una sola página maestra [34]".

En el presente trabajo de titulación, la página maestra realiza la función de un menú de acceso a todos los demás formularios, dependiendo del tipo de usuario que quiere acceder. Se decidió que la posición de dicho menú será en la parte izquierda de los formularios y su diseño se lo puede observar en la Figura 2.20.

Figura 2.20: Diseño de la página maestra

2.6.4 Diseño Admin_Usuarios.aspx

Este formulario permite modificar los datos de los usuarios, definidos en el análisis de requerimientos, por lo que posee campos de texto para cada uno de los datos, además se incluye un selector el cual permite escoger que acción desea realizar el usuario como se puede observar en la Figura 2.21.

Figura 2.21: Diseño de la administración de usuarios

2.6.5 Diseño Admin_Clientes.aspx

Este formulario permite la creación o modificación de un cliente, se incluye un selector que permite escoger la acción que desea realizar el usuario. El diseño se lo puede observar en la Figura 2.22.

Figura 2.22: Diseño de la administración de clientes

Para la modificación de un cliente es necesario seleccionar el cliente que se desea modificar por lo que al momento de escoger la opción *modificar cliente* se añade el campo que se puede observar en la Figura 2.23.

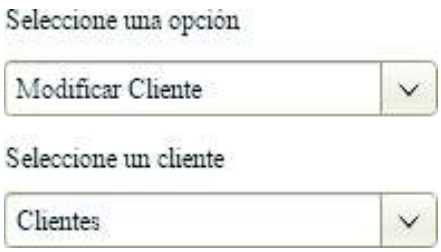

Figura 2.23: Diseño de la modificación de clientes

2.6.6 Diseño Admin_Habitaciones.aspx

Para realizar la administración de las habitaciones que existen en el hotel se necesitan realizar tres acciones: crear una habitación, modificar una habitación y modificar el tipo de habitación por lo que se añade un selector con dichas acciones.

Para la acción de crear una habitación se decidió realizar el diseño de un formulario emergente, dicho diseño se puede observar en la Figura 2.24.

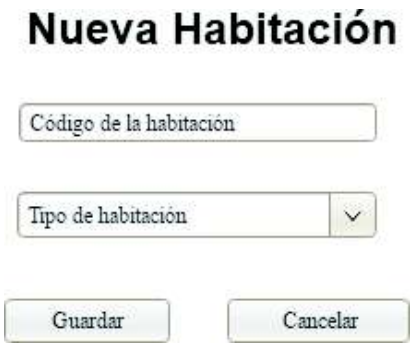

Figura 2.24: Diseño de la creación de habitaciones

Para la modificación de una habitación se necesita poder seleccionar las habitaciones existentes en el hotel por lo que se añade un selector de habitaciones, una vez seleccionada la habitación que se desea modificar se habilita un panel que contiene los datos que se pueden modificar de dicha habitación como se muestra en la Figura 2.25.

Para la última acción la cual corresponde a la modificación del tipo de habitación se decidió, al igual que en la creación de una habitación, crear un formulario emergente con los datos que se muestran en la Figura 2.26.

Figura 2.25: Diseño de la modificación de una habitación

Figura 2.26: Diseño de la modificación del tipo de habitación

2.6.7 Diseño Mi_Perfil.aspx

Este formulario permite modificar los datos correspondientes al usuario que se encuentra conectado en ese instante, por lo tanto, solo puede acceder a sus datos como se muestra en la Figura 2.27.

2.6.8 Diseño Confirmar_Reservas.aspx

El diseño de la página para la confirmación de las reservas se puede observar en la Figura 2.28.

Figura 2.27: Diseño del formulario *Mi Perfil*

Figura 2.28: Diseño del formulario de confirmación de reservas

Una vez realizada una reserva por parte del cliente en el formulario reservas.aspx es necesario poder visualizar dichas reservas en este formulario por lo que se añade un

selector con todas las reservas realizadas por los clientes y los datos necesarios para su confirmación.

Se decidió permitir la creación de clientes directamente desde este formulario ya que es aquí donde se visualizarán todas las posibles reservas con clientes nuevos para ello se crea un panel de diferente color el cual permanecerá oculto al cargar la página y se puede mostrar u ocultar mediante el botón *Más* y el botón *Menos* respectivamente.

Una vez seleccionado las fechas desde y hasta de la reserva se procederá a realizar la búsqueda de las habitaciones que se encuentran disponibles para esas fechas y asignar la cantidad de personas adultas y menores que se alojarán en las mismas.

2.6.9 Diseño Modificar_Reservas.aspx

En este formulario se realizarán las tareas de creación, modificación y eliminación de una reserva por lo que es necesario permitir al usuario seleccionar que acción desea realizar, la acción de crear una reserva se encuentra seleccionada en la Figura 2.29.

Figura 2.29: Diseño del formulario de crear/modificar reservas

2.6.10 Diseño Registro_Entrada.aspx

El formulario `Registro_Entrada.aspx` debe permitir al usuario realizar el primer proceso definido en el módulo de registro de entrada-salida de un cliente, como se observa en la Figura 2.2; dicho proceso está asociado a una reserva de una habitación a nombre del cliente por lo tanto primero debe cumplirse dicho requisito.

En este formulario se muestra una lista de las reservas realizadas por parte de los clientes, en un elemento *DropDownList*, siempre y cuando la fecha de ingreso no sea menor a la fecha actual. Luego de seleccionar la reserva del cliente se procede a ingresar la fecha y hora del registro en dos campos separados, como se muestra en la Figura 2.30; el campo correspondiente a la fecha de ingreso se llenará de acuerdo con la reserva seleccionada y la hora de ingreso se llena con la hora actual del servidor.

Figura 2.30: Diseño del formulario de registro de entrada

Una vez ingresados los datos del registro, se debe ingresar los datos de las personas que se hospedarán en la habitación asignada, como se especifica en el análisis de requerimientos, para ello se puede seleccionar la cantidad de personas según lo cual se habilitarán los paneles que contienen los datos correspondientes al huésped, como se muestra en la Figura 2.30.

Debido a que el cliente a nombre de quien se realizará el registro de entrada puede o no ser un huésped del hotel existe la posibilidad de eliminar los datos del mismo del panel de huéspedes.

2.6.11 Diseño Registro_Salida.aspx

En este formulario se puede seleccionar un registro de ingreso de un cliente mediante el uso de un *DropDownList*. Una vez seleccionado el registro de entrada se podrá asignar la fecha y hora de salida del hotel como se muestra en la Figura 2.31.

Figura 2.31: Diseño del formulario de registro de salida

2.6.12 Diseño Modificar_Registro.aspx

En este formulario se puede seleccionar un registro de entrada existente y modificar sus datos, al igual que el de todos los huéspedes asociados a dicho registro, el diseño de dicho formulario se puede observar en la Figura 2.32.

Figura 2.32: Diseño del formulario de modificación de registros

2.6.13 Diseño Comprobante.aspx

Este formulario permite la creación y envío de un comprobante de pago en formato PDF al correo electrónico del cliente, para ello se escoge un registro de entrada-salida, se calcula en base a la edad de los huéspedes cuantas personas adultas y cuantas personas menores se registraron, luego se calcula el costo que tendrá que cancelar el cliente dependiendo del costo de la habitación el cual se definió en el análisis de requerimientos, en la Figura 2.33 se puede observar el diseño que permite crear el comprobante.

Figura 2.33: Diseño del formulario de creación de comprobante de pago

Además de los datos del registro y el costo a cancelar del servicio de hospedaje, se puede añadir elementos extra al comprobante, como se definió en el análisis de requerimientos, para poder añadir dichos elementos, hasta un total de cinco, se creó el diseño que se muestra en la Figura 2.33.

2.6.14 Diseño Reporte_Registro.aspx

En este formulario se presenta los reportes de registros de entrada-salida entre dos fechas deseadas, para poder generar dicho reporte se permite el ingreso de las fechas como se lo puede ver en la Figura 2.34.

Figura 2.34: Diseño del formulario de reportes de registros

Para poder generar el reporte se utiliza *ReportViewer* de Microsoft el cual requiere de un diseño de un reporte que se puede observar en la Figura 2.35

Figura 2.35: Diseño del reporte de registros para *ReportViewer*

2.6.15 Diseño Reporte_Reservas.aspx

Este formulario es similar al formulario *Reporte_Registro.aspx* la diferencia que posee se encuentra en el diseño para *ReportViewer* el cual se muestra en la Figura 2.36

Figura 2.36: Diseño del reporte de reservas para *ReportViewer*

2.7 CREACIÓN DE LA BASE DE DATOS LOCAL

Para la creación de la base de datos se utilizó la herramienta *SQL Server Management Studio 2012* y el diagrama relacional y en cuanto a la realización de consultas desde los formularios se lo hace mediante el uso de los procedimientos almacenados definidos en el diagrama de clases.

Primero se crearon las tablas definidas en el diagrama relacional cada una con un identificador auto-incremental, en el Segmento de Código 2.1 se puede observar un ejemplo de creación de la tabla Usuario.

```
1 --Comando para la creación de una tabla de nombre Usuario
2 CREATE TABLE Usuario (
3    --Todas las tablas poseen un Id autoincremental
4    -- El Id es la clave primaria de cada tabla
5    Id_Usuario INT IDENTITY(1,1) PRIMARY
6    --Se crean las                KEY,
7    -- Cada columna                columnas de los datos
8    Nombre VARCHAR(100) UNIQUE,
9    Correo NVARCHAR(254) UNIQUE,
10   Alias  VARCHAR(50)  UNIQUE,
11   --La clave se guarda en el tipo varbinary.
12   Clave VARBINARY(MAX),
13   Tipo_Usuario TINYINT,
14   Estado TINYINT,
15   Ultimo_Acceso DATETIME )
```

Segmento de Código 2.1: Creación de la tabla Usuario

Para la creación de los procedimientos almacenados se tomó en cuenta los datos de entrada necesarios y los datos de salida que utilizaría el procedimiento almacenado.

En el Segmento de Código 2.2 se puede observar un ejemplo de creación del procedimiento almacenado para el registro de un huésped, en dicho procedimiento, denominado `sp_c_huesped`, se requieren seis datos de entrada correspondientes al identificador del registro de entrada que se está modificando y los datos del huésped que se desea ingresar.

Se realiza un control de excepciones ya que se afectan dos tablas y mediante el uso de transacciones se asegura que se realicen correctamente todas las consultas.

```
-- Inicio del procedimiento almacenado
CREATE PROCEDURE sp_c_huesped
--Se definen los parámetros de entrada del procedimiento
@id_registro INT,@doc VARCHAR(50),@nombre VARCHAR(50),
@f_nac DATE,@nacionalidad VARCHAR(20),@genero VARCHAR(20)
AS
BEGIN TRY -- Inicio del control de excepciones
   BEGIN TRAN_c_huesped --Inicio de la transacción
      DECLARE @id_huesped INT
     -- Se realiza solo si es que no existe el hué sped
     IF NOT EXISTS(SELECT Id_Huesped FROM Huesped WHERE
       Documento_H = @doc)
     BEGIN -- Se crea el hué sped y se crea el registro
       INSERT INTO Huesped VALUES (@doc, @nombre,
         @nacionalidad, @genero,@f_nac)
       SET @id_huesped = SCOPE_IDENTITY()
       -- Se inserta el registro
       INSERT INTO huesped_registro VALUES (@id_registro,
         @id_huesped)
     END
     ELSE -- Se realiza solo si es que existe el hué sped
     BEGIN
       -- Se obtiene el id del hué sped para modificarlo
       SET @id_huesped = (SELECT Id_Huesped FROM Huesped
           WHERE Documento_H = @doc)
       -- Se inserta el registro
       INSERT INTO huesped_registro VALUES (@id_registro,
         @id_huesped)
       UPDATE Huesped SET Documento_H = @doc,
       Nombre_H = @nombre, F_Nacimiento = @f_nac,
       Nacionalidad_H = @nacionalidad,
       Genero_H = @genero
       WHERE Id_Huesped = @id_huesped
     END
   --Si no existió errores se realiza toda la transacción
      COMMIT TRAN_c_huesped
END TRY
BEGIN CATCH
   --Si existió errores se regresa al inicio de la transacción
     ROLLBACK TRAN_c_huesped
     SELECT ERROR_MESSAGE() AS errMessage
     END CATCH
```

Segmento de Código 2.2: Creación de un procedimiento almacenado

Una vez creadas todas las tablas y los procedimientos almacenados necesarios, se procedió a crear una clave asimétrica para el cifrado de las contraseñas de usuarios, como se puede observar en el Segmento de Código 2.3. Dicha clave asimétrica se utiliza en el cifrado de datos mediante la sentencia de la línea 8 y 9 del Segmento de Código 2.3, y para el descifrado mediante; la sentencia de la línea 11 y 12 del Segmento de Código 2.3.

```
1  --Código para la creación de la clave asimétrica
2  CREATE ASYMMETRIC KEY Clave Usuarios
3    WITH ALGORITHM = RSA_ 2048
4    --Se indica la clave
5     ENCRYPTION BY PASSWORD = N'RobDav'
6  GO
7  --Código para el cifrado con la clave asimétrica creada
8   EncryptByAsymKey(AsymKey_ID('Id_Clave_Asimetrica'),
9          Dato_a_cifrar)
10 --Código para el descifrado de la clave guardada en la base
11  DecryptByAsymKey(AsymKey_ID('Id_Clave_Asimetrica'),
12       [Clave], N'Clave' ))
```

Segmento de Código 2.3: Creación de un procedimiento almacenado

Una vez creada la clave asimétrica se procede a la creación de un usuario administrador y la creación de los tipos de habitaciones que posee el hotel, definidos en el análisis de requerimientos.

El *script* de la base de datos incluyendo todos los procedimientos almacenados necesarios para el funcionamiento del sistema se encuentra en el ANEXO IV, dicho documento contiene comentarios que facilitan el entendimiento del mismo.

2.8 CODIFICACIÓN DEL SISTEMA

Para la codificación del sistema primero se creó una solución en Microsoft *Visual Studio 2017* y luego se añadieron cada uno de los formularios, definidos en el análisis de requerimientos, a medida que se los iba desarrollando como se indica a continuación:

- `Reservas.aspx`
- `Login.aspx`
- `MasterPage.Master`

- `Inicio.aspx`
- `Admin_Usuarios.aspx`
- `Usuarios_Conectados.aspx`
- `Admin_Habitaciones.aspx`
- `Nueva_Habitacion.aspx`
- `Modificar_Tipo_H.aspx`
- `Admin_Clientes.aspx`
- `Mi_Perfil.aspx`
- `Confirmar_Reservas.aspx`
- `Modificar_Reservas.aspx`
- `Registro_Entrada.aspx`
- `Modificar_Registro.aspx`
- `Registro_Salida.aspx`
- `Comprobante.aspx`
- `Reporte_Registros.aspx`
- `Reporte_Reservas.aspx`

Los elementos de cada formulario se encuentran definidos en los diagramas de clases correspondientes, al igual que sus variables, la ubicación de los elementos se definió en el diseño visual del sistema y además se tuvo en consideración que en la actualidad la mayoría de personas ingresan a una página web por medio de un celular inteligente, y diferentes exploradores web, por lo que para realizar el diseño responsivo de los formularios se utilizó dos *framework*: *Bootstrap* y *Skeleton,* basados en CSS3.

Bootstrap al ser un *framework* que ofrece mejores características que *Skeleton,* se utiliza en la página de reservas en línea. La página de reserva de habitaciones en línea permite el acceso de cualquier persona desde cualquier navegador web y cualquier dispositivo. Para su correcto funcionamiento es necesario incluir los archivos que comprenden el *framework Bootstrap* como se observa en la Figura 2.37. En cuanto a los archivos que forman parte del *framework* de *Skeleton* se los puede observar en la Figura 2.38.

Una vez creados los formularios con sus elementos respectivos y añadido el estilo *CS*S3 correspondiente, se procede a crear una *Clase de LINQ a SQL*, denominada `dbExterna.dbml`, la cual permite convertir las tablas de SQL Server en clases, al igual

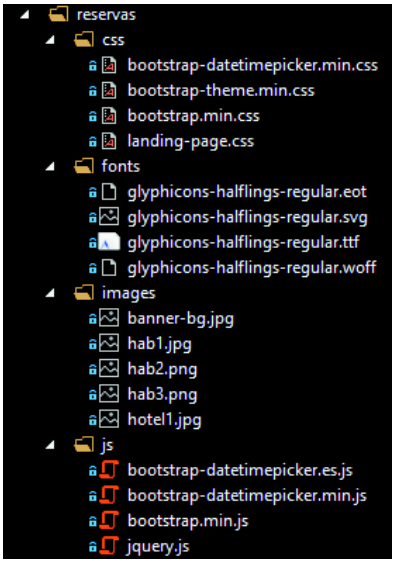

Figura 2.37: Archivos que utiliza *Bootstrap*

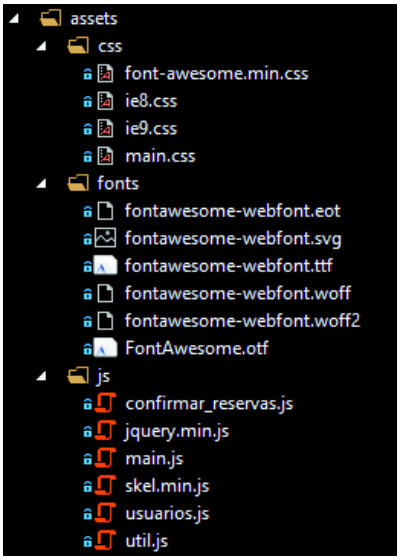

Figura 2.38: Archivos que utiliza *Skeleton*

que los procedimientos almacenados, permitiendo ejecutarlos mediante el uso de LINQ en los diferentes formularios del sistema, como se muestra en la Figura 2.14.

Para cada formulario se utiliza un *UpdatePanel*, el cual es un control que "permite actualizar las partes seleccionadas de una página en lugar de actualizar toda la página con una devolución de datos. Esto se conoce como actualización parcial de la página [35]".

La función que cumple cada formulario, en cuanto a la codificación se refiere, es principalmente la de realizar validaciones de los datos ingresados, mostrar datos

almacenados en la base de datos e ingresar o modificar datos en la base de datos mediante el uso de los procedimientos almacenados.

Las validaciones se las realizan en cada formulario y obligan a que los parámetros ingresados por el usuario sean correctos, y correspondan a los que se deben ingresar en la base de datos; mientras no se validen todos los campos necesarios no se procederá a realizar ninguna acción. Para ello se utilizan las herramientas de validación de ASP.NET que se muestran en la Figura 2.39.

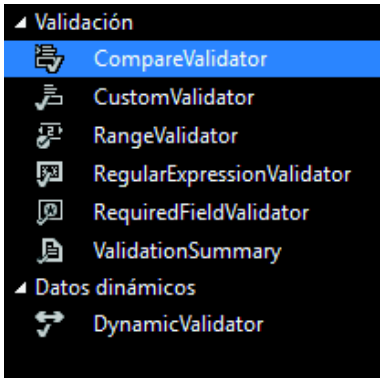

Figura 2.39: Herramientas de validación de ASP.NET

En el sistema se utilizan los controles `CustomValidator`, `RangeValidator`, `RegularExpressionValidator` y `RequiredFiedlValidator`.

a. Custom Validator

Permite realizar validaciones personalizadas mediante un método que se ejecute en el lado del cliente con JavaScript o en el lado del servidor con C#. Para realizar estas validaciones se escogió C#. En el Segmento de Código 2.4 se puede observar la validación del ingreso correcto de las fechas *desde* y *hasta* de un formulario.

```
1    //custDate_ServerValidate corresponde al método que realiza
2    //la validación
3    protected void custDate_ServerValidate(object sender,
4    ServerValidateEventArgs e) {
5    try {
6    //Se obtienen los valores de las fechas
7    DateTime f_desde = DateTime.Parse(inp_f_desde.Value);
```

Segmento de Código 2.4 (Parte 1 de 2): Validación del ingreso de fechas en un formulario

```
8     DateTime f_hasta = DateTime.Parse(inp_f_hasta.Value);
9     //Se calcula el número de días solicitados
10    TimeSpan dias = f_hasta - f_desde;
11    //Se valida que la fecha hasta sea
12    //desde y que el número de días se
13          if (f_hasta > f_desde && dias.Days <= 20)
14                e.IsValid = true; else
15                e.IsValid = false;
```

Segmento de Código 2.4 (Parte 2 de 2): Validación del ingreso de fechas en un formulario

b. RangeValidator

Permite conocer si un dato ingresado en un campo se encuentra en el rango definido; cabe aclarar que es necesario identificar qué tipo de dato se desea validar ya que si el tipo es texto y el rango es un valor numérico no se realizará correctamente la validación.

c. RegularExpressionValidator

Se utiliza en todo el sistema para verificar que se ingrese en el campo indicado, un correo electrónico con el formato correcto como se muestra en el Segmento de Código 2.5.

```
1   <asp:RegularExpressionValidator
2   ID="email" runat="server" ForeColor="Red"
3   CssClass="validator" Display="Dynamic"
4   ControlToValidate="inp_email"
5   <!--Mensaje que se mostrará en caso de existir un error-->
6   ErrorMessage="Ingrese un email válido"
7   <!--Aquí es donde se valida que se posea el formato correcto-->
8   ValidationExpression=
9   "\w+([-+.']\w+)*@\w+([-.]\w+)*\.\w+([-.]\w+)*">
10</asp:RegularExpressionValidator>
```

Segmento de Código 2.5: Validación de un correo electrónico

d. RequiredFieldValidator

Esta herramienta es la que verifica que un campo indicado no se encuentre vacío, en todo el sistema existen campos que pueden ser opcionales, por lo que serán los únicos que no posean este validador.

Una vez validados todos los campos se pueden realizar otras validaciones dependiendo del formulario y los procesos que este realice.

2.8.1 Configuración inicial del sistema

Existen varias características que se deben configurar antes de realizar la creación de los formularios. Existen formularios que realizan el envío de correos electrónicos a los clientes, para ello es necesario utilizar el puerto `TCP/IP 25` y en el caso de que este se encuentre bloqueado el puerto `TCP/IP 587`, que corresponde a SMTP. También es necesario un servidor de correo, en el sistema se utiliza `smtp.gmail.com` y una cuenta de correo personal, para el uso de este servicio es necesario la librería `System.Net.Mail`.

a. Variable Session

En los formularios ASP.NET es posible crear variables de sesión las cuales permanecen almacenadas mientras dure la sesión en el explorador web, en el Segmento de Código 2.6 se puede observar los parámetros que tienen dichas variables.

```
1 <system.web>
2     <sessionState mode="InProc" timeout="40"
3 </system.web
```

Segmento de Código 2.6: Validación de un correo electrónico

El parámetro *mode,* en la línea 2 del Segmento de Código 2.6, posee varios tipos de valores; el valor por *default* corresponde a *InProc,* el cual almacena el estado de sesión en memoria en el servidor web. El siguiente modo es *StateServer,* el cual almacena el estado de sesión en un proceso separado. El modo *SQLServer* permite almacenar el estado de sesión en una base de datos *SQLServer*. El modo *Custom mode* permite especificar un proveedor de almacenamiento personalizado y el modo *Off* deshabilita el estado de sesión.

El parámetro *timeout* permite establecer el tiempo máximo de duración de cada sesión en minutos, una vez finalizada la sesión es necesario cambiar el estado del usuario que se encontraba conectado a desconectado; esto se lo realiza en el archivo de configuración global del sistema.

En el Segmento de Código 2.7 se puede observar el cambio de estado del usuario mediante el uso del procedimiento almacenado `sp_m_estado_id`.

```
1 //Session_End indica que se realizará las acciones cuando
2 // la variable haya expirado
3 protected void Session_End(object sender, EventArgs e){
4   objects.dbExternaDataContext db =
5   new objects.dbExternaDataContext();
6   //Se verifica que exista la variable sesión
7   if (Session["id"] != null)
8   { // Se obtiene el valor de la variable
9     int id = (Int32)Session["id"];
10    // Se cambia el estado del usuario a desconectado
11    db.sp_m_estado_id(id);
12  }}
```

Segmento de Código 2.7: Variable de sesión

2.8.2 Reservas.aspx

Este formulario puede ser accedido por cualquier persona, sin necesidad de un control de acceso. Se incluyen dos secciones relevantes dentro del formulario: información de habitaciones y reservas en línea de habitaciones, de las cuales la única que realiza procesos es la de reservas.

Para las reservas en línea es necesario ingresar los datos de la tabla `Posible_Reserva`, para ello primero es necesario realizar validaciones de dichos datos con el uso de los validadores de ASP.NET y además se debe realizar las validaciones de disponibilidad de habitaciones.

En el caso de no existir habitaciones disponibles para las fechas seleccionadas, se mostrará un mensaje al cliente solicitando el cambio de fechas. Luego se valida que la cantidad de personas adultas y personas menores que desean hospedarse se encuentre dentro del rango definido por habitación dependiendo del tipo: familiar máximo cinco personas y matrimonial máximo dos personas. Una vez validados los datos se utilizan el procedimiento almacenado `sp_c_posible_reserva` para realizar el ingreso de la posible reserva en la base de datos.

Después de crear todas las posibles reservas, se realiza el envío de un correo de confirmación al cliente indicando las fechas desde y hasta de la reserva, y el identificador de la *posible reserva* creada, denominada *ticket*, en el correo.

El diseño de la página se la puede observar en la Figura 2.18 y la creación con todos los elementos necesarios, incluyendo los *scripts* necesarios para el funcionamiento de *Bootstrap* encuentra disponible en el ANEXO IV formulario `Reservas.aspx`. En el Segmento de Código 2.8 se observa un ejemplo de creación de los elementos de `Reservas.aspx`.

```
<!--Inicio de la división de los campos para Reservas-->
<div class="row uniform">
  <div class="col-lg-5 col-sm-6">
    <!--Se coloca el título de la sección Reservas-->
    <hr class="section-heading-spacer" />
      <div class="clearfix"></div>
        <h2 class="section-heading">Reservas</h2>
      </div></div>
  <div class="row uniform">
    <!--Creación del campo nombre en la sección reservas-->
    <div class="col-lg-3 col-lg-offset-2 col-sm-5 col-xs-11">
      <!--Etiqueta del campo nombre-->
      <label for="inp_nombre">Ingrese su nombre:</label>
      <!--Campo nombre del tipo input-->
      <input id="inp_nombre" runat="server" type="text"
      placeholder="Nombre" maxlength="100" />
      <!--Validador de campo requerido-->
      <asp:RequiredFieldValidator ID="RequiredFieldValidator1"
  runat =" server " Display =" Dynamic "
  ControlToValidate="inp_nombre" ForeColor="Red"
  ErrorMessage="*"></asp:RequiredFieldValidator>
</div>
<!--El código continúa con todos los elementos necesarios
para el funcionamiento especificado-->
```

Segmento de Código 2.8: Ejemplo de creación de *Reservas.aspx*

En el Segmento de Código 2.8 se observa la creación de divisiones, con clases de *Bootstrap,* como se explican en la Tabla 1.1. Dichas divisiones ocuparán el número de columnas indicadas, de acuerdo con el tamaño del explorador web. También se puede observar otras clases como *uniform* la cual indica que la división será del tipo uniforme.

2.8.3 Login.aspx

Este formulario solo posee dos campos de texto, una etiqueta y un botón, el primer

campo de texto es para el ingreso del nombre de usuario y el segundo para el ingreso de la contraseña y la etiqueta permite mostrar un mensaje de error.

Se utiliza validadores de ASP.NET para comprobar que los campos de usuario y contraseña no se encuentren vacíos. Una vez comprobados los campos se realiza la comparación del usuario y contraseña ingresados con los de la tabla Usuario de la base de datos, para ello se utiliza el procedimiento almacenado `sp_b_clave`, el cual devuelve la clave del usuario ingresado en texto plano, si no existe el usuario en la base de datos se mostrará el mensaje: *No existe el usuario.* Si el usuario existe, pero la contraseña no coincide, se mostrará el mensaje: *Contraseña incorrecta*.

Si el usuario y contraseña coinciden se procederá a crear una variable *Session,* llamada `id`, que almacena el identificador del usuario. Luego se almacenará la fecha y hora de acceso del usuario en la base de datos.

El diseño se lo puede observar en la Figura 2.19 y para la implementación mediante ASP.NET se utiliza el código que se puede observar en el Segmento de Código 2.9.

```
<table style="width:  100% ;">
  <tr><td style="width: 33% ;"> 
    <!--Campo para el ingreso del usuario-->
    <asp:TextBox ID="txtUser" runat="server"
    OnTextChanged="txtUser_TextChanged">
    </asp:TextBox><br/></td></tr>
  <tr><td></td>
  <td style="width:33%; text-align: center;"> 
    <br/><h3>Contraseña:</h3></td>
  <td style="width:33%;"> 
    <!--Campo para el ingreso de la contraseña-->
    <asp:TextBox ID="txtPass" runat="server" TextMode="Password"
    OnTextChanged="txtPass_TextChanged">
    </asp:TextBox>
    <!--Label para mostrar los mensajes de error-->
    <asp:Label ID="lblWarning" runat="server"
    Text=""></asp:Label>
```

Segmento de Código 2.9 (Parte 1 de 2): Código para la creación de *Login*

```
18    </td></tr>
19    <tr><td> </td>
20    <td> </td>
21    <td><br/>
22      <!--Botón para realizar el ingreso a la página
23      principal del sistema -->
24      <asp:Button ID="btnLogin" runat="server"
25        Text="Ingresar" OnClick="btnLogin_Click"/>
26    </td></tr>
27 </table>
```

Segmento de Código 2.9 (Parte 2 de 2): Código para la creación de *Login*

2.8.4 MasterPage.Master

En la página maestra se encuentra el enlace a los otros formularios que posee el sistema. Su diseño se lo puede observar en la Figura 2.20. En este formulario también es donde se realiza el control de acceso y para ello se utiliza la variable *Session* `id` creada en *Login.aspx*.

Por medio del procedimiento almacenado `sp_b_usuario_id`, se obtiene el tipo y el estado del usuario conectado; dependiendo del tipo de usuario se oculta los enlaces a los cuales no posee acceso como se definió en la Tabla 2.2 y si el estado del usuario no corresponde a "conectado" se regresará al formulario `Login.aspx` en donde se resetea `id`.

El control de acceso también se lo realiza en cada formulario, cada vez que se realice un cambio que afecte a la base de datos, ya que mediante el formulario `Usuarios_Conectados.aspx` se puede desconectar a un usuario específico. La creación en ASP.NET de la página maestra se la puede observar en el Segmento de Código 2.10.

2.8.5 Admin_Usuarios.aspx

Este formulario puede ser accedido solo por un administrador del sistema, ya que permite cambiar datos de todos los usuarios existentes. Para realizar la creación o modificación de un usuario es necesario validar que los datos ingresados sean correctos, si se desea crear un nuevo usuario se debe hacer una validación en la base de

datos para verificar que el nombre de usuario no exista.

```
<nav id="menu">
  <h2>Menu</h2>
      <! -- Links a cada uno de los formularios que conforman
      el sistema -->
    <ul><li><a href="Reservas.aspx">Reservas Web</a></li>
      <li><a href="Inicio.aspx">Inicio</a></li>
      <li id="admin" runat="server">
      <!-- Módulo de administración -->
      <span  class="opener">Administración</span>
        <ul><li id="a_usuarios" runat="server">
          <a href="Admin_Usuarios.aspx">Usuarios</a></li>
          <li id="a_habitaciones" runat="server">
          <a href="Admin_Habitaciones.aspx">Habitaciones</a></li>
          <li id="a_clientes" runat="server">
          <a href="Admin_Clientes.aspx">Clientes</a></li>
          <li id="a_perfil" runat="server">
          <a href="Mi_Perfil.aspx">Mi Perfil</a></li>
        </ul>
<!-- El código continúa con el resto de enlaces -->
```

Segmento de Código 2.10: Creación de la página maestra

Si se desea modificar un usuario primero se debe buscar por nombre o número de identificación, y una vez seleccionado, se procederá a la modificación. Luego de haber validado todos los datos necesarios, se procede a realizar la confirmación de contraseña; para asegurarse que la contraseña ingresada se la correcta. Luego se realiza la acción seleccionada, ya sea crear o modificar. Se utilizan los procedimientos almacenados `sp_c_usuario` y `sp_m_usuario` para la creación y modificación de un usuario respectivamente.

Este formulario posee un botón que permite abrir el formulario emergente `Usuarios_Conectados.aspx` como se puede observar en el diseño de la Figura 2.21.

a. Usuarios_Conectados.aspx

Este formulario carga un *DropDownList* con todos los usuarios cuyo estado sea el de *conectado*, una vez seleccionado un usuario de la lista se muestra la última fecha y hora

de acceso al sistema por parte del mismo y si se desea se puede desconectar a dicho usuario cambiando el estado a *desconectado*.

2.8.6 Admin_Habitaciones.aspx

Para realizar la administración de habitaciones es necesario realizar tres acciones, como se definió en el análisis de requerimientos. Para la creación de una nueva habitación se abrirá el formulario emergente `Nueva_Habitacion.aspx`, para la modificación del tipo de habitación se abrirá el formulario emergente `Modificar_Tipo_H.aspx` y para la modificación de una habitación se habilita un *DropDownList,* que contiene las habitaciones existentes.

Una vez seleccionada la habitación que se desea modificar, se procede a habilitar el panel de modificación, con los datos de la habitación. Se verifica que el código de la habitación corresponda al formato definido en el análisis de requerimientos, el cual es una letra seguida de un guión y seguida del número de habitación desde el 00 hasta 99.

a. Nueva_Habitacion.aspx

Este formulario emergente permite la creación de una habitación. Los datos necesarios son el código de la nueva habitación y el tipo de habitación. La validación del código se la realiza igual que en `Admin_Habitaciones.aspx`, luego se verifica que no exista en la base de datos el código ingresado. El tipo de habitación se escoge de un *DropDownList*.

Una vez verificados los datos se procede a ingresar los datos mediante el procedimiento almacenado `sp_c_habitacion`.

b. Modificar_Tipo_H.aspx

El tipo de habitación permite conocer la cantidad de camas, la cantidad de baños y los extras que posee una habitación. Para poder seleccionar que tipo de habitación es el que se desea modificar se carga un *DropDownList*, con los tipos de habitaciones existentes; una vez seleccionado el tipo de habitación que se desea modificar se procede a validar mediante ASP.NET los datos para proceder a ingresarlos a la base de datos, mediante el procedimiento almacenado `sp_m_tipo_habitacion`.

2.8.7 Admin_Clientes.aspx

Al formulario de administración de clientes pueden acceder todos los tipos de usuarios y es el que permite crear o modificar un cliente. Para crear un cliente se poseen datos obligatorios y datos opcionales, los cuales se definieron en el análisis de requerimientos y para la validación de los datos se utiliza ASP.NET.

Una vez validados los datos, se verifica que no exista el cliente en la base de datos, mediante la búsqueda por número de identificación; si no existe se procede a la creación del cliente mediante el procedimiento almacenado `sp_c_cliente`, caso contrario se mostrará un mensaje de error.

Para realizar la modificación de un cliente se muestra un *DropDownList* con todos los clientes registrados en el sistema, alternativamente se puede buscar por nombre; una vez seleccionado el cliente a modificar se procede a realizar las validaciones correspondientes y luego se modifica los datos mediante el procedimiento almacenado `sp_m_cliente`.

2.8.8 Mi_Perfil.aspx

Este formulario carga los datos del usuario que se encuentra actualmente conectado, permitiendo la modificación del nombre de usuario, el correo y la contraseña con su respectiva confirmación.

Una vez se han validado los datos que se desea modificar, se verifica que el nombre de usuario ingresado no esté asignado a otro usuario, de ser el caso se mostrará un mensaje solicitando que se ingrese otro nombre de usuario. Luego es necesario verificar que el correo no se encuentre asignado a otro usuario, si no se encuentra en la base de datos el nombre de usuario o correo ingresado, se procede a la modificación en la base de datos del usuario conectado, mediante el procedimiento almacenado `sp_m_usuario`.

2.8.9 Confirmar_Reservas.aspx

Al iniciar este formulario, se carga un *DropDownList* con todas las posibles reservas realizadas en el formulario *Reservas.aspx*; una vez seleccionada una de las reservas se cargan los campos con los datos de dicha reserva y se puede confirmar o eliminar la

posible reserva.

Para realizar la confirmación de una reserva, es necesario que exista un cliente al cual asignar la reserva, por lo que desde este formulario también se permite la creación de clientes. Una vez ingresado los datos de un cliente existente se procede a realizar la búsqueda de habitaciones disponibles, y dependiendo de la cantidad de habitaciones que desee el cliente se mostrarán los paneles de asignación de la habitación que se muestran en la Figura 2.40.

Figura 2.40: Panel para la asignación de habitaciones

Se realiza la validación de los datos ingresados, mediante validadores ASP.NET; luego se realiza la comprobación que el cliente ingresado exista en la base de datos, que la cantidad de personas asignadas a las habitaciones no sobrepase la cantidad máxima y que si se asigna más de una habitación no se haya seleccionado la misma dos o más veces.

Luego de comprobar que los datos necesarios son correctos, se procede a la creación de la reserva en la base de datos mediante el procedimiento almacenado `sp_c_reserva`, el cual devuelve el identificador de la reserva creada, para de esta manera crear un *ticket* y adjuntarlo al correo que se envía al cliente con los datos de la reserva.

Para eliminar la posible reserva se muestra un cuadro de confirmación, mediante el uso de código JavaScript; una vez confirmada la eliminación se procede a llamar a un método que se encuentra en C#, mediante el uso de AJAX, dicho código de confirmación se puede observar en el Segmento de Código 2.11.

```
function Confirmacion () {
       var conf = false ;
       var lbl1 = $('#Label1');
       lbl1.text('Registro eliminado!');
       conf = confirm("Desea eliminar el registro?")
       if (conf) {
       $( function () {
           var f_rsv = $("#ddl_pos_r").val();
           var nombre_pr = $("#ddl_pos_r :selected").text();
           var param_pr = { nombre: nombre_pr, fecha: f_rsv };
           $.ajax({
               url: "Confirmar_Reservas.aspx/Eliminar_PR",
               data: JSON.stringify(param_pr),
               dataType: "json",
               type: "POST",
               contentType: "application/json; charset=utf-8",
               dataFilter: function (data) { return data; },
               success:function ShowMessage(sender,args){
                   var x = document.getElementById("snackbar")
                   x.className = "show";
                   setTimeout(function () { x.className =
                   x.className.replace("show", ""); }, 3000);},
               error: function (XMLHttpRequest, textStatus,
               errorThrown) {
                   alert(errorThrown);
               } });
     });
   }
}
```

Segmento de Código 2.11: Código de confirmación de JavaScript

El método que se llama desde JavaScript tiene por nombre *Eliminar_PR* y se lo puede observar en el Segmento de Código 2.12.

Para el envío de los parámetros que necesita *Eliminar_PR*, nombre y fecha, se utiliza JSON, si el método se ejecutó correctamente se procede a mostrar un mensaje que la operación se realizó con éxito caso contrario se mostrará un mensaje de error.

```
1 [WebMethod] //Especifica que es un método web
2 public static void Eliminar_PR(string nombre, string fecha){
3   try {
4     DateTime f_rsv = Convert.ToDateTime(fecha);
5     // Se ejecuta el procedimiento almacenado
6     var exc = db.sp_e_pos_rsv_fynom(f_rsv, nombre);
7   } catch ( Exception ex){}
8 }
```

Segmento de Código 2.12: Método `Eliminar_PR` en C#

2.8.10 Modificar_Reservas.aspx

Este formulario permite la creación, modificación o eliminación de una reserva. Para la creación de una reserva se requiere de un cliente existente, luego se requiere ingresar las fechas desde y hasta de la reserva, para proceder a buscar las habitaciones disponibles en las fechas seleccionadas.

Una vez se han buscado las habitaciones disponibles, se procede a asignar la cantidad de adultos y menores que se alojarán en la habitación seleccionada. Luego se procede a realizar la verificación de la existencia del cliente, verificar que las fechas desde y hasta sean correctas y que la cantidad de personas ingresadas no supere la capacidad de la habitación. Una vez validados estos parámetros, se crea la reserva por medio del procedimiento almacenado `sp_c_reserva`.

Para realizar la modificación de una reserva, se muestra el panel de búsqueda que se puede observar en la Figura 2.41; una vez seleccionado el parámetro de búsqueda de reservas se procede a seleccionar la reserva que se desea, o en el caso de búsqueda por identificador, los datos de la reserva se cargarán automáticamente, siempre y cuando exista la reserva con el identificador ingresado.

Figura 2.41: Opciones de búsqueda de una reserva

Los nuevos datos de la reserva seleccionada se validan de igual manera que en la creación de una reserva y se procede a la modificación mediante el procedimiento almacenado `sp_m_reserva`.

Si se desea eliminar la reserva seleccionada se realiza el mismo procedimiento que en la eliminación de una posible reserva, en el formulario `Confirmar_Reservas.aspx`; con la diferencia que el método `Eliminar_R` se encuentra ahora en el formulario `Modificar_Reserva.aspx`, y éste hace uso del procedimiento almacenado `sp_e_reserva_id`.

2.8.11 Registro_Entrada.aspx

Este formulario permite realizar el registro de entrada al hotel por parte de un cliente, para ello es necesario que el cliente que desea registrarse posea una reserva de habitación, por lo tanto, se carga un *DropDownList* con todas las reservas cuya fecha sea mayor a la fecha actual.

Una vez seleccionada la reserva del cliente se procede a ingresar la fecha y hora de ingreso; por defecto se ingresa la hora del servidor. También se ingresa la cantidad de huéspedes que se hospedarán en la habitación asignada.

Al seleccionar la cantidad de huéspedes se habilita el panel huésped, donde se ingresan los datos definidos en el análisis de requerimientos. Para poder almacenar la información se verifican tanto los datos del registro de entrada, como los datos de los huéspedes; luego de verificar los datos se procede a la creación del registro de entrada mediante el procedimiento almacenado `sp_c_registro`.

Para la creación de los huéspedes se utiliza el procedimiento almacenado `sp_c_huesped` el cual se puede observar en el Segmento de Código 2.2, este procedimiento verifica si es que existe el huésped en la base de datos mediante la comprobación del documento de identidad, si existe procederá a actualizar los datos, si no existe procederá a crear un nuevo huésped.

Para poder llenar los datos de huéspedes ya registrados en el hotel de una manera más rápida se añadió la búsqueda automática de huésped por documento, esto se lo realiza con el evento *TextChanged* del campo de texto documento.

2.8.12 Modificar_Registro.aspx

Este formulario permite la modificación de los registros de entrada realizados y de los huéspedes asociados a dicho registro, por lo tanto, es necesario cargar un *DropDownList* con todos los registros que no posean un registro de salida. Se selecciona un registro de la lista para cargar los datos correspondientes, al igual que todos los datos de los huéspedes que se registraron.

Se procede a modificar los datos deseados del registro y de los huéspedes para luego validar dichos datos mediante el uso de validadores de ASP.NET, una vez comprobado que los datos ingresados son válidos se almacenan los datos correspondientes al registro mediante el uso del procedimiento almacenado `sp_m_registro`.

Para modificar el registro de los huéspedes se procede a comprobar la existencia de los datos del huésped en la base de datos, en caso de no encontrarse registrado se crea un nuevo huésped mediante el procedimiento almacenado `sp_c_huesped`, si el huésped ya se encuentra registrado se procede a realizar las modificaciones correspondientes mediante el procedimiento almacenado `sp_m_huesped`.

2.8.13 Registro_Salida.aspx

Para la realización de un registro de salida por parte de un cliente es necesario obtener los registros de entrada los cuales se cargan en un *DropDownList*, una vez seleccionado el registro se permiten ingresar la fecha y hora de salida, luego se valida los datos mediante un *customValidator,* para proceder a almacenarlos en la base de datos haciendo uso del procedimiento almacenado `sp_c_registro_salida`.

2.8.14 Comprobante.aspx

La etapa final del proceso de registro de entrada-salida corresponde a la creación de un comprobante de pago, con los datos correspondientes al registro de entrada-salida del cliente; para ello es necesario cargar todos los registros que tengan una fecha desde mayor a la fecha actual en un *DropDownList*.

Una vez seleccionado el registro del cual se desea crear un comprobante, se verifica que dicho registro no posea ya un comprobante de pago; si no es el caso, se calcula el tiempo de estadía basándose en la fecha y hora de ingreso al hotel y la fecha y hora de

salida del hotel. Considerando que la hora de salida es al mediodía y que si se supera cinco horas dicha hora de salida se añadirá un día extra de estadía.

También se permite agregar elementos extra al comprobante de pago, con un máximo de cinco, los cuales contienen la descripción, el costo y la cantidad; para ello se valida que las cantidades y costos contengan el formato correcto mediante validadores ASP.NET.

Después de realizar la creación de un comprobante de pago en la base de datos, se crea un archivo PDF con todos los elementos necesarios del comprobante de pago, para luego adjuntar dicho archivo en un correo electrónico y enviarlo al cliente. En el caso de ser necesario se puede reenviar dicho archivo, seleccionando un registro que posea ya un comprobante de pago.

2.8.15 Reporte_Reservas.aspx

El reporte de las reservas realizadas se lo crea en un período determinado, haciendo uso de *ReportViewer,* el cual usa el archivo `Reporte_Reservas.rdlc` que contiene el formato y diseño del reporte, incluyendo los datos que se pueden observar en la Figura 2.42.

Figura 2.42: Diseño del reporte de reservas

2.8.16 Reporte_Registros.aspx

El reporte de los registros realizadas posee un formato similar al reporte de reservas, en este reporte se utiliza el archivo `Reporte_Registros.rdlc`, que contiene el formato y diseño del reporte, este incluye los datos que se pueden observar en la Figura 2.43.

Figura 2.43: Diseño del reporte de registros

2.9 IMPLEMENTACIÓN DEL SISTEMA EN AWS

Una vez comprobado el funcionamiento del sistema de manera local, se procede a la implementación del mismo en un ambiente *cloud,* para ello se utiliza una cuenta gratuita proporcionada por AWS. De la cuenta mencionada se utilizarán los servicios de RDS, EC2 y S3. En la Figura 1.1 se puede observar un esquema del funcionamiento de dichos servicios.

2.9.1 Creación de una instancia RDS

Para la creación de una base de datos en *Amazon Web Services* es necesario primero crear una instancia de un servidor RDS. En el presente trabajo de titulación se creó una instancia *SQL Server* como se puede observar en la Figura 2.44.

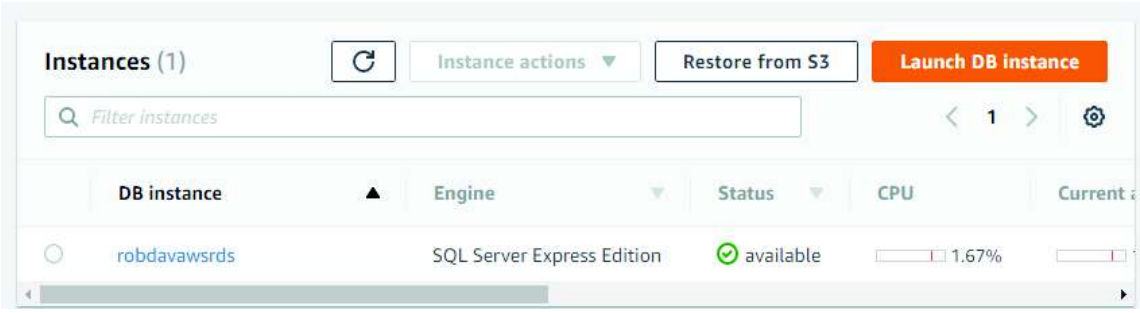

Figura 2.44: Instancia creada en RDS

La instancia SQL Server es del tipo *Express Edition*, que es una versión gratuita que presenta limitaciones en el espacio de almacenamiento, específicamente 20 GB, como se puede observar en la Figura 2.45.

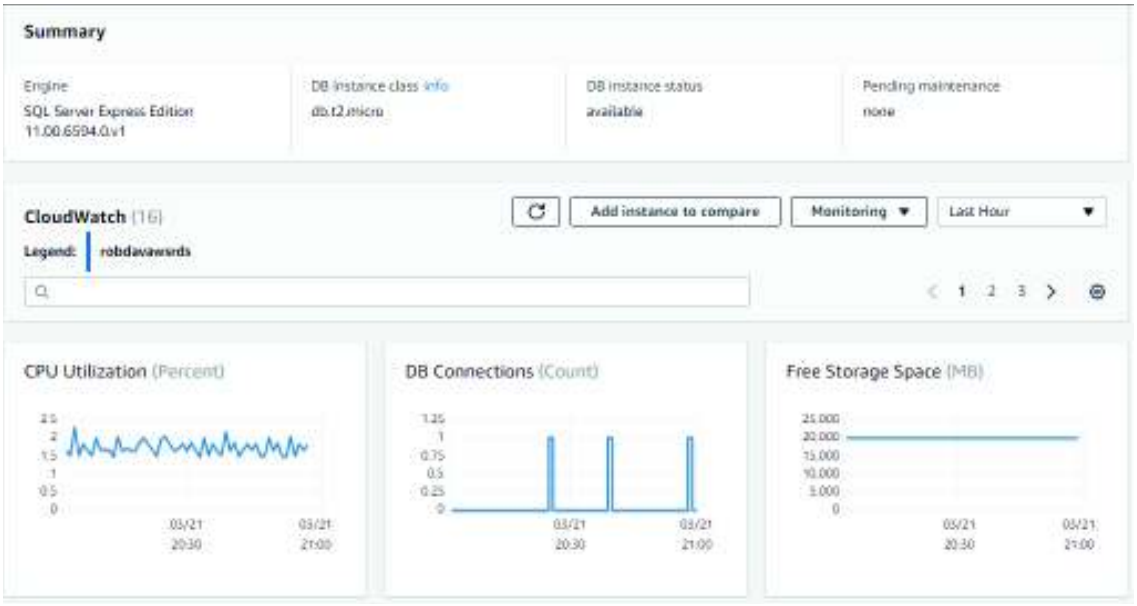

Figura 2.45: Resumen de la instancia *SQL Server Express*

Una vez creada la instancia se realiza la conexión a *SQL Server Management* mediante un *endpoint*, un puerto *TCP*; que en este caso es el puerto por defecto 1433 y una contraseña configurada en el momento de la creación de la instancia.

Una vez conectada la instancia en RDS*,* se procede a la ejecución del *script* creado en la etapa de creación de la base de datos local; con dicho *script* se obtiene una base de datos completamente funcional implementada en RDS.

Al crear la base de datos en RDS*,* la cadena de conexión cambia, por lo tanto es necesario actualizarla en el archivo de configuración de *ASP.NET*, para ello se ingresa el Segmento de Código 2.13, en donde se puede observar que se posee un usuario y contraseña; dichos datos corresponden a los configurados al momento de realizar la creación de la instancia en el administrador de RDS.

```
<connectionStrings>
    <add name="SistemaHotelConnectionString" connectionString="
  Source=robdavawsrds.clqpfaegojom.us-west-2.rds.amazonaws.com,1433;
  Initial Catalog=SistemaHotel;User ID=RobDav;Password=********"
  providerName="System.Data.SqlClient" />
</connectionStrings>
```

Segmento de Código 2.13: Creación de la página maestra

2.9.2 Creación de un servidor web en AWS

Mediante el uso del servicio *Elastic Beanstalk* se realiza la creación de una instancia EC2*,* como se puede observar en la Figura 2.46; la instancia seleccionada es *Windows Server 2016*, en el cual automáticamente *Elastic Beanstalk* implementa un servidor IIS, cuya versión seleccionada corresponde a la 10.0.

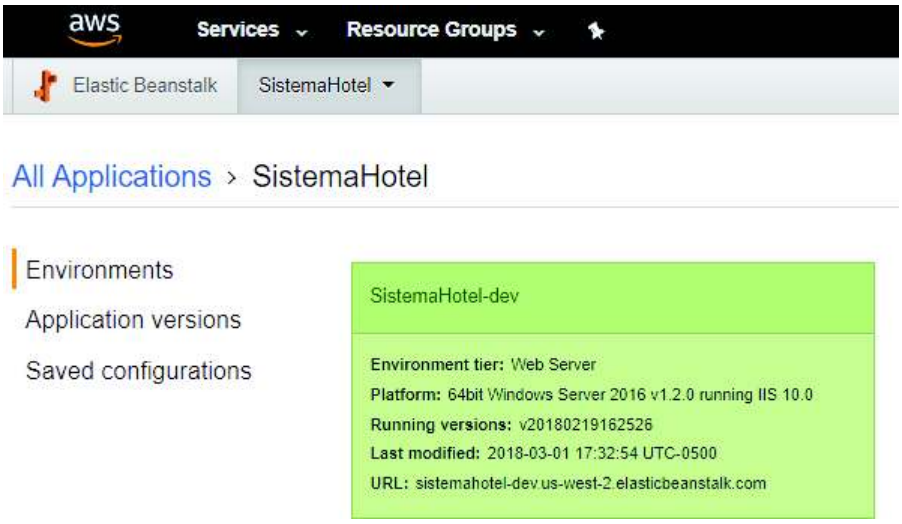

Figura 2.46: Servidor *Windows* creado en EC2

Una vez creada la instancia EC2*,* se puede observar el estado que presenta mediante el panel, en el cual se muestran varias características del servidor. Las características de la instancia EC2 creada se pueden observar en la Figura 2.47.

Entre las características mencionadas se encuentra el estado de la instancia, la plataforma utilizada, la versión de *ASP.NET* que está corriendo, la última fecha de modificación y la *URL* que se utiliza para ingresar a la aplicación.

Figura 2.47: Estadísticas de la instancia EC2

Los archivos que se deseen almacenar en el servidor creado se deben colocar en un *bucket* de S3, *Elastic Beanstalk* crea de manera automática dicho *bucket* como se puede

observar en la Figura 2.48.

Figura 2.48: *Bucket* de S3 creado por *Elastic Beanstalk*

Una vez creado el *bucket*, el servidor se encuentra listo para poder desplegar la aplicación web, para ello se utilizó un *plugin* para *Visual Studio 2017* de AWS, el cual permite publicar la aplicación directamente a la instancia EC2 creada. En la Figura 2.49 se puede observar cómo se almacenan las versiones publicadas desde *Visual Studio* a AWS.

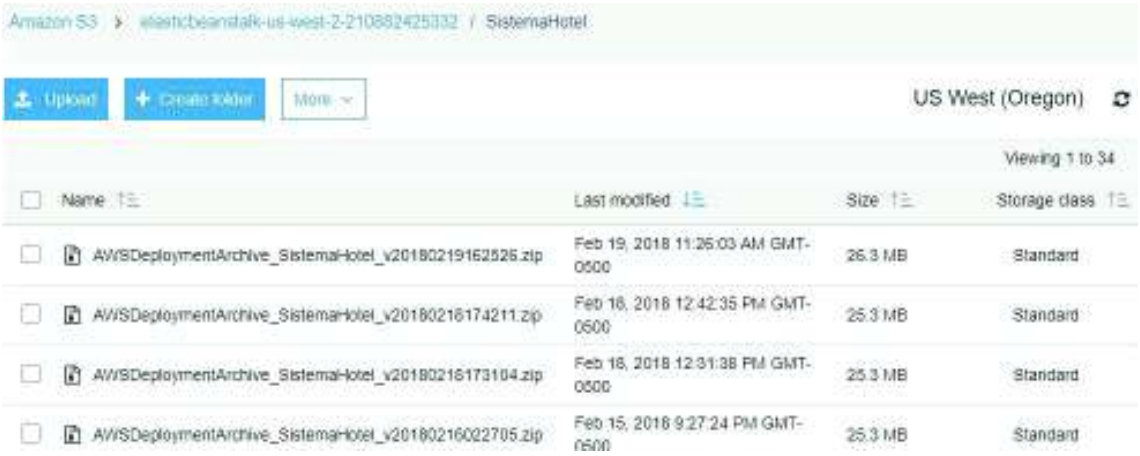

Figura 2.49: Versiones desplegadas desde *Visual Studio* a S3

Cada versión almacenada en el *bucket* de S3 se puede recuperar, en caso de existir alguna falla en las demás versiones; siempre y cuando no sobrepase el espacio otorgado por el servicio.

2.9.3 Aplicación web implementada en AWS

Una vez finalizada la creación de cada una de las instancias se procedió a desplegar la aplicación mediante el *plugin* de *Elastic Beanstalk*, en la Figura 2.50 se puede observar la página principal funcionando.

Figura 2.50: Página principal funcionando en AWS

3 RESULTADOS Y DISCUSIÓN

En este capítulo se describen los resultados de las pruebas de funcionamiento y de integración realizadas al sistema, al igual que el análisis de dichos resultados.

3.1 RESULTADOS DE LAS PRUEBAS DE FUNCIONAMIENTO

Las pruebas de funcionamiento, que se encuentran en el ANEXO III, permitieron comprobar que los procesos que posee el sistema se realicen de la manera correcta, dichas pruebas se las realizaron en el sistema implementado en AWS.

Para las funciones que requieran el control de acceso se las realizó como administrador, ya que es el que posee permiso para ingresar a todos los formularios. Al momento de realizar las pruebas se crearán todos los parámetros necesarios para realizar las pruebas y se indicarán todas las variables de la prueba.

3.1.1 Reservas Cliente

Para probar los procesos de las reservas de una habitación realizadas por un cliente se utilizó el formulario `Reservas.aspx` y no es necesario ningún prerrequisito para este proceso. En la Tabla 3.1 se observa las características de la prueba de funcionamiento realizada.

3.1.2 Control de acceso

El control de acceso requiere de la creación de un usuario con permiso para ingresar al sistema. Para realizar la creación de dicho usuario se utiliza el formulario de `Login.aspx`. En la Tabla 3.2 se muestra la prueba de funcionamiento realizada para comprobar el proceso de creación de un usuario.

Una vez creado el usuario de prueba, el cual corresponde a un prerrequisito para la siguiente prueba de funcionamiente, se procede a realizar el procedimiento de modificación de datos en Mi_Perfil, en la Tabla 3.3 se muestra la prueba de modificación de usuario.

Tabla 3.1: Prueba de funcionamiento de reservas

Caso de prueba	Prueba de funcionamiento reservas cliente	
Caso de prueba #	1	
Variables	**Proceso**	**Resultado**
Nombre: Cliente de Prueba **Correo:** correo@prueba.com **Teléfono 1:** 654321 **Teléfono 2:** 123456 **Fecha desde:** 21/05/2018 **Fecha hasta:** 25/05/2018	En el formulario de reservas para clientes se procede a crear la reserva con los datos indicados. Luego se ingresa al formulario de confirmación de reservas para comprobar que la reserva se creó.	Creación de reserva correcta.
Habitación 1 **Tipo:** Familiar **Cantidad adultos:** 2 **Cantidad menores:** 2		Visualización en el formulario de confirmación de reserva correcta.
Habitación 2 **Tipo:** Matrimonial **Cantidad adultos:** 2 **Cantidad menores:** 0		
Comentario: Prueba de funcionamiento de reservas		

Tabla 3.2: Prueba de funcionamiento administración usuarios

Caso de prueba	Prueba de funcionamiento de administración de usuarios	
Caso de prueba #	2	
Variables	**Proceso**	**Resultado**
Nombre: Usuario de Prueba **Correo:** usuario@prueba.com **Tipo:** Administrador **Alias:** Usuario de Prueba **Contraseña:** ClavePrueba	Como administrador se ingresa al formulario de administración de usuarios y se crea el usuario de prueba con los datos que se indican. Realizar el ingreso al sistema con los datos del nuevo usuario	Creación de usuario correcto.
		Ingreso al sistema correcto.

Tabla 3.3: Prueba de funcionamiento de modificación de usuarios

Caso de prueba	Prueba de funcionamiento modificación usuarios	
Caso de prueba #	3	
Variables	**Proceso**	**Resultado**
Nombre: Usuario de Prueba **Correo:** usuario@prueba.com **Tipo:** Administrador	Ingresar con el usuario de prueba a *Mi Perfil* y modificar el alias y la contraseña, una vez modificado cerrar sesión e ingresar con los nuevos datos al sistema.	Modificación del alias y la contraseña correcta.
Nuevo Alias: UsuarioPrueba **Nueva Contraseña:** 12345		Ingreso con el nuevo alias y nueva contraseña correcto.

3.1.3 Administración

Para realizar la prueba de funcionamiento correspondiente a la administración de habitaciones es necesario como prerrequisito que en la base de datos se encuentren creados los tipos de habitaciones existentes en el hotel, ya que estos no se pueden crear mediante el sistema, solo modificar. En la Tabla 3.4 se puede observar la prueba realizada a la administración de habitaciones.

Tabla 3.4: Prueba de funcionamiento de administración de habitaciones

Caso de prueba	Prueba de funcionamiento de administración de habitaciones	
Caso de prueba #	4	
Variables	**Proceso**	**Resultado**
Código: P-01 **Tipo habitación:** Familiar alta **Estado:** Activa	Se ingresa como administrador al formulario de administración de habitaciones, se procede a crear la habitación de prueba y luego se procede a realizar la modificación del estado.	Creación de la habitación correcta.
Nuevo Estado: Clausurada		Modificación de la habitación correcta.

Para la realización de la prueba de funcionamiento de la administración de clientes no se requiere de ningún prerrequisito. La Tabla 3.5 muestra la prueba de funcionamiento

realizada.

Tabla 3.5: Prueba de funcionamiento administración clientes

Caso de prueba	Prueba de funcionamiento de administración de clientes	
Caso de prueba #	5	
Variables	**Proceso**	**Resultado**
Nombre: Cliente 01 **Identificación:** 01 **Dirección:** Quito **Correo:** cliente@prueba.com **Teléfono 1:** 123456 **Teléfono 2:** 123456 **Nacionalidad:** Ecuatoriano **Género:** Masculino **Fecha Nacimiento:** 22/03/1990	Crear un nuevo cliente con los datos indicados en el formulario de administración de clientes. Una vez creado modificar el nombre del cliente y su identificación.	Creación de usuario correcto.
Nuevo Nombre: Cliente de Prueba **Nueva Identificación:** 123456789		Modificación del cliente correcta.

3.1.4 Reservas

Las reservas poseen dos formularios `Confirmar_Reservas.aspx` *y* `Modificar_Reservas.aspx`. Para realizar la prueba de funcionamiento de una confirmación de una reserva es necesario el prerrequisito de creación de una posible reserva en el formulario `Reservas.aspx`, una vez creada la posible reserva se realiza la prueba como se muestra en la Tabla 3.6.

La Tabla 3.7 muestra la prueba de funcionamiento realizada a la creación y modificación de reservas mediante el formulario `Modificar_Reservas.aspx`.

3.1.5 Registros de entrada-salida

Para el proceso de registro de entrada-salida de un cliente del hotel se utilizan cuatro formularios. Para realizar las pruebas de funcionamiento correspondientes al formulario de registro de entrada es necesario poseer una reserva confirmada como prerrequisito. En la Tabla 3.8 se muestra la prueba de funcionamiento realizada al registro de entrada.

Tabla 3.6: Prueba de funcionamiento de confirmación de reservas

Caso de prueba	Prueba de funcionamiento de confirmación de reservas	
Caso de prueba #	6	
Variables	**Proceso**	**Resultado**
Nombre: Cliente de Prueba **Identificación:** 123456789 **Fecha desde:** 21/05/2018 **Fecha hasta:** 25/05/2018	En el formulario de confirmación de reservas se procede a seleccionar la posible reserva creada y asignar las habitaciones correspondientes.	Confirmación de reserva correcta.
Habitación 1 **Tipo:** Familiar **Cantidad adultos:** 2 **Cantidad menores:** 2		
Habitación 2 **Tipo:** Matrimonial **Cantidad adultos:** 2 **Cantidad menores:** 0		

Una vez realizada la prueba de funcionamiento para el registro de entrada se procede a realizar la prueba para la modificación del registro de prueba creado en la prueba anterior, la cual es un prerrequisito. En la Tabla 3.9 se muestran los resultados obtenidos.

Para realizar la prueba de funcionamiento del registro de salida se requiere como prerrequisito la creación de un registro de entrada, en la Tabla 3.10 se muestran los resultados obtenidos en dicha prueba.

Para la prueba de funcionamiento del comprobante de pago se requiere como prerrequisito tener creado un registro de entrada-salida, en la Tabla 3.11 se muestra la prueba realizada.

3.1.6 Reportes

Para comprobar el funcionamiento del reporte de reservas es necesario como prerrequisito tener al menos 2 reservas confirmadas dentro del período correspondiente al 21/05/2018 hasta el 25/05/2018, en la Tabla 3.12 se puede observar la prueba de funcionamiento realizada.

Tabla 3.7: Prueba de funcionamiento crear/modificar reservas

Caso de prueba	Prueba de funcionamiento de creación y modificación de reservas	
Caso de prueba #	7	
Variables	**Proceso**	**Resultado**
Crear **Nombre:** Cliente de Prueba **Identificación:** 123456789 **Fecha desde:** 21/05/2018 **Fecha hasta:** 25/05/2018	En el formulario de modificación de reservas se procede a crear una reserva de prueba con los datos indicados y luego se modificará dicha reserva.	Creación de reserva correcta.
Habitación 1 **Tipo:** Familiar Alta **Cantidad adultos:** 2 **Cantidad menores:** 0		Modificación de reserva correcta.
Modificar **Fecha desde:** 23/05/2018 **Fecha desde:** 27/05/2018 **Habitación 1** **Tipo:** Familiar Alta **Cantidad adultos:** 1 **Cantidad menores:** 1		

Para comprobar el funcionamiento del reporte de los registros es necesario como prerrequisito tener al menos 2 registros de entrada-salida dentro del período correspondiente al 21/05/2018 hasta el 25/05/2018, en la Tabla 3.13 se puede observar la prueba de funcionamiento realizada.

3.2 PRUEBAS DE INTEGRACIÓN

Para realizar las pruebas de integración se dividió en tres partes los procesos. Cada proceso depende del otro, de esta manera se comprueba el funcionamiento completo del sistema, las tres partes corresponden a:

- **Control de acceso** Se creará un usuario y se comprobará el acceso al sistema.
- **Reservas de habitaciones** Se creará una posible reserva y con el usuario creado en la prueba de control de acceso se realizarán las reservas.
- **Registro de entrada-salida** Con la reserva creada en la prueba de reserva de habitaciones se procede a realizar el registro de entrada-salida.

Tabla 3.8: Prueba de funcionamiento de registro de entrada

Caso de prueba	Prueba de funcionamiento de registro de entrada	
Caso de prueba #	8	
Variables	**Proceso**	**Resultado**
Nombre: Cliente de Prueba **Identificación:** 123456789 **Fecha ingreso:** 23/05/2018 **Hora ingreso:** 11:50	Se ingresa al formulario de registro de entrada para seleccionar la reserva de prueba. Luego se asigna al registro de entrada el cliente, la fecha y hora de ingreso al hotel y los datos de los huéspedes.	Registro de entrada correcto.
Huésped 1 **Nombre:** Cliente de prueba **Identificación:** 123456789 **Género:** Masculino **Nacionalidad:** Ecuatoriano **Fecha Nacimiento:** 22/03/1990		
Huésped 2 **Nombre:** Huésped Prueba **Identificación:** 987654321 **Género:** Femenino **Nacionalidad:** Ecuatoriano **Fecha Nacimiento:** 11/02/2008		

Los resultados de las pruebas de integración se encuentran disponibles en el ANEXO III.

3.2.1 Control de acceso

Para la realización de la prueba de integración del control de acceso se posee como prerrequisito la creación de un usuario administrador del sistema. En la Tabla 3.14 se puede observar la prueba de integración realizada para este proceso.

3.2.2 Reserva de habitaciones

Una vez creado el usuario de prueba, se procede a realizar el ingreso al sistema con dicho usuario para proceder a la siguiente prueba la cual corresponde a las reservas de habitaciones. En la Tabla 3.15 se puede observar la prueba de integración realizada al proceso mencionado.

Tabla 3.9: Prueba de funcionamiento de modificación de registro

Caso de prueba	Prueba de funcionamiento de modificación de registro de entrada	
Caso de prueba #	9	
Variables	**Proceso**	**Resultado**
Nombre: Cliente de Prueba **Identificación:** 123456789 **Fecha ingreso:** 23/05/2018 **Nueva hora ingreso:** 13:50	Se debe ingresar al sistema como administrador, luego se ingresa al formulario de modificación de los registros de entrada, se procede a seleccionar el registro de prueba y se modifica los datos indicados en las variables.	Modificación de registro de entrada correcto
Huésped 1 **Nombre:** Cliente de prueba **Identificación:** 123456789 **Género:** Masculino **Nacionalidad:** Ecuatoriano **Fecha Nacimiento:** 22/03/1990		
Nuevo Huésped 2 **Nombre:** Huésped Prueba **Identificación:** 987654333 **Género:** Femenino **Nacionalidad:** Ecuatoriano **Fecha Nacimiento:** 22/03/2008		

Tabla 3.10: Prueba de funcionamiento de registro de salida

Caso de prueba	Prueba de funcionamiento de registro de salida	
Caso de prueba #	10	
Variables	**Proceso**	**Resultado**
Nombre: Cliente de Prueba **Identificación:** 123456789 **Fecha salida:** 27/05/2018 **Hora salida:** 11:59	Se debe ingresar al formulario de registro de salida, seleccionar el registro de entrada de prueba y asignar la fecha y hora de salida.	Registro de salida correcto.

3.2.3 Registro de entrada-salida

Los resultados de la prueba de integración, realizada al proceso de registro de entrada-salida, se los puede observar en la Tabla 3.16.

Para la realización de dicha prueba fueron necesarios los siguientes prerrequisitos:

- El usuario creado en la prueba de integración de control de acceso
- El cliente creado en la prueba de integración de reserva de habitaciones.
- La reserva creada en la prueba de integración de reserva de habitaciones.

Tabla 3.11: Prueba de funcionamiento de comprobante de pago

Caso de prueba	Prueba de funcionamiento de comprobante de pago	
Caso de prueba #	11	
Variables	**Proceso**	**Resultado**
Nombre: Cliente de Prueba **Identificación:** 123456789 **Cantidad completo:** 1 **Cantidad descuento:** 1	Se ingresa al formulario de los comprobantes de pago, en donde se selecciona el registro de prueba, se añaden los extras indicados y se crea el comprobante. Luego se envía el comprobante por correo electrónico.	Creación de comprobante correcta.
Extra 1 **Descripción:** Servicio habitación **Cantidad:** 1 **Costo:** 25,50		Envío de comprobante correcto.
Extra 2 **Descripción:** Alquiler balón **Cantidad:** 2 **Costo:** 1,25		

Tabla 3.12: Prueba de funcionamiento de reporte de reservas

Caso de prueba	Prueba de funcionamiento de reporte de reservas	
Caso de prueba #	12	
Variables	**Proceso**	**Resultado**
Fecha desde: 21/05/2018 **Fecha hasta:** 28/05/2018	Se ingresa en el formulario de los reportes de reservas, se cambian las fechas desde y hasta a las fechas que se indican en las variables y se genera el reporte.	Creación de reporte de reservas correcto.

Tabla 3.13: Prueba de funcionamiento de reporte de registros

Caso de prueba	Prueba de funcionamiento de reporte de registros	
Caso de prueba #	13	
Variables	**Proceso**	**Resultado**
Fecha desde: 21/05/2018 **Fecha hasta:** 28/05/2018	Se ingresa en el formulario de los reportes de registros, se cambian las fechas desde y hasta a las fechas que se indican en las variables y se genera el reporte.	Creación de reporte de registros correcto.

Tabla 3.14: Prueba de integración de control de acceso

Caso de prueba	Prueba de integración de control de acceso	
Caso de prueba #	14	
Formularios	**Proceso**	**Resultado**
Login Admin_Usuarios Mi_Perfil	El administrador debe ingresar al formulario de administración de usuarios y crear un usuario de prueba, luego se cierra sesión y se ingresa al formulario de *Login*, en donde se ingresa con los datos del nuevo usuario. Luego se procede a ingresar a *Mi Perfil* para modificar el alias y la contraseña. Se cierra sesión y en el formulario *Login* se ingresa el nuevo alias y contraseña para finalizar la prueba.	Creación de usuario correcta. Ingreso de nuevo usuario correcto. Modificación de usuario correcta.

3.3 ANÁLISIS DE LOS RESULTADOS DE LAS PRUEBAS DE FUNCIONAMIENTO

A continuación, se realiza el análisis de los resultados obtenidos en las pruebas de funcionamiento.

3.3.1 Reservas Cliente

Al realizar la prueba de una reserva de habitaciones por parte de un cliente no se encontró ningún error en el funcionamiento del mismo, pero si se pudo notar ciertas interacciones que se podrían mejorar como por ejemplo las validaciones de los datos ingresados en los campos del formulario solo se los realiza al presionar el botón

Tabla 3.15: Prueba de integración de reservas

Caso de prueba	Prueba de integración de reservas	
Caso de prueba #	15	
Formularios	**Proceso**	**Resultado**
Reservas Confirmar_Reservas Modificar_Reservas Admin_Clientes Reporte_Reservas	En el formulario de reservas se crea una posible reserva de dos habitaciones. Se accede al formulario de confirmación de reservas, en donde se visualiza la posible reserva, y ahí se crea un cliente con los datos del cliente de la posible reserva. Luego se selecciona la posible reserva y se realiza la confirmación de la misma. Se ingresa al formulario de modificación de reservas y se selecciona las reservas para proceder a asignar dos habitaciones diferentes. Finalmente se genera un reporte de las reservas de prueba.	Creación de posible reserva correcta. Creación de cliente correcto. Confirmación de reserva correcta. Modificación de reserva correcta. Generación de reporte correcto.

Reservar lo cual obliga al usuario a revisar que campos se encuentran incorrectos luego de presionar el botón, cuando se podría realizar la revisión al momento de ingresar los datos automáticamente.

3.3.2 Control de acceso

Luego de realizar la prueba del control de acceso no se encontraron errores en el funcionamiento del proceso, la creación de los usuarios por parte de un administrador se la puede realizar correctamente con los datos indicados, al igual que la modificación de dichos datos. Una vez creado el usuario este puede cambiar sus datos correctamente con las restricciones indicadas. Sin embargo, se podría mejorar la forma de mostrar los datos ya que se utiliza un *input* de solo lectura, cuando se podría utilizar un *label*.

3.3.3 Administración

Los procesos de administración corresponden a la administración de las habitaciones y a la administración de los clientes ya que la administración de los usuarios se consideró

en el control de acceso.

Tabla 3.16: Prueba de integración de registros

Caso de prueba	Prueba de integración de registros entrada-salida	
Caso de prueba #	16	
Formularios	**Proceso**	**Resultado**
Registro_Entrada Modificar_Registro Registro_Salida Comprobante Reporte_Registro	En el formulario de registros de entrada se selecciona la reserva de prueba y se asigna una fecha y hora de ingreso y los datos de un huésped. En el formulario de modificación de registro de entrada se selecciona el registro de prueba creado y se procede a cambiar los datos de hora y del huésped. En el formulario de registro de salida se selecciona el registro que se acaba de modificar para proceder a asignar la fecha y hora de salida. En el formulario de comprobante de pago se crea el mismo con el registro de prueba y se realiza el envío por correo. Finalmente se genera un reporte del registro de prueba.	Creación de registro de entrada correcto. Creación de huésped correcta. Modificación de registro correcta. Creación de registro de salida correcta. Creación de comprobante correcta. Generación de reporte correcta.

a. Administración de habitaciones

El proceso de administración de habitaciones comprende la creación de habitaciones, modificación de habitaciones y la modificación del tipo de habitación para cada uno de esos procesos las pruebas de funcionamiento realizadas fueron exitosas. En cuanto a la modificación de una habitación se encontró que al momento de ingresar el código se podría dar el formato especificado por el hotel por lo que se realizó dicha modificación.

En lo que se refiere a la modificación de un tipo de habitación se podría permitir la creación de un nuevo tipo de habitación en caso de que dicho tipo de habitación se

construya en el hotel, pero ya que no se tomó en consideración modificaciones de la estructura del hotel no se lo realizó en el presente trabajo de titulación.

b. Administración de clientes

La prueba realizada a la creación y modificación de un cliente se la realizó sin encontrar errores de funcionamiento. En cuanto a la creación de un cliente se podría mejorar la visualización de que proceso se está realizando, ya sea el de crear o modificar un cliente, ya que puede modificarse un cliente por error con datos de un cliente que se está creando.

3.3.4 Reservas de habitaciones

Las pruebas de funcionamiento realizadas al proceso de confirmación de reservas no tuvieron errores, en cuanto a la prueba realizada a la modificación de la reserva tampoco se encontraron errores. Para poder modificar una reserva se lo tiene que realizar seleccionando individualmente cada reserva, pero se podría mejorar seleccionando todas las reservas de un cliente, de una misma fecha, en conjunto para modificarlas.

3.3.5 Registros de entrada-salida

Al realizar las pruebas de funcionamiento de los registros de entrada, modificación de registros, el registro de salida y la generación de un comprobante no se encontraron errores.

Debido a que el registro de los huéspedes se lo realizará a varias personas y existe la posibilidad de que se ingresen datos erróneos se podría mejorar dicho proceso mediante las validaciones inmediatas de los datos ingresados en los campos y de esta manera evitar tener que solicitar nuevamente la información. De igual manera en el caso de la modificación del registro de entrada.

El comprobante de pago que se crea con los datos de los registros de entrada-salida, y los extras que se hayan indicado, se lo podría mejorar para que cumpla con los requisitos solicitados por el SRI y de esta manera ingresar al sistema de facturación electrónica implementado en el país.

3.3.6 Reportes

En las pruebas de funcionamiento realizadas a los procesos de generación de reportes de reservas y registros no se encontraron errores, sin embargo, se observó que los datos que se muestran en cada uno de los registros podrían ser más detallados en cada uno de los reportes. De igual manera se observó que al momento de realizar la exportación a PDF del reporte éste contenía páginas en blanco.

3.4 ANÁLISIS DE LOS RESULTADOS DE LAS PRUEBAS DE INTEGRACIÓN

A continuación, se realiza el análisis de los resultados obtenidos en las pruebas de funcionamiento.

3.4.1 Control de acceso

La prueba de integración realizada al proceso de control de acceso se realizó con éxito teniendo como resultado la creación de un usuario. El proceso se podría mejorar al momento de realizar cualquier acción en la administración de clientes o *Mi Perfil* desconectando al usuario que se modificó por motivos de seguridad.

Como se mencionó en el análisis de requerimientos el envío de la contraseña se lo realiza en texto plano por lo que también se podría mejorar dicho aspecto utilizando un cifrado en todas las páginas para de esta manera tener mayor seguridad en los datos.

3.4.2 Reservas de habitaciones

La prueba de integración realizada al proceso de reserva de habitaciones culminó exitosamente y como resultado se obtuvo una reserva confirmada. Se pudo observar que la confirmación de las reservas y de la creación y modificación de las mismas se lo podría realizar en un solo formulario ya que en el mismo se podría administrar todo lo relacionado a las reservas.

3.4.3 Registros de entrada-salida

La prueba de integración realizada al proceso de registros de entrada-salida finalizó sin encontrar errores. Como observación se tiene que el proceso de registro de entrada y registro de salida se podría realizar en un solo formulario ya que el registro de salida

simplemente permite añadir la fecha y hora de salida del hotel por lo que es una acción pequeña que se puede fusionar.

4 CONCLUSIONES Y RECOMENDACIONES

En este capítulo se presentan las conclusiones luego de haber finalizado el presente trabajo de titulación, al igual que las recomendaciones.

4.1 CONCLUSIONES

- El uso de las tecnologías de la información facilita al administrador de un hotel la realización de los procesos de reserva de habitaciones y registro de entrada-salida de clientes, además de permitir organizar de una mejor manera los datos recopilados en dichos procesos.
- La creación de una página web en la cual se permite realizar reserva de habitaciones en línea facilita al cliente la búsqueda de un hotel, en las fechas y con las características que desee el cliente.
- La etapa de diseño de la base de datos y el diseño visual influye considerablemente en todas las etapas siguientes, es por ello que la correcta realización del diseño del sistema agilita el proceso de desarrollo y la corrección de errores que puedan surgir.
- El uso de las herramientas de diseño UML facilitó la creación de los diagramas que comprenden el diseño del sistema en el presente trabajo de titulación se utilizó *StarUML* del tipo gratuito.
- El uso de procedimientos almacenados influyó en el diseño y el desarrollo del sistema ya que existen diferentes formas de realizar procesos que se necesitan varias veces, al usar procedimientos almacenados la programación se la realiza en SQL por lo que el manejo de datos es más sencillo.
- El lenguaje orientado a objetos C# en conjunto con ASP.NET permiten la creación de un sin número de aplicaciones, en el presente trabajo de titulación se utilizan para procesos de ingreso y lectura de datos, en caso de necesitar más características los lenguajes mencionados proporcionan las herramientas necesarias.

4.2 RECOMENDACIONES

- Al momento de realizar el análisis de requerimientos es recomendable ser lo más específico posible con la persona que conoce los procesos del sistema, y de ser necesario se debe realizar varias entrevistas hasta dejar en claro cuáles serán dichos requerimientos; de esta manera se evitan modificaciones tardías en el diseño.
- La revisión exhaustiva de los procesos de diseño del sistema puede evitar errores en las siguientes etapas de la creación del sistema, es por lo que se recomienda la realización de un diseño profundo tanto en lo funcional como en lo visual.
- Es importante realizar un código reutilizable para agilitar el desarrollo de la aplicación, ya que existen varios procesos que requieren de una misma acción. En el presente trabajo de titulación se utilizaron procedimientos almacenados para este fin.

BIBLIOGRAFÍA

[1] AWS | *Elastic compute cloud (*EC2*) de capacidad modificable en la nube*. [En línea]. Disponible en: //aws.amazon.com/es/ec2/. [Último acceso: 14-01-2018].

[2] AWS | *Almacenamiento de datos seguro en la nube (*S3*)*. [En línea]. Disponible en:
//aws.amazon.com/es/s3/. [Último acceso: 14-01-2018].

[3] AWS | *Servicio de bases de datos relacionales (*RDS*)*. [En línea]. Disponible en: / / aws.amazon.com/es/rds/. [Último acceso: 14-01-2018].

[4] *What is a Web Application?* [En línea]. Disponible en: https://www.maxcdn.com/one/ visual-glossary/web-application/?. [Último acceso: 22-01-2018].

[5] *The World Wide Web project*. [En línea]. Disponible en: https://www.w3.org/History/ 19921103-hypertext/hypertext/WWW/TheProject.html. [Último acceso: 23-01-2018].

[6] *static Web page Definition from PC Magazine Encyclopedia*. [En línea]. Disponible en: https://www.pcmag.com/encyclopedia/term/52045/static-web-page. [Último acceso: 16-05-2018].

[7] *Static Web page*. [En línea]. Disponible en: https://upload.wikimedia.org/wikipedia/
commons/5/57/Scheme_static_page_en.svg. [Último acceso: 17-05-2018].

[8] *dynamic Web page Definition from PC Magazine Encyclopedia*. [En línea]. Disponible en: https://www.pcmag.com/encyclopedia/term/42199/dynamic-web-page. [Último acceso: 16-05-2018].

[9] *WordPress Features « WordPress Codex*. [En línea]. Disponible en: https:// codex. wordpress.org/WordPress_Features. [Último acceso: 23-01-2018].

[10] D. Importare, *Tipos de Sitio Web (estático v dinámico)*, en-US, feb. de 2016. [En línea]. Disponible en: http:// blog. importare. mx/ tipos- de- sitio- web- estatico- vs-dinamico. [Último acceso: 17-05-2018].

[11] *What is the Document Object Model?* [En línea]. Disponible en: https://www.w3.org/ TR/WD-DOM/introduction.html. [Último acceso: 23-01-2018].

[12] *HTML: The Markup Language (an HTML language reference)*. [En línea]. Disponible en: https : / / www. w3 . org / TR / 2012 / WD - html - markup - 20121025/. [Último acceso: 23-01-2018].

[13] *HTML5*. [En línea]. Disponible en: https : / / developer. mozilla . org / es / docs / HTML / HTML5. [Último acceso: 23-01-2018].

[14] *Cascading Style Sheets*. [En línea]. Disponible en: https://www.w3.org/Style/CSS/ Overview.en.html. [Último acceso: 16-05-2018].

[15] *Bootstrap · The most popular HTML, CSS, and JS library in the world.* [En línea]. Disponible en: http://getbootstrap.com/. [Último acceso: 23-01-2018].

[16] *Grid system*. [En línea]. Disponible en: https://v4-alpha.getbootstrap.com/layout/grid/. [Último acceso: 23-01-2018].

[17] *Tryit Editor v3.5*. [En línea]. Disponible en: https://www.w3schools.com/bootstrap/tryit. asp?filename=trybs_jumbotron2&stacked=h. [Último acceso: 23-01-2018].

[18] *Skeleton: Responsive CSS Boilerplate*. [En línea]. Disponible en: http://getskeleton. com/. [Último acceso: 23-01-2018].

[19] JavaScript. [En línea]. Disponible en: https:// developer. mozilla. org/ es/ docs/ Web/ JavaScript. [Último acceso: 24-01-2018].

[20] j. F.-. jquery.org jQuery, *About jQuery | jQuery Learning Center*. [En línea]. Disponible en: http://learn.jquery.com/about-jquery/. [Último acceso: 24-01-2018].

[21] *jQuery UI*. [En línea]. Disponible en: https : / / jqueryui . com/. [Último acceso: 24-01-2018].

[22] *Active Server Pages*. [En línea]. Disponible en: https : / / msdn . microsoft . com / en - us/library/aa286483.aspx. [Último acceso: 06-02-2018].

[23] Rick-Anderson, *ASP.NET overview*. [En línea]. Disponible en: https://docs.microsoft. com/en-us/aspnet/overview. [Último acceso: 06-02-2018].

[24] BillWagner, *A Tour of C# - C# Guide*. [En línea]. Disponible en: https://docs.microsoft. com/en-us/dotnet/csharp/tour-of-csharp/. [Último acceso: 06-02-2018].

[25] *Introducción a las bases de datos*. [En línea]. Disponible en: http : / / es . ccm . net / contents/66-introduccion-a-las-bases-de-datos. [Último acceso: 06-02-2018].

[26] *Editions and supported features of SQL Server 2017* | Microsoft *Docs*. [En línea]. Disponible en: https : / / docs . microsoft . com / en - us / sql / sql - server / editions - and - components-of-sql-server-2017. [Último acceso: 16-05-2018].

[27] *SQL Server Language Reference*. [En línea]. Disponible en: https://technet.microsoft. com/en-us/library/ms166026(v=sql.90).aspx. [Último acceso: 06-02-2018].

[28] *Transact-SQL Reference (Transact-SQL)*. [En línea]. Disponible en: https:// technet. microsoft.com/en-us/library/ms189826(v=sql.90).aspx. [Último acceso: 06-02-2018].

[29] *JSON*. [En línea]. Disponible en: http://www.json.org/json- es.html. [Último

acceso: 24-01-2018].

[30] AWS | *Informática en la nube. Ventajas y Beneficios*. [En línea]. Disponible en: //aws.amazon.com/es/what-is-cloud-computing/. [Último acceso: 17-02-2018].

[31] *¿Qué es* AWS*? – Amazon Web Services*. [En línea]. Disponible en: //aws.amazon.com/es/what-is-aws/. [Último acceso: 17-02-2018].

[32] M. de Turismo, *Reformas al reglamento de alojamiento, Ministerio de Turismo*. [En línea]. Disponible en: https://www.turismo.gob.ec/wp-content/uploads/2016/07/18-02- 2016-Reformas-al-Reglamento-de-Alojamiento.pdf. [Último acceso: 11-04-2018].

[33] *Free templates for HTML5*. [En línea]. Disponible en: https : / / www. html5up . com. [Último acceso: 18-01-2018].

[34] *Páginas maestras ASP.NET*. [En línea]. Disponible en: https://msdn.microsoft.com/ es-ec/library/cc295533.aspx. [Último acceso: 18-02-2018].

[35] *Información general sobre el control UpdatePanel*. [En línea]. Disponible en: https://msdn.microsoft.com/es-es/library/bb386454(v=vs.100).aspx. [Último acceso: 21-02-2018].

ÍNDICE DE ANEXOS

ANEXO I ENTREVISTA

Entrevista realizada a:

Lcda. IRINA GALARRAGA, GERENTE GENERAL PLAYA CRISTAL RESORT.

1. ¿QUÉ TIPO DE HABITACIONES POSEE EL HOTEL?

Las habitaciones son del tipo suites minimalistas, en total existen 64 camas. Se tienen dos tipos de camas:

- Queens: Cuyo tamaño es de 2 plazas y media
- Camas individuales: Cuyo tamaño es de 1 plaza y media

Las habitaciones se dividen en dos tipos:

- Suite matrimonial
- Suite Familiar

Los baños son tipo moderno.

En cuanto a los extras que posee cada habitación estos son: televisión por cable, aire acondicionado e Internet.

2. ¿CUÁL ES LA CAPACIDAD DE UNA HABITACIÓN?

La capacidad de una habitación está definida por el tipo de habitación, en las suites matrimoniales la capacidad es de 2 personas y en las suites familiares 5 personas.

Cuando se hospeda un niño menor a 12 años se lo hace mediante una habitación compartida con los padres.

3. ¿CUÁNTAS HABITACIONES POSEE EL HOTEL?

El hotel posee 28 habitaciones en total, de estas 4 son del tipo suites matrimoniales y 12 suites familiares, las suites familiares se dividen en 2 habitaciones independientes, en la cual cada una posee su propio baño, como se detalla a continuación:

- Suite Familiar Alta: Posee espacio para 2 personas
- Suite Familiar Baja: Posee espacio para 3 personas.

4. ¿CÓMO SE IDENTIFICA A LAS HABITACIONES EN EL HOTEL?

Se las identifica mediante códigos para cada habitación.

Para las suites matrimoniales se utiliza la letra “M” seguida del número de habitación desde el 1 hasta el 4, ejemplo: **M-01** identifica a la habitación matrimonial 1.
Para las suites familiares se utiliza la letra “A” para identificar a la parte alta de la habitación y la letra “B” para identificar a la parte baja de la habitación, seguido del número de habitación desde el 1 hasta el 12, ejemplo **A-05** identifica a la suite familiar alta 5.

5. ¿CÓMO SE REALIZA UNA RESERVA DE HABITACIÓN EN EL HOTEL?

Las reservas se las realizan mediante contacto telefónico y se confirma dicha reserva cuando se ha depositado el 50 % del valor total.

6. ¿QUÉ DATOS SE SOLICITA AL CLIENTE AL MOMENTO DE REALIZAR UNA RESERVA DE HABITACIÓN?

Al momento de realizar una reserva se pide el nombre, teléfono celular, teléfono convencional, correo, las fechas de ingreso y de salida que desea hospedarse en el hotel, el número de personas adultas que se hospedarán y el número de personas menores a 12 años que se hospedarán.

7. ¿CÓMO SE REALIZA EL REGISTRO DE ENTRADA-SALIDA DE UN CLIENTE?

La administración en Quito envía los datos de la reserva al administrador del hotel en Pedernales, el cual recibe al cliente y realiza la acomodación en la habitación, si no posee una reserva el administrador del hotel realiza el registro de entrada solicitando los datos en ese momento.

8. ¿QUÉ DATOS SE SOLICITA AL CLIENTE PARA EL REGISTRO DE ENTRADA-SALIDA?

Se solicita el nombre, cédula de identidad, correo, teléfono celular, teléfono convencional y la edad de la persona.

9. ¿CÓMO SE CALCULA EL TIEMPO A COBRARSE POR EL HOSPEDAJE?

Se cobra por noche hospedada y si es la persona realiza el registro de salida pasadas 5 horas a la hora de salida se cobra una noche extra.

10. ¿A QUÉ HORA SE TIENE QUE SALIR DEL HOTEL? (REALIZAR REGISTRO DE SALIDA)

La hora de salida del hotel es a las 12:00 h (12:00 p.m.)

11. ¿CUÁL ES EL COSTO DE HOSPEDAJE EN EL HOTEL POR NOCHE?

Para personas mayores a 12 años el costo es de 60 dólares, para personas de la tercera edad y menores a 12 años el costo es de 40 dólares.

12. ¿CUÁNTAS PERSONAS ESTÁN ENCARGADAS DE REALIZAR EL REGISTRO DE ENTRADA-SALIDA DE LOS CLIENTES?

1 persona se encarga del registro de los clientes.

13. ¿CUÁNTAS PERSONAS SE ENCARGAN DE LAS RESERVAS DE HABITACIONES?

2 personas en Quito se encargan de realizar las reservas.

Los siguientes anexos se encuentran disponibles en el disco compacto adjunto.

ANEXO II DIAGRAMAS DEL SISTEMA

ANEXO III PRUEBAS DEL SISTEMA

ANEXO IV CÓDIGO FUENTE

ANEXO V MANUAL DE USUARIO

ORDEN DE EMPASTADO