Informe del Proyecto

1. Introducción

- Descripción general del proyecto

  Uniandes ha decidido implementar HOTEL DE LOS ANDES, una aplicación que apoye a las cadenas hoteleras en su operación diaria. Cada hotel que opera utilizando HOTEL DE LOS ANDES tiene su propia instancia de la aplicación.

- Objetivos del proyecto

  - Desarrollar habilidades en el proceso de generación de un modelo de datos relacional a partir de un modelo conceptual.
  - Incorporar elementos de calidad del modelo de datos, con respecto a integridad de la información.
  - Implementar una aplicación informática de mediana complejidad que involucre bases de datos relacionales.

2. Arquitectura del Proyecto

Para nuestro proyecto en la parte de backend nos enfocamos en tres partes puntualmente: controladores, modelo y repositorios, donde se logró implementar toda la lógica necesaria contemplando las reglas de negocio y las diferentes partes involucradas en la operación general del hotel.

Desglose de cada división:

- Controllers
  Los controllers serán los componentes encargados de manejar las solicitudes HTTP entrantes de los clientes o usuarios. Estos fueron realizados, ya que se encargarán de recibir las solicitudes, procesarlas y coordinar la lógica de negocio adecuada para manejarlas. Estos a su vez están interactuando con las entidades y repositorios para realizar operaciones de lectura o escritura en la base de datos. En resumen, están actuando como intermediarios entre la interfaz de usuario y la lógica de negocio.

  Creados en nuestro proyecto:

  BarController
  ConsumoController
  EmboladaController
  GimnasioController
  HabitacionController
  HotelController
  InternetController
  LavanderiaController
  PiscinaController
  PlanEstadiaController
  (Enumeracion creada en el UML)
  ReservaController
  RestauranteController
  SalonController
  ServicioController
  SpaController
  SupermercadoController
  TiendaController
  TieneController (Relación)
  TipoHabitacionController
  TipoUsuarioController
  UsuarioController

- Entidades
  Las entidades serán las que representan las clases diseñadas en nuestro diagrama inicialmente planteados (UML), donde se consideran todas las partes involucradas en cada hotel. Por lo tanto, cada entidad en nuestro proyecto de Java contiene sus respectivos atributos propios y las relaciones respectivas que maneja con otras clases del modelo.

  Creados en nuestro proyecto:

  Bar
  Consumo
  Contiene – ContienePK
  Embolada
  Gimnasio
  Habitación
  Hotel
  Internet
  Lavanderia
  Piscina
  PlanEstadia
  Reserva
  Restaurante
  Salón
  Servicio
  Spa
  Supermercado
  Tienda
  Tiene – TienePK
  TipoHabitacion (Enumeracion creada en el UML)
  TipoUsuario
  Usuario

- Repositorios
  Estos fueron diseñados como una capa de acceso a datos que se utiliza para interactuar con la base de datos (SQL). Además, proporcionan métodos para realizar operaciones CRUD en las entidades. En conclusión, tradujeron las operaciones del sistema de objetos a operaciones en la base de datos.

  Creados en nuestro proyecto:

  BarRepository
  ConsumoRepository
  ContieneRepository
  EmboladaRepository
  GimnasioRepository
  HabitacionRepository
  InternetRepository
  LavanderiaRepository
  PiscinaRepository
  PlanEstadiaRepository
  ReservaRepository
  RestauranteRepository
  SalonRepository
  ServicioRepository
  SpaRepository
  SupermercadoRepository
  TiendaRepository
  TieneRepository
  TipoHabitacionRepository
  TipoUsuarioRepository
  UsuarioRepository

3. Configuración

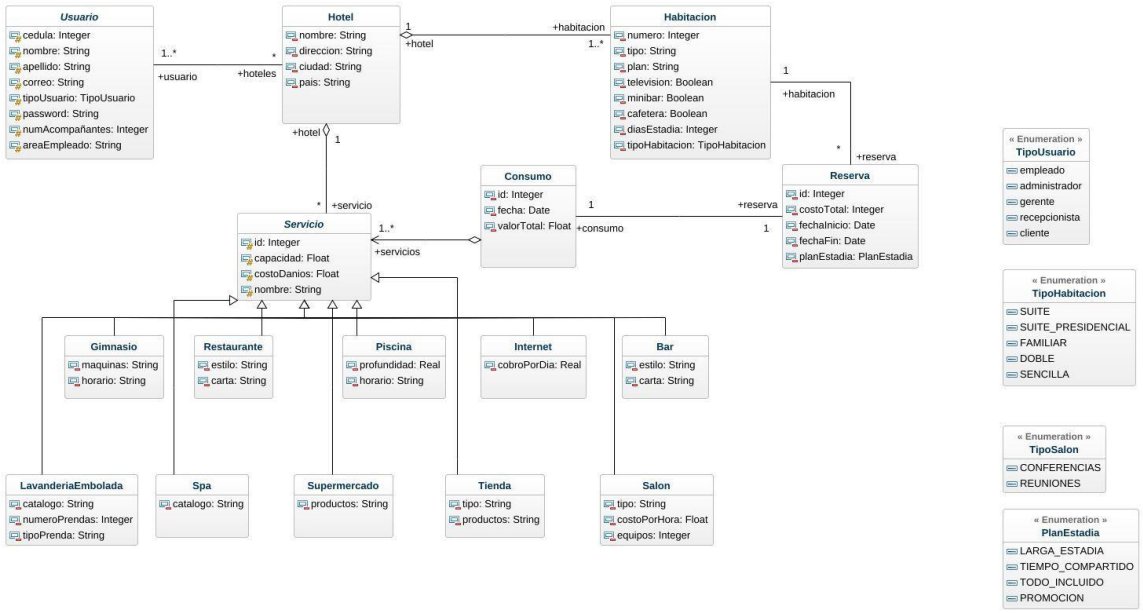
- Conexión a la base de datos en SQL.

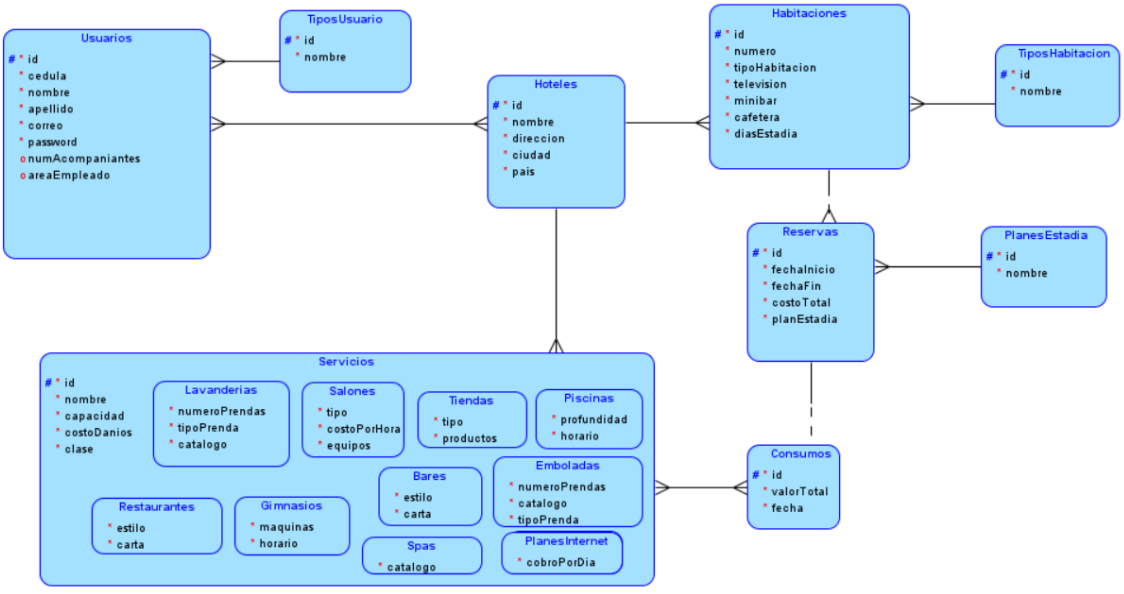
4. Documentación Técnica

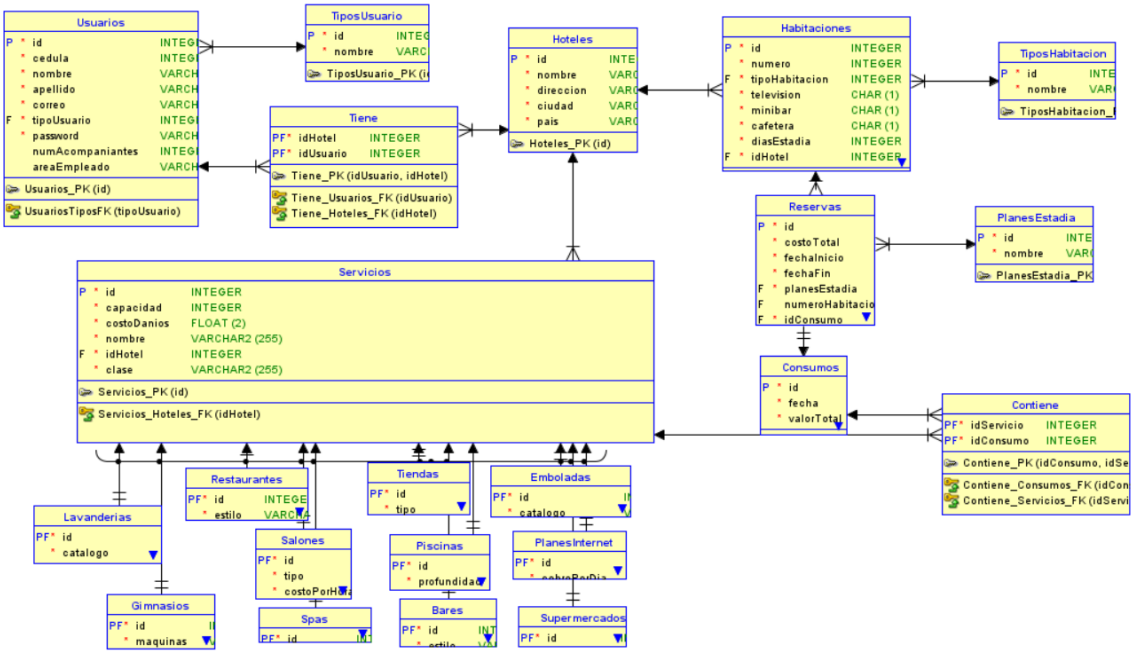
Importaciones

- Jakarta Persistence
- JPA (Java Persistence API)
- JavaUtil
- SpringFramework
- Java SQL

5. Modelos







Primera Forma Normal (1NF):

Las tablas *bares, restaurantes, emboladas, gimnasios, lavanderias, piscinas, planesinternet, salones, spas, supermercados, tiendas, tiposhabitacion, tiposusuario y usuarios* cumplen con la 1NF, ya que todos sus atributos contienen valores atómicos.

Las tablas *consumos, planesestadia, servicios, reservas, contiene, y habitaciones* también cumplen con la 1NF.

Segunda Forma Normal (2NF):

La tabla *consumos* cumple con la 2NF, ya que el atributo no primo valortotal depende completamente de la clave primaria id.

La tabla *contiene* no tiene atributos no primos, por lo que no aplicamos la 2NF.

La tabla *habitaciones* cumple con la 2NF, ya que los atributos no primos television, minibar, cafetera, diasestadia y idhotel dependen completamente de la clave primaria id.

La tabla *hoteles* también cumple con la 2NF, ya que los atributos no primos nombre, direccion, ciudad y pais dependen completamente de la clave primaria id.

La tabla *reservas* cumple con la 2NF, ya que los atributos no primos costototal, fechainicio, fechafin, planesestadia, numerohabitacion e idconsumo dependen completamente de la clave primaria id.

La tabla *servicios* cumple con la 2NF, ya que los atributos no primos capacidad, costodanios, nombre, idhotel y clase dependen completamente de la clave primaria id.

Tercera Forma Normal (3NF):

La Tercera Forma Normal establece que los atributos no primos deben depender solo de la clave primaria y no de otros atributos no primos.

La tabla *consumos* cumple con la 3NF, ya que el atributo no primo valortotal depende solo de la clave primaria id.

La tabla *habitaciones* cumple con la 3NF, ya que los atributos no primos dependen solo de la clave primaria id.

La tabla *hoteles* cumple con la 3NF, ya que los atributos no primos dependen solo de la clave primaria id.

La tabla *reservas* cumple con la 3NF, ya que los atributos no primos dependen solo de la clave primaria id.

La tabla *servicios* cumple con la 3NF, ya que los atributos no primos dependen solo de la clave primaria id.

Como las tablas cumplen con 3FN, también cumple el modelo con BCFN.

6. Requerimientos Funcionales

RF1 - Tipos de Usuarios:

El sistema otorga al administrador la habilidad de administrar completamente los tipos de usuarios. Esto significa que el administrador puede establecer, revisar, actualizar o eliminar cualquier tipo de usuario detallado en el enunciado. Recibe el nombre o el id necesario para la operación.

RF2 - Usuarios:

El sistema facilita la gestión de perfiles de usuario por parte del administrador de datos del hotel. Cada perfil tiene atributos clave como tipo de cedula, nombre, correo electrónico, etc. Esta funcionalidad puede recibir la info de los atributos de cada usuario necesaria para ejecutar las operaciones.

RF3 - Tipos de Habitación:

El administrador de datos del hotel tiene el poder de gestionar los diferentes tipos de habitaciones que el hotel ofrece. Esto incluye determinar las características y comodidades de cada tipo de habitación. Dicho esto esta funcionalidad recibe la información necesaria para para lograr estas funcionalidades básicas.

RF4 - Habitación:

La gestión detallada de cada habitación individual en el hotel es facilitada por el sistema y es responsabilidad del administrador de datos del hotel. Esta funcionalidad puede recibir información clave como el numero de habitación, valores booleanos de los objetos que tiene entre otras.

RF5 - Servicio del Hotel:

El administrador de datos del hotel tiene la capacidad de detallar y gestionar todos los servicios que el hotel ofrece, desde piscinas hasta bares y restaurantes. Esta funcionalidad recibe el id, nombre, el costo por posible daños asociados a uno de estos servicios para lograr hacer estas operaciones básicas.

RF6 - Plan de Consumo:

Los planes de consumo del hotel, ya sean estándar o promocionales, son gestionados por el administrador de datos del hotel. Estos planes determinan las ofertas y paquetes disponibles para los huéspedes durante su estancia, lo que puede incluir comidas, servicios y otras comodidades. Recibe esta información.

RF7 - Reserva de Alojamiento:

Los clientes tienen la capacidad de reservar habitaciones para períodos específicos a través del sistema, siempre que la habitación deseada esté disponible.

Ningún requerimiento anterior devuelve un mensaje de exitoso sino que simplemente agregar a la base de datos las nuevas modificaciones del código. En caso de haber un error si se mostrar en pantalla cual es el debido error.