hotel\_reservas.puml

@startuml

left to right direction

skinparam actorStyle awesome

title Sistema de Reservas de Hotel – Vista de Escenario (Casos de uso)

actor Cliente

actor "Recepcionista/Operador" as Operador

actor "Sistema de Pagos" as Pago

rectangle "Motor de Reservas" {

usecase "Buscar disponibilidad" as UC\_Buscar

usecase "Ver tarifas y equipamiento" as UC\_Tarifas

usecase "Registrar cliente" as UC\_Registro

usecase "Crear reserva" as UC\_Reserva

usecase "Modificar reserva" as UC\_Modificar

usecase "Cancelar reserva" as UC\_Cancelar

usecase "Procesar pago" as UC\_Pago

usecase "Recibir comprobante QR" as UC\_Comprobante

}

Cliente --> UC\_Buscar

Cliente --> UC\_Tarifas

Cliente --> UC\_Reserva

Cliente --> UC\_Modificar

Cliente --> UC\_Cancelar

Operador --> UC\_Registro

Operador --> UC\_Reserva

Operador --> UC\_Modificar

Operador --> UC\_Cancelar

UC\_Reserva .> UC\_Registro : <<include>>

UC\_Reserva .> UC\_Pago : <<extend>>\n(si pago en línea)

UC\_Pago --> Pago : API/Redirect

UC\_Pago .> UC\_Comprobante : <<include>>

@enduml

activity\_crear\_reserva.puml

@startuml activity\_crear\_reserva

title Actividad: Crear reserva

start

:Cliente ingresa fechas y cantidad de personas;

:Sistema busca habitaciones disponibles;

if (¿Hay disponibilidad?) then (Sí)

:Muestra opciones (tarifa/equipamiento);

:Cliente selecciona habitación;

:Sistema calcula total (precio \* noches);

if (¿Pago en línea?) then (Sí)

:Redirige a pasarela de pago;

if (¿Pago aprobado?) then (Sí)

:Registra pago;

:Genera comprobante (QR);

else (No)

:Informa error de pago;

stop

endif

else (No)

:Marca reserva como pendiente de pago en hotel;

endif

:Crea registro de reserva;

:Envía confirmación (correo/QR);

stop

else (No)

:Informa sin disponibilidad;

stop

endif

@enduml

Diagrama de clases

@startuml class\_modelo\_reservas

title Modelo Lógico – Sistema de Reservas

class Hotel {

+hotel\_id: int

+nombre: string

+direccion: string

+categoria: string

}

class Habitacion {

+habitacion\_id: int

+tipo: string

+capacidad: int

+precio: decimal

+disponible(): boolean

}

class Cliente {

+cliente\_id: int

+nombre: string

+apellido: string

+correo: string

+telefono: string

}

class Reserva {

+reserva\_id: int

+fecha\_entrada: date

+fecha\_salida: date

+cantidad\_personas: int

+calcularTotal(): decimal

}

class Pago {

+pago\_id: int

+monto: decimal

+medio: string

+estado: string

+fecha: datetime

}

class Comprobante {

+codigoQR: string

+emitidoEn: datetime

}

Hotel "1" -- "0..\*" Habitacion : posee

Habitacion "1" -- "0..\*" Reserva : asigna

Cliente "1" -- "0..\*" Reserva : realiza

Reserva "0..1" -- "1" Pago : se\_liquida\_por

Pago "0..1" -- "1" Comprobante : genera >

@enduml

Componentes / Despliegue

@startuml components\_despliegue

title Arquitectura - Diagrama de Componentes\nHotel Pacific Reef – Sistema de Reservas

skinparam componentStyle rectangle

skinparam packageStyle rectangle

actor Cliente as client

package "Front-end" {

[Web App (Cliente)] as WebClient <<component>>

}

node "Hotel Pacific Reef (Infraestructura)" {

[Nginx/Load Balancer] as LB <<component>>

[Reserva API] as API <<component>>

[Auth Service] as AUTH <<component>>

[QR/Comprobante Service] as QR <<component>>

database "DB Reservas" as DB

}

cloud "Proveedor de Pagos" {

[Pasarela de Pago] as PAY <<external>>

}

client --> WebClient : Navega / UI

WebClient --> LB : HTTPS

LB --> API : /api/\* (HTTP/REST)

API --> AUTH : JWT/OAuth2

API --> DB : JDBC/SQL

API --> PAY : Crear pago / Webhook

API --> QR : Genera comprobante (QR)

QR --> DB : Guarda comprobante

note right of API

Casos clave soportados:

- Consultar disponibilidad

- Crear/Modificar/Cancelar reserva

- Registrar cliente

- Procesar pago (pasarela)

end note

note bottom of DB

Tablas: Hotel, Habitacion, Cliente,

Reserva, Pago, Comprobante

end note

@enduml