Trabajo Final. SISTEMA DE RESERVAS

POR:
ANDERSON ELIAN GUTIERREZ BUENO - 1007441902

UNIVERSIDAD NACIONAL DE COLOMBIA - SEDE MEDELLÍN
FACULTAD DE MINAS

DISEÑO Y CONSTRUCCIÓN DE PRODUCTOS DE SOFTWARE

PROF. ALBEIRO ESPINOSA BEDOYA

MEDELLÍN
MARZO DE 2024
SEMESTRE 2024-02

ÍNDICE

1. DIAGRAMA DE CASOS DE USO

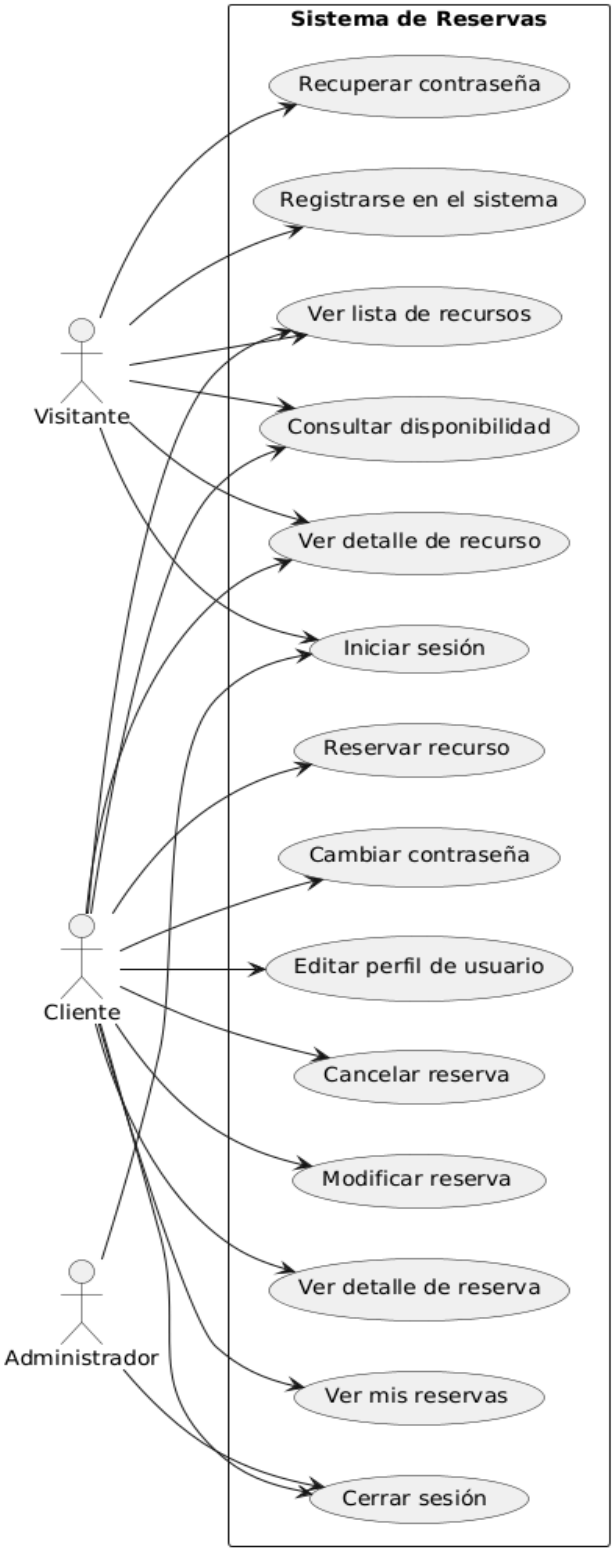
CU01 – Registrarse: Un visitante crea una nueva cuenta en el sistema.
CU02 – Iniciar sesión: Un usuario (cliente o administrador) inicia sesión con sus credenciales.
CU03 – Cerrar sesión: El usuario cierra su sesión actual.
CU04 – Recuperar contraseña: Un visitante (usuario no autenticado) recupera su contraseña olvidada mediante el sistema.
CU05 – Ver lista de recursos: Consultar el listado de recursos disponibles para reserva.
CU06 – Ver detalle de recurso: Consultar información detallada de un recurso (descripción, características, etc.).
CU07 – Consultar disponibilidad: Ver la disponibilidad (fechas/horas libres) de un recurso específico.
CU08 – Reservar recurso: Realizar una nueva reserva de un recurso (seleccionando fechas/horas disponibles).
CU09 – Ver mis reservas: Un cliente consulta el listado de sus propias reservas realizadas.
CU10 – Ver detalle de reserva: Ver información detallada de una reserva específica (fechas, recurso, estado).
CU11 – Modificar reserva: Un cliente modifica los detalles (p. ej. fecha/hora) de una de sus reservas existentes.
CU12 – Cancelar reserva: Un cliente cancela una de sus reservas.
CU13 – Editar perfil: El usuario actualiza los datos de su perfil (nombre, email, etc.).
CU14 – Cambiar contraseña: El usuario autenticado cambia su contraseña dentro del sistema.
CU15 – Agregar recurso: Un administrador registra (da de alta) un nuevo recurso en el sistema.
CU16 – Editar recurso: Un administrador edita la información de un recurso existente.
CU17 – Eliminar recurso: Un administrador elimina (o deshabilita) un recurso del sistema.
CU18 – Ver todas las reservas: El administrador consulta el listado de todas las reservas realizadas en el sistema.
CU19 – Modificar reserva de usuario: El administrador modifica los detalles de una reserva de un cliente (por ejemplo, cambiar la fecha a petición del cliente).
CU20 – Cancelar reserva de usuario: El administrador cancela una reserva en nombre de un cliente (por ejemplo, ante una incidencia).
CU21 – Gestionar disponibilidad de recurso: El administrador bloquea o habilita rangos de fechas/horas en los que un recurso no estará disponible para reserva (por mantenimiento u otros motivos).
CU22 – Asignar rol de administrador: Un administrador otorga privilegios de administrador a un usuario existente (o revoca dichos privilegios).
CU23 – Bloquear usuario: Un administrador inhabilita la cuenta de un usuario (impidiendo su acceso), o la reactiva.
CU24 – Generar reporte de reservas: El administrador genera un informe o reporte del sistema de reservas (por ejemplo, estadísticas de reservas en un período).

Código PlantUML para el diagrama de casos de uso:

Unset

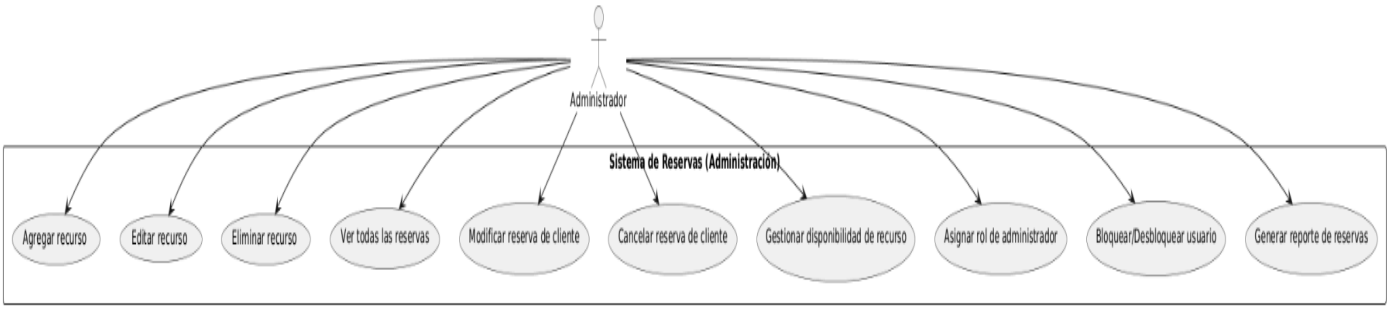
```
@startuml

left to right direction

actor "Visitante" as Visitante
actor "Cliente" as Cliente
actor "Administrador" as Administrador

rectangle "Sistema de Reservas" {
  usecase "Registrarse en el sistema" as CU01
  usecase "Iniciar sesión" as CU02
  usecase "Cerrar sesión" as CU03
  usecase "Recuperar contraseña" as CU04
  usecase "Ver lista de recursos" as CU05
  usecase "Ver detalle de recurso" as CU06
  usecase "Consultar disponibilidad" as CU07
  usecase "Reservar recurso" as CU08
  usecase "Ver mis reservas" as CU09
  usecase "Ver detalle de reserva" as CU10
  usecase "Modificar reserva" as CU11
  usecase "Cancelar reserva" as CU12
  usecase "Editar perfil de usuario" as CU13
  usecase "Cambiar contraseña" as CU14
}

Visitante --> CU01
Visitante --> CU02
Visitante --> CU04
Visitante --> CU05
Visitante --> CU06
Visitante --> CU07
Cliente --> CU03
Cliente --> CU05
Cliente --> CU06
Cliente --> CU07
Cliente --> CU08
Cliente --> CU09
Cliente --> CU10
Cliente --> CU11
Cliente --> CU12
Cliente --> CU13
Cliente --> CU14
Administrador --> CU02
```

```
Administrador --> CU03

@enduml
```



En el diagrama anterior, el actor **Visitante** (no autenticado) puede registrarse, iniciar sesión, recuperar contraseña y consultar recursos/disponibilidad. El actor **Cliente** (usuario autenticado) puede realizar y gestionar sus reservas (crear, ver, modificar, cancelar), además de las consultas y acciones sobre su perfil. Nótese que **Administrador** también participa en los casos de uso de inicio/cierre de sesión, aunque sus acciones principales se detallan en el siguiente diagrama.

```
Unset
@startuml

actor "Administrador" as Admin

rectangle "Sistema de Reservas (Administración)" {
  usecase "Agregar recurso" as CU15
  usecase "Editar recurso" as CU16
  usecase "Eliminar recurso" as CU17
  usecase "Ver todas las reservas" as CU18
  usecase "Modificar reserva de cliente" as CU19
  usecase "Cancelar reserva de cliente" as CU20
  usecase "Gestionar disponibilidad de recurso" as CU21
  usecase "Asignar rol de administrador" as CU22
  usecase "Bloquear/Desbloquear usuario" as CU23
  usecase "Generar reporte de reservas" as CU24
}

Admin --> CU15
Admin --> CU16
Admin --> CU17
Admin --> CU18
Admin --> CU19
Admin --> CU20
Admin --> CU21
Admin --> CU22
Admin --> CU23
Admin --> CU24
@enduml
```

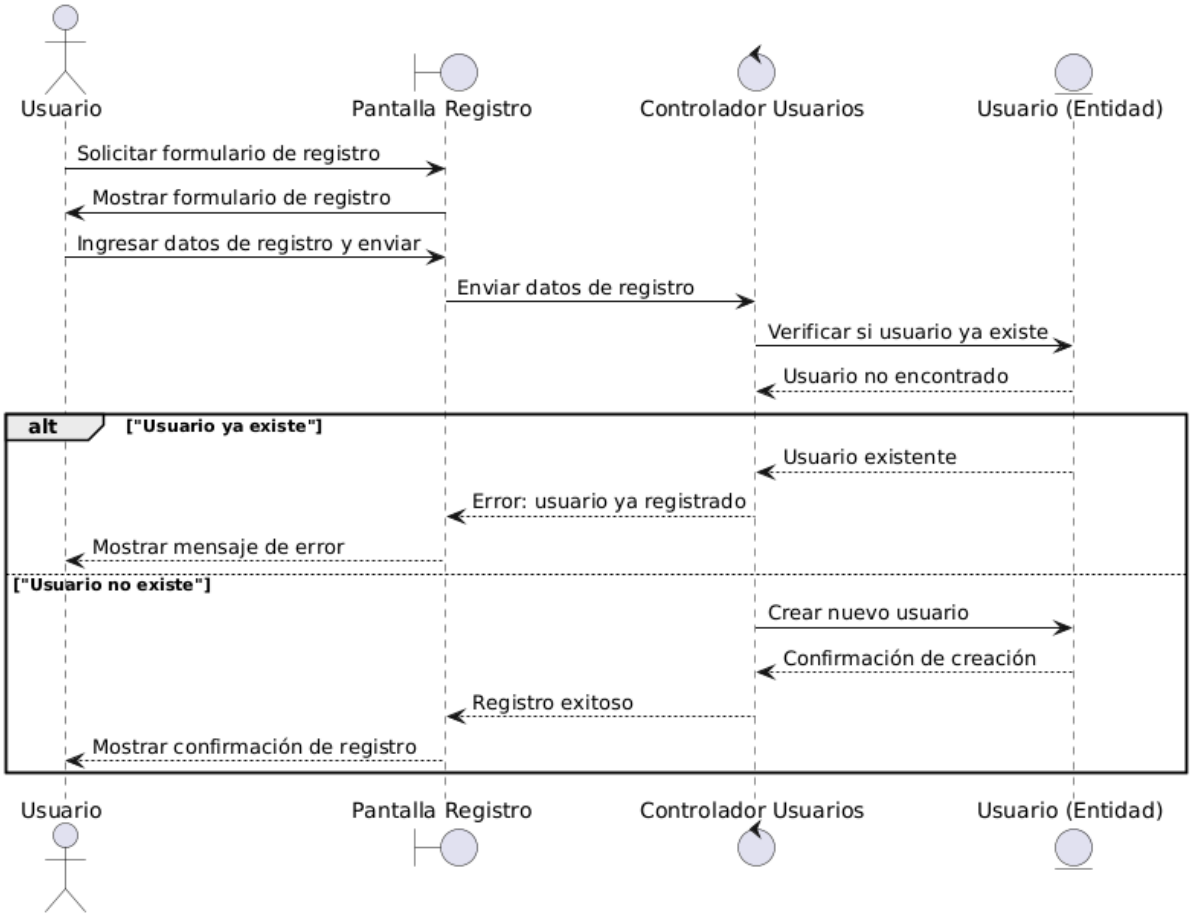


En el diagrama de administración, el actor **Administrador** puede gestionar los recursos del sistema (agregar/editar/eliminar), revisar y actuar sobre las reservas de los usuarios (ver todas, modificar o cancelar como administrador) y llevar a cabo tareas de gestión del sistema: gestionar la disponibilidad de los recursos (por ejemplo bloquear fechas no disponibles), administrar usuarios y roles (asignar rol admin, bloquear usuarios) y obtener reportes de las reservas.

2. DIAGRAMAS DE SECUENCIA POR CASO DE USO

CU01 – Registrarse: Un visitante crea una nueva cuenta en el sistema.

```
Unset
@startuml
actor "Usuario" as usuario
boundary "Pantalla Registro" as pantallaRegistro
control "Controlador Usuarios" as ctrlUsuarios
entity "Usuario (Entidad)" as usuarioEnt

usuario -> pantallaRegistro: Solicitar formulario de registro
pantallaRegistro -> usuario: Mostrar formulario de registro
usuario -> pantallaRegistro: Ingresar datos de registro y enviar
pantallaRegistro -> ctrlUsuarios: Enviar datos de registro
ctrlUsuarios -> usuarioEnt: Verificar si usuario ya existe
usuarioEnt --> ctrlUsuarios: Usuario no encontrado
alt "Usuario ya existe"
    usuarioEnt --> ctrlUsuarios: Usuario existente
    ctrlUsuarios --> pantallaRegistro: Error: usuario ya registrado
    pantallaRegistro --> usuario: Mostrar mensaje de error
else "Usuario no existe"
    ctrlUsuarios -> usuarioEnt: Crear nuevo usuario
    usuarioEnt --> ctrlUsuarios: Confirmación de creación
    ctrlUsuarios --> pantallaRegistro: Registro exitoso
    pantallaRegistro --> usuario: Mostrar confirmación de registro
end alt
@enduml
```

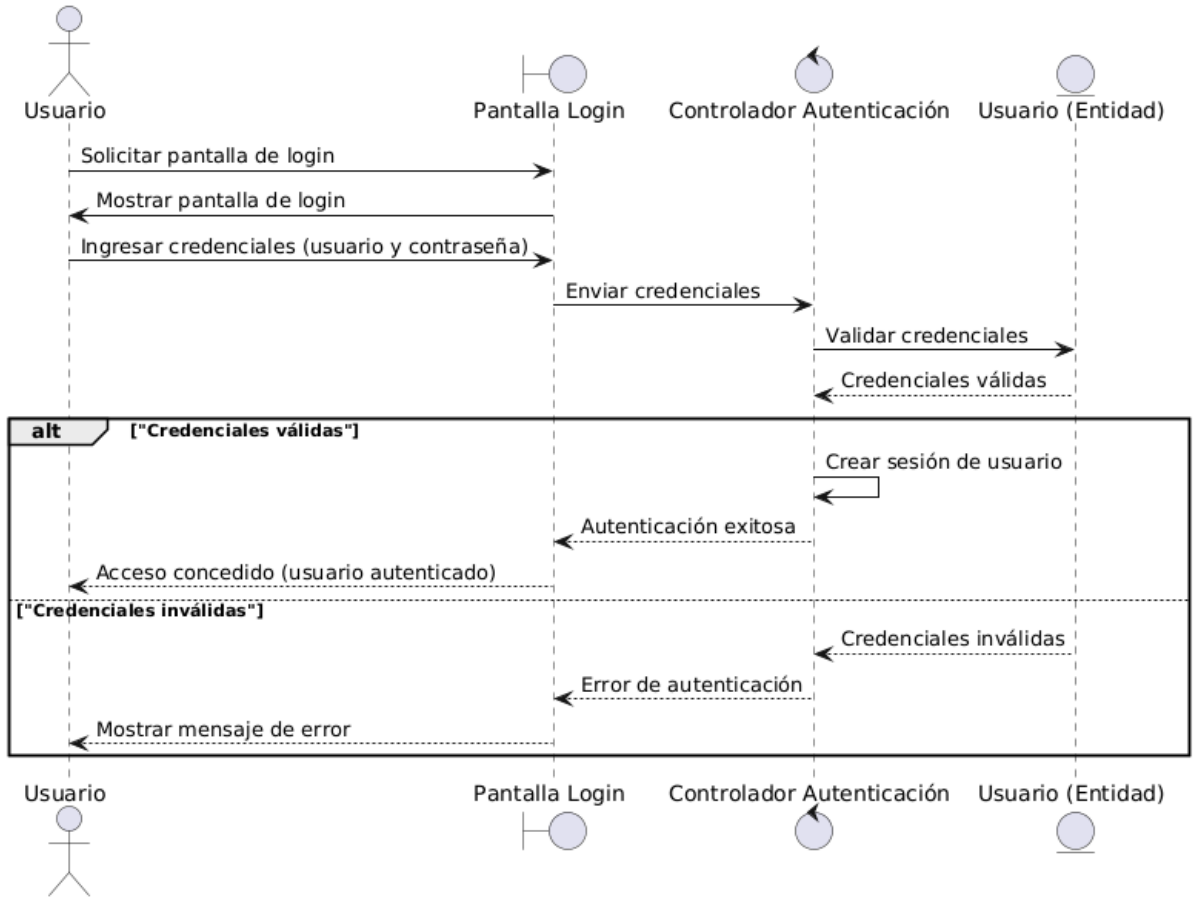


CU02 – Iniciar sesión: Un usuario (cliente o administrador) inicia sesión con sus credenciales.

Unset

```
@startuml
actor "Usuario" as usuario
boundary "Pantalla Login" as pantallaLogin
control "Controlador Autenticación" as ctrlAuth
entity "Usuario (Entidad)" as usuarioEnt

usuario -> pantallaLogin: Solicitar pantalla de login
pantallaLogin -> usuario: Mostrar pantalla de login
usuario -> pantallaLogin: Ingresar credenciales (usuario y
contraseña)
pantallaLogin -> ctrlAuth: Enviar credenciales
ctrlAuth -> usuarioEnt: Validar credenciales
usuarioEnt --> ctrlAuth: Credenciales válidas
alt "Credenciales válidas"
    ctrlAuth -> ctrlAuth: Crear sesión de usuario
    ctrlAuth --> pantallaLogin: Autenticación exitosa
    pantallaLogin --> usuario: Acceso concedido (usuario autenticado)
```

```
else "Credenciales inválidas"
    usuarioEnt --> ctrlAuth: Credenciales inválidas
    ctrlAuth --> pantallaLogin: Error de autenticación
    pantallaLogin --> usuario: Mostrar mensaje de error
end alt
@enduml
```

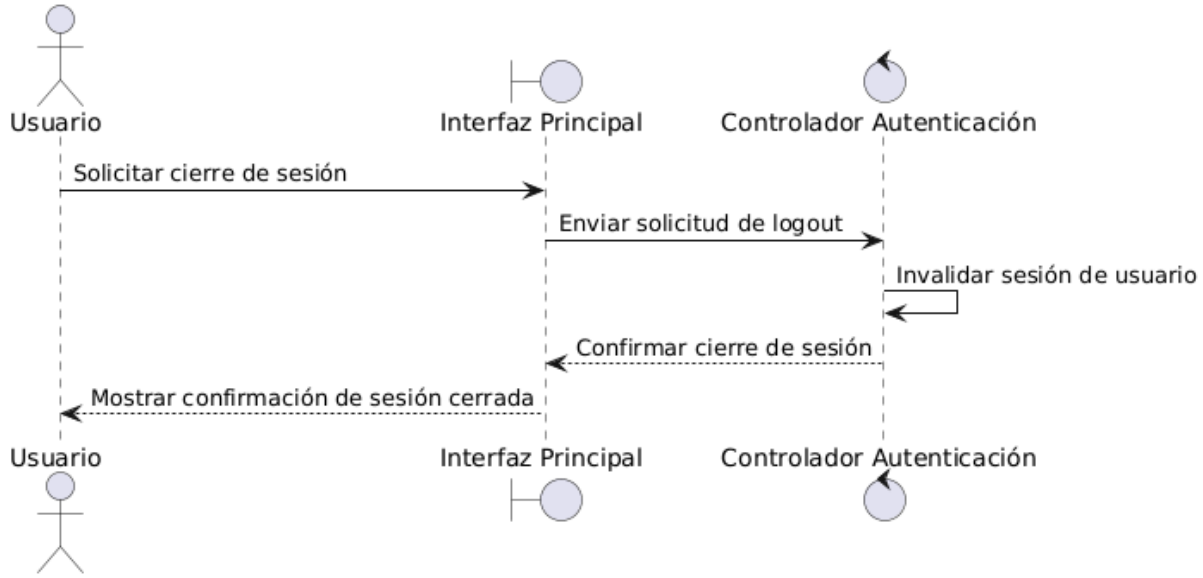


CU03 – Cerrar sesión: El usuario cierra su sesión actual.

Unset

```
@startuml
actor "Usuario" as usuario
boundary "Interfaz Principal" as interfazPrincipal
control "Controlador Autenticación" as ctrlAuth

usuario -> interfazPrincipal: Solicitar cierre de sesión
interfazPrincipal -> ctrlAuth: Enviar solicitud de logout
```

```
ctrlAuth -> ctrlAuth: Invalidar sesión de usuario
ctrlAuth --> interfazPrincipal: Confirmar cierre de sesión
interfazPrincipal --> usuario: Mostrar confirmación de sesión cerrada
@enduml
```

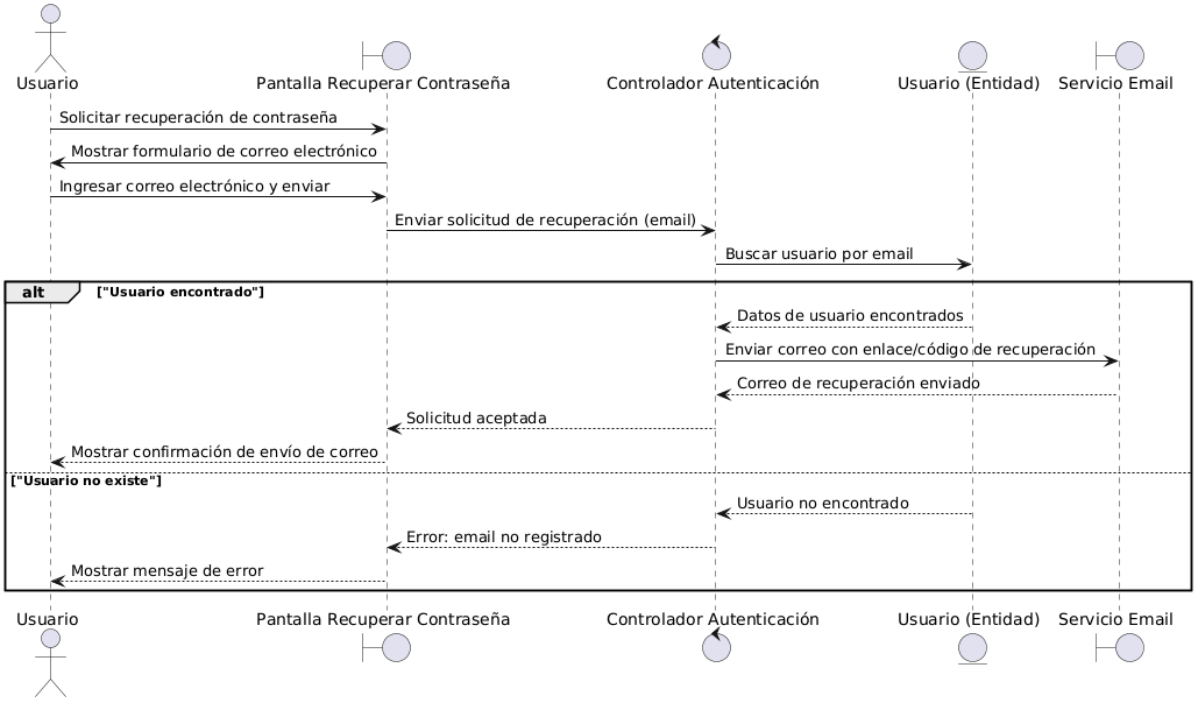


CU04 – Recuperar contraseña: Un visitante (usuario no autenticado) recupera su contraseña olvidada mediante el sistema.

Unset

```
@startuml
actor "Usuario" as usuario
boundary "Pantalla Recuperar Contraseña" as pantallaRecup
control "Controlador Autenticación" as ctrlAuth
entity "Usuario (Entidad)" as usuarioEnt
boundary "Servicio Email" as emailSvc

usuario -> pantallaRecup: Solicitar recuperación de contraseña
pantallaRecup -> usuario: Mostrar formulario de correo electrónico
usuario -> pantallaRecup: Ingresar correo electrónico y enviar
pantallaRecup -> ctrlAuth: Enviar solicitud de recuperación (email)
ctrlAuth -> usuarioEnt: Buscar usuario por email
alt "Usuario encontrado"
    usuarioEnt --> ctrlAuth: Datos de usuario encontrados
    ctrlAuth -> emailSvc: Enviar correo con enlace/código de
recuperación
    emailSvc --> ctrlAuth: Correo de recuperación enviado
    ctrlAuth --> pantallaRecup: Solicitud aceptada
```

```
    pantallaRecup --> usuario: Mostrar confirmación de envío de
correo
else "Usuario no existe"
    usuarioEnt --> ctrlAuth: Usuario no encontrado
    ctrlAuth --> pantallaRecup: Error: email no registrado
    pantallaRecup --> usuario: Mostrar mensaje de error
end alt
@enduml
```

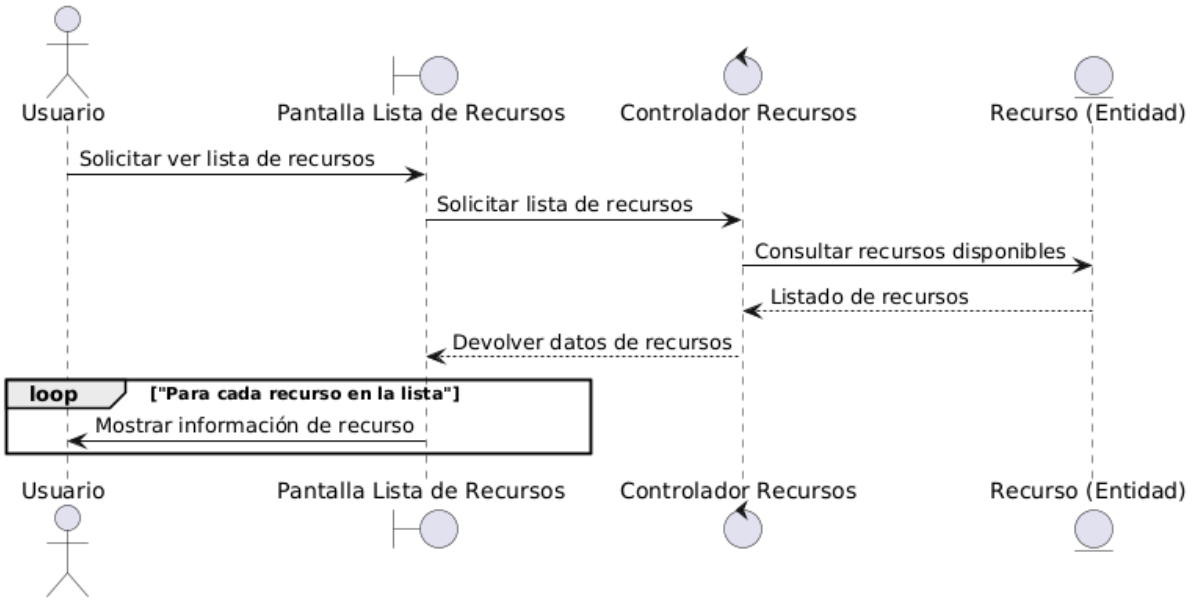


CU05 – Ver lista de recursos: Consultar el listado de recursos disponibles para reserva.

Unset

```
@startuml
actor "Usuario" as usuario
boundary "Pantalla Lista de Recursos" as pantallaListaRec
control "Controlador Recursos" as ctrlRecursos
entity "Recurso (Entidad)" as recursoEnt

usuario -> pantallaListaRec: Solicitar ver lista de recursos
pantallaListaRec -> ctrlRecursos: Solicitar lista de recursos
ctrlRecursos -> recursoEnt: Consultar recursos disponibles
recursoEnt --> ctrlRecursos: Listado de recursos
```

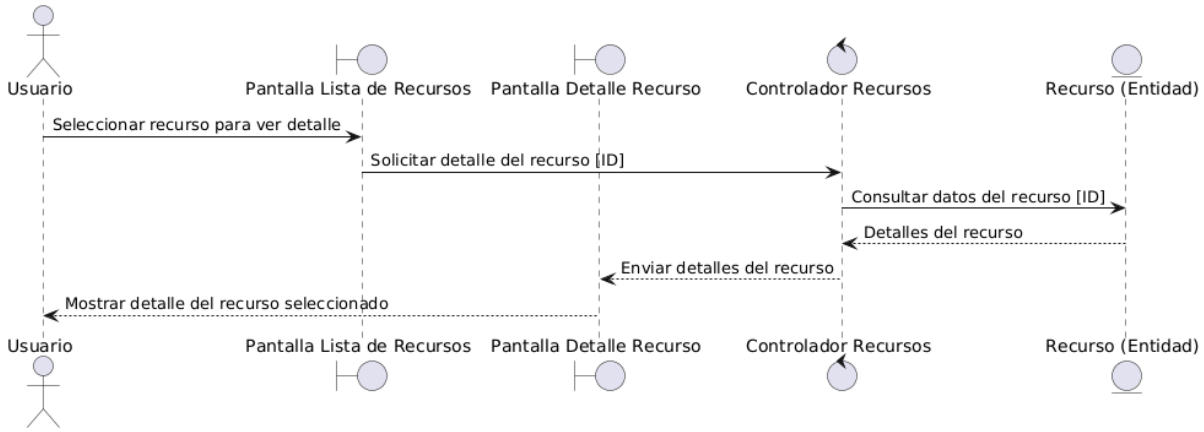
```
ctrlRecursos --> pantallaListaRec: Devolver datos de recursos
loop "Para cada recurso en la lista"
    pantallaListaRec -> usuario: Mostrar información de recurso
end loop
@enduml
```



CU06 – Ver detalle de recurso: Consultar información detallada de un recurso

Unset

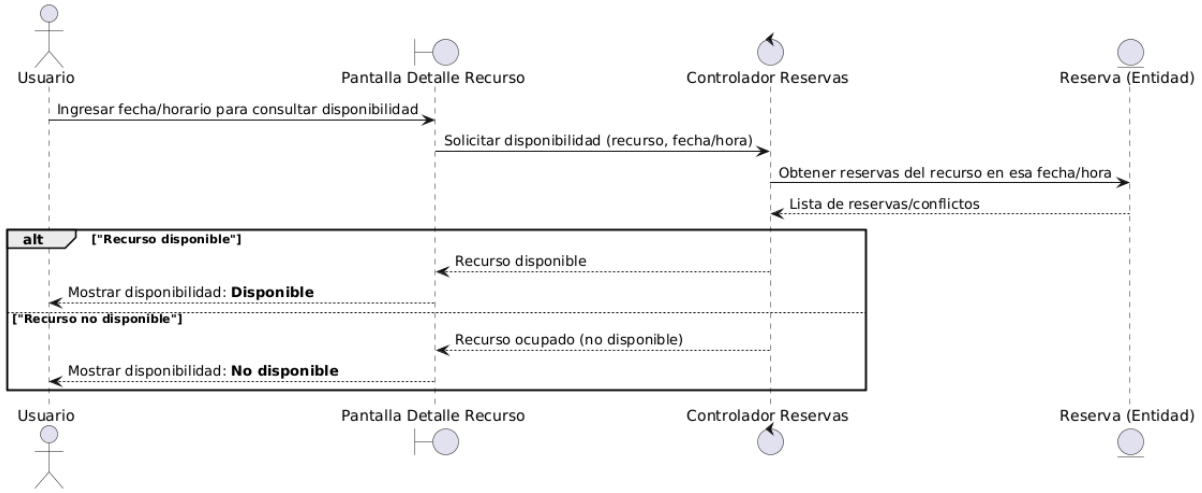
```
@startuml
actor "Usuario" as usuario
boundary "Pantalla Lista de Recursos" as pantallaListaRec
boundary "Pantalla Detalle Recurso" as pantallaDetRec
control "Controlador Recursos" as ctrlRecursos
entity "Recurso (Entidad)" as recursoEnt

usuario -> pantallaListaRec: Seleccionar recurso para ver detalle
pantallaListaRec -> ctrlRecursos: Solicitar detalle del recurso [ID]
ctrlRecursos -> recursoEnt: Consultar datos del recurso [ID]
recursoEnt --> ctrlRecursos: Detalles del recurso
ctrlRecursos --> pantallaDetRec: Enviar detalles del recurso
pantallaDetRec --> usuario: Mostrar detalle del recurso seleccionado
@enduml
```



CU07 – Consultar disponibilidad: Ver la disponibilidad (fechas/horas libres) de un recurso específico.

Unset

```
@startuml
actor "Usuario" as usuario
boundary "Pantalla Detalle Recurso" as pantallaDetRec
control "Controlador Reservas" as ctrlReservas
entity "Reserva (Entidad)" as reservaEnt

usuario -> pantallaDetRec: Ingresar fecha/horario para consultar
disponibilidad
pantallaDetRec -> ctrlReservas: Solicitar disponibilidad (recurso,
fecha/hora)
ctrlReservas -> reservaEnt: Obtener reservas del recurso en esa
fecha/hora
reservaEnt --> ctrlReservas: Lista de reservas/conflictos
alt "Recurso disponible"
    ctrlReservas --> pantallaDetRec: Recurso disponible
    pantallaDetRec --> usuario: Mostrar disponibilidad:
**Disponible**
else "Recurso no disponible"
    ctrlReservas --> pantallaDetRec: Recurso ocupado (no disponible)
    pantallaDetRec --> usuario: Mostrar disponibilidad: **No
disponible**
end alt
@enduml
```

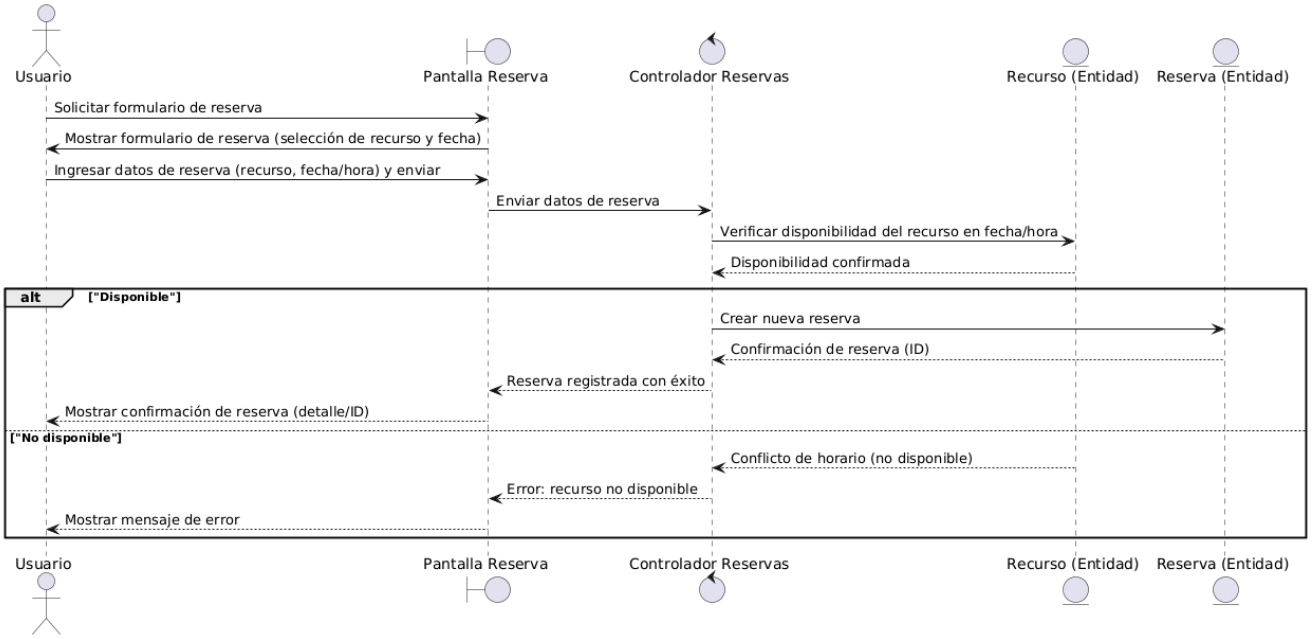


CU08 – Reservar recurso

Unset

```
@startuml
actor "Usuario" as usuario
boundary "Pantalla Reserva" as pantallaReserva
control "Controlador Reservas" as ctrlReservas
entity "Recurso (Entidad)" as recursoEnt
entity "Reserva (Entidad)" as reservaEnt

usuario -> pantallaReserva: Solicitar formulario de reserva
pantallaReserva -> usuario: Mostrar formulario de reserva (selección
de recurso y fecha)
usuario -> pantallaReserva: Ingresar datos de reserva (recurso,
fecha/hora) y enviar
pantallaReserva -> ctrlReservas: Enviar datos de reserva
ctrlReservas -> recursoEnt: Verificar disponibilidad del recurso en
fecha/hora
recursoEnt --> ctrlReservas: Disponibilidad confirmada
alt "Disponible"
    ctrlReservas -> reservaEnt: Crear nueva reserva
    reservaEnt --> ctrlReservas: Confirmación de reserva (ID)
    ctrlReservas --> pantallaReserva: Reserva registrada con éxito
    pantallaReserva --> usuario: Mostrar confirmación de reserva
(detalle/ID)
else "No disponible"
    recursoEnt --> ctrlReservas: Conflicto de horario (no disponible)
    ctrlReservas --> pantallaReserva: Error: recurso no disponible
    pantallaReserva --> usuario: Mostrar mensaje de error
```

```
end alt
@enduml
```

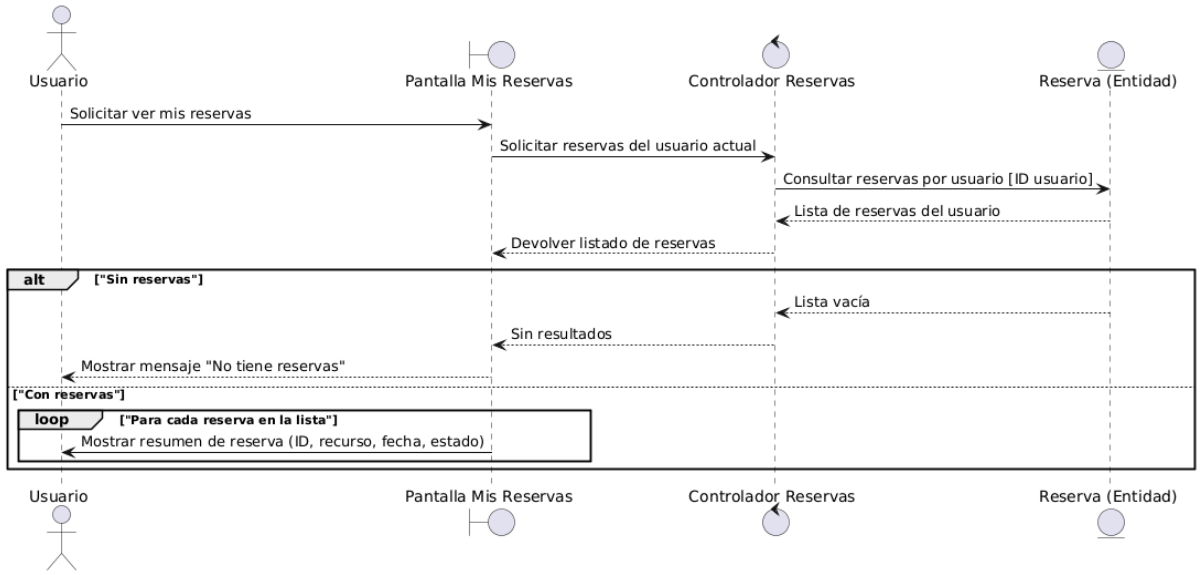


CU09 – Ver mis reservas: Un cliente consulta el listado de sus propias reservas realizadas.

Unset

```
@startuml
actor "Usuario" as usuario
boundary "Pantalla Mis Reservas" as pantallaMisRes
control "Controlador Reservas" as ctrlReservas
entity "Reserva (Entidad)" as reservaEnt

usuario -> pantallaMisRes: Solicitar ver mis reservas
pantallaMisRes -> ctrlReservas: Solicitar reservas del usuario actual
ctrlReservas -> reservaEnt: Consultar reservas por usuario [ID
usuario]
reservaEnt --> ctrlReservas: Lista de reservas del usuario
ctrlReservas --> pantallaMisRes: Devolver listado de reservas
alt "Sin reservas"
    reservaEnt --> ctrlReservas: Lista vacía
    ctrlReservas --> pantallaMisRes: Sin resultados
    pantallaMisRes --> usuario: Mostrar mensaje "No tiene reservas"
else "Con reservas"
```

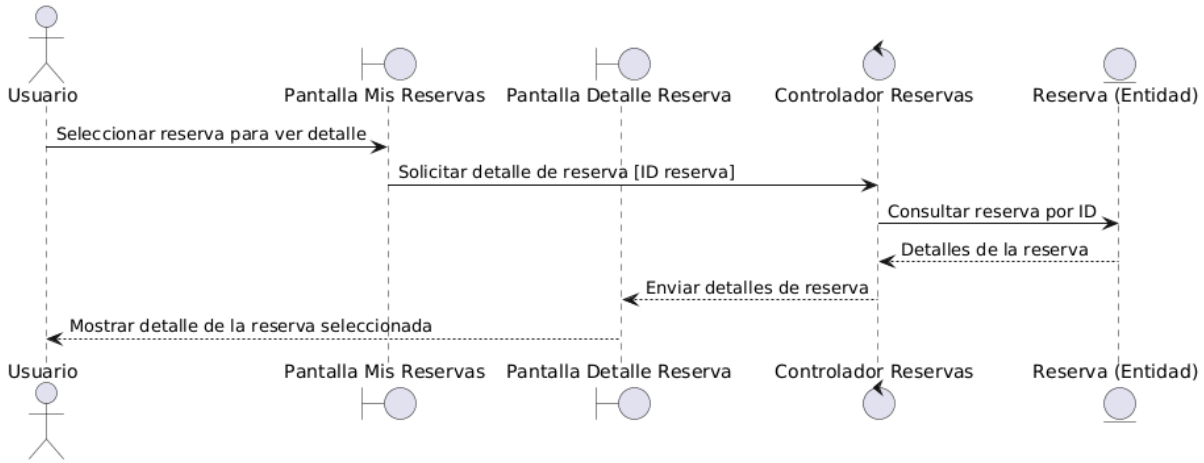
```
    loop "Para cada reserva en la lista"
        pantallaMisRes -> usuario: Mostrar resumen de reserva (ID,
recurso, fecha, estado)
    end loop
end alt
@enduml
```



CU10 – Ver detalle de reserva: Ver información detallada de una reserva específica (fechas, recurso, estado).

Unset

```
@startuml
actor "Usuario" as usuario
boundary "Pantalla Mis Reservas" as pantallaMisRes
boundary "Pantalla Detalle Reserva" as pantallaDetRes
control "Controlador Reservas" as ctrlReservas
entity "Reserva (Entidad)" as reservaEnt

usuario -> pantallaMisRes: Seleccionar reserva para ver detalle
pantallaMisRes -> ctrlReservas: Solicitar detalle de reserva [ID
reserva]
ctrlReservas -> reservaEnt: Consultar reserva por ID
reservaEnt --> ctrlReservas: Detalles de la reserva
ctrlReservas --> pantallaDetRes: Enviar detalles de reserva
pantallaDetRes --> usuario: Mostrar detalle de la reserva
seleccionada
@enduml
```

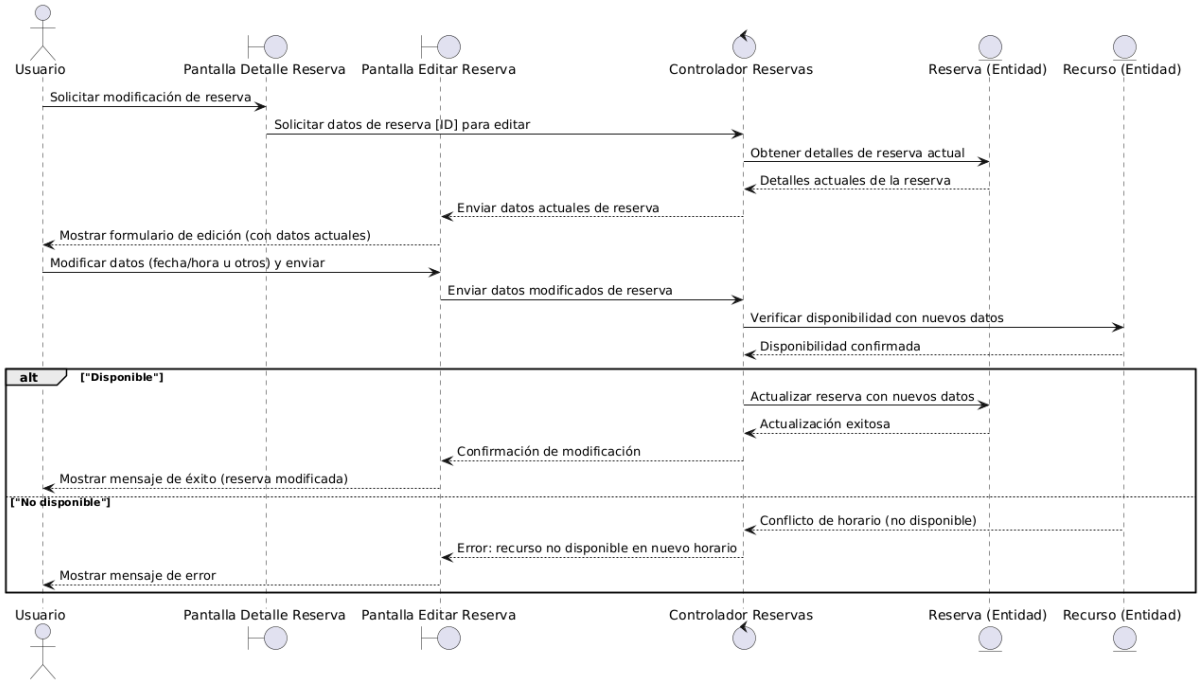


CU11 – Modificar reserva: Un cliente modifica los detalles (p. ej. fecha/hora) de una de sus reservas existentes.

Unset

```
@startuml
actor "Usuario" as usuario
boundary "Pantalla Detalle Reserva" as pantallaDetRes
boundary "Pantalla Editar Reserva" as pantallaEditRes
control "Controlador Reservas" as ctrlReservas
entity "Reserva (Entidad)" as reservaEnt
entity "Recurso (Entidad)" as recursoEnt

usuario -> pantallaDetRes: Solicitar modificación de reserva
pantallaDetRes -> ctrlReservas: Solicitar datos de reserva [ID] para
editar
ctrlReservas -> reservaEnt: Obtener detalles de reserva actual
reservaEnt --> ctrlReservas: Detalles actuales de la reserva
ctrlReservas --> pantallaEditRes: Enviar datos actuales de reserva
pantallaEditRes --> usuario: Mostrar formulario de edición (con datos
actuales)
usuario -> pantallaEditRes: Modificar datos (fecha/hora u otros) y
enviar
pantallaEditRes -> ctrlReservas: Enviar datos modificados de reserva
ctrlReservas -> recursoEnt: Verificar disponibilidad con nuevos datos
recursoEnt --> ctrlReservas: Disponibilidad confirmada
alt "Disponible"
    ctrlReservas -> reservaEnt: Actualizar reserva con nuevos datos
    reservaEnt --> ctrlReservas: Actualización exitosa
    ctrlReservas --> pantallaEditRes: Confirmación de modificación
    pantallaEditRes --> usuario: Mostrar mensaje de éxito (reserva
modificada)
else "No disponible"
```

```
    recursoEnt --> ctrlReservas: Conflicto de horario (no disponible)
    ctrlReservas --> pantallaEditRes: Error: recurso no disponible en
nuevo horario
    pantallaEditRes --> usuario: Mostrar mensaje de error
end alt
@enduml
```

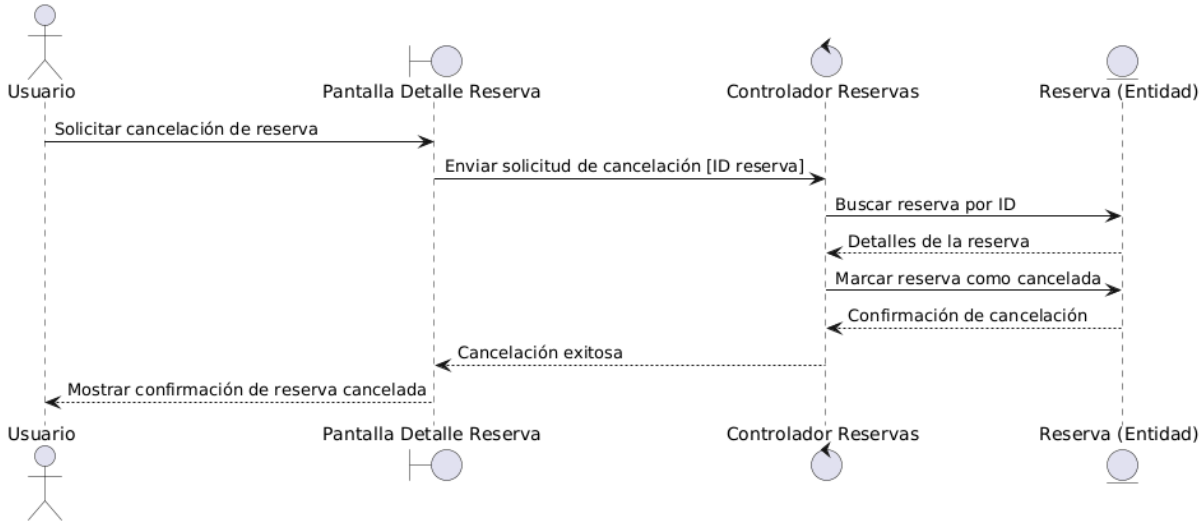


CU12 – Cancelar reserva: Un cliente cancela una de sus reservas.

Unset

```
@startuml
actor "Usuario" as usuario
boundary "Pantalla Detalle Reserva" as pantallaDetRes
control "Controlador Reservas" as ctrlReservas
entity "Reserva (Entidad)" as reservaEnt

usuario -> pantallaDetRes: Solicitar cancelación de reserva
pantallaDetRes -> ctrlReservas: Enviar solicitud de cancelación [ID
reserva]
ctrlReservas -> reservaEnt: Buscar reserva por ID
reservaEnt --> ctrlReservas: Detalles de la reserva
ctrlReservas -> reservaEnt: Marcar reserva como cancelada
reservaEnt --> ctrlReservas: Confirmación de cancelación
ctrlReservas --> pantallaDetRes: Cancelación exitosa
pantallaDetRes --> usuario: Mostrar confirmación de reserva cancelada
```

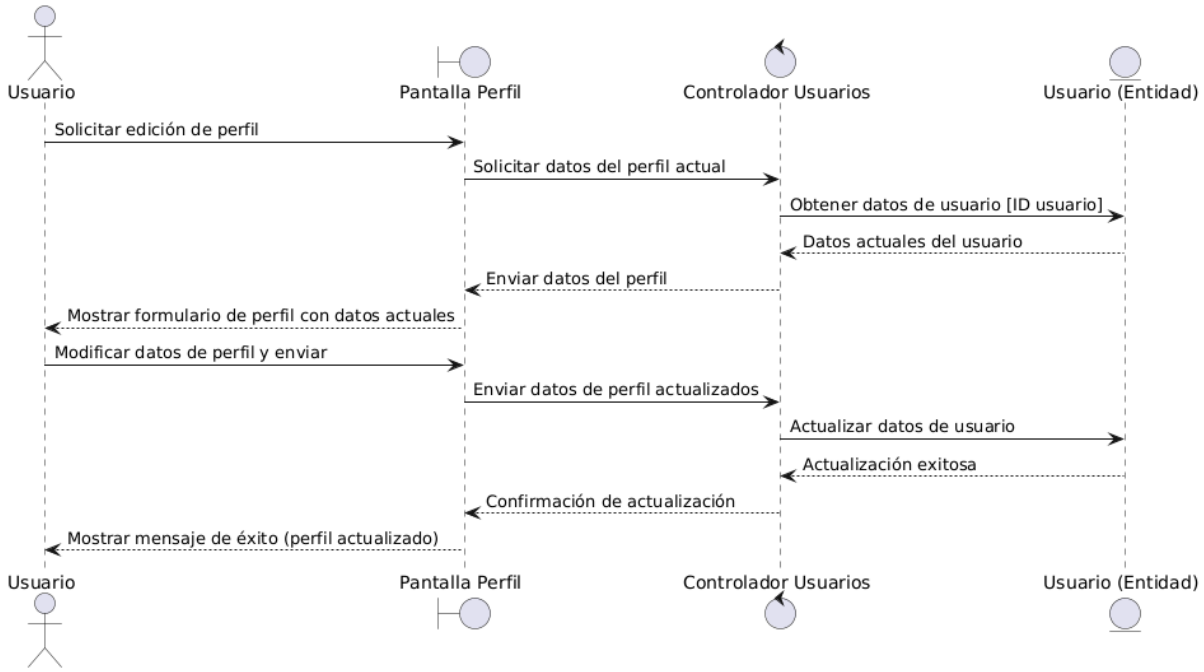
```
@enduml
```



CU13 – Editar perfil: El usuario actualiza los datos de su perfil (nombre, email, etc.).

Unset

```
@startuml
actor "Usuario" as usuario
boundary "Pantalla Perfil" as pantallaPerfil
control "Controlador Usuarios" as ctrlUsuarios
entity "Usuario (Entidad)" as usuarioEnt

usuario -> pantallaPerfil: Solicitar edición de perfil
pantallaPerfil -> ctrlUsuarios: Solicitar datos del perfil actual
ctrlUsuarios -> usuarioEnt: Obtener datos de usuario [ID usuario]
usuarioEnt --> ctrlUsuarios: Datos actuales del usuario
ctrlUsuarios --> pantallaPerfil: Enviar datos del perfil
pantallaPerfil --> usuario: Mostrar formulario de perfil con datos
actuales
usuario -> pantallaPerfil: Modificar datos de perfil y enviar
pantallaPerfil -> ctrlUsuarios: Enviar datos de perfil actualizados
ctrlUsuarios -> usuarioEnt: Actualizar datos de usuario
usuarioEnt --> ctrlUsuarios: Actualización exitosa
ctrlUsuarios --> pantallaPerfil: Confirmación de actualización
pantallaPerfil --> usuario: Mostrar mensaje de éxito (perfil
actualizado)
@enduml
```

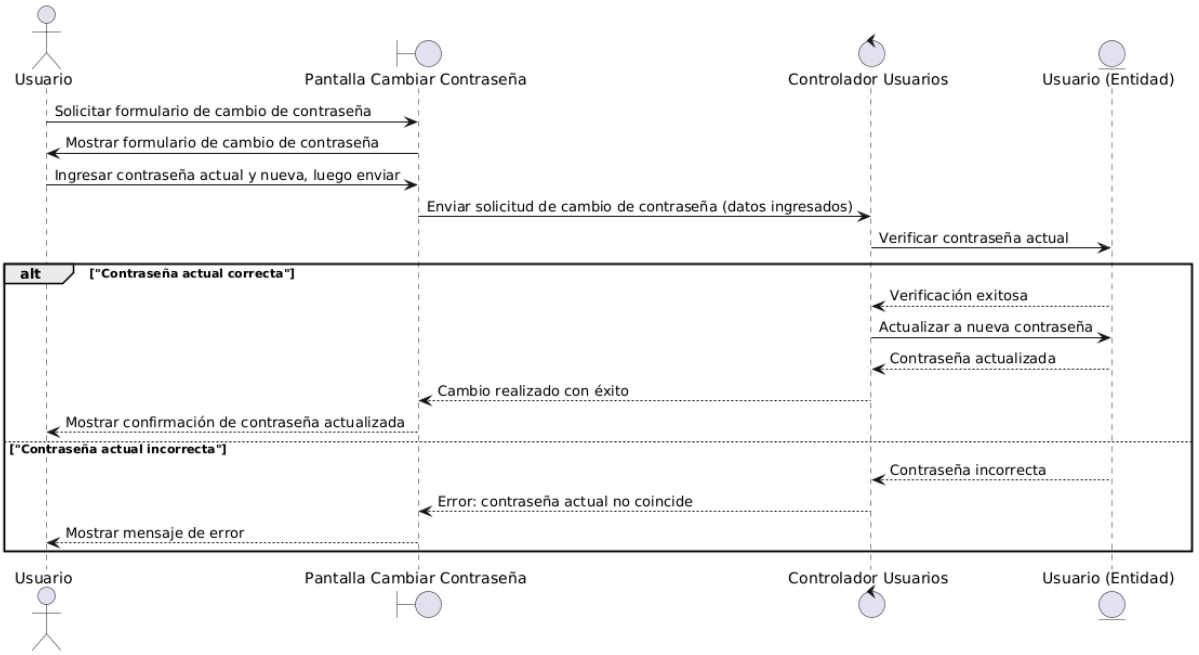


CU14 – Cambiar contraseña: El usuario autenticado cambia su contraseña dentro del sistema.

Unset

```
@startuml
actor "Usuario" as usuario
boundary "Pantalla Cambiar Contraseña" as pantallaCambiarPass
control "Controlador Usuarios" as ctrlUsuarios
entity "Usuario (Entidad)" as usuarioEnt

usuario -> pantallaCambiarPass: Solicitar formulario de cambio de
contraseña
pantallaCambiarPass -> usuario: Mostrar formulario de cambio de
contraseña
usuario -> pantallaCambiarPass: Ingresar contraseña actual y nueva,
luego enviar
pantallaCambiarPass -> ctrlUsuarios: Enviar solicitud de cambio de
contraseña (datos ingresados)
ctrlUsuarios -> usuarioEnt: Verificar contraseña actual
alt "Contraseña actual correcta"
    usuarioEnt --> ctrlUsuarios: Verificación exitosa
    ctrlUsuarios -> usuarioEnt: Actualizar a nueva contraseña
    usuarioEnt --> ctrlUsuarios: Contraseña actualizada
    ctrlUsuarios --> pantallaCambiarPass: Cambio realizado con éxito
    pantallaCambiarPass --> usuario: Mostrar confirmación de
contraseña actualizada
else "Contraseña actual incorrecta"
```

```
    usuarioEnt --> ctrlUsuarios: Contraseña incorrecta
    ctrlUsuarios --> pantallaCambiarPass: Error: contraseña actual no
coincide
    pantallaCambiarPass --> usuario: Mostrar mensaje de error
end alt
@enduml
```

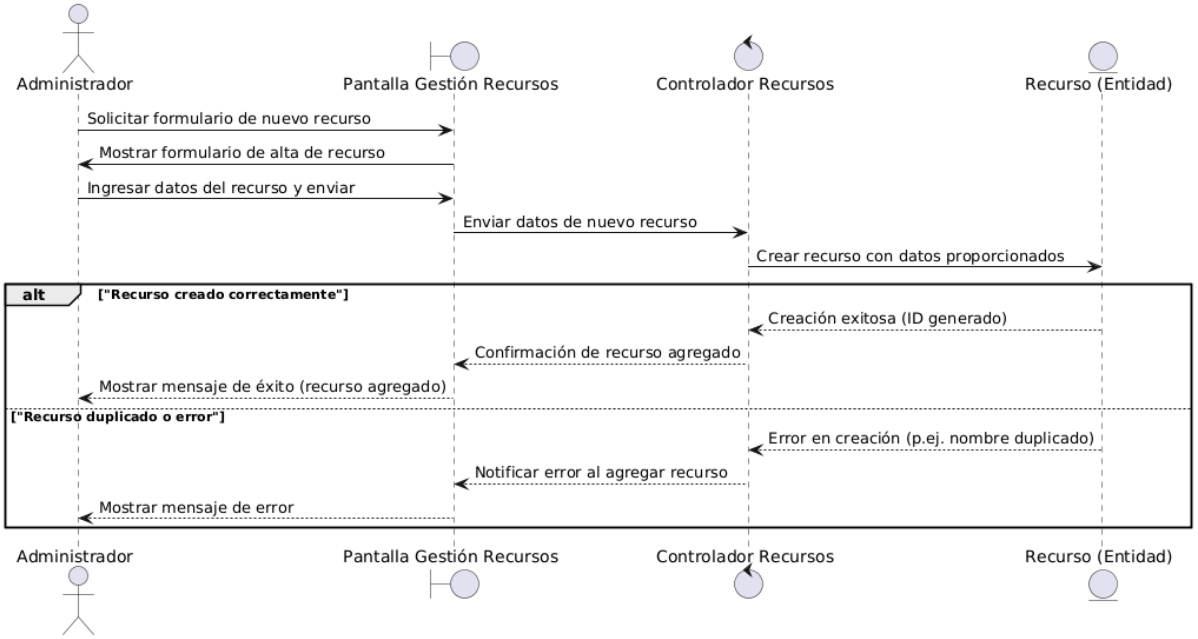


CU15 – Agregar recurso: Un administrador registra (da de alta) un nuevo recurso en el sistema.

Unset

```
@startuml
actor "Administrador" as admin
boundary "Pantalla Gestión Recursos" as pantallaGestRec
control "Controlador Recursos" as ctrlRecursos
entity "Recurso (Entidad)" as recursoEnt

admin -> pantallaGestRec: Solicitar formulario de nuevo recurso
pantallaGestRec -> admin: Mostrar formulario de alta de recurso
admin -> pantallaGestRec: Ingresar datos del recurso y enviar
pantallaGestRec -> ctrlRecursos: Enviar datos de nuevo recurso
ctrlRecursos -> recursoEnt: Crear recurso con datos proporcionados
alt "Recurso creado correctamente"
    recursoEnt --> ctrlRecursos: Creación exitosa (ID generado)
```

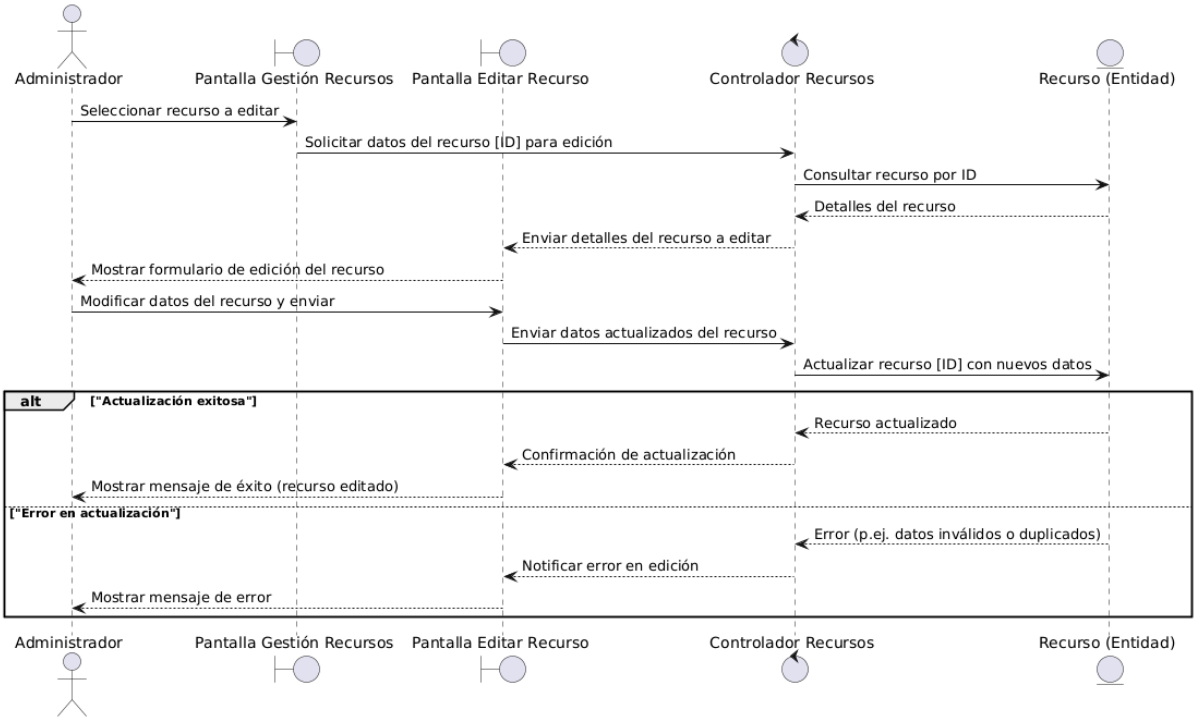
```
    ctrlRecursos --> pantallaGestRec: Confirmación de recurso
agregado
    pantallaGestRec --> admin: Mostrar mensaje de éxito (recurso
agregado)
else "Recurso duplicado o error"
    recursoEnt --> ctrlRecursos: Error en creación (p.ej. nombre
duplicado)
    ctrlRecursos --> pantallaGestRec: Notificar error al agregar
recurso
    pantallaGestRec --> admin: Mostrar mensaje de error
end alt
@enduml
```



CU16 – Editar recurso: Un administrador edita la información de un recurso existente.

```
Unset
@startuml
actor "Administrador" as admin
boundary "Pantalla Gestión Recursos" as pantallaGestRec
boundary "Pantalla Editar Recurso" as pantallaEditRec
control "Controlador Recursos" as ctrlRecursos
entity "Recurso (Entidad)" as recursoEnt

admin -> pantallaGestRec: Seleccionar recurso a editar
```

```
pantallaGestRec -> ctrlRecursos: Solicitar datos del recurso [ID]
para edición
ctrlRecursos -> recursoEnt: Consultar recurso por ID
recursoEnt --> ctrlRecursos: Detalles del recurso
ctrlRecursos --> pantallaEditRec: Enviar detalles del recurso a
editar
pantallaEditRec --> admin: Mostrar formulario de edición del recurso
admin -> pantallaEditRec: Modificar datos del recurso y enviar
pantallaEditRec -> ctrlRecursos: Enviar datos actualizados del
recurso
ctrlRecursos -> recursoEnt: Actualizar recurso [ID] con nuevos datos
alt "Actualización exitosa"
    recursoEnt --> ctrlRecursos: Recurso actualizado
    ctrlRecursos --> pantallaEditRec: Confirmación de actualización
    pantallaEditRec --> admin: Mostrar mensaje de éxito (recurso
editado)
else "Error en actualización"
    recursoEnt --> ctrlRecursos: Error (p.ej. datos inválidos o
duplicados)
    ctrlRecursos --> pantallaEditRec: Notificar error en edición
    pantallaEditRec --> admin: Mostrar mensaje de error
end alt
@enduml
```



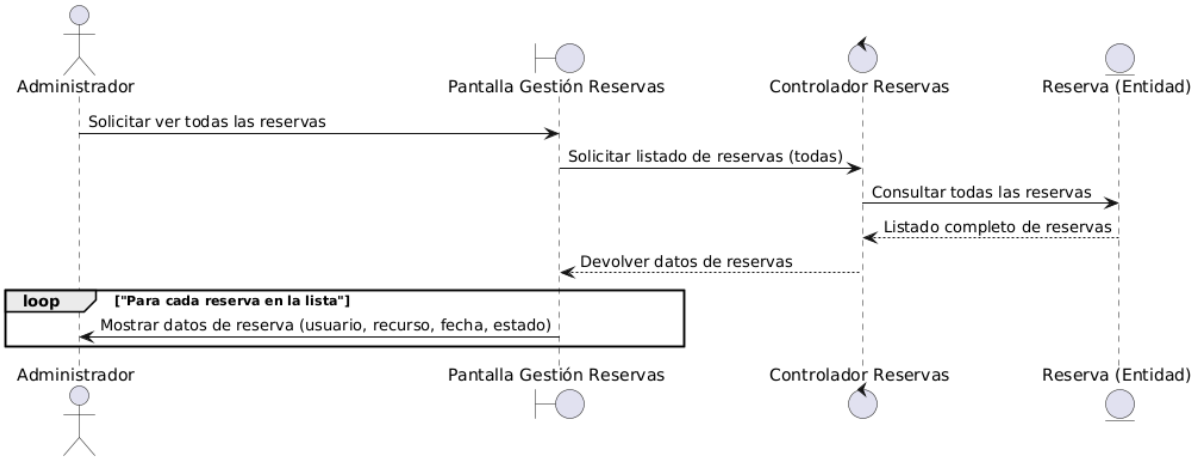
CU17 – Eliminar recurso: Un administrador elimina (o deshabilita) un recurso del sistema.

```
Unset
@startuml
actor "Administrador" as admin
boundary "Pantalla Gestión Recursos" as pantallaGestRec
control "Controlador Recursos" as ctrlRecursos
entity "Recurso (Entidad)" as recursoEnt
entity "Reserva (Entidad)" as reservaEnt

admin -> pantallaGestRec: Seleccionar recurso a eliminar
pantallaGestRec -> ctrlRecursos: Enviar solicitud de eliminación [ID
recurso]
ctrlRecursos -> recursoEnt: Verificar existencia del recurso y
relaciones
ctrlRecursos -> reservaEnt: Verificar reservas asociadas al recurso
alt "Sin reservas activas"
    reservaEnt --> ctrlRecursos: No hay reservas activas
    recursoEnt --> ctrlRecursos: Recurso encontrado (eliminable)
    ctrlRecursos -> recursoEnt: Eliminar recurso
    recursoEnt --> ctrlRecursos: Eliminación exitosa
    ctrlRecursos --> pantallaGestRec: Confirmar recurso eliminado
    pantallaGestRec --> admin: Mostrar mensaje de recurso eliminado
else "Con reservas activas o no encontrado"
    reservaEnt --> ctrlRecursos: Existen reservas activas
    recursoEnt --> ctrlRecursos: Error (no se puede eliminar)
    ctrlRecursos --> pantallaGestRec: Notificar imposibilidad de
eliminar
    pantallaGestRec --> admin: Mostrar mensaje de error (no se puede
eliminar recurso)
end alt
@enduml
```

CU18 – Ver todas las reservas: El administrador consulta el listado de todas las reservas realizadas en el sistema.

```
Unset
@startuml
actor "Administrador" as admin
boundary "Pantalla Gestión Reservas" as pantallaGestRes
control "Controlador Reservas" as ctrlReservas
entity "Reserva (Entidad)" as reservaEnt
```

```
admin -> pantallaGestRes: Solicitar ver todas las reservas
pantallaGestRes -> ctrlReservas: Solicitar listado de reservas
(todas)
ctrlReservas -> reservaEnt: Consultar todas las reservas
reservaEnt --> ctrlReservas: Listado completo de reservas
ctrlReservas --> pantallaGestRes: Devolver datos de reservas
loop "Para cada reserva en la lista"
    pantallaGestRes -> admin: Mostrar datos de reserva (usuario,
recurso, fecha, estado)
end loop
@enduml
```

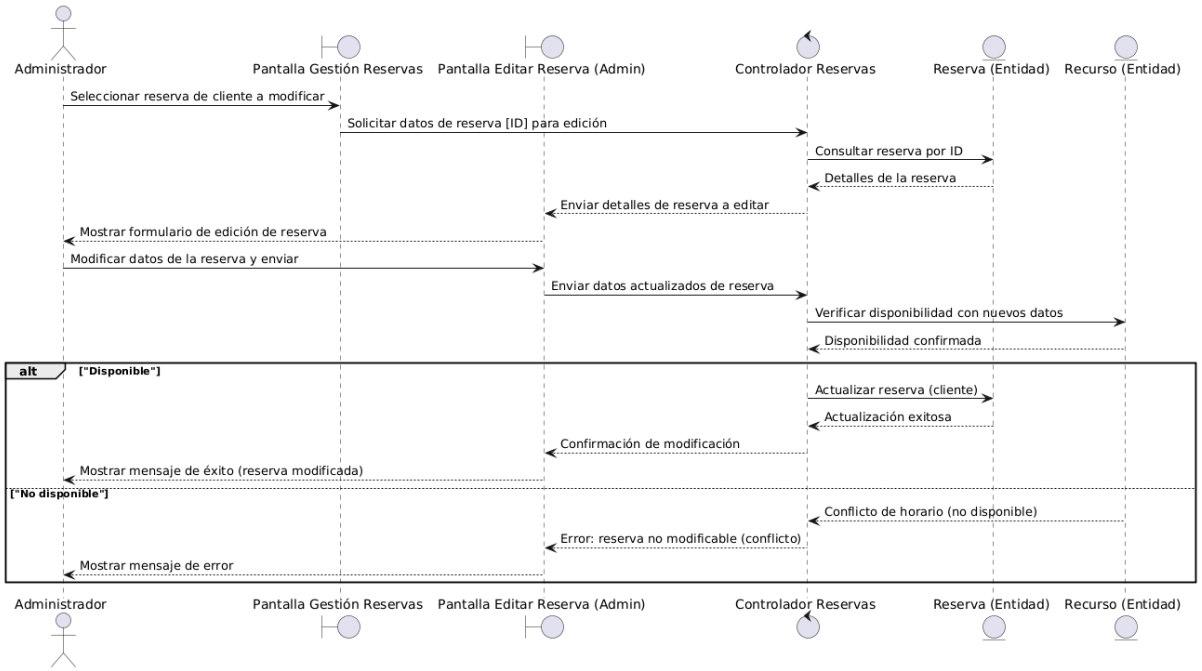


CU19 – Modificar reserva de usuario: El administrador modifica los detalles de una reserva de un cliente (por ejemplo, cambiar la fecha a petición del cliente).

Unset

```
@startuml
actor "Administrador" as admin
boundary "Pantalla Gestión Reservas" as pantallaGestRes
boundary "Pantalla Editar Reserva (Admin)" as pantallaEditResAdmin
control "Controlador Reservas" as ctrlReservas
entity "Reserva (Entidad)" as reservaEnt
entity "Recurso (Entidad)" as recursoEnt

admin -> pantallaGestRes: Seleccionar reserva de cliente a modificar
pantallaGestRes -> ctrlReservas: Solicitar datos de reserva [ID] para
edición
ctrlReservas -> reservaEnt: Consultar reserva por ID
reservaEnt --> ctrlReservas: Detalles de la reserva
```

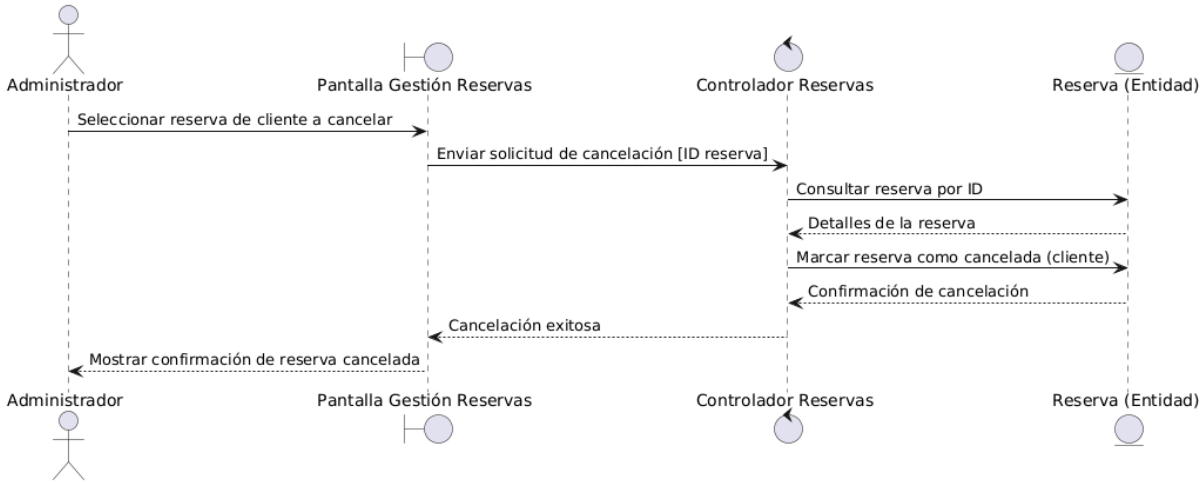
```
ctrlReservas --> pantallaEditResAdmin: Enviar detalles de reserva a
editar
pantallaEditResAdmin --> admin: Mostrar formulario de edición de
reserva
admin -> pantallaEditResAdmin: Modificar datos de la reserva y enviar
pantallaEditResAdmin -> ctrlReservas: Enviar datos actualizados de
reserva
ctrlReservas -> recursoEnt: Verificar disponibilidad con nuevos datos
recursoEnt --> ctrlReservas: Disponibilidad confirmada
alt "Disponible"
    ctrlReservas -> reservaEnt: Actualizar reserva (cliente)
    reservaEnt --> ctrlReservas: Actualización exitosa
    ctrlReservas --> pantallaEditResAdmin: Confirmación de
modificación
    pantallaEditResAdmin --> admin: Mostrar mensaje de éxito (reserva
modificada)
else "No disponible"
    recursoEnt --> ctrlReservas: Conflicto de horario (no disponible)
    ctrlReservas --> pantallaEditResAdmin: Error: reserva no
modificable (conflicto)
    pantallaEditResAdmin --> admin: Mostrar mensaje de error
end alt
@enduml
```



CU20 – Cancelar reserva de usuario: El administrador cancela una reserva en nombre de un cliente (por ejemplo, ante una incidencia).

Unset

```
@startuml
actor "Administrador" as admin
boundary "Pantalla Gestión Reservas" as pantallaGestRes
control "Controlador Reservas" as ctrlReservas
entity "Reserva (Entidad)" as reservaEnt

admin -> pantallaGestRes: Seleccionar reserva de cliente a cancelar
pantallaGestRes -> ctrlReservas: Enviar solicitud de cancelación [ID
reserva]
ctrlReservas -> reservaEnt: Consultar reserva por ID
reservaEnt --> ctrlReservas: Detalles de la reserva
ctrlReservas -> reservaEnt: Marcar reserva como cancelada (cliente)
reservaEnt --> ctrlReservas: Confirmación de cancelación
ctrlReservas --> pantallaGestRes: Cancelación exitosa
pantallaGestRes --> admin: Mostrar confirmación de reserva cancelada
@enduml
```

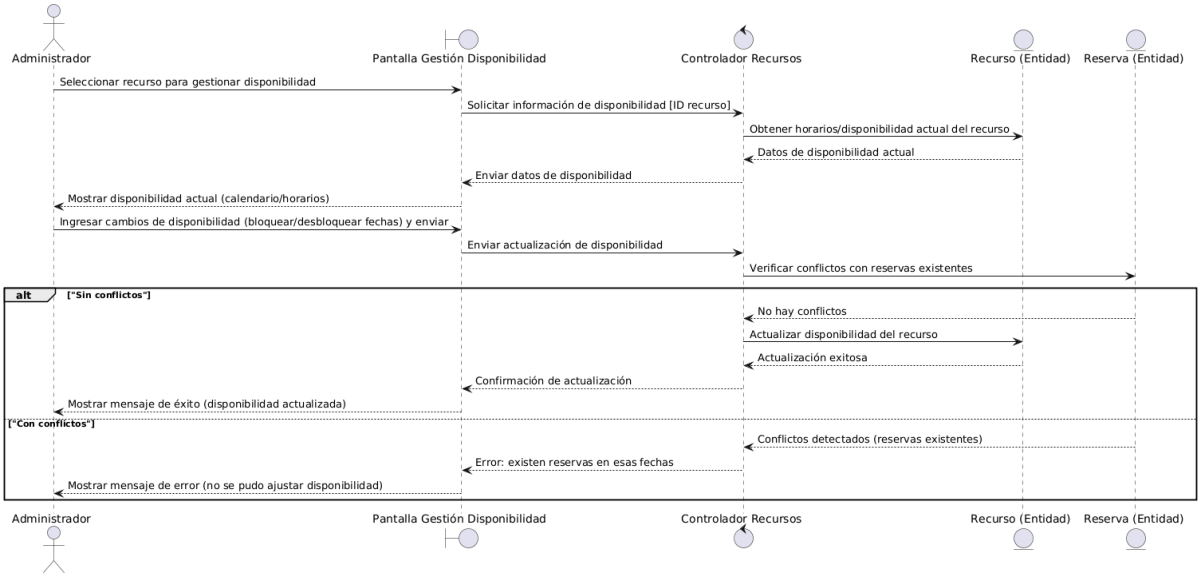


CU21 – Gestionar disponibilidad de recurso: El administrador bloquea o habilita rangos de fechas/horas en los que un recurso no estará disponible para reserva (por mantenimiento u otros motivos).

Unset

```
@startuml
actor "Administrador" as admin
boundary "Pantalla Gestión Disponibilidad" as pantallaDisp
control "Controlador Recursos" as ctrlRecursos
entity "Recurso (Entidad)" as recursoEnt
entity "Reserva (Entidad)" as reservaEnt
```

```
admin -> pantallaDisp: Seleccionar recurso para gestionar
disponibilidad
pantallaDisp -> ctrlRecursos: Solicitar información de disponibilidad
[ID recurso]
ctrlRecursos -> recursoEnt: Obtener horarios/disponibilidad actual
del recurso
recursoEnt --> ctrlRecursos: Datos de disponibilidad actual
ctrlRecursos --> pantallaDisp: Enviar datos de disponibilidad
pantallaDisp --> admin: Mostrar disponibilidad actual
(calendario/horarios)
admin -> pantallaDisp: Ingresar cambios de disponibilidad
(bloquear/desbloquear fechas) y enviar
pantallaDisp -> ctrlRecursos: Enviar actualización de disponibilidad
ctrlRecursos -> reservaEnt: Verificar conflictos con reservas
existentes
alt "Sin conflictos"
    reservaEnt --> ctrlRecursos: No hay conflictos
    ctrlRecursos -> recursoEnt: Actualizar disponibilidad del recurso
    recursoEnt --> ctrlRecursos: Actualización exitosa
    ctrlRecursos --> pantallaDisp: Confirmación de actualización
    pantallaDisp --> admin: Mostrar mensaje de éxito (disponibilidad
actualizada)
else "Con conflictos"
    reservaEnt --> ctrlRecursos: Conflictos detectados (reservas
existentes)
    ctrlRecursos --> pantallaDisp: Error: existen reservas en esas
fechas
    pantallaDisp --> admin: Mostrar mensaje de error (no se pudo
ajustar disponibilidad)
end alt
@enduml
```

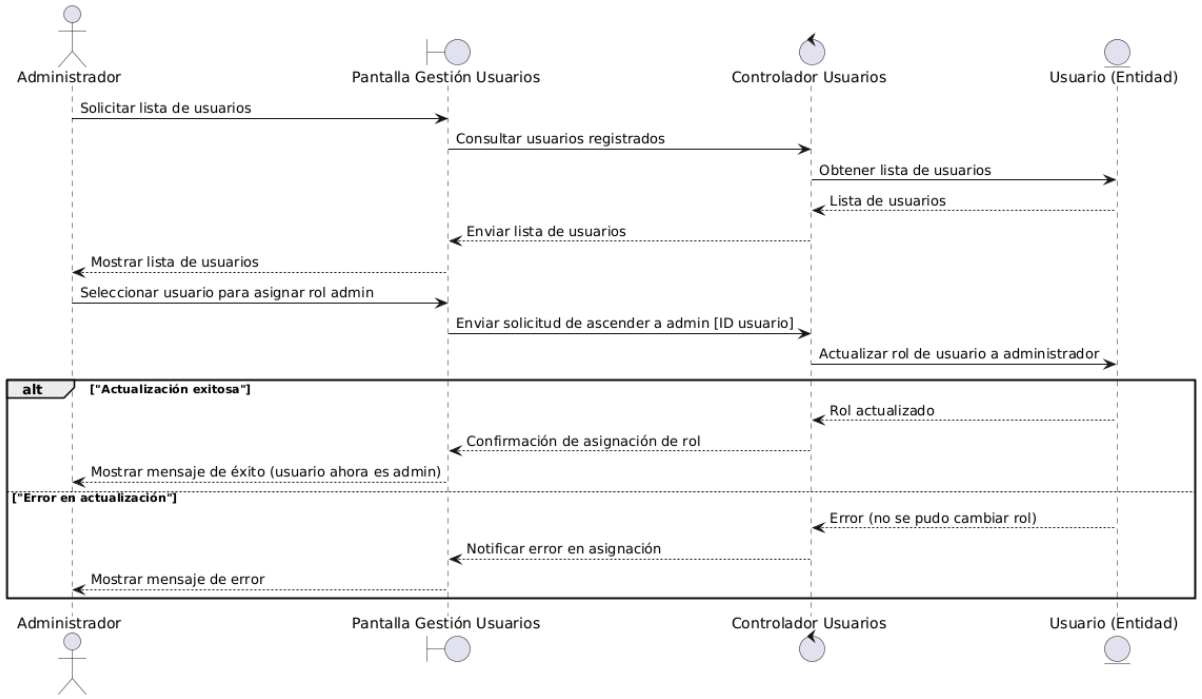


CU22 – Asignar rol de administrador: Un administrador otorga privilegios de administrador a un usuario existente (o revoca dichos privilegios).

Unset

```
@startuml
actor "Administrador" as admin
boundary "Pantalla Gestión Usuarios" as pantallaGestUsu
control "Controlador Usuarios" as ctrlUsuarios
entity "Usuario (Entidad)" as usuarioEnt

admin -> pantallaGestUsu: Solicitar lista de usuarios
pantallaGestUsu -> ctrlUsuarios: Consultar usuarios registrados
ctrlUsuarios -> usuarioEnt: Obtener lista de usuarios
usuarioEnt --> ctrlUsuarios: Lista de usuarios
ctrlUsuarios --> pantallaGestUsu: Enviar lista de usuarios
pantallaGestUsu --> admin: Mostrar lista de usuarios
admin -> pantallaGestUsu: Seleccionar usuario para asignar rol admin
pantallaGestUsu -> ctrlUsuarios: Enviar solicitud de ascender a admin
[ID usuario]
ctrlUsuarios -> usuarioEnt: Actualizar rol de usuario a administrador
alt "Actualización exitosa"
    usuarioEnt --> ctrlUsuarios: Rol actualizado
    ctrlUsuarios --> pantallaGestUsu: Confirmación de asignación de
rol
    pantallaGestUsu --> admin: Mostrar mensaje de éxito (usuario
ahora es admin)
else "Error en actualización"
    usuarioEnt --> ctrlUsuarios: Error (no se pudo cambiar rol)
```

```
    ctrlUsuarios --> pantallaGestUsu: Notificar error en asignación
    pantallaGestUsu --> admin: Mostrar mensaje de error
end alt
@enduml
```

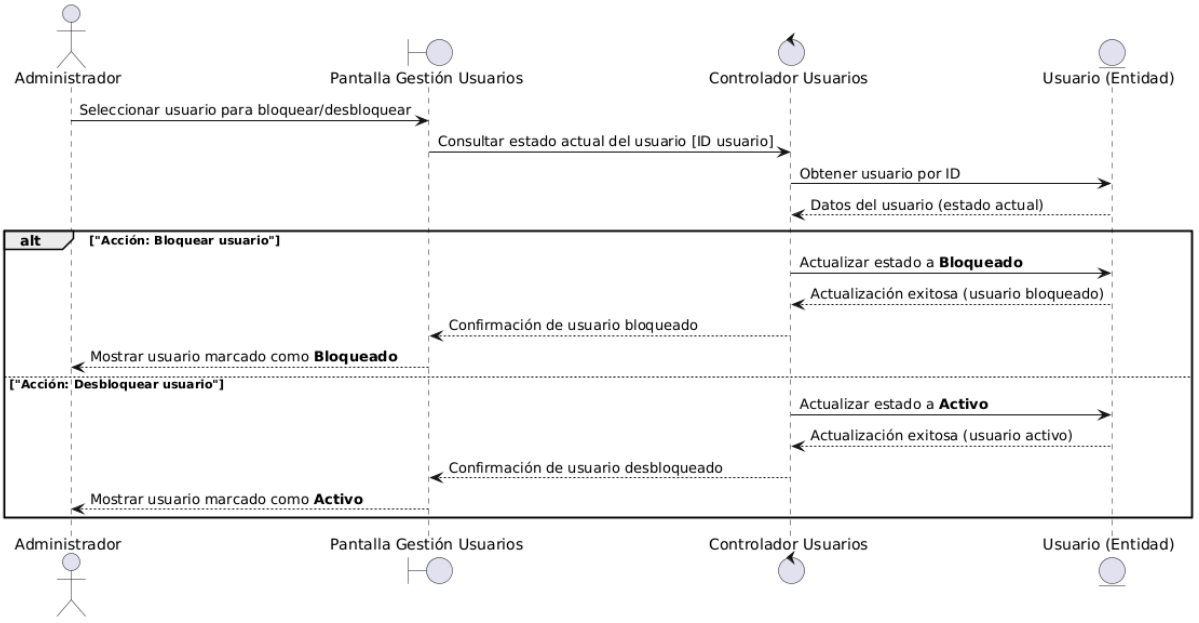


CU23 – Bloquear usuario: Un administrador inhabilita la cuenta de un usuario (impidiendo su acceso), o la reactiva.

Unset

```
@startuml
actor "Administrador" as admin
boundary "Pantalla Gestión Usuarios" as pantallaGestUsu
control "Controlador Usuarios" as ctrlUsuarios
entity "Usuario (Entidad)" as usuarioEnt

admin -> pantallaGestUsu: Seleccionar usuario para
bloquear/desbloquear
pantallaGestUsu -> ctrlUsuarios: Consultar estado actual del usuario
[ID usuario]
ctrlUsuarios -> usuarioEnt: Obtener usuario por ID
usuarioEnt --> ctrlUsuarios: Datos del usuario (estado actual)
alt "Acción: Bloquear usuario"
    ctrlUsuarios -> usuarioEnt: Actualizar estado a **Bloqueado**
```

```
    usuarioEnt --> ctrlUsuarios: Actualización exitosa (usuario
bloqueado)
    ctrlUsuarios --> pantallaGestUsu: Confirmación de usuario
bloqueado
    pantallaGestUsu --> admin: Mostrar usuario marcado como
**Bloqueado**
else "Acción: Desbloquear usuario"
    ctrlUsuarios -> usuarioEnt: Actualizar estado a **Activo**
    usuarioEnt --> ctrlUsuarios: Actualización exitosa (usuario
activo)
    ctrlUsuarios --> pantallaGestUsu: Confirmación de usuario
desbloqueado
    pantallaGestUsu --> admin: Mostrar usuario marcado como
**Activo**
end alt
@enduml
```

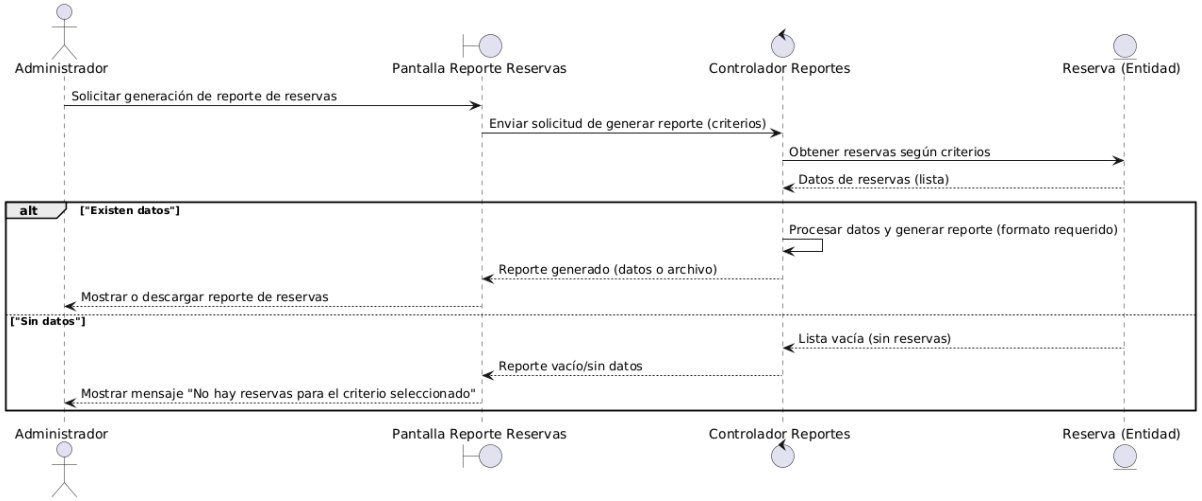


CU24 – Generar reporte de reservas: El administrador genera un informe o reporte del sistema de reservas (por ejemplo, estadísticas de reservas en un período).

Unset

```
@startuml
actor "Administrador" as admin
boundary "Pantalla Reporte Reservas" as pantallaReporte
control "Controlador Reportes" as ctrlReportes
entity "Reserva (Entidad)" as reservaEnt
```

```
admin -> pantallaReporte: Solicitar generación de reporte de reservas
pantallaReporte -> ctrlReportes: Enviar solicitud de generar reporte
(criterios)
ctrlReportes -> reservaEnt: Obtener reservas según criterios
reservaEnt --> ctrlReportes: Datos de reservas (lista)
alt "Existen datos"
    ctrlReportes -> ctrlReportes: Procesar datos y generar reporte
(formato requerido)
    ctrlReportes --> pantallaReporte: Reporte generado (datos o
archivo)
    pantallaReporte --> admin: Mostrar o descargar reporte de
reservas
else "Sin datos"
    reservaEnt --> ctrlReportes: Lista vacía (sin reservas)
    ctrlReportes --> pantallaReporte: Reporte vacío/sin datos
    pantallaReporte --> admin: Mostrar mensaje "No hay reservas para
el criterio seleccionado"
end alt
@enduml
```



3. DIAGRAMAS DE RUSTEZ POR CASO DE USO

CU01 – Registrarse:

Flujo Básico

1. El sistema muestra la ventana Registrar Usuario.
2. El Usuario ingresa su email y contraseña.

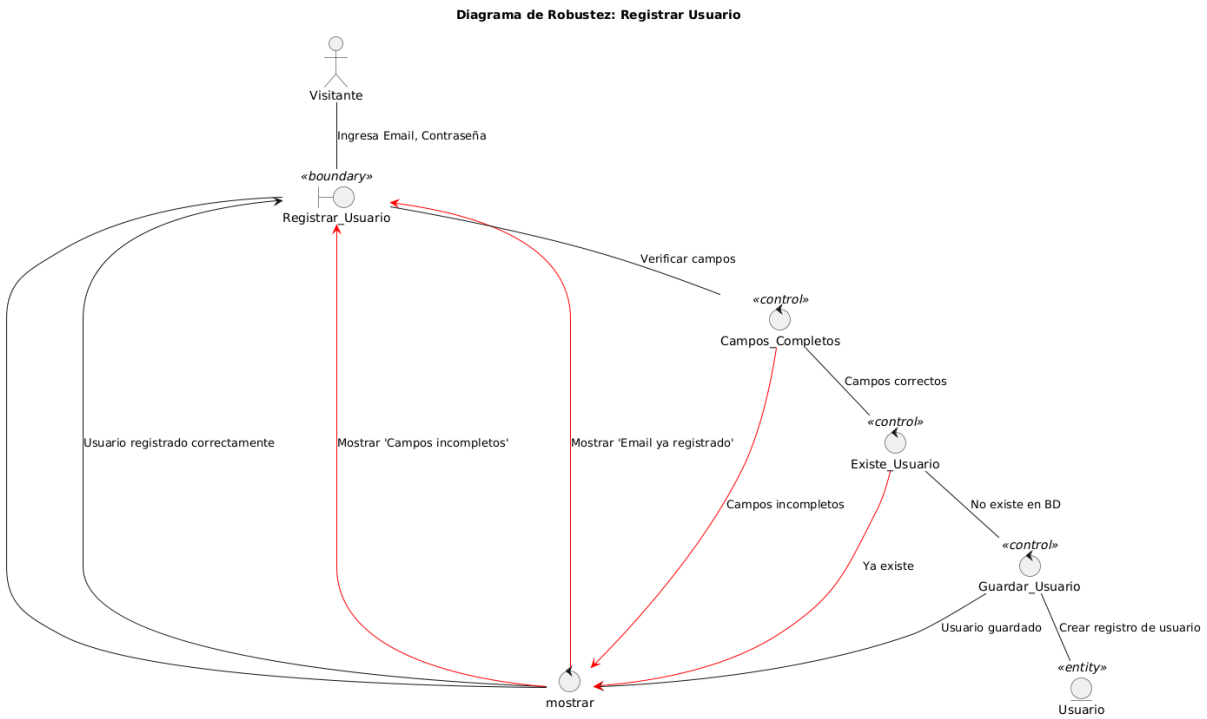
3. El sistema valida que los campos estén completos y que no exista ya un usuario con ese email.
4. El sistema guarda el nuevo usuario en la base de datos.
5. El sistema muestra un mensaje “Usuario registrado correctamente”.

Flujo Alternativo

- Campos incompletos: mensaje “Campos incompletos”.
- Email ya existente: mensaje “El email ya está registrado”.

Unset

```
@startuml
title Diagrama de Robustez: Registrar Usuario

actor "Visitante" as Visitante
control mostrar as cMostrar
boundary Registrar_Usuario as bRegUser <<boundary>>
control Campos_Completos as cCampos <<control>>
control Existe_Usuario as cExisteUser <<control>>
entity Usuario as eUser <<entity>>
control Guardar_Usuario as cGuardarUser <<control>>

cMostrar -- bRegUser

Visitante -- bRegUser : "Ingresa Email, Contraseña"
bRegUser -- cCampos : "Verificar campos"
cCampos -- cExisteUser : "Campos correctos"
cExisteUser -- cGuardarUser : "No existe en BD"
cGuardarUser -- eUser : "Crear registro de usuario"
cGuardarUser --> cMostrar : "Usuario guardado"
cMostrar --> bRegUser : "Usuario registrado correctamente"

cCampos -[#red]-> cMostrar : "Campos incompletos"
cMostrar -[#red]-> bRegUser : "Mostrar 'Campos incompletos'"

cExisteUser -[#red]-> cMostrar : "Ya existe"
cMostrar -[#red]-> bRegUser : "Mostrar 'Email ya registrado'"

@enduml
```



CU02 – Iniciar sesión: Un usuario (cliente o administrador) inicia sesión con sus credenciales.

Flujo Básico

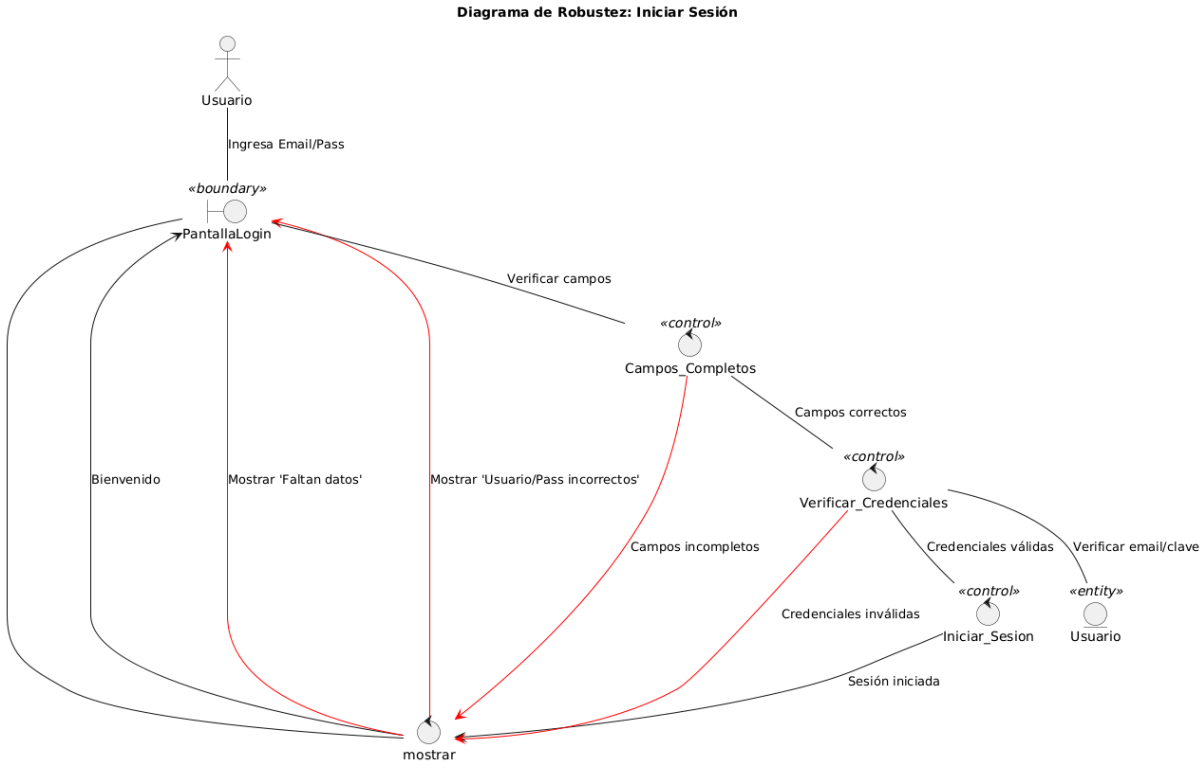
1. El sistema muestra la ventana Login.
2. El Usuario ingresa email y contraseña.
3. El sistema valida los campos.
4. El sistema verifica credenciales en la BD.
5. El sistema inicia la sesión y muestra “Bienvenido”.

Flujo Alternativo

- Campos incompletos: “Faltan datos”.
- Credenciales inválidas: “Usuario/Contraseña incorrecta”.

```
Unset
@startuml
title Diagrama de Robustez: Iniciar Sesión

actor "Usuario" as user
control mostrar as cMostrar
boundary PantallaLogin as bLogin <<boundary>>
```

```
control Campos_Completos as cCampos <<control>>
control Verificar_Credenciales as cCreds <<control>>
control Iniciar_Sesion as cSesion <<control>>
entity Usuario as eUser <<entity>>

cMostrar -- bLogin

user -- bLogin : "Ingresa Email/Pass"
bLogin -- cCampos : "Verificar campos"
cCampos -- cCreds : "Campos correctos"
cCreds -- eUser : "Verificar email/clave"
cCreds -- cSesion : "Credenciales válidas"
cSesion --> cMostrar : "Sesión iniciada"
cMostrar --> bLogin : "Bienvenido"

cCampos -[#red]-> cMostrar : "Campos incompletos"
cMostrar -[#red]-> bLogin : "Mostrar 'Faltan datos'"

cCreds -[#red]-> cMostrar : "Credenciales inválidas"
cMostrar -[#red]-> bLogin : "Mostrar 'Usuario/Pass incorrectos'"

@enduml
```



CU08 – Reservar recurso: Realizar una nueva reserva de un recurso (seleccionando fechas/horas disponibles).

Flujo Básico

1. El sistema muestra la ventana Reservar Recurso.
2. El Usuario ingresa la fecha/hora de reserva y selecciona el recurso.
3. El sistema valida los campos.
4. El sistema verifica si el recurso está disponible.
5. El sistema guarda la reserva en la base de datos.
6. El sistema muestra “Reserva creada correctamente”.

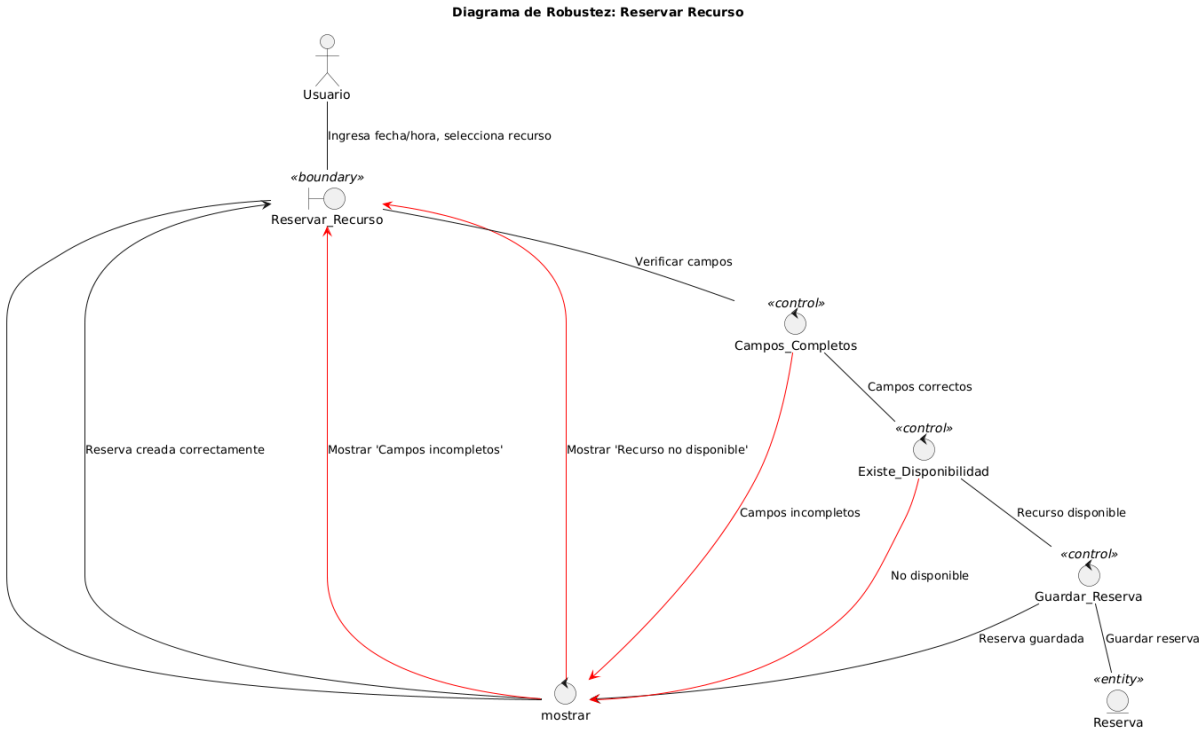
Flujos Alternativos

- Campos incompletos: “Campos incompletos”.
- Recurso no disponible: “El recurso no está disponible en ese horario”.

Unset

```
@startuml
title Diagrama de Robustez: Reservar Recurso

actor "Usuario" as user
control mostrar as cMostrar
boundary Reservar_Recurso as bReservar <<boundary>>
control Campos_Completos as cCampos <<control>>
control Existe_Disponibilidad as cDisp <<control>>
control Guardar_Reserva as cGuardar <<control>>
entity Reserva as eResv <<entity>>

cMostrar -- bReservar

user -- bReservar : "Ingresa fecha/hora, selecciona recurso"
bReservar -- cCampos : "Verificar campos"
cCampos -- cDisp : "Campos correctos"
cDisp -- cGuardar : "Recurso disponible"
cGuardar -- eResv : "Guardar reserva"
cGuardar --> cMostrar : "Reserva guardada"
cMostrar --> bReservar : "Reserva creada correctamente"

cCampos -[#red]-> cMostrar : "Campos incompletos"
cMostrar -[#red]-> bReservar : "Mostrar 'Campos incompletos'"

cDisp -[#red]-> cMostrar : "No disponible"
cMostrar -[#red]-> bReservar : "Mostrar 'Recurso no disponible'"
```

```
@enduml
```



CU12 – Cancelar reserva: Un cliente cancela una de sus reservas.

Flujo Básico

1. El sistema muestra la ventana Detalle de Reserva.
2. El Usuario hace clic en “Cancelar”.
3. El sistema solicita confirmación.
4. El sistema cancela la reserva en la base de datos.
5. El sistema muestra “Reserva cancelada”.

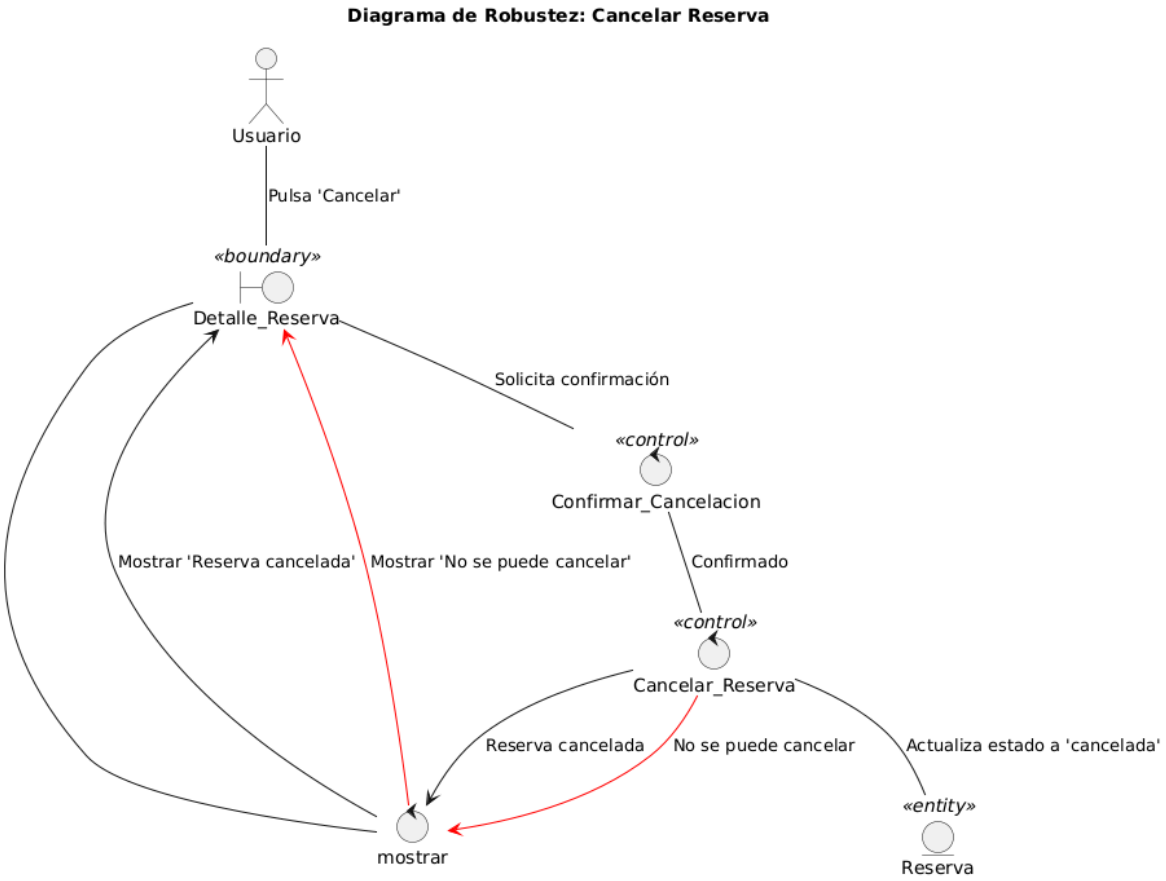
Flujo Alternativo

- Si la reserva ya fue cancelada o no existe, se muestra “No se puede cancelar”.

Unset

```
@startuml
title Diagrama de Robustez: Cancelar Reserva

actor "Usuario" as user
control mostrar as cMostrar
boundary Detalle_Reserva as bDetRes <<boundary>>
control Confirmar_Cancelacion as cConfirm <<control>>
```

```
control Cancelar_Reserva as cCancel <<control>>
entity Reserva as eResv <<entity>>

cMostrar -- bDetRes

user -- bDetRes : "Pulsa 'Cancelar'"
bDetRes -- cConfirm : "Solicita confirmación"
cConfirm -- cCancel : "Confirmado"
cCancel -- eResv : "Actualiza estado a 'cancelada'"
cCancel --> cMostrar : "Reserva cancelada"
cMostrar --> bDetRes : "Mostrar 'Reserva cancelada'"

cCancel -[#red]-> cMostrar : "No se puede cancelar"
cMostrar -[#red]-> bDetRes : "Mostrar 'No se puede cancelar'"

@enduml
```



CU13 – Editar perfil: El usuario actualiza los datos de su perfil (nombre, email, etc.).

Flujo Básico

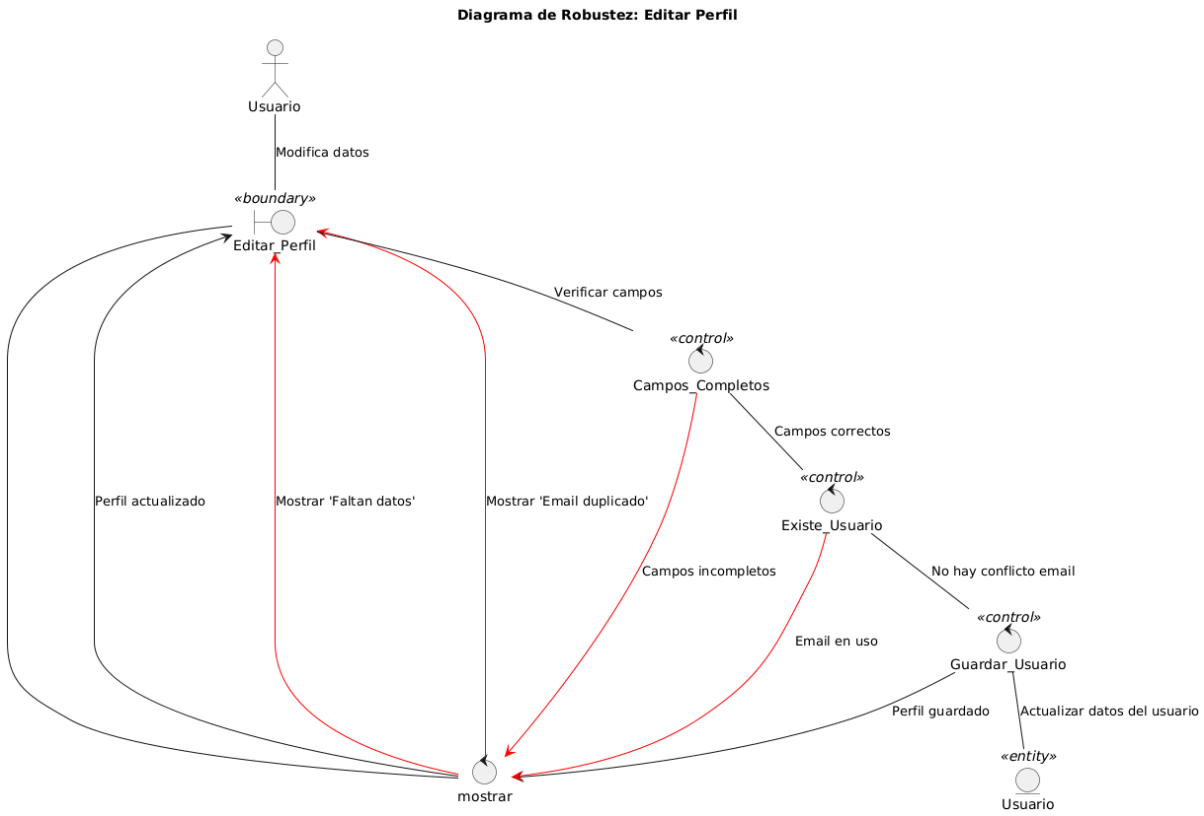
1. El sistema muestra la ventana Editar Perfil (mostrando datos actuales).
2. El Usuario modifica nombre, correo, etc.
3. El sistema valida campos.
4. El sistema guarda los cambios en la BD.
5. El sistema muestra “Perfil actualizado”.

Flujo Alternativo

- Campos incompletos: “Faltan datos”.
- El email ya está en uso por otro usuario: “Email duplicado”.

Unset

```
@startuml
title Diagrama de Robustez: Editar Perfil

actor "Usuario" as user
control mostrar as cMostrar
boundary Editar_Perfil as bEditPerfil <<boundary>>
control Campos_Completos as cCampos <<control>>
control Existe_Usuario as cExisteUser <<control>>
control Guardar_Usuario as cGuardar <<control>>
entity Usuario as eUser <<entity>>

cMostrar -- bEditPerfil

user -- bEditPerfil : "Modifica datos"
bEditPerfil -- cCampos : "Verificar campos"
cCampos -- cExisteUser : "Campos correctos"
cExisteUser -- cGuardar : "No hay conflicto email"
cGuardar -- eUser : "Actualizar datos del usuario"
cGuardar --> cMostrar : "Perfil guardado"
cMostrar --> bEditPerfil : "Perfil actualizado"

cCampos -[#red]-> cMostrar : "Campos incompletos"
cMostrar -[#red]-> bEditPerfil : "Mostrar 'Faltan datos'"

cExisteUser -[#red]-> cMostrar : "Email en uso"
cMostrar -[#red]-> bEditPerfil : "Mostrar 'Email duplicado'"

@enduml
```



CU14 – Cambiar contraseña: El usuario autenticado cambia su contraseña dentro del sistema.

Flujo Básico

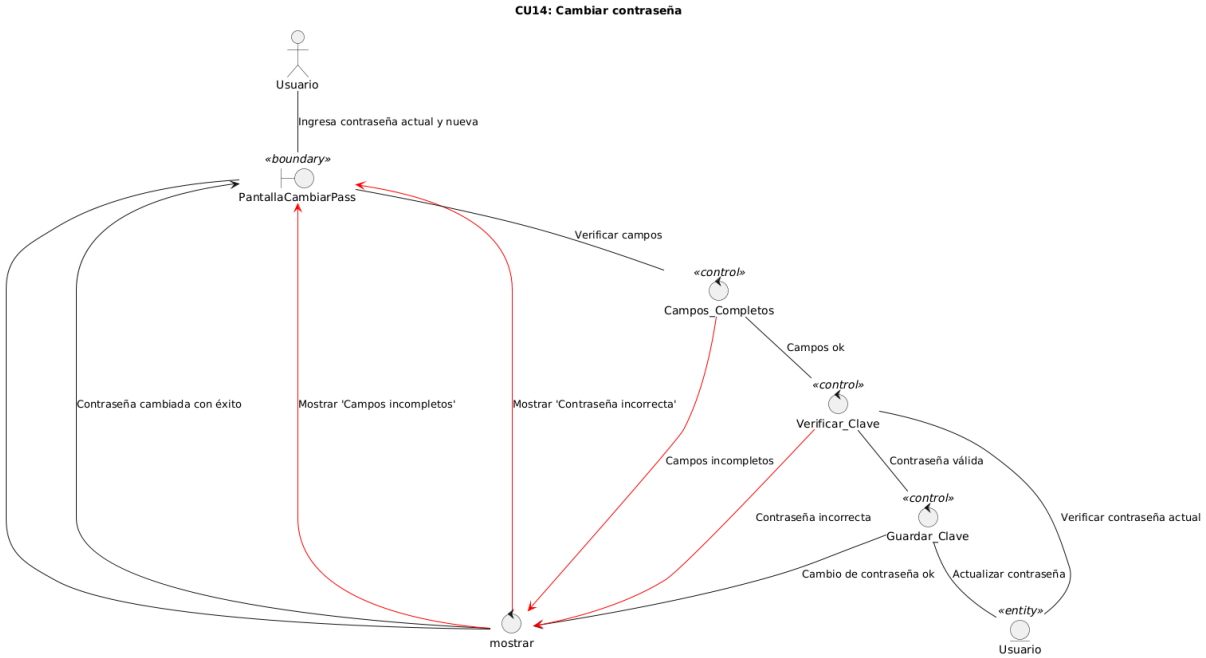
1. El usuario accede a la pantalla Cambiar Contraseña.
2. El usuario ingresa la contraseña actual y la nueva contraseña.
3. El sistema valida que los campos estén completos.
4. El sistema verifica que la contraseña actual sea correcta.
5. El sistema actualiza la contraseña en la entidad Usuario.
6. El sistema muestra el mensaje “Contraseña cambiada correctamente”.

Flujo Alternativo

- Campos incompletos: “Campos incompletos”.
- Contraseña actual inválida: “Contraseña incorrecta”.

```
Unset
@startuml
title CU14: Cambiar contraseña

actor "Usuario" as user
control mostrar as cMostrar
boundary PantallaCambiarPass as bCambiar <<boundary>>
```

```
control Campos_Completos as cCampos <<control>>
control Verificar_Clave as cClave <<control>>
control Guardar_Clave as cGuardar <<control>>
entity Usuario as eUser <<entity>>

cMostrar -- bCambiar

user -- bCambiar : "Ingresa contraseña actual y nueva"
bCambiar -- cCampos : "Verificar campos"
cCampos -- cClave : "Campos ok"
cClave -- eUser : "Verificar contraseña actual"
cClave -- cGuardar : "Contraseña válida"
cGuardar -- eUser : "Actualizar contraseña"
cGuardar --> cMostrar : "Cambio de contraseña ok"
cMostrar --> bCambiar : "Contraseña cambiada con éxito"

cCampos -[#red]-> cMostrar : "Campos incompletos"
cMostrar -[#red]-> bCambiar : "Mostrar 'Campos incompletos'"

cClave -[#red]-> cMostrar : "Contraseña incorrecta"
cMostrar -[#red]-> bCambiar : "Mostrar 'Contraseña incorrecta'"

@enduml
```



CU15 – Agregar recurso: Un administrador registra (da de alta) un nuevo recurso en el sistema.

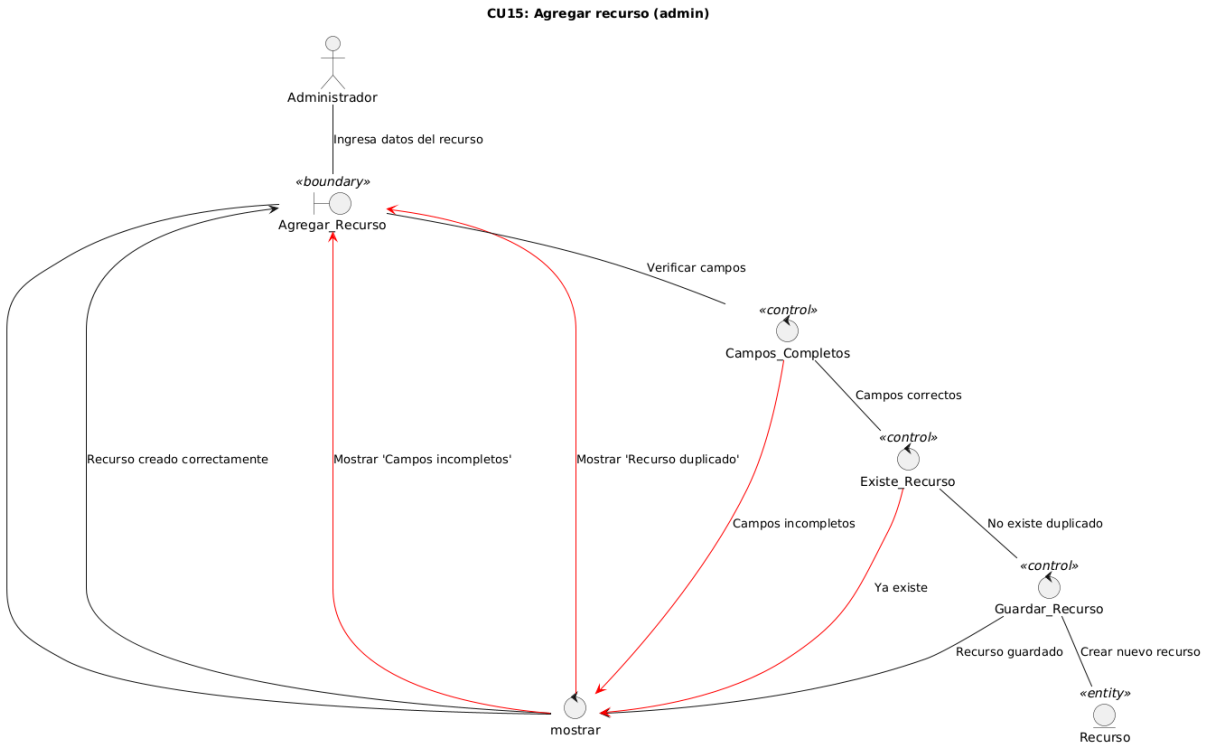
Flujo Básico

1. El administrador ingresa a la pantalla Agregar Recurso.
2. El sistema muestra los campos para nombre, descripción, ubicación, etc.
3. El administrador completa esos datos y presiona “Crear”.
4. El sistema verifica que los campos estén completos.
5. El sistema guarda el recurso en la base de datos.
6. El sistema muestra “Recurso creado correctamente”.

Flujos Alternativos

- Campos incompletos: “Campos incompletos”.
- (Opcional) Recurso con nombre duplicado: “Ya existe un recurso con ese nombre”.

```
Unset
@startuml
title CU15: Agregar recurso (admin)

actor "Administrador" as admin
control mostrar as cMostrar
boundary Agregar_Recurso as bAddRec <<boundary>>
control Campos_Completos as cCampos <<control>>
control Existe_Recurso as cExisteRec <<control>>
control Guardar_Recurso as cGuardar <<control>>
entity Recurso as eRes <<entity>>

cMostrar -- bAddRec

admin -- bAddRec : "Ingresa datos del recurso"
bAddRec -- cCampos : "Verificar campos"
cCampos -- cExisteRec : "Campos correctos"
cExisteRec -- cGuardar : "No existe duplicado"
cGuardar -- eRes : "Crear nuevo recurso"
cGuardar --> cMostrar : "Recurso guardado"
cMostrar --> bAddRec : "Recurso creado correctamente"

cCampos -[#red]-> cMostrar : "Campos incompletos"
cMostrar -[#red]-> bAddRec : "Mostrar 'Campos incompletos'"

cExisteRec -[#red]-> cMostrar : "Ya existe"
cMostrar -[#red]-> bAddRec : "Mostrar 'Recurso duplicado'"

@enduml
```



CU16 – Editar recurso: Un administrador edita la información de un recurso existente.

Unset

CU17 – Eliminar recurso: Un administrador elimina (o deshabilita) un recurso del sistema.

Unset

CU18 – Ver todas las reservas: El administrador consulta el listado de todas las reservas realizadas en el sistema.

Flujo Básico

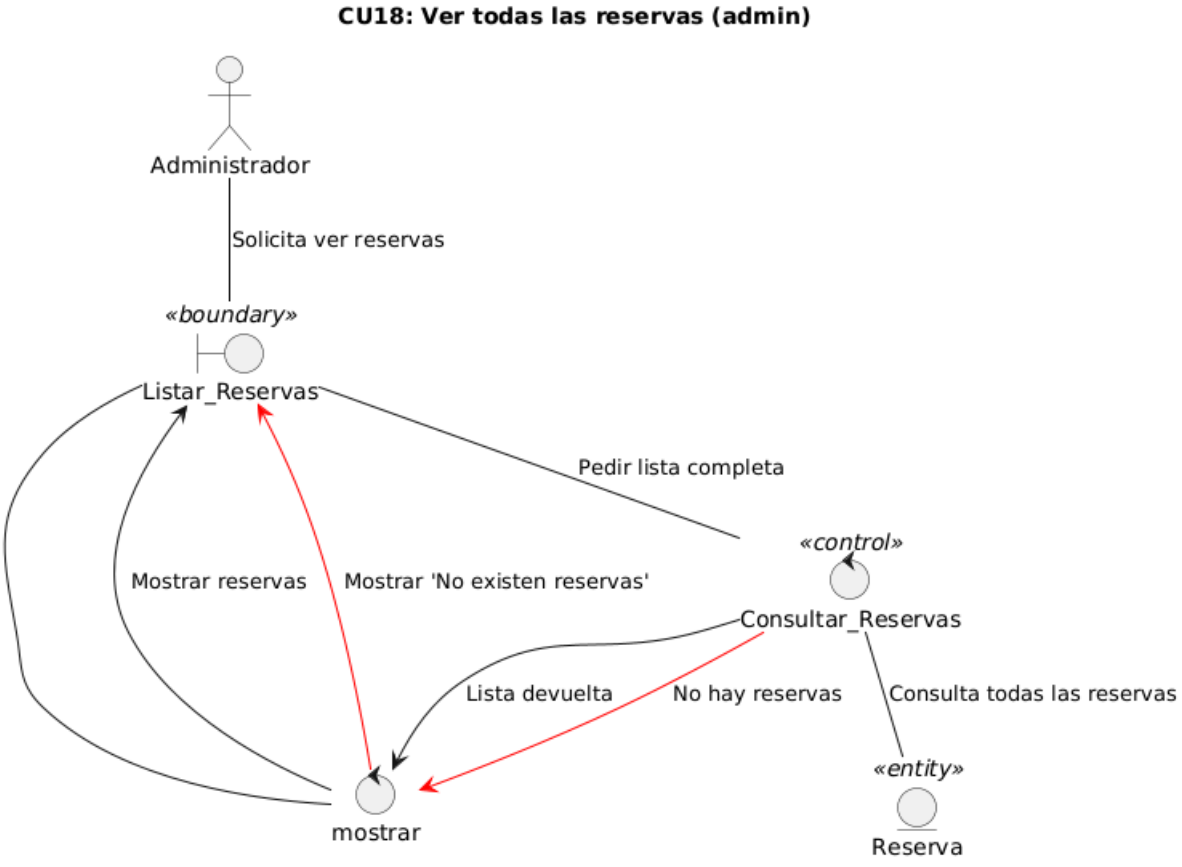
1. El administrador accede a la pantalla Listar Reservas.
2. El sistema consulta todas las reservas en la BD.
3. El sistema muestra la lista completa de reservas.

Flujos Alternativos

- (Opcional) Si no hay reservas, se muestra “No existen reservas registradas”.

Unset

```
@startuml
title CU18: Ver todas las reservas (admin)

actor "Administrador" as admin
control mostrar as cMostrar
boundary Listar_Reservas as bListRes <<boundary>>
control Consultar_Reservas as cReservas <<control>>
entity Reserva as eResv <<entity>>

cMostrar -- bListRes

admin -- bListRes : "Solicita ver reservas"
bListRes -- cReservas : "Pedir lista completa"
cReservas -- eResv : "Consulta todas las reservas"
cReservas --> cMostrar : "Lista devuelta"
cMostrar --> bListRes : "Mostrar reservas"

cReservas -[#red]-> cMostrar : "No hay reservas"
cMostrar -[#red]-> bListRes : "Mostrar 'No existen reservas'"

@enduml
```



CU20 – Cancelar reserva de usuario: El administrador cancela una reserva en nombre de un cliente (por ejemplo, ante una incidencia).

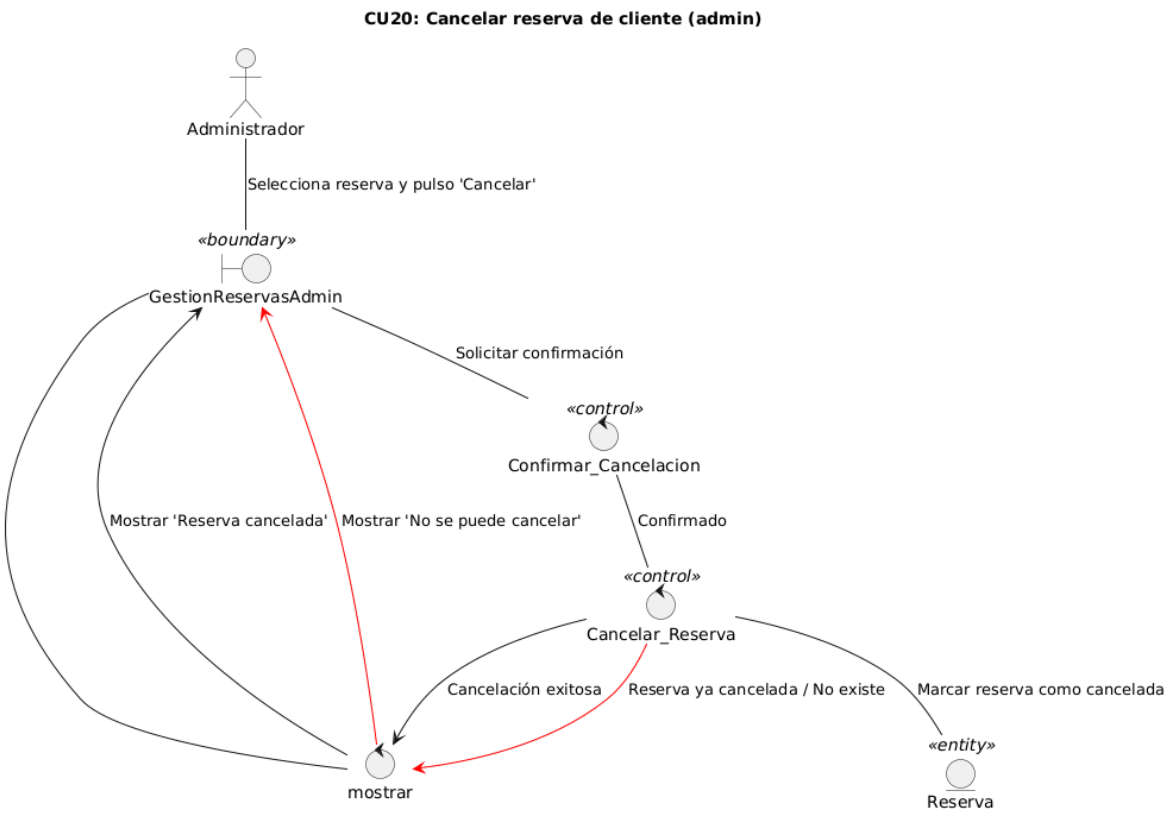
Flujo Básico

1. El administrador ingresa a la pantalla de Gestión de Reservas, selecciona la reserva de un cliente.
2. El administrador hace clic en “Cancelar” reserva.
3. El sistema solicita confirmación.
4. El sistema marca la reserva como cancelada.
5. El sistema muestra “Reserva cancelada correctamente”.

Flujos Alternativos

- La reserva ya está cancelada: “La reserva ya estaba cancelada”.
- La reserva no existe (ID inválido): “No existe la reserva”.

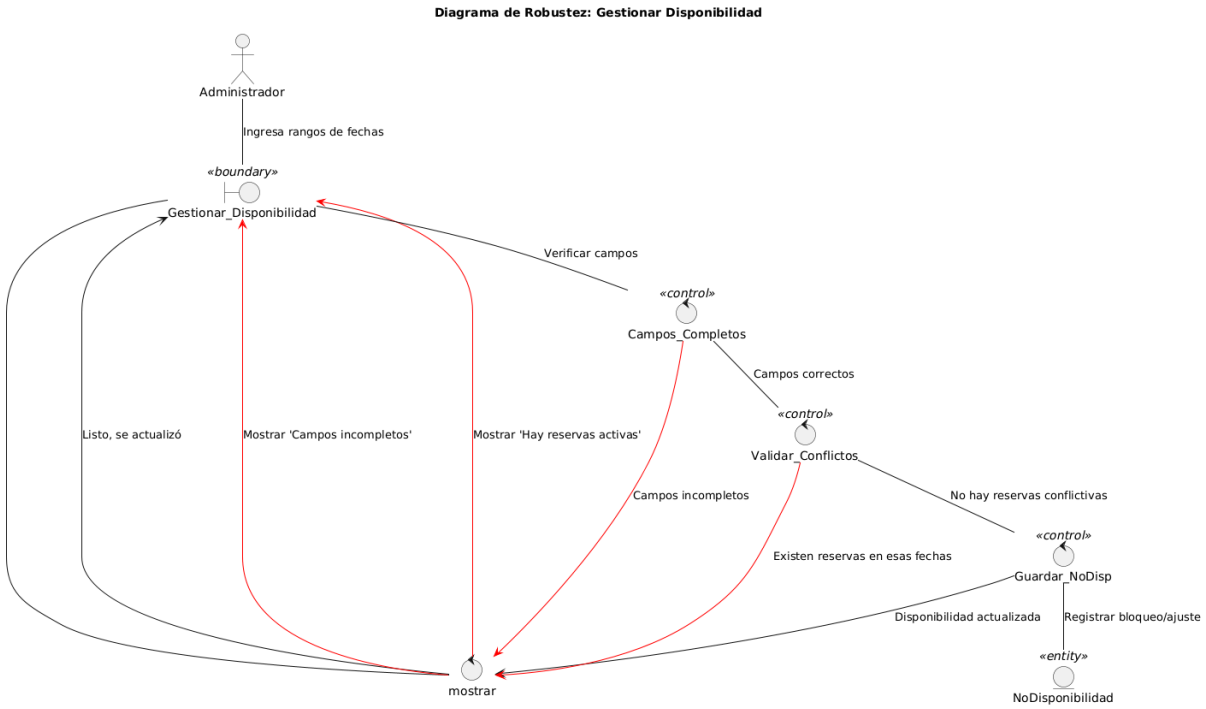
```
Unset
@startuml
title CU20: Cancelar reserva de cliente (admin)

actor "Administrador" as admin
control mostrar as cMostrar
boundary GestionReservasAdmin as bGestRes <<boundary>>
control Confirmar_Cancelacion as cConfirm <<control>>
control Cancelar_Reserva as cCancel <<control>>
entity Reserva as eResv <<entity>>

cMostrar -- bGestRes

admin -- bGestRes : "Selecciona reserva y pulso 'Cancelar'"
bGestRes -- cConfirm : "Solicitar confirmación"
cConfirm -- cCancel : "Confirmado"
cCancel -- eResv : "Marcar reserva como cancelada"
cCancel --> cMostrar : "Cancelación exitosa"
cMostrar --> bGestRes : "Mostrar 'Reserva cancelada'"

cCancel -[#red]-> cMostrar : "Reserva ya cancelada / No existe"
cMostrar -[#red]-> bGestRes : "Mostrar 'No se puede cancelar'"

@enduml
```



CU21 – Gestionar disponibilidad de recurso: El administrador bloquea o habilita rangos de fechas/horas en los que un recurso no estará disponible para reserva (por mantenimiento u otros motivos).

```
Unset
@startuml
title Diagrama de Robustez: Gestionar Disponibilidad

actor "Administrador" as admin
control mostrar as cMostrar
boundary Gestionar_Disponibilidad as bGDisp <<boundary>>
control Campos_Completos as cCampos <<control>>
control Validar_Conflictos as cConflictos <<control>>
control Guardar_NoDisp as cGuardar <<control>>
entity NoDisponibilidad as eNoDisp <<entity>>

cMostrar -- bGDisp

admin -- bGDisp : "Ingresa rangos de fechas"
bGDisp -- cCampos : "Verificar campos"
cCampos -- cConflictos : "Campos correctos"
cConflictos -- cGuardar : "No hay reservas conflictivas"
```

```
cGuardar -- eNoDisp : "Registrar bloqueo/ajuste"
cGuardar --> cMostrar : "Disponibilidad actualizada"
cMostrar --> bGDisp : "Listo, se actualizó"

cCampos -[#red]-> cMostrar : "Campos incompletos"
cMostrar -[#red]-> bGDisp : "Mostrar 'Campos incompletos'"

cConflictos -[#red]-> cMostrar : "Existen reservas en esas fechas"
cMostrar -[#red]-> bGDisp : "Mostrar 'Hay reservas activas'"

@enduml
```



CU23 – Bloquear usuario: Un administrador inhabilita la cuenta de un usuario (impidiendo su acceso), o la reactiva.

Flujo Básico

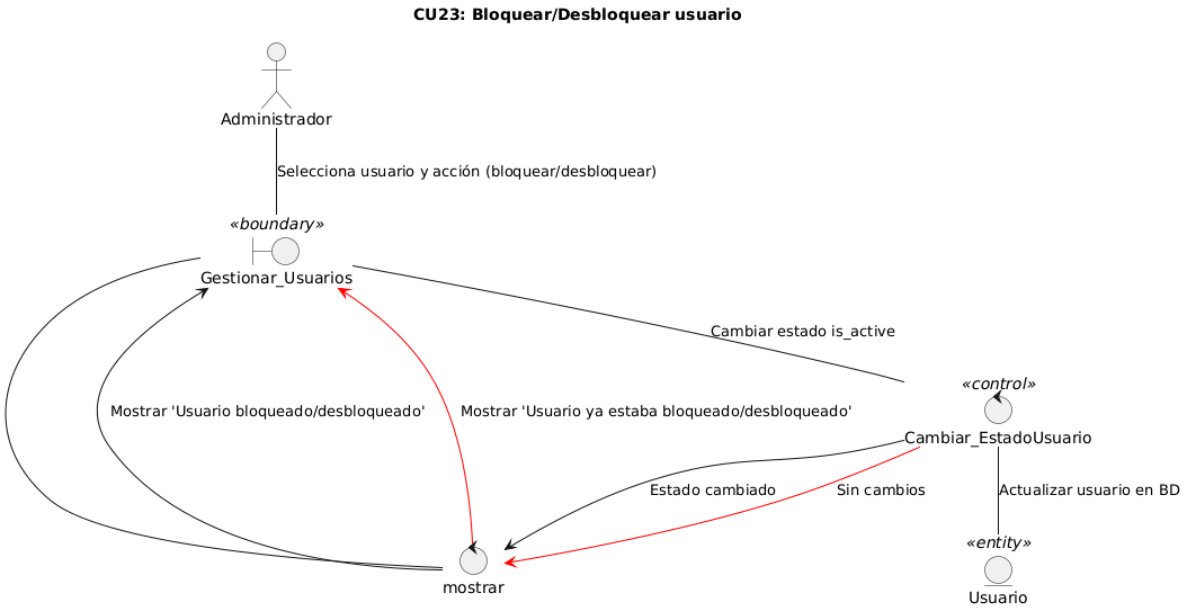
1. El administrador ve la pantalla de Gestión de Usuarios.
2. Selecciona un usuario en la lista y decide bloquear o desbloquearlo.
3. El sistema actualiza el atributo “is_active” del usuario en la BD.
4. El sistema muestra “Usuario bloqueado” o “Usuario desbloqueado” según la acción.

Flujos Alternativos

- El usuario ya estaba bloqueado/desbloqueado: “No se cambió el estado”.

Unset

```
@startuml
title CU23: Bloquear/Desbloquear usuario

actor "Administrador" as admin
control mostrar as cMostrar
boundary Gestionar_Usuarios as bGUser <<boundary>>
control Cambiar_EstadoUsuario as cEstado <<control>>
entity Usuario as eUser <<entity>>

cMostrar -- bGUser

admin -- bGUser : "Selecciona usuario y acción
(bloquear/desbloquear)"
bGUser -- cEstado : "Cambiar estado is_active"
cEstado -- eUser : "Actualizar usuario en BD"
cEstado --> cMostrar : "Estado cambiado"
cMostrar --> bGUser : "Mostrar 'Usuario bloqueado/desbloqueado'"

cEstado -[#red]-> cMostrar : "Sin cambios"
cMostrar -[#red]-> bGUser : "Mostrar 'Usuario ya estaba
bloqueado/desbloqueado'"

@enduml
```



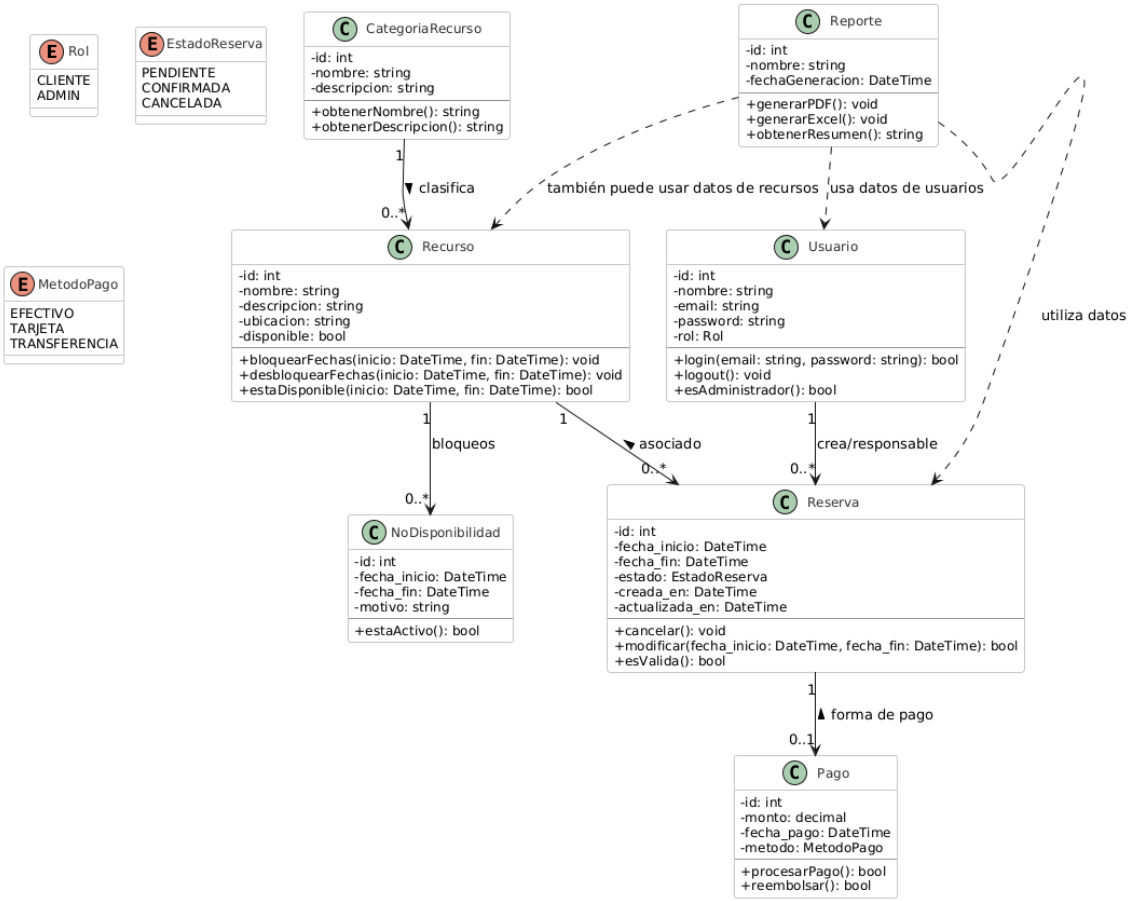
4. DIAGRAMA DE CLASES

Unset

```
@startuml
skinparam classAttributeIconSize 0
skinparam classFontColor #333
skinparam classFontSize 14
skinparam classAttributeFontSize 12
skinparam classBackgroundColor #FFFFFF
skinparam classBorderColor #999999
skinparam shadowing false

enum Rol {
  CLIENTE
  ADMIN
}

enum EstadoReserva {
  PENDIENTE
  CONFIRMADA
  CANCELADA
}

enum MetodoPago {
  EFECTIVO
  TARJETA
  TRANSFERENCIA
}

class Usuario {
  -id: int
  -nombre: string
  -email: string
  -password: string
  -rol: Rol
  --
  +login(email: string, password: string): bool
  +logout(): void
  +esAdministrador(): bool
}

class CategoriaRecurso {
```

```
  -id: int
  -nombre: string
  -descripcion: string
  --
  +obtenerNombre(): string
  +obtenerDescripcion(): string
}

class Recurso {
  -id: int
  -nombre: string
  -descripcion: string
  -ubicacion: string
  -disponible: bool
  --
  +bloquearFechas(inicio: DateTime, fin: DateTime): void
  +desbloquearFechas(inicio: DateTime, fin: DateTime): void
  +estaDisponible(inicio: DateTime, fin: DateTime): bool
}

class Reserva {
  -id: int
  -fecha_inicio: DateTime
  -fecha_fin: DateTime
  -estado: EstadoReserva
  -creada_en: DateTime
  -actualizada_en: DateTime
  --
  +cancelar(): void
  +modificar(fecha_inicio: DateTime, fecha_fin: DateTime): bool
  +esValida(): bool
}

class NoDisponibilidad {
  -id: int
  -fecha_inicio: DateTime
  -fecha_fin: DateTime
  -motivo: string
  --
  +estaActivo(): bool
}

class Pago {
```

```
  -id: int
  -monto: decimal
  -fecha_pago: DateTime
  -metodo: MetodoPago
  --
  +procesarPago(): bool
  +reembolsar(): bool
}

class Reporte {
  -id: int
  -nombre: string
  -fechaGeneracion: DateTime
  --
  +generarPDF(): void
  +generarExcel(): void
  +obtenerResumen(): string
}

Usuario "1" --> "0..*" Reserva : crea/responsable
Recurso "1" --> "0..*" Reserva : < asociado
Recurso "1" --> "0..*" NoDisponibilidad : bloqueos
CategoriaRecurso "1" --> "0..*" Recurso : clasifica >
Reserva "1" --> "0..1" Pago : < forma de pago

Reporte ..> Reserva : utiliza datos
Reporte ..> Usuario : usa datos de usuarios
Reporte ..> Recurso : también puede usar datos de recursos

@enduml
```


Rol
CLIENTE
ADMIN
EstadoReserva
PENDIENTE
CONFIRMADA
CANCELADA
CategoriaRecurso
-id: int
-nombre: string
-descripcion: string
+obtenerNombre(): string
+obtenerDescripcion(): string
Reporte
-id: int
-nombre: string
-fechaGeneracion: DateTime
+generarPDF(): void
+generarExcel(): void
+obtenerResumen(): string
1
clasifica
0..*
también puede usar datos de recursos
usa datos de usuarios
utiliza datos
MetodoPago
EFECTIVO
TARJETA
TRANSFERENCIA
Recurso
-id: int
-nombre: string
-descripcion: string
-ubicacion: string
-disponible: bool
+bloquearFechas(inicio: DateTime, fin: DateTime): void
+desbloquearFechas(inicio: DateTime, fin: DateTime): void
+estaDisponible(inicio: DateTime, fin: DateTime): bool
Usuario
-id: int
-nombre: string
-email: string
-password: string
-rol: Rol
+login(email: string, password: string): bool
+logout(): void
+esAdministrador(): bool
1
bloqueos
0..*
1
asociado
0..*
1
crea/responsable
0..*
NoDisponibilidad
-id: int
-fecha_inicio: DateTime
-fecha_fin: DateTime
-motivo: string
+estaActivo(): bool
Reserva
-id: int
-fecha_inicio: DateTime
-fecha_fin: DateTime
-estado: EstadoReserva
-creada_en: DateTime
-actualizada_en: DateTime
+cancelar(): void
+modificar(fecha_inicio: DateTime, fecha_fin: DateTime): bool
+esValida(): bool
1
forma de pago
0..1
Pago
-id: int
-monto: decimal
-fecha_pago: DateTime
-metodo: MetodoPago
+procesarPago(): bool
+reembolsar(): bool