| Ejercicio 1 |
| --- |
| **sequenceDiagram**  **autonumber**  **Actor Cliente**  **participant Reservas**  **participant Pagos**  **participant Notificaciones**    **Cliente ->>+ Reservas: seleccionarVuelo(infoUsuario, infoVuelo)**  **Reservas ->> Reservas: ConfirmarDisponibilidadVuelo(infoVuelo)**    **alt SI hay disponibilidad de vuelo**  **Reservas -->>- Cliente: Retorna [Solicitud de reserva]**  **Cliente ->>+ Reservas: ConfirmaVuelo(infoVuelo)**  **Reservas ->>+ Pagos: ComienzaProcesoDePagos(callback)**  **Pagos -->>- Reservas: Retorna [webAPI]**  **Reservas -->>- Cliente: Retorna [WebAPI]**  **Cliente ->>+ Reservas: ProcesaWebAPI(infoPago)**  **Reservas ->>+ Pagos: ProcesaPago(infoPago)**  **Pagos -->>- Reservas: Retorna [resultado del pago]**  **Reservas ->> Reservas: VerificaEstadoPago(resultado del pago)**  **alt SI el resultado es aprobado**  **Reservas ->> Reservas: ConfirmaReserva(infoUser, infoVuelo)**  **Reservas ->>+ Notificaciones: SolicitudDeNotificacion(Reserva, infoUser)**  **Notificaciones ->>- Cliente: Notificacion [confirmacion de Reserva]**  **end**  **alt SI el resultado es rechazado**  **Reservas ->> Cliente: Informa pago rechazado y pide metodo alternativo o que cancele la reserva**  **end**  **end**  **alt si no hay disponibilidad de vuelo**  **Reservas -->>- Cliente: Retorna No hay disponibilidad de vuelo**  **end** |

| Ejercicio 2 |
| --- |
| **sequenceDiagram**  **autonumber**  **Actor Usuario**  **participant Sistema de Streaming**  **participant Sistema de Gestion de Perfiles**  **participant Sistema de Recomendaciones**  **Usuario ->>+ Sistema de Streaming: registroPerfil(infoUsuario)**  **Sistema de Streaming ->>+ Sistema de Gestion de Perfiles: crearUsuario(infoUsuario)**  **Sistema de Gestion de Perfiles -->>- Sistema de Streaming : retorna[vinculacion de Cuenta]**  **Sistema de Streaming -->>- Usuario: retorna [vinculacion de cuenta]**  **Usuario ->>+ Sistema de Streaming: iniciarSesion(infoUsuario)**  **Sistema de Streaming ->>+ Sistema de Gestion de Perfiles: solicitaUsuario(infoUsuario)**  **Sistema de Gestion de Perfiles -->>- Sistema de Streaming: retorna[usuario]**  **Sistema de Streaming ->>+ Sistema de Recomendaciones: generarRecomendaciones(usuario)**  **Sistema de Recomendaciones ->> Sistema de Recomendaciones: Analiza y genera recomendaciones en base al usuario**  **Sistema de Recomendaciones -->>- Sistema de Streaming: retorna [Lista de Recomendaciones]**  **Sistema de Streaming -->>- Usuario: muestra esa lista de recomendaciones** |

| Ejercicio 3 |
| --- |
| **sequenceDiagram**  **autonumber**  **Actor Usuario**  **participant Sistema de Red Social de Cocina**  **participant Sistema de Gestion de Perfiles**  **participant Sistema de Gestion de Recetas**  **Usuario ->>+ Sistema de Red Social de Cocina: registrarse(infoUsuario)**  **Sistema de Red Social de Cocina ->>+ Sistema de Gestion de Perfiles: solicitudNuevoPerfil(infoUsuario)**  **Sistema de Gestion de Perfiles -->>- Sistema de Red Social de Cocina: retorna[vinculacionCuentaNueva]**  **Sistema de Red Social de Cocina -->>- Usuario: retorna[vinculacionCuentaNueva]**  **Usuario ->>+ Sistema de Red Social de Cocina: iniciaSesion(infoUsuario)**  **Sistema de Red Social de Cocina ->>+ Sistema de Gestion de Perfiles: solicitaPerfilUsuario(infoUsuario)**  **Sistema de Gestion de Perfiles -->>- Sistema de Red Social de Cocina: retorna[perfil del Usuario]**  **Sistema de Red Social de Cocina -->>- Usuario: retorna[perfil del Usuario]**  **Usuario ->>+ Sistema de Red Social de Cocina: nuevaReceta(receta)**  **Sistema de Red Social de Cocina ->>+ Sistema de Gestion de Recetas: guardarReceta(receta)**  **Sistema de Gestion de Recetas -->>- Sistema de Red Social de Cocina: retorna[confirmacion de la receta guardada]**  **Sistema de Red Social de Cocina -->>- Usuario: retorna [muestra la nueva receta]** |

| Ejercicio 4 |
| --- |
| **sequenceDiagram**  **autonumber**  **Actor Usuario**  **participant Chatbot de Whatsapp**  **participant Base de Datos de Restaurantes**  **participant API de Google Maps**  **Usuario ->>+ Chatbot de Whatsapp: solicitaRestaurantes**  **Chatbot de Whatsapp ->> Chatbot de Whatsapp: procesa solicitud**  **Chatbot de Whatsapp -->> Usuario: pideUbicacion**  **Usuario ->> Chatbot de Whatsapp: envia ubicacion**  **Chatbot de Whatsapp ->>+ Base de Datos de Restaurantes: restaurantesCercanos(ubicacion)**  **Base de Datos de Restaurantes -->>- Chatbot de Whatsapp: retorna[lista con restaurantes]**  **Chatbot de Whatsapp -->>- Usuario: retorna[lista con restaurantes]**  **Usuario ->>+ Chatbot de Whatsapp: indicacionesComoLlegar(restauranteSeleccionado)**  **Chatbot de Whatsapp ->>+ API de Google Maps: solicitudIndicaciones(ubicacionUsuario, ubicacionRestaurante)**  **API de Google Maps ->> API de Google Maps: procesa solicitud y genera indicaciones**  **API de Google Maps -->>- Chatbot de Whatsapp: retorna[indicaciones de como llegar al restaurante]**  **Chatbot de Whatsapp ->> Chatbot de Whatsapp: procesa las indicaciones recibidas**  **Chatbot de Whatsapp -->>- Usuario: retorna[mensaje con las indicaciones de como llegar al restaurante]** |