**VICTUS Residencias**

En este documento se da un primer vistazo a la propuesta de VICTUS residencias, que es un producto administrativo orientado a facilitar la administración de espacios comunes en lugares residenciales con administración activa.

**Contexto**

Para las personas que viven en parques o conjuntos residenciales, es común encontrar de manera activa, sistemas de administración de reservas con papel, siendo estos poco prácticos y con una facilidad alta a presentar errores

**1.**  **Por qué estoy aquí?**

**a. Juan Daniel Rodríguez Giraldo:** Para programar en el entorno de desarrollo InteliJ y en la interfaz del proyecto, valor agregado: creativo y novedoso, Desventaja: Tiene muchas obligaciones, tanto laborales como académicas.

**b. Juan Pablo Avendano Duque:** El dueño de la idea principal y propósito y con los contactos posibles a donde se puede aplicar, y conocimiento de la comunidad objetivo del proyecto.

**c. Andres Felipe Velez Alcaraz:** Para encargarme del diseño y desarrollo del software también coordinando tareas del equipo y asegurando que se cumplan los plazos y objetivos.

**d. Luis Miguel Luis Miguel Gaviria Rendon:** Encargado de coordinar y asegurar que se cumplan los plazos y objetivos, además de ayudar con las diversas tareas del equipo aplicando los conocimientos adquiridos.

**2.**  **Elevator pitch**

**a. Para:** Es un proyecto enfocado para la administración de espacios públicos en zonas residenciales, que presenta una administración activa.

**b. Que tiene:** La necesidad de una organización mayor y método simple de acceso a la información necesaria para la administración de espacios comunes dentro de una zona residencial administrada, la cual tiene cupos máximos, que a falta de administración puede generar conflictos a la hora del uso compartido de estas zonas.

**c. Nuestro producto**

**i. Nombre:** VICTUS Residencias

**d. Es un:** Producto de tipo administrativo.

**e. Que:** Facilita la información de reservas entre residentes y su comunicación a la hora de interactuar con los espacios comunes.

**f. A diferencia de:**  Los métodos tradicionales basados en papel o soluciones digitales genéricas no diseñadas específicamente para la administración residencial, VICTUS Residencias ofrece una plataforma centralizada, intuitiva y en tiempo real que:

* Elimina errores humanos mediante automatización de reservas y seguimiento de cupos.
* Facilita acceso inmediato a la información para residentes y administradores, sin trámites físicos o demoras.
* Integra comunicación directa entre usuarios y administración, evitando conflictos por malentendidos o falta de claridad.
* Se adapta exclusivamente a las necesidades de conjuntos residenciales, con funciones como control de aforo, historial de reservas y notificaciones personalizadas.

**3. Visión box**

****

**4.**  **Qué sí – Qué no:**

**Qué sí:** El proyecto incluirá funcionalidades como:

* Creación, modificación, eliminación y consulta de personas como Administradores, Residentes y Porteros que serán los usuarios del proyecto.
* Gestión de conjuntos residenciales, zonas comunes y Agendas específicas para cada zona común.
* Sistema de distribución de turnos en intervalos con un tiempo específico según el tiempo de inicio y final para cada zona común.
* Sistema de cambios en tiempo real para mostrar si un turno ya esta reservado o no para evitar solapamiento de reservas.
* Notificaciones automáticas para los residentes con el propósito de informar el estado de la reserva realizada en tiempo real.
* Sistema de confirmación de reservas, como una asistencia del Residente a la zona común, que va a usar siendo el usuario del Portero con el fin de visualizar que reservas hay en la zona común específica como una programación y dar Check o no Check a la reserva para así sancionar el residente que reservo y no asistió.
* Sistema de reportes con el fin de saber que tanto están reservando una zona común, las reservas que ha realizado un residente en cierto tiempo, cantidad de invitado que lleva un residente y datos de satisfacción de la app por los residentes para toma de decisiones administrativas.

**Qué no:** No se desarrollarán funcionalidades relacionadas con:

* La gestión de pagos o finanzas.
* Integración con sistemas externos como cámaras de seguridad.
* La automatización de servicios como limpieza o mantenimiento.
* Un portal de comunicación entre los residentes y la administración.

**En duda:**

* La implementación de una app móvil nativa está en discusión, dependiendo de la disponibilidad de recursos y el tiempo de desarrollo.

Seguros de hacer: La creación de un portal web que gestione las reservas de espacio, así como la base de datos que soporte toda la información de los usuarios y reservas.

**5. La comunidad:**



**Roles Principales:**

**Administradores y Porteros del conjunto:** Usuarios clave que gestionan las reservas y reciben reportes.

**Residentes:** Beneficiarios directos que utilizarán la aplicación para hacer reservas y recibir notificaciones.

**Sectores u Organizaciones:** Sectores Residenciales y proveedores de servicios para sectores residenciales

**Equipo de Desarrollo:** Compuesto por los cuatro estudiantes, quienes diseñan y desarrollan el sistema.

**Beneficiados:**

**Residentes:** Ahorro de tiempo y facilidad para hacer reservas.

**Administradores:** Mejora en la gestión de espacios y comunicación eficiente con los residentes.

**Porteros:** Ayuda a la gestión rápida de reservas y uso de espacios compartidos para un seguimiento mejor.

**Comprometidos:**

**Equipo de Desarrollo:** Enfocados en la implementación y éxito del proyecto.

**Administración del conjunto:** Comprometidos con la adopción y uso del sistema.

6. La solución:

La solución consiste en desarrollar un sistema web que facilite la gestión de reservas de espacios comunes en conjuntos residenciales. La aplicación incluirá un calendario interactivo, un sistema de notificaciones para mantener a los residentes informados, y un panel de control para la administración que permita visualizar reportes y gestionar las solicitudes. La tecnología incluye un framework como INTELLIJ para desarrollar el BACKEND usando el lenguaje de programación JAVA, se usará de base de datos PostgreSQL y de FRONTEND usaremos el framework REACT para componetizar bien nuestra solución, los servicios que vamos a usar para publicar nuestra aplicación son RENDER.COM para subir el backend y la base de datos y para nuestro frontend usaremos NETLIFY.COM por temas de mejor ruta miento entre las páginas.

Para los servicios usaremos las versiones gratuitas para hacer pruebas y presentar la aplicación.

**7. Los miedos:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Riesgos | Descripción | Probabilidad de ocurrencia | Impacto | Plan de mitigación |
| Falta de experiencia en ciertas tecnologías | La falta de conocimiento y experiencia en ciertos procesos que son necesarios para el desarrollo de las herramientas para la aplicación que podría dificultar el desarrollo de esta. | Alto | Medio | Implementación de un plan de estudios individual o grupal para explorar las tecnologías necesarias para el desarrollo adecuado de las funcionalidades |
| Retrasos en la entrega | La carga académica junto al trabajo particular de los integrantes del equipo de trabajo puede demorar los momentos de trabajo del equipo al momento de desarrollar. | Bajo | Alto | Manejo de 4 contextos los cuales son dependientes ir entregando lo que en el tiempo sea posible ya que no medimos con claridad la capacidad de cada integrante. |
| Problemas de seguridad: | Vulnerabilidades en el sistema que podrían comprometer la información de los usuarios. | Alto | Alto | Implementación de un plan de estudios enfocado principalmente en la CiberSeguridad, el cual reforzará la seguridad y evitará posibles ataques externos. |

**8. Tamaño:**

El proyecto tiene un tamaño mediano, con un tiempo estimado de ejecución de 4 a 6 meses, incluyendo etapas de planificación, desarrollo, pruebas y ajustes. Dado que somos cuatro estudiantes, cada uno podría tomar un rol específico para dividir las tareas y trabajar en paralelo, lo que podría optimizar el tiempo de desarrollo.

[Eficiencia actual] ▰▰▰▰▰▱▱▱▱ 60%

[Objetivo] ▰▰▰▰▰▰▰▱▱ 80% (+20% con mejor coordinación)

El tamaño mediano permite equilibrar ambición y que sea rentable para el equipo estudiantil. Con roles definidos, comunicación constante y herramientas ágiles, el proyecto es ejecutable sin sobrecargar a los integrantes

[TIEMPO] 4-6 meses ≈ 120-180 días

[RIESGO] 30% (Principal: Dependencias entre módulos)

[BUFFER] 15% tiempo para imprevistos (≈3 semanas)

**9. El TRADE-OFF (Prioridades):**

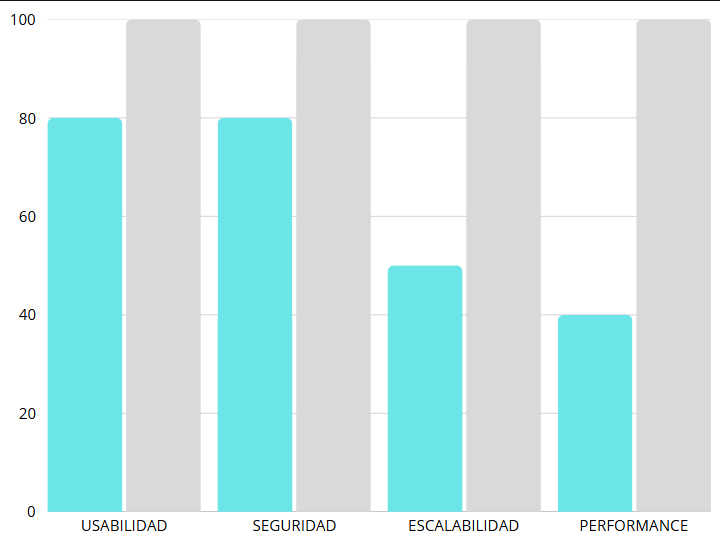
**Atributos de calidad priorizados:**

**Usabilidad:** Alta prioridad para asegurar que la aplicación sea intuitiva y fácil de usar para todos los usuarios.

**Seguridad:** Alta prioridad para proteger los datos de los residentes y la integridad del sistema.

**Escalabilidad:** Media prioridad para permitir el crecimiento del sistema con nuevas funcionalidades en el futuro.

**Performance:** Media prioridad, ya que la velocidad de respuesta debe ser adecuada, pero no es crítico si es ligeramente más lento debido a la carga.



[PRIORIDADES]

USABILIDAD: ████████░░ 80% (Alta)

SEGURIDAD: ████████░░ 80% (Alta)

ESCALABILIDAD: █████░░░░ 50% (Media)

PERFORMANCE: ████░░░░░ 40% (Media)

**10. ¿Cuánto cuesta?**

**Costos principales:**

**Desarrollo:** Principalmente el tiempo y esfuerzo de los cuatro estudiantes, que podría estimarse como costo de oportunidad.

**Hosting y Dominio:** Gastos asociados con el servidor web y el nombre de dominio, pero solo usaremos las versiones gratuidad de RENDER Y NETLIFY para hacer pruebas al inicio.

**Software y Licencias:** Dependiendo de las herramientas utilizadas (por ejemplo, software de diseño o IDEs de pago), podría haber costos adicionales, pero usaremos las versiones gratuitas por el momento.

**Testing y Deploy:** Potencialmente se pueden incluir algunos gastos en servicios de testing y despliegue, aunque podrían ser mínimos o evitados si se utilizan servicios gratuitos o de código abierto.

**VICTUS RESIDENCIAS requisitos**

1. **Requisitos Funcionales (Lo que el sistema debe hacer)**

Usaremos como identificador único UUID para cada entidad y poderla diferenciar en la base de datos.

**Gestión de Conjuntos Residenciales**

1. El sistema debe permitir crear, editar y eliminar conjuntos residenciales.
2. Cada conjunto debe tener un nombre único y un número de teléfono único.

**Gestión de Zonas Comunes**

1. El sistema debe permitir agregar zonas comunes a un conjunto residencial.
2. Cada zona común debe estar asociada a un conjunto existente.
3. Debe ser posible editar y eliminar zonas comunes de un conjunto.

**Gestión de Viviendas**

1. El sistema debe permitir registrar viviendas dentro de un conjunto (casa y apartamento).
2. Debe ser posible editar y eliminar viviendas dentro de un conjunto.

**Gestión de Residentes**

1. El sistema debe permitir registrar residentes con sus datos personales.
2. Un residente debe estar asociado a una vivienda específica.
3. Un residente no puede estar asociado a más de una vivienda al mismo tiempo.
4. Debe permitir la edición y eliminación de residentes.

**Gestión de la Agenda de Reservas**

1. El administrador debe poder definir los turnos y horarios permitidos en cada zona común.
2. No deben existir horarios sobrepuestos dentro de una misma zona común.
3. El sistema debe mostrar los turnos disponibles para cada zona común.

**Sistema de Reservas**

1. Los residentes deben poder reservar turnos en las zonas comunes disponibles.
2. Una vez realizada una reserva, el turno debe bloquearse para que no pueda ser reservado nuevamente.
3. Un residente solo puede tener una reserva activa en una misma zona común a la vez.
4. El sistema debe permitir la cancelación de reservas por parte del residente o del administrador.
5. Se debe notificar a los residentes cuando su reserva sea confirmada o cancelada.
6. El sistema debe permitir la consulta del historial de reservas.

**2. Requisitos No Funcionales (Cómo debe funcionar el sistema)**

**Rendimiento y Escalabilidad**

1. El sistema debe responder en menos de 3 segundos para consultas y reservas.
2. Debe ser escalable para admitir múltiples conjuntos residenciales sin afectar el rendimiento.

**Usabilidad**

1. La interfaz debe ser intuitiva y fácil de usar para administradores y residentes.
2. Debe estar optimizada para dispositivos móviles y de escritorio.

**Seguridad**

1. El acceso al sistema debe ser mediante autenticación con usuario y contraseña.
2. Los administradores y residentes deben tener diferentes niveles de permisos.
3. Los datos personales de los residentes deben ser protegidos según normativas de privacidad.

**Disponibilidad y Fiabilidad**

1. El sistema debe estar disponible al menos el 99% del tiempo.
2. Debe contar con copias de seguridad automáticas de la base de datos.

**3. Requisitos de Información (Datos que manejará el sistema todos heredan de una clase la cual les proporcionara el UUID como identificador unico)**

**Información de País**

* Nombre.

**Información de Departamento**

* Nombre.
* Pais (foreign key).

**Información de Ciudad**

* Nombre.
* Departamento (foreign key)

**Información de Conjuntos Residenciales**

* Nombre del conjunto residencial (único).
* Teléfono de contacto (único).
* Dirección
* Descripción
* Ciudad
* Correo
* Administrador (foreign key)
* Portero (foreing Key)

**Información de Administrador**

* Nombre.
* Apellido
* Tipo de documento
* Numero de documento
* Numero contacto
* Email
* Contraseña

**Información de Portero**

* Nombre.
* Apellido
* Tipo de documento
* Numero de documento
* Numero contacto
* Email
* Contraseña

**Información de Zonas Comunes**

* Nombre de la zona común (piscina, gimnasio, etc.).
* Descripción de la zona.
* Conjunto residencial al que pertenece.
* Imagen
* Capacidad
* Tiempo de uso por día
* Normas
* Conjunto residencial(foreign key)

**Información de Inmuebles**

* Tipo de vivienda (casa, apartamento).
* Numero de vivienda.
* Tipo de zona de vivienda (Torres, Manzana).
* Numero de zona de vivienda.
* Conjunto residencial (foreign key).

**Información de Residentes**

* Nombre completo.
* Tipo de identificación.
* Número de identificación.
* Fecha de nacimiento.
* Contacto (correo, teléfono).
* Inmueble (foreign key)
* Contraseña.
* Estado (activo, no activo)

**Información de Reservas**

* Zona común (foreign key).
* Residente que hizo la reserva (foreign key).
* Agenda (foreign key).
* Estado de la reserva (activa, cancelada, completada).
* Numero de invitados.

**Información de Agenda de Reservas**

* Turno (foreign key).
* Zona comun (foreign key)
* Tipo de recurrencia.
* Fecha y hora de inicio o fin.

**Información de turnos**

* nombre.
* Numero de turno
* Hora de inicio
* Hora fin
* estado

**PBIS DE Victus Residencias**

**Módulo: Gestión de Administradores**

**PBI 1: Gestión de Administradores**

**Descripción: Este PBI comprende la funcionalidad para crear, buscar, modificar y eliminar administradores del sistema. Los administradores son los usuarios con permisos especiales para gestionar conjuntos residenciales, zonas comunes, residents y todas las funcionalidades administrativas de la plataforma. Esta gestión incluye el registro de información personal y de contacto, así como la validación de datos únicos como el correo electrónico.**

**HU1: Registrar Administrador**

**Descripción: Como administrador del sistema, necesito poder registrar nuevos administradores para que puedan gestionar los conjuntos residenciales y sus funcionalidades asociadas.**

**Criterios de aceptación:**

1. **El sistema debe permitir registrar un nuevo administrador con la siguiente información:** 
   * 1. **Nombre (obligatorio, máximo 50 caracteres)**
     2. **Apellido (obligatorio, máximo 50 caracteres)**
     3. **Tipo de documento (obligatorio: Cédula de Ciudadanía, Cédula de Extranjería, Pasaporte)**
     4. **Número de documento (obligatorio, único en el sistema)**
     5. **Número de contacto (obligatorio, formato válido de teléfono)**
     6. **Email (obligatorio, formato válido, único en el sistema)**
     7. **Contraseña (obligatoria, cumplir políticas de seguridad)**
2. **El sistema debe validar que:** 
   * 1. **Todos los campos obligatorios estén completos**
     2. **El email tenga un formato válido**
     3. **El número de documento sea único en el sistema**
     4. **El email sea único en el sistema**
     5. **La contraseña cumpla con las políticas de seguridad (mínimo 8 caracteres, al menos una mayúscula, un número y un carácter especial)**
3. **Al registrar exitosamente, el sistema debe:** 
   * 1. **Generar un ID único para el administrador**
     2. **Encriptar la contraseña antes de almacenarla**
     3. **Mostrar un mensaje de confirmación**
     4. **Limpiar el formulario para permitir un nuevo registro**
4. **El sistema debe proporcionar retroalimentación inmediata sobre errores de validación.**
5. **Debe existir un botón para cancelar el registro y limpiar el formulario.**
6. **El administrador recién registrado debe quedar activo por defecto.**
7. **El sistema debe permitir asignar permisos específicos al momento del registro (opcional).**

**HU2: Buscar por nombre Administrador**

**Descripción: Como administrador del sistema, necesito poder buscar administradores existentes por su nombre para encontrar rápidamente la información que necesito gestionar.**

**Criterios de aceptación:**

1. **El sistema debe proporcionar un campo de búsqueda visible en la interfaz de gestión de administradores.**
2. **La búsqueda debe permitir buscar por:** 
   * 1. **Nombre completo o parcial**
     2. **Apellido completo o parcial**
     3. **Combinación de nombre y apellido**
3. **La búsqueda debe ser:** 
   * 1. **Insensible a mayúsculas y minúsculas**
     2. **Tolerante a espacios adicionales**
     3. **Con resultados en tiempo real (mientras se escribe)**
4. **Los resultados deben mostrarse en una tabla o lista que incluya:** 
   * 1. **Nombre completo**
     2. **Email**
     3. **Número de contacto**
     4. **Acciones disponibles (editar, eliminar)**
5. **Si no se encuentran resultados, debe mostrar un mensaje apropiado: "No se encontraron administradores con ese nombre".**
6. **Al limpiar el campo de búsqueda, debe mostrar todos los administradores registrados.**
7. **La búsqueda debe poder combinarse con otros filtros si existen (estado, fecha de registro, etc.).**
8. **Los resultados deben paginarse si son más de un número específico (ej. 20 administradores por página).**

**HU3: Editar Administrador**

**Descripción: Como administrador del sistema, necesito poder modificar la información de administradores existentes para mantener los datos actualizados y corregir posibles errores.**

**Criterios de aceptación:**

1. **El sistema debe permitir acceder a la función de edición desde:** 
   * 1. **Un botón "Editar" en cada fila de la tabla de administradores**
     2. **La vista detallada de un administrador específico**
2. **Al seleccionar editar, el sistema debe:** 
   * 1. **Cargar todos los datos actuales del administrador en el formulario**
     2. **Mantener el ID del administrador (no editable)**
     3. **Permitir modificar todos los campos editables**
3. **Campos editables:** 
   * 1. **Nombre**
     2. **Apellido**
     3. **Tipo de documento**
     4. **Número de documento**
     5. **Número de contacto**
     6. **Email**
     7. **Contraseña (opcional, solo si se desea cambiar)**
4. **Las mismas validaciones del registro deben aplicarse en la edición:** 
   * 1. **Formato válido de email**
     2. **Unicidad del email (excepto para el mismo administrador)**
     3. **Unicidad del número de documento (excepto para el mismo administrador)**
     4. **Políticas de contraseña (si se cambia)**
5. **Al guardar cambios exitosamente:** 
   * 1. **Mostrar mensaje de confirmación**
     2. **Actualizar la información en la lista/tabla**
     3. **Regresar a la vista principal de administradores**
6. **Debe existir opción para cancelar la edición y regresar sin guardar cambios.**
7. **Si otro administrador modifica el mismo registro simultáneamente, debe mostrar un mensaje de conflicto.**
8. **Los cambios críticos (email, contraseña) pueden requerir confirmación adicional.**

**HU4: Eliminar Administrador**

**Descripción: Como administrador del sistema, necesito poder eliminar administradores que ya no requieren acceso al sistema, manteniendo la integridad de los datos relacionados.**

**Criterios de aceptación:**

1. **El sistema debe proporcionar una opción para eliminar en:** 
   * 1. **Un botón "Eliminar" en cada fila de la tabla de administradores**
     2. **La vista detallada de un administrador específico**
2. **Al seleccionar eliminar, el sistema debe:** 
   * 1. **Mostrar un modal de confirmación**
     2. **Indicar claramente qué administrador se va a eliminar (nombre completo y email)**
     3. **Requerir confirmación explícita**
3. **El modal de confirmación debe incluir:** 
   * 1. **Mensaje claro: "¿Está seguro de que desea eliminar al administrador [Nombre Apellido]?"**
     2. **Advertencia sobre las consecuencias**
     3. **Botones "Confirmar" y "Cancelar" claramente diferenciados**
4. **Antes de eliminar, el sistema debe verificar:** 
   * 1. **Si el administrador tiene conjuntos residenciales asignados**
     2. **Si es el único administrador del sistema (no permitir eliminación)**
     3. **Si tiene acciones pendientes o reservas gestionadas**
5. **Opciones de eliminación:** 
   * 1. **Eliminación física: remover completamente el registro**
     2. **Eliminación lógica: marcar como inactivo pero conservar los datos**
6. **Al eliminar exitosamente:** 
   * 1. **Mostrar mensaje de confirmación**
     2. **Actualizar la lista de administradores**
     3. **Registrar la acción en un log de auditoría**
7. **Si la eliminación no es posible, mostrar mensaje explicativo con las razones.**
8. **No debe ser posible que un administrador se elimine a sí mismo del sistema.**

**PBI 2: Interfaz Gestión Administrador**

**Descripción: Este PBI se centra en el desarrollo de interfaces de usuario intuitivas y funcionales para la gestión de administradores. Incluye todas las pantallas necesarias para registrar, editar y eliminar administradores de forma eficiente, garantizando una experiencia de usuario óptima y consistente con el diseño general de la aplicación.**

**HU5: Interfaz Registrar Administrador**

**Descripción: Como administrador del sistema, necesito una interfaz clara y funcional para registrar nuevos administradores en el sistema de manera eficiente.**

**Criterios de aceptación:**

1. **La interfaz debe incluir un formulario con los siguientes campos organizados de manera lógica:** 
   * 1. **Nombre (campo de texto)**
     2. **Apellido (campo de texto)**
     3. **Tipo de documento (lista desplegable con opciones: Cédula de Ciudadanía, Cédula de Extranjería, Pasaporte)**
     4. **Número de documento (campo numérico)**
     5. **Número de contacto (campo de texto con formato de teléfono)**
     6. **Email (campo de email)**
     7. **Contraseña (campo de contraseña con opción para mostrar/ocultar)**
2. **Características de la interfaz:** 
   * 1. **Diseño responsive que funcione en escritorio y dispositivos móviles**
     2. **Campos claramente etiquetados**
     3. **Indicadores visuales para campos obligatorios (asterisco \*)**
     4. **Validación en tiempo real con mensajes de error específicos**
3. **Botones de acción:** 
   * 1. **"Crear" (de color verde/azul para indicar acción principal)**
     2. **"Cancelar" (para limpiar el formulario)**
     3. **"Regresar" (para volver a la lista de administradores)**
4. **Funcionalidades de la interfaz:** 
   * 1. **Auto-focus en el primer campo al cargar la página**
     2. **Navegación con tabulador entre campos**
     3. **Mensajes de error específicos para cada campo**
     4. **Indicador visual de campos con errores (borde rojo)**
5. **Al completar el registro exitosamente:** 
   * 1. **Mostrar mensaje de éxito con el nombre del administrador creado**
     2. **Opción para crear otro administrador o regresar a la lista**
     3. **Limpiar automáticamente todos los campos**
6. **La interfaz debe incluir ayuda contextual para campos que puedan generar dudas.**
7. **Debe mantener consistencia visual con el resto de la aplicación (colores, tipografía, espaciado).**

**HU6: Interfaz Editar Administrador**

**Descripción: Como administrador del sistema, necesito una interfaz intuitiva para modificar la información de administradores existentes, pre-cargando la información actual y permitiendo cambios selectivos.**

**Criterios de aceptación:**

1. **La interfaz de edición debe:** 
   * 1. **Utilizar el mismo layout del formulario de registro**
     2. **Pre-cargar todos los datos actuales del administrador**
     3. **Mantener el ID del administrador visible pero no editable**
2. **Características específicas de la edición:** 
   * 1. **Título del formulario debe indicar "Editar Administrador: [Nombre Apellido]"**
     2. **Campo de contraseña debe aparecer vacío con opción "Cambiar contraseña"**
     3. **Checkbox para "Cambiar contraseña" que habilite/deshabilite ese campo**
3. **Botones de acción:** 
   * 1. **"Guardar cambios" (color diferenciado para indicar que es una actualización)**
     2. **"Cancelar" (que retorne a la vista anterior sin guardar)**
     3. **"Eliminar" (opcional, con confirmación adicional)**
4. **Validaciones durante la edición:** 
   * 1. **Mismas validaciones que en el registro**
     2. **Permitir mantener el email actual (no marcar como duplicado)**
     3. **Permitir mantener el número de documento actual**
5. **Retroalimentación visual:** 
   * 1. **Campos modificados deben marcarse visualmente (por ejemplo, con un asterisco o color diferente)**
     2. **Loading indicator mientras se guardan los cambios**
     3. **Mensaje de confirmación al guardar exitosamente**
6. **Funcionalidades adicionales:** 
   * 1. **Botón para resetear campos a sus valores originales**
     2. **Navegación entre administradores con botones "Anterior" y "Siguiente"**
     3. **Historial de cambios (opcional, mostrar última fecha de modificación)**
7. **La interfaz debe manejar errores de validación mostrando mensajes específicos sin pérdida de datos ingresados.**

**HU7: Interfaz Eliminar Administrador**

**Descripción: Como administrador del sistema, necesito una interfaz clara y segura para eliminar administradores, con confirmaciones apropiadas y información sobre las consecuencias de la acción.**

**Criterios de aceptación:**

1. **Acceso a la función de eliminación:** 
   * 1. **Botón "Eliminar" disponible en cada fila de la tabla de administradores**
     2. **Botón "Eliminar" en la vista de edición del administrador**
     3. **Ícono de eliminación claramente identificable (basurero o X)**
2. **Modal de confirmación debe incluir:** 
   * 1. **Título claro: "Eliminar Administrador"**
     2. **Información del administrador a eliminar: nombre completo y email**
     3. **Mensaje de advertencia sobre la irreversibilidad de la acción**
     4. **Lista de elementos que podrían verse afectados (conjuntos gestionados, etc.)**
3. **Diseño del modal:** 
   * 1. **Fondo semi-transparente que oscurezca el resto de la pantalla**
     2. **Centrado en la pantalla**
     3. **Botones claramente diferenciados:** 
        1. **"Confirmar eliminación" (color rojo, indicando peligro)**
        2. **"Cancelar" (color neutro)**
4. **Comportamiento del modal:** 
   * 1. **Debe cerrarse al hacer clic fuera de él (cancelar operación)**
     2. **Debe cerrarse con la tecla Escape**
     3. **Focus automático en el botón "Cancelar" para evitar eliminaciones accidentales**
5. **Después de confirmar la eliminación:** 
   * 1. **Mostrar loading indicator durante el proceso**
     2. **Mensaje de confirmación exitosa**
     3. **Actualizar automáticamente la lista de administradores**
     4. **Redirigir a la lista principal si se estaba editando el administrador eliminado**
6. **Casos especiales:** 
   * 1. **Si no se puede eliminar (último administrador, restricciones de integridad), mostrar mensaje explicativo**
     2. **Si hay procesos pendientes, listar específicamente qué debe resolverse primero**
7. **Funcionalidad adicional:** 
   * 1. **Opción para en lugar de eliminar, "desactivar" el administrador**
     2. **Registro en log de auditoría indicando quién eliminó qué administrador y cuándo**

**PBI 3: Restricción evitar Administradores con el mismo correo**

**Descripción: Este PBI se enfoca en garantizar la unicidad del correo electrónico en el sistema de administradores, implementando validaciones que impidan registrar o actualizar administradores con emails duplicados. Esta restricción es fundamental para la seguridad y funcionamiento correcto del sistema de autenticación.**

**HU8: Mensaje de error de que no se puede registrar un admin con el mismo correo Administrador**

**Descripción: Como administrador del sistema, necesito recibir un mensaje de error claro y específico cuando intente registrar o editar un administrador con un correo electrónico que ya está en uso por otro administrador.**

**Criterios de aceptación:**

1. **Momento de la validación:** 
   * 1. **Al salir del campo email (validación en tiempo real)**
     2. **Al intentar enviar el formulario de registro**
     3. **Al intentar guardar cambios en la edición**
2. **El mensaje de error debe:** 
   * 1. **Ser específico: "Este correo electrónico ya está registrado por otro administrador"**
     2. **Aparecer inmediatamente debajo del campo de email**
     3. **Usar color rojo para indicar error**
     4. **Incluir un ícono de error para mayor claridad visual**
3. **Comportamiento del formulario ante el error:** 
   * 1. **El botón "Crear" o "Guardar" debe estar deshabilitado mientras persista el error**
     2. **El campo email debe resaltarse con borde rojo**
     3. **El error debe desaparecer automáticamente al corregir el email**
4. **Validación durante la edición:** 
   * 1. **Debe permitir mantener el email actual del administrador (no generar error por su propio email)**
     2. **Solo generar error si el email ya pertenece a otro administrador diferente**
     3. **Al cambiar a un email que ya existe, mostrar inmediatamente el error**
5. **Sugerencias adicionales en el mensaje:** 
   * 1. **Opcional: "¿Desea recuperar la cuenta existente?" con enlace a recuperación**
     2. **Opcional: "Verifique si ya tiene una cuenta con este email"**
6. **Validación en backend:** 
   * 1. **Validar también en el servidor antes de guardar**
     2. **Si por alguna razón la validación frontend falla, mostrar error del servidor**
     3. **Manejar errores de conexión o timeout apropiadamente**
7. **Casos especiales:** 
   * 1. **Validar format del email antes de verificar duplicados**
     2. **Considerar emails en diferentes casos (mayúsculas/minúsculas) como iguales**
     3. **Eliminar espacios en blanco antes y después del email antes de validar**
8. **Retroalimentación positiva:** 
   * 1. **Cuando se ingrese un email válido y disponible, mostrar un checkmark verde**
     2. **Mensaje de confirmación discreto: "Email disponible"**

**Módulo: Gestión de Conjuntos Residenciales**

**PBI 1: Gestionar entidad conjunto residencial**

**Descripción:** Este PBI abarca la funcionalidad CRUD (Crear, Leer, Actualizar, Eliminar) para la entidad de conjunto residencial. Permite al administrador del sistema crear nuevos conjuntos residenciales en la base de datos, modificar sus datos cuando sea necesario y eliminarlos cuando ya no sean requeridos. Cada conjunto residencial debe ser identificado con un UUID único.

**HU1: Crear conjunto residencial**

**Descripción:** Como administrador del sistema, necesito poder crear un nuevo conjunto residencial en la plataforma para incluirlo en la gestión de VICTUS RESIDENCIAS.

**Criterios de aceptación:**

1. El sistema debe generar un UUID único para cada nuevo conjunto residencial.
2. El formulario de creación debe incluir los siguientes campos obligatorios:
   * 1. Nombre del conjunto residencial (único)
     2. Teléfono de contacto (único)
     3. Dirección
     4. Descripción
     5. Administrador asignado
3. El sistema debe validar que no exista otro conjunto con el mismo nombre o teléfono.
4. El sistema debe confirmar la creación exitosa mediante un mensaje apropiado.
5. El nuevo conjunto debe ser almacenado en la base de datos.
6. El sistema debe responder en menos de 3 segundos al procesar la creación.

**HU2: Editar conjunto residencial**

**Descripción:** Como administrador del sistema, necesito poder editar la información de un conjunto residencial existente para mantener los datos actualizados.

**Criterios de aceptación:**

1. El sistema debe mostrar un formulario con los datos actuales del conjunto seleccionado.
2. Se deben poder modificar todos los campos:
   * 1. Nombre del conjunto residencial
     2. Teléfono de contacto
     3. Dirección
     4. Descripción
     5. Administrador asignado
3. El sistema debe validar que al editar no se duplique el nombre o teléfono con otro conjunto existente.
4. El sistema debe confirmar la actualización exitosa mediante un mensaje apropiado.
5. Los cambios deben reflejarse inmediatamente en la base de datos.
6. El sistema debe mantener la integridad referencial con las entidades asociadas (zonas comunes, viviendas).

**HU3: Eliminar conjunto residencial**

**Descripción:** Como administrador del sistema, necesito poder eliminar un conjunto residencial cuando ya no esté en funcionamiento o haya sido registrado por error.

**Criterios de aceptación:**

1. El sistema debe mostrar una confirmación antes de eliminar el conjunto.
2. El sistema debe validar que no existan entidades dependientes (viviendas, zonas comunes, residentes) antes de permitir la eliminación.
3. Si existen entidades dependientes, el sistema debe notificar al usuario y ofrecer opciones:
   * 1. Cancelar la eliminación
     2. Proceder con una eliminación en cascada (con advertencia clara)
4. El sistema debe confirmar la eliminación exitosa mediante un mensaje apropiado.
5. El conjunto eliminado no debe aparecer más en los listados de conjuntos activos.
6. Se debe mantener un registro de la eliminación para propósitos de auditoría.

**PBI 2: Interfaz Gestionar conjunto residencial**

**Descripción:** Este PBI se centra en el desarrollo de interfaces de usuario intuitivas y responsivas para la gestión de conjuntos residenciales. Incluye todas las pantallas necesarias para crear, editar y eliminar conjuntos de forma eficiente, garantizando una experiencia de usuario óptima tanto en dispositivos móviles como de escritorio.

**HU4: Interfaz crear acceso para el registro de conjunto residencial**

**Descripción:** Como administrador del sistema, necesito una interfaz clara y funcional para registrar nuevos conjuntos residenciales en la plataforma.

**Criterios de aceptación:**

1. La interfaz debe ser accesible desde el menú principal en la sección "Gestión de Conjuntos".
2. El formulario debe presentar todos los campos requeridos con etiquetas descriptivas.
3. Los campos obligatorios deben estar claramente marcados.
4. La interfaz debe incluir validación en tiempo real para:
   * 1. Formato válido de teléfono
     2. Campo de nombre no vacío
     3. Dirección no vacía
5. Debe incluir botones claramente identificables para "Guardar" y "Cancelar".
6. La interfaz debe ser responsiva y funcionar correctamente en dispositivos móviles y de escritorio.
7. Debe incluir ayuda contextual o tooltips para campos que requieran explicación adicional.

**HU5: Interfaz editar conjunto residencial**

**Descripción:** Como administrador del sistema, necesito una interfaz que me permita modificar de forma sencilla la información de un conjunto residencial existente.

**Criterios de aceptación:**

1. La interfaz debe ser accesible desde:
   * 1. El listado de conjuntos residenciales
     2. La vista detallada de un conjunto específico
2. El formulario debe mostrar los datos actuales del conjunto seleccionado.
3. Todos los campos deben ser editables.
4. La interfaz debe incluir validación en tiempo real similar a la de creación.
5. Debe incluir botones para "Guardar cambios", "Cancelar" y "Volver al listado".
6. Los cambios en campos clave (como nombre o teléfono) deben solicitar confirmación.
7. La interfaz debe ser responsiva y mantener la consistencia visual con el resto de la aplicación.

**HU6: Interfaz eliminar conjunto residencial**

**Descripción:** Como administrador del sistema, necesito una interfaz clara y segura para eliminar conjuntos residenciales, previniendo eliminaciones accidentales.

**Criterios de aceptación:**

1. Debe haber un botón o enlace para eliminar en:
   * 1. El listado de conjuntos residenciales
     2. La vista detallada del conjunto
2. Al seleccionar la opción de eliminar, debe aparecer un modal de confirmación.
3. El modal debe:
   * 1. Advertir claramente sobre las consecuencias de la eliminación
     2. Mostrar el nombre del conjunto a eliminar para confirmar la intención
     3. Incluir botones "Confirmar" y "Cancelar" claramente diferenciados
4. Si el conjunto tiene entidades dependientes, debe mostrarse una lista de estas entidades.
5. La interfaz debe proporcionar feedback visual claro tras la eliminación.
6. Después de la eliminación, redirigir al listado de conjuntos actualizado.

**PBI 3: Validación evitar conjunto residencial nombre y teléfono iguales**

**Descripción:** Este PBI se enfoca en garantizar la integridad y unicidad de los datos de conjuntos residenciales en la base de datos. Implementa validaciones tanto en el backend como en la interfaz de usuario para prevenir duplicaciones de nombre o número de teléfono en los conjuntos residenciales, asegurando que cada conjunto sea únicamente identificable.

**HU7: Mensaje error o advertencia (interfaz)**

**Descripción:** Como administrador del sistema, necesito recibir notificaciones claras cuando intente crear o editar un conjunto residencial con un nombre o teléfono que ya existe en el sistema.

**Criterios de aceptación:**

1. Al introducir un nombre de conjunto ya existente, debe aparecer un mensaje de error inmediato (validación en tiempo real).
2. Al introducir un teléfono ya registrado, debe aparecer un mensaje de error inmediato.
3. Los mensajes de error deben ser específicos:
   * 1. "Ya existe un conjunto residencial con este nombre. Por favor, ingrese un nombre diferente."
     2. "Este número de teléfono ya está registrado para otro conjunto. Por favor, verifique e ingrese un número diferente."
4. Los campos con error deben destacarse visualmente (por ejemplo, con borde rojo).
5. El botón de guardar debe deshabilitarse mientras existan errores de validación.
6. Al intentar enviar el formulario con errores, debe mostrarse un resumen de los problemas a corregir.
7. La validación debe funcionar tanto en la creación como en la edición de conjuntos.
8. Después de corregir el error, el mensaje debe desaparecer automáticamente y el campo debe volver a su estado normal.

**Módulo: Gestión de Zonas Comunes**

**PBI 4: Gestionar entidad Zona común**

**Descripción:** Este PBI comprende la funcionalidad CRUD (Crear, Leer, Actualizar, Eliminar) para la entidad de zona común dentro de los conjuntos residenciales. Permite al administrador del sistema registrar nuevas zonas comunes para un conjunto específico, modificar su información cuando sea necesario y eliminarlas cuando corresponda. Cada zona común debe ser identificada con un UUID único y estar asociada a un conjunto residencial existente.

**HU8: Registrar Zona común**

**Descripción:** Como administrador del sistema, necesito poder registrar una nueva zona común asociada a un conjunto residencial específico para permitir su gestión y posterior reserva por parte de los residentes.

**Criterios de aceptación:**

1. El sistema debe generar un UUID único para cada nueva zona común.
2. El formulario de registro debe incluir los siguientes campos:
   * 1. Nombre de la zona común (piscina, gimnasio, etc.)
     2. Descripción detallada de la zona
     3. Conjunto residencial al que pertenece (selección obligatoria)
     4. Imagen representativa (opcional)
     5. Capacidad máxima de personas
     6. Tiempo máximo de uso por día
     7. Normas de uso
3. Solo se deben mostrar conjuntos residenciales existentes en el sistema.
4. El sistema debe validar que todos los campos obligatorios estén completos.
5. La zona común debe quedar correctamente vinculada al conjunto residencial seleccionado.
6. El sistema debe confirmar el registro exitoso mediante un mensaje apropiado.
7. La nueva zona común debe ser almacenada en la base de datos.

**HU9: Editar Zona común**

**Descripción:** Como administrador del sistema, necesito poder editar la información de una zona común existente para mantener los datos actualizados o corregir información incorrecta.

**Criterios de aceptación:**

1. El sistema debe mostrar un formulario con los datos actuales de la zona común seleccionada.
2. Se deben poder modificar todos los campos:
   * 1. Nombre de la zona común
     2. Descripción
     3. Imagen representativa
     4. Capacidad
     5. Tiempo de uso
     6. Normas
3. No debe ser posible cambiar el conjunto residencial al que pertenece la zona común.
4. El sistema debe validar que todos los campos obligatorios mantengan información válida.
5. El sistema debe confirmar la actualización exitosa mediante un mensaje apropiado.
6. Los cambios deben reflejarse inmediatamente en la base de datos.
7. El sistema debe mantener la integridad referencial con las entidades asociadas (reservas).

**HU10: Eliminar Zona común**

**Descripción:** Como administrador del sistema, necesito poder eliminar una zona común cuando ya no esté disponible en el conjunto residencial o haya sido registrada por error.

**Criterios de aceptación:**

1. El sistema debe mostrar una confirmación antes de eliminar la zona común.
2. El sistema debe validar que no existan reservas activas o futuras para esa zona común.
3. Si existen reservas activas o futuras, el sistema debe:
   * 1. Notificar al administrador sobre esta situación
     2. Ofrecer opciones para cancelar las reservas existentes o abortar la eliminación
4. El sistema debe confirmar la eliminación exitosa mediante un mensaje apropiado.
5. La zona común eliminada no debe aparecer más en los listados de zonas disponibles.
6. Se debe mantener un registro de la eliminación para propósitos de auditoría.
7. Los residentes con reservas canceladas deben ser notificados automáticamente.

**PBI 5: Interfaz Gestionar Zona común**

**Descripción:** Este PBI se centra en el desarrollo de interfaces de usuario intuitivas y funcionales para la gestión de zonas comunes en los conjuntos residenciales. Incluye todas las pantallas necesarias para registrar, editar y eliminar zonas comunes de forma eficiente, garantizando una experiencia de usuario óptima tanto en dispositivos móviles como de escritorio.

**HU11: Interfaz registrar zona común**

**Descripción:** Como administrador del sistema, necesito una interfaz clara y funcional para registrar nuevas zonas comunes dentro de un conjunto residencial.

**Criterios de aceptación:**

1. La interfaz debe ser accesible desde:
   * 1. El menú principal en la sección "Gestión de Zonas Comunes"
     2. La vista detallada del conjunto residencial al que pertenecerá
2. El formulario debe presentar todos los campos requeridos con etiquetas descriptivas.
3. Los campos obligatorios deben estar claramente marcados.
4. Debe incluir un selector para elegir el conjunto residencial asociado.
5. Debe permitir la carga de imágenes para la zona común con vista previa.
6. Debe incluir un editor para las normas de uso que permita formato básico (negritas, viñetas).
7. La interfaz debe incluir validación en tiempo real para los campos obligatorios.
8. Debe incluir botones claramente identificables para "Guardar" y "Cancelar".
9. La interfaz debe ser responsiva y funcionar correctamente en dispositivos móviles y de escritorio.

**HU12: Interfaz editar zona común**

**Descripción:** Como administrador del sistema, necesito una interfaz que me permita modificar de forma sencilla la información de una zona común existente.

**Criterios de aceptación:**

1. La interfaz debe ser accesible desde:
   * 1. El listado de zonas comunes
     2. La vista detallada de una zona común específica
     3. La vista de zonas comunes dentro del conjunto residencial
2. El formulario debe mostrar los datos actuales de la zona común seleccionada.
3. Todos los campos deben ser editables excepto el conjunto residencial asociado.
4. La imagen actual debe mostrarse y debe existir la opción de mantenerla o cambiarla.
5. La interfaz debe incluir validación en tiempo real para los campos obligatorios.
6. Debe incluir botones para "Guardar cambios", "Cancelar" y "Volver al listado".
7. La interfaz debe ser responsiva y mantener la consistencia visual con el resto de la aplicación.
8. Debe mostrar la fecha de la última modificación si aplica.

**HU13: Interfaz eliminar zona común**

**Descripción:** Como administrador del sistema, necesito una interfaz clara y segura para eliminar zonas comunes, previniendo eliminaciones accidentales y gestionando adecuadamente las reservas existentes.

**Criterios de aceptación:**

1. Debe haber un botón o enlace para eliminar en:
   * 1. El listado de zonas comunes
     2. La vista detallada de la zona común
2. Al seleccionar la opción de eliminar, debe aparecer un modal de confirmación.
3. El modal debe:
   * 1. Advertir claramente sobre las consecuencias de la eliminación
     2. Mostrar el nombre de la zona común a eliminar para confirmar la intención
     3. Incluir botones "Confirmar" y "Cancelar" claramente diferenciados
4. Si la zona común tiene reservas activas o futuras, el modal debe:
   * 1. Mostrar un listado de las reservas afectadas
     2. Ofrecer opciones para gestionar estas reservas (cancelar todas, notificar a usuarios)
5. La interfaz debe proporcionar feedback visual claro tras la eliminación.
6. Después de la eliminación, redirigir al listado de zonas comunes del conjunto residencial.

**PBI 6: Restricción Asociar Zona común a Conjunto residencial Existentes**

**Descripción:** Este PBI se enfoca en garantizar la integridad y unicidad de los datos de zonas comunes en la base de datos. Implementa validaciones tanto en el backend como en la interfaz de usuario para prevenir duplicaciones de zonas comunes dentro de un mismo conjunto residencial y asegurar que solo se puedan asociar a conjuntos residenciales existentes en el sistema.

**HU14: Mensaje error zona común ya existente en un conjunto residencial**

**Descripción:** Como administrador del sistema, necesito recibir notificaciones claras cuando intente registrar una zona común con un nombre que ya existe dentro del mismo conjunto residencial.

**Criterios de aceptación:**

1. Al introducir un nombre de zona común que ya existe en el conjunto residencial seleccionado, debe aparecer un mensaje de error inmediato (validación en tiempo real).
2. El mensaje de error debe ser específico:
   * 1. "Ya existe una zona común con este nombre en el conjunto residencial seleccionado. Por favor, ingrese un nombre diferente."
3. El campo con error debe destacarse visualmente (por ejemplo, con borde rojo).
4. El botón de guardar debe deshabilitarse mientras exista este error de validación.
5. Al intentar enviar el formulario con el error, debe mostrarse nuevamente el mensaje indicando el problema a corregir.
6. La validación debe funcionar tanto en el registro como en la edición de zonas comunes.
7. Después de corregir el error, el mensaje debe desaparecer automáticamente y el campo debe volver a su estado normal.
8. El sistema debe permitir que existan zonas comunes con el mismo nombre en diferentes conjuntos residenciales.
9. La selección del conjunto residencial solo debe mostrar conjuntos existentes y activos en el sistema.

**Módulo: Gestión de Viviendas**

**PBI 7: Gestionar entidades vivienda**

**Descripción:** Este PBI abarca la funcionalidad CRUD (Crear, Leer, Actualizar, Eliminar) para la entidad de vivienda dentro de los conjuntos residenciales. Permite al administrador del sistema registrar nuevas viviendas para un conjunto específico, modificar su información cuando sea necesario y eliminarlas cuando corresponda. Cada vivienda debe ser identificada con un UUID único y estar asociada a un conjunto residencial existente, con información sobre su tipo (casa o apartamento) y su ubicación dentro del conjunto.

**HU15: Registrar Vivienda**

**Descripción:** Como administrador del sistema, necesito poder registrar una nueva vivienda asociada a un conjunto residencial específico para posteriormente asignarle residentes.

**Criterios de aceptación:**

1. El sistema debe generar un UUID único para cada nueva vivienda.
2. El formulario de registro debe incluir los siguientes campos obligatorios:
   * 1. Tipo de vivienda (casa, apartamento)
     2. Número de vivienda
     3. Tipo de zona de vivienda (Torres, Manzana)
     4. Número de zona de vivienda
     5. Conjunto residencial al que pertenece (selección)
3. Solo se deben mostrar conjuntos residenciales existentes para la selección.
4. El sistema debe validar que la combinación de número de vivienda, tipo y número de zona sea única dentro del conjunto seleccionado.
5. El sistema debe confirmar el registro exitoso mediante un mensaje apropiado.
6. La nueva vivienda debe ser almacenada en la base de datos.
7. La vivienda debe quedar correctamente vinculada al conjunto residencial seleccionado.

**HU16: Editar Vivienda**

**Descripción:** Como administrador del sistema, necesito poder editar la información de una vivienda existente para mantener los datos actualizados o corregir información incorrecta.

**Criterios de aceptación:**

1. El sistema debe mostrar un formulario con los datos actuales de la vivienda seleccionada.
2. Se deben poder modificar los siguientes campos:
   * 1. Tipo de vivienda (casa, apartamento)
     2. Número de vivienda
     3. Tipo de zona de vivienda (Torres, Manzana)
     4. Número de zona de vivienda
3. No debe ser posible cambiar el conjunto residencial al que pertenece la vivienda.
4. El sistema debe validar que la nueva combinación de datos no duplique una vivienda existente en el mismo conjunto.
5. El sistema debe confirmar la actualización exitosa mediante un mensaje apropiado.
6. Los cambios deben reflejarse inmediatamente en la base de datos.
7. El sistema debe mantener la integridad referencial con las entidades asociadas (residentes).

**HU17: Eliminar Vivienda**

**Descripción:** Como administrador del sistema, necesito poder eliminar una vivienda cuando ya no esté disponible en el conjunto residencial o haya sido registrada por error.

**Criterios de aceptación:**

1. El sistema debe mostrar una confirmación antes de eliminar la vivienda.
2. El sistema debe validar que no existan residentes asignados a esa vivienda.
3. Si existen residentes asignados, el sistema debe:
   * 1. Notificar al administrador sobre esta situación
     2. Ofrecer opciones para desasociar los residentes o abortar la eliminación
4. El sistema debe confirmar la eliminación exitosa mediante un mensaje apropiado.
5. La vivienda eliminada no debe aparecer más en los listados de viviendas del conjunto.
6. Se debe mantener un registro de la eliminación para propósitos de auditoría.
7. Los residentes desasociados (si se eligió esta opción) deben quedar en estado "sin vivienda asignada".

**PBI 8: Interfaz Gestionar Vivienda**

**Descripción:** Este PBI se centra en el desarrollo de interfaces de usuario intuitivas y funcionales para la gestión de viviendas en los conjuntos residenciales. Incluye todas las pantallas necesarias para registrar, editar y eliminar viviendas de forma eficiente, garantizando una experiencia de usuario óptima tanto en dispositivos móviles como de escritorio.

**HU18: Interfaz registrar Vivienda**

**Descripción:** Como administrador del sistema, necesito una interfaz clara y funcional para registrar nuevas viviendas dentro de un conjunto residencial.

**Criterios de aceptación:**

1. La interfaz debe ser accesible desde:
   * 1. El menú principal en la sección "Gestión de Viviendas"
     2. La vista detallada del conjunto residencial correspondiente
2. El formulario debe presentar todos los campos requeridos con etiquetas descriptivas.
3. Los campos obligatorios deben estar claramente marcados.
4. Para el tipo de vivienda debe presentarse un selector con opciones predefinidas (casa, apartamento).
5. Para el tipo de zona de vivienda debe presentarse un selector con opciones predefinidas (Torres, Manzana).
6. Debe incluir un selector para elegir el conjunto residencial asociado.
7. La interfaz debe incluir validación en tiempo real para los campos obligatorios y formatos correctos.
8. Debe incluir botones claramente identificables para "Guardar" y "Cancelar".
9. La interfaz debe ser responsiva y funcionar correctamente en dispositivos móviles y de escritorio.
10. Debe ofrecer la posibilidad de registrar múltiples viviendas en secuencia sin volver al menú principal.

**HU19: Interfaz editar Vivienda**

**Descripción:** Como administrador del sistema, necesito una interfaz que me permita modificar de forma sencilla la información de una vivienda existente.

**Criterios de aceptación:**

1. La interfaz debe ser accesible desde:
   * 1. El listado de viviendas
     2. La vista detallada de una vivienda específica
     3. La vista de viviendas dentro del conjunto residencial
2. El formulario debe mostrar los datos actuales de la vivienda seleccionada.
3. Todos los campos deben ser editables excepto el conjunto residencial asociado.
4. La interfaz debe incluir validación en tiempo real para los campos obligatorios.
5. Debe mostrar información sobre los residentes actualmente asignados a la vivienda (solo lectura).
6. Debe incluir botones para "Guardar cambios", "Cancelar" y "Volver al listado".
7. La interfaz debe ser responsiva y mantener la consistencia visual con el resto de la aplicación.
8. Debe mostrar la fecha de la última modificación si aplica.

**HU20: Interfaz eliminar Vivienda**

**Descripción:** Como administrador del sistema, necesito una interfaz clara y segura para eliminar viviendas, previniendo eliminaciones accidentales y gestionando adecuadamente los residentes asignados.

**Criterios de aceptación:**

1. Debe haber un botón o enlace para eliminar en:
   * 1. El listado de viviendas
     2. La vista detallada de la vivienda
2. Al seleccionar la opción de eliminar, debe aparecer un modal de confirmación.
3. El modal debe:
   * 1. Advertir claramente sobre las consecuencias de la eliminación
     2. Mostrar los datos identificativos de la vivienda a eliminar para confirmar la intención
     3. Incluir botones "Confirmar" y "Cancelar" claramente diferenciados
4. Si la vivienda tiene residentes asignados, el modal debe:
   * 1. Mostrar un listado de los residentes afectados
     2. Ofrecer opciones para gestionar estos residentes (desasociarlos o cancelar la eliminación)
5. La interfaz debe proporcionar feedback visual claro tras la eliminación.
6. Después de la eliminación, redirigir al listado de viviendas del conjunto residencial.

**PBI 9: Validación evitar Vivienda duplicadas en el conjunto residencial**

**Descripción:** Este PBI se enfoca en garantizar la integridad y unicidad de los datos de viviendas en la base de datos. Implementa validaciones tanto en el backend como en la interfaz de usuario para prevenir duplicaciones de viviendas dentro de un mismo conjunto residencial, considerando la combinación única de número de vivienda, tipo y número de zona.

**HU21: Mensaje error Vivienda ya existente en un conjunto residencial**

**Descripción:** Como administrador del sistema, necesito recibir notificaciones claras cuando intente registrar una vivienda con una combinación de identificadores que ya existe dentro del mismo conjunto residencial.

**Criterios de aceptación:**

1. Al introducir una combinación de número de vivienda, tipo de zona y número de zona que ya existe en el conjunto residencial seleccionado, debe aparecer un mensaje de error inmediato (validación en tiempo real).
2. El mensaje de error debe ser específico:
   * 1. "Ya existe una vivienda con este número en la misma zona dentro del conjunto residencial seleccionado. Por favor, verifique la información."
3. Los campos con error deben destacarse visualmente (por ejemplo, con borde rojo).
4. El botón de guardar debe deshabilitarse mientras exista este error de validación.
5. Al intentar enviar el formulario con el error, debe mostrarse nuevamente el mensaje indicando el problema a corregir.
6. La validación debe funcionar tanto en el registro como en la edición de viviendas.
7. Después de corregir el error, el mensaje debe desaparecer automáticamente y los campos deben volver a su estado normal.
8. El sistema debe permitir que existan viviendas con la misma numeración en diferentes conjuntos residenciales o en diferentes zonas del mismo conjunto.
9. La validación debe considerar la combinación completa (número de vivienda + tipo de zona + número de zona) como clave única dentro del conjunto.

**Módulo: Gestión de Residentes**

**PBI 10: Gestionar entidades Residentes**

**Descripción:** Este PBI comprende la funcionalidad CRUD (Crear, Leer, Actualizar, Eliminar) para la entidad de residentes dentro del sistema VICTUS RESIDENCIAS. Permite registrar nuevos residentes con sus datos personales, asignarlos a viviendas específicas, actualizar su información personal y eliminarlos del sistema cuando sea necesario. El PBI garantiza que cada residente esté correctamente identificado y que no pueda estar asignado a más de una vivienda simultáneamente.

**HU22: Registrar Residente**

**Descripción:** Como administrador del sistema, necesito poder registrar un nuevo residente en la plataforma con sus datos personales completos para incluirlo en la gestión de VICTUS RESIDENCIAS.

**Criterios de aceptación:**

1. El sistema debe generar un UUID único para cada nuevo residente.
2. El formulario de registro debe incluir los siguientes campos obligatorios:
   * 1. Nombre completo
     2. Tipo de identificación (cédula, pasaporte, etc.)
     3. Número de identificación
     4. Fecha de nacimiento
     5. Contacto (correo electrónico y teléfono)
     6. Contraseña inicial para acceso al sistema
3. El sistema debe validar el formato correcto de:
   * 1. Correo electrónico (formato válido)
     2. Número de teléfono (formato numérico válido)
     3. Número de identificación (según el tipo seleccionado)
4. El sistema debe verificar que no exista otro residente con el mismo número de identificación.
5. La fecha de nacimiento debe corresponder a una persona mayor de edad (18 años o más).
6. El sistema debe confirmar el registro exitoso mediante un mensaje apropiado.
7. El residente puede registrarse inicialmente sin asociación a una vivienda específica.

**HU23: Asignar vivienda a un residente**

**Descripción:** Como administrador del sistema, necesito poder asignar una vivienda específica a un residente previamente registrado para establecer su lugar de residencia dentro del conjunto.

**Criterios de aceptación:**

1. El sistema debe permitir seleccionar un residente existente y una vivienda disponible dentro de un conjunto residencial.
2. Solo se deben mostrar viviendas que pertenezcan a conjuntos residenciales activos.
3. El sistema debe verificar que el residente no esté ya asignado a otra vivienda.
4. El sistema debe confirmar la asignación exitosa mediante un mensaje apropiado.
5. La asignación debe reflejarse inmediatamente en la base de datos.
6. El sistema debe actualizar el estado del residente a "con vivienda asignada".
7. El sistema debe permitir visualizar todos los residentes asignados a una misma vivienda.
8. Debe ser posible cambiar a un residente de una vivienda a otra (desasignación y nueva asignación).

**HU24: Editar información de un residente**

**Descripción:** Como administrador del sistema, necesito poder editar la información personal de un residente existente para mantener los datos actualizados o corregir información incorrecta.

**Criterios de aceptación:**

1. El sistema debe mostrar un formulario con los datos actuales del residente seleccionado.
2. Se deben poder modificar los siguientes campos:
   * 1. Nombre completo
     2. Tipo y número de identificación
     3. Fecha de nacimiento
     4. Datos de contacto (correo, teléfono)
3. El sistema debe validar el formato correcto de todos los campos modificados.
4. Si se modifica el número de identificación, debe verificarse que no exista duplicidad con otro residente.
5. No debe ser posible modificar directamente la vivienda asignada desde esta interfaz (debe usarse la funcionalidad específica).
6. El sistema debe confirmar la actualización exitosa mediante un mensaje apropiado.
7. Los cambios deben reflejarse inmediatamente en la base de datos.
8. Debe existir la opción de restablecer la contraseña del residente si es necesario.

**HU25: Eliminar un residente**

**Descripción:** Como administrador del sistema, necesito poder eliminar un residente cuando ya no viva en el conjunto residencial o haya sido registrado por error.

**Criterios de aceptación:**

1. El sistema debe mostrar una confirmación antes de eliminar al residente.
2. El sistema debe verificar si el residente tiene reservas activas en zonas comunes.
3. Si existen reservas activas, el sistema debe:
   * 1. Notificar al administrador sobre esta situación
     2. Ofrecer opciones para cancelar las reservas o abortar la eliminación
4. El sistema debe desasociar automáticamente al residente de su vivienda asignada.
5. El sistema debe confirmar la eliminación exitosa mediante un mensaje apropiado.
6. El residente eliminado no debe aparecer más en los listados de residentes activos.
7. Se debe mantener un registro de la eliminación para propósitos de auditoría.
8. Las credenciales de acceso del residente deben ser invalidadas inmediatamente.

**PBI 11: Interfaz Gestionar Residentes**

**Descripción:** Este PBI se centra en el desarrollo de interfaces de usuario intuitivas y funcionales para la gestión de residentes. Incluye todas las pantallas necesarias para registrar, editar y eliminar residentes de forma eficiente, garantizando una experiencia de usuario óptima tanto en dispositivos móviles como de escritorio.

**HU26: Interfaz registrar Residentes**

**Descripción:** Como administrador del sistema, necesito una interfaz clara y funcional para registrar nuevos residentes en la plataforma.

**Criterios de aceptación:**

1. La interfaz debe ser accesible desde:
   * 1. El menú principal en la sección "Gestión de Residentes"
     2. La vista detallada de una vivienda (para registro directo con asignación)
2. El formulario debe presentar todos los campos requeridos con etiquetas descriptivas.
3. Los campos obligatorios deben estar claramente marcados.
4. Para el tipo de identificación debe presentarse un selector con opciones predefinidas.
5. La fecha de nacimiento debe incluir un selector de calendario.
6. La interfaz debe incluir validación en tiempo real para:
   * 1. Formato válido de correo electrónico
     2. Formato válido de número telefónico
     3. Edad mínima (18 años)
     4. Formato correcto del número de identificación según el tipo seleccionado
7. Debe incluir una opción para asignar vivienda inmediatamente o dejar pendiente esta asignación.
8. Debe incluir botones claramente identificables para "Guardar" y "Cancelar".
9. La interfaz debe ser responsiva y funcionar correctamente en dispositivos móviles y de escritorio.
10. Debe ofrecer la posibilidad de registrar múltiples residentes en secuencia sin volver al menú principal.

**HU27: Interfaz editar Residentes**

**Descripción:** Como administrador del sistema, necesito una interfaz que me permita modificar de forma sencilla la información personal de un residente existente.

**Criterios de aceptación:**

1. La interfaz debe ser accesible desde:
   * 1. El listado de residentes
     2. La vista detallada de un residente específico
     3. La vista de residentes asociados a una vivienda
2. El formulario debe mostrar los datos actuales del residente seleccionado.
3. Todos los campos personales deben ser editables.
4. Debe mostrar información sobre la vivienda actualmente asignada (solo lectura).
5. Debe incluir un botón para cambiar la asignación de vivienda que redirija a la funcionalidad específica.
6. La interfaz debe incluir validación en tiempo real para los campos obligatorios.
7. Debe incluir una sección para gestionar las credenciales de acceso:
   * 1. Opción para restablecer contraseña
     2. Opción para activar/desactivar acceso
8. Debe incluir botones para "Guardar cambios", "Cancelar" y "Volver al listado".
9. La interfaz debe ser responsiva y mantener la consistencia visual con el resto de la aplicación.

**HU28: Interfaz eliminar Residentes**

**Descripción:** Como administrador del sistema, necesito una interfaz clara y segura para eliminar residentes, previniendo eliminaciones accidentales y gestionando adecuadamente sus asociaciones y reservas.

**Criterios de aceptación:**

1. Debe haber un botón o enlace para eliminar en:
   * 1. El listado de residentes
     2. La vista detallada del residente
2. Al seleccionar la opción de eliminar, debe aparecer un modal de confirmación.
3. El modal debe:
   * 1. Advertir claramente sobre las consecuencias de la eliminación
     2. Mostrar el nombre completo e identificación del residente a eliminar
     3. Incluir botones "Confirmar" y "Cancelar" claramente diferenciados
4. Si el residente tiene reservas activas en zonas comunes, el modal debe:
   * 1. Mostrar un listado de las reservas afectadas
     2. Ofrecer opciones para gestionar estas reservas (cancelar todas o mantenerlas)
5. La interfaz debe mostrar información sobre la vivienda que quedará desasociada.
6. La interfaz debe proporcionar feedback visual claro tras la eliminación.
7. Después de la eliminación, redirigir al listado general de residentes.

**PBI 12: Restricción evitar residentes que pertenezcan a más de una vivienda**

**Descripción:** Este PBI se enfoca en garantizar la integridad de los datos y el cumplimiento de la regla de negocio que establece que un residente no puede estar asignado a más de una vivienda simultáneamente. Implementa validaciones tanto en el backend como en la interfaz de usuario para prevenir asignaciones múltiples.

**HU29: Mensaje error este residente ya fue asignado a una vivienda**

**Descripción:** Como administrador del sistema, necesito recibir notificaciones claras cuando intente asignar un residente a una vivienda y este ya se encuentre asignado a otra vivienda en cualquier conjunto residencial.

**Criterios de aceptación:**

1. Al intentar asignar un residente que ya tiene una vivienda asignada, debe aparecer un mensaje de error inmediato.
2. El mensaje de error debe ser específico:
   * 1. "Este residente ya está asignado a la vivienda [número] en [zona] del conjunto [nombre del conjunto]. Un residente no puede pertenecer a más de una vivienda simultáneamente."
3. El mensaje debe incluir información sobre la vivienda actualmente asignada al residente.
4. El sistema debe ofrecer las siguientes opciones:
   * 1. Cancelar la nueva asignación
     2. Transferir al residente a la nueva vivienda (desasignándolo automáticamente de la anterior)
5. Si se elige la opción de transferencia, el sistema debe solicitar confirmación adicional.
6. El sistema debe registrar el historial de cambios de vivienda para cada residente.
7. La validación debe aplicarse tanto durante el registro con asignación directa como durante la asignación posterior.
8. La interfaz debe proporcionar un indicador visual que muestre si un residente ya tiene una vivienda asignada cuando se está buscando residentes para asignar.

**Módulo: Gestión de la Agenda de Reservas**

**PBI 13: Gestionar entidades agenda reservas**

**Descripción:** Este PBI comprende la funcionalidad para configurar, gestionar y visualizar la agenda de reservas para las zonas comunes de los conjuntos residenciales. Permite al administrador definir los turnos y horarios disponibles para cada zona común, evitar sobreposiciones de horarios, y garantizar una gestión eficiente de la disponibilidad de estos espacios. La agenda de reservas es la base sobre la cual los residentes podrán posteriormente realizar sus reservas.

**HU30: Configurar turnos y horarios de reservas**

**Descripción:** Como administrador del sistema, necesito poder configurar los turnos y horarios disponibles para reservas en cada zona común, definiendo franjas horarias específicas según las necesidades del conjunto residencial.

**Criterios de aceptación:**

1. El sistema debe permitir seleccionar una zona común específica para configurar su agenda.
2. Debe ser posible definir múltiples franjas horarias para cada día de la semana.
3. Para cada franja horaria, se deben configurar:
   * 1. Hora de inicio
     2. Hora de finalización
     3. Días de la semana aplicables
     4. Capacidad máxima (si difiere de la capacidad general de la zona)
4. El sistema debe permitir establecer la duración mínima y máxima de una reserva.
5. Debe ser posible configurar el tiempo mínimo de antelación para realizar una reserva.
6. El sistema debe permitir establecer el máximo de reservas que un residente puede tener activas simultáneamente.
7. Debe ser posible configurar restricciones adicionales como:
   * 1. Límite de invitados por reserva
     2. Periodo de tiempo entre reservas del mismo residente
8. Los cambios en la configuración no deben afectar a las reservas ya existentes.
9. El sistema debe confirmar la configuración exitosa mediante un mensaje apropiado.

**HU31: Evitar solapamiento de horarios**

**Descripción:** Como administrador del sistema, necesito que la plataforma evite automáticamente el solapamiento de horarios al configurar la agenda de reservas para una zona común, garantizando la integridad de los turnos disponibles.

**Criterios de aceptación:**

1. El sistema debe validar en tiempo real que no exista solapamiento entre las franjas horarias que se están configurando para un mismo día.
2. Al intentar crear una franja horaria que se solapa con otra existente, el sistema debe:
   * 1. Mostrar un mensaje de error específico
     2. Indicar claramente cuál es la franja horaria con la que se produce el conflicto
     3. No permitir guardar la configuración hasta resolver el conflicto
3. El sistema debe considerar solapamiento cuando:
   * 1. La hora de inicio de una franja está dentro del rango de otra franja
     2. La hora de finalización de una franja está dentro del rango de otra franja
     3. Una franja contiene completamente a otra franja
4. Debe existir la opción de definir un tiempo mínimo entre franjas horarias consecutivas (por ejemplo, para limpieza o preparación).
5. La validación debe aplicarse tanto en la creación como en la edición de franjas horarias.
6. El sistema debe proporcionar una visualización clara que permita identificar fácilmente los horarios ya configurados.

**HU32: Visualizar horarios disponibles**

**Descripción:** Como administrador o residente del sistema, necesito poder visualizar claramente los horarios disponibles para reserva en cada zona común, identificando fácilmente los turnos libres y ocupados.

**Criterios de aceptación:**

1. El sistema debe proporcionar una vista de calendario que muestre todos los horarios configurados.
2. La visualización debe permitir filtrar por:
   * 1. Zona común específica
     2. Conjunto residencial
     3. Rango de fechas
     4. Disponibilidad (ocupado/disponible)
3. Los horarios deben distinguirse visualmente según su estado:
   * 1. Disponible: color verde o similar
     2. Reservado: color rojo o similar
     3. Pendiente de confirmación: color amarillo o similar
4. Al hacer clic en un horario específico, debe mostrar información detallada:
   * 1. Para horarios disponibles: capacidad, restricciones
     2. Para horarios reservados: nombre del residente (solo visible para administradores), número de invitados
5. Debe ser posible cambiar la visualización entre vista diaria, semanal y mensual.
6. La interfaz debe actualizarse automáticamente o tener un botón de actualización para reflejar los cambios en tiempo real.
7. La visualización debe ser responsiva y adaptarse a diferentes dispositivos.
8. Debe incluir indicadores claros cuando un horario no esté disponible por mantenimiento o causas especiales.

**HU33: Crear agenda**

**Descripción:** Como administrador del sistema, necesito poder crear una agenda completa para una zona común específica, estableciendo patrones recurrentes de disponibilidad que faciliten la configuración de horarios a largo plazo.

**Criterios de aceptación:**

1. El sistema debe permitir seleccionar una zona común para crear su agenda.
2. Debe ser posible establecer un patrón de recurrencia:
   * 1. Diaria (todos los días)
     2. Semanal (días específicos de la semana)
     3. Mensual (días específicos del mes)
3. Para cada patrón recurrente, se deben definir:
   * 1. Hora de inicio
     2. Hora de finalización
     3. Frecuencia de repetición
     4. Fecha de inicio y de finalización del patrón (opcional)
4. El sistema debe permitir establecer excepciones a los patrones:
   * 1. Días festivos o fechas específicas sin disponibilidad
     2. Horarios especiales para fechas concretas
5. El sistema debe generar automáticamente todos los horarios según la configuración establecida.
6. Debe ser posible previsualizar los horarios generados antes de confirmar la creación.
7. El sistema debe alertar sobre posibles conflictos con configuraciones existentes.
8. Debe permitir crear agendas independientes para diferentes temporadas (alta, baja, vacaciones, etc.).
9. El sistema debe confirmar la creación exitosa de la agenda mediante un mensaje apropiado.

**PBI 14: Interfaz Gestionar Agenda**

**Descripción:** Este PBI se centra en el desarrollo de interfaces de usuario intuitivas y funcionales para la gestión de la agenda de reservas. Incluye todas las pantallas necesarias para registrar, editar y eliminar configuraciones de agenda de forma eficiente, garantizando una experiencia de usuario óptima tanto para administradores como para residentes.

**HU34: Interfaz registrar Agenda**

**Descripción:** Como administrador del sistema, necesito una interfaz clara y funcional para crear y configurar la agenda de reservas para las zonas comunes.

**Criterios de aceptación:**

1. La interfaz debe ser accesible desde:
   * 1. El menú principal en la sección "Gestión de Agendas"
     2. La vista detallada de una zona común específica
2. Debe incluir un selector para elegir la zona común a configurar.
3. La interfaz debe presentar un calendario visual para seleccionar fechas y horarios.
4. Debe ofrecer herramientas para definir patrones de recurrencia:
   * 1. Selectores de días de la semana
     2. Opciones para repetición diaria, semanal o mensual
     3. Campos para fecha de inicio y fin del patrón
5. Debe permitir configurar múltiples franjas horarias por día mediante:
   * 1. Selectores de hora tipo reloj o campos de entrada de hora
     2. Opción para añadir o eliminar franjas
6. Debe incluir una sección para configurar restricciones:
   * 1. Capacidad máxima por franja
     2. Límite de invitados
     3. Tiempo mínimo de antelación
7. Debe ofrecer una vista previa de los horarios generados antes de guardar.
8. Debe incluir botones claramente identificables para "Guardar", "Vista previa" y "Cancelar".
9. La interfaz debe ser responsiva y funcionar correctamente en dispositivos móviles y de escritorio.
10. Debe incluir ayuda contextual para explicar las diferentes opciones de configuración.

**HU35: Interfaz editar agenda**

**Descripción:** Como administrador del sistema, necesito una interfaz que me permita modificar configuraciones de agenda existentes para zonas comunes, ajustando horarios o restricciones según las necesidades cambiantes del conjunto residencial.

**Criterios de aceptación:**

1. La interfaz debe ser accesible desde:
   * 1. El listado de agendas configuradas
     2. La vista detallada de una zona común
     3. El calendario de disponibilidad
2. Debe cargar y mostrar la configuración actual de la agenda seleccionada.
3. Debe permitir modificar todos los aspectos de la configuración:
   * 1. Franjas horarias
     2. Patrones de recurrencia
     3. Restricciones y reglas
4. Debe proporcionar opciones para:
   * 1. Editar la configuración completa
     2. Modificar solo días o franjas específicas
     3. Añadir o eliminar excepciones
5. La interfaz debe marcar claramente qué franjas horarias ya tienen reservas asociadas.
6. Debe mostrar advertencias si los cambios afectarán a reservas existentes.
7. Debe ofrecer opciones para gestionar el impacto en reservas existentes:
   * 1. Mantener las reservas existentes como excepción
     2. Cancelar y notificar a los afectados
     3. Reprogramar automáticamente si es posible
8. Debe incluir una vista previa de cómo quedará el calendario tras los cambios.
9. La interfaz debe ser responsiva y mantener la consistencia visual con el resto de la aplicación.

**HU36: Interfaz eliminar agenda**

**Descripción:** Como administrador del sistema, necesito una interfaz clara y segura para eliminar configuraciones de agenda que ya no sean necesarias, gestionando adecuadamente las reservas existentes.

**Criterios de aceptación:**

1. Debe haber un botón o enlace para eliminar en:
   * 1. El listado de agendas configuradas
     2. La vista detallada de una configuración de agenda
2. Al seleccionar la opción de eliminar, debe aparecer un modal de confirmación.
3. El modal debe:
   * 1. Advertir claramente sobre las consecuencias de la eliminación
     2. Mostrar información sobre qué configuración se eliminará (zona común, periodo)
     3. Incluir botones "Confirmar" y "Cancelar" claramente diferenciados
4. Si existen reservas asociadas a la agenda a eliminar, el modal debe:
   * 1. Mostrar el número de reservas afectadas
     2. Ofrecer opciones para gestionar estas reservas (cancelar todas y notificar, mantener como excepciones)
5. Debe permitir eliminar:
   * 1. La agenda completa de una zona común
     2. Solo un patrón específico
     3. Solo excepciones o modificaciones específicas
     4. Desde una fecha en adelante
6. La interfaz debe proporcionar feedback visual claro tras la eliminación.
7. Después de la eliminación, redirigir al listado de agendas o al calendario de la zona común.

**PBI 15: Restricción evitar solapamiento de horarios**

**Descripción:** Este PBI se enfoca en garantizar la integridad de la agenda de reservas implementando validaciones que impidan el solapamiento de horarios dentro de una misma zona común. Estas validaciones se aplican tanto en el backend como en la interfaz de usuario para evitar conflictos en la disponibilidad de las zonas comunes.

**HU37: Mensaje error solapamiento horarios**

**Descripción:** Como administrador del sistema, necesito recibir notificaciones claras cuando intente configurar franjas horarias que se solapan dentro de la agenda de una zona común, para evitar conflictos en las reservas.

**Criterios de aceptación:**

1. Al intentar configurar una franja horaria que se solapa con otra existente, debe aparecer un mensaje de error inmediato (validación en tiempo real).
2. El mensaje de error debe ser específico:
   * 1. "La franja horaria propuesta (HH- HH) se solapa con una existente (HH

- HH) para el mismo día/patrón."

1. Los campos o selecciones con error deben destacarse visualmente (por ejemplo, con borde rojo).
2. El sistema debe mostrar gráficamente el solapamiento en el calendario visual.
3. El botón de guardar debe deshabilitarse mientras existan errores de solapamiento.
4. El sistema debe sugerir alternativas cuando sea posible:
   * 1. Horarios disponibles cercanos
     2. Ajustes menores que eliminarían el conflicto
5. La validación debe considerar:
   * 1. Solapamientos directos entre franjas
     2. Tiempos mínimos configurados entre franjas consecutivas
     3. Excepciones y patrones recurrentes
6. El sistema debe validar solapamientos incluso entre diferentes patrones de recurrencia que coincidan en fechas específicas.
7. Después de corregir el error, el mensaje debe desaparecer automáticamente y los elementos de la interfaz deben volver a su estado normal.

**Módulo: Sistema de Reservas**

**Módulo cuya intención es enfocarse en la gestión del proceso de las reservas; incluyendo la disponibilidad de los espacios y las reservas que los residentes efectúan**

**PBI 16: Sistema Reservas**

**Descripción**: Este PBI comprende la funcionalidad para que los residentes puedan realizar reservas de las zonas comunes del conjunto residencial según la disponibilidad configurada en la agenda. Permite a los usuarios consultar disponibilidad, realizar reservas, cancelarlas cuando sea necesario y recibir notificaciones sobre el estado de sus solicitudes. También incluye herramientas para que el administrador pueda bloquear turnos específicos por mantenimiento u otros motivos.

**HU38: Realizar una reserva**

**Descripción**: Como residente del conjunto, necesito poder realizar una reserva de una zona común en un horario disponible, especificando los detalles necesarios como número de invitados y propósito de la reserva.

**Criterios de aceptación**:

1. El sistema debe mostrar solo los horarios disponibles para reserva según la configuración de la agenda.
2. Para realizar una reserva, el residente debe proporcionar:
   * 1. Fecha y horario (dentro de las franjas disponibles)
     2. Número de invitados (sin exceder el límite configurado)
     3. Propósito de la reserva (evento, reunión, celebración, etc.)
     4. Información de contacto adicional (opcional)
3. El sistema debe validar que el residente no exceda:
   * 1. El número máximo de reservas activas permitidas
     2. El número máximo de reservas por periodo (semana/mes)
     3. El tiempo mínimo de antelación configurado
4. El sistema debe confirmar la disponibilidad en tiempo real antes de completar la reserva.
5. Una vez realizada la reserva, el sistema debe:
   * 1. Generar un código o identificador único para la reserva
     2. Mostrar un resumen con todos los detalles
     3. Actualizar inmediatamente la disponibilidad en el calendario
6. El residente debe recibir una confirmación por el canal preferido (email, notificación en app, etc.).
7. La reserva debe quedar en estado "confirmada" o "pendiente de aprobación" según la configuración del conjunto.
8. El sistema debe permitir adjuntar documentos si la reserva lo requiere (por ejemplo, lista de invitados).

**HU39: Cancelar Reserva**

**Descripción**: Como residente, necesito poder cancelar una reserva realizada previamente, liberando el horario para que pueda ser utilizado por otros residentes.

**Criterios de aceptación**:

1. El sistema debe permitir al residente visualizar todas sus reservas activas.
2. La opción de cancelación debe estar disponible para cada reserva no vencida.
3. Al seleccionar cancelar una reserva, el sistema debe:
   * 1. Solicitar confirmación mediante un diálogo
     2. Permitir ingresar un motivo de cancelación (opcional)
     3. Informar sobre posibles penalizaciones por cancelación tardía
4. El sistema debe validar la política de cancelación:
   * 1. Si la cancelación cumple con el tiempo mínimo establecido, se procesa sin penalización
     2. Si la cancelación es tardía, se debe informar y aplicar la penalización configurada
5. Tras la cancelación, el sistema debe:
   * 1. Actualizar inmediatamente la disponibilidad en el calendario
     2. Enviar una confirmación de cancelación al residente
     3. Notificar al administrador sobre la cancelación
6. Si existe lista de espera para ese horario, el sistema debe notificar automáticamente al siguiente residente en la lista.
7. El historial debe registrar la cancelación con fecha, hora y motivo.
8. Si la reserva requería pago y éste ya se realizó, el sistema debe indicar la política de reembolso aplicable.

**HU40: Bloquear Turno reservado**

**Descripción**: Como administrador del sistema, necesito poder bloquear temporalmente turnos o franjas horarias específicas para mantenimiento, eventos especiales o cualquier otra situación que requiera suspender la disponibilidad de una zona común.

**Criterios de aceptación**:

1. El administrador debe poder bloquear:
   * 1. Un turno específico
     2. Múltiples turnos consecutivos
     3. Una franja horaria completa
     4. Un día completo o varios días
2. Al bloquear un turno, se debe especificar:
   * 1. Motivo del bloqueo (mantenimiento, evento administrativo, etc.)
     2. Duración del bloqueo (fecha y hora de inicio y fin)
     3. Notas adicionales (opcional)
3. Si el turno a bloquear tiene reservas existentes:
   * 1. El sistema debe mostrar un listado de las reservas afectadas
     2. Ofrecer opciones para gestionar estas reservas: cancelar automáticamente, reubicar, o mantener
4. El sistema debe permitir configurar notificaciones automáticas para:
   * 1. Informar a los residentes afectados
     2. Personalizar el mensaje explicando el motivo del bloqueo
5. Los turnos bloqueados deben aparecer visualmente diferenciados en el calendario (por ejemplo, color gris o con un ícono específico).
6. El bloqueo debe poder ser temporal o permanente.
7. El administrador debe poder desbloquear un turno previamente bloqueado cuando sea necesario.
8. El sistema debe mantener un registro de todos los bloqueos realizados y sus motivos.

**HU41: Notificar confirmación reserva**

**Descripción**: Como residente o administrador, necesito recibir y enviar notificaciones sobre el estado de las reservas (confirmación, recordatorio, cancelación) a través de los canales configurados.

**Criterios de aceptación**:

1. El sistema debe enviar notificaciones automáticas en los siguientes eventos:
   * 1. Al confirmar una nueva reserva
     2. Al cancelar una reserva (por el residente o el administrador)
     3. Como recordatorio 24 horas antes del turno reservado
     4. Al modificar los detalles de una reserva
     5. Cuando un turno reservado es bloqueado por el administrador
2. Las notificaciones deben enviarse a través de los canales configurados:
   * 1. Correo electrónico
     2. Notificaciones push en la aplicación móvil
     3. SMS (opcional)
     4. Otro canal especificado por el residente
3. Cada notificación debe incluir:
   * 1. Identificador único de la reserva
     2. Datos completos (fecha, hora, zona común)
     3. Motivo de la notificación
     4. Acciones disponibles (cancelar, confirmar asistencia, etc.)
     5. Información de contacto del administrador
4. Los residentes deben poder configurar sus preferencias de notificación:
   * 1. Canales preferidos
     2. Tipos de notificaciones que desean recibir
     3. Tiempo de anticipación para recordatorios
5. El administrador debe poder:
   * 1. Enviar notificaciones personalizadas sobre una reserva específica
     2. Configurar plantillas para los diferentes tipos de notificaciones
     3. Ver el historial de notificaciones enviadas
6. El sistema debe confirmar el envío exitoso de las notificaciones.
7. Las notificaciones deben adaptarse al idioma configurado por el usuario.
8. En caso de fallos en el envío, el sistema debe reintentar y registrar el incidente.

**HU42: Consultar Historial Reservas**

**Descripción**: Como residente o administrador, necesito poder consultar el historial completo de reservas realizadas para analizar patrones de uso, verificar disponibilidad pasada o resolver disputas.

**Criterios de aceptación**:

1. Los residentes deben poder consultar:
   * 1. Todas sus reservas históricas
     2. Reservas activas (pendientes)
     3. Reservas canceladas
     4. Reservas completadas (pasadas)
2. Los administradores deben poder consultar:
   * 1. Todas las reservas del conjunto
     2. Reservas filtradas por zona común
     3. Reservas filtradas por residente
     4. Reservas filtradas por estado
3. El historial debe incluir para cada reserva:
   * 1. Datos completos de la reserva (fecha, hora, zona común, propósito)
     2. Estado final (confirmada, cancelada, completada)
     3. Cambios realizados (con fecha y usuario que los realizó)
     4. Notificaciones enviadas
     5. Incidencias reportadas
4. La interfaz debe permitir filtrar por:
   * 1. Rango de fechas
     2. Zona común específica
     3. Estado de la reserva
     4. Residente (solo para administradores)
5. Debe ser posible exportar el historial en formatos comunes (CSV, PDF).
6. El sistema debe permitir generar reportes estadísticos:
   * 1. Tasa de ocupación por zona
     2. Horarios más solicitados
     3. Frecuencia de cancelaciones
7. El historial debe mantener los registros por el tiempo configurado en la política de retención de datos.
8. Los residentes solo deben poder ver su propio historial de reservas.

**PBI 17: Interfaz Visualizar reservas**

**Descripción**: Este PBI se centra en el desarrollo de interfaces de usuario intuitivas para la visualización, gestión y realización de reservas de zonas comunes. Incluye las pantallas necesarias para que tanto residentes como administradores puedan interactuar con el sistema de reservas de manera eficiente y clara.

**HU43: Interfaz reserva**

**Descripción**: Como residente o administrador, necesito una interfaz clara e intuitiva para visualizar, realizar y gestionar reservas de las zonas comunes del conjunto residencial.

**Criterios de aceptación**:

1. La interfaz debe proporcionar vistas de calendario con las siguientes opciones:
   * 1. Vista diaria
     2. Vista semanal
     3. Vista mensual
     4. Vista por zona común
2. La visualización del calendario debe utilizar códigos de colores para distinguir:
   * 1. Horarios disponibles
     2. Horarios reservados por el usuario actual
     3. Horarios reservados por otros usuarios
     4. Horarios bloqueados por administración
     5. Horarios en lista de espera
3. Al hacer clic en un horario disponible, debe abrir un formulario de reserva con:
   * 1. Campos para los datos requeridos
     2. Resumen de las reglas aplicables
     3. Botones para confirmar o cancelar
4. Para las reservas existentes del usuario, al hacer clic debe mostrar:
   * 1. Detalles completos de la reserva
     2. Opción para cancelar
     3. Opción para modificar (si está permitido)
     4. Historial de cambios
5. Para los administradores, la interfaz debe incluir:
   * 1. Vista de todas las reservas
     2. Opción para filtrar por usuario, zona o estado
     3. Herramientas para bloquear horarios
     4. Acceso al panel de notificaciones
6. La interfaz debe ser responsiva y adaptarse a diferentes dispositivos.
7. Debe incluirse una barra de búsqueda para encontrar rápidamente horarios disponibles según criterios específicos.
8. La interfaz debe incluir indicadores visuales para:
   * 1. Tiempo mínimo de antelación
     2. Límites de reservas alcanzados
     3. Conflictos con otras reservas
9. Debe ofrecer opciones de accesibilidad para usuarios con diferentes capacidades.
10. La interfaz debe actualizarse en tiempo real o incluir un botón para refrescar datos.

Sprints para separar producción

**Fase 1: Base del sistema**

* PBIs: 1-3 (Conjuntos residenciales), 4-6 (Zonas comunes), 7-9 (Viviendas).

**Fase 2: Gestión de residentes y reservas iniciales**

* PBIs: 10-12 (Residentes), 13-15 (Agenda de reservas

**Fase 3: Reservas y mejoras finales**

* PBIs: 16-17 (Sistema de reservas).

Link Azure DevOps : <https://dev.azure.com/ViktusResidencias/Victus%20Residencias>