



Proyecto final - RealEstateApp

[2025] Documento Funcional

Versión 1.3

01 de Julio, 2025

© Itla 2025

Autopista Las Américas, Km. 27, PCSD,
La Caleta, Boca Chica 11606.

Tel. 809-738-4852

Índice

Objetivo general	3
Funcionalidades generales	3
Home	3
Agentes	5
Únete a la app	6
Iniciar sesión (Login)	6
Funcionalidades del cliente	7
Home del cliente	7
Detalles de propiedades(Cliente)	7
Funcionalidades del Agente	9
Home del agente	9
Detalle de propiedades(Agente)	9
Mantenimiento de propiedades	10
Mi perfil	12
Funcionalidades del Administrador	12
Home del administrador	12
Listado de los agentes	13
Mantenimiento de administradores	13
Mantenimiento de desarrolladores	14
Mantenimiento de tipo de propiedades	15
Mantenimiento de tipo de ventas	17
Mantenimiento de mejoras	18
Seguridad	19
Funcionalidades del Api	20
Funcionalidades generales	20
Login y Seguridad	20
Controlador de propiedades	21
Controlador de agentes	22
Mant. de tipo de propiedades	23
Mant. de tipo de ventas	25
Mant. de mejoras	26
Requerimientos técnicos	28

Objetivo general















- Crear una aplicación para el manejo de propiedades inmobiliarias, utilizando asp net mvc 8 o 9.

Funcionalidades generales

Home

Esta es la pantalla inicial del sistema, en esta pantalla se deben listar todas las propiedades creadas en el sistema(solo mostrar la que están en estado disponible), listadas desde la más reciente a la más antigua.

De las propiedades se debe mostrar la siguiente informaciones, Tipo de propiedad(apartamento,casa, etc.), una imagen de la propiedad, el código de la propiedad, el tipo de venta de esa propiedad (alquiler, venta , etc), el valor de la propiedad, la cantidad de habitaciones, la cantidad de baños y el tamaño de la propiedad en metros, ejemplo de esto sería esta imagen.

 <p>Proyecto Atlántica, Puerto Plata</p> <p>Código 161582</p> <p>Venta US\$ 285,000 - US\$ 545,000</p> <p> 166mt</p>	 <p>Apartamento Iberstate, Bavaro</p> <p>Código 161533</p> <p>Venta Amueblado US\$ 249,000</p> <p> 2  2  156mt</p>	 <p>Casa Sabana Perdida, Santo Domingo Norte</p> <p>Código 161531</p> <p>Venta RD\$ 5,000,000</p> <p> 5  3  241mt</p>	 <p>Apartamento Urbanización Thomen, Santiago</p> <p>Código 161499</p> <p>Alquiler Amueblado US\$ 1,200</p> <p> 3  2  107mt</p>
---	---	---	--

En esta pantalla tendremos un formulario que nos permite buscar por el código de la propiedad.

Y otro formulario para poder realizar los siguientes filtros:

- Filtrar por tipo de propiedad (Casa, apartamento, etc)
- Un filtro que permite buscar por un precio mínimo y un precio máximo
- Un filtro para colocar la cantidad de habitacion y de baños

Todos estos filtros se pueden aplicar al mismo tiempo.

Al hacer click sobre una propiedad debe enviarnos al detalle de la propiedad, aquí se muestra las siguientes informaciones: un slider con todas las imágenes de la propiedad, el código de la propiedad, el tipo de propiedad, el tipo de venta, el costo de la propiedad, cantidad de habitaciones, cantidad de baños, tamaño en metros, la descripción de la propiedad, el listado de mejoras que tiene la propiedad, el nombre del agente inmobiliario, el número de celular del agente, la foto del agente y el correo del agente, como por ejemplo en esta imagen:

Oops, model not available.
If the problem persists see our troubleshooting guide

Código: 161533

Apartamento

Venta Amueblado
US\$ 249,000

2 Habitación(es) 2 Baño(s) 156 m²

Descripción
¡Fantástica oportunidad de invertir en uno de los lugares turísticos más populares de República Dominicana! Cana Bay tiene todo lo que deseas en una casa de vacaciones. Club de playa privado con piscina infinita y restaurante, campo de golf diseñado por Jack Nicholson, centro de tenis y r quetbol, pronto se desarrollar  un spa y centro de bienestar y el Hard Rock Hotel para bailar o jugar por las noches. Este espectacular Penthouse de 2 pisos en el residencial Cana Rock est  listo para llevar tu maleta. 2 rec maras, 2 ba os, sala/comedor, balc n en la planta baja y gran terraza en la azotea con espectacular vista panor mica al campo de golf. El jacuzzi para 6 personas y la sala de estar cubierta son perfectos para relajarse y disfrutar despu s de un duro d a en el campo de golf o en la playa. Cana Rock tambi n ofrece a sus propietarios un  rea de juegos para ni os y una piscina de un cuarto de milla para descansar, alquiler de piscina, servicios de mucama y servicio de transporte a la playa y otras comodidades. Cana Bay est  a solo 25 minutos del Aeropuerto Internacional de Punta Cana y a 7-8 minutos del centro de Punta Cana para ir de compras, restaurantes, bancos y m s. Para obtener m s informaci n o programar una visita,  ntactame ahora! This is a wonderful opportunity to own your own penthouse just 4 minutes from one of the most beautiful beaches on the island.

 Est s interesado(a)?  Solicita Asesor a!

Marci Rodriguez

  Celular
(809) 829-1465

  WhatsApp
(809) 829-1465

  Email
marci@remaxip.com

Mejoras

  Balc n   Sala/Comedor   Cocina   Escalera   Ascensor

  Terraza Techada   Piscina   Parqueo   Jacuzzi   Lavado   Telecable

  A/C   Abanicos   Calentador   A/C Preinstalado   A/C Split

  Conserje   Acceso a Techo   Gimnasio

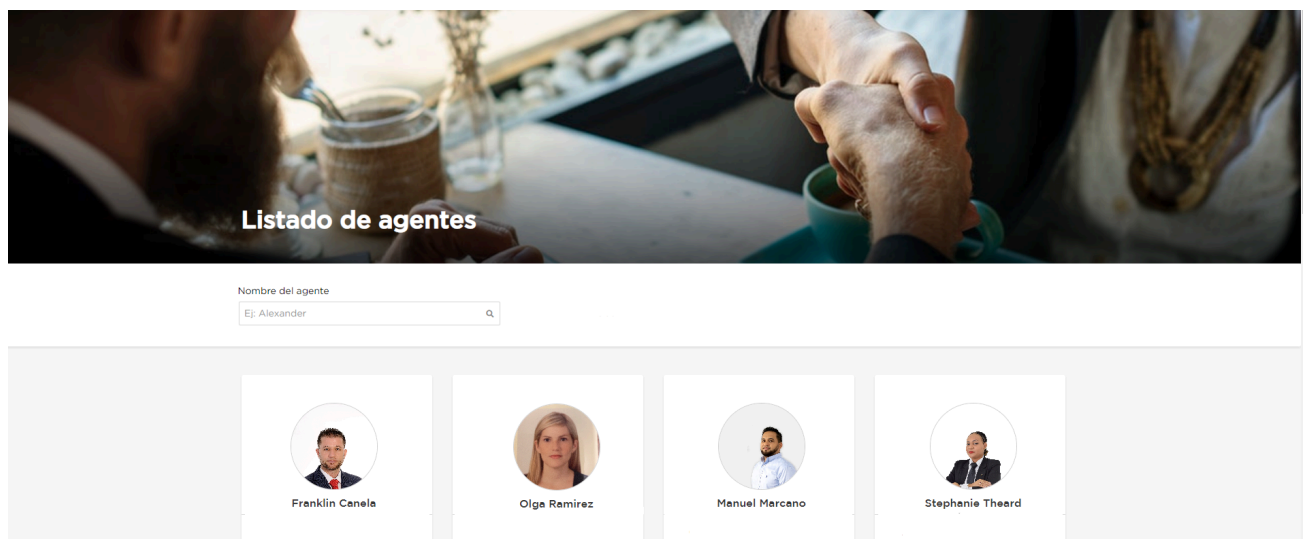
Finalmente en esta pantalla debe haber un menú con las siguientes opciones:

- Home
- Agentes
- Únete a la app
- Iniciar sesión

Agentes

Al ingresar a la opción de agentes desde el menú principal de la app, esta nos enviará a una pantalla con el listado de agentes registrados y activos en el sistema.

Aquí se listara las siguientes informaciones de los agentes, foto del agente, nombre del agente y hacer click sobre un agente nos debe llevar a una pantalla donde se listen todas las propiedades disponibles de ese agente, de las propiedades se deben mostrar las misma informaciones que en el home e igual al hacer click sobre esa propiedad del agente nos debe llevar al detalle de esa propiedad. Los agentes se debe mostrar organizado alfabéticamente y se debe poder buscar agente por nombre, un ejemplo de esta pantalla sería la siguiente imagen:



Únete a la app

Al ingresar a la opción de unete a la app desde el menú principal de la app, esta nos enviará a una pantalla con un formulario para registrarse en la app.

El formulario de esta pantalla tendrá los siguientes campos: nombre, apellido, teléfono, foto de usuario, nombre de usuario, correo , contraseña , confirmar contraseña, select con el tipo de usuario (Cliente o Agente).

Las validaciones de este formulario que todos los campos son requeridos.

Hay dos flujos diferentes a seguir si el usuario seleccionó la opción de cliente el usuario se crea en estado esta inactivo y se le debe enviar un correo al cliente para que active su usuario, pero si es un agente que se está registrado, el usuario igual se crear inactivo pero no se le envia un correo para que lo active ya que la activación de agente la debe realizar un administrador de la app.

En ambos casos una vez creado el usuario se debe redirigir a la pantalla de inicio de sesión de la app.

Iniciar sesión (Login)

Al ingresar a la opción de inicio de sesión desde el menú principal de la app, esta nos enviará a una pantalla con un formulario para poder hacer esta acción.

Aquí debe haber un formulario con los siguientes campos: correo o nombre de usuario y un campo para la contraseña y un botón para iniciar sesión.

Las validaciones de este formulario que todos los campos son requeridos.

Si los datos de acceso son incorrectos debe notificarle al usuario que los mismo están mal y de ser correcto pues el sistema aquí tiene tres posibilidades dependiendo el rol del usuario que inicio sesión.

El sistema tiene tres roles en la web app: cliente, agentes y administrador, entonces para describir que sucede en cada caso vamos a dividir el documento en tres secciones ahora, las secciones serán: Funcionalidades del cliente, funcionalidades del agente y funcionalidades del administrador.

Funcionalidades del cliente

Home del cliente

Si el usuario que inicio sesión es del rol cliente el sistema lo debe redireccionar a la pantalla de home de la app donde se listan todas las propiedades disponibles creadas en el sistema. Lo que sí debe variar en este listado de propiedades del home es que el usuario debe poder visualizar un botón en cada propiedad del listado con la que pueda marcar o desmarcar esa propiedad como favorita.

Aparte de ese cambio en listado de propiedades el menú del cliente será el siguiente:

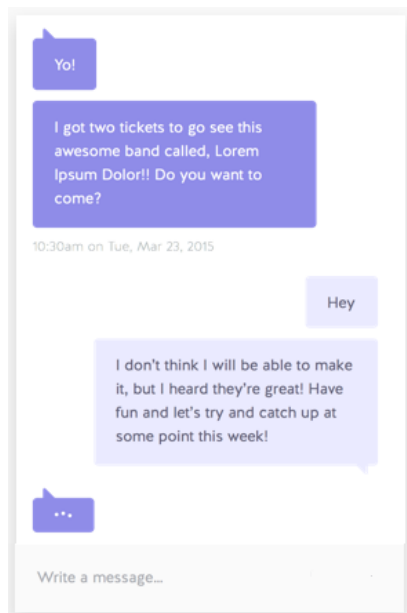
- Home(Misma funcionalidad que si no esta logueado lo único que con la posibilidad de marcar propiedades como favoritas)
- Agentes
- Mis propiedades (aquí solo se debe listar las propiedades que estén disponible que el cliente a marcado como favoritas)
- Cerrar sesión(al hacer click se elimina la sesión y manda el usuario a la pantalla de home de la app y se muestra el menú normal de la app(el descrito en la sesión de home del documento)).

Detalles de propiedades(Cliente)

Cuando el usuario esté logueado como un cliente e ingrese a la sección del detalle de una propiedad aparte de mostrarse los datos del detalle de la propiedad se habilitarán dos nuevas funcionalidades.

la primera de esa funcionalidad será una sesión de chat, donde habrá una caja de texto para que el cliente pueda dejarle un mensaje al agente que está llevando la propiedad y si el agente le responde se le mostrará en esa misma sección la respuesta del agente de manera que puede establecer una conversación con el agente inmobiliario.

Un ejemplo de esta sección sería algo parecido a esto:



La otra funcionalidad es permitirle al cliente poder hacer una oferta por la propiedad, para esto debe haber una sección con el listado de ofertas, aquí se muestran un listado con todas las ofertas que ha realizado el cliente para esa propiedad, de las ofertas se debe mostrar la fecha en la que se realizó la oferta, la cifra que se ofertó y el estado en el que se encuentra la oferta, la oferta tiene tres estados uno es cuando se crea la oferta que la misma se crea en estado pendiente, luego si el agente rechaza la oferta cambia a estado rechazada y si el agente la acepta se pone en estado aceptada.

Arriba del listado de oferta debe haber un botón para enviar una nueva oferta (este botón se deshabilita cuando ya exista una oferta aprobada para esta propiedad aunque sea de otro cliente o haya una oferta en estado pendiente del cliente logueado), al hacer click sobre este botón y que el mismo esté habilitado se mostrará un modal con un campo de texto donde el cliente pueda colocar la cifra de la oferta y un botón para enviar esa cifra y se creará una nueva oferta con esa cifra, con la fecha actual y estado pendiente.

Funcionalidades del Agente

Home del agente

Si el usuario que inicio sesión es del rol agente el sistema lo debe redireccionar a una pantalla donde se listen todas las propiedades que este agente ha registrado tanto la que está disponible como la que está vendida, y la que fueron vendida deben tener una etiqueta que indique que la misma ya fue vendida, de las propiedades se debe mostrar las misma informaciones que en el listado del home. Aparte en esta pantalla el agente tendrá el siguiente menú:

- Home del agente
- Mi perfil
- Mantenimiento de propiedades
- Cerrar sesión(al hacer click se elimina la sesión y manda el usuario a la pantalla de home de la app y se muestra el menú normal de la app(el descrito en la sesión de home del documento)).

Detalle de propiedades(Agente)

Cuando el usuario esté logueado como un agente e ingrese a la sección del detalle de una propiedad aparte de mostrarse los datos del detalle de la propiedad se habilitarán dos nuevas funcionalidades.

la primera de esa funcionalidad será una sesión de chat, aquí el agente puede mantener conversación con varios clientes a la vez, por lo tanto en esta sección, aquí al agente se le mostrará un listado con todos los nombre de los clientes y al hacer click sobre el nombre de un cliente se enviará a una nueva pantalla donde se mostrará el chat que se ha tenido con ese cliente y donde aparte de poder ver el chat el agente tendrá una caja de texto para poder responderle a ese cliente.

La otra funcionalidad es permitirle al agente poder responder a las diferentes ofertas que le envían los clientes para esa propiedad, aquí se muestra un listado con los nombre de los clientes que ha realizado oferta, al hacer click sobre el nombre de un cliente debe enviarlo a una pantalla nueva donde se listen todas las ofertas que ha realizado el cliente seleccionado, de las ofertas se debe mostrar la fecha de realización de la oferta, la cifra que se ofreció y el estado de la oferta.

En ese listado si hay una oferta en estado pendiente debe darle la opción al agente de responder y decir si acepta o rechaza la oferta, al rechazar la oferta la misma se marca con el estado rechazada y si se acepta la oferta se marca con el estado aceptada(Aquí tener en cuenta que si el agente acepta una oferta todas las demás ofertas de esta propiedad que se encuentre en estado pendiente aunque sean de otro cliente debe ser rechazada de manera automática y la propiedad debe ponerse en estado vendida y no permite a ningún otro cliente poder hacer más oferta sobre esta propiedad).

Mantenimiento de propiedades

Al ingresar a la opción de mantenimiento de propiedades desde el menú principal de la app, esta nos enviará al mantenimiento de propiedades.

En la pantalla inicial de este mantenimiento debemos tener un listado con todas las propiedades creadas y disponible por el agente que está logueado(no listar las vendidas), de las propiedades se debe mostrar las misma informaciones que en el home, aparte de mostrar esta información de cada propiedad se debe tener un botón para editarla y otro para eliminarla.

Arriba del listado de propiedades debe haber un botón para crear una propiedad nueva, al hacer clic sobre este botón de crear se enviará al usuario a una pantalla nueva donde habrá un formulario con los siguientes campos: un select con los tipos de propiedades creada en el sistema, un select con los tipos de ventas creada en el sistema, un campo para colocar el precio de la propiedad (En peso dominicano el precio), un campo para la descripción de la propiedad, un campo para poner el tamaño de la propiedad en metro, otro campo para poner la cantidad de habitaciones que tiene la propiedad, otro campo para poner la cantidad de baños que tiene la propiedad, un select múltiple con todas las mejoras creada en el sistema y un campo o varias campos que permita seleccionar de 1 a 4 imagenes (Si no existen tipo de propiedad o tipo de ventas o mejoras no debe dejarnos crear propiedades y mostranos un mensaje que nos indique que no hay tipo de propiedades o tipo de ventas o mejoras creada).

Este formulario debe tener las siguientes validaciones que todos los campos son requeridos y que en el caso de las imágenes de las 4 debe haber al menos una.

Al final del formulario de creación debe haber dos botones, un botón para volver atrás que devuelve a la pantalla inicial del mantenimiento donde están listadas las propiedades y otro botón para crear la propiedad, al hacer clic en el botón de crear

se envía la información y se crea la propiedad, al momento de crear la propiedad se debe generar un código único de 6 dígitos que no se puede repetir y se le debe asignar a esa propiedad que se está creando, la propiedad se crea en estado disponible, una vez creada la misma se debe redirigir a la pantalla inicial de mantenimiento de propiedades.

Por otra parte, En el listado de propiedades del mantenimiento, cada propiedad tiene su botón para editarla o borrarla, al presionar sobre el botón editar se debe enviar al usuario a una pantalla con un formulario con los siguientes campos: un select con los tipos de propiedades creada en el sistema, un select con los tipos de ventas creada en el sistema, un campo para colocar el precio de la propiedad (En peso dominicano el precio), un campo para la descripción de la propiedad, un campo para poner el tamaño de la propiedad en metro, otro campo para poner la cantidad de habitaciones que tiene la propiedad, otro campo para poner la cantidad de baños que tiene la propiedad, un select múltiple con todas las mejoras creada en el sistema y un campo o varios campos que permita seleccionar de 1 a 4 imágenes (Si no existen tipo de propiedad o tipo de ventas o mejoras no debe dejarnos crear propiedades y mostrarnos un mensaje que nos indique que no hay tipo de propiedades o tipo de ventas o mejoras creada), como es una edición estos campos debe venir con los valores guardados para la propiedad que estamos editando, incluso se deben visualizar las imágenes que tiene la propiedad.

Este formulario debe tener las siguientes validaciones que todos los campos son requeridos, menos las imágenes.

Al final del formulario de edición debe haber dos botones, un botón para volver atrás que devuelve a la pantalla inicial del mantenimiento donde están listadas las propiedades y otro botón para guardar la propiedad el cual al hacer clic en el botón se envía la información y se edita la propiedad con los valores nuevo, una vez editada la propiedad se debe redirigir a la pantalla inicial del mantenimiento de propiedades.

Finalmente al presionar sobre el botón de eliminar se envía al usuario a una pantalla con un mensaje que debe decir: ¿Está seguro que desea eliminar esta propiedad? debajo de este mensaje dos botones uno de cancelar y otro de aceptar, si se pulsa sobre cancelar se envía a la pantalla inicial del mantenimiento de propiedades (El listado), pero al presionar sobre el aceptar se elimina la propiedad y se redirecciona a la pantalla inicial del mantenimiento de propiedades (El listado).

Mi perfil

Al ingresar en esta pantalla se debe mostrar un formulario que permita modificar las siguientes informaciones del agente logueado.

Los campos de este formulario son los siguientes: nombre, apellido, teléfono, foto de usuario(del agente), estos campos tienen que venir con los valores guardados actualmente.

Este formulario debe tener dos botones, un botón para volver atrás al home del agente y otro botón para guardar la información que se edite del agente con las nuevas colocadas en el formulario.

la validación de este formulario que todos los campos son requeridos, menos la foto del usuario.

Funcionalidades del Administrador

Home del administrador

Si el usuario que inicio sesión es del rol administrador el sistema lo debe redireccionar a la pantalla inicial de un administrador se mostrará un menú con las siguientes opciones:

- Home del administrador
- Listado de los agentes
- Mantenimiento de administradores
- Mantenimiento de desarrolladores
- Mantenimiento de tipo de propiedades
- Mantenimiento de tipo de ventas
- Mantenimiento de mejoras

Aparte del menú debe mostrar en esa pantalla los siguientes indicadores:

- * Las cantidades de propiedades disponibles y vendidas registradas en el sistema.

- * La cantidad de agentes activos y los inactivos

- * La cantidad de clientes activos y los inactivos.

- * La cantidad de desarrolladores activos y los inactivos.

Listado de los agentes

Al ingresar en esta opción se nos debe mostrar una pantalla con un listado de los usuario con el rol de agente, en este listado de los agentes se debe mostrar el nombre , el apellido, la cantidad de propiedades que tiene este agente y el correo.

En este listado también debe haber dos botones: un botón para activar o inactivar un agente y otro botón para eliminar un agente.

Al presionar sobre el botón de eliminar se envía al usuario a una pantalla con un mensaje que debe decir: ¿Está seguro que desea eliminar este agente y todas sus propiedades ? Debajo de este mensaje dos botones uno de cancelar y otro de aceptar, si se pulsa sobre cancelar se envía a la pantalla del listado de agentes, pero al presionar sobre el aceptar se elimina el agente y sus propiedades y se redirecciona a la pantalla del listado de agentes.

Mantenimiento de administradores

Al ingresar en esta opción se nos debe mostrar una pantalla con un listado de los usuario con el rol de administrador, en este listado de los administradores se debe mostrar el nombre , el apellido, el nombre de usuario, cédula y el correo.

En la parte superior del listado hay un botón para crear administradores al hacer clic sobre este botón se envía a una nueva pantalla o se abre un modal con un formulario con los siguientes campos, Nombre , Apellido , Cédula , Correo , Usuario , Contraseña , confirmar contraseña, este formulario debe tener dos botones uno para volver al listado de administradores y otro para crear el

usuario tipo administrador.

Este formulario tiene las siguientes validaciones, todos los campos son requeridos. Una vez creado el usuario administrador(el mismo se crea en estado activo por defecto) debe enviarnos al listado del mantenimiento de administradores.

Volviendo al listado de administradores desde este se pueden inactivar o activar un administrador (El usuario administrador logueado en el momento no puede ni editar ni activar su usuario), para inactivar un usuario existe un botón que al pulsar sobre este te sale una confirmación de si esta seguro que desea inactivar ese usuario si la respuesta es Si se inactiva , si la respuesta es No se cierra la confirmación y no se realiza ninguna acción. Si un usuario ya se encuentra inactivo se mostrará un botón que permitirá activar al usuario e igual al pulsar sobre este se le preguntará si está seguro que desea activar ese usuario.

Otra acción que se puede realizar sobre un usuario es la edición del mismo, al editar el administrador se puede modificar los siguientes campos (Nombre , Apellido , Cédula , Correo , Usuario , Contraseña), para modificar la contraseña se debe tener con el campo confirmar contraseña.

Este formulario tiene las siguientes validaciones, todos los campos son requeridos. Una vez editado el usuario administrador debe enviarnos al listado de mantenimiento de administradores.

Mantenimiento de desarrolladores

Al ingresar en esta opción se nos debe mostrar una pantalla con un listado de los usuario con el rol de desarrollador, en este listado de los desarrolladores se debe mostrar el nombre , el apellido, el nombre de usuario, cédula y el correo.

En la parte superior del listado hay un botón para crear desarrollador al hacer clic sobre este botón se envía a una nueva pantalla o se abre un modal con un formulario con los siguientes campos, Nombre , Apellido , Cédula , Correo , Usuario , Contraseña , confirmar contraseña, este formulario debe tener dos botones uno para volver al listado de desarrolladores y otro para crear el usuario tipo desarrollador.

Este formulario tiene las siguientes validaciones, todos los campos son requeridos. Una vez creado el usuario desarrollador (el mismo se crea en estado activo por defecto) debe enviarnos al listado del mantenimiento de desarrolladores.

Volviendo al listado de desarrolladores desde este se pueden inactivar o activar un desarrollador, para inactivar un usuario existe un botón que al pulsar sobre este te sale una confirmación de si esta seguro que desea inactivar ese usuario si la respuesta es Si se inactiva , si la respuesta es No se cierra la confirmación y no se realiza ninguna acción. Si un usuario ya se encuentra inactivo se mostrará un botón que permitirá activar al usuario e igual al pulsar sobre este se le preguntará si está seguro que desea activar ese usuario.

Otra acción que se puede realizar sobre un usuario es la edición del mismo, al editar el desarrollador se puede modificar los siguientes campos (Nombre , Apellido , Cédula , Correo , Usuario , Contraseña), para modificar la contraseña se debe tener con el campo confirmar contraseña.

Este formulario tiene las siguientes validaciones, todos los campos son requeridos. Una vez editado el usuario desarrollador debe enviarnos al listado del mantenimiento de desarrolladores.

Mantenimiento de tipo de propiedades

Al ingresar a la opción de mantenimiento de tipo de propiedades desde el menú principal de la app, esta nos enviará al mantenimiento de tipo de propiedades.

En la pantalla inicial de este mantenimiento debemos tener un listado con todos los tipo de propiedades creado en el sistema, de los tipo de propiedades se debe mostrar el nombre del tipo, la descripción del mismo y la cantidad de propiedades con este tipo asignado, aparte de mostrar esta información cada tipo de propiedad debe tener un botón para editarlo y otro para eliminarlo.

Arriba del listado de tipo de propiedades debe haber un botón para crear un tipo de propiedad nuevo, al hacer clic sobre este botón de crear se enviará al usuario a una pantalla nueva donde habrá un formulario con los siguientes campos: nombre del tipo y descripción.

Este formulario debe tener las siguientes validaciones que todos los campos son requeridos.

Al final del formulario de creación debe haber dos botones, un botón para volver atrás que devuelve a la pantalla inicial del mantenimiento donde están listados los tipo de propiedades y otro botón para crear el tipo de propiedad, al hacer clic en el botón se envía la información y se crea el tipo de propiedad , una vez creado el tipo de propiedad se debe redirigir a la pantalla inicial del mantenimiento de tipo de propiedades.

Por otra parte, En el listado de tipo de propiedades del mantenimiento, cada tipo de propiedad tiene su botón para editarlo o borrarlo, al presionar sobre el botón editar se debe enviar al usuario a una pantalla con un formulario con los siguientes campos: nombre del tipo y descripción del mismo , como es una edición estos campos debe venir con los valores guardados para el tipo de propiedad que estamos editando

Este formulario debe tener las siguientes validaciones que todos los campos son requeridos.

Al final del formulario de edición debe haber dos botones, un botón para volver atrás que devuelve a la pantalla inicial del mantenimiento donde están listados los tipo de propiedades y otro botón para guardar el tipo de propiedad el cual al hacer clic en el botón se envía la información y se edita el tipo de propiedad con los valores nuevo, una vez editado el tipo de propiedad se debe redirigir a la pantalla inicial de mantenimiento de tipo de propiedades.

Finalmente al presionar sobre el botón de eliminar se envía al usuario a una pantalla con un mensaje que debe decir: ¿Está seguro que desea eliminar este tipo de propiedad y todas las propiedades con este tipo? debajo de este mensaje dos botones uno de cancelar y otro de aceptar, si se pulsa sobre cancelar se envía a la pantalla inicial del mantenimiento de tipo de propiedades (El listado), pero al presionar sobre el aceptar se elimina el tipo de propiedad y todas las propiedades asociadas y se redirecciona a la pantalla inicial del mantenimiento de tipo de propiedades(El listado).

Mantenimiento de tipo de ventas

Al ingresar a la opción de mantenimiento de tipo de ventas desde el menú principal de la app, esta nos enviará al mantenimiento de tipo de ventas.

En la pantalla inicial de este mantenimiento debemos tener un listado con todos los tipo de ventas creado en el sistema, de los tipo de ventas se debe mostrar el nombre del tipo, la descripción y la cantidad de propiedades que tienen asociado este tipo de venta, aparte de mostrar esta información cada tipo de venta debe tener un botón para editarlo y otro para eliminarlo.

Arriba del listado de tipo de ventas debe haber un botón para crear un tipo de venta nuevo, al hacer clic sobre este botón de crear se enviará al usuario a una pantalla nueva donde habrá un formulario con los siguientes campos: nombre del tipo y descripción.

Este formulario debe tener las siguientes validaciones que todos los campos son requeridos.

Al final del formulario de creación debe haber dos botones, un botón para volver atrás que devuelve a la pantalla inicial del mantenimiento donde están listados los tipo de ventas y otro botón para crear el tipo de venta, al hacer clic en el botón se envía la información y se crea el tipo de venta, una vez creado el tipo de venta se debe redirigir a la pantalla inicial del mantenimiento de tipo de ventas.

Por otra parte, En el listado de tipo de ventas del mantenimiento, cada tipo de venta tiene su botón para editarlo o borrarlo, al presionar sobre el botón editar se debe enviar al usuario a una pantalla con un formulario con los siguientes campos: nombre del tipo y descripción, como es una edición estos campos debe venir con los valores guardados para el tipo de venta que estamos editando

Este formulario debe tener las siguientes validaciones que todos los campos son requeridos.

Al final del formulario de edición debe haber dos botones, un botón para volver atrás que devuelve a la pantalla inicial del mantenimiento donde están listados los tipo de ventas y otro botón para guardar el tipo de venta el cual al hacer clic

en el botón se envía la información y se edita el tipo de venta con los valores nuevo, una vez editado el tipo de venta se debe redirigir a la pantalla inicial de mantenimiento de tipo de ventas.

Finalmente al presionar sobre el botón de eliminar se envía al usuario a una pantalla con un mensaje que debe decir: ¿Está seguro que desea eliminar este tipo de venta y las propiedades asociadas? debajo de este mensaje dos botones uno de cancelar y otro de aceptar, si se pulsa sobre cancelar se envía a la pantalla inicial del mantenimiento de tipo de ventas (El listado), pero al presionar sobre el aceptar se elimina el tipo de venta y sus propiedades y se redirecciona a la pantalla inicial del mantenimiento de tipo de ventas (El listado).

Mantenimiento de mejoras

Al ingresar a la opción de mantenimiento de mejoras desde el menú principal de la app, esta nos enviará al mantenimiento de mejoras.

En la pantalla inicial de este mantenimiento debemos tener un listado con todas las mejoras creadas en el sistema, de las mejoras se debe mostrar el nombre de la mejora y la descripción, aparte de mostrar esta información cada mejora debe tener un botón para editarla y otro para eliminarla.

Arriba del listado de mejoras debe haber un botón para crear una mejora nueva, al hacer clic sobre este botón de crear se enviará al usuario a una pantalla nueva donde habrá un formulario con los siguientes campos: nombre de la mejora y descripción.

Este formulario debe tener las siguientes validaciones que todos los campos son requeridos.

Al final del formulario de creación debe haber dos botones, un botón para volver atrás que devuelve a la pantalla inicial del mantenimiento donde están listadas las mejoras y otro botón para crear la mejora, al hacer clic en el botón se envía la información y se crea la mejora, una vez creada la mejora se debe redirigir a la pantalla inicial del mantenimiento de mejoras.

Por otra parte, En el listado de mejoras del mantenimiento, cada mejora tiene su botón para editarla o borrarla, al presionar sobre el botón editar se debe enviar

al usuario a una pantalla con un formulario con los siguientes campos: nombre de la mejora y descripción de la misma , como es una edición estos campos debe venir con los valores guardados para la mejora que estamos editando

Este formulario debe tener las siguientes validaciones que todos los campos son requeridos.

Al final del formulario de edición debe haber dos botones, un botón para volver atrás que devuelve a la pantalla inicial del mantenimiento donde están listadas las mejoras y otro botón para guardar la mejora el cual al hacer clic en el botón se envía la información y se edita la mejora con los valores nuevo, una vez editada la mejora se debe redirigir a la pantalla inicial de mantenimiento de mejoras.

Finalmente al presionar sobre el botón de eliminar se envía al usuario a una pantalla con un mensaje que debe decir: ¿Está seguro que desea eliminar esta mejora? debajo de este mensaje dos botones uno de cancelar y otro de aceptar, si se pulsa sobre cancelar se envía a la pantalla inicial del mantenimiento de mejoras (El listado), pero al presionar sobre el aceptar se elimina la mejora y se redirecciona a la pantalla inicial del mantenimiento de mejoras(El listado).

Seguridad

Si un usuario no está logueado no puede acceder a las funcionalidades ni del administrador ni del cliente y ni del agente. Cuando intente entrar a esos enlaces y no esté logueado debe redirigirlo al home.

También se debe validar que si el usuario logueado es un cliente no puede ingresar a las opciones del administrador o agente o si un agente está logueado no puede entrar a las opciones de cliente o administrador y si un administrador está logueado no pueda entrar a las funcionalidades de los clientes o agentes , si intenta acceder a una sección que no tiene autorizado a entrar debe redirigirlo a una pantalla de acceso denegado desde donde haya un enlace a sus respectivos home.

Todas estas validaciones deben realizarse usando los filtros de authorize que implementa identity, también usando los seed debe crear un usuario default de tipo admin, otro de tipo cliente y otro de tipo agente.

Funcionalidades del Api

Funcionalidades generales

Login y Seguridad

Este sistema tendrá dos roles un rol administrador y un rol desarrollador, la seguridad en los endpoints del api deben de validarse a través del uso de JWT

El sistema debe crear por defectos estos roles y crear un usuario con el rol administrador, otro con el rol desarrollador.

El sistema tiene que tener un controlador para el manejo de las cuentas(AccountController) que tenga las siguientes funcionalidades:

- **Login:** (Permite autenticarse y obtener el token JWT para poder usar las demás funcionalidades de la aplicación)
- **Registro de usuario desarrollador:** (Permite crear usuario con el rol de desarrollador)
- **Registro de usuario administrador:** (Permite crear usuario con el rol de administrador este debe tener la seguridad de que debe haber un usuario logueado y que debe ser de tipo administrador para poder usar este endpoint)

Los usuarios se crean activos por lo cual no hay que hacer proceso de confirmación de cuenta. Si un usuario no está logueado no puede acceder a las funcionalidades ni del administrador ni del desarrollador. Cuando intente entrar a esos enlaces y no esté logueado debe mostrarle un mensaje de que no tiene autorización y el status code de la respuesta debe ser 401.

También se debe validar que si el usuario logueado es un desarrollador no puede ingresar a las opciones del administrador y viceversa que si un administrador está logueado no pueda entrar a las funcionalidades de los desarrolladores, si intenta acceder a una sección que no tiene autorizado a entrar debe retornar una respuesta con el mensaje de acceso denegado y status code debe ser 403.

Todas estas validaciones deben realizarse usando los filtros de authorize que

implementa identity con el uso de JWT.

Controlador de propiedades

Estos son los datos de una propiedad:

- Id
- Código
- Tipo de propiedad
- Tipo de venta
- Precio
- Tamaño del terreno en metro
- Cantidad de habitaciones
- Cantidad de baños
- Descripción
- Listados de las mejoras
- Nombre del agente
- Id del agente
- Estado de la propiedad(disponible o vendida)

Esto son los endpoint que debe tener el controlador de propiedades:

API Propiedades V 1.0					
Nombre	Método	Parámetros	Respuesta Ok	Respuesta KO	Roles permitidos
List	GET	-	200 HTTP (listado de todas las propiedades creada en formato json)	500 (error interno del servidor) 204 (no existen propiedades)	Administrador, Desarrollador
GetById	GET	- Id de la propiedad	200 HTTP(datos de la propiedad formato json)	500 (error interno del servidor) 204 (no existe la	Administrador, Desarrollador

				propiedad)	
GetByCode	GET	- Código de la propiedad	200 HTTP(datos de la propiedad formato json)	500 (error interno del servidor) 204 (no existe la propiedad)	Administrador, Desarrollador

Controlador de agentes

Estos son los datos de un agente:

- Id
- Nombre
- Apellido
- Cantidad de propiedades
- Correo
- Teléfono

Esto son los endpoint que debe tener el controlador de agentes:

API Agentes V 1.0					
Nombre	Método	Parámetros	Respuesta Ok	Respuesta KO	Roles permitidos
List	GET	-	200 HTTP (listado de todos los agentes creado en formato json)	500 (error interno del servidor) 204 (no existen agentes)	Administrador, Desarrollador
GetById	GET	- Id del agente	200 HTTP(datos del agente formato json)	500 (error interno del servidor)	Administrador, Desarrollador

				204 (no existe el agente)	
GetAgentProperty	GET	- Id del agente	200 HTTP(datos de las propiedades del id del agente que pasaron por parámetros formato json)	500 (error interno del servidor) 204 (no existen la propiedades)	Administrador, Desarrollador
ChangeStatus	ChangeStatus	- Id del agente - Estatus(true[activo] o false[inactivo])	204 HTTP (sin contenido)	500 error interno del servidor	Administrador

Mant. de tipo de propiedades

Estos son los datos de un tipo de propiedades:

- Id
- Nombre
- Descripción

Esto son los endpoint que debe tener el controlador de tipo de propiedades:

API Tipo De Propiedades V 1.0					
Nombre	Método	Parámetros	Respuesta Ok	Respuesta KO	Roles permitidos
Create	POST	- Datos del tipo de propiedades	201 HTTP (created)	500 error interno del servidor	Administrador

				400(error bad request)	
Update	PUT	<ul style="list-style-type: none"> - Código del tipo de propiedad - Datos del tipo de propiedad 	200 HTTP (datos de el tipo de propiedad actualizado en formato json)	500 (error interno del servidor) 400(error bad request)	Administrador
List	GET	-	200 HTTP (listado de los tipo de propiedades en formato json)	500 (error interno del servidor) 204 (no existen tipos de propiedades)	Administrador, Desarrollador
GetByld	GET	- Código del tipo de propiedad	200 HTTP (datos de el tipo de propiedad en formato json)	500 (error interno del servidor) 204 (no existe el tipo de propiedad)	Administrador, Desarrollador
Delete	Delete	- Código del tipo de propiedad	204 HTTP (sin contenido) Se elimina el tipo de propiedad con sus propiedades	500 (error interno del servidor)	Administrador

Mant. de tipo de ventas

Estos son los datos de un tipo de venta:

- Id
- Nombre
- Descripción

Esto son los endpoint que debe tener el controlador de tipo de ventas:

API Tipo De Ventas V 1.0					
Nombre	Método	Parámetros	Respuesta Ok	Respuesta KO	Roles permitidos
Create	POST	- Datos del tipo de venta	201 HTTP(created)	500 error interno del servidor 400(error bad request)	Administrador
Update	PUT	- Código del tipo de venta - Datos del tipo de venta	200 HTTP (datos de el tipo de venta actualizado en formato json)	500 (error interno del servidor) 400(error bad request)	Administrador
List	GET	-	200 HTTP (listado de los tipo de ventas en formato json)	500 (error interno del servidor) 204 (no existen tipos de ventas)	Administrador,De sarrollador

GetByld	GET	- Código del tipo de venta	200 HTTP(datos de el tipo de venta en formato json)	500 (error interno del servidor) 204 (no existe el tipo de venta)	Administrador,De sarrollador
Delete	Delete	- Código del tipo de venta	204 HTTP (sin contenido) Se elimina el tipo de venta con sus propiedades	500 (error interno del servidor)	Administrador

Mant. de mejoras

Estos son los datos de una mejora:

- Id
- Nombre
- Descripción

Esto son los endpoint que debe tener el controlador de mejoras:

API Mejoras V 1.0					
Nombre	Método	Parámetros	Respuesta Ok	Respuesta KO	Roles permitidos
Create	POST	- Datos de la mejora	201 HTTP (created)	500 error interno del servidor 400(error bad request)	Administrador

Update	PUT	<ul style="list-style-type: none"> - Código de la mejora - Datos de la mejora 	200 HTTP (datos de la mejora actualizado en formato json)	500 (error interno del servidor) 400(error bad request)	Administrador
List	GET	-	200 HTTP (listado de las mejoras en formato json)	500 (error interno del servidor) 204 (no existen mejoras)	Administrador, Desarrollador
GetByld	GET	- Código de la mejora	200 HTTP (datos de la mejora en formato json)	500 (error interno del servidor) 204 (no existe la mejora)	Administrador, Desarrollador
Delete	Delete	- Código de la mejora	204 HTTP (sin contenido) Se elimina la mejora	500 (error interno del servidor)	Administrador

Requerimientos técnicos

ViewModels y Validaciones

- Se deben utilizar **ViewModels** para la capa de presentación del web app.
- Todas las validaciones deben implementarse directamente en los **ViewModels**, aprovechando las herramientas de validación del framework.

Persistencia de Datos

- Se debe utilizar **Entity Framework Core con el enfoque Code First** para la gestión y persistencia de datos.

Interfaz de Usuario

- El proyecto debe contar con una interfaz **intuitiva y visualmente clara**, utilizando **Bootstrap** como framework de diseño o cualquier otro framework de css.

Arquitectura

- El proyecto debe implementar **Onion Architecture**, aplicada de manera correcta y consistente al **100%**.
- Cualquier error en su aplicación será considerado como una implementación incorrecta.

Repositorios y Servicios

- Se deben crear **repositorios genéricos** y **servicios genéricos** para la gestión de datos.
- Los servicios deben ser utilizados por los controladores de la **WebApp**.

Autenticación y Seguridad

- Se debe implementar **ASP.NET Identity** para el manejo de usuarios y roles.
- La **API** debe estar protegida mediante **JWT (JSON Web Tokens)**.
- Si un usuario de tipo **desarrollador** intenta iniciar sesión en la **WebApp**, se le debe notificar que no tiene permisos para acceder.
- Si un usuario de tipo **cliente** o **agente** intenta iniciar sesión en la **Web API**, se le debe indicar que no tiene permisos para acceder.

Mapeo de Datos

- Se debe utilizar **AutoMapper** para el mapeo entre **ViewModels**, **Entities** y **DTOs**.

Patrones de Diseño

- Se deben implementar los patrones **CQRS** y **Mediator** en los endpoints de la **API**.
- Se deben utilizar **Behaviors** para las validaciones de los **Commands** y **Queries**, empleando **FluentValidation**.

Documentación de la API

- La **API** debe documentarse utilizando **Swagger** y sus herramientas de generación de documentación.

Manejo de Excepciones

- Las excepciones deben manejarse de forma centralizada utilizando un **Global Exception Handler**.
- Las respuestas de error deben generarse con el estándar **Problem Details (RFC 7807)**, tanto en la **API** como en la **WebApp**.

Requisitos Funcionales Específicos

- Todos los precios de las propiedades deben estar expresados en **pesos dominicanos (DOP)**.
- Todas las pantallas que listan propiedades deben incluir los **mismos filtros** definidos en la sección **Home** del documento.

Pruebas Automatizadas

- Se deben implementar **pruebas unitarias con xUnit** para:
 - **Commands** y **Queries** de la capa de aplicación.
 - **Servicios** (services) de la lógica de negocio.
- Se deben realizar **pruebas de integración** para los **repositorios**, validando las operaciones de persistencia con la base de datos.