Comunnity House

INTEGRANTES:
-LAYNUS UGARTE
-Cristian Salvo
-Leonardo Castillo

Profesor: Ricardo Alberto Aravena Videla

Contexto

Community House es una aplicación móvil desarrollada con Ionic, Angular y Firebase, creada para mejorar la comunicación en comunidades residenciales.

Permite que residentes y conserjes se conecten de forma rápida y segura mediante un sistema de mensajería en tiempo real, un muro de anuncios y una bitácora digital para el registro de visitas.

Su objetivo principal es centralizar la información del edificio, reemplazando los métodos tradicionales (WhatsApp, carteles o llamadas) por una herramienta moderna, accesible y escalable.

El proyecto se enfoca en ofrecer una interfaz intuitiva, roles bien definidos y una arquitectura tecnológica robusta que garantice privacidad, eficiencia y trazabilidad dentro de la comunidad.

Alcances (que hace y que no hace) del proyecto

LO QUE HACE

AUTENTICACIÓN Y USUARIOS
Registro e inicio de sesión (Residente/Conserje)
Gestión de perfiles con foto
Estados: En línea/Desconectado

MENSAJERÍA EN TIEMPO REAL

Chat individual tipo WhatsApp
Historial de conversaciones
Lista de contactos
Mensajes leídos/no leídos

ANUNCIOS COMUNITARIOS
Crear y publicar anuncios
Adjuntar imágenes
Visualización para todos los usuarios
Categorización (perdidos, eventos, avisos)

BITÁCORA (SOLO CONSERJE)

Registro de entrada/salida de visitantes
Datos: nombre, documento, apartamento, hora, placa Vehiculo
Historial y búsqueda de visitas
Vista de visitantes del día

LO QUE NO HACE

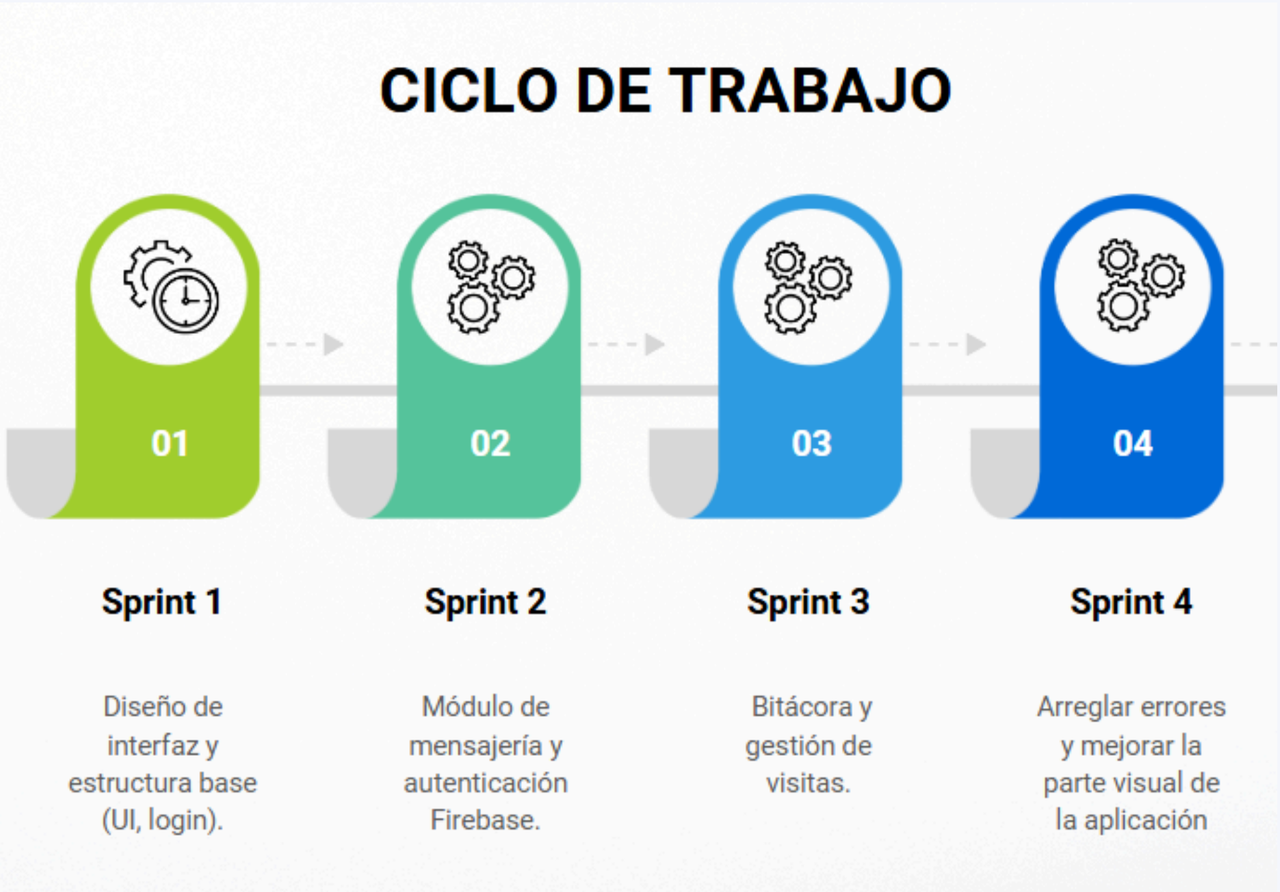
FUERA DE ALCANCE
❌ Pagos o gestión financiera
❌ Reserva de áreas comunes
❌ Videollamadas o llamadas de voz
❌ Votaciones o asambleas virtuales
❌ Gestión de mantenimiento
❌ Control de acceso físico (puertas/torniquetes)

Tipo	Resultado esperado	Resultado obtenido	Módulo	Resultado	Observación	Responsable	Complejidad
Funcional	Usuario creado y redirigido	Ejecutada	Auth	OK	La creación de la cuenta se realizó sin inconvenientes y los datos fueron almacenados correctamente.	Tester	Media
Funcional	Error de email duplicado	Ejecutada	Auth	OK	El sistema detectó correctamente que el email ya estaba registrado y mostró el mensaje correspondiente.	Tester	Media
Funcional	Acceso correcto	Ejecutada	Auth	OK	El sistema permitió el acceso correctamente y redirigió al usuario a su panel según su rol.	Tester	Media
Funcional	Mostrar error	Ejecutada	Auth	OK	La aplicación rechazó la autenticación con contraseña incorrecta y mostró el mensaje de error esperado sin exponer información sensible.	Tester	Media
Funcional	Sesión cerrada	Ejecutada	Auth	OK	La sesión fue cerrada correctamente y el usuario fue redirigido a la pantalla inicial sin errores.	Tester	Baja
Funcional	Entrega <2s	Ejecutada	Chat	OK	El mensaje fue enviado correctamente y apareció en ambos chats dentro del tiempo esperado.	Tester	Alta
Funcional	Lista cargada	Ejecutada	Chat	OK	La lista de contactos se cargó correctamente mostrando todos los usuarios excepto el actual.	Tester	Baja
Funcional	Mensajes previos cargados	Ejecutada	Chat	OK	El historial de mensajes cargó correctamente y mostró todas las conversaciones previas sin errores.	Tester	Media
Funcional	Badge visible	Ejecutada	Chat	OK	El estado en línea se mostró correctamente cuando el usuario estuvo activo en la aplicación.	Tester	Baja
No Funcional	Publicación correcta	Pendiente	Anuncios	OK	Al intentar crear un anuncio sin imagen, el sistema muestra un error y no permite completar la publicación.	Tester	Media
Funcional	Imagen publicada	Ejecutada	Anuncios	OK	El anuncio con imagen se creó correctamente y se mostró en la lista sin problemas.	Tester	Media
Funcional	Eliminado	Ejecutada	Anuncios	OK	El usuario pudo eliminar su anuncio correctamente y este fue removido de la lista sin errores.	Tester	Alta
Funcional	Botón no visible	Ejecutada	Anuncios	OK	El sistema ocultó correctamente el botón de eliminación para anuncios que no pertenecen al usuario, impidiendo su eliminación.	Tester	Baja
Funcional	Entrada registrada	Ejecutada	Bitácora	OK	La entrada del visitante fue registrada correctamente y se generó la hora de ingreso de forma automática.	Tester	Media
Funcional	Salida registrada	Ejecutada	Bitácora	OK	La salida del visitante fue registrada correctamente y se actualizó la hora de salida sin errores.	Tester	Media
Funcional	Solo activos	Ejecutada	Bitácora	OK	El filtro mostró únicamente los visitantes que mantienen una entrada activa y ocultó correctamente los que ya registraron salida.	Tester	Media

Metodología de Desarrollo

ÁGIL SCRUM

¿Por qué se eligió?
El proyecto requería retroalimentación constante del equipo y del profesor.
Permitió priorizar funcionalidades clave (login, chat, bitácora) en cada etapa.
Favoreció la entrega continua de versiones funcionales y pruebas rápidas.



Ventajas obtenidas
✅ Mejor comunicación del equipo
✅ Incrementos funcionales del producto en poco tiempo
✅ Mayor control de avances y prioridades
✅ Facilidad para adaptar cambios durante el desarrollo

Cronograma

FASES	ACTIVIDADES
DEFINICIÓN DE PROYECTO	REUNION INICIAL (KICK OFF)
	PLANILLA PRODUCT BACKING
	DEFINIR OBJETOS ACTORES Y REQUERIMIENTOS
	RADACCIÓN DE FICHA TÉCNICA
	EDT ORGANIGRAMA
	CRONOGRAMA
	ROADMAP
DISEÑO Y PROTOTIPO	DISEÑO DE ARQUITECTURA PROPUESTA
	CREACIÓN DE MUCKUP
	ELABORACIÓN DE DIAGRAMAS UML (CASO DE USO, ACTIVIDAD Y CLASES)
	DEFINICIÓN DE MVP
	PLAN DE PROYECTO
	PLAN RIESGO
	PLAN DE GESTION DE CALIDAD
	CONFIGURACION DE REPOSITORIO GITHUB
DESARROLLO INICIAL	IMPLEMENTACIÓN DE LOGIN
	IMPLEMENTACION DE REGISTER
	BITÁCORA DIGITAL (CONSERJE)
	VISTA DE USUARIOS(CONSERJE)
	VISTA DE EL PERFIL DE EL CONSERJE (USUARIO)
	LLAMADA AL RESIDENTE (VISITA)
	MENSAJERIA CON RESIDENTE
	VISTA DEL RESIDENTE
	COMENTARIOS DE LA ATENCION DEL CONSERJE DEL DIA
	NOTIFICACIÓIN PUSH (RESIDENTE)
	CHAT COMUNITARIO
PRUEBA Y AJUSTES	PRUEBA DE INTEGRACIÓN
	PRUEBAS DEL CONSERJE
	PRUEBAS DEL RESIDENTE (USUARIO)
	VALIDACION DE CONSERJE
	VALIDACIÓN DE USUARIO
	CORRECIÓN DE ERROES
ENTREGA FINAL	DOCUMENTACIÓN FINAL
	PREPARACIÓN DE PRESENTACIÓN
	DEMO DE SISTEMA

Lenguajes, frameworks y librerías.

Angular: framework frontend robusto para construir interfaces modernas y dinámicas.

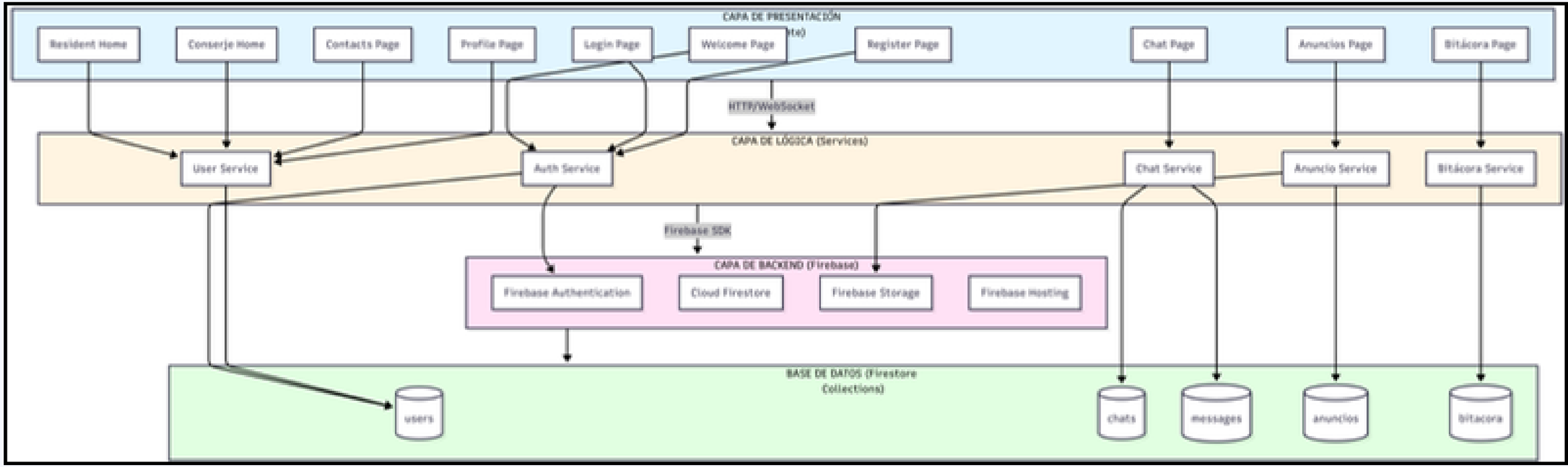
Ionic: permite crear aplicaciones móviles híbridas de forma rápida y multiplataforma.

Capacitor: facilita la integración con funcionalidades nativas del dispositivo (notificaciones, cámara, etc.).

Node.js: entorno de ejecución que permite manejar el backend en JavaScript con alto rendimiento.

Firebase (de momento): servicios en la nube para autenticación, base de datos y notificaciones push.

Arquitectura de la Solución



Demostración de la Solución