Manual técnico del sistema

Integrantes:

Juan Andrés Toro Blandón

Mariana Andrea Nisperuza Puerta

Yenifer Tamayo Villa

Centro tecnológico del mobiliario Itagüí

Fecha:

25/02/2025

CONTENIDO

LISTA DE TABLAS

PRESENTACIÓN

El avance de la tecnología ha revolucionado la manera en que las personas se comunican, interactúan y colaboran en distintos ámbitos. En este contexto, Bubblebox surge como una red social innovadora diseñada para facilitar la conexión entre individuos, sin importar su experiencia o nivel de conocimientos, brindando un espacio ideal para el intercambio de ideas, la colaboración en proyectos y el aprendizaje mutuo.

Con el propósito de garantizar un uso eficiente y una gestión adecuada de la plataforma, se ha desarrollado el presente manual, el cual proporciona toda la información necesaria para la instalación, mantenimiento y exploración de Bubblebox. Este documento está dirigido tanto a desarrolladores como a administradores de la plataforma, brindando una referencia completa que permite comprender la arquitectura del sistema y realizar modificaciones o mejoras según sea necesario.

El manual ofrece una guía detallada sobre la estructura del software, lo que facilita a los desarrolladores que trabajan con ReactJS y NodeJS comprender el funcionamiento de la plataforma y llevar a cabo tareas de desarrollo y optimización de manera eficiente. Asimismo, se proporciona información relevante sobre las tecnologías utilizadas, los diagramas de modelado de datos y la configuración del entorno de desarrollo, aspectos fundamentales para la evolución y mejora continua del sistema.

La documentación aquí presentada tiene como objetivo garantizar la sostenibilidad y escalabilidad de Bubblebox, permitiendo que su comunidad de usuarios disfrute de una experiencia fluida, segura y enriquecedora. Se espera que este manual sirva como una herramienta de referencia clave para todos aquellos que contribuyan al crecimiento y fortalecimiento de la plataforma.

RESUMEN

Este manual proporciona una descripción detallada de los aspectos técnicos e informáticos del software **Bubblebox**, con el objetivo de explicar su estructura a quienes deseen administrarlo, editarlo o configurarlo.

El documento está organizado en secciones que abarcan las herramientas utilizadas en el desarrollo del software, incluyendo una explicación paso a paso de su implementación. Además, se detallan las distintas funcionalidades de la plataforma, especificando los requisitos de hardware y software necesarios para su correcto funcionamiento.

Asimismo, se ofrecen recomendaciones para el uso adecuado del sistema de información, garantizando su eficiencia, seguridad y escalabilidad.

OBJETIVO

Dar a conocer el uso adecuado del software Bubblebox, proporcionando una descripción detallada e ilustrada de los componentes y funcionalidades que garantizan el correcto funcionamiento del sistema de información.

FINALIDAD DEL MANUAL

Este manual técnico tiene como finalidad instruir a las personas que deseen administrar, editar o configurar el software Bubblebox, brindando las herramientas y conocimientos necesarios para su correcta gestión y mantenimiento.

INTRODUCCIÓN

El presente manual ha sido elaborado con el propósito de proporcionar una guía detallada sobre el software Bubblebox, abordando sus aspectos técnicos para que cualquier persona encargada de su administración, edición o configuración pueda hacerlo de manera eficiente y apropiada.

El documento está estructurado en diferentes secciones que permiten comprender a profundidad el funcionamiento del sistema de información:

- **Aspectos Teóricos:** Se presentan conceptos, definiciones y explicaciones de los componentes del software desde una perspectiva teórica, facilitando el entendimiento del sistema y sus herramientas.
- **Diagramas de Modelamiento:** Incluye diagramas e ilustraciones que representan visualmente el funcionamiento del aplicativo.
- **Aspecto Técnico del Desarrollo del Sistema:** Proporciona información detallada sobre los componentes del software desde un enfoque técnico, abarcando el almacenamiento de datos, la estructura de desarrollo y recomendaciones para su correcto uso.
- **Requerimientos del Software:** Especifica los requisitos básicos de hardware y software necesarios para el correcto funcionamiento de Bubblebox.

ASPECTOS TECNICOS

El software Bubblebox ha sido desarrollado con el objetivo de facilitar la conexión entre personas, fomentando la interacción y el aprendizaje colaborativo. Para garantizar su correcto funcionamiento, es fundamental comprender los aspectos técnicos que conforman su estructura y asegurar su adecuado mantenimiento.

Se recomienda que este manual sea utilizado únicamente por aquellas personas encargadas de administrar, editar o configurar Bubblebox, con el fin de preservar la integridad y seguridad de los datos almacenados en la base de datos, evitando posibles usos indebidos de la información.

A continuación, se detallan los principales aspectos técnicos del sistema:

- **Arquitectura del Software:** Bubblebox es una aplicación web desarrollada con ReactJS para el frontend y Node.js para el backend, utilizando MySQL como sistema de gestión de bases de datos.
- **Seguridad y Accesibilidad:** Se han implementado medidas de seguridad como autenticación de usuarios, encriptación de contraseñas y control de accesos para proteger la información almacenada en la base de datos.
- **Gestión de Datos:** La base de datos ha sido diseñada para almacenar información de usuarios, publicaciones, interacciones y demás datos relevantes para el funcionamiento de la plataforma.
- **Requisitos de Hardware y Software:** Se especifican las configuraciones mínimas y recomendadas necesarias para ejecutar el software de manera óptima.
- **Mantenimiento y Actualización:** Se brindan recomendaciones para la actualización del sistema, resolución de problemas y optimización del rendimiento.

HERRAMIENTAS UTILIZADAS PARA EL DESARROLLO

En esta sección se procede a explicar las herramientas informáticas empleadas para el desarrollo del aplicativo:

VISUAL STUDIO CODE

VSCode es un editor de código ligero, multiplataforma y altamente extensible, desarrollado por Microsoft. Ofrece soporte integrado para múltiples lenguajes de programación, resaltado de sintaxis, autocompletado inteligente y depuración avanzada. Además, permite la instalación de extensiones para mejorar la productividad, como herramientas para ReactJS, Node.js, Git y control de versiones.

GIT Y GITHUB

Git es un sistema de control de versiones distribuido que permite gestionar el código fuente de manera eficiente, facilitando el seguimiento de cambios y la colaboración entre desarrolladores. GitHub, por su parte, es una plataforma basada en la nube que proporciona repositorios remotos para almacenar proyectos, colaborar en código, realizar revisiones y gestionar versiones de software de manera organizada y segura.

POSTMAN

Postman es una herramienta utilizada para el desarrollo y prueba de APIs. Permite enviar solicitudes HTTP, visualizar respuestas, gestionar entornos y automatizar pruebas, facilitando la comunicación entre el frontend y el backend. Es ampliamente utilizada en el desarrollo de aplicaciones web y RESTful APIs, ya que simplifica la verificación del correcto funcionamiento de los servicios.

DIAGRAMAS DE MODELAMIENTO

Los diagramas de modelamiento representan visualmente la estructura, el flujo de datos y la interacción entre los diferentes componentes del sistema Bubblebox. Estos diagramas son esenciales para comprender el diseño, la arquitectura y el comportamiento de la plataforma.

A continuación, se presentan los principales diagramas utilizados en el desarrollo del sistema:

DIAGRAMA DE CLASES

El diagrama de clases está compuesto de las entidades y atributos que se crearon para el almacenamiento de datos del software.

Link: https://miro.com/app/board/uXjVIMtn6yA=/?share_link_id=692357557650

DIAGRAMA DE CASOS DE USO

En las siguientes ilustraciones, se detallan los papeles que desempeñan los diferentes actores en relación con el aplicativo Bubblebox. Estos diagramas muestran las interacciones que cada usuario puede realizar dentro del sistema, permitiendo visualizar las funcionalidades disponibles para cada uno.

Diagrama de caso de uso: https://drive.google.com/file/d/1ITVv5s99_l13wg9p2MrOrS1i4-R5KewB/view?usp=drive_link

ACTORES

Usuario No Autenticado

Representa a un visitante que aún no ha iniciado sesión en la plataforma.

Puede registrarse o iniciar sesión, pero no tiene acceso a funciones restringidas a usuarios autenticados.

Usuario Autenticado

Representa a un usuario que ha iniciado sesión en Bubblebox.

Puede realizar acciones como gestionar su perfil, interactuar con publicaciones, enviar mensajes y unirse a comunidades.

Administrador

Tiene permisos avanzados para moderar el contenido, gestionar usuarios y generar informes.

Se encarga de la seguridad y el mantenimiento de la plataforma.

Creador de Comunidad

Es un usuario con privilegios para administrar comunidades dentro de Bubblebox.

Puede aprobar miembros, gestionar publicaciones y moderar interacciones en su comunidad.

CASOS DE USO PRINCIPALES

1. Autenticación y Seguridad

- Registro de Usuario: Permite a nuevos usuarios crear una cuenta en la plataforma.
- Inicio de Sesión (JWT): Autenticación mediante JSON Web Token (JWT) para sesiones seguras.
- Recuperación de Contraseña: Proceso para restablecer la clave de acceso.
- Verificación en Dos Pasos: Método adicional de seguridad para proteger el acceso a la cuenta.
- Cerrar Sesión: Permite al usuario salir de la plataforma.
- Cerrar Sesión en Todos los Dispositivos: Opción para desconectarse de todas las sesiones activas.
- Recuperación de Cuenta por Contacto: Método alternativo para recuperar una cuenta en caso de pérdida de acceso.

2. Gestión de Perfil

- Actualizar Información Personal: Modificar datos del perfil como nombre, correo, etc.
- Configurar Privacidad y Seguridad: Ajustes sobre quién puede ver la información del usuario.
- Cambiar Avatar: Modificar la foto de perfil.
- Gestionar Preferencias de Notificaciones: Personalizar qué tipo de alertas recibir.

3. Amistades y Redes

- Enviar Solicitud de Amistad: Conectar con otros usuarios.
- Aceptar/Rechazar Solicitud: Decidir si se desea agregar a alguien como amigo.
- Bloquear Usuario: Restringir interacciones con usuarios no deseados.
- Ver Lista de Amigos: Consultar las conexiones del usuario.
- Ver Amigos en Común: Ver qué amigos comparten dos usuarios.

4. Publicaciones e Interacciones

- Crear Publicación: Publicar contenido en la red social.
- Editar/Eliminar Publicación: Modificar o eliminar posts creados.
- Comentar Publicaciones: Añadir comentarios en publicaciones de otros usuarios.
- Responder Comentarios: Permitir respuestas en hilos de comentarios.
- Dar Reacciones: Agregar me gusta, me encanta, etc., a publicaciones.
- Guardar Publicación: Opción para marcar publicaciones como favoritas o para revisarlas después.

5. Historias y Reels

- Crear Historia/Reel: Subir historias temporales o videos cortos.
- Ver Historias/Reels: Acceder al contenido efímero de otros usuarios.
- Comentar Historias/Reels: Participar en la interacción de estos contenidos.
- Reaccionar a Historias/Reels: Expresar emociones mediante reacciones.

6. Comunidades y Grupos

- Crear Comunidad: Establecer un espacio temático dentro de Bubblebox.
- Unirse a Comunidad: Participar en grupos de interés.
- Publicar en Comunidad: Compartir contenido con miembros de una comunidad.
- Administrar Comunidad: Gestionar miembros, publicaciones y configuraciones.

7. Mensajería y Notificaciones

- Enviar Mensaje Privado: Comunicación directa entre usuarios.
- Recibir Mensajes en Tiempo Real: Integración con WebSockets para mensajería instantánea.
- Marcar Mensajes como Leídos: Indicar que un mensaje ha sido visto.

8. Moderación y Seguridad

- Reportar Contenido Inapropiado: Alertar sobre publicaciones ofensivas.
- Reportar Usuario: Informar sobre cuentas que violan las normas.
- Revisar Reportes: Permitir a los administradores gestionar denuncias.
- Suspender Cuentas o Contenidos: Acción tomada por administradores para moderar el sistema.
- Filtrado de Contenido Automático: Implementación de IA o reglas para evitar publicaciones inadecuadas.

9. Dashboard del Administrador

- Gestionar Usuarios: Modificar, suspender o eliminar cuentas.
- Gestionar Publicaciones: Supervisar y moderar contenido.
- Generar Informes y Estadísticas: Obtener datos analíticos sobre la actividad de la plataforma.
- Configurar Reglas de la Plataforma: Definir políticas y restricciones.
- Gestionar Copias de Seguridad: Administrar respaldos del sistema.
- Configurar Seguridad Avanzada: Aplicar medidas adicionales de protección.

RELACIONES ENTRE CASOS DE USO

«include» (incluir) → Indica que un caso de uso es parte esencial de otro.

«extend» (extender) → Indica que un caso de uso puede ampliar opcionalmente el comportamiento de otro.

EJEMPLOS EN EL DIAGRAMA:

- Recuperación de contraseña → incluye Recuperación de cuenta por contacto.
- Publicar en comunidad → incluye Crear publicación.
- Revisar reportes → incluye Suspender cuentas o contenidos.
- Generar informes y estadísticas → incluye Gestionar publicaciones.

DIAGRAMA DE DB (BASE DE DATOS)

Este diagrama representa el esquema de una base de datos relacional diseñada para Bubblebox. La estructura está compuesta por múltiples tablas interconectadas, cada una con un propósito específico dentro del sistema. A continuación, se presenta una descripción general de los principales módulos que conforman la base de datos:

Link:https://drive.google.com/file/d/1uofI6h2PpZTTT_GWDqL5S8yBvdA6poJ2/view?usp=sharing

Descripción de las Tablas Principales

1. Usuarios y Configuración

- **usuarios**: Contiene los datos básicos de los usuarios, como nombre, email, contraseña, avatar, y estado de la cuenta.
- **configuraciones_usuarios**: Permite almacenar configuraciones personalizadas de cada usuario, como privacidad, notificaciones y visibilidad en línea.

2. Publicaciones e Interacción

- **publicaciones**: Guarda las publicaciones creadas por los usuarios, asociadas a una comunidad o individuales.
- **comentarios**: Tabla para los comentarios en publicaciones, con información del usuario que comenta y el contenido del comentario.
- **reacciones**: Permite registrar reacciones (me gusta, me encanta, etc.) a publicaciones o comentarios.
- **respuestas_comentarios_publicacion**: Almacena respuestas específicas a comentarios de publicaciones.
- **reels**: Guarda información sobre videos cortos publicados por los usuarios.

3. Grupos y Comunidades

- **grupos**: Representa grupos creados por los usuarios con una imagen y descripción.
- **miembros_grupo**: Relación entre usuarios y los grupos a los que pertenecen.
- **comunidades**: Contiene datos de comunidades dentro de la plataforma, con nombre, descripción y avatar.
- **usuarios_comunidad**: Relación entre usuarios y las comunidades a las que se han unido.

4. Mensajería y Comunicación

- **messages**: Tabla de mensajes privados enviados entre usuarios, con soporte para texto, audio y estado del mensaje.

5. Historias y Vistas

- **historias**: Almacena historias temporales publicadas por los usuarios con fecha de expiración.
- **vistas_historias**: Registra qué usuarios han visto las historias de otros.

6. Notificaciones y Reportes

- **notificaciones**: Guarda alertas enviadas a los usuarios por actividad relevante.
- **reportes**: Permite a los usuarios reportar contenido inadecuado o problemas en la plataforma.

7. Relaciones y Amistades

- **amistades**: Controla las solicitudes de amistad y el estado de las relaciones entre usuarios.
- **vista_sugerencias_amigos**: Una posible vista en la base de datos que sugiere amigos basados en relaciones en común.

8. Seguridad y Backups

- **backups**: Almacena registros de copias de seguridad de los datos del usuario.
- **backup_restores**: Lleva un control de las restauraciones de backups realizadas.

DIAGRAMA DE ARQUITECTURA

La arquitectura del sistema está basada en microservicios, lo que permite modularidad, escalabilidad y facilidad de mantenimiento. Cada servicio está diseñado para una función específica y se comunica con los demás a través del API Gateway.

Link:https://drive.google.com/file/d/1B83WaOw4Bb8_spEVVrCmkQ_PIK0qM05q/view?usp=drive_link

Cliente (Frontend)

El usuario accede a Bubblebox a través de una aplicación web (ReactJS) o móvil. El frontend interactúa con la API Gateway para solicitar información y enviar datos a los microservicios.

API Gateway

Este componente actúa como intermediario entre el frontend y los microservicios. Se encarga de:

- Enrutamiento: Redirigir las solicitudes al microservicio correspondiente.
- Autenticación: Verificar que los tokens JWT sean válidos.
- Balanceo de carga: Distribuir las solicitudes de manera eficiente entre instancias de los microservicios.

Microservicios Principales

Cada uno de estos microservicios es independiente y se encarga de una funcionalidad específica dentro de Bubblebox:

- **Auth Service (Servicio de Autenticación)**
  Gestiona el inicio de sesión y la autenticación con JWT.
  Implementa 2FA (Autenticación en dos pasos) para mayor seguridad.
- **User Service (Servicio de Usuarios)**
  Maneja la información del usuario: perfil, configuración y privacidad.
- **Friendships Service (Servicio de Amistades y Bloqueos)**
  Permite a los usuarios agregar/bloquear amigos.
  Implementa un sistema de sugerencias de amistad basado en amigos en común.

- **Posts Service (Servicio de Publicaciones)**
  Permite la creación, edición y eliminación de publicaciones.
  Maneja archivos multimedia (imágenes, videos).
- **Comments Service (Servicio de Comentarios)**
  Permite a los usuarios comentar y responder publicaciones.
  Gestiona moderación y eliminación de comentarios inadecuados.
- **Reactions Service (Servicio de Reacciones)**
  Gestiona las interacciones con publicaciones ("Me gusta", "Me encanta", etc.).
  Realiza un análisis de engagement de los usuarios.
- **Reels Service (Servicio de Reels)**
  Permite la subida, reproducción y eliminación de videos cortos.
  Controla la expiración automática de los reels
- **Stories Service (Servicio de Historias)**
  Permite la publicación de historias temporales con privacidad configurable.
  Controla la expiración automática de historias.
- **Communities Service (Servicio de Comunidades)**
  Permite la creación y administración de comunidades dentro de la red social.
  Gestiona la moderación y administración de miembros.
- **Chats Service (Servicio de Mensajería)**
  Gestiona la mensajería en tiempo real mediante WebSockets.
  Permite mensajes privados y en grupos con soporte para multimedia.
- **Notifications Service (Servicio de Notificaciones)**
  Genera notificaciones en la plataforma y notificaciones push.
  Notifica eventos importantes como menciones, reacciones o nuevos mensajes.
- **Reports Service (Servicio de Moderación y Reportes)**
  Permite denunciar contenido y usuarios.
  Gestiona la moderación automática de publicaciones y comentarios ofensivos.
- **Stats Service (Servicio de Estadísticas y Análisis)**
  Analiza tendencias dentro de la plataforma.
  Genera reportes de actividad para administradores.

- **Backups Service (Servicio de Copias de Seguridad)**
  Gestiona respaldo y restauración de datos para evitar pérdida de información.

Servicios Externos

Estos servicios son utilizados por los microservicios para funcionalidades adicionales:

- Cloud Storage: Almacena imágenes y videos subidos por los usuarios.
- Email Provider: Gestiona el envío de correos para recuperación de cuentas y notificaciones.

Base de Datos Unificada

Todos los microservicios almacenan datos en una base de datos centralizada (MySQL). Cada microservicio accede a sus respectivas tablas o colecciones de datos sin afectar el rendimiento de otros servicios

Flujo General de la Arquitectura

- Un usuario accede desde el **frontend (web/móvil)** y envía una solicitud.
- La solicitud llega al **API Gateway**, que la redirige al microservicio correspondiente.
- El **microservicio** procesa la solicitud y consulta la **base de datos** si es necesario.
- Si el microservicio necesita información de otro servicio (ejemplo: una publicación con comentarios), hace una llamada interna entre microservicios.
- La respuesta se envía de vuelta al **API Gateway**, que la devuelve al **frontend**.
- Si la solicitud genera una notificación (ejemplo: una reacción a un post), el **Notification Service** se encarga de enviarla.

DIAGRAMA DE SECUENCIA

El diagrama de secuencia representa la interacción entre los diferentes componentes del sistema en el tiempo. En este caso, muestra el flujo de eventos dentro de la red social Bubblebox.

Link:https://drive.google.com/file/d/1PuVGIHlukNi4MrDhsZw7og0JA3x8x-OE/view?usp=sharing

Elementos del diagrama:

1. **Actores principales:**
    - **Usuario:** Representa a una persona que interactúa con la red social.
    - **Sistema Bubblebox:** Procesa las acciones del usuario.
    - **Base de datos:** Almacena la información relacionada con usuarios, publicaciones, comentarios, etc.
2. **Flujo general de interacción:**
    - **Inicio de sesión:**
        - El usuario ingresa sus credenciales.
        - El sistema valida los datos en la base de datos.
        - Si las credenciales son correctas, se concede acceso; de lo contrario, se muestra un error.
    - **Publicación de contenido:**
        - El usuario crea una nueva publicación (texto, imagen o video).
        - El sistema guarda la publicación en la base de datos y la muestra en el feed.

- **Interacción con publicaciones:**
  - Un usuario puede reaccionar, comentar y compartir publicaciones.
  - El sistema actualiza la base de datos y refleja los cambios en la interfaz.
- **Gestión de amigos/comunidades:**
  - Un usuario puede enviar una solicitud de amistad o unirse a una comunidad.
  - El sistema verifica la acción y la actualiza en la base de datos.
- **Notificaciones:**
  - Se generan notificaciones cuando un usuario recibe una reacción, comentario o solicitud de amistad.
  - El sistema envía la notificación al destinatario correspondiente.

3. **Mensajes clave en el diagrama:**
   - Flechas indicando solicitudes y respuestas entre usuario, sistema y base de datos.
   - Procesos de validación y almacenamiento de datos.
   - Confirmaciones visuales en la interfaz del usuario después de cada acción.

El diagrama de secuencia refleja el comportamiento dinámico de Bubblebox y muestra cómo los usuarios interactúan con el sistema y la base de datos. Cada acción sigue un flujo estructurado que permite la correcta gestión de datos y la experiencia del usuario en la red social.

DIAGRAMA DE ESTADOS

El diagrama de estados muestra los diferentes estados por los que puede pasar una cuenta de usuario en la red social **Bubblebox**, así como las transiciones entre estos estados según diferentes eventos o acciones. A continuación, se detallan los principales estados y sus transiciones:

Link:https://drive.google.com/file/d/1ZLoBmARIexBtvAUfCdZSCetXeidBV2e2/view?usp=drive_link

Estados Principales:

1. **NoRegistrado**: Estado inicial de un usuario antes de crear una cuenta.
    - Transición: *Registro exitoso* → pasa a **Registrado**.
2. **Registrado**: El usuario ha completado el registro, pero aún no ha verificado su cuenta.
    - Transición: *Requiere verificación de correo* → pasa a **NoVerificado**.
3. **NoVerificado**: La cuenta está pendiente de confirmación de correo electrónico.
    - Transición: *Usuario confirma correo* → pasa a **Verificado**.
4. **Verificado**: El usuario ha confirmado su correo y puede acceder a la plataforma.
    - Transición: *Primer inicio de sesión exitoso* → pasa a **Activo**.
5. **Activo**: El usuario está autenticado y puede interactuar con la red social.
    - Transición: *Usuario cierra sesión* → pasa a **Desconectado**.
    - Transición: *Usuario desactiva su cuenta temporalmente* → pasa a **Desactivado**.
    - Transición: *Demasiados intentos de inicio fallidos* → pasa a **Bloqueado**.
    - Transición: *Reportes o infracciones detectadas* → pasa a **Suspendido**.

6. **Desconectado**: El usuario ha cerrado sesión, pero su cuenta sigue activa.
    - Transición: *Usuario inicia sesión nuevamente* → vuelve a **Activo**.
7. **Bloqueado**: La cuenta ha sido bloqueada debido a múltiples intentos fallidos de inicio de sesión.
    - Transición: *Usuario desbloquea cuenta con 2FA o soporte* → vuelve a **Activo**.
8. **Suspendido**: El usuario ha sido suspendido por reportes o infracciones.
    - Transición: *Revisión exitosa del moderador* → vuelve a **Activo**.
    - Transición: *Cuenta suspendida permanentemente* → pasa a **Eliminado**.
9. **Desactivado**: El usuario ha decidido desactivar temporalmente su cuenta.
    - Transición: *Usuario reactiva su cuenta* → vuelve a **Activo**.
10. **Eliminado**: La cuenta ha sido eliminada de forma definitiva.
    - Transición: No tiene transiciones de regreso.
11. **Baneo Definitivo**: La cuenta ha sido eliminada por infracciones graves y no puede ser recuperada.
    - Transición: No tiene transiciones de regreso.

Este diagrama refleja el ciclo de vida de un usuario dentro de Bubblebox, incluyendo estados normales, suspensiones y bloqueos. Se contemplan mecanismos de seguridad como la verificación de correo, bloqueos por intentos fallidos y sanciones por incumplimiento de normas.

DICCIONARIO DE DATOS

Para el almacenamiento de datos del software, se definen los campos necesarios para cada una de las entidades relacionadas con el aplicativo.

Tabla 1 Diccionario de datos modelo Usuarios

USUARIOS				
Nombre Campo	Tipo dato	PK/INDEX(FK)	Tamaño	Descripción
id	int	PK, AI		Identificador de usuario
nombre	varchar		250	Nombre real del usuario
username	varchar		250	Nombre ficticio que usara en el aplicativo
email	varchar	UNIQUE	250	Correo electrónico del usuario
contraseña	varchar		250	Contraseña que usara para iniciar sesión
avatar	varchar		250	Imagen de perfil del usuario
estado	enum		(“conectado”, “desconectado”, “suspendido”)	Identificador si esta conectado o desconectado en el aplicativo
lastSeen	timestamp			Ultima conexión
descripcion_usuario	text			Descripción libre que pondrá el usuario para expresarse
created_at	timestamp			Fecha de creación de la cuenta del usuario
secret2FA	varchar		255	Código que utiliza la autenticación de doble factor
token_recuperacion	varchar		255	Token de recuperación que se le envía al usuario a través de correo para realizar cambio de contraseña
token_expiracion	bigint			Token expiración que tiene el token de recuperación una vez este expire se eliminara el token de recuperación
rol	enum		(“usuario”, “administrador”)	Rol del usuario dentro del sistema

Tabla 2 Diccionario de datos modelo publicaciones

PUBLICACIONES				
Nombre Campo	Tipo dato	PK/INDEX(FK)	Tamaño	Descripción
id	int	PK / AI		Identificador de publicación
titulo	varchar		255	Título de la publicación
contenido	text			Contenido que tendrá la publicación
id_usuario	int	INDEX (FK)		Llave foránea que relaciona la tabla usuarios
id_comunidad	int	INDEX (FK)		Llave foránea que relaciona la tabla comunidades
es_comunidad	tinyint		1	Booleano dependiendo si la publicación se hizo en una comunidad o no
imagen	varchar		255	Imagen que contiene la publicación
fecha_creacion	timestamp			Fecha de creación de la publicación

Tabla 3 Diccionario de datos modelo comunidades

COMUNIDADES				
Nombre Campo	Tipo dato	PK / INDEX(FK)	Tamaño	Descripción
id	int	PK / AI		Identificador de la comunidad
nombre	varchar		255	Nombre de la comunidad
descripcion	text			Descripción de la comunidad
id_creador	int	INDEX (FK)		Llave foránea que contiene el id del creador de la comunidad
imagen	varchar		255	Imagen de la comunidad
tipo_privacidad	enum		publica, privada, suspendido	Privacidad de comunidad
motivo	varchar		255	Motivo de la suspensión

duracion	int			Duración de la suspensión
fecha_fin_suspension	timestamp			Fecha de la suspensión
fecha_creacion	timestamp			Fecha de creación de la comunidad

Tabla 4 Diccionario de datos modelo usuarios_comunidades

USUARIOS COMUNIDADES				
Nombre Campo	Tipo dato	PK / INDEX (FK)	Tamaño	Descripción
id	int	PK / AI		Identificador de relación
id_usuario	int	INDEX (FK)		Llave foránea que relaciona la tabla usuarios
id_comunidad	int	INDEX (FK)		Llave foránea que relaciona la tabla comunidades
fecha_union	timestamp			Fecha de creación de la relación

Tabla 5 Diccionario de datos modelo reels

REELS				
Nombre Campo	Tipo dato	PK / INDEX(FK)	Tamaño	Descripción
id	int	PK / AI		Identificador del reel
usuario_id	int	INDEX (FK)		Llave foránea que relaciona la tabla usuarios
archivo_video	varchar		255	Video que contiene el reel
descripcion	text			Descripción del video
fecha_creacion	timestamp			Fecha de creación del reel

Tabla 6 Diccionario de datos modelo historias

HISTORIAS				
Nombre Campo	Tipo dato	PK / INDEX (FK)	Tamaño	Descripción
id	int	PK / AI		Identificador de historia
usuario_id	int	INDEX (FK)		Llave foránea que relaciona la tabla usuarios
contenido	text			Contenido que tendrá la historia
tipo	varchar		255	Tipo de contenido
fecha_creacion	timestamp			Fecha de creación de la historia
fecha_expiracion	timestamp			Fecha expiración de la historia

Tabla 7 Diccionario de datos modelo vistas_historias

VISTAS_HISTORIAS				
Nombre Campo	Tipo dato	PK / INDEX (FK)	Tamaño	Descripción
historia_id	int	INDEX (FK)		Llave foránea que relaciona la tabla historias
usuario_id	int	INDEX (FK)		Llave foránea que relaciona la tabla usuarios
fecha_vista	timestamp			Fecha de vista

Tabla 8 Diccionario de datos modelo reacciones

REACCIONES				
Nombre Campo	Tipo dato	PK / INDEX (FK)	Tamaño	Descripción
id	int	PK / AI		Identificador de reacción
tipo	varchar		50	Tipo de reacción
id_usuario	int	INDEX (FK)		Llave foránea que relaciona la tabla usuarios
id_contenido	int	INDEX (FK)		Llave foránea que relaciona contenido (publicación, etc)
tipo_contenido	enum		Publicación, reel, historia	tipo de contenido

fecha_creacion	timestamp			Fecha creación de la reacción

Tabla 9 Diccionario de datos modelo comentarios

COMENTARIOS				
Nombre Campo	Tipo dato	PK / INDEX(FK)	Tamaño	Descripción
id	int	PK / AI		Identificador de comentario
id_usuario	int	INDEX (FK)		Llave foránea que relaciona la tabla usuarios
id_contenido	int	FK		Llave foránea que relaciona contenido (publicación, etc)
tipo_contenido	enum		Publicación, reel, historia	tipo de contenido
contenido	text			Contenido del comentario
fecha_creacion	timestamp			Fecha creación del comentario

Tabla 10 Diccionario de datos modelo amistades

AMISTADES				
Nombre Campo	Tipo dato	PK / INDEX (FK)	Tamaño	Descripción
id	int	PK / AI		Identificador de amistades
id_usuario1	int	INDEX (FK)		Llave foránea que relaciona la tabla usuarios
id_usuario2	int	INDEX (FK)		Llave foránea que relaciona la tabla usuarios
estado	enum		Aceptada, rechazada, bloqueado, pendiente	Estado de la solicitud de amistad
fecha_creacion	timestamp			Fecha creación de la amistad
fecha_actualizacion	timestamp			Fecha actualización del estado de la amistad
estado_anterior	varchar		50	Estado anterior de la solicitud de amistad

Tabla 11 Diccionario de datos modelo mensajes

MESSAGES				
Nombre Campo	Tipo dato	PK / INDEX (FK)	Tamaño	Descripción
id	int	PK / AI		Identificador de mensaje
sender_id	int	INDEX (FK)		Llave foránea que relaciona la tabla usuarios
receiver_id	int	INDEX (FK)		Llave foránea que relaciona la tabla usuarios
message	text			Contenido del mensaje
audio_path	varchar		255	Ruta audio
duration	varchar		10	Duración del audio
read_status	tinyint		1	Estado de leido
created_at	timestamp			Fecha creación del mensaje

Tabla 12 Diccionario de datos modelo notificaciones

NOTIFICACIONES				
Nombre Campo	Tipo dato	PK / INDEX (FK)	Tamaño	Descripción
id	int	PK / AI		Identificador de la notificacion
usuario_id	int	INDEX (FK)		Llave foránea que relaciona la tabla usuarios
tipo	varchar		50	Tipo de notificacion
contenido	text			Contenido de la notificacion
leida	tinyint		1	Estado de la notificación si esta leida o no
referencia_id	int	INDEX (FK)		Llave foránea que relaciona con la tabla amistades
fecha_creacion	timestamp			Fecha creación del grupo

Tabla 13 Diccionario de datos modelo respuestas_comentarios_publicaciones

RESPUESTAS_COMENTARIOS_PUBLICACIONES				
Nombre Campo	Tipo dato	PK / INDEX (FK)	Tamaño	Descripción
id	int	PK / AI		Identificador de la respuesta
id_usuario	int	INDEX (FK)		Llave foránea que relaciona la tabla usuarios
id_comentario	int	INDEX (FK)		Llave foránea que relaciona la tabla comentarios
contenido	text			Contenido de la respuesta
fecha_creacion	timestamp			Fecha creación del grupo

Tabla 14 Diccionario de datos modelo respuestas_comentarios_reels

RESPUESTAS_COMENTARIOS_REELS				
Nombre Campo	Tipo dato	PK / INDEX (FK)	Tamaño	Descripción
id	int	PK / AI		Identificador de la respuesta
id_usuario	int	INDEX (FK)		Llave foránea que relaciona la tabla usuarios
id_comentario	int	INDEX (FK)		Llave foránea que relaciona la tabla comentarios
contenido	text			Contenido de la respuesta
fecha_creacion	timestamp			Fecha creación del grupo

Tabla 15 Diccionario de datos modelo backups

BACKUPS				
Nombre Campo	Tipo dato	PK / INDEX (FK)	Tamaño	Descripción
id	int	PK / AI		Identificador del backup
filename	varchar		255	Nombre del archivo
created_at	datetime			Fecha de creación
created_by	int			Id usuario quien lo creo

Tabla 16 Diccionario de datos modelo backups_restores

BACKUPS_RESTORES				
Nombre Campo	Tipo dato	PK / INDEX (FK)	Tamaño	Descripción
id	int	PK / AI		Identificador de la restauración
backup_filename	varchar		255	Nombre del archivo
restored_at	datetime			Fecha de restauración
restored_by	int			Id usuario quien lo restauro

Tabla 17 Diccionario de datos modelo reportes

REPORTES				
Nombre Campo	Tipo dato	PK / FK	Tamaño	Descripción
id	int	PK / AI		Identificador de reporte
tipo_reporte	enum		publicacion, reel, comentario, comunidad, usuario, contenido	Tipo de reporte
id_contenido	int	INDEX (FK)		Llave foránea que relaciona las tablas de contenido
id_usuario_reportante	int	INDEX (FK)		Llave foránea que relaciona la tabla usuarios
motivo	varchar		255	Motivo del reporte
descripcion	text			Descripción del reporte
estado	enum		pendiente, revisado, resuelto, rechazado	Estado del reporte
fecha_reporte	timestamp			Fecha de creación del reporte
fecha_resolucion	timestamp			Fecha de solución
id_admin_resolucion	int	INDEX (FK)		Llave foránea que relaciona la tabla usuarios
accion_tomada	varchar		255	Acción tomada contra la persona infractora

Tabla 18 Diccionario de datos modelo intereses

INTERESES				
Nombre Campo	Tipo dato	PK / INDEX (FK)	Tamaño	Descripción
id	int	PK / AI		Identificador del interes
user_id	int	INDEX (FK)		Llave foránea que relaciona la tabla usuarios
interes	varchar		255	Descripción del interes
created_at	timestamp			Fecha de creación

Tabla 19 Diccionario de datos modelo configuraciones_usuario

CONFIGURACIONES_USUARIO				
Nombre Campo	Tipo dato	PK / FK	Tamaño	Descripción
id	int	PK / AI		Identificador de las configuraciones
usuario_id	int	INDEX (FK)		Llave foránea que relaciona la tabla usuarios
privacidad	varchar		20	Descripcion del tipo de privacidad del perfil
notificaciones	tinyint		1	Booleando para ver si las notificaciones están activas o no
audio_enable	tinyint		1	Booleano para ver si el usuario tiene permiso en la app de grabar audio
files_access_enable	tinyint		1	Booleano para ver si el usuario tiene permisos a la app para acceder a sus archivos
online_visibility	tinyint		1	Booleano para ver si el usuario tiene permitido que otros usuarios lo vean en línea o no
idioma	varchar		5	Idioma de la app para el usuario
autoplay_videos	tinyint		1	Si el usuario tiene permitido que los videos se reproduzcan automáticamente
fecha_creacion	timestamp			Fecha de creación
fecha_actualización	timestamp			Fecha actualización

ESTRUCTURA DE ARCHIVOS Y CARPETAS

Para mantener un desarrollo organizado y escalable, Bubblebox sigue una estructura de archivos clara tanto en el backend como en el frontend. A continuación, se describe la distribución de carpetas y su propósito en el sistema.

FRONTEND (REACTJS)

El frontend está basado en React y sigue una estructura organizada dentro de la carpeta src/.

Link:https://drive.google.com/file/d/1byIgCXANQxTzjHsEhTZeVUwtCDRFCd97/view?usp=drive_link

La organización modular facilita la reutilización de código y mantiene una separación clara entre lógica y presentación.

BACKEND (NODEJS, EXPRESS, MYSQL)

El backend está basado en una arquitectura de microservicios, donde cada servicio está encapsulado en su propia carpeta dentro de services/.

Link: https://drive.google.com/file/d/1B2aA-OEpfuF6_NB1OFpmuYqTd1P1Y7Eq/view?usp=drive_link

Cada microservicio se encarga de una funcionalidad específica, lo que facilita la escalabilidad y el mantenimiento del sistema.

DASHBOARD (PHP)

El panel de administración está desarrollado en PHP y organiza sus vistas dentro de views/.

Link:https://drive.google.com/file/d/1SXufjjwiW7_Crxn8Y7LwhnwIozsdXWJj/view?usp=drive_link

El panel de administración permite gestionar los datos de la red social, incluyendo usuarios, publicaciones y reportes.

SEGURIDAD Y CONTROL DE ACCESO

La seguridad del sistema es una prioridad para garantizar la integridad de los datos, la privacidad de los usuarios y la protección contra accesos no autorizados. A continuación, se detallan las medidas implementadas para reforzar la seguridad y el control de acceso en la plataforma.

1. Autenticación y Autorización

1.1. Autenticación de Usuarios

El sistema utiliza un mecanismo seguro para autenticar a los usuarios:

- **JSON Web Tokens (JWT)**: Se generan tokens firmados para autenticar a los usuarios en el backend.
- **Expiración de tokens**: Los tokens tienen un tiempo de vida.
- **Hashing de Contraseñas**: Se utiliza **bcrypt** para almacenar contraseñas de forma segura.

1.2. Roles y Permisos

Cada usuario tiene un **rol** asignado que define su nivel de acceso:

- **Administrador**: Acceso completo al sistema, gestión de usuarios y contenido.
- **Usuario estándar**: Acceso a funcionalidades básicas como publicar, comentar y reaccionar.

El backend valida los permisos antes de procesar cualquier solicitud sensible.

2. Protección contra Ataques

2.1. Prevención de Inyección SQL

- Uso de **consultas preparadas** con mysql2 en Node.js.
- Validación y saneamiento de datos en todas las entradas del usuario.

2.2. Protección contra XSS (Cross-Site Scripting)

- Se escapan caracteres especiales en las respuestas del backend.
- Uso de **Content Security Policy (CSP)** en el frontend para restringir fuentes de contenido externo.

2.3. Protección contra CSRF (Cross-Site Request Forgery)

- Implementación de **tokens CSRF** en el panel de administración y formularios críticos.

2.4. Protección contra Ataques de Fuerza Bruta

- **Limitación de intentos de login** con express-rate-limit.

2.5. Protección contra Robo de Sesiones

- **Regeneración de tokens JWT** en cada inicio de sesión.

3. Seguridad en la Comunicación

- **Cifrado de datos sensibles** en la base de datos (tokens, correos, etc.).

4. Seguridad en la Infraestructura

- **Control de acceso a servidores** mediante SSH con autenticación por clave.
- **Backups automáticos** con cifrado para recuperación ante incidentes.

Estas medidas garantizan un sistema robusto contra amenazas y ataques, protegiendo la integridad y privacidad de los usuarios.

MANTENIMIENTO Y SOPORTE

El mantenimiento y soporte del sistema son esenciales para garantizar su estabilidad, seguridad y rendimiento óptimo. A continuación, se describen las estrategias implementadas para el mantenimiento proactivo y correctivo de la plataforma.

1. Mantenimiento Preventivo

1.1. Actualización de Dependencias

- Monitoreo continuo de **actualizaciones de paquetes** en Node.js, React, PHP y otros componentes.
- Uso de herramientas como npm audit para identificar vulnerabilidades.
- Aplicación de **parches de seguridad** en cuanto estén disponibles.

1.2. Optimización de Base de Datos

- **Indexación** de tablas para mejorar la velocidad de las consultas.
- **Archivar o eliminar registros antiguos** para reducir la carga de datos.
- **Backups automáticos** diarios con almacenamiento en un servidor seguro.

2. Mantenimiento Correctivo

2.1. Detección y Corrección de Errores

- Uso de **logs centralizados** para identificar fallos en tiempo real.
- **Monitoreo de errores** con herramientas como Sentry o LogRocket.
- Implementación de pruebas unitarias y de integración antes de cada despliegue.

2.2. Gestión de Incidentes

- **Canal de comunicación** para reportar problemas de los usuarios.
- Procedimientos de escalamiento de errores según prioridad.
- **Registro de incidentes** para análisis y prevención de problemas recurrentes.

3. Soporte Técnico

3.1. Documentación

- Guías para instalación y configuración del sistema.
- Manual de usuario para el Dashboard y las funcionalidades principales.
- Documentación interna para el equipo de desarrollo.

3.2. Canales de Soporte

- Sistema de **tickets** para reportar errores y consultas.
- **Foro de la comunidad** para preguntas frecuentes y soluciones colaborativas.
- **Soporte directo** por correo o chat para clientes premium.

4. Monitoreo y Rendimiento

4.1. Análisis de Rendimiento

- Uso de **PM2** para monitorear el backend en producción.
- Optimización del tiempo de carga en el frontend con lazy loading y code splitting.
- **Pruebas de estrés** periódicas para evaluar la escalabilidad.

4.2. Detección de Anomalías

- Monitoreo de **tiempo de respuesta del servidor** con herramientas como New Relic.
- Alertas automáticas en caso de consumo elevado de recursos.
- Evaluaciones de seguridad para identificar vulnerabilidades antes de un ataque.

5. Plan de Recuperación ante Desastres

- **Backups diarios** almacenados en múltiples ubicaciones seguras.
- Procedimiento para **restauración rápida** en caso de fallos críticos.
- **Planes de contingencia** para mitigar impacto en los usuarios.

6. Seguridad en el Mantenimiento

- **Despliegues seguros** en entornos de prueba antes de pasar a producción.
- Restricción de permisos en tareas de mantenimiento crítico.
- Uso de **entornos de staging** para validar cambios sin afectar a los usuarios.

BIBLIOGRAFIA

Creadores. (s.f.). *Casos de uso.* Obtenido de https://drive.google.com/file/d/10b6DKxRV12o4wqP_lQp_k--eQbdJtdS7/view?usp=drive_link.

Creadores. (s.f.). *Diagrama DB.* Obtenido de https://drive.google.com/file/d/1RXXntxyeuKzXWeMktxHQJVwQXaYUbver/view?usp=drive_link.

Creadores. (s.f.). *Diagrama de componentes.* Obtenido de https://drive.google.com/file/d/1S52UBY8Yr2FuuKzIEAbPH5m1NBQ8jvEi/view?usp=drive_link.

Creadores. (s.f.). *Diagrama interfaces.* Obtenido de https://drive.google.com/file/d/1YQTrBrEzbtQW-W0a0PR1XiAPsPe8McE4/view?usp=drive_link.

Creadores. (s.f.). *Diagrama microservicios.* Obtenido de https://drive.google.com/file/d/1QnU2LZs0YijjRaTClVl4DqN0lqykDwMt/view?usp=drive_link.

Creadores. (s.f.). *Diagrama secuencia.* Obtenido de https://drive.google.com/file/d/1-Oq0EDLlFJ9ZGV083HGh-mNAjyNsdlZt/view?usp=drive_link.

Creadores. (s.f.). *Script esquema DB.* Obtenido de https://drive.google.com/file/d/12X7PM0FIWDqEgRpScavU5rB7o-r5xcX_/view?usp=drive_link.

documentation, G. (s.f.). *GitHub*. Obtenido de https://docs.github.com.

Foundation, O. (s.f.). *Node.js*. Obtenido de https://nodejs.org/es.

Microsoft. (s.f.). *Visual Studio Code*. Obtenido de https://code.visualstudio.com.