DELVALLE
GRUPO EDUCATIVO

Astrid Glauser
Samuel Argueta
Donald Cuellar
Alejandro Martínez

Proyecto 2

Cifrados de Información

Informe de Desarrollo y Errores Encontrados

Proyecto 2 - Cifrados Seguros

Curso: Cifrado de Información

Objetivo General: Diseñar e implementar un sistema de comunicaciones seguras, integrando mecanismos modernos de autenticación, cifrado, firma digital, integridad de datos y almacenamiento seguro mediante blockchain. La aplicación permite funcionalidades de chat grupal y P2P, aseguradas mediante criptografía avanzada.

Integrantes del Proyecto

Nombre	Carné	Rol
Samuel Argueta	211024	Frontend + Autenticación
Alejandro Martínez	21430	Backend + Chat & Crypt
Astrid Glauser	21299	Backend + Frontend Chat
Dolan Raúl	21965	Backend + Blockchain

Funcionalidades Implementadas

Autenticación Segura

- **OAuth 2.0 con Google:** Inicio de sesión mediante cuenta Google con redirección segura y autorización de cliente.
- **TOTP (Two-Factor Authentication):** Generación de códigos temporales asociados a cada usuario para verificación en segundo paso.
- **JWT + Refresh Tokens:** Control seguro de sesiones con expiración, renovación y verificación de tokens.

Chats Protegidos

- **Chat Grupal:** Encriptación de mensajes con **AES-256-GCM** compartida entre miembros del grupo.

- **Chat P2P:** Cifrado de extremo a extremo mediante **X3DH** (Intercambio de claves) y **Double Ratchet** (actualización continua de claves).

Firma Digital y Hashing

- **Firmas con ECDSA:** Cada mensaje es firmado con la clave privada del usuario emisor.

- **Verificación de integridad:** Se aplica SHA-256 o SHA-3 sobre el mensaje original para validar que no fue alterado.

Mini Blockchain

- Implementación de estructura simple de bloques con hash encadenado.

- Cada bloque contiene: `nombre`, `fecha_envio`, `data_enviada`, `hash_anterior`, `hash_actual`.

- Los bloques son inmutables y se agregan de forma secuencial.

Estructura del Proyecto

proyecto-cifrados/
├── backend/ # API en FastAPI con toda la lógica criptográfica y autenticación
├── frontend/ # Aplicación web desarrollada en React (Vite)

Tecnologías Utilizadas

Backend (FastAPI)

- **FastAPI**

- **OAuthlib / Authlib**

- **PyOTP**

- **PyJWT**
- **cryptography** (SHA-256, SHA-3, ECDSA, AES)
- **PyNaCl** (X3DH + Double Ratchet)
- **PostgreSQL + SQLAlchemy**
- **Docker** (para base de datos)

Frontend (React + Vite)

- **React + Vite**
- **react-router-dom**
- **CSS Modules**
- **Hooks personalizados (useAuth)**
- **Google OAuth 2.0 Integration**

Instalación y Ejecución Local

Requisitos:

- Node.js 18+
- Python 3.11+
- Docker + Docker Compose

Backend (FastAPI)

```
cd backend
python -m venv env
source env/bin/activate
pip install -r requirements.txt
uvicorn app.main:app --reload
```

Frontend (React)

cd frontend
npm install
npm run dev

Docker

docker-compose up --build

Variables de Entorno (.env ejemplo)

POSTGRES_USER=admin
POSTGRES_PASSWORD=securepass
POSTGRES_DB=cifrados_db
DATABASE_URL=postgresql://admin:securepass@localhost:5432/cifrados_db

SECRET_KEY=supersecretkey
SESSION_SECRET_KEY=sessionkeyhere

GOOGLE_CLIENT_ID=XXX.apps.googleusercontent.com
GOOGLE_CLIENT_SECRET=XXX
GOOGLE_REDIRECT_URI=http://localhost:3000/oauth/callback

Errores Encontrados y Soluciones

Autenticación

Problema: Error 401 al cargar contactos tras login.
Solución: El token JWT no se estaba enviando correctamente en el `Authorization header`. Se verificó su existencia antes de la llamada.

Problema: No se ejecutaba el flujo TOTP para usuarios nuevos.
Solución: Se integró correctamente el flujo de registro de secreto TOTP en la primera sesión.

Cifrado y Firma

Problema: La firma ECDSA generada no era verificable por el backend.
Solución: Se ajustó el formato de la firma en DER vs RAW según correspondía. En ECC se usó `asn1.js` y `elliptic` para parsing.

Problema: Los mensajes se mostraban al revés en la interfaz.
Solución: Se invirtió el orden en el renderizado de mensajes en el frontend para preservar orden cronológico ascendente.

Blockchain

Problema: Hash del bloque no coincidía después de una edición.
Solución: Se recalculó el hash actual con todos los campos relevantes y se bloqueó la edición directa.

Pruebas Realizadas

- Flujo de registro y autenticación.
- Verificación de firmas digitales.
- Cifrado P2P con mensajes entre dos usuarios distintos.
- Validación de integridad mediante hashes SHA.
- Confirmación de bloques inmutables en el blockchain.

Recomendaciones

- Mejorar la rotación automática de claves en el protocolo P2P.
- Automatizar pruebas de regresión sobre el backend con pytest y Postman.
- Considerar cifrado homomórfico para mensajes almacenados (investigación futura).

Conclusión

El sistema implementado cumple con los requerimientos principales del proyecto, integrando múltiples capas de seguridad en autenticación, cifrado, firma e integridad. El uso de blockchain para garantizar la inmutabilidad refuerza la seguridad general de la aplicación. Todos los módulos funcionan y se integraron de forma colaborativa.

Todos los detalles adicionales se encuentran en los archivos README de backend y frontend.