Universidad
Carlos III de Madrid

Criptografía y seguridad informática

G24 - Reporte 1

Javier Martín Pizarro

Alberto Pascau Sáez

Raúl Armas Seriña

Índice general

0.1. Propósito de la aplicación. Estructura interna

0.1.1. Propósito de la aplicación

La aplicación simula una página web, levantada en local por el propio usuario, en la que se pueden crear una serie de desafíos (*challenges*) tanto privados como públicos. Dependiendo de qué tipo de sea cada desafío, el flujo interno (de código) será distinto.

- **Desafío público:** cualquier usuario registrado tiene acceso a ellos.
- **Desafío privado:** solamente pueden ser compartidos con un único usuario. El creador del desafío selecciona al otro usuario que será capaz de verlo.

El propósito de esta aplicación es generar un sistema informático que cumpla unos requisitos mínimos. Nótese que a medida que la práctica avance, esta lista podrá verse modificada.:

1. El sistema debe de ser capaz de registrar usuarios y que su información confidencial quede correctamente cifrada.
2. El sistema debe ser capaz de permitir un inicio de sesión fluido donde se sea capaz de obtener y comparar los datos cifrados de los usuarios de la base de datos con los proporcionados por el cliente.
3. El sistema debe de ser capaz de crear y recuperar desafíos cuya información pueda o no estar cifrada.
4. El sistema debe de ser capaz de permitir a un usuario A leer el desafío creado por el usuario B si este se lo ha compartido.

0.1.2. Estructura interna

Esta aplicación consta de tres partes fundamentales:

- Frontend: esencial para la experiencia de usuario. Actua como interfaz entre el cliente y el backend.
- Backend: donde se encuentra la API, hecha en Flask. Todos los mecanismos de cifrado, autenticación, *hasheo* se encuentran en este directorio.
- Base de datos: creada usando MariaDB SQL debido a su simplicidad. La base de datos tiene únicamente dos tablas, *users* y *challenges*.

La base del backend y de la base de datos han sido tomadas desde el repositorio Open Source **backend-builderplate** del alumno y participante en esta práctica Javier Martín, una iniciativa que permite agilizar y automatizar el proceso de levantar una API y una base de datos que complemente a una interfaz de usuario. El código de este proyecto hereda del repositorio original.[1]

```
.
backend
    app.py
    Dockerfile
    requirements.txt
    src
        mariaDB
            connection.py
```

[1] Repositorio original: `https://github.com/jmartinpizarro/backend-builderplate`.

```
            query_users.py
        UserManager.py
        utils
            HashManager.py
db_data
docker-compose.yml
frontend
    CSS
        styles.css
    HTML
        challengecreate.html
        challenges.html
        login.html
        newuser.html
        settings.html
    JavaScript
        challengesr.js
        loginsr.js
        newusersr.js
        settingssr.js
    python_server.py
init.sql
memory
    main.pdf
    main.tex
    uc3m.jpg
README.md
```

El código anterior enseña en forma de árbol la estructura del proyecto. Las funciones de cifrado, *hasheo* y demás se pueden encontrar en la carpeta de **utils**. Las rutas de la API se encuentran dentro de la carpeta **api**.

Nótese que la carpeta **db_data** es generada automáticamente a la hora de levantar el proyecto. Se necesitan permisos de administrador para eliminarla, ya que ahí se encuentran los datos de la propia base de datos (existe permanencia de datos incluso si cerramos el contenedor de Docker). Para levantar su proyecto en local, recomendamos que se lea las instrucciones del archivo *README.md*.

0.2. Autentiación de usuarios. Algoritmia

0.3. Explicación del cifrado usado y su algoritmia

0.4. Autenticación usando MAC

0.5. Anexo

Este manuscrito y su correspondiente práctica ha sido realizado por:

- Javier Martín Pizarro, 100495861@alumnos.uc3m.es
- Alberto Pascau Núñez, 100xxxxxx@alumnos.uc3m.es
- Raúl Armas Seriña, 100xxxxxx@alumnos.uc3m.es