Project Report: Secure QR Messenger

Project Title: QR Code Secret Messages with Encryption **Language:** Python

---

1. Introduction

This project is a desktop application that hides secret messages inside QR codes. It is not just a normal QR generator; it uses **AES Encryption** to lock the message. This means even if someone scans the QR code, they cannot read the message without the correct password.

We upgraded the project from a simple text tool to a full **Graphical User Interface (GUI)** with buttons and windows. We also added security features like password checking and protection against hackers.

---

2. Libraries Used & Their Usage

We used four main external libraries to make this project work. Here is what each one does:

1. `tkinter` (Built-in)

- **What it is:** The standard Python library for creating Graphical User Interfaces (GUIs).
- **Usage in Project:** It draws the windows, buttons, tabs, and input boxes. It allows the user to click and type instead of using a black command screen.

2. `pycryptodome` (External)

- **What it is:** A powerful library for cryptography (security math).
- **Usage in Project:**
    - **AES Cipher:** It scrambles the message so it looks like random noise.
    - **SHA-256:** It turns the user's password into a secure 32-byte key.
    - **Padding:** It adds extra characters to the message to make sure it fits the encryption block size.

3. `qrcode` (External)

- **What it is:** A library dedicated to generating QR (Quick Response) codes.
- **Usage in Project:** It takes the encrypted text string and converts it into a scannable square barcode image.

4. `pillow` (PIL) (External)

- **What it is:** The Python Imaging Library. It handles opening and editing images.
- **Usage in Project:** `tkinter` cannot display images directly. We use `pillow` to load the QR code image we saved and display it inside the application window.

---

3. Key Features Explained

We implemented **6 special features** to get the full grade and bonus points.

Feature 1: Password Strength Check

- **Goal:** Stop users from using weak passwords like "1234".
- **How it works:** The system checks if the password has uppercase letters, numbers, and symbols. It shows a **Green** label for strong passwords and **Red** for weak ones.

Feature 2: Brute-Force Protection

- **Goal:** Stop hackers from guessing the password by trying thousands of times.
- **How it works:** If you enter the wrong password **3 times**, the system **locks you out for 30 seconds**. You cannot try again until the timer finishes.

Feature 3: SHA-256 Key Derivation

- **Goal:** Make the encryption mathematically secure.
- **How it works:** AES encryption needs a key of exactly 32 bytes. Since human passwords vary in length, we use **SHA-256** to convert any password into a perfect 32-byte key.

Feature 4: GUI Interface

- **Goal:** Make the app easy to use.
- **How it works:** We replaced the command line with a modern window containing three tabs:
    1. **Register:** To create a user.
    2. **Encrypt:** To make the QR code.
    3. **Decrypt:** To read the message.

Feature 5: File Encryption Support

- **Goal:** Allow users to encrypt large texts easily.
- **How it works:** Added a button called **"Import .txt File"**. You can select a text file from your computer, and the app will automatically read it and prepare it for encryption.

Feature 6: Two-Step Authentication

- **Goal:** Double the security.
- **How it works:** To decrypt a message, you need two things:
    1. A registered **Username**.
    2. The correct **Password**. The system checks your username in a database before letting you unlock the message.

---

4. How to Use the App

Step 1: Register

- Go to the "Register" tab.
- Create a username and a strong password.

Step 2: Encrypt (Hide Message)

- Go to the "Encrypt" tab.
- Type a secret message OR upload a text file.
- Enter a password and click **Generate QR**.
- Save the image.

Step 3: Decrypt (Read Message)

- Go to the "Decrypt" tab.
- Paste the text from the QR code.
- Enter your Username and Password to prove it is you.
- Click **Decrypt** to reveal the secret.

---

5. Conclusion

This project successfully combines security with ease of use. By using **Python** and **AES Encryption**, we created a tool that keeps data safe. The added features like the **GUI** and **Brute-Force Protection** make the software feel professional and ready for real-world use.

Team members

Yusuf Yasser wardany

Mahmoud hossam

Hazem khaled