AUTOMATED LOGGING SYSTEM USING QR CODES (CSE-332)

1. Objectives:

The primary objective of this project is to design and implement a **multi-factor student verification system** that combines **QR code authentication**, **OTP verification**, **AES encryption**, and **facial recognition** to ensure secure, reliable, and efficient student identification.

This system is intended to replace traditional manual verification methods, which are time-consuming and vulnerable to impersonation or data breaches. By integrating cryptographic encryption with biometric and OTP-based authentication, the project ensures a robust and layered security framework.

The specific goals of this project are:

- Develop a **unique QR-based identity mechanism** for each student.
- Implement **AES encryption** to securely store and retrieve sensitive data from the database.
- Integrate a **multi-stage verification pipeline**:
    1. **QR Code Scan:** Identify the student and retrieve the encrypted record.
    2. **OTP Verification:** Send a one-time password (OTP) to the registered contact for secondary verification and fetch detailed student information.
    3. **Facial Recognition:** Confirm identity using cosine similarity between live and stored facial embeddings.

- Ensure the system can **confirm or reject authentication** at each stage (QR → OTP → Face).

2. System Design and Architecture:

2.1 Overview

The QR Code-Based Student Verification System follows a **multi-factor authentication architecture** that combines three layers of security:

1. **QR-based identification**
2. **OTP-based secondary verification**
3. **Facial recognition-based biometric verification**

Each layer progressively strengthens trust in the authentication process. The system relies on **AES encryption** to secure all stored data and ensures that sensitive student information remains confidential even in database storage.

2.2 Security Layers

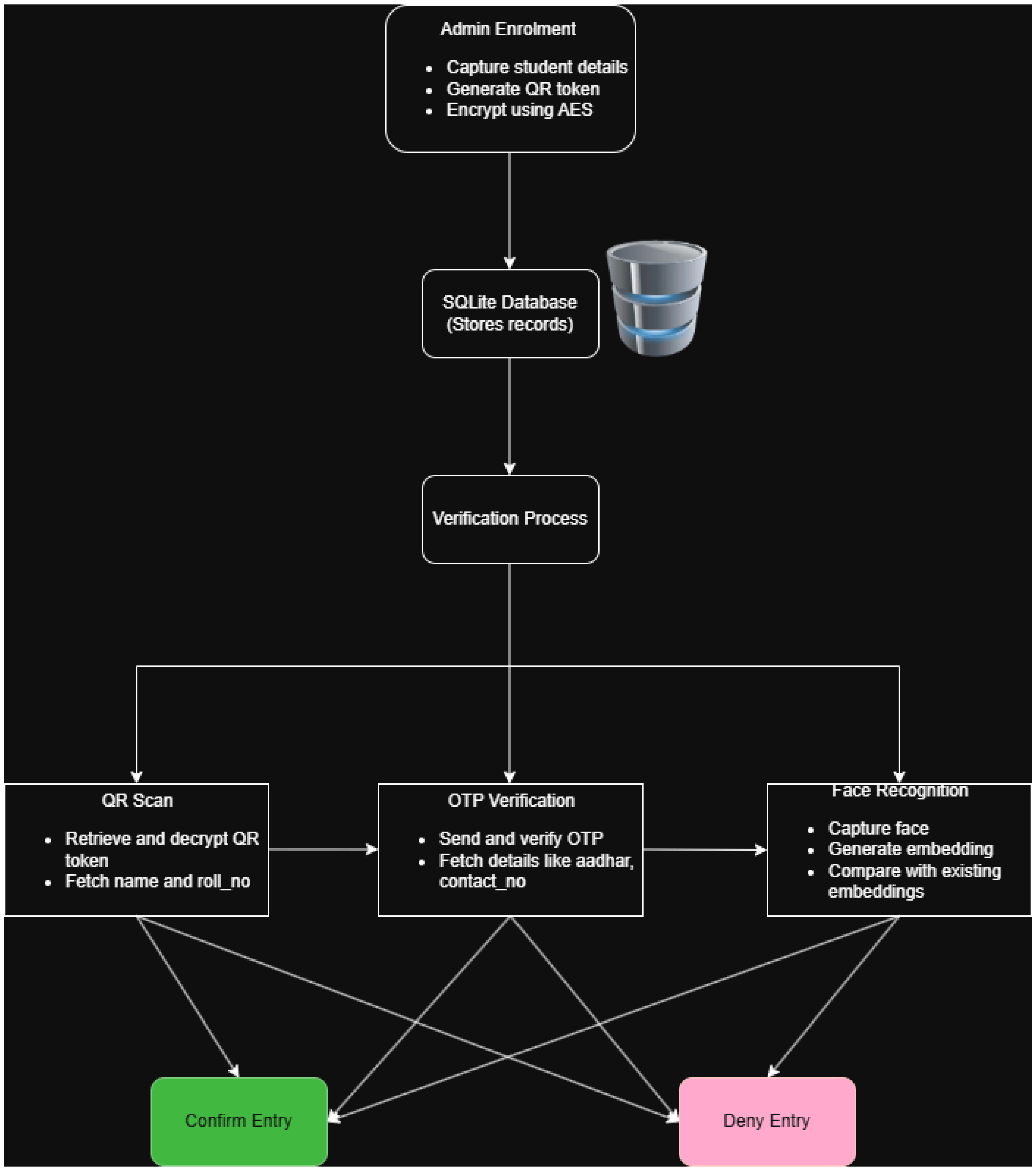
AES Encryption:
All stored data (personal info and facial embeddings) is encrypted using AES-GCM, ensuring confidentiality and integrity.

QR Tokenization:
Each student's identity is referenced through a random, non-reversible token rather than plain identifiers.

OTP Verification:
Acts as a secondary authentication layer to confirm possession of a registered communication channel.

Biometric Verification:
Provides a final and definitive identity check through facial embeddings and cosine similarity matching.

2.3 Architectural Diagram



3. Implementation Details:

3.1 Overview and Tech Stack used

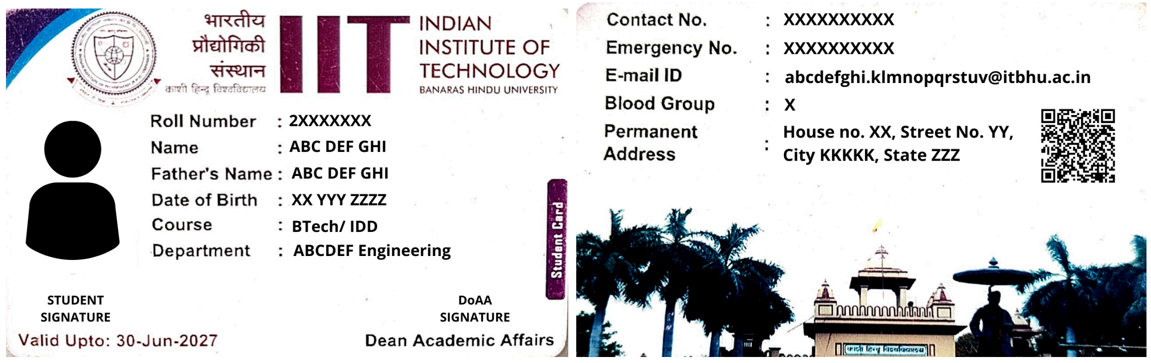
The implementation of the **QR Code-Based Student Verification System** integrates cryptography, database management, and computer vision techniques into a unified workflow. The system is implemented in **Python**, leveraging libraries such as **OpenCV** for face recognition, **cryptography** for AES encryption, and **SQLite3** for secure data storage.

3.1.1 Encryption

We use AES-GCM (Advanced Encryption Standard Galois Counter Mode) as our encryption algorithm. First, the **secrets** library is used to assign a random 128-bit qr_token to each student, which will serve as the PK for all future purposes. This qr_token is then encrypted using AESGCM with a 256-bit key which is generated at the beginning and is to be kept safe throughout the use of this system.

3.1.2 QR Code generation

We use the **qrcode** library to generate a QR code for the encrypted qr_token, which will be embedded into the ID cards of the students as follows:



भारतीय प्रौद्योगिकी संस्थान
काशी हिन्दू विश्वविद्यालय
IIT INDIAN INSTITUTE OF TECHNOLOGY
BANARAS HINDU UNIVERSITY

Roll Number	: 2XXXXXXX
Name	: ABC DEF GHI
Father's Name	: ABC DEF GHI
Date of Birth	: XX YYY ZZZZ
Course	: BTech/ IDD
Department	: ABCDEF Engineering

Student Card

STUDENT SIGNATURE
Valid Upto: 30-Jun-2027

DoAA SIGNATURE
Dean Academic Affairs

Contact No.	: XXXXXXXXXX
Emergency No.	: XXXXXXXXXX
E-mail ID	: abcdefghi.klmnopqrstuv@itbhu.ac.in
Blood Group	: X
Permanent Address	: House no. XX, Street No. YY, City KKKKK, State ZZZ

3.1.3 QR Code Scanning

We use OpenCV to capture the QR code and **pyzbar**'s "decode" functionality to extract the encrypted qr_token from the QR code, which is then decrypted using the key. This fetches the basic details such as name and roll_no for the proctor on the app.

3.1.4 OTP Verification

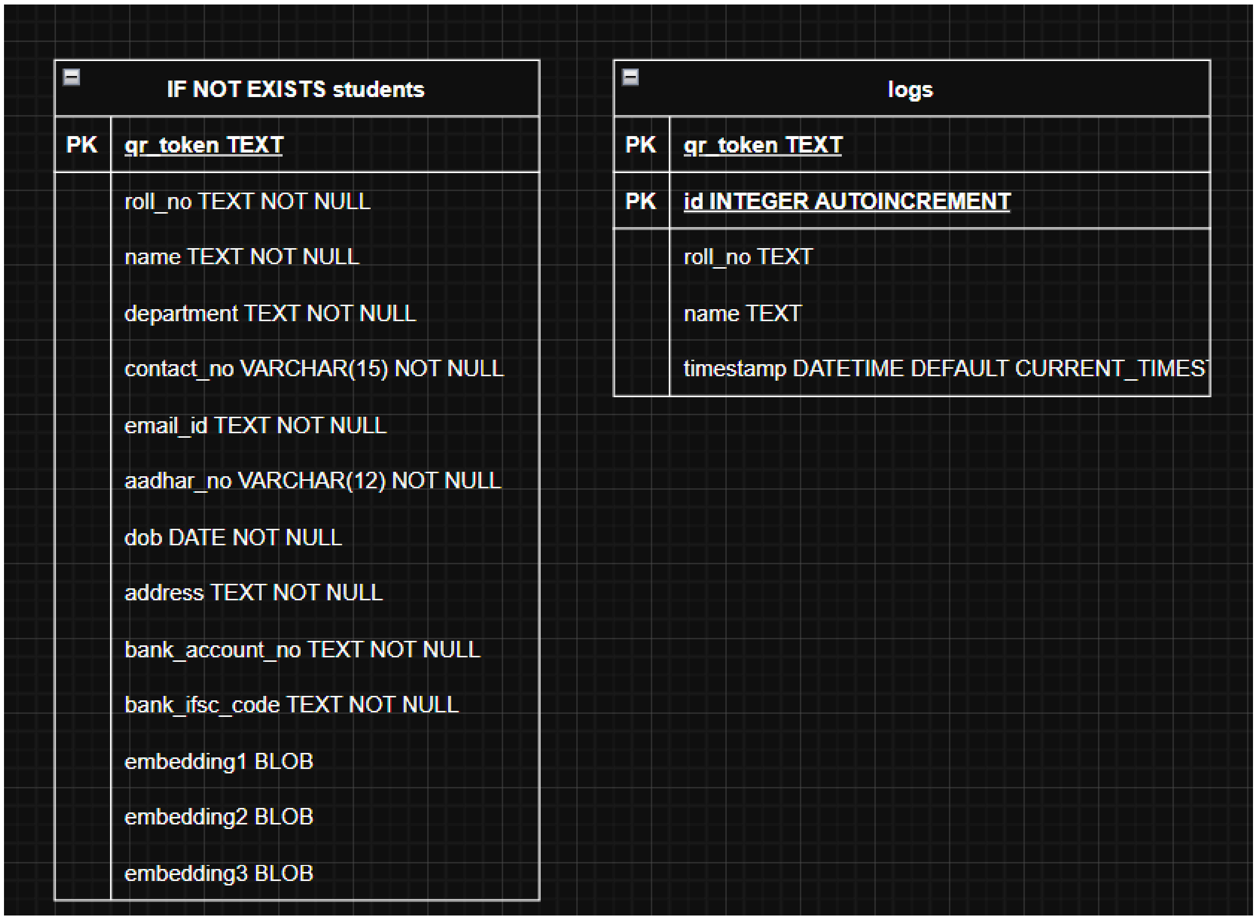
We make use of **Twilio's SMS API** to handle OTPs. The proctor can send an OTP to the student's phone, which on verifying will fetch further details like the student's aadhar_no, address etc.

3.1.5 Face Scanning

This is to confirm the student's identity biometrically in case of doubts even after OTP verification. OpenCV is used to capture a live image of the student. We use the open source **AuraFace** model from **Hugginface** to generate and store 3 embeddings for each student during admin entry. At inference, a live embedding is generated and we calculate the average cosine similarity against the 3 stored embeddings. If the average is >=0.5, it's considered valid.

$$\text{similarity} = \frac{A \cdot B}{\|A\| \|B\|}$$

3.1.6 DB Schema



	IF NOT EXISTS students
PK	qr_token TEXT
	roll_no TEXT NOT NULL
	name TEXT NOT NULL
	department TEXT NOT NULL
	contact_no VARCHAR(15) NOT NULL
	email_id TEXT NOT NULL
	aadhar_no VARCHAR(12) NOT NULL
	dob DATE NOT NULL
	address TEXT NOT NULL
	bank_account_no TEXT NOT NULL
	bank_ifsc_code TEXT NOT NULL
	embedding1 BLOB
	embedding2 BLOB
	embedding3 BLOB

	logs
PK	qr_token TEXT
PK	id INTEGER AUTOINCREMENT
	roll_no TEXT
	name TEXT
	timestamp DATETIME DEFAULT CURRENT_TIMES

The detailed instructions on how to set up the project locally, along with the source code, can be found at : https://github.com/dhairya-1105/qr_code_cso332