Problem Statement 11

QR-to-Database Real-Time Interaction System

Reference: QR itself is ISO/IEC 18004; for secure QR-based auth/check-ins use IETF: JWT (RFC 7519), CWT (RFC 8392), OAuth 2.0 (RFC 6749); and W3C: WebAuthn, Verifiable Credentials, and Payment Request API.

Background

QR codes are widely used for authentication, attendance, payments, and event check-ins. However, most solutions require manual data entry or multiple steps (scan → open link → upload). For seamless user experiences in education, events, or enterprise workflows, a **direct QR capture-to-database pipeline** is needed.

This hackathon challenge focuses on building a **two-way interactive system**:

- A **user-facing web interface** displays a QR code.
- The **user's mobile device** scans the QR code, automatically opens a secure link, and captures/returns data.
- The **backend database** receives the decoded QR data in real time.
- The **web interface updates dynamically** without refresh.

Problem Statement

Develop a **real-time QR interaction platform** where:

1. A user scans a QR code displayed on a webpage using their phone.
2. The phone opens a link, captures the QR content, and securely submits it.
3. The server decodes and inserts the data directly into a database.
4. The webpage auto-refreshes or dynamically updates to show the decoded data instantly.

Key Requirements

- **Front-end:**
  - Generate unique QR codes tied to session/user ID.
  - Webpage updates dynamically (WebSocket / AJAX / SSE) when new data arrives.
- **Mobile Workflow:**
  - Native camera or web-based QR scanner (using JS libraries or mobile intents).
  - Automatic submission of decoded QR payload to the backend link.
- **Backend:**
  - Secure API endpoint to accept scanned data.
  - Database integration (e.g., MySQL, PostgreSQL, MongoDB).
  - Validation logic (e.g., prevent duplicates, check QR authenticity).
- **Two-Way Process:**
  - Server → user: Display QR for scanning.
  - User → server: Capture & return decoded QR data.

Deliverables

- A working **demo system** with:
  - QR code generation on a webpage.
  - Mobile phone capture + submission workflow.
  - Real-time backend database update.
  - Webpage auto-refresh to show decoded data.
- Documentation of architecture, APIs, and database schema.