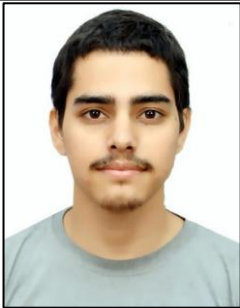





PROJECT AND TEAM INFORMATION

PROJECT TITLE

ADVANCED REAL TIME HYBRID ATTENDANCE MANGEMENT SYSTEM (ARTHAMS)

STUDENT / TEAM INFORMATION

TEAM NAME	TEAM ASTRA
TEAM MEMBER 1 (TEAM LEADER) NAME- PAWAN JOSHI STUDENT ID- 24021002 EMAIL- joshipawan2021@gmail.com	
TEAM MEMBER 2 NAME- ANUSHKA STUDENT ID- 24022656 EMAIL- goelanushka84@gmail.com	
TEAM MEMBER 3 NAME- JASMINE STUDENT ID- 24022556 EMAIL- jasminemahajan336@gmail.com	
TEAM MEMBER 4 NAME- AADITYA UNIYAL STUDENT ID- 240211539 EMAIL- aaditya.uniya122@gmail.com	

PROPOSAL DESCRIPTION

Motivation

Traditional attendance systems (manual registers, RFID swipes, or plain QR codes) are highly vulnerable to proxy and spoofing. Students can easily share QR screenshots, punch cards for friends, or even fake their location. This not only reduces fairness but also defeats the integrity of attendance-based evaluation.

*Our project proposes a **hybrid, proxy-proof attendance solution** that combines **dynamic QR codes, device binding, location/Wi-Fi validation**, and **face liveness checks** on mobile devices, with **fingerprint verification** as a fallback. This ensures multi-factor, real-time validation while maintaining usability.*

State of the Art / Current solution

- **QR-based attendance** is widely used but can be easily shared or screenshotted.
- **RFID/Biometrics-only systems** (fingerprint or iris) require physical hardware, making them less scalable and often leading to delays.
- **Geo-fencing or Wi-Fi-based attendance** exists but can be spoofed via VPNs or MAC spoofing. No single approach is foolproof — hence, a **multi-factor hybrid solution** that integrates software (mobile app + backend logic) with hardware fallback (fingerprint) is necessary.

Project Goals and Milestones

GOALS

- Mark attendance for a 200+ student class **in under 2 minutes**.
- Reduce proxy attendance probability to **<1%** through multi-factor checks.
- Core system implemented in **C using explicit DSAs** (hash table, linked lists, stacks, queues, heap/AVL) — for learning and performance.

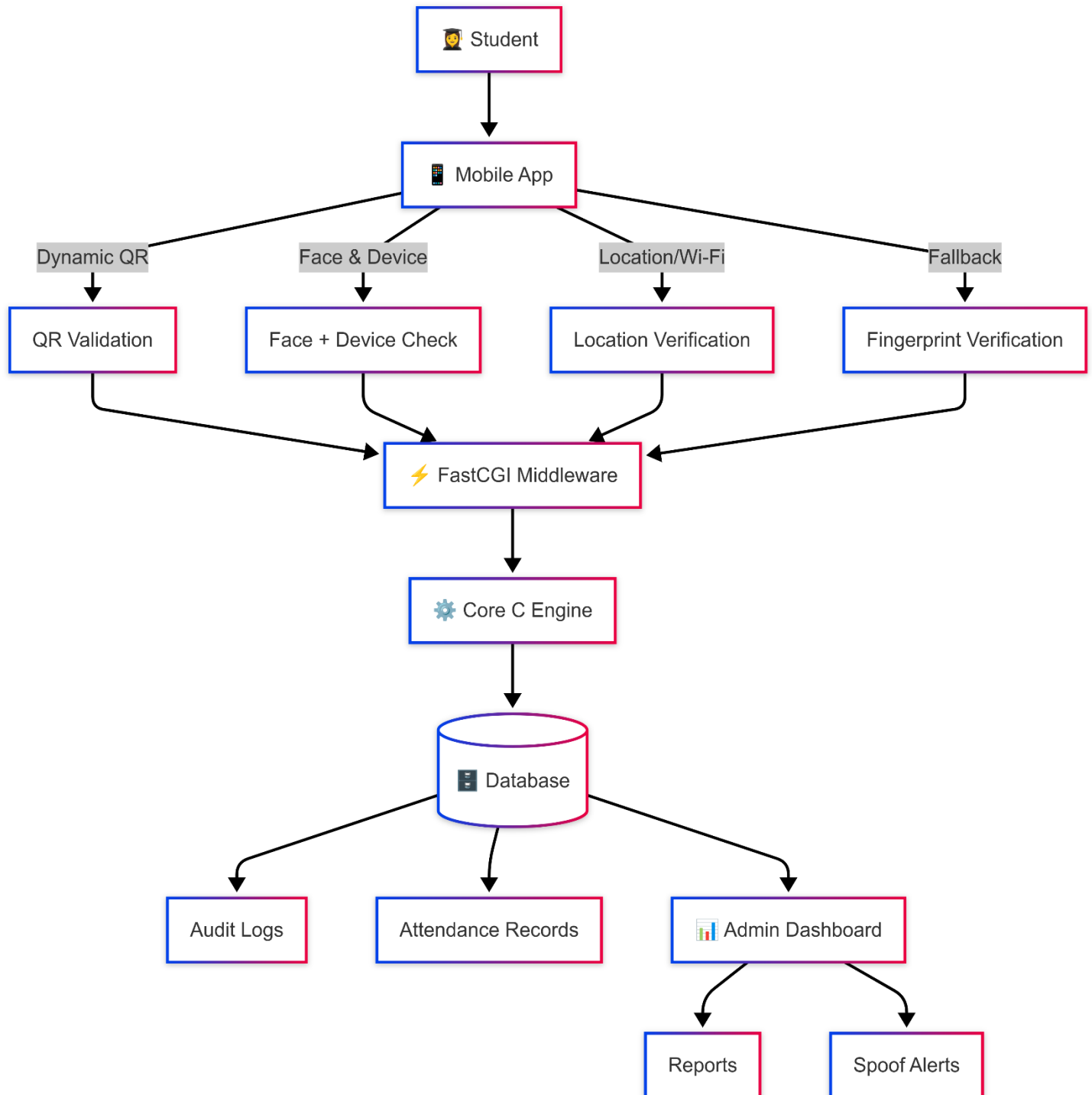
MILESTONES

- System design + database schema (SQLite/MySQL)
- Core logic in C (DSA-first implementation)
- FASTCGI integration for web/mobile access
- Mobile app module for QR + face liveness + device binding
- Location/Wi-Fi verification module
- Fingerprint fallback integration
- Testing with spoof attempts + audit logging
- Deployment + final validation

Project Approach

- **Backend Core:** Implemented in C, focusing on efficient data structures (hash maps for device binding, queues for dynamic QR management, logs stored in indexed structures).
- **Middleware:** Use FastCGI to expose C logic as web APIs.
- **Database:** Store attendance records, device fingerprints, and logs in SQLite/MySQL with indexing for fast lookups.
- **Frontend:** Mobile/web app (Android/iOS + browser client) for scanning dynamic QR, performing liveness checks, and syncing data.
- **Security:**
 - QR codes are time-bound and unique.
 - Device ID binding ensures only registered devices are valid.
 - Wi-Fi + GPS cross-check prevents spoofing.
 - Face liveness detection prevents photo/video replay.
 - Fingerprint fallback ensures robustness.
 - Audit Logging: Immutable logs for every transaction, enabling traceability and spoof detection.

System Architecture (High Level Diagram)



Project Outcome / Deliverables

- *Fully working hybrid attendance platform (mobile + web).*
- *C-based core engine with FastCGI APIs.*
- *Database with attendance + spoof attempt logs.*
- *Admin reporting dashboard.*
- *Demo dataset showing real vs. proxy attempts blocked.*

Assumptions

- *Students have access to smartphones with cameras, GPS, and fingerprint sensors.*
- *Institutional Wi-Fi is available for location verification.*
- *Fingerprint fallback devices are available at entry points.*

References

- *Research papers on **liveness detection and biometric spoofing prevention**.*
- *FastCGI developer documentation.*
- *SQLite/MySQL official documentation.*
- *Tutorials on **secure QR code generation**.*