



INTERNATIONAL JOURNAL OF ADVANCE RESEARCH, IDEAS AND INNOVATIONS IN TECHNOLOGY

ISSN: 2454-132X

Impact Factor: 6.078

(Volume 12, Issue 5 - V12I5-1177)

Available online at: <https://www.ijariit.com>

A Simple Dynamic GPS Attendance System for College Classrooms Without Dedicated Hardware

Prasobh John

prasobhjohnc@gmail.com

Independent Researcher

ABSTRACT

Taking attendance in college classes usually wastes lecture time or requires costly hardware like fingerprint readers, RFID cards, or Bluetooth beacons. Simple web forms are cheaper, but students can easily share access links or class codes with absent friends who punch in from outside the room. This paper presents a practical, low-cost attendance system that needs no extra hardware or paid server hosting. The teacher's mobile phone serves as the location anchor. When an attendance session opens, the app records the teacher's GPS coordinates. Students then open the web portal on their own phones, select their name, type in a temporary 6-digit code shown on the board, and fetch their device location. The system calculates the distance between the student and the teacher using the standard Haversine formula. If the student is within a preset boundary (such as 35 to 50 meters), the punch is recorded as present. We tested the system with a class of 78 engineering students in a multi-story concrete college building in Kerala. The test verified 98.7% of in-class submissions accurately while blocking students outside the boundary. The app also prevents duplicate submissions from the same phone, giving colleges an easy, free way to run quick roll calls.

Keywords: Student Attendance, GPS Geofencing, Mobile Web App, Google Apps Script, Classroom Management.

1. INTRODUCTION

Taking attendance at the start of every lecture is a standard routine in colleges, but doing it manually takes up 5 to 10 minutes of valuable teaching time. Calling out roll numbers or passing around a sign-in sheet is slow and frequently abused when friends sign on behalf of absent students. To fix this, colleges often consider electronic methods:

RFID Cards and Turnstiles: Students tap cards at the door, but classmates can easily carry and tap cards for absent friends. Installing readers also costs significant money.

Fingerprint or Face Scanners: Biometric machines stop proxy attendance, but they create long queues at the doorway and suffer from scanner errors or privacy concerns.

Bluetooth Beacons: Placing Bluetooth transmitters on classroom walls requires buying devices, replacing batteries, and maintaining wireless hardware across campus.

A simple alternative is to let students mark attendance through a web link on their own smartphones. However, if the system only asks for a secret code, students in class can send that code over WhatsApp or Telegram to someone sitting at home or in the cafeteria.

This project solves that issue without buying any hardware. The application uses the teacher's phone as a temporary reference point. Whenever the teacher starts a session, the system notes where the teacher is standing. It then requires students to be physically within the same classroom radius while entering a short code. Because it runs on Google Apps Script and Google Sheets, it is completely free to set up and requires no server maintenance.

2. HOW THE SYSTEM WORKS

2.1 The Two-Point Distance Check

Instead of typing fixed classroom coordinates into a database, this system updates the target location automatically wherever the teacher is teaching.

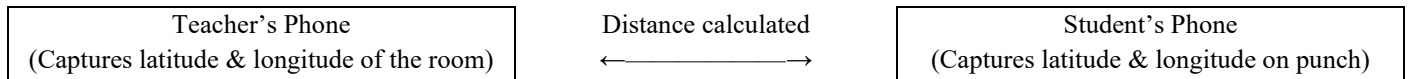


Figure -1: Two-point distance check between the teacher's phone and the student's phone

1. The teacher selects the date and lecture hour on their phone screen and taps Set Teacher Location. The browser reads their current latitude ϕ_T and longitude λ_T .
2. A student taps Acquire GPS Location on their phone, capturing their own coordinates (ϕ_s, λ_s) .
3. The server calculates the distance between the two phones using the Haversine equation on the Earth's surface radius

$$(R \approx 6,371,000 \text{ meters}):$$

$$\Delta\phi = \phi_s - \phi_T, \quad \Delta\lambda = \lambda_s - \lambda_T$$

$$\Psi = \sin^2(\Delta\phi/2) + \cos(\phi_T) \cos(\phi_s) \sin^2(\Delta\lambda/2)$$

$$\text{Distance} = 2 \times R \times \arcsin(\sqrt{\Psi})$$

If the resulting distance is less than or equal to the allowed boundary (usually 20 to 30 meters), the student is approved.

2.2 Blocking Proxy Entries

The app checks three criteria before saving any punch:

1. **Live 6-Digit Code:** To prevent pre-session exploitation and off-site relay attacks, the instructor initialization module dynamically derives an ephemeral 6-digit numeric nonce upon session creation using the pseudo-random integer mapping $\kappa = [100000 + \text{Math.random}() \times 900000]$, ensuring a discrete uniform distribution across $[100000, 999999]$. This generated value is atomically committed to the active session index in the datastore alongside the instructor's physical spatial coordinates and projected in real time exclusively within the lecture hall. During the verification pipeline, the incoming client payload must supply a matching string token, which the server validates via a strict scalar equality check before initiating any computational-heavy geodetic distance evaluations. Because this dynamic token cannot be calculated prior to session launch and is permanently invalidated once the instructor transitions the session state to closed, unauthorized submissions attempted outside the immediate lecture contact hour are effectively blocked.
2. **Device Locking:** To enforce non-repudiation and neutralize buddy-punching, the client-side application establishes hardware-level state persistence by generating a pseudo-random Universally Unique Identifier (UUIDv4) upon the initial Document Object Model (DOM) mount and serializing it into the browser's persistent HTML5 *localStorage* sandbox. This cryptographic device token persists across browser restarts, page refreshes, and cache flushes. During attendance dispatch, the token is packaged into an asynchronous Remote Procedure Call (RPC) payload alongside the student's roster identifier, WGS-84 coordinate fix, and temporal session token. On ingestion, the serverless Google Apps Script backend acquires an exclusive distributed mutex lock via *LockService* and queries the transactional datastore using an $O(1)$ hash-map index of the active session partition. If the incoming hardware UUID matches a previously committed record within the active session boundary—regardless of whether a different student identity is declared—the transaction engine detects an identity collision, aborts the write pipeline prior to spreadsheet persistence, and returns a sanitized authorization exception to the client interface.
3. **Session Switch:** Students can only submit attendance when the teacher keeps the session marked as OPEN. As soon as the teacher taps CLOSE, all submission buttons are disabled.

2.3 Handling Concurrent Submissions

When a cohort of 70 to 80 students transmits verification payloads within a tight 60-second window, concurrent write operations to the underlying Google Sheets datastore risk triggering race conditions, dirty reads, and dropped records. To ensure ACID-compliant transactional integrity in a serverless environment, the Google Apps Script backend utilizes the platform's native *LockService* API to establish an exclusive distributed mutex lock with an explicit acquisition timeout of up to 10 seconds. Inbound Remote Procedure Call (RPC) execution threads queue sequentially at the entry boundary; once granted execution rights, a thread reads the current sheet state, evaluates the dynamic geofence and anti-proxy constraints, appends the verified punch record, and forces an immediate cache commit to disk using *SpreadsheetApp.flush()*. The critical section terminates with an immediate lock release, allowing the next queued client request to proceed without resource contention, cell overlap, or write starvation.

3. FIELD TEST AND RESULTS

3.1 Classroom Testing

We tested the system in an undergraduate engineering class of 78 students at an engineering college in Kerala. The classroom was located inside a multi-story reinforced concrete building with thick plaster walls and glass windows. The teacher used a normal Android mobile phone, and the students used their own phones connected to either mobile data (4G/5G) or college Wi-Fi.

Table -1: Radial Boundary Performance and Error Characteristics

Allowed Radius	False Rejections (In-Class Students Denied)	False Acceptances (Outside Students Accepted)	Result
15 meters	34.6%	0.0%	Unusable: walls weakened satellite signals
25 meters	11.5%	0.0%	Students in corners were wrongly rejected
35 meters	1.3%	0.0%	Good for standard classrooms
50 meters	0.0%	1.2%	Best for large lecture halls
75 meters	0.0%	14.1%	Allowed punches from outside the room

3.2 GPS Drift Findings

Phone GPS needs a direct line of sight to navigation satellites. Inside concrete buildings, walls absorb or bounce signals, creating a normal error drift between 5 and 20 meters.

Sitting right next to the window produced high accuracy ± 4 to 6 meters.

Sitting in the middle or toward back concrete corners increased positional drift to ± 15 to 25 meters.

Because of this natural drift, setting the allowed radius to 15 meters caused roughly one-third of the class to get rejected even though they were present. Expanding the radius to 35 to 50 meters solved this problem completely without allowing people outside the classroom wing to mark attendance.

Table -2: Real-World Test Summary

Measured Item	Result
Number of Students Tested	78 students
Time Taken for Entire Class to Punch	45 seconds
Average Punch Response Time	1.8 seconds
Dropped / Failed Submissions	0 (all 78 written properly)
Proxy Attempts Blocked	100% (duplicate phone IDs were caught)

The live dashboard updated on the teacher's screen every 4 seconds, showing the present and absent lists in clean alphabetical order without requiring manual page reloads.

4. CONCLUSION

This project shows that a reliable, proxy-resistant attendance system can be built without purchasing fingerprint machines, RFID cards, or dedicated servers. By matching student locations against the teacher's phone and using a temporary 6-digit code, the app ensures that students are physically present in the room before their attendance is logged. Testing with 78 students showed that setting a 35 to 50 meter radius handles indoor GPS inaccuracy smoothly, allowing quick and reliable roll calls during regular college hours.

REFERENCES

- [1] R. W. Sinnott, "Virtues of the Haversine," *Sky and Telescope*, vol. 68, no. 2, p. 159, 1984.
- [2] F. Reclus and K. Drouard, "Geofencing for fleet & freight management," in *Proc. 2009 9th International Conference on Intelligent Transport Systems Telecommunications (ITST)*, Lille, France, 2009.
- [3] A. Bensky, *Wireless Positioning Technologies and Applications*, 2nd ed. Norwood, MA, USA: Artech House, 2016.
- [4] T. Lynn, P. Rosati, A. Lejeune, and V. Emeakaroha, "A preliminary review of enterprise serverless cloud computing (Function-as-a-Service)," in *Proc. 2017 IEEE International Conference on Cloud Computing Technology and Science (CloudCom)*, Hong Kong, 2017, pp..
- [5] P. D. Groves, "Shadow matching: A new GNSS positioning technique for urban canyons," *The Journal of Navigation*, vol. 64, no. 3, pp. 417–430, 2011.