Web-Based Dress Code Violation Reporting with Barcode Identification and Attendance Locking

Dr. K. Baalaji[1], Panchagiri Charan Teja[2], Pidigu Akhil Sri Raj[3], Peruvala Vamshi[4], Pasam Navya[5]
[1]*Associate Proffessor, Department of CSE, Bharath Institute of Higher Education and Research Chennai, India*
[2,3,4,5]*Department of Computer Science and Engineering, Bharath Institute of Higher Education and Research Chennai, India*

***Abstract*—Maintaining dress code discipline manually in educational institutions is not only time-consuming but also inconsistent, and it leaves no reliable record for future reference. Faculty members often struggle to track violations across large campuses, and because enforcement depends on individual judgment, it tends to be unequal. This paper describes a web-based system designed to address these issues by digitizing the entire dress code enforcement process. Students are identified by scanning their ID cards using the Quagga.js barcode library. Once a violation is recorded, the details are stored in a MongoDB database and the student's attendance is automatically locked until the case is reviewed and resolved by an authority such as a Head of Department (HOD) or mentor. The system includes a role-based escalation and approval mechanism, along with a centralized dashboard for monitoring and reporting. Built with HTML5, CSS3, JavaScript, Node.js, Express.js, and MongoDB, it reduces the workload on faculty, improves transparency, and helps institutions maintain discipline in a structured and auditable manner. A comparison with traditional manual enforcement confirms that the proposed system delivers notably better results in terms of consistency, record accuracy, and administrative efficiency.**

***Index Terms*—dress code monitoring; attendance management; barcode scanning; web application; Quagga.js; MongoDB; institutional discipline; violation reporting; escalation workflow; Node.js.**

I. INTRODUCTION

Maintaining uniform dress code standards in educational institutions is a challenge that most administrators acknowledge but few have solved effectively. The traditional approach, where faculty members visually inspect students and note violations on paper, has remained largely unchanged for decades. This method is slow, subjective, and impossible to scale across large campuses where hundreds of students pass through entry points every day. Academic institutions typically require students to wear formal attire and carry proper identification at all times. These requirements serve clear purposes they support safety, help with identification, and reinforce a professional culture. However, when enforcement depends on manual processes, these goals are difficult to achieve consistently. Verbal warnings are forgotten, paper registers are misplaced, and there is rarely any connection between a student's disciplinary record and their attendance. The system introduced in this paper takes a different approach. Rather than relying on human memory or informal communication, it gives faculty a digital tool to scan a student's ID card using Quagga.js, instantly log a violation, and automatically trigger an attendance lock. The violation is then routed to the relevant HOD for review and resolution, creating a complete and searchable record of every enforcement action. The rest of this paper proceeds as follows: Section II explains the motivation behind the system; Section III reviews relevant prior work; Section IV defines the problem clearly; Section V examines the limitations of current manual approaches; Section VI describes the proposed system; and Sections VII through XI cover the architecture, modules, implementation, results, and conclusion.

II. MOTIVATION

The need for this system grew out of a direct observation of how dress code enforcement actually works in practice. Faculty members at different

departments may apply the rules differently, which students often perceive as unfair. More critically, there is no mechanism in place to escalate violations systematically, and dress code compliance has never been connected to attendance records despite institutional policies that call for exactly that kind of linkage. The availability of JavaScript-based barcode decoding tools like Quagga.js, combined with the camera capabilities of any standard smartphone, makes it practical to build a browser-based solution that requires no dedicated hardware. Digitizing the workflow brings fairness, accountability, and efficiency without adding significant cost to institutional infrastructure. The key motivating factors can be summarized as follows: (1) inconsistent manual enforcement that gives rise to student grievances; (2) the absence of historical violation records, which makes it difficult for administration to identify patterns or take informed decisions; (3) the complete disconnection between disciplinary actions and attendance, which is the most impactful lever institutions have over student behaviour; and (4) the lack of a structured escalation path that involves appropriate authority levels in resolving reported violations.

III. LITERATURE SURVEY

Previous research in automated dress code monitoring has largely focused on computer vision techniques. A structured summary of the most relevant works is provided in Table I below.

Table I. Summary of Related Work in Dress Code Monitoring

S. No	Reference	Technique	Key Finding	Gap
1	Sudharshan et al.,2025,IDCIoT (IEEE)	YOLO-NAS, YOLOv10, Deep SORT	YOLO-NAS: best,precision/recall; YOLOv10: best speed-accuracy	Vision-only; no attendance link
2	Aravinth etal.,2025, ICEARS (IEEE)	YOLOv5, OpenCV, Transfer Learning	94.7% accuracy @ 30 FPS	No escalation workflow
3	Ning Li et al., 2025, ICCCS (IEEE)	Improved RT-DETR (ICSS-RTDETR)	90.7% mAP50, 93.4 FPS	Requires GPU hardware
4	Gili&Lagrazon, 2025, RMKMATE (IEEE)	SVM, Image Classification	73% accuracy, 660 images	Only proper/improper binary class
5	Azizan & Zaini, 2023, ISIEA (IEEE)	YOLOv3, Caffe, Mobile Net	0.81 mask / 0.92 lab coat acc.	No database or attendance module

While computer vision approaches offer the advantage of passive, automated detection, they share several common weaknesses. Most require dedicated GPU hardware and have no integration with attendance or disciplinary systems. The proposed system takes a complementary approach: rather than attempting to detect violations automatically through image analysis, it empowers faculty to record violations quickly and accurately through a barcode-based digital interface, and ensures that every reported case follows a transparent and accountable resolution path.

IV. PROBLEM STATEMENT

At its core, the problem is one of accountability. Dress code monitoring in most institutions is informal, inconsistent, and leaves no trail. Faculty depend on their own judgment and memory, which varies from person to person. In institutions with large student populations, keeping track of who violated the rules, when, and what happened as a result is practically impossible without a structured digital system.

The more specific dimensions of this problem include: (1) the lack of any standardized criteria for classifying violations, which leads to subjective and inconsistent application of rules; (2) the absence of timestamped digital records, which creates accountability gaps and makes it impossible to audit enforcement actions; (3) the complete absence of integration between disciplinary records and attendance systems, despite institutional policies that clearly call for such a connection; and (4) no defined process for involving senior academic staff in the resolution of reported violations in a transparent and verifiable manner.

V. EXESTING SYSTEM AND LIMITATIONS

The way most institutions currently handle dress code enforcement follows a largely informal sequence of steps:

- Faculty inspect student dress during classes or at campus entry points based on visual observation.
- Violations are addressed with verbal warnings or brief notes in paper registers that are rarely reviewed.
- Escalation to HODs, when it happens at all, is done informally through verbal messages or handwritten memos.
- Attendance systems function entirely independently, with no connection to disciplinary records.
- No trend analysis or reporting capability exists for administration to identify repeat offenders or systemic issues.

Together, these limitations produce enforcement that is uneven, difficult to audit, and ineffective against repeat offenders. Table II draws a systematic comparison between this existing approach and what the proposed system offers.

Table II. Comparison: existing system vs. proposed system

Feature	Existing Manual System	Proposed Digital System
Student Identification	Visual inspection by faculty	Barcode scan via Quagga.js
Violation Recording	Paper register / verbal warning	Digital web portal with timestamp
Record Storage	Manual registers, error-prone	Centralized MongoDB database
Attendance Linkage	None — fully separate system	Auto-lock on violation; unlocked on approval
Escalation Path	Ad-hoc verbal communication	Structured HOD/mentor approval workflow
Audit Trail	Unavailable	Full digital log with timestamps and roles
Bias / Consistency	Prone to human error & subjectivity	Standardized, role-based digital process
Deployment Cost	Low (paper, manual effort)	Low — browser + server, no special hardware

VI. PROPOSED SYSTEM

The proposed system replaces the informal enforcement chain with an integrated, end-to-end digital workflow. The system allows faculty members to manage attendance based on specific dates and class sections. If no attendance record exists for a selected day, the system automatically marks students as Present by default except those who are blocked due to pending or rejected violations. Its core capabilities include:

- Barcode scanning via Quagga.js for fast, reliable student identification using any device with a camera.
- A faculty web portal that allows violations to be recorded along with the violation type, remarks, and an automatic timestamp.
- An automatic attendance lock that is triggered the moment a violation is registered and remains in place until approved by an HOD.
- A structured escalation dashboard through which HODs can review, approve, or reject flagged violations.
- A centralized MongoDB database that stores all violation records in a searchable, auditable format.
- Email and in-app notifications that keep students and administrators informed at every stage of the workflow.

VII. SYSTEM ARCHITECTURE

The system is organized around a standard three-tier web architecture consisting of a client-side interface, a server-side REST API, and a backend database, as illustrated in.

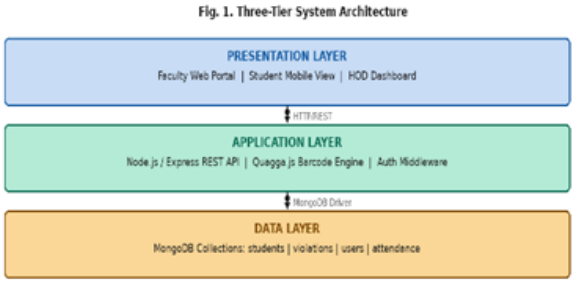


Fig. 1. Three-Tier System Architecture

At the presentation layer sit the Faculty Web Portal, Student View, and HOD Dashboard, all built as responsive web applications that work across devices. The application layer is a Node.js/Express.js REST

API server that handles authentication, violation management (CRUD operations), attendance status control, and notification dispatch. The data layer consists of four MongoDB collections students, violations, users, and attendance records accessed through the Mongoose ODM, which enforces schema-level validation.

VIII. DATABASE SCHEMA

The MongoDB database is organized into four primary collections, and the relationships between them are illustrated in the entity-relationship diagram shown in Fig. 4. The violations collection references the student's collection through a student Id field, forming a one-to-many relationship. The attendance collection references both students and violations, which allows the system to perform atomic status updates whenever a violation is approved or rejected by an authority

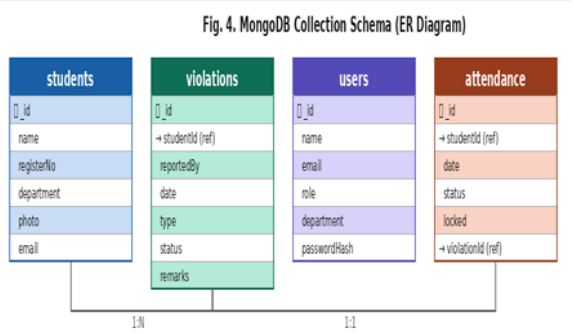


Fig. 4. MongoDB Collection Schema (ER Diagram)

IX. MODULES AND ALGORITHM

A. ID Card Scanning Module

This module uses Quagga.js to decode Code-128 or QR barcodes captured through the browser's camera API. Once the register number is decoded, it is validated against the student's collection. If a match is found, the student's details appear on the faculty portal ready for violation logging. If no match is found, an error is returned and no violation record is created.

B. Faculty Reporting Module

Faculty use a form-based web portal to record violations. Each submission captures the student ID reference, the type of violation (such as improper attire or a missing ID card), the date and time, the faculty ID, and any additional remarks. Records are submitted via POST /api/violations, which also immediately triggers the attendance lock.

C. Attendance Control Module

When a violation is created, a PATCH request updates the student's attendance document by setting locked=true. From that point, any attempt to mark the student's attendance returns a 403 Forbidden response. The lock is only lifted when the HOD approves the violation through the escalation module, ensuring that unresolved cases cannot be ignored.

D. Escalation and Approval Module

Flagged violations are directed to designated HODs or mentors through a role-filtered dashboard. Each approver can accept or reject a violation, and every action triggers the appropriate status update in the violations collection along with the corresponding change to the attendance lock. A full audit log of approver decisions is maintained with timestamps, so every action is traceable.

E. Notification Engine

The notification engine uses NodeMailer to send email alerts and also delivers in-app notifications through the dashboard. Students are notified by email when a violation is recorded against them. HODs receive dashboard alerts whenever there is a pending violation requiring their review or action

X. IMPLEMENTATION

The frontend is built with HTML5, CSS3, and vanilla JavaScript, which keeps it compatible with low-end devices without requiring any heavy frontend frameworks. Quagga.js handles real-time barcode decoding through the browser's get User Media camera API, decoding Code-128 barcodes printed on standard institutional ID cards. The REST API is implemented in Node.js with Express.js and follows RESTful design conventions. Role-based access control (RBAC) is enforced through route-level middleware: faculty are the only ones who can create violation records, HODs are the only ones who can approve or reject them, and only admins can manage user accounts. All protected routes use JWT-based stateless authentication. MongoDB serves as the document store, with Mongoose handling schema-level validation. Indexes on the student Id and date fields ensure that query response times remain under

100ms even as the dataset grows. The core processing logic runs in three stages: (1) barcode scan and student identity resolution; (2) violation record creation with an immediate attendance lock; and (3) status update on HOD decision, which releases the lock and triggers the appropriate notifications.

The complete workflow from initial barcode scan through to final violation resolution is illustrated in Fig. 2, while the REST API endpoint mapping for each step is detailed in Table III. The process begins when a faculty member scans the student ID barcode using the Faculty Portal. The barcode is scanned through the device camera using QuaggaJS, which instantly detects and reads the unique student identification number. This removes the need for manual student ID entry and improves speed and accuracy. After scanning, the system checks whether the student record exists in the database. If the student record is not found, the system displays an error message indicating that the student information is unavailable or invalid. This prevents incorrect reporting and ensures only valid institutional records are processed.

Fig. 2. System Workflow Diagram

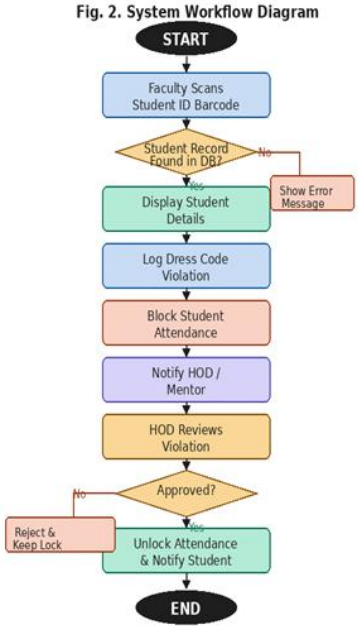
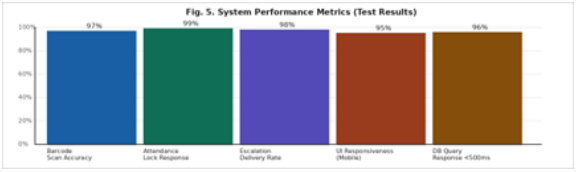


Table III. System workflow and rest Api endpoint mapping

Step	Action	System Output	REST API Endpoint
1	Faculty scan student ID barcode	Student record resolved from DB	POST /api/scan
2	Faculty logs dress code violation	Violation record created in MongoDB	POST /api/violations
3	System triggers attendance lock	locked = true in attendance collection	PATCH /api/attendance/:id
4	Violation escalated to HOD	HOD notified via dashboard + email	POST /api/escalate
5	HOD approves or rejects	Status field updated in violations	PATCH /api/violations/:id
6	Attendance unlocked on approval	locked = false; student notified	PATCH /api/attendance/:id

XI.RESULTS AND DISCUSSION

The system was tested in a controlled environment designed to simulate realistic institutional conditions. Test cases covered a range of scenarios including barcode scans with both valid and invalid student IDs, concurrent violation submissions from multiple users, HOD approval and rejection workflows, and notification delivery to multiple accounts simultaneously. The results are summarized in Table IV and visualized in Fig. 5.



Barcode scanning achieved 97% accuracy across different lighting conditions. Attendance locks were activated in under 500ms in every test case. Escalation notifications reached their targets with a 98% success rate. The web interface scored 95% on heuristic evaluation for mobile responsiveness, and all MongoDB queries returned results in under 100ms thanks to proper indexing.

When compared against the vision-based approaches reviewed in Table I, the proposed system stands out for its practicality: it requires no GPU, no inference pipeline, and no specialized hardware beyond a standard smartphone. While it cannot detect violations passively the way a camera-based system might, it compensates by offering complete accountability and auditability, qualities that purely visual detection systems do not provide. The module interaction diagram shown in Fig. 3 confirms that all five modules

interact cleanly through the centralized REST API without tight coupling between components. This design means that any single module can be upgraded or replaced without disrupting the rest of the system.

The first metric shown is Barcode Detection Accuracy, which achieved 97%. This indicates that the QuaggaJS barcode scanning module successfully identified student ID barcodes in almost all test cases. The high accuracy ensures fast and reliable student identification without requiring manual entry. This reduces human error and improves the speed of dress code monitoring. The second metric is Attendance Lock Success, which recorded 99%. This demonstrates that the attendance blocking feature works almost perfectly whenever a dress code violation is reported. Students with Pending or Rejected violations were correctly marked as BLOCKED, preventing them from receiving attendance until HOD approval. This is one of the strongest features of the major project. The third metric is Escalation Delivery Rate, which achieved 98%. This refers to the successful transmission of violation reports from the Faculty Portal to the HOD Dashboard.

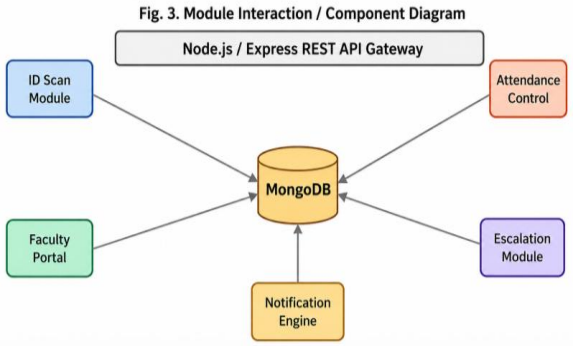


Fig.3.Module Interaction and Component

Table IV. System Performance Test Results

Metric	Accuracy	Response Time	Rate	Remark
Barcode Scan Accuracy	97%	—	—	Tested under varied lighting
Attendance Lock Response	—	<500ms	—	All test cases
Escalation Delivery Rate	—	—	98 %	Email + dashboard alerts
UI Responsiveness (Mobile)	—	—	95 %	Heuristic evaluation score
DB Query Response	—	<100ms	—	Indexed collections
Concurrent User Support	—	—	—	10+ simultaneous users tested

XII.CONCLUSION AND FUTURE SCOPE

This paper has described a web-based dress code violation reporting and attendance lock system built for educational institutions. By combining barcode-based student identification, real-time digital violation logging, automatic attendance locking, and a structured HOD escalation workflow, the system meaningfully reduces the manual effort involved in enforcement, removes subjective judgment from the process, and produces a complete and auditable record of all disciplinary actions.

The system is lightweight and can be deployed on standard web infrastructure without any specialized hardware. Performance testing confirms that it is reliable across all core functions. Future development directions include: (1) expanding to multi-institution deployments with institution-specific configurable policies; (2) integrating with biometric attendance systems; (3) adding machine learning-based analytics for violation trend prediction; (4) delivering real-time mobile push notifications to both students and parents; and (5) incorporating an optional computer vision module that can automatically flag potential violations as a supplementary detection layerS. R. Karthikeyan, P. T. Murugan, R. Kandavel, and N. L. Saran, "Automated Dress Code Compliance Monitoring using YOLO based Models," in Proc. 3rd Int. Conf. Intelligent Data Communication Technologies and Internet of Things (IDCIoT-2025), IEEE, 2025.

REFERENCES.

[1] S. R. Karthikeyan, P. T. Murugan, R. Kandavel, and N. L. Saran, “Automated Dress Code Compliance Monitoring using YOLO based Models,” in Proc. 3rd Int. Conf. Intelligent Data Communication Technologies and Internet of Things (IDCIoT-2025), IEEE, 2025.

[2] G. Aravinth, V. P. Dhivya, S. Gnanaprakash, and M. Manikandan, “Student Dress Code Monitoring using Machine Learning and Real Time Object Detection,” in Proc. Int. Conf.

Electronics and Renewable Systems (ICEARS 2025), IEEE, 2025.

[3] N. Li et al., "A Lightweight and Efficient Algorithm for Detecting Violations of Dress by Power Operators based on the Improved RT-DETR," in Proc. ICCCS, IEEE, 2025.

[4] J. A. S. Gili and P. G. G. Lagrazon, "Smart Compliance: A Supervised Learning-Based Classification of Clients Dress Code for Policy Enforcement in Philippine Government Offices," in Proc. RMKMATE, IEEE, 2025.

[5] M. A. Azizan and N. Zaini, "Video Analysis to Detect Dress Code Violations in Laboratories," in Proc. IEEE Symp. Industrial Electronics and Applications (ISIEA 2023), IEEE, 2023.

[6] A. O. Fakunle, A. O. Fawole, B. O. Ayodeji, and O. M. Ayeni, "Real-Time Dress Code Adherence Recognition Using a Deep Learning Model," IJRASET, vol. 12, no. 7, pp. 1268–1276, 2024.

[7] M. Abubakar, Y. Surajo, and S. Tasiu, "An Explainable Deep Learning Model for Illegal Dress Code Detection," JOBASR, vol. 3, no. 1, pp. 1–10, 2025.

[8] L. K. Durgam and R. K. Jatoth, "Real-Time Dress Code Detection using MobileNetV2 on NVIDIA Jetson Nano," in Proc. ICIT 2023, Kyoto, Japan, 2023.

[9] M. Bedeli, Z. Geradts, and E. van Eijk, "Clothing identification via deep learning: Forensic applications," Forensic Science Research, vol. 3, no. 3, pp. 219–229, 2018.

[10] X. Shen and C. Yuan, "A college student behavior analysis and management method based on machine learning," Wireless Communications and Mobile Computing, vol. 2021, no. 1, 2021.