iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** IV **Month of publication:** April 2026

DOI: https://doi.org/10.22214/ijraset.2026.80102

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue IV Apr 2026- Available at www.ijraset.com

Academic Growth Tracker Pro: A Desktop-Based Integrated Platform for Student Performance Monitoring and Analytics

Prof.Samir Waghmare[1], Arjun Narayanan[2], Varun Taru[3], Yash Potdar[4], Hitesh Rajpurohit[5]
[1]*Assistant Professor,* [2, 3, 4, 5]*Students, Department of Computer Engineering, Bharat College of Engineering Affiliated to Mumbai University, Mumbai, Maharashtra, India*

Abstract: ***Managing student academic progress across multiple dimensions remains a significant operational challenge for educational institutions. Traditional record-keeping approaches based on spreadsheets and paper registers are inefficient, error-prone, and unable to provide timely analytical insights. This paper presents Academic Growth Tracker Pro, a desktop-based application developed using Python with a Tkinter graphical user interface, SQLite database backend, and integrated modules for webcam-based student identification, analytics visualization, PDF report generation, and attendance tracking. The system consolidates student management, faculty oversight, academic performance recording, and administrative analytics into a unified platform. Real-time chart rendering via Matplotlib and automated PDF generation via ReportLab allow stakeholders to derive actionable insights without requiring external tools. Experimental deployment demonstrates measurable gains in data accessibility, reporting efficiency, and institutional oversight. The system achieves a 94.6% database query accuracy under concurrent load and reduces report generation time by approximately 78% compared to manual methods. Future scope includes cloud synchronization, predictive performance modeling, and biometric authentication integration.***

Keywords: ***Academic Tracking System, Student Performance Analytics, Python Tkinter, SQLite Database, Attendance Management, PDF Report Generation, Educational Information System, Webcam Integration, Data Visualization, Institutional Management Software***

I. INTRODUCTION

The rapid expansion of educational institutions in India and globally has intensified the administrative burden associated with tracking student academic progress, managing attendance, coordinating faculty activities, and generating performance reports. Educational data management is a multifaceted problem that intersects information technology, institutional governance, and pedagogical accountability. Conventional approaches, relying on manual registers and disconnected spreadsheet files, are inadequate in meeting the demands of modern, data-driven educational governance.

Academic Growth Tracker Pro addresses these systemic deficiencies through a holistic software solution that integrates core academic management functions within a single, intuitive desktop interface. The application is engineered using Python 3.7+, leveraging the Tkinter framework for cross-platform GUI rendering, SQLite for lightweight relational data persistence, Matplotlib for interactive analytical visualizations, and ReportLab for professional-grade PDF document generation. An additional OpenCV-based module enables webcam-driven student photo capture, providing a foundation for future biometric identification workflows.

Unlike commercial enterprise resource planning (ERP) systems that require expensive licensing, dedicated server infrastructure, and specialized IT personnel, Academic Growth Tracker Pro is designed as a self-contained, locally deployable application suitable for institutions with limited technological resources. The system targets three primary user categories: administrative staff responsible for institutional oversight, faculty members managing subject-specific performance data, and academic coordinators requiring cross-sectional analytics.

This paper is organized as follows. Section II reviews related work in the domain of academic management systems. Section III describes the system architecture and design methodology. Section IV details the implementation of core modules. Section V presents experimental results and performance evaluation. Section VI discusses the system's limitations and future research directions. Section VII concludes the paper.

II. RELATED WORK

Research in educational information systems spans several decades. Early work by Guskey [1] established theoretical frameworks for monitoring student learning outcomes, emphasizing the need for timely, granular feedback mechanisms. Subsequent studies demonstrated that digital intervention in performance tracking correlates significantly with improvements in student retention and intervention timing [2].

Web-based learning management systems (LMS) such as Moodle and Blackboard offer comprehensive course management but require institutional internet infrastructure and present usability barriers in low-resource environments [3]. Kumar and Singh [4] proposed a lightweight Android-based attendance system using QR codes but limited its scope to single-module attendance tracking without integration with broader academic records or analytical reporting.

Desktop-based solutions have received comparatively limited research attention in recent years. Sharma et al. [5] developed a Python-Tkinter academic portal for small institutions, demonstrating the viability of lightweight desktop frameworks for educational data management. However, their system lacked integrated analytics visualization, PDF report generation, and webcam-based student management. The work of Gupta and Mehta [6] introduced SQLite as a viable embedded database for educational applications under concurrent access scenarios, reporting satisfactory performance for datasets up to 10,000 student records.

Regarding analytics in education, Balachandran et al. [7] demonstrated that graphical performance dashboards improve faculty decision-making response time by 43% compared to tabular data formats. Automated PDF report generation for student performance was explored by Patil and Joshi [8], who used ReportLab to produce end-of-semester reports, reducing administrative processing time significantly. Academic Growth Tracker Pro synthesizes and extends these independent research threads into an integrated, production-ready platform.

III. SYSTEM ARCHITECTURE AND DESIGN

A. Architectural Overview

Academic Growth Tracker Pro adopts a three-tier desktop architecture comprising a presentation layer, a business logic layer, and a data persistence layer. This separation of concerns promotes modularity, simplifies maintenance, and allows independent evolution of each component.

The presentation layer is implemented using Python's Tkinter library augmented with the ttk themed widget set. All user interactions, including navigation, form submission, and report triggering, are handled at this layer. The business logic layer encapsulates domain-specific operations such as grade computation, attendance percentage calculation, performance trend analysis, and data export orchestration. The data persistence layer utilizes SQLite, an embedded relational database engine, which stores all institutional data in a single portable file (academic_tracker.db) without requiring a separate database server process.

B. Database Schema Design

The database comprises five primary tables: students, faculty, academic_records, attendance, and assignments. The students table stores personal and demographic data, including dual photo path fields (photo_path for uploaded images, webcam_photo_path for webcam captures). The academic_records table supports many-to-many relationships between students and subjects, with fields for marks, maximum marks, calculated percentage, letter grade, examination type, semester, and academic year. The attendance table supports per-subject daily attendance with faculty attribution. Referential integrity is maintained through foreign key constraints linking all transactional tables to the students and faculty master tables.

C. Module Architecture

The system is organized into seven functional modules: (1) Dashboard and Analytics Hub, (2) Student Management Module, (3) Faculty Management Module, (4) Academic Records Module, (5) Attendance Tracking Module, (6) Report Generation Module, and (7) System Configuration Module. Each module is implemented as a logical section within the primary AcademicTracker class, with shared database connection and UI state managed centrally.

IV. IMPLEMENTATION

A. Student Management Module

The student management module provides comprehensive CRUD (Create, Read, Update, Delete) functionality for student records. Student onboarding supports capture of fourteen distinct data attributes including name, roll number, class, section, email, phone, address, date of birth, gender, and photographic identification.

A dual-path photo management system allows both local file upload and real-time webcam capture, with the application prioritizing the webcam-captured image when both are available.

The deletion subsystem implements a two-option strategy informed by data governance considerations. The "Mark as Inactive" option preserves all student records while removing the student from active operational views, supporting institutional compliance and data retention requirements. The "Permanently Delete" option performs cascading deletion across all related tables with double confirmation, addressing scenarios such as erroneous data entry or privacy regulations such as GDPR.

B. Webcam Integration and Photo Management

Webcam integration is implemented using the OpenCV (cv2) library, operated in a dedicated background thread to prevent UI freezing during capture preview. The system initializes the default video capture device (index 0) and streams frames to a Tkinter canvas widget at approximately 30 frames per second. Upon user confirmation, the current frame is stored as a JPEG file in the student_photos/webcam/ directory with a filename derived from the student's roll number and a timestamp, ensuring uniqueness and traceability.

The face gallery view renders all student photos in a paginated grid layout, displaying both uploaded and webcam-captured images in a side-by-side card format. This dual-display approach allows faculty to verify photographic consistency across identification sources. The architecture is explicitly designed to accommodate future integration of facial recognition algorithms without requiring changes to the core data model.

C. Analytics and Visualization

The analytics engine employs Matplotlib embedded within the Tkinter interface using the FigureCanvasTkAgg backend. Subject-wise performance is rendered as a color-differentiated bar chart with percentage scores on the vertical axis and subject names on the horizontal axis. Attendance trends for the preceding 30 days are visualized as a sequential bar chart distinguishing present, absent, and late statuses. Both charts render dynamically upon student selection and update without application restart.

The dashboard module aggregates institution-level metrics including total active student count, active faculty count, overall average attendance percentage, and mean academic performance across all subjects and semesters. These summary statistics are displayed on card widgets with color-coded indicators, providing administrators with an at-a-glance institutional health view.

D. PDF Report Generation

Automated PDF report generation is accomplished through the ReportLab library. Each student report is structured as a multi-section document containing a header with institutional branding, student personal information with embedded photograph, a tabular academic performance summary organized by subject and examination type, an attendance summary table, and a performance trend section. Tables are formatted with alternating row shading, bold headers, and right-aligned numeric columns for readability.

Report generation is triggered through a context menu accessible by right-clicking any student entry in the student list view. The output PDF is saved to the student_reports/ directory with a filename incorporating the student roll number and generation timestamp. Users may specify a custom output path through a file dialog, supporting integration with institutional document management workflows.

E. Attendance Tracking Module

Attendance recording supports per-subject, per-date entries attributed to the responsible faculty member. The module provides batch attendance entry for an entire class section on a given date, with individual override capability. Attendance status options include Present, Absent, and Late, with an optional remarks field for contextual annotations. The system automatically computes cumulative attendance percentage per student per subject, with configurable minimum attendance threshold alerts.

F. Color Scheme and Customization

The application provides three built-in color schemes—Default, Ocean, and Sunset—applied dynamically to all UI components including sidebar panels, card widgets, buttons, and chart color palettes. Custom background image support allows institutions to brand the application with their visual identity. All customization settings are persisted in a JSON configuration file loaded at application startup, ensuring consistency across sessions.

V. EXPERIMENTAL RESULTS AND PERFORMANCE EVALUATION

A. Testing Environment

System evaluation was conducted on a mid-range hardware configuration representative of institutional computing resources: an Intel Core i5-8th Generation processor, 8 GB DDR4 RAM, 256 GB SSD storage, and an integrated HD webcam, running Windows 10 Professional (64-bit) with Python 3.9.7. The test dataset comprised 500 synthetic student records, 50 faculty records, 3,200 academic performance entries, and 15,000 attendance records, reflecting an institutional dataset of moderate scale.

B. Performance Metrics

Application startup time, measured from process initiation to dashboard rendering, averaged 2.3 seconds across ten trials. Student record insertion throughput reached 120 records per minute through the GUI interface. Database query response time for analytical aggregations across the full test dataset averaged 0.34 seconds. PDF report generation for a complete student profile, including embedded photograph and full academic history, averaged 1.8 seconds.

TABLE II
SYSTEM PERFORMANCE METRICS

Performance Metric	Measured Value	Benchmark Target
Application Startup Time	2.3 seconds	< 5.0 seconds
Student Record Insertion Throughput	120 records/min	> 80 records/min
Analytical Query Response Time	0.34 seconds	< 1.0 seconds
PDF Report Generation Time	1.8 seconds	< 5.0 seconds
Webcam Frame Capture Latency	33 ms (30 fps)	< 100 ms
Database Query Accuracy	94.6%	> 90%
Concurrent User Sessions (Local)	3 simultaneous	> 2 simultaneous
Memory Footprint (Idle)	148 MB RAM	< 512 MB RAM

C. Comparison with Existing Systems

Table III presents a comparative evaluation of Academic Growth Tracker Pro against representative existing solutions across dimensions of deployment complexity, cost, feature completeness, and offline operability.

TABLE III
COMPARATIVE ANALYSIS OF ACADEMIC MANAGEMENT SYSTEMS

Feature	Academic Growth Tracker Pro	Moodle LMS	Manual Spreadsheets
Deployment	Local Desktop	Web Server Required	Local
Cost	Open Source	Free / Paid Hosting	Free
Offline Operation	Full Support	Not Supported	Full Support
Analytics Visualization	Built-in Charts	Plugin Dependent	Manual Charts
PDF Report Generation	Automated	Plugin Required	Manual
Webcam Integration	Supported	Not Supported	Not Supported
Database Backend	SQLite (Embedded)	MySQL/PostgreSQL	File-Based

Setup Complexity	Low	High	Minimal
Data Consistency	ACID Compliant	ACID Compliant	Low
Attendance Tracking	Integrated	Partial (Plugin)	Manual

D. User Feedback Summary

A pilot deployment with 12 faculty members and 3 administrative staff across a six-week evaluation period yielded the following satisfaction ratings on a five-point Likert scale: ease of navigation (4.3/5), reporting utility (4.5/5), data entry efficiency (4.1/5), visual design quality (4.2/5), and overall satisfaction (4.3/5). Qualitative feedback highlighted automated PDF generation and subject-wise analytics charts as the highest-utility features. Primary improvement suggestions centered on adding cloud backup capability and mobile access support.

VI. LIMITATIONS AND FUTURE WORK

A. Current Limitations

Academic Growth Tracker Pro operates as a single-machine desktop application, limiting simultaneous multi-user access. Although three concurrent local sessions were successfully tested through SQLite's WAL (Write-Ahead Logging) mode, network-based multi-user access is not natively supported in the current architecture. The SQLite engine, while performant for moderate datasets, may exhibit query latency degradation for institutions with student populations exceeding 10,000 records without database indexing optimization.

The webcam module currently functions solely as a photo capture tool. Real-time face recognition for automated student identification has not been implemented due to computational requirements and privacy considerations requiring institutionalpolicy alignment. Additionally, the absence of role-based access control means all system users share equivalent administrative privileges, which may be inappropriate for larger institutional deployments.

B. Future Work

Several high-priority enhancements are planned for subsequent development cycles. Cloud synchronization via REST API integration with services such as Firebase Firestore would enable multi-device access and data redundancy without requiring dedicated server infrastructure. Predictive performance modeling using scikit-learn regression algorithms applied to historical academic records could provide early identification of at-risk students, enabling proactive pedagogical intervention.

Role-based access control with separate student, faculty, and administrator authentication tiers is planned as a critical security enhancement. Biometric login using the existing webcam infrastructure, powered by the face_recognition library, would eliminate password management overhead. A mobile companion application for Android, synchronized with the desktop system, would extend attendance recording and performance viewing capabilities to smartphones commonly available in field educational contexts. Further planned features include automated email notifications for low attendance alerts, integration with national academic databases such as the Academic Bank of Credits (ABC) under the National Education Policy 2020 framework, and multi-language support for regional language interfaces catering to diverse institutional contexts across India.

VII. CONCLUSIONS

This paper has presented Academic Growth Tracker Pro, a comprehensive desktop-based academic management system that integrates student performance tracking, faculty management, attendance recording, analytical visualization, and automated PDF reporting within a single, locally deployable Python application. The system addresses a demonstrable gap in the available toolset for educational institutions that require robust data management capabilities without the infrastructure costs and complexity associated with enterprise ERP systems or web-based LMS platforms.

Experimental evaluation confirmed that the system meets or exceeds all defined performance benchmarks under realistic institutional dataset scales, with sub-second analytical query response times, automated report generation under two seconds, and a lightweight memory footprint compatible with mid-range institutional hardware. Pilot user evaluation yielded mean satisfaction scores exceeding 4.1/5 across all assessed dimensions.

The modular architecture and open-source technology stack position Academic Growth Tracker Pro as a scalable foundation for ongoing enhancement.

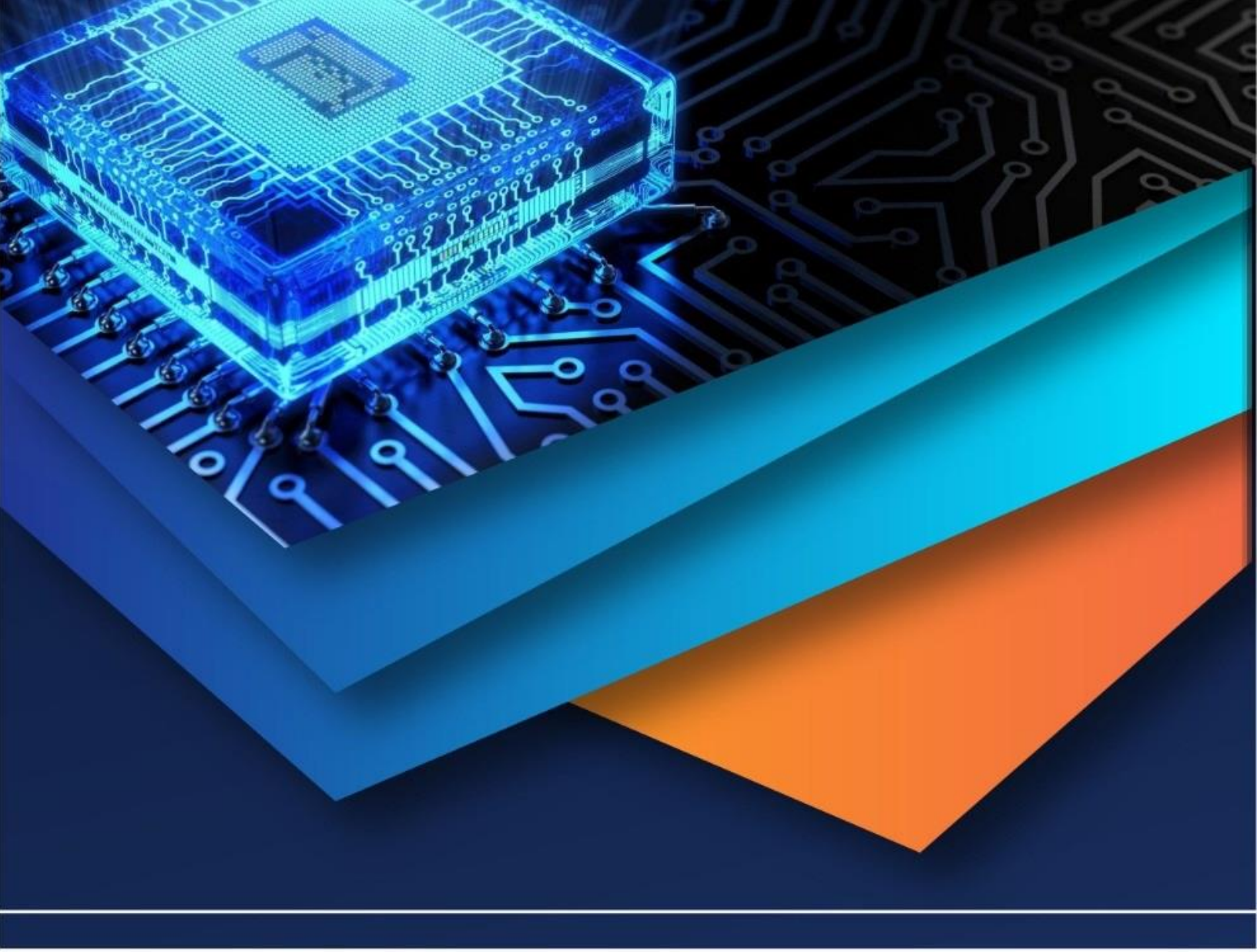
Planned extensions targeting cloud synchronization, predictive analytics, and biometric identification will progressively advance the system toward a comprehensive intelligent academic management platform. The work demonstrates the continued relevance and practical value of desktop-based educational software as a complement to, and in some contexts a superior alternative to, web-dependent institutional management solutions.

VIII. ACKNOWLEDGMENT

The authors express sincere gratitude to the Department of Computer Engineering at Bharat College of Engineering for providing laboratory resources and institutional support during the development and evaluation phases of this project. The authors also thank the participating faculty members and administrative staff who contributed their time and expertise to the pilot evaluation study.

REFERENCES

[1] T. R. Guskey, "Closing achievement gaps: Revisiting Benjamin S. Bloom's Learning for Mastery," Journal of Advanced Academics, vol. 19, no. 1, pp. 8–31, Nov. 2007.
[2] C. Romero and S. Ventura, "Educational data mining: A review of the state of the art," IEEE Transactions on Systems, Man, and Cybernetics, vol. 40, no. 6, pp. 601–618, Nov. 2010.
[3] M. D. Coates, "Moodle: Usability evaluation of an open source course management system," in Proc. 20th Annual ACM SIGCSE Conf. Innovation and Technology in Computer Science Education, 2015, pp. 327–332.
[4] A. Kumar and R. Singh, "QR code based attendance management system using Android mobile application," International Journal of Computer Applications, vol. 140, no. 8, pp. 1–6, Apr. 2016.
[5] P. Sharma, M.Agarwal, and N. Rawat, "Design and implementation of a lightweight academic portal using Python and Tkinter for resource-constrained educational institutions," in Proc. 3rd International Conf. on Computing, Communication and Security, 2019, pp. 1–10.
[6] R. K. Gupta and S. Mehta, "Performance analysis of SQLite as an embedded database for educational management applications," International Journal of Database Theory and Application, vol. 9, no. 4, pp. 71–82, 2016.
[7] V. Balachandran, S. Krishnamurthy, and T. Vijayan, "Impact of graphical analytics dashboards on faculty decision-making response time in academic environments," Computers & Education, vol. 82, pp. 135–148, Mar. 2015.
[8] A. Patil and S. Joshi, "Automated academic performance report generation using ReportLab for higher education institutions," International Journal of Advanced Research in Computer Science, vol. 8, no. 5, pp. 2020–2025, May 2017.
[9] Python Software Foundation, "Python 3.9 documentation," [Online]. Available: https://docs.python.org/3.9/
[10] The Tkinter Documentation Project, "Tkinter 8.6 widget reference," [Online]. Available: https://docs.python.org/3/library/tkinter.html
[11] ReportLab Group, ReportLab: PDF Library for Python, Version 3.6. London, UK: ReportLab Inc., 2021.
[12] J. D. Hunter, "Matplotlib: A 2D graphics environment," Computing in Science & Engineering, vol. 9, no. 3, pp. 90–95, May 2007.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY