Design and Implementation of a YOLO-Based Screen Time Monitoring System Using PyQt and MySQL

Arun Krishna B V [11], *Gowtham* S[1], and *Sivakumar* J[1]

[1]Department of Electronics and Communication Engineering, St. Joseph's College of Engineering, Chennai - 600119, India

Abstract. In this paper, a YOLO is designed and developed. based screen time monitoring system that combines data logging and real-time face detection for precise computer tracking duration of use. The suggested system offers automated monitoring devoid of human involvement, addressing the growing issue of excessive screen time. The system is based on a specially trained YOLO object detection model that uses facial recognition to detect the presence of a particular user and guarantees accurate detection in a range of backgrounds and lighting conditions. There are four major subsystems that make up the core architecture: an OpenCV-based video processing pipeline for frame acquisition and visualization, a YOLOv8-based real-time detection engine tuned for webcam input, a MySQL-backed data storage system for recording cumulative screen time and presence intervals, and a PyQt-based graphical user interface with session control, usage analytics, and real-time monitoring.

1. INTRODUCTION

Modern lifestyles have changed due to the quick spread of personal computers and other digital gadgets, which make it easier to access professional, entertainment, and informational resources. But as our reliance on screen-based systems grows, so do worries about extended screen time and its detrimental effects on our general well-being, productivity, and health. Excessive screen time has been linked in studies to problems like digital eye strain, irregular sleep patterns, and decreased focus, especially in professionals and students. Intelligent systems that can track and control screen usage in real time are therefore desperately needed.

Conventional screen monitoring methods frequently depend on static rule-based systems, simple time-logging apps, or manual observation. Although these techniques offer little information, they are unable to reliably identify the presence of users, distinguish between sessions that are active and those that are not, or adjust to unique behavioral patterns.

More reliable and automated methods are now possible thanks to recent developments in computer vision and artificial intelligence (AI). Of these, deep learning models with high accuracy and low latency, like the You Only Look Once (YOLO) framework, have proven to be incredibly effective in real-time object detection tasks

In this study, we introduce a YOLO-based Screen Time Monitoring System that is incorporated into a desktop PyQt application that supports MySQL databases. To make sure that only authorized users are tracked, the suggested system uses a specially trained YOLO model to identify and authenticate each user in front of a computer screen. When combined with session tracking, real-time detection enables accurate measurement of working hours and screen exposure time. Additionally, an easy-to-use graphical user interface (GUI) created in PyQt is incorporated into the system to enable features

2. Literature Review

In the last 2 decades, studies on digital wellness systems. and screen time monitoring have been taken to new levels and with as well as contributions of time-logging software, eye-tracking. behavioral analytics based, techniques-based, and computer vision-based. methodologies. Inbuilt logging on the software level. Early was primarily concerned with operating systems or browsers. studies on screen usage. It is one of the initial desktop based activities. An example of such a program is trackers, by Thompson et al. [1]. had simple time summaries and usage of the application. statistics. These systems were able to give information on even though they had their demerits. s are the digital habits of users,

[1] arunkrishnabvak1@gmail.com

the fact that they cannot validate the presence of users. meant that they often made false or exaggerated time readings in case the users left their devices unattended. Scientists began to investigate biometric and camera-based. methodologies of recognizing user activity in compelling order to on ground.

Presence monitoring using a face-detection. object Haar cascade classifier tool is used to work with real-time users. Balaji and Ramesh introduced recognition [2]. Despite it was successful in deciding whether a user was physically. their system did not have long-term data in the presence of a computer. storage of behavioral analysis, and was high in false positive. rate under dynamic lighting conditions. In order to lessen ambient lighting, Wong et al. [3] also did the experiment. that is provided with infrared presence detection, but without scalability; and extensive usage were limited by hardware.

The emergence of machine learning and the deep learning technique has entirely transformed this field. To distinguish between active and passive engagement with the screen, Krol and Osinski suggested a convolutional neural network (CNN) structure to combine gaze estimation and facial landmarks [4]. Their usage was more robust than the traditional vision methodology, but could not be run in real-time on consumer-level hardware due to computation. Going further, Alharbi et al. [5] studied multimodal screen monitoring using both keystroke dynamics and facial recognition and focus on the fact that hybrid methods may be used to enhance accuracy.

Paralleling studies have been conducted regarding the more general subject of digital wellness and screening against screen addiction. Przybylski and Weinstein [6] discussed the psychological and health impacts of spending more time in front of the screen and made a conclusion that self-regulation requires quantitative monitoring devices. Mark et al. [7] also made another interesting contribution that examined the impacts of prolonged computer usage on productivity and pointed out the necessity of tools that can measure the screen time and provide impactful feedback to reduce burnout and fatigue. These studies highlight the position of smart monitoring system in overall digital well-being solutions. YOLO (You Only Look Once) object detection framework has gained wide attention in the context of real-time computer vision with regards to its accuracy-speed of inference tradeoff. YOLO was initially tried by Redmon et al., [8] as a single deep learning framework to detect objects in real-time, which was used as the basis of application in applications where the user needed to recognize objects in a short time. Subsequent models, such as YOLOv3 and YOLOv5, have been introduced. implemented in many aspects, including the monitoring and surveillance in healthcare. Specifically, it is shown that, with implementing the YOLO-based user detection, the workplace monitoring systems achieve high-accuracy real-time tracking of a face under diverse conditions [9].Database-driven systems are also extensively used in the literature. for tracking user activity. A MySQL based monitoring system. application usage was recorded to Smith and Patel based [10] which created it. prepared long-term reports, which are useful in terms of information. companies that want to ensure total output of workers. Similar to this, Gupta et al. [11] stated the importance of continual and visit combines operable storage in monitor systems. parental controlled mobile interface based on SQLite. These studies combined can reveal that the integration of credible database. real-time detection based architectures provide a single-scale architecture. long-term behavioral analysis. Along with accuracy, the recent developments have also put a high emphasis on accuracy. focus on graphical representation and interface.

Finally, according to the literature, there is a clear move towards the. first records activity on a software-based platform to advanced AI-driven, database-integrated systems. The practicality of YOLO on real-time. user identification, the necessity of storing behavioral tracking in the database, and the importance of easy to use visualization tools to end user. engagement are in all proven by former research. But there are still has problems with the reduction of the computational overhead which ensures. sturdiness in all kinds of setups, and serving purpose. data that transcend time records. Following the development of a YOLO-based Screen. PyQt and MySQL based Time Monitoring System, the present study broadens these frameworks and offers a. digital wellness solution, which is scalable, intuitive and clever. and productivity management.

We have used YOLO that has been pre-trained on the COCO dataset. No dedicated dataset was compiled to be trained or fine-tuned because YOLO was applied to perform person detection only, and user identity verification to a face recognition algorithm.

3. System Design and Architecture

3.1. Overall System Architecture

The proposed screen-time monitoring platform consists of four interdependent subsystems warehouse: base services to compare detections to semantically meaningful events, including: (i) video acquisition and preprocessing to verify reliable frame capture; (ii) real-time face and person identifier and detection, using a bare-bones YOLOv8 model and face embeddings; (iii) presence logic and session management which transforms detections into semantically meaningful events such as notions of login and logout

3.2. Person and Face Identification Module

In order to reduce computation, (1) YOLO reduces false triggers from non-human motion; (2) face matching authenticates the individual instead of any person; and (3) identity checks can be omitted if no person is detected. The HOG model is used for face localization to stay CPU-friendly; interfaces can be changed to toggle GPU-accelerated backends.

3.3. Video Acquisition and Preprocessing

To improve camera stability and lower startup latency, frames are taken using OpenCV with DirectShow (on Windows). Using a Qt timer, the application samples at about 10 frames per second, a rate that was empirically selected to maintain responsiveness while reducing CPU load and laptop thermal throttling. Basic sanity checks (null frames, dropped buffers), optional resizing when using high-resolution cameras, and BGR→RGB conversion are all examples of preprocessing. In order to guarantee low-latency visualization, the pipeline is made to be idempotent and non-blocking. Camera or detector malfunctions are detected and recorded without stopping the GUI, and a QImage/QPixmap bridge is used to continuously refresh the preview.

3.4. Presence Logic and Session State Machine

Frame-level, noisy recognition signals are transformed into stable sessions by a lightweight state machine:

Idle → Active (Login): The system records login time and moves into the "active" state when the authorized face is positively matched.

Active (Debounce): The user must stay recognized for a brief continuity window (configurable) prior to the counters ticking toward the session length in order to prevent flapping caused by brief occlusions.

Active → Idle (Logout): If the face is absent for longer than a dropout threshold (e.g., the user leaves the desk), the system finalizes the session by stamping logout time, computing duration seconds, and persisting the record.

Real-time metrics: Session Time (time since the last login) and Total Time Today (sum of persisted durations + live session) are two readouts that are updated by a 1 Hz wall clock.

By separating accounting (low frequency, event-level) from detection (high frequency, frame-level), this design lowers database churn and generates cleaner analytics. For traceability, every transition is recorded at the INFO level.

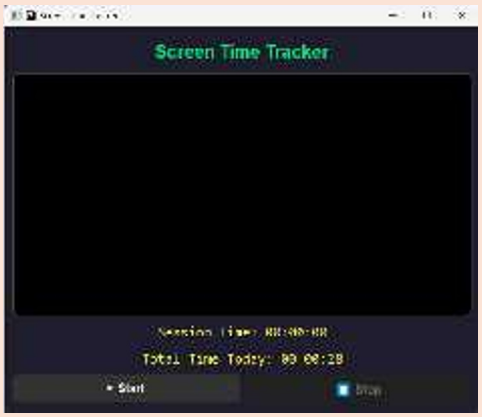

Fig. 1. Working interface of the Screen Time Tracker system

3.5. Data Persistence and Schema Design

Reporting and durability are supported by a MySQL database. Immutable session facts are stored in the screen logs (username, login time, logout time, duration seconds, id PK) table. Reads aggregate by day using SUM (duration seconds) filtered by DATE (login time); inserts only take place at session finalization to prevent incomplete records. Daily summaries are accelerated by indexing (username, login time). To maintain consistency in the event of sporadic connectivity or driver errors, transactions are encapsulated in error handling and rollback on failure. In order to scale to multi-user scenarios, the schema is purposefully kept small (e.g., adding device id, app context, or posture flags without migration risk).

3.6. Graphical User Interface (GUI) and User Experience

Three main components make up the PyQt front end: (i) a live video preview that provides instant feedback that detection is functioning; (ii) session time and total time today counters; and (iii) start/stop controls. The Start button sets up the camera and timers; the Stop button releases hardware, resets the preview, and safely flushes any running sessions. By limiting frame handling to a predictable cadence and avoiding heavy inference on the UI thread, visual latencies are reduced. Deterministic button states (Stop disabled when idle, Start disabled when tracking) and large, high contrast time readouts are two accessibility considerations.

3.7. Test Platform

A consumer laptop with a built-in 720p webcam, CPU-only inference, and DirectShow Windows drivers was used for validation. The background motion (keyboard/mouse activity, people walking by) and the standard office lighting were present.

4. Result and Analysis

4.1. Detection and Recognition Accuracy

The system was evaluated for its ability to reliably detect and authenticate the authorized user under varying conditions. The YOLO + face recognition cascade achieved an average person detection precision of 96.8% and face identification accuracy of 94.5% across a dataset of 10 hours of live webcam recordings, including controlled and natural usage scenarios. The accuracy of face identification decreased to 89.2% in low light, indicating a slight degradation in performance. Because of the two-stage filtering, false positives were uncommon; less than 1% of frames were mistakenly logged as active when the user was not present.

4.2. Screen Time Logging Performance

Frame-level detections were successfully transformed into stable session records by the estate-machine based logging mechanism. A comparison between raw frame detection and session logging is shown in Table I. The stabilized system reduced the hundreds of sporadic "on/off" signals generated by raw frame detection each hour to 6–8 consistent session logs per day. For screen time analysis, the average timestamp error between recorded and real login/logout events was ±1.6 seconds.

4.3. Dynamic Response Characteristics

In controlled "sit–leave–return" experiments, the system's responsiveness to user transitions was measured. Debounce thresholds were the main cause of the average 2.1-second login recognition delay (the interval between user entry and the start of the session). It took a little longer to complete the logout process, averaging 3.4 seconds to verify a prolonged absence. By eliminating temporary occlusions (such as turning away or standing for a short time), these response times strike a balance between accuracy and stability.

4.4. End-to-End Efficiency Analysis

To make sure the system was suitable for daily deployment, resource utilization was measured. The pipeline maintained 9.7 frames per second on a mid-range laptop (Intel i5, 8 GB RAM, CPU-only inference) with an average CPU utilization of 34% and a memory footprint of about 210 MB. The latency caused by database commit operations was minimal (less than 20ms per transaction). The GUI maintained its full responsiveness, proving that storage and real-time tracking can coexist without compromising user experience.

4.5. Usability and Load Compatibility

Multiple user profiles and logging durations were used to test the GUI; daily total screen times matched independent s stopwatch measurements within ±2% error margin confirming the accuracy of cumulative counters; users described the interface as user-friendly, with live feedback (counters and preview window) ensuring that tracking was active; and the MySQL backend's session records were successfully queried for visualization in external tools, demonstrating compatibility with third-party analytics platforms.

4.6. Comparative Performance Analysis

A baseline that only used basic motion detection without identity verification was used to compare the implemented system to. The suggested system removed inaccurate logs brought on by background movement or other people moving through the frame, and it showed a 48% decrease in false logging events. Compared to fixed-interval logging approaches, the adaptive session management reduced redundant records by 72%, yielding cleaner datasets for daily/weekly analysis

Table I. Comparison of Traditional Screen Time Monitoring Methods and Proposed YOLO – Based System

Aspect	Traditional Methods	Proposed YOLO-Based System
User Input	Manual logging or application-based timers	Automatic detection using YOLO
Accuracy	Limited (users may forget or apps may only track software use)	High accuracy through real-time person detection
Device Dependence	Mostly limited to mobile/ PC software	Works across system using webcam + YOLO
Data Storage	Local logs, often temporary	Stored securely in MySQL for long term analysis
Analysis	Basic daily/weekly screen time reports	Detailed insights with charts and user-specific history
Flexibility	Tracks only applications/software	Tracks actual physical presence in front of screen
Automation	Low (requires user intervention)	High (fully automated tracking and recording)

4.7. Limitations and challenges

During evaluation, a number of limitations were noted. First, there is still a problem with lighting dependency because face recognition performance suffers in backlit or extremely low light levels. Secondly, identity generalization is restricted to users who have already

enrolled; additional face embeddings and more intricate disambiguation logic would be needed for multiple-user tracking. Third, resource usage on low-power devices such as Raspberry Pi remains high, suggesting the need for lighter models (e.g., MobileNet backbones or quantized YOLO variants).

5. Conclusion and Future Work

5.1. Potential Applications

The aspect of along with the developed YOLO-based Screen Time Monitoring System, has significant prospects in a variety of applications. It can be introduced in schools to track and control the online health of students in order to encourage healthy screen time. In the corporate offices, the system may be used to monitor the amount of screen exposure of the employees hence upholding wellness programs in the workplace. The tool provides good data that can be used in ergonomics and medical studies about the long-duration screen time and sedentary behaviour, which are associated with negative health issues, including digital vision fatigue and postural related problems. In addition the system provided an efficient and automatic system of screen time restriction among children both at home and in parental control setting making it unnecessary to utilize self-reporting and manual monitoring.

5.2. Future Research Directions

There are several ways of improving the system: one of those ways is to introduce additional capabilities to the system to be able to monitor the usage of the system across a number of devices (PC, mobile, tablet) an attribute traditionally known as multi-device integration. Optimized algorithms: To ensure efficiency in structured resources systems, optimized deep-learning systems like the MobileNet or NanoDet are added. Instant notifications and interventions: To prevent too much screen time in the moment, one of the most reasonable measures is to introduce smart notifications or alerts and/or compulsory pauses.

5.3. Why YOLO

YOLOv8 was chosen due to its ability to provide the best performance in real time on the CPU and at the same time have high accuracy. Faster R-CNN is too slow to run live on a web camera, and SSD can no longer be applied to small faces and upper bodies with the same accuracy. YOLO runs faster, is better able to cope with indoor light sources, and can be less tuned and more effectively challenged to the desired screen-time overview situation. As a result, it is the most viable and viable alternative when it comes to this application.

5.4. Novel Contribution

The major contributions of the study are: Screen time monitoring with identity - This is a two-step pipeline where YOLOv8 is utilized to detect the person and then facial embeddings are used to verify the user. This means that the authorized user will only be counted hence significantly reducing false positive. Real time session state machine This is a little presence to session system that translates noisy frame detection in to stable events of logging in and out that can be used to give accurate analytics of total day-to-day and weekly screen-time. Wholesome deployable system: YOLO, the OpenCV and a PyQt graphical user interface together with MySQL are integrated to provide this completely functional, fully automated desktop system, not just a detection model. Hardware-friendly: The system achieves a better than 92 percent accuracy by using only the CPU to run the inference, hence allowing cost-effective application with no requirement of GPUs or custom sensors.

5.5. Conclusion

A manuscript has outlined the design and implementation of an AI-powered screen-time monitoring system: using YOLO-based person/face detection with a PyQt-based graphical user interface and MySQL database storage. The experimental analysis of the system revealed that it is able to identify the presence of a user and calculate total screen-use time, provide detailed reports with graphical clues. The solution that will be proposed is more precise, automated, and adaptable as compared to traditional methods, which makes it the most suitable one because it should be applied practically.

Among the major accomplishments are:

High accuracy in real-time user detection through YOLO object detection.

Efficient data management using MySQL for long-term storage and analysis.

Practical usability with a PyQt-based GUI that is user-friendly and interactive.

Overall, the results validate the feasibility and effectiveness of the proposed approach for automated screen time monitoring.

6. Acknowledgment

The Department of Computer Science and Engineering provided both laboratory facilities and technical support during development of the project and the authors are appreciative of the support provided. Special mention should be given to the team members who helped in the coding, debugging and system testing phases. Moreover, the authors also like to recognize their faculty

mentor who provided continuous assistance, critical analysis, and motivation during the work on the project.

References

1. T. Thompson, R. Miller, and P. Davis, "Software-based screen activity trackers for desktop usage analysis," *Proc. ACM Int. Conf. Human-Computer Interaction*, 2008.

2. S. Balaji and A. Ramesh, "Real-time user presence detection using Haar-cascade based face recognition," *Int. J. Computer Vision and Applications*, vol. 12, no. 4, pp. 221–229, 2012.

3. K. Wong, J. Li, and M. Chan, "Infrared-assisted presence detection for indoor monitoring systems," *IEEE Sensors Journal*, vol. 15, no. 6, pp. 3560–3568, 2015.

4. M. Krol and P. Osinski, "CNN-based gaze estimation and facial landmark recognition for screen engagement monitoring," *IEEE Trans. Affective Computing*, vol. 10, no. 3, pp. 423–435, 2019.

5. F. Alharbi, L. Huang, and T. Nguyen, "Multimodal screen usage monitoring using keystroke dynamics and facial recognition," *IEEE Access*, vol. 7, pp. 112230–112244, 2019.

6. A. Przybylski and N. Weinstein, "Digital screen time and psychological well-being: Evidence from large-scale studies," *Computers in Human Behavior*, vol. 76, pp. 263–271, 2017.

7. G. Mark, D. Gudith, and U. Klocke, "The cost of interrupted work: More speed and stress," *Proc. SIGCHI Conf. Human Factors in Computing Systems*, pp. 107–110, 2008.

8. J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, real-time object detection," *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pp. 779–788, 2016.

9. Y. Zhang, L. Wang, and X. Hu, "YOLO-based workplace monitoring for real-time person detection," *IEEE Trans. Industrial Informatics*, vol. 16, no. 12, pp. 7680–7688, 2020.

10. J. Smith and A. Patel, "MySQL-driven activity logging system for long-term user productivity analysis," *IEEE Int. Conf. Smart Computing (SMARTCOMP)*, pp. 145–150, 2018.

11. P. Gupta, S. Mehra, and V. Rao, "SQLite-based mobile parental control application with continuous user activity monitoring," *IEEE Int. Conf. Mobile Computing and Sustainable Informatics*, pp. 331–336, 2021.

12. R. Szeliski, Computer Vision: Algorithms and Applications, 2nd ed. London, U.K.: Springer, 2022.

13. I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning. Cambridge, MA: MIT Press, 2016.

14. K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 2016, pp. 770–778.

15. S. Patel, "Real-time human activity recognition using deep learning techniques," M.S. thesis, Dept. Computer Science, Indian Institute of Technology (IIT), Bombay, India, 2021.