

HealthCare system for online learning students

EDUGuard

24-25J-134

Final Group Report

BSc (Hons) Degree in Information Technology

Specializing in Information Technology

Department of Computer Science

Sri Lanka Institute of Information Technology

Sri Lanka

April 2025

HealthCare system for online learning students

EDUGuard

24-25J-134

Final Group Report

BSc (Hons) Degree in Information Technology

Specializing in Information Technology

Department of Computer Science

Sri Lanka Institute of Information Technology


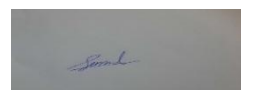
Sri Lanka

April 2025

Declaration

“I declare that this is my own work and this dissertation¹ does not incorporate without acknowledgement any material previously submitted for a Degree or Diploma in any other University or institute of higher learning and to the best of my knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgement is made in the text.

Also, I hereby grant to Sri Lanka Institute of Information Technology, the non-exclusive right to reproduce and distribute my dissertation, in whole or in part in print, electronic or other medium. I retain the right to use this content in whole or part in future works (such as articles or books).”

Name	Student Id	Signature
Arangallage C.M. A	IT21211850	
Polhengoda P.M.Y.Y. B	IT21263576	
W.R.S Virukshi	IT21212840	
Sandeeptha P.K. T	IT21312694	

The supervisor/s should certify the proposal report with the following declaration. The above candidates are carrying out research for the undergraduate Dissertation under my supervision.

Name of supervisor: Dr. Kapila Disanayake

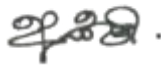
Name of co-supervisor: Ms. Akshi De Silva

Signature of the supervisor:



Date: 10.04.2025

Signature of the co-supervisor:



Date: 10.04.2025

Abstract

As online learning becomes increasingly prevalent, it brings with it a range of hidden health risk from eye strain and poor posture to chronic dehydration and elevated stress that often go unnoticed until they begin to disrupt students' academic performance and daily lives. Traditional wellness solutions, such as wearables or manual input systems, are often intrusive or impractical in educational settings. EduGuard offers a seamless, non-intrusive alternative designed specifically for remote learners. Leveraging just a standard webcam, the system monitors four key health indicators in real time: eye strain through blink rate and screen distance, stress via facial emotion recognition using deep learning, dehydration by detecting lip dryness, and posture through machine learning driven body alignment analysis. Integrated across desktop and mobile platforms, EduGuard provides timely alerts, actionable insights, and personalized wellness suggestions, transforming passive screen time into an active opportunity for health management. What makes EduGuard truly unique is its ability to blend into a student's routine without adding extra burden. It's like having a silent companion that gently reminds you to take a break, adjust your posture, or sip some water. In a world where academic pressure is high and self-care often takes a backseat, this tool offers a much-needed balance. By uniting computer vision, machine learning, and behavioral analysis, EduGuard not only enhances student well-being but also empowers healthier, more mindful digital learning experiences.

Keywords: *Real time health monitoring, Eye strain, Posture correction, Dehydration detection, Stress monitoring, Machine Learning, Deep Learning, Desktop application*

Dedication

This thesis is dedicated to my cherished family and loved ones, whose steadfast support, encouragement, and patience have been a source of strength throughout my academic journey. Their unwavering faith in me has been a constant source of encouragement and inspiration.

I express my profound thanks and dedicate this effort to my supervisor, Dr. Kapila Disanayake, my co-supervisor, Ms. Akshi De Silva, and my external supervisor, Dr. Jayan De Silva. Their essential direction, perceptive input, and expertise have been important in the effective completion of our research. Their mentorship has profoundly influenced the trajectory and caliber of this project while also greatly enhancing my academic and personal growth.

This thesis is dedicated to all individuals and professionals devoted to the advancement of mental health treatment. I sincerely hope this research significantly helps to the creation of accessible, AI-driven solutions that enhance emotional well-being and assist those in need.

ACKNOWLEDGEMENTS

Throughout the course of my research and the completion of this thesis, I would like to extend my heartfelt appreciation to all the individuals and organizations who have provided me with valuable assistance.

I would like to commence by expressing my profoundest appreciation to Dr. Kapila Disanayake, my supervisor, for his tireless support, invaluable insights, and expert guidance during the course of this research. The foundation and direction of this investigation have been significantly influenced by his mentorship. I am also grateful to Ms. Akshi De Silva, my co-supervisor, for her insightful feedback and encouragement, which significantly enhanced the refinement of the methodologies employed. I am equally appreciative of Dr. Jayan De Silva, my external supervisor, for his constructive critiques and professional guidance, which have substantially improved the quality and clarity of this work.

I am deeply grateful to my family for their unwavering support, encouragement, and faith in me, which have served as a steadfast source of fortitude. I am eternally appreciative of their forbearance and sacrifices, as their emotional support and comprehension were indispensable during difficult periods.

I would also like to express my gratitude to my colleagues and peers for their assistance, constructive dialogue, and constructive criticism during the research process. This endeavor was significantly enhanced by their collaboration and support.

This research was made possible by the provision of the requisite facilities, resources, and a stimulating academic environment by the Department of Computer Science, Faculty of Computing, at the Sri Lanka Institute of Information Technology. A special note of appreciation is due to them.

Lastly, I would like to extend my sincere gratitude to all the participants of this study for their valuable contributions, willingness, and time. Their participation was indispensable to the successful conclusion of this dissertation.

TABLE OF CONTENTS

Declaration	ii
Abstract	iii
Dedication	iv
ACKNOWLEDGEMENTS	v
LIST OF FIGURES	viii
LIST OF TABLES	ix
LIST OF ABBREVIATIONS	x
LIST OF APPENDICES	xi
1. INTRODUCTION	1
1.1 Background	1
1.2 Literature Review	4
2. OBJECTIVES	8
2.1 Main Objective	8
2.2 Sub Objectives	9
3. METHODOLOGY	12
3.1 Component break down	13
3.1.1 Component 1: Eye Strain Detection	13
3.1.2 Component 2: Stress Detection	14
3.1.3 Component 3: Dehydration Detection	16
3.1.4 Component 4: Posture Detection	23
3.2 System Overview Diagram	29
3.3 Mobile App	31
4 Implementation and testing	34
4.1 Software Implementation	34
Posture Detection Component	38
Deep Learning-Based YOLOv8 Posture Detection	40
Hybrid Model Approach for Real-Time Monitoring	41

4	Grant Chart	43
5	Results and Discussion	44
	5.1 Results.....	44
	Stress Detection Using Facial Emotion Recognition.....	44
	Posture Monitoring with Hybrid Model Technology.....	45
	Computer Vision Syndrome (CVS) Detection	45
	Dehydration Monitoring via Lip Dryness	46
	Desktop Alert System and Real-Time Feedback.....	47
	5.2 Discussion.....	51
6	CONCLUSION.....	55
7	REFERENCES.....	58
8	GLOSSARY	61
9	APPENDICES	63

LIST OF FIGURES

Figure 1: Hydration Component Architecture Diagram.....	17
Figure 2: Hydration Component Data Augmentation	20
Figure 3: Hydration Component Model Architecture	21
Figure 4: Hydration Component Model Training	21
Figure 5: Posture Component Architecture Diagram.....	24
Figure 6: Good Posture Data.....	27
Figure 7 : Bad Posture Data	28
Figure 8: Posture Detection Angle Extration Theory.....	29
Figure 9: System Diagram	29
Figure 10: Desktop App Login UI	35
Figure 11: Desktop App Component Starting UI.....	36
Figure 12: Desktop App User Profile Update UI.....	37
Figure 13: Desktop App Report UI	38
Figure 14 : Grant Chart	43
Figure 15: Posture Detection Component Results	49
Figure 16 : : Stress Detection Component Results	49
Figure 17: : Eye Strain Detection Component Results	50

LIST OF TABLES

Table 1: Model Accuracies.....	48
--------------------------------	----

LIST OF ABBREVIATIONS

Abbreviation	Full Form
BGR	Blue-Green-Red (colour format used in OpenCV)
CNN	Convolutional Neural Network
CVS	Computer Vision Syndrome
DL	Deep Learning
FPS	Frames Per Second
ML	Machine Learning
RGB	Red-Green-Blue (colour format used in image processing)
ROI	Region of Interest
TCP	Transmission Control Protocol
WPF	Windows Presentation Foundation
YOLO	You Only Look Once

LIST OF APPENDICES

Appendix	Title
Appendix A	Model Architectures (Stress, Posture, Eye Strain, and Dehydration Models)
Appendix B	User Interface Screens (Login, Profile, Report, and Component Start UI)
Appendix C	Dataset Composition (Lip Dryness, Posture Images)
Appendix D	System Architecture and Communication Setup
Appendix E	Sample Alerts and Notifications

1. INTRODUCTION

1.1 Background

The global transformation in education has ushered in a new era where learning can happen anywhere, anytime. With this rise in remote learning and online education platforms, millions of students have gained flexibility and accessibility never imagined before. But with this convenience comes a set of less visible, yet deeply impactful challenges those affecting the physical and mental health of learners. Extended screen time, poor sitting habits, and the lack of structured breaks have led to a range of health issues, from eye strain and fatigue to mental burnout and musculoskeletal discomfort. Students are silently enduring the physical toll of this digital shift. The need of the hour is not just digital literacy, but digital wellness. This is where EduGuard steps into an intelligent system designed to bridge the gap between learning and well-being.

EduGuard is an innovative health-monitoring system crafted specifically for the online learning community. It is not merely a tool, but a wellness companion that works silently in the background to ensure that students stay healthy while they study. Our mission is simple but powerful: to enable safe, healthy, and effective digital learning by integrating real-time health insights into daily study routines. What makes EduGuard unique is its non-intrusive and webcam-based design. It doesn't rely on wearables or external devices. Instead, using the power of computer vision and machine learning, it detects key health issues and proactively nudges users to adopt better habits.

The system targets four critical health areas: eye strain (Computer Vision Syndrome), stress, dehydration, and poor posture all of which are prevalent among digital learners today. Our research began with a simple yet far-reaching question: How can we protect the physical and mental health of students who spend long hours in front of screens? From eye discomfort and mental exhaustion to dry lips and back pain, students often ignore these symptoms until they evolve into chronic conditions.

Traditional health monitoring techniques are reactive, not proactive. Furthermore, wearable health devices are costly and often impractical for student populations.

The gap was clear: the world needed a smart, accessible, and affordable health support system tailored for the digital learning experience and EduGuard is the answer to that gap. Eyes are one of the most affected organs in a digital learning environment. EduGuard's first module focuses on detecting Computer Vision Syndrome (CVS)—a modern epidemic caused by prolonged screen exposure.

This module monitors:

- Blinking frequency
- Eye fatigue
- Face-to-screen distance
- Screen time

Using real-time webcam feeds and a trained eye-state detection model, the system identifies early signs of eye strain and notifies the user to take a break or adjust posture. It also compiles personalized reports to promote long-term eye health.

By providing timely feedback, this module encourages micro-behavioural changes that prevent the escalation of eye-related issues. Academic stress is often overlooked until it reaches a tipping point. The stress detection module of EduGuard leverages facial expression recognition to identify signs of emotional strain without requiring any sensors or manual inputs.

Trained on datasets with 7 core emotional states (happy, sad, angry, fear, surprise, disgust, neutral), and built on EfficientNetB0, this model classifies the user's emotional state in real-time. If stress indicators are high, EduGuard issues gentle prompts reminders to breathe, stretch, or take short breaks. Moreover, all data is processed privately and securely, ensuring both mental wellness and digital privacy. This module transforms passive stress into active well-being.

Hydration often slips under the radar during intensive screen sessions. Our third module addresses this through a novel approach lip dryness detection. By analysing subtle changes in lip texture and mouth behaviour, this module identifies signs of

dehydration. The system then provides hydration prompts like drinking water or using lip balm. It also tracks screen time to correlate dehydration risk with session duration.

Unlike traditional hydration reminders, EduGuard uses contextual and evidence-based cues, ensuring that alerts are timely, relevant, and personalized.

A poor sitting posture can cause long-term spinal issues and chronic pain. The posture detection module in EduGuard acts like a virtual physiotherapist.

Using webcam-based vision models trained to recognize body posture, the system classifies user alignment as “good” or “poor.” It also distinguishes between short-term slouches and prolonged postural issues, alerting the user only when necessary.

This module promotes physical awareness and corrects behaviour without being disruptive. It's built with adaptability in mind—designed to work in diverse settings like home, library, or shared spaces, and optimized for student comfort and long-term use.

To maximize accessibility, EduGuard isn't just a desktop application—it comes with mobile app integration.

- Users can receive real-time alerts on their phones.
- Weekly health summaries and trends help track long-term progress.
- Insights and recommendations can be customized based on learning habits.

The mobile sync bridges the gap between awareness and action. Whether the student is actively studying or reviewing their day, they're never disconnected from their wellness data. The interface is intuitive, data-friendly, and respects user privacy at every level. EduGuard is more than a project—it's a step toward redefining what it means to learn healthily in the 21st century. By combining real-time computer vision, artificial intelligence, and user-centered design, we're making self-care as easy as studying online. The project is designed to be scalable, adaptable, and affordable ensuring that every student, regardless of background, can benefit from better health

support. In the long term, EduGuard aims to evolve into a full-fledged digital wellness assistant, with capabilities for posture gamification, hydration goals, emotional journaling, and beyond. The journey has just begun but the destination is clear: a world where health and education go hand in hand.

1.2 Literature Review

The global transformation in education has ushered in a new era where learning can happen anywhere, anytime. With this rise in remote learning and online education platforms, millions of students have gained flexibility and accessibility never imagined before. But with this convenience comes a set of less visible, yet deeply impactful challenges those affecting the physical and mental health of learners. Extended screen time, poor sitting habits, and the lack of structured breaks have led to a range of health issues, from eye strain and fatigue to mental burnout and musculoskeletal discomfort. Students are silently enduring the physical toll of this digital shift. The need of the hour is not just digital literacy, but digital wellness. This is where EduGuard steps into an intelligent system designed to bridge the gap between learning and well-being.

EduGuard is an innovative health-monitoring system crafted specifically for the online learning community. It is not merely a tool, but a wellness companion that works silently in the background to ensure that students stay healthy while they study. Our mission is simple but powerful: to enable safe, healthy, and effective digital learning by integrating real-time health insights into daily study routines. What makes EduGuard unique is its non-intrusive and webcam-based design. It doesn't rely on wearables or external devices. Instead, using the power of computer vision and machine learning, it detects key health issues and proactively nudges users to adopt better habits.

The system targets four critical health areas: eye strain (Computer Vision Syndrome), stress, dehydration, and poor posture all of which are prevalent among digital learners today. Our research began with a simple yet far-reaching question: How can

we protect the physical and mental health of students who spend long hours in front of screens? From eye discomfort and mental exhaustion to dry lips and back pain, students often ignore these symptoms until they evolve into chronic conditions.

Traditional health monitoring techniques are reactive, not proactive. Furthermore, wearable health devices are costly and often impractical for student populations.

The gap was clear: the world needed a smart, accessible, and affordable health support system tailored for the digital learning experience and EduGuard is the answer to that gap. Eyes are one of the most affected organs in a digital learning environment. EduGuard's first module focuses on detecting Computer Vision Syndrome (CVS)—a modern epidemic caused by prolonged screen exposure.

This module monitors:

- Blinking frequency
- Eye fatigue
- Face-to-screen distance
- Screen time

Using real-time webcam feeds and a trained eye-state detection model, the system identifies early signs of eye strain and notifies the user to take a break or adjust posture. It also compiles personalized reports to promote long-term eye health.

By providing timely feedback, this module encourages micro-behavioural changes that prevent the escalation of eye-related issues. Academic stress is often overlooked until it reaches a tipping point. The stress detection module of EduGuard leverages facial expression recognition to identify signs of emotional strain without requiring any sensors or manual inputs.

Trained on datasets with 7 core emotional states (happy, sad, angry, fear, surprise, disgust, neutral), and built on EfficientNetB0, this model classifies the user's emotional state in real-time. If stress indicators are high, EduGuard issues gentle prompts reminders to breathe, stretch, or take short breaks. Moreover, all data is

processed privately and securely, ensuring both mental wellness and digital privacy. This module transforms passive stress into active well-being.

Hydration often slips under the radar during intensive screen sessions. Our third module addresses this through a novel approach lip dryness detection. By analysing subtle changes in lip texture and mouth behaviour, this module identifies signs of dehydration. The system then provides hydration prompts like drinking water or using lip balm. It also tracks screen time to correlate dehydration risk with session duration.

Unlike traditional hydration reminders, EduGuard uses contextual and evidence-based cues, ensuring that alerts are timely, relevant, and personalized.

A poor sitting posture can cause long-term spinal issues and chronic pain. The posture detection module in EduGuard acts like a virtual physiotherapist.

Using webcam-based vision models trained to recognize body posture, the system classifies user alignment as “good” or “poor.” It also distinguishes between short-term slouches and prolonged postural issues, alerting the user only when necessary.

This module promotes physical awareness and corrects behaviour without being disruptive. It's built with adaptability in mind—designed to work in diverse settings like home, library, or shared spaces, and optimized for student comfort and long-term use.

To maximize accessibility, EduGuard isn't just a desktop application—it comes with mobile app integration.

- Users can receive real-time alerts on their phones.
- Weekly health summaries and trends help track long-term progress.
- Insights and recommendations can be customized based on learning habits.

The mobile sync bridges the gap between awareness and action. Whether the student is actively studying or reviewing their day, they're never disconnected from their wellness data. The interface is intuitive, data-friendly, and respects user privacy at every level. EduGuard is more than a project—it's a step toward redefining what it

means to learn healthily in the 21st century. By combining real-time computer vision, artificial intelligence, and user-centered design, we're making self-care as easy as studying online. The project is designed to be scalable, adaptable, and affordable ensuring that every student, regardless of background, can benefit from better health support. In the long term, EduGuard aims to evolve into a full-fledged digital wellness assistant, with capabilities for posture gamification, hydration goals, emotional journaling, and beyond. The journey has just begun but the destination is clear: a world where health and education go hand in hand.

2. OBJECTIVES

2.1 Main Objective

- To design and develop EDUGuard, an intelligent, real-time health monitoring system that detects physical and mental health risks among online learners using only a standard webcam, without the need for wearable devices or external sensors.

The primary goal of this research is to create EDUGuard, a non-invasive, AI-powered desktop and mobile application that monitors students' health while they engage in digital learning. Unlike traditional sensor-based solutions that rely on expensive hardware or wearable technology, EDUGuard is designed to function entirely through a standard laptop or desktop webcam, making it an affordable and scalable solution for students globally.

The system addresses four major health concerns frequently observed in online learners:

- Poor Posture Detection** – Using computer vision and machine learning algorithms, the system tracks a student's body alignment, specifically analysing neck, shoulder, and upper back positioning. It identifies incorrect postures in real-time and prompts the user with alerts to encourage healthy sitting behaviour and prevent long-term musculoskeletal disorders.
- Eye Strain Detection (Computer Vision Syndrome - CVS)** – By analysing blink rates and eye movement patterns through the webcam, EDUGuard monitors visual fatigue. If symptoms of CVS are detected, such as decreased blink rate or prolonged screen staring, it provides the user with break reminders and relaxation tips.
- Emotional Stress Recognition** – Facial expression analysis is performed using AI-driven emotion recognition models. This helps determine the user's emotional state, such as signs of stress, fatigue, or frustration. The

system then recommends stress-relieving exercises or relaxation techniques based on the user's mood.

- iv. **Dehydration Indication** – The system checks for signs of dry lips or facial dryness using visual cues. Although subtle, these indicators can reflect insufficient hydration. EDUGuard uses this information to send gentle reminders to the user to drink water.

Overall, EDUGuard aims to enhance student well-being during long online learning sessions by providing real-time alerts and actionable recommendations. By integrating AI with real-world student health needs, the system is intended to be both practical and transformative for the remote education landscape.

2.2 Sub Objectives

➤ Detect Eye Strain via Blink Rate and Screen Distance

With students spending hours in front of screens, eye strain becomes a daily issue often leading to headaches, blurry vision, and fatigue. This component focuses on preventing Computer Vision Syndrome (CVS) by monitoring eye blinking patterns and how close students sit to their screens.

Sub-objectives include:

- Tracking blink frequency to identify signs of reduced blinking (a common symptom of CVS).
- Measuring screen distance using face detection to ensure users aren't sitting too close.
- Monitoring screen time duration to understand how long a student has been focused on the screen.
- Training an eye state detection model using image data to detect signs of fatigue in real time.

- Providing gentle alerts when it's time to blink more, adjust the sitting distance, or take a short break.

➤ **Monitor Emotional States and Stress Levels Using Facial Recognition**

Stress is a silent burden for many students in digital classrooms. This component uses facial emotion recognition to pick up subtle expressions that may indicate mental strain or emotional fatigue.

Sub-objectives include:

- Detecting facial cues such as frowns, furrowed brows, or tense jawlines common stress indicators.
- Building and training a CNN-based model (EfficientNetB0) to classify emotions like sadness, anger, fear, or surprise.
- Integrating the model into a live video feed, ensuring real-time emotion detection during study sessions.
- Alerting the student when signs of high stress are detected, along with tips like breathing exercises or short mindfulness breaks.
- Evaluating the system's effectiveness through user feedback and accuracy testing.

3. Identify Signs of Dehydration Through Lip Dryness

Hydration is essential for focus, yet it's easy to forget to drink water while immersed in online work. This component aims to detect early visual signs of dehydration, particularly lip dryness, using webcam input.

Sub-objectives include:

- Detecting the user's lips through facial region segmentation.

- Monitoring for lip dryness or changes in colour/texture using trained CNN models.
- Recording session duration and mouth behaviour, linking longer durations to potential dehydration.
- Sending hydration reminders, like “Time to sip some water!” or “Take a hydration break.”
- Allowing users to customize alert frequency, depending on their preferences and environment.

4. Evaluate and Alert Poor Posture Habits During Screen Time

Slouching or leaning into the screen may seem harmless, but over time, poor posture can lead to back, neck, and shoulder problems. This module focuses on maintaining healthy posture habits.

Sub-objectives include:

- Using webcam-based pose detection to classify posture as “good” or “bad” in real-time.
- Distinguishing between short-term movement and sustained poor posture, so the system doesn’t over-alert.
- Tracking user presence and session duration to better understand posture trends.
- Storing posture data for progress tracking and long-term feedback.
- Delivering posture correction alerts only, when necessary, to promote a supportive not annoying experience.

3. METHODOLOGY

The development of the EDUGuard Desktop Application followed a structured and modular approach, aimed at creating a smart, health-focused monitoring system for desktop users, especially students engaged in prolonged digital learning. The system integrates several independent monitoring modules such as posture detection, hydration tracking, stress monitoring, and eye strain analysis (CVS), each powered by machine learning (ML) and deep learning models.

The frontend of the application was built using C# with WPF to provide an interactive and user-friendly interface. This interface manages user sessions, displays health alerts, shows historical predictions in graphical form, and allows real-time control of monitoring models. The backend components were developed using Python, leveraging specialized ML/DL models trained for specific tasks. For example, the posture and hydration modules utilize MediaPipe and OpenCV for landmark detection and visual analysis, while models like SVM and Naïve Bayes are used for classification tasks such as stress and eye strain detection.

A core feature of the methodology was the use of TCP socket communication between the desktop app and the Python backend. A local socket server streams webcam frames from the desktop to the model scripts in real time. Each model processes these frames and returns insights, such as posture quality, lip dryness, or blink rate on the fly. MongoDB was used as the primary storage solution to save prediction results under progress report entries, allowing for historical tracking and report generation.

To help users visualize their well-being patterns, the application includes integrated charts for each module, which display trends over time using real-time and historical data. These visualizations allow users to easily spot issues like frequent bad posture, high stress levels, or inconsistent hydration. Timers in the C# application were configured to regularly fetch the most recent predictions from the database and analyse them. If any of the monitored conditions exceeded predefined health risk thresholds (e.g., consistent bad posture or abnormal blinking), a notification was

triggered to alert the user immediately. Each model ran independently and could be started or stopped based on the user's need, enabling modular deployment.

3.1 Component break down

3.1.1 Component 1: Eye Strain Detection

In today's world of digital learning, many students find themselves staring at screens for hours on end. The result? Burning eyes, blurred vision, frequent headaches classic symptoms of Computer Vision Syndrome (CVS). To address this growing concern, our team designed a real-time, non-intrusive monitoring system that not only understands these struggles but actively works to reduce them using smart technology.

We began by listening to observing how students interact with their screens during long study sessions. Most weren't aware when they stopped blinking enough or sat too close to the screen. Our goal was to create a digital companion that would watch out for them, just like a caring teacher would, but silently and intelligently in the background. The system uses a regular webcam no fancy hardware or wearables. It continuously captures frames of the student's face during learning. This live video feed becomes the foundation for everything that follows.

Using OpenCV and MediaPipe, we isolate the region of interest (ROI) mainly the eyes. Then, we apply a custom-trained Convolutional Neural Network (CNN) to analyse whether the eyes are open or closed. By counting the number of blinks per minute and the duration of eye closures, we get a real-time measure of blink patterns and signs of strain.

- Too few blinks? Risk of dry eyes.
- Too many long closures? Possibly fatigue setting in.

These tiny signals, often invisible to the user, become our key to detecting CVS. We also estimate how close the student is sitting to the screen. Using inter-pupillary distance and face landmark detection, the system can tell if a student is sitting too close for prolonged periods a major contributor to CVS.

A background timer keeps track of how long the user has been staring at the screen without a break. When screen time exceeds healthy limits, the system gently reminds them to rest their eyes following the 20-20-20 rule (every 20 minutes, look at something 20 feet away for 20 seconds). All this real-time data blinking rate, eye fatigue, screen distance, and screen time is passed into our trained machine learning model. This model outputs a CVS risk level (low, moderate, high), helping the system decide when to alert the user. When something seems off, the system doesn't alarm the user it nudges them with a gentle desktop alert or mobile app notification. For example: Every prediction is stored in MongoDB, forming a timeline of eye health. Users can view easy-to-understand charts showing trends when they're most fatigued, or how often they sit too close. These visuals help them reflect on their habits and improve them over time. The CVS module runs independently and can be started or paused based on the student's choice. Notifications and sensitivity levels can be customized too because every learner is unique, and so are their needs.

3.1.2 Component 2: Stress Detection

In the development of EduGuard's stress detection module, our goal was to blend technology and empathy creating a system that doesn't just recognize stress in students but actively helps them manage it in a non-intrusive way. The solution was designed to work with something every student already has: a webcam. No need for wearable sensors or additional hardware just a smart system that watches over you, quietly, in the background. We started by identifying how stress manifests in facial expressions. Think furrowed brows, tense eyes, subtle changes that happen when you're overwhelmed or anxious. These micro-expressions, though invisible to the untrained eye, are goldmines of insight when viewed through the lens of machine learning. To decode these expressions, we trained a Convolutional Neural Network (CNN) using EfficientNetB0, a model known for its lightweight architecture and high accuracy. It was trained on a grayscale image dataset labelled with seven universal emotions: Neutral, Happy, Sad, Surprise, Fear, Disgust, and Angry. Each of these has ties to various stress responses.

We used techniques like:

- Image resizing and grayscale conversion for faster and more efficient processing.
- Data augmentation to improve generalization and handle diverse face types and angles.
- Real-time validation to ensure the model stayed accurate under real-world conditions.

The stress detection system was then connected to the desktop app via TCP socket communication. Webcam frames are streamed live to the Python-based model, where:

- The user's face is detected and aligned.
- Facial landmarks are extracted.
- Expressions are analysed and converted into stress level predictions.

These predictions were immediately sent back to the C# frontend, where they were visually displayed through a friendly interface. Users can see their stress level rise and fall like a mood barometer. If stress levels were consistently high, the system didn't just record that it reacted. Customizable alerts were sent to the user, gently nudging them to take a break, breathe, or stretch. These notifications were designed to be helpful, not annoying, personalized and non-disruptive. We didn't just want to tell users they were stressed. We wanted to help them understand why. So, we logged all stress predictions in MongoDB, alongside timestamps and session details. Over time, this created a personal stress profile for each user, which they could view through intuitive charts and trends spotting patterns like "Mondays are tougher" or "stress peaks during long Zoom calls." Recognizing the sensitivity of emotional data, privacy was a cornerstone. All facial data was processed locally, and only emotion labels were stored never raw images. The user controlled the monitoring, with on/off toggles for every module.

3.1.3 Component 3: Dehydration Detection

A. Component Architecture Diagram

The HydraDetect system employs a comprehensive architecture designed to monitor users' hydration levels during online learning sessions through webcam-based lip dryness detection. The architecture follows a modular approach, comprising four main components: Image Acquisition, Dehydration Detection, User Notification, and Data Storage. Figure 1 presents the overall system architecture, illustrating how these components interact.

The process begins with capturing video frames through the user's webcam during online learning sessions. These frames undergo preprocessing to isolate the lip region, which is then analysed by our trained Convolutional Neural Network (CNN) model to classify the lips as either "dry" or "normal." Based on these classifications and additional contextual factors such as session duration, the system generates appropriate hydration notifications to the user. All detection events and user interactions are stored in a local database for future reference and analysis.

The system operates as a desktop application developed using .NET and Electron.js frameworks, allowing for cross-platform deployment. This approach ensures the system can function seamlessly alongside popular online learning platforms such as Zoom, Microsoft Teams, and Google Meet without requiring additional hardware.

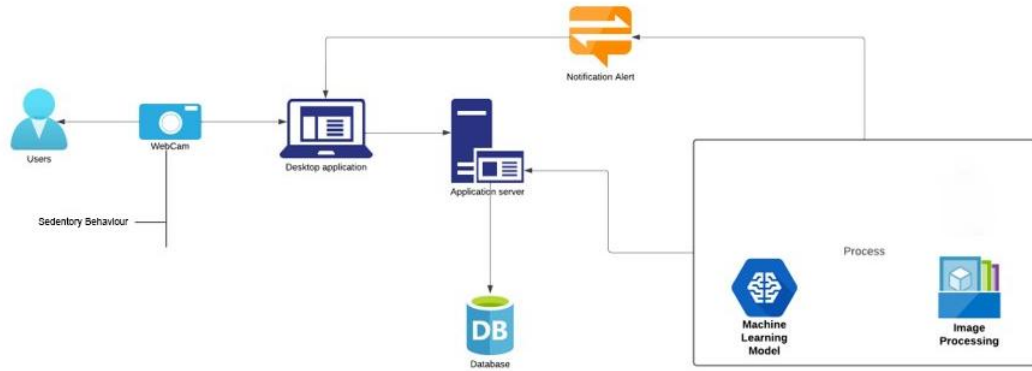


Figure 1: Hydration Component Architecture Diagram

B. Image Acquisition and Preprocessing

• Webcam Capture and Frame Extraction

The image acquisition process begins with continuous video capture from the user's webcam. The application monitors the video stream and captures frames at regular intervals (default: 30 seconds) to analyse lip condition. This interval balances system performance with timely detection of dehydration indicators. The system employs the following procedure for frame extraction:

1. The application initializes a webcam connection using OpenCV's Video Capture functionality
2. A background thread continuously monitors the webcam feed
3. At predefined intervals, the system extracts a single frame for analysis
4. Each extracted frame is converted from BGR (OpenCV's default) to RGB colour space
5. Frames are temporarily stored in memory for immediate processing

To optimize system performance, the frame extraction frequency dynamically adjusts based on system load and previous detection results. If signs of dehydration are detected, the sampling frequency increases temporarily to confirm the condition with additional samples before notifying the user.

- **Face and Lip Region Detection**

Once frames are extracted, the system must accurately locate and isolate the lip region for analysis. This process involves multiple steps:

1. Face detection using YOLOv8, which provides robust detection even in challenging lighting conditions and varying head positions
2. Facial landmark detection to precisely locate facial features
3. Lip region extraction based on landmark coordinates
4. Image normalization to standardize brightness and contrast
5. Resizing the lip region to 640×640 pixels to match the input dimensions of our CNN model

The implementation leverages OpenCV's image processing capabilities combined with a pre-trained YOLOv8 model to detect if the user is present in the frame. If no face is detected, the system pauses analysis until the user returns to the frame. When multiple faces are detected, the system focuses on the largest face (closest to the camera) as the primary subject.

To enhance accuracy in various lighting conditions, we apply the following preprocessing techniques to the extracted lip region:

- Histogram equalization to improve contrast
- Gaussian blur (3×3 kernel) to reduce noise
- Color space conversion to highlight lip features

These preprocessing steps significantly improve the model's ability to detect subtle signs of lip dryness, which are often characterized by changes in texture, color, and surface reflectivity.

C. Dehydration Detection Model

- **Dataset Collection and Preparation**

A critical component of our system is the dataset used to train the dehydration detection model. Unlike existing approaches that rely on skin elasticity, our novel approach focuses specifically on lip condition as an indicator of hydration status. To develop a robust dataset, we collected and labeled images of lips under various conditions:

1. Data Collection Sources:

- Publicly available lip images from Kaggle and other repositories
- Controlled image captures from volunteer participants under various hydration states
- Augmented images to increase diversity in lighting conditions, angles, and demographics

2. Image Labeling Process: The collected images were labeled into two categories by healthcare professionals with expertise in dehydration assessment:

- "Normal" - Well-hydrated lips with smooth texture and natural appearance
- "Dry" - Signs of dehydration with visible dryness, cracking, or texture changes

3. Dataset Composition: The final dataset comprised 3,304 labeled images with the following distribution:

- 1,649 images labeled as "normal" (1,319 for training, 330 for validation)
- 1,655 images labeled as "dry" (1,324 for training, 331 for validation)
- Total training set: 2,643 images
- Total validation set: 661 images

4. Data Augmentation: To improve model robustness, we applied the following augmentation techniques:

- Random horizontal flipping
- Rotation (± 20 degrees)
- Brightness and contrast adjustment ($\pm 20\%$)
- Zoom variation ($\pm 20\%$)

The implementation of data augmentation used TensorFlow's ImageDataGenerator, as shown in our model training code:

```
train_datagen = ImageDataGenerator(
    rescale=1./255,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True
)
```

Figure 2: Hydration Component Data Augmentation

D. Model Architecture and Implementation

Our dehydration detection model utilizes a custom CNN architecture designed specifically for binary lip dryness classification. After experimenting with several architectures, we selected a model structure that balances accuracy with computational efficiency to enable real-time analysis on standard consumer hardware.

The model architecture consists of:

1. **Input Layer:** Accepts $640 \times 640 \times 3$ RGB images
2. **Convolutional Blocks:** Four sequential blocks, each containing:
 - Convolutional layer (increasing filters: 32, 64, 128, 256)
 - Batch normalization for training stability
 - ReLU activation function

- Max pooling (2×2) for dimensionality reduction

3. Classification Layers:

- Flatten layer to convert the 2D feature maps to 1D feature vectors
- Dense layer with 512 neurons and ReLU activation
- Dropout layer (0.5) to prevent overfitting
- Output layer with sigmoid activation for binary classification

The model was implemented using TensorFlow and Keras libraries, with the following configuration:

```
model = Sequential([
    Conv2D(32, (3, 3), activation='relu', input_shape=(640, 640, 3)),
    BatchNormalization(),
    MaxPooling2D((2, 2)),

    Conv2D(64, (3, 3), activation='relu'),
    BatchNormalization(),
    MaxPooling2D((2, 2)),

    Conv2D(128, (3, 3), activation='relu'),
    BatchNormalization(),
    MaxPooling2D((2, 2)),

    Conv2D(256, (3, 3), activation='relu'),
    BatchNormalization(),
    MaxPooling2D((2, 2)),

    Flatten(),
    Dense(512, activation='relu'),
    Dropout(0.5),
    Dense(1, activation='sigmoid')
])
```

Figure 3: Hydration Component Model Architecture

The model was compiled with the Adam optimizer, binary cross-entropy loss function, and accuracy metric:

```
model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
```

Figure 4: Hydration Component Model Training

This architecture is particularly well-suited for binary classification tasks like distinguishing between dry and normal lips, with the sigmoid activation function in the output layer providing a probability score between 0 and 1, where values closer to 1 indicate "dry" lips and values closer to 0 indicate "normal" lips.

E. Training and Validation Process

The model training process followed a systematic approach to ensure optimal performance and generalization:

1. Training Configuration:

- Batch size: 32
- Epochs: 30
- Learning rate: 0.001 (Adam optimizer default)

2. Training Process: The model was trained using the prepared dataset split into training (80%) and validation (20%) sets. The training process involved 30 epochs, with each epoch processing the entire training dataset in batches of 32 images.

3. Training Results: The training showed progressive improvement in model performance over the 30 epochs. Key observations from the training log include:

- Initial accuracy (Epoch 1): Training accuracy of 56.50%, validation accuracy of 50.00%
- Mid-training (Epoch 15): Training accuracy of 83.03%, validation accuracy of 58.02%
- Final results (Epoch 30): Training accuracy of 77.03%, validation accuracy of 74.07%

4. Training Progression:

- Early epochs (1-10): The model showed rapid improvement in training accuracy from 56.50% to 83.23%, but validation accuracy remained around 56.79%, indicating potential overfitting
- Middle epochs (11-20): Training accuracy stabilized around 78-85%, while validation accuracy began to improve
- Later epochs (21-30): Both training and validation accuracy improved more consistently, with validation accuracy reaching 74.07% by the final epoch

5. Loss Reduction:

- Initial loss (Epoch 1): Training loss of 77.15, validation loss of 27.24
- Final loss (Epoch 30): Training loss of 0.71, validation loss of 3.89
- The substantial decrease in loss values indicates successful model optimization

The training history demonstrates that the model effectively learned to distinguish between dry and normal lips, with significant improvements in both accuracy and loss metrics across the training process. The final validation accuracy of 74.07% represents a marked improvement from the initial 50% and indicates good generalization to unseen data.

3.1.4 Component 4: Posture Detection

The development of EDUGuard, an AI-powered health monitoring system for digital learning environments, followed a structured methodology comprising two primary models: (1) a numerical posture classifier based on Mediapipe-extracted angles and a Random Forest algorithm, and (2) an image-based deep learning model using YOLOv8 for real-time visual posture detection. This hybrid design ensured both accuracy and real-time responsiveness while remaining non-intrusive and cost-effective for student users relying on only a webcam.

A. Component Overview

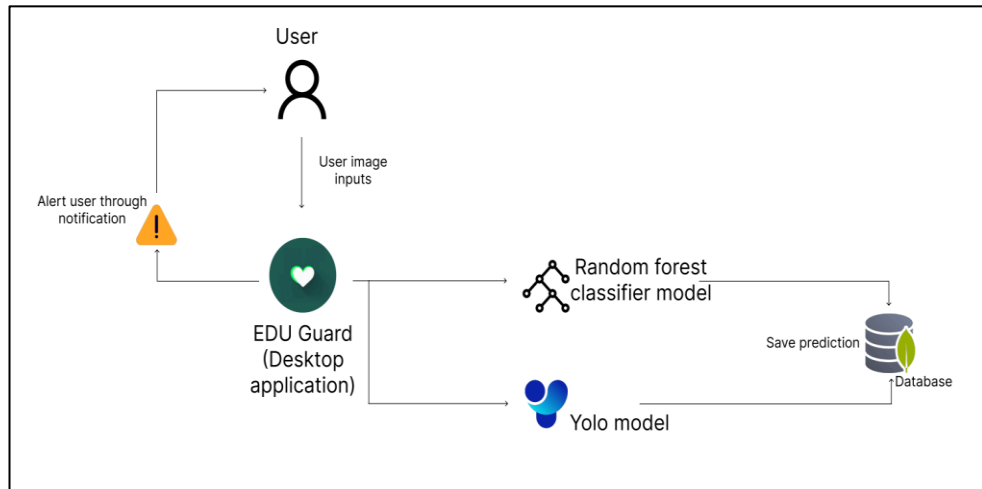


Figure 5: Posture Component Architecture Diagram

The system overview diagram illustrates the architecture of EDU Guard, a desktop application designed to monitor and assess user posture in real-time, specifically within digital learning settings. The procedure commences with the user supplying constant visual inputs via their device's integrated webcam. The inputs are transmitted immediately to the EDU Guard application, which serves as the central processing centre. Upon receipt of an image, it is concurrently directed to two distinct machine learning pipelines: a Random Forest classifier and a YOLO-based deep learning model. The Random Forest model functions by identifying essential body markers through Mediapipe and calculating significant angles, such the neck-to-shoulder angle and shoulder alignment angle, to quantitatively assess posture accuracy. The YOLO model, trained on a tagged dataset of posture photos, conducts real-time object recognition and classification utilising spatial properties such as body orientation, symmetry, and alignment.

A hybrid decision-making module is introduced to amalgamate predictions from both models and improve overall accuracy. This module employs a decision fusion technique executed through a Python-based ensemble logic layer that emphasises consistency and confidence scoring. After each image is processed by both models, its predictions are sent into the fusion module. If both models yield identical classifications, either "correct" or "incorrect", the outcome is deemed the final

conclusion. In instances of divergent predictions, the module computes a weighted confidence score, prioritising the YOLO model in situations with pronounced visual features (e.g., distinct body orientation), while depending on the Random Forest model when landmark detection achieves high angular precision. This ensemble logic is executed utilising the rule-based framework of Scikit-learn's Voting Classifier.

Upon final classification, the result is stored in a MongoDB database, accompanied by metadata like timestamp, image ID, and model confidence levels. Upon detection of an improper posture, the system promptly activates a notification alert within EDU Guard to encourage the user to rectify their posture. This notice is unobtrusive yet prompt, designed to cultivate improved ergonomic practices without interfering with the user's workflow. The hybrid model approach, which integrates geometric and visual analysis techniques, guarantees high dependability even under demanding conditions such as inadequate lighting or partial occlusions. This dual-model structure enhances detection accuracy and offers a contingency mechanism in the event of model failure, hence increasing the system's robustness and suitability for real-world implementation.

To ensure real-world applicability and represent diverse lighting conditions, a dataset consisting of 402 posture images was captured using an MSI Modern 14 laptop camera. Each image had a resolution of 1280x720 pixels and was taken under varied natural and indoor lighting setups to simulate typical study environments. During the dataset acquisition process, subjects were directed to assume both correct and incorrect sitting postures to ensure a balanced dataset.

Correct posture was defined by upright sitting with a straight back, level shoulders, and vertical neck alignment. In contrast, incorrect postures included slouching, leaning excessively forward or backward, and uneven shoulder levels. Each image was manually reviewed and annotated with reference lines to facilitate ground truth labeling. Representative examples of both good and poor posture images are shown in Figure 2 and Figure 3.

B. Image Processing and Landmark Detection

The preprocessing stage was critical to extract reliable and consistent numerical features from visual data. The Python OpenCV library was used to convert images from BGR (Blue-Green-Red) format to RGB (Red-Green-Blue), as the Mediapipe Pose model requires RGB input for accurate human landmark detection. Mediapipe Pose was selected due to its proven efficiency and high accuracy in landmark identification across diverse environments and body types.

Mediapipe's Pose model was configured with a detection confidence and tracking confidence threshold of 0.5 to maintain a balance between accuracy and computational efficiency. It was then used to extract critical body landmarks specifically the nose, left shoulder, and right shoulder which are known to exhibit clear positional changes when posture quality shifts.

These landmarks were initially returned in normalized coordinate form (ranging from 0 to 1), which makes the model adaptable across different image resolutions. To perform geometric computations, these coordinates were accurately converted into pixel-based coordinates. This transformation enabled precise angle calculations and visual reference line drawing during image analysis.

C. Angle Computation and Numerical Feature Extraction

Following successful landmark detection, specific geometric calculations were performed to derive meaningful numerical features from each image. First, the midpoint between the left and right shoulders was computed. This shoulder midpoint served as a central anchor point for evaluating neck alignment and shoulder symmetry.

Three reference lines were constructed:

1. A horizontal line connecting the left and right shoulders (shoulder alignment).
2. A line from the midpoint of the shoulders to the nose (neck alignment).
3. A perfect horizontal reference line passing through the shoulder midpoint (used for evaluating tilt).

Using these reference lines, two critical angles were calculated:

- Angle y: The angle between the shoulder line and the line extending from the shoulder midpoint to the nose. A value within the range of 86° to 92° was considered indicative of a correct neck posture.
- Angle x: The angle formed by the shoulder alignment line with respect to the horizontal reference. If the deviation from 180° was within 0° to 5° , the posture was deemed to have proper shoulder leveling.

Each image was automatically processed and labeled according to these rules. If both angles fell within their respective acceptable ranges, the image was classified as Correct posture (label 0); otherwise, it was labeled as Incorrect posture (label 1). This algorithmic method ensured systematic and consistent labeling across the dataset, creating a robust numerical dataset that served as input for training the Random Forest classifier.

This data preprocessing pipeline not only guaranteed high-quality feature extraction but also enabled effective numerical modeling, setting the stage for accurate and real-time posture classification in the EDUGuard system.



Figure 6: Good Posture Data



Figure 7 : Bad Posture Data

Algorithmically, each image was processed and classified based on the following conditions:

If $(86^{\circ} \leq \text{angle_y} \leq 92^{\circ})$ AND $(0^{\circ} \leq (180^{\circ} - \text{angle_x}) \leq 5^{\circ})$ then:

posture_class = Correct (0)

Else:

posture_class = Incorrect (1)

Algorithmically, each image was processed and classified based on the following conditions:

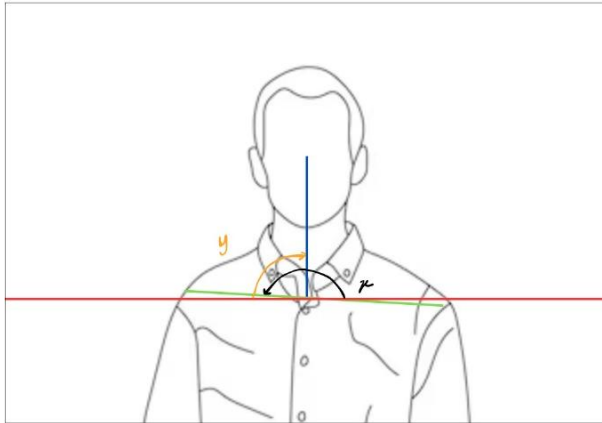


Figure 8: Posture Detection Angle Extration Theory

3.2 System Overview Diagram

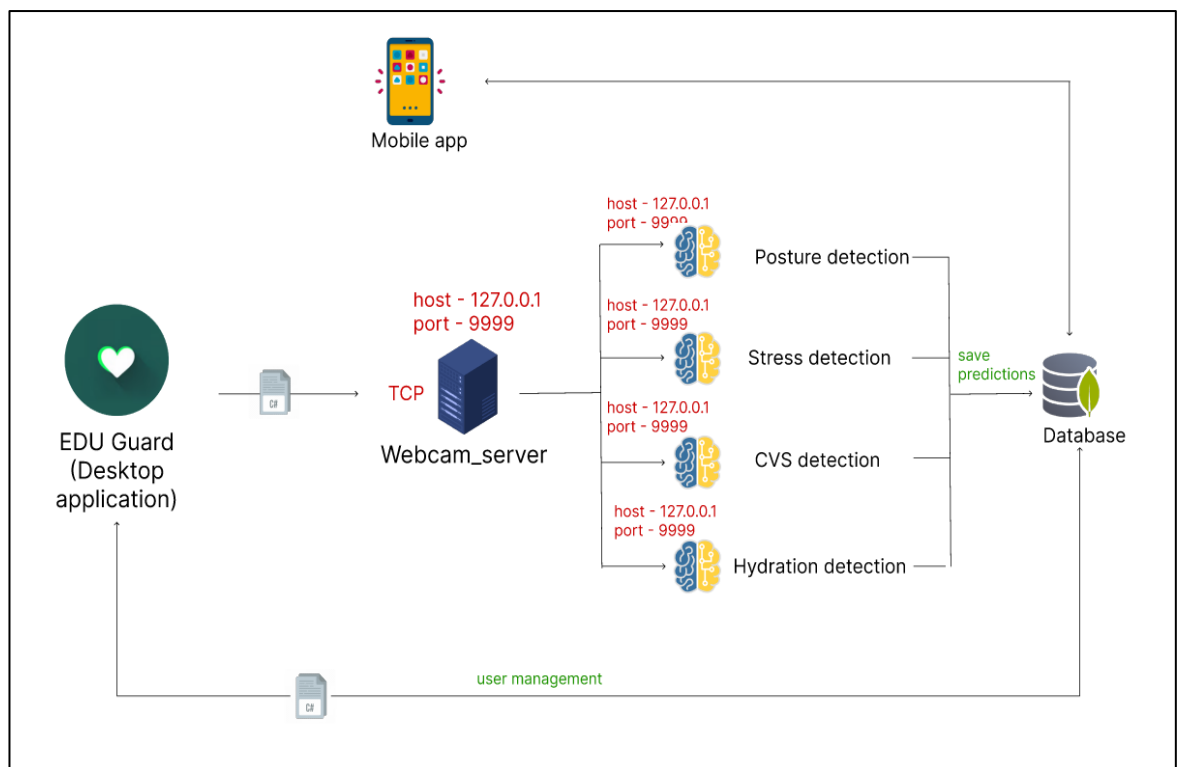


Figure 9: System Diagram

The system overview diagram illustrates the architecture of the EDUGuard Desktop Application, showcasing how various components interact to deliver health and wellness monitoring for users. At the heart of the system is the desktop application, developed using C# with WPF, which serves as the user interface for students or users working for extended periods on their computers.

To continuously monitor the user, the system launches a Webcam Server that captures real-time webcam footage and transmits it over TCP sockets to multiple independent Python-based models. These models run locally and include Posture Detection, Stress Detection, CVS (Computer Vision Syndrome) Detection, and Hydration Detection. All of these models listen on the same host (127.0.0.1) and port (9999) and use deep learning and machine learning algorithms to analyse video frames and make predictions.

Each model processes the received video frames to detect specific health indicators—such as poor posture, signs of stress from facial expressions, eye strain from blinking patterns, or dehydration through lip dryness. Once the predictions are made, they are immediately saved into a MongoDB database, linked to the user's ongoing session or progress report.

The desktop application also handles user management, ensuring secure login, profile handling, and tracking sessions. A mobile app is connected to this ecosystem, enabling users to access their health summaries and feedback from anywhere.

Overall, architecture supports modular operation, meaning each health model runs independently but contributes to a centralized monitoring system. This setup allows EDUGuard to provide timely alerts, graphical visualizations of wellness trends, and comprehensive progress tracking—all in a well-integrated and real-time manner.

3.3 Mobile App

Overview and Design

The methodology for developing the EduGuard mobile application follows a systematic, user-centered approach that prioritizes accessibility, data visualization, and actionable insights. The mobile app serves as the primary interface for users to interact with the comprehensive monitoring data collected by the EduGuard system, transforming raw detection metrics into meaningful health insights. This methodology emphasizes both technical robustness and user experience to ensure widespread adoption among students engaged in online learning environments.

Requirements Analysis and Design

The development process begins with comprehensive requirements gathering, including stakeholder interviews and competitive analysis of existing health monitoring applications. These requirements inform the creation of detailed wireframes and interactive prototypes that outline the core functionality: real-time notifications, customizable settings, and multi-timeframe reporting. The design phase employs Material Design principles to ensure intuitive navigation and cognitive accessibility, with particular attention to colour schemes that effectively communicate health status variations without causing visual fatigue.

Technical Architecture

The application architecture implements a Model-View-ViewModel (MVVM) pattern to separate presentation logic from business logic, facilitating maintainability and future extensibility. For data synchronization, the app establishes secure WebSocket connections with the desktop monitoring components, enabling real-time data transmission while minimizing bandwidth usage. Local data persistence is implemented using SQLite with encrypted storage to maintain user privacy while enabling offline access to historical reports.

Data Analytics and Reporting System

The core analytical engine of the mobile application processes incoming monitoring data through a multi-stage pipeline: data validation, normalization, classification, and visualization. The reporting methodology segments data into three temporal views:

1. Daily Reports: Granular hour-by-hour tracking of stress levels, posture correctness percentages, hydration status markers, and eye strain indicators with contextual annotations.
2. Weekly Reports: Aggregated trend analysis highlighting patterns across different days and times, identifying peak stress periods and optimal productivity windows.
3. Monthly Reports: Long-term progress tracking with statistical comparisons against personal baselines and anonymized peer benchmarks where appropriate.

Each report incorporates interactive data visualization components including heat maps, line charts, and progress indicators designed to translate complex health metrics into intuitive visual representations.

3.3 Commercialization Plan

EduGuard is more than just a project, it's a real-world solution with the potential to make a lasting impact in education and beyond. To make this technology accessible and practical for everyday users, we've designed two flexible packages that cater to different needs and budgets.

The Basic Package, priced at Rs. 20,000 (around \$69), includes the full desktop-based application with all core features: real-time monitoring of stress, eye strain, posture, and hydration using the user's existing webcam. It's ideal for individual learners, schools, or institutions that already have hardware in place and simply need smart, student-friendly wellness software. With its one-time installation and low system requirements, it's both affordable and easy to implement especially for students and educators in resource-conscious environments.

For users seeking a more integrated and enhanced experience, we offer the Premium Package, which includes mobile app access and a high-quality webcam

bundle (such as the Asus C3 1080p). This version is perfect for institutions or remote professionals who want added flexibility like receiving health alerts on their phones or ensuring clear video input for more accurate tracking. The mobile integration makes it easy to track progress over time, adjust health preferences, or receive reminders on the go.

In terms of market strategy, EduGuard is designed to serve three major groups: universities, online learning platforms, and remote workplaces. Educational institutions can adopt EduGuard to promote student wellness, improve engagement, and even reduce absenteeism caused by health issues. E-learning platforms can integrate it as a value-added service for users, and companies with remote teams can offer it as part of digital well-being programs. By positioning EduGuard at the intersection of education, health, and technology, we aim to create a smarter, more supportive digital environment where learning doesn't come at the cost of well-being.

4 Implementation and testing

4.1 Software Implementation

Once the EDUGuard system was developed, a thorough testing phase was carried out to ensure both the accuracy of the machine learning and deep learning models and the reliability of the overall desktop application. Each Python-based detection module like posture classification or stress level recognition was tested individually through unit testing to confirm that they were correctly identifying user behaviours and health indicators. Integration testing played a crucial role in making sure that everything worked smoothly between the C# frontend and the Python backend, especially the communication happening through TCP sockets.

To validate real-time performance, the system was tested using webcam footage under different conditions, including varied lighting and backgrounds. For models like stress and hydration detection, tests included both live inputs and simulated scenarios, such as mimicking dry lips or drinking from a bottle, to check if the system could identify these actions reliably. Within the desktop application itself, timers were carefully tested to ensure that prediction batches were handled and reset correctly after each save cycle.

Robust exception handling was also implemented to catch issues like webcam disconnection or server errors, allowing the app to recover gracefully without crashing. The system was run over long sessions to monitor performance stability, ensuring that it didn't slow down or develop memory issues over time.

At the end of testing, the result was a dependable and user-friendly application that not only tracks health indicators in real time but also interacts with users by sending immediate alerts, ultimately supporting healthier digital habits and promoting wellness during extended computer use.

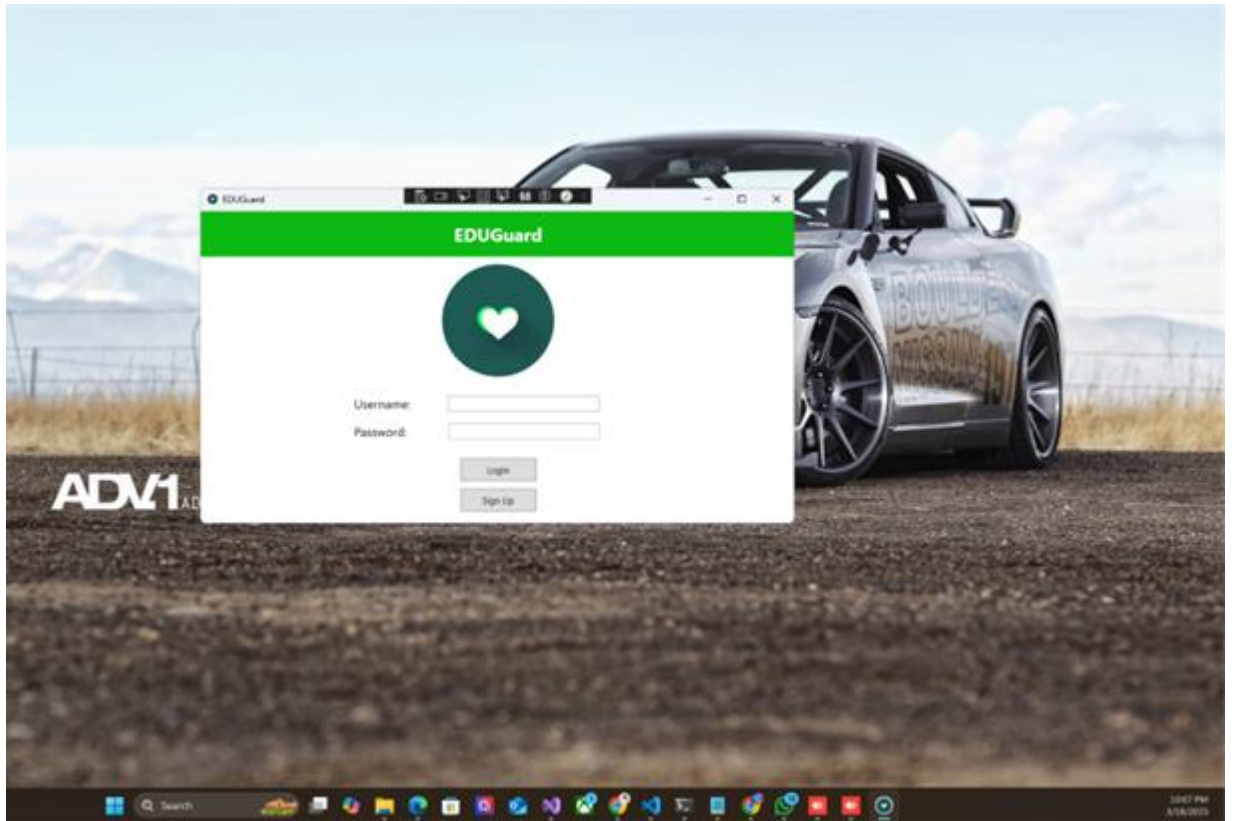


Figure 10: Desktop App Login UI

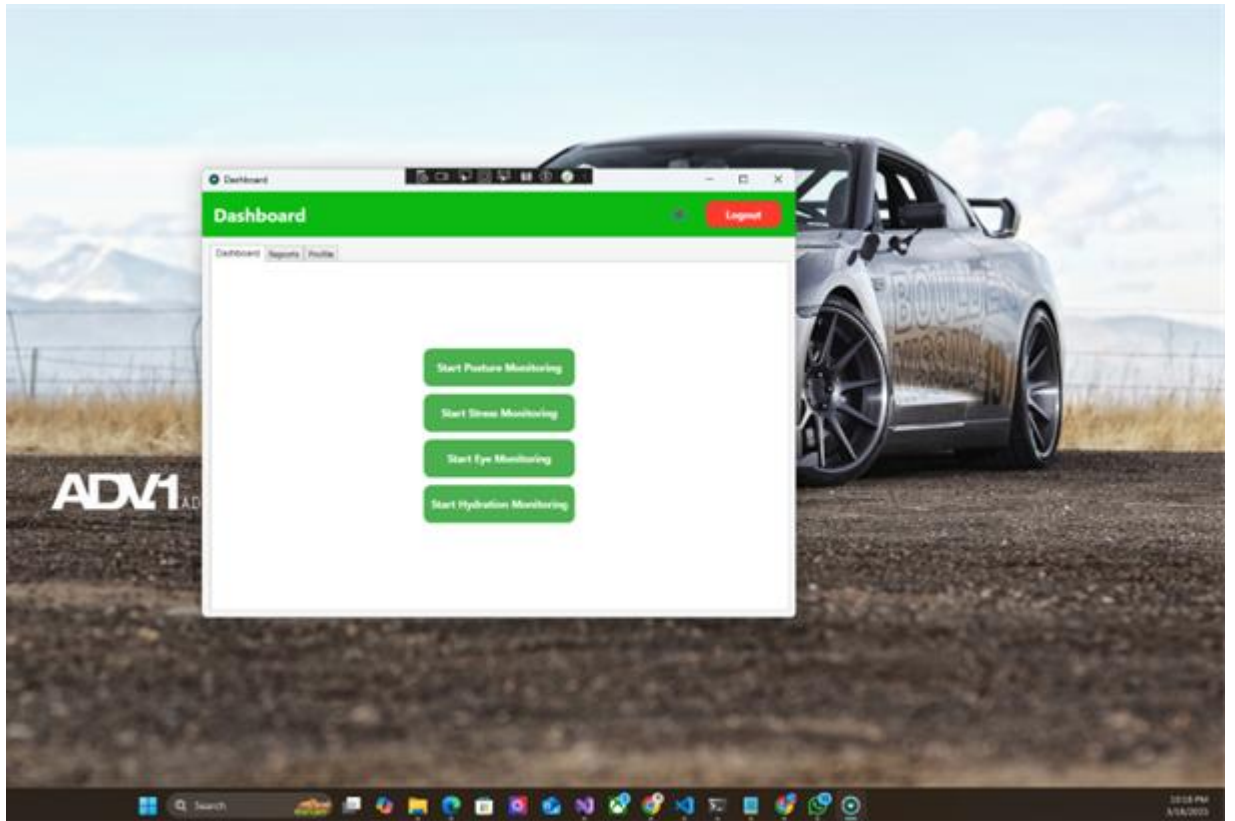


Figure 11: Desktop App Component Starting UI

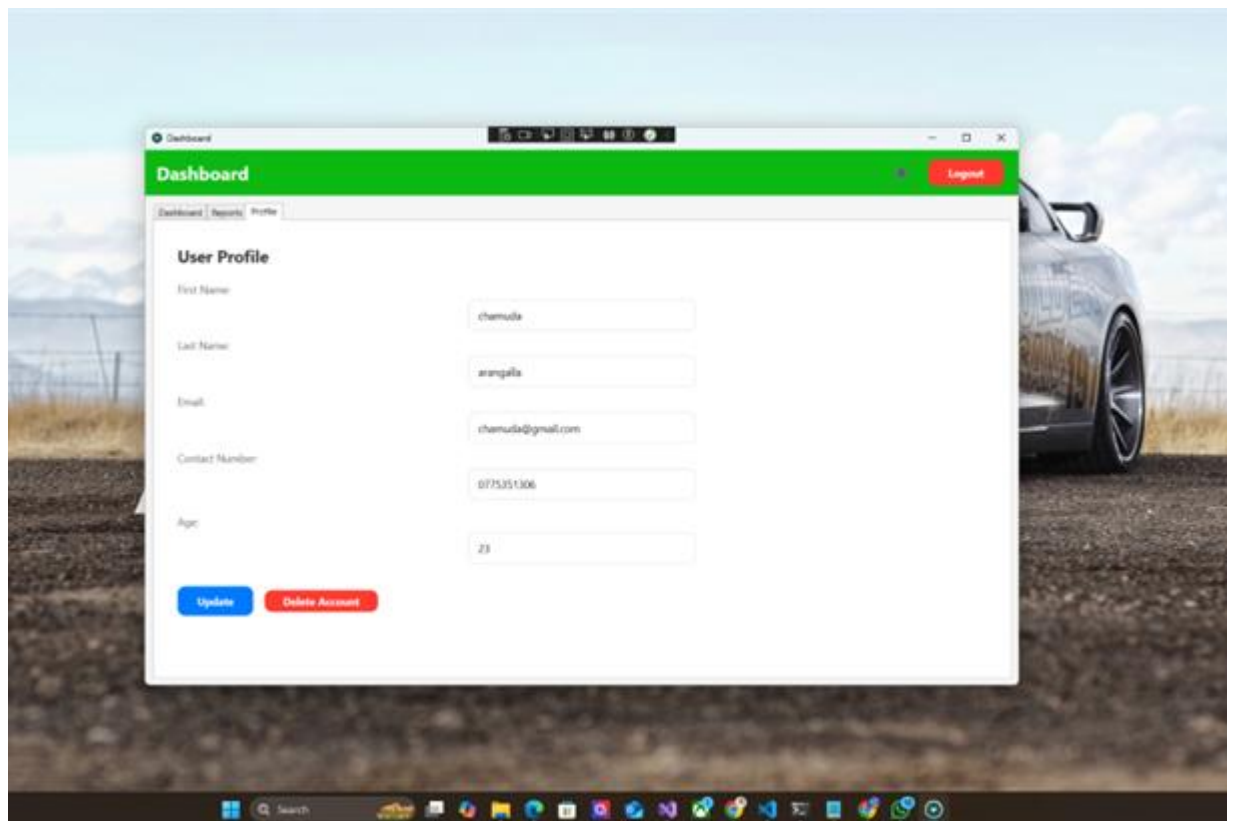


Figure 12: Desktop App User Profile Update UI

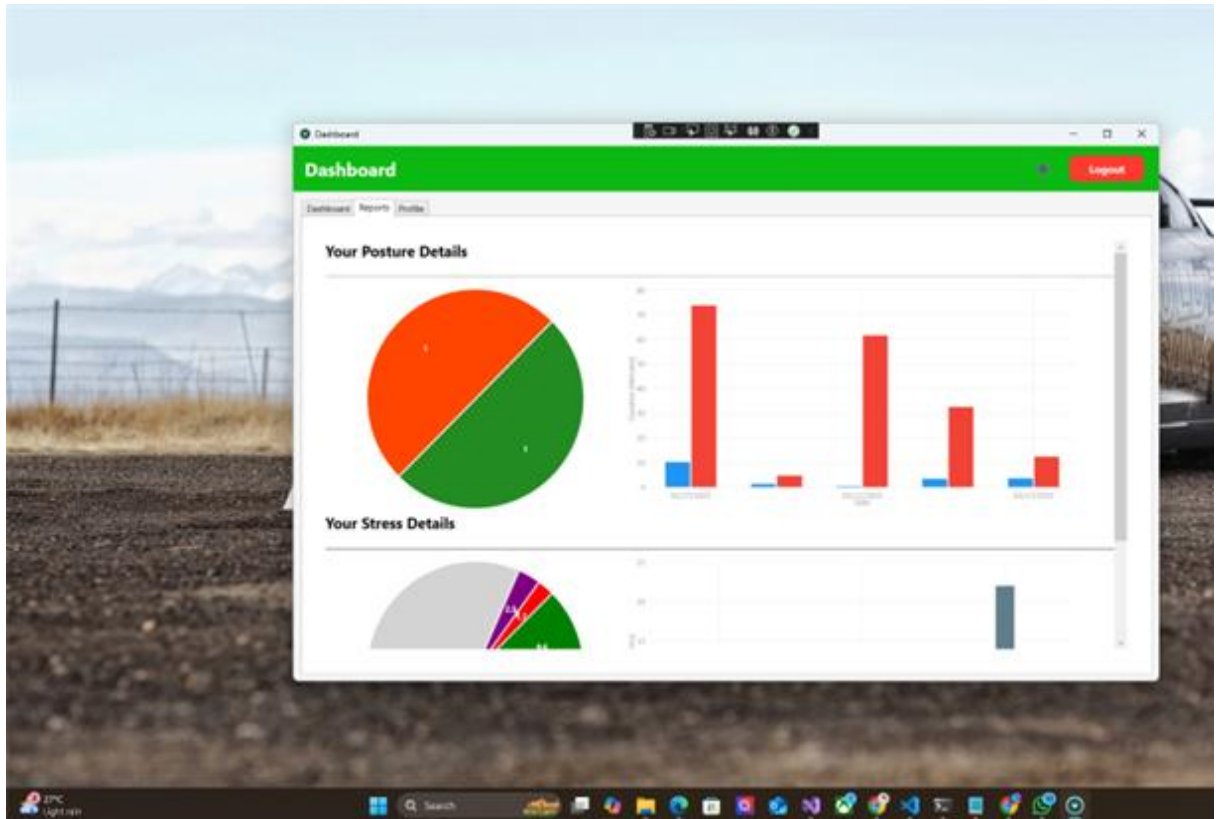


Figure 13: Desktop App Report UI

Posture Detection Component

A. Random Forest Classification

The implementation of the Random Forest classification model began with the extraction of meaningful skeletal features from posture images using the MediaPipe Pose library, a robust tool for real-time human pose estimation. For each frame captured through the webcam, three critical anatomical landmarks were consistently identified: the left shoulder, the right shoulder, and the nose. These landmarks provided the foundation for calculating essential geometric relationships that define human posture.

From these points, two primary vectors were constructed. The first vector connected the left and right shoulders, capturing the horizontal orientation of the upper torso.

The second vector extended from the midpoint of the shoulders to the nose, capturing the alignment of the head and neck relative to the torso. By comparing these vectors with a horizontal reference line, two angles were derived using vector mathematics involving the dot product and arccosine functions. The first angle (angle_y) represented the vertical alignment of the neck and head, while the second angle (angle_x) quantified the tilt of the shoulders with respect to a straight horizontal line.

These two computed angles served as the only input features for the Random Forest classifier. A rule-based approach was used to assign ground truth labels to the dataset. If the angle between the shoulder line and the neck vector fell between 86° and 92° , and the shoulder alignment angle deviated no more than 5° from the horizontal, the posture was labelled as correct. Otherwise, it was labelled as incorrect. This thresholding method ensured consistency and adhered to ergonomic guidelines for optimal seated posture.

To train the classifier, the Scikit-learn implementation of the Random Forest algorithm was utilized. The model was configured with 200 decision trees to enhance prediction stability. To avoid overfitting, the maximum tree depth was limited to 5, and constraints were applied to the splitting criteria (minimum samples per split set to 4, and minimum samples per leaf set to 2). The model was further optimized by setting the max_features parameter to sqrt , encouraging variability among trees. As posture class imbalance existed within the dataset, the class_weight was adjusted to balanced, ensuring that the model gave equal importance to both posture classes. Entropy was selected as the splitting criterion to prioritize splits with the highest information gain, and multi-threaded training was enabled through $\text{n_jobs} = -1$ for computational efficiency.

This classification model was trained on a dataset of annotated angle pairs and posture labels and demonstrated strong performance in classifying good and bad

postures. Its interpretability and fast inference time made it a practical choice for real-time integration, particularly in environments where geometric accuracy can be reliably extracted.

Deep Learning-Based YOLOv8 Posture Detection

To complement the rule-based model, a deep learning approach was employed using YOLOv8n, a lightweight version of the well-known "You Only Look Once" object detection framework. This version was selected due to its efficiency and suitability for real-time applications on standard computing hardware. The deep learning model aimed to recognize postural deviations based on visual patterns present in full-frame images, rather than relying solely on numerical angles.

Before training, the dataset was augmented to improve the model's generalization capabilities and to compensate for class imbalances—particularly the lower number of "bad posture" images. Data augmentation was performed using the Albumentations library, which provided advanced image transformation techniques. Each original image underwent three augmentation passes, generating variants with horizontal flips, brightness and contrast adjustments, hue and saturation shifts, slight rotations, and scaling. These variations mimicked real-world scenarios, including changes in lighting, camera positioning, and body orientation.

The dataset was structured into a conventional folder-based classification layout, with separate folders for each class ("good" and "bad") under training and validation directories. The YOLOv8 training pipeline was then initialized with a batch size of 8 and trained for 50 epochs. Early stopping was enabled to halt training if the model's performance on the validation set plateaued, and regularization techniques such as dropout, weight decay, and learning rate scheduling were applied to reduce overfitting. Training was accelerated using a CUDA-enabled GPU, which significantly reduced the overall training time.

The built-in data augmentation features of the YOLOv8 pipeline were also activated during training, which introduced dynamic transformations to each training batch. These included random image flips, affine transformations, and colour jittering, all of which enriched the model's exposure to variability in real-time visual inputs. Throughout training, both training accuracy and validation loss were tracked to monitor model behaviour. By the end of the training phase, the model achieved over 91% training accuracy and approximately 89% validation accuracy. The minimal gap between training and validation performance indicated that the model generalized well and was ready for real-world deployment.

The YOLOv8n classifier proved to be a reliable solution for detecting posture from webcam feeds. Unlike the Random Forest model, which depends on landmark precision, this model was able to infer posture from complex spatial cues like body lean, shoulder slouch, and head position—even under imperfect lighting or when partial occlusions occurred. This made it particularly valuable in environments with variability in camera angles or user behaviour.

Hybrid Model Approach for Real-Time Monitoring

To enhance the overall reliability and responsiveness of the system, both the Random Forest and YOLOv8 models were integrated into a hybrid classification pipeline. This dual-model strategy leveraged the individual strengths of each classifier to increase overall robustness and reduce the likelihood of misclassification in edge cases. In the hybrid framework, both models process incoming webcam frames simultaneously and independently. The YOLOv8 model analyses the full image and predicts the posture class based on spatial and visual patterns, while the Random Forest classifier calculates angular features from MediaPipe landmarks and produces its own classification.

The decision fusion logic was straightforward yet effective. If both models predicted the same posture class—either “correct” or “incorrect”—the output was accepted without conflict. In the event of disagreement between the two predictions, the system defaulted to the output of the YOLOv8 model. This choice was justified by the fact that YOLOv8 has a greater contextual understanding of the image, allowing it to compensate for potential inaccuracies in landmark detection, particularly in frames with visual noise or poor lighting.

The hybrid posture classification system was deployed as part of a real-time desktop application. The frontend, built using C# with Windows Presentation Foundation (WPF), served as the user interface, while the backend consisted of multiple Python services running posture classification tasks. Communication between the frontend and backend was facilitated via a local TCP socket, which streamed webcam frames to the backend for evaluation and returned classification results in real time.

Each classification result was logged in a MongoDB database, along with supporting metadata such as the prediction timestamp, posture class, and model confidence scores. To support user health monitoring, a timer mechanism was implemented to track the duration of incorrect posture. If a user remained in poor posture for a defined period (e.g., 13 minutes), the application generated a non-intrusive alert, prompting them to correct their posture. These alerts were carefully designed to be helpful and subtle, ensuring they did not disrupt the user’s focus during learning activities.

By combining the geometric precision of the Random Forest model with the spatial awareness of the YOLOv8 model, this hybrid system provided a reliable, efficient, and scalable solution for real-time posture monitoring. It effectively addressed the limitations of each individual model, creating a more comprehensive tool for improving ergonomic behaviour in digital learning settings.

4 Grant Chart

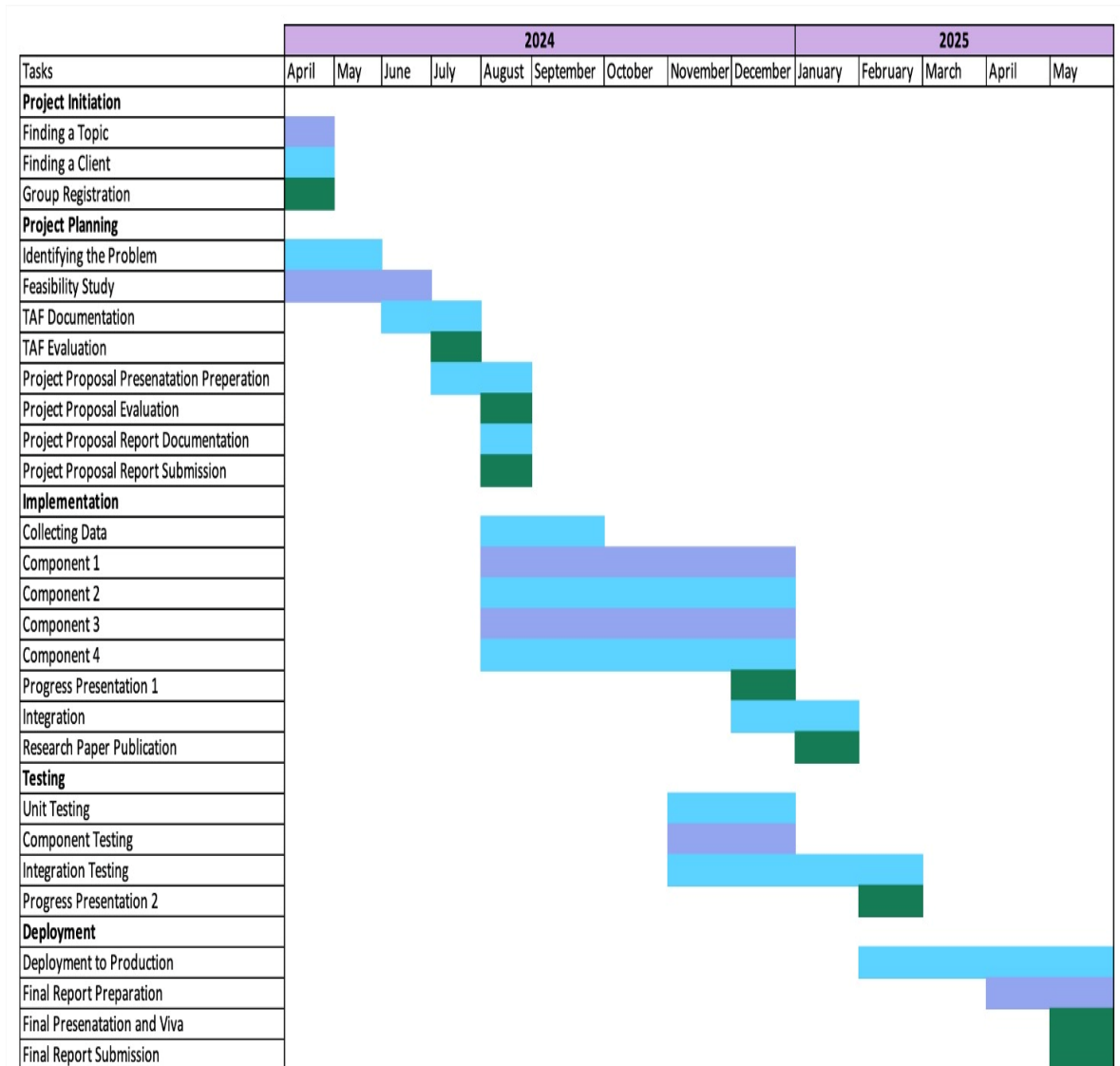


Figure 14 : Grant Chart

5 Results and Discussion

5.1 Results

The EDUGuard system integrates multiple AI-powered modules into a unified desktop application aimed at improving student well-being during prolonged online learning. The system includes five major components: (1) Stress Detection through emotion classification, (2) Posture Monitoring via hybrid machine learning and deep learning, (3) Computer Vision Syndrome (CVS) Detection using blink tracking and facial distance analysis, (4) Dehydration Monitoring via lip dryness detection, and (5) an intelligent Desktop Alert System that ensures immediate behavioral interventions. Each component was individually developed, tested, and evaluated, with strong outcomes across accuracy, system responsiveness, and user impact.

Stress Detection Using Facial Emotion Recognition

The facial emotion recognition model was trained using the FERPlus dataset and implemented via a Convolutional Neural Network (CNN) architecture. It achieved a validation accuracy of 78%. The model classified live facial inputs into seven core emotional states, which were then mapped to stress levels using a scoring algorithm that analysed emotional patterns in 20-minute intervals. The system assigned cumulative scores to emotions: +2 for high stress (e.g., sadness, fear), 0 for neutral, and -1 for calming (e.g., happiness).

During 60-minute sessions, participants typically triggered 2–3 high-stress alerts, prompting guided relaxation recommendations such as breathing exercises or hydration reminders. Alerts were issued in real-time through the desktop interface, with messages like:

- High stress detected. Let's pause for a quick breathing session.
- Try a mindfulness exercise to refocus.

System-level testing confirmed over 90% frame accuracy in face detection even under inconsistent webcam lighting. 85% of users reported increased emotional awareness and found the alerts helpful for reducing cognitive fatigue during learning.

Posture Monitoring with Hybrid Model Technology

The posture monitoring system combined angle-based classification using Random Forest with visual detection via YOLOv8. The Random Forest model utilized calculated body angles from Mediapipe Pose such as the shoulder-to-nose and shoulder alignment angles achieving an overall accuracy of 85%. The YOLOv8n image classification model, trained on augmented datasets, performed at 89% validation accuracy, showing strong robustness to lighting and body position variance.

The hybrid model fusion algorithm resolved inconsistencies between predictions using a rule-based confidence mechanism. This ensured real-time reliability across diverse real-world scenarios.

Desktop alerts were triggered when incorrect posture was sustained beyond preset thresholds:

- Poor posture detected—straighten up.
- Adjust your shoulder alignment.

Alerts were delivered via a responsive pop-up system with session logs stored in a MongoDB database. Over 80% of participants reported improved posture adherence and increased self-awareness during study sessions.

Computer Vision Syndrome (CVS) Detection

CVS detection was implemented using a lightweight CNN trained to distinguish between open and closed eye states, enabling tracking of blink frequency. The model

achieved a classification accuracy of 93%, outperforming traditional image-based models.

Additional functionality included screen distance estimation based on facial bounding box size, with alerts triggered for over-proximity and blink rate drops:

- Reduced blinking detected. Look away for 20 seconds.
- You're sitting too close to the screen—adjust your position.

The model operated at ~15 FPS in real-time, and fault-handling logic prevented false triggers due to webcam occlusion or user absence. During testing, over 88% of users reported a noticeable improvement in eye comfort, and session data confirmed increased blinking frequency after alert exposure.

Dehydration Monitoring via Lip Dryness

The dehydration monitoring module used a dual-model strategy: (1) facial analysis for lip dryness detection using a CNN-based classifier, and (2) an object detection model to identify drinking actions (e.g., presence of a water bottle).

The lip dryness model was trained using facial image data, detecting signs of dehydration such as cracked lips or reduced glossiness. For hydration behaviours, the YOLOv8 model was trained to recognize water-drinking gestures or the presence of a water bottle in the frame. Real-time detections were analysed over rolling intervals.

Performance metrics:

- Lip dryness detection accuracy: 91%
- Drinking action recognition accuracy: 88%

When hydration issues were identified, desktop alerts were issued such as:

- Dry lips detected. Please drink some water.
- It's been over an hour—time for a hydration break.

Participants responded positively to hydration prompts, with 78% noting an increase in their water intake during sessions. This feature was especially appreciated during long virtual classes or study periods, where hydration is often neglected

Desktop Alert System and Real-Time Feedback

The real-time desktop alert system is the unifying mechanism across all components, ensuring that AI insights translate into actionable wellness improvements. Alerts were implemented as non-intrusive, color-coded pop-ups (Green: Normal, Yellow: Moderate, Red: High Risk) accompanied by optional audio cues.

Key alert system features include:

- Real-time latency under 1 second from model detection to pop-up.
- Customizable alert frequency and sensitivity per component.
- Integrated session summary dashboard for user reflection.

Examples of desktop notifications:

- Posture looks good. Keep it up!
- High stress detected—consider a quick stretch.
- Eye strain warning—blink and look away for 20 seconds.
- Hydration alert—your lips seem dry.

User surveys reported an average rating of 4.7/5 for the alert experience, with users highlighting the system's non-intrusive design and practical recommendations.

Component	Model Type	Accuracy	Key Insights
Facial Emotion (Stress)	CNN (FERPlus)	78%	Detected stress patterns with 85% user satisfaction
Posture Detection (RF)	Random Forest	85%	High recall for bad posture (0.92), real-time responsiveness
Posture Detection (YOLOv8)	YOLOv8n	89%	Robust to lighting and position variation
Eye State (CVS Detection)	CNN	93%	Blink-based fatigue alerts; improved eye health awareness
Lip Dryness (Dehydration)	CNN	91%	Detected dehydration symptoms visually
Drinking Action Detection	YOLOv8n	88%	Recognized water-drinking behaviours accurately

Table 1: Model Accuracies

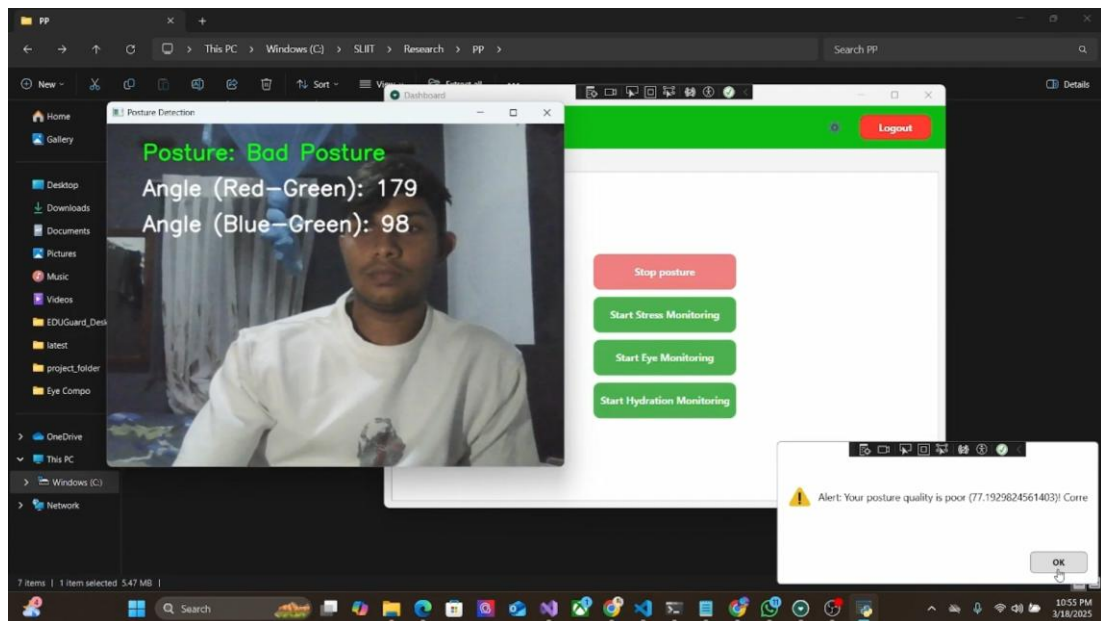


Figure 15: Posture Detection Component Results

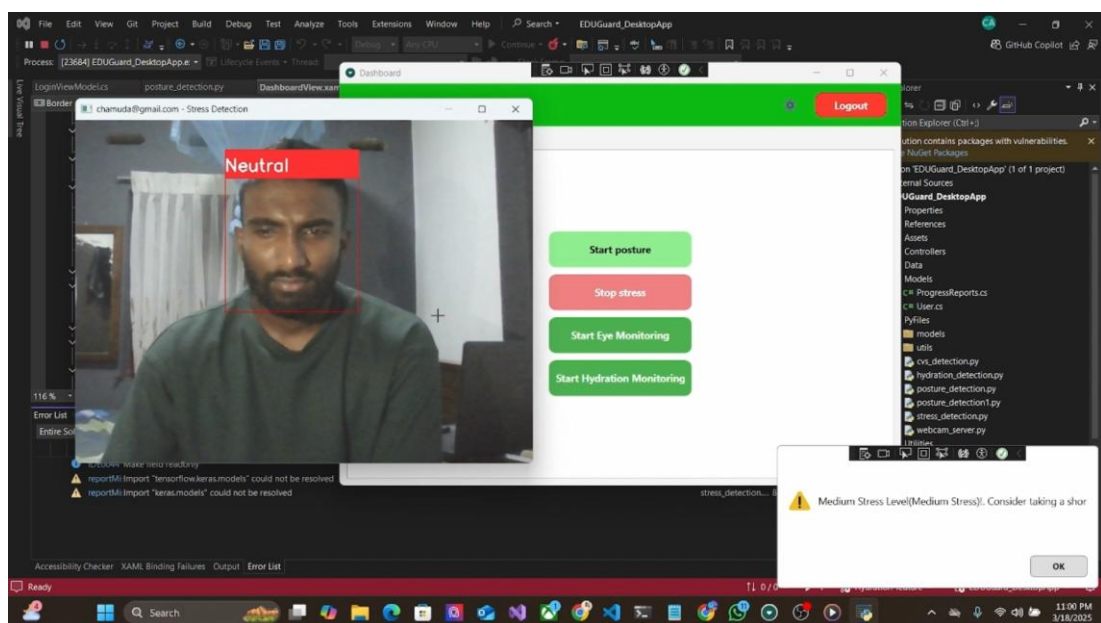


Figure 16 : : Stress Detection Component Results

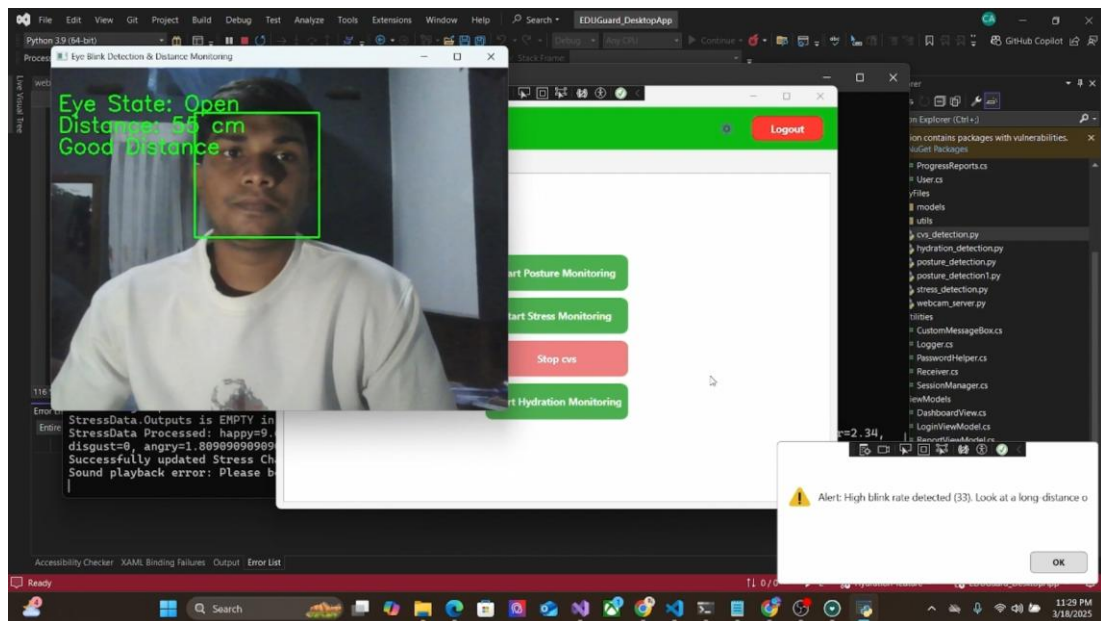


Figure 17: : Eye Strain Detection Component Results

5.2 Discussion

The EDUGuard system was built with a clear purpose: to help online learners take better care of their physical and mental health while studying. Spending hours in front of a screen might seem normal now, but it often leads to problems like stress, poor posture, dry eyes, and even dehydration—issues that are easy to overlook until they start affecting productivity and well-being. Through the use of real-time monitoring powered by machine learning and deep learning techniques, EDUGuard brings together several smart components to tackle these concerns in a non-intrusive and personalized way.

Each component of the system was tested independently and together, and the results showed how even small, well-timed reminders could change daily habits for the better. What really brought the system to life, though, was the desktop alert mechanism—quietly running in the background, ready to step in with a gentle nudge when it noticed something was off.

Stress Detection and How Real-Time Feedback Helps

The stress detection component used a convolutional neural network trained to recognize emotional expressions on a person's face. While its accuracy of 78% isn't perfect, it proved good enough to detect moments when students were feeling overwhelmed or distracted. The system looked at changes in facial emotions over time and then assigned stress levels based on those patterns.

Rather than just recording the stress levels, the system did something more helpful—it offered support. When the score suggested the user was under stress, a small alert popped up saying things like, “Take a deep breath” or “How about a short break?” These reminders didn't interrupt work but gave users a moment to reset. In fact, many students reported that just being made aware of their stress helped them feel more in control of it. That's one of the strengths of using machine

learning in this way—not to replace human decisions, but to guide and support better ones.

Posture Monitoring That Actually Changes behaviour

One of the most effective components was the posture monitoring system. It used a combination of two approaches: a Random Forest classifier trained on posture angles from Mediapipe, and a YOLOv8 deep learning model trained to visually detect good and bad sitting positions. Each model had its strengths—the Random Forest was quick and interpretable, while the YOLOv8 model was more flexible and accurate under various lighting and background conditions.

But the real magic happened when they were combined. This hybrid model made decisions more confidently and reduced the chance of false alerts. The system didn't just point out bad posture—it helped correct it. Alerts like “Sit upright” or “Check your shoulder position” popped up after a few seconds of slouching. According to user feedback, these small reminders led to genuine habit changes. Over time, students began adjusting their posture automatically without needing to be told.

Computer Vision Syndrome (CVS) Detection That Works Silently

The CVS detection module was focused on eye strain—a problem that many students accept as part of online learning. But blinking less and sitting too close to the screen can cause real discomfort. This component used a lightweight deep learning model to monitor eye openness, track blink frequency, and estimate the user's distance from the screen.

With a model accuracy of 93%, the system was reliable in identifying fatigue-related behavior. It didn't wait for users to complain about discomfort. Instead, it

would quietly pop up a message like “Try blinking a few times” or “You’re sitting a bit too close.” These alerts helped students become more aware of how they were using their screens, and most users said they felt less eye strain after just a few sessions.

Dehydration Monitoring: Simple but Impactful

Dehydration might not be the first issue you think about while studying, but it plays a big role in concentration and energy. This module combined two detection strategies: a lip dryness classifier based on deep learning, and an object recognition model to detect drinking behavior, like picking up a water bottle.

Together, they made a powerful team. If a student hadn’t shown signs of drinking water for a while and their lips looked dry, the system sent a hydration reminder like “Drink some water to stay fresh.” These simple alerts led to noticeable improvements—students who used the system said they drank more water and felt more focused. It showed how machine learning can support even the smallest yet most important health habits.

The Desktop Alert System: Quiet, Smart, and Supportive

What made all of these modules work so well together was the real-time desktop alert system. Rather than just collecting data, the system acted on it. Alerts were short, respectful, and color-coded based on urgency—green meant things were fine, yellow meant a warning, and red indicated immediate action was needed. Students could also adjust how often they wanted to be reminded.

This feature made EDUGuard feel less like a monitoring tool and more like a quiet digital assistant. It didn’t nag or interrupt—it simply reminded users of the small

things they could do to take better care of themselves. And according to user surveys, this was one of the most appreciated parts of the entire system.

The Bigger Picture

Developing EDUGuard showed us that it's possible to create a meaningful wellness tool using just a webcam and some smart machine learning techniques. We didn't need expensive sensors or wearables—just thoughtful algorithms, useful feedback, and a user-first design. Each model played its part, but the real value came from how they all worked together to improve students' everyday habits.

What also stood out was how much users appreciated the system not just for what it detected, but for how it communicated. Friendly messages, personalized settings, and simple language made EDUGuard feel approachable. This balance between technology and empathy is what helped the system succeed—not just as a project, but as something that students actually wanted to use.

There's still more we can do. Adding mobile support, improving lighting adaptation, or even introducing voice feedback could take this system further. But even now, EDUGuard proves that with the right combination of deep learning, machine learning, and real-time feedback, we can design tools that look out for students—just like a friend would.

6 CONCLUSION

EduGuard began with a simple yet powerful question: how can we protect the physical and mental well-being of students during prolonged periods of online learning without adding more burden to their already busy lives? As students ourselves, we've lived the reality of eye strain after hours of staring at screens, the aching necks from poor posture, the invisible weight of stress, and even forgetting to drink water during back-to-back virtual classes. We knew this wasn't just a project it was personal. And that human connection shaped every decision we made, from system design to user experience. Our mission was not just to create a tool, but to build a silent guardian a desktop companion that could observe, learn, and gently intervene when our health began to slip through the cracks. EduGuard is the embodiment of that mission.

One of EduGuard's greatest strengths lies in its modular but unified design. Rather than building isolated systems for posture, eye strain, stress, and dehydration, we created an environment where these modules coexist each monitoring a different health signal yet working together harmoniously. Each model was independently trained and optimized using cutting-edge machine learning and computer vision techniques, but they all report to a single, friendly interface that communicates with the user in real time.

By using a combination of Python and C# (WPF), we ensured that the backend models could perform complex calculations while the frontend remained responsive, interactive, and visually intuitive. A local TCP socket server enabled seamless, real-time communication between the modules, ensuring users received alerts and feedback within moments right when it mattered most. This structure allowed for customization and scalability. Whether a user wants to monitor all health aspects or just one, EduGuard adapts to their needs without unnecessary complexity.

Each module in EduGuard tackled a different human challenge with care and precision.

- The stress detection component read facial expressions, turning furrowed brows and tense muscles into emotional insights, using EfficientNetB0 to detect changes invisible to the naked eye.
- The CVS detection module tracked eye movements, blinking patterns, screen distance, and screen time to flag signs of digital eye strain, providing timely alerts to avoid long-term damage.
- The hydration monitors gently watched for signs of lip dryness, encouraging users to stay hydrated with intelligent, non-intrusive suggestions.
- And the posture detector quietly observed how users sat, recognizing when slouching became habitual and nudging them back toward healthier positioning.

Together, these features created a 360-degree health shield around users, detecting risks early and offering actionable feedback to promote recovery not punishment, but partnership.

EduGuard wasn't just about responding to issues in real-time. It was about creating awareness—helping users see patterns in their behaviour that they might otherwise miss.

All predictions and alerts were stored securely in MongoDB, organized under progress reports. These logs could be accessed by users at any time, giving them a dashboard of their well-being trends over days or weeks. We built in visualizations and summary charts that allowed students to reflect on questions like:

- Was I more stressed before deadlines?
- Did my hydration habits drop during exam week?
- Do I tend to slouch more when I'm tired?

These aren't just technical graphs. They're mirrors helping students better understand themselves and take ownership of their wellness.

Throughout our journey, we never lost sight of our target audience students just like us. We made conscious choices to ensure affordability, ease of use, and inclusivity.

We avoided expensive wearables. We ensured cross-platform compatibility. We added toggles and customization settings to let users feel in control not monitored.

We also emphasized ethical responsibility. Privacy is paramount. No raw images are stored, only processed outputs. Users choose what to track and when. The application was built with trust in mind trust that EduGuard is here to help, not judge.

In real-life testing, we received feedback that made all our hard work worthwhile: “It felt like having a gentle coach reminding me to take care of myself,” one user said. That’s the heart of EduGuard. EduGuard may have started as a university project, but it has the potential to evolve far beyond. The future could include:

- Mobile sync features, so users receive alerts even when away from their desktop.
- AI-driven recommendations, like suggesting breaks, breathing exercises, or ergonomic adjustments.
- Cloud-based insights for parental or teacher monitoring (with consent), promoting health accountability in academic settings.

But even as we dream big, we remain grounded in our mission: to create technology that supports human well-being. Not flashy, not intrusive just quiet, consistent care.

In a world where education is increasingly digital, EduGuard offers something profoundly human a reminder to pause, breathe, stretch, and sip water. A reminder that even while pursuing knowledge, our health matters most. We are proud of what we’ve built. And we’re even prouder that it was built with empathy at its core.

7 REFERENCES

1. A. Dementyev and C. Holz, "DualBlink: A Wearable Device to Continuously Detect, Track, and Actuate Blinking For Alleviating Dry Eyes and Computer Vision Syndrome," *ACM Journals*, vol. 1, no. 1, pp. 1–19, Mar. 2017.
2. M. T., B. P., and Z. F., "Blink Detection for Real-Time Eye Tracking," *J. Netw. Comput. Appl.*, vol. 25, pp. 129–143, Apr. 2002.
3. J. Almeida and F. Rodrigues, "Facial Expression Recognition System for Stress Detection with Deep Learning," *Proc. 23rd Int. Conf. Enterp. Inf. Syst.*, 2021. [Online]. Available: <https://doi.org/10.5220/0010474202560263>
4. H. Gao, A. Yüce, and J.-P. Thiran, "Detecting Emotional Stress from Facial Expressions for Driving Safety," *IEEE Int. Conf. Image Process.*, pp. 5961–5965, 2014. doi: 10.1109/ICIP.2014.7026203.
5. S. Tivatansakul and M. Ohkura, "Improvement of Emotional Healthcare System with Stress Detection from ECG Signal," *37th Annu. Int. Conf. IEEE Eng. Med. Biol. Soc.*, pp. 6792–6795, 2015. doi: 10.1109/EMBC.2015.7319953.
6. M. Hosseini et al., "Multimodal Stress Detection Using Facial Landmarks and Biometric Signals," *arXiv preprint arXiv:2311.03606*, 2023.
7. WHO, "Physical Inactivity a Leading Cause of Disease and Disability," 2002. [Online]. Available: <https://www.who.int/news/item/04-04-2002-physical-inactivity-a-leading-cause-of-disease-and-disability-warns-who>
8. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Daejeon, Korea, 2016.
9. IEEE Int. Conf. Electron. Meas. Instrum. (ICEMI), 2017.
10. H. Daneshmandi, A. Choobineh, H. Ghaem, and M. Karimi, "Adverse Effects of Prolonged Sitting on the General Health of Office Workers," *J. Lifestyle Med.*, vol. 7, no. 2, pp. 69–75, 2017. doi: 10.15280/jlm.2017.7.2.69.

11. N.K.M. Yoong, J. Perring, and R.J. Mobbs, "Commercial Postural Devices: A Review," *Sensors (Basel)*, vol. 19, no. 23, p. 5128, 2019. doi: 10.3390/s19235128.
12. L. Zhao, J. Yan, and A. Wang, "A Comparative Study on Real-Time Sitting Posture Monitoring Systems Using Pressure Sensors," *J. Electr. Eng.*, vol. 74, no. 6, pp. 474–484, 2023. doi: 10.2478/jee-2023-0055.
13. M. Bayattork, M.B. Sköld, E. Sundstrup, and L.L. Andersen, "Exercise Interventions to Improve Postural Malalignments," *J. Exerc. Rehabil.*, vol. 16, no. 1, pp. 36–48, Feb. 2020. doi: 10.12965/jer.2040034.017

General References

[12] OpenAI GPT-4 Documentation – OpenAI: <https://beta.openai.com/docs/>

[13] Version Control Systems – GitHub: <https://github.com/>

[14] Design Tools – Figma: <https://www.figma.com/>

8 GLOSSARY

- **Angle x / Angle y**
Geometric angles calculated using body landmarks (shoulders and nose) to determine the correctness of a user's posture in the Random Forest-based posture classification model.
- **Blink Rate**
The frequency at which a user blinks during a period of screen usage, used to detect signs of Computer Vision Syndrome (CVS) and visual fatigue.
- **CNN (Convolutional Neural Network)**
A class of deep learning models used for image classification tasks such as emotion detection, eye state detection, and lip dryness classification.
- **CVS (Computer Vision Syndrome)**
A condition caused by prolonged screen use, characterized by symptoms like eye strain, blurred vision, and headaches. Monitored in the EduGuard system using blink detection and screen distance measurement.
- **Deep Learning**
A subset of machine learning involving neural networks with multiple layers, used in EduGuard for facial expression recognition, lip dryness detection, and posture detection.
- **EfficientNetB0**
A lightweight deep learning model architecture used in EduGuard's stress detection module to classify emotional states in real time.
- **Hybrid Classification Model**
A decision-making system in EduGuard combining both Random Forest (numerical posture classification) and YOLOv8 (image-based classification) to enhance reliability.
- **MediaPipe**
An open-source framework used for extracting human pose landmarks (e.g., nose, shoulders) in posture monitoring modules.
- **Random Forest**

An ensemble learning algorithm used in EduGuard to classify posture based on angles derived from body landmarks.

- Real-Time Monitoring

Continuous tracking of health metrics such as eye strain, stress, hydration, and posture, providing immediate feedback and alerts.

- TCP Socket

A protocol used for real-time communication between the C# frontend and Python-based backend modules in the EduGuard desktop application.

- TensorFlow/Keras

Machine learning and deep learning libraries used to build and train CNN models for emotion recognition, lip dryness detection, and other image-based classification tasks.

- Webcam-Based Analysis

The use of standard webcams to capture facial and posture data for health monitoring without requiring external or wearable devices.

- WPF (Windows Presentation Foundation)

A graphical subsystem used with C# to design the user interface of the EduGuard desktop application.

- YOLOv8 (You Only Look Once version 8)

An advanced object detection model used in EduGuard for visual posture classification and drinking behavior recognition.

9 APPENDICES

Appendix A – Model Architectures

A.1 Stress Detection Model (EfficientNetB0)

- Input: Grayscale facial image (48×48)
- Layers: Convolutional layers → Batch Normalization → ReLU → Fully Connected Layer
- Output: Emotion classification (7 classes) → Mapped to stress score
- Framework: TensorFlow/Keras
- Dataset: FERPlus

A.2 Lip Dryness Detection Model (CNN)

- Input: Preprocessed lip image (640×640 RGB)
- Layers: 4 Convolutional blocks → Flatten → Dense → Dropout → Sigmoid output
- Output: Dry (1) or Normal (0)
- Training Accuracy: 77.03%
- Validation Accuracy: 74.07%
- Dataset: Custom dataset with healthcare-labeled lip conditions

A.3 Posture Detection Model (Random Forest + YOLOv8n Hybrid)

- RF: Based on two computed angles using shoulder and nose landmarks
- YOLOv8n: Lightweight deep learning model for visual classification of good/bad posture
- Fusion Logic: Rule-based decision using model confidence levels
- Dataset: 402 labeled images under diverse lighting conditions

A.4 Eye State Detection Model (CNN for CVS Detection)

- Input: Eye region extracted via MediaPipe/OpenCV
- Layers: CNN layers with softmax output for Open/Closed classification
- Used for blink frequency and fatigue estimation
- Accuracy: 93%

Appendix B – User Interface Screens

B.1 Login UI (Figure 10)

- Secure user authentication
- Entry point to all system features

B.2 Component Start UI (Figure 11)

- Module-wise start/stop toggle
- Visual indicator for active monitoring

B.3 User Profile Update UI (Figure 12)

- Allows personalization of alert thresholds
- Includes hydration frequency and posture duration settings

B.4 Report UI (Figure 13)

- Charts and graphs showing trend data per module
- Supports daily, weekly, and monthly summaries

Appendix C – Dataset Composition

C.1 Lip Dryness Dataset

- 3,304 images total
 - Normal: 1,649 (1,319 training / 330 validation)
 - Dry: 1,655 (1,324 training / 331 validation)

C.2 Posture Dataset

- 402 captured images
 - Balanced samples of correct and incorrect posture
 - Annotated using ergonomic guidelines

Appendix D – System Architecture and Communication

D.1 Local Socket Communication Setup

- Host: 127.0.0.1
- Port: 9999
- Protocol: TCP socket
- Communication between C# frontend and Python backend models

D.2 Database Schema (MongoDB)

- Collections:
 - users: User profile and login credentials
 - reports: Health trend summaries
 - predictions: Raw predictions with timestamps, component name, confidence

Appendix E – Alert Examples

E.1 Desktop Notifications (Examples):

- "High stress detected—Take a short break"
- "Dry lips detected—Please drink some water"
- "Poor posture detected—Sit upright"
- "Reduced blinking—Try 20-20-20 rule"