Focus Tracker App

Authors: Arnav Arora, Karen Li, Rohan Sonecha
Affiliation: Electrical and Computer Engineering, Carnegie Mellon University

***Abstract*—Navigating productivity in the digital workplace is increasingly challenging due to pervasive distractions. Traditional tools for enhancing focus fall short, offering only surface-level solutions like app blockers or time trackers. The Focus Tracker App introduces a novel approach by leveraging EEG and visual data for a comprehensive analysis of the user's focus state and surrounding distractions. It employs machine learning algorithms to identify and categorize distractions with a precision rate exceeding 70% in F-score and a 90% recall rate. Designed for the modern work environment, the app provides real-time feedback within 3 seconds, aiming for a 90% user satisfaction score.**

***Index Terms*—Focus State, Productivity Score, Distraction, EEG, Convolutional Neural Network, Object Detection, Facial and Hand Landmarking**

1 INTRODUCTION

In today's digital era, where social media and instant connectivity are at the forefront, maintaining sustained focus has become a formidable challenge for many. Additionally, there exist distractions beyond just the digital realm, including ambient noises, impromptu discussions, constant emails and work-related communications, and even physical discomforts. Recognizing the need to combat these pervasive distractions is crucial in fostering a more productive work environment.

Current productivity technologies depend heavily on the user's ability to self-regulate and follow through with the app's recommendations or features. While some tools [5] [4] can track app usage or screen time, they often do not delve deep enough into analyzing the patterns of distraction or the root causes behind them. Additionally, a significant number of focus-enhancing technologies offer generic solutions that do not account for the individual differences in work habits, environments, and the nature of distractions faced by users.

The Focus Tracker App enables users to measure their focus and associated distractions during work sessions to help them identify actionable steps to improve productivity. It stands out from other productivity tools by offering a dual-purpose solution: it not only quantifies focus levels in real-time but also identifies and categorizes potential distractions that may impede one's workflow. This approach allows users to gain a comprehensive understanding of their work habits, providing valuable insights into the factors that disrupt their focus. This app utilizes an EEG headset and a web camera, coupled with machine learning algorithms, to accurately detect focus levels and identify distractions in real time. This integration transforms the user experience, offering interactive graphs and user interfaces that visually represent their focus patterns, enabling users to pinpoint specific distractions and develop targeted strategies to mitigate them.

With its emphasis on accessibility and user-friendliness, the Focus Tracker App aims to change the way individuals approach their work sessions. By providing real-time feedback and actionable insights, the app is a valuable resource for anyone looking to enhance their focus, improve their productivity, and achieve a healthier work-life balance in today's fast-paced digital world.

2 USE-CASE REQUIREMENTS

The primary objective of the Focus Tracker App is to develop a web application aimed at assisting users in monitoring and enhancing their focus and productivity during work sessions. By measuring Focus State, environmental distractions, and distracted behaviors, and then informing the user, we help them understand how their focus varies over time and what is holding them back. This empowers users to take actionable steps to improve their focus. From a social and well-being perspective, the Focus Tracker App addresses concerns relating to distractions and lack of focus in work environments. The information we provide to the user can empower them to improve how they feel about their work and improve their mental well-being and productivity in work environments. Also, by helping individuals improve their focus and productivity, the Focus Tracker App can contribute to overall efficiency in the workplace, which can have a positive global economic impact.

Overall, the Focus Tracker App must feature accurate, real-time monitoring of Focus State and distractions. It must also provide a usable interface that allows for real-time monitoring of work sessions as well as the history and analysis of previous work sessions. It will provide the user with their Focus State (focused vs. not focused), and will detect: (a) yawning, (b) microsleeps, (c) off-screen gazing, (d) interruptions from others, (e) restlessness, and (f) phone pick-ups. It will also provide the user with a Productivity Score at the end of a work session. To provide accurate monitoring of focus and productivity, we require that at least 90% of users find the generated Focus State and Productivity Score for a work session to match their personal assessment. To provide accurate behavior and distraction detection, we require an F-score of at least 70% and a recall of at least 90%. A higher recall is required because the cost of false negatives is higher–it is more costly to miss the detection of a behavior or distraction. To ensure real-time monitoring, we require a latency of at least less than 3 seconds between data capture and data analysis (Focus State generation and distraction detection). We also require that at least 90% of users find the user experience to be seamless and easy to use.

3 ARCHITECTURE AND/OR PRINCIPLE OF OPERATION

The Focus Tracker App will be built using Python. For camera-based detection, we will read camera data using OpenCV and pass images to existing libraries, such as YOLOv8 and MediaPipe, for object detection, facial detection, and face and hand landmark detection. After the initial processing of the images, data will be passed to the distraction and behavior detection algorithms we developed. This includes phone pick-ups, microsleeps, yawning, gaze, human disruption, and restlessness detection.

For the EEG-based focused detection, we will read the power values from each of the frequency bands from the AF3 and AF4 sensors which correspond to the prefrontal cortex which is closely related to focus levels. Based on the reported EEG quality, we will filter out noisy readings to avoid making a low quality determination of Focus State. If the EEG quality is high enough, we will pass the power readings for each of the frequency bands and the AF3/AF4 sensors through the learned model and report the Focus State returned as output from the model.

Django operates as the backend framework orchestrating data flow and application logic. It interacts with the PostgreSQL database to manage and store data, including user profiles, session metrics, and real-time analysis results from the EEG and camera inputs. Django will also process the raw data from these devices, filtering and organizing it into a structured format that can be used for further analysis or immediate feedback. The Django REST framework is leveraged to create API endpoints, which allow for the data transfer between the server and the client-side application. React operates as the frontend and it fetches data from the Django backend through API calls. React will handle live updates, such as changes in focus levels or notifications of detected distractions, and display the analysis of focus-related metrics through interactive dashboards.

Figure 2 presents the block diagram describing this architecture.

To use the Focus Tracker App, the user will begin by opening the app to the Home page 3a. They will click the New Session button to initiate a new session. This will take them to a Calibration page 3b, where the user will receive instructions on how to properly secure the Emotiv headset to their head and ensure accurate EEG readings. Camera calibration will begin, ensuring the camera's field of view is wide enough to detect the user picking up their phone and the face. The calibration stage will also ask the user to close their eyes and yawn to determine the detection thresholds for the individual user. Once calibration is complete, the work session will start and the user can monitor their work session in real-time on the Current Session page 3c. The user will see their video feed and real-time updates as distractions and behaviors are detected. There will also be a graph showing the user's Focus State over time and the detected distractions and behaviors. When the session is complete, the user will press the stop button, which will bring them to the Session Summary page 3d. Here, the user will be provided with a Productivity Score, summarizing the overall productivity of the session. It will also display the amount of time focused vs. distracted, a graph of Focus State over time, and any top distractions and behaviors detected. Lastly, the user can view the Session History page 3e, which summarizes the user's previous work sessions. This page informs the user of their progress over work sessions.

4 DESIGN REQUIREMENTS

The design requirements of the Focus Tracker App, including both camera-based detection and EEG-based focus level tracking, align with the user requirements for accurate, real-time monitoring of Focus States and distractions, complemented by a user-friendly interface to bring it all together.

Utilizing the TedGem 1080p camera with a processing rate of at least 10 fps ensures high-quality real-time monitoring of physical indicators of distraction or loss of focus, such as yawning, microsleeps, off-screen gazing, interruptions from others, restlessness, and phone pick-ups. This capability aligns with the user requirement for real-time detection of distracted behaviors, aiming for an F-score of at least 70% and a recall of at least 90% to minimize the risk and impact of false negatives. This ensures that nearly all instances of distraction are accurately identified, aligning with the use case requirement for accurate behavior and distraction detection. With a mean absolute error threshold of $\leq 5^{\circ}$ for the yaw in head pose estimation, the app can precisely assess where the user is looking and whether the user's attention is directed toward their work, providing an additional layer of Focus State analysis. The app will require a facial recognition accuracy of $\geq$95% and maintain a false positive rate of $\leq$5%, adeptly distinguishing between the user and others, thereby identifying interruptions from others. We will use the YOLOv8 model for phone object detection, requiring an average precision of $\geq$85%, ensuring highly accurate detection of phone pick-ups, a key indicator of distraction.

By integrating EEG power data captured at 5 samples per second per channel and frequency band, the app will provide a direct measurement of the user's Focus State (focused, distracted, or neutral). This sampling rate combined with filtering out noisy data where the reported EEG quality is too low to glean any signal will allow us to train a model to detect Focus State using Professor Jocelyn Dueck's labels as our ground truth. Professor Dueck is an expert in identifying Focus States in her piano students and is working closely with us to generate high-quality training data. In terms of integration and user experience, the application is designed to present both real-time data and historical analysis of work sessions through a clear and intuitive interface. This ensures that users can easily engage with the app to monitor their current Focus State and review their focus and productivity trends over time.

By combining the insights from camera-based behavior detection and EEG-based focus tracking, the app can accurately determine the user's Focus State and calculate a Productivity Score that reflects their performance during each work session. This Productivity Score, alongside the Focus State, is designed to closely match users' personal assessments, meeting the requirement that at least 90% of users find these metrics accurate. This level of accuracy is achieved through the high F-score and recall targets set for distraction detection and Focus State analysis.

The real-time monitoring system is optimized for low latency, with a requirement for less than 3 seconds between data capture and analysis, ensuring that users receive immediate feedback without noticeable delays. The user interface is crafted to be intuitive and accessible, enabling users to navigate the app effortlessly and making the technology accessible to a broad audience. This aligns with the requirement that at least 90% of users find the app seamless and easy to use.

5 SYSTEM IMPLEMENTATION

5.1 Camera-Based Distraction and Behavior Detection

Camera-based distraction and behavior detection will detect (a) yawning, (b) microsleeps, (c) off-screen gazing, (d) interruptions from others, (e) restlessness, and (f) phone pick-ups. These algorithms for detection are implemented in Python and begin with reading images from a 1080p external web camera via OpenCV. For (a) yawning and (b) microsleep detection, the images are passed into MediaPipe's facial landmark detector [7]. The mouth aspect ratio and eye aspect ratio describe how open the eyes and mouth are [10]. Using eight distinct points on the mouth (MediaPipe landmarks 61, 39, 0, 269, 291, 405, 17, 181) and six distinct points on the eyes (MediaPipe landmarks 33, 160, 158, 133, 153, 144), we can calculate the mouth aspect ratio and eye aspect ratio. For the yawning and microsleep detection to work on all users with different eye and mouth shapes, the program starts with calibration. It first measures the ratios on a neutral face and then measures the ratios when the user is yawning and when the user's eyes are closed. This is used to determine the corresponding thresholds. The ratios are normalized by calculating a Z-score for each measurement when the program is running detection. Head pose estimation [6] is used to detect (c) off-screen gazing. The Perspective-n-Point pose computation problem is solved to calculate the rotation of the head in space. The goal is to find the rotation and translation that minimize the reprojection error from 3D-2D point correspondences. Five points on the face are used for this correspondence: two points on the outside of the eyes, one point on the nose, two points on the outside of the mouth, and one on the chin (MediaPipe landmarks 1, 9, 57, 130, 287, 359). The 3D coordinates of a face looking forward without any rotation are known, and the 2D coordinates are obtained through MediaPipe's facial landmark detector. Using helper functions from OpenCV and linear algebra principles, the rotation matrix is converted to Euler angles, providing the roll, pitch, and yaw of the head. The yaw describes how far the user is gazing to the left and right, and the pitch describes how far the user is gazing up and down. Detecting (d) interruptions from others involves detecting other faces in the frame. This will require distinguishing between the user and any other people that are detected in the frame. The Face Recognition library [3] will be used for this task. MediaPipe's pose landmark detector [7] will be used for (e) restlessness detection. The pose landmark detector detects the position of various landmarks on the human body, including the elbows, shoulders, and several points on the face. By comparing the positions of landmarks across consecutive frames, the system can calculate the magnitude and direction of movements, quantify the user's motion, and determine when the user has become restless. Detecting (f) phone pick-ups will involve a combination of object detection and hand landmark detection. The YOLOv8 object detector [14] will be trained on the MUID-IITR dataset [11], a dataset of images of people using smartphones while performing day-to-day activities. The object detection will be combined with MediaPipe's hand landmark detector [7], which will provide information about the position of the hand in frame, to determine when the phone has been picked up and is being used.

5.2 Focus State Detection

The first step in Focus State detection is the data collection phase. We have implemented a system for Professor Dueck to work with which allows us to collect power data from each of the frequency bands while one of her students wears the headset and Professor Dueck labels her student as focused, distracted, or neutral with sub-second granularity. Professor Dueck coaches her students in collaborative piano and has spent a significant amount of time trying to understand flow states in music, which she describes as analogous to a flow state in sports and work settings. Given her many years of experience seeking out this elusive state, she has developed a knack for identifying when her students are focused, distracted, or neutral with impressive speed and accuracy which makes her an invaluable asset to this project for determining ground truth [2]. Unfortunately, while the contact quality reported by the headset tends to stabilize at 100 (highest possible score), the EEG quality is a bit finicky and even once it reaches 100, it will randomly drop down to 0 and then jump back up. In order to avoid training our model on low quality data when the EEG quality has dropped significantly, we filter out noisy readings before training the model. Furthermore, the AF3 and AF4 sensors are located on the user's forehead which means the data from these sensors will correspond closely to the activity of the prefrontal cortex which is closely related to focus, so we will be specifically looking at the readings from these sensors [1]. The EmotivPRO software takes the raw time-series EEG data and translates it into the frequency domain, outputting power values for 5 frequency bands from each of the 5 sensors on the headset. The frequency bands delta (1–3 Hz), theta (4–7 Hz), alpha (8–12 Hz), beta (13–30 Hz) correspond to sleep, deep relaxation/inward focus, relaxation/passive attention, and active/external attention respectively [12]. We will filter out readings with low EEG quality at the AF3 and AF4 sensors and then pass in a matrix into the model as input. The matrix input will be made up of individual columns which will contain the power readings from the prefrontal cortex sensors at individual time stamps. There will be 25 columns because of our decision to pass in 5 seconds worth of samples and the 5 samples per second sample rate for the power readings. We plan to develop a custom model using PyTorch to apply a combination of fully-connected and convolutional layers to train a network that will detect Focus State and output either focused, distracted, or neutral to the user. We chose to implement a convolutional neural network so that the model can capture spatial and temporal patterns. The convolutional layers will extract high-level features, pooling layers will compress feature maps while preserving information to reduce noise from the EEG data collection, and the fully-connected layers will learn non-linear combinations of the high-level features from the outputs of the convolutional/pooling layers [13]. We plan to experiment with how many of each layer to include in the final network which will be determined based on model performance as well as computation/time efficiency during inference. Once the model is trained, we use the Emotiv Cortex API to subscribe to live data which is passed through the inference model and the output is forwarded to the user. Similar to our filtration of low quality readings during training, we will filter out low quality readings during inference so as not to make low confidence predictions based on noise. We will pass in the same time period worth of data during inference as we determine to be optimal in the training phase (between 5-10 seconds worth of samples) as input to the model which will define the granularity of our predictions.

6 DESIGN TRADE STUDIES

6.1 Camera-Based Distraction and Behavior Detection

One option that was considered for the camera-based detection system was a series of image binary classifiers. For example, for yawning detection the mouth would be extracted from the image and classified as open or closed. The same process would be used to classify eyes as open or closed for microsleep detection. After further investigation, we proposed another solution of using an existing library to extract the position of face landmarks to determine if the eyes or mouth are opened. This would decrease the engineering cost and remove the need to collect large amounts of data to train the image classifiers. We would then be able to focus more effort on detecting a larger number of distractions and behavior types. We could have similarly applied image classification to determine if the user is looking away from the screen, but instead opted for using a combination of face landmarks to solve the Perspective-n-Point problem, providing us with the exact roll, pitch, and yaw of the head positioning. This provides finer granularity and an actual measure of the user's head angle, rather than a positive or negative marker for off-screen gazing. Two libraries were considered for detecting face, hand, and body landmarks: MediaPipe and Dlib. We opted for MediaPipe due to several advantages that MediaPipe has over Dlib. Firstly, MediaPipe has higher accuracy and speed than Dlib when applied to determine eye state using face landmark detectors [9]. Notably, MediaPipe performed at 120 fps while Dlib performed at only 60 fps. MediaPipe also provides a much higher resolution for its facial landmarks, estimating 468 3D facial landmarks vs Dlib's mere 68 2D facial landmarks. This provides much more flexibility when working with facial landmarks. MeidaPipe also offers built-in, publicly available models not only for facial landmark detection but also for hand and body pose landmark detectors. Furthermore, MediaPipe's documentation is much more thorough than Dlib's documentation and provides extensive code examples for each of the provided detectors (face landmark, hand pose landmark, and body pose landmark detectors). Overall, these factors contribute to MediaPipe being the preferred choice for detecting face, hand, and body landmarks in the camera-based distraction and behavior detection system.

6.2 EEG-Based Focus State Detection

We selected the Emotiv Insight EEG headset because of its balance of high quality readings and cost effectiveness. The Insight headset has 5 sensors which places it right in the middle of other EEG headset options which have anywhere from 1-32 sensors to measure EEG signals from different regions of the brain. For our purposes, 5 sensors are plenty and because the Emotiv Insight was already in the ECE Inventory, we decided to move forward with this headset. Before deciding to measure the raw power values from each of the frequency bands from the AF3 and AF4 sensors on the headset, we were considering using the Emotiv Performance Metrics to either train a model or compute some sort of correlation metrics. The Emotiv Performance Metrics output numerical values for brain states such as attention, interest, boredom, cognitive stress, and others which we thought could provide interesting insights for detecting focus. For example, as cognitive stress rises, does focus increase or decrease? While this seemed like an interesting approach, we were concerned regarding the fidelity of the readings for these performance metrics because they would cut in and out with EEG quality fluctuating. We also found that the performance metrics are only reported every 10 seconds which is too infrequent for the level of granularity we want to display to the user. We realized that the raw power values are reported 5 times a second which would yield significantly more data and would enable us to pick up on the level of granular shifts in focus that Professor Dueck notices in her students and we hope to identify for our users. We also spent some time playing with the data we collected in initial phases of data collection to determine how to deal with readings with low EEG quality. We were initially concerned that if we filtered out all the data where the overall EEG quality was less than 100, we would not have enough data to train our model. However, after doing some simple analyses of the data and looking over how many samples actually had EEG quality of 100, we realized that instead of looking at the overall EEG quality, we could just look at the EEG quality of the AF3 and AF4 sensors since those would provide the signals relevant to focus detection. With this in mind, we found that those sensors did in fact have a full score for EEG quality for a significant amount of samples, so we decided to filter out any reading with an EEG quality less than 100 for both AF3 or AF4. In terms of the neural network design, we decided to use PyTorch for its simplicity and the fact that we have worked with it before. We considered other options such as Keras and Scikit which are also easy to use, but we had more familiarity with PyTorch. We will experiment with fully-connected and convolutional layers to see what yields the best performance in terms of focus detection. Because there is not much documentation or past details on past work using EEG, we expect our implementation to be closely coupled with our testing process and plan to be flexible with many of our key decisions as we continue to collect data and further understand the constraints we are working with.

7 TEST & VALIDATION

7.1 Camera-Detection Tests

Camera-Detection tests will consist of accurately processing high-quality images, detecting specific user behaviors, and identifying objects with precision.

To validate the app's ability to process video at 10 frames per second (fps) with a 1080p resolution, we will conduct tests that measure the average time taken between consecutive frame processing. This involves capturing the video, processing the video through the app, and calculating the average processing time per frame.

For testing the following behaviors: Yawning, Microsleeps, Off-Screen Gazing, Interruptions from others, Restlessness, and Phone Pick-Ups, a combination of custom and publicly available datasets, such as the Head Pose Image Database for head pose estimation and the MUID-IITR dataset for phone object detection, will be used. Each dataset will be split into training and testing subsets to train the model and then evaluate its performance. The focus will be on achieving an F1 score of at least 0.7 and a recall of at least 0.8 for the detection of these behaviors, reflecting the higher cost of false negatives. Specifically, the system's ability to detect interruptions from others will be tested using frames that include and exclude other individuals, aiming for $\geq 95\%$ accuracy in facial recognition and a $\leq 5\%$ false positive rate.

The YOLOv8 algorithm for phone object detection will be tested using the MUID-IITR dataset, focusing on its ability to detect phone usage within the frame. The testing will aim for an average precision of $\geq 85\%$, ensuring the app can accurately identify instances of phone pick-ups in real-time. In the case that training our own object detector is not fruitful, we have a mitigation plan to use an pre-trained object detector on the Roboflow platform [8].

7.2 EEG-Headset Tests

To test the EEG-Headset, we will conduct a series of 10 trials involving different individuals tasked with performing a range of focused-based activities (including puzzling solving, memory games, short coding exercises, etc.). After each session, participants will be surveyed to rate, on a scale from 0 to 10, how accurately they felt the headset's feedback matched their subjective experience of focus. The objective is to achieve a minimum average satisfaction score of 9 out of 10 across all participants.

7.3 Integration Tests

The integration tests will help evaluate the performance and reliability of various components used together in the Focus Tracker App, specifically the EEG headset and web camera.

To accurately measure and compare the latency across different stages of the Focus Tracker App, from data acquisition through EEG and camera to processing and display, we will break down the overall latency into discrete, measurable segments. First, we will test the Data Acquisition Latency, which measures the time taken from the actual EEG and camera data capture to the point where this data is available for processing. Since these devices will operate on different platforms, timestamping the exact moment data is captured and then made available to the processing unit will be crucial. We will use the Unix epoch time as the default timestamp format to ensure we can compare data between multiple devices. Next, we will test the Processing Latency. This is the time required to analyze the captured data, including Focus State determination and distraction detection. This will involve embedding timestamps at the entry and exit points of the processing algorithm. Lastly, we will test the Display Latency, which measures the time from the completion of data processing to when the results are visually represented on the user interface. We will timestamp the moment processing is complete and compare this timestamp to when the user interface is updated with the new information. To ensure that each of these latencies contributes appropriately to the total latency of less than 3 seconds, benchmarks will be established for each stage. Data acquisition latency should not exceed 0.5 seconds, processing latency should be within 2 seconds, and display latency should be under 0.5 seconds.

We will test the overall system to validate that the app accurately integrates real-time detection of user distractions and Focus States with the calculation of the Productivity Score and the overall Focus State assessment. We will simulate the different distractions and Focus States identified in the use-case requirements and use both the camera and EEG headset to capture user data. Test sessions will take place where users are subjected to various distractions at known intervals while trying to focus on a task. We will record the app's Focus State assessments and Productivity Scores for each session and compare them against the known occurrences of distractions and Focus States.

7.4 User and Frontend Tests

Users will test the overall application by participating in multiple focus sessions (each lasting 5-10 minutes long), during which various controlled distractions will be introduced. After each session, users will rate on a scale of 0-10 how accurately they feel the app tracked their focus and identified distractions, and how user-friendly and informative the user interface was. The objective is to achieve a 90% satisfaction rate over a series of 10 test sessions, conducted with a minimum of two distinct participants.

The calibration process is crucial for ensuring accurate data collection and analysis. We will assess the simplicity and intuitiveness of the calibration steps, aiming for users to complete the process without confusion or significant delays. A successful calibration test requires that at least 90% of users can independently calibrate both the camera and EEG headset within a given timeframe, with minimal instructions.

The graphs are a central feature for visualizing focus and productivity trends over time. This test will assess the graphs for clarity, accuracy, and user engagement. Effective graphs should allow users to easily understand their focus patterns and identify areas for improvement. We aim for at least 90% of users to rate the graphs as helpful and engaging, with specific attention to the ability to interact with the data for deeper analysis.

Lastly, we will evaluate the comprehensiveness and relevance of the information provided to the user, including the Focus State, Productivity Score, and detected distractions. The test will determine if users find the information actionable and if it aligns with their personal assessment of their productivity. Success in this area requires that at least 90% of users feel the information enhances their understanding of their work habits and aids in improving their focus and productivity.

8 PROJECT MANAGEMENT

8.1 Schedule

Our schedule is shown in Figure 4. In terms of camera-based distraction detection, we have implemented yawning and microsleep detection, head pose estimation, and are in progress on phone pick-up detection. For the EEG-based focused measurements, we have begun data collection with the pianists in Prof. Dueck's classes and have set up a platform to monitor Prof. Dueck's labels regarding her students' focus states. Finally, we have formulated some UI mockups and have begun integrating the outputs from the camera and EEG headsets so that they can be processed together and presented to the end-user.

8.2 Team Member Responsibilities

For the Focus Tracker App, the project components are structured into three categories: Frontend/ Backend Integration, Camera-based Detection, and EEG Headset-based Signal Processing. Each team member's responsibilities are tailored to leverage their skills effectively within these categories.

Arnav will work on the frontend and backend development, ensuring seamless integration between the user interface and the server-side logic. Arnav will also help develop the EEG data labeling platform for Professor Dueck and assist Rohan in training the machine learning models that will analyze the EEG data to assess focus levels.

Karen's role is centered around the visual components of the app, specifically the detection of distracted behaviors and environmental distractions using camera inputs. Karen will help identify and categorize various distractions, from physical movements such as eye gazing away from the screen to external environmental factors.

Rohan's role is centered around EEG-based focus detection aspect of the app. Rohan will work on processing EEG input signals to accurately detect the user's focus state. This involves developing algorithms to filter and analyze brainwave data and computing the duration users spend in focused versus unfocused states.

8.3 Bill of Materials and Budget

Please refer to Table 1 for our BOM.

8.4 Risk Mitigation Plans

In case the EEG Headset does not produce accurate results, we will shift to focus more on providing feedback and helping the user to increase their productivity. We plan to integrate microphone data, PyAudioAnalysis/MediaPipe for audio analysis, and Meta's LLaMA LLM for personalized feedback.

We will use the microphone on the user's device to capture audio data during work sessions and implement real-time audio processing to analyze background sounds and

Table 1: Bill of materials

Description	Part #	Manufacturer	Source	Cost @	Total
Web Camera	F22071	TedGem	ECE Inventory	$0	$0
EEG Headset	F21066	Emotiv	ECE Inventory	$0	$0
EmotivPRO Student License	F18500	Emotiv	EmotivPRO Website	$29/month	$116
					$116.00

detect potential distractions. The library PyAudioAnalysis will help us extract features from the audio data, such as speech, music, and background noise levels. MediaPipe will help us with real-time audio visualization, gesture recognition, and emotion detection from speech. PyAudioAnalysis/MediaPipe will help us categorize distractions based on audio cues and provide more insight into the user's work environment. Next, we will integrate Meta's LLaMA LLM to analyze the user's focus patterns and distractions over time. We will train the LLM on a dataset of focus-related features, including audio data, task duration, and other relevant metrics. The LLM will generate personalized feedback and suggestions based on the user's focus data.

In addition, we will provide actionable insights such as identifying common distractions, suggesting productivity techniques, or recommending changes to the work environment that will further help the user improve their productivity. Lastly, we will display the real-time focus metrics and detect distractions on multiple dashboards similar to the camera and EEG headset metrics we have planned.

To test the integration of microphone data, we will conduct controlled experiments where users perform focused tasks while the app records audio data. We will analyze the audio recordings to detect distractions such as background noise, speech, and device notifications. Specifically, we will measure the accuracy of distraction detection by comparing it against manually annotated data, aiming for a detection accuracy of at least 90%.

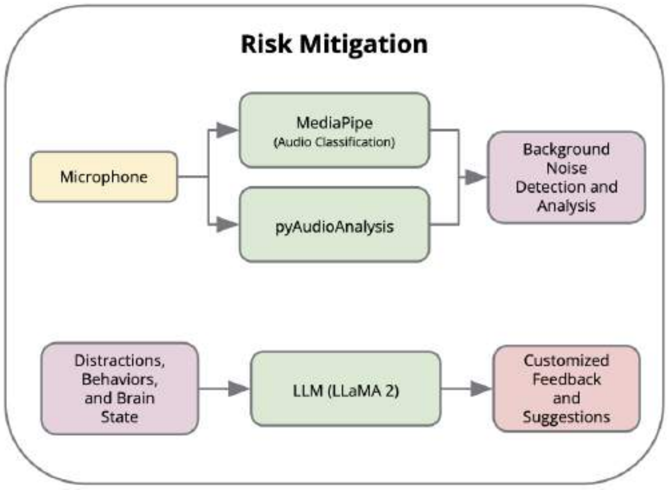

Risk Mitigation
Microphone
MediaPipe (Audio Classification)
pyAudioAnalysis
Background Noise Detection and Analysis
Distractions, Behaviors, and Brain State
LLM (LLaMA 2)
Customized Feedback and Suggestions

Figure 1: Risk Mitigation

9 RELATED WORK

Enhancing focus through technological means has seen various innovative approaches, particularly in leveraging biometric data to gauge attention levels. Research in this area includes the utilization of EEG technology to monitor cognitive states. RescueTime and Freedom are two well-known products that primarily focus on tracking digital usage and blocking distracting websites to improve user productivity. Products like the Muse headband provide users with feedback on their meditation and focus levels through EEG technology. Similarly, the Brainwave Visualizer by NeuroSky offers a direct visualization of brain activity, allowing users to observe changes in their attention and meditation levels.

The Focus Tracker App distinguishes itself by not only incorporating EEG-based monitoring of cognitive states but also by integrating computer vision to detect physical indicators of distraction, including eye movement and facial expressions. Leveraging both EEG data and visual cues sets it apart from other products by providing a more complete analysis of the user's focus state.

10 SUMMARY

The Focus Tracker App aims to improve workplace productivity by integrating EEG and visual data to monitor and enhance users' focus in real-time. Utilizing an EEG headset for brainwave analysis and a high-definition camera for monitoring visual cues of distraction, the app feeds data into a machine-learning model. This model, powered by convolutional neural networks for object detection and facial and hand landmarking, identifies and categorizes distractions with high precision. The user will be able to interact with the visual data through a display interface hosted on a user-friendly platform.

Implementing this system presents several challenges, including the accurate calibration of EEG and camera inputs to synchronize with the user's actual state of focus and distraction. Developing a machine learning model that can effectively process diverse data types—ranging from EEG patterns to visual distractions—and deliver reliable predictions is another significant obstacle. Additionally, ensuring the system's responsiveness, with a latency of less than 3 seconds for real-time feedback, is crucial to meet the use-case and design requirements. Meeting the target of a 90% user satisfaction score will require an intuitive and informative user interface design.

The Focus Tracker App can make a substantial impact on enhancing productivity and focus in the digital workplace. By offering personalized insights into work habits and providing actionable steps for improvement, the app addresses a critical need for solutions that go beyond traditional focus-enhancing tools.

References

[1] *A Neurosurgeon's Overview of the Brain's Anatomy.* URL: `https://www.aans.org/en/Patients/Neurosurgical-Conditions-and-Treatments/Anatomy-of-the-Brain#:~:text=The%20prefrontal%20cortex%20plays%20an,%2C%20concentration%2C%20temper%20and%20personality.`.

[2] *Dr. Jocelyn Dueck CMU Bio.* URL: `https://www.cmu.edu/cfa/music/people/Bios/dueck_jocelyn.html`.

[3] *Face Recognition.* URL: `https://pypi.org/project/face-recognition/`.

[4] *Focus Tracker Automatic Time Tracker.* URL: `https://www.focustrackerapp.com`.

[5] *Forest App.* URL: `https://www.forestapp.cc`.

[6] *Head Pose Estimation with MediaPipe and OpenCV in Javascript.* URL: `https://medium.com/@susanne.thierfelder/head-pose-estimation-with-mediapipe-and-opencv-in-javascript-c87980df3acb`.

[7] *Mediapipe.* URL: `https://github.com/google/mediapipe`.

[8] *mobilephone Object Detection.* 2022. URL: `https://universe.roboflow.com/m17865515473-163-com/mobilephone-wusj2`.

[9] Kiranraj Siva Shanmugam, Nasreen Badruddin, and Vijanth S Asirvadam. "Comparative Study of State-of-the-art Face Landmark Detectors for Eye State Classification in Subjects with Face Occlusion". In: *2022 IEEE 4th Eurasia Conference on Biomedical Engineering, Healthcare and Sustainability (ECBIOS).* 2022, pp. 24–27. DOI: `10.1109/ECBIOS54627.2022.9945018`.

[10] T. V. N. S. R. Sri Mounika et al. "Driver Drowsiness Detection Using Eye Aspect Ratio (EAR), Mouth Aspect Ratio (MAR), and Driver Distraction Using Head Pose Estimation". In: *ICT Systems and Sustainability* (2022).

[11] Lakshya Taragi et al. "MUID-IITR: Mobile-Phone Usage Image Dataset". In: (Jan. 2023).

[12] *The Science of Brainwaves = the Language of the Brain.* URL: `https://nhahealth.com/brainwaves-the-language/`.

[13] *Time Series Classification with Deep Learning.* URL: `https://towardsdatascience.com/time-series-classification-with-deep-learning-d238f0147d6f`.

[14] *Ultralytics.* URL: `https://github.com/ultralytics/ultralytics`.

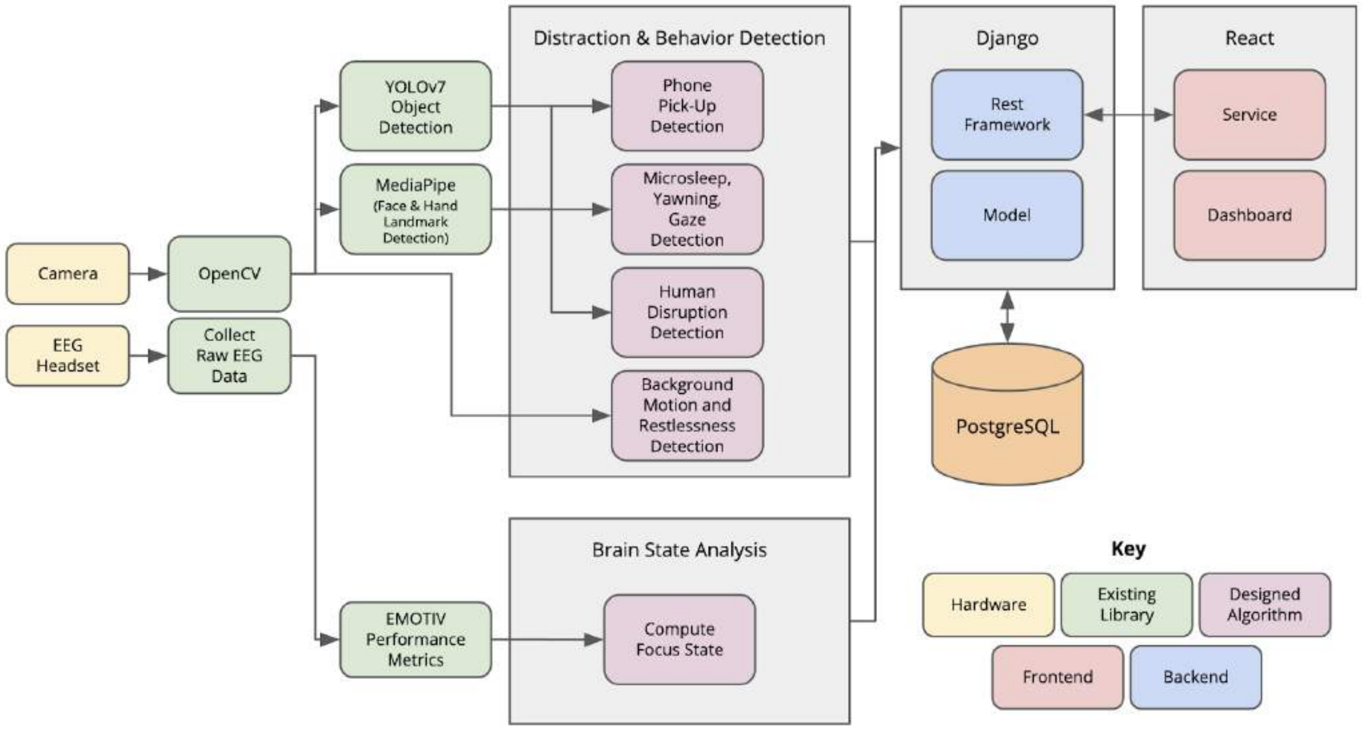
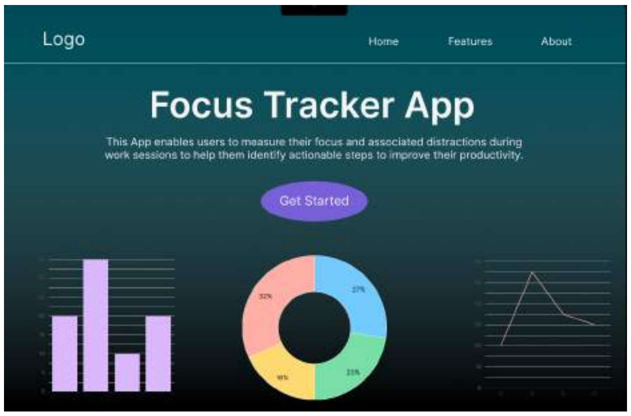
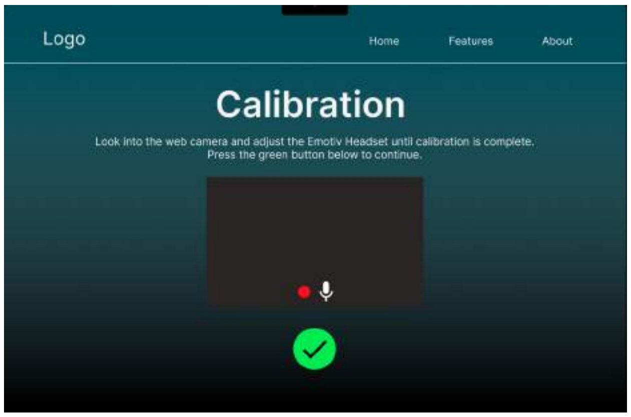


Figure 2: Block Diagram



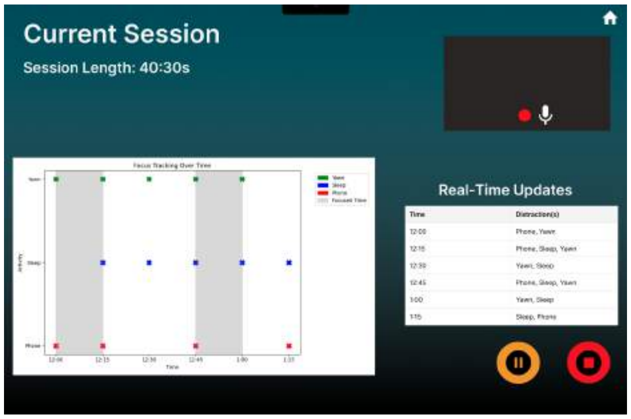
This App enables users to measure their focus and associated distractions during work sessions to help them identify actionable steps to improve their productivity.

(a) Home Page



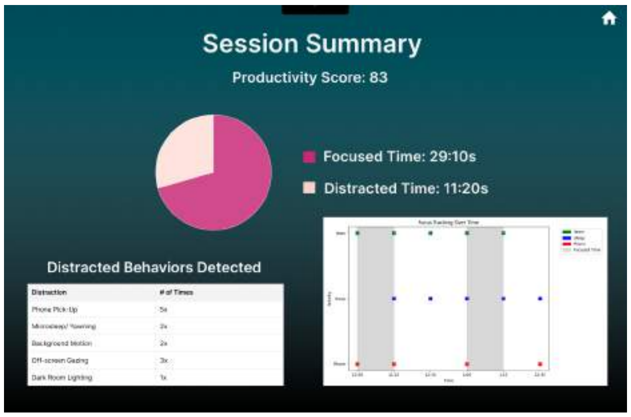
Look into the web camera and adjust the Emotiv Headset until calibration is complete. Press the green button below to continue.

(b) Calibration Page



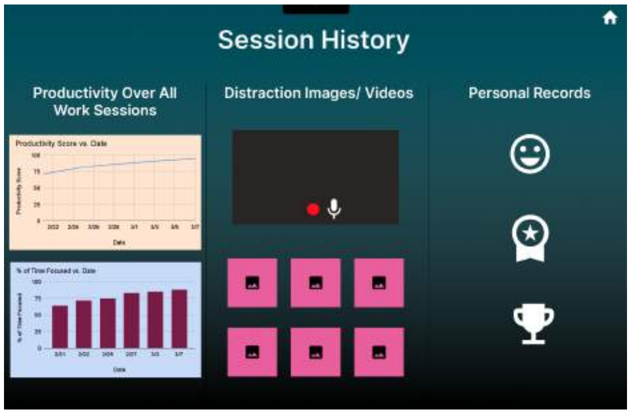
Time	Distraction(s)
12:00	Phone, Yawn
12:15	Phone, Sleep, Yawn
12:30	Yawn, Sleep
12:45	Phone, Sleep, Yawn
1:00	Yawn, Sleep
1:15	Sleep, Phone

(c) Current Session Page



Distraction	# of Times
Phone Pick-Up	5x
Microsleep/ Yawning	2x
Background Motion	2x
Off-screen Gazing	3x
Dark Room Lighting	1x

(d) Session Summary Page


Session History
Productivity Over All Work Sessions
Distraction Images/ Videos
Personal Records
Productivity Score vs. Date
% of Time Focused vs. Date
Date

(e) Session History Page

Figure 3: User Interface

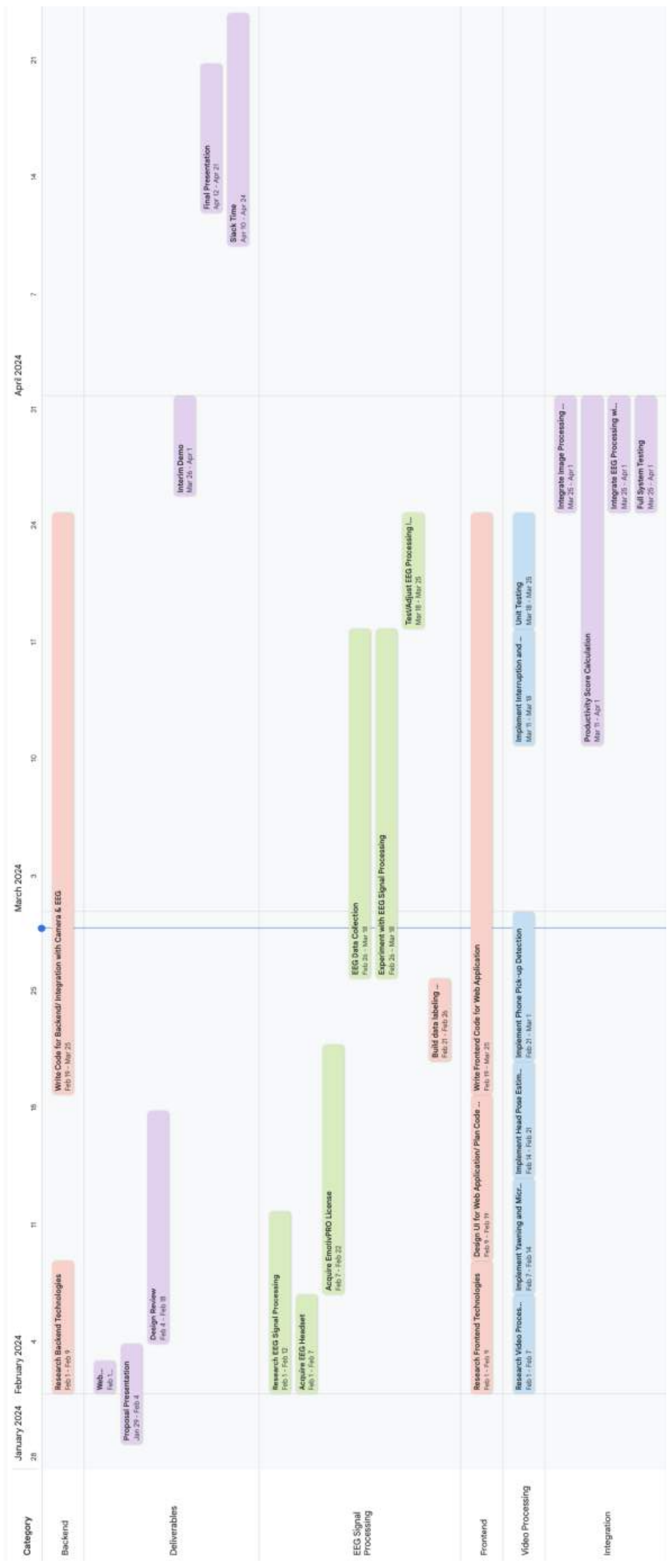


Figure 4: Gantt Chart