Neuro-Behavioral Stress Prediction Via Daily Routine and Online Activity

Nishan[1], Athmaranjan K[2], Bangera Ujwal Ganesh[3], Rohit[4], Shriharsha[5]
[1,3,4,5]*Student, Department of Information Science and Engineering, Srinivas Institute of Technology, Mangalore, Karnataka, India*
[2]*Associate Professor, Department of Information Science and Engineering, Srinivas Institute of Technology, Mangalore, Karnataka, India*
doi.org/10.64643/IJIRTV12I12-206833-459

***Abstract*—The traditional approaches for detecting stress mainly focus on using expensive clinical testing or intrusive devices in the healthcare sector. Stress usually goes undetected until it manifests through significant psychological or physiological symptoms. An automatic approach is developed in this study to detect stress based on behavioral patterns in digital data. The digital behavioral pattern is generated by compiling multiple behavioral factors, including the screen time, application usage, and emotions displayed in social networking. The combined behavioral pattern is assessed against a predefined baseline pattern using Machine Learning algorithms, which include Logistic Regression for text analysis and Random Forest for numerical features. The predictive model can provide timely alerts before burnout occurs.**

***Index Terms*—Behavioral Analysis, Machine Learning, Screen Time, Sentiment Analysis, Stress Prediction.**

I. INTRODUCTION

In today's world, proper psychological stress management is crucial to ensure sound mental and physical well-being [1]. Unchecked, chronic stress [2] acts as an acknowledged trigger for increased anxiety, work-related burnout, and significant drops in productivity. However, when living in an increasingly digital age, recognizing early behavioral changes that signal a potential psychological breakdown proves extremely challenging. Self-monitoring involves great awareness on behalf of the individual, whereas professional treatment tends to be reactive and occurs after reaching the peak of the disorder.

This paper aims at bridging this gap through the introduction of a methodology for automated behavioral analytics based on digitization [3]. As opposed to traditional approaches which involve lengthy and cumbersome questionnaires and complex wearable devices, this methodology uses digital data that are created by users on an ongoing basis, including screen times, application usage patterns, and social media messages. Through the integration of these variables into an overall machine learning model [4], the methodology allows for the continual evaluation of users' behaviors compared to pre-existing stress models. In the event that any anomalies pointing to high levels of stress are detected, a notification is automatically generated, prompting the user to rest.

II. LITERATURE REVIEW

2.1. Improving Students' Daily Life Stress Forecasting using LSTM Neural Networks

In this research, we consider several methods for predicting the degree of stress experienced on a daily basis by college students to help people modify their behavior in order to achieve good health in the future. For implementing the method, we used multimodal data from 142 users, including physiological data, mobile phone usage, geographic information, and behavior questionnaires. The data collection was carried out over N days, and they were then fed into time-series models to predict the binary value of stress on the next day. We used Long Short-Term Memory (LSTM) methods and compared them to Logistic Regression and Support Vector Machine models. From experiments, it can be seen that LSTM models clearly surpass other models, reaching accuracy of 83.6% using seven previous days. It is demonstrated that highly accurate predictions of stress can be made using only objective data without the need for daily behavior surveys

2.2. Screen Time Usage and its Relations with Social Support, Perceived Distress and Psychological Well-Being

This study examines the direct relationship between screen time usage and its impact on the mental health of Indian youth, assessing the pattern of screen time use and its correlation with perceived distress, social support, and psychological well-being. Data was collected from 201 participants using an 18-item Screen Time Questionnaire and the Perceived Stress Scale. The research discovered that people used smartphones more than any other electronic device. The research discovered that people who spent excessive time on screens experienced higher distress levels. The research obtained from screen time logs provides valid data which shows psychological distress patterns thus proving their use in the NBSP system.

2.3. Smartphone App Usage Analysis: Datasets, Methods, and Applications

The article offers an extensive exploration of the various methods used to investigate smartphone app usage data. The usage patterns formed by smartphone users based on their daily activities provide insights into how various types of users engage with distinct categories of apps. This survey outlines the methods involved in collecting data and provides insight into the various areas of analysis.

User Profiling is the main emphasis of this research paper, where user data is acquired from the app usage patterns and used to make inferences about user demographics, such as age and gender, and psychological state, involving stress levels, happiness, and emotions. It has been demonstrated that app usage patterns have been used to make inferences about people's psychological state, making them an essential component of NBSP behavioral modules.

2.4. Sentiment Analysis Methods, Applications, and Challenges

In this study, a complete literature review is provided for Sentiment Analysis (SA) [9], which is a technique used for the automated detection of opinion and emotion in large volumes of text. Techniques for this can be broadly divided into two types: dictionary techniques, which make use of dictionaries of positive and negative words, and machine learning techniques. Machine learning techniques include the use of algorithms like Naive Bayes, SVMs, and Deep Learning techniques that identify the sentiment of the text as being positive, negative, or neutral. The NBSP sentiment analysis algorithm uses its approach based on logistic regression along with TF-IDF vectorization [10].

III. SYSTEM DESIGN AND METHODOLOGY

The process of system design establishes all system elements through architectural design and component selection together with module definition and interface creation and data specification to meet established requirements. The complete architectural design of the NBSP system appears in this chapter through standard UML diagrams that include use case diagrams and sequence diagrams and data flow diagrams and activity diagrams.

3.1. Use Case Diagram for Stress Prediction System

A Use Case Diagram captures the actors who interact with the system. The diagram consists of three elements: the System (depicted as a bounding rectangle), Actors (shown as stick figures — in this case, the User and the System/Administrator), and Use Cases (solid-bordered ovals).

The User uses credentials to register and log into the application according to Figure 1. The User permits the application to access Screen Time logs and social media text feeds after logging in. The System (backend administrator) collects this data, pre-processes it, and feeds it into Machine Learning models. The System creates a Stress Score which it presents on the dashboard. The System uses a high stress level situation to activate an automatic 'Take a Break' notification system.

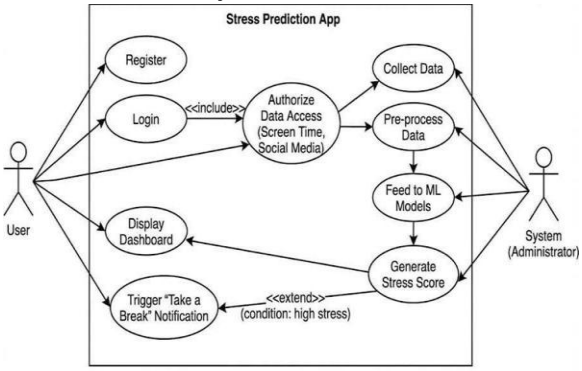

Figure 1. Use case diagram for user and system

3.2 Sequence Diagram for User and Prediction Engine
The Sequence Diagram in UML demonstrates how processes interact with each other following a specific sequence of operations. The system displays its processes through vertical lifelines which run parallel to each other while horizontal arrows display their message exchanges in the order of actual events.

The complete sequence of the NBSP prediction engine operation consists of the following steps:

1. User registers and logs into the mobile application.
2. User inputs or synchronizes data (screen time logs and social media posts) — this request is sent to the Backend Server.
3. Backend Server forwards text data to the Sentiment Analysis Module (Logistic Regression) and numeric data to the Behavioral Module (Random Forest).
4. Sentiment Module vectorizes the text via TF-IDF and predicts a probability score, returned to the Server.
5. Behavioral Module scales screen time features and predicts a probability score, returned to the Server.
6. Server aggregates both scores to calculate the final Stress Level.
7. Server stores the result in the Database and sends the final status (Stress/No Stress) back to the User interface.
8. If the result is 'Stress,' the User receives an alert notification.

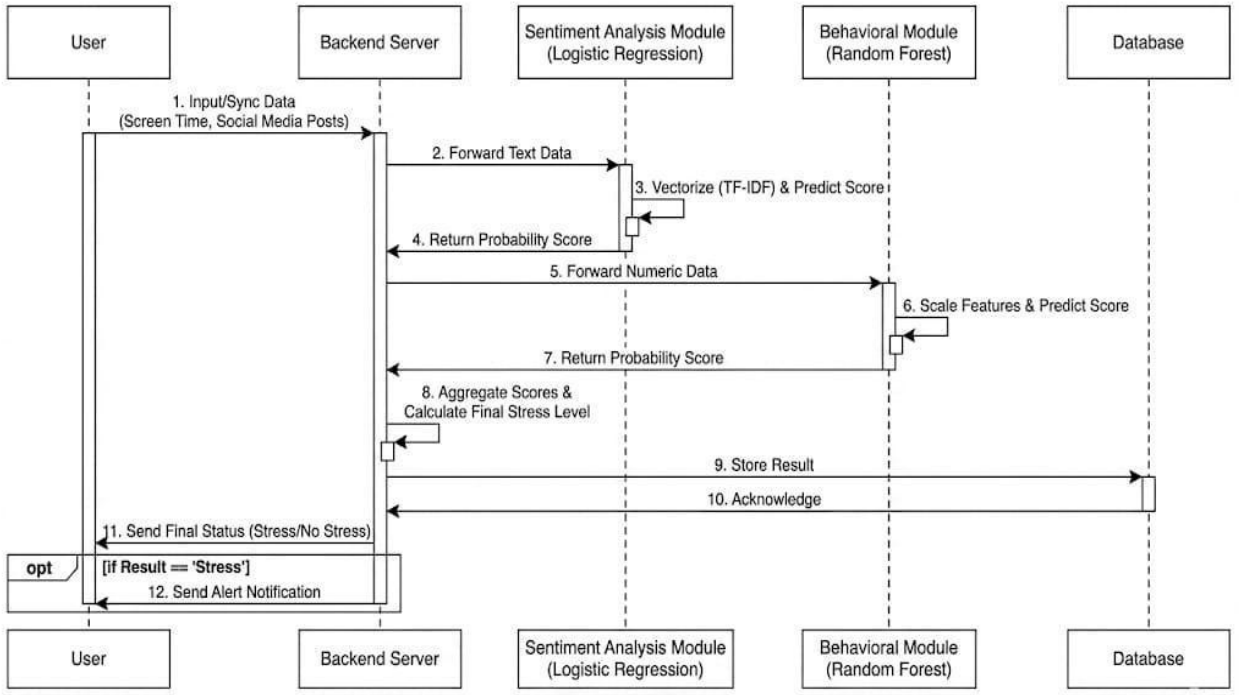

Figure 2: Sequence Diagram for User and Prediction Engine

3.3 Data Flow Diagram
A Data Flow Diagram (DFD) serves as a visual model that shows how information moves through an information system. DFDs use a restricted set of shapes which include Ovals for start and end points Arrows for data movement Parallelograms for input and output operations Rectangles for processing tasks and Diamonds for making decisions.

User Input marks the starting point of the flow which shows in Figure 3. Data Pre-processing [11] receives Raw Screen Time logs together with Text posts. The system divides the pre-processed data into two separate paths which send one stream to the Sentiment Analysis Model (Text) and the other stream to the Behavioral Analysis Model (Numeric) [12].

The Aggregator & Prediction Logic combines their scores to create a Final Stress Score. The decision node leads to two different outcomes which are 'Generate Alert' for high stress situations and 'Display Normal Status' for all other cases.

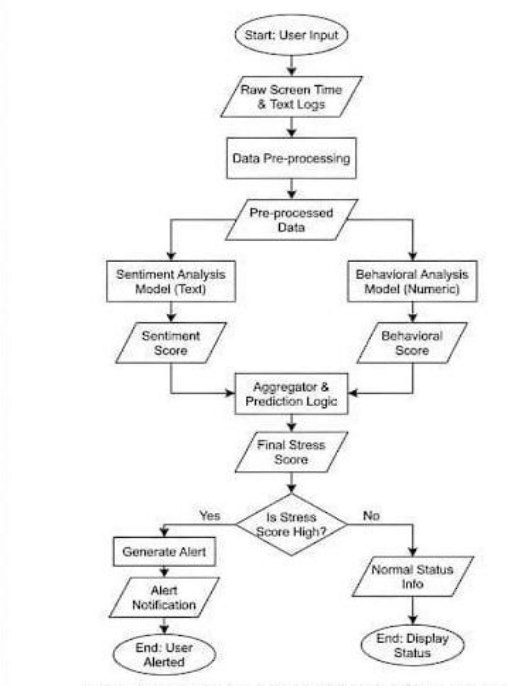

Figure 3. Data flow diagram for stress prediction system

3.4 Activity Diagram for Data Processing and Prediction

An Activity Diagram visually presents a series of actions or flow of control in a system, similar to a flowchart.

The prediction logic activity flow shows its flow through Figure 4:

- The activity starts with the Initial State, which receives raw data inputs.
- Pre-processing activity cleans text and scales numerical values.
- Flow splits into two concurrent activities: 'Predict Sentiment' and 'Predict Routine Behavior.'
- Flow merges at the 'Calculate Final Score' activity.
- A decision state: 'Is Score > Threshold?' — if True, 'Generate Alert'; if False, 'Log Normal Status.'
- Activity terminates at the Final State.

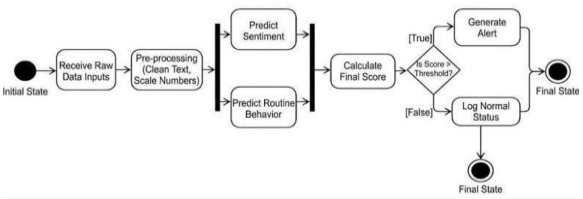

Figure 4. Activity diagram for Data Processing & Prediction

3.5 Activity Diagram for User Alert System

Figure 5 is a detailed description of the user alert system activity diagram, explaining under what conditions it needs to proceed upon the receiving of a prediction result:

The activity starts when the App receives a prediction result:

- If 'High Stress' is detected, the system fetches Recommendation Strategies (e.g., Take a break).
- The system displays the alert on the User's screen.
- The User acknowledges the alert by clicking 'OK.'
- The activity terminates.

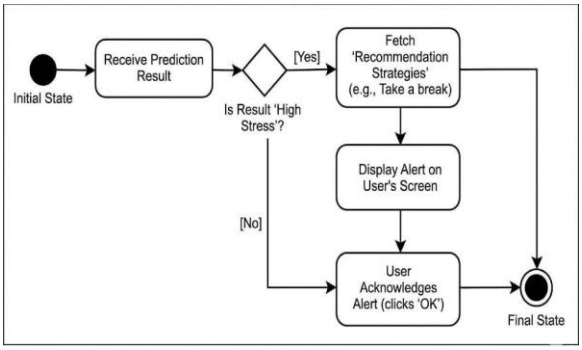

Figure 5. Activity diagram for user alert system

IV. IMPLEMENTATION DETAILS

System implementation is the stage where the theoretical design is converted into a working system. The NBSP system uses Python 3.9 together with its scientific computing stack to build a system that operates through a Flask backend and delivers a web-based frontend prototype. The project team selected each tool and library because it would help them achieve their project objectives.

A. Data Preprocessing and Feature Engineering

The Pandas and NumPy libraries were used to process dataset manipulations. The behavioral data underwent two processing steps which included normalizing daily application usage minutes and encoding categorical variables that distinguished between weekdays and weekends.

The training data received class imbalance correction through the implementation of Synthetic Minority Over-sampling Technique (SMOTE) [13]. The process of text sanitization [14] involved converting all characters to lowercase while stop words were

eliminated before the text was converted into numerical vectors through a TF-IDF (Term Frequency-Inverse Document Frequency) Vectorizer.

B. Machine Learning Models

The Scikit-learn library provided the tools necessary for training machine learning models. The Random Forest Classifier with 55 estimators was used to study the screen-time behavioral features.

The natural language processing system used a Logistic Regression model which was developed through training on TF-IDF vectorized social media text arrays. During real-time operation, both models yield independent probability matrices which are averaged to yield the ultimate prediction.

4.1 Algorithms for the Stress Prediction System

4.1.1. Algorithm for Model Training (Offline Process)

The model training algorithm encompasses two parallel workflows — one for the Screen Time behavioral model and one for the Sentiment model:

1. Start.
2. Load 'screen_time_data.csv' into a Pandas DataFrame.
3. Preprocess Screen Time Data: normalize numbers, encode 'Day_Type' column.
4. Apply SMOTE (Synthetic Minority Over-sampling Technique) to balance the dataset.
5. Perform Train-Test Split on the screen time data.
6. Initialize Random Forest Classifier (n_estimators = 55) and train it on training data.
7. Save trained model as 'screen_time_model.pkl' and scaler as 'screen_scaler.pkl'.
8. Load 'social_media_data.csv' (Dreaddit dataset) into a Pandas DataFrame.
9. Preprocess Text Data: convert to lowercase, remove stop words.
10. Perform Train-Test Split on the sentiment data.
11. Initialize TfidfVectorizer, fit on training text, and transform it to feature vectors.
12. Initialize Logistic Regression Classifier and train on TF-IDF features.
13. Save trained model as 'sentiment_model.pkl' and vectorizer as 'text_vectorizer.pkl'.
14. End.

4.1.2. Algorithm for Real-Time Stress Prediction (Backend API)

1. Start Server.
2. Load all four model artifacts into memory: screen_time_model.pkl, screen_scaler.pkl, sentiment_model.pkl, text_vectorizer.pkl.
3. Wait for user data: WHATSAPP_MIN, INSTA_MIN, YOUTUBE_MIN, DAY_TYPE, SOCIAL_POST.
4. Convert DAY_TYPE to numeric: Weekday = 0, Weekend = 1.
5. Create feature array: [WHATSAPP_MIN, INSTA_MIN, YOUTUBE_MIN, DAY_TYPE_NUM].
6. Transform feature array using loaded screen_scaler.pkl.
7. Predict PROB_1 = screen_time_model.predict_proba(scaled_features) — probability of 'Stress'.
8. Vectorize SOCIAL_POST using loaded text_vectorizer.pkl.
9. Predict PROB_2 = sentiment_model.predict_proba(text_features) — probability of 'Stress'.
10. Compute FINAL_STRESS_SCORE = (PROB_1 + PROB_2) / 2.
11. Apply thresholds: FINAL_STRESS_SCORE > 0.7 → 'High Stress'; > 0.4 → 'Medium Stress'; else → 'Low Stress'.
12. Send RESULT back to the user's app.
13. End.

V. RESULTS AND DISCUSSIONS

This chapter presents the performance metrics and results of the trained machine learning models, along with a comprehensive discussion of the NBSP system's architecture for stress prediction and user alerts. The results validate that the proposed multi-modal approach achieves practical accuracy for a non-invasive stress detection system.

A. Model Performance

The sentiment classifying component powered by Logistic Regression resulted in an overall accuracy of 74.65%.

Table 1. Classification Report Stress Sentiment Analysis Model

Class	Precision	Recall	F1-Score	Support
0	0.72	0.75	0.74	280
1	0.77	0.74	0.75	288
Accuracy	-	-	0.75	568
Macro Avg	0.75	0.75	0.75	568

The precision metric for the "Stress" class reached 0.77, because it correctly identifies stress cases with high accuracy while showing low false alarm rates that could induce panic. The balanced F1-scores across both the stressed (0.75) and non-stressed (0.74) categories demonstrate that the model avoids bias toward the majority class.

B. Alert Mechanics and User Interface

The raw probability scores produced through the aggregate models are translated into different actionable categories.

Table 2. Stress Level Thresholds and Alert Actions

Combined Stress Score Range :P_(Final)	Stress Level	Stress Level
0.00 to 0.33	Low	No action required.
0.34 to 0.66	Medium	Monitor your usage. Consider a short break.
0.67 to 1.00	High	Take a break, you seem to be too stressed.

The model uses a range of values starting at 0.00 up to 0.33, which suggests a low-stress level that does not need any action to be taken. The monitoring process starts when the user reaches the medium level of stress, which ranges from 0.34 to 0.66. An alarm goes off as soon as the user crosses the mark of 0.67 [15].

VI. DEPLOYMENT OVERVIEW (PROTOTYPE)

- The primary interface for the Frontend App Prototype, named “MindMetrics,” permits users to provide a text string and screen time information, which generates instant results in terms of stress levels along with relevant alert messages.
- Backend (Server): On startup, the server opens up the .pkl file. The text provided in each request is first processed using the TF-IDF Vectorizer and then evaluated through the Logistic Regression model.

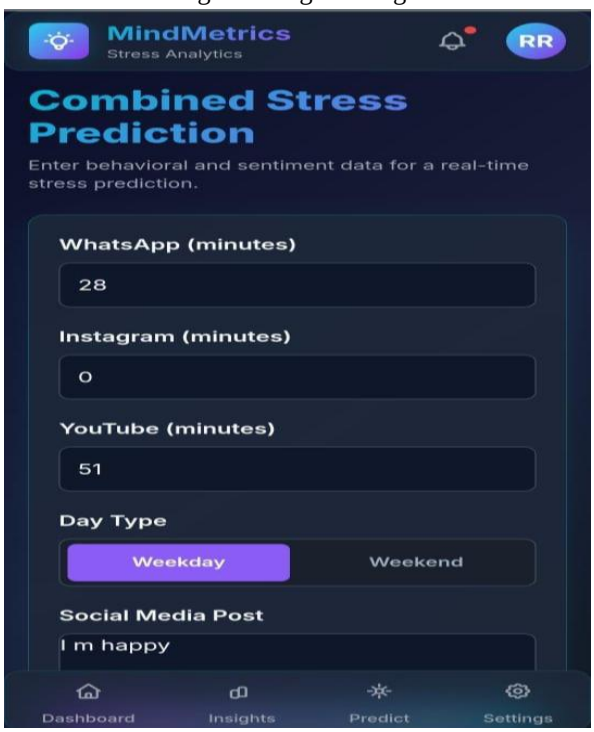

Figure 6: User Input for Stress Prediction

MindMetrics uses the daily behavior pattern and sentiment of the users to predict their stress level. The model keeps track of the behavior of the users in terms of their usage of WhatsApp and YouTube along with their content on social media.

The system utilizes the integrated data for real-time analysis, predicting user stress levels through visualization of the interaction between their emotional expressions and behavior to generate accurate predictions. The module shows functionality of the application by allowing users to enter data, thereby generating personalized results. The prediction becomes more accurate owing to use of several sources of input data, allowing the system to

learn the specific patterns of usage by users. The system establishes a base for progress through its current functions which enable automatic data gathering and advanced AI prediction systems.

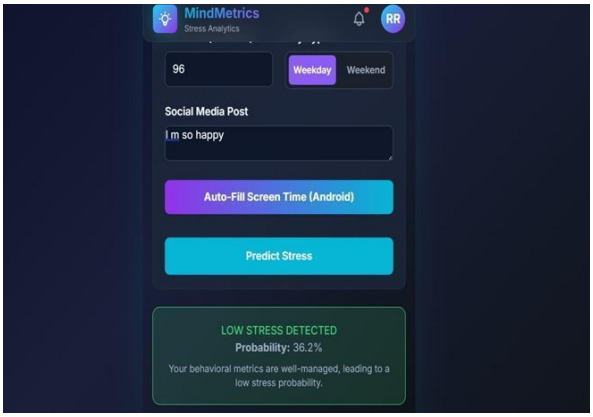

Figure 7: Stress Result

The working prototype demonstration uses a screenshot which is displayed in Figure 7. In the example, a user enters 96 minutes of screen time and the social media post 'I am so happy.'The system output shows LOW STRESS DETECTED with a probability of 36.2% which means Your behavioral metrics are well-managed, leading to a low stress probability.

Figure 8: User Dashboard

The MindMetrics dashboard provides a quick snapshot of the user's well-being. The user shows moderate stress with a 12% improvement from yesterday complex screen time usage of 6.5 hours and his neutral mood status and activity score of 78 out of 100. The weekly graph displays screen time patterns which assist users in comprehending their daily habits to maintain effective management.

VII. CONCLUSION AND FUTURE SCOPE

In this study, a system is built that can detect stress without the use of invasive methods. The proposed approach uses the regular metrics of the digital footprint, such as the usage of screens and social media, to detect stress without employing any expensive wearables or long clinical evaluations. The model uses state-of-the-art machine learning models to provide customized warnings that could help users recognize the signs of burnout before their occurrence. The upcoming versions of this platform will research Deep Learning systems [16] that use LSTM networks to achieve better results in understanding extended time patterns of human behavior. The system will achieve better data protection measures through its shift to Federated Learning [17] since all data assessment will take place on the user's mobile device. The system would benefit from using smartwatch biometric data such as heart rate variability to create a complete digital health system that improves its precision.

VII. FUTURE SCOPE

Future development of NBSP focuses on enhancing prediction accuracy, real-time capability, and user privacy:
The main objective of upcoming NBSP development work is to improve three specific areas which are prediction accuracy and real-time operational capacity and user privacy protection.

- Deep Learning Integration:

Our current system requires an upgrade to LSTM models which will improve our capability to analyze extended behavioral patterns and intricate textual structures.

- Real-Time Data Fetching:

Enhance the app fetches data through respective Android/iOS APIs instead of manually entering it.

- Integration of Wearable Devices:

The integration of wearable devices to track physiological parameters like heart rate and sleep is recommended to obtain a full health status.

- Data Processing for Privacy Protection:

Any process where all data are analyzed in the user's device can be considered as privacy-protective, implying that Federated Learning is better than Privacy since it guarantees that all data will be stored only on the user's device and, therefore, maintains privacy.

- Networking with Professionals:

A set of strategies to integrate the platform so that people with high stress levels can directly interact with mental health professionals has been developed.

REFERENCES

[1] B. Wang, T. Katsube, N. Begum, and M. Nenoi, "Revisiting the health effects of psychological stress—Its influence on susceptibility to ionizing radiation: A mini-review," *J. Radiat. Res.*, vol. 57, no. 4, pp. 325–335, Jul. 2016, doi: 10.1093/jrr/rrw035.

[2] Y. Fukuoka and A. Ishida, "Chronic stress evaluation using neural networks," IEEE Eng. Med. Biol. Mag., vol. 19, no. 1, pp. 34–38, Jan.–Feb. 2000, doi: 10.1109/51.816242.

[3] S. Karabatak and M. Karabatak, "Z Generation Students and Their Digital Footprints," in Proc. 8th Int. Symp. Digital Forensics Security (ISDFS), Beirut, Lebanon, 2020, pp. 1–5, doi: 10.1109/ISDFS49300.2020.9116455.

[4] K. Sharma, V. M., K. K., and S. G., "UML-Based Modeling of Quantum Machine Learning Systems: A Software Engineering Approach," in Proc. 4th Int. Conf. Innovative Mechanisms for Industry Applications (ICIMIA), Tirupur, India, 2025, pp. 1915–1919, doi: 10.1109/ICIMIA67127.2025.11200580.

[5] H. Kurniawan and M. Pechenizkiy, "Towards the Stress Analytics Framework: Managing, Mining, and Visualizing Multi-Modal Data for Stress Awareness," in Proc. 27th IEEE Int. Symp. Computer-Based Medical Systems (CBMS), New York, NY, USA, 2014, pp. 541–542, doi: 10.1109/CBMS.2014.129.

[6] J. Lu, Q. Zhang, Z. Yang, and M. Tu, "A Hybrid Model Based on Convolutional Neural Network and Long Short-Term Memory for Short-Term Load Forecasting," in Proc. IEEE Power Energy Soc. General Meeting (PESGM), Atlanta, GA, USA, 2019, pp. 1–5, doi: 10.1109/PESGM40551.2019.8973549.

[7] W. Chen, X. Liu, and B. Litt, "Logistic-Weighted Regression Improves Decoding of Finger Flexion from Electrocorticographic Signals," in Proc. 36th Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. (EMBC), Chicago, IL, USA, 2014, pp. 2629–2632, doi: 10.1109/EMBC.2014.6944162.

[8] Z. Hao, L. Shaohong, and S. Jinping, "Unit Model of Binary SVM with DS Output and Its Application in Multi-Class SVM," in Proc. 4th Int. Symp. Computational Intelligence and Design (ISCID), Hangzhou, China, 2011, pp. 101–104, doi: 10.1109/ISCID.2011.34.

[9] Z. Lingli et al., "SA-Model: Multi-Feature Fusion Poetic Sentiment Analysis Based on a Hybrid Word Vector Model," in Proc. 5th Int. Conf. Pattern Recognition and Artificial Intelligence (PRAI), Chengdu, China, 2022, pp. 984–988, doi: 10.1109/PRAI55851.2022.9904158.

[10] A. Gupta and U. Sharma, "Machine Learning Based Sentiment Analysis of Hindi Data with TF-IDF and Count Vectorization," in Proc. 7th Int. Conf. Computing, Communication and Security (ICCCS), Seoul, Republic of Korea, 2022, pp. 1–5, doi: 10.1109/ICCCS55188.2022.10079323.

[11] A. Juneja and N. N. Das, "Big Data Quality Framework: Pre-Processing Data in Weather

Monitoring Application," in Proc. Int. Conf. Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon), Faridabad, India, 2019, pp. 559–563, doi: 10.1109/COMITCon.2019.8862267.

[12] C.-C. Kuo, M.-J. Lee, I.-C. Tsai, C.-N. J. Liu, and C.-J. Huang, "An Accurate PLL Behavioral Model for Fast Monte Carlo Analysis Under Process Variation," in Proc. IEEE Int. Behavioral Modeling and Simulation Workshop (BMAS), San Jose, CA, USA, 2007, pp. 110–114, doi: 10.1109/BMAS.2007.4437535.

[13] H. Lee, S. Jung, M. Kim, and S. Kim, "Synthetic Minority Over-Sampling Technique Based on Fuzzy C-Means Clustering for Imbalanced Data," in Proc. Int. Conf. Fuzzy Theory and Its Applications (iFUZZY), Pingtung, Taiwan, 2017, pp. 1–6, doi: 10.1109/iFUZZY.2017.8311793.

[14] S. Hedegaard, S. Houen, and J. G. Simonsen, "LAIR: A Language for Automated Semantics-Aware Text Sanitization Based on Frame Semantics," in Proc. IEEE Int. Conf. Semantic Computing (ICSC), Berkeley, CA, USA, 2009, pp. 47–52, doi: 10.1109/ICSC.2009.79.

[15] S. Seydnejad, "Fixed Threshold Modeling of an Adjustable Threshold Integrate-and-Fire Neuronal Model," in Proc. 30th Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. (EMBS), Vancouver, BC, Canada, 2008, pp. 2481–2484, doi: 10.1109/IEMBS.2008.4649703.

[16] K. B. Lee and H. S. Shin, "An Application of a Deep Learning Algorithm for Automatic Detection of Unexpected Accidents Under Bad CCTV Monitoring Conditions in Tunnels," in Proc. Int. Conf. Deep Learning and Machine Learning in Emerging Applications (Deep-ML), Istanbul, Turkey, 2019, pp. 7–11, doi: 10.1109/Deep-ML.2019.00010.

[17] IEEE Draft Guide for Architectural Framework and Application of Federated Machine Learning, IEEE Standard P3652.1/D6, pp. 1–70, Jun. 2020.