ДОДАТОК А

Звіт результатів перевірки на унікальність тексту в базі Хнуре

StrikePlagiarism.com

Дата звіту 6/17/2025

Дата редагування ---

Звіт не був оцінений

Звіт подібності

метадані

Назва організації
Kharkiv National University of Radio Electronics

Заголовок
2025_M_ПІ_ІП3_23_4_Кіданов_А_О_скорочений

Автор
Кіданов Артем Олександрович

Науковий керівник / Експерт
Олена Опійник

підрозділ
каф. ПІ

Обсяг знайдених подібностей

Коефіцієнт подібності визначає, який відсоток тексту по відношенню до загального обсягу тексту було знайдено в різних джерелах. Зверніть увагу, що високі значення коефіцієнта не автоматично означають плагіат. Звіт має аналізувати компетентна / уповноважена особа.

0.87% 0.87% КП 1

0.92% 0.92% КЦ

25
Довжина фрази для коефіцієнта подібності 2

8713
Кількість слів

70034
Кількість символів

Тривога

У цьому розділі ви знайдете інформацію щодо текстових спотворень. Ці спотворення в тексті можуть говорити про МОЖЛИВІ маніпуляції в тексті. Спотворення в тексті можуть мати навмисний характер, але частіше характер технічних помилок при конвертації документа та його збереженні, тому ми рекомендуємо вам підходити до аналізу цього модуля відповідально. У разі виникнення запитань, просимо звертатися до нашої служби підтримки.

Заміна букв		14
Інтервали		0
Мікропробіли		0
Білі знаки		0
Парафрази (SmartMarks)		6

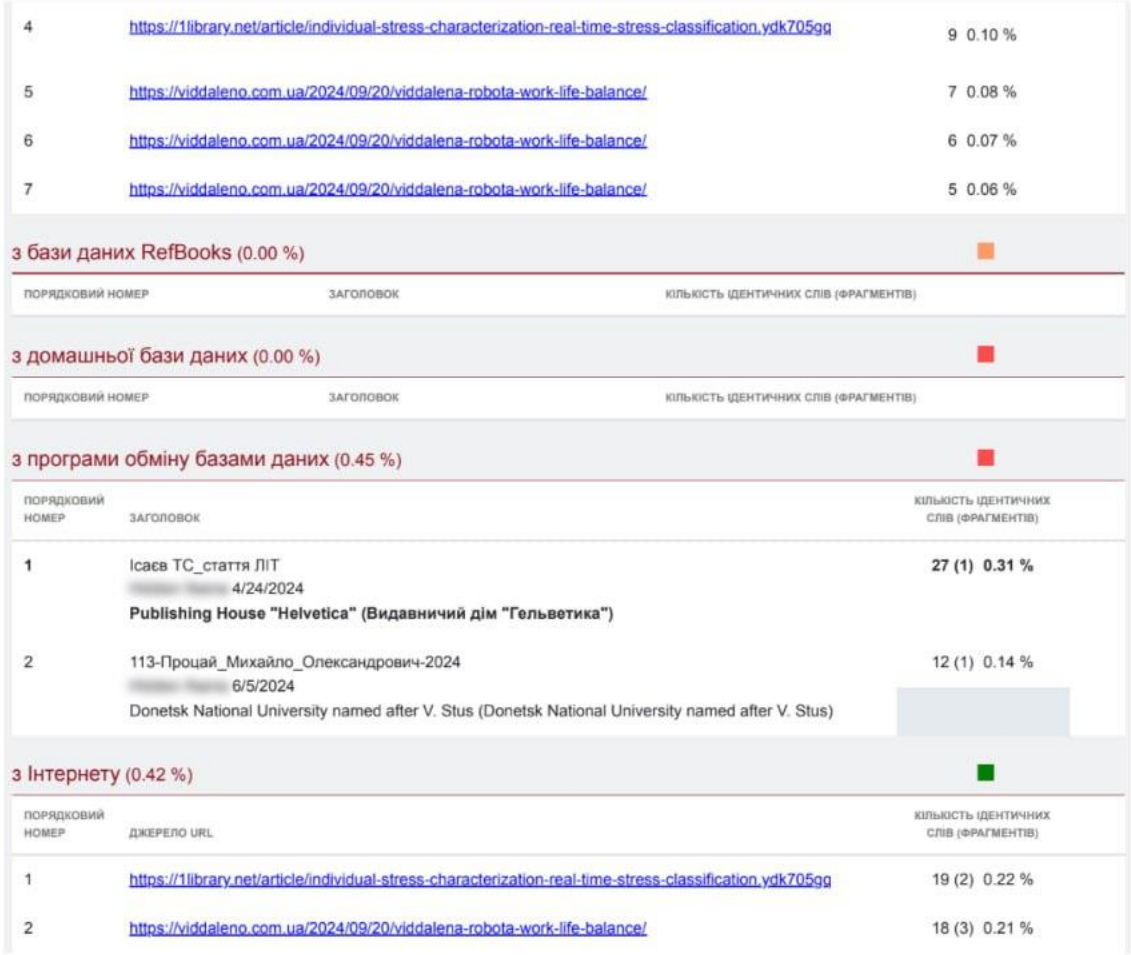
Подібності за списком джерел

Нижче наведений список джерел. В цьому списку є джерела із різних баз даних. Колір тексту означає в якому джерелі він був знайдений. Ці джерела і значення Коефіцієнту Подібності не відображають прямого плагіату. Необхідно відкрити кожне джерело і проаналізувати зміст і правильність оформлення джерела.

10 найдовших фраз — Колір тексту

ПОРЯДКОВИЙ НОМЕР	НАЗВА ТА АДРЕСА ДЖЕРЕЛА URL (НАЗВА БАЗИ)	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
1	Icaca TC_стаття ЛІТ [redacted] 4/24/2024 Publishing House "Helvetica" (Видавничий дім "Тельветика")	27 0.31 %
2	113-Процай_Михайло_Олександрович-2024 [redacted] 6/5/2024 Donetsk National University named after V. Stus (Donetsk National University named after V. Stus)	12 0.14 %
3	https://1library.net/article/individual-stress-characterization-real-time-stress-classification.ydk705qq	10 0.11 %

Рисунок А.1 – Перша сторінка звіту на унікальність тексту



4	https://1library.net/article/individual-stress-characterization-real-time-stress-classification.ydk705gq	9 0.10 %
5	https://viddaleno.com.ua/2024/09/20/viddalena-robota-work-life-balance/	7 0.08 %
6	https://viddaleno.com.ua/2024/09/20/viddalena-robota-work-life-balance/	6 0.07 %
7	https://viddaleno.com.ua/2024/09/20/viddalena-robota-work-life-balance/	5 0.06 %

з бази даних RefBooks (0.00 %)

ПОРЯДКОВИЙ НОМЕР	ЗАГОЛОВОК	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)

з домашньої бази даних (0.00 %)

ПОРЯДКОВИЙ НОМЕР	ЗАГОЛОВОК	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)

з програми обміну базами даних (0.45 %)

ПОРЯДКОВИЙ НОМЕР	ЗАГОЛОВОК	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
1	Ісаєв ТС_стаття ЛІТ [redacted] 4/24/2024 **Publishing House "Helvetica" (Видавничий дім "Гельветика")**	**27 (1) 0.31 %**
2	113-Процай_Михайло_Олександрович-2024 [redacted] 6/5/2024 Donetsk National University named after V. Stus (Donetsk National University named after V. Stus)	12 (1) 0.14 %

з Інтернету (0.42 %)

ПОРЯДКОВИЙ НОМЕР	ДЖЕРЕЛО URL	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
1	https://1library.net/article/individual-stress-characterization-real-time-stress-classification.ydk705gq	19 (2) 0.22 %
2	https://viddaleno.com.ua/2024/09/20/viddalena-robota-work-life-balance/	18 (3) 0.21 %

Список прийнятих фрагментів (немає прийнятих фрагментів)

ПОРЯДКОВИЙ НОМЕР	ЗМІСТ	КІЛЬКІСТЬ ОДНАКОВИХ СЛІВ (ФРАГМЕНТІВ)

ВСТУП

Дистанційна робота стала невід'ємною частиною сучасного ринку праці, значно трансформуючи підходи до організації трудових процесів. Пандемія COVID-19, яка змусила мільйони працівників перейти на віддалений формат, стала каталізатором цифрової революції у бізнесі. Цей формат надає переваги як компаніям (зниження витрат, підвищення гнучкості), так і працівникам (покращення балансу між роботою та особистим життям). Однак водночас він створює низку викликів: зростання випадків вигорання, стресу та ізоляції, що суттєво впливає на продуктивність і благополуччя працівників.

Сучасні технології, такі як аналіз текстових даних, комп'ютерне бачення та фізіологічний аналіз, пропонують нові можливості для моніторингу емоційного стану. Водночас вони стикаються з такими перешкодами, як високі витрати, етичні питання та складнощі інтеграції.

Метою цього дослідження є аналіз існуючих методів оцінки емоційного стану працівників та розробка інтегрованої системи, яка поєднує найефективніші з них для надання рекомендацій із покращення емоційного стану. Особлива увага приділяється пошуку рішень, які будуть доступними для різних типів компаній, зокрема тих, що використовують дистанційний або гібридний формат роботи.

Завданнями роботи є:

Рисунок А.2 – Друга сторінка звіту на унікальність тексту

ДОДАТОК Б

Слайди презентації

Дослідження методів аналізу дистанційної роботи працівників. Рекомендації для покращення емоційного стану

ст. гр. ІПЗм-23-4 Кіданов А. О.
науковий керівник: доц. Каф. ПІ Лещинський В. О.

1

ДОСЛІДЖЕННЯ

- Масовий перехід на дистанційну роботу під час пандемії спричинив зростання рівня стресу та частоти емоційного вигорання серед працівників.
- Метою дослідження стало створення інноваційного IT-рішення для раннього виявлення ознак стресу, вигорання та депресивних станів.
- Запропонована система має інтегрувати кілька каналів аналізу, оперативно виявляти ознаки депресії та надавати рекомендації користувачам.

2

АКТУАЛЬНІСТЬ ДОСЛІДЖЕННЯ

- Емоційне вигорання та стрес стали одними з найпоширеніших проблем працівників в умовах дистанційної та гібридної роботи.
- Більшість існуючих рішень є або надто поверхневими (опитування), або дорогими та інвазивними (моніторинг активності, відеоспостереження).
- Зростає потреба у комплексному, доступному й етично обґрунтованому підході до оцінки емоційного стану персоналу.
- Моя робота спрямована на розробку такої системи з використанням сучасних технологій багатоканального аналізу.

ПОСТАНОВКА ЗАДАЧІ

Мета: створити інтегровану систему багатоканального моніторингу та аналізу емоційного стану.

Завдання на дослідження:

- Проаналізувати існуючі методи
- Порівняти їх ефективність
- Вибрати 2–3 найкращих методи
- Розробити прототип системи
- Протестувати систему

ПЛАН ДОСЛІДЖЕННЯ

1. Аналіз літературних джерел
2. Визначення та обгрунтування вибору методів
3. Визначення критеріїв для оцінки методів
4. Збір та нормалізація критеріїв
5. Розрахунок ефективності методів
6. Планування архітектури системи
7. Розробка MVP системи
8. Оцінка результатів

МЕТОДИ АНАЛІЗУ

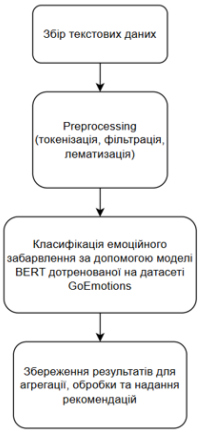
1. NLP-аналітика
2. Voice Emotion Recognition
3. Комп'ютерне бачення
4. Опитування
5. Behavioral Biometrics
6. Фізіологічний аналіз
7. Моніторинг активності

NLP-АНАЛІТИКА

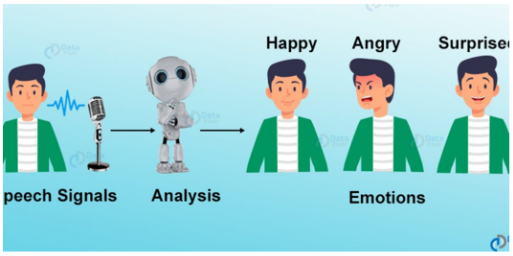
- Автоматичний аналіз текстових повідомлень (чати, звіти)
- Використання моделі GoEmotions (класифікація 27 емоцій)
- Висока точність – до 90%

Проблеми:

- залежність від контексту та похибки формулювання



РОЗПІЗНАВАННЯ ЕМОЦІЙ У ГОЛОСІ



- Аналіз тональності, тембру, пауз, швидкості мовлення
- Нейронні мережі в основі (наприклад, wav2vec)

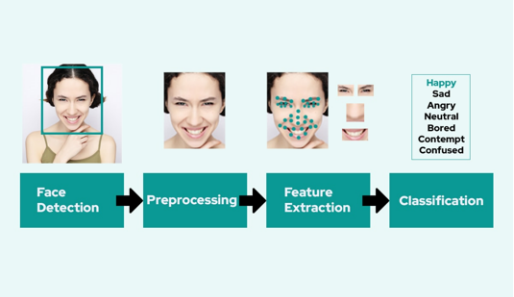
Проблеми:

- висока чутливість до шуму

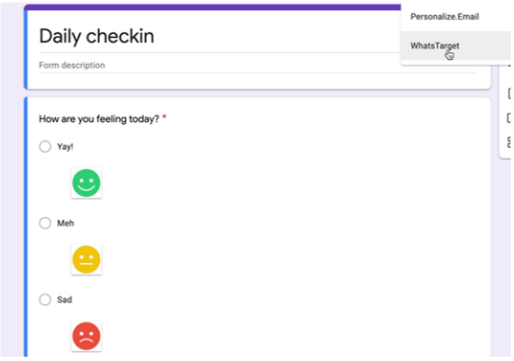
КОМП'ЮТЕРНЕ БАЧЕННЯ

- Аналіз міміки на відео / відеозв'язку
- Використання Mediapipe, CNN
- Виявлення емоцій у реальному часі

Проблеми:
- Якість відео
- Висока інвазивність



ОПИТУВАННЯ



- Анкети типу PANAS або власні форми
- Суб'єктивна оцінка самопочуття
- Можна персоналізувати рекомендації

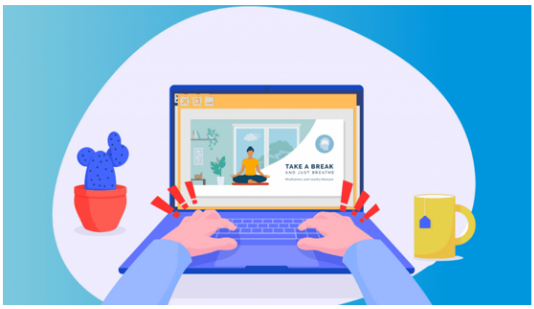
Проблеми:
- Чесність користувачів
- Людська похибка
- Поверхневість
- Суб'єктивність

ПОВЕДІНКОВІ БІОМЕТРИЧНІ ДАНІ

- Патерни натискань клавіш, рух миші
- Низька інвазивність, непомітний збір
- Можна виявити зміну настрою через зміну поведінки

Проблеми:

- Обмежена точність

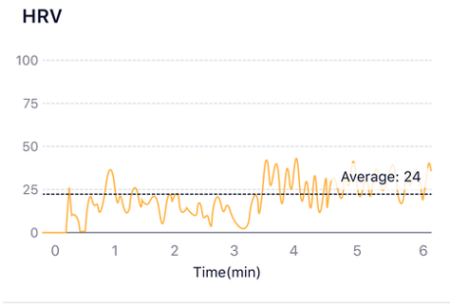


ФІЗІОЛОГІЧНИЙ АНАЛІЗ

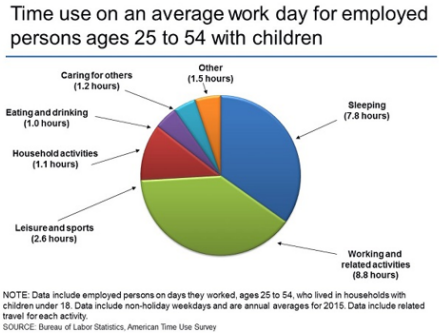
- Дані: ЧСС, дихання, мікрорухи
- Можливість зменшення ризику похибки через алгоритми
- Висока точність – до 92%

Проблеми:

- Інвазивність
- Додаткове обладнання



МОНІТРИНГ АКТИВНОСТІ



- Вимірювання продуктивності, тривалості перерв, зміни в динаміці роботи
- Використання: RescueTime, аналіз графіків активності
- Простий та автоматизований метод

Проблеми:

- Не дає розуміння саме до емоцій
- Необхідність знаходити додаткові відрношення до емоцій

КРИТЕРІЇ ОЦІНКИ

Таблиця 4.1 – Типи та оцінка шкал критеріїв (таблиця виконана самостійно)

Критерій	Тип шкали	Оцінка
Точність	Інтервальна	0 – 100 %
Економічна та технічна доцільність	Інтервальна	0 – 10 (оцінка)
Інвазивність	Номінальна	Висока, середня, низька
Масштабованість	Інтервальна	0 – 100 %
Надійність / Стійкість до шуму	Інтервальна	0 – 100 %
Оперативність виявлення змін	Інтервальна	N часу

ЗБІР ДАНИХ

Метод	Точність	Економічна та технічна доцільність	Інвазивність	Масштабованість	Надійність / Стійкість до шуму	Оперативність виявлення змін
NLP-аналітика	85	8	середня	90	85	0.5 хв
Комп'ютерне бачення	80	7	середня	90	70	1.2 хв
Фізіологічний аналіз	92	6	середня	80	90	10 хв
Behavioral Biometrics	70	8	середня	85	65	2 хв
Voice Emotion Recognition	80	8	середня	75	60	1 хв
Опитування	60	8	низька	95	60	2 хв
Системи моніторингу активності	65	9	низька	95	75	60 хв

15

НОРМАЛІЗАЦІЯ ДАНИХ

Метод	Точність	Економічна та технічна доцільність	Інвазивність	Масштабованість	Надійність / Стійкість до шуму	Оперативність виявлення змін
NLP-аналітика	0,78	0,67	0	0.75	0,83	1
Комп'ютерне бачення	0,63	0.33	0	0,75	0,33	0,98
Фізіологічний аналіз	1	0	0	0,25	1	0,84
Behavioral Biometrics	0.31	0,67	0	0,5	0,16	0,97
Voice Emotion Recognition	0,63	0,67	0	0	0	0,99
Опитування	0	0,67	1	1	0	0,97
Системи моніторингу активності	0,16	1	1	1	0,5	0

16

ОЦІНКА КОРИСНОСТІ

$$U = \sum_{i=1}^{n} w_i * X_{ij}$$

, де U – коефіцієнт ефективності,

w_i - ваговий коефіцієнт критерію i,

X_i - нормалізоване значення критерію і для методу j.

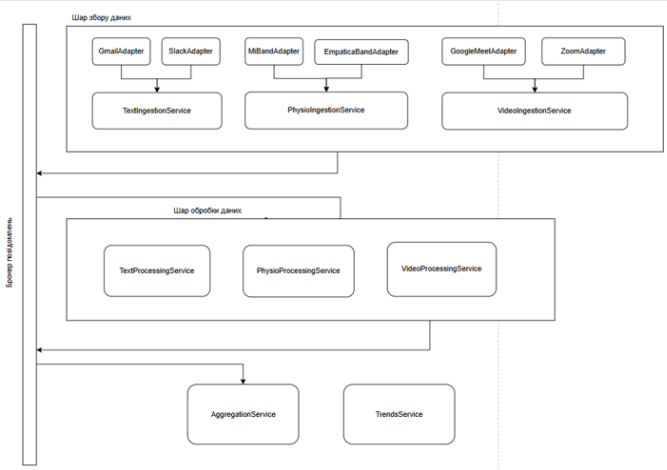
Таблиця 4.5 – Коефіцієнти ефективності методів (таблиця виконана самостійно)

Метод	Коефіцієнт корисності
NLP-аналітика	0.722
Фізіологічний аналіз	0.614
Комп'ютерне бачення	0.548
Системи моніторингу активності	0.522
Опитування	0.496
Behavioral Biometrics	0.440
Voice Emotion Recognition	0.436

РЕЗУЛЬТАТИ ТЕОРЕТИЧНОГО ДОСЛІДЖЕННЯ

- Проаналізовано 7 сучасних методів обробки та визначення емоційного стану людини
- Проведено кількісне порівняння методів за 6 критеріями
- Застосовано адитивну згортку і обрано 3 методи з найкращими показниками ефективності:
  - NLP-аналітика (0.722)
  - Фізіологічний аналіз (0.614)
  - Комп’ютерне бачення (0.548)
- Підтверджено доцільність багато-канального алгоритму визначення стресу, депресії та вигорання

АРХІТЕКТУРА СИСТЕМИ



ВИМОГИ ДО СИСТЕМИ

Функціональні:

- збір даних
- обробка даних
- аналіз даних
- агрегація результату аналізу
- аналіз емоційного стану
- генерація рекомендацій
- візуалізація даних
- аутентифікація та авторизація
- реальний час та інтерактивність

Нефункціональні:

- продуктивність і оперативність
- масштабованість
- безпека та конфіденційність
- надійність
- зручність використання
- підтримуваність

MVP-ВЕРСІЯ СИСТЕМИ

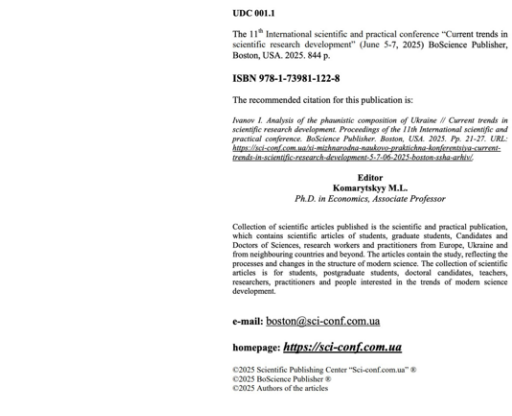
Допущення:

- Збір реальних даних замінено на генерацію
- Відсутня реалізація адаптерів
- Виключено мобільний додаток
- Спрощений інтерфейс

Реалізовано:

- Сервіси збору даних (з генерацією)
- Аналіз даних та виокремлення емоцій та інших показників
- Агрегація даних та обчислення трендів
- Визначення індексів стресу
- Web-додаток для відображення даних по користувачам
- Web-розширення для повідомлення користувачу

АПРОБАЦІЯ РЕЗУЛЬТАТІВ РОБОТИ



UDC 001.1

The 11th International scientific and practical conference "Current trends in scientific research development" (June 5-7, 2025) BoScience Publisher, Boston, USA. 2025. 844 p.

ISBN 978-1-73981-122-8

The recommended citation for this publication is:

Ivanov I. Analysis of the phaunistic composition of Ukraine // Current trends in scientific research development. Proceedings of the 11th International scientific and practical conference. BoScience Publisher. Boston, USA. 2025. Pp. 21-27. URL: https://sci-conf.com.ua/xi-mizhnarodna-naukovo-praktichna-konferentsiya-current-trends-in-scientific-research-development-5-7-06-2025-boston-usha-arhiv/

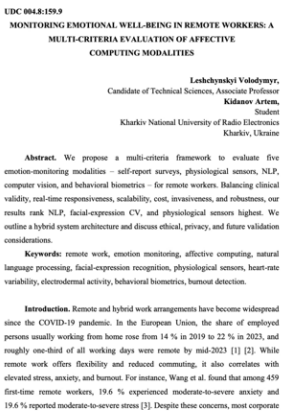
Editor
Komarytskyy M.L.
Ph.D. in Economics, Associate Professor

Collection of scientific articles published is the scientific and practical publication, which contains scientific articles of students, graduate students, Candidates and Doctors of Sciences, research workers and practitioners from Europe, Ukraine and from neighbouring countries and beyond. The articles contain the study, reflecting the processes and changes in the structure of modern science. The collection of scientific articles is for students, postgraduate students, doctoral candidates, teachers, researchers, practitioners and people interested in the trends of modern science development.

e-mail: boston@sci-conf.com.ua

homepage: ***https://sci-conf.com.ua***

3



UDC 004.8:159.9

MONITORING EMOTIONAL WELL-BEING IN REMOTE WORKERS: A MULTI-CRITERIA EVALUATION OF AFFECTIVE COMPUTING MODALITIES

Leshchynskyi Volodymyr,
Candidate of Technical Sciences, Associate Professor
Kidanov Artem,
Student
Kharkiv National University of Radio Electronics
Kharkiv, Ukraine

Abstract. We propose a multi-criteria framework to evaluate five emotion-monitoring modalities – self-report surveys, physiological sensors, NLP, computer vision, and behavioral biometrics – for remote workers. Balancing clinical validity, real-time responsiveness, scalability, cost, invasiveness, and robustness, our results rank NLP, facial-expression CV, and physiological sensors highest. We outline a hybrid system architecture and discuss ethical, privacy, and future validation considerations.

Keywords: remote work, emotion monitoring, affective computing, natural language processing, facial-expression recognition, physiological sensors, heart-rate variability, electrodermal activity, behavioral biometrics, burnout detection.

Introduction. Remote and hybrid work arrangements have become widespread since the COVID-19 pandemic. In the European Union, the share of employed persons usually working from home rose from 14 % in 2019 to 22 % in 2023, and roughly one-third of all working days were remote by mid-2023 [1] [2]. While remote work offers flexibility and reduced commuting, it also correlates with elevated stress, anxiety, and burnout. For instance, Wang et al. found that among 459 first-time remote workers, 19.6 % experienced moderate-to-severe anxiety and 19.6 % reported moderate-to-severe stress [3]. Despite these concerns, most corporate

224

SE software engineering | NURE | ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ

ВИСНОВКИ

- Проведено глибокий аналіз 7 методів оцінки емоційного стану працівників.
- Визначено ключові критерії ефективності: точність, інвазивність, масштабованість тощо.
- Розраховано коефіцієнти корисності для кожного методу з використанням адитивної згортки.
- Обрано 3 найефективніші методи: NLP, фізіологічний аналіз, комп'ютерне бачення.
- Розроблено архітектуру системи моніторингу емоційного стану з багатоканальним збором даних.
- Реалізовано MVP-прототип із тестовими даними та базовим функціоналом.
- Отримані результати підтверджують доцільність багатоканального підходу до моніторингу емоційного стану.

ДОДАТОК В

Діаграми архітектури програми

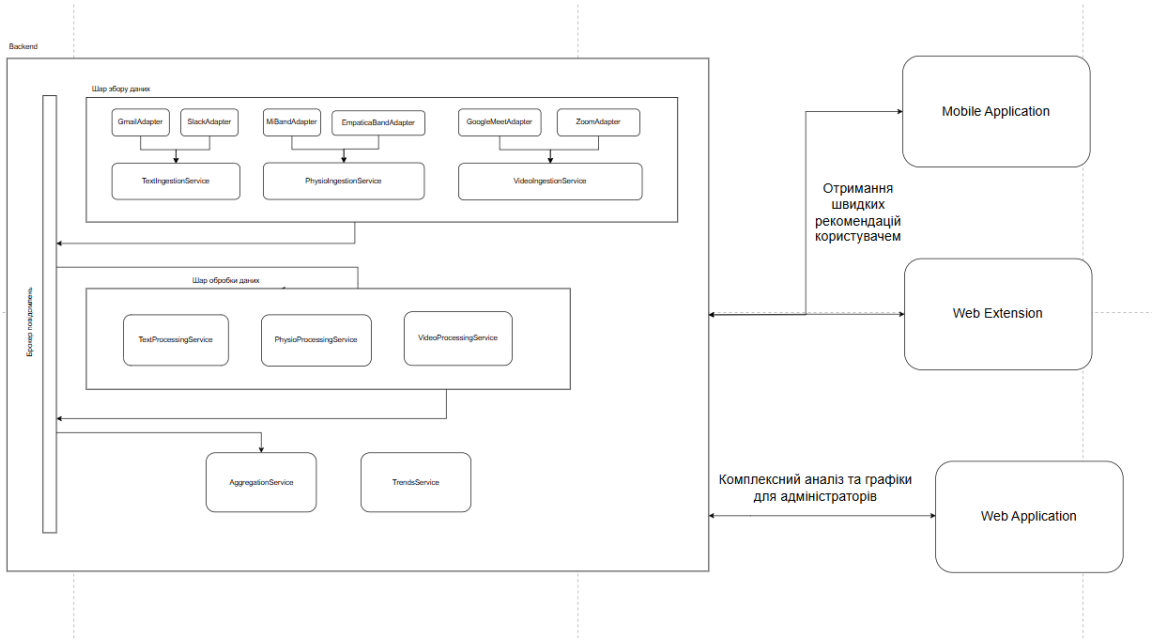


Рисунок В.1 – UML-діаграма компонентів (рисунок виконаний самостійно)

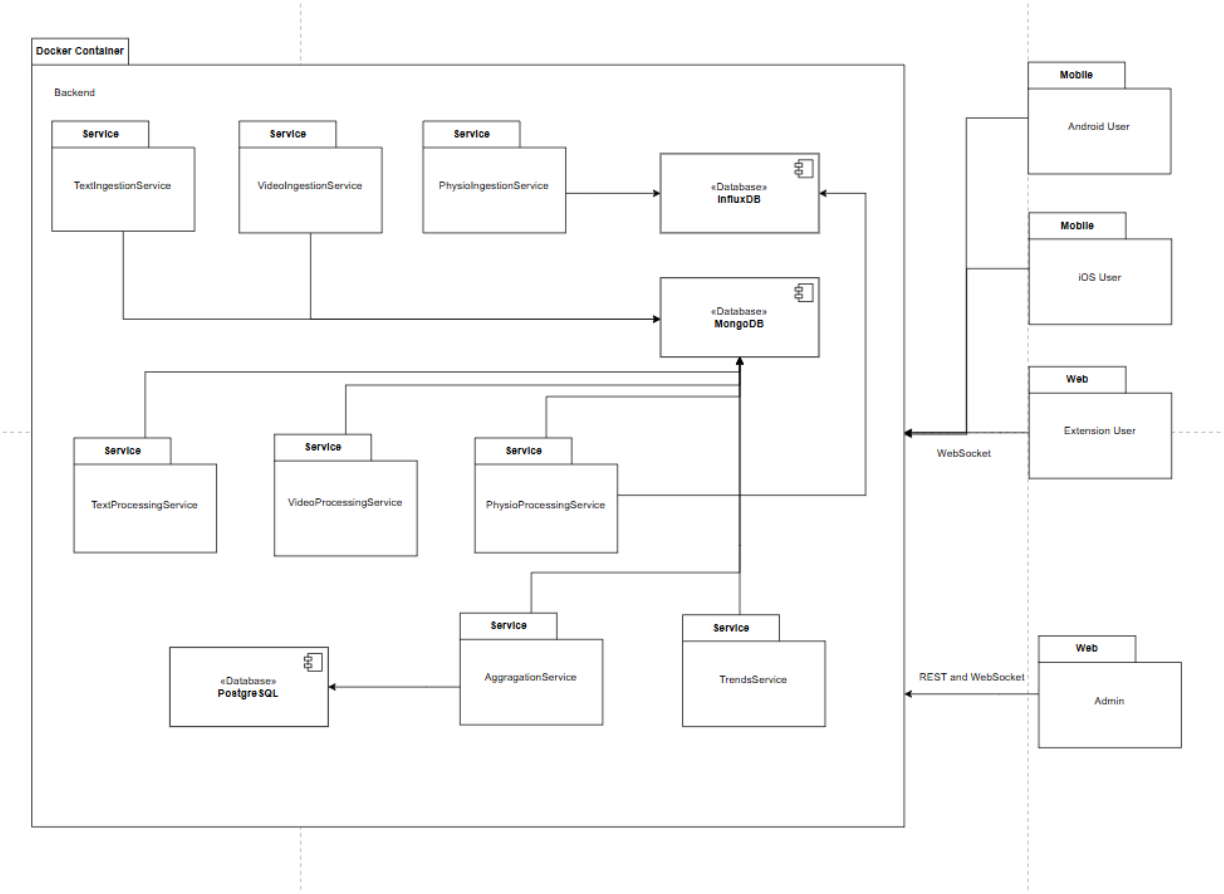


Рисунок В.2 – UML-діаграма розгортання (рисунок виконаний самостійно)

ДОДАТОК Г

Апробація результатів роботи

UDC 004.8:159.9

MONITORING EMOTIONAL WELL-BEING IN REMOTE WORKERS: A MULTI-CRITERIA EVALUATION OF AFFECTIVE COMPUTING MODALITIES

Leshchynskyi Volodymyr,
Candidate of Technical Sciences, Associate Professor
Kidanov Artem,
Student
Kharkiv National University of Radio Electronics
Kharkiv, Ukraine

Abstract. We propose a multi-criteria framework to evaluate five emotion-monitoring modalities – self-report surveys, physiological sensors, NLP, computer vision, and behavioral biometrics – for remote workers. Balancing clinical validity, real-time responsiveness, scalability, cost, invasiveness, and robustness, our results rank NLP, facial-expression CV, and physiological sensors highest. We outline a hybrid system architecture and discuss ethical, privacy, and future validation considerations.

Keywords: remote work, emotion monitoring, affective computing, natural language processing, facial-expression recognition, physiological sensors, heart-rate variability, electrodermal activity, behavioral biometrics, burnout detection.

Introduction. Remote and hybrid work arrangements have become widespread since the COVID-19 pandemic. In the European Union, the share of employed persons usually working from home rose from 14 % in 2019 to 22 % in 2023, and roughly one-third of all working days were remote by mid-2023 [1] [2]. While remote work offers flexibility and reduced commuting, it also correlates with elevated stress, anxiety, and burnout. For instance, Wang et al. found that among 459 first-time remote workers, 19.6 % experienced moderate-to-severe anxiety and 19.6 % reported moderate-to-severe stress [3]. Despite these concerns, most corporate

Рисунок Г.1 – Публікація наукової статті

UDC 001.1

The 11th International scientific and practical conference "Current trends in scientific research development" (June 5-7, 2025) BoScience Publisher, Boston, USA. 2025. 844 p.

ISBN 978-1-73981-122-8

The recommended citation for this publication is:

Ivanov I. Analysis of the phaunistic composition of Ukraine // Current trends in scientific research development. Proceedings of the 11th International scientific and practical conference. BoScience Publisher. Boston, USA. 2025. Pp. 21-27. URL: https://sci-conf.com.ua/xi-mizhnarodna-naukovo-praktichna-konferentsiya-current-trends-in-scientific-research-development-5-7-06-2025-boston-ssha-arhiv/.

Editor
Komarytskyy M.L.
Ph.D. in Economics, Associate Professor

Collection of scientific articles published is the scientific and practical publication, which contains scientific articles of students, graduate students, Candidates and Doctors of Sciences, research workers and practitioners from Europe, Ukraine and from neighbouring countries and beyond. The articles contain the study, reflecting the processes and changes in the structure of modern science. The collection of scientific articles is for students, postgraduate students, doctoral candidates, teachers, researchers, practitioners and people interested in the trends of modern science development.

e-mail: boston@sci-conf.com.ua

homepage: *https://sci-conf.com.ua*

Рисунок Г.2 – Журнал конференції

ДОДАТОК Д

експертний висновок результатів перевірки кваліфікаційної роботи на відповідність оформлення вимогам ДСТУ 3008: 2015

Експертний висновок результатів перевірки кваліфікаційної роботи

студент (посада) | програмної інженерії (кафедра) | ІПЗм-23-4 (група)

Кіданов Артем Олександрович
(прізвище, ім'я, по батькові)

Зауваження

Пункт ДСТУ 3008-2015	Зміст пункту	Сторінка кваліфікаційної роботи
1	2	3
	7.1 Загальні положення	
	7.3 Нумерація сторінок звіту	
	7.4 Нумерація розділів, підрозділів, пунктів, підпунктів	
	7.5 Рисунки	
	7.6 Таблиці	
	7.7 Переліки	
	7.8 Примітки	
	7.9 Виноски	
	7.10 Формули та рівняння	
	7.11 Посилання	
	7.13 Список авторів	
	7.14 Скорочення та умовні познаки	
	7.15 Додатки	

зауважень немає

Експерт _____ (підпис) Олена ОЛІЙНИК (прізвище, ініціали)

18.06.2025

Рисунок Д.1 – Висновок

ДОДАТОК Е

Приклади програмного коду

Генератор тестових даних:

```
import asyncio
import random
import numpy as np
from datetime import datetime, timezone
from typing import Dict, Any
import json
import math

from influx.influx_writer import InfluxWriter

class PhysioMockGenerator:
def init(self, user_ids: list, influx_writer: InfluxWriter,
generation_interval: float = 1.0):
self.user_ids = user_ids
self.influx_writer = influx_writer
self.generation_interval = generation_interval
self.running = False

# Enhanced base values for each user with individual characteristics
self.user_profiles = {}
for user_id in user_ids:
self.user_profiles[user_id] = self._create_user_profile()

# Activity state tracking
self.activity_states = {}
for user_id in user_ids:
self.activity_states[user_id] = {
'current_activity': 'resting',
'activity_start_time': datetime.now(timezone.utc),
'activity_duration': 0,
'stress_episode': False,
'stress_start_time': None,
'daily_step_count': 0
}

def _create_user_profile(self) -> Dict[str, Any]:
"""Create individual user profile with realistic characteristics"""
fitness_level = random.choice(['low', 'medium', 'high'])
stress_tendency = random.choice(['low', 'medium', 'high'])

# Base physiological values based on fitness and stress tendency
if fitness_level == 'high':
base_hr = random.randint(55, 70)
hrv_baseline = random.uniform(35, 55)
elif fitness_level == 'medium':
base_hr = random.randint(65, 80)
hrv_baseline = random.uniform(25, 40)
else:
base_hr = random.randint(75, 90)
hrv_baseline = random.uniform(15, 30)
```

```
return {
'fitness_level': fitness_level,
'stress_tendency': stress_tendency,
'base_heart_rate': base_hr,
'hrv_baseline': hrv_baseline,
'base_eda': random.uniform(0.2, 0.5),
'circadian_phase': random.uniform(0, 2 * math.pi), # Individual circadian
rhythm
'activity_preference': random.choice(['morning', 'afternoon', 'evening']),
'step_counter': 0
}

def generate_heart_rate(self, user_id: str) -> int:
profile = self.user_profiles[user_id]
state = self.activity_states[user_id]
current_time = datetime.now(timezone.utc)

base_hr = profile['base_heart_rate']

# Circadian rhythm effect (5-10 bpm variation)
hour_of_day = current_time.hour
circadian_effect = 5 * math.sin(2 * math.pi * (hour_of_day - 6) / 24 +
profile['circadian_phase'])

# Activity effect
activity_boost = 0
if state['current_activity'] == 'walking':
activity_boost = random.randint(15, 25)
elif state['current_activity'] == 'running':
activity_boost = random.randint(40, 60)
elif state['current_activity'] == 'exercising':
activity_boost = random.randint(30, 50)

# Stress episode effect
stress_boost = 0
if state['stress_episode']:
stress_duration = (current_time -
state['stress_start_time']).total_seconds() / 60
if stress_duration < 10: # Stress episode lasts up to 10 minutes
stress_boost = random.randint(10, 20)
else:
state['stress_episode'] = False

# Random noise
noise = random.gauss(0, 3)

# Calculate final HR
hr = base_hr + circadian_effect + activity_boost + stress_boost + noise

return max(45, min(180, int(hr)))

def generate_eda(self, user_id: str) -> float:
profile = self.user_profiles[user_id]
state = self.activity_states[user_id]

base_eda = profile['base_eda']

# Slow drift for baseline EDA
```

```
drift = random.gauss(0, 0.02) > Artem: profile['base_eda'] = max(0.1,
min(0.8, base_eda + drift))

# Activity effect on EDA
activity_boost = 0
if state['current_activity'] in ['walking', 'running', 'exercising']:
activity_boost = random.uniform(0.05, 0.15)

# Stress episode effect
stress_boost = 0
if state['stress_episode']:
stress_boost = random.uniform(0.1, 0.3)

# Occasional EDA spikes (emotional responses)
spike = 0
if random.random() < 0.05: # 5% chance of EDA spike
spike = random.uniform(0.1, 0.4)

eda = profile['base_eda'] + activity_boost + stress_boost + spike
return round(max(0.1, min(1.0, eda)), 3)

def generate_steps(self, user_id: str) -> int:
profile = self.user_profiles[user_id]
state = self.activity_states[user_id]
current_time = datetime.now(timezone.utc)

# Update activity state
self._update_activity_state(user_id, current_time)

steps = 0
if state['current_activity'] == 'walking':
steps = random.randint(8, 15) # Walking pace
elif state['current_activity'] == 'running':
steps = random.randint(20, 35) # Running pace
elif state['current_activity'] == 'exercising':
steps = random.randint(5, 12) # Exercise movements
elif random.random() < 0.1: # Occasional movement while resting
steps = random.randint(1, 3)

# Update daily step counter
state['daily_step_count'] += steps
profile['step_counter'] += steps

return steps

def _update_activity_state(self, user_id: str, current_time: datetime):
"""Update user's activity state based on realistic patterns"""
state = self.activity_states[user_id]
profile = self.user_profiles[user_id]

# Calculate activity duration
activity_duration = (current_time -
state['activity_start_time']).total_seconds() / 60

# Determine if activity should change
should_change_activity = False

if state['current_activity'] == 'resting':
```

```
# Probability of starting activity based on time of day and preference
hour = current_time.hour
activity_prob = self._get_activity_probability(hour,
profile['activity_preference'])

if random.random() < activity_prob and activity_duration > 5: # At least 5
min rest
should_change_activity = True
else:
# Active state - check if should return to rest
max_duration = {
'walking': random.randint(5, 15),
'running': random.randint(3, 8),
'exercising': random.randint(10, 25)
}

if activity_duration > max_duration.get(state['current_activity'], 10):
should_change_activity = True

# Change activity if needed
if should_change_activity:
if state['current_activity'] == 'resting':
# Start new activity
activities = ['walking', 'running', 'exercising']
weights = [0.6, 0.2, 0.2] # Walking most common
state['current_activity'] = random.choices(activities, weights=weights)[0]
else:
# Return to rest
state['current_activity'] = 'resting'

state['activity_start_time'] = current_time

# Randomly trigger stress episodes
if not state['stress_episode'] and random.random() < 0.002: # 0.2% chance
per second
if profile['stress_tendency'] == 'high':
stress_chance = 0.005
elif profile['stress_tendency'] == 'medium':
stress_chance = 0.003
else:
stress_chance = 0.001

if random.random() < stress_chance:
state['stress_episode'] = True > Artem: state['stress_start_time'] =
current_time

def _get_activity_probability(self, hour: int, preference: str) -> float:
"""Get probability of starting activity based on time and preference"""
base_prob = 0.001 # Base probability per second

# Time-based modifiers
if 6 <= hour <= 9: # Morning
time_modifier = 1.5 if preference == 'morning' else 1.0
elif 12 <= hour <= 14: # Lunch time
time_modifier = 1.2
elif 17 <= hour <= 19: # Evening
time_modifier = 1.5 if preference == 'evening' else 1.0
elif 9 <= hour <= 17: # Work hours
```

```
time_modifier = 0.3
else: # Night/early morning
time_modifier = 0.1

return base_prob * time_modifier

def generate_sensor_quality(self) -> str:
# More realistic sensor quality distribution
rand = random.random()
if rand < 0.92:
return "good"
elif rand < 0.97:
return "fair"
else:
return "poor"

def generate_data_point(self, user_id: str) -> Dict[str, Any]:
timestamp = datetime.now(timezone.utc)

return {
"user_id": user_id,
"timestamp": timestamp.isoformat(),
"heart_rate": self.generate_heart_rate(user_id),
"eda": self.generate_eda(user_id),
"steps": self.generate_steps(user_id),
"sensor_quality": self.generate_sensor_quality()
}

async def generate_for_user(self, user_id: str):
while self.running:
try:
data_point = self.generate_data_point(user_id)
await self.influx_writer.write_data_point(data_point)

# Enhanced logging with activity state
state = self.activity_states[user_id]
activity_info = f"[{state['current_activity']}]"
if state['stress_episode']:
activity_info += " [STRESS]"

print(f"Generated data for {user_id} {activity_info}:
HR={data_point['heart_rate']}, EDA={data_point['eda']},
Steps={data_point['steps']}")
await asyncio.sleep(self.generation_interval)
except Exception as e:
print(f"Error generating data for {user_id}: {e}")
await asyncio.sleep(1)

async def start(self):
self.running = True
print(f"Starting enhanced physio data generation for users:
{self.user_ids}")

# Print user profiles
for user_id, profile in self.user_profiles.items():
print(f"{user_id}: {profile['fitness_level']} fitness,
{profile['stress_tendency']} stress tendency, prefers
{profile['activity_preference']} activity")
```

```
tasks = []
for user_id in self.user_ids:
task = asyncio.create_task(self.generate_for_user(user_id))
tasks.append(task)

await asyncio.gather(*tasks)

def stop(self):
print("Stopping enhanced physio data generation")
self.running = False
```

Інтерпретатор стресу за фізіологічними даними:

```
from typing import Dict, Any
from datetime import datetime

class StressInterpreter:
    def init(self):
        # Baseline thresholds for stress assessment
        self.hr_baseline = 75.0
        self.hrv_low_threshold = 30.0
        self.eda_spike_threshold = 3
        self.movement_low_threshold = 0.05
        self.stress_index_high = 300

        # Historical stress tracking for burnout detection
        self.stress_history = {}

    def interpret_features(self, user_id: str, features: Dict[str, float],
window_start: str, window_end: str) -> Dict[str, Any]:
        # Calculate stress score
        stress_score = self._calculate_stress_score(features)

        # Determine stress level
        stress_level = self._determine_stress_level(stress_score)

        # Generate explanation
        explanation = self._generate_explanation(features)

        # Check for anomalies
        anomaly_detected = self._detect_anomalies(features)

        # Update stress history and check burnout
        possible_burnout_flag = self._check_burnout_risk(user_id,
stress_score)

        return {
            "user_id": user_id,
            "window_start": window_start,
            "window_end": window_end,
            "stress_level": stress_level,
            "stress_score": round(stress_score, 3),
            "possible_burnout_flag": possible_burnout_flag,
            "anomaly_detected": anomaly_detected,
```

```
            "source": "physiology",
            "explanation": explanation,
            "features": features
        }

    def _calculate_stress_score(self, features: Dict[str, float]) -> float:
        if features.get('data_quality') == 'poor':
            return 0.0

        score_components = []

        # Heart rate component (0-0.3)
        hr = features.get('avg_heart_rate', 0)
        if hr > 0:
            hr_score = min(0.3, max(0, (hr - self.hr_baseline) / 50.0 *
0.3))
            score_components.append(hr_score)

        # HRV component (0-0.25) - lower HRV = higher stress
        hrv = features.get('hrv_rmssd', 0)
        if hrv > 0:
            hrv_score = min(0.25, max(0, (self.hrv_low_threshold - hrv) /
self.hrv_low_threshold * 0.25))
            score_components.append(hrv_score)

        # EDA spikes component (0-0.2)
        eda_peaks = features.get('eda_peak_count', 0)
        eda_score = min(0.2, (eda_peaks / 10.0) * 0.2)
        score_components.append(eda_score)

        # Movement component (0-0.15) - low movement = higher stress
        movement = features.get('movement_mean', 0)
        movement_score = min(0.15, max(0, (self.movement_low_threshold -
movement) / self.movement_low_threshold * 0.15))
        score_components.append(movement_score)

        # Stress index component (0-0.1)
        stress_index = features.get('stress_index', 0)
        si_score = min(0.1, (stress_index / self.stress_index_high) * 0.1)
        score_components.append(si_score)

        # Combine components
        total_score = sum(score_components)

        # Normalize to 0-1 range
        return min(1.0, max(0.0, total_score))

    def _determine_stress_level(self, stress_score: float) -> str:
        if stress_score < 0.3:
            return "low"
        elif stress_score < 0.6:
            return "moderate"
        elif stress_score < 0.8:
            return "high"
        else:
            return "critical"
```

```
    def _generate_explanation(self, features: Dict[str, float]) ->
Dict[str, bool]:
        explanation = {}

        # Heart rate above baseline
        hr = features.get('avg_heart_rate', 0)
        explanation['heart_rate_above_baseline'] = hr > self.hr_baseline

        # EDA spikes detected
        eda_peaks = features.get('eda_peak_count', 0)
        explanation['eda_spikes_detected'] = eda_peaks >=
self.eda_spike_threshold

        # Low movement
        movement = features.get('movement_mean', 0)

> Artem:
explanation['movement_low'] = movement < self.movement_low_threshold

        # Low HRV
        hrv = features.get('hrv_rmssd', 0)
        explanation['hrv_low'] = hrv < self.hrv_low_threshold and hrv > 0

        # High stress index
        stress_index = features.get('stress_index', 0)
        explanation['stress_index_elevated'] = stress_index >
self.stress_index_high

        # Data quality issues
        data_quality = features.get('data_quality', 'poor')
        explanation['data_quality_poor'] = data_quality == 'poor'

        return explanation

    def _detect_anomalies(self, features: Dict[str, float]) -> bool:
        # Check for wildly inconsistent values
        hr = features.get('avg_heart_rate', 0)
        eda = features.get('eda_mean', 0)

        # Extreme heart rate
        if hr > 150 or (hr > 0 and hr < 40):
            return True

        # EDA flatline (exactly 0 for extended period)
        if eda == 0.0 and features.get('eda_peak_count', 0) == 0:
            return True

        # Impossible stress index values
        stress_index = features.get('stress_index', 0)
        if stress_index > 1000:
            return True

        return False

    def _check_burnout_risk(self, user_id: str, stress_score: float) ->
bool:
        # Initialize user history if not exists
        if user_id not in self.stress_history:
```

```
            self.stress_history[user_id] = []

        # Add current stress score to history
        self.stress_history[user_id].append(stress_score)

        # Keep only last 10 measurements for burnout assessment
        if len(self.stress_history[user_id]) > 10:
            self.stress_history[user_id] = self.stress_history[user_id][-
10:]

        # Check if last 3 measurements are high stress (>0.8)
        recent_scores = self.stress_history[user_id][-3:]

        if len(recent_scores) >= 3:
            high_stress_count = sum(1 for score in recent_scores if score >
0.8)
            return high_stress_count >= 3

        return False

    def get_user_stress_history(self, user_id: str) -> list:
        return self.stress_history.get(user_id, [])
```