МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ТАРАСА ШЕВЧЕНКА
Факультет інформаційних технологій
Кафедра інтелектуальних технологій

ВИПУСКНА КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА
НА ТЕМУ

Підсистема моніторингу стану виснаження на роботі на основі семантичного аналізу твітів співробітників

Галузь знань 12 «Інформаційні технології»
Спеціальність 122 «Комп'ютерні науки»
Освітня програма «Комп'ютерні науки»
Освітній рівень: бакалавр

Виконала: студентка 4 курсу, групи КН- 41
Боженко Д.М._____
(прізвище та ініціали)
Керівник Доманецька І.М_____
(прізвище та ініціали)
*к.т.н., доцент*_____
(науковий ступінь, звання)

Випускна кваліфікаційна робота бакалавра допущена до захисту
рішенням кафедри *інтелектуальних технологій*
Протокол № 13 від 05.06.2023 р.
зав. кафедри_____ доц. Іларіонов О.Є.

Київ – 2023

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ТАРАСА ШЕВЧЕНКА

Факультет інформаційних технологій

Кафедра інтелектуальних технологій

Спеціальність 122 «Комп'ютерні науки»

ЗАТВЕРДЖУЮ:
Зав. кафедри інтелектуальних технологій
_____
(звання, прізвище та ініціали)
_____
(підпис)
«___» _____20__ р.

ЗАВДАННЯ

НА ВИПУСКНУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТОВІ

Боженко Дар'я Миколаївна

(прізвище, ім'я, по батькові)

1. Тема роботи

Підсистема моніторингу стану виснаження на роботі на основі семантичного аналізу твітів співробітників
затверджена протоколом засідання кафедри від « 11 » листопада 2022 р. протокол №4

2. Термін здачі студентом закінченого роботи 30 травня

3. Вихідні дані до роботи

а) набір даних з заздалегідь класифікованими даними за класами, які визначають стрес та самотність, а також звичайні твіти;
б) набір даних з інформацією про працівників.

4. Зміст пояснювальної записки (перелік питань, що їх належить розробити)

а) Аналіз існуючих методів та моделей аналізу виснаження за допомогою аналізу соціальної платформи Twitter.
б) Проектування застосунку
в) Програмна реалізація продукту

5. Перелік презентаційного матеріалу (з точним зазначенням обов'язкових презентацій)

1. Постановка задачі: мета, об'єкт, предмет, вимоги (2)
2. Предметна область: огляд існуючих методів та підходів, вибір методу вирішення задачі (3)
3. Проектування застосунку: функціональне моделювання роботи підсистеми, проектування БД, розробка архітектури застосунку (5)
4. Програмна реалізація застосунку (6)

6. Консультанти з випускної кваліфікаційної роботи із зазначенням її розділів, що їх стосуються

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання

__15.02.2023_____

Керівник _____ / Доманецька І. М. /

(підпис) (ініціали та прізвище)

Завдання прийняв до виконання _____ /Боженко Д.М. /

(підпис) (ініціали та прізвище)

КАЛЕНДАРНИЙ ПЛАН

Пор. №	Назва етапів випускної кваліфікаційної роботи	Термін виконання етапів випускної кваліфікаційної роботи	Примітка
1	Опрацювання літератури	22.01 – 01.02	
2	Робота над розділом 1. Аналіз проблеми виявлення ознак виснаження. Постановка задачі	01.02 – 01.03	
3	Робота над розділом 2. Проектування додатку	01.03 – 10.04	
4	Робота над розділом 3. Розробка застосунку	10.04 – 10.05	
5	Робота над оформленням пояснювальної записки	10.05 – 19.05	
6	Робота над презентацією	19.05 – 28.05	

Студент _____ /Боженко Д.М. /

(підпис) (ініціали та прізвище)

Керівник випускної кваліфікаційної роботи _____ / Доманецька І. М. /

(підпис) (ініціали та прізвище)

Анотація

Боженко Дар’я Миколаївна виконала випускну кваліфікаційну роботу на тему «Підсистема моніторингу стану виснаження на роботі на основі семантичного аналізу твітів співробітників» за спеціальністю 122 – «Комп’ютерні науки».

У випускній кваліфікаційній роботі проведено аналіз методологій класифікації текстів, проведено дослідження параметрів та їх вплив на роботу моделі, розроблено веб-застосунок з інтелектуальним ядром на основі методу опорних векторів, для визначення ризику виснаження працівників на основі аналізу текстів твітів.

Ключові слова: аналіз тексту, метод опорних векторів, класифікація тексту.

Summary

The degree project: «Subsystem for monitoring the state of exhaustion at work based on the semantic analysis of employee tweets» has completed by Daria Bozhenko specialty 122 – «Computer Sciences».

In this graduation thesis the text classification methodologies were analyzed and compared, a study of parameters and their influence on the model was carried out, a web application with an intelligent core based on the support vector machines method was developed to determine the risk of employee burnout based on tweet texts.

Keywords: text analysis, support vector machines, text classification.

ЗМІСТ

Перелік умовних позначень і скорочень

NLP – обробка природньої мови

БД – база даних

SVM – метод опорних векторів

WSD – усунення неоднозначності в змісті слова

ШІ – штучний інтелект

СУБД – система управління базами даних

ВСТУП

Двадцять перше століття називають віком технологій, час де розвиток відбувається в рази швидше ніж в попередні роки. І хоч з прогресом ми отримуємо безліч переваг і нових можливостей, не варто й забувати про зворотній бік цього явища. Пандемія, нестабільна політична ситуація у світі, екологічні проблеми, інфляція, це все корелюється зі збільшенням рівню стресу, який відчувають люди. Проте окрім глобальних проблем, великих змін зазнав і робочий простір. Збільшення вимог до навичок, велика конкуренція, культ продуктивності, швидка зміна технологій – це все продукує підвищений рівень стресу на роботі.

Ось деяка статистика[1] опублікована у 2022 році щодо стресу на робочому місці в США:

80% співробітників кажуть, що відчувають певний рівень стресу на робочому місці;

26% співробітників кажуть, що стрес є причиною вигорання на роботі;

50% співробітників кажуть, що їм потрібна допомога в боротьбі зі стресом;

63% працівників повідомляють, що стрес, пов’язаний з роботою, спричинив проблеми вдома;

63% співробітників готові звільнитися з роботи через стрес на роботі;

Більше половини працівників повідомили, що стрес негативно впливає на їх роботу.

Як бачимо, стрес це дуже поширена проблема, з якою необхідно щось робити, адже окрім негативного впливу стресу на робочому місці на фізичне та психічне здоров’я працівників, він також має високу ціну у вигляді втрати продуктивності. За даними Фонду психічного здоров’я Великобританії, працівники, які перебувають у стресовому стані, втрачають у середньому 24 дні роботи щороку через погане здоров’я.

Стрес на робочому місці може не тільки спричинити хвороби, але й старіння раніше часу.

Дослідження, проведене Мічиганським університетом, показало, що стрес скорочує теломери, мікроскопічні органели, відповідальні за реплікацію клітин. Теломери природним чином скорочуються з часом і часто пов’язані зі старінням і віковими захворюваннями. [1]

Мета: створення автоматизованих засобів моніторингу показників працівників, які можуть вказувати на виснаження на роботі за допомогою методів штучного інтелекту.

Об’єкт дослідження: виявлення ознак стресу й виснаження у людини.

Предмет дослідження: аналіз текстових записів (твітів) з мережі Twitter для визначення рівню виснаження.

1 ВИЯВЛЕННЯ ПОКАЗНИКІВ ВИСНАЖЕННЯ, ЯК ПРОБЛЕМА ШТУЧНОГО ІНТЕЛЕКТУ

1.1 Аналіз особливостей задачі виявлення ознак виснаження працівників

Емоційний стан людей все більше й більше привертає увагу науковців в останнє десятиліття. Порівняно з 20 роками тому люди вдвічі частіше повідомляють, що вони завжди виснажені. Близько 50% людей кажуть, що вони часто або постійно виснажуються роботою, це на 32% більше, ніж два десятиліття тому. Більше того, існує суттєва кореляція між почуттям самотності та втомою від роботи: чим більше люди виснажені, тим самотнішими вони почуваються. [2]

Занадто сильний стрес може спричинити певні проблеми, такі як втрата сну, дратівливість, болі в спині чи головні болі, а також може сприяти потенційно небезпечним для життя захворюванням, таким як високий кров'яний тиск і хвороби серця.[3]

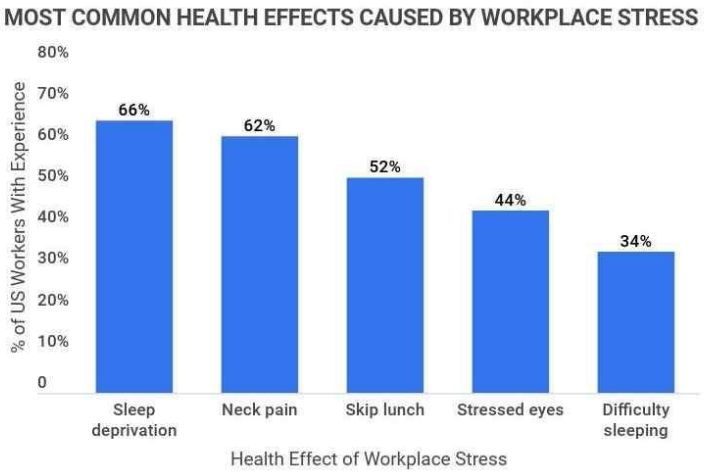

Рисунок 1.1 – Найбільш поширені проблеми викликані стресом[4]

Постійний стрес негативно впливає не тільки на людину, але й на все чим людина займається. Опитування ComPsych показало, що 37% людей кажуть, що втрачають годину або більше на день у продуктивності через стрес. Більше половини говорять, що стрес змушує їх пропускати роботу принаймні один день на рік. [3]

З представлених даних, можна з точністю сказати, що моніторинг емоційного стану співробітників й надання їм допомоги в кризових ситуаціях має бути обов'язковою частиною робочого процесу.

Проте постає проблема у виконанні цієї задачі. Немає одного стандартного показника, щоб сказати що людина переживає.

Покладатися на те що працівник сам скаже менеджеру про виснаження ненадійно. Адже людина може боятися звільнення, пониження в посаді або окладі, зміни відношення або ігноруванні звернення.

Одним з варіантом визначення рівню виснаженості є тестування, і хоч тести розроблені психологами можуть точно визначити проблему, немає ніякої гарантії, що працівники будуть надавати правдиві відповіді, при чому це може бути як з однієї сторони так і з іншої.

Іншим варіантом, на якому буде зосереджена увага в даній дипломній роботі, є аналіз текстів мікроблогів співробітників. Варто зазначити, що даний підхід може застосовуватись для будь-якої соціальної мережі або корпоративного сервісу, в даному випадку було обрано мережу Twitter. Одним з плюсів Twitter, як платформи для аналізу, є те що все більше людей діляться в своїх постах звичайними подіями, такі як спілкування з друзями, обід, перегляд нового фільму або переживаннями за день. Крім того Twitter забезпечує можливість простого та ефективного збору даних.

Проте існує і ряд проблем, з якими ми стикаємось в цьому підході. Так важливим питанням є збір даних для датасетів, створення ефективних моделей та інтерпретації отриманих результатів.

Датасети в більшості випадків формуються мануально людьми, а це важкий і кропіткий процес, адже необхідно забезпечити різноманіття текстового наповнення, поділити тексти на класи, навіть при пошуку за хештегами й ключовими словами це залишається великим завданням.

Інтерпретація на рівні загальної класифікації стану людини, а не конкретного тексту також постає проблемою. Адже не зрозуміло, який відсоток

твітів від усіх має мати ознаки виснаження, щоб провести межу між нормою та «патологією». Адже природньо, що навіть, якщо людина відчуває самотність або інші негативні емоції, вона не буде писати про це в кожному своєму твіті, особливо якщо це не анонімна, а офіційна сторінка.

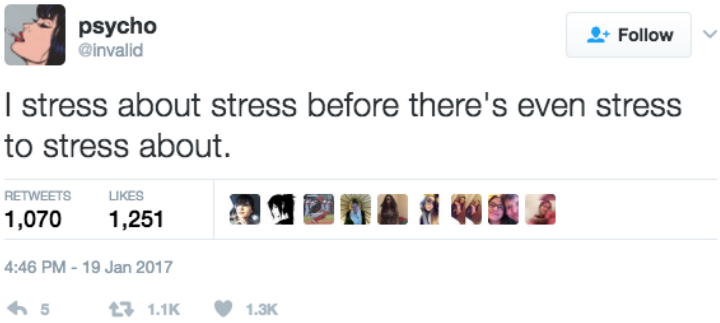

Рисунок 1.2 – Приклад твіту[5]

Також варто зазначити, що однією з проблем при використання даних Twitter полягає в тому, що дописи мають обмеження щодо кількості текстової інформації, яку вони можуть містити, що становить максимум 280 символів.[6] А також те, що максимальна кількість твітів, які можна отримати зі сторінки юзера, становить 200 твітів, і хоч для середньостатистичної людини цього буде більш ніж достатньо, необхідно пам’ятати про дане обмеження.

1.2 Задача виявлення ознак виснаження як задача штучного інтелекту

Якщо дивитися на задачу глобально, то її можна віднести до групи задач аналітики соціальних медіа, так як дані для аналізу ми отримуємо з соціальних мереж, зокрема з мережі Twitter. Аналітика соціальних мереж - це здатність збирати та знаходити сенс у даних, зібраних із соціальних каналів. [7]

Так як, дані отримані з соціальних мереж неструктуровані та великі за обсягом для їх аналізу використовують обробку природної мови (NLP), машинне навчання(ML) та методи великих даних.

Обробка природної мови може бути використана для підготовки даних для навчання моделі. За допомогою методів NLP отриманий текст твітів спрощується

- залишаються лише важливі, індикуючі слова й сполучення слів, уніфікується – зводиться до одного регістру, прибираються знаки пунктуації та зайві пробіли.

Також, задачу можна розглядати, як задачу семантичного аналізу текстів. Семантичний аналіз тексту – це процес визначення смислу та інтерпретації значення слів та фраз у контексті, в якому вони вживаються.

Проте задачу аналізу текстів на наявність ознак стресу та вигоряння можна розглядати як більш вузьку задачу сентимент-аналізу. Сентимент аналіз – це процес визначення та класифікації емоційного тону тексту, зокрема виявлення позитивних, негативних або нейтральних відтінків висловлювань.

Як можна побачити, велику роль в вирішенні даної задачі відіграють датасети - набір даних для побудови моделі. Існує декілька підходів до формування датасету:

1. Формування набору слів індикаторів
2. Формування датасету із оброблених твітів

Кращим варіантом для застосування є другий варіант, так як в такому випадку ми не лише отримуємо слова-індикатори, але й контекст в яких вони з’являються. В загальному варіанті в датасеті наявні лише дві колонки: текст та клас якому належить даний текст.

Важливим фактором також є збалансованість датасету (приблизно однакова кількість екземплярів), особливо якщо застосовується мультикласова класифікація, та розмірність датасету, адже необхідно мати достатньо прикладів для навчання, щоб модель могла спрогнозувати точні результати.

1.3 Огляд існуючих підходів, методів та моделей

Традиційне виявлення психологічного стресу в основному ґрунтується на особистих інтерв’ю, опитувальниках або датчиках.

Проте у контексті робочого простору всі три традиційних методи мають певні суттєві недоліки. Проводити особисті інтерв’ю з кожним співробітником

це дуже часомісткий процес в якому необхідно задіяти з десяток психологів, а повторювати його регулярно ще більш проблематично. При використанні опитувальників ми хоч і значно скорочуємо час, але при цьому втрачаємо гарантію достовірності результатів, адже співробітники можуть і не казати правди під час тестування. З датчиками ми стикаємося зі схожою проблемою, як і при використанні інтерв'ю: датчик повинен знаходитися на працівнику тривалий час, важкість процесу аналізу даних, незручність для працівників, адже найбільш точні результати дають датчики закріплені на грудях, що для звичайної людини в повсякденному житті може викликати багато незручностей при використанні.

Проте, ці методи все ж використовуються для вирішення поставленої задачі, як наприклад у статті "Mental Stress Detection Using Artificial Intelligence Models"[8] розглядається класифікація емоцій людини на три класи: нейтрайльний, стресовий, радісний стани. Для цього використовується датасет, який був сформований за допомогою поставлення людей в ситуації, де б вони відчували спокій, стрес та радість. Дані заміялися за допомогою пристроїв на зап'ясті та на грудях. Дані складалися з 8 функцій – прискорення пульсу за трьома, температура тіла, рівень дихання, EDA, ЕКГ та ЕМГ.

У цьому дослідженні було оцінено такі моделі як: штучна нейронна мережа (ANN), гібрид штучної нейронної мережі та опорної векторної машини (SVM), класифікатор стекування та радіальна базисна функція (RBF). Серед усіх моделей класифікатор стекування дав найкращу точність 99,92%. SVM дав точність 91,48%, штучна нейронна мережа – 90,58%, а радіальна базисна функція лише 84,48%.

Окрім традиційних методів визначення стресу представлених вище, на практиці використовується ще один вид рішення даної задачі – це виявлення стресу через аналіз текстової інформації.

Існує багато методів і моделей, які застосовуються для аналізу й класифікації текстів. Найпопулярнішими техніками машинного навчання для даної проблеми є:

- Support vector machines (SVM)
- Naive-Bayers (NB)
- Random Forest (RF)

Як приклад, у статті “ Detection and Analysis of Stress using Machine Learning Techniques”[9] представлений алгоритм вирішення завдання виявлення та аналізу стресу у тексті за допомогою машинного навчання, алгоритм нараховує 5 кроків.

Перший крок це отримання даних для аналізу.

Другий крок це стадія передобробки. Він складається з видалення не буквених символів, спеціальних символів, посилань та стоп слів. Стоп словами вважаються слова, які не змінюють значення речення і не впливають на визначений рівень досліджуваного показника.

Третій крок це усунення неоднозначності у значенні слів. На даному етапі використовується техніка WSD. Спочатку виконується POS тегування: цей крок класифікує слово, як частину речення. Для подальшої обробки беруться лише іменники, дієслова, прислівники та прикметники. Після використання тегів, необхідно опрацювати позицію слова, для цього використовуються N-грами (Uni, Bi, Tri-gram). За допомогою цих кроків ми отримаємо унікальне значення слова, навіть якщо воно має більше одного значення.

Четвертий крок присвячений знаходженню оцінки слів. Для цього у статті використовується фреймворк TensiStrength, який дозволяє поставити слову оцінку від -5 до +5, де -5 вказує на великий стрес, а +5 - на велике розслаблення.

На п’ятому кроці застосовується алгоритм SVM або NB, на вхід подаються знайдені оцінки. В дослідженні були наявні такі класи як депресія, стрес, розслабленість, щастя, норма. Найкращий результат був досягнутий при використанні алгоритму SVM (precision - 65%, recall - 67%).

Ідея аналізу й виявлення стресу через текстову інформацію розвивається й з’являються нові методи, які покращують існуючі алгоритмів.

Цікаву ідею, щодо вдосконалення звичайного підходу до класифікації тексту було запропоновано в статті PrashanthKVTKN та TeneRamakrishnudu “A novel method for detecting psychological stress at tweet level using neighborhood tweets” [6]

Запропонована модель - NTSD (neighborhood tweet-based stress detection), використовує не тільки сам твіт, але й сусідні твіти для визначення рівню стресу. В основу даного метод покладено ідею, що стрес - це відчуття, яке зберігається протягом тривалого часу, відповідно якщо безпосередньо попередні твіти мають ознаки стресу, то існує висока ймовірність того, що даний твіт також викликає стрес.

Для аналізу твітів використовуються такі функції:

- Підрахунок слів для десяти категорій лінгвопсихологічної бібліотеки на основі EMPATH(EMPATH — це безкоштовна бібліотека для психолого-лінгвістичного підрахунку слів);
- Підрахунок кількості позитивних та негативних дієслів;
- Підрахунок кількості позитивних та негативних прикметників;
- Обчислення інтенсивності прислівників;
- Підрахунок кількості емодзі;
- Підрахунок пунктуації;
- Визначення рівню сарказму (так як при використанні сарказму твіти не передають чітко зазначеного значення);
- Соціальні особливості твіту (уподобання, ретвіти, коментарі);

Для навчання моделі використовуються як сам твіт, так і сусідні набори даних твітів. Вікно околиці для сусідніх твітів створюється на основі кількості попередніх твітів, які розглядаються з відповідного основного твіту. Іншими словами, вікно містить останні попередні твіти користувача даного основного

твіту. Потім, як первинні, так і сусідні набори даних твітів використовуються для навчання моделі.

Дана модель використовує метод логістичної регресії з сигмоподібною або логістичною функцією.

З експериментів описаних в статті можна побачити, що запропонована модель перевищує базові моделі машинного навчання.

1.4 Вибір та обґрунтування обраного методу вирішення задачі

Першою проблемою при аналізі задачі виявився пошук даних для навчання моделі. Так як задача формування датасетів може розглядатися як окрема проблема, було прийнято рішення зупинитися на вже сформованому публічному датасеті “Behavioural Tweets” [10]. В даному наборі даних були зібрані твіти з соціальної мережі з метою відстеження психічного здоров’я. Датасет складається з 4 наборів: нормальні твіти (Normal_Tweets), твіти з ознаками тривожності (Anxious_Tweets), твіти з ознаками відчуття самотності (Lonely_Tweets), твіти з ознаками стресу (Stressed_Tweets).

Кожен файл містить понад 8000 попередньо оброблених текстів твітів, які підходять під обраний клас. Так як основна увага цієї роботи присвячена аналізу стану працівників, а як було зазначено виснаження в більшій мірі характеризується поєднанням стресу та самотності. Для подальшої роботи було обрано 3 класи: нормальні твіти, твіти, які сигналізують про ознаки відчуття самотності та твіти, які сигналізують про ознаки стресу.

Після аналізу поставленої проблеми вибір методу відбувався між методом наївного Байєса та методом опорних векторів, так як саме вони найчастіше використовувались в аналізованих роботах. Для кращого розуміння, який з методів обрати було розглянуто статтю з порівнянням різних класифікаційних моделей саме на текстових даних.[11] Результати дослідження показані на рисунку 1.3.

All Accuracies Are in Percentages and Rounded to 2 Decimal Places		LR	DT	SVM	AB	RF	NB	GB	MLP
20 News-groups	Best Test Accuracy	68.28	44.44	70.03	45.61	62.28	60.62	60.12	69.46
	Best Hyper-parameters	Regularization: l2 C: 5.0 tol: 0.0001	criterion: 'gini' stopwords: sklearn	C: 0.3 tol: 1e-05	n_estimators: 150	criterion: 'gini' n_estimators: 400	N/A	loss: 'deviance' learning_rate: 0.1 n_estimators: 200	hidden_layer_sizes: (100,) activation: 'relu'
IMDb Reviews	Best Test Accuracy	88.28	71.64	88.32	84.63	84.84	83.02	83.56	86.04
	Best Hyper-parameters	Regularization: l2 C: 2.0 tol: 0.0001	criterion: 'entropy' stopwords: nltk	C: 0.4 tol: 1e-04	n_estimators: 200	n_estimators: 200 criterion: 'entropy'	N/A	loss: 'deviance' learning_rate: 0.1 n_estimators: 200	hidden_layer_sizes: (100,) activation: 'relu'
Overall		83.63	64.84	84.09	75.59	79.62	77.83	77.13	82.20

Рисунок 1.3 – Таблиця-порівняння, у якій показано точність, отриману всіма моделями на двох наборах. Найкраща та найгірша точності тесту виділено блакитним і зеленим відповідно[11]

Як можна побачити з представленої вище таблиці, метод опорних векторів має найвищу точність на двох різних текстових наборах даних, на другому місці опинився метод лінійної регресії, а метод наївного Байєса, який також часто розглядається в роботах, показав тільки шостий результат з восьми.

Даний результат може бути пояснений тим, що більшість завдань класифікації тексту є лінійно роздільними, а модель опорних векторів вивчає лінійні порогові функції. Також SVM мають здатність добре узагальнювати навіть за наявності багатьох функцій, що є ще однією сприятливою рисою для категоризації тексту, оскільки текстові дані часто є багатовимірними.

В статті було досліджено вплив різних ядер (лінійного, гаусівського, поліноміального і сигмоїдного) та їх вплив на точність роботи SVM (рисунок 1.4).

	Linear Kernel	Gaussian Kernel	Polynomial Kernel	Sigmoid Kernel
Train Accuracy (%)	92.08 ± 0.35	97.01 ± 0.64	97.20 ± 0.42	92.13 ± 0.13
Test Accuracy (%)	74.81 ± 0.64	71.88 ± 1.27	62.44 ± 0.76	65.81 ± 0.34

Рисунок 1.4 - Таблиця, що показує точність перехресної перевірки та навчання, отриману від усіх моделей SVM[11]

З рисунку 1.4 можна побачити, що на тренувальній вибірці найкраще відпрацювали SVM з поліноміальним та гаусівським ядрами, проте на тестовій вибірці, SVM з лінійним ядром є беззаперечним переможцем.

Відповідно до описаних вище результатів, доцільно вважати, що найкращим рішенням для вирішення поставленої задачі, буде обрати модель SVM з лінійним ядром та можливістю багатокласової класифікації.

Метод опорних векторів (SVM) – це контрольований алгоритм машинного навчання, який допомагає в проблемах класифікації або регресії. Він спрямований на пошук оптимальної межі між можливими результатами. У базовій формі, лінійному розділенні, SVM намагається знайти лінію, яка максимізує розділення між двокласовим набором даних двовимірних точок простору. Узагальнюючи, мета полягає в тому, щоб знайти гіперплощину, яка максимізує відстань розділення точок даних на їхні потенційні класи в n-вимірному просторі. Точки даних з мінімальною відстанню до гіперплощини (найближчі точки) називають опорними векторами.

Існує дві стратегії мультикласової класифікації в методі опорних векторів: «один-до-одного», який розбиває багатокласову проблему на кілька проблем бінарної класифікації. Використовуючи двійковий класифікатор для кожної пари класів. «Один-до-решти», розбивка встановлюється на бінарний класифікатор для кожного класу. Це означає, що поділ враховує всі моменти, розділяючи їх на дві групи; група для балів класу та група для всіх інших балів. Наприклад, зелена лінія намагається максимально збільшити відстань між зеленими точками та всіма іншими точками одночасно (рис 1.6)[12]

Для отримання більш точних показників про рівень стресу або самотності було вирішено брати до уваги не лише передбачене значення класу, але й ймовірності того, що текст належить певному класу.

Прогнозування ймовірностей дає певну гнучкість, включаючи рішення про те, як інтерпретувати ймовірності, представлення прогнозів із невизначеністю та надання більш тонких способів оцінки навичок моделі.

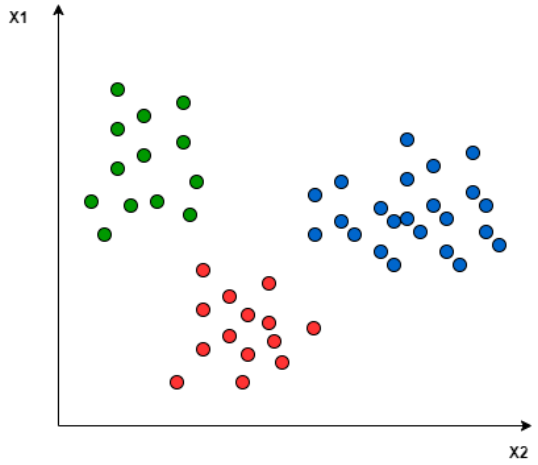

Рисунок 1.5 – Точки, які необхідно класифікувати

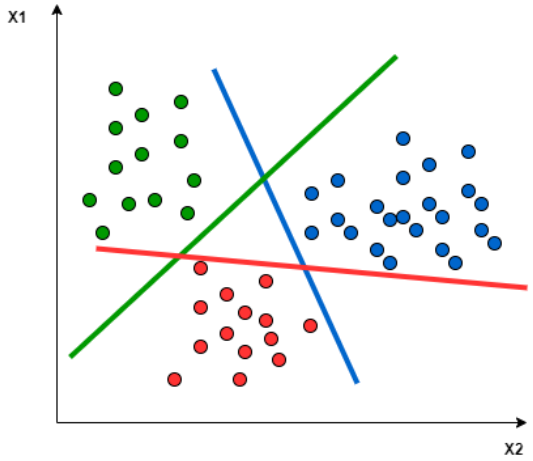

Рисунок 1.6 – Знайдені опорні вектори для класифікації

Прогнозовані ймовірності, які відповідають очікуваному розподілу ймовірностей для кожного класу, називаються каліброваними.

Доцільність даного рішення можна зрозуміти з простого прикладу. Припустимо, що ймовірність належності тексту класу N є 45%, а класу S – 44%. Модель надасть відповідь, що текст належить першому класу, хоча відмінність у ймовірності належності є невеликою.

1.5 Постановка задачі та формування вимог до застосунку

Постановка задачі: проаналізувати методи визначення вигоряння, порівняти методи аналізу тексту, розробити інтелектуальний модуль аналізу тексту на виявлення ознак виснаження, розробити веб-застосунок для репрезентації результатів роботи інтелектуального модулю.

Функціональні вимоги:

1. Доступ до системи можливий лише після авторизації;
2. Можливість задати наступні параметри для проведення аналізу: дата, посада працівника;
3. Можливість отримати результати всіх працівників з результатами сканування;

4. Можливість переглянути працівників згруповано: «Високий ризик», «Середній ризик», «Низький ризик», «Погіршились результати», «Покращились результати»;
5. Можливість візуалізації статистики результатів у вигляді діаграм з можливістю обирати часовий проміжок результатів, групування за посадою, часом роботи в компанії та датою аналізу, можливість представлення статистики діаграмами виду: секторна діаграма, стовпчаста діаграма, нормована стовпчаста діаграма та лінійна діаграма;

Нефункціональні вимоги:

1. Досягнення точності інтелектуального модуля у 90% і вище;
2. Захист конфіденційності отриманої інформації при аналізі текстів твітів;
3. Дані про працівників повинні зберігатися в структурованому вигляді під управлінням реляційної СУБД;
4. Наявність документації, яка визначає правила проведення аналізу результатів та дії в залежності від стану працівника.

Вхідна інформація: база даних в якій зберігається інформація про працівників.

Вихідна інформація: дані про наявність ризику вигоряння працівників.

Висновок до першого розділу

Як можна побачити з матеріалу викладеного вище, проблема виснаження, виявлення його ознак та усунення є важливою проблемою з якою стикається будь-який робочий колектив. Важливо помітити перші ознаки на ранніх стадіях та якнайшвидше відреагувати на них, щоб ні працівник, ні його праця не постраждали.

Існує декілька методів визначення стресу, проте для подальшої роботи було обрано метод визначення виснаження на основі текстів із соціальних мереж, так як він найбільш ефективний при великій кількості співробітників. З плюсів цього методу можна виділити швидкість проведення тестування та невелику кількість необхідних ресурсів, на відміну від методів з застосуванням фізичних датчиків або психологів. До мінусів можна віднести залежність результату від наявності соцмережі працівника, а також того наскільки відвертим він є в своїх постах.

Було проаналізовано методи із галузі штучного інтелекту які використовуються для вирішення поставленої задачі. Було обрано метод опорних векторів з лінійним ядром, ймовірнісним представленням та можливістю багатокласової класифікації. Найбільшою перевагою SVM є те, що модель добре підходить для категоризації тексту, так як має здатність добре узагальнювати навіть за наявності багатьох функцій.

Для навчання моделі було обрано датасет з текстами розділеними на класи: нормальні твіти, твіти, які сигналізують про ознаки відчуття самотності та твіти, які сигналізують про ознаки стресу. Дані класи в поєднанні можуть допомогти оцінити ймовірність виснаження працівника.

2 ПРОЕКТУВАННЯ ЗАСТОСУНКУ ВИЗНАЧЕННЯ РИЗИКУ ВИСНАЖЕННЯ ПРАЦІВНИКІВ ЧЕРЕЗ ТЕКСТИ З СОЦМЕРЕЖ

2.1 Аналіз інформаційних складових підсистеми визначення ризику виснаження

Основною інформацією, якою оперує підсистема є інформація про співробітників та інформація про результати сканування стану працівників.

Інформація про працівників включає в себе ім’я та прізвище, стать, дату народження, дату прийняття на роботу, посаду, фото, корпоративну пошту та юзернейм у Твіттері. Варто зауважити, що дана інформаціє не є повною, адже даний застосунок є підсистемою, повна інформація про роботу компанії знаходиться на головному сервері. Інформація зберігається в базі даних, а доступ до неї можна отримати через запит до БД.

Дана інформація була обрана, так як вона або необхідна для коректної роботи аналітичного модулю, або для кращого формування статистики.

Для коректної роботи моделі аналізу необхідна інформація про юзернейм працівника в Твітер, так як саме з цієї соцмережі видобуваються дані (твіти) для аналізу стану співробітника. Якщо з якоїсь причини співробітник не має аккаунту у соцмережі, не публікує дописи або не надав даної інформації, аналіз такого працівника є неможливим.

Для формування статистики та аналізу тенденції тієї чи іншої групи працівників використовується інформація про посаду, стать, час роботи в компанії.

Не менш важливою є інформація про всі результати сканування працівників, адже сама вона використовується для більш детального аналізу стану співробітника та формування загальної статистики. Кожен результат містить дату проведеного сканування, статус, який був наданий при аналізі результатів та результати, які були видані моделлю: кількість твітів

класифікованих до певного класу, а також середню відсоткову належність твітів до певного класу, за результатом ми отримуємо 6 показників.

Інформація про твіти співробітників отримуються за допомогою Twitter API у форматі об’єкту (JSON).

2.2 Методологія аналізу результатів моделі

Процес аналізу стану працівника відбувається в три етапи (рис. 2.1):

1. Аналіз твітів працівника за допомогою інтелектуального модулю;
2. Аналіз поточного результату;
3. Аналіз стану працівника з урахуванням попередніх результатів.

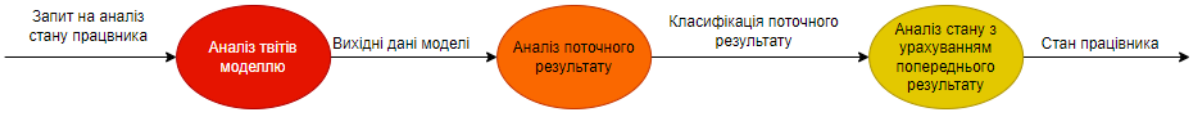

Рисунок 2.1 – Послідовність аналізу стану працівника

На першому етапі модель використовує твіти працівника для їх аналізу за допомогою методу опорних векторів. На виході отримуємо наступні дані:

1. Кількість твітів класифікованих як нормальні;
2. Кількість твітів класифікованих, як ті, що мають ознаки стресу;
3. Кількість твітів класифікованих, як ті, що мають ознаки самотності;
4. Середнє відсоткове значення належності твітів до класу нормальних;
5. Середнє відсоткове значення належності твітів до класу з ознаками стресу;
6. Середнє відсоткове значення належності твітів до класу з ознаками самотності.

На другому етапі аналізу формується висновок належності результату до одного з 4 класів: високий ризик, середній ризик, низький ризик та відсутність даних. При аналізі результатів зважаємо на дані, що виснаження складається з

поєднання декількох станів: самотності та стресу. Правила можна побачити в таблиці 2.1.

Однак, якщо за критеріями таблиці 2.1, стан працівника було класифіковано як низький ризик, але існує зміна в кількості опублікованих твітів за місяць більше ніж на 70% при кількості твітів більше 15, тоді працівник класифікується з середнім ризиком.

Таблиця 2.1 – Правила класифікації результатів

Показники відсоткові	Загальна к-сть твітів	Клас
-	0	Відсутність даних
Нормальність > 75%	-	Низький ризик
Нормальність <75%	-	Середній ризик
Ознаки стресу + самотності > 35%	-	Високий ризик

На третьому етапі для аналізу використовуються результати другого етапу а також результат попереднього сканування. На виході отримуємо два показники стан та прогрес стану. Прогрес стану описує тенденцію працівника й описується трьома значеннями: погіршився, стабільний та покращився. Окрім порівняння двох станів (попереднього та нового) для визначення стану використовується значення загальної кількості опублікованих твітів користувачем. Якщо кількість опублікованих твітів змінилась більше ніж на 70%, при кількості твітів більше 15, стан працівника встановлюється на «Середній ризик», адже великі зміни в активності користувача, є приводом для більш детального аналізу.

2.3 Функціональне моделювання роботи підсистеми моніторингу емоційного стану співробітників

Для кращого розуміння предметної області та розуміння процесів було вирішено побудувати діаграму в нотації IDEF0. IDEF0 – це метод функціонального моделювання призначений для моделювання рішень, дій і

діяльності організації або системи. Даний підхід можна використовувати для визначення вимог і специфікацій функцій, а потім для розробки реалізації.

Розглянемо основні кордони обстеження предметної області:

Предметна область – аналіз емоційного стану працівників для виявлення ознак вигоряння.

Територіальні межі – довільна компанія, яка піклується про стан своїх працівників.

Період обстеження – кожен місяць.

Точка зору – працівник який відповідає за добробут співробітників.

Функції – аналіз стану працівників за визначений період, формування звітів про стан працівників, аналіз стану окремого працівника, аналіз статистики стану працівників, виділення працівників в зоні ризику.

Контекстна діаграма процесу «Підсистема моніторингу емоційного стану співробітників» ЯК БУДЕ (TO BE) наведена на рисунку 2.2.

Вхідні дані:

- Дані працівників;
- Статистика станів працівників;
- Дані про твіти працівників;
- Параметри сканування.

Вихідні дані:

- Статистика стану кожного працівника;
- Дані працівників, які знаходяться в високому ризику вигоряння;
- Статистика за обраними показниками;
- Статистика за обраним працівником.

Керівні документи:

- Політика конфіденційності інформації.

Механізми:

- Менеджер з добробуту працівників;

- Програмний застосунок.

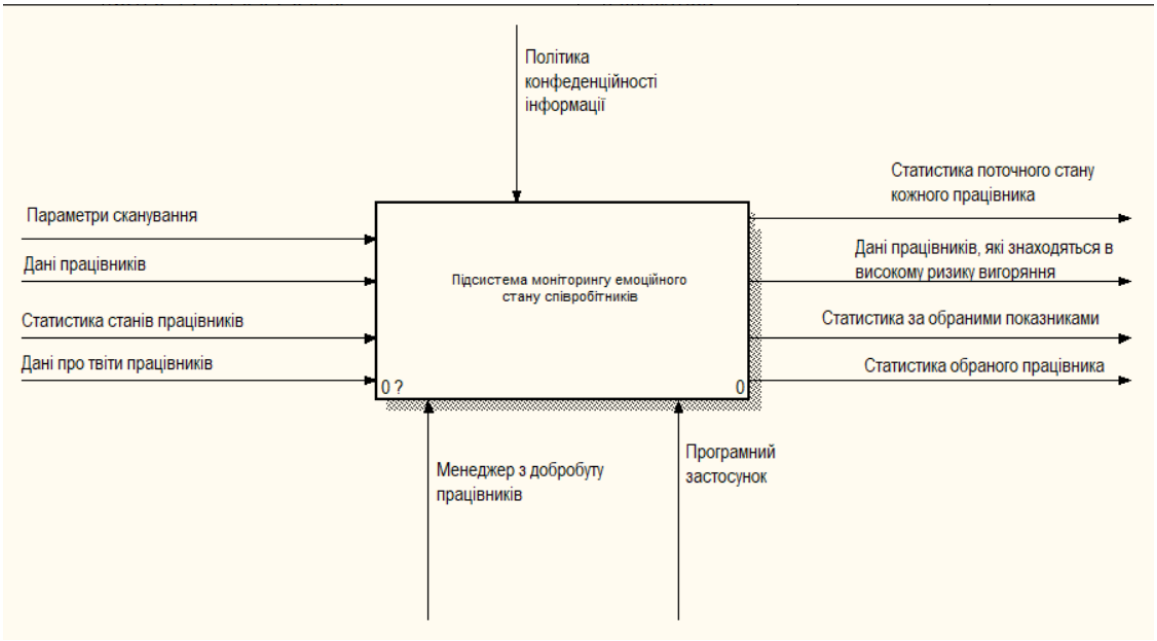

Рисунок 2.2 - Контекстна діаграма процесу «Підсистема моніторингу емоційного стану співробітників» ЯК БУДЕ

Розглянемо декомпозицію контекстної діаграми «Підсистема моніторингу емоційного стану співробітників» (рис. 2.3).

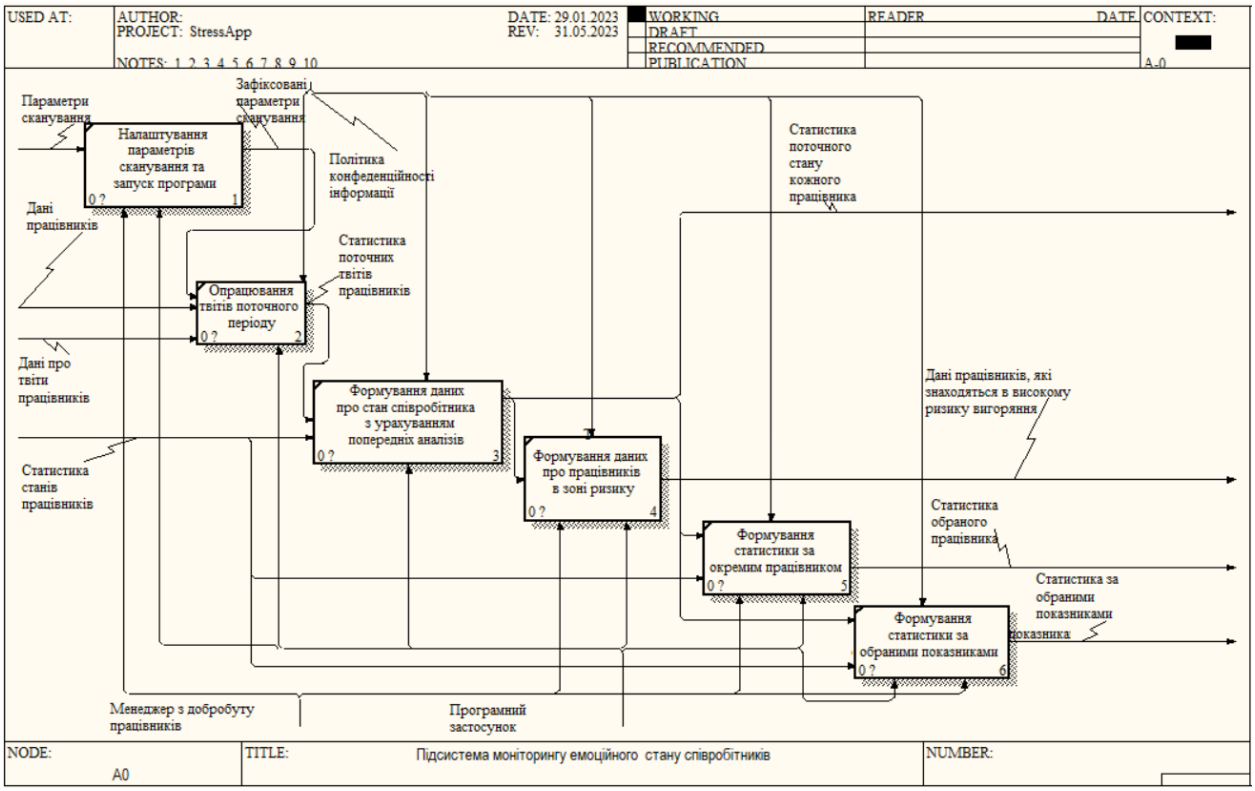

Рисунок 2.3 – Декомпозиція контекстної діаграми процесу «Підсистема моніторингу емоційного стану співробітників» ЯК БУДЕ

Опис бізнес процесів наведений у таблиці 2.2.

Таблиця 2.2 – Опис бізнес процесів ЯК БУДЕ

№	Назва процесу	Вхідна інформація	Вихідна інформація	Управління	Механізми
1	Налаштування параметрів сканування та запуск програми	Параметри сканування	Зафіксовані параметри сканування	Політика конфіденційності інформації	Менеджер з добробуту працівників, Програмний застосунок
2	Опрацювання твітів поточного періоду	Дані працівників, Дані про твіти працівників, Зафіксовані параметри сканування	Статистика поточних твітів працівників	Політика конфіденційності інформації	Програмний застосунок
3	Формування даних про стан співробітників з урахуванням попередніх аналізів	Статистика поточних твітів працівників, Статистика станів працівників	Статистика поточного стану кожного працівника	Політика конфіденційності інформації	Програмний застосунок
4	Формування даних про працівників у зоні ризику	Статистика поточного стану кожного працівника	Дані працівників, у високій зоні ризику	Політика конфіденційності інформації	Менеджер з добробуту працівників, Програмний застосунок
5	Формування статистики за окремим працівником	Статистика поточного стану кожного працівника, Статистика станів працівників	Статистика за обраним працівником	Політика конфіденційності інформації	Менеджер з добробуту працівників, Програмний застосунок
6	Формування статистики за обраними показниками	Статистика поточного стану кожного працівника, Статистика станів працівників	Статистика за обраними показниками	Політика конфіденційності інформації	Менеджер з добробуту працівників, Програмний застосунок

2.4 Проектування бази даних застосунку

Зважаючи на структурованість даних які зберігатимуться в базі даних, було вирішено використовувати реляційну СУБД. Для того щоб наглядно показати структуру бази даних було змодельовано концептуальну модель предметного середовища, логічну та фізичну моделі даних, які зображені на рисунку 2.4, 2.5, 2.6. На концептуальній моделі можна побачити дві сильні сутності – Стан, Працівник, та дві слабкі – Результат, Нікнейм які залежать від сутності Працівник.

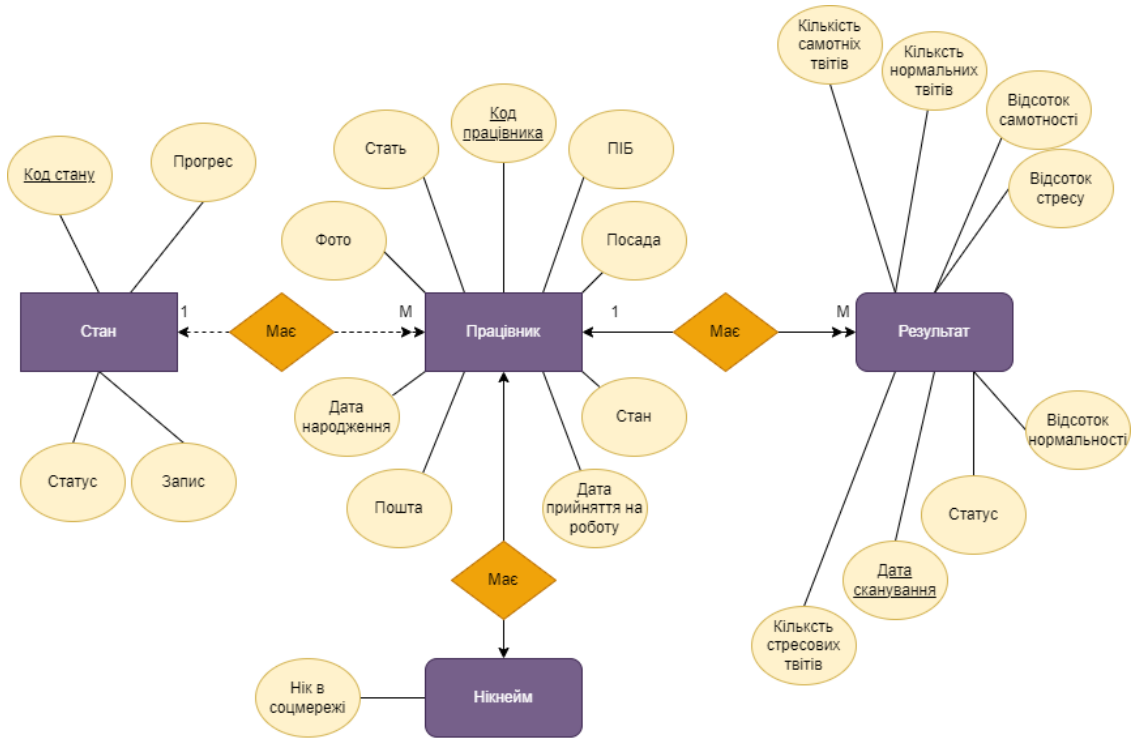

Рисунок 2.4 – Концептуальна модель бази даних

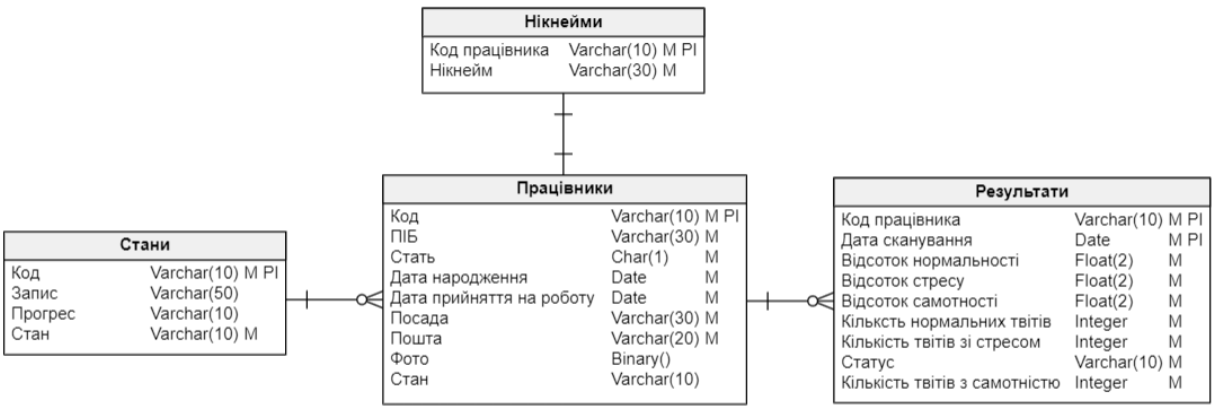

Рисунок 2.5 – Логічна модель бази даних

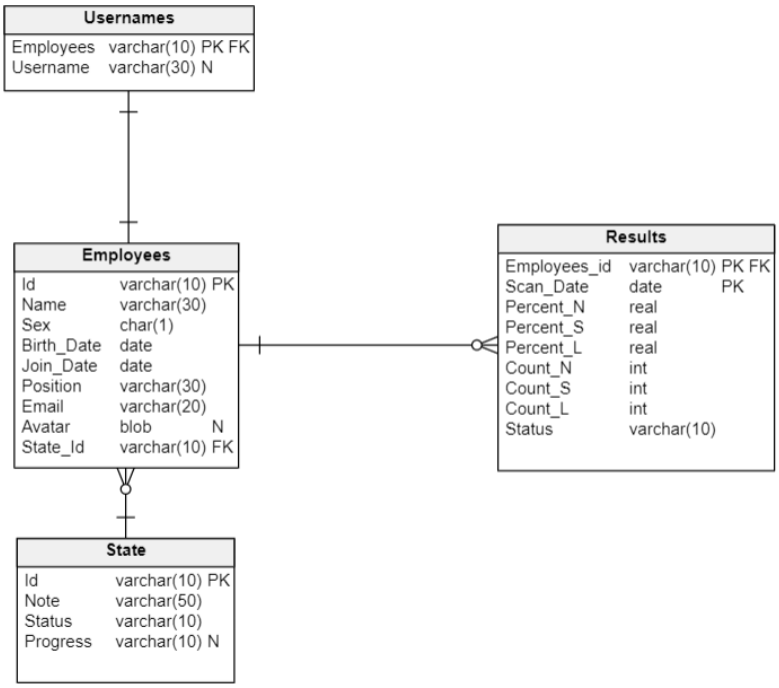

Рисунок 2.6 – Фізична модель бази даних

База даних складається з 4 таблиць: Usernames, Employees, States, Results.

Usernames – таблиця в якій зберігаються дані про ідентифікатор користувача в соціальній мережі.

Таблиця 2.3 – Опис полів таблиці Usernames

№	Назва	Тип	Обмеження на довжину	Тип ключа	Характеристика
1	Employees	Символьний	10	PK, FK	ID працівника
2	Username	Символьний	30		Ідентифікатор користувача в соціальній мережі (username)

Employees – таблиця в якій зберігаються всі дані про працівника.

Таблиця 2.4 – Опис полів таблиці Employees

№	Назва	Тип	Обмеження на довжину	Тип ключа	Характеристика
1	Id	Символьний	10	PK	ID працівника
2	Name	Символьний	30		ПІБ працівника

Продовження таблиці 2.4

№	Назва	Тип	Обмеження на довжину	Тип ключа	Характеристика
3	Sex	Символьний	1		Стать працівника. Може набувати значень “M”, “F”, “N”
4	Birth_Date	Дата			Дата народження працівника
5	Join_Date	Дата			Дата початку роботи
6	Position	Символьний	30		Посада
7	Email	Символьний	20		Корпоративна пошта працівника
8	Avatar	Блоб			Фото працівника
9	State_Id	Символьний	10	FK	Ід стану працівника

Results – таблиця в якій зберігаються результати сканувань всіх працівників.

Таблиця 2.5 – Опис полів таблиці Results

№	Назва	Тип	Обмеження на довжину	Тип ключа	Характеристика
1	Emplouees_Id	Символьний	10	PK, FK	ID працівника
2	Scan_Date	Дата		PK	Дата сканування
3	Percent_N	Дійсний			Відсоткове значення нормальності стану
4	Percent_S	Дійсний			Відсоткове значення стресовості стану
5	Percent_L	Дійсний			Відсоткове значення наявності ознак самотності
6	Count_N	Ціле			Кількість твітів класифікованих як нормальні
7	Count_S	Ціле			Кількість твітів класифікованих з ознаками стресу

Продовження таблиці 2.5

№	Назва	Тип	Обмеження на довжину	Тип ключа	Характеристика
8	Count_L	Ціле			Кількість твітів класифікованих з ознаками самотності
9	Status	Символьний	10		Текстова класифікація результату. Може набувати значення “Low”, “Moderate”, ”High”

States – таблиця в якій зберігаються всі можливі варіанти стану працівника.

Таблиця 2.6 – Опис полів таблиці States

№	Назва	Тип	Обмеження на довжину	Тип ключа	Характеристика
1	Id	Символьний	10	PK	ID стану
2	Note	Символьний	50		Текстова записка до стану
3	Status	Символьний	10		Текстове значення стану. Може набувати значення “Low”, “Moderate”, ”High”
4	Progress	Символьний	10		Прогресування стану. Може набувати значень “Improved”, “Stable”, “Worsened”

Зв’язки між сутностями:

Таблиця 2.7 – Зв’язки між сутностями

№	Сутності що утворюють зв’язок	Тип зв’язку	Пояснення
1	Employees – Usernames	Один-до-Одного	За кожним працівником закріплений один юзернейм (ідентифікатор у обраній платформі). Одному юзернейму може відповідати лише 1 співробітник.

Продовження таблиці 2.7

№	Сутності що утворюють зв’язок	Тип зв’язку	Пояснення
2	Employees – Results	Один-до-Багатьох	Співробітник може мати багато результатів, а може не мати жодного. Результат належить одному конкретному співробітнику.
3	State - Employees	Один-до-Багатьох	Кожному стану може відповідати декілька співробітників, або жоден. Співробітник може мати лише один стан.

2.5. Розробка архітектури програмного застосунку

З огляду на вимоги, що висуваються до створюваного застосунку, були виділені програмні компоненти та розроблена схема їх взаємодії.

На рисунку 2.7 представлена діаграма компонентів створюваного застосунку, що визначає архітектуру системи та описує залежності між програмними компонентами.

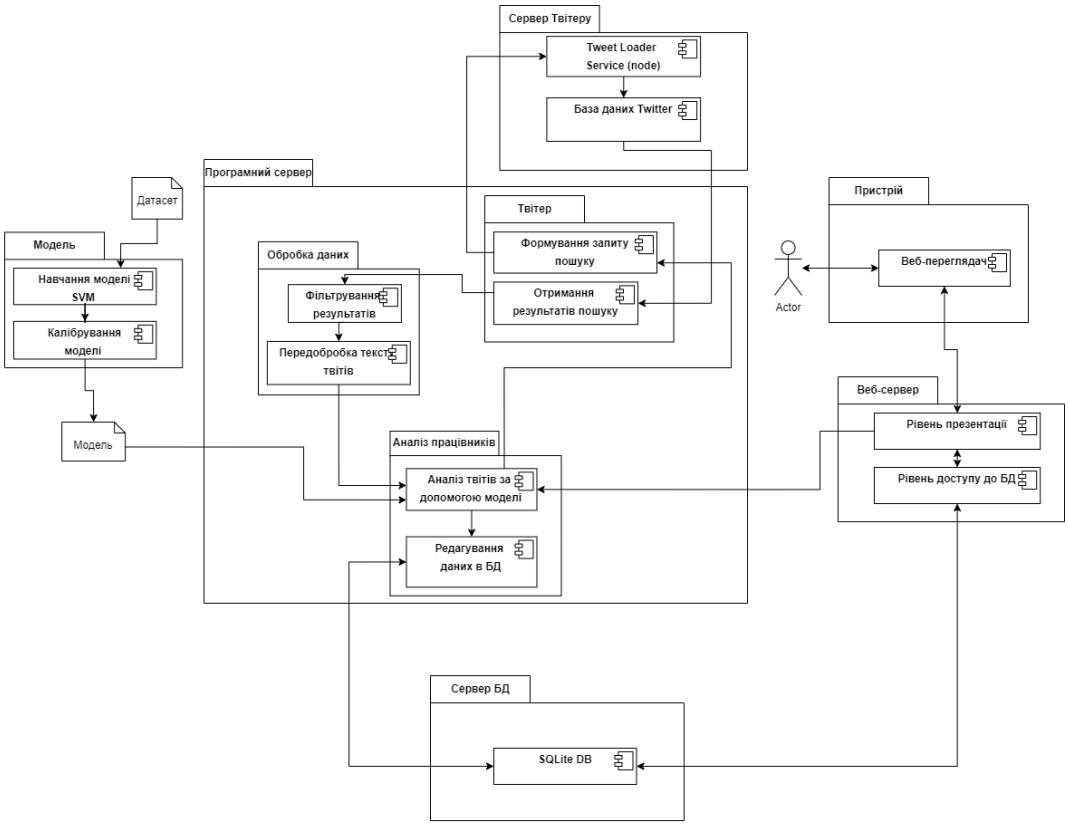

Рисунок 2.7 – Діаграма компонентів застосунку

Можна визначити 5 головних модулів у застосунку та 3 підмодулі.

Головні модулі:

Модель - даний модуль складається з навчання моделі SVM та її подальшого калібрування, на вхід подається дані з обраного датасету, а на виході ми маємо навчену модель, яка зберігається в файл.

Пристрій – модуль, який складається з веб-переглядача та відповідає за взаємодію з користувачем.

Веб-сервер – даний модуль складається з двох рівнів: рівню презентації (веб інтерфейс) та рівню доступу до бази даних. Рівень презентації також пов’язаний з підмодулем Аналіз працівників та передає в даний підмодуль дані про працівників та часовий проміжок, які необхідно проаналізувати.

Сервер БД – модуль, який містить в собі компонент бази даних SQLite.

Програмний сервер – модуль, який складається з трьох підмодулів, які необхідні для проведення аналізу текстів з соцмережі працівників.

Підмодулі:

Твітер - даний підмодуль описує сторонні компоненти, які спрацьовують при використанні Twitter API або бібліотеки python tweepy, яка була використана в даній роботі, та допомагають нам отримати дані про твіти користувача, на вхід подаються дані працівників, а саме їх юзернейми в соціальній мережі Twitter, на виході ми отримуємо дані про записи, які публікував користувач.

Модуль обробки даних, пов’язаний із попереднім модулем, адже отримує від нього дані про публікації користувача, також на вхід подається дата, для якої проводиться аналіз. В модулі знаходяться два компонента: фільтрування результатів, які залишають дані лише у проміжку від поданої дати мінус місяць до поданої дати та передобробка тексту твітів, в якій використовується техніки NLP для підготовки тексту до класифікації, а саме видалення стоп слів, пунктуації, зайвих пробілів, приведення букв до одного регістру та інші. На виході отримуємо спрощені тексти твітів для кожного з користувачів.

Аналіз працівників - модуль, який отримує на вхід дані про працівників, стан яких необхідно проаналізувати, та часові проміжки, передаючи ці дані в модуль Твітер. Окрім цього модуль використовує файл з навченою моделлю та передоброблені тексти твітів. За допомогою моделі проводиться аналіз твітів для кожного співробітника, зберігаються результати тестування в базу даних та вносяться зміни щодо поточного стану працівника.

2.6 Розробка дизайну веб-застосунку

Для розробки дизайну було використано платформу Figma.

На рисунку 2.8 можна побачити граф переходів між сторінками сайту.

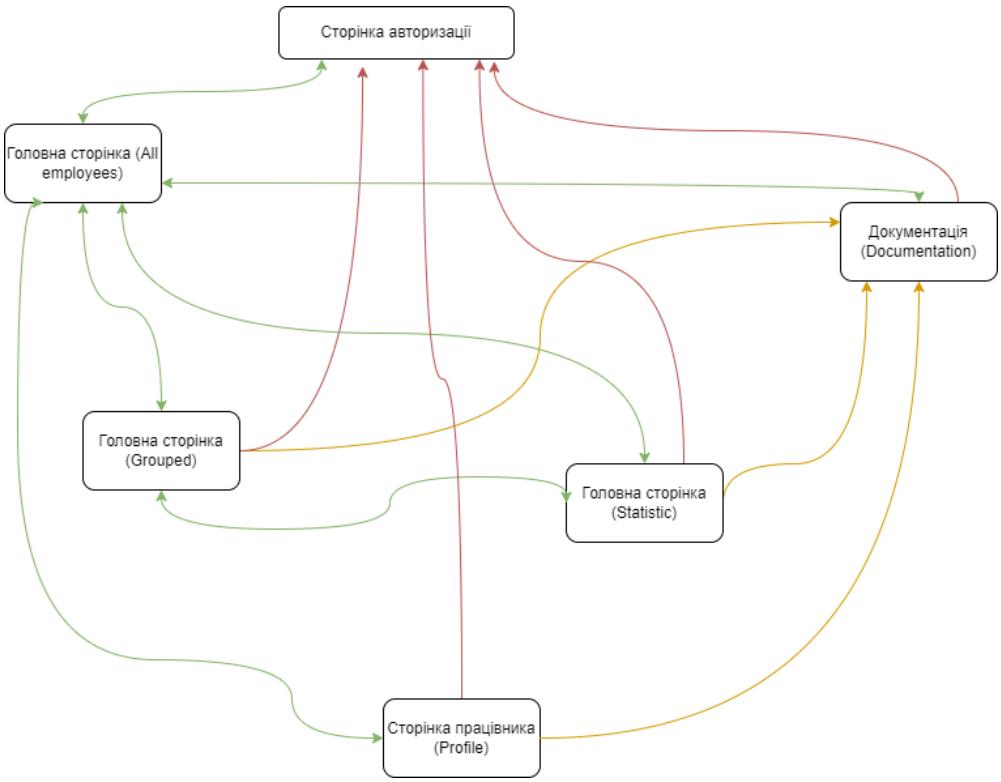

Рисунок 2.8 – Граф переходу між сторінками сайту

Сайт складається з шести основних сторінок. Так як застосунок працює з конференційною інформацією, користувач не може потрапити до основної частини сайту доки не зайде за допомогою свого аккаунту. На рисунку 2.9 Можна побачити сторінку авторизації. Реєструватися самостійно юзер не має

можливості, так як доступ до застосунку можуть мати лише люди, які слідкують за станом співробітників.

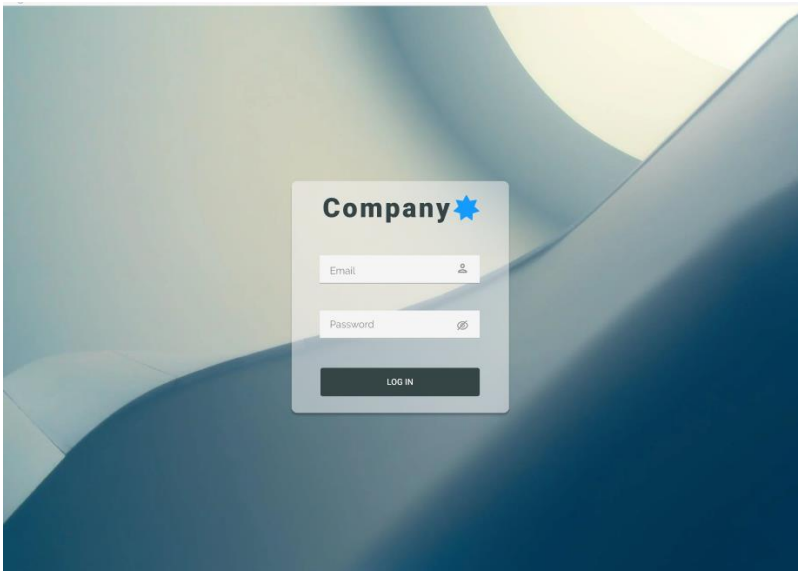

Рисунок 2.9 – Дизайн сторінки авторизації

Після успішної авторизації користувач перенаправляється на головну сторінку, де відображені результати сканування всіх користувачів рисунок 2.10. На екрані також знаходиться кнопка “Start Scanning”, яка відкриває форму в якій можна заповнити дату та почати сканування всіх співробітників рисунок 2.11.

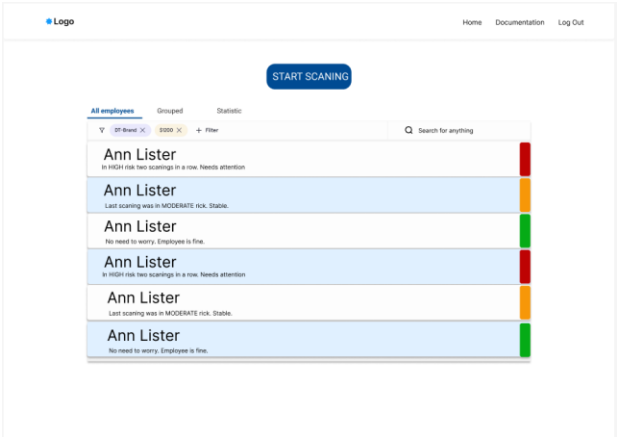

Рисунок 2.10 – Дизайн сторінки з усіма результатами

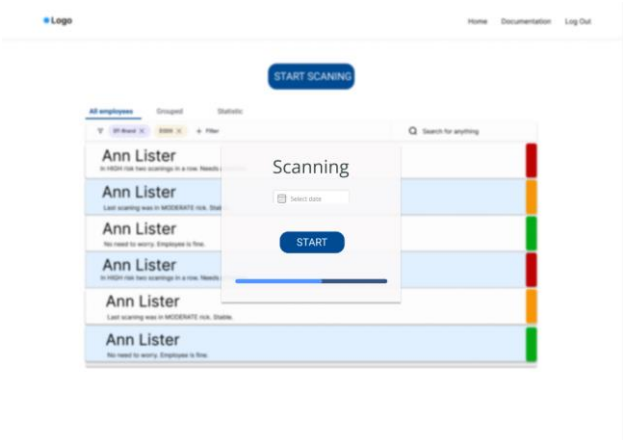

Рисунок 2.11 – Дизайн форми початку тестування

Згори сторінки знаходиться навігація. При натисканні кнопки Home відбувається перехід на головну сторінку рисунок 2.10, Documentation – перехід на сторінку з документацією рисунок 2.12, Log Out – вихід з акаунту та повернення на сторінку авторизації.

На сторінці документації відображаються правила, які допоможуть працівнику аналізувати дані, та вживати заходи у разі високого ризику працівників.

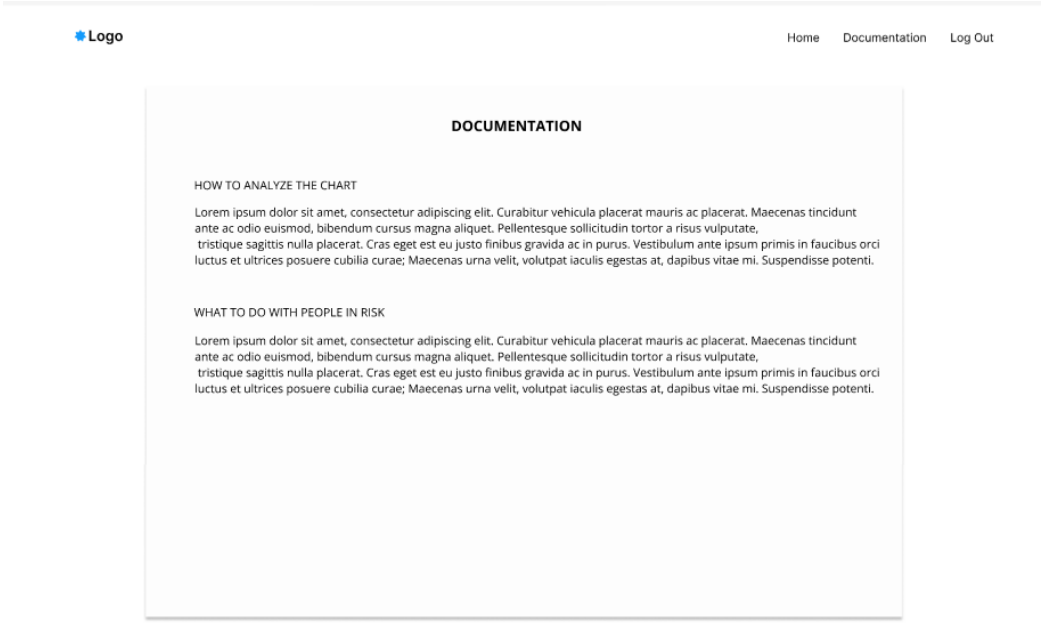

Рисунок 2.12 - Дизайн сторінки документації

Також на головній сторінці знаходяться вкладки “All employees” – відображає результати всіх працівників, “Grouped” відображає результати за групами (рис. 2.13), “Statistic” – візуально відображає статистику сканувань (рис 2.14).

Вкладка “Grouped” поділяється на 5 підвкладок, які показуються згруповані результати: “High” – працівники у високій зоні ризику, “Moderate” – працівники в середній зоні ризику, “Low” – працівники в нормальному стані, “Improved” – працівники стан яких покращився, “Worsened” – працівники стан яких погіршився.

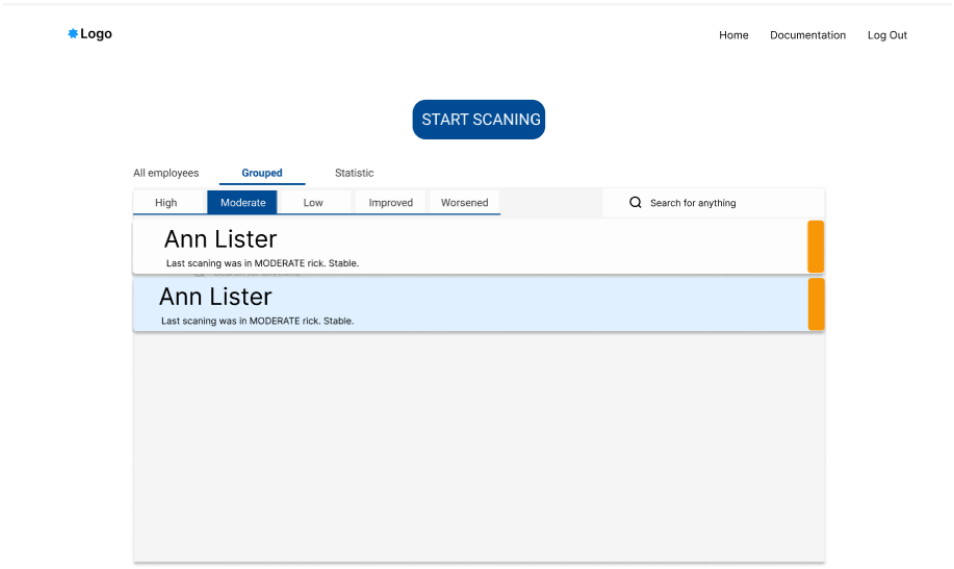

Рисунок 2.13 – Дизайн головної сторінки вкладки Grouped

На вкладці статистики результати сканувань відображаються у вигляді діаграм, є можливість відфільтрувати за датою, а також згрупувати за посадою, часом роботи і датою.

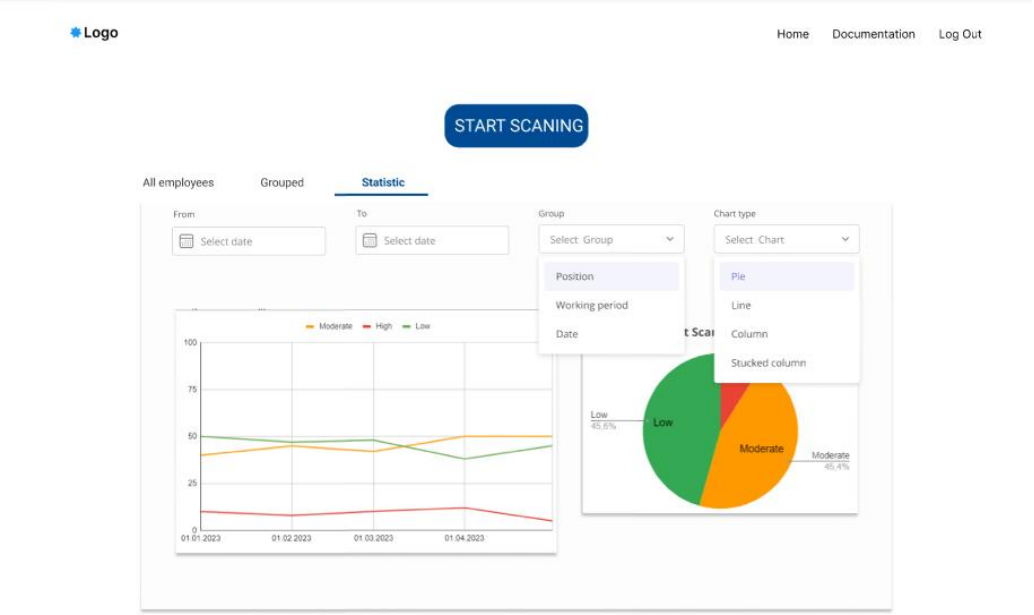

Рисунок 2.14 – Дизайн головної сторінки вкладки статистики

Для того щоб перейти у профіль працівника, необхідно натиснути на його результат у таблиці на головній сторінці. На профільній сторінці відображається загальна інформація про працівника, інформація про його статус а також графічне представлення результатів (рис 2.15).

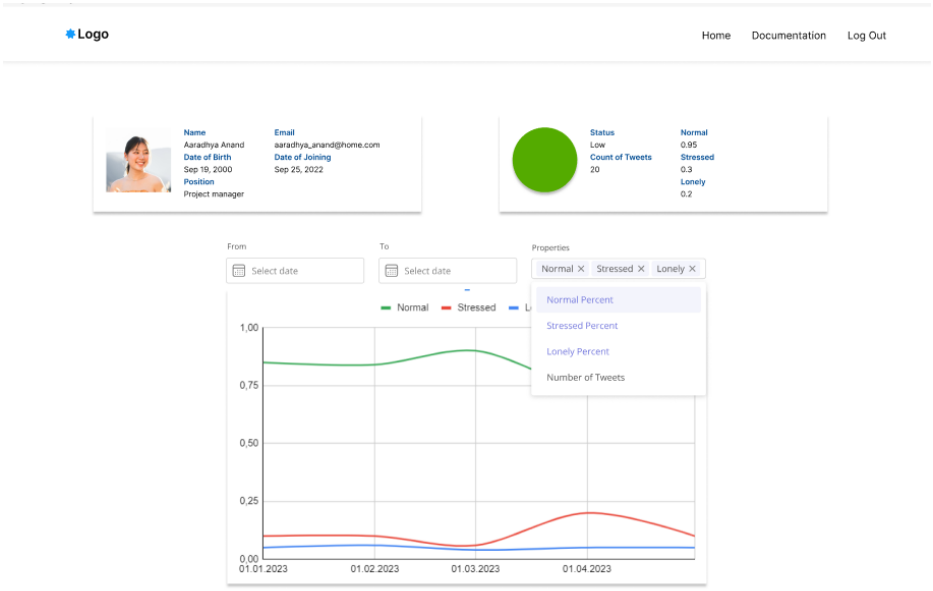

Рисунок 2.15 – Дизайн сторінки профілю працівника

Висновок до другого розділу

У ході проектування застосунку, було визначено, яка інформація необхідна для роботи підсистеми. Проаналізовано очікуваний вихідний формат результатів роботи моделі та сформовано методологію оцінки результатів роботи моделі аналізу текстів з соцмережі Twitter й визначення стану працівника на їх основі.

Для більш детального аналізу предметної області було визначено кордони предметної області та змодельовано контекстну діаграму IDEF0 ЯК БУДЕ з подальшою декомпозицією та описом бізнес процесів.

На основі визначених інформаційних даних, необхідних для роботи системи, було спроектовано базу даних застосунку та змодельовано фізичну модель даних.

Зважаючи на вимоги сформовані в першому розділі, були виділені програмні компоненти та розроблена схема їх взаємодії. Результати представлені за допомогою діаграми компонентів, що визначає архітектуру системи.

За допомогою платформи Figma, було розроблено дизайн сторінок сайту та сформовано логіку переходів між ними.

3 РЕАЛІЗАЦІЯ ПІДСИСТЕМИ ВИЯВЛЕННЯ ОЗНАК ВИСНАЖЕННЯ ПРАЦІВНИКІВ

3.1 Вибір інструментальних засобів для програмної реалізації застосунку

З 2003 року Python входить до десятки найпопулярніших мов програмування в індексі спільноти програмістів TIOBE[13]. Як можна побачити з рисунку 3.1, популярність Python стрімко зростає з 2018 року й станом на 2022 рік дана мова програмування посідає перше місце.

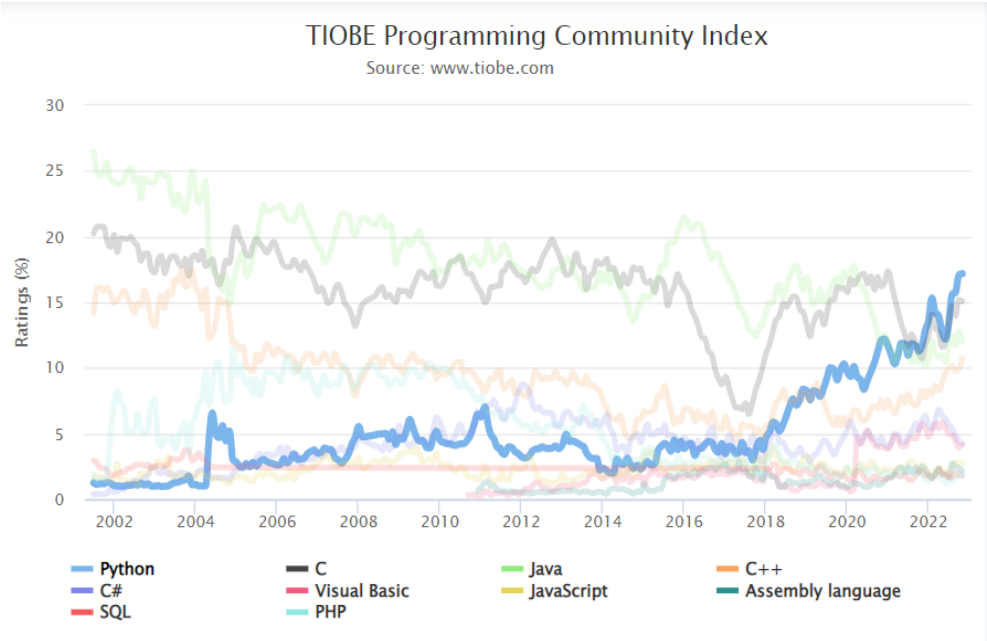

Рисунок 3.1 – Рейтинг мов програмування згідно індексу TIOBE з 2002 до 2022 років [13]

Пов’язати стрімкий ріст популярності мови можна з тим, що це основна мова, яка використовується для машинного навчання та розробки штучного інтелекту інтерес до яких також стрімко зростає. Одним з переваг мови Python є те що вона досить проста для вивчення, має читабельний, інтуїтивно зрозумілий код. Проте простота мови не є основним фактором, чому дана мова програмування була обрана для вирішення поставленої задачі.

Основною особливістю мови є великий вибір бібліотек і фреймворків. Python має багатий набір технологій, які використовуються для задач штучного інтелекту та машинного навчання.

Зважаючи на популярність Python, велику кількість фреймворків та бібліотек, спроектованих для машинного навчання, для вирішення поставленої задачі було обрано дану мову програмування.

Так як підсистема буде представлена у вигляді веб-застосунку, для його реалізації було обрано веб-фреймворк Django. Фреймворк підтримує бази даних, має вбудований інтерфейс адміністратора, розширювану систему шаблонів з тегами та наслідуваннями, бібліотеку для роботи з формами на основі існуючої моделі з бази даних та інші функції.

Для реалізації інтелектуального модуля застосунку було використано бібліотеку Scikit-learn. Вона містить різноманітні алгоритми класифікації, регресії та кластеризації, включаючи машини опорних векторів, випадкові ліси, посилення градієнта, k-середні та DBSCAN, і розроблено для роботи з числовими та науковими бібліотеками Python NumPy та SciPy. Для реалізації моделі були використані наступні функції:

1. tfidVectorizer – функція, яка дозволяє конвертувати необроблені дані в матрицю функцій TF-IDF (статистичний показник, що використовується для оцінки важливості слів у контексті документа, що є частиною колекції документів). Додатково дозволяє уніфікувати текст;
2. LinearSVC – модель класифікації методом лінійних опорних векторів. Подібний до SVC з параметром kernel='linear', але реалізований у термінах liblinear, а не libsvm, тому він має більшу гнучкість у виборі штрафних санкцій і функцій втрат і повинен краще масштабуватися для великої кількості зразків. Підтримка кількох класів обробляється за схемою «один проти решти»[14];
3. CalibratedClassifierCv – клас, який дозволяє відкалібрувати вже встановлені класифікатори.

Для видобування інформації про пости користувачів, було використано бібліотеку Tweepy. З її допомогою відбувається авторизація в акаунті розробника

Twitter, зчитування інформації про користувачів. Можна проводити різні операції зі сторінкою в Twitter: репостити, постити нові, видаляти твіти.

Також було використано брокер повідомлень Redis, для створення черги завдань, щоб розподілити процес аналізу всіх працівників між потоками.

3.2 Методи обробки тексту для ефективної роботи моделі

Так як в роботі були використані для аналізу тексти з Твітеру, одим із завдань є приведення тексту до уніфікованого формату, такого, який буде легко аналізувати моделі.

Виконання цієї задачі було зроблено в декілька етапів:

«Очищення» - на цьому етапі текст було приведено до нижнього регістру, Видалено всі посилання, емоджі, цифри, та знаки.

«Спрощення» - за допомогою корпусу англійських стоп слів було видалено всі незначущі слова з тексту.

Лематизація - на останньому етапі було проведено лематизацію слів й повернення їх до початкової форми за допомогою бібліотеки spacy.

Для прикладу розглянемо зміни в наступному реченні:

До обробки - «It's been an awful day ((((I just hate how I have to work 16 hours a day and then crying myself to sleep».

Очищення – «it s been an awful day i just hate how i have to work hours a day and then crying myself to sleep»

Спрощення – «awful day hate work hours day crying sleep»

Лематизація – «awful day hate work hour day cry sleep»

Можна побачити, що після етапів обробки текст значно скоротився й були залишені лише значущі слова, які далі будуть проаналізовані моделлю.

У таблиці 3.1 та на рисунку 3.2 можна побачити, як змінюється відповідь моделі в залежності від якості обробки тексту, який подається для аналізу. Так як в даному тексті немає багато символів та посилань вплив першого етапу

обробки тексту не впливає на результат, проте можна побачити, що етап лематизації є найважливішим з усіх трьох, так як саме після нього результат наданий моделлю значно покращується.

Таблиця 3.1 – якість роботи моделі в залежності від обробки тексту

Параметр	До обробки	Очищення	Спрощення	Лематизація
Нормальність	69	69	75	26
Стрес	28	28	21	43
Самотність	2	2	4	31

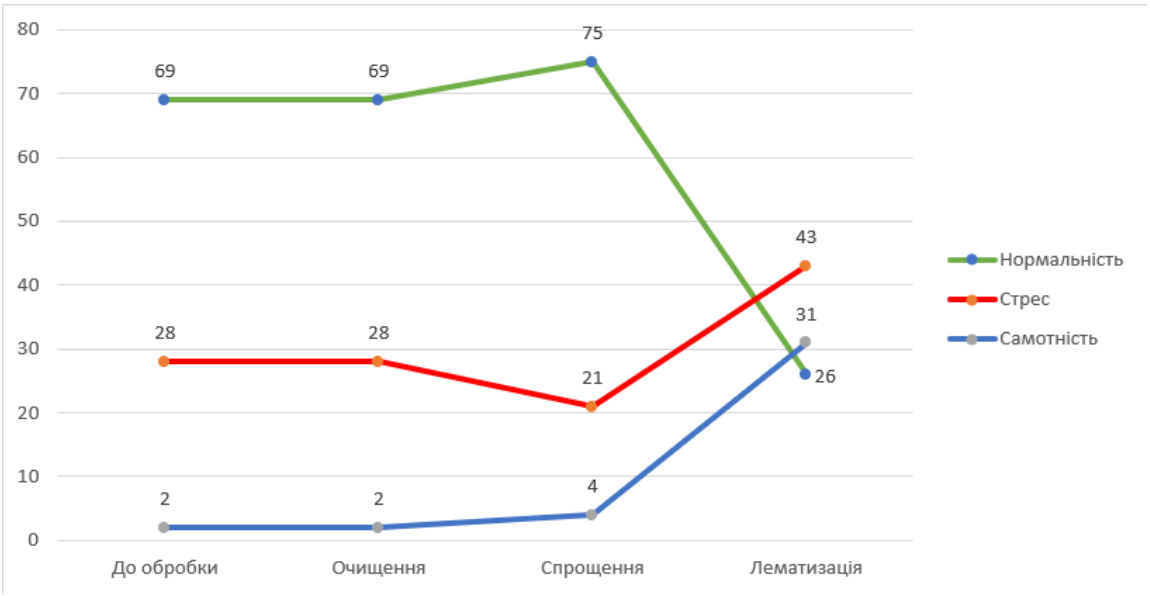

Рисунок 3.2 – Зміна результатів роботи моделі в залежності від етапу обробки тексту

3.3 План проведення експериментальних досліджень SVM

Для аналізу текстів твітів на ознаки виснаження, була обрана модель CalibratedClassifier на базі LinerSVC (SVM метод з лінійним ядром). CalibratedClassifier дозволяє отримувати ймовірностні значення належності екземпляру до класу, так як модель LinerSVC не має такого функціоналу. Принциповою умовою, зазначеною в документації, є те що дані для навчання

моделі класифікації та калібрування мають бути різними, відповідно до цього правила існує дві стратегії використання поєднання цих моделей:

Варіант 1:

1. Тренування моделі LinerSVC на даних “A”;
2. Створення калібратора на основі тренованої моделі з прапором cv = “prefit”, що означає, що модель вже тренована;
3. Використання функції fit на інших даних для відкаліброваної моделі.

Варіант 2:

1. Створюємо модель LinerSVC проте не проводимо навчання;
2. Створюємо калібратор й визначаємо значення cv – тип крос-валідації;
3. Навчаємо та калібруємо модель створену на кроці два на одних даних.

Було проведено дослідження наступних параметрів моделі LinerSVC:

Штраф (penalty) – визначає норму, яка використовується для штрафування рішення. Існує два варіанти - L1 та L2. L2 додає «квадратичну величину» коефіцієнта як штрафний термін до функції втрат. L1 - додає «абсолютне значення величини» коефіцієнта як штрафний термін до функції втрат. Значення в розробленій моделі – L2.

Class_weight – надає можливість встановити ваги класів. Якщо обраний збалансований режим, то коригування здійснюється обернено пропорційно до частот класу вхідних даних. Значення в розробленій моделі – збалансований режим.

C - Параметр регуляризації. Параметр визначає скільки неправильної класифікації ми можемо допустити. Жорсткий запас SVM зазвичай має високі значення C. М’який запас SVM зазвичай має невеликі значення C. Жорсткий запас SVM не допускає будь-якої неправильної класифікації. SVM з м’яким запасом допускає деяку неправильну класифікацію, послаблюючи жорсткі обмеження.

У ході розробки було проведено дослідження наступних параметрів:

1. Залежність результату від типу поєднання моделі на калібрування;
2. Залежність результату від значення встановлених вагів класів;
3. Залежність результату від типу штрафування;
4. Залежність результату від параметру регуляризації.

В таблиці 3.2 представлені дані в залежності від поєднання моделі та калібратора. На тренувальних даних варіант 1 показав кращі результати, проте на тестових даних гірші.

Таблиця 3.2 – оцінка точності моделі в залежності від поєднання моделі й калібратора

Вид поєднання	Точність на тренувальних даних	Точність на тестових даних
Варіант 1	0.988	0.94
Варіант 2	0.987	0.93

З таблиці 3.3 можемо побачити, що обидва варіанти значень вагів показали ідентичні результати, з чого можна зробити висновки, що навчальні дані, які подаються в модель, є збалансованими.

Таблиця 3.3 – оцінка точності моделі в залежності від значення вагів

Значення вагів	Точність на тренувальних даних	Точність на тестових даних
balanced	0.987	0.93
1	0.987	0.93

З таблиці 3.4 можемо побачити, що штраф типу L1, надав гірші результати як на тренувальних даних так і на тестових даних. Можна припустити, що відмінності викликані різними підходами цих функцій: L1 – дає вихідні дані у двійкових вагах від 0 до 1, та зазвичай використовується для зменшення кількості

ознак, а L2 розсіює умови помилок у всіх вагових коефіцієнтах, що приводить до більш точних налаштованих остаточних моделей.

Таблиця 3.4 – оцінка точності моделі в залежності від параметру штрафу

Значення штрафу	Точність на тренувальних даних	Точність на тестових даних
L1	0.966	0.92
L2	0.987	0.93

З таблиці 3.5 можемо оцінити вплив параметру регуляризації на точність навчання моделі. На тренувальних даний найкращих результатів було досягнуто при значення параметру 1.5, проте на тестових даних дане значення не досягло найкращих результатів.

Таблиця 3.5 – оцінка точності моделі в залежності від параметру C

Значення C	Точність на тренувальних даних	Точність на тестових даних
0.3	0.967	0.91
0.5	0.977	0.93
1.0	0.987	0.93
1.3	0.989	0.92
1.5	0.990	0.92

Графічно результати експериментів над параметром регуляризації показано на рисунку 3.3. Можна побачити, що оптимальним значенням C є значення 1, точність на тестовій вибірці далі зростає незначно, але при цьому падає точність тестової вибірки.

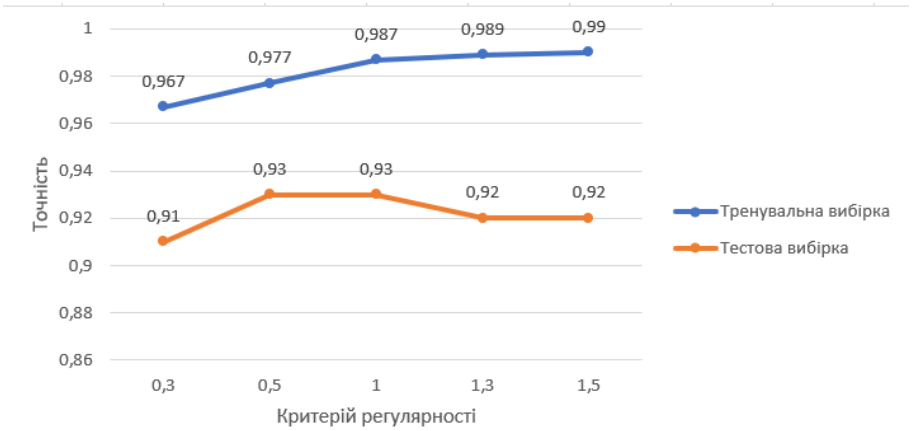

Рисунок 3.3 – Графік зміни точності в залежності від значення параметру C

3.3 Протоколи навчання та оцінки точності роботи реалізованих алгоритмів в процесі навчання

Важливу роль при розробці та навчанні моделі відіграють дані, які використовуються для навчання моделі. На рисунку 3.4 зображена діаграма, яка відповідає кількісному представленню даних за класами.

З діаграми можна побачити, що дані є досить збалансованими, а також, що кількість даних для навчання по кожному класу перевищує вісім тисяч екземплярів.

Розділення даних на тренувальну та тестову вибірки здійснюється за допомогою бібліотеки Scikit-learn у відношенні 8/2.

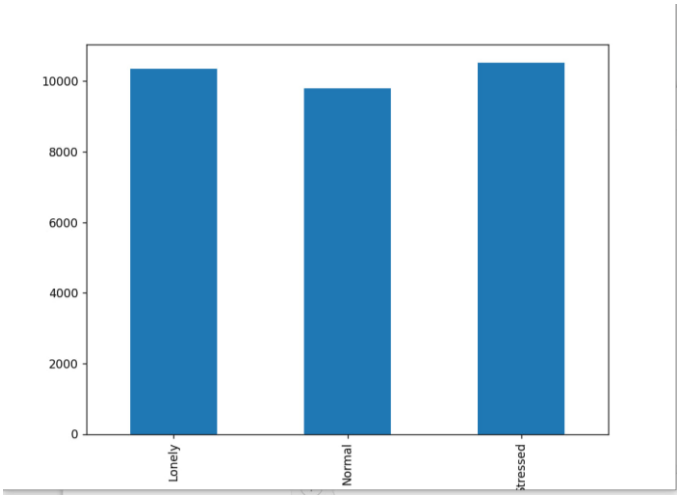

Рисунок 3.4 – Діаграма, яка відображає кількість даних у кожному класі

Втрата (loss) – у методі SVM, використовують функцію шарнірних витрат, або квадрату шарнірних витрат.

Таблиця 3.6 – значення параметрів моделі

Параметр	Значення
Поєднання класифікатора та калібратора	Варіант 2
Штраф	L2
Помилка	Квадрат шарнірних витрат
Точність	0.0001
C	1
Стратегія мультикласової класифікації	Один проти решти
Значення вагів	збалансовані

Результати навчання моделі, оцінені на тестовій вибірці, відображені у вигляді матриці невідповідності (таблиці особливого компонування, що дає можливість наочно оцінити продуктивність алгоритму) на рисунку 3.5.

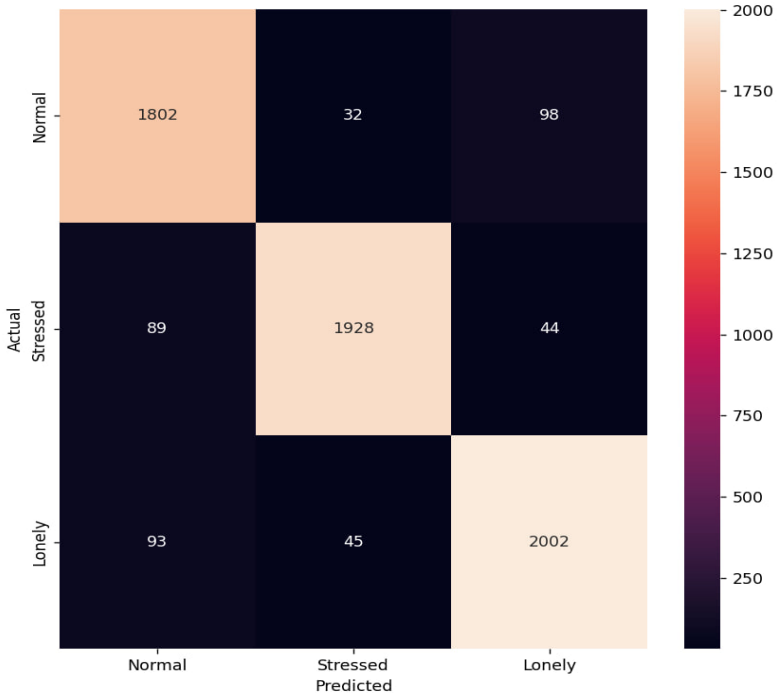

Рисунок 3.5 – Матриця невідповідності для тестової вибірки

Можна побачити, що модель чітко навчилася класифікувати текст відповідно до необхідного класу. Кількість екземплярів, які були класифіковані неправильно, є невеликою в порівнянні з загальною кількістю даних.

Звіт з відображенням основних показників класифікації можна побачити на рисунку 3.6. Значення досягнутої точності на навчальній вибірці дорівнює 98.7%.

```
Model calibrated accurace score:  0.9877695234643957
Time =  181.8236 seconds
              precision    recall  f1-score   support

      Normal       0.91      0.93      0.92      1932
      Lonely       0.96      0.94      0.95      2061
    Stressed       0.93      0.94      0.93      2140

    accuracy                           0.93      6133
   macro avg       0.93      0.93      0.93      6133
weighted avg       0.94      0.93      0.93      6133
```

Рисунок 3.6 – Звіт основних показників класифікації

Результати роботи моделі на тестових даних поділяються за наступними показниками:

Точність (precision) – говорить про те, наскільки точна модель. Скільки серед прогнозованих позитивних результатів, є фактично позитивних.

Повнота (recall) – коефіцієнт істинних позитивних результатів) обчислюється шляхом ділення істинних позитивних результатів на все, що слід було передбачити як позитивне.

F-міра – підсумовує прогностичну продуктивність моделі шляхом поєднання двох інших конкуруючих показників — точності(precision) та повноти(recall).

Показники розраховуються для кожного класу окремо, а також середні значення: макросереднє (усереднення незваженого середнього значення на мітку), зважене середнє (усереднення зваженого середнього значення на мітку) та середнє значення вибірки.

З отриманих значень бачимо, що найкраще модель навчилася класифікувати клас твітів, які характеризують самотність з F-мірою 96%.

Найгірше було класифіковано нормальні твіти, при чому для даного класу більш характерно помилятися в правдивості позитивного прогнозу.

Середня точність на тестовій вибірці серед усіх класів складає 93%, що є прийнятною точністю для даної роботи.

3.4 Опис та аналіз результатів тестових прикладів роботи інтелектуального застосунку

Для того, щоб краще оцінити, як працює модель для оцінки текстів, було зроблено вибірку з твітів та проведено їх аналіз (таблиця 3.7). Можна побачити, що всі твіти були класифіковані правильно в залежності на їх контекст.

Таблиця 3.7 – тестові речення та їх класифікація моделлю

№	Текст	Нормальність	Стрес	Самотність
1	A failed talking stage is not the same as a failed relationship. Of course it hurts on the party who was expecting it to work, but at this stage we were figuring it out and it's okay when we find that we aren't actually what we are looking for each other to amicably go out	0.59	0.15	0.26
2	I feel like you're either a dog with cat like behaviour or vice versa	0.95	0.02	0.02
3	I am worried about tomorrow	0.19	0.81	0.00
4	Why did I begin a 4500 word assignment 24 hours before the deadline? When am I going to learn I'm not conditioned to create my best work under stress, I'm just conditioned to gaslight myself to think I do	0.13	0.87	0.00
5	me??? tired??? stressed??? exhausted??? i wanna cry??? yes	0.00	1.00	0.00
6	one of the hardest truths I’ve come to terms with is how lonely you get when you are in uni, especially if you live faraway from your family, I dont even have anyone to talk to or ask how my day is going, luckily I have friends but they aren’t that close. I just need to adapt	0.00	0.08	0.82

7	to feeling alone to finding a group/community at the end that you feel safe and accepted with, so much so that you don't want to go home.... i'll cry.	0.00	0.00	0.99

Розглянемо перший приклад (рис 3.7), в тексті говориться «Невдала стадія зустрічей – це не те саме, що невдалі стосунки. Звичайно, це шкодить стороні, яка очікувала, що це спрацює, але на цьому етапі ми це з'ясовували, і це нормально, коли ми виявляємо, що насправді ми не ті, кого шукаємо одне для одного, щоб зустрічатися». Цей твіт не несе сильного стресового забарвлення чи показників самотності, проте з контексту їх все ще можна уловити. Як ми бачимо на рисунку 3.6 система класифікувала даний твіт на 59% як нормальний, 15% – стресовий та 26% - самотній, що загалом відповідає змісту твіту.

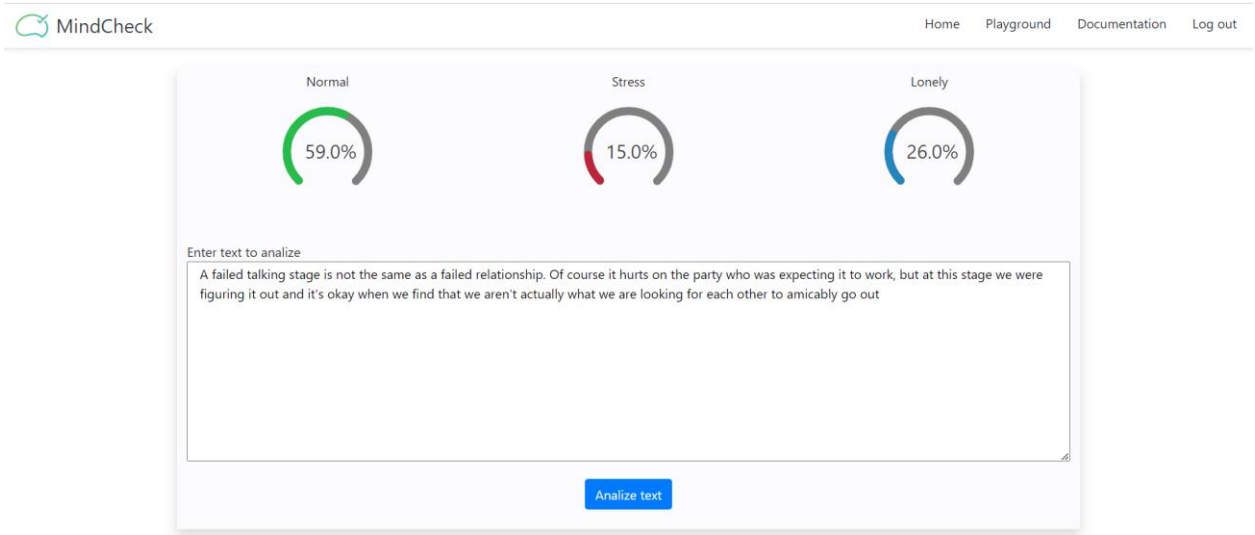

Рисунок 3.7 – Результати аналізу тестового прикладу 1

Розглядаючи переклад тестового прикладу 4 («Чому я розпочав завдання з 4500 слів за 24 години до кінцевого терміну? Коли я зрозумію, що я не створюю свої найкращі роботи під час стресу, я просто змушений зраджувати себе, думаючи, що я це роблю»), зрозуміло, що людина переживає певний рівень стресу через навчання та дедлайни. Система класифікувала даний твіт як стресовий, що відповідає його змісту (рис 3.8).

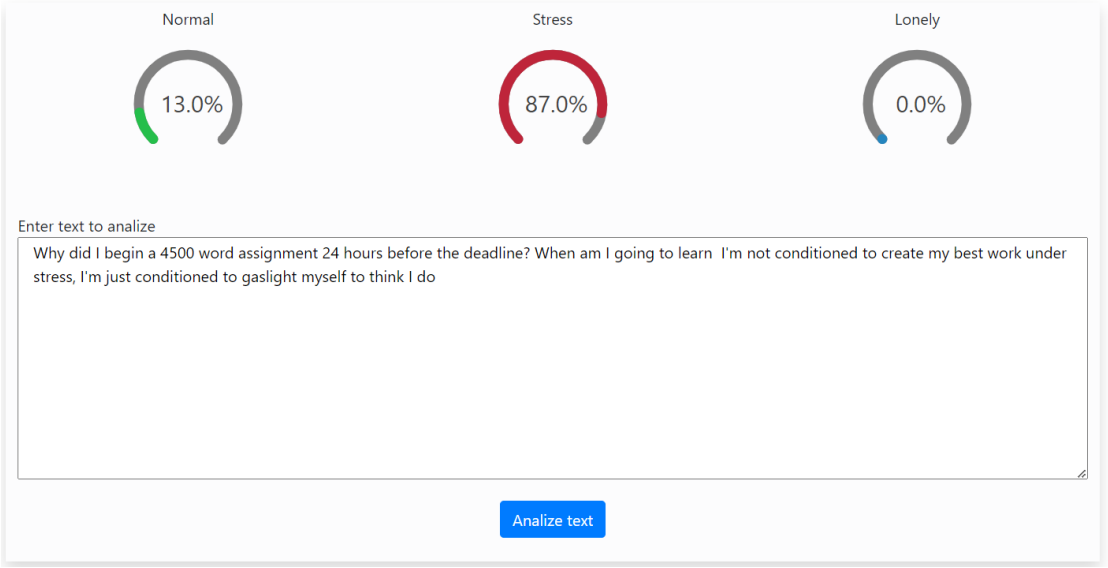

Рисунок 3.8 – Результати аналізу тестового прикладу 4

3.5 Опис та аналіз результатів тестових прикладів роботи застосунку

Фреймворк Django автоматично створює адмін панель з якої можливо редагувати дані в базі даних. Окрім описаних у 2 розділі таблиць, які необхідні для роботи підсистеми, також були створені таблиці «Акаунт» - в якій зберігаються дані про користувачів, які мають доступ до системи (рис. 3.9), а також група таблиць “Celery results”, які використовуються брокером повідомлень при виконанні завдань з черги.

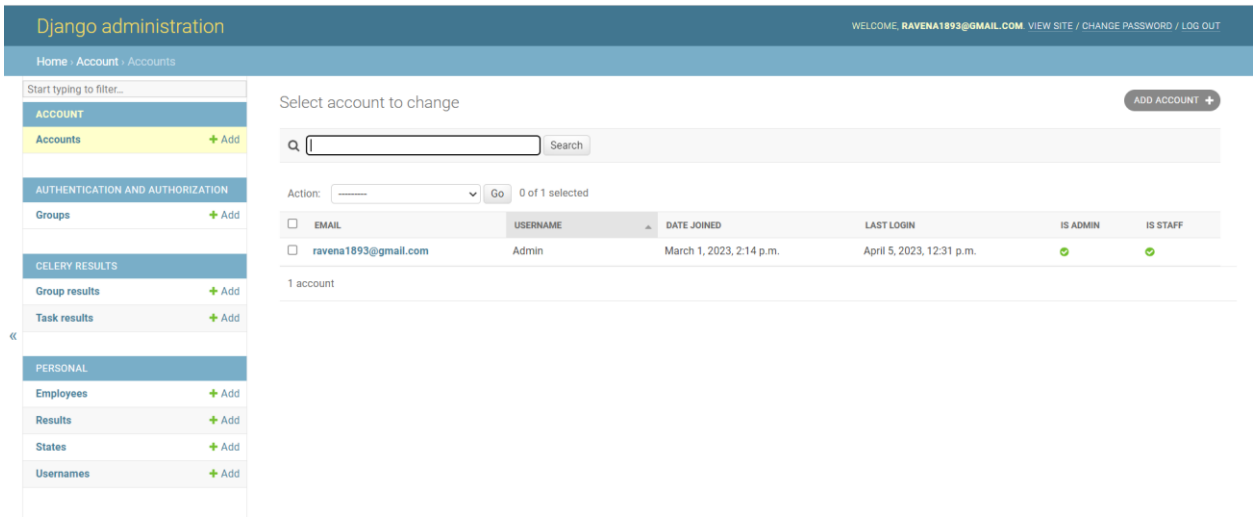

Рисунок 3.9 – Таблиця акаунтів, які мають доступ до системи на адмін панелі

Для того щоб отримати доступ до системи, користувач має бути занесений до таблиці, працівником, який має права надавати дане право, самостійно зареєструватися користувач не має можливості в цілях безпеки. При будь-якій спробі перейти за посиланнями системи неавторизований працівник буде направлений на сторінку входу (рис. 3.10).

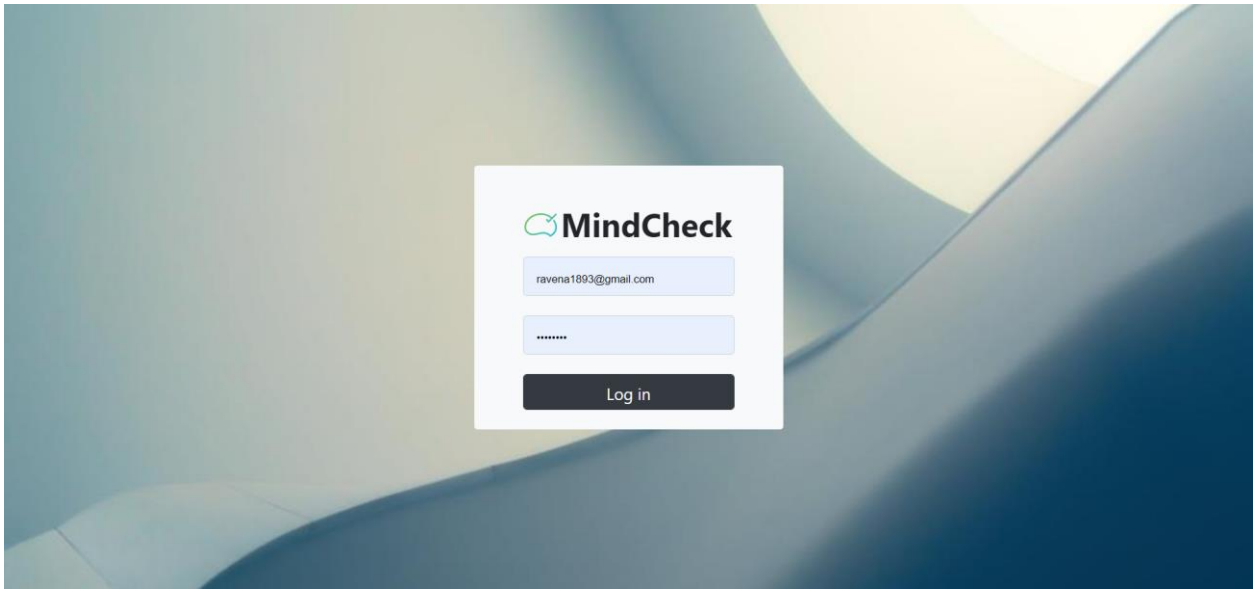

Рисунок 3.10 – Сторінка входу в систему

Для кращого розуміння як працювати з системою, було створено сторінку документації (рис. 3.11) на якій зберігаються відповіді на найпоширеніші запити, при натисканні на запит відкривається детальна інструкція (рис. 3.12).

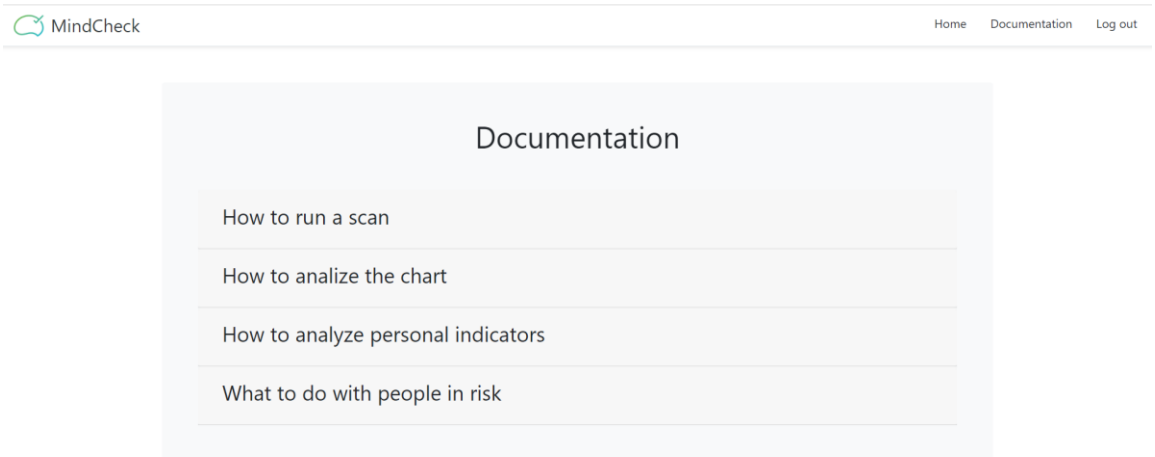

Рисунок 3.11 – Сторінка документації

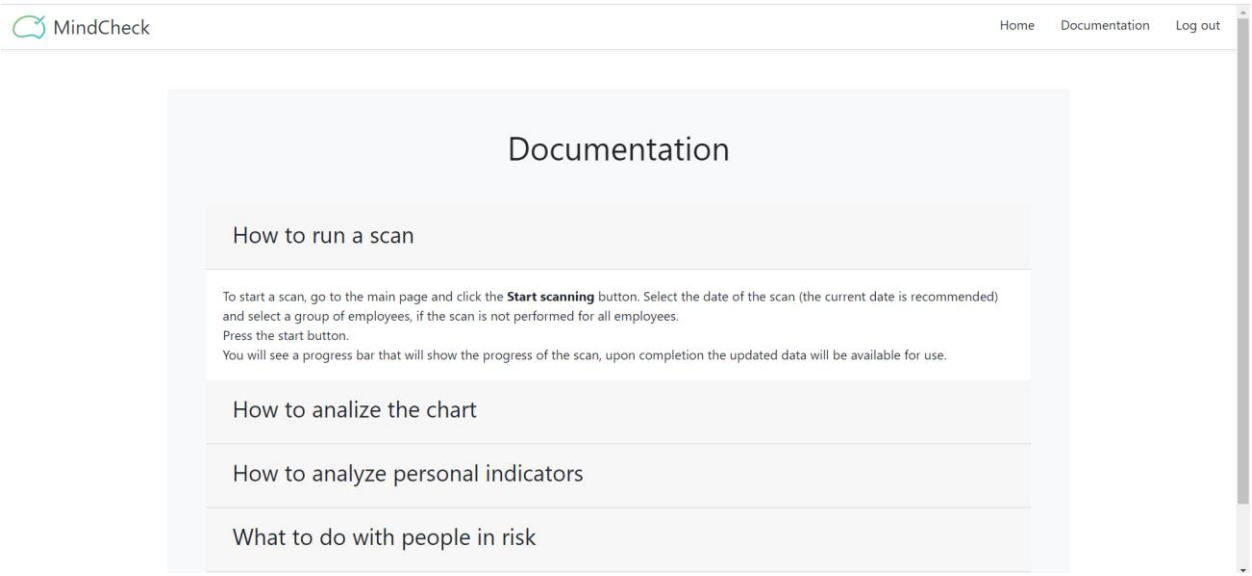

Рисунок 3.12 – Сторінка документації з розгорнутою відповіддю

Основним інструментом підсистеми є можливість сканування працівників, для цього необхідно натиснути на кнопку «Почати сканування». Відкриється вікно, де можна обрати необхідну дату сканування а також посаду працівників, які потребуються сканування (рис 3.13).

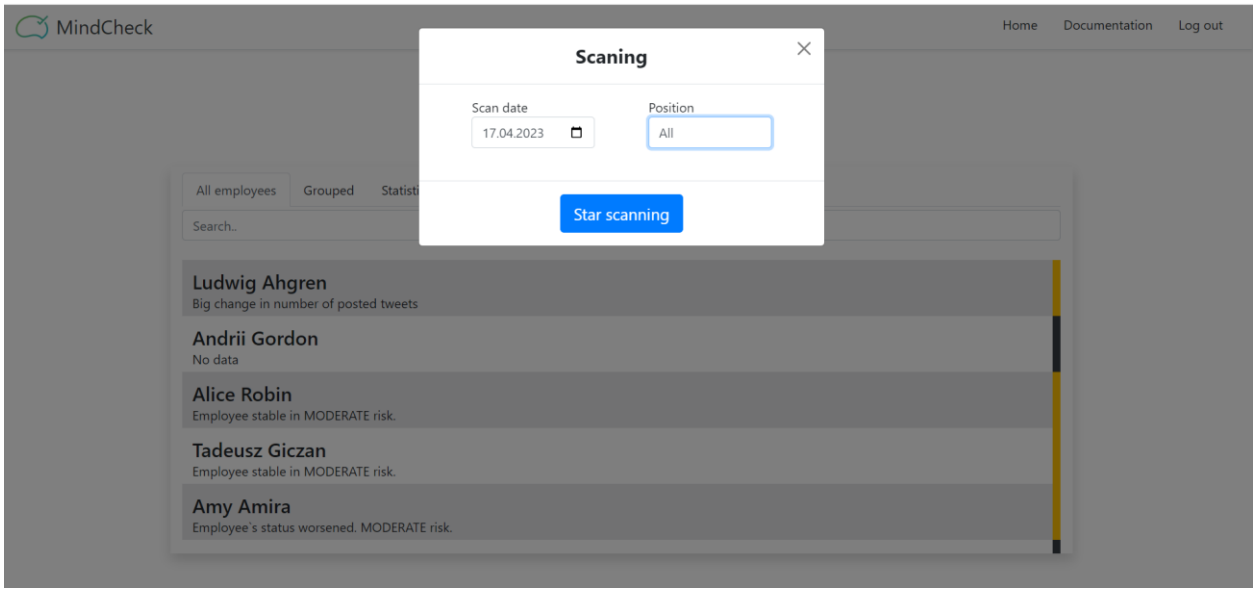

Рисунок 3.13 – Модульне вікно сканування

Після натискання кнопки старт, запускається сканування та аналіз текстів твітів працівників, блокується кнопка «Почати сканування», а також з’являється індикатор виконання, який показує прогрес сканування (рис 3.14).

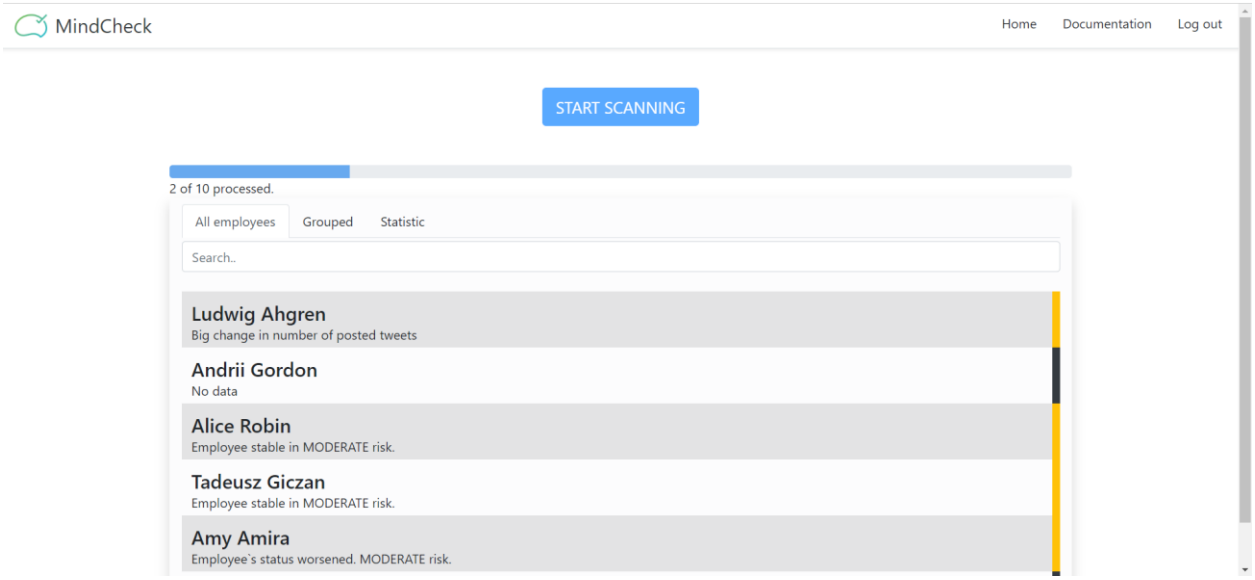

Рисунок 3.14 – Головна сторінка застосунку в процесі виконання сканування

Під час роботи сканування, користувач може вільно переміщатись між вкладками та сторінками, при цьому індикатор виконання буде сигналізувати про виконання завдання (рис. 3.15). Після завершення сканування індикатор прогресу покаже успішне завершення (рис. 3.16) і при наступному переході на іншу сторінку – зникне.

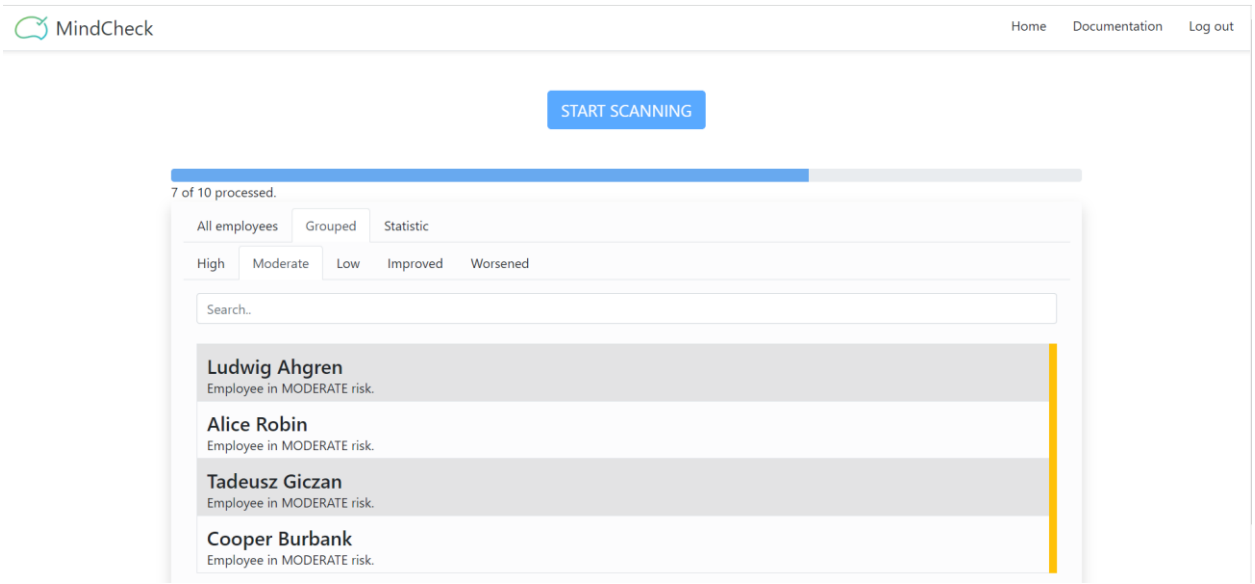

Рисунок 3.15 – Сторінка згрупованих результатів в процесі виконання сканування

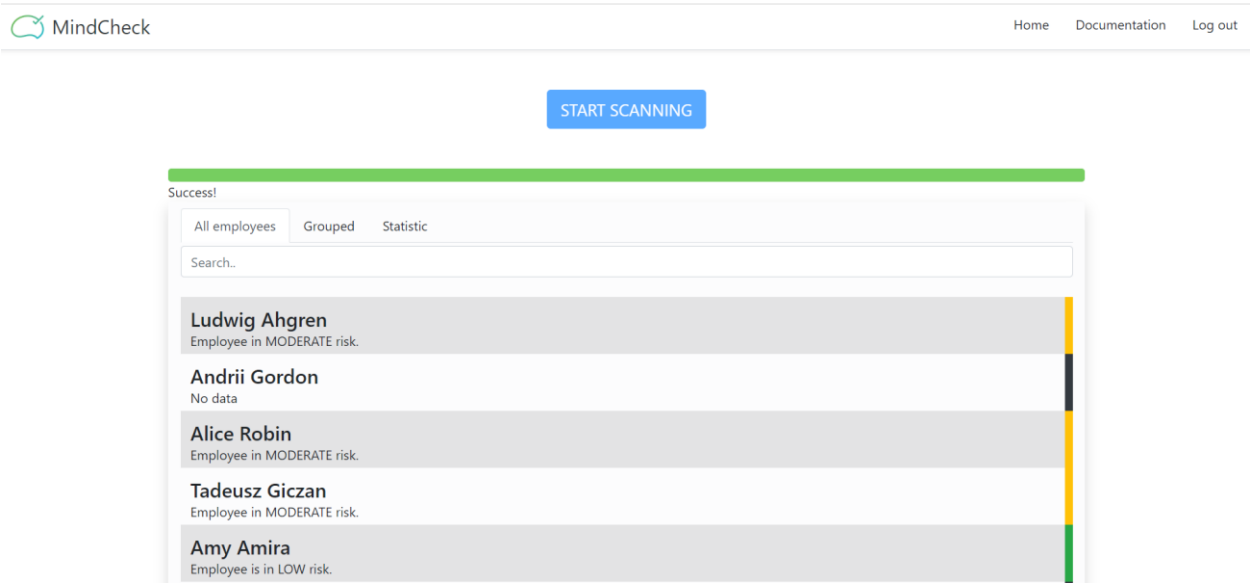

Рисунок 3.16 – Головна сторінка застосунку після успішного виконання сканування

Окрім головної сторінки, де показані всі працівники, існує можливість подивитись працівників згруповано за результатами. На рисунку 3.17 зображено працівників стан яких покращився в порівнянні з попереднім скануванням.

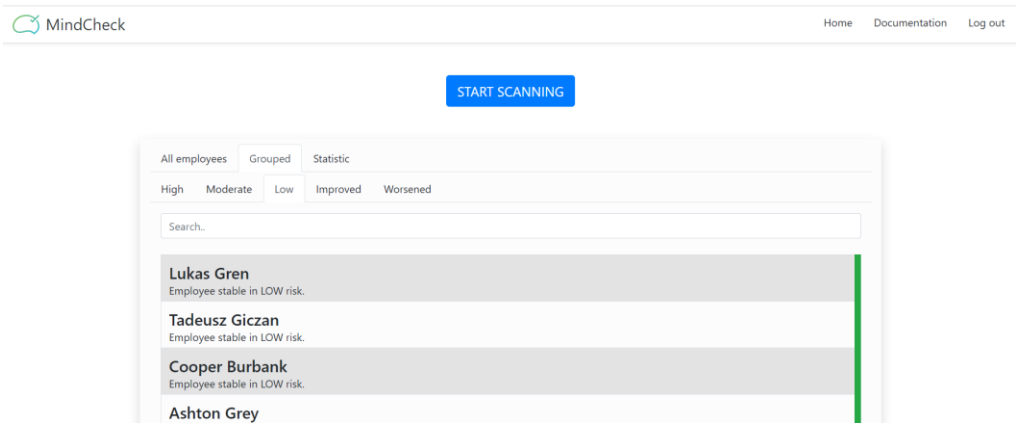

Рисунок 3.17 – Сторінка відображення групи працівників стан яких покращився

Важливим інструментом при роботі із даними, є можливість наглядно оцінити тенденції загалом і за категоріями. У розділі статистики працівник може керувати такими параметрами як початкова дата, кінцева дата, фактор групування (дата, посада, кількість років роботи в компанії) та вид представлення (кругова діаграма, лінійна діаграма, стовпчаста діаграма та стовпчаста діаграма з накопиченням).

На рисунку 3.18 представлений варіант лінійної діаграми згрупованою за датою сканування, де червоним позначений графік кількості працівників в високій зоні ризику, помаранчевим – в середній, зеленим – в низькій, а сірим – працівники дані яких відсутні. На рисунку 3.19 представлена стовпчаста діаграма з накопиченням за аналогічними даними.

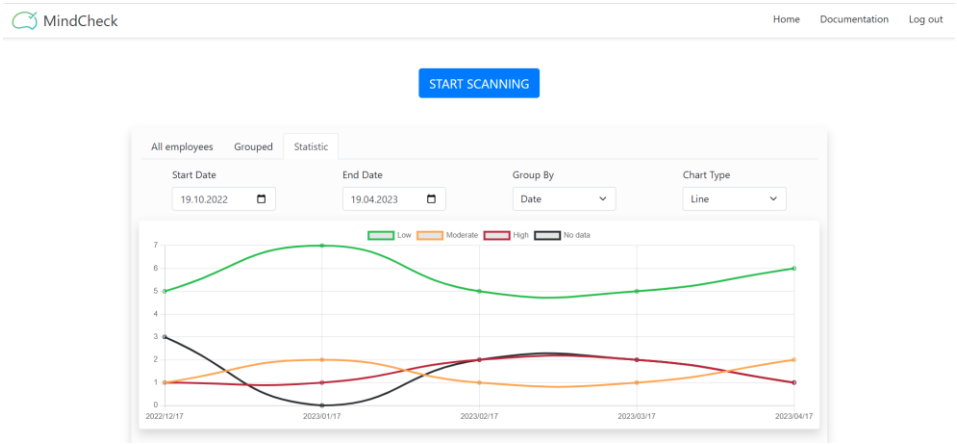

Рисунок 3.18 – Статистичні дані згруповані за датою сканування представлені лінійною діаграмою

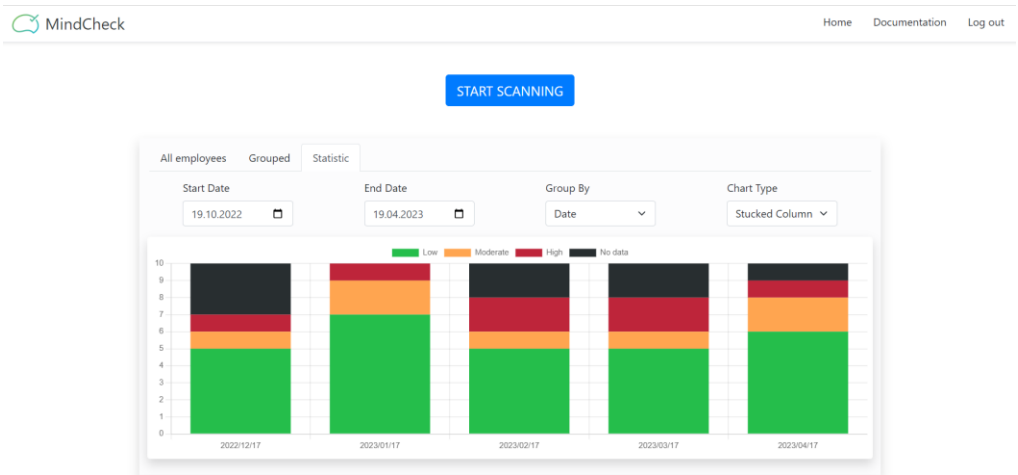

Рисунок 3.19 – Статистичні дані згруповані за датою сканування представлені стовпчастою діаграмою з накопиченням

На рисунку 3.20 представлений варіант кругової діаграми згрупованою за посадою працівників, кожна кругова діаграма відповідає певному рівню ризику і показує розподілення людей за посадою в даній групі.

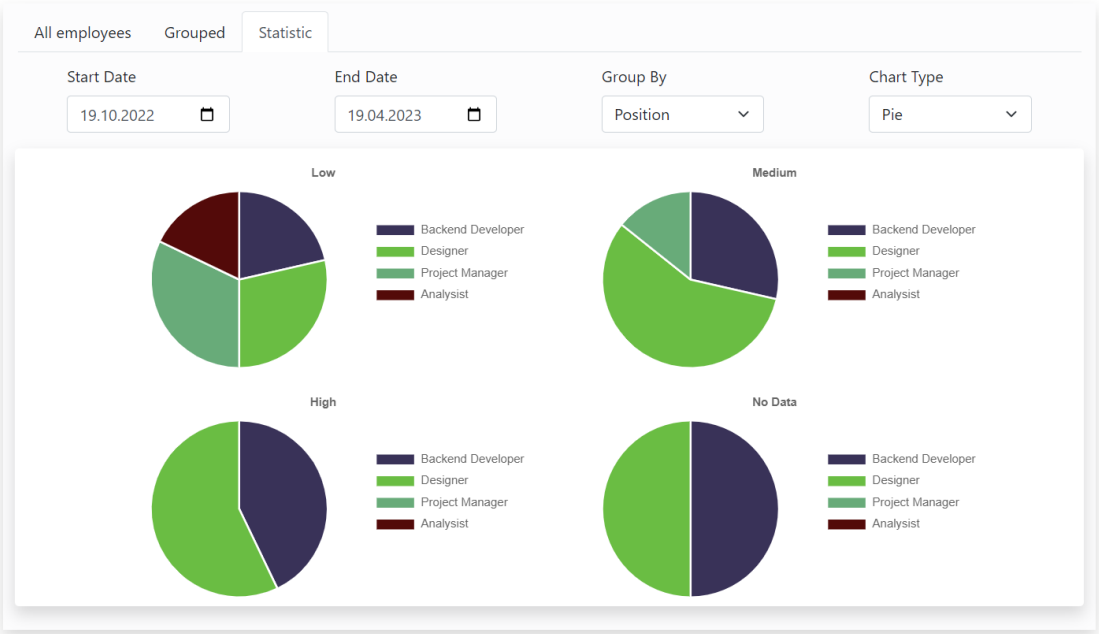

Рисунок 3.20 – Статистичні дані згруповані за посадою працівників представлені круговими діаграмами

На рисунку 3.21 представлена стовпчаста діаграма за аналогічними даними за останнє сканування, що визначено початковою та кінцевою датою сканувань.

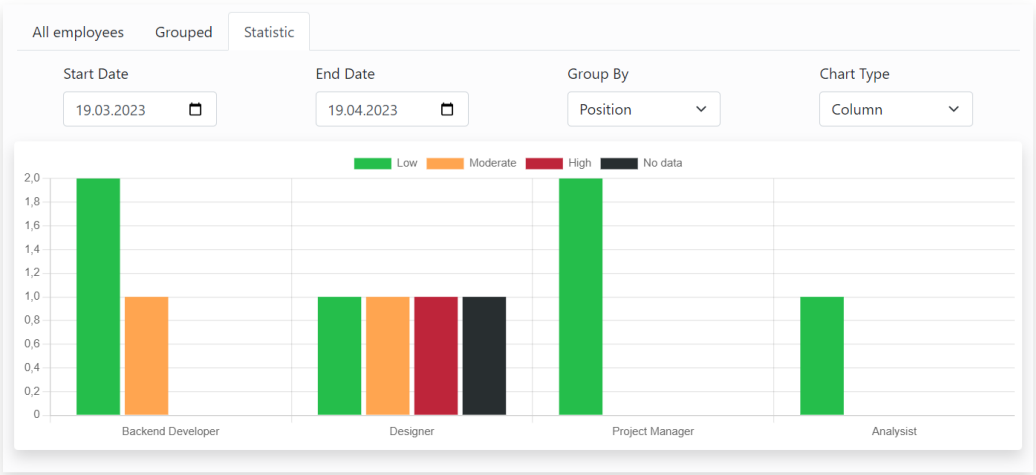

Рисунок 3.21 – Статистичні дані згруповані за посадою працівників представлені стовпчастою діаграмою

На лінійній діаграмі для даних згрупованих за посадою (рис 3.22) є можливість фільтрувати зображувані дані за допомогою випадаючих списків за посадою для більш детального аналізу.

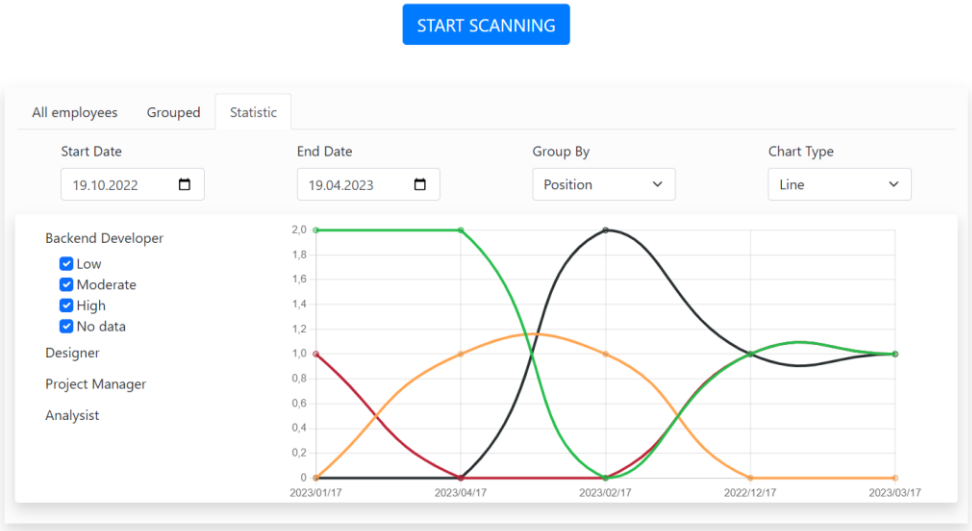

Рисунок 3.22 – Статистичні дані згруповані за посадою працівників представлені лінійною діаграмою

На рисунку 3.23 представлені дані згруповані за часом, який працівник працює в компанії. Дані представлені круговими діаграми й складаються з кількості працівників, які належать до певної категорії, а також в дужках відсоткове значення від усіх працівників в даній категорії.

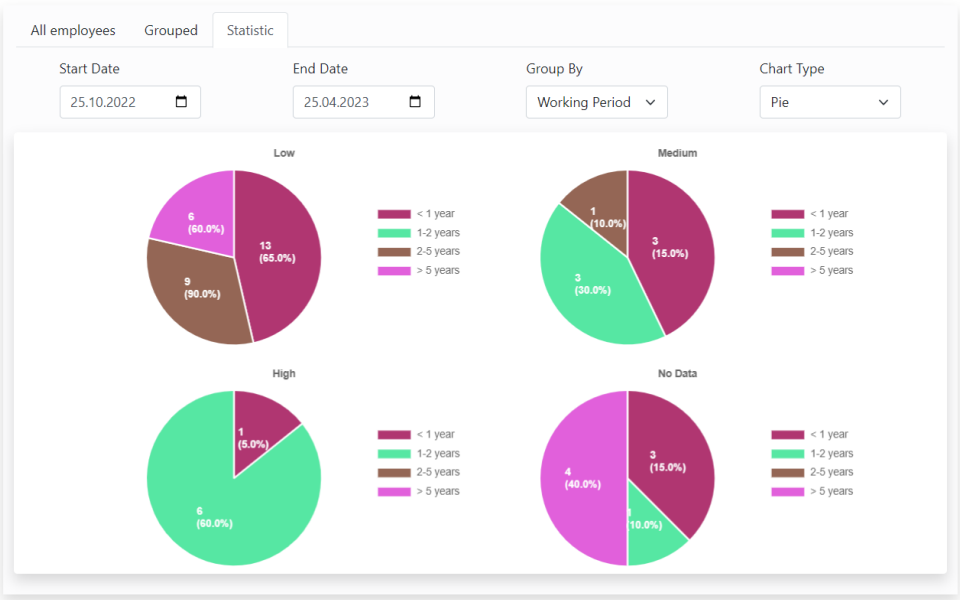

Рисунок 3.23 – Статистичні дані згруповані за часом роботи працівника в компанії представлені круговими діаграмами

Окрім загальних даних по компанії, користувач може переглянути сторінку кожного працівника. На ній відображаються всі дані працівника, його

останні показники сканування, а також графік всіх індикаторів (рис. 3.24). Окрім відсотку «Нормальних», «Стресових», «Самотніх» твітів, відображається показник «Вигоряння», який вираховується як сума «Стресових» та «Самотніх» твітів, а також кількість опублікованих твітів (рис. 3.25).

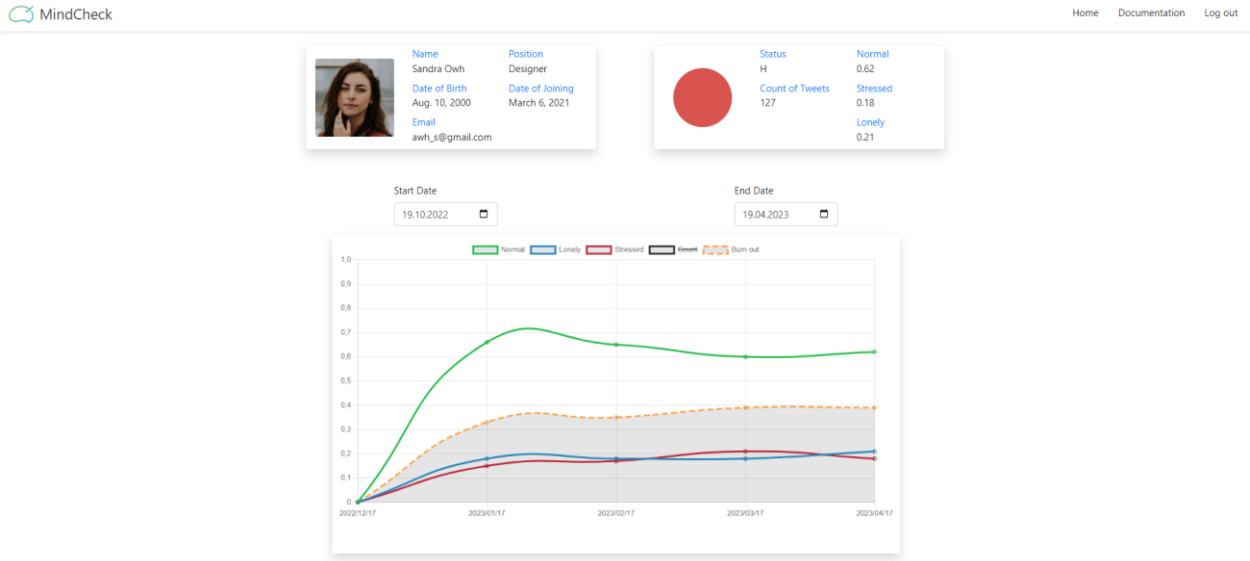

Рисунок 3.24 – Сторінка працівника у високій зоні ризику з представленими даними аналізу

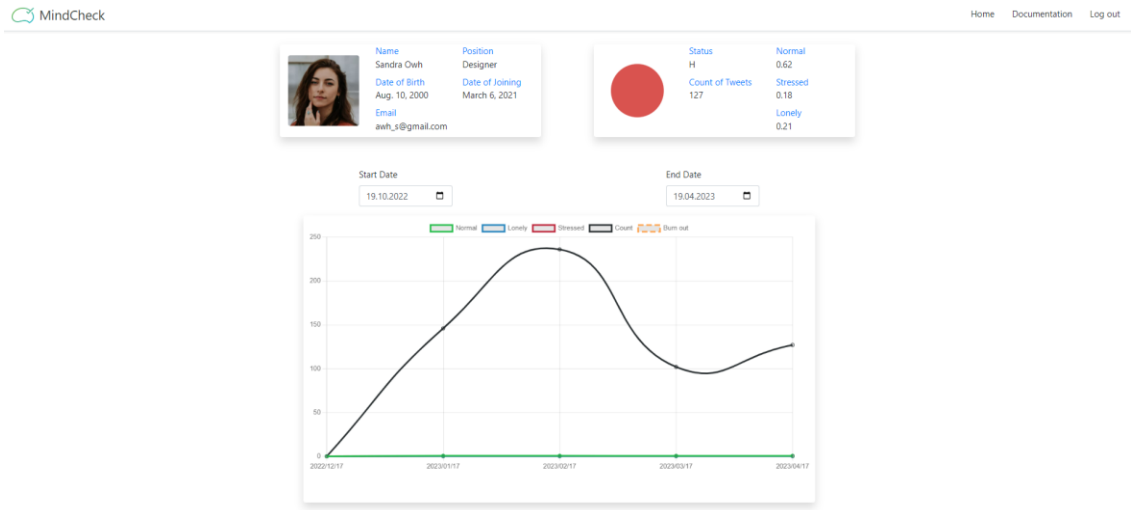

Рисунок 3.25 – Сторінка працівника у високій зоні ризику з даними про кількість опублікованих постів у Твітер

На рисунку 3.25 ми бачимо дані працівника, у якого високий рівень ризику вигоряння відсоток нормальних твітів = 62%, а значення виснаження = 39%. Для

порівняння на рисунку 3.26 зображена сторінка працівника в низькій зоні ризику, значення виснаження якого не перевищує 15%.

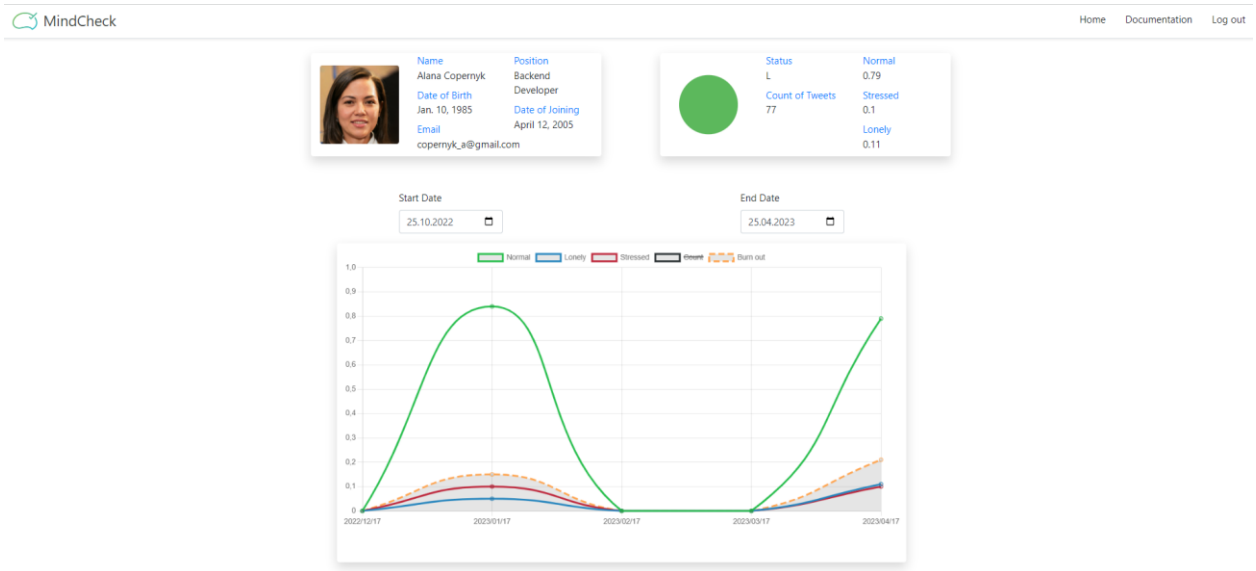

Рисунок 3.26 – Сторінка працівника у низькій зоні ризику з даними про кількість опублікованих постів у Твітер

Так як для аналізу в ролі працівників були обрані акаунти реальних людей, можна порівняти їх сторінки та тексти твітів. З рисунку 3.27 можна побачити, що твіти містять в собі певне відчуття стресу, незадоволеності та самотності, що і було визначено моделлю. В той час як на рисунку 3.28, всі твіти мають нейтральний або позитивний характер.

Рисунок 3.27 – Твіти працівника в високому рівні ризику

Рисунок 3.28 – Твіти працівника в низькому рівні ризику

Даний веб-застосунок не рекомендований для використання на мобільних девайсах, проте при розробці було враховано можливість адаптивності сайту. На рисунках 3.29, 3.30 зображені сторінки веб-застосунку при середньому розширенні екрану.

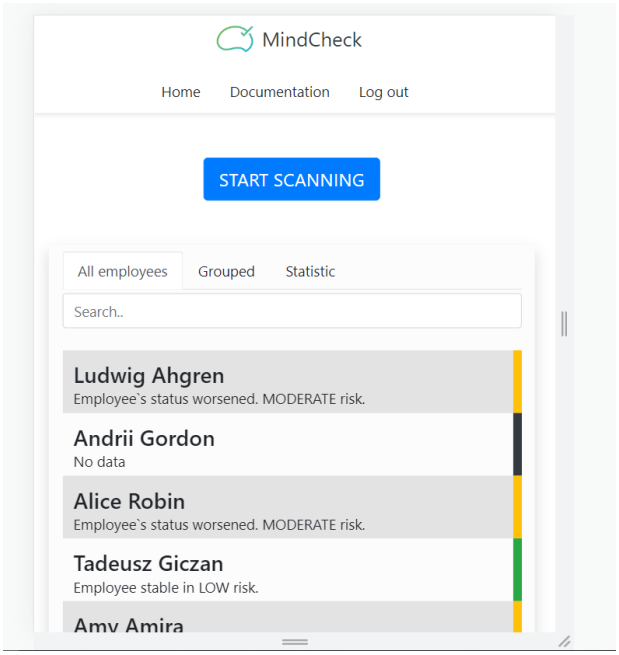

Рисунок 3.29 – Головна сторінка застосунку на екрані планшету

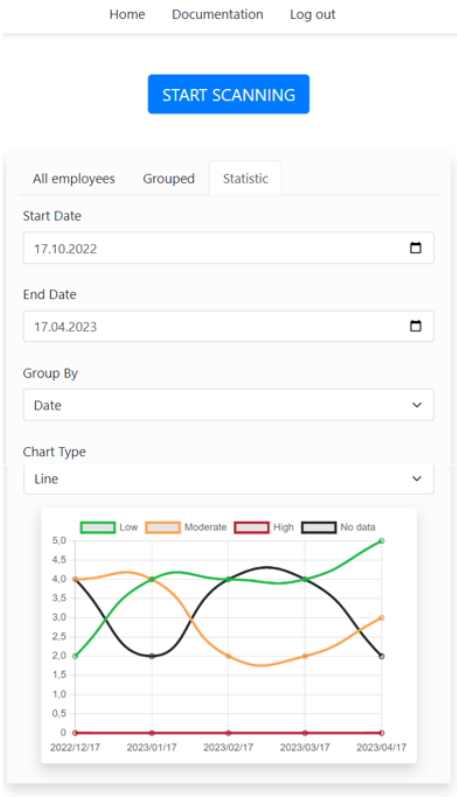

Рисунок 3.30 – Сторінка статистики застосунку на екрані планшету

Висновок до третього розділу

У ході роботи над задачею було реалізовано веб-застосунок з використанням фреймворку Django, з інтелектуальним модулем на основі методу опорних векторів з калібруванням. Було проведено експерименти, щодо впливу на точність моделі зміни основних її параметрів. Було протестовано роботу системи як на окремо відібраних текстах, так і на реальних постах людей.

Результати роботи програми було графічно показано в розробленому веб-застосунку.

З отриманих результатів можна стверджувати, що програма успішно класифікує як окремі тексти, так і стан працівників в цілому з можливістю подальшого аналізу.

ВИСНОВОК

У ході роботи над задачею було проаналізовано актуальність поставленого завдання, проблематику обраної області, розглянуто наявні рішення в сфері визначення стресу та виснаження.

Було розглянуто роботи, які спеціалізуються на класифікації текстів з твітеру та в цілому, порівняно результати різних моделей при роботі з текстами та обрано за основу метод опорних векторів з лінійним ядром для реалізації інтелектуального модуля.

Було проведена розробка та модифікація обраної моделі, додано можливість представлення результатів у вигляді належності до класу. Після навчання модель продемонструвала наступні показники точності 98,7% на навчальній і 93% на тестовій вибірках. Для знаходження оптимальних параметрів моделі, було проведено експеременти для визначення впливу наступних параметрів на точність моделі: тип поєднання калібратора та моделі, штраф, ваги класів, параметр регуляризації.

Було визначено основні інформаційні складові застосунку, спроектовано його архітектуру, базу даних, відношення між компонентами, дизайн.

З використанням мови програмування Python та Django було розроблено веб-застосунок, де було реалізовано можливість запуску аналізу твітів зі сторінок реальних людей (працівників), графічне відображення результатів аналізу, графічне відображення статистики по всім працівникам.

До переваг розробки можна віднести:

1. Точність визначення ризику виснаження;
2. Економія часу та людської сили;
3. Графічне представлення результатів;
4. Отримання текстів для аналізу із загально доступної соцмережі.

До недоліків розробки можна віднести:

1. Довгий час аналізу при великій кількості працівників;

2. Нестабільність мережі твітер й залежність роботи програми від політики компанії.

Для подальшої роботи над проектом, можна виділити такі завдання:

1. Реалізація аналізу стану працівників багатопотоково;
2. Додавання можливості завантаження тексту для аналізу з текстового файлу;
3. Можливість навчання моделі на власних даних.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Rob Errera. 2022. «Hard to believe workplace stress statistics (US & Global)» Електр. дані. – Режим доступу: https://www.tonerbuzz.com/blog/workplace-stress-statistics/
2. Emma Seppälä and Marissa King. 2017. “Burnout at Work Isn’t Just About Exhaustion. It’s Also About Loneliness”. Електр. дані. – Режим доступу: https://hbr.org/2017/06/burnout-at-work-isnt-just-about-exhaustion-its-also-about-loneliness
3. Susan M. Healthfield. 2022 “How Stress Affects Your Work”. Електр. дані. – Режим доступу: https://www.thebalancemoney.com/understanding-stress-and-how-it-affects-the-workplace-1919200
4. Steven Zauderer. 2022 «67 Workplace Stress Statistics In 2023». Електр. дані. – Режим доступу: https://www.crossrivertherapy.com/stress-statistics-and-facts
5. Crystal Ro. 2017 “17 Funny Tweets That Are Like “YASS, This Is Totally Me””. Електр. дані. – Режим доступу: https://www.buzzfeed.com/crystalro/ugh-this-is-all-of-us?utm_term=.ikjjrBLA9&sub=4445765_10363748&epik=dj0yJnU9RWk2ZW9FQ2c2eGxzRlhKR0JENGRpRTdHU3FVVy1pcnAmcD0wJm49TGdrS1JIM3VYVV95ZWpIcDYyRHBqdyZ0PUFBQUFBR1BxUnVJ
6. Prashanth KVTKN, Tene Ramakrishnudu, A novel method for detecting psychological stress at tweet level using neighborhood tweets, Journal of King Saud University - Computer and Information Sciences, Volume 34, Issue 9, 2022. Електр. дані. – Режим доступу: https://www.sciencedirect.com/science/article/pii/S1319157821002184?via%3Dihub

7. Nijhawan, T., Attigeri, G. & Ananthakrishna, T. Stress detection using natural language processing and machine learning over social interactions. J Big Data 9, 33 (2022). Електр. дані. – Режим доступу : https://rdcu.be/cY9V5
8. Vishal Dham et al 2021 J. Phys.: Conf. Ser. **1950** 012047 Електр. дані. – Режим доступу: https://iopscience.iop.org/article/10.1088/1742-6596/1950/1/012047/pdf
9. Baheti, R.R., & Kinariwala, S. (2019). Detection and Analysis of Stress using Machine Learning Techniques. International Journal of Engineering and Advanced Technology. Електр. дані. – Режим доступу: https://www.ijeat.org/wp-content/uploads/papers/v9i1/F8573088619.pdf
10. Arsh Kandroo. Behavioural Tweets. Dataset. Електр. дані. – Режим доступу: https://www.kaggle.com/datasets/arshkandroo/behavioural-tweets
11. Hsu, Bi-Min. 2020. "Comparison of Supervised Classification Models on Textual Data" Mathematics 8, no. 5 Електр. дані. – Режим доступу: https://www.mdpi.com/2227-7390/8/5/851/htm
12. Baeldung. 2022. Multiclass Classification Using Support Vector Machines. Електр. дані. – Режим доступу: https://www.baeldung.com/cs/svm-multiclass-classification
13. Tiobe Index. Електр. дані. – Режим доступу: https://www.tiobe.com/tiobe-index/
14. Documentation scikitLearn. Електр. дані. – Режим доступу: https://scikit-learn.org/stable/modules/generated/sklearn.svm.LinearSVC.html

ДОДАТОК

Програмний код моделі аналізу текстів твітів (LinerSVC.py)

```
import pickle
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import seaborn as sns
from sklearn import metrics
from sklearn.calibration import CalibratedClassifierCV
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics import confusion_matrix
from sklearn.model_selection import train_test_split
from sklearn.svm import LinearSVC
from sklearn.model_selection import cross_val_score
import time
time1 = time.perf_counter()

normal_df = pd.read_csv("data\\Dataset2\\Normal_Tweets.csv")
normal_df['type'] = "Normal"
normal_df['type_id'] = 0

stressed_df = pd.read_csv("data\Dataset2\Stressed_Tweets.csv")
stressed_df['type'] = "Stressed"
stressed_df['type_id'] = 2

lonely_df = pd.read_csv("data\Dataset2\Lonely_Tweets.csv")
lonely_df['type'] = "Lonely"
lonely_df['type_id'] = 3

all_data_df = pd.concat([normal_df,lonely_df], ignore_index=True)
all_data_df.drop(columns=['index'])
all_data_df = pd.concat([all_data_df, stressed_df], ignore_index=True)

np.savetxt(r'data\\Dataset2\\alldata.txt', all_data_df.text.values, fmt='%s')

type_id_df = all_data_df[['type', 'type_id']].drop_duplicates().sort_values('type_id')
type_to_id = dict(type_id_df.values)
id_to_type = dict(type_id_df[['type_id', 'type']].values)

fig = plt.figure(figsize=(8,6))
all_data_df.groupby('type').text.count().plot.bar(ylim=0)

vectorizer = TfidfVectorizer(sublinear_tf=True,norm='l2')
text = vectorizer.fit_transform(all_data_df.text).toarray()

labels = all_data_df['type_id']

X_train, X_test, y_train, y_test = train_test_split(text, labels,  test_size=0.2, random_state=0)

model = LinearSVC(class_weight='balanced', penalty='l2', C=1, max_iter=100, dual=False, tol=0.0001)
```

```
calibrated_svc = CalibratedClassifierCV(base_estimator=model)
calibrated_svc.fit(X_train,y_train)
predicted = calibrated_svc.predict(X_test)

score = calibrated_svc.score(X_train, y_train)
print("Model calibrated accurace score: ", score)

filename = 'saved_model\model2\model_dataset2.pkl'
with open(filename,'wb') as f:
   pickle.dump(calibrated_svc,f)

time2 = time.perf_counter()

print(f"Time =  {time2 - time1:0.4f} seconds")

conf_mat = confusion_matrix(y_test, predicted)
fig, ax = plt.subplots(figsize=(8,8))
sns.heatmap(conf_mat, annot=True, fmt='d',
          xticklabels=type_id_df.type.values, yticklabels=type_id_df.type.values)
plt.ylabel('Actual')
plt.xlabel('Predicted')
plt.show()
print(metrics.classification_report(y_test, predicted, target_names=all_data_df['type'].unique()))
```

Програмний код видобутку та обробки текстів з мережі Твітер (Tweets.py)

```
import configparser
import tweepy
from dateutil.relativedelta import relativedelta
import re
import datetime
from nltk.corpus import stopwords
import snscrape.modules.twitter as sntwitter
import pandas as pd
import spacy
nlp = spacy.load('en_core_web_sm', disable=['parser', 'ner'])

class Twitter(object):
   def __init__(self, stop_words = 'english', config = 'analizer\config.ini'):
      self.stop_words = set(stopwords.words(stop_words))
      self.config_path = config

   def config(self,url):
      config =configparser.RawConfigParser()
      config.read(url)

      api_key = config['twitter']['api_key']
      api_key_secret = config['twitter']['api_key_secret']
      access_token = config['twitter']['access_token']
      access_token_secret = config['twitter']['access_token_secret']

      auth = tweepy.OAuthHandler(api_key, api_key_secret)
```

```
        auth.set_access_token(access_token, access_token_secret)

        api = tweepy.API(auth)
        return api

    def get_tweets_snc(self, username, end_date):
        start_date = end_date + relativedelta(months=-1)
        number_of_tweets=200
        tweets = []
        for i,tweet in enumerate(sntwitter.TwitterSearchScraper(f'from:{username} since:{start_date}
until:{end_date}').get_items()):
            if i>number_of_tweets:
                break
            cleaned = self.clean_tweets(tweet.rawContent)
            if cleaned:
                tweets.append(cleaned)
        return tweets


    def get_tweets(self,username,end_date):
            api = self.config(self.config_path)
            start_date = end_date + relativedelta(months=-1)
            number_of_tweets=200

            tweets = []
            tmpTweets = api.user_timeline(screen_name=username,count=number_of_tweets,exclude_replies = True)
            for tweet in tmpTweets:
                if tweet.created_at.date() < end_date and tweet.created_at.date() > start_date:
                    cleaned = self.clean_tweets(tweet.text)
                    if cleaned:
                        tweets.append(cleaned)

            while (tmpTweets[-1].created_at.date() > start_date):
                tmpTweets = api.user_timeline(screen_name=username, count=number_of_tweets,exclude_replies = True,
max_id = tmpTweets[-1].id)
                if len(tmpTweets[1:])!=0:
                    for tweet in tmpTweets[1:]:
                        if tweet.created_at.date() <= end_date and tweet.created_at.date() > start_date:
                            cleaned = self.clean_tweets(tweet.text)
                            if cleaned:
                                tweets.append(cleaned)
                else:
                    break
            return tweets

    def clean_tweets(self,text):
        text = text.lower().strip()
        text = re.sub('@[^\s]+','',text)
        text = re.sub('(http|ftp|https):\/\/([\w_-]+(?:(?:\.[\w_-]+)+))([\w.,@?^=%&:\/~+#-]*[\w@?^=%&\/~+#-])','',text)
        text = re.sub('[^\x00-\x7F]+','',text)
        text =  re.sub('[^a-zA-Z]', ' ', text)
        text = re.sub(' +', ' ', text)
        clean_text=""
```

```
        for x in text.split():
            if x not in self.stop_words:
                clean_text=" ".join([clean_text, x])
        doc = nlp(clean_text)
        clean_text=" ".join([token.lemma_ for token in doc])
        return clean_text
```

Модуль інтелектуального ядра для аналізу текстів твітів та інтерпретації результатів (Analyzer.py)

```
import pickle
import numpy as np
import analizer.Tweets as tw
from datetime import datetime
from sklearn.feature_extraction.text import TfidfVectorizer

from celery import shared_task
from celery_progress.backend import ProgressRecorder
from personal.models import Employee, State, Result
import re
from nltk.corpus import stopwords
import spacy
nlp = spacy.load('en_core_web_sm', disable=['parser', 'ner'])

def openFiles():
    f = open("analizer\data\\alldata.txt", "r")

    textData =f.read()
    textData = textData.split('\n')

    with open('analizer\Model\model.pkl', 'rb') as f:
        model = pickle.load(f)
    return textData, model

def getAverage(arr):
    probs = [0., 0., 0.]
    counts= [0,0,0]
    if len(arr)!=0:
        for item in arr:
            probs+=item
            max_val = np.max(item)
            counts[np.where(item == max_val)[0][0]]+=1
        probs =[round(x/len(arr),2) for x in probs]
    return (probs,counts)

def change_status(worker, new_status_id):
    emp = Employee.objects.get(employee_id=worker['employee_id_id'])
    state = State.objects.get(state_id=new_status_id)

    emp.state_id = state
    emp.save()
```

```
    return

def save_results(worker, date, probs, count):
    employee = Employee.objects.get(employee_id=worker['employee_id_id'])
    result = {"percent_N" : probs[0], "percent_S": probs[1], "percent_L" : probs[2],
    "count_N" : count[0], "count_S" : count[1], "count_L" : count[2]}
    result['status'], _ = analize_results(result)
    Result.objects.update_or_create(
    employee_id=employee, scan_date = date,
    defaults=result,
    )

def get_average_count(results):
    count = 0
    number = 0
    for result in results:
        if result['count_N']+result['count_S']+result['count_L'] !=0:
            count+= result['count_N']+result['count_S']+result['count_L']
            number+=1
    if number==0:
        return 0.
    return count/number

def analize_results(results):
    if (results['count_N']+results['count_S']+results['count_L']==0):
        return "N",results['count_N']+results['count_S']+results['count_L']
    elif results['percent_N']>=0.75:
        return "L",results['count_N']+results['count_S']+results['count_L']
    elif (results['percent_S'] + results['percent_L']>=0.35):
        return "H",results['count_N']+results['count_S']+results['count_L']
    else:
        return "M",results['count_N']+results['count_S']+results['count_L']

def find_new_status_id(new_res, new_count, prev_res=None, avr_count=None):
    if prev_res is None:
        new_status_id = State.objects.filter(status=new_res[0]).filter(progress__exact='').values_list('state_id',
flat=True)[0]
        return new_status_id
    elif prev_res=='N' and new_res=='N':
        new_status_id = State.objects.filter(status=new_res).filter(progress__exact='').values_list('state_id', flat=True)[0]
        return new_status_id

    if avr_count is not None and avr_count!=0:
        change = (abs(new_count- avr_count)) * 100 / avr_count
        if change >70 and new_count>15 and avr_count>=15 and new_res !='H':
            new_status_id = State.objects.filter(status='M').filter(note = "Big change in number of posted
tweets").values_list('state_id', flat=True)[0]
            return new_status_id

    if (new_res=='N'):
        new_status_id = State.objects.filter(status=new_res).filter(progress__exact='').values_list('state_id', flat=True)[0]
    elif (prev_res == new_res):
        new_status_id = State.objects.filter(status=new_res).filter(progress='S').values_list('state_id', flat=True)[0]
```

```
    elif(new_res=='L'):
        new_status_id = State.objects.filter(status=new_res).filter(progress='I').values_list('state_id', flat=True)[0]
    elif(new_res=='H'):
        new_status_id = State.objects.filter(status=new_res).filter(progress='W').values_list('state_id', flat=True)[0]
    elif(new_res=='M' and prev_res=='L'):
        new_status_id = State.objects.filter(status=new_res).filter(progress='W').values_list('state_id', flat=True)[0]
    else:
        new_status_id = State.objects.filter(status='M').filter(progress='I').values_list('state_id', flat=True)[0]
    return new_status_id

def form_new_status(worker):
    employee = Employee.objects.get(employee_id=worker['employee_id_id'])
    last_results = Result.objects.filter(employee_id=employee).order_by('-scan_date').values()
    last_result_status, last_count = analize_results(last_results[0])
    if last_results.count() > 1:
        prev_result_status,_ = analize_results(last_results[1])
        avr_count = get_average_count(last_results[1:3])
        new_status_id=find_new_status_id(last_result_status,last_count, prev_result_status, avr_count)
    else:
        new_status_id=find_new_status_id(last_result_status, last_count)

    change_status(worker,new_status_id)


@shared_task(bind=True)
def analize_tweets(self,workers,date):
    progress_recorder = ProgressRecorder(self)
    textData,model=openFiles()
    i=0
    twitter = tw.Twitter()
    date = datetime.strptime(str(date), "%Y-%m-%d").date()
    for worker in workers:
      # tweets = twitter.get_tweets(worker['username'],date)
        tweets = twitter.get_tweets_snc(worker['username'],date)
        results = []
        for tweet in tweets:
            to_predict = [tweet]
            vectorizer = TfidfVectorizer(sublinear_tf=True,norm='l2')
            vectorizer.fit_transform(textData).toarray()
            to_predict_vec = vectorizer.transform(to_predict)
            probs = np.round(model.predict_proba(to_predict_vec)[0],decimals=2)
            results.append(probs)
            print(probs)

        average = getAverage(results)
        save_results(worker,date,average[0], average[1])
        form_new_status(worker)
        i+=1
        progress_recorder.set_progress(i,len(workers))
    return ''

def analize_text(text):
    text = tw.clean_text(text)
```

```
textData,model=openFiles()
vectorizer = TfidfVectorizer(sublinear_tf=True,norm='l2')
vectorizer.fit_transform(textData).toarray()
to_predict_vec = vectorizer.transform([text])
probs = np.round(model.predict_proba(to_predict_vec)[0],decimals=2)
return probs
```