https://doi.org/10.31891/2219-9365-2025-81-12
УДК 004.42

ВАЛЬ Олександр
Чернівецький національний університет імені Юрія Федьковича
e-mail: ol.val@chnu.edu.ua
https://orcid.org/0009-0001-0323-9741
ВОДЯНЧУК Валерія
Middle React developer
ГАЗДЮК Катерина
Чернівецький національний університет імені Юрія Федьковича
e-mail: k.gazdiuk@chnu.edu.ua
https://orcid.org/0000-0002-7568-4422
ТАРНОВЕЦЬКА Ольга
Чернівецький національний університет імені Юрія Федьковича
e-mail: savinskaolga@gmail.com
https://orcid.org/0009-0001-4366-5366

CHROME РОЗШИРЕННЯ ДЛЯ ПІДТРИМКИ ЗДОРОВ'Я ПІД ЧАС РОБОТИ В БРАУЗЕРІ

Робота присвячена дослідженню та впровадженню розширення для браузера, яке допомагає користувачам підтримувати здоров'я під час тривалої роботи за комп'ютером. Основні завдання цього розширення включають: нагадування користувачам про необхідність регулярних перерв для виконання фізичних вправ, що допоможуть зняти напругу з м'язів спини та очей, надання користувачам можливості налаштовувати частоту та тип нагадувань відповідно до їхніх індивідуальних потреб і уподобань, надання доступу до спеціалізованих вправ та рекомендацій через вебсторінки, що сприяють поліпшенню здоров'я та самопочуття.

Аналіз альтернатив показав важливість створення розширення, яке міститиме такі функції: нагадування для очей та постави, інформування про важливість підтримання водного балансу в організмі, можливість активувати та деактивувати сповіщення, налаштування часу нагадувань, сторінку з вправами для постави та очей, зворотній відлік під час виконання вправ, перевірку правильності постави та сповіщення про її неправильність, навіть якщо користувач знаходиться на інших вкладках.

Проаналізовано функціональні та нефункціональні вимоги до розширення та створено діаграму варіантів використання. Реалізовано програмний продукт за допомогою Chrome API, використовуючи такі API: storage (для зберігання даних користувача), alarms (для відстеження часу), notifications (для надсилання сповіщень користувачеві), tts (для озвучування тексту) та tabs (для управління вкладками веббраузера). Обґрунтовано вибір технологій, які використані для розробки програмного продукту, а саме: мова програмування TypeScript, бібліотека React, бібліотека компонент MUI для створення користувацьких інтерфейсів, як середовище розробки використаноVisual Studio Code та Vite для збірки проекту.

Для оцінки постави використано результати машинного навчання.

Ключові слова: програмний продукт, браузерне розширення, вебсторінка, Chrome API, TypeScript, React.

VAL Oleksandr
Yuriy Fedkovych Chernivtsi National University
VODYANCHUK Valeriia
Middle React developer
HAZDIUK Kateryna, TARNOVETSKA Olha
Yuriy Fedkovych Chernivtsi National University

CHROME EXTENSION FOR MAINTAINING HEALTH DURING BROWSER WORK

This paper is dedicated to the research and implementation of a browser extension that helps users maintain their health during prolonged computer work. The main tasks of this extension include reminding users about the need for regular breaks to perform physical exercises that help relieve muscle tension in the back and eyes, providing users with the ability to adjust the frequency and type of reminders according to their individual needs and preferences, and providing access to specialized exercises and recommendations through web pages that contribute to improved health and well-being.

An analysis of alternatives highlighted the importance of creating an extension that includes the following features: reminders for eyes and posture, informing about the importance of maintaining water balance in the body, the ability to activate and deactivate notifications, configure reminder times, a page with exercises for posture and eyes, a countdown timer during exercises, posture correctness checking, and notifications about incorrect posture even if the user is on other tabs.

Functional and non-functional requirements for the extension were analysed and a use case diagram was created. The software product was implemented using the Chrome API, utilizing the following APIs: storage (for storing user data), alarms (for time tracking), notifications (for sending notifications to the user), tts (for text-to-speech), and tabs (for managing browser tabs). The choice of technologies used for the development of the software product was justified: TypeScript programming language, React library, MUI component library for creating user interfaces, Visual Studio Code as the development environment, and Vite for project build.

The results of machine learning were used to estimate posture.

Keywords: software product, browser extension, web page, Chrome API, TypeScript, React.

ПОСТАНОВКА ПРОБЛЕМИ У ЗАГАЛЬНОМУ ВИГЛЯДІ ТА ЇЇ ЗВ'ЯЗОК ІЗ ВАЖЛИВИМИ НАУКОВИМИ ЧИ ПРАКТИЧНИМИ ЗАВДАННЯМИ

У сучасному світі майже кожна друга професія передбачає тривале перебування перед екраном комп'ютера. Ця тенденція також поширюється на освітню сферу, де багато учнів та студентів навчаються онлайн, спершу через поширення вірусу, а тепер через зовнішню загрозу. Сидячий спосіб життя негативно впливає на здоров'я людей, з часом призводячи до численних проблем зі спиною, зором та до супутніх хвороб. Середньостатистична людина проводить у сидячому положенні більше часу, ніж у русі, що спричиняє гіподинамію та інші негативні наслідки для організму. Зокрема, ослаблення м'язів, оскільки недостатнє використання м'язів призводить до їх атрофії та зниження сили; погіршення кровообігу, що підвищує ризик утворення тромбів; збільшення ризику серцево-судинних захворювань, проблеми з опорно-руховим апаратом, можуть виникати болі в спині, шиї та суглобах через слабкість м'язів і неправильну поставу; зниження обміну речовин, що може призвести до набору ваги, ожиріння та розвитку діабету 2 типу.

Ситуація з очима є подібною. Тривале напруження очей і робота з комп'ютером понад 3 години на день можуть призвести до синдрому сухого ока, головного болю та зниження продуктивності. Погане освітлення, неправильна відстань до монітора, та тривале фокусування на близьких об'єктах також сприяють зоровій втомі. Очам потрібні регулярні перерви для відпочинку та розслаблення, щоб уникнути цих проблем.

Іншою важливою проблемою є недостатнє споживання води протягом дня. Багато людей не звертають уваги на те, що недостатній рівень гідратації може призвести до загальної втоми, зниження концентрації та погіршення когнітивних функцій. Регулярне нагадування про необхідність пити воду може допомогти підтримувати оптимальний рівень гідратації та загальне здоров'я.

Тому під час роботи за ноутбуком важливо дотримуватися певних правил, щоб уникнути негативного впливу на зір і опорно-руховий апарат.

Метою даної роботи є розробка Chrome-розширення[1], що допоможе розвантажити спину та очі людини, нагадуючи їй про необхідність регулярних розминок під час роботи в браузері та буде нагадувати про необхідність гідрації організму. Це допоможе залишатися більш сконцентрованим протягом дня та уникнути перевтоми.

Розширення має простий та зрозумілий дизайн, дозволяє гнучко налаштовувати нагадування і містить додатковий функціонал. Користувач може виконувати запропоновані системою вправи, перейшовши на відповідну вебсторінку, доступну у мережі для перегляду у веббраузері.

Окрім вищезазначених функцій, розширення має механізм перевірки правильності постави та сповіщення користувача у разі її порушення. Це дозволить уникнути проблем, пов'язаних з неправильною поставою навіть тоді, коли користувач працює на інших вкладках. Варто зауважити, що дане розширення підтримується одним із найпопулярніших браузерів Google Chrome, що робить його актуальним для широкого кола користувачів.

АНАЛІЗ ДОСЛІДЖЕНЬ ТА ПУБЛІКАЦІЙ

На сьогоднішній день ринок розширень для Chrome не настільки популярний, як традиційні вебдодатки, але з кожним днем їхня популярність зростає. Це пов'язано з тим, що розширення дозволяють доповнити функціонал вебдодатків або впровадити нові функції поверх вебсторінок.

На сучасному ринку існує кілька програмних продуктів, які допомагають користувачам розвантажити спину та очі, нагадуючи про необхідність робити розминку під час роботи в браузері.

Розглянемо більш детально кожен з них.

Posture Reminder [2] - це розширення, яке періодично нагадує користувачеві про необхідність коригувати свою поставу. Його особливості включають вбудований перемикач увімкнення/вимкнення, можливість налаштування нагадувань через різні інтервали: 15 хвилин, 30 хвилин, 1 годину, 2 години або випадкові (від 30 хвилин до 4 годин). Це дозволяє сформувати звичку підтримувати здорову поставу під час роботи за комп'ютером.

PostureMinder[3] - це розширення, яке дозволяє користувачеві встановлювати частоту нагадувань про необхідність коригувати поставу. Основні функції включають можливість увімкнення/вимкнення, налаштування інтервалів часу перед закриттям нагадування та можливість закриття за допомогою кліку. Розширення також надає нагадування про потребу в погулянці.

Eye Care Notification[4] - це розширення, яке регулярно нагадує користувачеві про необхідність відводити погляд від екрана кожні 20 хвилин (цей інтервал часу можна налаштувати). Рекомендується на кожні 20 хвилин припиняти роботу, фокусуватися на об'єктах на відстані 20 метрів і активно мигати протягом 20 секунд. Також доступна опція відкриття зеленого екрана для відпочинку та розслаблення очей.

Pomodoro Work + Stretch[5]. Розширення для підвищення продуктивності, яке пропонує 25-хвилинні інтервали роботи та 5-хвилинні перерви для зручної розтяжки за столом, включаючи визначення правильної форми вправ. Його особливості включають в себе таймер для роботи та розтяжки, набір вправ для зменшення болю та напруги м'язів під час 5-хвилинної перерви, детектор форми для підтримки правильної

позиції під час виконання вправ, зворотний відлік до наступної перерви чи роботи, сповіщення через вкладки і спливаючі вікна, а також налаштовану функцію увімкнення/вимкнення.

Posture! Posture! Posture! [6] - це розширення, яке допомагає користувачеві відслідковувати його поставу під час перегляду веб-сторінок. Воно спочатку фіксує базову позицію, в якій користувач сидить з правильною поставою. Коли користувач відхиляється від цієї позиції, на екрані застосовується ефект "розмиття". Як тільки користувач повертається до правильної постави, ефект розмиття зникає.

Розглянемо функціональні можливості перелічених розширень у вигляді таблиці(Табл. 1).

Примітка. N/A (not applicable) використовується для розширень, коли певний параметр не є застосовним або не має значення, наприклад немає нагадувань, отже немає і можливості вимкнути чи увімкнути їх, також додаткові конфігураційні опції не застосовуються).

Таблиця 1

Функціональні можливості альтернатив(частина 1)

Розширення	Нагадування для постави	Нагадування для очей	Нагадування випити води	Вимкнути/ ввімкнути	Конфігурація часу
Posture Reminder	+	-	-	+	+
PostureMinder	+	-	-	+	+
Eye Care Notification	-	+	-	-	+
Pomodoro Work + Stretch	+	-	-	+	-
Posture! Posture! Posture!	-	-	-	N/A	N/A

Таблиця 2

Функціональні можливості альтернатив(частина 2)

Розширення	Сторінка з вправами для постави	Сторінка з вправами для очей	Зворотній тай-мер при вико-нанні вправ	Перевірка правильності постави	Сповіщення про неправильну поставу
Posture Reminder	-	-	N/A	-	-
PostureMinder	-	-	N/A	-	-
Eye Care Notification	-	-	N/A	-	-
Pomodoro Work + Stretch	+	-	+	+	-
Posture! Posture! Posture!	-	-	N/A	+	+

Після аналізу альтернатив стає очевидним, що немає комплексної системи для здорового інтернет-серфінгу, яка б включала всі важливі аспекти, такі як правильна постава, захист очей та нагадування про водний баланс організму. Існуючі розширення зазвичай зосереджені на одному аспекті. Це свідчить про необхідність розробки системи, яка поєднає функціональність всіх цих рішень в одному продукті.

Отже, розроблений додаток повинен включати наступні функції: нагадування про правильну поставу, захист очей та регулярне пити воду; можливість увімкнення/вимкнення сповіщень; налаштування інтервалів нагадувань; сторінку з вправами для покращення постави та здоров'я очей; зворотній відлік часу під час виконання вправ; перевірку правильності постави і сповіщення про неправильність навіть на інших вкладках браузера.

Обгрунтування вибору технологій. Виходячи з можливостей, які надає розширення Chrome, такі як підтримка JavaScript, HTML і CSS, а також з урахуванням сучасних інструментів для розробки і оптимізації коду, ми обрали наступний стек технологій:

- мова програмування *TypeScript;*
- бібліотека *React*;
- бібліотека UI компонент *MUI*;
- збірник *Vite*;
- середовище для розробки *Visual Studio Code.*

Опишемо детально вибір стеку технологій.

Typescript [7] — позиціонується як засіб розробки вебзастосунків, що розширює можливості JavaScript. JavaScript є динамічною, об'єктно-орієнтованою прототипною мовою програмування, яка використовується для створення скриптів на веб-сторінках. JavaScript дозволяє взаємодіяти з користувачем, керувати браузером, асинхронно обмінюватися даними з сервером і змінювати структуру та вигляд веб-сторінок. На відміну від JavaScript, яка має динамічну типізацію, TypeScript пропонує статичну типізацію, що дозволяє попередити багато помилок у програмному коді завдяки строгим типам даних.

React [8] - це JavaScript-бібліотека для створення користувацьких інтерфейсів, яка значно спрощує процес створення інтерактивних веб-додатків. Для роботи з React вам потрібно лише описати, як різні частини інтерфейсу повинні виглядати в різних станах вашого додатку. React автоматично визначає, які компоненти потрібно оновити або перерендерити, коли дані змінюються, що забезпечує ефективність і швидкість в роботі програми.

Visual Studio Code – засіб для створення, редагування та зневадження сучасних веб застосунків і програм для хмарних систем. Це програмне забезпечення розповсюджується безкоштовно і має версії для операційних систем Windows, Linux і macOS. Microsoft представила Visual Studio Code на конференції Build 2015 у квітні 2015 року.

Bundler або збірник, є на сьогодні незамінним інструментом у сучасній веб розробці [9]. Це дозволяє розробникам легко працювати з кодом на мовах, які не підтримуються браузером за замовчуванням(TypeScript, SCSS). Також він відповідає, за адаптацію програми для середовища development або production. У випадку створення власного проєкту, bundler не лише допоможе максимально оптимізувати код, а й зробить програму більш ефективною. Особливість Vite[10,11] полягає в тому, що він використовує переваги модулів ES (ECMAScript). У середовищі розробки Vite запускається сервер, який відповідає за компіляцію і обслуговування всіх необхідних залежностей через модулі ES. Цей підхід дозволяє Vite оптимізувати процеси і забезпечувати лише необхідний на даний момент код.

Material-UI (Mui) - це бібліотека, яка надає широкий набір інструментів для створення інтерфейсів користувача. Вона пропонує готові до використання компоненти і дозволяє швидко і ефективно впроваджувати нові функції. Можна використовувати Material-UI з власною системою дизайну або почати з їх повністю завантаженої бібліотеки компонентів.

Опис структури chrome розширення. Структура chrome розширення полягає в наборі запакованих файлів, що містять переважно код, зображення, конфігураційні файли та інші веб-ресурси, необхідні для його функціонування. Ці розширення базуються на веб-технологіях, таких як HTML, CSS і JavaScript.

Веб-браузер виконує ці програмні рішення в ізольованому середовищі, що розділяє їх від коду, який працює на всіх ваших відкритих сторінках веббраузера.

Chrome розширення складається з 3 частин:

1. Content script - це базові JavaScript файли, які виконуються поверх веб-сторінки, але не вбудовуються безпосередньо в неї. Це середовище, яке зазвичай використовується для роботи більшості основних розширень браузера. У content script можна читати та змінювати більшість елементів на веб-сторінці шляхом маніпуляцій з DOM (об'єктна модель документа).
2. У background.js ми маємо найвищий рівень доступу до більшості API браузера, тому це зазвичай основна частина для обробки складніших розширень. Цей скрипт працює у фоновому режимі браузера, зазвичай за запитом контентного скрипта, без будь-якої веб-сторінки або доступу до будь-якого DOM. Після завершення роботи він зупиняється. В основному background.js використовується для виконання операцій високого рівня, які заборонені або неефективні для контентного скрипта.
3. Popup – це HTML-сторінка, яка з'являється у спливаючому вікні після натискання на піктограму розширення. Використання попапів не є обов'язковим, оскільки натискання на піктограму розширення можна прив'язати до функції у фоновому скрипті, яка виконає необхідний код без додаткового візуального інтерфейсу. Попапи мають обмежені розміри і зазвичай служать як простий інтерфейс для користувача. Взаємодія між попапом і фоновим скриптом здійснюється через обмін повідомленнями за допомогою функції sendMessage Chrome API.

Схема роботи всіх частин Chrome розширення представлена на рис.1.

ISSN 2219-9365

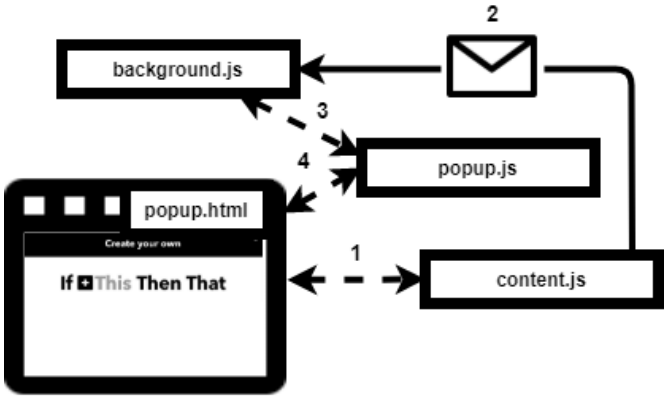

Рис. 1. Схема взаємодії частин розширення

Визначення варіантів використання. Функціональні вимоги до розширення наведені на діаграмі варіантів використання (рис.2).

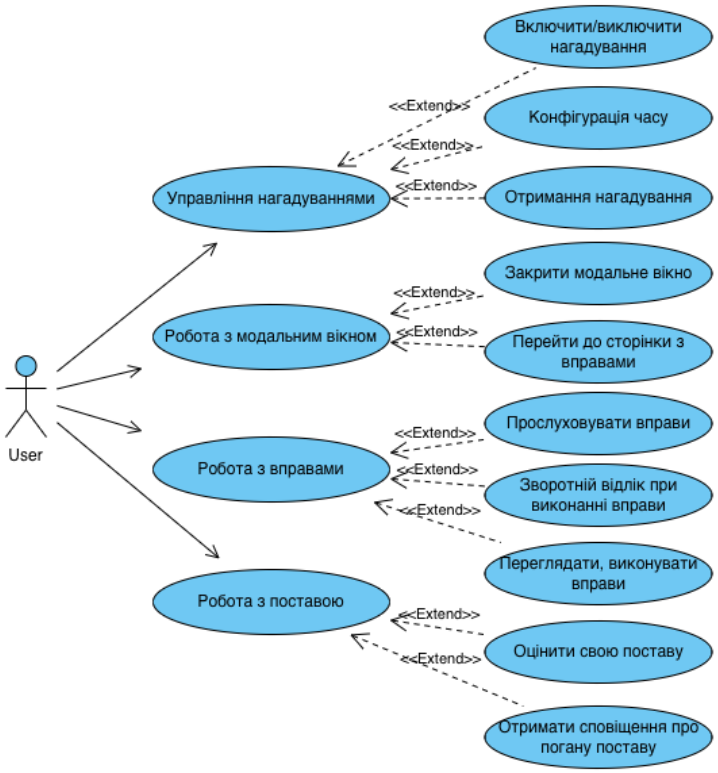

Рис. 2. Діаграма варіантів використання

Інтерфейс розширення. Для створення прототипів і кольорового дизайну розширення Chrome був використаний сервіс Figma. Після аналізу кількох кольорових варіантів було обрано основний колір.(Рис.3).

ISSN 2219-9365

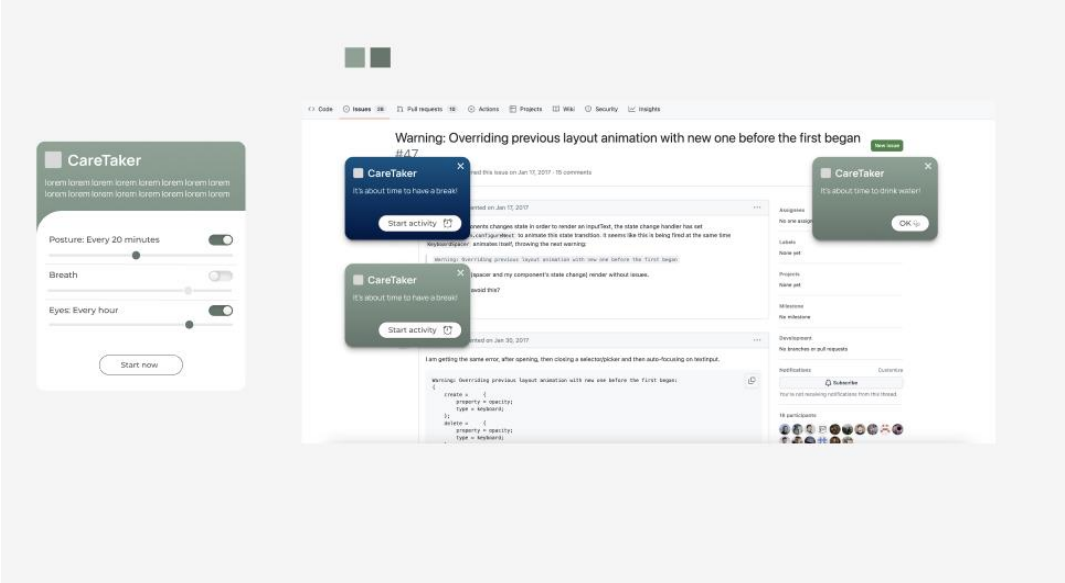

Рис. 3. Дизайн модальних вікон у Figma

При завантаженні розширення та натисканні на піктограму користувач побачить спливаюче вікно для налаштування роботи з розширенням. (рис. 4)

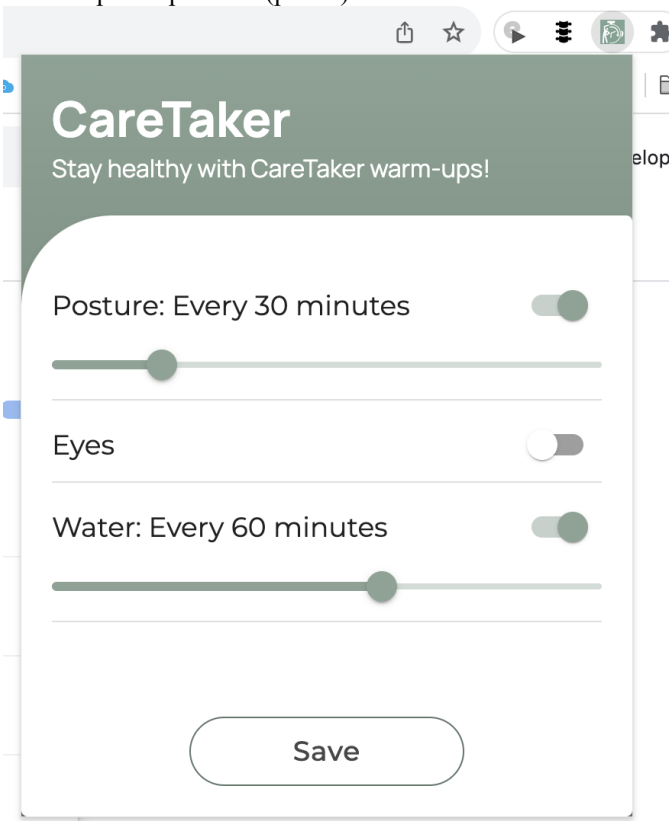

Рис. 4. Модальне вікно при натисканні на іконку

Всі поля є опційними та конфігураційними. Користувач може обрати, які пункти йому включити або виключити, а також налаштувати час за допомогою слайдера. Після натискання кнопки "зберегти" і після зазначеного часу, користувач отримає нагадування у вигляді модального вікна поверх контенту в браузері, з яким він безпосередньо взаємодіє. У нього буде вибір: закрити нагадування або перейти на сторінку з вправами, натиснувши відповідну кнопку. Це нагадування стосується постави та очей. Якщо користувач активував нагадування про воду, він отримає Chrome-сповіщення про необхідність випити воду.

Користувач, перейшовши до сторінки з вправами, зможе переглянути вебсторінки, які містять ілюстрації для вправи по покращенню постави та очей. Передбачено зворотний відлік часу протягом 2

хвилин, після якого автоматично виводиться нова вправа, але користувач також має можливість самостійно переходити до наступної вправи. Кожна вправа супроводжується іконкою звуку, яка дозволяє користувачеві прослухати інструкції з виконання вправи для більшої зручності. Після виконання вправ користувач отримає модальне вікно, з якого, натиснувши на посилання, він зможе перейти на останню відкриту вкладку в браузері та продовжити роботу. Після закриття даного модального вікна знову починається відлік часу для наступного сповіщення користувача.

До розширення Chrome додано вебсторінку, на якій користувачі можуть перевірити правильність своєї постави. (Рис. 5).

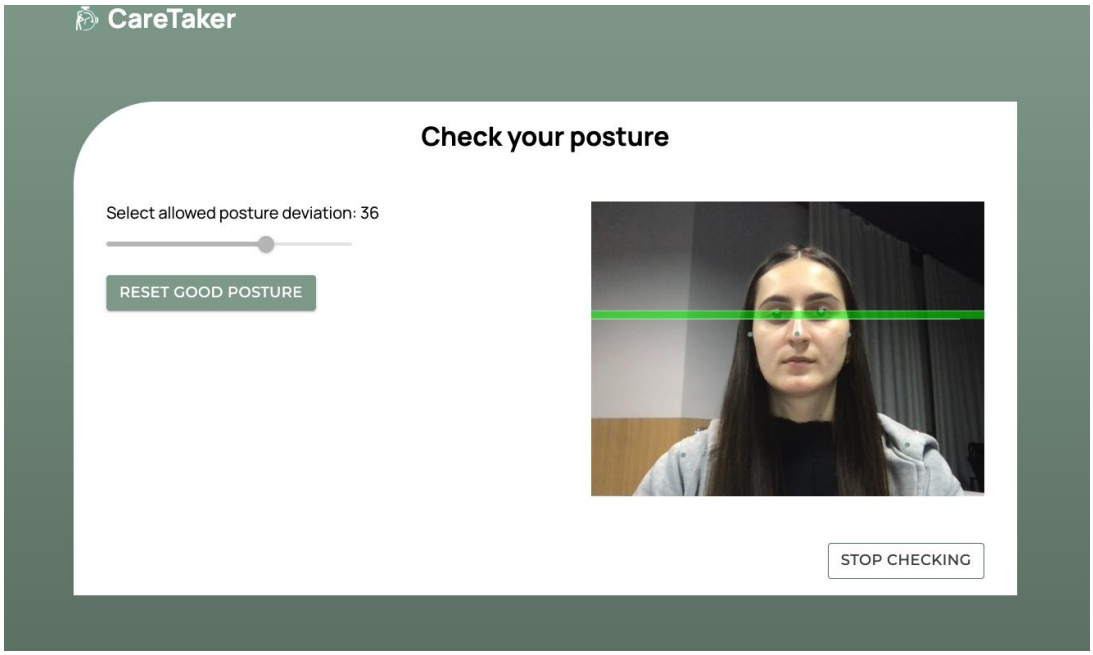

Рис. 5. Вебсторінка для перевірки правильності постави

Оскільки підтвердженої інформації про те, як визначити правильність постави за ключовими точками очей та плечей не знайдено, ця функція надається користувачеві. Користувач самостійно обирає правильну поставу та натискає кнопку "Встановити правильну поставу". Після цього він бачитиме відхилення від встановленої постави. Якщо відхилення знаходиться в межах допустимого (користувач може встановити межу самостійно від 10 до 50), він бачитиме зелений колір. Якщо відхилення перевищує встановлену межу, буде показуватися червоний колір.

Також є можливість включити або виключити перевірку постави.

Якщо користувач відкрив і почав перевірку своєї постави, а потім перейшов на інші вкладки браузера та продовжив працювати, вебсторінка буде продовжувати стежити за його поставою. Як тільки вона погіршиться, користувач отримає Chrome-сповіщення про це.

Реалізація програмного продукту за допомогою Chrome API. Для роботи більшості розширень потрібен доступ до одного або кількох API розширень Chrome. Спочатку потрібно налаштувати файл маніфесту з необхідними дозволами та вказати шлях до нашого background script. Також в маніфесті вказується шлях до content script та попапу за допомогою ключа “action” у файлі manifest.json. Оскільки в проекті використовувався bundler Vite, було створено файл manifest.js, де все описується об’єктом, а під час збірки цей файл перетворюється у формат json. (Рис. 6).

На рисунку видно, що об’єкт містить такі параметри:

1. **manifest_version** – версія маніфесту, яка може бути 2 або 3, але версія 2 вже визнана застарілою.
2. **name** – назва розширення, що відображатиметься на сторінці всіх розширень.
3. **version** – версія розширення.
4. **description** – опис розширення.
5. **permissions** – важлива частина chrome-розширення. Тут вказуються всі доступи, необхідні для повного функціонування розширення, наприклад, доступ до сховища для збереження даних користувача, доступ до вкладок, сповіщень, а також функція TTS (text to speech) для озвучування текстів.
6. **background** – вказує шлях до фонового скрипту розширення.

7. **action** – поле, що відповідає за модальне вікно розширення, яке з'являється при натисканні на іконку розширення. Це об'єкт, що включає доступ до popup.js та іконку, яка відображатиметься в панелі розширень.
8. **icons** – містить іконки, які відображатимуться в Chrome Web Store. Можна додати іконки різних розмірів.
9. **options_page** – шлях до сторінки з налаштуваннями. Зазвичай ця сторінка використовується для додаткових налаштувань розширення, у нашому випадку там розташована сторінка для перевірки постави.
10. **web_accessible_resources** – ресурси, доступні в Інтернеті. Це файли всередині розширення, до яких можуть отримати доступ веб-сторінки або інші розширення. Зазвичай використовується для показу зображень або інших ресурсів, які потрібно завантажити на веб-сторінки, але будь-який ресурс, включений у пакет розширення, можна зробити доступним у мережі.

```
const manifest = {
  manifest_version: 3,
  name: 'CareTaker',
  version: '1.0.0',
  description: 'Stay healthy with CareTaker warm-ups!',
  permissions: ['storage', 'alarms', 'tabs', 'notifications', 'tts', 'contextMenus'],
  background: {
    service_worker: 'src/pages/background/index.js',
    type: 'module',
  },
  action: {
    default_popup: 'src/pages/popup/index.html',
    default_icon: 'icon.png',
  },
  icons: {
    128: 'icon.png',
  },
  options_page: 'src/pages/options/index.html',
  content_scripts: [
    {
      matches: ['http://*/*', 'https://*/*', '<all_urls>'],
      js: ['src/pages/content/index.js'],
      // KEY for cache invalidation
      css: ['assets/css/contentStyle<KEY>.chunk.css'],
    },
  ],
  web_accessible_resources: [
    {
      resources: [
        'assets/js/*.js',
        'assets/css/*.css',
        'assets/img/*',
        'exercises.html',
      ],
      matches: ['*://*/*'],
    },
```

Рис. 6. Файл конфігурації розширення

Розглянемо деякі API, які використовувались в даній системі.

Storage API. Chrome.storage використовується для зберігання, отримання та відстеження змін даних користувача. Це API надає специфічний для розширень спосіб збереження даних і стану користувача. Воно схоже на веб-платформні API зберігання (IndexedDB і Storage), але розроблено для задоволення потреб розширень. Нижче наведено кілька основних функцій:

1. Усі контексти розширень, включаючи фоновий скрипт і скрипти вмісту (content scripts), мають доступ до API збереження.
2. Серіалізовані значення JSON зберігаються як властивості об'єкта.
3. API зберігання є асинхронним і підтримує масові операції читання та запису.
4. Дані зберігаються, навіть якщо користувач очищує кеш-пам'ять і історію веб-перегляду.

5. Збережені налаштування залишаються доступними навіть при використанні анонімного перегляду.

6. Включає ексклюзивну керовану область зберігання, доступну лише для читання, для коректного забезпечення корпоративних політик.

Щоб використовувати API збереження, необхідно оголосити дозвіл на зберігання в маніфесті розширення. API збереження поділено на такі області:

1. **storage.local**. Дані зберігаються локально та видаляються після видалення розширення. Обмеження пам'яті становить 10 МБ (5 МБ у Chrome 113 і раніших версіях), але його можна збільшити, запросивши дозвіл «unlimitedStorage». Рекомендується використовувати storage.local для зберігання більших обсягів даних.

2. **storage.sync**. Якщо синхронізацію ввімкнено, дані синхронізуються з будь-яким браузером Chrome, у якому користувач увійшов. Якщо синхронізація вимкнена, він поводиться як storage.local. Chrome зберігає дані локально, коли веб-переглядач перебуває в автономному режимі, і відновлює синхронізацію, коли він знову підключається до мережі. Обмеження становить приблизно 100 КБ, 8 КБ на елемент. Рекомендується використовувати storage.sync для збереження налаштувань користувача в синхронізованих браузерах. Для конфіденційних даних користувача рекомендується використовувати storage.session.

3. **storage.session**. Зберігає дані в пам'яті протягом сеансу браузера. За замовчуванням він недоступний для content script, але цю поведінку можна змінити, налаштувавши chrome.storage.session.setAccessLevel(). Обмеження пам'яті становить 10 МБ (1 МБ у Chrome 111 і попередніх версіях).

4. **storage.managed**. Адміністратори можуть використовувати схему та політику підприємства для налаштування параметрів розширення в керованому середовищі. Ця область зберігання є лише для читання.

Для наших потреб найкраще підійде сховище local.

Alarms API. API chrome.alarms використовується для планування періодичного запуску коду або для виконання в певний час у майбутньому. Щоб використовувати API chrome.alarms, необхідно оголосити дозвіл "alarms" у маніфесті. Поведінка alarms, коли пристрій переходить у режим сну, наразі не визначена. Alarms ніколи не спрацює раніше, ніж заплановано, але може спрацювати значно пізніше, якщо пристрій перейде в режим сну після його налаштування. API chrome.alarms має метод create, який використовується для створення alarm. Він приймає назву та налаштування часу, коли alarm має спрацювати. Також це API має можливість прослуховувати подію спрацювання alarm і виконувати певні дії після цього.

Tabs API використовується для взаємодії із системою вкладок веб-переглядача. Ви можете використовувати цей API для створення, зміни та перегрупування вкладок у браузері. API вкладок не лише пропонує функції для маніпулювання та керування вкладками, але також може визначати мову вкладки, робити знімок екрана та спілкуватися зі сценаріями вмісту вкладки. Для використання більшості функцій не потрібні дозволи. Наприклад: створення нової вкладки, перезавантаження вкладки, перехід до іншої URL-адреси тощо. Під час роботи з Tabs API розробникам слід знати про три дозволи.

1. Дозвіл tabs. Цей дозвіл не надає доступу до простору імен chrome.tabs. Натомість він дозволяє розширенню використовувати tabs.query() для отримання чотирьох конфіденційних властивостей об'єктів tabs.Tab: url, pendingUrl, title і favIconUrl.

2. Дозволи хоста. Ці дозволи дозволяють розширенню читати та запитувати чотири конфіденційні властивості відповідної вкладки. Вони також дозволяють безпосередньо взаємодіяти з вкладками за допомогою таких методів, як tabs.captureVisibleTab(), tabs.executeScript(), tabs.insertCSS() і tabs.removeCSS().

3. Дозвіл activeTab. Цей дозвіл надає розширенню тимчасовий дозвіл хоста для поточної вкладки у відповідь на дію користувача. На відміну від дозволів хоста, activeTab не викликає жодних попереджень.

Notification API використовується для створення розширених сповіщень за допомогою шаблонів і їх показу користувачам. Цей API надає функцію create, яка приймає id сповіщення, заголовок, повідомлення та шлях до зображення, яке буде показуватися у сповіщенні. Також є додаткові поля для конфігурації. Щоб використовувати Notification API, необхідно надати відповідний дозвіл у файлі manifest.json.

TTS API(text to speech) – використовується для відтворення синтезованого тексту в мову (TTS). Chrome підтримує мовлення на Windows (за допомогою SAPI 5), Mac OS X і ChromeOS, використовуючи можливості синтезу мовлення, надані операційною системою. На всіх платформах користувач може встановлювати розширення, які реєструються як альтернативні механізми мовлення. Для відтворення тексту використовується метод speak(), який може приймати додаткові параметри, такі як мова, швидкість і налаштування голосу. Також доступні методи pause(), stop() і resume().

TensorFlow модель. TensorFlow — це відкрита програмна бібліотека для машинного навчання, розроблена компанією Google для вирішення широкого спектра задач. Вона була створена для задоволення потреб Google у системах, здатних будувати та тренувати нейронні мережі для виявлення та розшифровування образів і кореляцій, подібно до того, як це роблять люди під час навчання та розуміння.

Оцінка пози — це завдання використання моделі ML (машинного навчання) [12-14] для визначення пози людини на основі зображень або відео, оцінюючи просторове розташування ключових суглобів тіла (ключових точок). Це відноситься до методів комп'ютерного зору, які виявляють людські фігури на зображеннях і відео, щоб можна було визначити, наприклад, де на зображенні знаходиться лікоть. Важливо зазначити, що оцінка пози лише визначає розташування ключових суглобів тіла і не ідентифікує, хто саме зображений на зображенні чи відео.

MoveNet — це конкретна модель для оцінки пози, яка побудована на базі TensorFlow. Вона спеціалізується на швидкому та точному визначенні ключових точок тіла на зображеннях і відео. MoveNet використовує передові алгоритми глибокого навчання для аналізу положення людського тіла в реальному часі. В якості вхідних даних приймається оброблене зображення з камери, і на його основі виводиться інформація про ключові точки. Виявлені ключові точки мають ідентифікатори частин тіла та оцінку достовірності від 0,0 до 1,0. Оцінка достовірності вказує на ймовірність того, що ключова точка дійсно знаходиться в цій позиції.

Різні суглоби тіла, виявлені моделлю оцінки пози, наведено в табл. 3:

Таблиця 3

Частини тіла за індексами

Індекс	Частина тіла	Індекс	Частина тіла
0	Ніс	9	ліве зап'ястя
1	ліве око	10	праве зап'ястя
2	праве око	11	ліве бедро
3	ліве вухо	12	праве бедро
4	праве вухо	13	ліве коліно
5	ліве плече	14	праве коліно
6	праве плече	15	ліва щиколотка
7	лівий лікоть	16	права щиколотка
8	правий лікоть		

MoveNet – це надшвидка та високоточна модель, здатна визначати 17 ключових точок тіла. Вона може працювати зі швидкістю понад 50 кадрів на секунду на сучасних ноутбуках і смартфонах.

ВИСНОВКИ З ДАНОГО ДОСЛІДЖЕННЯ І ПЕРСПЕКТИВИ ПОДАЛЬШИХ РОЗВІДОК У ДАНОМУ НАПРЯМІ

У результаті проведеного дослідження та реалізації розширення для браузера було встановлено, що на ринку відсутнє розширення, яке б комплексно задовольняло потреби користувачів у нагадуваннях про здоров'я під час роботи за комп'ютером. Було створено розширення для Google Chrome, яке містить нагадування для очей і постави, інформування про важливість підтримання водного балансу, а також можливість гнучкого налаштування часу нагадувань та активації/деактивації сповіщень.

Для реалізації розширення було обґрунтовано вибір технологій: HTML5, CSS3, TypeScript, бібліотеку React, UI бібліотеку компонентів MUI та середовище розробки Visual Studio Code. Структуру розширення становлять основні компоненти — popup, content script та background script, які взаємодіють між собою для забезпечення функціональності.

Реалізація продукту здійснена за допомогою Chrome API, використовуючи такі API як storage для зберігання даних користувача, alarms для відстеження часу, notifications для сповіщень, tts для озвучування тексту та tabs для управління вкладками браузера.

Особлива увага виділена оцінці постави користувача. Для цього було використано модель машинного навчання TensorFlow та конкретно модель MoveNet, яка забезпечує високу швидкість та точність визначення ключових точок тіла.

У перспективі система може бути вдосконалена за допомогою впровадження засобів штучного інтелекту для покращення аналізу постави та додавання нових функціональних можливостей для користувачів.

Література

1. Hongkai Chen, Mohammad Hossain.(2021). Developing a Google Chrome Extension for Detecting Phishing Emails. EPiC Series in Computing, 77. 13–22. https://doi.org/10.29007/hwp7
2. Posture Reminder. (02.07.2022). Chrome Web Store. https://chromewebstore.google.com/detail/posture-reminder/kjfkmlhcgelgmepdemdhkgioggaffibj?hl=uk&pli=1
3. Posture Minder. (6.02.2018). Chrome Web Store. https://chromewebstore.google.com/detail/postureminder/dkmkfopiihabelocpelofchappjjnpkm?hl=uk
4. Eye Care Notification. (3.12.2023). Chrome Web Store. https://chromewebstore.google.com/detail/eye-care-notification/abcpljfhdegieapcboeabpmfhmdhbooa

5. Pomodoro Work + Stretch. (1.04.2021). Chrome Web Store. https://chromewebstore.google.com/detail/pomodoro-work-+-stretch/mbimichnnpdkniapbjenpkdenmmaokcd?hl=uk+
6. Posture! Posture! Posture!. 19.11.2021). Chrome Web Store. https://chromewebstore.google.com/detail/postureposture/ekleaenhplpaiincbdkbebmeccfgmbll?hl=uk
7. Zammetti F. (2020). Modern Full-Stack Development: Using TypeScript, React, Node.js, Webpack, and Docker. Apress Berkeley, CA. https://doi.org/10.1007/978-1-4842-5738-8
8. Duldulao, D.B., Cabagnot, R.J.L. (2021). Getting Started with the Node Package Manager. In: Practical Enterprise React. Apress, Berkeley, CA. 11-19. https://doi.org/10.1007/978-1-4842-6975-6_2
9. Rack, J., Staicu, C.-A. (2023). Jack-in-the-box: An Empirical Study of JavaScript Bundling on the Web and its Security Implications. Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security. 3198–3212. https://doi.org/10.1145/3576915.3623140
10. Vite. Next Generation Frontend Tooling. (2019). https://vitejs.dev/
11. Fusco D. (2023). Large Scale Apps with Vue, Vite and TypeScript. https://www.damianofusco.com/book-sample-vue-typescript
12. Ashis Kumar Mishra, Debashis Sahoo, Ipsit Shubhankar, Isha Samal. (2024). YogaSiddhi: AI-Powered Pose Analysis using MoveNet for Yoga Refinement. International Journal of Computer Applications, 186(8). 33-39. https://doi.org/10.5120/ijca2024923427
13. Rahul Patil. (2024). Comparative Analysis of Human Pose Estimation Methods with Fall Detection and Smart Alarming System. Journal of Electrical Systems, 20(2). 2296-2305. https://doi.org/10.52783/jes.1996
14. Toshpulatov M., Lee W., Lee S., Roudsari A. (2022). Human pose, hand and mesh estimation using deep learning: a survey. J Supercomput, 78. 7616–7654. https://doi.org/10.1007/s11227-021-04184-7

References

1. Hongkai Chen, Mohammad Hossain.(2021). Developing a Google Chrome Extension for Detecting Phishing Emails. EPiC Series in Computing, 77. 13–22. https://doi.org/10.29007/hwp7
2. Posture Reminder. (02.07.2022). Chrome Web Store. https://chromewebstore.google.com/detail/posture-reminder/kjfkmlhcgelgmepdemdhkgioggaffibj?hl=uk&pli=1
3. Posture Minder. (6.02.2018). Chrome Web Store. https://chromewebstore.google.com/detail/postureminder/dkmkfopiihabelocpelofchappjjnpkm?hl=uk
4. Eye Care Notification. (3.12.2023). Chrome Web Store. https://chromewebstore.google.com/detail/eye-care-notification/abcpljfhdegieapcboeabpmfhmdhbooa
5. Pomodoro Work + Stretch. (1.04.2021). Chrome Web Store. https://chromewebstore.google.com/detail/pomodoro-work-+-stretch/mbimichnnpdkniapbjenpkdenmmaokcd?hl=uk+
6. Posture! Posture! Posture!. (19.11.2021). Chrome Web Store. https://chromewebstore.google.com/detail/postureposture/ekleaenhplpaiincbdkbebmeccfgmbll?hl=uk
7. Zammetti F. (2020). Modern Full-Stack Development: Using TypeScript, React, Node.js, Webpack, and Docker. Apress Berkeley, CA. https://doi.org/10.1007/978-1-4842-5738-8
8. Duldulao, D.B., Cabagnot, R.J.L. (2021). Getting Started with the Node Package Manager. In: Practical Enterprise React. Apress, Berkeley, CA. 11-19. https://doi.org/10.1007/978-1-4842-6975-6_2
9. Rack, J., Staicu, C.-A. (2023). Jack-in-the-box: An Empirical Study of JavaScript Bundling on the Web and its Security Implications. Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security. 3198–3212. https://doi.org/10.1145/3576915.3623140
10. Vite. Next Generation Frontend Tooling. (2019). https://vitejs.dev/
11. Fusco D. (2023). Large Scale Apps with Vue, Vite and TypeScript. https://www.damianofusco.com/book-sample-vue-typescript
12. Ashis Kumar Mishra, Debashis Sahoo, Ipsit Shubhankar, Isha Samal. (2024). YogaSiddhi: AI-Powered Pose Analysis using MoveNet for Yoga Refinement. International Journal of Computer Applications, 186(8). 33-39. https://doi.org/10.5120/ijca2024923427
13. Rahul Patil. (2024). Comparative Analysis of Human Pose Estimation Methods with Fall Detection and Smart Alarming System. Journal of Electrical Systems, 20(2). 2296-2305. https://doi.org/10.52783/jes.1996
14. Toshpulatov M., Lee W., Lee S., Roudsari A. (2022). Human pose, hand and mesh estimation using deep learning: a survey. J Supercomput, 78. 7616–7654. https://doi.org/10.1007/s11227-021-04184-7