Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)
Кафедра комп'ютерних наук
(повна назва кафедри)

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня

магістр
(назва освітнього ступеня)

на тему: Дослідження адаптивних алгоритмів регулювання когнітивного навантаження у веб-тренажері пам'яті

Виконав: студент VI курсу, групи СНнм-61
спеціальності 122 Комп'ютерні науки
(шифр і назва спеціальності)

_____ Лесик Р.В.
(підпис) (прізвище та ініціали)

Керівник _____ Никитюк В.В.
(підпис) (прізвище та ініціали)

Нормоконтроль _____ Дуда О.М.
(підпис) (прізвище та ініціали)

Завідувач кафедри _____ Боднарчук І.О.
(підпис) (прізвище та ініціали)

Рецензент _____ Загородна Н.В.
(підпис) (прізвище та ініціали)

Тернопіль
2026

Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)

Кафедра комп'ютерних наук
(повна назва кафедри)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Боднарчук І.О.
(підпис) (прізвище та ініціали)
«___»_____ 2026 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня Магістр
(назва освітнього ступеня)

за спеціальністю 122 Комп'ютерні науки
(шифр і назва спеціальності)

Студенту Лесику Роману Володимировичу
(прізвище, ім'я, по батькові)

1. Тема роботи Дослідження адаптивних алгоритмів регулювання когнітивного навантаження у веб-тренажері пам'яті

Керівник роботи Никитюк Вячеслав Вячеславович, к.т.н., доцент., доцент кафедри КН
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом ректора від « 10 » березня 2026 року № 4/9-150

2. Термін подання студентом завершеної роботи 25 травня 2026 р.

3. Вихідні дані до роботи Науково-технічні публікації (друковані та розміщені в Інтернеті)

4. Зміст роботи (перелік питань, які потрібно розробити)
Вступ. 1 Аналіз підходів до регулювання когнітивного навантаження у вебтренажерах пам'яті 2 Проєктування системи адаптивного тренажера пам'яті. 3 Реалізація адаптивного тренажера пам'яті. 4 Охорона підприємств у воєнний час. Висновки. Перелік джерел. Додатки.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, слайдів)
1 Титульна сторінка. 2 Актуальність теми дослідження. 3 Мета, об'єкт і предмет дослідження. 4 Наукова новизна та практичне значення одержаних результатів. 5 Аналіз існуючих підходів до побудови вебтренажерів пам'яті. 6 Загальна логіка організації тренувального процесу. 7 Архітектура адаптивного вебтренажера пам'яті. 8 Структура даних для збереження результатів тренувань. 9 Адаптивний алгоритм регулювання складності вправ. 10 Реалізовані тренувальні режими вебтренажера пам'яті. 11 Інтерфейс статистики та аналізу прогресу користувача. 12 Аналіз ефективності адаптивного алгоритму. 13 Основні результати виконаної роботи. 14 Висновки. 15 Завершальний слайд.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Охорона праці	Сенчишин В.С., доцент каф. МТ		
Безпека в надзвичайних ситуаціях	Теслюк В.М., проректор з адміністративно-господарської роботи та будівництва		

7. Дата видачі завдання 13 квітня 2026 р

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1.	Ознайомлення з завданням до кваліфікаційної роботи	13.04.2026	*Виконано*
2.	Підбір та опрацювання наукових публікацій, збір	13.04.2026-20.04.2026	*Виконано*
	даних по темі роботи		
3.	Виконання дослідження згідно теми кваліфікаційної	21.04.2026-03.05.2026	*Виконано*
	Роботи		
4.	Оформлення розділу «Аналіз підходів до регулювання	04.05.2026-10.05.2026	*Виконано*
	когнітивного навантаження у вебтренажерах пам’яті»		
5.	Оформлення розділу «Проєктування системи	04.05.2026-10.05.2026	*Виконано*
	адаптивного вебтренажера пам’яті»		
6.	Оформлення розділу «Реалізація адаптивного вебтренажера пам’яті»	04.05.2026-10.05.2026	*Виконано*
7.	Виконання завдання до підрозділу «Охорона праці»	27.04.2026-10.05.20	*Виконано*
8.	Виконання завдання до підрозділу «Безпека в	27.04.2026-10.05.20	*Виконано*
	надзвичайних ситуаціях»		
9.	Оформлення кваліфікаційної роботи	11.05.2026-13.05.2026	*Виконано*
10.	Нормоконтроль	14.05.2026	*Виконано*
11.	Перевірка на плагіат	18.05.2026	*Виконано*
12.	Попередній захист кваліфікаційної роботи	20.05.2026	*Виконано*
13.	Захист кваліфікаційної роботи	27.05.2026	*Виконано*

Студент _____ Лесик Р.В.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Никитюк В.В.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Дослідження адаптивних алгоритмів регулювання когнітивного навантаження у вебтренажері пам’яті // Кваліфікаційна робота освітнього рівня «Магістр» // Лесик Роман Володимирович // Тернопільський національний технічний університет імені Івана Пулюя, факультет комп’ютерно-інформаційних систем і програмної інженерії, кафедра комп’ютерних наук, група СНнм-61 // Тернопіль, 2026 // С. 79, рис. – 18, табл. – 4, додат. – 2, бібліогр. – 54.

Ключові слова: адаптивне навчання, клієнт-серверна архітектура, PHP, JavaScript, MySQL, алгоритми адаптації, метрики ефективності, вебдодаток.

Кваліфікаційна робота присвячена дослідженню та розробленню адаптивного вебтренажера пам’яті, призначеного для регулювання когнітивного навантаження користувача під час виконання інтерактивних тренувальних вправ. У роботі розглянуто сучасні підходи до побудови адаптивних навчальних вебсистем, особливості організації тренувального процесу у вебтренажерах пам’яті, методи формування та обробки результатів виконання завдань, а також використання метрик ефективності користувача для прийняття рішень щодо зміни складності вправ.

У межах дослідження обґрунтовано архітектуру вебдодатка, спроєктовано структуру зберігання даних і розроблено алгоритм адаптивного регулювання складності на основі точності відповідей, часу реакції та стабільності результатів користувача. Реалізовано вебтренажер пам’яті з інтерактивними вправами, підсистемою автентифікації, збереженням результатів тренувань, статистичним аналізом прогресу та автоматичною зміною параметрів складності. Отримані результати підтверджують доцільність застосування адаптивного підходу для персоналізації тренувального процесу та підвищення ефективності взаємодії користувача із системою.

ANNOTATION

Study of Adaptive Algorithms for Cognitive Load Regulation in a Web-based Memory Training System // Qualification Work for the Educational Level “Master” // Lesyk Roman Volodymyrovych // Ivan Puluj Ternopil National Technical University, Faculty of Computer Information Systems and Software Engineering, Department of Computer Sciences, Group СНнм-61 // Ternopil, 2026 // P. 79, fig. – 18, tabl. – 4, append. – 2, bibliogr. – 54.

Key words: adaptive learning, client-server architecture, PHP, JavaScript, MySQL, adaptation algorithms, performance metrics, web application.

The qualification work is devoted to the research and development of an adaptive web-based memory trainer intended to regulate the user’s cognitive load during the performance of interactive training exercises. The work considers modern approaches to the construction of adaptive educational web systems, the organization of the training process in web-based memory trainers, methods for collecting and processing task performance results, as well as the use of user performance metrics for making decisions on exercise difficulty adjustment.

Within the study, the architecture of the web application was substantiated, the data storage structure was designed, and an adaptive difficulty regulation algorithm was developed based on response accuracy, reaction time, and the stability of user results. A web-based memory trainer was implemented with interactive exercises, an authentication subsystem, training result storage, statistical progress analysis, and automatic adjustment of difficulty parameters. The obtained results confirm the feasibility of applying an adaptive approach to personalize the training process and improve the effectiveness of user interaction with the system.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

API – Application Programming Interface (інтерфейс програмування застосунків).

AJAX – Asynchronous JavaScript and XML (асинхронна взаємодія клієнта із сервером).

CRUD – Create, Read, Update, Delete (базові операції роботи з даними).

CSS – Cascading Style Sheets (каскадні таблиці стилів).

DOM – Document Object Model (об’єктна модель документа).

HTML – HyperText Markup Language (мова розмітки гіпертексту).

HTTP – HyperText Transfer Protocol (протокол передачі гіпертексту).

HTTPS – HyperText Transfer Protocol Secure (захищений протокол передачі даних).

IDE – Integrated Development Environment (інтегроване середовище розробки).

IP – Internet Protocol (інтернет-протокол).

JSON – JavaScript Object Notation (текстовий формат обміну даними).

JVM – Java Virtual Machine (віртуальна машина Java).

LAN – Local Area Network (локальна комп’ютерна мережа).

MVC – Model–View–Controller (архітектурний шаблон побудови вебдодатків).

MySQL – My Structured Query Language (система керування базами даних).

OOP – Object-Oriented Programming (об’єктно-орієнтоване програмування).

PHP – PHP: Hypertext Preprocessor (серверна мова програмування).

RAM – Random Access Memory (оперативна пам’ять).

REST – Representational State Transfer (архітектурний стиль вебсервісів).

SQL – Structured Query Language (мова структурованих запитів).

UI – User Interface (інтерфейс користувача).

URL – Uniform Resource Locator (уніфікований локатор ресурсу).

USB – Universal Serial Bus (універсальна послідовна шина).

UX – User Experience (користувацький досвід).

VPN – Virtual Private Network (віртуальна приватна мережа).

WWW – World Wide Web (всесвітня павутина).

XML – eXtensible Markup Language (розширювана мова розмітки).

БД – база даних.

ІС – інформаційна система.

ООП – об’єктно-орієнтоване програмування.

ПЗ – програмне забезпечення.

ПП – програмний продукт.

СУБД – система управління базами даних.

ТЗ – технічне завдання.

ЗМІСТ

ВСТУП

Актуальність теми. У сучасних умовах активного розвитку інформаційних технологій та зростання ролі цифрових освітніх ресурсів особливої актуальності набувають системи, орієнтовані на розвиток когнітивних здібностей людини. Веборієнтовані тренажери пам’яті стають популярним інструментом для тренування уваги, швидкості мислення та короткочасної пам’яті, оскільки забезпечують доступність, інтерактивність та можливість індивідуалізації навчального процесу. Разом із цим ефективність таких систем значною мірою залежить від здатності адаптувати складність завдань відповідно до поточного стану користувача.

Мета і задачі дослідження. Метою даної кваліфікаційної роботи освітнього рівня «Магістр» є розроблення та дослідження адаптивного вебтренажера пам’яті, у якому регулювання складності тренувальних вправ здійснюється на основі аналізу точності відповідей, часу реакції та стабільності результатів користувача.

Для досягнення поставленої мети необхідно виконати такі завдання:

– проаналізувати сучасні підходи до побудови адаптивних навчальних вебсистем і регулювання когнітивного навантаження;

– дослідити особливості функціонування вебтренажерів пам’яті та використання метрик ефективності користувача;

– обґрунтувати архітектуру адаптивного вебтренажера пам’яті та спроєктувати структуру його основних компонентів;

– розробити модель зберігання результатів тренувань і засоби їх подальшого аналізу;

– реалізувати адаптивний алгоритм зміни складності вправ на основі точності виконання, часу реакції та стабільності результатів;

– створити вебдодаток із підтримкою інтерактивних тренувальних вправ, автентифікації користувача та відображення статистики прогресу;

– провести перевірку працездатності системи та проаналізувати ефективність застосування адаптивного алгоритму.

Об’єкт дослідження: процеси організації та адаптивного регулювання тренувального процесу у веборієнтованих системах розвитку пам’яті.

Предмет дослідження: методи адаптації складності завдань, алгоритми аналізу результатів користувача та засоби реалізації інтерактивних вебтренажерів пам’яті.

Наукова новизна одержаних в кваліфікаційній роботі результатів полягає у вдосконаленні підходу до адаптивного регулювання складності тренувальних завдань шляхом поєднання показників точності виконання, швидкості реакції та стабільності результатів користувача для динамічного налаштування параметрів вебтренажера пам’яті.

Практичне значення одержаних результатів. Розроблений вебдодаток адаптивного тренажера пам’яті може використовуватись у навчальних, персональних та дослідницьких цілях для розвитку когнітивних здібностей користувачів. Система забезпечує інтерактивне виконання вправ, автоматичне регулювання складності та формування статистики результатів, що дозволяє підвищити ефективність тренувального процесу та спростити аналіз прогресу користувача.

Апробація результатів магістерської роботи. Основні результати проведених досліджень апробовано на XIV науково-технічній конференції «Актуальні задачі сучасних технологій» та XIII міжнародній студентській науково-практичній конференції молодих учених і студентів «Інформаційні моделі, системи та технології» Тернопільського національного технічного університету імені Івана Пулюя (м. Тернопіль, 2025 р.).

Публікації. Основні результати кваліфікаційної роботи опубліковано у двох працях конференції (додатки А1 та А2).

Структура й обсяг кваліфікаційної роботи. Кваліфікаційна робота складається зі вступу, чотирьох розділів, висновків, списку літератури з 54 найменувань та 2 додатків. Загальний обсяг кваліфікаційної роботи складає 79 сторінок, з них 62 сторінки основного тексту, який містить 18 рисунків та 4 таблиць.

1 АНАЛІЗ ПІДХОДІВ ДО РЕГУЛЮВАННЯ КОГНІТИВНОГО НАВАНТАЖЕННЯ У ВЕБТРЕНАЖЕРАХ ПАМ'ЯТІ

1.1 Загальні принципи побудови адаптивних навчальних систем

Сучасний етап розвитку інформаційних технологій характеризується активним впровадженням інтелектуальних систем у сферу освіти та самонавчання. Веборієнтовані платформи поступово переходять від статичного подання матеріалу до інтерактивних моделей взаємодії, у яких ключову роль відіграє адаптація до індивідуальних особливостей користувача. У цьому контексті особливого значення набувають адаптивні навчальні вебсистеми, здатні змінювати параметри навчального процесу залежно від поведінки користувача та результатів його діяльності.

Основою таких систем є принцип персоналізації, який передбачає врахування індивідуальних характеристик користувача. До них належать рівень підготовки, швидкість сприйняття інформації, здатність до запам'ятовування та концентрації уваги. Реалізація цього принципу дозволяє формувати навчальний процес, що максимально відповідає можливостям конкретного користувача, що є особливо важливим у задачах тренування пам'яті, де надмірне або недостатнє навантаження може суттєво впливати на ефективність [10].

Важливим компонентом адаптивних вебсистем є модель користувача, яка формується на основі накопичених даних про його взаємодію з системою. Така модель включає показники успішності виконання завдань, швидкість реакції, стабільність результатів та інші параметри, що характеризують когнітивні можливості. На основі цієї інформації система приймає рішення щодо зміни складності завдань або структури навчального процесу.

Ще одним ключовим принципом є динамічна адаптація контенту, яка реалізується шляхом зміни параметрів завдань у режимі реального часу. Це може включати варіювання кількості елементів для запам'ятовування, часу на виконання або рівня складності стимулів [11]. Такий підхід дозволяє

підтримувати оптимальний рівень когнітивного навантаження, уникаючи як перевантаження, так і недостатньої стимуляції.

Окрім цього, адаптивні системи передбачають наявність механізмів зворотного зв’язку, які забезпечують інформування користувача про результати його діяльності. Це може бути як миттєва оцінка виконання окремого завдання, так і узагальнена інформація про прогрес у межах сесії або більш тривалого періоду. Наявність такого зворотного зв’язку сприяє підвищенню мотивації та дозволяє користувачу краще розуміти власні досягнення.

З технічної точки зору адаптивні вебсистеми реалізуються з використанням клієнт-серверної архітектури, що забезпечує розподіл обчислювального навантаження та можливість масштабування. Обробка даних користувача, формування моделей та реалізація алгоритмів адаптації можуть виконуватися на серверній стороні, тоді як клієнтська частина відповідає за інтерактивну взаємодію та візуалізацію результатів, що безпосередньо пов’язано з організацією тренувального процесу у вебтренажерах пам’яті [12].

1.1.1 Організація тренувального процесу у тренажерах пам’яті

Організація тренувального процесу у вебтренажерах пам’яті є ключовим елементом, що визначає ефективність розвитку когнітивних здібностей користувача. На відміну від традиційних навчальних систем, такі тренажери орієнтовані не лише на передачу інформації, а на формування та вдосконалення функцій пам’яті через систематичне виконання спеціалізованих вправ. Це зумовлює необхідність побудови чіткої та логічно узгодженої структури взаємодії, яка враховує як послідовність завдань, так і індивідуальні особливості користувача [10].

Тренувальний процес зазвичай організовується у вигляді окремих сесій, кожна з яких включає набір вправ певного типу або рівня складності. Такий підхід дозволяє структурувати процес навчання, забезпечити його поетапність та поступове зростання складності. Початковий етап часто передбачає проведення діагностичного тестування, метою якого є визначення базового рівня

когнітивних здібностей користувача. На основі отриманих результатів система формує початкові параметри тренування, що надалі можуть змінюватися [13].

Загальна логіка організації тренувального процесу представлена на рисунку 1.1.

Рисунок 1.1 – Структура тренувального процесу у вебтренажері пам’яті

Як показано на рисунку 1.1, процес тренування має циклічний характер. На першому етапі виконується діагностика, під час якої визначається початковий рівень користувача. Далі користувач переходить до виконання вправ, які формуються відповідно до встановленого рівня складності. Усі результати фіксуються системою, після чого здійснюється їх аналіз із використанням відповідних метрик.

Етап аналізу передбачає оцінювання таких параметрів, як точність виконання, час реакції, кількість помилок та стабільність результатів. На основі цього формується узагальнене уявлення про ефективність діяльності користувача. Наступним кроком є адаптація параметрів тренування, що може включати зміну складності завдань, їх тривалості або структури. Після цього цикл повторюється у межах наступної сесії, що забезпечує безперервний процес навчання [13].

Важливою характеристикою тренувального процесу є різноманітність типів завдань. Вебтренажери пам’яті можуть включати вправи на запам’ятовування числових або символьних послідовностей, відтворення візуальних образів, встановлення асоціативних зв’язків, а також задачі на просторову пам’ять і швидкість реакції. Комбінування різних типів вправ дозволяє задіяти різні когнітивні механізми та забезпечує більш комплексний розвиток пам’яті.

Побудова тренувального процесу також передбачає визначення параметрів виконання завдань, серед яких особливе значення мають тривалість сесії, кількість повторень та інтервали між вправами. Оптимальне налаштування цих параметрів є критичним для підтримання ефективного рівня когнітивного навантаження. Перевищення допустимого рівня складності або інтенсивності може призводити до втоми та зниження продуктивності, тоді як недостатнє навантаження не забезпечує належного тренувального ефекту [13].

Окрему роль у структурі тренувального процесу відіграють мотиваційні механізми. У сучасних вебтренажерах активно використовуються елементи гейміфікації, зокрема система балів, рівнів, досягнень та візуальних індикаторів прогресу. Такі елементи сприяють підвищенню залученості користувача та стимулюють регулярне виконання вправ, що є важливою умовою ефективного тренування пам’яті [14].

Організація тренувального процесу, що поєднує циклічну структуру, варіативність завдань та адаптивне налаштування параметрів, безпосередньо пов’язана з необхідністю детального збору, фіксації та подальшої обробки результатів виконання вправ, що розглядається у наступному підпункті.

1.1.2 Формування та обробка результатів виконання тренувальних завдань

Формування результатів у вебтренажерах пам’яті відбувається безпосередньо в процесі взаємодії користувача з системою та передбачає фіксацію всіх значущих подій, що виникають під час виконання тренувальних вправ. На відміну від традиційних підходів оцінювання, у вебсередовищі результати формуються як сукупність дрібних дій користувача, які реєструються

в реальному часі та можуть бути використані для подальшої інтерпретації його поведінки.

До таких дій належать вибір відповіді, послідовність натискань, затримки між діями, повторні спроби виконання завдання, а також реакція на зміну умов. Фіксація подібних подій дозволяє отримати більш детальне уявлення про процес виконання вправи, а не лише про її кінцевий результат [15]. Це особливо важливо для тренажерів пам’яті, де значення має не тільки правильність відповіді, а й спосіб її досягнення.

З технічної точки зору формування результатів реалізується через механізми обробки подій на клієнтській стороні, які передаються на сервер для подальшого збереження. Такий підхід забезпечує мінімальні затримки та дозволяє фіксувати навіть короткочасні реакції користувача. У деяких випадках частина обробки може виконуватися локально, що зменшує навантаження на сервер та підвищує швидкодію системи.

Процес обробки результатів на початковому рівні включає очищення та впорядкування отриманих даних. Це необхідно для усунення випадкових або некоректних значень, які можуть виникати через технічні збої або нестабільність з’єднання. Після цього дані приводяться до єдиного формату, що дозволяє забезпечити їх подальше використання у системі [16].

Окрему роль відіграє синхронізація результатів у випадку роботи користувача на різних пристроях або при перериванні сесії. Система повинна забезпечувати коректне відновлення стану тренування та уникати втрати даних. Це досягається шляхом періодичного збереження проміжних результатів і використання механізмів ідентифікації сесій [17].

Важливим аспектом є також часове маркування подій, яке дозволяє відстежувати послідовність виконання дій та аналізувати поведінку користувача у часовому вимірі. Наявність таких даних створює передумови для більш глибокого аналізу взаємодії, зокрема визначення моментів зниження уваги або перевантаження [18].

У процесі обробки результатів також може здійснюватися первинна інтерпретація отриманих даних, яка полягає у визначенні базових характеристик

виконання завдання без їх узагальнення. Це дозволяє системі оперативно реагувати на дії користувача та змінювати параметри поточного тренування, що безпосередньо пов’язано з подальшим аналізом існуючих вебтренажерів пам’яті та реалізованих у них підходів до роботи з результатами.

1.2 Аналіз існуючих вебтренажерів пам’яті

Сучасні вебтренажери пам’яті представлені як спеціалізовані онлайн-платформи та сервіси, що спрямовані на розвиток короткочасної, робочої та довготривалої пам’яті шляхом виконання структурованих когнітивних вправ. Такі системи активно використовуються як у сфері саморозвитку, так і в освітніх цілях, а їх функціональність базується на поєднанні психологічних методик тренування пам’яті та інтерактивних вебтехнологій.

Одним із найвідоміших прикладів є платформа Lumosity, яка пропонує набір міні-ігор, спрямованих на тренування пам’яті, уваги та швидкості мислення. Користувачі виконують короткі вправи, що адаптуються за рівнем складності залежно від попередніх результатів. Система формує персональні “тренувальні профілі” та надає щоденні рекомендації щодо тренувань, що дозволяє підтримувати регулярність занять [19].

Іншим поширеним рішенням є CogniFit, яка орієнтована на більш детальну діагностику когнітивних здібностей. Платформа спочатку проводить тестування користувача, після чого формує індивідуальну програму тренувань [20]. Особливістю є використання великої кількості когнітивних параметрів, включаючи пам’ять, концентрацію та швидкість обробки інформації, що дозволяє отримати більш комплексну оцінку стану користувача.

Також популярною є система BrainHQ, яка базується на наукових дослідженнях у сфері нейропластичності. Вправи в ній спрямовані на поступове покращення когнітивної продуктивності через регулярні короткі тренування. Особливістю є чітка структурованість завдань та акцент на повторюваності, що дозволяє відстежувати прогрес у динаміці [21].

Окремо можна виділити Peak, яка поєднує елементи тренування пам’яті з гейміфікацією. Система пропонує щоденні набори вправ, аналізує результати користувача та порівнює їх із середніми показниками інших користувачів [22]. Це створює додаткову мотивацію до регулярного використання платформи.

Для узагальнення основних характеристик існуючих вебтренажерів пам’яті доцільно подати порівняння у вигляді таблиці.

Таблиця 1.1 – Порівняння сучасних вебтренажерів пам’яті

Платформа	**Основна спрямованість**	**Тип вправ**	**Особливості адаптації**	**Додаткові функції**
Lumosity	Загальний розвиток когнітивних здібностей	Міні-ігри	Динамічна зміна складності	Персональні звіти
CogniFit	Когнітивна діагностика і тренування	Тестові та тренувальні вправи	Адаптація на основі діагностики	Когнітивний профіль
BrainHQ	Науково обґрунтоване тренування мозку	Структуровані вправи	Поступове ускладнення	Аналіз прогресу
Peak	Гейміфіковане тренування	Ігрові завдання	Базова адаптація	Порівняння з іншими користувачами

Аналіз представлених систем показує, що більшість сучасних вебтренажерів використовує гейміфікований або напівнауковий підхід до організації тренувального процесу. При цьому ключовим елементом є регулярність виконання вправ та поступове підвищення складності, що дозволяє користувачу поступово розвивати когнітивні навички.

Водночас у більшості розглянутих систем адаптація складності реалізується переважно на рівні зміни загального рівня вправ або вибору наступного завдання. Глибший аналіз індивідуальних стратегій користувача, характеру помилок або динаміки когнітивного навантаження використовується обмежено.

Окремою особливістю є те, що більшість платформ орієнтована на масового користувача, тому їхні алгоритми адаптації є узагальненими та не завжди враховують індивідуальні відмінності у когнітивних процесах. Це створює передумови для розробки більш точних та гнучких вебтренажерів пам’яті з розширеними механізмами аналізу результатів і регулювання складності тренувальних завдань.

1.3 Підходи до структурування та узагальнення результатів тренувань у вебсистемах

Ефективне функціонування адаптивних вебтренажерів пам’яті значною мірою залежить від способів організації та узагальнення результатів тренувальної діяльності користувача. Якщо на попередньому етапі здійснюється фіксація окремих подій, то на цьому рівні відбувається їх впорядкування, логічне групування та підготовка до подальшого використання в алгоритмах адаптації.

Одним із базових підходів є ієрархічне структурування даних, при якому результати організовуються за кількома рівнями узагальнення. Як правило, виділяються рівні окремої дії, вправи, сесії та користувача. Така багаторівнева організація дозволяє розглядати результати як у межах виконання конкретного завдання, так і в більш широкому часовому контексті [23]. Завдяки цьому забезпечується можливість гнучкого доступу до даних залежно від поставлених задач аналізу. Приклад ієрархічного структурування даних продемонстровано на рис. 1.2.

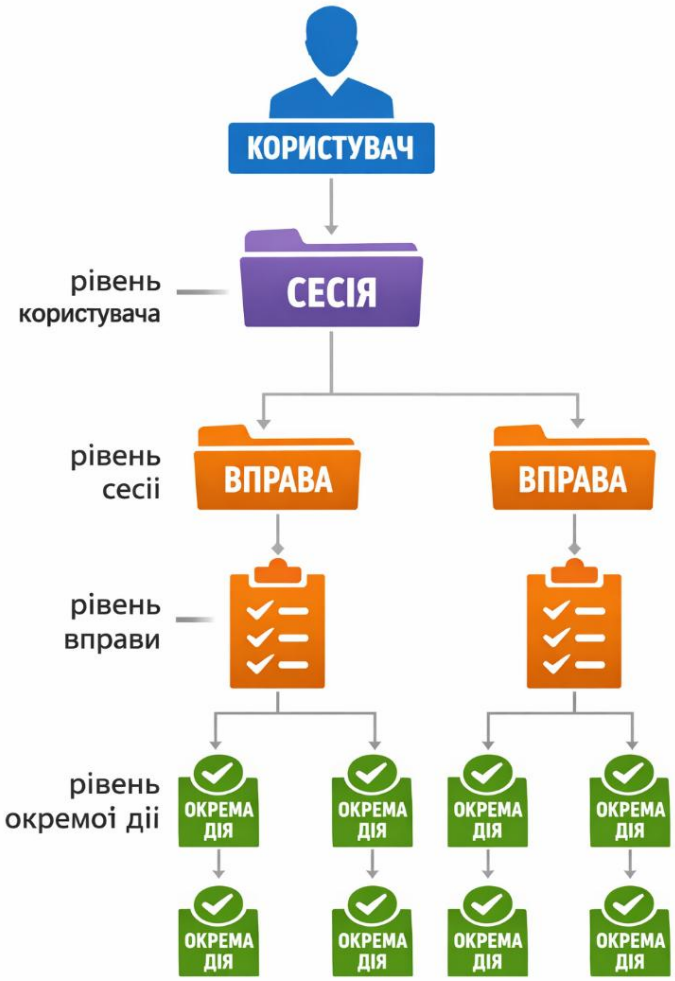


Рис. 1.2 - Ієрархічна структура бази даних у вебтренажері пам’яті

Поряд із цим широко використовується табличний підхід до структурування, який реалізується за допомогою баз даних. У цьому випадку результати зберігаються у вигляді записів із чітко визначеними полями, що описують параметри виконання завдань. Такий формат забезпечує ефективність виконання запитів, спрощує обробку великих обсягів даних і дозволяє інтегрувати результати з іншими компонентами системи.

Важливим елементом є агрегування даних, яке передбачає об’єднання великої кількості окремих результатів у більш узагальнені представлення. Це може здійснюватися на рівні окремих вправ, сесій або часових інтервалів. Агрегування дозволяє зменшити обсяг інформації, що підлягає обробці, та забезпечити компактне представлення результатів без втрати їх змістовності [25].

Окрему роль відіграє часове впорядкування результатів, що дозволяє зберігати послідовність виконання завдань і аналізувати зміни у поведінці користувача з часом. Такий підхід забезпечує можливість відстеження розвитку навичок та виявлення характерних тенденцій у процесі тренування, що є важливим для подальшого налаштування навчального процесу [18].

Для підвищення ефективності обробки також застосовуються методи групування даних за певними ознаками, такими як тип завдання, рівень складності або умови виконання. Це дозволяє формувати більш структуроване уявлення про результати та спрощує їх подальший аналіз у межах системи.

У практичних реалізаціях вебтренажерів пам’яті, як правило, використовується комбінований підхід, що поєднує кілька способів структурування. Наприклад, ієрархічна логіка організації даних може поєднуватися з табличною формою зберігання, що дозволяє досягти балансу між логічною зрозумілістю та технічною ефективністю.

Організація та узагальнення результатів тренувань створює основу для їх подальшого використання у механізмах зміни параметрів навчального процесу, що безпосередньо пов’язано з реалізацією методів адаптації складності в інтерактивних системах.

1.4 Методи адаптації складності в інтерактивних системах

Адаптація складності є одним із ключових механізмів забезпечення ефективної взаємодії користувача з інтерактивними системами, зокрема вебтренажерами пам’яті. Вона спрямована на підтримання оптимального рівня когнітивного навантаження шляхом динамічної зміни параметрів завдань відповідно до поточних можливостей користувача. Реалізація таких механізмів дозволяє уникнути як перевантаження, так і недостатньої стимуляції, що безпосередньо впливає на результативність тренування [13].

Одним із найпростіших підходів є порогова адаптація, яка базується на заздалегідь визначених умовах переходу між рівнями складності. У цьому випадку система змінює параметри завдань при досягненні певного рівня

успішності або, навпаки, при зниженні результатів. Наприклад, при стабільному виконанні вправ без помилок складність може поступово підвищуватися шляхом збільшення кількості елементів для запам’ятовування або скорочення часу на відповідь. Такий підхід є простим у реалізації, проте має обмежену гнучкість, оскільки не враховує детальні особливості поведінки користувача.

Більш гнучким є адаптивний підхід, заснований на безперервному аналізі результатів. У цьому випадку зміна складності відбувається не дискретно, а поступово, з урахуванням поточних показників діяльності. Наприклад, система може не лише змінювати рівень завдань, а й варіювати окремі параметри, такі як швидкість подання інформації, тривалість експозиції стимулів або кількість повторень. Це дозволяє більш точно підлаштовувати навчальний процес під конкретного користувача.

Окрему групу становлять методи, що базуються на моделюванні стану користувача. У таких системах формується уявлення про рівень підготовки, стійкість уваги та здатність до обробки інформації. На основі цієї моделі система прогнозує оптимальний рівень складності завдань і відповідним чином коригує навчальний процес. Такий підхід є більш складним з точки зору реалізації, але забезпечує вищу точність адаптації.

У контексті інтерактивних систем широко застосовується параметрична адаптація, яка передбачає зміну окремих характеристик завдання без переходу між рівнями. До таких параметрів можуть належати:

– кількість об’єктів, що підлягають запам’ятовуванню;

– складність візуальних або текстових стимулів;

– час на виконання завдання;

– швидкість появи нових елементів;

– кількість відволікаючих факторів.

Такий підхід дозволяє здійснювати тонке налаштування складності без різких змін у структурі тренування, що позитивно впливає на стабільність когнітивного навантаження.

Ще одним важливим методом є адаптація темпу навчання, яка полягає у зміні швидкості переходу між завданнями або сесіями. У разі високих

результатів користувача система може прискорювати подачу матеріалу, тоді як при виникненні труднощів — уповільнювати процес і збільшувати кількість повторень. Це дозволяє забезпечити комфортний темп роботи та знизити ризик перевантаження.

У більш складних системах застосовуються алгоритмічні підходи до адаптації, які можуть включати використання статистичних моделей або методів машинного навчання. Такі алгоритми здатні враховувати багатовимірні дані про поведінку користувача та приймати рішення на основі виявлених закономірностей. Наприклад, система може визначати, які типи завдань викликають найбільші труднощі, і відповідно змінювати структуру тренування. Важливою особливістю адаптації є її прозорість для користувача. У більшості випадків зміна складності відбувається непомітно, без явного інформування, що дозволяє зберегти природність взаємодії. Водночас деякі системи можуть надавати користувачу можливість часткового контролю, наприклад вибору рівня складності або типу завдань.

Практична реалізація методів адаптації зазвичай поєднує кілька підходів одночасно. Наприклад, порогова логіка може використовуватися разом із параметричною адаптацією, а результати аналізу — доповнюватися алгоритмічними методами. Така комбінована стратегія дозволяє досягти більшої гнучкості та ефективності системи.

Застосування різних методів адаптації складності формує основу для оцінювання ефективності діяльності користувача, що потребує використання відповідних кількісних показників і метрик.

1.5 Використання метрик ефективності користувача

Оцінювання ефективності діяльності користувача у вебтренажерах пам’яті є необхідною умовою для реалізації адаптивних механізмів та контролю якості навчального процесу. Для цього використовуються спеціалізовані метрики, які дозволяють кількісно описати результати виконання завдань і поведінку користувача під час взаємодії із системою.

Метрики ефективності формуються на основі даних, отриманих у процесі виконання вправ, і відображають різні аспекти когнітивної діяльності. Вони дозволяють оцінити не лише кінцевий результат, а й процес досягнення цього результату, що є важливим для більш точного налаштування навчального середовища [25].

Загальну структуру формування метрик ефективності доцільно подати у вигляді схеми.

Рисунок 1.3 – Процес формування метрик ефективності користувача

Як показано на рисунку 1.3, формування метрик є багатоступеневим процесом. На першому етапі відбувається збір сирих даних про взаємодію користувача з системою. Далі ці дані проходять первинну обробку, під час якої усуваються аномалії та приводяться до уніфікованого вигляду. Після цього обчислюються числові показники, які відображають різні характеристики діяльності. Отримані метрики інтерпретуються з урахуванням контексту виконання завдань і використовуються для прийняття рішень щодо подальшої адаптації [25].

У вебтренажерах пам’яті застосовується широкий спектр метрик, які можна умовно поділити на кілька груп.

До першої групи належать метрики результативності, що відображають успішність виконання завдань. Вони характеризують правильність відповідей, рівень досягнення поставленої мети та загальну ефективність виконання вправ.

Друга група включає часові метрики, які описують швидкість реакції користувача. Вони дозволяють оцінити, наскільки швидко користувач обробляє інформацію та приймає рішення. Часові показники особливо важливі для тренажерів пам’яті, оскільки швидкість відтворення інформації є одним із ключових аспектів когнітивної діяльності.

Окрему категорію становлять поведінкові метрики, що відображають особливості взаємодії користувача із системою. До них належать кількість повторних спроб, послідовність дій, частота помилок у певних типах завдань. Такі показники дозволяють глибше зрозуміти стратегію виконання вправ і виявити проблемні зони.

Також використовуються комбіновані метрики, які поєднують кілька показників у єдину оцінку. Наприклад, можуть враховуватися одночасно точність і швидкість виконання, що дозволяє отримати більш збалансовану характеристику діяльності користувача.

Таблиця 1.2 – Основні типи метрик ефективності користувача

Тип метрики	**Характеристика**
Результативні	Відображають правильність виконання завдань
Часові	Оцінюють швидкість реакції
Поведінкові	Аналізують процес виконання
Комбіновані	Поєднюють кілька показників

Використання метрик ефективності дозволяє здійснювати порівняння результатів у межах однієї сесії або між різними періодами часу. Це створює основу для виявлення тенденцій у навчанні та оцінювання змін у когнітивному стані користувача.

У практичних системах метрики застосовуються не лише для внутрішніх обчислень, а й для візуалізації результатів. Вони можуть відображатися у вигляді графіків, індикаторів або шкал прогресу, що дозволяє користувачу отримувати

зворотний зв’язок у зрозумілій формі. Це сприяє підвищенню мотивації та кращому усвідомленню власних досягнень [25].

Різноманітність метрик та способів їх використання зумовлює необхідність вибору відповідних алгоритмів, які забезпечують їх ефективне застосування у процесі адаптивного навчання.

1.6 Огляд алгоритмів адаптивного навчання

Адаптивне навчання у вебтренажерах пам’яті реалізується за допомогою спеціалізованих алгоритмів, які забезпечують автоматичне налаштування параметрів навчального процесу відповідно до індивідуальних характеристик користувача. Такі алгоритми використовують дані про результати виконання завдань, поведінку користувача та динаміку змін показників для визначення оптимальної складності та структури тренування [13].

Найпростішими з точки зору реалізації є алгоритми на основі правил, які функціонують за принципом умовних переходів. У таких системах заздалегідь визначаються порогові значення показників, при досягненні яких змінюється рівень складності завдань. Наприклад, якщо користувач демонструє стабільно високі результати, система підвищує складність, а у разі погіршення показників — знижує її. Цей підхід є ефективним для базових систем, однак має обмежену гнучкість через відсутність глибокого аналізу поведінки користувача.

Більш складними є алгоритми, що базуються на статистичних моделях. Вони враховують ймовірнісні характеристики виконання завдань і дозволяють оцінювати рівень підготовки користувача з урахуванням невизначеності. Наприклад, можуть використовуватися моделі, що визначають ймовірність правильної відповіді залежно від складності завдання та попередніх результатів. Такий підхід забезпечує більш точне налаштування навчального процесу.

Окрему групу становлять алгоритми, засновані на байєсівських методах. Вони дозволяють оновлювати оцінку знань користувача у процесі виконання завдань, використовуючи нові дані. Перевагою цього підходу є можливість поступового уточнення моделі користувача без необхідності накопичення

великого обсягу даних. Це робить його ефективним у системах, де важлива швидка адаптація.

У сучасних інтерактивних системах також застосовуються алгоритми машинного навчання, які дозволяють автоматично виявляти закономірності у поведінці користувачів. Такі алгоритми можуть використовуватися для класифікації користувачів за рівнем підготовки, прогнозування результатів виконання завдань або визначення оптимальної стратегії навчання. Вони здатні враховувати велику кількість параметрів і забезпечують високий рівень гнучкості.

Одним із перспективних напрямів є використання методів навчання з підкріпленням, у яких система поступово навчається обирати оптимальні дії (наприклад, зміну складності завдання) на основі отриманого досвіду. У цьому випадку кожна дія системи оцінюється з точки зору її впливу на результат навчання, що дозволяє поступово покращувати стратегію адаптації.

Таблиця 1.3 – Порівняння алгоритмів адаптивного навчання

Тип алгоритму	**Характеристика**	**Переваги**	**Обмеження**
Правиловий	Заснований на умовах і порогах	Простота реалізації	Низька гнучкість
Статистичний	Використання ймовірнісних моделей	Точність оцінювання	Потребує даних
Байєсівський	Оновлення оцінок у процесі	Адаптивність	Складність реалізації
Машинне навчання	Виявлення закономірностей	Висока гнучкість	Високі обчислювальні витрати
Навчання з підкріпленням	Оптимізація дій системи	Самонавчання	Складність налаштування

У практичних системах адаптивного навчання часто застосовується комбінування кількох алгоритмічних підходів. Наприклад, базова логіка може реалізовуватися за допомогою правилових алгоритмів, тоді як уточнення параметрів здійснюється із використанням статистичних або машинних методів. Такий підхід дозволяє поєднати простоту реалізації з гнучкістю адаптації.

Вибір конкретного алгоритму або їх комбінації залежить від складності системи, обсягу доступних даних та вимог до точності адаптації, що формує основу для узагальнення отриманих результатів у межах першого розділу.

1.7 Висновок до першого розділу

У першому розділі кваліфікаційної роботи виконано аналітичне дослідження підходів до регулювання когнітивного навантаження у вебтренажерах пам’яті, що дозволило визначити основні принципи побудови адаптивних навчальних систем та особливості їх функціонування у вебсередовищі. Розглянуто організацію тренувального процесу, яка базується на структурованому виконанні вправ і необхідності динамічного налаштування параметрів навчання відповідно до результатів користувача.

Встановлено, що формування та обробка результатів тренувальних завдань є ключовою основою для оцінювання стану користувача та подальшої адаптації складності. При цьому важливу роль відіграють не лише кінцеві показники, а й часові характеристики та поведінкові особливості під час виконання завдань.

Аналіз існуючих вебтренажерів пам’яті показав, що більшість рішень використовує спрощені механізми адаптації, які не повною мірою враховують індивідуальні когнітивні особливості користувача, що обумовлює потребу у більш гнучких підходах до регулювання складності.

Розглянуті методи адаптації складності охоплюють як прості правилові алгоритми, так і більш складні підходи на основі машинного навчання, що дозволяє забезпечити більш точне налаштування навчального процесу відповідно до можливостей користувача.

Використання метрик ефективності користувача та алгоритмів адаптивного навчання створює основу для побудови систем, здатних динамічно реагувати на зміни результатів діяльності та коригувати навчальні сценарії в реальному часі, підвищуючи ефективність тренувального процесу.

2 ПРОЄКТУВАННЯ СИСТЕМИ АДАПТИВНОГО ВЕБТРЕНАЖЕРА ПАМ'ЯТІ

2.1 Архітектурні підходи до побудови вебтренажера

Архітектура вебтренажера пам'яті визначає загальні принципи організації програмної системи, способи взаємодії її компонентів та підходи до розподілу відповідальності між рівнями обробки даних. Оскільки адаптивний вебтренажер пам'яті передбачає постійну взаємодію користувача із системою та динамічну зміну складності завдань, архітектура повинна забезпечувати швидку обробку подій, масштабованість і зручність розширення функціоналу.

У сучасній веброзробці найчастіше використовуються кілька базових архітектурних підходів, кожен із яких має свої переваги та обмеження залежно від складності системи.

Монолітна архітектура передбачає реалізацію всієї системи як єдиного програмного модуля, де інтерфейс, бізнес-логіка та робота з даними знаходяться в одному середовищі [26]. Такий підхід є простим у початковій розробці, однак ускладнює масштабування, тестування та внесення змін, особливо у системах з великою кількістю взаємопов'язаних функцій, таких як адаптивні алгоритми тренування.

Клієнт–серверна архітектура розділяє систему на дві основні частини: клієнтську (frontend) та серверну (backend). Клієнт відповідає за взаємодію з користувачем, а сервер — за обробку даних, бізнес-логіку та збереження інформації. Такий підхід є більш гнучким і дозволяє незалежно розвивати інтерфейс та алгоритмічну частину системи [26].

Багаторівнева (layered) архітектура передбачає поділ системи на логічні рівні: рівень представлення, рівень бізнес-логіки та рівень доступу до даних. Це дозволяє чітко розмежувати відповідальність компонентів і спрощує підтримку програмного коду [27].

Подієво-орієнтована архітектура (event-driven) використовується у системах, де важлива реакція на дії користувача в реальному часі. У таких

системах кожна дія розглядається як подія, яка ініціює обробку та зміну стану системи. Для вебтренажера пам’яті це особливо актуально, оскільки кожен тріал користувача впливає на подальшу складність завдань [28].

Мікросервісна архітектура передбачає розподіл системи на набір незалежних сервісів, кожен з яких виконує окрему функцію. Хоча цей підхід є найбільш масштабованим, для даного проєкту він є надмірним, оскільки система має обмежену кількість основних функцій (тренажери, сесії, статистика, адаптація складності) [27].

Таблиця .2.1 - Порівняння архітектурних підходів

Архітектура	**Переваги**	**Недоліки**	**Доцільність для вебтренажера**
Монолітна	Простота реалізації	Складність масштабування	Низька
Клієнт–серверна	Чіткий поділ логіки, гнучкість	Потребує API-взаємодії	Висока
Багаторівнева	Структурованість, підтримка	Більша складність проєктування	Висока
Подієва	Реакція в реальному часі	Складність контролю станів	Середня/висока
Мікросервісна	Масштабованість	Надмірна складність	Низька

Для адаптивного вебтренажера пам’яті найбільш доцільним є використання клієнт–серверної архітектури з елементами багаторівневого та подієвого підходів, оскільки вони забезпечують баланс між продуктивністю, структурованістю та можливістю реалізації адаптивних алгоритмів регулювання когнітивного навантаження.

2.2 Обгрунтування вибору архітектурної системи

Вибір клієнт-серверної архітектури для розробки адаптивного вебтренажера пам’яті зумовлений необхідністю досягнення балансу між швидкістю реакції інтерфейсу та надійністю обробки аналітичних даних. Оскільки специфіка тренажера передбачає інтенсивну взаємодію користувача з

елементами на екрані, використання мови JavaScript на стороні клієнта дозволяє реалізувати ігровий цикл без затримок, пов’язаних із мережевим обміном. Це критично важливо для точності вимірювання часу реакції користувача, оскільки будь-яка затримка сервера при генерації нового завдання могла б спотворити статистичні показники когнітивного навантаження.

Розподіл системи на фронтенд та бекенд забезпечує безпеку та цілісність адаптивного алгоритму. Винесення логіки обчислення складності на сторону сервера за допомогою PHP гарантує, що процес прийняття рішень базується на достовірних даних із бази MySQL, до яких клієнт не має прямого доступу. Такий підхід унеможливлює маніпуляції з результатами сесій та дозволяє системі аналізувати динаміку прогресу користувача, спираючись на історію останніх трьох або п’яти тріалів. Завдяки цьому складність вправ підлаштовується під індивідуальні можливості людини в режимі реального часу, що є ключовою функцією адаптивного тренажера [29].

Загальний приклад взаємодії в клієнт-серверній архітектурі представлений на рисунку 2.1.

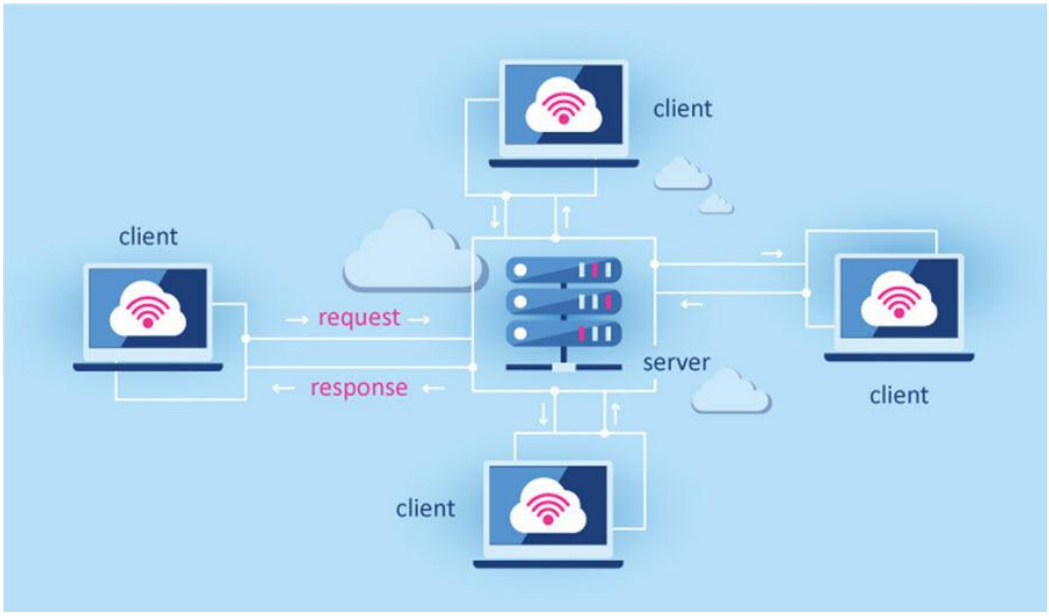


Рис. 2.1 – Схема загальної архітектури та алгоритму роботи вебтренажера

Важливим аспектом обраної архітектури є використання технології AJAX через API-запити, що дозволяє передавати результати кожного раунду на сервер

у фоновому режимі. Це забезпечує безперервність процесу тренування, оскільки користувачеві не потрібно перезавантажувати сторінку після кожної відповіді. Такий підхід не лише покращує досвід взаємодії, але й дозволяє системі миттєво повертати оновлені параметри сесії, як-от поточна точність або змінений рівень складності. Це створює відчуття живого відгуку системи на дії користувача [30].

З точки зору подальшого розвитку та підтримки проєкту, обрана структура є найбільш раціональною завдяки своїй масштабованості. Використання універсального обробника запитів у файлі api.php та єдиної бази даних дозволяє легко розширювати функціонал, додаючи нові типи когнітивних вправ без докорінної зміни архітектури. Це робить систему гнучкою до впровадження нових метрик, таких як Memory Score, та забезпечує стабільну роботу аналітичного модуля, який візуалізує статистику через Chart.js на основі структурованих JSON-даних.

2.3 Вибір технологічного стеку для реалізації системи

Вибір технологічного стеку для проєкту «Адаптивний веб-тренажер пам’яті» зумовлений необхідністю створення гнучкої клієнт-серверної моделі, здатної обробляти велику кількість мікро-взаємодій у реальному часі. Основним інструментом розробки фронтенду обрано мову JavaScript, яка реалізує логіку тренажерів у файлі trainer.js. Це дозволяє динамічно генерувати різнотипний контент — від числових послідовностей та текстових масивів (Words) до складних об’єктів, таких як emoji-кольори, геометричні форми та музичні ноти. Для візуального оформлення використано фреймворк Tailwind CSS, який дозволяє впровадити сучасний чорно-білий дизайн з високим рівнем контрастності. Використання таких компонентів, як animate-spin для лоадерів та ефектів hover для карток тренажерів, забезпечує високу якість користувацького досвіду (UX) та візуальний відгук системи під час відправки даних.

Серверна частина базується на мові PHP та архітектурному поєднанні файлів init.php для ініціалізації середовища та api.php для обробки бізнес-логіки [29]. Для забезпечення аналітичного функціоналу було обрано реляційну базу

даних MySQL, структура якої включає таблиці users, sessions та trials. Такий підхід дозволяє зберігати не лише загальні результати сесій, а й деталізовану інформацію про кожен окремий тріал, включаючи час показу послідовності (shown_ms), час реакції (reaction_ms) та правильність відповіді. Взаємодія між клієнтом і сервером реалізована через API за допомогою методу fetch, що дозволяє передавати JSON-пакети даних без перезавантаження сторінки, підтримуючи безперервність когнітивного тренування [30].

Окремим критично важливим інструментом у стеку технологій є бібліотека Chart.js, яка інтегрована в модуль статистики. Вона використовується для перетворення статистичних вибірок PHP у візуальні звіти: лінійні графіки для відстеження прогресу складності та гістограми для аналізу розподілу точності (accuracy buckets) і часу реакції (reaction buckets) [29]. Для оцінки ефективності користувача впроваджено специфічні алгоритмічні формули, зокрема Memory Score, що комбінує точність і швидкість виконання завдань. Весь комплекс обраних інструментів — від PHP-функцій фільтрації даних до JS-методів обробки масивів — спрямований на створення адаптивного середовища, яке здатне автоматично регулювати складність вправ у діапазоні від 1 до 10 рівнів на основі математичного аналізу стабільності та результативності користувач.

Як приклад практичної реалізації використання стеку технологій можна навести обробку результату тріалу. Після завершення вправи фронтенд формує JSON-запит через fetch і надсилає його на api.php.

Лістинг 2.1 - Приклад відправки результатів тріалу на сервер через API (fetch-запит)

```
fetch('/api.php?action=submit_trial', {
  method: 'POST',
  body: JSON.stringify({
    session_id,
    sequence,
    response,
    reaction_ms
```

```
  })
});
```

На сервері PHP-скрипт приймає ці дані, порівнює відповідь користувача з правильною послідовністю та одразу зберігає результат у таблицю trials. Після цього виконується аналіз останніх 5 тріалів, і якщо точність перевищує 70% при швидкій реакції, система автоматично збільшує difficulty у таблиці sessions. Таким чином, технології PHP, MySQL і JavaScript працюють у зв’язці, забезпечуючи повністю адаптивну поведінку тренажера в реальному часі.

2.4 Проєктування загальної архітектури вебдодатку

Проєкт побудований на основі сучасних принципів розробки вебзастосунків, де ключова увага приділяється безперервній взаємодії між діями користувача та алгоритмічними обчисленнями на сервері. Проєктування загальної архітектури вебдодатку передбачає створення цілісного механізму, що об’єднує інтерфейсну частину, логіку обробки даних та систему динамічного зворотного зв’язку. Така структура дозволяє реалізувати не просто статичний набір вправ, а інтелектуальну систему, яка в реальному часі реагує на когнітивні показники людини, підлаштовуючи умови завдання під її поточні можливості.

Архітектурна модель системи базується на циклічній логіці, де кожен успішний або невдалий крок користувача ініціює ланцюжок технічних подій: від фіксації мілісекунд реакції до перерахунку складності всього тренувального середовища. Це вимагає чіткої синхронізації між клієнтським сценарієм, який відповідає за візуальну презентацію, та серверним скриптом, що виконує роль аналітичного центру. Основна мета такої архітектури — мінімізувати технічні затримки та забезпечити плавність переходу між етапами запам’ятовування, відтворення та оцінювання результату [16].

Загальна архітектурна схема взаємодії та логіка роботи системи представлена на рисунку 1.2.

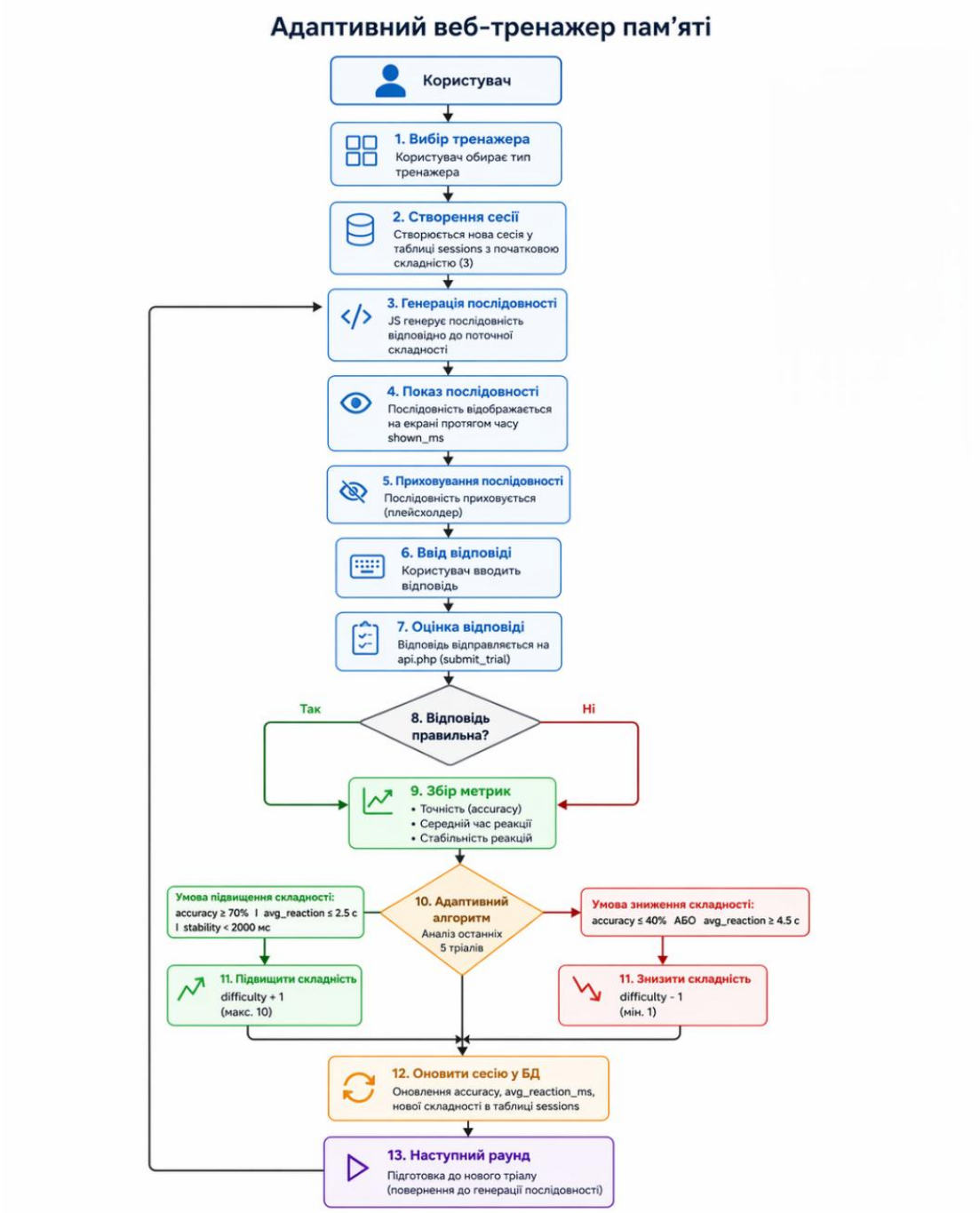


Рис. 2.2 – Схема загальної архітектури та алгоритму роботи вебтренажера

Як показано на рисунку 1.2, архітектура додатка має виражену ітераційну структуру. Процес розпочинається з ініціалізації сесії, де система створює унікальний запис у базі даних, фіксуючи початкові параметри, такі як базовий рівень складності. На етапі генерації послідовності вступає в дію програмний модуль на базі JavaScript, який створює набір стимулів (цифр, слів, форм тощо) відповідно до встановлених обмежень поточного рівня. Важливим аспектом проєктування є етап «Показу та приховування», де тривалість відображення

контенту жорстко контролюється таймерами, що є невід'ємною частиною когнітивного навантаження.

Центральне місце в архітектурі займає блок оцінки відповіді та адаптивного алгоритму. Після введення даних користувачем, система здійснює перевірку за двома ключовими векторами: правильність (correct) та швидкість (reaction_ms). Логіка розгалуження, представлена на схемі, демонструє, що навіть за умови правильної відповіді, низька швидкість реакції (наприклад, понад 1.5 секунди) може стати сигналом для системи не підвищувати складність або навіть залишити її на попередньому рівні для закріплення результату. Це забезпечує захист від перевантаження та підтримує стан «потоку», де завдання залишається складним, але доступним для виконання.

Технічна реалізація цієї схеми передбачає використання API-ендпоінтів для миттєвого оновлення стану сесії. Кожен завершений раунд (тріал) завершується відправкою JSON-об'єкта на сервер, де PHP-скрипт проводить ретроспективний аналіз останніх п'яти спроб. На основі цього аналізу оновлюються поля difficulty, accuracy та avg_reaction_ms у таблиці сесій. Така архітектура дозволяє не лише адаптувати наступний раунд, а й накопичувати масив даних для модуля статистики, який у подальшому використовує ці записи для побудови графіків прогресу через бібліотеку Chart.js.

Важливою перевагою обраної архітектурної моделі є її універсальність щодо типів даних. Як видно з правої частини схеми, структура підтримує паралельну роботу з різними тренажерами: від числових послідовностей та текстових масивів до візуальних форм, кольорових emoji та музичних ритмів. Кожен із цих типів інтегрується в єдину логіку «Генерація – Показ – Ввід – Оцінка», що робить систему масштабованою та дозволяє легко впроваджувати нові методики тренування без зміни фундаментальної архітектури вебдодатку. Таким чином, спроєктована система забезпечує повний цикл когнітивного контролю, поєднуючи високу швидкість обробки подій із глибокою аналітикою результатів.

Практичним результатом реалізації даної архітектури є розроблений вебсайт, що функціонує як персоналізоване інтелектуальне середовище. На

готовому ресурсі це проявляється через високу плавність взаємодії: система миттєво реагує на дії користувача, надаючи візуальне підтвердження успіху та коригуючи складність без переривання тренувального процесу. Підсумком роботи архітектурних рішень є не лише стабільний інструмент для вправ, а й повноцінний аналітичний кабінет. Він автоматично перетворює накопичені в базі даних цифри на наочні графіки та рекорди, дозволяючи користувачеві чітко відстежувати динаміку свого когнітивного розвитку в режимі реального часу.

2.5 Моделювання структури даних та організації зберігання інформації

Ефективне функціонування адаптивного вебтренажера пам’яті безпосередньо залежить від раціонально спроєктованої структури бази даних, яка повинна забезпечувати не лише надійне зберігання інформації, а й швидкий доступ до статистичних показників для роботи адаптивного алгоритму. Моделювання структури даних у даному проєкті базується на реляційній моделі, що дозволяє чітко розмежувати інформацію про користувачів, їхні навчальні сесії та результати конкретних вправ. Такий підхід забезпечує цілісність даних та дозволяє реалізувати складні вибірки для побудови аналітичних звітів і графіків прогресу.

Основою системи зберігання є ієрархічна модель зв’язків, де кожен рівень деталізації відповідає за свій функціональний блок. На верхньому рівні знаходиться інформація про облікові записи, що є необхідним для персоналізації досвіду тренування. Середній рівень агрегує дані про окремі підходи до тренувань (сесії), зберігаючи інтегральні показники успішності. Найнижчий рівень — тріали — фіксує кожну мікровзаємодію користувача із системою, що створює масив даних для динамічного регулювання складності.

Як показано на рисунку 2.1, база даних складається з трьох ключових взаємопов’язаних таблиць. Таблиця users є базовою та містить ідентифікатори користувачів, їхні контактні дані (email) та хешовані паролі для забезпечення безпечної авторизації. Зв’язок між користувачем та його активністю реалізується

через таблицю sessions за допомогою зовнішнього ключа user_id. У цій таблиці зберігаються кумулятивні дані по кожному тренуванню: поточний рівень складності (difficulty), середня точність (accuracy) та середній час реакції (avg_reaction_ms). Це дозволяє системі миттєво отримувати зріз успішності користувача без необхідності щоразу перераховувати всі історичні записи.

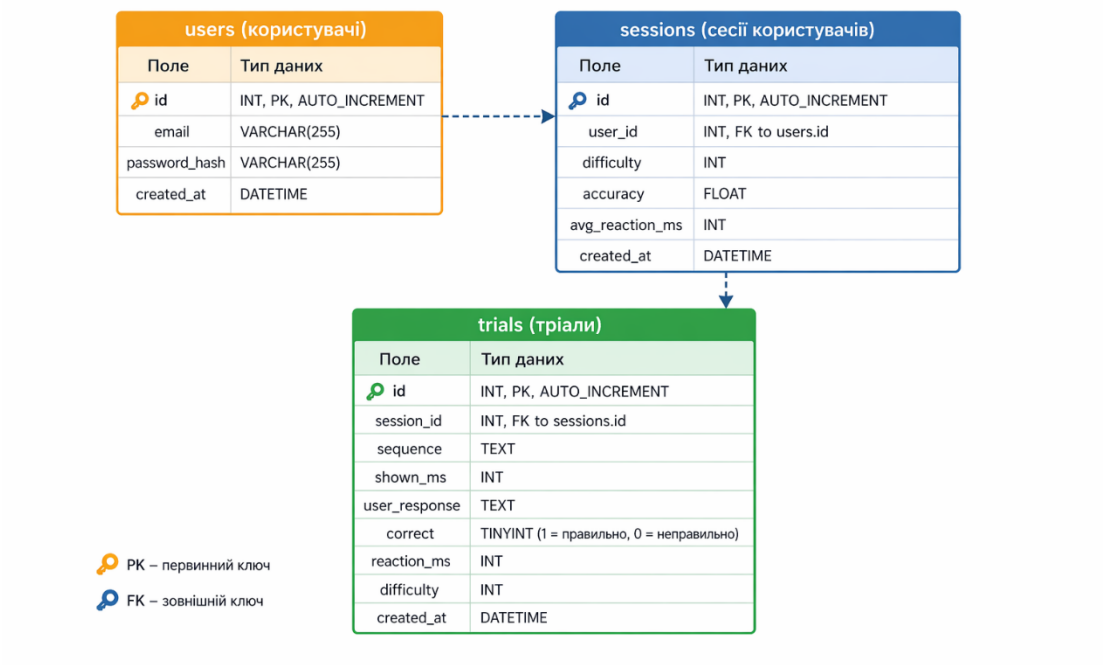


Рисунок 2.3 – Схема бази даних тренажера пам’яті

Деталізація кожної сесії відбувається в таблиці trials, яка пов’язана з таблицею сесій через поле session_id. Кожен запис у цій таблиці відповідає одній спробі відтворення послідовності. Тут фіксуються критично важливі для адаптивного алгоритму параметри: згенерована послідовність (sequence), тривалість її показу (shown_ms), відповідь користувача (user_response) та точний час реакції в мілісекундах. Використання типу даних DATETIME для поля created_at дозволяє відстежувати часову динаміку тренувань, що є необхідним для побудови графіків у модулі статистики.

Важливою особливістю організації зберігання є використання механізму ON DELETE CASCADE для зовнішніх ключів. Це гарантує, що при видаленні користувача або окремої сесії всі пов’язані з ними записи в таблиці тріалів будуть видалені автоматично, що запобігає появі «сміттєвих» даних у системі [31].

Обрана модель даних дозволяє ефективно реалізувати аналітичні функції, такі як розрахунок Memory Score, оскільки всі необхідні компоненти для формули — точність і швидкість — структуровані та проіндексовані для швидкого пошуку. Таке проектування забезпечує масштабованість вебдодатка, дозволяючи зберігати тисячі записів про спроби без втрати продуктивності при розрахунку адаптивної складності.

2.6 Проєктування адаптивного алгоритму регулювання складності

Проєктування адаптивного алгоритму регулювання складності є центральним елементом архітектури системи, оскільки саме цей компонент перетворює статичний вебдодаток на інтелектуальне середовище тренування. Основна мета алгоритму полягає у підтримці користувача в стані «когнітивного потоку», де рівень завдань постійно балансує між надмірною легкістю та демотивувальною складністю. В основі розробки лежить ідея динамічного зворотного зв’язку, що реалізується через серверний API-ендпоінт, який обробляє кожен результат взаємодії користувача з тренажером.

Технічна реалізація алгоритму базується на архітектурі POST-запитів, де клієнтська частина передає деталізований масив даних про кожен завершений тріал. Процес обробки включає не лише перевірку правильності відповіді, а й глибокий аналіз контексту виконання завдання, що дозволяє системі приймати обґрунтовані рішення про зміну параметрів наступного раунду. Такий підхід забезпечує високу персоналізацію навчання, оскільки складність підлаштовується під швидкість засвоєння інформації конкретним користувачем у режимі реального часу [32].

Як показано на рисунку 2.3, процес ініціюється запитом користувача до api.php з дією submit_trial. На першому етапі PHP-бекенд виконує валідацію сесії та перевірку авторизації, що гарантує цілісність даних. Ключовою операцією є порівняння введеної користувачем відповіді з оригінальною послідовністю (Compare response & sequence), що визначає параметр правильності. Паралельно

з цим система фіксує часові метрики, які є критичними для наступного етапу — аналізу останніх п'яти тріалів.

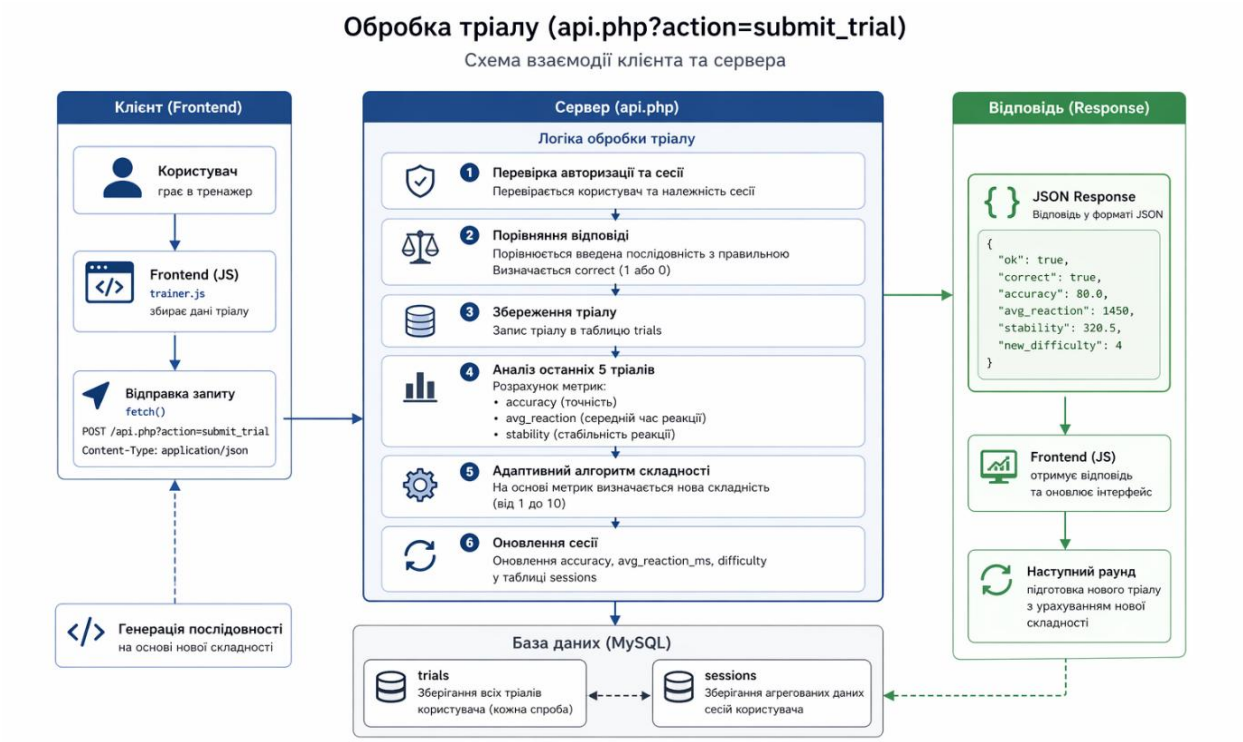


Рисунок 2.4 – Алгоритм роботи серверної логіки та адаптивної зміни складності

Логіка адаптації складності (Adaptive difficulty logic) використовує накопичений масив результатів для розрахунку трьох метрик: точності, середнього часу реакції та стабільності показників [33]. На схемі видно, що після обчислення нового рівня складності (Calculate newDifficulty), результати негайно зберігаються в базу даних, оновлюючи поточний стан сесії. Фінальним етапом є формування JSON-відповіді, яка містить не лише підтвердження правильності поточної відповіді, а й нові параметри складності, що будуть застосовані фронтендом для наступного завдання.

Використання такої архітектури дозволяє реалізувати гнучкий механізм регулювання: якщо користувач демонструє високу точність при швидкій реакції, алгоритм ініціює підвищення складності. У випадку погіршення результатів система автоматично знижує навантаження, забезпечуючи комфортні умови для відновлення когнітивного ресурсу. Завдяки тому, що всі обчислення винесені на

бекенд, алгоритм залишається захищеним від зовнішніх маніпуляцій, а клієнтська частина отримує готові інструкції для відображення, що забезпечує стабільну та швидку роботу вебтренажера.

2.7 Проєктування взаємодії користувача з тренажером

Проєктування взаємодії користувача з тренажером базується на принципах максимальної інформативності та мінімального когнітивного опору інтерфейсу. Основна мета цього етапу — створити інтуїтивно зрозумілий шлях користувача від моменту вибору вправи до отримання детального аналізу своїх результатів. Взаємодія розглядається не просто як набір натискань на кнопки, а як безперервний діалог між людиною та системою, де кожен крок супроводжується візуальним підтвердженням, а складність середовища адаптується до темпу засвоєння інформації.

Центральним елементом проєктування інтерфейсу є розділення на активну фазу тренування та аналітичну фазу перегляду статистики. Під час тренування фокус користувача спрямований на висококонтрастні об'єкти, що дозволяє уникнути відволікань. Після завершення сесії система перемикає увагу на аналітику, де важливо надати дані в агрегованому та візуально доступному вигляді. Це досягається шляхом інтеграції складних серверних розрахунків у прості графічні компоненти, що забезпечує прозорість процесу навчання [15].

Логіка формування аналітичного фідбеку та структура взаємодії на сторінці статистики представлена на рисунку 2.4.

Як показано на рисунку 2.4, шлях користувача розпочинається з відкриття сторінки статистики, що ініціює складний внутрішній процес обробки даних. PHP-бекенд виконує запити до бази даних для вилучення історії сесій, після чого проводить математичну обробку: розрахунок середніх значень, рекордів та інтегрального показника Memory Score. Окремим важливим процесом є генерація даних для гістограм (histogram buckets), які дозволяють користувачеві побачити не просто середній результат, а розподіл своїх зусиль за часом та точністю.

Рисунок 2.5 – Схема взаємодії компонентів при формуванні статистики

Вихідні дані структуруються у два потоки для забезпечення оптимального рендерингу сторінки. Перший потік відповідає за статичні елементи, такі як HTML-картки з рекордами та інтерактивна таблиця всіх сесій. Другий потік передає дані у форматі JSON до фронтенд-модуля на базі JavaScript та Chart.js. Це дозволяє створювати динамічні графіки, які реагують на дії користувача, наприклад, зміну часового діапазону або фільтрацію за типом тренажера. Така організація взаємодії дозволяє користувачеві самостійно досліджувати свій прогрес, виявляти слабкі місця та коригувати стратегію подальших тренувань.

Кінцевий інтерфейс відображає комплексний зріз когнітивної діяльності: від швидких показників поточної сесії до довгострокових трендів у вигляді лінійних графіків та гістограм. Завдяки продуманій архітектурі взаємодії, користувач отримує відчуття повного контролю над процесом навчання, а візуалізація успіхів через Memory Score слугує потужним стимулом для підтримки регулярності занять. Такий підхід до проєктування забезпечує не лише функціональність вебдодатка, а й високу залученість користувача, що є критично важливим для досягнення реального ефекту в тренуванні пам'яті.

2.8 Забезпечення продуктивності та маштабованості системи

У межах проєктування адаптивного вебтренажера пам’яті особлива увага приділена забезпеченню продуктивності та можливості подальшого масштабування системи, оскільки вона передбачає часті короткі запити користувача, обробку тріалів у реальному часі та постійну взаємодію з базою даних.

Клієнтська частина системи реалізована таким чином, щоб мінімізувати навантаження на сервер. Логіка генерації тренувальних послідовностей (числа, слова, кольори, форми, музика) виконується на стороні браузера за допомогою JavaScript (trainer.js). Це дозволяє уникнути зайвих серверних запитів під час кожного раунду та зменшує затримки у відображенні стимулів, оскільки контроль часу показу та приховування елементів виконується локально.

Серверна частина реалізована на PHP 8+ у вигляді легких HTTP-ендпоінтів (api.php), які обробляють лише критичні операції: перевірку відповідей, збереження тріалів та оновлення параметрів сесії. Такий підхід дозволяє зменшити час обробки запитів, оскільки сервер не виконує зайву бізнес-логіку, а лише обробляє дані, необхідні для адаптації складності.

Для взаємодії між клієнтом і сервером використовується JSON API через fetch, що забезпечує асинхронну обробку тріалів без перезавантаження сторінки. Це дозволяє підтримувати високу швидкість роботи інтерфейсу навіть при великій кількості послідовних тренувальних раундів.

Рівень роботи з даними реалізовано на основі MySQL із використанням двох основних таблиць — sessions та trials. Таблиця trials зберігає детальні результати кожної спроби, тоді як sessions агрегує показники продуктивності користувача. Така структура дозволяє розділити детальні та узагальнені дані, що зменшує навантаження на запити при формуванні статистики.

Оптимізація запитів до бази даних реалізується через вибіркове отримання останніх результатів (наприклад, аналіз лише останніх 5 тріалів для адаптивного алгоритму). Це дозволяє уникнути повного сканування таблиць і зменшує час виконання SQL-запитів при великій кількості записів.

Масштабованість системи забезпечується модульною структурою проєкту. Кожен тренажер реалізований як окрема сторінка (trainer_numbers.php, trainer_words.php тощо), а логіка обробки уніфікована через спільний API (api.php). Це дозволяє легко додавати нові типи тренажерів без зміни основної архітектури.

Додатково адаптивний алгоритм регулювання складності працює на основі обмеженого набору останніх даних (точність, реакція, стабільність), що знижує обчислювальне навантаження та забезпечує стабільну продуктивність незалежно від кількості сесій у базі.

Таким чином, продуктивність системи досягається за рахунок клієнтської обробки тренувальної логіки, мінімізації серверних операцій, оптимізованої роботи з базою даних та використання асинхронної взаємодії через JSON API, а масштабованість забезпечується модульною структурою та уніфікованою архітектурою обробки даних.

2.9 Висновки до другого розділу

У процесі проєктування адаптивного вебтренажера пам’яті було здійснено аналіз архітектурних підходів, структури даних та алгоритмів обробки результатів, що дозволило визначити оптимальну клієнт–серверну модель системи. Такий підхід забезпечує розділення інтерфейсу користувача та серверної логіки, що відповідає вимогам до системи з динамічним регулюванням когнітивного навантаження.

Аналіз технологічного стеку, що включає PHP 8+, MySQL, PDO, JavaScript (trainer.js), HTML, Tailwind CSS та JSON API, підтвердив його придатність для реалізації вебтренажера. PHP використовується для обробки запитів та реалізації адаптивного алгоритму, MySQL — для зберігання даних сесій і тріалів, а JavaScript забезпечує інтерактивну роботу тренажерів у браузері. Tailwind CSS використовується для побудови адаптивного інтерфейсу.

Узагальнення структури даних показало доцільність використання таблиць sessions і trials, що дозволяє розділити агреговані та детальні результати

користувача. Це забезпечує можливість аналізу прогресу та формування статистики когнітивної продуктивності.

Адаптивний алгоритм регулювання складності базується на аналізі точності, часу реакції та стабільності результатів останніх тріалів, що дозволяє індивідуально підлаштовувати рівень навантаження під користувача.

Проведений аналіз створює основу для подальшої реалізації системи, яка забезпечує динамічну адаптацію складності та відстеження когнітивного прогресу користувача.

3 РЕАЛІЗАЦІЯ АДАПТИВНОГО ВЕБТРЕНАЖЕРА ПАМ'ЯТІ

3.1 Загальна архітектура реалізованої системи

Реалізація вебдодатка «Адаптивний веб-тренажер пам'яті» базується на клієнт-серверній архітектурі, що забезпечує чіткий поділ між логікою обробки даних, збереженням стану та інтерфейсом користувача. Обрана модель дозволяє ефективно розподілити ресурси: фронтенд відповідає за динамічне відображення стимульного матеріалу, тоді як бекенд бере на себе обчислювальне навантаження з аналізу ефективності та адаптації складності.

Загальна структура системи побудована навколо інтерактивної взаємодії в режимі реального часу. Усі запити від клієнта до сервера передаються через асинхронні технології (fetch API), що дозволяє проводити тренування без перезавантаження сторінок [34]. Центральним елементом архітектури є програмний інтерфейс (API), який обробляє результати кожного тріалу, порівнює дані з еталоном та розраховує метрики успішності.

Логіка збереження даних організована через реляційну модель, де розмежовано інформацію про користувачів, агреговані дані сесій та детальні логи кожної спроби. Взаємодія компонентів відбувається за циклічним принципом: збір реакції користувача, передача на сервер та отримання оновлених параметрів складності. Це створює замкнутий контур адаптивного керування, де кожен наступний крок системи залежить від попередніх результатів користувача.

Структура проєкту чітко диференційована: окремі файли відповідають за типи вправ, універсальний JavaScript-модуль керує ігровою логікою, а централізований PHP-обробник забезпечує математичну підтримку адаптивного алгоритму.

3.2 Розробка серверного компонента та схеми даних

Серверна частина вебдодатку адаптивного тренажера пам’яті реалізована з використанням мови програмування PHP та системи керування базами даних MySQL. Її основним призначенням є обробка HTTP-запитів від клієнтської частини, збереження результатів тренувань, обчислення статистичних показників та забезпечення роботи адаптивного алгоритму зміни складності [29].

Точкою входу в серверну логіку виступає файл api.php, який функціонує як універсальний API-ендпоінт для взаємодії з фронтендом. Усі запити від клієнта надходять у форматі JSON і обробляються залежно від значення параметра action. Основною операцією є обробка результатів тріалу (submit_trial), що включає повний цикл аналізу відповіді користувача.

На початковому етапі роботи сервер виконує підключення до конфігураційного файлу init.php, який відповідає за ініціалізацію сесії, встановлення з’єднання з базою даних та перевірку автентифікації користувача. Якщо користувач не авторизований, запит одразу завершується з відповідним повідомленням про помилку, що запобігає несанкціонованому доступу до функціоналу системи.

Структура бази даних розроблена відповідно до логіки роботи тренажера та включає три основні таблиці: users, sessions та trials, що зображено на рисунку 3.1.

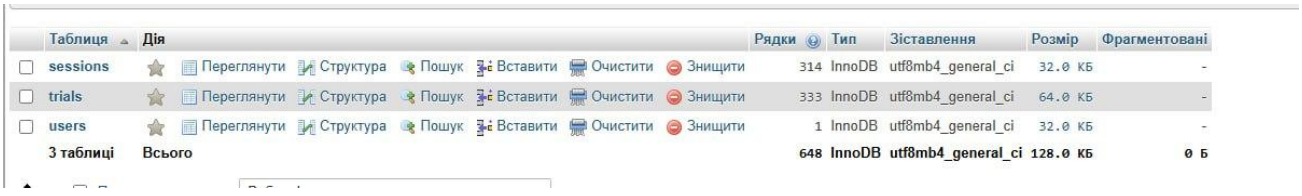


Рисунок 3.1 – Інтерфейс phpMyAdmin, що показує перелік таблиць бази даних

У середовищі адміністрування бази даних відображено таблиці системи, зокрема sessions, trials та users, із зазначенням кількості записів, типу рушія зберігання (InnoDB), кодування (utf8mb4_general_ci) та загального розміру.

Використання рушія InnoDB забезпечує підтримку транзакцій та зовнішніх ключів, що є важливим для підтримання цілісності даних. Кодування utf8mb4 дозволяє коректно зберігати текстові дані, включаючи спеціальні символи.

Таблиця sessions призначена для збереження узагальнених характеристик кожної сесії тренування. Вона містить ідентифікатор сесії, ідентифікатор користувача, поточний рівень складності, середню точність виконання завдань та середній час реакції. Дані цієї таблиці використовуються для швидкого доступу до актуального стану користувача та відображення статистики.

Таблиця trials зберігає детальну інформацію про кожен окремий тріал. До її складу входять поля для збереження згенерованої послідовності, відповіді користувача, часу показу, часу реакції, правильності виконання та рівня складності. Саме ці дані використовуються для подальшого аналізу та роботи адаптивного алгоритму.

Таблиця users реалізує базову функціональність автентифікації та містить інформацію про облікові записи користувачів, зокрема електронну пошту та хеш пароля.

Основний алгоритм обробки тріалу починається з отримання та декодування вхідних даних із тіла HTTP-запиту. Дані передаються у форматі JSON та містять інформацію про ідентифікатор сесії, згенеровану послідовність, відповідь користувача, час реакції та рівень складності. Після цього виконується перевірка коректності отриманих значень і відповідності сесії поточному користувачу [32].

Логіка перевірки відповіді реалізована шляхом прямого порівняння введеної послідовності з еталонною. У випадку їх повного збігу результат вважається правильним, що фіксується у вигляді булевого значення. Даний процес наведено в лістингу 3.1.

Лістинг 3.1 – Перевірка відповіді користувача

```
$correct = ($response === $sequence) ? 1 : 0;
```

Після визначення правильності відповіді система виконує збереження результатів у базу даних. Для цього використовується таблиця trials, яка містить детальну інформацію про кожну спробу користувача. Збереження здійснюється через підготовлений SQL-запит, що забезпечує захист від SQL-ін’єкцій та коректну обробку даних.

Лістинг 3.2 – Збереження тріалу в базу даних

```
$stmt = $pdo->prepare("
INSERT INTO trials
(session_id, sequence, shown_ms, user_response, correct, reaction_ms, difficulty)
VALUES (?, ?, ?, ?, ?, ?, ?)
");
$stmt->execute([$session_id, $sequence, $shown_ms, $response, $correct, $reaction_ms, $difficulty]);
```

Після збереження тріалу сервер виконує оновлення агрегованих показників сесії. Для цього розраховується середня точність і середній час реакції на основі останніх виконаних завдань. Отримані значення записуються у відповідні поля таблиці sessions, що дозволяє підтримувати актуальний стан прогресу користувача.

Лістинг 3.3 – Оновлення параметрів сесії

```
$stmt = $pdo->prepare("
UPDATE sessions
SET accuracy = ?, avg_reaction_ms = ?, difficulty = ?
WHERE id = ?
");
$stmt->execute([$accuracy, $avgReaction, $newDifficulty, $session_id]);
```

Окрему увагу приділено організації взаємодії між клієнтською та серверною частинами. Обмін даними здійснюється у форматі JSON, що забезпечує універсальність і зручність інтеграції з JavaScript. Після завершення обробки сервер формує структуровану відповідь, яка містить результат перевірки, обчислені показники та новий рівень складності.

Лістинг 3.4 – Формування JSON-відповіді

```
echo json_encode([
    'ok' => true,
    'correct' => (bool)$correct,
    'accuracy' => $accuracy,
    'avg_reaction' => $avgReaction,
    'stability' => round($stability, 2),
    'new_difficulty' => $newDifficulty
]);
```

Серверний компонент системи забезпечує повний цикл обробки даних: від прийому результатів тренування до їх збереження, аналізу та повернення користувачеві. Використання чітко структурованої бази даних, підготовлених SQL-запитів і JSON-взаємодії дозволяє досягти високої надійності, безпеки та масштабованості системи.

3.3 Реалізація адаптивного модуля керування когнітивним навантаженням

Адаптивний модуль є ключовим компонентом розробленого вебтренажера пам’яті, оскільки саме він забезпечує динамічне підлаштування складності вправ під індивідуальні можливості користувача. Основною метою модуля є підтримання оптимального рівня когнітивного навантаження, при якому користувач не перевантажується, але водночас отримує достатній рівень складності для розвитку пам’яті.

Реалізація адаптивного механізму базується на аналізі результатів виконання користувачем окремих тріалів, які зберігаються у базі даних. Для оцінки ефективності використовуються три основні показники: точність відповідей, середній час реакції та стабільність виконання. Точність визначає частку правильних відповідей, час реакції характеризує швидкість обробки інформації, а стабільність відображає рівномірність результатів користувача.

Обробка даних виконується на серверній стороні у файлі api.php після кожного завершеного тріалу. Система отримує дані від клієнта, перевіряє їх коректність та зберігає у таблиці trials, після чого здійснює аналіз останніх результатів. Для цього вибираються декілька останніх записів (як правило,

п’ять), що дозволяє оцінити актуальний стан користувача без урахування застарілих даних.

На основі отриманої вибірки розраховується відсоток правильних відповідей та середній час реакції. Додатково визначається стабільність, яка обчислюється як відхилення часу реакції між тріалами. Такий підхід дозволяє врахувати не лише середні значення, але й рівень коливань у результатах, що є важливим для коректної адаптації складності.

Зміна рівня складності відбувається за визначеними правилами. Якщо користувач демонструє високі результати, зокрема високу точність та швидку реакцію, система підвищує рівень складності. У протилежному випадку, коли спостерігається значна кількість помилок або повільна реакція, складність знижується. Важливою особливістю є наявність обмежень, які не дозволяють виходити за межі допустимого діапазону рівнів складності.

Такий механізм забезпечує поступове ускладнення або спрощення завдань без різких змін, що позитивно впливає на користувацький досвід. Користувач не стикається з надмірно складними або занадто простими вправами, оскільки система постійно підтримує баланс між викликом і доступністю.

Адаптивний модуль також тісно інтегрований із клієнтською частиною додатку. Після кожного тріалу сервер повертає оновлені значення показників і новий рівень складності у форматі JSON. Клієнтська частина, у свою чергу, використовує ці дані для генерації наступної послідовності, змінюючи її довжину та час відображення відповідно до нового рівня.

Завдяки такій архітектурі досягається ефект реального часу: система реагує на дії користувача миттєво, без необхідності перезавантаження сторінки. Це створює відчуття інтерактивності та підвищує ефективність тренування.

Реалізований адаптивний модуль забезпечує інтелектуальне керування когнітивним навантаженням шляхом аналізу поведінки користувача та автоматичної зміни параметрів тренування. Його використання дозволяє створити персоналізоване середовище навчання, у якому складність завдань постійно відповідає поточному рівню підготовки користувача.

3.3.1 Програмна обробка вхідних даних для оцінки результативності

Програмна обробка вхідних даних є першим і ключовим етапом роботи адаптивного модуля, оскільки саме на цьому рівні формується база для подальшого аналізу результативності користувача. Усі дані надходять із клієнтської частини додатку у вигляді JSON-запиту до файлу api.php після завершення кожного тріалу.

На початковому етапі сервер зчитує тіло HTTP-запиту та виконує декодування JSON у внутрішню структуру даних. Отриманий масив містить основні параметри виконаного завдання: ідентифікатор сесії, згенеровану послідовність, відповідь користувача, час реакції, час показу та поточний рівень складності. Далі виконується перевірка коректності отриманих даних і їх відповідності очікуваному формату.

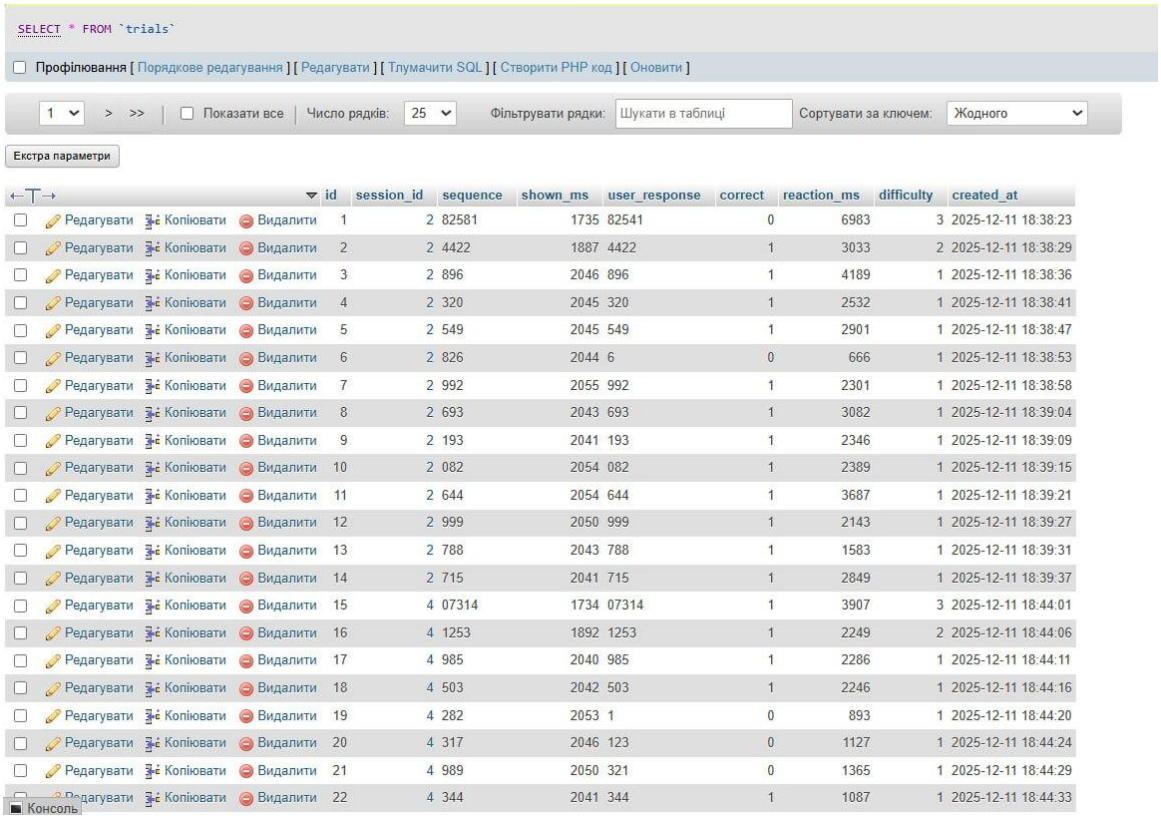
Лістинг 3.5 – Отримання та декодування вхідних даних

```
$data = json_decode(file_get_contents('php://input'), true);
```

Після цього система виконує перевірку належності сесії поточному користувачу. Це необхідно для забезпечення безпеки та запобігання несанкціонованому доступу до чужих даних. У випадку невідповідності або відсутності сесії обробка запиту припиняється.

Наступним кроком є визначення правильності відповіді користувача. Для цього виконується пряме порівняння введеної послідовності з тією, що була згенерована системою. Результат цього порівняння фіксується у вигляді бінарного значення (1 — правильно, 0 — неправильно), що надалі використовується для розрахунку точності.

Після перевірки відповідь разом з усіма супутніми параметрами зберігається у таблиці trials, яка містить повну історію виконання завдань користувачем. Структура цієї таблиці дозволяє детально аналізувати кожну спробу, включаючи швидкість реакції та рівень складності на момент виконання.



SELECT * FROM `trials`

Профілювання [Порядкове редагування] [Редагувати] [Тлумачити SQL] [Створити PHP код] [Оновити]

1 > >> | Показати все | Число рядків: 25 | Фільтрувати рядки: Шукати в таблиці | Сортувати за ключем: Жодного

Екстра параметри

	id	session_id	sequence	shown_ms	user_response	correct	reaction_ms	difficulty	created_at
Редагувати Копіювати Видалити	1	2	82581	1735	82541	0	6983	3	2025-12-11 18:38:23
Редагувати Копіювати Видалити	2	2	4422	1887	4422	1	3033	2	2025-12-11 18:38:29
Редагувати Копіювати Видалити	3	2	896	2046	896	1	4189	1	2025-12-11 18:38:36
Редагувати Копіювати Видалити	4	2	320	2045	320	1	2532	1	2025-12-11 18:38:41
Редагувати Копіювати Видалити	5	2	549	2045	549	1	2901	1	2025-12-11 18:38:47
Редагувати Копіювати Видалити	6	2	826	2044	6	0	666	1	2025-12-11 18:38:53
Редагувати Копіювати Видалити	7	2	992	2055	992	1	2301	1	2025-12-11 18:38:58
Редагувати Копіювати Видалити	8	2	693	2043	693	1	3082	1	2025-12-11 18:39:04
Редагувати Копіювати Видалити	9	2	193	2041	193	1	2346	1	2025-12-11 18:39:09
Редагувати Копіювати Видалити	10	2	082	2054	082	1	2389	1	2025-12-11 18:39:15
Редагувати Копіювати Видалити	11	2	644	2054	644	1	3687	1	2025-12-11 18:39:21
Редагувати Копіювати Видалити	12	2	999	2050	999	1	2143	1	2025-12-11 18:39:27
Редагувати Копіювати Видалити	13	2	788	2043	788	1	1583	1	2025-12-11 18:39:31
Редагувати Копіювати Видалити	14	2	715	2041	715	1	2849	1	2025-12-11 18:39:37
Редагувати Копіювати Видалити	15	4	07314	1734	07314	1	3907	3	2025-12-11 18:44:01
Редагувати Копіювати Видалити	16	4	1253	1892	1253	1	2249	2	2025-12-11 18:44:06
Редагувати Копіювати Видалити	17	4	985	2040	985	1	2286	1	2025-12-11 18:44:11
Редагувати Копіювати Видалити	18	4	503	2042	503	1	2246	1	2025-12-11 18:44:16
Редагувати Копіювати Видалити	19	4	282	2053	1	0	893	1	2025-12-11 18:44:20
Редагувати Копіювати Видалити	20	4	317	2046	123	0	1127	1	2025-12-11 18:44:24
Редагувати Копіювати Видалити	21	4	989	2050	321	0	1365	1	2025-12-11 18:44:29
Редагувати Копіювати Видалити	22	4	344	2041	344	1	1087	1	2025-12-11 18:44:33

Консоль

Рисунок 3.2 – Вміст таблиці trials

У таблиці відображаються такі поля: id, session_id, sequence, shown_ms, user_response, correct, reaction_ms та дата створення запису. Така структура дозволяє зберігати повну інформацію про кожен тріал і використовувати її для подальших обчислень. Наявність окремого запису для кожної спроби дає змогу аналізувати як короткострокову динаміку (останні кілька тріалів), так і довгостроковий прогрес користувача.

Після збереження даних виконується вибірка останніх тріалів для аналізу. Зазвичай використовується обмежена кількість записів (наприклад, п’ять), що дозволяє оцінити поточний стан користувача без впливу застарілих результатів.

Лістинг 3.6 – Отримання останніх тріалів для аналізу

```
$stmt = $pdo->prepare("
SELECT correct, reaction_ms
FROM trials
WHERE session_id = ?
ORDER BY id DESC
LIMIT 5
");
$stmt->execute([$session_id]);
$recent = $stmt->fetchAll();
```

На основі отриманих даних обчислюються ключові показники результативності. Точність визначається як відсоток правильних відповідей серед вибраних тріалів. Середній час реакції обчислюється як середнє арифметичне значень reaction_ms. Додатково визначається стабільність, яка характеризує рівень коливань часу реакції між окремими спробами.

Зібрані та оброблені показники передаються до наступного етапу — модуля керування складністю, де на їх основі приймається рішення про зміну рівня складності тренування.

3.3.2 Розробка керуючої логіки переходу між рівнями складноті

Керуюча логіка переходу між рівнями складності є завершальним етапом роботи адаптивного модуля та безпосередньо визначає, як система реагує на результати користувача. Її основне завдання полягає у прийнятті рішення про підвищення, зниження або збереження поточного рівня складності на основі обчислених показників результативності.

Вхідними даними для цього етапу є агреговані показники, отримані під час аналізу останніх тріалів: точність (accuracy), середній час реакції (avgReaction) та стабільність (stability). Саме ці параметри дозволяють оцінити не лише правильність виконання завдань, але й швидкість та рівномірність роботи користувача.

Логіка прийняття рішень реалізована у вигляді набору умовних правил. Якщо користувач демонструє високий рівень точності та швидку реакцію при достатній стабільності, система підвищує рівень складності. Це означає збільшення довжини послідовності або зменшення часу її відображення. У випадку, коли точність низька або час реакції значно перевищує допустимі межі, складність знижується для полегшення виконання завдань.

Лістинг 3.7 – Логіка зміни рівня складності

```
      if ($accuracy >= 70 && $avgReaction <= 2500 && $stability < 
2000) {
          $newDifficulty = $difficulty + 1;
      } elseif ($accuracy <= 40 || $avgReaction >= 4500) {
          $newDifficulty = $difficulty - 1;
```

```
} else {
    $newDifficulty = $difficulty;
}
```

Важливою особливістю реалізації є використання обмежень для рівня складності. Значення складності не може виходити за межі визначеного діапазону (від 1 до 10), що запобігає появі надто простих або надмірно складних завдань.

Лістинг 3.8 – Обмеження рівня складності

```
$newDifficulty = max(1, min(10, $newDifficulty));
```

Після визначення нового рівня складності система виконує оновлення відповідного запису у таблиці sessions, яка зберігає агреговані результати кожної сесії користувача.

У таблиці відображаються результати ігрових сесій для користувача, зокрема для user_id = 1. Вона містить такі основні поля: ідентифікатор сесії, рівень складності (difficulty), точність (accuracy), середній час реакції (avg_reaction_ms) та інтегральний показник (score). Наявність цих даних дозволяє не лише адаптувати складність у процесі гри, але й аналізувати загальний прогрес користувача.

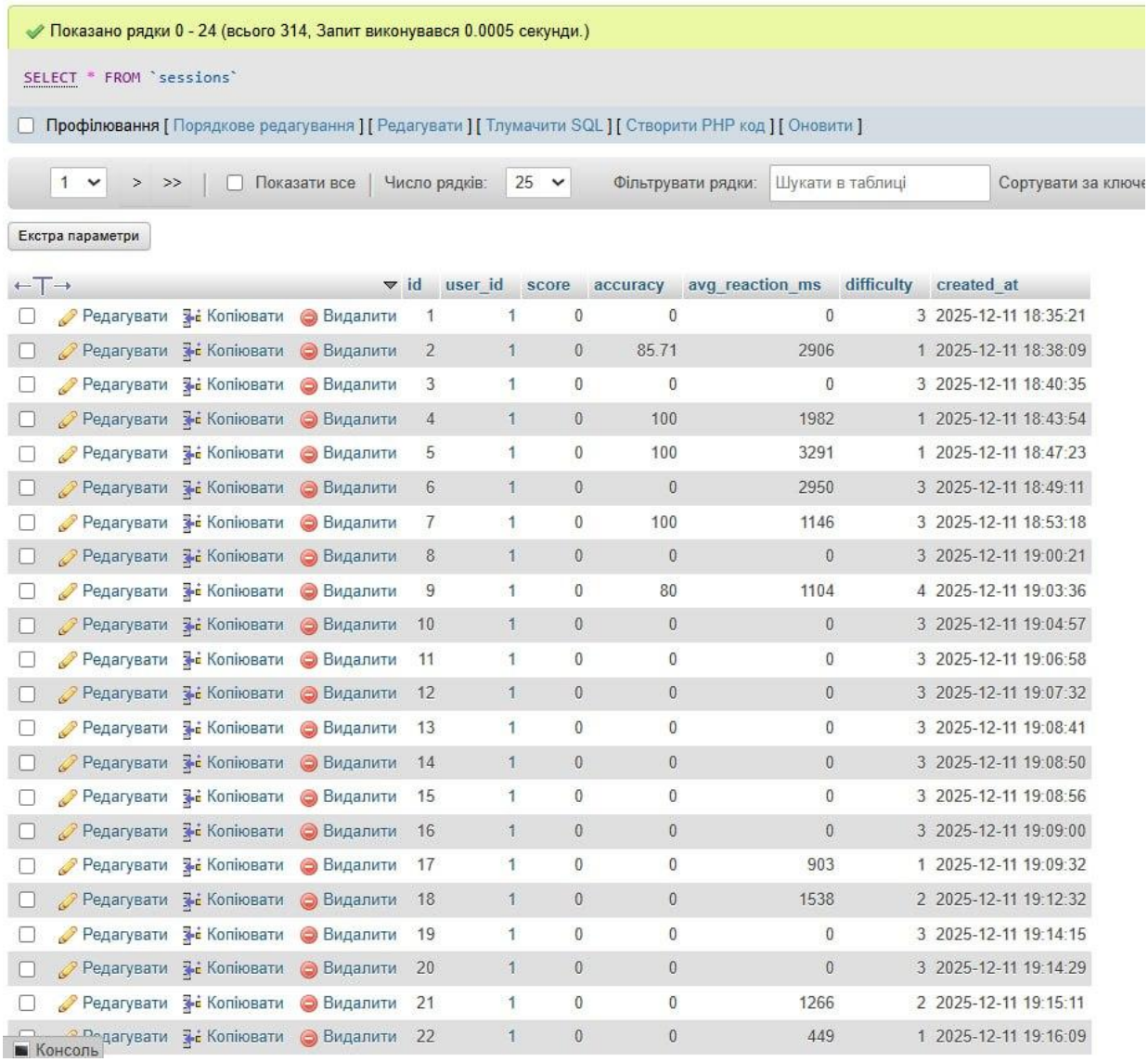


Рисунок 3.3 – Вміст таблиці sessions

Після оновлення складності нові значення повертаються на клієнтську сторону у вигляді JSON-відповіді. Фронтенд використовує ці дані для генерації наступного завдання з урахуванням зміненого рівня. Таким чином, кожен наступний тріал безпосередньо залежить від результатів попередніх, що забезпечує безперервну адаптацію процесу тренування.

Запропонована керуюча логіка забезпечує плавний перехід між рівнями складності без різких змін, що є важливим для підтримання мотивації користувача. Вона дозволяє уникнути ситуацій перевантаження або втрати інтересу, забезпечуючи оптимальний баланс між складністю та доступністю завдань.

3.4 Реалізація функціональних блоків ігрових режимів

У межах розробки вебтренажера пам’яті реалізовано набір ігрових режимів, кожен з яких спрямований на розвиток різних типів пам’яті: зорової, слухової та короткочасної. Кожен тренажер реалізований як окрема сторінка (наприклад, trainer_numbers.php, trainer_colors.php, trainer_shapes.php), але використовує спільний модуль trainer.js, що відповідає за генерацію послідовностей, їх відображення та взаємодію з сервером.

До візуальних тренажерів належать режими з використанням сітки або кольорових/геометричних елементів. Користувачу короткочасно демонструється певна конфігурація, після чого вона зникає, і необхідно її відтворити.

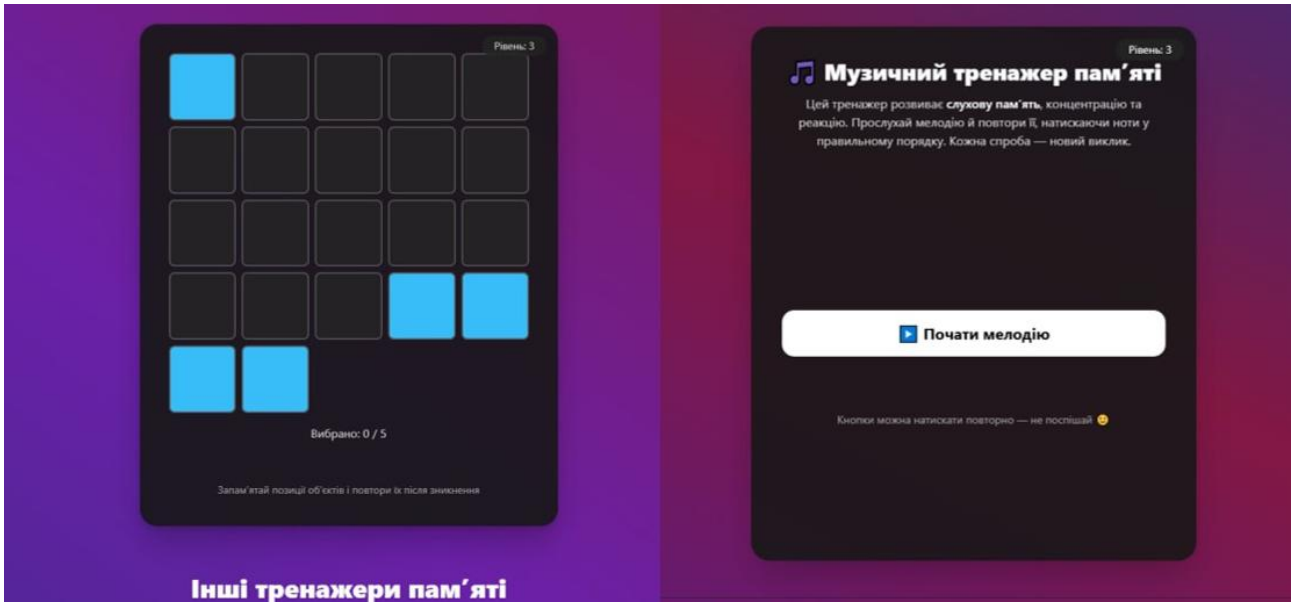


Рисунок 3.4 – Інтерфейси візуального та музичного тренажерів пам’яті

На рисунку представлено як приклад сітковий тренажер (5×5 з підсвіченими елементами), так і музичний режим, у якому користувач взаємодіє через кнопку запуску мелодії. Візуальні вправи розвивають просторову пам’ять і увагу, тоді як музичний тренажер орієнтований на слухову пам’ять і концентрацію.

Окрему категорію становить тренажер послідовностей, який є базовим для оцінки короткочасної пам’яті. У цьому режимі користувачу спочатку демонструється послідовність символів (наприклад, цифр), після чого вона приховується і необхідно відтворити її у спеціальних полях введення.

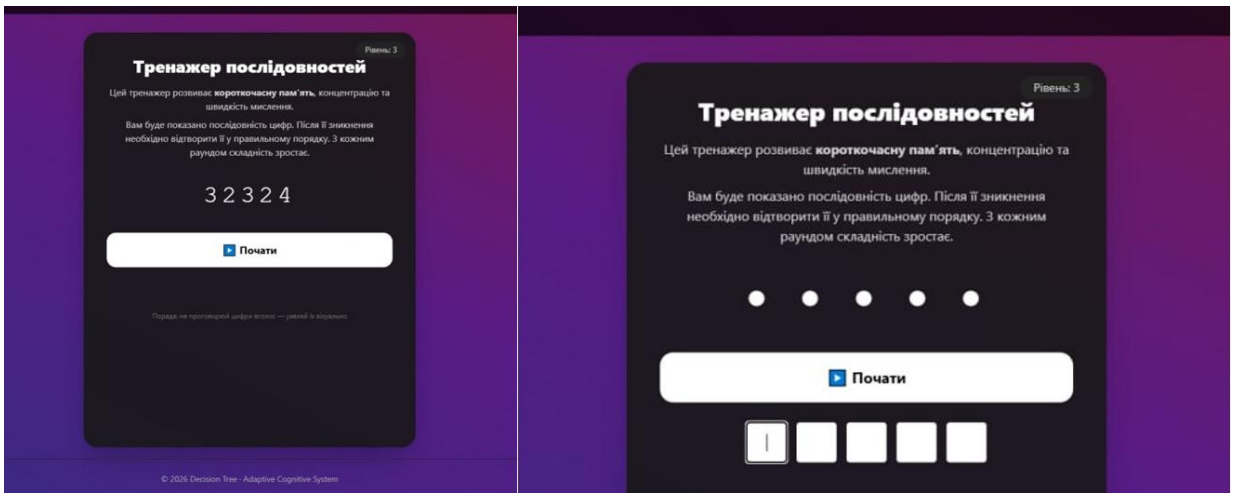


Рисунок 3.5 – Етап показу та введення у тренажері послідовностей

Інтерфейс містить комірки для введення відповіді, кнопку запуску та індикатор рівня складності. Логіка роботи включає генерацію послідовності, її відображення протягом обмеженого часу та подальшу перевірку відповіді через сервер.

Усі режими інтегровані з адаптивним алгоритмом, який змінює їх параметри (довжину послідовності, кількість елементів, час показу) залежно від результатів користувача. Це забезпечує індивідуалізацію тренування та підтримання оптимального рівня складності.

Реалізовані ігрові режими формують гнучку систему вправ, що забезпечує комплексний розвиток пам’яті та ефективну взаємодію користувача з додатком.

3.5 Оптимізація алгоритмів пошуку інформації на вебсайті за допомогою машинного навчання

У межах розробки вебтренажера пам’яті під оптимізацією пошуку інформації розуміється інтелектуальна обробка та аналіз даних користувача з

метою покращення доступу до вправ, адаптації інтерфейсу та підвищення ефективності навчального процесу. Замість класичного пошуку за запитом у системі реалізовано підхід, що базується на аналізі поведінки користувача та його результатів.

Головним джерелом даних для такої оптимізації є результати сесій і тріалів, які зберігаються у базі даних. На їх основі система формує узагальнені показники, що дозволяють визначити рівень підготовки користувача, його сильні та слабкі сторони. Це дає змогу не лише адаптувати складність, але й фактично рекомендувати відповідні типи тренажерів.

Інтерфейс вибору вправ реалізовано у вигляді набору плиток, що представляють різні типи тренажерів.

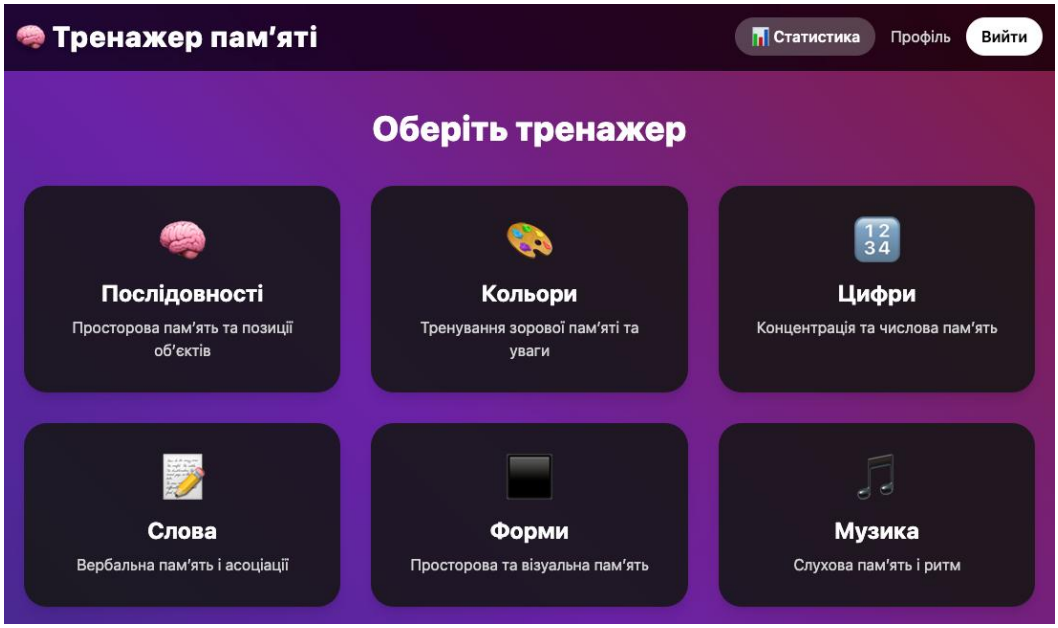


Рисунок 3.6 – Головне меню вибору тренажерів пам’яті

Користувач має доступ до таких режимів, як послідовності, кольори, цифри, слова, форми та музика. Така організація дозволяє швидко знаходити потрібний тип вправ, а також створює основу для подальшої персоналізації — наприклад, відображення більш релевантних тренажерів залежно від попередніх результатів.

Ключовим елементом оптимізації є сторінка статистики, яка виконує роль аналітичного модуля. Вона відображає результати тренувань у вигляді графіків і гістограм, що дозволяє як користувачу, так і системі оцінювати прогрес.

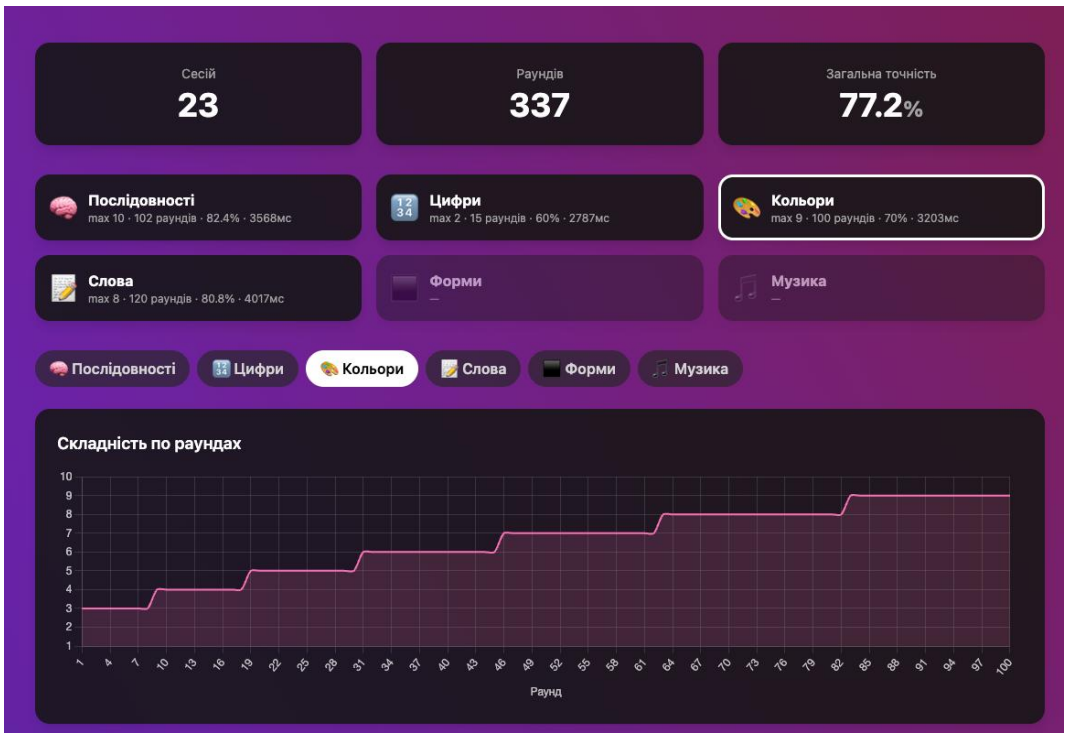


Рисунок 3.7 – Сторінка статистики та прогресу користувача

На сторінці представлено лінійний графік прогресу (складність, точність, час реакції), а також гістограми розподілу показників. Такі візуалізації дозволяють виявляти закономірності у поведінці користувача та використовувати їх для подальшої оптимізації.

Одним із ключових обчислюваних показників є інтегральний коефіцієнт ефективності (Memory Score), який поєднує точність і швидкість реакції. Його використання дозволяє узагальнити результати та застосовувати їх для ранжування або аналізу.

Лістинг 3.9 – Обчислення інтегрального показника Memory Score

```
function memory_score($accuracy, $reaction){
    $reactionScore = max(0, 100 - ($reaction / 50));
    return round(($accuracy * 0.6) + ($reactionScore * 0.4), 2);
}
```

Крім цього, для аналізу розподілу результатів використовуються гістограми, які дозволяють групувати дані за діапазонами значень. Це дає змогу оцінити не лише середні показники, але й варіативність результатів користувача.

Лістинг 3.10 – Формування гістограм точності та реакції

```
$accuracyBuckets = array_fill(0,5,0);
$reactionBuckets = array_fill(0,5,0);

foreach($sessionsAll as $s){
    $a = (int)($s['accuracy'] ?? 0);
    $r = (int)($s['avg_reaction_ms'] ?? 0);

    $accuracyBuckets[min(floor($a/20),4)]++;
    $reactionBuckets[min(floor($r/1000),4)]++;
}
```

Хоча у системі не використовується класичне машинне навчання у вигляді нейронних мереж або моделей, реалізований підхід відповідає принципам data-driven адаптації. Система аналізує накопичені дані, виявляє залежності та використовує їх для покращення взаємодії з користувачем.

3.6 Реалізація підсистеми автентифікації та профілю

Підсистема автентифікації у вебтренажері пам’яті реалізована як окремий функціональний компонент, що забезпечує ідентифікацію користувача, керування сесіями та доступ до персоналізованих даних. Основною її метою є обмеження доступу до функціоналу системи, збереження результатів тренувань та підтримка індивідуального адаптивного процесу навчання.

Інтерфейси авторизації та керування профілем реалізовані у вигляді окремих сторінок, що забезпечують взаємодію користувача із системою. На рисунку 3.8 зображено сторінку входу до системи разом із формою редагування профілю користувача.

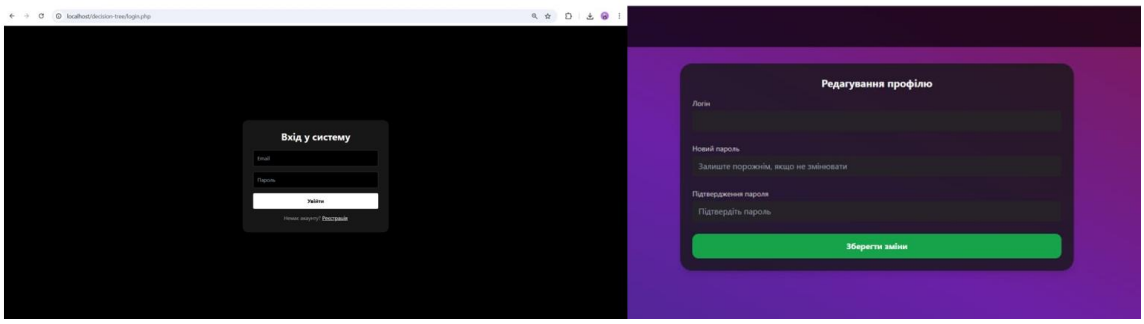


Рисунок 3.8 – Сторінка авторизації та редагування профілю користувача

Сторінка авторизації містить форму введення електронної пошти та пароля. Після введення даних виконується POST-запит до серверної частини, де відбувається перевірка коректності введеної інформації. Пароль не зберігається у відкритому вигляді — використовується механізм хешування, що забезпечує безпечне збереження облікових даних. Під час входу застосовується порівняння введеного значення із збереженим хешем, що гарантує захист конфіденційної інформації.

Після успішної авторизації створюється серверна сесія, у якій зберігається ідентифікатор користувача. Ця сесія використовується для контролю доступу до всіх функціональних модулів системи, включаючи тренажери, статистику та особистий кабінет.

Функціональність профілю дозволяє користувачу змінювати свої персональні дані та параметри безпеки. Інтерфейс редагування профілю містить поля для оновлення логіну та пароля, причому зміна пароля є необов’язковою. Перед збереженням змін виконується перевірка введених даних, після чого інформація оновлюється у базі даних із використанням підготовлених запитів.

Зберігання даних користувачів реалізовано у таблиці users, структура якої наведена на рисунку 3.9.

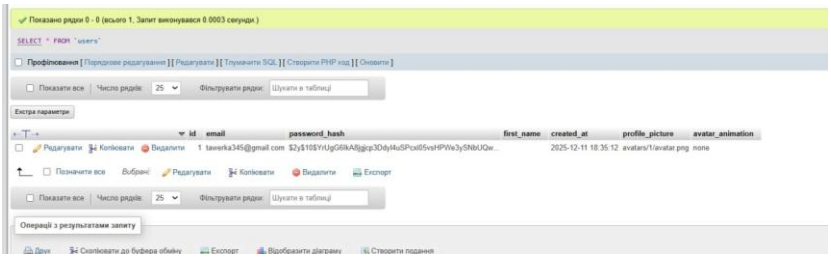


Рисунок 3.9 – Вміст таблиці users у базі даних

Таблиця містить основні атрибути користувача, включаючи електронну пошту, хеш пароля, дату реєстрації та додаткові параметри профілю, такі як аватар і налаштування його відображення. Використання хешування паролів і відсутність відкритих даних підвищує загальний рівень безпеки системи.

Взаємодія між клієнтською та серверною частинами відбувається за стандартною схемою: користувач вводить дані у форму, після чого вони передаються на сервер для обробки, перевірки та збереження. Такий підхід забезпечує цілісність даних і надійний контроль доступу.

3.7 Тестування програмного забезпечення та аналіз безпеки

У процесі розробки адаптивного вебтренажера пам’яті передбачалося виконання базового тестування програмного забезпечення з метою перевірки коректності роботи основних функціональних модулів та загальної стабільності системи. Основна увага приділялась перевірці ключових сценаріїв взаємодії користувача з додатком, зокрема проходженню тренажерів, обробці відповідей, роботі адаптивного алгоритму та збереженню результатів у базі даних.

Функціональна частина тестування охоплює перевірку роботи API, який забезпечує обробку тріалів та оновлення параметрів сесії. Зокрема, аналізується коректність прийому вхідних даних, їх формат та відповідність очікуваній структурі, а також формування відповіді для клієнтської частини у форматі JSON.

Лістинг 3.11 – Перевірка авторизації користувача в API

```
if (!is_logged()) {
    echo json_encode(['ok' => false, 'error' => 'Неавторизований']);
    exit;
}
```

Даний фрагмент демонструє базовий механізм контролю доступу до серверної логіки, який обмежує виконання операцій у випадку відсутності активної сесії користувача.

Окремо розглядається перевірка роботи адаптивного алгоритму, який відповідає за зміну рівня складності. У рамках тестових сценаріїв оцінюється поведінка системи при різних умовах виконання завдань, зокрема при високій точності відповідей, наявності помилок та різному часі реакції. Це дозволяє перевірити логіку зміни параметра difficulty в межах встановлених обмежень.

Також увага приділяється роботі з базою даних, зокрема запису результатів тріалів у таблицю trials та оновленню агрегованих показників у таблиці sessions. Використання параметризованих запитів розглядається як основний підхід до взаємодії з базою даних, що зменшує ризики некоректної обробки вхідних даних.

Лістинг 3.12 – Збереження тріалу в базу даних

```
INSERT INTO trials (session_id, sequence, shown_ms, user_response,
correct,                  reaction_ms,                  difficulty)
VALUES (?, ?, ?, ?, ?, ?, ?)
```

У межах аналізу безпеки враховуються базові механізми захисту, які реалізовані в системі. Зокрема, використовується перевірка автентифікації для доступу до захищених функцій, зберігання паролів у хешованому вигляді, а також перевірка вхідних даних на серверній стороні. Це дозволяє зменшити ймовірність некоректної обробки запитів та підвищити загальний рівень надійності системи.

Додатково передбачається обробка помилок у форматі JSON-відповідей, що забезпечує узгоджену взаємодію між серверною та клієнтською частинами навіть у випадку некоректних або неповних запитів.

У межах розробки було враховано основні аспекти тестування та безпеки, які дозволяють забезпечити базову стабільність роботи вебтренажера та створюють основу для подальшого вдосконалення системи.

3.8 Аналіз ефективності адаптивного алгоритму

Для обґрунтування ефективності адаптивного алгоритму було проведено порівняння прогресу користувача у вебтренажері пам’яті при використанні системи без адаптації складності та з реалізованим адаптивним алгоритмом. Аналіз здійснювався на основі зміни рівня складності під час проходження тренувань протягом 30 днів, та зображено на рисунку 3.10.

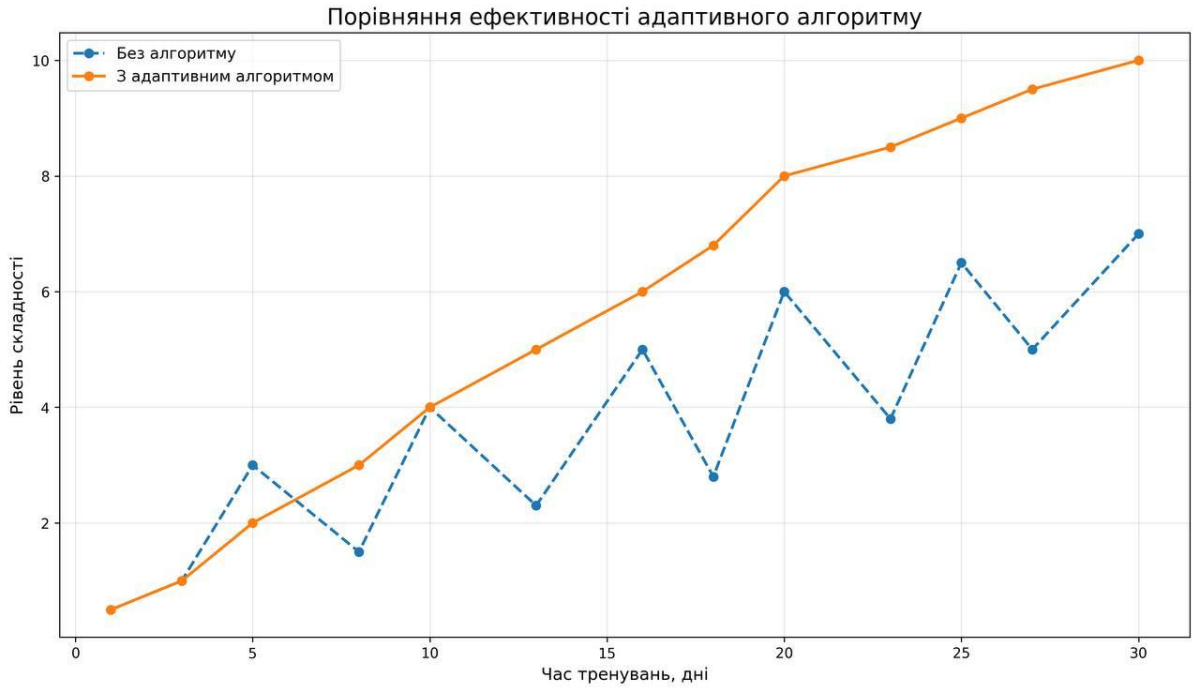


Рисунок 3.10 – Порівняння ефективності адаптивного алгоритму

На рисунку показано, що без використання адаптивного алгоритму прогрес користувача є нестабільним, а рівень складності часто змінюється нерівномірно.

Кількісне порівняння результатів показує, що на завершення 30-денного періоду тренувань користувач у режимі без адаптивного алгоритму досяг приблизно 7-го рівня складності, тоді як із використанням запропонованого алгоритму — 10-го рівня. Отже, підсумковий рівень складності в адаптивному режимі є вищим приблизно на 42,9 %. Крім того, графік без адаптації містить близько п’яти виражених спадів рівня складності після попереднього зростання, тоді як при використанні адаптивного алгоритму зниження складності не спостерігається. Це свідчить не лише про швидший, а й про стабільніший

характер прогресу користувача, що підтверджує доцільність застосування розробленого підходу до регулювання когнітивного навантаження.

3.9 Висновки до третього розділу

У межах даного розділу було здійснено практичну реалізацію основних функціональних компонентів адаптивного вебтренажера пам’яті, включаючи різні типи тренажерів, серверну обробку тріалів, роботу з базою даних та механізм адаптації рівня складності залежно від результатів користувача.

Розробка системи базується на клієнт-серверній архітектурі з чітким поділом логіки між інтерфейсом, API та базою даних, що забезпечує структурованість рішення, зручність його підтримки та можливість подальшого розширення функціоналу без порушення основної логіки роботи.

Реалізований адаптивний алгоритм дозволяє динамічно змінювати складність тренувань на основі точності відповідей, часу реакції та стабільності виконання завдань, що забезпечує індивідуальний підхід до користувача та поступове ускладнення вправ відповідно до його рівня.

Додатково передбачено збереження результатів тренувань у базі даних та їх подальший аналіз у вигляді статистичних показників, що дозволяє відстежувати прогрес користувача та оцінювати ефективність тренувального процесу в динаміці.

4 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ

4.1 Підвищення стійкості роботи промислових підприємств у воєнний час

Ефективність функціонування економіки держави значною мірою визначається здатністю окремих галузей господарства стабільно працювати не лише у звичайних умовах, а й під час надзвичайних ситуацій мирного та воєнного часу. Наслідки надзвичайних ситуацій можуть призводити до масштабних руйнувань, людських втрат і суттєвого скорочення обсягів промислового та сільськогосподарського виробництва, що негативно впливає на економічний потенціал держави. Саме тому важливим є завчасне впровадження заходів, спрямованих на забезпечення стійкої роботи об’єктів господарської діяльності в умовах надзвичайних ситуацій.

Знання можливих надзвичайних ситуацій, характерних для певної території чи виробництва, дозволяє більш ефективно планувати та реалізовувати заходи щодо попередження аварій, катастроф і стихійних лих або мінімізації їх наслідків. Під стійкістю роботи об’єкта господарської діяльності розуміють його здатність у надзвичайних умовах забезпечувати випуск продукції у встановленому обсязі та відповідній номенклатурі, а у випадку часткових руйнувань чи порушення постачання — швидко відновлювати виробництво власними ресурсами.

На стійкість функціонування об’єкта впливає комплекс факторів, серед яких рівень захисту персоналу від уражальних чинників надзвичайних ситуацій, здатність будівель, споруд, обладнання та інженерних мереж витримувати руйнівний вплив аварій, катастроф або сучасних засобів ураження, надійність забезпечення електроенергією, водою, паливом і матеріальними ресурсами, а також готовність до проведення аварійно-рятувальних та відновлювальних робіт. Важливу роль відіграє і ефективність управління виробничими процесами та системами цивільного захисту в умовах надзвичайних ситуацій.

Підвищення стійкості об’єктів господарської діяльності досягається шляхом реалізації комплексу інженерно-технічних, технологічних та організаційних заходів. Інженерно-технічні рішення спрямовані на підвищення надійності будівель, споруд, обладнання та комунально-енергетичних систем. Технологічні заходи забезпечують зменшення ймовірності розвитку аварійних ситуацій шляхом удосконалення виробничих процесів і спрощення технологічних схем. Організаційні заходи передбачають планування дій керівного складу, служб цивільного захисту та персоналу щодо захисту працівників, проведення рятувальних робіт і оперативного відновлення виробництва.

В останні роки особлива увага приділяється питанням екологічної безпеки та забезпечення стійкості функціонування об’єктів господарської діяльності в умовах надзвичайних ситуацій. Актуальність цих питань зростає через складність сучасних виробничих процесів і високий рівень взаємозалежності між різними галузями промисловості. Забезпечення стабільної роботи підприємств у воєнний час є важливою умовою підтримання економіки держави, збереження виробничого потенціалу та безперервності виробничих ланцюгів для потреб цивільного населення і держави.

Стійкість функціонування об’єктів у надзвичайних умовах визначається рівнем захисту персоналу, надійністю технічних систем, стабільністю постачання необхідних ресурсів, готовністю до відновлення виробництва та ефективністю управління. Розв’язання цих завдань потребує комплексного підходу до організації виробництва та систематичної підготовки підприємств до можливих надзвичайних ситуацій. Важливою складовою загальної стійкості підприємств є також фінансова стабільність та забезпечення належного рівня соціального захисту працівників.

Для об’єктів, діяльність яких не пов’язана безпосередньо з виробництвом матеріальних цінностей, зокрема транспорту чи зв’язку, стійкість роботи визначається здатністю виконувати свої функції в умовах надзвичайних ситуацій. Для економіки держави загалом стійкість функціонування означає можливість забезпечення життєдіяльності країни, роботи енергетики,

транспорту, зв’язку та виробництва продукції навіть у складних умовах воєнного часу чи інших надзвичайних обставин.

Основними напрямами підвищення стійкості роботи промислових об’єктів є забезпечення надійного захисту працівників, захист виробничих фондів від уражальних факторів, підвищення ефективності управління виробництвом і системами цивільного захисту, забезпечення стабільного постачання ресурсів та підготовка підприємств до проведення відновлювальних робіт. Реалізація цих заходів потребує завчасного проведення інженерно-технічних, технологічних та організаційних рішень, спрямованих на підвищення міцності споруд, удосконалення технологічних процесів і планування дій персоналу в умовах надзвичайних ситуацій.

Підвищення стійкості підприємств досягається насамперед посиленням найбільш вразливих елементів виробничої системи. Для цього проводяться відповідні дослідження, на основі яких плануються організаційні та технічні заходи. Сучасні досягнення науки й техніки дозволяють створювати об’єкти, здатні витримувати значні навантаження та руйнівні впливи, однак реалізація таких рішень потребує значних матеріальних витрат і є доцільною переважно для особливо важливих виробничих об’єктів. При проектуванні та реконструкції промислових споруд доцільним є використання високоміцних і стійких матеріалів, що дозволяють підвищити загальний рівень стійкості підприємств та забезпечити їх надійне функціонування в умовах надзвичайних ситуацій.

4.2 Заходи, що покращують умови праці оператора

Трудове законодавство визначає, що роботодавець зобов’язаний щороку забезпечувати виконання заходів, спрямованих на покращення умов праці та підвищення рівня охорони праці, у тому числі тих, які були розроблені за результатами атестації робочих місць та оцінювання професійних ризиків. Для реалізації цих вимог формується перелік заходів, що мають виконуватися роботодавцем з метою зниження рівня професійних ризиків і підвищення безпеки працівників.

До таких заходів належать проведення атестації робочих місць та оцінювання професійних ризиків, реалізація рішень щодо покращення умов праці, а також упровадження систем автоматичного й дистанційного керування виробничим обладнанням, технологічними процесами, підйомними та транспортними механізмами. Значна увага приділяється встановленню систем аварійної сигналізації, засобів аварійної зупинки обладнання та пристроїв, які запобігають виникненню небезпечних ситуацій у разі перебоїв енергопостачання або його відновлення.

Важливим напрямом є облаштування захисних огорож виробничого обладнання від дії рухомих елементів і можливого розльоту предметів, а також використання блокувальних, герметизуючих та інших захисних пристроїв. Крім цього, здійснюється модернізація або встановлення нових засобів колективного захисту працівників від впливу небезпечних і шкідливих виробничих факторів. На виробниче обладнання, елементи конструкцій, комунікації та органи управління наносяться сигнальні кольори й знаки безпеки для підвищення рівня безпечної експлуатації.

У виробничих умовах також впроваджуються системи автоматичного контролю небезпечних і шкідливих факторів на робочих місцях, а також технічні засоби, що забезпечують захист працівників від ураження електричним струмом. Для безпечної експлуатації виробничих комунікацій, обладнання та споруд використовуються запобіжні, сигнальні й захисні пристрої, призначені для аварійного захисту парових, газових, водяних, кислотних, лужних та інших систем. Окрема увага приділяється механізації та автоматизації технологічних процесів, пов’язаних із зберіганням, транспортуванням і використанням токсичних, агресивних, легкозаймистих і горючих речовин.

Конкретний перелік заходів щодо покращення умов і охорони праці визначається роботодавцем з урахуванням особливостей діяльності підприємства та специфіки виробничих процесів. Фінансування заходів з охорони праці здійснюється роботодавцем і має становити не менше 0,2 % від суми витрат на виробництво продукції, виконання робіт або надання послуг. При

цьому працівники не несуть витрат, пов’язаних із реалізацією заходів щодо покращення умов та безпеки праці.

4.3 Значення автоматизації виробничих процесів в питаннях охорони праці

Процес автоматизації виробництва протягом тривалого часу залишається одним із найважливіших напрямів технічного прогресу. Розвиток автоматизованих систем дозволив значно зменшити безпосередню участь людини у виробничих процесах, що сприяло покращенню умов праці та створило можливості для виконання складніших завдань. У сучасних умовах автоматизовані системи не лише замінюють фізичну працю людини, а й виконують функції управління виробничими процесами.

Під час розроблення промислових роботів і автоматизованих систем необхідно враховувати особливості їх конструкції, функціонування, динаміки руху та алгоритмів керування робочими органами. Важливу роль відіграє правильне проектування засобів захисту працівників і обслуговуючого персоналу, оскільки помилки в організації автоматизованих процесів можуть знизити рівень безпеки та створити додаткові ризики. Засоби захисту повинні враховувати можливість перебування персоналу в робочій зоні автоматизованих систем під час їх запуску, програмування, технічного обслуговування та контролю роботи.

У автоматизованому виробництві процеси отримання, перетворення, передавання та використання енергії, матеріалів і інформації здійснюються переважно в автоматичному режимі. При цьому обслуговуючий персонал виконує функції налагодження обладнання та систем керування. Автоматизація є закономірним продовженням механізації виробничих процесів і водночас новим етапом розвитку виробництва. Її впровадження дозволяє підвищити продуктивність обладнання, зменшити собівартість продукції, скоротити кількість браку, підвищити рівень безпеки праці та покращити санітарний стан виробничих приміщень.

Розвиток автоматизованих систем суттєво впливає на технічний прогрес, оскільки сучасні технологічні рішення створюються з урахуванням комплексного підходу до автоматичного управління. Такі системи широко використовуються на складних виробництвах для забезпечення безперервного контролю та точного регулювання параметрів технологічних процесів, що позитивно впливає на якість готової продукції. Важливим чинником удосконалення виробничих процесів є також цифровізація, яка забезпечує ефективніше управління та контроль виробничої діяльності.

4.4 Висновки до четвертого розділу

У даному розділі було розглянуто основні підходи до забезпечення стійкості роботи промислових підприємств у надзвичайних ситуаціях, а також заходи, спрямовані на покращення умов праці та підвищення рівня безпеки виробничих процесів. Проведений аналіз показав, що стійке функціонування об’єктів господарської діяльності залежить від рівня захисту персоналу, надійності технічних систем, ефективності управління та готовності підприємства до відновлення роботи в умовах надзвичайних ситуацій.

У роботі також було розглянуто основні організаційні, технічні та технологічні заходи, які сприяють зниженню професійних ризиків і покращенню умов праці працівників. Встановлено, що важливу роль у забезпеченні безпечної експлуатації обладнання відіграють системи автоматичного контролю, захисні пристрої, модернізація засобів колективного захисту та впровадження сучасних технологій управління виробничими процесами.

Окрема увага приділена значенню автоматизації виробничих процесів у питаннях охорони праці. Автоматизація дозволяє зменшити фізичне навантаження на працівників, підвищити продуктивність праці, покращити якість продукції та забезпечити більш безпечні умови виробничої діяльності. Водночас ефективність автоматизованих систем значною мірою залежить від правильного проектування засобів захисту та організації взаємодії персоналу з автоматизованим обладнанням.

ВИСНОВКИ

У кваліфікаційній роботі досягнуто поставленої мети — розроблено та досліджено адаптивний вебтренажер пам’яті, у якому регулювання складності тренувальних вправ здійснюється на основі аналізу результатів діяльності користувача. У процесі виконання роботи проаналізовано підходи до побудови адаптивних навчальних вебсистем, особливості організації тренувального процесу у вебтренажерах пам’яті, методи обробки результатів і використання метрик ефективності для підтримання належного рівня когнітивного навантаження.

Обґрунтовано архітектуру програмної системи та розроблено структуру вебдодатка, що забезпечує взаємодію клієнтської і серверної частин, збереження результатів тренувань у базі даних, автентифікацію користувачів і подальший аналіз накопиченої інформації. Реалізовано комплекс інтерактивних вправ для тренування пам’яті, модулі збору й обробки результатів, сторінку статистики прогресу та алгоритм адаптивної зміни складності. Запропонований алгоритм враховує точність відповідей, час реакції та стабільність виконання завдань, що дозволяє персоналізувати тренувальний процес і поступово змінювати навантаження відповідно до поточного рівня користувача.

Проведена перевірка роботи програмного продукту підтвердила коректність реалізації основних функціональних компонентів, зокрема обробки тренувальних спроб, збереження результатів, зміни рівня складності та формування статистичних показників. За результатами порівняльного аналізу встановлено, що на завершення 30-денного періоду тренувань у режимі без адаптивного алгоритму досягнуто приблизно 7-го рівня складності, тоді як із використанням розробленого алгоритму — 10-го рівня. Це відповідає підвищенню підсумкового рівня складності приблизно на 42,9 %. Також у неадаптивному режимі зафіксовано близько п’яти спадів складності після попереднього зростання, тоді як при використанні адаптивного алгоритму таких спадів не спостерігається. Отримані результати свідчать про більш ефективний і

стабільний характер прогресу користувача за умов адаптивного регулювання когнітивного навантаження.

Наукова новизна одержаних результатів полягає у вдосконаленні підходу до адаптивного регулювання когнітивного навантаження у вебтренажері пам’яті шляхом поєднаного використання показників точності виконання, швидкості реакції та стабільності результатів для динамічного налаштування складності вправ.

ПЕРЕЛІК ДЖЕРЕЛ

1. Любомир Матійчук, Володимир Готович, Віталій Бонар (Статті. ТНТУ ім.І.Пулюя), Порівняння ефективності методів некерованого машинного навчання для виявлення аномалій в obd2 даних.

2. Л.В. Волинець, Н.А. Гарматюк, В.А Готович (Статті. ТНТУ ім.І.Пулюя), Великі за обсягом набори біомедичних даних та машинне навчання.

3. В.А Готович, А.В Мачужак (Матеріали XI Міжнародної науково-практичної конференції молодих учених та студентів „ Актуальні задачі сучасних технологій “), Застосування методології CI/CD для автоматизації процесів тестування та розгортання програмного забезпечення.

4. Grigorii Shymchuk, Iaroslav Lytvynenko, Roman Hromyak, Sergii Lytvynenko, Volodymyr Hotovych, CITI, 2023, Gas Consumption Forecasting Using Machine Learning Methods and Taking Into Account Climatic Indicators.

5. Vyacheslav Nykytyuk, Vasyl Dozorskyy, Nataliia Kunanets, Volodymyr Pasichnyk, Oleksandr Matsiuk, Ihor Bodnarchuk: Electrical Probe-Signal Processing and Criterion for the Determination of Time Parameters of the Teeth Filling Material Polymerization Process in Dentistry. 4th IDDM 2021: Valencia, Spain. P. 54-63

6. Oleksii Duda, Nataliia Kunanets, Serhii Martsenko, Vyacheslav Nykytyuk, Volodymyr Pasichnyk. Information technology platform for the selection and analytical processing of information on COVID-19. 2021 IEEE 16th International Conference on Computer Sciences and Information Technologies (CSIT). Volume 2, Lviv, Ukraine 22-25 Sept. 2021. P. 231-328. Electronic ISBN:978-1-6654-4257-2, Print on Demand(PoD) ISBN:978-1-6654-4258-9, Electronic ISSN: 2766-3639, Print on Demand(PoD) ISSN: 2766-3655. DOI: 10.1109/CSIT52700.2021.9648839.

7. Oleksii Duda, Nataliia Kunanets, Serhii Martsenko, Vyacheslav Nykytyuk, Volodymyr Pasichnyk. COVID-19 data collections and analytical processing. 2021 IEEE 16th International Conference on Computer Sciences and Information Technologies (CSIT). Volume 2, Lviv, Ukraine 22-25 Sept. 2021. P. 252-257. Electronic ISBN:978-1-6654-4257-2, Print on Demand (PoD) ISBN:978-1-6654-

4258-9, Electronic ISSN: 2766-3639, Print on Demand (PoD) ISSN: 2766-3655. DOI: 10.1109/CSIT52700.2021.9648839.

8. Vyacheslav Nykytyuk, Vasil Dozorskyi, Oksana Dozorska, Andrii Karnaukhov and Liubomyr Matiichuk. The Method of User Identification by Speech Signal. The 2nd International Workshop on Information Technologies: Theoretical and Applied Problems (ITTAP-2022) Ternopil, Ukraine, November 22-24, 2022. Vol-3309 urn:nbn:de:0074-3309-1. P.225-232. ISSN 1613-0073 DOI: 10.1425/jsdtl.

9. hor Bodnarchuk, Yuriy Skorenkyy, Taras Kramar, Oleksii Duda and Vyacheslav Nykytyuk. Use of Analytical Hierarchy Process in Scenarios Design for a Digital Museum with XR components. The 2nd International Workshop on Information Technologies: Theoretical and Applied Problems (ITTAP-2022) Ternopil, Ukraine, November 22-24, 2022. Vol-3309 urn:nbn:de:0074-3309-1. P. 414-425. ISSN 1613-0073 DOI: 10.1425/jsdtl.

10. Як працює персоналізація в онлайн навчанні, URL: https://kwiga.com/ua/blog/yak-pracyuye-personalizaciya-v-onlajn-navchanni (дата звернення: 16.05.2026)

11. Динамічний контект: визначення, типи та поради для впровадження, URL: https://www.promodo.ua/blog/dinamichniy-kontent-viznachennya-tipi-ta-poradi-dlya-provadzhennya (дата звернення: 16.05.2026)

12. Клієнт-серверна архітектура, URL: https://training.qatestlab.com/blog/technical-articles/client-server-architecture/ (дата звернення: 16.05.2026)

13. Building Adaptive Learning Engines Using Real-Time Data, URL: https://6b.education/insight/edtech-development-with-ai-building-adaptive-learning-engines-using-real-time-data/ (дата звернення: 16.05.2026)

14. Гейміфікація в освіті як засіб підвищення її ефективності, URL: https://bestcleverslms.com/piznavalne/heymifikatsiia-v-osviti-iak-zasib-pidvyshchennia-ii-efektyvnosti/ (дата звернення: 16.05.2026)

15. Аналітика клієнтського досвіду: відображення та оптимізація багатоканального шляху користувача, URL: https://doisz.com/uk/blog/user-experience/ (дата звернення: 16.05.2026)

16. Обробка даних: Визначення, Етапи та Використання в Аналітиці, URL: https://bitimpulse.com/obrobka-danyh/ (дата звернення: 16.05.2026)

17. Розбір сховищ даних у браузері, URL: https://blog.ithillel.ua/articles/cookies-indexeddb (дата звернення: 16.05.2026)

18. The role of timestamps, URL: https://www.lenovo.com/in/en/glossary/timestamp/?orgRef=https%253A%252F%252Fwww.google.com%252F (дата звернення: 16.05.2026)

19. Lumosity, URL: https://app.lumosity.com/landing (дата звернення: 07.05.2026)

20. Cognifit, URL: https://www.cognifit.com/ru/en (дата звернення: 16.05.2026)

21. BrainHQ, URL: https://www.brainhq.com/ (дата звернення: 16.05.2026)

22. Peak, URL: https://www.peak.net/ (дата звернення: 16.05.2026)

23. Ієрархічна база даних, URL: https://bazidanih5.blogspot.com/p/blog-page.html (дата звернення: 16.05.2026)

24. Розуміння агрегації даних, URL: https://www.vpnunlimited.com/ua/help/cybersecurity/data-aggregation?srsltid=AfmBOooD36uYT1bTqmkxPSO4fzHiC6HIk7-lY5B5rcINn-LGwNdoTZhg (дата звернення: 16.05.2026)

25. Метрики та ключові показники ефективності, URL: https://www.maxzosim.com/key-performance-indicators-kpis/ (дата звернення: 16.05.2026)

26. Основні типи архітектури програмного забезпечення, URL: https://www.artofba.com/post/main-types-of-software-architecture (дата звернення: 16.05.2026)

27. Про архітектуру додатків, URL: https://foxminded.ua/arkhitektura-zastosunku/ (дата звернення: 16.05.2026)

28. Що таке орієнтована на події архітектура?, URL: https://www.sap.com/ukraine/resources/what-is-event-driven-architecture (дата звернення: 16.05.2026)

29. PHP docs, URL: https://www.php.net/docs.php (дата звернення: 16.05.2026)

30. Application Programming Interface (API): що це, як і де працює, URL: https://goit.global/ua/articles/application-programming-interface-api-shcho-tse-iak-i-de-pratsiuie/ (дата звернення: 16.05.2026)

31. SQL ON DELETE CASCADE: Automatically Remove Dependent Data, URL: https://www.datacamp.com/tutorial/sql-on-delete-cascade (дата звернення: 16.05.2026)

32. HTTP, URL: https://code.tutsplus.com/uk/http-the-protocol-every-web-developer-must-know-part-1--net-31177t (дата звернення: 16.05.2026)

33. What Is Adaptive AI in Gaming?, URL: https://www.hp.com/us-en/shop/tech-takes/adaptive-ai-in-games-explained (дата звернення: 16.05.2026)

34. Fetch API, URL: https://developer.mozilla.org/en-US/docs/Web/API/Fetch_API (дата звернення: 16.05.2026)

35. Роберт Мартін. Чиста архітектура. Мистецтво розробки програмного забезпечення / Роберт Мартін, 2019 – с. 368.

36. О.Д. Ситнікова. Інтелектуальні системи навчання: проектування та реалізація / О.Д. Ситнікова, 2018 – с. 192.

37. В.П. Пасічник. Організація баз даних та знань / В.П. Пасічник, 2017 – с. 356.

38. В.В. Пасічник. Інформаційні технології розробки веб-систем : навч. посіб. / В.В. Пасічник, О.В. Пасічник, 2021 – с. 340.

39. C. Richardson. Microservices Patterns: With examples in Java / C. Richardson, 2018 – с. 520.

40. Robin Nixon. Learning PHP, MySQL & JavaScript / Robin Nixon, 2021 – с. 832.

41. K.M. Kapp. The Gamification of Learning and Instruction: Game-based Methods and Strategies for Training and Education / K.M. Kapp, 2012 – с. 336.

42. П. Брусіловський. Адаптивні гіпермедіа системи / П. Брусіловський, 2011 – с. 280.

43. D. Flanagan. JavaScript: The Definitive Guide. 7th ed. / D. Flanagan, 2020 – с. 704.

44. N. Marz. Big Data: Principles and practices of scalable realtime data systems / N. Marz, J. Warren, 2015 – с. 328.

45. К. Шифлетт. Безпека Web-застосувань на PHP (Essential PHP Security) / К. Шифлетт, 2005 – с. 128.

46. Е. Дастін. Автоматизоване тестування програмного забезпечення / Е. Дастін, Дж. Решкі, Дж. Пол.

47. M. Kleppmann. Making Sense of Stream Processing / M. Kleppmann, 2016 – с. 74.

48. T. Geig. Designing Web APIs: Building APIs That Developers Love / T. Geig, 2018 – с. 210.

49. Kryazhych O., Itskovych V., Iushchenko K., Hrytsyshyna V., Bruvier D., Nykytyuk V., Bodnarchuk I. (2023) The use of abstract moore automaton to control the sensors of a service-oriented alarm and emergency notification network. Scientific Journal of TNTU (Tern.), vol 109, no 1, pp. 111–120. ISSN 2522-4433

50. Dediv, L., Dozorska, O., Kukuruza, V., Nykytyuk, V., Kovalyk, S. Computer Simulation Modeling of Voice Signals in the Matlab Environment for the Task of Computerized Diagnostic Systems Testing. The 1st International Workshop on “Computer information technologies in Industry 4.0” (CITI-2023) will be held in Ternopil, Ukraine, from June 14 to 16, 2023. The Workshop is organized by the Faculty of Applied Information Technologies and Electrical Engineering of Ternopil Ivan Puluj National Technical University. 2023, 3468, pp. 257–262. Vol-3468 urn:nbn:de:0074-3468-8, ISSN 1613-0073

51. Dozorskyi, V., Dediv, I., Sverstiuk, S., Nykytyuk, V., Karnaukhov, A. The Method of Commands Identification to Voice Control of the Electric Wheelchair. The Workshop is organized by the Faculty of Applied Information Technologies and Electrical Engineering of Ternopil Ivan Puluj National Technical University. The 1st International Workshop on “Computer information technologies in Industry 4.0” (CITI-2023) will be held in Ternopil, Ukraine, from June 14 to 16, 2023. The Workshop is organized by the Faculty of Applied Information Technologies and

Electrical Engineering of Ternopil Ivan Puluj National Technical University. *2023,* 3468, pp. 233–240. Vol-3468 urn:nbn:de:0074-3468-8, ISSN 1613-0073

52. Sverstiuk, A., Matiichuk, L., Polyvana, U., Stanko, A., Nykytyuk, V.. Analytical analysis of approaches to assessing the quality of life in smart cities. BAIT'2024: The 1st International Workshop on "Bioinformatics and applied information technologies", October 02-04, 2024, Zboriv, Ukraine. CEUR Workshop Proceedings, 2024, 3842, pp. 75–91. ISSN: 1613-0073

53. Koroliuk, R., Nykytyuk, V., Tymoshchuk, V., Soyka, V., Tymoshchuk, D.. Automated monitoring of bee colony movement in the hive during winter season. BAIT'2024: The 1st International Workshop on "Bioinformatics and applied information technologies", October 02-04, 2024, Zboriv, Ukraine. CEUR Workshop Proceedings, 2024, 3842, pp. 147–156. ISSN: 1613-0073

54. Oleh Yasniy, Iryna Didych, Dmytro Tymoshchuk, Iaroslav Pasternak, Vyacheslav Nykytyuk, Hryhorii Shymchuk, Dmytro Radyk. Fatigue crack growth prediction of automotive steels using ensemble-based machine learning methods. Procedia Structural Integrity, VIII International Conference "In-service Damage of Materials: Diagnostics and Prediction" Volume 81, (15 -17 October 2025.) 2026, P.116-122.

ДОДАТКИ

Додаток А1 – Тези конференції 1

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Тернопільський національний технічний університет імені Івана Пулюя
Маріборский університет (Словенія)
Технічний університет у Кошице (Словаччина)
Вільнюський технічний університет ім. Гедимінаса (Литва)
Краківський економічний університет (Польща)
Вроцлавський економічний університет (Польща)
Університет «Опольська Політехніка» (Польща)
Національний університет «Полтавська політехніка імені Юрія Кондратюка»
Вінницький національний аграрний університет
Львівський національний університет ім. І. Франка
Головне управління Пенсійного фонду в Тернопільській області
Наукове товариство ім. Шевченка
Тернопільський обласний комунальний інститут післядипломної педагогічної освіти
Сумський державний педагогічний університет
Запорізький національний університет

АКТУАЛЬНІ ЗАДАЧІ СУЧАСНИХ ТЕХНОЛОГІЙ

Збірник

тез доповідей

XIV Міжнародної науково-технічної конференції молодих учених та студентів

11-12 грудня 2025 року

УКРАЇНА
ТЕРНОПІЛЬ – 2025

УДК 004.4
Р.В. Лесик
(Тернопільський національний технічний університет імені Івана Пулюя, Україна)

ДОСЛІДЖЕННЯ АДАПТИВНИХ АЛГОРИТМІВ РЕГУЛЮВАННЯ КОГНІТИВНОГО НАВАНТАЖЕННЯ У ВЕБТРЕНАЖЕРІ ПАМ'ЯТІ

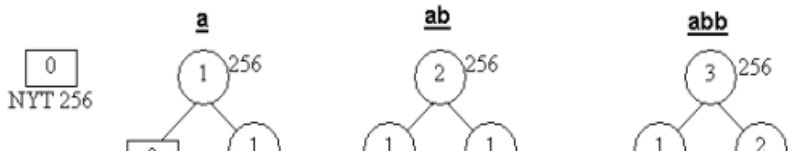
R.V. Lesyk

RESEARCH ON ADAPTIVE ALGORITHMS FOR COGNITIVE LOAD REGULATION IN A WEB MEMORY TRAINER

Адаптивні математичні моделі та алгоритми регулювання когнітивного навантаження у вебтренажерах пам'яті є актуальною задачею сучасних технологій, пов'язаною з розробкою інтелектуальних систем навчання та цифрових платформ для персоналізованого розвитку когнітивних здібностей. Вони забезпечують динамічне підлаштування складності завдань відповідно до індивідуальної продуктивності користувача, підвищуючи ефективність тренування короткочасної та робочої пам'яті. Використання таких алгоритмів у вебсередовищі дозволяє реалізувати моніторинг показників продуктивності, автоматичну корекцію рівня складності завдань та аналіз динаміки навантаження у реальному часі. У межах цієї роботи планується дослідження методів побудови адаптивних алгоритмів, оцінка їх ефективності у вебтренажері та визначення перспектив подальшого розвитку інтелектуальних навчальних систем [1].

Для реалізації адаптивного регулювання складності завдань у вебтренажері пам'яті використовуватимуться математичні моделі, що оцінюють ймовірність правильної відповіді та динаміку попередніх результатів користувача.

Такий підхід дозволяє підвищити ефективність навчання, а також створює платформу для подальшого впровадження методів штучного інтелекту та машинного навчання для прогнозування продуктивності користувачів і оптимізації структури тренувальних сесій [2]. Це дозволяє не лише прогнозувати можливі помилки, а й ефективно кодувати інформацію про завдання та їх складність для швидкого доступу алгоритму. Враховуючи потребу в оптимальному представленні частотних характеристик завдань та ефективному зберіганні інформації, подальше дослідження може бути пов'язане з використанням алгоритму Хаффмана для кодування та управління структурою завдань у вебтренажері.



[3].

Реалізація адаптивного алгоритму у вебтренажері передбачає багаторівневу структуру програмного забезпечення, що включає модулі обробки результатів користувача, генерації завдань та корекції рівня складності. Дані про виконання завдань зберігаються у структурованому вигляді, що дозволяє швидко отримувати статистику для подальшого аналізу та прийняття рішень алгоритмом. Використання алгоритму Хаффмана дозволяє оптимально кодувати інформацію про частотні характеристики завдань, забезпечуючи економне зберігання та швидкий доступ до даних під час динамічної зміни складності. Така архітектура відкриває можливість інтеграції сучасних вебтехнологій та графічних бібліотек для створення інтерактивного та візуально адаптивного середовища, де алгоритм може в реальному часі регулювати складність завдань залежно від продуктивності користувача [4].

Алгоритм керує структурою завдань, їх частотою та складністю на основі показників продуктивності користувача, тоді як інтерфейс забезпечує реальне відображення цих змін у режимі реального часу. Така інтеграція дозволяє не лише ефективно кодувати та зберігати інформацію, а й створювати персоналізовані тренувальні сесії, де складність підлаштовується під індивідуальні здібності користувача, підвищуючи ефективність навчання та мотивацію до тренування.

У межах дослідження актуальних задач сучасних технологій для адаптивного алгоритму вебтренажера пам'яті особлива увага має приділятися аналізу його впливу на продуктивність користувачів та ефективність навчального процесу. Оцінка

ефективності адаптивного алгоритму у вебтренажері пам'яті проводиться шляхом аналізу показників продуктивності користувачів, таких як точність відповідей, швидкість реакції та динаміка прогресу протягом тренувальних сесій. Використання структурованих даних та частотних характеристик завдань дозволяє порівнювати адаптивний підхід із традиційним статичним розподілом завдань, визначаючи переваги персоналізованої регуляції складності. Експериментальна оцінка показала, що інтеграція адаптивних моделей та алгоритму Хаффмана забезпечує більш рівномірний розподіл когнітивного навантаження та підвищує ефективність тренування пам'яті. Отримані результати демонструють перспективність впровадження таких алгоритмів у сучасні інтелектуальні навчальні системи, а також можливість подальшої оптимізації та масштабування для широкого кола користувачів.

Література

1. Алгоритми та структура даних URL: https://dou.ua/lenta/articles/what-you-should-know-about-algorithms/. Доступ до ресурсу: 30.11.2025

2. Pedro Domingos. The Master Algorithm: How the Quest for the Ultimate Learning Machine Will Remake Our World / Pedro Domingos.

3. Дослідження оптимального коду Хоффмана URL: https://naurok.com.ua/metodichni-vkazivki-schodo-vikonannya-praktichno-roboti-doslidzhennya-optimalnogo-kodu-haffmana-359165.html /(дата звернення: 30.11.2025)

4. Жихарський П.О. Дослідження методів та алгоритмів компресії

Додаток А2 – Тези конференції 2

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ТЕРНОПІЛЬСЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ
ІМЕНІ ІВАНА ПУЛЮЯ

М А Т Е Р І А Л И

XIII НАУКОВО-ТЕХНІЧНОЇ КОНФЕРЕНЦІЇ

«ІНФОРМАЦІЙНІ МОДЕЛІ, СИСТЕМИ ТА ТЕХНОЛОГІЇ»

17-18 грудня 2025 року

ТЕРНОПІЛЬ
2025

УДК 004.4
Р. Лесик
(Тернопільський національний технічний університет імені Івана Пулюя, Україна)

ДОСЛІДЖЕННЯ АДАПТИВНИХ АЛГОРИТМІВ РЕГУЛЮВАННЯ КОГНІТИВНОГО НАВАНТАЖЕННЯ У ВЕБТРЕНАЖЕРІ ПАМ'ЯТІ

UDC 004.4
R. Lesyk

RESEARCH ON ADAPTIVE ALGORITHMS FOR COGNITIVE LOAD REGULATION IN A WEB MEMORY TRAINER

Розробка вебтренажерів пам'яті з адаптивним регулюванням складності завдань передбачає динамічне підлаштування рівня завдань відповідно до продуктивності користувача. Алгоритми аналізують час реакції, точність відповідей та частоту помилок, що дозволяє коригувати тренувальний процес у реальному часі та прогнозувати продуктивність користувача [1].

Математичні моделі оцінюють ймовірність правильної відповіді та динаміку результатів. На їх основі формується адаптивний алгоритм, який регулює складність завдань та оптимізує когнітивне навантаження. Використання принципів алгоритму Хаффмана дозволяє впорядкувати завдання за частотою та складністю, забезпечуючи швидкий доступ до необхідної інформації [2].

Вебтренажер з адаптивним алгоритмом включає модулі обробки даних користувача, генерації та відбору завдань, аналізу результатів та корекції рівня складності. Дані про виконання завдань структуровано зберігаються, що дозволяє алгоритму швидко визначати наступне завдання та коригувати тренувальний процес у реальному часі. Подібна архітектура забезпечує масштабованість системи та відкриває можливості для інтеграції додаткових методів штучного інтелекту та машинного навчання для прогнозування продуктивності [3].

Експериментальна оцінка алгоритму передбачає аналіз точності відповідей та швидкості реакції користувачів. Порівняння з традиційним статичним підходом показує переваги адаптивного регулювання складності та рівномірного розподілу когнітивного навантаження, підвищуючи ефективність тренувань.

Перевагами підходу є персоналізація тренувальних сесій, оптимізація ресурсів та інтеграція додаткових алгоритмів аналізу продуктивності. Водночас виклики залишаються: складність побудови адаптивних моделей, обробка великого обсягу даних та забезпечення безпеки інформації. Подальші дослідження можуть зосередитися на вдосконаленні кодування завдань та інтеграції нових методів адаптації [4].

Література

1. Алгоритми та структура даних [Електронний ресурс] : https://dou.ua/lenta/articles/what-you-should-know-about-algorithms/. Доступ до ресурсу: 09.12.2025.
2. Дослідження оптимального коду Хоффмана [Електронний ресурс] : https://naurok.com.ua/metodichni-vkazivki-schodo-vikonannya-praktichno-roboti-doslidzhennya-optimalnogo-kodu-haffmana-359165.html / Доступ до ресурсу: 09.12.2025.
3. Ден Гасфілд. Integer Linear Programming in Computational and Systems Biology: An Entry-Level Text and Course / Ден Гасфілд, 2019 – с. 428 (Лінійне програмування).
4. Структури даних та алгоритми [Електронний ресурс] : https://studfile.net/preview/10025806/page:21/. Доступ до ресурсу: 09.12.2025.

Додаток Б – Сторінка статистики користувача