МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Чорноморський національний університет імені Петра Могили

Факультет комп'ютерних наук

Кафедра інженерії програмного забезпечення

ДОПУЩЕНО ДО ЗАХИСТУ

Завідувач кафедри _____ Є. О. Давиденко

підпис

«__» лютого 2024 р.

КВАЛІФІКАЦІЙНА РОБОТА МАГІСТРА

Програмне забезпечення на базі фреймворку Django для формування навичок розв'язання математичних задач школярами

Спеціальність «Інженерія програмного забезпечення»

121–КРМ.1–608м.21810915

Студент _____ О. П. Кисса

підпис

«__» лютого 2024 р.

Керівник д-р техн. наук, професор _____ І. М. Журавська

підпис

«__» лютого 2024 р.

Миколаїв – 2024

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Чорноморський національний університет імені Петра Могили

Факультет комп'ютерних наук

Кафедра інженерії програмного забезпечення

ЗАТВЕРДЖУЮ

Завідувач кафедри _____ Є. О. Давиденко

підпис

«___» _____2023 р.

ЗАВДАННЯ

на виконання кваліфікаційної роботи магістра

Видано студенту групи 608м факультету комп'ютерних наук

Кисса Олег Петрович.

(прізвище, ім'я, по батькові студента)

1. Тема кваліфікаційної роботи

Програмне забезпечення на базі фреймворку Django для формування навичок розв'язання математичних задач школярами

Затверджена наказом по ЧНУ ім. Петра Могили від 11 листопада 2023 р. № 234.

2. Строк представлення кваліфікаційної роботи 27.02.2024.

3. Очікуваний результат роботи та початкові дані, якщо такі потрібні

Початкові дані: предметно-орієнтований контент, функціональні вимоги до застосунку для розв'язання математичних завдань.

4. Перелік питань, що підлягають розробці:

- визначення ідеї застосунку;
- моделювання, проєктування програмного застосунку;
- створення інтерфейсу користувача;
- кодування та тестування застосунку.

5. Перелік графічних матеріалів

презентація

Керівник роботи д-р техн. наук, проф. Журавська Ірина Миколаївна

(посада, прізвище, ім'я, по батькові)

_____

(підпис)

Завдання прийнято до виконання Кисса Олегом Петровичем

(прізвище, ім'я, по батькові студента)

_____

(підпис)

Дата видачі завдання «____» _____ 20_____ р.

КАЛЕНДАРНИЙ ПЛАН

виконання дипломної роботи

Тема:_ «Програмне забезпечення на базі фреймворку Django для формування навичок розв'язання математичних задач школярами»

№	Найменування роботи	Початок	Закінчення	Примітки
1.	Розробка та затвердження завдання на виконання КРБ	10.11.2023	10.11.2023	виконано
2.	Огляд літератури за темою роботи	17.09.2023	19.09.2023	виконано
3.	Складання календарного плану КРБ	20.09.2023	21.10.2023	виконано
4.	Аналіз предметної області	21.10.2023	07.11.2023	виконано
5.	Розробка архітектури	08.11.2023	25.11.2023	виконано
6.	Створення користувацького інтерфейсу.	26.11.2023	02.12.2023	виконано
7.	Розробка застосунку	03.12.2023	13.01.2024	виконано
8.	Розробка спеціальної частини з охорони праці	14.01.2024	29.01.2024	виконано
9.	Відгук керівника КРМ	30.01.2024	31.01.2024	виконано
10.	Оформлення КРМ та презентації	01.02.2024	12.02.2024	виконано
11.	Попередній захист	13.02.2024	13.02.2024	виконано
12.	Рецензування	20.02.2024	21.02.2024	
13.	Завершення оформлення КРМ та презентації	16.02.2024	17.02.2024	
14.	Захист кваліфікаційної роботи			

Розробив студент Кисса Олег Петрович
(прізвище, ім'я, по батькові студента) (*підпис*)

«____» _____ 2023 р.

Керівник роботи д-р техн. наук, професор, Журавська Ірина Миколаївна
(*ступень, звання, прізвище, ім'я, по батькові)* *(підпис)*

«____» _____ 2023 р.

АНОТАЦІЯ

до кваліфікаційної роботи магістра

«Програмне забезпечення на базі фреймворку Django для формування навичок розв'язання математичних задач школярами»

студента гр. 608м: Кисса Олег Петрович

Керівник: д-р техн. наук, професор Журавська Ірина Миколаївна

Об’єктом кваліфікаційної роботи – процес розробки програмного забезпечення (ПЗ) на базі фреймворку Django, який надає можливість школярам тренувати свої математичні навички.

Предметом кваліфікаційної роботи є технології створення застосунку з використанням фреймворку Django.

Метою кваліфікаційної роботи є покращення формування навичок розв'язання математичних задач школярами шляхом використання спеціалізованого програмного забезпечення.

У першому розділі представлені основні поняття предметної сфери огляд методології розробки програмного забезпечення, також порівняння та аналіз онлайн-тренажерів з математики. У другому розділі розглянуто математичні методи аналізу статистичних даних, отриманих з внутрішньої аналітики сайту, на прикладі математичних тренажерів. У третьому розділі – створення ідеї застосунку та підготовка до розробки онлайн-тренажеру з математики. У четвертому розділі представлено опис процесу розробки тренажеру з математики.

В результаті виконаної роботи було зроблено висновки щодо створення та експлуатації програмного застосунку.

Сторінок – ___, таблиць – ___, рисунків – ___, посилань – ___, додатків – ___.

Ключові слова: *математичні задачі, школярі, онлайн-тренажер, вебінтерфейс, Django.*

ABSTRACT

of the Master's Thesis

« Software based on the Django framework for developing pupils' skills in solving mathematical tasks»

Student of gr. 608m: Kyssa Oleh Petrovych

Supervisor: Doctor of Technical Sciences, Professor, Zhuravska Iryna Mykolaivna

The object of the qualification work is the process of developing software based on the Django framework, which allows pupils to train their mathematical skills.

The subject of the qualification work is the technology of creating an application using the Django framework.

The purpose of the qualification work is to improve the development of pupils' skills in solving mathematical tasks through the use of specialized software.

The first section introduces the basic concepts of the subject area, provides an overview of the software development methodology, and compares and analyses online mathematics simulators. The second section discusses mathematical methods of analysing statistical data obtained from internal website analytics, using the example of maths simulators. Section 3 describes the creation of an application idea and preparation for the development of an online maths simulator. The fourth section describes the process of developing a maths simulator.

As a result of the work performed, conclusions were drawn regarding the creation and operation of the software application.

Pages –____, tables – ____ , figures – ____ , references – ____ , appendices – _____.

Keywords: *math tasks, pupils, online simulator, web interface, Django.*

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

ДПА – Державна підсумкова атестація

ДСЯО – Державна служба якості освіти

ЗНО – Зовнішнє незалежне оцінювання

ІС – інформаційна система

ІТ – інформаційна технологія

НМТ – Національний мультипредметний тест

ПЗ – програмне забезпечення

API – Application Programming Interface

CRM – Customer Relationship Management

CVS – Comma-separated values

CSS – Cascading Style Sheets

IDE – Integrated Development Environment

SMA – Simple Moving Average

SMS – Short Message Service

SVN – Subversion Version Control System

ВСТУП

Пандемія COVID-19 принесла до нашого світу великі зміни, особливо це торкнулося сфери освіти. Школам терміново довелося переходити на дистанційний формат навчання. Старі методи навчання погано працювали при проведенні дистанційних занять, і багато вчителів не змогли адаптуватися до нової цифрової реальності та перейшли до сектору додаткової освіти. Через зменшення кількості вчителів, часу, що приділяється кожному учневі, поменшало, що вдарило в першу чергу за практичними навичками учнів, що позначилося на результатах ЗНО та НМТ в останні роки.

Повертаючись до проблеми учнів, хотілося відзначити, що вони часто роблять помилки в елементарних математичних операціях, навичка вирішення яких має сформуватися на практичних заняттях у школі. Ця проблема була і до пандемії, але перехід на дистанційне навчання лише посилив її. Для формування навички необхідно багаторазово повторювати певні дії. Наприклад, щоб сформувати математичну навичку, вчителю необхідно придумати або знайти відповідне завдання, яке учень вирішить, а потім вчителю необхідно перевірити це завдання та оцінити наскільки учень освоїв навичку, що відпрацьовується. Якщо учень не зміг виконати завдання коректно, то доведеться повторити ті самі дії. Дана робота складається з рутинних операцій, які можна автоматизувати, створивши інформаційну систему (ІС), яка сама генеруватиме завдання та перевірятиме правильність їх рішень.

Об’єкт кваліфікаційної роботи – процес розробки програмного забезпечення (ПЗ) на базі фреймворку Django, який надає можливість школярам тренувати свої математичні навички.

Предметом кваліфікаційної роботи є технології створення застосунку з використанням фреймворку Django.

Метою кваліфікаційної роботи є покращення формування навичок розв'язання математичних задач школярами шляхом використання спеціалізованого програмного забезпечення.

Для досягнення мети поставлено такі завдання:

– проаналізувати існуючі на вітчизняному ринку освітніх послуг онлайн-тренажери;

– спроєктувати математичний застосунок, виконати всі етапи розробки з використанням фреймворку Django;

– розробити вебінтерфейс онлайн-тренажера;

– виконати тестування розробленого програмного забезпечення.

Практична значимість дослідження полягає у створенні ІС онлайн-тренажера, який автоматизує процес відпрацювання математичних навичок у школярів. Такий тренажер можуть використовувати батьки школярів для відстеження прогресу навчання, а також індивідуальні репетитори або школи для автоматизації видачі домашніх та практичних завдань. Інформаційно-емпіричною базою цього дослідження стали загальноприйняті методології розробки ІС, алгоритмів та архітектур ІС, а також результати дослідження діяльності існуючих ІТ-компаній у сфері розробки освітніх ІТ-рішень.

Апробація результатів КРМ. Результати роботи було представлено на Всеукраїнської науково-практичної конференції молодих вчених, аспірантів і студентів «Інформаційні технології та інженерія» (31 січня – 2 лютого 2024 р., ЧНУ ім. Петра Могили) [1].

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ. СПЕЦИФІКАЦІЯ ВИМОГ

1.1 Аналіз кількості учнів в Україні

Згідно зі статистикою ЗНО 2021 року, опублікованою Державною службою якості освіти, кількість осіб, які брали участь у складанні іспиту, необхідного для вступу на технічні спеціальності:

а) ЗНО з профільної математики – 365 тис. осіб;

б) ЗНО з фізики – 128 тис. осіб;

в) ЗНО з інформатики – 94 тис. осіб;

г) ЗНО з хімії –93 тис. осіб.

За даними Державної служби якості освіти України (ДСЯО) за 2021 рік вікова структура населення України розподілена таким чином (рис. 1.1):

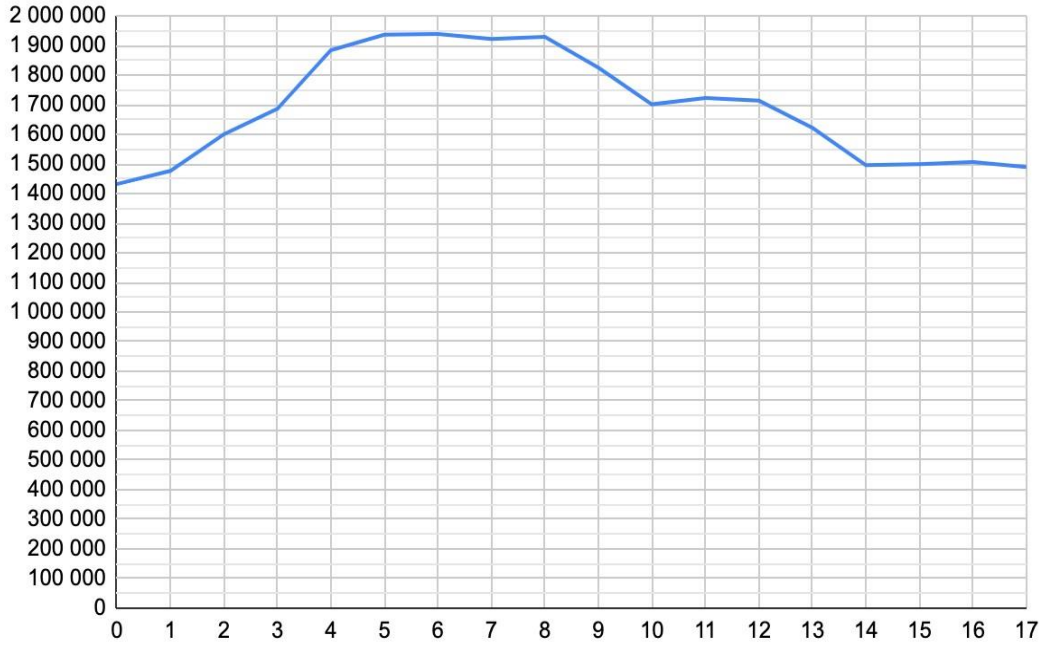

Рисунок 1.1 – Графік залежності кількості населення від віку

У 2023 році ЗНО/НМТ складають школярі, народжені переважно 2005 року, а ДПА – школярі, народжені 2007 року. Як видно з рис. 1.1, кількість населення, віком від 17 років до 8 років зростає на 21 %. Отже, зростатиме і

кількість тих, хто складатиме ДПА та ЗНО/НМТ, а також ринок освітніх послуг, зокрема й ринок, що надає послуги у сфері математичної освіти.

1.2 Соціологічне опитування батьків учнів щодо зацікавленості онлайн тренажером з математики

Для вивчення потреб в онлайн-тренажері з математики та готовності платити за нього було проведено аналіз опитування 53 батьків дітей шкільного віку.

До опитування входили такі запитання:

1) Ваш рівень доходу?

2) Чи займаються Ваші діти додатково?

3) За яких умов ви б найняли для своєї дитини репетитора з математики?

4) Яка причина категоричної відмови від репетиторства?

5) Зручно було б Вам, якби Ваша дитина займалася математикою додатково не з репетитором, а на онлайн-платформі?

6) Яку суму на місяць Ви готові платити за такий сервіс?

За рівнем доходу батьки були розділені на 3 групи:

1) група з низьким доходом – до 14 тисяч гривень на місяць (16 осіб);

2) група із середнім доходом – від 14 до 20 тисяч гривень на місяць (17 осіб);

3) група з високим доходом – понад 20 тисяч гривень на місяць (20 осіб).

Результати опитування щодо зацікавленості батьків в онлайн тренажері з математики представлено на рис. 1.2. Зеленим позначено відсоток зацікавлених в онлайн тренажері батьків, а червоним – незацікавлених.

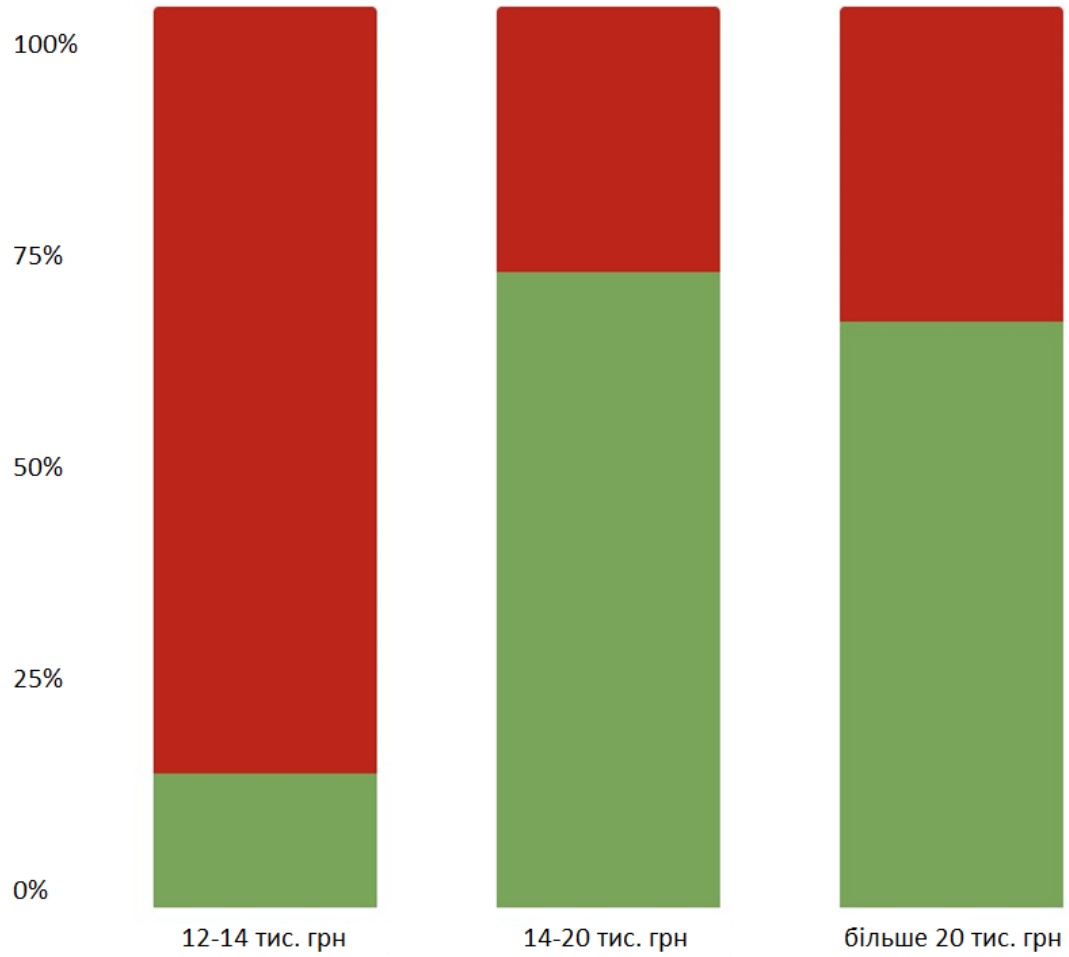

Рисунок 1.2 – Гістограма результатів опитування батьків

За даними опитування група з низьким доходом не може дозволити своїм дітям займатися з репетитором. У цієї групи також низька зацікавленість в онлайн-тренажері (менше 20 %). Група із середнім доходом у середньому витрачають на репетитора 6000 грн/міс., і для скорочення витрат на репетиторів готові перейти на використання онлайн-сервісів. Група з високим доходом також показали високу зацікавленість в онлайн-навчанні, але вже з метою додаткового підвищення ефективності навчання своєї дитини [2].

1.3 Аналіз онлайн-тренажерів з математики

1.3.1 Аналіз функціоналу існуючих сервісів

Як основних конкурентів були відібрані тільки тренажери на базі вебінтерфейсів українською мовою, такі як Learning.ua, OnlineMSchool, «На Урок» та Matific [8].

Онлайн сервіс Learning.ua – це освітня платформа, де процес навчання відбувається за сценарієм дітей. На сайті можна пройти інтерактивну онлайн програму з математики для малюків, дошкільнят, учнів 1–11 класів загальноосвітніх навчальних закладів з поглибленим вивченням [6].

Навряд чи сайт підійде для глибокого вивчення математики, але точно стане в пригоді як допоміжний інструмент. Інтерактивні математичні ігри наочно демонструють як влаштована ця наука. Методологія навчання на Learning.ua дещо відходить від стандартних освітніх принципів. Замість зубріння учням тут пропонують пограти.

Інтерфейс тренажера Learning.ua складається з семи основних елементів (рис. 1.3):

- назва завдання;
- загальний час виконання тренажера;
- запитання у форматі тексту або картинки;
- кількість зароблених очок під час проходження тесту;
- кнопка «Паузи»;
- кнопка «Вимкнути звук»;
- кнопка «Поділитися».

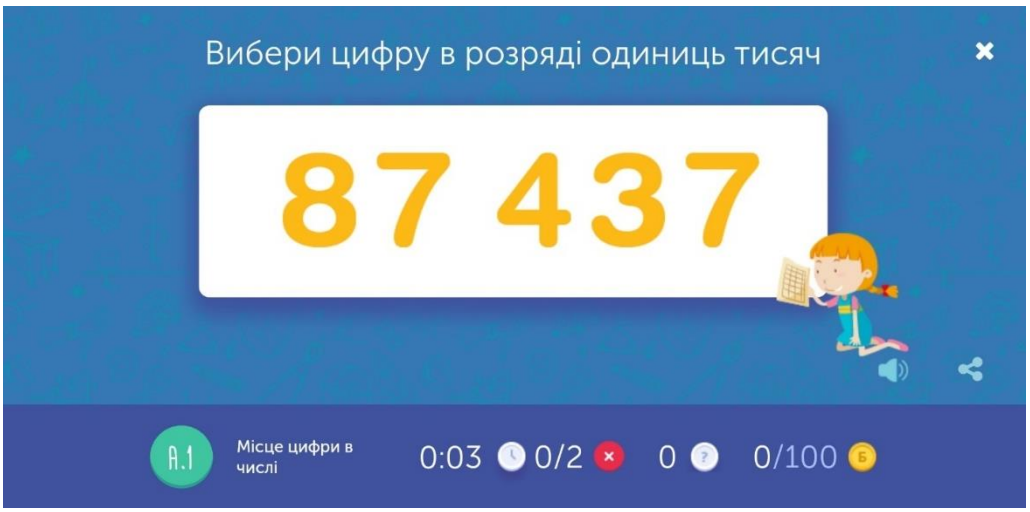

Рисунок 1.3 – Інтерфейс тренажера **Learning.ua**

Онлайн сервіс OnlineMSchool створений для допомоги школярам та студентам в розв’язанні математичних задач та вивченні математики. Тут зібрані задачі, розв’язуючи які, ви зможете вдосконалювати свої математичні навички. Усі вони різного рівня складності створені лише для самостійного опрацювання. Автори проєкту стверджують, що розв’язання задач – це найкращий спосіб підготуватися до контрольних, екзаменів або тестів з математики [7].

Також на сайті є великий довідник з теоретичними матеріалами. Частина з них зібрана в зручні таблиці та формули. Ними можна користуватися під час розв’язання задач. Також є декілька видів вбудованих калькуляторів, за допомогою яких можна перевіряти правильність своїх відповідей.

Єдиний недолік сайту – це його інтерфейс. Він застарілий, незручний і тому дещо відлякує учнів. Однак з часом до нього можна звикнути.

Інтерфейс тренажера OnlineMSchool складається з шести елементів (рис. 1.4):

– вкладки «Вправа» та «Інструкція»;
– назва завдання;
– запитання у форматі числового виразу;
– зелена лінія – індикатор проходження тесту;

– область вводу відповіді;
– кнопка «Почати завдання».

Рисунок 1.4 – Інтерфейс тренажера OnlineMSchool

Онлайн сервіс «На Урок» – українська цифрова освітня екосистема для роботи та професійного зростання освітян України. Ми переконані, що неформальна освіта здатна створювати унікальні можливості для професійного розвитку кожного педагога, оскільки дозволяє навчатися за будь-яких умов – незалежно від часу та технічного оснащення. Керуючись засадами вільного доступу до навчання, наша команда робить все, аби вчителі України вільно отримували якісні матеріали та інструменти для щоденної роботи [9].

«На Урок» є суб’єктом підвищення кваліфікації відповідно до чинного законодавства, тому наша діяльність спрямована на створення комфортних умов для розвитку професійних навичок учасників освітнього процесу.

У сервісі «На Урок» педагоги можуть використовувати інноваційні ресурси світового рівня, адаптовані до українських реалій з використанням інтерактивних інструментів, які допомагають в організації ефективної роботи дистанційно і в класі. Чат онлайн-тренажера «На Урок» – перший українськомовний освітній продукт на базі штучного інтелекту ChatGPT.

В зазначеній системі онлайн-тестів накопичено базу матеріалів із понад 2 млн зразків авторських тестів, відповідних шкільній програмі, організовано

середовище вчительського партнерства задля обміну досвідом у Бібліотеці навчальних матеріалів. Також створено низку методичних матеріалів – статті різної тематики, шаблони навчальних проєктів тощо.

Освітній проєкт «На Урок» має статус ЗМІ від 2019 року, тому на ньому відслідковуються освітні події, моніторяться зміни освітнього законодавства й оперативно аналізується та публікується інформація відповідно до журналістських стандартів.

Інтерфейс тренажера складається з чотирьох елементів (рис. 1.5):

– запитання в форматі плиток;
– номер завдання;
– завдання в білому полі;
– відповіді на запитання в кольорових плитках.

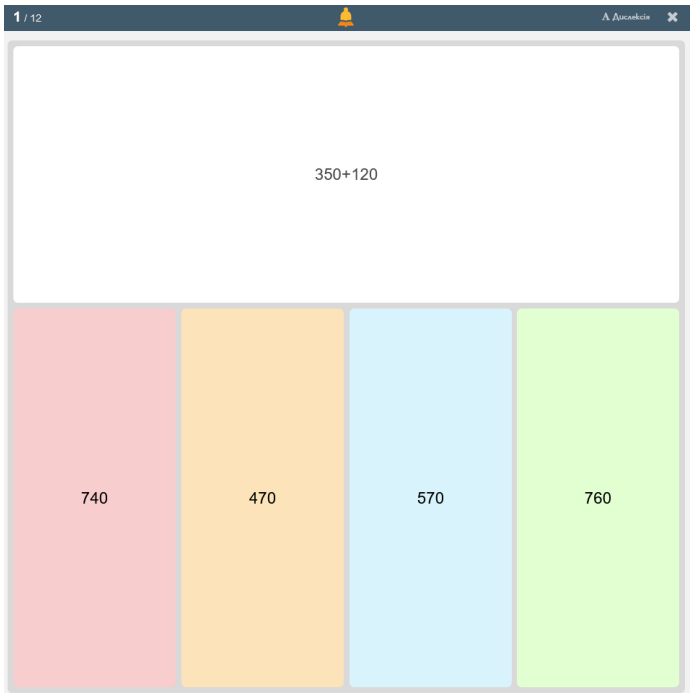

Рисунок 1.5 – Інтерфейс тренажера Mathsimple

Онлайн-сервіс Matific – це платформа для вивчення математики від освітніх експертів. Вона створена для взаємодії педагогів та дітей. Вчителі просто заохочують своїх учнів використовувати Matific протягом 30 хвилин на тиждень. Робота на платформі покаже, що діти знають добре, а які напрямки

варто вдосконалити. Вчителям також надходитимуть щотижневі оновлення, щоб інформувати їх про успіхи учнів [8].

Якщо вчителі хочуть охопити певний зміст, вони можуть просто призначити учням завдання на необхідні теми, вони будуть інтегровані в робочий процес. Вчитель може планувати заздалегідь і складати графік роботи на місяці вперед. За інформацією засновників сайту використання Matific у класі підвищує успішність учнів на 34 %. Хоча проєкт є міжнародним, на сайті доступна українська мова.

Інтерфейс тренажера Matific складається з чотирьох елементів (рис. 1.6):

– запитання у форматі числового виразу;
– область введення відповідічас виконання;
– кнопка «Корзина» очистити відповідь;
– кнопка «Готово».

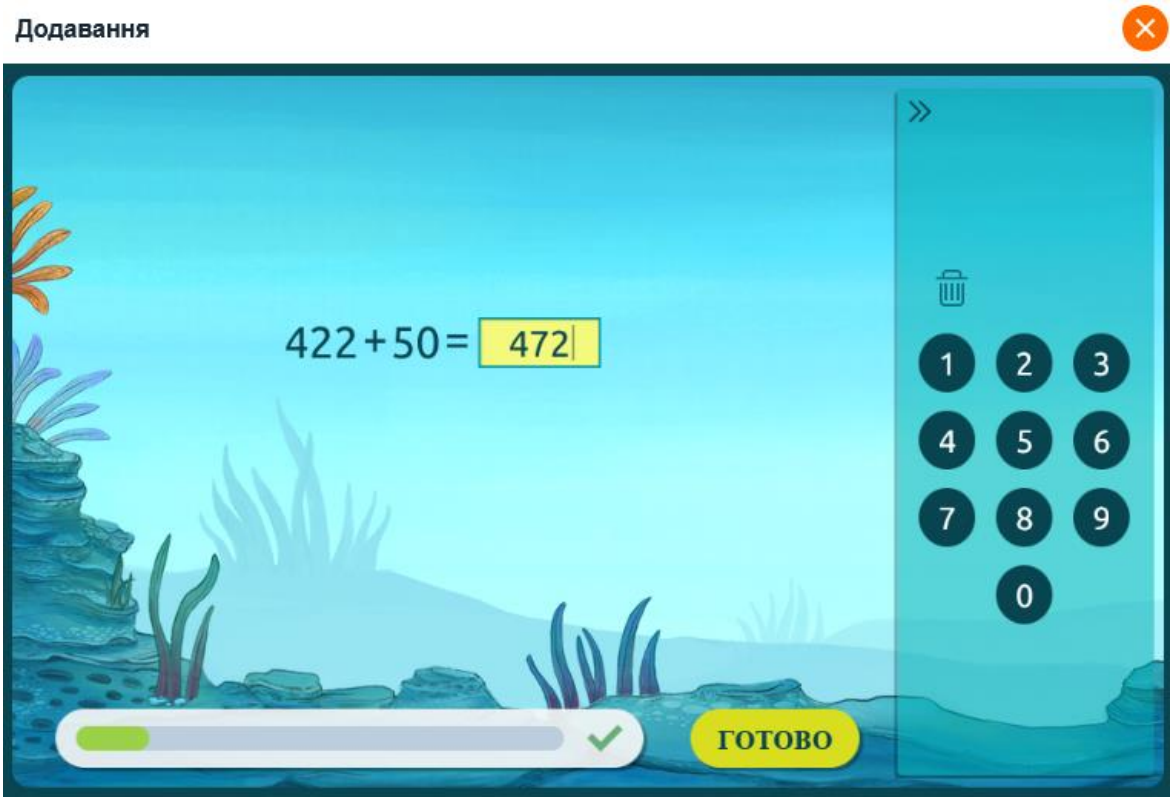

Рисунок 1.6 – Інтерфейс тренажера Matific

1.3.2 Дослідження юзабіліті існуючих сервісів

Learning.ua – це освітня онлайн-платформа, яка пропонує широкий вибір навчальних матеріалів для дітей і дорослих. Платформа пропонує уроки з різних предметів, включно з математикою, мовами, природничими науками, гуманітарними науками тощо.

Недоліки сайту Learning.ua:

– **організація контенту.** Контент на сайті learning.ua організований не дуже зручно. Пошук потрібного матеріалу може бути ускладнений через те, що матеріали розділені на кілька категорій, які не завжди чітко відповідають одна одній.

– **інтерфейс.** Інтерфейс сайту learning.ua трохи застарілий і не дуже зручний для використання. Він не такий інтуїтивно зрозумілий, як міг би бути;

– **супровід учнів.** Сайт learning.ua не пропонує достатнього супроводу учнів. Учні не можуть отримати допомогу від викладачів або інших учнів у разі виникнення запитань або труднощів.

Mathsimple – це освітня онлайн-платформа, яка пропонує відеоуроки з математики для учнів початкової та середньої школи. Платформа пропонує уроки з різних тем математики, включаючи арифметику, алгебру, геометрію та тригонометрію.

Недоліки сайту Mathsiple:

– **якість відео**. Деякі відео на сайті mathsimple мають низьку якість зображення або звуку. Це може ускладнити учням розуміння матеріалу;

– **пояснення**. Деякі відео на сайті mathsimple не містять достатніх пояснень. Це може ускладнити учням розуміння складних математичних понять;

– **задачі**. На сайті mathsimple недостатньо завдань для практики. Це може ускладнити учням закріплення матеріалу.

«На Урок» – це освітня онлайн-платформа, яка пропонує широкий спектр навчальних матеріалів для учнів початкової та середньої школи.

Платформа пропонує уроки з різних предметів, включаючи математику, мови, природні науки та гуманітарні науки.

Недоліки сайту «На урок»:

– **якість матеріалів**. Деякі матеріали на сайті «На Урок» не мають достатньої якості. Наприклад, деякі уроки мають низьку якість відео або аудіо. Це може ускладнити учням розуміння матеріалу;

– **актуальність матеріалів**. Деякі матеріали на сайті «На Урок» не є актуальними. Наприклад, деякі уроки використовують застарілу термінологію або методи навчання. Це може ускладнити учням засвоєння матеріалу;

– **відсутність індивідуального підходу**. Сайт «На Урок» пропонує загальні матеріали, які не враховують індивідуальні потреби учнів. Це може ускладнити учням, які мають проблеми з розумінням матеріалу, отримати необхідну допомогу.

OnlineMSchool пропонує широкий спектр програм з різних галузей, включаючи бізнес, медицину, освіту та інформатику. Школа пропонує як дипломні, так і сертифікаційні програми.

Недоліки сайту OnlineMSchool:

– **вартість**. Вартість навчання в OnlineMSchool може бути високою, особливо якщо ви хочете отримати диплом. Наприклад, вартість дипломної програми в галузі бізнесу становить 10’000 доларів США за рік;

– **якість навчання**. Якість навчання в OnlineMSchool може бути різною залежно від програми. Деякі програми пропонують якісний контент і підтримку викладачів, тоді як інші програми можуть бути менш якісними;

– **самонавчання**. Онлайн-навчання вимагає значної самодисципліни та самомотивації. Студентам потрібно бути готовими самостійно навчатися і виконувати завдання;

– **соціальний аспект**. Онлайн-навчання може бути ізольованим, оскільки студенти не будуть взаємодіяти з іншими студентами так само, як у традиційному класі.

1.4 Порівняння сервісів

Як видно з табл. 1.1, сервіси Matific, Learning.ua, «На Урок» не мають жодної бізнес-моделі, тобто вони розповсюджуються повністю безкоштовно. Сервіси OnlineMSchool, Learning.ua працюють за передплатою. У всіх платних сервісів присутній період безкоштовного використання після реєстрації.

Таблиця 1.1 – Порівняльна таблиця онлайн-сервісів

Тренажер	**Вартість за 1 рік, грн**	**Бізнес-модель**	**Кількість навичок з математики, шт.**
Learning.ua	900	за передплатою	241
«На Урок»	0	–	57
OnlineMSchool	1455	–	147
Matific	0	необмежена ліцензія	154

Під час тестування всіх тренажерів, було зібрано набір усіх функціональних особливостей, що зустрічаються:

1) кількість навичок з математики (без спеціальних тренажерів для підготовки до ДПА, ЗНО/НМТ);

2) загальне вхідне тестування;

3) вхідне тестування за класами;

4) вхідне тестування за кожною навичкою окремо;

5) групування навичок за класами навчання;

6) групування навичок за групами;

7) при відповіді показує правильну відповідь;

8) тест, обмежений за кількістю запитань;

9) тест, обмежений за часом;

10) відпрацювання неправильної відповіді;

11) розклад проходження навичок;

12) робота з репетитором;

13) Рейтинг учнів;

14) Реферальна програма;

15) Складові навички;

16) Спеціальні навички для підготовки до ДПА, ЗНО/НМТ;

17) Статистика за навичками;

18) Відеоуроки;

19) Пробний період або демоверсія;

20) Пояснення до завдань;

21) Особистий кабінет;

22) Реєстрація та авторизація;

23) Швидка реєстрація та авторизація через соціальні мережі;

24) Повідомлення на електронну пошту;

25) Повідомлення на телефон;

26) Повідомлення в соціальні мережі або в месенджери;

27) Мобільний застосунок;

28) Ігровий формат завдань;

29) Кабінет батька;

30) Індивідуальна траєкторія;

31) Досягнення;

32) Кабінет учителя;

33) Об'єднання учнів у класи;

34) Багатомовність;

35) Наголос тільки на математиці.

Після виявлення всіх функціональних особливостей кожен тренажер ще раз тестувався протягом тижня на пошук цих самих особливостей. Результати тестування тренажерів зібрано в табл. 1.2.

Таблиця 1.2 – Порівняльна таблиця функціональності тренажерів

Показники	**Learning.ua**	**На Урок**	**OnlineMSchool**	**Matific**
Вартість за 1 рік, грн	900	0	1455	0
Кількість навичок з математики, шт.	241	57	147	154
Загальне вхідне тестування	0	0	0	0
Вхідне тестування за класами	1	0	0	0
Вхідне тестування за кожною навичкою	1	0	0	0
Групування навичок за класами навчання	1	0	0	1
Групування навичок за групами	1	1	1	1
При відповіді показує правильну відповідь	1	0	0	0
Тест обмежені за кількістю запитань	1	0	0	0
Тест, обмежений за часом	0	1	0	0
Відпрацювання неправильної відповіді	1	0	0	0
Розклад проходження навичок	1	0	0	0
Робота з репетитором	1	0	0	1
Рейтинг учнів	1	0	0	0
Складові навички	0	0	0	0
Спеціальні навички для підготовки до іспитів	1	1	0	0
Статистика за навичками	1	0	0	1
Відеоуроки з математики	0	0	0	0
Пробний період/демо	1	0	1	1
Пояснення	0	1	1	1
Особистий кабінет	1	1	0	1
Реєстрація/авторизація	1	1	1	1
Швидка реєстрація/авторизація через соціальні мережі	1	0	0	1
Повідомлення на електронну пошту	1	1	0	1
Повідомлення на телефон	0	0	0	0
Повідомлення в месенджери	0	0	0	0

Із зібраних даних (див. табл. 1.2) з використанням засобів аналізу даних мови програмування Python було отримано коефіцієнти кореляції з вартістю тренажерів (рис. 1.7).

Рисунок 1.7 – Коефіцієнти кореляції функціоналу з вартістю

Найбільші коефіцієнти кореляції з вартістю сервісу спостерігаються у таких функцій:

– батьківський кабінет;
– вчительський кабінет;
– групування учнів за класами;
– кількість математичних навичок;
– індивідуальна траєкторія;
– групування навичок за групами.

Негативні коефіцієнти кореляції тільки в багатомовності та в тестах, обмежених за часом, тому що ці функції присутні тільки в безкоштовних сервісах.

1.5 Огляд методологій розроблення програмного забезпечення

Методологія розроблення ПЗ – це сукупність методів, які застосовують на різних стадіях життєвого циклу програмного забезпечення, що мають загальний філософський підхід [10].

На сьогодні виділяють дві основні методології розроблення ПЗ:

1) каскадна або водоспадна методологія;

2) гнучка методологія або Agile-методологія.

1.5.1 Каскадна модель розроблення ПЗ

За каскадної моделі процес розроблення має вигляд потоку дій, що послідовно проходять одна за одною (рис. 1.8).

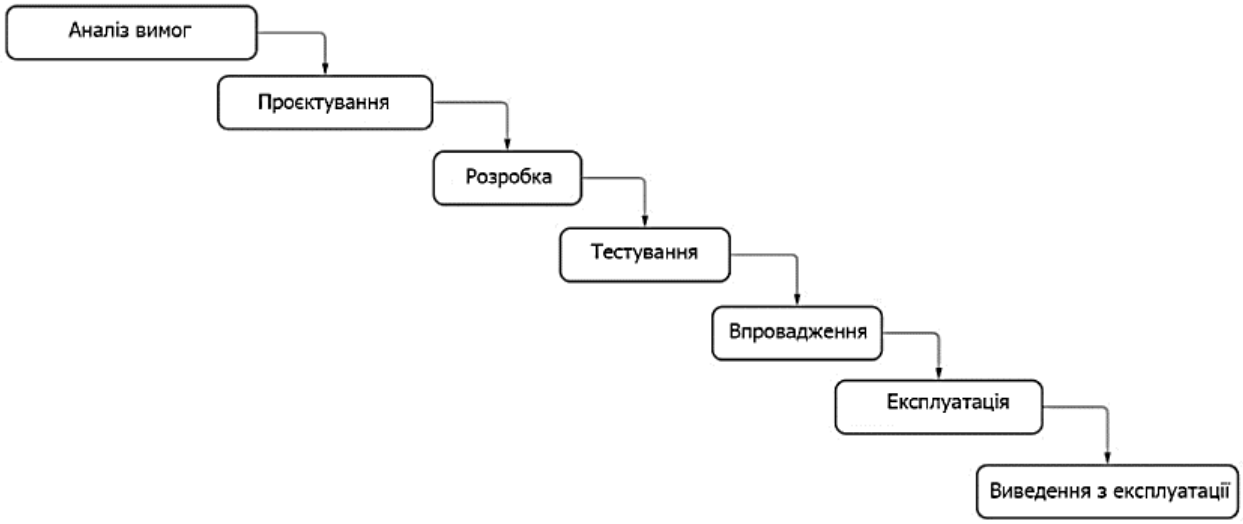

Рисунок 1.8 – Візуальне представлення каскадної моделі

За каскадної методології існують строго визначені терміни виконання роботи, тому ніколи не передбачається повернення на вже виконані етапи. Для того щоб додати якийсь новий функціонал до вже розроблюваного застосунку або щось виправити в ньому, необхідно завжди довести поточну роботу до кінця, а потім починати з самого початку і проходити всі етапи заново. Ціна помилки під час використання цієї методології досить велика.

Каскадна методологія добре підходить для проєктів із добре визначеними вимогами до розроблюваної інформаційної системи, а також коли задано суворі строки виконання робіт і обмежені бюджети.

Зазвичай виділяють 7 основних етапів:

1) аналіз вимог;

2) проєктування;

3) розробка;

4) тестування;

5) впровадження;

6) експлуатація;

7) виведення з експлуатації.

Ця методологія погано підходить для проєктів із неявними вимогами, які постійно змінюються. У цьому питанні на допомогу приходять гнучкі методології.

1.5.2 Гнучкі моделі розроблення ПЗ

Гнучкі методології розроблення – це набір підходів до розроблення програмного забезпечення, які орієнтовані на використання ітеративного розроблення (рис. 1.9), динамічне формування вимог і забезпечення їхньої реалізації внаслідок постійної взаємодії всередині робочих груп, які самоорганізуються та складаються з фахівців різного профілю.

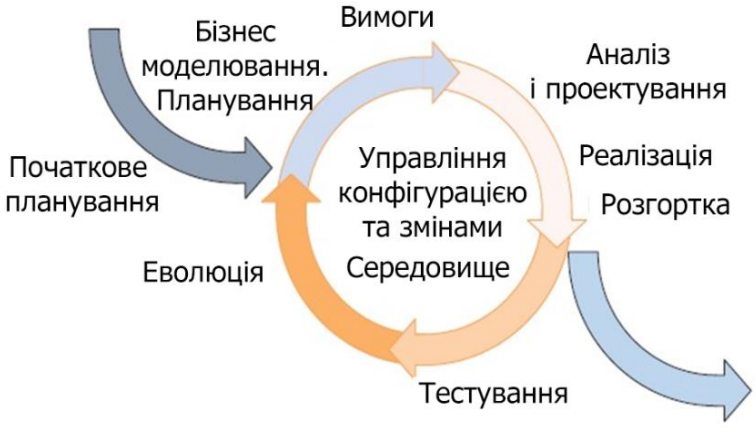

Рисунок 1.9 – Схема ітеративного підходу до розробки ПЗ

В основі Agile-методологій лежить маніфест, що визначає цінності:

1) люди і взаємодія важливіші за процеси та інструменти;
2) працюючий продукт важливіший за вичерпну документацію;
3) співпраця із замовником важливіша за узгодження умов контракту;
4) готовність до змін важливіша за дотримання плану.

Також існує 11 принципів:

1) найвищим пріоритетом для нас є задоволення потреб замовника, завдяки регулярному та ранньому постачанню цінного програмного забезпечення;

2) зміна вимог вітається, навіть на пізніх стадіях розробки. agile-процеси дають змогу використовувати зміни для забезпечення замовнику конкурентної переваги;

3) працюючий продукт слід випускати якомога частіше, з періодичністю від кількох тижнів до кількох місяців.

4) протягом усього проєкту розробники та представники бізнесу мають щодня працювати разом;

5) над проєктом мають працювати мотивовані професіонали. щоб робота була зроблена, створіть умови, забезпечте підтримку і повністю довіртеся їм;

6) безпосереднє спілкування є найпрактичнішим та найефективнішим способом обміну інформацією як із самою командою, так і всередині команди.

7) працюючий продукт - основний показник прогресу;

8) інвестори, розробники та користувачі повинні мати змогу підтримувати сталий ритм нескінченно. Agile допомагає налагодити такий стійкий процес розробки;

9) простота - мистецтво мінімізації зайвої роботи вкрай необхідна.

10) найкращі вимоги, архітектурні та технічні рішення народжуються в команд, що самоорганізуються;

11) команда повинна систематично аналізувати можливі способи поліпшення ефективності та відповідно коригувати стиль своєї роботи.

Висновки до розділу 1

У цьому розділі було розглянуто статистику Державною службою якості освіти (ДСЯО) за попередній рік щодо кількості тих, хто здає ЗНО з предметів, необхідних для вступу на технічні спеціальності, а також демографічне становище в України за даними ДСЯО.

Також було проведено аналіз зацікавленості батьків учнів в онлайн тренажері з математики, який показав зацікавленість понад 70 % у сімей з доходом від 40 тис. гривень на людину на місяць.

Фінальною частиною аналізу ринку освітніх онлайн-послуг було порівняння наявних рішень, яке показало:

1) які функціональні особливості існують на даний момент;

2) яка кореляція у функціональних особливостей із вартістю.

У другій частині цього розділу було розглянуто основні сучасні методології розроблення програмного забезпечення (ПЗ), які будуть використовуватися в розробленні спеціалізованого ПЗ для формування навичок розв'язання математичних задач школярами.

2 АНАЛІЗ ДАНИХ ТА ВИЯВЛЕННЯ ПРИЧИННО-НАСЛІДКОВИХ ЗВ'ЯЗКІВ

2.1 Методи математичної обробки даних

Багато досліджень мають на меті точно відобразити об'єктивні процеси, виявляючи суттєві взаємозв'язки та оцінюючи їх кількісно. Для цього необхідно виявити причинні залежності, тобто зв'язок між процесами, при якому зміна одного є результатом зміни іншого.

Одним з центральних питань аналітичної практики є дослідження змін показників з часом та вивчення динаміки розвитку різних процесів. Методи аналізу та прогнозування динаміки одновимірних часових рядів є важливою частиною аналітичних досліджень у різних сферах.

Причинно-наслідковий зв'язок – це зв'язок, де зміна одного фактора (причини) веде до зміни іншого фактора (наслідку). Виявлення таких зв'язків є ключовим завданням аналітичної роботи.

Динаміка часових рядів – це зміна показників у часі. Аналіз динаміки дозволяє досліджувати, як розвиваються різні процеси, і робити прогнози на майбутнє [11].

Ковзне середнє – це один з найпопулярніших індикаторів в технічному аналізі. Він являє собою криву, де кожна точка показує середнє значення цін за певний період. Ковзне середнє є популярним і універсальним індикатором у технічному аналізі, який показує середнє значення цін за певний період часу. Основним параметром індикатора є «період», який визначає кількість періодів, за який буде проведено розрахунок. Чим довший період, тим більше значень буде враховано в розрахунку і тим гладкішим буде зображений поточний тренд.

Універсальність ковзного середнього полягає в тому, що воно може використовуватися на будь-якому таймфреймі. Це робить його цінним інструментом для аналізу як короткострокових, так і довгострокових трендів.

Ковзне середнє може використовуватися не лише в технічному аналізі, а й в інших сферах, наприклад, для аналізу даних про відвідування вебсайтів [12].

Переваги ковзного середнього:

- Простий у розрахунку;
- Легко інтерпретувати;
- Універсальний інструмент.

Недоліки ковзного середнього:

- Нечутливий до короткострокових коливань;
- може давати запізнення у сигналах.

Сімейство функцій ковзного середнього включає метод простого ковзного середнього та метод зваженого ковзного середнього:

- просте ковзне середнє: це найпростіший метод, де всі значення в розрахунковому періоді мають однакову вагу;
- зважене ковзне середнє: в цьому методі значенням в розрахунковому періоді присвоюються різні ваги.

Одним з інструментів аналізу даних є *просте ковзне середнє* (англ. Simple Moving Average, SMA). Це один з найпростіших і популярних індикаторів у технічному аналізі, який належить до групи індикаторів, що слідують за трендом. SMA допомагає визначити початок нової тенденції та її завершення. SMA допомагає згладити випадкові коливання даних і виявити основні тренди.

Просте ковзне середнє є методом, де прогнозований показник розраховується як середнє значення цього показника за кілька попередніх моментів часу. Цей метод є різновидом згладжування показників. Формула простого ковзного середнього виглядає наступним чином:

Просте ковзне середнє, яке визначається як середнє арифметичне значення, обчислюється за формулою (2.1):

Формула простого ковзного середнього виглядає наступним чином:

$$f_k = \frac{1}{N}\sum_{t=1}^{N} x_{t_{k-i}} \quad (2.1)$$

де f_k – прогноз характеристики на момент часу t_k;

N – число попередніх моментів часу, які використовуються при розрахунку;

$x_{t_{k-i}}$ – реальне значення показника в момент часу t_{k-i}.

Переваги SMA:

- простий у розрахунку;
- легко інтерпретувати;
- ефективний для виявлення трендів.

Недоліки SMA:

- нечутливий до короткострокових коливань;
- може давати запізнення у сигналах.

Використання SMA:

- визначення трендів;
- виявлення моментів розвороту тренду;
- фільтрація шуму.

Метод *зваженого ковзного середнього* є розширенням методу простого змінного середнього, яке враховує різний вплив даних за різні періоди часу на можливі зміни. Для цього вводиться поняття «ваги», які визначають ступінь впливу даних вхідного часового ряду на прогноз. Зазвичай ваги є частками, сума яких дорівнює одиниці. Формула для розрахунку зваженого ковзного середнього (2.2):

$$f_k = \frac{\sum_{i=1}^{N} w_{k-i} x_{k-i}}{\sum_{i=1}^{n} w_{k-i}}, \quad (2.2)$$

де w_{k-i} – вага, з яким показник x_{k-1} використовується в розрахунках.

Переваги зваженого ковзного середнього:

– гнучкість: можна налаштувати ваги таким чином, щоб дати більшу вагу більш важливим даним;

– чутливість: може краще реагувати на короткострокові коливання даних;

– точність: може давати більш точні прогнози, ніж просте ковзне середнє.

При зваженому ковзному середньому значення ряду замінюються на середні, розраховані за вікном з певними вагами. Ці ваги відображають внесок кожного члена ряду в закономірності досліджуваного процесу. Просте ковзне середнє, яке визначається як середнє арифметичне значення, обчислюється за формулою (2.3):

$$y_t = \frac{1}{m} \sum_{i=t-p}^{t+p} y_i \quad , \tag{2.3}$$

де y_i – фактичне значення i-го рівня;

m – число рівнів, що входять в інтервал згладжування – поточний рівень ряду динаміки, m –непарне число;

i – порядковий номер рівня в інтервалі згладжування;

p – при непарному m має значення $p = (m – 1) / 2$.

Інтервал згладжування – це кількість рівнів m, що визначаються за певними правилами. Якщо необхідно згладити невеликі, хаотичні коливання, то інтервал згладжування буде великим. Якщо потрібно зберегти більше невеликих коливань і позбутися лише випадкових великих сплесків, то інтервал згладжування зазвичай зменшують.

Таким чином, метод згладжування дозволяє визначити середні значення різних показників, таких як кількість користувачів сайту, середня кількість продажів, кількість кліків на товар (онлайн-послугу) тощо, протягом будь-якого періоду часу. Він може бути корисним для комерційних сайтів, які

мають сторінки з продукцією, що викликає інтерес в окремих користувачів або групи користувачів. Основною метою в даному випадку є сприяння користувачеві здійснити бажану дію (конверсію або мікроконверсію), наприклад, додати товар в корзину, підписатися на новинний канал або здійснити оплату замовлення.

Окрім статистичної аналітики, можна використовувати різні методи прогнозування, які використовують вже відомі значення певної величини для передбачення її майбутніх значень. До таких методів належать прості моделі, такі як ковзне середнє, експоненціальне згладжування, а також більш складні моделі, які передбачають майбутні значення величини на основі попередніх значень попиту. Ці методи можуть застосовуватися як для короткострокового прогнозування (з терміном до 3 місяців) так і для середньострокового (з терміном до кількох років). Важливо, щоб ці методи могли враховувати сезонні, циклічні та тренд-фактори, які можуть впливати на величину [13].

2.2 Основні статистичні показники для розрахунків

Часовий ряд – це сукупність значень, де кожне значення має своє часове значення. Часто це означає, що ми маємо множину спостережень, де кожне спостереження відбувається в певний момент часу. Часовий ряд містить дві змінні: незалежну змінну *X*, яка представляє інтервали часу, і залежну змінну *Y*, яка представляє характеристику досліджуваного явища в певний момент часу.

Середнє змінне – це статистичний метод, який дозволяє виявити характер змін значення *Y* протягом певного інтервалу часу та передбачити його в майбутньому. Цей метод діє, коли значення *Y* простежують чітку тенденцію в динаміці.

Середнє значення для будь-якого показника, який буде досліджено, розраховується як сума всіх спостережень, поділена на кількість спостережень формула (2.4):

$$\bar{x} = \frac{x_1 + x_2 + \cdots + x_n}{n}, \quad (2.4)$$

де x_1, x_2, x_n – числові значення величини, яка досліджується;

$\bar{x}$ – середнє значення величини, яка досліджується;

n – загальна кількість значень.

Абсолютне відхилення – це різниця між отриманим показником та базовим значенням, виражена за модулем, тобто без урахування знаку плюс або мінус. Це показує, наскільки дане значення відрізняється від базового значення.

Використовуючи абсолютне відхилення, можна оцінити тенденцію зміни величини. Наприклад, якщо абсолютне відхилення зростає, це може означати, що величина віддаляється від базового значення.

В деяких застосуваннях абсолютне відхилення використовується для визначення, наскільки величина, отримана в результаті дослідження, відрізняється від норми. Наприклад, у вимірах якості абсолютне відхилення $|x - \bar{x}|$ може бути використане для визначення, наскільки результат вимірювання відхиляється від заданого стандарту якості.

Відносне відхилення – $\frac{|x - \bar{x}|}{x} * 100\%$.

Для того, щоб розрахувати абсолютне відхилення, необхідно від отриманого показника відняти базовий. Величина відхилення виражається по модулю, тобто без урахування знаків «плюс чи «мінус» перед значенням. За отриманого відхилення можливо оцінити тенденцію зміни величини. У деяких вимірах абсолютне відхилення використовується для визначення наскільки величина, отримана в результаті досвіду, відрізняється від норми.

Для розрахунку абсолютного відхилення у відсотках необхідно обчислити відсоткову частку отриманого відхилення від базового показника. Отже, необхідно обчислене абсолютне відхилення помножити на 100 відсотків і розділити на базовий показник. Обчислена величина обчислюється у

відсотках і показує яку частку від базового показника займає абсолютне відхилення.

Потім потрібно відняти середнє арифметичне від кожного елемента вибірки. Кожну отриману різницю слід звести в квадрат і у результаті буде отримано середнє квадратичне відхилення формула (2.5):

$$\partial = \sqrt{\frac{(x_1 - \bar{x})^2 + \ldots + (x_n - \bar{x})^2}{n}}. \quad (2.5).$$

Середнє квадратичне відхилення так само, як і середнє лінійне відхилення, показує, на скільки в середньому відхиляються конкретні значення ознаки від середнього їхнього значення. Середнє квадратичне відхилення завжди більше середнього лінійного відхилення.

2.3 Використання методу ковзного середнього для аналізу даних

Припустимо, що є вебсторінка, де представлено математичні тренажери (комерційна сторінка сайту). На ній відображаються різні групи математичних тренажерів, які є об'єктом інтересу відвідувачів сайту. Більшість таких сторінок мають плиткову структуру, де для кожного тренажера надається мініатюрне зображення, інформація про вартість, рейтингові індикатори, спеціальні пропозиції, маркетингові переваги, посилання на відгуки користувачів та детальну інформацію про кожен математичний тренажер (рис. 2.1).

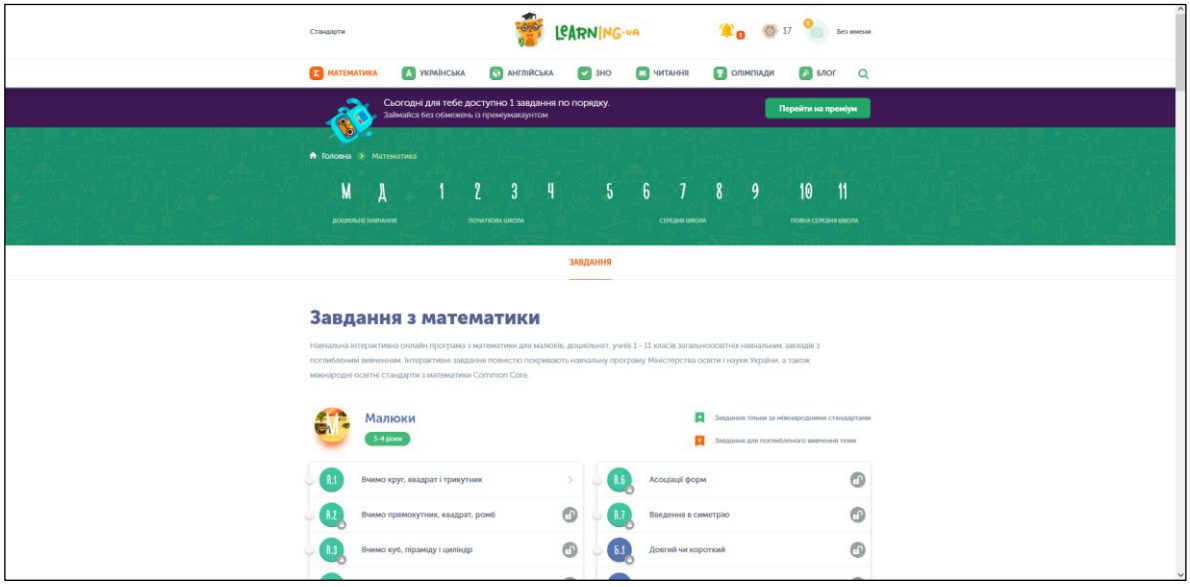

Рисунок 2.1 – Сторінка сервісу Learning.ua

Коли перед споживачем стоїть вибір серед багатьох математичних тренажерів у одній групі, це може бути досить складно, оскільки зовні вони виглядають схожими, а характеристики мають незначні відхилення, зазвичай у одному діапазоні. Крім ціни, зрозуміти різницю між ними та визначити, який тренажер кращий, є досить важко.

Основними проблемами, які негативно впливають на зручність користування сайтом, є складна або незрозуміла структура; відсутність зручної навігації; невдале використання реклами; невдале розташування та дизайн елементів; надто повільне завантаження. Наявність хоча б однієї з перерахованих проблем може досить негативно вплинути на місце сайту в пошуковій видачі, знизити його «видимість» у інформаційному просторі, що, у свою чергу, призведе до зниження відвідуваності сайту, втрати потенційних клієнтів та зниження його конверсії.

2.4 Експериментальні дослідження за результатами аналітики сайту

Сучасний вебсайт для математичних тренажерів є інтерактивною платформою, яка рекламує різні тренажери або пов'язану з ними послугу, приймає замовлення, пропонує різні варіанти оплати та надає рахунок. Такі

сайти створюються з використанням спеціальних систем управління контентом, оснащених потрібними модулями.

Математичний тренажер є інтерактивною електронною платформою, яка призначена для представлення, демонстрації та навчання математичних навичок онлайн. Це реалізоване навчальне представництво певної освітньої організації або приватного підприємства, яке функціонує як складова частина системи дистанційної освіти та навчання. За допомогою створення власного вебсервера, математичні тренажери надають можливість користувачам отримати доступ до навчальних матеріалів та навичок онлайн, що сприяє зручності та доступності навчання з будь-якого місця та будь-якої пори.

Аналіз інформації на сайті.

Для аналізу даних щодо продажів та інших конверсійних дій проведено дослідження на сайті освітньої платформи Learning.ua. Компанія займається реалізацією різних математичних тренажерів різного рівня складності, призначених для навчання різних вікових категорій. Каталог продукції розташований за посиланням «https://learning.ua». Головна сторінка сайту представлена на рис. 2.2.

На сторінках цього сайту представлена інформація про різні математичні тренажери, які є об'єктом інтересу відвідувачів. Також є можливість відстежувати корисну дію кожного тренажера, таку як конверсія або мікроконверсія, наприклад, фактів використання певного тренажера або отримання позитивного відгуку. На сайті компанії можна знайти каталоги різних математичних тренажерів, вибирати та порівнювати їх характеристики, але це може зайняти багато часу. Часто потенційний користувач може залишити сайт без жодного вибору.

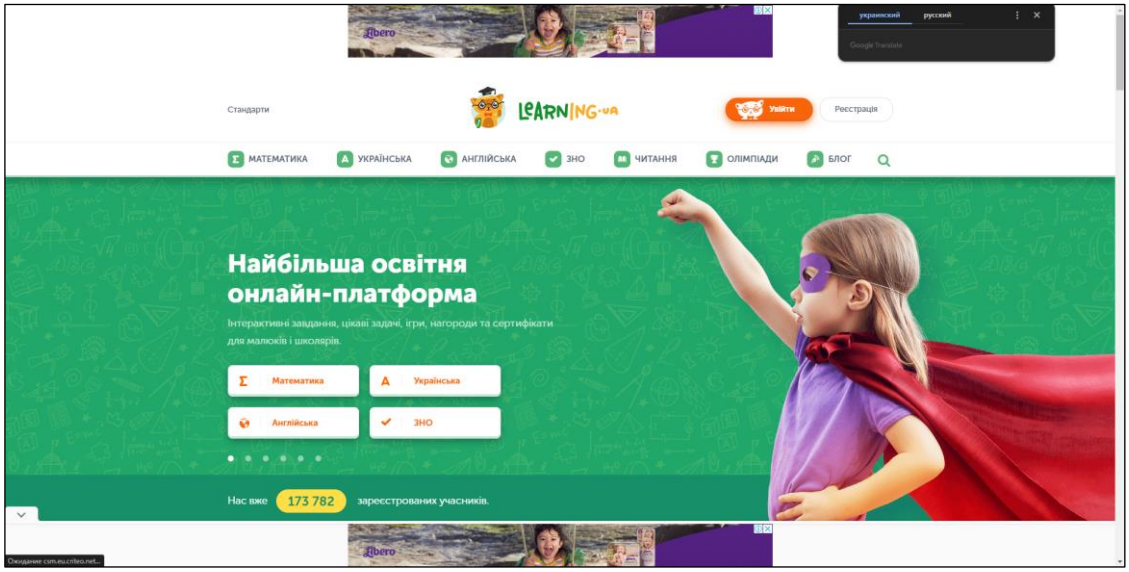

Рисунок 2.2 – Каталог тренажерів сайту «Learning.ua»

Інструменти сайту детально відстежують різні конверсійні дії протягом доби для кожного тренажеру , аналогічно до можливостей Google Analytics для обліку конверсійних дій на сайті. На прикладі рис. 2.3 та рис. 2.4 можна відстежити кількість відвідувачів протягом певного періоду часу.

Аналогічним чином, отримано дані про продаж товарів на сайті компанії "Learning.ua" за допомогою внутрішніх сервісів сайту. За даними про кількість продаж окремих товарів (онлайн-послуг) складено табл. 2.1. Отримано дані за один місяць, підраховані середні значення за попередні 2, 3 та 5 днів відповідно. За допомогою функції «Середнє згладжене за період» у електронній таблиці Microsoft Excel сформовано згладжені часові ряди методом змінного середнього (2.4). Знайдено середні відхилення згладжених часових рядів від заданого ряду – кількості продажів онлайн-послуги. Результати представлено у табл. 2.1 та на графіках вихідних даних зі згладженими значеннями за попередні 2, 3 та 5 днів відповідно.

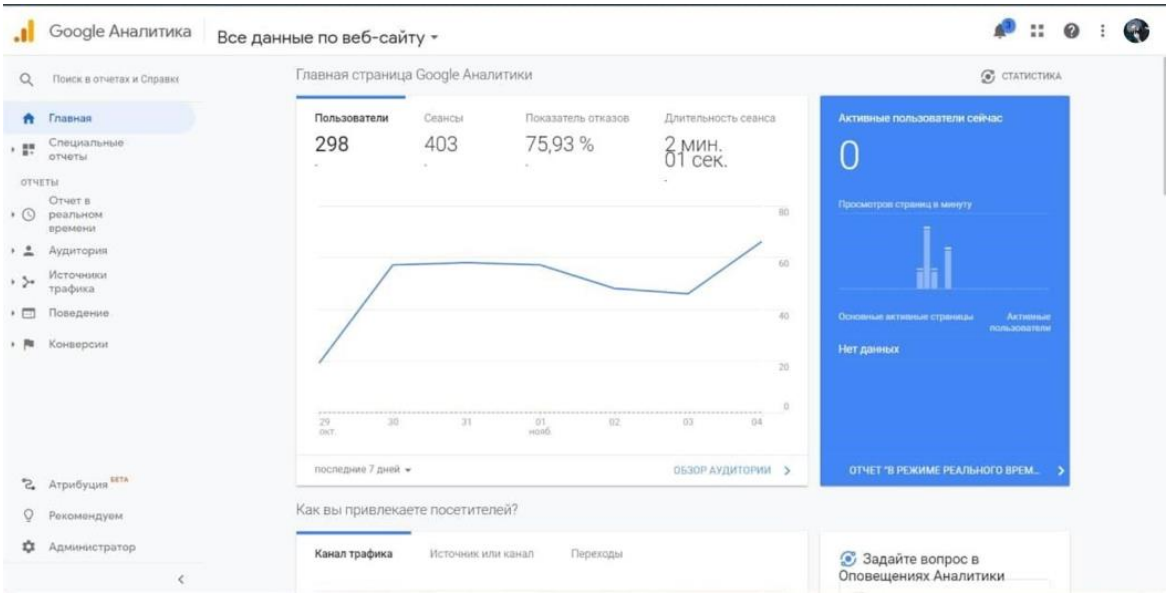

Рисунок 2.3 – Аналітика сайту, отримана за допомогою Google Analitics

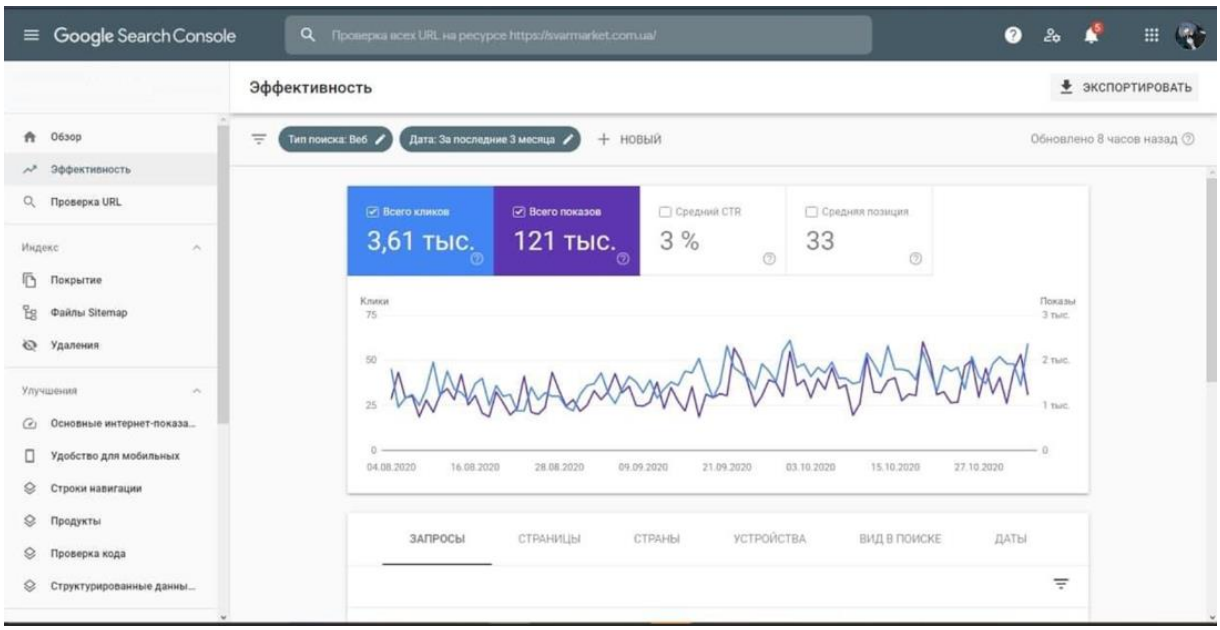

Рисунок 2.4 – Кількість кліків та ефективність сайту протягом одного місяця

Таблиця 2.1 – Ковзне середнє, абсолютне та відносне відхилення

Дні тижня	Кількість продаж, шт.	Ковзне середнє			Абсолютне відхилення			Відносне відхилення, %		
		за 2 дні	за 3 дні	за 5 днів	за 2 дні	за 3 дні	за 5 днів	за 2 дні	За 3 дні	за 5 днів
04.12.2023	26									
05.12.2023	50									
06.12.2023	45	38								

Кінець таблиці 2.1

Дні тижня	Кількість продаж, шт.	Ковзне середнє			Абсолютне відхилення			Відносне відхилення, %		
		за 2 дні	за 3 дні	за 5 днів			за 2 дні	за 3 дні	за 5 днів	
07.12.2023	16	23,75	20							
08.12.2023	14	19,25	21							
09.12.2023	15	15	17,5	18,1	0	2,5	3,1	0	8,335	10,335
10.12.2023	25	14,5	15	16,875	10,5	10	8,125	21	20	16,25
11.12.2023	15	20	18	17,5	5	3	2,5	16,665	10	8,335
12.12.2023	22	20	18,5	17,25	2,5	4	5,25	5,555	9,26	11,665
13.12.2023	15	18,75	21	19,375	3,75	6	4,375	12,5	19,445	14,585
14.12.2023	13	18,75	17,5	19,375	5,75	4,5	6,375	22,115	17,31	24,52
15.12.2023	17	14	17	16,375	3	0	0,625	8,825	0,49	1,84
16.12.2023	18	15	15	16,875	3,5	3,5	1,625	9,46	9,46	4,39
17.12.2023	12	17,75	16	15,875	5,25	3,5	3,375	21	14,665	13,5
18.12.2023	13	15,5	16	15,25	2	2,5	1,75	7,405	9,26	6,48
19.12.2023	15	13	15	15,375	2	0	0,375	6,665	0,555	1,25
20.12.2023	20	14,25	13,5	14,875	5,75	6,5	5,125	14,375	15,835	12,815
21.12.2023	15	17,5	16	15,25	2,5	1	0,25	8,335	3,89	0,835
22.12.2023	12	17,5	16,5	15,875	5	4	3,375	20	16,665	13,5
23.12.2023	16	13,75	16	15,625	2,25	0	0,375	7,03	0,52	1,17
24.12.2023	14	14,25	14,5	15,875	0,25	0	1,375	0,86	0	4,74
25.12.2023	12	15,25	14,5	14,5	2,75	2	2	11	7,335	8
26.12.2023	12	13,5	14,5	13,875	1,5	2,5	1,875	6,25	9,72	7,815
27.12.2023	14	12,25	13	13,75	1,75	1	0,25	6,25	3,57	0,895
28.12.2023	13	13	13	13,25	0,5	0,5	0,25	1,85	2,47	0,925
29.12.2023	12	13,75	13	13	1,25	0,5	0,5	5	2,665	2

Згідно з даними табл. 2.1 були створені діаграми для показу кількості продаж окремого товару (онлайн-послуги), з урахуванням ширини вікна згладжування за допомогою рухомого середнього за періоди 2, 3 та 5 днів (рис. 2.5).

На рис. 2.5 можна побачити, що тренди ліній згладжених часових рядів (для періодів 2, 3 та 5 днів) зсунуті відносно trend лінії вихідного часового ряду – кількості продаж за обраним товаром (онлайн-послугою). Це відбувається тому, що значення згладжених часових рядів, розрахованих на

основі даних попередніх спостережень, затримуються в порівнянні з відповідними значеннями заданого ряду. Проте, лінії графіку на рис. 2.6 виглядають більш гладкими.

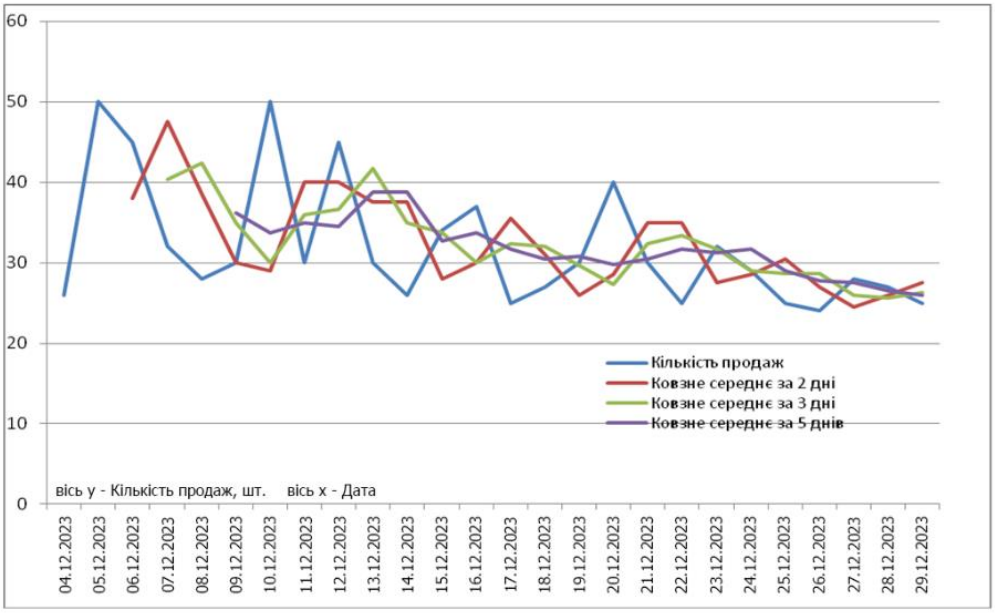

Рисунок 2.5 – Графіки вихідних даних зі згладженими значеннями за попередні два, три та п’ять днів

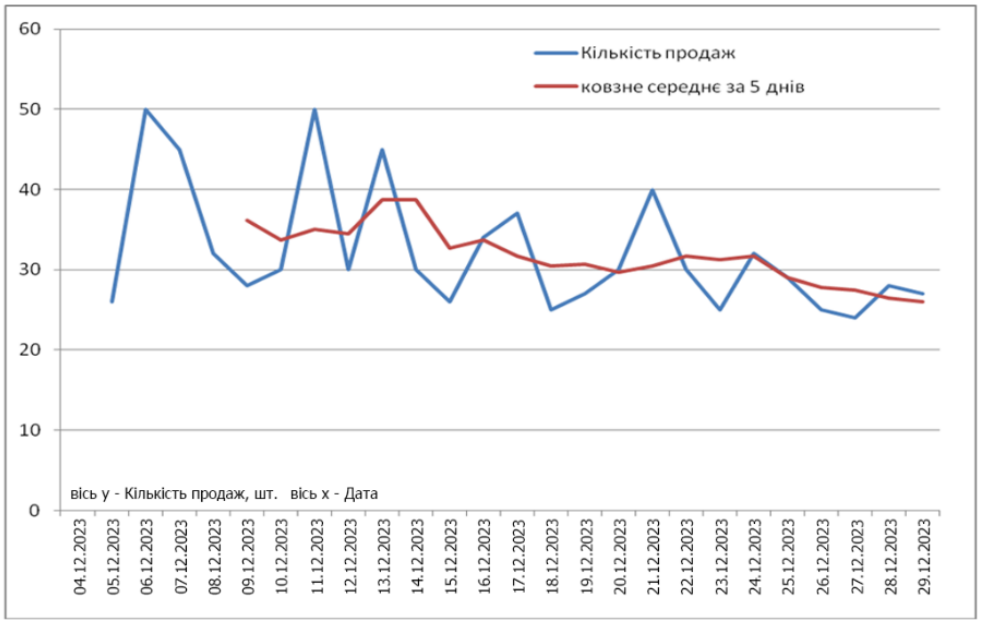

Рисунок 2.6 – Вихідний ряд кількості продаж та ковзне середнє за 5 попередніх днів

Для визначення найкращого значення вікна згладжування для прогнозування продажів товару (онлайн-послуги), проведено порівняльний аналіз похибок згладжених часових рядів за різними періодами спостереження (2, 3 та 5 днів). Виявилося, що середнє квадратичне відхилення для 5-денного періоду є мінімальним (6,68), тоді як для 2-денного (7,92) та 3-денного (7,41) – вище. Таким чином, для даного набору значень продажів товару обрано значення вікна згладжування у 5 днів, щоб зменшити вплив поодиноких значних відхилень від середньостатистичних показників продажів товару (онлайн-послуги). Аналогічний аналіз необхідно провести для кожної товарної позиції.

Висновки до розділу 2

У другому розділі було розглянуто математичні методи аналізу статистичних даних, отриманих з внутрішньої аналітики сайту, на прикладі математичних тренажерів. Також на основі розглянутих розрахунків методом ковзного середнього проведено експериментальні дослідження шляхом аналізу продаж освітніх онлайн-послуг. При цьому визначався розмір вікна згладжування вихідних даних, представлені графіки динаміки використання математичних тренажерів. Для дослідження використовувались вихідні дані з внутрішньої аналітики діючого сайту "Learning.ua", який спеціалізується в реалізації математичних та інших тренажерів для школярів.

3 ПРОЄКТУВАННЯ ПРОГРАМНОЇ СИСТЕМИ

3.1 Життєвий цикл інформаційної системи

Будь-яка інформаційна система має свій життєвий цикл, який можна розбити на кілька етапів:

1) аналіз вимог.

2) проєктування.

3) розробка.

4) тестування.

5) впровадження.

6) експлуатація.

7) модернізація.

8) виведення з експлуатації.

Для побудови таблиці етапів життєвого циклу можна розбити онлайн-тренажер з математики на кілька підсистем, виходячи з функціональних вимог (табл. 3.1–3.7):

– база даних зберігає дані користувачів, їхній прогрес в освоєнні навичок, рейтинги учнів тощо;

– особистий кабінет учня показує учневі його прогрес, тренажери, які йому необхідно повторити, а також різні налаштування;

– сервіс тренажерів генерує завдання для тренажерів.

– сервіс вхідної діагностики визначає поточні навички учня, щоб правильно побудувати для нього індивідуальну траєкторію навчання.

– сервіс оплати дає змогу оплачувати послуги онлайн тренажера різними методами;

– особистий кабінет батьків дає змогу керувати налаштуваннями профілю учня, бачити його прогрес та інше;

– особистий кабінет репетитора дає змогу керувати налаштуваннями тренажерів для окремо взятих учнів, відстежувати статистику учнів і коригувати індивідуальні траєкторії.

Таблиця 3.1 – Підсистема 1. База даних

Аналіз вимог	Виявлення вимог до БД
Проєктування	Проєктування архітектури БД
Розробка	Документування БД; Розробка БД
Тестування	Навантажувальне тестування БД
Впровадження	Передача, встановлення та налаштування БД
Експлуатація	Адміністрування; Виявлення недоліків під час експлуатації
Модернізація	Усунення недоліків БД, виявлених під час експлуатації
Виведення з експлуатації	Виведення системи з експлуатації

Таблиця 3.2 – Підсистема 2. Сервіс тренажерів

Аналіз вимог	Виявлення та документування функціональних вимог для сервісу тренажерів
Проєктування	Проєктування архітектури сервісу тренажерів
Розробка	Розробка дизайну для сервісу тренажерів; Написання коду для сервісу тренажерів
Тестування	Складання тестів для сервісу тренажерів; Мануальне тестування
Впровадження	Передача, встановлення та налаштування сервісу тренажерів
Експлуатація	Адміністрування; Виявлення недоліків сервісу тренажерів під час експлуатації
Модернізація	Усунення недоліків сервісу тренажерів, виявлених під час експлуатації
Виведення з експлуатації	Виведення системи з експлуатації

Таблиця 3.3 – Підсистема 3. Особистий кабінет учня

Аналіз вимог	Виявлення та документування функціональних вимог для особистого кабінету учня
Проєктування	Проєктування архітектури особистого кабінету учня
Розробка	Розробка дизайну для особистого кабінету учня; Написання коду для особистого кабінету учня
Тестування	Складання тестів для особистого кабінету учня; Мануальне тестування
Впровадження	Передача, встановлення та налаштування особистого кабінету учня
Експлуатація	Виявлення недоліків особистого кабінету клієнта під час експлуатації
Модернізація	Усунення недоліків особистого кабінету учня, виявлених під час експлуатації
Виведення з експлуатації	Виведення системи з експлуатації

Таблиця 3.4 – Підсистема 4. Сервіс сплати

Аналіз вимог	Виявлення та документування функціональних вимог для сервісу оплати
Проєктування	Проєктування архітектури сервісу оплати
Розробка	Розробка дизайну для сервісу оплати
Тестування	Складання тестів для сервісу оплати. Мануальне тестування
Впровадження	Передача, встановлення та налаштування сервісу оплати
Експлуатація	Адміністрування. Виявлення недоліків сервісу оплати під час експлуатації
Модернізація	Усунення недоліків сервісу оплати, виявлених під час експлуатації
Виведення з експлуатації	Виведення системи з експлуатації

Таблиця 3.5 – Підсистема 5. Сервіс вхідної діагностики

Аналіз вимог	Виявлення та документування функціональних вимог для сервісу вхідної діагностики
Проєктування	Проєктування архітектури сервісу вхідної діагностики
Розробка	Розробка дизайну для сервісу вхідної діагностики
Тестування	Складання тестів для сервісу вхідної діагностики; Мануальне тестування
Впровадження	Передача, встановлення та налаштування сервісу вхідної діагностики
Експлуатація	Адміністрування; Виявлення недоліків сервісу вхідної діагностики під час експлуатації
Модернізація	Усунення недоліків сервісу вхідної діагностики, виявлених під час експлуатації
Виведення з експлуатації	Виведення системи з експлуатації

Таблиця 3.6 – Підсистема 6. Особистий кабінет репетитора

Аналіз вимог	Виявлення та документування функціональних вимог для особистого кабінету репетитора
Проєктування	Проєктування архітектури особистого кабінету репетитора
Розробка	Розробка дизайну для особистого кабінету репетитора Написання коду для особистого кабінету репетитора Підключення особистого кабінету репетитора до БД
Тестування	Складання тестів для особистого кабінету репетитора. Мануальне тестування
Впровадження	Передача, встановлення та налаштування особистого кабінету репетитора
Експлуатація	Оперативне обслуговування; Адміністрування. Виявлення недоліків системи під час експлуатації

Кінець таблиці 3.6

Модернізація	Усунення недоліків системи, виявлених під час експлуатації; Впровадження нового функціоналу в систему
Виведення з експлуатації	Виведення системи з експлуатації

Таблиця 3.7 – Підсистема 7. Особистий кабінет батьків

Аналіз вимог	Виявлення та документування функціональних вимог для особистого кабінету батьків
Проєктування	Проєктування архітектури особистого кабінету батька;
Розробка	Розробка дизайну для особистого кабінету батьків. Написання коду для особистого кабінету співробітника. Підключення особистого кабінету співробітника до БД
Тестування	Складання тестів для особистого кабінету батьків. Мануальне тестування
Впровадження	Передача, встановлення та налаштування особистого кабінету батьків
Експлуатація	Адміністрування. Виявлення недоліків особистого кабінету батьків під час експлуатації
Модернізація	Усунення недоліків особистого кабінету батька, виявлених під час експлуатації
Виведення з експлуатації	Виведення системи з експлуатації

3.2 Концепція тренажера

Математичний тренажер для школярів має бути хмарним рішенням, що складається з генератора завдань (на кожну відпрацьовувану навичку), бази даних (містить інформацію про користувачів і пройдену ними навчальну траєкторію), архіву відеофрагментів із прикладами та поясненнями за кожною навичкою, діагностичного блока, аналітичного блока (формує індивідуальну навчальну траєкторію, контролює проходження кожної її ділянки), блока візуалізації (business intelligence, візуалізує індивідуальну навчальну

траєкторію в просторі математичних навичок і накопичені досягнення), блок модератора-підказувача (генерує підказки щодо виконання завдань і дотримання технології тренувань на підставі моніторингу дій користувача), блок CRM (комунікації з учнем його куратором – батьком чи педагогом), блок приймання заявок і навігації по партнерах сервісу з урахуванням територіальної локалізації (для трьох груп – батьків і самих школярів, репетиторів, шкіл/навчальних центрів).

Тренажер призначений для відпрацювання таких п'яти основних груп математичних навичок:

– переведення умови задачі з природної мови на мову математичних символів;
– складання математичної моделі процесу/об'єкта;
– складання рівняння за наявною моделлю;
– розв'язання рівняння;
– розв'язання задач.

Кожна з перелічених груп навичок складається з набору елементарних, комбінованих, складових навичок.

Зокрема, група навичок розв'язування рівнянь містить у собі набір таких арифметичних та алгебраїчних навичок:

– додавання і віднімання в різних межах і різними способами;
– розбиття числа на суму в різних варіантах;
– таблиця множення;
– множення різними методами;
– дії зі звичайними дробами;
– перетворення звичайних дробів у десяткові та навпаки;
– подання одного й того самого числа в різних видах;
– заміна множення на ділення і навпаки;
– підбір коренів;

– перенесення членів рівняння з боку в бік;
– групування та приведення подібних;
– дія за алгоритмом або визначенням.

Після вхідної діагностики поточного рівня володіння математичними навичками користувачеві пропонується індивідуальна навчальна траєкторія. Кожен наступний за складністю розділ тренажера дає змогу відпрацювати комбінацію використання простіших навичок. Навички відпрацьовуються одна за одною по черзі згідно з авторською методикою. Завдання для відпрацювання кожної навички генеруються за певним алгоритмом, таким чином завдання не повторюються, а їхня можлива кількість дуже велика.

Повна діаграма варіантів використання сервісу наведена на рис. 2.1.

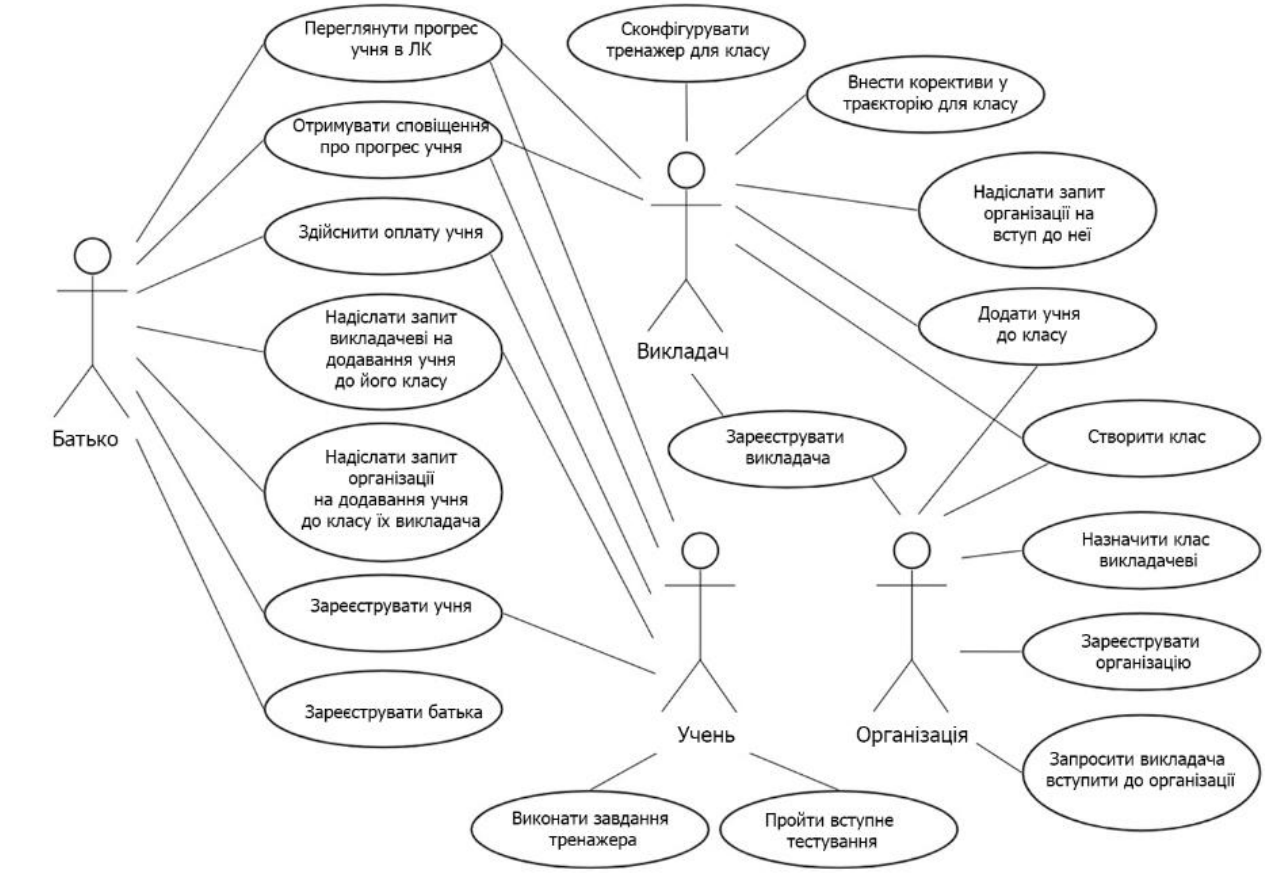

Рисунок 3.1 – Діаграма варіантів використання сервісу

Перехід до відпрацювання наступної навички допускається тільки після повного відпрацювання попередньої (виконання поспіль заданої кількості

завдань без помилок за певний час). Вибір наступної для відпрацювання пропонується системою згідно з авторською методикою з метою спрямування користувача за потрібною (оптимальною) навчальною траєкторією.

За наявності помилок у відпрацюванні складніших навичок система може запропонувати повторення простіших навичок, що становлять цю складну навичку, щоб локалізувати причину помилок, яких припускаються, і їх точкового опрацювання.

У разі невпевнених дій користувача (під час виконання окремого завдання, у разі відхилення від рекомендованого режиму тренувань тощо) система пропонує йому підказку (спливаюче вікно). Інформацію про всі завершені дії користувача (чергове тренування, черговий етап навчальної траєкторії тощо) і досягнуті ним результати автоматично надсилають батькам і педагогу на електронну пошту.

У тренажері також передбачено особисті кабінети для батьків, репетиторів (вчителів) і організацій (репетиторських центрів або шкіл).

У кабінеті батька виводиться прогрес дитини, приходять і налаштовуються повідомлення. Через кабінет батька можна зареєструвати дитину і поповнити її рахунок. Також батьки можуть направляти запити репетиторам або репетиторським центрам на додавання своєї дитини в їхню програму навчання.

Організації можуть як реєструвати нових репетиторів, так і запрошувати вже існуючих приєднається до організації. Кожна організація може створювати класи, в які можна буде додавати учнів. Організації можуть призначати вчителів у класи.

Функціонал створення та управління класами також доступний і для індивідуальних репетиторів. Основною роллю репетиторів у тренажері буде його конфігурація для учнів. Репетитори також бачитимуть усю статистику як окремо взятого учня, так і цілого класу для більш якісної конфігурації тренажерів і коригування індивідуальних траєкторій навчання.

3.3 Модель проєктування

Для цього вебзастосунку обрана модель клієнт-сервер, яка широко застосовується для створення вебзастосунків. За цією моделлю, клієнтська та серверна частини працюють разом через мережу, здійснюючи обмін даними та обробку запитів і відповідей (рис. 3.2).

Рисунок 3.2 –Клієнт-серверна архітектура

Інтерфейс користувача, який є клієнтською частиною, може бути реалізований у вигляді веббраузера, мобільного застосунку або іншого програмного забезпечення для доступу до ресурсів сервера. Він відповідає за візуалізацію інформації, збір та обробку даних від користувача, а також надсилання запитів до сервера.

Сервер, з іншого боку, обробляє ці запити, виконує бізнес-операції, надає доступ до баз даних та надсилає відповіді назад клієнту. Сервер може бути або фізичним, або віртуальним комп'ютером, на якому працює серверний застосунок або вебсервер.

Коли користувач заходить на вебсайт або використовує мобільний застосунок, його пристрій (клієнт) надсилає запит до сервера за допомогою протоколу HTTP. Запит містить необхідні дані, такі як URL, параметри запиту, заголовки тощо. Сервер отримує цей запит, обробляє його та виконує необхідні дії, після чого надсилає відповідь назад до клієнта.

Клієнт-серверна модель дозволяє ефективно розподілити завдання між двома частинами, забезпечуючи гнучкість, масштабованість, високу

швидкість роботи та легкість у розробці, підтримці безпеки та контролі доступу. Цей підхід широко використовується у вебзастосунках, оскільки він дозволяє різноманітним пристроям доступ до серверних ресурсів, а сервери забезпечують централізований доступ до ресурсів та бізнес-логіки. Клієнт-серверна модель дозволяє створювати потужні застосунки, які можуть обслуговувати велику кількість користувачів одночасно.

Клієнт-серверна архітектура має ряд переваг, зокрема:

1) розділення ролей клієнта та сервера: користувацький інтерфейс та взаємодії з ним обробляються на клієнтській стороні, в той час як сервер займається обробкою інформації, бізнес-логікою та зберіганням даних;

2) масштабованість системи: сервер можна масштабувати за потребою, додаючи ресурси або додаткові сервери, що забезпечує ефективність при високих навантаженнях;

3) високий рівень безпеки: клієнти не мають прямого доступу до сервера, що знижує ризик несанкціонованого доступу. Використання методів аутентифікації та авторизації додатково захищає конфіденційність даних;

4) використання різних технологій: можна використовувати різні платформи та технології на клієнтській та серверній сторонах, що дозволяє створювати застосунки, сумісні з різними пристроями та операційними системами;

5) співпраця в командах: розробка клієнтської та серверної частин може проводитися незалежно, навіть різними командами або компаніями, що сприяє розподілу робочих процесів, прискоренню розробки та підвищенню якості програмного забезпечення.

3.4 Вимоги до тренажера

На підставі аналізу тренажерів і запропонованої концепції було сформовано такі вимоги:

– автоматичний генератор завдань на кожну навичку;

– індивідуальна траєкторія відпрацювання навичок;

– кількісна оцінка рівня освоєння навички та стійкості отриманого результату;

– кількість навичок має бути не менше 300 штук;

– точна вхідна діагностика;

– зручний і простий у використанні інтерфейс;

– здатність витримати навантаження не менше 2’500 користувачів одночасно;

– масштабованість.

Функціональні вимоги:

– система тренажерів;

– особистий кабінет учня;

– особистий кабінет батьків;

– особистий кабінет репетитора;

– кабінет організації;

– модуль оплати;

– модуль повідомлень;

– модуль вхідної діагностики;

– модуль класів;

– модуль статистики;

– ігрові механіки.

3.5 Вибір технологій для розробки

Розробка прототипів сторінок проводиться в онлайн-сервісі Wireframe. Цей сервіс дає змогу створювати прості прототипи, які являють собою спрощене розташування різних елементів сторінки.

Інтерфейс Wireframe інтуїтивно-зрозумілий і простий в освоєнні, тому робота з ним не вимагає спеціального навчання.

Після розробки прототипу сторінки, необхідна розробка дизайну. Найпопулярнішим безкоштовним рішення є сервіс Figma. Figma – це графічний онлайн-редактор для спільної роботи. На етапі розробки дизайну розташовані елементи прототипу з Wireframe отримують кольори, тіні, картинки, а також анімації. Figma додатково дає змогу створювати слайдери, ефекти наведення і паралакса, а також дає змогу імітувати поведінку вже реального сайту. Для роботи з Figma потрібні спеціалізовані навички, але при правильному використанні цього інструменту можна отримати робочий макет сайту, не написавши жодного рядка коду. Також при грамотній роботі дизайнера фронтенд розробник зможе отримати готові CSS-стилі, які можна буде використовувати в розробці, що значно скоротить час розробки інтерфейсу.

Для виконання вимог щодо максимального навантаження і масштабованості сервісу тренажера необхідно грамотно побудувати архітектуру застосунку і вибудувати грамотну і безперервну взаємодію програмістів і сисадмінів. У цьому добре допомагає методологія DevOps. DevOps з'явився 2009 року як відповідь на проблему в комунікації між системними адміністраторами та програмістами. Розробники створювали код і передавали його сисадмінам, які займалися підтримкою та експлуатацією, у формі архіву з інструкцією для встановлення. Інструкція часто була різною для різних версій цього коду і часом не враховувала специфіку оточення. Така схема істотно подовжувала розробку ПЗ.

У DevOps є два головні принципи:

1) сприйняття інфраструктури (сервери, налаштування) як єдиного коду, в який досить внести одну поправку, щоб весь проєкт змінився. Ця ідея виключає необхідність ручного налаштування окремих елементів;

2) багатофункціональність програмістів, які не просто пишуть код, а усвідомлюють, як це впливає на весь проєкт.

Для виконання цих принципів необхідний такий інструментарій:

– система контролю версій;
– система CI/CD;
– система пакування застосунку;
– система автоматичної оркестровки та балансування.

Як систему контролю версій найкраще використовувати Git. Git – це найпопулярніша сучасна система керування версіями. У неї активна підтримка і відкритий вихідний код.

Git – це система керування версіями з розподіленою архітектурою. На відміну від колись популярних систем на кшталт CVS і Subversion (SVN), де повна історія версій проєкту доступна лише в одному місці, у Git кожна робоча копія коду сама по собі є репозиторієм. Це дозволяє всім розробникам зберігати історію змін у повному обсязі.

Для зберігання основної кодової бази було обрано онлайн-платформу GitLab. GitLab – вебінструмент життєвого циклу DevOps з відкритим вихідним кодом, що представляє систему управління репозиторіями коду для Git з власною вікі, системою відстеження помилок, а також з функцією CI/CD.

Для пакування різних частин застосунку (бази даних, фронтенд-сервера, бекенд-сервера тощо) найпопулярнішим інструментом є Docker. Docker упаковує кожен мінімально-залежний один від одного сервіс у віртуальний контейнер, що дає змогу ізолювати його від зовнішнього впливу, а також вирішити безліч проблем із сумісністю різних залежностей.

Як основна база для заявлених вимог добре підходить PostgreSQL. PostgreSQL – це сучасна об'єктно-реляційна база даних з відкритим вихідним кодом, яку активно розробляють понад 30 років. Вона заслужила міцну репутацію за надійність, широкий набір функцій і високу продуктивність.

Для швидкого і якісного розроблення бекенд-сервера підходить мова програмування Python і фреймворк Django [14; 15]. Це повністю безкоштовний фреймворк із відкритим вихідним кодом. Для того, щоб зробити з нього REST API, необхідно використовувати розширення DRF (Django REST framework).

Для розробки вебінтерфейсу було обрано безкоштовний фреймворк із відкритим вихідним кодом Vue [21], його storage manager Vuex [22], а також його SSR-обгортку.

3.6 Розробка дизайну тренажера

Процес розробки починається зі створення прототипів основних сторінок у Wireframe (рис. 3.2):

1) Головна сторінка;
2) Сторінка списку тренажерів;
3) Сторінка тренажера;
4) Сторінка запущеного тренажера;
5) Сторінка результатів завершеного тренажера.

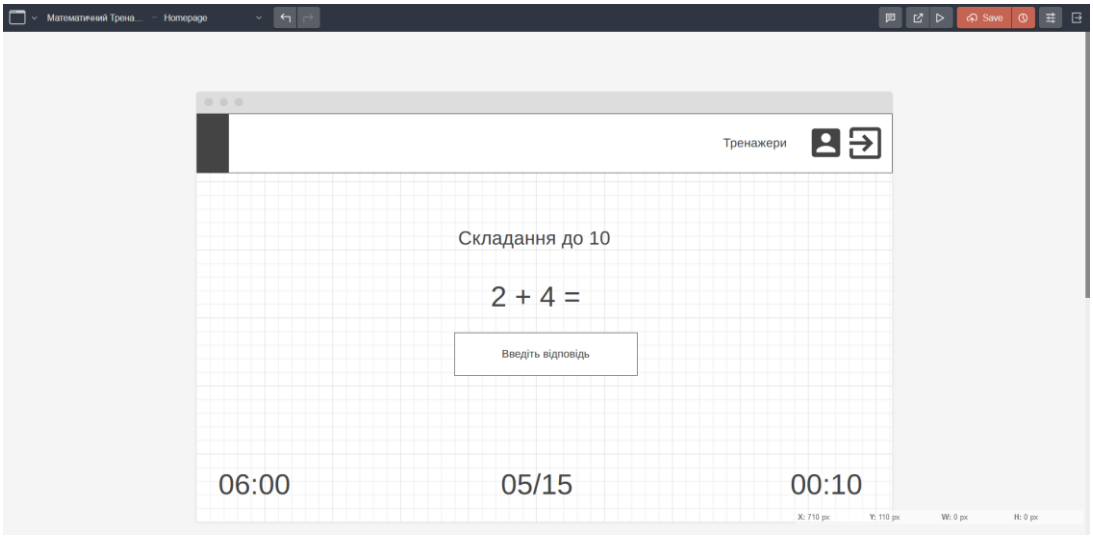

Рисунок 3.2 – Приклад прототипу сторінки запущеного тренажера

Після завершення створення прототипів необхідно створити повноцінний дизайн сторінок у Figma (рис. 3.3).

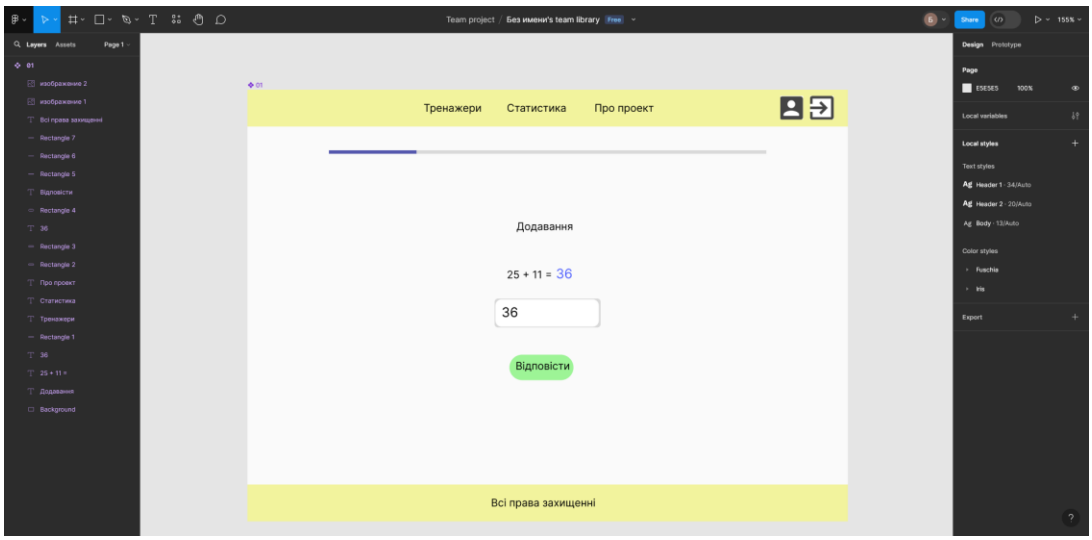

Рисунок 3.3 – Макет сторінки в Figma

Макет сторінки в Figma дозволяє детально пропрацювати кожну елемент сторінки, надати їй зручний та інтуїтивно зрозумілий інтерфейс, а також оптимізувати взаємодію між користувачем та системою. Розробка дизайну тренажера є невіддільною частиною створення успішного навчального інструменту для школярів. Таким чином, розробка дизайну тренажера є важливою складовою розробки інформаційної системи онлайн-тренажера для навчання математики у школярів та забезпечує зручне та ефективне навчання.

Висновки до розділу 3

У третьому розділі було розглянуто життєвий цикл інформаційної системи, також описані етапи життєвого циклу інформаційної системи: аналіз вимог, проєктування, розробка, тестування, впровадження, експлуатація, модернізація, виведення з експлуатації.

На підставі аналізу конкурентів із розділу 1 було побудовано концепцію тренажера. Визначено всі його функціональні особливості, а також вимоги. Було підібрано технології та інструменти для подальшого розроблення.

Фінальною частиною став процес розробки та прототипування дизайну в Wireframe і в Figma. Проєкт має значний потенціал для покращення освітнього процесу для школярів.

4 РОЗРОБКА МАТЕМАТИЧНОГО ОНЛАЙН-СЕРВІСУ

4.1 Моделі бази даних

Оскільки онлайн-сервіс для формування навичок розв'язання математичних задач школярами передбачає особистий кабінет і збір даних про учнів, необхідна модель користувача:

1) ідентифікатор;
2) електронна пошта;
3) номер телефону;
4) дата і час реєстрації;
5) дата останньої активної сесії;
6) фотографія.

Щоб не створювати безліч моделей користувачів для кожного типу профілю (учень, батько або організація), необхідно закласти для них окремі моделі, пов'язані з моделлю користувача за ідентифікатором. Модель профілю учня:

1) ідентифікатор;
2) ідентифікатор користувача;
3) ім'я;
4) прізвище;
5) по батькові;
6) стать;
7) рік навчання в школі.

Модель тренажера, необхідна для опису тренажера через панель адміністратора:

1) ідентифікатор;
2) заголовок;
3) опис;
4) посилання;

5) залежності.

Для створення залежностей між тренажерами буде використовуватися проміжна таблиця.

Модель конфігурації тренажера дасть змогу в майбутньому конфігурувати тренажери індивідуально під кожного учня, через кабінет вчителя, що дасть змогу вибудовувати індивідуальні траєкторії. Модель конфігурації тренажера:

1) ідентифікатор;
2) ідентифікатор користувача;
3) ідентифікатор тренажера;
4) тип тестів (обмежений за часом або обмежений за кількістю запитань);
5) обмеження за часом;
6) обмеження за кількістю запитань.

Модель результатів тренажера показує зв'язок між користувачем і тренажером, необхідна для розробки складових навичок:

1) ідентифікатор;
2) ідентифікатор користувача;
3) ідентифікатор тренажера;

Модель тесту:

1) ідентифікатор;
2) ідентифікатор користувача;
3) ідентифікатор тренажера;

Модель тесту:

1) ідентифікатор;
2) ідентифікатор користувача;
3) ідентифікатор тренажера;
4) обмеження за часом;
5) обмеження за кількістю запитань;

6) дата і час початку виконання;
7) дата і час завершення виконання;
8) прапор – «тест пройдено?»;
9) оцінка.

Модель згенерованого завдання:

1) ідентифікатор;
2) ідентифікатор тесту;
3) ідентифікатор попереднього завдання;
4) дата і час початку виконання;
5) дата і час завершення виконання;
6) опис;
7) правильна відповідь;
8) відповідь;
9) прапор – «відповідь правильна?»;
10) час, витрачений на відповідь.

Повна архітектура розробленої бази даних представлена на рис. 4.2.

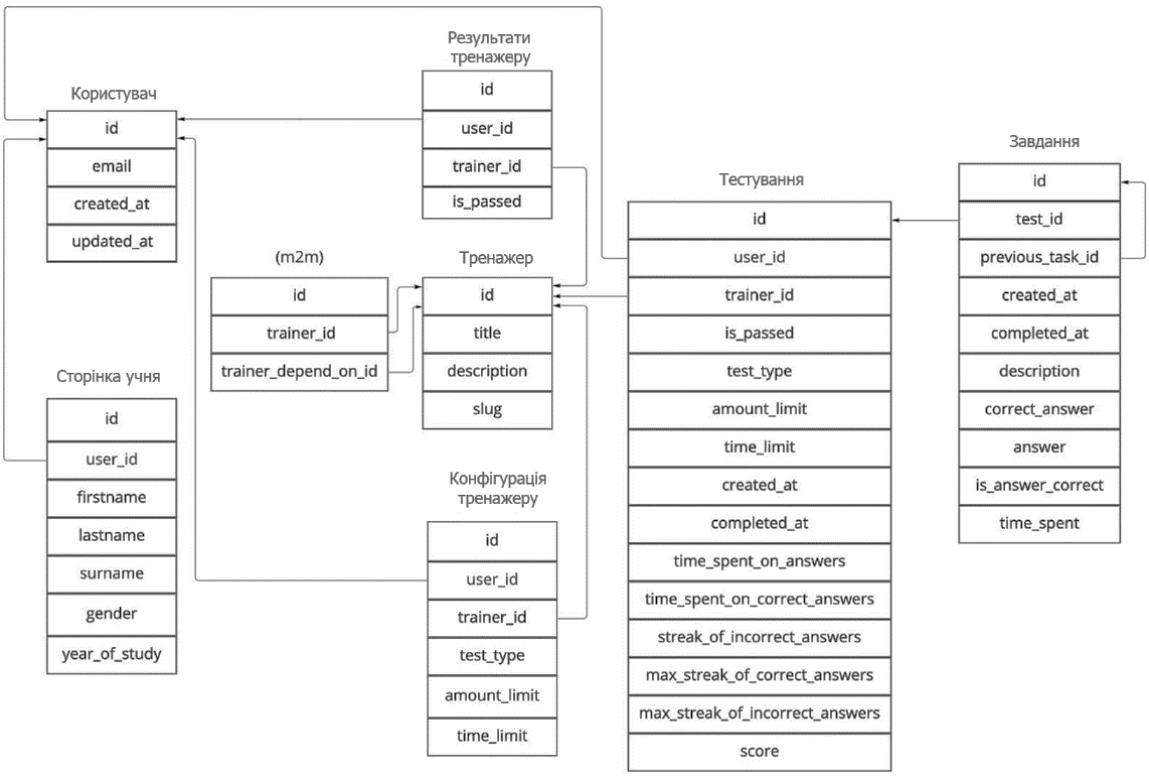

Рисунок 4.1 – Архітектура БД

4.2 Налаштування IDE

Для написання великих проєктів використовується IDE. Для даного проєкту підходить кросплатформний редактор коду від Microsoft з відкритим вихідним кодом Visual Studio Code. Даний редактор коду може працювати з усіма мовами програмування, і у нього більше ніж 5’000 різних розширень.

Для роботи необхідно встановити наступний набір розширень:

1) Code Spell Checker – перевірка орфографії;
2) Docker – швидкий доступ до контейнерів з IDE;
3) DotENV – для підсвічування синтаксису файлів типу *.env;
4) EditorConfig;
5) ESLint – автоматична корекція коду, написаного мовами програмування JavaScript і TypeScript;
6) GitLens – UI-інтерфейс для роботи з Git;
7) Python – для підсвічування синтаксису файлів типу *.py;
8) Sass – для підсвічування синтаксису файлів типу *.sass і *.scss;
9) Stylelint – автоматична корекція стилів;
10) Vetur – набір розширень для швидкої роботи з VueJS;
11) Vue VSCode Snippets – набір заготовок для автодоповнення коду у файлах типу *.vue;
12) JavaScript (ES6) code snippets – набір заготовок для автодоповнення коду у файлах типу *.js і *.ts.

4.3 Розробка алгоритму роботи тренажера та архітектури застосунку

Під час виконання роботи було розроблено спрощений алгоритм роботи тренажера (рис. 4.2). Цей алгоритм підходить як для тренажерів, обмежених за часом розв'язання, так і для тренажерів, обмежених загальною кількістю заданих завдань.

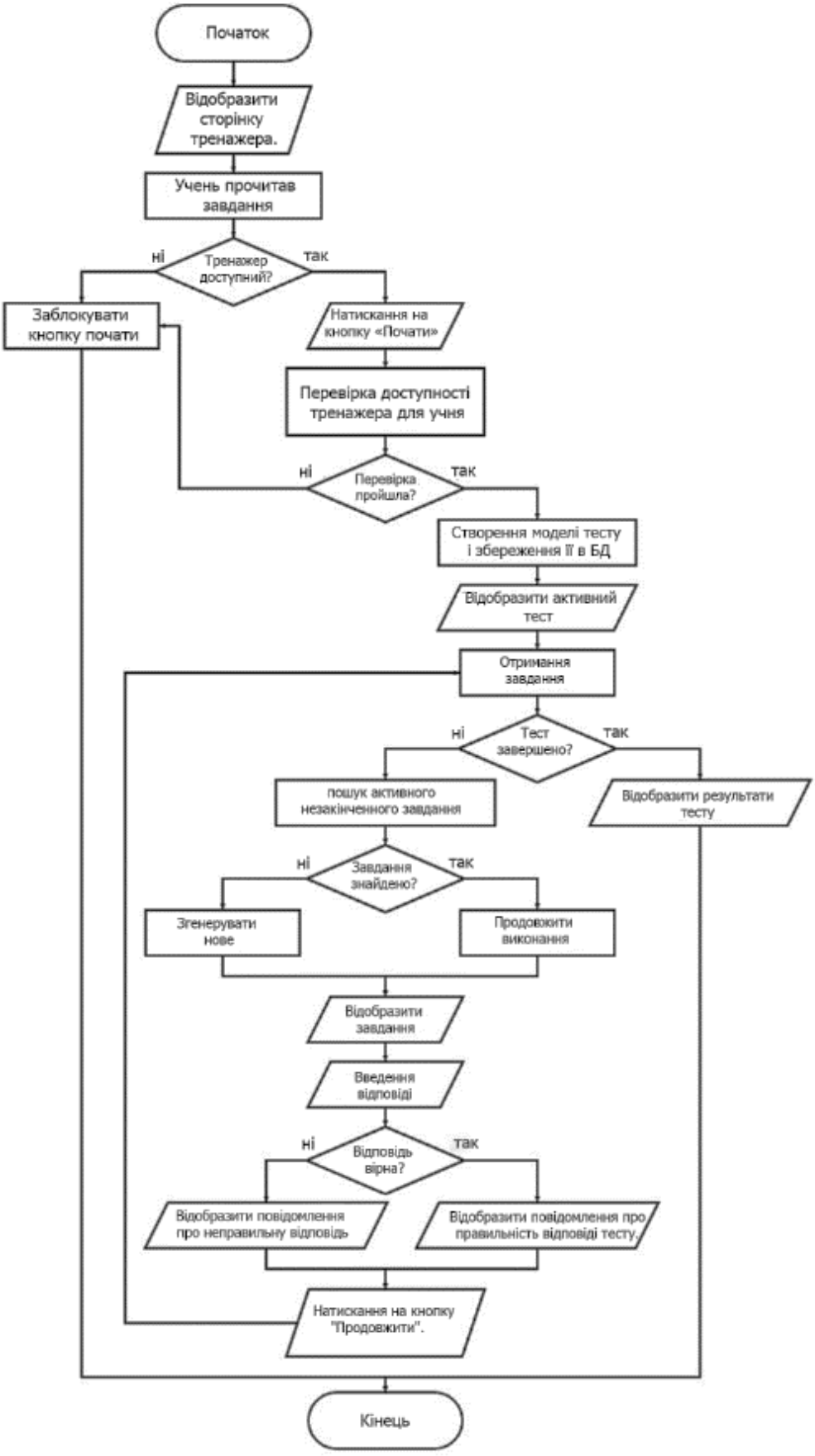

Рисунок 4.2 – Алгоритм роботи тренажера

Для реалізації алгоритму та розгортання ІС використання Kubernetes буде надлишковим, але використання Docker для контейнеризації PostgreSQL дасть змогу заощадити на розробці велику кількість часу. Налаштування docker-compose для даних сервісів, які були розглянуті раніше.

Також раніше було вже обрано поділ на вебінтерфейс і API-сервер.

З огляду на всі особливості сервісів, було побудовано архітектуру (рис. 4.3).

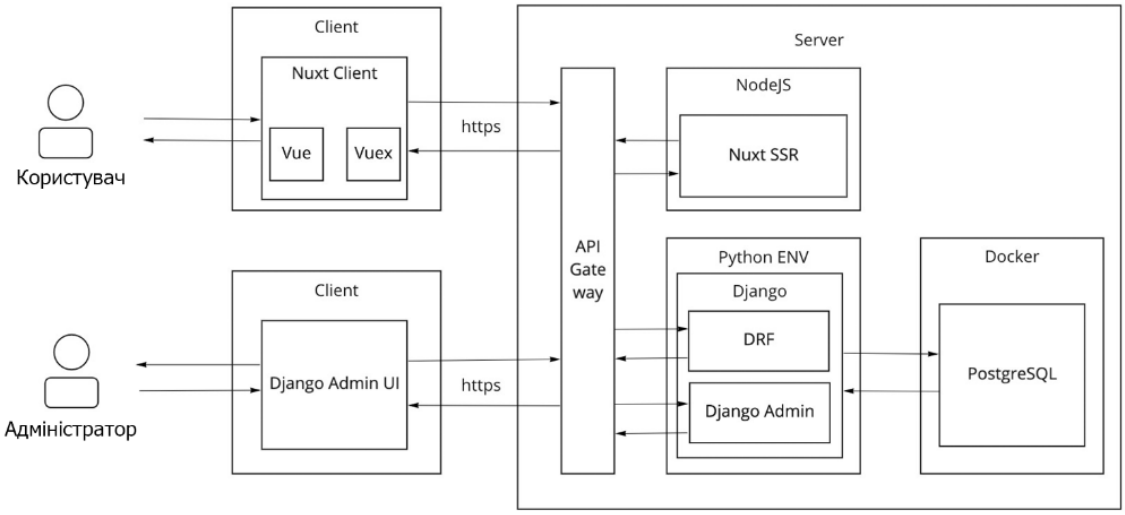

Рисунок 4.3 – Архітектура Model-View-Presenter

API сервера буде запущений на власному сервері і спілкуватися з клієнтом буде тільки через REST API-інтерфейс. У Django також є вбудована панель адміністратора, у якій є свій UI-інтерфейс, але до цієї панелі будуть допущені лише модератори та адміністратори.

Клієнтський інтерфейс має свою особливість – під час першого заходу на сайт спрацьовує SSR, тобто генерація першої сторінки відбувається на сервері NodeJS.

4.4 Налаштування API-сервера

Перед початком розробки бекенд-частини необхідно переконатися, що база даних PostgreSQL запущена в Docker. Це можна зробити через GUI Docker, як було показано в розділі 2.

Для початку розроблення проєкту на Django необхідно створити каталог проєкту, перейти в нього і відкрити в терміналі.

Щоб ізолювати проєкт Django від системного середовища, необхідно створити віртуальне оточення. Якщо цього не зробити, то не вийде розділяти версії пакетів у різних застосунках, створених на Python [16].

Віртуальне оточення створюється командою "python -m venv .venv". Щоб воно запрацювало, його необхідно активувати в системі командою

"source ./.venv/bin/activate".

Далі встановлюються необхідні пакети через пакетний менеджер мови Python - pip. Необхідно встановити такі пакети:

1) Django;

2) django-cors-headers – пакет для роботи з CORS заголовками http;

3) django-phonenumber-field – пакет, що додає поле телефону в Django ORM;

4) djangorestframework;

5) djangorestframework-simplejwt – пакет, що додає авторизацію за JWT-токену;

6) python-dotenv – пакет для роботи із системним оточенням.

Встановивши ці пакети командою "pip install ім'я_пакета", необхідно зафіксувати їхні версії, інакше можуть бути конфлікти пакетів у різних розробників. Фіксація пакетів проводиться командою

"pip freeze > requirements.txt". Пакети та їхні версії запишуться у файл requirements.txt (рис 4.4).

```
asgiref==3.5.0
autopep8==1.6.0
backports.zoneinfo==0.2.1
Django==4.0.4
django-cors-headers==3.11.0
django-phonenumber-field==6.1.0
djangorestframework==3.13.1
djangorestframework-jwt==1.11.0
djangorestframework-simplejwt==5.1.0
phonenumbers==8.12.46
Pillow==9.1.0
psycopg2-binary==2.9.3
pycodestyle==2.8.0
PyJWT==2.3.0
python-dotenv==0.20.0
pytz==2022.1
sqlparse==0.4.2
toml==0.10.2
```

Рисунок 4.4 – Файл requirements.txt

Надалі розробники під час інсталяції проєкту деінде використовуватимуть команду "pip install -r requirements.txt", яка встановлює всі пакети зазначених версій із файлу requirements.txt.

Щоб додати пакет у вже створений файл, достатньо встановити його командою "pip install ім'я_пакета", а потім знову скористатися командою "pip freeze > requirements.txt".

Встановлення та налаштування проєкту

Створюємо проєкт Django командою "django-admin startproject app". Ця команда автоматично генерує основні файли проєкту. Можна одразу згенерувати сервіси для роботи з користувачами і тренажерами командами "python manage.py startapp accounts" і "python manage.py startapp trainers".

Для локальної розробки в корені проєкту створюємо файл env і прописуємо такі налаштування:

- дані для підключення до бази даних;
- секретний ключ;
- прапор режиму налагодження.

Підсумковий варіант налаштувань у файлі оточення представлений на (рис. 4.5).

```
DEBUG=True

SECRET_KEY="lnm=_c!yj=1r9q-y2uee&ge(4ng6b5)1%)e)s3_t%_v1)9!vb$"

DBNAME="math_trainer"
DBUSER="math_trainer_root"
DBPASS="zCpzLRbcJi5aYTh"
DBHOST="localhost"
DBPORT=5432
```

Рисунок 4.5 – Файл налаштування оточення

Тепер можна налаштувати проєкт Django у файлі settings.py. Спершу необхідно зчитати налаштування зі змінних оточення. Для цього в settings.py імпортуємо стандартну бібліотеку os, функцію load_dotenv з пакета pythondotenv і функцію Path зі стандартної бібліотеки pathlib [17]. За допомогою функції Path отримуємо шлях до файлу змінних оточення і передаємо його у функцію env, яка встановить змінні оточення в систему. Щоб отримати змінні оточення, достатньо викликати функцію getenv з бібліотеки os, передавши в неї аргументом ключ змінної оточення (рис. 4.6).

```
import os

from dotenv import load_dotenv
from pathlib import Path

from django.utils.translation import gettext_lazy as _

BASE_DIR = Path(__file__).resolve().parent.parent

ENV_PATH = BASE_DIR / '.env'
load_dotenv(dotenv_path=ENV_PATH)


SECRET_KEY = os.getenv("SECRET_KEY")
```

Рисунок 4.6 – Отримання змінних оточення

Раніше було встановлено додаткові пакети і створено два Django додатки: accounts і trainers. Щоб вони запрацювали, їх необхідно додати в налаштування INSTALLED_APPS (рис. 4.7).

```
39  INSTALLED_APPS = [
40      'django.contrib.admin',
41      'django.contrib.auth',
42      'django.contrib.contenttypes',
43      'django.contrib.sessions',
44      'django.contrib.messages',
45      'django.contrib.staticfiles',
46
47      'corsheaders',
48      'phonenumber_field',
49      'rest_framework',
50      'rest_framework_simplejwt.token_blacklist',
51
52      'accounts',
53      'trainers',
54  ]       You, last month • Добавили MVP
```

Рисунок 4.7 – Підключення застосунків у Django

Далі необхідно підключити запущену в Docker-контейнері базу даних у налаштуванні DATABASES (рис 4.8).

```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.postgresql_psycopg2',
        'NAME': os.getenv("DBNAME"),
        'USER': os.getenv("DBUSER"),
        'PASSWORD': os.getenv("DBPASS"),
        'HOST': os.getenv("DBHOST"),
        'PORT': os.getenv("DBPORT"),
        'ATOMIC_REQUESTS': True,
    }
}
```

Рисунок 4.8 – Підключення бази данних

Для авторизації необхідно налаштувати Django REST framework і в налаштуванні класу авторизації за замовчуванням вказати клас із пакета djangorestframework-simplejwt [18; 19]. Також можна окремо налаштувати цей пакет.

У його налаштуваннях вказуються такі параметри:

– оновлення refresh-токена;
– додавання старих токенів до чорного списку після оновлення;
– оновлення часу останньої сесії користувача.

Ці налаштування підвищать безпеку використання тренажера, а також допоможуть відстежувати сесії користувача.

4.5 Розробка сервісу для роботи з користувачами

Раніше було створено і підключено у файлі налаштувань застосунок Django accounts. Тепер розглянемо докладніше його структуру:

1) Каталог migrations – до нього потрапляють усі міграції, пов'язані з моделями з файлу models.

2) Файл admin – у ньому відбувається підключення моделі з файлу models до панелі адміністратора Django.

3) Файл models – у ньому описуються моделі, які Django ORM перетворить на таблиці в базі даних.

4) Файл serializers – у ньому описуються класи, які приводять довільні об'єкти до моделей із файлу models, а також проводять їхні перевірки на відповідність.

5) Файл urls – у ньому зв'язуються шляхи, за якими клієнт звертатиметься до API, і подання з файлу views.

6) Файл views – у ньому описується бізнес-логіка.

7) Файл managers – файл, у якому перевизначається стандартний клас менеджера.

Для початку необхідно перенести розроблену архітектуру бази даних у моделі Django. Починаємо з моделі користувача (рис. 4.9).

```python
class Account(AbstractBaseUser):
    email = models.EmailField(
        max_length=60,
        unique=True,
        verbose_name=_('Email')
    )
    date_joined = models.DateTimeField(
        auto_now_add=True,
        verbose_name=_('Date joined')
    )
    last_login = models.DateTimeField(
        auto_now=True,
        verbose_name=_('Last login')
    )
    is_admin = models.BooleanField(default=False)
    is_active = models.BooleanField(default=True)
    is_staff = models.BooleanField(default=False)
    is_superuser = models.BooleanField(default=False)
    account_image = models.ImageField(
        max_length=255,
        upload_to=get_account_image_filepath,
        null=True,
        blank=True,
        default=get_default_account_image
    )

    objects = AccountManager()

    USERNAME_FIELD = 'email'
    )
```

Рисунок 4.9 – Модель користувача

Через те, що змінюється стандартний клас користувача в Django, необхідно переписати клас менеджера користувача (рис. 4.10). У ньому описуються функції створення звичайного користувача і користувача з найвищими правами доступу. Ця дія була необов'язковою, але в майбутньому планується додавання реєстрації через телефон, і описані функції сильно в цьому допоможуть.

```python
class AccountManager(BaseUserManager):
    def create_user(self, email, password=None):
        if not email:
            raise ValueError("Users must nave an email address.")
        user = self.model(
            email=self.normalize_email(email)
        )
        user.set_password(password)
        user.save(using=self._db)
        return user

    def create_superuser(self, email, password):
        user = self.create_user(
            email=email,
            password=password,
        )
        user.is_admin = True
        user.is_staff = True
        user.is_superuser = True
        user.save(using=self._db)

        return user
```

Рисунок 4.10 – Клас менеджера моделі користувача

Оскільки в наступних версіях передбачається профіль репетитора, то такі поля можна винести в абстрактний клас профілю:

– користувач;
– ім'я;
– прізвище;
– по батькові;
– стать.

Клас профіль учня успадковуватиметься від класу абстрактного класу користувача (рис. 4.11). У ньому додатково буде описано рік навчання.

```
113  class StudentProfile(AbstractBaseProfile):
         You, last month | 1 author (You)
114      class YearOfStudy(models.IntegerChoices):
115          FIRST = 1, _('First year')
116          SECOND = 2, _('Second year')
117          THIRD = 3, _('Third year')
118          FOURTH = 4, _('Fourth year')
119          FIFTH = 5, _('Fifth year')
120          SIXTH = 6, _('Sixth year')
121          SEVENTH = 7, _('Seventh year')
122          EIGHTH = 8, _('Eighth year')
123          NINTH = 9, _('Ninth year')
124          TENTH = 10, _('Tenth year')
125          ELEVENTH = 11, _('Eleventh year')
126
127      year_of_study = models.IntegerField(
128          null=True,
129          default=None,
130          choices=YearOfStudy.choices,
131          verbose_name=_('Year of study')
132      )
133
             You, last month | 1 author (You)
134      class Meta:
135          verbose_name = _('Student profile')
136          verbose_name_plural = _('Student profiles')
```

Рисунок 4.11 – Модель профілю учня

Інші моделі з розробленої архітектури БД будуть переноситися аналогічно.

У файлі views створюються подання для реєстрації користувача, отримання даних про користувача і завершення сесії користувача.

Ці подання підключаються до роутера Django у файлі urls (рис. 4.12). Також у цьому файлі підключаються подання на отримання токенів з бібліотеки djangorestframework-simplejwt [20].

```python
urlpatterns = [
    path('token', TokenObtainPairView.as_view()),
    path('token/refresh', TokenRefreshView.as_view()),

    path('logout', logout_view, name='logout'),

    path('sign-up', sign_up_view, name="sign-up"),

    path('user', get_user_view, name="get-user"),
]
```

Рисунок 4.12 – Підключення уявлень до роутера Django

4.6 Розробка сервісу для роботи з тренажерами

Відмінна особливістю розробки сервісу тренажерів будуть функції, що відповідають за генерацію завдань (рис. 4.13). Для прискорення процесу розробки функції генерації окремих завдань прив'язані до моделі тесту, але в майбутньому буде необхідно їх виділити в окремий пакет.

Рисунок 4.13 – Блок-схема функції генерації завдань

Функція генерації завдань приймає як аргументи контекст і попереднє завдання. Далі зі словника генератора завдань витягується потрібний генератор і викликається. Якщо генератор згенерував завдання, то в об'єкт цього завдання записується ідентифікатор попереднього завдання. Наприкінці функція повертає об'єкт завдання.

Розберемо одну функцію генерації завдання на прикладі додавання до 10 (рис 4.14). Ця функція приймає як аргумент тест, у контексті якого вона створює завдання. У майбутньому планується передача аргументів для більш тонкого налаштування генерації завдання.

```
365  def generate_add_within_ten_task(test):
366      a = randint(1, 9)
367      b = randint(1, 10 - a)
368
369      return Task(
370          test=test,
371          description=f'{a} + {b} =',
372          correct_answer=f'{a + b}'
373      )
```

Рисунок 4.14 – Приклад функції генерації завдання

Як можна побачити, спочатку генерується перший доданок з використання функції randint зі стандартного пакета random. Із заданими аргументами ця функція поверне випадкове число від 1 до 9. Далі генерується другий доданок у такий самий спосіб, але як другий аргумент у функцію random передається різниця 10 і першого доданка. Це необхідно для того, щоб сума першого і другого доданків не перевищила 10. Наприкінці створюється і повертається новий об'єкт завдання.

Ще однією відмінною особливістю тренажерів є створення залежностей між тренажерами, що дає змогу вибудовувати проходження тренажерів у певному порядку. Допоки не будуть пройдені тренажери нижнього рівня, від яких є залежності тренажерів наступного рівня, усі тренажери рівня вище попереднього не будуть доступні учневі.

У поточній реалізації було створено 6 тренажерів, які утворюють 4 рівні (рис 4.15).

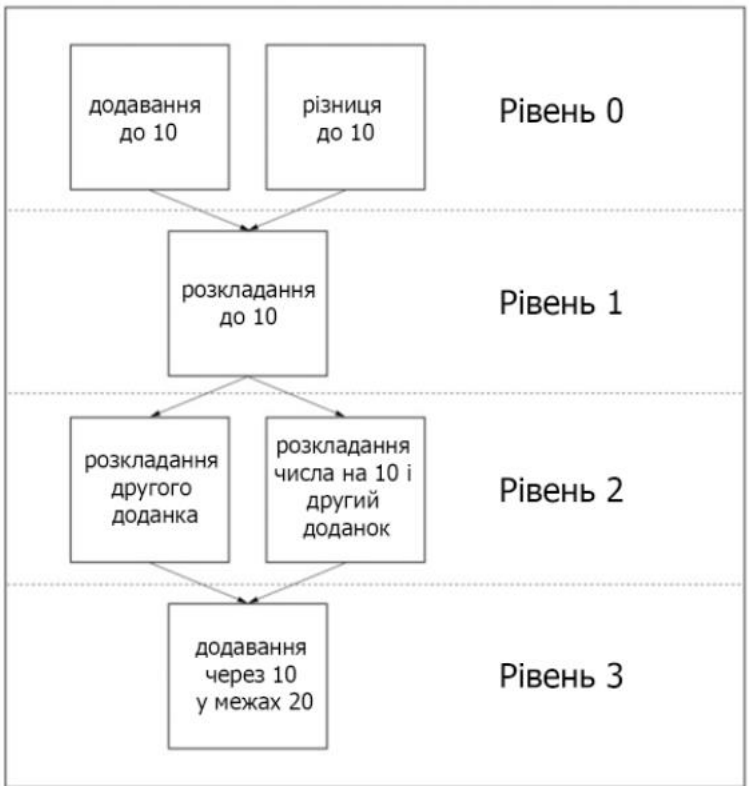

Рисунок 4.15 – Приклад рівнів тренажеру

На нульовому рівні тренажери "Додавання до 10" і "Різниця до 10" не мають залежностей, тож учень може почати відпрацьовувати на них навички одразу після реєстрації. Після їх опанування учневі відкриється тренажер першого рівня - "Розкладання 10". Опанувавши цю навичку, учень отримає доступ до тренажерів другого рівня: "Розкладання другого доданка" та "Розкладання числа на 10 і другий доданок". У фіналі учень дійде до цільової навички "Додавання через 10 у межах 20".

Оскільки рівні в цій реалізації виконані лише у вигляді абстракцій, то можливо створювати зв'язки між тренажерами не сусідніх рівнів. Наприклад, між нульовим і відразу третім.

4.7 Підключення панелі адміністратора

Для того щоб створювати первинні зв'язки між тренажерами та налаштовувати їхній опис у панелі адміністратора Django, необхідно під'єднати моделі застосунку у файлі admin (рис. 4.16).

```python
from django.contrib import admin

from trainers.models import Trainer

You, 2 months ago | 1 author (You)
class TrainerAdmin(admin.ModelAdmin):
    list_display = (
        'id',
        'title',
        'slug',
    )
    search_fields = ('id', 'title', 'slug')

admin.site.register(Trainer, TrainerAdmin)
```

Рисунок 4.16 – Підключення моделі в панелі адміністратора

У поточних налаштуваннях налаштовуються поля, які будуть показані в списку тренажерів, а також вказуються поля, за якими буде доступний пошук. Після заповнення інформації по кожному з тренажерів, список тренажерів в панелі адміністратора буде виглядати так, як показано на рис. 4.17.

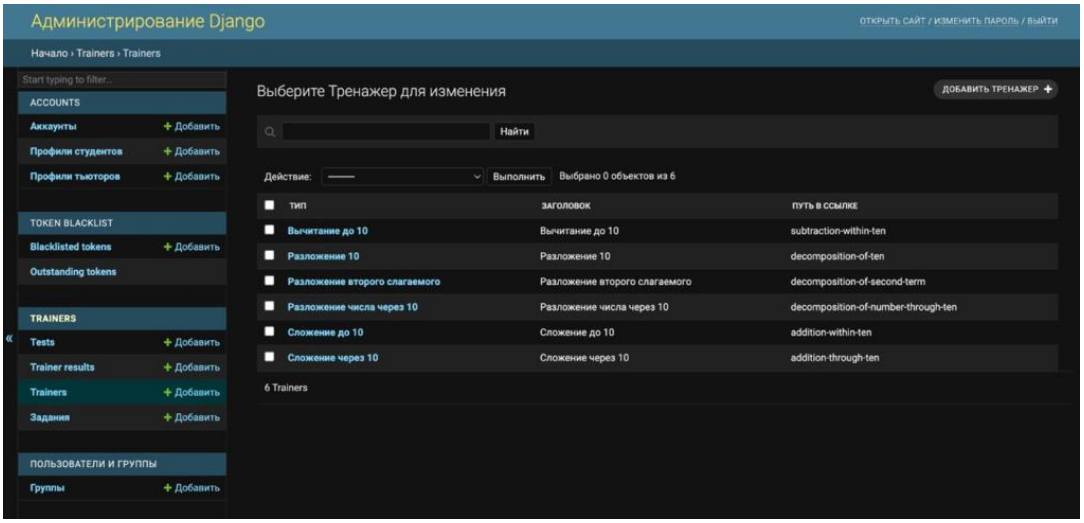

Рисунок 4.17 – Список тренажерів у панелі адміністратора

4.8 Результат розробки API-сервера та розробка вебінтерфейсу

4.8.1 Методи взаємодії з API-сервером

У результаті розробки API-сервера було отримано повністю самостійний сервіс, який не залежатиме від клієнта.

Методи взаємодії з API-сервером:

1) POST-метод реєстрації учня;
2) POST-метод авторизації через JWT-токен;
3) POST-метод оновлення токена;
4) POST-метод завершення сесії;
5) GET-метод отримання списку тренажерів;
6) GET-метод отримання тренажера;
7) GET-метод створення та отримання тесту;
8) GET-метод отримання тесту за ідентифікатором;
9) GET-метод отримання списку пройдених тестів;
10) GET-метод генерації наступного завдання або завершення тесту, якщо минув час на рішення;
11) POST-метод відповіді на завдання;
12) GET-метод отримання списку завдань за ідентифікатором тесту.

API-сервер має REST-інтерфейс для взаємодії з клієнтами, який зможе працювати як з вебінтерфейсом, так і з мобільними клієнтами.

4.8.2 Попередні налаштування та підключення пакетів

Для початку розробки проєкту на Nuxt необхідно створити каталог проєкту, перейти до нього і відкрити в терміналі. Далі проєкт створюється командою "npx createnuxt-app". Під час створення проєкту установник запропонує кілька налаштувань, але залишаємо все за замовчуванням.

Після створення проєкту необхідно встановити кілька пакетів, які надалі будуть використовуватися в розробці:

а) "@nuxtjs/axios" – пакет для зручного надсилання запитів на сервер;

б) "@nuxtjs/auth-next" – пакет для роботи з авторизацією через JWTтокен;

в) "@nuxtjs/proxy" – пакет для переадресації запитів на зазначені в налаштуваннях ресурс

г) "@nuxtjs/fontawesome" – пакет для роботи з іконками з ресурсу Fontawesome;

д) "bem-cn-lite" – пакет для формування назви класів за методологією БЕМ.

Усі пакети встановлюються через пакетний менеджер npm командою "npm і ім'я_пакета". Встановлені пакети та їхні версії відображаються у файлі package.json у полі dependencies.

Далі необхідно налаштувати проєкт у файлі nuxt.config.js. Насамперед підключимо встановлені пакети до розроблюваного проєкту (рис. 4.18).

```
27    // Modules for dev and build (recommended): https://go.nuxtjs.dev/config-modules
28    buildModules: [
29      '@nuxtjs/fontawesome',
30    ],
31
32    // Modules: https://go.nuxtjs.dev/config-modules
33    modules: [
34      '@nuxtjs/axios',
35      '@nuxtjs/auth-next',
36    ],
```

Рисунок 4.18 – Підключення пакетів до проєкту

Згідно з документацією, пакети "@nuxtjs/axios" і "@nuxtjs/auth-next" підключаються до модулів, оскільки вони не потребують додаткового збирання. Пакет "@nuxtjs/fontawesome" під'єднують через buildModules, оскільки в ньому можна налаштувати повноцінне збирання, що полегшить кінцевий зібраний файл.

Далі відбувається налаштування надсилання запиту через axios з використанням proxy (рис. 4.19). У налаштуванні axios вмикається proxy, а потім уже в налаштуванні proxy описуються параметри переадресації запиту. Коли на клієнті надсилатиметься запит без повної вказівки домену через слеш, браузер передусім звертатиметься за доменом клієнта, але proxy змінить його на домен сервера, а також відредагує шлях, якщо це необхідно. Це налаштування дає змогу тримати всі API в одному місці й не писати щоразу повний шлях із доменом.

```
    axios: {
        proxy: true,
    },

    proxy: {
        '/api/': {
            target: 'http://localhost:8000/',
            pathRewrite: {
                '^/api/': '',
            },
        },
    },
```

Рисунок 4.19 – Налаштування відправки запитів на сервер

Наприкінці необхідно налаштувати модуль авторизації (рис. 4.20). Насамперед підключається проміжний обробник запитів у полі router. Цей обробник в автоматичному режимі буде перенаправляти користувача на зазначену в налаштуваннях сторінку, якщо він спробує зайти на сторінку, яка доступна тільки авторизованим користувачам.

```
  router: {
      middleware: ['auth'],
  },

  auth: {
      redirect: {
          login: '/sign-in',
          logout: '/',
          // eslint-disable-next-line id-blacklist
          callback: '/sign-in',
          home: '/',
      },

      strategies: {
          local: {
              scheme: 'refresh',
              token: {
                  property: 'access',
                  maxAge: 60 * 5,
                  global: true,
                  type: 'Bearer',
              },
              refreshToken: {
                  property: 'refresh',
                  data: 'refresh',
                  maxAge: 60 * 60 * 24,
                  tokenRequired: true,
              },
              user: {
                  property: 'user',
              },
              endpoints: {
                  login: { url: '/api/accounts/token', method: 'post' },
                  refresh: { url: '/api/accounts/token/refresh', method: 'post' },
                  user: { url: '/api/accounts/user', method: 'get' },
                  logout: { url: '/api/accounts/logout', method: 'post' },
              },
          },
      },
  },
```

Рисунок 4.20 – Налаштування модуля авторизації

Далі в полі auth проводиться налаштування самого модуля авторизації. У полі redirect вказуються налаштування переадресацій. У полі strategies можна описати всі можливі варіанти авторизації. Сервер працює через JWT-авторизацію, тому було встановлено налаштування для неї. Найключовіше налаштування тут – це налаштування шляхів запитів на сервер.

4.8.3 Налаштування роутингу

Nuxt автоматично налаштовує роутинг, достатньо створити файли і каталоги в певній структурі в каталозі pages. У каталозі pages було створено файл index.vue – він відповідатиме за відображення головної сторінки. Файли signup.vue і sign-in.vue – це сторінки авторизації та реєстрації. У каталозі trenazhery також є файл index.vue – у ньому відображатиметься список усіх тренажерів. Перехід на сторінку тренажера відбувається через динамічний шлях, тому необхідно назвати каталог slug з нижнього пробілу, щоб була можливість отримати цей шлях у вигляді змінної та використати його у надсиланні запиту на сервер у файлі index.vue каталогу _slug. Останньою сторінкою буде сторінка самого тесту, на яку можна потрапити за його ідентифікатором. У результаті виходить структура сайту, представлену на рис. 4.21.

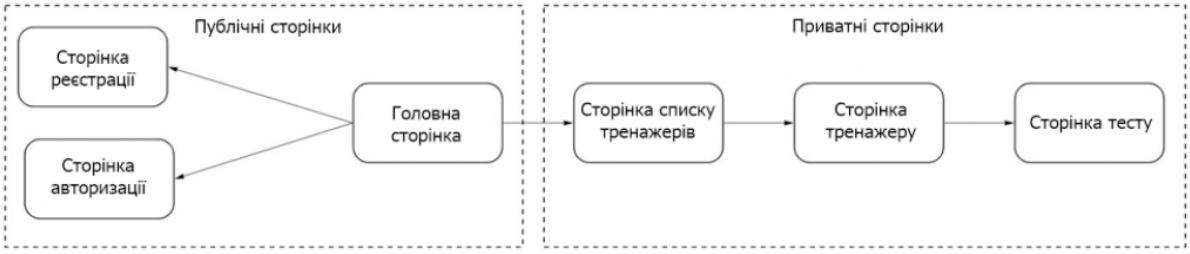

Рисунок 4.21 – Структура сайту

4.8.4 Створення компонентів

Розробка клієнтської частини програми відбувається за компонентним підходом. Кожен компонент є незалежною сутністю і може бути перевикористаний безліч разів. Усі інкрементальні компоненти зберігаються в каталозі components.

Розберемо створення компонентів на прикладі компонента таймера. Для цього в каталозі components створимо підкаталог timer. У ньому основним файлом буде index.vue. Весь HTML, CSS і JS код можна писати в цьому файлі, але тоді файл компонента буде занадто великим і таким, що не читається, тому була проведена його розбивка на кілька файлів:

1) у файлі index.vue міститиметься верстка, а також підключення скриптів і стилів;

2) у файлі scripts.js буде прописана вся логіка роботи компонента;

3) файл styles.scss відповідає за стилізацію компонента. Цей файл імпортує в себе name і vars.

Повну структуру компонента можна побачити на рис. 4.22.

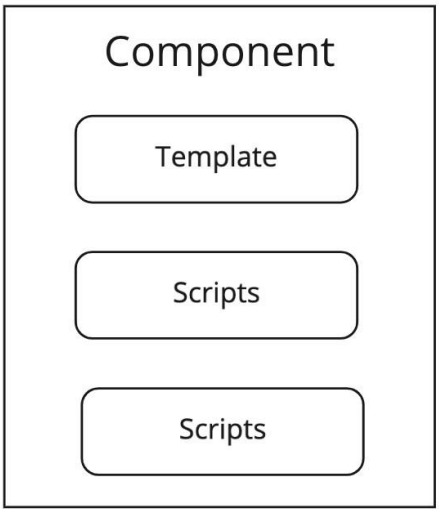

Рисунок 4.22 – Структура компоненту

За такою схемою створюються компоненти, що залишилися:

- кнопка;
- посилання;
- поле введення;
- контейнер;
- іконка;
- картинка;
- поточний рахунок користувача;

– секундомір;
– таймер.

4.8.5 Створення модулів

Компоненти модулів не відрізняються структурою від звичайних компонентів. Головною їхньою відмінною рисою є наявність бізнес-логіки. Вони можуть запитувати дані з АПІ або з менеджера станів Vuex.

Головним модулем є модуль app. Він відповідає за підключення загальних утиліт застосунку, таких як визначення фокусу, визначення наведення, гортання вікна або зміна розмірів вікна. Також у цьому модулі підключаються модуль шапки сайту і модуль підвалу.

Особливу увагу варто приділити модулю тренажерів. Цей модуль містить у собі такі компоненти:

– завдання;
– результати відповіді на завдання;
– результати тесту з тренажера;
– список пройдених тестів за тренажером;
– список тренажерів.

4.8.6 Створення загального сховища даних

За загальне зберігання даних відповідає менеджер станів Vuex [22]. Спершу в необхідно в корені проєкту створити каталог store. Це стандартний каталог сховища в Nuxt [23]. Далі створюється в store ще 3 підкаталоги:

1) app – загальне сховище, що заповнюється через модуль app;
2) trainers – сховище тренажерів;
3) user – сховище даних про користувача.

У результаті виходить структура сховища даних на клієнті, представлена на рис. 4.23.

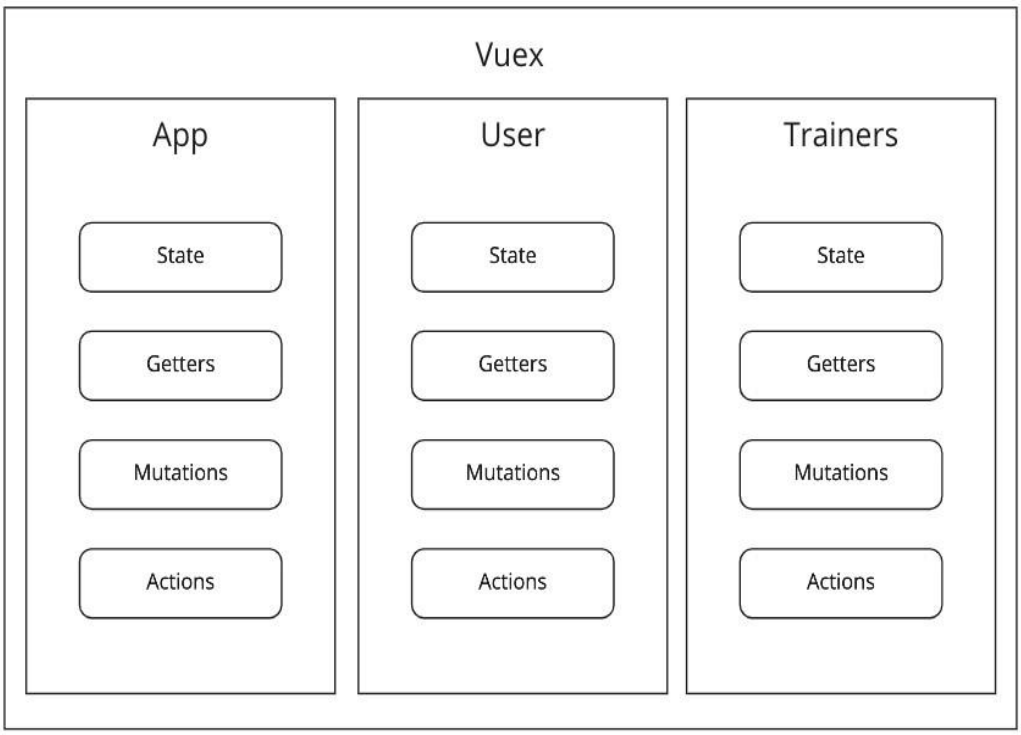

Рисунок 4.23 – Структура сховища даних

4.8.7 Результат розробки вебінтерфейсу

Зібравши всі частини воєдино, було отримано вебінтерфейс, який містить 6 сторінок:

- головна сторінка;
- авторизація;
- реєстрація;
- список тренажерів;
- сторінка тренажера (рис 4.24);
- сторінка тесту.

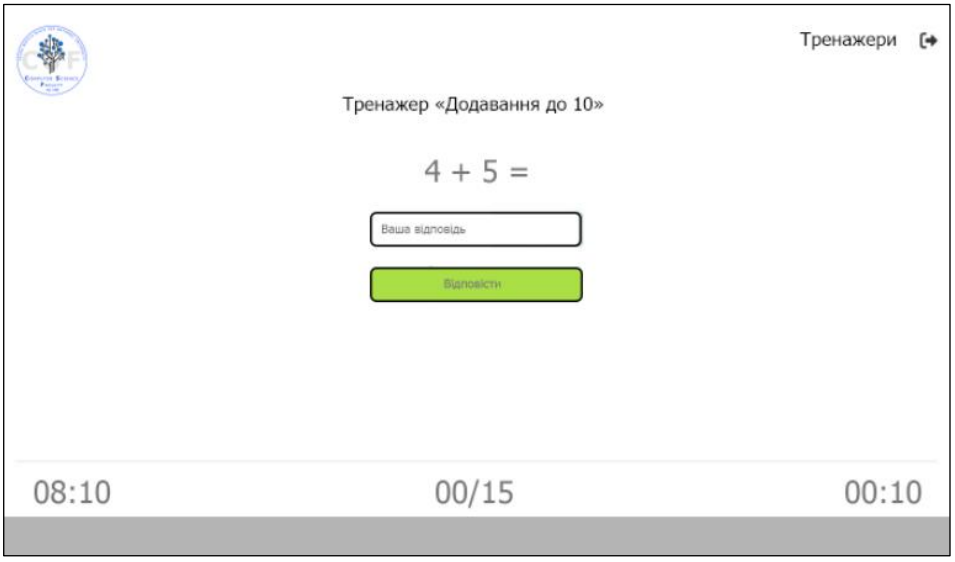

Рисунок 4.24 – Сторінка тренажера

Висновки до розділу 4

У цьому розділі було описано архітектуру онлайн-сервісу для формування навичок розв'язання математичних задач для школярів. Були створені окремі моделі для учня, тренажера, конфігурації тренажера, результатів тренажера та тесту. Для роботи з користувачами та тренажерами створені сервіси accounts та trainers відповідно. Для зберігання даних використовується PostgreSQL з Docker та Django ORM. Для авторизації використовується JWT-токен. Розроблено алгоритм роботи тренажера, який підходить для тренажерів з часовим та кількісним обмеженням. Проведено розробку онлайн-тренажера: від прототипів у Wireframe до повноцінного застосунку, що складається з бази даних зі складною архітектурою, API-сервера на Django і вебінтерфейсу.

ВИСНОВКИ

Рішення, спрямовані на надання освітніх послуг школярам, сильно зав'язуються на традиційній шкільній моделі навчання, тому не можуть підвищити загальну якість освіти в Україні, а головне, що вони не розв'язують проблем, пов'язаних з опануванням учнів елементарних навичок.

Навички розв'язування елементарних математичних задач впливають не тільки на опанування математики та алгебри, вони також дуже впливають на такі предмети як фізика, хімія, інформатика, статистика та економіка.

Саме тому тема цієї роботи присвячена створенню хмарного рішення для відпрацювання математичних навичок.

У першому розділі актуальність теми підтвердилася аналізом результатів опитувань, проведених серед батьків школярів. Ці опитування показали високу зацікавленість у подібному рішенні. Також було розглянуто наявні рішення, які намагаються вирішувати цю проблему. Завдяки їм було визначено всі функціональні особливості розроблюваної ІС. У фінальній частині першого розділу було коротко розібрано методології розроблення цих систем, які впливають на їхній життєвий цикл.

У другому розділі було проведено підготовку до розроблення ПЗ математичного тренажера:

1) розроблено концепцію тренажера і вимоги до нього;
2) розроблено вибір технологій для розробки;
3) розроблено дизайн тренажера;
4) було розглянуто етапи життєвого циклу ПЗ;
5) проведено прототипування та розроблення дизайну;
6) було проведено налаштування засобів оточення, IDE і допоміжних утиліт;
7) було розроблено основний алгоритм роботи тренажера та архітектуру бази даних;

8) було послідовно розроблено API-сервер і вебінтерфейс з використанням новітніх технічних рішень;

Для створення онлайн-системи тренажерів з метою покращення математичних навичок у школярів, було виконано низку завдань. Спочатку проведено аналіз ринку та порівняно вже існуючі рішення в цій сфері. Розглянуто різні методології розробки інформаційних систем. Розроблено концепцію та вимоги до системи, обрано відповідні інструменти та технології для її розробки. Розроблено архітектуру та алгоритми роботи системи, а також створено мінімальний виробничий продукт цього ПЗ для школярів. Діяльність MVP включає реєстрацію, автентифікацію та навчальну діяльність з відпрацювання математичних навичок. Цей застосунок стане основою для подальшого розширення функціональності та розвитку інформаційної системи.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Кисса О. П., Журавська І. М. Програмне забезпечення на базі фреймворку Django для формування навичок розв'язання математичних задач школярами. *Інформаційні технології та інженерія* : тези доп. Всеукр. наук.-практ. конф. Миколаїв, 31 січ. – 02 лют. 2024 р. Миколаїв : Чорном. нац. ун-т ім. Петра Могили, 2024. С. 134–136.

2. Статистичні дані 2023. URL: https://testportal.gov.ua/statystychni-dani/ (дата звернення: 20.12.2023).

3. Чисельність населення України за статтю і віком. URL: https://index.minfin.com.ua/ua/reference/people/ (дата звернення: 20.12.2023).

4. Найкращі сайти для вивчення математики онлайн. URL: https://www.mathema.me/blog/sayty-dlya-vyvchennya-matematyky/ (дата звернення: 20.12.2023).

5. Wireframe.cc – the go-to free, online wireframing tool. URL: https://wireframe.cc/ (Last accessed: 30.11.2023).

6. Learning.ua. URL: https://learning.ua (дата звернення: 30.11.2023).

7. OnlineMSchool. URL: https://ua.onlinemschool.com (дата звернення: 30.11.2023).

8. Matific | Ігри та тести з математики онлайн. URL: https://www.matific.com//home/ (дата звернення: 30.11.2023).

9. Освітній проект «На Урок» для вчителів. URL: https://naurok.com.ua/ (дата звернення: 30.11.2023).

10. 12 кращих методологій розробки програмного забезпечення з перевагами та недоліками. https://www.smart-it.com/uk/2021/08/12-best-software-development-methodologies-with-pros-and-cons/ (дата звернення: 24.12.2023).

11. Auffarth B. Machine learning for time-series with Python. Packt Publ., 2021. 370 p.

12. Pallant J., Danaher P. J., Sands S., Tracey S. Danaher T. S. An empirical analysis of factors that influence retail website visit types. Journal of Retailing and Consumer Services. Nov. 2017. Vol. 39. P. 62–70. DOI: 10.1016/j.jretconser.2017.07.003.

13. Awaid M. S., Muqaibal H. O., Huzaili T. A., Farooque M. M. Analysis of E-Commerce websites GCC using data science approach. Sept. 2023. DOI: 10.46610/JBADV.2023.v04i02.002.

14. Siavashi A., Momtazpour M.Cloudy: A Python-based simulator for modern cloud environments. Nov. 2023. DOI: 10.21203/rs.3.rs-3665148/v1.

15. Документація Django 5.0. URL: https://django.fun/docs/django/5.0/ref/ (дата звернення: 11.12.2023).

16. Adil M. K. Methodology for solving a number of problems in Python and C++ languages with the application of mathematical knowledge. The American Journal of Interdisciplinary Innovations and Research. Oct. 2023. Vol. 5(10). P. 19–23. DOI: 10.37547/tajiir/Volume05Issue10-04.

17. Python. URL: https://www.python.org/ (Last accessed: 05.01.2024).

18. Django – The web framework for perfectionists with deadlines. URL: https://www.djangoproject.com/ (Last accessed: 06.01.2024).

19. Dinder M. Becoming an enterprise Django developer. Packt Publ., 2022. 526 p.

20. Django REST framework. URL: https://www.django-rest-framework.org/ (Last accessed: 06.01.2024).

21. Vue.js – The Progressive JavaScript Framework. URL: https://vuejs.org/ (Last accessed: 07.01.2024).

22. What is Vuex?. URL: https://vuex.vuejs.org/ (Last accessed: 05.01.2024).

23. Nuxt.js – The Intuitive Vue Framework. https://nuxtjs.org/ (Last accessed: 05.01.2024).

24. ДСТУ 3582:2013. Бібліографічний опис. скорочення слів і словосполучень українською мовою. Загальні вимоги та правила [Чинний від 2014–01–01]. Київ : Мінекономрозвитку України, 2014. 15 с. (Інформація та документація). URL: http://lib.pnu.edu.ua/files/DSTU_3582_2013%20Скорочення%20слів%20українською%20мовою.pdf (дата звернення: 15.02.2024).

25. ДСТУ 3008:2015. Звіти у сфері науки і техніки. Структура та правила оформлювання [Чинний від 2017–07–01]. Київ : УкрНДНЦ України, 2016. 26 с. (Інформація та документація). URL: https://knmu.edu.ua/wp-content/uploads/2021/06/dstu3008-2015_metr.pdf (дата звернення: 16.02.2024).

ДОДАТОК А

Матеріали апробації роботи

Б.1 Всеукраїнська науково-практична конференція «Інформаційні технології та інженерія»

Міністерство освіти і науки України
Чорноморський національний університет
імені Петра Могили

PETRO MOHYLA BLACK SEA NATIONAL UNIVERSITY
COMPUTER SCIENCE FACULTY
Est. 1996

«Інформаційні технології та інженерія»

Всеукраїнська науково-практична конференція молодих вчених, аспірантів і студентів

ТЕЗИ

31 січня – 2 лютого 2024 року

Миколаїв – 2024

ЗМІСТ

Інформаційні системи та їх інтелектуалізація

Машинне навчання та штучний інтелект

2024 р.
Кисса О. П.
121–КРМ– 608м.21810915

Інтеграція штучного інтелекту відкриває нові перспективи у створенні та аналізі тестів. Використання алгоритмів ШІ дозволяє автоматизувати процес генерації питань, адаптувати тести до індивідуальних особливостей кожного студента, та навіть прогнозувати їх можливі результати. Це сприяє створенню більш ефективних та персоналізованих освітніх програм.

У доповіді підкреслено значення інноваційних підходів у вдосконаленні систем автоматизованого оцінювання знань. Розробка адаптивних алгоритмів, інтеграція зі штучним інтелектом, та використання клієнт-серверної архітектури відкривають шлях до створення більш ефективних, гнучких та індивідуалізованих навчальних інструментів, які здатні задовольнити вимоги сучасного освітнього середовища.

УДК 004.4'2:373.5

Кисса О. П., Журавська І. М.
Чорноморський національний університет ім. Петра Могили, м. Миколаїв, Україна

ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ НА БАЗІ ФРЕЙМВОРКУ DJANGO ДЛЯ ФОРМУВАННЯ НАВИЧОК РОЗВ'ЯЗАННЯ МАТЕМАТИЧНИХ ЗАДАЧ ШКОЛЯРАМИ

Пандемія COVID-19 принесла до нашого світу великі зміни, особливо це торкнулося сфери освіти. Школам терміново довелося переходити на дистанційний формат навчання. Старі методи навчання погано працювали при проведенні дистанційних занять, і багато вчителів не змогли адаптуватися до нової цифрової реальності та перейшли до сектору додаткової освіти. Через зменшення кількості вчителів зменшився час, який приділяється кожному учневі, що вдарило насамперед по практичним навичкам учнів. Це позначилося на результатах національного мультипредметного тесту (НМТ) – так, середній бал з математики на НМТ-2022 був 148,1, то на НМТ-2023 лише 135,4 [1].

Повертаючись до проблеми учнів, хотілося б відзначити, що вони часто роблять помилки в елементарних математичних операціях, навичка вирішення яких має сформуватися на практичних заняттях у школі. Ця проблема була і до пандемії, але перехід на дистанційне навчання лише посилив її. Для формування навички необхідно багаторазово повторювати певні дії. Наприклад, щоб сформувати математичну навичку, вчителю необхідно придумати або знайти

відповідне завдання, яке учень вирішить, а потім вчителю необхідно перевірити це завдання та оцінити, наскільки учень освоїв навичку, що відпрацьовується [2]. Якщо учень не зміг виконати завдання коректно, то доведеться повторити ті самі дії. Така робота складається з рутинних операцій, які можна автоматизувати, створивши інформаційну систему (ІС), що буде сама генеруватиме завдання та перевірятиме правильність їх рішень.

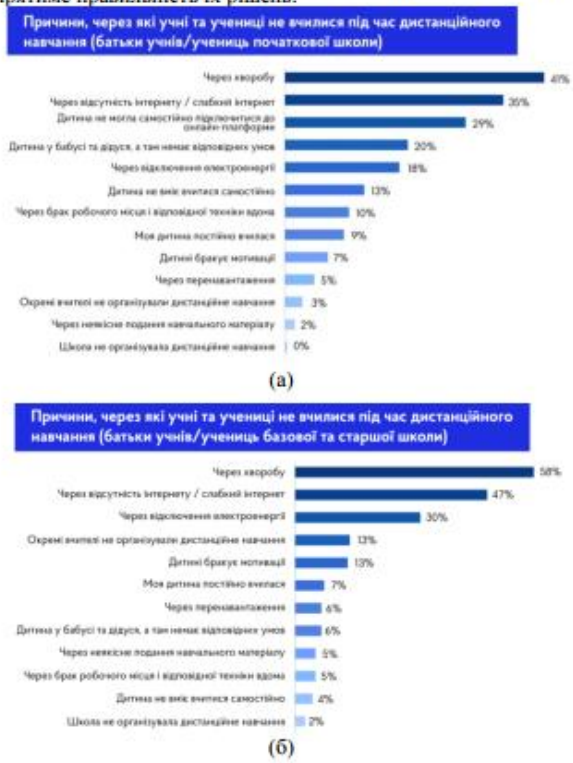

Рисунок 1 – Причини зниження навченості учнів початкової (а) та базової/старшої (б) школи [3]

2024 р. Кисса О. П. 121–КРМ– 608м.21810915

Актуальність розробки подібної ІС обумовлена проблемами школярів у вирішенні елементарних математичних завдань, проблемами дистанційної освіти, постійним зростанням ринку освітніх онлайн-послуг; а також стратегічними цілями України з цифрової трансформації.

Серед факторів, які негативно впливають на результати навчання учнів в умовах дистанційного навчання, вчителі передусім називають відсутність «живого спілкування»: 73 % вчителів про це відзначили у містах та 78 % у селах – для початкової школи (рис. 1, а), та 61 % та 57 % відповідно – для базової та старшої школи (рис. 1, б).

У ході роботи над даним проєктом доцільно використати такі інструменти:

- Visual Studio Code;
- Docker;
- мова програмування Python;
- мова програмування Javascript;
- HTML;
- SCSS.

Практична значимість роботи полягає у створенні ІС на основі онлайн-тренажера, який автоматизує процес відпрацювання математичних навичок у школярів. Цю ІС можуть використовувати батьки школярів для відстеження прогресу навчання, а також індивідуальні репетитори або школи для автоматизації видачі домашніх та практичних завдань. Інформаційно-емпіричною базою цього дослідження стали загальноприйняті методології розробки ІС, алгоритмів та архітектур ІС, а також результати дослідження діяльності існуючих ІТ-компаній у сфері розробки освітніх послуг.

Розробка MVP онлайн-тренажера з математики надасть можливість пришвидшити формування навичок у школярів як базової, так і старшої школи розв'язання математичних задач та підвищить якість їх підготовки до НМТ.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Opendata. Статистичні дані НМТ/основної сесії ЗНО. Український центр оцінювання якості освіти. URL: https://zno.testportal.com.ua/opendata (дата звернення: 27.08.2023).
2. Якобюк Л. І. Вивчення впливу дистанційної форми навчання під час пандемії на результати навчання. *Світ науки, культури, освіти*. 2020. № 5. С. 179-181.
3. Статистичні дані 2023. URL: https://testportal.gov.ua/statystychni-dani/ (дата звернення: 20.08.2023).

136