Міністерство освіти і науки України
Криворізький національний університет
Кафедра моделювання та програмного забезпечення

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття ступеня вищої освіти бакалавра

зі спеціальності 121 «Інженерія програмного забезпечення»

На тему: *Розробка програмного забезпечення для створення тематичних кросвордів*

Засвідчую, що в цій кваліфікаційній роботі немає запозичень із праць інших авторів без відповідних посилань.
Студент гр. ІПЗ-21-2
_____ *Буряк А. Б.*

Керівник кваліфікаційної роботи _____ */ Стрюк А. М. /*

Завідувач кафедри _____ */ Стрюк А. М. /*

Кривий Ріг
2025

Криворізький національний університет
Факультет: *Інформаційних технологій*
Кафедра: *Моделювання та програмного забезпечення*
Освітньо-кваліфікаційний рівень: *бакалавр*
Спеціальність: *121 "Інженерія програмного забезпечення"*

ЗАТВЕРДЖУЮ
Зав. кафедри
_____*Стрюк А. М.*
«___»_____ 2025__ р.

ЗАВДАННЯ
на кваліфікаційну роботу
студенту групи ІПЗ-21-2 Буряку Анатолію Борисовичу

1. Тема: *Розробка програмного забезпечення для створення тематичних кросвордів* затверджено наказом по КНУ №119 від «10» квітня 2025 р.
2. Термін подання студентом закінченого проекту «20» червня 2025 р.
3. Вихідні дані по роботі: *Пояснювальна записка: 102 сторінки, 19 рисунків, 1 додаток, 14 використаних у роботі джерел*
4. Зміст пояснювальної записки (перелік питань, що їх треба розробити): актуальність. Проектування програмного забезпечення. Розробка програмного забезпечення. Тестування.
5. Перелік графічного демонстраційного матеріалу: Приклади існуючих програм. Діаграма потоків даних. Діаграми використання. Блок-схеми основних алгоритмів. Структура бази даних. Приклади роботи розробленої програми.

Календарний план:

№	Найменування етапів кваліфікаційної роботи	Термін виконання етапів роботи
1	*Формулювання мети та задач роботи*	*13.02.2025-20.02.2025*
2	*Аналіз інформаційних джерел*	*21.02.2025-09.03.2025*
3	*Визначення вимог до програмного забезпечення*	*10.03.2025-18.03.2025*
4	*Розробка функціональної схеми*	*19.03.2025-23.03.2025*
5	*Розробка алгоритмів*	*24.03.2025-31.03.2025*
6	*Розробка бази даних*	*01.04.2025-14.04.2025*
7	*Розробка програмного забезпечення*	*15.04.2025-13.05.2025*
8	*Тестування програмного забезпечення*	*14.05.2025-20.05.2025*
9	*Оформлення пояснювальної записки*	*21.05.2025-12.06.2025*
10	*Розробка демонстраційних матеріалів*	*13.06.2025-17.06.2025*

Дата видачі завдання: «_____»_____ 2025 р.
Студент _____ Буряк А. Б.
Керівник роботи _____ Стрюк А. М.

РЕФЕРАТ

ГЕНЕРАТОР КРОСВОРДІВ, ТЕМАТИЧНИЙ КРОСВОРД, REACT, JAVASCRIPT, FIREBASE

Кваліфікаційна робота містить 102 сторінки, 19 рисунків, 14 джерел, 1 додаток.

Метою кваліфікаційної роботи є створення спеціалізованого програмного забезпечення, що надає можливість створювати тематичні кросворди за наборами слів, що задає користувач.

Визначено актуальність розробки програмного забезпечення для створення тематичних кросвордів. Розгялнуто кросворди як корисне дозвілля та засіб навчання. Виконано аналіз існуючих засобів генерації кросвордів.

Було визначено вимоги до програмного забезпечення для створення тематичних кросвордів. Спроєктовано потоків даних, діаграми використання, основні алгоритми, структуру бази даних

Програмне забезпечення реалізовано за допомогою React та мови JavaScript. Модульна структура надає можливість гнучко розвивати програму, вбудовувати її в інші веб-застосунки.

В створеній програмі є можливість задавати теми, формувати словники до кожної теми, генерувати на їх основі тематичні кросворди, зберігати їх та імпортувати в pdf для друку.

ABSTRACT

CROSSWORD GENERATOR, THEMATIC CROSSWORD, REACT, JAVASCRIPT, FIREBASE

The qualification work consists of 102 pages, 19 figures, 14 sources, and 1 appendix.

The purpose of the qualification work is to create specialized software that allows users to create thematic crosswords based on sets of words provided by the user.

The relevance of developing software for creating thematic crosswords has been established. Crosswords have been examined as useful leisure activities and educational tools. An analysis of existing crossword generation tools has been conducted.

The requirements for software designed to create thematic crosswords have been outlined. Data flow diagrams, usage diagrams, main algorithms, and database structures have been designed.

The software has been implemented using React and JavaScript. Its modular structure enables flexible program development and integration into other web applications.

The developed program offers the ability to set themes, create dictionaries for each theme, generate thematic crosswords based on them, save them, and export them to PDF for printing."

ЗМІСТ

ВСТУП

Кросворди є чудовим засобом не лише розважитись, але й навчитись чомусь новому. Вони сприяють розвитку логічного мислення, тренують пам’ять та допомагають розширити словниковий запас. Завдяки своїй універсальності кросворди активно використовують в різних навчальних ситуаціях: на уроках для перевірки засвоєного матеріалу, у позакласній діяльності для розвитку інтересу до певної теми чи навіть у мовних курсах для вивчення нових слів іншими мовами. Крім того, вони чудово розвивають увагу до деталей, оскільки людині потрібно порівнювати вже наявні літери з запропонованими ключами. У навчальних закладах кросворди допомагають перетворити процес запам’ятовування на веселу та захопливу гру, а для дорослих це ще й можливість тримати розум у тонусі, запобігаючи його пасивності.

Розвиток комп’ютерних засобів значно розширив можливості в різних сферах творчості та інтелектуальної діяльності, зокрема у створенні кросвордів. Завдяки сучасним програмним рішенням, генерація кросвордів стала набагато швидшою та ефективнішою. Спеціалізовані програми дозволяють автоматично підбирати слова, перевіряти їх перетин, оптимізувати розташування літер, а також створювати різні рівні складності завдань. Крім того, програмне забезпечення часто включає бази даних із тисячами термінів і тематичних слів, що дає змогу створювати комбінації відповідно до заданих тем або інтересів користувача. Сучасні інструменти також пропонують інтерактивні кросворди, які користувачі можуть розв’язувати безпосередньо на екранах своїх пристроїв, що відкриває нові можливості для навчання, розваги та тренування мислення. Отже, технології значно спростили цей процес, зробивши його доступним навіть для тих, хто раніше не володів подібними навичками.

Метою нашої роботи є створення спеціалізованого програмного забезпечення, що надає можливість створювати тематичні кросворди за наборами слів, що задає користувач.

Для досягнення мети нам необхідно виконати наступні задачі:

– визначити актуальність розробки програмного забезпечення для створення тематичних кросвордів

– дослідити феномен кросвордів як засобу навчання

– виконати аналіз існуючих засобів генерації кросвордів

– визначити вимоги до програмного забезпечення для створення тематичних кросвордів

– спроектувати програмне забезпечення для створення тематичних кросвордів

– спроєктувати потоки даних

– спроєктувати діаграми використання

– спроєктуванти основні алгоритми

– реалізувати програмне забезпечення для створення тематичних кросвордів

– добрати засоби розробки програмного забезпечення для створення тематичних кросвордів

– реалізувати основні компоненти

– протестувати програмне забезпечення.

1 АКТУАЛЬНІСТЬ РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ СТВОРЕННЯ ТЕМАТИЧНИХ КРОСВОРДІВ

1.1 Кросворди як корисне дозвілля та засіб навчання

Кросворди вже давно вважаються найулюбленішою та найпопулярнішою грою зі словами у всьому світі, що підкреслює їхню особливу чарівність і місце у нашому житті. Часто здається, ніби вони були з нами століттями, залишаючи глибокий слід у нашій культурі. Проте, сучасна форма кросвордів з'явилася не так давно, як можна подумати. Як гра, що відносно молода, змогла завоювати серця мільйонів і стати невід'ємною частиною культури? Це говорить про те, наскільки швидко і гармонійно вони вплелися в наше життя.

Швидкість, із якою кросворди стали «вкоріненими», показує не просто швидкоплинну примху, а їхню глибоку, потужну привабливість і, мабуть, суспільну потребу, яку вони вдало задовольняли. Це свідчить про те, що їхня популярність не була поверхневою, а відповідала фундаментальним людським прагненням до розумової діяльності, структурованого вирішення завдань і, можливо, доступної інтелектуальної втечі. Така швидка й глибока адаптація дозволила кросвордам швидко стати чимось позачасовим, попри їхню молодість, показуючи, наскільки вдало вони з’явилися у потрібний момент і як глибоко змогли знайти відгук у суспільстві. Таким чином, сучасний кросворд, хоч і є відносно недавнім винаходом, швидко пройшов шлях від газетної новинки до впливової культурної сили, даруючи втечу, освіту та стимул для розуму багатьом поколінням, міцно закріпивши своє місце як позачасове хобі.

До появи сучасного кросворду людство століттями займалося словесними іграми, що сягають найдавніших форм мови. Кросворди концептуально вкорінені у двох давніх традиціях: словесних квадратах та загадках. Найпрямішим попередником сучасної кросвордної сітки є словесний

квадрат — особливий тип акростичної головоломки, де слова розташовані так, що читаються однаково як по горизонталі, так і по вертикалі.

Докази стародавніх словесних головоломок включають Саторський квадрат, п'ятирядковий латинський паліндром з п'яти слів: SATOR AREPO TENET OPERA ROTAS (рис. 1.1)

Рисунок 1.1 – Саторський квадрат

Це перекладається як "фермер Арепо працює плугом". Цей квадрат був виявлений у численних римських місцях, найвідоміше – у руїнах Помпеїв, причому найдавніше відоме відкриття датується до 62 року нашої ери. Він був широко поширеним "мемом" Римської імперії, знайденим від Риму до

сучасної Англії та Сирії. Найдавніші Саторські квадрати були знайдені у "ROTAS-формі" (з "ROTAS" як верхнім рядком), хоча "SATOR-форма" стала більш домінуючою в середньовічний період. Складання словесних квадратів було за своєю суттю складним завданням, причому "святий грааль" задовільного 10-літерного словесного квадрата залишався невловимим навіть за допомогою сучасних комп'ютерів. Ця історична безперервність виявляє фундаментальну і стійку людську когнітивну рису: невід'ємне бажання ідентифікувати закономірності, розв'язувати лінгвістичні головоломки та займатися словесними іграми.

Широка популярність Саторського квадрата в давнину, що функціонував подібно до сучасного культурного "мема", свідчить про те, що такі інтелектуальні виклики не обмежувалися елітою, а резонували з широкою аудиторією, подібно до того, як кросворди зазвучали століттями пізніше. Це глибоке історичне коріння дає фундаментальне пояснення того, чому кросворди, коли вони нарешті з'явилися у своїй сучасній, вдосконаленій формі, так сильно і швидко знайшли відгук у масової аудиторії. Винахід не був абсолютно новим за своєю суттю; скоріше, він успішно використав давню, невід'ємну людську схильність до структурованої лінгвістичної гри.

У 19 столітті в Англії ці стародавні концепції словесних квадратів еволюціонували до більш примітивних форм кросвордів, призначених насамперед для дітей. Ці головоломки іноді включали малюнки як підказки або мали явний освітній ухил, з'являючись у дитячих книжках-головоломках та різних періодичних виданнях. Важливо, що ці ранні, елементарні форми не були значним дозвіллям для дорослих до появи "ворд-кросу" Артура Вінна в 1913 році (рис. 1.2), що ознаменувало чітку зміну цільової аудиторії головоломки та її сприйнятої інтелектуальної цінності.

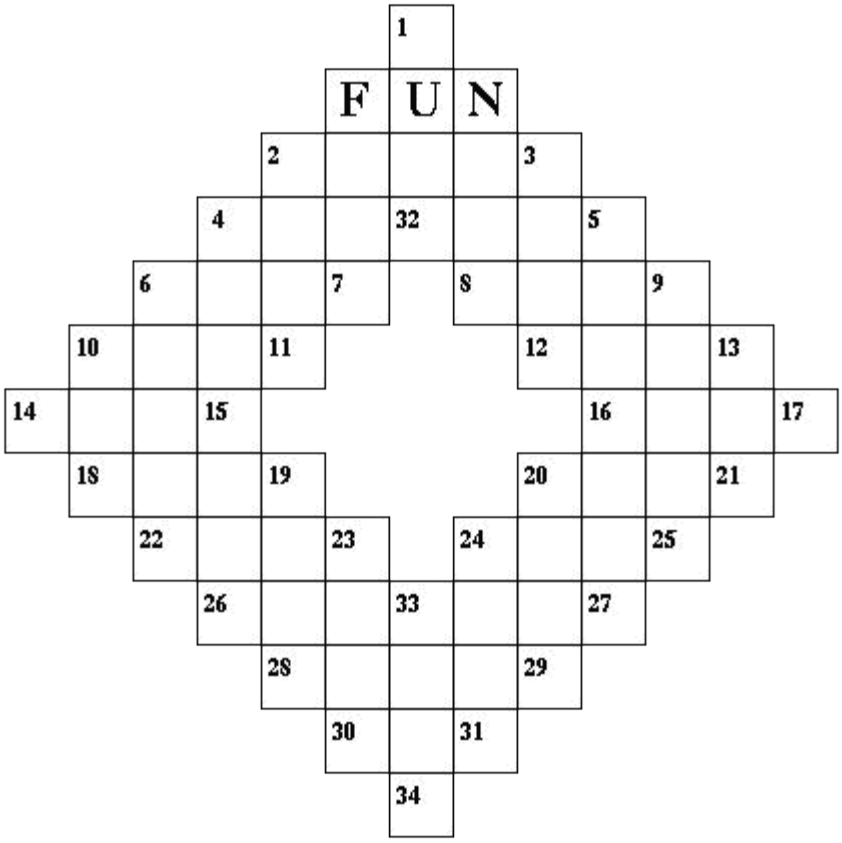

Рисунок 1.2 – Артур Вінн, 21 грудня 1913 року, газета The New York World

Цей перехід від "елементарних" головоломок, що друкувалися в "дитячих книжках-головоломках", до "серйозного дорослого дозвілля" в США відкриває цікаву трансформацію як у складності головоломки, так і в її цільовій аудиторії. Така зміна показує, що вдосконалення формату та правил головоломки, як це згодом було здійснено Вінном, зіграло важливу роль у її виході за межі простої дитячої розваги. Перехід від "елементарного" до "серйозного" відкриває розширені можливості для інтелектуального виклику, складнішої структури та нових способів зацікавлення, які стали привабливими для дорослих. Це дозрівання стало ключовим фактором для її популярності серед ширшої аудиторії, що вимагала більшого, і підняло її статус до рівня культурного феномена.

Після початкового успіху Вінна інші американські газети швидко усвідомили привабливість "щойно відкритого дозвілля" на початку 1920-х років. Протягом усього десятиліття кросворди стали регулярною рубрикою

майже в усіх американських газетах. До середини 1920-х років кросворди досягли величезної популярності, причому сучасні звіти стверджували, що вони "повністю захопили Нью-Йорк", ставши щоденною рубрикою в багатьох газетах. До 1925 року, за оцінками, понад десять мільйонів людей у США щодня розгадували кросворди.

Таблиця 1.1 – Ключові віхи в історії кросвордів

Дата	Подія/Віха	Ключова фігура/Публікація	Коротке значення
До 62 р. н.е.	Відкриття Саторського квадрата	Давній Рим (Помпеї)	Найдавніший відомий словесний квадрат, що демонструє давню людську схильність до словесних ігор.
19 ст.	Поява елементарних кросвордів	Англія (дитячі книги)	Примітивні попередники сучасних кросвордів, призначені для дітей.
21 грудня 1913	Публікація першого "Ворд-Кросу"	Артур Вінн, The New York World	Народження сучасного кросворду, що започаткував нову форму дозвілля.
Кілька тижнів після 1913	Зміна назви на "Крос-Ворд"	The New York World	Випадкова зміна назви, яка стала загальновизнаною.
1922 (лютий)	Перший кросворд у британському виданні	Pearson's Magazine	Початок поширення кросвордів у Великій Британії.
1924 (18 квітня)	Видання першої книги кросвордів	Simon & Schuster	Каталізатор масової популярності та комерціалізації кросвордів у США.
1924 (2 листопада)	Перший кросворд у британській національній газеті	Sunday Express	Подальше закріплення кросвордів у британській пресі.
1925 (липень)	Винахід криптичного кросворду	Daily Telegraph (Едвард Пауїс-Матерс)	Поява більш складної, характерної британської форми кросворду.
1930 (1 лютого)	Перший кросворд у The Times (Лондон)	The Times	Прийняття кросвордів престижними британськими виданнями.
1942 (лютий)	Перший кросворд у The New York Times	The New York Times	Остаточне прийняття кросвордів останнім великим американським противником, що підкреслює їхню неминучу популярність.

Широкомасштабна "кросвордна лихоманка" призвела до різкого зростання продажів словників та значного збільшення відвідуваності публічних бібліотек, оскільки розгадувачі прагнули отримати знання, необхідні для розгадування головоломок, і, у свою чергу, розширити свій лінгвістичний репертуар. Вплив головоломки на американський словниковий запас став "чутним", причому раніше "сплячі" або "неясні слова з трьох літер" раптово увійшли в повсякденну розмову. Сучасні спостерігачі оптимістично передбачали, що вдумлива робота над кросвордами виховає "більш

обережного та вільного користувача хорошої англійської мови". Цей факт демонструє, що, здавалося б, проста гра мала глибокі та відчутні економічні та освітні переваги, крім простої розваги. Вона стимулювала супутні галузі (такі як видавництво словників) та зміцнила цінність традиційних освітніх установ (бібліотек), стимулюючи громадську залученість. Для газет це стало беззаперечним стимулом для тиражу, довівши її значну комерційну життєздатність. Це ілюструє, як культурні примхи, навіть ті, що спочатку відкидалися як легковажні, можуть мати непередбачені позитивні наслідки для суміжних секторів та активно сприяти грамотності та інтелектуальній залученості, стаючи таким чином значною культурною та економічною силою.

Таблиця 1.2 – Культурні та суспільні впливи кросвордної манії

Категорія впливу	**Конкретний приклад/прояв**	**Короткий опис/значення**	**Відповідний період**
Економічний	Зростання продажів словників та відвідуваності бібліотек	Стимулювання супутніх галузей та підвищення цінності традиційних знань.	1920-ті
Лінгвістичний	Вплив на словниковий запас та мовлення	Введення "сплячих" слів у повсякденну розмову, покращення мовних навичок.	1920-ті
Соціальний	Моральна паніка та суспільні дебати	Звинувачення в "марнуванні розумових здібностей", "національній загрозі", вплив на гендерні ролі та розлучення.	1920-ті
Медійний	Збільшення тиражу газет	Кросворди стали ключовим фактором залучення читачів та комерційної життєздатності газет.	1920-ті - 1940-ті
Культурний	Натхнення для мистецтва та моди	Пісні, бродвейські шоу, мотиви на одязі, що відображають глибоке проникнення в повсякденне життя.	1920-ті
Психологічний	Механізм подолання в кризові часи	Надання відчуття контролю, відволікання та розваги під час Великої депресії та Другої світової війни.	1930-ті - 1940-ті
Стратегічний	Використання в розвідці	Застосування кросвордів для вербування та ненавмисний витік кодових назв під час Другої світової війни.	1940-ті

У 1920-х роках кросворди стали дуже популярними, і газети активно рекламували їх. Вони писали, що розгадування кросвордів допомагає людям вивчати нові слова і покращувати свій словниковий запас. Ця «кросвордна лихоманка» вплинула на освіту й грамотність: люди почали більше купувати

словники та частіше ходити до бібліотек, щоб шукати потрібну інформацію для розгадування головоломок. Завдяки цьому у повсякденній мові стали частіше з'являтися складні та рідковживані слова, які раніше майже не використовували. Це навело на думку, що кросворди допомагають людям краще володіти мовою і тренують мозок. Багато хто вважав, що вони здатні зробити англійську мову більш багатою і правильною у використанні. Загалом кросворди не лише розважають, а й розвивають словниковий запас, увагу до деталей та навички вирішення завдань. До того ж розгадування приносить задоволення від досягнення мети.

Кросворди — це не лише розвага, а й корисний спосіб навчання, який активно використовують у школах та університетах. Вони допомагають краще зрозуміти наукові поняття, вивчити нові терміни, запам'ятати інформацію та отримати важливі знання в різних сферах, таких як медицина, стоматологія чи фармація. Дослідження довели, що, якщо поєднувати кросворди зі звичайними лекціями, студенти краще засвоюють матеріал, показують кращі результати і з більшим задоволенням навчаються.

Особливо ефективними виявилися так звані "ігрові" кросворди, які допомагають покращити англійську мову і підвищують інтерес студентів до навчання. Ці завдання перетворюють навчання на цікаву гру і позбавляють його сухості та нудних правил. Вирішуючи кросворди, люди не лише розвивають знання, але й отримують задоволення від процесу, що мотивує їх продовжувати навчатися.

Кросворди вдало поєднують інтелектуальну активність із розвагою. Вони показують, що освіта може бути легкою, цікавою і корисною одночасно, залучаючи до навчання людей різного віку та рівня підготовки.

Крім того, що кросворди корисні для навчання, вони також допомагають краще працювати мозку. Вважається, що розгадування кросвордів середньої складності створює нові зв'язки між клітинами мозку. Зокрема, дослідження показало, що люди з легкими проблемами з пам'яттю, які регулярно розгадували кросворди, покращували свій стан більше, ніж ті, хто займався

іншими подібними іграми онлайн. Це доводить, що кросворди мають унікальні переваги.

Кросворди часто називають чудовою "зарядкою для мозку". Вони кидають розумний виклик, який допомагає мислити швидше і тримати мозок у тонусі. Для багатьох людей кросворди стали частиною їхнього щоденного життя, бо вони одночасно цікаві та корисні. Їхня популярність тримається через те, що вони допомагають не тільки відпочити, але й потренувати розум.

Наукові дослідження підтверджують, що кросворди позитивно впливають на мозок і навіть кращі за деякі інші вправи для розвитку мислення. Вони не лише розважають чи збагачують словниковий запас, але й допомагають людям підтримувати мозок у формі. Це особливо важливо для тих, хто хоче зберегти свою пам'ять і ясність мислення. Завдяки своїм користям, кросворди можна назвати простою та приємною вправою для зміцнення розумових здібностей.

1.2 Аналіз існуючих засобів генерації кросвордів

У сучасну епоху створення кросвордів значною мірою покладається на програмне забезпечення. Спробуємо надати всебічний огляд відомих програмних генераторів кросвордів, аналізуючи їхні ключові функції, цінову політику та, що особливо важливо для україномовних користувачів, підтримку різних мов, зокрема української та кирилиці.

Для об'єктивної оцінки програмних генераторів кросвордів було застосовано такі критерії:

Функціональність та можливості: Оцінюється простота використання, ступінь автоматизації (наприклад, генерація сітки, автоматичне створення підказок), інтеграція словників, підтримка різних типів кросвордів (наприклад, криптичні), а також наявність розширених функцій, що покращують процес створення.

Варіанти експорту та публікації: Розглядаються доступні формати файлів для збереження та поширення кросвордів (наприклад, PDF, Word, зображення, спеціалізовані формати), а також можливість ділитися ними онлайн або вбудовувати на веб-сайти.

Цінова політика: Аналізується модель ціноутворення, включаючи безкоштовні версії, платні ліцензії (одноразова покупка, підписка, вбудовані покупки), а також наявність пробних версій або freemium-моделей.

Підтримка мов, зокрема української та кирилиці: Це критичний критерій. Оцінюється наявність явних заяв про підтримку Unicode, кирилиці або конкретних мов, а також непрямі докази, такі як приклади використання. Важливо враховувати, що різні мови мають унікальні правила для кросвордів, наприклад, обробка діакритичних знаків або диграфів, що може впливати на сумісність програмного забезпечення.

Цільова аудиторія та сценарії використання: Визначається, чи орієнтований інструмент на викладачів, професійних розробників, ентузіастів-хобістів чи широку публіку, що допомагає зрозуміти його основне призначення та оптимальні сценарії застосування.

1. Crossword Labs

Crossword Labs є популярним безкоштовним онлайн-інструментом, широко відомим завдяки своїй простоті та швидкості створення кросвордів. Його доступність та інтуїтивно зрозумілий інтерфейс зробили його одним з найпоширеніших виборів для швидкої генерації головоломок.

Ключові функції: Crossword Labs дозволяє створювати кросворди "за лічені секунди" без необхідності реєстрації, відображення реклами чи додавання водяних знаків, що є значною перевагою для швидкого та безперешкодного використання. Ця простота є ключовим фактором його широкої популярності та доступності для користувачів будь-якого рівня. Створені кросворди можна легко поширювати за допомогою унікального URL-посилання, розгадувати їх безпосередньо онлайн або вбудовувати на веб-

сайти. Це робить його ідеальним для інтерактивного навчання, створення контенту для блогів або швидких розважальних заходів.

Рисунок 1.3 – Crossword Labs

Інструмент підтримує експорт головоломок у різні формати, включаючи PDF, Microsoft Word, зображення та SVG. Така різноманітність форматів забезпечує гнучкість для друку, подальшого редагування або використання в графічному дизайні. Користувачі також можуть обирати один з трьох рівнів приватності для своїх кросвордів: публічний (доступний для всіх і пошукових систем, безкоштовно), прихований (доступний лише за прямим посиланням, для членів) та приватний (доступний лише для творця, для членів).

Мовна підтримка (з акцентом на українську/кирилицю): Хоча в загальному описі Crossword Labs прямо не згадується підтримка кирилиці, аналіз наявних даних виявляє конкретні приклади вбудованих кросвордів, які використовують кириличні символи як у підказках, так і у відповідях. Це є вагомим доказом того, що Crossword Labs ефективно підтримує кирилицю, а отже, має бути сумісним і з українською мовою. Ця можливість є значною перевагою для україномовних користувачів, які шукають інструмент для створення кросвордів рідною мовою.

Ціна та доступність: Основні функції та створення публічних кросвордів є безкоштовними. Функції "прихований" та "приватний" доступні лише для "членів", що може вказувати на платне членство для розширених можливостей контролю приватності, хоча конкретна ціна не вказана у наданих джерелах.

Рекомендовані сценарії використання: Crossword Labs є відмінним вибором для викладачів, яким потрібно швидко створювати навчальні матеріали, для організаторів заходів, які шукають прості розважальні головоломки, або для будь-кого, хто потребує швидкого, безкоштовного та функціонального онлайн-інструменту для створення кросвордів, особливо з урахуванням підтримки кирилиці.

Додаткові висновки: Crossword Labs досягає широкої популярності завдяки своїй безкоштовній моделі та простоті використання, що робить його надзвичайно доступним для масового користувача. Однак, його "членські" функції для прихованих та приватних кросвордів вказують на модель freemium. Це дозволяє сервісу залучати велику базу користувачів, пропонуючи базові функції безкоштовно, а потім монетизувати більш просунуті потреби, такі як підвищена приватність або контроль над контентом. Така стратегія забезпечує стійкість сервісу, одночасно задовольняючи потреби як випадкових, так і більш вимогливих користувачів.

2. EclipseCrossword

EclipseCrossword — це безкоштовне програмне забезпечення, призначене для ПК на базі Windows, що дозволяє швидко створювати кросворди. Воно особливо популярне в освітніх колах завдяки своїй простоті та орієнтації на навчальні потреби.

Ключові функції: Користувач надає слова та підказки, а EclipseCrossword за лічені секунди перетворює їх на готову головоломку. Створені головоломки можна друкувати або поширювати онлайн. Програма ідеально підходить для викладачів та батьків, допомагаючи переглядати словниковий запас, орфографію, викладати нові терміни та проводити вікторини. Можливість створювати власні підказки дозволяє адаптувати вміст

до конкретних студентів та предметів, а онлайн-поширення підтримує навчання вдома. Кросворди можна додавати на веб-сайти для залучення користувачів, при цьому програма не додає реклами та не відстежує відвідувачів. Це також дозволяє покращити інформаційні бюлетені, додаючи посилання на онлайн-головоломки або файли для імпорту в Word.

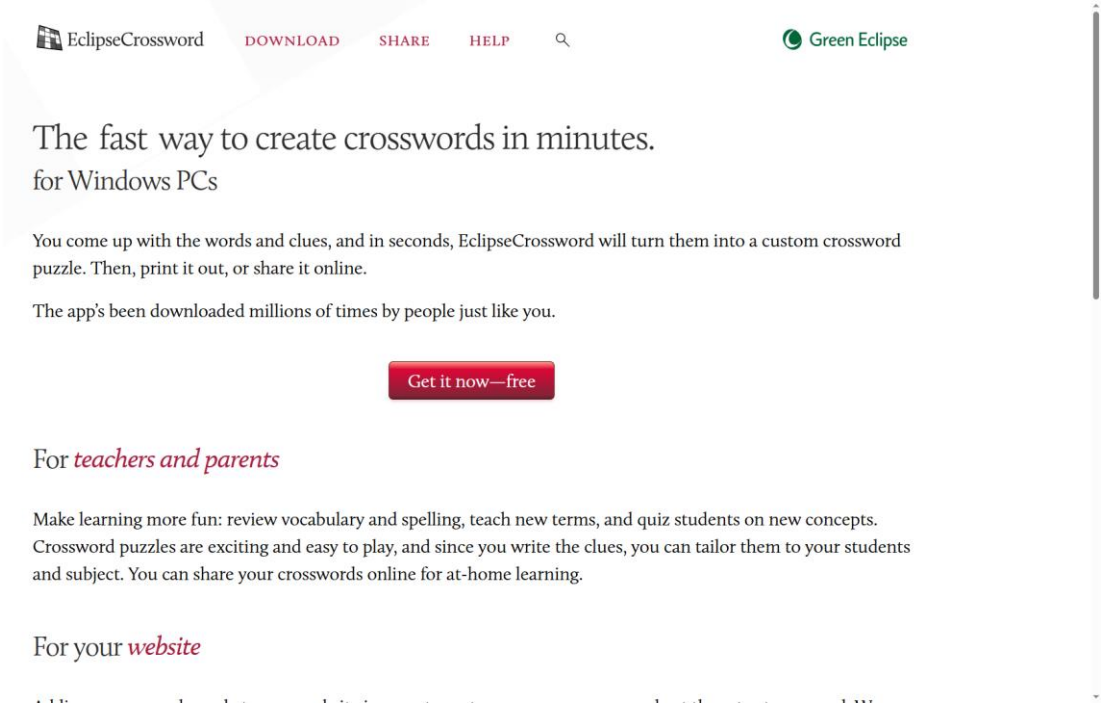

Рисунок 1.4 – EclipseCrossword

Мовна підтримка (з акцентом на українську/кирилицю): EclipseCrossword підтримує англійську та інші західноєвропейські мови, такі як фінська, французька, німецька, ірландська, італійська, норвезька, португальська, іспанська та шведська. Програма має обмеження щодо мов, які вимагають спеціальної обробки пар літер як одного символу (наприклад, голландська "IJ") або читаються справа наліво (наприклад, іврит, тайська). Це свідчить про те, що її архітектура не оптимізована для нелатинських алфавітів, що робить її менш придатною для української мови.

Ціна та доступність: EclipseCrossword є "абсолютно безкоштовним" програмним забезпеченням, а не пробною версією.

Рекомендовані сценарії використання: Цей інструмент є відмінним вибором для викладачів, батьків та невеликих організацій, які шукають

безкоштовне, надійне програмне забезпечення для Windows для створення освітніх або розважальних кросвордів, якщо їхній вміст не вимагає підтримки кирилиці.

Додаткові висновки: EclipseCrossword чітко орієнтований на західноєвропейські мови, про що свідчить перелік підтримуваних мов. Відсутність згадки про кирилицю та явні обмеження щодо обробки диграфів або письма справа наліво вказують на те, що програма не була розроблена з урахуванням складності нелатинських алфавітів. Це означає, що, незважаючи на її безкоштовність та функціональність для латинських мов, вона, ймовірно, не підходить для створення кросвордів українською мовою. Цей висновок є критичним для користувача, який шукає підтримку української мови, оскільки він вказує на потенційні проблеми сумісності, які можуть виникнути при спробі використовувати кириличні символи.

3. edHelper.com

edHelper.com є безкоштовним онлайн-конструктором кросвордів, який є частиною більшої освітньої платформи, що пропонує різноманітні навчальні ресурси. Цей інструмент спеціалізується на створенні кросвордів, орієнтованих на словниковий запас, для дітей та учнів, що робить його цінним ресурсом для вчителів та батьків.

Ключові функції: Інструмент спеціалізується на створенні кросвордів, орієнтованих на словниковий запас, для дітей та учнів. Це робить його цінним ресурсом для вчителів та батьків. edHelper.com також пропонує інші види головоломок, такі як пошук слів та головоломки для орфографії, що дозволяє створювати комплексні навчальні робочі зошити. Користувачі можуть обирати відповідний рівень складності, від дитячого садка до дорослого, що забезпечує адаптивність контенту. Для створення головоломки потрібно ввести щонайменше десять слів та підказок.

Мовна підтримка (з акцентом на українську/кирилицю): У наданих джерелах немає прямої згадки про підтримку мов, крім загального акценту на "словниковому запасі". Це свідчить про те, що інструмент, ймовірно,

розроблений переважно для англійської мови, і підтримка кирилиці не гарантується. Для українського користувача це означає, що інструмент, швидше за все, не підтримує українську мову, оскільки це не є його основною функцією, і немає жодних доказів підтримки кирилиці.

Рисунок 1.5 – edHelper.com

Ціна та доступність: edHelper.com пропонує свій конструктор кросвордів як безкоштовний онлайн-інструмент.

Рекомендовані сценарії використання: Ідеально підходить для вчителів початкової та середньої школи, які створюють прості, освітні кросворди для перевірки словникового запасу, а також для батьків, які шукають додаткові навчальні матеріали, за умови, що вміст є англомовним.

Додаткові висновки: edHelper.com чітко позиціонує себе як ресурс для освіти, орієнтований на дітей та вчителів. Це означає, що його функціонал, ймовірно, спрощений і не включає складних функцій, які можуть бути потрібні професіоналам, або розширеної багатомовної підтримки. Його безкоштовна модель добре відповідає бюджетним обмеженням освітніх установ. Відсутність згадки про багатомовність, у поєднанні з його фокусом на "словниковому запасі", вказує на те, що він, швидше за все, призначений

для англомовного ринку, що робить його менш придатним для українського користувача без додаткових перевірок. Це підкреслює важливість перевірки мовної сумісності для будь-якого інструменту, особливо коли він не позиціонується як багатомовний.

4. Puzzle-Maker.com

Puzzle-Maker.com є дуже простим онлайн-інструментом для створення кросвордів, що дозволяє користувачам вручну вводити власні слова та підказки. Його мінімалістичний дизайн орієнтований на пряме та швидке створення головоломок.

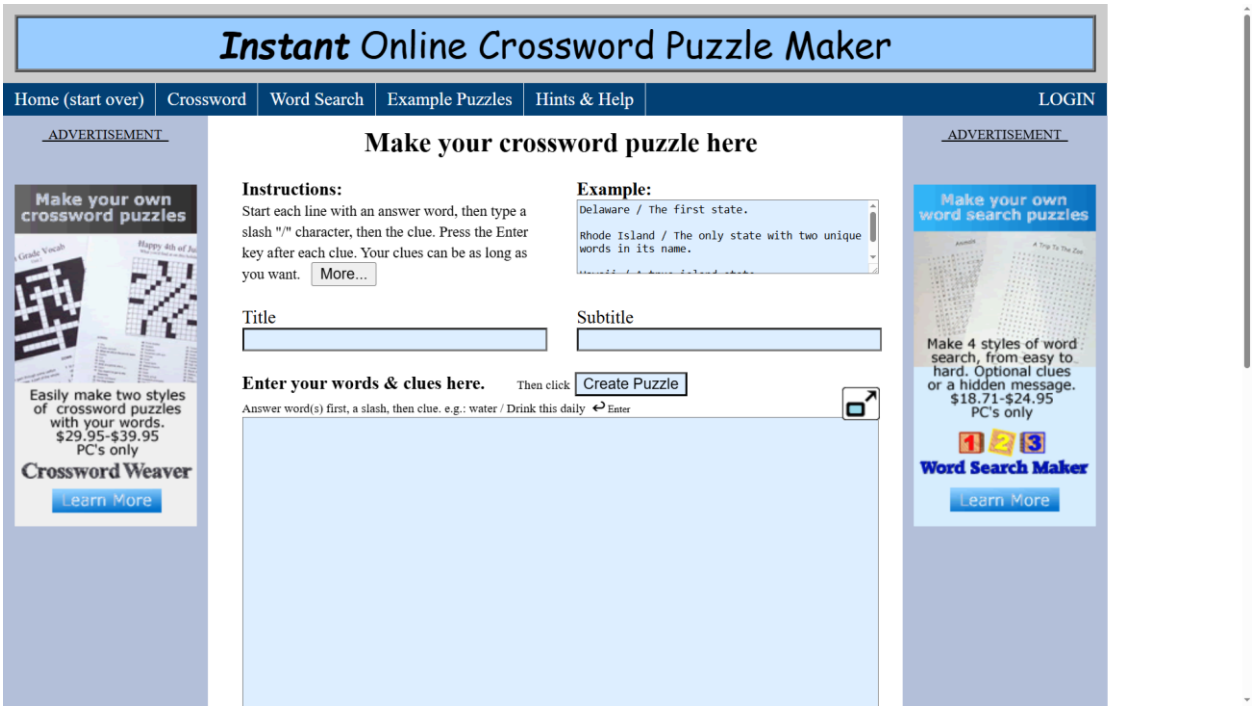

Рисунок 1.6 – Puzzle-Maker.com

Ключові функції: Інструмент надає прямий інтерфейс, де користувачі вводять кожне слово та відповідну підказку, розділяючи їх скісною рискою, а потім натискаючи Enter. Цей метод забезпечує повний контроль над вмістом. Після створення головоломки користувачі можуть легко редагувати слова та підказки, якщо це необхідно. Також підтримується використання довгих підказок, що дозволяє надавати детальніші описи.

Мовна підтримка (з акцентом на українську/кирилицю): У наданих джерелах немає жодної згадки про підтримку конкретних мов або символів. З

огляду на його мінімалістичний дизайн та відсутність розширених функцій, він, ймовірно, орієнтований на латинські алфавіти, і підтримка кирилиці не гарантується. Якщо кирилиця підтримується, то це буде лише на рівні введення символів, без інтелектуальної допомоги чи словників. Для українського користувача це означає, що інструмент не надасть жодної інтелектуальної допомоги, лише дозволить вводити символи, якщо система їх підтримує, що вимагає від користувача повної відповідальності за правильність введення та форматування.

Ціна та доступність: Ціна не вказана , але простий інтерфейс та відсутність будь-яких згадок про оплату свідчать про те, що інструмент, ймовірно, є безкоштовним для використання.

Рекомендовані сценарії використання: Цей інструмент підходить для користувачів, яким потрібен дуже базовий, швидкий інструмент для створення невеликих, індивідуальних кросвордів без необхідності складних функцій, автоматизації чи інтеграції зі словниками.

Додаткові висновки: Puzzle-Maker.com є яскравим прикладом мінімалістичного генератора кросвордів. Його простота є як перевагою (швидкість, відсутність відволікань), так і обмеженням (відсутність автоматизації, інтеграції словників, розширених функцій). Це свідчить про те, що він задовольняє дуже специфічну, низькотехнологічну потребу, де користувач має повний контроль над кожним словом і підказкою.

5. Crossword Puzzle Generator (crossword-generator.com)

Crossword Puzzle Generator (crossword-generator.com) — це онлайн-конструктор, що пропонує швидке та настроюване створення кросвордів для різних цілей, від освіти до розваг.

Ключові функції: Інструмент використовує інтелектуальний алгоритм для миттєвого створення персоналізованих кросвордів на основі наданих питань та відповідей. Користувачі можуть регулювати розміри сітки, параметри складності та інші налаштування, щоб створити ідеальний кросворд. Інструмент позиціонується як зручний для викладачів та студентів

(для навчальних матеріалів), а також для батьків (для розваг без екрана) та любителів головоломок (для вечірок та ігрових вечорів). Він також дозволяє генерувати та завантажувати головоломки на будь-якому пристрої, що забезпечує доступність у будь-який час та в будь-якому місці. Створені головоломки можна завантажити у форматі PDF для зручного друку.

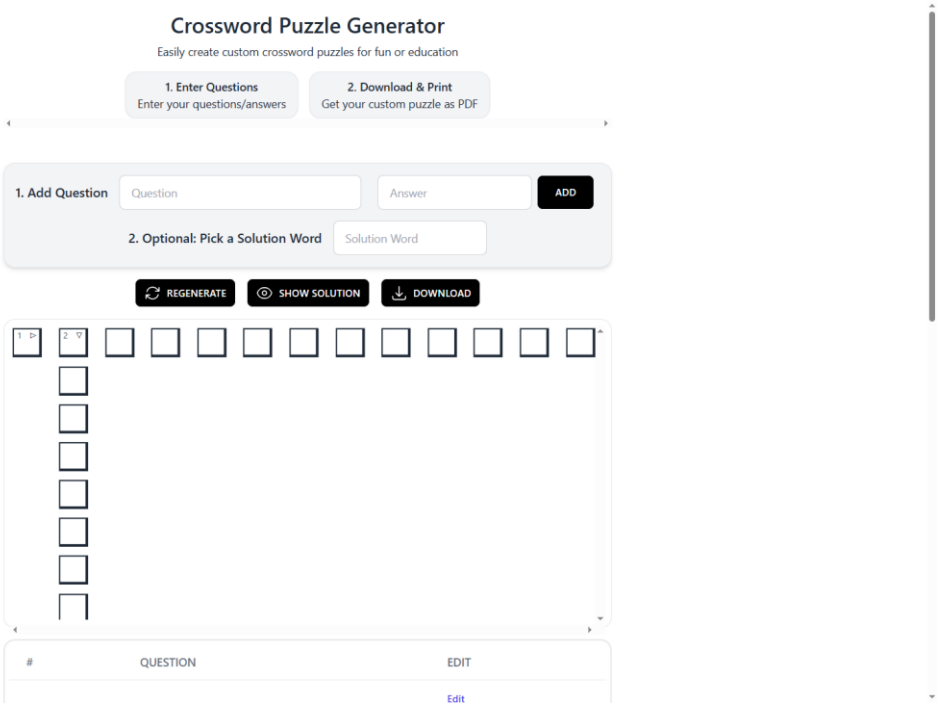

Рисунок 1.7 – Crossword Puzzle Generator

Мовна підтримка (з акцентом на українську/кирилицю): на сайті продукту немає прямої згадки про підтримку конкретних мов або символів. Як і інші базові онлайн-інструменти, він, ймовірно, орієнтований на латинські алфавіти, і підтримка кирилиці не гарантується. Для використання української мови потрібно буде перевірити сумісність символів, оскільки відсутність явних заяв про підтримку Unicode або кирилиці може вказувати на потенційні обмеження.

Ціна та доступність: Інструмент заявлений як "безкоштовний онлайн-конструктор кросвордів". Однак, існує певна неоднозначність у джерелах щодо цінової моделі та назви. Хоча його офіційний опис стверджує, що він є безкоштовним, інші джерела згадують "Crossword Creator" або "Crossword Generator" як платні продукти (наприклад, $10 на Itch.io або різні ціни на Etsy).

Це може свідчити про наявність різних продуктів під схожими назвами на ринку або про модель freemium, де онлайн-версія є безкоштовною, а більш просунуті версії (можливо, з іншим функціоналом або як окреме програмне забезпечення) є платними. Для цілей цього звіту вважається, що crossword-generator.com є безкоштовним, як зазначено в його власному описі, але користувачам варто бути обережними при виборі інструментів зі схожими назвами.

Рекомендовані сценарії використання: Цей генератор підходить для широкого кола користувачів, включаючи викладачів, батьків та любителів головоломок, які шукають швидкий, настроюваний та безкоштовний онлайн-інструмент для створення кросвордів, за умови, що мовна підтримка відповідає їхнім потребам.

Додаткові висновки: Існує помітна неоднозначність у ціновій моделі та потенційна плутанина назв, що може ускладнити вибір для потенційних користувачів. Хоча crossword-generator.com позиціонується як безкоштовний, наявність платних продуктів зі схожими назвами в інших джерелах вимагає уважності. Це підкреслює важливість ретельної перевірки інформації про продукт, особливо коли назви схожі, щоб уникнути непорозумінь щодо функціоналу та вартості.

6. Crossword Creator (Bearwood Labs)

Crossword Creator від Bearwood Labs — це платний інструмент, що дозволяє створювати кросворди, які легко інтегруються з популярними програмами для презентацій та документів, такими як PowerPoint, Google Slides та Keynote.

Ключові функції: Інструмент дозволяє створювати кросворди з повним контролем над їхнім зовнішнім виглядом, використовуючи можливості PowerPoint або Google Slides для налаштування шрифтів та контурів. Процес створення простий: користувачі додають відповіді та підказки, і головоломка генерується. Створені кросворди можуть використовуватися як для комерційних, так і для особистих цілей, що надає гнучкість у застосуванні.

Завантажений файл є файлом PowerPoint (.pptx), який можна додатково налаштовувати. Він також сумісний з Google Slides та Keynote. Інструмент підтримує додавання необмеженої кількості підказок та відповідей, при цьому генератор створює макети, щоб вмістити всі слова.

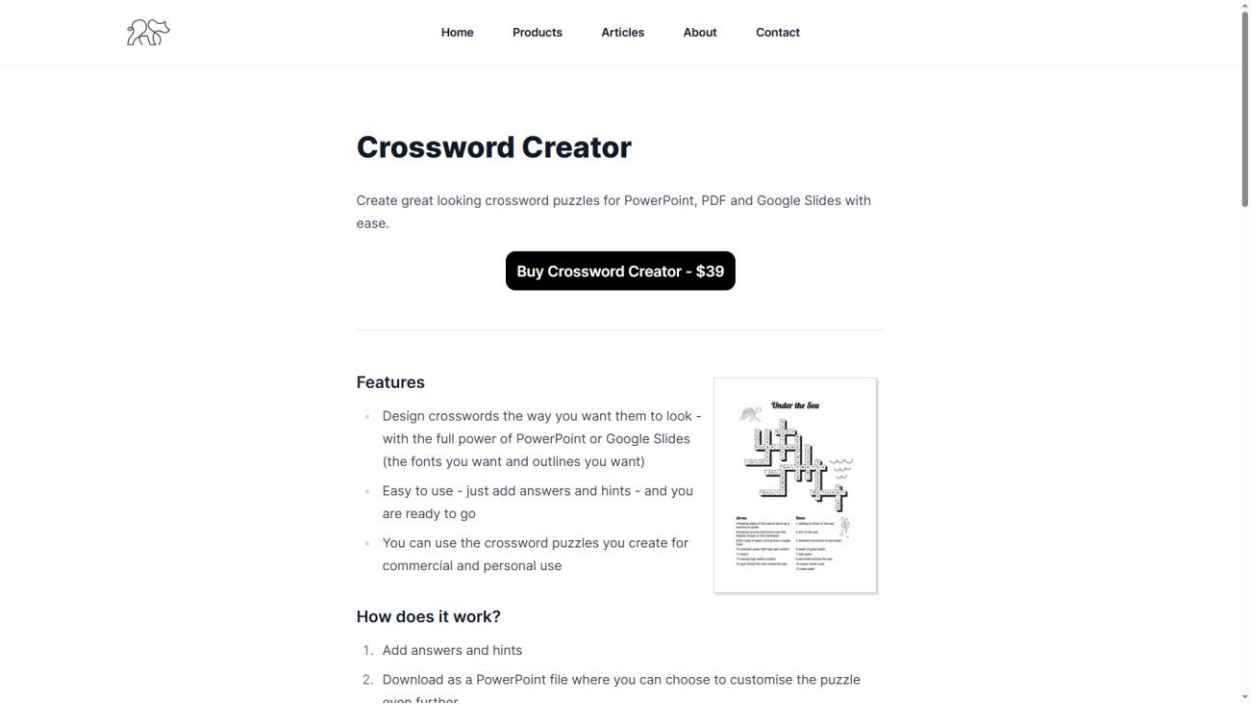

Рисунок 1.8 – Crossword Creator

Мовна підтримка (з акцентом на українську/кирилицю): У наданих джерелах немає жодної згадки про підтримку конкретних мов або символів. Оскільки кінцевий продукт є файлом PowerPoint, сумісність з українською мовою та кирилицею, ймовірно, залежатиме від підтримки шрифтів та кодування в самому PowerPoint або Google Slides. Це означає, що, хоча пряма підтримка не заявлена, можливість використання кирилиці може бути опосередкованою через можливості програмного забезпечення для презентацій.

Ціна та доступність: Crossword Creator є платним продуктом, його вартість становить $39. Після одноразової покупки користувач отримує можливість створювати необмежену кількість кросвордів для комерційного та особистого використання.

Рекомендовані сценарії використання: Цей інструмент ідеально підходить для професіоналів, викладачів або творців контенту, яким потрібні високоякісні, настроювані кросворди для презентацій, навчальних матеріалів або комерційних продуктів, і які вже використовують PowerPoint або Google Slides у своїй роботі.

Додаткові висновки: Crossword Creator відрізняється від більшості інших генераторів своєю інтеграцією з офісним програмним забезпеченням. Це дозволяє користувачам максимально використовувати знайомі інструменти для дизайну та форматування, перетворюючи кросворд на повноцінний візуальний елемент, а не просто сітку з текстом. Платна модель та орієнтація на комерційне використання вказують на те, що цей інструмент призначений для користувачів, які цінують високу якість, гнучкість дизайну та можливість монетизації свого контенту. Це відрізняє його від безкоштовних онлайн-інструментів, які часто пропонують менші можливості для налаштування зовнішнього вигляду.

Аналіз популярних програм для створення кросвордів показує, що вони різні за функціями, цінами і підтримкою мов, залежно від того, що потрібно користувачам.

Популярність і доступність: Онлайн-інструменти, як-от Crossword Labs, які дають можливість створювати кросворди безкоштовно, дуже поширені. Їх цінують за те, що ними легко користуватися, і немає потреби платити. Це показує, що багато людей хочуть швидко і просто створювати головоломки, не витрачаючи гроші.

Багато програм для створення кросвордів, наприклад, EclipseCrossword та edHelper.com, створені спеціально для використання в освіті. Вони допомагають навчатися та перевіряти знання, показуючи, як кросворди можуть бути корисним інструментом у навчальному процесі.

Мовна підтримка та кирилиця: Це є критичним аспектом для україномовних користувачів. Хоча деякі інструменти, як-от Crossword Labs, демонструють непряму підтримку кирилиці через приклади використання

російської мови, інші, як EclipseCrossword, прямо вказують на обмежену підтримку нелатинських алфавітів. Це означає, що користувачам, які бажають створювати кросворди українською мовою, необхідно ретельно перевіряти сумісність, оскільки відсутність явних заяв про підтримку Unicode або кирилиці може призвести до проблем з відображенням або функціональністю.

Різноманітність цінових моделей: Ринок генераторів кросвордів включає як повністю безкоштовні інструменти, так і платні рішення з одноразовою покупкою або freemium-моделями. Вибір залежить від потреб користувача в розширених функціях, приватності та можливостях комерційного використання.

Еволюція та спеціалізація: Від простих веб-форм до інтегрованих рішень для презентацій, програмне забезпечення для створення кросвордів еволюціонує, пропонуючи спеціалізовані функції для різних ніш, від швидкого створення для особистого використання до професійного дизайну для комерційних цілей.

Отже, не зважаючи на значну кількість доступних на ринку генераторів кросвордів, жодні з них не підтримують повноцінно українську мову, що обмежує їх використання як навчального засобу в українських університетах, школах або інших навчальних закладах.

Таким чином, розробка програмного забезпечення для створення тематичних кросвордів є актуальною задачею.

1.3 Визначення вимог до програмного забезпечення для створення тематичних кросвордів

Цей проєкт має на меті створити інструмент для створення кросвордів, який буде працювати саме з українською мовою. Основні можливості програми включають легке введення слів і підказок вручну, а також автоматичний підбір слів за заданою темою. Це допоможе зробити процес більш цікавим і корисним для навчання. Програма враховуватиме особливості

української мови, щоб кросворди були точними і відповідали культурному контексту.

Створення такого програмного забезпечення має кілька важливих труднощів, для яких є свої рішення. Найперше — особливості української мови, як-от її кирилиця, букви "ґ", "ї", "є", м'який знак (ь), апостроф (') та сполучення літер (дж, дз). Усе це створює проблеми для правильного розташування слів у таблицях, розпізнавання символів і мовної точності. Щоб вирішити це, потрібно впровадити підтримку Unicode та врахувати правила української орфографії й фонетики при створенні та показі таблиць. Також знадобляться спеціальні інструменти для обробки української мови, які допоможуть розбирати слова за формами та розуміти їх значення.

Щоб створити справді якісний вибір слів і контекстуальні підказки, недостатньо просто порівнювати ключові слова. Потрібно добре розуміти, як слова пов’язані між собою за змістом і значенням та враховувати контекст. Запропоноване рішення передбачає використання сучасних методів обробки мови (НЛП), таких як українські моделі для визначення схожості між словами за змістом й інструменти типу українського WordNet для побудови зв’язків між значеннями слів. А також важливо використовувати або створити українську велику мовну модель (ВММ), яка допоможе створювати якісні та змістовні підказки, що відповідають контексту.

Третій важливий момент – знайти баланс між складними і розумними функціями створення та простим і зручним інтерфейсом для користувачів. Це потрібно, щоб більше людей могли користуватися продуктом і залишалися задоволеними. Для цього планується створити зручний інструмент, що дозволяє легко та швидко налаштовувати кросворди, отримувати підказки в режимі реального часу, а також надавати багато варіантів оформлення та збереження готових кросвордів.

Для ефективної роботи генератора кросвордів необхідний потужний рушій, здатний перетворювати вхідні дані на структуровану сітку.

Механізми введення

Програмне забезпечення повинно надавати користувачам гнучкі способи введення слів для кросворду.

Введення слів вручну: Система має бути простою та зрозумілою для користувачів, щоб вони могли вводити список слів і підказок до них самостійно. Під час введення система повинна одразу перевіряти дані: чи слова справді існують в українській мові, чи немає повторів або помилок у написанні.

Вибір слів за темою: Користувачі повинні мати можливість обрати тему (наприклад, "Історія України", "Українські письменники", "Терміни з біології"). Далі система автоматично підбиратиме список слів, пов'язаних із цією темою, використовуючи сучасні технології обробки тексту. Користувачі зможуть переглянути цей список, обрати потрібні слова та внести корективи перед додаванням їх до кросворду. Для цього в системі потрібно створити складний модуль, що відповідатиме за тематичний добір слів.

Алгоритми побудови сітки

Головний механізм роботи повинен використовувати розумні алгоритми, які вміють правильно розміщувати слова у кросворді так, щоб вони добре перетиналися між собою і утворювали гарну цілісну картину. Основний метод для цього — поетапно розміщувати слова, пробуючи різні варіанти, змішуючи їх список і вибираючи найбільш підходящі позиції. Щоб вийшов якісний кросворд, потрібно додатково перевіряти і покращувати результат за допомогою спеціальних правил або показників. Наприклад, важливо, щоб було заповнено якнайбільше клітинок, сітка мала зручну форму (наближену до квадрата), а слова були достатньо довгими.

Кожне нове слово має розміщуватися там, де воно найкраще відповідає критеріям і де загальний вигляд кросворда буде найбільш вдалим. Якщо алгоритми недостатньо продумані, сітка може вийти незручною або погано пов’язаною між собою, що лише ускладнить розгадування і засмутить людину. Тому важливо продумати збалансований підхід: оцінювати не тільки, чи вмістилися слова, а й чи вони добре перетинаються, чи сітка виглядає охайно,

і чи її зручно читати та розгадувати. Такий ретельний підхід допоможе створювати дійсно цікаві й якісні кросворди, які принесуть задоволення як їх авторам, так і тим, хто їх розгадує.

Принципи дизайну сітки

Щоб створити якісні та зручні для розгадування кросворди, потрібно дотримуватися кількох простих правил:

1. Симетрія: Кросворд має бути симетричним, щоб виглядати гармонійно. Наприклад, якщо ви повернете сітку на 180 градусів, вона повинна виглядати так само. Це стандарт у професійних кросвордах, як в американських виданнях (наприклад, New York Times). Симетрія робить головоломку більш приємною на вигляд і справедливою.

2. Довжина слів: Усі слова в кросворді повинні бути не менше трьох літер. Кожна літера має зустрічатися як у горизонтальному, так і у вертикальному слові, щоб не було "ізольованих" літер. Короткі слова з двох літер зазвичай не використовуються, але для українських кросвордів це правило можна змінити за потреби.

3. Чорні квадрати: Ці квадрати відокремлюють слова в сітці. Їх не повинно бути занадто багато (наприклад, до 16% від усієї сітки для американських кросвордів). Великих груп чорних квадратів, як блоки 3x3, краще уникати, бо вони виглядають негарно.

4. Переплетені слова: Усі слова в сітці повинні бути пов'язані між собою. Не можна допускати окремих "островів" слів або вузьких "коридорів" з однієї-двох клітинок, бо це може зробити головоломку нецікавою або надто складною для розгадування.

Сітка для кросвордів має бути налаштованою за розміром. Найчастіше використовують непарні розміри, як-от 15x15 для щоденних кросвордів або 21x21 чи 23x23 для більших, наприклад, недільних. При створенні сітки важливо дотримуватися правил: симетрія, переплетення слів, мінімальна довжина слів і правильне розташування чорних квадратів. Це не лише робить сітку красивішою, а й впливає на те, наскільки кросворд буде цікавим і

професійним. Якщо ці правила ігнорувати, кросворд може вийти незручним, нудним і виглядати недопрацьованим, через що людям не захочеться його розв'язувати. Тому алгоритм створення сітки повинен обов’язково враховувати ці правила, щоб отримати якісний результат, який відповідатиме стандартам.

Керування та генерація підказок

Успішна робота генератора кросвордів залежить від того, як добре можна управляти підказками до слів. Ось основні аспекти цього:

Ручне введення та редагування підказок:

Користувачі повинні самі вводити підказки для слів чи змінювати існуючі. Це потрібно, коли слова вводяться вручну або коли потрібно виправити автоматично створені підказки, щоб вони краще підходили під тему чи стиль кросворда.

Автоматичне створення підказок:

Програма повинна вміти автоматично створювати підказки за допомогою штучного інтелекту. Цей штучний інтелект враховує контекст, тему й навіть цікавинки у формулюванні запитань. Для покращення підказок використовується інший метод, де:

- Один "агент" створює підказку.

- Другий намагається її розгадати.

- А третій уточнює підказку, якщо щось не сходиться.

Цей процес повторюється, поки підказки не стануть чіткими, цікавими та відповідними темі.

Особливо важливо створювати підказки, які не просто пояснюють слово, а враховують особливості мови (наприклад, української) та її культурний контекст. Такі підказки додають глибини кросворду, роблять його цікавішим і піднімають задоволення від гри.

Завдяки цьому генератор стає не просто інструментом для введення слів, а вашим розумним і креативним помічником.

Тематичний вибір слів та інтеграція

Однією з головних особливостей цього генератора є можливість вибирати слова за темою.

Користувачі можуть вказати загальну тему (наприклад, "Українське народне мистецтво" або "Космос") чи додати конкретні слова, які вони хочуть бачити в кросворді.

Щоб знайти слова, які пов’язані з цією темою, система використовує сучасні мовні технології. Вона аналізує значення слів через спеціальні моделі, що розроблені саме для української мови (наприклад, fastText або MPNet-base). У цих моделях слова "перетворюються" на числа і розташовуються так, що схожі слова розміщені близько одне до одного. Завдяки цьому можна легко знайти слова, які мають схожий зміст.

Також можна використовувати інші способи, наприклад, аналіз великих текстів для виявлення прихованих ідей, які можуть бути корисними для створення тематичного кросворду. Дуже важливо, щоб система використовувала саме українські моделі мови, оскільки загальні або багатомовні моделі можуть не враховувати особливості української лексики і культури. Це допоможе краще відображати значення слів і зв’язок між ними.

Коли система знаходить слова за темою, вони стають основою для створення кросворду. Ці слова розміщуються першими, а інші слова вже додаються навколо них, формуючи повну сітку.

Користувацький інтерфейс (UI) та користувацький досвід (UX)

Зручний та інтуїтивно зрозумілий інтерфейс є запорукою успіху програмного забезпечення.

Зручний дизайн: Інтерфейс програми має бути простим і зрозумілим, щоб користувачі могли легко вводити слова, підказки та отримувати чіткий зворотний зв'язок під час створення кросворду.

Інтерактивний редактор сітки: Головна частина програми — це візуальний редактор, який дозволяє працювати зі словами в реальному часі. Користувачі можуть одразу бачити, як слова перетинаються, змінювати їхнє розташування і переглядати варіанти заповнення для окремих місць або всієї

сітки. Такий підхід дає можливість відчувати контроль над процесом, на відміну від автоматичних генераторів, де користувачі не завжди розуміють, як все працює.

Налаштування кросворду: Користувачі мають можливість змінювати різні елементи вигляду, наприклад, кольори, шрифти, розмір сітки або навіть додавати фон.

Банк слів: Корисним доповненням може стати спеціальний банк слів, який допоможе користувачам, особливо в освітніх цілях або для дітей, легше знаходити відповіді.

Керування кросвордами

Ефективне керування створеними кросвордами є важливим для продуктивності користувача.

Збереження, завантаження та керування кросвордами: Користувачі повинні мати можливість безпечно зберігати створені кросворди, завантажувати їх для майбутніх сеансів редагування та ефективно керувати колекцією своїх кросвордів. Щоб забезпечити гнучкість, слід розглянути варіанти як локального (наприклад, зберігання на основі браузера), так і хмарного зберігання.

Параметри експорту:

Друковані формати: Програмне забезпечення повинно підтримувати експорт кросвордів у широко використовувані друковані формати, такі як PDF та поширені формати зображень (JPG, PNG, TIFF, BMP). Крім того, експорт у формат Rich Text Format (RTF) та документи Microsoft Word був би цінним для користувачів, які бажають виконати подальше редагування або інтеграцію в інші документи.

Цифрові формати: Підтримка стандартних форматів файлів для кросвордів дуже важлива, щоб вони працювали на різних програмах. Сюди входять формати, як-от Across Lite (.puz), XML, JPZ, а також відкритий формат ipuz (заснований на JSON). Вибір цих форматів впливає не лише на зручність,

але й на те, як довго кросворди залишатимуться доступними та сумісними з іншими програмами.

Підтримка популярних і відкритих форматів, як-от .puz, XML чи JPZ, дає змогу легко переносити кросворди в інші програми, ділитися ними онлайн або навіть використовувати їх в інших системах. Таким чином, користувачі не будуть "прив'язані" до конкретного програмного забезпечення. Це робить створені кросворди більш універсальними, зручними у використанні та гарантує, що вони залишатимуться доступними у майбутньому.

Онлайн-обмін: Зараз дуже популярною і корисною функцією у програмах для створення кросвордів є можливість отримати посилання для їх поширення або вставляти кросворди прямо на вебсайти. Це дає змогу легше ділитися ними, працювати разом і привертати більше уваги до створених кросвордів.

Створення складних генераторів кросвордів потребує добре продуманої системи, яка поєднує продумане програмування та сучасні технології, що допомагають працювати з текстами.

Основні алгоритми генерації кросвордів

Ітеративне розміщення з оптимізацією: Основний підхід повинен включати ітеративний алгоритм розміщення, де слова вибираються зі списку (перемішаного для випадковості або пріоритетного для тематичних слів) і розміщуються в сітці по одному. Перше слово може бути розміщено центрально, а наступні слова розташовуються для максимізації перетинів.

Принципи задачі задоволення обмежень (CSP): Генерація кросвордів може бути формально змодельована як задача задоволення обмежень (CSP), де змінні представляють слова, які потрібно заповнити, їхні області є можливими позиціями та орієнтаціями в сітці, а обмеження включають довжину слова, відповідність словнику та перетини літер у точках перетину. Хоча чистий підхід CSP може бути обчислювально інтенсивним, інтеграція принципів CSP (наприклад, перевірка обмежень та обмежений зворотний

відхід на кожному кроці розміщення) може значно покращити якість та надійність рішення.

Оцінка та повторна генерація: Потрібно створити функцію, яка оцінюватиме якість сіток за такими критеріями: скільки в сітці заповнених клітин у порівнянні з порожніми, наскільки форма сітки близька до квадрата, і загальна організованість переплетених слів. План такий: генеруємо кілька сіток і вибираємо найкращу за оцінкою. Для цього можна розглянути всі можливі способи додавання нового слова та вибрати той, який покращить оцінку.

Тут є дві стратегії: або поступово додавати слова за правилами, які виглядають просто, або використовувати більш хитрий підхід, що схожий на вирішення задач з обмеженнями (CSP). У цьому є певний компроміс: перший спосіб простіший, але може дати не ідеальний результат; другий – складніший, зате може знайти більш якісне рішення.

Можна також спробувати поєднати ці два підходи. Наприклад, додавати слова поступово, але перевіряти правила та виправляти помилки за допомогою більш інтелектуальних перевірок. Це важливо, бо із сітками українською мовою є додаткові труднощі, зокрема через мовні правила та потребу враховувати тематичний зміст. Тому потрібна розумніша стратегія, ніж звичайне поступове додавання слів.

Використання українських лексичних ресурсів

Словники: Доступ до всеосяжних українських словників є фундаментальним для перевірки слів, отримання визначень для підказок та розширення тематичних списків слів. Такі ресурси, як "Словник сентиментів для української мови" або "Словник наголосів в українській мові" , можуть надавати цінну лінгвістичну метадані, хоча першочерговою вимогою є загальний тлумачний словник.

Морфологічні аналізатори: Необхідні для обробки багатої морфології української мови. Такі інструменти, як nlp-uk (на основі словника VESUM та рушія LanguageTool), Pymorphy2 (підтримує російську та українську) та Stanza

(бібліотека Стенфорда, що використовує корпус UD), пропонують критично важливі функціональні можливості, такі як токенізація, лематизація (зведення слів до їхньої базової форми) та аналіз частин мови (POS). Це дозволяє системі точно розпізнавати різні відмінкові форми слова як належні до однієї лексичної одиниці, що є вирішальним для словникових запитів та тематичного зіставлення.

Вбудовування слів: Критично важливі для забезпечення семантичного пошуку та надання інтелектуальних тематичних пропозицій слів. Доступні високоякісні, великі набори даних для навчання українських векторних представлень слів (наприклад, lang-uk/ukr-paraphrase-multilingual-mpnet-base , моделі fastText, навчені на різних жанрах). Ці ресурси дозволяють обчислювати семантичну подібність (наприклад, за допомогою косинусної подібності) між словами та визначеними користувачем темами. Наявність та ефективна інтеграція спеціалізованих українських ресурсів НЛП (словників, морфологічних аналізаторів, вбудовувань слів) є критично важливими передумовами для цього проєкту. Без них функціональні можливості "тематичної" генерації та "генерації підказок" для української мови були б сильно обмежені в точності та якості. Ключовий виклик полягає не лише в ідентифікації цих ресурсів, а й у їхній безшовній інтеграції в конвеєр генерації кросвордів, враховуючи, що деякі з них є відкритим вихідним кодом, а інші можуть вимагати доступу до API з відповідними обмеженнями використання або витратами (наприклад, Lingvanex API). Це підкреслює необхідність ретельного вибору ресурсів та стратегічної інтеграції.

Інтеграція з українським WordNet або подібними семантичними мережами

WordNet — це лексична база даних, яка організує слова в "синсети" (набори синонімів, що представляють поняття) та визначає різні семантичні зв'язки між цими поняттями (наприклад, гіперонімія, гіпонімія, меронімія). Ukrajinet 1.0 є першою версією українського WordNet з відкритим вихідним кодом, яка наразі зосереджена на галузі фізики, але активно розробляється в

рамках ширшого проєкту Open Multilingual WordNet (OMWUkrajinet — це важливий проєкт, який допомагає створювати семантичну мережу для української мови. Хоча він поки що перебуває на ранньому етапі розвитку і охоплює не всі теми, цей ресурс уже є корисним фундаментом. Відкрите вихідне код проєкту дозволяє легко використовувати його для розробки програмного забезпечення, що значно економить час і сили, а також сприяє кращому розумінню зв'язків між словами і темами. Важливо побудувати систему так, щоб вона ставала ще більш ефективною у процесі вдосконалення та розвитку.

2 ПРОЕКТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ СТВОРЕННЯ ТЕМАТИЧНИХ КРОСВОРДІВ

2.1 Проєктування потоків даних

Діаграми потоків даних допомагають візуалізувати, як інформація переміщується всередині системи та між нею, а також зовнішніми сутностями. Ми створимо дві діаграми:

Контекстна діаграма (рівень 0): Показує систему як єдиний процес, взаємодіючий із зовнішніми сутностями.

Діаграма верхнього рівня (рівень 1): Деталізує основні процеси всередині системи, показуючи потоки даних між ними.

Контекстна діаграма (рівень 0)

Ця діаграма відображає "Генератор Тематичних Кросвордів" як одну центральну систему, яка взаємодіє з Користувачем та потенційно з Зовнішніми Лінгвістичними Базами Даних/API.

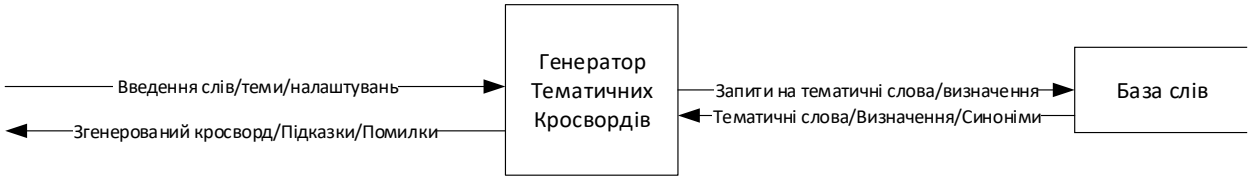

Рисунок 2.1 – Контекстна діаграма

Користувач: Зовнішня сутність, яка взаємодіє з системою.

Генератор Тематичних Кросвордів: Центральна система, яку ми розробляємо.

Зовнішні Лінгвістичні Бази Даних/API: Зовнішні системи (наприклад, словники, тезауруси, сервіси вбудовування слів, LLM API), які надають дані для тематичного пошуку та генерації підказок.

Потоки даних:

Введення слів/теми/налаштувань: Користувач надає слова для кросворду (довільні), визначає тему (ключові слова, категорії, приклади слів) та встановлює параметри генерації (розмір сітки, складність).

Згенерований кросворд/Підказки/Помилки: Система повертає згенерований кросворд (сітка, слова, підказки), повідомлення про помилки або попередження.

Запити на тематичні слова/визначення: Система надсилає запити до зовнішніх джерел для пошуку тематичних слів, отримання визначень або синонімів.

Тематичні слова/Визначення/Синоніми: Зовнішні джерела надають відповідні дані для обробки системою.

Діаграма верхнього рівня (рівень 1)

Ця діаграма деталізує основні процеси всередині вашого "Генератора Тематичних Кросвордів", показуючи, як дані переміщуються між ними.

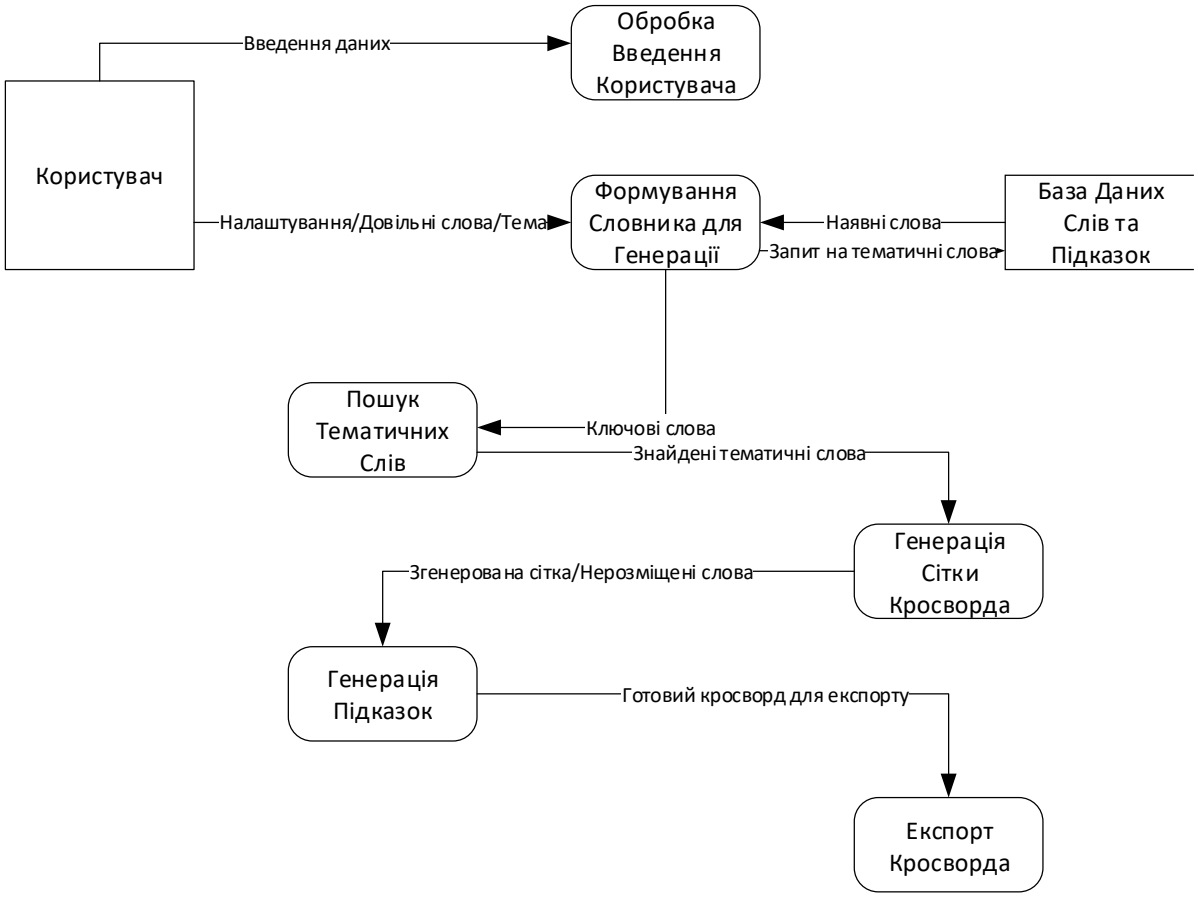

Рисунок 2.2 – Діаграма потоків даних

Користувач: Зовнішня сутність.

Зовнішні Лінгвістичні Ресурси/API: Зовнішня сутність (словники, тезауруси, WordNet, API для вбудовування слів, LLM API).

Обробка Введення Користувача: Процес, що приймає дані від користувача (довільні слова, тему, налаштування розміру, складності тощо).

Формування Словника для Генерації: Процес, що об'єднує довільні слова користувача та слова, знайдені за темою, фільтрує їх за довжиною, частотою, морфологією та готує фінальний список слів для розміщення в сітці. Може також автоматично генерувати або доповнювати підказки.

Пошук Тематичних Слів: Процес, відповідальний за взаємодію із зовнішніми лінгвістичними джерелами для знаходження слів, пов'язаних із заданою темою. Використовує ключові слова, категорії, приклади слів для семантичного пошуку.

Генерація Сітки Кросворда: Основний процес, який розміщує слова у сітці, використовуючи алгоритми розміщення слів та враховуючи обмеження (перетини, розмір сітки, відсутність ізольованих літер).

Генерація Підказок: Процес, який створює підказки для кожного слова кросворда. Може використовувати автоматичні методи (ШІ/LLM, словникові визначення, шаблони) або брати підказки, надані користувачем.

Експорт Кросворда: Процес, який форматує та зберігає готовий кросворд у різних форматах (PDF, HTML, PNG, .puz тощо).

Сховища даних:

База Даних Проекту: Зберігає поточний стан кросворда, над яким працює користувач: налаштування, довільні слова, визначену тему, згенеровану сітку та фінальні слова з підказками. Може зберігати кілька проектів.

База Даних Слів та Підказок: Основне сховище для словникового запасу системи. Містить слова (з лемами, частинами мови, частотою), їхні визначення, синоніми, тематичні теги та, можливо, вектори вбудовування слів для семантичного пошуку. Це внутрішнє сховище, яке може наповнюватися та оновлюватися.

Потоки даних:

Введення даних: Користувач вводить налаштування, довільні слова та визначає тему.

Налаштування/Довільні слова/Тема: Збереження введених користувачем даних у Базу Даних Проекту.

Наявні слова/Налаштування: Завантаження слів та налаштувань проекту з Бази Даних Проекту для подальшої обробки.

Запит на тематичні слова: Запит до процесу пошуку тематичних слів на основі теми.

Ключові слова/Приклади: Ключові слова теми передаються до зовнішніх лінгвістичних ресурсів.

Знайдені тематичні слова/Семантичні зв'язки: Результати пошуку тематичних слів та їхні семантичні зв'язки повертаються.

Тематичні слова/Їхні визначення: Знайдені тематичні слова та їхні визначення передаються для формування фінального словника.

Фінальний словник слів та їхніх підказок: Підготовлений список слів з підказками передається до процесу генерації сітки.

Згенерована сітка/Нерозміщені слова: Результати генерації сітки (успішна сітка та слова, які не вдалося розмістити) передаються.

Слова/Підказки: Фінальні слова та підказки передаються для зберігання.

Дані для підказок: Запит до внутрішньої Бази Даних Слів та Підказок для отримання даних, необхідних для генерації підказок.

Фінальний кросворд (сітка, слова, підказки): Повністю сформований кросворд зберігається в Базі Даних Проекту.

Готовий кросворд для експорту: Завантаження даних готового кросворда для експорту.

Згенерований файл кросворда: Експортований файл кросворда надається користувачеві.

Дані сітки: Збереження згенерованої сітки кросворда.

Запити на слова/визначення: Запити від Бази Даних Слів та Підказок до зовнішніх ресурсів для наповнення або оновлення власних даних.

Визначення/Семантичні дані: Повернення визначень та семантичних даних для Бази Даних Слів та Підказок.

Ці діаграми надають чітке візуальне представлення того, як дані будуть текти через генератор кросвордів, що є чудовою відправною точкою для подальшого детального проектування та розробки!

2.2 Проєктування діаграми використання

Діаграма сценаріїв використання демонструє взаємодію між основними акторами та функціями (варіантами використання) системи "Генератор Тематичних Кросвордів". Вона візуалізує, хто взаємодіє із системою та які основні дії вони можуть виконувати.

Актори — це зовнішні сутності, які взаємодіють із системою. Вони можуть бути людьми (користувачами) або іншими системами. На цій діаграмі представлені два основні актори:

Користувач: Основний користувач програмного забезпечення. Це може бути викладач, творець контенту, ентузіаст кросвордів або будь-яка інша особа, яка бажає створити кросворд. Користувач ініціює дії та отримує результати від системи.

Зовнішні Лінгвістичні Бази Даних/API: Це інша система, з якою взаємодіє "Генератор Тематичних Кросвордів". Вона надає необхідні лінгвістичні дані, такі як словникові визначення, синоніми або семантично пов'язані слова, що використовуються для тематичного підбору слів та генерації підказок.

Прямокутник з пунктирною рамкою представляє собою межі системи. У цьому випадку це "Генератор Тематичних Кросвордів". Усе, що знаходиться всередині цих меж, є частиною системи, яку ми розробляємо.

Варіанти Використання (Use Cases)

Варіанти використання — це функції або послуги, які система надає акторам. Вони описують, що система робить для актора, не деталізуючи, як саме це робиться. Кожен варіант використання представлений овалом:

Введення слів/теми/налаштувань: Цей варіант використання дозволяє Користувачу надавати системі слова для кросворду (довільні або за темою), вказувати бажану тему та налаштовувати параметри генерації (наприклад, розмір сітки, складність). Цей варіант використання також може опосередковано взаємодіяти із Зовнішніми Лінгвістичними Базами Даних/API для отримання пропозицій тематичних слів.

Генерація кросворду та підказок: Основний варіант використання, який відповідає за створення самого кросворду. Система бере слова та налаштування, генерує сітку, розміщує слова, а також створює підказки для кожного слова. Цей процес активно використовує дані із Зовнішніх Лінгвістичних Баз Даних/API для пошуку тематичних слів та генерації контекстуальних підказок.

Керування кросвордами (збереження/експорт): Цей варіант використання дозволяє Користувачу зберігати створені кросворди, завантажувати їх для подальшого редагування та експортувати у різних форматах (наприклад, PDF, зображення, спеціалізовані формати кросвордів), а також потенційно ділитися ними онлайн.

Лінії між акторами та варіантами використання називаються асоціаціями. Вони показують, який актор взаємодіє з яким варіантом використання. Напрямок стрілки вказує на ініціатора взаємодії або потік даних.

Лінії від Користувача до варіантів використання вказують, що Користувач ініціює ці дії.

Лінії між Зовнішніми Лінгвістичними Базами Даних/API та варіантами використання відображають обмін інформацією, необхідною для функціонування генератора кросвордів (наприклад, запити на дані та отримання даних).

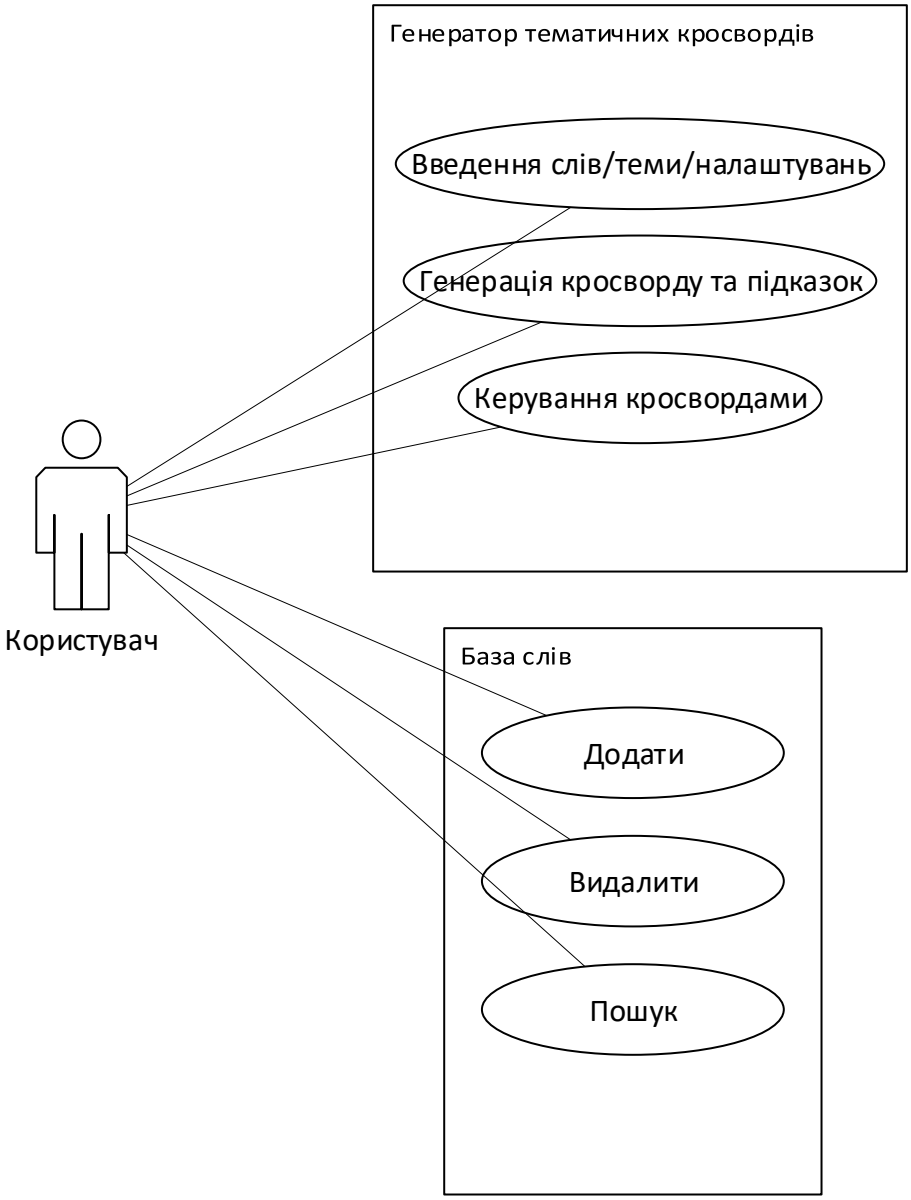

Рисунок 2.3 – Діаграма сценаріїв використання

2.3 Проєктування основних алгоритмів

Задача основного алгоритму нашої системи – взяти наданий список слів (як вручну введених, так і тематично підібраних) та інтелектуально розмістити їх у сітці таким чином, щоб вони ефективно перетиналися та формували цілісну головоломку, яка відповідає професійним стандартам.

1. Ітеративне розміщення слів

Це фундамент алгоритму. Слова розміщуються в сітці одне за одним.

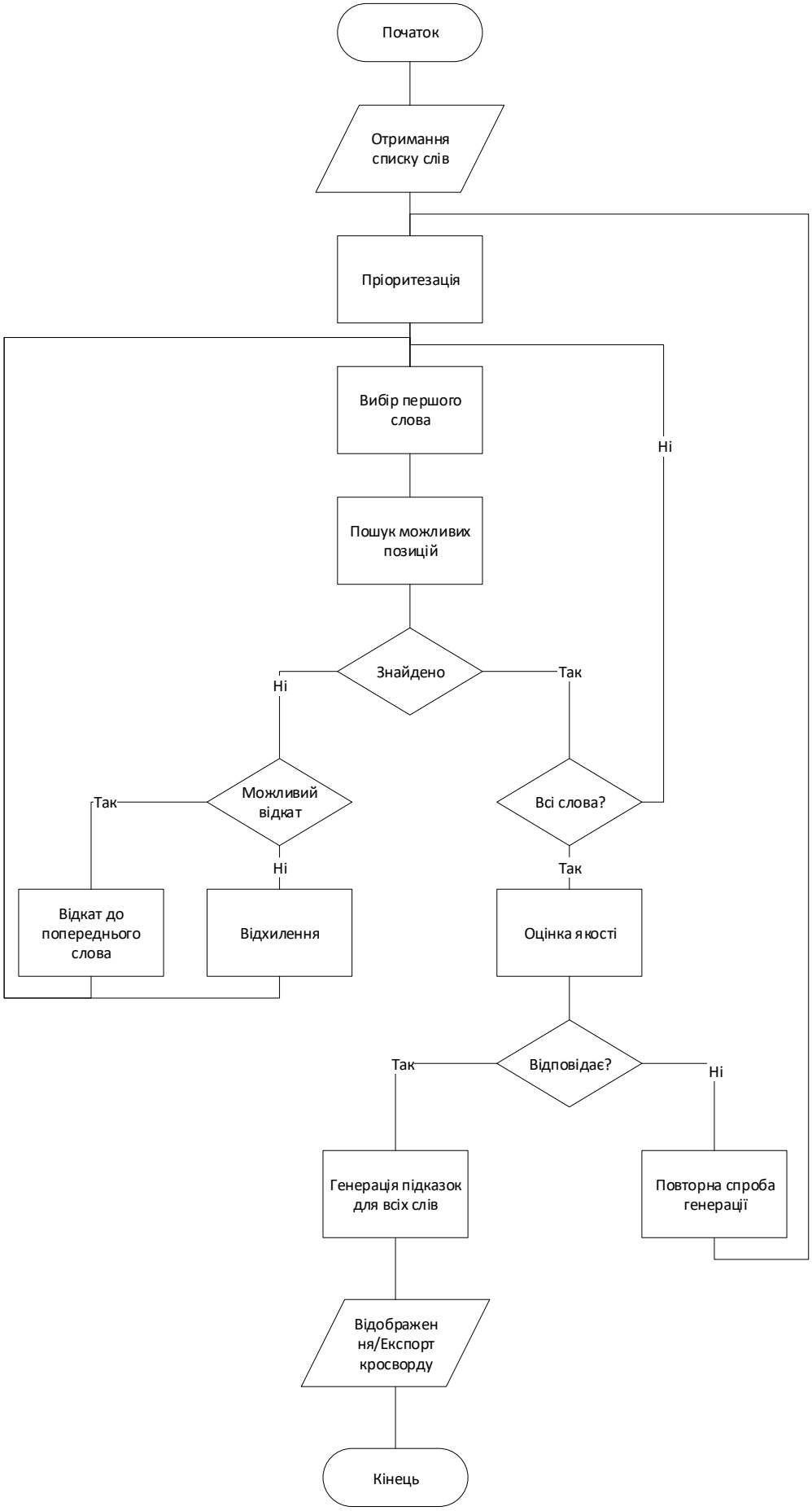

Рисунок 2.4 – Блок-схема основного алгоритму

Вибір першого слова: Перше слово (часто ключове або найдовше, або тематичне) може бути розміщене в центрі сітки. Це створює початкову "основу" для головоломки.

Пріоритезація слів: Якщо ви маєте тематичні слова, вони зазвичай отримують пріоритет. Алгоритм спочатку намагається розмістити ці слова, оскільки вони є центральними для теми кросворду. Список слів також може бути перемішаний, щоб уникнути одноманітних або передбачуваних шаблонів.

Послідовне розміщення: Для кожного наступного слова система шукає можливі позиції та орієнтації (горизонтально або вертикально), де слово може перетинатися з уже розміщеними словами. Мета — максимізувати кількість перетинів та забезпечити, щоб кожна літера перетиналась принаймні один раз ("перевірені квадрати").

2. Застосування принципів задоволення обмежень

Генерація кросворду може бути формально розглянута як задача задоволення обмежень. Це означає, що існує набір змінних (слів, які потрібно розмістити) з їхніми областями (можливі позиції та орієнтації в сітці) та обмеженнями, які необхідно задовольнити.

Перевірка обмежень: На кожному кроці розміщення нового слова алгоритм перевіряє низку обмежень:

Довжина слова: Чи відповідає слово кількості доступних квадратів?

Відповідність словнику: Чи є це дійсне українське слово?

Перетини літер: Чи збігаються літери в точках перетину? Наприклад, якщо слово "ХЛІБ" перетинається зі словом "МІСТ", то буква "І" повинна бути однаковою в обох словах у точці перетину.

Обмежений зворотний відхід: Якщо слово не може бути розміщено відповідно до всіх обмежень або якщо розміщення призводить до "тупика" (інші слова не можуть бути розміщені пізніше), алгоритм може спробувати іншу позицію для поточного слова або навіть "відкотитися" до попереднього кроку та спробувати інший варіант.

3. Оцінка та повторна генерація сітки (Оптимізація)

Недостатньо просто розмістити слова; важливо, щоб отримана сітка була високоякісною.

Показники якості: Кожна згенерована сітка оцінюється за кількома показниками:

Щільність сітки: Співвідношення заповнених квадратів до загальної кількості квадратів. Вища щільність зазвичай краща.

Квадратність (співвідношення сторін): Співвідношення ширини до висоти сітки. Ідеально, якщо це співвідношення ближче до 1.0 для більш квадратної форми.

Якість переплетення: Наскільки добре слова перетинаються по всій сітці, уникнення "ізольованих" секцій або вузьких "перешийків", які ускладнюють розгадування.

Розподіл чорних квадратів: Чорні квадрати повинні бути розумно та симетрично розподілені, без великих, непривабливих блоків (наприклад, 3x3).

Мінімальна довжина слів: Всі слова повинні бути щонайменше трьох літер (або іншої визначеної мінімальної довжини).

Симетрія: Дотримання 180-градусної обертальної симетрії (для професійного вигляду).

Стратегія оптимізації: Алгоритм може генерувати кілька варіантів сіток (або досліджувати всі можливі розміщення для кожного нового слова) і вибирати той, який отримує найвищий бал за вищезазначеними критеріями. Це забезпечує створення "найкращих" кросвордів, а не просто "можливих".

Цей комплексний підхід гарантує, що згенеровані кросворди не тільки функціональні, але й естетично привабливі та відповідають високим стандартам якості, що є ключовим для забезпечення приємного досвіду для користувача та того, хто розв'язує кросворд.

2.4 Проєктування структури бази даних генератора тематичних кросвордів

Для ефективного функціонування генератора тематичних кросвордів необхідна база даних, яка може зберігати слова, їхні підказки, тематичні категорії та лінгвістичні метадані. Нижче представлена пропонована структура бази даних з кількома таблицями, що забезпечують гнучкість та розширюваність.

Таблиця Words (Слова)

Ця таблиця зберігає основну інформацію про кожне слово, яке може бути використане в кросворді.

Words (Слова)

- word_id (PK)
- text
- lemma
- part_of_speech
- length
- frequency
- is_common_thematic_base
- semantic_vector

Таблиця Clues (Підказки)

Ця таблиця зберігає різні підказки для кожного слова. Одне слово може мати кілька підказок.

Clues (Підказки)

- clue_id (PK)
- word_id (FK)
- text
- is_auto_generated
- difficulty_rating
- contextual_theme

Таблиця Themes (Теми)

Ця таблиця містить визначені теми, за якими можна групувати слова. Таблиця WordThemes (СловаТеми) - Таблиця зв'язку

Ця таблиця реалізує зв'язок "багато-до-багатьох" між словами та темами, оскільки одне слово може належати до кількох тем, а одна тема може містити багато слів.

Themes (Теми)

- theme_id (PK)
- name
- description
- keywords

WordThemes (СловаТеми)

- word_id (FK, PK)
- theme_id (FK, PK)
- relevance_score

Таблиця UserCrosswords (Кросворди Користувачів)

UserCrosswords (Збережені Кросворди)

- crossword_id (PK)
- user_id (FK)
- name
- settings
- grid_data
- clues_data
- created_at
- last_modified
- is_public
- views_count
- solves_count

UserCrosswordSolutions (Спроби Розв'язання)

- solution_id (PK)
- crossword_id (FK)
- user_id (FK)
- solution_state
- status
- time_spent
- started_at
- completed_at

Взаємозв'язки Між Сутностями

1. **Words та Clues**:
    - **Тип**: Один-до-багатьох (One-to-Many).
    - **Опис**: Одне слово (Words) може мати **багато** підказок (Clues), але кожна підказка належить лише до **одного** слова.
    - **Зв'язок**: Words.word_id (PK) -> Clues.word_id (FK)
2. **Words та WordThemes**:
    - **Тип**: Багато-до-багатьох (Many-to-Many).
    - **Опис**: Одне слово (Words) може належати до **багатьох** тем (Themes), і одна тема може містити **багато** слів. Таблиця WordThemes є сполучною таблицею для реалізації цього зв'язку.
    - **Зв'язок**: Words.word_id (PK) -> WordThemes.word_id (FK)
3. **Themes та WordThemes**:
    - **Тип**: Багато-до-багатьох (Many-to-Many).
    - **Опис**: Одна тема (Themes) може містити **багато** слів (Words), і одне слово може належати до **багатьох** тем. WordThemes є сполучною таблицею.
    - **Зв'язок**: Themes.theme_id (PK) -> WordThemes.theme_id (FK)
4. **UserCrosswords та UserCrosswordSolutions**:
    - **Тип**: Один-до-багатьох (One-to-Many).

- **Опис**: Один збережений кросворд (UserCrosswords) може мати **багато** спроб розв'язання (UserCrosswordSolutions), але кожна спроба розв'язання стосується лише **одного** кросворду.
- **Зв'язок**: UserCrosswords.crossword_id (PK) -> UserCrosswordSolutions.crossword_id (FK)

5. **Users (гіпотетична) та UserCrosswords**:
   - **Тип**: Один-до-багатьох (One-to-Many).
   - **Опис**: Один користувач (Users) може створити **багато** кросвордів (UserCrosswords), але кожен кросворд створений лише **одним** користувачем. (Примітка: таблиця Users не була деталізована, але є імпліцитною для полів user_id.)
   - **Зв'язок**: Users.user_id (PK) -> UserCrosswords.user_id (FK)
6. **Users (гіпотетична) та UserCrosswordSolutions**:
   - **Тип**: Один-до-багатьох (One-to-Many).
   - **Опис**: Один користувач (Users) може мати **багато** спроб розв'язання кросвордів (UserCrosswordSolutions), але кожна спроба розв'язання належить лише **одному** користувачеві.
   - **Зв'язок**: Users.user_id (PK) -> UserCrosswordSolutions.user_id (FK

Для візуалізації залежностей між сутностями бази даних побудуємо ERD-діаграму (рис. 2.5).

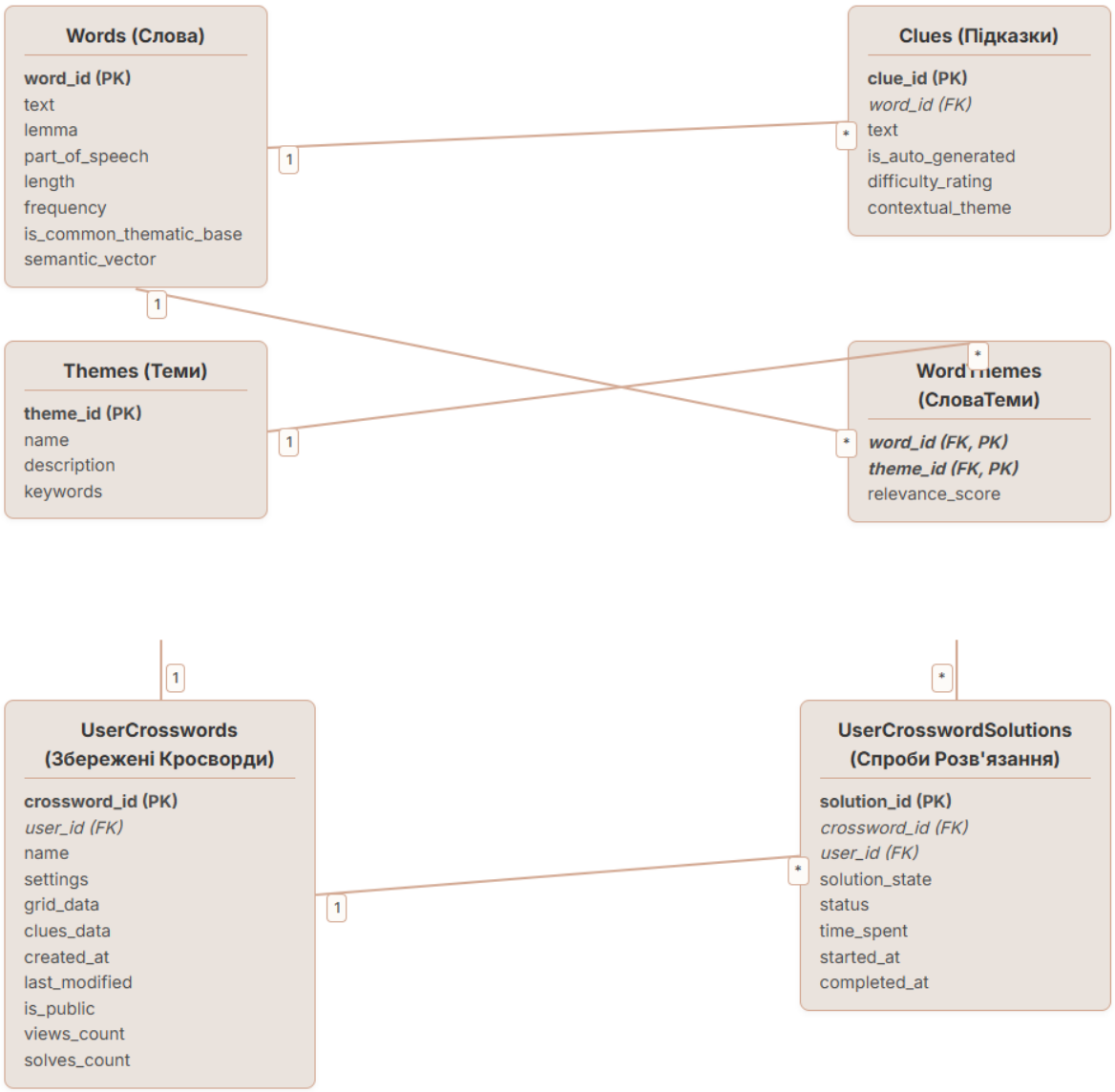

Рисунок 2.5 – ERD-діаграма

3 РЕАЛІЗАЦІЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ СТВОРЕННЯ ТЕМАТИЧНИХ КРОСВОРДІВ

3.1 Добір засобів розробки програмного забезпечення для створення тематичних кросвордів

Для реалізації застосунку "Генератор тематичних кросвордів" ми обрали наступний стек засобів розробки:

HTML (HyperText Markup Language): Це стандартна мова розмітки, яка використовується для створення основної структури веб-сторінок та їхнього контенту. HTML дозволяє визначати і організовувати текст, зображення, відео та інші елементи на сторінці, а також встановлювати зв’язки між ними. HTML є основою веб-розробки і працює у поєднанні з мовами стилів, такими як CSS (Cascading Style Sheets), для визначення зовнішнього вигляду сторінки, та з мовами програмування, такими як JavaScript, для додавання інтерактивності. HTML є ключовим інструментом у створенні сучасного веб-контенту і забезпеченні доступності сторінок для перегляду на різних пристроях.

CSS (Cascading Style Sheets): Мова стилізації, яка використовується для визначення зовнішнього вигляду та розташування HTML-елементів на веб-сторінці. За допомогою CSS можна додавати кольори, змінювати шрифти, встановлювати відступи, межі, розташовувати елементи в певному порядку, задавати анімації, адаптувати сторінку для різних екранів і пристроїв. CSS працює разом із HTML, забезпечуючи відділення візуальної частини веб-дизайну від структурного коду. Основні переваги CSS включають зручність у підтримці коду, можливість створення адаптивного дизайну і швидкість завантаження сторінки завдяки стилям, що зберігаються в зовнішніх файлах.

У цьому проекті використовується Tailwind CSS — сучасний, високоефективний CSS-фреймворк, що значно спрощує процес розробки стильового оформлення веб-сторінок. Він базується на підході утилітарного дизайну, пропонуючи набір готових класів без необхідності писати власні стилі з нуля. Tailwind CSS дозволяє розробникам інтегрувати стилі

безпосередньо в HTML-код шляхом використання коротких та інтуїтивно зрозумілих класів. Наприклад, класи `bg-slate-100` задають світло-сірий колір фону, `rounded-2xl` створює округлені кути елемента великого радіусу, а `font-bold` використовується для створення жирного шрифту. Завдяки Tailwind CSS створення адаптивного та привабливого дизайну зручне і швидке, оскільки цей фреймворк надає розробникам повний контроль над виглядом елементів, без потреби писати складні кастомні стилі. Крім того, його модульний підхід забезпечує високу гнучкість, дозволяючи легко налаштовувати конфігурацію для створення унікального стилю, що відповідає потребам конкретного проекту. Tailwind CSS підтримує інтеграцію з багатьма фронтенд-фреймворками та інструментами, такими як React, Vue, Angular, а також ефективно працює з системами збору, що забезпечує його популярність серед фронтенд-розробників у всьому світі.

Власні стилі: Невелика кількість кастомних CSS-правил у тезі <style> для специфічних елементів, таких як клітинки кросворду (.crossword-grid td) та смуга прокрутки.

JavaScript (JS): Це одна з основних мов програмування, що використовується для створення інтерактивності, динамічної логіки й функціональності фронтенду нашого застосунку. У контексті працювання над генератором кросвордів, його завдання охоплюють:

1. Маніпуляція DOM:

JavaScript дозволяє взаємодіяти з елементами HTML-документа, забезпечуючи динамічне оновлення інформації на сторінці. У випадку створення кросвордів його головна роль полягає у:

- Додаванні нових слів до списку, який користувач заповнює під час складання кросворду. Наприклад, вводячи слово у текстове поле й натискаючи кнопку "Додати", JS оновлює відповідний контейнер у DOM.

- Видаленні непотрібних слів зі списку за допомогою кнопки або контекстного меню, що дозволяє користувачу редагувати список перед створенням кросворду.

- Оновленні відображення готового кросворду, динамічно додаючи елементи сітки (клітинки) та заповнюючи їх буквами в потрібних позиціях.

2. Алгоритм генерації:

JavaScript виконує логічні обчислення для побудови кросворду. Це включає:

- Розміщення слів: JS аналізує список слів, визначає їхню довжину, знаходить можливі точки перетину між літерами різних слів і позиціює їх на сітці так, щоб вони утворювали логічну правильну структуру.

- Пошук перетинів: Скрипт порівнює літери різних слів, шукаючи спільні символи, що дозволяють інтегрувати їх у кросворд із мінімальним конфліктом.

- Обробку конфігурацій: Генератор за потреби тестує кілька варіантів побудови сітки, щоб досягти найоптимальнішого розміщення слів.

3. Обробка подій:

JS відповідає за реакцію програми на взаємодії зі сторінкою. Це реалізується через "слухачі подій", які стежать за діями користувача, наприклад:

- Натискання кнопки "Додати": Після введення слова й натискання кнопки система додає його до списку, очищує поле для вводу та оновлює список на екрані.

- Кнопка "Створити кросворд": Активує алгоритм генерації, розміщує слова на сітці, оновлює DOM-структуру, щоб показати користувачу готовий кросворд.

- Кнопка "Експорт в PDF": Дозволяє зберегти результат роботи у вигляді PDF-документа. JavaScript може інтегрувати зовнішні бібліотеки, наприклад `jsPDF`, щоб перетворити HTML-контент у формат PDF, включаючи представлену сітку кросворда.

Додаткові функції:

- Перевірка введених даних (наприклад, недопустимість порожніх строк чи занадто довгих слів).

- Динамічне масштабування сітки залежно від розміру слів у кросворді.

- Відображення повідомлень про помилки або підказок, якщо слова неможливо розмістити через обмеження простору.

Сторонні JavaScript-бібліотеки, використані в проєкті, та їх застосування:

jsPDF – бібліотека для створення PDF-файлів безпосередньо в браузері. Її основна функція — дозволити розробникам створювати документи PDF, не покладаючись на серверну обробку. Вона підтримує текст, графіку, таблиці, та навіть інтерактивні елементи, такі як гіперпосилання. У нашому проєкті jsPDF використовується для реалізації функціоналу експорту, що дає можливість користувачам безпосередньо зберігати створений контент, кросворди, у форматі PDF. Це зручно для збереження результатів роботи, друку або подальшого аналізу.

html2canvas – ця бібліотека спеціалізується на "захопленні" HTML-контенту і його перетворенні у вигляд зображення (формати PNG, JPEG тощо). Вона дозволяє створити віртуальний "скріншот" будь-якого елемента на веб-сторінці, враховуючи стильові атрибути, зображення та позицію на екрані. У нашому випадку html2canvas використовується для відображення сітки кросворду — вона "фотографує" HTML-елемент, що представляє кросворд, і перетворює його на графічний файл. Далі це графічне зображення вставляється у PDF-файл за допомогою можливостей jsPDF. Така інтеграція дозволяє зберегти зовнішній вигляд кросворду максимально наближеним до його веб-версії, незалежно від формату сторінки.

Ці дві бібліотеки працюють у тандемі, забезпечуючи функціонал експорту веб-контенту у формат PDF і дозволяючи користувачам легко зберігати необхідні дані. Разом вони є потужним і гнучким інструментом для генерації документації прямо з веб-додатка.

Бек-енд логіка, яка включає зберігання даних і алгоритм генерації кросвордів, буде реалізована з використанням JavaScript у поєднанні з React для інтерактивного представлення даних. Для зберігання слів та тем буде

використовуватися Firestore, що є частиною хмарного пакету Firebase. Ця база даних надає можливість безпечного зберігання та обробки великих обсягів даних, а також реального часу синхронізації між користувачами, що працюють у своєму браузері.

Процес генерації кросворду буде автоматизований за допомогою кастомного алгоритму, який працює над пошуком оптимального розташування слів відповідно до вибраних тем. Користувач зможе через React-платформу вибирати теми, додавати власні слова, переглядати сформовані кросворди та зберігати їх для подальшого використання. Інтеграція Firestore забезпечить надійне зберігання даних, щоб вони були доступними для редагування або створення нових кросвордів у будь-який момент.

Цей підхід дозволить створити зручний і швидкий інструмент для роботи з кросвордами прямо в браузері, сумісний із сучасними веб-технологіями, забезпечуючи інтуїтивно зрозумілий інтерфейс і динамічний досвід для користувача.

3.2 Огляд роботи генератора тематичних кросвордів

Створений нами застосунок має декілька режимів роботи:

– управління темами та словами;

– генерація кросворду;

Управління темами та словами дозволяє додавати нові тематичні категорії для кросвордів.

Після вибору теми, можна додавати слова та відповідні підказки до цієї теми.

Всі теми та слова зберігаються в базі даних Firestore, що забезпечує їх стійкість та доступність.

Відкривши обрану тему, можна переглядати, які слова вже додані до вибраної теми, та видаляти як окремі слова, так і цілі теми.

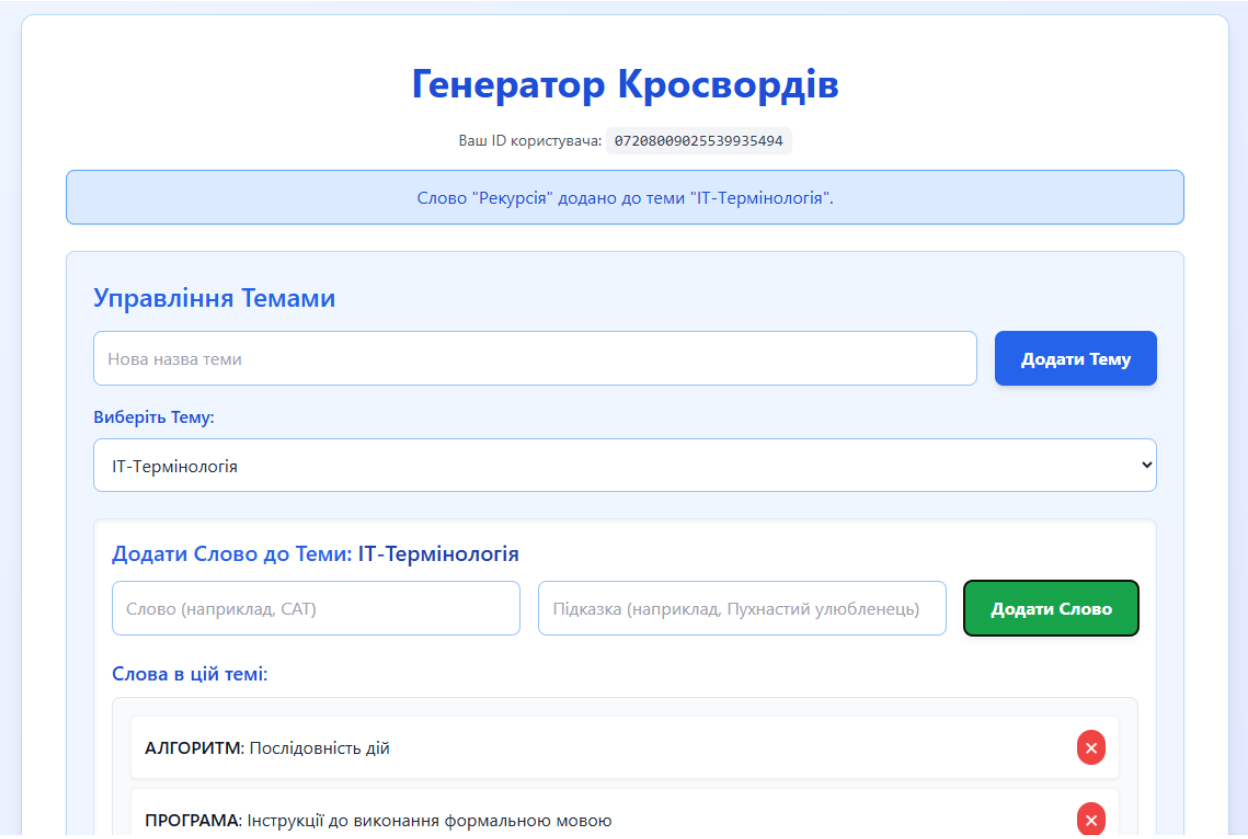

Рисунок 3.1 – Управління темами та словами

Додавання слів виконується у формі зліва. Введіть слово та підказку до нього. Натисніть кнопку "Додати слово". Слово з'явиться у списку нижче. Ви можете видалити будь-яке слово зі списку.

Генерація кросворду. Коли ви додали всі бажані слова, натисніть "Створити кросворд". Ви перейдете до форми створення кросворду.

В цій формі ви можете переглянути слова, що були раніше закріплені за певною темою. Ви можете видалити певні слова або додати нові, щоб згенерувати кросворд з потрібних вам слів.

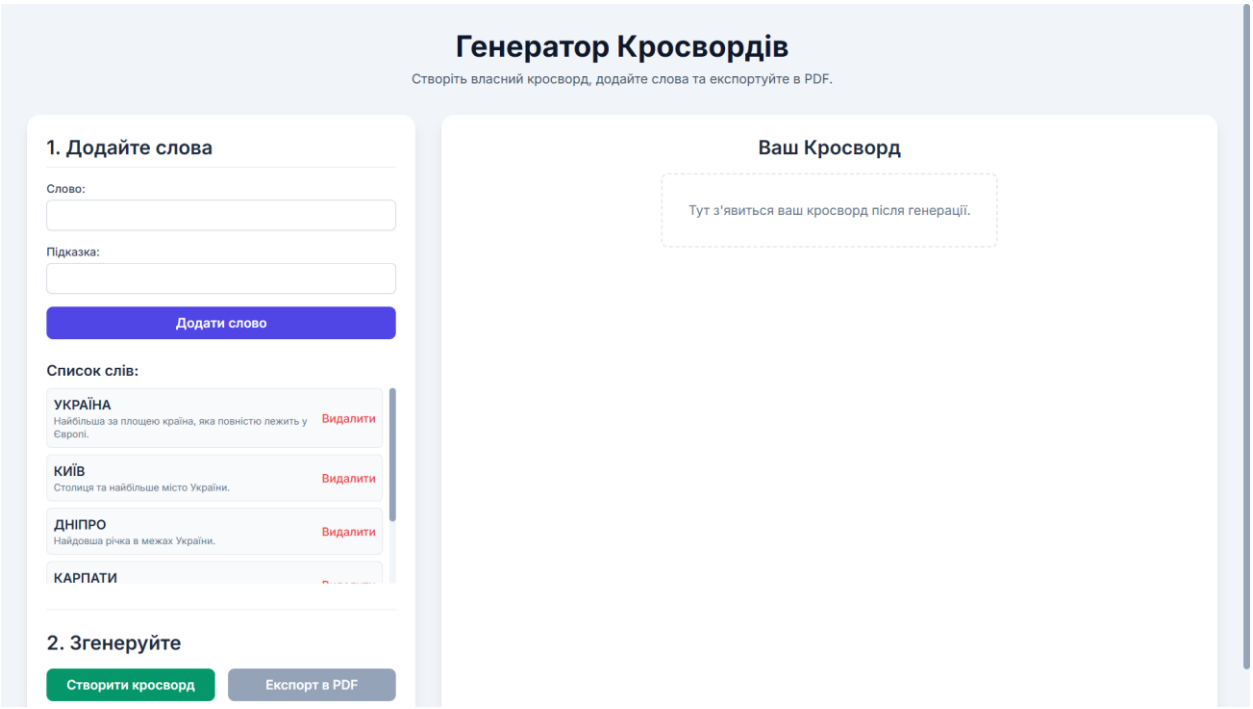

Рисунок 3.2 – Генерація кросворду

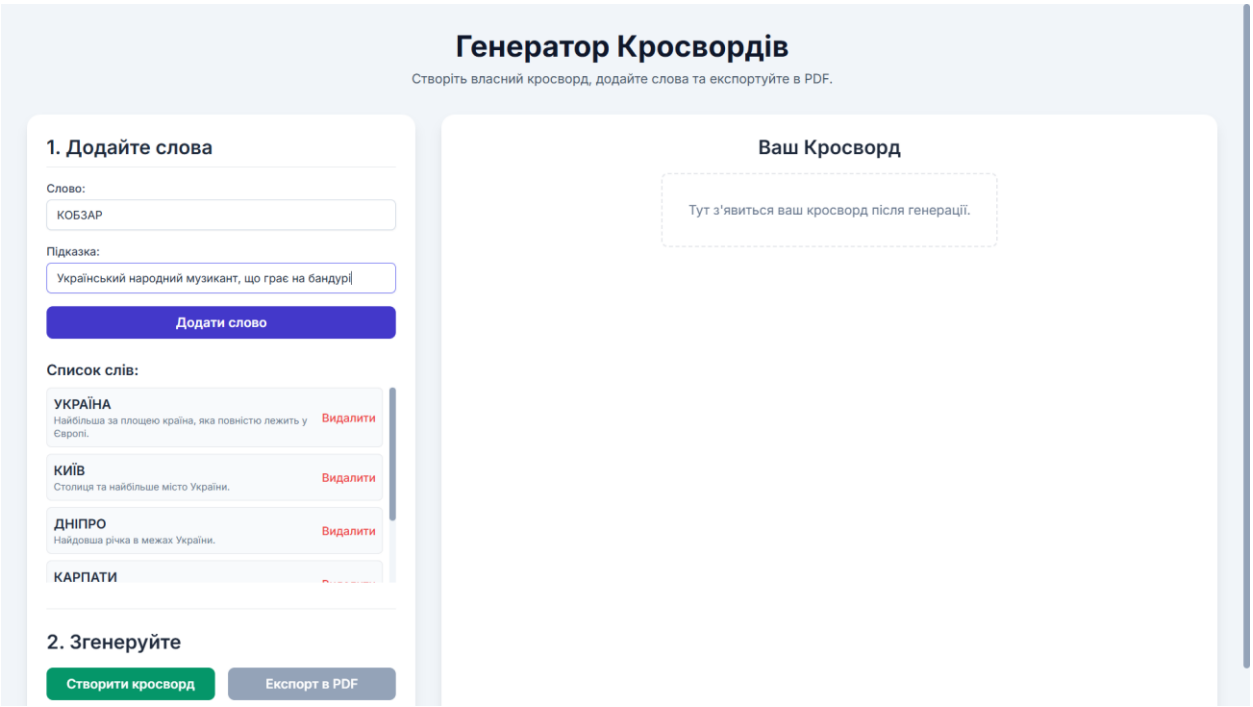

Рисунок 3.3 – Додавання нового слова до обраної теми

Коли всі слова та підказки введено, необхідно натиснути кнопку «Згенерувати». Запускається алгоритм генерації, результат роботи якого відображається в полі зліва. Кросворд можна зберегти, а також імпортувати в формат pdf, для подальшого друку та використання в різних ситуаціях.

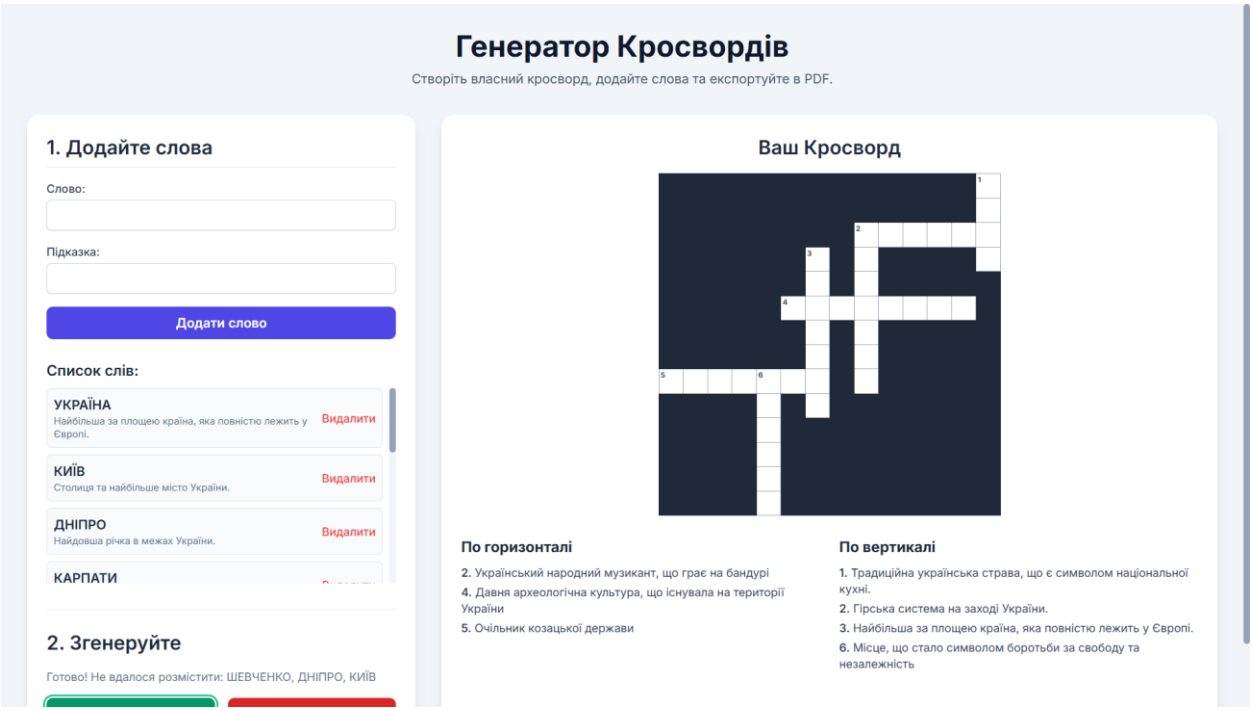

Рисунок 3.4 – Перегляд створеного кросворду

На окремій сторінці можна переглянути створені вами кросворди.

Кожен зі створених кросвордів можна переглянути та за необхідності відредагувати.

Після редагування (додавання або виключення слів) відбувається повторна генерація кросворду.

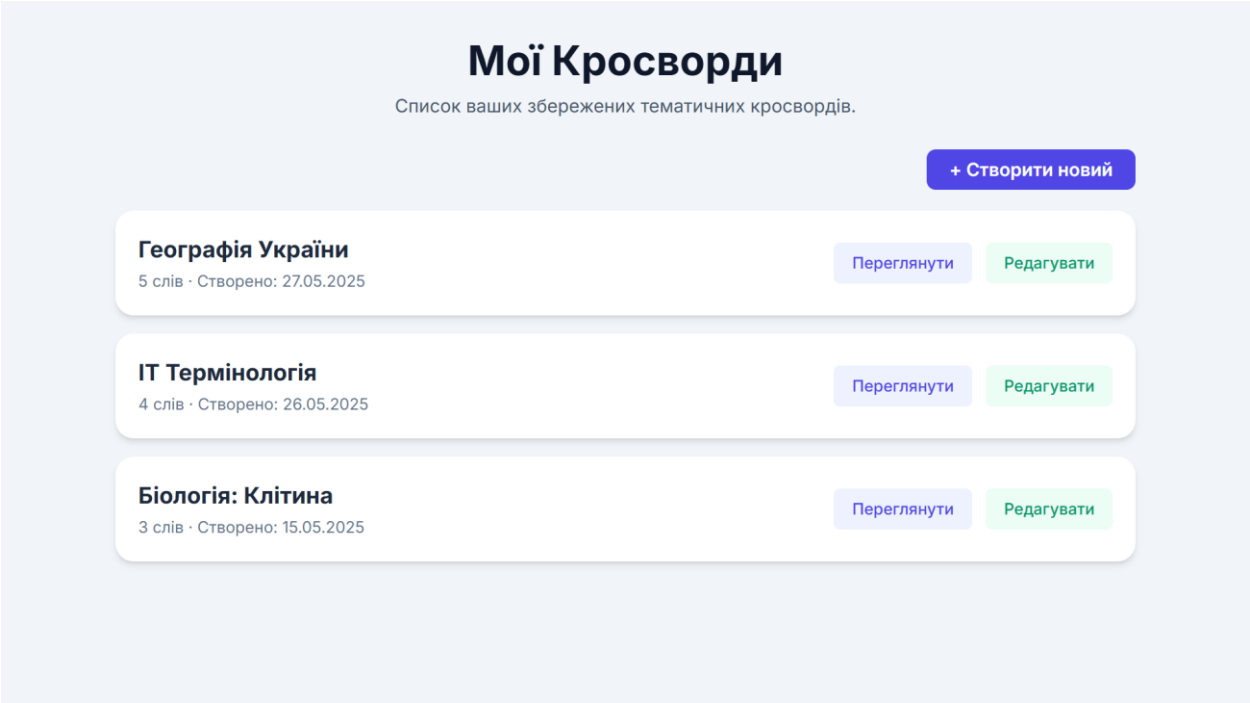

Рисунок 3.5 – Перегляд списку створених кросвордів

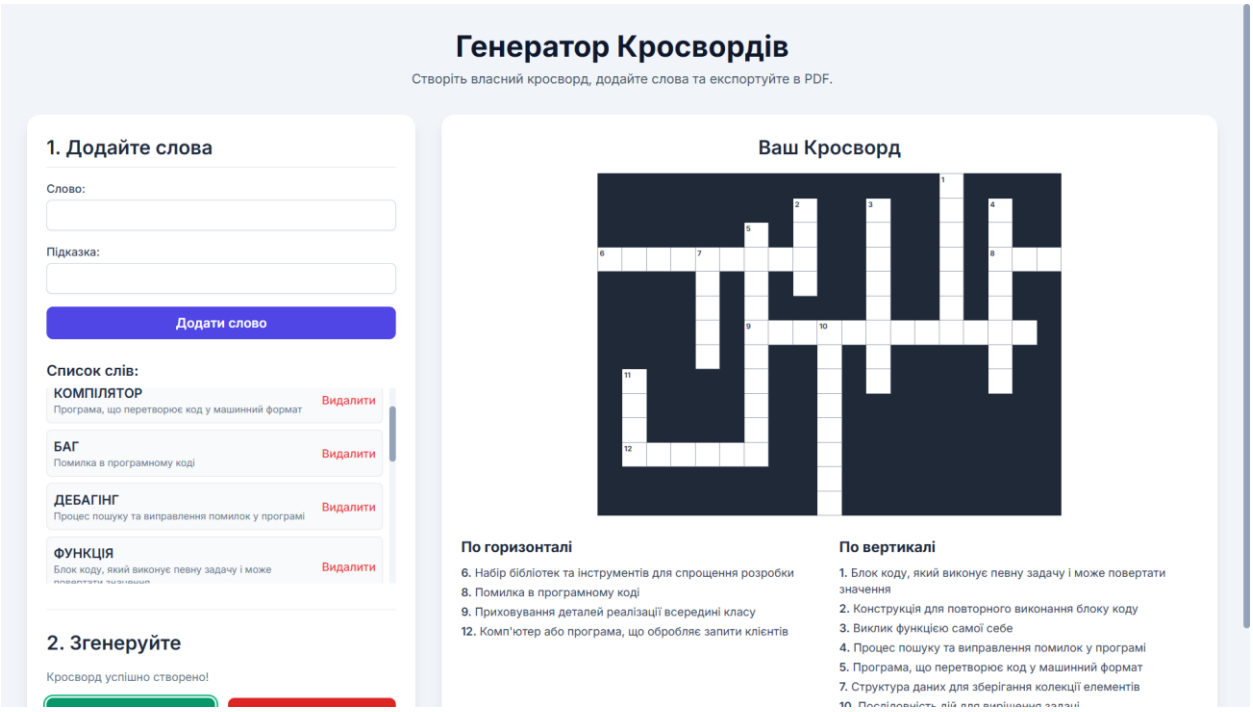

Рисунок 3.6 – Редагування створеного кросворду

Таким чином, ми розробили унікальний застосунок, який дозволяє користувачам створювати тематичні кросворди відповідно до їхніх інтересів.

У програмі передбачена можливість додавання нових тем та слів, що забезпечує необмежену варіативність для створення цікавих та інтерактивних кросвордів. Інструменти генерації автоматично формують логічно побудовані кросворди, які можна використовувати для навчання, розваг або інтелектуальних викликів.

Окрім створення, наш застосунок також містить функцію збереження готових кросвордів, що дозволяє легко організовувати та управляти своєю колекцією. Завдяки функції експортування користувачі можуть конвертувати свої кросворди у формат PDF, що ідеально підходить для друку або обміну з іншими людьми. Така багатофункціональність робить наш застосунок зручним у використанні, ефективним для будь-якої аудиторії та здатним зацікавити як любителів пазлів, так і людей, які шукають інструменти для розвитку творчих здібностей чи розширення словникового запасу.

ВИСНОВКИ

У роботі розглянуто, як кросворди впливають на наше життя, зокрема в навчанні, відпочинку та розвитку розумових здібностей. Показано, що вони допомагають краще думати, запам’ятовувати, зосереджуватись і розширювати словниковий запас. Особливо наголошено на тому, як кросворди можна використовувати в школах, щоб закріплювати знання, розвивати уяву і робити навчання більш цікавим для учнів.

У дослідженні також розповідається про історію кросвордів і те, як вони перейшли із друкованих форматів у цифрові. Завдяки цьому доступ до них став набагато простішим. Було запропоновано ідею створити програму, яка автоматично генерує тематичні кросворди. Така програма дозволила б адаптувати кросворди до різних тем, рівнів складності та інтересів користувачів, що особливо корисно для навчання.

Окрім цього, в роботі аналізуються нинішні технології створення кросвордів, їхні переваги й недоліки. На основі цього запропоновано концепцію програмного середовища, яке можна інтегрувати в освітні платформи, щоб навчання стало ще більш захопливим і допомогло учням активніше освоювати нові знання.

Відповідно до визначених вимог за допомогою сучасної платформи React та мови програмування JavaScript було розроблено інтерактивний веб-додаток, який забезпечує користувачам можливість створювати тематичні кросворди з широким функціоналом. Користувачі можуть генерувати кросворди на основі вибраних тем, включати слова з власноруч створених або завантажених словників, а також редагувати теми відповідно до своїх потреб. Окрім цього, додаток надає функцію управління словниками: додавання нових слів, видалення чи редагування вже існуючих.

Для зручності роботи з кросвордами передбачено функцію експорту готових кросвордів у формат PDF. Це дозволяє користувачам зберігати результати роботи, роздруковувати кросворди або використовувати їх у

навчальному чи розважальному процесах. Додаток побудований з використанням компонентного підходу React, що забезпечує гнучкість у розробці, покращення продуктивності та масштабованість системи. Завдяки використанню JavaScript проект має високу інтерактивність, а також підтримує зручний і інтуїтивно зрозумілий інтерфейс, який спрощує процес взаємодії з програмою для користувачів різного рівня підготовки.

Цю програму можна використовувати в багатьох ситуаціях, залежно від того, що потрібно користувачеві. Вона допоможе весело і корисно проводити час, розвиваючи логічне мислення та знання. Завдяки налаштуванням, програму можна змінювати під різні теми, наприклад, історію, географію, математику чи іноземні мови. Це буде корисно для вчителів, які хочуть зробити навчання цікавішим. Також програму можна використовувати для командних ігор, турнірів або творчих завдань, об’єднуючи навчання і розваги.

ПЕРЕЛІК ПОСИЛАНЬ

1. Кросворд для навчання – URL: https://historybook.netlify.app/lessons/krosvord-dlia-navchannia/
2. Brief History of Crossword Puzzles – URL: https://www.crosswordtournament.com/more/wynne.html
3. A Brief History of Word Games – URL: https://www.bunkhistory.org/resources/a-brief-history-of-word-games
4. The Crossword Puzzle History – URL: https://www.factmonster.com/crossword-puzzle-history
5. The use of crossword puzzles as an educational tool – URL: https://pmc.ncbi.nlm.nih.gov/articles/PMC8106739/
6. Proficient in english with advanced vocabulary using game-based learning: narrative crossword puzzle – URL: https://ejournal.umm.ac.id/index.php/celtic/article/view/35510
7. Кросворд – URL: https://uk.wikipedia.org/wiki/%D0%9A%D1%80%D0%BE%D1%81%D0%B2%D0%BE%D1%80%D0%B4
8. Crossword Labs – URL: https://crosswordlabs.com/
9. EclipseCrossword – URL: https://www.eclipsecrossword.com/
10. Free Crossword Puzzle Maker – URL: https://www.edhelper.com/crossword.htm
11. Instant Online Crossword Puzzle Maker – URL: https://www.puzzle-maker.com/CW
12. Crossword Puzzle Generator – URL: https://www.crossword-generator.com/
13. Crossword Creator – URL: https://www.bearwoodlabs.com/products/crosswordcreator/
14. Generating a Crossword Puzzle – URL: https://www.baeldung.com/cs/generate-crossword-puzzle

Додаток А

Текст програми

```
    import React, { useState, useEffect, useRef } from 'react';
    import { initializeApp } from 'firebase/app';
    import { getAuth, signInAnonymously, signInWithCustomToken,
onAuthStateChanged } from 'firebase/auth';
    import { getFirestore, collection, addDoc, onSnapshot, query,
doc, updateDoc, deleteDoc, getDocs } from 'firebase/firestore';

    function App() {
      const [db, setDb] = useState(null);
      const [auth, setAuth] = useState(null);
      const [userId, setUserId] = useState(null);
      const [isAuthReady, setIsAuthReady] = useState(false); //
State to track auth readiness

      const [themes, setThemes] = useState([]);
      const [selectedThemeId, setSelectedThemeId] = useState('');
      const [newThemeName, setNewThemeName] = useState('');
      const [newWord, setNewWord] = useState('');
      const [newClue, setNewClue] = useState('');

      const [crosswordGrid, setCrosswordGrid] = useState([]);
      const [crosswordClues, setCrosswordClues] = useState([]);
      const [message, setMessage] = useState('');
      const     [showConfirmModal,     setShowConfirmModal]     =
useState(false);
      const [confirmAction, setConfirmAction] = useState(null);

      // Initialize Firebase and Auth
      useEffect(() => {
        try {
          const appId = typeof __app_id !== 'undefined' ? __app_id
: 'default-app-id';
          const      firebaseConfig      =      JSON.parse(typeof
__firebase_config !== 'undefined' ? __firebase_config : '{}');

          const app = initializeApp(firebaseConfig);
          const firestore = getFirestore(app);
          const firebaseAuth = getAuth(app);

          setDb(firestore);
          setAuth(firebaseAuth);

          const  unsubscribe  =  onAuthStateChanged(firebaseAuth,
async (user) => {
            if (!user) {
              try {
                //   Use   __initial_auth_token   if   available,
otherwise sign in anonymously
```

```
                if (typeof __initial_auth_token !== 'undefined'
&& __initial_auth_token) {
                  await      signInWithCustomToken(firebaseAuth,
__initial_auth_token);
                } else {
                  await signInAnonymously(firebaseAuth);
                }
              } catch (error) {
                console.error("Firebase Auth Error:", error);
                setMessage(`Помилка              аутентифікації:
${error.message}`);
              }
            }
            setUserId(firebaseAuth.currentUser?.uid           ||
crypto.randomUUID());
            setIsAuthReady(true); // Set auth ready after initial
check/sign-in
          });

          return () => unsubscribe();
        } catch (error) {
          console.error("Firebase    Initialization    Error:",
error);
          setMessage(`Помилка      ініціалізації      Firebase:
${error.message}`);
        }
      }, []);

      // Listen for real-time updates to themes
      useEffect(() => {
        if (!db || !isAuthReady || !userId) {
          return; // Ensure db, auth, and userId are ready
        }

        // Public data path for themes
        const      themesCollectionRef      =      collection(db,
`artifacts/${__app_id}/public/data/crosswordThemes`);
        const q = query(themesCollectionRef);

        const unsubscribe = onSnapshot(q, (snapshot) => {
          const themesData = snapshot.docs.map(doc => ({
            id: doc.id,
            ...doc.data()
          }));
          setThemes(themesData);
          setMessage(''); // Clear any previous messages
        }, (error) => {
          console.error("Error fetching themes:", error);
          setMessage(`Помилка          завантаження          тем:
${error.message}`);
        });

        return () => unsubscribe();
```

```
    }, [db, isAuthReady, userId]);

    const showModal = (action) => {
      setConfirmAction(() => action); // Store the function to
be called
      setShowConfirmModal(true);
    };

    const closeModal = () => {
      setShowConfirmModal(false);
      setConfirmAction(null);
    };

    const handleConfirm = () => {
      if (confirmAction) {
        confirmAction();
      }
      closeModal();
    };

    const handleAddTheme = async () => {
      if (!db) {
        setMessage("Firebase не ініціалізовано.");
        return;
      }
      if (newThemeName.trim() === '') {
        setMessage("Будь ласка, введіть назву теми.");
        return;
      }
      try {
        await                              addDoc(collection(db,
`artifacts/${__app_id}/public/data/crosswordThemes`), {
          name: newThemeName.trim(),
          words: []
        });
        setNewThemeName('');
        setMessage(`Тема "${newThemeName}" успішно додана.`);
      } catch (e) {
        console.error("Error adding document: ", e);
        setMessage(`Помилка додавання теми: ${e.message}`);
      }
    };

    const handleAddWord = async () => {
      if (!db) {
        setMessage("Firebase не ініціалізовано.");
        return;
      }
      if (selectedThemeId === '') {
        setMessage("Будь ласка, виберіть тему.");
        return;
      }
      if (newWord.trim() === '' || newClue.trim() === '') {
```

```
            setMessage("Будь ласка, введіть слово та підказку.");
            return;
          }

          const themeDocRef = doc(db, `artifacts/${__app_id}/public/data/crosswordThemes`, selectedThemeId);
          const selectedTheme = themes.find(t => t.id === selectedThemeId);

          if (selectedTheme) {
            const updatedWords = [...selectedTheme.words, { word: newWord.trim().toUpperCase(), clue: newClue.trim() }];
            try {
              await updateDoc(themeDocRef, { words: updatedWords });
              setNewWord('');
              setNewClue('');
              setMessage(`Слово "${newWord}" додано до теми "${selectedTheme.name}".`);
            } catch (e) {
              console.error("Error updating document: ", e);
              setMessage(`Помилка додавання слова: ${e.message}`);
            }
          } else {
            setMessage("Вибрана тема не знайдена.");
          }
        };

        const handleDeleteTheme = async (themeId, themeName) => {
          if (!db) {
            setMessage("Firebase не ініціалізовано.");
            return;
          }
          showModal(async () => {
            try {
              await deleteDoc(doc(db, `artifacts/${__app_id}/public/data/crosswordThemes`, themeId));
              setMessage(`Тема "${themeName}" успішно видалена.`);
              setSelectedThemeId(''); // Deselect if deleted
            } catch (e) {
              console.error("Error deleting theme: ", e);
              setMessage(`Помилка видалення теми: ${e.message}`);
            }
          });
        };

        const handleDeleteWord = async (themeId, wordToDelete, themeName) => {
          if (!db) {
            setMessage("Firebase не ініціалізовано.");
            return;
          }
```

```
        showModal(async () => {
          const themeDocRef = doc(db, `artifacts/${__app_id}/public/data/crosswordThemes`, themeId);
          const selectedTheme = themes.find(t => t.id === themeId);

          if (selectedTheme) {
            const updatedWords = selectedTheme.words.filter(word => word.word !== wordToDelete.word || word.clue !== wordToDelete.clue); // Filter by both word and clue to handle duplicates
            try {
              await updateDoc(themeDocRef, { words: updatedWords });
              setMessage(`Слово "${wordToDelete.word}" видалено з теми "${themeName}".`);
            } catch (e) {
              console.error("Error deleting word: ", e);
              setMessage(`Помилка видалення слова: ${e.message}`);
            }
          } else {
            setMessage("Вибрана тема не знайдена.");
          }
        });
      };

      // --- Crossword Generation Logic ---
      const generateCrossword = () => {
        const selectedTheme = themes.find(t => t.id === selectedThemeId);
        if (!selectedTheme || selectedTheme.words.length === 0) {
          setMessage("Будь ласка, виберіть тему з принаймні одним словом.");
          setCrosswordGrid([]);
          setCrosswordClues([]);
          return;
        }

        const wordsToPlace = selectedTheme.words.map(w => ({ word: w.word, clue: w.clue }));

        // Sort words by length in descending order
        wordsToPlace.sort((a, b) => b.word.length - a.word.length);

        const MAX_GRID_SIZE = 25; // Max size for the grid to prevent too large crosswords
        const minGridSize = wordsToPlace.length > 0 ? wordsToPlace[0].word.length + 5 : 10;
        const gridSize = Math.min(Math.max(minGridSize, 15), MAX_GRID_SIZE); // Ensure a reasonable size
```

```
        let grid = Array(gridSize).fill(null).map(() => Array(gridSize).fill(' '));
        let placedWords = [];
        let attempts = 0;
        const MAX_ATTEMPTS = 500; // Limit attempts to prevent infinite loops for difficult layouts

        const canPlaceWord = (currentGrid, wordObj, row, col, direction) => {
          const word = wordObj.word;
          const len = word.length;

          if (direction === 'across') {
            if (col + len > gridSize) return false; // Out of bounds
            if (col > 0 && currentGrid[row][col - 1] !== ' ') return false; // Letter before
            if (col + len < gridSize && currentGrid[row][col + len] !== ' ') return false; // Letter after

            for (let i = 0; i < len; i++) {
              const char = currentGrid[row][col + i];
              if (char !== ' ' && char !== word[i]) {
                return false; // Conflict with existing letter
              }
              // Check surroundings for conflicts (letters above/below unless intersection)
              if (char === ' ') { // Only check if not an intersection point
                if (row > 0 && currentGrid[row - 1][col + i] !== ' ') return false; // Letter above
                if (row < gridSize - 1 && currentGrid[row + 1][col + i] !== ' ') return false; // Letter below
              }
            }
          } else { // 'down'
            if (row + len > gridSize) return false; // Out of bounds
            if (row > 0 && currentGrid[row - 1][col] !== ' ') return false; // Letter before
            if (row + len < gridSize && currentGrid[row + len][col] !== ' ') return false; // Letter after

            for (let i = 0; i < len; i++) {
              const char = currentGrid[row + i][col];
              if (char !== ' ' && char !== word[i]) {
                return false; // Conflict with existing letter
              }
              // Check surroundings for conflicts (letters left/right unless intersection)
              if (char === ' ') { // Only check if not an intersection point
```

```
                if (col > 0 && currentGrid[row + i][col - 1] !==
' ') return false; // Letter left
                if (col < gridSize - 1 && currentGrid[row + i][col
+ 1] !== ' ') return false; // Letter right
              }
            }
          }
          return true;
        };

        const placeWord = (currentGrid, wordObj, row, col,
direction) => {
          const word = wordObj.word;
          const newGrid = currentGrid.map(arr => [...arr]); //
Deep copy
          if (direction === 'across') {
            for (let i = 0; i < word.length; i++) {
              newGrid[row][col + i] = word[i];
            }
          } else { // 'down'
            for (let i = 0; i < word.length; i++) {
              newGrid[row + i][col] = word[i];
            }
          }
          return newGrid;
        };

        // Place the first word (longest one) in the center,
across
        const firstWord = wordsToPlace.shift();
        if (firstWord) {
          const startRow = Math.floor(gridSize / 2);
          const     startCol     =     Math.floor((gridSize     -
firstWord.word.length) / 2);
          if (canPlaceWord(grid, firstWord, startRow, startCol,
'across')) {
            grid = placeWord(grid, firstWord, startRow, startCol,
'across');
            placedWords.push({ ...firstWord, row: startRow, col:
startCol, direction: 'across' });
          } else {
            setMessage("Не вдалося розмістити перше слово.
Спробуйте менший розмір сітки або інше слово.");
            setCrosswordGrid([]);
            setCrosswordClues([]);
            return;
          }
        }


        wordsToPlace.forEach(wordObj => {
          let wordPlaced = false;
          attempts = 0;
```

```
          // Try to place by intersecting with existing words
          for (let p of placedWords) {
            for (let i = 0; i < wordObj.word.length; i++) { //
For each letter in the new word
              const newWordLetter = wordObj.word[i];
              for (let j = 0; j < p.word.length; j++) { // For
each letter in an already placed word
                const placedWordLetter = p.word[j];

                if (newWordLetter === placedWordLetter) { //
Potential intersection
                  let tryRow, tryCol, tryDirection;

                  if (p.direction === 'across') {
                    // New word should be 'down'
                    tryRow = p.row - i;
                    tryCol = p.col + j;
                    tryDirection = 'down';
                  } else { // p.direction === 'down'
                    // New word should be 'across'
                    tryRow = p.row + j;
                    tryCol = p.col - i;
                    tryDirection = 'across';
                  }

                  if (tryRow >= 0 && tryRow < gridSize && tryCol
>= 0 && tryCol < gridSize) {
                    if   (canPlaceWord(grid,   wordObj,   tryRow,
tryCol, tryDirection)) {
                      grid  =  placeWord(grid,  wordObj,  tryRow,
tryCol, tryDirection);
                      placedWords.push({     ...wordObj,     row:
tryRow, col: tryCol, direction: tryDirection });
                      wordPlaced = true;
                      break; // Move to next wordObj
                    }
                  }
                }
                if (wordPlaced) break;
              }
              if (wordPlaced) break;
            }
            if (wordPlaced) break;
          }

          // If not placed by intersection, try random placement
(less ideal)
          if (!wordPlaced) {
            let randomAttempts = 0;
            const MAX_RANDOM_ATTEMPTS = 50;
            while     (!wordPlaced     &&     randomAttempts     <
MAX_RANDOM_ATTEMPTS) {
```

```
                const randRow = Math.floor(Math.random() *
gridSize);
                const randCol = Math.floor(Math.random() *
gridSize);
                const randDirection = Math.random() < 0.5 ?
'across' : 'down';

                if (canPlaceWord(grid, wordObj, randRow, randCol,
randDirection)) {
                  grid = placeWord(grid, wordObj, randRow, randCol,
randDirection);
                  placedWords.push({ ...wordObj, row: randRow, col:
randCol, direction: randDirection });
                  wordPlaced = true;
                }
                randomAttempts++;
              }
            }
          });

          // Generate clues in correct format
          const cluesAcross = placedWords
            .filter(w => w.direction === 'across')
            .sort((a, b) => (a.row - b.row) || (a.col - b.col))
            .map((w, index) => ({ number: index + 1, word: w.word,
clue: w.clue, row: w.row, col: w.col }));

          const cluesDown = placedWords
            .filter(w => w.direction === 'down')
            .sort((a, b) => (a.row - b.row) || (a.col - b.col))
            .map((w, index) => ({ number: index + 1, word: w.word,
clue: w.clue, row: w.row, col: w.col }));

          // Assign numbers to the grid for reference
          let numberedGrid = grid.map(row => row.map(cell => (cell
=== ' ' ? ' ' : ''))); // Use original grid content
          let clueNumber = 1;

          for (let r = 0; r < gridSize; r++) {
            for (let c = 0; c < gridSize; c++) {
              let isStartOfAcross = false;
              let isStartOfDown = false;

              // Check if current cell is the start of an 'across'
word
              const acrossWord = cluesAcross.find(cw => cw.row ===
r && cw.col === c);
              if (acrossWord) {
                acrossWord.gridNumber = clueNumber;
                isStartOfAcross = true;
              }
```

```
            // Check if current cell is the start of a 'down'
word
            const downWord = cluesDown.find(cw => cw.row === r &&
cw.col === c);
            if (downWord) {
              downWord.gridNumber = clueNumber;
              isStartOfDown = true;
            }

            if (isStartOfAcross || isStartOfDown) {
              numberedGrid[r][c] = `${clueNumber}`;
              clueNumber++;
            }
          }
        }


        setCrosswordGrid(grid);
        setCrosswordClues({ across: cluesAcross, down: cluesDown
});
        setMessage("Кросворд успішно згенеровано!");

        // Helper to add numbers to the grid based on placed words
        const finalGrid = grid.map((row, rIdx) => row.map((cell,
cIdx) => {
            const  matchingAcross  =  cluesAcross.find(clue  =>
clue.row === rIdx && clue.col === cIdx);
            const matchingDown = cluesDown.find(clue => clue.row
=== rIdx && clue.col === cIdx);

            if (cell !== ' ' && (matchingAcross || matchingDown))
{
                // Display the number if it's the start of an
across or down word
                return (
                    <div                 key={`${rIdx}-${cIdx}`}
className="relative w-8 h-8 md:w-10 md:h-10 border border-gray-
400 flex items-center justify-center bg-gray-100 text-gray-800
font-bold rounded-sm text-xs md:text-base">
                        <span  className="absolute  top-0.5  left-
0.5         text-[8px]         md:text-[10px]         text-gray-
600">{matchingAcross?.gridNumber                                ||
matchingDown?.gridNumber}</span>
                        {cell}
                    </div>
                );
            } else if (cell !== ' ') {
                // Just display the letter for non-starting cells
                return (
                    <div  key={`${rIdx}-${cIdx}`}  className="w-8
h-8  md:w-10  md:h-10  border  border-gray-400  flex  items-center
justify-center  bg-gray-100  text-gray-800  font-bold  rounded-sm
text-xs md:text-base">
```

```
                        {cell}
                    </div>
                );
            } else {
                // Empty cell
                return (
                    <div key={`${rIdx}-${cIdx}`} className="w-8
h-8 md:w-10 md:h-10 border border-gray-400 bg-gray-300 rounded-
sm"></div>
                );
            }
        }));
        setCrosswordGrid(finalGrid);
    };
    // --- End Crossword Generation Logic ---


    return (
        <div  className="min-h-screen  bg-gradient-to-br  from-
blue-50 to-indigo-100 p-4 md:p-8 font-sans antialiased text-gray-
800">
            <script src="https://cdn.tailwindcss.com"></script>
            <link
href="https://fonts.googleapis.com/css2?family=Inter:wght@300;40
0;500;600;700&display=swap" rel="stylesheet" />
            <style>
                {`
                body { font-family: 'Inter', sans-serif; }
                .scrollable-content { max-height: 400px; overflow-
y: auto; }
                /* Custom scrollbar for better aesthetics */
                .scrollable-content::-webkit-scrollbar {
                    width: 8px;
                }
                .scrollable-content::-webkit-scrollbar-track {
                    background: #f1f1f1;
                    border-radius: 10px;
                }
                .scrollable-content::-webkit-scrollbar-thumb {
                    background: #888;
                    border-radius: 10px;
                }
                .scrollable-content::-webkit-scrollbar-thumb:hover
{
                    background: #555;
                }
                .grid-cell {
                  width: 2rem; /* 32px */
                  height: 2rem; /* 32px */
                  @media (min-width: 768px) { /* md breakpoint */
                    width: 2.5rem; /* 40px */
                    height: 2.5rem; /* 40px */
                  }
```

```
                border: 1px solid #9ca3af; /* gray-400 */
                display: flex;
                align-items: center;
                justify-content: center;
                background-color: #f3f4f6; /* gray-100 */
                color: #1f2937; /* gray-800 */
                font-weight: 700; /* bold */
                border-radius: 0.125rem; /* rounded-sm */
                font-size: 0.75rem; /* text-xs */
                @media (min-width: 768px) {
                  font-size: 1rem; /* text-base */
                }
                position: relative;
              }
              .grid-cell.empty {
                background-color: #d1d5db; /* gray-300 */
              }
              .grid-cell .number {
                position: absolute;
                top: 2px;
                left: 2px;
                font-size: 8px;
                @media (min-width: 768px) {
                  font-size: 10px;
                }
                color: #4b5563; /* gray-600 */
              }
            `}
          </style>

          <div className="max-w-6xl mx-auto bg-white p-6 md:p-10
rounded-xl shadow-lg border border-blue-200">
            <h1 className="text-3xl md:text-4xl font-bold text-
center text-blue-700 mb-6">Генератор Кросвордів</h1>
            {userId && <p className="text-center text-gray-600
mb-4 text-sm">Ваш ID користувача: <span className="font-mono bg-
gray-100 px-2 py-1 rounded-md text-gray-700">{userId}</span></p>}

            {message && (
              <div className="bg-blue-100 border border-blue-400
text-blue-700 px-4 py-3 rounded-lg mb-6 shadow-sm text-center">
                {message}
              </div>
            )}

            {/* Theme Management */}
            <div className="mb-8 p-6 bg-blue-50 rounded-lg border
border-blue-200 shadow-sm">
              <h2 className="text-2xl font-semibold text-blue-
600 mb-4">Управління Темами</h2>
              <div className="flex flex-col md:flex-row gap-4 mb-
4">
                <input
```

```
                    type="text"
                    placeholder="Нова назва теми"
                    value={newThemeName}
                    onChange={(e)                                    =>
setNewThemeName(e.target.value)}
                    className="flex-grow p-3 border border-blue-
300 rounded-lg focus:ring-2 focus:ring-blue-400 focus:border-
transparent transition"
                  />
                  <button
                    onClick={handleAddTheme}
                    className="bg-blue-600 hover:bg-blue-700 text-
white font-bold py-3 px-6 rounded-lg shadow-md transition
transform hover:scale-105"
                  >
                    Додати Тему
                  </button>
                </div>

                <div className="mb-4">
                  <label htmlFor="select-theme" className="block
text-blue-700 font-medium mb-2">Виберіть Тему:</label>
                  <select
                    id="select-theme"
                    value={selectedThemeId}
                    onChange={(e)                                    =>
setSelectedThemeId(e.target.value)}
                    className="w-full p-3 border border-blue-300
rounded-lg bg-white focus:ring-2 focus:ring-blue-400
focus:border-transparent transition"
                  >
                    <option value="">-- Виберіть тему --</option>
                    {themes.map(theme => (
                      <option                          key={theme.id}
value={theme.id}>{theme.name}</option>
                    ))}
                  </select>
                </div>

                {selectedThemeId && (
                  <div  className="mt-6  p-4  bg-white  rounded-lg
border border-blue-100 shadow-inner">
                    <h3  className="text-xl  font-semibold  text-
blue-600 mb-3">Додати Слово до Теми: <span className="text-blue-
800">{themes.find(t            =>            t.id            ===
selectedThemeId)?.name}</span></h3>
                    <div className="flex flex-col md:flex-row gap-
4 mb-4">
                      <input
                        type="text"
                        placeholder="Слово (наприклад, CAT)"
                        value={newWord}
```

```
                        onChange={(e)                                 =>
setNewWord(e.target.value)}
                        className="flex-grow  p-3  border  border-
blue-300     rounded-lg     focus:ring-2     focus:ring-blue-400
focus:border-transparent transition"
                      />
                      <input
                        type="text"
                        placeholder="Підказка           (наприклад,
Пухнастий улюбленець)"
                        value={newClue}
                        onChange={(e)                                 =>
setNewClue(e.target.value)}
                        className="flex-grow  p-3  border  border-
blue-300     rounded-lg     focus:ring-2     focus:ring-blue-400
focus:border-transparent transition"
                      />
                      <button
                        onClick={handleAddWord}
                        className="bg-green-600 hover:bg-green-700
text-white  font-bold  py-3  px-6  rounded-lg  shadow-md  transition
transform hover:scale-105"
                      >
                        Додати Слово
                      </button>
                    </div>

                    <h4 className="text-lg font-medium text-blue-
700 mt-5 mb-2">Слова в цій темі:</h4>
                    {themes.find(t        =>        t.id        ===
selectedThemeId)?.words.length > 0 ? (
                      <div  className="scrollable-content  bg-gray-
50 p-4 rounded-lg border border-gray-200">
                        <ul className="space-y-2">
                          {themes.find(t      =>      t.id      ===
selectedThemeId)?.words.map((word, index) => (
                            <li     key={index}     className="flex
justify-between  items-center  bg-white  p-3  rounded-md  shadow-sm
border border-gray-100">
                              <span>
                                <span     className="font-semibold
text-gray-900">{word.word}</span>: {word.clue}
                              </span>
                              <button
                                onClick={()                           =>
handleDeleteWord(selectedThemeId, word, themes.find(t => t.id ===
selectedThemeId)?.name)}
                                className="bg-red-500    hover:bg-
red-600 text-white p-2 rounded-full text-xs font-bold transition
transform hover:scale-110"
                                title="Видалити слово"
                              >
                                ×
```

```
                            </button>
                          </li>
                        ))}
                      </ul>
                    </div>
                  ) : (
                    <p className="text-gray-500 italic">Слова ще
не додані до цієї теми.</p>
                  )}
                   <button
                     onClick={()                                  =>
handleDeleteTheme(selectedThemeId,  themes.find(t  =>  t.id  ===
selectedThemeId)?.name)}
                     className="mt-6  bg-red-600  hover:bg-red-
700 text-white font-bold py-3 px-6 rounded-lg shadow-md transition
transform hover:scale-105 w-full md:w-auto"
                   >
                     Видалити Вибрану Тему
                   </button>
                 </div>
               )}
             </div>

             {/* Crossword Generation */}
             <div  className="mb-8  p-6  bg-indigo-50  rounded-lg
border border-indigo-200 shadow-sm">
               <h2 className="text-2xl font-semibold text-indigo-
600 mb-4">Генерація Кросворду</h2>
               <button
                 onClick={generateCrossword}
                 className="bg-indigo-600    hover:bg-indigo-700
text-white  font-bold py-3 px-8 rounded-lg shadow-xl transition
transform   hover:scale-105   focus:outline-none   focus:ring-2
focus:ring-indigo-400 focus:ring-offset-2"
                 disabled={!selectedThemeId  ||  themes.find(t  =>
t.id === selectedThemeId)?.words.length === 0}
               >
                 Згенерувати Кросворд
               </button>
               {!selectedThemeId && <p className="text-sm text-
gray-500 mt-2">Виберіть тему, щоб згенерувати кросворд.</p>}
             </div>

             {/* Generated Crossword Display */}
             {crosswordGrid.length > 0 && (
               <div className="p-6 bg-white rounded-xl shadow-lg
border border-gray-200">
                 <h2 className="text-2xl font-semibold text-gray-
700 mb-4 text-center">Ваш Кросворд</h2>
                 <div className="flex flex-col lg:flex-row gap-8">
                   {/* Crossword Grid */}
                   <div  className="flex-1  flex  justify-center
items-start overflow-auto">
```

```
                    <div className="inline-grid gap-px" style={{
                      gridTemplateColumns:
`repeat(${crosswordGrid[0]?.length || 0}, minmax(0, 1fr))`
                    }}>
                      {crosswordGrid.flat()}      {/*      Render
flattened grid cells */}
                    </div>
                  </div>

                  {/* Clues */}
                  <div      className="flex-1      min-w-[300px]
scrollable-content p-4 border border-gray-300 rounded-lg bg-gray-
50">
                    <h3  className="text-xl  font-semibold  text-
gray-800 mb-3">Підказки:</h3>
                    {crosswordClues.across.length > 0 && (
                      <div className="mb-4">
                        <h4 className="text-lg font-medium text-
gray-700 mb-2">ПО ГОРИЗОНТАЛІ:</h4>
                        <ul   className="list-disc   list-inside
space-y-1">
                          {crosswordClues.across.map((clue,
index) => (
                            <li          key={`across-${index}`}
className="text-gray-800">
                              <span             className="font-
bold">{clue.gridNumber}.</span> {clue.clue}
                            </li>
                          ))}
                        </ul>
                      </div>
                    )}
                    {crosswordClues.down.length > 0 && (
                      <div>
                        <h4 className="text-lg font-medium text-
gray-700 mb-2">ПО ВЕРТИКАЛІ:</h4>
                        <ul   className="list-disc   list-inside
space-y-1">
                          {crosswordClues.down.map((clue, index)
=> (
                            <li            key={`down-${index}`}
className="text-gray-800">
                              <span             className="font-
bold">{clue.gridNumber}.</span> {clue.clue}
                            </li>
                          ))}
                        </ul>
                      </div>
                    )}
                  </div>
                </div>
              </div>
            )}
```

```
          </div>

          {/* Confirmation Modal */}
          {showConfirmModal && (
            <div className="fixed inset-0 bg-black bg-opacity-50
flex items-center justify-center p-4 z-50">
              <div className="bg-white p-6 rounded-lg shadow-xl
max-w-sm w-full border border-gray-300">
                <h3 className="text-lg font-semibold text-gray-
800 mb-4">Підтвердіть ДІЮ</h3>
                <p className="text-gray-600 mb-6">Ви впевнені, що
хочете виконати цю дію? Її не можна буде скасувати.</p>
                <div className="flex justify-end gap-3">
                  <button
                    onClick={closeModal}
                    className="px-5 py-2 bg-gray-200 text-gray-
800 rounded-lg hover:bg-gray-300 transition"
                  >
                    Скасувати
                  </button>
                  <button
                    onClick={handleConfirm}
                    className="px-5 py-2 bg-red-600 text-white
rounded-lg hover:bg-red-700 transition"
                  >
                    Підтвердити
                  </button>
                </div>
              </div>
            </div>
          )}
        </div>
      );
    }

    export default App;
```

```
    <!DOCTYPE html>
    <html lang="uk">
    <head>
        <meta charset="UTF-8">
        <meta    name="viewport"    content="width=device-width,
initial-scale=1.0">
        <title>Мої Кросворди</title>
        <script src="https://cdn.tailwindcss.com"></script>
        <link
href="https://fonts.googleapis.com/css2?family=Inter:wght@400;50
0;600;700&display=swap" rel="stylesheet">
```

```
        <style>
            body {
                font-family: 'Inter', sans-serif;
            }
            /* Custom scrollbar */
            ::-webkit-scrollbar {
                width: 8px;
            }
            ::-webkit-scrollbar-track {
                background: #f1f5f9;
            }
            ::-webkit-scrollbar-thumb {
                background: #94a3b8;
                border-radius: 10px;
            }
            ::-webkit-scrollbar-thumb:hover {
                background: #64748b;
            }
        </style>
    </head>
    <body class="bg-slate-100 text-slate-800">

        <div class="container mx-auto p-4 md:p-8">
            <header class="text-center mb-8">
                <h1 class="text-3xl md:text-4xl font-bold text-
slate-900">Мої Кросворди</h1>
                <p  class="text-slate-600  mt-2">Список  ваших
збережених тематичних кросвордів.</p>
            </header>

            <div class="max-w-4xl mx-auto">
                <div class="text-right mb-6">
                    <a  href="#"  id="create-new-btn"  class="bg-
indigo-600 text-white font-semibold py-2 px-5 rounded-lg hover:bg-
indigo-700  focus:outline-none  focus:ring-2  focus:ring-offset-2
focus:ring-indigo-500 transition-colors duration-200">
                        + Створити новий
                    </a>
                </div>

                <div id="crossword-list" class="space-y-4">
                    <!--  Crosswords  will  be  dynamically  added
here -->
                </div>
            </div>
        </div>

        <script>
            document.addEventListener('DOMContentLoaded',  ()  =>
{
                const          crosswordListContainer          =
document.getElementById('crossword-list');
```

```
                // --- Mock Data ---
                // In a real application, this data would be saved
from the generator page.
                //  For  now,  we  create  some  mock  data  to
demonstrate the list page.
                function setupMockData() {
                    const mockCrosswords = [
                        {
                            id: 'geo2025',
                            title: 'Географія України',
                            wordCount: 5,
                            createdAt: '2025-05-27T10:00:00Z',
                            words: [
                                {   word:   "УКРАЇНА",   clue:
"Найбільша за площею країна, яка повністю лежить у Європі." },
                                { word: "КИЇВ", clue: "Столиця
та найбільше місто України." },
                                {    word:    "ДНІПРО",    clue:
"Найдовша річка в межах України." },
                                {   word:   "КАРПАТИ",   clue:
"Гірська система на заході України." },
                                {    word:    "БОРЩ",    clue:
"Традиційна українська страва, що є символом національної кухні."
}
                            ]
                        },
                        {
                            id: 'it2025',
                            title: 'IT Термінологія',
                            wordCount: 4,
                            createdAt: '2025-05-26T14:30:00Z',
                            words: [
                                {   word:   "КОМПІЛЯТОР",   clue:
"Програма, що перетворює вихідний код на машинний." },
                                {   word:   "ФРЕЙМВОРК",   clue:
"Каркас для розробки програмного забезпечення." },
                                {   word:   "АЛГОРИТМ",   clue:
"Послідовність інструкцій для вирішення задачі." },
                                {   word:   "БІБЛІОТЕКА",   clue:
"Набір готових функцій та класів." }
                            ]
                        },
                         {
                            id: 'bio2025',
                            title: 'Біологія: Клітина',
                            wordCount: 3,
                            createdAt: '2025-05-25T09:00:00Z',
                            words: [
                                { word: "ЯДРО", clue: "Центральна
органела клітини, що містить ДНК." },
                                {   word:   "МІТОХОНДРІЯ",   clue:
"Енергетична станція клітини." },
```

```
                                    {    word:    "МЕМБРАНА",    clue:
"Зовнішня оболонка клітини." }
                                ]
                            }
                        ];

                        if (!localStorage.getItem('crosswords')) {
                            localStorage.setItem('crosswords',
JSON.stringify(mockCrosswords));
                        }
                    }

                    // --- Rendering Logic ---
                    function renderCrosswordList() {
                        const            savedCrosswords            =
JSON.parse(localStorage.getItem('crosswords') || '[]');

                        if (savedCrosswords.length === 0) {
                            crosswordListContainer.innerHTML = `
                                <div class="bg-white p-8 rounded-2xl
shadow-md text-center">
                                    <h3 class="text-xl font-semibold
text-slate-700">Список порожній</h3>
                                    <p class="text-slate-500 mt-2">У
вас ще немає збережених кросвордів. Створіть свій перший!</p>
                                </div>
                            `;
                            return;
                        }

                        crosswordListContainer.innerHTML           =
savedCrosswords.map(cw => `
                            <div  class="bg-white  p-5  rounded-2xl
shadow-md  flex  items-center  justify-between  transition-shadow
hover:shadow-lg">
                                <div class="flex-grow">
                                    <h3   class="text-xl   font-bold
text-slate-800">${cw.title}</h3>
                                    <p class="text-sm text-slate-500
mt-1">
                                        <span>${cw.wordCount}
слів</span> &middot;
                                        <span>Створено:        ${new
Date(cw.createdAt).toLocaleDateString('uk-UA')}</span>
                                    </p>
                                </div>
                                <div    class="flex-shrink-0    flex
items-center gap-x-3 ml-4">
                                     <a  href="#"  data-id="${cw.id}"
class="view-btn  text-sm  font-medium  text-indigo-600  hover:text-
indigo-800 py-2 px-4 rounded-md bg-indigo-50 hover:bg-indigo-100
transition-colors">
                                        Переглянути
```

```
                                    </a>
                                    <a  href="#"  data-id="${cw.id}"
class="edit-btn text-sm font-medium text-emerald-600 hover:text-
emerald-800 py-2 px-4 rounded-md bg-emerald-50 hover:bg-emerald-
100 transition-colors">
                                        Редагувати
                                    </a>
                                </div>
                            </div>
                        `).join('');
                    }

                    // --- Event Listeners ---
                    crosswordListContainer.addEventListener('click',
(e) => {
                        const target = e.target.closest('a');
                        if (!target) return;

                        const crosswordId = target.dataset.id;

                        if (target.classList.contains('edit-btn')) {
                            // In a real app, this would link to the
generator with the ID
                            // For example: window.location.href =
`crossword_generator.html?id=${crosswordId}`;
                            alert(`Редагування кросворду з ID:
${crosswordId}`);
                            e.preventDefault();
                        }

                        if (target.classList.contains('view-btn')) {
                            // This would link to a viewer page or
the generator in a "view" mode
                            alert(`Перегляд кросворду з ID:
${crosswordId}`);
                            e.preventDefault();
                        }
                    });

                    document.getElementById('create-new-
btn').addEventListener('click', (e) => {
                        // This would link to the generator page to
create a new crossword
                        alert('Перехід на сторінку створення нового
кросворду...');
                        e.preventDefault();
                    });

                    // --- Initial Setup ---
                    setupMockData();
                    renderCrosswordList();
                });
            </script>
```

```
    </body>
    </html>



    <!DOCTYPE html>
    <html lang="uk">
    <head>
        <meta charset="UTF-8">
        <meta    name="viewport"    content="width=device-width,
initial-scale=1.0">
        <title>Генератор Кросвордів</title>
        <script src="https://cdn.tailwindcss.com"></script>
        <script
src="https://cdnjs.cloudflare.com/ajax/libs/jspdf/2.5.1/jspdf.um
d.min.js"></script>
        <script
src="https://cdnjs.cloudflare.com/ajax/libs/html2canvas/1.4.1/ht
ml2canvas.min.js"></script>
        <link
href="https://fonts.googleapis.com/css2?family=Inter:wght@400;50
0;600;700&display=swap" rel="stylesheet">
        <style>
            body {
                font-family: 'Inter', sans-serif;
            }
            .crossword-grid table {
                border-collapse: collapse;
            }
            .crossword-grid td {
                width: 30px;
                height: 30px;
                text-align: center;
                vertical-align: middle;
                border: 1px solid #9ca3af;
                font-size: 16px;
                position: relative;
                text-transform: uppercase;
            }
            .crossword-grid .filled {
                background-color: #ffffff;
            }
            .crossword-grid .empty {
                background-color: #1f2937;
                border-color: #1f2937;
            }
            .crossword-grid .number {
                position: absolute;
                top: 1px;
                left: 2px;
                font-size: 9px;
                font-weight: bold;
```

```
                color: #374151;
            }
            /* Custom scrollbar */
            ::-webkit-scrollbar {
                width: 8px;
            }
            ::-webkit-scrollbar-track {
                background: #f1f5f9;
            }
            ::-webkit-scrollbar-thumb {
                background: #94a3b8;
                border-radius: 10px;
            }
            ::-webkit-scrollbar-thumb:hover {
                background: #64748b;
            }
        </style>
    </head>
    <body class="bg-slate-100 text-slate-800">

        <div class="container mx-auto p-4 md:p-8">
            <header class="text-center mb-8">
                <h1 class="text-3xl md:text-4xl font-bold text-
slate-900">Генератор Кросвордів</h1>
                <p class="text-slate-600 mt-2">Створіть власний
кросворд, додайте слова та експортуйте в PDF.</p>
            </header>

            <div class="flex flex-col lg:flex-row gap-8">
                <!-- Панель керування -->
                <div class="lg:w-1/3 w-full bg-white p-6 rounded-
2xl shadow-lg">
                    <h2 class="text-2xl font-semibold mb-4
border-b pb-2">1. Додайте слова</h2>
                    <div class="space-y-4">
                        <div>
                            <label for="word-input" class="block
text-sm font-medium text-slate-700">Слово:</label>
                            <input type="text" id="word-input"
class="mt-1 block w-full px-3 py-2 bg-white border border-slate-
300 rounded-md shadow-sm placeholder-slate-400 focus:outline-none
focus:ring-indigo-500 focus:border-indigo-500 sm:text-sm">
                        </div>
                        <div>
                            <label for="clue-input" class="block
text-sm font-medium text-slate-700">Підказка:</label>
                            <input type="text" id="clue-input"
class="mt-1 block w-full px-3 py-2 bg-white border border-slate-
300 rounded-md shadow-sm placeholder-slate-400 focus:outline-none
focus:ring-indigo-500 focus:border-indigo-500 sm:text-sm">
                        </div>
                        <button id="add-word-btn" class="w-full
bg-indigo-600 text-white font-semibold py-2 px-4 rounded-lg
```

```
hover:bg-indigo-700 focus:outline-none focus:ring-2 focus:ring-
offset-2 focus:ring-indigo-500 transition-colors duration-200">
                            Додати слово
                        </button>
                    </div>

                    <div id="word-list-container" class="mt-6">
                        <h3 class="font-semibold text-lg mb-
2">Список слів:</h3>
                        <div id="word-list" class="max-h-60
overflow-y-auto space-y-2 pr-2">
                            <!-- Words will be dynamically added
here -->
                        </div>
                    </div>

                     <div class="mt-8 border-t pt-6">
                        <h2 class="text-2xl font-semibold mb-
4">2. Згенеруйте</h2>
                         <p id="status" class="text-sm text-
slate-500 mb-4"></p>
                        <div class="flex space-x-4">
                            <button id="generate-btn" class="w-
full bg-emerald-600 text-white font-semibold py-2 px-4 rounded-lg
hover:bg-emerald-700 focus:outline-none focus:ring-2 focus:ring-
offset-2 focus:ring-emerald-500 transition-colors duration-200">
                                Створити кросворд
                            </button>
                            <button id="pdf-btn" disabled
class="w-full bg-slate-400 text-white font-semibold py-2 px-4
rounded-lg cursor-not-allowed transition-colors duration-200">
                                Експорт в PDF
                            </button>
                        </div>
                     </div>
                </div>

                <!-- Область кросворду -->
                <div class="lg:w-2/3 w-full bg-white p-6 rounded-
2xl shadow-lg">
                    <h2 class="text-2xl font-semibold mb-4 text-
center">Ваш Кросворд</h2>
                    <div id="crossword-output" class="space-y-
6">
                        <div id="crossword-container"
class="flex justify-center items-start">
                            <div id="crossword-grid"
class="crossword-grid">
                                <div class="text-slate-500 text-
center p-8 border-2 border-dashed rounded-lg">
                                    Тут з'явиться ваш кросворд
після генерації.
                                </div>
```

```
                        </div>
                    </div>
                    <div                id="clues-container"
class="hidden">
                        <div  class="flex  flex-col  md:flex-
row gap-6">
                            <div class="w-full md:w-1/2">
                                <h3 class="font-bold text-lg
mb-2 text-slate-800">По горизонталі</h3>
                                <div       id="across-clues"
class="space-y-1 text-slate-700 text-sm"></div>
                            </div>
                            <div class="w-full md:w-1/2">
                                <h3 class="font-bold text-lg
mb-2 text-slate-800">По вертикалі</h3>
                                <div         id="down-clues"
class="space-y-1 text-slate-700 text-sm"></div>
                            </div>
                        </div>
                    </div>
                </div>
            </div>
        </div>

        <script type="module">
            // Basic setup for jsPDF
            const { jsPDF } = window.jspdf;

            // Application state
            let words = [
                { word: "УКРАЇНА", clue: "Найбільша за площею
країна, яка повністю лежить у Європі." },
                { word: "КИЇВ", clue: "Столиця та найбільше місто
України." },
                { word: "ДНІПРО", clue: "Найдовша річка в межах
України." },
                { word: "КАРПАТИ", clue: "Гірська система на
заході України." },
                { word: "БОРЩ", clue: "Традиційна українська
страва, що є символом національної кухні." }
            ];

            // DOM Elements
            const  wordInput  =  document.getElementById('word-
input');
            const  clueInput  =  document.getElementById('clue-
input');
            const  addWordBtn  =  document.getElementById('add-
word-btn');
            const  wordList  =  document.getElementById('word-
list');
```

```
            const generateBtn =
document.getElementById('generate-btn');
            const pdfBtn = document.getElementById('pdf-btn');
            const crosswordGrid =
document.getElementById('crossword-grid');
            const acrossClues = document.getElementById('across-
clues');
            const downClues = document.getElementById('down-
clues');
            const cluesContainer =
document.getElementById('clues-container');
            const statusEl = document.getElementById('status');

            // --- Word Management ---

            function renderWordList() {
                wordList.innerHTML = '';
                if (words.length === 0) {
                    wordList.innerHTML = '<p class="text-slate-
500 text-sm">Додайте ваше перше слово.</p>';
                    return;
                }
                words.forEach((item, index) => {
                    const li = document.createElement('div');
                    li.className = 'p-2 border rounded-md bg-
slate-50 flex justify-between items-center';
                    li.innerHTML = `
                        <div class="flex-grow">
                            <span class="font-semibold text-
slate-800">${item.word.toUpperCase()}</span>
                            <p class="text-xs text-slate-
500">${item.clue}</p>
                        </div>
                        <button data-index="${index}"
class="remove-btn ml-4 text-red-500 hover:text-red-700 text-sm
font-medium">Видалити</button>
                    `;
                    wordList.appendChild(li);
                });
            }

            function addWord() {
                const word =
wordInput.value.trim().toUpperCase().replace(/[^А-ЯІЇЄҐ]/g, '');
                const clue = clueInput.value.trim();
                if (word && clue) {
                    words.push({ word, clue });
                    wordInput.value = '';
                    clueInput.value = '';
                    renderWordList();
                    wordInput.focus();
                } else {
```

```
                alert('Будь ласка, введіть і слово, і підказку.');
            }
        }

        wordList.addEventListener('click', (e) => {
            if (e.target.classList.contains('remove-btn')) {
                const index = e.target.dataset.index;
                words.splice(index, 1);
                renderWordList();
            }
        });

        // --- Crossword Generation Logic ---

        function generateCrossword() {
            if (words.length < 2) {
                alert("Будь ласка, додайте принаймні два слова.");
                return;
            }

            statusEl.textContent = 'Генерація... Це може зайняти деякий час.';
            pdfBtn.disabled = true;
            pdfBtn.classList.add('cursor-not-allowed', 'bg-slate-400');
            pdfBtn.classList.remove('bg-red-600', 'hover:bg-red-700');

            // Use a timeout to allow the UI to update before the heavy computation starts
            setTimeout(() => {
                const sortedWords = [...words].sort((a, b) => b.word.length - a.word.length);
                let gridSize = 25;
                let grid = Array(gridSize).fill(null).map(() => Array(gridSize).fill(null));
                let placedWords = [];
                let unplacedWords = [];

                // Place the first (longest) word
                const firstWord = sortedWords.shift();
                const startPos = Math.floor((gridSize - firstWord.word.length) / 2);
                for (let i = 0; i < firstWord.word.length; i++) {
                    grid[startPos][startPos + i] = firstWord.word[i];
                }
                placedWords.push({
                    ...firstWord,
                    row: startPos,
```

```
                        col: startPos,
                        direction: 'across',
                        number: 1
                    });

                    // Try to place the rest of the words
                    for (const wordObj of sortedWords) {
                        let bestFit = null;
                        let maxIntersections = -1;

                        for (let i = 0; i < placedWords.length;
i++) {
                            const placed = placedWords[i];
                            for (let j = 0; j <
placed.word.length; j++) {
                                for (let k = 0; k <
wordObj.word.length; k++) {
                                    if (placed.word[j] ===
wordObj.word[k]) {
                                        let newRow, newCol;
                                        const newDirection =
placed.direction === 'across' ? 'down' : 'across';

                                        if (placed.direction ===
'across') {
                                            newRow = placed.row
- k;
                                            newCol = placed.col
+ j;
                                        } else {
                                            newRow = placed.row
+ j;
                                            newCol = placed.col
- k;
                                        }

                                        if
(canPlaceWord(wordObj.word, newRow, newCol, newDirection, grid))
{
                                            // Simple heuristic:
first valid placement is fine for this example.
                                            //     A     better
heuristic would count intersections.
                                            bestFit = { row:
newRow, col: newCol, direction: newDirection };
                                            break;
                                        }
                                    }
                                }
                                if (bestFit) break;
                            }
                            if (bestFit) break;
                        }
```

```
                        if (bestFit) {
                            for (let l = 0; l < wordObj.word.length; l++) {
                                if (bestFit.direction === 'across') {
                                    grid[bestFit.row][bestFit.col + l] = wordObj.word[l];
                                } else {
                                    grid[bestFit.row + l][bestFit.col] = wordObj.word[l];
                                }
                            }
                            placedWords.push({ ...wordObj, ...bestFit });
                        } else {
                            unplacedWords.push(wordObj.word);
                        }
                    }

                    finalizeAndRender(grid, placedWords, unplacedWords);
                }, 100);
            }

            function canPlaceWord(word, row, col, direction, grid) {
                const gridSize = grid.length;
                if (row < 0 || col < 0) return false;

                if (direction === 'across') {
                    if (col + word.length > gridSize) return false;
                    // Check for collisions and adjacent letters
                    if (col > 0 && grid[row][col - 1] !== null) return false;
                    if (col + word.length < gridSize && grid[row][col + word.length] !== null) return false;

                    for (let i = 0; i < word.length; i++) {
                        if (grid[row][col + i] !== null && grid[row][col + i] !== word[i]) return false;
                        if (grid[row][col + i] === null) {
                            if (row > 0 && grid[row - 1][col + i] !== null) return false;
                            if (row < gridSize - 1 && grid[row + 1][col + i] !== null) return false;
                        }
                    }
                } else { // 'down'
                    if (row + word.length > gridSize) return false;
                    // Check for collisions and adjacent letters
```

```
                    if (row > 0 && grid[row - 1][col] !== null)
return false;
                    if (row + word.length < gridSize && grid[row
+ word.length][col] !== null) return false;

                    for (let i = 0; i < word.length; i++) {
                        if (grid[row + i][col] !== null &&
grid[row + i][col] !== word[i]) return false;
                         if (grid[row + i][col] === null) {
                            if (col > 0 && grid[row + i][col - 1]
!== null) return false;
                            if (col < gridSize - 1 && grid[row +
i][col + 1] !== null) return false;
                        }
                    }
                }
                return true;
            }

            function    finalizeAndRender(grid,    placedWords,
unplacedWords) {
                // Trim the grid to fit the content
                let minRow = Infinity, maxRow = -1, minCol =
Infinity, maxCol = -1;
                for (let r = 0; r < grid.length; r++) {
                    for (let c = 0; c < grid[r].length; c++) {
                        if (grid[r][c] !== null) {
                            minRow = Math.min(minRow, r);
                            maxRow = Math.max(maxRow, r);
                            minCol = Math.min(minCol, c);
                            maxCol = Math.max(maxCol, c);
                        }
                    }
                }

                if (maxRow === -1) { // No words placed
                    statusEl.textContent = "Не вдалося створити
кросворд. Спробуйте інші слова.";
                    return;
                }

                const trimmedGrid = grid.slice(minRow, maxRow +
1).map(row => row.slice(minCol, maxCol + 1));

                // Adjust placed words coordinates
                placedWords.forEach(w => {
                    w.row -= minRow;
                    w.col -= minCol;
                });

                // Assign numbers
                placedWords.sort((a,b) => (a.row * 100 + a.col)
- (b.row * 100 + b.col));
```

```
                let number = 1;
                const starts = new Set();
                for (const word of placedWords) {
                    const key = `${word.row},${word.col}`;
                    if (!starts.has(key)) {
                        starts.add(key);
                        word.number = number;
                        //  Find  all  words  starting  at  this
position
                        placedWords.filter(w   =>   w.row   ===
word.row && w.col === word.col).forEach(w => w.number = number);
                        number++;
                    }
                }

                renderGrid(trimmedGrid, placedWords);
                renderClues(placedWords);

                statusEl.textContent = unplacedWords.length > 0
                    ?   `Готово!   Не   вдалося   розмістити:
${unplacedWords.join(', ')}`
                    : 'Кросворд успішно створено!';

                pdfBtn.disabled = false;
                pdfBtn.classList.remove('cursor-not-allowed',
'bg-slate-400');
                pdfBtn.classList.add('bg-red-600',    'hover:bg-
red-700');
            }


            // --- Rendering ---

            function renderGrid(grid, words) {
                crosswordGrid.innerHTML = '';
                const table = document.createElement('table');
                const numberedCells = new Map();

                words.forEach(word => {
                    if(word.number){
                        const key = `${word.row},${word.col}`;
                        if(!numberedCells.has(key)){
                           numberedCells.set(key, word.number);
                        }
                    }
                });

                for (let r = 0; r < grid.length; r++) {
                    const tr = document.createElement('tr');
                    for (let c = 0; c < grid[r].length; c++) {
                        const td = document.createElement('td');
                        if (grid[r][c]) {
```

```
                            td.textContent = ''; // grid[r][c];
// Keep letters hidden for puzzle
                            td.classList.add('filled');
                            const key = `${r},${c}`;
                            if (numberedCells.has(key)) {
                                const        numberSpan        =
document.createElement('span');
                                numberSpan.className = 'number';
                                numberSpan.textContent         =
numberedCells.get(key);
                                td.appendChild(numberSpan);
                            }
                        } else {
                            td.classList.add('empty');
                        }
                        tr.appendChild(td);
                    }
                    table.appendChild(tr);
                }
                crosswordGrid.appendChild(table);
            }

            function renderClues(words) {
                acrossClues.innerHTML = '';
                downClues.innerHTML = '';

                const across = words.filter(w => w.direction ===
'across').sort((a,b) => a.number - b.number);
                const down = words.filter(w => w.direction ===
'down').sort((a,b) => a.number - b.number);

                across.forEach(w => {
                    const div = document.createElement('div');
                    div.innerHTML    =    `<span    class="font-
semibold">${w.number}.</span> ${w.clue}`;
                    acrossClues.appendChild(div);
                });

                down.forEach(w => {
                    const div = document.createElement('div');
                    div.innerHTML    =    `<span    class="font-
semibold">${w.number}.</span> ${w.clue}`;
                    downClues.appendChild(div);
                });
                cluesContainer.classList.remove('hidden');
            }

            // --- PDF Export ---

            async function exportToPDF() {
                const                  gridEl                  =
document.getElementById('crossword-grid');
```

```
                const cluesEl = document.getElementById('clues-
container');

                statusEl.textContent = 'Готуємо PDF...';

                try {
                    // 1. Create canvas from the grid
                    const canvas = await html2canvas(gridEl, {
                        scale: 2, // Higher scale for better
quality
                        backgroundColor: null // Transparent
background
                    });
                    const imgData =
canvas.toDataURL('image/png');

                    const doc = new jsPDF({
                        orientation: 'p',
                        unit: 'mm',
                        format: 'a4'
                    });

                    const pageWidth =
doc.internal.pageSize.getWidth();
                    const pageHeight =
doc.internal.pageSize.getHeight();
                    const margin = 15;

                    // Title
                    doc.setFont('helvetica', 'bold');
                    doc.setFontSize(22);
                    doc.text('Кросворд', pageWidth / 2, margin,
{ align: 'center' });

                    // Grid Image
                    const imgWidth = canvas.width;
                    const imgHeight = canvas.height;
                    const ratio = imgWidth / imgHeight;
                    let pdfImgWidth = pageWidth - margin * 2;
                    let pdfImgHeight = pdfImgWidth / ratio;

                    const gridBottomY = margin + 10 +
pdfImgHeight;
                    if (gridBottomY > pageHeight / 2) {
                       pdfImgHeight = pageHeight / 2.5;
                       pdfImgWidth = pdfImgHeight * ratio;
                    }

                    doc.addImage(imgData, 'PNG', (pageWidth -
pdfImgWidth) / 2, margin + 10, pdfImgWidth, pdfImgHeight);

                    // Clues
                    let y = margin + 10 + pdfImgHeight + 10;
```

```
                    doc.setFont('helvetica', 'bold');
                    doc.setFontSize(14);

                    const acrossWords = words.filter(w =>
w.direction === 'across').sort((a,b) => a.number - b.number);
                    const downWords = words.filter(w =>
w.direction === 'down').sort((a,b) => a.number - b.number);

                    const colWidth = (pageWidth - margin*2) / 2
- 5;

                    if (y > pageHeight - 40) { // Check if we
need a new page for clues
                        doc.addPage();
                        y = margin;
                    }

                    doc.text('По горизонталі', margin, y);
                    doc.text('По вертикалі', margin + colWidth +
10, y);
                    y += 7;

                    doc.setFont('helvetica', 'normal');
                    doc.setFontSize(9);

                    let acrossY = y;
                    let downY = y;

                    for (const word of acrossWords) {
                        const text = `${word.number}.
${word.clue}`;
                        const lines = doc.splitTextToSize(text,
colWidth);
                        if (acrossY + lines.length * 4 >
pageHeight - margin) {
                            doc.addPage();
                            acrossY = margin;
                            downY = margin; // Reset both columns
on page break
                        }
                        doc.text(lines, margin, acrossY);
                        acrossY += lines.length * 4;
                    }

                    for (const word of downWords) {
                         const text = `${word.number}.
${word.clue}`;
                        const lines = doc.splitTextToSize(text,
colWidth);
                         if (downY + lines.length * 4 >
pageHeight - margin) {
```

```
                            // This logic gets complex if one
column breaks while other doesn't.
                            // For simplicity, we assume they
fill somewhat evenly.
                            // A more robust solution would track
pages per column.
                        }
                        doc.text(lines, margin + colWidth + 10,
downY);
                        downY += lines.length * 4;
                    }

                    // --- Add Answer Key Page ---
                    doc.addPage();
                    doc.setFont('helvetica', 'bold');
                    doc.setFontSize(22);
                    doc.text('Відповіді', pageWidth / 2, margin,
{ align: 'center' });
                    const answerGridEl = gridEl.cloneNode(true);
                    // Fill in the letters for the answer key
                    const                cells                =
answerGridEl.getElementsByTagName('td');
                    words.forEach(word => {
                        if (word.row !== undefined) { // Check if
word was placed
                            for(let   i=0;   i<word.word.length;
i++) {
                                let cellIndex;
                                if (word.direction === 'across')
{
                                    cellIndex   =   word.row   *
trimmedGrid[0].length + word.col + i;
                                } else {
                                    cellIndex = (word.row + i) *
trimmedGrid[0].length + word.col;
                                }
                                if(cells[cellIndex]           &&
cells[cellIndex].classList.contains('filled')) {
                                    //  Clear  existing  content
(like number spans)
                                    cells[cellIndex].innerHTML =
'';
                                    cells[cellIndex].textContent
= word.word[i];

cells[cellIndex].style.fontSize = '16px';

cells[cellIndex].style.verticalAlign = 'middle';
                                }
                            }
                        }
                    });
```

```
                    document.body.appendChild(answerGridEl);  //
Needs to be in DOM for html2canvas
                    const       answerCanvas       =       await
html2canvas(answerGridEl, { scale: 2 });
                    document.body.removeChild(answerGridEl);

                    const             answerImgData             =
answerCanvas.toDataURL('image/png');
                    doc.addImage(answerImgData,           'PNG',
(pageWidth  -  pdfImgWidth)  /  2,  margin  +  10,  pdfImgWidth,
pdfImgHeight);


                    doc.save('crossword.pdf');
                    statusEl.textContent    =    'PDF    успішно
створено!';
                } catch (error) {
                    console.error("Error    exporting    PDF:",
error);
                    statusEl.textContent = 'Помилка при експорті
в PDF.';
                }
            }

            // --- Event Listeners ---
            addWordBtn.addEventListener('click', addWord);
            wordInput.addEventListener('keypress',  (e)  =>  e.key
=== 'Enter' && clueInput.focus());
            clueInput.addEventListener('keypress',  (e)  =>  e.key
=== 'Enter' && addWord());
            generateBtn.addEventListener('click',
generateCrossword);
            pdfBtn.addEventListener('click', exportToPDF);

            // Initial render
            renderWordList();
        </script>
    </body>
    </html>
```