МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Кафедра комп'ютеризованих систем управління

ДОПУСТИТИ ДО ЗАХИСТУ
Завідувач кафедри

_____Литвиненко О.Є.
"_____"_____2021 р.

ДИПЛОМНИЙ ПРОЄКТ
(ПОЯСНЮВАЛЬНА ЗАПИСКА)

ЗДОБУВАЧА ОСВІТНЬОГО СТУПЕНЯ
"БАКАЛАВР"

Тема: «Інтерактивний електронний підручник для розвитку мислення»

Виконавець: студент, СП-426, Ліннік Дмитро Сергійович

Керівник: старший викладач Сябрук Ігор Миколайович

Нормоконтролер: Тупота Євгеній Вікторович

Київ 2021

НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Факультет Кібербезпеки, комп'ютерної та програмної інженерії
Кафедра комп’ютеризованих систем управління
Спеціальність 123 «Комп'ютерна інженерія»
(шифр, найменування)

Освітньо-професійна програма «Системне програмування»
Форма навчання денна

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Литвиненко О.Є.
«_____»_____2021 р.

ЗАВДАННЯ
на виконання дипломної роботи (проєкту)
Лінніка Дмитра Сергійовича
(прізвище, ім'я, по батькові випускника в родовому відмінку)

1. Тема дипломної роботи (проєкту) «Інтерактивний електронний підручник для розвитку мислення»
затверджена наказом ректора від «04» лютого 2021 р. № 135/ст
2. Термін виконання роботи (проєкту): з 17.05.2021 по 20.06.2021
3. Вихідні дані до роботи (проєкту): інтерактивний електронний підручник для розвитку мислення.
4. Зміст пояснювальної записки:
1. Етапи та особливості розробки електронного підручника.
2. Аналіз існуючих рішень.
3. Створення інтерактивного підручника.
5. Перелік обов'язкового графічного (ілюстративного) матеріалу:
1. Схема алгоритму обробки форми авторизації
2. Діаграма прецедентів
3. Основні вікна додатку

6. Календарний план-графік

№ пор.	Завдання	Термін виконання	Відмітка про виконання
1	Проведення огляду та аналізу існуючих графічних редакторів	17.05.2021– 20.05.2021	
2	Написання першого розділу	20.05.2021– 23.05.2021	
3	Проаналізувати літературні джерела за темою дипломної роботи та основні етапи створення інтерактивного підручника	23.05.2021– 24.05.2021	
4	Написати другий розділ	25.05.2021– 26.05.2021	
5	Реалізувати інтерактивний підручник для розвитку мислення та описати у пояснювальній записці	26.05.2021– 29.05.2021	
6	Оформити пояснювальну записку	29.05.2021– 02.06.2021	
7	Підготовка графічних та ілюстративних матеріалів	02.06.2021– 04.06.2021	

7. Дата видачі завдання: “ 17 ” травня 2021 р.

Керівник дипломної роботи (проєкту) _____ Сябрук І.М.
(підпис керівника) (П.І.Б.)

Завдання прийняв до виконання _____ Ліннік Д.С.
(підпис випускника) (П.І.Б.)

РЕФЕРАТ

Пояснювальна записка до дипломного проекту «Інтерактивний електронний підручник для розвитку мислення»: 42 сторінок, 17 рисунків, 15 літературних джерел, 1 додаток.

Ключові слова: ЕЛЕКТРОННИЙ ПІДРУЧНИК, МОВА ПРОГРАМУВАННЯ *C#*, *XAML*, *WPF*, *LINQ*.

Об’єкт дослідження – електронний підручник для розвитку мислення.

Предмет дослідження – розробка електронного інтерактивного підручника для розвитку мислення.

Мета дипломної роботи – розробити додаток, що являє собою електронний підручник, що міститиме в собі збірник інтерактивних тренажерів для розвитку для мислення. Також тестів для відстежування прогресу користувача при використанні додатку.

Важливість розробки даного продукту полягає у створенні повністю безкоштовного додатку для операційної системи *Windows*, що працюватиме без обов’язкового підключення до мережі Інтернет, з використанням мультимедійних засобів подання інформації та простим, інтуїтивно зрозумілим інтерфейсом.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ, ТЕРМІНІВ

XAML (*EXtensible Application Markup Language*) – розширювана мова розмітки.

LINQ (*Language-Integrated Query*) – мова запитів до джерела даних.

API (*Application Programming Interface*) – програмний інтерфейс додатку.

MSIL (*Microsoft Intermediate Language)* – проміжна мова псевдокоду.

Фреймворк – зовнішній програмний продукт.

ВСТУП

Тема саморозвитку стала неймовірно популярною в останні кілька десятиліть. Особистий розвиток – це зміна мислення, щоб поліпшити аспекти вашого емоційного, фізичного, розумового і духовного стану.

Індустрія особистого розвитку охоплює всі аспекти індивідуальної підготовки до професійного та особистого житті. Підвищена обізнаність про самосвідомість та психічному здоров'ї стимулюватиме попит на програми особистого розвитку. Ці програми допомагають людям аналізувати свої сильні та слабкі сторони, що може поліпшити їх самосвідомість і допомогти їм визначити можливості для особистого зростання та професійного розвитку. Процес пізнання розпочинається з відчуттів та спостережень, щоб пройшовши стадії уявлень і пам'яті, перетвориться в мислення і завершення в процесах прийняття рішень та виконанні дій.

Мислення – це сукупність розумових процесів, що лежать в основі пізнання; до мислення саме відносять активну сторону пізнання: увага, сприйняття, процес асоціацій, утворення понять і суджень. Розвиток цих здібностей необхідний на всіх етапах нашого життя. Для розвитку когнітивних функцій, незалежно від віку, корисно вирішувати головоломки, читати різну літературу, вчитися аналізувати інформацію, знаходити їй нестандартні застосування. Існує велика кількість спеціальних вправ та тренувань, які сприяють з розвитку людини, тренують пам'ять, увагу, розвивають аналітичні здібності та ін.

Мета створення додатку, розробити електронний інтерактивний підручник, що допоможе розвити та вдосконалити мислення користувача базуючись на основі доведення до користувача теоретичних відомостей та постулатів психологічного мислення людини. Додаток допоможе краще зрозуміти роботу мозку, приймати більш правильні рішення, швидше та ефективніше думати.

Створенні інтерактивні тренажери та тести дадуть змогу відстежування та порівняння результатів прогресу розвитку користувача при використанні додатку. Кожен користувач має власний обліковий запис, що може зберігатися локально

(при роботі додатку в режимі офлайн) або на сервері (при роботі додатку в режимі онлайн), дозволяючи переносити досягнутий прогрес на інші пристрої.

Для створення додатку необхідно дослідити основні особливості розробки електронних підручників, розглянути їх структуру та етапи створення. Обрання мови програмування та технологій для створення електронного підручника визначаються в залежності від операційної системи та задач, що виконує додаток.

Основною мовою програмування додатку було обрано об'єктно-орієнтовану мову *C#*, що є ефективним та гнучким механізмом для створення додатків під операційну систему *Windows*, разом з використанням засобів середовища *.NET Framework*. Для створення та організації користувацького інтерфейсу було обрано платформу *Windows Presentation Foundation*. База даних, для збереження облікових записів та досягнень користувачів, створена за допомогою системи управління реляційними базами даних *Microsoft SQL Server* та з використанням мови інтегрованих запитів *Language Integrated Query*, що за допомогою технології *LINQ to Sql* дозволяє отримати доступ до таблиць бази даних безпосередньо з додатка.

Кінцевий продукт повинен являти собою інтерактивний додаток, що включатиме в себе електронну версію друкованої книги, з відтворення тренувальних вправ для розвитку мислення, застосовуючи різноманітні інтерактивні засоби подання інформації. Також додаток включатиме в себе взаємодію з базою даних, що зберігатиме облікові записи кожного користувача. Основною перевагою додатка є можливість роботи без постійного інтернет з’єднання із можливістю збереження досягнень користувача.

РОЗДІЛ 1
АНАЛІЗ ОБЛАСТІ ДОСЛІДЖЕННЯ

1.1. Загальні відомості з розробки комп’ютерних додатків

Комп’ютерна програма – набір інструкцій, що являє собою певний алгоритм дій заданий комп’ютеру. Інструкції можуть подаватися у вигляді певних слів, кодів, команд, або у будь-якому іншому вигляді, виражених у формі, зрозумілій для комп’ютера, які формують послідовність дій для досягнення поставленої мети або результату.

В залежності від своєї функціональності всі комп’ютерні програми можна поділити на системні програмні засоби або прикладні програмні засоби. В основі системних програм лежить операційна система, яка організовує взаємодію між комп'ютерним обладнанням та прикладними програмами. Основа ціль операційної системи – надати середовище, в якому працюватиме прикладна програма. До системних програм також відносять утиліти, які допомагають операційній системі в керуванні комп'ютером. Прикладні програми – це програми, що спрямовані для взаємодії з користувачем, їх робота спрямована на створення та обробку інформації. Для створення комп’ютерних програм використовуються різноманітні мови програмування, що полегшує їх створення та редагування людиною.

1.2. Дослідження предметної галузі

Електронний підручник – комп'ютерний педагогічний програмний засіб, що призначений у першу чергу для подання нової інформації, яка доповню друкарські видання або копіює вже існуючі книги. Основне призначення електронного підручника, організація індивідуального навчання, що супроводжується тестуванням одержаних знання та оцінку вмінь. На сьогодні існує велика кількість

різноманітних варіантів класифікації електронних підручників, але єдиної думки з цього питання і, відповідно, загальної класифікації немає. Однією з найпопулярніших є класифікація, що базується на цілях і завданнях навчальної системи або режимах використання електронного підручника. З даної класифікації виділяються наступні типи електронних посібників: ті, що ілюструють, консультують, операційні середовища, тренажери, навчальний контроль. Як правило, електронний підручник неможливо віднести до одного з типів, так як в більшості випадків, в них реалізовано декілька з вище перерахованих типів із зазначенням режимів його використання.

1.3. Створення електронного додатку

В загальному вигляді концепція створення всіх електронних підручників, незалежно від типу, мають єдину структуру. Насамперед, при розробці електронного підручника приділяють велику увагу психологічним принципам взаємодії людини з комп'ютером. Однією з найважливіших частин навчальної програми є структурованість та гармонійне поєднання дизайну. Якість розробленого дизайну впливає на значну кількість факторів таких як швидкість сприйняття нової інформації її узагальнення й аналіз, швидкість запам'ятовування.

Основні властивості, на які варто звернути увагу в першу чергу, це розташування інформації на сторінці (екрані монітора), послідовність сторінок, створення зручного та інтуїтивно зрозумілого інтерфейсу користувача. Також, не менш важливим є колір інтерфейсу. Як відомо, колір здатен стимулювати різні ділянки головного мозку та гіпофізу, що по різному впливає на роботу нашого мозку. Наприклад, фіолетовий колір відповідає за інтуїцію і знання, допомагає побороти будь-які страхи, а значить допомагає сконцентруватися. Отже, вибір даного кольору позитивно вплине при процесі отриманні знань. На перший погляд це незначні елементи, але вони відіграють дуже важливу роль в усвідомленні й розумінні змісту матеріалу користувачем, в подальшому перенесенні в довготривалу пам'ять для зберігання отриманої інформації та здобутих навичок з метою їх подальшого використання.

1.4. Основні етапи створення електронного підручника

Перший етап – перетворення звичайного друкованого посібника в електронну інформацію. Існує декілька способів виконання цього процесу, так текст підручника може переписуватися в ручну людини з книги в будь-який текстовий редактор, або ж за допомогою технічного пристрою, сканера. В цьому випадку, весь підручник сканується та перетворюється в окремий рисунок. Відсканований матеріал редагується за допомогою різноманітних програмних засобів. Текст, що знаходиться на рисунку перетворюється в електронний текст, а ілюстрації, що наявні в підручнику, відокремлюються та оброблюються окремо.

Другий етап – створення загальної схеми та аналіз наявної інформації. На даному етапі виокремлюється інформація яка буде використана при створенні електронного підручника, так можливе повне або часткове використання друкованої інформації. Також розроблюється загальний алгоритм роботи програми (електронного підручника, навчального комплексу).

Третій етап – розробка та створення дизайну (візуального вигляду). Визначаються основні компоненти та функціональні вимоги. Обирається мова програмування додатку та загальний вигляд на екрані монітору.

Четвертий етап – подання мультимедійного матеріалу підручника. Вставка анімації та звуку до вже створеного додатка.

1.5 Мова програмування *C#*

Мова *C#* розроблена та створена компанією Microsoft є мовою програмування високого рівня. Сьогодні вона є досить популярною та застосовується в різних сферах розробки різноманітних програм, таких як розробка прикладних програм, створення веб-сервісів або мобільних додатків. Основою метою створення даної мови було необхідність задовільнення власних потреб платформи *Microsoft .NET,* але з часом дана мова отримала широке застосування та визнання серед інших розробників.

Розробка даної мови програмування почалась в 1998 році, а перша робоча версія з’явилась у 2001 році. Очолив команду розробників відомий фахівець Андерс Хейлсберг. Підтримка та регулярне оновлення версії *C#* виходять і досі, це додає новий функціонал із розширенням бібліотек, виправлення помилок та незначних доопрацювань, що робить дану мову програмування актуальною і зараз.

Нині *C#* є дуже гнучким та потужним механізмом з розробки програм. Його застосовують для створення як простих веб-додатків, так і для створення потужних програмних систем, завдяки зручному синтаксису та суворому структуруванню, з можливістю застосування великої кількості різноманітних фреймворків та бібліотек.

1.5.1. Зв'язок *C#* із середовищем *.NET Framework*

Мова програмування *C#* самодостатня мова програмування, але незважаючи на це, у нього є особливий взаємозв'язок із середовищем *.NET Framework*. Причинами такого взаємозв'язку пояснюється двома особливостями. По-перше, мова *C#* спершу розроблювалась під середовище виконання *.NET Framework*.

По-друге, всі бібліотеки, що використовуються мовою *C#* визначені в середовищі *.NET Framework*. Отже, можна зробити висновок, що C# і *.NET Framework* тісно пов'язані між собою, хоча C# і можливо використовувати без середовища *.NET Framework*.

1.5.2. Особливості *NET Framework*

NET Framework – середовище для підтримки розробки та виконання розподілених компонентних додатків. Дана технологія організовує спільне використання різних мов програмування в одному проєкті, а також забезпечує безпеку й платформонезалежність програм в загальній моделі програмування для платформи *Windows*. При використанні з мовою *C#,* середовище *.NET Framework* визначає два дуже важливі елементи. Перший елемент загальномовне середовище виконання (*Common Language Runtime – CLR*). Дана система керує ходом виконання програми та підтримує багатомовне програмування. Другим елементом

середовища *.NET Framework* є бібліотека класів, що надає програмі доступ до середовища виконання. Так, якщо програма використовує лише бібліотеки класів *.NET*, то таку програму на C# можна виконати будь-де, де підтримується середовище *.NET Framework*.

1.5.3. Взаємодія середовище *CLR* з *.NET Framework*

Середовище *CLR* виконує управлінські функції над виконанням коду *.NET*. Воно діє за наступним принципом. Після завершення компіляції програми на мові *C#* створюється файл, що містить проміжний псевдокод, названий *Microsoft Intermediate Language, MSIL* (проміжна мова *Microsoft*). За допомогою створеного псевдокоду *MSIL* визначається набір сумісних інструкцій, які використовуються незалежно від моделі обраного процесора.

Призначення *CLR* – перетворити проміжний код у виконуваний код по ходу виконання програми. Дана особливість надає змогу використовувати скомпільований псевдокод *MSIL*, в будь-якому середовищі, де є реалізація *CLR*. За допомогою даного механізму досягається часткова сумісність в середовищі *.NET Framework*. Перетворення коду виконується середовищем *CLR*, що активізує процеси *JIT*-компілятора, який перетворює псевдокод *MSIL* у власний код системи для кожної частини програми. Отже, всі програми створенні мовою програмування *C#* фактично виконується як власний код, незважаючи на те, що спочатку вона скомпільована в псевдокод *MSIL*. Використання даного механізму, дозволяє отримати всі переваги платформонезалежності псевдокоду *MSIL*, без втрат у швидкості програми.

При використанні даної технології, середовище *CLR* здійснює контроль при розподілу пам'яті, організації робочого процесу, виконання коду, виконує перевірку безпеки коду в різних системних процесах. Кожній компоненті внутрішньому процесу керованого коду присвоюється різні ступені довіри (наприклад, Інтернет, корпоративну мережу або локальний комп'ютер).Так встановлюється доступ до операцій, файлів, механізму роботи з реєстром або інших важливих функцій, незважаючи на те в якому додатку він використовується.

1.5.4. Об'єктно-орієнтоване програмування

Основним поняттям *C#* є об'єктно-орієнтоване програмування (ООП). *C#* є об'єктно-орієнтованою мовою програмування, саме тому всі програми створенні на *C#* містять в собі принципи об'єктно-орієнтованого програмування.

Основний принцип ООП: "дані керують доступом до коду". Це означає, що при використанні методики об'єктно-орієнтованого програмування при створені програми визначаються дані та код, що має доступ до цих даних. На практиці це означає, що тип даних повинен точно визначати операції, які можуть бути виконані над даними. В загальному вигляді всі мови програмування, що підпорядковуються принципам ООП, в тому числі і *C#*, мають три загальні властивості: інкапсуляція, поліморфізм та наслідування.

1.5.4.1. Інкапсуляція

Інкапсуляція – це механізм програмування, який об'єднує код і дані, якими він маніпулює. Використання даного механізму захищає код від небажаного втручання ззовні та неправильного використання даних. Це дозволяє помістити дані та код в так званий автономний чорний ящик, всередині якого буде захищено від небажаного зовнішнього втручання весь необхідний код та дані. При поєднанні коду та даних в такий спосіб, утворюється об'єкт.

Внутрішнє тіло об'єкта може містить код, дані або ж і те й інше, доступ до яких може бути встановлений закритий або ж відкритий. При встановленні даних або коду закритими, вони становляться видимі та доступні лише в об'єкті в якому об’явленні, що робить їх використання неможливим за межами об'єкту. Якщо доступ до даних або коду встановлений як відкритий, то вони доступні іншим частинам програми, хоча і об’явленні всередині об'єкта. Загалом, відкриті частини об'єкта використовуються для організації інтерфейсу.

Яскравим прикладом застосування інкапсуляції в *C#* є клас, що визначає форму об'єкта. В ньому описуються дані та код, який буде ними оперувати. Дані,

що описуються в класі, називають полями. А код, який ними оперує, методами. Отже, доступ до полів класу здійснюється через методи, цього самого класу.

1.5.4.2. Поліморфізм

Поліморфізм, що з грецької означає "безліч форм". Дана властивість дозволяє одному інтерфейсу отримувати доступ до загального класу дій. Для прикладу розглянемо стек, тобто область пам'яті, що функціонує за принципом "останнім прийшов - першим обслужений". Припустимо, що в програмі потрібні три різні типи стеків: один - для цілих значень, інший - для значень з плаваючою точкою, третій - для символьних значень. Використаємо один єдиний алгоритм для реалізації всіх вище перерахованих стеків, що зберігають різнотипні данні. При використанні мови, що не підпорядковується принципам об'єктно-орієнтованого програмування, для цієї мети довелося б створити три різних набору стекових підпрограм з різними іменами. Саме завдяки поліморфізму досягається використання одного загального набору підпрограм для реалізації всіх трьох типів стеків.

В загальному вигляді поняття поліморфізму нерідко виражається в такий спосіб: "один інтерфейс - безліч методів". Використання поліморфізму допомагає спростити програму, використовуючи один і той же інтерфейс для опису загального класу дій. Компілятор самостійно обирає метод, що підходить для кожного окремого випадку. Це полегшує роботу програміста, адже йому не потрібно визивати кожен метод самостійно, а достатньо правильно використати загальний інтерфейс.

1.5.4.3. Наслідування

Наслідування в програмуванні являє собою процес, в ході якого один об'єкт набуває властивостей іншого об'єкта. Цей процес забезпечує принцип ієрархічної класифікації.

Саме завдяки ієрархіє, стає можливим багаторазове використання властивостей описаних в одному об’єкті без необхідності повторного визначення властивостей в іншому об’єкті. Застосовуючи наслідування, потрібно визначити тільки ті властивості які притаманні розглянутому об’єкту. Наступний об'єкт успадковує властивості свого батька. При використанні механізму наслідування один об'єкт стає окремим екземпляром більш загального класу.

1.6 *Windows Presentation Foundation*

Windows Presentation Foundation (*WPF*) – це *API*-інтерфейс, призначений для створення додатків, який інтегрує різноманітні *API*-інтерфейси в єдину об'єктну модель і забезпечує чіткий поділ завдяки розмітці *XAML*. Використовуючи *WPF* та мову розмітки *XAML* можна створювати та налаштовувати елементи управління, графіку, тривимірні зображення й анімації.

Нижче коротко перераховані основні функціональні можливості *WPF*.

- Велика кількість механізмів компонування, що забезпечує гнучкість при розміщенні та позиціонуванні елементів.
- Розширений механізм прив'язки даних для користувацького інтерфейсу.
- Різноманітний набір стилів для додатка *WPF*.
- Векторна графіка з підтримкою автоматичної зміни розміру в залежності від екрану монітора.
- Підтримка двовимірної і тривимірної графіки, анімації, а також відтворення відео та аудіо.
- Підтримка інших елементів управління користувацького інтерфейсу

Наприклад, при розробці програми з використанням *WPF* можна використовувати спеціальні елементи управління *Windows Forms*.

1.7 Висновки до розділу

В першому розділі досліджено основні етапи створення електронного підручника. Розглянуто його загальна структура, класифікації та види. Електронний інтерактивні підручник та навчальні комплекси відносяться до

складних програмних продуктів, що мають достатньо складну структуру та потребують для своєї реалізації не тільки засобів комп'ютерної графіки, але й певне технічне обладнання. Додатки даного типу, подають інформацію як в текстовому, так і ілюстративну вигляді. Ілюстративний матеріал демонструє процеси, які неможливо відобразити при використанні друкованих посібників. Тематика електронного підручника розроблюється відповідно до обраного програмного курсу. При розробці електронного додатку потрібно приділити особливу увагу інтерфейс користувача, який повинен мати зручне та продумане меню. Кожна тема описана в додатку повинна міститься в окремому розділі. Навчальна програма розроблюється як відкрита модульна система, що надає можливість вносити зміни та доповнювати існуючу інформацію при виникненні такої необхідності в будь-який момент часу.

РОЗДІЛ 2
АНАЛІЗ СТРУКТУРИ ЕЛЕКТРОННИХ ПІДРУЧНИКІВ

2.1 Основні елементи структури додатку

В загальному вигляді, додаток, містить певну кількість тренажерів, кожен з яких умовно належати до певного класу вправ, що вдосконалюють відповідну область мислення (Швидкість, Пам'ять, Увага). Кожен з тренажерів має певні обмеження в часі або кількості неправильних відповідей, а деякі з вправ мають різну складність, що поступово збільшується. Після проходження кожного з тренажерів, накопичується певна кількість балів, що вираховується базуючись на швидкості виконання вправи. Бали використовуються для відстеження та збереження прогресу користувача. Додаток працює у двох режимах збереження даних, офлайн (дані про обліковий запис зберігаються на комп'ютері користувача) або онлайн (дані про обліковий запис зберігаються в базі даних і на комп'ютері користувача, в разі подальшого використання облікового запису офлайн). Також користувачеві надаються ознайомчі статті запозичені з книги Гарета Мура «Тренажер мозку: Як розвинути гнучкість мислення за 40 днів», з програмно відтвореною інтерпретацією вправ, поданих наприкінці кожного розділу книги.

Загальна логіка роботи додатку:

- Користувач, авторизується в одному з режимів додатку (онлайн або офлайн), після чого система завантажує обліковий запис.
- Користувач, потрапляє на головне меню додатку, в якому обирає один із тренажерів, який він хоче виконати.
- Додаток, відображає короткі теоретичні відомості про обрану вправу та інструкції щодо її виконання. Закінчивши виконання тренування, відображається кінцевий результат, та відбувається повернення до головного меню додатку, де

виконується обрахування загальної кількості набраних балів та їх збереження, для використання в наступних сесіях.

- Після проходження певної кількості тренувань, додатком буде запропоновано пройти тестування за таблицею Шульте, що складається з п’яти етапів. На основі результатів буде побудовано діаграму, що базується на трьох критеріях (Ефективність роботи, Ступінь опрацювання, Психологічна стійкість).

2.2 Огляд існуючих аналогів

Існує два найпопулярніших тренажерів для розвитку мислення Фоксворд та Викиум.

Фоксфорд – платформа, що позиціонує себе як онлайн-школа для учнів 1-11 класів, вчителів та батьків. Даний додаток включає велику кількість онлайн-курсів та індивідуальних занять з репетитором на яких школяря готують до вступних іспитів, олімпіад, вивчають шкільні предмети.

Система підтримує багатоплатформність, що дозволяє використовувати додаток на будь-якому пристрої: комп'ютер, планшет, смартфон. Фоксфорд містить різноманітні варіанти навчання: курси для школярів і вчителів, індивідуальний репетитор, участь в олімпіадах, домашня школа, дитячі табори. Додаток має навчальний матеріал, призначений для вивчення, за темами дисциплін освітніх установ (шкіл), що містить текстовий, графічний і аудіовізуальний контент.

Механізм реєстрації досить простий, платформа підтримує авторизацію з великої кількості різноманітних соціальних мереж або *Google*-аккаунта.

Викиум – онлайн-платформа створена для покращення діяльності мозку і спрямована на розвиток таких когнітивних функцій: уваги, пам'яті, мислення за допомогою онлайн-тренажерів і спеціалізованих курсів. В основі кожного тренажеру лежать різноманітні методики тренування пізнавальних процесів, наукові знання про структуру сприйняття, увагу, пам'ять, мислення.

До кожного користувача система має власний підхід, кожен день генеруючи індивідуальну програму розвитку. Вона базується на основі результатів отриманих після проходження вступного тестування.

Результат онлайн-тренажеру фіксується на кожному етапі його виконання, визначається мозкова активність до, під час і після проходження тренування. Аналізуючи отримані результати формується висновок про ефективність виконання тренажера.

Система містить режим дуелі – це можливість випробувати свій інтелектуальний потенціал і змагатися на тренажерах з друзями або випадковим суперником.

Платформа має преміальний доступ. Це платна версія профілю, яку можна підключити на певний період або назавжди. В дуелях преміум-користувач має право самостійно обрати тренажер, що надає певну перевагу над звичайним користувачем.

Преміум-користувачам доступна повна щоденна програма для саморозвитку, додаткові програми розвитку, а також необмежена кількість дуелей і підходів до всіх тренажерів. Преміум-профіль дозволяє бачити повну статистику тренувань, а також порівнювати свої досягнення з іншими користувачами.

2.3 Вимоги до створення електронних підручників

Проаналізувавши існуючі додатки, можна виділити основні вимоги, які слід дотримуватися при розробці електронного підручника. Насамперед вся інформації, що подана у посібнику повинна бути добре структурованою, а всі фрагменти розглянутого напряму закінченими. Кожен елемент мультимедійного напряму, будь-то аудіозапис, відео або інтерактивний тренажер, повинен містити в собі елемент інтерфейсу для повторного відтворювання. Перед виконанням тренувальної вправи, потрібно ознайомити користувача з детальною інструкцією, щодо її виконання, при необхідності створити демонстраційний приклад виконання тренажеру. Після проходження певної частини курсу проводити тестування, для оцінки якості засвоєння розглянутого матеріалу або оцінки ефективності розвитку користувача при використанні додатку.

Тест (в перекладі з англійської "випробування") – певний набір завдань для оцінки знань, якостей та навичок. Розрізняються наступні види завдань:

- Завдання з педагогічного тестування. Використовується для оцінки знань та навичок.
- Завдання з психологічного тестування. Використовується для оцінки певних якостей та здібностей.

Загалом, всі завдання можуть бути подані словесно (у вигляді тексту) або усно (у вигляді картинок, відео, аудіо або інших мультимедійних форматах). На кожний тест накладаються певні обмеження, наприклад, часові або на кількість допущених помилок.

2.4 Висновки до розділу

В другому розділі розглянуто основу структуру та описано особливості дипломного додатку. Що фактично являє собою електронний підручник з набором різноманітних програмованих тренажерів для розвитку мислення. Також, додаток, являє собою, часткову інтерпретацію книги Гарета Мура «Тренажер мозку: Як розвинути гнучкість мислення за 40 днів», в якій описану щоденно тренувальну програму розвитку мислення. По ходу виконання тренувань, користувачеві відкриваються нові та більш складні завдання. Весь прогрес користувача зберігається локально, це не потребує підключення до мережі інтернет, або ж в базі даних, що надає змогу переносити досягнутий прогрес на інші пристрої.

В розділі розглянуто додатки схожого типу, які є найбільш популярними. Електронні інтерактивні підручники та тренажери не мають єдино правильної концепції, й розроблюються відповідно до розглянутої предметної області. Кожен додаток є унікальним та має свої переваги та недоліки в порівнянні з іншими. Більшість додатків мають безкоштовну основу із застосуванням преміальних облікових записів, що мають значну перевагу над звичайним та вимагають постійного й безперервного підключення до мережі інтернет.

Розглянувши основні тенденції в сучасних розвиваючих додатках можна прийти до висновку, що створення електронного інтерактивного підручника, який поєднує в собі всі переваги викладення та подання інформації на рівні з

друкованою книгою, разом з інтерактивністю та багаторазовістю виконання різноманітних тренажерів та тестувань є дуже актуальним в наш час.

РОЗДІЛ 3
СТВОРЕННЯ ЕЛЕКТРОННОГО ПІДРУЧНИКА

3.1. Основні використанні компоненти

Проєкт розроблений на мові *C#*, для створення та організації інтерфейсу користувача використано *Windows Presentation Foundation* (*WPF*) та *.NET Framework 4.7.2* в середовищі розробки *Microsoft Visual Studio 2019*. В проєкті використовується, власна база даних, яка зберігає певну інформацію про користувача. Організація адміністрування та роботи з базую даних виконується за допомогою *SQL Server 2019* та засобами *LINQ*.

Microsoft Visual Studio – це програмне середовище з розробки додатків для операційної системи *Windows*, що дозволяє створювати як консольні програми, так і з графічним інтерфейсом.

Функціональна структура середовища включає в себе:

- редактор вихідного коду;
- відладчик коду;
- редактор форм, для спрощеного конструювання графічних інтерфейсів;
- веб-редактор;
- дизайнер класів;
- дизайнер схем баз даних;

SQL Server Management Studio (*SSMS*) – це безкоштовне графічне середовище для управління інфраструктурою *SQL Server*, розроблене компанією *Microsoft*. *SQL Server Management Studio* призначений як для розробників, так і для адміністраторів серверів. *SQL Server* – основа платформи обробки даних *Microsoft*, що забезпечує хорошу надійність та продуктивність при обробці інформації різнотипних даних.

C# є основною мовою розробки програм на платформі *.NET* корпорації

Microsoft. У ньому вдало поєднуються випробувані засоби програмування з самими останніми нововведеннями та надається можливість для ефективного і дуже практичного написання програм, призначених для обчислювального середовища сучасних підприємств.

LINQ (*Language-Integrated Query*) – мова інтегрованих запитів. Це поняття охоплює ряд засобів, що дозволяють отримувати інформацію з джерела даних. *LINQ* доповнює *C#* засобами, що дозволяють формувати запити для будь-якого *LINQ*-сумісного джерела даних. При цьому синтаксис, який використовується для формування запитів, залишається незмінним, незалежно від типу джерела даних. Це, зокрема, означає, що синтаксис, що потребується для формування запиту до реляційної бази даних, практично нічим не відрізняється від синтаксису запиту даних, що зберігаються в масиві.

Windows Presentation Foundation (*WPF*) – технологія реалізації користувальницьких інтерфейсів для *Windows*-додатків. *WPF* надає змогу поєднувати в одній програмі традиційні інтерфейси, тривимірну графіку, аудіо і відео та анімацію. Користувацький інтерфейс, створений за технологією *WPF*, фактично визначається мовою розмітки *Extensible Application Markup Language*, який називають *XAML*. Ця мова заснована на мові *XML* і спеціально розроблена компанією *Microsoft* для використання в системі *WPF*.

Для створення інтерфейсу використовуються додаткові набори дизайну, розробленого під використання в *WPF* див. рис. 3.1 (*MaterialDesignColors* та *MaterialDesignThemes*). Щоб завантажити даний додаток, в середовищі *Microsoft Visual Studio 2019*, потрібно скористатися системою управління пакетами *NuGet*. *NuGet* розроблена компанією *Microsoft* під платформу *Windows*, в першу чергу для бібліотек *.NET Framework*.

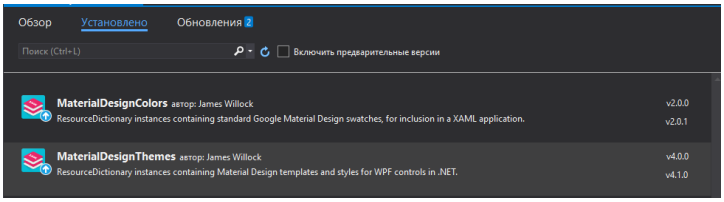

Рис.3.1. Додатковий набір дизайну інтерфейсу

3.2. Розробка додатку

Структура файлів програми:

Registry.xaml / Registry.xaml.cs

MainWindow.xaml/MainWindow.xaml.cs

CardNumber.xaml/CardNumber.xaml.cs

Colors.xaml/Colors.xaml.cs

Comparison.xaml/Comparison.xaml.cs

EndGame.xaml/EndGame.xaml.cs

Mosaic.xaml/Mosaic.xaml.cs

SchulteTest.xaml/SchulteTest.xaml.cs

При запуску додатку користувача зустрічає вікно реєстрації/авторизації. Дизайн вікна міститься в файлі *Registry.xaml*, а програмована логіка міститься в файлі *Registry.xaml.cs*. Користувачу пропонується авторизуватися в додатку, або ж пройти реєстрацію, в разі відсутності профілю. При реєстрації програма перевіряє коректність чотирьох заповнених полів: логіну, паролю, підтвердження паролю, електронної адреси(необхідно для відновлення профілю в разі втрати логіна або паролю). При успішній перевірці даних, програма здійснює запит до таблиці в БД, де зберігаються дані про профіль користувача. Надалі, профіль, що зберігається в БД, може бути використаний, в разі проходження авторизації, на іншому пристрої, що дозволяє перенести досягнутий прогрес, проходження тренажерів. Після виконані авторизації (онлайн або офлайн), в кореневому каталозі програми буде створений бінарний файл під назвою *Userdate.dat,* який локально зберігатиме прогрес користувача, при онлайн авторизації дані файлу синхронізуються з даними в БД.

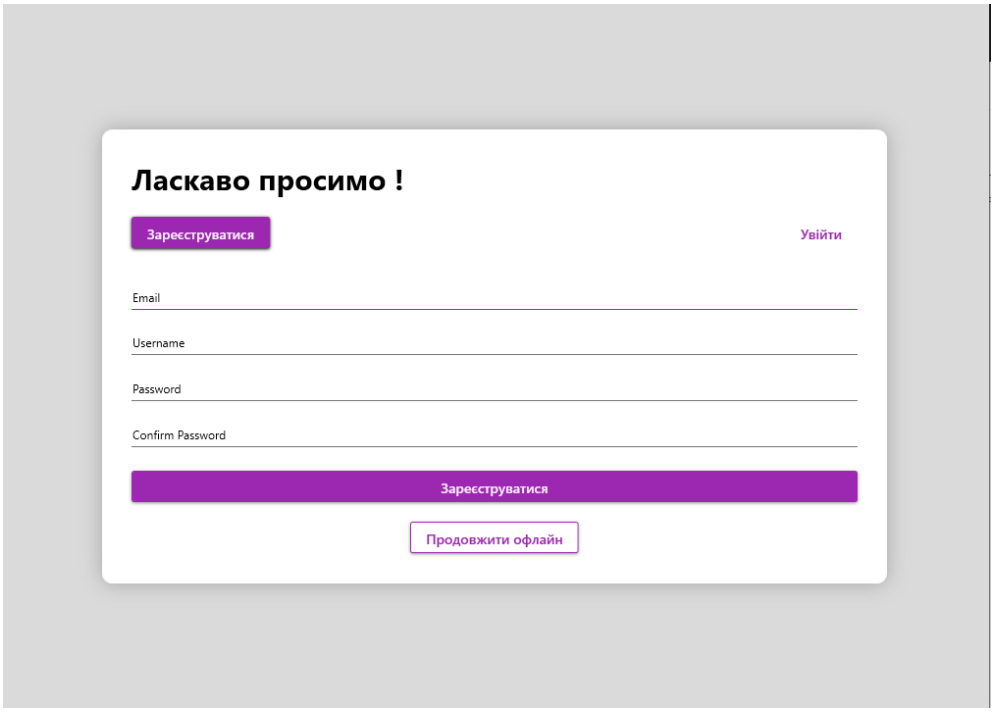

Рис.3.2. Вікно реєстрації

Після проходження авторизації, при першому запуску програми, буде відкрито вікно головного меню програми (Рис. 3.2.) та автоматично створений бінарний файл під назвою *UserConfig.dat*, в якому зберігатиметься прогрес проходження щоденного набору тренажерів, та індивідуальні налаштування користувача, такі як зображуваний аватар, ім’я користувача та встановлений режим контрою тренувань.

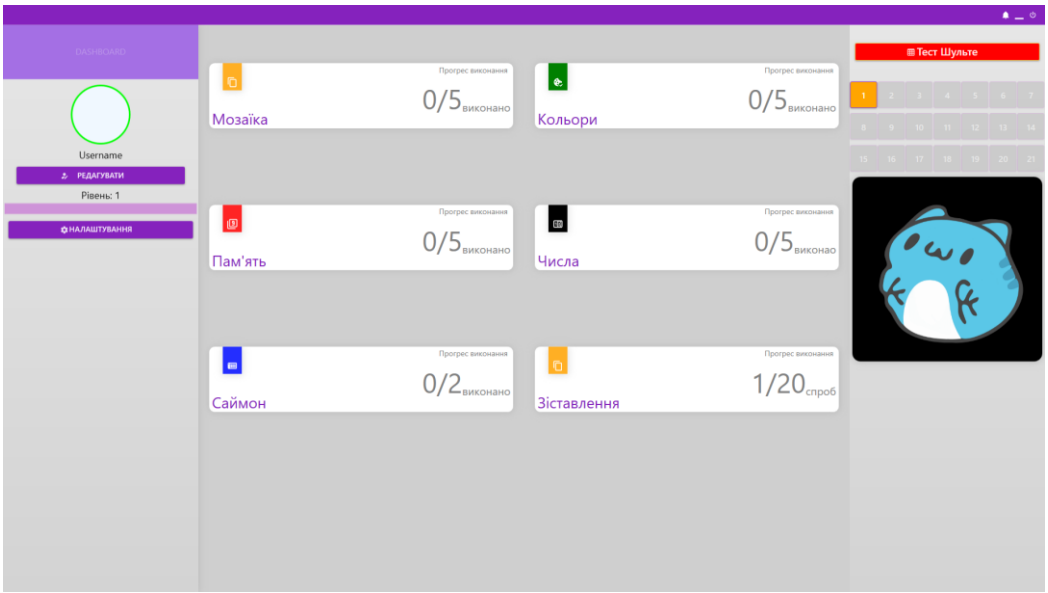

Рис. 3.3. Головне меню додатку

Головне вікно додатку, як і всі інші вікна створенні в програмі будується на основі *XAML* розмітки. У всіх вікнах програми для розташування та позиціювання елементів управління використовується контейнер *Grid*.

Grid – це один з найскладніших та найпопулярніших контейнерів, що використовуються при створенні додатків використовуючи мову розмітки *XAML*. *Grid* може містити безліч рядків і стовпців. З його допомогою можна задати висоту рядків та ширину стовпців в абсолютних пікселях або у відсотках від загального розміру, або встановлюєте значення *auto*, яке автоматично задає розміри рядків або стовпців в залежності від змісту.

Головне вікно, поділене на три стовпці та два рядки. За допомогою властивостей вікна *Window* в розмітці *XAML* використовуються наступні властивості:

WndowStyle – визначає зовнішній вигляд границі вікна. Можливі значення: *SingleBorderWindow* (за замовчуванням), *ToolWindow* (зображується тонка межа) і *None* (візуалізується дуже тонка межа без області для рядка заголовка).

WindowResize – визначає, чи може користувач змінювати розміри вікна. Також впливає на видимість кнопок, що відповідають за розгортання і згортання вікна. Для повного блокування зміни розміру вікна, використовується значення *NoResize*. Щоб користувач міг тільки згортати вікно, використовується значення *CanMinimize*. Щоб користувач міг змінювати розмір вікна усіма можливими способами, вказуєте значення *CanResize*.

При встановленні *WndowStyle = None* та *WindowResize = NonResize*, прибрано елементи управління вікна, присутні в усіх стандартних додатках на платформі *Windows*. Данні органи управління були відтворені програмно на першому рядку, в правому кутку, контейнера *Grid*.

Складові головного меню

Головне меню програми зображене на рис. 3.3 складається з трьох частин.

Ліва панель меню відображає аватар обраний користувачем, ім’я користувача та його рівень з індикатором виконання, що показує поточну кількість набраних

балів та необхідну границю балів для наступного рівня, для цього використовується елемент *ProgressBar*.

ProgressBar являє собою індикатор, що відображає прогрес виконання певного процесу. Він містить властивість *Orientation*, яка дозволяє змінювати розташування індикатора у просторі. Для зв'язку з процесами вся логіка зміни індикатора, використовується властивість *Value*, яка заповнює тіло індикатор та повинна оброблюватися в коді. Також для повного функціонування потрібно встановлювати межі, за допомогою властивостей *Maximum* та *Minimum*, що відповідають числовим межам мінімуму та максимуму відповідно.

Також на панелі наявні дві кнопки, «Редагувати» та «Налаштування», для яких використовується елемент *Button*. Що являти собою елемент керування «Кнопка *Windows*», який буде реагувати на подію натискання на неї (*Click*).

При натисканні першої, відкривається вікно в якому користувачу надається змога змінити відображене ім’я та обрати аватар. При натисканні кнопки «Налаштування» користувач може відключити програмний контроль виконання щоденних тренувань, що наддасть змогу проходити щоденні тренування незалежно, від результатів минулих тренажерів, та обнулити щоденну статистику, проходження тренажерів.

Умовна названа центральна частина меню, містить основний набір тренажерів, проходження яких є обов'язковим, для відкриття нового для тренування. Кожен тренажер візуально розташований в окремому блоці, створеного за допомогою графічного елементу двомірної графіки *Rectangle* (прямокутник). В тілі якого розташована назва тренажеру, індивідуальні графічна іконка та *TextBlock* (текстовий елемент управління), що відображає прогрес виконання.

TextBlock – елемент розмітки, що використовується для відображення текстової інформації та створення простих написів. Основною властивістю даного елементу є властивість *Text*, що задає текстовий зміст. Даному елементу притаманні всі загальні властивості, що використовуються для стилізації тексту, такі як *FontFamily* (шрифт), *FontSize* (розмір шрифту) та інші, що дозволяють

налаштувати зовнішній вигляд тексту. Також використовуючи властивість *TextWrapping* задається спосіб перенесення тексту та *TextAlignment*, що встановлює вирівнювання тексту в залежності від заданого параметру: по лівому краю (*Left*), по центру (*Center*), по правому краю (*Right*). .

Права частина меню, відображає двадцять одну кнопку, що являє собою вісімнадцять унікальних тренажерів, що фактично є вибірковою інтерпретацією книги «Тренажер мозку: Як розвинути гнучкість мислення за 40 днів» Гарета Мура. З цієї книги запозичено вісімнадцять статей, що описують основі процеси формування та розвитку мислення, наприкінці яких пропонується проходження унікальних тренувань. В кінці кожного тижня, користувачеві пропонується пройти набір тренувань по таблиці Шульте, що надає змогу відстежувати прогрес розвитку мислення. Перехід до наступної статті, при увімкненому режимі щоденного контролю, проходить після виконання тренування в кінці минулої статті та успішного проходження заданої кількості тренажерів, що знаходяться в центрі головного меню. Також на панелі знаходиться елемент *MediaElement*, що використовується для програвання певної зацикленої анімації з файлу розширенням *.gif*, що періодично змінюється в залежності від прогресу виконання тренувань користувачем.

MediaElement – елемент, який слугує оболонкою функціональності класу *MediaPlayer*. Як і всі елементи, *MediaElement* розміщується у заздалегідь визначеному місці користувацького інтерфейсу. При використанні *MediaElement* для відтворення аудіо його розташування не має значення, але якщо відтворюється відео, він повинен бути розміщений там, де планується зображуватися відео-вікно. Основними властивостями даного елементу є властивість *Source*, що задає джерело даних та *AutoPlay* при встановленні значенні *true* аудіо або відео почне автоматично програватися після завантаження сторінки. Головними методами є *Play* () – запускає програвання медійного елементу, *Pause* () – тимчасово призупиняє та *Stop* () – повністю зупиняє програвання відео.

3.3. Вимоги до програмного та технічного забезпечення

Для повноцінного функціонування програми достатньо комп’ютера з наступними технічними характеристиками:

Процесор – не менше 1 ГГц.

Оперативна пам’ять – від 512 Мбайт.

Вільний простір на жорсткому диску – 18 Мбайт.

Відеоадаптер: *DirectX* 9 або новішої версії з драйвером *WDDM* 1.0.

Екран: 1024 x 768.

Для роботи системи необхідно встановити операційну систему – версією не нижче *Windows* 7 та *.NET Framework*, версії не нижче 4.0.

3.4. Керівництво користувача

Для запуску програмного додатка потрібно запустити файл AssemblyBuild0705.exe, двічі натиснувши лівою кнопкою миші по імені файлу в каталозі проєкт або ж натиснути правою кнопкою миші по файлу, у відображеному списку обрати пункт «Відкрити». Після запуску, відкриється вікно авторизації в якому необхідно здійснити вхід в систему, при наявності існуючого профілю або ж пройти реєстрацію нового облікового запису.

Також в разі відсутності підключення до мережі інтернет або ж небажання створення облікового запису надається можливість продовжити роботу з додатком в режимі офлайн, натиснувши відповідну кнопку. В разі роботи додатку в режимі офлайн, зберігаються локально на комп’ютері користувача.

Після успішного проходження авторизації, відбувається перехід до головного вікна програми. В лівій частині головного вікна відображається ім’я користувача, поточний рівень та індикатор виконання, що показує набрану кількість балів по відношенню до межі наступного рівня. Під зображенням профілю знаходиться кнопки редагування профілю, а також кнопка налаштування.

В центрі головного вікна зібрані тренажери, виконання яких відкриває доступ до нових тренувань. На кожному з тренажерів міститься індикатор виконання.

Наприклад позначка типу 2/5 свідчить, що для повного завершення тренувального дня потрібно виконати ще три вдалі спроби вказаного тренажеру.

Для повного завершення тренувального дня потрібно виконати певну кількість вдалих спроб на всіх тренажерах та виконати тренування в кінці кожного тренувального дня.

Після закінчення тренувального дня, відкривається доступ до нового щоденного тренування, а індикатори на тренажерах оновлюються.

Всі тренувальні дні відображаються в правій частині головного меню, у вигляді календаря, що складається з двадцяти одного дня. Для того, щоб відкрити новий тренувальний день, необхідно і достатньо завершити прочитання статті, виконати тренування наприкінці та вдало завершити проходження тренажерів. Новий доступний тренувальний день буде окрашений у відповідний колір, як на рис.3.4. В залежності від прогресу проходження тренувального дня, змінюється відображувана анімація.

Рис.3.4. Панель щоденних тренувань

Всі тренажери можна умовно віднести до певного класу вправ, що вдосконалюють відповідну область мислення (Швидкість, Пам'ять, Увага). Кожен з тренажерів має певні обмеження в часі або кількості неправильних відповідей, а деякі з вправ мають різну складність, що поступово збільшується.

Розглянемо кожний з тренажерів більш детально.

Тренажер «Мозаїка» – потрібно запам’ятати, які з клітинок мають зелений колір, після чого вони приймуть свій звичайний вигляд. Далі потрібно обрати клітинки, що були окрашені. Дане тренування допоможе легше засвоювати новий матеріал, розвиває логічні та аналітичні здібності.

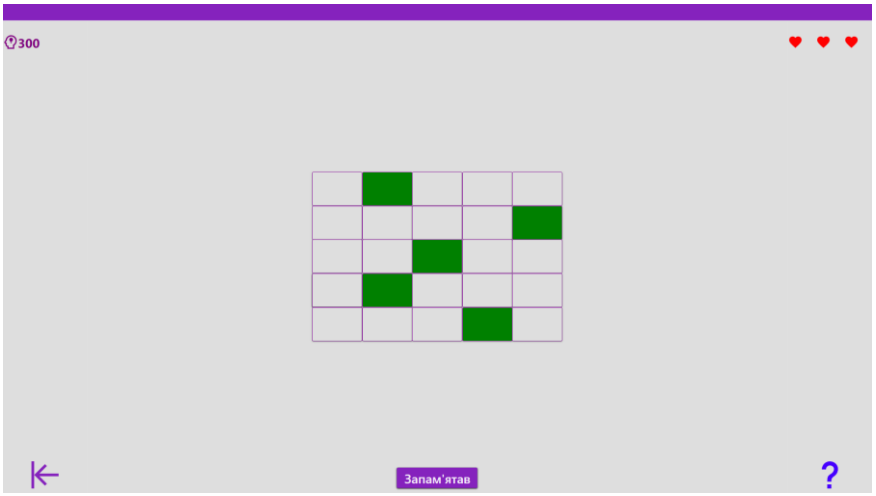

Рис.3.5. Тренажер «Мозаїка»

Тренажер «Кольори» – потрібно, за відведений час, обрати коло відповідного кольору, назва якого подана в буквенному вигляді.

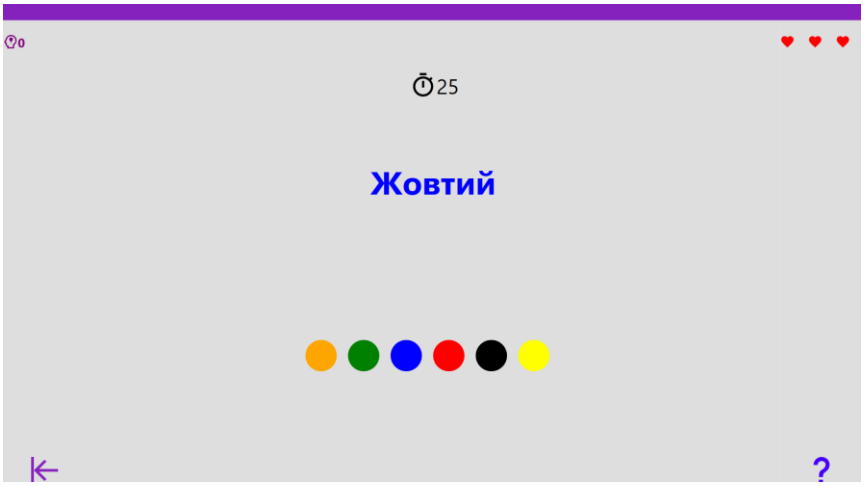

Рис.3.6. Тренажер «Кольори»

Тренажер «Запам'ятай порядок» в ігровій формі розвиває механічну пам'ять. Користувачеві, потрібно запам’ятати та обрати всі клітинки в порядку збільшення, після обрання першої клітинки, числа з інших зникають, й потрібно відтворити їх послідовність по пам’яті. При вдалому виконанні, складність підвищується, показана матриця й число клітинок для запам’ятовування збільшується, доки користувач не почне помилятися. В процесі тренування мозок вчиться групувати і структурувати отриману інформацію.

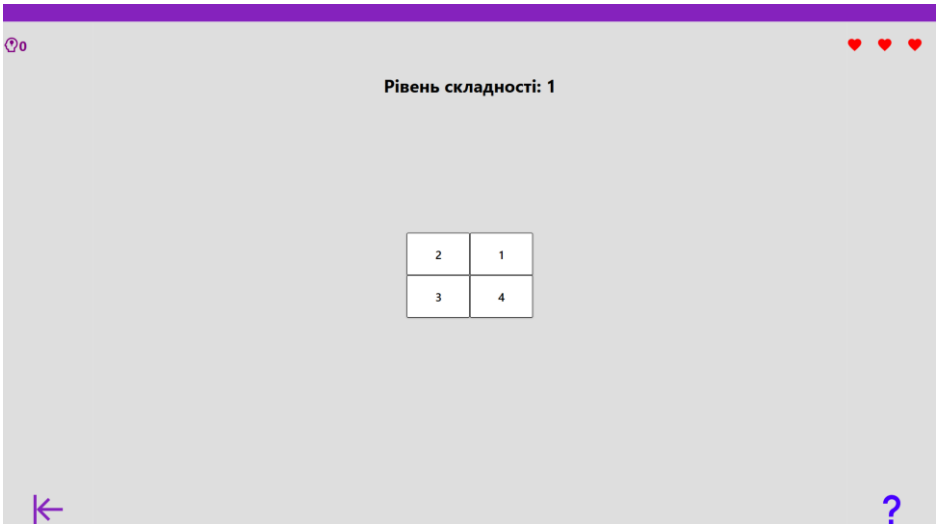

Рис.3.7. Тренажер «Запам'ятай порядок»

Тренажер «Число» – потрібно, за відведений час, встановити парність або непарність числа в центрі. В інтерфейсі користувача наявні дві стрілки, які використовуються для встановлення відповідності чисел (ліва – непарне, права – парне).

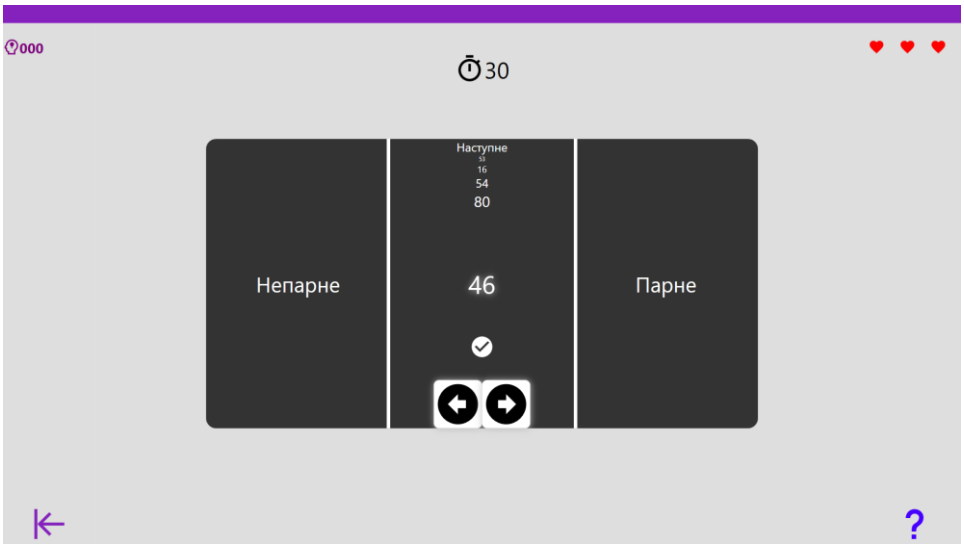

Рис.3.8. Тренажер «Число»

Тренажер «Саймон» – на екрані з’являється певна послідовність клітинок, з часовим інтервалом в одну секунду, після чого користувачеві потрібно відтворити показану послідовність. При вдалому виконані, збільшується складність, кількість активних клітинок послідовності збільшується, доки користувач не почне помилятися.

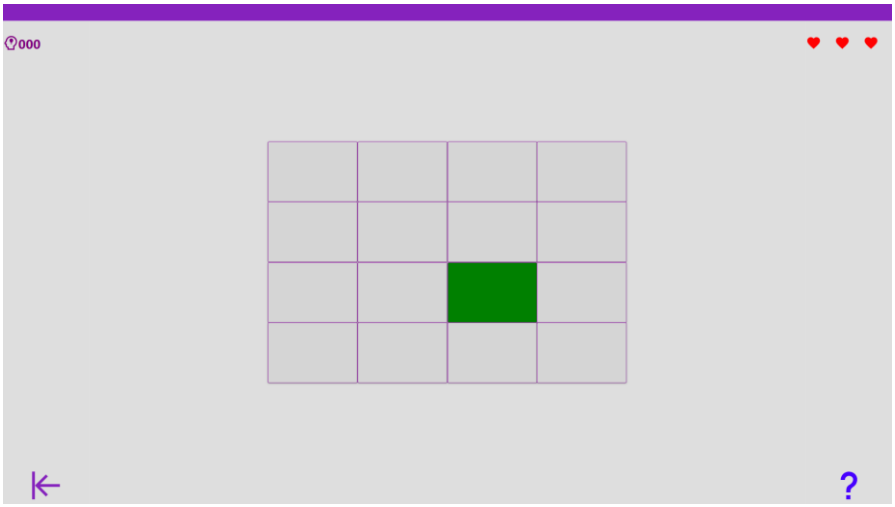

Рис.3.9. Тренажер «Саймон»

Тренажер «Зіставлення» – на екрані з’являється дві таблиці зі знаком «+» або «-». Користувачеві потрібно зіставити дві таблиці та обрати один з чотирьох запропонованих варіантів. При вдалому виконані завдання, складність збільшується, кількість таблиць які потрібно зіставити збільшується.

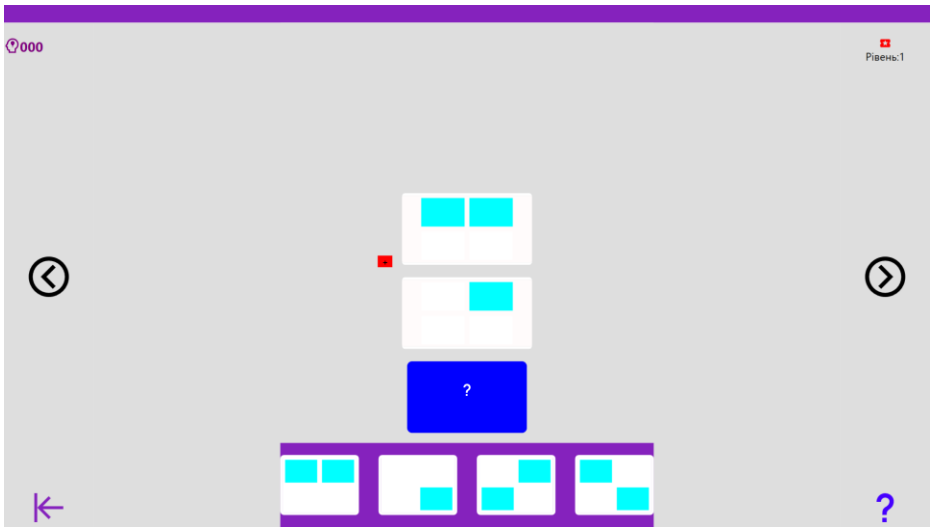

Рис.3.10. Тренажер «Зіставлення»

Після виконання кожного з вище перерахованих тренажерів, з’являється фінальне вікно, що відображає кінцевий результат проходження тренування. В разі успішного виконання, вікно набуває вигляду показаного на рис.3.11, в якому відображається набрана кількість балів по ходу проходження тренування.

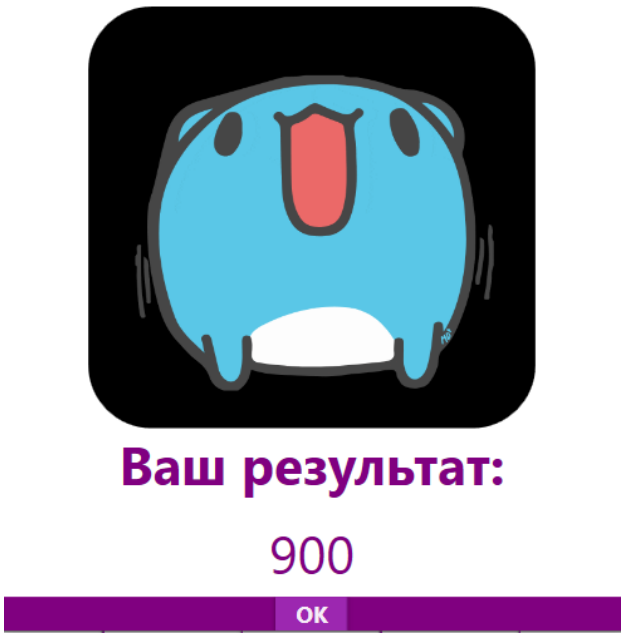

Рис.3.11. Вікно успішного проходження тренажеру

Якщо, в ході виконання тренування, не вдалося досягти мінімальної границі проходження тренажеру вікно набуде вигляду рис.3.12, а виконання тренажеру не буде зараховано.

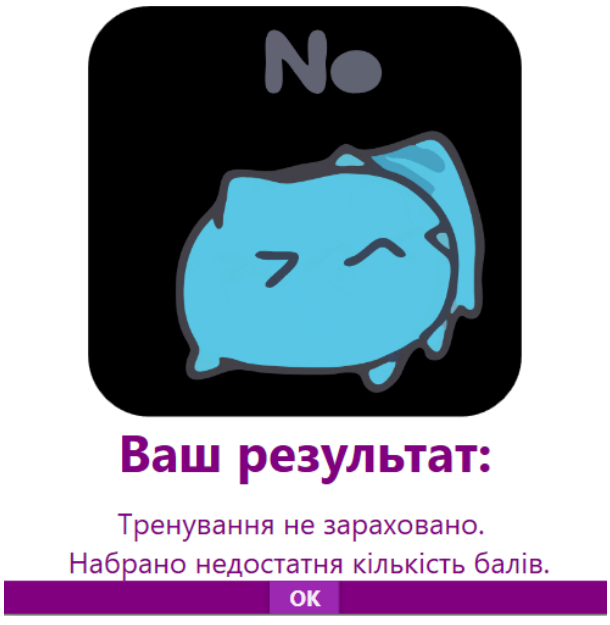

Рис.3.12. Вікно невдалого завершення тренажеру

В правій частині головного вікна розташовані щоденні завдання. Кожне, з цих завдань являє собою один з тренувальних днів описаних в книзі Гарета Мура «Тренажер мозку: Як розвинути гнучкість мислення за 40 днів», що складається з

двох частин теоретичної, в якій описується процеси формування мислення та тренувальне завдання наприкінці. Для відкриття нового тренувального дня, потрібно виконати тренувальне завдання та певну кількість вдалих спроб на тренажерах в головному вікні.

Розглянемо приклад виконання тренувального дня. Обираємо невиконаний тренувальний день в головному меню, він виділений оранжевим кольором, натискаємо по ньому лівою кнопкою миші.

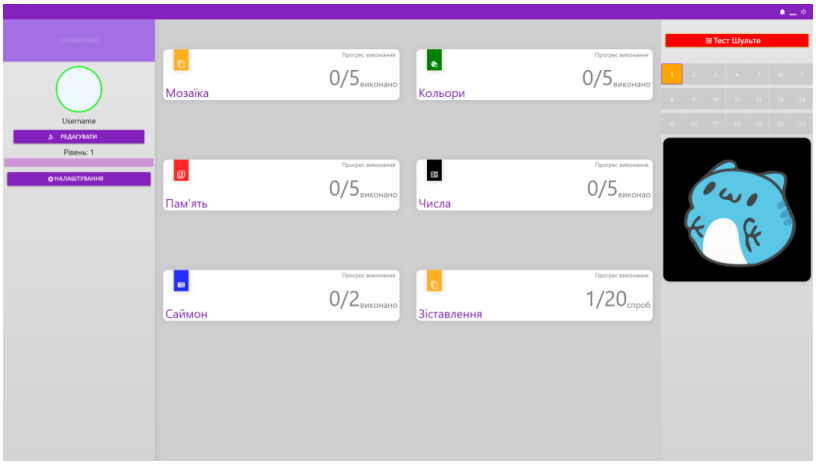

Рис.3.13. Головне меню додатку

Перед нами з’являється вікно, в якому описується деяка теоретична інформація, що сприяє саморозвитку. В правій та лівій частині екрану знаходяться органи управління, у вигляді стрілки, що дозволяються перегорнути сторінку на наступну, або повернутися до попередньої.

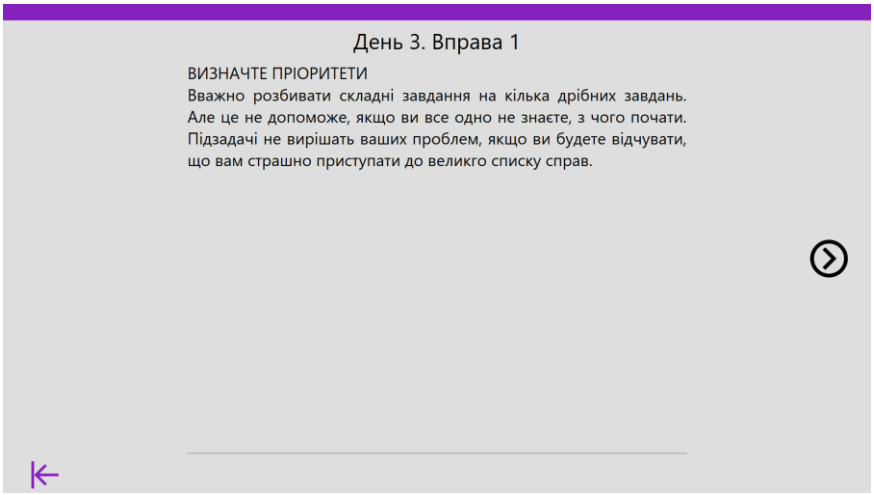

Рис. 3.14. Приклад першої сторінки щоденного тренування

Наприкінці кожної статті знаходяться правила, щодо виконання кожної вправи, знизу в центрі знаходиться копка «Почати», яка запускає тренування.

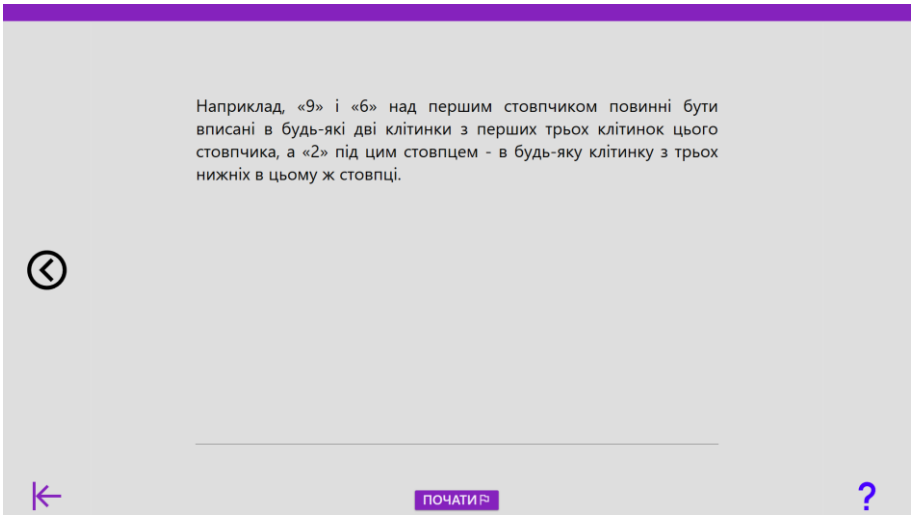

Рис.3.15. Правила щоденного тренування

Якщо, вправа виявилась занадто складною, в нижньому правому кутку знаходиться спеціальний символ у вигляді «?», даний символ являється підказкою, після активації якої обрана вправа виконається автоматично.

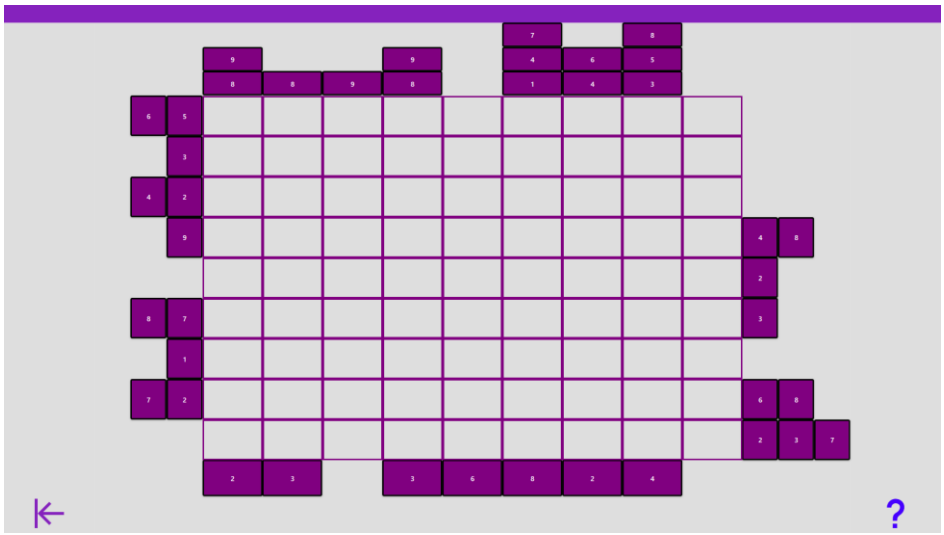

Рис.3.16. Приклад однієї з вправ

Після успішного виконання вправи, з’явиться відповідне повідомлення, з кнопкою «ОК», після натискання якої тренувальний день буде зараховано й відбудеться повернення до головного меню програми.

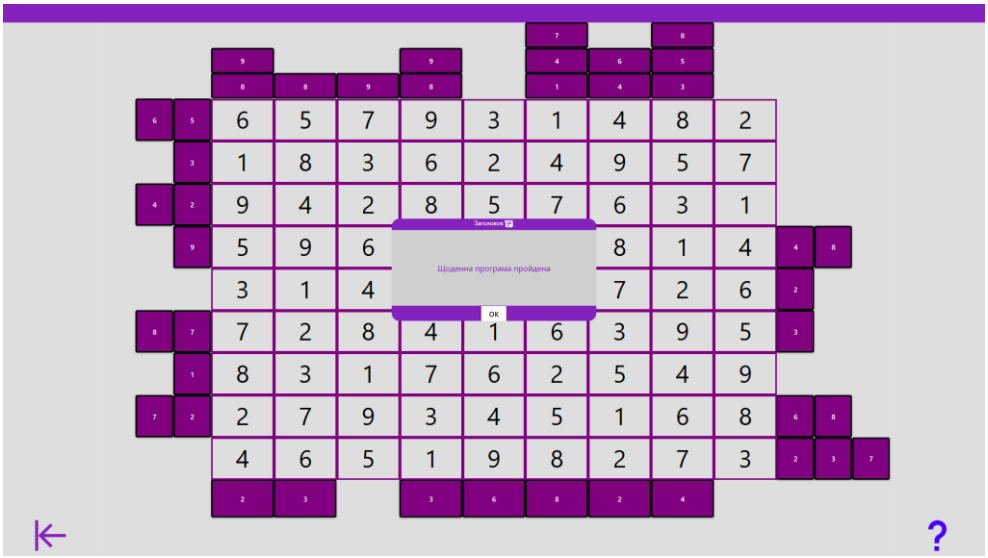

Рис.3.17. Приклад успішного виконання тренувального дня

3.5. Висновки до розділу

В третьому розділі розглянуто та описано основні технології й компоненти, що використовувалися для створення додатку. За основу створення додатку було обрано мову програмування *C#* з використанням платформи *.NET Framework* та технологією *WPF* в середовищі розробки *Microsoft Visual Studio 2019*. Організація роботи з базою даних виконується за допомогою *SQL Server Management Studio* та засобами *LINQ*.

Надано перелік основних тренажерів для розвитку мислення, які використовуються в програмі, з детальним описом кожного з них. Описана загальна структура щоденних тренувань.

Також визначено мінімальні вимоги до програмного та технічного забезпечення. Створено керівництво користувача.

ВИСНОВОК

У ході виконання дипломного проєкту було створено інтерактивний підручник для розвитку мислення. При розробці було розглянута концепція створення електронного підручника, його структура, види та способи подання інформації. Визначено основні переваги використання мультимедійних засобів над друкованим поданням інформації. Серед яких можна виділити краще та пришвидшене засвоєння інформації, з можливістю внесення змін або доповнень до вже існуючої інформації в будь-який момент часу. Проведено аналіз схожих програмних продуктів з визначенням основних переваг та недоліків. Серед суттєвих недоліків найяскравіше виділяється необхідність постійного підключення до мережі інтернет та умовна безкоштовна основа в якій застосовується система преміальних облікових записів, що мають значну перевагу над звичайним.

Основою мовою розробки програми було обрано об’єктно-орієнтовану мову програмування C#, із застосуванням програмної платформи *.NET Framework*. Реалізовано графічного користувацький інтерфейс використовуючи мову розмітки *XAML* в технології *WPF*. В якості середовища розробки використано *Microsoft Visual Studio 2019*. Створено модель бази даних використовуючи середовище *SQL Server 2019* та мову інтегрованих запитів *LINQ*.

Кінцевий продукт являє собою інтерактивний безкоштовний додаток для розвитку мислення, що здатен працювати або при підключення до мережі інтернет, зі збереження прогресу користувача на сервері, або ж без інтернет з’єднання, з локальним збереженням прогресу на комп’ютері користувача.

Основними функціями додатку є:

- Збереження індивідуального прогресу кожного з користувачів.
- Наявність різноманітних тренажерів для розвитку мислення.
- Електронна інтерпретація книги Гарета Мура «Тренажер мозку: Як розвинути гнучкість мислення за 40 днів», з програмним відтворенням вправ.
- Програмна реалізації тестування за таблицею Шульте для оцінки ефективності тренувань та досягнень користувача.

Після створення додатку у звіті перелічено та описано всі технології та засоби, що використовувались для реалізації інтерактивного підручника, надано керівництво користувача.

СПИСОК БІБЛІОГРАФІЧНИХ ПОСИЛАНЬ ВИКОРИСТАНИХ ДЖЕРЕЛ

1. *metod-kopilka.ru* [Електронний ресурс]. – Режим доступу: *http://www.metod-kopilka.ru/page-article-30.html*
2. *metod-kopilka.ru* [Електронний ресурс]. – Режим доступу: *http://www.metod-kopilka.ru/page-article-30.html*
3. method.saitar.com [Електронний ресурс]. – Режим доступу: http://method.saitar.com/index/0-14.html
4. ДСТУ 3008-95 Документація. Звіти у сфері науки і техніки. Структура і правила оформлення : видання офіційне, Київ : Держстандарт України, 1995. – 38 с.
5. Мур Г. Тренажер мозку: Як розвинути гнучкість мислення за 40 днів. /Г. Мур; Пер. с англ. – Москва: «Альпина Паблишер», 2020. – 171 с.
6. Шилдт Г. *C# 4.0*: полное руководство. / Г. Шилдт; Пер. с англ. – Москва: ООО «Вильямс», 2011. – 1074 с.
7. Рихтер Д. *CLR* via *C#*. Программирование на платформе Microsoft *.NET Framework 4.5* на языке *C#*. 4-е изд. / Д. Рихтер – СПб.: Питер, 2013. – 896 с.
8. Шевчук А. *Design Patterns* via *C#*. Приемы объектно-ориентированного проектирования. / Шевчук А., Охрименко Д., Касьянов А. – Київ: Видавн. «*ITVDN*», 2015. – 288 с.
9. Скит Д. *C#* для профессионалов: тонкости программирования, 3-е изд./Д. Скит Пер. с англ. – Москва : ООО «Вильямс», 2014. – 608 с.
10. Тепляков С. Паттерны проектирования на платформе .NET. / С. Тепляков – СПб.: Питер, 2015. – 320 с.
11. Троелсен Е. Язык программирования *C#7* и платформы *.NET* и *.NET Core*, 8-е изд. / Троелсен Е., Джепикс Ф. Пер. с англ. – СПб. : ООО «Диалектика», 2018 – 1328 с.
12. *github.com* [Електронний ресурс]. – Режим доступу: *https://github.com/MaterialDesignInXAML/MaterialDesignInXamlToolkit*

13. *metanit.com* [Електронний ресурс]. – Режим доступу: *https://metanit.com/sharp/uwp/13.1.php*
14. *docs.microsoft.com* [Електронний ресурс]. – Режим доступу: *https://docs.microsoft.com/en-us/dotnet/desktop/wpf/getting-started/walkthrough-my-first-wpf-desktop-application?view=netframeworkdesktop-4.8*
15. *github.com* [Електронний ресурс]. – Режим доступу: *https://github.com/Abel13/Dashboard1*