ЗІДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

Навчально–науковий інститут Інформаційних технологій

Кафедра Інженерії програмного забезпечення

Пояснювальна записка

до бакалаврської роботи
на ступень вищої освіти бакалавр

на тему «**РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ІНТЕРАКТИВНОЇ СИСТЕМИ ДЛЯ ВИВЧЕННЯ ІСТОРІЇ ЗА ДОПОМОГОЮ МОВИ ПРОГРАМУВАННЯ JAVA»**

Виконав: студент 4 курсу, групи ПД-42
спеціальності
121 Інженерії програмного забезпечення
(шифр і назва спеціальності)
Щербина Андрій Сергійович
(прізвище та ініціали)
Керівник Золотухіна О.А.
(прізвище та ініціали)
Рецензент
(прізвище та ініціали)

КИЇВ – 2021

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

Навчально–науковий інститут Інформаційних технологій

Кафедра Інженерії прорамного забезпечення
Ступінь вищої освіти «Бакалавр»
Спеціальність підготовки Програмна інженерія

ЗАТВЕРДЖУЮ
Завідувач кафедри
Інженерії прорамного
забезпечення
_____О.В.Негоденко
“____” _____ 2021 року

З А В Д А Н Н Я
НА БАКАЛАВРСЬКУЮ РОБОТУ СТУДЕНТУ

Щербині Андрію Сергійовичу
(прізвище, ім’я, по батькові)

1.Тема роботи: «Розробка програмного забезпечення інтерактивної системи для вивчення історії за допомогою мови програмування Java»

Керівник роботи Золотухіна Оксана Анатоліївна, к.т.н.,
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)
затверджені наказом вищого навчального закладу від «12» березня 2021 року № 65

2. Строк подання студентом роботи «1» червня 2021року

3. Вихідні дані до роботи: чинна програма ЗНО по історії України, тестові завдання ЗНО по історії України 2020 року, методи та засоби проектування та розробки програмного забезпечення, принципи побудови інтерактивних систем

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити).
4.1 Огляд предметної області
4.2 Вибір засобів реалізації
4.3Проектування системи
4.4 Розробка програмного забезпечення системи
4.5 Тестування системи

5. Перелік графічного матеріалу (презентація)
1.Діаграма використання системи 2.Діаграма діяльності системи 3.Оцінювання досягнень користувача 4.Блок схеми 5. ER дыаграма БД 6.Структурна схема БД 7. Діаграма класів 8.Приклади роботи програми

6. Дата видачі завдання__19 квітня 2021 року__

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів бакалаврської роботи	Строк виконання етапів роботи	Примітка
1	Отримання завдання на бакалаврську роботу	19.04.21	
2	Огляд предметної області та аналіз об’єкту автоматизації	24.04.21	
3	Вибір інструментальних засобів реалізації		
4	Проектування системи	26.04.21	
5	Розробка алгоритмічного забезпечення системи	29.04.21	
6	Розробка програмного забезпечення системи	13.05.21	
7	Тестування програмної системи	16.05.21	
8	Написання та оформлення пояснювальної записки	20.05.21	
9	Розробка графічних та презентаційних матеріалів	27.05.21	
10	Захист бакалаврської роботи	01.06.21	

Студент _____ А.С. Щербина
(підпис) (прізвище та ініціали)

Керівник роботи _____ О.А. Золотухіна
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Текстова частина бакалаврської роботи: 122с., 60 рис., 2 дод., 18 джерела.

ІНТЕРАКТИВНА СИСТЕМА, ІСТОРІЯ УКРАЇНИ, ЗАВДАННЯ, ТЕСТ, ОЦІНЮВАННЯ ЗНАНЬ, АЛГОРИТМ, БАЗА ДАНИХ, ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ, JAVA

Об`єкт дослідження – процес вивчення історії умовах застосування сучасних інформаційних технологій.

Предмет дослідження – створення інтерактивної навчальної системи для підготовки до зовнішнього незалежного оцінювання з історії України.

Мета роботи – спрощення процесу вивчення історії України при підготовці до зовнішнього незалежного оцінювання за рахунок використання інтерактивної системи навчання.

Методи дослідження – методи теорії інформації, методи педагогічного тестування, методи проектування та розробки користувацьких інтерфейсів, методи проектування та розробки інтерактивних програмних додатків.

У роботі проведено аналіз сучасного стану питання підготовки до ЗНО по історії України, ринку програмних продуктів, що використовуються для цього, особливостей розробки інтерактивних додатків

Здійснено проектування програмної системи та бази даних, розроблено структуру завдань та алгоритмічне забезпечення для їх формування та обробки результатів виконання.

Розроблено алгоритмічне забезпечення, що до отримання, обробки, та зберігання інформації в процесі побудови завдань та виконання завдань при навчанні, завантаження та збереження даних, розрахунку та візуалізації оцінок результатів навчання. Реалізовано інтерактивну систему мовою програмування Java, яка дозволяє збільшити об’єм та підвищити рівень знань з історії України при підготовці до ЗНО О.

ЗМІСТ

Стор.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

CSS	–	Cascading Style Sheets, каскадні таблиці стилів
HTTP	–	HyperText Transfer Protocol, протокол передачі текста
JRE	–	Java Runtime Environment, середовище виконання Java
JVM	–	Java Virtual Machine, віртуальна машина Java
RIA	–	Rich Internet Applications, насичені Інтернет-додатки
SQL	–	Structured Query Language, мова програмування структурованих запитів
XML	–	eXtensible Markup Language, розширювана мова розмітки
БД	–	База даних
ЗНО	–	Зовнішнє незалежне оцінювання

ВСТУП

Вивчення історії має важливе практичне значення, оскільки найважливіший сенс історії полягає в тому, що вона є ключем до розуміння того, що відбувається в сьогоденні та ключем до розуміння майбутнього. Вилучення цього ключа, як у окремої людини, так і у народу в цілому, позбавляє його можливості рухатися далі в правильному для нього напрямку. Історія виховує людину мислячою та вільну у своїх судженнях, що здатна відрізнити правду від пропаганди та нав'язуваної ідеології, формує ментальність суспільства, а значить, потрібна для розуміння своєї власної особистості. Знання та розуміння історії надає можливість орієнтуватися в поточних політичних ситуаціях і стосунках між державами, а значить розуміти місце своєї країни на світовій арені.

Інтерактивне навчання, яке значною мірою є самостійним, з використанням сучасних інформаційних технологій у теперішній час є одним з найважливіших та більш затребуваних напрямків вдосконалення системи освіти. Швидкий розвиток телекомунікацій, інформаційних технологій та мережі Інтернет створило технологічну основу для обміну інформацією між різного роду користувачами, незалежно від їх соціального статусу, національної приналежності, географічного положення і стало потужним стимулом розвитку дистанційної освіти.

Встановлення карантинних обмежень, переведення навчальних закладів на дистанційну форму навчання, ймовірність необхідності дотримання режиму самоізоляції у зв'язку з поточною пандемією коронавірусной інфекції (COVID-19) збільшили попит користувачів на інтерактивні засобі навчання, особливо учнів старших класів з метою підготовки до зовнішнього незалежного тестування, освітні заклади на розробку та впровадження інструментарію для дистанційного навчання.

Таким чином, завдання розробки інтерактивної системи вивчення історії є сучасним та актуальним.

Об`єкт дослідження – процес вивчення історії умовах застосування сучасних інформаційних технологій.

Предмет дослідження – створення інтерактивної навчальної системи для підготовки до зовнішнього незалежного оцінювання з історії України.

Мета роботи – спрощення процесу вивчення історії України при підготовці до зовнішнього незалежного оцінювання за рахунок використання інтерактивної системи навчання.

Методи дослідження – методи теорії інформації, методи педагогічного тестування, методи проектування та розробки користувацьких інтерфейсів, методи проектування та розробки інтерактивних програмних додатків.

Практична значущість результатів полягає в використанні розробленої системи в процесі підготовки то проходження зовнішнього незалежного оцінювання з історії України.

1 АНАЛІЗ СТАНУ ПИТАННЯ ТА ПОСТАНОВКА ЗАДАЧІ РОЗРОБКИ

1.1. Аналіз предметної області

1.1.1. Зовнішнє незалежне оцінювання

Зовнішнє незалежне оцінювання (ЗНО, раніше також Зовнішнє тестування, ЗТ) – комплекс організаційних процедур спрямований на визначення рівня навчальних досягнень випускників середніх навчальних закладів при їхньому вступі до закладів вищої освіти. Метою ЗНО є підвищення рівня освіти населення України та забезпечення реалізації конституційних прав громадян на рівний доступ до якісної освіти, здійснення контролю за дотриманням Державного стандарту базової та повної середньої освіти й аналізу стану системи освіти, прогнозування її розвитку. Результати зовнішнього незалежного оцінювання зараховуються як результати державної підсумкової атестації та як результати вступних іспитів до закладів вищої освіти.

В Україні система ЗНР формується з 2004р. за підтримки міжнародних і громадських організацій, здійснюється Українським центром оцінювання якості освіти у співпраці з місцевими органами управління освітою, обласними інститутами післядипломної педагогічної освіти, навчальними закладами, координація роботи учасників процесу ЗНО здійснюється через регіональні центри оцінювання якості освіти.

Впровадження ЗНО принесло гарні результати щодо значного підвищення якості середньої та вищої освіти в Україні, позитивно позначилося на загальному рівні освіченості та стало вагомим фактором у боротьбі з корупцією. Таким чином, перевагами впровадження ЗНО можна зазначити наступні:

− створення умов рівного доступу до освіти для всіх, незалежно від матеріальних можливостей;

− об'єктивність (однак, при перевірці власних висловлювань у ЗНО з української мови та літератури трапляється людський фактор);

− долання корупції на локальному рівні;

– наближення до європейських стандартів [1].

Визначення рівня навчальних досягнень випускників середніх навчальних закладів виконується шляхом тестування.

Знання з історії є невід'ємною частиною сучасної розвинутої людини. Освіченість в історії та історичних подіях має важливе практичне значення, бо найважливіший сенс історії полягає в тому, що вона є ключем до розуміння того, що відбувається в сьогоденні. Крім того, історія є ключем до розуміння майбутнього. Вилучення цього ключа, як у окремої людини, так і у народу в цілому, позбавляє його можливості рухатися далі та розвиватися. Саме тому, історія України є одним з предметів, за якими виконується ЗНО випускників шкіл.

ЗНО з історії України 2021 року вміщує в собі 31 тему зі шкільної програми за період 7-11 класу, складається з 60 тестових завдань, що розділені на дві частини: історія України 20го – початку 21 століття та історія України від найдавніших часів до кінця 19 століття. Кожне із завдань оцінюється за відповідними критеріями, на виконання завдань учням відведено 150 хвилин.

В тестовому зошиті представленні завдання таких форм:

– з вибором однієї правильної відповіді, завдання складається з основи і чотирьох варіантів відповіді, з яких лише один правильна;

– на встановлення відповідності («логічні пари»), завдання складається з основи і двох колонок інформації, позначених цифрами (ліворуч) і буквами (праворуч), а виконання завдання передбачає встановлення відповідності (утворення «логічних пар») між інформацією, позначеної цифрами та буквами;

– на встановлення правильної послідовності, завдання складається з основи і переліку подій (явищ, фактів, процесів і т.д.), позначених буквами, які потрібно розташувати у правильній послідовності;

– з вибором 3 правильних відповідей із 7 запропонованих варіантів, завдання складається з основи і семи варіантів відповіді, позначених цифрами, серед яких тільки три правильні. [2].

Програма ЗНО з історії України хронологічно охоплює весь зміст шкільного курсу від найдавніших часів до сучасності, та передбачає перевірку знань про

основні політичні, соціально-економічні, культурні події, явища і процеси минулого, діяльність видатних історичних діячів, а також сформованості в учнів загальнопредметних історичних умінь. Невід'ємною складовою програми ЗНО з історії України є перелік пам'яток архітектури та образотворчого мистецтва, обов'язкових для розпізнавання учасниками тестування [2].

На теперішній час найпопулярнішими форматами підготовки до тестування з історії України є наступні:

- самостійна підготовка;
- заняття з репетитором;
- проходження онлайн-курсів.

Експерти зазначають, крім вищезгаданих форматів, на ринку уже з'являються альтернативні варіанти, наприклад, офлайн-курси у групах до 10 людей, коли всі працюють командою, або курси для підготовки до ЗНО, які поєднують у собі кілька методів підготовки: теоретичні матеріали можна переглядати онлайн, а закріплювати матеріал – під час аудиторних занять.

1.1.2. Види завдань для перевірки знань та вмінь

Збільшення ефективності використання завдань при контролі знань та забезпечення належного рівня отриманих результатів при навчанні безпосередньо пов’язано з вибором видів тестових завдань, що використовуються. Кожен вид завдання відповідає своїй меті: деякі краще підходять для з'ясування обов'язкових знань, інші спроможні визнати не тільки знання, а й навички, або творчі здібності. Виділяють наступні види завдань присвячені та форми завдань:

- завдання з вибором єдиної правильної відповіді;
- завдання з вибором декількох правильних відповідей;
- завдання, що мають дві відповіді, одна є вірною, а інша ні;
- завдання на завершення речень, фраз, або висловлювань;
- питання на встановлення взаємозв'язків;
- завдання впізнання;

– завдання пошуку відмінності;

– завдання співвідношення;

– завдання підстановки [3].

Також існує наступна класифікація завдань.

Завдання закритої форми, в яких необхідно обрати правильну відповідь з наданих. Таке завдання має готові відповіді, з яких повинен вибрати вірну. У такій формі завдань можна додатково виділити декілька: тестові завдання з вибором однієї правильної відповіді та завдання з вибором кількох правильних відповідей. При чому кількість відповідей, що надаються може варіюватися від двох до п’яти-шести. Завдання такої форми є найбільше поширеними в тестовій практиці, саме такі види завдань називають тестами. Причинами широкого використання цієї форми тестових завдань є простота виконання та зручність для швидкого та автоматизованого контролю. При складанні таких завдань найбільш важливим є підбір варіантів неправильних, але «правдоподібних» відповідей, які називається дистракторами (від анг. to distract- відволікати). Якщо в завданні є k відповідей, то відповідь, частка вибору якої близька до значення 1/k вважається найкращим дистрактором, а менше, ніж 1/k називають домінуючим дистрактором. Наприклад, при використанні чотирьох варіантів відповідей ймовірність вгадування правильної відповіді становить 25%, а при п'яти відповідях вже 20%. Завданням з вибором кількох правильних допускається вибір кількох правильних відповідей з числа наданих. Ефективність використання цього виду завдання підвищується якщо використовувати серію завдань, формулювати завдання з однозначним тлумаченням та обмежувати час виконання [4].

Завданнями відкритої форми, або завдання на додаток. Це завдання, де не вказуються можливі варіанти відповіді, що вимагає самостійно формулювати відповідь, а не вибрати з готових. Перевагою завдань закритої форми є те, що вони не допускають можливості вгадування.

Завдання на встановлення відповідності. Під таким завданням необхідність встановити відповідність елементів однієї множини елементами іншої, при цьому бажано, щоб кількість елементів в цих множинах було неоднаковим. Такі

завдання обмежують сферу застосування перевіркою знання певних фактів. Найбільш цінним у використанні такого типу завдань є можливість визначити серед інших асоціативну складову знань, тобто знання взаємозв'язку визначень, наприклад авторів і творів, зв'язок між законами та формулами і т.д.

Завдання на встановлення правильної послідовності, в яких необхідно вказати певний порядок. Разом із завданням пропонується елементи та необхідно зробити вибір правильного порядку їх проходження. Ці завдання розробляються для перевірки умінь здійснювати послідовність процесів, дій або операцій, виявлення сформованості конкретних понять [5].

1.1.3. Тестування як контроль знань

Під тестуванням (від слова test – випробування, перевірка) у загальному сенсі розуміють визначення відповідності предмета, що тестується, заданим характеристикам. Методи тестування широко застосовуються у багатьох розділах наук, а також у освіті для визначення придатності об'єкта тестування (навчаємого) для виконання тих чи інших функцій, які передбачені метою освіти. При цьому у завдання тестування немає мети визначити причин, з яких об’єкт не відповідає заданим вимогам .

Належна якість та достовірність результатів тестування у великій мірі залежить від тестера. Таким чином, тест визначають як випробування, що є стандартизованими, короткими та обмеженими у часі, які визначені для встановлення кількісних і якісних індивідуальних відмінностей об’єкту тестування.

Традиційний тест уявляє з себе стандартизований метод діагностики та рівня і підготовленості того, хто навчається. Такий тест містить перелік питань і декілька різних варіантів відповідей на нього, а кожне питання оцінюється в певну кількість балів. Результат повного тесту залежить від кількості питань, на які було надано вірну відповідь. У такому тесті всі об’єкти тестування виконують на одні й ті ж самі завдання, в однаковий час, в однакових умовах і з однаковими правилами оцінювання результатів. Головна мета застосування традиційних

тестів - встановити рівень знань і на цій основі визначити місце (або рейтинг) кожного тестуємого. Для досягнення цієї мети можна створити незліченну кількість тестів, і всі вони можуть відповідати досягненню поставленої задачі.

Специфічний тест відрізняється від традиційного тим, що завдання тесту являють собою не питання з відповідями, а сформульовані у формі висловлювань, істинних або помилкових, в залежності від відповідей. Труднощі завдання може визначатися двояко: умоглядно, на основі передбачуваного числа і характеру розумових операцій, необхідних для успішного виконання завдань, і після досвідченого випробування завдань, з підрахунком частки неправильних відповідей [6].

У загальний тест намагаються відібрати мінімально достатню кількість завдань, які дозволять відносно точно визначити структуру та рівень підготовленості. Результати тестування інтерпретуються переважно з використанням середньої арифметичної та процентних норм, які показують скільки відсотків тестуємих мають тестовий результат гірший, ніж у будь-якого іншого тестуємого. Така інтерпретація тестових результатів називають нормативно-орієнтованою.

У порівнянні з іншими формами контролю знань тестування має свої переваги та недоліки.

До переваг тестування можна віднести наступні:

- більш якісний та об'єктивний спосіб оцінювання;
- більш справедливий метод оцінювання, ставить всіх тестуємих у рівні умови;
- більш об'ємний інструмент у порівнянні з іншими, бо тестування може включати в себе завдання з усіх тем курсу;
- більш точний інструмент, ніж мовне опитування;
- більш ефективний з економічної точки зору та витрат ресурсів на отримання результатів;
- більш м'який інструмент, бо ставить тестуємих в рівні умови, використовуючи єдину процедуру і єдині критерії оцінки.

До недоліків такої форми контролю знань як тестування можна віднести:

− процес розробки якісного тестового інструментарію − тривалий, трудомісткий і дорогий;

− отриманні результаті тестування дані містять інформацію про прогалини в знаннях, але не дозволяють судити про причини їх виникнення;

− високі та продуктивні рівні знань, пов'язані з творчістю неможливо перевірити за допомогою тестування;

− повторне застосування тесту не є ефективним, існує необхідність внесення змін до завдання;

− присутність елементу випадковості, що може бути пов’язано як з помилкою, так і з вгадуванням вірної відповіді.

1.1.4. Поняття інтерактивної навчальної системи

Інтерактивна система – це система компонентів апаратного і програмного забезпечення, яка отримує інформацію, що вводиться користувачем, і передає йому свою відповідь, допомагаючи в роботі або виконанні завдання.

Інтерактивність (в контексті інформаційної системи) – це можливість інформаційно-комунікаційної системи по-різному реагувати на будь-які дії користувача в активному режимі. Інформаційні технології є неодмінною умовою для функціонування високоефективної моделі навчання, основною метою якої є активне залучення кожного з учнів в освітній і дослідницький процеси.

Застосування новітніх технологій в навчанні підвищує наочність, полегшує сприйняття матеріалу. Це сприятливо впливає на мотивацію учнів і загальну ефективність освітнього процесу.

В даний час все більшу кількість навчальних закладів оснащує свої аудиторії і класи інтерактивними засобами навчання. Їх використання під час заняття дає можливість побачити реалістичні 2-D і 3-D моделі об'єктів вивчення, спостерігати за їх змінами та керувати ними.

Переваги інтерактивних методів навчання:

− навчання стає більш індивідуальним, враховує особливості особистості, інтереси і потреби кожного, хто навчається ;

− з'являється можливість ємко і стисло уявити будь-який обсяг навчальної інформації;

− значно спрощується процес засвоєння навчального матеріалу, завдяки поліпшенню візуального сприйняття;

− активізується пізнавальна діяльність учнів, вони отримують теоретичні знання та практичні навички.

Індустрія комп'ютерних засобів навчання налічує вже понад двадцять п'ять років. На перших порах в навчальному процесі використовувалися різні програмно-методичні комплекси для освоєння студентами елементів інформаційних технологій. Потім стали створюватися комп'ютерні навчальні системи на базі електронних підручників з різних дисциплін з текстовими та графічними фрагментами [7].

У теперішній час найчастіше використовуються інші форми інтерактивного навчання, що наведено нижче.

E-learning (електронне навчання). Навчання здійснюється за допомогою мультимедіа та Інтернету. Засоби навчання: електронна пошта, Інтернет-спільноти. Інтернет-форуми, відеоконференції, відео-лекції, онлайн-тестування, case-study, експертні системи, онлайн-консультування і т.д. E-learning – це процес отримання знань і певних навичок з використанням освітнього середовища, в основі якої лежить використання інформаційних технологій, що надають навчальну інформацію і забезпечують її обмін на відстані, що здійснюють необхідний супровід і адміністрування навчання.

B- learning. Уявляє собою змішане навчання, взаємозв'язок віртуального і очного спілкування з викладачем. В освітньому процесі присутній електронна компонента (комп'ютер) освіти, інформаційні технології та Інтернет та класична безпосередня взаємодія викладача і студента.

Виділяють наступні форми інтерактивності:

− дієва інтерактивність, при якій навчаються керують програмою (застосування гіпертекстової розмітки, структура електронних енциклопедій, довідників, баз даних), ефективна при дистанційному навчанні;

− реактивна інтерактивність полягає в навчанні того, що надає програма: поверхове ознайомлення або демонстрація досліджуваного матеріалу;

− взаємна інтерактивність - взаємне пристосування учня і програми один до одного (тренажери, ігри-пригоди, навчальні програми, практикуми і т.д.).

Інтерактивність розкриває ступінь і характер взаємодії між об'єктами, це принцип організації освітньої системи, що дозволяє досягти поставленої мети взаємодією елементів системи за допомогою інформаційного обміну.

Використання інтерактивних засобів навчання – це використання спеціальних навчальних програм, які допомагають кожному в доступному для нього темпі проходити навчальний матеріал, освоювати його і здавати контрольне тестування [7.]

До інтерактивних засобів у програмних системах можна віднести такі, як:

− управління об'єктами на екрані;

− лінійна навігація за допомогою полос прокрутки;

− діалогова функція довідки;

− використання гіперпосилань та ієрархічна навігація;

− зворотній зв'язок, що дозволяє оцінити якість дій користувача;

− конструктивна взаємодія, забезпечення можливостей для побудови цілей безпосередньо на екрані;

− взаємодія у вигляді рефлексії.

1.1.5. Гейміфікація

У системах для навчання доволі широко використовують елементи гейміфікація. Гейміфікація (або ігрофікація) – це використання ігрових підходів для неігрових процесів. Це методика, що дозволяє отримати нові знання, за рахунок використання елементів комп'ютерних ігор і ігрове мислення. Гейміфікація підвищує інтерес тих, хто навчається, допомагаючи швидше

засвоювати нову інформацію при організації різних навчальних програм. Важливим є доцільне впровадження геймофікації в інтерактивні додатки з метою отримання від елементів гри максимальної користі.

В останні роки гейміфікація стала користуватися особливою популярністю при розробці навчальних програм. Подібна зацікавленість у використанні елементів комп'ютерних ігор обумовлена наступними основними факторами:

- вплив ринку ігрових додатків;
- зростання інтересу до ігор серед дорослої аудиторії;
- результативність навчання при використанні ігрових програм.

Обсяг ринку комп'ютерних ігор у 2019 році склав біля 106,5 млрд. доларів, у 2020 біля 124,6 млрд. доларів, а до кінця поточного року, за прогнозами, складе майже 130 млрд. доларів. Однак, вплив ринку є не єдиним фактором підвищеного інтересу до гейміфікація навчання.

Згідно висновків фахівців, існує зв'язок між іграми і процесом запам'ятовування інформації. Дошкільнята і школярі показують доволі високий рівень результатів засвоєння інформації при використанні ігрових методик. Результативність подібних підходів була підтверджена дослідженнями. Крім того, вчені з усього світу сходяться думкою, що впровадження ігрових методик активізує ділянки мозку, відповідальні за навчання, а сама гра (а саме цикл завдання-досягнення-нагорода) сприяє виробленню допаміну в мозку, що підсилює бажання грати. Завдяки використанню ігрових елементів у системах навчання вдається домогтися високих результатів засвоєння буквально за кілька тижнів. Найбільш позитивним ефектом використання елементів гри є поліпшення пам'яті.

Одним з найважливіших аспектів впровадження гейміфікації є опрацювання структури та забезпечення динаміки ігрового навчального процесу, де ігровим процесом (або геймплеєм) називають особливості взаємодії людини із грою, найчастіше, які створюються за допомогою правил, завдань та способів їх вирішення, які пропонує гра [8].

Основою будь-якого ігрового процесу є повторюваний ланцюжок дія-реакція-зворотний зв'язок:

− дія – інформація, яку гравець надає грі з ігрових контролерів (маніпулятор миша, клавіатура, геймпад тощо);

− реакція – відповідь гри на дії гравця, сформована на основі наданої ним інформації, але ще не виражена для нього в доступній формі;

− відгук – інформація, отримувана з гри гравцем., що поділяється на два основних види: явний (очевидна відповідь на дії) і неявний (служить фоном чи, якимось чином, інформує гравця).

Основні методи гейміфікації прагнуть максимально залучити природні людські інстинкти, такі як конкуренція, досягнення, статус, самовираження, вирішення завдання [8]. Головний сенс використання гейміфікації в системах навчання є перетворення «нудних» процесів в «цікаві» - таким чином у людей з'являється мотивація завершити розпочате – для цього використовують такі компоненти:

− динаміка – сценарії, що вимагають уваги користувача та реакцію в реальному часі;

− механіка – бали, рівні, шкала прогресу, рейтинги, постійний зворотний зв'язок, тощо;

− винагорода – мотивує саме факт перемоги над іншими гравцями, а винагорода є підтвердженням статусу переможця;

− вимірювання – принципи оцінювання результатів;

− естетика – сприяє емоційній замученості, підвищує лояльність користувачів і допомагає їм отримувати задоволення від процесу.

Для створення ігрового навчального процесу більш привабливим до нього можуть бути доданими наступні елементи:

− таблиця лідерів – повинна відображати досягнення в напрямку, важливому для навчальної програми;

− схеми-образи – відомі та асоціативні елементи, послідовності, що додаються до ігрового контексту.

− колекціонування – накопичення балів у процесі навчання, отримання нагороди і відзнак, або інших об'єктів, які відносяться до контенту;

− сюрпризи, спонтанні радості – отримання винагород несподівано;

− подарунки – можливість давати бали іншим учасникам, з метою стратегічного мислення у командній ігри;

− статус – кожен учасник може досягнути важливу ціль, що стосуються контенту [9].

Важливою складовою успішного навчання в ході ігрового процесу є підвищення рівню залученості учасників. Щоб досягти певних цілей перед користувачем треба ставити здійсненні завдання. При розробці завдань дотримуються наступних вимог:

− завдання ставлять таким чином, щоб мати можливість відстежувати дії користувача;

− за виконання поставлених завдань потрібно надавати нагороди, коли мета досягнута користувач переходить на новий рівень або стає переможцем в грі;

− завдання повинні бути різнотипними, повинні відрізнятися складністю, тривалістю та іншими параметрами;

− завдання повинні мати різний рівень складності, шляхом обмеження часу на їх виконання.

При розробці навчальних інтерактивних програм використовують наступні прийоми гейміфікіції:

− ризик: можливість припускання помилки та повернення до збереженого етапу, щоб спробувати вирішити завдання ще раз;

− подолання труднощів: вирішувати складні завдання є цікавим та мотивує продовжувати удосконалювати та закріплювати отриманні навички;

− право на помилку: користувач повинен мати кілька спроб для вирішення задачі, отримувати пояснення що до зробленої помилки, варіанти правильної дії або усунення наслідків помилки;

− введення системи балів: дозволяє відстежувати прогрес на шляху просування до мети;

− відкриття інформації: поступове отримання доступу до нової інформації;

− прогрес: надає інформацію користувачам на якому етапі навчання вони знаходяться, для цього використовують індикатори прогресу, лічильники рівнів та інші подібні прийоми.

Зворотний зв'язок має свої різновиди: візуальний (бачене гравцем на екрані), звуковий (чуте через пристрої відтворення звуку), дійовий (подія в самій грі – відповідь на дії гравця, зазвичай поєднання зображення зі звуком), неігрових персонажів (відповідь персонажів, що «населяють» гру, на дії гравця), акумулятивний (виражений певним чином прогрес гри), емоційний (емоції, які викликає гра), здійснення (вираження сенсу окремих завдань і всієї гри, їх завершення), інформативний (відомості, які гравець отримує в контексті грання). Отриманий зворотний зв'язок спонукає до нових дій та до завершення гри [9]. Використання тематичних пунктів дозволяє залучити учасників до динаміку курсу навчання. Це є одним з важливих критеріїв ефективності процесу навчання.

1.2. Аналіз аналогічних програмних продуктів

Сучасний ринок програмного забезпечення пропонує велику кількість програмних додатків для вивчення історії України з метою підготовки до ЗНО. Здебільшого подібні розробки виконано для використання на мобільних платформах Android та iOS. Кожний бажаючий може обрати для себе відповідний до його смаків та потреб додаток та використовувати його для засвоєння матеріалу з історії України або підготовки до ЗНО. На ринку присутня велика кількість безкоштовних програмних продуктів подібного типу, онлайнові та розроблені у вигляді окремих додатків, з функціями оцінювання та збереження результатів навчання, з елементами гри та багато інших. Окрім того додатки, що використовуються для підготовки к ЗНО, постійно оновлюються відповідно до змін, що вносяться у принципи тестування та обсяг теоретичного матеріалу з історії України, що виноситься до перевірки ЗНО.

Основними критеріями оцінювання програмних додатків вивчення історії України можна вказати наступні:

- зовнішнє оформлення;
- методичні вказівки;
- зміст вправ;
- засоби надання та закріплення матеріалу;
- можливість налаштування.

Розглянемо деякі з багатьох аналогічних програмних продуктів.

Додаток «Історія України ЗНО 2021». На Play Маркет має оцінку 4,4 з загальної кількості оцінок 770. На рисунку 1.1 наведено скріншоти різних етапів роботи додатку. До основних переваг додатку можна віднести наступні:

- відповідність чинній програмі ЗНО: дати, події, персоналії і пам’ятки для візуального розпізнавання взяті з офіціально затвердженого джерела;
- наявність теоретичного матеріалу впорядкованого за категоріями: терміни, факти та персоналії;
- можливість навчання у процесі розв’язання тестів: після неправильної відповіді буде показаний правильний варіант;
- визначена мета: при проходженні кожного з тестів набрати якомога більше балів за один раз;
- можливість робити помилки при досягненні мети
- наявність тестів на відповідність для історичних подій та дат, персоналій і пам’яток мистецтва;
- наявність тесту – «вірю/не вірю».

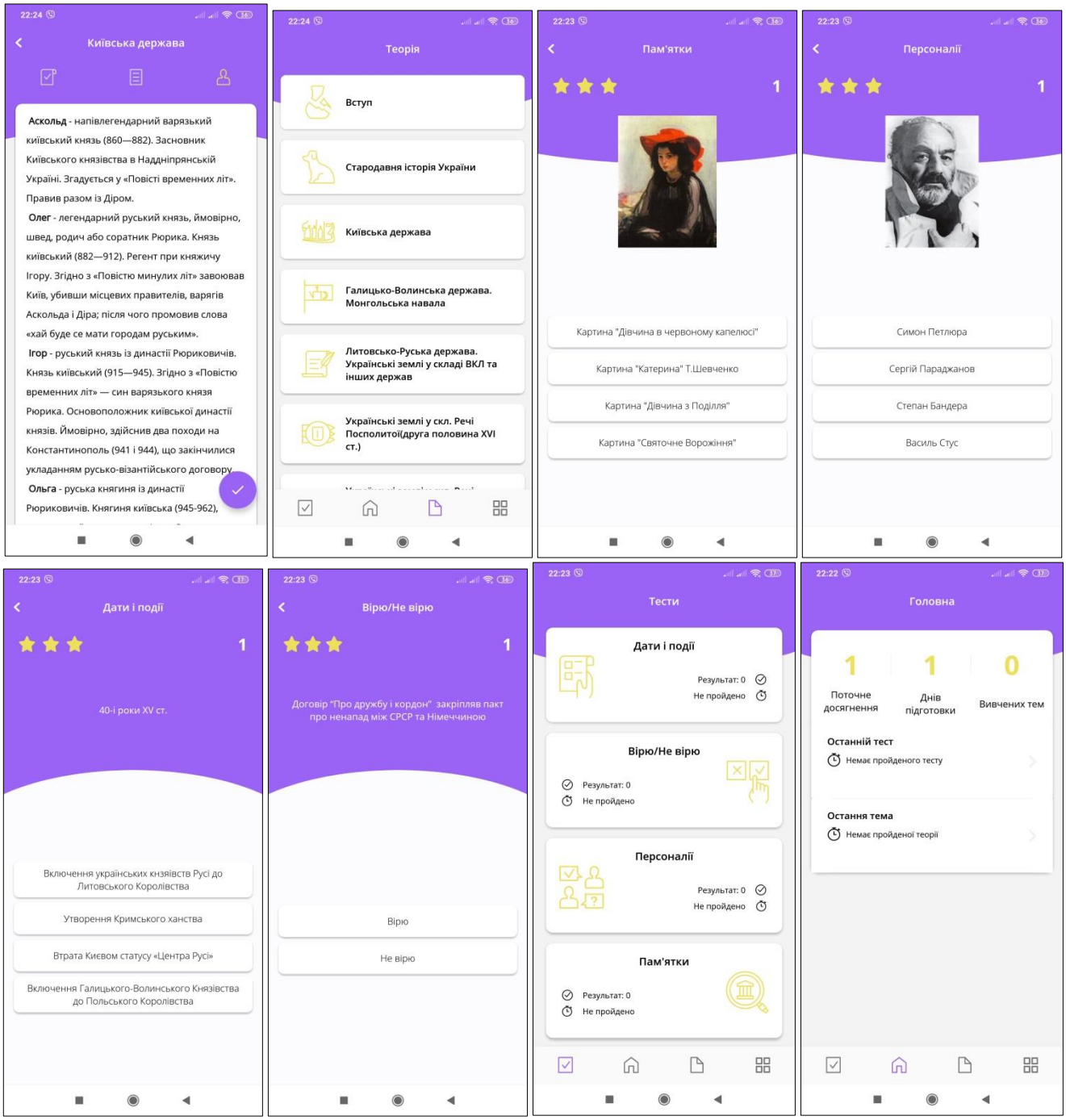


Рисунок 1.1 – Скриншоти додатку «Історія України ЗНО 2021»

До недоліків додатку можна віднести наступні:

відсутність теоретичного матеріалу про персоналії, мапи та пам’ятки мистецтва, необхідність пошуку інформації на інших ресурсах;

- в теоретичних матеріалах відсутні ілюстрації;
- обмежена кількість персоналій для вивчення;

– наявність платної версії;

– недоліки формування питань у вигляді двох однакових варіантів відповіді.

Додаток «ЗНО 2020 тести: Історія України». На Play Маркет має оцінку 4,5 з загальної кількості оцінок 256. На рисунку 1.2 наведено скріншоти різних етапів роботи додатку. До основних переваг додатку можна віднести наступні:

– наявність цитатнику видатних істориків світу

– можливість особистих налаштувань запитань, можливість обрати для генерування певні теми;

– можливість функціонування у режимі offline;

– наявність елементу змагань та обміну успіхами з друзями в соціальних мережах;

– велика база даних, біля 1500 запитань, яка регулярно доповнюється;

– можливість запропонувати власні питання.

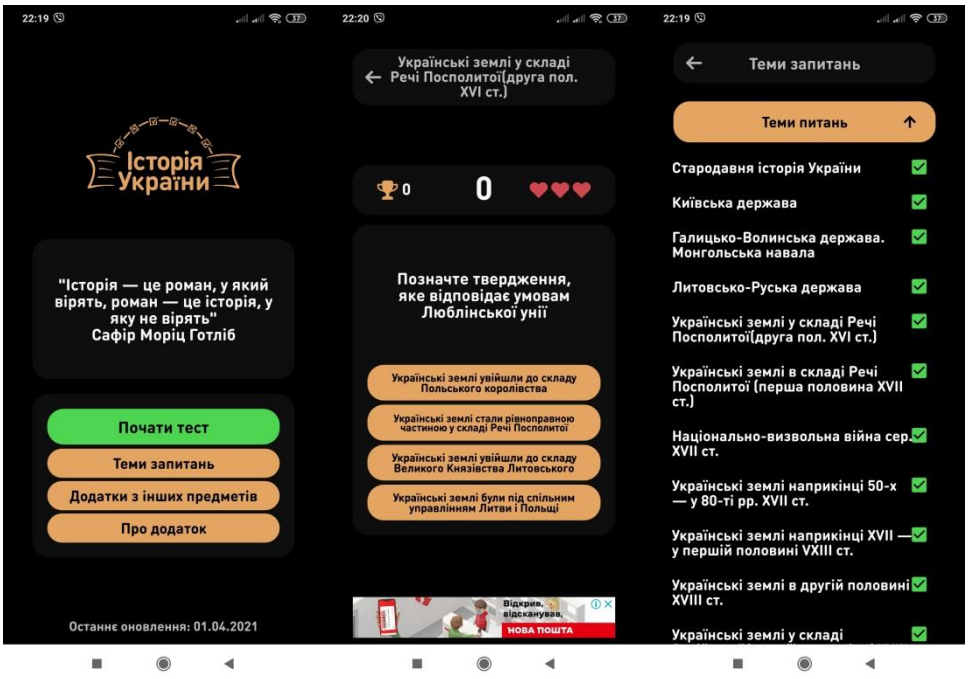


Рисунок 1.2 – Скриншоти додатку «ЗНО 2020 тести: Історія України»

До недоліків додатку можна віднести наступні:

− не всі питання, що пов’язані з зображеннями відображають зображення;

− висока частота півторюваст і питань, особливо у межах однієї чи декількох тем;

Додаток «ЗНО 2021. Історія України» На Play Маркет має оцінку 4,9 з загальної кількості оцінок 1776. На рисунку 1.3 наведено скріншоти різних етапів роботи додатку. До основних переваг додатку можна віднести наступні:

− простий дизайн, що допомагає зосередитись безпосередньо на підготовці до ЗНО;

− інформація відповідає чинній програмі ЗНО – 2021;

− можливість налаштування інтерфейсу додатку;

− надано теоретичний матеріал та таблиці дат та подій;

− можливість виконувати «роботу над помилками» повертаючись до попередніх виконаних тестів;

− можливість використання підказок (показати правильну відповідь, 50 на 50) або пропустити питання.

–

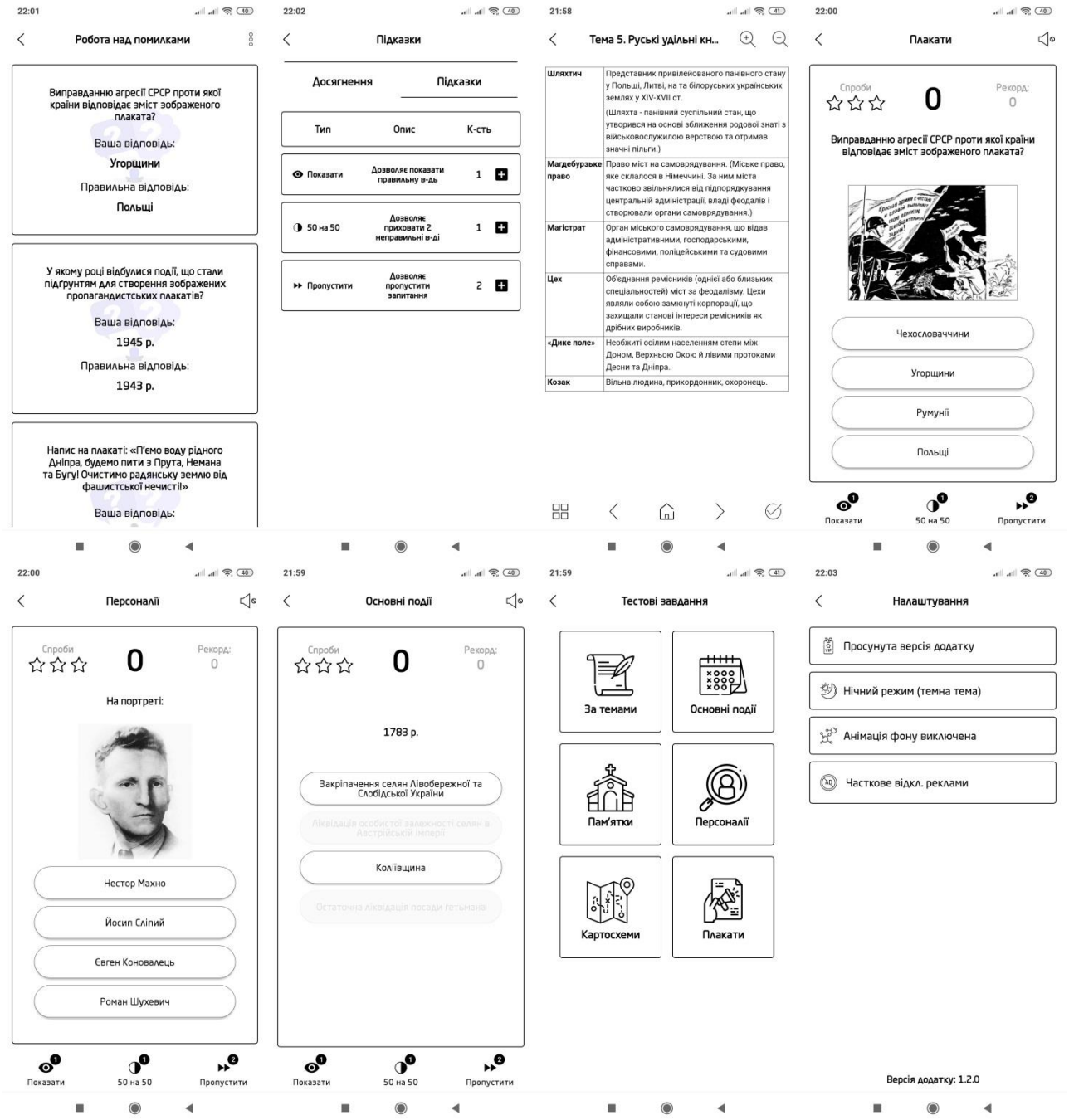


Рисунок 1.3 – Скриншоти додатку «ЗНО 2021. Історія України»

1.3. Аналіз програмних засобів розробки

Для створення інтерактивного програмних продуктів існує багато програм та середовищ, як платних, так і з безкоштовними пакетами версій. Розробка інтерактивного додатку на теперішній час не становить занадто великих витрат як

раніше, але вибір програмного продукту, що буде використано, потребує певного часу. Для реалізації інтерактивної системи було обрано мову програмування Java.

Java – мова програмування загального призначення, відноситься до об'єктно-орієнтованим мовам програмування, до мовам з сильною типізацією. Мову було розроблено компанією Sun Microsystems (в подальшому придбаної компанією Oracle) на початку 90-х років XX століття. Мова є основою практично для всіх типів мережевих додатків і загальним стандартом для розробки і поширення вбудованих і мобільних додатків, ігор, веб-контенту та корпоративного програмного забезпечення. У світі налічується понад 9 мільйонів фахівців, які розробляють додатки на Java, яка дозволяє ефективно розробляти, впроваджувати і використовувати додатки та послуги [10]. Також Java застосовується для роботи з Big Data, розробки програм для наукових цілей, наприклад, обробки природних мов, програмування приладів тощо.

Java дозволяє розробляти високопродуктивні портативні програми практично на всіх комп'ютерних платформах завдяки компіляції написаного на Java коду в байт-код, який виконується середовищем виконання Java (JRE, Java Runtime Environment) або віртуальною машиною Java (JVM, Java Virtual Machine - частина середовища JRE), яка не залежіть від платформи.

В Java реалізований механізм управління пам'яттю, який називається «складальником сміття» або garbage collector. Розробник створює об'єкти, а JRE за допомогою збирача сміття очищає пам'ять, коли об'єкти перестають використовуватися.

Основні можливості, що надаються мовою Java:

- автоматичне управління пам'яттю;
- надання розширених можливостей обробки виняткових ситуацій;
- великий набір засобів фільтрації введення-виведення;
- наявність набору стандартних колекцій: масив, список, стек і т.д.;
- наявність простих засобів створення мережевих додатків;
- наявність класів для виконання HTTP-запитів, отримувати та обробляти відповіді;

− вбудовані в мову засоби створення багатопоточних додатків, які перенесено та використовується багатьма мовами (наприклад, мова Python);

− уніфікований доступ до баз даних: на рівні окремих SQL-запитів та на рівні концепції об'єктів, що володіють здатністю до зберігання в базі даних;

− підтримка узагальнень;

− підтримка лямбда, замикань, можливості функціонального програмування [11].

За результатами рейтингу State of Octoverse 2020 (розраховується за кількістю репозиторіїв на відповідній мові, які зберігаються на GitHub) та рейтингу RedMonk за червень 2020 року Java входить до трійки найбільш популярний мов програмування, випереджає за популярністю PHP, C #, C ++, TypeScript і інші затребувані мови, а поступається тільки JavaScript і Python.

В індексі TIOBE на березень 2021 року Java займає друге місце. Індекс TIOBE – індикатор популярності мов програмування, який розраховується за складною методикою з урахуванням кількості пошукових запитів, які стосуються тій чи іншій мові. При цьому перше місце в цьому індексі займає мову програмування C.

JavaFX — платформа та набір інструментів для створення функціонально насичених Internet-додатків (RIA англ. Rich Internet Applications) здатних працювати в будь-яких браузерах, в мобільних телефонах, телевізорах, плеєрах Blu-ray і інших пристроях. Незважаючи на відмінність мов, платформа JavaFX інтегрована із середовищем JRE, а робоче середовище JavaFX, в свою чергу, інтегрована в пакет завантаження Java. Вперше платформу продемонстровано компанією Sun Microsystems на Міжнародній конференції Java-розробників JavaOne у травні 2007 [11]

Особливості JavaFX:

− представляє інструментарій для створення кроссплатформенних графічних додатків на платформі Java;

− дозволяє створювати додатки з багатою насиченою графікою завдяки використанню апаратного прискорення графіки;

− поставляється з наявністю великого набору частин графічного інтерфейсу, елементів керування, можливостей по роботі з мультимедіа, двомірної і тривимірною графікою;

− використовує стилі CSS, а можливість використання спеціального формату FXML для створення графічного користувацького інтерфейсу полегшує розміщення або зміну зовнішнього вигляду.

IntelliJ IDEA Community Edition – це безкоштовна інтегроване середовище розробки програмного забезпечення для багатьох мов програмування, зокрема Java, JavaScript, Python. Середовище розробляється та підтримується компанією JetBrains, є одним з найбільш функціональних середовищ для Java розробки та оснащена системою інтелектуальної допомоги в написанні коду, налаштовує роботу автодоповнення і доступність інструментів. Велика кількість інструментів дозволяє прискорити розробку, наприклад, за допомогою шаблонів і повторень, а також збільшити продуктивність кінцевої програми. Величезна кількість плагінів і надбудов під будь-яке завдання роблять середу Java розробки IDEA дуже популярним інструментом розробки [12].

Переважні особливості середовища:

− наявність інструмент тестування JUnit;

− наявність інструментів тестування, налагодження і перевірки коду;

− функція завершення коду;

− підтримка множинного рефакторинга;

− наявність редактору коду XML і Java, компілятора, складальника, відгадчика;

− наявність візуального графічного інструмента побудови.

Таким чином, середовище IntelliJ IDEA Community Edition можна вважати найбільш вдалим вибором для розробки інформаційної системи.

1.4. Постановка задачі

Метою виконання роботи бакалавра є проектування та програмна реалізація інтерактивної системи для вивчення історії України з метою підготовки до зовнішнього незалежного тестування.

Для досягнення поставленої мети необхідно виконати наступні задачі:

1) провести огляд предметної області, виконати аналіз сучасного стану питання;
2) оглянути основні види навчаючих завдань та використання тестування для оцінювання якості отриманих знань;
3) розглянути питання використання інтерактивних систем та гейміфікації в освіті та процесі навчання;
4) обґрунтувати обрання засобів програмної реалізації програмної системи;
5) виконати проектування структури, схеми функціонування та інтерфейсу програмної системи;
6) виконати проектування бази даних системи для зберігання даних користувачів та даних для побудови завдань;
7) розробити структуру завдань для тренування та принципів оцінювання результатів навчання при використанні програмної системи;
8) розробити алгоритмічне забезпечення програмної системи;
9) розробити програмне забезпечення програмної системи;
10) провести тестування та оцінку якості роботи програмної системи.

Початкові дані для роботи програмної системи:

1) теоретична інформація з історії України;
2) структурована інформація що до історичних подій, персоналій, витворів мистецтв та архітектурних пам’ятників України відповідно до чинної програми ЗНО по історії України за 2020 рік (текстова та візуальна).

Результат роботи програмної системи:

1) статистичні характеристики якості навчання;

2) підсумкові та проміжні дані результатів навчання користувачів.

Вимоги до програмної системи:

3) простий та зрозумілий інтерфейс користувача, що дозволяє легко орієнтуватися у системі, не відволікає та допомагає зосередитись безпосередньо на меті роботи з програмою;

4) забезпечення багатокористувацького доступу до використання системи, відокремлення результатів навчання окремих користувачів;

5) можливість реєстрації та авторизації користувачів у системі;

6) можливість обрання тематики навчання та виду завдань для перевірки знань;

7) розрахунок та відображення статистичних результатів виконання завдань безпосередньо по закінченні їх виконання;

8) надання користувачеві як підсумкової так і проміжної інформації етапів навчання;

9) збереження поточних результатів навчання користувача;

10) надання користувачеві інформації, що до досягнень інших користувачів з можливістю сортування інформації за бажаним полем;

11) надання користувачеві інструкції, що до використання програмною системою.

2 ПРОЕКТУВАННЯ ПРОГРАМНОЇ СИСТЕМИ

2.1. Діаграма варіантів використання

Опишемо функціональність та поведінку системи за допомогою діаграми варіантів використання.

Діаграми використання дозволяють описувати системи на концептуальному рівні. У найзагальному випадку, діаграма варіантів використання уявляє з себе граф, що надає інформацію про сервіси, які повинна надавати система, а також визначає варіанти використання і типовий спосіб взаємодії користувача з системою. Діаграма призначена для спрощення взаємодії з майбутніми користувачами та іншими клієнтами системи, для визначення необхідних характеристик системи, а також відображає те, що система повинна робити, не вказуючи та не розкриваючи методи, що використовуються для цього.

Основні елементи діаграми:

1) варіанти використання – вказують специфікацію загальних особливостей поведінки системи або будь-якої іншої сутності предметної області без розгляду внутрішньої структури цієї сутності, кожен варіант використання визначає послідовність дій, які повинні бути виконані проектованої системою при взаємодії її з відповідним актором;

2) актори – будь-яка зовнішня по відношенню до системи, що моделюється, сутність, яка взаємодіє з системою і використовує її функціональні можливості для досягнення певної мети або вирішення приватних завдань;

3) відношення – описують взаємодію екземплярів одних акторів і варіантів використання з екземплярами інших акторів і варіантів, використовуються наступні види відносин між акторами та варіантами використання:

- відношення асоціації (association relationship) – використовується для позначення специфічної ролі актора в окремому варіанті використання;

– відношення розширення (extend relationship) – визначають взаємозв'язок екземплярів окремого випадку використання з більш загальним варіантом, властивості якого визначаються на основі способу спільного об'єднання даних екземплярів;
– відношення узагальнення (generalization relationship) – вказують на факт, що деякий варіант використання А може бути узагальнено до варіанту використання В, у цьому випадку варіант А буде спеціалізацією варіанту В;
– відношення включення (include relationship) – вказує, що деякий заданий поведінка для одного варіанта використання включається як складовий компонент в послідовність поведінки іншого варіанту використання.

На рисунку 2.1 наведено розроблену діаграму варіантів використання.

При проектуванні системи було виявлено акторів:

1) користувач – особа, що є кінцевим користувачем системи;

2) база даних - використовує сервіси отримання, додання та зміни сукупності даних, що зберігаються;

3) файлова система – використовує сервіси надання даних.

При проектуванні системи були виділені наступні варіанти використання системи:

1) авторизація користувача – проведення пошуку наданих користувачем даних (логін, пароль) у базі даних серед зареєстрованих користувачів, при наявності вказаних даних надання користувачеві можливості користуватися системою;

2) реєстрація користувача – перевірка наявністі вказаних користувачем даних (логін, e-mail) у базі даних серед зареєстрованих користувачів, при відсутності аналогічних даних додання даних нового користувача до бази даних користувачів системи, при наявності аналогічних даних надання можливості зміни даних, що надав користувач для реєстрації;

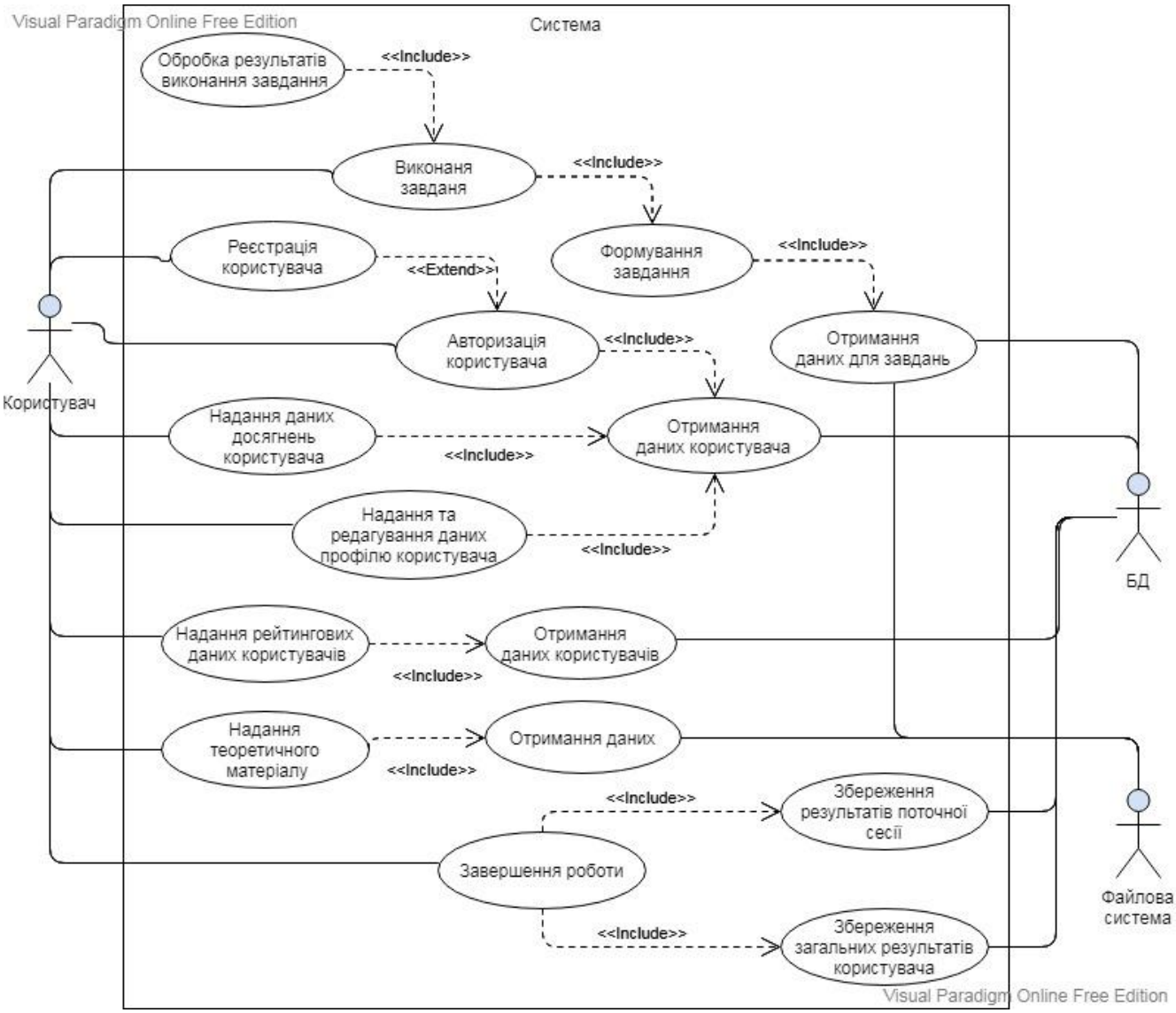


Рисунок 2.1 – Діаграма варіантів використання системи

3) отримання даних користувача – запит до бази даних, отримання підсумкових та проміжних оцінок користувача, що до результатів виконання завдань;

4) отримання даних для завдань – запит до бази даних, отримання даних для формування завдань, збереження отриманої інформації у структурах даних системи;

5) формування завдань – формування даних для завдання відповідно до його типу, збереження сформованих даних у відповідній структурі даних системи;

6) виконання завдання – візуалізація сформованих даних завдань, відповідно до їх типу та змісту, отримання даних від користувача: відстеження

подій маніпулятора «миша» та натискання кнопок форми, визначення коректності наданих користувачем даних, врахування обмежень, що до виконання завдань;

7) обробка результатів виконання завдання – розрахунок оцінок результатів виконання завдання, надання користувачеві інформації, що до отриманих результатів розрахунків

8) надання та редагування даних користувача – надання користувачеві можливості перегляду, зміни та збереження наданих даних, що до профілю користувача, які зберігаються у базі даних;

9) надання даних досягнень користувача – візуалізація поточних та проміжних оцінок результатів виконання завдань користувачем;

10) надання рейтингових даних користувачів – візуалізація поточних оцінок результатів виконання завдань усіх користувачів системи;

11) отримання даних користувачів – запит до бази даних, отримання підсумкових оцінок усіх користувачів, що до результатів виконання завдань;

12) надання теоретичного матеріалу – візуалізація даних з файлів файлової системи, що містять інформацію, що необхідно засвоїти для успішного виконання завдань;

13) отримання даних – звернення до файлової системи та читання даних файлів;

14) збереження результатів поточної сесії – запит до бази даних, збереження проміжних оцінок результатів виконання завдань користувачем;

15) збереження загальних результатів користувача – запит до бази даних, збереження поточних стогових оцінок результатів виконання завдань користувачем;

16) завершення роботи – збереження даних, що до результатів виконання користувачем завдань, завершення процесів системи.

2.2. Діаграма діяльності

Моделювання поведінки проектованої або аналізованої системи потребує необхідності деталізувати особливості логічної та алгоритмічної реалізації операцій, що виконуються системою. Для моделювання процесу виконання операцій системи використовують діаграми діяльності, які саме і показують дії, що призводять до зміну станів системи. Кожній дії на діаграмі діяльності відповідає виконанню деякої елементарної операції, а перехід до наступної дії спрацьовує тільки при операції в попередньому стані. Графічно діаграма діяльності уявляє з себе граф діяльності, вершинами якого є дії системи, а дугами - переходи від одної дії до іншої.

Діаграми діяльності складаються з наступних елементів:

– дія або операція (action state), вузол управління (control node) – це абстрактний вузол дії, який координує потоки дій:

– рішення, вузол рішення призначений для визначення правила розгалуження і різних варіантів подальшого розвитку сценарію, при цьому в точку розгалуження входить рівно один перехід, а виходить - два або більше;

– початок (розгалуження) і закінчення (сходження) розгалуження дій, вузол об'єднання має два і більше вузла, що входять вузла і один вихідний;

– переходи, йдуть від початку до кінця процесу і показують потоки управління, потоки об'єктів або даних;

– початковий вузол діяльності ((activity initial node) є вузлом управління, в якому починається потік (або потоки) при виклику даної діяльності ззовні;

– кінцевий вузол діяльності (activity final node) є вузлом управління, який зупиняє всі потоки даної діаграми діяльності, при цьому на діаграмі може бути більш одного кінцевого вузла.

На рисунку 2.2 наведено розроблену діаграму діяльності для програмної системи.

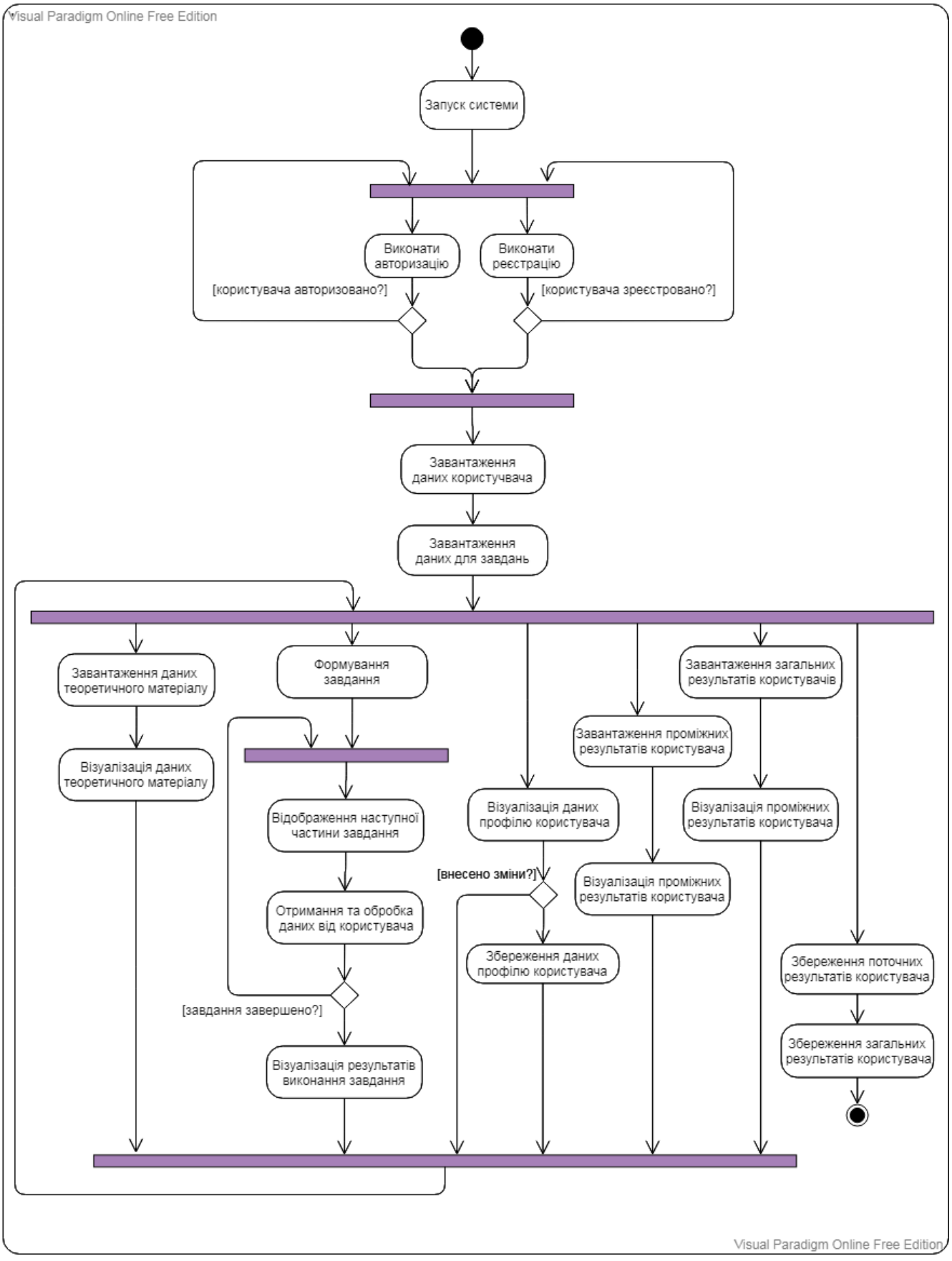


Рисунок 2.2 – Діаграма діяльності

Робота системи починається з запуску програми та відображення вікна авторизації.

Після надання користувачем даних для авторизації чи реєстрації система перевіряє надані дані та дозволяє продовжувати роботу чи очікує інших даних від користувача. При продовженні роботи система завантажує дані користувача та даних, необхідних системі для формування завдань та переходить у режим очікування подальших дій користувача. Користувач повинен мати можливість:

– виконувати завдання відповідно обраним типу та тематики (система повинна формувати завдання, відображати їх, обробляти результати їх виконання);

– отримати результати виконання завдання (система повинна візуалізувати результати);

– переглянути свої загальні та проміжні результати навчання (система повинна завантажити відповідні дані з бази даних та візуалізувати їх);

– переглянути рейтингові результати навчання усіх користувачів системи (система повинна завантажити відповідні дані з бази даних та візуалізувати їх);

– переглянути та змінити свої дані (дані профілю) (система повинна завантажити та візуалізувати їх, зберегти зміни у базі даних)

– отримати теоретичні дані, що необхідні для виконання завдань (система повинна завантажити та візуалізувати відповідні дані);

– завершити роботу з системою (система повинна зберегти поточні та загальні результати навчання).

2.3. Розробка структури завдань та алгоритмів

2.3.1. Вибір змісту завдань та розробка макетів розміщення даних

Відповідно до огляду видів та структури завдань перевірки знань та умінь (п.1.1.2), видів тестів (п.1.1.3), форм та структур тестових завдань ЗНО (п.1.1.1) було обрано для реалізації в інформаційні системі завдань трьох типів:

- питання з вибором однієї правильної відповіді з чотирьох («правильні відповідь»);
- надання факту з підтвердженням чи є він істиною («вірно чи ні»);
- встановлення відповідності між двома фактами («знайти відповідність»).

Відповідно до огляду тематичного змісту завдань с ЗНО по історії України (п.1.1.1) було обрано чотири тематичних розділи для вивчення:

- персоналії;
- історичні дати;
- образотворче мистецтво;
- архітектура.

Тематичний розділ «Персоналії» містить інформацію про суспільно-політичні і військових діячів та діячі культури, освіти і науки. Для використання у навчанні було обрано наступний набір даних для об’єктів розділу:

- ім’я;
- роки життя;
- портретне зображення;
- коротка характеристика діяльності, що дозволяє однозначно ідентифікувати історичну особу.

Тематичний розділ «Історичні дати» містить інформацію про історичні події та коли вони відбувалися. Для використання у навчанні було обрано наступний набір даних для об’єктів розділу:

- назва події;
- дата (період) коли подія відбувалася.

Тематичний розділ «Образотворче мистецтво» містить інформацію про історично значущі пам’ятники, скульптури, художні витвори, пам'ятки давньої історії тощо. Для використання у навчанні було обрано наступний набір даних для об’єктів розділу:

- назва витвору мистецтва;
- фотографічне зображення.

Тематичний розділ «Архітектура» містить інформацію про пам’ятки архітектури. Для використання у навчанні було обрано наступний набір даних для об’єктів розділу:

- назва пам’ятки архітектури;
- фотографічне зображення.

Відповідно до обраних типів питань та наборів даних було розроблено макети розміщення даних при побудові завдань:

- для питання «правильна відповідь» розділу «Персоналії» (рис 2.3);
- для питання «правильна відповідь» розділу «Історичні дати» (рис. 2.4);
- для питання «правильна відповідь» розділу «Образотворче мистецтво» та «Архітектура» (рис. 2.5 а, б відповідно);
- для питання «правда чи ні» розділу «Персоналії» (рис. 2.6);
- для питання «правда чи ні» розділу «Історичні дати» (рис. 2.7)
- для питання «правда чи ні» розділів «Образотворче мистецтво» та «Архітектура» (рис. 2.8);
- для завдання «знайти відповідність» розділу «Персоналії» (рис.2.9);
- для завдання «знайти відповідність» розділу «Історичні дати» (рис. 2.10);
- для завдання «знайти відповідність» розділів «Історичні дати» «Образотворче мистецтво» та «Архітектура» (рис. 2.11).

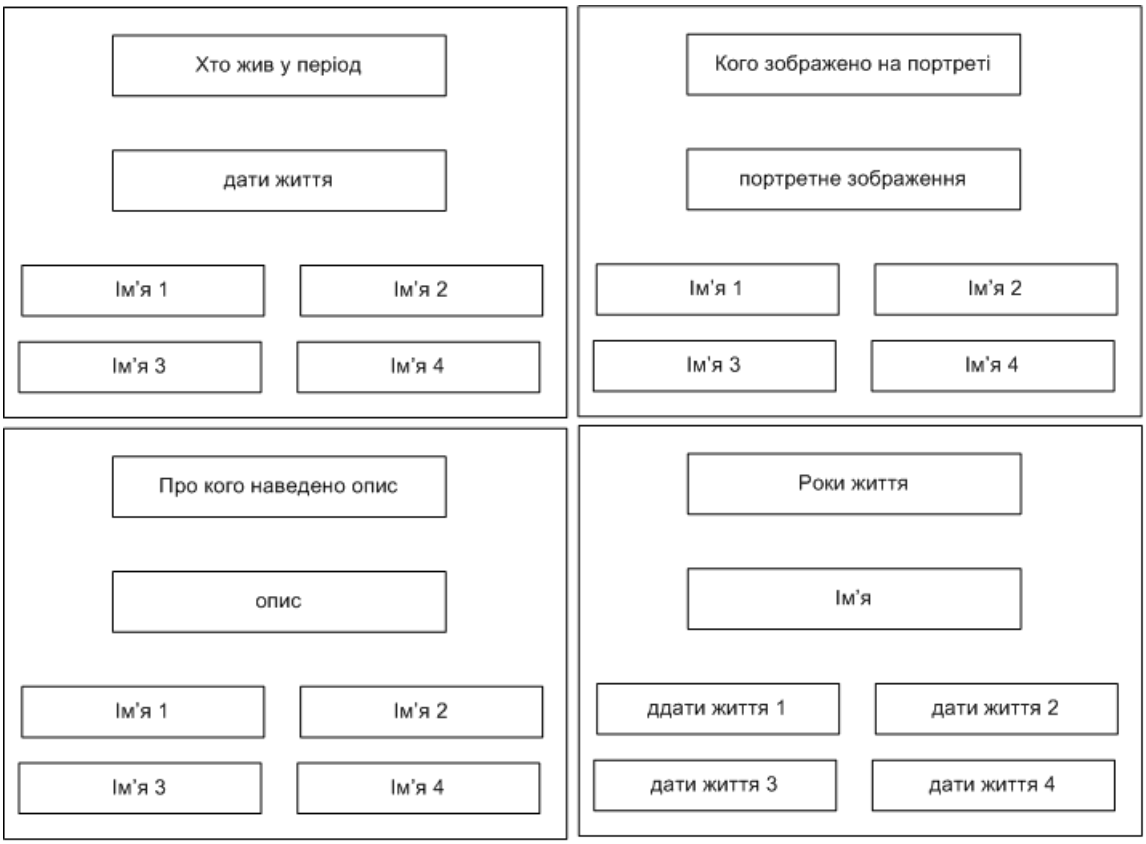


Рисунок 2.3 – Макети розміщення даних для питань «правильна відповідь» розділу «Персоналії»

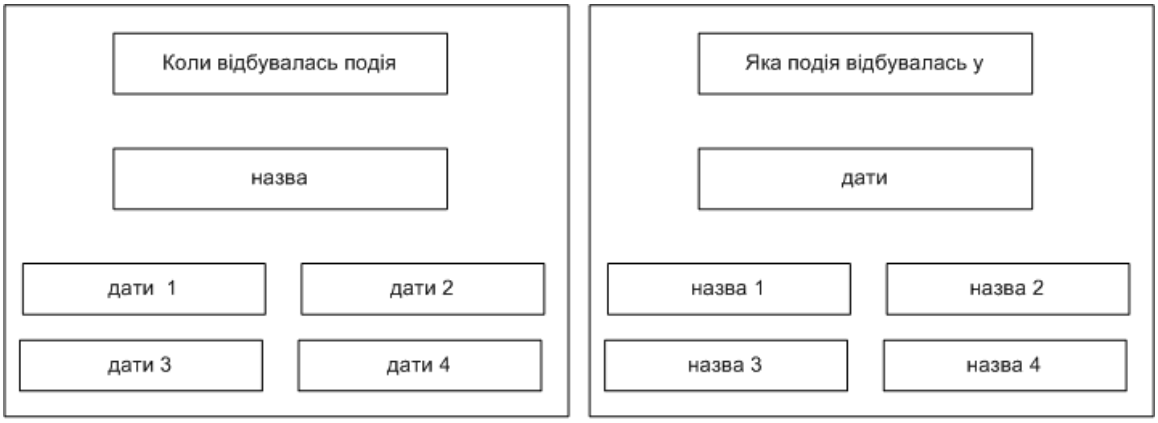


Рисунок 2.4 – Макети розміщення даних для питань «правильна відповідь» розділу «Історичні дати»

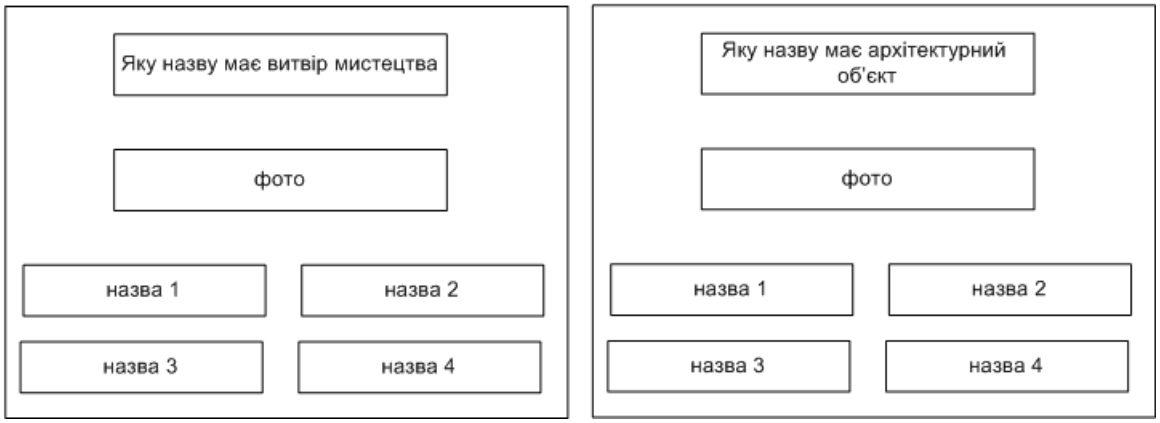


Рисунок 2.5 – Макети розміщення даних для питань «правильна відповідь» розділів «Образотворче мистецтво» (а) та «Архітектура» (б)

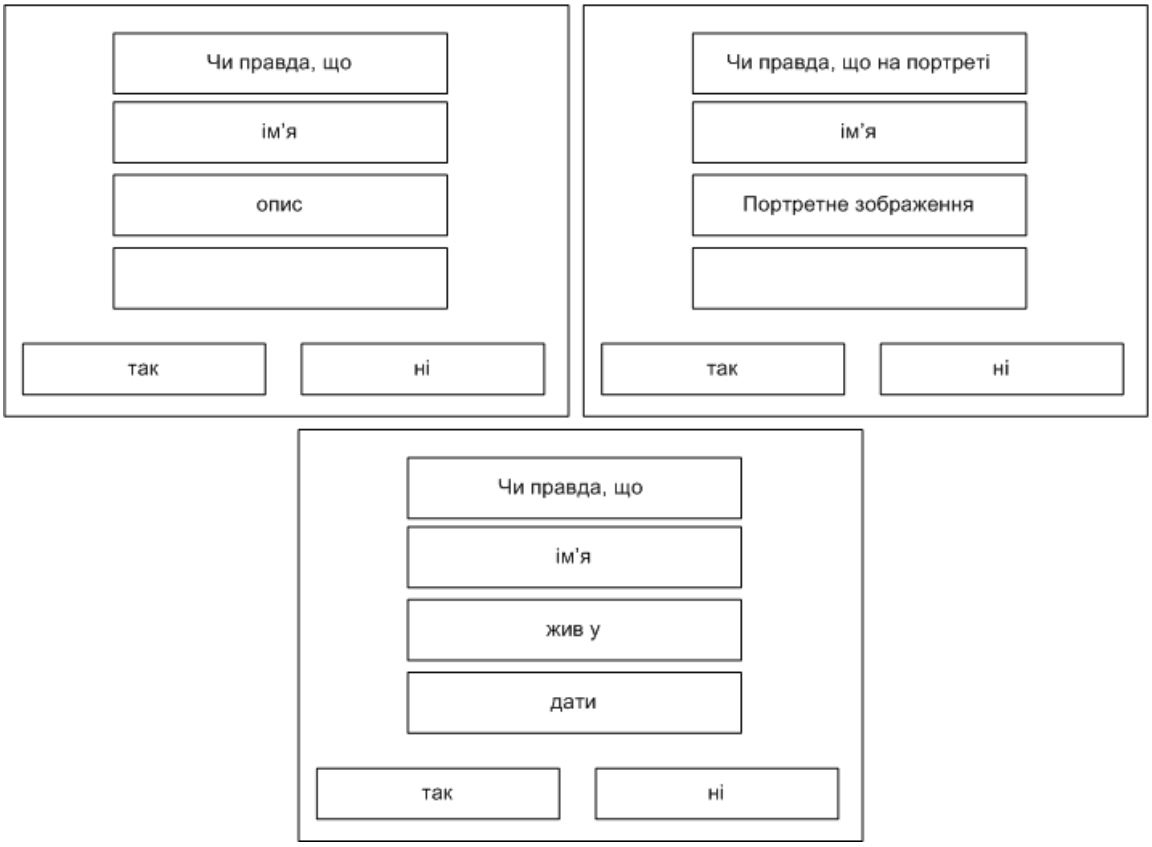


Рисунок 2.6 – Макети розміщення даних для питань «правда чи ні» «Персоналії»

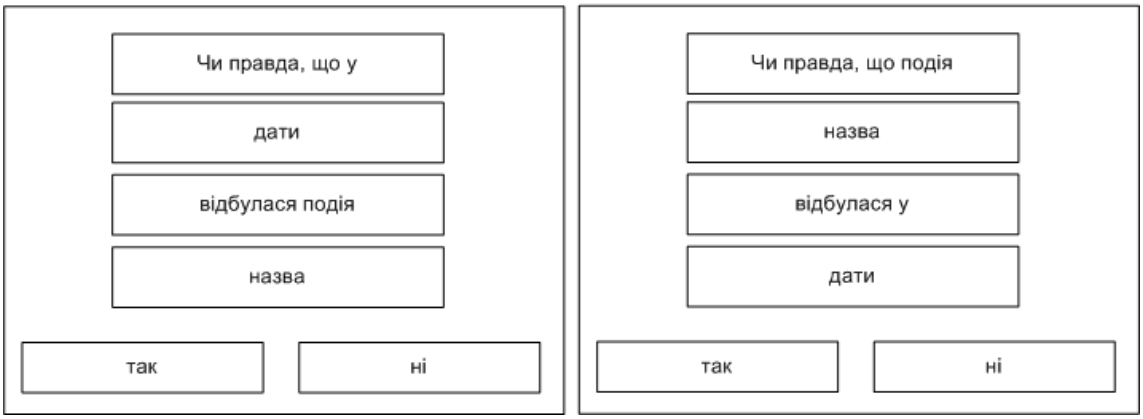


Рисунок 2.7 – Макети розміщення даних для питань «правда чи ні» розділу «Історичні дати»

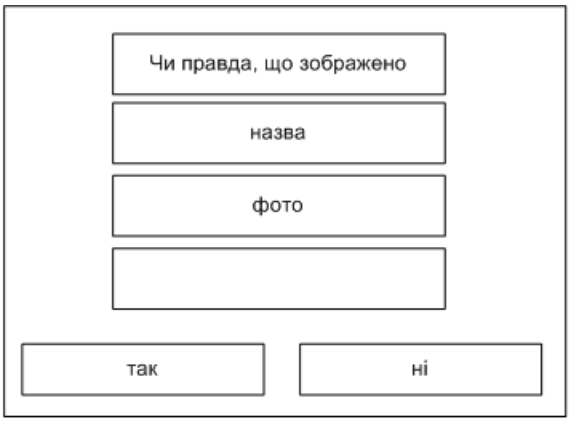


Рисунок 2.8 – Макет розміщення даних для питань «правда чи ні» розділів «Образотворче мистецтво» та «Архітектура»

Рисунок 2.9 – Макет розміщення даних для завдання «знайти відповідність» розділу «Персоналії»

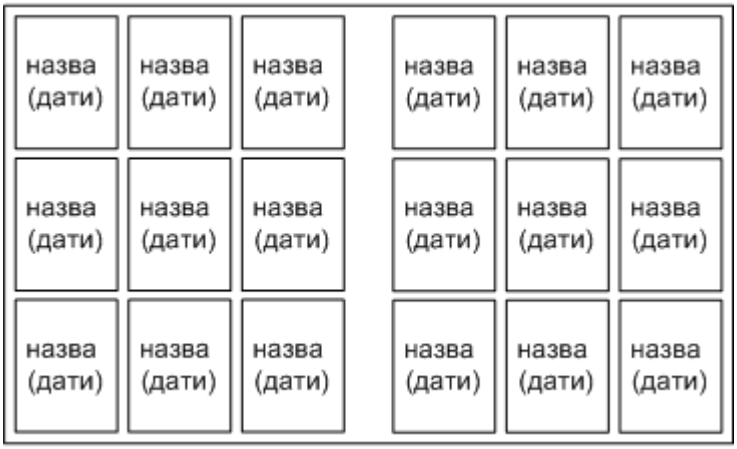


Рисунок 2.10 – Макет розміщення даних для завдання «знайти відповідність» розділу «Історичні дати»

Рисунок 2.11 – Макет розміщення даних для питання «знайти відповідність» розділів «Образотворче мистецтво» та «Архітектура»

2.3.2. Розробка алгоритмів побудови завдань

Відповідно до розроблених макетів було розроблено алгоритми формування завдань. У розроблених блок-схемах використовуються наступні позначення:

– int numQuestion – кількість запитань у завданні;

– list <obj> – перелік записів за тематичним розділом;

– int recordNum – номер запису з переліку записів за тематичним розділом;

– int set – множина унікальних номерів обраних записів;

– obj – запис з переліку записів за тематичним розділом;

– int nRight – номер відповіді, що є вірною на запитання;

– string Str1 – загальне питання;

– string Str2 – перше текстове поле питання (хедер);

– string Str3 – зображення до питання;

– string Str4 – друге текстове поле (опис);

– string Str5 – третє текстове поле (футер);

– string Answers[4] – масив відповідей;

– int num - номер запису з переліку записів за розділом, для отримання дестрактних відповідей (невірних);

– QuestionList <Questio> – перелік питань;

– Question – об’єкт типу «питання»;

− random(a) – метод отримання випадковго цілого значення в діапазоні [0,a-1]

− boolean isTrue – логічне значення, висловлювання в запитанні істина чи ні

− imageList <Label> - перелік елементів керування для відображення зображень;

− textList <Label> - перелік елементів керування для відображення тексту;

− numList <int> - перелік номерів обраних записів за тематичним розділом;

− String text – текст для відображення;

− String image – шлях до файлу с зображенням зображення для відображення;

− add() – метод додання об’єкту до переліку;

− new() – метод створення нового об’єкту;

− getDate() – метод отримання властивості date об’єкту;

− getDescription() – метод отримання властивості description об’єкту;

− getName() – метод отримання властивості name об’єкту;

− getImage() - метод отримання властивості image об’єкту;

− init() – метод ініціалізації переліку;

− remove() – метод видалення об’єкту з переліку;

− visualization() – метод візуалізації;

− processing() – метод обробки даних.

На рисунку 2.12 наведено блок-схему розробленого алгоритму формування переліку питань для завдання типу «правильна відповідь» для розділу «Персоналії».

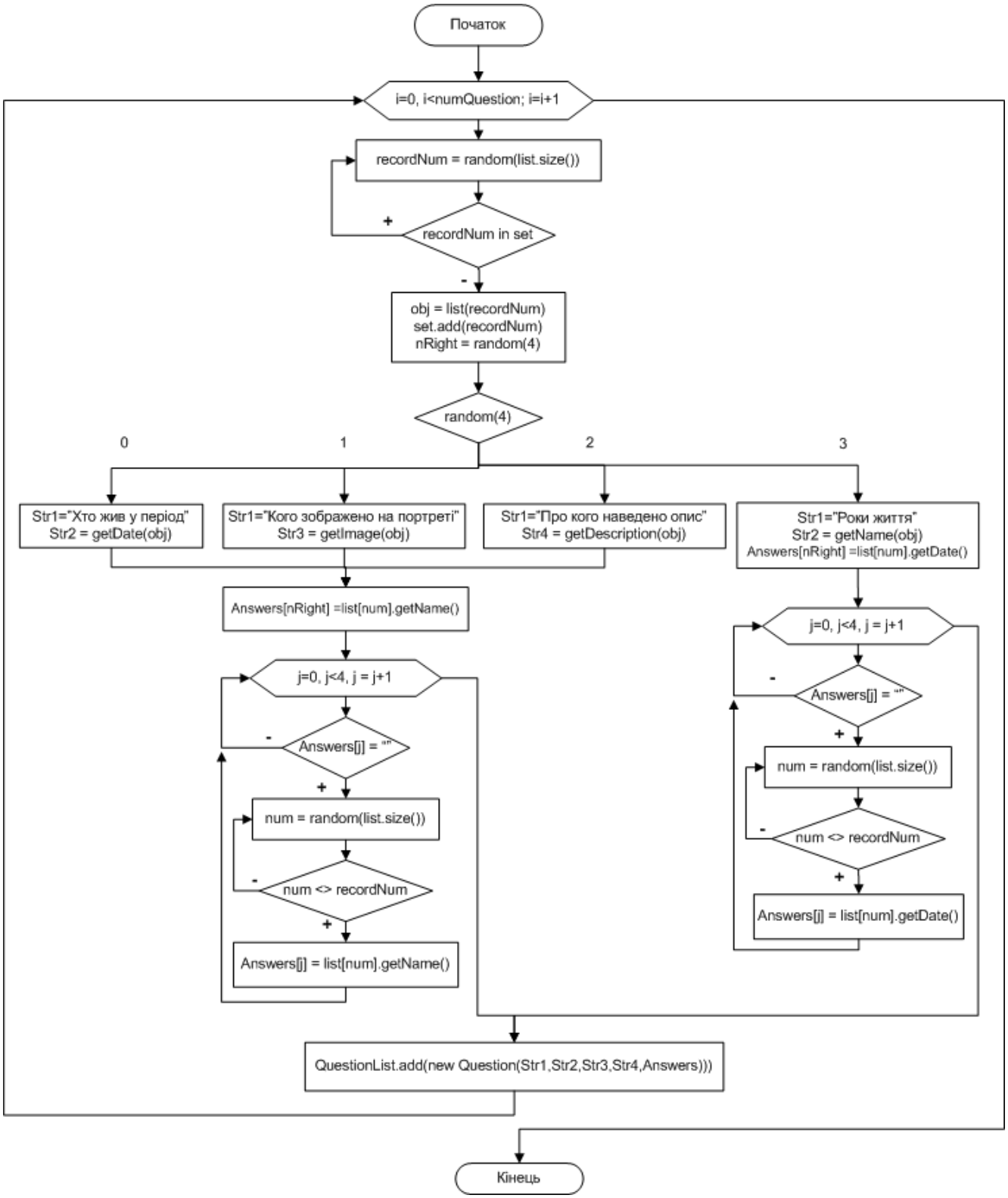


Рисунок 2.12 – Блок-схема алгоритму побудови переліку питань завдання «правильна відповідь» розділу «Персоналії»

Для обрання даних для побудови чергового питання випадковим чином обирається значення номеру запису у переліку даних про персоналії таким чином, щоб він не повторювався з попередніми, зчитується обїекту з переліку персоналій за отриманим номером, а номер додається до множини унікальних значень. Випадковим чином обирається номер відповіді, яка буде уявляти вірну відповідь на питання Наступним кроком обирається набор даних для побудови запитання (чотири для питань за розділом «Персоналії», см.рис.2.3) випадковим чином:

- перший набор – питання «Хто жив у певний період»+дати життя;
- другий набор – питання «Кого зображено на портреті»+портрет;
- третій набір – питання «Про кого наведено опис»+опис.

Для наданого переліку відповідями будуть імена персоналій, тому у якості вірної відповіді заноситься інформація щодо імені поточної персоналії, для визначення інших дистрактних відповідей випадковим чином визначаються обїекту інших персоналій та їх імена заносяться до масиву відповідей.

Для набору даних питання «Роки життя»+ім’я відповідями будуть дати життя персоналії, тому у якості вірної відповіді заноситься інформація щодо дат життя поточної персоналії, для визначення інших дистрактних відповідей випадковим чином визначаються обїекту інших персоналій та їх дати заносяться до масиву відповідей. Наступним кроком створюється об’єкт питання Question з сформованим набором даних та додається до переліку питань. Формування питання виконується стальки разів, скільки питань у завданні.

Блок-схеми розроблених алгоритмів формування переліку питань для завдання типу «правильна відповідь» розділів «Образотворче мистецтво» та «Архітектура» аналогічні та відрізняються лише текстів загального питання, тому наведемо блок-схеми на одному рисунку 2.13, та приведемо загальний опис роботи алгоритму.

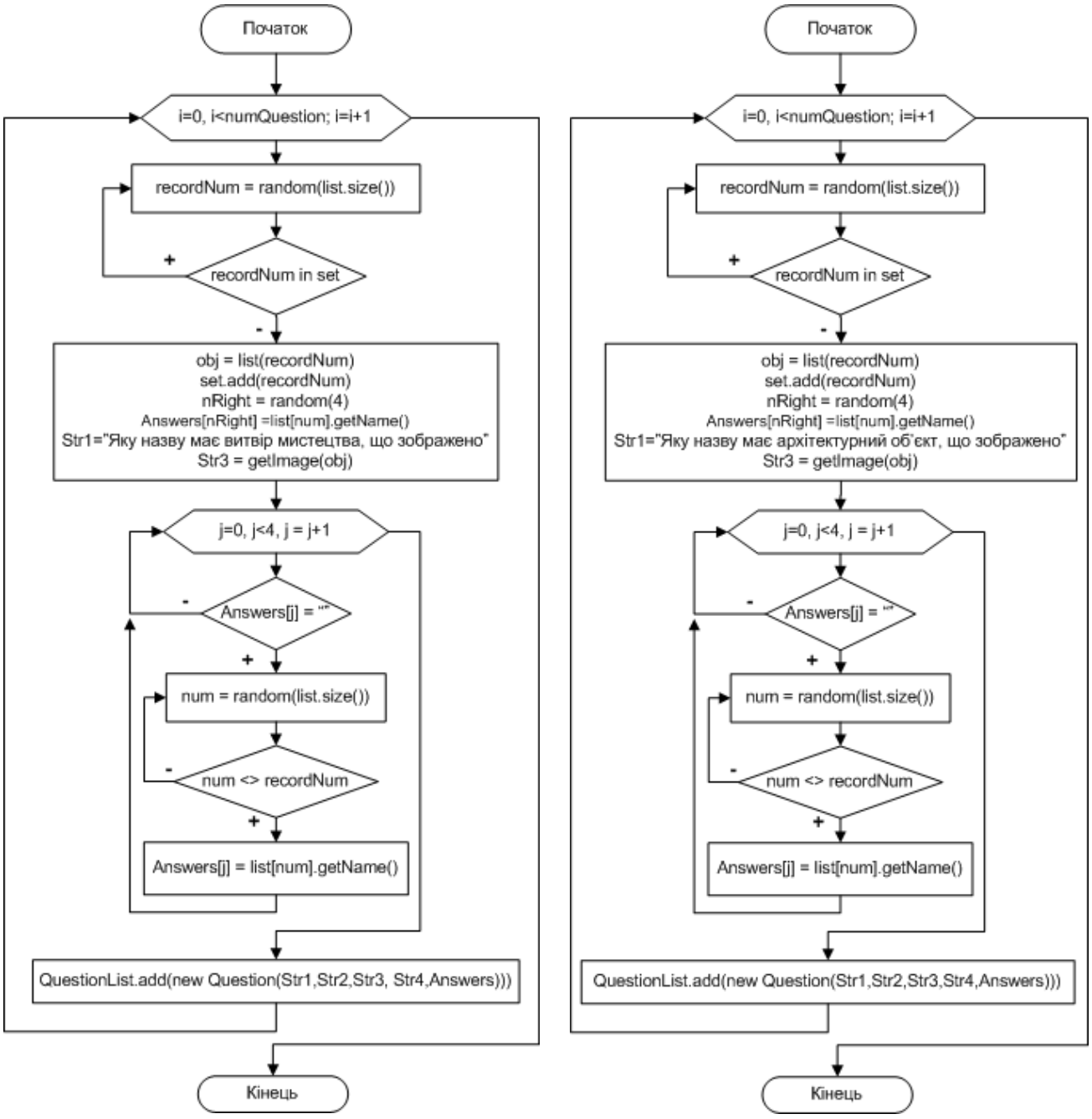


Рисунок 2.13 – Блок-схеми алгоритму побудови переліку питань завдання «правильна відповідь» розділів «Образотворче мистецтво» (а) та «Архітектура»

Для обрання даних для побудови чергового питання випадковим чином обирається значення номеру запису у переліку даних про витвір мистецтва (архітектури) таким чином, щоб він не повторювався з попередніми, зчитується об’єкту з переліку витворів мистецтва (архітектури) за отриманим номером, а номер додається до множини унікальних значень. Зчитується шлях до файлу с зображенням та формується питання до зображення «Яку назву має витвір

мистецтва (архітектурний об’єкт), що зображено. Випадковим чином обирається номер відповіді, яка буде уявляти вірну відповідь на питання. У якості вірної відповіді заноситься інформація щодо назви поточного об’єкту витвору мистецтва (архітектури), для визначення інших дистрактних відповідей випадковим чином визначаються об’єкти інших витворів мистецтва (архітектури) та їх назви. Наступним кроком створюється об’єкт питання Question з сформованим набором даних та додається до переліку питань. Формування питання виконується стальки разів, скільки питань у завданні.

На рисунку 2.14 наведено блок-схему розробленого алгоритму формування переліку питань для завдання типу «правильна відповідь» для розділу «Історичні дати».

При обранні даних для побудови чергового питання випадковим чином обирається значення номеру запису у переліку даних про історичні дати таким чином, щоб він не повторювався з попередніми, зчитується об’єкт з переліку історичних дат за отриманим номером, а номер додається до множини унікальних значень. Випадковим чином обирається номер відповіді, яка буде уявляти вірну відповідь на питання Наступним кроком обирається набор даних для побудови запитання (два для питань за розділом «Історичні дати») випадковим чином.

Для набору даних питання «Коли відбулася оподія»+назва події відповідями будуть дати, тому у якості вірної відповіді заноситься інформація щодо дати, коли відбулася подія, для визначення інших дистрактних відповідей випадковим чином визначаються об’єкти інших історичних подій та їх дати заносяться до масиву відповідей.

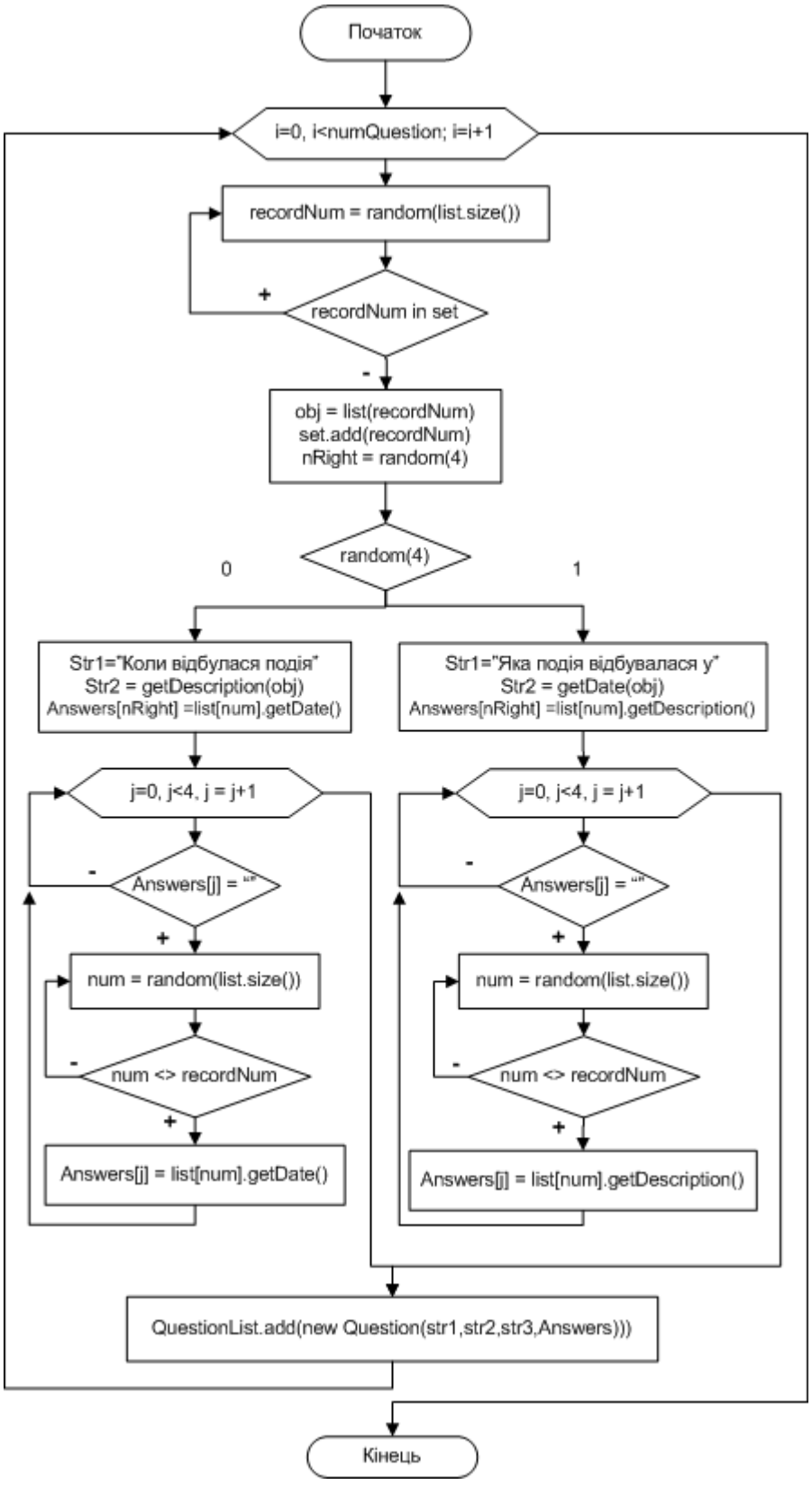


Рисунок 2.14 – Блок-схеми алгоритму побудови переліку питань завдання «правильна відповідь» розділу «Історичні дати»

Для набору даних питання «Яка подія відбувалася у»+дата відповідями будуть назви події, тому у якості вірної відповіді заноситься інформація щодо назви події, для визначення інших дистрактних відповідей випадковим чином визначаються об’єкти інших історичних подій та їх назви заносяться до масиву відповідей. Наступним кроком створюється об’єкт питання Question з сформованим набором даних та додається до переліку питань. Формування питання виконується стальки разів, скільки питань у завданні.

На рисунку 2.15 наведено блок-схему розробленого алгоритму формування переліку питань для завдання типу «правда чи ні» для розділу «Персоналії».

При обранні даних для побудови чергового питання випадковим чином обирається значення номеру запису у переліку даних про персоналії дати таким чином, щоб він не повторювався з попередніми, зчитується об’єкт з переліку персоналій за отриманим номером, а номер додається до множини унікальних значень. Зчитується ім’я персоналії. Випадковим чином обирається тип набору даних для питання, чи буде сформований факт істиною. Якщо факт повинен буде сформований як помилковий, випадковим чином обирається інший об’єкт з переліку персоналій для отримання невідповідних даних. Наступним кроком випадковим чином обирається тип набору даних. Відповідно отриманому значенню формується загальне питання та зчитується інша частина: опис, портретне зображення, або дати життя, а поточного об’єкту персоналії зчитується ім’я. Наступним кроком створюється об’єкт питання Question з сформованим набором даних та додається до переліку питань. Формування питання виконується стальки разів, скільки питань у завданні.

Блок-схеми розроблених алгоритмів формування переліку питань для завдання типу «правда чи ні» розділів «Образотворче мистецтво» та «Архітектура» повністю аналогічні, тому наведемо загальну блок-схему на одному рисунку 2.16, та приведемо загальний опис роботи алгоритму.

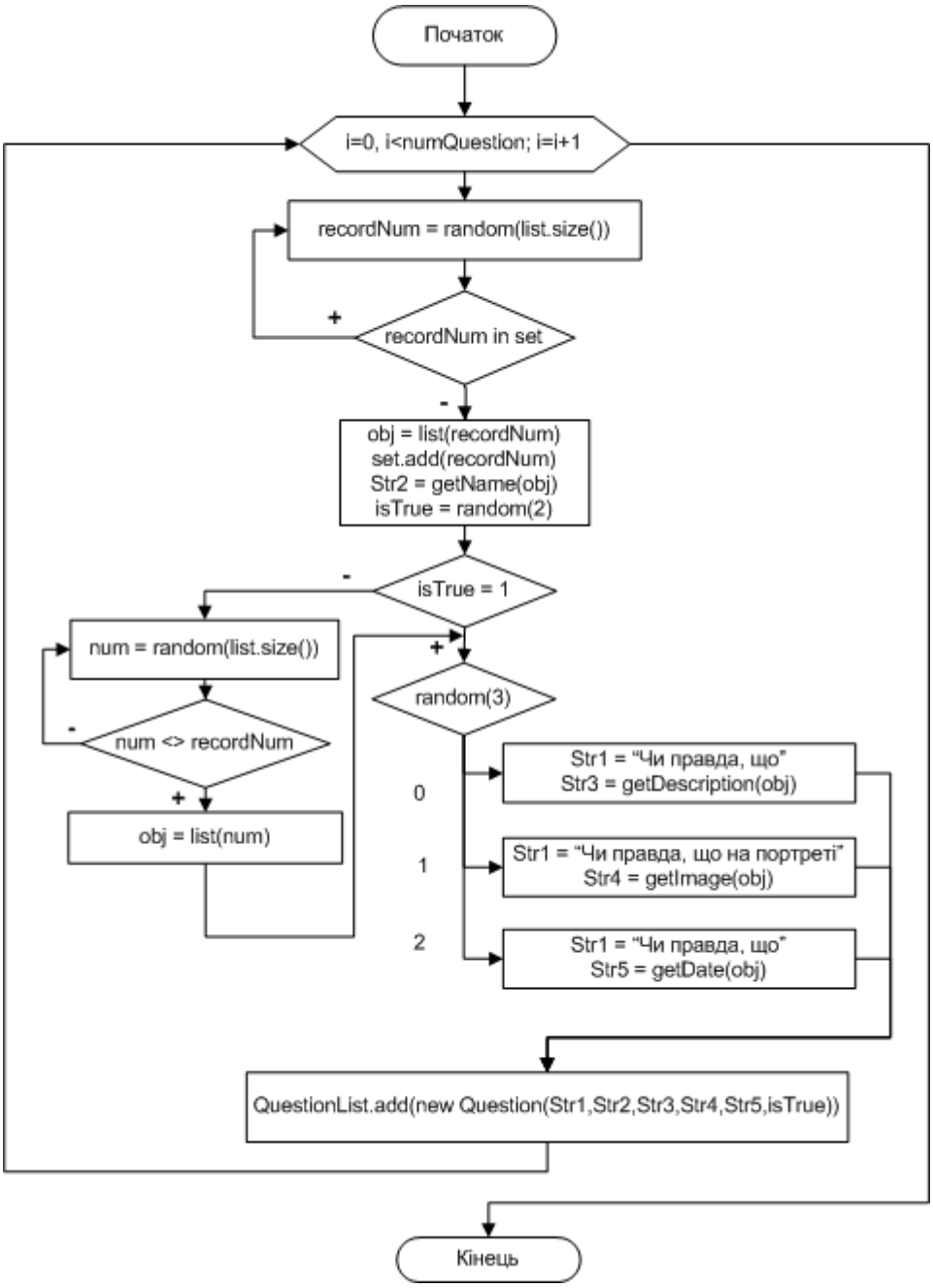


Рисунок 2.15 – Блок-схеми алгоритму побудови переліку питань завдання «правда чи ні» розділу «Персоналії»

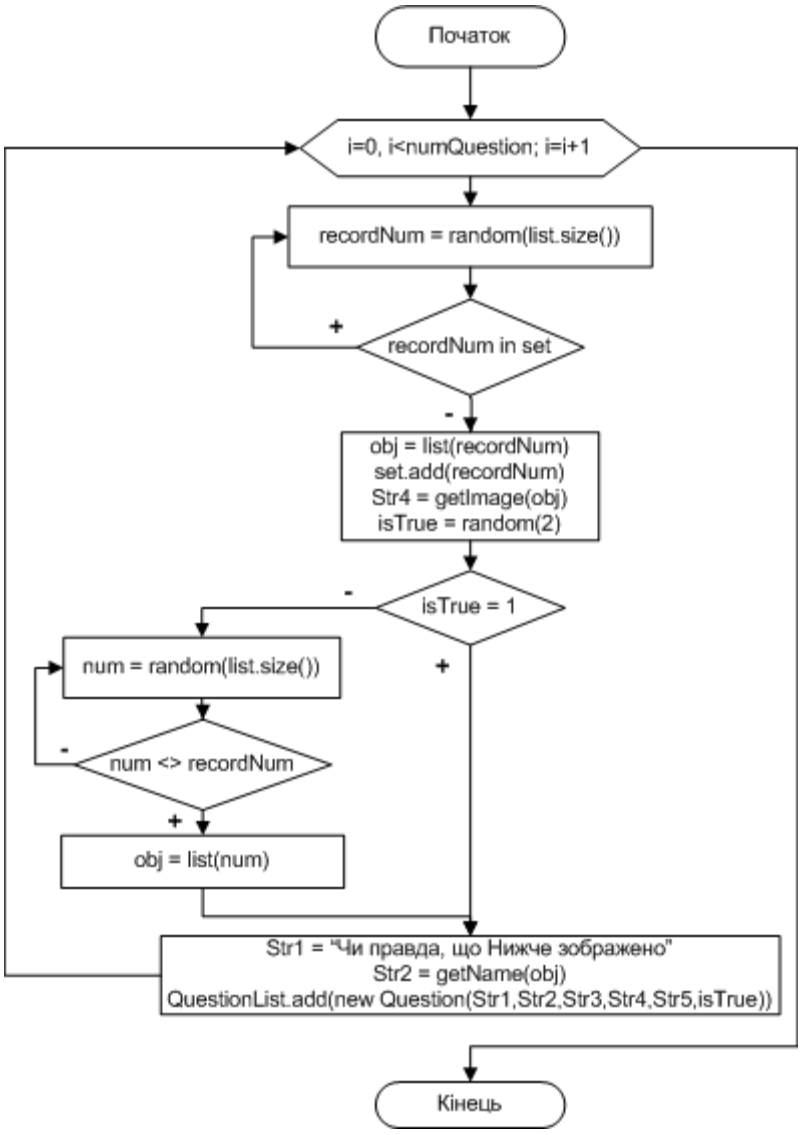


Рисунок 2.66 – Блок-схеми алгоритму побудови переліку питань завдання «правда чи ні» розділів «Образотворче мистецтво» та «Архітектура»

При обранні даних для побудови чергового питання випадковим чином обирається значення номеру запису у переліку даних про витвір мистецтва (архітектури) таким чином, щоб він не повторювався з попередніми, зчитується об’єкт з переліку витворів мистецтва (архітектури) за отриманим номером, а номер додається до множини унікальних значень. Зчитується шлях до файлу с зображенням витвору мистецтва (архітектури). Випадковим чином обирається тип набору даних для питання, чи буде сформований факт істиною. Якщо факт

повинен буде сформований як помилковий, випадковим чином обирається інший об’єкт з переліку витворів мистецтва (архітектури) для отримання невідповідних даних. Формується загальне питання та зчитується назва поточного витвору мистецтва (архітектури) Наступним кроком створюється об’єкт питання Question з сформованим набором даних та додається до переліку питань. Формування питання виконується стальки разів, скільки питань у завданні.

На рисунку 2.17 наведено блок-схему розробленого алгоритму формування переліку питань для завдання типу «правда чи ні» для розділу «Історичні дати».

При обранні даних для побудови чергового питання випадковим чином обирається значення номеру запису у переліку даних про історичні дати таким чином, щоб він не повторювався з попередніми, зчитується об’єкт з переліку історичних дат за отриманим номером, а номер додається до множини унікальних значень. Випадковим чином обирається тип набору даних для питання, чи буде сформований факт істиною. Якщо факт повинен буде сформований як помилковий, випадковим чином обирається інший об’єкт з переліку персоналій для отримання невідповідних даних. Наступним кроком випадковим чином обирається тип набору даних. Відповідно до отриманого значення формується набор «Чи правда, що в »+дата+«відбулася подія» або «Чи правда, що подія»+назва+ «відбулася у». З поточного об’єкту історичної дати зчитується або назва, або дата, відповідно до типу набору даних питання. Наступним кроком створюється об’єкт питання Question з сформованим набором даних та додається до переліку питань. Формування питання виконується стальки разів, скільки питань у завданні.

Для завдань типу «знайти відповідність» не формується перелік, і відразу у процесі відображення та обробки формується набор даних для завдання.

На рисунку 2.18 наведено блок-схему розробленого алгоритму формування переліку питань для завдання типу «знайти відповідність» для розділу «Персоналії».

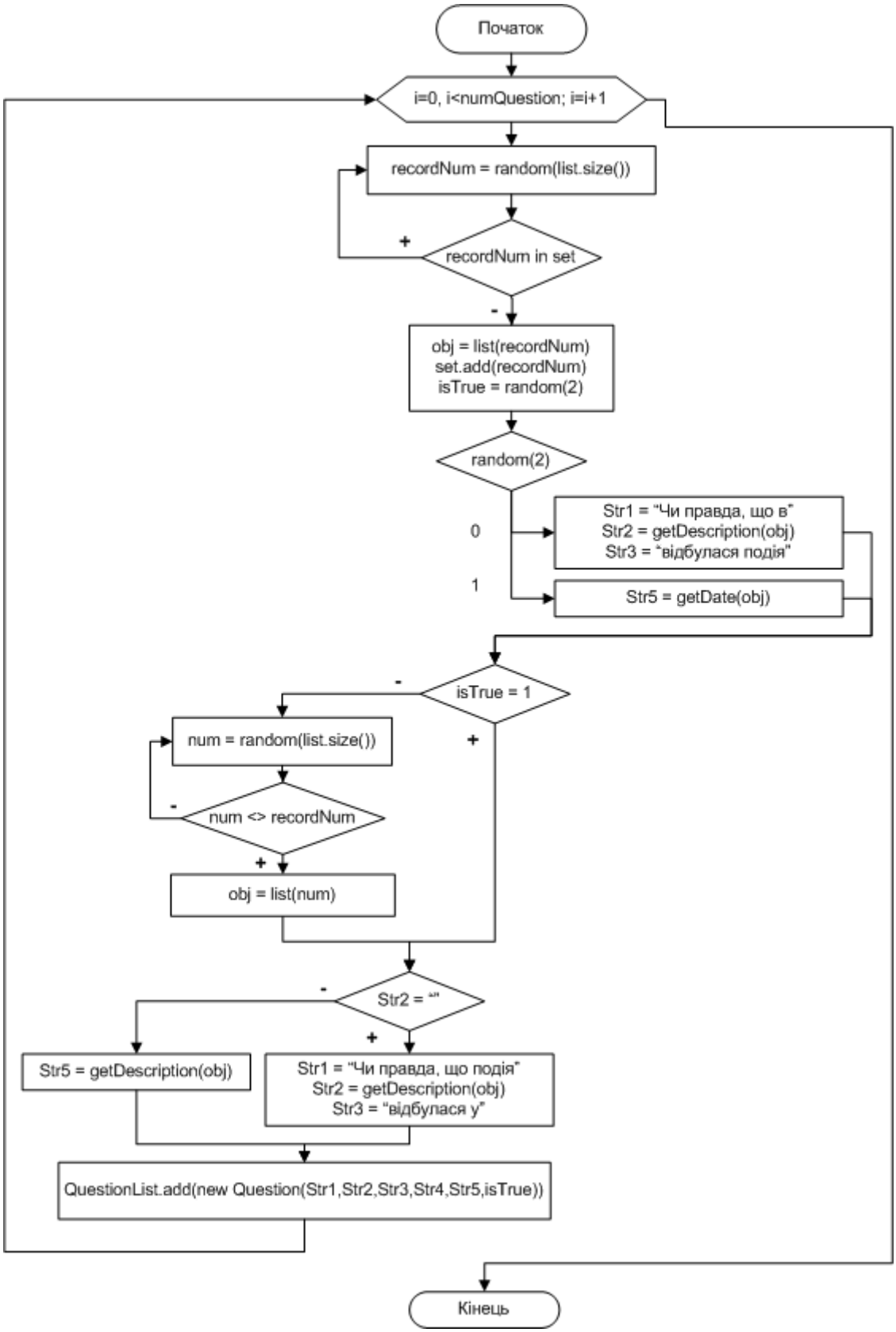


Рисунок 2.17 – Блок-схеми алгоритму побудови переліку питань завдання «правда чи ні» розділу «Історичні дати»

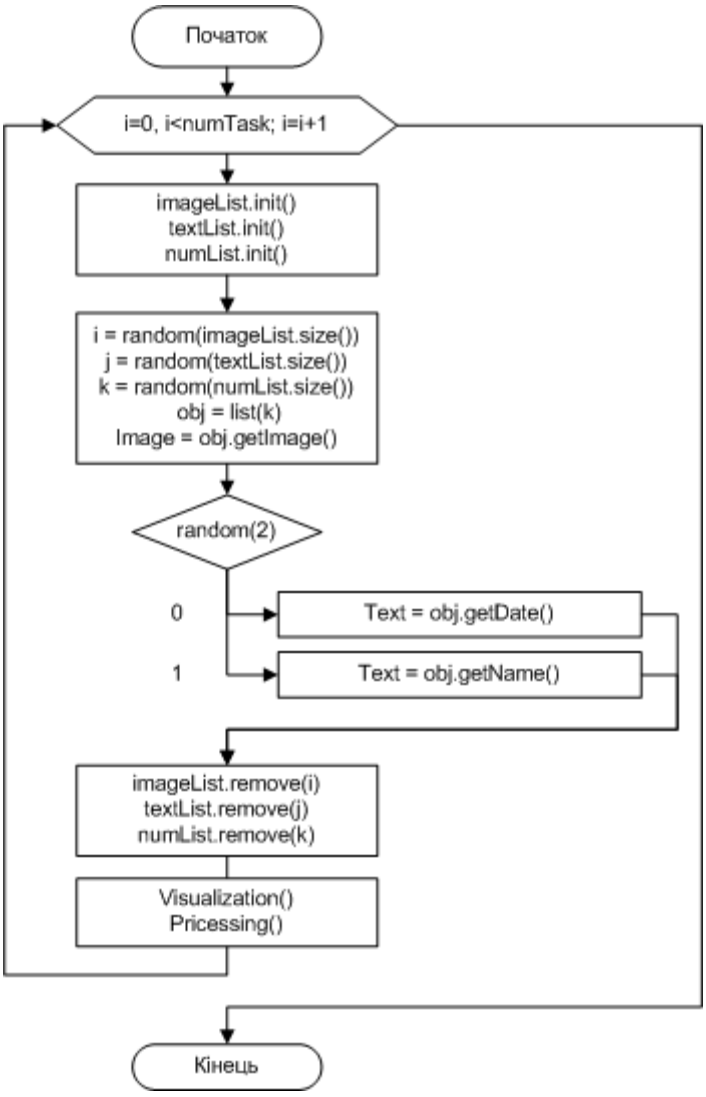


Рисунок 2.18 – Блок-схеми алгоритму побудови переліку питань завдання «знайти відповідність» розділу «Персоналії»

Спочатку формування завдання ініціюються переліки елементів керування, які будуть використовуватися у завданні (зображення та текстові мітки), а також перелік номерів об’єктів з переліку даних про персоналії. Випадковим чином обирається значення номеру запису у переліку даних про персоналії та зчитується об’єкт обраного номеру. Випадковим чином обирається елемент керування – зображення та елемент керування текстова мітка для розміщення у них даних одного й того об’єкту персоналій. У зображення заноситься шлях до портретного зображення, до текстової мітки дати життя, або ім’я персоналії – що саме обирається випадковим чином. Наступним кроком обрані елементи керування

видаляються з відповідних переліків, щоб унеможливити їх повторне використання, а номер запису об’єкту персоналії з переліку номерів. Після заповнення усіх елементів керування даними завдання відображується та обробляються відповіді користувача.

Блок-схеми розроблених алгоритмів формування переліку питань для завдання типу «знайти відповідність» розділів «Образотворче мистецтво» та «Архітектура» повністю аналогічні, тому наведемо загальну блок-схему на одному рисунку 2.19, та приведемо загальний опис роботи алгоритму.

Спочатку формування завдання ініціюються переліки елементів керування, які будуть використовуватися у завданні (зображення та текстові мітки), а також перелік номерів об’єктів з переліку даних про витвори мистецтва (архітектури). Випадковим чином обирається значення номеру запису у переліку даних про витвори мистецтва (архітектури) та зчитується об’єкт обраного номеру. Випадковим чином обирається елемент керування – зображення та елемент керування текстова мітка для розміщення у них даних одного й того об’єкту персоналій. У зображення заноситься шлях до портретного зображення, до текстової мітки його назва. Наступним кроком обрані елементи керування видаляються з відповідних переліків, щоб унеможливити їх повторне використання, а номер запису об’єкту персоналії з переліку номерів. Після заповнення усіх елементів керування даними завдання відображується та обробляються відповіді користувача.

На рисунку 2.20 наведено блок-схему розробленого алгоритму формування переліку питань для завдання типу «знайти відповідність» для розділу «Історичні дати».

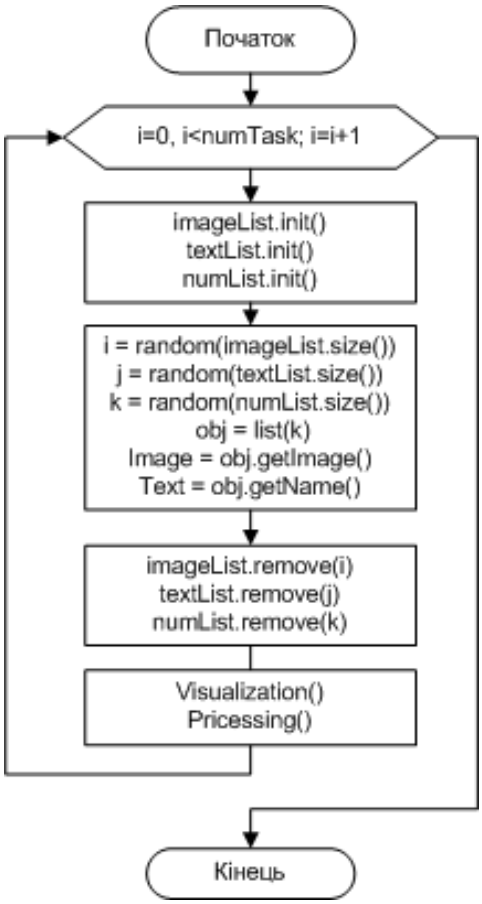


Рисунок 2.19 – Блок-схеми алгоритму побудови переліку питань завдання «знайти відповідність» розділів «Образотворче мистецтво» та «Архітектура»

Спочатку формування завдання ініціюються переліки елементів керування, які будуть використовуватися у завданні (зображення та текстові мітки), а також перелік номерів об’єктів з переліку даних про історичні дати. Випадковим чином обирається значення номеру запису у переліку даних про історичні дати та зчитується об’єкт обраного номеру. Випадковим чином обирається елемент керування – зображення та елемент керування текстова мітка для розміщення у них даних одного й того об’єкту персоналій. У зображення заноситься дата події або назва, у текстову відповідно навпаки, що саме обирається випадковим чином. Наступним кроком обрані елементи керування видаляються з відповідних переліків, щоб унеможливити їх повторне використання, а номер запису об’єкту персоналії з переліку номерів. Після заповнення усіх елементів керування даними завдання відображується та обробляються відповіді користувача.

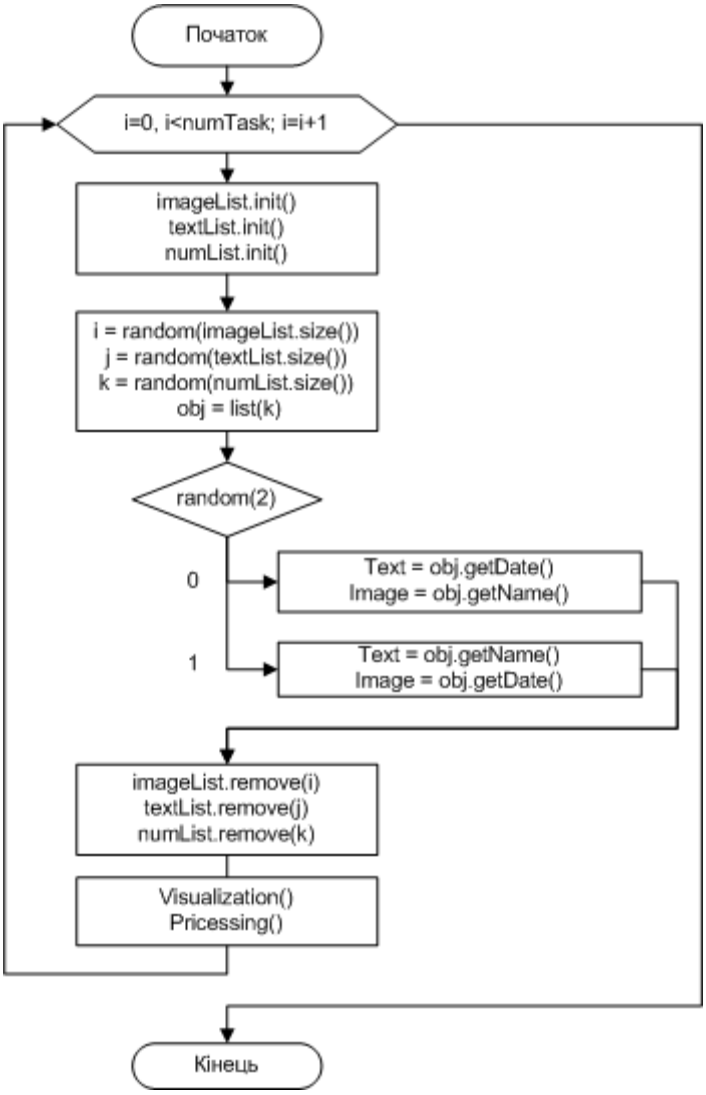


Рисунок 2.20 – Блок-схеми алгоритму побудови переліку питань завдання «знайти відповідність» розділу «Історичні дати»

2.3.3. Розрахунок характеристик якості виконання завдання

В інтерактивній системі, що розробляється, використовуються наступні засобі гейміфікації для викликання більшої зацікавленості користувачем процесом навчання:

- система балів, що дозволяє відстежувати прогрес на шляху просування до мети;
- прогрес, який надає інформацію користувачам на якому етапі навчання вони знаходяться,
- право на помилку, користувач має кілька спроб для вирішення задачі;
- різнотипні завдання, відрізняються складністю, тривалістю та іншими параметрами;
- завдання мають різний рівень складності, шляхом обмеження часу на їх виконання;
- винагороди, отримуються за певних умов виконання завдання;
- таблиця лідерів, що відображає досягнення користувачів системи.

Результати вивчення оцінюються за результатами виконання користувачем наданих за його вибором завдань. Оскільки система передбачає вивчення різних розділів, сумарний результат отриманих при навчанні балів повинен формуватися з кількості балів за кожним розділом.

Оскільки рівень знання не є постійним, може покращуватися при регулярних тренування та повторюваннях, або погіршуватися з часом без повторення та використання, оцінка повинна буду пов’язана з статистичними характеристиками виконання завдань.

Пропонується реалізувати в системі отримання оцінки рівня знань за кожним розділом за формулою:

$$\text{Оцінка за розділом} = \frac{\text{Кількість виконаних завдань за розділом}}{\text{Кількість успішно виконаних завдань}}$$

Підсумкову оцінку по всіх розділах відповідно:

$$\text{Оцінка за розділами } = \sum_{i=1}^{4} \frac{\text{Кількість виконаних завдань за розділом}}{\text{Кількість успішно виконаних завдань}}$$

Таким чином, оцінка буде тим більше, чим більше вірних виконаних завдань буде у користувача, а також оцінка з часом може як збільшуватися, так і зменшуватися.

Пропонується отриману оцінку переводити у вигляд «балів», цілих значень, для більш зручного уявлення для користувача наступним чином:

```
sumScore = round(perTestSuccess/perTestCount * 100)
         + round(hisTestSuccess/hisTestCount * 100)
         + round(artTestSuccess/artTestCount * 100)
         + round(archTestSuccess/archTestCount * 100),
```

де: sumScore – сумарна кількість балів за всіма розділами;

perTestCount – кількість виконаних завдань за розділом «Персоналії»;

perTestSuccess – кількість успішно виконаних завдань за розділом «Персоналії»;

hisTestCount – кількість виконаних завдань за розділом «Історичні дати»;

hisTestSuccess – кількість успішно виконаних завдань за розділом «Історичні дати»;

artTestCount – кількість виконаних завдань за розділом «Образотворче мистецтво»;

artTestSucces – кількість успішно виконаних завдань за розділом «Образотворче мистецтво»;

archTestCount – кількість виконаних завдань за розділом «Архітектура»;

archTestSuccess – кількість успішно виконаних завдань за розділом «Архітектура».

При такому підході існує ліміт кількості балів, що можна отримати, що може викликати невдоволення та розчарування у користувачів системи, тому пропонується система додаткових балів, які користувач буде отримувати відповідно до умов:

- певна кількість підряд вірно виконаних завдань;
- регулярність тренування;
- бонуси за передвчасне вірне виконання завдань з лімітованим часом, або не використання помилок, що використовуються.

Додаткові бали не є змінними, зберігаються постійно та накопичуються, однак у той же час система їх нарахування не надає можливості їх отримання без певного рівня засвоєння матеріалу.

Таким чином, стогова кількість балів буде розраховуватися за формулою:

$$\text{totalScore} = \text{sumScore} + \sum \text{addScore},$$

де: totalScore – стогова кількість балів користувача;

sumScore – сумарна кількість балів за всіма розділами;

addScore – додатковий бал.

Завдання типу «правильна відповідь» та «правда чи ні» містять по 10 запитань та враховується як виконаними успішно при отриманні більш чи рівно 80% вірних відповідей (тобто від 8 до 10 з 10х). Час на отримання кожної відповіді обмежено. При закінченні часу відповідь враховується як невірна та виконується перехід до наступного запитання.

Завдання типу «знайти відповідність» розділів «Персоналії» та «Історичні дати» містять по 3 питання для пошуку 9 відповідностей. Завдання та враховується як виконаними успішно при отриманні більш 65% вірних відповідей (тобто 2 або 3 з 3х). Час на отримання кожної відповіді обмежено. При закінченні часу відповідь враховується як невірна та виконується перехід до наступного запитання. Також лімітовано кількість спроб вказати відповідність для

запобігання підбору правильних відповідностей, можна зробити не більш ніж 6 невірних вказань.

Завдання типу «знайти відповідність» розділів «Образотворче мистецтво» та «Архітектура» містять по 6 питання для пошуку 4х відповідностей. Завдання та враховується як виконаними успішно при отриманні більш 70% вірних відповідей (тобто 3 або 4 з 4х). Час на отримання кожної відповіді обмежено. При закінченні часу відповідь враховується як невірна та виконується перехід до наступного запитання. Кількість спроб вказати відповідність також лімітовано, можна зробити не більш ніж 2 невірних вказання.

2.4. Проектування бази даних

Для проектування бази даних системи використаємо ER-діаграму. Схема «сутність-зв'язок» (також ERD або ER-діаграма) – це різновид блок-схеми, де показано, як різні «сутності» (люди, об'єкти, концепції і так далі) пов'язані між собою всередині системи. ER-діаграми найчастіше застосовуються для проектування і налагодження реляційних баз даних в сфері освіти, дослідження, розробки програмного забезпечення та інформаційних систем.

В результаті аналізу було обрано наступні сутності, інформація про які потребує збереження у базі даних:

– користувач;

– сесія користувача;

– історичні персони;

– історичні події;

– витвори мистецтва;

– об’єкти архітектури.

Розроблену ER-діаграму відношень між сутностями системи наведено на рисунку 2.21.

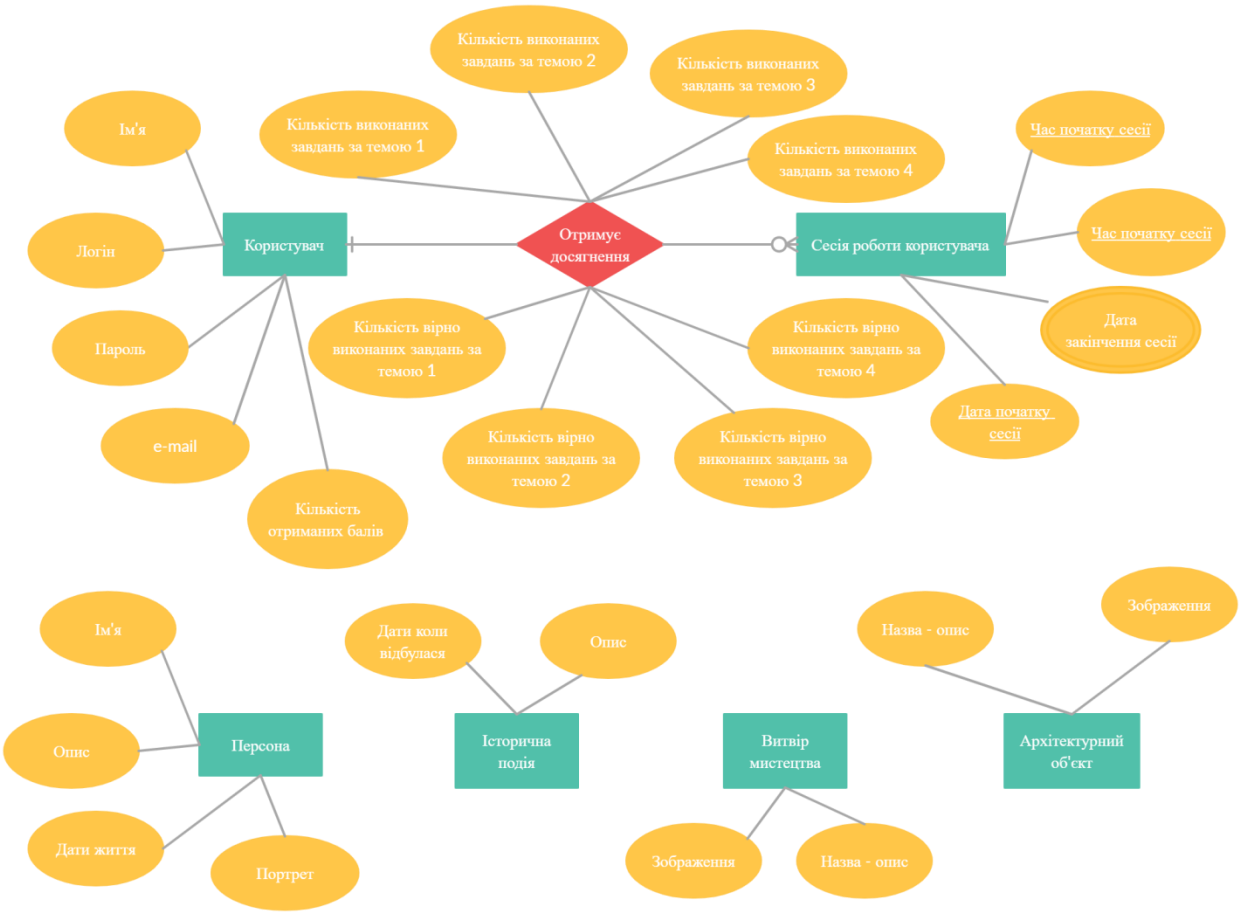


Рисунок 2.21 – ER-діаграма бази даних

Базу даних за назвою zno_schema було розроблено з використанням MySQL – вільної реляційної система управління базами даних та MySQL Workbench – уніфікованого візуального інструменту для архітекторів та розробників баз даних, що забезпечує моделювання даних, розробку SQL, надає комплексні інструменти адміністрування для конфігурації сервера, адміністрування користувачів та багато іншого. На рисунку 2.22 наведено структуру бази даних, що було розроблено.

База даних складається з наступних таблиць, що відповідають сутностям, визначеним ER-діаграмою.

– users (користувач);

– useractions (сеccія);

– historicaldates (історична подія);

– persons (історична персона);

– arts (витвір мистецтва);

– architectures (архітектурний об’єкт).

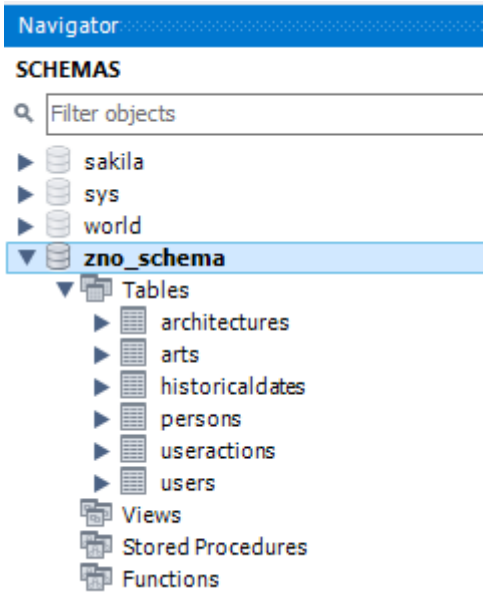


Рисунок 2.22 – ER-діаграма бази даних

На рисунку 2.23 наведено схему розробленої бази даних zno_schema.

Відповідно до вимог, користувач повинен мати можливість перегляду як загального рейтингу користувачів, так й своїх проміжних досягнень. Таким чином, властивості відношення «отримує досягнення», що зображено на рисунку 22 є необхідними як для сутності «користувач» так і для сутності «сесія користувача». Як властивості користувача вони відіграють роль загальної підсумкової інформації що до результатів навчання, як властивості сесії – роль проміжної інформації досягнень користувача, у рамках однієї сесії.

Таким чином відповідно до вимог було розроблено структури таблиць бази даних zno_schema, які наведено на рисунках 2.24 – 2.29.

Усі таблиці мають первинні ключі – числові значення автоматичної генерації та обмеження на необхідність бути заповненими для усіх полів таблиць.

У таблиці users поля login та email визначено як унікальні, тобто таблиця не повинна мати дублікатів цих значень.

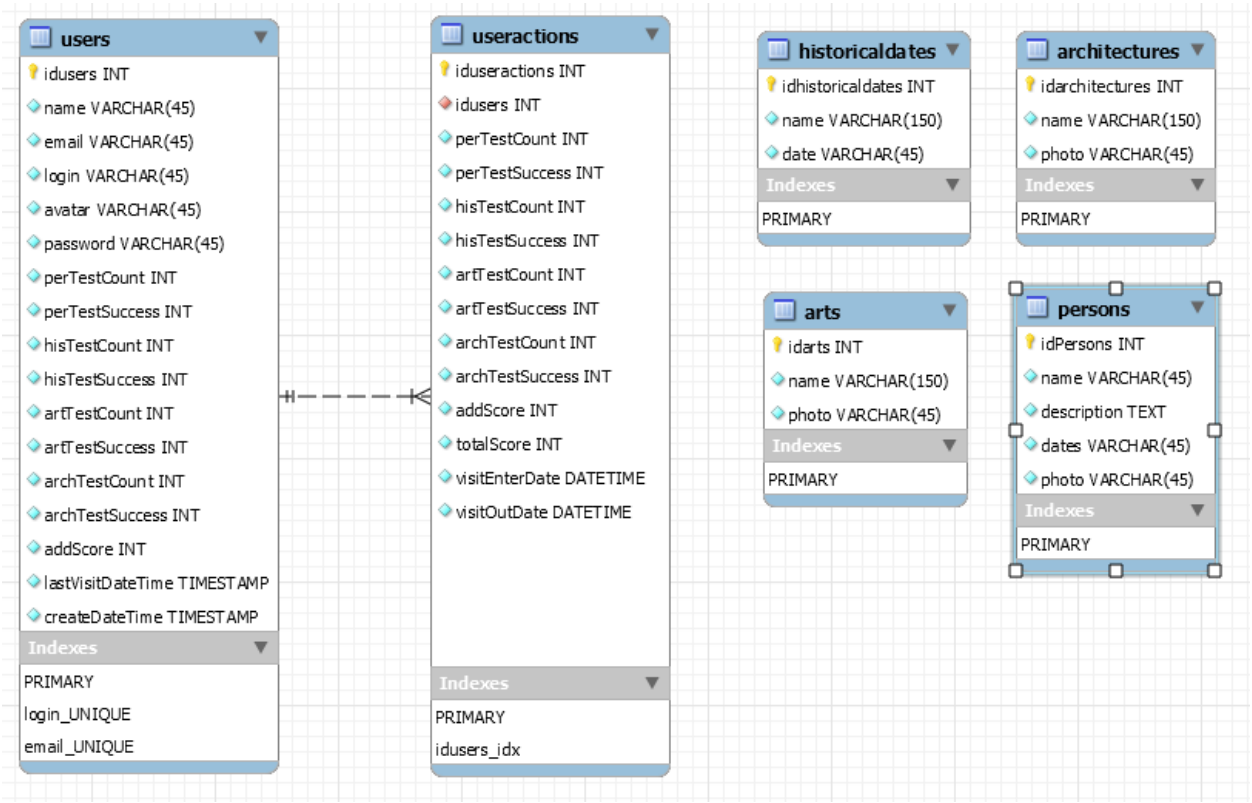


Рисунок 2.23 – Схема бази даних zno_schema

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
idusers	INT	☑	☑	☐	☐	☐	☐	☑	☐	
name	VARCHAR(45)	☐	☑	☐	☐	☐	☐	☐	☐	
email	VARCHAR(45)	☐	☑	☑	☐	☐	☐	☐	☐	
login	VARCHAR(45)	☐	☑	☑	☐	☐	☐	☐	☐	
avatar	VARCHAR(45)	☐	☑	☐	☐	☐	☐	☐	☐	
password	VARCHAR(45)	☐	☑	☐	☐	☐	☐	☐	☐	
perTestCount	INT	☐	☑	☐	☐	☐	☐	☐	☐	'0'
perTestSuccess	INT	☐	☑	☐	☐	☐	☐	☐	☐	'0'
hisTestCount	INT	☐	☑	☐	☐	☐	☐	☐	☐	'0'
hisTestSuccess	INT	☐	☑	☐	☐	☐	☐	☐	☐	'0'
artTestCount	INT	☐	☑	☐	☐	☐	☐	☐	☐	'0'
artTestSuccess	INT	☐	☑	☐	☐	☐	☐	☐	☐	'0'
archTestCount	INT	☐	☑	☐	☐	☐	☐	☐	☐	'0'
archTestSuccess	INT	☐	☑	☐	☐	☐	☐	☐	☐	'0'
addScore	INT	☐	☑	☐	☐	☐	☐	☐	☐	'0'
lastVisitDateTime	TIMESTAMP	☐	☑	☐	☐	☐	☐	☐	☐	'0000-00-00 00:00:00'
createDateTime	TIMESTAMP	☐	☑	☐	☐	☐	☐	☐	☐	'0000-00-00 00:00:00'
		☐	☐	☐	☐	☐	☐	☐	☐	

Рисунок 2.24 – Структура таблиці «users»

У таблицю useractions додано вторинний ключ – ідентифікатор користувача idusers, який ідентифікує приналежність проміжних даних до конкретного користувача та відповідно до побудови таблиць при відношенні один-до-багатьох.

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
iduseractions	INT	☑	☑	☐	☐	☐	☐	☑	☐	
idusers	INT	☐	☑	☐	☐	☐	☐	☐	☐	
perTestCount	INT	☐	☑	☐	☐	☐	☐	☐	☐	'0'
perTestSuccess	INT	☐	☑	☐	☐	☐	☐	☐	☐	'0'
hisTestCount	INT	☐	☑	☐	☐	☐	☐	☐	☐	'0'
hisTestSuccess	INT	☐	☑	☐	☐	☐	☐	☐	☐	'0'
artTestCount	INT	☐	☑	☐	☐	☐	☐	☐	☐	'0'
artTestSuccess	INT	☐	☑	☐	☐	☐	☐	☐	☐	'0'
archTestCount	INT	☐	☑	☐	☐	☐	☐	☐	☐	'0'
archTestSuccess	INT	☐	☑	☐	☐	☐	☐	☐	☐	'0'
addScore	INT	☐	☑	☐	☐	☐	☐	☐	☐	'0'
totalScore	INT	☐	☑	☐	☐	☐	☐	☐	☐	'0'
visitEnterDate	DATETIME	☐	☑	☐	☐	☐	☐	☐	☐	
visitOutDate	DATETIME	☐	☑	☐	☐	☐	☐	☐	☐	
		☐	☐	☐	☐	☐	☐	☐	☐	

Рисунок 2.25 – Структура таблиці «useractions»

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
idPersons	INT	☑	☑	☐	☐	☐	☐	☑	☐	
name	VARCHAR(45)	☐	☑	☐	☐	☐	☐	☐	☐	
description	TEXT	☐	☑	☐	☐	☐	☐	☐	☐	
dates	VARCHAR(45)	☐	☑	☐	☐	☐	☐	☐	☐	
photo	VARCHAR(45)	☐	☑	☐	☐	☐	☐	☐	☐	
		☐	☐	☐	☐	☐	☐	☐	☐	

Рисунок 2.26 – Структура таблиці «persons»

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
idhistoricaldates	INT	☑	☑	☐	☐	☐	☐	☑	☐	
name	VARCHAR(150)	☐	☑	☐	☐	☐	☐	☐	☐	
date	VARCHAR(45)	☐	☑	☐	☐	☐	☐	☐	☐	
		☐	☐	☐	☐	☐	☐	☐	☐	

Рисунок 2.27 – Структура таблиці «historicaldates»

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
idarts	INT	☑	☑	☐	☐	☐	☐	☑	☐	
name	VARCHAR(150)	☐	☑	☐	☐	☐	☐	☐	☐	
photo	VARCHAR(45)	☐	☑	☐	☐	☐	☐	☐	☐	
		☐	☐	☐	☐	☐	☐	☐	☐	

Рисунок 2.28 – Структура таблиці «arts»

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
idarchitectures	INT	☑	☑	☐	☐	☐	☐	☑	☐	
name	VARCHAR(150)	☐	☑	☐	☐	☐	☐	☐	☐	
photo	VARCHAR(45)	☐	☑	☐	☐	☐	☐	☐	☐	
		☐	☐	☐	☐	☐	☐	☐	☐	

Рисунок 2.29 – Структура таблиці «architectures»

2.5. Розробка інтерфейсу системи

Інтерфейс системи розроблено мовою розробки користувацьких інтерфейсів FXML з використанням інструменту розробки SceneBuilder, що забезпечує візуальне середовище макетування, дозволяє швидко розробляти користувальницькі інтерфейси та дозволяє автоматично створювати код FXML для макету безпосередньо під час створення макету інтерфейсу.

Розглянемо розроблені макети форм вікон програмної системи.

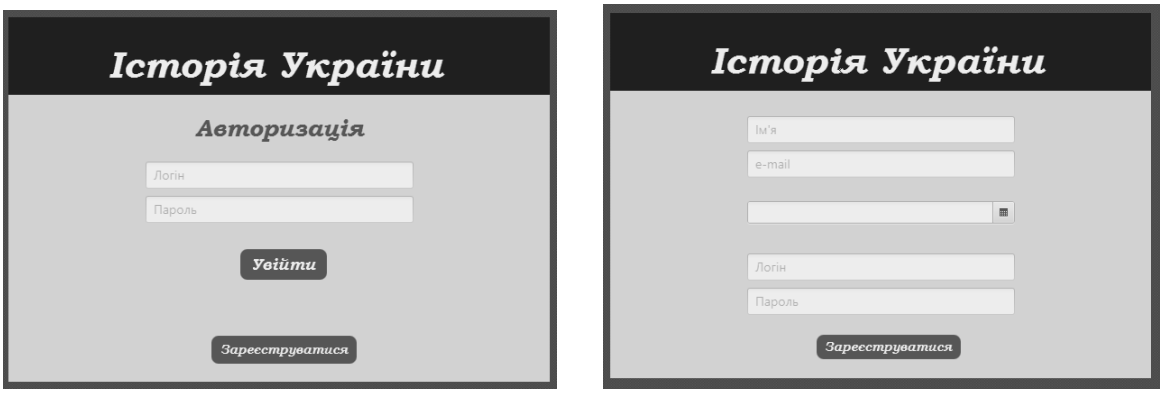
Вікно авторизації. Після запуску додатку відображається форма авторизації, макет форми наведено на рис. 2.30а). Вікно оснащено наступними елементами керування:

– поле для введення «Логін» – потребує вводу логіну існуючого користувача для авторизації;

– поле для введення «Пароль» – потребує вводу паролю існуючого користувача для авторизації;

– кнопка «Увійти» – виконує перевірку наданих користувачем даних на наявність у БД, при відсутності даних виконується анімація елементів керування;

– кнопка «Реєстрація» – викликає вікно реєстрації користувача (див. рис. 2.30).



а) б)

Рисунок 2.30 – Макети форм входу до системи (а – авторизація користувача, б – реєстрація користувача)

Вікно реєстрації призначено для отримання даних нового користувача та занесення отриманих даних до БД, макет форми наведено на рис. 2.30 б. Вікно оснащено наступними елементами керування:

– поле для введення «Ім’я» – потребує введення даних: імені користувача, за яким до нього будуть виконуватися усі звернення системи;

– поле для введення «E-mail» – потребує введення адреси електронної пошти користувача;

– поле для введення «Дата народження» – потребує введення дати народження користувача з використання елементу керування – календар;

– поле для введення «Логін» – потребує вводу логіну існуючого користувача для авторизації;

– поле для введення «Пароль» – потребує вводу паролю існуючого користувача для авторизації;

– кнопка «Зареєструватися» – виконує перевірку наданих користувачем даних на наявність у БД, при відсутності даних виконується реєстрація та занесення даних користувача до БД.

При успішному виконанні авторизації чи реєстрації відображається головне вікно програми, макет форми наведено на рисунку 2.31.

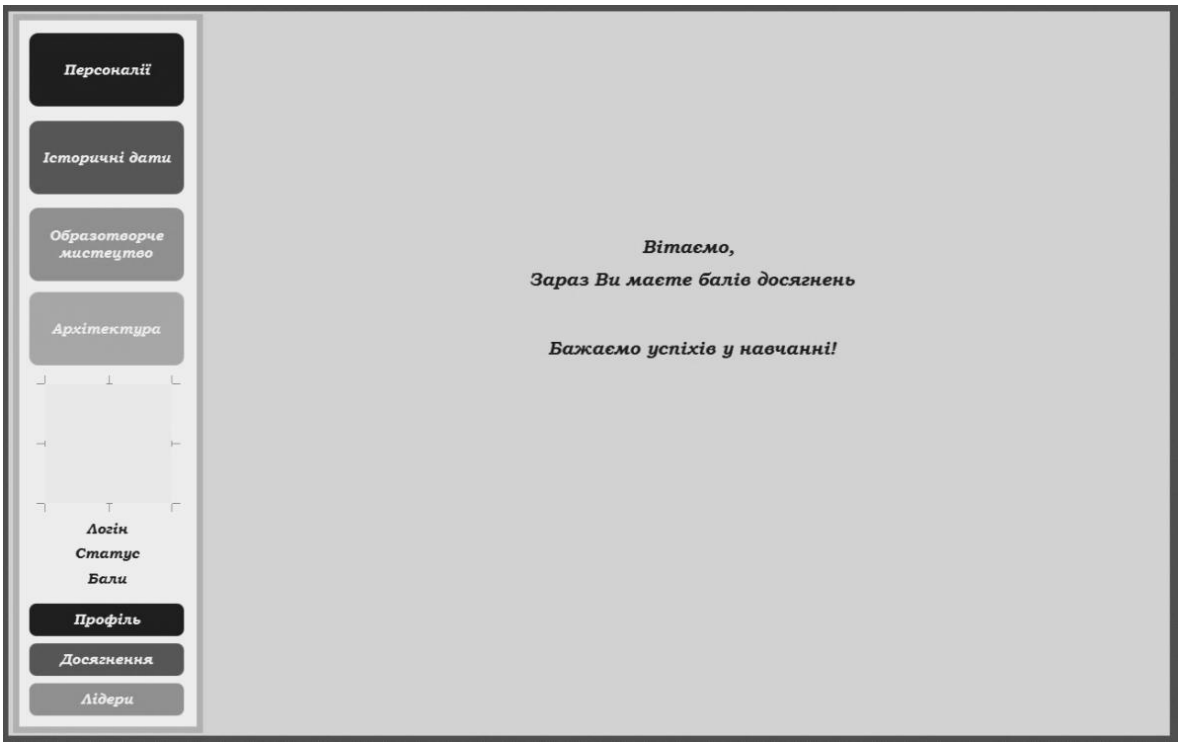


Рисунок 2.31 – Макет форми головного вікна програмної системи

Головне та усі інші вікна програми оснащено панеллю з кнопками, що розміщено ліворуч, для обрання тематики навчання, інших режимів роботи, а також відображає загальну поточну інформацію про користувача та його досягнення. Панель оснащено наступними елементами керування:

– кнопки «Персоналії», «Історичні дати», «Образотворче мистецтво», «Архітектура» – виконують перехід до вікна обрання типу завдання для навчання (рис. 2.32).

– кнопка «Профіль» – виконує перехід до вікна відображення та редагування даних профілю користувача (рис. 2.33);

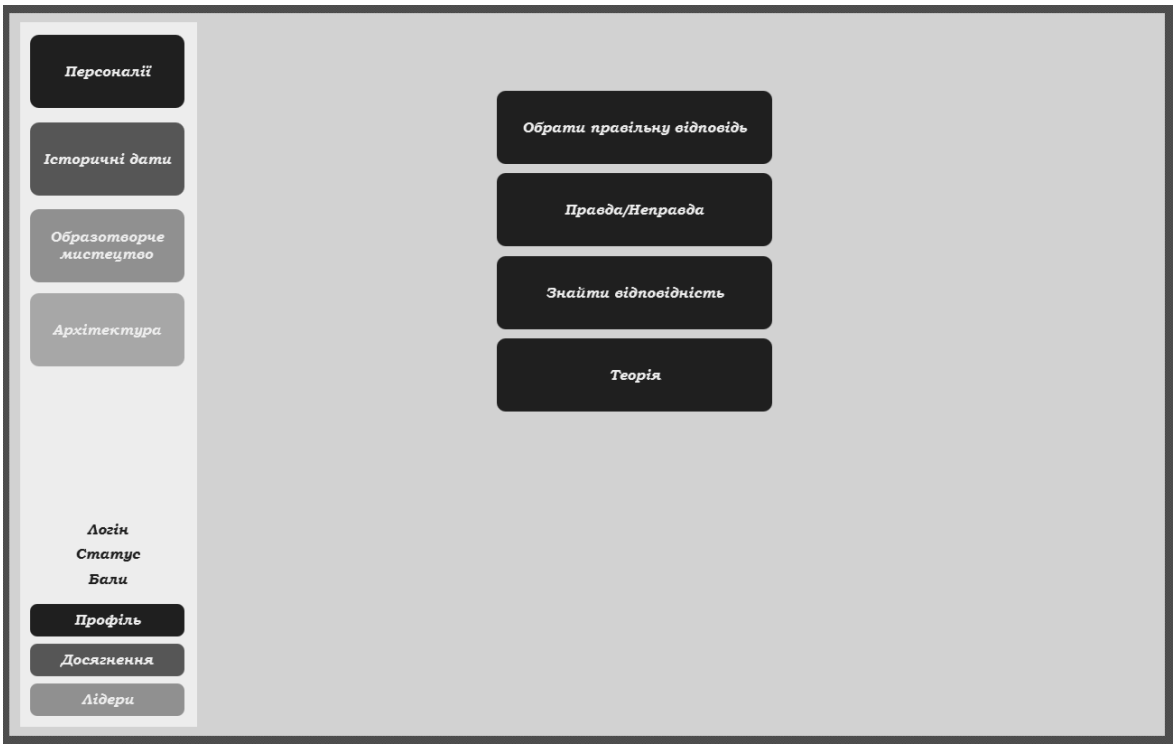


Рисунок 2.32 – Макет форми вікна вибору типу завдання

Рисунок 2.33 – Макет форми вікна перегляду та редагування даних користувача

– кнопка «Досягнення» – виконує перехід до вікна відображення стогових та проміжних досягнень (рис. 2.34);

– кнопка «Лідери» – виконує перехід до вікна відображення рейтингу досягнень усіх користувачів системи (2.35);

– зображення – відображає умовне зображення (аватар) користувача;

– поле «Логін» – відображає логін поточного користувача системи;

– поле «Статус» – умовний рівень досягнень користувача у навчанні;

– поле «Бали» – відображає стогову кількість отриманих користувачем балів за увесь період навчання.

Вікно обрання типу завдання для навчання, макет форми вікна наведено на рис. 2.32. Вікно оснащено наступними елементами керування :

– кнопка «Обрати правильну відповідь» – виконує перехід до вікна відображення завдань типу «обрання правильної відповіді з чотирьох наданих варіантів» (рис. 2.36);

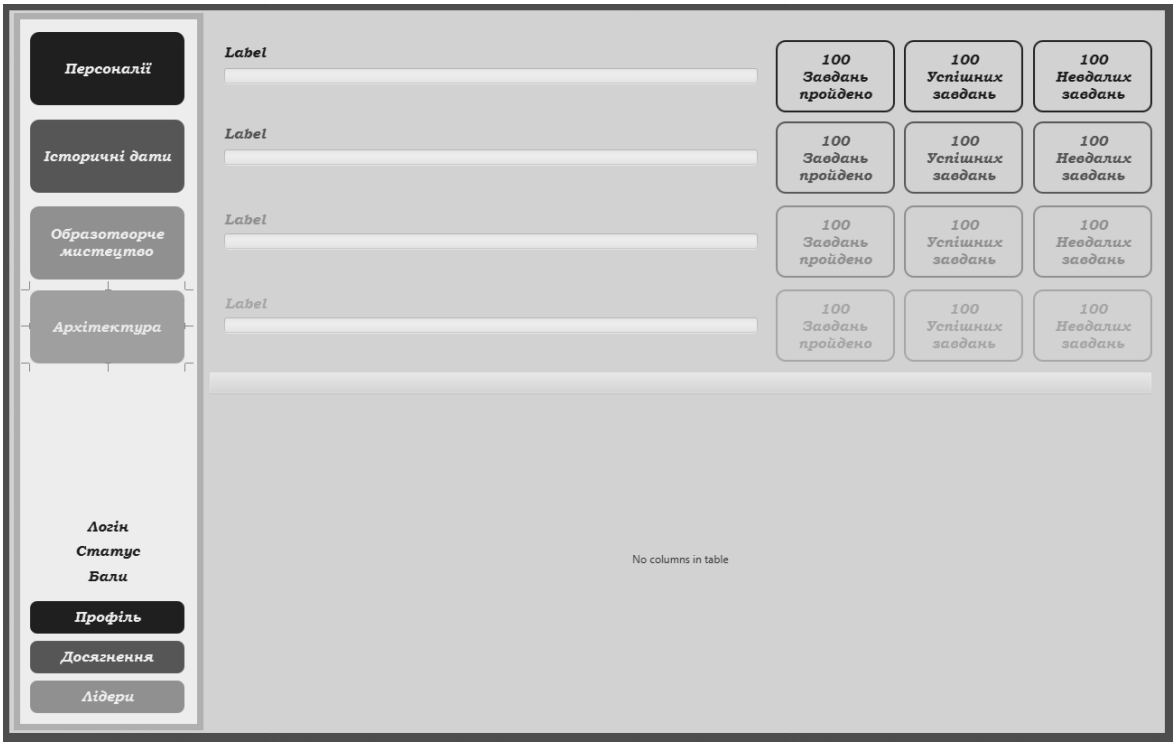


Рисунок 2.34 – Макет форми вікна перегляду досягнень користувача

Рисунок 2.35 – Макет форми вікна перегляду рейтингу користувачів системи

– кнопка «Правда/неправда» – виконує перехід до вікна відображення завдань типу «правда чи ні» (рис. 2.37);

– кнопка «Знайти відповідність» – виконує перехід до вікна відображення завдання типу «вказати відповідність» (рис. 2.38 – 2.39);

– кнопка «Теорія» – виконує перехід до вікна відображення теоретичної інформації з відповідної тематики.

Вікно відображення та редагування даних профілю користувача відривається натискання кнопки «Профіль», макет форми вікна наведено на рисунку 2.33. Вікно оснащено наступними елементами керування:

– зображення – відображає поточне умовне зображення (аватар) користувача;

– кнопка «Обрати» - відкриває стандартне вікно обрання файлу, дозволяє обрати довільне зображення у якості аватару користувача;

– поле для введення «Ім’я» – надає можливості введення нового імені користувача, за яким до нього будуть виконуватися усі звернення системи;

– поле для введення «E-mail» – надає можливості введення нової адреси електронної пошти користувача;

– поле для введення «Поточний пароль» – потребує вводу поточного паролю користувача за бажанням змінити існуючий для аутентифікації користувача;

– поле для введення «Новий пароль» – потребує вводу нового пароля за бажанням змінити існуючий.

Вікно відображення досягнень користувача відкривається натисканням кнопки «Досягнення» (рис. 2.34). Вікно оснащено наступними елементами керування:

– прогресбари для індикації пройденого прогресу навчання за кожною категорією;

– текстові мітки для відображення кількості отриманих завдань, успішно та невдало виконаних за кожною тематикою;

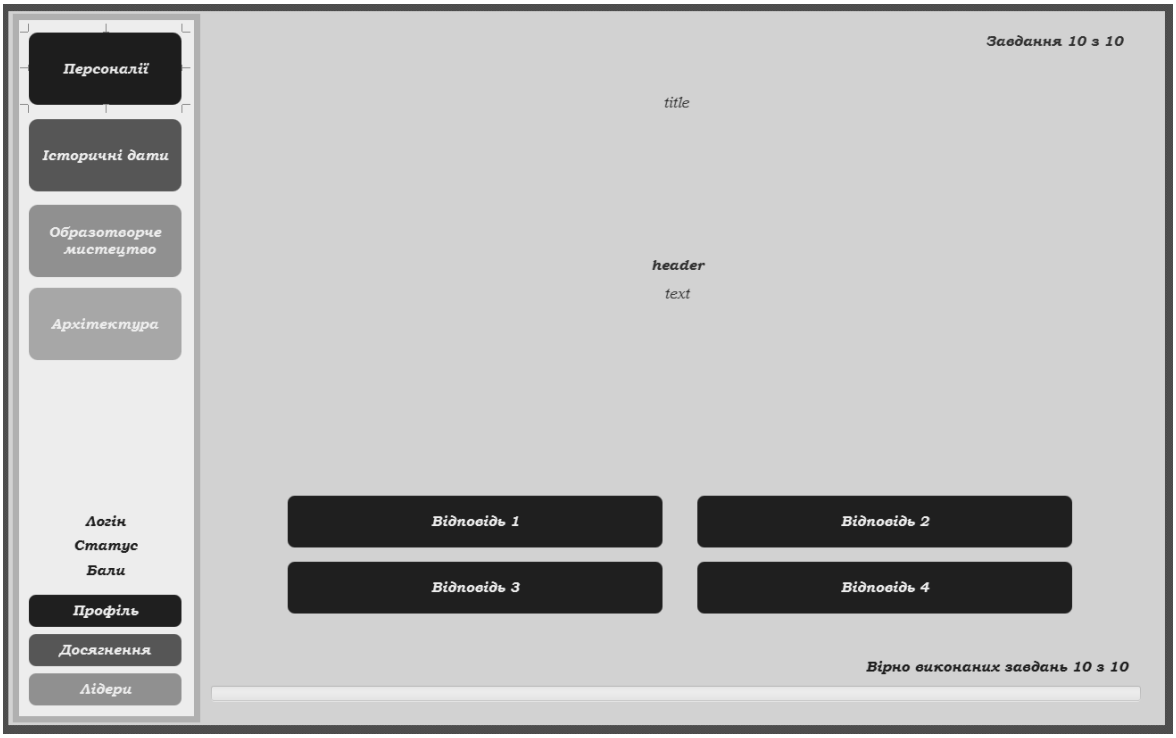


Рисунок 2.36 – Макет форми вікна завдання на обрання правильної відповіді

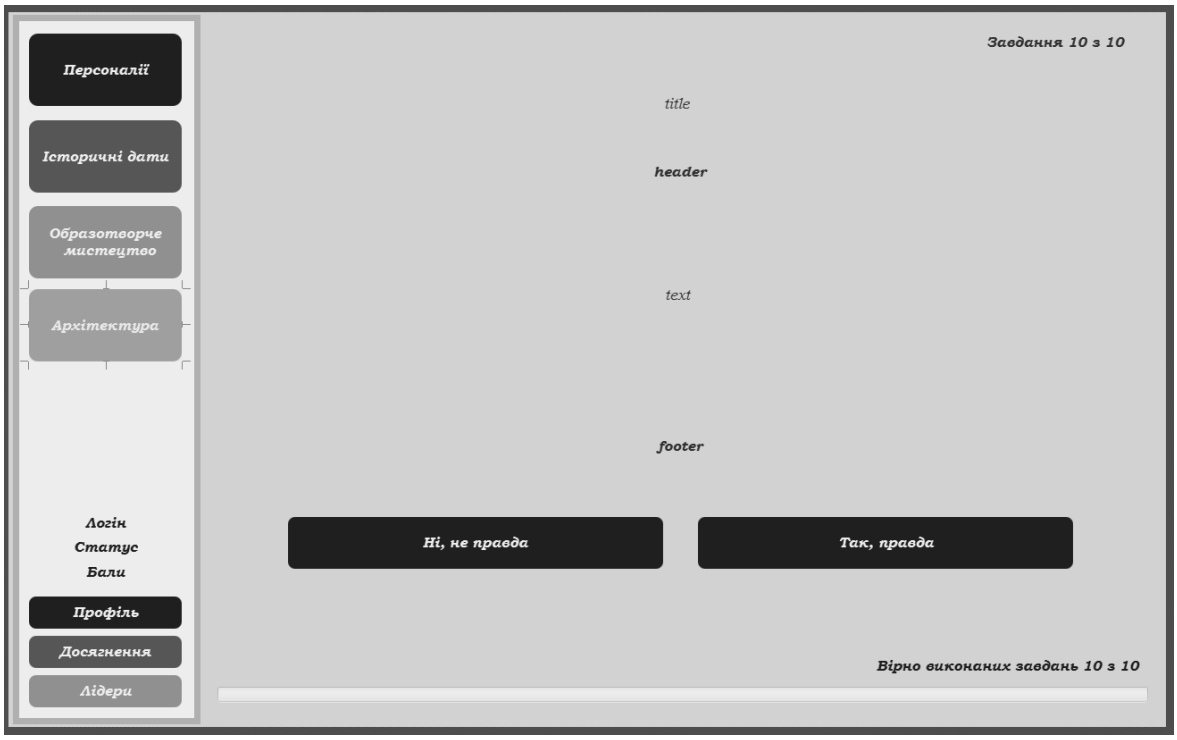


Рисунок 2.37 – Макет форми вікна завдання типу «правда чи ні»

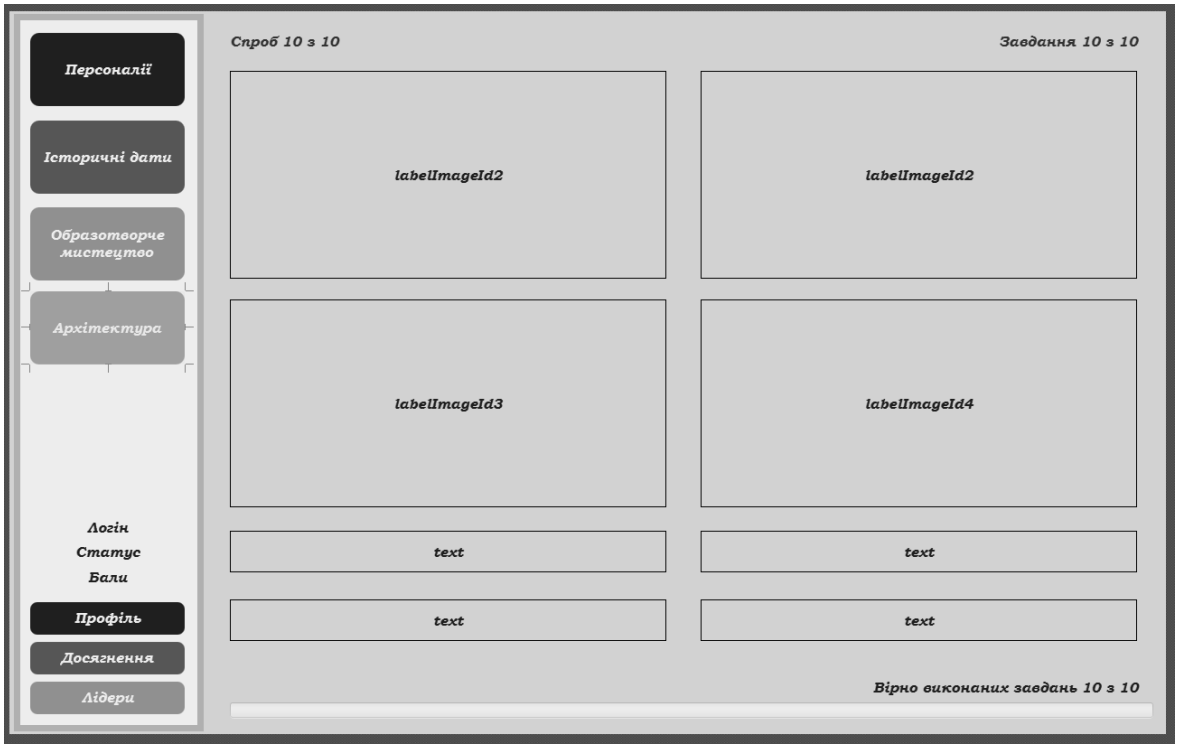


Рисунок 2.38 – Макет форми вікна завдання знаходження відповідностей

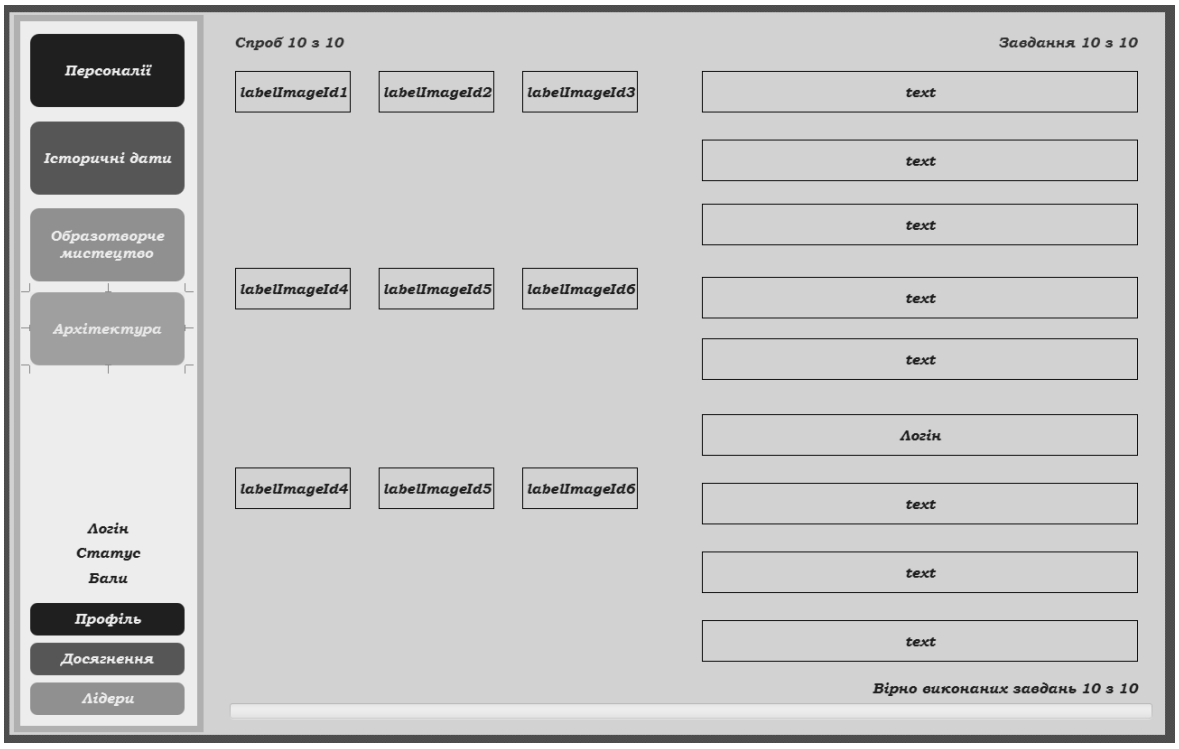


Рисунок 2.39 – Макет форми вікна завдання знаходження відповідностей

– таблицю, що відображає дані проміжних досягнень користувача.

Вікно відображення рейтингу досягнень усіх користувачів системи відкривається при натисканні кнопки «Лідери». Макет форми наведено на рисунку 2.35. Вікно оснащено елементом керування таблиця, що відображає дані досягнень користувачів системи.

Вікно відображення завдань типу «обрання правильної відповіді з чотирьох наданих варіантів» відкривається при натисканні кнопки «Обрати правильну відповідь». Макет форми вікна наведено на рисунку 2.36. Вікно оснащено наступними елементами керування:

– прогресбар – інформує користувача о кількості часу, що залишилося для виконання завдання;

– текстові мітки: загального питання, заголовку, основного тексту та футеру – для відображення текстових частин завдання;

– зображення – відображає графічну частину питання;

– чотири кнопки – надають чотири варіанти відповіді на завдання;

– текстові мітки – інформують користувача про кількість виконаних завдань та кількість завдань, що виконано вірно.

Вікно відображення завдань типу «правда чи ні» відкривається при натисканні кнопки «Правда/Неправда». Макет форми вікна наведено на рисунку 2.37. Вікно оснащено наступними елементами керування:

– прогресбар – інформує користувача о кількості часу, що залишилося для виконання завдання;

– текстові мітки: загального питання, заголовку, основного тексту та футеру – для відображення текстових частин завдання;

– зображення – відображає графічну частину питання;

– дві кнопки – надають варіанти відповідей на запитання так чи ні;

– текстові мітки – інформують користувача про кількість виконаних завдань та кількість завдань, що виконано вірно.

Вікно відображення завдань типу «знайти відповідності» відкривається при натисканні кнопки «Знайти відповідність». Форма вікна має два макети, які використовуються для завдань розділу «Персоналії», «Історичні дати» та «Образотворче мистецтво», «Архітектура». Макети форми вікна наведено на рисунках 2.38 та 2.39. Вікно оснащено наступними елементами керування:

– прогрес-бар – інформує користувача о кількості часу, що залишилося для виконання завдання;

– текстові мітки – текстова частина завдання, для завдань розділів «Персоналії», «Історичні дати» – дев’ять міток, для «Образотворче мистецтво», «Архітектура» – чотири;

– текстові мітки зображень – відображають графічну частину завдання, для завдань розділів «Персоналії», «Історичні дати» – дев’ять міток, для «Образотворче мистецтво», «Архітектура» – чотири;

– текстові мітки – інформують користувача про кількість виконаних завдань, кількість завдань, що виконано вірно, кількість спроб вказання відповідностей та кількість використаних спроб.

Вікно відображення теоретичного матеріалу відкривається при натисканні кнопки «Теорія» та відображає інформацію використовуючи елемент керування веб-уявлення, що інтерпретує файли електронних сторінок.

Вікно відображення результатів виконання завдань відкривається по закінченню виконання любого с завдань та інформує користувача про результати його виконання. Макети форми вікна наведено на рисунку 2.40. Вікно оснащено наступними елементами керування:

– зображення – надає графічну інтерпретацію результатів виконання завдань;

– текстові мітки – інформують користувача що до загальної кількості завдань та кількості вірно виконаних, текстова частина завдання, чи зараховане загальне завдання як виконане, зміни стогової кількості балів користувача та зміни його статусу.

Рисунок 2.40 – Макет форми вікна перегляду результатів виконання завдання

3. РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ СИСТЕМИ

3.1 Опис програмного проекту

Програмний продукт було виконано на ноутбуці Ноутбук HP 250 G3 з наступними характеристиками:

– процесор Intel Pentium N3530 (2.16 - 2.58 ГГц);

– ОЗП 4.00 ГБ (DDR3L 1600 МГц);

– графічний адаптер Intel HD Graphics (інтегрований);

– маніпулятор «миша».

Програмний продукт було розроблено під операційною системою Windows 10 та функціонує на персональних комп’ютерах з операційною системою Microsoft Windows 10 та вище.

Програмний продукт у середовищі розробки програмного забезпечення IntelliJ IDEA Community 2020.3, Runtime version 11.0.10+8-b1145.96 amd64, VM OpenJDK 64-bit Server VM by JetBrains з використанням платформи створення додатків з насиченим графічним інтерфейсом JavaFX. Програмний продукт реалізовано мовою Java та мовою розробки користувацьких інтерфейсів FXML.

Структуру проекту «HistoryStudyQuiz» наведено на рисунку 3.1.

Програмний продукт складається з таких файлів:

– achievements.fxml – файл розмітки вікна відображення досягнень користувача;

– AchievementsController.java – файл реалізації класу вікна відображення досягнень користувача;

– Architecture.java – файл реалізації класу Architecture, що уявляє сутність інформації про архітектурні об’єкти;

– Art.java – файл реалізації класу Art, що уявляє сутність інформації про об’єкти мистецтва;

– choice.fxml – файл розмітки вікна вибору типу завдань;

– ChoiceController.java – файл реалізації класу вікна вибору типу завдань;

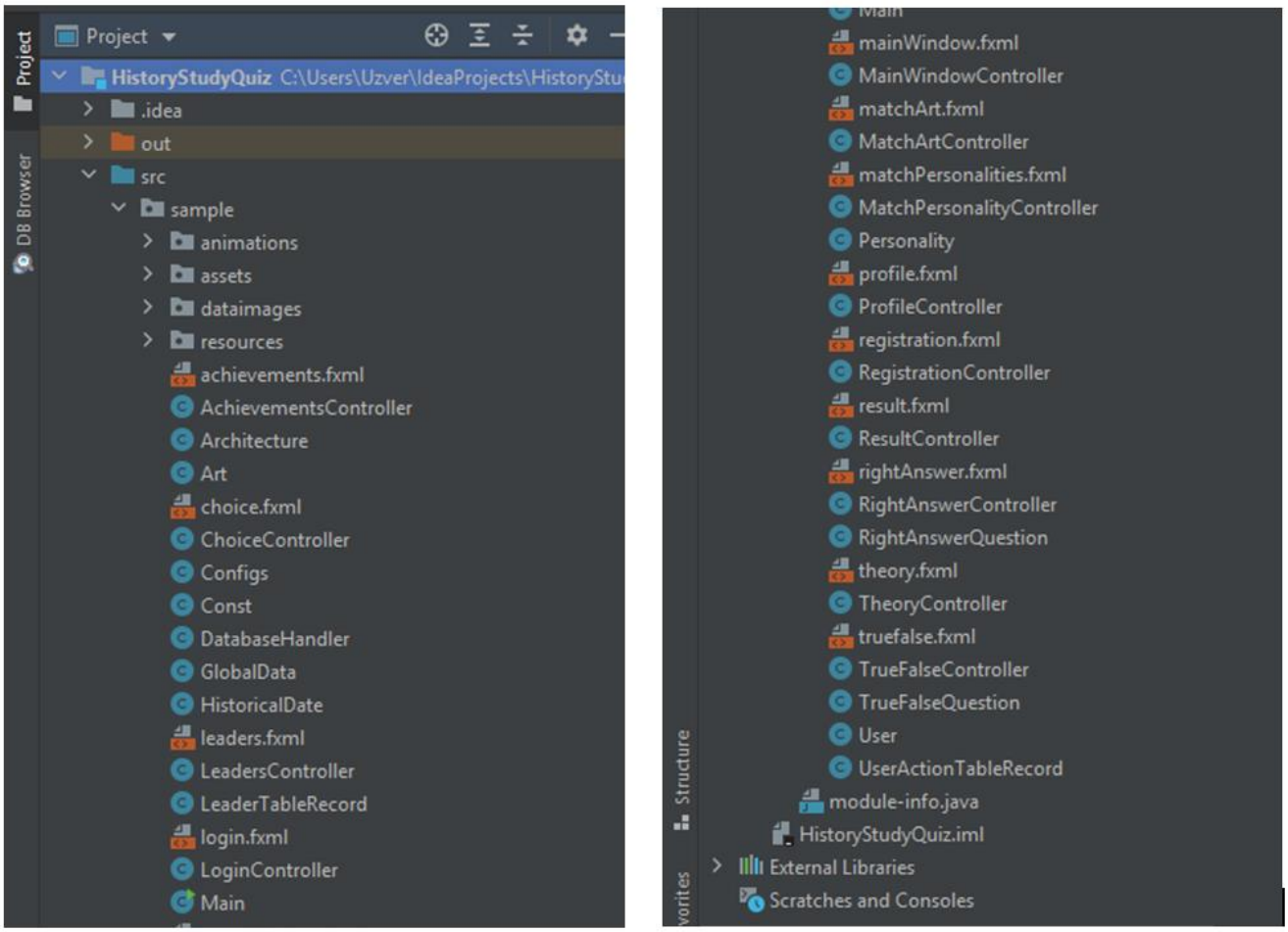


Рисунок 3.1 – Структура проекту «HistoryStudyQuiz»

– Configs.java – файл реалізації класу, що містить конфігураційні дані для підключення до бази даних;

– Const.java – файл реалізації класу, що містить загальносистемні константи;

– DatabaseHandler.java – файл реалізації класу DatabaseHandler, який виконує підключення та запити до базі динах;

– GlobalData.java – файл реалізації класу GlobalData, що містить загальні глобальні дані системи;

– HistoricalDate.java – файл реалізації класу HistoricalDate, що уявляє сутність інформації про історичні події;

– leaders.fxml – файл розмітки вікна відображення рейтингових даних користувачів системи;

– LeadersController.java – файл реалізації класу вікна відображення рейтингових даних користувачів системи;

– LeaderTableRecord.java – файл реалізації класу LeaderTableRecord, що уявляє строку у таблиці рейтингу користувачів системи;

– login.fxml – файл розмітки вікна авторизації в системі;

– LoginController.java – файл реалізації класу вікна відображення вікна авторизації в системі;

– Main.java – файл реалізації класу Main, що містить точку входу у програмну систему;

– mainWindow.fxml – файл розмітки головного вікна системи;

– MainWindowController.java – файл реалізації класу головного вікна системи;

– matchArt.fxml – файл розмітки вікна завдання пошуку відповідностей для вивчення об’єктів архітектури та мистецтва;

– MatchArtController.java – файл реалізації класу вікна завдання пошуку відповідностей для вивчення об’єктів архітектури та мистецтва;

– matchPersonality.fxml – файл розмітки вікна завдання пошуку відповідностей для вивчення історичних дат та персон;

– MatchPersonalityController.java – файл реалізації класу вікна завдання пошуку відповідностей для вивчення історичних дат та персон;

– Personality.java – файл реалізації класу Personality, що уявляє сутність інформації про історичну персону;

– profile.fxml – файл розмітки вікна відображення та редагування даних користувача;

– ProfileController.java – файл реалізації класу вікна відображення та редагування даних користувача

– registration.fxml – файл розмітки вікна реєстрації користувача;

– RegistrationController.java – файл реалізації класу вікна реєстрації користувача;

– result.fxml – файл розмітки вікна відображення результатів виконання завдання;

– ResultController.java – файл реалізації класу вікна відображення результатів виконання завдання;

– rightAnswer.fxml – файл розмітки вікна відображення завдання пошуку правільної відповіді на питання;

– RightAnswerController.java – файл реалізації класу вікна відображення завдання пошуку правільної відповіді на питання;

– RightAnswerQuestion – файл реалізації класу RightAnswerQuestion, який уявляє сутність об’єкту завдання пошуку правільної відповіді на питання;

– theory.fxml – файл розмітки вікна відображення теоретичного матеріалу за розділом;

– TheoryController.java – файл реалізації класу вікна відображення теоретичного матеріалу;

– trueFalse.fxml – файл розмітки вікна відображення завдання визначення коректності наведених даних;

– TrueFalseController.java – файл реалізації класу вікна відображення завдання визначення коректності наведених даних;

– TrueFalseQuestion.java – файл реалізації класу TrueFalseQuestion, який уявляє сутність завдання визначення коректності наведених даних;

– User.java – файл реалізації класу User, уявляє сутність об’єкту «користувач»;

– UserActionTableRecord.java – файл реалізації класу UserActionTableRecord, що уявляє строку у таблиці проміжних результатів навчання користувача системи;

– module-info.java містить опис модуля: ім'я, залежності, що експортуються пакети, сервіси, які споживаються та надаються;

– тека dataimages містить теки для зберігання зображень до завдань: architevture, art та personality відповідно.

– тека recources містить службові файли системи (стандартні та обрані користувачами зображення для профілю),

– тека theory містить файли та зображення для відображення теоретичного матеріалу.

3.2 Діаграма та опис класів

Діаграма класів визначає типи класів системи та різноманітні статичні зв'язки, що існують між ними. На діаграмах класів зображуються також атрибути класів, операції класів та обмеження, які накладаються на зв’язки між ними. Основними зв’язками між класами є наступні:

– асоціація – це відносини між двома класами, яка означає, що екземпляри одного класу відповідні до другого класу, тобто, асоціюються з ним;

– агрегація – це асоціація між двома класами, яка означає, що екземпляри (один або кілька) одного класу входять до складу другого класу, як частина – ціле);

– композиція – це асоціація між двома класами, яка накладає більш сильні обмеження, ніж агрегація: композиційно частина одного класу може входити тільки в одне ціле іншого, частина існує, тільки поки існує ціле і припиняє своє існування разом з цілим;

– залежність – це асоціація між двома класами, яка означає, що зміна специфікації класу-постачальника може вплинути на роботу залежного класу, але не навпаки.

Для забезпечення оптимальної структури взаємовідносин для функціонування та подальшої модернізації системи було розроблено множену класів, загальну діаграму яких наведено на рисунку 3.2. На рисунку 3.3 наведено частину діаграми класів, що відображає відносини між класами, які зберігають та маніпулюють даними. На рисунку 3.4 наведено частину діаграми класів, що відображає відносини між класами, які забезпечують інтерфейс системи.

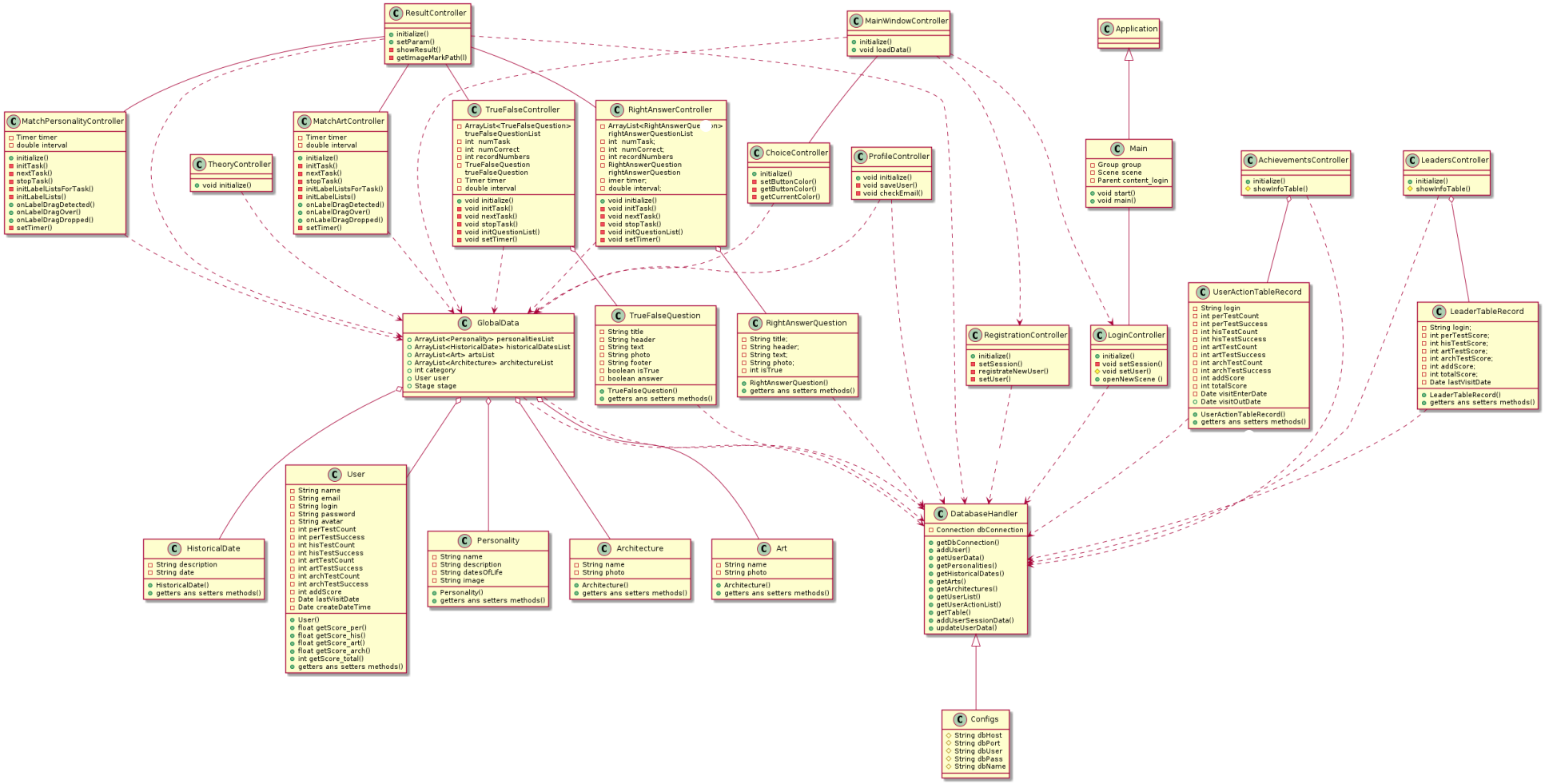


Рисунок 3.2 – Діаграма клас сів

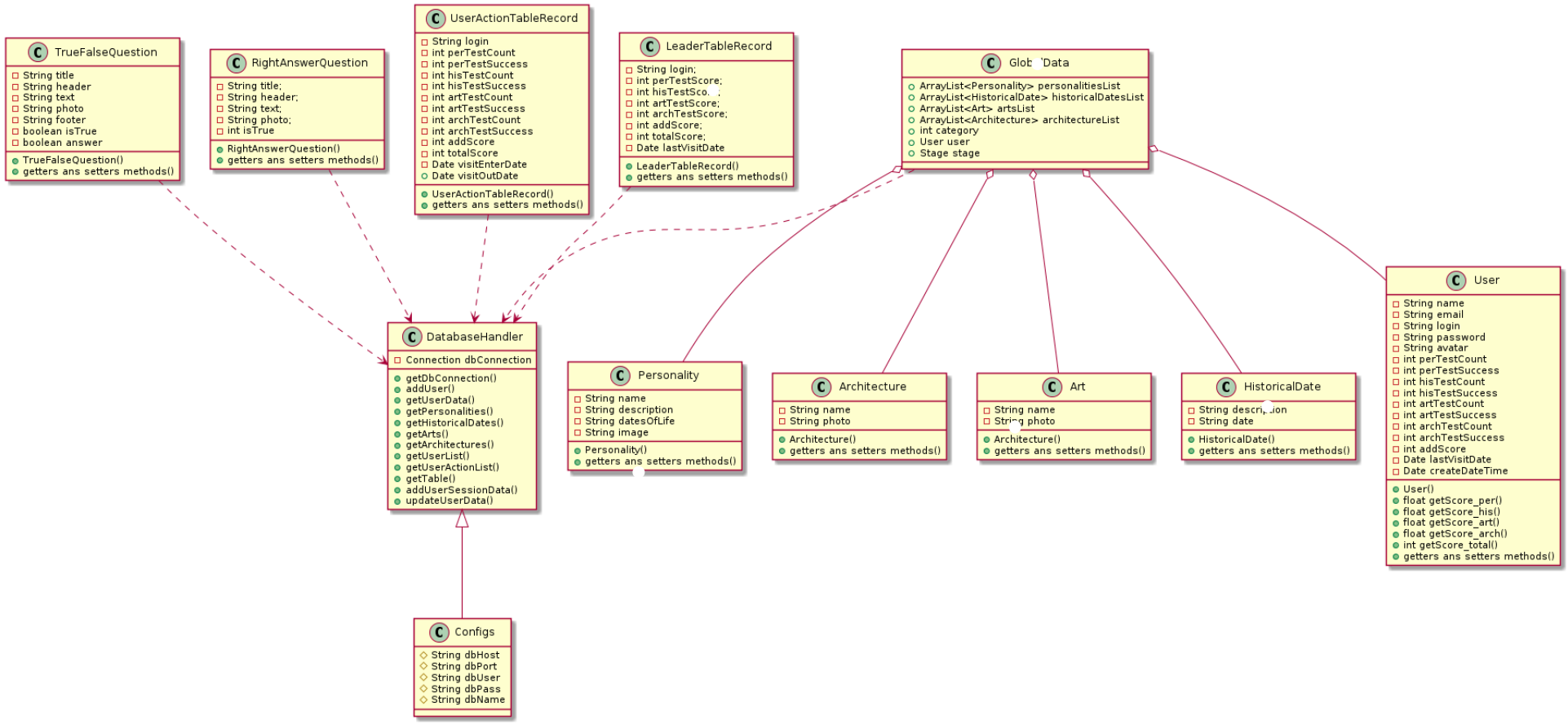


Рисунок 3.3 – Діаграма клас сів даних

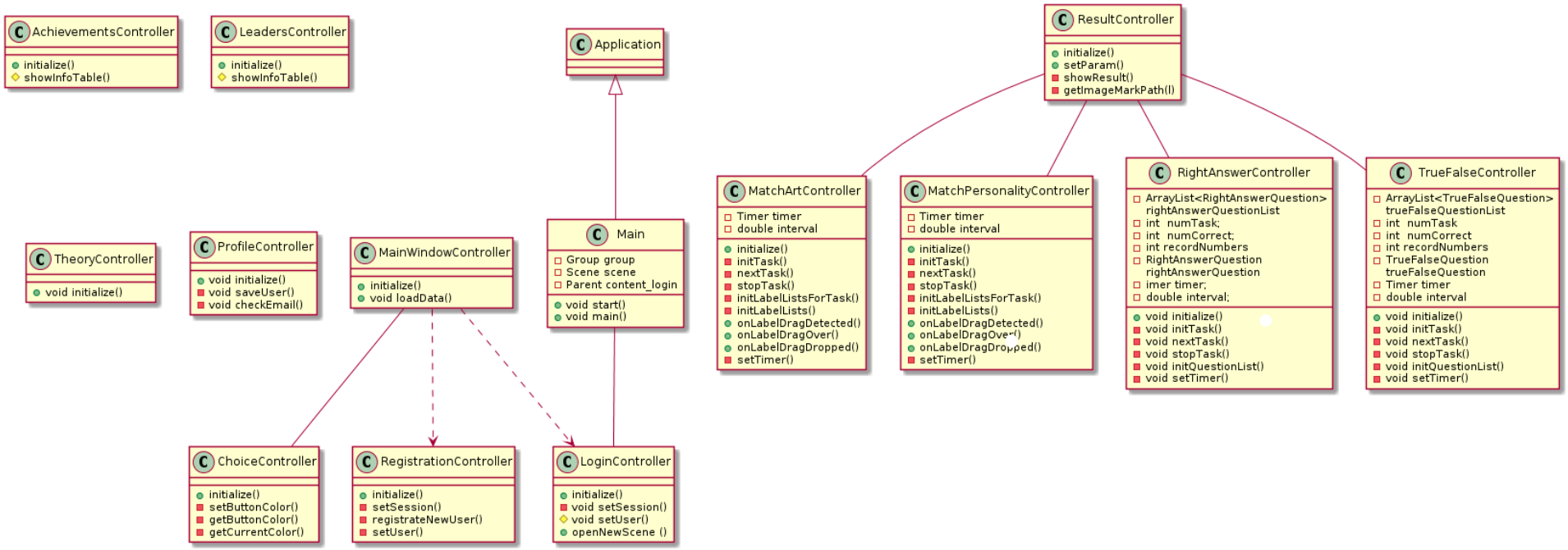


Рисунок 3.4 – Діаграма клас сів інтерфейсу

Наведемо опис розроблених класів.

Клас Main описує сутність «додаток», клас успадкований від класу javafx.application.Application. Методи класу:

– public void start(Stage primaryStage) - визначає графічний інтерфейс.

– public static void main(String [] args) – точка входу в програму, викликає метод launch() – запуск додатку.

Клас LoginController описує сутність «вікно авторизації». Методи класу:

– void initialize() – ініціалізація елементів керування вікна;

– private void setUser(String _strLogin, String _strPass) – зчитує дані користувача з БД, створює об’єкт «користувач» заповнюючи зчитаними даними, параметри: _strLogin – логін, _strPass – пароль користувача;

– public void openNewScene (String window) – відкриття нової сцени (головної форми), параметр window – шлях до FXML файлу опису нової сцени;

– private void setSession()- ініціалізація результатів навчання поточної сесії користувача

Клас RegistrationController описує сутність «вікно реєстрації». Методи класу:

– void initialize() – ініціалізація елементів керування вікна

– private void setSession()- ініціалізація результатів навчання поточної сесії користувача;

– private void setUser(String _strLogin, String _strPass) – зчитує дані користувача з БД, створює об’єкт «користувач» заповнюючи зчитаними даними, параметри: _strLogin – логін, _strPass – пароль користувача;

– public void openNewScene (String window) відкриття нової сцени (головної форми), параметр window – шлях до FXML файлу опису нової сцени.

Клас MainWindowController описує сутність «головне вікно програми». Методи класу:

– void initialize() – ініціалізація елементів керування вікна

– public void loadData() – завантаження даних з БД, які використовуються для побудови завдань, заповнює глобальні дані - переліки з інформацією про

персоналії (personalitiesList), історичні дати (historicalDatesList), об’єкти мистецтва (artsList) та архітектури (architectureList);

– public void openNewScene (String window) відкриття нової сцени (головної форми), параметр window – шлях до FXML файлу опису нової сцени.

Клас ChoiceController описує сутність «вікно обрання завдання». Методи класу:

– void initialize() – ініціалізація елементів керування вікна;

– void setButtonColor (String _color) – встановлення коліру кнопок, параметр _color – строкове уявлення коліру;

– private String getCurrentColor() – обрання кольору кнопок відповідно до поточної тематики;

– public void openNewScene (String window) відкриття нової сцени (головної форми), параметр window – шлях до FXML файлу опису нової сцени.

Клас ProfileController описує сутність «вікно перегляду та редагування профілю користувача». Методи класу:

– void initialize() – ініціалізація елементів керування вікна

– private updateUser() – збереження змінених даних профілю користувача у БД;

– public void openNewScene (String window) відкриття нової сцени (головної форми), параметр window – шлях до FXML файлу опису нової сцени.

Клас LeadersController описує сутність «вікно перегляду рейтингу користувачів системи». Методи класу:

– void initialize() – ініціалізація елементів керування вікна;

– private void showInfoTable() – створення об’єктів строк таблиці рейтингу (LeaderTableRecord), заповнення переліку строк таблиці рейтингу (userObservableList), заповнення таблиці даними з переліку;

– public void openNewScene (String window) відкриття нової сцени (головної форми), параметр window – шлях до FXML файлу опису нової сцени.

Клас AchievementsController описує сутність «вікно перегляду досягнень користувача». Методи класу:

– void initialize() – ініціалізація елементів керування вікна;

– private void showInfoTable()– створення об’єктів строк таблиці проміжних рейтингів користувача (UserActionTableRecord), заповнення переліку строк таблиці проміжних рейтингів (userObservableList), заповнення таблиці даними з переліку;

– public void openNewScene (String window) відкриття нової сцени (головної форми), параметр window – шлях до FXML файлу опису нової сцени.

Клас TheoryController описує сутність «вікно перегляду теоретичного матеріалу». Методи класу:

– void initialize() – ініціалізація елементів керування вікна;

– public void openNewScene (String window) відкриття нової сцени (головної форми), параметр window – шлях до FXML файлу опису нової сцени.

Клас TrueFalseController описує сутність «вікно завдання визначення істинності наданої інформації».

Властивості класу:

– private ArrayList<TrueFalseQuestion> trueFalseQuestionList – перелік завдань;

– private static Set<Integer> recordNumbers – хеш-множина, зберігає номери обраних для створення завдання запитання;

– private int numTask – номер поточного завдання;

– private int numCorrect – кількість правильно виконаних завдань;

– private TrueFalseQuestion trueFalseQuestion – дані поточного завдання;

– Timer timer – таймер відліку часу, що відведено на виконання завдання;

– double interval – поточне значення кількості часу що залишилося на виконання завдання.

Методи класу:

– void initialize() – ініціалізація елементів керування вікна;

– void setButtonColor (String _color) – встановлення кольору кнопок, параметр _color – строкове уявлення кольору;

– private String getCurrentColor() – обрання кольору кнопок відповідно до поточної тематики завдання;

– private void initQuestionList() – формування переліку завдань (trueFalseQuestionList), метод має чотири перевантажені реалізації для формування відповідного до тематики переліку завдань;

– private void nextTask() – відображення наступного завдання;

– private void stopTask() – завершення завдання та перехід до вікна результатів;

– public void setTimer() – встановлення параметрів та запуск таймеру відліку часу на виконання завдання.

Клас RightAnswerController описує сутність «вікно завдання визначення правільної відповіді на питання».

Властивості класу:

– private ArrayList<RightAnswerQuestion> rightAnswerQuestionList – перелік завдань;

– private static Set<Integer> recordNumbers = new HashSet<Integer>();

– private int numTask – номер поточного завдання;

– private int numCorrect – кількість правильно виконаних завдань;

– private RightAnswerQuestion rightAnswerQuestion - – дані поточного завдання;

– private ArrayList<Button> buttonList – перелік кнопок з відповідями;

– Timer timer – таймер відліку часу, що відведено на виконання завдання;

– double interval – поточне значення кількості часу що залишилося на виконання завдання.

Методи класу:

– void initialize() – ініціалізація елементів керування вікна;

– void setButtonColor (String _color) – встановлення кольору кнопок, параметр _color – строкове уявлення кольору;

– private String getCurrentColor() – обрання кольору кнопок відповідно до поточної тематики завдання;

– private void initQuestionList() – формування переліку завдань (rightAnswerQuestionList), метод має чотири перевантажені реалізації для формування відповідного до тематики переліку завдань;

– private void nextTask() – відображення наступного завдання;

– private void stopTask() – завершення завдання та перехід до вікна результатів;

– public void setTimer() – встановлення параметрів та запуск таймеру відліку часу на виконання завдання.

Клас MatchArtController описує сутність «вікно завдання знаходження відповідностей» для вивчення об’єктів мистецтва та архітектури.

Властивості класу:

– private ArrayList<Label> imageList – перелік міток, що містять зображення для завдання;

– private ArrayList<Label> labelList - перелік міток, що містять текст для завдання;

– private Map<Label,Label> mapLabel – мапа пар міток ключ-значення, які є вірними відповідностями;

– private ArrayList<Integer> numList – перелік номерів записів даних для формування завдання;

– private int numTask – номер поточного завдання;

– private int numCorrect – кількість правильно виконаних завдань;

– private int numTry – використана кількість спроб вказання відповідностей;

– Timer timer – таймер відліку часу, що відведено на виконання завдання;

– double interval – поточне значення кількості часу що залишилося на виконання завдання.

Методи класу:

– void initialize() – ініціалізація елементів керування вікна;

– void setButtonColor (String _color) – встановлення кольору кнопок, параметр _color – строкове уявлення кольору;

– private String getCurrentColor() – обрання кольору кнопок відповідно до поточної тематики завдання;

– private void initLabelListsForTask() – ініціалізація переліку елементів керування, які використовуються у завданні;

– private void initLabelLists()– призначення обробників подій елементам керування, які використовуються у завданні;

– private void onLabelDragDetected(MouseEvent mouseEvent) – обробник події виявлення жесту drag-and-drop, параметр mouseEvent – подія від маніпулятора «миша»;

– public void onLabelDragOver(DragEvent dragEvent) – обробник події виявлення наведення об’єкту, що перетягується на інший об’єкт, параметр dragEvent – подія від об’єкту, ще перетягується;

– public void onLabelDragDropped(DragEvent dragEvent) - обробник події завершення жесту drag-and-drop, параметр dragEvent – подія від об’єкту, ще перетягується;

– private void nextTask() – відображення наступного завдання;

– private void stopTask() – завершення завдання та перехід до вікна результатів;

– public void setTimer() – встановлення параметрів та запуск таймеру відліку часу на виконання завдання.

Клас MatchPersonalityController описує сутність «вікно завдання знаходження відповідностей» для вивчення історичних персоналій та історичних подій.

Властивості класу:

– private ArrayList<Label> imageList – перелік міток, що містять зображення для завдання;
private ArrayList<Label> labelList - перелік міток, що містять текст для завдання; private Map<Label,Label> mapLabel – мапа пар міток ключ-значення, які є вірними відповідностями;

– private ArrayList<Integer> numList – перелік номерів записів даних для формування завдання;

– private int numTask – номер поточного завдання;

– private int numCorrect – кількість правильно виконаних завдань;

– private int numTry – використана кількість спроб вказання відповідностей;

– Timer timer – таймер відліку часу, що відведено на виконання завдання;

– double interval – поточне значення кількості часу що залишилося на виконання завдання.

Методи класу:

– void initialize() – ініціалізація елементів керування вікна;

– void setButtonColor (String _color) – встановлення кольору кнопок, параметр _color – строкове уявлення кольору;

– private String getCurrentColor() – обрання кольору кнопок відповідно до поточної тематики завдання;

– private void initLabelListsForTask() – ініціалізація переліку елементів керування, які використовуються у завданні;

– private void initLabelLists()– призначення обробників подій елементам керування, які використовуються у завданні;

– private void onLabelDragDetected(MouseEvent mouseEvent) – обробник події виявлення жесту drag-and-drop, параметр mouseEvent – подія від маніпулятора «миша»;

– public void onLabelDragOver(DragEvent dragEvent) – обробник події виявлення наведення об’єкту, що перетягується на інший об’єкт, параметр dragEvent – подія від об’єкту, ще перетягується;

– public void onLabelDragDropped(DragEvent dragEvent) – обробник події завершення жесту drag-and-drop, параметр dragEvent – подія від об’єкту, ще перетягується;

– private void nextTask() – відображення наступного завдання;

– private void stopTask() – завершення завдання та перехід до вікна результатів;

– public void setTimer() – встановлення параметрів та запуск таймеру відліку часу на виконання завдання.

Клас DatabaseHandler описує «робота з БД».

Властивість Connection dbConnection – об’єкт з’єднання з базою даних.

Методи класу:

– public Connection getDbConnection() – встановлення з’єднання з базою даних;

– public int addUser(User user) – додання нового користувача до БД

– public ResultSet getUserData(String _strLogin, String _strPass) – отримання даних користувача з БД, параметри _strLogin – логін, _strPass – пароль користувача;

– public ResultSet getPersonalities()- зчитування з БД даних про історичні персоналії;

– public ResultSet getHistoricalDates()- зчитування з БД даних про історичні події;

– public ResultSet getArts()- зчитування з БД даних про об’єкти мистецтва;

– public ResultSet getArchitectures()- зчитування даних з БД про об’єкти архітектури;

– public ResultSet getUserList()- зчитування з БД даних про досягнення користувачів;

– public ResultSet getUserActionList(User user)- зчитування з БД даних про проміжні досягнення користувач, параметр user – об’єкт користувача;

– public ResultSet getTable(String _table) – зчитування з БД даних з таблиці, параметр _table – ім’я таблиці у БД;

– public void addUserSessionData(User user) – запис у БД поточних результатів навчання користувача, параметр user – об’єкт користувача;

– public void updateUserData(User user) – оновлення у БД даних користувача, параметр user – об’єкт користувача.

Клас Configs описує сутність «конфігурації БД»

Властивості:

– protected String dbHost = "localhost" – хост

– protected String dbPort = "3306" – порт;

– protected String dbUser = "root" – ім’я облікового запису адміністратора БД;

– protected String dbPass = "Bltynbabrfwbz_1234" – пароль облікового запису адміністратора БД;

– protected String dbName = "zno_schema" – ім’я БД.

Клас GlobalData описує сутність «глобальні дані системи».

Властивості класу:

– public static ArrayList<Personality> personalitiesList – перелік записів даних об історичних персоналіях;

– public static ArrayList<HistoricalDate> historicalDatesList – перелік записів даних об історичних подіях;

– public static ArrayList<Art> artsList – перелік записів даних об об’єктах мистецтва;

– public static ArrayList<Architecture> architectureList – перелік записів даних об об’єктах архітектури;

– public static int category – тематика завдань;

– public static User user – об’єкт користувача -системи;

– public static int curPerTestCount – поточна кількість завдань, що виконувалась з тематики «Персоналії»;

– public static int curPerTestSuccess - поточна кількість завдань, що вірно виконано з тематики «Персоналії»;

– public static int curHisTestCount– поточна кількість завдань, що виконувалась з тематики «Історичні дати»;

– public static int curHisTestSuccess - поточна кількість завдань, що вірно виконано з тематики «Історичні дати»;

– public static int curArtTestCount– поточна кількість завдань, що виконувалась з тематики «Мистецтво»;

– public static int curArtTestSuccess - поточна кількість завдань, що вірно виконано з тематики «Мистецтво»;

– public static int curArchTestCount - поточна кількість завдань, що виконувалась з тематики «Архітектура»;

– public static int curArchTestSuccess - поточна кількість завдань, що вірно виконано з тематики «Архітектура»;

– public static int curAddScore – кількість додаткових балів;

– public static Date enterDateTime – час початку сесії;

– public static Stage stage – головна сцена (форма) додатку.

Клас LeaderTableRecord описує сутність «строка таблиці рейтенга»

Властивості класу:

– private String login – логін користувача;

– private int perTestScore – кількість балів за виконання завдань з тематики «Персоналії»;

– private int hisTestScore - кількість балів за виконання завдань з тематики «Історичні дати»;

– private int artTestScore - кількість балів за виконання завдань з тематики «Мистецтво»;

– private int archTestScore - кількість балів за виконання завдань з тематики «Архітектура»;

– private int addScore – кількість додаткових балів;

– private int totalScore – загальна кількість балів;

– private Date lastVisitDate – дата та час останнього відвідування користувачем системи.

Методи класу:

– public LeaderTableRecord(String login, int perTestScore, int hisTestScore, int artTestScore, int archTestScore, int addScore, Date lastVisitDate) – конструктор класу;

– гетери та сетери - методи доступу до приватних властивостей класу.

Клас UserActionTableRecord описує сутність «строка таблиці досягнень користувача».

Властивості класу:

– private String login – логін користувача;

– private int perTestCount – кількість завдань, що виконувалась користувачем з тематики «Персоналії»;

– private int perTestSuccess – кількість завдань, що вірно виконано користувачем з тематики «Персоналії»;

– private int hisTestCount – кількість завдань, що виконувалась користувачем з тематики «Історичні дати»;

– private int hisTestSuccess – кількість завдань, що вірно виконано користувачем з тематики «Сторічні дати»;

– private int artTestCount – кількість завдань, що виконувалась користувачем з тематики «Сторічні дати»;

– private int artTestSuccess – кількість завдань, що вірно виконано користувачем з тематики «Сторічні дати»;

– private int archTestCount – кількість завдань, що виконувалась користувачем з тематики «Сторічні дати»;

– private int archTestSuccess - кількість завдань, що вірно виконано користувачем з тематики «Сторічні дати»;

– private int addScore – кількість додаткових балів;

– private int totalScore – загальна кількість балів;

– private Date visitEnterDate – дата та час входу до системи (початок сесії);

– private Date visitOutDate – дата та час виходу з системи (закінчення сесії);

Методи класу:

– public UserActionTableRecord(String login, int perTestCount, int perTestSuccess, int hisTestCount,int hisTestSuccess, int artTestCount, int artTestSuccess, int archTestCount, int archTestSuccess, int addScore, int totalScore, Date visitEnterDate, Date visitOutDate) – конструктор класу;

– гетери та сетери - методи доступу до приватних властивостей класу.

Клас TrueFalseQuestion описує сутність «завдання визначення вірності інформації».

Властивості класу:

– private String title – загальне текстове питання;

– private String header – заголовок питання;

– private String text – текстова частина питання ;

– private String photo – зображення до питання;

– private String footer – футер питання;

– private boolean isTrue – правильна відповідь (так/ні);

– private boolean answer – відповідь користувача.

Методи класу:

– public TrueFalseQuestion(String title, String header, String text, String photo, String footer, boolean isTrue, boolean answer) – конструктор класу.

– гетери та сетери - методи доступу до приватних властивостей класу.

Клас RightAnswerQuestion описує сутність «завдання знайти правильну відповідь».

Властивості класу:

– private String title – загальне текстове питання;

– private String header – заголовок питання;

– private String text – текстова частина питання ;

– private String photo – зображення до питання;

– private String [] answerList – перелік відповідей;

– private int isTrue – номер правільної відповіді.

Методи класу:

– public RightAnswerQuestion(String title, String header, String text, String photo, int isTrue, String [] answerList) – конструктор класу;

– гетери та сетери - методи доступу до приватних властивостей класу.

– Клас User описує сутність «користувач».

– Властивості класу:

– private String name – ім’я користувача;

– private String email – електрона пошта користувача;

– private String login – логін користувача;

– private String password – пароль користувача;

– private String avatar – ім’я файлу – аватару користувача;

– private int perTestCount – кількість завдань, що виконувалась користувачем з тематики «Персоналії»;

– private int perTestSuccess – кількість завдань, що вірно виконано користувачем з тематики «Персоналії»;

– private int hisTestCount – кількість завдань, що виконувалась користувачем з тематики «Історичні дати»;

– private int hisTestSuccess – кількість завдань, що вірно виконано користувачем з тематики «Історичні дати»;

– private int artTestCount – кількість завдань, що виконувалась користувачем з тематики «Історичні дати»;

– private int artTestSuccess – кількість завдань, що вірно виконано користувачем з тематики «Історичні дати»;

– private int archTestCount – кількість завдань, що виконувалась користувачем з тематики «Історичні дати»;

– private int archTestSuccess – кількість завдань, що вірно виконано користувачем з тематики «Історичні дати»;

– private int addScore – кількість додаткових балів;

– private int totalScore – загальна кількість балів;

– private Date visitEnterDate – дата та час входу до системи (початок сесії);

– private Date visitOutDate – дата та час виходу з системи (закінчення сесії).

Методи класу:

– public User(String name, String email, String login, String password, int perTestCount, int perTestSuccess,int hisTestCount, int hisTestSuccess, int artTestCount, int artTestSuccess, int archTestCount, int archTestSuccess, int addScore, Date lastVisitDate, String avatar) - конструктор класу;

– гетери та сетери - методи доступу до приватних властивостей класу.

3.3 Запити до бази даних

Відповідно до функціональних вимог програмної системи необхідно розробити множину запитів до бази даних. Наведемо перелік та реалізацію запитів:

1) запит при авторизації для перевірки існування реєстрації користувача у системі: «отримати записи користувача з заданим логіном та паролем»

```
SELECT * from users WHERE login = ? and password = ?
```

2) запит при реєстрації для перевірки унікальності поля логіну, якій є унікальним індексом: «отримати записи з таблиці користувачів за заданим логіном» :

```
SELECT * from users WHERE login = ?
```

3) запит при реєстрації для перевірки унікальності поля адресу електронної пошти, яке є унікальними індексом «отримати записи з таблиці користувачів за заданою адресою електронної пошти»:

```
SELECT * from users WHERE email = ?
```

4) запит для створення об’єкту класу User «отримати усіх полів запису користувача за заданими логіном та паролем користувача»:

```
SELECT name, email, perTestCount, perTestSuccess,
     hisTestCount, hisTestSuccess, artTestCount,
     artTestSuccess, archTestCount, archTestSuccess,
     addScore, avatar
FROM users
WHERE login = ? and password = ?";
```

5) запит для створення об’єкту класу LeaderTableRecord: «отримати усі записи таблиці користувачів з розрахуванням кількості балів за виконання завдань»

```
SELECT us.login,
     CASE WHEN us.perTestCount>0
     THEN round(us.perTestSuccess/us.perTestCount * 100) ELSE 0
     END as perScore,
     CASE WHEN us.hisTestCount>0
     THEN round(us.hisTestSuccess/us.hisTestCount * 100) ELSE 0
     END as hisScore,
     CASE WHEN us.artTestCount>0
     THEN round(us.artTestSuccess/us.artTestCount * 100) ELSE 0
     END as artScore,
     CASE WHEN us.archTestCount>0
     THEN round(us.archTestSuccess/us.archTestCount * 100) ELSE 0
     END as archScore,
```

```
    us.addScore,
    us.lastVisitDateTime
from users as us
```

6) запит для створення об’єкту класу UserActionTableRecord при відображенні проміжних досягнень користувача: «отримати усі записи результатів виконання завдань, додаткових та підсумкових балів, а також час початку та кінця сесії з таблиці сесій за заданим логіном користувача»:

```
SELECT usa.perTestCount,
    usa.perTestSuccess,
    usa.hisTestCount,
    usa.hisTestSuccess,
    usa.artTestCount,
    usa.artTestSuccess
    usa.archTestCount,
    usa.archTestSuccess,
    usa.addScore,
    usa.totalScore,
    visitEnterDate,
    visitOutDate
FROM useractions as usa
join users as us
    on us.idusers = usa.idusers
    and us.login = ?;
```

7) запит при завершенні роботи з системою: «додати запис поточних результатів до таблиці сесій за заданим логіном користувача»:

```
INSERT INTO useractions
      (idusers, perTestCount, perTestSuccess,
       hisTestCount, hisTestSuccess, artTestCount,
       artTestSuccess, archTestCount,archTestSuccess,
```

```
        addScore, totalScore, visitEnterDate, visitOutDate)
    SELECT ?,?,?,?,?,?,?,?,?,?,?,?,? from users WHERE login = ?"
```

8) запити при реєстрації користувача: «додати запис у таблицю користувачів»:

```
INSERT INTO users ( name, email, login, password,
lastVisitDateTime, createDateTime VALUES(?,?,?,?,?,?)
```

9) запит при завершенні роботи з системою «оновити запис новими стоговими даними у таблиці користувачів за заданим логіном користувача»:

```
UPDATE users
     set perTestCount = ? ,
     perTestSuccess = ?,
     hisTestCount = ?,
     hisTestSuccess = ?,
     artTestCount =? ,
     artTestSuccess = ?,
     archTestCount = ?,
     archTestSuccess = ?,
     addScore = ?,
     lastVisitDateTime = ?
WHERE login = ?
```

10) запит при оновленні даних профілю користувача «оновити запис новими даними у таблиці користувачів за заданим логіном користувача»:

```
UPDATE users
 set name = ? ,
 password = ?,
 email = ?,
WHERE login = ?
```

11) запити при завантажуванні даних системи для заповнення структур даних для побудови завдань даними з бази: «отримати усі записи з заданої таблиці»

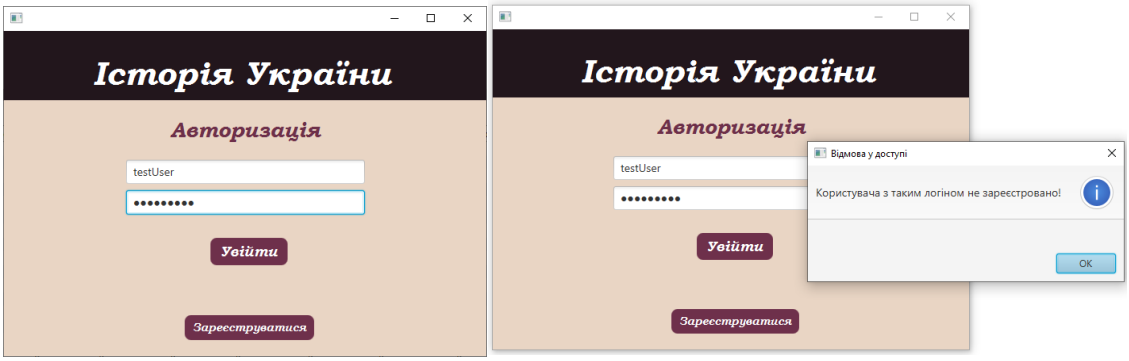
```
SELECT * from ?
```

3.4 Тестування програмного забезпечення

Для перевірки коректної роботи програмної системи виконаємо тестування її роботи.

Тест 1. Авторизація користувача.

Після запуску додатку відображається вікно авторизації користувача (див. рис. 3.5а). При введенні даних неіснуючого користувача, система попереджає про відсутність даних (див. рис. 3.5а).

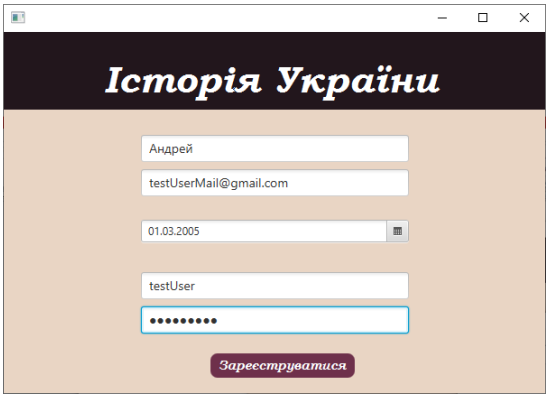


а б

Рисунок 3.5 – Вікно авторизації користувача

Тест 2. Реєстрація користувача.

Вікно авторизації має кнопку «Зареєструватися», по натисненню на яку відкривається вікно реєстрації. Після натискання кнопки «Зареєструватися» (див. рис. 3.6а) відкривається головне вікно програми (див. рис.3.6 б).



а

б

Рисунок 3.5 – Реєстрація користувача

Тест 3. Обрання розділу, виду тесту на проведення тестування.

Головне вікно має кнопки обрання розділу «Персоналії», «Історичні дати», «Образотворче мистецтво» та «Архітектура». При натисненні відображається перелік типів завдань (рис. 3.6).

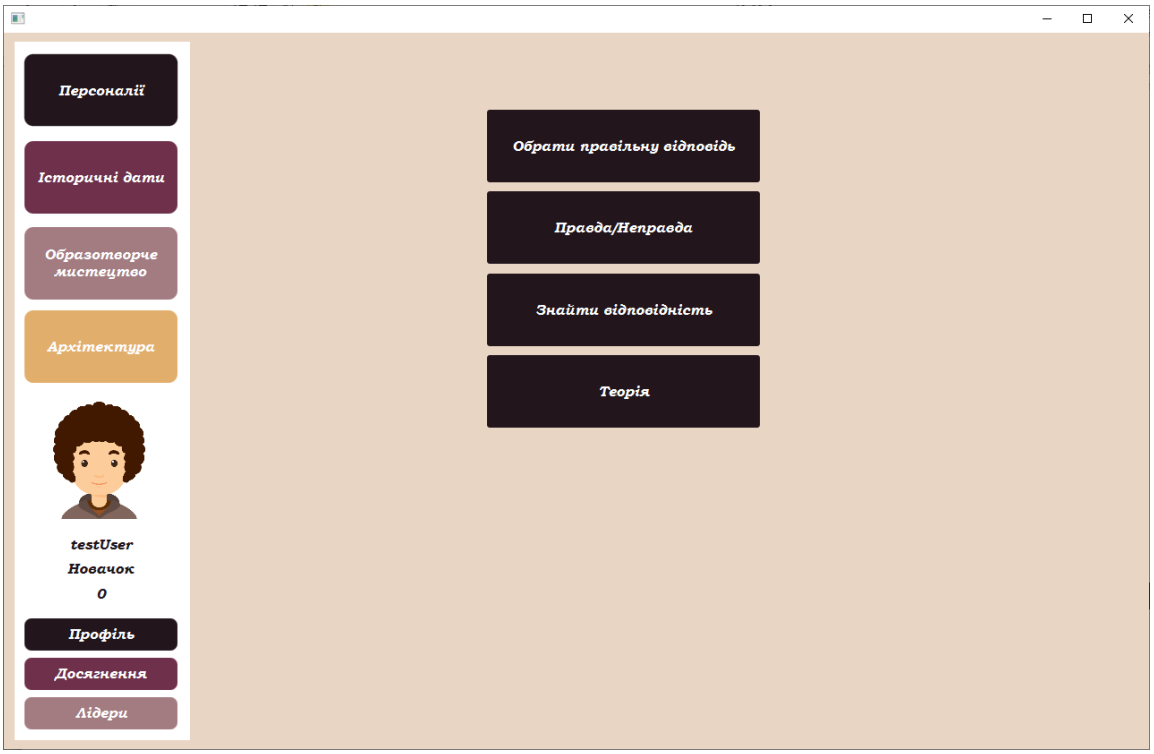


Рисунок 3.6 – Вікно вибору типу завдань

Приклади вигляду етапів виконання завдань різного типу та різних розділів наведено на рисунках 3.7 – 3.12.

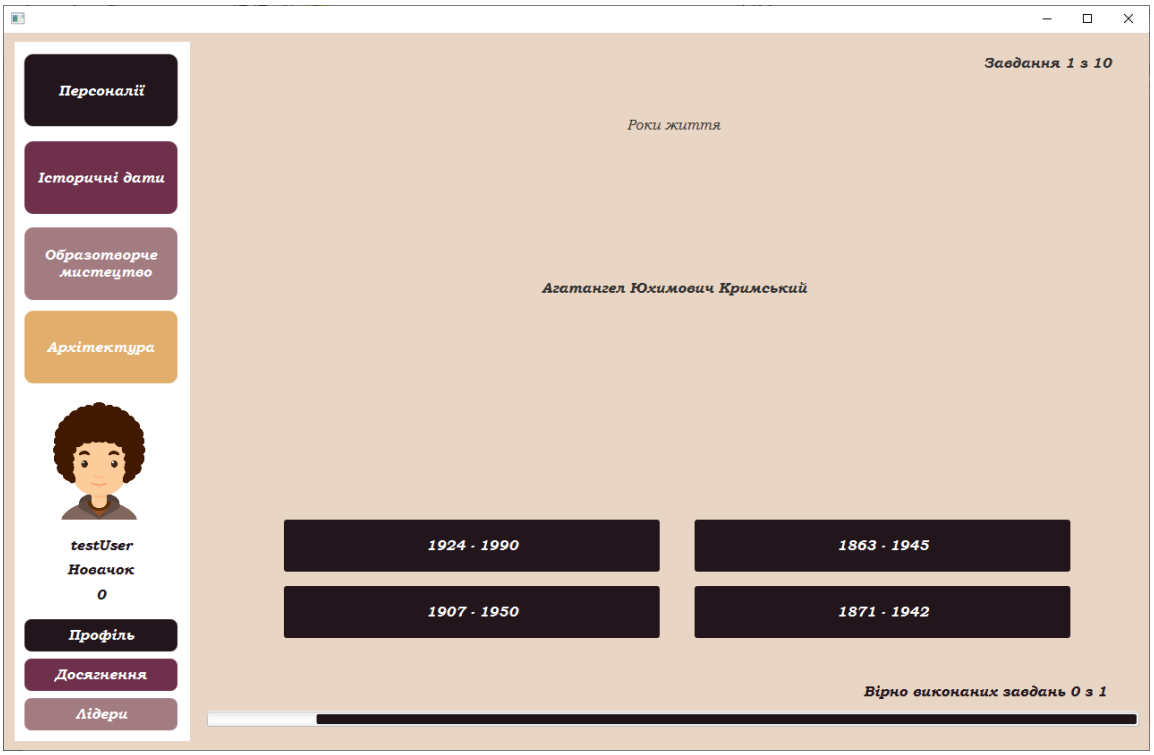


Рисунок 3.7 – Завдання «правильна відповідь» розділу «Персоналії»

Рисунок 3.8 – Завдання «правильна відповідь» розділу «Персоналії»

Рисунок 3.9 – Завдання «знайти відповідність» розділу «Персоналії»

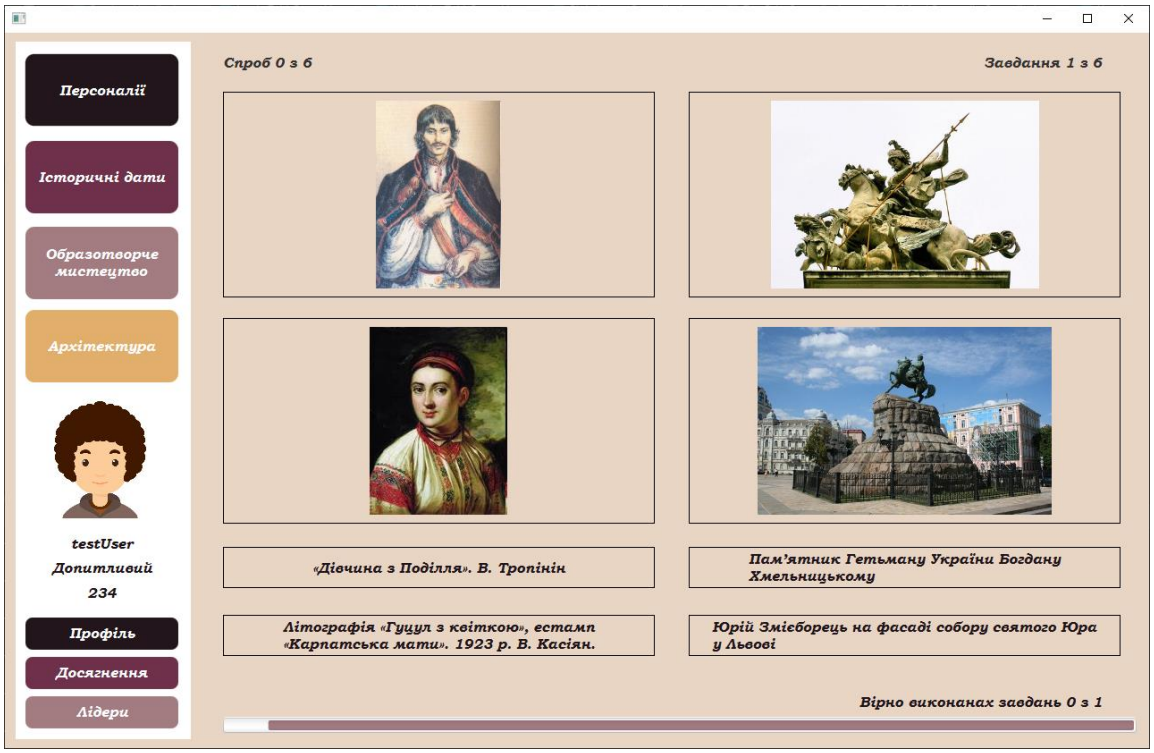


Рисунок 3.10 – Завдання «правильна відповідь» розділу «Образотворче мистецтво»

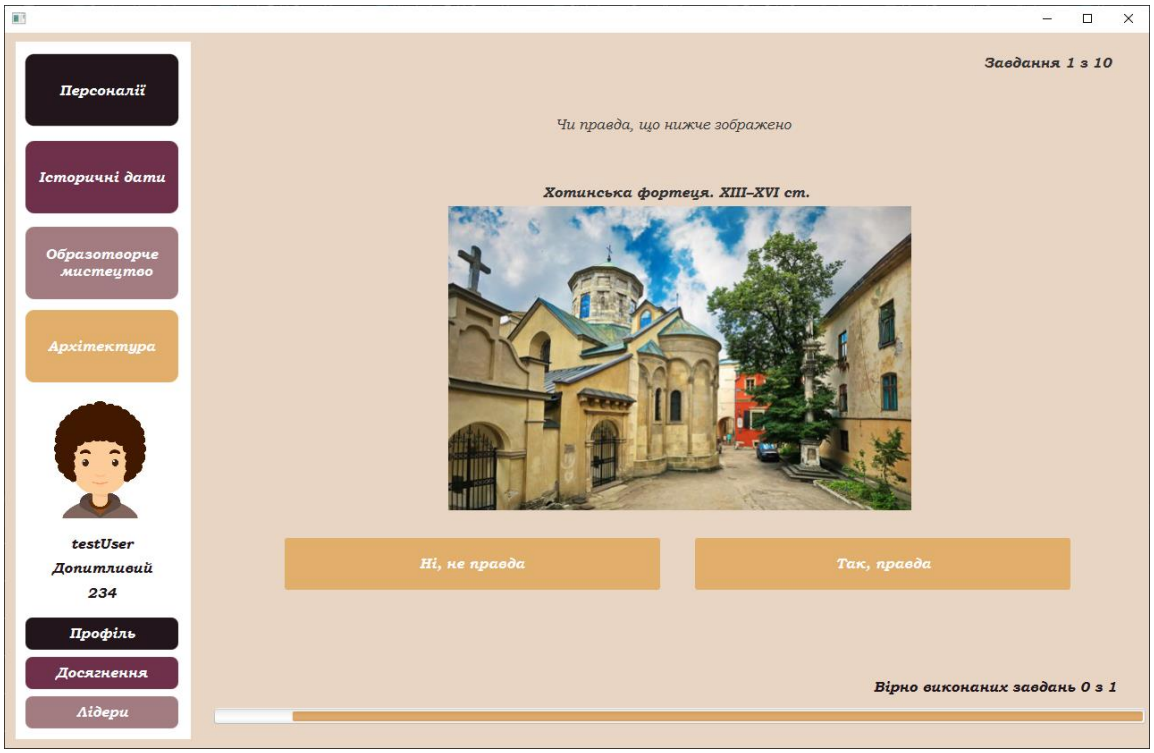


Рисунок 3.11 – Завдання «правда чи ні» розділу «Архітектура»

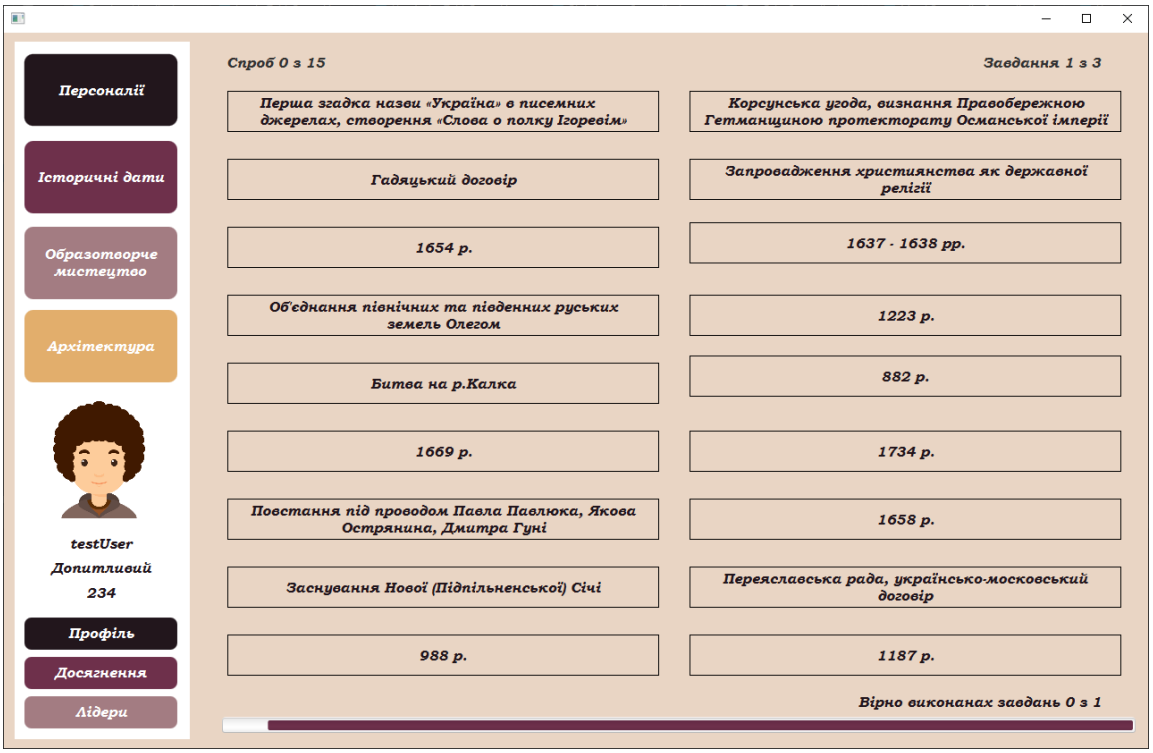


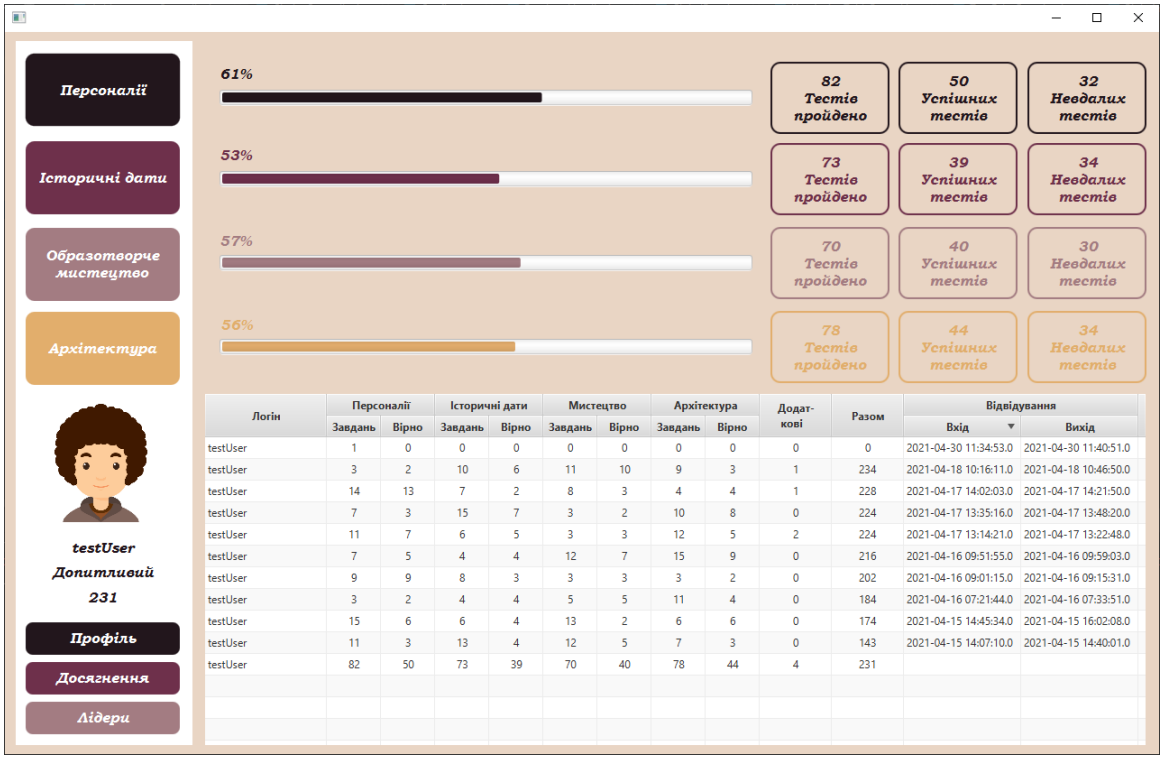
Рисунок 3.12 – Завдання «знайти відповідність» розділу «Історичні дати»

Після закінчення виконання завдання відображається вікно з результатами, що отримав користувач (див. рис. 3.13).

Рисунок 3.13 – Результати виконання завдання

Тест 4. Перегляд досягнень користувача.

При натисканні кнопки «Досягнення» користувачеві надається загальна та проміжна інформація, що до його досягнень (див. рис. 3.14).



Логін	Персоналії		Історичні дати		Мистецтво		Архітектура		Додат-кові	Разом	Відвідування	
	Завдань	Вірно	Завдань	Вірно	Завдань	Вірно	Завдань	Вірно			Вхід	Вихід
testUser	1	0	0	0	0	0	0	0	0	0	2021-04-30 11:34:53.0	2021-04-30 11:40:51.0
testUser	3	2	10	6	11	10	9	3	1	234	2021-04-18 10:16:11.0	2021-04-18 10:46:50.0
testUser	14	13	7	2	8	3	4	4	1	228	2021-04-17 14:02:03.0	2021-04-17 14:21:50.0
testUser	7	3	15	7	3	2	10	8	0	224	2021-04-17 13:35:16.0	2021-04-17 13:48:20.0
testUser	11	7	6	5	3	3	12	5	2	224	2021-04-17 13:14:21.0	2021-04-17 13:22:48.0
testUser	7	5	4	4	12	7	15	9	0	216	2021-04-16 09:51:55.0	2021-04-16 09:59:03.0
testUser	9	9	8	3	3	3	3	2	0	202	2021-04-16 09:01:15.0	2021-04-16 09:15:31.0
testUser	3	2	4	4	5	5	11	4	0	184	2021-04-16 07:21:44.0	2021-04-16 07:33:51.0
testUser	15	6	6	4	13	2	6	6	0	174	2021-04-15 14:45:34.0	2021-04-15 16:02:08.0
testUser	11	3	13	4	12	5	7	3	0	143	2021-04-15 14:07:10.0	2021-04-15 14:40:01.0
testUser	82	50	73	39	70	40	78	44	4	231		

Рисунок 3.14 – Вікно досягнень користувача

Тест 5. Перегляд рейтингу користувачів.

При натисканні кнопки «Лідери» користувачеві надається загальна інформація що до досягнень усіх користувачів системи (див. рис. 3.15).

Тест 6. Перегляд теоретичного матеріалу.

При натисканні кнопки «Теорія» у вікні вибору типу завдань (дмв.рис.3.6) користувачеві відображається теоретичний матеріал за обраним розділом (рис. 3.16).

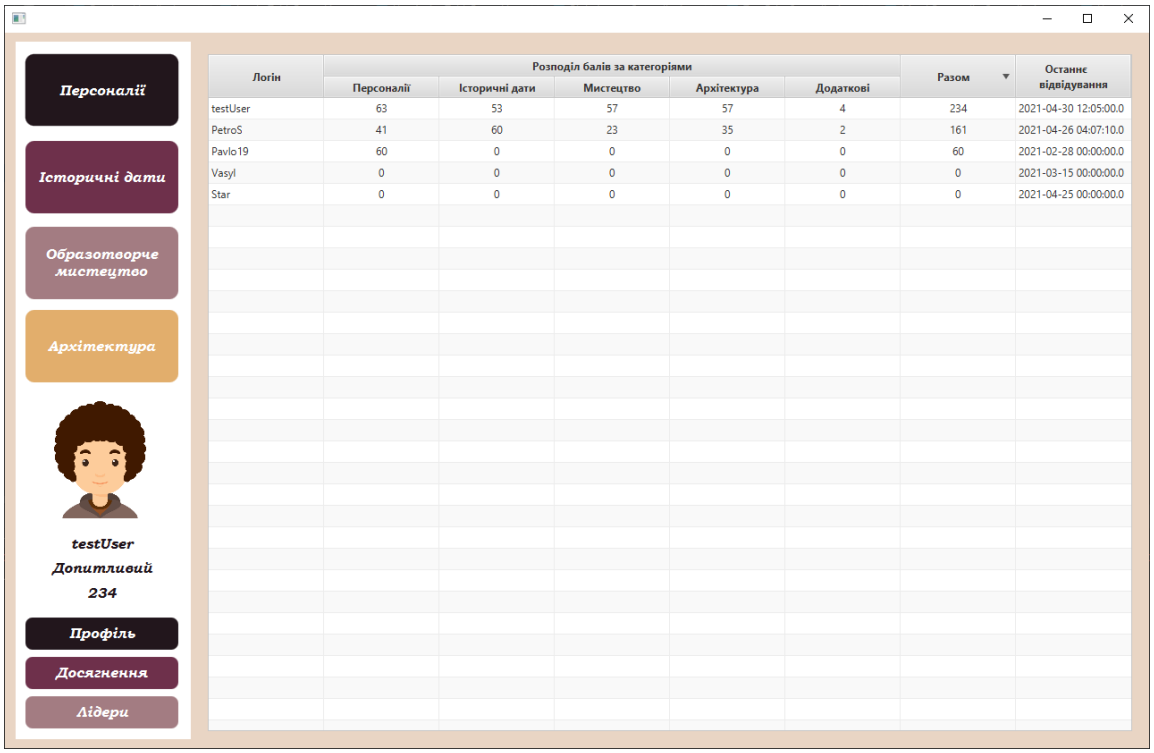


Рисунок 3.15 – Результати виконання завдання

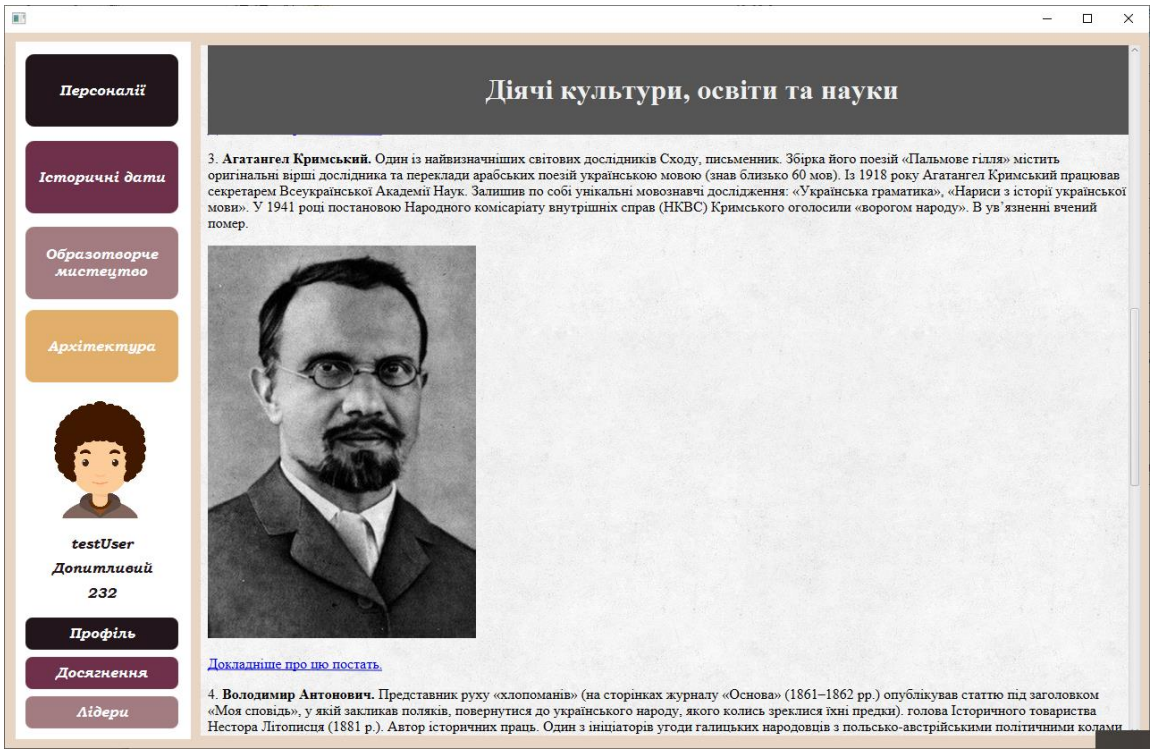


Рисунок 3.16 – Теоретичний матеріал за розділом «Персоналії»

Тест 7. Перегляд та редагування даних профілю користувача.

При натисканні кнопки «Профіль» користувачеві відображається дані його профілю з можливістю їх коригування (рис. 3.17).

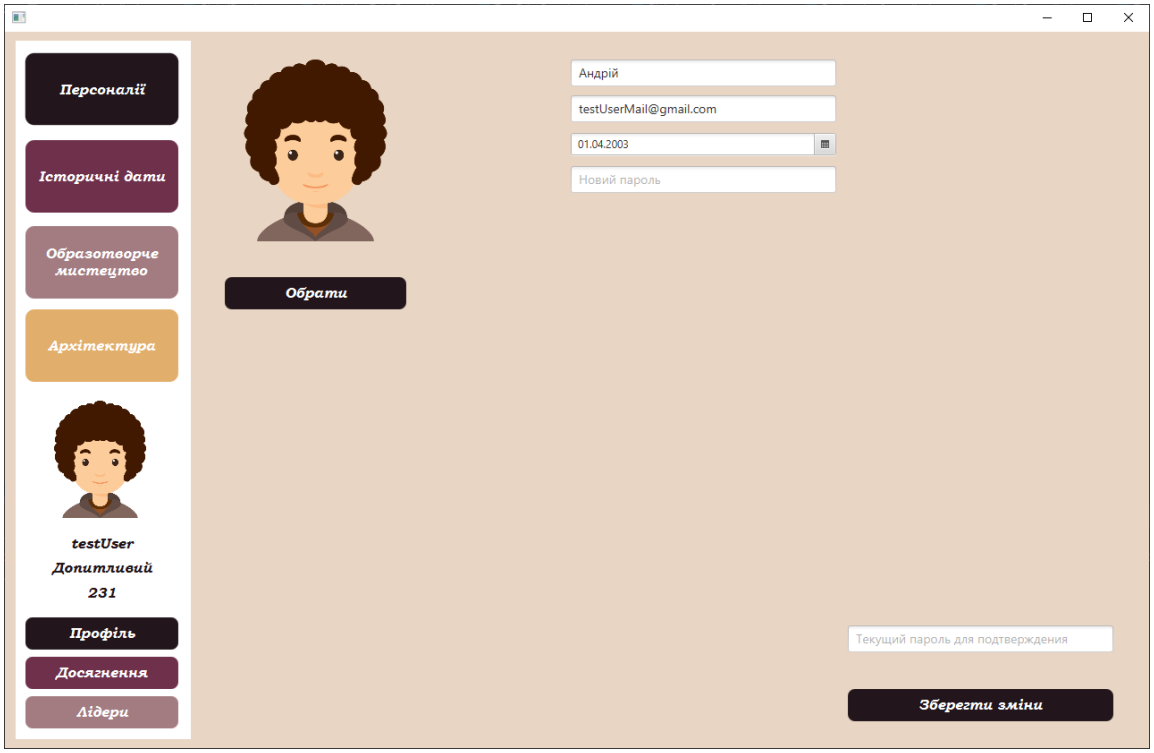


Рисунок 3.17 – Дані профілю користувача

В результаті проведеного тестування програмного додатку отримано наступні результати:

1) на всіх етапах роботи додаток веде себе стабільно, не викликає помилок та аварійного завершення;
2) усі дані зображень та тексту завантажуються коректно, не спостерігається втрат даних, або зайвих даних;
3) у процесі виконання завдань процес візуалізації та виконання завдання відповідає вимогам, коректно відображаються дії користувача;
4) дані результатів оцінки рівня навчання розраховуються, відображаються та зберігаються коректно;

5) наведення табличного уявлення даних стогових та проміжних досягнень користувача відповідає дійсності, запити до бази даних виконуються коректно.

Таким чином, можна зробити висновок, що за результатами тестування розроблена програмна система повністю відповідає усім вимогам, що було висунуто при постановці задачі розробки.

ВИСНОВКИ

В результаті виконання бакалаврської роботи було спроектовано та реалізовано програмно інтерактивну систему для вивчення історії України з метою підготовки до ЗНО.

При виконанні роботи було виконано наступні задачі.

1. Розглянуто поняття зовнішнього незалежного тестування та його місце у сфері освіти, проаналізовано основні форми та види завдань для перевірки знань, визначено роль тестування в оцінки знань та навичок в сфері сучасної освіти, розглянуто принцип побудови тестових зошитів по ЗНО з історії України, форми тестових завдань, що використовуються в зошитах.

2. Розглянуть поняття інтерактивної системи та використання елементів гейміфікації у інтерактивних системах навчання.

3. Проаналізовано сучасних існуючі програмні продукти для вивчення історії України та підготовки до ЗНО з історії України, виявлено їх загальну структуру, принципи функціонування, а також переваги та недоліки. Взаголом аналогічні програмні системи мають певний набір візуальних елементів, подібні принципи функціонування та схожі методи оцінювання результатів. Суттєвими відмінностями є обсяг та принцип побудови тестових завдань перевірки знань.

4. Обґрунтовано вибір засобів програмної реалізації інтерактивної системи мовою Java з використанням середовища розробки програмного забезпечення IntelliJ IDEA Community 2020.3, платформи створення додатків з насиченим графічним інтерфейсом JavaFX та інструменту візуального макетування SceneBuilder з використанням мови розробки користувацьких інтерфейсів FXML.

5. Виконано проектування структури, схеми функціонування та інтерфейсу програмної системи. Засобами мови UML розроблено діаграми використання, діяльності та класів, засобами мови FXML розроблено простий, зручний та зрозумілий інтерфейс користувача, який надає можливості візуалізації даних та обробки дії користувача, отримання користувачем як особистих даних показників

якості навчання так і результатів інших користувачів у вигляді рейтингових таблиць.

6. Спроектовано базу даних для зберігання даних користувачів та даних для побудови завдань, розроблено ER-діаграму (схему «сутність-зв'язок»), засобами реляційної система управління базами даних MySQL та візуального інструменту MySQL Workbench спроектовано структуру бази даних та таблиць.

7. Обрано форми та розроблено структуру завдань для закріплення та перевірки знань: обрання однієї правильної відповіді серед чотирьох наданих варіантів відповідей на поставлене питання, підтвердження чи спростування істинності наданого факту та встановлення відповідності між елементами двох множин фактів.

8. Розроблено алгоритмічне забезпечення програмної системи, функцій побудови завдань, процесу виконання завдання, завантаження та збереження даних, розрахунку та візуалізації оцінок результатів навчання.

9. Розроблено програмне забезпечення програмної системи, яке відповідає вимогам та задачам, що було висунуто.

10. Проведено тестування роботи програмної системи. За результатами тестування розроблена програмна система повністю відповідає усім вимогам, що було висунуто до системи при проектуванні.

ПЕРЕЛІК ПОСИЛАНЬ

1. Вікіпедія. Вільна енциклопедія. Зовнішнє незалежне оцінювання [Електронний ресурс] – Режим доступу: https://uk.wikipedia.org/wiki/_Зовнішнє_незалежнеоцінювання (дата звернення 29.04.2021). – Назва с екрану.

2. ЗНО-ОНЛАЙН. Онлайн — тести зовнішнього оцінювання [Електронний ресурс] – Режим доступу: https://zno.osvita.ua/ (дата звернення 29.04.2021). – Назва с екрану.

3. Як підготуватися до ЗНО з історії України 2021: лайфхаки та поради експертів [Електронний ресурс] – Режим доступу: https://life.pravda.com.ua/society/2020/03/13/240163/ (дата звернення 29.04.2021). – Назва с екрану.

4. Самылкина Н.Н. Современные средства оценивание результатов обучения / Н.Н. Самылкина, - М: БИНОМ.,2007.-172с.

5. Мышко С.А. Проблема тестирования в системе образования США. Дисс.к.п.н./ С.А. Мышко -Ужгород, 1982.

6. Аванесов В.С. Композиция тестовых задании. -2-ое изд.испр. и доп. / В.С. Аванесов, - М.,1998.-217с.

7. Челышкова Теория и практика конструирования педагогических тестов / М.Б. Челышкова, -М.,2001- 432 с.

8. Сисоєва С.О. Інтерактивні технології навчання дорослих: навчальнометодичний посібник / Сисоєва С.О.; НАПН України, Ін-т педагогічної освіти і освіти дорослих. – К.: ВД «ЕКМО», 2011. – 324 с.

9. Вікіпедія. Вільна енциклопедія Ігровий процес [Електронний ресурс] – Режим доступу: https://uk.wikipedia.org/wiki/ Ігровий_процес (дата звернення 29.04.2021). – Назва с екрану.

10. Oxland K. Gameplay and Design / K. Oxland (en), - Addison-Wesley, 2004 – 368 p.

11. Подробнее о технологии Java [Електронний ресурс] – Режим доступу: https://www.java.com/ru/about/ (дата звернення 29.04.2021). – Назва с екрану.

12. Вікіпедія. Вільна енциклопедія. JavaFx [Електронний ресурс] – Режим доступу: https://uk.wikipedia.org/wiki/JavaFX (дата звернення 29.04.2021). – Назва с екрану.

13. Вікіпедія. Вільна енциклопедія. Itellij IDEA [Електронний ресурс] – Режим доступу: https://uk.wikipedia.org/wiki/IntelliJ_IDEA (дата звернення 29.04.2021). – Назва с екрану.

14. Java Is the Language of Possibilities [Електронний ресурс] – Режим доступу: https://www.oracle.com/java/technologies/ (дата звернення 29.04.2021). – Назва с екрану.

15. Кей С. Хорстманн Java SE 8. Вводный курс / Кей С. Хорстманн, - «Вільямс», 2014 – 208 с.

16. Фрэд Лонг Руководство для программиста на Java: 75 рекомендаций по написанию надежных и защищенных программ / Фрэд Лонг, - «Вільямс», 2014–- 256 с.

17. Учебник по JavaFX: FXML и SceneBuilder [Електронний ресурс] – Режим доступу: https://habr.com/ru/post/474982/ (дата звернення 29.04.2021). – Назва с екрану.

18. Хабр. Сообщество IT специалистов. Как подключить MySQL к бесплатной версии Intellij IDEA (community) [Електронний ресурс] – Режим доступу: https://habr.com/ru/sandbox/146588/ (дата звернення 29.04.2021). – Назва с екрану.

Додаток А

ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ (Презентація)

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ
НАВЧАЛЬНО–НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
Кафедра інженерії програмного забезпечення

БАКАЛАВРСЬКА РОБОТА
на ступінь вищої освіти бакалавр
на тему: «**Розробка програмного забезпечення інтерактивної системи для вивчення історії за допомогою мови програмування Java**»

Виконав: студент 4 курсу, групи ПД–42,
спеціальності «Програмна інженерія»
Щербина Андрій Сергійович
Керівник: Золотухіна Оксана Анатоліївна, к.т.н.

Київ - 2021

Постановка задачі

2

Тема роботи: Розробка програмного забезпечення інтерактивної системи для вивчення історії за допомогою мови програмування Java .

Мета роботи: спрощення процесу вивчення історії України при підготовці до зовнішнього незалежного оцінювання за рахунок використання інтерактивної системи навчання.

Об'єкт дослідження: процес вивчення історії умовах застосування сучасних інформаційних технологій.

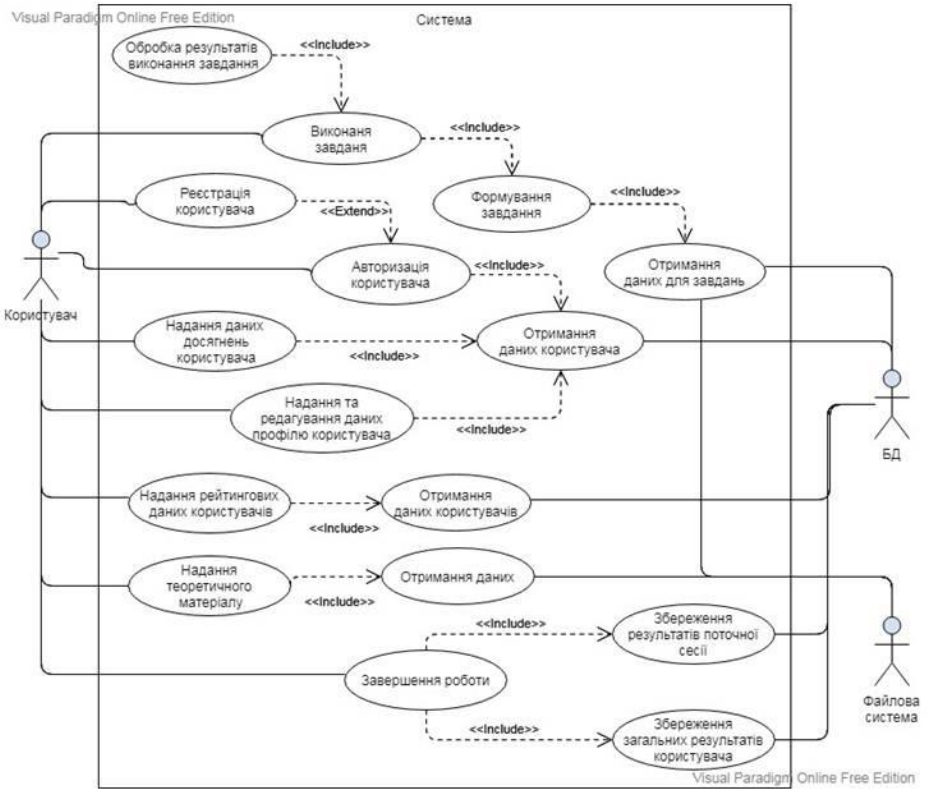
Предмет дослідження: створення інтерактивної навчальної системи для підготовки до зовнішнього незалежного оцінювання з історії України

Методи дослідження: методи теорії інформації, методи педагогічного тестування, методи проектування та розробки користувацьких інтерфейсів, методи проектування та розробки інтерактивних програмних додатків.

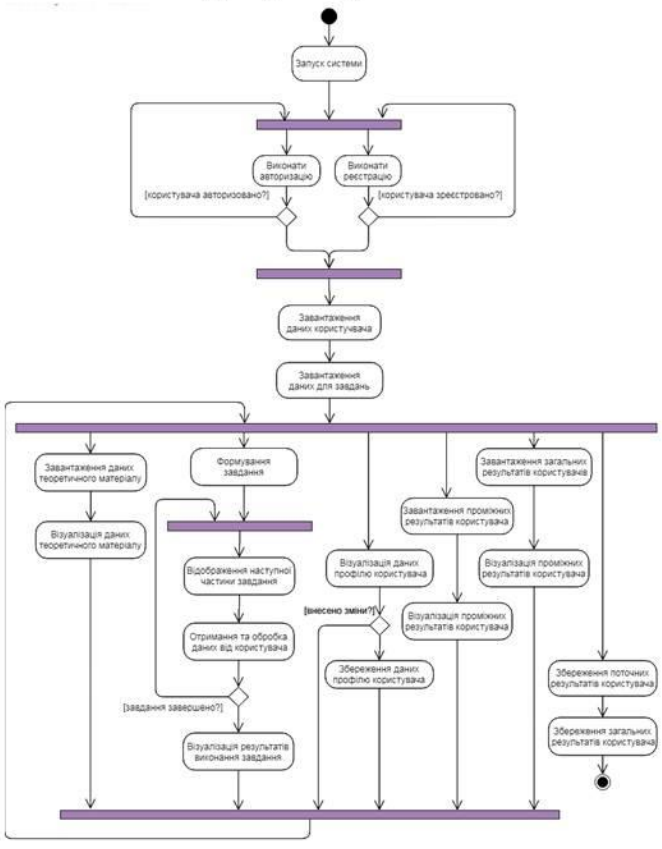
Задачі роботи:

1) провести огляд предметної області, виконати аналіз сучасного стану питання;
2) оглянути основні види навчаючих завдань та використання тестування для оцінювання якості отриманих знань;
3) розглянути питання використання інтерактивних систем та гейміфікації в освіті та процесі навчання;
4) обгрунтувати обрання засобів програмної реалізації програмної системи;
5) виконати проектування структури, схеми функціонування та інтерфейсу програмної системи;
6) виконати проектування бази даних системи для зберігання даних користувачів та даних для побудови завдань;
7) розробити структуру завдань для тренування та принципів оцінювання результатів навчання при використанні програмної системи;
8) розробити алгоритмічне забезпечення програмної системи;
9) розробити програмне забезпечення програмної системи;
10) провести тестування та оцінку якості роботи програмної системи.

Діаграма варіантів використання



Діаграма діяльності



Оцінювання досягнень користувача

Типи завдань

1. «Правильна відповідь» – обрати одну вірну відповідь з чотирьох наданих
2. «Правда чи ні» – вказати, чи є істиною наведений факт
3. «Знайти відповідність» – знайти пари пов'язаних елементів

Система оцінки виконання завдань

$$\text{totalScore} = \text{sumScore} + \sum \text{addScore}$$

де: totalScore – підсумкова кількість балів користувача;
sumScore – сумарна кількість балів за всіма розділами;
addScore – додаткові бали.

```
sumScore = round(perTestSuccess/perTestCount * 100)
 + round(hisTestSuccess/hisTestCount * 100)
 + round(artTestSuccess/artTestCount * 100)
 + round(archTestSuccess/archTestCount * 100),
```

Тематичні розділи та інформація, яка використовується

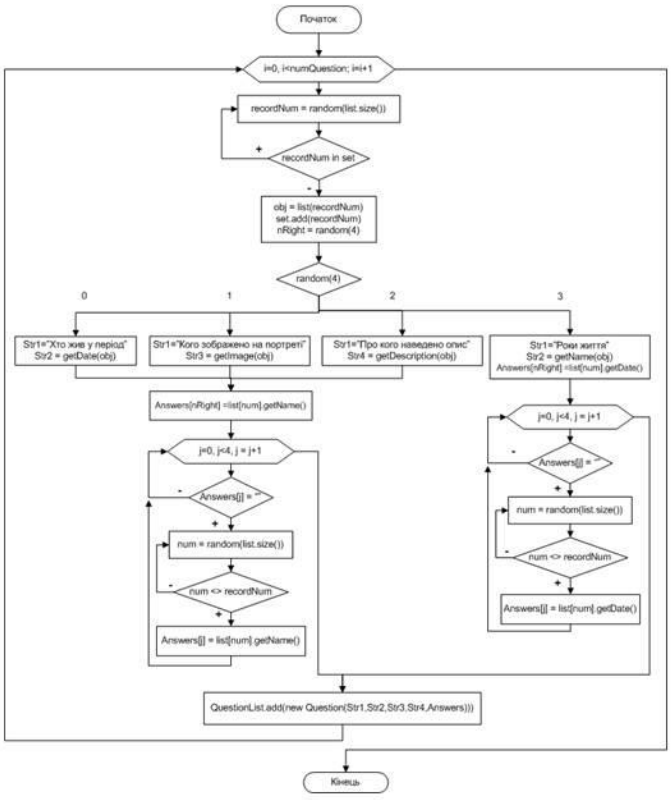
1. «Персоналії»:
   – ім'я;
   – роки життя;
   – портретне зображення;
   – короткі характеристика та опис діяльності.
2. «Історичні дати»:
   – назва події;
   – дата (період) коли подія відбувалася.
3. «Образотворче мистецтво»:
   – назва витвору мистецтва;
   – фотографічне зображення.
4. «Архітектура»:
   – назва пам'ятки архітектури;
   – фотографічне зображення.

де: sumScore – сумарна кількість балів за всіма розділами;
perTestCount – кількість виконаних завдань за розділом «Персоналії»;
perTestSuccess – кількість успішно виконаних завдань за розділом «Персоналії»;
hisTestCount – кількість виконаних завдань за розділом «Історичні дати»;
hisTestSuccess – кількість успішно виконаних завдань за розділом «Історичні дати»;
artTestCount – кількість виконаних завдань за розділом «Образотворче мистецтво»;
artTestSucces – кількість успішно виконаних завдань за розділом «Образотворче мистецтво»;
archTestCount – кількість виконаних завдань за розділом «Архітектура»;
archTestSuccess – кількість успішно виконаних завдань за розділом «Архітектура».

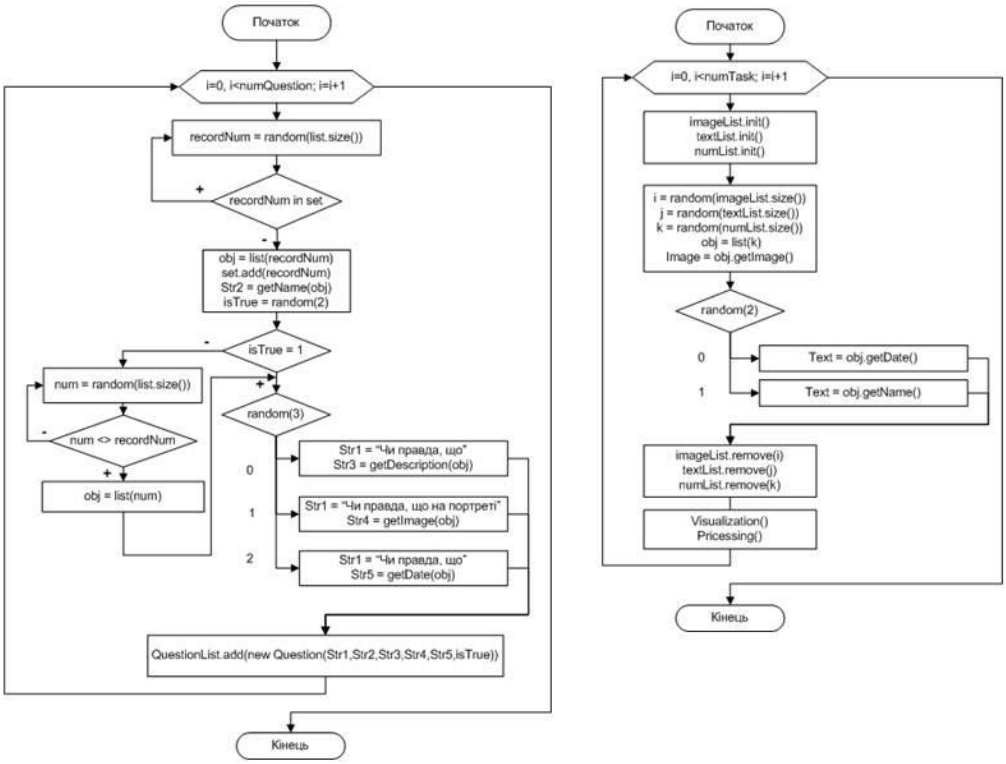
Завдання виконано успішно

– якщо > 80% правильних відповідей в завданнях «правильна відповідь» та «правда чи ні»;
– якщо >65% правильних відповідей в завданнях «знайти відповідність» розділів «Персоналії» та «Історичні дати»
– якщо >70% правильних відповідей в завданнях «знайти відповідність» розділів «Образотворче мистецтво» та «Архітектура»

Узагальнена блок-схема формування набору даних завдання «правильна відповідь»



Узагальнена блок-схема формування набору даних завдань «правда чи ні» та «зайти відповідність»



ER діаграма бази даних

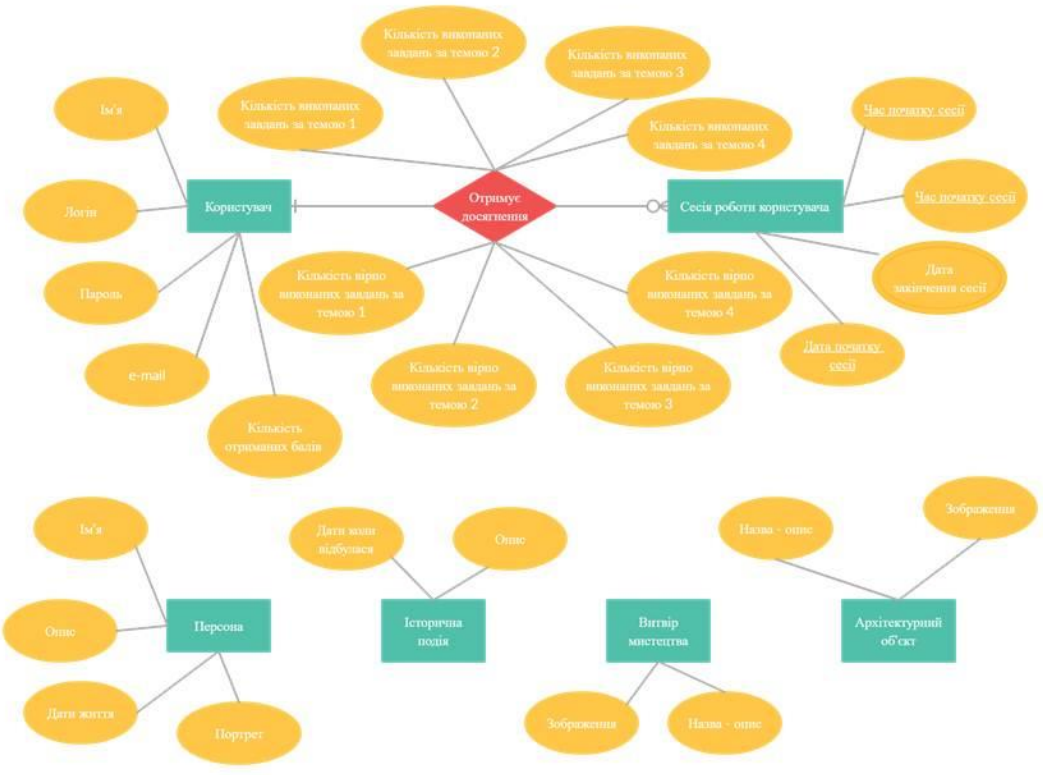
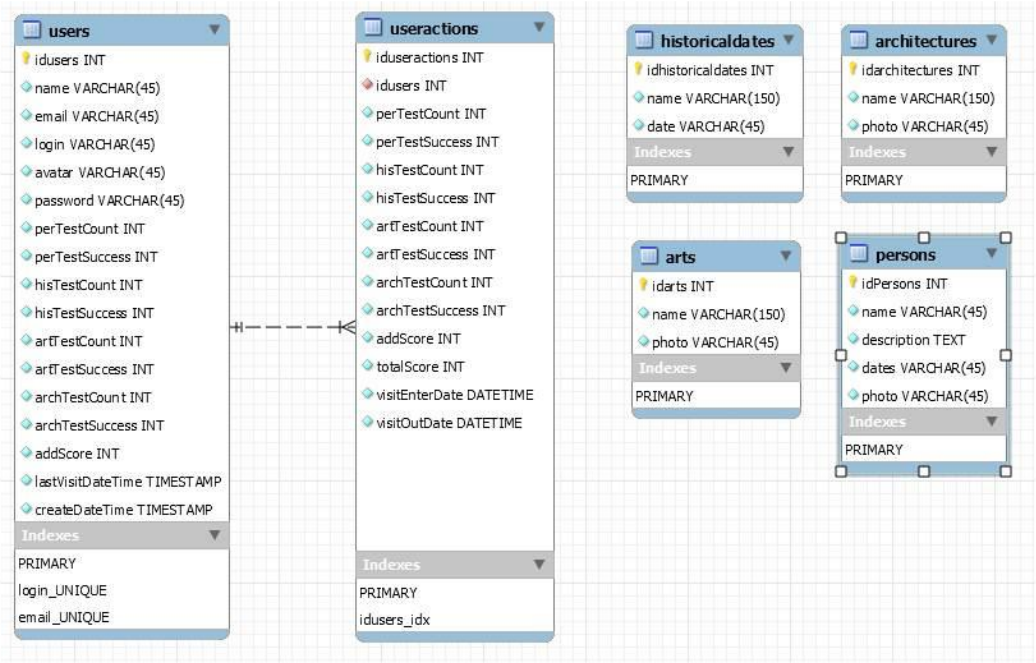
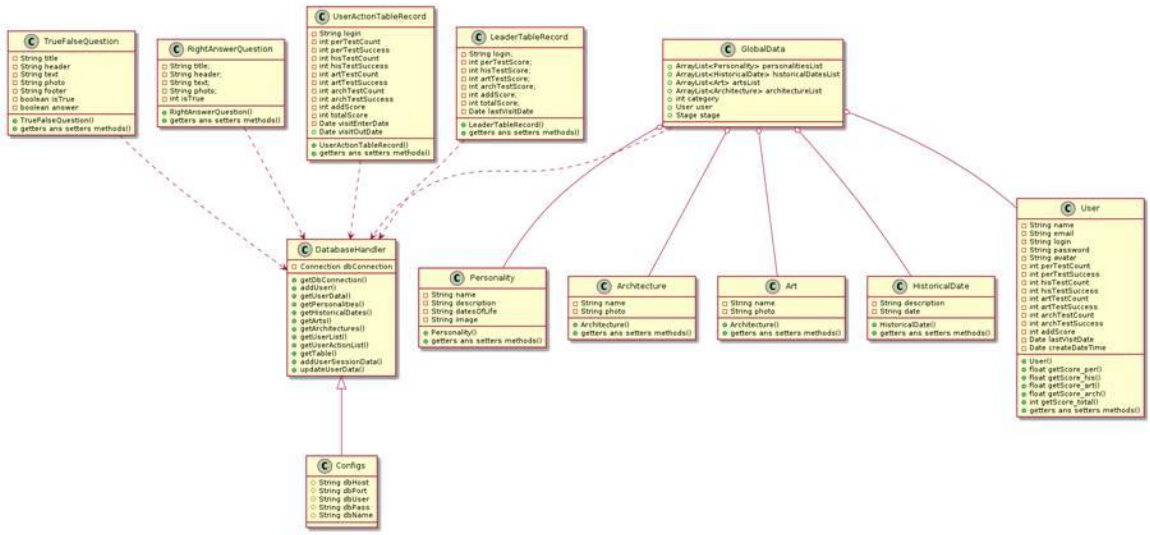


Схема бази даних



Діаграма классів



Приклади роботи програми

Висновки

В результаті виконання бакалаврської роботи було спроектовано та реалізовано програмно інтерактивну систему для вивчення історії України з метою підготовки до ЗНО.

При виконанні роботи було виконано наступні задачі.

1. Розглянуто поняття зовнішнього незалежного тестування та його місце у сфері освіти, проаналізовано основні форми та види завдань для перевірки знань, визначено роль тестування в оцінки знань та навичок в сфері сучасної освіти, розглянуто принцип побудови тестових зошитів по ЗНО з історії України, форми тестових завдань, що використовуються в зошитах.

2. Розглянуть поняття інтерактивної системи та використання елементів гейміфікації у інтерактивних системах навчання.

3. Проаналізовано сучасних існуючі програмні продукти для вивчення історії України та підготовки до ЗНО з історії України, виявлено їх загальну структуру, принципи функціонування, а також переваги та недоліки..

4. Обґрунтовано вибір засобів програмної реалізації інтерактивної системи мовою Java з використанням середовища розробки програмного забезпечення IntelliJ IDEA Community 2020.3, платформи створення додатків з насиченим графічним інтерфейсом JavaFX та інструменту візуального макетування SceneBuilder з використанням мови розробки користувацьких інтерфейсів FXML.

5. Виконано проектування структури, схеми функціонування та інтерфейсу програмної системи, розроблено діаграми використання, діяльності та класів.

6. Спроектовано базу даних для зберігання даних користувачів та даних для побудови завдань, розроблено ER-діаграму (схему «сутність-зв'язок»), спроектовано структуру бази даних та таблиць.

7. Обрано форми та розроблено структуру завдань для закріплення та перевірки знань: обрання однієї правильної відповіді серед чотирьох наданих варіантів відповідей на поставлене питання, підтвердження чи спростування істинності наданого факту та встановлення відповідності між елементами двох множин фактів.

8. Розроблено алгоритмічне забезпечення програмної системи, функцій побудови завдань, процесу виконання завдання, завантаження та збереження даних, розрахунку та візуалізації оцінок результатів навчання.

9. Розроблено програмне забезпечення програмної системи, яке відповідає вимогам та задачам, що було висунуто.

10. Проведено тестування роботи програмної системи. За результатами тестування розроблена програмна система повністю відповідає усім вимогам, що було висунуто до системи при проектуванні.

Додаток Б