МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

МАТЕМАТИЧНИЙ ФАКУЛЬТЕТ
Кафедра комп'ютерних наук

КВАЛІФІКАЦІЙНА РОБОТА МАГІСТРА

на тему: **«РОЗРОБКА ТЕХНОЛОГІЇ ПРОВЕДЕННЯ АДАПТИВНОГО ТЕСТУВАННЯ ДЛЯ ДІАГНОСТИКИ РІВНЯ ПІДГОТОВЛЕНОСТІ ДО ЗНО З МАТЕМАТИКИ»**

Виконала: студентка 2 курсу, групи 8.1228-з
спеціальності 122 комп'ютерні науки
освітньої програми комп'ютерні науки

І. В. Гузь
(ініціали та прізвище)

Керівник доцент кафедри комп'ютерних наук, доцент, к.т.н. Решевська К. С.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Рецензент доцент кафедри загальної математики, доцент, к.ф.-м.н. Стєганцев Є. В.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Запоріжжя – 2020

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет математичний

Кафедра комп'ютерних наук

Рівень вищої освіти магістр

Спеціальність 122 комп'ютерні науки
(шифр і назва)

Освітня програма комп'ютерні науки

ЗАТВЕРДЖУЮ

Завідувач кафедри комп'ютерних наук, к.т.н., доцент Борю С. Ю.

(підпис)

« 29 » 05 2019 р.

З А В Д А Н Н Я

НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТЦІ

Гузь Ірині Віталіївні
(прізвище, ім'я та по-батькові)

1. Тема роботи (проекту) Розробка технології проведення адаптивного тестування для діагностики рівня підготовленості до ЗНО з математики

керівник роботи (проекту) Решевська Катерина Сергіївна, доцент, к.т.н.
(прізвище, ім'я та по-батькові, науковий ступінь, вчене звання)

затверджені наказом ЗНУ від « 29 » 05 2019 року № 812-с

2. Строк подання студентом роботи 27.12.2019

3. Вихідні дані до роботи 1. Постановка задачі.
2. Перелік літератури.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
1. Дослідження предметної області
2. Розробка структури системи та бази даних
3. Розробка програмного забезпечення та тестування його роботи

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)
ілюстрації до тексту роботи, графіки розрахунків, таблиці

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання 09.09.2019

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1.	Розробка плану роботи.	09.09.19	
2.	Збір вихідних даних.	10.09.19	
3.	Обробка методичних та теоретичних	15.09.19	
	джерел.		
4.	Розробка першого розділу.	20.09.19	
5.	Розробка другого розділу.	10.10.19	
6.	Розробка третього розділу.	10.11.19	
7.	Оформлення та нормоконтроль	02.12.19	
	кваліфікаційної роботи.		
8.	Попередній захист кваліфікаційної роботи	17.12.19	

Студент _____ І. В. Гузь
(підпис) (ініціали та прізвище)

Керівник роботи _____ К. С. Решевська
(підпис) (ініціали та прізвище)

Нормоконтроль пройдено

Нормоконтролер _____ О. Г. Спиця
(підпис) (ініціали та прізвище)

РЕФЕРАТ

Кваліфікаційна робота магістра «Розробка технології проведення адаптивного тестування для діагностики рівня підготовленості до ЗНО з математики»: 75 рис., 30 рис., 18 табл., 15 джерел, 1 додаток.

АДАПТИВНІ ТЕСТИ, ЗОВНІШНЄ НЕЗАЛЕЖНЕ ОЦІНЮВАННЯ, КОМП'ЮТЕРНЕ АДАПТИВНЕ ТЕСТУВАННЯ, ТЕСТИ, ITEM RESPONSE THEORY

Об'єкт дослідження – процес якісної перевірки рівня підготовленості до ЗНО з математики.

Мета роботи: розробка технології проведення адаптивного тестування для діагностики рівня підготовленості до ЗНО з математики.

Методи дослідження – аналітичний, синтез-метод, аналітико-синтетичний, практичний, порівняльний.

У кваліфікаційній роботі розглянуто основні теоретичні відомості про зовнішнє незалежне оцінювання, про основні види комп'ютерного тестування та про адаптивні тести, про переваги комп'ютерного тестування над паперовим, про переваги адаптивного тестування над звичайним; також розглянуто алгоритм розробки сайту з адаптивним тестуванням із математики. Результати роботи, можуть бути використанні вчителями загальноосвітніх шкіл та викладачами закладів вищої освіти.

SUMMARY

Master's Qualification Thesis «Development of Adaptive Testing Technology for the Diagnostic of the Preparation Level for Math ZNO»: 75 pages, 30 figures, 18 tables, 15 references, 1 supplement.

ADAPTIVE TESTS, INDEPENDENT EXTERNAL ASSESSMENT, COMPUTER ADAPTIVE TESTING, TESTS, ITEM RESPONSE THEORY

The object of the study – the process is a qualitative review of the level of preparedness for the ZNO in mathematics.

The aim of the study: development of technology of adaptive testing for diagnosing the level of preparedness for the ZNO in mathematics.

The methods of the research are analytical, synthesis, analytical-synthetic and practical, comparative.

In the qualifying paper, the basic theoretical information on Independent External Evaluation, the main types of computer-based testing and adaptive tests, the benefits of computer based test over paper, the advantages of adaptive testing over conventional, the algorithm of development of the site with adaptive testing in mathematics. The results can be used by teachers of secondary schools and lecturers of higher education institutions.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

КАТ – комп’ютерне адаптивне тестування.

КТ – комп’ютерне тестування.

ЗНО – зовнішнє незалежне оцінювання.

IRT – Item Response Theory.

БД – база даних.

ВСТУП

Розширення обсягу змісту освіти в умовах науково-технічної революції кінця XX століття, збільшення варіативності навчальних програм, розробка державних освітніх стандартів і введення незалежної атестації випускників навчальних закладів породжують спектр проблем, умов і вимог, націлених на модернізацію контрольно-оцінюючих систем в освіті. Один з напрямків модернізації пов'язано зі збільшенням масштабів тестування, розвитком його модифікацій з метою навчання, переходом від статичних оцінок рівня навчальних досягнень до динамічних оцінками якості підготовленості учнів, адаптацією контрольно-оцінюючих систем за рахунок негайного реагування на індивідуальні особливості підготовки випробовуваних при пред'явленні завдань, словом, з усіма тими змінами, які багато в чому не сумісні з обмеженими можливостями традиційних тестів.

У зв'язку з цим, на сьогоднішній день в освіті складається ситуація, під впливом якої традиційне тестування, здійснюване за допомогою стандартизованих тестів фіксованою довжини, переростає в сучасні ефективні форми адаптивного тестування, що базуються на відмінних від традиційних теоретико-методологічних основах та інших технологіях конструювання і пред'явлення тестів.

Можна виділити ряд невирішених проблем. Слабо представлені питання класифікації видів адаптивного тестування, а окремі збудовані класифікаційні схеми не відповідають повною мірою вимогам повноти і несуперечності. Не дослідженими виявилися принципи організації адаптивного тестування і його функції, теоретичні питання організації адаптивного навчання та адаптивного тестового контролю. Недостатньо вивчені можливості застосування адаптивного тестування в педагогіці співробітництва. У рішенні ряду теоретичних питань нерідко відсутня системність, в окремих випадках відзначається суперечливість підходів і суджень дослідників.

Пошуки шляхів подолання зазначених проблем, недоліків і недоробок послужили вибору тематики даного дослідження, в якому теоретико-методологічні та технологічні основи адаптивного тестування будуються з опорою на сучасну методологію конструювання тестів, що отримала в працях зарубіжних вчених назву Item Response Theory або скорочено IRT. Основними особливостями дослідження, що зумовили можливість отримання принципово нових результатів, є такі властивості, як інтегративність і міждисциплінарність. Формалізовані за допомогою математичних моделей IRT ідеї адаптивного тестування в дослідженні вплітаються в контекст суміжних з проблематики психолого-педагогічних напрацювань вітчизняних учених і окремі інновації педагогів-практиків особистісно-орієнтованого навчання.

В основу теоретичних результатів було покладено аналіз практичного досвіду по створенню адаптивних контрольно-навчальних програм початку 90-х років, розроблених творчим колективом викладачів Київського інституту інженерів цивільної авіації. В процесі роботи над контрольно-навчальними програмами по ряду розділів курсу вищої математики проявилися окремі недоліки зарубіжних технологій адаптивного тестування, відсутність належного теоретико-методологічного фундаменту і необхідної глибини опрацювання ряду проблем, що і послужило в цілому спонукальним моментом до визначення об’єкта і висунення гіпотези даного дослідження.

Мета дослідження: автоматизація проведення адаптивного тестування абітурієнтів для визначення рівня підготовленості до ЗНО з математики.

Об’єкт дослідження можна визначити як контрольно-оцінюючий процес в освіті, що розглядається в контексті спільної діяльності педагогів і учнів по досягненню запланованих результатів навчання.

Предмет дослідження – теоретико-методологічне та технологічне обґрунтування процесу адаптивного тестування, що забезпечує зростання ефективності функціонування контрольно-оцінюючих систем в освіті. Контекст предмета дослідження досить широкий, оскільки на тлі традиційного для адаптивного тестування процесу підгонки завдань за складністю до рівня

підготовленості учнів в дослідженні розглядаються більш загальні проблеми адаптивного навчання та контролю в плані їх теоретико-методологічного обґрунтування.

Задачі роботи:

– дослідження предметної області адаптивного тестування;

– розробка алгоритму проведення адаптивного тестування з математики;

– вибір програмного забезпечення для реалізації поставленої мети;

– реалізувати програмну частину роботи;

– розробити інструкцію використання готового програмного забезпечення.

Наукова новизна: взагалі, існує багато різних адаптивних тестів у мережі Інтернет з різних предметів, але у зв'язку з тим, що ЗНО з математики стане обов'язковим в Україні, та тим, що математика є одним з найважчих предметів, було обрано саме математичну тематику. Щоб правильно навчити математиці необхідно володіти знанням про те, у яких саме розділах існують пробіли у знаннях, саме тому було обрано адаптивні тести, за допомогою яких можна точніше визначити тематику проблем у знаннях, та за допомогою комп'ютера та комп'ютерних технологій, можна зберегти сили та час для виконання цієї задачі. Виходячи з вищесказаного, було обрано тему розробки технології проведення адаптивного тестування для діагностики рівня підготовленості до ЗНО з математики.

1 ДОСЛІДЖЕННЯ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Історія розвитку адаптивного тестування

Давно було помічено, що важкі завдання виявляються зазвичай абсолютно марними при тестуванні найбільш слабких випробовуваних, разом з тим найлегші завдання не додають ніякої інформації про оцінки тих, хто добре підготовлений до виконання тесту. У зв'язку з цим з'явилася ідея адаптації завдань за складністю до підготовки тестованих і конструювання таким шляхом ефективних тестів.

Першим, хто спробував реалізувати адаптивні стратегії при видачі тесту для оцінки розумових здібностей, був Binet (Binet, Simon, 1905р.). Однак в наступний період, аж до кінця 60-х років, мало хто з практиків, а тим більше теоретиків, звертався до адаптивного тестування, оскільки склалася обстановка в сфері тестування, яка в цілому не сприяла розвитку ідей адаптації в силу обмежених можливостей.

Таким чином, першим періодом розвитку адаптивного тестування можна назвати часовий інтервал з початку XX століття до кінця 60-х років, протягом якого ідеї Binet підтримувалися не тільки вченими – дослідниками, скільки деякими практиками. Останні зазвичай йшли по шляху розбиття групи випробовуваних на частини для представлення окремим підгрупам різних за діапазоном труднощі наборів завдань або за шляхлм індивідуалізації контролю в режимі негайного реагування екзаменатора на результати виконання випробуваним кожного завдання тесту [1].

Відсутність масштабних досліджень в цей період було далеко не випадково, оскільки адаптивне тестування викликало насторожене ставлення і у педагогів, і у фахівців. Навряд чи варто говорити про те, наскільки незвичною для педагогів здавалася ідея оцінки підготовленості випробовуваних

тестуваннями, які відрізняються за складністю, за кількістю і за змістом наборів завдань адаптивних тестів. В результаті у більшості з них складалося цілком з'ясовне уявлення про втрату ряду переваг стандартизованого тестування, що забезпечує порівнянність результатів випробовуваних при виконанні традиційних тестів.

Неприйняття адаптивного тестування тестологами виникало з інших причин, пов'язаних з недоліками класичної теорії тестів. Оскільки в рамках класичної теорії відсутня властивість інваріантності оцінок випробовуваних від складності завдань тестів, то для зіставлення результатів адаптивного тестування доводилося задіяти численні методики вирівнювання, що дозволяють порівняти результати випробовуваних з різних адаптивним тестів. Ускладнення процесу обробки даних тестування супроводжувалося додатковою роботою з аналізу змісту різних наборів завдань, використаних при адаптивних випробуваннях. Внаслідок цього підвищення ефективності в момент тестування оберталося значним збільшенням роботи тестологів з обробки та аналізу результатів виконання адаптивних тестів.

Початок другого періоду розвитку теорії адаптивного тестування можна віднести до 70-х років, коли чільна до цього моменту класична теорія тестів поступово поступилася місцем новій, що отримала в зарубіжній науковій літературі назву Item Response Theory або скорочено IRT. Вирішальний внесок у становлення наукових і практичних робіт другого періоду зробив F.M. Lord, почавши в рамках Educational Testing Service широкомасштабну дослідницьку програму по адаптивному тестуванню на основі наукового апарату IRT. Згодом результати цієї роботи він виклав у своїй монографії 1980, яка зіграла величезну роль у розвитку наукових методів генерації та пред'явлення адаптивних тестів.

У процесі досліджень F.M. Lord зіткнувся з низькою ефективністю традиційних тестів фіксованою довжини при оцінюванні рівня підготовленості найбільш слабких і сильних піддослідних групи (до 17% по краях нормального

розподілу), яка випливає із запланованої на момент створення орієнтації на середній рівень підготовленості учнів або студентів.

1.2 Можливості комп'ютерного тестування

Комп'ютерне тестування (КТ) дає можливість використовувати додаткові можливості при розробці завдань і їх адмініструванні в порівнянні з традиційним бланковим тестуванням. Інноваційні особливості, доступні при КТ включають звук, графіку, анімацію, відео. Причому все це може бути включено і в самі завдання, і в відповідь. Інші інновації стосуються адміністрування завдань.

Наприклад, випробовувані, використовуючи комп'ютер, можуть виділяти текст, клацати мишкою на графіках, пересувати об'єкти по екрану, змінювати порядок елементів або картинок. Далі, з'являється можливість інтерактивного тестування. Наприклад, в процесі відповіді випробуваного на екрані може з'являтися додаткова інформація, різна в залежності від відповіді. З'являється набагато більше можливостей в завданнях з генеруванням відповідей, а не простим вибором правильної відповіді з набору запропонованих [1].

Комп'ютерне тестування може використовуватися в трьох видах:

– комп'ютерне тестування як альтернативна форма пред'явлення тесту (варіанти, а, отже, і порядок пред'явлення завдань фіксовані);

– комп'ютерне тестування з автоматичним формуванням різних варіантів тесту (варіанти формуються автоматично з наявного набору завдань за правилами, заданими розробником);

– комп'ютерне адаптивне тестування (для кожного випробуваного в процесі тестування формується індивідуальний набір завдань).

Відмінні риси КАТ в порівнянні з іншими формами тестування:

– кожен випробовуваний отримує свій власний набір завдань, тому і зміст, і довжина тесту можуть відрізнятися для різних випробуваних;

– кожен випробовуваний оцінюється індивідуально (на своєму рівні) з мінімальною помилкою вимірювання.

Основні переваги КАТ в порівнянні з іншими формами тестування:

– ефективність: потрібно істотно менше завдань для оцінювання рівня підготовленості випробуваного;

– точність: можливість оцінити рівень підготовленості кожного випробуваного на його рівні з мінімальною помилкою вимірювання;

– випробувані не витрачають час і сили на завдання, що не відповідають їх рівню підготовки (занадто легкі для них або занадто важкі), тому зменшується вплив на результати додаткових факторів (стомлення, занепокоєння, неакуратність);

– добре розвинена теорія КАТ, тому добре розроблений комп'ютерний адаптивний тест надійніший;

– учасники тестування більш мотивовані і спокійні (тому що їм не пропонується завдань, занадто для них важких).

Класична теорія тестування не підходить для адаптивного тестування: саме поняття первинного бала при використанні технологій КАТ не має сенсу, так як кількість завдань, які пропонуються різними випробуваним, по-різному. Таким чином, вся теорія КАТ базується на сучасній теорії тестування IRT. Необхідною умовою проведення КАТ є наявність досить великого банку якісних завдань, створення якого можливо тільки в рамках IRT [15].

Процес тестування у КАТ виглядає наступним чином. Випробуваному пропонується якесь завдання. Якщо він відповів на нього правильно, тоді йому пропонується більш важке завдання. У разі неправильної відповіді на перше завдання, пропонується більш легке завдання.

Після кожної відповіді рівень підготовленості випробуваного переоцінюється, і вибирається наступне завдання, яке найбільш підходе до його рівня. Процес завершується, коли досягнута необхідна точність оцінювання рівня підготовленості (але можуть використовуватися інші правила закінчення тестування).

Основні проблеми, які потребують вирішення при розробці алгоритму КАТ, такі:

а) як почати тестування, тобто, як вибрати перше завдання для даного випробуваного;

б) як продовжити тестування, тобто, як після кожної відповіді вибирати наступне завдання;

в) як закінчити тестування, тобто, коли процес тестування можна вважати завершеним.

Різні технології комп'ютерного адаптивного тестування розрізняються підходами до вирішення поставлених проблем, тобто вони розрізняються по процедурам вибору завдань, методам оцінювання випробовуваних, правилам закінчення тестування і т.д [14].

Психологічні особливості адаптивного тестування:

– в основу проведення тестування покладено принцип чіткого розуміння випробуваними, що необхідно робити;

– адаптивний тест може визначити рівень знань тестованого з допомогою меншої кількості питань. При виконанні одного і того ж адаптивного тесту тестовані з високим рівнем підготовки і тестовані з низьким рівнем підготовки побачать абсолютно різні набори питань: перший побачить більше число складних питань, а останній – легких. Частки правильних відповідей у обох можуть збігатися, але так як перший відповідав на більш складні питання, то він набере більшу кількість балів;

– тестований проходить тест спокійно, ніхто сторонній його не підганяє. Система автоматично вимкнеться, як тільки закінчиться час тестування;

– після проходження тесту тестований може попросити індивідуальної бесіди з викладачем. Викладач, в свою чергу, може дати свою відповідь на основі статистики проходження студентом тесту. Зрозуміло, тестування не замінює і не відміняє традиційних форм педагогічного контролю, заснованих на безпосередньому спілкуванні викладача зі студентом. Такий контроль виконує

важливі навчальні функції, озброює викладачів інформацією про рівень знань учнів.

Однак традиційні форми педагогічного контролю носять багато в чому суб'єктивний характер і не дозволяють отримати порівняльні дані, необхідні для управління процесом освіти.

1.3 Процедура вибору завдань

Правило вибору першого завдання визначається метою тестування. Для критеріально-орієнтованого тестування перше завдання зазвичай вибирається близько до граничного значення (прохідного балу). Для нормативно-орієнтованого тестування перше завдання зазвичай вибирається середньої складності [12].

Використання IRT дає можливість використовувати різні процедури вибору завдань. Найбільшого поширення при нормативно-орієнтованому тестуванні отримав метод максималізації інформації, при якому кожне наступне завдання підбирається з банку як найбільш інформативне для оцінювання рівня підготовленості даного тестованого. В критеріально-орієнтованому тестуванні наступне завдання вибирається трохи більш важким (при правильній відповіді) або трохи легшим (при неправильній відповіді) [13].

При комп'ютерному адаптивному тестуванні можуть використовуватися різні правила закінчення тестування. Найбільш поширене правило, засноване на досягненні необхідної точності вимірювань: процес тестування кожного випробуваного закінчується, коли досягнута точність вимірювання його рівня підготовленості стає менше наперед заданого значення. Іноді використовується інші правила зупинки: фіксований час тестування, фіксована кількість завдань і т.д. Часто поєднуються два підходи.

Наприклад, встановлюється, що максимальне число завдань дорівнює 100, але тестування може бути припинена раніше, якщо необхідна точність буде досягнута [2].

1.4 Моделі адаптивного тестування

При моделюванні відповідей нині найбільш розвинений аналіз IRT теорії, яка використовує для моделювання вірогідності правильних відповідей логістичну криву. Проведений порівняльний аналіз логістичного і нормального розподілів. Показано, що, розглядаючи логістичний розподіл дуже добре апроксимується нормальним. У свою чергу нормальний закон є граничним випадком біномного розподілу. Цей факт можна формально інтерпретувати так, що "рівень знань" є долею вирішених завдань, оскільки число вирішених із загального числа завдань при заданій вірогідності рішення підпорядковане біномному розподілу [11].

Далі передбачається, що складність завдання задана деяким числовим значенням, і в результаті виконана формалізація процесу тестування у вигляді марківського ланцюга, в якому вірогідність переходів по складнощам визначаються на підставі логістичної кривої. Передбачається, що відповіді на завдання – незалежні величини. Тому використовується однорідний марківський ланцюг, де станами ланцюга є заходи складності завдань. Показано, що для побудованого ланцюга існує єдине, незалежне від початкового стану, стаціонарний розподіл. Знайдено аналітичне рішення стаціонарної вірогідності [10].

Збільшуючи дискретизацію складності, тобто збільшуючи кількість станів марківського ланцюга показана збіжність до безперервного розподілу. Знайдений граничний розподіл, який використовується для візуалізації перетворень "складності" в "знання". На практиці найбільш природні випадки, коли оцінки мають постійну дисперсію або постійний коефіцієнт варіації.

Постійний коефіцієнт варіації пояснюється збільшенням невизначеності при зростанні "рівня знань". Постійна дисперсія може використовуватися, коли зміна рівня знань невелика. Для постійної дисперсії показано, що перетворення носить експоненціальний характер. Експоненціальна функція монотонна і великим значенням функції "рівень знань" відповідають великі значення щільності розподілу "складності" вирішуваної задачі. Відповідно максимум щільності доводиться на максимум цільовій функції. Для постійного коефіцієнта варіації (g) показано, що перетворення описується статечною функцією, а при g=1 функція щільності вірогідності з точністю до постійного множника на усій області визначення співпадає з середнім значенням функціонала. Таким чином, якщо є міра "складності" завдання, то визначена і міра "рівня знань" і вона співпадає з щільністю розподілу адаптивного алгоритму тестування [3].

Математична модель адаптивного контролю знань визначає рівень навченості учнів залежно від важкості завдань. Теоретичною основою адаптивного контролю є теорія IRT у поєднанні з дидактичним принципом індивідуалізації навчання. Інший підхід до створення педагогічних тестів і до інтерпретації результатів їх виконання представлений в так званій сучасній теорії педагогічних вимірів Item Response Theory (IRT), що отримала широкий розвиток в 60-й, – 80-і роки у ряді західних країн. До досліджень останніх років в цьому напрямі відносяться праці В. С. Аванесова, В. П. Беспалько, Л. В. Макарової, В. І. Міхеєва, Б. У. Родіонова, А. О. Татура, В. С. Черепанова, Д. В. Люсіна, М. Б. Челишкової, Т. Н. Родигіної. Е. Н. Лебедєвої та ін [4].

До найбільш значимих переваг IRT відносять вимір значень параметрів випробовуваних і завдань тесту в одній і тій же шкалі, що дозволяє співвіднести рівень знань будь-якого випробовуваного з мірою трудності кожного завдання тесту. Саме на цій властивості оцінок параметрів випробовуваних і завдань грунтована організація сучасного адаптивного контролю знань. Критики тестів інтуїтивно усвідомлювали неможливість точного виміру знань випробовуваних різного рівня підготовки за допомогою

одного і того ж тіста. Це одна з причин того, що в практиці прагнули зазвичай створювати тести, розраховані на вимір знань випробовуваних найчисленнішого, середнього рівня підготовленості. Природно, що при такій орієнтації тесту знання у сильних і слабких випробовуваних вимірювалися з меншою точністю [8].

Обробка результатів тестування по IRT: визначається групова адаптивність, на відповідність середнього логіта трудності завдань тесту Вср. і середнього логіта навченої випробовуваних $Qcp.$ за виразом:

$$\text{Агр} = 1 - [Qcp. - \text{Вср.}]$$

Групова адаптивність $\text{Агр.} = 1$ при ідеальній відповідності Вср. і $Qcp.$ Результати попереднього тестування групи учнів мають значення Агр. далеко не рівними одиниці, тому наступним кроком є зміна значення групової адаптивності шляхом виключення з тесту "непрацюючих" завдань в цій групі тестованих з $Bj = 0$ і визначення рівня навченості кожного випробовуваного, а також отримання в цьому випадку індивідуальних характеристичних кривих випробовуваних. Надалі визначається істинний бал як сума усієї вірогідності відповідей кожного випробовуваного на кожне завдання тесту і оцінюється рівень знань.

Таким чином, з’являється можливість для цього випробовуваного вибирати той, що відповідає його рівню набір тестових завдань і випробовувані можуть бути протестовані тестами складеними індивідуально для них. Для групи тестованих створюються адаптивні тести, що мають різну довжину і час виконання для сильних, слабких і таких, що середніх, що навчаються у цій групі. Такий процес вимагає комп’ютерної технології створення, зберігання тестів, проведення тестування і обробки результатів тестування [5].

По суті, ця технологія дає початок новій організації як тестового контролю знань, так і учбового процесу в цілому, на більш високому науковому рівні.

В умовах масової освіти адаптивне навчання дає можливість ефективній практичній реалізації принципу індивідуалізації навчання [9].

Застосування адаптованих тестових вимірників дозволить підвищити об’єктивність оцінки рівня знань випробовуваних з використанням меншої кількості тестових завдань в порівнянні із звичайним тестуванням за рахунок зниження похибки виміру у вибірці учнів, адаптивній пропонованим труднощам тестових завдань.

1.5 Тестування по методу ланцюжків питань

Автоматизована система контролю знань пропонує викладачеві використати систему ланцюжка питань, коли декілька питань об’єднуються у фіксовану послідовність (ланцюжок) за деякою смисловою ознакою, визначуваною викладачем, а кожному питанню в ланцюжку привласнюється деякий коефіцієнт важливості цього питання в цьому ланцюжку. Цей коефіцієнт змінюється від 0 і сума коефіцієнтів питань в ланцюжку приймається рівною 1. Сенс коефіцієнта розкривається при обробці результатів тестування: оцінка за відповіді на питання, об’єднані в ланцюжок, виставляється залежно від важливості питань, на які були дані правильні відповіді. Ланцюжок може містити необмежене число питань, об’єднаних по семантичній ознаці усередині вибраної теми тестування. Виродженим випадком структури ланцюжка є наявність в ланцюжку усього лише одного питання. В цьому випадку коефіцієнт його важливості, очевидно, встановлюється рівним одиниці [6].

1.6 Метод адаптивного автоматизованого тестування знань

Адаптивним тестуванням знань називатимемо спосіб екзаменаційного контролю рівня підготовки навчаного, при якому процедура вибору і пред'явлення йому чергового тестового завдання на $(t+1)$ -му кроці тестування визначається відповідями навчаного на попередніх t кроках тесту. Математичну основу такого обліку складає модель об'єднання тестових завдань в тематичні послідовності із зваженим ранжируванням як окремих завдань, так і цілих послідовностей і виведенням підсумкової оцінки за тест з урахуванням нормованої суми балів, що накопичується за вибрані навчаним варіанти відповідей.

1.7 Висновки за 1 розділом

Дивлячись на історичний огляд адаптивного тестування, можна побачити, що розвиток адаптивного тестування почався ще давно, але розвиток науки та техніки вже давно випередив ці методи і тому, слід впроваджувати вже не просто адаптивне тестування, а саме комп'ютерне адаптивне тестування.

Комп'ютерне адаптивне тестування розвинуто доволі такі сильно і є найбільш суттєвим у навчанні. Саме тому було обрано найважчу та найважливішу з дисциплін теперішнього часу – математику, а саме підготовку учнів до ЗНО з математики, шляхом застосування адаптивного тестування.

2 РОЗРОБКА СТРУКТУРИ СИСТЕМИ ТА БАЗИ ДАНИХ

2.1 Формалізація задач системи тестування

Дослідження предметної області дозволило виявити акторів та прецеденти, розроблюваної системи (рис. 2.1):

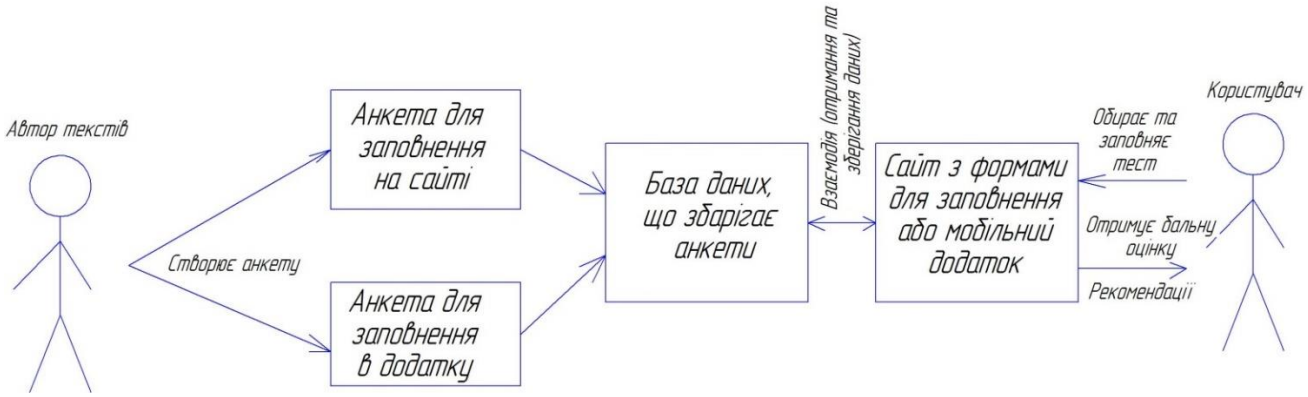

Рисунок 2.1 – Алгоритм роботи системи у вигляді UML діаграми

Задачі, які виконує програмне забезпечення з проведення адаптивного тестування:

а) автор тестів складає тест, де вказує для кожного питання рівень складності та кількість балів, що отримує користувач, якщо надасть правильну відповідь;

б) створений тест може бути призначений лише для тестування на сайті або для тестування на мобільному додатку;

в) розроблений тест зберігається у базі даних на сервері;

г) користувач обирає тест з рівнем складності який він бажає або проходить тест який йому система обере самостійно;

д) в залежності від того наскільки правильно користувач відповів на поставлені питання він отримує за кожне з них бальну оцінку;

е) бальні оцінки сумуються та видаються користувачеві у вигляді загальної оцінки;

ж) додатково системою в залежності від кількості набраних балів за кожною з тем надаються рекомендації щодо необхідності додаткового про опрацювання якихось окремих тем, за якими він отримав мінімальні бали [7].

2.2 Проектування бази даних збору інформації

Оскільки у нас в онлайн тестуванні, згідно розробленої структури, реалізується додатковий функціонал, що покликаний проводити автоматичне тестування знань учнів з математики, який реалізовується на PHP скриптах, додатково створюються таблиці бази даних завдання яких містити інформацію про тести, а також результати їх заповнення учнями з можливістю перевірки правильності заповнення.

Покажемо ER-діаграму з’єднань таблиць бази даних (рис. 2.2).

Дані таблиці наведені нижче (табл. 2.1-2.11).

Таблиця 2.1 – Структура таблиці БД Autors (облікові дані вчителя)

№	Назва стовбця	Тип даних	Опис даних, що зберігаються	На які таблиці посилається
1	Id	INT	Первинний ключ	
2	_login	varchar(50)	Логін користувача	
3	password_md5hash	varbinary(max)	Хеш паролю	
4	access_level	INT	Рівень доступу	
5	first_name	varchar(50)	Ім’я автора (вчителя)	
6	last_name	varchar(50)	Прізвище автора (вчителя)	

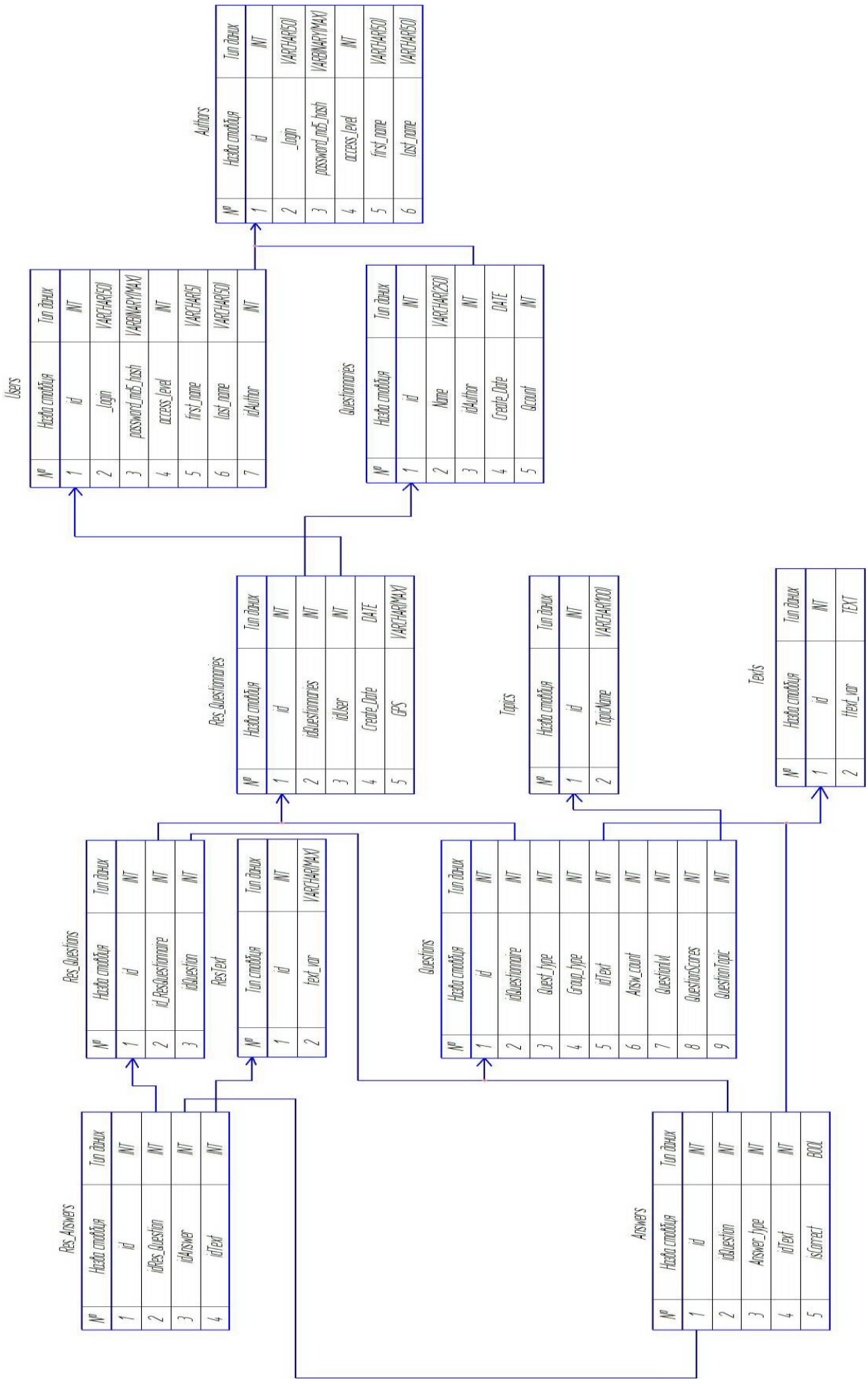

Рисунок 2.2 – ER-діаграма з’єднань таблиць бази даних

Стовбець id зберігає унікальний цілочисельний ідентифікатор запису.

Стовбець _login зберігає ім’я користувача (автора/вчителя) довжиною до 50 символів (бажано, для запобігання не правильному кодуванню, задавати значення латинськими буквами, можливо з використанням чисел).

Стовбець password_md5hash зберігає хеш паролю користувача (автора/вчителя). Використовується саме хеш, а не пароль у чистому вигляді, оскільки це дає змогу захистити пароль від прямого перегляду.

Паролі користувачів саме хешируються, а не шифруються з наступних причин:

а) трудомісткість. Шифрування може тривати довше, а яке перетворення ми б не вибрали, його доведеться пробробляти при кожній перевірці пароля. Однією з вимог до хеш-функцій ж є швидкість виконання;

б) довжина вихідних значень. Результат шифрування має змінну довжину, результат хешування – завжди однакову, а зберігати однорідні за розміром дані в базі даних дуже вже зручно. Не кажучи вже про те, що довжина пароля в зашифрованому вигляді буде давати деяку інформацію про довжину вихідного пароля. Однакова довжина, правда, призводить до можливості виникнення колізій, але про це нижче;

в) управління ключами. Для шифрування потрібно ключ, який теж десь доведеться зберігати і сподіватися, що його ніхто не знайде. У будь-якому випадку, генерація і управління ключами це окрема історія (вони не повинні бути слабкими, їх потрібно регулярно міняти і так далі).

Hash – хеш функція – функція однозначного відображення рядка (будь-якої довжини) на кінцеву множину (рядок заданої довжини).

Саме число (рядок) хеш – результат обчислення хеш-функції над даними.

Існують криптографічні та некриптографічні (класифікуються окремо, до них відносяться, наприклад, контрольні суми) хеш-функції.

Для криптографічних хеш є три додаткових умови, які відрізняють їх від всіх інших:

а) незворотність: для заданого значення хеш-функції m повинно бути обчислювально нездійсненно знайти блок даних X, для якого H (X) = m;

б) стійкість до колізій першого роду: для заданого повідомлення M має бути обчислювально нездійсненно підібрати інше повідомлення N, для якого H (N) = H (M);

в) стійкість до колізій другого роду: має бути обчислювально нездійсненно підібрати пару повідомлень ~ (M, M ‘), що мають однаковий хеш

Стовбець access_level – зберігає рівень доступу користувача до редагування анкет (учень, вчитель, адміністратор системи).

Стовбець first_name – власне ім’я автора/вчителя до 50 символів (не обов’язково співпадає з логіном).

Стовбець last_name – прізвище автора/вчителя до 50 символів.

Таблиця 2.2 – Структура таблиці БД Users (облікові данні учня)

№	Назва стовбця	Тип даних	Опис даних, що зберігаються	На які таблиці посилається
1	Id	INT	Первинний ключ	
2	_login	varchar(50)	Логін користувача	
3	password_md5hash	varbinary(max)	Хеш паролю	
4	access_level	INT	Рівень доступу	
5	first_name	varchar(max)	Ім’я учня	
6	last_name	varchar(max)	Прізвище учня	
7	idAutor	INT	id автора (вчителя)	Autors.id

Стовбці цієї таблиці майже повністю співпадають з попередньою з різницею в тому, що тут зберігаються дані щодо учнів та вони відносяться до свого окремого вчителя, посилання на якого зберігається у стовбці idAutor.

Таблиця 2.3 – Структура таблиці БД Questionnaires (Дані об анкетах)

№	Назва стовбця	Тип даних	Опис даних, що зберігаються	На які таблиці посилається
1	Id	INT	Первинний ключ	
2	Name	varchar(250)	Назва анкети	
3	idAutor	INT	id автора	Autors.id
4	Create_Date	date	дата та час створення	
5	Qcount	INT	Максимальна кількість заповнених анкет	

Стовбець id зберігає унікальний цілочисельний ідентифікатор запису.

Стовбець Name – назва анкети (до 250 символів). Назва анкети може

Стовбець idAutor – числове посилання на автора анкети/вчителя, що створив цю анкету.

Стовбець Create_Date – зберігає дату та час створення анкети.

Стовбець Qcount – Зберігає максимальну кількість анкет яку може заповнити один учень для того щоб кількість спроб не була нескінченною і можна було б адекватно оцінити швидкість навчання. Можна також проводити оцінювання не лише на основі останньої спроби, а й враховувати кількість спроб для отримання позитивного результату.

Таблиця 2.4 – Структура таблиці БД Res_Questionnaires (Дані стосовно заповненої анкети)

№	Назва стовбця	Тип даних	Опис даних, що зберігаються	На які таблиці посилається
1	Id	INT	Первинний ключ	
2	idQuestionnaires	INT	id Анкети	Questionnaires (id)
3	idUser	INT	id учня	Users (id)

Продовження таблиці 2.4

4	Create_Date	smalldatetime	Дата та час заповнення анкети	
5	GPS	varchar(max)	Координати GPS	

Стовбець id зберігає унікальний цілочисельний ідентифікатор запису. Кількість записів таблиці відповідає кількості повністю заповнених анкет.

Стовбець idQuestionnaires зберігає посилання на ідентифікатор анкети яку було заповнено.

Стовбець idUser зберігає посилання на ідентифікатор користувача (учня), який заповнив анкету.

Стовбець Create_Date зберігає дату та час закінчення заповнення анкети

Стовбець GPS зберігає координати де було заповнено анкету, для виявлення, можливості заповнення не на території окремо відведених даних.

Таблиця 2.5 – Структура таблиці БД Texts (тексти питань та відповідей)

№	Назва стовбця	Тип даних	Опис даних, що зберігаються	На які таблиці посилається
1	Id	INT	Первинний ключ	
2	text_var	text	Текст	

Стовбець text_var зберігає всі тексти анкети. Тексти винесені в окрему таблицю, а в питаннях та відповідях на тексти зроблено лише посилання оскільки тексти можуть повторюватись.

Стовбець idQuestionnaires зберігає посилання на ідентифікатор анкети до якої відноситься це питання.

Стовбець Quest_type зберігає числовий тип питання. Питання можуть розділятися на питання за деякою кількістю фіксованих відповідей (індекс 0) та питання з відкритою відповіддю (індекс 1).

Таблиця 2.6 – Структура таблиці БД Questions (Дані о питаннях)

№	Назва стовбця	Тип даних	Опис даних, що зберігаються	На які таблиці посилається
1	id	INT	Первинний ключ	
2	idQuestionnaire	INT	id анкети	Questionnaires (id)
3	Quest_type	INT	Тип питання 0 – питання з фіксованою відповіддю, 1 – питання зі своїм варіантом відповіді	
4	Group_type	INT	Тип групування 0 – звичайне питання, 1 – стать, 2 – вік, 3 – територія	
5	idText	INT	id текста	Texts (id)
6	Answ_count	INT	Кількість одночасних відповідей в питанні	
7	Questionlvl	INT	Рівень складності питання	
8	QuestionScores	INT	Кількість балів за правильну відповідь на це питання	
9	QuestionTopic	INT	Посилання на тему до якої відноситься відповідь	Topics(id)

Стовбець Group_type зберігає номер типу групи питання. Питання можуть бути характеристиками статі та віку учнів, або території та класу де проводиться опитування – у цьому випадку ці питання будуть впливати на загальну статистику але не будуть впливати на оцінку за тестування.

Стовбець idText зберігає індекс тексту питання. Сам текст зберігається в таблиці Texts.

Стовбець Answ_count зберігає кількість відповідей, що можуть бути обрані одночасно. Звісно ця кількість не має перевищувати загальну кількість відповідей у питанні і не може бути менша за одиницю.

Стовбець Questionlvl зберігає рівень складності питання, що дозволить при встановленні різного рівня складності користувачем обирати той рівень, що підходить для нього.

Стовбець QuestionScores зберігає кількість балів, яку отримає користувач якщо надасть правильну відповідь на це питання.

Стовбець QuestionTopic зберігає посилання на тему до якої належить питання для того щоб надавати користувачеві рекомендації щодо підготовки по темах по яких він набрав найнижчу кількість балів.

Таблиця 2.7 – Структура таблиці БД Res_Questions (питання, що заповнені в заповнених анкетах)

№	Назва стовбця	Тип даних	Опис даних, що зберігаються	На які таблиці посилається
1	Id	INT	Первинний ключ	
2	idRes_Questionnaire	INT	Посилання на анкету, що заповнена	Res_Questionnaires (id)
3	idQuestion	INT	Посилання на оригінальне питання	Questions(id)

Ця таблиця (табл. 2.7) зберігає результати заповнення питання з індексом idQuestion для заповненої анкети з індексом idRes_Questionnaire.

Таблиця 2.8 зберігає дані про відповіді.

Стовбець idQuestion зберігає індекс питання до якого відноситься ця відповідь.

Стовбець answer_type зберігає номер типу відповіді. Дорівнює 0 якщо відповідь з фіксованим текстом, що надається вчителем, та дорівнює 1 якщо текст відповіді вводиться учнем самостійно.

Стовбець idText зберігає індекс тексту відповіді. Сам текст зберігається в таблиці Texts.

Стовбець isCorrect зберігає ознаку чи є ця відповідь вірною для питання до якого вона належить. Якщо користувач обирає саме цю відповідь він отримує ту кількість балів, що вказана у відповідному полі питання.

Таблиця 2.8 – Структура таблиці БД Answers (Дані про відповіді)

№	Назва стовбця	Тип даних	Опис даних, що зберігаються	На які таблиці посилається
1	Id	INT	Первинний ключ	
2	idQuestion	INT	Посилання на питання	Questions(id)
3	answer_type	INT	Тип відповіді. 0 –фіксована відповідь 1 – відкрите питання, відповідь вводиться самостійно	
4	idText	INT	Посилання на текст	Texts (id)
5	isCorrect	BOOL	Ознака, що ця відповідь є вірною для відповідного питання	

Таблиця ResTexts (табл.2.9) відповідає за зберігання власних відповідей коли відповіді вводяться самостійно учнем.

Таблиця Res_Answers (табл.2.10) зберігає посилання на заповнене питання idRes_Question заповненої анкети з вказанням вибраного варіанту фіксованої відповіді idAnswer, або посилання на текст власної відповіді idText.

Таблиця 2.9 – Структура таблиці БД ResTexts (зберігання власних варіантів відповідей)

№	Назва стовбця	Тип даних	Опис даних, що зберігаються	На які таблиці посилається
1	Id	INT	Первинний ключ	
2	text_var	varchar(max)	Текст	

Таблиця 2.10 – Структура таблиці БД Res_Answers (відповіді на питання в заповненій анкеті)

№	Назва стовбця	Тип даних	Опис даних, що зберігаються	На які таблиці посилається
1	Id	INT	Первинний ключ	
2	idRes_Question	INT	Посилання на питання в заповнений анкеті	Res_Questions(id)
3	idAnswer	INT	Посилання на оригінальну відповідь у анкеті	Answers(id)
4	idText	INT	id тексту в таблиці ResTexts, якщо відповідь маж власний текст, якщо ні – то null	ResTexts(id)

Таблиця 2.11 – Структура таблиці БД Topics (теми до яких відносяться питання)

№	Назва стовбця	Тип даних	Опис даних, що зберігаються	На які таблиці посилається
1	Id	INT	Первинний ключ	
2	TopicName	VARCAR (100)	Назва теми	

2.3 Висновки за 2 розділом

У другому розділі показано структуру системи та бази даних, показано ER та UML діаграми. Показано задачі, які виконує програмне забезпечення з проведення адаптивного тестування

3 РОЗРОБКА ПРОГРАМНОГО ЗАПЕЗПЕЧЕННЯ ТА ТЕСТУВАННЯ ЙОГО РОБОТИ

3.1 Вибір хостингу, домену, установка та налаштування CMS

Реєстрація хостингу та домену

При створенні сайту першим кроком є вибір хостингу для розміщення файлів даних та бази даних та вибір доменного ім’я (адресу сторінки) сайту.

Хостинг (англ. Hosting) – послуга з надання ресурсів для розміщення інформації на сервері, що постійно перебуває в мережі (зазвичай Інтернет). Зазвичай хостинг входить в пакет з обслуговування сайту і має на увазі як мінімум послугу розміщення файлів сайту на сервері, на якому запущене ПЗ, необхідне для обробки запитів до цих файлів (веб-сервер). Як правило, в обслуговування вже входить надання місця для поштової кореспонденції, баз даних, DNS, файлового сховища на спеціально виділеному файл-сервері і т. П., А також підтримка функціонування відповідних сервісів.

Доменне ім’я – символьне ім’я, що служить для ідентифікації областей, які є одиницями адміністративної автономії в мережі Інтернет, в складі вищестоящої по ієрархії такої області. Кожна з таких областей називається доменом. Спільний простір імен Інтернету функціонує завдяки DNS – системі доменних імен. Доменні імена дають можливість адресації інтернет-вузлів і розташованим на них мережевих ресурсів (веб-сайтів, серверів електронної пошти, інших служб) бути представленими в зручній для людини формі.

В якості хостінг провайдера було обрано сервіс Ukraine.com.ua (рис.3.1) оскільки це український надійний сервіс з багаторічною практикою перевірений на протязі великої кількості часу та який має помірні ціни в порівнянні з закордонними сервісами такого рівня.

Після вибору хостингу необхідно обрати тарифний план, що підійде для розміщення сайту навчального курсу.

Оскільки навантаження на сайт навчального курсу буде не дуже велике та кількість матеріалу навчального курсу не дуже велика – було обрано самий простий тариф представлений на сервері хостингу (рис. 3.1). Покажемо параметри обраного тарифного плану (табл. 3.1).

Таблиця 3.1 – Параметри обраного тарифного плану хостингу

Параметр	Значення	Параметр	Значення
Загальні параметри		PHP	
Місце на SSD диску	3ГБ	Версії, що підтримуються	4.4–7.4
SSL сертифікат	Безкоштовний	Повноцінний режим PHP без safe mode	Присутній
Сайтів	1	Підтримка CURL	Присутня
Ліміт оперативної пам’яті	512МБ	Zend Optimizer	Присутній
Кількість субдоменів	Необмежено	ionCube Loader	Присутній
MySQL		Phalcon	2,3
Кількість баз даних	необмежено	Memcache	Присутній
Створення тригерів, view, процедур і т.д.	Присутній	Redis	Присутній
Віддалене підключення до MySQL	Присутнє	Підтримка HTTP/2	Присутня
Доступ к phpMyAdmin	Присутнє		

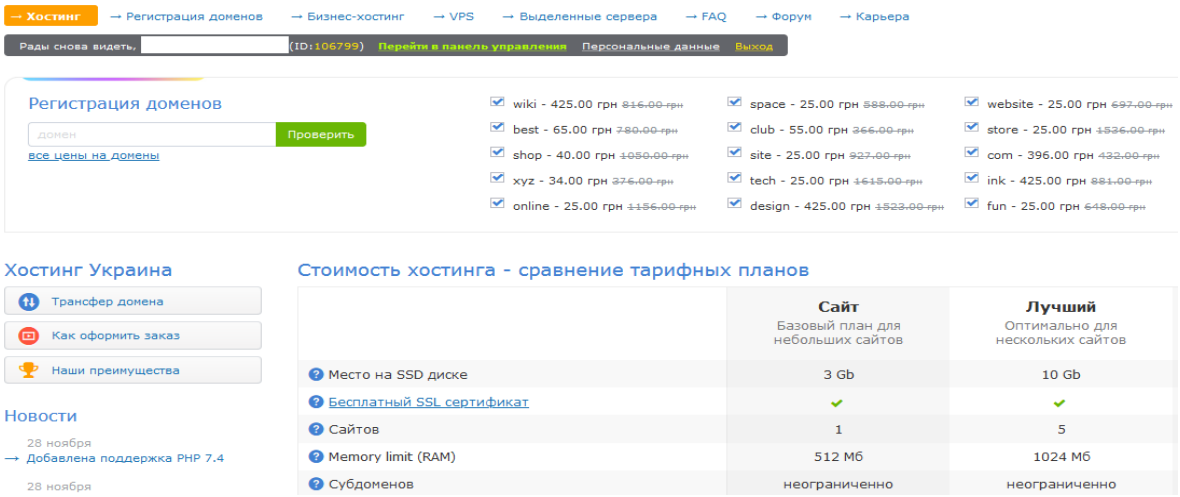

Рисунок 3.1 – Вибір тарифного плану на хостингу

Після вибору тарифного плану хостингу обираємо доменне ім’я сайту. Обираємо ім’я mathtest.site оскільки домен другого рівня mathtest характеризує наявність тестуванню з математики, а домен першого рівня site характеризує, що це буде сайт з цього приводу та це ім’я має невелику щорічну оплату.

Наступним етапом є встановлення системи керування контентом (CMS Drupal) на сервері.

Для встановлення на сервері CMS Drupal в нашому випадку можна використати 2 способи:

– встановлення з офіційного сайту drupal.org;

– встановлення інструментами хостингу.

Після встановлення і прив’язки порожня база даних має структуру приведену в файлі 0.xml

По суті при виклику локальної адреси для установки Drupal відбувається виконання на сервері PHP коду який має наступний вигляд:

<? Php

define (‘DRUPAL_ROOT’, getcwd ()); – отримуємо ім’я поточного робочого каталогу і присвоюємо його в якості значення константи DRUPAL_ROOT тобто по суті кореневого каталогу Друпал

require_once DRUPAL_ROOT. ‘/Includes/bootstrap.inc’; – включаємо і виконуємо файл bootstrap.inc в якому задаються константи і функції (bootstrap

перекладається як початкова завантаження). Файл bootstrap.inc з російськими коментарями представлений в папці зі зміненими інсталянтамі.

drupal_bootstrap (DRUPAL_BOOTSTRAP_FULL); – виклик функції з підключеного файлу

menu_execute_active_handler ();

Після того як CMS встановлено ми отримаємо базову структуру сайту. Наступним етапом є створення зовнішнього вигляду кінцевого сайту, що включає в себе наступні операції (рис. 3.2):

– вибір мови користувача;
– розробка назви та слогана (гасла);
– розробка логотипу;
– розробка зовнішнього оформлення сайту;
– розробка меню.

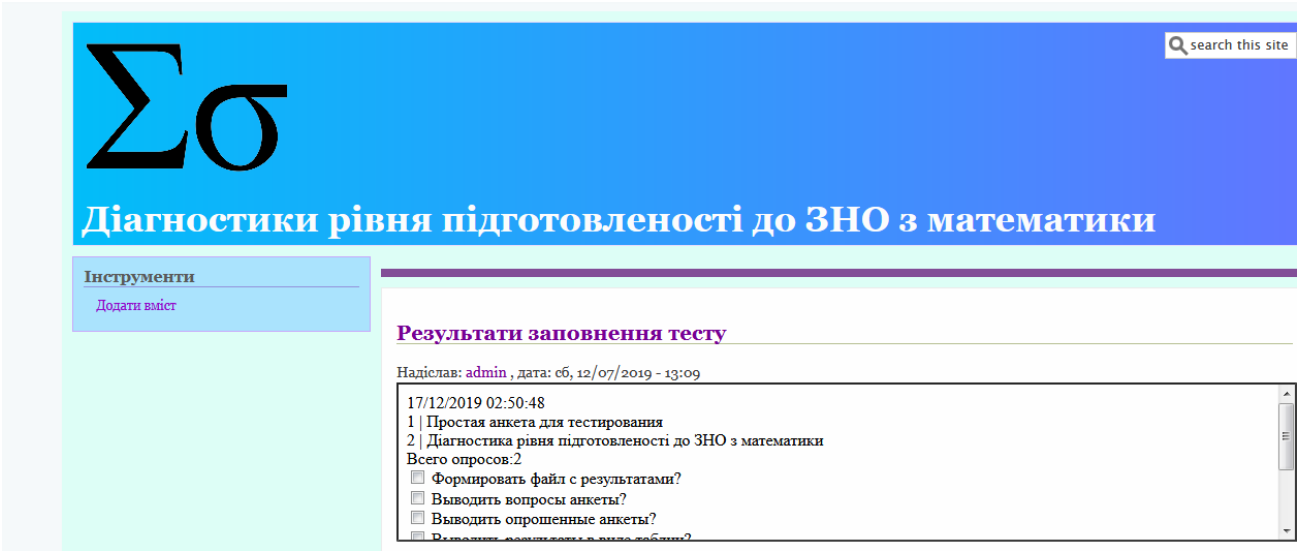

Рисунок 3.2 – Зовнішній вигляд сайту після встановлення CMS

Для роботи з різними мовами користувача CMS Drupal має власні готові інструменти. Для їх підключення необхідно увійти на сайт під адміністраторським паролем, обрати пункт розширення та встановити галочки напроти компонентів в області «Multilingual» (рис. 3.3).

Рисунок 3.3 – Активація модулів для роботи з різними мовами оформлення

Наступним кроком для активації мов користувача є встановлення додаткових мов в меню «Конфігурація» сайту в розділі «Мова та переклад матеріалу» (рис. 3.4).

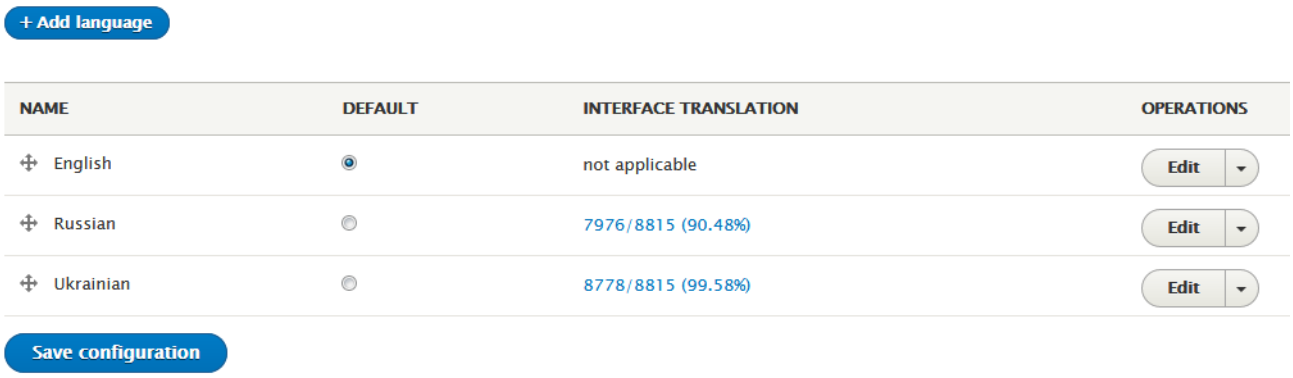

Рисунок 3.4 – Встановлення та підключення російської та української мови для оформлення сайту

Після встановлення та вибору у якості мови за замовчуванням української мови всі адміністративні пункти меню автоматично будуть переведені.

Наступним кроком є зміна назви сайту, що відображається в заголовку закладки браузера та в верхній частині сайту в залежності від його оформлення. Також це основний тег сайту. Для зміни назви та гасла сайту обираємо пункт «Основні налаштування сайту» в пункті адміністративного меню «Конфігурація» та вводимо у нашому випадку назву «Діагностика рівня підготовленості к ЗНО з математики» та гасло «Будемо вступати в ВНЗ разом!».

Оскільки курс, що нами розробляється, є авторським та унікальним – створення логотипу зробимо також самостійно та не будемо завантажувати готовий з інтернету.

3.2 Розробка графічного інтерфейсу системи тестування

Для роботи онлайн тестувань зі знань математики треба створити відповідні меню для тих хто проходить тестування та для авторів тестів де має бути присутній конструктор анкет.

Меню користувача буде складатися з наступних пунктів:

- пройти тестування;
- обрати рівень складності;
- результати минулих тестувань;
- налаштування облікового запису.

Меню автора тестів буде складатись з наступних пунктів:

- створити новий онлайн тест;
- створити тест для мобільного додатку;
- керування існуючими тестами;
- перегляд результатів тестування.
- додати нового користувача (учня).

Для створення меню в CMS Drupal обираємо пункт «Структура» – «Меню» та натискаємо кнопку «+Додати меню», потім заповнюємо поля як в прикладі (рис. 3.5).

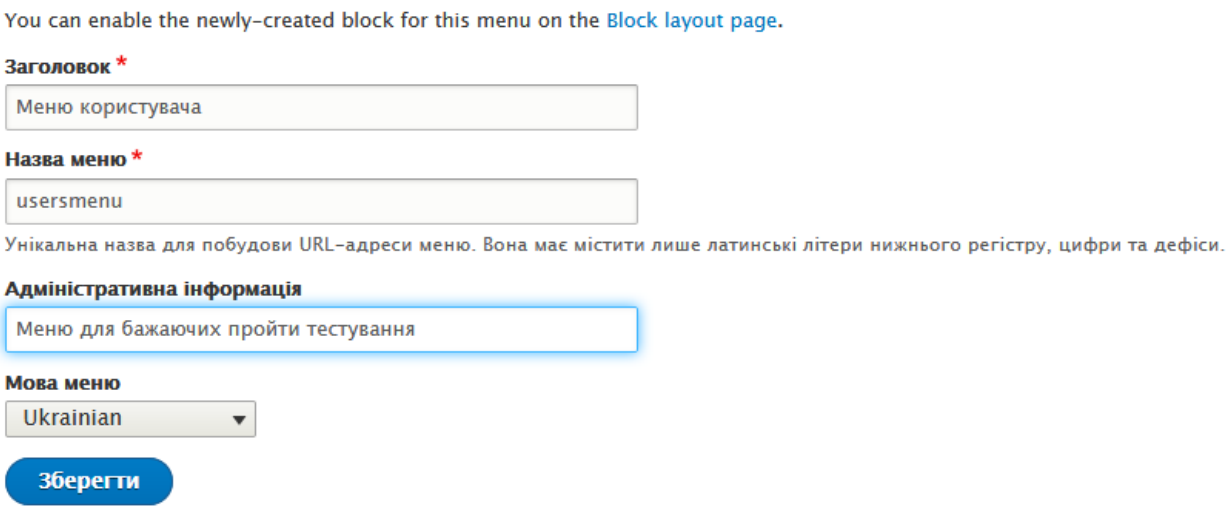

Рисунок 3.5 – Створення нового меню

Коли меню створено, необхідно створити його пункти, для цього треба обрати меню в якому треба додати новий пункт та натиснути кнопку «+Додати посилання», а потім заповнити поля (рис. 3.6).

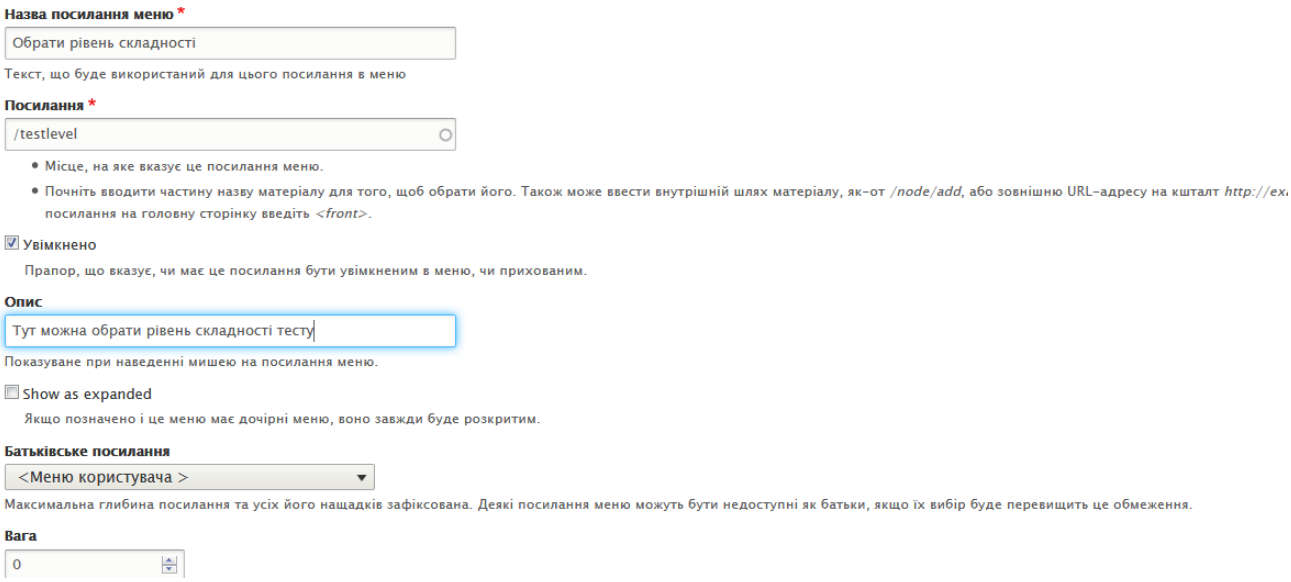

Рисунок 3.6 – Створення нового пункту меню

Створене меню не буде відображатись доки воно не буде розташовано в певному блоці сайту, а отже його необхідно розмістити. Переходимо на вкладку «Структура» – «Схема блоків» та обираємо блок в який ми бажаємо помістити для висвітлення наше меню, натискаємо кнопку «Розмістіть блок» та обираємо меню для розміщення (рис. 3.7).

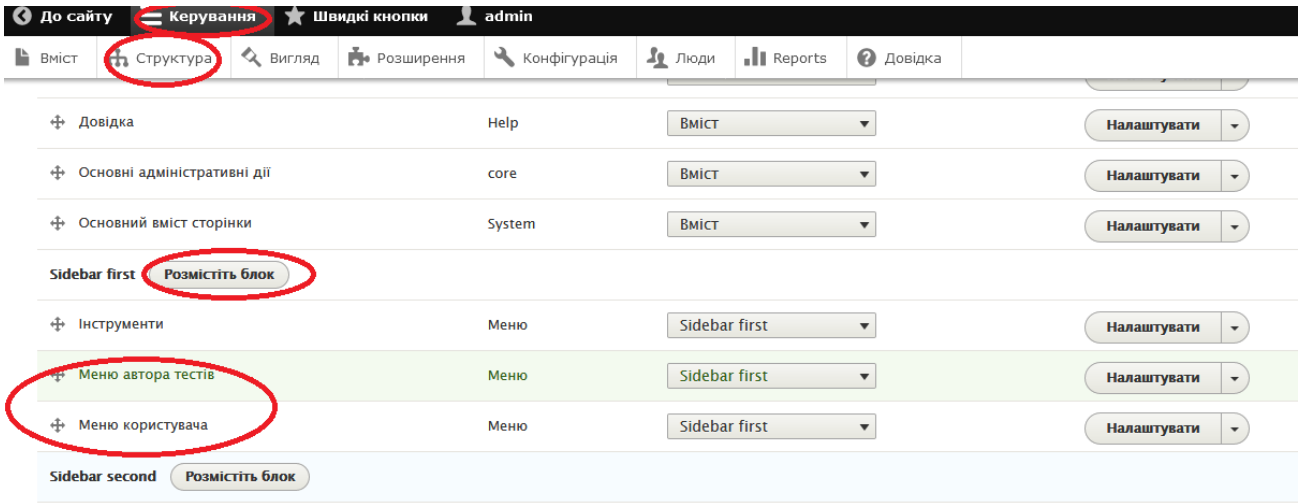

Рисунок 3.7 – Розміщення меню у лівому блоці сайту

Після правильного розміщення блоків меню вони будуть відображатися зліва на сайті (рис. 3.8).

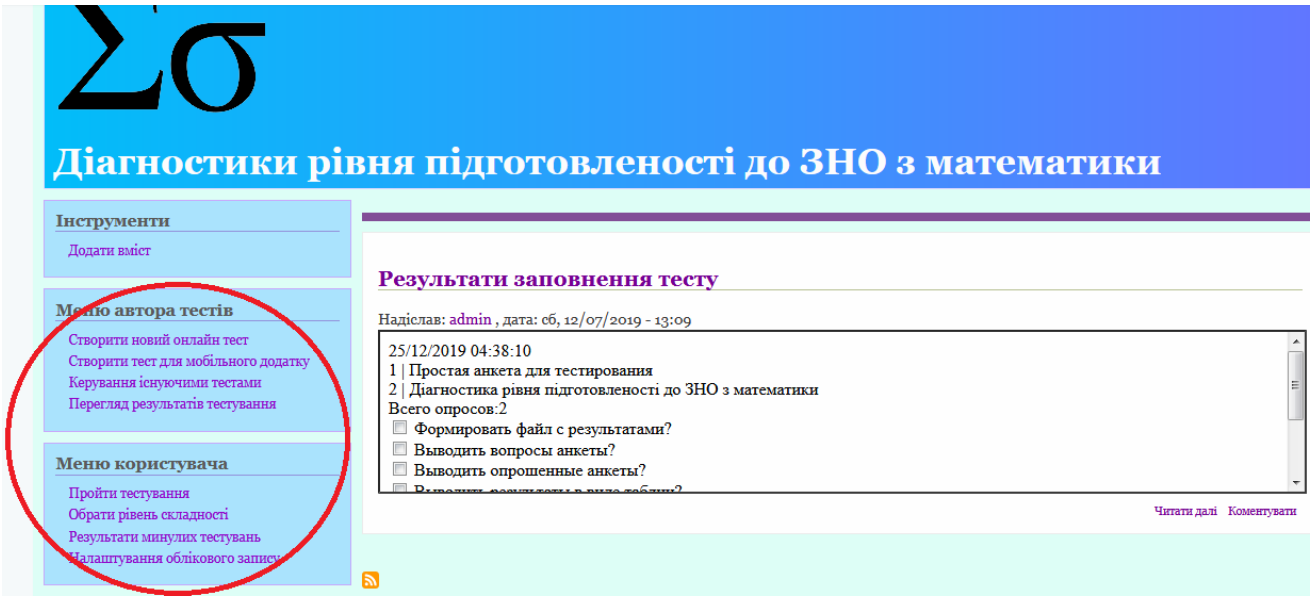

Рисунок 3.8 – Відображення створених меню

Перед початком створення тестів та початку тестування треба створити користувачів які будуть приймати участь в онлайн опитуванні. Для цього було розроблено скрипт реєстрації користувачів та створено пункт меню в «Меню автора анкет» (рис. 3.9).

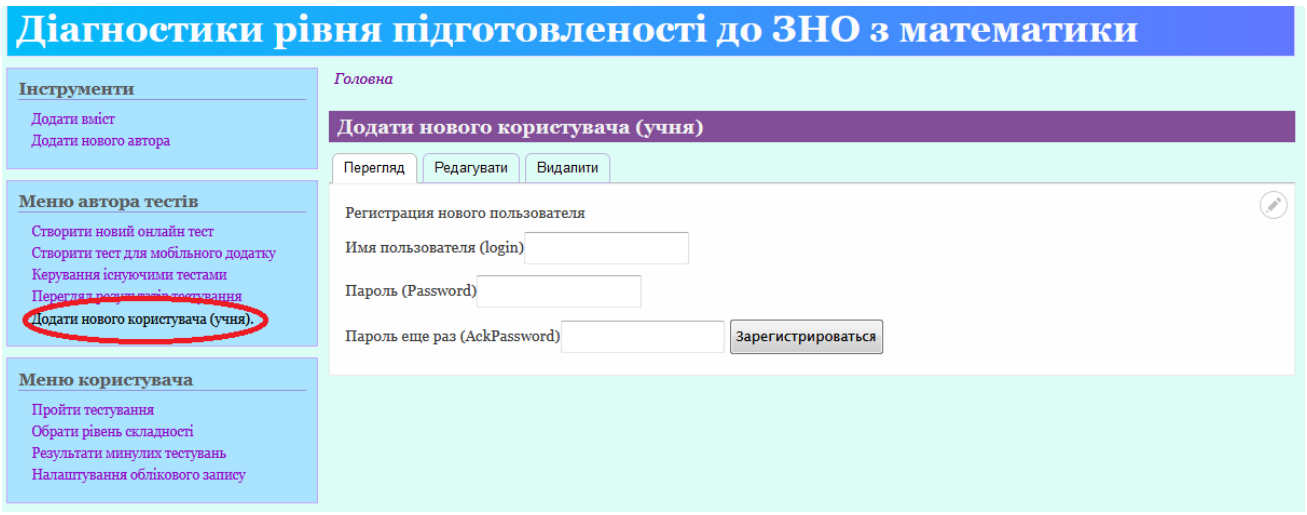

Рисунок 3.9 – Додавання нового користувача

Користувач, який бажає пройти тест обирає пункт меню «Пройти тестування», обирає тест який він бажає пройти та відповідає на питання тесту (рис. 3.10).

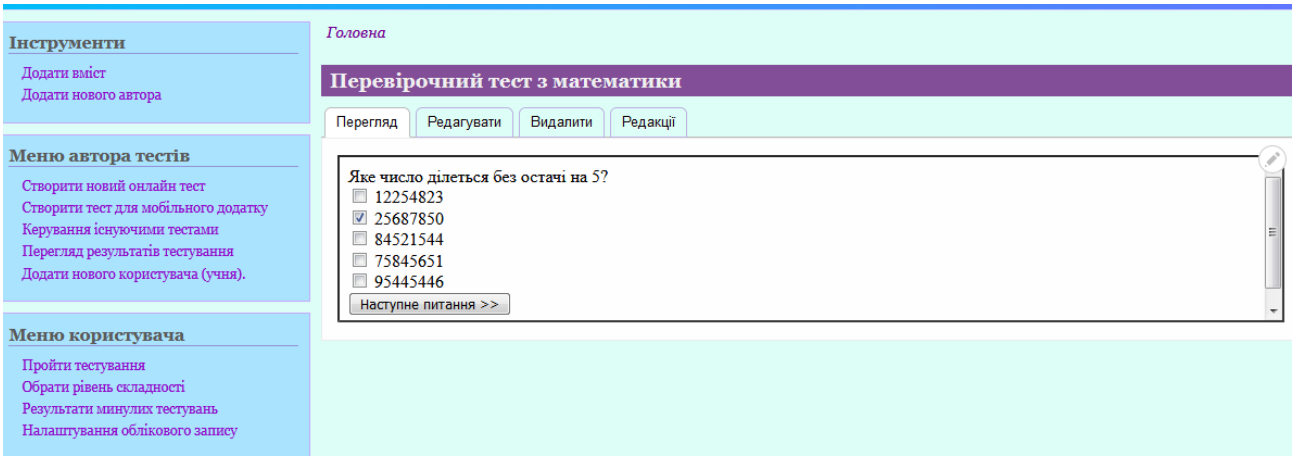

Рисунок 3.10 – Приклад заповнення тесту

Автор тестів (вчитель) при необхідності може перевірити як кожен з користувачів заповнив тест. Для цього автор переходить в меню «Перегляд результатів тестування» та отримує дані у вигляді таблиць (рис. 3.11)

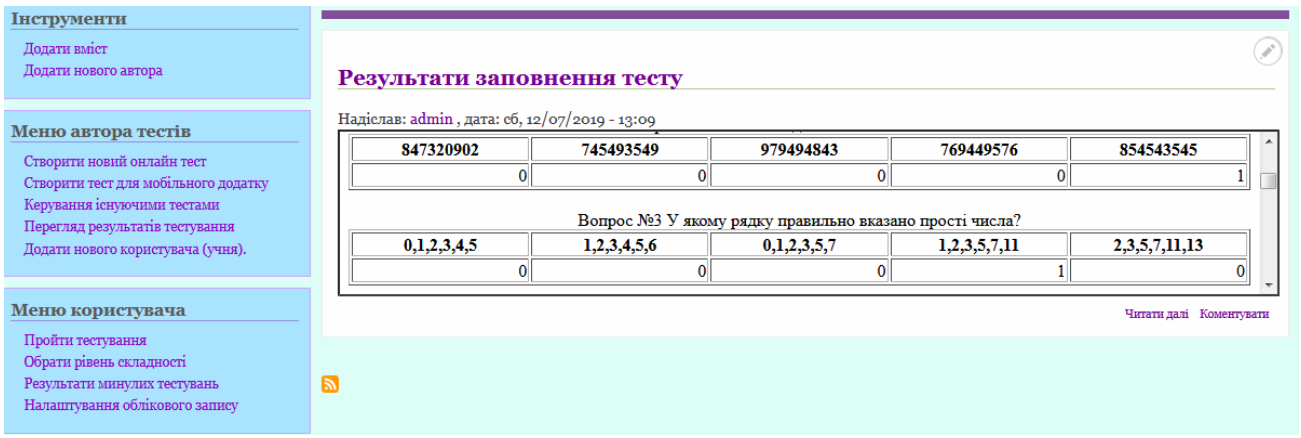

Рисунок 3.11 – Результати заповнення тестів

3.3 Розробка програмного коду

Скрипт для створення нового користувача має наступний вигляд:

Створюємо форму вводу даних:

```
<?php
	require_once 'configtest.php';
	require_once 'functionstest.php';
	$s_temp1="Реєстація нового користувача (учня)";
	$s_temp2="Імя користувача (login)";
	$s_temp3="Пароль (Password)";
	$s_temp4="Пароль ще раз (AckPassword)";
	$s_temp5="Зареєструвати";
?>
<?php echo($s_temp1) ?>
<form action="" method="post">
	<p><?php echo($s_temp2) ?><input type="text" name="LoginField" />
	<p><?php echo($s_temp3) ?><input type="text" name="PassField" />
	<p><?php echo($s_temp4) ?><input type="text" name="AckPassField" />

	<input type="submit" value="<?php echo($s_temp5) ?>"/></p>
</form>
```

Перевіряємо чи заповнена форма:

```
<?php
//Проверяем заполнена ли форма – если заполнена выводим ее результаты
if (isset($_POST['LoginField'])) {
	$UserLogin = $_POST['LoginField'];
```

```
$UserPass = $_POST['PassField'];
$AckPass = $_POST['AckPassField'];
```

Перевіряємо чи правильно заповнені поля форми:

```
if ($UserPass == ""){
    $s_temp="Поле пароля пустое! Введите пароль!";
    $s_temp=iconv("utf-8", "windows-1251", $s_temp);
    echo ($s_temp."<br>");
    exit();
    }
if ($AckPass == "") {
    $s_temp="Поле подтверждения пароля пустое! Введите подтверждение пароля!";
    $s_temp=iconv("utf-8", "windows-1251", $s_temp);
    echo ($s_temp."<br>");
    exit();
    }
if ($UserPass != $AckPass) {
    $s_temp="Пароль и подтверждение пароля не совпадают! Заполните поля заново!";
    $s_temp=iconv("utf-8", "windows-1251", $s_temp);
    echo ($s_temp."<br>");
    exit();
    }
$UserMD5 = md5($UserPass);
```

Перевіряємо чи немає вже користувача з таким ім’ям:

```
    //Проверяем чтоб не было уже пользователя с таким именем
    $query = "SELECT _login FROM Users;";
    $res = mysqli_query($db, $query);
    if ($res){
        while($row = mysqli_fetch_assoc($res)) {
            $ReadedLogin=$row['_login'];
            if ($UserLogin == $ReadedLogin) {
                $s_temp="Автор с таким именем уже существует – придумайте другое имя!";
                $s_temp=iconv("utf-8", "windows-1251", $s_temp);
                echo ($s_temp."<br>");
                exit();
                }
            }
        }
```

Якщо всі умови виконуються визначаємо максимальний індекс та робимо запис в базу даних:

```
//Определяем максимальный индекс в таблице пользователей
$query = "SELECT MAX(id) as _MaxID FROM Users;";
$res = mysqli_query($db, $query);
    if ($res){
        while($row = mysqli_fetch_assoc($res)) {
            $MaxID=$row['_MaxID'];
            }
        }
    else {
        $MaxID=0;
    }
$MaxID++;
//Записываем нового пользователя в БД
$s_temp = "INSERT INTO Users VALUES (".$MaxID.", '".$UserLogin."', '".$UserMD5."', NULL, 0, NULL, NULL, 1);";
//echo($s_temp."<br>");
$res = mysqli_query($db, $s_temp);
    if ($res){}
    else {
        echo "CAN NOT CREATE NEW AUTOR<br>";
        exit();
    }
$s_temp="Користувач з ім ям ".$UserLogin." вдало створений с ID=".$MaxID;
echo ($s_temp."<br>");
}?>
```

У Drupal існує два варіанти тестування – модульне і функціональне.

Тестування функціональності через клас DrupalWebTestCase є більш поширеним, можуть виникнути потреби написати традиційний модульний тест. Модуль simpletest дозволяє тестувати будь-яким з цих методів.

Алгоритм в модульному тестуванні такий: по суті, викликається якась функція або метод, як аргументи якої передаються всі можливі варіанти значень. У нас є така функція:

```
function simpletest_example_empty_mysql_date ($ date_string) {
  if (empty ($ date_string) || $ date_string == '0000-00-00' || $ date_string == '0000-00-00 00:00:00') {
    return true;
  }
```

```
  return false;
}
```

Виходячи з написаного, функція повинна повертати TRUE тільки в тому випадку, якщо змінна $ date_string є коректною для вставки в базу даних. Нам треба перевірити повернені значення функції, якщо їй будуть передані різні дані в аргументі. Наприклад, змінна $ date_string може дорівнювати NULL, порожній рядку, ‘0000-00-00’, ‘0000-00-00 00:00:00’, числу, не порожній рядку і т.д.

Створюємо тест.

Насамперед треба створити файл з тестом. Він створюється в папці з модулем і має розширення .test. У нашому прикладі це simpletest_example.test.

Щоб Drupal знайшов ваш тест, вам треба описати його в info файлі модуля в конструкції files []. Якщо модуль вже був включений – після додавання тесту обов’язково очистіть кеш на сайті. приклад:

```
files [] = simpletest_example.test
```

Тепер відкриваємо файл з тестом simpletest_example.test і створіть клас для модульного тесту. Він повинен бути успадкований від DrupalUnitTestCase:

```
<? Php
 class SimpletestUnitTestExampleTestCase extends DrupalUnitTestCase {
 }
```

Так само як і в функціональному тестуванні, в тесті повинен бути метод getInfo (), щоб цей тест було видно з адмінки simpletest’а. Він повинен містити в собі ім’я, опис і, якщо треба, групу до якої він відноситься.

```
public static function getInfo () {
  return array (
    ‘Name’ => ‘Simpletest Example unit tests’,
    ‘Description’ => ‘Test that simpletest_example_empty_mysql_date works properly.’,
```

```
    'Group' => 'Examples',
  );
}
```

Після імплементації методу getInfo () можна переходити до написання тесту. Всі модульні тести повинні починатися зі слова 'test' в нижньому регістрі. Всі методи, які починаються з цього слова, автоматично розпізнаються Simpletest'ом і запускаються на етапі тестування. Можна відокремити різні вставки (assertations) в різні методи, проте рекомендується для цього використовувати один і той же метод. Зробити це можна в такий спосіб:

```
function testEmptyMySQLDate () {
   $ Result = simpletest_example_empty_mysql_date (NULL);
  $ Message = t ( 'A NULL value should return TRUE.');
  $ This-> assertTrue ($ result, $ message);
   $ Result = simpletest_example_empty_mysql_date ( '');
  $ Message = t ( 'An empty string should return TRUE.');
  $ This-> assertTrue ($ result, $ message);
   $ Result = simpletest_example_empty_mysql_date ( '0000-00-00');
  $ Message = t ( 'An "empty" MySQL DATE should return TRUE.');
  $ This-> assertTrue ($ result, $ message);
   $ Result = simpletest_example_empty_mysql_date (date ( 'Y-m-d'));
  $ Message = t ( 'A valid date should return FALSE.');
  $ This-> assertFalse ($ result, $ message);
 }
```

3.4 Створення мобільного додатку для опитування

Для написання мобільного додатку було використано середу розробки Android Studio оскільки це спеціальна безкоштовна середа для пристроїв з операційною системою Android (рис. 3.12).

Наш мобільний додаток складається не з одного типового екрану. Типовий екран в Android Studio називається Activity.

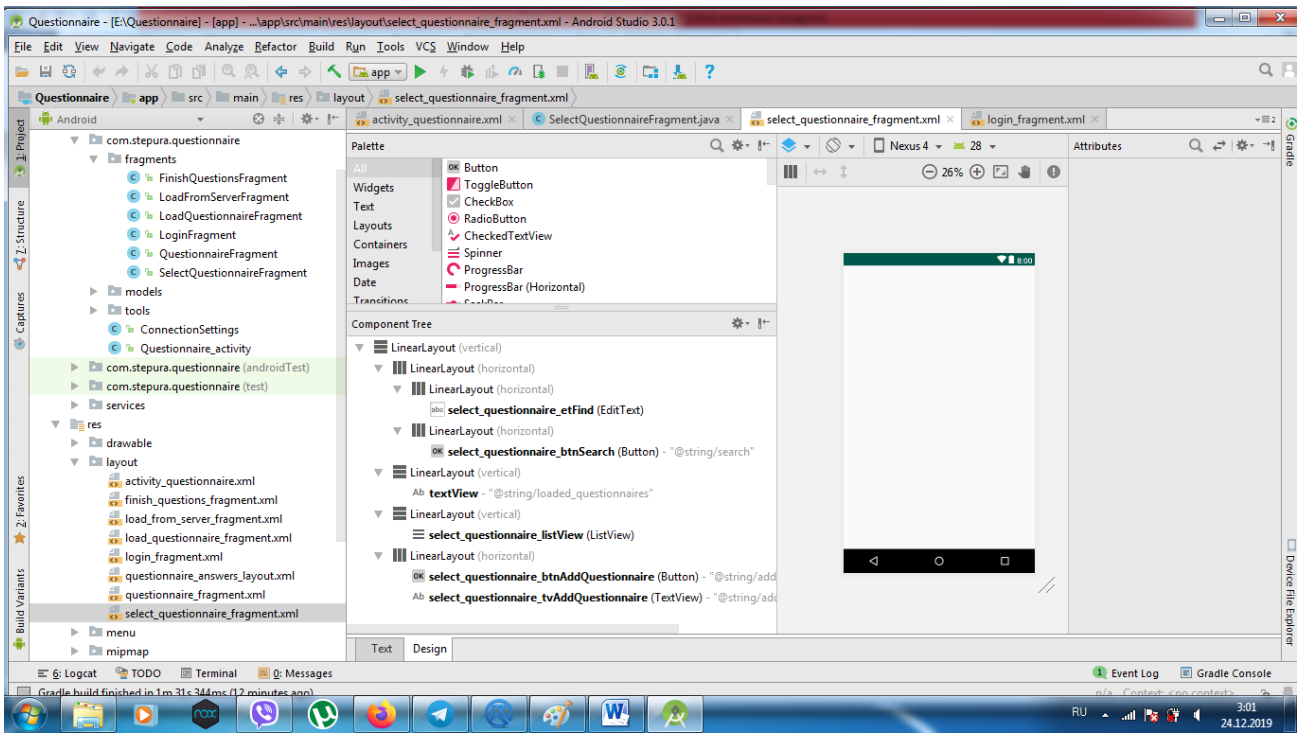

Рисунок 3.12 – Проект завантажений в Android Studio

Реалізовані в мобільному додатку Activity:

– login_fragment;

– select_questionnare_fragment;

– load_from_server_fragment;

– questionnare_fragment;

– finish_questions_fragment;

Розгрянемо розроблені Activity більш докладно.

Екран login_fragment використовується для входу у програму через введення імені користувача та паролю. Зовнішній вигляд цього екрану наведено на рис. 3.13.

Для створення будь якого екрану мобільного додатку розробляється інтерфейс у вигляді XML файлу, однак для спрощення створення екранів в Android Studio присутня палітра компонентів для розміщення елементів перетягуванням (рис. 3.14).

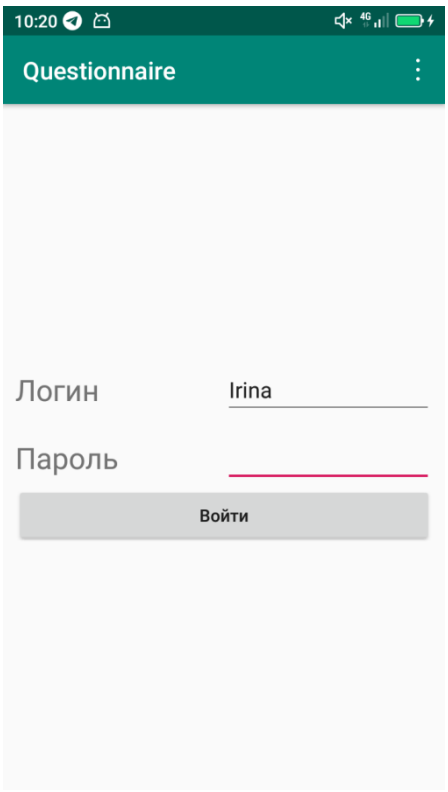

Рисунок 3.13 – Зовнішній вигляд login_fragment (форми авторизації)

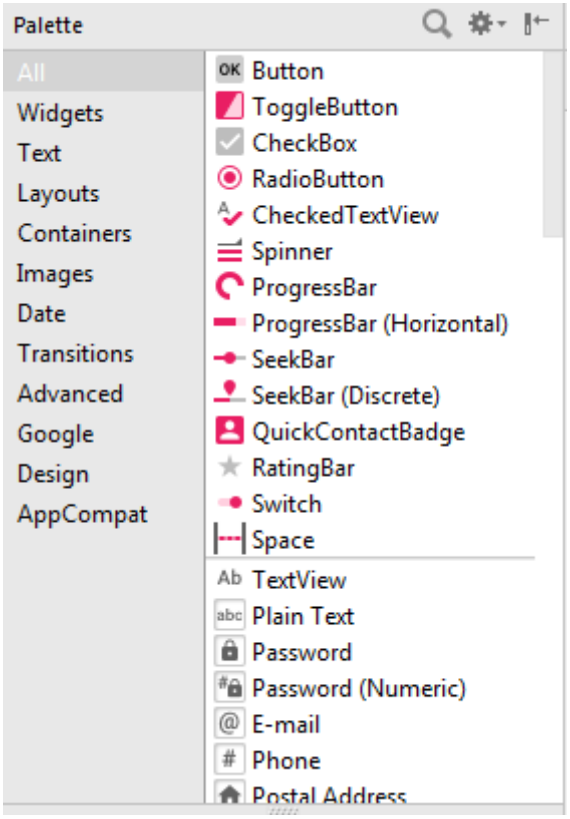

Рисунок 3.14 – Палітра елементів екрану

Приклад оформлення екрану у вигляді XML файлу:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="vertical">

        <Button
            android:id="@+id/load_from_server_btnReturn"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="@string/return_str"
            android:textAllCaps="false" />
    </LinearLayout>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:orientation="vertical">

        <ListView
            android:id="@+id/load_from_server_listView"
            android:layout_width="match_parent"
            android:layout_height="match_parent" />
    </LinearLayout>
</LinearLayout>
```

При натисканні на кнопку запускається обробник натиснення який викликає процедуру login (рис 3.15)

```
public void onClick(View view) {
    switch (view.getId()){
        case R.id.login_btnSignIn:
            login(etLogin.getText().toString(), etPass.getText().toString());
            break;
        case R.id.login_btnLanguage:
```

```
            dialogBuilder = new AlertDialog.Builder(getActivity());
            dialogBuilder.setTitle(R.string.select_language).setItems(languages, new
DialogInterface.OnClickListener() {
                @Override
                public void onClick(DialogInterface dialogInterface, int i) {
                    ((Questionnaire_activity)getActivity()).changeLocale(languages[i]);
                    updateView();
                }
            });
            dialogBuilder.create();
            dialogBuilder.show();
            break;
    }
}
private void login(String login, String pass){
    String[] data = {login, pass};
    String connectionString = ConnectionSettings.getCodeLoginInterviewer() + " " +
gson.toJson(data);
    ConnectionSettings.setConnectionString(connectionString);
    String message = ConnectionSettings.getCodeGetAutoLog();
    ((Questionnaire_activity)getActivity()).sendMessageToServer(message);
}
```

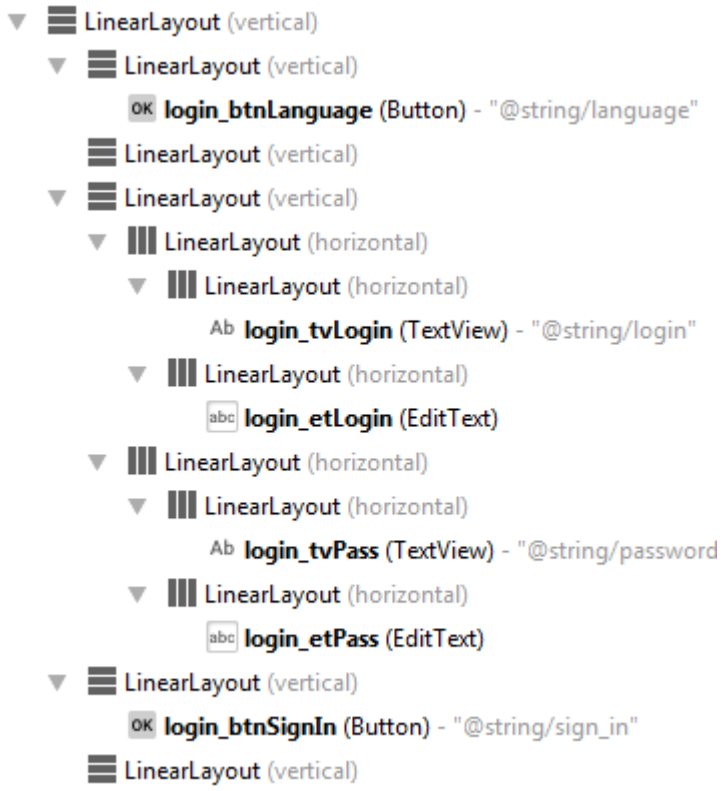

Рисунок 3.15 – Структура Activity Login_fragment

Треба зазначити, що перевірка паролю відбувається на сервері тому викликається процедура sendMessageToServer(message).

Наступним екраном мобільного додатку є перелік анкет для опитування, що зберігається у локальній базі даних (рис. 3.16).

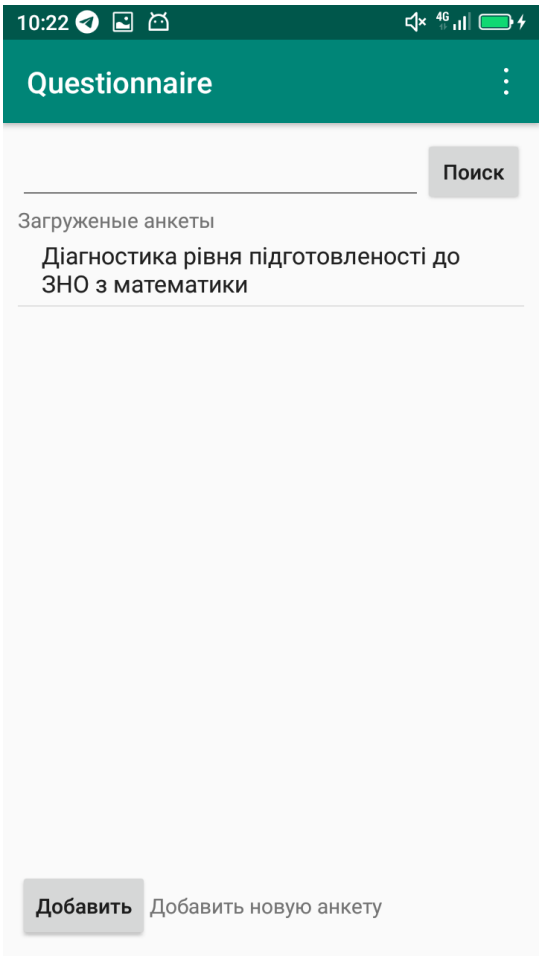

Рисунок 3.16 – Зовнішній вигляд списку завантажених тестів

Після вибору тесту програма запитує користувача чи дійсно він хоче пройти тестування та виводить параметри тесту (рис.3.17–3.18).

Для виводу списку анкет, що можуть бути використані для проходження тестування було створена процедура refreshListView

```
public void refreshListView(ArrayList<Questionnaire> newQuestionnaires){
   questionnaires.removeAll(questionnaires);
   questionnaires.addAll(newQuestionnaires);
   adapter.notifyDataSetChanged();
}
```

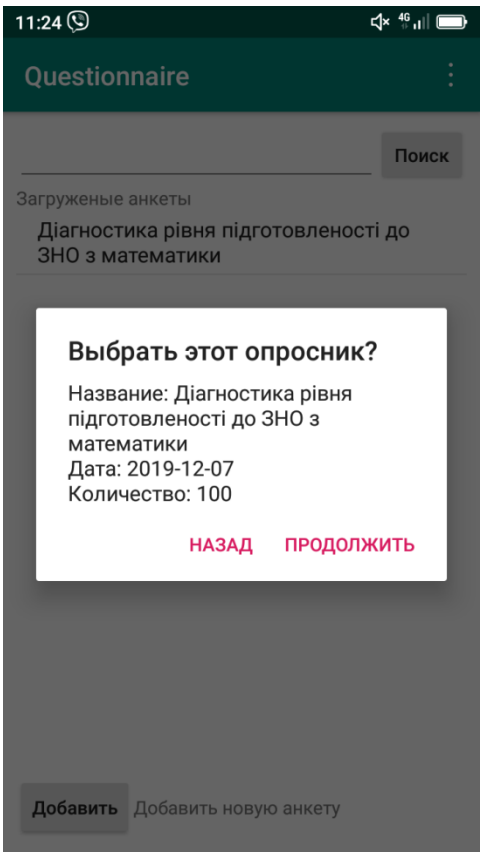

Рисунок 3.17 – Оповіщення з параметрами анкети

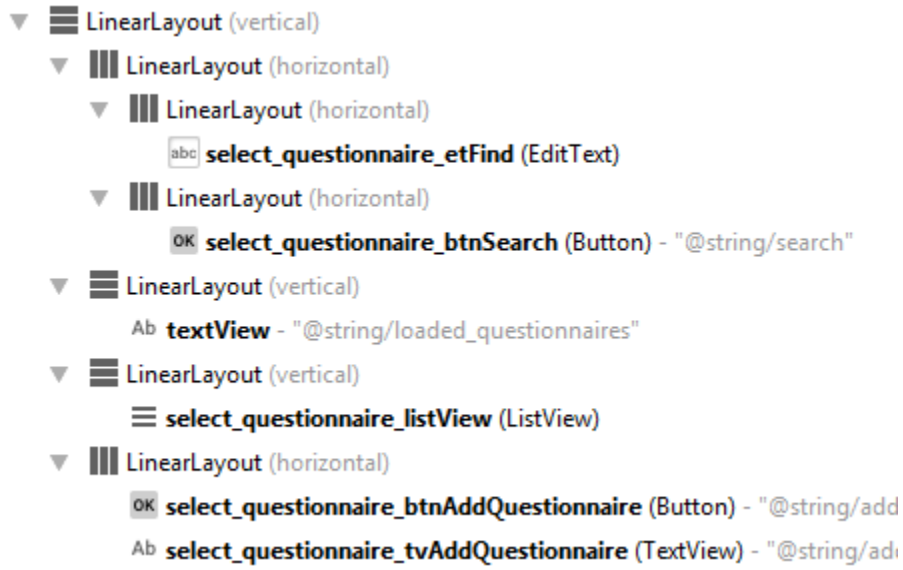

Рисунок 3.18– Структура Activity Select_Quitionnare_fragment

При виборі анкети для тестування використовується процедура обробника натискання на об’єкт:

```
public void onItemClick(AdapterView<?> adapterView, View view, int i, long l) {
   final Questionnaire q = questionnaires.get(i);

   String message = getString(R.string.name) + ": "+ q.getName() + "\n" +getString(R.string.date) +
": "+ q.getDate() +"\n"+ getString(R.string.count)  + ": " + String.valueOf(q.getqCount());

   alertBuilder = new AlertDialog.Builder(getActivity());
   alertBuilder.setTitle(getString(R.string.select_this_questionnaire)).setMessage(message);
   alertBuilder.setNegativeButton(R.string.return_str, new DialogInterface.OnClickListener() {
      @Override
      public void onClick(DialogInterface dialogInterface, int i) {

      }
   });
   alertBuilder.setPositiveButton(R.string.continue_str, new DialogInterface.OnClickListener() {
      @Override
      public void onClick(DialogInterface dialogInterface, int i) {
         LoadQuestionnaireAsync questionnaireAsync = new LoadQuestionnaireAsync();
         questionnaireAsync.execute(q);

      }
   });
   alertBuilder.show();
}
```

Для завантаження анкети з сервера, що відбувається переходом на наступний екран при натисканні кнопки «добавить» відбувається процедура обробки:

```
      @Override public void onClick(View view) {
   switch (view.getId()){
      case R.id.select_questionnaire_btnSearch:
         GetQuestionnairesAsync async = new GetQuestionnairesAsync();
         async.execute(etFind.getText().toString());
         break;
      case R.id.select_questionnaire_btnAddQuestionnaire:
         activity.setLoadFromServerFragment();
         break;
   }
}
```

Екран завантаження анкет з сервера має вигляд як показано на рис. 3.19.

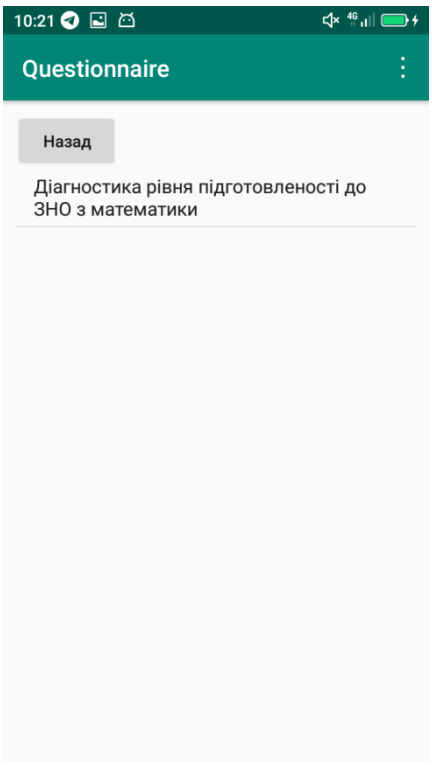

Рисунок 3.19 – Зовнішній вигляд списку тестів на сервері

Першочергово при появі екрану програма зв’язується з сервером та оновлює перелік тестів (рис 3.20–3.21).

```
public View onCreateView(@NonNull LayoutInflater inflater, @Nullable ViewGroup container,
@Nullable Bundle savedInstanceState) {
    View v = inflater.inflate(R.layout.load_from_server_fragment, null);
    activity = (Questionnaire_activity)getActivity();
    dataBaseHelper = activity.getDataBaseHelper();
    questionnaires = new ArrayList<>();
    gson = new Gson();

    btnReturn = v.findViewById(R.id.load_from_server_btnReturn);
    list = v.findViewById(R.id.load_from_server_listView);

    btnReturn.setOnClickListener(this);
    arrayAdapter = new ArrayAdapter<>(v.getContext(), android.R.layout.simple_list_item_1,
questionnaires);
    list.setAdapter(arrayAdapter);
    list.setOnItemClickListener(this);


activity.sendMessageToServer(ConnectionSettings.getCodeSelectAllQuestionnaireSortByName());
```

```
    return v;
}
```

Після завантаження анкети з сервера вона зберігається в локальній базі даних.

```
private class InsertDownloadedQuestionnaireToSQLite extends AsyncTask<Questionnaire, Void, String>{

    @Override
    protected String doInBackground(Questionnaire... questionnaires) {

        Questionnaire questionnaire = questionnaires[0];
        SQLiteDatabase db = dataBaseHelper.getWritableDatabase();

        String query = "SELECT serv_id FROM Questionnaires WHERE serv_id = " + questionnaire.getId();

        Cursor cursor = db.rawQuery(query, null);
        while(cursor.moveToNext()){
            db.close();
            return getString(R.string.already_uploaded);
        }

        ContentValues contentValues = new ContentValues();
        contentValues.put("serv_id", questionnaire.getId());
        contentValues.put("id_author", questionnaire.getIdAuthor());
        contentValues.put("name", questionnaire.getName());
        contentValues.put("date", questionnaire.getDate());
        contentValues.put("q_count", questionnaire.getqCount());
        db.insert("Questionnaires", null, contentValues);

        ArrayList<Question> questions = questionnaire.getQuestions();

        for(int i = 0; i<questions.size(); i++){
            Question question = questions.get(i);

            ContentValues content = new ContentValues();
            content.put("serv_id", question.getId());
            content.put("quest_type", question.getQuestType());
            content.put("group_type", question.getGroupType());
            content.put("text", question.getText());
            content.put("answ_count", question.getAnswCount());
```

```
            content.put("id_questionnaire", question.getIdQuestionnaire());
            db.insert("Questions", null, content);

            ArrayList<Answer> answers = question.getAnswers();

            for(int j = 0; j<answers.size(); j++){
                Answer answer = answers.get(j);

                ContentValues values = new ContentValues();
                values.put("serv_id", answer.getId());
                values.put("type", answer.getType());
                if(answer.getType()==0){
                    values.put("text", answer.getText());
                }
                else{
                    values.put("text", "");
                }
                values.put("id_question", answer.getIdQuestion());
                db.insert("Answers", null, values);
            }
        }
        db.close();
        return getString(R.string.downloaded);
    }

    @Override
    protected void onPostExecute(String string) {
        activity.showMessage(string);
        super.onPostExecute(string);
    }
}
```

При виборі анкети висвітлюється оповіщення стосовно того, чи впевнений користувач, що хоче завантажити анкету

```
public void onItemClick(AdapterView<?> adapterView, View view, int i, long l) {
    Questionnaire q = questionnaires.get(i);
    final int id = q.getId();
    String message = getString(R.string.name) + ": "+ q.getName() + "\n" +getString(R.string.date) +
": "+ q.getDate() +"\n"+ getString(R.string.count)  + ": " + String.valueOf(q.getqCount());
    alertBuilder = new AlertDialog.Builder(getActivity());
    alertBuilder.setTitle(getString(R.string.download_this_questionnaire)).setMessage(message);
    alertBuilder.setNegativeButton(R.string.return_str, new DialogInterface.OnClickListener() {
        @Override
```

```
        public void onClick(DialogInterface dialogInterface, int i) {

        }
    });
    alertBuilder.setPositiveButton(R.string.continue_str, new DialogInterface.OnClickListener() {
        @Override
        public void onClick(DialogInterface dialogInterface, int i) {

            String[] idQuestionnaire = {String.valueOf(id)};
            activity.sendMessageToServer(ConnectionSettings.getCodeLoadFullQuestionnaire() + " " +
gson.toJson(idQuestionnaire));
        }
    });
    alertBuilder.show();
}
```

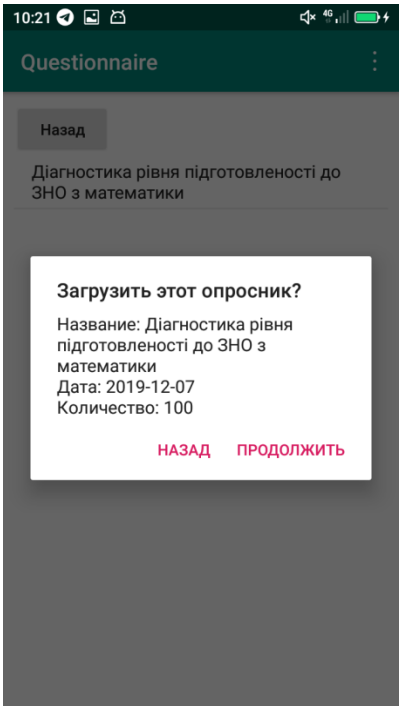

Рисунок 3.20 – Оповіщення щодо завантаження анкети

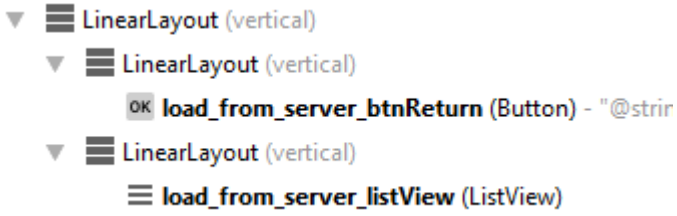

Рисунок 3.21 – Структура load_from_server_fragment

При натисканні кнопки «Назад» відбувається перехід до екрану з вибором анкет

```
public void onClick(View view) {
   switch (view.getId()){
      case R.id.load_from_server_btnReturn:
         activity.setSelectQuestionnaireFragment();
         break;
```

Після того як користувач обрав тест для початку тестування йому висвічуються питання з варіантами відповіді (рис. 3.22–3.23) questionnare_fragment

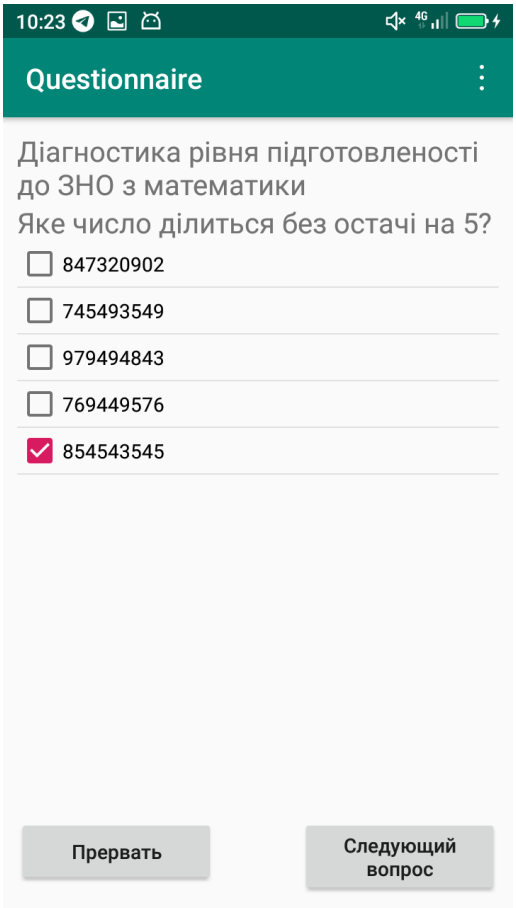

Рисунок 3.22 – Зовнішній вигляд питання з варіантами відповіді

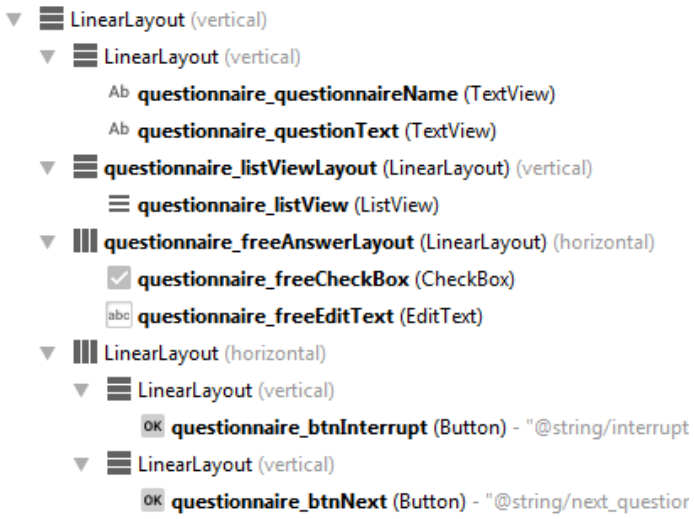

Рисунок 3.23 – Структура questionnare_fragment

При старті та оновленні екрану виконується процедура, яка завантажує наступне питання (рис. 3.24–3.26):

```
public void onStart() {
   super.onStart();
   if(answers.size()==0)
   {
      cbFree.setChecked(true);
   }
   tvQuestionnaireName.setText(questionnaire.getName());
   tvQuestionText.setText(currentQuestion.getText());
}
```

При натисканні на кнопки викликається наступний обробник:

```
public void onClick(View view) {
   switch (view.getId()){
      case R.id.questionnaire_btnNext:
         result = questionAdapter.getChekedAnswers();
         if(isCustom && cbFree.isChecked())
         {
            customAnswer.setText(etFree.getText().toString());
            result.add(customAnswer);
         }
         nextQuestion(result);
         break;
```

```
        case R.id.questionnaire_btnInterrupt:
            activity.setFinishQuestionsFragment();
            break;
    }
}
```

Завантаження відповідей для питання відбувається викликом процедури:

```
private void nextQuestion(ArrayList<Answer> result) {
    currentQuestion.setAnswers(result);
    numOfCurrentQuestion++;
    if(numOfCurrentQuestion > questions.size()-1){
        questionnaire.setQuestions(questions);
        SaveResult saveResult = new SaveResult();
        saveResult.execute(questionnaire);
    }
    else {
        currentQuestion = questions.get(numOfCurrentQuestion);
        answers = currentQuestion.getAnswers();
        questionAdapter.setNewData(answers, currentQuestion.getAnswCount());
        isCustom = isCustomAnswerInQuestion(answers);
        updateFragment();
    }
}
```

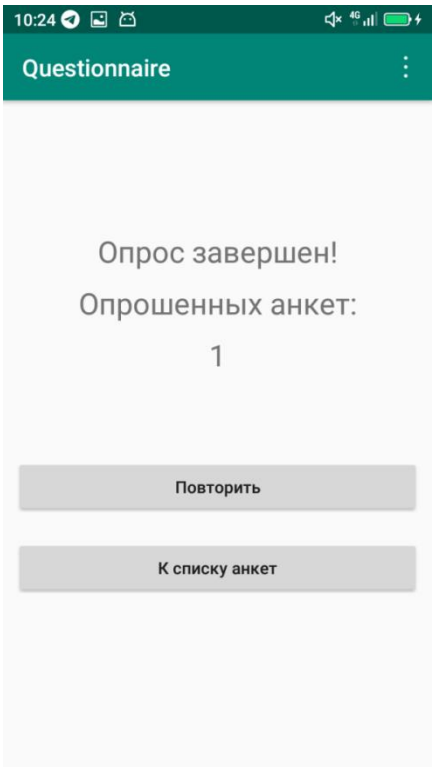

Рисунок 3.24 – Зовнішній вигляд екрана при завершенні тестування

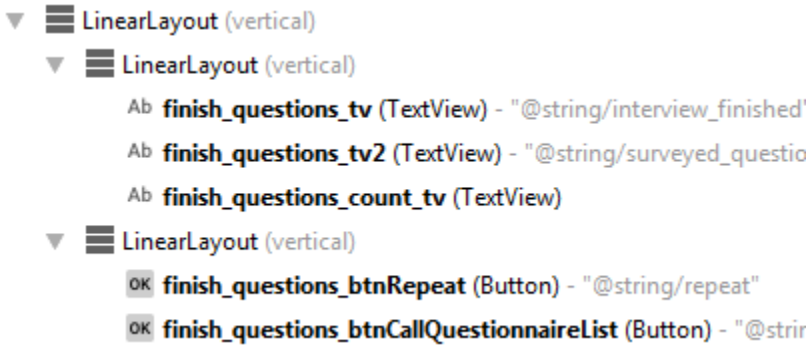

Рисунок 3.25 – Структура finish_questions_fragment

Рисунок 3.26 – Зовнішній вигляд екрана при вході в меню налаштувань

3.5 Апробація тестів

Щоб перевірити чим відрізняється тестування офіційного тесту ЗНО з математики за 2019 рік та наше адаптивне тестування з математики, було виконано аналіз результатів 20 учнів 11 класу Запорізької гімназії №46

Запорізької міської ради. Для цього учні відповіли на 20 тестових питань ЗНО з математики за 2019 рік та пройшли наше адаптивне тестування у мобільному додатку. Результати було переведено у таблицю, у якій правильні відповіді позначаються 1, а неправильні відповіді позначаються 0. Аналіз було проведено у MS Excel за допомогою функції Аналіз даних (табл. 3.2–3.7).

Таблиця 3.2 – Аналіз результатів ЗНО з математики за 2019 рік

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	
20	0	1	0	0	1	0	1	1	1	1	1	0	1	0	1	1	1	1	0	1	13
19	0	1	0	0	1	1	1	1	0	0	1	1	0	1	1	1	1	0	1	0	12
18	0	1	1	1	1	0	1	0	0	1	0	0	0	0	1	1	0	1	0	0	9
15	1	1	0	0	0	0	1	0	0	0	0	0	1	0	0	1	1	0	1	0	7
16	1	0	1	0	0	1	0	0	0	0	1	0	1	0	1	0	0	1	0	0	7
17	0	0	0	1	1	0	0	1	0	1	0	0	0	0	1	0	1	1	0	0	7
13	0	1	1	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	1	0	5
14	0	1	0	0	0	0	0	0	0	1	1	0	0	1	0	0	0	0	0	1	5
10	0	0	0	1	0	0	0	1	0	0	0	0	0	0	1	0	0	0	1	0	4
11	0	0	0	1	0	0	1	1	0	0	0	0	0	0	0	0	0	0	1	0	4
12	0	0	1	0	1	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	4
5	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	3
6	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	3
7	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	3
8	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	3
9	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	3
4	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	2
1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
2	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1
3	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
	2	7	6	6	6	3	8	7	4	5	5	2	4	3	6	6	4	4	7	2	

У таблиці 3.2 по строкам йдуть учні, по стовпчикам номер завдань. Останній стовпчик – це кількість набраних балів кожного учня за 20 можливих,

упорядкованих за спаданням. Остання строчка – це кількість правильних відповідей з кожного завдання окремо (рис. 3.27).

Далі розглянуто частоту балів та побудовано відповідну діаграму.

Таблиця 3.3 – Частота балів ЗНО з математики за 2019 рік

бал	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
частота	3	1	5	3	2	0	3	0	1	0	0	1	1	0	0	0	0	0	0

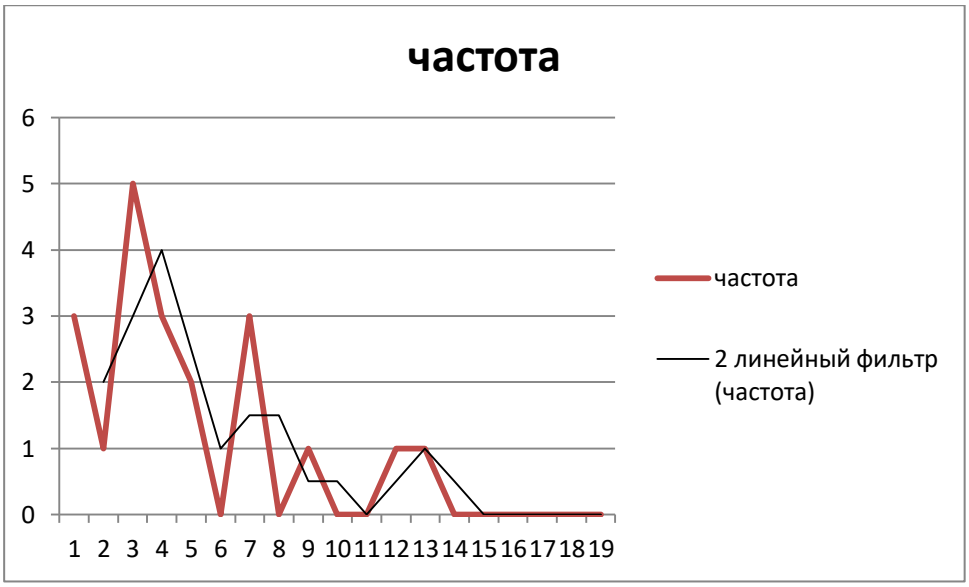

Рисунок 3.27 – Діаграма частот балів із ЗНО з математики за 2019 рік

Таблиця 3.4 – Пакетний аналіз даних результатів пройденого тесту

Среднее	4,85
Стандартная ошибка	0,761836
Медиана	4
Мода	3
Стандартное отклонение	3,407036
Дисперсия выборки	11,60789
Эксцесс	0,841692
Асимметричность	1,153835
Интервал	12
Минимум	1
Максимум	13

Продовження таблиці 3.4

Сумма	97
Счет	20
Уровень надежности(50,0%)	0,523855

Таблиця 3.5 – Аналіз результатів адаптивного тестування з математики

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	
15	1	1	1	1	0	1	1	1	1	1	1	1	0	1	1	0	0	1	1	0	15
14	0	1	1	0	0	1	1	1	0	0	1	1	1	1	1	1	1	1	0	1	14
11	0	0	1	1	0	1	1	1	0	0	1	0	1	1	1	1	0	1	1	1	13
12	1	0	1	1	1	1	0	1	1	0	1	1	0	1	0	0	0	1	0	1	12
1	0	0	1	1	0	1	1	1	0	0	1	1	0	0	1	1	0	1	1	1	12
20	0	1	0	0	0	1	1	1	0	0	1	0	1	0	1	1	0	1	1	1	11
3	0	0	1	0	0	1	0	1	0	0	0	0	1	1	1	1	1	1	1	1	11
8	1	0	1	0	1	1	0	1	0	0	0	0	1	0	1	0	0	1	1	1	10
9	1	1	1	1	0	1	1	1	0	0	1	0	1	0	0	0	0	0	0	1	10
17	0	0	1	1	0	1	0	0	0	0	0	0	1	1	1	1	0	1	1	1	10
18	1	0	1	1	0	0	1	0	1	0	0	1	0	0	1	0	0	1	0	1	9
19	1	0	0	1	0	0	1	0	0	0	1	1	0	0	1	1	0	1	0	1	9
2	0	0	1	0	0	1	1	1	0	0	1	0	0	0	1	0	0	1	1	1	9
13	0	0	1	0	0	1	1	1	0	0	0	0	0	0	1	1	0	0	1	1	8
16	0	0	1	0	0	1	0	1	0	0	1	0	0	1	1	0	0	0	1	1	8
6	1	0	1	1	0	1	1	0	0	0	1	1	0	0	0	0	0	0	0	0	7
10	0	1	0	0	0	1	1	1	0	0	0	0	0	0	0	0	1	1	0	1	7
5	0	0	1	0	0	0	1	1	0	0	1	0	0	0	0	0	0	1	0	1	6
4	0	0	0	0	0	1	0	1	1	0	0	0	0	0	1	0	0	1	0	1	6
7	1	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	1	1	5
	8	5	15	9	2	16	13	16	4	1	12	7	7	7	15	8	3	15	11	18	

У таблиці 3.5 по строкам йдуть учні, по стовпчикам номер завдань. Останній стовпчик – це кількість набраних балів кожного учня за 20 можливих, упорядкованих за спаданням. Остання строчка – це кількість правильних відповідей з кожного завдання окремо (рис. 3.28).

Далі розглянуто частоту балів та побудовано відповідну діаграму.

Таблиця 3.6 – Частота балів адаптивного тестування з математики

бал	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
частота	0	0	0	0	1	2	2	2	3	3	2	2	1	1	1	0	0	0	0

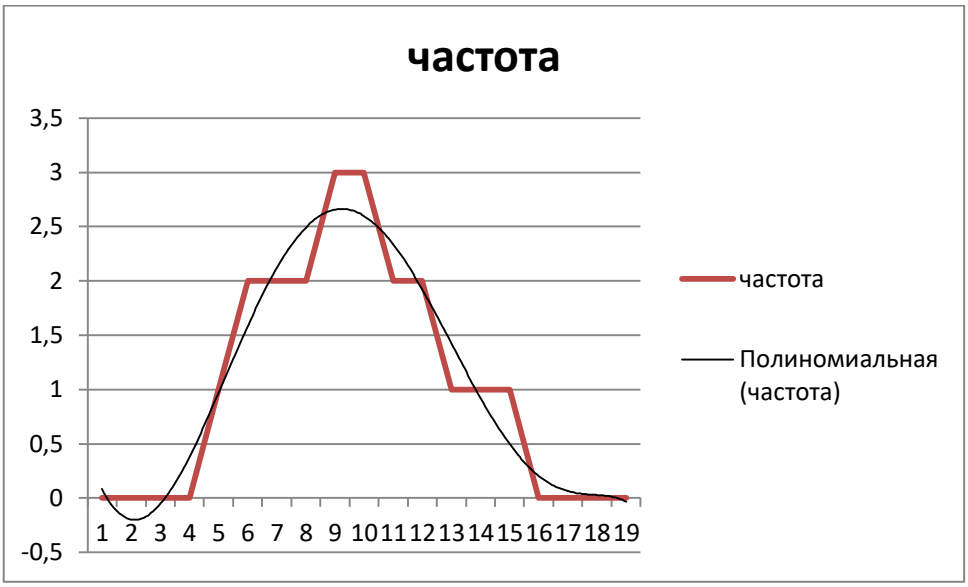

Рисунок 3.28 – Діаграма частот балів адаптивного тестування з математики

Таблиця 3.7 – Пакетний аналіз даних результатів пройденого тесту

Среднее	9,6
Стандартная ошибка	1,161668
Медиана	8,5
Мода	15
Стандартное отклонение	5,195139
Дисперсия выборки	26,98947
Эксцесс	-1,22446
Асимметричность	-0,02142
Интервал	17
Минимум	1
Максимум	18
Сумма	192
Счет	20
Уровень надежности(50,0%)	0,798788

Дивлячись на результати пакетного аналізу результатів тестування, можна зробити висновок, що результати проходження адаптивного тесту

краща, це видно зі значення, наприклад, асиметрії – у адаптивному тесті значення асиметрії близьке до нуля, тому можна зробити висновок, про те, що тест добре підібраний за трудністю, та за діаграмою видно, що результати адаптивного тестування відповідає нормальному розподілу (розподілу Гауса).

3.6 Висновки за 3 розділом

У третьому розділі показано алгоритм розробки програмного забезпечення для проведення адаптивного тестування з математики, показано використані скрипти. Проведено аналіз апробації пройдених тестів учнів за допомогою MS Excel.

ВИСНОВКИ

У кваліфікаційній роботі розглянуто основні теоретичні відомості про зовнішнє незалежне оцінювання, про основні види комп’ютерного тестування та про адаптивні тести; також розглянуто алгоритм розробки сайту з адаптивним тестуванням з математики.

У роботі представлено:

– структуру системи та бази даних, показано ER та UML діаграми;

– задачі, які виконує програмне забезпечення з проведення адаптивного тестування;

– алгоритм розробки програмного забезпечення для проведення адаптивного тестування з математики;

– використані скрипти;

– аналіз апробації пройдених тестів учнів за допомогою MS Excel.

Досліджено предметну область адаптивного тестування, розроблено алгоритм проведення адаптивного тестування з математики, вибірано програмне забезпечення та реалізовано поставлену мету.

Дивлячись на всі переваги адаптивного тестування, стає зрозумілим те, що треба вводити до навчання, для перевірки рівня знань, адаптивні тести. Саме тому, завданням цієї роботи було введення адаптивного тестування з математики для перевірки рівня підготовленості учнів з математики, шляхом проходження адаптивного тестування.

ПЕРЕЛІК ПОСИЛАНЬ

1. Кашина Г. С., Сергієнко В. П. Зовнішнє незалежне оцінювання в освіті України. Курс лекцій : навч. посіб. Луцьк, 2010. 115 с.

2. Федорук П. І. Адаптивні тести: статистичні методи аналізу результатів тестового контролю знань // *Математичні машини і системи*. 2007. № 3,4. С. 122–138.

3. Федорук П. І. Моделі і методи діагностики знань з використанням адаптивних тестів // *УСиМ*. 2007. № 5. С. 68–76.

4. Аванесов В. С. Научные проблемы тестового контроля знаний. Москва : Учебный центр при исследовательском центре проблем качества подготовки специалистов, 1994. 135 с.

5. Федорук П. І. Технологія розробки навчального модуля в адаптивній системі дистанційного навчання та контролю знань // *Математичні машини і системи*. 2005. № 3. С.155–165.

6. Теслер Г. С. Новая кибернетика. Киев : Логос, 2004. 404 с.

7. Челышкова М. Б. Теория и практика конструирования педагогических тестов : Учебное пособие. Москва : Логос, 2002. 432 с.

8. Дюк В. А. Компьютерная психодиагностика. Санкт Петербург : Братство, 1994. 360 с.

9. Ивлиев М. К. Разработка тестовых заданий для компьютерного тестирования : Учебно-методическое пособие. Москва : ИМПЭ им. А. С. Грибоедова, 2001. 69 с.

10. Аванесов В. С. Теория и методика педагогических измерений. Москва : ЦТ и МКО УГТУ-УПИ, 2005. 98 с.

11. Кривицкий Б. Х. К вопросу о компьютерных программах учебного контроля знаний. // *Международный журнал "Образовательные технологии и общество"*. 2004. Т. 7. № 2. С. 158–169.

12. Аванесов В. С. Композиция тестовых заданий. Москва : Адепт, 2002. 240 с.

13. Рудинский И. Д., Соловей Е. В. Реализация алгоритмов прямого тестирования в интеллектуальной автоматизированной системе контроля знаний // Проблемы информатики в образовании, управлении, экономике и технике : Мат-лы Всероссийской научно-технической конференции. Пенза, 2001. – 2с.

14. Рудинский И. Д. Метод адаптивного автоматизированного контроля знаний // Проблемы информатики в образовании, управлении, экономике и технике : Мат-лы Всероссийской научно-технической конференции. Пенза, 2001. 4 с.

15. Челышкова М. Б. Разработка педагогических тестов на основе современных математических моделей : Уч. пособие. Москва : Исследовательский центр проблем качества подготовки специалистов, 1995.

ДОДАТОК А

Створення пунктів меню та розробка інтерфейсу тестування

```
//Скрипт для приема запроса с новой анкетой
   header("Content-Type: text/html; charset=windows-1251");
        include ‘Settings.php’;
        $MaxAnketeIndex=0;//максимальный индекс анкеты в БД
        $MaxTextIndex=0;//максимальный индекс текста в БД
        $MaxQuestionIndex=0;//максимальный индекс вопроса в БД
        $MaxAnswerIndex=0;//максимальный индекс ответат в БД
        //преобразовываем полученный запрос в массив
// Получить JSON как строку
        $json_str = file_get_contents(‘php://input’);
//!!!временно сохраним в журнал
        saveLog($json_str);
//Получить объект
        $json_obj = json_decode($json_str);
        $anketeName = $json_obj->Name;
        $QuestionsQnty = $json_obj->QuestionsQnty;
        $NumberOfRespondents = $json_obj->NumberOfRespondents;
        $Questions = $json_obj->Questions;//массив с вопросами
        saveLog(" ".$anketeName." ".$QuestionsQnty." ".$NumberOfRespondents);//!!!!!
        //соединяемся с БД
        $dbcnx = @mysql_connect($dblocation,$dbuser,$dbpasswd);
        if (!$dbcnx) // Если дескриптор равен 0 соединение не установлено
   {
    echo("ERROR_DB_CONNECT_0<br>");
    exit();
   }
        if ($dbcnx>0)
        {
                //echo("DB_CONNECT_OK<br>");
        }
// Код соединения с базой данных
        if (!@mysql_select_db($dbname, $dbcnx))
        {
        echo( "ERROR_DB_CONNECT_1<br>" );
        exit();
        }
//Указываем кодировку для записи в БД
        mysql_query("SET NAMES UTF8");
```

```
	mysql_query("SET CHARACTER SET UTF8");
//создаем новую анкету
	//проверяем наличие анкеты с таким же именем
	if (FindAnketeIndex($mysqli, $anketeName)>0) {
		echo("ERROR Ankete with this Name allrady exists");
		exit();
	}
	$MaxAnketeIndex=GetMaxAnketeIndex($mysqli);//максимальный индекс анкеты в БД
	$MaxAnketeIndex++;
	AddNewAnkete($mysqli,$MaxAnketeIndex, $anketeName, 1, $NumberOfRespondents);
//создаем новые вопросы
	$MaxTextIndex=GetMaxTextIndex($mysqli);//максимальный индекс текста в БД
	$MaxQuestionIndex=GetMaxQuestionIndex($mysqli);//максимальный индекс вопроса в БД
	$MaxAnswerIndex=GetMaxAnswerIndex($mysqli);//максимальный индекс ответат в БД

	for ($i=0; $i<$QuestionsQnty; $i++) {
		$QuestNumb=$Questions[$i]->Number;
		$QuestionText = $Questions[$i]->Text;
		$AnswersQnty = $Questions[$i]->AnswersQnty;
		$MaxAnswersTogether = $Questions[$i]->MaxAnswersTogether;
		$Answers = $Questions[$i]->Answers;
		saveLog(" ".$QuestNumb." ".$QuestionText." ".$AnswersQnty." ".$MaxAnswersTogether);//!!!!!!!!!!!!!
		if (TestTextAvailability($mysqli, $QuestionText)<1) {//если текст ранее не встречался – добавляем
			$MaxTextIndex++;
			AddNewText($mysqli,$MaxTextIndex, $QuestionText);//добавляем новый текст
			$MaxQuestionIndex++;
			AddNewQuestion($mysqli,$MaxQuestionIndex, $MaxAnketeIndex, $MaxTextIndex, $MaxAnswersTogether);//добавляем новый вопрос
		}
		else { //если такой текст уже был – указываем его индекс
			$ThisTextIndex = FindTextIndex($mysqli, $QuestionText);//Находим индекс текста в БД
			$MaxQuestionIndex++;
			AddNewQuestion($mysqli,$MaxQuestionIndex, $MaxAnketeIndex, $ThisTextIndex, $MaxAnswersTogether);//добавляем новый вопрос
		}
		//создаем новые ответы
		for ($j=0; $j<$AnswersQnty; $j++) {
			$AnswNumb = $Answers[$j]->AnswNumber;
			$AnswText = $Answers[$j]->AnswText;
			$AnswType = $Answers[$j]->Type;
			saveLog(" ".$AnswNumb." ".$AnswText." ".$AnswType);//!!!!!!!!!!!!!
```

```
                        if (TestTextAvailability($mysqli, $AnswText)<1) {//если текст ранее не встречался – добавляем
                                $MaxTextIndex++;
                                AddNewText($mysqli,$MaxTextIndex, $AnswText);//добавляем новый текст
                                $MaxAnswerIndex++;
                                AddNewAnswer($mysqli,$MaxAnswerIndex, $MaxQuestionIndex, $AnswType, $MaxTextIndex);//добавляем новый ответ
                        }
                        else { //если такой текст уже был – указываем его индекс
                                $ThisTextIndex = FindTextIndex($mysqli, $AnswText);//Находим индекс текста в БД
                                $MaxAnswerIndex++;
                                AddNewAnswer($mysqli,$MaxAnswerIndex, $MaxQuestionIndex, $AnswType, $ThisTextIndex);//добавляем новый ответ
                        }
                }
        }
        echo("Ankete SENDED OK");
//узнаем максимальный индекс анкет на сервере
function GetMaxAnketeIndex($mysqli){
        $query="SELECT MAX(id) as _maxID FROM questionnaires;";
        $res = mysql_query($query);
        if ($res){
                while($row = mysql_fetch_array($res)) {
                        $result=$row['_maxID'];
                }
        }
        return $result;
}
//узнаем максимальный индекс текста на сервере
function GetMaxTextIndex($mysqli){
        $query="SELECT MAX(id) as _maxID FROM texts;";
        $res = mysql_query($query);
        if ($res){
                while($row = mysql_fetch_array($res)) {
                        $result=$row['_maxID'];
                }
        }
        return $result;
}
//узнаем максимальный индекс вопроса на сервере
function GetMaxQuestionIndex($mysqli){
        $query="SELECT MAX(id) as _maxID FROM questions;";
        $res = mysql_query($query);
        if ($res){
```

```
                while($row = mysql_fetch_array($res)) {
                        $result=$row['_maxID'];
                }
        }
        return $result;
}
//узнаем максимальный индекс ответа на сервере
function GetMaxAnswerIndex($mysqli){
        $query="SELECT MAX(id) as _maxID FROM answers;";
        $res = mysql_query($query);
        if ($res){
                while($row = mysql_fetch_array($res)) {
                        $result=$row['_maxID'];
                }
        }
        return $result;
}
//добавляем новый текст с индексом в БД
function AddNewText($mysqli,$textid, $newtext){
        $query="INSERT INTO texts VALUES (".$textid.", '".$newtext."');";
        $res = mysql_query($query);
}
//добавляем новую анкету с индексом в БД
function AddNewAnkete($mysqli,$anketeid, $name, $author, $respondents){
        $datenow=date('Y-m-d', time());
        $query="INSERT INTO questionnaires VALUES (".$anketeid.", '".$name."', ".$author.",
'".$datenow."', ".$respondents.");";
        $res = mysql_query($query);
}
//добавляем новый вопрос с индексом в БД
function AddNewQuestion($mysqli,$quest_id, $ankete_id, $qtext_id, $answers_lim){
        $query="INSERT INTO questions VALUES(".$quest_id.", ".$ankete_id.", 0, 0,
".$qtext_id.", ".$answers_lim.");";
        $res = mysql_query($query);
}
//добавляем новый ответ с индексом в БД
function AddNewAnswer($mysqli,$answ_id, $quest_id, $answ_type, $qtext_id){
        $query="INSERT INTO answers VALUES(".$answ_id.", ".$quest_id.", ".$answ_type.",
".$qtext_id.");";
        $res = mysql_query($query);
}
//Проверяем наличие текста в БД – возращает сколько раз текст встречается в БД
function TestTextAvailability($mysqli, $testtext){
        $query="SELECT COUNT(*) as _textQnty FROM texts WHERE text_var='".$testtext."';";
        $res = mysql_query($query);
        if ($res){
```

```
		while($row = mysql_fetch_array($res)) {
			$result=$row[‘_textQnty’];
		}
	}
	return $result;
}
//Находим индекс текста в БД
function FindTextIndex($mysqli, $testtext) {
	$query="SELECT MAX(id) as _textID FROM texts WHERE text_var=‘".$testtext."‘;";
	$res = mysql_query($query);
	if ($res){
		while($row = mysql_fetch_array($res)) {
			$result=$row[‘_textID’];
		}
	}
	return $result;
}
//Находим индекс анкеты с указанным именем в БД
function FindAnketeIndex($mysqli, $testname) {
	$query="SELECT MAX(id) as _anketeID FROM questionnaires WHERE
Name=‘".$testname."‘;";
	$res = mysql_query($query);
	if ($res){
		while($row = mysql_fetch_array($res)) {
			$result=$row[‘_anketeID’];
		}
	}
	return $result;
}
//запись запросов в журнал
function saveLog($string){
	$file = fopen(‘logs.txt’, ‘a’);
	$serverdate = date(‘d/m/Y h:i:s a’, time());
	$clientIP=$_SERVER[‘REMOTE_ADDR’];
	fwrite($file, $serverdate." ".$clientIP.":".$string."\r\n");
	fclose($file);
}
```