

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

КВАЛІФІКАЦІЙНА РОБОТА
на здобуття ступеня вищої освіти бакалавра
зі спеціальності 121 – Інженерія програмного забезпечення

На тему: Розробка системи адаптивного навчання на основі поведінки користувача

Засвідчую, що в цій
кваліфікаційній роботі
немає запозичень із праць
інших авторів без
відповідних посилань.
Студент гр. ІПЗ-22-1
_____/О.О.Макусінський /

Керівник кваліфікаційної роботи	_____	/ <u>Д. В. Швець</u> /
Завідувач кафедри	_____	/ <u>А. М. Стрюк</u> /

Криворізький національний університет
Факультет: Інформаційних технологій
Кафедра: Моделювання та програмного забезпечення
Ступінь вищої освіти: бакалавр
Спеціальність: 121 – Інженерія програмного забезпечення

ЗАТВЕРДЖУЮ:

Зав. кафедри

_____ А. М. Стрюк

«__» _____ 20__р.

ЗАВДАННЯ

на кваліфікаційну роботу

студенту групи ІПЗ-22-1 Олексію Олеговичу Макусінському

1. Тема: Розробка системи адаптивного навчання на основі поведінки користувача затверджено наказом по КНУ № 22 від «28» листопада 2025 р.
2. Термін подання студентом закінченої роботи: «20» травня 2026р.
3. Вихідні дані по роботі: розроблювана програмна система повинна забезпечувати введення теоретичного матеріалу користувачем, автоматичне формування тестових завдань, адаптацію рівня складності залежно від результатів проходження тестів, а також збереження та обробку результатів користувача.
4. Зміст пояснювальної записки (перелік питань, що їх треба розробити): виконати аналіз існуючих систем адаптивного навчання та підходів до формування тестових завдань; визначити вимоги до функціональності та інтерфейсу програмної системи; спроєктувати структуру та архітектуру системи адаптивного навчання; реалізувати механізм генерації тестів на основі введеного тексту; реалізувати алгоритм зміни рівня складності відповідно до результатів користувача; здійснити програмну реалізацію системи засобами мови програмування Java; реалізувати взаємодію з базою даних для збереження історії проходження тестів; провести тестування розробленої програмної системи.
5. Перелік ілюстративного матеріалу: діаграма класів програмної системи, діаграма взаємодії користувача із системою, блок-схема алгоритму роботи програми, схема взаємодії програмних модулів, функціональна схема системи, схема бази даних, знімки екранних форм інтерфейсу програми.

Зворотна сторінка аркуша завдання на кваліфікаційну роботу

Календарний план:

№	Найменування етапів кваліфікаційної роботи	Термін виконання етапів роботи
1	Огляд літературних джерел за тематикою адаптивного навчання та аналіз сучасних навчальних систем	27.02.2026 – 02.03.2026
2	Проведення порівняльного аналізу методів планування розкладу в мобільних додатках для освітян	03.03.2026 – 10.03.2026
3	Проведення аналізу методів формування тестових завдань та способів генерації питань	11.03.2026 – 17.03.2026
4	Підготовка матеріалів першого розділу роботи	18.03.2026 – 24.03.2026
5	Розроблення функціональної схеми та алгоритму роботи системи	25.03.2026 – 31.03.2026
6	Підготовка матеріалів другого розділу роботи	01.04.2026 – 06.04.2026
7	Підготовка матеріалів третього розділу роботи	07.04.2026 – 18.04.2026
8	Розроблення програмного забезпечення системи адаптивного навчання та реалізація функціональних модулів	19.04.2026 – 12.05.2026
9	Реалізація взаємодії з базою даних та тестування роботи програмної системи	13.05.2026 – 20.05.2026

Дата видачі завдання:

«26» лютого 2026 р.

Студент

_____/О.О.Макусінський/

Керівник роботи

_____/Д. В. Швець/

РЕФЕРАТ

АДАПТИВНЕ НАВЧАННЯ, ПОВЕДІНКА КОРИСТУВАЧА,
ПЕРСОНАЛІЗАЦІЯ НАВЧАННЯ, ТЕСТУВАННЯ, БАЗА ДАНИХ,
JAVA, MYSQL.

Пояснювальна записка: 57 с., 1 табл., 13 рис., 2 дод., 14 джерел.

Метою кваліфікаційної роботи є розробка системи адаптивного навчання на основі поведінки користувача для підвищення ефективності освітнього процесу шляхом автоматичного налаштування навчального матеріалу відповідно до рівня знань користувача.

У роботі проведено аналіз сучасних систем адаптивного навчання та методів персоналізації освітнього процесу. Досліджено підходи до формування навчальних матеріалів і адаптації рівня складності завдань залежно від результатів користувача.

Розроблено структуру системи, що включає модулі генерації завдань, аналізу відповідей користувача та збереження результатів навчання. Реалізовано автоматичне створення тестових завдань на основі введеного матеріалу та зміну складності подальших завдань залежно від успішності проходження попередніх рівнів.

Програмну реалізацію виконано мовою Java з використанням бази даних MySQL[3] у середовищі IntelliJ IDEA[7]. Проведене тестування підтвердило правильність роботи та стабільність системи.

Результатом роботи є функціональна система адаптивного навчання, яка може бути використана для персоналізації процесу навчання та підвищення якості засвоєння знань.

ABSTRACT

ADAPTIVE LEARNING, USER BEHAVIOR, LEARNING PERSONALIZATION, TESTING, DATABASE, JAVA, MYSQL.

Thesis in 57 p., 1 tab., 13 fig., 2 app., 14 references.

The purpose of this qualification work is to develop an adaptive learning system based on user behavior in order to improve the effectiveness of the educational process through automatic adjustment of learning materials according to the user's knowledge level.

The work includes an analysis of modern adaptive learning systems and methods for personalizing the educational process. Approaches to generating educational materials and adapting task complexity depending on user performance have been studied.

The structure of the system has been developed, including modules for task generation, user response analysis, and storage of learning results. Automatic generation of test tasks based on the entered material has been implemented, along with adjustment of the complexity of further tasks depending on the success of previous levels.

The software implementation was developed in Java using the MySQL database in the IntelliJ IDEA development environment. Testing confirmed the correctness and stability of the system.

The result of this qualification work is a functional adaptive learning system that can be used to personalize the learning process and improve the quality of knowledge acquisition.

ЗМІСТ

РЕФЕРАТ	4
ВСТУП.....	7
1 АКТУАЛЬНІСТЬ КВАЛІФІКАЦІЙНОЇ РОБОТИ.....	8
1.1 Критичний аналіз літературних джерел за темою.....	8
1.2 Аналіз існуючих аналогів	9
1.3 Актуальність теми кваліфікаційної роботи	11
1.4 Цілі та завдання кваліфікаційної роботи	13
2 РОЗРОБКА ФУНКЦІОНАЛЬНОЇ СХЕМИ І АЛГОРИТМУ	16
2.1 Розробка та докладний опис функціональної схеми програми	16
2.2 Розробка та докладний опис алгоритму роботи програми	19
2.3 Розробка інтерфейсу програми	22
3 РОЗРОБКА БАЗИ ДАНИХ І ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ	28
3.1 Аналіз обраної середовища програмування	28
3.2 Розробка бази даних.....	29
3.3 Програмна реалізація основних функцій	30
3.4 Методика роботи користувача.....	33
ВИСНОВОК	35
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	36
ДОДАТОК А.....	38
ДОДАТОК Б	57

ВСТУП

Розвиток інформаційних технологій значно вплинув на сферу освіти, зробивши можливим створення інтелектуальних систем навчання, які враховують індивідуальні особливості користувача. Сучасні освітні платформи поступово переходять від статичного подання матеріалу до динамічного, де навчальний процес змінюється залежно від результатів і поведінки користувача.

Особливої актуальності набувають системи, що здатні аналізувати рівень засвоєння знань та автоматично підлаштовувати складність завдань. У традиційних підходах навчання всі користувачі отримують однаковий матеріал, що не завжди є ефективним, оскільки не враховує індивідуальний темп навчання та рівень підготовки.

У зв'язку з цим виникає потреба у створенні програмного рішення, яке забезпечує адаптацію навчального процесу на основі результатів користувача. Такі системи дозволяють підвищити якість засвоєння матеріалу, роблячи навчання більш гнучким та персоналізованим.

Метою даної роботи є розробка системи адаптивного навчання, яка автоматично формує навчальні завдання, аналізує відповіді користувача та змінює рівень складності залежно від його успішності.

Реалізація подібної системи є актуальною в умовах цифровізації освіти та сприяє підвищенню ефективності навчального процесу за рахунок індивідуального підходу до кожного користувача.

1 АКТУАЛЬНІСТЬ КВАЛІФІКАЦІЙНОЇ РОБОТИ

1.1 Критичний аналіз літературних джерел за темою

У сучасних умовах цифровізації освіти все більшої популярності набувають системи адаптивного навчання, які дозволяють змінювати складність навчального матеріалу залежно від результатів користувача. Такі системи спрямовані на підвищення ефективності навчання та індивідуалізацію освітнього процесу.

Під час аналізу літературних джерел було встановлено, що більшість існуючих рішень у цій сфері використовують складні алгоритми машинного навчання або штучного інтелекту для побудови адаптивних моделей. Водночас такі підходи є досить складними у реалізації та потребують значних обчислювальних ресурсів і великих обсягів даних.[1]

У більш простих навчальних системах адаптація часто реалізується через базові правила: підвищення або зниження рівня складності залежно від відсотка правильних відповідей користувача. Саме такий підхід є більш доступним для реалізації в навчальних проєктах і дозволяє досягти основної мети — адаптації навчання без використання складних алгоритмів штучного інтелекту.

У ході аналізу також було відзначено, що значна частина існуючих систем недостатньо враховує поведінку користувача під час проходження тестів, обмежуючись лише підрахунком балів. Це створює можливість для розробки більш гнучких рішень, які можуть змінювати типи завдань (наприклад, тестові запитання, пропуски у реченнях або пояснювальні питання) залежно від результатів користувача.

Таким чином, актуальність даної роботи полягає у створенні простої, але ефективної системи адаптивного навчання, яка базується на правилах оцінювання результатів користувача та дозволяє поступово змінювати

складність навчальних завдань без використання складних алгоритмів машинного навчання.

1.2 Аналіз існуючих аналогів

Аналіз існуючих аналогів систем адаптивного навчання дозволяє визначити основні підходи до реалізації персоналізованого навчального процесу, а також виявити їхні сильні та слабкі сторони з точки зору архітектури, функціональності та зручності використання. Серед найбільш поширених та відомих платформ у цій сфері можна виділити Duolingo, Moodle та Coursera, які демонструють різні підходи до реалізації адаптивності та організації навчального контенту.

Платформа Duolingo[8] є яскравим прикладом сучасної системи з високим рівнем інтегрованої адаптивності, яка базується на постійному аналізі відповідей користувача та автоматичному коригуванні складності завдань у реальному часі. Система формує індивідуальну навчальну траєкторію, підлаштовуючи вправи під рівень знань користувача та повторюючи проблемні теми для закріплення матеріалу. Такий підхід значно підвищує залученість користувачів, оскільки навчання відбувається у вигляді коротких інтерактивних сесій із миттєвим зворотним зв'язком. Водночас слід зазначити, що Duolingo має обмежену глибину налаштувань для кінцевого користувача та орієнтована переважно на конкретну предметну область — вивчення мов, що обмежує її універсальність у застосуванні для інших навчальних дисциплін.

Платформа Moodle[9] представляє інший підхід до організації навчального процесу, оскільки є повноцінною системою управління навчанням, яка дозволяє викладачам створювати курси, керувати матеріалами та налаштовувати структуру навчального процесу. Moodle підтримує елементи адаптації через використання тестів, умов доступу до матеріалів та додаткових модулів, однак базова система не містить складних вбудованих алгоритмів адаптивного навчання. Для реалізації більш

просунутій адаптивності зазвичай необхідно встановлювати додаткові плагіни або здійснювати кастомізацію системи, що потребує додаткових технічних знань і часу на розробку. Попри це, основною перевагою Moodle є її відкритість, гнучкість та можливість масштабування під потреби конкретного навчального закладу або курсу.

Coursera[10], у свою чергу, представляє більш сучасний і комплексний підхід до онлайн-освіти, поєднуючи елементи адаптивного навчання з аналітикою великих даних та рекомендаційними системами. Платформа аналізує поведінку користувача, його успішність, активність та прогрес, після чого пропонує індивідуальні рекомендації щодо курсів або окремих тем. Таким чином формується умовно персоналізована навчальна траєкторія, яка дозволяє користувачеві ефективніше досягати своїх освітніх цілей. Проте варто зазначити, що внутрішні алгоритми адаптації є закритими для користувача і розробника, а сама система значною мірою залежить від складних аналітичних моделей, які не завжди доступні для модифікації або інтеграції у власні проєкти.

Таким чином, аналіз існуючих рішень у сфері адаптивного навчання показує, що більшість сучасних платформ або використовують складні інтелектуальні алгоритми та закриті системи обробки даних, або ж реалізують адаптивність у вигляді обмежених правил і сценаріїв без глибокої персоналізації. Це, у свою чергу, підтверджує доцільність розробки власної програмної системи адаптивного навчання, яка поєднуватиме простоту реалізації, прозору логіку роботи та достатній рівень адаптації на основі поведінки користувача, що дозволить забезпечити ефективний навчальний процес у межах навчального проєкту.

Таблиця 1 - Порівняльна таблиця аналогів

№	Критерій	Duolingo	Moodle	Coursera
1	Тип системи	Мобільний додаток	LMS платформа	Онлайн-платформа
2	Рівень адаптивності	Високий	Середній	Високий

		(автоматичний)	(через правила)	(алгоритмічний)
№	Критерій	Duolingo	Moodle	Coursera
3	Використання ІІІ	Так	Обмежено	Так
4	Гнучкість налаштування	Низька	Висока	Середня
5	Складність реалізації	Висока	Середня	Висока
6	Орієнтація	Вузька (мови)	Універсальна	Універсальна
7	Підходить для власної розробки	Ні	Так	Частково

1.3 Актуальність теми кваліфікаційної роботи

Сьогодні освітній процес активно переходить у цифровий формат, що супроводжується стрімким збільшенням кількості онлайн-платформ, електронних освітніх ресурсів, тестових систем та програмних засобів підтримки навчання. Використання інформаційних технологій дозволяє зробити процес отримання знань більш доступним, гнучким та зручним для користувачів, забезпечуючи можливість навчатися незалежно від місця перебування та часу. Особливо актуальними стали дистанційні та змішані форми навчання[11], які потребують використання сучасних програмних рішень для організації взаємодії між навчальним матеріалом і користувачем.

Незважаючи на значний розвиток електронних освітніх систем, більшість існуючих платформ залишаються переважно статичними та використовують однаковий підхід до всіх користувачів. Навчальні матеріали та тестові завдання часто формуються за єдиним шаблоном без урахування індивідуальних особливостей, рівня знань, швидкості засвоєння інформації та результатів попередньої діяльності користувача. Такий підхід не забезпечує достатньої гнучкості та не дозволяє ефективно адаптувати процес навчання під конкретні потреби людини.

У практичному використанні це може призводити до ситуацій, коли користувач отримує навчальний матеріал або завдання, рівень складності яких не відповідає його поточній підготовці. Для частини користувачів запропоновані завдання можуть бути надто складними, що створює труднощі

у процесі навчання та може спричинити втрату інтересу до подальшого проходження курсу. Водночас інші користувачі можуть отримувати надто прості завдання, які не сприяють розвитку необхідних навичок і не створюють достатнього рівня навчального навантаження. У результаті ефективність освітнього процесу знижується, а рівень мотивації користувачів поступово погіршується.

Крім того, важливим фактором є необхідність врахування поведінки користувача під час роботи із системою. До таких параметрів можуть належати результати виконання тестів, кількість правильних і неправильних відповідей, швидкість проходження завдань, частота помилок та інші характеристики взаємодії із системою. Аналіз подібних показників дозволяє визначати рівень засвоєння матеріалу та своєчасно змінювати складність навчального контенту відповідно до отриманих результатів.

Слід зазначити, що значна кількість сучасних адаптивних систем орієнтована на використання складних методів аналізу даних та алгоритмів штучного інтелекту або машинного навчання. Подібні підходи можуть забезпечувати високий рівень точності адаптації, проте їх реалізація потребує значних обчислювальних ресурсів, великого обсягу навчальних даних та складних механізмів обробки інформації. Для невеликих освітніх проєктів, навчальних систем або студентських розробок використання таких технологій не завжди є доцільним через підвищену складність реалізації та необхідність додаткових ресурсів.

У зв'язку з цим виникає потреба у створенні більш простих та ефективних рішень, які дозволяють реалізувати механізм адаптації без використання складних моделей штучного інтелекту. Одним із можливих підходів є використання логічних правил та аналізу поведінки користувача для автоматичної зміни рівня складності завдань. Такий підхід дозволяє створити систему, яка буде реагувати на результати навчання в режимі реального часу, поступово коригуючи навчальний процес відповідно до поточного рівня знань користувача.

Отже, розробка системи адаптивного навчання на основі поведінки користувача є актуальним завданням, оскільки дозволяє підвищити ефективність навчального процесу, забезпечити індивідуальний підхід до користувачів та покращити якість засвоєння навчального матеріалу. Використання простих механізмів адаптації робить таке рішення практичним для реалізації в межах навчальних проєктів і дозволяє поєднати відносну простоту програмної реалізації з функціональністю та зручністю використання системи.

1.4 Цілі та завдання кваліфікаційної роботи

Метою кваліфікаційної роботи є розробка простої програмної системи адаптивного навчання, яка враховує результати користувача під час проходження тестових завдань і автоматично змінює їх складність залежно від рівня його успішності. Така система орієнтована на підтримку більш індивідуального підходу до навчального процесу та підвищення ефективності засвоєння матеріалу за рахунок поступового ускладнення або спрощення завдань. Основною ідеєю є забезпечення гнучкого процесу навчання, при якому користувач отримує завдання, що максимально відповідають його поточному рівню знань і навичок. Завдяки цьому процес проходження навчального матеріалу може стати більш ефективним, оскільки користувач не буде перевантажений занадто складними завданнями або втрачати інтерес через надто прості питання. Крім цього, система повинна сприяти підтримці зацікавленості користувача у процесі навчання та забезпечувати більш послідовний розвиток знань шляхом поступового переходу між рівнями складності. Реалізація подібного підходу дозволяє зробити процес навчання більш динамічним і наближеним до індивідуальних потреб користувача. Також важливим аспектом є створення зручного програмного інструменту, який може використовуватись для автоматизації контролю знань та аналізу результатів проходження тестування.

Особливість даної роботи полягає у тому, що адаптація реалізується не

через складні алгоритми або моделі машинного навчання, а за допомогою простих правил логіки, заснованих на результатах проходження тестів. Це робить систему більш зрозумілою для реалізації в межах навчального проєкту та дозволяє зосередитися на практичній частині розробки програмного забезпечення. Використання такого підходу дає можливість спростити архітектуру системи та зменшити кількість додаткових технологій, необхідних для її створення. На відміну від складних інтелектуальних систем, які потребують великих масивів даних та тривалого навчання моделей, запропоноване рішення базується на аналізі безпосередніх результатів роботи користувача. Система визначає рівень успішності на основі кількості правильних відповідей, отриманих балів або інших показників тестування та відповідно змінює параметри наступних завдань. Подібний механізм дозволяє отримати адаптивну поведінку системи без значного ускладнення програмної реалізації. Крім того, така модель є зручною для подальшого вдосконалення, оскільки в майбутньому до неї можна додати додаткові критерії оцінювання або розширити функціональні можливості системи.

Для досягнення поставленої мети необхідно виконати такі завдання:

- а) дослідити існуючі підходи до побудови адаптивних навчальних систем та визначити основні принципи їх роботи, зокрема способи зміни складності навчального матеріалу;
- б) проаналізувати методи оцінювання результатів користувача під час проходження тестів, включаючи підрахунок правильних відповідей та визначення рівня підготовки;
- в) визначити просту логіку адаптації, яка дозволяє змінювати рівень складності завдань залежно від отриманого результату, без використання складних алгоритмів обробки даних;
- г) спроектувати структуру програмної системи, визначивши основні компоненти, їх функції та взаємодію між собою;
- д) реалізувати механізм створення та проходження тестів різного

рівня складності, які можуть змінюватися в процесі роботи системи;

- е) розробити систему підрахунку результатів користувача, а також механізм їх збереження для подальшого використання;
- ж) реалізувати базову адаптивну логіку, яка буде змінювати складність наступних завдань відповідно до результатів попередніх відповідей користувача;
- з) перевірити працездатність розробленої системи на різних сценаріях використання, включаючи випадки високої та низької успішності користувача.

2 РОЗРОБКА ФУНКЦІОНАЛЬНОЇ СХЕМИ І АЛГОРИТМУ

2.1 Розробка та докладний опис функціональної схеми програми

Функціональна схема розроблюваної системи адаптивного навчання відображає основні етапи роботи програми та взаємодію її ключових компонентів. Система побудована таким чином, щоб забезпечити повний цикл навчання користувача: від проходження тестів до аналізу результатів і зміни рівня складності наступних завдань.



Рисунок 1.1 Контексна діаграма DFD [6]

Після запуску програми користувач переходить до основного інтерфейсу проходження тестування, який є центральною частиною взаємодії із системою. На даному етапі система завантажує необхідний набір тестових завдань та відображає їх відповідно до поточного рівня складності. Початковий рівень може встановлюватися автоматично за визначеними параметрами або базуватися на результатах попереднього проходження тестів, якщо користувач уже працював із системою раніше. Під час проходження тесту користувач послідовно відповідає на запитання, взаємодіючи з графічним інтерфейсом системи. Після вибору або введення відповіді інформація передається до внутрішніх модулів системи для подальшої обробки. Крім того, система може здійснювати контроль коректності введених даних та забезпечувати швидкий перехід між запитаннями для зручності користувача.

Модуль перевірки відповідей виконує аналіз правильності наданих

відповідей та здійснює їх порівняння з інформацією, яка зберігається в базі даних або у внутрішній структурі тестів. У процесі перевірки система визначає кількість правильних і неправильних відповідей, обчислює підсумковий результат та формує показники успішності проходження тестування. Отриманий результат може відображатися у вигляді кількості набраних балів, відсотка правильних відповідей або рівня засвоєння матеріалу. Такі показники дозволяють оцінити поточний рівень знань користувача та використовуються як основа для подальшої роботи адаптивного механізму. Додатково результати можуть бути представлені у зрозумілому вигляді для користувача, що дозволяє швидко оцінити власний прогрес та визначити сильні й слабкі сторони підготовки.

Після завершення перевірки результатів починає працювати модуль адаптації складності, який є одним із ключових компонентів системи. Його основним завданням є аналіз отриманих результатів та зміна параметрів подальшого тестування відповідно до визначених правил. Логіка адаптації реалізована на основі простих умов, які враховують рівень успішності користувача. Наприклад, якщо користувач показує високий результат та правильно відповідає на більшість запитань, система може автоматично підвищити рівень складності наступних тестів. Якщо ж результат виявляється недостатнім, складність може залишатися незмінною або знижуватися для полегшення процесу засвоєння матеріалу. Такий механізм дозволяє створити більш індивідуальний підхід до навчання та забезпечити поступове підвищення рівня знань користувача без різких переходів між складними та простими завданнями.

Важливою складовою системи є модуль зберігання даних, який відповідає за накопичення та збереження інформації про результати роботи користувача. У ньому можуть зберігатися результати тестів, дата проходження, кількість набраних балів, рівень складності та інші параметри, необхідні для подальшого аналізу. Наявність такого механізму дозволяє використовувати попередню інформацію під час повторного запуску

програми та забезпечує можливість відстеження змін у результатах користувача протягом певного періоду часу. Крім цього, збереження даних створює основу для реалізації додаткових функцій, таких як перегляд історії тестувань, аналіз успішності або формування статистичних показників.

Інтерфейс користувача забезпечує взаємодію з усіма компонентами системи та виконує роль засобу обміну інформацією між користувачем і програмою. Через графічний інтерфейс користувач отримує тестові завдання, вводить відповіді, переглядає результати тестування та може переходити між різними елементами системи. При розробці інтерфейсу особлива увага приділяється простоті використання, логічному розташуванню елементів керування та мінімізації кількості зайвих дій. Це дозволяє зробити процес проходження тестів більш зрозумілим та комфортним для користувача. Зручний інтерфейс також сприяє зменшенню часу на освоєння системи та забезпечує більш ефективну взаємодію користувача з функціональними можливостями програмного продукту.

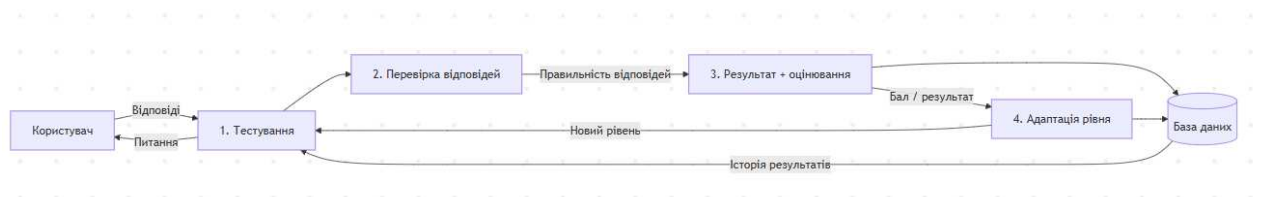


Рисунок 2.2 Декомпозиція DFD

Таким чином, функціональна схема системи включає наступні основні етапи: генерацію тестових завдань[13], проходження тесту користувачем, перевірку відповідей, розрахунок результату, адаптацію рівня складності та збереження даних для подальшого використання. Така структура забезпечує логічну послідовність роботи програми та реалізацію базового принципу адаптивного навчання.

2.2 Розробка та докладний опис алгоритму роботи програми

Алгоритм роботи програмної системи адаптивного навчання побудований на послідовному виконанні декількох взаємопов'язаних етапів: формування тестових завдань, проходження тесту користувачем, перевірка відповідей, обчислення результатів та автоматична зміна рівня складності наступного тесту. Загальна логіка функціонування системи базується на використанні простих правил і визначених умов, що дозволяє реалізувати адаптивний механізм без використання складних математичних моделей або алгоритмів машинного навчання. Такий підхід забезпечує простоту програмної реалізації та одночасно дозволяє створити динамічний процес навчання, який реагує на дії користувача. Основною перевагою такого алгоритму є можливість поступового налаштування навчального процесу залежно від результатів проходження тестів, що сприяє більш ефективному засвоєнню навчального матеріалу.

На початку роботи програми система виконує етап початкового налаштування, під час якого встановлюється стартовий рівень складності тестових завдань. Початковий рівень може бути заданий автоматично або визначатися на основі попередньо збережених результатів користувача. Якщо користувач працює із системою вперше, встановлюється базовий рівень складності, який дозволяє оцінити початковий рівень знань. Після цього програма завантажує необхідний набір тестових запитань, які відповідають поточному рівню складності. Формування набору питань може здійснюватися шляхом вибору відповідних записів із бази даних або внутрішнього сховища системи. Після завантаження тестових завдань система відображає їх через графічний інтерфейс користувача.

На етапі проходження тестування користувач послідовно переглядає запитання та надає відповіді. Взаємодія між користувачем і системою здійснюється через елементи інтерфейсу, які забезпечують зручний вибір або введення відповідей. Після вибору відповіді інформація передається до модуля перевірки, де здійснюється її тимчасове збереження та подальша

обробка. Цей процес повторюється для всіх запитань тесту до моменту завершення проходження користувачем усього набору завдань. Додатково система може здійснювати перевірку коректності введених даних, контролювати кількість відповідей або не дозволяти завершити тестування без проходження всіх етапів.

Після завершення тесту запускається етап обробки результатів. На цьому етапі система виконує аналіз усіх отриманих відповідей та порівнює їх із правильними варіантами, які зберігаються у відповідній структурі даних. Після перевірки здійснюється підрахунок кількості правильних відповідей, неправильних відповідей та загального рівня успішності проходження тесту. Отримані результати можуть бути представлені у вигляді кількості балів, відсотка правильних відповідей або іншого показника оцінювання. Обчислення результатів дозволяє оцінити рівень засвоєння матеріалу користувачем та підготувати необхідні дані для подальшої роботи адаптивного механізму системи.

Наступним і одним із найважливіших етапів є адаптація рівня складності тестових завдань. Після визначення результатів проходження тесту система аналізує отримані показники відповідно до визначених правил. Якщо користувач демонструє високий результат і набирає значний відсоток правильних відповідей, наприклад понад 70–80 %, система автоматично підвищує рівень складності наступних завдань. Це дозволяє поступово збільшувати навантаження та забезпечувати розвиток знань користувача. Якщо результат знаходиться у середньому діапазоні, система залишає поточний рівень без змін, оскільки це свідчить про необхідність закріплення вже отриманих знань. У випадку отримання низького результату складність тестів може бути знижена, що дозволяє користувачу повторно опрацювати навчальний матеріал у більш доступній формі та уникнути надмірного навантаження.

Після визначення нового рівня складності система виконує збереження результатів проходження тестування. До сховища даних записується

інформація про кількість правильних відповідей, отриманий результат, рівень складності та інші параметри, що можуть використовуватися для аналізу прогресу користувача. Збереження результатів дозволяє забезпечити безперервність навчального процесу та використовувати попередні дані при подальшій роботі системи. Крім цього, накопичення інформації створює можливість для побудови статистики, перегляду історії проходження тестів та аналізу змін рівня знань користувача протягом тривалого часу.



Рисунок 3.3 Діаграма адаптації складності

Алгоритм роботи програми можна умовно поділити на такі основні кроки:

- а) ініціалізація системи та встановлення початкового рівня складності;
- б) завантаження тестових завдань відповідного рівня;
- в) відображення питань користувачу та отримання відповідей;
- г) перевірка відповідей та підрахунок результату;
- д) визначення відсотка успішності;

- е) зміна рівня складності відповідно до результату;
- ж) збереження результатів та підготовка до наступного тесту.

Таким чином, алгоритм забезпечує циклічну роботу системи, де кожен новий тест формується на основі результатів попереднього, що і реалізує базовий принцип адаптивного навчання.

Функціональна схема системи адаптивного навчання



Рисунок 4.4 Діаграма алгоритму програми[5]

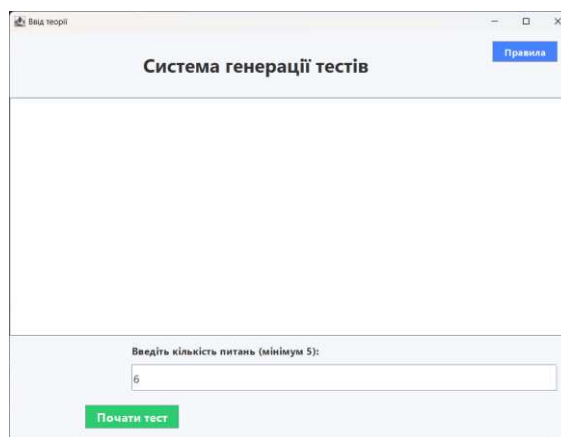
2.3 Розробка інтерфейсу програми

Інтерфейс програмної системи адаптивного навчання розроблено з урахуванням принципів максимальної простоти використання, інтуїтивної зрозумілості та мінімізації зайвих дій користувача під час взаємодії з програмою. Оскільки ключовою метою системи є проходження тестів, оцінювання результатів та адаптація рівня складності відповідно до поведінки користувача, інтерфейс побудовано таким чином, щоб забезпечити швидкий доступ до основних функцій без перевантаження зайвими

графічними елементами або складними навігаційними структурами.

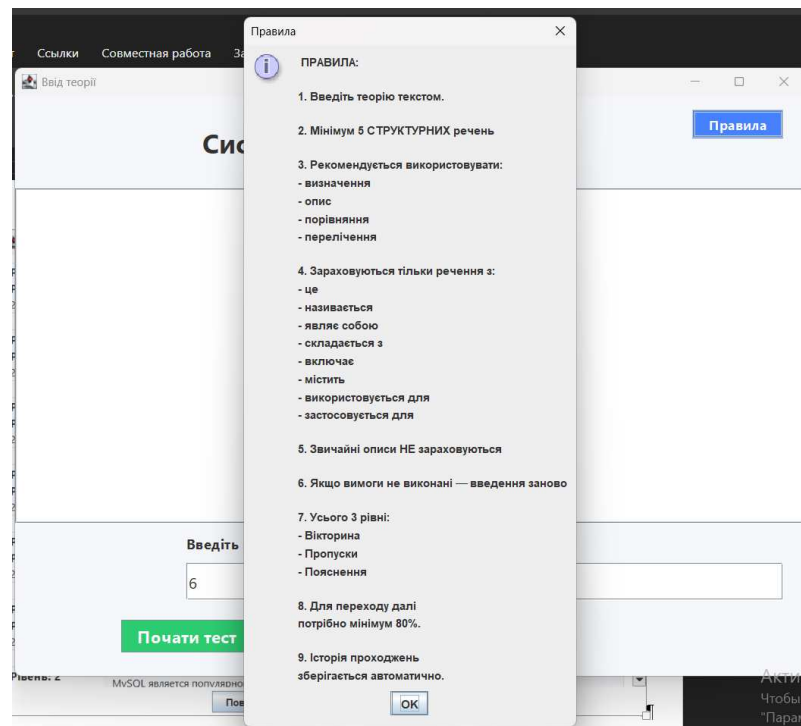
Під час розробки програмного забезпечення було використано графічний інтерфейс користувача, реалізований на основі технології Java Swing[2], яка є стандартним інструментом для створення настільних (desktop) застосунків мовою Java. Дана технологія дозволяє реалізовувати віконні програми з базовими та розширеними елементами взаємодії, такими як кнопки, текстові поля, перемикачі, списки, таблиці та текстові області. Використання Java Swing є доцільним у межах навчального проєкту, оскільки воно забезпечує достатній рівень функціональності при відносно простій реалізації та не потребує складної додаткової конфігурації, як у випадку сучасних фреймворків.

Головне вікно програми виконує роль стартового екрану та центральної точки навігації. Саме з нього користувач розпочинає взаємодію із системою. На цьому екрані відображається поточний рівень складності, якщо він уже визначений на основі попередніх результатів, а також основна кнопка запуску тестування. Додатково можуть бути розміщені елементи для перегляду історії проходження тестів, статистики або налаштувань користувача, якщо такі передбачені функціоналом системи. Після натискання кнопки початку тесту користувач переходить до інтерфейсу проходження завдань.



2.5 Скріншот стартової сторінки для вводу даних

Окрему увагу приділено інформаційному наповненню головного вікна. У ньому передбачено розділ із правилами користування системою, який містить детальні інструкції щодо введення теоретичного матеріалу, вимоги до мінімальної кількості коректних відповідей або речень, а також умови переходу між рівнями складності. Крім того, користувач може ознайомитися зі структурою рівнів адаптації, що дозволяє краще зрозуміти логіку роботи системи та очікувані результати навчання. Такий підхід підвищує прозорість роботи програми та зменшує кількість помилок, пов'язаних із неправильним використанням функцій.



2.6 Скріншот вікна з правилами

Екран тестування є основним робочим модулем інтерфейсу. На ньому послідовно відображаються питання, сформовані на основі введеного теоретичного матеріалу або попередньо підготовленої бази тестів. Для кожного питання користувачу пропонуються варіанти відповідей, серед яких необхідно обрати один або декілька правильних. Після вибору відповіді користувач переходить до наступного питання за допомогою відповідної

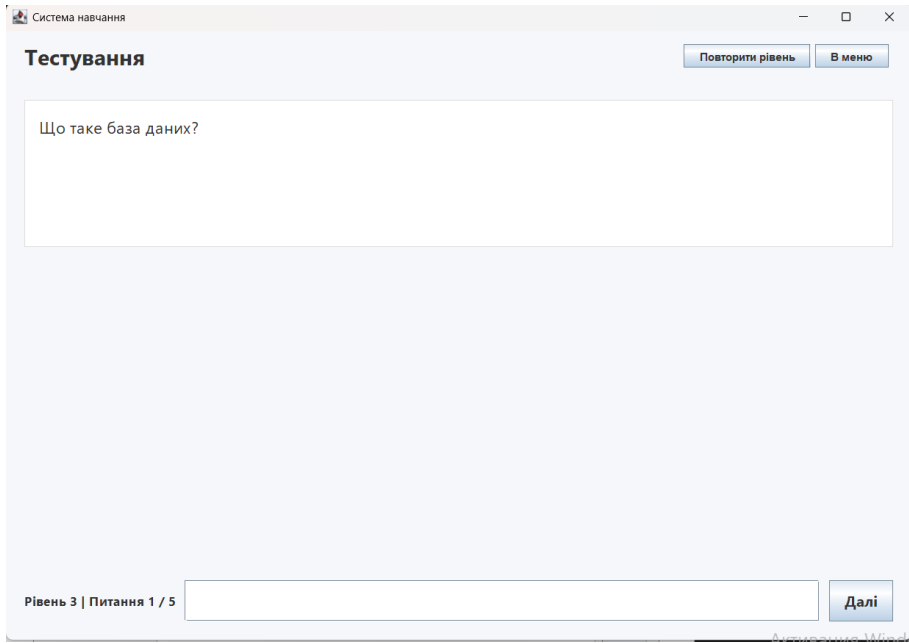
кнопки або автоматичного переходу, залежно від реалізації логіки програми. Така структура забезпечує послідовне та логічно впорядковане проходження тесту без зайвих відволікаючих елементів.

The screenshot shows a window titled "Система навчання" (Learning System). The main heading is "Тестування" (Testing). In the top right corner, there are two buttons: "Повторити рівень" (Repeat level) and "В меню" (In menu). The question is "Що являє собою модель даних?" (What is a data model?). There are four radio button options: "основою для збереження та організації інформації" (basis for storage and organization of information), "виконання послідовності дій" (execution of a sequence of actions), "програмного та апаратного забезпечення" (software and hardware), and "спосіб опису структури інформації" (way of describing the structure of information). At the bottom left, it says "Рівень 1 | Питання 3 / 5" (Level 1 | Question 3 / 5). At the bottom right, there is a "Далі" (Next) button.

2.7 Скріншот першого рівня тесту (вікторини)

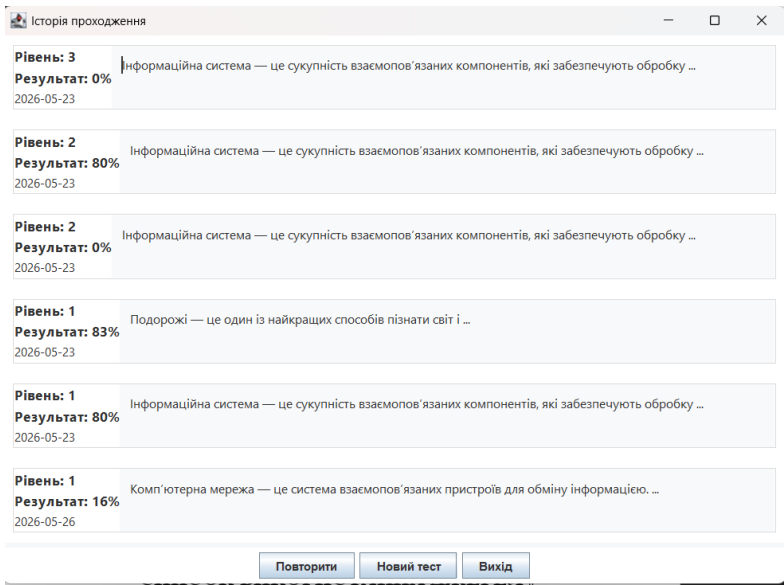
The screenshot shows a window titled "Система навчання" (Learning System). The main heading is "Тестування" (Testing). In the top right corner, there are two buttons: "Повторити рівень" (Repeat level) and "В меню" (In menu). The question is "_____ це сукупність взаємопов'язаних компонентів, які забезпечують обробку даних" (_____ is a set of interconnected components that ensure data processing). At the bottom left, it says "Рівень 2 | Питання 4 / 5" (Level 2 | Question 4 / 5). At the bottom right, there is a "Далі" (Next) button.

2.8 Скріншот другого рівня тесту (пропуски)



2.9 Скріншот третього рівня тесту (пояснення)

Окремо реалізовано екран результатів, де користувач може переглянути підсумкову інформацію про проходження тесту. Тут відображається кількість правильних і неправильних відповідей, загальний відсоток та новий рівень складності, який буде застосовано в наступному тестуванні.



2.10 Скріншот третього рівня тесту (пояснення)

Інтерфейс розроблено таким чином, щоб уникнути складних елементів керування та забезпечити інтуїтивно зрозумілу взаємодію. Основний акцент зроблено на логічну послідовність дій: запуск тесту → проходження питань → отримання результату → зміна рівня складності. Така структура дозволяє користувачу швидко зрозуміти принцип роботи системи без необхідності додаткового навчання або ознайомлення з інструкціями, що особливо важливо для навчальних програм, орієнтованих на широке коло користувачів.

Додатково інтерфейс побудований таким чином, щоб мінімізувати кількість зайвих переходів між вікнами та зменшити когнітивне навантаження на користувача. Кожен екран виконує лише одну чітко визначену функцію, що дозволяє уникнути перевантаження інформацією та підвищує загальну зручність використання системи. Завдяки цьому користувач концентрується безпосередньо на навчальному процесі, а не на навігації між елементами програми.

Також важливою особливістю є візуальна простота оформлення інтерфейсу, яка досягається за рахунок використання мінімальної кількості графічних елементів та чіткої структуризації інформації. Основні елементи керування розташовані логічно та послідовно, що забезпечує швидкий доступ до основних функцій системи. Такий підхід сприяє підвищенню ефективності взаємодії користувача з програмою та робить процес навчання більш комфортним і передбачуваним.

3 РОЗРОБКА БАЗИ ДАНИХ І ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Аналіз обраної середовища програмування

Для розробки програмної системи адаптивного навчання було обрано мову програмування Java та інтегроване середовище розробки IntelliJ IDEA. Даний вибір є обґрунтованим тим, що Java належить до числа найбільш поширених та стабільних мов програмування, які активно використовуються для створення як навчальних, так і промислових програмних продуктів. Крім того, Java забезпечує кросплатформеність[12], що дозволяє запускати розроблені застосунки на різних операційних системах без необхідності суттєвих змін у коді, що є важливою перевагою для універсального використання навчальної системи.

Однією з ключових переваг Java є її об'єктно-орієнтована парадигма програмування, яка дозволяє ефективно структурувати програму на окремі класи, об'єкти та модулі[14]. Такий підхід значно спрощує розробку складних систем, оскільки кожен компонент виконує чітко визначену роль і може бути незалежно змінений або розширений. У контексті даної роботи це особливо важливо, оскільки система адаптивного навчання складається з кількох взаємопов'язаних логічних частин, зокрема інтерфейсу користувача, модуля формування та проходження тестів, підсистеми обробки результатів, а також алгоритму адаптації рівня складності відповідно до успішності користувача.

Важливим фактором вибору Java також є наявність широкого спектра стандартних та сторонніх бібліотек, які значно спрощують процес розробки програмного забезпечення. Зокрема, у даній роботі використовується бібліотека Java Swing, що дозволяє створювати графічний інтерфейс користувача у вигляді віконного застосунку. За її допомогою можна реалізувати різноманітні елементи взаємодії, такі як кнопки, текстові поля, списки, діалогові вікна та інші компоненти. Це забезпечує зручність

використання програми та підвищує її наочність, що є важливим аспектом саме для навчальних систем, орієнтованих на кінцевого користувача.

Середовище розробки IntelliJ IDEA також відіграє важливу роль у процесі створення програмного забезпечення. Воно надає розширені можливості для написання, редагування, налагодження та тестування коду, що дозволяє значно підвищити ефективність розробки. Зокрема, IntelliJ IDEA підтримує автоматичне завершення коду, підсвічування синтаксису, швидку навігацію між класами та методами, а також вбудовані інструменти для пошуку та виправлення помилок. Це суттєво зменшує час розробки та мінімізує кількість синтаксичних і логічних помилок у програмі.

Окремо слід відзначити можливості Java щодо роботи з файловою системою та базами даних. Завдяки цьому можна реалізувати збереження результатів тестування користувачів, історії проходження завдань, а також рівнів складності, які змінюються в процесі адаптивного навчання. Це відкриває можливості для подальшого розширення функціональності системи, наприклад, додавання аналітики, персоналізованих рекомендацій або більш складних алгоритмів адаптації, що робить програму більш гнучкою та масштабованою.

3.2 Розробка бази даних

Для збереження результатів проходження тестів у програмній системі адаптивного навчання використовується реляційна база даних. Вона призначена для фіксації інформації про навчальні сесії користувача, включаючи введену теоретичну частину, дату проходження, отриманий результат та рівень складності тесту.

У межах даної роботи було обрано просту структуру бази даних, яка складається з однієї таблиці sessions. Такий підхід є достатнім для реалізації базового функціоналу системи та дозволяє зберігати всі необхідні дані без ускладнення структури.

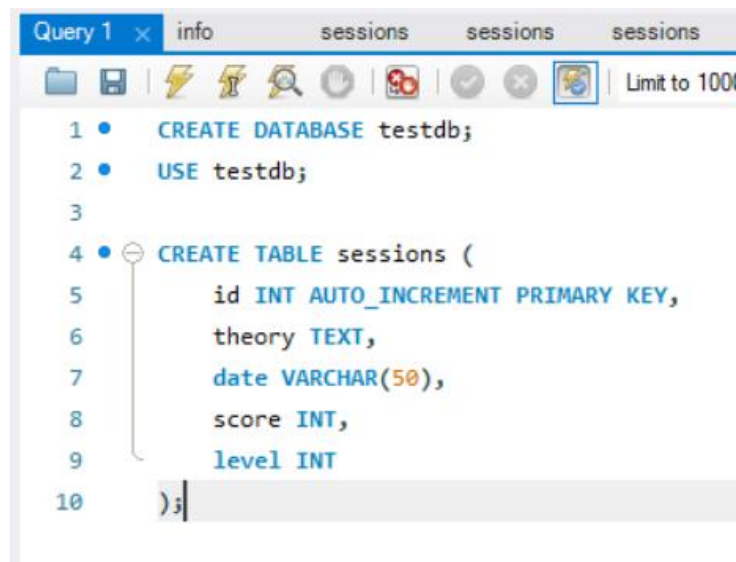


Рисунок 3.1 Код розробки бази даних

3.3 Програмна реалізація основних функцій

Структура програмної системи адаптивного навчання базується на модульному підході, що забезпечує чіткий розподіл відповідальності між окремими компонентами програми та значно спрощує її подальше розширення, супровід і модифікацію. Така архітектура дозволяє уникнути монолітності коду, підвищує його читабельність і забезпечує більш гнучку взаємодію між різними частинами системи, що є особливо важливим для навчальних програмних продуктів, які можуть поступово ускладнюватися та доповнюватися новими функціями.

Логіка роботи системи умовно розділена на кілька основних функціональних блоків, кожен з яких виконує свою чітко визначену роль у загальному процесі функціонування програми. Такий підхід відповідає принципам модульного та об'єктно-орієнтованого програмування, що дозволяє ізолювати різні рівні логіки та забезпечити їх незалежний розвиток:

- а) UI (інтерфейс користувача): Даний модуль представлений класами InputFrame, QuizFrame та HistoryFrame, які відповідають за реалізацію графічного інтерфейсу та взаємодію користувача з системою. Клас InputFrame забезпечує введення теоретичного матеріалу, який використовується як основа для формування

тестових завдань. QuizFrame відповідає за безпосередній процес проходження тестування, відображення запитань та варіантів відповідей, а також взаємодію користувача з тестовою системою. HistoryFrame використовується для перегляду історії результатів, що дозволяє користувачу відстежувати власний прогрес та аналізувати попередні спроби проходження тестів. Такий поділ інтерфейсу на окремі вікна підвищує зручність використання системи та робить навігацію більш логічною і структурованою;

- б) Logic (логіка генерації завдань): Клас QuestionGenerator реалізує ключову частину системи, яка відповідає за обробку введеного користувачем тексту та автоматичне формування навчальних завдань. На основі аналізу теоретичного матеріалу система генерує різні типи питань, зокрема тести з варіантами відповідей, завдання на заповнення пропусків, а також питання відкритого типу, що потребують пояснення. Такий підхід дозволяє зробити процес навчання більш різноманітним та ефективним, оскільки користувач взаємодіє з матеріалом у різних формах, що сприяє кращому засвоєнню знань;
- в) Control (керування процесом навчання): Клас QuizFrame виконує не лише роль інтерфейсного компонента, але й містить основну керуючу логіку процесу навчання. Він відповідає за послідовність проходження тестів, перевірку відповідей користувача, підрахунок результатів та визначення рівня успішності. Окрім цього, саме в цьому модулі реалізовано механізм переходу між рівнями складності, що є ключовою особливістю адаптивної системи. Залежно від результатів користувача система може підвищувати або знижувати складність наступних завдань, забезпечуючи індивідуальний підхід до навчання;
- г) Data (збереження даних): Модуль збереження даних

представлений класами DB та Session, які відповідають за фіксацію та зберігання результатів тестування користувача. У цих класах зберігається інформація про дату проходження тесту, отриманий бал, рівень складності, а також інші параметри сесії. Така реалізація дозволяє накопичувати статистику, що може бути використана для подальшого аналізу прогресу користувача, формування історії навчання та потенційного розширення системи аналітичними функціями. Крім того, розділення логіки збереження даних в окремий модуль забезпечує незалежність роботи з даними від інтерфейсу та бізнес-логіки, що відповідає принципам чистої архітектури програмного забезпечення.

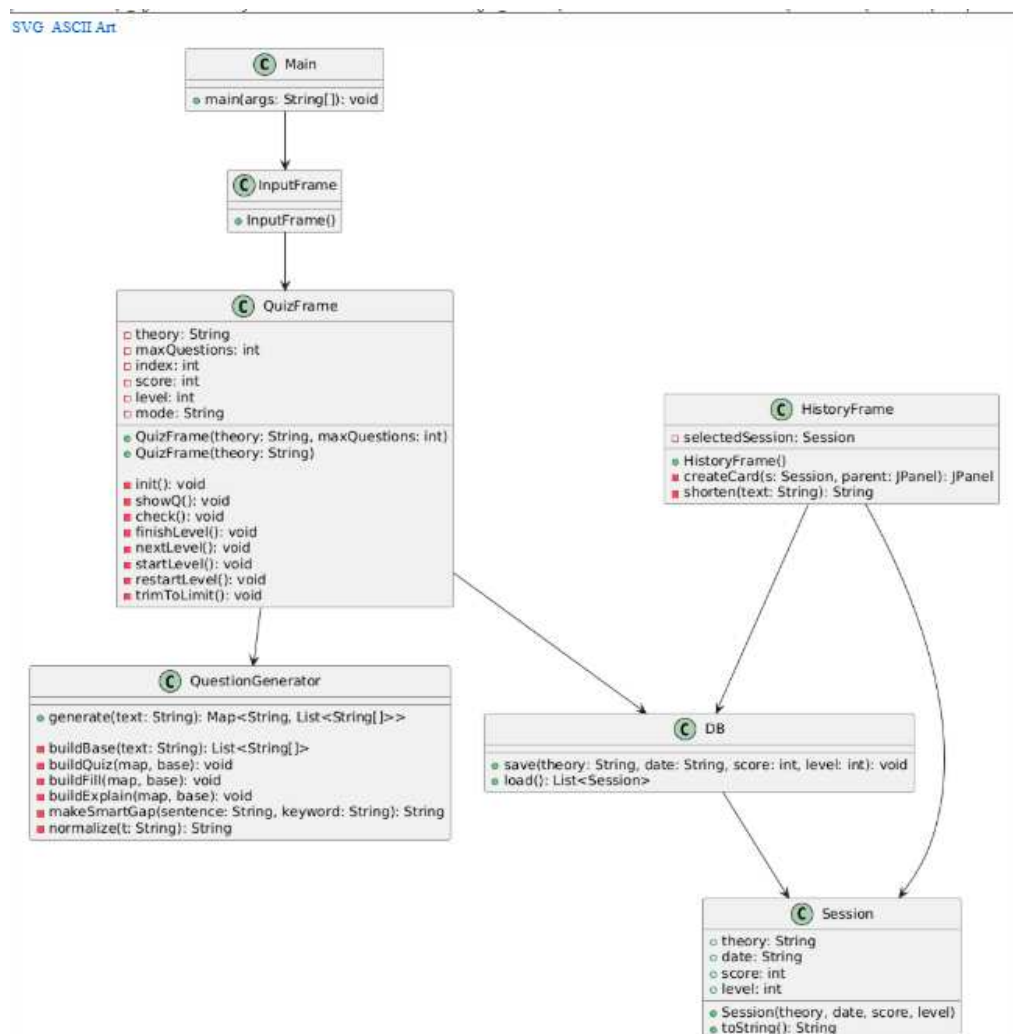


Рисунок 3.2 Діаграма класів розробки системи навчання [4]

Реалізовано наступний функціонал:

- а) введення користувачем теоретичного матеріалу для подальшої генерації тестів;
- б) автоматична генерація трьох типів завдань на основі введеного тексту;
- в) проходження тестування з автоматичною перевіркою відповідей;
- г) адаптація складності тестів залежно від результатів користувача;
- д) збереження історії проходження тестів у базі даних;
- е) перегляд попередніх результатів та можливість повторного проходження тесту.

3.4 Методика роботи користувача

Після запуску програми користувач потрапляє на початковий екран системи, де може ввести необхідний теоретичний матеріал для подальшого формування тестових завдань. Інтерфейс було розроблено таким чином, щоб забезпечити просту та зрозумілу взаємодію з програмою, мінімізувати кількість зайвих дій та зробити процес проходження тестування максимально зручним. Після введення даних користувач може розпочати тестування, а система автоматично сформує завдання відповідно до введеного матеріалу.

У процесі роботи користувач проходить декілька рівнів тестування, які можуть містити питання з варіантами відповідей, завдання на заповнення пропусків та пояснювальні питання. Система автоматично перевіряє відповіді, обчислює результат та визначає подальший рівень складності залежно від успішності проходження попереднього етапу.

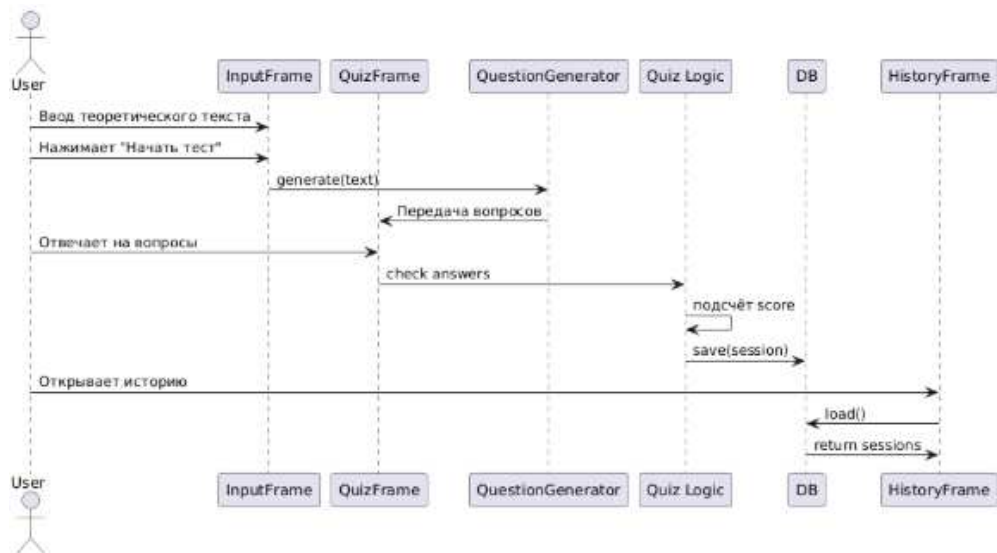


Рисунок 3.3 Діаграма взаємодії користувача

У даній частині роботи було детально розглянуто етапи створення бази даних та програмної реалізації системи адаптивного навчання. Вибір середовища розробки, структури програмних компонентів та способу збереження даних був обумовлений необхідністю забезпечення стабільної роботи системи, зручності її використання та можливості подальшого розширення функціональних можливостей.

ВИСНОВОК

У процесі виконання кваліфікаційної роботи було розроблено програмну систему адаптивного навчання на основі поведінки користувача. Основною метою роботи було створення системи, яка дозволяє автоматично змінювати складність навчальних завдань залежно від результатів проходження тестів та забезпечувати більш індивідуальний підхід до навчального процесу.

У ході виконання роботи було проведено аналіз існуючих підходів до адаптивного навчання, досліджено методи оцінювання результатів користувача та способи формування тестових завдань. На основі отриманих результатів було визначено вимоги до програмної системи та спроектовано її структуру.

Під час реалізації практичної частини було створено програмний застосунок із графічним інтерфейсом користувача, який дозволяє вводити теоретичний матеріал, автоматично формувати тестові завдання та здійснювати їх проходження. У системі реалізовано декілька типів завдань, механізм перевірки відповідей, підрахунок результатів та зміну рівня складності відповідно до успішності користувача.

Також була реалізована база даних для збереження історії проходження тестів, що дозволяє зберігати результати користувачів та використовувати їх у подальшій роботі системи. Проведене тестування показало коректну роботу основного функціоналу та підтвердило працездатність реалізованого рішення.

Отриманий результат демонструє, що поставлені завдання були виконані, а мета кваліфікаційної роботи досягнута. Розроблена система може бути використана як основа для подальшого розвитку, розширення функціональних можливостей та впровадження більш складних механізмів адаптації навчального процесу.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. ScienceDirect. Adaptive Learning Systems: principles and applications in education [Електронний ресурс]. – Режим доступу: <https://www.sciencedirect.com/topics/computer-science/adaptive-learning>
2. Oracle. The Java Tutorials – Swing GUI programming guide [Електронний ресурс]. – Режим доступу: <https://docs.oracle.com/javase/tutorial/>
3. MySQL. MySQL official documentation [Електронний ресурс]. – Режим доступу: <https://dev.mysql.com/doc/>
4. Lucidchart. UML Class Diagram: Definition, purpose and examples [Електронний ресурс]. – Режим доступу: <https://www.lucidchart.com/pages/uml-class-diagram>
5. GeeksforGeeks. Introduction to Flowcharts and Algorithms [Електронний ресурс]. – Режим доступу: <https://www.geeksforgeeks.org/an-introduction-to-flowcharts/>
6. Lucidchart. Data Flow Diagram (DFD) – complete guide and examples [Електронний ресурс]. – Режим доступу: <https://www.lucidchart.com/pages/data-flow-diagram>
7. JetBrains. IntelliJ IDEA Documentation – Overview of features and development tools [Електронний ресурс]. – Режим доступу: <https://www.jetbrains.com/help/idea/discover-intellij-idea.html>
8. Duolingo. Офіційний сайт платформи для вивчення мов [Електронний ресурс]. – Режим доступу: <https://uk.duolingo.com/>
9. Moodle. Open-source learning platform documentation and overview [Електронний ресурс]. – Режим доступу: <https://www.coursera.org/>
10. Coursera. Online courses and learning platform overview [Електронний ресурс]. – Режим доступу: <https://moodle.org/>
11. Гліб Веселовський. Види та форми дистанційного навчання в Україні [Електронний ресурс]. – Режим доступу: <https://expert.com.ua/147680-vidi-ta-formi-distancijnogo-navchannya-v-ukra%D1%97ni.html>

12. FoxmindEd. Кроссплатформний застосунок: особливості та принципи розробки [Електронний ресурс]. – Режим доступу: <https://foxminded.ua/krosplatformnyi-zastosunok/>

13. Martin Kochloefl. What is Test Data Generation [Електронний ресурс]. – Режим доступу: <https://ua.linkedin.com/pulse/what-test-data-generation-martin-kochloefl-4mnte?tl=uk>

14. FoxmindEd. Клас Java: основи та принципи програмування [Електронний ресурс]. – Режим доступу: <https://foxminded.ua/klas-java/>

ДОДАТОК А

Клас Session – модель сесії

```
package db;

public class Session {

    public String theory;
    public String date;
    public int score;
    public int level;

    public Session(String theory, String date, int score, int level) {
        this.theory = theory;
        this.date = date;
        this.score = score;
        this.level = level;
    }

    @Override    public String toString() {
        return " Уровень " + level +
            " | " + score + "%" +
            " | " + date +
            "\n " + theory;
    }
}
```

Клас DB – доступ к бд Mysql

```
package db;

import java.sql.*;
import java.util.ArrayList;
import java.util.List;

public class DB {

    private static final String URL =
        "jdbc:mysql://localhost:3306/testdb";

    private static final String USER = "root";
    private static final String PASS = "0988626338Am";

    public static void save(String theory, String date, int score, int level)
    {

        try {

            Connection c =
                DriverManager.getConnection(URL, USER, PASS);

            PreparedStatement ps =
                c.prepareStatement(
                    "INSERT INTO sessions (theory, date, score,
level) VALUES (?, ?, ?, ?)"
                );

            ps.setString(1, theory);
            ps.setString(2, date);
            ps.setInt(3, score);
            ps.setInt(4, level);

            ps.executeUpdate();
        }
    }
}
```

```

        c.close();

    } catch (Exception e) {
        e.printStackTrace();
    }
}

public static List<Session> load() {

    List<Session> list = new ArrayList<>();

    try {

        Connection c =
            DriverManager.getConnection(URL, USER, PASS);

        Statement st = c.createStatement();

        ResultSet rs = st.executeQuery(
            "SELECT * FROM sessions " +
            "ORDER BY level DESC, score DESC"
        );

        while (rs.next()) {

            list.add(new Session(
                rs.getString("theory"),
                rs.getString("date"),
                rs.getInt("score"),
                rs.getInt("level")
            ));

        }

        c.close();

    } catch (Exception e) {
        e.printStackTrace();
    }

    return list;
}

public static String getTheory(String value) {

    try {

        Connection c =
            DriverManager.getConnection(URL, USER, PASS);

        PreparedStatement ps =
            c.prepareStatement(
                "SELECT theory FROM sessions WHERE CONCAT(level, '
',score,' ',date,' ',theory) LIKE ? LIMIT 1"
            );

        ps.setString(1, "%" + value + "%");

        ResultSet rs = ps.executeQuery();

        if (rs.next()) {
            return rs.getString("theory");
        }

        c.close();
    }
}

```

```

    } catch (Exception e) {
        e.printStackTrace();
    }

    return "";
}
}

```

Клас QuestionGenerator – клас генерации тестов (основной клас)

```

package logic;

import java.util.*;

public class QuestionGenerator {

    static Random r = new Random();

    static String[][] patterns = {

        {"это", "Что такое %s?"},
        {"является", "Что такое %s?"},
        {"называется", "Как называется %s?"},
        {"представляет собой", "Что представляет собой %s?"},

        {"состоит из", "Из чего состоит %s?"},
        {"включает", "Что включает %s?"},
        {"содержит", "Что содержит %s?"},

        {"используется для", "Для чего используется %s?"},
        {"применяется для", "Для чего применяется %s?"}

    };

    static Set<String> stopWords = new HashSet<>(Arrays.asList(
"это", "и", "или", "но", "а", "в", "на", "с", "к", "у", "за", "от", "до", "из", "по",
    "который", "которая", "например", "также", "чтобы",
"является", "используется", "предназначен", "система", "процесс", "модель"
    ));

    public static Map<String, List<String[]>> generate(String text) {

        Map<String, List<String[]>> map = new HashMap<>();

        map.put("quiz", new ArrayList<>());
        map.put("fill", new ArrayList<>());
        map.put("explain", new ArrayList<>());

        List<String[]> base = buildBase(text);

        Collections.shuffle(base);

        if (base.size() > 20) {
            base = base.subList(0, 20);
        }

        buildQuiz(map, base);

        buildFill(map, base);

        buildExplain(map, base);

        return map;
    }
}

```

```

private static List<String[]> buildBase(String text) {
    List<String[]> base = new ArrayList<>();
    String[] sentences = text.split("[.!?]");
    for (String raw : sentences) {
        String original = raw.trim();
        String s = normalize(original);
        if (s.isEmpty()) continue;
        for (String[] p : patterns) {
            String key = p[0];
            String template = p[1];
            if (s.contains(key)) {
                String[] parts = s.split(key);
                if (parts.length >= 2) {
                    String left = normalize(parts[0]);
                    String right = normalize(parts[1]);
                    if (left.length() > 2 &&
                        right.length() > 2) {
                        base.add(new String[]{
                            original,
                            String.format(
                                template,
                                left
                            ),
                            right,
                            left,
                            template
                        });
                    }
                }
                break;
            }
        }
    }
    return base;
}

private static void buildQuiz(
    Map<String, List<String[]>> map,
    List<String[]> base
) {

```

```

List<String[]> quiz = map.get("quiz");

for (String[] b : base) {

    String question = b[1];

    String correct = b[2];

    List<String> options =
        new ArrayList<>();

    options.add(correct);

    List<String[]> shuffled =
        new ArrayList<>(base);

    Collections.shuffle(shuffled);

    for (String[] x : shuffled) {

        String cand = x[2];

        if (!options.contains(cand)) {

            options.add(cand);

        }

        if (options.size() == 4) {
            break;
        }

    }

    while (options.size() < 4) {

        options.add(
            "неверный вариант"
        );

    }

    Collections.shuffle(options);

    quiz.add(new String[]{

        question,

        correct,

        options.get(0),

        options.get(1),

        options.get(2),

        options.get(3)

    });

}

private static void buildFill(
    Map<String, List<String[]>> map,
    List<String[]> base
) {

    List<String[]> fill = map.get("fill");

```

```

    for (String[] b : base) {

        String sentence =
            b[0].toLowerCase();

        String concept = b[3];

        fill.add(new String[]{

            makeSmartGap(
                sentence,
                concept
            ),

            concept
        });
    }

    while (fill.size() < 6 &&
        !base.isEmpty()) {

        String[] b =
            base.get(
                r.nextInt(base.size())
            );

        fill.add(new String[]{

            makeSmartGap(
                b[0].toLowerCase(),
                b[3]
            ),

            b[3]
        });
    }

    Collections.shuffle(fill);
}

private static void buildExplain(
    Map<String, List<String[]>> map,
    List<String[]> base
) {

    List<String[]> explain =
        map.get("explain");

    for (String[] b : base) {

        String concept = b[3];

        String pattern = b[4];

        String question =
            pattern.replace(
                "%s",
                concept
            );

        explain.add(new String[]{

            question,

```

```

        b[2]
    });
}

while (explain.size() < 6 &&
       !base.isEmpty()) {

    String[] b =
        base.get(
            r.nextInt(base.size())
        );

    String question =
        b[4].replace(
            "%s",
            b[3]
        );

    explain.add(new String[]{
        question,
        b[2]
    });
}

Collections.shuffle(explain);
}

private static String makeSmartGap(
    String sentence,
    String keyword
) {

    if (keyword == null ||
        keyword.isEmpty()) {

        return sentence;
    }

    String[] markers = {

        " — это ",
        " это ",
        " является ",
        " представляет собой ",
        " состоит из "
    };

    for (String m : markers) {

        if (sentence.contains(keyword + m)) {

            return sentence.replaceFirst(
                keyword + m,
                "_____ " + m
            );
        }

        if (sentence.contains(m + keyword)) {

```



```

        return sentence.replaceFirst(
            m + keyword,
            m + " _____"
        );
    }

    return sentence.replaceFirst(
        keyword,
        " _____"
    );
}

private static String normalize(String t) {

    return t
        .toLowerCase()
        .trim();
}
}

```

Клас InputFrame – интерфейс стартовой страницы

```

package ui;

import javax.swing.*;
import java.awt.*;

public class InputFrame extends JFrame {

    public InputFrame() {

        setTitle("Ввод теории");
        setSize(850, 650);
        setDefaultCloseOperation(EXIT_ON_CLOSE);
        setLocationRelativeTo(null);
        setLayout(new BorderLayout());

        getContentPane().setBackground(new Color(245, 247, 250));

        JLabel title = new JLabel("Система генерации тестов",
            SwingConstants.CENTER);
        title.setFont(new Font("Segoe UI", Font.BOLD, 28));
        title.setBorder(BorderFactory.createEmptyBorder(20, 20, 20, 20));

        JButton rules = new JButton("Правила");

        rules.setFocusPainted(false);
        rules.setFont(new Font("Segoe UI", Font.BOLD, 14));
        rules.setBackground(new Color(70, 130, 255));
        rules.setForeground(Color.WHITE);

        rules.addActionListener(e -> JOptionPane.showMessageDialog(
            this,
            " ПРАВИЛА:\n\n" +
                "1. Введите теорию текстом.\n\n" +
                "2. Минимум 5 СТРУКТУРНЫХ предложений\n\n" +
                "3. Рекомендуется использовать:\n" +
                "- определения\n" +
                "- описания\n" +
                "- сравнения\n" +
                "- перечисления\n\n" +
                "4. Засчитываются только предложения с:\n" +
                "- это\n- является\n- называется\n- представляет\n- состоит из\n- включает\n- содержит\n- используется для\n-

```

```

применяется для\n\n" +
    "5. Обычные описания НЕ засчитываются\n\n" +
    "6. Если требования не выполнены – ввод заново\n\n" +
    "7. Всего 3 уровня:\n" +
    "- Викторина\n" +
    "- Пропуски\n" +
    "- Объяснение\n\n" +
    "8. Для перехода дальше\n" +
    "нужно минимум 80%.\n\n" +

    "9. История прохождений\n" +
    "сохраняется автоматически.",
    "Правила",
    JOptionPane.INFORMATION_MESSAGE    ));

JPanel top = new JPanel(new BorderLayout());
top.setBackground(new Color(245, 247, 250));
top.setBorder(BorderFactory.createEmptyBorder(10, 20, 10, 20));

top.add(title, BorderLayout.CENTER);

JPanel right = new JPanel(new FlowLayout(FlowLayout.RIGHT));
right.setOpaque(false);
right.add(rules);

top.add(right, BorderLayout.EAST);

JTextArea area = new JTextArea();
area.setFont(new Font("Segoe UI", Font.PLAIN, 16));
area.setLineWrap(true);
area.setWrapStyleWord(true);

area.setBorder(BorderFactory.createCompoundBorder(
    BorderFactory.createLineBorder(new Color(210, 210, 210)),
    BorderFactory.createEmptyBorder(15, 15, 15, 15)
));

JScrollPane scroll = new JScrollPane(area);

JLabel countLabel = new JLabel("Введите количество вопросов (минимум
5):");
countLabel.setFont(new Font("Segoe UI", Font.BOLD, 15));

JTextField countField = new JTextField("6");
countField.setFont(new Font("Segoe UI", Font.PLAIN, 16));
countField.setPreferredSize(new Dimension(120, 40));

JButton start = new JButton("Начать тест");
start.setFont(new Font("Segoe UI", Font.BOLD, 18));
start.setBackground(new Color(46, 204, 113));
start.setForeground(Color.WHITE);
start.setFocusPainted(false);

start.addActionListener(e -> {

    try {

        int count = Integer.parseInt(countField.getText().trim());
        String text = area.getText().trim();

        if (count < 5) {
            JOptionPane.showMessageDialog(this,
                "Минимум 5 вопросов!");
            countField.setText("");

```

```

        return;
    }

    if (text.isEmpty()) {
        JOptionPane.showMessageDialog(this,
            "Введите теорию");
        return;
    }

    String[] sentences = text.split("[.!?]");

    String[] patterns = {
        " это ",
        " является ",
        " называется ",
        " представляет собой ",
        " состоит из ",
        " включает ",
        " содержит ",
        " используется для ",
        " применяется для "
    };

    int valid = 0;

    for (String s : sentences) {

        String lower = s.toLowerCase().trim();

        if (lower.length() < 5) continue;

        boolean ok = false;

        for (String p : patterns) {
            if (lower.contains(p)) {
                ok = true;
                break;
            }
        }

        if (ok) valid++;
    }

    if (valid < 5) {
        JOptionPane.showMessageDialog(this,
            "Нужно минимум 5 корректных предложений!");
        area.setText("");
        return;
    }

    if (count > valid) {
        JOptionPane.showMessageDialog(this,
            "Количество вопросов не может быть больше\n" +
            "чем количество корректных
предложений!\n\n" +
            "Доступно: " + valid);

        countField.setText("");
        return;
    }

    new QuizFrame(text, count).setVisible(true);
    dispose();

} catch (Exception ex) {

```

```

        JOptionPane.showMessageDialog(this,
            "Введите корректное число");
    }
});

JPanel bottom = new JPanel();
bottom.setLayout(new BoxLayout(bottom, BoxLayout.Y_AXIS));
bottom.setBackground(new Color(245, 247, 250));
bottom.setBorder(BorderFactory.createEmptyBorder(10, 25, 25, 25));

countLabel.setAlignmentX(Component.LEFT_ALIGNMENT);
countField.setAlignmentX(Component.LEFT_ALIGNMENT);
start.setAlignmentX(Component.CENTER_ALIGNMENT);

bottom.add(countLabel);
bottom.add(Box.createVerticalStrut(10));
bottom.add(countField);
bottom.add(Box.createVerticalStrut(20));
bottom.add(start);

add(top, BorderLayout.NORTH);
add(scroll, BorderLayout.CENTER);
add(bottom, BorderLayout.SOUTH);
}
}

```

Клас QuizFrame – интерфейс окна с тестами

```

package ui;

import db.DB;
import logic.QuestionGenerator;

import javax.swing.*;
import java.awt.*;
import java.time.LocalDate;
import java.util.List;
import java.util.Map;

public class QuizFrame extends JFrame {

    JTextArea question = new JTextArea();
    JTextField answer = new JTextField();

    JRadioButton a = new JRadioButton();
    JRadioButton b = new JRadioButton();
    JRadioButton c = new JRadioButton();
    JRadioButton d = new JRadioButton();

    ButtonGroup group = new ButtonGroup();

    JPanel answersPanel = new JPanel();

    JButton next = new JButton("Далее");
    JButton retryBtn = new JButton("Повторить уровень");
    JButton exitBtn = new JButton("В меню");

    JLabel info = new JLabel();

    Map<String, List<String[]>> data;
    List<String[]> current;

    int index = 0;
    int score = 0;

```

```

String theory;
int maxQuestions;

boolean finished = false;

int level = 1;
String mode = "quiz";

public QuizFrame(String theory, int maxQuestions) {
    this.theory = theory;
    this.maxQuestions = maxQuestions;
    init();
}

public QuizFrame(String theory) {
    this.theory = theory;
    this.maxQuestions = 6;
    init();
}

private void init() {

    setTitle("Система обучения");
    setSize(1000, 700);
    setDefaultCloseOperation(EXIT_ON_CLOSE);
    setLocationRelativeTo(null);

    getContentPane().setBackground(new Color(245, 247, 250));
    setLayout(new BorderLayout());

    data = QuestionGenerator.generate(theory);
    trimToLimit();
    current = data.get(mode);

    JPanel top = new JPanel(new BorderLayout());
    top.setBackground(new Color(245, 247, 250));
    top.setBorder(BorderFactory.createEmptyBorder(10, 20, 10, 20));

    JLabel title = new JLabel("Тестирование");
    title.setFont(new Font("Segoe UI", Font.BOLD, 24));

    JPanel right = new JPanel(new FlowLayout(FlowLayout.RIGHT));
    right.setOpaque(false);

    retryBtn.setFocusPainted(false);
    exitBtn.setFocusPainted(false);

    right.add(retryBtn);
    right.add(exitBtn);

    top.add(title, BorderLayout.WEST);
    top.add(right, BorderLayout.EAST);

    question.setEditable(false);
    question.setLineWrap(true);
    question.setWrapStyleWord(true);

    question.setFont(new Font("Segoe UI", Font.PLAIN, 18));
    question.setBackground(Color.WHITE);

    question.setBorder(BorderFactory.createCompoundBorder(
        BorderFactory.createLineBorder(new Color(220, 220, 220)),
        BorderFactory.createEmptyBorder(15, 15, 15, 15)
    ));

```

```

    ));

    question.setMaximumSize(new Dimension(Integer.MAX_VALUE, 160));

    JPanel center = new JPanel();
    center.setLayout(new BoxLayout.Y_AXIS());
    center.setBackground(new Color(245, 247, 250));
    center.setBorder(BorderFactory.createEmptyBorder(20, 20, 20, 20));

    center.add(question);
    center.add(Box.createVerticalStrut(15));

    group.add(a);
    group.add(b);
    group.add(c);
    group.add(d);

    a.setFont(new Font("Segoe UI", Font.PLAIN, 16));
    b.setFont(new Font("Segoe UI", Font.PLAIN, 16));
    c.setFont(new Font("Segoe UI", Font.PLAIN, 16));
    d.setFont(new Font("Segoe UI", Font.PLAIN, 16));

    answersPanel.setLayout(new GridLayout(4, 1, 8, 8));
    answersPanel.setBackground(new Color(245, 247, 250));

    answersPanel.add(a);
    answersPanel.add(b);
    answersPanel.add(c);
    answersPanel.add(d);

    center.add(answersPanel);

    JPanel bottom = new JPanel(new BorderLayout(10,10));
    bottom.setBackground(new Color(245, 247, 250));
    bottom.setBorder(BorderFactory.createEmptyBorder(10,20,20,20));

    answer.setFont(new Font("Segoe UI", Font.PLAIN, 16));
    answer.setPreferredSize(new Dimension(200, 45));

    next.setFont(new Font("Segoe UI", Font.BOLD, 16));
    next.setFocusPainted(false);

    info.setFont(new Font("Segoe UI", Font.BOLD, 14));

    bottom.add(info, BorderLayout.WEST);
    bottom.add(answer, BorderLayout.CENTER);
    bottom.add(next, BorderLayout.EAST);

    retryBtn.addActionListener(e -> restartLevel());

    exitBtn.addActionListener(e -> {
        new HistoryFrame().setVisible(true);
        dispose();
    });

    next.addActionListener(e -> check());

    add(top, BorderLayout.NORTH);
    add(center, BorderLayout.CENTER);
    add(bottom, BorderLayout.SOUTH);

    showQ();
}

```

```

void trimToLimit() {
    for (String key : data.keySet()) {
        List<String[]> list = data.get(key);

        List<String[]> fixed = new java.util.ArrayList<>();

        int limit = Math.min(maxQuestions, list.size());

        for (int i = 0; i < limit; i++) {
            fixed.add(list.get(i));
        }

        data.put(key, fixed);
    }
}

void showQ() {
    if (index >= current.size()) {
        finishLevel();
        return;
    }

    String[] q = current.get(index);

    question.setText(q[0]);

    info.setText("Уровень " + level +
        " | Вопрос " + (index + 1) +
        " / " + maxQuestions);

    if (mode.equals("quiz")) {
        answer.setVisible(false);
        answersPanel.setVisible(true);

        a.setText(q[2]);
        b.setText(q[3]);
        c.setText(q[4]);
        d.setText(q[5]);

        group.clearSelection();
    } else {
        answer.setVisible(true);
        answersPanel.setVisible(false);
        answer.setText("");
    }

    revalidate();
    repaint();
}

void check() {
    if (finished) return;

    String[] q = current.get(index);

    if (mode.equals("quiz")) {

```

```

        String selected = "";

        if (a.isSelected()) selected = a.getText();
        if (b.isSelected()) selected = b.getText();
        if (c.isSelected()) selected = c.getText();
        if (d.isSelected()) selected = d.getText();

        if (selected.equalsIgnoreCase(q[1])) score++;

    } else {

        if (answer.getText().trim().equalsIgnoreCase(q[1].trim())) {
            score++;
        }
    }

    index++;
    showQ();
}

void finishLevel() {

    finished = true;

    int percent = (score * 100) / maxQuestions;

    DB.save(theory, LocalDate.now().toString(), percent, level);

    if (percent >= 80) {

        JOptionPane.showMessageDialog(this,
            "Уровень пройден: " + percent + "%");

        nextLevel();

    } else {

        int choice = JOptionPane.showOptionDialog(
            this,
            "Не пройден: " + percent + "%\nПовторить?",
            "Результат",
            JOptionPane.YES_NO_OPTION,
            JOptionPane.INFORMATION_MESSAGE,
            null,
            new String[]{"Повторить", "В меню"},
            "Повторить"
        );

        if (choice == 0) restartLevel();
        else {
            new HistoryFrame().setVisible(true);
            dispose();
        }
    }
}

void nextLevel() {

    if (mode.equals("quiz")) {
        mode = "fill";
        level = 2;
    } else if (mode.equals("fill")) {
        mode = "explain";
        level = 3;
    } else {

```



```

        new HistoryFrame().setVisible(true);
        dispose();
        return;
    }

    startLevel();
}

void startLevel() {
    index = 0;
    score = 0;
    finished = false;
    current = data.get(mode);
    showQ();
}

void restartLevel() {
    startLevel();
}
}

```

Клас HistoryFrame – интерфейс окна с историями тестов

```

package ui;

import db.DB;
import db.Session;

import javax.swing.*;
import java.awt.*;
import java.util.List;

public class HistoryFrame extends JFrame {

    private Session selectedSession = null;

    public HistoryFrame() {

        setTitle("История прохождения");
        setSize(800, 600);
        setLocationRelativeTo(null);
        setDefaultCloseOperation(EXIT_ON_CLOSE);

        JPanel mainPanel = new JPanel();
        mainPanel.setLayout(new BoxLayout(mainPanel, BoxLayout.Y_AXIS));
        mainPanel.setBackground(new Color(245, 247, 250));

        List<Session> sessions = DB.load();

        for (Session s : sessions) {
            mainPanel.add(createCard(s, mainPanel));
        }

        JScrollPane scrollPane = new JScrollPane(mainPanel);
        scrollPane.setBorder(null);

        JButton repeat = new JButton("Повторить");
        JButton newTest = new JButton("Новый тест");
        JButton exit = new JButton("Выход");

        repeat.addActionListener(e -> {

            if (selectedSession != null) {

```

```

        new QuizFrame(selectedSession.theory).setVisible(true);

        dispose();
    } else {

        JOptionPane.showMessageDialog(
            this,
            "Выбери карточку сначала"
        );
    }
});

newTest.addActionListener(e -> {

    new InputFrame().setVisible(true);

    dispose();
});

exit.addActionListener(e -> System.exit(0));

JPanel bottom = new JPanel();
bottom.setBackground(Color.WHITE);

bottom.add(repeat);
bottom.add(newTest);
bottom.add(exit);

add(scrollPane, BorderLayout.CENTER);
add(bottom, BorderLayout.SOUTH);
}

private JPanel createCard(Session s, JPanel parent) {

    JPanel card = new JPanel(new BorderLayout());
    card.setBackground(Color.WHITE);

    card.setBorder(BorderFactory.createCompoundBorder(
        BorderFactory.createEmptyBorder(10, 10, 10, 10),
        BorderFactory.createLineBorder(new Color(220, 220, 220))
    ));

    JPanel left = new JPanel();
    left.setLayout(new GridLayout(3, 1));
    left.setBackground(Color.WHITE);

    JLabel level = new JLabel("Уровень: " + s.level);
    JLabel score = new JLabel("Результат: " + s.score + "%");
    JLabel date = new JLabel("'" + s.date);

    Font bold = new Font("Segoe UI", Font.BOLD, 14);
    Font normal = new Font("Segoe UI", Font.PLAIN, 12);

    level.setFont(bold);
    score.setFont(bold);
    date.setFont(normal);

    left.add(level);
    left.add(score);
    left.add(date);

    JTextArea theory = new JTextArea(shorten(s.theory));

    theory.setLineWrap(true);

```

```

theory.setWrapStyleWord(true);
theory.setEditable(false);

theory.setFont(new Font("Segoe UI", Font.PLAIN, 13));

theory.setBackground(new Color(248, 249, 251));

theory.setBorder(BorderFactory.createEmptyBorder(10, 10, 10, 10));

card.setCursor(new Cursor(Cursor.HAND_CURSOR));

card.addMouseListener(new java.awt.event.MouseAdapter() {

    @Override      public void
mouseClicked(java.awt.event.MouseEvent e) {

        selectedSession = s;

        // reset colors          for (Component c :
parent.getComponents()) {
            c.setBackground(Color.WHITE);
        }

        card.setBackground(new Color(220, 235, 255));
    }
});

card.add(left, BorderLayout.WEST);
card.add(theory, BorderLayout.CENTER);

return card;
}

private String shorten(String text) {

    if (text == null) return "";

    String[] words = text.trim().split("\\s+");

    StringBuilder sb = new StringBuilder();

    int limit = Math.min(10, words.length);

    for (int i = 0; i < limit; i++) {
        sb.append(words[i]).append(" ");
    }

    if (words.length > 10) {
        sb.append("...");
    }

    return sb.toString().trim();
}
}

```

Клас HistoryFrame – окно запуска программы

```

import ui.InputFrame;

public class Main {
    public static void main(String[] args) {
        new InputFrame().setVisible(true);
    }
}

```


ДОДАТОК Б

Посилання на стиснуту папку в гугл діску:

https://drive.google.com/file/d/14uNiuelxKj7rghsJc6rowQbYPFCzFPYg/view?usp=drive_link