УДК 004.415.2:004.85:37.091.3

АРХІТЕКТУРА МОБІЛЬНОГО АДАПТИВНОГО ТРЕНАЖЕРА ДЛЯ САМОПЕРЕВІРКИ ЗНАНЬ ІЗ ML-МОДУЛЕМ ПІДБОРУ ЗАПИТАНЬ

Дема М.С.[1]; Крісілов В. А., д.т.н., проф.[2]
[1,2] Національний університет «Одеська політехніка»
[1,2] Україна, Одеса
[1] 7930157@stud.op.edu.ua; [2] krisilov@op.edu.ua

Анотація. *Спроєктовано архітектуру мобільного застосунку для Android – адаптивного тренажера для самоперевірки знань. Запропоновано архітектуру з клієнта, платформи Firebase та окремого ML-сервісу, який оцінює ймовірність правильної відповіді й добирає наступне запитання. Результатом тренування є рівні опанування тем, а не бали. Рішення використовує власні запитання.*
Ключові слова: *адаптивне тестування; машинне навчання; мобільний застосунок; Android; Firebase.*

Вступ. Тестування знань широко використовується не лише для контролю, а й для самоперевірки: студент самостійно перевіряє себе та повертається до незрозумілих тем. Проте класичний електронний тест зазвичай має фіксований набір і порядок запитань, однаковий для всіх, не показує, з якими темами виникають труднощі, а його результат зводиться до одного числа. Крім того, ефективність практики залежить від складності завдань: надто легкі запитання майже нічого не дають, а надто складні знижують мотивацію.

Наукові дослідження з адаптивного навчання зосереджені переважно на моделях відстеження знань (knowledge tracing), від байєсівських до глибоких нейромережевих і гібридних [1, 2]. У промислових продуктах персоналізація також реалізована через ML: наприклад, модель Birdbrain у Duolingo добирає складність вправ під рівень учня. Водночас такі рішення є закритими платформами з власним контентом, а сучасні моделі потребують великих масивів відповідей, яких не має окремий викладач. Проведений аналіз п’яти аналогів (Duolingo, Khan Academy, ALEKS, Anki, Quizizz) показав, що жодне з них не поєднує власні запитання викладача, адаптивний підбір за історією відповідей студента, виявлення тем із труднощами.

Мета роботи. Спроєктувати архітектуру мобільного застосунку для Android, що забезпечує адаптивну самоперевірку знань із виявленням тем із труднощами за допомогою моделі машинного навчання.

Основна частина. Запропоновано систему з трьох частин (рис. 1): мобільного застосунку, платформи Firebase та окремого ML-сервісу. Застосунок побудовано за шаблоном MVVM із шаром Repository: інтерфейс на Jetpack Compose відображає стан, ViewModel керує ходом тренування, а Repository приховує джерела даних. Firebase Authentication виконує реєстрацію та вхід (зокрема через Google), Cloud Firestore зберігає тести, запитання, спроби й оцінки опанування тем [3]. Використання лише цих сервісів дозволяє працювати в межах безкоштовного тарифу Spark. ML-сервіс реалізовано мовою Python (FastAPI, scikit-learn [4]) як окремий компонент із HTTP API, тож модель можна замінити без змін у застосунку.

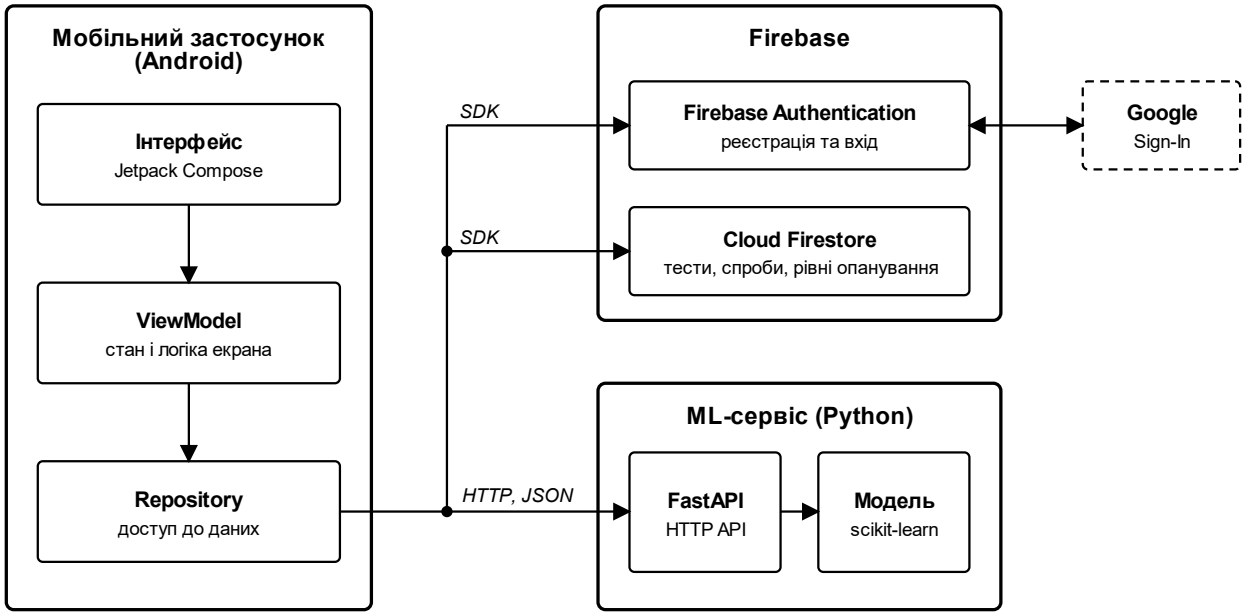


Рисунок 1 – Архітектура системи

Після кожної відповіді студента застосунок спочатку зберігає її у Firestore, щоб не втратити через збій, а потім надсилає ML-сервісу знеособлений ідентифікатор студента, історію відповідей і перелік запитань-кандидатів. Сервіс не зберігає даних між запитами; у відповіді він повертає запитання, прогнозовану ймовірність правильної відповіді, причину вибору й оновлені оцінки опанування тем. Така схема мінімізує обсяг персональних даних, що передаються за межі застосунку.

Для кожного кандидата модель оцінює ймовірність правильної відповіді за ознаками: оцінка опанування теми, початкова складність запитання, кількість спроб у темі та середній час відповіді. Як модель обрано логістичну регресію [3], бо вона інтерпретована й повертає ймовірність, що дає змогу пояснити студентові вибір запитання. Серед запитань із тем із найнижчим рівнем опанування добирається те, ймовірність якого найближча до цільової 0,65, тобто посильне, але не тривіальне (прийнято діапазон 0,5–0,8). За відсутності історії (холодний старт) добирається запитання середньої початкової складності, яку викладач задає за шкалою від 1 до 5. Якщо ML-сервіс недоступний або не відповів за встановлений час, застосунок використовує резервне правило (тема з найнижчою часткою правильних відповідей і запитання відповідної складності) та повідомляє про це студента.

За адаптивного підбору різні студенти отримують різні запитання, тому порівнювати їхні бали некоректно. Результатом тренування є рівень опанування кожної теми («слабко», «середньо», «добре») і рекомендації щодо повторення. Викладач бачить зведену аналітику: теми з найнижчою часткою правильних відповідей і запитання, фактична складність яких відрізняється від початкової.

Структуру Firestore спроєктовано так, щоб мінімізувати кількість звернень до бази даних: запитання тесту завантажуються один раз на початку спроби й зберігаються в пам’яті, на кожну відповідь виконуються два записи (відповідь і оновлення оцінок тем), а підсумки спроби для аналітики записуються один раз наприкінці. Правила безпеки обмежують студента власними даними, а викладача – власними тестами.

Для прототипу визначено цілі: наступне запитання відображається не довше ніж за 2 с (відповідь ML-сервісу – до 1 с), ROC-AUC моделі на тестовій вибірці не нижче 0,7, не менше ніж 70 % підібраних запитань мають прогнозовану ймовірність 0,5–0,8.

Висновки: Спроєктовано архітектуру мобільного адаптивного тренажера для Android, що поєднує клієнт, Firebase та незалежний ML-сервіс. Визначено інтерфейс взаємодії та алгоритм адаптивного підбору запитань з урахуванням холодного старту й резервного режиму. Спроєктовано структуру даних, що відповідає вимогам безпеки та обмеженням безкоштовних квот. Запропоновано подавати результат як рівні опанування тем, що усуває некоректне порівняння студентів за адаптивного підбору.

Використання інструментів ШІ. Claude (Anthropic) Sonnet 5.5. Використано для перекладу анотації, перевірки граматики та пунктуації.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

[1] Settles, B., & Meeder, B. (2016). A trainable spaced repetition model for language learning. In K. Erk & N. A. Smith (Eds.), *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (pp. 1848-1858). Berlin, Germany: Association for Computational Linguistics. https://doi.org/10.18653/v1/P16-1174

[2] Falmagne, J.-C., Albert, D., Doble, C., Eppstein, D., & Hu, X. (Eds.). (2013). *Knowledge spaces: Applications in education*. Berlin, Germany: Springer. https://doi.org/10.1007/978-3-642-35329-1

[3] Firebase. (n.d.). *Cloud Firestore*. Retrieved from https://firebase.google.com/docs/firestore

[4] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O. … Duchesnay, É. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12, 2825-2830. Retrieved from https://jmlr.org/papers/v12/pedregosa11a.html

Dema M.S., Krisilov V.A.

Architecture of a mobile adaptive trainer for knowledge self-assessment with an ML module for question selection

Abstract. *The paper considers the design of an Android mobile application, an adaptive trainer for knowledge self-assessment. An architecture consisting of a client, the Firebase platform and a separate ML service that estimates the probability of a correct answer and selects the next question is proposed. The result of a practice session is a mastery level for each topic rather than a score. The solution uses custom questions.*

Keywords: *adaptive testing; machine learning; mobile application; Android; Firebase.*