УДК 621.326
Козлівський В. – ст. гр. СПс-41
Тернопільський національний технічний університет імені Івана Пулюя

МОБІЛЬНА СИСТЕМА ОБЛІКУ ТА УПРАВЛІННЯ НАВЧАЛЬНИМИ ЗАНЯТТЯМИ НА БАЗІ FLUTTER

Науковий керівник: к.т.н., доцент Бачинський М. В.

Kozlivskyi V.
Ternopil Ivan Puluj National Technical University

MOBILE LESSON MANAGEMENT SYSTEM BASED ON FLUTTER

Supervisor: Ph.D., Associate Professor Bachynskyi M. V.

Ключові слова: Flutter, мобільний застосунок, управління навчальними заняттями, Django REST Framework, Firebase
Keywords: Flutter, mobile application, lesson management, Django REST Framework, Firebase

Ринок репетиторства та приватного навчання демонструє стале зростання, проте більшість доступних цифрових інструментів є або надто загальними (Google Calendar, Calendly), або надто вузькоспеціалізованими і не враховують специфіку взаємодії між викладачем та групою студентів. Серед ключових незадоволених потреб – підтримка повторюваного розкладу з механізмом виключень, рольова модель «викладач-студент», система email-запрошень до навчальних кімнат та своєчасне push-сповіщення учасників про зміни. Саме тому актуальною є розробка спеціалізованої мобільної крос-платформної системи, орієнтованої на ці задачі [1].

Предметна область управління навчальними заняттями містить ряд нетривіальних технічних викликів, які відсутні у типових планувальниках. По-перше, повторювані заняття потребують механізму виключень, що дозволяє скасовувати або переносити окремі екземпляри без зміни базового шаблону розкладу. По-друге, різниця часових зон між учасниками вимагає централізованого зберігання часу в UTC з конвертацією на клієнті. По-третє, логіка виявлення конфліктів є асиметричною, для викладача паралельне ведення двох занять неможливе, тоді як студенту достатньо попередження. По-четверте, узгодженість стану запрошень, учасників та ємності кімнати вимагає транзакційного контролю на рівні сервера.

Для вирішення зазначених задач доцільно застосовувати стек, що поєднує крос-платформну мобільну розробку з надійним серверним шаром. Flutter – фреймворк від Google, дозволяє генерувати нативний код для Android та iOS з єдиної кодової бази, забезпечуючи продуктивний UI-рендеринг через графічний рушій Impeller без залежності від платформенних компонентів. Django REST Framework надає зрілу ORM, гнучку систему дозволів і ViewSet-архітектуру для швидкого проектування REST API. Firebase забезпечує хмарну автентифікацію через ID-токени та надійну доставку push-сповіщень засобами Firebase Cloud Messaging.

Застосування Clean Architecture на клієнті дозволяє чітко розмежувати відповідальності між шарами presentation, domain та data, що суттєво спрощує тестування та подальшу підтримку коду. Патерн BLoC (Business Logic Component) забезпечує передбачуване управління станом на основі потоків подій,

унеможливлюючи неконтрольовані побічні ефекти. На сервері використання патерну Service/Selector дозволяє відокремити бізнес-логіку від читаючих запитів до бази даних, знижуючи зв'язність компонентів та спрощуючи модульне тестування кожного шару незалежно [2].

Ключовою перевагою запропонованого підходу до управління розкладом є підтримка двох типів занять – одноразових та повторюваних, зі спільним алгоритмом генерації для довільного часового діапазону. Механізм виключень через окрему сутність дозволяє гнучко коригувати окремі екземпляри повторюваного заняття, не порушуючи цілісності базового шаблону. Інтеграція підсистеми сповіщень через реактивні події сервера (Django Signals) гарантує доставку FCM push-повідомлень на всі зареєстровані пристрої учасників одразу після будь-якої зміни у розкладі, незалежно від поточного стану мобільного застосунку.

Модель даних системи організована навколо п'яти ключових сутностей. Room агрегує налаштування розкладу та місткість і виступає межею навчальної групи. Lesson зберігає або конкретну дату й час (одноразове заняття), або шаблон повторення – бітову маску днів тижня і денний час початку – разом із часовим поясом кімнати. LessonException посилається на конкретний екземпляр повторюваного заняття та перевизначає його стан (скасовано або перенесено на інший час). Invitation пов'язує кімнату з email потенційного студента та відстежує стан прийняття. UserDevice реєструє FCM-токен кожного мобільного клієнта, що забезпечує адресну доставку push-повідомлень на конкретні пристрої.

Серверний REST API задокументований засобами OpenAPI (Swagger UI), що дозволяє сторонньому клієнту або майбутній веб-версії застосунку інтегруватися без додаткових узгоджень. Кожен ендпоінт захищений Firebase ID-токеном, що валідується бібліотекою firebase-admin без звернення до стороннього сервера верифікації. Застосування крос-платформного Flutter скорочує час виходу на ринок для Android та iOS одночасно, а модульна організація Django-застосунку та чітке розмежування шарів архітектури спрощують майбутню інтеграцію з платіжними системами, платформами відеоконференцій чи аналітичними інструментами без необхідності переосмислення базової архітектури [2].

Запропонована система охоплює повний цикл взаємодії між викладачем та студентами, від реєстрації та вибору ролі до управління розкладом і отримання сповіщень. Подієво-орієнтована інфраструктура сповіщень масштабується природним чином із зростанням кількості користувачів і кімнат без змін в архітектурі. Поєднання суворого розмежування шарів, транзакційної бізнес-логіки та автоматизованого тестового покриття різних рівнів формує надійне підґрунтя для розвитку продукту, додавання нових функцій – аналітичної панелі викладача, системи оплати занять або інтеграції відео-дзвінків – можливе без ризику регресії вже реалізованої функціональності.

Список використаних джерел:

[1] Flutter Documentation. – [Електронний ресурс]. – URL: https://docs.flutter.dev (дата звернення: 06.04.2026).
[2] Martin R.C. Clean Architecture: A Craftsman's Guide to Software Structure and Design. – Boston: Pearson, 2017. – 432 с.