Автор:
Рудник Микола Олексійович
студент 4 курсу
спеціальності 121
Інженерія програмного забезпечення
Науковий керівник:
Галицький Олександр Вадимович
кандидат педагогічних наук, доцент,
доцент кафедри комп'ютерної та
програмної інженерії

РОЗРОБЛЕННЯ AI-ОРІЄНТОВАНОГО КРОСПЛАТФОРМЕНОГО КЛІЄНТСЬКОГО ЗАСТОСУНКУ УНІВЕРСИТЕТСЬКОГО РОЗКЛАДУ

Анотація. Робота присвячена розробленню AI-орієнтованого кросплатформеного клієнтського застосунку університетського розкладу в межах системи Alure. На відміну від серверної частини, яка відповідає за зберігання даних, правила, авторизацію, аудит та інтеграції, клієнтський застосунок зосереджується на щоденній взаємодії користувачів із розкладом. Він має забезпечити студентам і викладачам швидкий перегляд денного та тижневого розкладу, отримання повідомлень про зміни, роботу з персоналізацією, мовами, темою, доступністю та AI-підказками. Для адміністратора клієнтська частина також включає web-інтерфейс керування довідниками, налаштуваннями та конструктором розкладу. Основою реалізації обрано Flutter і Dart, що дозволяє створити спільну кодову базу для Web, Android і майбутніх desktop-адаптацій [1, 2]. AI у клієнтському застосунку не замінює відповідальну особу, а подає пояснення, рекомендації, нагадування та допоміжні сценарії у зрозумілій для користувача формі.

Ключові слова: Alure, університетський розклад, кросплатформений застосунок, Flutter, Dart, AI-помічник, UI/UX, Android, Web, персоналізація, доступність, push-сповіщення.

Вступ. Університетський розклад є щоденним інструментом для студентів, викладачів і деканату фвкультету. Якщо серверна частина відповідає за правильність даних і правила побудови розкладу, то клієнтський застосунок визначає, наскільки швидко й зручно користувач зможе отримати потрібні відомості. Навіть якісно побудована backend-платформа втрачає практичну цінність, якщо інтерфейс перевантажений, не адаптується до мобільного екрана, не показує актуальні зміни або не пояснює користувачеві причину оновлень.

Для сучасного освітнього середовища важливо, щоб розклад був доступний не лише як таблиця, а як персоналізований цифровий сервіс. Студент має швидко бачити найближчу пару, аудиторію, викладача, статус заняття й зміни. Викладачеві потрібні власні заняття, групи, аудиторії, службові повідомлення та швидке розуміння навантаження. Адміністратору необхідний web-інтерфейс із таблицями, фільтрами, модальними вікнами, drag-and-drop та правою панеллю деталей. Саме тому клієнтський застосунок Alure розглядається як окрема важлива частина системи, що поєднує UI/UX, адаптивність, доступність і AI-орієнтовану взаємодію.

Постановка задачі. Необхідно розробити клієнтський застосунок Alure, який працює з єдиною AI-сервісною платформою університетського розкладу та забезпечує окремі сценарії для студента, викладача, адміністратора і супер-адміністратора. Застосунок має підтримувати авторизацію, персональні екрани розкладу, перегляд карток пар, повідомлення про скасування або перенесення занять, налаштування облікового запису, локалізацію, світлу/темну тему, режим високої контрастності та збереження базових користувацьких налаштувань.

Для адміністративної частини потрібно передбачити web-панель із головною сторінкою, довідниками викладачів, груп та аудиторій, сторінками правил, інтеграцій, AI-модулів і конструктором розкладу. Для студентського та викладацького інтерфейсів

потрібно сформувати простіші мобільні та web-екрани, орієнтовані на швидке щоденне використання. Також необхідно врахувати роботу з API, realtime-оновленнями, локальним кешем, помилками мережі, fallback-поведінкою та безпечним відображенням персональних даних.

Мета роботи. Метою роботи є формування проєктного опису та клієнтської логіки AI-орієнтованого кросплатформеного застосунку Alure, який забезпечує зручну взаємодію студентів, викладачів і адміністраторів із університетським розкладом. Для досягнення мети необхідно визначити ролі користувачів, ключові екрани, навігацію, принципи UI/UX, архітектуру Flutter-застосунку, механізми персоналізації, доступності, локалізації, синхронізації з backend та сценарії використання AI-підказок.

Основна частина. Клієнтський застосунок Alure доцільно будувати на Flutter, оскільки ця технологія дозволяє створити єдину кодову базу для Web та Android, а надалі адаптувати інтерфейс під desktop-середовище [1]. Dart використовується як основна мова розробки, що дає змогу поєднати клієнтську логіку, моделі даних, валідацію форм і роботу з API в одному технологічному стеку [2]. Такий підхід зменшує дублювання коду й полегшує підтримку однакової візуальної логіки на різних платформах.

Архітектура клієнта має складатися з окремих шарів: UI-компонентів, стану застосунку, сервісів API, локального кешу, моделей даних і навігації. UI-рівень відповідає за сторінки, картки, таблиці, модальні вікна та адаптивну розмітку. Шар стану зберігає поточного користувача, роль, тему, мову, стан розкладу, активні фільтри й повідомлення. API-сервіси звертаються до backend Alure, а локальний кеш дозволяє швидко відкривати останній отриманий розклад навіть за нестабільного інтернет-з’єднання.

Для студентського сценарію головним екраном є денний розклад. Він має показувати найближчу пару, час, аудиторію, викладача, статус заняття, позначки перенесення або скасування та коротке пояснення зміни. Тижневий екран потрібний для планування, але не повинен бути перевантаженим: студенту важливо бачити структуру тижня, вільні вікна, поточний день і повідомлення про критичні зміни. AI-підказки можуть допомагати пояснювати, чому пара перенесена, радити підготовку до занять або показувати короткі нагадування з урахуванням розкладу.

Для викладача застосунок має відображати власні заняття, групи, аудиторії, зміни, службові повідомлення і навантаження. На відміну від студента, викладач частіше працює з кількома групами, тому інтерфейс повинен швидко фільтрувати заняття за днем, групою або типом пари. AI-орієнтована частина може стисло підсумовувати зміни дня, нагадувати про аудиторії, допомагати сформувати коротке повідомлення для групи або пояснювати конфлікт у розкладі.

Адміністративна web-панель має іншу логіку, бо адміністратор працює з великими наборами даних. Її стиль доцільно будувати як light blue dashboard: світло-блакитний фон, білі картки, сині акценти, округлені елементи, пошук, фільтри, таблиці та права панель деталей. Сторінки викладачів, груп і аудиторій виконують роль довідників. Конструктор розкладу є центральною функцією web-клієнта: він повинен підтримувати тижневу сітку, незаплановані пари, drag-and-drop, підсвічування слотів, модальні вікна додавання й редагування, а також зрозуміле відображення помилок і попереджень.

Особливу увагу потрібно приділити персоналізації. У застосунку мають бути мови інтерфейсу, зокрема українська, англійська та польська, перемикач теми, режим високої контрастності та збереження вибраних налаштувань. Після виходу з акаунта персоналізація не повинна некоректно переноситися на екран входу іншого користувача. Для інтерфейсу входу доцільно залишити нейтральну стандартну тему, але дозволити обрати мову і високий контраст до авторизації. Це покращує доступність і зменшує ризик плутанини між профілями.

Доступність є важливою частиною клієнтського застосунку. Інтерфейс повинен мати достатній контраст, читабельні розміри шрифтів, зрозумілу структуру фокусів, коректні підписи кнопок і передбачувану навігацію. Орієнтація на рекомендації Material

Design 3 та WCAG допомагає уникнути ситуацій, коли красивий інтерфейс стає незручним для користувачів із порушеннями зору або для тих, хто працює з малим екраном [3, 4].

AI у клієнтському застосунку має бути поданий як допоміжний шар, а не як прихований механізм, який самостійно приймає рішення. Користувач повинен бачити, де система дає факт із розкладу, а де формує рекомендацію. Для адміністратора AI може пояснювати конфлікти, пропонувати зручні слоти, коротко описувати ризики та готувати текст повідомлення. Для студента AI може підсумовувати день, радити час для підготовки, пояснювати зміни і допомагати організувати вільні вікна. Для викладача AI може формувати короткий огляд занять, нагадування і повідомлення групам.

Синхронізація з backend повинна бути передбачуваною. Сервер залишається єдиним джерелом правди, а клієнт не має самостійно вирішувати, чи можна зберегти конфліктну пару або змінити роль користувача. Клієнт лише відправляє запити, показує результат, обробляє помилки і візуально пояснює стан. Для критичних подій, наприклад скасування пари або зміни аудиторії, доцільно використовувати realtime-оновлення, а для звичайного перегляду - кешування та періодичне оновлення через API [5].

Безпека клієнтської частини полягає не лише в екрані входу. Застосунок має коректно працювати з токенами, не показувати зайві персональні дані, блокувати адміністративні екрани для користувачів без прав, очищати чутливий стан після logout і не довіряти локальним даним більше, ніж серверній відповіді. Для адміністративного доступу важливо підтримати 2FA, а для звичайних користувачів - безпечний сценарій входу через корпоративний акаунт або fallback-механізми, визначені платформою.

Інтеграції в клієнтському застосунку повинні бути непомітними для користувача, але корисними в щоденній роботі. Календарні підписки можуть експортувати розклад у зовнішні календарі, Moodle може доповнювати заняття дедлайнами та курсами, push-сповіщення повідомляють про зміни, а Telegram або email можуть використовуватися як додаткові канали [6, 7, 8]. Якщо інтеграція тимчасово недоступна, клієнт повинен пояснювати це просто і не блокувати базовий перегляд розкладу.

Висновки. Розроблення AI-орієнтованого кросплатформеного клієнтського застосунку університетського розкладу є необхідним етапом створення цілісної системи Alure. Саме клієнтська частина перетворює серверні дані, правила й AI-підказки на зрозумілий щоденний інструмент для студента, викладача та адміністратора. Для MVP доцільно реалізувати Flutter-застосунок із підтримкою Web та Android, денним і тижневим розкладом, картками пар, push-сповіщеннями, персоналізацією, локалізацією, доступністю та базовими AI-поясненнями.

Перевага такого підходу полягає в балансі між простотою для кінцевого користувача та функціональністю для адміністратора. Студент і викладач отримують швидкі персональні екрани без зайвої складності, а адміністратор - web-панель для керування розкладом і довідниками. AI виступає не автономним керівником, а пояснюваним помічником, який підсилює зручність інтерфейсу. Подальший розвиток варто спрямувати на повноцінний UI-прототип, тестування адаптивності, перевірку сценаріїв доступності, інтеграцію realtime-оновлень і формалізацію критеріїв приймання клієнтського застосунку.

Список використаних джерел

1. Flutter. Flutter Documentation [Електронний ресурс]. – Режим доступу: https://docs.flutter.dev/ (дата звернення: 15.05.2026).

2. Dart. Dart Documentation [Електронний ресурс]. – Режим доступу: https://dart.dev/ (дата звернення: 15.05.2026).

3. Material Design. Material Design 3 Guidelines [Електронний ресурс]. – Режим доступу: https://m3.material.io/ (дата звернення: 15.05.2026).

4. W3C. Web Content Accessibility Guidelines (WCAG) 2.2 [Електронний ресурс]. – Режим доступу: https://www.w3.org/TR/WCAG22/ (дата звернення: 15.05.2026).

5. Serverpod. Serverpod Documentation [Електронний ресурс]. – Режим доступу: https://docs.serverpod.dev/ (дата звернення: 15.05.2026).

6. IETF. RFC 5545: Internet Calendaring and Scheduling Core Object Specification (iCalendar) [Електронний ресурс]. – Режим доступу: https://datatracker.ietf.org/doc/html/rfc5545 (дата звернення: 15.05.2026).

7. Google. Google Calendar API v3 - API Reference [Електронний ресурс]. – Режим доступу: https://developers.google.com/workspace/calendar/api/v3/reference (дата звернення: 15.05.2026).

8. Moodle Developer Docs. External Services API [Електронний ресурс]. – Режим доступу: https://moodledev.io/docs/apis/subsystems/external (дата звернення: 15.05.2026).

9. Кархут В.Я., Галицький О.В. (2025). Інтеграція принципів вебдоступності в підготовку майбутніх розробників програмного забезпечення. *Науковий часопис Українського державного університету імені Михайла Драгоманова Серія 2 Комп'ютерно-орієнтовані системи навчання,* №24 (31). С. 109-117. https://doi.org/10.31392/UDU-nc.series2.2025.24(31).10

10. Рибачек Д. С., Галицький О. В. Віртуальні навчальні середовища як інструмент інклюзивної освіти. *Наукові записки. Серія: Педагогічні науки,* 215. 2024. С. 257-263. DOI: https://doi.org/10.36550/2415-7988-2024-1-215-257-263