УДК 004.41

В.Ю. Ткач
О.В. Романюк

РОЗРОБКА АРХІТЕКТУРИ ТА ОПИС ТЕХНОЛОГІЙ МОБІЛЬНОГО ЗАСТОСУНКУ «LEARNMATICS» ДЛЯ ГЕЙМІФІКАЦІЇ ПРОЦЕСУ ВИВЧЕННЯ МАТЕМАТИКИ

Вінницький національний технічний університет

Анотація
Розглянуто архітектуру та основні технології мобільного застосунку «LearnMatics».
Ключові слова: мобільний застосунок, технології, архітектура.

Abstract
The architecture and basic technologies of the LearnMatics mobile application are considered.
Keywords: mobile application, technology, architecture.

Вступ

В епоху стрімкого розвитку мобільних технологій важливо створювати застосунки, що поєднують високу ефективність, продуктивність та зручність використання [1]. Одним із підходів, що забезпечує ці властивості, є Single-Activity Architecture. У застосунку «LearnMatics», спрямованому на гейміфікацію процесу вивчення математики, цей підхід дозволяє оптимізувати використання ресурсів пристрою та полегшити управління навігацією. Крім того, використання таких архітектурних підходів, як Clean Architecture та шаблону MVVM (Model-View-ViewModel), сприяє створенню стабільного та легко підтримуваного коду.

Метою роботи є розробка архітектури та опис технологій мобільного застосунку «LearnMatics», що дозволить створити високоефективний і зручний у використанні інструмент для гейміфікації навчального процесу, що відповідає сучасним вимогам та очікуванням користувачів.

Основна частина

У застосунку «LearnMatics» Single-Activity Architecture відіграє важливу роль у забезпеченні ефективності та продуктивності. Замість традиційного підходу з використанням окремих активностей для кожного екрану, Single-Activity Architecture консолідує всю навігацію та управління фрагментами в одній активності. Це дозволяє уникнути перевантаження системи та забезпечити більш ефективне використання ресурсів пристрою.

Однією із основних переваг цього підходу є зменшення витрат на відтворення та зниження часу запуску застосунку. Також Single-Activity Architecture сприяє полегшенню управління станом застосунку, що робить його більш стабільним та прогностичним. Використання фрагментів у межах однієї активності спрощує управління навігацією, особливо для застосунків із складними потоками навігації. Замість того, щоб керувати кількома окремими активностями, все управління навігацією може бути оброблене в межах однієї активності, що спрощує розробку та підтримку застосунку.

Одна активність може бути більш ефективною з точки зору використання пам'яті, оскільки системі не потрібно керувати багатьма компонентами. Це особливо корисно на пристроях з обмеженими ресурсами, де кожен байт пам'яті важливий. Фрагменти у межах однієї активності можуть легко обмінюватися ресурсами та даними, що робить зручним передачу даних між різними частинами користувацького інтерфейсу. Це дозволяє створювати більш зручні та злагоджені застосунки.

Анімації переходів між фрагментами є більш плавними та безшовними, що покращує загальний досвід користувача. Плавні переходи дозволяють користувачам більш комфортно взаємодіяти із застосунком та збільшують його привабливість.

При розробці Android застосунків, особливо великих і складних проектів, важливо враховувати архітектурні принципи, щоб забезпечити гнучкість, розширюваність та підтримуваність коду. Clean Architecture є одним з таких підходів, який дозволяє створювати застосунки, що легко тестуються та зберігаються [2]. Clean Architecture – це архітектурний підхід, розроблений Робертом Мартіном (також відомий як «Дядько Боб»), який спрямований на розділення програмного забезпечення на рівні відповідно до його функціональності та залежностей. Головна ідея полягає в тому, щоб розділити застосунок на незалежні від інших шари, що дозволяє змінювати один шар без впливу на інші.

Clean Architecture включає такі компоненти:

– Domain Layer – це високорівневий шар, який містить бізнес-логіку та правила застосунку. В цьому шарі розміщуються моделі даних та інтерфейси, які визначають, що може робити застосунок;

– Data Layer - цей шар відповідає за доступ до даних. Він містить реалізації інтерфейсів, визначених у доменному рівні, та забезпечує взаємодію з джерелами даних, такими як бази даних, API або кеш;

– Presentation Layer - це шар, який відповідає за відображення інформації користувачу та обробку дій користувача. Він включає у себе UI компоненти, такі як активності, фрагменти та презентери.

Clean Architecture – це потужний підхід до розробки Android застосунків, який дозволяє створювати код, який є гнучким, тестовим та легко підтримуваним. Він дозволяє розділити застосунок на незалежні від інших компоненти, що полегшує розуміння та зміну коду в майбутньому, саме тому для побудови архітектури Android-застосунку «LearnMatics» було обрано саме Clean Architecture.

При розробці мобільного застосунку «LearnMatics» також було використано один із найпопулярніших шаблонів програмування, відомий як Model-View-ViewModel, або MVVM. Структуру взаємодії між компонентами Model, View і ViewModel на головному екрані мобільного застосунку «LearnMatics» показано на рисунку 1.

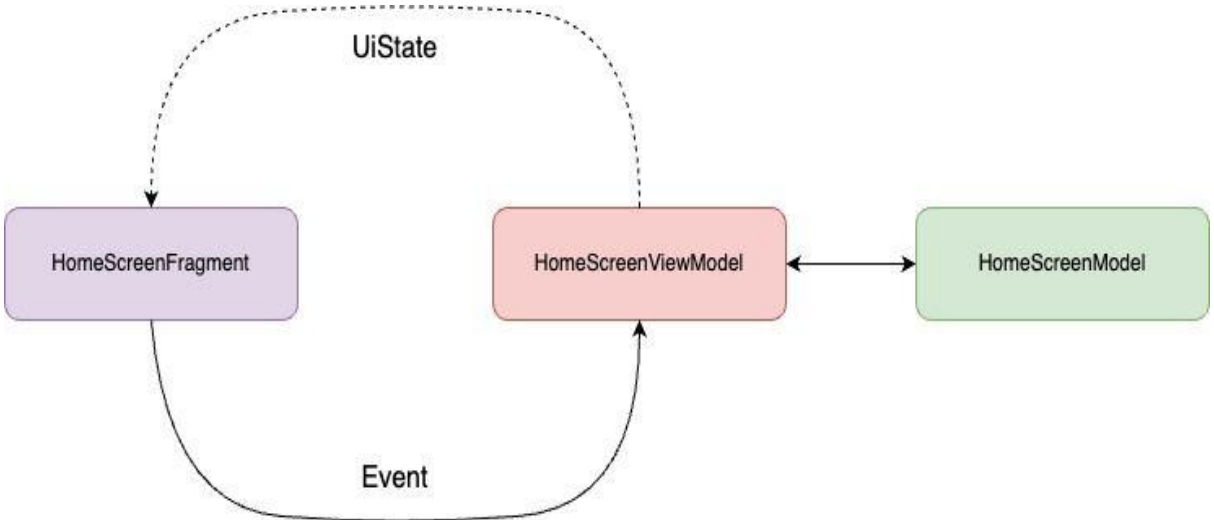

Рисунок 1 – Взаємодія Model-View-ViewModel у застосунку «LearnMatics»

MVVM включає в себе такі компоненти [3]:

– Model, яка представляє бізнес-логіку застосунку та дані, з якими працює застосунок;

– View, яке відображає інформацію користувачу та реагує на дії користувача;

– ViewModel, яка є посередником між моделлю та представленням, який управляє відображеними даними та діями користувача.

У Clean Architecture MVVM може бути використаний для реалізації рівнів представлення. ViewModel виступає в якості посередника між domain та presentation рівнями. Він отримує дані з domain рівня та готує їх для відображення у вигляді, зрозумілому для View. Він також обробляє дії

користувача та ініціює відповідні зміни в моделі. View в MVVM відповідає за відображення даних та реагування на дії користувача. У контексті Android це може бути Activity, Fragment або RecyclerView. View не має повинності знаходити або змінювати дані, вона просто відображає дані, що надходять з ViewModel.

У розробці Android застосунків, особливо тих, що використовують Clean Architecture, важливо вибрати ефективний метод для виконання асинхронних операцій, таких як мережеві запити до сервера. Корутини – це один з таких методів, який дозволяє зручно та ефективно виконувати асинхронні операції, саме тому для асинхронного виконання запитів на сервері було використано корутини.

Корутини – це легкі потоки виконання, що дозволяють виконувати асинхронний код послідовно та ефективно. Вони забезпечують зручний спосіб виконання операцій на фоні, не блокуючи основний потік виконання. Для забезпечення виконання мережевих запитів чи роботи з базою даних без блокування основного потоку виконання, можна використовувати корутини.

Висновок

Отже, впровадження Single-Activity Architecture у застосунку «LearnMatics» сприяє покращенню ефективності та продуктивності за допомогою оптимізації використання ресурсів пристрою та спрощення управління навігацією. Крім того, використання Clean Architecture та Model-View-ViewModel (MVVM) дозволяє створити код, який є гнучким, тестованим та легко підтримуваним. Ці архітектурні концепції доповнюють один одного, створюючи ефективне середовище для розробки мобільних застосунків, таких як «LearnMatics».

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Ткач В.Ю. Розробка мобільного застосунку «LEARNMATICS» для вивчення математики школярами/ В.Ю. Ткач, О. В. Романюк // Матеріали LIII Науково-технічної конференції факультету інформаційних технологій та комп'ютерної інженерії, Вінниця, 2024. URL:https://conferences.vntu.edu.ua/index.php/all-fitki/all-fitki-2024/author/submission/20558 (data of access 15.05.2024)
2. Чиста архітектура. Видання друге, Роберт К. Мартін. 2022, p. 368.
3. Model-View-ViewModel. URL: https://uk.wikipedia.org/wiki/Model-View-ViewModel (data of access 15.05.2024)

Ткач Вікторія Юріївна – студентка групи 4ПІ-20б, факультет інформаційних технологій та комп’ютерної інженерії, Вінницький національний технічний університет, м. Вінниця, e-mail: viktoriatkac11@gmail.com

Романюк Оксана Володимирівна – к.т.н., доцент кафедри програмного забезпечення, Вінницький національний технічний університет, м. Вінниця, e-mail: romaniukoksanav@gmail.com

Tkach Viktoriia – student of group 4PI-20b, Faculty for Information Technologies and Computer Engineering, Vinnytsia National Technical University, Vinnytsia, e-mail: viktoriatkac11@gmail.com

Oksana Romaniuk – Candidate of Technical Sciences, Associate Professor of the Software Chair, Vinnytsia National Technical University, Vinnytsia, e-mail: romaniukoksanav@gmail.com