1. Специфікація вимог до програмного забезпечення
2. Дослідження архітектурних рішень крос-платформного програмного забезпечення для створення сучасних мобільних застосунків
3. Версія 1.0
4. Автор Червінська Анастасія
5. ХНУРЄ
6. 30.05.2025

### 1. Вступ

1. 1.1. Огляд ПЗ

Сьогодні смартфони є у кожного, і це вже далеко не просто модний гаджет. Смартфон – це невід'ємна частина нашого життя. Він дозволяє спілкуватися з друзями та рідними в будь-якій точці світу, відправляти повідомлення і швидко знаходити потрібну інформацію.

Але це не лише про спілкування, це гарний інструмент для розвитку. Через смартфон можна дізнаватися нові речі, слухати аудіокниги та подкасти, дивитися відеоуроки та дізнаватися про різні культури. Також вони допомагають підтримувати здоровий спосіб життя, дотримуватись режиму харчування або водного балансу. Після встановлення такого виду застосунків можна отримати інструмент для фітнесу або щоденник для харчування.

Сучасні мобільні застосунки мають багато можливостей. Від бронювання квитків до онлайн-банкінгу чи онлайн знайомств. Лише кілька дотиків можуть легко вирішити проблему, яка раніше займала години. Та вирішити проблему за

декілька хвилин.

Ці технології роблять наше життя простіше, допомагають працювати та навчатись віддалено. Ці пристрої стали невід'ємною частиною нашого повсякденного життя.

У сучасному світі мобільні застосунки надають багато можливостей, роблять наше повсякденне життя більш комфортним. Важливо, що успіх мобільних застосунків пов'язаний з якістю їх програмного забезпечення, а також швидкістю їх розробки та оновлення для пристосування к постійним змінам сучасних реалій.

Сьогодні перед розробниками постає велика кількість технологій, які можливо використати для розробки мобільних застосунків. Вибираючи технологію, розробнику необхідно враховувати попередній досвід, функціональність майбутнього програмного забезпечення і терміни реалізації проекту.

Завдяки швидкому розвитку технологій і мінливим вимогам багато розробників обирають крос-платформені рішення. Для цього є багато причин: швидка реалізація для різних платформ, легка підтримка завдяки загальній кодової базі та економія часу.

Для розробника під час вибору технології для створення застосунку важливим є баланс між якістю програмного забезпечення та швидкістю розробки. Хороший мобільний застосунок повинен відповідати усім функціональним вимогам, працювати без збоїв на різних пристроях і операційних системах. Саме тому розробники вибирають такі технології, які дозволяють забезпечити надійність

та швидкість.

Останнім часом кросплатформні рішення привертають до себе все більше уваги. Адже вони не лише скорочують час розробки, але й допомагають створювати продукти, які виглядають і працюють однаково добре на будь-якому пристрої.

1. 1.2. Мета

Метою роботи є проведення дослідження сучасних крос-платформних фреймворків, які використовуються в індустрії мобільних застосунків, зробити аналіз їх переваг та недоліків, та запропонувати вдосконалення стратегії розробки.

1.3. Призначення

Цей документ описує функціональні та нефункціональні вимоги до клієнт-серверних мобільних застосунків, які використовується для експериментального дослідження продуктивності трьох сучасних фреймворків.

1.4. Сфера застосування

Програмне забезпечення призначене для розробників і дослідників, які аналізують ефективність мобільних фреймворків. Застосунок отримує дані про студентів з API, відображає їх на інтерфейсі та вимірює продуктивність (час виконання, кількість помилок, швидкість запуску).

1.5. Визначення, акроніми та скорочення

* API — Application Programming Interface;
* UI — User Interface;
* iOS — операційна система від Apple;
* macOS — операційна система для комп’ютерів Apple.

### 2. Загальний опис

2.1 Перспективи продукту

Застосунок реалізований у трьох версіях з використанням різних фреймворків (Flutter, React Native, .NET MAUI). Усі реалізації мають однаковий функціонал і використовують спільний API для отримання даних.

2.2. Функції продукту

* отримання та відображення списку студентів з відкритого API;
* вимірювання часу: завантаження даних з API, запуск застосунку, відображення інтерфейсу;
* автоматичний збір і збереження статистичних даних.

2.3. Користувацькі характеристики

Цільова аудиторія — технічні спеціалісти з базовими знаннями в галузі мобільної розробки.

2.4. Обмеження

* використовує лише вбудовані елементи фреймворків;
* для кожного фреймворку реалізовано окремий застосунок.

2.5. Припущення та залежності

* для збору статистики використовується вбудований функціонал таймерів;
* для відвалювання UI використовуються вбудовані елементи.

2.6. Вимоги до апаратного інтерфейсу

Мінімальні вимоги до апаратного інтерфейсу:

* мобільна ОС Android 10, IOS 13 та вище ;
* доступ до мережі Інтернет.

### 3. Функціональні вимоги

3.1. Ініціалізація та запуск додатка

* додаток має запускатися з головного екрану мобільного пристрою;
* при запуску додатка вимірюється час до повної готовності інтерфейсу.

3.2 Завантаження даних з API

* додаток автоматично надсилає GET-запит до відкритого API для отримання списку студентів;
* дані мають бути отримані у форматі JSON;
* час, витрачений на отримання та обробку відповіді, має бути зафіксований.

3.3 Відображення даних на екрані

* додаток повинен відобразити отриманий список студентів у вигляді прокручуваного списку;
* інтерфейс має бути адаптований до різних розмірів екранів;
* час від запиту до повного виводу даних на екран має бути виміряний.

3.4 Збір та збереження статистичних даних

* під час використання додатка автоматично збирається така інформація: час запуску додатка, час виконання запиту до API, кількість помилок під час виконання операцій.
* ці дані мають зберігатися локально на пристрої для подальшого аналізу.

3.5 Інтеграція з системою контролю версій

1. Проект додатка має підтримувати інтеграцію з системою Git для відстеження змін у коді та спрощення розробки.

### 4. Нефункціональні вимоги

4.1 Продуктивність

Застосунки в реліз версії повинен запускатися менш ніж за 2 секунди та відображати дані менш ніж за 1 секунду після завантаження.

4.2. Надійність

Кількість критичних помилок під час запуску — 0. Обробка мережевих помилок має бути реалізована.

4.3. Зручність інтерфейсу

* мінімалістичний дизайн;
* використання лише базових UI-елементів;
* інтуїтивна навігація.

4.4. Кросплатформність

Мобільний застосунок повинен працювати коректно на операційних системах Android та IOS.