Implementasi Arsitektur MVVM Dalam Pengembangan Aplikasi Study Club Duck.Sc Menggunakan Metode Rest Api Berbasis Mobile

Ridho Alfandi[1], Yuvi Darmayunata[2], Roki Hardianto[3], Mhd Arief Hasan[4]
[1,2,3,4]Program Studi Teknik Informatika Fakultas Ilmu Komputer Universitas Lancang Kuning
[1,2,3,4]Jl. Yos Sudarso KM. 8 Rumbai, Pekanbaru, Riau, telp. 0811 753 2015
e-mail: [1]ridho.alfandi47@gmail.com, [2]yuvi@unilak.ac.id, [3]roki@unilak.ac.id, [4]m.arif@unilak.ac.id

Abstrak

Duck.sc adalah komunitas belajar di Fakultas Ilmu Komputer Universitas Lancang Kuning yang fokus pada pengembangan keterampilan pemrograman. Untuk mendukung proses belajar, dikembangkan aplikasi mobile berbasis Learning Management System (LMS) menggunakan arsitektur MVVM, Flutter, dan REST API. Pengembangan mengikuti metode Extreme Programming (XP) dengan autentikasi menggunakan JSON Web Token (JWT). Sebagai evaluasi, arsitektur MVP dibandingkan dengan MVVM untuk efisiensi kode. Hasil pengujian menunjukkan aplikasi memiliki performa stabil dengan respons rata-rata 94,6 ms, serta penggunaan sumber daya dalam batas wajar. MVVM mampu mengurangi jumlah kode sebesar 20,2%. Tingkat keberhasilan penggunaan mencapai 100%, dan aplikasi memperoleh skor Net Promoter Score (NPS) sebesar 62,5.

***Kata Kunci**: Flutter, MVVM, REST API, Extreme Programming.*

Abstract

Duck.sc is a learning community at the Faculty of Computer Science, Lancang Kuning University, focused on developing programming skills. To support the learning process, a mobile application based on a Learning Management System (LMS) was developed using the MVVM architecture, Flutter, and REST API. Development followed the Extreme Programming (XP) method with authentication using JSON Web Token (JWT). As an evaluation, the MVP architecture was compared with MVVM for code efficiency. Testing results show that the application has stable performance with an average response time of 94.6 ms, and resource usage within reasonable limits. MVVM reduced the amount of code by 20.2%. The success rate of usage reached 100%, and the application achieved a Net Promoter Score (NPS) of 62.5.

***Keywords**: Flutter, MVVM, REST API, Extreme Programming.*

1. PENDAHULUAN

Perkembangan teknologi informasi dan komunikasi telah membawa perubahan besar dalam berbagai aspek kehidupan, termasuk dunia pendidikan. Salah satu perubahan signifikan adalah meningkatnya penggunaan perangkat mobile dalam mendukung proses pembelajaran yang lebih praktis dan menyenangkan dibanding metode tradisional [1]. Duck.sc merupakan study club di Fakultas Ilmu Komputer Universitas Lancang Kuning yang berfokus pada pengembangan keterampilan pemrograman. Saat ini, komunitas ini memiliki dua kelas aktif, yaitu Web *Design* dan Web *Programming*.

Untuk meningkatkan efektivitas pembelajaran, peneliti mengembangkan aplikasi mobile berbasis *Learning Management System* (LMS) yang memfasilitasi kegiatan komunitas. Aplikasi ini diharapkan dapat meningkatkan aksesibilitas materi, mempermudah pemantauan kemajuan anggota, serta membuat proses belajar lebih efisien dan terstruktur. Salah satu tantangan utama dalam pengembangan aplikasi mobile

Prosiding- SEMASTER: Seminar Nasional Teknologi Informasi & Ilmu Komputer
Vol. 4 No.1 2025 Hal 189 - 199

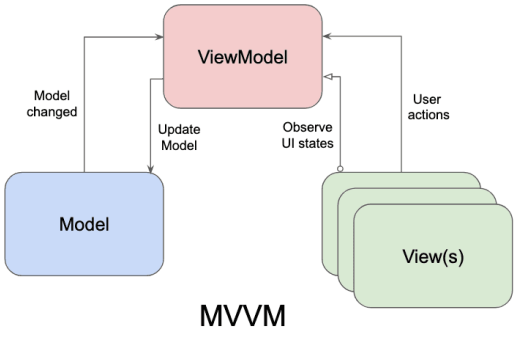
adalah pemilihan arsitektur dan metode distribusi data. Dalam penelitian ini, arsitektur *Model-View-ViewModel* (MVVM) dipilih karena kemampuannya memisahkan antarmuka pengguna dan logika bisnis [2], serta metode *Representational State Transfer* (REST) yang digunakan dalam komunikasi data antara aplikasi dan server melalui API dan *JSON Web Token* (JWT). REST API digunakan karena efisiensi dan fleksibilitas dalam penggunaannya [3]. Penggunaan arsitektur MVVM juga memudahkan pengembangan user *interface* dan logika bisnis yang dapat berjalan secara independent [4] serta dapat menghasil kode user *interface* yang lebih bersih dan lebih mudah dalam pemeliharaan [5].

Meskipun MVVM menawarkan banyak keunggulan, implementasinya memerlukan perancangan yang rapi agar sinkronisasi antara *View* dan *ViewModel* berjalan optimal. Jika tidak terstruktur, dapat terjadi pembaruan antarmuka yang berlebihan dan berdampak pada performa aplikasi [6]. Penggunaan arsitektur ini juga telah diterapkan dalam pengembangan Sistem Informasi Akademik menggunakan REST API [7] di UNIDA Gontor. Dalam pengembangan aplikasi Duck.sc, peneliti menggunakan *Flutter*, framework lintas platform yang mendukung pengembangan efisien dengan bahasa pemrograman Dart [8].

Flutter memungkinkan pengembangan cepat dengan satu basis kode untuk berbagai platform. Dengan latar belakang tersebut, penelitian ini membahas implementasi arsitektur MVVM menggunakan *Flutter* dan metode *Extreme Programming* dalam pengembangan aplikasi mobile Duck.sc. Untuk autentikasi pengguna, digunakan *JSON Web Token* (JWT), serta arsitektur Model-*View*-Presenter (MVP) digunakan sebagai pembanding dalam evaluasi performa. Penelitian ini diharapkan memberikan kontribusi dalam pengembangan aplikasi pendidikan yang efisien dan mudah digunakan.

2. METODE PENELITIAN

Penelitian ini dilakukan dalam konteks pengembangan aplikasi mobile *Learning Management System* (LMS) untuk mendukung kegiatan belajar di komunitas Duck.sc, yang berada di Fakultas Ilmu Komputer Universitas Lancang Kuning. Pengembangan aplikasi ini bertujuan untuk menciptakan media pembelajaran yang efektif, efisien, dan terstruktur melalui pemanfaatan teknologi mobile. Untuk mencapai tujuan tersebut, aplikasi ini dibangun menggunakan arsitektur Model-*View-ViewModel* (MVVM) yang memisahkan secara jelas antara antarmuka pengguna (*View*), logika bisnis (*ViewModel*), dan data (Model) [9].



Gambar 1. Arsitektur MVVM

Arsitektur MVVM dipilih karena mampu mempermudah proses pemeliharaan dan pengembangan aplikasi, serta meningkatkan modularitas dan keterbacaan kode [2]. Dalam alur kerjanya, *View* menampilkan tampilan kepada pengguna dan meneruskan

interaksi ke *ViewModel*. *ViewModel* akan memproses aksi tersebut, berinteraksi dengan Model jika diperlukan, dan memperbarui state untuk ditampilkan kembali oleh *View*.

Metode pengembangan yang digunakan dalam penelitian ini adalah *Extreme Programming* (XP), salah satu metode dalam Agile Software Development yang memungkinkan pengembangan sistem secara cepat dan adaptif terhadap perubahan [10]. Proses pengembangan dilakukan melalui empat tahapan utama, yaitu planning, *Design*, coding, dan testing. Pada tahap planning, dilakukan diskusi dengan pihak Duck.sc untuk merumuskan kebutuhan pengguna, menambahkan fitur yang sebelumnya tidak teridentifikasi, dan menyepakati revisi fitur. Pada tahap *Design*, dilakukan perancangan tampilan awal, penyusunan ulang layout berdasarkan fitur tambahan, serta penyempurnaan desain. Tahap coding dilakukan menggunakan bahasa pemrograman Dart dan framework *Flutter*, dimulai dari pengembangan prototipe awal, pengembangan versi lanjutan berdasarkan perancangan terbaru, hingga finalisasi aplikasi berdasarkan kebutuhan akhir. Terakhir, tahap testing dilakukan dengan melakukan validasi prototipe awal, pengujian fungsionalitas versi lanjutan, serta evaluasi akhir secara menyeluruh hingga aplikasi diterima pengguna.

Untuk pertukaran data antara aplikasi dan server, digunakan metode REST API. Metode ini dipilih karena kesederhanaannya, skalabilitas tinggi, serta kemampuannya untuk mendukung integrasi antar system [11]. REST API adalah metode interaksi API dengan layanan web yang didasarkan pada prinsip-prinsip REST, yang menekankan kesederhanaan dan skalabilitas [12]. Penggunaan REST API yang fleksibel juga dapat meminimalisir kerusakan saat kode mengalami perubahan besar maupun kecil [13]. Penggunaan REST API untuk komunikasi data memungkinkan pengaksesan dan manipulasi sumber daya di server jarak jauh melalui metode HTTP standar seperti GET, POST, PUT, dan DELETE [14]. REST API memfasilitasi pertukaran data melalui format JSON dan sangat mendukung metode XP karena memungkinkan penambahan fitur baru secara fleksibel. Pengujian terhadap performa REST API dilakukan dalam dua aspek, yaitu Response Time, dan Throughput. Response Time diukur untuk mengetahui rata-rata waktu server merespons permintaan dari klien. Throughput digunakan untuk mengetahui jumlah permintaan yang berhasil diproses dalam periode tertentu.

Untuk keamanan otentikasi, digunakan *JSON Web Token* (JWT) yang merupakan sebuah token yang digunakan dalam otentikasi, otorisasi dan keamanan dalam bertukar data. Token tersebut berbentuk string yang terdiri dari tiga bagian, yaitu header, payload, dan signature [15]. Penggunaan JWT dapat memberikan perlindungan terhadap ancaman dan serangan dengan melarang permintaan yang tidak sah [16].

Evaluasi performa arsitektur MVVM dilakukan dengan membandingkan jumlah baris kode (Lines of Code/LoC) antara implementasi arsitektur MVVM dan arsitektur Model-*View*-Presenter (MVP). Perbandingan ini bertujuan untuk mengetahui efisiensi struktur kode dalam pengembangan aplikasi. Selain itu, dilakukan pula pengujian terhadap penggunaan sumber daya seperti CPU dan memori untuk memastikan aplikasi berjalan dalam batas performa yang optimal. Dari sisi pengguna, dilakukan evaluasi usability dengan metode Net Promoter Score (NPS) untuk mengukur sejauh mana pengguna bersedia merekomendasikan aplikasi, serta Task Completion Rate untuk mengukur persentase tugas yang berhasil diselesaikan pengguna tanpa hambatan. Evaluasi ini memberikan gambaran menyeluruh mengenai kualitas aplikasi dari sisi teknis maupun kepuasan pengguna, sekaligus menjadi acuan untuk pengembangan lebih lanjut.

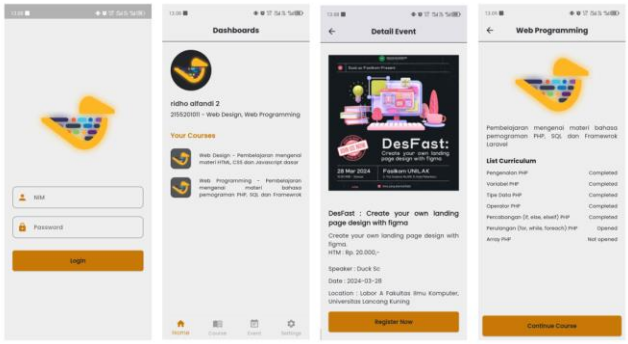
3. HASIL DAN PEMBAHASAN

Tahap ini menjelaskan hasil dari perancangan sistem, implementasi antarmuka, hingga hasil pengujian dari arsitektur MVVM.

***Prosiding- SEMASTER**: Seminar Nasional Teknologi Informasi & Ilmu Komputer*
Vol. 4 No.1 2025 Hal 189 - 199

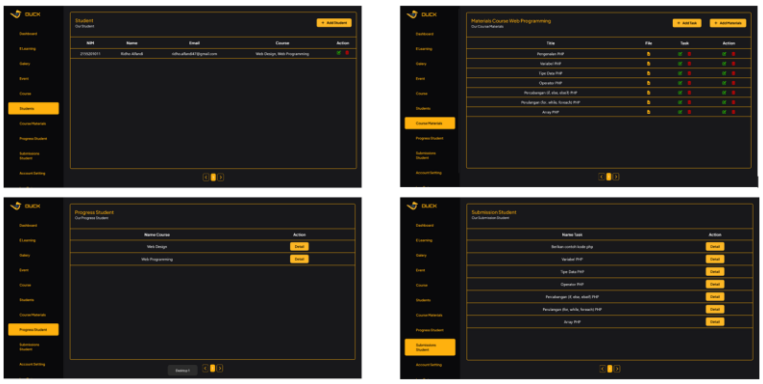
3.1 Implementasi Antarmuka

Desain yang telah dirancang kemudian diimplementasikan menjadi sebuah aplikasi Android dan Website.



Gambar 2. Implementasi Antarmuka Aplikasi

Hasil penerapan desain antarmuka ditunjukkan melalui beberapa halaman utama yang telah diimplementasikan secara fungsional, yaitu halaman login, halaman *dashboard*, halaman *event*, dan halaman LMS. Masing-masing halaman dirancang dengan memperhatikan aspek kemudahan penggunaan, konsistensi visual, serta keterpaduan fitur sesuai dengan kebutuhan pengguna.



Gambar 3. Implementasi Antarmuka Website

Hasil penerapan desain antarmuka ditunjukkan melalui beberapa halaman utama yang telah diimplementasikan secara fungsional, yaitu halaman *student*, halaman material course, halaman progress student, dan halaman submissions student. Masing-masing halaman dirancang dengan memperhatikan aspek kemudahan penggunaan, konsistensi visual, serta keterpaduan fitur sesuai dengan kebutuhan pengguna.

3.2 Implementasi Antarmuka Hasil Implementasi

Tahap ini merupakan proses penerapan dari desain sistem yang telah dirancang sebelumnya. Penerapan implementasi ini menggunakan Visual Studio Code sebagai editor, MySQL untuk basis data, dan Laravel sebagai framework backend berbasis REST API. Sementara itu, Flutter dipakai untuk membangun aplikasi mobile Duck.sc, didukung Android Studio sebagai SDK environment.

Dalam pengembangan *Flutter*, pendekatan arsitektur MVVM diterapkan. Dimulai dengan pembuatan *Model* sebagai representasi struktur data dari API. Model ini berfungsi

menghubungkan aplikasi dengan data yang dikirim dari server. Setelah itu, dibangun *Repository* untuk mengatur proses ambil-kirim data ke server, sehingga *ViewModel* tidak langsung bergantung pada API.

Langkah berikutnya adalah membuat *ViewModel* yang mengelola logika aplikasi, memanggil fungsi repository, serta menyimpan state yang akan digunakan oleh *View*. Dengan memisahkan logika dari tampilan, kode menjadi lebih bersih, mudah diuji, dan terorganisir.

Terakhir, *View* berfungsi menampilkan data ke pengguna. Dalam kasus Duck.sc, komponen UI *Flutter* memanfaatkan *Obx* dari GetX untuk merespons perubahan data secara reaktif. *View* tidak menyimpan logika bisnis, sehingga perubahan pada data cukup diatur oleh *ViewModel* tanpa perlu mengubah tampilan.

Untuk autentikasi, digunakan JWT (*JSON Web Token*) sebagai mekanisme login yang aman. Framework Laravel dipadukan dengan paket tymon/jwt-auth, yang memungkinkan pengelolaan token secara efisien. Setelah login berhasil, server mengirim token ke klien yang kemudian digunakan dalam setiap permintaan API melalui header Authorization.

JWT memastikan keamanan data tanpa menyimpan sesi di server, mengurangi beban dan meningkatkan skalabilitas. Format data JSON yang dikirim REST API juga ringan dan efisien, cocok untuk kebutuhan aplikasi mobile.

Berikut ini merupakan hasil pencatatan dan pengujian terhadap performa REST API yang dilakukan pada dua kondisi waktu berbeda, yaitu pada siang dan malam hari. Tujuan dari pencatatan ini adalah untuk mengamati konsistensi kinerja layanan API dalam menghadapi variasi kondisi jaringan yang mungkin dipengaruhi oleh waktu penggunaan. Selama proses pengamatan, perangkat pengujian terhubung ke jaringan internet 4G melalui koneksi WiFi dengan kecepatan rata-rata sekitar 20 Mbps, yang dianggap cukup representatif untuk penggunaan sehari-hari.

Pengujian dilakukan menggunakan bantuan perangkat lunak *Postman*, yang berfungsi untuk mencatat sejumlah metrik penting seperti waktu respons (*response time*), status kode (*status code*), serta ukuran data (*response size*) yang diterima dari server. Dengan membandingkan hasil pengujian antara siang dan malam hari, diharapkan dapat diperoleh gambaran mengenai sejauh mana performa REST API dipengaruhi oleh faktor eksternal, seperti tingkat kepadatan trafik jaringan atau fluktuasi waktu akses oleh pengguna lain. Hasil ini menjadi dasar dalam mengevaluasi kestabilan layanan API dalam berbagai kondisi operasional.

TABEL 1. Hasil Metode REST API Siang Hari

Endpoint URL	Response Time (ms)	Status	Size (KB)
/login	363	Success	1,43
/student/dashboard	151	Success	1,49
/student/events	107	Success	1,38
/student/update-profile	144	Success	1,09
/student/update-password	667	Success	1,08
/student/course	109	Success	1,35
/student/course/{course_id}	123	Success	1,95
/student/course/{course_id} /{material_id}	108	Success	1,86
/student/course/{course_id} /{material_id}/progress	117	Success	1,09
/student/course/{course_id} /{material_id}/submissions	94	Success	1,06

Prosiding- SEMASTER: Seminar Nasional Teknologi Informasi & Ilmu Komputer
Vol. 4 No.1 2025 Hal 189 - 199

TABEL 2. Hasil Metode REST API Malam Hari

Endpoint URL	Response Time (ms)	Status	Size (KB)
/login	1130	Success	1,43
/student/dashboard	175	Success	1,49
/student/events	172	Success	1,38
/student/update-profile	190	Success	1,09
/student/update-password	1010	Success	1,08
/student/course	479	Success	1,35
/student/course/{course_id}	474	Success	1,95
/student/course/{course_id} /{material_id}	494	Success	1,86
/student/course/{course_id} /{material_id}/progress	458	Success	1,09
/student/course/{course_id} /{material_id}/submissions	384	Success	1,06

Dari hasil pengujian pada dua waktu yang berbeda, ditemukan adanya perbedaan pada *response time* yang dicatat. Perbedaan ini kemungkinan disebabkan oleh faktor eksternal, seperti tingkat kepadatan lalu lintas jaringan (*network traffic*) atau fluktuasi waktu akses yang dilakukan oleh pengguna lain, yang dapat memengaruhi stabilitas dan kecepatan respons dari layanan REST API.

3.3 Hasil Pengujian

Tahap ini merupakan proses pengujian untuk mendeteksi kerusakan atau kesalahan yang ada saat aplikasi dijalankan. Terdapat beberapa pengujian yang meliputi pengujian metode REST API yang dibangun, pengujian peforma aplikasi serta kepuasan pengguna.

a. Pengujian *Response Time* dan *Throughput*

Pengujian ini menggunakan *endpoint API /api/student/dashboard* sebanyak 11 kali permintaan selama 1 menit.

TABEL 3. Pengujian Response Time dan Throughput

Request	Response Time (ms)	Status
1	141	Success
2	87	Success
3	83	Success
4	92	Success
5	83	Success
6	112	Success
7	85	Success
8	89	Success
9	85	Success
10	89	Success
11	376	To Many Request

Berdasarkan data tersebut, total response time selama 10 kali permintaan ialah 946 ms dan total permintaan yang sukses selama 1 menit adalah 10 permintaan. Hasil dari rata-rata response time tersebut dihitung dengan rumus:

$$Average\ Response\ Time = \frac{\sum Response\ Time}{Jumlah\ Permintaan\ API} \quad (1)$$

$$Average\ Response\ Time = \frac{946}{10} \quad (2)$$

$$Average\ Response\ Time = 94{,}6\ ms \quad (3)$$

Pengujian reponse time yang dilakukan menunjukan rata-rata waktu yang dibutuhkan API /api/student/dashboard untuk merespon permintaan ialah 94,6 ms. Pengujian Throughput menggunakan rumus:

$$Throughput = \frac{Jumlah\ Permintaan\ yang\ Berhasil\ Diproses}{Total\ Waktu\ (Detik)} \quad (4)$$

$$Throughput = \frac{10}{60} \quad (5)$$

$$Throughput = 0{,}1666\ request/detik \quad (6)$$

Pengujian throughput menunjukkan bahwa API memproses rata-rata 0,1666 request/detik (10 request/menit) selama periode pengujian.

b. Pengujian *Task Completion Rate*

Pengujian ini dilakukan dengan melakukan survei menggunakan formulir yang diisi oleh anggota dan pengurus Duck.sc. Pengujian melibatkan 32 responden dengan beberapa tugas yang harus diselesaikan, antara lain:

1) Apakah Anda berhasil login ke dalam aplikasi? (T1)
2) Apakah Anda berhasil membuka halaman course? (T2)
3) Apakah Anda berhasil melihat detail dan silabus materi course? (T3)
4) Apakah Anda berhasil melihat detail materi course? (T4)
5) Apakah Anda berhasil mendownload file dari materi? (T5)
6) Apakah Anda berhasil mengirimkan submissions? (T6)
7) Apakah Anda berhasil melihat dan membuka detail event? (T7)
8) Apakah Anda berhasil memperbarui profil? (T8)
9) Apakah Anda berhasil mengubah password? (T9)

Hasil pengujian diolah dengan rumus:

$$Task\ Completion\ Rate\ (\%) = \frac{Jumlah\ Pengguna\ yang\ Success}{Jumlah\ Responden} x\ 100\% \quad (7)$$

$$Task\ Completion\ Rate\ (\%) = \frac{32}{32} x\ 100\% \quad (8)$$

$$Task\ Completion\ Rate\ (\%) = 100\% \quad (9)$$

Pengujian Task Completion Rate menunjukan bahwa aplikasi memiliki Tingkat keberhasilan tugas sebesar 100% untuk semua scenario yang diuji.

c. Pengujian CPU dan Memori

Prosiding- SEMASTER: Seminar Nasional Teknologi Informasi & Ilmu Komputer
Vol. 4 No.1 2025 Hal 189 - 199

Pengujian ini dilakukan dengan pemantauan penggunaan CPU dan memori aplikasi dengan memanfaatkan emulator Pixel 4 yang dijalankan melalui Android Studio.

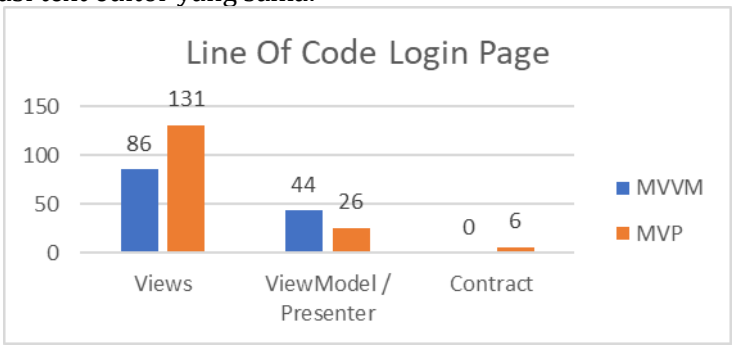
TABEL 4. Pengujian CPU dan Memori

Skenario Uji	CPU (%)	Memori (MB)	Keterangan
Login	7%	71	Menampilkan tampilan login dan melakukan proses autentikasi melalui API.
Membuka dashboard pertama kali	8,9%	71	Menampilkan data awal pada dashboard, termasuk proses pemanggilan API.
Navigasi ke halaman lain dari dashboard	0	72	Mengamati apakah terdapat pelepasan memori saat berpindah halaman.
Dibiarkan selama 1 menit tanpa interaksi	0	72	Digunakan untuk mendeteksi adanya potensi kebocoran memori (*memory leak*).

Hasil dari pengujian monitoring penggunaan CPU dan Memory yang relatif ringan dan stabil pada berbagai skenario. Pada saat proses login dan saat pertama kali membuka dashboard, penggunaan CPU terbesat tercatat masih dalam batas wajar, yaitu masing-masing sebesar 7% dan 8,9%, dengan penggunaan memori yang konsisten sebesar 71 MB.

d. Pengujian Lines of Code

Pengujian ini dilakukan dengan membuat perbandingan menggunakan arsitektur MVP. Perbandingan dilakukan pada halaman Login dalam aplikasi, dengan menggunakan Model dan konfigurasi text editor yang sama.



Gambar 4. Grafik Line of Code

Hasil data tersebut diolah dengan rumus:

$$LoC\ Reduction\ (\%) = \frac{LoC\ (MVP) - LoC\ (MVVM)}{LoC\ (MVP)}\ x\ 100\% \quad (10)$$

$$LoC\ Reduction\ (\%) = \frac{163-130}{163}\ x\ 100\% \quad (11)$$

$$LoC\ Reduction\ (\%) = 20{,}2\% \quad (12)$$

Pada pengujian Lines of Code hasil pengujian menunjukkan bahwa penggunaan arsitektur MVVM menghasilkan pengurangan jumlah baris kode sebesar 20,2% dibandingkan dengan arsitektur MVP.

e. Pengujian Net Promoter Score

Pada pengujian ini, peneliti melakukan survei menggunakan formulir yang diisi oleh anggota dan pengurus Duck.sc. Pengujian melibatkan 32 responden yang memberikan penilaian subjektif terhadap berbagai aspek pengalaman pengguna aplikasi Study Club SC dengan menggunakan skala 1–10. Adapun pertanyaan yang diajukan dalam survei tersebut meliputi:

1) Seberapa mudah Anda login ke dalam aplikasi Study Club SC? (T1)
2) Seberapa mudah Anda menemukan dan membuka course yang tersedia di aplikasi? (T2)
3) Seberapa jelas dan informatif tampilan detail course menurut Anda? (T3)
4) Seberapa mudah Anda mendownload materi atau file dari course? (T4)
5) Seberapa mudah proses pengumpulan (submit) tugas melalui aplikasi? (T5)
6) Seberapa lancar pengalaman Anda saat memperbarui profil di aplikasi? (T6)
7) Seberapa mudah Anda menemukan dan menggunakan fitur ubah password? (T7)
8) Seberapa cepat dan responsif aplikasi ini saat digunakan? (T8)
9) Seberapa nyaman antarmuka (tampilan) aplikasi ini menurut Anda? (T9)
10) Seberapa puas Anda secara keseluruhan dengan pengalaman menggunakan aplikasi Study Club SC? (T10)

TABEL 5. Pengujian NPS

R	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	Avg
R1	10	10	10	10	10	10	10	10	10	10	10
R2	10	10	7	10	10	9	10	10	7	9	9,2
R3	9	7	7	10	10	10	10	10	7	8	8,8
R4	10	10	10	10	7	10	10	10	5	8	9
R5	7	8	7	7	8	8	8	8	7	7	7,5
R6	9	8	6	9	7	8	8	7	5	7	7,4
R7	5	7	4	8	8	5	7	7	7	7	6,5
R8	9	9	9	9	9	9	9	9	9	9	9
R9	10	10	10	10	10	10	10	10	10	10	10
R10	7	8	9	9	8	9	10	9	9	10	8,8
R11	10	9	8	9	9	10	10	10	8	10	9,3
R12	10	10	10	10	10	10	10	10	6	7	9,3
R13	10	10	10	10	10	10	10	10	10	10	10
R14	10	8	8	10	9	10	10	8	8	9	9
R15	10	10	8	9	9	10	10	10	8	10	9,4
R16	7	7	5	6	8	8	4	7	5	7	6,4
R17	10	10	8	10	8	9	8	10	8	9	9
R18	10	8	10	8	10	10	10	10	10	10	9,6
R19	10	9	9	9	9	8	9	9	9	9	9
R20	10	7	5	10	5	10	5	8	8	10	7,8
R21	10	10	10	10	10	10	10	10	10	10	10
R22	6	9	9	9	9	9	10	9	10	8	8,8
R23	10	10	10	10	10	10	10	10	10	10	10
R24	8	8	7	6	10	10	9	10	7	8	8,3
R25	9	10	8	10	10	10	10	10	9	9	9,5
R26	10	10	10	10	10	10	10	10	10	10	10
R27	8	8	8	9	8	8	8	8	8	8	8,1
R28	10	10	10	10	10	10	10	10	10	10	10
R29	8	9	8	10	9	8	10	8	9	10	8,9

Prosiding- SEMASTER: Seminar Nasional Teknologi Informasi & Ilmu Komputer
Vol. 4 No.1 2025 Hal 189 - 199

R30	7	9	8	8	9	9	10	9	9	9	8,7
R31	10	10	10	10	10	10	10	10	10	10	10
R32	10	9	10	10	9	8	10	9	8	10	9,3

Hasil data tersebut diolah dengan rumus:

$$NPS = \frac{Promoters - Detractors}{Jumlah\ Responden} x\ 100 \quad (13)$$

$$NPS = \frac{20 - 0}{32} x\ 100 \quad (14)$$

$$NPS = 62{,}5 \quad (15)$$

Hasil dari pengujian Net Promoter Score menghasilkan nilai NPS sebesar 62,5 yang mengindikasikan bahwa sebagian besar pengguna memiliki persepsi positif dan loyalitas yang cukup tinggi terhadap aplikasi Study Club SC.

4. KESIMPULAN

Berdasarkan hasil penelitian, pengembangan dan pengujian yang telah dilakukan, dapat ditarik kesimpulan sebagai berikut:

1. Aplikasi Study Club SC berhasil dikembangkan dengan *Flutter* dan arsitektur MVVM, menggunakan metode *Extreme Programming* (XP) dalam dua iterasi selama kurang dari satu bulan.
2. Efisiensi kode meningkat sebesar 20,2% dibandingkan pendekatan MVP.
3. Performa sistem stabil dengan rata-rata respons API 94,6 ms, tanpa error, serta penggunaan CPU dan memori dalam batas wajar (maksimal 8,9% dan 72 MB).
4. Pengujian pengguna menunjukkan tingkat keberhasilan tugas 100% dan Net Promoter Score (NPS) sebesar 62,5, menandakan kepuasan dan loyalitas pengguna yang tinggi.
5. Aplikasi dinilai layak untuk digunakan secara luas dan memiliki peluang besar untuk dikembangkan lebih lanjut.

UCAPAN TERIMAKASIH

Terima kasih kepada DUCK SC Fakultas Ilmu Komputer Universitas Lancang Kuning selaku tempat penelitian, kepada lembaga dan pihak-pihak yang telah memberikan izin serta dukungan, kepada dosen pembimbing dan rekan-rekan yang membantu dalam diskusi, serta kepada keluarga atas doa dan motivasi yang diberikan hingga penelitian ini dapat terselesaikan dengan baik.

DAFTAR PUSTAKA

[1] N. A. Dahri, W. M. Al-Rahmi, A. S. Almogren, N. Yahaya, M. S. Vighio, and Q. Al-Maatuok, "Mobile-Based Training and Certification Framework for Teachers' Professional Development," *Sustain.*, vol. 15, no. 7, Apr. 2023, doi: 10.3390/su15075839.

[2] F. Maulana, R. Afyenni, and A. Erianda, "Aplikasi Manajemen Laboratorium Menggunakan Metode MVVM Berbasis Android," 2022. [Online]. Available: http://jurnal-itsi.org

[3] T. Hu *et al.*, "SEAPP: A secure application management framework based on REST API access control in SDN-enabled cloud environment," *J. Parallel Distrib. Comput.*, vol. 147, pp. 108–123, Jan. 2021, doi: 10.1016/j.jpdc.2020.09.006.

[4] W. Sheikh and N. Sheikh, "A Model-*View-ViewModel* (MVVM) Application Framework for Hearing Impairment Diagnosis," 2019, doi: 10.48550/arXiv.1911.08289.
[5] A. Vijaywargi and U. K. Boddapati, "Architectural Patterns in Android Development: Comparing MVP, MVVM, and MVI," *Int. J. Res. Appl. Sci. Eng. Technol.*, vol. 12, no. 4, pp. 4611–4616, Apr. 2024, doi: 10.22214/ijraset.2024.60762.
[6] J. Olusegun, "Evaluating Android Architectural Patterns: A Deep Dive into MVP, MVVM, and MVI," 2024. [Online]. Available: https://www.researchgate.net/publication/385104510
[7] M. S. Arif, A. Musthafa, and D. Muriyatmoko, "Implementation of Model-*View-ViewModel* (MVVM) Architecture Pattern in the Sistem Informasi Akademik UNIDA Gontor Mobile Application," vol. 3, pp. 283–289, 2020.
[8] H. Hussain, K. Khan, F. Farooqui, Q. Ali Arain, and I. Farah Siddiqui, "Comparative Study of Android Native and *Flutter* App Development." [Online]. Available: https://www.statista.com/statistics/1020964
[9] D. Sharma, "Android MVVM — Must-haves," *https://medium.com/@sharma.dev2506/android-mvvm-must-haves-aa1ad9b7e223*, Jan. 19, 2023.
[10] A. Trisnadoli, "Implementasi *Extreme Programming* (XP) Agile Software Development pada Pengembangan Sistem Informasi KELUARGAKU," vol. 6, no. 2, pp. 305–311, 2021, doi: 10.32493/informatika.v6i2.10088.
[11] F. Hanif, I. Ahmad, D. Darwis, I. Lazuardi Putra, and M. Fauzan Ramadhani, "Analisa Perbandingan Metode Graphql Api Dan Rest Api Dengan Menggunakan Asp.Net Core Web Api Framework," 2022.
[12] K. N. Markert *et al.*, "*Design* and implementation of a BigQuery dataset and application programmer *interface* (API) for the U.S. National Water Model," *Environ. Model. Softw.*, vol. 179, Aug. 2024, doi: 10.1016/j.envsoft.2024.106123.
[13] A. Lercher, J. Glock, C. Macho, and M. Pinzger, "Microservice API Evolution in Practice: A Study on Strategies and Challenges," *J. Syst. Softw.*, vol. 215, Sep. 2024, doi: 10.1016/j.jss.2024.112110.
[14] P. I. Sari, K. Nashirin, M. Arifudin, Y. Setiawan, and B. O. Learning, "Android Mobile Application System For Pet Care Services Using Mvvm Architecture," 2023.
[15] S. Dalimunthe, J. Reza, and A. Marzuki, "The Model For Storing Tokens In Local Storage (Cookies) Using *Json Web Token* (Jwt) With Hmac (Hash-Based Message Authentication Code) In E-Learning Systems."
[16] A. Mustapha, K. Abdellah, L. Mohamed, L. Khalid, H. Hamid, and K. Ali, "DLDiagnosis: A mobile and web application for diseases classification using Deep Learning," *SoftwareX*, vol. 23, Jul. 2023, doi: 10.1016/j.softx.2023.101488.