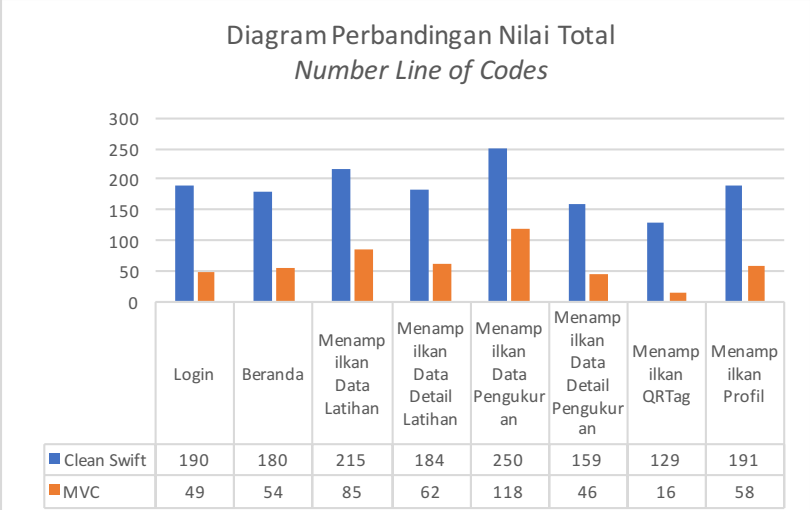
BAB 6
SIMPULAN DAN SARAN

6.1 Simpulan

Berdasarkan hasil penelitian yang dilakukan, arsitektur perangkat lunak dapat membantu *programmer* dalam menyelesaikan pengembangan perangkat lunak, akan tetapi kualitas dari sebuah perangkat lunak tetap bergantung pada kode program itu sendiri. Arsitektur hanya membantu dalam membuat fondasi sebuah perangkat lunak. Penyelesaian pengembangan perangkat lunak dapat selesai tepat waktu atau tidak akan kembali bergantung pada kemampuan dari *programmer* itu sendiri, tetapi dengan adanya arsitektur dapat mengefisiensikan waktu yang terbuang, seperti mencari kode program yang perlu diubah.

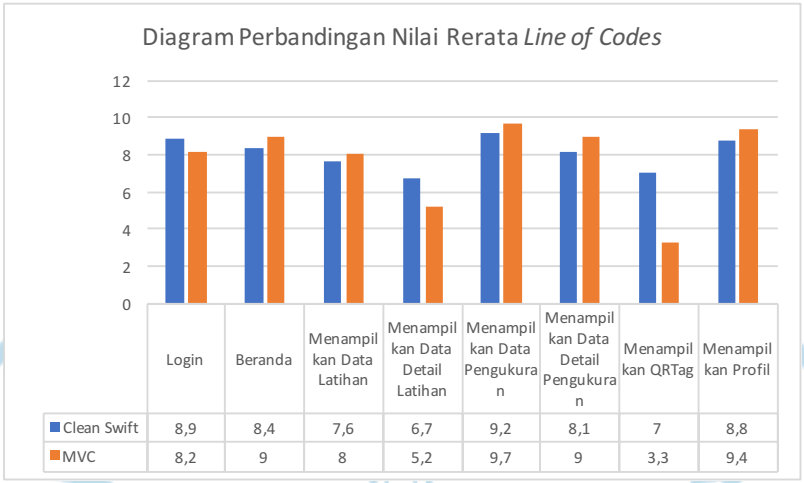
Berdasarkan hasil analisis yang dilakukan, arsitektur Clean Swift disarankan digunakan pada pembuatan perangkat lunak jangka panjang dan akan mengalami banyak perubahan kode program, sedangkan arsitektur MVC lebih disarankan digunakan pada pembuatan perangkat lunak jangka pendek dan jarang mengalami perubahan kode program.



Gambar 6.1 Diagram Perbandingan Nilai Total *Line of Codes*

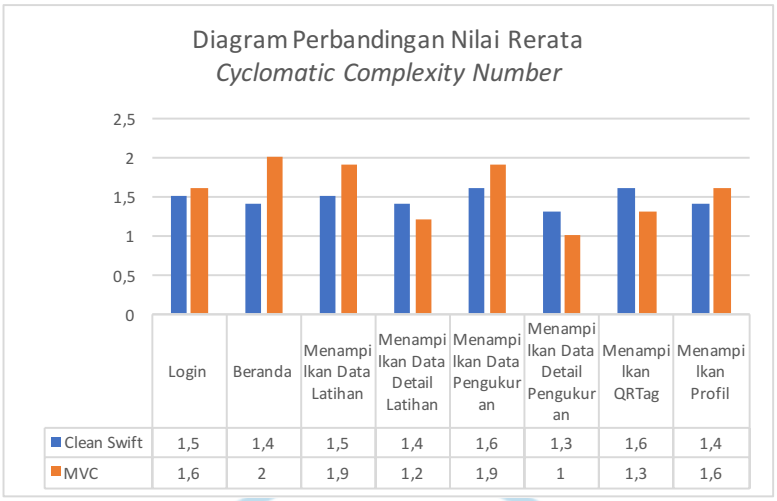
Arsitektur Clean Swift memiliki pemisahan kode program yang baik, sehingga kode program lebih mudah dimengerti dibandingkan dengan MVC. Hal ini akan membantu *programmer* dalam menemukan kode program yang perlu diubah. Pemisahan kode program yang baik ini tentunya memerlukan *effort* yang

lebih besar dibandingkan dengan MVC. *Effort* yang lebih besar ini terlihat dari perbedaan jumlah NLOC antara MVC dengan Clean Swift. Hal ini dapat terlihat pada Gambar 6.1. *Effort* ini juga akan berpengaruh pada waktu yang diperlukan dalam proses pembuatan aplikasi, sehingga untuk pembuatan perangkat lunak jangka pendek lebih disarankan menggunakan MVC karena waktu yang diperlukan dalam proses pembuatan aplikasi lebih rendah.



Gambar 6.2 Diagram Perbandingan Nilai Rerata *Line of Codes*

Gambar 6.2. adalah hasil dari pengujian nilai rerata *line of codes*. Nilai ini bergantung pada kompleks atau tidak sebuah modul. Penggunaan arsitektur MVC lebih disarankan pada pembuatan perangkat lunak yang sederhana. Hal ini dikarenakan penggunaan arsitektur Clean Swift pada modul sederhana akan menyebabkan rerata *line of codes* menjadi besar, seperti yang terjadi pada modul menampilkan *QRTag*. Hal ini terjadi karena adanya *boilercode* dari Clean Swift yang membuat *line of code* dari sebuah modul menjadi besar. Jumlah baris kode program untuk modul sederhana memiliki jumlah yang rendah, sehingga banyak kode program yang berasal dari *boilercode* membuat rerata *line of codes* menjadi besar. Untuk modul yang kompleks, penggunaan Clean Swift lebih disarankan karena penggunaan Clean Swift akan membuat nilai rerata dari *line of codes* menjadi lebih rendah. Hal ini dikarenakan kode program dibagi menjadi 7 sub-struktur yang membuat rerata *line of codes* menjadi lebih rendah.



Gambar 6.3 Diagram Perbandingan Nilai Rerata *Cyclomatic Complexity Number*

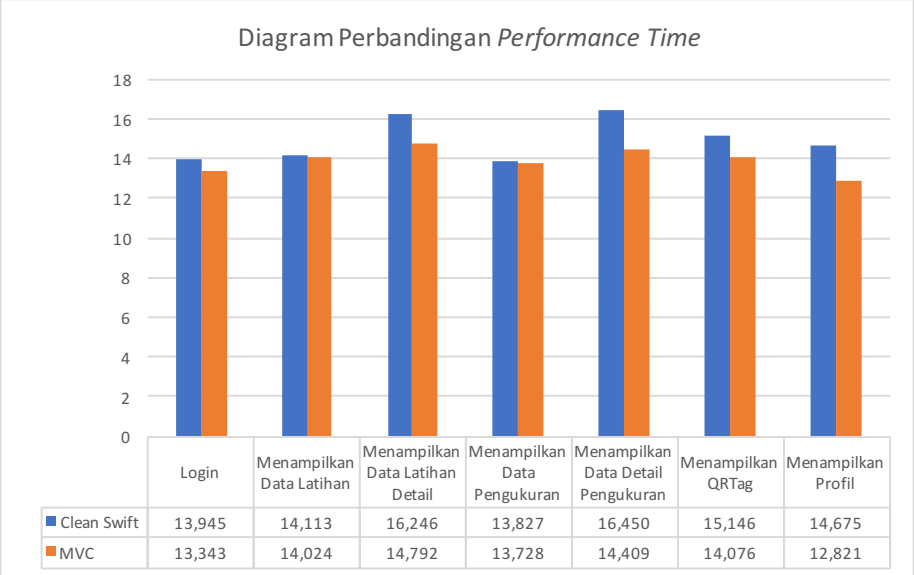
Gambar 6.3 adalah hasil dari pengujian nilai CCN. CCN ini akan mempengaruhi kompleks atau tidaknya bentuk dari sebuah kode program. Nilai ini bergantung pada sederhana atau tidak sebuah modul. Penggunaan Clean Swift pada modul yang rumit akan membuat nilai CCN menjadi lebih rendah, tetapi modul yang sederhana akan membuat nilai CCN menjadi lebih tinggi dikarenakan *boilercode* yang disediakan oleh Clean Swift. Penggunaan Clean Swift lebih disarankan untuk proses pembuatan aplikasi yang rumit karena Clean Swift akan membuat nilai CCN menjadi lebih rendah untuk modul yang rumit.

Tabel 6.1 Tabel Hasil Pengujian Nilai SCCM

Modul	**Clean Swift**		**MVC**	
	SCCM	***Threshold***	**SCCM**	***Threshold***
Login	0,294	Baik	0,400	Buruk
Beranda	0,274	Baik	0,333	Baik
Menampilkan Data Latihan	0,246	Baik	0,250	Buruk
Menampilkan Data Detail Latihan	0,333	Baik	0,500	Cukup
Menampilkan Data Pengukuran	0,241	Baik	0,250	Baik
Menampilkan Data Detail Pengukuran	0,466	Baik	0,500	Cukup
Menampilkan *QRTag*	0,600	Baik	1,000	Baik
Menampilkan Profil	0,250	Baik	0,400	Buruk

Tabel 6.1 adalah hasil pengujian nilai SCCM. Berdasarkan tabel tersebut, Clean Swift secara keseluruhan memiliki nilai SCCM yang baik, sedangkan MVC beraneka ragam. Nilai SCCM adalah nilai ketergantungan antara satu kelas dengan kelas lainnya. Arsitektur Clean Swift lebih disarankan untuk proses pembuatan aplikasi yang rumit karena memiliki nilai SCCM yang baik. Nilai SCCM yang baik

ini tidak terlepas dari *effort* yang diperlukan dalam pembuatan perangkat lunak dengan arsitektur Clean Swift. Arsitektur MVC memiliki nilai yang beragam bergantung pada ketergantungan kelas yang digunakan dalam pembuatan sebuah modul. Ada beberapa nilai SCCM yang diburuk pada arsitektur MVC, tetapi hal ini dapat di toleransi dengan waktu yang diperlukan dalam pembuatan perangkat lunak.



	Login	Menampilkan Data Latihan	Menampilkan Data Latihan Detail	Menampilkan Data Pengukuran	Menampilkan Data Detail Pengukuran	Menampilkan QRTag	Menampilkan Profil
Clean Swift	13,945	14,113	16,246	13,827	16,450	15,146	14,675
MVC	13,343	14,024	14,792	13,728	14,409	14,076	12,821

Gambar 6.4 Diagram Perbandingan *Performance Time*

Gambar 6.4 adalah hasil pengujian nilai *performance time*. Berdasarkan Gambar 6.4, MVC memiliki nilai *performance time* yang lebih baik dibandingkan dengan Clean Swift. Perbedaan *performance time* berada pada ambas batas 0,89s hingga 2,41s. Perbedaan untuk tingkat *performance* dari aplikasi cukup tinggi, yaitu mencapai waktu 2,41s. Perbedaan yang terjadi ini terjadi karena pada arsitektur Clean Swift memerlukan waktu untuk menggabungkan 7 buah berkas yang berbeda menjadi satu kesatuan. Pembuatan *object* untuk setiap berkas yang ada akan memerlukan waktu yang akibatnya mempengaruhi *performance time* dari arsitektur Clean Swift.

6.2 Saran

Pelaksanaan penelitian ini sudah membahas dua buah arsitektur, yaitu MVC dan Clean Swift, sedangkan arsitektur pada sistem operasi iOS ada lebih dari dua. Harapannya penelitian ini dapat dilanjutkan dengan menambahkan perbandingan arsitektur yang lain, seperti Model-View-Presenter. Harapan lainnya juga arsitektur

ini dapat diuji dengan pengujian *scenario-based,* sehingga nilai metrik yang lain dapat diuji.