BAB 1
PENDAHULUAN

1.1 Latar Belakang Masalah

Perkembangan teknologi informasi di masa sekarang sangatlah cepat. Banyak aplikasi yang sudah diciptakan dan dikembangkan menjadi lebih besar. Adapula aplikasi yang telah diciptakan namun tidak dapat berkembang. Aplikasi yang tidak berkembang dapat disebabkan oleh beberapa faktor, salah satunya dikarenakan kualitas dari perangkat lunak yang kurang baik. Kualitas perangkat lunak yang kurang baik dapat dipengaruhi salah satunya oleh struktur atau desain arsitektur dari perangkat lunak tersebut yang kurang sesuai[1]. Oleh karena itu, sangat penting untuk menjaga kualitas dari perangkat lunak agar dapat terus berkembang. Perangkat lunak yang baik bukan hanya dapat beroperasi dengan benar sesuai kebutuhannya, akan tetapi perangkat lunak yang baik dapat digunakan secara jangka panjang maupun memiliki kemampuan yang baik untuk dikembangkan selanjutnya. Perangkat lunak yang memiliki masa hidup panjang atau *"Long-Living Software"* sangat berkaitan dengan kualitas kode dari perangkat lunak tersebut [2].

Peningkatan kualitas kode menjadi satu upaya untuk menjaga eksistensi dari perangkat lunak. Hal ini juga yang dilakukan oleh SkinPals. SkinPals merupakan suatu perusahaan yang dalam proses bisnisnya sangat bergantung pada teknologi. Produk mereka adalah aplikasi untuk mendeteksi penyakit kulit dengan menggunakan citra foto.

Sampai saat ini aplikasi SkinPals masih terus dilakukan pengembangan. Dari hasil wawancara awal dengan pengembang terkait, proses pemeliharaan maupun pengembangan dilakukan sebanyak tiga sampai empat kali dalam setahun dengan proses *turn over* yang cukup sering. Namun kelas dan fungsi pada kode *backend* menghambat proses tersebut. *Backend* aplikasi ini bertindak sebagai *API* (*Application Programming Interface*).

Dari hasil prariset dengan melakukan observasi terhadap *backend* aplikasi SkinPals. Ditemukan beberapa permasalahan pada kode sumber yaitu penggabungan antara logika bisnis, akses data, validasi request dan penanganan

response, penamaan *variable* yang ambigu, terdapat fungsi yang melakukan banyak tugas sekaligus, dan duplikasi pada kode sumber. Selain itu terdapat 10 kelas dengan total 18 fungsi yang memiliki nilai *maintainability index* dengan kategori rendah atau sulit dipelihara. Dan pada penilaian *readability*, dari 18 fungsi hanya dua fungsi dengan klasifikasi mudah dibaca. Apabila temuan masalah tersebut dibiarkan, dapat menghambat proses pemeliharaan dan pengembangan selanjutnya. Adapun bukti dari hasil inspeksi kode sumber dapat dilihat pada **LAMPIRAN A HASIL INSPEKSI KODE SUMBER**.

Berdasarkan masalah tersebut, dilakukan studi literatur untuk meningkatkan kualitas perangkat lunak. Menurut McCall, kualitas perangkat lunak dikelompokkan menjadi tiga perspektif, yaitu *Product Revision*, *Product Transistion* dan *Product Operation*[3]. Masing – masing perspektif tersebut memiliki faktor penilaiannya tertentu. Pada penelitian ini, perspektif yang akan dinilai adalah *Product Revision*. *Product Revision* merupakan kemampuan suatu perangkat lunak dalam mengatasi perubahan setelah melewati tahap pengembangan, dimana faktor kualitas yang akan diambil adalah faktor *maintainability*. *Maintainability* merupakan faktor kualitas yang mengukur seberapa mudah perangkat lunak dapat dipelihara atau di *maintain*[4]. Selain itu, dalam penulisan kode terdapat sebuah konsep yaitu *clean code*. *Clean code* adalah suatu konsep yang menjelaskan bahwa kode yang baik atau berkualitas merupakan kode yang mudah dipahami oleh pengembang lain. Kode yang mudah dipahami akan lebih mudah untuk dilakukan perubahan atau pemeliharaan (*maintainable*)[5].

Berdasarkan masalah yang saat ini terjadi di *backend* aplikasi SkinPals, maka *refactoring* kode sumber *backend* dengan mengimplementasikan *clean code* merupakan solusi karena dapat memberikan peluang peningkatan kualitas kode sehingga dapat lebih mudah dipelihara. Selain itu, penerapan *design pattern* juga memberikan peluang lebih dalam upaya peningkatannya. Hasil dari *refactoring* diharapkan dapat meningkatkan kualitas kode dan membuat proses pemeliharaan dan pengembangan di aplikasi SkinPals berjalan baik.

1.2 Rumusan Masalah

Berdasarkan latar belakang di atas, maka masalah yang dibahas pada penelitian ini adalah apakah dengan implementasi *clean code* dan *design pattern* dapat meningkatkan *maintainability* dan *readability* pada *backend* aplikasi pendeteksi penyakit kulit.

1.3 Maksud dan Tujuan

1.3.1 Maksud

Maksud dilaksanakannya penelitian ini adalah untuk menerapkan *clean code* dan *design pattern* pada *backend* aplikasi pendeteksi penyakit kulit.

1.3.2 Tujuan

Adapun untuk tujuan penelitian adalah meningkatkan *maintainability* dan *readability* pada *backend* aplikasi pendeteksi penyakit kulit.

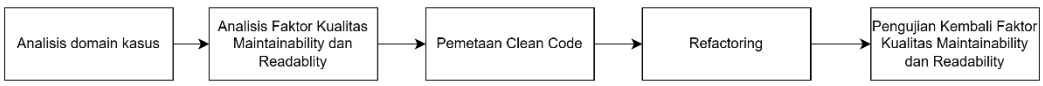
1.4 Batasan Masalah

Dalam penelitian ini, terdapat batasan masalah agar penelitian ini lebih terarah dan tidak menyimpang dari tujuan yang ingin dicapai. Adapun batasan masalah pada penelitian ini yaitu:

1. Analisis yang dilakukan yaitu dengan pendekatan *object oriented.*
2. Bahasa pemrograman yang digunakan adalah PHP.
3. Kode yang dilakukan *refactoring* hanya pada kode sumber *backend*.
4. Output dari penelitian ini adalah kode yang telah dilakukan *refactoring*.

1.5 Metodologi Penelitian

Penelitian yang dilakukan merupakan penelitian kuantitatif[6]. Metodologi yang dilakukan pada penelitian ini disesuaikan dengan kebutuhan. Berikut adalah diagram yang menggambarkan proses penelitian yang akan dilakukan:



Gambar 1.1. Metodologi Penelitian

Berikut merupakan penjelasan mengenai proses penelitian di atas:

1. **Analisis Domain Kasus**

   Pada tahap ini dilakukan analisis mengenai sistem secara umum. Sistem yang dianalisis meliputi analisis kebutuhan fungsional dengan pendekatan berorientasi objek dan analisis masalah yang ditemukan pada aplikasi.

2. **Analisis Faktor Kualitas Maintainability dan Readability**

   Pada tahap ini dilakukan analisis faktor kualitas *maintainability* dan *readability* perangkat lunak yang berkaitan dengan masalah yang ditemukan pada aplikasi.

3. **Pemetaan *Clean code***

   Pada tahap ini dilakukan pemetaan mengenai konsep *clean code* terhadap modul yang sudah diukur pada tahap sebelumnya.

4. ***Refactoring***

   Pada tahap ini dilakukan proses *refactoring* terhadap modul yang sebelumnya telah diukur dan menerapkan konsep *clean code* yang dipetakan sebelumnya.

5. **Pengujian Kembali Faktor Kualitas Maintainability dan Readability Perangkat Lunak**

   Pada tahap ini dilakukan pengujian kembali terhadap tingkat pada perangkat lunak yang sebelumnya sudah dilakukan proses *refactoring*.

1.6 Sistematika Penulisan

Sistematika dari penulisan proposal ini disusun untuk memberikan gambaran mengenai penelitian yang akan dikerjakan. Berikut merupakan sistematika penulisan dalam proposal penelitian ini:

BAB I PENDAHULUAN

Pada bab ini membahas mengenai latar belakang masalah, rumusan masalah, maksud dan tujuan, batasan masalah, metodologi penelitian dan sistematika penulisan.

BAB II LANDASAN TEORI

Pada bab ini membahas mengenai landasan teori yang digunakan, seperti konsep dasar, teori dari para ahli, teori yang berkaitan dengan teknologi dalam penelitian ini. Mengumpulkan berbagai informasi yang menjadi acuan pada penelitian ini.

BAB III ANALISIS DAN PEMETAAN *CLEAN CODE*

Pada bab ini membahas tentang analisa yang digunakan pada penelitian ini dengan hasil yang akan dikaji serta pemetaan *clean code*

BAB IV *REFACTORING* DAN PENGUJIAN KEMBALI MAINTAINABILITY

Pada bab ini membahas mengenai implementasi dari *refactoring* dan pengujian kembali maintainability.

BAB V KESIMPULAN DAN SARAN

Pada bab ini membahas mengenai kesimpulan dari hasil penelitian ini serta saran untuk pengembangan penelitian yang lebih lanjut.