BAB I
PENDAHULUAN

Bab ini memaparkan latar belakang masalah, rumusan masalah, tujuan, manfaat, ruang lingkup, serta sistematika penulisan laporan skripsi yang berjudul "Rancang Bangun Aplikasi Permohonan Surat Keterangan Mahasiswa dengan Deteksi dan Refactoring Masalah Kualitas Kode Menggunakan Static Code Analysis".

1.1 Latar Belakang

Proses administrasi persuratan di Fakultas Sains dan Matematika (FSM) Universitas Diponegoro, khususnya untuk Surat Keterangan Mahasiswa (AK.007), saat ini masih dilakukan menggunakan kertas. Berdasarkan hasil wawancara dengan Supervisor Akademik FSM UNDIP, mekanisme manual ini menimbulkan dua kendala operasional utama. Pertama, waktu pemrosesan surat sangat bergantung pada kehadiran fisik petugas di tempat, sehingga standar operasional prosedur (SOP) penyelesaian 3 hari kerja menjadi sulit untuk selalu dijamin. Kedua, sering terjadi kesalahan pengisian data atau ketidaksesuaian format kertas oleh mahasiswa yang mengakibatkan surat harus ditolak. Dalam proses manual, dokumen yang ditolak hanya diletakkan pada keranjang fisik khusus, yang menyebabkan kurangnya transparansi dan lambatnya penyampaian informasi penolakan kepada mahasiswa yang bersangkutan. Untuk mengatasi hal tersebut, sistem *E-Office* FSM perlu diperluas dengan menambahkan *workflow* khusus untuk Surat Keterangan Mahasiswa (AK.007). Pengembangan *workflow* ini melibatkan divisi Tata Usaha serta Akademik. Berdasarkan hasil studi literatur Darmansah dkk., (2022) dan Syafitri dkk., (2020), aplikasi berbasis teknologi web dengan *workflow management* yang terstruktur dapat memangkas dan mengoptimalkan alur proses bisnis, meningkatkan kemudahan penggunaan bagi pemangku kepentingan, serta menyediakan transparansi dalam setiap tahapan proses.

Selain tantangan operasional di lapangan, terdapat pula permasalahan dalam aspek keilmuan yang berkaitan dengan *software quality assurance*. Pengembangan perangkat lunak seringkali melibatkan modifikasi dan perluasan *existing codebase* yang telah berjalan dalam *production environment*. Proses penambahan fitur baru pada *existing code* berpotensi

menimbulkan penurunan kualitas kode jika tidak dikelola dengan baik (Li dkk., 2022; Verdecchia dkk., 2021). Penurunan kualitas ini seringkali baru teridentifikasi ketika sudah berdampak signifikan pada produktivitas tim pengembang atau ketika terjadi kegagalan kritis pada lingkungan produksi (Besker dkk., 2018).

Penelitian ini mengusulkan pendekatan yang memanfaatkan static *code analysis* sebagai mekanisme untuk mengungkap *code quality problems* pada *codebase*, khususnya *code smells* dalam kategori *Large Class*, *Long Method*, dan *Duplicated Code*. Pengembangan fitur *workflow* Surat Keterangan Mahasiswa (AK.007) pada sistem *E-Office* FSM memberikan kesempatan untuk mengobservasi bagaimana proses analisis kualitas kode dapat mengungkap dan memperbaiki kualitas implementasi kode melalui siklus deteksi-*refactoring*-validasi. Dengan pendekatan ini, penelitian tidak hanya menghasilkan fitur *workflow* yang berfungsi dengan baik, tetapi juga menghasilkan dokumentasi sistematis tentang *code smells* yang ditemukan menggunakan SonarQube, proses *refactoring* yang dilakukan untuk meningkatkan metrik kualitas kode seperti *Cyclomatic Complexity*, *Lines of Code*s, dan mengurangi duplikasi kode, serta validasi *improvement* melalui pengukuran ulang metrik kode dan perbandingan *runtime performance* (Parsa dkk., 2025).

Dalam penelitian ini, analisis kualitas kode difokuskan pada karakteristik *maintainability* (keterpeliharaan) sesuai standar ISO/IEC 25010 (ISO/IEC 25010, 2011). Pemilihan karakteristik ini didasarkan pada temuan Mäntylä & Lassenius, (2006) yang menyatakan bahwa *code smells* merupakan indikator utama dari struktur perangkat lunak dengan tingkat *evolvability* (kemudahan pengembangan) yang rendah. Dengan kata lain, semakin banyak *code smells* yang teridentifikasi, semakin besar potensi penurunan kualitas keterpeliharaan sistem. Hal ini diperkuat oleh penelitian Arcelli Fontana dkk., (2019) yang menunjukkan bahwa keberadaan *code smells* berkorelasi langsung dengan peningkatan *maintenance effort* serta *change proneness*, yang berdampak pada aspek *modularity* dan *modifiability* dalam ISO/IEC 25010.

Secara spesifik, penelitian ini membatasi ruang lingkup pada tiga *code smells*, yaitu *Large Class*, *Long Method*, dan *Duplicated Code*. Berdasarkan klasifikasi dari Mäntylä & Lassenius, (2006), *Large Class* dan *Long Method* termasuk kategori *Bloaters*, yakni elemen kode yang membesar tanpa terkendali sehingga meningkatkan kompleksitas dan menurunkan modularitas

serta kemudahan analisis. Penelitian Arcelli Fontana dkk., (2019) juga menemukan bahwa *Large Class* (sering disebut *God Class*) dan *Long Method* memiliki korelasi konsisten dengan degradasi arsitektur sistem. Sementara itu, *Duplicated Code* dikategorikan sebagai *Dispensables*, yaitu elemen yang seharusnya tidak diperlukan dalam desain yang baik.

Duplikasi kode secara langsung menghambat aspek *modifiability*, karena perubahan pada satu logika mengharuskan modifikasi di banyak lokasi, sehingga meningkatkan risiko inkonsistensi dan kerentanan terhadap perubahan. Oleh karena itu, deteksi dini terhadap ketiga *code smells* tersebut melalui *static code analysis* menjadi langkah strategis untuk menjaga dan meningkatkan kualitas *maintainability* pada *codebase* modul Surat Keterangan Mahasiswa.

Pendekatan *Agile Software* Development dengan ICONIX *Process* dipilih sebagai metode pengembangan. Metode ICONIX *Process* memungkinkan mekanisme iteratif pada tahapan pengembangan di fase kebutuhan, perancangan, dan implementasi (Rosenberg dkk., 2005).

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah dijelaskan pada sub bab sebelumnya, maka dapat dirumuskan masalah sebagai berikut:

1. Bagaimana mengembangkan modul Surat Keterangan Mahasiswa (AK.007) pada aplikasi *E-Office* FSM UNDIP menggunakan metode ICONIX *Process* untuk mengatasi kendala operasional persuratan manual?
2. Bagaimana menerapkan *static code analysis* untuk mendeteksi *code smells* (*Large Class*, *Long Method*, *Duplicated Code*) pada modul AK.007 Aplikasi *E-Office* FSM?
3. Bagaimana melakukan *refactoring* yang tepat untuk mengatasi *code smells* yang ditemukan tanpa mengubah perilaku eksternal sistem?
4. Bagaimana dampak *refactoring* terhadap peningkatan metrik kualitas kode dan performa aplikasi?

1.3 Tujuan Penelitian

Tujuan yang diharapkan dapat dicapai dari penelitian ini adalah:

1. Mengembangkan modul untuk Surat Keterangan Mahasiswa (AK.007) pada aplikasi *E-Office* FSM yang berfungsi dengan baik dan memiliki code quality yang tinggi.
2. Mengidentifikasi *code quality problems* pada *codebase* sistem *E-Office* FSM khususnya pada modul Surat Keterangan Mahasiswa (AK.007) menggunakan *static code analysis* dengan fokus pada *code smells*: *Large Class*, *Long Method*, dan *Duplicated Code*.
3. Melakukan *refactoring* terhadap *code* yang memiliki *code quality problems* untuk meningkatkan metrik kualitas kode sistem.
4. Mengevaluasi dampak penerapan *refactoring* terhadap peningkatan metrik kualitas kode (*code quality metrics*) dan performa aplikasi.

1.4 Manfaat Penelitian

Di bawah ini merupakan beberapa manfaat dari hasil penelitian yang telah dilakukan oleh penulis:

1. Meningkatkan efisiensi dan transparansi proses persuratan mahasiswa melalui implementasi *workflow* untuk Surat Keterangan Mahasiswa (AK.007).
2. Menyediakan *workflow* yang terstruktur melibatkan divisi Tata Usaha dan Akademik.
3. Meningkatkan kualitas kode aplikasi *E-Office* FSM melalui proses *refactoring* yang sistematis.

1.5 Ruang Lingkup

Dalam laporan skripsi ini perlunya batasan agar laporan skripsi ini tidak menyimpang dari tujuan pengembangan perangkat lunak dan tujuan yang telah ditetapkan. Batasan yang ditetapkan dalam penulisan ini adalah sebagai berikut:

1. Penelitian ini dilakukan pada sistem *E-Office* Fakultas Sains dan Matematika (FSM) Universitas Diponegoro.
2. Pengembangan fitur terbatas pada *workflow* untuk Surat Keterangan Mahasiswa (AK.007).
3. *Workflow* yang diimplementasikan melibatkan dua aktor utama, yaitu divisi Tata Usaha dan divisi Akademik.

4. *Static code analysis* difokuskan pada *codebase* modul Surat Keterangan Mahasiswa (AK.007) yang dikembangkan dalam penelitian ini, dengan fokus pada *backend code* (*controllers*, *services*).
5. Pemilihan *backend* sebagai objek analisis didasarkan pada pertimbangan bahwa *backend* berperan sebagai pusat logika bisnis dan pengolahan data, sehingga kualitas kode pada sisi ini memiliki dampak risiko dan performa yang lebih signifikan terhadap integritas sistem secara keseluruhan.
6. Analisis *code quality problems* dibatasi pada *code smells*, khususnya: *Large Class*, *Long Method*, dan *Duplicated Code*.
7. Teknologi yang digunakan dalam pengembangan aplikasi ini adalah UmiJS sebagai *FrontEnd* dan ElysiaJS sebagai *Backend* dengan *runtime* Bun.
8. *Database* yang digunakan adalah PostgreSQL.
9. Metode pengembangan perangkat lunak yang digunakan adalah ICONIX *Process*.
10. Pengukuran *code quality* menggunakan SonarQube untuk analisis statis. Pengukuran *runtime performance* Postman untuk membandingkan *execution time* sebelum dan sesudah *refactoring* sebagai validasi *improvement*.
11. *Refactoring* dibatasi pada *code* yang teridentifikasi memiliki *code smells* berdasarkan hasil *static code analysis*.
12. Validasi *improvement* dilakukan melalui pengukuran ulang *metrics* setelah *refactoring* dan perbandingan dengan *baseline metrics* sebelum *refactoring*.

1.6 Sistematika Penulisan

Sistematika penulisan yang digunakan dalam dokumen skripsi ini terbagi menjadi beberapa pokok bahasan berikut:

BAB I PENDAHULUAN

Bab ini berisikan pembahasan mengenai latar belakang masalah yang menguraikan pentingnya *static code analysis* sebagai mekanisme deteksi *code quality problems*, permasalahan pada proses persuratan di FSM UNDIP, serta gap dalam deteksi *code smells* sejak tahap awal pengembangan. Selain itu, bab ini juga memuat rumusan masalah yang fokus pada deteksi *code smells* (*Large Class*, *Long Method*, *Duplicated Code*) melalui *static code analysis*, tujuan

penelitian yang mencakup implementasi dan dokumentasi siklus analisis-*refactoring*-validasi, manfaat penelitian bagi FSM UNDIP dan *software developers*, ruang lingkup penelitian yang membatasi fokus pada *workflow* Surat Keterangan Mahasiswa (AK.007), serta sistematika penulisan.

BAB II LANDASAN TEORI

Bab ini berisikan pembahasan mengenai landasan teori yang diacu dan diimplementasikan oleh penulis dalam skripsi ini, meliputi: konsep *static code analysis* dan *tools* yang digunakan (SonarQube), *code quality* dan *code smells*, *metrics* untuk mengukur *code quality*, *workflow*, ICONIX *Process* sebagai metodologi pengembangan, serta teknologi yang digunakan dalam implementasi sistem *E-Office* FSM.

BAB III METODE PENELITIAN

Bab ini berisikan pembahasan mengenai metode penelitian yang digunakan oleh penulis dalam skripsi ini menggunakan ICONIX *Process* yang terdiri atas tahap *requirements analysis*, *analysis* dan *preliminary design*, *detailed design*, *implementation* dengan fokus pada *static code analysis* dan *refactoring*, serta testing dan validasi. Selain itu, bab ini juga menjelaskan metodologi untuk mengidentifikasi *code smells* menggunakan SonarQube, *metrics* yang digunakan untuk mengukur *code quality*, prosedur *refactoring* berdasarkan hasil analisis, serta prosedur validasi *improvement* menggunakan pengukuran ulang *metrics* dan *runtime performance comparison*.

BAB IV HASIL DAN PEMBAHASAN

Bab ini berisikan pembahasan mengenai pengimplementasian dan hasil implementasi dari landasan teori dan metode yang telah ditentukan. Pembahasan mencakup: hasil implementasi fitur *workflow* Surat Keterangan Mahasiswa (AK.007), dokumentasi *code smells* yang ditemukan melalui *static code analysis* dengan SonarQube, proses *refactoring* yang dilakukan dan dampaknya terhadap *metrics* kualitas kode, analisis *improvement* pada *code quality* menggunakan *metrics* yang telah ditentukan, perbandingan *runtime performance* sebelum dan

sesudah *refactoring*, serta *lesson learned* dan *guidelines* untuk mencegah *architectural degradation* pada pengembangan berikutnya.

BAB V PENUTUP

Bab ini berisikan mengenai kesimpulan dari penelitian yang menjawab rumusan masalah tentang bagaimana *static code analysis* dapat mengungkap dan memperbaiki *code quality*, serta saran untuk penelitian lanjutan dan implementasi di sistem lain berdasarkan hasil penelitian yang berjudul "Rancang Bangun Aplikasi Permohonan Surat Keterangan Mahasiswa dengan Deteksi dan Refactoring Masalah Kualitas Kode Menggunakan Static Code Analysis".