BAB 1 PENDAHULUAN

1.1. Latar Belakang

Low Code Development Platform (LCDP) adalah perangkat lunak pada cloud yang memiliki target market non-programmer yang ingin membangun aplikasi tanpa pengetahuan pemograman yang mendalam. Karena memanfaatkan antarmuka pengguna grafis dan konfigurasi yang dapat disesuaikan, platform ini memungkinkan pengembangan aplikasi dengan lebih cepat dan efisien [1]. Menurut laporan State of Application Development 2025, 74% perusahaan ingin mengembangkan sepuluh aplikasi atau lebih dalam setahun ke depan. Mereka akan mengadopsi teknologi seperti AI generatif, pengembangan berbasis AI, dan platform *low-code* untuk menciptakan aplikasi yang berfokus pada pengalaman pelanggan [2].

Terdapat tantangan dalam pengujian di LCDP yang hanya menggunakan *Low-Code Testing Framework* (LCTF) bawaan platform sering kali terbatas oleh fitur yang sudah disediakan [1]. Outsystems menggunakan BDD Framework sebagai LCTF. BDD *Framework* menggunakan format *Given-When-Then* yang tidak cukup presisi, dan pengujian yang dihasilkan tidak dapat mencakup skenario tes secara keseluruhan. Tantangan tersebut menyebabkan hasil skenario tes yang tidak lengkap dan tidak merepresentasikan model sistem secara menyeluruhS [3], [4], [5].

Pengujian LCDP dapat menerapkan *Data Driven Testing* (DDT), *Model-Based Testing* (MBT), *Test-Driven Development* (TDD), maupun *record-and-replay* [6], [7]. OutSystems mengadopsi pendekatan model sebagai bahasa visual untuk mendefinisikan *model data*, *business logic*, *workflow process*, dan UI pada platform ini, sehingga menciptakan sinergi antara development dan testing menggunakan MBT [1]. Dengan MBT, tester dapat secara otomatis menghasilkan kasus uji yang dapat meningkatkan efektivitas dan ketelitian pengujian sistem dan memberikan deskripsi terbaik tentang perilaku sistem [5]. Ini membantu tester melakukan pengujian yang lebih luas dan mendalam dibandingkan jika tester

menulis kasus uji secara manual. Dengan kata lain, MBT memungkinkan tester memastikan sistem berfungsi seperti yang diharapkan dengan cara yang lebih efisien dan menyeluruh [8]. *Extended Finite State Machine* (EFSM) digunakan sebagai model pengujian MBT karena mampu memformalkan perilaku sistem dan sangat cocok karena EFSM mendukung pengambilan keputusan melalui *guard* pada transisinya. *Guard* ini penting dalam merepresentasikan keputusan (*decision*) yang terjadi dalam model [9].

Meskipun memiliki keunggulan, adopsi MBT di lingkungan *low-code* seperti OutSystems masih terbatas. Dengan adanya kekurangan cakupan pengujian oleh BDD *framework* pada Outsystems , maka tujuan utama dari Tugas Akhir (TA) ini adalah penggunaan MBT untuk meningkatkan cakupan serta ketelitian pengujian aplikasi pada platform *low-code* OutSystem dan meneliti bagaimana model *Action Flow* Outsystems dapat dikonversi menjadi model formal seperti EFSM. Sehingga dari kedua tujuan tersebut menghasilkan dua masalah utama yang dibahas dalam penelitian ini: (1) upaya proses konversi dari model *Action Flow* pada Outsystems ke UML *Activity Diagram* menjadi model EFSM, dan (2) mengatasi kurangnya cakupan pengujian pada *framework* saat ini. Studi ini memperkenalkan sistem untuk mengubah *action flow* OutSystems menjadi model EFSM, memfasilitasi pengujian integrasi berbasis model yang sebelumnya tidak dapat dilakukan di lingkungan OutSystems menggunakan aturan transformasi sistematis, menghubungkan *business logic low-code* dengan pengembangan *test* model formal. *TestOptimal* digunakan untuk melakukan MBT, menilai *test case*, dan memberikan *feedback* terhadap hasil eksekusi dibandingkan perilaku yang diharapkan di EFSM. Alat TestOptimal juga cocok untuk menggunakan EFSM sebagai bahasa pemodelan [10].

1.2. Rumusan Masalah

Berangkat dari kekurangan cakupan pengujian yang ada pada BDD *Framework* di Outsystems, diangkat identifikasi masalah utama dari pengujian pada LCDP Outsystems di antaranya:

- Bagaimana proses konversi dari model *Action Flow* pada Outsystems ke UML Activity Diagram menjadi model EFSM ?
- Bagaimana mengatasi kurangnya cakupan pengujian pada LCDP Outsystems yang ada saat ini?

1.3. Tujuan dan Manfaat

Tujuan serta manfaat dari penelitian ini di antaranya:

- Mengembangkan pendekatan pengujian Model-Based Testing (MBT) pada platform Low-Code Development Platform (LCDP) Outsystems dengan konversi model Action Flow ke UML AD menjadi model EFSM.
- Mengatasi kurangnya cakupan pengujian pada kerangka kerja pengujian yang ada saat ini dengan mengimplementasikan pendekatan MBT.

1.4. Batasan Masalah

Dalam pelaksanaan tugas akhir ini, terdapat beberapa batasan yang ditetapkan agar ruang lingkup penelitian menjadi lebih fokus dan dapat diselesaikan secara optimal sesuai waktu dan sumber daya yang tersedia. Batasan-batasan tersebut meliputi:

- Pembuatan model EFSM dibatasi oleh kapasitas kanvas pada TestOptimal yang hanya dapat menampung maksimal 97 *nodes* dengan sebuah *initial state* dan *end state* yang membuat model sulit dibaca, sehingga model dibuat terpisah berdasarkan fungsi aplikasi, tidak dalam satu kanvas utuh.
- Pengujian difokuskan hanya pada studi kasus fitur utama aplikasi CustomerApp seperti *login*, operasi CRUD data *customer*, pencarian, dan pengurutan data.
- Data uji yang digunakan terbatas pada jumlah dan variasi yang sederhana serta skenario umum aplikasi bisnis sederhana.

- Proses konversi *action flow* dari OutSystems ke EFSM dilakukan secara semi-otomatis dengan masih terdapat intervensi manual.

Batasan-batasan ini diharapkan dapat mengarahkan penelitian agar lebih terfokus dan memberikan ruang bagi pengembangan lanjutan di masa depan, terutama dalam hal cakupan fitur, otomatisasi model, dan penerapan pada platform lain.

1.5. Metode Penelitian

Pada penelitian ini, metode yang digunakan berpedoman pada kerangka *Software Testing Life Cycle* (STLC) sebagai acuan utama dalam seluruh tahapan pengujian MBT pada aplikasi yang dikembangkan dengan LCDP OutSystems. Penerapan STLC dipilih untuk memastikan tata kelola proses pengujian berjalan sistematis, terstruktur, dan terukur sesuai standar industri serta dapat mendukung proses validasi mulai dari perencanaan hingga evaluasi hasil testing model EFSM.

1.6. Sistematika Penulisan

Sistematika penulisan tugas akhir ini terdiri dari enam bab utama, dimulai dari Bab Pendahuluan yang memaparkan latar belakang, tujuan, rumusan masalah, dan manfaat penelitian; dilanjutkan Bab Tinjauan Pustaka yang membahas LCDP, Model-Based Testing (MBT), OutSystems, EFSM, dan metrik evaluasi pengujian; Bab Perancangan Sistem yang menguraikan metode penelitian STLC serta menjelaskan proses transformasi *Action Flow* ke UML Activity Diagram dan EFSM sebagai solusi; Bab Hasil dan Analisis yang mendokumentasikan tahapan pengembangan sistem serta mengevaluasi hasil konversi *Action Flow* menjadi EFSM serta capaian *test coverage* dengan membandingkan MBT dengan BDD; diakhiri Bab Kesimpulan dan Saran yang merangkum temuan utama, kontribusi penelitian, rekomendasi pengembangan lebih lanjut, dan harapan terhadap adopsi solusi yang diusulkan di industri dan penelitian lanjutan.