1. Pendahuluan

1.1. Latar Belakang

Aplikasi Presentia merupakan sistem absensi berbasis *web* yang dirancang untuk mempermudah pengelolaan kehadiran siswa di sekolah. *Backend* aplikasi ini dikembangkan menggunakan *framework* Laravel, dengan fokus pada pengelolaan data absensi. Sistem ini diharapkan dapat menyediakan solusi absensi modern yang mengatasi kekurangan metode manual, seperti pencatatan data yang lambat dan tidak akurat [1]. Selain itu, berdasarkan observasi terhadap sistem absensi manual yang masih banyak digunakan, ditemukan bahwa proses pencatatan kehadiran membutuhkan banyak tenaga kerja dan sulit memantau kehadiran siswa secara *real-time* [1]. Hal ini mendorong kebutuhan untuk menciptakan sistem yang lebih efisien dan dapat diimplementasikan pada berbagai institusi pendidikan. Pengembangan Presentia dilakukan secara iteratif dan cepat untuk merespons kebutuhan lapangan yang mendesak, di mana sistem diharapkan dapat segera digunakan oleh sekolah-sekolah yang membutuhkan solusi digital absensi. Selain itu, Presentia juga dirancang sebagai aplikasi *multi-tenant*, sehingga dapat digunakan secara bersamaan oleh banyak sekolah.

Dalam pengembangan perangkat lunak, otomatisasi menjadi salah satu langkah penting untuk meningkatkan efisiensi dan konsistensi. Alat seperti GitHub Actions, sebuah *tool* berbasis *event* untuk mengotomatisasi alur kerja pengembangan, kini semakin sering digunakan di repositori perangkat lunak. GitHub Actions menangani tugas-tugas berulang, seperti pengujian dan integrasi, dengan lebih efisien [2]. Meskipun baru sebagian kecil repositori yang mengadopsi GitHub Actions, alat ini memiliki respon positif karena dapat membantu pengembang menulis kode kustom untuk mengotomatisasi *workflow* mereka dan meningkatkan efisiensi pengembangan [2]. Dalam konteks pengembangan aplikasi Presentia, GitHub Actions digunakan untuk mengotomatisasi proses pengujian *backend* secara langsung setiap kali terjadi perubahan kode, sehingga setiap iterasi pengembangan dapat segera divalidasi tanpa keterlibatan manual. Hal ini sangat mendukung karakteristik pengembangan Presentia yang cepat dan terus berkembang, sekaligus memastikan bahwa kualitas fungsional tetap terjaga di tengah proses integrasi berkelanjutan. Dengan kemampuan ini, pengujian otomatis dapat dilakukan lebih cepat dan efisien, yang mendukung kualitas perangkat lunak secara keseluruhan.

Saat ini, masih terdapat perbedaan signifikan antara pengujian manual yang cenderung memakan waktu, rentan terhadap kesalahan manusia, dan bergantung pada keahlian individu, dengan pengujian otomatis yang menawarkan proses yang lebih cepat, efisien, dan dapat diulang secara konsisten[3], [4], [5], [6]. Pengujian otomatis memungkinkan pelaksanaan sejumlah besar pengujian dalam waktu singkat, sementara pengujian manual membutuhkan keterlibatan langsung penguji untuk menargetkan bagian sistem yang dianggap lebih rentan terhadap kesalahan [7]. Selain itu, proses pengujian otomatis dapat memperpendek siklus hidup pengujian melalui alat atau utilitas khusus, sehingga meningkatkan efisiensi secara keseluruhan [7]. Hal inilah yang menjadi alasan utama dipilihnya pengujian otomatis sebagai fokus utama penelitian ini.

Berdasarkan permasalahan yang telah disebutkan sebelumnya, penelitian ini mengusulkan penerapan pengujian otomatis berbasis *unit testing* dan *integration testing* pada backend aplikasi Presentia yang dikembangkan menggunakan Laravel. Tujuan utama dari pengujian perangkat lunak adalah untuk memastikan bahwa sistem bekerja sesuai dengan kebutuhan pengguna [8]. Pengujian otomatis dengan *unit testing* dan *integration testing* dapat mengurangi jumlah *bug* dalam aplikasi sejak tahap awal, sehingga meningkatkan kualitas perangkat lunak secara keseluruhan [8]. Selain itu, pengujian otomatis memungkinkan aplikasi diuji dengan efisien menggunakan alat otomatisasi, yang tidak hanya menghemat waktu tetapi juga memastikan setiap komponen sistem berfungsi sesuai tujuannya [8]. Proses ini membantu pengembang mengurangi usaha manual yang diperlukan dan memastikan pengujian lebih konsisten [8].

1.2. Topik dan Batasannya

Berdasarkan latar belakang yang telah dipaparkan di atas, penelitian ini berfokus pada pengujian otomatis untuk aplikasi Presentia, yaitu sebuah sistem absensi berbasis *web* yang dikembangkan menggunakan *framework* Laravel. Pengujian dilakukan dengan menerapkan metode *unit testing* dan *integration testing* untuk memastikan bahwa setiap unit dan alur fungsional pada sistem berjalan sesuai dengan kebutuhan yang telah ditentukan. Pengujian ini terintegrasi dengan GitHub Actions sebagai alat otomatisasi, yang digunakan untuk menjalankan proses validasi secara berkelanjutan setiap terjadi perubahan kode. Lingkup pembahasan dibatasi pada pengujian sisi *backend*, tanpa mencakup aspek *frontend* atau pengujian sistem secara keseluruhan. Proses pengujian difokuskan pada tahapan analisis kebutuhan pengujian, perencanaan *test case* berdasarkan spesifikasi fungsional, implementasi pengujian otomatis, serta evaluasi terhadap efisiensi dan efektivitas pengujian menggunakan *workflow* GitHub Actions.

Ada beberapa batasan masalah yang menjadi fokus pada penelitian ini agar penelitian lebih terarah dan terstruktur:

a. Jenis pengujian yang diterapkan terbatas pada *unit testing* dan *integration testing* pada *backend*, tanpa mencakup pengujian lainnya seperti *end-to-end testing* atau *performance testing*, karena pengujian lainnya melibatkan cakupan luar yang tidak terkait dengan *backend*.
b. *Framework* Laravel digunakan sebagai *backend* utama dalam penelitian ini, sehingga implementasi

pengujian hanya mencakup ekosistem Laravel.

c. PHPUnit digunakan sebagai alat utama dalam pembuatan *test case* untuk unit dan integrasi.
d. GitHub Actions digunakan sebagai alat otomatisasi untuk menjalankan *workflow* pengujian secara otomatis, mengingat Presentia dikembangkan dalam repository GitHub. Oleh karena itu, tidak dilakukan perbandingan dengan alat CI/CD lain.
e. Efisiensi pengujian diukur berdasarkan perbandingan waktu eksekusi antara pengujian manual dan otomatis, tanpa mempertimbangkan faktor lain seperti konsumsi sumber daya atau spesifikasi infrastruktur.

1.3. Pertanyaan dan Tujuan Penelitian

Berdasarkan uraian topik permasalahan sebelumnya, berikut adalah pertanyaan-pertanyaan yang menjadi dasar dari penelitian ini:

a. Bagaimana penerapan pengujian otomatis berbasis *unit* dan *integration testing* pada pengembangan aplikasi Presentia dapat memastikan bahwa setiap unit dan fungsi berjalan dengan baik sesuai dengan kebutuhan fungsional yang telah ditentukan?
b. Bagaimana GitHub Actions sebagai alat otomatisasi dapat mendukung implementasi pengujian otomatis berbasis *unit* dan *integration* untuk memastikan efisiensi waktu pengujian dan konsistensi dalam setiap perubahan kode?

Melalui pertanyaan-pertanyaan tersebut, penelitian ini bertujuan untuk mengembangkan *workflow* pengujian otomatis berbasis unit dan integrasi yang dapat memastikan bahwa setiap unit dan fungsi dalam aplikasi Presentia bekerja sesuai kebutuhan fungsional yang telah ditentukan. Selain itu, penelitian ini juga bertujuan untuk membuktikan bahwa penerapan GitHub Actions sebagai alat otomatisasi dapat meningkatkan efisiensi pengujian dengan mengurangi waktu eksekusi hingga X% dan memastikan konsistensi hasil pengujian dalam setiap iterasi perubahan kode.

1.4. Organisasi Tulisan

Jurnal Tugas Akhir ini terdiri dari beberapa bagian, yaitu pada Bagian 2 Studi Terkait membahas studi-studi sebelumnya serta teori-teori yang menjadi dasar penelitian ini. Bagian 3 Alur Pengujian menjelaskan metodologi yang diterapkan dan tahapan-tahapan pengujian yang dilakukan. Selanjutnya, Bagian 4 berisi evaluasi yang mencakup hasil pengujian dan analisis terhadap hasil pengujian yang telah dilakukan. Terakhir, Bagian 5 Kesimpulan berisi ringkasan penelitian dan saran untuk penelitian selanjutnya.