BAB 1. PENDAHULUAN

1.1 Latar Belakang

Perkembangan perangkat lunak telah menjadi komponen utama dalam aspek kehidupan, mulai dari bisnis, pendidikan, kesehatan, hingga hiburan. Dengan meningkatnya kebutuhan pengguna, kualitas perangkat lunak sangat diperhatikan pada pengembangan perangkat lunak. Pengujian perangkat lunak merupakan cara penting untuk memastikan kualitas perangkat lunak [1]. Hal ini bertujuan untuk memaksimalkan fitur perangkat lunak seperti fungsionalitas dan kegunaan. *Test-Driven Development* merupakan salah satu pengembangan perangkat. *Test-Driven Development* di rancang untuk memastikan perangkat lunak berkualitas sejak tahap awal pengembangan.

Test-Driven Development adalah metode pengembangan perangkat lunak yang berfokus pada penulisan pengujian, dengan pengembang menulis pengujian terlebih dahulu sebelum menulis kode yang dijalankan [2]. TDD mendorong pengembang untuk menulis kasus uji terlebih dahulu sebagai persyaratan kode, lalu mendefinisikan kode hingga semua kasus uji terpenuhi, sehingga mengurangi kesalahan dan memastikan penilaian kode berkelanjutan [3]. Tetapi TDD memiliki beberapa tantangan, salah satunya yaitu kesulitan dalam menulis tes yang efektif. Menulis tes yang efektif sangat di perlukan dalam pengujian perangkat lunak. Karena akan mempengaruhi kualitas perangkat lunak tersebut. Semakin baik hasil pengujiannya semakin baik juga kualitas perangkat lunaknya. Kesulitan dalam menulis pengujian yang efektif dan kurangnya keterampilan dalam mengembangkan pengujian yang baik menjadi hambatan bagi banyak pengembang [4].

Large Language Models (LLM) menjadi salah satu solusi untuk tantangan pada TDD. Perkembangan LLM ini juga mengalami perkembangan yang cukup signifikan. LLM menunjukkan kemampuan luar biasa dalam

berbagai tugas NLP, seperti penerjemahan mesin, menjawab pertanyaan, meringkas, pembuatan teks, pemeriksaan tata bahasa, dan sebagainya [5]. LLM ini mampu menghasilkan dan memahami bahasa setara manusia dengan konteks yang mendalam dan makna yang kaya [6]. *Large Language Models* (LLM) memungkinkan konversi dari bahasa alami ke bahasa pemrograman, menurunkan hambatan dalam penulisan kode [7].

Seiring dengan perkembangan LLM yang pesat, berbagai model tersedia dengan karakteristik dan keunggulannya masing-masing. Oleh karena itu, pemilihan model yang tepat menjadi penting untuk memastikan hasil yang optimal. Dari beberapa pilihan yang ada, penelitian ini memanfaatkan LLaMA *3* yang dapat diakses secara gratis melalui GroqCloud dengan kecepatan *inference* yang tinggi, sehingga mendukung replikasi penelitian tanpa biaya tinggi. Model ini dipilih karena memiliki performa yang kompetitif dalam berbagai *benchmark*, bahkan dalam beberapa kasus mengungguli model besar lainnya dalam kelas parameter yang setara. Misalnya, pada *benchmark* HumanEval untuk kemampuan pemrograman, LLaMA 3 Instruct 70B mengungguli Mistral-7B dan Gemma-2 9B serta mendekati skor GPT-3.5 Turbo [8].

Dengan memanfaatkan LLM dengan model LLAMA 3, diharapkan dapat meningkatkan efisiensi dan efektivitas dalam pengembangan perangkat lunak berbasis TDD terutama dalam penulisan pengujian. Selain itu, penelitian ini juga diharapkan dapat memberikan manfaat dalam memanfaatkan teknologi LLM dengan evaluasi yang diberikan.

1.2 Rumusan Masalah

Berdasarkan latar belakang tersebut, berikut adalah rumusan masalah yang diangkat dalam penelitian ini:

a. Bagaimana pemanfaatan *Large Language Models* (LLM) dapat mendukung proses implementasi *Test-Driven Development* (TDD) dalam menghasilkan Tes yang sesuai dengan spesifikasi masalah?

b. Bagaimana efektivitas LLM dalam mendukung pengembangan perangkat lunak berbasis TDD?

1.3 Tujuan

Berdasarkan perumusan masalah yang telah disampaikan, penelitian ini bertujuan untuk:

a. Mengimplementasikan *Large Language Models* (LLM) dalam proses pengembangan perangkat lunak berbasis *Test-Driven Development* (TDD).
b. Mengevaluasi kinerja LLM dalam membantu penulisan pengujian dan generasi kode pada proses pengembangan perangkat lunak.

1.4 Manfaat

Penelitian ini diharapkan dapat memberikan manfaat sebagai berikut:

a. Menyediakan referensi bagi peneliti dan praktisi dalam mengintegrasikan LLM untuk penulisan pengujian dan pembuatan kode secara otomatis.
b. Menyajikan informasi mengenai efektivitas penggunaan LLM dalam pengembangan perangkat lunak, khususnya yang berbasis TDD.

1.5 Batasan Masalah

Penelitian ini memiliki beberapa batasan untuk memastikan fokus yang jelas dan pencapaian tujuan penelitian, yaitu:

a. **Lingkup proyek:** Penelitian ini menggunakan proyek Pretix, yang merupakan aplikasi manajemen tiket berbasis Python. Proyek ini dipilih karena memiliki kompleksitas fitur yang memadai, dokumentasi yang baik, dan struktur kode yang jelas, sehingga relevan untuk menguji kemampuan LLM dalam skenario *Test-Driven Development* (TDD). Penggunaan satu proyek dipilih untuk menjaga konsistensi

konteks pengujian dan memungkinkan analisis yang lebih mendalam dan terkontrol terhadap efektivitas LLM.

b. **Model yang digunakan:** Penelitian ini menggunakan model LLaMA 3, yang diakses melalui layanan penyedia model bernama GroqCloud. GroqCloud di sini berfungsi sebagai infrastruktur *inference* dan bukan merupakan model itu sendiri. Pemilihan LLaMA *3* dilakukan karena model ini memiliki kemampuan pemrosesan bahasa alami yang kuat serta ketersediaan akses gratis melalui GroqCloud, sehingga mendukung replikasi penelitian tanpa biaya tinggi.

c. **Fokus penelitian:** Penelitian difokuskan pada proses *pembuatan test case* otomatis berdasarkan *requirement*, khususnya *requirement* untuk API dan tidak mencakup pembuatan kode fungsional atau logika *backend* dari sistem. Pembatasan ini dilakukan untuk mempersempit ruang lingkup penelitian, memfokuskan evaluasi pada kemampuan LLM menghasilkan *test case*, serta menghindari variabel tambahan yang dapat memengaruhi hasil analisis.

d. **Metrik Evaluasi:** Penilaian kualitas *test case* dilakukan menggunakan tiga metrik evaluasi otomatis, yaitu:

    1) BLEU, untuk mengevaluasi kesamaan urutan token antar *test case*. BLEU dipilih karena merupakan metrik umum dalam *Natural Language Processing* yang efektif mengukur kesamaan secara tekstual, sehingga berguna untuk membandingkan *test case* yang dihasilkan LLM dengan *test case* acuan.
    2) CodeBLEU, untuk menilai kemiripan struktur dan logika kode. Metrik ini dipilih karena dirancang khusus untuk evaluasi kode pemrograman, dengan mempertimbangkan tidak hanya kesamaan teks, tetapi juga kesesuaian sintaks dan semantik kode.
    3) chrF, untuk menilai kemiripan pada tingkat karakteristik penulisan. chrF dipilih untuk menangkap kesamaan pada

tingkat penulisan, yang penting ketika struktur kata atau token berbeda tetapi maksud pengujian tetap sama.

Pemilihan ketiga metrik ini dimaksudkan untuk memberikan penilaian yang lebih komprehensif, menggabungkan evaluasi dari aspek struktur kode, kesamaan teks, dan kesesuaian karakter.

1.6 Metode Penelitian

Penelitian ini menggunakan pendekatan eksperimental untuk mengevaluasi efektivitas model *Large Language Model* (LLM) dalam menghasilkan *test case*. Beberapa tahapan utama dalam metode penelitian ini meliputi:

a. Studi Literatur: Peneliti melakukan kajian pustaka secara mendalam terhadap literatur yang relevan terkait *Test-Driven Development* (TDD), *Large Language Models* (LLM), *Test Generation,* dan pengujian perangkat lunak.
b. Perancangan: Proses perancangan dimulai dengan menyiapkan proyek yang akan dipakai. Selanjutnya, pemilihan model LLM yang akan di gunakan untuk membuat *test*. Di tahap ini juga di lakukan pengumpulan data spesifikasi atau *requirement* fungsional proyek.
c. Implementasi: Proses implementasi mencakup pembuatan *test case* berdasarkan data *requirement* yang telah di kumpulkan.
d. Pengujian dan Evaluasi: Setelah *test case* di buat, dilakukan pengujian dan evaluasi dengan membandingkan *test* yang dihasilkan oleh LLM dengan *test* yang sudah ada menggunakan metrik CodeBLEU. Metrik ini digunakan untuk mengukur *test* yang dihasilkan oleh LLM.

Dengan metode penelitian ini, diharapkan dapat meningkatkan efisiensi dan efektivitas dalam pengembangan perangkat lunak berbasis TDD terutama dalam penulisan pengujian. Selain itu, metode penelitian ini juga diharapkan dapat memberikan manfaat dalam memanfaatkan teknologi LLM dengan evaluasi yang diberikan.

1.7 Jadwal Pelaksanaan

Jadwal pelaksanaan penelitian ini disusun dalam bentuk *milestone* untuk memastikan bahwa setiap tahapan dilaksanakan tepat waktu. Berikut adalah jadwal pelaksanaan yang direncanakan untuk penelitian ini:

Tabel 1.1. Jadwal Pelaksanaan Tugas Akhir.

No.	Deskripsi Tahapan	Bulan 1	Bulan 2	Bulan 3	Bulan 4	Bulan 5	Bulan 6
1	Studi Literatur						
2	Perancangan						
3	Implementasi						
4	Pengujian dan Evaluasi						
5	Penyusunan Laporan/Buku TA						