Rancang Bangun Asisten *QA Hybrid* NLP dan *Rule-based* untuk *User Story* Indonesia

Design and Development of a Hybrid NLP and Rule-based QA Assistant for Indonesian User Stories

[1]Cheria Sevani Apiani*, [2]Ichsan Ibrahim
[1,2]Program Studi Teknik Informatika, STMIK Indonesia Mandiri Bandung
[1,2]Jl. Belitung No.7, Merdeka, Kec. Sumur Bandung, Kota Bandung, Jawa Barat, Indonesia
*e-mail: cheriaapiani@gmail.com

(***received*:** 30 April 2026, ***revised*:** 6 May 2026, ***accepted*:** 7 May 2026)

Abstrak

Penelitian ini bertujuan untuk mengatasi tantangan dalam pengujian perangkat lunak berbasis *Agile*, di mana penyusunan *test case* secara manual dari *user story* sering kali memakan waktu dan menghasilkan kualitas yang tidak konsisten. Meskipun teknik *Natural Language Processing* (NLP) dan sistem *rule-based* telah diusulkan, masing-masing memiliki keterbatasan dalam menangani ambiguitas linguistik dan variasi struktur kalimat, khususnya pada bahasa Indonesia. Penelitian ini mengusulkan sebuah asisten *Quality Assurance* (*QA*) dengan pendekatan *hybrid* yang mengintegrasikan model *Named Entity Recognition* (NER) berbasis *IndoBERT* dengan sistem *rule-based* deterministik. Model NER digunakan untuk mengekstraksi elemen fungsional (aktor, aksi, objek, kondisi, dan hasil yang diharapkan), sementara sistem *rule-based* memetakan elemen tersebut ke dalam *template test case* yang terstruktur. Evaluasi kualitatif oleh praktisi *QA* menunjukkan bahwa pendekatan *hybrid* mencapai skor rata-rata 4,67 pada skala Likert 5 poin, melampaui pendekatan *NLP-only* (3,87) dan *rule-only* (4,60). Sistem ini terbukti meningkatkan efisiensi proses pengujian hingga lebih dari 99% dengan menghasilkan *test case* yang lebih lengkap, terbaca, dan memiliki keterlacakan yang baik. Temuan ini menegaskan bahwa integrasi fleksibilitas NLP dan konsistensi *rule-based* sangat efektif dalam mengotomatisasi *QA* untuk konteks lokal Indonesia.

Kata kunci: pendekatan hibrida, *user story* Indonesia, pemrosesan bahasa alami, sistem berbasis aturan, generasi *test case*,

Abstract

This study aims to address challenges in Agile-based software testing, where manually creating test cases from user stories is often time-consuming and produces inconsistent quality. Although Natural Language Processing (NLP) techniques and rule-based systems have been proposed, each approach has limitations in handling linguistic ambiguity and variations in sentence structure, particularly in the Indonesian language context. This research proposes a hybrid Quality Assurance (QA) assistant that integrates an IndoBERT-based Named Entity Recognition (NER) model with a deterministic rule-based system. The NER model is used to extract functional elements, including actors, actions, objects, conditions, and expected outcomes, while the rule-based system maps these elements into structured test case templates. Qualitative evaluation conducted by QA practitioners showed that the hybrid approach achieved an average score of 4.67 on a 5-point Likert scale, outperforming both the NLP-only approach (3.87) and the rule-only approach (4.60). The proposed system was proven to improve testing efficiency by more than 99% while generating test cases that are more complete, readable, and traceable. These findings confirm that integrating the flexibility of NLP with the consistency of rule-based systems is highly effective for automating Quality Assurance processes in the Indonesian local context.

Keywords: *hybrid approach, indonesian user stories, natural language processing, rule-based systems, test case generation*

 ISSN:2302-8149
e-ISSN:2540-9719

1 Pendahuluan

Dalam pengembangan perangkat lunak modern, pengujian merupakan tahap kritis yang bertujuan menemukan cacat sistem sebelum produk dirilis kepada pengguna. Proses pengujian perangkat lunak diketahui menyerap proporsi yang signifikan dari keseluruhan sumber daya pengembangan, yakni berkisar antara 40% hingga 60% dari total waktu, biaya, dan tenaga yang dialokasikan [1]. Seiring dengan meningkatnya kompleksitas sistem perangkat lunak, aktivitas pengujian juga menjadi semakin penting karena kegagalan perangkat lunak dapat menimbulkan kerugian baik dari sisi teknis maupun bisnis.

Salah satu komponen penting dalam proses pengujian perangkat lunak adalah penyusunan *test case*. *Test case* merupakan skenario pengujian yang dirancang untuk memastikan apakah sistem berjalan sesuai dengan spesifikasi kebutuhan yang telah ditetapkan. Perancangan *test case* yang baik memungkinkan proses pengujian dilakukan secara sistematis dan terstruktur sehingga kesalahan pada perangkat lunak dapat diidentifikasi secara lebih efektif. Namun, dalam praktiknya proses penyusunan *test case* sering kali dilakukan secara manual oleh tim *Quality Assurance* (*QA*) sehingga membutuhkan waktu yang cukup lama dan berpotensi menghasilkan inkonsistensi kualitas antar penguji [2]. Hal ini menjadi tantangan tersendiri terutama ketika sistem perangkat lunak yang dikembangkan memiliki kompleksitas tinggi dan perubahan kebutuhan yang cepat.

Dalam pengembangan perangkat lunak berbasis *Agile*, kebutuhan sistem umumnya dinyatakan dalam bentuk *user story* yang ditulis menggunakan bahasa alami. Dalam lingkungan *Agile*, *user story* merupakan penghubung antara pelanggan dan tim pengembang, sekaligus menjadi pilar utama dalam memahami kebutuhan produk [3]. Namun demikian, penggunaan bahasa alami dalam *user story* juga menimbulkan sejumlah tantangan dalam proses pengujian perangkat lunak. Struktur kalimat yang bervariasi, potensi ambiguitas makna, serta informasi yang tidak selalu lengkap dapat menyulitkan proses transformasi kebutuhan sistem menjadi skenario pengujian yang terstruktur. Selain itu, dalam lingkungan *Agile* kebutuhan sistem dapat berubah secara cepat sehingga proses pengelolaan kebutuhan harus dilakukan secara berkelanjutan sepanjang proses pengembangan perangkat lunak [3].

Untuk mengatasi permasalahan tersebut, berbagai penelitian telah mengusulkan pendekatan otomatisasi dalam generasi *test case* dari dokumen kebutuhan perangkat lunak menggunakan teknik *Natural Language Processing* (NLP). Teknik-teknik NLP yang paling banyak digunakan meliputi *POS tagging*, *dependency parsing*, dan tokenisasi, yang diimplementasikan dengan alat seperti *Stanford CoreNLP* dan *Natural Language Toolkit* (NLTK) [4]. Pendekatan berbasis NLP dapat digunakan untuk memproses dokumen kebutuhan dan mengidentifikasi elemen penting seperti aktor, aksi, objek, serta kondisi yang berkaitan dengan perilaku sistem. Dengan memanfaatkan teknik NLP, proses pembentukan *test case* dapat dilakukan secara otomatis dari *user story [5]*. Selain itu, teknik pembelajaran mesin juga dapat dimanfaatkan untuk mengklasifikasikan dan memprioritaskan *test case* berdasarkan informasi yang diekstraksi dari *user story*, sehingga tim pengujian dapat memfokuskan pengujian pada skenario yang paling kritis terlebih dahulu [6].

Di sisi lain, pendekatan berbasis aturan (*rule-based system*) juga banyak digunakan dalam otomatisasi pengujian perangkat lunak karena mampu menghasilkan proses pengujian yang konsisten dan mudah ditelusuri. Metode berbasis aturan terbukti mampu mencapai akurasi hingga 95% untuk kebutuhan yang bersifat lebih sederhana dalam proses ekstraksi informasi dari dokumen kebutuhan [7]. Sementara itu, pendekatan berbasis *machine learning* memiliki kemampuan adaptasi yang lebih baik dalam menangani data yang kompleks, tetapi sering kali membutuhkan *dataset* yang besar serta memiliki tingkat interpretabilitas yang lebih rendah. Oleh karena itu, beberapa penelitian mulai mengusulkan pendekatan *hybrid* yang menggabungkan teknik *machine learning* dengan sistem berbasis aturan untuk meningkatkan efektivitas otomatisasi *QA* [7]. Pendekatan *hybrid* memungkinkan sistem memanfaatkan fleksibilitas pembelajaran mesin sekaligus mempertahankan konsistensi logika melalui aturan deterministik dalam proses pengujian perangkat lunak.

Meskipun berbagai penelitian telah mengkaji otomatisasi generasi *test case*, masih terdapat tiga lapisan keterbatasan mendasar yang belum terjawab. Pertama, pendekatan berbasis *NLP-only* rentan terhadap ambiguitas linguistik yang menghasilkan *test case* tidak konsisten. Kedua, pendekatan *rule-only* bersifat kaku dan tidak mampu menangani variasi struktur kalimat dalam bahasa alami. Ketiga, dan yang paling krusial, sebagian besar penelitian masih terbatas pada bahasa Inggris dan belum mengakomodasi karakteristik linguistik bahasa Indonesia. Sehingga diperlukan pendekatan *hybrid*

yang mampu menggabungkan fleksibilitas NLP dan ketegasan *rule-based* dalam konteks lokal. Selain itu, dari ketiga pendekatan tersebut, belum ada evaluasi komparatif sistematis dalam konteks *user story* berbahasa Indonesia, sehingga potensi pendekatan *hybrid* untuk bahasa lokal belum dapat dibuktikan secara empiris [8]. Kondisi ini berpotensi menyebabkan inefisiensi signifikan dalam proses *QA* pada industri perangkat lunak lokal, karena tim *QA* masih harus menyusun *test case* secara manual dari *user story* berbahasa Indonesia dengan hasil yang tidak konsisten dan tidak terstandarisasi.

Untuk menjawab kesenjangan tersebut, penelitian ini menawarkan pendekatan yang berbeda dibandingkan penelitian sebelumnya, yakni pengembangan Asisten *Quality Assurance* berbasis *hybrid NLP–rule-based* yang mampu menghasilkan *test case* secara otomatis dari *user story* berbahasa Indonesia. Dalam pendekatan ini, *user story* berbahasa Indonesia digunakan sebagai satu-satunya *input*, dengan *output* berupa *test case* terstruktur dalam format: *Test case ID*, *Scenario Name*, *Type*, *Precondition*, *Steps to Perform*, dan *Expected Result*. Selain itu, digunakan juga teknik *Named Entity Recognition* (NER) berbasis *IndoBERT* untuk mengekstraksi elemen penting seperti aktor, aksi, dan objek dari *user story*, sedangkan sistem berbasis aturan digunakan untuk membentuk skenario pengujian secara sistematis.

Adapun kontribusi utama penelitian ini adalah mengembangkan skema ekstraksi *user story* berbahasa Indonesia berbasis NLP; merancang *engine hybrid* yang menggabungkan NER dan *rule-based system*; menghasilkan *generator test case* otomatis berbasis templat terstruktur; serta melakukan evaluasi komparatif antara pendekatan *rule-only*, *NLP-only*, dan *hybrid* sebagai *ablation study* untuk membuktikan efektivitas pendekatan yang diusulkan.

2 Tinjauan Literatur

Upaya mengotomatisasi pembuatan *test case* berbasis bahasa alami telah menjadi isu sentral dalam rekayasa perangkat lunak modern. Zhao et al. [9] melalui *systematic mapping study* terhadap 404 studi NLP4RE menegaskan bahwa NLP berpotensi digunakan untuk menganalisis *requirement* dan menghasilkan *test case*. Namun, mayoritas solusi yang ada masih didominasi teknik leksikal dan sintaktik, dengan hanya sekitar tujuh persen yang telah divalidasi di lingkungan industri nyata. Kesenjangan antara riset dan praktik ini semakin diperkuat oleh Navarro dan Ibarra [10], yang melalui pemetaan sistematis terhadap 61 studi primer menunjukkan bahwa teknik NLP dominan masih sebatas *POS tagging*, *dependency parsing*, dan tokenisasi. Dari seluruh studi yang dikaji, hanya sembilan yang secara eksplisit menerapkan teknik desain *test case*. Temuan ini mempertegas bahwa integrasi antara pendekatan NLP dengan metodologi pengujian formal seperti *equivalence partitioning* masih sangat terbatas.

Beberapa penelitian empiris telah berupaya menjembatani celah tersebut dengan berbagai strategi. Fischbach et al. [11] mengembangkan pendekatan CiRA (*Conditionals in Requirements Artifacts*), yang menggunakan NLP untuk mengekstraksi pernyataan kondisional dari *requirement* informal, kemudian mengonversinya menjadi *acceptance test case* secara otomatis. Dalam studi kasus bersama tiga perusahaan industri, 71,8% dari 578 *test case* yang dibuat secara manual berhasil digenerate otomatis. Pendekatan ini membuktikan bahwa NLP dapat menjangkau *requirement* dalam bahasa natural yang tidak terstruktur, namun masih terbatas pada bahasa Inggris dan belum mengeksplorasi kombinasi dengan sistem *rule-based* untuk bahasa lain.

Gröpler et al. [12] melangkah lebih jauh dengan menggabungkan NLP berbasis spaCy dan aturan deterministik untuk memformalkan *requirement* teks menjadi model UML, kemudian menghasilkan *test case* dengan konsistensi ekstraksi entitas melampaui 95%. Pendekatan *hybrid* ini terbukti unggul dibandingkan *rule-only* maupun *NLP-only*, namun generalisasinya masih terbatas pada 14 *requirement* satu domain industri berbahasa Inggris. Fatima et al. [13] mengembangkan *pipeline end-to-end* berbasis BERT yang mengintegrasikan deteksi ambiguitas, validasi *requirement*, dan klasifikasi prioritas dengan akurasi 92–97%. Penelitian tersebut juga terbatas pada bahasa Inggris dan belum mengeksplorasi integrasi *Named Entity Recognition* (NER) dengan sistem *rule-based* deterministik untuk bahasa lain.

Penggunaan *user story* sebagai sumber masukan NLP untuk generasi *test case* juga telah dieksplorasi dari berbagai sudut. Chinnaswamy et al. [5] mengusulkan teknik tiga fase berbasis NLP yang mencakup kategorisasi masukan-keluaran melalui analisis sentimen, pembentukan ekspresi

 ISSN:2302-8149
e-ISSN:2540-9719

reguler, dan generasi *test case* dari *user story*. Pendekatan ini mengurangi keterlibatan manusia secara signifikan, namun belum divalidasi pada bahasa selain Inggris dan belum membandingkan performa antarpendekatan secara eksplisit. Prabu et al. [14] menggabungkan NLP, KeyBERT, *Naive Bayes*, dan *neural network* untuk mengotomatisasi klasifikasi komponen *Arrange-Act-Assert* dari *user story*. Hasilnya menunjukkan potensi besar *deep learning*, meskipun terdapat indikasi *overfitting* dan belum divalidasi lintas proyek. Gupta et al. [15] menawarkan perspektif berbeda dengan menggunakan *decision tree* berbasis *story board* dan NLP untuk menghasilkan *test case*; pendekatan ini menunjukkan pengurangan upaya yang signifikan dibanding metode konvensional, namun masih terbatas pada satu studi kasus tanpa eksplorasi bahasa non-Inggris.

Seiring berkembangnya *Large Language Model* (LLM), beberapa penelitian mulai mengeksplorasi perannya dalam generasi *test case*. Farisi dan Marva [16] mengevaluasi teknik *prompt engineering* pada Gemini 2.5 Pro untuk menghasilkan skenario *unit testing* pada *smart contract*. *Role-Based Prompting* mencapai *coverage* tertinggi sebesar 92,02%, membuktikan bahwa cara merumuskan instruksi ke model berpengaruh signifikan terhadap kualitas *test case*. Namun, penelitian ini hanya mencakup satu model LLM pada domain *blockchain* dan belum menyentuh pendekatan *hybrid* berbasis aturan. Dari sisi pengujian berbasis aturan, Hardika et al. [17] menerapkan *Equivalence Partitioning* dalam *Black Box Testing* dan menemukan tingkat validitas 56,25% dari 16 skenario uji. Studi ini relevan sebagai landasan teknik desain *test case* berbasis aturan yang dapat diintegrasikan ke dalam sistem otomatis, meskipun tidak mengintegrasikan NLP dan masih bersifat manual.

Dalam konteks NLP untuk bahasa Indonesia, ketersediaan model bahasa pra-latih yang kuat menjadi prasyarat penting. Koto et al. [18] memperkenalkan *IndoBERT*, model bahasa berbasis BERT yang dilatih menggunakan lebih dari 220 juta kata dari korpus Indonesia. Melalui *benchmark* IndoLEM yang mencakup tujuh tugas NLP, *IndoBERT* terbukti secara konsisten mengungguli model *multilingual* seperti mBERT dan XLM-RoBERTa pada tugas-tugas morfosintaksis, semantik, dan koherensi wacana, termasuk *task Named Entity Recognition* (NER) bahasa Indonesia. Sejalan dengan itu, Widyawan et al. [19] menunjukkan bahwa pendekatan *fusion* antara *IndoBERT* sebagai representasi fitur menggunakan arsitektur *deep learning* seperti BiLSTM, BiGRU, dan CRF mampu mencapai akurasi hingga 95,85% pada NER bahasa Indonesia. Hasil ini membuktikan bahwa *IndoBERT* dapat diadaptasi secara efektif untuk tugas ekstraksi entitas dalam teks berbahasa Indonesia, termasuk entitas spesifik domain seperti aktor, aksi, dan objek yang lazim ditemukan dalam *user story*.

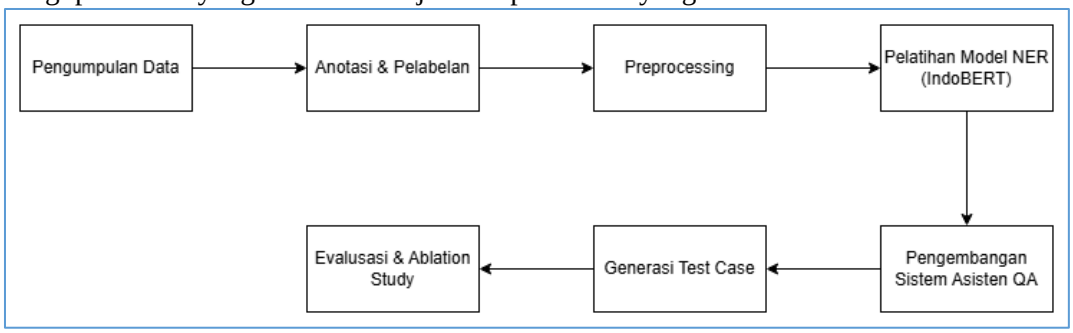
Berdasarkan kajian menyeluruh di atas, teridentifikasi tiga celah utama yang belum terjawab secara memadai. Pertama, meskipun *IndoBERT* telah terbukti unggul untuk NER bahasa Indonesia [18], [19] dan pendekatan *hybrid* NLP dengan *rule-based* telah terbukti efektif dalam generasi *test case* [12], belum ada penelitian yang secara sistematis mengombinasikan keduanya dalam satu kerangka yang dirancang khusus untuk memproses *user story* berbahasa Indonesia. Teknik NER berbasis *IndoBERT* digunakan untuk mengekstraksi elemen semantik dari *user story*, seperti aktor, aksi, dan objek—yang selanjutnya menjadi masukan bagi sistem *rule-based* deterministik dalam menghasilkan *test case*. Kedua, sebagian besar studi yang ada berfokus pada bahasa Inggris [5], [9], [10], [11], [12], [13], [14], [15], sehingga validasi terhadap konteks bahasa Indonesia, dengan karakteristik morfologi dan struktur kalimat yang berbeda—masih sangat terbatas. Ketiga, belum terdapat evaluasi komparatif sistematis yang membandingkan pendekatan *rule-only*, *NLP-only*, dan *hybrid* secara terstruktur dalam konteks *user story* berbahasa Indonesia. Evaluasi semacam ini bersifat krusial karena tanpa *ablation study* yang terdesain dengan baik, klaim keunggulan pendekatan *hybrid* tidak dapat diverifikasi secara ilmiah [12]. Penelitian ini mengisi ketiga celah tersebut sekaligus dengan mengembangkan asisten *QA* berbasis *hybrid* NLP dan *rule-based* yang dirancang khusus untuk memproses *user story* berbahasa Indonesia, serta mengevaluasi secara komparatif performa ketiga pendekatan tersebut secara terstruktur.

3 Metode Penelitian

Penelitian ini menerapkan *Design Science Research Methodology* (DSRM) sebagai kerangka utama yang dipadukan dengan pendekatan evaluasi eksperimental komparatif. DSRM dipilih karena penelitian ini berfokus pada perancangan dan pengembangan artefak berupa sistem Asisten *QA* guna

 ISSN:2302-8149
e-ISSN:2540-9719

menyelesaikan permasalahan nyata dalam proses penyusunan *test case*. Sementara itu, pendekatan eksperimental komparatif digunakan untuk menilai dan membandingkan efektivitas tiga konfigurasi metode secara kuantitatif berdasarkan metrik yang terukur. Gambar 1 menggambarkan alur metodologi penelitian yang terdiri dari tujuh tahapan utama yang dilaksanakan secara berurutan.



Gambar 1 Tahapan penelitian

3.1. Pengumpulan Data

Dataset yang digunakan dalam penelitian ini bersumber dari dua kumpulan *user story* berbahasa Indonesia, yaitu *dataset* dari sistem pinjaman (*Loan Origination System*) berbasis lingkungan perbankan yang berisi 270 *user story*, serta *dataset* FederalSpending yang berisi 292 *user story* dari domain manajemen pengeluaran pemerintah. Secara keseluruhan, *dataset* mencakup 562 *user story* yang telah dinormalisasi berdasarkan panduan struktur minimal, yaitu keberadaan elemen aktor, aksi, dan objek. Dataset dipecah menjadi 460 data latih dan 102 data validasi.

Setiap *user story* pada *dataset* ditulis dalam format alami berbahasa Indonesia dengan variasi struktur kalimat yang beragam, tidak terbatas pada pola tunggal. Seluruh *user story* dianotasi secara manual menggunakan skema pelabelan *Named Entity Recognition* (NER) berbasis format BIO (*Beginning-Inside-Outside*). Anotasi dilakukan terhadap lima kategori entitas fungsional yang relevan dengan komponen *user story*, sebagaimana diuraikan pada Tabel 1.

Tabel 1 Kategori label entitas NER

Label Entitas	Deskripsi	Contoh
B/I-ACTOR	Peran pengguna yang melakukan tindakan	admin , *Relationship Manager* (RM), *Team Lead* (TL), pengguna
B/I-ACTION	Aksi atau aktivitas yang dilakukan aktor	melihat, mengelola, mengunduh, mencari
B/I-OBJECT	Entitas yang menjadi target aksi	halaman overview, laporan, data,
B/I-EXPECTED_OUTCOME	Hasil atau tujuan yang diharapkan dari aksi	mengetahui status, memverifikasi data, berhasil login, berhasil menambah data
B/I-CONDITION	Kondisi prasyarat atau syarat tertentu	jika data tersedia, ketika login, jika user belum terdaftar, ketika user tidak aktif
O	Token yang tidak termasuk entitas manapun	saya, ingin, sehingga, dapat

3.2. *Preprocessing*

Sebelum memasuki tahap pelatihan model, setiap *user story* diproses dengan tahap *preprocessing* untuk meningkatkan konsistensi dan kualitas data masukan. Tahapan ini dijalankan secara otomatis oleh sistem dan mencakup beberapa proses berikut:

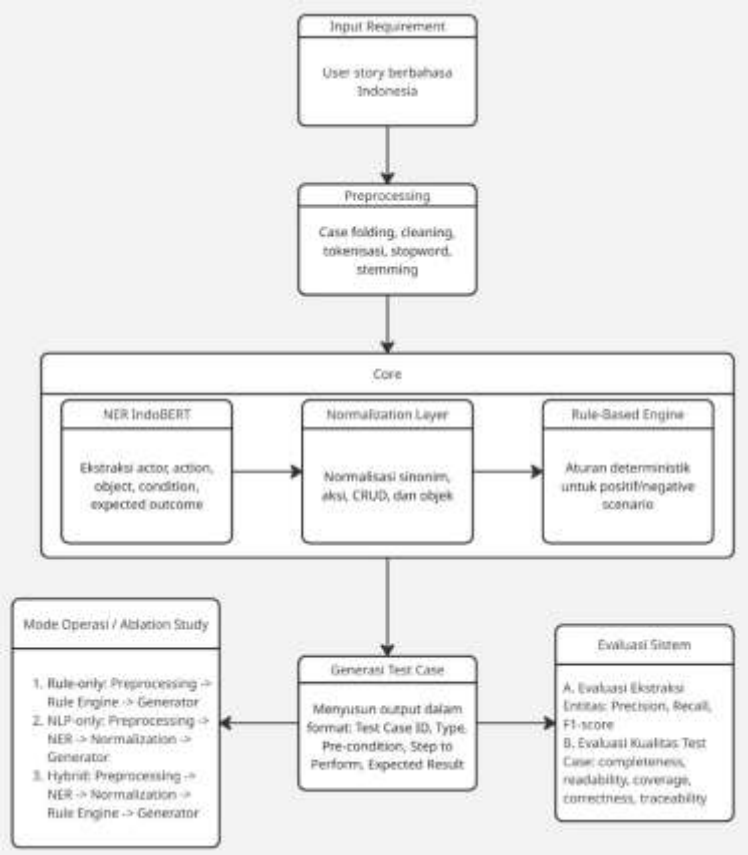
a. Case folding, adalah penyeragaman seluruh teks menjadi huruf kecil agar tidak terjadi perbedaan representasi akibat variasi penulisan huruf kapital.

 ISSN:2302-8149
e-ISSN:2540-9719

b. Pembersihan teks, yaitu penghapusan karakter-karakter yang tidak relevan, seperti tanda baca dan simbol khusus, serta normalisasi spasi berlebih untuk menghasilkan teks yang bersih dan seragam.
c. Tokenisasi, yaitu pemecahan teks menjadi satuan kata atau token sebagai dasar pemrosesan pada tahap selanjutnya.
d. Penghapusan *stopwords*, yaitu penyaringan kata-kata yang tidak membawa makna fungsional dalam konteks *user story*, seperti "sebagai", "saya", "ingin", dan "sehingga". Daftar *stopwords* disesuaikan secara khusus agar kata kunci struktural yang berulang tidak mengganggu proses ekstraksi entitas.
e. Stemming, adalah pengubahan kata berimbuhan menjadi kata dasar, sehingga variasi bentuk kata dapat dikenali dan dicocokkan secara konsisten pada tahap berikutnya.
f. Normalisasi aksi dan objek, yaitu pemetaan kata aksi hasil *stopwords* ke kategori CRUD standar melalui kamus aksi yang telah ditentukan. Objek yang diekstraksi juga dinormalisasi untuk menghilangkan karakter yang tidak diperlukan, sehingga representasi data menjadi seragam.

3.3. Arsitektur Sistem

Sistem Asisten *QA* yang dikembangkan dalam penelitian ini dirancang menggunakan arsitektur modular berbasis *pipeline* deterministik. Arsitektur ini mendukung tiga konfigurasi pemrosesan, yaitu *rule-only*, *NLP-only*, dan *hybrid*, yang digunakan sebagai dasar analisis perbandingan kinerja. Gambaran menyeluruh arsitektur sistem ditunjukkan pada Gambar 2.



Gambar 2 Arsitektur sistem asisten *QA*

Secara modular, sistem terdiri atas enam komponen utama:

1. *Requirement Ingestion Module*: Membaca dan memvalidasi struktur *user story* dari sumber masukan.
2. *Preprocessing* Module: Melakukan normalisasi awal dan tokenisasi teks.

 ISSN:2302-8149
e-ISSN:2540-9719

a. Information Extraction Module (NLP): Mengidentifikasi entitas penting menggunakan model NER berbasis *IndoBERT* dan memetakannya ke dalam slot fungsional (actor, action, object, condition, expected outcome).
b. *Normalization Layer*: Menyelaraskan istilah sinonim dan memastikan konsistensi representasi entitas antar *user story*.
c. *Rule-based Engine*: Mengimplementasikan aturan deterministik berbasis pola kebutuhan fungsional untuk membentuk skenario uji positif dan negatif beserta *precondition*, langkah uji, dan *expected result*.
d. *Test case Generator*: Menyusun seluruh komponen menjadi *template test case* klasik yang mencakup *Test case* ID, *Type* (Positive/Negative), *Pre-condition*, *Steps to Perform*, dan *Expected Result*.

Pada mode *rule-only*, modul NLP dilewati dan *rule engine* bekerja langsung menggunakan pola deterministik dari teks hasil *preprocessing*. Pada mode *NLP-only*, *rule engine* dibatasi pada pemetaan langsung hasil ekstraksi tanpa aturan tambahan. Sedangkan pada mode *hybrid*, kedua modul bekerja secara terintegrasi untuk membentuk skenario uji secara adaptif dan konsisten. Ketiga mode operasi ini juga berfungsi sebagai dasar *ablation study* untuk menganalisis kontribusi masing-masing komponen terhadap kualitas *test case* yang dihasilkan.

3.4. Model NER Berbasis *IndoBERT*

Komponen utama dalam modul ekstraksi informasi adalah model *Named Entity Recognition* (NER) yang dikembangkan menggunakan *fine-tuning* pada model *IndoBERT* versi *indobenchmark/indobert-base-p1*. *IndoBERT* dipilih karena telah dilatih menggunakan lebih dari 4 miliar token berbahasa Indonesia sehingga memiliki kemampuan representasi kontekstual yang kuat untuk teks berbahasa Indonesia[18].

Arsitektur model NER terdiri atas *IndoBERT* sebagai *feature extractor* yang menghasilkan representasi kontekstual per token, dilanjutkan dengan *token classification head* berupa lapisan *fully connected* dengan fungsi aktivasi *softmax* untuk mengklasifikasikan setiap token ke dalam salah satu dari 11 label entitas (B-ACTOR, I-ACTOR, B-ACTION, I-ACTION, B-OBJECT, I-OBJECT, B-EXPECTED_OUTCOME, I-EXPECTED_OUTCOME, B-CONDITION, I-CONDITION, O). Konfigurasi *hyperparameter* yang digunakan dalam proses pelatihan disajikan pada Tabel 2.

Tabel 2 Konfigurasi hyperparameter model NER

Hyperparameter	Nilai
Model Dasar	indobenchmark/indobert-base-p1
Learning Rate	2e-5
Batch Size	8
Max Sequence Length	128 token
Jumlah Epoch	20 (dengan early stopping, patience=5)
Dropout	0.1
Optimizer	AdamW
Skema Label	BIO (Beginning-Inside-Outside)

Proses *fine-tuning* dilakukan menggunakan data *train.json* yang berisi 460 sampel *user story* teranotasi. Setiap sampel terdiri atas pasangan token dan label dalam format BIO, di mana token yang merupakan awalan suatu entitas diberi prefiks "B-" dan token lanjutan entitas diberi prefiks "I-". Token yang tidak termasuk entitas diberi label "O". Validasi performa model dilakukan menggunakan *val.json* yang berisi 102 sampel secara terpisah dari data latih untuk menghindari kebocoran data (*data leakage*).

3.5. Evaluasi Sistem

Evaluasi sistem dilakukan menggunakan dua tingkatan pengukuran. Pertama, evaluasi model NER menggunakan metrik *precision*, *recall*, dan *F1-score* per entitas serta makro rata-rata, dihitung berdasarkan data validasi. Kedua, evaluasi kualitas *test case* yang dihasilkan mencakup tiga aspek utama, yaitu akurasi ekstraksi elemen *user story*, kelengkapan

 ISSN:2302-8149
e-ISSN:2540-9719

(*completeness*) skenario uji, dan kesesuaian struktur (*correctness*) terhadap format *template* klasik.

Sebagai *ablation study*, evaluasi komparatif dilakukan terhadap tiga konfigurasi sistem: *rule-only*, *NLP-only*, dan *hybrid*. Perbandingan ini bertujuan untuk membuktikan secara empiris bahwa pendekatan *hybrid* memberikan performa yang lebih unggul dibandingkan dua pendekatan lainnya dalam konteks *user story* berbahasa Indonesia.

3.5.1. Evaluasi Kualitas *Test Case* oleh Reviewer *QA*

Selain evaluasi kuantitatif berbasis metrik otomatis, penelitian ini juga melibatkan evaluasi kualitatif oleh reviewer berlatar belakang *QA Engine*er dari kalangan praktisi industri pengembangan perangkat lunak. Evaluasi ini bertujuan untuk mengukur kualitas *test case* yang dihasilkan sistem dari perspektif praktisi, yakni sejauh mana *test case* tersebut dapat digunakan secara langsung dalam proses pengujian perangkat lunak nyata.

Reviewer yang dilibatkan merupakan praktisi aktif di bidang *Quality Assurance* dengan pengalaman dalam penyusunan dan eksekusi *test case* pada proyek perangkat lunak berbasis *Agile*. Penilaian dilakukan terhadap sampel *test case* yang dihasilkan oleh ketiga konfigurasi sistem, yaitu *rule-only*, *NLP-only*, dan *hybrid*, menggunakan instrumen rubrik penilaian terstruktur dengan rentang nilai 1–5. Interpretasi skor mengacu pada kategori berikut: 1 (sangat kurang, tidak memenuhi kriteria sama sekali), 2 (kurang, memenuhi sebagian kecil kriteria), 3 (cukup baik, memenuhi sebagian besar kriteria), 4 (baik, memenuhi kriteria dengan baik), dan 5 (sangat baik, memenuhi semua kriteria secara sempurna). Hasil penilaian reviewer selanjutnya dianalisis secara deskriptif dan dibandingkan antarkonfigurasi sistem untuk melengkapi temuan evaluasi kuantitatif. Rubrik penilaian mencakup lima aspek sebagaimana disajikan pada Tabel 3.

Tabel 3 Rubrik penilaian kualitas *test case* oleh reviewer *QA*

Aspek Penilaian	Deskripsi	Skala Penilaian (1–5)
Kelengkapan *(Completeness)*	Seluruh elemen *test case* (*Test case* ID, *Type*, *Pre-condition*, *Steps to Perform*, *Expected Result*) terisi dengan benar dan tidak ada komponen yang kosong.	1 = Hampir semua elemen kosong, 2 = Beberapa elemen terisi, tapi masih banyak yang kosong, 3 = Sebagian elemen terisi, 4 = Sebagian besar elemen terisi dengan baik, 5 = Semua elemen terisi lengkap dan tepat
Keterbacaan *(Readability)*	Langkah uji dan *expected result* ditulis dengan bahasa yang jelas, runtut, dan mudah dipahami oleh tester tanpa memerlukan penjelasan tambahan.	1 = Tidak dapat dipahami, 2 = Sulit dipahami, butuh banyak usaha, 3 = Dapat dipahami dengan usaha, 4 = Cukup jelas, hanya perlu sedikit interpretasi, 5 = Sangat jelas tanpa interpretasi tambahan
Cakupan Skenario *(Coverage)*	*Test case* mencakup skenario positif (alur normal) dan negatif (kondisi batas atau data tidak valid) yang relevan terhadap *user story* sumbernya.	1 = Hanya satu skenario (positif atau negatif saja), 2 = Ada skenario positif dan negatif tapi sangat terbatas, 3 = Ada positif dan negatif namun tidak lengkap, 4 = Positif dan negatif terwakili dengan baik, sedikit kurang, 5 = Skenario positif dan negatif terwakili secara komprehensif
Kebenaran Format *(Correctness)*	Struktur *test case* sesuai dengan format *template* klasik yang ditetapkan, termasuk urutan komponen, penamaan field, dan konsistensi kategori (Positive/Negative).	1 = Format tidak sesuai sama sekali, 2 = Hanya sedikit bagian yang sesuai format, 3 = Sebagian besar sesuai format *template*, 4 = Hampir sepenuhnya sesuai, ada ketidakkonsistenan kecil, 5 = Format sepenuhnya sesuai standar *template*

 ISSN:2302-8149
e-ISSN:2540-9719

Aspek Penilaian	Deskripsi	Skala Penilaian (1–5)
Keterlacakan *(Traceability)*	*Test case* dapat ditelusuri kembali ke *user story* asal melalui *Test case* ID, dan isi *test case* secara logis mencerminkan kebutuhan yang dinyatakan dalam *user story* sumber.	1 = Tidak ada keterkaitan sama sekali dengan *user story* asal, 2 = Keterkaitan dengan *user story* sangat lemah, 3 = Sebagian isi *test case* mencerminkan *user story*, tapi tidak konsisten, 4 = Sebagian besar terlacak dan konsisten dengan *user story*, 5 = Sepenuhnya terlacak dan konsisten dengan *user story* asal

4 Hasil dan Pembahasan

4.1. Evaluasi Model NER

Evaluasi awal dilakukan untuk mengukur kinerja model *Named Entity Recognition* (NER) dalam mengekstraksi elemen-elemen penting dari *user story*. Pengukuran efektivitas model ini dilakukan dengan menggunakan metrik *precision*, *recall*, dan *F1-score* pada data pengujian.

Tabel 4 Hasil evaluasi model NER per Entitas

Entitas	*Precision*	*Recall*	F1-Score
ACTOR	0.9806	1.0000	0.9902
ACTION	0.9783	1.0000	0.9890
OBJECT	0.8454	0.9111	0.8770
CONDITION	0.8235	0.9032	0.8615
EXPECTED_OUTCOME	0.9032	0.9545	0.9282
Macro Average	**0.9062**	**0.9538**	**0.9292**

Berdasarkan evaluasi (Tabel 4), model menunjukkan kinerja sangat baik dengan Macro F1-score sebesar 0,929. Entitas ACTOR dan ACTION memiliki performa hampir sempurna (F1 > 0,98), menandakan pola penulisan subjek dan aksi dalam *user story* cukup konsisten dan terstruktur. Sebaliknya, entitas CONDITION mencatat performa terendah (*precision* 0,8235; *F1-score* 0,8615) akibat variasi bahasa yang tinggi, struktur klausa yang lebih kompleks, serta batasan teks yang cenderung ambigu. Entitas OBJECT juga mencatatkan performa yang lebih rendah dibandingkan ACTOR dan ACTION yaitu *precision* 0,8454 dan F1-score 0,8770, yang disebabkan oleh variasi panjang span dan ambiguitas batas entitas, terutama ketika OBJECT berdampingan langsung dengan frasa CONDITION dalam kalimat yang sama. Secara keseluruhan, model andal dalam mengekstraksi komponen inti yang baku, namun masih menghadapi tantangan pada bagian yang lebih bebas dan tidak terstruktur, khususnya pada kondisi dan objek.

4.2. Evaluasi Komparatif Sistem (*Ablation study*)

Evaluasi dilakukan dengan membandingkan tiga pendekatan utama, yaitu *rule-only*, *NLP-only*, dan *hybrid* (kombinasi NLP dan *rule-based*). Perbandingan ini bertujuan untuk melihat perbedaan performa masing-masing pendekatan dalam menghasilkan *test case* yang akurat dan relevan. Metrik *precision*, *recall*, dan *F1-score* pada Tabel 5 dihitung berdasarkan kecocokan komponen *test case* yang dihasilkan oleh sistem terhadap *ground truth test case* yang disusun dari *user story* referensi. Evaluasi dilakukan pada level elemen fungsional dan struktural *test case*, meliputi *precondition*, *steps to perform*, dan *expected result*. *Precision* menunjukkan proporsi komponen *test case* hasil sistem yang benar terhadap seluruh komponen yang dihasilkan sistem. *Recall* menunjukkan proporsi komponen *ground truth* yang berhasil dihasilkan kembali oleh sistem. *F1-score* digunakan sebagai ukuran keseimbangan antara *precision* dan *recall*.

 ISSN:2302-8149
e-ISSN:2540-9719

Tabel 5 Perbandingan kinerja sistem

Metode	*Precision*	*Recall*	F1-Score
Rule-only	0.87	0.72	0.79
NLP-only	0.90	0.80	0.85
Hybrid	**0.94**	**0.88**	**0.91**

Berdasarkan hasil evaluasi, model *hybrid* menunjukkan performa terbaik dengan *precision* 0.94, *recall* 0.88, dan *F1-score* 0.91. Hal ini menunjukkan model tidak hanya akurat, tetapi juga mampu menangkap sebagian besar kasus yang relevan, sehingga memiliki keseimbangan yang optimal. Model NLP memiliki performa cukup baik (*F1-score* 0.85) karena fleksibel dalam memahami variasi bahasa. Namun, fleksibilitas ini kadang menyebabkan ketidakkonsistenan struktur atau kurang tepatnya konteks. Sementara itu, model *rule-based* memiliki F1-score terendah (0.79). Model ini kuat pada struktur dan konsistensi, tetapi kurang mampu menangani variasi kalimat sehingga banyak kasus tidak terdeteksi.

Secara umum, *rule-based* unggul pada struktur namun lemah pada fleksibilitas, sedangkan NLP fleksibel tetapi kurang konsisten. *hybrid* menggabungkan kelebihan keduanya dan menghasilkan performa terbaik, sehingga dapat disimpulkan pendekatan ini lebih unggul dibandingkan metode tunggal.

4.3. Evaluasi Kualitas *Test Case* oleh Reviewer *QA*

Evaluasi kualitatif ini dilakukan oleh praktisi *Quality Assurance* (*QA*) untuk menilai kualitas *test case* yang dihasilkan oleh sistem. Penilaian menggunakan skala Likert (1–5) yang mencakup lima aspek utama: *Completeness*, *Readability*, *Coverage*, *Correctness*, dan *Traceability*.

Tabel 6 Hasil evaluasi kualitatif *test case*

Aspek	*NLP-only*	*Rule-only*	*Hybrid*
Completeness	3.67	4.33	4.67
Readability	4.00	4.67	4.67
Coverage	3.67	4.67	4.67
Correctness	4.00	4.67	4.67
Traceability	4.00	4.67	4.67
Rata-Rata	**3.87**	**4.60**	**4.67**

Berdasarkan evaluasi QA (Tabel 6), pendekatan *hybrid* menjadi yang terbaik dengan rata-rata skor 4,67, unggul tipis dari *rule-only* (4,60) dan jauh di atas *NLP-only* (3,87). *Hybrid* konsisten mendapat nilai tinggi di semua aspek, menunjukkan kualitas *test case* yang paling memenuhi standar. Pada level aspek, *hybrid* dan *rule-only* memiliki performa identik pada *Readability, Coverage, Correctness, dan Traceability* (4,67), menegaskan pentingnya *rule-based* dalam menjaga struktur dan keterlacakan. Keunggulan *hybrid* terlihat pada *Completeness*, sementara *NLP-only* lemah pada *Completeness* dan *Coverage* (3,67) karena sering kehilangan detail konteks. Secara keseluruhan, *rule-based* membangun struktur yang rapi, NLP memperkaya pemahaman konteks, dan kombinasi keduanya dalam *hybrid* menghasilkan solusi paling optimal untuk otomatisasi *test case*.

4.4. Evaluasi Efisiensi Waktu

Evaluasi efisiensi (Tabel 7) waktu dilakukan untuk membandingkan durasi penyusunan *test case* secara manual dengan durasi generasi *test case* menggunakan sistem *hybrid*. Evaluasi ini melibatkan 3 responden yang berprofesi sebagai QA dan *Software Tester* dengan pengalaman kerja yang bervariasi: dua responden memiliki pengalaman 3–5 tahun dan satu responden memiliki pengalaman lebih dari 5 tahun. Setiap responden diminta memperkirakan rata-rata waktu yang dibutuhkan untuk menyusun satu *test case* secara manual. Hasil estimasi menunjukkan variasi waktu antara 2 hingga 10 menit dengan rata-rata 5 menit per *test case*, bergantung pada kompleksitas *user story* dan pengalaman individu. Sementara itu, sistem mampu

 ISSN:2302-8149
e-ISSN:2540-9719

menghasilkan *test case* dalam hitungan detik, sehingga peningkatan efisiensi waktu mencapai lebih dari 99%.

Tabel 7 Perbandingan efisiensi waktu

Metode Manual	***Hybrid* Sistem**
2–10 menit / *test case* (Rata-rata: 5 menit)	0.1130 detik / *test case*

Sistem menunjukkan peningkatan efisiensi yang sangat signifikan dengan memangkas waktu pengerjaan dari rata-rata 5 menit (300 detik) menjadi hanya 0,113 detik, atau lebih dari 99%. Selain itu, sistem mampu menghasilkan *output* secara konsisten dalam waktu di bawah satu detik, tanpa terpengaruh faktor kelelahan seperti pada proses manual yang bisa mencapai 10 menit. Tingkat efisiensi ini menjadikannya sangat relevan untuk industri, khususnya dalam metodologi *Agile* yang menuntut pembuatan *test case* secara cepat dan adaptif terhadap perubahan kebutuhan.

4.5. Contoh Hasil Generasi *Test Case*

Untuk memberikan gambaran konkret mengenai kinerja sistem yang dikembangkan, bagian ini menyajikan contoh hasil generasi *test case* menggunakan pendekatan *hybrid*. Contoh ini diambil dari salah satu *user story* berbahasa Indonesia yang merepresentasikan kebutuhan fungsional dalam sistem.

***User story*:**

Sebagai Admin, saya ingin menghapus data pengeluaran yang tidak relevan, sehingga data pengeluaran tetap akurat dan terkini.

Berdasarkan *user story* tersebut, sistem *hybrid* menghasilkan dua skenario pengujian, yaitu skenario positif dan negatif. Hasil generasi *test case* ditunjukkan pada Tabel 8.

Tabel 8 Hasil generasi *hybrid*

Komponen	Hasil
Test case ID	TC-DELETE-DATA-PENGELUARAN
Scenario Name	Admin berhasil menghapus data pengeluaran yang tidak relevan
Type	Positive
Pre-condition	Admin telah login ke dalam sistem
Steps to Perform	1. Admin membuka halaman data pengeluaran yang tidak relevan 2. Admin memilih data yang akan dihapus 3. Admin mengkonfirmasi penghapusan data pengeluaran yang tidak relevan 4. Sistem menghapus data pengeluaran yang tidak relevan
Expected Result	Sistem berhasil menghapus data pengeluaran sehingga data tetap akurat dan terkini
Test case ID	TC-DELETE-DATA-PENGELUARAN-N1
Scenario Name	Membatalkan penghapusan data pengeluaran yang tidak relevan
Type	Negative
Pre-condition	Admin telah login ke dalam sistem
Steps to Perform	1. Admin membuka halaman data pengeluaran yang tidak relevan 2. Admin memilih data yang akan dihapus 3. Admin menekan tombol hapus 4. Admin menekan tombol batal pada dialog konfirmasi
Expected Result	Data pengeluaran tidak terhapus dan tetap tersedia dalam sistem

Hasil pada Tabel 8 menunjukkan bahwa sistem mampu menghasilkan *test case* dengan struktur yang lengkap dan sistematis, mencakup elemen penting seperti identitas *test case*, skenario, kondisi awal, langkah pengujian, serta hasil yang diharapkan. Selain itu, sistem juga mampu membedakan skenario positif dan negatif secara otomatis berdasarkan konteks *user story*. Hal ini menunjukkan bahwa pendekatan *hybrid* tidak hanya mampu memahami kebutuhan

 ISSN:2302-8149
e-ISSN:2540-9719

fungsional, tetapi juga menerjemahkannya menjadi skenario pengujian yang siap digunakan oleh tim *QA*.

4.6. Analisis Error

Pengujian terhadap ketiga pendekatan mengungkap pola kesalahan yang khas pada masing-masing metode. Kesalahan paling dominan secara keseluruhan terjadi pada entitas CONDITION dan OBJECT, yang memiliki variasi linguistik lebih tinggi dibandingkan entitas lain seperti ACTOR atau ACTION. Frasa kondisi sering kali ditulis secara implisit atau menggunakan struktur kalimat yang tidak baku, misalnya "jika tersedia", "ketika akun tidak aktif", atau "setelah verifikasi selesai", sehingga model mengalami kesulitan dalam menentukan batas entitas secara presisi. Hal ini sejalan dengan temuan evaluasi bahwa recall untuk CONDITION merupakan yang terendah di antara seluruh kategori entitas. Sementara itu, entitas OBJECT turut menjadi sumber kesalahan yang signifikan karena span-nya sangat bervariasi, mulai dari satu kata seperti "laporan" hingga frasa panjang seperti "laporan keuangan bulanan per divisi", sehingga model kerap kesulitan menentukan batas akhir entitas secara tepat. Kesulitan ini diperparah oleh adanya multi-aksi dalam satu kalimat, di mana satu OBJECT diasosiasikan dengan lebih dari satu ACTION sekaligus, serta oleh ambiguitas batas antara OBJECT dan CONDITION yang berdampingan dalam struktur kalimat yang kompleks.

Selain itu, struktur *user story* dalam data nyata tidak selalu mengikuti pola standar "Sebagai [aktor], saya ingin [aksi] [objek], agar [hasil]." Banyak *user story* mengandung multi-aksi dalam satu kalimat, *expected outcome* implisit, kondisi yang terpisah, atau struktur bebas seperti "Admin dapat melihat dan mengunduh laporan jika data tersedia." Variasi semacam ini meningkatkan kompleksitas ekstraksi informasi dan dapat menurunkan konsistensi prediksi model. Faktor lain yang turut berkontribusi adalah ukuran *dataset* yang masih relatif terbatas, yang berpotensi membatasi kemampuan generalisasi model terhadap variasi *user story* yang lebih luas. Selain itu, *preprocessing* yang terlalu agresif, seperti penghapusan *stopword* struktural ("jika", "agar", "ketika") atau *stemming* berlebihan, berpotensi menghilangkan penanda linguistik penting yang justru diperlukan dalam tugas NER.

Analisis error dilakukan terhadap 100 *test case* hasil generasi dari masing-masing pendekatan, yaitu *NLP-only*, *rule-only*, dan *hybrid*. Pada pendekatan *NLP-only*, ditemukan bahwa 50 dari 100 *test case* bertipe positif belum memiliki penomoran langkah yang konsisten. Selain itu, 24 dari 100 *test case* mengandung pengulangan kata atau frasa akibat tokenisasi yang tidak sempurna, seperti "(harian, bulanan, bulanan, tahunan)" atau frasa yang muncul hingga beberapa kali berturut-turut dalam satu kolom. Pendekatan *rule-only* menghasilkan error yang lebih sistematis: 49 dari 100 nama skenario diakhiri tanda koma yang tidak diperlukan akibat pola *template* yang terlalu kaku, dan 19 dari 100 *test case* memiliki *expected result* yang terlalu generik, seperti "sesuai *scope role*", tanpa kondisi verifikasi yang konkret.

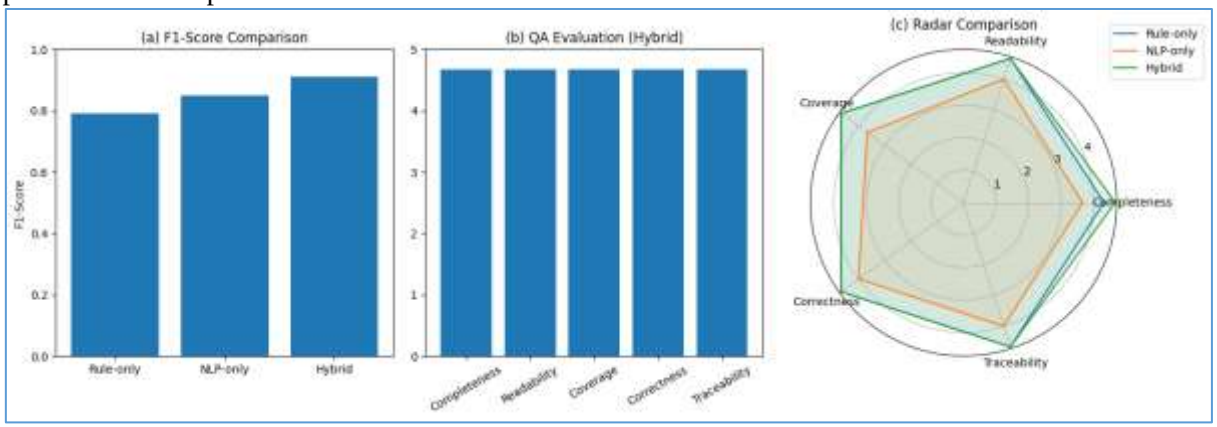
Pendekatan *hybrid* berhasil mengeliminasi hampir seluruh error struktural tersebut, dengan penomoran langkah yang lebih konsisten dan *output* yang lebih bebas dari artefak duplikasi. Meski demikian, masih ditemukan satu langkah dengan aksi ganda serta empat dari 100 *expected result* yang masih bersifat generik — jauh lebih sedikit dibandingkan pendekatan murni *rule-only*. Secara keseluruhan, temuan ini menunjukkan bahwa integrasi NLP dan *rule-based* mampu menekan kesalahan masing-masing pendekatan tunggal, meskipun kompleksitas linguistik *user story* berbahasa Indonesia tetap menjadi tantangan utama yang menggarisbawahi pentingnya perluasan *dataset*, peningkatan konsistensi anotasi, optimasi *hyperparameter*, dan penyempurnaan aturan *post-processing*.

4.7. Diskusi

Hasil penelitian menunjukkan bahwa pendekatan *hybrid* meningkatkan akurasi ekstraksi, kualitas *test case*, dan efisiensi *QA* melalui NLP dan *rule-based* system yang mampu menggabungkan fleksibilitas pemahaman konteks dan konsistensi struktur. Dibandingkan penelitian sebelumnya, pendekatan ini memperkuat temuan Gröpler et al. [12] terkait efektivitas pendekatan *hybrid* serta melengkapi keterbatasan pendekatan *NLP-only* yang dikemukakan oleh Fatima et al. [13], khususnya dalam menjaga konsistensi struktur. Selain itu, penelitian ini memperluas penerapan pada *user story* berbahasa Indonesia melalui evaluasi komparatif yang

 ISSN:2302-8149
e-ISSN:2540-9719

sistematis. Dari sisi kualitas, hasil penilaian praktisi *QA* menunjukkan bahwa metode *hybrid* unggul secara konsisten dengan menghasilkan *test case* yang lebih lengkap, jelas, dan terlacak. Sebagaimana ditunjukkan pada Gambar 3, pendekatan *hybrid* secara konsisten memberikan performa terbaik pada seluruh metrik evaluasi.



Gambar 3. Perbandingan kinerja metode *rule-only*, *NLP-only*, dan *hybrid*

Panel (a) menunjukkan peningkatan signifikan pada F1-score, panel (b) menunjukkan kualitas *test case* yang tinggi, dan panel (c) menegaskan keunggulan metode *hybrid* pada seluruh aspek evaluasi, sehingga membuktikan efektivitas integrasi NLP dan *rule-based* dalam otomatisasi *QA*; dari sisi efisiensi, waktu pengerjaan berkurang dari beberapa menit menjadi kurang dari satu detik sehingga mendukung kebutuhan pengembangan *Agile*, dan secara keseluruhan penelitian ini memvalidasi pendekatan *hybrid* pada bahasa Indonesia melalui integrasi NER berbasis *IndoBERT* dan *rule-based* system serta membuktikan efektivitasnya dalam meningkatkan kualitas dan efisiensi proses QA.

4.8. Threats to Validity

Beberapa keterbatasan penelitian perlu diperhatikan dalam menafsirkan hasil yang diperoleh. Pertama, dataset yang digunakan mencakup 562 *user story* dari dua domain spesifik, yaitu sistem pinjaman perbankan (270 *user story*) dan manajemen pengeluaran pemerintah (292 *user story*). Meskipun kedua domain memberikan variasi kebutuhan fungsional, cakupan ini belum sepenuhnya mewakili seluruh jenis *user story* pada berbagai sektor industri perangkat lunak seperti kesehatan, pendidikan, maupun *e-commerce*, sehingga kemampuan generalisasi model lintas domain masih perlu diuji lebih lanjut.

Kedua, anotasi entitas dilakukan secara manual menggunakan skema BIO, sehingga potensi subjektivitas *annotator* tetap ada. Tanpa pengukuran formal seperti *inter-annotator agreement* (misalnya *Cohen's Kappa*), konsistensi anotasi belum dapat diverifikasi secara kuantitatif sepenuhnya. Hal ini terutama berlaku pada entitas CONDITION dan EXPECTED_OUTCOME yang memiliki batas frasa lebih fleksibel dibandingkan entitas lainnya.

Ketiga, ukuran dataset yang masih relatif terbatas berpotensi menyebabkan *overfitting* terhadap pola dominan dan kurang *robust* terhadap variasi struktur baru. Distribusi entitas dalam dataset pun tidak sepenuhnya seimbang: entitas ACTOR dan ACTION cenderung lebih sering muncul dan lebih konsisten strukturnya, sementara CONDITION memiliki frekuensi lebih rendah dengan variasi bahasa yang lebih kompleks, dan OBJECT meskipun relatif sering muncul memiliki variasi panjang *span* yang tinggi dengan batas entitas yang kerap ambigu, sehingga kedua faktor ini dapat memengaruhi stabilitas performa model.

Keempat, evaluasi kualitatif melibatkan sejumlah terbatas *reviewer* berlatar belakang *QA Engineer*, sehingga representativitas perspektif praktisi masih terbatas. Penilaian menggunakan rubrik skala *Likert* juga berpotensi mengandung subjektivitas antar-*reviewer*, meskipun deskriptor telah didefinisikan secara eksplisit. Di sisi teknis, evaluasi NER menggunakan *exact-match* F1-*score* yang cenderung ketat terhadap perbedaan batas token, sehingga prediksi yang secara semantik cukup benar tetap dianggap salah apabila *span* token tidak identik dengan *ground truth*.

 ISSN:2302-8149
e-ISSN:2540-9719

Meskipun terdapat keterbatasan tersebut, kombinasi evaluasi kuantitatif, *ablation study*, dan validasi praktisi memberikan dasar yang cukup kuat bahwa sistem *hybrid* yang dikembangkan memiliki manfaat praktis dan ilmiah yang signifikan. Untuk penelitian selanjutnya, disarankan agar dataset diperluas ke domain yang lebih beragam, melibatkan *reviewer* QA yang lebih banyak dari berbagai jenis industri, serta mengeksplorasi integrasi sistem dengan *tools* QA seperti Jira untuk memvalidasi sistem di lingkungan pengembangan perangkat lunak yang nyata.

5 Future Work

Penelitian ini membuka sejumlah peluang pengembangan lanjutan. Pertama, perluasan *dataset* menjadi prioritas utama dengan menambahkan *user story* lintas domain seperti *e-commerce*, kesehatan, pendidikan, dan layanan publik guna meningkatkan kemampuan generalisasi model. Strategi yang dapat diterapkan mencakup *paraphrase augmentation*, generasi *user story* sintetis menggunakan LLM, *active learning* berbasis sampel yang sulit diprediksi, *oversampling* label minoritas CONDITION dan EXPECTED_OUTCOME, serta *k-fold cross-validation* untuk meningkatkan stabilitas evaluasi pada *dataset* yang relatif kecil.

Kedua, pengembangan *annotation guideline* yang lebih formal disertai pengukuran *inter-annotator agreement* menggunakan metrik Cohen's Kappa atau Fleiss' Kappa perlu dilakukan untuk memperkuat validitas metodologis. Hal ini penting karena perbedaan interpretasi antar anotator berpotensi terjadi, khususnya pada entitas yang bersifat kompleks seperti CONDITION dan EXPECTED_OUTCOME. Nilai $\kappa > 0{,}80$ akan mengindikasikan konsistensi anotasi yang sangat baik dan memperkuat kredibilitas *ground truth* yang digunakan dalam pelatihan model.

Ketiga, optimasi arsitektur model dapat dieksplorasi melalui penambahan lapisan *Conditional Random Field* (CRF), penerapan *focal loss* atau *class-weighted loss*, guna memperbaiki performa pada entitas minoritas. Keempat, adopsi *partial-match evaluation* dapat memberikan gambaran yang lebih realistis mengenai kegunaan praktis sistem dibandingkan *exact-match* semata. Kelima, integrasi LLM seperti GPT atau Gemini sebagai komponen augmentatif dapat dieksplorasi untuk meningkatkan kemampuan interpretasi *user story* yang kompleks, dengan tetap mempertahankan *rule-based system* sebagai kontrol struktural. Keenam, integrasi sistem dengan platform industri seperti Jira, Zephyr, TestRail, atau *pipeline* otomasi Selenium akan meningkatkan relevansi implementatif di lingkungan pengembangan perangkat lunak nyata. Dengan arah pengembangan tersebut, sistem *hybrid* ini berpotensi berkembang dari prototipe akademik menjadi solusi *QA* otomatis yang lebih *robust*, adaptif, dan siap digunakan secara industri.

6 Kesimpulan

Penelitian ini berhasil merancang dan mengembangkan asisten *Quality Assurance* berbasis pendekatan *hybrid* yang mengintegrasikan *Natural Language Processing* dan *rule-based system* untuk menghasilkan *test case* secara otomatis dari *user story* berbahasa Indonesia. Hasil evaluasi menunjukkan bahwa pendekatan *hybrid* memberikan performa terbaik dibandingkan *rule-only* dan *NLP-only*, baik dari sisi akurasi ekstraksi dengan F1-*score* 0,91, kualitas *test case* dengan skor rata-rata 4,67, maupun efisiensi waktu yang meningkat lebih dari 99%. Dengan demikian, tantangan utama yang dikemukakan pada bagian pendahuluan, yaitu penyusunan *test case* manual yang memakan waktu dan menghasilkan kualitas tidak konsisten, dapat diatasi melalui sistem otomatis yang lebih cepat, terstruktur, dan mudah ditelusuri. Secara teoretis, penelitian ini memberikan kontribusi pada pengembangan kajian otomatisasi QA berbasis NLP untuk bahasa Indonesia melalui integrasi NER berbasis IndoBERT dan *rule-based system* dalam satu kerangka *rule-based*. Secara praktis, sistem yang dikembangkan dapat membantu tim QA mengurangi beban kerja manual, meningkatkan konsistensi struktur *test case*, dan mempercepat proses pengujian pada lingkungan *Agile*. Meskipun demikian, penelitian ini masih memiliki keterbatasan pada cakupan domain *dataset*, jumlah data teranotasi untuk pelatihan NER, serta jumlah *reviewer* QA yang terlibat dalam evaluasi kualitatif. Penelitian selanjutnya disarankan untuk memperluas *dataset* lintas domain, meningkatkan kualitas anotasi melalui *inter-annotator agreement*, menerapkan *data augmentation*, serta mengintegrasikan sistem dengan platform pengujian seperti Jira, Zephyr, atau TestRail.

 ISSN:2302-8149
e-ISSN:2540-9719

Referensi

[1] A. Mustafa *et al.*, "*Automated Test Case Generation from Requirements: A Systematic Literature Review*," *Computers, Materials and Continua*, Vol. 67, No. 2, pp. 1819–1833, Jan. 2021, DOI: 10.32604/CMC.2021.014391.

[2] V. Garousi, S. Bauer, and M. Felderer, "*NLP-Assisted Software Testing: A Systematic Mapping of the literature*," *Inf. Softw. Technol.*, Vol. 126, p. 106321, Oct. 2020, DOI: 10.1016/J.INFSOF.2020.106321.

[3] H. Imhmed, K. Ahmed, Y. Salem, and H. Zulzalil, "*Leveraging Latent Natural Language Processing Techniques for User Story Management in Agile Software Development*," *Journal of Pure & Applied Sciences*, Vol. 22, No. 2, pp. 5–9, Oct. 2023, DOI: 10.51984/jopas.v22i2.2599.

[4] J. W. Lim *et al.*, "*Test Case Information Extraction from Requirements Specifications using NLP-based Unified Boilerplate Approa*ch," *Journal of Systems and Software*, Vol. 211, p. 112005, May 2024, doi: 10.1016/J.JSS.2024.112005.

[5] A. Chinnnaswamy, B. Sabarish, and R. Menan, "*User Story based Automated Test Case Generation using NLP*," in *Computational Intelligence in Data Science*, Chennai, India, May 2024, pp. 156–166. DOI: 10.1007/978-3-031-69982-5_12.

[6] T. Rahman and Y. Zhu, "*Automated User Story Generation with Test Case Specification using Large Language Model*," Apr. 2024, [Online]. Available: http://arxiv.org/abs/2404.01558

[7] N. Medeshetty, A. N. Ghazi, S. Alawadi, and F. Alkhabbas, "*From Requirements to Test Cases: An NLP-based Approach for High-Performance ECU Test Case Automation*," 2025, DOI: https://doi.org/10.48550/arXiv.2505.00547.

[8] M. Boukhlif, M. Hanine, N. Kharmoum, A. Ruigomez Noriega, D. Garcia Obeso, and I. Ashraf, "*Natural Language Processing-based Software Testing: A Systematic Literature Review*," *IEEE Access*, Vol. 12, pp. 79383–79400, 2024, DOI: 10.1109/ACCESS.2024.3407753.

[9] L. Zhao *et al.*, "*Natural Language Processing for Requirements Engineering*," *ACM Comput. Surv.*, Vol. 54, No. 3, Jan. 2022, DOI: 10.1145/3444689.

[10] J. Navarro and R. Ibarra, "*Automatic Test Case Generation using Natural Language Processing: A systematic mapping study*," *Inf. Softw. Technol.*, Vol. 189, Jan. 2026, DOI: 10.1016/j.infsof.2025.107929.

[11] J. Fischbach *et al.*, "*Automatic Creation of Acceptance Tests by Extracting Conditionals from Requirements: NLP approach and case study*," *Journal of Systems and Software*, Vol. 197, p. 111549, Mar. 2023, DOI: 10.1016/J.JSS.2022.111549.

[12] R. Gröpler, V. Sudhi, E. J. Calleja García, and A. Bergmann, "*NLP-based Requirements Formalization for Automatic Test Case Generation*," in *29th International Workshop on Concurrency, Specification and Programming (CS&P'21)*, Magdeburg, Germany: CEUR Workshop Proceedings, 2021. [Online]. Available: http://ceur-ws.org

[13] A. Fatima, A. Haider, and S. Reza, "*Automated Test Case Generation From Natural Language Requirements using Natural Language Processing*," *Journal of Computing and Biomedical Informatics*, Vol. 9, No. 2, Aug. 2025, DOI: 10.56979/902/2025.

[14] A. Prabu, S. L. A, S. B A, and A. K. C, "*User Story-based Automatic Test Case Classification and Prioritization using Natural Language Processing-based Deep Learning*," *IEEE Potentials*, Vol. 43, No. 5, pp. 20–28, 2024, DOI: 10.1109/MPOT.2023.3342366.

[15] N. Gupta, V. Yadav, and M. Singh, "*Decision Tree based Test Case Generation using Story Board and Natural Language Processing*," in *Advances in Computing and Data Sciences*, Cham: Springer Nature Switzerland, 2023, pp. 581–591.

[16] S. El Farisi and F. A. Marva, "Evaluasi Teknik *Prompting* pada *Large Language* Model untuk Otomatisasi Penyusunan Skenario Unit Testing *Smart Contract*," *Jurnal Teknologi Informasi dan Multimedia*, Vol. 8, No. 1, pp. 99–108, Jan. 2026, DOI: 10.35746/jtim.v8i1.912.

[17] B. Hardika *et al.*, "Pengujian *Blackbox Testing Website* Garuda Farm menggunakan Teknik *Equivalence Partitioning*," *Jurnal Kridatama Sains dan Teknologi*, Vol. 06, No. 02, pp. 747–751, 2024.

ISSN:2302-8149
e-ISSN:2540-9719

[18] F. Koto, A. Rahimi, J. H. Lau, and T. Baldwin, "*IndoLEM and IndoBERT: A Benchmark Dataset and Pre-trained Language Model for Indonesian NLP*," Nov. 2020, [Online]. Available: http://arxiv.org/abs/2011.00677

[19] W. Widyawan, B. P. Utomo, and M. N. Rizala, "*A Novel Fusion of Machine Learning Methods for Enhancing Named Entity Recognition in Indonesian Language Text*," *Jurnal Sistem Informasi Bisnis*, Vol. 14, No. 4, pp. 311–320, Oct. 2024, DOI: 10.21456/vol14iss4pp311-320.