

Kombinasi Algoritma CRNN dan LLM untuk Pengembangan Aplikasi Pembagian Tagihan Otomatis

Daud Aldo Santoso^{*1}, Ryan Putranda Kristianto², Andre Hartanto³, Brielt Bella Gracitwo⁴

^{1, 2, 3, 4*} Program Studi Ilmu Informatika, Fakultas Teknik, Universitas Katolik Darma Cendika, Surabaya

*Co-author: daud.santoso@student.ukdc.ac.id

Informasi Artikel	Abstrak
Tanggal: Dikirim : 08 Ags 2026 Direvisi : 20 Ags 2026 Diterima : 20 Sep 2026 Kata Kunci: CRNN, T5, OCR, nota restoran, pembagian tagihan Keyword: CRNN, T5, OCR, receipt, bill splitting 	Sektor penyediaan akomodasi maupun makan minum yang terus berkembang meningkatkan aktivitas pembagian tagihan. Perhitungan nominal secara manual memakan waktu lama serta rentan terhadap kesalahan hitung, terutama pada pembagian beban pajak maupun service charge. Untuk mengatasi masalah tersebut, penelitian ini merancang sebuah aplikasi pembagian tagihan otomatis. Metode yang digunakan mencakup algoritma Convolutional Recurrent Neural Network (CRNN) dengan optimasi Open Visual Inference and Neural Network Optimization Toolkit (OpenVINO) untuk membaca teks dari nota fisik, yang dilatih dari awal menggunakan data sintesis nota berbahasa Indonesia dan dataset CORD sehingga mencapai Word Accuracy Rate (WAR) 89,7%, Character Accuracy Rate (CAR) 95,88%, dan Character Error Rate (CER) 4,12%. Teks mentah tersebut kemudian diproses menggunakan model T5 yang dilatih sendiri (self-hosted) untuk menyusun data pesanan beserta nominal harga serta mendeteksi skema penerapan service charge sebelum maupun sesudah pemotongan pajak, dengan Precision 87,8%, Recall 88,2%, dan F1-Score 88,0%. Sistem dibangun menggunakan kerangka kerja Flutter pada antarmuka aplikasi seluler Android dan iOS, serta FastAPI pada sisi server. Pengujian fungsionalitas menggunakan blackbox testing menunjukkan seluruh 15 skenario berjalan sesuai harapan, sedangkan pengujian usability menggunakan System Usability Scale (SUS) terhadap 42 responden menghasilkan skor rata-rata 98,81 dari 100 (kategori Excellent, grade A).

Abstract

Manual bill splitting during group dining is time-consuming and prone to calculation errors, particularly when allocating taxes and service charges proportionally. This study develops an automatic bill-splitting application for Android and iOS platforms. The method includes a Convolutional Recurrent Neural Network (CRNN) algorithm optimized with the Open Visual Inference and Neural Network Optimization

Toolkit (OpenVINO) for reading text from physical receipts, trained from scratch using synthetic Indonesian receipt data and the CORD dataset, achieving Word Accuracy Rate (WAR) of 89.7%, Character Accuracy Rate (CAR) of 95.88%, and Character Error Rate (CER) of 4.12%. OCR text is processed by a self-hosted T5 model to structure order data and detect service charge schemes, achieving Precision 87.8%, Recall 88.2%, and F1-Score 88.0%. The system is built using Flutter for Android and iOS, and FastAPI on the server side. Blackbox testing across 15 scenarios all passed, and System Usability Scale (SUS) testing with 42 respondents yielded an average score of 98.81 out of 100 (Excellent, Grade A).

1. PENDAHULUAN

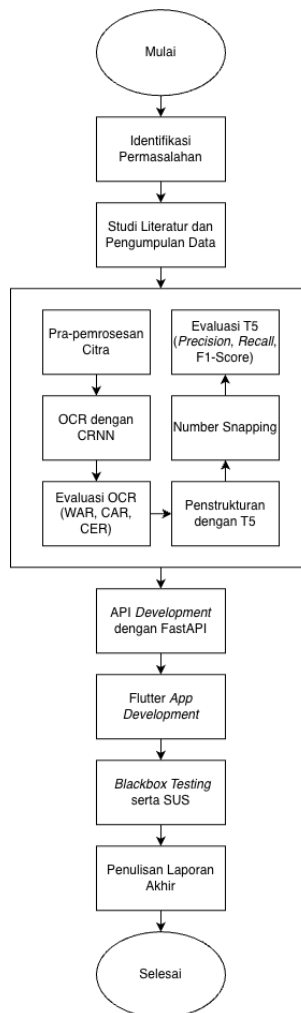
Sektor penyediaan akomodasi serta makan minum di Indonesia tumbuh dengan cepat, yang menunjukkan tingginya jumlah transaksi pada restoran maupun kafe [1]. Makan bersama sudah menjadi kebiasaan berbagai kalangan usia, mulai dari pelajar dan mahasiswa, pekerja kantoran, hingga keluarga dan komunitas. Semakin sering kegiatan ini dilakukan, semakin sering pula muncul kebutuhan untuk membagi tagihan secara bersama-sama, yaitu menentukan berapa nominal yang harus dibayar oleh masing-masing orang [2]. Kebutuhan ini sudah dijawab oleh beberapa penyedia layanan keuangan digital di Indonesia lewat fitur pembagian tagihan otomatis. GoPay menyediakan fitur *Split Bill* yang memungkinkan pengguna memfoto nota untuk dihitung secara otomatis, lalu menagihkan bagian masing-masing anggota melalui nomor telepon yang terdaftar. ShopeePay menyediakan fitur serupa yaitu memindai atau mengunggah nota, menambahkan anggota, lalu mengatur pembagian biaya. Aplikasi LINE menyediakan fitur *Split Bill* yang terhubung dengan LINE Pay untuk membagi tagihan antar sesama pengguna.

Meskipun begitu, ketiga aplikasi tersebut memiliki satu keterbatasan yang sama, yaitu seluruh proses pemasukan data hanya dapat dilakukan oleh satu orang saja. Selain itu, proses pengenalan nota berjalan pada layanan tertutup milik penyedia sehingga model, dataset, maupun tingkat akurasi tidak dapat diketahui maupun direplikasi untuk keperluan penelitian. Dari sisi penelitian akademis, Liu et al. [3] menerapkan arsitektur CRNN untuk pengenalan teks, Wei et al. [4] membuktikan LLM mampu melakukan ekstraksi informasi secara *zero-shot*, Blecher et al. [5] mengembangkan model Nougat berbasis *transformer* untuk ekstraksi teks dokumen akademik, serta Peng et al. [6] dan Lv et al. [7] mengembangkan model multimodal yang memproses gambar dan teks secara bersamaan, namun memiliki beban komputasi yang berat. Berdasarkan uraian di atas, penelitian ini berfokus pada keterbatasan yang belum diselesaikan oleh ketiga aplikasi tersebut, yaitu belum tersedianya sesi bersama yang memungkinkan setiap anggota memilih sendiri pesannya. Oleh karena itu, penelitian ini merancang aplikasi pembagian tagihan otomatis pada perangkat Android dan iOS yang menyediakan sesi bersama, sehingga satu nota dapat dibuka oleh beberapa anggota sekaligus dan tiap anggota dapat menandai sendiri pesanan miliknya dari perangkat masing-masing tanpa perlu menunggu pembuat tagihan mengaturnya. Untuk mendukung sesi tersebut, penelitian ini menyusun alur tiga tahap yang seluruhnya dijalankan sendiri. Tahap pertama adalah pra-pemrosesan citra untuk mencari dan menandai posisi baris teks pada nota. Tahap kedua adalah pembacaan huruf memakai CRNN yang dilatih dari awal, dipilih karena bentuknya ringan, tidak perlu memotong teks per huruf, dan bisa diubah ke format OpenVINO supaya bisa berjalan di server yang hanya memakai CPU tanpa GPU. Tahap ketiga

adalah penyusunan teks mentah hasil OCR memakai model bahasa *Text-to-Text Transfer Transformer* (T5) yang dilatih sendiri dengan data nota, dipilih karena ukurannya kecil sehingga bisa dijalankan sendiri, bersifat *open-source* sehingga hasilnya bisa diuji ulang oleh peneliti lain, serta mampu merapikan teks yang berantakan menjadi daftar nama menu beserta harganya tanpa perlu tahu posisi teksnya. Semua model dihubungkan ke aplikasi seluler lewat API yang dibuat memakai FastAPI, dengan model CRNN diubah ke format OpenVINO supaya waktu prosesnya lebih cepat.

2. METODE PENELITIAN

Penelitian ini terdiri dari enam tahapan. Pertama, identifikasi permasalahan pembagian tagihan manual. Kedua, studi literatur serta pengumpulan dataset CORD v2 [14] dan 8.000 nota sintetis berbahasa Indonesia. Ketiga, perancangan, pelatihan, serta evaluasi model CRNN menggunakan metrik CER, CAR, dan WAR. Keempat, pelatihan dan evaluasi model T5 menggunakan metrik *Precision*, *Recall*, dan *F1-Score*. Kelima, pengembangan *back-end* FastAPI dan aplikasi Flutter Android serta iOS. Keenam, pengujian *blackbox* dan pengujian *usability* menggunakan SUS. Alur keseluruhan tahapan penelitian ditampilkan pada Gambar 1.



Gambar.1 Alur Metode Penelitian

A. Model CRNN

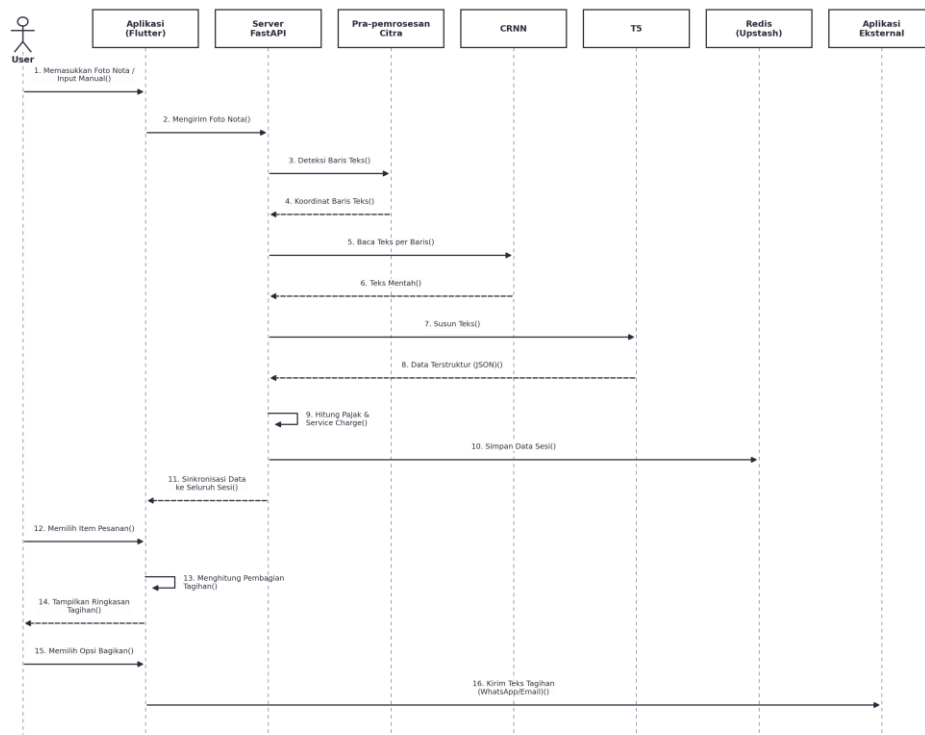
Model CRNN bertugas membaca urutan huruf pada tiap baris teks nota. Arsitektur terdiri dari tujuh lapisan konvolusi (CNN) untuk mengambil fitur visual, dua lapisan *Bidirectional Long Short-Term Memory* (BLSTM) untuk menyusun urutan karakter, dan satu lapisan *Connectionist Temporal Classification* (CTC) yang mengubah prediksi per timestep menjadi teks utuh tanpa perlu memotong gambar per huruf [3]. Model dilatih dari awal menggunakan 30.804 potongan teks dari dataset CORD v2 [14] dan data sintetis on-the-fly dengan charset 46 karakter (huruf kapital A-Z, angka 0-9, dan simbol umum nota).

B. Model T5

Model T5-*small* dilatih ulang (*fine-tuning*) menggunakan gabungan dataset CORD v2 dan 8.000 nota sintetis berbahasa Indonesia [8], menghasilkan sekitar 10.000 pasangan data latih. Setiap pasangan terdiri dari masukan berupa teks mentah nota dengan awalan "ekstrak nota: " dan keluaran berupa data terstruktur dalam format ITEM, TAX, SERVICE, dan DISCOUNT. Proses *fine-tuning* dijalankan selama 3 *epoch* menggunakan *optimizer* AdamW dengan *learning rate* 3×10^{-4} dan *batch size* 8.

C. Arsitektur Sistem

Sistem dibangun menggunakan FastAPI pada sisi server dan Flutter pada sisi klien Android dan iOS. Model CRNN diekspor ke format OpenVINO untuk mengoptimalkan kecepatan inferensi pada CPU, menghasilkan peningkatan kecepatan 2,7 kali lipat dibandingkan model PyTorch asli. Sinkronisasi data antar perangkat menggunakan WebSocket, dengan Redis (Upstash) sebagai penyimpanan bersama dengan masa berlaku otomatis 6 jam. Alur komunikasi antar komponen sistem digambarkan pada Gambar 2.



Gambar.2 Sequence Diagram

3. HASIL DAN PEMBAHASAN

A. Evaluasi Model CRNN

Model CRNN diuji menggunakan 1.540 potongan teks dari data uji yang belum pernah dilihat model. Hasil evaluasi ditampilkan pada Tabel 1.

Tabel.1 Hasil Evaluasi Model CRNN

Metrik	Nilai
<i>Word Accuracy Rate</i> (WAR)	89,7%
<i>Character Accuracy Rate</i> (CAR)	95,88%
<i>Character Error Rate</i> (CER)	4,12%
<i>Epoch</i> Terbaik	23 dari 25
<i>Validation Loss</i>	0,027

Berdasarkan Tabel 1, model CRNN berhasil mencapai WAR 89,7% dan CER 4,12%. Dari setiap 100 kata yang dipindai, sekitar 90 kata terbaca seratus persen benar. Untuk menguji kemampuan generalisasi, dilakukan pengujian tambahan menggunakan 100 gambar nota dari test split dataset CORD v2 menggunakan koordinat anotasi resmi, menghasilkan 2.348 potongan kata. Hasil ditampilkan pada Tabel 2.

Tabel.2 Hasil Evaluasi Model CRNN

Dataset	Jumlah Data	WAR	CAR	CER
CORD v2 (<i>training crops</i>)	1.540 potongan	89,7%	95,88%	4,12%
CORD v2 (<i>test split</i> , 100 gambar)	2.348 potongan	96,3%	99,04%	0,96%

Berdasarkan Tabel 2, model CRNN menunjukkan WAR 96,3% dan CER 0,96% pada data uji yang belum pernah digunakan selama pelatihan. Hasil ini lebih tinggi dibandingkan evaluasi pada *training crops*, yang menunjukkan bahwa model tidak mengalami *overfitting* dan memiliki kemampuan generalisasi yang baik.

B. Evaluasi Model T5

Model T5 dievaluasi menggunakan 100 sampel nota uji dari dataset CORD v2. Hasil evaluasi ditampilkan pada Tabel 3.

Tabel.3 Hasil Evaluasi Model T5

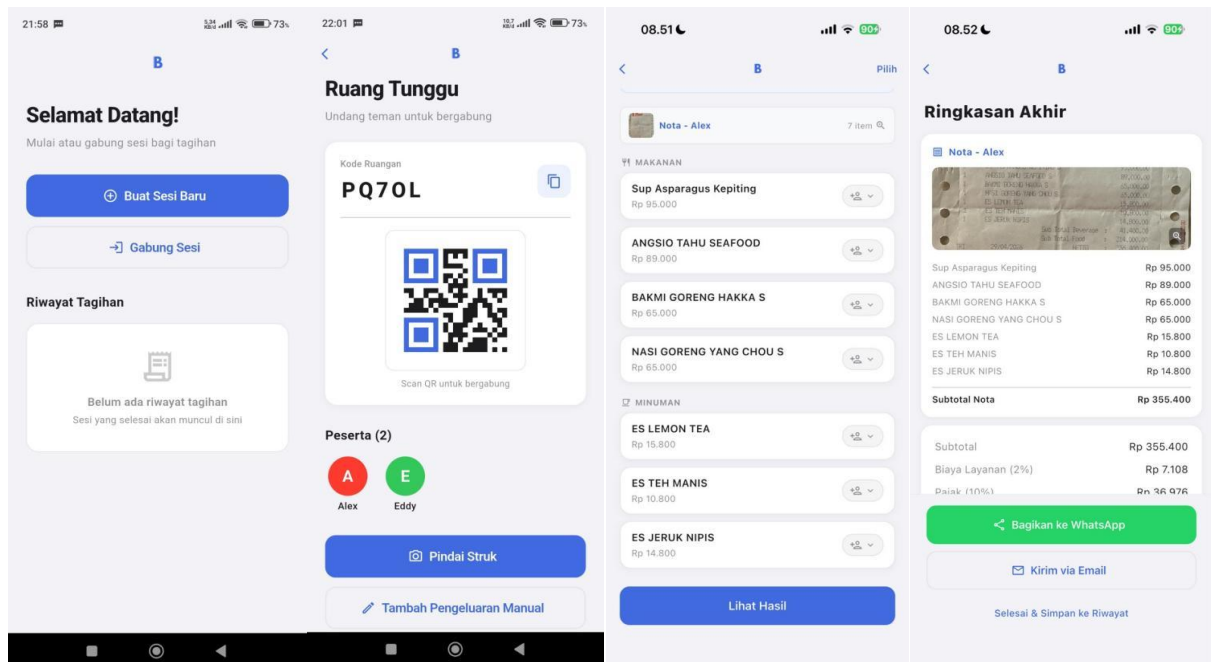
Metrik	Nilai
<i>Precision</i>	87,8%
<i>Recall</i>	88,2%
<i>F1-Score</i>	88,0%
Akurasi Pajak	98%
Akurasi <i>Service Charge</i>	99%
Akurasi Diskon	99%

Berdasarkan Tabel 3, model T5 mencapai *F1-Score* 88,0% untuk ekstraksi item menu beserta nominalnya. Nilai *Precision* dan *Recall* yang berdekatan menunjukkan model seimbang antara ketepatan dan kelengkapan ekstraksi. Contoh masukan: "ekstrak nota: NASI GORENG 45000 ES TEH 15000 PB1 6000", dan

contoh keluaran: "ITEM: NASI GORENG | 45000 | 1 ITEM: ES TEH | 15000 | 1 TAX: 6000 SERVICE: 0 DISCOUNT: 0".

C. Implementasi Aplikasi Flutter

Aplikasi seluler dibangun menggunakan Flutter dengan pola pengelolaan state memakai pustaka *Provider*. Satu kode program yang sama dikompilasi menjadi dua aplikasi, satu untuk Android (minimal SDK 21) [19] dan satu untuk iOS (minimal versi 14) [20]. Pengujian kolaborasi menggunakan satu perangkat Android dan satu perangkat iOS yang bergabung dalam sesi yang sama menunjukkan bahwa perubahan data langsung tersinkronisasi secara *real-time* melalui koneksi WebSocket. Tampilan antarmuka aplikasi pada kedua platform ditampilkan pada Gambar 3.



Gambar.3 Screenshot Aplikasi

D. Pengujian Blackbox

Pengujian *blackbox* dilakukan dengan mencoba 15 skenario penggunaan utama di Android dan iOS [10]. Seluruh 15 skenario menghasilkan status Valid. Ringkasan pengujian ditampilkan pada Tabel 4.

Pengujian dilakukan menggunakan dua perangkat seluler secara bersamaan, yaitu satu perangkat Android (Xiaomi Redmi Note 11 Pro, Snapdragon 732G, Android 11, RAM 8 GB, storage 128 GB) dan satu perangkat iOS (iPhone 16 Pro Max, Apple A18 Pro, iOS 18, RAM 8 GB, storage 1 TB). Koneksi internet menggunakan jaringan WiFi dengan kecepatan rata-rata 300 Mbps. Server *back-end* dijalankan pada platform Hugging Face Spaces menggunakan CPU AMD EPYC dengan RAM 16 GB tanpa GPU.

Pengujian dilakukan oleh dua penguji yang berperan sebagai user dalam satu sesi yang sama. Penguji pertama berperan sebagai *host* (pembuat sesi) yang bertanggung jawab membuat ruang sesi, memindai nota, dan membagikan kode room. Penguji kedua berperan sebagai *guest* (teman yang bergabung) yang masuk ke sesi menggunakan kode *room* dan memilih pesanan miliknya dari perangkat terpisah.

Tabel.4 Ringkasan Hasil Pengujian *Blackbox*

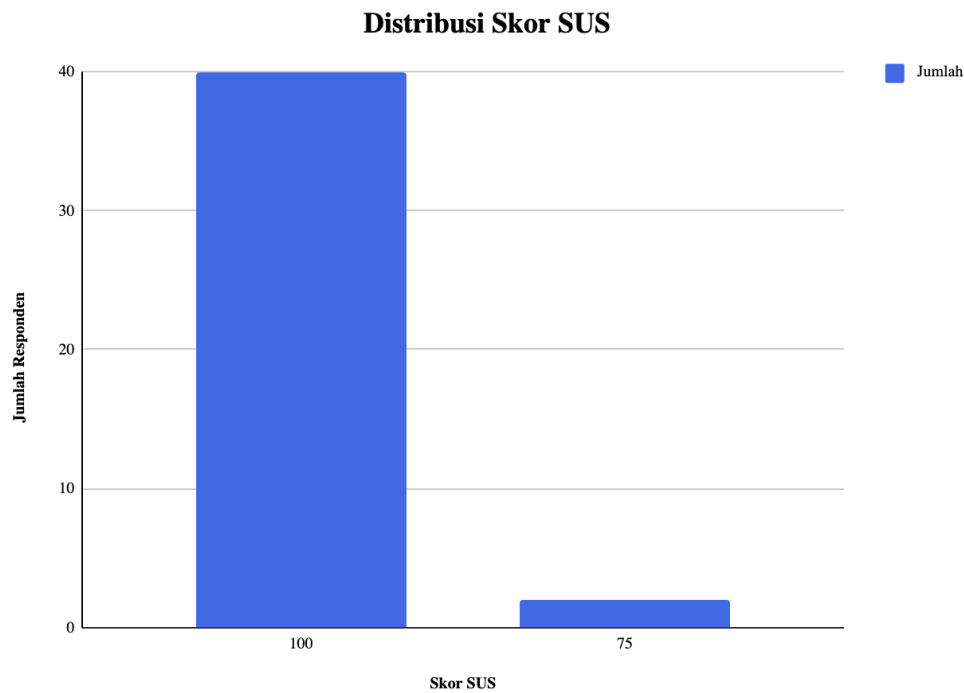
No	Skenario Pengujian	Hasil yang Diharapkan	Status
1	Membuat ruangan sesi baru	Kode ruangan dan <i>QR Code</i> muncul	Valid
2	Bergabung dengan kode valid	Berhasil masuk ke ruang tunggu	Valid
3	Bergabung dengan kode tidak valid	Pesan kesalahan ditampilkan	Valid
4	Memindai foto nota restoran	Item dan harga berhasil terbaca	Valid
5	Deteksi skema pajak otomatis	Skema pajak terdeteksi dengan benar	Valid
6	Deteksi diskon pada nota	Nominal diskon terdeteksi dengan benar	Valid
7	Penugasan item ke partisipan	Avatar partisipan muncul pada item	Valid
8	Sinkronisasi antar perangkat	Data <i>ter-update</i> secara <i>real-time</i>	Valid
9	Entri pengeluaran manual	Item pengeluaran muncul di daftar	Valid
10	Kalkulasi tagihan proporsional	Nominal tagihan terhitung dengan benar	Valid
11	Edit nominal tagihan akhir	Nominal <i>ter-update</i> pada semua perangkat	Valid
12	Membagikan hasil via WhatsApp	Aplikasi WhatsApp terbuka dengan ringkasan	Valid
13	Penyimpanan riwayat sesi	Riwayat muncul pada halaman beranda	Valid
14	Keluar dari sesi	Partisipan terhapus dari semua perangkat	Valid
15	<i>Auto-save</i> saat berpindah aplikasi	Sesi dapat dilanjutkan tanpa kehilangan data	Valid

E. Pengujian *Usability* (SUS)

Pengujian *usability* dilakukan menggunakan instrumen SUS terhadap 42 responden [9]. Setiap jawaban dikonversi menggunakan rumus $(\text{skor jawaban} - 1) \times 2,5$ untuk mendapatkan skor SUS berskala 0-100. Distribusi skor menunjukkan 40 responden (95,2%) memberikan skor sempurna 100, sedangkan 2 responden (4,8%) memberikan skor 75. Distribusi skor SUS dari seluruh responden ditampilkan pada Gambar 4. Skor rata-rata keseluruhan sebesar 98,81 dari 100 masuk dalam kategori Excellent (Grade A), menunjukkan aplikasi dinilai mudah digunakan oleh responden.

Pengujian dilakukan menggunakan perangkat yang sama dengan pengujian *blackbox*, yaitu perangkat Android dan iOS. Kuesioner SUS disebarikan secara digital menggunakan Google Forms dan diisi langsung oleh responden setelah mencoba aplikasi secara langsung di perangkat masing-masing.

Responden terdiri dari 42 orang dengan latar belakang beragam, meliputi mahasiswa (71,4%), pekerja (19,1%), dan umum (9,5%), dengan rentang usia 18 hingga 35 tahun. Seluruh responden sudah pernah melakukan kegiatan makan bersama dan membagi tagihan sebelumnya, sehingga dapat memberikan penilaian yang relevan terhadap aplikasi.



Gambar.4 Grafik distribusi skor SUS

4. KESIMPULAN

Penelitian ini berhasil merancang dan mengimplementasikan aplikasi pembagian tagihan otomatis yang bersifat self-hosted pada platform Android dan iOS berbasis Flutter. Kontribusi utama penelitian ini mencakup tiga hal. Pertama, *pipeline* OCR dua tahap yang menggabungkan CRNN berformat OpenVINO

dengan model T5-*small* yang dilatih sendiri, sehingga seluruh sistem dapat berjalan pada server CPU tanpa bergantung pada GPU maupun API komersial berbayar. Kedua, penerapan model T5 untuk mengekstraksi data terstruktur dari teks OCR nota restoran Indonesia tanpa analisis tata letak spasial, mencapai *F1-Score* 88,0%. Ketiga, integrasi sesi multi-pengguna *real-time* berbasis WebSocket dan Redis dalam satu aplikasi multiplatform dari satu basis kode Flutter.

Model CRNN mencapai WAR 89,7% dan CER 4,12% pada data uji, serta WAR 96,3% dan CER 0,96% pada pengujian generalisasi. Model T5 mencapai *F1-Score* 88,0%. Pengujian *blackbox* pada 15 skenario seluruhnya Valid, dan pengujian SUS terhadap 42 responden menghasilkan skor rata-rata 98,81 dari 100 (kategori Excellent, Grade A).

DAFTAR PUSTAKA

- [1] Aisy, R., Mursityo, Y. T., & Wijoyo, S. H. (2024). Evaluasi Usability Aplikasi Mobile Sampingan Menggunakan Metode Usability Testing dan System Usability Scale (SUS). *Jurnal Teknologi Informasi Dan Ilmu Komputer*, 11(1), 19–26. <https://doi.org/10.25126/jtiik.20241116613>
- [2] Amelia, M. (2025). Mengenal Split Bill, Sistem Patungan Bayar Ala Gen Z yang Praktis tapi Kadang Bikin Rusak Pertemanan. *JawaPos.Com*.
- [3] Blecher, L., Cucurull, G., Scialom, T., & Stojnic, R. (2023). Nougat: Neural Optical Understanding for Academic Documents. *arXiv:2308.13418*.
- [4] BPS-Statistics Indonesia. (2024). Food and Beverage Service Activities Statistics 2023. BPS-Statistics Indonesia.
- [5] Fan, Y., Liu, Y., Yao, Z., Yu, J., Hou, L., & Li, J. (2024). Evaluating generative language models in information extraction as subjective question correction. *arXiv:2404.03532*.
- [6] FastAPI - FastAPI. *Tiangolo.Com*. Retrieved July 14, 2026, from <https://fastapi.tiangolo.com/>
- [7] Feng, W., Zhu, W., Fu, T.-J., Jampani, V., Akula, A., He, X., Basu, S., Wang, X. E., & Wang, W. Y. (2023). LayoutGPT. *arXiv:2305.15393*.
- [8] Flutter - Build apps for any screen. *Flutter.Dev*. Retrieved July 14, 2026, from <https://flutter.dev/>
- [9] Gabín, J., Eduardo Ares, M., & Parapar, J. (2024). Enhancing automatic keyphrase labelling with T5 architecture. *arXiv:2409.16760*.
- [10] Handayanto, I. S., & Nuryasin, I. (2024). Pengujian Blackbox Decision Table pada Sistem Aplikasi Mobile Sharing Story App. *Smart Comp*, 13(2), 383–394.
- [11] Liu, Y., Wang, Y., & Shi, H. (2023). A convolutional recurrent neural-network-based machine learning for scene text recognition application. *Symmetry*, 15(4), 849.
- [12] Lv, T., et al. (2023). KOSMOS-2.5: A Multimodal Literate Model. *arXiv:2309.11419*.
- [13] OpenVINO 2026.2 Documentation. *Openvino.Ai*. Retrieved July 14, 2026, from <https://docs.openvino.ai/>
- [14] Park, S., Shin, S., Lee, B., Lee, J., Surh, J., Seo, M., & Lee, H. (2019). CORD: A consolidated receipt dataset for post-OCR parsing. *NeurIPS Workshop*.
- [15] Peng, Z., Wang, W., Dong, L., Hao, Y., Huang, S., Ma, S., & Wei, F. (2023). Kosmos-2. *arXiv:2306.14824*.
- [16] Touvron, H., et al. (2023). LLaMA: Open and efficient foundation language models. *arXiv:2302.13971*.

- [17]Wei, X., et al. (2023). ChatIE: Zero-shot information extraction via chatting with ChatGPT. arXiv:2302.10205.
- [18]Zhao, W. X., et al. (2023). A survey of large language models. arXiv:2303.18223.
- [19]Android Open Source Project. (2026). Android Open Source Project. Retrieved from <https://source.android.com/>
- [20]Apple Inc. (2026). Apple Developer. Retrieved from <https://developer.apple.com/>