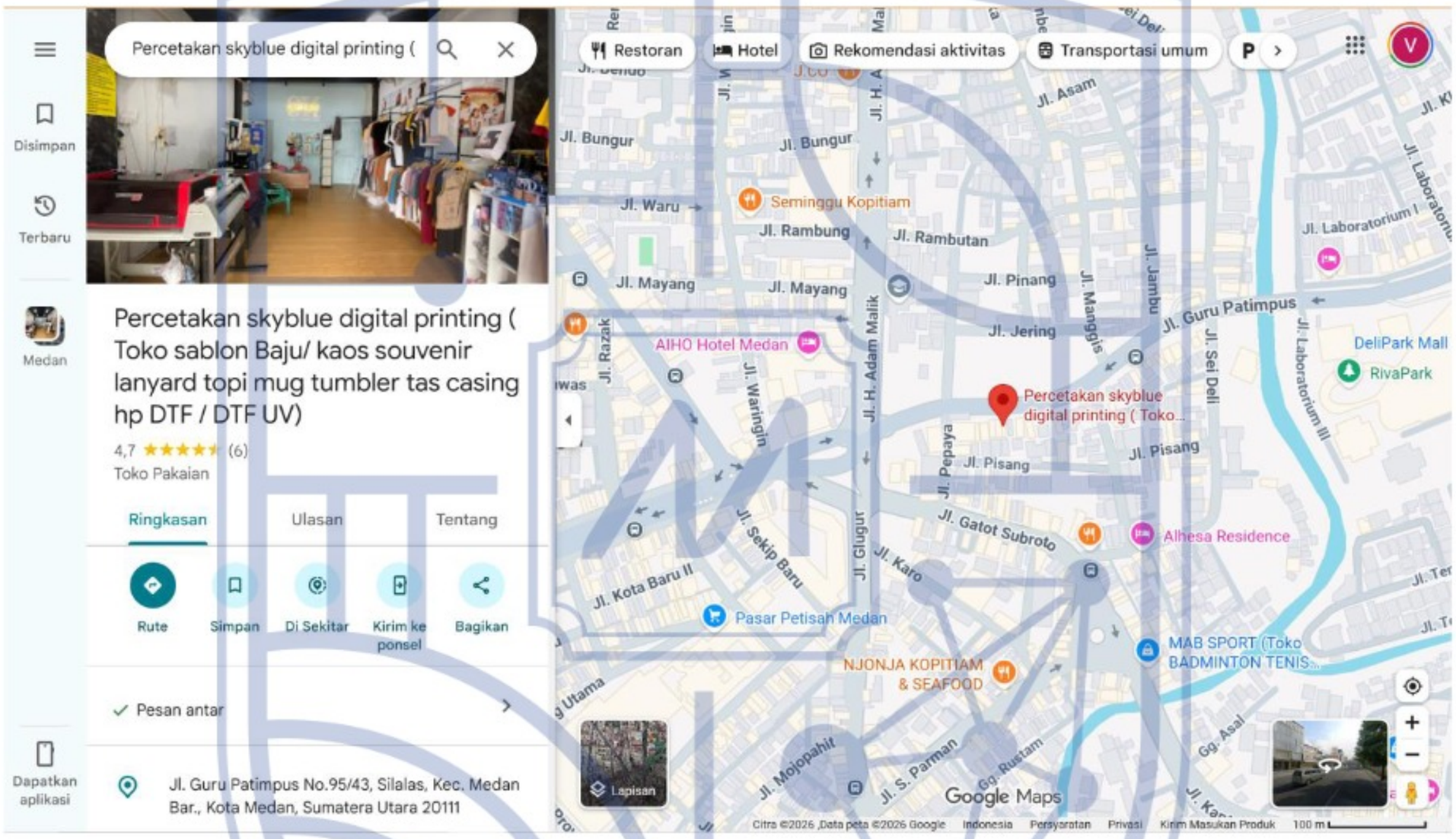
BAB II
KAJIAN LITERATUR

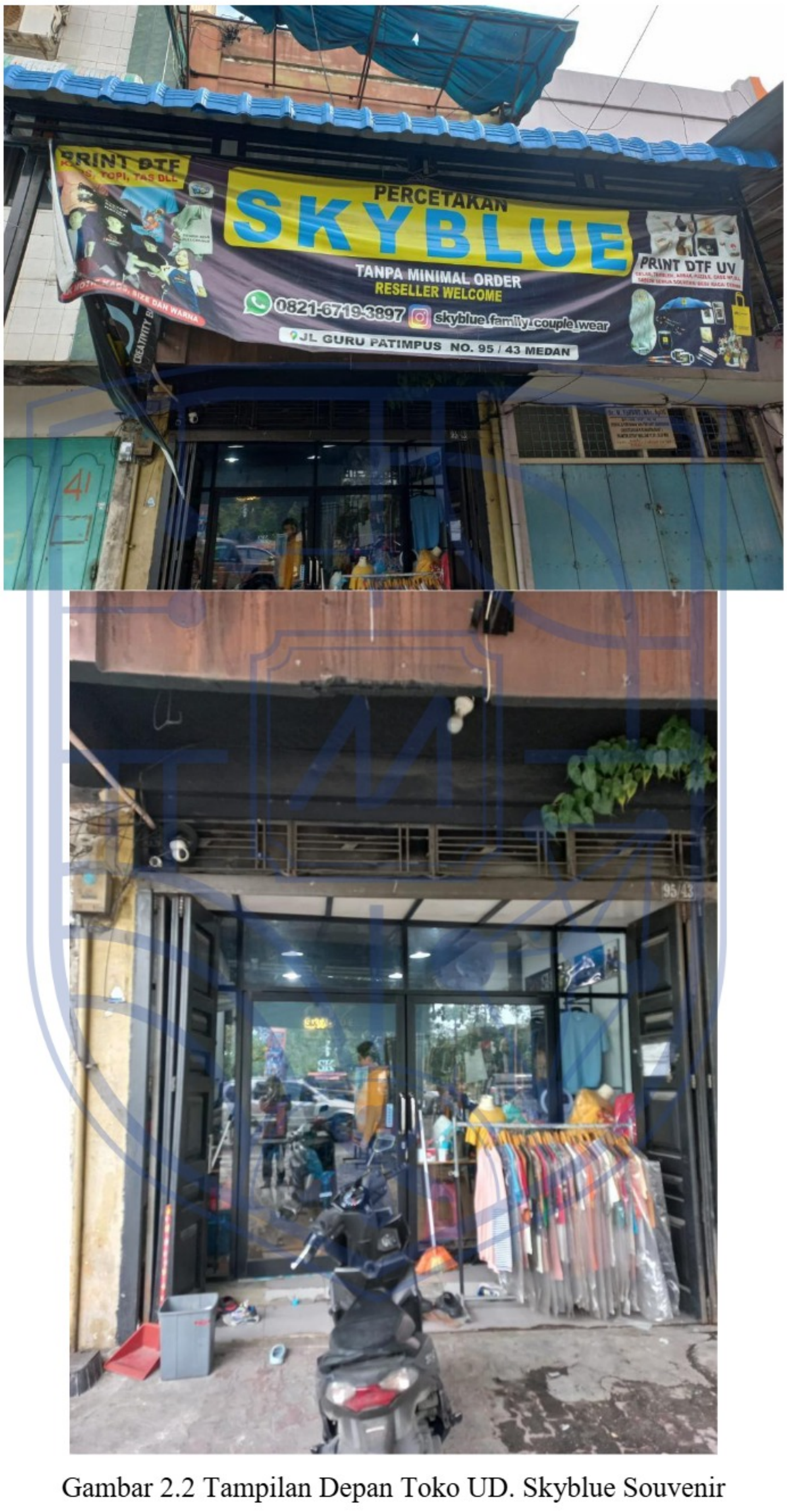
2.1 Profil Mitra (UD. Skyblue Souvenir)

UD. Skyblue Souvenir merupakan UMKM yang bergerak di bidang percetakan dan perdagangan barang souvenir serta kustomisasi produk. UD. Skyblue Souvenir berlokasi di Jl. Guru Patimpus No. 95/43. Lokasi toko UD. Skyblue Souvenir dapat diakses melalui layanan *Google Maps* ditunjukkan pada Gambar 2.1.



Gambar 2.1 Lokasi UD. Skyblue Souvenir pada *Google Maps*

Dalam menjalankan kegiatan operasional bisnisnya, UD. Skyblue Souvenir melayani pembelian langsung secara *offline* melalui toko fisik maupun *online* melalui toko digital pada platform Shopee. Tampilan fisik bagian depan toko UD. Skyblue Souvenir disajikan pada Gambar 2.2.



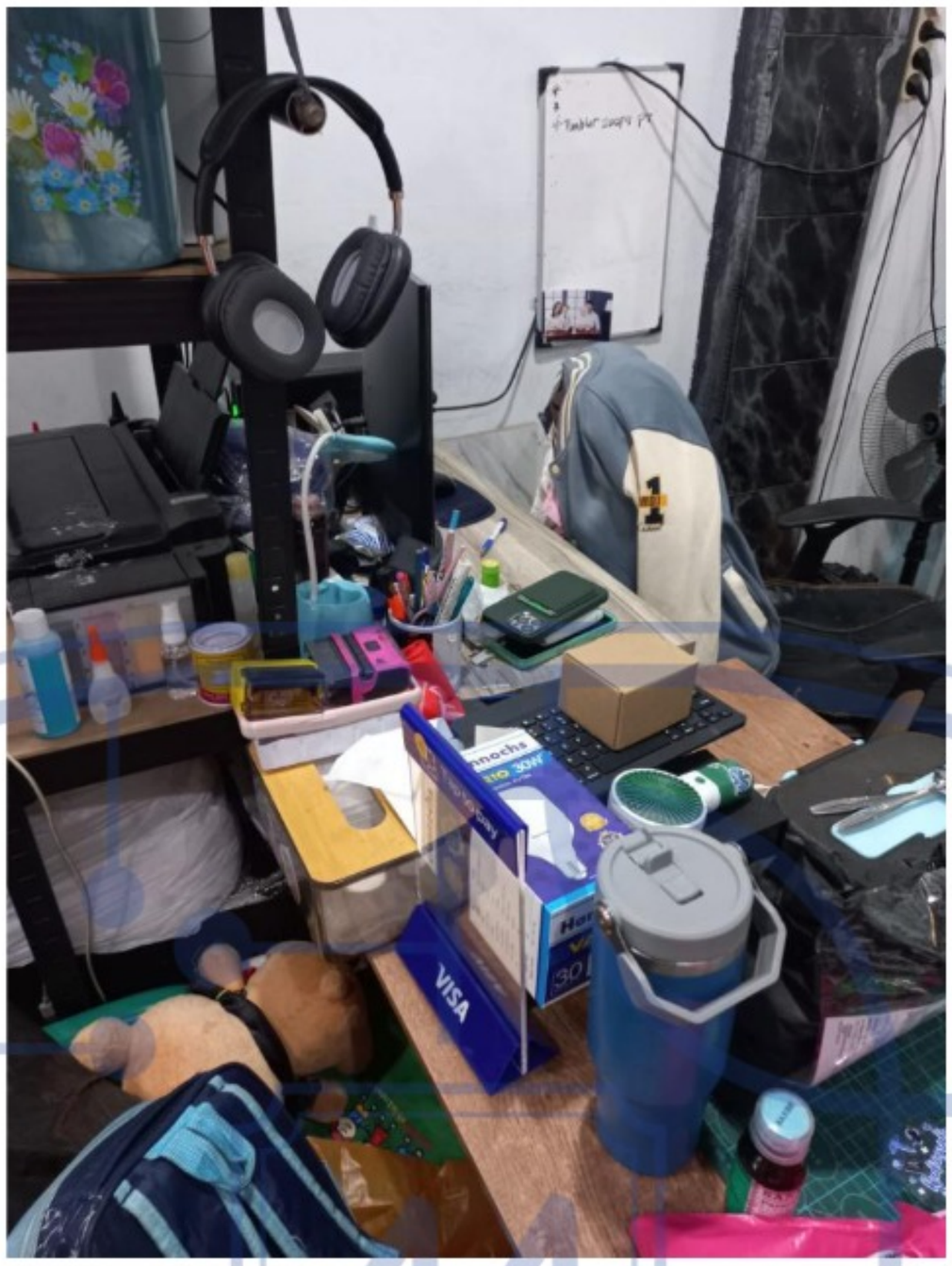
Gambar 2.2 Tampilan Depan Toko UD. Skyblue Souvenir

Produk-produk yang ditawarkan oleh UD. Skyblue Souvenir meliputi custom design kaos, gelas, mug, piring, payung, bantal, tas, gantungan kunci, topi, dan souvenir lainnya. Untuk menunjang kualitas hasil cetakan pada berbagai media produk souvenir

tersebut, mitra mengandalkan perangkat *printer* pencetakan khusus yang ditunjukkan pada Gambar 2.3. Perangkat ini berfungsi mencetak desain kustom ke media transfer sebelum diaplikasikan ke bahan produk.

Gambar 2.3 Perangkat Printer Operasional Produksi

Seluruh proses penerimaan pesanan, komunikasi dengan pelanggan *online*, hingga pembuatan rancangan visual (*design*) dilakukan di area meja kerja operasional admin. Seperti yang terlihat pada Gambar 2.4, area ini menjadi pusat pengelolaan aktivitas administrasi toko dan pemrosesan pesanan masuk dari platform Shopee maupun *offline*.

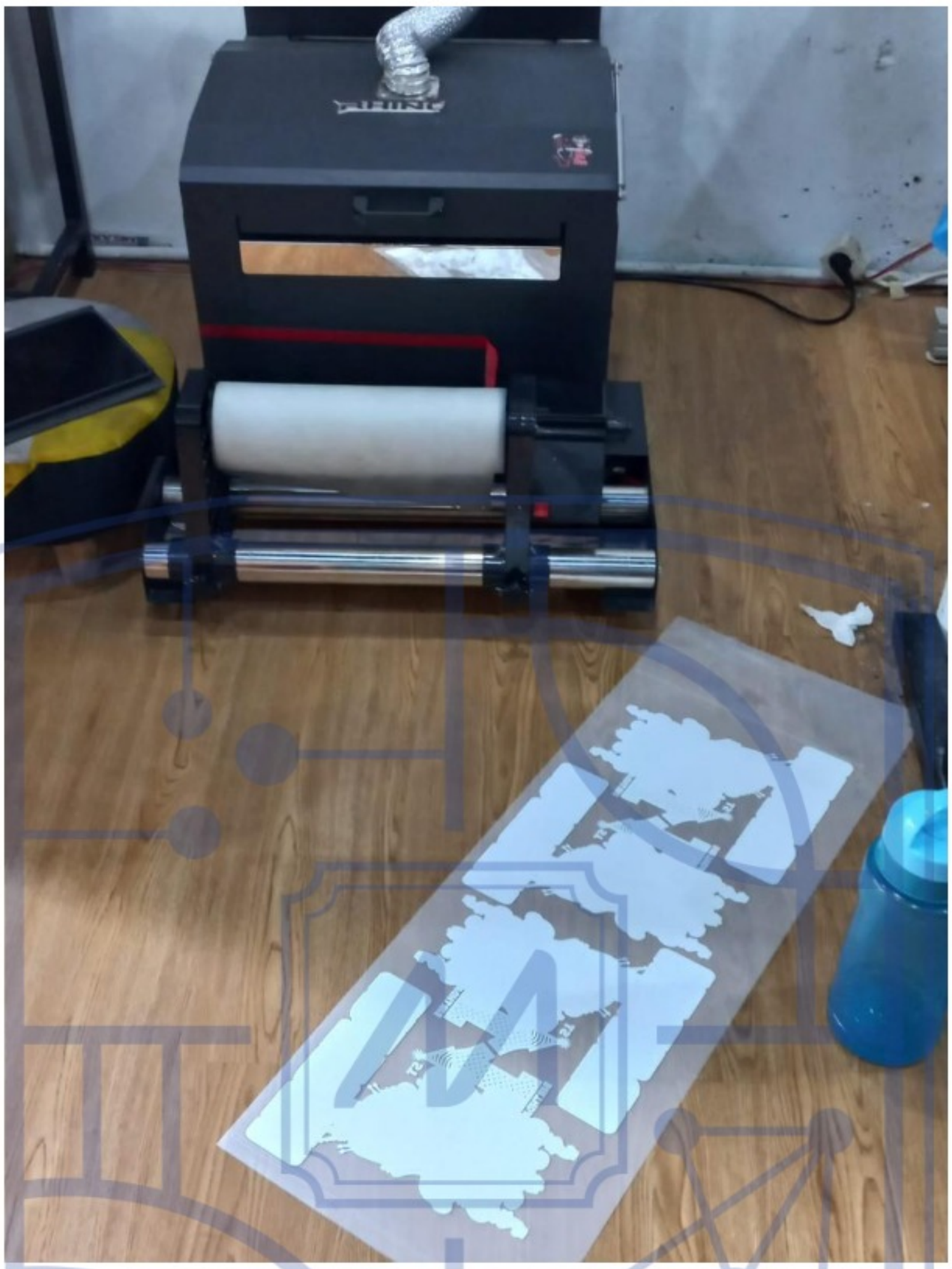


Gambar 2.4 Area Meja Kerja Operasional dan Admin

Ketersediaan barang mentah maupun produk siap cetak disimpan pada area penyimpanan stok bahan baku fisik (Gambar 2.5). Pada area ini, admin melakukan Pengecekan jumlah stok ketersediaan barang secara manual sebelum pesanan diproses lebih lanjut ke tahap pencetakan.

Gambar 2.5 Penyimpanan Stok Bahan Baku Fisik

Tahapan akhir dari pembuatan pesanan kustom dilaksanakan pada area pelaksanaan proses percetakan dan sublimasi produk (Gambar 2.6). Pada area ini, bahan baku polos diproses menggunakan teknik pres panas (*heat press*) atau sublimasi sesuai dengan spesifikasi pesanan pelanggan hingga menjadi barang jadi yang siap dikirim.



Gambar 2.6 Proses Percetakan dan Kustomisasi Souvenir

2.2 Manajemen Persediaan (*Inventory Management*)

Manajemen persediaan (*inventory management*) merupakan kegiatan pengelolaan barang yang mencakup proses perencanaan, pengendalian, dan pemantauan terhadap jumlah, lokasi, serta nilai barang yang dimiliki perusahaan agar kegiatan operasional dapat berjalan efektif dan efisien [5]. Tujuan utama manajemen persediaan adalah menjaga keseimbangan antara biaya persediaan dan tingkat ketersediaan barang, sehingga permintaan pelanggan dapat terpenuhi tanpa menimbulkan pemborosan biaya penyimpanan maupun risiko kekurangan stok [5]. Dalam perusahaan manufaktur, manajemen persediaan menjadi elemen penting karena berhubungan langsung dengan ketersediaan bahan baku dan kelancaran proses produksi [5].

Penelitian oleh Amelia et al. [5] menunjukkan bahwa penerapan metode *Economic Order Quantity* (EOQ) mampu membantu perusahaan dalam menentukan jumlah pemesanan optimal sehingga dapat menekan total biaya persediaan. Hal ini menegaskan

bahwa pendekatan kuantitatif dalam manajemen persediaan dapat meningkatkan efisiensi operasional, khususnya dalam pengendalian jumlah pembelian dan frekuensi pemesanan [5].

Namun, penelitian tersebut lebih berfokus pada aspek perhitungan dan optimasi biaya persediaan, serta belum membahas integrasi sistem pengelolaan stok secara digital, terutama pada lingkungan usaha kecil dan menengah (UMKM) yang memiliki keterbatasan dalam pencatatan dan sinkronisasi data secara *real-time* [5]. Selain itu, belum terdapat pembahasan terkait integrasi antara sistem *internal* perusahaan dengan platform eksternal seperti *marketplace* yang saat ini banyak digunakan dalam aktivitas penjualan [5].

2.3 Stok

Stok dapat didefinisikan sebagai persediaan yang pada kegiatan operasionalnya akan dijual dan berbentuk bahan baku atau perlengkapan yang akan digunakan [6]. Stok mencakup jumlah barang, lokasi penyimpanan, kondisi fisik barang, dan nilai ekonomisnya [6]. Ketidakakuratan dalam pencatatan stok dapat menimbulkan masalah seperti kelebihan stok, kekurangan stok, atau selisih antara stok yang tercatat dan stok fisik [7]. Ketidaksesuaian data stok antara sistem dan kondisi aktual menjadi tantangan utama pengelolaan persediaan [6,7].

Penelitian oleh Putri dan Handoko [8] mengungkapkan bahwa ketidaksesuaian stok sering terjadi akibat kesalahan input data, ketidaksesuaian prosedur saat stok opname, serta keterlambatan dalam pencatatan barang masuk dan keluar. Hal ini menunjukkan bahwa proses pencatatan manual memiliki tingkat risiko kesalahan yang tinggi [8]. Selain itu, Nursyanti dan Partisia [9] menemukan bahwa kurangnya standar operasional prosedur (SOP), lemahnya pengawasan, serta sistem penyimpanan yang tidak terintegrasi menjadi faktor utama terjadinya ketidaksesuaian inventaris. Sementara itu, Maulana dan Radiyanto [10] menunjukkan bahwa penggunaan aplikasi persediaan berbasis *Android* dapat meningkatkan akurasi data stok melalui pencatatan yang lebih cepat dan terstruktur. Sistem berbasis *mobile* memungkinkan pencatatan dilakukan secara langsung di lokasi, sehingga mengurangi keterlambatan *input* data dan selisih stok [10].

Meskipun demikian, penelitian-penelitian tersebut masih memiliki keterbatasan, yaitu belum mengintegrasikan sistem pengelolaan stok dengan platform penjualan *online* atau *marketplace* [8-10]. Padahal, dalam konteks bisnis modern, khususnya UMKM yang memanfaatkan *e-commerce*, ketidaksesuaian stok tidak hanya terjadi karena faktor internal, tetapi juga karena tidak adanya sinkronisasi antara penjualan *offline* dan *online* [6,7].

2.4 Struktur Organisasi Usaha Dagang (UD)

Dalam konteks UD. Skyblue Souvenir, struktur organisasi berperan penting dalam pengelolaan persediaan dan alur operasional harian. Usaha ini dipimpin oleh James Rusli yang bertanggung jawab sebagai pemilik dalam pemantauan laporan penjualan dan pembelian dan kontrol penuh terhadap manajemen akun dan data barang. Operasional sehari-hari dibantu oleh dua karyawan yang menangani berbagai tugas mulai dari transaksi penjualan dan pembelian, pengecekan pesanan, pembaruan stok, produksi barang (seperti pakaian, tumbler, dan souvenir), hingga persiapan pengiriman. Admin *Marketplace* bertanggung jawab mengelola pesanan *online* melalui Shopee, Instagram, dan WhatsApp, termasuk mengunggah produk, merespons pelanggan, dan menangani proses konfirmasi pesanan sebelum produksi.

Gambar 2.7 Struktur Organisasi UD. Skyblue Souvenir

Penggunaan sistem inventori nantinya akan melibatkan Pemilik, Admin *Marketplace*, dan dua karyawan sesuai fungsi masing–masing. Pemilik memiliki akses penuh ke laporan, manajemen akun, dan data barang, sementara dua karyawan bertugas melakukan transaksi, pengurangan stok otomatis, pengecekan stok fisik, dan operasional produksi, sedangkan Admin *Marketplace* memastikan stok *online* tetap selaras dengan kondisi aktual. Saat ini proses pencatatan stok masih dilakukan secara manual sehingga sering terjadi selisih antara stok fisik dan stok di Shopee, menyebabkan risiko pembatalan pesanan, keterlambatan update, dan *over-selling*.

Dengan struktur organisasi tersebut, kebutuhan utama sistem adalah integrasi antara stok *offline* (toko fisik) dan *online* (Shopee), pencatatan otomatis setiap transaksi, serta penyediaan laporan dalam rentang waktu tertentu yang dapat diekspor ke CSV. Sistem yang terintegrasi diharapkan dapat meningkatkan akurasi stok, mempercepat proses kerja, dan mendukung pemilik dalam mengambil keputusan berdasarkan data penjualan yang lebih akurat.

2.5 *Marketplace*

Marketplace adalah platform *online* yang mempertemukan penjual dan pembeli dalam ekosistem *online*, seperti model B2C, C2C atau B2B [11]. *Marketplace* memiliki pengaruh positif dan signifikan terhadap pertumbuhan penjualan UMKM [11]. *Marketplace* mempengaruhi bagaimana stok barang dikelola karena transaksi yang terjadi di platform tersebut harus disinkronisasikan dengan stok fisik dan sistem internal toko agar tidak terjadi *overselling* dan kekosongan barang [11,12]. Banyak UMKM yang terbantu oleh *marketplace* untuk memperluas jangkauan pasar dan mengelola produknya secara *online* [12]. *Marketplace* menjadi komponen penting karena data transaksi dari *marketplace* berpotensi dikirim ke sistem persediaan secara *real-time*. Dengan demikian, pemahaman tentang *marketplace* sebagai kanal distribusi, pengaruhnya terhadap stok, dan bagaimana sistem internal harus terhubung dengan *marketplace* [11,12].

Penelitian oleh Rini [13] menunjukkan bahwa penggunaan *marketplace* mampu menjadi mediator antara keunggulan *e-commerce* dan kinerja manajemen rantai pasokan UMKM, termasuk dalam mendukung sinkronisasi stok dan pengendalian persediaan. Hasil penelitian tersebut menunjukkan bahwa integrasi teknologi digital dapat meningkatkan efisiensi proses bisnis UMKM, khususnya dalam pengelolaan distribusi dan informasi persediaan [13].

Penelitian oleh Sutarto, Pranata, dan Rahayuningsih [14] menemukan bahwa transformasi digital pada manajemen persediaan di era *e-commerce* mampu meningkatkan akurasi data stok, mempercepat proses pembaruan stok, serta mengurangi risiko *overstock* maupun kekurangan stok. Penelitian ini menegaskan bahwa penggunaan teknologi digital memiliki peran penting dalam meningkatkan efektivitas pengelolaan persediaan UMKM [14].

Selain itu, Sulistyaningsih, Affiat, dan Hapsari [15] menjelaskan bahwa integrasi antara manajemen gudang dan teknologi *e-commerce* dapat meningkatkan ketersediaan barang serta kepuasan pelanggan pada layanan distribusi barang. Temuan tersebut

menunjukkan bahwa sinkronisasi data antara sistem gudang dan platform digital menjadi faktor penting dalam menjaga kelancaran operasional dan pelayanan pelanggan [15].

Meskipun demikian, penelitian-penelitian sebelumnya masih berfokus pada pengaruh *marketplace* dan transformasi digital terhadap peningkatan kinerja bisnis atau efisiensi pengelolaan persediaan secara umum [13-15]. Penelitian tersebut belum membahas secara spesifik implementasi sistem pengelolaan stok yang mampu melakukan sinkronisasi otomatis dan real-time antara sistem internal UMKM dengan *marketplace* tertentu, seperti Shopee. Selain itu, sebagian penelitian hanya meninjau aspek konseptual tanpa mengembangkan sistem yang secara langsung dapat digunakan untuk mengatasi permasalahan selisih stok pada proses penjualan *online* dan *offline* [13,14].

2.6 *Framework* Flutter

Framework Flutter merupakan *open-source* UI *toolkit* yang dikembangkan oleh Google untuk membangun aplikasi *mobile*, *web*, dan *desktop* menggunakan satu basis kode (*single codebase*) [16]. Dengan pendekatan tersebut, pengembang dapat membangun aplikasi lintas platform tanpa perlu menulis kode yang berbeda untuk setiap sistem operasi, sehingga proses pengembangan dan pemeliharaan aplikasi menjadi lebih efisien [16]. Selain itu, Flutter menyediakan kumpulan *widget* yang memungkinkan antarmuka aplikasi ditampilkan secara konsisten pada berbagai platform [16]. Zindl menunjukkan bahwa Flutter telah berkembang menjadi framework yang layak digunakan untuk pengembangan aplikasi *desktop* Windows, dengan performa yang memadai serta kemudahan dalam membangun antarmuka pengguna menggunakan satu basis kode [16].

Secara arsitektur, Flutter terdiri atas beberapa komponen utama, yaitu Flutter *Framework*, Flutter *Engine*, dan *Embedder* [16]. Flutter *Framework* menyediakan kumpulan *widget*, pengelolaan *state*, navigasi, serta berbagai pustaka yang digunakan dalam pengembangan antarmuka aplikasi menggunakan bahasa Dart [16]. Flutter *Engine* bertanggung jawab terhadap proses rendering grafis, animasi, pengolahan teks, serta komunikasi dengan GPU melalui mesin grafis Skia [16]. Sementara itu, *Embedder* berfungsi sebagai penghubung antara Flutter *Engine* dengan sistem operasi sehingga aplikasi dapat dijalankan secara native pada platform seperti Android, iOS, Windows, macOS, maupun Linux [16].

Cara kerja Flutter berbeda dengan *framework hybrid* yang mengandalkan *WebView* ataupun komponen antarmuka bawaan sistem operasi [17]. Flutter melakukan proses rendering antarmuka menggunakan rendering *engine* miliknya sendiri sehingga

menghasilkan tampilan yang konsisten pada setiap platform [17]. Evaluasi yang dilakukan oleh Białkowski dan Smołka menunjukkan bahwa Flutter memiliki efisiensi yang baik dalam pengembangan *user interface* serta mampu memberikan performa yang kompetitif dibandingkan pengembangan native pada berbagai tugas *interface* [17]. Penelitian tersebut juga menunjukkan bahwa penggunaan satu basis kode dapat mengurangi kompleksitas pengembangan tanpa mengorbankan kualitas tampilan aplikasi [17].

Keunggulan Flutter juga telah dibuktikan melalui implementasi pada berbagai aplikasi nyata [18]. Lovrić et al. mengembangkan aplikasi layanan konsultasi kesehatan menggunakan Flutter dan melakukan evaluasi terhadap *framework* tersebut dari aspek pengembangan lintas platform, kemudahan pemeliharaan, serta pengalaman pengguna [18]. Hasil penelitian menunjukkan bahwa Flutter mampu menghasilkan aplikasi dengan antarmuka yang responsif, konsisten, dan mudah dikembangkan pada berbagai platform melalui penggunaan satu basis kode, sehingga sesuai digunakan untuk aplikasi yang memerlukan interaksi pengguna secara intensif [18].

Meskipun demikian, penelitian-penelitian sebelumnya masih berfokus pada aspek performa *framework*, arsitektur, dan efisiensi pengembangan aplikasi secara umum [16-18]. Penelitian tersebut belum membahas implementasi Flutter secara spesifik pada sistem pengelolaan stok terintegrasi *marketplace* yang membutuhkan sinkronisasi data secara *real-time*, pengelolaan transaksi, *monitoring* stok, dan integrasi API eksternal secara langsung [16,18].

2.7 Supabase

Supabase merupakan platform *Backend as a Service* (BaaS) yang menyediakan berbagai layanan backend, seperti basis data PostgreSQL, autentikasi pengguna, penyimpanan berkas (*storage*), *Edge Functions*, serta layanan Realtime dalam satu platform [19]. Basis data pada Supabase dibangun di atas PostgreSQL sehingga mendukung transaksi relasional, integritas data, serta kemampuan menjalankan perintah SQL secara langsung [19]. Melalui Supabase *Dashboard*, pengembang dapat membuat dan mengelola tabel, relasi antar tabel, *view*, indeks, kebijakan keamanan (*Row Level Security*), maupun fungsi SQL tanpa harus melakukan konfigurasi server secara manual [19]. Selain itu, setiap tabel yang dibuat secara otomatis memiliki REST API sehingga memudahkan integrasi dengan aplikasi eksternal [19].

Pada penelitian ini, Supabase digunakan sebagai layanan *cloud database* untuk menyimpan data pengguna, data produk, data stok, serta riwayat transaksi yang berasal

dari aplikasi *desktop* maupun *mobile* [19,20]. Seluruh data tersimpan secara terpusat sehingga perubahan yang dilakukan pada salah satu perangkat dapat langsung disinkronkan ke basis data [19]. Dengan memanfaatkan PostgreSQL sebagai sistem manajemen basis data, Supabase mampu menyediakan penyimpanan data yang terstruktur, mendukung relasi antar tabel, serta menjaga konsistensi data selama proses transaksi berlangsung [19].

Integrasi Supabase dengan Flutter dilakukan melalui *package* supabase_flutter, yang menyediakan antarmuka untuk menghubungkan aplikasi Flutter dengan layanan Supabase [20]. Proses integrasi diawali dengan inisialisasi proyek menggunakan Project URL dan Publishable Key, kemudian aplikasi dapat memanfaatkan berbagai layanan seperti autentikasi pengguna, operasi Create, Read, Update, Delete (CRUD), penyimpanan berkas, serta Realtime Database melalui satu SDK [20]. Dengan pendekatan tersebut, pengembang dapat mengakses basis data secara langsung dari aplikasi Flutter tanpa perlu membangun layanan backend secara terpisah [19,20].

Dalam penelitian ini, package supabase_flutter dimanfaatkan untuk mengelola autentikasi pengguna serta menyimpan riwayat transaksi dan data stok yang telah disinkronkan dengan API Shopee [20]. Integrasi ini memungkinkan aplikasi *desktop* dan *mobile* menggunakan basis data yang sama sehingga informasi stok, transaksi, maupun data pengguna tetap konsisten pada seluruh platform [19,20]. Selain itu, dukungan Realtime pada Supabase juga memungkinkan aplikasi menerima perubahan data secara langsung ketika terjadi pembaruan pada basis data, sehingga proses sinkronisasi informasi dapat berlangsung lebih cepat dan efisien [19,20].

2.8 Integrasi *Application Programming Interface* (API) Shopee

Integrasi *Application Programming Interface* (API) merupakan komponen penting dalam pengembangan sistem informasi modern karena memungkinkan pertukaran data antarplatform secara otomatis dan *real-time* [21]. API bekerja sebagai antarmuka komunikasi antar sistem sehingga aplikasi dapat saling bertukar data tanpa proses manual [21]. Dalam *marketplace*, API berperan penting untuk mendukung sinkronisasi data transaksi, stok, dan pengiriman secara otomatis sehingga meningkatkan efisiensi dan akurasi operasional [21,22]. Shopee *Open Platform* menyediakan API yang mendukung kebutuhan integrasi sistem *internal* dengan *marketplace* Shopee [21].

Dokumentasi Shopee menjelaskan bahwa API dapat digunakan untuk menghubungkan sistem pihak ketiga seperti sistem manajemen stok, POS, maupun sistem akuntansi dengan layanan Shopee [21]. Selain itu, Shopee menyediakan mekanisme API

calls yang mencakup struktur *request*, parameter autentikasi, serta validasi keamanan menggunakan algoritma HMAC-SHA256 untuk memastikan integrasi berjalan aman dan konsisten [22]. Shopee juga menerapkan mekanisme OAuth 2.0 dalam proses otorisasi dan autentikasi agar akses data penjual tetap aman dan terkontrol melalui penggunaan *access token* dan *refresh token* [23]. Selain itu, mekanisme API *Calls* dan autentikasi yang dijelaskan Shopee [22,23] mendukung proses komunikasi data secara aman dan *real-time* sehingga dapat meminimalkan risiko keterlambatan pembaruan data maupun kesalahan sinkronisasi stok antara *marketplace* dan sistem *internal*.

Pemanfaatan Shopee API pada penelitian ini ditujukan untuk membangun integrasi langsung antara sistem internal toko dan Shopee [21,22], sehingga setiap perubahan transaksi maupun stok dapat diperbarui secara otomatis guna mengeliminasi selisih data stok fisik dan digital.

2.9 *User Acceptance Testing* (UAT)

User Acceptance Testing (UAT) merupakan tahap pengujian perangkat lunak yang dilakukan untuk memastikan bahwa sistem yang dikembangkan telah sesuai dengan kebutuhan pengguna dan dapat diterima untuk digunakan pada lingkungan operasional [24]. UAT berfokus pada validasi fungsi sistem berdasarkan perspektif pengguna akhir (*end-user*) sehingga sistem tidak hanya berjalan secara teknis, tetapi juga mampu mendukung proses bisnis dan kebutuhan operasional pengguna [24,25].

Instrumen pengujian UAT disusun dalam bentuk kuesioner yang mencakup empat variabel evaluasi utama, yaitu fungsionalitas sistem, kinerja sistem, pengalaman dan tampilan antarmuka sistem, serta efisiensi dan produktivitas sistem [25]. Rincian indikator pertanyaan yang digunakan dalam pengujian UAT disajikan pada Tabel 2.1 [25].

Tabel 2.1 Daftar Pertanyaan Kuesioner

No	Variabel	Pertanyaan	(P)
1	**Evaluasi Fungsionalitas Sistem**	Saya dapat menambahkan dan memperbarui data tanpa masalah.	A1
2		Apakah laporan yang dihasilkan oleh sistem ini akurat dan sesuai dengan kebutuhan pengguna.	A2
3		Sistem ini dapat menghasilkan informasi yang lengkap dan dapat diandalkan.	A3

4		Saya dapat mencari dan menemukan data tertentu dalam sistem ini dengan mudah dan sesuai kebutuhan.	A4
5		Fitur pencarian atau *filter* pada sistem ini berfungsi dengan baik dan sangat membantu.	A5
6		Sistem ini dapat memproses masukan dengan benar.	A6
7	**Evaluasi Kinerja Sistem**	Sistem ini selalu tersedia ketika saya membutuhkannya.	B1
8		Kecepatan respon sistem ini sangat baik.	B2
9		Data yang saya masukkan ke dalam sistem selalu tersimpan dengan benar.	B3
10		Sistem ini menyediakan fitur yang cukup untuk memenuhi kebutuhan pekerjaan.	B4
11	**Evaluasi Pengalaman & Tampilan Antarmuka Sistem**	Sistem ini tidak mengalami gangguan saat digunakan.	C1
12		Informasi yang disajikan dalam antarmuka sistem ini mudah dibaca dan dimengerti.	C2
13		Tampilan sistem ini telah memiliki komposisi warna yang sesuai.	C3
14		Informasi yang disajikan sesuai dengan apa yang diharapkan pengguna.	C4
15		Antarmuka pengguna (tampilan) dari sistem ini mudah dipahami.	C5
16		Saya merasa nyaman menggunakan sistem ini dalam aktivitas sehari-hari.	C6
17		Sistem ini membantu meningkatkan efisiensi kinerja saya.	C7
18		Sistem ini mudah di kendalikan (Navigasi) dan sangat responsif (klik, input data, dll.).	C8
19		Tata letak menu dan button pada sistem sudah sesuai.	C9
20	**Evaluasi Efisiensi & Produktivitas Sistem**	Apakah sistem ini membantu mengurangi waktu yang dibutuhkan untuk menyelesaikan tugas.	D1
21		Saya merasa puas dengan fitur otomatisasi yang disediakan oleh sistem ini.	D2
22		Sistem ini membantu dalam mengurangi beban kerja	D3

		manual.	
23		Penggunaan sistem ini mengurangi kesalahan atau yang sering terjadi dalam proses kerja.	D4
24		Sistem ini mempermudah dalam meningkatkan kolaborasi antar rekan kerja tim.	D5
25		Dengan sistem ini, proses pengambilan keputusan menjadi lebih cepat dan akurat.	D6

Penelitian oleh Puntadewa dkk. [26] menunjukkan bahwa metode UAT digunakan untuk mengukur tingkat penerimaan pengguna terhadap sistem melalui penyebaran kuesioner berbasis skala Likert. Pengujian dilakukan dengan meminta pengguna mencoba fitur-fitur utama sistem, kemudian memberikan penilaian terhadap kemudahan penggunaan, kesesuaian fungsi, dan kepuasan pengguna terhadap sistem yang dikembangkan. Hasil pengujian kemudian dihitung dalam bentuk persentase untuk menentukan tingkat kelayakan sistem [26]. Perhitungan nilai UAT dilakukan menggunakan rumus sebagai berikut [26]:

$$P = \frac{f}{N} x\ 100\%$$ Rumus Persentase Nilai UAT (2.1)

Keterangan:

P = Skor persentase yang dicari

f = Perolehan skor oleh validator

N = Skor maksimal

Hasil persentase kemudian diinterpretasikan berdasarkan kategori kelayakan sistem seperti pada Tabel 2.2 [26].

Tabel 2.2 Kategori Kelayakan Sistem Berdasarkan Nilai UAT

Persentase	Kategori
81% – 100%	Sangat Baik
61% – 80%	Baik
41% – 60%	Cukup
21% – 40%	Kurang
0% – 20%	Sangat Kurang

Meskipun demikian, penelitian sebelumnya [24-26] masih berfokus pada pengukuran tingkat penerimaan pengguna terhadap aplikasi umum dan belum banyak membahas implementasi UAT pada sistem pengelolaan stok terintegrasi *marketplace* yang memiliki sinkronisasi data secara *real-time* antara sistem internal dan platform *e-commerce*. Selain itu, penelitian sebelumnya lebih menitikberatkan pada aspek *usability* secara umum dan belum secara spesifik mengevaluasi kesesuaian fitur sinkronisasi stok otomatis terhadap kebutuhan operasional UMKM [25,26].

2.10 *Extreme Programming* (XP)

Extreme Programming (XP) merupakan salah satu metodologi pengembangan perangkat lunak yang termasuk dalam kategori *agile software development* [27]. Pendekatan ini sangat menekankan pada aspek teknis pengembangan, terutama pada pengkodean *(coding*) yang menjadi aktivitas utama dalam seluruh siklus hidup pengembangan sistem [27]. Karakteristik utama dari XP adalah kemampuannya untuk bersikap responsif terhadap perubahan kebutuhan pengguna yang dinamis, fleksibilitas tinggi, serta penguatan komunikasi yang intensif antara tim pengembang dan pemangku kepentingan [27,28]. Metode ini dirancang untuk menciptakan proses pengembangan yang adaptif, efisien, dan fokus pada kualitas melalui iterasi yang pendek serta umpan balik yang cepat [27,28]. Dengan mengutamakan prinsip kesederhanaan, XP memungkinkan tim untuk menghadapi tantangan teknis secara inovatif tanpa terjebak dalam dokumentasi yang berlebihan, sehingga mempercepat proses rilis produk ke tangan pengguna [28].

Berdasarkan standar pengembangan yang umum digunakan, tahapan dalam metode *Extreme Programming* terdiri dari empat fase utama [27,28]:

1. *Planning* (Perencanaan)

   Tahap perencanaan merupakan fondasi awal untuk memahami konteks bisnis dari aplikasi yang akan dibangun [27]. Pengembang dan pengguna berkolaborasi untuk mendefinisikan output, fitur, fungsionalitas, serta alur pengembangan aplikasi secara keseluruhan [27,28]. Tujuannya adalah menyelaraskan pemahaman agar risiko kesalahan di tahap berikutnya dapat diminimalkan [27,28].

   Contoh: Pembuatan *User Stories*. Pengguna menuliskan kebutuhan mereka dalam bentuk cerita pendek (misalnya: "Sebagai mahasiswa, saya ingin melihat jadwal orientasi agar tidak tertinggal informasi"). *User stories* ini kemudian dianalisis oleh pengembang untuk menentukan estimasi waktu dan prioritas fitur yang akan dikerjakan terlebih dahulu [27,28].

2. *Design* (Perancangan)

Pada tahap ini, hasil analisis kebutuhan diterjemahkan ke dalam pemodelan sistem yang sederhana [27]. Fokus utama XP dalam perancangan adalah menjaga desain tetap ringkas namun mampu menggambarkan hubungan antar data dan logika sistem secara jelas. Alat bantu desain digunakan untuk memetakan struktur aplikasi sebelum masuk ke tahap teknis [27,28].

Contoh: Penggunaan *Unified Modelling Language* (UML). Tim membuat *Use Case Diagram* untuk menggambarkan interaksi pengguna dengan sistem, *Activity Diagram* untuk memetakan alur kerja aplikasi, dan *Class Diagram* untuk mendefinisikan struktur data. Selain itu, penggunaan kartu CRC (*Class Responsibility Collaborator*) sering diterapkan untuk memetakan kelas-kelas yang akan dibangun dalam sistem [27,28].

3. *Coding* (Pengkodean)

Pengkodean adalah inti dari metode XP [27]. Semua rancangan yang telah dibuat diimplementasikan ke dalam bahasa pemrograman yang dapat dikenali oleh komputer [27,28]. Tahap ini bukan sekadar menulis baris kode, melainkan proses teknis untuk menciptakan solusi konkret atas masalah yang dihadapi pengguna [27].

Contoh: Implementasi menggunakan stack teknologi modern seperti bahasa pemrograman JavaScript dengan *framework* React (Next.js) dan *library* Tailwind CSS untuk antarmuka. Untuk pengelolaan data, pengembang bisa mengintegrasikannya dengan layanan *Cloud Storage* seperti Firebase agar proses pengembangan lebih efisien dan terstruktur [28].

4. *Testing* (Pengujian)

Tahapan terakhir ini bertujuan untuk memastikan bahwa layanan, fitur, dan fungsionalitas aplikasi berjalan sesuai dengan yang direncanakan [27]. Pengujian dilakukan secara berkala dan sering (seringkali dilakukan bersamaan dengan pengkodean) untuk menemukan bug sedini mungkin dan memastikan kepuasan pengguna akhir [27,28].

Contoh: Penerapan teknik *Black-box Testing*. Penguji melakukan simulasi input pada antarmuka aplikasi tanpa melihat kode internalnya. Misalnya, penguji mencoba mengisi formulir pendaftaran mahasiswa dan menekan tombol "Kirim" untuk memastikan data tersimpan dengan benar di *database* dan aplikasi

memberikan respon yang sesuai. Jika terjadi kesalahan pada antarmuka, tim akan segera melakukan perbaikan kembali ke tahap coding [28].

Meskipun demikian, penelitian-penelitian sebelumnya [27,28] masih berfokus pada pengembangan aplikasi umum seperti sistem informasi organisasi dan aplikasi karier. Penelitian tersebut belum membahas penerapan metode *Extreme Programming* pada pengembangan sistem pengelolaan stok terintegrasi *marketplace* yang memiliki kebutuhan sinkronisasi data secara *real-time*, integrasi API eksternal, dan pengelolaan transaksi *online* maupun *offline* secara bersamaan.