JIK: Jurnal Ilmu Komputer
Vol. 08, No. 01, Juni, 2023 p-ISSN: 2527-9653
DOI: 10.47007/komp.v%vi%i.6711 e-ISSN: 2797-8389

IMPLEMENTASI MIDDLEWARE UNTUK KEBUTUHAN BRIDGING DATA TRANSAKSIONAL ERP DAN MARKETPLACE

Lukas Tantyo[1], Tri Ismardiko Widyawan[*2]

[1,2]Program Studi Teknik Informatika, Fakultas Ilmu Komputer, Universitas Esa Unggul Jakarta
Jl. Arjuna Utara No. 9 Kb. Jeruk, Jakarta Barat, 11510
Email: [*]ismardiko@esaunggul.ac.id

(Diserahkan: 08-06-2023, Artikel *Review*: 11-08-2023, Diterima: 17-08-2023, Diterbitkan: 20-08-2023)

Abstrak

PT. XYZ menggunakan *Shopee* untuk menjual produknya secara *online* dan *ERPNext* sebagai sistem informasi untuk melacak setiap transaksi penjualan. Akan tetapi PT. XYZ menjadi terbebani dengan proses pencatatan saat menangani volume transaksi penjualan yang dapat mencapai hingga 20.000 transaksi per hari, yang meningkatkan risiko kesalahan manusia. Proses perekaman data pada akhirnya akan menemui kendala dan kesalahan. Untuk menyiasatinya, peneliti akan melakukan diskusi dengan pengguna serta analisa referensi penelitian sebelumnya dan menggunakan metode *Rapid Application Development* untuk merancang *middleware* yang mengintegrasi *Marketplace* dengan *ERP* menggunakan *Frappe Framework*. Kemudahan pencatatan transaksi penjualan dan peningkatan akurasi data akan dimungkinkan oleh *middleware* yang dirancang dari penelitian ini. Selain itu, penggunaan *middleware* akan mengurangi risiko kesalahan manusia dan mempersingkat proses pencatatan. Ini akan memastikan bahwa data yang akurat dan andal tersedia untuk tujuan analisis dan pengambilan keputusan. Selain itu, *middleware* juga dapat menghemat waktu dan meningkatkan efisiensi dengan meniadakan kebutuhan entri data manual, memungkinkan karyawan untuk fokus pada tugas penting lainnya. Secara keseluruhan, menerapkan sistem ini dapat memberikan banyak manfaat bagi perusahaan, termasuk peningkatan akurasi, efisiensi, dan produktivitas. Hasil pengujian menggunakan *blackbox testing* dan *load testing* seperti yang diharapkan. Dari hasil tersebut diketahui bahwa *middleware* yang dikembangkan layak untuk digunakan dan berhasil mengatasi kendala yang dihadapi oleh PT. XYZ.

Kata kunci: *ERPNext, Marketplace, Integrasi, Middleware, Frappe Framework*

MIDDLEWARE IMPLEMENTATION FOR BRIDGING ERP AND MARKETPLACE TRANSACTIONAL DATA

Abstract

PT. XYZ uses Shopee to sell its products online and ERPNext as an information system to track every sales transaction. However, PT. XYZ becomes burdened with the record-keeping process when handling sales transaction volume, which can reach up to 20,000 transactions per day, which increases the risk of human error. The data recording process will eventually encounter problems and errors. To work around this, researchers will conduct discussions with users as well as analyze references to previous research and use the Rapid Application Development method to design middleware that integrates the Marketplace with ERP using the Frappe Framework. The ease of recording sales transactions and increasing data accuracy will be made possible by the middleware developed from this research. In addition, the use of middleware reduces the risk of human error and streamlines the logging process. This will ensure that accurate and reliable data is available for analysis and decision-making purposes. Furthermore, by removing the requirement for manual data entry, middleware can save time and improve efficiency, allowing employees to focus on other important tasks. Overall, implementing these systems can provide a company with many benefits, including increased accuracy, efficiency, and productivity. Test results using black box testing and load testing were as expected. These findings demonstrate the viability of using the designed middleware and have succeeded in overcoming the obstacles faced by PT. XYZ.

Keywords: *ERPNext, Marketplace, Integration, Middleware, Frappe Framework*

PENDAHULUAN

Kemajuan teknologi membuat banyak pelaku bisnis yang kini memanfaatkan teknologi dengan menjual barangnya di *e-commerce* dan menggunakan sistem informasi untuk melacak setiap transaksi penjualan. Hal tersebut juga dimanfaatkan oleh PT. XYZ dengan menggunakan *Shopee* dan *ERPNext*. *Shopee* adalah salah satu *e-commerce* berbasis *marketplace* terbesar yang beroperasi di Indonesia [1]. Dan *ERPNext* adalah sistem informasi web yang banyak digunakan oleh operasi bisnis kecil dan menengah. *ERPNext* merupakan sistem informasi *ERP* yang dikembangkan oleh *Frappe Technologies Pvt Ltd. ERPNext* menawarkan alat manajemen untuk bisnis di bidang akuntansi, manajemen proyek, penjualan, pembelian, inventaris, manajemen hubungan pelanggan, sumber daya manusia, dan produksi [2][3].

Namun, ketika sebuah perusahaan masih secara manual memasukkan setiap data transaksi penjualan ke dalam sistem *ERP* dengan volume yang tinggi. Prosedur ini bisa melelahkan dan rawan kesalahan, serta dapat meningkatkan biaya operasional [4]. Hal tersebut juga dialami oleh PT. XYZ, sehingga untuk menjamin pencatatan yang akurat dan membantu bisnis dalam mempertahankan gambaran yang jelas tentang kinerja penjualannya serta untuk meningkatkan efisiensi proses penjualan, maka perusahaan diharuskan untuk menerapkan sistem yang sepenuhnya otomatis. Karena integrasi menghilangkan persyaratan untuk entri ulang data. Selain itu, dapat mengurangi duplikasi informasi, kesalahan manusia, dan keterlibatan manusia [2].

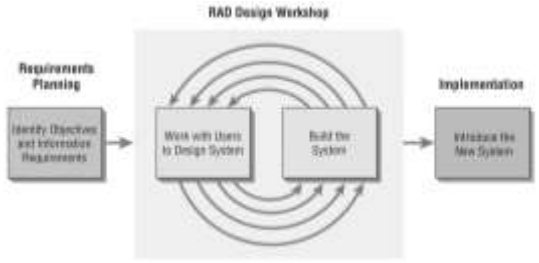
Berdasarkan penelitian sebelumnya yang dilakukan pada integrasi *e-commerce* dan *ERP*. Bahwa integrasi yang efisien dapat meningkatkan kapasitas perusahaan dalam adaptasi terhadap kondisi pasar yang berubah-ubah, memuaskan keinginan pelanggan dan meningkatkan pendapatan dalam skenario pasar ekonomi saat ini [5]. Integrasi dengan memanfaatkan konektor *middleware* yang dapat menerima informasi dan memperbarui informasi, memungkinkan perusahaan untuk mengintegrasikan sistem penjualan dengan *ERP* [6][7]. Pengembangan integrasi *ERP* dengan *e-commerce* dapat dilakukan dengan mengimplementasikan *REST API* dan *Webhook* [8][9][10]. *REST API* berfungsi untuk berkomunikasi antar sistem dan sistem dapat menerima informasi secara instan menggunakan *Webhook* sehingga dapat mengingkatkan keefektifan pembaruan data antara sistem [11][12].

Berdasarkan informasi yang diberikan di atas, maka dilakukan penelitian untuk membuat sistem *middleware* untuk integrasi pencatatan data penjualan *Shopee* dengan sistem *ERPNext*. Pembangunan *middleware* akan dibangun menggunakan *Frappe Framework*, sebuah *framework* yang digunakan juga oleh *ERPNext*. Tujuan penggunaan *framework* yang sama adalah supaya perusahaan dapat menggunakan data secara terintegrasi tanpa mengorbankan keunggulan *ERP* karena rancangannya akan berfungsi sebagai komponen dari *ERP* [13]. Dengan dibangunnya sistem ini diharapkan proses pencatatan penjualan *Shopee* dapat dilakukan dengan benar dan akurat.

METODE PENELITIAN

Metode penelitian yang diimplementasikan dalam penelitian ini yaitu *Rapid Application Development* (RAD). *RAD* adalah metode untuk mengembangkan perangkat lunak yang menggunakan teknik pengembangan sistem yang dapat diterapkan dengan cepat yang dapat dimodifikasi seiring kemajuan proyek [14].

Menurut [15], *RAD* dibagi menjadi 3 (tiga) fase yakni fase perencanaan kebutuhan, fase perancangan *workshop RAD*, dan fase implementasi, seperti yang terlihat pada Gambar 1.



Gambar 1. *Rapid Application Development* (RAD)

Fase Perencanaan Kebutuhan

Berdasarkan penelitian pada referensi, diskusi dengan pengguna, dan temuan dari hasil diskusi tersebut, kebutuhan yang diperlukan akan direncanakan selama fase ini. Informasi dan prosedur yang akan digunakan sistem terdapat dalam persyaratan fungsional dan non-fungsional. Tabel 1 berikut ini adalah persyaratan fungsional.

Tabel 1. Persyaratan Fungsional

Kode	Kebutuhan	Keterangan
F001	*Create Sales Order*	Pembuatan *Sales Order* pada status pesanan "*Processed*"
F002	*Cancel Sales Order*	Pembuatan *Sales Order* pada status pesanan "*Cancelled*"
F003	*Create Delivery Note*	Pembuatan *Sales Order* pada status pesanan "*Shipped*"
F004	*Create Sales Invoice*	Pembuatan *Sales Order* pada status pesanan "*Completed*"
F005	*Create Sales Invoice Return*	Pembuatan *Sales Order* pada status pesanan "*Return*"
F006	*Create Item*	Pembuatan *Item* pada *Shopee*
F007	*Update Stock*	Pembaruan stok pada *Shopee*

Sedangkan pada persyaratan non-fungsional, peneliti akan fokus pada sifat perilaku sistem. Tabel 2 berikut ini adalah persyaratan non-fungsional.

Tabel 2. Persyaratan Non-fungsional

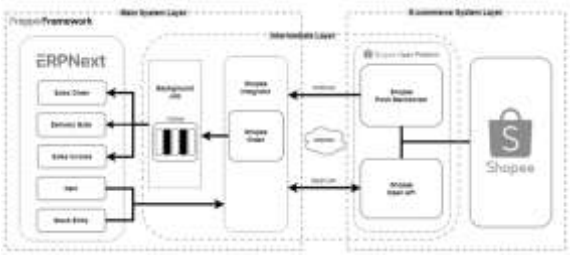
Kode	Kebutuhan	Keterangan
NF001	Peforma	Sistem mampu memproses data transaksi dengan volume tinggi dan tetap mempertahankan kinerjanya

Berdasarkan persyaratan-persyaratan yang ada maka peneliti menyimpulkan bahwa *middleware* akan dikembangkan dengan menggunakan *Frappe Framework*, sebuah *framework* yang digunakan oleh sistem informasi yang sedang digunakan yaitu *ERPNext*. Langkah selanjutnya adalah menghubungkan *Shopee* dengan *ERPNext* menggunakan *custom app* bernama *Shopee Integrator* sehingga peneliti dapat menambahkan fungsionalitas yang diperlukan tanpa harus mengubah kode utama *ERPNext*.

Untuk membuat *Sales Order*, *Delivery Note*, dan *Sales Invoice*, peneliti akan melakukan integrasi dengan memanfaatkan teknologi *Webhook* untuk mengumpulkan data transaksi dari *Shopee* secara *real time*. Peneliti juga menggunakan teknologi *REST API* untuk mengirimkan data produk dan stok ke *Shopee*. Menurut [16], penelitian yang dilakukan telah membuktikan dengan menggunakan teknologi *Message Queue* dapat meningkatkan kinerja sistem dalam pemrosesan data yang besar. Sehingga peneliti memutuskan untuk menggunakan teknologi *Message Queue* dalam mengontrol data yang akan disimpan ke dalam database untuk mempertahankan kinerja *ERPNext* dalam menghadapi kenaikan volume transaksi yang disebabkan oleh proses integrasi.

Fase Rancangan Workshop RAD

Peneliti kemudian merancang sistem berdasarkan hasil dari tahap perencanaan kebutuhan. Peneliti mengilustrasikan arsitektur yang diajukan untuk merepresentasikan komponen-komponen dari *middleware* yang dirancang, arsitektur *middleware* dapat dilihat pada gambar 2.



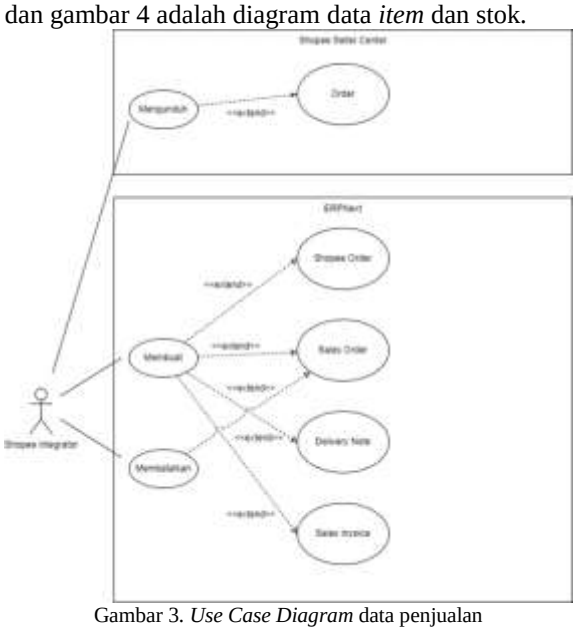
Gambar 2. Arsitektur *Middleware*

Pada gambar 2 menjelaskan 3 (tiga) lapisan dasar rancangan *middleware* yang terdiri dari lapisan sistem utama (*Main System Layer*), lapisan tengah (*Intermediate Layer*), dan lapisan sistem *e-commerce* (*E-commerce System Layer*).

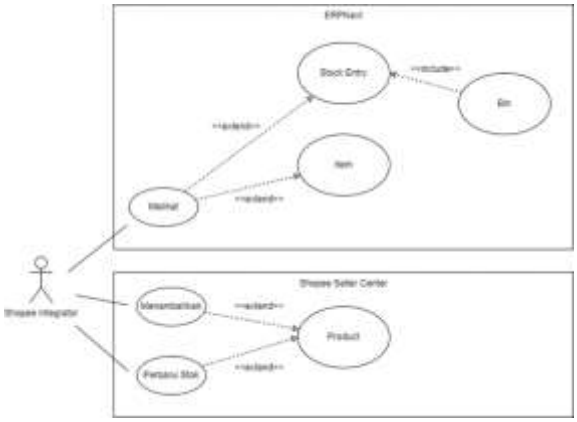
Pada lapisan sistem utama terdapat aplikasi dan *doctype* yang terdiri dari *ERPNext*, *Background Job*, dan *custom apps Shopee Integrator*. Kemudian pada lapisan tengah terdapat *Background Job*, *custom apps Shopee Integrator* dan modul *Shopee Open Platforms*. Dan lapisan yang terakhir yaitu lapisan sistem *e-commerce* merupakan lapisan terdiri dari *Shopee Open Platform* dengan modul *Shopee Push Mechanism* dan *Shopee Open API*.

Komponen sistem perangkat lunak divisualisasikan, didefinisikan, dibangun, dan didokumentasikan menggunakan bahasa grafis yang dikenal sebagai *Unified Modelling Language* (UML). *UML* yang dipergunakan adalah *activity diagram*, *use case diagram*, dan *class diagram* [17].

Kegiatan peran *middleware* pada sistem ini divisualisasikan dengan *use case diagram* yang terlihat pada gambar 3 adalah diagram data penjualan dan gambar 4 adalah diagram data *item* dan stok.



Gambar 3. *Use Case Diagram* data penjualan



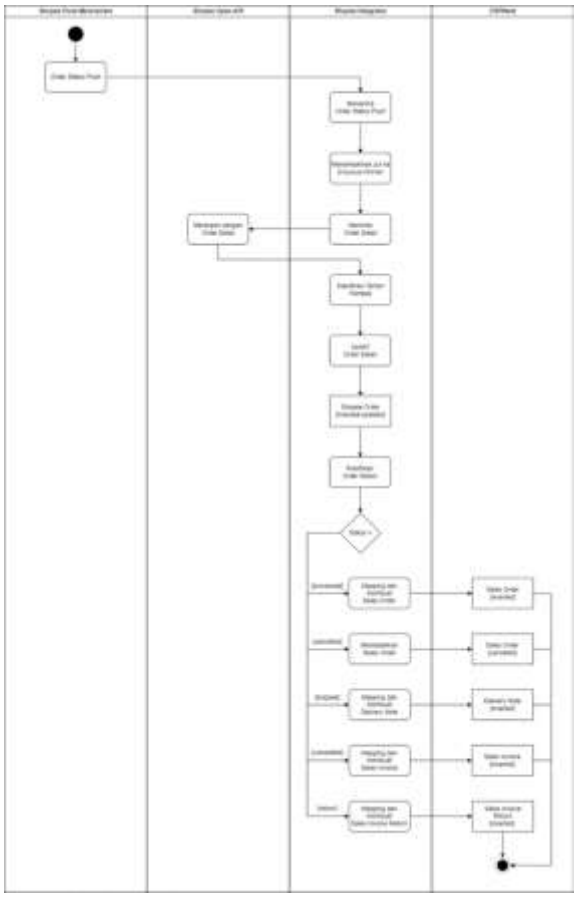
Gambar 4. *Use Case Diagram* data *Item* dan stok

Berdasarkan *use case diagram* pada gambar 3 dan 4, dapat dipahami bahwa:

a) 2 (dua) sistem yang mencakup seluruh sistem yang digunakan pada PT. XYZ yaitu: *Shopee* dan *ERPNext*.

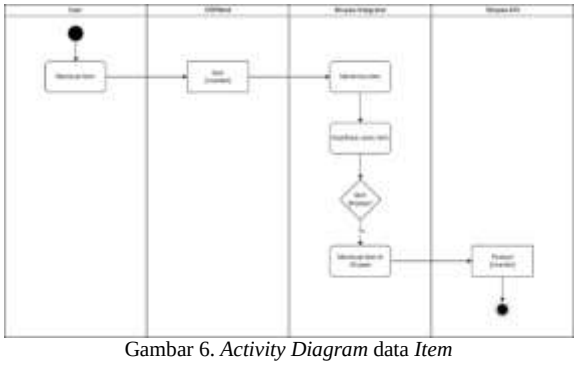
b) 1 (satu) aktor yang terlibat dalam kegiatan berikut yaitu: *Shopee Integrator* (*middleware*).
c) Aktor tersebut menjalankan 10 (sepuluh) skenario penggunaan, beberapa di antaranya adalah: mengunduh data penjualan, membuat *Sales Order*, membuat *Delivery Note*, membuat *Sales Invoice*, membatalkan *Sales Order*, melihat *Stock Entry*, melihat *Item*, melihat *Bin*, menambahkan produk, dan perbarui stok produk.

Sedangkan aktivitas *middleware* pada sistem ini divisualisasikan dengan *activity diagram.* Aktivitas *middleware* dibagi menjadi 3 (tiga) bagian. Gambar 5 menggambarkan aktivitas data penjualan, gambar 6 untuk aktivitas data *item* dan gambar 7 untuk aktivitas data stok.



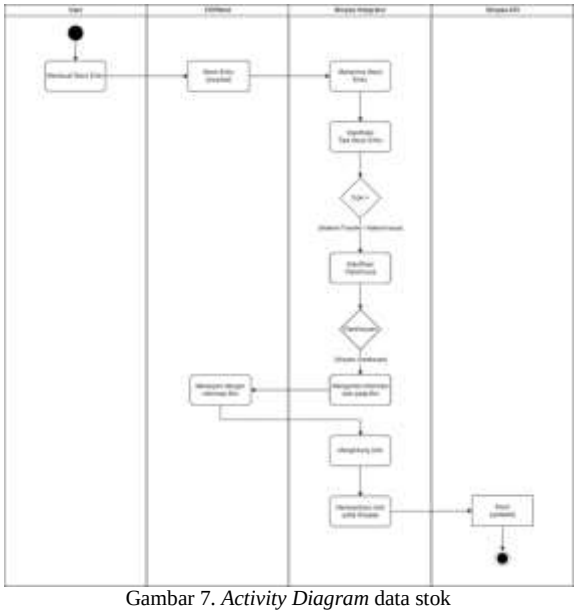
Gambar 5. *Activity Diagram* data penjualan

Pada gambar 5 menjelaskan *activity diagram* data penjualan yang dimulai dengan penerimaan data *webhook* serta penarikan data dari *Shopee* dan akan diklasifikasi oleh *Shopee Integrator* dalam pembuatan data penjualan pada *ERPNext* berdasarkan status pesanan yaitu: *Processed* akan membuat *Sales Order*, *Cancelled* akan membatalkan *Sales Order*, *Shipped* akan membuat *Delivery Note*, *Completed* akan membuat *Sales Invoice*, *Return* akan membuat *Sales Invoice Return*.



Gambar 6. *Activity Diagram* data *Item*

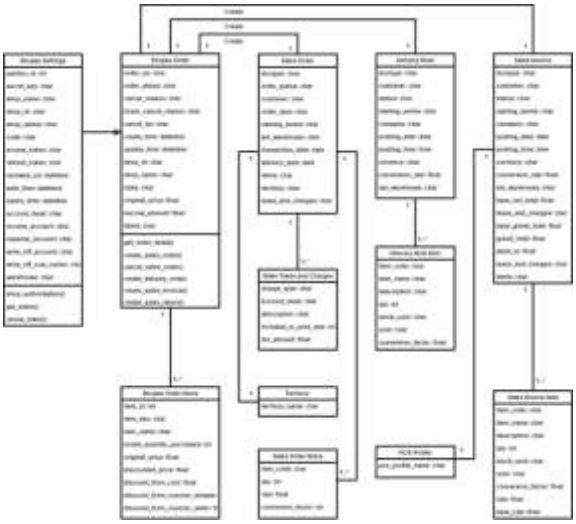
Pada gambar 6 menjelaskan *activity diagram* data *item* yang dimulai dengan pembuatan data *Item* pada *ERPNext* dan akan diklasifikasi oleh *Shopee Integrator* untuk melakukan pembuatan produk pada *Shopee* yaitu ketika *item* tersebut adalah *item Shopee.*



Gambar 7. *Activity Diagram* data stok

Pada gambar 7 menjelaskan *activity diagram* data stok yang dimulai dengan pembuatan *Stock Entry* pada *ERPNext* dan akan diklasifikasi oleh *Shopee Integrator* untuk melakukan pembaruan stok produk pada *Shopee* yaitu ketika *warehouse* yang digunakan adalah *Shopee Warehouse*.

Untuk menggambarkan atribut, kelas, operasi, dan hubungan *middleware* divisualisasikan dalam dua bentuk *class diagram* seperti pada gambar 8 untuk data penjualan, sedangkan gambar 9 untuk data *item* dan stok.



Gambar 8. *Class Diagram* data penjualan

Pada gambar 8 dan 9 menjelaskan *class diagram* data penjualan dan data stok. Terdapat 11 (sebelas) entitas yang digunakan dalam *middleware* ini yaitu *Shopee Settings*, *Shopee Order*, *Sales Order*, *Sales Taxes and Charges*, *Sales Order Items*, *Territory*, *Delivery Note*, *Delivery Note Item*, *Sales Invoice*, *POS Profile*, dan *Sales Invoice Item*.

Gambar 9. *Class Diagram* data *item* dan stok

Pada fase ini juga dirancang tampilan *shopee settings.* Halaman ini berfungsi sebagai tempat untuk konfigurasi *Shopee Integrator* seperti yang terlihat pada gambar 10. Kemudian rancangan tampilan *Shopee Order*. Pada halaman ini terdapat daftar pesanan yang didapat dari *Shopee* seperti yang terlihat pada gambar 11.

Gambar 10. *Mockup* tampilan *Shopee Settings*

Pada gambar 10 menjelaskan pada halaman *Shopee Settings* mempunyai isian *Partner Id* dan *Secret Key.* Serta menampilkan informasi *Shop Name, Shop ID, Shop Status, Access Token, Refresh Token,* dan *Expired on.*

Gambar 11. *Mockup* tampilan *Shopee Order*

Pada gambar 11 menjelaskan pada halaman *Shopee Order* terdapat tampilan informasi *Order No, Order Status, Create Time, Buyer Territory, Payment Method,* dan *Total Amount.* Serta tabel berisikan informasi *Item Name, Item SKU, Item Quantity,* dan *Item Price*.

Fase Implementasi

Pada fase implementasi peneliti akan melakukan implementasi yang terdiri dari dua bagian yaitu konstruksi dan pengujian. Pada bagian konstruksi terdapat fase yang dibagi menjadi beberapa tahapan yaitu pembuatan *custom app Shopee Integrator*, pembuatan tampilan *Shopee Integrator*, implementasi *REST API*, implementasi *Webhook*, dan implementasi *Message Queue*.

Proses pembuatan *custom app* pada *Frappe Framework* dapat dilihat pada gambar 12. Proses pembuatan tampilan *Shopee Settings* dapat dilihat pada gambar 13, *Shopee Order Items* pada gambar 14, dan *Shopee Order* pada gambar 15.

Gambar 12. Perintah pembuatan *custom app*

Pada gambar 12 menjelaskan proses pembuatan *custom app* menggunakan *Frappe Framework*. Pembuatan dapat dilakukan dengan menjalankan perintah *bench new-app shopee_integrator* dan mengisi informasi *custom app*.

Gambar 13. Proses pembuatan tampilan *Shopee Settings*

Pada gambar 13 menjelaskan proses pembuatan isian pada *Shopee Settings* menggunakan *Frappe Framework.* Berikut adalah label dan tipe data isian yang digunakan yaitu: *Partner Id* (*integer*), *Secret Key* (*password*), *Shop Name* (*data*), *Shop ID* (*data*), *Shop Status* (*data*), *Access Token* (*password*), *Refresh Token* (*password*), dan *Expired on* (*datetime*).

Gambar 14. Proses pembuatan tampilan *Shopee Order Items*

Pada gambar 14 menjelaskan proses pembuatan isian pada *Shopee Order Items* menggunakan *Frappe Framework.* Berikut adalah label dan tipe data isian yang digunakan yaitu: *Item Name* (*data*), *Item SKU* (*data*), *Item Quantity* (*integer*), dan *Item Price* (*float*).

Gambar 15. Proses pembuatan tampilan *Shopee Order*

Pada gambar 15 menjelaskan proses pembuatan isian pada *Shopee Order* menggunakan *Frappe Framework.* Berikut adalah label dan tipe data isian yang digunakan yaitu: *Order No* (*data*), *Order Status* (*data*), *Create Time* (*datetime*), *Buyer Territory* (*data*), *Payment Method* (*data*), dan *Total Amount* (*float*).

Kemudian implementasi pengkodean *Webhook* pada *Frappe Framework* dapat dilihat pada gambar 16, pengkodean *Message Queue* pada *Frappe Framework* dapat dilihat pada gambar 17, pengkodean *REST API* pada *Frappe Framework* dapat dilihat pada gambar 18.

Gambar 16. Pengkodean *Webhook*

Pada gambar 16 adalah pengkodean *python yang* memanfaatkan *Remote Procedure Calls* (RPC) dari *Frappe Framework* untuk menerima kiriman data *webhook* dari *Shopee.*

Gambar 17. Pengkodean *Message Queue*

Pada gambar 17 adalah pengkodean *python Message Queue* yang memanfaatkan *Background Jobs* dari *Frappe Framework* untuk meregulasi antrean pembuatan data penjualan.

Gambar 18. Pengkodean *REST API*

Pada gambar 18 adalah pengkodean python yang mengunakan *library request* untuk melakukan penarikan dan pengiriman data dari/ke *endpoint* yang disediakan oleh *Shopee*.

Kemudian pengujian dan penilaian sistem akan dilakukan setelah tahap konstruksi selesai. Pendekatan *blackbox testing* dan *load testing* digunakan selama pengujian untuk memastikan sistem dibangun sesuai dengan kriteria fungsional dan non-fungsional.

HASIL DAN PEMBAHASAN

Dengan adanya *middleware* ini, perusahaan diharapkan dapat terbantu dengan terintegrasinya *ERPNext* dan *Shopee*. Sehingga tidak perlu melakukan pencatatan secara manual serta dapat memantau pergerakan stok secara *realtime.* Tentunya hal ini akan mengurangi risiko kesalahan manusia dan mempersingkat proses pencatatan.

Hasil Implementasi

Hasil implementasi yang dilakukan pada penelitian ini adalah *middleware*. Fitur sistem ini dibagi menjadi 2 (dua) bagian, yaitu pencatatan data penjualan dari *Shopee* dan pengiriman informasi *item* dan stok kepada *Shopee*. Terdapat 2 tampilan pada *middleware* ini yaitu halaman *Shopee Settings* seperti yang terlihat pada gambar 19 dan halaman *Shopee Order* yang terlihat pada gambar 20.

Gambar 19. Halaman *Shopee Settings*

Pada gambar 19 menunjukkan halaman *Shopee Settings* yang tedapat isian *Partner ID* yang sudah terisi dengan nomor id *API* dan isian *Secret Key* yang sudah terisi dengan kredensial *API*. Pada bagian bawah juga terdapat informasi tambahan yang didapat setelah otorisasi *API* berhasil berupa *Shop Name, Shop ID, Shop Status, Access Token, Refresh Token*, dan *Expired on*.

Gambar 20. Halaman *Shopee Order*

Pada gambar 20 menunjukkan halaman *Shopee Order* yang tedapat isian *Order Status* yang sudah terisi dengan status pesanan, isian *State* yang sudah terisi dengan nama daerah, isian *Create Time* yang sudah terisi dengan tanggal terjadinya pesanan, isian *Update Time* yang sudah terisi terisi dengan tanggal pesanan diperbarui, dan juga tabel yang terisi dengan informasi barang pesanan.

Gambar 21. *Sales Order* yang terbuat

Pada gambar 21 menunjukkan *Sales Order* yang berhasil terbuat secara otomatis dan berisikan data pesanan yang sudah sesuai dengan informasi pada *Shopee Order*.

Gambar 22. *Delivery Note* yang terbuat

Pada gambar 22 menunjukkan *Delivery Order* yang berhasil terbuat secara otomatis dan berisikan data pesanan yang sudah sesuai dengan informasi pada *Shopee Order*.

Gambar 23. *Sales Invoice* yang terbuat

Pada gambar 23 menunjukkan *Sales Invoice* yang berhasil terbuat secara otomatis dan berisikan data pesanan yang sudah sesuai dengan informasi pada *Shopee Order*.

Sedangkan untuk data *item*, *middleware* akan mengirimkan informasi *item* ke *Shopee* pada saat *item* di *ERPNext* dibuat. Kemudian pada data stok, *middleware* akan mengirimkan informasi stok ke *Shopee* saat terjadi pergerakan stok pada *ERPNext*.

Gambar 24. *Item* yang terbuat

Pada gambar 24 menunjukkan *item* yang berhasil terbuat di *Shopee* sesuai data pada *Item ERPNext*.

Gambar 25. Stok yang terbarui

Pada gambar 25 menunjukkan stok yang berhasil terbarui di *Shopee* sesuai jumlah stok pada *ERPNext.*

Pengujian

Setelah sistem *middleware* selesai dikembangkan, selanjutnya akan dilakukan proses pengujian untuk kelayakan. Peneliti melakukan pengujian dengan metode *blackbox* dan *load testing*. Bertujuan untuk menjamin fungsionalitas dari sistem yang dirancang, dilakukan pengujian *blackbox*. Metode *blackbox*, yang digunakan untuk menemukan masalah dan hanya berkonsentrasi pada persyaratan fungsional perangkat lunak [18].

Pengujian dilakukan pada fitur pembuatan data penjualan secara otomatis pada *ERPNext* serta pembuatan *item* dan pembaruan jumlah stok otomatis pada *Shopee*. Pengujian dilakukan bersama pengguna kedua sistem tersebut yaitu admin penjualan dan admin toko *online* dengan skenario yang telah ditentukan. Hasil pengujian menggunakan *blackbox testing* dapat dilihat pada tabel 3.

Tabel 3. Hasil dari pengujian menggunakan *Blackbox Testing*

Fitur	Kesimpulan
Pembuatan *Sales Order* pada status pesanan "*Processed*"	Berhasil
Pembuatan *Sales Order* pada status pesanan "*Cancelled*"	Berhasil
Pembuatan *Sales Order* pada status pesanan "*Shipped*"	Berhasil
Pembuatan *Sales Order* pada status pesanan "*Completed*"	Berhasil
Pembuatan *Sales Order* pada status pesanan "*Return*"	Berhasil
Pembuatan *Item* pada *Shopee*	Berhasil
Pembaruan jumlah stok pada *Shopee*	Berhasil

Hasil pengujian *blackbox testing* pada tabel 3 menunjukkan bahwa tidak diperlukan perbaikan karena fungsionalitas *middleware* telah sepenuhnya direalisasikan.

Sementara itu, pengujian beban akan digunakan dengan *Apache JMeter* untuk melakukan pengujian non-fungsionalitas. *Apache JMeter* adalah aplikasi yang dikembangkan menggunakan *Java* yang secara umum digunakan untuk mensimulasikan beban berat pada server, jaringan, *cluster server*, dan objek lainnya serta menilai daya tahan atau kinerja objek secara keseluruhan di bawah berbagai beban [19].

Pengujian ini dilakukan untuk mengetahui apakah implementasi *message queue* dapat mempercepat waktu respon sistem *ERPNext* dalam pembuatan data penjualan setelah melakukan proses integrasi menggunakan *middleware*. Hasil pengujian menggunakan *load testing* dapat dilihat pada tabel 4.

Tabel 4. Hasil dari pengujian menggunakan *Load Testing*

Pengujian	Waktu Respon Tanpa *Message Queue*	Waktu Respon Dengan *Message Queue*
Pembuatan *Sales Invoice* oleh *Middleware*	6,485 detik	3,018 detik
Pembuatan *Sales Invoice* oleh Admin Penjualan	6,498 detik	4,908 detik

Hasil pengujian *load testing* pada tabel 4 menunjukkan bahwa kebutuhan non-fungsionalitas telah terpenuhi. Berdasarkan hasil perhitungan bahwa pembuatan *Sales Invoice* menggunakan *message*

queue mempunyai waktu respon 3,467 detik lebih cepat untuk *middleware* dan waktu respon 1,59 detik lebih cepat untuk admin penjualan.

KESIMPULAN

Berdasarkan temuan dari penelitian, dapat dipahami bahwa dengan integrasi *ERPNext* dan *Shopee* menggunakan *middleware* dapat menghemat waktu dan meningkatkan efisiensi dengan meniadakan kebutuhan entri data penjualan *Shopee*, pembuatan produk, dan pembaruan stok produk secara manual, sehingga memungkinkan karyawan untuk fokus pada tugas penting lainnya. Secara keseluruhan, menerapkan sistem integrasi dapat memberikan banyak manfaat bagi PT. XYZ, termasuk peningkatan akurasi, efisiensi, dan produktivitas.

Proses perencanaan kebutuhan dilakukan dengan diskusi dengan pengguna serta analisa referensi penelitian sebelumnya. *Use case diagram*, *activity diagram*, dan *class diagram* digunakan dalam perancangan *workshop RAD* sebagai hasil dari perencanaan yang telah dilakukan. Kemudian implementasi dilakukan dengan *Frappe Framework* dan pengujian menggunakan metode *blackbox testing* dan *load testing*. Pada pengujian *blackbox testing*, hasil integrasi sudah sesuai dengan kebutuhan fungsional pengguna. Pada hasil pengujian *load testing* mempunyai kesimpulan bahwa *middleware* dengan implementasi *message queue* mempunyai waktu respon yang lebih cepat, sehingga hal ini menunjukkan bahwa kebutuhan non-fungsional sudah terpenuhi dengan baik.

Selain integrasi dengan *Shopee*, pengembangan selanjutnya dapat dilakukan dengan menambahkan integrasi terhadap *marketplace-marketplace* lainnya yang tersedia di Indonesia seperti *Zalora, Lazada, Bukalapak, Tokopedia*, dan lain-lainnya, yang akan menjadikan *middleware* ini solusi untuk melengkapi dukungan fitur pada sistem *ERPNext.*

DAFTAR PUSTAKA

[1] C. M. Annur, "Aplikasi Belanja Online Paling Banyak Digunakan, Ini Dia Juaranya," *Databoks.Katadata.Co.Id*, Jul. 18, 2022. https://databoks.katadata.co.id/datapublish/2022/07/18/aplikasi-belanja-online-paling-banyak-digunakan-ini-dia-juaranya

[2] L. . Krithika, B. Prabadevi, N. Deepa, and S. Bhavanasi, "Integration of E-Commerce System with Various ERP Tools," in *2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE)*, Feb. 2020, pp. 1–8. doi: 10.1109/ic-ETITE47903.2020.43.

[3] A. H. Al-Badi and A. Khan, "Enterprise Resource Planning Systems Development in Omani Higher Education Institutions from the Perspectives of Software Project Managers and Developers," *J. Business, Commun. Technol.*, vol. 1, no. 1, pp. 14–23, 2022, doi: 10.56632/bct.2022.1102.

[4] F. Santos and R. Martinho, "Architectural Challenges on the Integration of e-Commerce and ERP Systems: A Case Study," 2021.

[5] Y. Wang and Y. Shi, "Analysis on the integration of ERP and e-commerce," 2017, p. 020137. doi: 10.1063/1.4992954.

[6] E. Abbasi, A. W. Farooqui, M. F. Batra, M. A. Rehmani, and S. M. Anas, "Bridging the Gap between ERP Applications and eCommerce Solutions," *Int. J. e-Education, e-Business, e-Management e-Learning*, vol. 7, no. 2, pp. 111–122, 2017, doi: 10.17706/ijeeee.2017.7.2.111-122.

[7] M. Saranya and A. A. Priya, "A Study on Middleware Technologies in Cloud Computing," *Int. J. Innov. Res. Sci. Technol.*, vol. 4, pp. 31–36, 2017.

[8] M. Kubus, "Integrace e-shopového řešení a ERP systému [Integration of e-shop and ERP]," Masaryk University, Faculty of Economics and Administration, Brno, 2019.

[9] J. Östman, "Development of Magento 1.9 Extension for Briox ERP Integration," Yrkeshögskolan Novia, 2018.

[10] R. Rodriguez and W. Maritza, "Integración De La Plataforma E-Commerce Shopify Con Sap Business One Para PYMES [Shopify E-Commerce Platform Integration with Sap Business One for SMEs]," Universidad Nacional Tecnológica de Lima Sur, 2021.

[11] T.-P. Kaewprathum, "Architectural analysis of Retail Omni-channel and integration of Cash IT Point-Of-Sale software with E-commerce platform," Karlstad University, Faculty of Health, Science and Technology (starting 2013), 2018.

[12] E. A. Kavats and A. A. Kostenko, "Analysis Of Connection Methods Of Telegram Robots With Server Part," *Syst. Technol.*, vol. 3, no. 122, pp. 19–24, Oct. 2019, doi: 10.34185/1562-9945-3-122-2019-03.

[13] A. Kaya and Ö. Aydin, "E-Commerce in Turkey and SAP Integrated E-Commerce System," *ArXiv*, vol. abs/2104.0, 2019.

[14] K. C. Laudon and J. P. Laudon, *Management Information Systems: Managing the Digital Firm*, 16th ed. Pearson, 2020.

[15] K. E. Kendall and J. E. Kendall, *Systems Analysis and Design*, 8th ed. Pearson College Div, 2010.

[16] J. S. Tanuwidjaja, "Aplikasi Batch Image Editor Berbasis Web Menggunakan Arsitektur Message Queue," Institut Bisnis dan Informatika Kwik Kian Gie, Jakarta,

2020.

[17] R. S. Pressman and B. R. Maxim, *Software Engineering: A Practitioner's Approach*, 8th ed. New York, NY : McGraw-Hill Education, [2015] : McGraw-Hill Education, 2015.

[18] R. Chopra, *Software Testing: A Self-Teaching Introduction*, Illustrated. Mercury Learning and Information , 2018.

[19] D. I. Permatasari, "Pengujian Aplikasi menggunakan metode Load Testing dengan Apache JMeter pada Sistem Informasi Pertanian," *J. Sist. dan Teknol. Inf.*, vol. 8, no. 1, p. 135, 2020, doi: 10.26418/justin.v8i1.34452.