Jurnal Teknologi Sistem Informasi dan Aplikasi **ISSN: 2654-3788**
Penerbit: Program Studi Teknik Informatika Universitas Pamulang **e-ISSN: 2654-4229**
Vol. 6, No. 2, April 2023 (54-62) **DOI: 10.32493/jtsi.v6i2.29195**

Pembangunan *Stock Monitoring Application* untuk Pembuatan Laporan *Goods in Transit* Menggunakan Flask

Muhammad Al Frizal Deva Syach Putra[1], Yeremia Alfa Susetyo[2]

Teknik Informatika, Universitas Kristen Satya Wacana, Jl. Dr. O. Notohamidjojo No.1, Salatiga, Jawa Tengah, Indonesia, 50715
e-mail: [1]672019199@student.uksw.edu, [2]yeremia.alfa@uksw.edu

Submitted Date: March 13th, 2023 Reviewed Date: March 20th, 2023
Revised Date: March 24th, 2023 Accepted Date: March 30th, 2023

Abstract

PT XYZ is the largest retail company in Indonesia. PT XYZ is definitely inseparable from problems regarding stock items. The problem is generally about the difference in stock of goods that can occur due to the condition of goods in transit from the warehouse to the store or vice versa (Goods in Transit), goods lost on the way, or the store has not made a goods receipt report. One strategy to overcome this problem is to build a Stock Monitoring application for web-based Goods in Transit (GIT) report generation using the Flask Framework. The application is built using the Python programming language and the Waterfall method. The application of the MVC (Model, View, Controller) architecture of the Flask Framework and the Waterfall method can support the development of the Stock Monitoring application because the code settings are easy to manage and each process is carried out systematically so that the workflow becomes structured and easy. The result of this research is the application of the Flask Framework and the Waterfall method in the Stock Monitoring application which is used to generate GIT reports to overcome the problem of missing stock.

Keywords: Flask; Goods In Transit; Python; Website Application

Abstrak

PT. XYZ merupakan perusahaan retail terbesar di Indonesia. PT. XYZ pasti tidak terlepas dengan permasalahan mengenai stok barang. Masalah tersebut umumnya mengenai selisih stok barang yang bisa terjadi karena adanya kondisi barang dalam masa perjalanan dari gudang ke toko atau sebaliknya (Goods in Transit), barang hilang dalam perjalanan, atau toko belum membuat laporan penerimaan barang. Salah satu strategi untuk mengatasi masalah tersebut adalah membangun aplikasi Stock Monitoring untuk pembuatan laporan Goods in Transit (GIT) berbasis web menggunakan Framework Flask. Aplikasi dibangun menggunakan bahasa pemrograman Python dan metode Waterfall. Penerapan arsitektur MVC (Model, View, Controller) dari Framework Flask dan metode Waterfall dapat mendukung pengembangan aplikasi Stock Monitoring karena pengaturan kode yang mudah dikelola dan setiap proses dilakukan secara sistematis sehinga alur kerja menjadi terstruktur dan mudah. Hasil penelitian ini adalah penerapan Framework Flask dan metode Waterfall pada aplikasi Stock Monitoring yang digunakan untuk pembuatan laporan GIT untuk mengatasi masalah missing stock.

Kata Kunci: Flask; Goods In Transit; Python; Aplikasi Website

1. Pendahuluan

Pertumbuhan bisnis perusahaan yang secara cepat tidak terlepas dari perkembangan teknologi informasi yang bertumbuh semakin pesat. Perusahaan yang bergerak di bidang ritel perlu memberikan informasi yang lebih cepat dan akurat, terutama mengenai hal yang berkaitan dengan ketersediaan stok barang. (Junaidi, Sutrisno, & Janah, 2019). PT. XYZ merupakan salah satu perusahaan retail terbesar di Indonesia. Untuk

memperlancar akomodasi dan proses bisnis barang dari gudang ke toko, PT. XYZ memerlukan berbagai sistem yang saling tersinkronisasi satu dengan lainnya untuk mendukung dan memenuhi kebutuhan proses bisnis tersebut. Monitoring stok barang perlu dan penting dilakukan secara berkala karena stok barang menjadi lebih akurat dan mudah dikelola serta seluruh barang selalu berada dalam pengawasan. (Martono, 2021). Agar stok barang yang tersedia di dalam gudang stabil, maka jumlah barang masuk maupun keluar perlu diperhatikan. *Goods In Transit* (GIT) adalah kondisi barang dalam masa perjalanan. GIT dapat dikategorikan menjadi 2 jenis, yaitu GIT *Transfer* dan GIT *Return Out*. GIT *Transfer* merupakan barang yang sedang dalam perjalanan dari gudang ke toko. Sedangkan GIT Return Out merupakan barang yang sedang dalam perjalanan dari toko ke gudang.

Pada kenyataannya, masih banyak perusahaan yang belum mampu melakukan pengelolaan persediaan stok barang dengan baik. Terdapat beberapa faktor yang mempengaruhi hal tersebut, salah satunya adalah informasi yang tidak lengkap mengenai persediaan stok barang. Perusahaan menjadi tidak memiliki kejelasan mengenai ketersediaan barang sehingga akhirnya tidak dapat memenuhi kebutuhan pelanggan. Faktor lainnya adalah permintaan barang dari setiap toko ritel berbeda-beda tergantung dari waktu dan jarak antara gudang dengan toko serta keterbatasan kapasitas kendaraan dalam memenuhi permintaan. Terlebih lagi apabila jarak antara gudang dengan toko berbeda pulau sehingga pengiriman barang akan memakan banyak waktu. Selain itu, perkembangan rute jalan yang bertambah akan menjadi rumit jika tujuan dari pengiriman barang semakin banyak sedangkan jumlah kendaraan tidak mencukupi. Fakta di lapangan, jarak antara gudang dengan toko akan mempengaruhi terjadinya GIT, semakin jauh jarak toko dari gudang maka berpotensi GIT terjadi sehingga menimbulkan selisih stok barang untuk sementara waktu. Selisih stok barang disini bisa saja terjadi karena barang sedang dalam perjalanan, kehilangan barang sewaktu dalam perjalanan atau barang sudah sampai di toko namun pihak toko belum membuat laporan penerimaan barang.

Flask Framework adalah kerangka kerja web yang disediakan oleh *Python*. *Flask framework* tidak membutuhkan *library* atau *tools* sehingga dapat digolongkan sebagai *micro-framework*. Flask menggunakan dependensi *Werkzeug* dan *Jinja Template Engine*. Pada *Flask*, file gambar dan kode status seperti kode *JavaScript* dan kode CSS disimpan pada *Static File*. Sedangkan, *template Jinja* termasuk juga kode HTML disimpan pada *Template File*. Pada *Flask*, komponen umum dan sebagian besar fungsi tidak terpasang secara bawaan. Namun, pihak ketiga yang bersifat sebagai ekstensi menyediakan fungsi dan komponen-komponen tersebut. Sehingga fungsi dan komponen tersebut seakan diimplementasikan oleh *Flask Framework* sendiri. (Ngantung & Pakereng, 2021). Dengan menggunakan *Flask Framework*, pengembangan sebuah website dapat menjadi lebih terstruktur dengan lebih mudah. Forum untuk berdiskusi mengenai masalah terkait *Flask* tersedia banyak di internet dan didukung dengan dokumentasi yang baik. Berdasarkan kelebihan yang ditawarkan *Flask*, implementasi *framework* ini cocok sebagai infrastruktur website yang mengolah data dan pembuatan laporan barang dengan cepat. Aplikasi website tersebut dapat digunakan untuk melakukan laporan pengecekan *missing stock* atau item selisih. Penerapan MVC (*Model*, *View*, *Controller*) *Flask* sangat mendukung dalam pembuatan aplikasi website *Stock Monitoring* untuk pembuatan laporan GIT. Dengan penerapan arsitektur MVC, semakin kompleks aplikasi yang dibuat, pengaturan kode didalamnya juga akan mudah dikelola. Sehingga, berdasarkan kelebihan dan penerapan arsitektur dari *Flask*, pembuatan aplikasi *Stock Monitoring* untuk pembuatan laporan GIT menggunakan *framework Flask* sangatlah cocok. Tujuan dari penelitian ini adalah membangun aplikasi *Stock Monitoring* untuk pembuatan laporan *Goods In Transit* (GIT) berbasis web menggunakan *Framework Flask* di PT. XYZ.

2. Penelitian Terdahulu

Menurut Setiawan dkk (Setiawan, et al, 2021) menyatakan bahwa kesalahan dalam perhitungan *item stock* dan pembuatan laporan seringkali terjadi di PT. Intermetal Indo Mekanika karena perusahaan tersebut dalam pelaporan dan perhitungan *item stock* masih menggunakan sistem manual. Penelitian ini menggunakan metode analisis PIECES, metode perancangan dengan menggunakan diagram UML dan metode pengujian perangkat lunak menggunakan *Black Box*. Hasil penelitian ini berupa suatu aplikasi

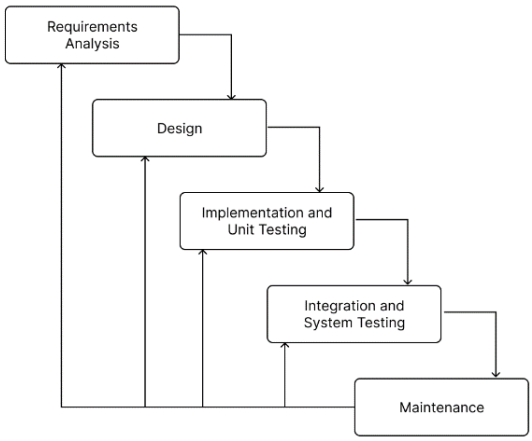
monitoring aplikasi stok barang untuk memantau ketersediaan barang menjadi akurat.

Menurut Novindri (Novindri & Saian, 2022) menyatakan bahwa diperlukan sistem minimal *order* untuk menentukan nilai *order* tiap toko agar dapat memaksimalkan keuntungan dan mengurangi kerugian dalam penjualan tiap barang yang ada di toko. Penelitian ini menggunakan metode Waterfall yang terdiri dari tahapan *System Analyst*, *Planning and Design System*, *Implementation* dan *Testing*. Penelitian ini menghasilkan sistem yang berfungsi untuk mengatur minimal order tiap item barang.

Menurut Sholeh dkk (Sholeh, et al, 2022) menyatakan bahwa beberapa layanan pemesanan makanan pada hal pengolahan data menu makanan masih bersifat manual sehingga kurang efisien dari segi waktu dan pelayanan. Penelitian ini menggunakan metode *waterfall* yang terdiri dari analisis sistem, desain sistem, desain basis data dan coding serta evaluasi. Hasil dari penelitian ini adalah aplikasi layanan menu makanan berbasis website.

3. Metodologi

Metode penelitian yang dipakai dalam membangun aplikasi monitoring stok untuk pembuatan laporan *Good In Transit* (GIT) adalah *Waterfall Model*. Model *Waterfall* merupakan salah satu pendekatan atau metode Software Development Life Cycle (SDLC) yang sering digunakan untuk pengembangan *software*. (Wahid, 2020). Metode *Waterfall* adalah salah satu jenis pengembangan aplikasi dengan penekanan pada fase yang sistematis dan berurutan. Model *Waterfall* merupakan model yang sering dipakai dalam pengembangan perangkat lunak atau *software engineering* dan termasuk ke dalam model generic pada rekayasa perangkat lunak. (Widiyanto, 2018). Model *Waterfall* memiliki beberapa kelebihan yang bisa didapatkan dalam proses implementasinya, yaitu proses pengembangan dilakukan secara satu per satu sehingga dapat meminimalisir kesalahan yang mungkin akan terjadi, setiap proses dilakukan secara sistematis dan bertahap sehinga alur kerja menjadi jelas dan terukur, lebih hemat biaya karena tidak banyak membutuhkan sumber daya, dokumentasi yang baik, dan cocok untuk pengembangan perangkat lunak berskala besar seperti pembangunan aplikasi *Stock Monitoring* untuk pembuatan laporan *Good In Transit* (GIT). Model *Waterfall* memiliki beberapa tahapan metode yang dapat dilihat pada gambar berikut:



Gambar 1. Metode *Waterfall*

Tahapan penelitian yang ditunjukan pada Gambar 1 adalah sebagai berikut:

1) Tahap *Requirements Analysis*
   Analisis kebutuhan merupakan tahapan pertama dalam model *Waterfall*. Dengan adanya permasalahan dalam pengiriman barang dan jarak antara gudang dengan toko, maka dibangunlah sebuah sistem yang dapat melakukan pengecekan *missing stock* atau item selisih. Semua informasi mengenai data gudang, data toko dan data barang kemudian dianalisa dan diolah sehingga didapatkan informasi yang lengkap mengenai spesifikasi kebutuhan pengguna dan perangkat lunak yang akan dikembangkan.
2) Tahap *Design*
   Tahapan yang kedua adalah *design.* Tahapan *design* adalah proses perancangan dan pengembangan *system software* berdasarkan informasi kebutuhan pengguna. Dalam pengembangan aplikasi *Stock Monitoring* untuk pembuatan laporan *Good In Transit* (GIT), perancangan sistem dibuat dengan menggunakan *Unified Modelling Language* (UML) yang digambarkan dengan *Use Case Diagram* dan *Activity Diagram*.
3) Tahap *Implementation and Unit Testing*
   Tahapan yang ketiga adalah *Implementation and Unit Testing*. Tahapan

ini adalah tahapan implementasi yang mengarah pada proses pemrograman atau penulisan *code program*. Dalam pengembangan aplikasi *Stock Monitoring* untuk pembuatan laporan *Good In Transit* (GIT), rancangan yang sudah dibuat pada tahap sebelumnya diimplementasikan menggunakan *Bootsrap* untuk mempercantik implementasi dari tampilan antarmuka aplikasi pada sisi *front-end*, sedangkan pada sisi *back-end* diimplementasikan ke dalam bahasa pemrograman *Python* dengan menggunakan kerangka kerja aplikasi *Flask Framework*, database menggunakan *PostgreSQL* dengan *tools DBeaver*.

4) Tahap *Integration and System Testing*
Tahap keempat adalah *Integration and System Testing* yang mengacu kepada pengintegrasian dari setiap modul. Pengujian dan pemeriksaan secara keseluruhan dilakukan untuk mengidentifikasi kemungkinan adanya kesalahan dan kegagalan sistem. Dalam pengembangan aplikasi *Stock Monitoring* untuk pembuatan laporan *Good In Transit* (GIT), pengujian dilakukan dengan menggunakan pengujian *blackbox*. Pengujian *blackbox* dilakukan dari sisi fungsionalitasnya agar dapat memenuhi fungsi sebagaimana mestinya.

5) Tahap *Maintenance*
Tahap *Maintenance* merupakan tahapan terakhir dari model *Waterfall*. Tahapan ini dilakukan agar sistem terpelihara dan memastikan sistem tetap berjalan baik sesuai dengan fungsinya. Tahapan ini meliputi beberapa proses, antara lain perbaikan implementasi unit sistem, peningkatan performa sistem sesuai dengan kebutuhan, dan perbaikan error.

4. Hasil dan Pembahasan

Penelitian ini menghasilkan modul Report *Goods In Transit* (GIT) berbasis website pada aplikasi *Stock Monitoring* yang dibangun menggunakan *framework flask*. Pengembangan modul dikerjakan dengan menggunakan database *dummy* dan server perusahaan yang dapat diakses melalui *Virtual Private Network* (VPN).

4.1. Analisis Kebutuhan Sistem

Implementasi dari *framework flask* di aplikasi *Stock Monitoring* hanya dapat diakses menggunakan jaringan server perusahaan. Dalam pembangunan aplikasi *Stock Monitoring* memerlukan beberapa kebutuhan perangkat lunak dan perangkat keras yang dipergunakan untuk mendukung pembangunan aplikasi ini, antara lain:

Kebutuhan perangkat lunak:
1. Sistem Operasi Windows
2. Visual Studio Code
3. Bahasa Pemrograman *Python* dan *Javascript*
4. *Database* PostgreSQL
5. FortiClient VPN

Kebutuhan perangkat keras:
1. Memori RAM 8GB
2. Hardisk

4.2. Arsitektur Sistem

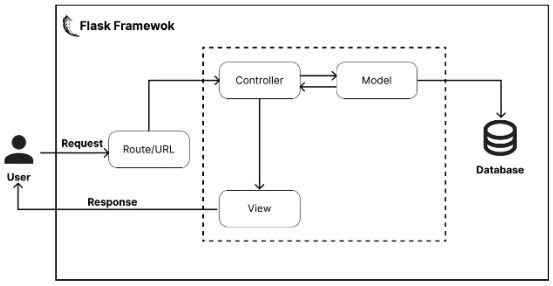
Arsitektur sistem yang dibuat pada awal perancangan sistem digunakan untuk menjelaskan alur data dari sistem. Arsitektur yang digunakan dalam membangun aplikasi *Stock Monitoring* adalah arsitektur *Model View Controller* (MVC). Dalam pembuatan aplikasi website, MVC merupakan sebuah pola desain arsitektur yang memisahkan kode menjadi tiga bagian. Ketiga bagian tersebut adalah *model*, *view*, dan *controller*.

Model merupakan bagian yang bertugas untuk mengatur dan mengelola data serta berhubungan langsung dengan database. Di dalam aplikasi *Stock Monitoring*, model diberi nama sebagai DAO. Pada bagian DAO berisikan bagian kode yang mengatur proses *database*.

View merupakan bagian bertugas untuk menyajikan desain tampilan dan informasi yang akan berinteraksi langsung dengan user dalam bentuk *Graphical User Interface* (GUI). Di dalam aplikasi *Stock Monitoring*, *view* merupakan file HTML yang berada di dalam *folder templates*. Dengan menggunakan *Template Jinja* dari *flask* sangat membantu dan mempercepat pekerjaan pada *file* HTML yang sedang dikerjakan.

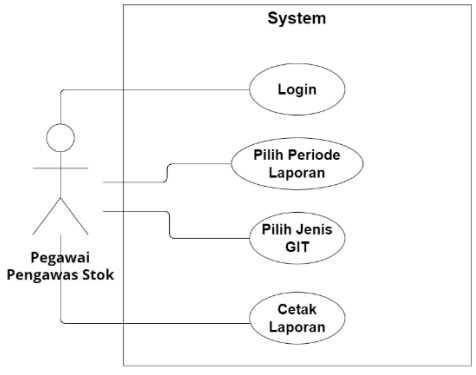
Controller merupakan bagian yang bertugas untuk mengatur dan menghubungkan antara model dengan *view* dalam setiap proses *request* dari *user* agar dapat saling terhubung. Di dalam aplikasi *Stock Monitoring*, *controller* merupakan bagian yang menampung semua *route* proses yang

menghubungkan antara *user* dengan tampilan maupun tampilan dengan *database*.



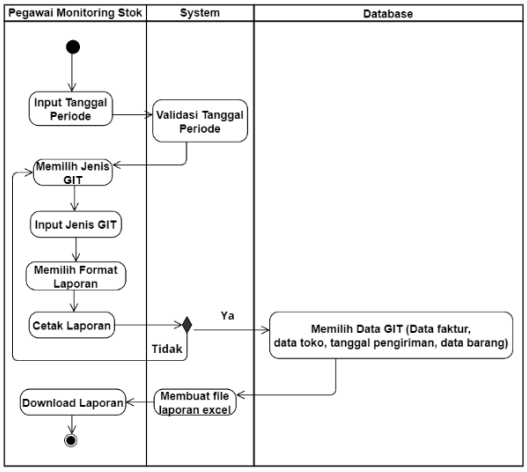
Gambar 2. Arsitektur Sistem

Berdasarkan Gambar 2, dalam proses pembuatan laporan GIT, hal pertama yang dilakukan adalah *user* melakukan *request* melalui URL. *Request* tersebut diterima oleh *controller* sehingga akan menampilkan halaman *Report GIT*. Pembuatan laporan akan dilakukan oleh *user* setelah selesai memasukkan data form pada halaman tersebut. Setelah semua data form tervalidasi, sistem akan melakukan *request* kepada *controller*. *Controller* akan menerima permintaan tersebut dan akan dikirimkan ke bagian *model* untuk diproses. Selanjutnya *model* akan mencari dan mengolah data yang diminta di dalam database. Setelah data ditemukan dan diolah, *model* akan mengirimkan data tersebut kepada *controller*. *Controller* mengambil hasil olahan tersebut dan membuatkan file laporan yang akan dikirimkan di bagian *view* untuk ditampilkan dan diunduh kepada *user*. Dari alur program yang berjalan, penggunaan *framework flask* terdapat pada pemindahan data dari *client* ke *server* menggunakan AJAX dan penentuan rute.



Gambar 3. *Use Case Diagram*

Pada Gambar 3 di atas adalah *Use Case Diagram* yang digunakan di dalam penelitian ini. Pada *Use Case* tersebut aktornya adalah *user*. *User* disini merupakan karyawan perusahaan yang menjalankan aplikasi yaitu pegawai pengawas stok. Pengguna dapat melakukan *login* ke aplikasi untuk membuat laporan GIT. Pada saat proses pembuatan laporan, pengguna dapat memilih periode, jenis GIT, dan mencetak laporan.



Gambar 4. *Activity Diagram*

Pada Gambar 4 merupakan *activity diagram* yang menggambarkan aktivitas yang dilakukan oleh *user* pada saat membuat laporan GIT. Untuk membuat laporan GIT, *user* diwajibkan mengisi semua data dengan benar. Yang pertama dilakukan adalah mengisi tanggal periode untuk laporan yang mau dicetak. Setelah kedua tanggal periode diisi, sistem akan melakukan validasi. Validasi dilakukan untuk pengecekan tanggal sudah sesuai atau belum. Apabila tanggal kedua bernilai lebih kecil daripada tanggal pertama, maka data periode tanggal tidak valid. Setelah itu, *user* memilih jenis GIT dan menginput data sesuai dengan jenis yang dipilih. Setelah semua data terisi, *user* dapat mencetak laporan GIT. Sistem akan memvalidasi data dari *user*. Apabila semua data sudah terisi dan valid maka sistem akan meneruskan proses ke database. Di dalam database, data diolah sesuai dengan permintaan *user*. Setelah data diolah dan didapatkan, sistem akan membuatkan laporan file berformat excel. Setelah laporan selesai dibuat oleh sistem aplikasi, *user* dapat mengunduh file laporan tersebut.

4.3. Implementasi

Bahasa pemrograman *Python* dan *Framework Flask* digunakan untuk

mengimplementasikan hasil dari perancangan sistem yang telah dibuat. Hasil dari penelitian yang dilakukan adalah aplikasi berbasis website yang digunakan untuk monitoring *stock flow* di gudang dan toko atas seluruh transaksi yang berlangsung didalamnya khususnya pembuatan laporan *Goods In Transit* (GIT).

Kode Program 1. Main.py

```
from aplikasi import app

if __name__ == '__main__':
        app.run(debug = True, host =
"0.0.0.0", port = 5050)
```

Dalam pembuatan modul *Report* GIT, *flask server* dari aplikasi dijalankan pada file main.py yang berisikan host dan port yang digunakan untuk mengakses aplikasi. Kode pada baris pertama bertujuan untuk mengimport app dari modul aplikasi. Pada baris kode selanjutnya aplikasi dapat dijalankan dengan metode *run()* dengan membawa argumen *debug=True* yang berfungsi untuk identifikasi atau pelacakan *error* pada aplikasi. Selain itu, terdapat konfigurasi dari aplikasi *Flask* yang dapat diakses menggunakan *localhost* dengan port 5000.

Kode Program 2. Init.py

```
from flask import Flask
app = Flask(__name__)

app.config.from_pyfile('settings.py')

with app.app_context():
    from aplikasi.controllers.report.git
    import (rep_git_con)
```

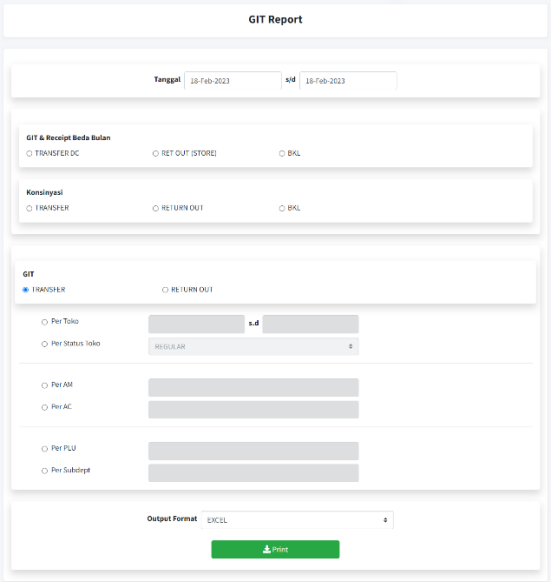
Implementasi *Framework Flask* di dalam aplikasi dapat dilakukan dengan cara mengimport *flask* dan membuat objek dari kelas *Flask*. Pada contoh Kode Program 2, kode *from flask import Flask* digunakan untuk mengimport kelas *Flask* yang terdapat dalam modul *flask* yang sudah diinstal terlebih dahulu dengan menggunakan perintah *pip instal Flask*. Kode *app = Flask(__name__)* merupakan objek dari kelas *flask*. Pada baris kode empat, merupakan konfigurasi yang diambil dari file *settings.py*. Kode program pada baris terakhir merupakan proses yang digunakan untuk mendaftarkan *file controller* agar dapat diakses oleh *Flask*.

Kode Program 3. *Controller*

```
from flask import render_template
      from aplikasi import app

@app.route('/report/git/gitreport',
methods=['GET', 'POST'])
      @login_required
      def repGitReport():
      return
render_template('/report/git/rep_git.html',
title='GIT Report')
```

Baris kode program empat, merupakan tahapan deklarasi fungsi *route* dan *methods* yang akan digunakan. Fungsi *route* dan *@login_required* merupakan *decorator* yang memiliki tujuan berbeda. Fungsi *route* digunakan untuk memanggil URL oleh sistem sedangkan *@login_required* digunakan untuk mencegah *user* yang tidak *login* untuk masuk ke dalam aplikasi. Fungsi *render_template()* adalah menjalankan sebuah halaman HTML yang dituju dengan memanggil *template engine* yaitu Jinja 2, sehingga ketika URL pada aplikasi diakses akan menampilkan sebuah halaman *report* GIT.



Gambar 5. Halaman *Report* GIT

Pada Gambar 5 merupakan tampilan dari halaman *report* GIT dalam aplikasi monitoring stok. *User* dapat memilih tanggal periode untuk laporan yang akan dicetak. Pembuatan laporan akan disesuaikan dengan jenis laporan yang akan dipilih oleh *user*. Pada *text box* yang berwarna kuning merupakan *List of Value* (LOV) yang

dimana ketika *user* mengklik area tersebut maka akan muncul modal yang berisikan list data yang dapat dipilih oleh *user*. Format cetakan dapat dipilih melalui *select box*, untuk saat ini format yang tersedia hanyalah *excel* saja. Setelah semua data terisi dengan benar, user dapat mencetak laporan GIT dengan cara menekan tombol *print*.

Kode Program 4. Penggunaan AJAX

```
$.ajax({
        type: "POST",
        url: "/report/git/rep_git_export",
              data: {tgl1: tgl1, tgl2:
        tgl2,
        radioGit:radioGit,radioIsiGit:radio
        IsiGit, statusToko:getStatusToko(),
        toko1:toko1, toko2:toko2,
        kode_am:kode_am,kode_ac:kode_ac,
        :plu,subdept:subdept},
              xhrFields: {
                responseType: 'blob'
              },
        success: function (res) {
                let a =
        document.createElement('a');
                let url =
        window.URL.createObjectURL(res);
                a.href = url;
                a.download =
        setNameFile()+tanggal1+'_sd_'+tangg
        al2+'.xlsx';
                        document.body.append(a
);
                a.click();
                      a.remove();
                      window.URL.revokeObjec
                tURL(url);
                      hideLoading();
        },
        error: function() {
                toastr.error('Cetak Laporan
Gagal!');
               hideLoading();
        }
      });
```

Pembuatan laporan GIT menggunakan AJAX di dalam Jquery yang berguna untuk mengirimkan dan menerima data antara klien dan server. Kode Program 4 merupakan fungsi untuk pembuatan laporan GIT yang akan berjalan ketika tombol *print* ditekan. Pada baris kode program kedua, *Type* berfungsi untuk mengatur *method* yang akan digunakan di dalam proses. *Method POST* digunakan untuk mengirimkan data secara langsung ke URL yang dituju. Pengiriman data dapat dilakukan dengan mengirimkan data berupa JSON untuk data yang banyak dan dapat juga dilakukan dalam bentuk *string* untuk data tunggal. Fungsi *succes* akan berjalan ketika proses yang dilakukan berhasil. Sedangkan fungsi *error* akan menangani kesalahan ketika proses yang dilakukan terdapat kegagalan.

Kode Program 5. Kode Program untuk Membuat Excel

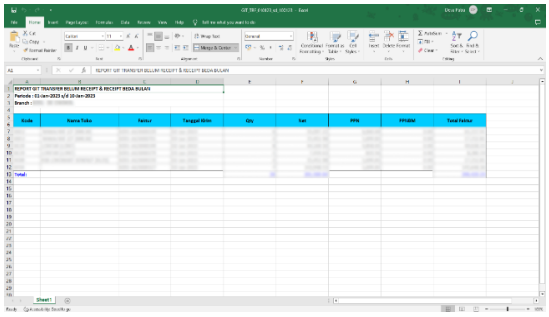
```
import io
import pandas as pd

out = io.BytesIO()
writer = pd.ExcelWriter(out)

dataframe = pd.DataFrame()
dataframe.to_excel(writer,
sheet_name='Sheet1', startrow=0,
header=False, index=False)

workbook = writer.book
worksheet = writer.sheets['Sheet1']
```

Pembuatan laporan GIT berformat excel dapat dilakukan dengan menggunakan bantuan dari *library* yang telah disediakan oleh *Python*. Pada baris kode program ketiga, *IO Python module* mengizinkan *developer* untuk mengelola file-file yang berkaitan operasi *input* dan *output*. Dengan menggunakan *BytesIO()* data dapat disimpan ke dalam bentuk byte di dalam memori. Dengan memanfaatkan *ExcelWriter* dari *Pandas*, kelas *ExcelWriter()* digunakan untuk menyimpan *DataFrame* ke *Excel sheet*. Dengan begitu, laporan dalam format bentuk excel dapat dibuat dengan cara mengatur panjang dari kolom-kolom excel, mengatur format penulisan, mengambil data dari *model* (DAO), menulis data, menyimpan dan menutup *writer* serta mengirimkan respon file excel kembali ke *view* untuk diunduh oleh *user*.



Gambar 6. Contoh Laporan GIT

Pada Gambar 6 menampilkan hasil laporan GIT yang telah dibuat oleh *user*. Data laporan tersebut digunakan oleh karyawan perusahaan untuk melakukan pengecekan *missing stock* atau item selisih.

4.4. Pengujian

Pengujian atau testing pada sistem aplikasi dilakukan dengan menggunakan metode *Black Box Testing*. Pengujian dilakukan untuk mengecek semua kemungkinan *bug* dan *error* pada aplikasi. Berikut merupakan hasil dari pengujian tersebut:

Tabel 1. Hasil Pengujian *Black Box*

No	Uraian	Hasil yang Diinginkan	Status
1	Periode tanggal 1 harus lebih kecil dari tanggal 2, jika tanggal 1 lebih besar dari tanggal 2 maka terdapat alert.	Sudah sesuai	OK
2	Jika kode toko kosong pada pilihan report GIT maka terdapat alert saat mencetak laporan.	Sudah sesuai	OK
3	Jika kode AM atau kode AC kosong pada pilihan report GIT maka terdapat alert saat mencetak laporan.	Sudah sesuai	OK
4	Jika kode PLU atau kode Subdept kosong pada pilihan report GIT maka terdapat alert saat mencetak laporan.	Sudah sesuai	OK
5	Menampilkan LOV sesuai dengan pilihan kategori pada report GIT.	Sudah sesuai	OK
6	Menampilkan laporan dengan format yang telah disesuaikan seperti format rupiah pada kolom net, PPN, PPN BM dan total faktur.	Sudah sesuai	OK

5. Kesimpulan

Berdasarkan hasil penelitian yang telah dilakukan dapat ditarik sebuah kesimpulan bahwa aplikasi *Stock Monitoring* untuk pembuatan laporan *Goods In Transit* (GIT) dapat digunakan dan mempermudahkan *user* untuk melakukan pengecekan *missing stock* atau item selisih.

Penggunaan *framework flask* dapat mempermudah pengembangan aplikasi dengan memanfaatkan *library* yang telah disediakan sehingga pengembangan bisa lebih cepat. Pada tahap pengujian sistem menunjukan bahwa hasil pembangunan aplikasi *Stock Monitoring* untuk pembuatan laporan *Goods In Transit* (GIT) telah sesuai dengan kebutuhan perusahaan sehingga aplikasi ini sudah dapat digunakan oleh *user*.

6. Saran

Berdasarkan penelitian yang telah dilakukan oleh peneliti, adapun saran yang dapat diberikan, yaitu dalam pengembangan selanjutnya diharapkan dapat memperhatikan pada bagian responsivitas tampilan aplikasi agar lebih baik lagi kedepannya.

Referensi

Junaidi, Sutrisno, & Janah, K. (2019). Model Aplikasi Purchasing System untuk Monitoring Stok dalam Mengurangi Tingkat Kerugian. *Journal Strategic of Education in Information System*, 86-98.

Kristianto, E. H., & Setiyawati, N. (2021). Pembangunan Aplikasi Virtual Inventory System (VIS) Berbasis Web Menggunakan Flask Framework . *MNEMONIC*, 128-135.

Martono. (2021). Perancangan Prototype Sistem Informasi Monitoring Stok Barang pada PT PWS. *Processor: Jurnal Ilmiah Sistem Informasi, Teknologi Informasi dan Sistem Komputer*, 96-107.

Martono, S., & Warnars, H. L. (2020). Penentuan Rute Pengiriman Barang dengan Metode Nearest Neighbor. *Jurnal Pengkajian dan Penerapan Teknik Informatika*, 44-57.

Nasution, S. W., Manurung, N., & Rahayu, E. (2022). Penerapan Supply Chain Management (SCM) Dalam Pemantauan Stok Barang Berbasis Web. *Building of Informatics, Technology and Science*, 361-368.

Ngantung, R. K., & Pakereng, M. A. (2021). Model Pengembangan Sistem Informasi Akademik Berbasis User Centered Design Menerapkan Framework Flask Python. *Jurnal Media Informatika Budidarma*, 1051-1062.

Ningtyas, D. F., & Setiyawati, N. (2021). Implementasi Flask Framework pada Pembangunan Aplikasi Purchasing Approval Request. *Jurnal Janitra Informatika dan Sistem Informasi*, 19-34.

Novindri, G. F., & Saian, P. O. (2022). Implementasi Flask Pada Sistem Penentuan Minimal Order untuk Tiap Item Barang di Distribution Center pada PT XYZ Berbasis Website. *MNEMONIC*, 80-85.

Pritalia, G. L. (2018). Penerapan Algoritma C4.5 untuk Penentuan Ketersediaan Barang E-commerce. *Indonesian Journal of Information Systems*, 47-56.

Putri, A. A., & Susetyo, Y. A. (2022). Implementasi Flask Untuk Pengecekan Stok Distribution Center & Toko Pada Aplikasi Monitoring Stock di PT. XYZ. *Jurnal Teknik Informatika*, 1265-1274.

Saputra, J. A., & Susetyo, Y. A. (2022). Pembangunan Modul Distribusi Harga Pada Aplikasi Merchandise Menggunakan Framework Flask di PT XYZ. *Jurnal Teknik Informatika*, 1293-1300.

Setiawan, A. B., Rachmawati, W., Arrahman, A. T., Natasyah, N., & Syeha, F. N. (2021). Aplikasi Monitoring Stok Barang Berbasis Web Pada PT. Intermetal Indo Mekanika. *Adi Bisnis Digital Interdisiplin Jurnal*, 1-6.

Sholeh, M., Aji, W. L., Riady, Y., & Qathasari, B. L. (2022). Pengelolaan Pemesanan Menu Makanan Menggunakan Framework Flask Python. *Jurnal Teknik Informatika dan Sistem Informasi*, 916-929.

Wahid, A. A. (2020). Analisis Metode Waterfall Untuk Pengembangan Sistem Informasi. *Jurnal Ilmu-ilmu Informatika dan Manajemen STMIK*, 1-5.

Widiyanto, W. W. (2018). Analisa Metodologi Pengembangan Sistem Dengan Perbandingan Model Perangkat Lunak Sistem Informasi Kepegawaian Menggunakan Waterfall Development Model, Model Prototype, Dan Model Rapidapplication Development (RAD). *Jurnal INFORMA Politeknik Indonusa Surakarta*, 34-40.