Jurnal Teknik Informatika (JUTIF)
Vol. 3, No. 5, Oktober 2022, hlm. 1265-1274

DOI: https://doi.org/10.20884/1.jutif.2022.3.5.334
p-ISSN: 2723-3863
e-ISSN: 2723-3871

IMPLEMENTATION OF FLASK FOR STOCK CHECKING IN DISTRIBUTION CENTER & STORE ON MONITORING STOCK APPLICATION IN PT. XYZ

Alpha Adarrani Wibowo Putri*[1], Yeremia Alfa Susetyo[2]

[1,2]Teknik Informatika, Fakultas Teknologi Informasi, Universitas Kristen Satya Wacana, Indonesia
Email: [1]672018091@student.uksw.edu, [2]yeremia.alfa@uksw.edu

(Naskah masuk: 23 Mei 2022, Revisi: 8 Juni 2022, diterbitkan: 24 Oktober 2022)

Abstract

In a retail company stock control is a very important feature. Stock control that is done manually has many weaknesses especially for large companies that has many transactions in a day. This research aims to implement flask framework for control stock on the Monitoring Stock application at PT. XYZ. Flask framework was chosen because of its easy use and has a lot of libraries. The implementation of this flask framework uses the MVC architecture (Model, View, Controller). Then, it will be tested using box testing to ensure all features can run properly. Through the implementation of flask framework on the Monitoring Stock application, stock control at PT. XYZ can be more controlled so it can avoid losses due to stock miscalculations.

Keywords*: Flask, Monitoring Stock, Python, Website*

IMPLEMENTASI FLASK UNTUK PENGECEKAN STOK DISTRIBUTION CENTER & TOKO PADA APLIKASI MONITORING STOCK DI PT. XYZ

Abstrak

Pada sebuah perusahaan ritel kontrol stok merupakan fitur yang sangat penting. Kontrol stok yang dilakukan secara manual memiliki banyak kelemahan terutama untuk perusahaan yang besar dan memiliki banyak transaksi dalam seharinya. Penelitian ini bertujuan menerapkan *framework flask* untuk pengecekan stok pada aplikasi berbasis *website* bernama Monitoring Stock di PT. XYZ. Aplikasi ini dibangun dengan menggunakan bahasa pemrograman *Python*. *Framework flask* dipilih karena penggunaannya yang mudah dan banyaknya *library* yang ada. Penerapan framework flask ini menggunakan arsitektur MVC (*Model, View, Controller*). Selanjutnya diuji dengan menggunakan metode *black box* untuk memastikan semua fitur dapat berjalan dengan baik. Dengan penerapan *framework flask* pada aplikasi Monitoring Stock pengecekan stok barang pada PT. XYZ dapat lebih dikontrol sehingga bisa terhindar dari kerugian karena salah perhitungan stok.

Kata Kunci*: Flask, Kontrol Stok, Python, Website*

1. PENDAHULUAN

Seiring dengan era yang terus berkembang, menyebabkan terjadinya proses modernisasi di lingkungan masyarakat. Salah satu dari dampak modernisasi yang terjadi adalah perubahan kebiasaan masyarakat dalam berbelanja. Masyarakat cenderung berbelanja di pasar modern ketimbang di pasar tradisional terutama di area kota besar. Hal ini ditandai dengan rasio keinginan masyarakat untuk berbelanja di pasar modern setiap tahunnya meningkat setidaknya 2% [1]. Pasar modern yang diminati masyarakat adalah minimarket, supermarket dan hypermarket [2].

Pola belanja masyarakat yang baru ini dimanfaatkan industri ritel di Indonesia untuk mengembangkan perusahaan mereka. Salah satu perusahaan ritel besar yang ada di Indonesia adalah PT. XYZ yang sudah berdiri kurang lebih 20 tahun dan memiliki lebih dari 14.300 toko yang aktif beroperasi dengan jumlah transaksi sekitar 5 juta transaksi setiap harinya. Selain di Indonesia, PT. XYZ juga memiliki 750 toko yang beroperasi di Filipina [3].

PT. XYZ menyediakan beragam jenis barang baik untuk konsumsi maupun bukan konsumsi. Barang-barang tersebut didapatkan dari *distribution center*. *Distribution center* sendiri merupakan tempat penyimpanan barang dari *supplier* sebelum dikirimkan ke masing-masing toko. Satu *distribution center* memiliki beberapa toko yang berada dibawahnya. Dengan begitu, ada banyak sekali proses bisnis yang terjadi di *distribution*

center. Setiap harinya ada banyak barang yang keluar masuk. Salah satu yang sangat perlu diperhatikan adalah pengecekan stok barang di *distribution center*.

Pengecekan stok ini merupakan fitur yang kritis untuk sebuah sistem informasi dagang dengan tujuan untuk mengetahui barang apa saja yang masih ada dan sudah habis serta untuk melakukan kontrol terhadap proses keluar masuknya barangnya [4]. Pengecekan ini akan susah terkontrol jika dilakukan secara manual dan menangani jumlah barang yang banyak dengan banyak varian. Jika sampai terjadi kesalahan dalam pengecekan stok dan barang mengalami kekosongan, akan menyebabkan kerugian bagi perusahaan. Oleh karena itu, untuk melakukan pengecekan barang pada PT. XYZ memerlukan aplikasi yang bernama *Monitoring Stock.*

Aplikasi *Monitoring Stock* ini berbasis web dengan menggunakan bahasa pemrograman *python* dan *framework flask. Python* sendiri merupakan bahasa pemrograman yang dianggap mudah untuk dipelajari bahkan oleh pemula dan merupakan bahasa pemrograman yang diklaim mudah untuk digunakan karena memiliki filosofi perancangan yang berfokus pada tingkat keterbacaan kode dilengkapi dengan *library* yang besar serta komprehensif[5]. Oleh karena itu, *python* cocok digunakan untuk membangun aplikasi yang cukup besar seperti aplikasi *Monitoring Stock* untuk perusahaan ritel. Penggunaan bahasa *python* akan memudahkan pengembang karena sudah tersedia banyak *library* yang bisa dimanfaatkan serta karena kode yang sederhana akan mempersingkat waktu dalam pengerjaan. Dengan sifat *python* yang multi-paradigma juga semakin membebaskan pengembang untuk memilih aplikasi tersebut akan berorientasi prosedural, objek atau bahkan keduanya sekaligus. *Python* sendiri juga memiliki pengelolaan memori secara otomatis[6] sehingga meskipun digunakan untuk membangun aplikasi yang besar memori akan dialokasikan secara otomatis dan efisien untuk menghemat penggunaan memori yang berlebihan.

Framework flask merupakan *web framework* yang dibangun dengan bahasa *python. Flask* sendiri tergolong ke dalam jenis *microframework*[7]. *Flask* menyediakan *library* yang bisa digunakan untuk membangun sebuah *website*, sehingga pengembang tidak perlu membangun semua dari awal. *Flask* memungkinkan pengembang untuk membangun aplikasi sesuai kebutuhan dan sekreatif mungkin dengan memanfaatkan *library* yang sudah disediakan[8]. Pengembang cukup menggunakan yang sudah disediakan dari *framework flask* ini sendiri dan membangun bagian yang tidak disediakan *flask*[8]. *Flask* juga mendukung ekstensi tambahan yang digunakan pada aplikasi yang sedang dikembangkan seperti ekstensi untuk *database*, validasi pada formulir dan *upload handling*[9]. Hal ini dapat mempersingkat waktu pembuatan aplikasi terutama untuk yang skala besar seperti *Monitoring Stock* pada perusahaan ritel.

Kelebihan *flask* yang lain adalah pengembang dapat mengembangkan dan membangun aplikasi dengan sekreatif mungkin sesuai dengan kebutuhan. Dengan begitu aplikasi dapat dibangun sesuai dengan kebutuhan dan permintaan *user*. Hal ini juga didukung dengan sifat *flask* yang memiliki fleksibilitas yang cukup tinggi dibanding dengan *framework* lainnya.

Pembangunan aplikasi *Monitoring Stock* ini menggunakan PostgreSQL untuk *database* yang nantinya digunakan. PostgreSQL merupakan *Database Object-Relational* (ORDBMD) yang mendukung banyak platform serta bebas dari lisensi. PostgreSQL bersifat *open source* sehingga pengembang dapat menggunakannya sesuai kebutuhan baik secara pribadi maupun berkelompok. PostgreSQL memiliki kemampuan untuk mendukung perintah dari SQL standar[10]. Saat mengeksekusi PostgreSQL pengembang dapat menggunakan beberapa tool contohnya adalah *pgAdmin* dan *Dbeaver*[11].

Berdasarkan permasalahan tersebut, penelitian ini bertujuan untuk mengimplementasikan *flask* untuk pembuatan menu pengecekan stok barang di *distribution center* dan juga toko pada aplikasi *Monitoring Stock* di PT. XYZ dengan platform web.

2. PENELITIAN TERDAHULU

Penelitian dengan inti masalah pengecekan stok barang pernah dilakukan oleh Maulana Hasanudin didalam jurnal yang berjudul **"*Rancang dan Bangun Sistem Informasi Inventori Barang Berbasis Web (Studi Kasus PT. Nusantara Sejahtera Raya)*"** yang dipublikasikan pada tahun 2018. Penelitian ini berfokus pada pembangunan sistem informasi yang bisa di gunakan untuk pengecekan stok barang di bioskop secara *realtime*. *Output* dari penelitian tersebut adalah sistem informasi untuk inventori pada bioskop. Pada penelitian ini dijelaskan seringnya terjadi kesalahan pencatatan saat pengecekan stok dilakukan secara manual. Kesalahan yang sering terjadi adalah adanya selisih barang yang tercatat dengan barang fisik yang ada. Maka dibangunlah sistem informasi inventori untuk melakukan pengecekan dan pencatatan stok secara otomatis. Dari penelitian tersebut setelah menggunakan sistem informasi ini pendataan barang masuk dan keluar serta pencarian barang dapat dilakukan dengan jadi lebih mudah dan tepat mudah[12].

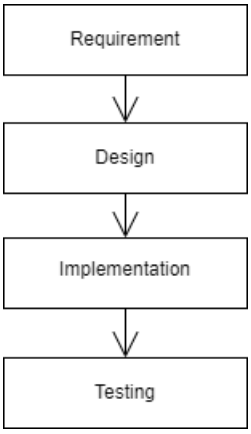
Implementasi *framework flask* juga pernah dilakukan oleh Dinda Fitri Ningtyas pada tahun 2021 pada jurnal yang berjudul ***"Implementasi Flask Framework pada Pembangunan Aplikasi Purchasing Approval Request".*** Yang membahas

tentang sistem manajemen *workflow* untuk meningkatkan efektivitas waktu dan mempermudah karyawan melakukan *purchasing request* dan *approval*. Pada penelitian ini menerapkan *framework flask* untuk pembangunan aplikasi karena memberikan kemudahan dan keleluasaan bagi pengembang dalam membangun aplikasi. Pengembang dapat dengan leluasa memanfaatkan *decorator* dan *library* yang sudah disediakan dari *flask* untuk membangun aplikasi sesuai dengan kebutuhannya. Seperti contohnya *jinja template* yang memudahkan transfer data dari *back-end* ke *front-end* aplikasi. Pemanfaatan tersebut memudahkan pengembang saat membangun sebuah aplikasi yang besar dan kompleks, selain itu juga dapat mengefisienkan waktu pembuatan. Penelitian ini menghasilkan aplikasi *Purchasing Approval Request* (PAR) dengan menerapkan *framework flask* yang memiliki keunggulan pengguna dapat dengan mudah memantau posisi atau status *purchasing request* sehingga dapat mengefisienkan waktu[13].

Framework flask juga pernah diterapkan di bidang *machine learning.* Contoh penerapannya adalah penelitian pada jurnal ***"Machine Learning Berbasis Desktop dan Web dengan Metode Jaringan Syaraf Tiruan Untuk Sistem Pendukung Keputusan"*** yang disusun oleh Rahmadya Trias dan Herlawati pada tahun 2020. Penelitian ini menjelaskan bahwa pada era industri 4.0 khususnya untuk sistem pendukung keputusan dituntut dapat diakses di media internet dan berjalan lewat web. Penulis membahas tentang tiga alat dalam pembuatan *machine learning* yakni *Google Collab, Flask*, dan *Matlab* serta perbandingan ketiganya. Perbandingan tersebut menghasilkan kesimpulan bahwa *flask* dapat digunakan untuk pembangunan antarmuka web yang interaktif. Dalam pembangunan antarmuka pengembang juga membutuhkan pengetahuan tentang HTML, CSS dan sejenisnya sebagai pendukung. Dengan dukungan *framework flask, python* memiliki keunggulan lebih pada era industri 4.0 baik untuk *front-end* maupun *back-end.* Hal ini karena aplikasi yang dihasilkan dapat diakses di multi perangkat yang dilengkapi dengan *library* bawaan yang cukup lengkap[14].

3. METODE PENELITIAN

Aplikasi *monitoring stock* merupakan aplikasi yang sangat penting untuk perusahaan, dengan begitu metode penelitian implementasi *flask* pada pembuatan menu pengecekan stok *distribution center* dan toko pada aplikasi *monitoring stock* yang paling cocok adalah dengan menyesuaikan kebutuhan perusahaan. Metode penelitian tersebut dapat dilihat pada Gambar 1.



Gambar 1 Metode Penelitian

1. *Requirement*

Tahapan pertama ini adalah pengumpulan data baik melalui wawancara, diskusi maupun *survey* langsung. Data yang didapatkan nanti akan digunakan dalam pembangunan aplikasi sehingga aplikasi yang dihasilkan sesuai dengan kebutuhan[6].

Pada penelitian kali ini data yang diperlukan untuk pembangunan aplikasi adalah:

- Data *distribution center*: yaitu data lengkap tentang *distribution center* yang ada di PT. XYZ. Seperti nama, alamat, jumlah barang yang ada, toko yang ada dibawah *distribution center*, dll.
- Data toko: yaitu data lengkap dari toko-toko yang ada. Data yang dimaksud adalah nama toko, kode toko, jumlah barang yang ada di toko, jumlah barang yang keluar, dll.
- Data karyawan: data ini digunakan untuk mengetahui siapa saja yang dapat mengakses aplikasi ini. Data yang diperlukan seperti nama, NIK, dll.
- Data barang: yaitu data tentang barang yang dikelola oleh PT.XYZ. Data yang diperlukan adalah seperti nama barang, kode barang, harga barang, jenis barang, dll.

2. *Design*

Tahapan kedua ini adalah membuat desain dari aplikasi yang akan dibangun. Desain aplikasi ini bertujuan untuk mendefinisikan arsitektur sistem secara keseluruhan dan menentukan perangkat keras yang digunakan[6].

Pada penelitian ini aplikasi akan dibangun denggan menggunakan bahasa *python* dan memanfaatkan *framework flask* yang akan di dukung dengan penggunaan HTML, CSS, *Bootstrap*, *Javascript*. Untuk *database* akan menggunakan PostgreSQL dan *tool* yang digunakan adalah pgAdmin 4.

3. *Implementation*

Selanjutnya adalah pengimplementasian kode program dengan menggunakan *tool* yang sesuai untuk membangun aplikasi baik untuk *front-end* maupun *back-end*[7].

Pada penelitian ini pembangunan aplikasi akan menggunakan IDE PyCharm yang mendukung pembuatan aplikasi dengan bahasa pemrograman *python.*

4. *Testing*

Setelah aplikasi sudah selesai dibangun, akan dilakukan pengecekan oleh pihak perusahaan atau yang disrbut dengan *testing.* Hal ini bertujuan apakah aplikasi yang dibangun sudah sesuai dengan tujuan yang diinginkan dan apakah modul pada aplikasi dapat berjalan dengan baik semua.

4. HASIL DAN PEMBAHASAN

4.1. Kebutuhan Sistem

Aplikasi pengecekan stok ini hanya dapat diakses dan dibangun dengan menggunakan jaringan perusahaan saja sehingga saat pengimplementasian *framework flask* pada aplikasi pengecekan stok memerlukan beberapa kebutuhan *software* dan *hardware.* Kebutuhan-kebutuhan tersebut adalah sebagai berikut:

Kebutuhan *hardware:*

1. RAM 8GB
2. Prosesor Intel Core i7-8750H
3. *Hard disk* 1T

Sedangkan untuk kebutuhan *software* adalah sebagai berikut:

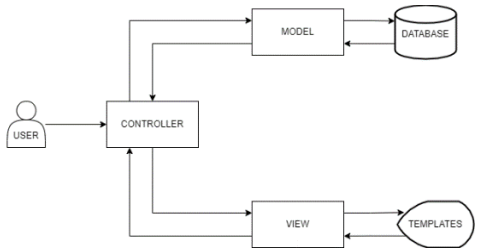
1. *FortiClient VPN* (untuk mengakses jaringan internal kantor ketika berada diluar kantor)
2. IDE *PyCharm*
3. *Flask*
4. PgAdmin 4
5. JavaScript
6. *Browser*

4.2. Arsitektur Sistem

Aplikasi *Monitoring Stock* ini memiliki peran penting dalam pengecekan stok yang ada pada perusahaan. Dalam proses pengecekan, aplikasi ini memiliki fungsi seperti *Create, Read, Update,* dan *Delete.* Fungsi *create* dapat ditemukan saat *user* menambahkan periode baru pengecekan stok kemudian memanggil prosedur untuk mengolah data stok barang sesuai dengan periode baru. Fungsi *read* dapat ditemukan saat sistem menampilkan data stok sesuai dengan periode yang dipilih. Fungsi *update* terdapat saat *user* melakukan pemrosesan ulang yang tujuannya untuk memperbarui data yang ditampilkan. Pemrosesan ulang ini dilakukan dengan cara memanggil ulang prosedur yang sama saat pemrosesan. Sedangkan fungsi *delete* digunakan saat *user* menutup modal detail data sistem akan otomatis menghapus detail data tersebut dari tabel *database*. Hal ini bertujuan agar data yang tersimpan di *database* tidak terlalu banyak.

Arsitektur yang digunakan untuk pembangunan aplikasi ini adalah arsitektur MVC atau *model, view, controller.*

Model-View-Controller (MVC) adalah sebuah konsep yang diperkenalkan oleh penemu Smalltalk (Trygve Reenskaug) untuk meng-enkapsulasi data bersama dengan pemrosesan (model), mengisolasi dari proses manipulasi (controller) dan tampilan (view) untuk direpresentasikan pada sebuah user interface[17]. *Model* sendiri merupakan bagian yang mengelola database atau yang pada aplikasi diberi nama DAO. *View* sendiri merupakan bagian *interface* yang akan berinteraksi langsung dengan *user*. Pada aplikasi bagian ini merupakan *file* HTML yang berada di folder *templates.* Sedangkan controller merupakan bagian yang menampung semua rute proses pada aplikasi *Monitoring Stock.* Rute ini yang menghubungkan antara *user* dengan tampilan *interface* aplikasi maupun antara *interface* dengan *database*. Berikut adalah skema arsitektur sistem MVC.



Gambar 2. Arsitektur MVC

Penggunaan *framework flask* sendiri ada pada bagian pemindahan data dari *client* menuju *server.* Pada bagian *front-end* data akan dikirim menggunakan *ajax* dan pada bagian *controller* data akan diterima dengan menggunakan *request* dari *flask.* Dan pengiriman data untuk sebaliknya menggunakan *return jsonify.*

Pada *controller flask* digunakan untuk menentukan rute. Penentuan rute ini menggunakan *@app.route* (rute yang ditentukan). Penggunaan rute ini bertujuan untuk menentukan bagian mana yang akan dijalankan selanjutnya. Selain untuk menentukan rute, pada *controller* juga menggunakan *render_template* untuk memanggil file HTML yang akan ditampilkan atau juga bisa menggunakan *return redirect url* untuk memanggil nama fungsi yang akan dieksekusi.

Contoh lain dari *library flask* yang digunakan adalah *flash. Flash* berfungsi untuk menampilkan notifikasi sebagai wujud respon dari kode program yang telah dijalankan.

4.3. Implementasi

Implementasi *framework flask* dapat ditemukan pada file main.py yang berisikan kode untuk menjalankan *flask server* pada aplikasi tersebut. Selain itu juga berisikan *host* dan *port* untuk mengakses aplikasi tersebut.

```
from aplikasi import app

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

Kode Program 1 Kode Program main.py

Pada baris pertama kode program, bertujuan untuk mengambil app yang berisi ini dari *flask service* difolder aplikasi. Kemudian dengan kode program selanjutnya aplikasi dapat dijalankan dengan mengakses *http://localhost:5050/*

```
NAMA_PROJECT = "MS"
NAMA_APLIKASI = " Monitoring System"
VERSION_APLIKASI = "1.0.0"

SECRET_KEY = "imsecret"
FLASK_APP = environ.get('FLASK_APP')
FLASK_ENV = environ.get('FLASK_ENV')
SQLALCHEMY_TRACK_MODIFICATIONS = False
SQLALCHEMY_DATABASE_URI = 'postgresql+psycopg2://postgres:123@localhost5432'
USER = os.getenv('db_im_uservpn')
PASS = os.getenv('db_im_passvpn')
DATA = os.getenv('db_im_datavpn')
HOST = os.getenv('db_im_hostvpn')
PORT = os.getenv('db_im_portvpn')
```

Kode Program 1 Kode program pada settings.py

Kode program 2 merupakan konfigurasi aplikasi yang ada didalam file settings.py. Kode ini berfungsi untuk mendiskripsikan aplikasi yang dibangun mulai dari nama project, nama aplikasi dan juga versi aplikasi tersebut. Selain itu digunakan untuk konfigurasi koneksi dengan *database SQLAlchemy.*

```
import psycopg2
from flask import Flask
from flask_login import LoginManager
from flask_sqlalchemy import SQLAlchemy

app = Flask(__name__)
app.config.from_pyfile('settings.py')
db = SQLAlchemy(app)

from aplikasi.controllers import (login, routes)
from aplikasi.controllers.monitoring import (mutasicon, registercon, historycon)
```

Kode Program 2 Kode Program __init__.py

Kode program 3 merupakan inti dari pengimplementasian flask pada aplikasi *Monitoring Stock.* Pada bagian pertama kode program digunakan untuk *import* fungsi dari *framework flask* yang sudah diinstal terlebih dahulu dengan menjalankan kode program *pip install Flask.* Lalu pada bagian kedua kode program merupakan deklarasi *flask service* yang diambil dari file setting.py. Sedangkan pada bagian ketiga merupakan kode program yang digunakan untuk mendaftarkan *file controller* yang terdapat pada aplikasi *Monitoring Stock. File controller* ini dibuat untuk masing masing menu yang sedang dikerjakan. Hal ini untuk menghindari kesalahan pemanggilan *controller.*

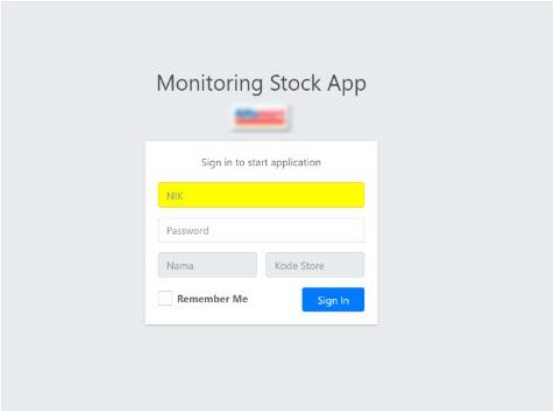
```
from flask import render_template, url_for, flash

@app.route('/monitoring/hisqtallstock', methods=['GET', 'POST'])
@login_required
def hisqtallstockcon():
    akses = cekAkses('H')
    if request.method == 'POST':
        cek = request.form.get("cek")
        if cek in ['1'] and akses[1] == 'F':
            flash('Anda tidak memiliki Akses!', 'F')
            return redirect(url_for('hisqtallstockcon'))
    return render_template('/monitoring/history/hisqtyallstock.html', title='History Quantity All Stock')
```

Kode Program 3 Controller

Pada kode program 4 merupakan contoh penggunaan *flask* pada *controller* yang dijalankan pertama kali untuk pengecekan akses terhadap user yang sedang login. Pada *controller* pertama kali akan mendeklarasikan rute untuk fungsi tersebut dan juga *methods* yang digunakan. *@login_required* merupakan decorator yang berguna untuk mencegah pengguna yang tidak

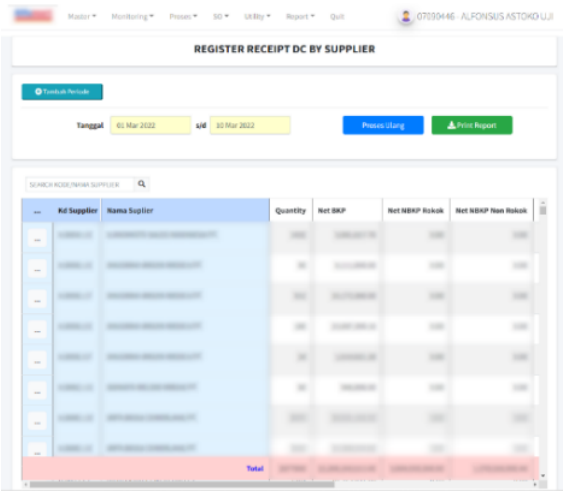
masuk untuk mengakses rute tersebut. *Flash* merupakan salah satu fungsi yang disediakan *flask* untuk menampilkan notifikasi sebagai respon dari hasil yang dikembalikan. *Flash* di panggil dengan mendeklarasikan notifikasi apa yang akan ditampilkan dan juga nilai nya (T/F). Bagian dari *flask* yang digunakan juga adalah *redirect*. Ini digunakan untuk memanggil langsung fungsi yang ada dalam *controller*. Terakhir adalah *render_template*, ini digunakan untuk menjalankan file HTML yang dituju.

4.4. Tampilan Aplikasi



Gambar 3 Halaman Login Aplikasi

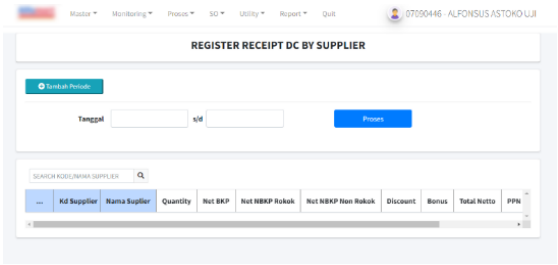
Gambar 3 merupakan tampilan halaman *login* aplikasi *Monitoring Stock*. Pada halaman ini, pengguna dapat memilih NIK yang sudah terdaftar untuk bisa akses aplikasi dengan cara klik pada kolom berwarna kuning yang nantinya akan muncul LOV berisikan NIK dan nama pengguna yang terdaftar. Setelah memilih NIK maka bagian kolom nama dan kode store akan terisi otomatis, pengguna cukup memasukan *password*.



Gambar 4 Halaman Menu Register

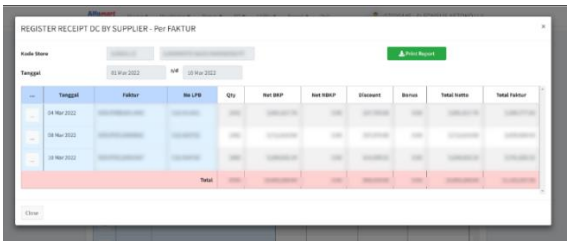
Pada gambar 4 menunjukan salah satu menu yang memiliki fungsi *read*. Pengguna perlu untuk memilih tanggal dengan cara klik pada kolom berwarna kuning untuk menampilkan LOV tanggal yang sudah diproses. Setelah memilih tanggal maka data berdasarkan tanggal tersebut akan ditampilkan pada tabel.

Pada halaman ini juga ada beberapa *button* seperti tambah periode, proses ulang dan *print report*. Tambah periode ini berguna untuk menambahkan periode baru yang belum ada pada LOV tanggal. Sedangkan proses ulang digunakan untuk memperbarui data pada tabel dan yang terakhir *print report* digunakan untuk mencetak data yang ditampilkan dalam bentuk *excel*.



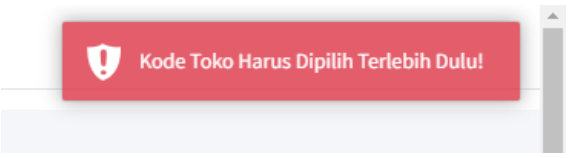
Gambar 5 Halaman Tambah Periode

Pada gambar 5 adalah ketika pengguna menekan tombol Tambah Periode. Yang terjadi saat tombol tersebut dipilih tombol Proses Ulang akan berubah menjadi tombol Proses, tombol Print Report menghilang, data yang tadi ditampilkan pada tabel akan hilang dan yang terakhir LOV pada tanggal berubah menjadi *datepicker*. Setelah memasukan periode yang baru dan klik tombol Proses, maka aplikasi akan menjalankan prosedur proses dan langsung menampilkan data pada tanggal tersebut.



Gambar 6 Register Per Faktur

Fungsi *delete* dapat ditemukan pada menu Register Per Faktur. Pada menu ini setelah pengguna memilih salah satu data maka akan muncul modal detail per fakturnya. Dan ketika modal ini ditutup akan langsung menjalankan fungsi *delete* pada bagian *back-end*.



Gambar 7 Notifikasi Flash dari Flask

Pada gambar 7 merupakan notifikasi dengan menggunakan *Flash* yang disediakan dari *framework flask*. Notifikasi ini merupakan respon error karena tidak ada kode toko yang dipilih. Pada

kode program parameter *flash* yang digunakan adalah F.

4.5. Testing

Pengujian aplikasi *Monitoring Stock* yang digunakan adalah *black box testing.* Pengujian ini dilakukan oleh tim *Quality Assurance* (QA) perusahaan.QA merupakan tim yang dibentuk perusahan untuk menguji sistem atau aplikasi sebelum digunakan langsung oleh *user*. Pengujian ini bertujuan untuk menemukan *bug* kesalahan fungsi pada aplikasi.

Pengujian ini memilih *black box testing* karena menguji perangkat lunak dan kelancaran program yang telah dibuat tanpa menguji desain dan mengetahui kode program yang dipakai [18]. *Black box* ini digunakan untuk melakukan pengujian terhadap hasil dari proses input maupun output pada aplikasi[19]. Adapun hasil dari pengujian tersebut adalah sebagai berikut:

Tabel 1. Tabel Blackbox Testing

Skenario Pengujian	Hasil yang Diharapkan	Hasil Pengujian	Status Pengujian
Memproses toko-toko baru di periode baru yang belum diinputkan	Sistem memproses untuk toko yang terpilih dengan menjalankan procedure untuk melakukan pengecekan dan penginputan toko toko tersebut, ketika proses berhasil akan menampilkan notifikasi 'Proses Berhasil'	Sistem menerima data toko dan menjalankan procedure kemudian menampilkan notifikasi 'Proses Berhasil' kemudian kembali ke halaman awal.	Valid
Melakukan proses ulang toko pada periode tertentu untuk memperbarui data pada toko dan tanggal tersebut	Sistem menampilkan modal sebelum di lakukannya pemrosesan ulang untuk toko tersebut. Setelahnya sistem akan menjalankan procedure untuk memperbarui data dan menampilkan pada tabel data	Sistem menerima toko yang dipilih dan menampilkan modal 'Toko sudah di proses. Lakukan proses ulang?' kemudian menjalan kan procedure dan setelah data selesai di perbarui akan langsung ditampilkan pada tabel data	Valid
Melakukan pengecekan apakah toko bisa diproses pada periode tersebut	Sistem menajalankan procedure untuk membaca apakah masih bisa dilakukan perubahan pada periode yang terpilih atau sudah tidak bisa kemudian menampilkan notifikasi keterangan	Sistem menerima data tanggal dan membaca procedure pengecekan. jika hasil dari procedure masih bisa dilakukan perubahan pada periode tersebut maka pemrosesan akan dilakukan, jika tidak sistem menampilkan notifikasi 'Periode sudah close. Proses tidak dapat di lakukan'	Valid
Pengecekan toko pada tampilan LOV berdasarkan inputan periode	Sistem menampilkan toko tertentu yang ada di periode tersebut	Sistem menjalankan query untuk pengecekan periode dan menampilkan toko yang ada di periode tersebut	Valid
Mengosongkan form dan datatabel saat melakukan penginputan toko baru	Sistem akan mengkosongkan tabel dan semua form input ketika button input toko di tekan	Sistem menerima trigger onclick saat button input di klik dan mehapus semua value di form input dan data pada tabel	Valid

Sorting pada bebarapa kolom data tabel	Sistem akan mengurutkan data pada tabel berdasarkan button sorting yang dipilih kemudian menampilakan notifikasi sorting selesai	Sistem menerima trigger button sorting yang dipilih kemudian akan menjalankan fungsi sorting setelah sorting selesai menampilkan notifikasi 'Sorting selesai di lakukan'	Valid
Search berdasarkan PLU dan deskripsi	Sistem akan menampilkan data pada tabel saat user inputkan inputan pada form searching	Sistem menerima inputan dari form searching kemudian hanya menampilkan data yang sesuai dengan inputan	Valid
Menampilkan data stok dan total akhirnya	Sistem menampilkan data tabel sesuai dengan periode dan toko yang dipilih	Sistem menerima inputan periode dan toko yang dipilih lalu menampilkan data berdasarkan parameter tersebut	Valid
Cetak report dalam bentuk excel	Sistem otomatis mengunduh berkas report dalam bentuk excel sesuai dengan periode dan toko yang dipilih	Sistem menerima inputan periode dan toko yang dipilih lalu otomatis mengunduh report untuk toko pada periode tersebut dalam bentuk excel	Valid
Enable button cetak dan proses ulang saat periode dan toko sudah terisi	Sistem otomatis mengaktifkan button untuk cetak report dan proses ulang saat ada inputan periode dan toko	Sistem menerima inputan periode dan toko yang dipilih lalu button untuk cetak dan proses ulang otomatis aktif	Valid

Berdasarkan hasil dari *Blackbox Testing* (Tabel 1) yang sudah dilakukan dapat disimpulkan bahwa semua fungsi sistem dapat berfungsi dengan baik. Hal ini dapat dilihat dari hasil pengujian yang memenuhi semua hasil yang diharapkan.

5. KESIMPULAN DAN SARAN

5.1. Kesimpulan

Berdasarkan pembahasan dan pemaparan diatas dapat disimpulkan bahwa dengan penggunaan *framework flask* pada pembangunan aplikasi *Monitoring Stock* dapat lebih mempermudah pengembang. Hal ini dikarenakan banyaknya *library* yang sudah disediakan oleh *framework flask*. Dengan begitu pengembang dapat lebih cepat dalam membangun aplikasi.

Dan dengan adanya aplikasi *monitoring stock* pengecekan stok barang pada *distribution center* dan toko dapat lebih terkontrol dengan baik. Pada aplikasi menampilkan barang yang masih tersedia dan sudah tidak tersedia. Selain itu, jika terdapat selisih stok barang pada *distribution center* dan toko aplikasi akan langsung menampilkan secara rinci berapa dan dibagian mana selisih tersebut.

Kelebihan dengan menggunakan *framework flask* ini salah satunya pada bagian konfigurasi yang ada pada *settings.py*. Dengan mengumpulkan semua baris kode konfigurasi pada *file* tersebut mempermudah pengembang dalam mengatur konfigurasi dan mengubahnya jika diperlukan sewaktu-waktu. Dengan memanfaatkan *render_template* juga mempermudah untuk menjalankan *file* HTML yang dituju.

5.2. Saran

Saran dari penulis untuk pengembangan aplikasi ini yaitu penggunaan *jinja template* saat menampilkan data dari *database* ke dalam data tabel agar kode program lebih ringkas lagi. Dan mungkin untuk tampilan pada *front-end* yang memiliki fungsi sama dapat disajikan dengan warna dan ukuran yang sama agar pengguna mudah mengingatnya.

DAFTAR PUSTAKA

[1] E. Soliha, "ANALISIS INDUSTRI RITEL DI INDONESIA," *Jurnal Bisnis dan Ekonomi (JBE)*, vol. 15, no. 2, pp. 128–142, 2008.

[2] I. D. Pramudiana, "PERUBAHAN PERILAKU KONSUMTIF MASYARAKAT DARI PASAR TRADISIONAL KE PASAR MODERN."

[3] Alfamart, "https://alfamart.co.id/tentang-perusahaan/profil-kami."

[4] T. Wijaya Dan *et al.*, "Penerapan Kontrol Stok dalam… ν Penerapan Kontrol Stok dalam Sistem Informasi Dagang Dengan Metode Perpetual Inventory System".

[5] R. R. Saragih, "PEMROGRAMAN DAN BAHASA PEMROGRAMAN."

[6] Enterprise Jubilee, *Python untuk Programmer Pemula*, 1st ed., vol. 1. Jakarta: Gramedia, 2019.

[7] Irsyad Rahadian, "Penggunaan Python Web Framework Flask Untuk Pemula," 2018.

[8] M. Grinberg, "Flask Web Development."

[9] A. Asroni, "Penerapan Model View Controller (MVC) Dengan Framework Codeigniter Pada Sistem Informasi Booking Wisata Klangon," *BERDIKARI : Jurnal Inovasi dan Penerapan Ipteks*, vol. 6, no. 2, 2018, doi: 10.18196/bdr.6239.

[10] "Documentation Flask." https://flask.palletsprojects.com/en/2.0.x/ (accessed Nov. 19, 2021).

[11] The PostgreSQL Global Development Group, "Documentation PostgreSQL." https://www.postgresql.org (accessed Nov. 19, 2021).

[12] S. Munawaroh, "Mengeksplorasi Database PostgreSQL dengan PgAdmin III," *Jurnal Teknologi Informasi DINAMIK*, vol. X, no. 2, pp. 103–107, 2005.

[13] M. Hasanudin, "RANCANG DAN BANGUN SISTEM INFORMASI INVENTORI BARANG BERBASIS WEB (STUDI KASUS PT. NUSANTARA SEJAHTERA RAYA)."

[14] D. F. Ningtyas and N. Setiyawati, "Implementasi Flask Framework pada Pembangunan Aplikasi Purchasing Approval Request Flask Framework Implementation in Development Purchasing Approval Request Application," *Jurnal Janitra Informatika dan Sistem Informasi*, vol. 1, no. 1, pp. 19–34, 2021, doi: 10.25008/janitra.v1i1.120.

[15] R. T. Handayanto and H. Herlawati, "Machine Learning Berbasis Desktop dan Web dengan Metode Jaringan Syaraf Tiruan Untuk Sistem Pendukung Keputusan," *Jurnal Komtika (Komputasi dan Informatika)*, vol. 4, no. 1, pp. 15–26, Jun. 2020, doi: 10.31603/komtika.v4i1.3698.

[16] A. A. Wahid, "Analisis Metode Waterfall Untuk Pengembangan Sistem Informasi".

[17] P. Simanjuntak and A. Kasnady, "ANALISIS MODEL VIEW CONTROLLER (MVC) PADA BAHASA PHP," 2016.

[18] F. C. Ningrum, D. Suherman, S. Aryanti, H. A. Prasetya, and A. Saifudin, "Pengujian Black Box pada Aplikasi Sistem Seleksi Sales Terbaik Menggunakan Teknik Equivalence Partitions," vol. 4, no. 4, 2019, [Online]. Available: http://openjournal.unpam.ac.id/index.php/informatika

[19] M. M. Gultom and Maryam, "Sistem Informasi Penjualan Material Bangunan Pada Toko Bangunan Berkah Information System of Sales Building Material (Case Study : Berkah Building Shop)," *J. Tek. Inform.*, vol. 1, no. 2, pp. 79–86, 2020