

LAPORAN

SISTEM STOCK MANAGEMENT

Diajukan Sebagai Tugas Akhir Mata Kuliah Praktikum Berorientasi Obyek



DOSEN PENGAMPU MATA KULIAH :

M. Bahrul Subkhi, M. Kom

Disusun Oleh :

Anggota Kelompok

1. Apliana Sriyanti Bili (2213020242)
2. Virginia Abuk Klau (2213020019)
3. Danika Ambadar Alfari (2213020256)

**PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS NUSANTARA PGRI KEDIRI**

2024

DAFTAR ISI

HALAMAN JUDUL	i
DAFTAR ISI	ii
BAB I	1
PENDAHULUAN	1
A. Latar Belakang	1
B. Rumusan Masalah.....	1
C. Tujuan.....	1
D. Claas Diagram	1
BAB II	2
A. Hasil Program dan Penjelasan	2
BAB III	2
A. Kesimpulan	3
B. Saran	3
DAFTAR PUSTAKA.....	4

BAB I

PENDAHULUAN

A. Latar Belakang

Perkembangan teknologi informasi khususnya teknologi informasi berbasis system stok management ini, dirasa sangat pesat dan hal ini berpengaruh terhadap aspek pekerjaan. Hampir semua perusahaan dalam hal pengambilan keputusan, penyebaran informasi, peningkatan efektifitas pekerjaan dan pelayanan telah menggunakan sistem informasi komputer, dan tidak menutup kemungkinan lebih memudahkan lagi apabila semua itu dilakukan dengan menggunakan sistem aplikasi system management, Bagi suatu perusahaan yang sedang berkembang seperti pada perusahaan produk barang yang bergerak di bidang produksi, tentunya suatu sistem inventory yang berguna untuk mengelola persediaan stok barang. Namun pendokumentasian yang digunakan saat ini kebanyakan secara manual atau masih terkomputerisasi sehingga menimbulkan kendala waktu maupun teknis dan pengetahuan dalam kinerja perusahaan. Pada mulanya perdagangan melalui internet dilakukan di desktop website, dimana untuk mengakses sebuah desktop website dibutuhkan perangkat komputer dan laptop. Meskipun dapat diakses menggunakan perangkat system stok managemen (handphone dan smartphone) situs-situs yang berbasis desktop website ini akan menjadi beban untuk perangkat tersebut. Selain karena membutuhkan koneksi internet yang besar, perangkat yang digunakan untuk mengakses desktop website tidak memiliki mobilitas tinggi khususnya komputer (Stefanus, 2013). Menurut (Fatmanto, 2013) dilakukan perancangan aplikasi yang difokuskan untuk membuat forum jual beli suatu barang dan proses transaksi yang aman bagi pengguna. Dikarenakan semakin banyaknya penjual barang online perseorangan dan juga pembeli atau penjual yang tidak tahu dimana dia dapat membeli suatu barang atau mengiklankan barangnya,

maka aplikasi ini ditujukan untuk mengumpulkan penjual dan pembeli di dalam suatu forum agar informasi yang didapatkan lebih mudah dan akurat.

B. Rumusan Masalah

1. Ketidakefisienan Persediaan:

Bagaimana meningkatkan efisiensi penyimpanan dan manajemen persediaan agar dapat mengurangi biaya penyimpanan yang tidak perlu?

2. Kehilangan Stok dan Kecurangan:

Bagaimana mencegah atau mengurangi kehilangan stok dan risiko kecurangan dalam sistem manajemen stok?

3. Optimalisasi Pemesanan:

Bagaimana meningkatkan proses pemesanan untuk menghindari kekurangan stok atau kelebihan persediaan yang tidak perlu?

4. Pengendalian Kualitas dan Keamanan Persediaan:

Bagaimana memastikan kualitas dan keamanan persediaan, terutama untuk produk yang memiliki batas waktu kadaluwarsa?

5. Pelacakan Stok Real-time:

Bagaimana meningkatkan kemampuan untuk melacak stok secara real-time guna mengidentifikasi perubahan cepat dalam permintaan atau persediaan?

C. Tujuan

1. Optimalisasi Persediaan:

Meminimalkan biaya persediaan sambil memastikan ketersediaan produk yang cukup.

2. Efisiensi Operasional:

Meningkatkan efisiensi proses pengelolaan stok untuk mengurangi biaya operasional.

3. Peningkatan Akurasi Prediksi:

Meningkatkan akurasi dalam meramalkan permintaan produk untuk menghindari kelebihan atau kekurangan stok.

4. Transparansi Stok:

Memberikan visibilitas yang tinggi terhadap stok secara real-time untuk memudahkan pengambilan keputusan.

5. Keamanan dan Pelacakan:

Memastikan keamanan stok dan kemampuan untuk melacak pergerakan barang dari gudang hingga pelanggan.

D. Class Diagram

Diagram memperlihatkan class, atribut, metode, dan hubungan antar kelas dalam sistem Class diagram adalah salah satu jenis diagram struktur dalam UML yang digunakan untuk menggambarkan detail struktur dengan jelas Class

Berikut adalah beberapa fungsi utama dari class diagram:

1. Menunjukkan struktur dari suatu sistem dengan jelas
2. Meningkatkan pemahaman tentang gambaran umum atau skema dari suatu sistem
3. Membantu kamu untuk menyampaikan kebutuhan dari suatu system

Class diagram memiliki komponen penyusun, yang mencakup nama class, atribut, dan metode

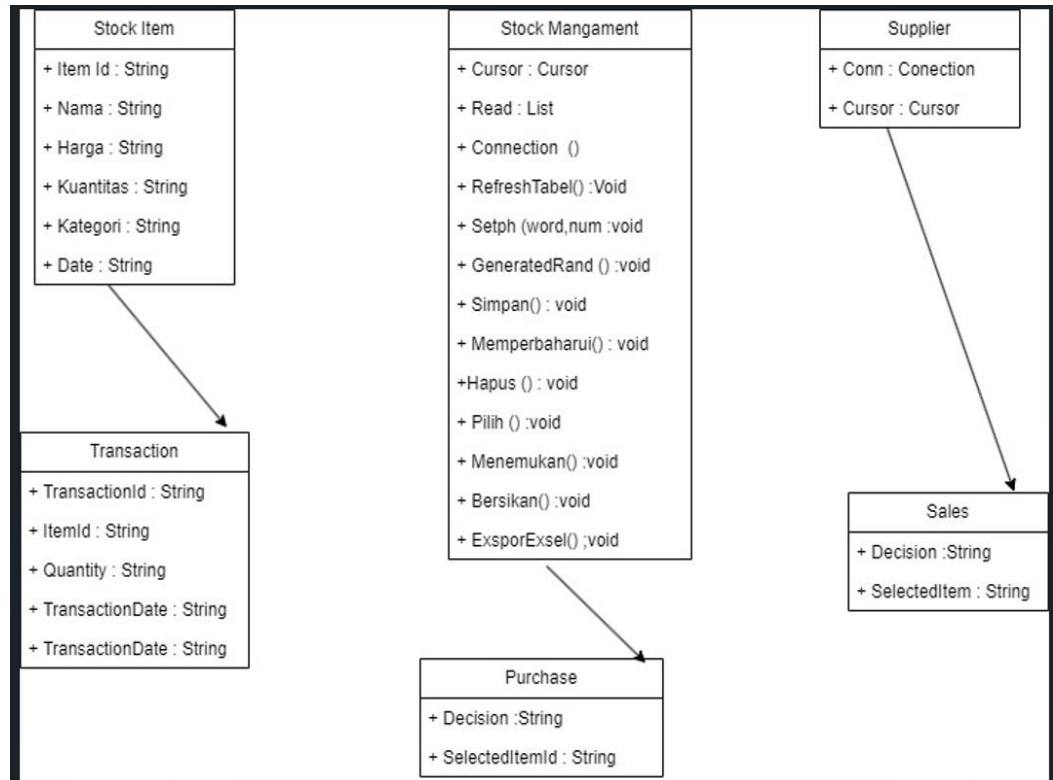
Ada tiga hubungan dalam diagram kelas: asosiasi, agregasi, dan pewarisan

- a. Asosiasi: Hubungan antara dua class yang bersifat statis, dapat diartikan sebagai hubungan antara class yang memiliki atribut tambahan seperti class lain
- b. Agregasi: Hubungan antara dua class di mana salah satu class merupakan bagian dari class lain, tetapi dua class ini dapat berdiri masing-masing
- c. Pewarisan: Kemampuan suatu class untuk mewarisi seluruh manfaat dari class yang diwarisi

Class diagram sering disebut dengan diagram struktur dan memiliki sifat statis, yakni hanya mampu menjelaskan hubungan apa yang terjadi, bukan menjelaskan apa yang terjadi jika kelasnya berhubungan

Class diagram dapat digunakan untuk memodelkan sistem dari sebuah perspektif bisnis dan merupakan satu-satunya diagram UML yang dapat dipetakan langsung menggunakan bahasa pemrograman

Berikut ini gambar class diagram



BAB II

PEMBAHASAN

A. Hasil Program dan Penjelasan

```
2 from tkinter import ttk
3 from tkinter import messagebox
4 import tkinter
5 import random
6 import pymysql
7 import csv
8 from datetime import datetime
9 import numpy as np
10
11 window=tkinter.Tk()
12 window.title("Stock Management System")
13 window.geometry("720x640")
14 my_tree=ttk.Treeview(window,show='headings',height=20)
15 style=ttk.Style()
16
17 placeholderArray=['',' ',' ',' ',' ']
18 numeric='1234567890'
19 alpha='ABCDEFGHIJKLMNOPQRSTUVWXYZ'
20
21 def connection():
22     conn=pymysql.connect([
23         host='localhost',
24         user='root',
25         password='',
26         db='stockmanagementsystem'
27     ])
28     return conn
29
30 conn=connection()
31 cursor=conn.cursor()
32
33 for i in range(0,5):
34     placeholderArray[i]=tkinter.StringVar()
35
36 def read():
37     cursor.connection.ping()
38     sql=f"SELECT `item_id`, `name`, `price`, `quantity`, `category`, `date` FROM stocks ORDER BY `id` DESC"
39     cursor.execute(sql)
40     results=cursor.fetchall()
41     conn.commit()
42     conn.close()
43     return results
44
45 def refreshTable():
46     for data in my_tree.get_children():
47         my_tree.delete(data)
48     for array in read():
49         my_tree.insert(parent='',index='end',iid=array,text="",values=(array),tag="orow")
50     my_tree.tag_configure('orow',background="#EEEEEE")
51     my_tree.pack()
52
53 def setph(word,num):
54     for ph in range(0,5):
55         if ph == num:
56             placeholderArray[ph].set(word)
57
58 def generateRand():
59     itemId=''
60     for i in range(0,3):
61         randno=random.randrange(0,(len(numeric)-1))
62         itemId=itemId+str(numeric[randno])
63     randno=random.randrange(0,(len(alpha)-1))
64     itemId=itemId+'-'+str(alpha[randno])
65     print("generated: "+itemId)
66     setph(itemId,0)
67
68 def save():
69     itemId=str(itemIdEntry.get())
70     name=str(nameEntry.get())
71     price=str(priceEntry.get())
72     qnt=str(qntEntry.get())
```



```

73 cat=str(categoryCombo.get())
74 valid=True
75 if not(itemId and itemId.strip()) or not(name and name.strip()) or not(price and price.strip()) or not(qnt and qnt.strip()) or not(cat and
76     messagebox.showwarning("", "Silakan isi semua entri")
77     return
78 if len(itemId) < 5:
79     messagebox.showwarning("", "Id Barang Tidak Valid")
80     return
81 if(not(itemId[3]=='-')):
82     valid=False
83 for i in range(0,3):
84     if(not(itemId[i] in numeric)):
85         valid=False
86         break
87 if(not(itemId[4] in alpha)):
88     valid=False
89 if not(valid):
90     messagebox.showwarning("", "Id Barang Tidak Valid")
91     return
92 try:
93     cursor.connection.ping()
94     sql=f"SELECT * FROM stocks WHERE `item_id` = '{itemId}' "
95     cursor.execute(sql)
96     checkItemNo=cursor.fetchall()
97     if len(checkItemNo) > 0:
98         messagebox.showwarning("", "Id Item sudah digunakan")
99         return
100     else:
101         cursor.connection.ping()
102         sql=f"INSERT INTO stocks (`item_id`, `name`, `price`, `quantity`, `category`) VALUES ('{itemId}', '{name}', '{price}', '{qnt}', '{cat}'
103         cursor.execute(sql)
104         conn.commit()
105         conn.close()
106         for num in range(0,5):
107             setph('', num)
108 except Exception as e:

```

```

    print(e)
    messagebox.showwarning("", "Error while saving ref: "+str(e))
    return
def refreshTable()

def update():
    selectedItemId = ''
    try:
        selectedItem = my_tree.selection()[0]
        selectedItemId = str(my_tree.item(selectedItem)['values'][0])
    except:
        messagebox.showwarning("", "Silakan pilih baris data")
    print(selectedItemId)
    itemId = str(itemIdEntry.get())
    name = str(nameEntry.get())
    price = str(priceEntry.get())
    qnt = str(qntEntry.get())
    cat = str(categoryCombo.get())
    if not(itemId and itemId.strip()) or not(name and name.strip()) or not(price and price.strip()) or not(qnt and qnt.strip()) or not(cat and
        messagebox.showwarning("", "Silakan isi semua entri")
        return
    if(selectedItemId!=itemId):
        messagebox.showwarning("", "Anda tidak dapat mengubah ID Item")
        return
    try:
        cursor.connection.ping()
        sql=f"UPDATE stocks SET `name` = '{name}', `price` = '{price}', `quantity` = '{qnt}', `category` = '{cat}' WHERE `item_id` = '{itemId}' "
        cursor.execute(sql)
        conn.commit()
        conn.close()
        for num in range(0,5):
            setph('', num)
    except Exception as err:
        messagebox.showwarning("", "Terjadi kesalahan ref: "+str(err))
        return
    refreshTable()

```

```

145 def delete():
146     try:
147         if(my_tree.selection()[0]):
148             decision = messagebox.askquestion("", "Hapus data yang dipilih?")
149             if(decision != 'yes'):
150                 return
151             else:
152                 selectedItem = my_tree.selection()[0]
153                 itemId = str(my_tree.item(selectedItem)['values'][0])
154                 try:
155                     cursor.connection.ping()
156                     sql=f"DELETE FROM stocks WHERE `item_id` = '{itemId}' "
157                     cursor.execute(sql)
158                     conn.commit()
159                     conn.close()
160                     messagebox.showinfo("", "Data telah berhasil dihapus")
161                 except:
162                     messagebox.showinfo("", "Maaf, terjadi kesalahan")
163                 refreshTable()
164             except:
165                 messagebox.showwarning("", "Silakan pilih baris data")
166
167 def select():
168     try:
169         selectedItem = my_tree.selection()[0]
170         itemId = str(my_tree.item(selectedItem)['values'][0])
171         name = str(my_tree.item(selectedItem)['values'][1])
172         price = str(my_tree.item(selectedItem)['values'][2])
173         qnt = str(my_tree.item(selectedItem)['values'][3])
174         cat = str(my_tree.item(selectedItem)['values'][4])
175         setph(itemId, 0)
176         setph(name, 1)
177         setph(price, 2)
178         setph(qnt, 3)
179         setph(cat, 4)

```

```

181         except:
182             messagebox.showwarning("", "Silakan pilih baris data")
183
184     def find():
185         itemId = str(itemIdEntry.get())
186         name = str(nameEntry.get())
187         price = str(priceEntry.get())
188         qnt = str(qntEntry.get())
189         cat = str(categoryCombo.get())
190         cursor.connection.ping()
191         if(itemId and itemId.strip()):
192             sql = f"SELECT `item_id`, `name`, `price`, `quantity`, `category`, `date` FROM stocks WHERE `item_id` LIKE '%{itemId}%' "
193         elif(name and name.strip()):
194             sql = f"SELECT `item_id`, `name`, `price`, `quantity`, `category`, `date` FROM stocks WHERE `name` LIKE '%{name}%' "
195         elif(price and price.strip()):
196             sql = f"SELECT `item_id`, `name`, `price`, `quantity`, `category`, `date` FROM stocks WHERE `price` LIKE '%{price}%' "
197         elif(qnt and qnt.strip()):
198             sql = f"SELECT `item_id`, `name`, `price`, `quantity`, `category`, `date` FROM stocks WHERE `quantity` LIKE '%{qnt}%' "
199         elif(cat and cat.strip()):
200             sql = f"SELECT `item_id`, `name`, `price`, `quantity`, `category`, `date` FROM stocks WHERE `category` LIKE '%{cat}%' "
201         else:
202             messagebox.showwarning("", "silakan isi salah satu entri")
203             return
204         cursor.execute(sql)
205         try:
206             result = cursor.fetchall()
207             for num in range(0,5):
208                 setph(result[0][num],(num))
209             conn.commit()
210             conn.close()
211         except:
212             messagebox.showwarning("", "Tidak ada data ditemukan")
213
214     def clear():
215         for num in range(0,5):
216             setph('',(num))
217
218     def exportExcel():
219         cursor.connection.ping()
220         sql=f"SELECT `item_id`, `name`, `price`, `quantity`, `category`, `date` FROM stocks ORDER BY `id` DESC"
221         cursor.execute(sql)
222         dataraw=cursor.fetchall()
223         date = str(datetime.now())
224         date = date.replace(' ', '_')
225         date = date.replace(':', '-')
226         dateFinal = date[0:16]
227         with open("stocks_"+dateFinal+".csv",'a',newline='') as f:
228             w = csv.writer(f, dialect='excel')
229             for record in dataraw:
230                 w.writerow(record)
231         print("saved: stocks_"+dateFinal+".csv")
232         conn.commit()
233         conn.close()
234         messagebox.showinfo("", "File Excel diunduh")
235
236     frame=tkinter.Frame(window,bg="#02577A")
237     frame.pack()
238
239     btnColor="#196E78"
240
241     manageFrame=tkinter.LabelFrame(frame,text="Manage",borderwidth=5)
242     manageFrame.grid(row=0,column=0,sticky="w",padx=[10,200],pady=20,ipadx=[6])
243
244     saveBtn=Button(manageFrame,text="SIMPAN",width=10,borderwidth=3,bg=btnColor,fg='white',command=save)
245     updateBtn=Button(manageFrame,text="PERBAHARUI",width=10,borderwidth=3,bg=btnColor,fg='white',command=update)
246     deleteBtn=Button(manageFrame,text="HAPUS",width=10,borderwidth=3,bg=btnColor,fg='white',command=delete)
247     selectBtn=Button(manageFrame,text="PILIH",width=10,borderwidth=3,bg=btnColor,fg='white',command=select)
248     findBtn=Button(manageFrame,text="MENEMUKAN",width=10,borderwidth=3,bg=btnColor,fg='white',command=find)
249     clearBtn=Button(manageFrame,text="BERSIHKAN",width=10,borderwidth=3,bg=btnColor,fg='white',command=clear)
250     exportBtn=Button(manageFrame,text="EXPOR EXCEL",width=15,borderwidth=3,bg=btnColor,fg='white',command=exportExcel)

```

```

251 saveBtn.grid(row=0,column=0,padx=5,pady=5)
252 updateBtn.grid(row=0,column=1,padx=5,pady=5)
253 deleteBtn.grid(row=0,column=2,padx=5,pady=5)
254 selectBtn.grid(row=0,column=3,padx=5,pady=5)
255 findBtn.grid(row=0,column=4,padx=5,pady=5)
256 clearBtn.grid(row=0,column=5,padx=5,pady=5)
257 exportBtn.grid(row=0,column=6,padx=5,pady=5)
258
259
260 entriesFrame=tkinter.LabelFrame(frame,text="Form",borderwidth=5)
261 entriesFrame.grid(row=1,column=0,sticky="w",padx=[10,200],pady=[0,20],ipadx=[0])
262
263 itemIdLabel=Label(entriesFrame,text="IDENTITAS",anchor="e",width=10)
264 nameLabel=Label(entriesFrame,text="NAMA",anchor="e",width=10)
265 priceLabel=Label(entriesFrame,text="HARGA",anchor="e",width=10)
266 qntLabel=Label(entriesFrame,text="KUANTITAS",anchor="e",width=10)
267 categoryLabel=Label(entriesFrame,text="KATEGORI",anchor="e",width=10)
268
269 itemIdLabel.grid(row=0,column=0,padx=10)
270 nameLabel.grid(row=1,column=0,padx=10)
271 priceLabel.grid(row=2,column=0,padx=10)
272 qntLabel.grid(row=3,column=0,padx=10)
273 categoryLabel.grid(row=4,column=0,padx=10)
274
275 categoryArray=['Alat Jaringan','BAGIAN KOMPUTER','Alat Perbaikan','gadget']
276
277 itemIdEntry=Entry(entriesFrame,width=50,textvariable=placeholderArray[0])
278 nameEntry=Entry(entriesFrame,width=50,textvariable=placeholderArray[1])
279 priceEntry=Entry(entriesFrame,width=50,textvariable=placeholderArray[2])
280 qntEntry=Entry(entriesFrame,width=50,textvariable=placeholderArray[3])
281 categoryCombo=ttk.Combobox(entriesFrame,width=47,textvariable=placeholderArray[4],values=categoryArray)
282
283 itemIdEntry.grid(row=0,column=2,padx=5,pady=5)
284 nameEntry.grid(row=1,column=2,padx=5,pady=5)
285 priceEntry.grid(row=2,column=2,padx=5,pady=5)
286 qntEntry.grid(row=3,column=2,padx=5,pady=5)
287
288 categoryCombo.grid(row=4,column=2,padx=5,pady=5)
289
290 generateIdBtn=Button(entriesFrame,text="MEMBUAT ID",borderwidth=3,bg=btnColor,fg='white',command=generateRand)
291 generateIdBtn.grid(row=0,column=3,padx=5,pady=5)
292
293 style.configure(window)
294 my_tree['columns']=(("Item Id","Name","Price","Quantity","Category","Date"))
295 my_tree.column("#0",width=0,stretch=NO)
296 my_tree.column("Item Id",anchor=W,width=70)
297 my_tree.column("Name",anchor=W,width=125)
298 my_tree.column("Price",anchor=W,width=125)
299 my_tree.column("Quantity",anchor=W,width=100)
300 my_tree.column("Category",anchor=W,width=150)
301 my_tree.column("Date",anchor=W,width=150)
302 my_tree.heading("Item Id",text="Identitas",anchor=W)
303 my_tree.heading("Name",text="Nama",anchor=W)
304 my_tree.heading("Price",text="harga",anchor=W)
305 my_tree.heading("Quantity",text="Kuantitas",anchor=W)
306 my_tree.heading("Category",text="Kategori",anchor=W)
307 my_tree.heading("Date",text="Tanggal & waktu",anchor=W)
308 my_tree.tag_configure('orow',background="#EEEEEE")
309 my_tree.pack()
310
311 refreshTable()
312
313 window.resizable(False,False)
314 window.mainloop()

```

- Penjelasan dari setiap alur codingan tersebut :

1. Imports:

Skrip ini dimulai dengan mengimpor modul-modul yang diperlukan seperti tkinter untuk GUI, ttk untuk widget Tkinter berthemes, messagebox untuk menampilkan kotak dialog, random untuk menghasilkan angka acak, pymysql untuk interaksi database MySQL, csv untuk menangani file CSV, dan datetime untuk bekerja dengan tanggal dan waktu.

2. Pengaturan Jendela:

Membuat jendela Tkinter (window) dengan judul, ukuran, dan geometri tertentu.

3. Pengaturan Treeview:

Membuat widget ttk.Treeview bernama my_tree untuk menampilkan data tabular (dalam hal ini, informasi stok). Mendefinisikan kolom dan judul untuk treeview.

4. Koneksi Database:

Mendefinisikan fungsi connection() untuk menjalin koneksi ke database MySQL. Parameter koneksi (host, user, password, db) ditentukan.

5. Inisialisasi Placeholder:

Membuat array placeholderArray untuk menyimpan instansi Tkinter StringVar. Ini akan digunakan untuk mengatur dan mendapatkan nilai dari berbagai widget masukan. Fungsi

6. Baca (Read):

Fungsi read() didefinisikan untuk mengambil data dari tabel database bernama 'stocks' dan mengembalikan hasilnya. Fungsi Perbarui Tabel (Refresh Table): Fungsi refreshTable() didefinisikan untuk menghapus data yang ada di treeview dan mengisinya dengan data terbaru dari database.

7. Fungsi Set Placeholder:

Fungsi setph(word, num) didefinisikan untuk mengatur nilai placeholderArray pada indeks tertentu.

8. Fungsi Generate Random ID:

Fungsi generateRand() menghasilkan ID item acak berdasarkan kombinasi angka dan alfabet.

9. Fungsi Simpan (Save):

Fungsi `save()` dipicu saat tombol "SIMPAN" diklik. Memvalidasi data masukan, memeriksa duplikasi ID item, dan menyisipkan rekaman baru ke dalam database.

10. Fungsi Perbarui (Update):

Fungsi `update()` dipicu saat tombol "PERBAHARUI" diklik. Memperbarui rekaman yang dipilih di database dengan nilai baru.

11. Fungsi Hapus (Delete):

Fungsi `delete()` dipicu saat tombol "HAPUS" diklik. Menghapus rekaman yang dipilih dari database setelah dikonfirmasi dengan pengguna.

12. Fungsi Pilih (Select):

Fungsi `select()` dipicu saat tombol "PILIH" diklik. Mengambil nilai rekaman yang dipilih dan mengisi widget masukan.

13. Fungsi Temukan (Find):

Fungsi `find()` dipicu saat tombol "MENEMUKAN" diklik. Mencari rekaman berdasarkan kriteria yang ditentukan pengguna dan mengisi widget masukan dengan rekaman pertama yang cocok.

14. Fungsi Bersihkan (Clear):

Fungsi `clear()` membersihkan nilai-nilai di widget masukan.

15. Fungsi Ekspor Excel:

Fungsi `exportExcel()` mengekspor seluruh data stok ke file CSV dengan nama berdasarkan tanggal dan waktu saat ini.

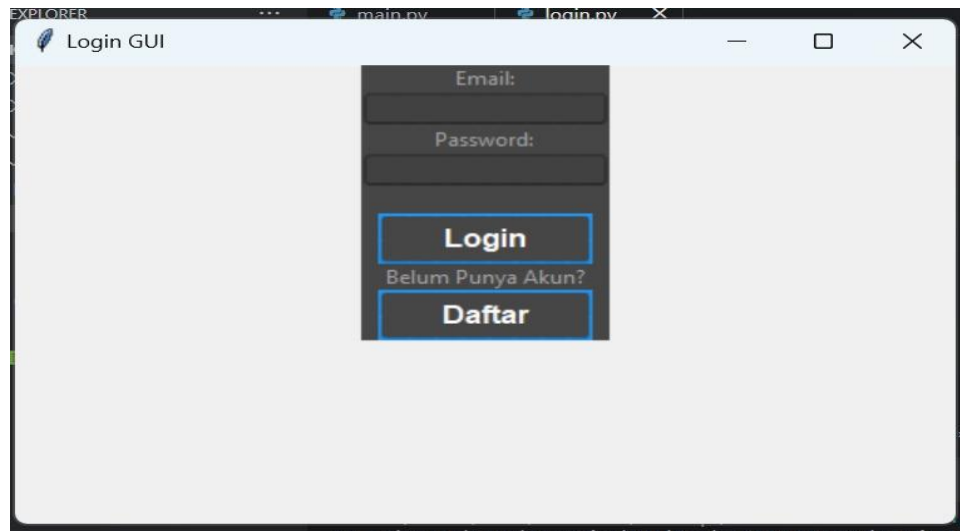
16. Tata Letak GUI:

Skrip menentukan tata letak GUI menggunakan frame, label, tombol, widget masukan, dan treeview untuk menampilkan data.

17. Perulangan Utama Jendela:

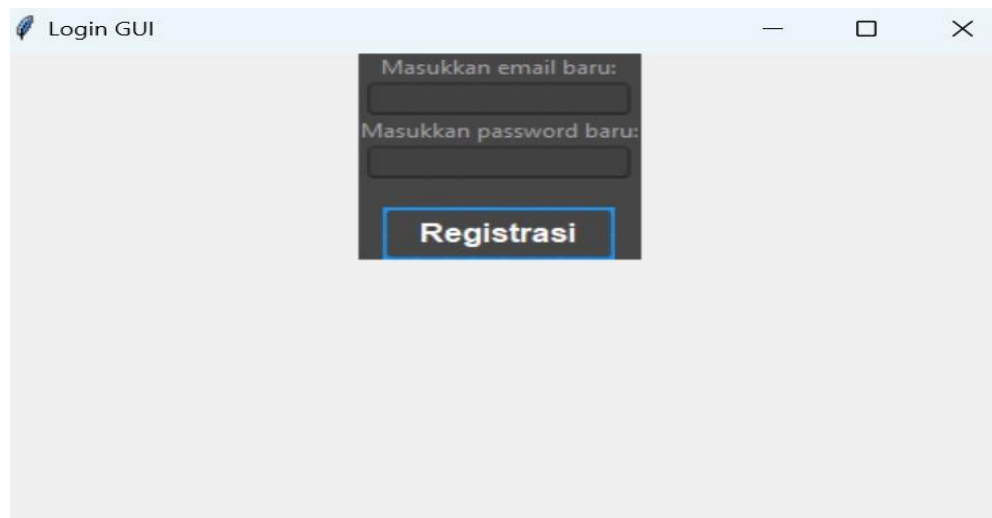
`mainloop()` Tkinter dipanggil untuk menjalankan aplikasi. Catatan: Skrip mengasumsikan adanya database MySQL bernama 'stockmanagementsystem' dengan tabel bernama 'stocks'. Struktur tabel 'stocks' tidak disediakan dalam kode, tetapi sepertinya memiliki kolom-kolom seperti 'item_id', 'name', 'price', 'quantity', 'category', dan 'date'. Kode tidak mengatasi pengecualian secara efektif dalam operasi database; perbaikan dapat dilakukan dalam hal penanganan kesalahan dan umpan balik pengguna.

- Tampilan Halaman Login :



The screenshot shows a window titled "Login GUI" with a light blue header bar. The main content area is light gray. In the center, there is a dark gray rectangular panel containing the following elements from top to bottom: an "Email:" label above a text input field, a "Password:" label above a text input field, a blue "Login" button, the text "Belum Punya Akun?" in a smaller font, and a blue "Daftar" button.

- Tampilan halman registrasi



The screenshot shows a window titled "Login GUI" with a light blue header bar. The main content area is light gray. In the center, there is a dark gray rectangular panel containing the following elements from top to bottom: a label "Masukkan email baru:" above a text input field, a label "Masukkan password baru:" above a text input field, and a blue "Registrasi" button.

- Tampilan Halaman Aplikasi GUI

The screenshot displays the 'Stock Management System' application window. It features a 'Manage' section with buttons for SIMPAN, PERBAHARUI, HAPUS, PILIH, MENEMUKAN, BERSIHKAN, and EXPOR EXCEL. Below this is a 'Form' section with input fields for IDENTITAS, NAMA, HARGA, Kuantitas, and KATEGORI, along with a MEMBUAT ID button. At the bottom, a table lists stock items with columns for Identitas, Nama, harga, Kuantitas, Kategori, and Tanggal & waktu.

Identitas	Nama	harga	Kuantitas	Kategori	Tanggal & waktu
429-X	AMD RYZEN 5	2000000	125	BAGIAN KOMPUTER	2024-01-02 05:54:12

BAB II

PENUTUP

A. KESIMPULAN

Setelah menyelesaikan penelitian Aplikasi Stok Barang Berbasis Web maka diperoleh kesimpulan sebagai berikut:

1. Aplikasi dapat digunakan untuk menyimpan data supplier, data harga, data barang, jenis barang, satuan barang, dan data user
2. Aplikasi dapat menampilkan dan mengunduh laporan transaksi barang masuk dan keluar
3. Meminimalisir terjadinya kehilangan dan kerusakan data karena data disimpan di dalam database
4. Waktu untuk mencari data barang lebih cepat sehingga dapat menghemat waktu
5. Keamanan data terjamin karena ketika ingin mengakses aplikasi, harus terlebih dahulu login dan perlu memasukkan username dan password yang terdaftar di database.

B. SARAN

Berdasarkan kesimpulan yang sudah diuraikan di atas, hal yang diharapkan atau saran untuk ke depannya adalah agar sistem yang dibangun dapat bekerja dengan lebih baik, perbaikan dari segi tampilan agar lebih menarik untuk dilihat serta penambahan fitur berupa monitoring stok agar memudahkan dalam mengendalikan stok barang baik itu masuk dari pengadaan maupun keluar untuk penjualan atau produksi.

DAFTAR PUSTAKA

A. S. Rosa., Shalahuddin, M. (2013). Rekayasa Perangkat Lunak Terstruktur Dan Berorientasi Objek. Bandung: Informatika.