TALLINNA TÖÖSTUSHARIDUSKESKUS

**ANDMEBAASI HALDAMINE RAAMATUKATALOOGIS**  
Arvestus töö

Õpetaja: Marina Oleinik  
Õpilane: Jelizaveta Ostapjuk  
Rühm: TARpv23

Tallinn  
2024

# SISUKORD

[SISUKORD 2](#_Toc166837308)

[SISEJUHATUS 3](#_Toc166837309)

[TEOREETILINE OSA 4](#_Toc166837310)

[PRAKTILINE OSA 5](#_Toc166837311)

[Ettevalmistused tööks 5](#_Toc166837312)

[Graafilise liidese loomine 5](#_Toc166837313)

[KASUTUSJUHEND 12](#_Toc166837314)

[KOKKUVÕTTE 14](#_Toc166837315)

[ALLIKAD 15](#_Toc166837316)

# Pildid

[Pilt 1. Andmete uuendamine 6](#_Toc166837090)

[Pilt 2. Andmete lisamine 7](#_Toc166837091)

[Pilt 3. Kirjete kustutamine 8](#_Toc166837092)

[Pilt 4. Andmete kuvamine 9](#_Toc166837093)

[Pilt 5. Algsete andmete lisamine 9](#_Toc166837094)

[Pilt 6. Tabeli loomine 10](#_Toc166837095)

[Pilt 7. Tabeli kustutamine 11](#_Toc166837096)

[Pilt 8. Programmi peaaken 12](#_Toc166837097)

# SISEJUHATUS

Projekti eesmärk on luua interaktiivne rakendus raamatukataloogi haldamiseks. See rakendus võimaldab kasutajatel vaadata teavet raamatute, autorite ja žanrite kohta ning lisada, uuendada ja kustutada andmeid andmebaasis. Seda rakendust saab kasutada tohutu hulk inimesi, alates raamatukogu töötajatest kuni lugemishuvilisteni isikliku raamatukogu kasutuselevõtuks. Raamatukataloogi haldamise rakenduse aktuaalsus seisneb selle tõhususes ja lihtsas haldamises, neid on vaja erinevatel tegevusaladel, näiteks akadeemilistel eesmärkidel või kaupluste ja raamatukogude jaoks.

Selle eesmärgi saavutamiseks kasutasin Visual Studio 2022 rakendust ja kirjutasin Pythoni keeles. Kasutas ka erinevaid abiraamatukogusid nagu tkinter ja sqlite3. Tkinterit kasutati väliskesta jaoks ja SQLite3 andmebaasi haldamiseks.

# TEOREETILINE OSA

**Python** - see on programmeerimiskeel, mida kasutatakse laialdaselt internetirakendustes, tarkvaraarenduses, andmeteaduses ja masinõppes (ML). Arendajad kasutavad Pythonit, sest see on tõhus, lihtne õppida ja töötab erinevatel platvormidel. Pythoni keele programme saab alla laadida tasuta, need ühilduvad igat tüüpi süsteemidega ja suurendavad arenduskiirust. (Amazon, kuupäev puudub)

**Tkinter** — See on Pythoni ristplatvormne graafiline liides, mis võimaldab töötada Tk-teegiga. See sisaldab graafilise kasutajaliidese (GUI - Graphical User Interface) elemente, mille abil saab luua erinevaid rakendusi. Algselt töötas Tkinteri välja 1988. aastal Berkeley ülikooli matemaatikaprofessor John Osterhout oma programmeerimiskeele Tcl jaoks. Hiljem on see aga kohandatud paljudele teistele YAP-dele, nagu Ruby Pärl jne. Alates 1994. aastast kuulub see tarkvarapakett Pythoni programmeerimiskeele standarddistributsiooni. (media, 2023)

**Tkinter.ttk -** Tkinteri paketis on ttk moodul, mis sisaldab stiliseeritud ja kaasaegsete vidinate klasse. Vaikimisi sõltub nende välimus operatsioonisüsteemist. ttk-s kasutatakse vidinate loomiseks uut mootorit. See mootor toetab teemasid ja kujundusstiile. Alates python 2.7 ja 3.1.2 versioonidest on Tkinterisse lisatud moodul ttk-ga töötamiseks. (wiki, kuupäev puudub) (Шапошникова, 2024)

**SQLite3 -** on sisseehitatud andmebaasihaldussüsteem klient-serverrakenduste jaoks. See SUBD erineb kompaktsusest - see võtab ühe faili, mis võimaldab rakendusel täita SUBD-i funktsioone autonoomselt, ilma serverit kasutamata. Samuti rakendatakse SQLite väikeste projektide veebiarenduses. SQLite on üks populaarsemaid avatud lähtekoodiga ja suure kommuuniga SQLID. (Workspace, kuupäev puudub)

# PRAKTILINE OSA

## Ettevalmistused tööks

Pythoni ja Tkinteri abil raamatukogu haldamise süsteemi loomiseks arendan rakenduse, mis võimaldab kasutajatel lisada raamatuid raamatukokku, vaadata nende nimekirja ja kustutada raamatuid nimekirjast. Selles protsessis kasutan andmebaasi, et salvestada teavet raamatute kohta, sealhulgas pealkiri, väljaandmise kuupäev, autor ja žanr.

Kõigepealt tuleb tuleb kasutada vajalikud komponendid:

* Python
* Tkinter (Graafilise liidese jaoks)
* SQLite (Andmebaasiga töötamiseks)

## Graafilise liidese loomine

Rakendus algab vajalike teekide (Tkinter,sqlite3) importimisest ja erinevate funktsioonide määratlemisest andmebaasiga suhtlemiseks ja liidese uuendamiseks.

Koodide näited ja mida nad teevad::

***Funktsioon andmete värskendamiseks tabelites***. Näiteks update\_autor\_nimi2 () uuendab autori nime andmebaasis, kasutades kasutaja sisestatud väärtusi. Funktsioonid update\_zanr\_nimi2 () ja update\_raamat\_pealkiri2 () teevad sarnaseid toiminguid, et uuendada vastavalt žanreid ja raamatute nimetusi***.***

def update\_autor\_nimi(connection, user\_data):

query = "UPDATE Autorid SET autor\_nimi = ? WHERE autor\_nimi = ?"

cursor = connection.cursor()

cursor.execute(query, user\_data)

connection.commit()

messagebox.showinfo("Edu", "Autori nimi uuendatud edukalt")

def update\_autor\_nimi2():

autor\_nimi = valitud\_autor\_nimi.get()

uus\_nimi = uus\_nimi\_entry.get()

update\_autor\_nimi(connection, (uus\_nimi, autor\_nimi))

uus\_nimi\_entry.delete(0, END)

update\_frame = tk.Frame(root)

update\_frame.pack(pady=10)

autor\_nimi = [row[0] for row in execute\_read\_query(connection, "SELECT autor\_nimi FROM Autorid")]

näidata\_andme\_frame = tk.Frame(root)

näidata\_andme\_frame.pack(pady=10)

autor\_nimi\_label = tk.Label(näidata\_andme\_frame, text="Valige Autori nimi:")

autor\_nimi\_label.pack(side=tk.LEFT, padx=5)

valitud\_autor\_nimi = tk.StringVar()

autor\_nimi\_combobox = ttk.Combobox(näidata\_andme\_frame, textvariable=valitud\_autor\_nimi, values=autor\_nimi)

autor\_nimi\_combobox.pack(side=tk.LEFT, padx=5)

uus\_nimi\_label = tk.Label(näidata\_andme\_frame, text="Uus nimi:")

uus\_nimi\_label.pack(side=tk.LEFT, padx=5)

uus\_nimi\_entry = tk.Entry(näidata\_andme\_frame)

uus\_nimi\_entry.pack(side=tk.LEFT, padx=5)

update\_autor\_button = tk.Button(näidata\_andme\_frame, text="Uuenda autorinime", command=update\_autor\_nimi2)

update\_autor\_button.pack(side=tk.LEFT, padx=5)

null

Pilt 1. Andmete uuendamine

***Funktsioon uute kirjete lisamiseks tabelitesse***. Funktsioon add\_autor () loob uue akna, kus kasutaja saab sisestada autori nime ja sünniaja ning seejärel lisab selle teabe andmebaasi. Samamoodi võimaldavad funktsioonid add\_zanr () ja add\_raamat () lisada vastavalt uusi žanre ja raamatuid. Need funktsioonid loovad andmete sisestamiseks ajutised aknad ja kasutavad SQL-i päringuid uute kirjete lisamiseks vastavatesse tabelitesse.

def add\_autor\_query(connection, user\_data):

query = "INSERT INTO Autorid(autor\_nimi, sünnikuupäev) VALUES (?,?)"

cursor = connection.cursor()

cursor.execute(query, user\_data)

connection.commit()

messagebox.showinfo("Edu","Autor lisas edukalt")

def add\_autor():

autor\_andmed\_entry = tk.Entry(root)

autor\_andmed\_entry.focus\_set()

autor\_andmed\_frame = tk.Toplevel(root)

autor\_andmed\_frame.title("Lisa autor")

tk.Label(autor\_andmed\_frame, text="Autori nimi:").grid(row=1, column=0)

autor\_nimi\_entry = tk.Entry(autor\_andmed\_frame)

autor\_nimi\_entry.grid(row=1, column=1)

tk.Label(autor\_andmed\_frame, text="Sünnipäev:").grid(row=2, column=0)

autor\_sünnipäev\_entry = tk.Entry(autor\_andmed\_frame)

autor\_sünnipäev\_entry.grid(row=2, column=1)

def add\_autor\_andmebaasi():

autor\_andmed = ( autor\_nimi\_entry.get(), autor\_sünnipäev\_entry.get())

add\_autor\_query(connection, autor\_andmed)

autor\_andmed\_frame.destroy()

add\_btn = tk.Button(autor\_andmed\_frame, text="Lisa autor", command=add\_autor\_andmebaasi)

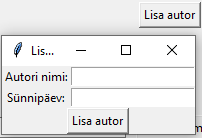
add\_btn.grid(row=3, column=0, columnspan=2)

add\_frame = tk.Frame(root)

add\_frame.pack(pady=10)

add\_autor\_btn = tk.Button(add\_frame, text="Lisa autor", command=add\_autor)

add\_autor\_btn.pack(side=tk.LEFT, padx=5)



Pilt 2. Andmete lisamine

***Kirjete kustutamise funktsiooni*** hulka kuuluvad delete\_raamat\_pealkirja\_jjergi (), delete\_Iltanri\_nime\_jjergi () ja delete\_autor\_nime\_jjergi (), mis võimaldavad kasutajatel kustutada raamatuid, žanre ja autoreid nende nimede järgi. Need funktsioonid loovad ka ajutisi aknaid andmete sisestamiseks ja täidavad vastavaid SQL-i päringuid kirjete kustutamiseks***.***

def delete\_raamat\_pealkiri(connection, pealkiri):

query = "DELETE FROM Raamatud WHERE pealkiri=?"

cursor = connection.cursor()

cursor.execute(query, (pealkiri,))

connection.commit()

messagebox.showinfo("Edu", "Raamat edukalt kustutatud")

def delete\_raamat\_pealkirja\_järgi():

uus\_window = tk.Toplevel(root)

uus\_window.title("Kustuta Raamat pealkirja järgi")

juhendamise = tk.Label(uus\_window, text="Sisestage kustutamiseks raamatu pealkiri:")

juhendamise.pack(pady=10)

global pealkiri\_entry

pealkiri\_entry = tk.Entry(uus\_window)

pealkiri\_entry.pack(pady=10)

def kinnitada\_kustutamist():

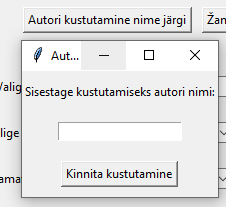
pealkiri = pealkiri\_entry.get()

delete\_raamat\_pealkiri(connection, pealkiri)

uus\_window.destroy()

kinnitusnupp = tk.Button(uus\_window, text="Kinnita kustutamine", command=kinnitada\_kustutamist)

kinnitusnupp.pack(pady=10)



Pilt 3. Kirjete kustutamine

***Tabelite andmete kuvamise funktsioon*** on tabel\_autorid (), tabel\_zanr () ja tabel\_raamatud (). Need funktsioonid loovad uusi Treview tabelitega aknaid, mis näitavad kõiki kirjeid andmebaasi vastavatest tabelitest. Näiteks tabel\_autorid () näitab autorite tabelit nende identifikaatorite, nimede ja sünnikuupäevadega.

def table\_autorid(conn):

window\_autorid = tk.Toplevel()

window\_autorid.title("Autorite tabel")

tree = ttk.Treeview(window\_autorid, column=("autor\_id", "autor\_nimi", "sünnikuupäev"), show="headings")

tree.column("autor\_id", anchor=CENTER)

tree.heading("autor\_id", text="autor\_id")

tree.column("autor\_nimi", anchor=CENTER)

tree.heading("autor\_nimi", text="autor\_nimi")

tree.column("sünnikuupäev", anchor=CENTER)

tree.heading("sünnikuupäev", text="sünnikuupäev")

try:

read = execute\_read\_query(conn, "SELECT \* FROM Autorid")

for row in read:

tree.insert("", END, values=row)

except Exception as e:

print(f"Viga tabelis autorid: {e}")

tree.pack()

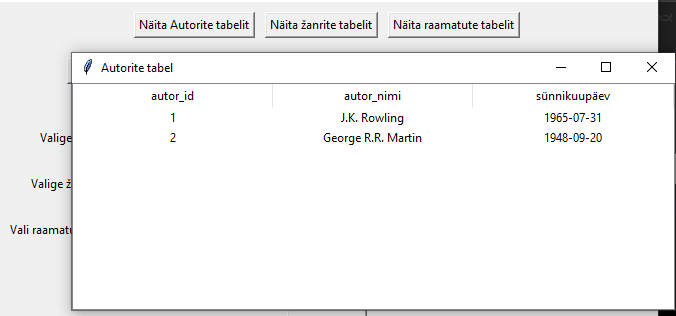
window\_autorid.mainloop()

näidata\_andme\_frame = tk.Frame(root)

näidata\_andme\_frame.pack(pady=10)

näidata\_autori\_btn = tk.Button(näidata\_andme\_frame, text="Näita Autorite tabelit", command=näita\_autorid\_tabelit)

näidata\_autori\_btn.pack(side=tk.LEFT, padx=5)



Pilt 4. Andmete kuvamine

***Tabelitesse algandmete lisamiseks kasutatakse funktsiooni*** insert\_initial\_data (). See funktsioon lisab etteantud kirjed valitud tabelisse, mis on kasulik andmebaasi algseadistamiseks.

def insert\_initial\_data():

valitud\_tabel = algsed\_andmed\_combobox.get()

alg\_andmed\_päringud = {

"Autorid": [

"INSERT INTO Autorid(autor\_nimi, sünnikuupäev) VALUES ('J.K. Rowling', '1965-07-31')",

"INSERT INTO Autorid VALUES ('George R.R. Martin', '1948-09-20')"

],

"Žanrid": [

"INSERT INTO Žanrid VALUES ('Fantaasia')",

"INSERT INTO Žanrid VALUES ('Sci-Fi')"

],

"Raamatud": [

"INSERT INTO Raamatud VALUES ('Harry Potter ja filosoofi kivi', '1997-06-26', 1, 1)",

"INSERT INTO Raamatud VALUES ('Troonide mäng', '1996-08-01', 2, 2)"

]

}

if valitud\_tabel in alg\_andmed\_päringud:

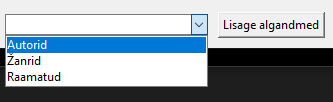
for query in alg\_andmed\_päringud[valitud\_tabel]:

execute\_query(connection, query)

messagebox.showinfo("Edu", f"Esialgsed andmed sisestati {valitud\_tabel} tabelisse edukalt")

else:

messagebox.showerror("Viga", "Kehtetu tabeli nimi")



Pilt 5. Algsete andmete lisamine

Rakendus sisaldab ka ***funktsioone andmebaasi tabelite loomiseks ja kustutamiseks***. Funktsioon create\_valitud\_tabel () loob valitud tabeli, kui seda veel ei ole, ja delete\_table\_clicked () kustutab määratud tabeli.

def create\_valitud\_tabel():

cursor = connection.cursor()

valitud\_tabel = table\_combobox.get()

if valitud\_tabel == "Autorid":

cursor.execute("""

CREATE TABLE IF NOT EXISTS Autorid (

autor\_id INTEGER PRIMARY KEY AUTOINCREMENT,

autor\_nimi TEXT NOT NULL,

sünnikuupäev DATE NOT NULL

)

""")

elif valitud\_tabel == "Žanrid":

cursor.execute("""

CREATE TABLE IF NOT EXISTS Žanrid (

žanr\_id INTEGER PRIMARY KEY AUTOINCREMENT,

žanri\_nimi TEXT NOT NULL

)

""")

elif valitud\_tabel == "Raamatud":

cursor.execute("""

CREATE TABLE IF NOT EXISTS Raamatud (

raamat\_id INTEGER PRIMARY KEY AUTOINCREMENT,

pealkiri TEXT NOT NULL,

väljaandmise\_kuupäev DATE NOT NULL,

autor\_id INTEGER,

žanr\_id INTEGER,

FOREIGN KEY(autor\_id) REFERENCES Autorid(autor\_id),

FOREIGN KEY(žanr\_id) REFERENCES Žanrid(žanr\_id)

)

""")

else:

print("Vigane valik.")

connection.commit()

messagebox.showinfo("Edu", f"Tabel '{valitud\_tabel}' on edukalt loodud")

create\_add\_frame = tk.Frame(root)

create\_add\_frame.pack(pady=10)

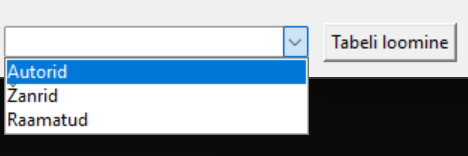
table\_combobox = ttk.Combobox(root, width=30, height=10)

table\_combobox['values'] = ("Autorid", "Žanrid", "Raamatud")

table\_combobox.pack(side=tk.LEFT, padx=(10, 5))

create\_button = tk.Button(root, text="Tabeli loomine", command=create\_valitud\_tabel)

create\_button.pack(side=tk.LEFT, pady=10, padx=(5, 10))



Pilt 6. Tabeli loomine

def delete\_table\_clicked():

tabel\_nimi = tabel\_nimi\_entry.get()

query = f"DROP TABLE IF EXISTS {tabel\_nimi}"

delete\_tabel(connection, query)

window.destroy()

def kustuta\_tabeli\_liides():

global window

window = tk.Toplevel(root)

window.title("Kustuta Tabel")

tabel\_nimi\_label = tk.Label(window, text="Tabeli nimi:")

tabel\_nimi\_label.pack(pady=10)

global tabel\_nimi\_entry

tabel\_nimi\_entry = tk.Entry(window)

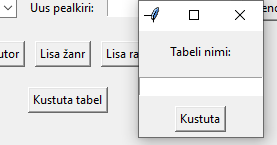
tabel\_nimi\_entry.pack(pady=5)

delete\_button = tk.Button(window, text="Kustuta", command=delete\_table\_clicked)

delete\_button.pack(pady=5)

delete\_table\_btn = tk.Button(root, text="Kustuta tabel", command=kustuta\_tabeli\_liides)

delete\_table\_btn.pack(pady=10)



Pilt 7. Tabeli kustutamine

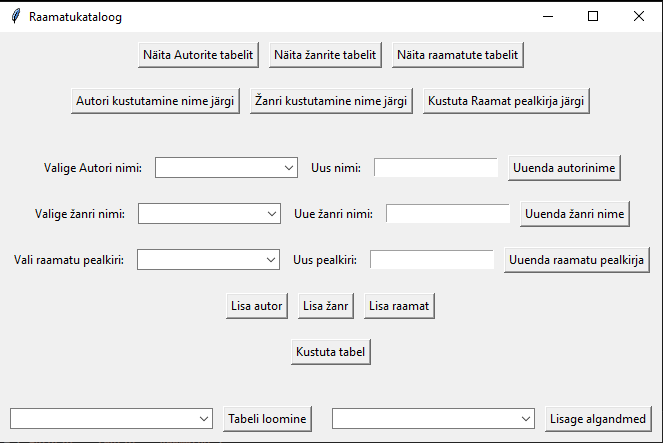
***Programmi kasutajaliides*** koosneb mitmest nupust ja ripploendist, mis asuvad erinevates raamides. Näiteks autorite, žanrite ja raamatute tabelite kuvamise nupud on ühes fraasis, uute kirjete lisamise nupud teises ja kirjete kustutamise nupud kolmandas. Kasutaja saab valida soovitud toimingu ja seda teha, vajutades vastavaid nuppe ja sisestades ilmuvates akendes vajalikud andmed.

# KASUTUSJUHEND

Rakendus koosneb mitmest põhifunktsioonist, mis võimaldavad andmebaasiga suhelda ja raamatukataloogi hallata.

**Pealiides**

Pärast rakenduse käivitamist avaneb nuppudega peaaken erinevate toimingute tegemiseks:



Pilt 8. Programmi peaaken

**Andmete vaatamine:**

Vaata autoreid: Klõpsake nuppu "Näita Autorite tabelit , et kuvada kõigi andmebaasi autorite nimekiri.

Vaata žanrid: Klõpsake nuppu "Näita žanrite tabelit , et kuvada andmebaasi kõigi žanrite nimekiri.

Vaata raamatuid: Klõpsake nuppu «Näita raamatute tabelit , et kuvada kõigi andmebaasi raamatute nimekiri.

**Uute kirjete lisamine:**

Lisa autor: Klõpsake nuppu "Lisa autor" ja sisesta ilmuvas aknas autori nimi ja sünniaeg.

Lisa žanr: Klõpsake nuppu "Lisa žanr ja sisesta ilmuvas aknas žanri nimi.

Lisa raamat: Klõpsake nuppu "Lisa raamat" ja kirjuta ilmuvas aknas raamatu pealkiri, autor, žanr ja ilmumise kuupäev.

**Andmete uuendamine:**

Uuenda autori nime: Vali rippmenüüst vana nimi ja sisesta tühjale väljale uus autori nimi, klõpsa nupule "Uuenda autorinime"

Uuenda žanrinimetust: Vali rippmenüüst vana nimi ja sisesta tühjale väljale uus nimi, klõpsa nupule "Uuenda žanri nime"

Uuenda raamatu nime: ja Vali rippmenüüst vana raamatu nimi ja sisesta uus nimi tühjale väljale, vajuta nupule "Uuenda raamatu pealkirja"

**Kirjete kustutamine:**

Kustuta raamat nime järgi: Klõpsake nuppu "Kustuta Raamat pealkirja järgi" ja kirjuta raamatu pealkiri, mis tuleb kustutada.

Žanri kustutamine nime järgi: Klõpsa nupule "Žanri kustutamine nime järgi" ja kirjuta žanri nimi, mis tuleb kustutada.

Kustuta autor nime järgi: Vajuta nupule "Autori kustutamine nime järgi" ja sisesta autori nimi, kelle pead kustutama.

Näited kasutamise kohta:

**Uue autori lisamine:**

Vajuta "Lisa autor" nuppu.

Kirjuta ilmuvas aknas autori nimi ja sünniaeg.

Autori lisamiseks andmebaasi klõpsa "Lisa autor".

**Vaata raamatute nimekirja:**

Vajutage nuppu "Näita raamatute tabelitl".

Uues aknas kuvatakse tabel kõigi andmebaasis olevate raamatutega.

**Raamatu pealkirja uuendamine:**

Kirjuta ilmuvasse aknasse raamatu vana pealkiri ja raamatu uus pealkiri.

Muudatuste salvestamiseks klõpsa nupule "Uuenda autorinime".

**Žanri eemaldamine:**

Vajuta nupule "Žanri kustutamine nime järgi".

Sisesta ilmuvas aknas selle žanri nimi, mis tuleb kustutada.

Klõpsake "Kinnita kustutamine", et eemaldada žanr andmebaasist.

# KOKKUVÕTTE

Selle interaktiivse raamatukataloogi haldamise rakenduse väljatöötamise tulemusena olen loonud mugava tööriista, mis hõlbustab raamatukogunduse või isikliku raamatukogu korraldamise protsessi. Rakendus pakub laialdasi võimalusi töötada raamatute, autorite ja žanrite andmetega, võimaldades kasutajatel teavet lisada, uuendada, kustutada ja vaadata.

Üks rakenduse omadusi on selle lihtne kasutamine tänu intuitiivsele liidesele, mis on jagatud mitmeks funktsionaalseks plokiks. Kasutajad saavad hõlpsasti õppida peamisi tegevusi, näiteks kirjete vaatamist, lisamist, värskendamist ja kustutamist.

Rakenduse tehniline rakendamine põhineb Pythoni programmeerimiskeelel, kasutades Tkinteri teeki, et luua andmebaasi haldamiseks graafiline liides ja SQLite3 teek. See tagab rakenduse kõrge jõudluse ja töökindluse.

Üldiselt on rakendus tõhus vahend raamatukataloogi korraldamiseks ja haldamiseks, mis võib olla kasulik nii raamatukogunduse professionaalidele kui ka lugemishuvilistele nende enda kogude haldamiseks.

[Minu projekt GitHubis](https://github.com/JOstapjuk/Andmebaasi_haldamine_raamatukataloogis)

# ALLIKAD

Amazon. (kuupäev puudub). *Что такое Python?* Allikas: amazon.com: https://aws.amazon.com/ru/what-is/python/

media, S. (04. 09 2023. a.). *Tkinter*. Allikas: blog.skillfactory.ru: https://blog.skillfactory.ru/glossary/tkinter/

wiki, P. (kuupäev puudub). *Tkinter ttk (модуль)*. Allikas: ts-python.fandom.com: https://ts-python.fandom.com/ru/wiki/Tkinter\_ttk\_(%D0%BC%D0%BE%D0%B4%D1%83%D0%BB%D1%8C)

Workspace. (kuupäev puudub). *SQLite*. Allikas: workspace.ru: https://workspace.ru/tools/database/sqllite/

Шапошникова, С. (2024). *Модуль tkinter.ttk*. Allikas: younglinux.info: https://younglinux.info/tkinter/ttk