TALLINNA TÖÖSTUSHARIDUSKESKUS

**ANDMEBAASI HALDAMINE RAAMATUKATALOOGIS**  
Arvestus töö

Õpetaja: Marina Oleinik  
Õpilane: Jelizaveta Ostapjuk  
Rühm: TARpv23

Tallinn  
2024

# SISUKORD

[SISUKORD 2](#_gjdgxs)

[SISEJUHATUS 3](#_30j0zll)

[TEOREETILINE OSA 4](#_1fob9te)

[1. Vahendid 4](#_3znysh7)

[PRAKTILINE OSA 6](#_2et92p0)

[2. Подготовка к работе 6](#_tyjcwt)

[3. Создание базы данных 6](#_3dy6vkm)

[4. Создание графического интерфейса 7](#_1t3h5sf)

[KASUTUSJUHEND 10](#_4d34og8)

[KOKKUVÕTTE 11](#_2s8eyo1)

[ALLIKAD 12](#_17dp8vu)

# **SISEJUHATUS**

Цель проекта заключается в создании интерактивного приложения для управления каталогом книг. Это приложение позволяет пользователям просматривать информацию о книгах, авторах и жанрах, а также добавлять, обновлять и удалять данные в базе данных. Этим приложением может воспользоваться огромный круг людей, начиная от работников библиотеки для ведения учета книг до любителей почитать для введения личной библиотеки. Актуальность приложения управлением каталога книг заключается в ее эффективности и простоте управления, они нужны различным сферам деятельности, к примеру для академических целей или же для магазинов и библиотек.

Для достижения этой цели я использовала приложение Visual Studio 2022 и писала на языке Python. Так же использовала различные вспомогательные библиотеки такие как tkinter и sqlite3. Tkinter использовался для внешней оболочки, а SQLite3 для управления базой данных.

# TEOREETILINE OSA

## Vahendid

* **Python** - это язык программирования, который широко используется в интернет-приложениях, разработке программного обеспечения, науке о данных и машинном обучении (ML). Разработчики используют Python, потому что он эффективен, прост в изучении и работает на разных платформах. Программы на языке Python можно скачать бесплатно, они совместимы со всеми типами систем и повышают скорость разработки. (Что такое Python?, 2023)
* **Tkinter** — это кроссплатформенный графический интерфейс Python, позволяющий работать с библиотекой Tk. Он содержит элементы графического интерфейса пользователя (GUI — Graphical User Interface), с помощью которых можно создавать различные приложения. Изначально Tkinter был разработан в 1988 году профессором математических наук в университете Беркли Джоном Остерхаутом для собственного языка программирования Tcl. Однако впоследствии он был адаптирован для многих других ЯП, таких как Ruby Perl и т.д. С 1994 года этот программный пакет входит в стандартный дистрибутив языка программирования Python. (Tkinter, 2023)
* **Tkinter.ttk -** В состав пакета tkinter входит модуль ttk, содержащий классы более стилизованных и современных виджет. По умолчанию их внешний вид зависит от операционной системы. В ttk используется новый движок для создания виджетов. Этот движок обладает поддержкой тем и стилей оформления. Начиная с версий python 2.7 и 3.1.2 в Tkinter включён модуль для работы с ttk. (Tkinter ttk (модуль), б.д.) (Шапошникова, 2024)
* **SQLite3 -** это встраиваемая система управления базами данных для клиент-серверных приложений. Эта СУБД отличается компактностью – она занимает один файл, благодаря чему встраивается в приложения, чтобы выполнять функции СУБД автономно, без использования сервера. Также SQLite применяется в веб-разработке небольших проектов. SQLite – одна из самых популярных СУБД с открытым исходным кодом и большим комьюнити. (SQLite, б.д.)

# PRAKTILINE OSA

## Подготовка к работе

Для создания системы управления библиотекой книг с использованием Python и Tkinter, я разработаю приложение, которое позволит пользователям добавлять книги в библиотеку, просматривать их список и удалять книги из списка. В этом процессе я буду использовать базу данных для хранения информации о книгах, включая название, дату выпуска, автора и жанр.

Для начала необходимо установить необходимые компоненты:

* Python
* Tkinter (Для графического интерфейса)
* SQLite (Для работы с базой данных)

## Создание графического интерфейса

Приложение начинается с импорта необходимых библиотек(Tkinter,sqlite3) и определения различных функций для взаимодействия с базой данных и обновления интерфейса.  
Примеры кодов и что они делают:

***Функция для обновления данных в таблицах***. Например, update\_autor\_nimi2() обновляет имя автора в базе данных, используя введенные пользователем значения. Функции update\_zanr\_nimi2() и update\_raamat\_pealkiri2() выполняют аналогичные операции для обновления жанров и названий книг соответственно.

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

***Функция для добавления новых записей в таблицы***. Функция add\_autor() создает новое окно, в котором пользователь может ввести имя автора и дату рождения, а затем добавляет эту информацию в базу данных. Аналогично, функции add\_zanr() и add\_raamat() позволяют добавлять новые жанры и книги соответственно. Эти функции создают временные окна для ввода данных и используют запросы SQL для вставки новых записей в соответствующие таблицы.

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)

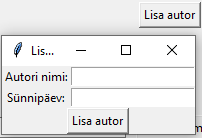
add\_btn.grid(row=3, column=0, columnspan=2)

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)



***Функция для удаления записей*** включают delete\_raamat\_pealkirja\_järgi(), delete\_žanri\_nime\_järgi() и delete\_autor\_nime\_järgi(), которые позволяют пользователям удалять книги, жанры и авторов по их именам. Эти функции также создают временные окна для ввода данных и выполняют соответствующие запросы SQL для удаления записей.

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():

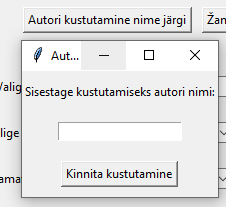
pealkiri = pealkiri\_entry.get()

delete\_raamat\_pealkiri(connection, pealkiri)

uus\_window.destroy()

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

kinnitusnupp.pack(pady=10)



***Функция для отображения данных из таблиц*** включают table\_autorid(), table\_zanr() и table\_raamatud(). Эти функции создают новые окна с таблицами Treeview, которые отображают все записи из соответствующих таблиц базы данных. Например, table\_autorid() отображает таблицу авторов с их идентификаторами, именами и датами рождения.

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()

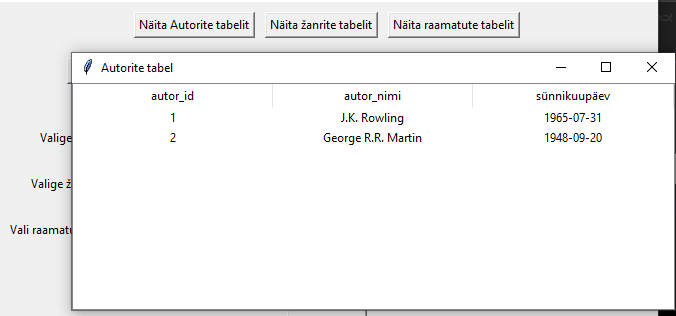
window\_autorid.mainloop()

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)



***Для добавления начальных данных в таблицы*** используется функция insert\_initial\_data(). Эта функция вставляет предопределенные записи в выбранную таблицу, что полезно для начальной настройки базы данных.

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:

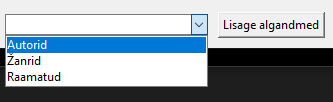
for query in alg\_andmed\_päringud[valitud\_tabel]:

execute\_query(connection, query)

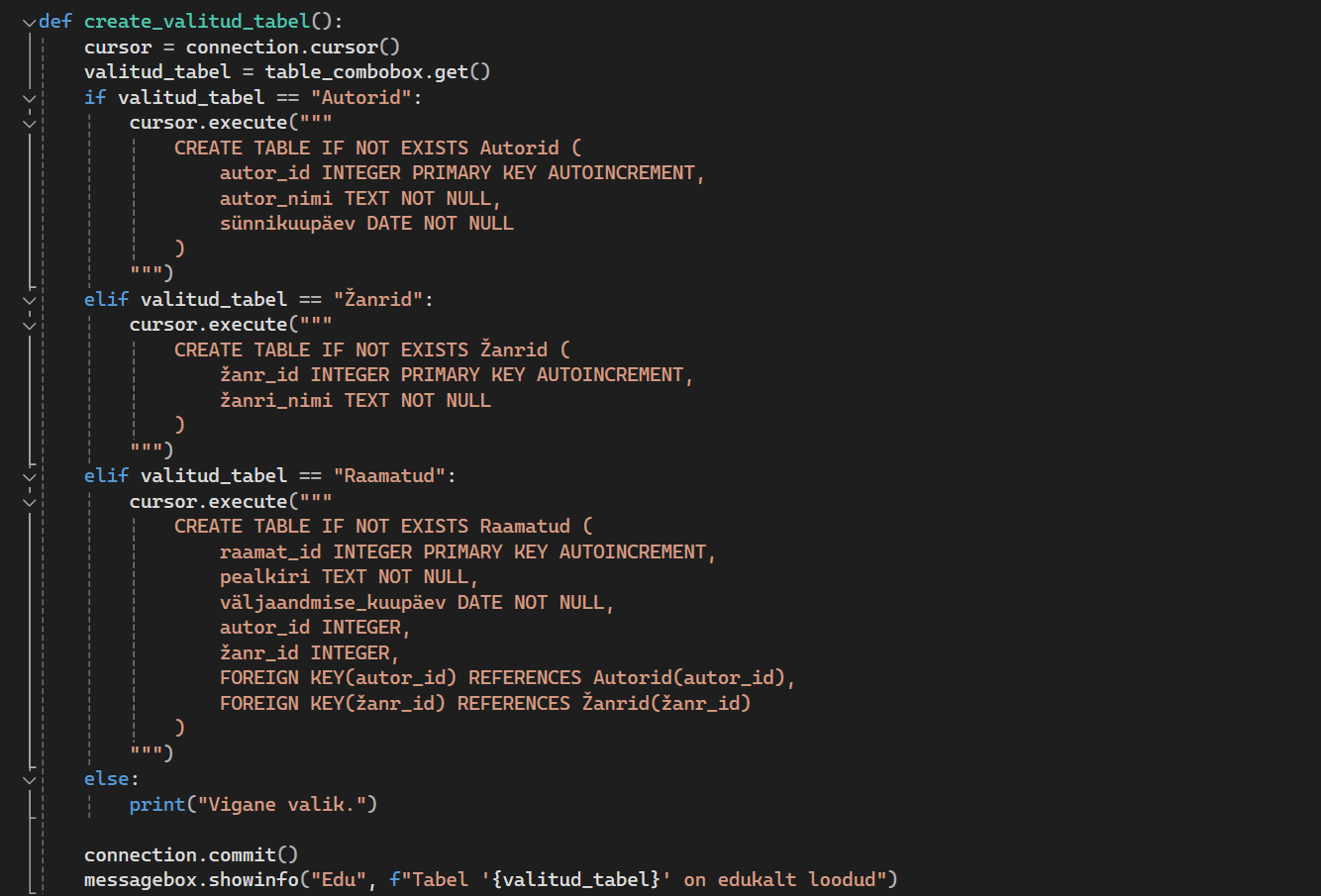
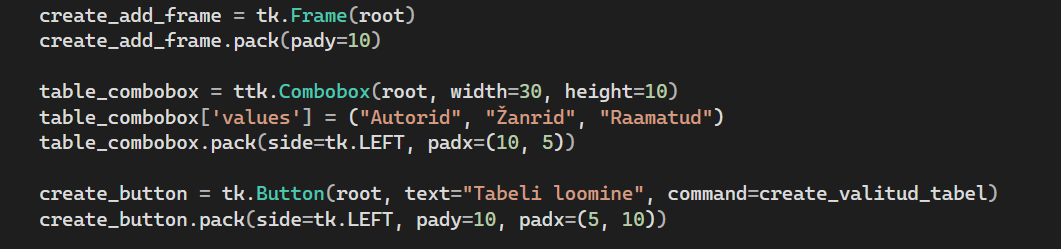
messagebox.showinfo("Edu", f"Esialgsed andmed sisestati {valitud\_tabel} tabelisse edukalt")

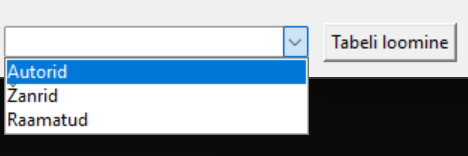
else:

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



***Приложение также включает функции для создания и удаления таблиц*** в базе данных. Функция create\_valitud\_tabel() создает выбранную таблицу, если она еще не существует, а delete\_table\_clicked() удаляет указанную таблицу.





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)

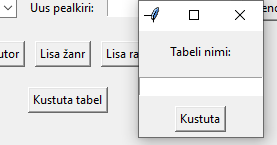
tabel\_nimi\_entry.pack(pady=5)

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)



***Интерфейс программы*** состоит из нескольких кнопок и выпадающих списков, расположенных в разных фреймах. Например, кнопки для отображения таблиц авторов, жанров и книг находятся в одном фрейме, кнопки для добавления новых записей — в другом, а кнопки для удаления записей — в третьем. Пользователь может выбрать нужное действие и выполнить его, нажимая соответствующие кнопки и вводя необходимые данные в появляющихся окнах.

# KASUTUSJUHEND

Приложение состоит из нескольких основных функций, которые позволяют взаимодействовать с базой данных и управлять каталогом книг.

**Главный интерфейс**

После запуска приложения откроется главное окно с кнопками для выполнения различных действий:

**Просмотр данных:**

Просмотреть авторов: Нажмите кнопку "Näita Autorite tabelit" для отображения списка всех авторов в базе данных.

Просмотреть жанры: Нажмите кнопку "Näita žanrite tabelit" для отображения списка всех жанров в базе данных.

Просмотреть книги: Нажмите кнопку "Näita raamatute tabelit" для отображения списка всех книг в базе данных.

**Добавление новых записей:**

Добавить автора: Нажмите кнопку "Lisa autor" и введите имя автора и дату рождения в появившемся окне.

Добавить жанр: Нажмите кнопку "Lisa žanr" и введите название жанра в появившемся окне.

Добавить книгу: Нажмите кнопку "Lisa raamat" и введите название книги, автора, жанр и дату выпуска в появившемся окне.

**Обновление данных:**

Обновить имя автора: Выберите старое имя в выпадающем списке и введите новое имя автора в пустом поле, нажмите кнопку "Uuenda autorinime"

Обновить название жанра: Выберите старое название жарна в выпадающем списке и введите новое название в пустом поле, нажмите кнопку "Uuenda žanri nime"

Обновить название книги: и Выберите старое название книги в выпадающем списке и введите новое название в пустом поле, нажмите кнопку "Uuenda raamatu pealkirja"

**Удаление записей:**

Удалить книгу по названию: Нажмите кнопку "Kustuta Raamat pealkirja järgi" и введите название книги, которую нужно удалить.

Удалить жанр по названию: Нажмите кнопку "Žanri kustutamine nime järgi" и введите название жанра, который нужно удалить.

Удалить автора по имени: Нажмите кнопку "Autori kustutamine nime järgi" и введите имя автора, которого нужно удалить.

Примеры использования:

**Добавление нового автора:**

Нажмите кнопку "Lisa autor".

В появившемся окне введите имя автора и дату рождения.

Нажмите "Lisa autor", чтобы добавить автора в базу данных.

**Просмотр списка книг:**

Нажмите кнопку "Näita raamatute tabelit".

В новом окне отобразится таблица со всеми книгами, содержащимися в базе данных.

**Обновление названия книги:**

Введите старое название книги и новое название книги в появившемся окне.

Нажмите кнопку "Uuenda autorinime" чтобы сохранить изменения.

**Удаление жанра:**

Нажмите кнопку "Žanri kustutamine nime järgi".

Введите название жанра, который нужно удалить, в появившемся окне.

Нажмите "Kinnita kustutamine", чтобы удалить жанр из базы данных.

# KOKKUVÕTTE

В результате разработки данного интерактивного приложения для управления каталогом книг я создала удобный инструмент, который облегчит процесс ведения библиотечного учета или организации личной библиотеки. Приложение предоставляет широкие возможности для работы с данными о книгах, авторах и жанрах, позволяя пользователям добавлять, обновлять, удалять и просматривать информацию.

Одной из особенностей приложения является его простота использования благодаря интуитивно понятному интерфейсу, который разделен на несколько функциональных блоков. Пользователи могут легко освоить основные действия, такие как просмотр, добавление, обновление и удаление записей.

Техническая реализация приложения основана на языке программирования Python с использованием библиотеки Tkinter для создания графического интерфейса и библиотеки SQLite3 для управления базой данных. Это обеспечивает высокую производительность и надежность работы приложения.

В целом, приложение представляет собой эффективный инструмент для организации и управления каталогом книг, который может быть полезен как профессионалам в области библиотечного дела, так и любителям чтения для управления их собственными коллекциями

# ALLIKAD

*SQLite*. (б.д.). Получено из workspace: https://workspace.ru/tools/database/sqllite/

*Tkinter*. (04 09 2023 г.). Получено из skillfactory: https://blog.skillfactory.ru/glossary/tkinter/

*Tkinter ttk (модуль)*. (б.д.). Получено из ts-python.fandom: https://ts-python.fandom.com/ru/wiki/Tkinter\_ttk\_(%D0%BC%D0%BE%D0%B4%D1%83%D0%BB%D1%8C)

*Что такое Python?* (2023). Получено из Amazon: https://aws.amazon.com/ru/what-is/python/

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