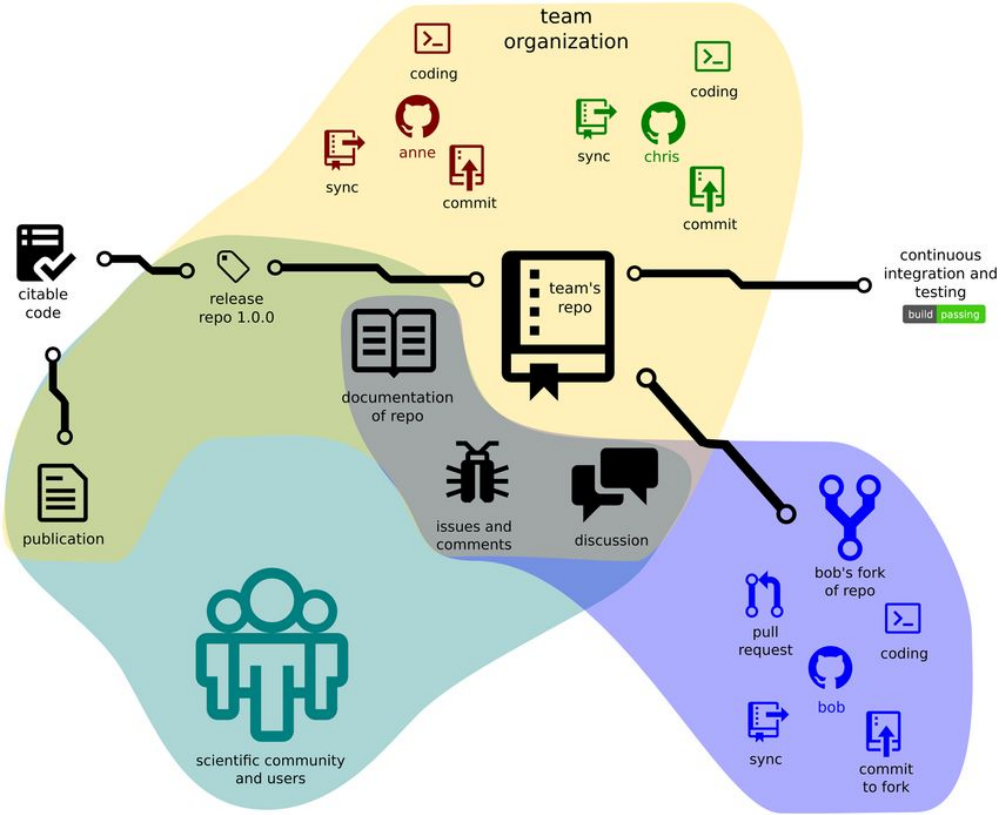
Git, Flask, GUI i baze podatka

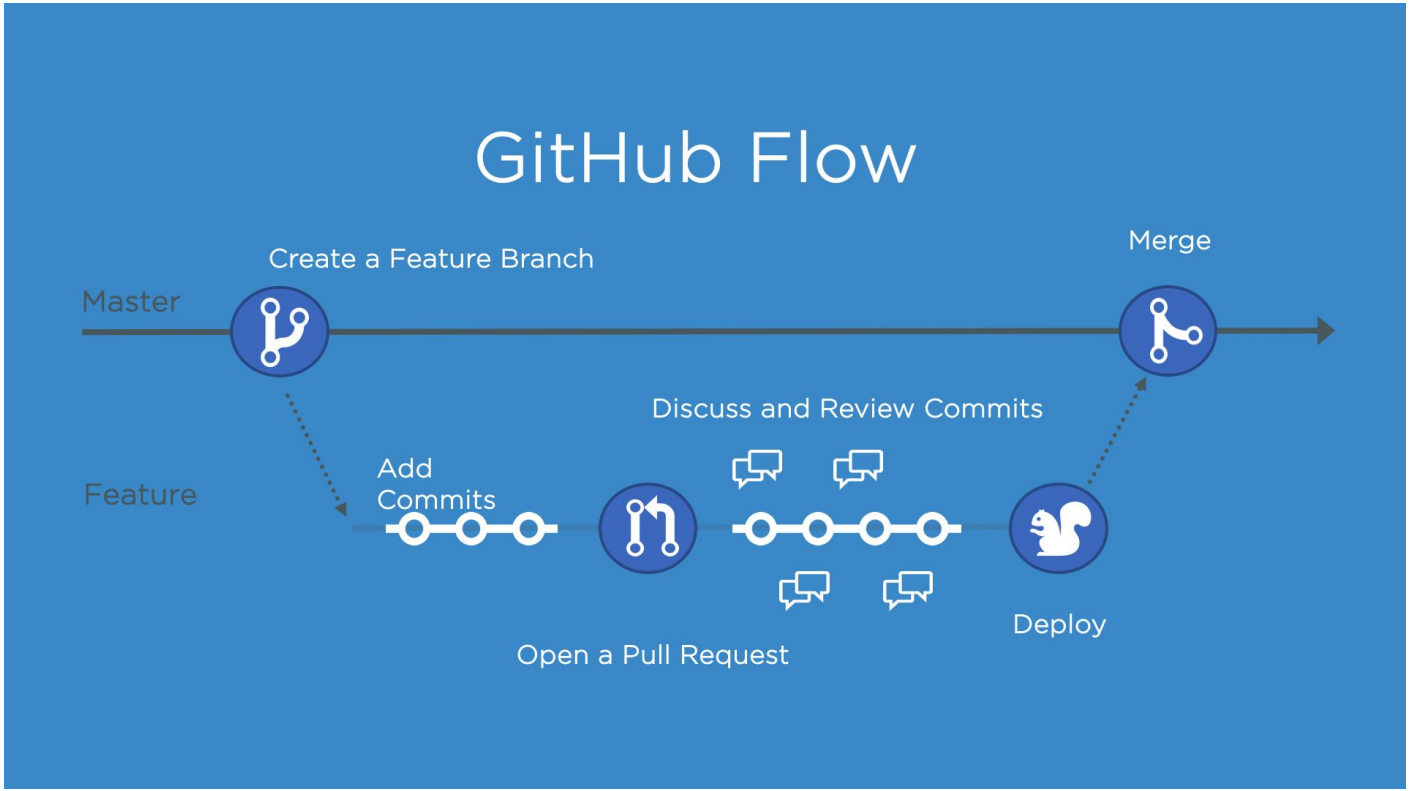
Pripremio: Stevan Čakić

Pregled

- Git struktura
- Flask
  - Instalacija i pokretanje servera
- Terminologija
  - Web server
  - HTTP verbs
- REST principi
- Tkinter i GUI



Izvor: Ten Simple Rules for Taking Advantage of Git and GitHub



Izvor: https://darkbabybeau.medium.com/github-flow-with-github-action-5d4f0bbd7eab

INSTALLATION & GUIS

With platform specific installers for Git, GitHub also provides the ease of staying up-to-date with the latest releases of the command line tool while providing a graphical user interface for day-to-day interaction, review, and repository synchronization.

GitHub for Windows
https://windows.github.com

GitHub for Mac
https://mac.github.com

For Linux and Solaris platforms, the latest release is available on the official Git web site.

Git for All Platforms
http://git-scm.com

SETUP

Configuring user information used across all local repositories

```
git config --global user.name "[firstname lastname]"
```
set a name that is identifiable for credit when review version history

```
git config --global user.email "[valid-email]"
```
set an email address that will be associated with each history marker

```
git config --global color.ui auto
```
set automatic command line coloring for Git for easy reviewing

SETUP & INIT

Configuring user information, initializing and cloning repositories

```
git init
```
initialize an existing directory as a Git repository

```
git clone [url]
```
retrieve an entire repository from a hosted location via URL

STAGE & SNAPSHOT

Working with snapshots and the Git staging area

```
git status
```
show modified files in working directory, staged for your next commit

```
git add [file]
```
add a file as it looks now to your next commit (stage)

```
git reset [file]
```
unstage a file while retaining the changes in working directory

```
git diff
```
diff of what is changed but not staged

```
git diff --staged
```
diff of what is staged but not yet committed

```
git commit -m "[descriptive message]"
```
commit your staged content as a new commit snapshot

BRANCH & MERGE

Isolating work in branches, changing context, and integrating changes

```
git branch
```
list your branches. a * will appear next to the currently active branch

```
git branch [branch-name]
```
create a new branch at the current commit

```
git checkout
```
switch to another branch and check it out into your working directory

```
git merge [branch]
```
merge the specified branch's history into the current one

```
git log
```
show all commits in the current branch's history

INSPECT & COMPARE

Examining logs, diffs and object information

```
git log
```
show the commit history for the currently active branch

```
git log branchB..branchA
```
show the commits on branchA that are not on branchB

```
git log --follow [file]
```
show the commits that changed file, even across renames

```
git diff branchB...branchA
```
show the diff of what is in branchA that is not in branchB

```
git show [SHA]
```
show any object in Git in human-readable format

TRACKING PATH CHANGES

Versioning file removes and path changes

```
git rm [file]
```
delete the file from project and stage the removal for commit

```
git mv [existing-path] [new-path]
```
change an existing file path and stage the move

```
git log --stat -M
```
show all commit logs with indication of any paths that moved

IGNORING PATTERNS

Preventing unintentional staging or commiting of files

```
logs/
*.notes
pattern*/
```
Save a file with desired patterns as .gitignore with either direct string matches or wildcard globs.

```
git config --global core.excludesfile [file]
```
system wide ignore pattern for all local repositories

SHARE & UPDATE

Retrieving updates from another repository and updating local repos

```
git remote add [alias] [url]
```
add a git URL as an alias

```
git fetch [alias]
```
fetch down all the branches from that Git remote

```
git merge [alias]/[branch]
```
merge a remote branch into your current branch to bring it up to date

```
git push [alias] [branch]
```
Transmit local branch commits to the remote repository branch

```
git pull
```
fetch and merge any commits from the tracking remote branch

REWRITE HISTORY

Rewriting branches, updating commits and clearing history

```
git rebase [branch]
```
apply any commits of current branch ahead of specified one

```
git reset --hard [commit]
```
clear staging area, rewrite working tree from specified commit

TEMPORARY COMMITS

Temporarily store modified, tracked files in order to change branches

```
git stash
```
Save modified and staged changes

```
git stash list
```
list stack-order of stashed file changes

```
git stash pop
```
write working from top of stash stack

```
git stash drop
```
discard the changes from top of stash stack

Izvor: https://education.github.com/git-cheat-sheet-education.pdf

Primjer, Github

- Prakticni primjer
  - Konfigurisati nalog
  - Kreirati repo (napomena za config i git bash za windows)
  - Dodati neki kod
  - Git add, commit, push

Flask, instalacija i pokretanje servera

- Flask je Python paket koji olakšava kreiranje APIa
- Komanda
  - pip install flask ili pip3 install flask
- Nakon toga moramo da importujemo flask paket, a iz njega Flask klasu
- Kreiramo instancu klase Flask
- Definišemo koje funkcije pozivamo na određene rute
- Obavezno pokrenemo server(aplikaciju) na određenom portu
- Pogledajmo primjer (**app_p1.py**)

Web server

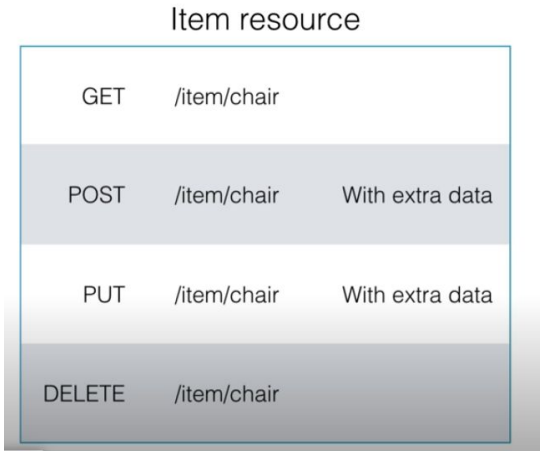
- Računar čiji softver je dizajniran tako da obrađuje requestove koje dobija
- Npr. Google ima veliki broj servera
  - Kada god ukucamo http://google.com u browseru, browser sa našeg računara šalje nešto jednom od Google web servera
  - Šta sve šaljemo web serverima (za naš primjer)?
    - GET Request
      - GET / HTTP 1.1
      - Host: google.com
  - Šta server vraća?
    - HTML kod
    - Neki tekst
    - Npr. grešku ako ne odbrađuje rutu /
    - Grešku ako HTTP 1.1 protokol na server nije podržan
    - Grešku ako je server trenutno nedostupan

HTTP keywords

Verb	Meaning	Example	
GET	Retrieve something	GET /item/1	
POST	Receive data, and use it	POST /item	`{ 'name': 'Chair', 'price': 9.99 }`
PUT	Make sure something is there	PUT /item	`{ 'name': 'Chair', 'price': 7.99 }`
DELETE	Remove something	DELETE /item/1	

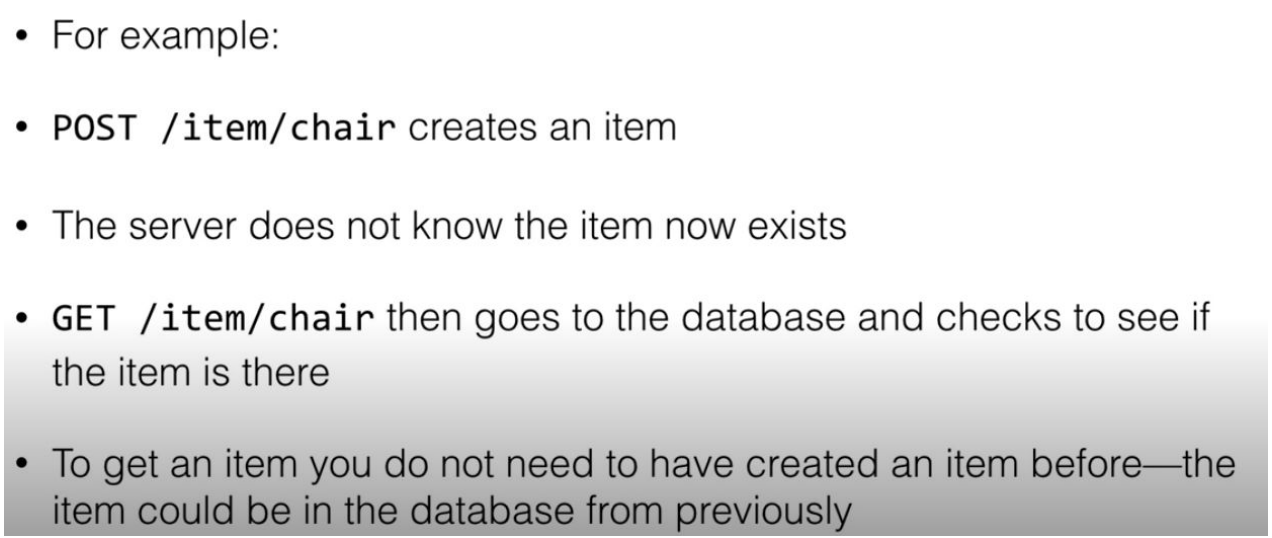
REST principi

- Način razmišljana kako vaš server odgovara (responds) na zahtjeve (requests).
- Ne odgovara tako što šalje samo podatke, već resurse.
- Šta su onda resursi?
  - Slično kao OOP
  - Posmatrajte server koji ima resurse i svaki je sposoban da vrši interakciju sa relevantnim zahtjevima



REST principi, nastavak

- Takođe, jedna od bitnih osobina RESTa je što bi trebao da je **stateless**
  - Jedan request ne bi trebao da zavisi od bili kog drugog requesta



REST API creation

- Kreiraćemo jednostavan API
- Potrebno je da instalirate Postman za testiranje APIa
  - U praksi rezultati koje generiše API vraćaju se strani koja je inicirala poziv (to je najčešće frontend)
- Primjer
- Pogledati dodatni primjer: **link**

Virtual environment

- Python čista instalacija (bez paketa)
- Svaki put kad smo radili **pip install ime_paketa** proširivali smo Python sa dodatnim paketima, a sve te pakete smo instalirali globalno
- Šta znači globalno ?
- Da provjerite koje pakete ste instalirali
  - **pip freeze**
- Vrlo bitno da u praksi vodite računa o verziji paketa, jer update paketa može da sruši rad vaše aplikacije (npr. nova verzija ima drugačije nazive metoda)
- Prvo što trebate da odradite za podešavanje virtuelnog okruženja
  - **pip install virtualenv**
  - **virtualenv naziv_enviromenta**
  - **Onda pokrenete activate skriptu (za pokretanje venv),**
    - **./naziv_environmeta/Scripts/activate.bat (source naziv_environmenta/bin/activate)**
  - **Za deaktiviranje, deactivate**

Virtual environment, nastavak

- Napomena: **pip install -r requirements.txt** za instalaciju paketa iz requirements.txt fajla
- Potrebno je da kreirate folder .vscode
- U njemu kreirajte fajl settings.json (ako je naziv za virtualenv **env**)

```
{
    "python.pythonPath": "project_path\\env\\Scripts\\python.exe"
}
```

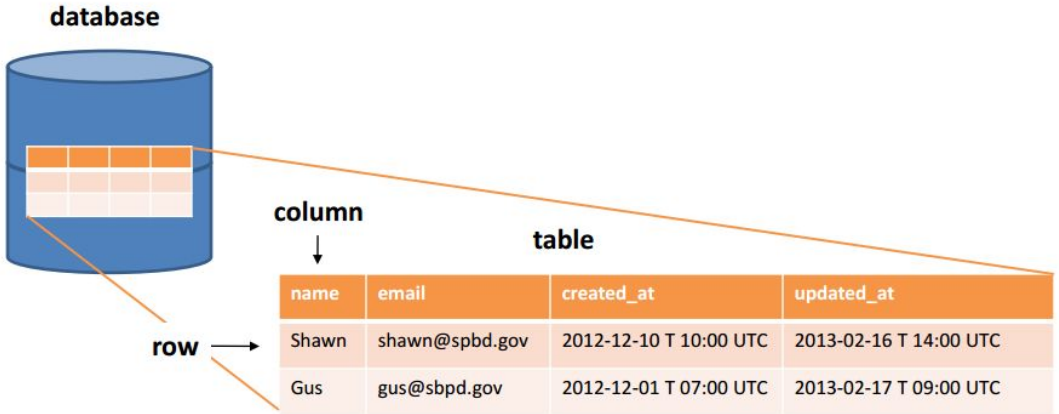
- Nakon što aktivirate određeno virtuelno okruženje, onda krećemo sa instalacijom određenih modula koji nam trebaju
- Posle toga
  - pip freeze > requirements.txt

Flask RESTful

- Ovaj paket služi za jednostavnije kreiranje RESTful APIa
- Instalirati paket Flask-RESTful
- Primjer

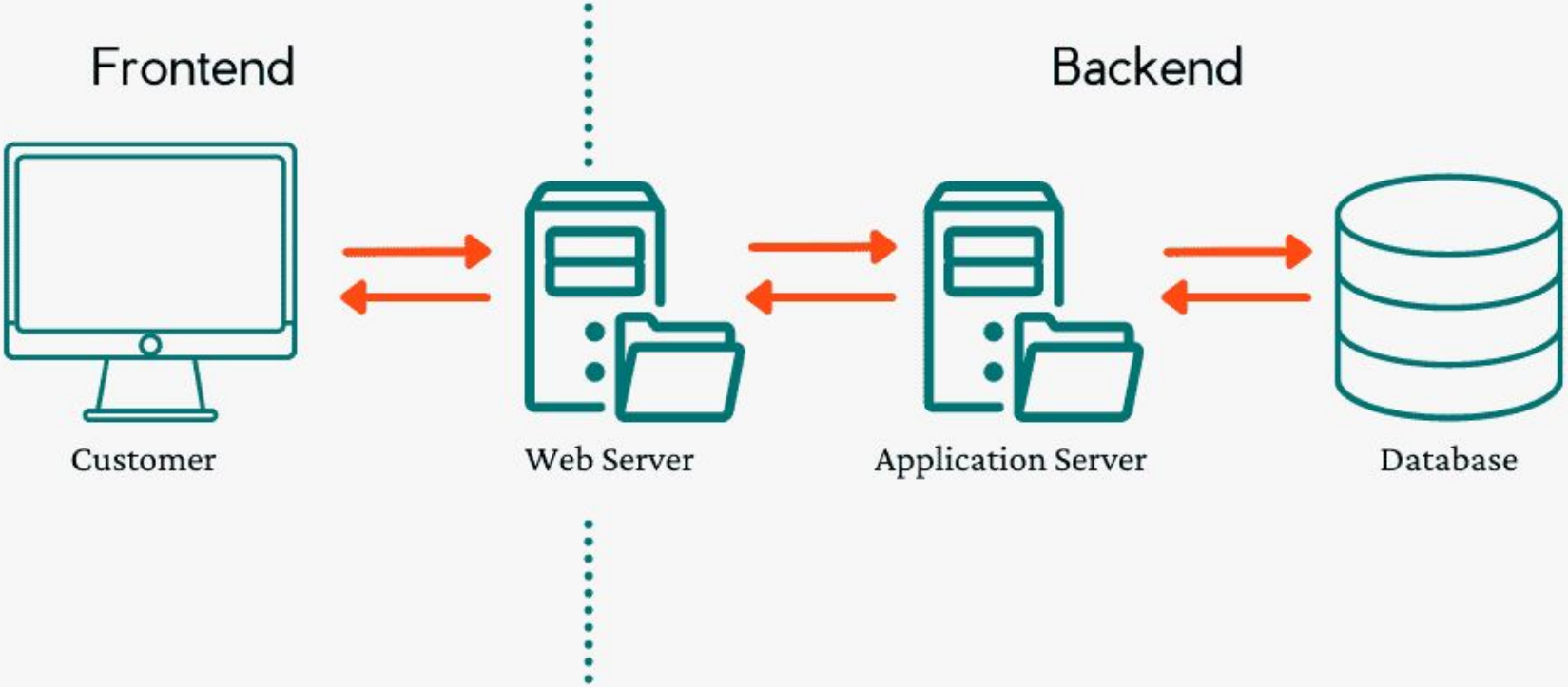
Relacione baze

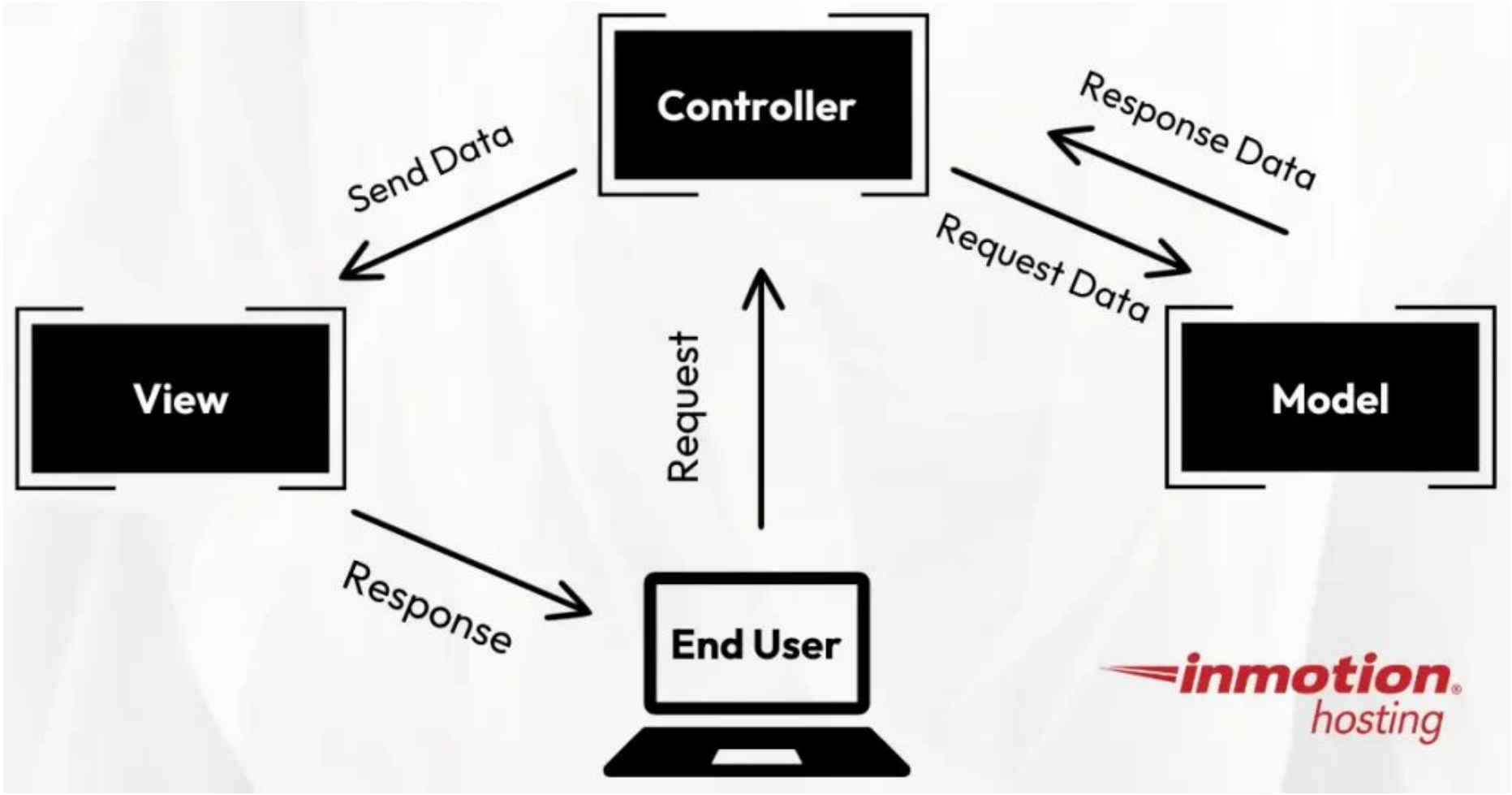
- Šta su to relacione baze podataka?
- Koje još vrste baza podataka postoje?
- Bitni termini
  - Entitet, atribut, veze, primarni ključ, strani ključ, upiti, relaciona algebra, SQL



ELEVATE X

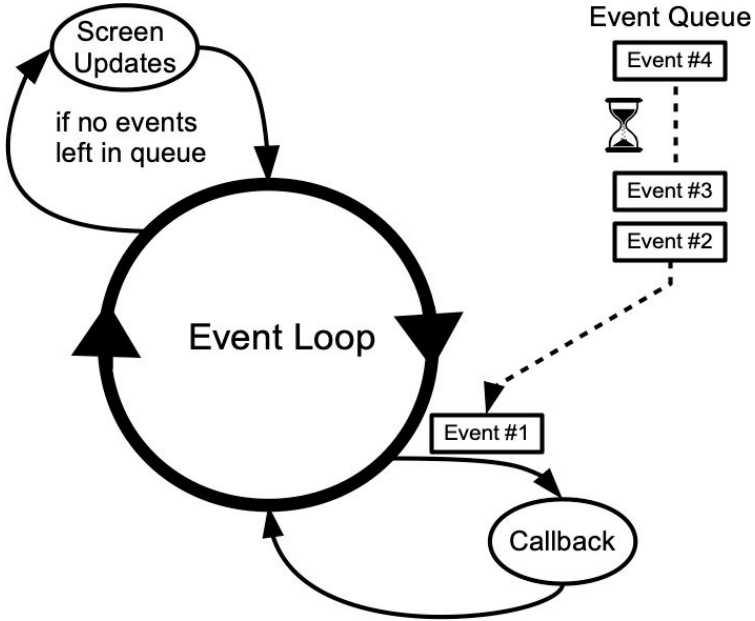
Frontend vs. Backend




Controller
Send Data
Response Data
Request Data
View
Model
Request
Response
End User
inmotion hosting

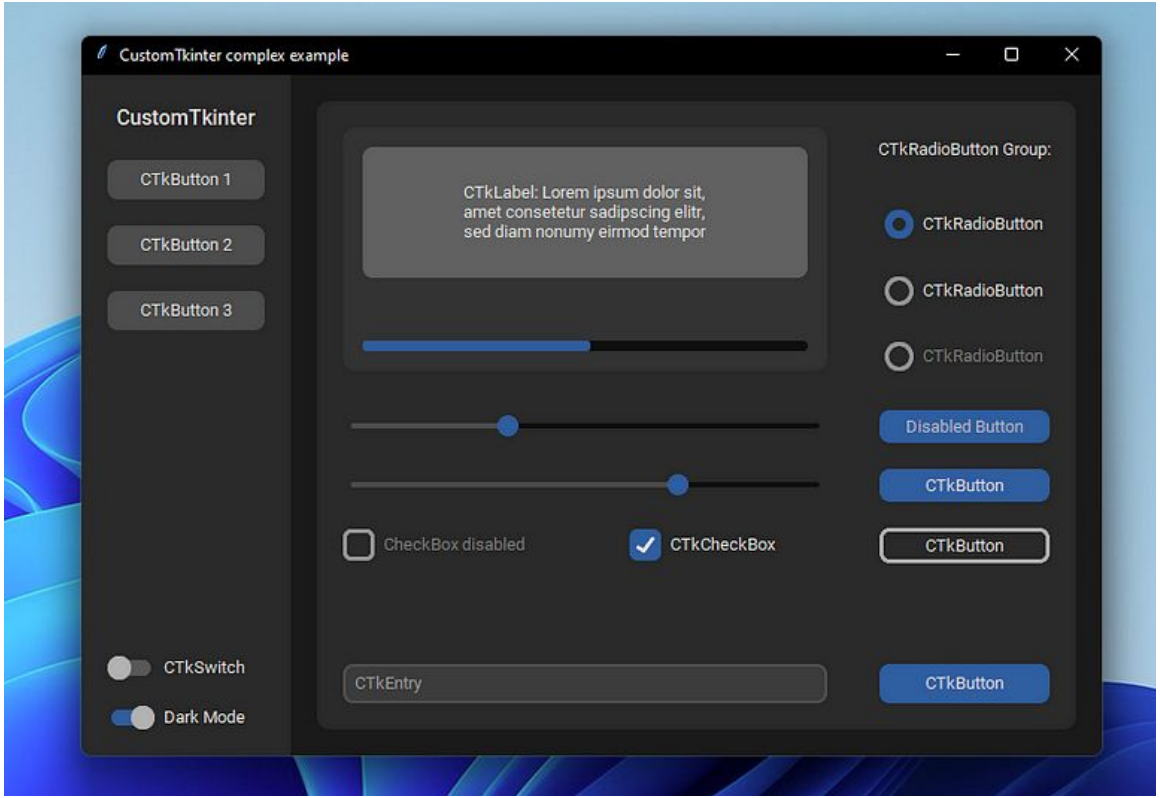
Event driven programiranje

- **Event Loop** i obrada događaja – Program čeka i obrađuje događaje (klikove, unose, poruke) putem glavne petlje.
- **Izbjegavanje blokiranja** – Dugotrajne operacije dijeliti na manje zadatke (after u Tk) ili koristiti asinhroni pristup.
- **Višenitnost** i GUI – GUI operacije moraju ostati u glavnoj niti, dok se sporedne operacije mogu izvoditi u pozadinskim nitima.



Izvor: link

Tkinter


CustomTkinter complex example
CustomTkinter
CTkButton 1
CTkButton 2
CTkButton 3
CTkSwitch
Dark Mode
CTkLabel: Lorem ipsum dolor sit,
amet consetetur sadipscing elitr,
sed diam nonumy eirmod tempor
CheckBox disabled
CTkCheckBox
CTkEntry
CTkRadioButton Group:
CTkRadioButton
CTkRadioButton
CTkRadioButton
Disabled Button
CTkButton
CTkButton
CTkButton

Custom Tkinter

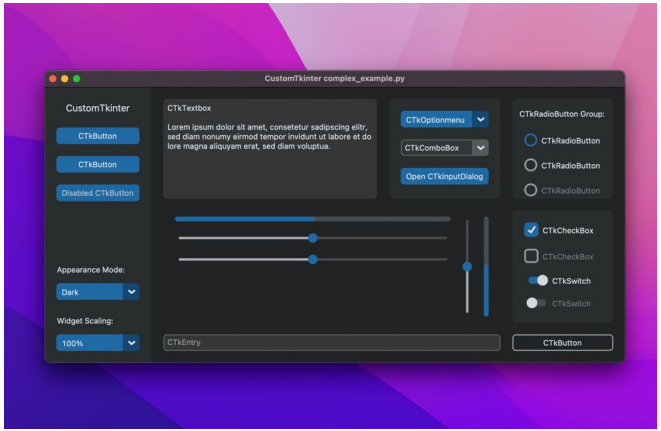
- Moderna Python biblioteka za GUI, bazirana na Tkinter-u. Potrebno je instalirati *pip install customtkinter.*
- Pruža savremene i prilagodljive widget-e.
- Prednosti korišćenja:
  - Jednostavnost,
  - Konzistentan izgled na različitim platformama
- U kodu sa strane prikazan je kod za kreiranje osnovnog prozora.
- Za dodavanje naslova i definisanje dimenzija obično se koristi sledeći kod:

```python
app.title("Moja aplikacija")
app.geometry("400x150")
```

- Više detalja o kreiranju prozora možete pronaći na linku.

```python
import customtkinter

app = customtkinter.CTk()
app.mainloop()
```

CustomTkinter elementi

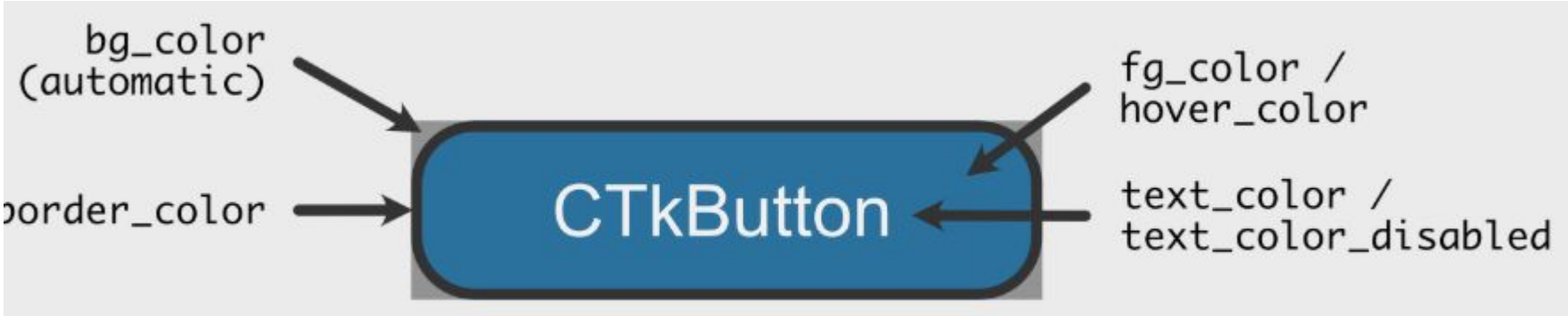


- CustomTkinter koristi više elemenata za prikaz inteterfejsa. Koristi se termin widget.
  - **CTkButton** je widget za kreiranje dugmeta.
  - **CTkLabel** je widget za prikazivanje teksta ili slike. Primjer upotrebe: nazivi, opisi, naslovi.
  - **CTkEntry** widget se koristi za unos kratkog teksta (jedna linija). Primjer upotrebe: unos imena, lozinke, email adrese.
  - **CTkTextbox** widget za višelinijski unos i prikaz teksta. Primjer upotrebe: bilješke, komentari, dnevnik aplikacija.
  - **CTkCheckBox** widget za biranje više opcija. Primer upotrebe: opcije podešavanja, pristajanje na uslove.
  - **CTkRadioButton** je widget za biranje jedne opcije među ponuđenim. Primjer upotrebe: izbor pola, jezika ili nivoa težine u aplikaciji.
  - **CTkSlider** je widget za izbor vrijednosti iz određenog raspona. Primjer upotrebe: podešavanje glasnoće, osvetljenja, veličine elemenata.
  - **CTkProgressBar** je widget za prikaz napretka operacije. Primjer upotrebe: preuzimanje fajlova, učitavanje sadržaja.
  - **CTkSwitch** je widget za uključivanje/isključivanje opcija. Primjer upotrebe: dark/light mode, uključivanje notifikacija.
  - **CTkComboBox** je widget koji omogućava izbor opcije iz padajućeg menija. Primjer upotrebe: izbor države, jezika ili teme.
  - **CTkOptionMenu** predstavlja padajući meni koji omogućava korisnicima izbor jedne opcije iz liste. Primjer upotrebe: izbor jezika, tema ili drugih podešavanja.

CustomTkinter dodatni elementi

- **CTkFrame** kontejner widget koji služi za organizaciju drugih widget-a. Primjer upotrebe: grupisanje widget-a, kreiranje različitih sekcija unutar GUI aplikacije.
- **CTkScrollableFrame** predstavlja okvir sa mogućnošću skrolovanja, koji omogućava prikaz većeg broja widgeta unutar ograničenog prostora. Primjer upotrebe: Prikaz liste stavki, poruka ili drugih sadržaja koji premašuju veličinu prozora.
- **CTkScrollbar** je klizač koji omogućava korisnicima navigaciju kroz sadržaj koji ne može biti prikazan u celosti unutar prozora. Primjer upotrebe: Skrolovanje kroz tekstualne dokumente ili liste.
- **CTkSegmentedButton** je grupa povezanih dugmadi koja omogućava izbor jedne opcije iz skupa. Primjer upotrebe: Prebacivanje između različitih prikaza ili režima rada unutar aplikacije.
- **CTkTabview** je widget koji omogućava organizaciju sadržaja u više tabova, slično kao u internet pregledačima. Primjer upotrebe: Grupisanje različitih sekcija aplikacije unutar jednog prozora.
- Više detalja možete pronaći u dokumentaciji (link)

Boje i tema



Tema se može podesiti sledećim kodom:
customtkinter.set_default_color_theme("dark-blue") # Dostupne teme: "blue" (standard), "green", "dark-blue"

Boja elemenata se može podesiti na sledeći način:
button = customtkinter.CTkButton(root_tk, fg_color="red")
button = customtkinter.CTkButton(root_tk, fg_color="#FF0000") button = customtkinter.CTkButton(root_tk, fg_color=("#DB3E39", "#821D1A")) # tuple boja za light i dark mode

Dodavanje dugmeta

Jedan od glavnih elemenata koji su dio interfejsa za CustomTkiner je dugme. Primjer postavljanja dugmeta na interfejs:

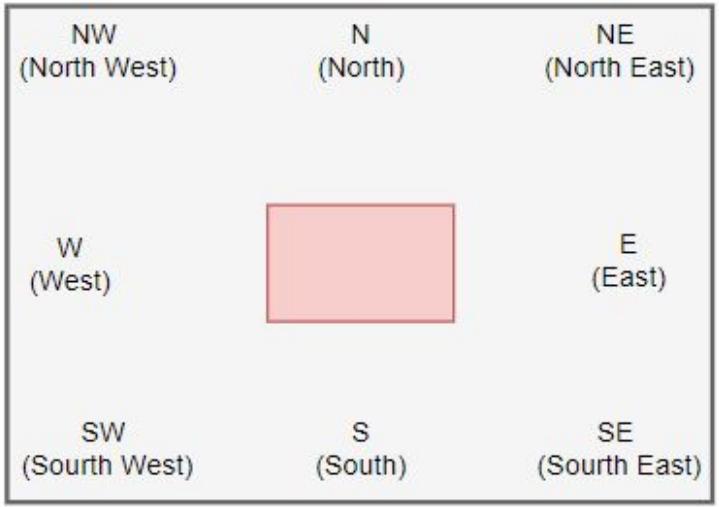
```
def dugme_callback():
    print("Dugme je pritisnuto")

dugme = customtkinter.CTkButton(app, text="Moje dugme", command=dugme_callback)
dugme.grid(row=0, column=0, padx=20, pady=20)
```

Grid menadžer

- Sistem za raspoređivanje vidžeta u redove i kolone.
- Preporučuje se zbog fleksibilnosti i responzivnosti.
- Postavljanje težine kolona i reda se može definisati na sledeći način:

  app.grid_columnconfigure(0, weight=1)
- Proširenje dugmeta unutar ćelije je moguće pomoću sticky parametra:

  dugme.grid(row=0, column=0, padx=20, pady=20, sticky="ew")
- Više detalja možete pronaći na linku.



Dodavanje više elemenata

- Naravno, moguće je dodati više elemenata u sami grid, Primjer sa checkbox:

```
checkbox_1 = customtkinter.CTkCheckBox(app, text="Opcija 1")
checkbox_1.grid(row=1, column=0, padx=20, pady=(0, 20), sticky="w")

checkbox_2 = customtkinter.CTkCheckBox(app, text="Opcija 2")
checkbox_2.grid(row=1, column=1, padx=20, pady=(0, 20), sticky="w")
```

- Moguće je proširiti da jedan element zauzme više kolona/redova:

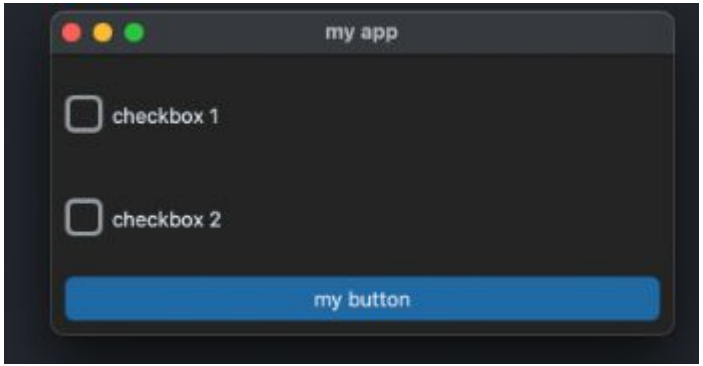
```
dugme.grid(row=0, column=0, padx=20, pady=20, sticky="ew", columnspan=2)
```

- Moguće je podesiti širine različitim kolonama, da budu iste:

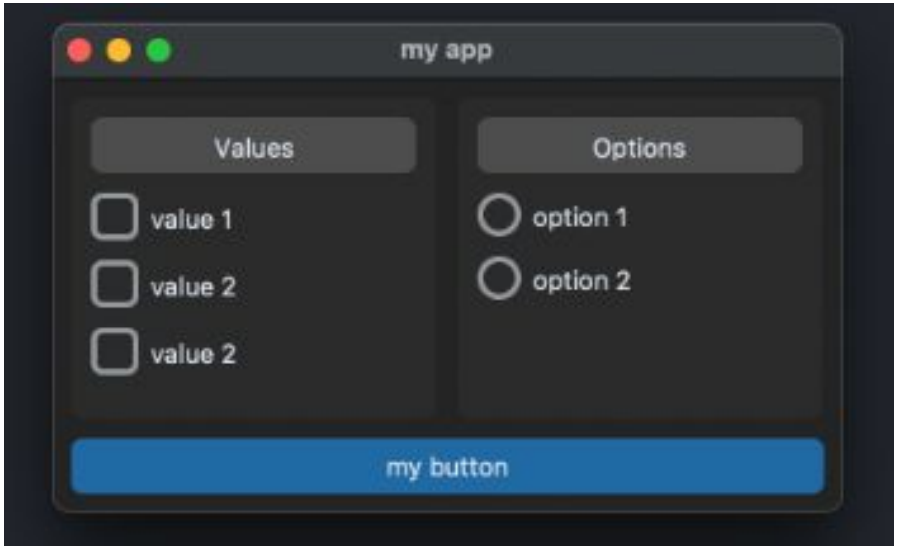
```
app.grid_columnconfigure((0, 1), weight=1)
```

Korišćenje frejmova

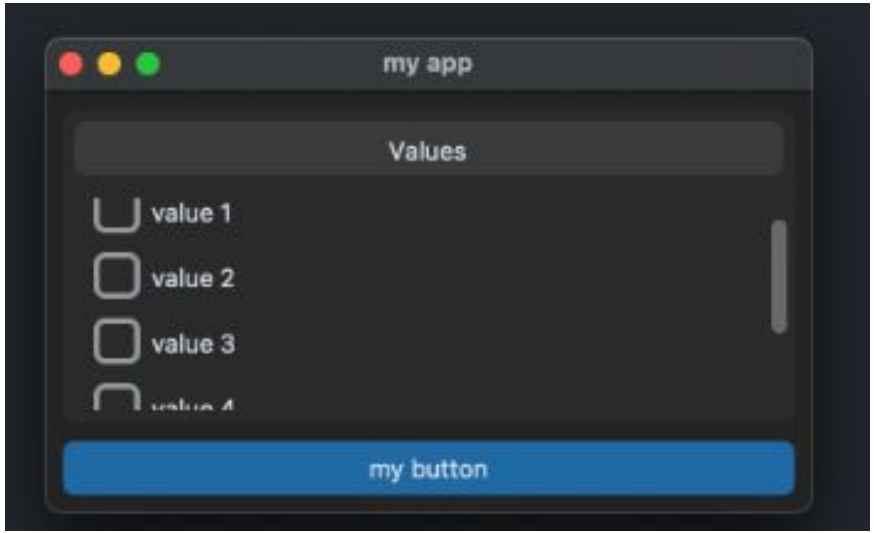
- Kontejneri za grupisanje drugih vidžeta.
- Omogućavaju bolju organizaciju i strukturu GUI aplikacija.
- Obično se koriste da povežu više widgeta u jednu smislenu grupu radi lakše kontrole čime se olakšava održavanje koda.
- Osim toga, stvara se mogućnost nezavisnog uređivanja djelova interfejsa.
- Takođe, kasnije je moguće upotrijebiti frejmove u različitim djelovima aplikacije.
- Primjer.



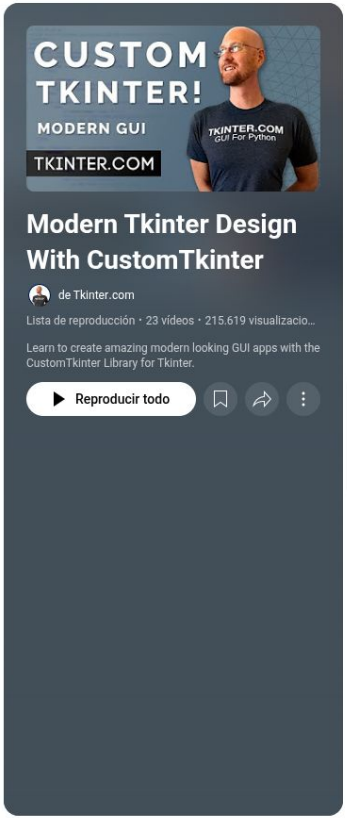
Primjer sa checkbox i option



Primjer 1



Primjer 2



Buscar

Crear

1 Modern GUI Design With CustomTkinter! - Tkinter CustomTkinter 1
Tkinter.com • 149 K visualizaciones • hace 1 año

2 Modern Buttons In CustomTkinter - Tkinter CustomTkinter 2
Tkinter.com • 39 K visualizaciones • hace 1 año

3 Entry Widgets in CustomTkinter - Tkinter CustomTkinter 3
Tkinter.com • 29 K visualizaciones • hace 1 año

4 Check Boxes in CustomTkinter - Tkinter CustomTkinter 4
Tkinter.com • 15 K visualizaciones • hace 1 año

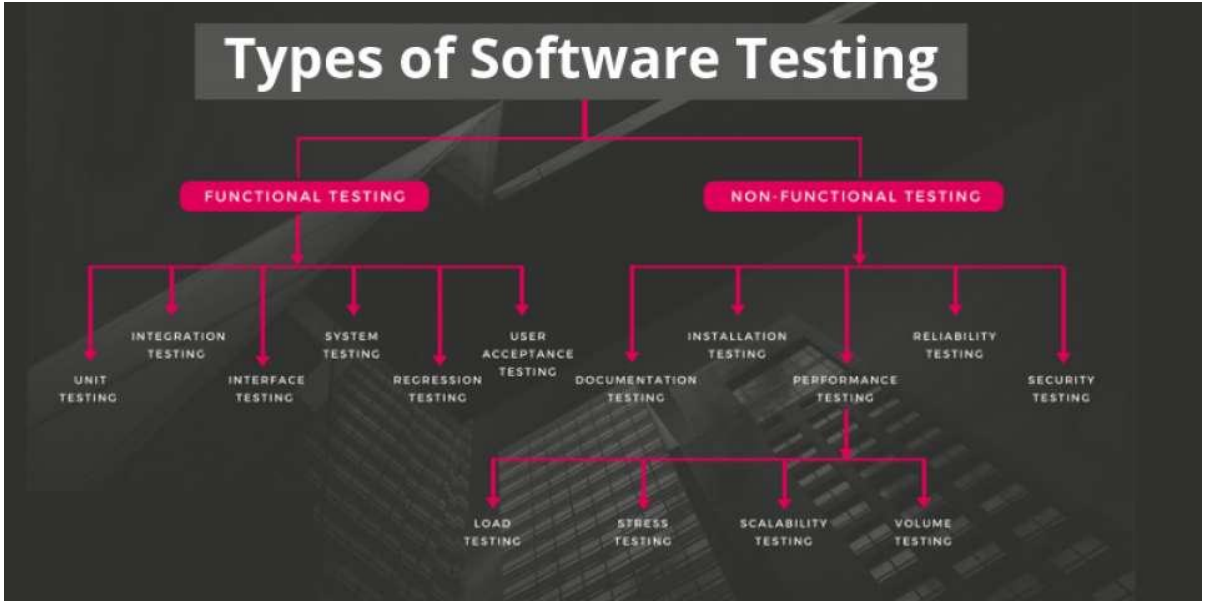
5 Combo Boxes in CustomTkinter - Tkinter CustomTkinter 5
Tkinter.com • 20 K visualizaciones • hace 1 año

6 Progress Bars in CustomTkinter - Tkinter CustomTkinter 6
Tkinter.com • 18 K visualizaciones • hace 1 año

7 Radio Buttons in CustomTkinter - Tkinter CustomTkinter 7
Tkinter.com • 9,6 K visualizaciones • hace 1 año

Izvor: link

Pisanje testova



Izvor: https://hackr.io/blog/types-of-software-testing

Primjer, benefiti testiranja softvera

- **Primjer**
- Poboljšava bezbjednost aplikacija
- Za veće aplikacija smanjuje troškove posmatrajući dugoročno
- Poboljšava kvalitet i stabilnost proizvoda
- Utiče pozitivno na performanse cjelokupnog proizvoda (aplikacije)
- Lakše i bezbjednije dodavanje novih funkcionalnosti

Pitanja