Univerzitet Crne Gore
ELEKTROTEHNIČKI FAKULTET
Studije primijenjenog računarstva

Softversko inženjerstvo

-dokumentacija mini-projekta-

Tema: Youtube Media Player

Projekat radio:

1. 1/19 Ćupić Ivan

Podgorica, --.06.2021. godine

Kratak opis projekta:

Svi su korisnici Youtube mreže u nekom trenutku bili prinuđeni da odgledaju makar jednu reklamu, onu od 5 sekundi, na početku snimka, prije nego što su je mogli preskočiti. Često se desi da postoji još po jedna u sredinu i na kraj snimka. U poslednje vrijeme, desio se nagli skok u broju reklama. Njih često ne možemo preskočiti i dužine su oko 30 sekundi. Ukoliko gledate neki dokumentarac od 90 minuta, odgledaćete približno 10 reklama. Množeći taj broj sa prosječnim trajanjem reklame, dobijamo da ćemo potrošiti 5 minuta gledajući te reklame. Želja za gledanjem snimaka bez reklama me je podstakla da napravim Youtube Player.

Ovom aplikacijom takođe želim da riješim još jedan bitan problem. Naime, neki snimci su previše tihi i ne mogu se kvalitetno čuti ni pomoću slušalica. Do sada je to bio neizbježan problem, koji sam morao da prihvatim. Dosjetio sam se da tu može doći do izražaja jedinstvenost VLC plejera, koji je jedan među rijetkima koji omogućava pojačavanje jačine zvuka na vrijednost veću od originalne (do x2 veću vrijednost). Ovim programom sam iskoristio tu specifičnu osobinu i omogućio da se i tihi Youtube snimci odlično čuju.

Youtube Media Player je aplikacija koja prikazuje željeni Youtube snimak. Glavni djelovi aplikacije su realizovani kroz 3 cjeline:

- Prva cjelina predstavlja grafički prikaz aplikacije koja je integrisana kroz Tkinter GUI.
- Druga cjelina predstavlja video plejer koji je realizovan kroz VLC open source biblioteku. Biblioteka pruža dosta mogućnosti za rad sa media fajlovima i pogodna je za strimovanje Youtube snimaka.
- Treci cjelina aplikacije je realizovan kroz Pafy biblioteku, čiji je osnovni zadatak preuzimanje informacija Youtube snimka (URL, naslov, opis …) u JSON formatu i to mi omogućava da prilagodim funkcionalnosti biblioteke svojim željama.

Opis zadataka i doprinosa članova projektnog tima

Ivan Ćupić:

Projektovao program, izvršio dizajn aplikacije, dodao svu funkcionalnost, vršio testiranje kao i pisanje čitave projektne dokumentacije.

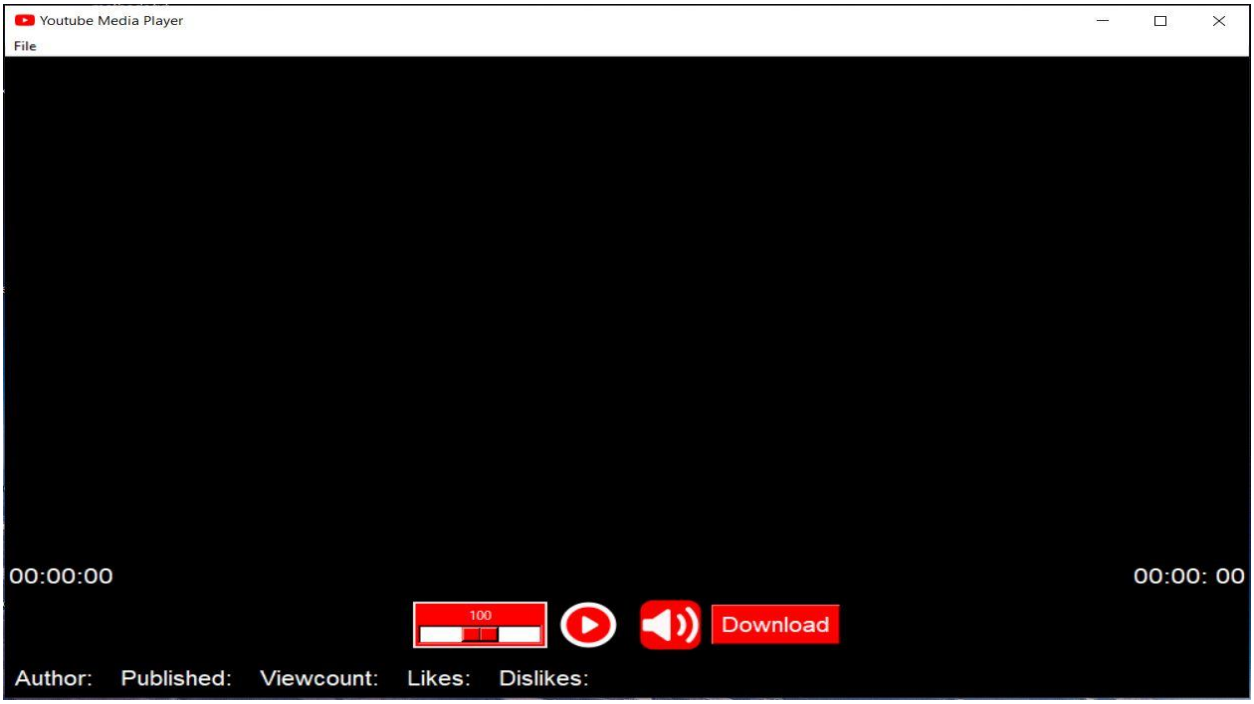
Opis opštih funkcionalnosti softvera

Aplikacija Youtube Media Player je realizovana kroz programski jezik Python i grafički je predstavljena kroz Tkinter GUI. U pitanju je video plejer koji identifikuje željeni snimak za reprodukciju pomoću web adrese snimka i prikazuje ga. Jedini uslov za njeno funkcionisanje, tj. prikazivanje željenog snimka, jeste pristup internetu.

Aplikacija je jednostavna i izuzetno intuitivna za korišćenje (*slika 1*). Praćeni su svi savjeti za dobar *user-interface*. Interfejs je prijatan za oko i koristi jedan set boja, crvenu i bijelu (crna je boja pozadine), zbog čega korisnikovo čulo vida nije vizuelno opterećeno. Interfejs je i

simetričan. Simetričnost je nešto na šta je naš mozak navikao i to i očekuje (naučno je dokazano da što je lice neke osobe simetričnije, to ga doživljavamo ljepšim).

U cilju poboljšanja *user-experience*, dizajn se fokusira na ono što je i srž ove aplikacije, a to je video plejer. Zato je on u sredini grafičkog prikaza, i zauzima njegov najveći dio. Odmah ispod njega su upravljačke kontrole. Kontrolna dugmad su dovoljno velika da ih je lako koristiti. Tekst koji prikazuje proteklo vrijeme i ukupno dužinu snimka su dovoljno velika da ih je lako čitati, ali opet ne previše glomazna, da ne bi odvraćale pažnju.



Slika 1 Grafički prikaz čitave aplikacije

Kada korisnik pokrene program (koji je u .exe formatu, te nije potrebno nikakvo dodatno okruženje za njega), pojavljuje se početna stranica sa *slike 1*. Aplikacije se sastoji od menija (*slika 2*), koji pruža mogućnost dodavanje URL youtube snimka (na taj način se i plejer povezuje sa youtube serverima i strimuje sadržaj u realnom vremenu) i opcije exit koja se koristi za izlazak iz programa i prekidanje svih pozadinskih procesa koji samom plejeru i daju funkcionalnost.

Sam grafički prozor se sastoji od video plejera koji je integrisan pomoću VLC biblioteke, zatim sekcije koja označava vrijeme koje je proteklo (bijeli tekst lijevo od kontrolnih dugmadi) i ukupno vremensko trajanje youtube snimka (bijeli tekst sa desne strane kontrolnih dugmadi). Oba vremena se prikazuju u standardnom evropskom formatu HH:MM:SS, gdje HH, MM i SS označava jedinicu vremena (sate, minute i sekunde) u dvocifrenom zapisu (npr. prikazaće se 02 umjesto 2 sata).

Potom imamo sekciju koja sadrži same upravljačke kontrole programa. Ona se sastoji od četiri dugmeta.

Prvo dugme s lijeva je **klizač** koji služi za precizno podešavanje jačine zvuka. Početna pozicija ovo klizača je na sredini, odnosno 100% jačine zvuke. Tada je zvuk one jačine i kakav stiže do računara. Pomjeranjem ovog klizača ka lijevo, smanjujete jačinu zvuka, tj. jačina zvuka se smanjuje sa 100% do najniže 0% (nema zvuka). Klizač odmah ažurira vrijednost jačine zvuka u procentima, tj. ne morati stati sa pomjeranjem klizača da bi vam se vrijednost učita, već je moguće u toku pomjeranja klizača, bez puštanja istog, dobiti vrijednost jačine zvuka u procentima u odnosnu na podrazumijevanu (originalnu) vrijednost. Pomjeranjem klizača mišem ka desno od podrazumijevane, dolazi do povećanja jačine zvuka. Tu dolazi do izražaja jedinstvenost VLC plejera, koji omogućava dvostruko uvećavanje originalne jačine zvuka (na klizaču to je položaj skroz desno, tj. očitavanje od 200%)

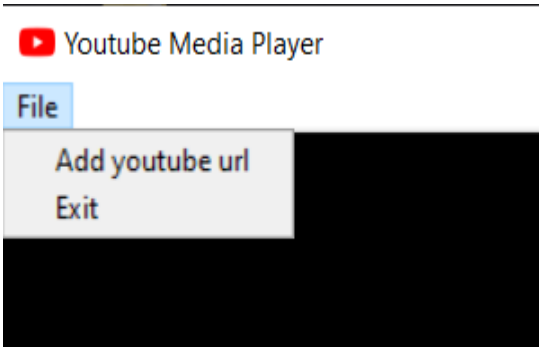
Drugo kontrolno dugme **je play/pause** dugme, koje u zavisnosti od toga da li je video pauziran ili ne, vrši kontra radnju, odnosno pauziranom snimku nastavlja reprodukciju, tj. pauzira snimak ukoliko nije bio. U cilju povećanja pozitivnog korisničkog doživljaja, tom prilikom dolazi i do promjene ikonice. Prateći savjete dobrog dizajna, trenutna ikonica ne pokazuje trenutnu radnju (ako je video pauziran, ne prikazuje se ikonica za pauziranje), već indikuje koju radnju ćete postići njenim pritiskom (ako je video pauziran, prikazuje se ikonica *play*).

Treće dugme je **mute/unmute** dugme koje isključuje ili uključuje zvuk snimka. Funkcionalni dizajn mu je isti kao i za play/pause dugme.

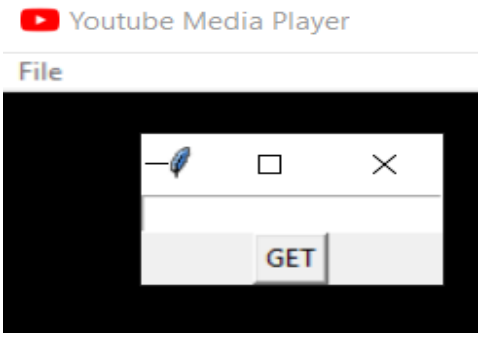
Na samom kraju ove kontrolne sekcije imamo četvrto dugme, tj. **download** dugme. Ovo dugme koristi funkciju u skopu **Pafy** biblioteke i korisniku pruža mogućnost preuzimanja youtube snimka za offline gledanje, što je jako korisno u slučaju potrebe za reprodukcijom ovog youtube snimka u periodu kada nemate pristup internetu. Preuzeti snimci se smještaju u direktorijumu tj. folderu gdje se i sam program nalazi. Snimci se preuzimaju u .mp4 formatu. U zavisnosti od brzine vašeg interneta, ovo može da potraje.

Poslednja sekcija sadrži informacije o samom youtube snimku. Ove informacije se preuzimaju direktno sa youtube-ovih servera i govori nam o imenu naloga koji je ovaj snimak i postavi, zatim datum i vrijeme objavljivanja snimka u (datum se prikazuje u formatu YYYY-MM-DD gdje su YYYY, MM i DD godina, mjesec i dan objavljivanja snimka u numeričkom obliku, tj. MM je redni broj mjeseca u godini, a ne slovna skraćenica npr. *05*, a ne *MAY.* Vrijeme se prikazuje u formatu opisanom kod prve sekcije). Sledeća informacija je *viewcount*, tj. ukupan broj pregleda u trenutku njegovog strimovanja. Poslednje dvije informacije nam govore o broju tkz. *sviđanja* i *nesviđanja*, odnosno o broju korisnika koji sadržaj ovog konkretnog snimka odobrava, odnosno neodobrava.

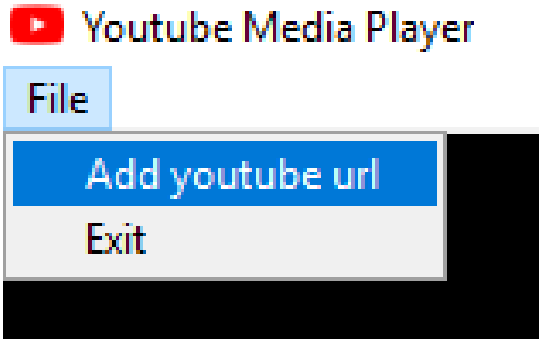
Posebna pažnja je posvećena brzini rada plejera, tako da je programski kod ove aplikacije efikasan i brz.



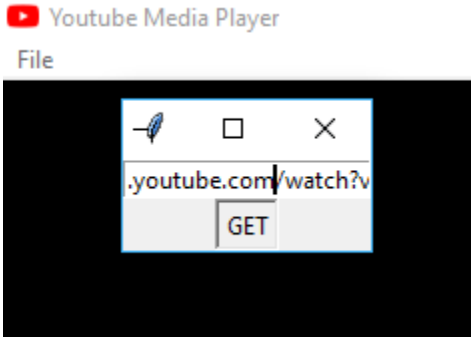
Slika 2 Meni



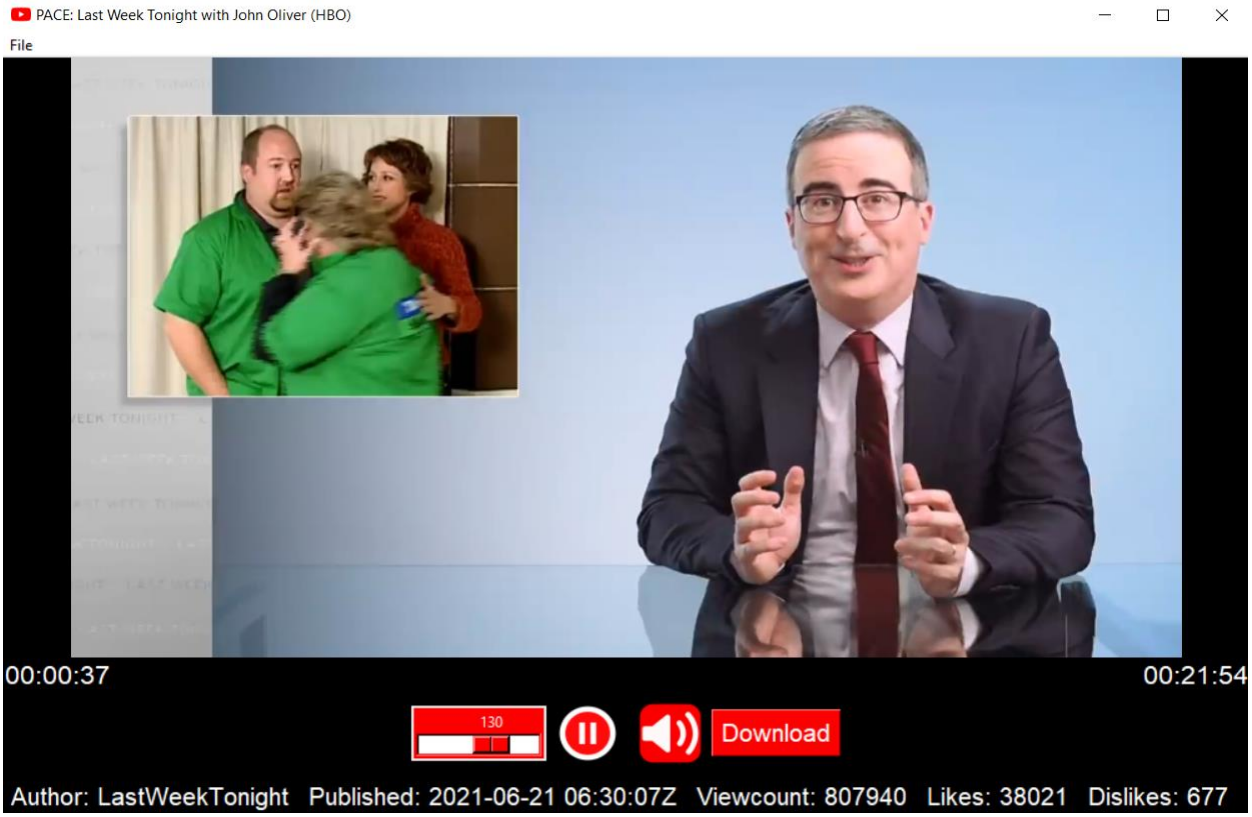
Slika 4 Dodavanje linka



Slika 3 Odabir opcije menija



Slika 5 Dodavanja URL adrese



Slika 6 Izgled video plejera tokom reprodukcije sadržaja

Uputstvo:

Kada korisnik odluči da reprodukuje youtube snimak, on to postiže tako što pritisne na *meni* opciju plejera, pod nazivom *File* (*slika 2*). Tu mu se otvara podmeni (detaljno prethodno opisan) i bira prvu opciju *menija*, tj. *Add youtube url* (*slika 3*). Tu mu iskače tkz. *modal window*, odnosno prozorčić (*slika 4*) gdje se vrši unos URL adrese youtube snimka.

Unos netačnog linka (npr. *google.com/mail*) neće ništa pokrenuti, već će program čekati unos ispravnog linka ili izlaz iz prozorčića. Nakon unosa, korisnik pritiska dugme *get* (*slika 5*).

Zatim, pomoću Pafy biblioteke, dobijamo odgovarajući JSON objekat koji sadrži sve informacije o snimku (neke od njih se prikazuju u poslednjoj sekciji programa i ranije su opisani), kao i sami link za njegovo strimovanje . Taj link se dalje prosleđuje VLC plejeru i snimak se reprodukuje (*slika 6*). Kao što se vidi sa te slike može vidjeti, naslov aplikacije (koji je prije reprodukcije bio *„Youtube Video Player“*) je zamijenjen nazivom snimka. Sada se i postavlja ukupno trajanje snimka i sve opisne informacije o snimku (o kojima je već dosta rečeno) se ispunjaju i prikazuju.

Dalje možete podešavati jačinu zvuka po volji, gasiti ton ili jednostavno pauzirati reprodukciju snimka. Po potrebi, pritiskom na *download* dugme, započinje se preuzimanje snimka.

Analiza softvera i njegovih segmenata

```
#konfiguracija glavnog Tk window
root = Tk()
root.config(width=1000, bg="black")
titleImage = PhotoImage(file = "images/youtube.png")
root.iconphoto(False, titleImage)
root.title("Youtube Media Player")

#konfiguracija video player-a
player = Screen(root)
player.config(width=1000, height=500)
player.pack(fill=X)
root.protocol('WM_DELETE_WINDOW', player.exitEverything)
#meni
toolbar = Menu(root, bg="black", fg="white" )
root.config(menu=toolbar)
submenu = Menu(toolbar, tearoff=0)
toolbar.add_cascade(label="File", menu=submenu)
submenu.add_command(label="Add youtube url", command=openUrlWindow)
submenu.add_command(label="Exit", command=player.exitEverything)
```

Slika 7 Inicijalizacija prozora aplikacije

```
class Screen(Frame):

    def __init__(self, parent, *args, **kwargs):
        Frame.__init__(self, parent, bg='black')
        self.parent = parent
        # Creating VLC player
        self.instance = vlc.Instance()
        self.player = self.instance.media_player_new()

    def GetHandle(self):
        # Getting frame ID
        return self.winfo_id()

    def init_play(self, _source):
        # Function to start player from given source
        Media = self.instance.media_new(_source)
        Media.get_mrl()
        self.player.set_media(Media)
        self.player.set_hwnd(self.winfo_id())
        self.player.play()
        self.player.audio_set_mute(False)
        updateTime()
```

Slika 8 Glavna klasa

Konfiguracija glavnog prozora aplikacije (*slika 7*) se sastoji iz 3 cjeline:

1. Prva cjelina pod komentarom #konfiguracija glavnog Tk window-a inicijalizuje glavni prozor aplikacije pomoću *Tk()*, ugrađene Tkinter klase. Potom podešava širinu prozora na 1000 piksela, pozadinsku boju na crnu kao i ikonicu glavnog prozora (Youtube-ov znak).
2. Drugi dio predstavlja konfiguraciju VLC plejera i to se postiže tako što je player instanca klase Screen kojoj se prosleđuje glavni prozor aplikacije; tj. root. Konstruktorom se prvo

vrši inicijalizacija svog parent elementa na root element; tj. prozor aplikacije. Promjenjiva player postaje instanca VLC klase sa svojim metodama. I na kraju dobijamo player koji predstavlja instancu Media Player klase. Funkcija init_play klase Screen za argument uzima _source tj. link youtube snimka. Suština ove funkcije je da zapravo uzme media objekat (u ovom programu to je youtube snimak) i automatski reprodukuje video (koji nije mutiran) i pokreće funkciju UpdateTime() koja prikazuje proteklo vrijeme video snimka (*slika 8*).

3. Treći i poslednji dio postavlja dodatne opcije u vidu opcija u meniju (*slika 7*). Taj meni dobija naziv File. Te opcije su dodavanje linka youtube snimka; tj. Add youtube url (*slike 3 i 4*) i izlazak iz programa; tj. opcija Exit (*slika 2*).

```
def openUrlWindow():
    urlWindow = Tk()
    urlWindow.title("Enter youtube URL")
    e1 = Entry(urlWindow)
    b3 = Button(urlWindow, text="GET", command= lambda: getUrl(e1, urlWindow))
    e1.pack()
    b3.pack()

def getUrl(e1, window):
    url = e1.get()
    if url !="":
        global best
        pafy.set_api_key("AIzaSyBkFqCEA2qp5hkxx6SP1P5ygDQii68qeI8")
        video = pafy.new(url)
        root.title(video.title)
        best = video.getbest()
        player.init_play(best.url)
        duzinaVidea["text"]= video.duration
        inforAboutVideo(video)
        window.destroy()
```

Slika 9 Dodavanje linka u plejeru

Dodavanje i puštanje youtube videa omogućavaju funkcije sa *slike 9*. Funkcija openUrlWindow kreira mini Tkinter prozor; tj. takozvani „modal window“ (*slika 5*). Pritiskom na dugme sa nazivom get na dotičnom modal window-u prosleđuje se uneseni youtube link. Pafy biblioteka pruža neke funkcionalnosti youtube API-a, ali zbog određenih ograničenja koje ozbiljno limitiraju moju tehničku zamisao, kreirao sam sopstveni koji mi pruža mnogo više mogućnosti. Obzirom da korisnici mogu imati različite brzine interneta, koje zavise od zemlje, internet provajdera i okruženosti objektima koji tu brzinu limitiraju, Youtube sve video snimke čuva u 5 različitih rezolucija (240p, 260p, 480p, 720p i 1080p) i u zavisnosti od trenutne korisnikove brzine, mijenja prikazivanu rezoluciju. Zbog jednostavnosti, ovaj program preuzima link najboljeg kvaliteta snimka i smješta ga u varijablu best i zatim ga prosleđuje metodi init_play klase Screen (*slika 8*) koja ga dalje reprodukuje u skladu sa prethodno opisanom procedurom. Nakon toga se glavnom prozoru programa dodaje se dužina video snimka i prosleđuje se *.json*

objekat. Kroz funkciju infoAboutVideo() se popunjavaju podaci o video snimku koji se reprodukuje (podaci o autoru, datumu objavljivanja, broju pregleda, broj sviđanja i nesviđanja) i zatim se *„modal window“* (u kojem se unosio link youtube snimka) uništava (*slika 10*).

```
def inforAboutVideo(video):
        author.configure(text="Author:  " + video.author)
        published.config(text="Published:  " + video.published)
        viewcount.config(text="Viewcount:  " + str(video.viewcount))
        likes.config(text="Likes:  " + str(video.likes))
        dislikes.config(text="Dislikes:  " + str(video.dislikes))
```

Slika 10 Funkcija koja dodaje informacije o snimku

Tom prilikom se mijenjaju podrazumijevane vrijednosti koje su postavljene kao placeholder na početku pokretanja programa (*slika 11*).

```
#informacije o video
infoOVideo = Frame(root, bg="black")
author = Label(infoOVideo, text="Author: ", fg="white",  font=("Arial", 15), bg="black")
author.pack(side=LEFT, padx=5, pady=5,fill=X)
published = Label(infoOVideo, text="Published: ", fg="white",  font=("Arial", 15), bg="black")
published.pack(side=LEFT, padx=5, pady=5,fill=X)
viewcount = Label(infoOVideo, text="Viewcount: " ,  fg="white",  font=("Arial", 15), bg="black")
viewcount.pack(side=LEFT, padx=5, pady=5,fill=X)
likes = Label(infoOVideo, text="Likes: " ,  fg="white",  font=("Arial", 15), bg="black")
likes.pack(side=LEFT, padx=5, pady=5,fill=X)
dislikes = Label(infoOVideo, text="Dislikes: ",  fg="white",  font=("Arial", 15), bg="black")
dislikes.pack(side=LEFT, padx=5, pady=5,fill=X)
infoOVideo.pack(fill=X)
```

Slika 11 Placeholder informacija o video snimku

Sledeća cjelina koja je zanimljiva je funkcija updateTime() koja se prvi put poziva na dnu init_play funkcije klase Screen (*slika 12*).

```
def updateTime():
    global cancel
    player.getVideoTime()
    if duzinaVidea["text"] != trenutnoVrijeme["text"]:
        cancel = root.after(1000, updateTime)
```

Slika 12 Ažuriranje vremena

UpdateTime() funkcija služi da ažurira proteklo vrijeme u reprodukciji youtube video snimka pomoću getVideoTime() funkcije koja predstavlja metodu klase Screen. Prvo se poziva getVideoTime() funkcija koja pomoću ugrađene funkcije VLC biblioteke uzima trenutno vrijeme u milisekundima, a potom iz tog vremena uzima sekunde, minute i sate. Zatim ih prikazuje u datetime formatu HH:MM:SS i ažurira varijablu trenutno vrijeme koja predstavlja labelu glavnog tkinter prozora (*slika 6, lijevo*).

Potom UpdateTime() funkcija provjerava da li se trenutno vrijeme razlikuje od ukupne dužine video snimka. Ukoliko video snimak nije došao do kraja, after() funkcijom nakon 1 sekunde rekurzivno poziva sebe dok se taj uslov ne ispuni.

```
def getVideoTime(self):
    millis = self.player.get_time()
    seconds=int((millis/1000)%60)+1
    if(seconds == 60):
        seconds = 0
    minutes=int((millis/(1000*60))%60)
    hours=int((millis/(1000*60*60))%24)
    time = datetime.time(hours, minutes, seconds)
    trenutnoVrijeme["text"]=time
```

Slika 13 Funkcija koja uzima proteklo vrijeme snimka

Preuzimanje video snimka

Kada korisnik poželi da preuzme dotični video snimak za offline gledanje, pritiskom na dugme download pokreće se funkcija downloadVideo() (*slika 14*) koja koristi ugrađenu metodu Pafy biblioteke i preuzima video snimak. Brzina zavisi od internet brzine. Preuzeti snimak se smješta u direktorijumu gdje se nalazi .exe fajl.

```
def downloadVideo():
    player.pause()
    best.download()
    player.pause()
```

Slika 14 Funkcija za preuzimanje video snimka

Izlazak iz programa

Zbog potrebe da se izmijeni podrazumijevano ponašanje X (close) dugmeta Tkinter glavnog prozora aplikacije, kao i potrebe omogućavanja funkcionalnosti Exit opcije u meniju File, napisao sam funkciju exitEverything(). Dotična funkcija osigurava da se ne nastavi rekurzivno pozivanje funkcije UpdateTime(). Takođe, zbog specifičnosti dinamike Tkinter i VLC modula, dodate su osiguravajuće naredbe koje garantuju da kada se izađe iz aplikacije, proces programa zaista i zaustavi. To se postiže tako što se glavni prozor i media plejer uništavaju, a pomoću sys.exit() se izlazi iz python koda i time omogućava čist izlazak iz programa (*slika 15*).

```
def exitEverything(self):
    if "cancel" in globals():
        root.after_cancel(cancel)
    root.destroy()
    self.player.release()
    sys.exit()
```

Slika 15 Funkcija za izlazak iz programa

Pregled literature, gotovih rješenja i softverske dokumentacije

[1] http://www.olivieraubert.net/vlc/python-ctypes/doc/

[2] https://pythonhosted.org/pafy/

[3] https://docs.python.org/3/library/tk.html