

DISCORD BOT APLIKACIJA ZA FILTRIRANJE SADRŽAJA RELIZIRANA NA RASPBERRY PI UREĐAJU

Bačić, Lovre

Undergraduate thesis / Završni rad

2023

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University of Split / Sveučilište u Splitu**

Permanent link / Trajna poveznica: <https://um.nsk.hr/um:nbn:hr:228:493451>

Rights / Prava: [In copyright](#) / [Zaštićeno autorskim pravom.](#)

Download date / Datum preuzimanja: **2024-10-20**



Repository / Repozitorij:

[Repository of University Department of Professional Studies](#)



UNIVERSITY OF SPLIT



SVEUČILIŠTE U SPLITU
SVEUČILIŠNI ODJEL ZA STRUČNE STUDIJE

Preddiplomski stručni studij Informacijska tehnologija

LOVRE BAČIĆ

ZAVRŠNI RAD

**DISCORD BOT APLIKACIJA ZA FILTRIRANJE
SADRŽAJA REALIZIRAN NA RASPBERRY PI
UREĐAJU**

Split, ožujak 2023.

SVEUČILIŠTE U SPLITU
SVEUČILIŠNI ODJEL ZA STRUČNE STUDIJE

Preddiplomski stručni studij Informacijska tehnologija

Predmet: Sigurnost računala i podataka

Z A V R Š N I R A D

Kandidat: Lovre Bačić

Naslov rada: Discord bot aplikacija za filtriranje sadržaja realiziran na Raspberry Pi uređaju

Mentor: Lada Sartori, v. predavač

Split, ožujak 2023.

Sadržaj

Sažetak	1
1. Uvod.....	2
2. Teorijska podloga.....	3
2.1. Python	3
2.2. Raspberry Pi.....	4
2.2.1. Raspbian operativni sustav	7
2.3. Discord	8
2.3.1. Botovi.....	10
2.3.2. API.....	11
3. Realizacija zadatka.....	12
3.1. Instalacija Raspebrry Pi-a.....	12
3.2. Instalacija Discord bota.....	15
3.3. Program za filtriranje sadržaja	18
3.4. Program za unos sadržaja.....	25
3.5. Start-up service za pokretanje Raspberry Pi-a.....	27
3.6. Program za ažuriranje datoteke pomoću crontaba	30
4. Zaključak	33
5. Literatura.....	34
6. Popis slika	35

Sažetak

Ovaj završni rad raspravlja o razvoju Discord bota dizajniranog za filtriranje nepoželjnog sadržaja i zabranu korisnika koji krše pravila poslužitelja. Bot je stvoren korištenjem biblioteke Discord.py i programiran da prepozna i označi neprikladni jezik pomoću popisa unaprijed definiranih zabranjenih riječi. Kada korisnik upiše poruku koja sadrži zabranjenu riječ, bot automatski briše poruku i šalje upozorenje korisniku, pokazujući da korištenje takvog jezika nije dopušteno na poslužitelju. Ako korisnik nastavi kršiti pravila, bot poduzima ozbiljnije radnje, poput isključivanja korisnika. Sve u svemu, Discord bot koji je tema ovog rada pruža učinkovitu i automatiziranu metodu za održavanje sigurnog i odgovarajućeg okruženja na Discord poslužitelju, osiguravajući da korisnici mogu komunicirati bez straha od susreta s uvredljivim ili neprikladnim sadržajem.

Ključne riječi: Discord, filter neželjenog sadržaja, Python, Raspberry Pi

Summary

Discord bot application for content filtering implemented on a Raspberry Pi device

This final paper discusses the development of a Discord bot designed to filter foul language and ban users who violate the server's rules. The bot was created using the Discord.py library and programmed to recognize and flag inappropriate language using a list of pre-defined banned words. When a user types a message that contains a prohibited word, the bot automatically deletes the message and sends a warning to the user, indicating that the use of such language is not allowed on the server. If the user continues to violate the rules, the bot takes more serious actions, such as banning the user. Overall, the Discord bot discussed in this document provides an efficient and automated method for maintaining a safe and appropriate environment on a Discord server, ensuring that users can interact without fear of encountering offensive or inappropriate content.

Keywords: Discord, Python, Raspberry Pi, spam filter

1. Uvod

U današnje digitalno doba korištenje botova u komunikacijskim platformama postalo je sve popularnije zbog njihove sposobnosti automatizacije različitih zadataka, od moderiranja razgovora do pružanja korisnih informacija korisnicima. Discord, popularna komunikacijska platforma među igračima, također je zabilježila porast upotrebe botova za poboljšanje korisničkog iskustva.

U ovom radu istražit će se proces stvaranja Discord bota pomoću programskog jezika Python i njegovu integraciju s Raspberry Pi, računalom veličine kreditne kartice koje omogućuje razne prilagodbe hardvera i softvera. Razmotrit će se različite funkcije koje se mogu uključiti u bota, uključujući slanje poruka, odgovaranje na korisničke unose i dohvaćanje informacija iz vanjskih API-ja (*engl. Application Programming Interface*).

Također, ukazat će se na prednosti korištenja Raspberry Pi za pokretanje Discord bota, kao što su niska potrošnja energije i prenosivost. Istaknut će se izazovi s kojima se susreće tijekom procesa razvoja i pružiti rješenja za uobičajene probleme.

Cilj rada je sveobuhvatno prikazati procesa stvaranja i integracije Discord bota s Python i Raspberry Pi, kao i potencijalne prednosti i izazove koje on donosi.

2. Teorijska podloga

U ovom poglavlju opisane su tehnologije korištene za izradu Discord bota preko Raspberry Pi uređaja.

2.1. Python

Python je računalni programski jezik koji se često koristi za izradu *web* stranica i softvera, automatizaciju zadataka i provođenje analize podataka, predstavljen 1991. Prvo izdanje 0.9.0 napravio je Guido van Rossum, nizozemski programer [1].

Prekretnica kod razvoja Pythona je Python 2.0, objavljen 2000. Python 3.0 objavljen je 2008. s velikim revizijama, ali sada više nije potpuno kompatibilan s prethodnim verzijama. Zadnje stabilno izdanje je 3.10.6, objavljeno 2. kolovoza 2022. godine.

Python je popularan izbor za programiranje Discord botova na Raspberry Pi-u iz nekoliko razloga.

Osnovni razlog je jednostavnost, Python ima jednostavnu i intuitivnu sintaksu koja je lako razumljiva za programere. Također, ima mnogo korisnih biblioteka i alata koji mogu olakšati izradu i razvoj Discord bota.

Python je popularan programski jezik s velikom zajednicom korisnika koji stvaraju i dijele biblioteke i modul koji se mogu koristiti za programiranje Discord bota. To znači da postoji puno podrške za Python i da se može pronaći mnogo korisnih resursa na internetu.

Postoji nekoliko biblioteka za Python koje su posebno dizajnirane za programiranje Discord bota, poput `discord.py`. Ove biblioteke imaju mnogo funkcija i metoda koje olakšavaju povezivanje osobnog bota s Discord API-jem i omogućuju stvaranje bota s bogatim funkcionalnostima. Python je programski jezik koji se može brzo razvijati i

prototipirati. Također, relativno je brz i učinkovit u radu. Sve ovo znači da se može razviti i testirati Discord bot, a zatim ga učiniti dostupnim korisnicima.

Raspberry Pi je mali i moćan uređaj, ali ima ograničene resurse. Python je programski jezik koji se može skalirati prema potrebama projekta. To znači da se može razviti i izgraditi Discord bot koji je prilagođen specifičnim zahtjevima projekta, bez da se previše opterećuje Raspberry Pi.

Ukratko, Python je popularan izbor za programiranje Discord bota na Raspberry Pi-u zbog njegove jednostavnosti, velike zajednice, dostupnih biblioteka, učinkovitosti i skalabilnosti. To ga čini dobrim izborom za programere koji žele stvoriti bot s bogatim funkcionalnostima na platformi Raspberry Pi [2].

2.2. Raspberry Pi

Raspberry Pi je jeftino računalo veličine kreditne kartice koje se priključuje na računalni monitor ili TV na Raspberry Pi se mogu priključiti standardna tipkovnica i miš, koje je razvila Zaklada Raspberry Pi u Ujedinjenom Kraljevstvu. Sa svojom niskom potrošnjom energije i podrškom za širok raspon prilagodbi hardvera i softvera, računalo je popularno među programerima, hobbistima i studentima [3].

Raspberry Pi može se koristiti u različite svrhe, uključujući izgradnju sustava kućne automatizacije, medijskih centara, igračih konzola i više. Ovo računalo ima mnoštvo GPIO (ulaz/izlaz opće namjene) pinova koji se mogu spojiti na razne senzore, motore, LED svjetla i druge elektroničke komponente.

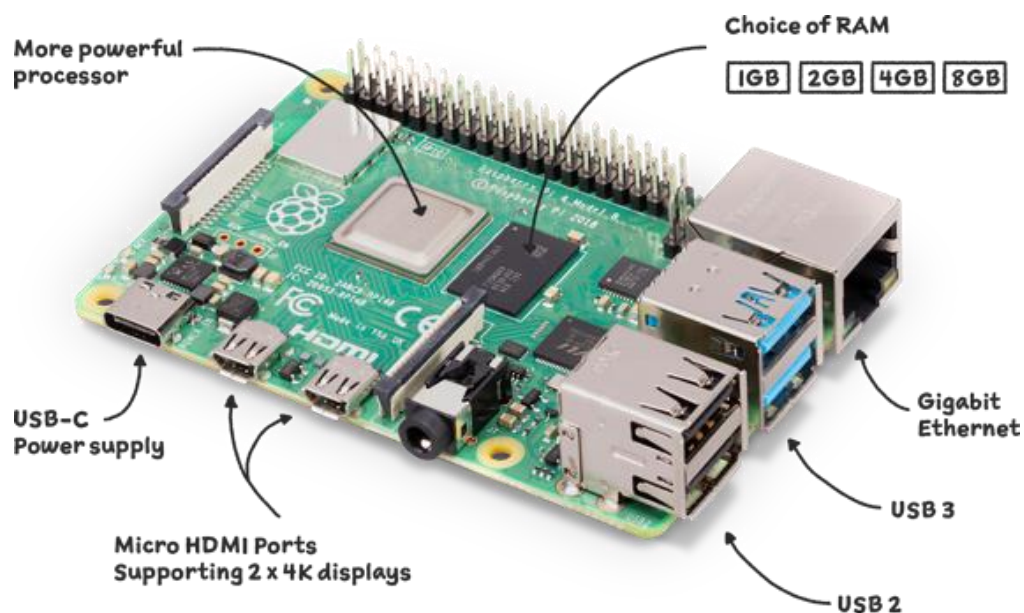
Raspberry Pi ima operativni sustav temeljen na Linuxu, te omogućava programiranje u raznim programskim jezicima, uključujući Python, C++, Java i druge. Ovo računalo također ima mnogo korisnih alata za razvoj softvera, uključujući uređivače teksta, grafička sučelja, terminale i još mnogo toga.

Zbog male veličine, niske potrošnje energije i prilagodljivosti, Raspberry Pi je postao popularan alat u različitim područjima, uključujući obrazovanje, istraživanje te razvoj softvera. Također, može se koristiti Raspberry Pi kako bi se napravilo malo IoT (Internet of Things) rješenje koje će kontrolirati pametne uređaje. Na primjer, može se

koristiti Raspberry Pi tako da se spoji pametni termostat ili rasvjeta i zatim se koristi Python programiranje kako bi se upravljalo uređajima udaljeno, čak ako korisnik i nije kod kuće. U kombinaciji s Pythonom, Raspberry Pi se često koristi za izradu različitih projekata, uključujući bote za Discord, robotske projekte, nadzorne kamere i mnogo više.

Raspberry Pi 4 Model B je jedan od najnovijih modela Raspberry Pi računala i dolazi s brojnim poboljšanjima u odnosu na prethodne modele. Ovdje su opće specifikacije [5]:

- Procesor: Broadcom BCM2711, četverojezgrena ARM Cortex-A72 procesor sa taktom od 1.5GHz
- RAM: 2GB, 4GB ili 8GB LPDDR4-3200 SDRAM (ovisno o verziji)
- Pohrana: MicroSD (do 128 GB)
- Mrežno sučelje: Gigabit Ethernet
- Bežična mreža: 2.4GHz i 5GHz IEEE 802.11.b/g/n/ac wireless, Bluetooth 5.0, BLE
- Video izlaz: 2 × mikroHDMI portovi
- Audio: 3,5 mm audio izlazni priključak, 2 × mikrofonski priključak
- USB priključci: 2 × USB 3.0 priključak, 2 × USB 2.0 priključak
- GPIO: Standardni 40-pinski GPIO header (sa 28 GPIO pinova, 2 × SPI, 2 × I2C, serijski, PWM, žičana Ethernet, 3,3 V, 5 V, i GND pinovi)
- Napajanje: 5V DC via USB-C priključak (min. 3A), ili GPIO header (min. 3A)
- Radna temperatura: 0-50°C
- Dimenzije: 88 x 58 x 19.5 mm, 46 g



Slika 1: Raspberry Pi 4 Model B

Pregled svih modela i verzija Raspberry Pi računala te datumi izlaska [3]:

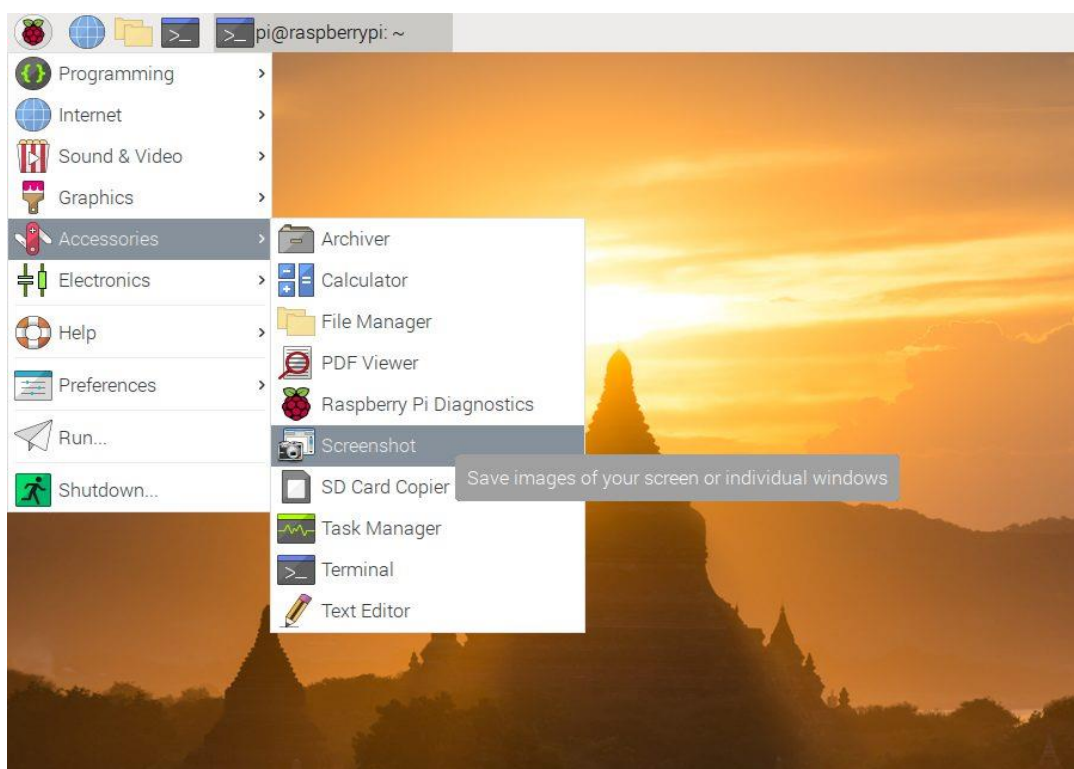
- Raspberry Pi Model B - veljača 2012.
- Raspberry Pi Model A - veljača 2013.
- Raspberry Pi Model B+ - srpanj 2014.
- Raspberry Pi 2 Model B - veljača 2015.
- Raspberry Pi Zero - studeni 2015.
- Raspberry Pi 3 Model B - veljača 2016.
- Raspberry Pi Zero W - veljača 2017.
- Raspberry Pi 3 Model B+ - ožujak 2018.
- Raspberry Pi 3 Model A+ - studeni 2018.
- Raspberry Pi 4 Model B - lipanj 2019.
- Raspberry Pi 400 - studeni 2020.
- Raspberry Pi Pico - siječanj 2021.

Svaki model i verzija Raspberry Pi računala ima svoje jedinstvene značajke i specifikacije, što ih čini pogodnima za različite vrste projekata i aplikacija. Ovaj završni rad napravljen na Raspberry Pi 4 Model B (slika 1) [4].

2.2.1. Raspbian operativni sustav

Raspbian je operativni sustav baziran na Debianu, optimiziran za Raspberry Pi računala [6]. Raspbian se preporučuje kao operativni sustav za Raspberry Pi jer je optimiziran za ovo računalo i uključuje sve potrebne alate i aplikacije za programiranje, upravljanje datotekama itd.

Raspbian dolazi u dvije verzije - Raspbian i Raspbian Lite. Raspbian izdanje uključuje desktop okruženje, grafičko sučelje, uređivač teksta i mnoge druge aplikacije koje olakšavaju rad s Raspberry Pi. Raspbian Lite izdanje je skraćena verzija Raspbiana bez desktop okruženja i aplikacija. Ovo izdanje je idealno za programere i napredne korisnike koji žele izgraditi prilagođene sustave. Radna površina Raspbiana operacijskog sustava je prikazan na slici 2.

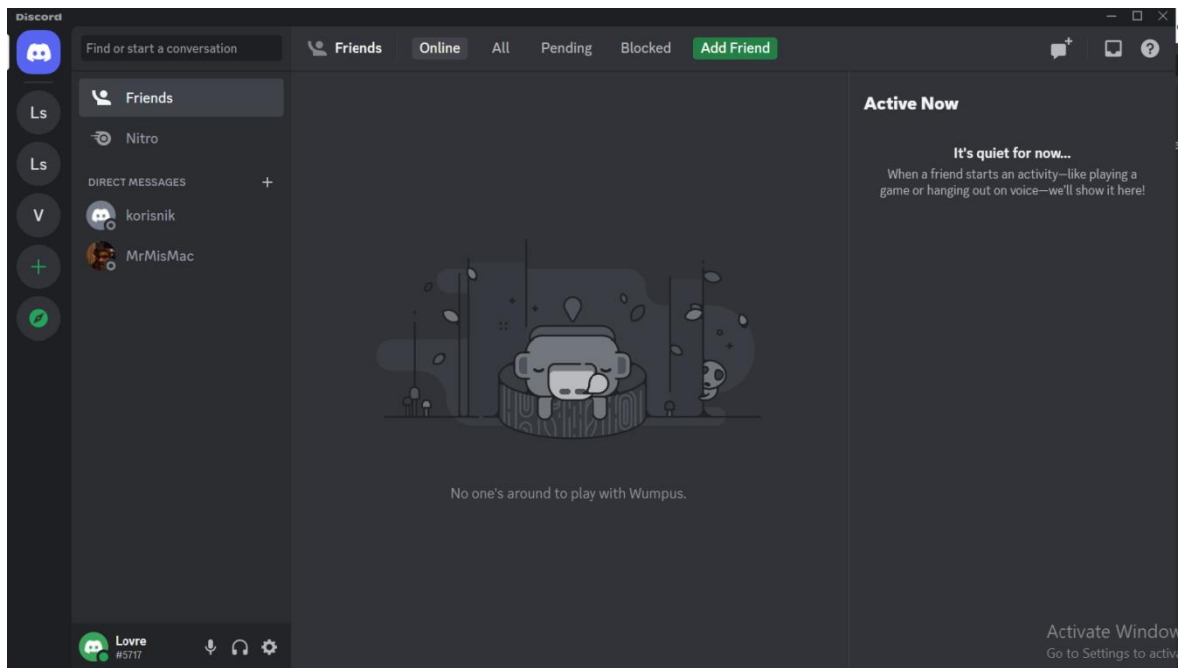


Slika 2: Radna površina Raspbian operacijskog sustava

2.3. Discord

Discord je besplatna platforma za glasovnu i tekstualnu komunikaciju popularna među igračima, ali se sve više koristi i u drugim zajednicama. Discord omogućuje korisnicima stvaranje i sudjelovanje na različitim poslužiteljima, koji se mogu prilagoditi prema potrebama korisnika, uključujući različite kanale za *chat*, integraciju s drugim aplikacijama, botove i još mnogo toga [7].

Prvo, naravno, potrebno je imati Discord korisnički račun. Discord poslužitelju mogu se postaviti razne postavke od privatnosti do upravljanja poslužiteljima te davanja različitih uloga korisnicima. Discord *chat* je centralni element Discord platforme, koristi se za komunikaciju među korisnicima i poslužiteljima. Korisnici mogu razgovarati u javnim ili privatnim kanalima, ovisno o postavkama poslužitelja i dozvolama koje su im dodijeljene. Javni kanali dostupni su svim korisnicima na poslužitelju, a mogu se podijeliti u kategorije kako bi se organizirali na tematski ili funkcionalni način. Korisnici mogu slati tekstualne poruke, slike, videa, emoji, gifove i druge vrste datoteka. Privatni kanali su dostupni samo odabranim korisnicima i korisni su za privatne ili grupne razgovore. Korisnici mogu stvoriti vlastite privatne kanale ili se pridružiti postojećim privatnim kanalima. Uz osnovne mogućnosti slanja poruka, Discord *chat* ima i dodatne funkcije kao što su označavanje korisnika (@mentions), označavanje važnih poruka (prikvačene poruke), mogućnost uređivanja i brisanja poruka te integraciju s drugim alatima poput botova i drugih aplikacija. U ovom radu istražiti će se proces stvaranja Discord bota pomoću programskog jezika Python i njegove integracije s Raspberry Pi. Pogledat će se različite funkcije koje se mogu uključiti u bota i prednosti korištenja Raspberry Pi za pokretanje bota. Također, razmotrit će se izazovi razvoja Discord bota i pružiti rješenja za uobičajene probleme. Korisnici Discord poslužitelja mogu međusobno komunicirati s botovima preko naredbi koje se obično pokreću korištenjem prefiksa. Primjerice, bot koji upravlja rasporedom događaja može reagirati na naredbu `!schedule` i prikazati popis nadolazećih događaja, dok bot za reprodukciju glazbe može reagirati na naredbe kao što su `!play` ili `!skip`. Na slici 3 prikazan je izgled Discord aplikacije.



Slika 3: Izgled Discord aplikacije

Nakon što se korisnik registrira na Discordu, može pretraživati različite poslužitelje i zajednice prema svojim interesima i pridružiti se njima. Ulogirani korisnik može pristupiti svojim poslužiteljima na lijevoj strani sučelja, gdje su prikazane ikone poslužitelja. Klikom na ikonu poslužitelja ulazi se na poslužitelj, a prelaskom miša preko slike poslužitelja postaje vidljivo ime poslužitelja. U gornjem lijevom kutu piše ime poslužitelja na koji je korisnik prijavljen. Klikom na padajući izbornik dobiju se opcije za postavke poslužitelja. Korisnik može poslati poruku na kanal klikom na polje za unos teksta na dnu sučelja. Dodatno, na vrhu sučelja prikazane su ikone za prijatelje (friends), obavijesti (notifications) i poruke (inbox) korisničkog računa. Prijatelji su prikazani s desne strane sučelja, gdje se prikazuje popis online prijatelja.

2.3.1. Botovi

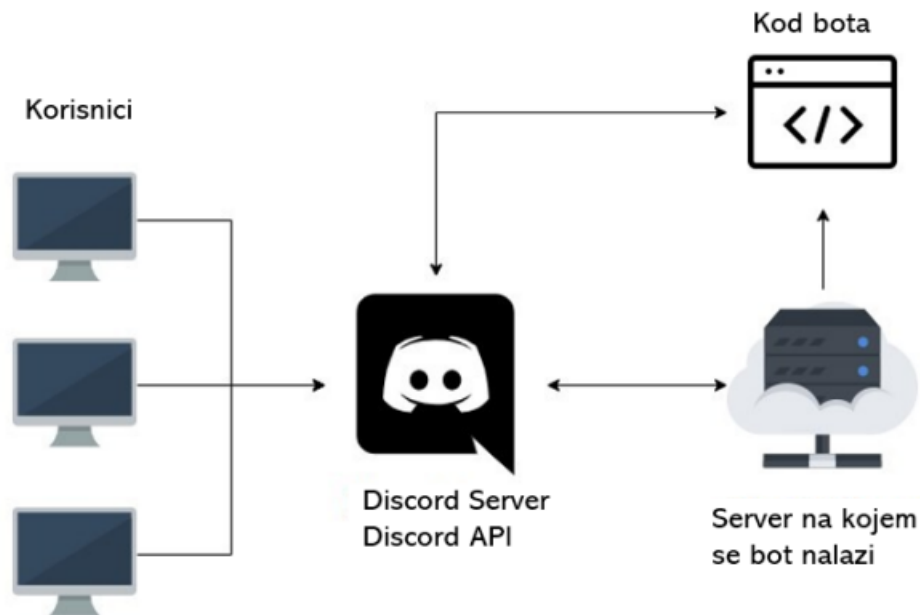
Discord bot je programski kôd koji omogućuje automatizaciju i pružanje dodatnih funkcionalnosti na Discord platformi. Koristeći Discord API, botovi se mogu integrirati u poslužitelja i kanale kako bi izvršavali različite zadatke i interakcije s korisnicima. Discord botovi se mogu programirati u više programskih jezika, kao što su Python, JavaScript, Ruby i mnogi drugi, ali Python je jedan od najpopularnijih jezika za izradu Discord botova zbog svoje jednostavnosti i lakoće korištenja.

Discord botovi mogu izvršavati različite zadatke, kao što su:

- Automatsko dobrodošlice novim korisnicima
- Praćenje aktivnosti korisnika na poslužitelju
- Upravljanje ulogama i dozvolama korisnika
- Automatsko postavljanje rasporeda događaja i obavještanje korisnika
- Reprodukcija glazbe na zahtjev korisnika
- Interakcija s API-jima drugih usluga, poput vremenskih prognoza, pretraživanja interneta i drugih.

Botovi mogu koristiti različite vrste interakcija s korisnicima, kao što su naredbe koje se unose u *chat*, reakcije na određene poruke i događaje, te različite vrste obavijesti i poruka. Botovi, također mogu koristiti umjetnu inteligenciju kako bi postali sve sofisticiraniji u svojim funkcijama i interakcijama s korisnicima. Za izradu Discord botova koristi se Discord API koji pruža različite metode i funkcije za interakciju s Discord platformom. Discord botovi također zahtijevaju „Oauth“ autorizaciju za integraciju s Discord poslužiteljima, te je stoga važno osigurati pravilnu konfiguraciju i zaštitu botova kako bi se izbjegle zloupotrebe i sigurnosni problemi.

Kako bi se bolje razumjela komunikacija korisnik-bot, može se zamisliti komunikaciju kao dijagram što je vidljivo na slici 4. Korisnici Discorda na poslužitelju s jedne strane, a sam Discord u sredini s API-jem i poslužiteljom, s druge strane, botovi i njihov kôd. Korisnici komuniciraju kanalima i šalju poruke, bot s njima komunicira putem API-ja, razlikuje poruke koje su mu poslone i šalje ih prikladno, unaprijed programirano.



Slika 4: Dijagram komunikacije između korisnika i bota

2.3.2. API

Discord API je programsko sučelje koje omogućuje developerima stvaranje aplikacija koje mogu komunicirati s Discord platformom. Ovaj API omogućuje programerima pristup različitim funkcijama Discord platforme, kao što su upravljanje korisnicima, upravljanje porukama, upravljanje kanalima, upravljanje poslužiteljima, upravljanje botovima i drugim funkcijama. Discord API je RESTful API koji koristi HTTPS protokol za komunikaciju između klijenata i poslužitelja [8]. Podržava različite HTTP metode, kao što su GET, POST, PUT i DELETE. Kako bi se koristio Discord API, programeri moraju prvo registrirati svoju aplikaciju na *Discord Developers Portalu* i dobiti pristupni token. Korisnici Discord API-a mogu kreirati različite vrste aplikacija, uključujući *chat* botove, igraće botove, muzičke botove, statističke alate i mnoge druge. Discord API je postao popularan među developerima zbog svoje jednostavnosti korištenja i bogatog skupa funkcija koje nudi.

3. Realizacija zadatka

3.1. Instalacija Raspebrry Pi-a

Discord je popularna platforma za razmjenu poruka i glasovni *chat* koju koriste milijuni ljudi diljem svijeta. Uz pomoć programskog jezika Python i Raspberry Pi, korisnici mogu kreirati vlastite Discord botove za obavljanje raznih zadataka.

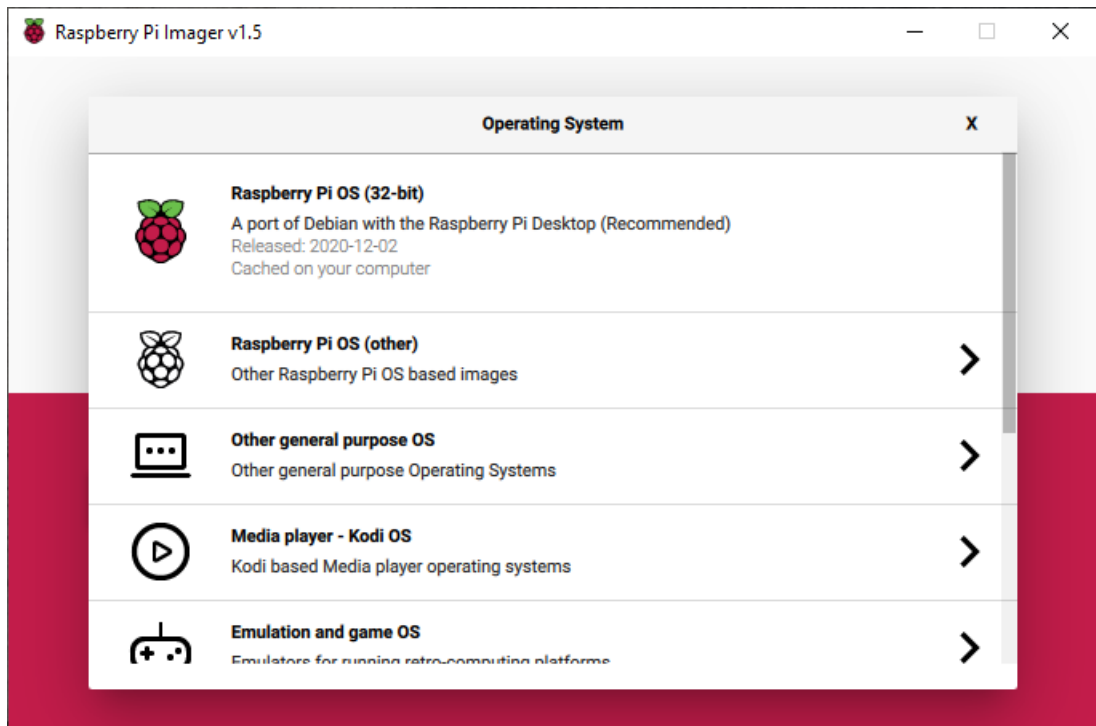
Kako bi se počeo koristiti Raspberry Pi, operacijski sustav mora biti instaliran na Raspberry Piu. Operacijski sustav koji će se koristiti za ovaj rad je Raspbian, službeni Raspberry Pi operacijski sustav.

Kod instalacije Raspbiana prvi korak je preuzeti najnoviju verziju sa službene internetske stranice Raspberry Pia. Preporučljivo je preuzeti verziju s grafičkim korisničkim sučeljem (desktop verzija) kako bi se mogao koristiti Raspberry Pi kao obično računalo. Opcije pri instalaciji Raspbian operacijskog sustava vide se na slici 5.



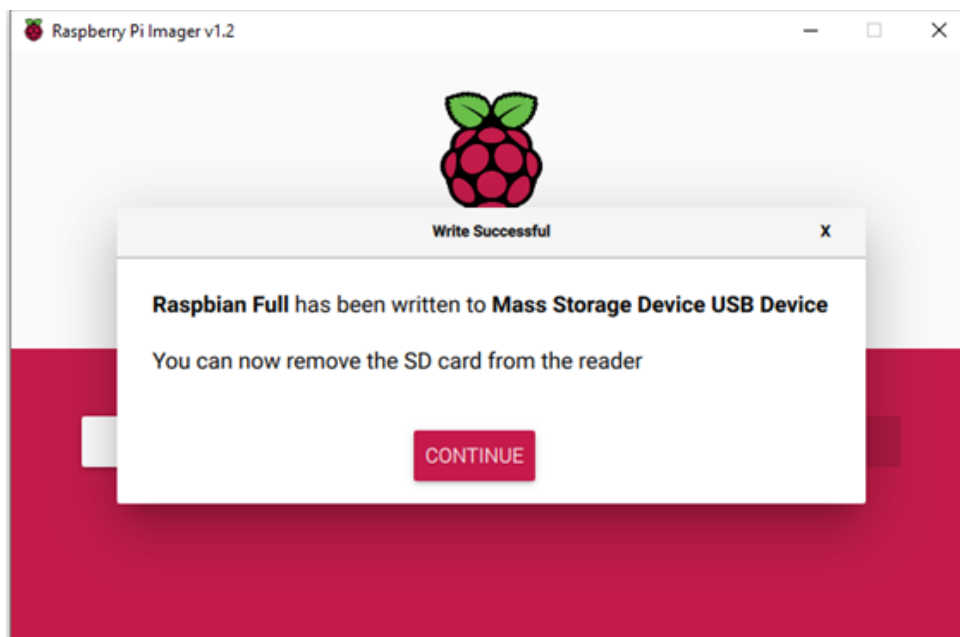
Slika 5: Instalacija Raspbian operacijskog sustava

Nakon preuzimanja Raspbiana, potrebno je formatirati MicroSD karticu. Preporučljivo je koristiti MicroSD karticu veličine najmanje 8 GB kako bi se osiguralo da postoji dovoljno prostora za instalaciju operacijskog sustava i aplikacija. Umetne se MicroSD karticu u čitač kartice računala. Potom se otvori *Raspberry Pi imager* alat i korisnik izabere opciju „Choose OS“. Izabere se Raspbian iz liste operativnih sustava koje nudi Raspberry Pi Imager, vidljive na slici 6.



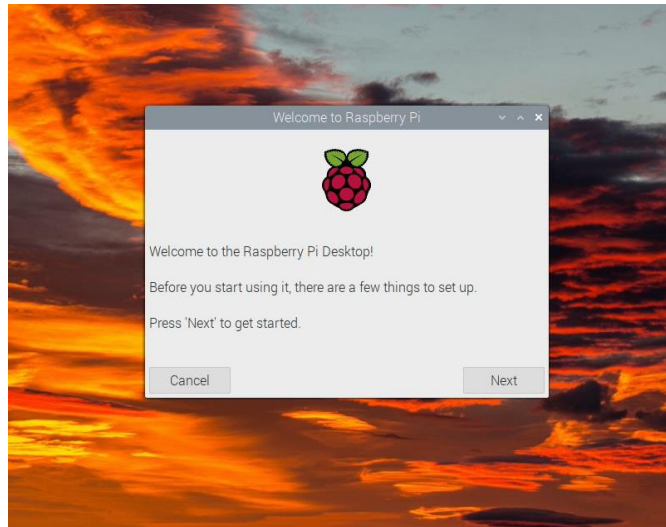
Slika 6: Odabir Raspbian operacijskog sustava

Odabere se MicroSD kartica na koju se želi instalirati operativni sustav. Klikom na "Write" započinje proces snimanja slike na karticu. Kada je proces snimanja završen dobije se poruka vidljiva na slici 7, izvadi se kartica iz računala i umetne se u Raspberry Pi računalo.



Slika 7: Instalacija Raspbian operacijskog sustava

Uključi se Raspberry Pi računalo i sačeka da se Raspbian operativni sustav pokrene. Prvi put kada se pokrene, bit će ponuđen "Setup Wizard" koji će korisnika voditi kroz nekoliko koraka za podešavanje osnovnih parametara sustava, kao što su podešavanje jezika, vremenske zone, mrežnih postavki i slično. Nakon što se završi podešavanje osnovnih parametara sustava, može se početi koristiti Raspbian na Raspberry Pi računalu. Raspberry Pi se poveže na monitor uz pomoć HDMI kabela te se na Raspberry Pi priključi miš i tipkovnicu kako bi se omogućio pristup operacijskom sustavu, prikazano na slici 8.



Slika 8: Postavljanje postavki za Raspbian OS

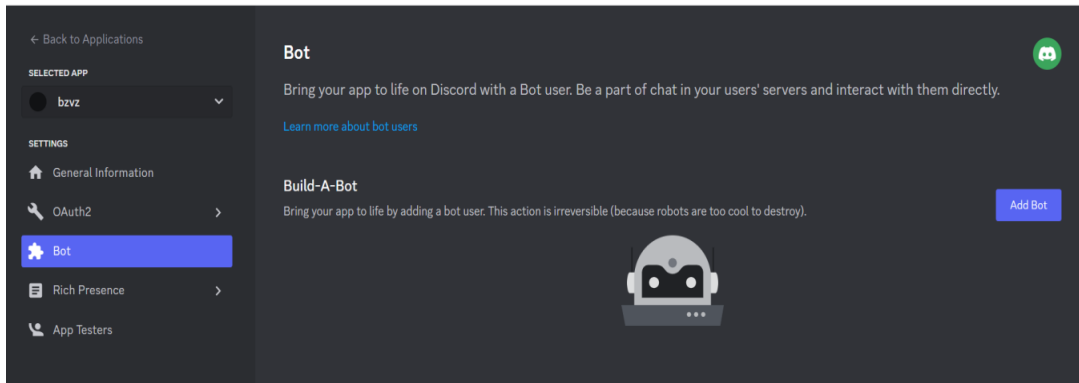
3.2. Instalacija Discord bota

Registracija na *Discord Developers Portal* je besplatna i omogućuje stvaranje novog bota. Kada se registrira, novi bota se stvori tako što se klikne na „New Application“ i unese naziv aplikacije za bota u ovom slučaju „RPIbot“ i klikne se „Create“ (slika 9)

A screenshot of the "CREATE AN APPLICATION" form on the Discord Developers Portal. The form has a dark background. At the top, it says "CREATE AN APPLICATION". Below that is a yellow box with the text: "Are you a game dev? We may already have your app in our database. Reach out to our **Dev Support** for more info and to claim your game!". Below the yellow box is a "NAME" field with a red asterisk, containing the text "RPIBot". Below the name field is a checked checkbox with the text: "By clicking Create, you agree to the Discord [Developer Terms of Service](#) and [Developer Policy](#)". At the bottom right, there are "Cancel" and "Create" buttons.

Slika 9: Kreiranje aplikacije za Discord bot

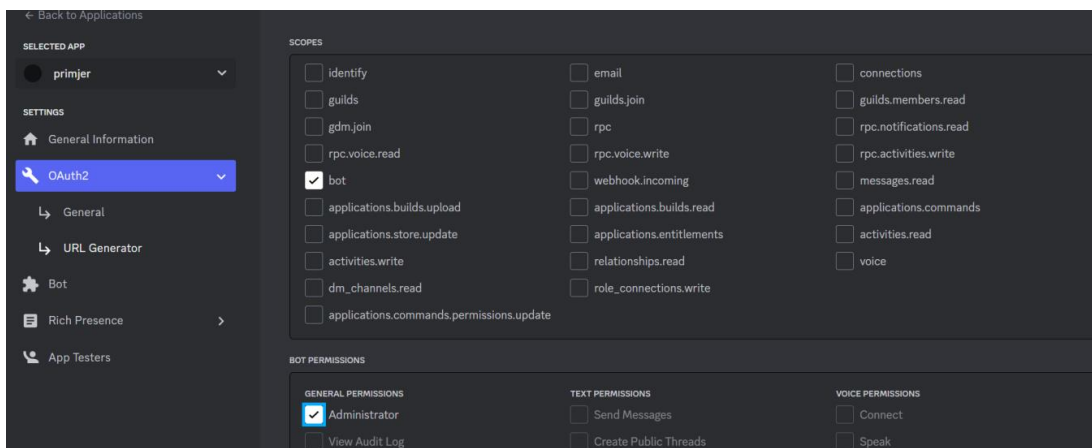
Korak dva je stvaranje bot računa, klikne se na „Bot“ u lijevom izborniku i zatim „Add Bot“ (slika 10). Svoj odabir potvrdimo klikom na „Yes, do it!“.



Slika 10: Stvaranje bot računa

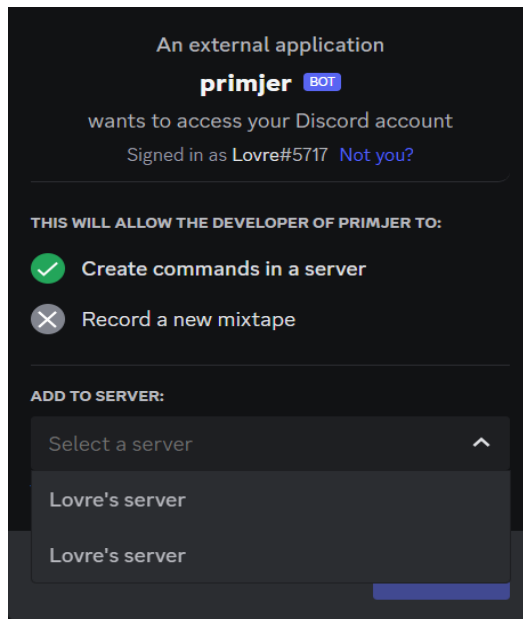
Prema potrebi može se prilagoditi ime i slika profila bota.

Kako bi dodao bot na Discord poslužitelj klikne se na „OAuth2“ u lijevom izborniku. Odabere su potrebne ovlasti koje se žele dodijeliti botu. U ovom primjeru, dodaju se „bot“ ovlasti i „administratorske“. Prikazano na slici 11.



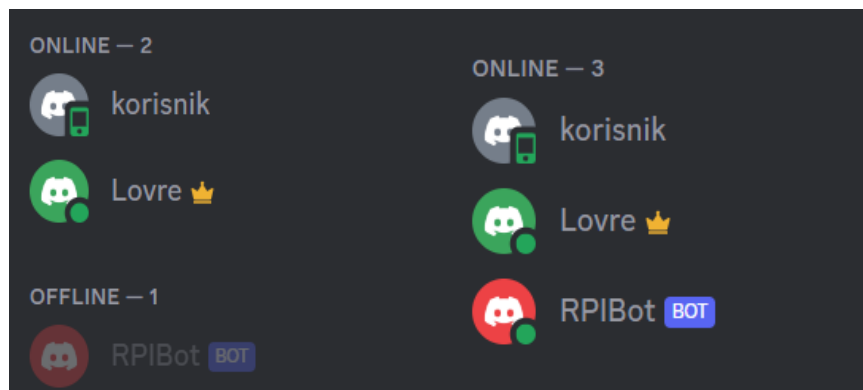
Slika 11: Dodavanje ovlasti botu

Kopira se generirani URL i otvori se u novoj kartici preglednika. Odabere se poslužitelj na koji se želi dodati bot i klikne se „Authorize“ (slika 12).



Slika 12: Izgled pozivnice za dodavanje bota na poslužitelj

Nakon toga, bot bi trebao biti aktivan i spreman za rad. Prikaz bota iz neaktivnog u aktivno stanje (slika 13).



Slika 13: Prikaz bota iz neaktivnog u aktivno stanje

3.3. Program za filtriranje sadržaja

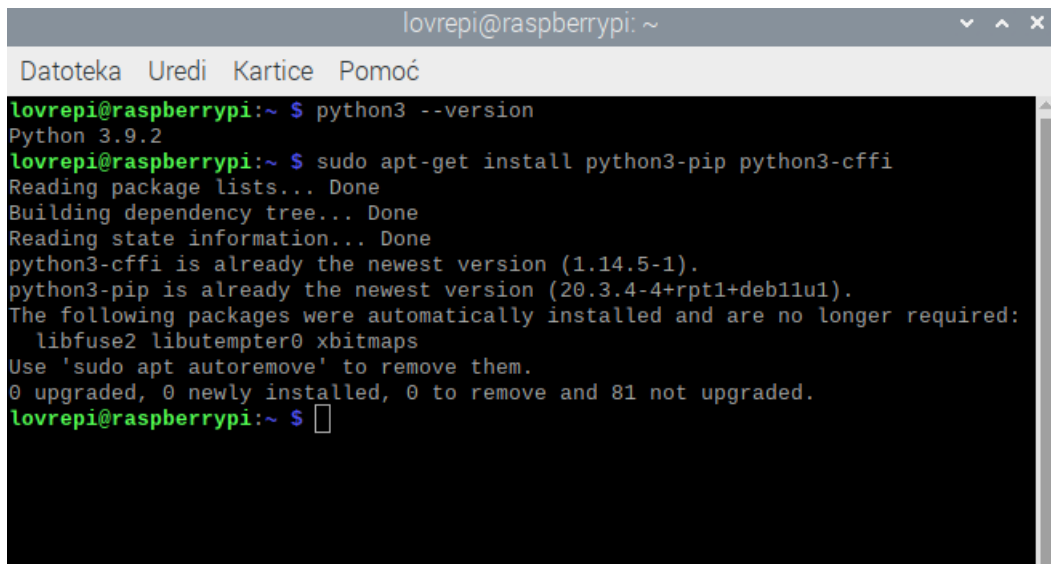
Nakon instalacije Discorda i pokretanja Raspberry Pi-a otvara se terminal gdje se instalira programski jezik Python. Instalacija Pythona na Raspberry Pi omogućuje korisnicima da programiraju u Pythonu direktno na svom Raspberry Pi računalu. Postoje različite verzije Pythona koje se mogu instalirati na Raspberry Pi, ovisno o verziji operativnog sustava koji se koristi.

Postupak instalacije Pythona na Raspberry Pi je sljedeći:

Provjeri se koju verziju Pythona korisnik već ima instaliranu na Raspberry Pi računalu. To se može provjeriti otvaranjem terminala i upisivanjem naredbe `python --version`. Ako korisnik već ima instaliranu verziju Pythona, možda nećete trebati instalirati novu verziju. Ako ipak želi instalirati novu verziju Pythona, prvo se ažurira paketna lista operativnog sustava pomoću naredbe `sudo apt-get update`. Nakon ažuriranja paketnih listi, upiše se sljedeća naredba za instalaciju Pythona 3: `sudo apt-get install python3-pip python3-cffi`.

Ova naredba instalira najnoviju verziju Pythona 3 paketa `pip` i `cffi` u Raspberry Pi operacijskom sustavu. `pip` je paketni upravitelj za Python koji olakšava instaliranje i upravljanje Python paketima i njihovim ovisnostima. `cffi` je Python biblioteka koja omogućuje pisanje i uporabu C ekstenzija u Python kôdu. Ova biblioteka se može koristiti za ubrzanje Python kôda ili za interakciju s bibliotekama napisanim u C-u. Ove naredbe potrebne su za instalaciju Discord Python API paketa i drugih Python paketa koji se mogu koristiti za razvoj Discord botova ili drugih aplikacija koje koriste Python.

Nakon što se Python 3 uspješno instalira, može se provjeriti koju verziju Pythona korisnik ima instaliranu pomoću naredbe `python3 --version`, vidljivo na slici 14. Ako se želi instalirati neka druga verzija Pythona, to se može učiniti ručnim preuzimanjem i instalacijom datoteke za instalaciju sa službene Pythonove *web* stranice.

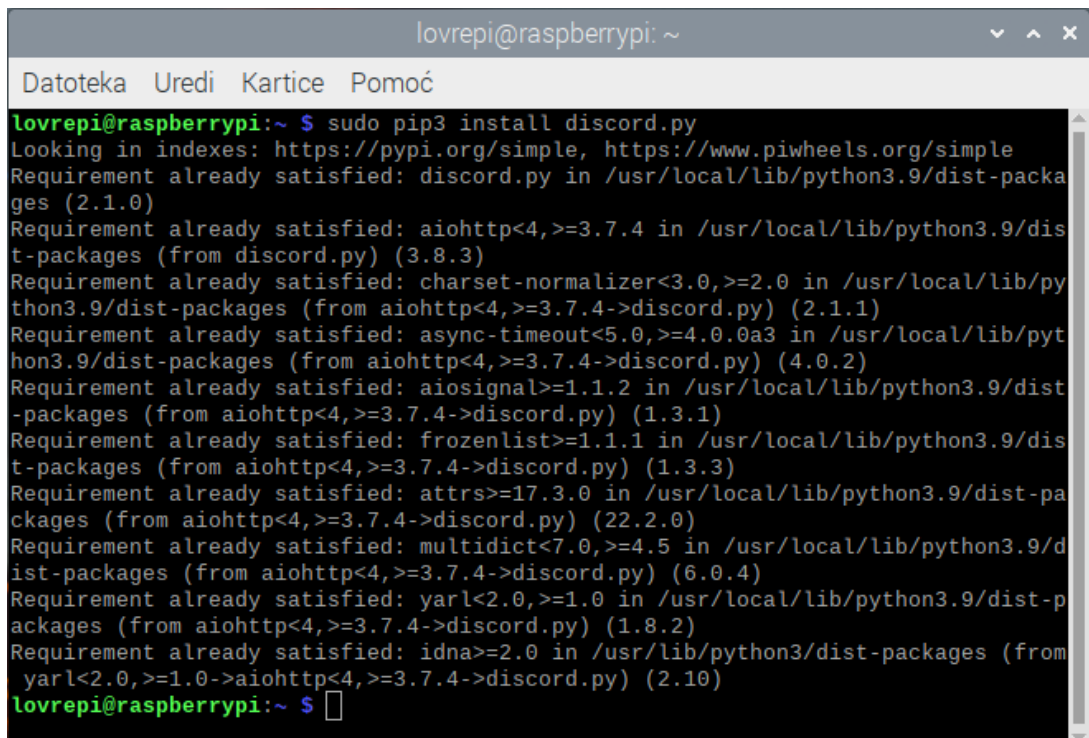


```
lovrepi@raspberrypi: ~  
Datoteka Uredi Kartice Pomoć  
lovrepi@raspberrypi:~ $ python3 --version  
Python 3.9.2  
lovrepi@raspberrypi:~ $ sudo apt-get install python3-pip python3-cffi  
Reading package lists... Done  
Building dependency tree... Done  
Reading state information... Done  
python3-cffi is already the newest version (1.14.5-1).  
python3-pip is already the newest version (20.3.4-4+rpt1+deb11u1).  
The following packages were automatically installed and are no longer required:  
  libfuse2 libutempter0 xbitmaps  
Use 'sudo apt autoremove' to remove them.  
0 upgraded, 0 newly installed, 0 to remove and 81 not upgraded.  
lovrepi@raspberrypi:~ $
```

Slika 14: Prikaze naredbe za provjeru Python verzije

Nakon instalacije Pythona, mogu se koristiti razne biblioteke za razvoj aplikacija i projekata na Raspberry Pi računalu. Također, može se koristiti Python za razvoj aplikacija koje se mogu koristiti u suradnji sa Discordom, kao što je razvoj botova za Discord.

Kako bi se instalirala Discord biblioteku za Python, koristit se naredba `pip` u terminalu. Otvori se terminal i unese sljedeća naredba: `sudo pip3 install discord.py`, prikazano na slici 15. Ova naredba će automatski preuzeti i instalirati biblioteku `discord.py` u Python okruženju na Raspberry Pi. Nakon što je biblioteka instalirana, može se početi koristiti `discord.py` za projekte koji koriste Discord API.



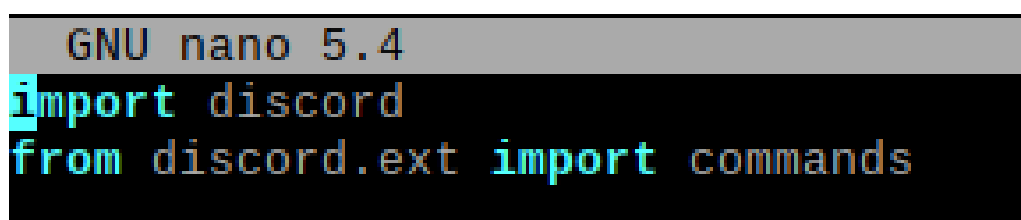
```
lovrepi@raspberrypi: ~  
Datoteka Uredi Kartice Pomoć  
lovrepi@raspberrypi:~ $ sudo pip3 install discord.py  
Looking in indexes: https://pypi.org/simple, https://www.piwheels.org/simple  
Requirement already satisfied: discord.py in /usr/local/lib/python3.9/dist-packa  
ges (2.1.0)  
Requirement already satisfied: aiohttp<4,>=3.7.4 in /usr/local/lib/python3.9/dis  
t-packages (from discord.py) (3.8.3)  
Requirement already satisfied: charset-normalizer<3.0,>=2.0 in /usr/local/lib/py  
thon3.9/dist-packages (from aiohttp<4,>=3.7.4->discord.py) (2.1.1)  
Requirement already satisfied: async-timeout<5.0,>=4.0.0a3 in /usr/local/lib/pyt  
hon3.9/dist-packages (from aiohttp<4,>=3.7.4->discord.py) (4.0.2)  
Requirement already satisfied: aiosignal>=1.1.2 in /usr/local/lib/python3.9/dist  
-packages (from aiohttp<4,>=3.7.4->discord.py) (1.3.1)  
Requirement already satisfied: frozenlist>=1.1.1 in /usr/local/lib/python3.9/dis  
t-packages (from aiohttp<4,>=3.7.4->discord.py) (1.3.3)  
Requirement already satisfied: attrs>=17.3.0 in /usr/local/lib/python3.9/dist-pa  
ckages (from aiohttp<4,>=3.7.4->discord.py) (22.2.0)  
Requirement already satisfied: multidict<7.0,>=4.5 in /usr/local/lib/python3.9/d  
ist-packages (from aiohttp<4,>=3.7.4->discord.py) (6.0.4)  
Requirement already satisfied: yarl<2.0,>=1.0 in /usr/local/lib/python3.9/dist-p  
ackages (from aiohttp<4,>=3.7.4->discord.py) (1.8.2)  
Requirement already satisfied: idna>=2.0 in /usr/lib/python3/dist-packages (from  
yarl<2.0,>=1.0->aiohttp<4,>=3.7.4->discord.py) (2.10)  
lovrepi@raspberrypi:~ $
```

Slika 15: Prikaze naredbe za instalaciju Discord biblioteke

Koristi se Nano editor za pisanje kôda za filtriranje poruka. Nano editor se može pokrenuti pomoću naredbe `nano ime_datoteke` gdje `ime_datoteke` predstavlja ime datoteke koja se želi urediti. Nakon što se datoteka otvori u Nano editoru, upisuju se naredbe za filtriranje sadržaja.

Sljedeći kôd je primjer Discord bota koji može filtrirati neprikladan sadržaj iz poruka i zabraniti rad korisnicima koji više puta prekše pravila.

Python program koristi biblioteku `discord` i Discord API kako bi implementirao jednostavan *chat* bot za Discord poslužitelj. Prvo se uvoze potrebni moduli `discord` i `commands`. Prikazano na slici 16.



```
GNU nano 5.4  
import discord  
from discord.ext import commands
```

Slika 16: Implementacija potrebnih modula

Zatim se stvaraju objekti `intents` i `bot`. Objekt `intents` definira koja će događanja bot oslušivati, a u ovom slučaju uključeni su svi događaji. Objekt `bot` predstavlja samog bota i specificira se prefiks naredbi (u ovom slučaju `"`) i `intents`, prikazano na slici 17.

```
intents=discord.Intents.default()
intents.message_content=True
bot = commands.Bot(command_prefix=' ', intents=intents)
```

Slika 17: Stvaranje potrebnih objekata

Sljedeće se stvara rječnik `brojac_rijeci` koji će se koristiti za brojanje broja neprimjerenih riječi koje korisnici napišu.

Zatim slijedi `@bot.event` ova funkcija obrađuje naredbe koje su registrirane za bota i druge grupe. Bez toga nijedna naredba neće biti pokrenuta. Ova se funkcija poziva kad novi korisnik pristupi poslužitelju. Ako korisnik već postoji u rječniku `brojac_rijeci`, briše se taj unos iz rječnika. U Pythonu, `async def` definira asinkronu funkciju, tj. funkciju koja se izvodi asinkrono, tj. ne blokira izvođenje drugog kôda tijekom čekanja na određenu operaciju. Primjer kôda je vidljiv na slici 18.

```
@bot.event
async def on_member_join(member):
    if member.author.id in brojac_rijeci:
        del brojac_rijeci[member.author.id]
```

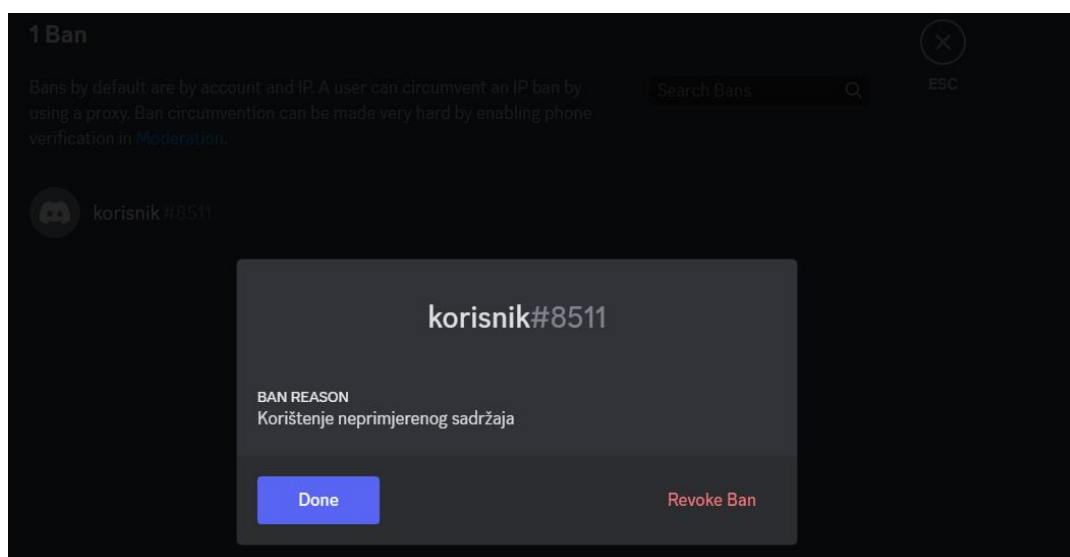
Slika 18: Funkcija za provjeru novih korisnika

Slijedi drugi `@bot.event` i funkcija `on_message()`. Ova se funkcija pokreće kad god bot primi poruku na poslužitelju, a prvo provjerava da li je poruka poslana od strane drugog bota. Ako je poruka poslana sa strane bota, funkcija se zaustavlja. U suprotnom, funkcija obrađuje poruku i provjerava sadrži li neprimjerene riječi. Primjer kôda je prikazan na slici 19.

```
@bot.event
async def on_message(message):
    if message.author.bot:
        return
```

Slika 19: Funkcija za provjeru pošiljatelja poruke

Ako poruka sadrži neprimjerene riječi, bot briše poruku i šalje poruku u kanal „Neprimjereni sadržaj i navodi autora“ i broji koliko puta je korisnik koristio neprimjerene riječi. Ako je korisnik tri puta koristio neprimjerene riječi, bot će ga banati s porukom "Korištenje neprimjerenog sadržaja", što je vidljivo na slici 20. Ako je korisnik dvaput upozoren, bot će ga upozoriti da će sljedeći put dobiti ban.



Slika 20: Prikaz poruke nakon što korisnik dobije ban

Korištenje asinkronih funkcija u ovom kontekstu omogućuje istovremeno izvršavanje funkcija s drugim operacijama na sustavu bez blokiranja izvršavanja ostatka kôda. Također, ključna riječ `await` koristi se kako bi se omogućilo funkciji da privremeno zaustavi i vrati kontrolu petlji događaja kad god je potrebno, dok se obavljaju druge asinkrone operacije. Primjer naredbe je prikazan na slici 21.

```

text=message.content.lower()
bad_words="/home/lovrepi/bbadwords.txt"
filter_text=filter_stars(text,bad_words,message.author.id)
if filter_text:
    await message.delete()
    await message.channel.send(f"{filter_text}, Neprimjereni sadržaj korisnika {message.author}")

if message.author.id in brojac_rijeci:
    brojac_rijeci[message.author.id] += 1
else:
    brojac_rijeci[message.author.id] = 1

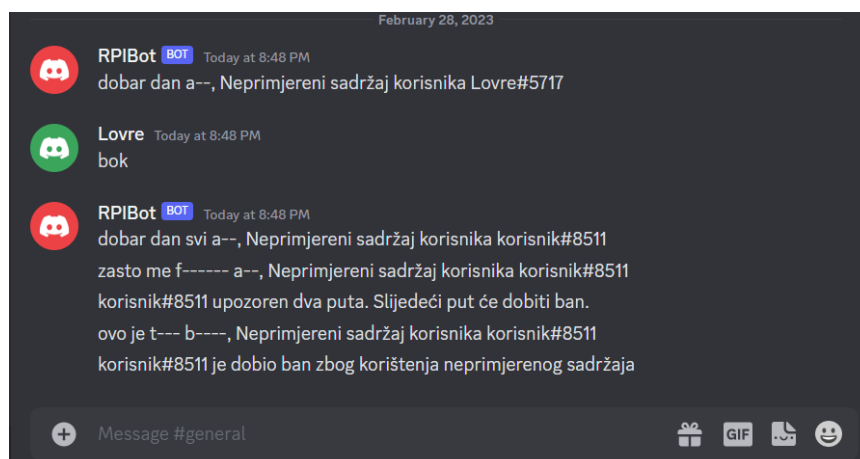
if brojac_rijeci[message.author.id] == 3:
    await message.author.ban(reason="Korištenje neprimjerenog sadržaja")
    await message.channel.send(f"{message.author} je dobio ban zbog korištenja neprimjerenog sadržaja")
    brojac_rijeci[message.author.id] = 0
elif brojac_rijeci[message.author.id] == 2:
    await message.channel.send(f"{message.author} upozoren dva puta. Slijedeći put će dobiti ban.")

```

Slika 21: Prikaz await naredbe

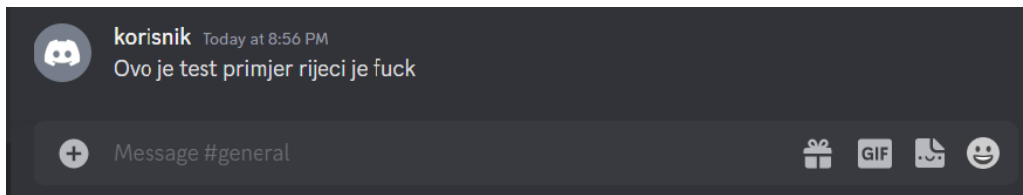
Funkcija `filter_stars` provjerava svaku riječ u primljenoj poruci i provjerava je li ta riječ sadržana u datoteci `bbadwords.txt` koja sadrži neprimjerene riječi. Ako je riječ u datoteci, zamjenjuje se prvih nekoliko slova minusima, ovisno o duljini riječi, a ako nije, dodaje se u popis `formatted_words`. Funkcija zatim vraća popis `formatted_words` kao niz, spajajući ga u jedan tekst.

Primjerice, ako se primi poruka "Zdravo svima, želimo vam lijep dan!", funkcija će se vratiti: "Zdravo svima, želim vam lijep dan!", jer nijedna riječ nije pronađena u datoteci `bbadwords.txt`. S druge strane, ako se primi poruku "Ovo je loša poruka s lošim riječima", funkcija će se vratiti: "Ovo je l--- poruka s l---- riječima", zamjenjujući prvih nekoliko slova svake nepristojne riječi minusima kao što se vidi na slici 22.



Slika 22: Primjer funkcioniranja Discord bota u aktivnom stanju

Za razliku kada je bot isključen, tada je dozvoljena upotreba neprimjerenog sadržaja, (slika 23).



Slika 23: Primjer funkcioniranja Discord bota u neaktivnom stanju

Funkcija prima tri argumenta:

- `text`: tekst poruke koju je potrebno provjeriti.
- `bad_words`: ime datoteke koja sadrži nepristojne riječi, a koju je potrebno provjeriti.
- `user_id`: ID korisnika koji šalje poruku.

Funkcija otvara datoteku `bbadwords.txt` i čita njezin sadržaj pomoću funkcije `open()`. Riječi u datoteci odvojene su zarezom i pohranjene su u jednom nizu. Funkcija prolazi kroz svaku riječ u tekstu i provjerava je li ta riječ u nizu `bad_words`. Nakon toga funkcija inicijalizira praznu listu `formatted_words` i postavlja boolean varijablu `formatted` na `False`. Lista `formatted_words` će se koristiti za spremanje filtriranog teksta, a `formatted` će se koristiti za provjeru jesu li pronađene loše riječi u tekstu. Ako je loša riječ u nizu, tada funkcija zamjenjuje prvih nekoliko slova s minusima i dodaje promijenjenu riječ u popis `formatted_words`. Ukoliko riječ nije u nizu, dodaje se u popis `formatted_words` u izvornom obliku. Funkcija zatim provjerava je li barem jedna riječ promijenjena, a ako jest, vraća promijenjeni tekst spajanjem riječi iz liste `formatted_words` pomoću metode `join()` i postavlja `formatted` na `True`. Ako nema promijenjenih riječi, vraća se `None`. Primjer kôda je prikazan na ispisu 1.

```

def filter_stars(text,bad_words,user_id):
    with open(bad_words,'r') as file:
        bad_words=file.read().split(',')
    formatted_words = []
    formatted = False
    for word in text.split():
        if word in bad_words:
            formatted_words.append(word[0] + '-' *
(len(word)-1))
            formatted=True
        else:
            formatted_words.append(word)
    return ' '.join(formatted_words) if formatted == True else
None
bot.run('MTA2MjAxNjIyNzYzMzAxMjc3Ng.Gv1He9.XD4Q58ItxgDhFeYC1IBI-
d0oIs526at842t2AA')

```

Ispis 1: Funkcija za filtriranje nepoželjnog sadržaja

I na kraju `bot.run(token)` je funkcija biblioteke `discord.py` koja se koristi za pokretanje Discord bota i povezivanje s Discord API-jem. Predstavlja tajni token osobnog bota koji se dobio prilikom stvaranja bota na *Discord Developer Portalu*. Tajni token bota se koristi za autentikaciju bota na Discord poslužitelju. Nikad ne bi trebalo dijeliti svoj tajni token s drugima jer bi to moglo dovesti do zlouporabe osobnog bota. Prilikom pozivanja funkcije `bot.run(token)`, bot će se povezati s Discord API-jem i bit će spreman za primanje događaja i obavljanje akcija na poslužitelju. Tokeni se koriste unutar kôda bota za slanje naredbi natrag i naprijed API-ju, koji zauzvrat kontrolira radnje bota.

3.4. Program za unos sadržaja

Ovaj kôd dodaje riječi koje korisnik unese u tekstualni dokument `bbadwords.txt` smješten na lokaciji `/home/lovrepi/`. Svaka nova riječ dodaje se i odvoja zarezom od ostalih riječi u dokumentu.

Na ispisu 2 je prikazan kôd, započinje definicijom funkcije `add_bad_word(word)` koja prima jedan argument `word`, što predstavlja riječ koju

želimo dodati u dokument. U funkciji, otvara se datoteka `bbadwords.txt` u načinu `a` (što znači da ćemo dodavati riječi u postojeći dokument) i koristi se `with` izraz kako bi se datoteka automatski zatvorila nakon završetka s pisanjem.

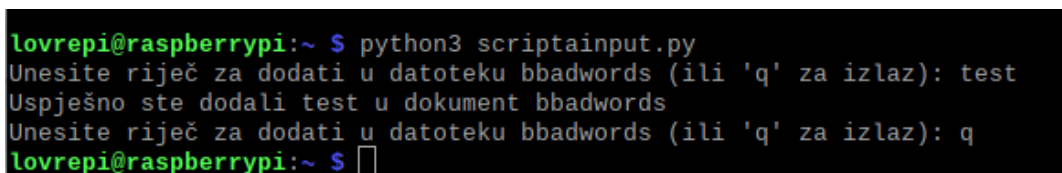
Zatim, se koristi `file.write()` metoda kako bi se dodala nova riječ na kraju dokumenta, odvojena zarezom od ostalih riječi. Nakon toga, ispisuje se poruka koja potvrđuje da je riječ uspješno dodana u dokument.

```
def add_bad_word(word):
    with open("/home/lovrepi/bbadwords.txt", "a") as file:
        file.write(',') + word
        print(f"Uspješno ste dodali {word} u dokument
bbadwords")

while True:
    new_word = input("Unesite riječ za dodati u datoteku
bbadwords (ili 'q' za izlaz): ")
    if new_word == 'q':
        break
    add_bad_word(new_word)
```

Ispis 2: Programski kôd za unos sadržaja

Nakon definiranja funkcije, kôd ulazi u beskonačnu petlju koja traži od korisnika da unese nove riječi. U slučaju da korisnik unese slovo "q", petlja se prekida i program završava. Svaka nova riječ koju korisnik unese prosljeđuje funkcijom `add_bad_word()` kako bi se dodala u dokument. Prikazano na slici 24.



```
lovrepi@raspberrypi:~ $ python3 scriptainput.py
Unesite riječ za dodati u datoteku bbadwords (ili 'q' za izlaz): test
Uspješno ste dodali test u dokument bbadwords
Unesite riječ za dodati u datoteku bbadwords (ili 'q' za izlaz): q
lovrepi@raspberrypi:~ $
```

Slika 24: Naredba za pokretanje programa za unos sadržaja

3.5. Start-up service za pokretanje Raspberry Pi-a

Start-up servisi su procesi koji se automatski pokreću prilikom pokretanja operativnog sustava. Postoji nekoliko prednosti upotrebe start-up servisa. Omogućavaju automatsko pokretanje vaših procesa nakon što se operativni sustav pokrene. To je korisno za procese koji se moraju pokrenuti uvijek kada se računalo ponovo pokrene ili kada se dogodi neki drugi događaj koji pokreće računalo. Pokretanje procesa putem start-up servisa može biti brže nego pokretanje procesa ručno. Start-up servisi obično se pokreću tijekom faze pokretanja sustava i mogu se paralelizirati s drugim procesima, što može dovesti do bržeg pokretanja osobnog procesa. Upotreba start-up servisa olakšava upravljanje procesima. Start-up servisi su konfigurirani na operacijskom sustavu, što znači da se treba brinuti o pokretanju procesa ili praćenju njihovog statusa. Ako treba promijeniti postavke pokretanja procesa, jednostavno treba ažurirati konfiguraciju start-up servisa. Start-up servisi mogu poboljšati stabilnost sustava tako što osiguravaju da se procesi uvijek pokreću i da ostanu pokrenuti, bez obzira na to tko je prijavljen na računalo. To je korisno za procese koji se trebaju pokrenuti tijekom faze pokretanja sustava, bez obzira na korisničke interakcije. Također, mogu povećati sigurnost osobnog sustava tako što omogućavaju da se proces pokrene kao korisnik s manjim ovlastima (npr. kao korisnik bez administratorskih ovlasti). To može smanjiti rizik od zlonamjernih napada na sustav ili štete koju korisnik može slučajno uzrokovati.

Start-up servis se pokreće preko `systemd service` datoteke koja opisuje kako pokrenuti Python skriptu `rpibot.py` prilikom pokretanja operacijskog sustava. U terminalu se upiše `sudo nano /etc/systemd/system/name.service`, (slika 25).

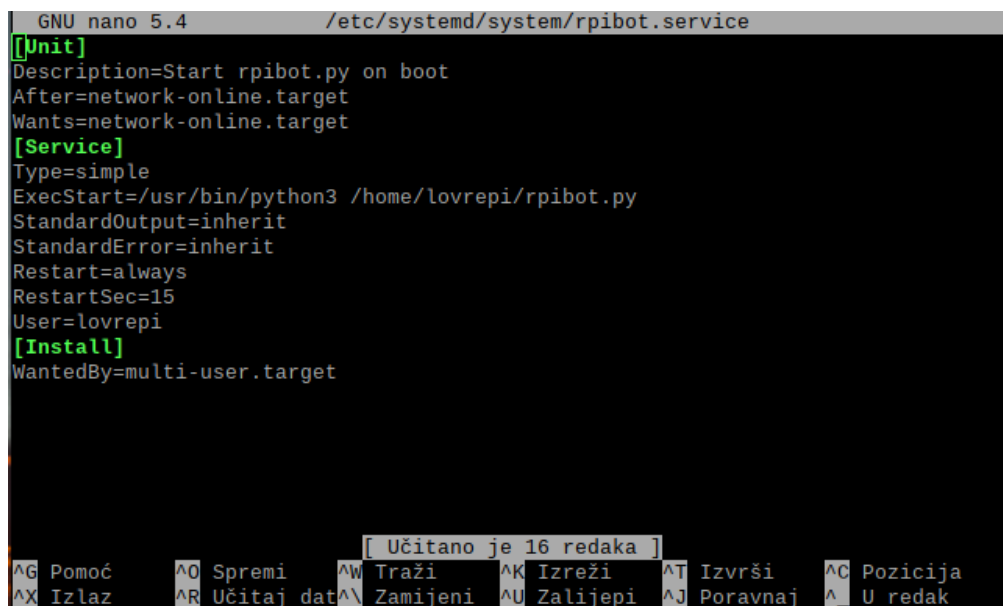
```
lovrepi@raspberrypi:~ $ sudo nano /etc/systemd/system/rpibot.service
lovrepi@raspberrypi:~ $ █
```

Slika 25: Naredba za otvaranje start-up servisa

Linije u zaglavlju `[Unit]` određuju ovisnosti servisa i kada bi se servis trebao pokrenuti. U ovom slučaju, servis se pokreće nakon što je dostupna mreža (`network-online.target`) i definirano je da želi pokrenuti tu uslugu (`Wants=network-online.target`).

Linije u zaglavlju [Service] definiraju samu uslugu. Type=simple znači da je ovo jednostavan servis koji se pokreće u istoj sesiji kao i systemd. ExecStart=/usr/bin/python3 /home/lovrepi/rpibot.py određuje koja naredba treba biti izvršena kada se servis pokrene, u ovom slučaju Python interpretator pokreće skriptu rpibot.py. StandardOutput=inherit i StandardError=inherit definiraju da sva izlazna i greška izlaza usluge nasljeđuju standardne izlaze i greške. Restart=always definira da se usluga automatski ponovno pokreće ako se zaustavi ili prekine. RestartSec=15 označava broj sekundi koji systemd čeka prije ponovnog pokretanja usluge. User=lovrepi određuje kojeg korisnika se koristi za pokretanje usluge.

Linije u zaglavlju [Install] određuju kada će se usluga pokrenuti - u ovom slučaju, pri pokretanju višekorisničkog sustava (multi-user.target) . primjer toga može se vidjeti na slici 26.



```
GNU nano 5.4 /etc/systemd/system/rpibot.service
[Unit]
Description=Start rpibot.py on boot
After=network-online.target
Wants=network-online.target
[Service]
Type=simple
ExecStart=/usr/bin/python3 /home/lovrepi/rpibot.py
StandardOutput=inherit
StandardError=inherit
Restart=always
RestartSec=15
User=lovrepi
[Install]
WantedBy=multi-user.target

[ Učitano je 16 redaka ]
^G Pomoć   ^O Spremi   ^W Traži    ^K Izreži  ^T Izvrši   ^C Pozicija
^X Izlaz    ^R Učitaj dat^_ Zamijeni ^U Zalijepi ^J Poravnaj ^_ U redak
```

Slika 26: Servis za automatsko pokretanje bota

Kada se ova systemd service datoteka spremi u direktorij /etc/systemd/system/ s imenom rpibot.service (ili neko drugo ime), može se aktivirati i pokrenuti naredbom: sudo systemctl enable rpibot.service i sudo systemctl start rpibot.service. To će omogućiti uslugu da se

pokrene prilikom pokretanja operacijskog sustava, ali prije toga u terminalu treba izvršiti naredbu `daemon reload`. `daemon reload` se koristi kada se napravi promjena u konfiguracijskim datotekama `systemd` servisa, a cilj je da se promjene odmah primijene, odnosno da se `systemd` ažurira s novim konfiguracijskim datotekama. Dakle, ako se promijeni konfiguracijska datoteka za `systemd` servis, potrebno je izvršiti `daemon-reload` kako bi se te promjene primijenile [9]. Ovo se obično radi nakon što se ažuriraju postavke `systemd` servisa, dodaju novi servisi ili promijene postojeći servisi. U slučaju da se pokrene `systemd` servis na Raspberry Pi-u, te se kasnije naprave neke promjene u konfiguracijskim datotekama koje se koriste za pokretanje tog servisa, potrebno je izvršiti `daemon-reload` naredbu kako bi se promjene primijenile na servis što je vidljivo na slici 27.

```
lovrepi@raspberrypi:~ $ sudo systemctl daemon-reload
lovrepi@raspberrypi:~ $
```

Slika 27: Prikaz `daemon-reload` naredbe u terminalu

Ako se želi provjeriti status usluge, izvršit će se sljedeća naredba `sudo systemctl status rpibot.service`, ovdje se vidi da je `rpibot.service` aktivan. Provjera aktivnosti servisa vidljiva je na slici 28.

```
lovrepi@raspberrypi:~ $ sudo systemctl status rpibot.service
● rpibot.service - Start rpibot.py on boot
   Loaded: loaded (/etc/systemd/system/rpibot.service; enabled; vendor preset:
   Active: active (running) since Tue 2023-02-28 20:54:01 CET; 16h ago
   Main PID: 1121 (python3)
     Tasks: 3 (limit: 3720)
           CPU: 830ms
   CGroup: /system.slice/rpibot.service
           └─1121 /usr/bin/python3 /home/lovrepi/rpibot.py

ožu 01 13:09:32 raspberrypi python3[1121]: File "/usr/local/lib/python3.9/dis
ožu 01 13:09:32 raspberrypi python3[1121]: conn = await self._connector.con>
ožu 01 13:09:32 raspberrypi python3[1121]: File "/usr/local/lib/python3.9/dis
ožu 01 13:09:32 raspberrypi python3[1121]: proto = await self._create_conne>
ožu 01 13:09:32 raspberrypi python3[1121]: File "/usr/local/lib/python3.9/dis
ožu 01 13:09:32 raspberrypi python3[1121]: _, proto = await self._create_di>
ožu 01 13:09:32 raspberrypi python3[1121]: File "/usr/local/lib/python3.9/dis>
ožu 01 13:09:32 raspberrypi python3[1121]: raise ClientConnectorError(req.c>
ožu 01 13:09:32 raspberrypi python3[1121]: aiohttp.client_exceptions.ClientConn>
ožu 01 13:09:40 raspberrypi python3[1121]: [2023-03-01 13:09:40] [INFO ] dis>
```

Slika 28: Prikaz za provjeru statusa servisa

3.6. Program za ažuriranje datoteke pomoću crontaba

Crontab je naredba koja omogućuje uređivanje Cron rasporeda zadataka u Linuxu. Cron je program u Unixu sličan *Windows Task Scheduleru*, koji omogućuje automatsko pokretanje zadataka u određeno vrijeme ili u određenim vremenskim intervalima.

Crontab zapis u terminalu je vidljiv na slici 29.:

```
lovrepi@raspberrypi:~ $ crontab -e
```

Slika 29: Prikaz crontab zapisa u terminalu

Linija u obliku "`* * * * *`" naredba" označava raspored izvršavanja zadatka u Cronu. Svaka zvjezdica (*) predstavlja jednu vrijednost i označava [10]:

- prva zvjezdica predstavlja minutu (0-59),
- druga zvjezdica predstavlja sat (0-23),
- treća zvjezdica predstavlja dan u mjesecu (1-31),
- četvrta zvjezdica predstavlja mjesec (1-12),
- peta zvjezdica predstavlja dan u tjednu (0-6, gdje 0 predstavlja nedjelju).

Stoga, "`54 18 * * *`" znači da će se naredba izvršiti u 18:54 svakog dana.

Ovaj Cron raspored omogućuje da skripta `/home/lovrepi/scriptainput.py` redovito ažurira datoteku `bbadwords.txt` kako bi se bot mogao ažurirati s novim zabranjenim riječima. Primjer skripte vidi se na slici 30.

```
lovrepi@raspberrypi: ~  
Datoteka Uredi Kartice Pomoć  
GNU nano 5.4 /tmp/crontab.nX6oAP/crontab  
# and what command to run for the task  
#  
# To define the time you can provide concrete values for  
# minute (m), hour (h), day of month (dom), month (mon),  
# and day of week (dow) or use '*' in these fields (for 'any').  
#  
# Notice that tasks will be started based on the cron's system  
# daemon's notion of time and timezones.  
#  
# Output of the crontab jobs (including errors) is sent through  
# email to the user the crontab file belongs to (unless redirected).  
#  
# For example, you can run a backup of all your user accounts  
# at 5 a.m every week with:  
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/  
#  
# For more information see the manual pages of crontab(5) and cron(8)  
#  
# m h dom mon dow command  
54 18 * * * /usr/bin/python3 /home/lovrepi/scriptainput.py
```

Slika 30: Prikaz skripte za ažuriranje datoteke

Ako se želi provjeriti status Crona, izvršit će se sljedeća naredba, `systemctl status cron`, ovdje se vidi da je Cron aktivan. Provjera aktivnosti Crona vidi se na slici 31.

```
lovrepi@raspberrypi: ~  
Datoteka Uredi Kartice Pomoć  
lovrepi@raspberrypi:~$ systemctl status cron  
● cron.service - Regular background program processing daemon  
   Loaded: loaded (/lib/systemd/system/cron.service; enabled; vendor preset: enabled)  
   Active: active (running) since Tue 2023-02-28 20:53:40 CET; 16h ago  
     Docs: man:cron(8)  
  Main PID: 340 (cron)  
    Tasks: 1 (limit: 3720)  
     CPU: 2.217s  
   CGroup: /system.slice/cron.service  
           └─340 /usr/sbin/cron -f  
  
velj 28 20:53:40 raspberrypi systemd[1]: Started Regular background program processing daemon.  
velj 28 20:53:40 raspberrypi cron[340]: (CRON) INFO (pidfile fd = 3)  
velj 28 20:53:40 raspberrypi cron[340]: (CRON) INFO (Running @reboot jobs)  
velj 28 20:53:40 raspberrypi CRON[343]: pam_unix(cron:session): session opened for user root (uid=0)  
velj 28 20:53:40 raspberrypi CRON[378]: (root) CMD (/bin/sleep 10; /usr/bin/python3 /home/lovrepi/scriptainput.py)  
velj 28 20:53:40 raspberrypi CRON[344]: pam_unix(cron:session): session opened for user lovrepi (uid=1000)  
velj 28 20:53:40 raspberrypi CRON[382]: (lovrepi) CMD (/usr/bin/python3 /home/lovrepi/scriptainput.py)  
velj 28 20:53:45 raspberrypi CRON[344]: (CRON) info (No MTA installed, discarding mail)  
velj 28 20:53:45 raspberrypi CRON[344]: pam_unix(cron:session): session closed for user lovrepi  
velj 28 20:53:52 raspberrypi CRON[343]: pam_unix(cron:session): session closed for user root  
lines 1-20/20 (END)
```

Slika 31: Prikaz za provjeru statusa Crona

Crontab datoteke se mogu uređivati pomoću naredbe `crontab`. Kada se pokrene bez argumenata, ova naredba prikazuje postojeći crontab korisnika. Kada se pokrene s argumentom `-e`, omogućuje uređivanje crontab datoteke.

Cron i crontab su moćni alati za automatizaciju zadanih zadataka i olakšavanje administracije Unix/Linux sistema.

4. Zaključak

U ovom završnom radu istražena je upotreba Raspberry Pi i Pythona za implementaciju Discord filtera za filtriranje sadržaja. Raspberry Pi Discord filter koristan je alat za uređivanje razgovora na Discord poslužiteljima, koji može pomoći u sprječavanju neprikladnog govora i održavanju poželjnog okruženja za komunikaciju. Implementacija u Pythonu olakšava korištenje i prilagođavanje filtra prema potrebama korisnika. U drugom poglavlju detaljno su objašnjene tehnologije koje su korištene za izradu Discord bota, opisuju se osnovni koncepti i funkcije svake od tehnologija. U trećem poglavlju opisan je detaljan postupak instalacije potrebnih programa za uspješnu izradu Discord bota te dobivamo jasnu sliku o tome kako se sve radilo u procesu izrade bota.

Na kraju, zaključuje se da je implementacija filtra Raspberry Pi Discord u Python koristan i jeftin alat za uređivanja razgovora na Discord poslužiteljima. Rad prikazuje jedan način pristupa filtriranju poruka na Discordu i može biti koristan za različite vrste Discord zajednica, uključujući one koje ciljaju na mlađu publiku ili imaju stroga pravila komunikacije.

Osim toga, ovaj rad može poslužiti i za daljnja istraživanja u području analize teksta i strojnog učenja, posebice u kontekstu uređivanja online zajednice. Budući da online komunikacija postaje sve važnija u modernom svijetu, primjena alata poput Raspberry Pi Discord filtera može pomoći u stvaranju sigurnijeg i ugodnijeg okruženja za online komunikaciju.

5. Literatura

- [1] <https://pythoninstitute.org/about-python> (zadnje posjećeno 17.02.2023.)

- [2] <https://raspberrytips.com/why-is-python-used-on-raspberry-pi/>
(zadnje posjećeno 17.02.2023.)

- [3] <https://opensource.com/resources/raspberry-pi> (zadnje posjećeno 18.02.2023.)

- [4] <https://www.raspberrypi.com/products/raspberry-pi-4-model-b/> (zadnje posjećeno 03.04.2023)

- [5] <https://static.raspberrypi.org/files/product-briefs/Raspberry-Pi-4-Product-Brief.pdf> (zadnje posjećeno 18.02.2023.)

- [6] <https://www.raspbian.org/> (zadnje posjećeno 05.03.2023.)

- [7] <https://store.epicgames.com/en-US/news/what-is-discord-and-what-is-it-used-for> (zadnje posjećeno 06.03.2023.)

- [8] <https://discord.com/developers/docs/reference> (zadnje posjećeno 06.03.2023.)

- [9] <https://www.makeuseof.com/what-is-systemd-launch-programs-raspberry-pi/>
(zadnje posjećeno 06.03.2023.)

- [10] <https://www.zyxware.com/articles/2672/the-linux-cron-crontab-and-cron-jobs-why-what-how> (zadnje posjećeno 06.03.2023.)

6. Popis slika

Slika 1: Raspberry Pi 4 Model B	6
Slika 2: Radna površina Raspbian operacijskog sustava	7
Slika 3: Izgled Discord aplikacije	9
Slika 4: Dijagram komunikacije između korisnika i bota	11
Slika 5: Instalacija Raspbian operacijskog sustava	12
Slika 6: Odabir Raspbian operacijskog sustava	13
Slika 7: Instalacija Raspbian operacijskog sustava	14
Slika 8: Postavljanje postavki za Raspbian OS	15
Slika 9: Kreiranje aplikacije za Discord bot	15
Slika 10: Stvaranje bot računa	16
Slika 11: Dodavanje ovlasti botu	16
Slika 12: Izgled pozivnice za dodavanje bota na poslužitelj	17
Slika 13: Prikaz bota iz neaktivnog u aktivno stanje	17
Slika 14: Prikaze naredbe za provjeru Python verzije	19
Slika 15: Prikaze naredbe za instalaciju Discord biblioteke	20
Slika 16: Implementacija potrebnih modula	20
Slika 17: Stvaranje potrebnih objekata	21
Slika 18: Funkcija za provjeru novih korisnika	21
Slika 19: Funkcija za provjeru pošiljatelja poruke	22
Slika 20: Prikaz poruke nakon što korisnik dobije ban	22
Slika 21: Prikaz await naredbe	23
Slika 22: Primjer funkcioniranja Discord bota u aktivnom stanju	23
Slika 23: Primjer funkcioniranja Discord bota u neaktivnom stanju	24
Slika 24: Naredba za pokretanje programa za unos sadržaja	26
Slika 25: Naredba za otvaranje start-up servisa	27
Slika 26: Servis za automatsko pokretanje bota	28
Slika 27: Prikaz daemon-reload naredbe u terminalu	29
Slika 28: Prikaz za provjeru statusa servisa	29
Slika 29: Prikaz crontab zapisa u terminalu	30
Slika 30: Prikaz skripte za ažuriranje datoteke	31
Slika 31: Prikaz za provjeru statusa Crona	31