|  |  |  |
| --- | --- | --- |
|  | | |
| RFID praćenje korisnika na radnom mestu sa Raspberry Pi 4b i mfrc522 rfid modulom | | |
|  |  |  |
| Student |  | Profesor |
| MILOš JELIć 28122008 |  | Janoš šimon |
|  | Subotica, 2023. godine |  |

Sadržaj

[1. O PROJEKTU 3](#_Toc126049765)

[2. KORIŠĆENE KOMPONETE U PROJEKTU 4](#_Toc126049766)

[2.1. Raspberry Pi 4B 4GB: 4](#_Toc126049767)

[2.2. MFRC522 RFID Modul: 4](#_Toc126049768)

[2.3. Crveni LED 5mm: 4](#_Toc126049769)

[2.4. Zeleni LED 5mm: 4](#_Toc126049770)

[2.5. Buzzer: 4](#_Toc126049771)

[2.6. LCD 16x2: 5](#_Toc126049772)

[2.7. Potenciometar 10K: 5](#_Toc126049773)

[2.8. Kratkospojnici: 5](#_Toc126049774)

[2.9. Breadboard: 5](#_Toc126049775)

[3. POVEZIVANJE 6](#_Toc126049776)

[3.1. Povezivanje LCD 16x2 sa potenciometrom i raspberry pi 6](#_Toc126049777)

[3.2. Povezivanje MFRC522 RFID čitača sa Rasperry pi 6](#_Toc126049778)

[3.3. Povezivanje buzzera, crvenog i zelenog leda. 6](#_Toc126049779)

[3.4. Šema povezivanja 7](#_Toc126049780)

[4. INSTALACIJA OPERATIVNOG SISTEMA NA RASPBERRY PI I KONFIGURACIJA 8](#_Toc126049781)

[5. PYTHON SKRIPTE 9](#_Toc126049782)

[5.1. EVIDENCIJA PROLASKA 9](#_Toc126049783)

[5.2. REGISTRACIJA KORISNIKA 11](#_Toc126049784)

[6. JAVA APLIKACIJA 15](#_Toc126049785)

[6.1. UREĐIVANJE KORISNIKA 15](#_Toc126049786)

[6.2. EVIDENCIJA PROLASKA 16](#_Toc126049787)

[6.3. KOMUNIKACIJA IZMEĐU JAVA APLIKACIJE I RASPBERRY PI 17](#_Toc126049788)

[7. LITERATURA 19](#_Toc126049789)

# O PROJEKTU

Projekat koristi Raspberry Pi 4B kao glavni uređaj koji je povezan sa MFRC522 RFID modulom. Modul čita RFID kartice i prosleđuje informacije Raspberry Pi-ju. Dve Python skripte koriste se za čitanje RFID kartica i registraciju korisnika. Prva skripta čita informacije sa kartice i proslijeđuje ih u bazu podataka gde se vrši evidencija prolaska.Druga skripta omogućuje registraciju korisnika gde se definišu opšti podaci o radniku i broj kartice. Svi podaci se čuvaju u MariaDB bazi podataka.

Java aplikacija koristi se za čitanje podataka iz baze i upravljanje njima. Korisnik može da izmeni postojeće podatke o korisniku u bazi, kao i da unosi nove. Takođe, Java aplikacija može da pokrene Python skripte putem SSH-a, što omogućava dinamičko upravljanje procesom čitanja RFID kartica.

Ukratko, projekat koristi Raspberry Pi 4B sa MFRC522 modulom za čitanje RFID kartica, Python skripte za čuvanje informacija u bazi, a Java 8 aplikacija za upravljanje podacima i pokretanje Python skripti.

# KORIŠĆENE KOMPONETE U PROJEKTU

## Raspberry Pi 4B 4GB:

* Četvorojezgarni procesor Broadcom BCM2711, Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz
* 4GB LPDDR4-3200 SDRAM (2x 2GB)
* Dual-band 802.11ac wireless, Bluetooth 5.0, BLE
* Gigabit Ethernet
* 2 USB 3.0 ports; 2 USB 2.0 ports.
* 40-pin GPIO header
* 2 × micro-HDMI port (up to 4kp60 supported)
* 3.5mm audio jack
* 5V DC power input via USB-C connector (minimum 3A\*)
* 5V DC power input via GPIO header (minimum 3A\*)

Mini računar sa četvorojezgarnim procesorom i 4GB radne memorije.

Podržava dual-band bežičnu mrežu, Bluetooth, Gigabit Ethernet i ima 4 USB porta.

Sa 40-pinski GPIO header-om, može da se koristi za razne projekte sa dodatnim senzorima i komponentama.

## MFRC522 RFID Modul:

* Radna frekvencija: 13.56 MHz
* Kompatibilnost sa ISO 14443A/MIFARE karticama
* Brzina komunikacije do 848 kbps
* Podrška za čitanje i pisanje na kartice

## Crveni LED 5mm:

* Napajanje: 3-5V
* Boja: crvena

## Zeleni LED 5mm:

* Napajanje: 3-5V
* Boja: zelena

## Buzzer:

* Napajanje: 3-5V
* Frekvencija: preko 3KHz

## LCD 16x2:

* 16 kolona, 2 reda
* Širina karaktera: 5x8 piksela
* Kontrolni kodovi: HD44780 (ili kompatibilni)

## Potenciometar 10K:

* Otpornost: 10KΩ
* Otpornost varira linijski sa okretanjem kapice

## Kratkospojnici:

* M/M, M/Ž, Ž/Ž
* Služe za povezivanje elektronskih komponenti.

## Breadboard:

* Ima mrežu odgovarajućih šupljih vodova za priključivanje komponenti.
* Lako se koristi za probne spojeve.
* Bez potrebe za lemljenjem, što ga čini idealnim za studentske projekte i prototipe.

# POVEZIVANJE

## Povezivanje LCD 16x2 sa potenciometrom i raspberry pi

**Pin 1** LCD (Uzemljenje) na breadboard uzemljenje liniju

**Pin 2** LCD (VCC / 5V) na breadboard pozitivnu liniju

**Pin 3** LCD (V0) na srednji pin potentiometra

**Pin 4** LCD (RS) na GPIO4 (Fizički Pin 7)

**Pin 5** LCD (RW) na breadboard uzemljenje liniju

**Pin 6** LCD (EN) na GPIO24 (Fizički Pin 18)

**Pin 11** LCD (D4) na GPIO23 (Fizički Pin 16)

**Pin 12** LCD (D5) na GPIO17 (Fizički Pin 11)

**Pin 13** LCD (D6) na GPIO18 (Fizički Pin 12)

**Pin 14** LCD (D7) na GPIO22 (Fizički Pin 15)

**Pin 15** LCD (LED +) na breadboard pozitivnu liniju

**Pin 16** LCD (LED -) na breadboard uzemljenje liniju

## Povezivanje MFRC522 RFID čitača sa Rasperry pi

**SDA** MFRC522 povezati na GPIO8 (Fizički Pin 24)

**SCK** MFRC522 povezati na GPIO11 (Fizički Pin 23)

**MOSI** MFRC522 povezati na GPIO10 (Fizički Pin 19)

**MISO** MFRC522 povezati na GPIO9 (Fizički Pin 21)

**GND** MFRC522 povezati na Breadboard uzemljenje liniju

**RST** MFRC522 povezati na GPIO25 (Fizički Pin 22)

**3.3v** MFRC522 povezati na 3v3 (Fizički Pin 1)

**IRQ** MFRC522 ostaje slobodan

## Povezivanje buzzera, crvenog i zelenog leda.

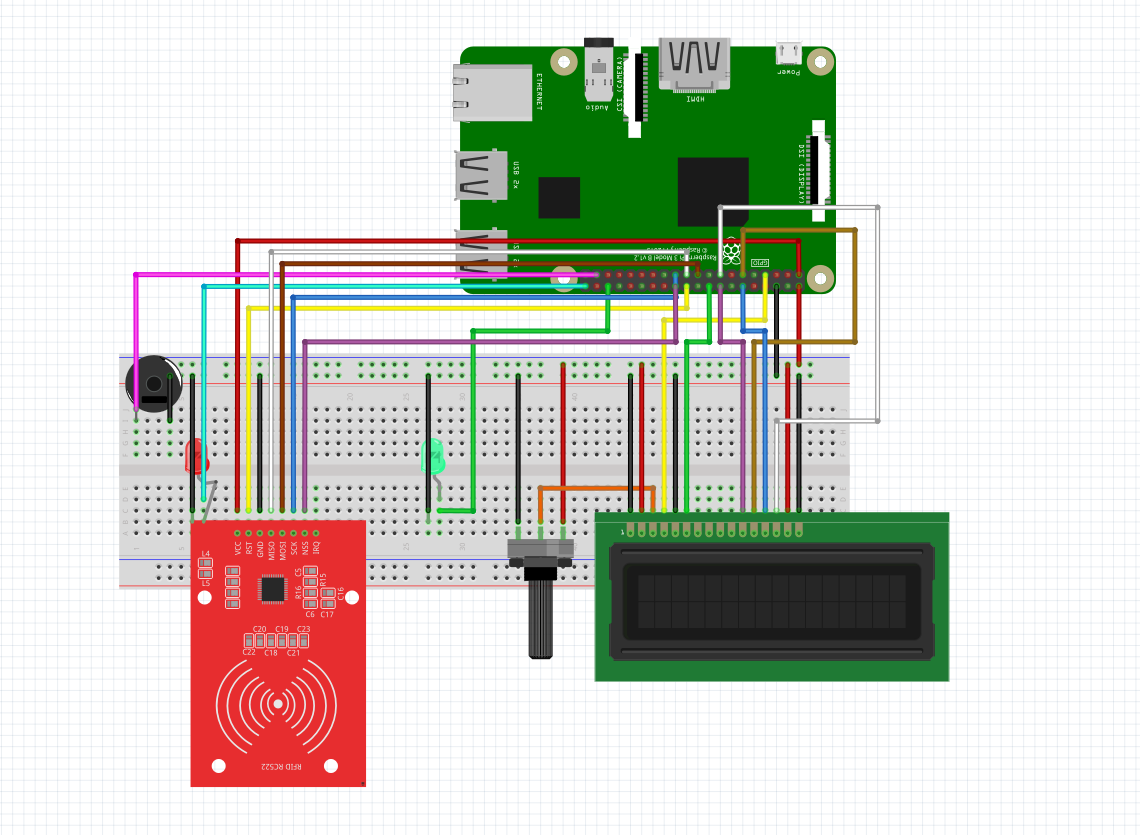
GND svih komponenti na breadboard uzemljenje liniju

Crvena LED + na GPIO 21 (Fizički Pin 40)

Zelena LED + na GPIO 16 (Fizički Pin 36)

Buzzer + na GPIO 26 (Fizički Pin 37)

## Šema povezivanja



# INSTALACIJA OPERATIVNOG SISTEMA NA RASPBERRY PI I KONFIGURACIJA

Za instalaciju Raspberry Pi OS-a pomoću Raspberry Pi Imager-a, preuzmite ga sa službenog sajta Raspberry Pi, pokrenite aplikaciju i odaberite željeni operativni sistem. Odaberite SD karticu na kojoj će se instalirati, kliknite na "Write" i sačekajte da se proces završi. Uključite Raspberry Pi i prijavite se na operativni sistem, zatim idite na "Preferences" -> "Raspberry Pi Configuration" -> "Interfaces" i omogućite SPI interfejs klikom na "Enabled". Ovom konfiguracijom završava se instalacija Raspberry Pi OS-a i omogućavanje SPI interfejsa.

# PYTHON SKRIPTE

## EVIDENCIJA PROLASKA

Ovo je Python skripta koja čita podatke sa RFID čitača i ažurira bazu podataka prisustva. Skripta koristi MFRC522 biblioteku za komunikaciju sa RFID čitačem i MySQL Connector za interakciju sa bazom podataka. Skripta takođe koristi GPIO biblioteku da kontroliše LED, zvučnik i karakter LCD da prikaže poruke.

Skripta uspostavlja vezu sa bazom podataka, podešava LED, zvučnik i karakter LCD i ulazi u beskonačnu petlju. U svakoj iteraciji, zahteva korisnika da skenira RFID karticu, čita ID sa kartice i preuzima odgovarajuće informacije o korisniku iz baze podataka. Ako korisnik postoji u bazi podataka, skripta beleži njihovo prisustvo, uključuje zeleni LED, zvučnik i prikazuje poruku dobrodošlice na karakter LCD-u. Ako korisnik ne postoji u bazi podataka, skripta uključuje crveni LED, zvučnik i prikazuje poruku o grešci na karakter LCD-u. Skripta zatim ponavlja proces u sledećoj iteraciji.

#!/usr/bin/env python

import time

import RPi.GPIO as GPIO

from mfrc522 import SimpleMFRC522

import mysql.connector

import board

import digitalio

import adafruit\_character\_lcd.character\_lcd as character\_lcd

db = mysql.connector.connect(

host="localhost",

user="test",

passwd="test123",

database="iotrfid"

)

LED\_PIN\_RED = 21

LED\_PIN\_GREEN = 16

BUZZER\_PIN = 26

cursor = db.cursor()

reader = SimpleMFRC522()

lcd\_rs = digitalio.DigitalInOut(board.D4)

lcd\_en = digitalio.DigitalInOut(board.D24)

lcd\_d7 = digitalio.DigitalInOut(board.D22)

lcd\_d6 = digitalio.DigitalInOut(board.D18)

lcd\_d5 = digitalio.DigitalInOut(board.D17)

lcd\_d4 = digitalio.DigitalInOut(board.D23)

lcd\_backlight = digitalio.DigitalInOut(board.D4)

lcd\_columns = 16

lcd\_rows = 2

lcd = character\_lcd.Character\_LCD\_Mono(lcd\_rs, lcd\_en, lcd\_d4, lcd\_d5, lcd\_d6, lcd\_d7, lcd\_columns, lcd\_rows, lcd\_backlight)

GPIO.setmode(GPIO.BCM)

GPIO.setup(LED\_PIN\_RED, GPIO.OUT)

GPIO.setup(LED\_PIN\_GREEN, GPIO.OUT)

GPIO.setup(BUZZER\_PIN, GPIO.OUT)

try:

while True:

lcd.clear()

lcd.message = 'Skeniraj karticu'

id, text = reader.read()

cursor.execute("Select id, name FROM users WHERE aktivan AND vazeci AND rfid\_uid="+str(id))

result = cursor.fetchone()

lcd.clear()

if cursor.rowcount >= 1:

lcd.message = "Dobrodosli \n" + result[1]

cursor.execute("INSERT INTO attendance (user\_id) VALUES (%s)", (result[0],) )

GPIO.output(LED\_PIN\_GREEN, GPIO.HIGH)

GPIO.output(BUZZER\_PIN, GPIO.HIGH)

time.sleep(0.15)

GPIO.output(BUZZER\_PIN, GPIO.LOW)

time.sleep(1.5)

GPIO.output(LED\_PIN\_GREEN, GPIO.LOW)

db.commit()

else:

lcd.message = "Ne postoji \nkorisnik."

GPIO.output(LED\_PIN\_RED, GPIO.HIGH)

GPIO.output(BUZZER\_PIN, GPIO.HIGH)

time.sleep(0.9)

GPIO.output(BUZZER\_PIN, GPIO.LOW)

time.sleep(1.5)

GPIO.output(LED\_PIN\_RED, GPIO.LOW)

time.sleep(1)

finally:

GPIO.cleanup()

## REGISTRACIJA KORISNIKA

Ovaj kod pokazuje kako se može koristiti RFID čitač kartica i LCD displej sa Raspberry Pi-jem za registraciju korisnika u bazu podataka MySQL. Korišćenjem biblioteke MFRC522 čita se RFID kartica, a korišćenjem biblioteke Adafruit\_Character\_LCD čita se LCD displej. Kada se učita RFID kartica, proverava se da li korisnik već postoji u bazi podataka. Ako postoji, korisnik ima opciju da ažurira svoje ime ili da nastavi sa novim korisnikom. Ukoliko korisnik ne postoji, unosi se ime korisnika i RFID broj kartice. Rezultati se čuvaju u bazi podataka i ispisuju na LCD displeju.

#!/usr/bin/env python

import time

import RPi.GPIO as GPIO

from mfrc522 import SimpleMFRC522

import mysql.connector

import board

import digitalio

import adafruit\_character\_lcd.character\_lcd as character\_lcd

db = mysql.connector.connect(

host="localhost",

user="test",

passwd="test123",

database="iotrfid"

)

LED\_PIN\_GREEN = 16

LED\_PIN\_YELLOW = 14

cursor = db.cursor()

reader = SimpleMFRC522()

lcd\_rs = digitalio.DigitalInOut(board.D4)

lcd\_en = digitalio.DigitalInOut(board.D24)

lcd\_d7 = digitalio.DigitalInOut(board.D22)

lcd\_d6 = digitalio.DigitalInOut(board.D18)

lcd\_d5 = digitalio.DigitalInOut(board.D17)

lcd\_d4 = digitalio.DigitalInOut(board.D23)

lcd\_backlight = digitalio.DigitalInOut(board.D4)

lcd\_columns = 16

lcd\_rows = 2

lcd = character\_lcd.Character\_LCD\_Mono(lcd\_rs, lcd\_en, lcd\_d4, lcd\_d5, lcd\_d6, lcd\_d7, lcd\_columns, lcd\_rows, lcd\_backlight)

GPIO.setmode(GPIO.BCM)

GPIO.setwarnings(False)

GPIO.setup(LED\_PIN\_YELLOW, GPIO.OUT)

GPIO.setup(LED\_PIN\_GREEN, GPIO.OUT)

try:

while True:

lcd.clear()

lcd.message ='Postavi karticu\nza registraciju'

id, text = reader.read()

cursor.execute("SELECT id FROM users WHERE rfid\_uid="+str(id))

cursor.fetchone()

if cursor.rowcount >= 1:

lcd.clear()

GPIO.output(LED\_PIN\_YELLOW,GPIO.HIGH)

lcd.message = "Korisnik vec\npostoji!"

overwrite = input("Korisnik vec postoji, da li zelite da azurirate ime (Y/N)? ")

if overwrite[0] == 'Y' or overwrite[0] == 'y':

lcd.clear()

GPIO.output(LED\_PIN\_YELLOW,GPIO.LOW)

lcd.message = "Izmena \nsacuvana."

#GPIO.output(LED\_PIN\_GREEN,GPIO.HIGH)

time.sleep(1)

#GPIO.output(LED\_PIN\_GREEN,GPIO.OUT)

sql\_insert = "UPDATE users SET name = %s WHERE rfid\_uid=%s"

else:

continue;

else:

sql\_insert = "INSERT INTO users (name, rfid\_uid) VALUES (%s, %s)"

lcd.clear()

GPIO.output(LED\_PIN\_YELLOW,GPIO.HIGH)

lcd.message = 'Unesi ime'

new\_name = input("Ime: ")

cursor.execute(sql\_insert, (new\_name, id))

db.commit()

lcd.clear()

GPIO.output(LED\_PIN\_YELLOW,GPIO.LOW)

lcd.message = "Sacuvan korisnik\n" + new\_name

print("Korisnik : " + new\_name + "\nBroj kartice : " + str(id) + "\nUspesno sacuvan")

GPIO.output(LED\_PIN\_GREEN,GPIO.HIGH)

time.sleep(2)

GPIO.output(LED\_PIN\_GREEN,GPIO.OUT)

finally:

GPIO.cleanup()

# JAVA APLIKACIJA

Java aplikacija sa dva korisčka interfejsa. Prvi interfejs se koristi za praćenje RFID kartica i prikazuje podatke u tabelarnom obliku. Tabela prikazuje sve informacije o RFID karticama koje su pročitane i obrađene. Pored tabele, interfejs takođe prikazuje detalje poslednjeg prijavljenog korisnika. Informacije su prikazane jasno i kratko, čime se olakšava razumevanje i korišćenje.

Interfejs koristi JSch biblioteku za komunikaciju sa Raspberry Pi putem SSH. Komunikacija se uspostavlja korišćenjem definisanog ključa, što osigurava sigurnu transmisiju podataka. Ova komunikacija omogućava interfejsu da izvršava Python skripte, čime je moguće pokretati razne automatizovane procese.

Drugi interfejs prikazuje sve registrovane korisnike aplikacije i mogućnost izmene njihovih informacija. Interfejs je dizajniran na način koji je korisnički prijateljski, čime se olakšava upravljanje informacijama korisnika. Interfejs je takođe dizajniran da bude visoko efikasan i može da obradi veliki broj korisnika bez problema sa performansom.

## UREĐIVANJE KORISNIKA

Kod predstavlja Java klasu UsersPregled koja proširuje klasu javax.swing.JPanel. Ona obezbeđuje korisčki interfejs za pregled i uređivanje informacija o korisnicima.

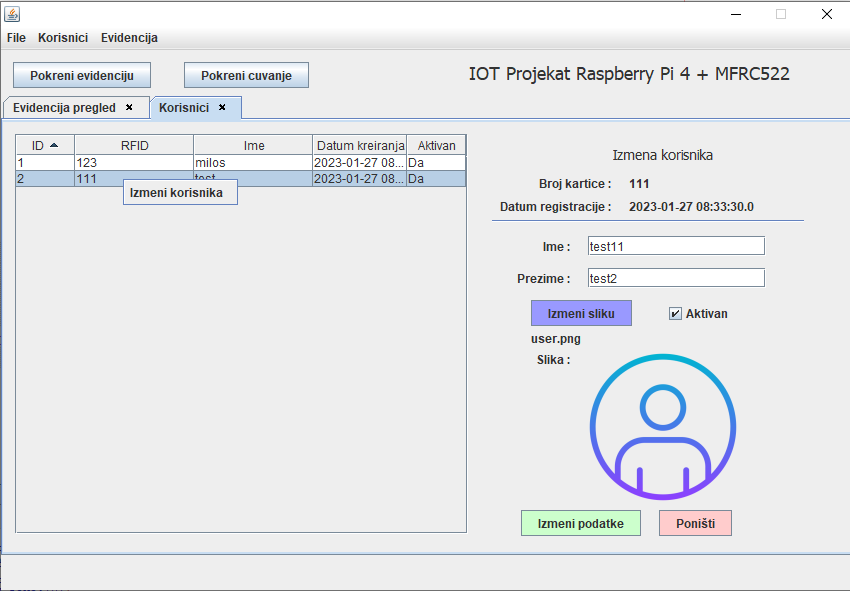
Ima privatnu statičku konekciju conSQL objekta, koja je konekcija sa MySQL bazom podataka. URL konekcije se čuva u konstantnom objektu connectionUrlMySQL i obezbeđuje informacije o tome kako se povezati sa bazom (IP adresa, port, ime baze, korisničko ime i lozinka).

Varijabla izmenaSlika se koristi za praćenje promene slike korisčka. Stringovi path i destination predstavljaju lokaciju slike, dok promenljiva nameSlika ćuva ime fajla slike.

Klasa UsersPregled ima podrazumevani table model tm koji se koristi za prikaz podataka u tabeli. Kod takođe podešava kolone za tabelu (ID, datum i status), i podešava željenu širinu za svaku kolonu.

Konstruktor podešava korisčki interfejs i povezuje se sa bazom koristeći konstantni objekat connectionUrlMySQL. Konstruktor poziva metodu getUsers kojapreuzima informacije o svim korisnicima iz baze i popunjava tabelu sa podacima. Metoda koristi TreeMap strukturu podataka za praćenje korisčka, i za svakog korisnika preuzima njihove ID, ime, prezime, datum kreiranja, aktivni status i informacije o slici iz baze. Zatim dodaje podatke u tabelu. Konstruktor takođe podešava desni klik miša za prikazivanje iskačućeg menija sa opcijama za izmenu svakog red u tabeli.

Metoda getSelectedUser prima ID i preuzima informacije o odabranom korisniku iz tabele. Podešava vrednosti za razne polja u korisčkom interfejsu, kao što su ime, prezime, datum kreiranja, aktivni status i slika, na osnovu informacija preuzetih iz baze. Određeni podaci o korisniku se mogu izmeniti.



## EVIDENCIJA PROLASKA

Ovaj kod definiše Java klasu pod nazivom "DataPregled" koja proširuje klasu JPanel iz Java Swing biblioteke. Klasa ima nekoliko polja na nivou klase, uključujući statički objekat tipa Connection nazvan "conSQL", statičku konstantnu string vrednost nazvanu "connectionUrlMySQL" koja sadrži URL konekcije do MySQL baze podataka, objekat tipa DefaultTableModel nazvan "tm", objekat druge klase pod nazivom "UsersPregled" nazvan "up".

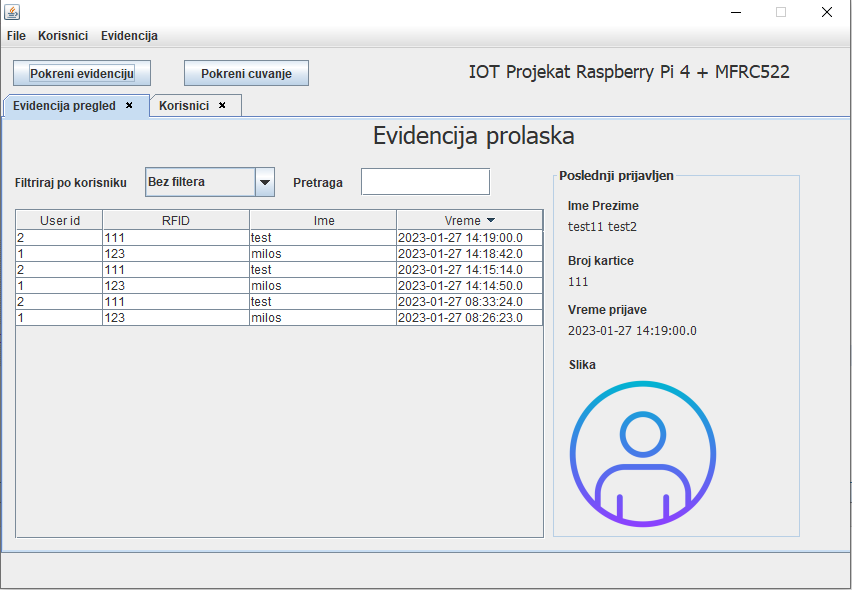
Klasa takođe ima konstruktor koji inicijalizuje nekoliko komponenata GUI-a, povezuje se sa MySQL bazom podataka, popunjava tabelu podacima iz baze podataka i sluša za ažuriranja u bazi.

Metoda "listenForUpdates" kreira objekat tipa Timer i objekat tipa TimerTask koji se pokreće svakih 5 sekundi. Run metoda objekta TimerTask uspostavlja konekciju sa bazom podataka i proverava broj redova u tabeli "prisustvo". Ako se broj redova promenio, metoda poziva metode "getData" i "lastEnter" da preuzme nove podatke i ažurira GUI.

Metoda "getData" preuzima podatke iz tabela "prisustvo" i "korisnici" u bazi podataka i dodaje te podatke u tabelu u GUI-u.

Metoda "lastEnter" preuzima poslednji red podataka iz tabele "prisustvo" i prikazuje ih u GUI-u.

Klasa takođe definiše dve unutrašnje klase, "Podaci" i "OpstiPodaci", koje čuvaju podatke iz tabela "prisustvo" i "korisnici".



## KOMUNIKACIJA IZMEĐU JAVA APLIKACIJE I RASPBERRY PI

Kod je napisan u javi i sadrži implementaciju protokola SSH za povezivanje sa udaljenim uređajem i izvršavanje Python skripti na njemu.

Postoji privatni statički string 'privateKey' koji sadrži put do datoteke privatnog ključa, i dva privatna statička stringa 'user' i 'host' koja čuvaju korisčko ime i IP adresu udaljenog uređaja

Kod ima dva metoda: 'sshCheck()' i 'sshCheckCancel()', koji se koriste za povezivanje sa udaljenim uređajem i izvršavanje Python skripte 'check.py' na njemu. Oba metoda slede istu logiku za uspostavljanje veze, otpremanje python datoteke i izvršavanje skripte.

Metod 'sshCheck()' koristi JSch biblioteku za implementaciju protokola SSH. Kreira novi JSch, dodaje privatni ključ u njega, postavlja korisčko ime i host udaljenog uređaja i postavlja svojstva za StrictHostKeyChecking i šifru. Zatim kreira novu sesiju i povezuje se sa udaljenim uređajem. Nakon toga, otvara SFTP kanal za otpremanje python datoteke i izvršava skriptu na uređaju. Boolean varijabla 'sshCheckBoolean' postavljena je na true nakon izvršenja. Na kraju, metod se odjavljuje sa udaljenog uređaja.

Metod 'sshCheckCanel()' koristi istu biblioteku i služi za zaustavljanje pokrenuti skripte check.py

# LITERATURA

[1] <https://pimylifeup.com/>

[2] <http://www.jcraft.com/jsch/>

[3] <https://www.youtube.com/watch?v=KjD7qEIMDeM> - Generate SSH key on Linux and Convert SSH key to PPK

# 