Razvoj programskog rješenja za kontrolu laboratorijskih uređaja

Soldatić, David

Sveučilište Jurja Dobrile u Puli
Tehnički fakultet u Puli

David Soldatić

RAZVOJ PROGRAMSKOG RJEŠENJA ZA KONTROLU LABORATORIJSKIH UREĐAJA
APPLICATION DEVELOPMENT FOR LABORATORY DEVICES CONTROL

Završni rad

Pula, rujan 2024. godine

Sveučilište Jurja Dobrile u Puli
Tehnički fakultet u Puli

David Soldatić

RAZVOJ PROGRAMSKOG RJEŠENJA ZA KONTROLU LABORATORIJSKIH UREĐAJA
APPLICATION DEVELOPMENT FOR LABORATORY DEVICES CONTROL

Završni rad

JMBAG: 0303101010, redoviti student
Studijski smjer: Računarstvo

Predmet: Informacijske tehnologije i društvo
Znanstveno područje: Tehničke znanosti
Znanstveno polje: Računarstvo
Znanstvena grana: Web tehnologije
Mentor: izv. prof. dr. sc. Sven Maričić

Pula, Rujan 2024.godine

SADRŽAJ

1. Uvod

Razvoj 3D ispisa omogućio je bržu i precizniju izradu fizičkih modela pružajući nove pristupe razvoju i proizvodnji prototipa. 3D pisači postaju sve češći u mnogim industrijama, poput inženjerstva, dizajna, obrazovanja i medicine. Ekstremna točnost s kojom ovi alati mogu proizvesti proizvode i prototipove umanjuje potrebu za tradicionalnim metodama proizvodnje.

Unatoč svim njihovim prednostima, upravljanje više 3D pisača odjednom može biti teško. Mali laboratoriji, proizvodni pogoni i obrazovne ustanove ponekad trebaju centraliziranu administraciju. Cilj ovog zadnjeg zadatka je stvoriti sustav za kontrolu nekoliko 3D printera pomoću jednostavnog programa "EasyPrint". Ovaj alat omogućuje daljinsko upravljanje ispisom i izmjene raznih parametara u stvarnom vremenu.

U projektu korišteni su 3D printeri Prusa MK2S s različitim glavama (0,4 mm i 0,25 mm). USB hub povezan je s Raspberry Pi 4 putem USB kabla iz razloga što Raspberry ima samo 4 USB priključka. Uz pomoć softvera EasyPrint, korisnici mogu koristiti program za odabir pisača na kojem žele ispisivati i za pokretanje automatiziranog procesa koji će zagrijati ili kalibrirati pisač prije ispisa. Cilj je bio izraditi aplikaciju koja će umrežiti sve printere i učiniti odabir printera jednostavnim za ispis kroz laboratorijski rad.

Softver olakšava praćenje ispisa zajedno s kontrolom raznih postupaka poput kalibracije temperature, umetanja i vađenja filamenata i kalibracije. Upravljanje svim povezanim pisačima postaje mnogo lakše kad se koristi Raspberry Pi 4 kao središnji uređaj za povezivanje. Moderne tehnologije kao što su Flask i Python, kao i uobičajene web tehnologije poput HTML, CSS i JavaScript, koriste se u tehničkoj implementaciji ovog projekta.

Ovakav sustav može pojednostaviti proces nadgledanja nekoliko 3D printera, omogućujući korisnicima da nadziru i upravljaju s udaljenosti bez potrebe da osobno posjećuju svaki printer. Želio bih se zahvaliti svima koji su mi pružili podršku tijekom razvoja aplikacije EasyPrint. Posebne zahvale idu mom mentoru, izv. prof. Svenu Maričiću, na njegovom neizmjernom strpljenju, savjetima i pomoći kroz cijeli proces izrade ove aplikacije. Njegova stručnost bila je ključna za rješavanje izazova i uspješan završetak rada.

2. Cilj rada

Cilj rada je razviti i implementirati web aplikaciju koja omogućava:

- Daljinsko upravljanje s više 3D printera
- Nadgledanje stanja printera
- Prijenos datoteka (GCode) i pokretanje ispisa direktno kroz web sučelje
- Upravljanje temperaturom ispisne glave i podloge
- Funkcije "preheat" i "cooldown" za optimizaciju ispisnog procesa
- Kalibraciju printera putem aplikacije, čime se omogućuje jednostavno podešavanje
- Funkcije “load” i “unload” filament
- Dinamičko prepoznavanje printera kako bi aplikacija automatski prepoznala i registrirala nove uređaje povezane putem USB-a, čime se povećava fleksibilnost i skalabilnost sustava

Osnovna funkcionalnost ove aplikacije je upravljanje 3D printerima putem jednostavnog i intuitivnog korisničkog sučelja.

3. Korištene tehnologije

3.1. Raspberry Pi 4

Raspberry Pi 4 odabran je kao glavna platforma za ovaj projekt zbog svoje prilagodljivosti, pristupačnosti, male veličine i kompatibilnosti s više uređaja. Također, njegova niska potrošnja energije čini ga idealnim za kontinuirani rad u okruženjima gdje je dugotrajan rad ključan. Raspberry Pi 4 ima četverojezgreni ARM Cortex-A72 procesor, 8 GB RAM-a (postoje alternativne verzije s 2 GB i 4 GB RAM-a), USB, HDMI i GPIO priključke. Ove karakteristike čine ga idealnim za rad s perifernom opremom poput 3D pisača.

Povezivost: korištenjem Ethernet priključka ili Wi-Fi mreže za povezivanje s internetom, Raspberry Pi 4 može se upravljati daljinski. CSI konektor omogućuje spajanje na kameru, što omogućuje praćenje ispisa u stvarnom vremenu. USB se koristi za integraciju 3D pisača, a njegova mogućnost proširenja omogućava povezivanje dodatnih perifernih uređaja ako je potrebno. GCode naredbe mogu se slati i primati između Raspberry Pi i 3D pisača putem standardnih serijskih protokola, čime se osigurava precizna i pouzdana kontrola nad svakim aspektom ispisa.

Glavne prednosti Raspberry Pi 4:

- Savršen je za kontinuirani rad zbog male potrošnje energije
- Mogućnosti proširenja: povezivanje za dodatnu funkcionalnost na različite kamere i senzore
- Podrška za različite operativne sustave: Iako su Ubuntu i Raspbian (sada Raspberry Pi OS) najčešće korišteni operativni sustavi, oni se također mogu koristiti

3.2. Python

Python je odabran kao pozadinski programski jezik zbog svoje lakoće čitanja i jednostavne sintakse koja nudi podršku kroz mnoštvo biblioteka. Osim toga, njegova velika fleksibilnost omogućava brzo prototipiranje i razvoj složenih aplikacija, što ga čini idealnim izborom za projekte koji zahtijevaju agilnost. Python je svestran za stvaranje aplikacija različitih razina složenosti jer podržava širok raspon programskih paradigmi, kao što su proceduralno, objektno orijentirano i funkcionalno programiranje, što doprinosi njegovoj širokoj primjeni u industriji.

Za ovaj projekt korišten je Pythonov okvir Flask; prikazan na primjeru 1:

```
from flask import Flask
app = Flask(   name )
@app.route('/')
def home():
    return "Dobrodošli u EasyPrint!"

if   name  == ' main ':
    app.run(debug=True)
```

Primjer 1. Prikaz Python okvira uz Flask

3.3. Flask

Python mikro web okvir nazvan Flask olakšava izradu web aplikacija. Flask ne dolazi s unaprijed izgrađenim komponentama kao što su sustavi provjere autentičnosti ili ORM-ovi, što ga čini vrlo prilagodljivim za manje projekte poput ovog. Umjesto korištenja unaprijed definiranih modula, ova fleksibilnost omogućuje programerima odabir alata i biblioteka potrebnih za određenu funkcionalnost.

Flask je bio očit izbor za ovaj projekt jer olakšava stvaranje RESTful API-ja, koji olakšava komunikaciju između sučelja i pozadine aplikacije. Implementirane su sve potrebne funkcije, uključujući kontrolu temperature, prikaz statusa ispisa i slanje GCode naredbi pisačima, korištenjem osnovne funkcije za definiranje ruta.

Primjer osnovne Flask aplikacije:

```
from flask import Flask, render_template

app = Flask(   name )

@app.route('/')
def index():
    return render_template('index.html')

if   name  == ' main ':
    app.run(debug=True)
```

Primjer 2. Prikaz osnovne Flask aplikacije

U EasyPrint aplikaciji, Flask upravlja komunikacijom između korisnika i backend sustava, osiguravajući da su svi podaci uvijek ažurirani u stvarnom vremenu.

3.4. HTML/CSS/JavaScript

Tri temeljne tehnologije za razvoj web aplikacija - HTML, CSS i JavaScript - koriste se za izradu prednjeg kraja aplikacije. Struktura stranice definirana je HTML-om (HyperText Markup Language). HTML oznake definiraju svaki element koji korisnik vidi na stranici, uključujući tekst, gumbe, slike i poveznice.

HTML dokumenti stilizirani su pomoću CSS-a (Cascading Style Sheets). Daje vam kontrolu nad izgledom stranice, uključujući brzinu odziva, raspored elemenata, sheme boja i slova. JavaScript je zadužen za dinamičke promjene i interaktivnost stranice.

Omogućuje programu da odgovori na korisnički unos i ažurira sadržaj stranice bez potrebe za ponovnim učitavanjem stranice. JavaScript se koristi u aplikaciji EasyPrint za dinamičko ažuriranje informacija o statusu pisača u stvarnom vremenu.

3.5. GCode

Uobičajeni jezik za upravljanje CNC opremom, poput 3D pisača, zove se GCode. Svaka GCode naredba označava instrukciju koju 3D printer izvršava. Svi zadaci koje 3D printer treba izvršiti za ispis navedeni su pomoću GCode-a. Ove naredbe reguliraju ekstruder, temperaturu, brzinu ispisa i druge parametre uz kretanje ispisne glave u X, Y i Z osi. GCode naredbe bitne su za točnu i preciznu kontrolu pisača, a u aplikaciji EasyPrint šalju se izravno na pisače putem USB sučelja. GCode omogućava ne samo kontrolu osnovnih funkcija ispisa, već i naprednih operacija poput automatske kalibracije i niveliranja podloge. Svaka naredba mora biti pažljivo isplanirana i izvršena kako bi se osigurao kvalitetan i pouzdan ispis. Time je osigurano da korisnik ima potpunu kontrolu nad procesom ispisa, od početne kalibracije do finalnog proizvoda.

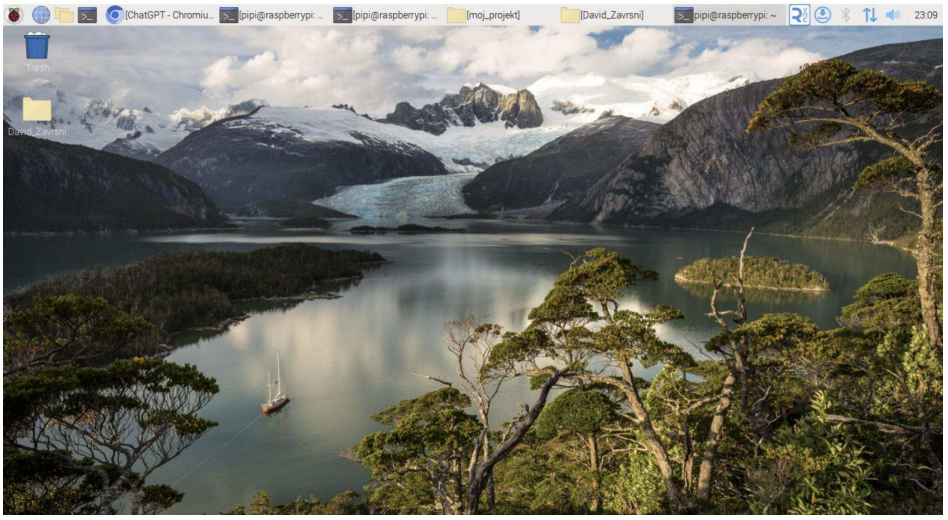
Primjeri naredbi GCode:

G28: Glava za ispis vraća se u početni položaj (početak)
G1 X100 Y100: Pomaknite glavu za ispis na položaje X=100 i Y=100
M104 S200: Na 200°C je postavljena temperatura ispisne glave
M140 S60: Postavljanje supstrata za ispis na 60 stupnjeva Celzijusa

Potpuna kontrola nad postupkom ispisa, uključujući kalibraciju, pokretanje i zaustavljanje ispisa i podešavanje parametara ispisa, omogućena je integracijom GCode naredbi u aplikaciju. Ova fleksibilnost GCode-a omogućava prilagodbu postupka ispisa različitim materijalima i potrebama korisnika, što je ključno za postizanje visokokvalitetnih rezultata.

4. Postavljanje i spajanje Raspberry Pi-a za upravljanje 3D printerima

Počeo sam instalirati operativni sustav i potrebne alate za izradu aplikacije za upravljanje 3D printerima nakon uspješnog spajanja Raspberry Pi na monitor pomoću HDMI kabela te spajanja miša i tipkovnice (slika 1). Zbog svoje prilagodljivosti, jednostavnosti i niske potrošnje resursa, Raspberry Pi OS - službeni operativni sustav za Raspberry Pi uređaje.



Slika 1. Prikaz Raspberry Pi sučelja

Instaliranje operativnog sustava Raspberry Pi bio je prvi korak. Koristeći aplikaciju Raspberry Pi Imager, preuzeta je najnovija verzija Raspberry Pi OS-a i snimljena je na SD karticu. Zatim je uključen Raspberry Pi umetanjem SD kartice u njega. Završio sam početno postavljanje sustava, što je uključivalo ažuriranje softverskih paketa i spajanje na obližnju Wi-Fi mrežu, nakon što se prvi put pokrenuo.

Instalacija potrebnih paketa započela je osvježavanjem repozitorija naredbom:

```
sudo apt update
sudo apt upgrade
```

Primjer 3. Prikaz instalacije potrebnih paketa

To jamči da su svi paketi sustava ažurni, što je ključno za perifernu kompatibilnost i stabilnost sustava. Zatim sam postavio Flask i Python koji su potrebni za pozadinsku aplikaciju. Raspberry Pi OS dolazi s unaprijed instaliranim Pythonom, ali u slučaju da

vam zatreba, možete ga instalirati ili ažurirati na najnoviju verziju sa sljedećim naredbama:

```
sudo apt install python3 sudo
apt install python3-pip
```

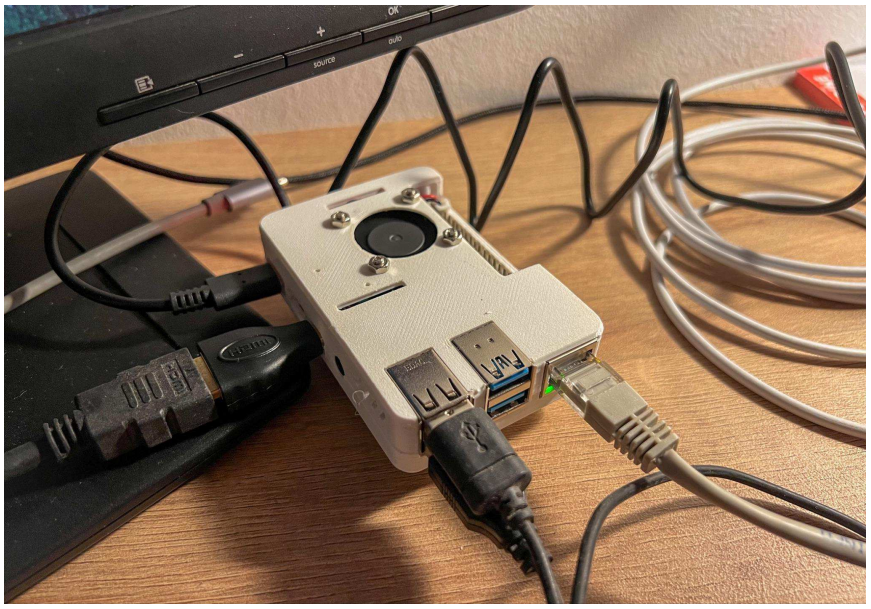
Primjer 4. Prikaz instalacije pythona

Flask, kao mikro web framework za Python, instaliran je pomoću `pip` upravitelja paketa:

```
pip3 install Flask
```

Primjer 5. Prikaz instalacije flask

3D pisač Prusa MK2S povezuje se s Raspberry Pi uređajem putem USB veze. Kako bi poslao GCode naredbe za kontrolu ispisa, Raspberry Pi upravlja serijskom komunikacijom s 3D printerima. USB hub koristi se za povećanje broja dostupnih USB priključaka na Raspberry Pi, što omogućuje povezivanje više pisača. To je zato što Raspberry Pi ima samo 4 USB priključka.



Slika 2. Prikaz Raspberry Pi uređaja

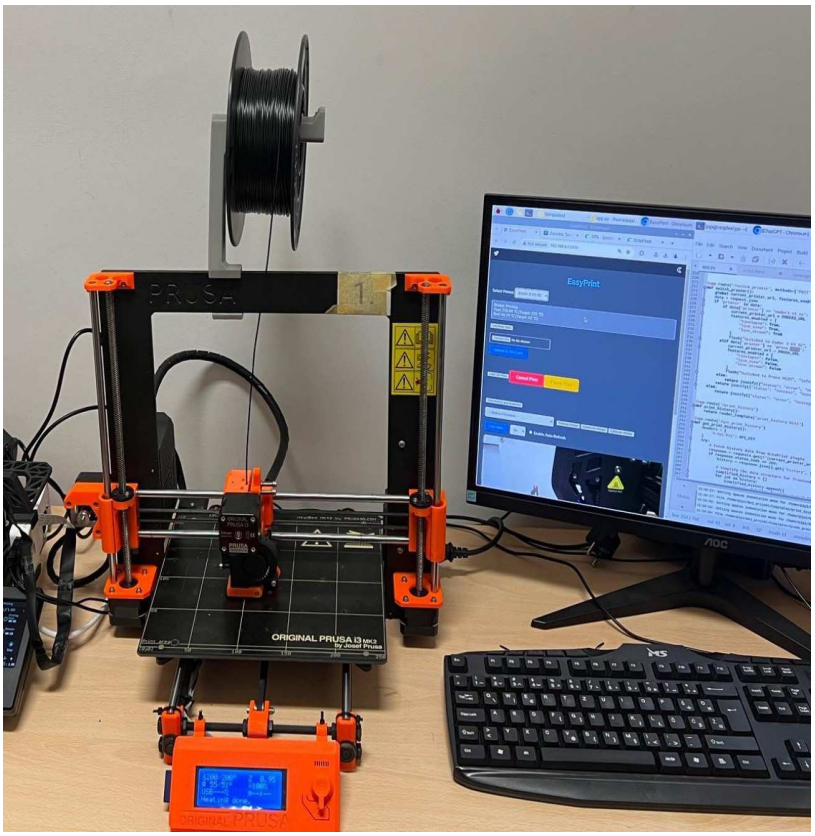
Jedan od najvažnijih koraka je dodijeliti korisniku Raspberry Pi-ja potrebna dopuštenja za pristup serijskim uređajima dodavanjem u grupu 'dialout'. Ova

konfiguracija je potrebna kako bi Raspberry Pi koristio serijski protokol za komunikaciju s 3D pisačima. Za uključivanje korisnika u 'dialout' grupu, koristite sljedeću naredbu:

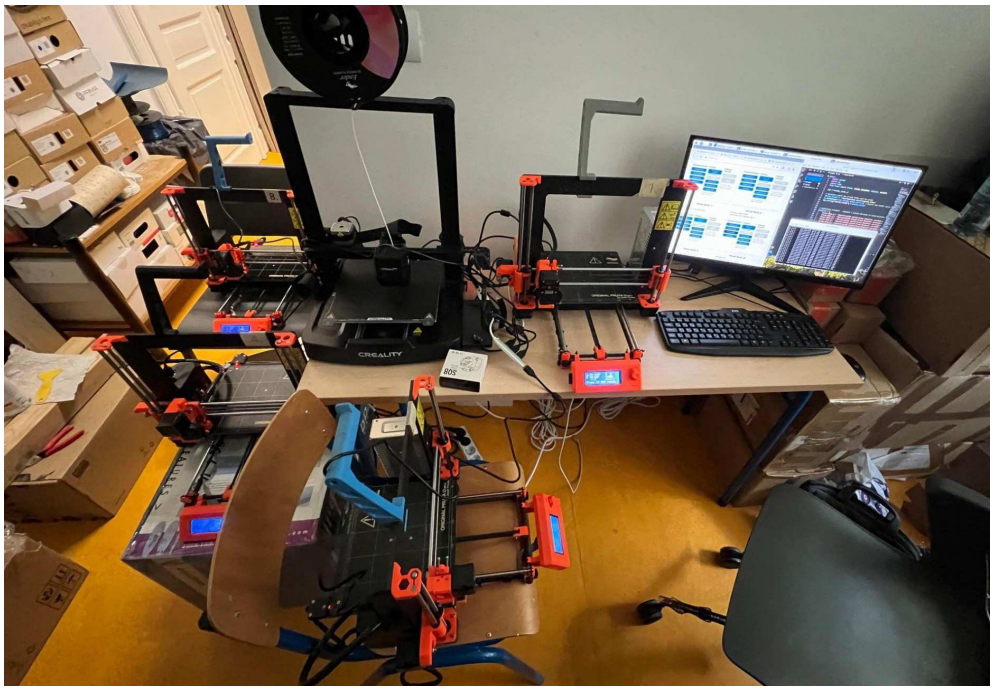
```
sudo usermod -aG dialout $USER
```

Primjer 6. Prikaz dodavanja dopuštenja za pristup

Da bi izmjene stupile na snagu, sustav se mora ponovno pokrenuti nakon pokretanja ove naredbe. S ovim konfiguracijama, svim povezanim 3D pisačima može se pristupiti i kontrolirati Raspberry Pi preko serijske veze. Raspberry Pi također je konfiguriran za daljinsko upravljanje, što znači da bilo koji uređaj na istoj mreži može pristupiti web aplikaciji koja prati i kontrolira sve povezane pisače. Korisnici mogu započeti nove ispise, mijenjati temperature ispisne glave i podloge te gledati ispise u stvarnom vremenu zahvaljujući ovoj funkcionalnosti. Aplikacija koristi USB serijsku komunikaciju za slanje odgovarajućih GCode naredbi svakom pisaču, uključujući Prusa MK2S, i povremeno ažurira status pisača u sučelju. Na primjer, naredbe M104 i M140 koriste se za postavljanje temperature glave pisača i podloge, a naredba G28 se koristi za automatsko vraćanje pisača u početni položaj (kalibracija). Skalabilno upravljanje nekoliko pisača iz jednog središnjeg sustava omogućeno je dodavanjem dodatnog USB čvorišta. Kada su upareni s ovim softverskim rješenjem, Raspberry Pi i Prusa MK2S pisač omogućuju korisniku kontrolu svakog aspekta procesa ispisa na daljinu putem online sučelja. To uključuje upravljanje nitima, praćenje statusa ispisa i daljinsku kalibraciju. Ova je infrastruktura idealna za proizvodne pogone ili laboratorije s više 3D pisača jer omogućuje optimizaciju tijeka rada te automatizaciju i praćenje ispisa na više uređaja odjednom. Dodatno, smanjuje potrebu za stalnom fizičkom prisutnošću korisnika, što poboljšava učinkovitost i smanjuje vrijeme zastoja. Korištenje takve konfiguracije također omogućuje jednostavno proširenje sustava bez značajnih izmjena u postojećem hardveru.



Slika 3. Prikaz jednog Prusa MK2S printera spojenog na aplikaciju



Slika 4. Prikaz nekoliko spojenih printera na aplikaciju

5. Struktura sustava

Frontend i backend su dvije primarne komponente EasyPrint sustava. Budući da se logika aplikacije i korisničko sučelje mogu odvojiti u ovoj arhitekturi, postoji veća fleksibilnost i modularnost. Nadalje, sustav je projektiran za prilagodbu brojnim 3D pisačima, što je ključna značajka za postavke kao što su laboratoriji ili proizvodne linije gdje se brojni uređaji koriste istovremeno.

Struktura projekta raspoređena je u nekoliko važnih direktorija i datoteka:

- **app.py**: Glavna Python datoteka koja pokreće Flask aplikaciju. U njoj se nalaze rute za API i komunikacija s 3D printerima.
- **templates/index.html**: HTML datoteka koja sadrži frontend korisničko sučelje aplikacije. Ovo je glavna web stranica koju korisnik vidi kada pristupi aplikaciji.
- **static/app.js**: JavaScript datoteka zadužena za dinamičke funkcionalnosti aplikacije, poput komunikacije s backendom i ažuriranja informacija na sučelju u stvarnom vremenu.
- **static/style.css**: CSS datoteka koja definira izgled aplikacije, uključujući stilove za gumbove, tablice te druge elemente sučelja.

Ova struktura datoteke u skladu je sa standardnom metodom za stvaranje web aplikacija s okvirom Flask, pri čemu HTML, CSS i JavaScript datoteka sadrži prednji kraj aplikacije, a Python datoteka logiku stražnjeg kraja.

5.1 Backend

Backend implementacija

Za komunikaciju preko serijskih portova s 3D printerima zadužen je backend aplikacije koji je kreiran s Pythonom i Flask frameworkom. GCode naredbe šalju se pisačima kako bi se olakšala komunikacija, omogućujući daljinsko upravljanje i praćenje procesa. U nastavku je objašnjenje glavnih dijelova koda koji omogućuju ovu funkcionalnost:

1. Uvoz potrebnih modula
   Uvozimo osnovne module potrebne za rad aplikacije:

- serial omogućuje serijsku komunikaciju s 3D printerima putem USB-a.
- os i time koriste se za rad s datotekama i vremenske operacije.
- Flask je framework koji omogućuje izradu web aplikacija, dok request i jsonify omogućuju obradu HTTP zahtjeva i vraćanje JSON odgovora.
- lru_cache omogućuje keširanje funkcija za optimizaciju performansi, dok se Thread i Queue koriste za upravljanje višezadaćnim procesima.

Konfiguracija aplikacije i priprema direktorija za GCode datoteke

Postavili smo direktorij za spremanje GCode datoteka koje se prenose kroz aplikaciju:

```
pip3 install Flask
 UPLOAD_FOLDER = '/home/pipi/moj_zavrsni/gcode_files'
os.makedirs(UPLOAD_FOLDER, exist_ok=True)
app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER
```

Primjer 7. Prikaz direktorija za spremanje GCode datoteke

Ova naredba dopušta učitavanje i spremanje GCode datoteka i provjerava da je direktorij prisutan.

Određivanje pisača u uporabi, da bi se pisači mogli povezati putem USB-a, moraju definirati svoje serijske brojeve:

```
printers = {
    'prusa_mk2s_1': '/dev/serial/by-id/usb-Prusa_Research...',
    ...
}
```

Primjer 8. Prikaz definiranja serijskih id-ijeva za svaki printer

Ovi serijski ID-ovi koriste se za uspostavu serijske veze s odgovarajućim printerom.

Upravljanje serijskim vezama s printerima

Funkcija get_serial_connection uspostavlja ili dohvaća postojeću serijsku vezu s

printerom:

```
def get_serial_connection(printer_id):
    serial_port = printers.get(printer_id)
    if serial_port:
        if printer_id     in   printer_connections          and
printer_connections[printer_id].is_open:
            return printer_connections[printer_id], None
        try:
         ser = serial.Serial(serial_port, 115200,
            timeout=60) time.sleep(2)
            printer_connections[printer_id] = ser
            return ser, None
        except serial.SerialException as e:
            return None, str(e)
    return None, "Printer not found"
```

Primjer 9. Prikaz funkcije get_serial_connection

2. Ova funkcija provjerava postoji li već otvorena serijska veza za određeni printer te je vraća ako je dostupna. Ako nije, uspostavlja novu vezu.

Da bi upravljanje više printera bilo efikasno, svaki printer ima svoj red čekanja za GCode naredbe:

```
    printer_command_queues = {} for
printer_id in printers:
    printer_command_queues[printer_id] = Queue()

def process_command_queue(printer_id):
    while True:
        command = printer_command_queues[printer_id].get() if
        command is None:
            break send_gcode(printer_id,
        command)
```

Primjer 10 . Prikaz GCode naredbe za efikasnost

3. Svaki printer ima poseban Thread koji obrađuje komande iz svog reda. To omogućava da se više printera kontrolira paralelno.

Slanje GCode naredbi printerima

Funkcija send_gcode šalje pojedinačne GCode naredbe printeru putem serijske veze:

```
def send_gcode(printer_id, gcode_command): ser,
error = get_serial_connection(printer_id) if ser:
    try:
        ser.write((gcode_command + '\n').encode()) ser.flush()
        return f"G-code sent: {gcode_command}" except
(serial.SerialException, OSError) as e:
       return f"Error sending G-code: {str(e)}"
else:
    return error or "Printer not found or disconnected"
```

Primjer 11. Prikaz slanja GCode naredbi

4. Keširanje statusa printera

   Da bi se smanjio broj poziva prema printerima, koristi se keširanje pomoću lru_cache:

```
@lru_cache(maxsize=10)
def fetch_printer_status_cached(printer_id): return
    get_printer_status(printer_id)
```

Primjer 12. Prikaz keširanja statusa printera

5. Dohvaćanje statusa printera

   Funkcija get_printer_status dohvaća informacije o trenutnom statusu ispisa, temperaturi ispisne glave i podloge:

```
def get_printer_status(printer_id):
    ser, error = get_serial_connection(printer_id) if
    ser:
        # Fetch print status, temperature, etc.
        ...
        return {
            'print_status': print_status,
         'hotend_temp': hotend_temp if hotend_temp else "N/A",
            'bed_temp': bed_temp if bed_temp else "N/A",
        }
```

Primjer 13. Prikaz dohvaćanja statusa printera

6. Prijenos i ispis GCode datoteka

   Korisnici mogu prenijeti GCode datoteku putem aplikacije i odmah započeti ispis:

```
        @app.route('/api/upload_gcode_and_print/<printer_id>',
methods=['POST'])
def upload_gcode_and_print(printer_id): if
    'file' not in request.files:
        return jsonify({'status': 'error', 'message': 'No file
provided'}), 400
    ...
    result = send_gcode_to_printer(printer_id, file_path)
    return jsonify(result)
```

Primjer 14. Prikaz prijenosa i ispis GCode datoteka

7. Glavne rute u aplikaciji

   Aplikacija ima nekoliko ruta koje omogućuju prijenos datoteka, slanje GCode naredbi i dohvaćanje statusa printera:

- /api/upload_gcode_and_print/<printer_id>: Prenosi GCode datoteku i šalje printeru na ispis.
- /api/control: Prima i šalje pojedinačne GCode naredbe.
- /api/status: Dohvaća trenutni status printera.
- /: Početna stranica aplikacije, prikazuje web sučelje za upravljanje.

Pokretanje aplikacije
Na kraju, aplikacija se pokreće na lokalnom poslužitelju:

```
if     name  == ' main ':
 app.run(host='0.0.0.0', port=5005)
```

Primjer 15. Prikaz pokretanja aplikacije

Uz pomoć ovog backenda, koji jamči pouzdanu i stabilnu komunikaciju s nekoliko 3D pisača, korisnici mogu slati GCode naredbe, kontrolirati proces ispisa putem online sučelja i pratiti status ispisa u stvarnom vremenu.

5.2. Frontend

U razvoju frontenda koriste se tehnologije HTML, CSS i JavaScript. Interakciju korisnika s aplikacijom u stvarnom vremenu omogućuje JavaScript, dok HTML nudi strukturu stranice, CSS stil i brzinu odziva.

Upravljačka ploča ili korisničko sučelje sastoji se od različitih bitnih komponenti:

- **Dashboard**: Glavni zaslon na kojem korisnici mogu pregledavati status svih povezanih 3D printera.
- **Upravljanje printerima**: Korisnici mogu birati aktivni printer, slati GCode naredbe i pratiti napredak ispisa.
- **Temperatura i kalibracija**: Mogućnost podešavanja temperature ispisne glave i podloge, te kalibracija printera.

Aplikacija se sastoji od nekoliko ključnih elemenata:

- **Tab navigacija**: Omogućuje korisnicima da prebacuju između različitih panela za upravljanje printerima (Control Panel, Calibration, itd.).
- **Kontrolni panel printera**: Sadrži informacije o svakom printeru i omogućuje korisnicima da kontroliraju pregrijavanje, ispis i slanje naredbi.
- **JavaScript funkcionalnosti**: Omogućuju dinamičku interakciju s backendom putem API-ja.

Primjer glavne strukture u HTML-u:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <link rel="stylesheet"    href="{{  url_for('static',
filename='style.css') }}">
    <script    src="{{   url_for('static', filename='app.js')
}}"></script>
    <title>EasyPrint - Control Your Printers</title>
</head>
<body>
    <h1>EasyPrint - 3D Printer Control</h1>

    <div class="tab">
        <button    class="tablinks"    onclick="openTab(event,
'ControlPanel')">Control Panel</button>
    </div>

    <!-- Control Panel Tab -->
    <div id="ControlPanel" class="tabcontent">
        <div id="control-panel-container" class="control-panel-
grid"></div>
    </div>
```

```
</body>
</html>
```

Primjer 16. Prikaz glavne strukture u HTML-u

- Element h1 za naslov stranice, gumb za promjenu kartice i ploča koja prikazuje kontrole 3D pisača među primarnim su elementima stranice koji se prikazuju u ovom primjeru HTML datoteke.

JavaScript funkcionalnosti

Omogućavanje dinamičkog sučelja koje koristi zahtjeve za dohvaćanje za komunikaciju s pozadinom zahtijeva JavaScript kod. Na primjer, naredbu GCode M27 koristi funkcija getPrinterStatus za dobivanje statusa odabranog pisača:

```
function getPrinterStatus() {
   const printerId = document.getElementById('printer-
select').value;

   fetch('/api/control', {
       method: 'POST',
       headers: { 'Content-Type': 'application/json' },
       body: JSON.stringify({ printer_id: printerId,
gcode_command: 'M27' })
   }).then(response => response.json())
     .then(data => {
         document.getElementById('printer-status').textContent
= `Status: ${data.response}`;
     });
}
```

Primjer 17. Prikaz JavaScript funkcionalnosti

Ova funkcija koristi API poziv na backend kako bi dohvatila status printera i prikazala ga na stranici.

U aplikaciji je omogućena provjera temperature ispisne glave i upravljanje filamentom. Primjer funkcije koja automatski provjerava temperaturu svakih 5 sekundi:

```
function checkHotendTemperature() {
   const printerIds = ['prusa_mk2s_1', 'prusa_mk2s_2', 'prusa_mk2s_3'];

   printerIds.forEach(printerId => {
       fetch('/api/control', {
           method: 'POST',
           headers: { 'Content-Type': 'application/json' },
           body: JSON.stringify({ printer_id: printerId, gcode_command: 'M105' }),
       }).then(response => response.json())
         .then(data => {
const hotendTemp = parseFloat(data.response.gcode_output.match(/T:(\d+\.?\d*)/)[1]);
             if (hotendTemp >= 190) {
                 enableFilamentButtons(printerId);
             } else {
                disableFilamentButtons(printerId);
             }
         });
   });
}

setInterval(checkHotendTemperature, 5000);
```

Primjer 18. Prikaz kontrole temperature i filamenta

Kada temperatura ispisne glave poraste iznad 190°C, ova funkcija šalje naredbu GCode pisaču M105, procjenjuje odgovor i uključuje gumbe za kontrolu filamenta.

Printanje - GCode datoteke mogu se učitati u pisače pomoću JavaScripta:

```
function startPrint(printerId) {

constfileInput=            document.getElementById(`file-input-
${printerId}`);
   const file = fileInput.files[0];
   if (!file) {
       alert('No file selected.');
       return;
   }
   const formData = new FormData();
   formData.append('file', file);
   fetch(`/api/upload_gcode_and_print/${printerId}`, {
       method: 'POST',
       body: formData
   }).then(response => response.json())
     .then(data => {
         if (data.status === 'success') {
            alert(`Print started for ${file.name}`);
         } else {
             alert(`Error starting print: ${data.message}`);
         }
}    });
```

Primjer 19. Prikaz funkcije printanja

Ova funkcija omogućava korisnicima da odaberu GCode datoteku, pošalje se na backend i započnu ispis na odabranom printeru.

CSS

Kako bismo zajamčili ugodno i jednostavno korisničko iskustvo u aplikaciji EasyPrint, CSS je neophodan. Osim standardnog oblikovanja elemenata kao što su gumbi, tablice i kartice, CSS također čini aplikaciju osjetljivom na različite zaslone i uređaje.

1. Struktura CSS datoteka

Izgled kartica, kartica pisača, gumba, obavijesti i modala među glavnim su kategorijama stila aplikacije. Ovo je ilustracija kako je stil organiziran. CSS datoteka:

```
   /* Stilizacija tabova */
.tab {
    overflow: hidden;
    background-color: #f1f1f1;
}
.tab button {
    background-color: inherit;
    color: black;
    float: left; border:
    none; outline: none;
    cursor: pointer;
    padding: 14px 16px;
    transition: 0.3s;
    font-size: 17px;
}
/* Stilizacija aktivnog taba */
.tab button.active {
    background-color: #ccc;
}
```

Primjer 20. Prikaz strukture CSS-a

Kartice: imaju osnovne efekte lebdenja, prijelaze boja i mogućnost prebacivanja između panela aplikacije.

2. Arhitektura mreže i odziv

Izgled rešetke, koji omogućuje prikaz pisača u mreži s dva stupca na većim zaslonima, dok se aplikacija prebacuje na jedan stupac radi bolje vidljivosti na manjim zaslonima, jedna je od glavnih značajki CSS-a u ovoj aplikaciji.

```
  /* Grid layout za kontrolni panel */
.control-panel-grid {
    display: grid;
   grid-template-columns: repeat(2, 1fr); /* Dva stupca na većim
ekranima */
    grid-gap: 20px;
    max-width: 1200px;
    margin: 0 auto;
}

/* Stilizacija kartice printera */
.printer-card {
    background-color: #f1f1f1;
    border-radius: 10px;
    padding: 20px;
    box-shadow: 0px 4px 8px rgba(0, 0, 0, 0.1);
    display: flex;
    flex-direction: column;
}
```

Primjer 21. Prikaz CSS-a za mobilno sučelje

Dizajn rešetke: stupci rešetke - predloška omogućuju nam da odredimo izgled pisača u dva stupca na većim zaslonima, a razmak mreže povećava udaljenost između kartica.

Kartice za pisače: Svaka je kartica vizualno privlačna i razlikuje se od ostalih elemenata zahvaljujući zaobljenim rubovima, sjeni i podstavi.

Pravila medijskog upita koriste se za optimizaciju aplikacije za manje uređaje. Na primjer, aplikacija se prebacuje na jednostavniji izgled u jednom stupcu za uređaje čija je širina zaslona manja od 768 piksela.

```
   /* Responzivni dizajn za manje ekrane */
@media only screen and (max-width: 768px) {
   .control-panel-grid {
       grid-template-columns: 1fr; /* Jedan stupac na manjim
ekranima */
       grid-gap: 10px;
   }

   .printer-card {
      width: 100%; /* Kartice zauzimaju cijelu širinu ekrana */
   }

   button {
       padding: 6px; /* Smanjuje padding na gumbima za mobilne
uređaje */
       font-size: 14px; /* Manja veličina fonta */
   }
}
```

Primjer 22. Prikaz CSS-a za responzivnost

Ovo pravilo mijenja veličine i izglede elemenata kako bi se osiguralo da je aplikacija i dalje čitljiva i funkcionalna na mobilnim uređajima.

Budući da korisnicima omogućuju upravljanje pisačima, gumbi su bitne komponente aplikacije. Svaki gumb ima jednostavan stil s prijelazima i efektima lebdenja.

```
button {
    background-color: #007BFF;
    color: white;
    padding: 10px;
    margin: 5px; border:
    none; border-radius:
    5px; cursor:
    pointer;
}

button:hover {
    background-color: #0056b3;
}

button:disabled {
    background-color: #ccc;
    cursor: not-allowed;
}
```

Primjer 23. Prikaz CSS-a za stil gumba

Hover (Efekt) lebdenja: Boja pozadine gumba mijenja se kako bi pokazala da je aktivan kada korisnik prijeđe pokazivačem iznad njega. U onemogućenom stanju, kursor gumba se mijenja, a pozadina posvjetljuje kako bi se pokazalo da nije u ispravnom stanju. Svaki pisač dolazi s karticom koja sadrži kontrole za punjenje, pregrijavanje i upravljanje ispisom. Ove kontrole su raspoređene u tri stupca unutar svake kartice.

```
   .printer-controls-grid { display:
    grid;
    grid-template-columns: repeat(3, 1fr);
    grid-gap: 20px;
    margin-top: 20px;
}

.control-column {
    display: flex;
    flex-direction: column;
}
```

Primjer 24 Prikaz CSS-a za kartice printera

Ovaj raspored omogućuje jasan i uredan prikaz kontrola, olakšavajući korisnicima interakciju s različitim funkcijama printera.

Upravljanje višestrukim printerima

Mogućnost upravljanja više pisača jedna je od primarnih značajki aplikacije. Korisnici se mogu brzo prebacivati između različitih pisača, provjeravati njihove statuse i slati im određene GCode naredbe pomoću sučelja.

Svaki pisač opremljen je jedinstvenom krajnjom točkom, a dinamička ažuriranja podataka u stvarnom vremenu postižu se putem API poziva. Za promjenu aktivnog pisača, na primjer, upotrijebite sljedeći API poziv:

```
<!-- Control Panel Tab -->
<div id="ControlPanel" class="tabcontent">
    <h2></h2>
    <div id="control-panel-container" class="control-panel-
grid">
 <!-- Printer 1 card -->
<div class="printer-card">
    <h3>PRUSA MK2S_1</h3>
```

```
    <div class="printer-status" id="status-prusa_mk2s_1">
        <p>Print Status: Loading...</p>
        <p>Hotend Temp: N/A °C | Bed Temp: N/A °C</p>
    </div>
    <div class="control-columns-container">
       <div class="control-column">
            <h4>Preheat/Cooldown</h4>
            <button            onclick="preheat('prusa_mk2s_1',
'PLA')">Preheat PLA</button>
            <button            onclick="preheat('prusa_mk2s_1',
'ABS')">Preheat ABS</button>
            <button            onclick="preheat('prusa_mk2s_1',
'PET')">Preheat PET</button>
            <button
onclick="cooldown('prusa_mk2s_1')">Cooldown</button>
            <button             id="load-filament-prusa_mk2s_1"
onclick="loadFilament('prusa_mk2s_1')" >Load Filament</button>
            <button           id="unload-filament-prusa_mk2s_1"
onclick="unloadFilament('prusa_mk2s_1')"               >Unload
Filament</button>
            <p><small>The hotend nozzle must be over 190°C to
enable these buttons.</small></p>
        </div>
        <div class="control-column">
            <h4>Calibration</h4>
            <button     onclick="autoHome('prusa_mk2s_1')">Auto
Home</button>
            <button
onclick="meshBedLeveling('prusa_mk2s_1')">Mesh              Bed
Leveling</button>
            <button
onclick="resetXYZCalibration('prusa_mk2s_1')">Reset         XYZ
Calibration</button>
        </div>
        <div class="control-column">
            <h4>Printing</h4>
            <input   type="file"   id="file-input-prusa_mk2s_1"
accept=".gcode" onchange="handleFileSelect('prusa_mk2s_1')"/>
            <label  for="file-input-prusa_mk2s_1" style="cursor:
pointer;">Choose File (.gcode)</label>
            <span  id="file-name-display-prusa_mk2s_1">No  file
selected</span>
```

```
            <button onclick="startPrint('prusa_mk2s_1')">Start
Print</button>
            <button onclick="pausePrint('prusa_mk2s_1')">Pause
Print</button>
            <button
onclick="resumePrint('prusa_mk2s_1')">Resume Print</button>
            <button
onclick="cancelPrint('prusa_mk2s_1')">Cancel Print</button>
        </div>
    </div>
</div>
```

Primjer 25. Prikaz koda za control panel

U primjeru 25 aplikacija definira kontrolnu karticu (printer karticu) kojih ima deset. Sada pogledajmo pobliže ovaj kod. Samo jedan od deset blokova kartica - jedan za svaki jedinstveni 3D printer - definiran je u ovom kodu. Ova kartica sadrži postavke za pisač uz koji se nalazi. Dostupni su i detalji poput statusa ispisa i temperature podloge i vruće glave. Svaka kartica ima tri kontrolna stupca, a to su: predgrijavanje/hlađenje, kalibracija i printanje.

1. Prethodno zagrijavanje/hlađenje: Ovaj odjeljak omogućuje upravljanje nitima (umetanje i izbacivanje), kao i pregrijavanje ili hlađenje pisača u različitim materijalima (PLA, ABS, PET). Gumbi koji pozivaju JavaScript metode, poput cooldown za hlađenje i preheat za predgrijavanje, pružaju funkcionalnost ako je temperatura hotenda viša od 190C.

2. Kalibracija: pisač se može kalibrirati pomoću gumba u ovom odjeljku za Auto Home, Mesh Bed Leveling i Reset XYZ Calibration. Za kontrolu kalibracije pisača, svaka od ovih operacija poziva posebnu JavaScript funkciju.

3. Printanje: Ovdje se upravlja postupkom ispisa. Pomoću elementa unosa <input type="file"> korisnik može odabrati datoteku a.gcode(upute za 3D ispis). Kada je datoteka odabrana, funkcija handleFileSelect('prusa_mk2s_1') utvrđuje je li odabrana ispravna vrsta datoteke i ako je sve u redu, aktivira gumb za početak ispisa. Korištenjem funkcija kao što su startPrint i cancelPrint pokreće se ili se prekida proces printanja, uz to imamo još i gumb pauziranje i ponovo kretanje ispisa.

6. Implementacija

6.1. Detaljan prikaz koda

Detaljno objašnjenje koda, koji uključuje sve značajke obrađene u prethodnim odjeljcima, može se pronaći u nastavku. Ovaj se kod sastoji od prednjeg koda za upravljanje pisačima i prikazivanje statusa, kao i API ruta i komunikacije pisača.

Frontend kod (HTML/CSS/JavaScript)

Frontend softver olakšava rad i upravljanje 3D pisačima s jednog uređaja. Osvježavanje podataka pisača u stvarnom vremenu omogućeno je pod JavaScriptom.

```
      // JavaScript funkcija za otvaranje tabova
        function openTab(evt, tabName) {
            var i, tabcontent, tablinks;
            tabcontent =
document.getElementsByClassName("tabcontent");
            for (i = 0; i < tabcontent.length; i++) {
                tabcontent[i].style.display = "none";
          }
            tablinks =
document.getElementsByClassName("tablinks");
            for (i = 0; i < tablinks.length; i++) {
                tablinks[i].className =
tablinks[i].className.replace(" active", "");
            }
            document.getElementById(tabName).style.display =
"block";
            evt.currentTarget.className += " active";
        }
```

```
        document.addEventListener('DOMContentLoaded', () => {
    // Open the "Control Panel" tab by default when the page
loads
    const controlPanelButton =
document.querySelector('button[onclick="openTab(event,
\'ControlPanel\')"]');
    if (controlPanelButton) {
        controlPanelButton.click(); // Simulate a click to open
the Control Panel tab
    }
```

Primjer 26. Prikaz Javascript funkcije za otvaranje tabova

U primjeru 3 imamo značajku koja korisnicima omogućuje navigaciju između različitih odjeljaka online aplikacije. Ova značajka je zadužena za skrivanje nekih kartica i otvaranje drugih. Funkcija openTab omogućuje jednostavnu navigaciju između kartica u sučelju, skrivanje neaktivnih kartica i prikaz samo aktivnih. Kada se stranica učita, kartice upravljačke ploče otvaraju se automatski, štedeći korisnika od potrebe da ručno biraju karticu i omogućujući mu da odmah vidi najvažniji dio sučelja.

```
function updatePrinterStatuses() {
    const selectedPrinters = getSelectedPrinters();
    const statusContainer = document.getElementById('status-
container');
    statusContainer.innerHTML = ''; // Clear current statuses

    if (selectedPrinters.length === 0) {
        statusContainer.innerHTML = '<div>No printers
selected</div>'; // Handle no selection case
        return;
    }
```

```
    selectedPrinters.forEach(printerId => {
        // Create a status element for each selected printer
        const statusItem = document.createElement('div');
        statusItem.classList.add('status-item');
        statusItem.id = `status-${printerId}`;
        statusItem.innerHTML =
`<strong>${printerId.replace('_', ' ').toUpperCase()}:</strong>
Loading status...`;
        statusContainer.appendChild(statusItem);

        // Fetch and display the actual status
        fetchPrinterStatus(printerId);
    });
}
```

Primjer 27. Prikaz koda za status svakog printera

Prikaz statusa za svaki od odabranih pisača ažurira se funkcijom updatePrinterStatuses(). Ako nema odabranih pisača, prikazuje se obavijest da nijedan pisač nije odabran. Za svaki odabrani pisač kreira se novi element, koji označava da se status privremeno učitava dok se stvarni status ne dobije pomoću funkcije fetchPrinterStatus().

6.2. Komunikacija između frontend-a i backend-a

Bez potrebe za osvježavanjem stranice, frontend dinamički ažurira podatke na stranici slanjem zahtjeva backendu putem AJAX tehnologije. Praćenje statusa u stvarnom vremenu i brzo, učinkovito upravljanje brojnim pisačima omogućeno je ovom komunikacijskom metodom.

Za slanje GCode naredbi, podešavanje temperature ili promjenu aktivnog pisača, sučelje šalje HTTP POST zahtjeve pozadinskom API-ju. Ove zahtjeve obrađuje backend, koji također koristi serijsko sučelje za povezivanje s 3D pisačima.

6.3. Funkcionalnosti aplikacije

Kalibracija pisača

Točni ispisi mogu se dobiti samo ispravnom kalibracijom pisača. Prije početka ispisa, svaka funkcija kalibracije igra određenu ulogu u osiguravanju ispravnog rada pisača.

Auto Home (automatska kalibracija):

Ova značajka uzrokuje da se pisač vrati u početni položaj na sve tri svoje osi (X, Y i Z) pokretanjem naredbe G28. Ovo je standardni radni postupak za prepoznavanje lokacije pisača i jamstvo da ispis počinje na pravom mjestu.

G28 pisač označen s printerId prima naredbu GCode od funkcije sendGcode(printerId, 'G28'). Dok addNotification(printerId, 'Auto Home executed.') omogućuje korisniku da vidi povratne informacije u korisničkom sučelju, potvrđujući uspješan početak kalibracije, ova jednostavna naredba pokreće proces kalibracije.

```
function autoHome(printerId) {
    sendGcode(printerId, 'G28'); // Naredba za automatsku
kalibraciju
    addNotification(printerId, 'Auto Home executed.'); //
Prikaz obavijesti korisniku
}
```

Primjer 28. Prikaz JavaScript funkcije za pokretanje Auto Home kalibracije

Mesh Bed Leveling:

- Ova funkcija automatski izravnava podlogu pisača pomoću kombinacije naredbi G80 i G29. Ove naredbe omogućuju precizan ispis i jamče da je podloga na koju ispisujete ravna.

- Dok G29 provodi stvarni proces izravnavanja, naredba G80 pokreće proces.

```
function meshBedLeveling(printerId) {
    sendGcode(printerId, 'G80'); // Pokreće proces niveliranja
    sendGcode(printerId, 'G29'); // Izvršava niveliranje
    addNotification(printerId, 'Mesh Bed Leveling in
progress.');
}
```

Primjer 29. Prikaz JavaScript funkcije za Mesh Bed Leveling

Resetiranje XYZ kalibracije:

Korisnik može brzo resetirati kalibraciju pisača u slučaju problema pomoću jednostavne naredbe GCode M502, koja vraća postavke kalibracije pisača na tvorničke vrijednosti.

```
function resetXYZCalibration(printerId) { sendGcode(printerId,
    'M502'); // Naredba za resetiranje
kalibracije
    addNotification(printerId, 'XYZ Calibration reset.');
}
```

Primjer 30. Prikaz JavaScript funkcije za resetiranje XYZ kalibracije

- Korisnik otvara aplikaciju i odabire printer na dashboardu.
- Klikom na "Kalibriraj" (postavlja se G28 naredba), printer se automatski kalibrira.
- Aplikacija vraća povratnu informaciju o uspješnosti kalibracije.
- Korisnik unosi željene temperature ispisne glave i podloge.
- Klikom na "Postavi temperature", GCode naredbe M104 i M140 se šalju printeru.

Upravljanje temperaturom

- Za uspješan ispis različitih materijala ključna je kontrola temperature ispisne glave i podloge. Svaki materijal ima određenu minimalnu i maksimalnu temperaturu.

Predgrijavanje ili podešavanje temperature:

- Što radi: Aplikacija će automatski prilagoditi temperaturu ispisne glave i podloge nakon što korisnik odabere vrstu materijala (kao što je PLA ili ABS). M140 se koristi za podlogu, a M104 se koristi za podešavanje temperature ispisne glave.
- Uvjetni blok if-else koristi se za postavljanje temperatura. Odlučuje koju će temperaturu postaviti na temelju materijala koji korisnik odabere. Funkcija sendGcode zatim se koristi od strane funkcije za slanje relevantnih GCode naredbi.

```
function preheat(printerId, material) { let
hotendTemp, bedTemp;
if (material === 'PLA') {
    hotendTemp = 215;
    bedTemp = 60;
} else if (material === 'ABS') {
    hotendTemp = 240;
    bedTemp = 100;
}
sendGcode(printerId, `M104 S${hotendTemp}`); // Postavlja
temperaturu ispisne glave
sendGcode(printerId, `M140 S${bedTemp}`);   // Postavlja
temperaturu podloge
}
```

Primjer 31 Prikaz funkcije za predzagrijavanje

Provjera trenutnih temperatura (Check Hotend Temperature):

- Pomoću naredbe M105 aplikacija povremeno mjeri temperaturu ispisne glave. Korisnik može puniti i izbacivati filament ako je temperatura viša od 190°C.

- Aplikacija šalje zahtjev pozadini, koja upravlja serijskom komunikacijom s pisačem, putem API-ja za dohvaćanje. Nakon primitka rezultata, temperatura se izuzima iz odgovora, a funkcije za žarnu nit su omogućene ili onemogućene na temelju temperature u tom trenutku.

```
      function checkHotendTemperature() { const
    printerIds = [...]; // Popis printera
    printerIds.forEach(printerId => {
        fetch('/api/control', {
            method: 'POST',
            headers: {'Content-Type': 'application/json'},
            body: JSON.stringify({ printer_id: printerId,
gcode_command: 'M105' }) // Naredba za provjeru temperature
        }).then(response => response.json())
          .then(data => {
              const hotendTemp =
parseFloat(data.response.gcode_output.match(/T:(\d+\.?\d*)/)[1]
);
              if (hotendTemp >= 190) {
                  enableFilamentButtons(printerId);
              }
          });
    });
}
```

Primjer 32. Prikaz funkcije za provjeru temperature

Učitavanje i upravljanje filamentom

Učitavanje i izbacivanje filamenta omogućuje korisnicima mijenjanje materijala tijekom procesa ispisa. Međutim, ovo je moguće samo ako je ispisna glava dovoljno zagrijana.

Učitavanje filamenta:

- Koristi naredbu M701 za učitavanje filamenta. Ova funkcija pokreće proces umetanja novog filamenta u glavu za ispis.
- Kako biste namjestili pisač da fizički uvuče filament u glavu za ispis, koristite naredbu M701. Dodatno, funkcija obavještava korisnika kada je filament uspješno umetnut.

```
        function loadFilament(printerId) {
    sendGcode(printerId, 'M701'); // Naredba za učitavanje
filamenta
    addNotification(printerId, 'Filament loaded');
}
```

Primjer 33. Prikaz funkcije za učitavanje filamenta

Izbacivanje filamenta:

Izbacuje filament pomoću naredbe M702, koja izbacuje filament iz ispisne glave.

- Proces je gotovo identičan kao za učitavanje, ali koristi se naredba M702 za fizičko izbacivanje filamenta iz ispisne glave.

```
        function unloadFilament(printerId) { sendGcode(printerId,
    'M702'); // Naredba za izbacivanje
filamenta
    addNotification(printerId, 'Filament unloaded');
}
```

Primjer 34 Prikaz funkcije za izbacivanje filamenta

Upload i ispis GCode datoteka

Funkcionalnost za upload i slanje GCode datoteka omogućava korisnicima slanje svojih dizajna printeru. Upload GCode datoteka i slanje printeru:

- Aplikacija omogućuje korisniku da odabere datoteku a.gcode sa svog računala, koja se zatim šalje u backend, pohranjuje na poslužitelju i šalje serijskom komunikacijom na pisač.
- Datoteka se može poslati u formatu multipart/form-data pomoću funkcije dohvaćanja. Nakon toga, backend koristi serijsku komunikaciju za slanje datoteke, red po red, na pisač iz direktorija na poslužitelju.

```
        function startPrint(printerId) {
const fileInput = document.getElementById(`file-input-
${printerId}`);
const file = fileInput.files[0]; if
    (!file) {
        alert('No file selected.');
        return;
    }
    const formData = new FormData();
    formData.append('file', file);

    fetch(`/api/upload_gcode_and_print/${printerId}`, { method:
        'POST',
        body: formData
    }).then(response => response.json())
      .then(data => {
          if (data.status === 'success') {
              addNotification(printerId, `Print started for
file: ${file.name}`);
          } else {
              alert(`Failed to print: ${data.message}`);
          }
      });
}
```

Primjer 35. Prikaz funkcije za pokretanje ispisa

Na backendu, Flask procesira datoteku, sprema je i koristi funkciju send_gcode_to_printer za slanje printeru:

```
   @app.route('/api/upload_gcode_and_print/<printer_id>',
methods=['POST'])
def upload_gcode_and_print(printer_id): if
    'file' not in request.files:
        return jsonify({'status': 'error', 'message': 'No file
provided'}), 400

    file = request.files['file'] if
    file.filename == '' ili not
file.filename.lower().endswith('.gcode'):
        return jsonify({'status': 'error', 'message': 'Invalid
file format'}), 400

    file_path = os.path.join(UPLOAD_FOLDER, file.filename)
    file.save(file_path)

    result = send_gcode_to_printer(printer_id, file_path)
    return jsonify(result)
```

Primjer 36. Prikaz funkcije send_gcode_to_printer za slanje printeru

Prikaz statusa printera u stvarnom vremenu

Praćenje statusa printera omogućuje korisnicima da vide trenutni status ispisa i temperature.

Periodičko dohvaćanje statusa printera:

- Povremeno aplikacija šalje naredbe M105 za dohvaćanje trenutne temperature i M27 za provjeru statusa ispisa.

- Svakih nekoliko sekundi ova značajka automatski dohvaća i prikazuje korisniku najnoviji status pisača koristeći setInterval.

```
    function getPrinterStatus() {
    const printerId = document.getElementById('printer-
select').value;
    sendGcode(printerId, 'M27'); // Naredba za dohvaćanje statusa
ispisa

    fetch('/api/control', { method:
        'POST',
        headers: {'Content-Type': 'application/json'},
        body: JSON.stringify({ printer_id: printerId,
gcode_command: 'M27' })
    }).then(response => response.json())
      .then(data => {
          document.getElementById('printer-status').textContent
= `Status: ${data.response}`;
      });
}
```

Primjer 37. Prikaz funkcije statusa printera u stvarnom vremenu

Aplikacija pruža povratne informacije korisniku o stanju ispisa u stvarnom vremenu, omogućujući bolje praćenje svakog zadatka.

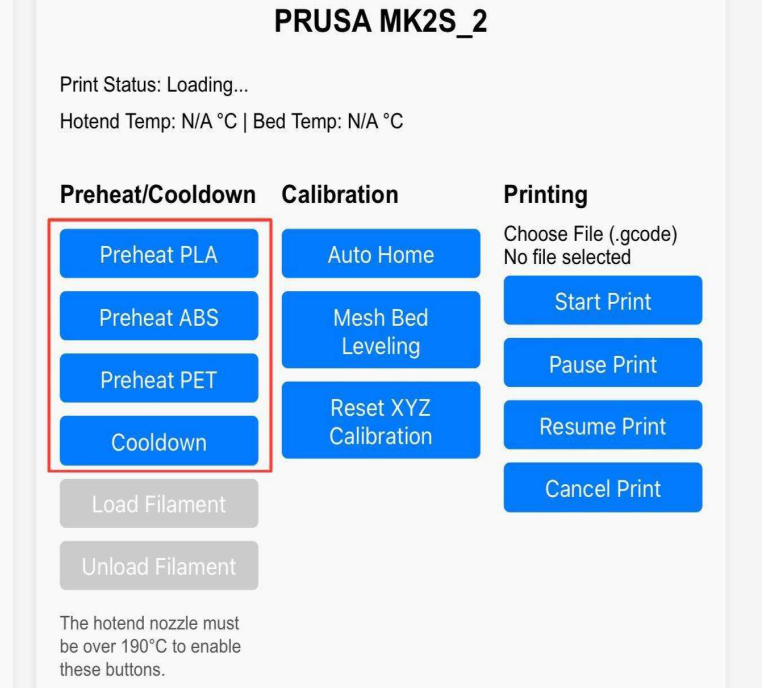
7. Upute za korisnika sučelja EasyPrint

Kroz sučelje aplikacije EasyPrint, koja se koristi za upravljanje 3D printerima, dostupno je nekoliko funkcija. U nastavku je objašnjena funkcija svakog gumba i što se događa kada ga kliknete:

7.1. Preheat/Cooldown (Postavljanje temperature)

- PLA: pritiskom na ovaj gumb, temperature za ispis PLA materijala su 215°C za ispisnu glavu i 60°C za supstrat.

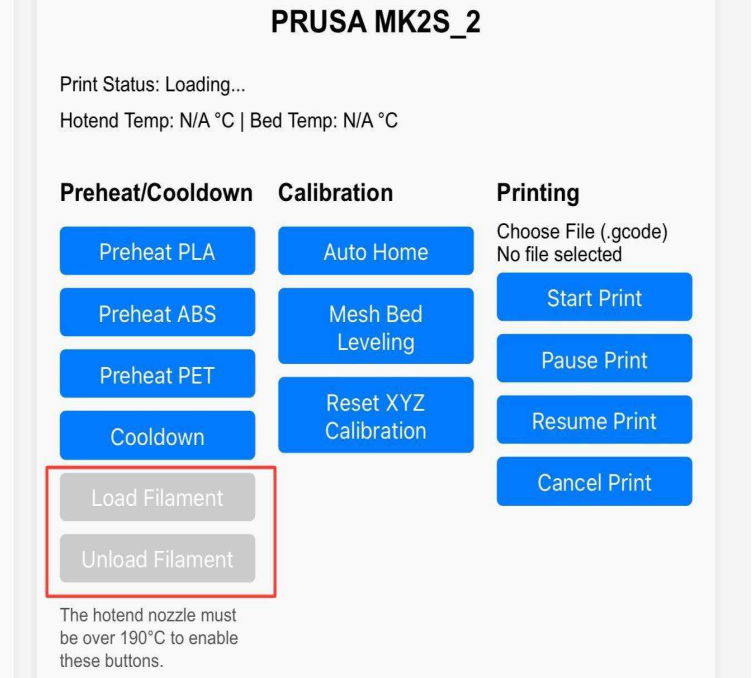
- ABS: glava za ispis treba biti zagrijana na 240°C, a podloga na 100°C.
- PET: se zagrijava na temperaturu od 90°C na podlozi i 230°C na glavi za ispis tijekom procesa predgrijavanja.
- Hlađenje: Snižava temperaturu ispisne glave i podloge na 0°C. Koristi se kada želite zaustaviti proces zagrijavanja ili nakon završetka ispisa.



Slika 5. Prikaz sučelja za postavljanje temperature

7.2. Load/Unload Filament (Upravljanje filamentom)

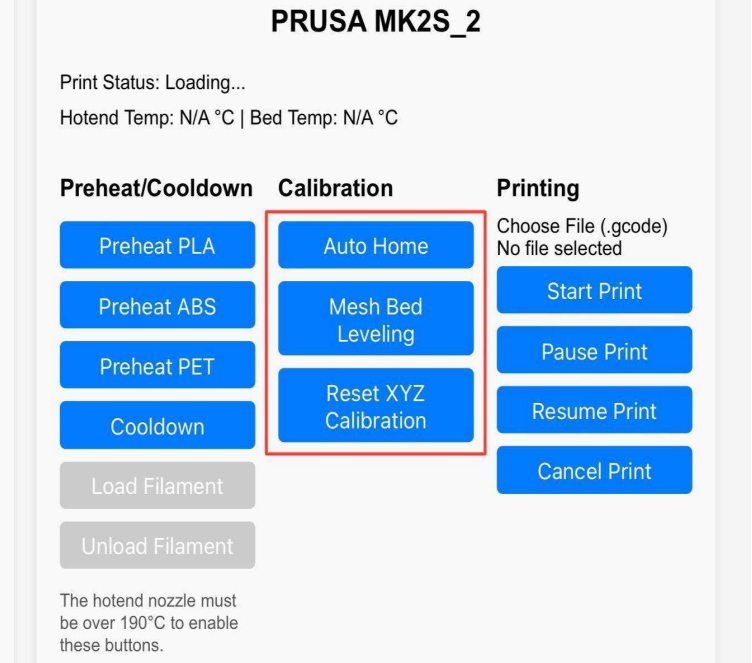
- Load Filament: Ovaj gumb pokreće naredbu M701 koja ubacuje nit u glavu za ispis. Dostupan je samo kada temperatura ispisne glave poraste iznad 190°C.
- Unload Filament: Naredba M702 koristi se za uklanjanje niti iz ispisne glave. Osim toga, ovaj se gumb može koristiti samo kada je temperatura glave pisača iznad 190°C.



Slika 6. Prikaz sučelja za upravljanje filamentom

7.3. Calibration (Kalibracija)

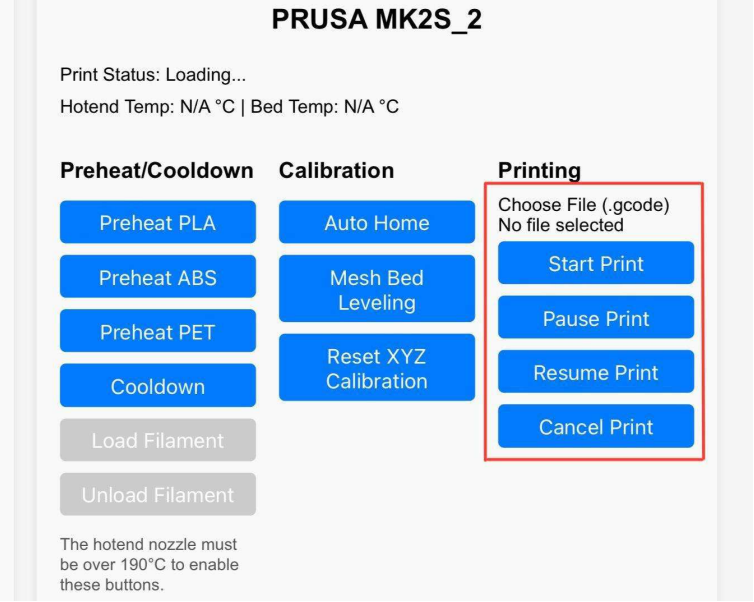
- Auto Home: Pritiskom na ovaj gumb pomaknut će se osi X, Y i Z pisača natrag u njihove početne položaje (naredba G28). Poduzimanje ovog ključnog koraka zahtijeva provjeru da je pisač svjestan svog ispravnog početnog položaja.
- Mesh Bed Leveling: pomoću naredbi G80 i G29, ova opcija automatski pokreće proces kalibracije. To jamči da je podloga ravna i pripremljena za točan ispis.
- Reset XYZ Calibration: Naredba M502 koristi se za vraćanje kalibracije XYZ osi na početne tvorničke postavke. Kada su potrebne nove postavke kalibracije, resetiranje kalibracije XYZ je najbolja opcija.



Slika 7. Prikaz sučelja za kalibraciju

7.4. Printing (Upravljanje ispisom)

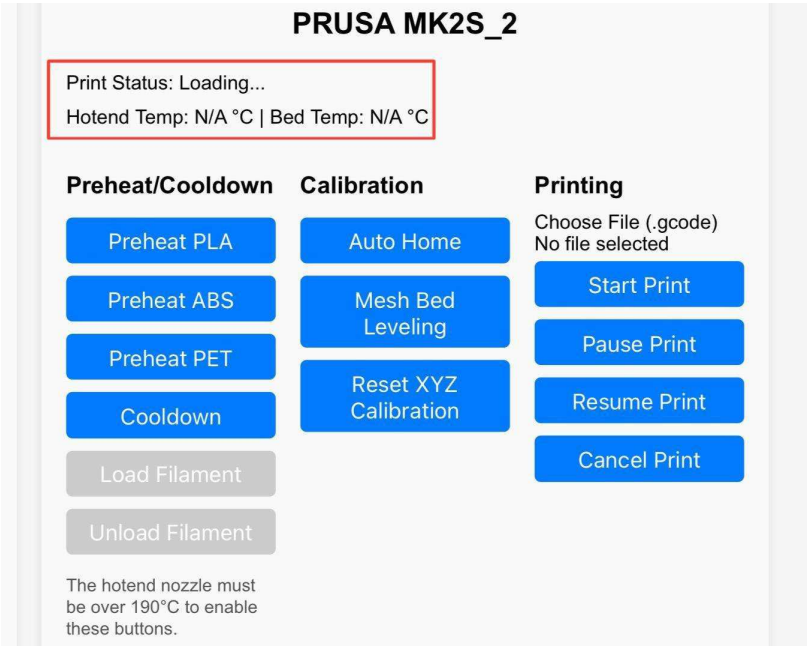
- Odaberite datoteku (.gcode): ova funkcija omogućuje korisniku da odabere GCode datoteku iz svog lokalnog sustava. Naziv datoteke pojavit će se u sučelju nakon što je odaberete.
- Start Print: pritisnite ovaj gumb za početak ispisa na odabranom pisaču nakon odabira datoteke .gcode.
- Pause Print: pauzira aktivni ispis.
- Resume Print: nastavlja ispis koji je pauziran.
- Cancel Print: prekida ispis i više ga nije moguće nastaviti



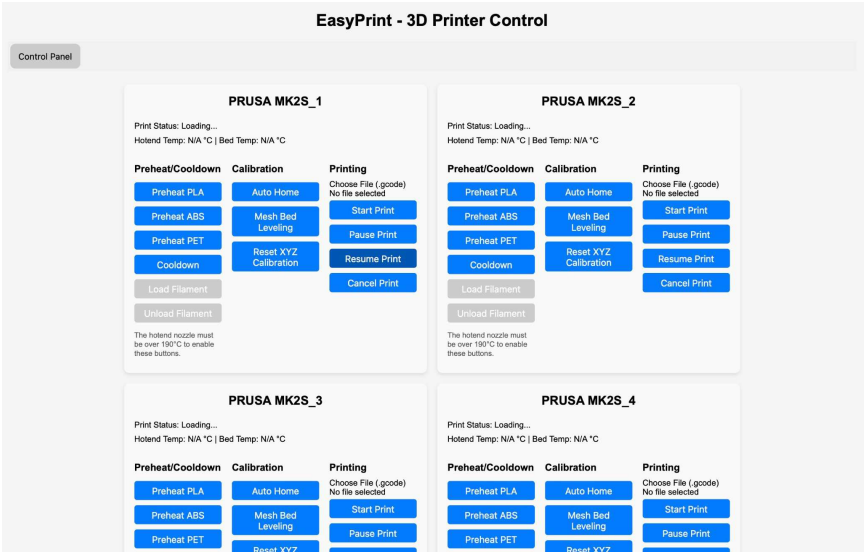
Slika 8. Prikaz print sučelja

7.5. Prikaz statusa printera

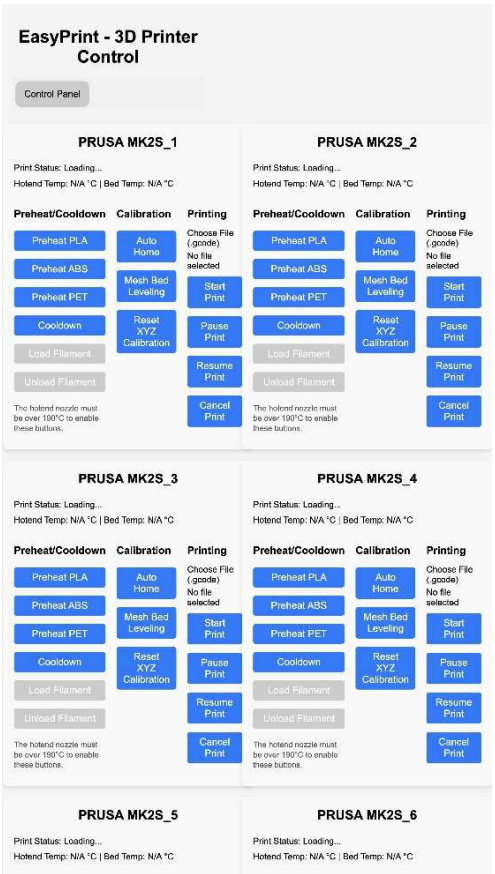
- Svaki pisač ima prikaz status ispisa (npr., "Status ispisa: Učitavanje...") i temperaturu podloge i ispisne glave ("Hotend Temp: N/A°C | Bed Temp: N/A°C") . Korisniku se prikazuju automatski ažurirane informacije.



Slika 9 Prikaz statusa printera



Slika 10. Prikaz desktop sučelja aplikacije



Slika 11 Prikaz mobilnog sučelja aplikacije

8. Zaključak

Razvoj aplikacije EasyPrint za daljinsko upravljanje laboratorijskim 3D pisačima, temeljen na tehnologijama poput Raspberry Pi 4, Python, Flask, i GCode, predstavlja značajan doprinos u optimizaciji proizvodnih i obrazovnih procesa koji uključuju više 3D pisača. Kroz ovaj projekt demonstrirano je kako se jednostavnim, ali efikasnim alatima može značajno unaprijediti kontrola nad 3D pisačima te smanjiti potreba za fizičkim nadzorom nad svakim uređajem pojedinačno. Time se olakšava rad u laboratorijima, školama, te manjim i većim proizvodnim pogonima, gdje se često koristi više uređaja istovremeno.

Jedan od ključnih doprinosa ovog rješenja je skalabilnost – sustav je jednostavno proširiv na više pisača te se lako može prilagoditi specifičnim potrebama korisnika. Zahvaljujući korištenju centralnog uređaja, poput Raspberry Pi 4, koji služi kao poveznica između različitih 3D pisača i korisnika, omogućeno je centralizirano praćenje i kontrola procesa ispisa. Daljinska kontrola omogućava ne samo nadzor trenutnog statusa pisača, već i podešavanje ključnih parametara kao što su temperatura i kalibracija, što rezultira uštedom vremena i resursa.

Tehnologije poput Flask frameworka i Pythona, zajedno s frontend alatima poput HTML-a, CSS-a i JavaScripta, omogućile su razvoj intuitivnog i funkcionalnog korisničkog sučelja. Kroz jednostavno i pregledno sučelje, korisnici mogu bez poteškoća upravljati ispisom, vršiti kalibracije i prilagođavati postavke pisača prema svojim potrebama. Implementacija GCode jezika omogućava preciznu kontrolu nad postupcima ispisa, što je neophodno za postizanje visokokvalitetnih rezultata u 3D ispisu.

Uvođenjem funkcionalnosti poput daljinskog upravljanja i praćenja ispisa, ova aplikacija omogućava veću fleksibilnost korisnicima, jer više nisu vezani za fizičku prisutnost uz svaki pisač. To posebno dolazi do izražaja u okruženjima gdje je potrebno upravljati s više uređaja odjednom, primjerice u edukacijskim ustanovama ili u proizvodnim laboratorijima. Aplikacija također omogućava lakše održavanje i upravljanje procesom ispisa, smanjujući mogućnost pogrešaka i omogućavajući korisnicima da pravovremeno reagiraju na eventualne probleme tijekom ispisa.

Jedna od potencijalnih nadogradnji ove aplikacije u budućnosti može uključivati implementaciju kamera za praćenje statusa ispisa u stvarnom vremenu, što bi dodatno unaprijedilo nadzor nad procesom ispisa. Također, dodatak analitičkih alata mogao bi pružiti korisnicima dublji uvid u performanse pojedinih pisača te omogućiti optimizaciju rada na temelju podataka dobivenih tijekom ispisa.

Zaključno, EasyPrint predstavlja iznimno korisno rješenje koje odgovara na potrebe modernih laboratorija i proizvodnih postrojenja, te omogućava efikasniji i jednostavniji rad s više 3D pisača. Ova aplikacija omogućava korisnicima da sa sigurnošću upravljaju uređajima i optimiziraju proizvodne procese bez potrebe za stalnim fizičkim nadzorom, čime doprinosi većoj produktivnosti i boljem korištenju resursa.

Literatura

1. „Original Prusa i3 MK2S“, Pristupljeno: [24.8.2024.], Dostupno na: https://help.prusa3d.com/tag/mk2s
2. „Basics of 3D Printing with Josef Prusa“, Pristupljeno: [3.9.2024.], Dostupno na: https://www.prusa3d.com/page/basics-of-3d-printing-with-josef-prusa_490/
3. „OctoPrint“, Pristupljeno: [22.8.2024.], Dostupno na: https://octoprint.org/
4. „Raspberry Pi Documentation“, Pristupljeno: [30.8.2024.], Dostupno na: https://www.raspberrypi.org/documentation/
5. „Flask Documentation“, Pristupljeno: [29.8.2024.], Dostupno na: https://flask.palletsprojects.com/en/2.0.x/
6. „Python Requests Documentation“, Pristupljeno: [4.9.2024.], Dostupno na: https://docs.python-requests.org/en/latest/
7. „Ubuntu 24.04.1 LTS“, Pristupljeno: [22.8.2024.], Dostupno na: https://ubuntu.com/download/desktop
8. „Wireless Printing for Original Prusa i3 MK2S“, Pristupljeno: [26.8.2024.], Dostupno na: https://forum.prusa3d.com/forum/original-prusa-i3-mk2-s-czech-cesky-ostatni-archiv/wireless-printing-which-works-best-with-mk2-pronterface - octoprint-tosiba-wifi-card/
9. Varga M., Strugar I., (2016). „Informacijski sustavi u poslovanju“, Pristupljeno: [2.9.2024.], Dostupno na: https://www.dropbox.com/scl/fi/3wak4yvpozvcuip0go7qx/PIS_LIT-Varga-Strugar-Informacijski_sustavi_u_poslovanju.pdf?rlkey=vhwek6a4f1kjvoeect6nh2or5&e=1&dl=0
10. Baltzan, P. (2013). „Business Driven Information Systems“, Pristupljeno: [2.9.2024.], Dostupno na: https://www.dropbox.com/scl/fi/kzuqzrdnwnr1cz25wfdmt/PIS_LIT-Paige-Baltzan-Business_Driven_Information_Systems.pdf?rlkey=on6y1bt0kfi3rok43qqt5qs2j&e=1&dl=0
11. Panian Ž., Ćurko K., (2010). „Poslovni informacijski sustavi“, Pristupljeno: [2.9.2024.], Dostupno na: https://www.dropbox.com/scl/fi/a235md7z9bed8w93q83nb/PIS_LIT-Panian-urko-Poslovni_informacijski_sustavi.pdf?rlkey=u9a01jxokf922rgd90et8u739&e=1&dl=0

Popis slika

Popis primjera

Sažetak

Ovaj projekt prikazuje stvaranje programa *EasyPrint* koji koristi Raspberry Pi i druge web tehnologije za omogućavanje daljinskog upravljanja višestrukim 3D pisačima. U područjima kao što su inženjerstvo, dizajn i obrazovanje, 3D ispis je postao neophodan, ali upravljanje nekoliko pisača odjednom može biti teško. Cilj ovog projekta bio je razviti intuitivno web-bazirano rješenje koje korisnicima omogućuje nadzor i upravljanje operacijama 3D ispisa, praćenje stanja pisača i prijenos GCode naredbi određenim pisačima. Nekoliko Prusa MK2S 3D printera spojeno je na Raspberry Pi 4 centralni hub, koji je služio kao razvojna platforma za *EasyPrint* sustav. Koristeći Python, Flask, HTML, CSS, JavaScript i GCode, glavne tehnologije aplikacije omogućuju komunikaciju u stvarnom vremenu. Rukovanje mnogim pisačima s jednog sučelja olakšano je značajkama sustava za upravljanje filament-a, kontrolu temperature i kalibraciju pisača. Aplikacija olakšava upravljanje 3D printerima i jamči da korisnici mogu uploadati i ispisivati datoteke, mijenjati parametre pisača i pratiti status ispisa bez potrebe da fizički budu prisutni. Zbog svoje skalabilnosti i fleksibilnosti, ova web-bazirana metoda je savršena za laboratorije, proizvodna mjesta i obrazovne ustanove koje trebaju imati centraliziranu kontrolu nad 3D pisačima. Učinkovito rješenje pokazuje kako se suvremeni hardver i web tehnologije mogu koristiti za podršku daljinskim proizvodnim operacijama. Buduća poboljšanja mogla bi uključivati naprednu analitiku ispisa i praćenje kamerom uživo za još veću optimizaciju.

Ključne riječi:
Daljinsko upravljanje 3D printerima, Raspberry Pi 4, 3D ispis, Kalibracija pisača, Upravljanje filamentom, Upravljanje višestrukim pisačima, Laboratorijski uređaji, Automatizacija 3D ispisa, Upravljanje ispisom putem weba.

Summary

This project shows the creation of an *EasyPrint* application that uses the Raspberry Pi and other web technologies to enable remote control of multiple 3D printers. In fields such as engineering, design and education, 3D printing has become indispensable, but managing several printers at once can be difficult. The goal of this project was to develop an intuitive web-based solution that allows users to monitor and manage 3D printing operations, monitor printer status, and transmit GCode commands to specific printers. Several Prusa MK2S 3D printers were connected to a Raspberry Pi 4 central hub, which served as a development platform for the *EasyPrint* system. Using Python, Flask, HTML, CSS, JavaScript and GCode, the application's main technologies enable real-time communication. Operating multiple printers from a single interface is made easy with system features for filament management, temperature control and printer calibration. The application facilitates the management of 3D printers and ensures that users can upload and print files, change printer parameters and monitor print status without having to be physically present. Due to its scalability and flexibility, this web-based method is perfect for laboratories, manufacturing sites and educational institutions that need to have centralized control over 3D printers. The effective solution demonstrates how modern hardware and web technologies can be used to support remote manufacturing operations.
Future improvements could include advanced print analytics and live camera view for even greater optimization.

Keywords:
Remote control of 3D printers, Raspberry Pi 4, 3D printing, Printer calibration, Filament management, Multi-printer control, Laboratory devices, 3D print automation, Web-based print management.