ETS Academy

PROJECT TITLE

LINKEDIN-BOT

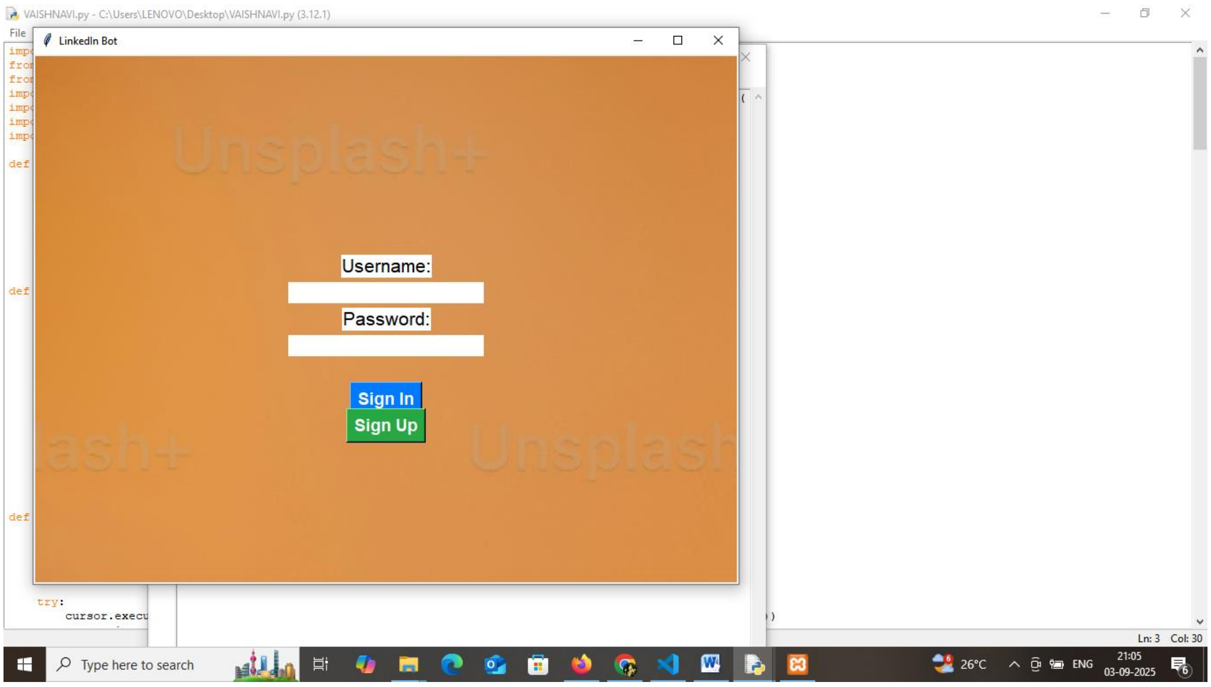
Python Development

```
import tkinter as tk
from tkinter import messagebox, Canvas, PhotoImage
from PIL import Image, ImageTk
import mysql.connector
import bcrypt
import json
import os


def get_db_connection():
    connection = mysql.connector.connect(
        host="localhost",
        user="root",
        password="",  # Use your MySQL root password here if you have one
        database="linkedin_bot"
    )
    return connection


def signin_user(username, password):
    connection = get_db_connection()
    cursor = connection.cursor(dictionary=True)

    cursor.execute("SELECT * FROM users WHERE username = %s LIMIT 1", (username,))
    user = cursor.fetchone()
    cursor.close()
    connection.close()

    if user and bcrypt.checkpw(password.encode('utf-8'), user['password'].encode('utf-8')):
        return True
    else:
        messagebox.showerror("Signin Error", "Invalid username or password.")
        return Falseo

def signup_user(username, password):
    connection = get_db_connection()
    cursor = connection.cursor()

    hashed_password = bcrypt.hashpw(password.encode('utf-8'), bcrypt.gensalt())

    try:
        cursor.execute("INSERT INTO users (username, password) VALUES (%s, %s)",
(username, hashed_password))
        connection.commit()
        print("User signed up successfully.")
        return True
    except mysql.connector.IntegrityError as e:
        print(f"Error: {e}")
        messagebox.showerror("Signup Error", "Username already exists.")
        return False
    finally:
        cursor.close()
        connection.close()
```

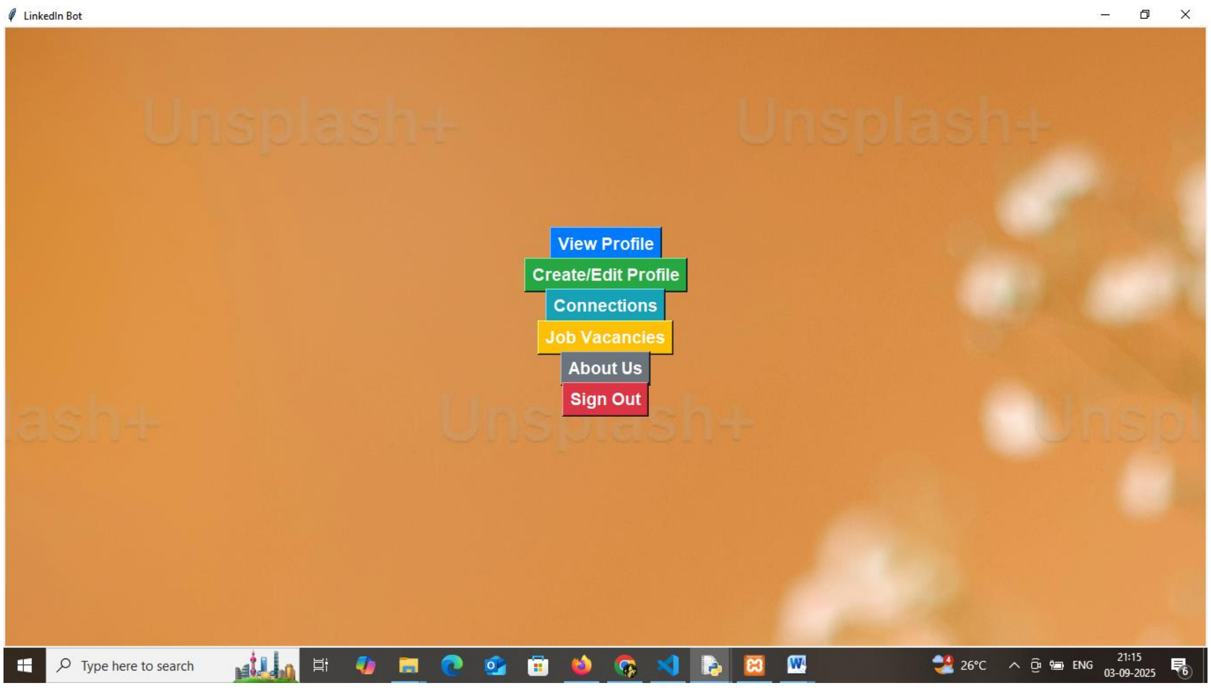
```
'''def signin_user(username, password):
    connection = get_db_connection()
    cursor = connection.cursor(dictionary=True)

    cursor.execute("SELECT * FROM users WHERE username = %s LIMIT 1", (username,))
    user = cursor.fetchone()  # Fetch the single result

    cursor.close()  # Close the cursor after fetching the result
    connection.close()  # Close the connection

    if user and bcrypt.checkpw(password.encode('utf-8'), user['password'].encode('utf-8')):
        return True
    else:
        messagebox.showerror("Signin Error", "Invalid username or password.")
        return False'''



# Continue with the rest of your Tkinter application code
# Profile functions
def load_profile(username):
    profile_path = f'profiles/{username}.json'
    if os.path.exists(profile_path):
        with open(profile_path, 'r') as file:
            return json.load(file)
    else:
        return None


def save_profile(username, profile_data):
    profile_path = f'profiles/{username}.json'
    with open(profile_path, 'w') as file:
        json.dump(profile_data, file)


class LinkedInBotApp:
    def __init__(self, root):
        self.root = root
        self.root.title("LinkedIn Bot")
        self.root.geometry("800x600")
        self.root.configure(bg="#f0f0f0")


        self.username = tk.StringVar()
        self.password = tk.StringVar()


        self.create_login_screen()


    def create_login_screen(self):
         self.clear_screen()
         self.set_background()  # Set the background first


         # Place labels and entry fields
         self.create_transparent_label("Username:").place(relx=0.5, rely=0.4, anchor="center")
         self.create_transparent_entry(self.username).place(relx=0.5, rely=0.45, anchor="center")


         self.create_transparent_label("Password:").place(relx=0.5, rely=0.5, anchor="center")
        self.create_transparent_entry(self.password, show="*").place(relx=0.5, rely=0.55, anchor="center")


        # Place buttons after everything else
        tk.Button(self.root, text="Sign In", font=("Helvetica", 14, "bold"), bg="#007BFF", fg="white", command=self.signin).place(relx=0.5, rely=0.65, anchor="center")
        tk.Button(self.root, text="Sign Up", font=("Helvetica", 14, "bold"), bg="#28A745", fg="white", command=self.signup).place(relx=0.5, rely=0.7, anchor="center")
    def signin(self):
        if signin_user(self.username.get(), self.password.get()):
            self.create_dashboard_screen()

    def signup(self):
        if signup_user(self.username.get(), self.password.get()):
            messagebox.showinfo("Signup Success", "Account created successfully.")
            self.create_login_screen()

    def create_dashboard_screen(self):
        self.clear_screen()
        self.set_background()

        tk.Button(self.root, text="View Profile", font=("Helvetica", 14, "bold"), bg="#007BFF", fg="white", command=self.view_profile).place(relx=0.5, rely=0.35, anchor="center")
        tk.Button(self.root, text="Create/Edit Profile", font=("Helvetica", 14, "bold"), bg="#28A745", fg="white", command=self.edit_profile).place(relx=0.5, rely=0.4, anchor="center")
        tk.Button(self.root, text="Connections", font=("Helvetica", 14, "bold"), bg="#17A2B8", fg="white", command=self.view_connections).place(relx=0.5, rely=0.45, anchor="center")
        tk.Button(self.root, text="Job Vacancies", font=("Helvetica", 14, "bold"), bg="#FFC107", fg="white", command=self.view_jobs).place(relx=0.5, rely=0.5, anchor="center")
        tk.Button(self.root, text="About Us", font=("Helvetica", 14, "bold"), bg="#6C757D", fg="white", command=self.about_us).place(relx=0.5, rely=0.55, anchor="center")
        tk.Button(self.root, text="Sign Out", font=("Helvetica", 14, "bold"), bg="#DC3545",
fg="white", command=self.signout).place(relx=0.5, rely=0.6, anchor="center")


    def view_profile(self):
        profile = load_profile(self.username.get())
        if profile:
            self.clear_screen()
            self.set_background()
            tk.Label(self.root, text=f"Name: {profile['name']}", font=("Helvetica", 14), bg="white",
fg="black").place(relx=0.5, rely=0.35, anchor="center")
            tk.Label(self.root, text=f"Email: {profile['email']}", font=("Helvetica", 14), bg="white",
fg="black").place(relx=0.5, rely=0.4, anchor="center")
            tk.Label(self.root, text=f"Headline: {profile['headline']}", font=("Helvetica", 14),
bg="white", fg="black").place(relx=0.5, rely=0.45, anchor="center")
            tk.Label(self.root, text=f"Summary: {profile['summary']}", font=("Helvetica", 14),
bg="white", fg="black").place(relx=0.5, rely=0.5, anchor="center")
            tk.Button(self.root, text="Back", font=("Helvetica", 14, "bold"), bg="#DC3545",
fg="white", command=self.create_dashboard_screen).place(relx=0.5, rely=0.6,
anchor="center")
        else:
            messagebox.showinfo("Profile", "Profile not found.")


    def edit_profile(self):
        self.clear_screen()
        self.set_background()


        profile = load_profile(self.username.get())
        name_var = tk.StringVar(value=profile['name'] if profile else "")
        email_var = tk.StringVar(value=profile['email'] if profile else "")
        headline_var = tk.StringVar(value=profile['headline'] if profile else "")
        summary_var = tk.StringVar(value=profile['summary'] if profile else "")
        self.create_transparent_label("Name:").place(relx=0.5, rely=0.35, anchor="center")
        self.create_transparent_entry(name_var).place(relx=0.5, rely=0.4, anchor="center")

        self.create_transparent_label("Email:").place(relx=0.5, rely=0.45, anchor="center")
        self.create_transparent_entry(email_var).place(relx=0.5, rely=0.5, anchor="center")

        self.create_transparent_label("Headline:").place(relx=0.5, rely=0.55, anchor="center")
        self.create_transparent_entry(headline_var).place(relx=0.5, rely=0.6, anchor="center")

        self.create_transparent_label("Summary:").place(relx=0.5, rely=0.65, anchor="center")
        self.create_transparent_entry(summary_var).place(relx=0.5, rely=0.7, anchor="center")

        def save():
            profile_data = {
                "name": name_var.get(),
                "email": email_var.get(),
                "headline": headline_var.get(),
                "summary": summary_var.get()
            }
            save_profile(self.username.get(), profile_data)
            self.create_dashboard_screen()

        tk.Button(self.root, text="Save Profile", font=("Helvetica", 14, "bold"), bg="#28A745", fg="white", command=save).place(relx=0.5, rely=0.75, anchor="center")
        tk.Button(self.root, text="Back", font=("Helvetica", 14, "bold"), bg="#DC3545", fg="white", command=self.create_dashboard_screen).place(relx=0.5, rely=0.8, anchor="center")

    def view_connections(self):
        self.clear_screen()
        self.set_background()

        connections = [
            "John Doe",
            "Jane Smith",
            "Robert Johnson",
            "Emily Davis",
            "Michael Brown"
        ]

        for idx, name in enumerate(connections):
            connection_button = tk.Button(self.root, text=name, font=("Helvetica", 16),
bg="#007BFF", fg="white",
                                      command=lambda n=name: self.connect_person(n))
            connection_button.place(relx=0.5, rely=0.3 + idx * 0.1, anchor="center")

        tk.Button(self.root, text="Back", font=("Helvetica", 14, "bold"), bg="#DC3545",
fg="white", command=self.create_dashboard_screen).place(relx=0.5, rely=0.8,
anchor="center")



# Continue with the rest of your Tkinter application code

    def connect_person(self, name):
        messagebox.showinfo("Connection", f"Connection successful with {name}!")
    def view_jobs(self):
        self.clear_screen()
        self.set_background()

        jobs = [
            {"company": "Company A", "title": "Software Engineer"},
            {"company": "Company B", "title": "Data Scientist"},
            {"company": "Company C", "title": "Product Manager"},
            {"company": "Company D", "title": "UX Designer"},
            {"company": "Company E", "title": "DevOps Engineer"}
        ]

        for idx, job in enumerate(jobs):
            company_label = tk.Label(self.root, text=f"{job['company']}:", font=("Helvetica", 16),
bg="white", fg="black")
            title_label = tk.Label(self.root, text=f"{job['title']}", font=("Helvetica", 14), bg="white",
fg="black")

            company_label.place(relx=0.4, rely=0.3 + idx * 0.1, anchor="center")
            title_label.place(relx=0.6, rely=0.3 + idx * 0.1, anchor="center")

        tk.Button(self.root, text="Back", font=("Helvetica", 14, "bold"), bg="#DC3545",
fg="white", command=self.create_dashboard_screen).place(relx=0.5, rely=0.8,
anchor="center")

    def about_us(self):
        self.clear_screen()
        self.set_background()
        about_text = (
            "Welcome to LinkedIn Bot!\n\n"
            "This application is designed to mimic some of the features of LinkedIn, "
            "allowing users to sign up, create profiles, connect with others, and explore job
vacancies. "
            "It's built using Python's Tkinter library for the user interface and MySQL for data
storage.\n\n"
            "Features:\n"
            "- Sign up and Sign in securely.\n"
            "- Create and edit your professional profile.\n"
            "- Connect with other users.\n"
            "- Browse job vacancies from top companies.\n\n"
            "This project is an educational tool to demonstrate how one might build a simple, "
            "desktop-based application with a database backend.\n\n"
            "Thank you for using LinkedIn Bot!"
        )

        about_label = tk.Label(self.root, text=about_text, font=("Helvetica", 14), bg="white",
fg="black", justify="left", wraplength=600)
        about_label.place(relx=0.5, rely=0.4, anchor="center")

        tk.Button(self.root, text="Back", font=("Helvetica", 14, "bold"), bg="#DC3545",
fg="white", command=self.create_dashboard_screen).place(relx=0.5, rely=0.9,
anchor="center")

    def signout(self):
        self.username.set("")
        self.password.set("")
        self.create_login_screen()
    def clear_screen(self):
        for widget in self.root.winfo_children():
            widget.destroy()

    def set_background(self):
        canvas = Canvas(self.root, width=800, height=600)
        canvas.pack(fill="both", expand=True)
        background_image = Image.open("C:/Users/LENOVO/Pictures/premium_photo-
1676070096487-32dd955e09e0.jpg")
        background_image = ImageTk.PhotoImage(background_image)

        canvas.create_image(0, 0, anchor=tk.NW, image=background_image)
        canvas.image = background_image

    def create_transparent_label(self, text):
        label = tk.Label(self.root, text=text, font=("Helvetica", 16), fg="black", bg="white", bd=0,
highlightthickness=0)
        return label

    def create_transparent_entry(self, variable, show=None):
        entry = tk.Entry(self.root, textvariable=variable, font=("Helvetica", 14), fg="black", bd=0,
highlightthickness=0, bg="white", show=show)
        return entry

if __name__ == "__main__":
        if not os.path.exists('profiles'):
            os.makedirs('profiles')

        root = tk.Tk()
        app = LinkedInBotApp(root)
    root.mainloop()
```

OUTPUT

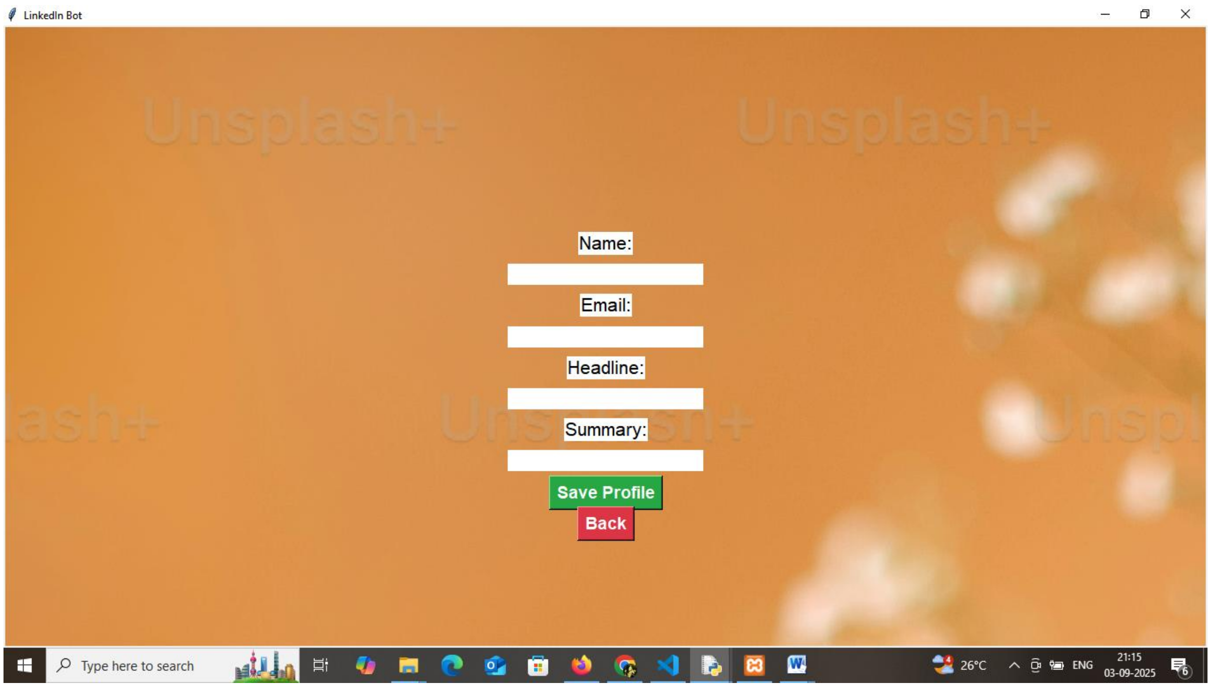
1.


VAISHNAVI.py - C:\Users\LENOVO\Desktop\VAISHNAVI.py (3.12.1)
LinkedIn Bot
Username:
Password:
Sign In
Sign Up
Ln: 3 Col: 30
Type here to search
26°C ENG 21:05 03-09-2025

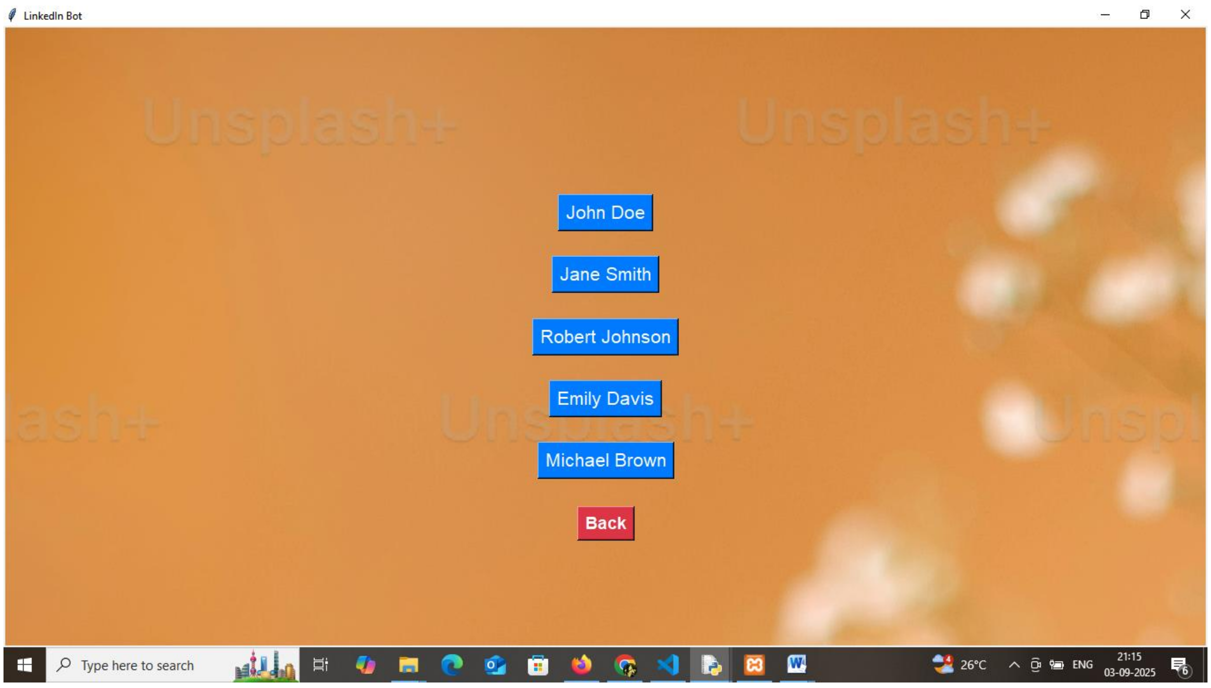
2.



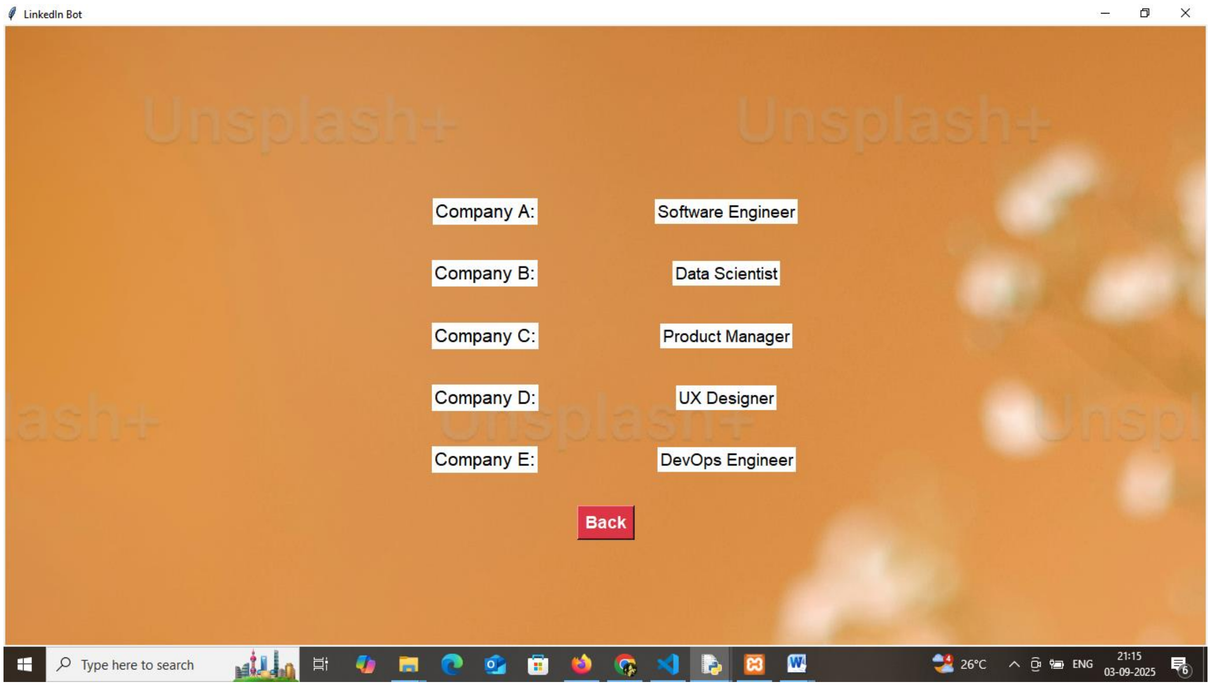
3.


LinkedIn Bot
Name:
Email:
Headline:
Summary:
Save Profile
Back
Type here to search
26°C
ENG
21:15
03-09-2025

4.

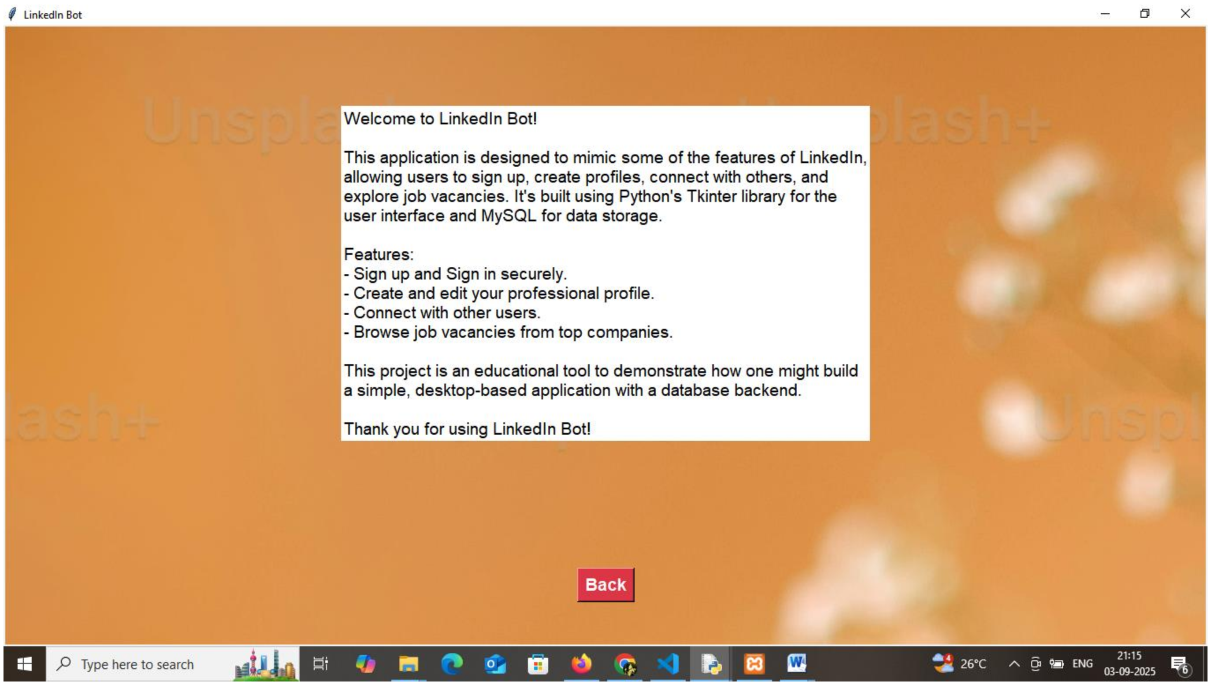

LinkedIn Bot
John Doe
Jane Smith
Robert Johnson
Emily Davis
Michael Brown
Back
Type here to search
26°C
ENG
21:15
03-09-2025

5.



Company A:	Software Engineer
Company B:	Data Scientist
Company C:	Product Manager
Company D:	UX Designer
Company E:	DevOps Engineer

6.



Welcome to LinkedIn Bot!

This application is designed to mimic some of the features of LinkedIn, allowing users to sign up, create profiles, connect with others, and explore job vacancies. It's built using Python's Tkinter library for the user interface and MySQL for data storage.

Features:
- Sign up and Sign in securely.
- Create and edit your professional profile.
- Connect with other users.
- Browse job vacancies from top companies.

This project is an educational tool to demonstrate how one might build a simple, desktop-based application with a database backend.

Thank you for using LinkedIn Bot!