RESUME BUILDER

(Python project report)

INTRODUCTION:

In the modern digital era, creating a professional and well-structured resume is a critical step toward securing employment opportunities. A resume builder is a tool that simplifies and automates this process by generating resumes based on user inputs. This project aims to develop a Resume Builder using Python, which allows users to input personal, educational, and professional details and generates a polished resume in a formatted document, such as PDF or DOCX.

<u>Why this code is useful for future?</u>

✅ 1. Practical Application of Coding Skills

✅ 2. Automates a Time-Consuming Task

✅ 3. Highly Customizable

A resume builder code is not just a small project — it's a useful, scalable, and impressive tool that showcases your skills and helps others too. It's especially great for students, job seekers, and beginner programmers.

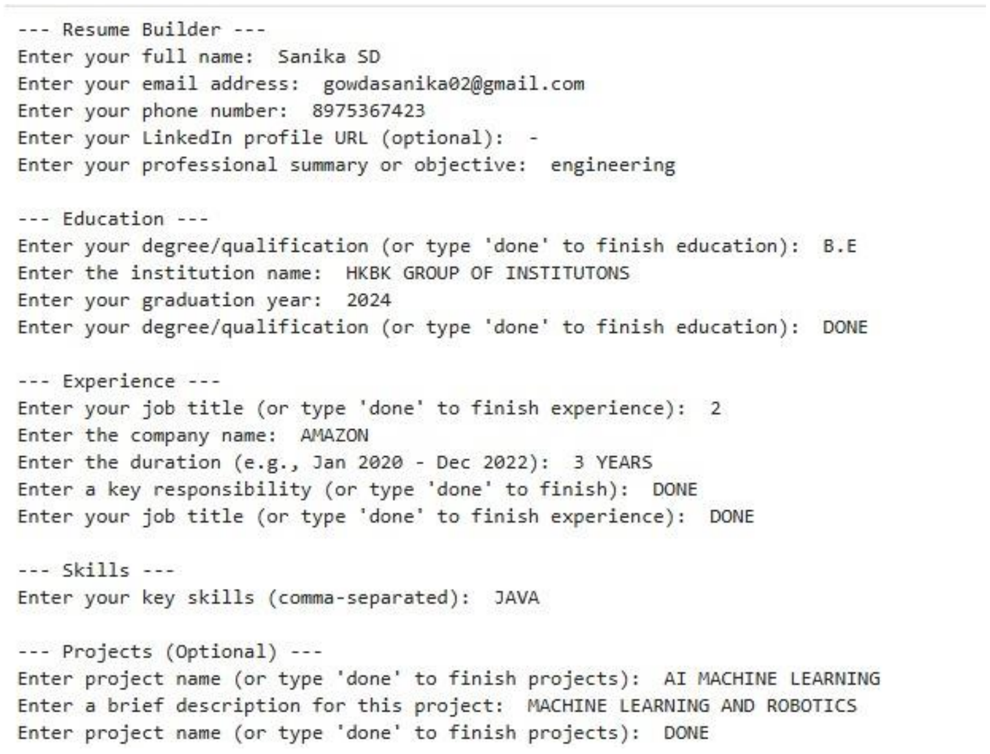
<u>PYTHON CODE</u>

```
def get_user_input(prompt):
    """Gets string input from the user."""
    return input(prompt).strip()


def get_list_input(prompt_singular, prompt_done_indicator):
    """Gets a list of items from the user until 'done' is entered."""
    items = []
    while True:
        item = get_user_input(f"{prompt_singular} (or type '{prompt_done_indicator}' to finish):
")
        if item.lower() == prompt_done_indicator.lower():
```

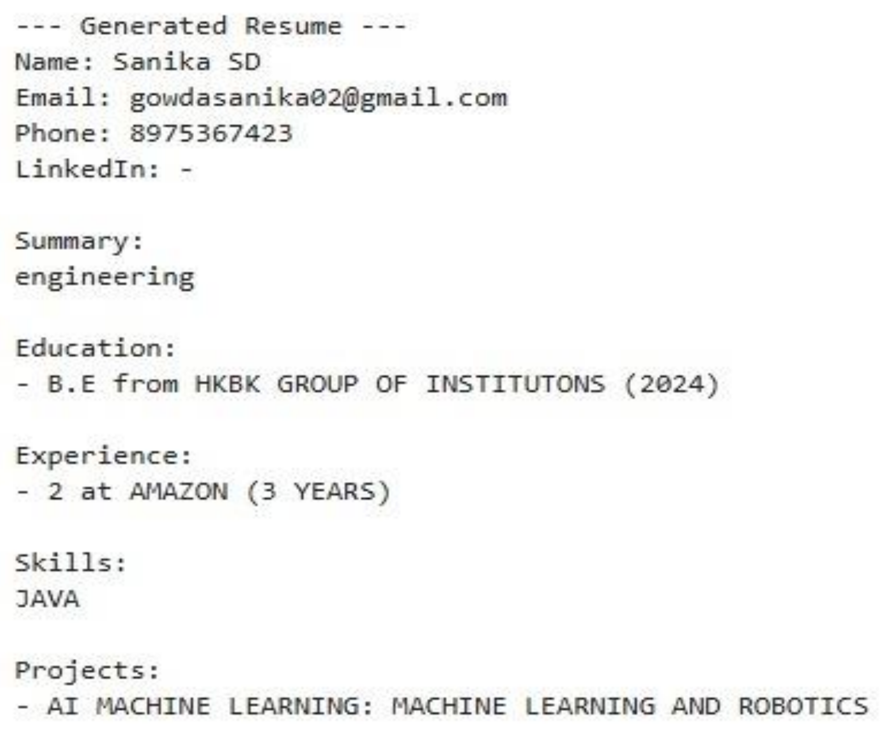
```
        break
      items.append(item)
    return items

def build_resume():
    """Collects resume information and prints it."""
    print("--- Resume Builder ---")

    # Personal Information
    name = get_user_input("Enter your full name: ")
    email = get_user_input("Enter your email address: ")
    phone = get_user_input("Enter your phone number: ")
    linkedin = get_user_input("Enter your LinkedIn profile URL (optional): ")

    # Summary/Objective
    summary = get_user_input("Enter your professional summary or objective: ")

    # Education
    print("\n--- Education ---")
    education_entries = []
    while True:
      degree = get_user_input("Enter your degree/qualification (or type 'done' to finish 
education): ")
      if degree.lower() == 'done':
        break
      institution = get_user_input("Enter the institution name: ")
```

```
        graduation_year = get_user_input("Enter your graduation year: ")
        education_entries.append({'degree': degree, 'institution': institution, 'year': 
graduation_year})

    # Experience
    print("\n--- Experience ---")
    experience_entries = []
    while True:
        job_title = get_user_input("Enter your job title (or type 'done' to finish experience): ")
        if job_title.lower() == 'done':
            break
        company = get_user_input("Enter the company name: ")
        duration = get_user_input("Enter the duration (e.g., Jan 2020 - Dec 2022): ")
        responsibilities = get_list_input("Enter a key responsibility", "done")
        experience_entries.append({'title': job_title, 'company': company, 'duration': duration, 
'responsibilities': responsibilities})

    # Skills
    print("\n--- Skills ---")
    skills = get_user_input("Enter your key skills (comma-separated): ").split(',')
    skills = [s.strip() for s in skills if s.strip()] # Clean and remove empty entries

    # Projects (Optional)
    print("\n--- Projects (Optional) ---")
    project_entries = []
    while True:
        project_name = get_user_input("Enter project name (or type 'done' to finish projects): ")
        if project_name.lower() == 'done':
            break
        project_description = get_user_input("Enter a brief description for this project: ")
        project_entries.append({'name': project_name, 'description': project_description})

    # Print the collected resume information
    print("\n--- Generated Resume ---")
    print(f"Name: {name}")
    print(f"Email: {email}")
    print(f"Phone: {phone}")
    if linkedin:
        print(f"LinkedIn: {linkedin}")

    print("\nSummary:")
    print(summary)

    if education_entries:
        print("\nEducation:")
        for edu in education_entries:
            print(f"- {edu['degree']} from {edu['institution']} ({edu['year']})")

    if experience_entries:
        print("\nExperience:")
        for exp in experience_entries:
            print(f"- {exp['title']} at {exp['company']} ({exp['duration']})")
            for resp in exp['responsibilities']:
            print(f"  - {resp}")


    if skills:
        print("\nSkills:")
        print(", ".join(skills))


    if project_entries:
        print("\nProjects:")
        for proj in project_entries:
            print(f"- {proj['name']}: {proj['description']}")


if __name__ == "__main__":
    build_resume()
```

CODE OUTPUT



```
--- Resume Builder ---
Enter your full name:  Sanika SD
Enter your email address:  gowdasanika02@gmail.com
Enter your phone number:  8975367423
Enter your LinkedIn profile URL (optional):  -
Enter your professional summary or objective:  engineering

--- Education ---
Enter your degree/qualification (or type 'done' to finish education):  B.E
Enter the institution name:  HKBK GROUP OF INSTITUTONS
Enter your graduation year:  2024
Enter your degree/qualification (or type 'done' to finish education):  DONE

--- Experience ---
Enter your job title (or type 'done' to finish experience):  2
Enter the company name:  AMAZON
Enter the duration (e.g., Jan 2020 - Dec 2022):  3 YEARS
Enter a key responsibility (or type 'done' to finish):  DONE
Enter your job title (or type 'done' to finish experience):  DONE

--- Skills ---
Enter your key skills (comma-separated):  JAVA

--- Projects (Optional) ---
Enter project name (or type 'done' to finish projects):  AI MACHINE LEARNING
Enter a brief description for this project:  MACHINE LEARNING AND ROBOTICS
Enter project name (or type 'done' to finish projects):  DONE
```



```
--- Generated Resume ---
Name: Sanika SD
Email: gowdasanika02@gmail.com
Phone: 8975367423
LinkedIn: -

Summary:
engineering

Education:
- B.E from HKBK GROUP OF INSTITUTONS (2024)

Experience:
- 2 at AMAZON (3 YEARS)

Skills:
JAVA

Projects:
- AI MACHINE LEARNING: MACHINE LEARNING AND ROBOTICS
```

WEBSITES & APPS USED

Google,Chatgpt, Jupyter notebook

<u>MODULES & LIBRARIES</u>

Name.	**Type.**	**Purpose**
Builtins.	**Built-in**	**Provides . input(), print(), . str, list, etc.**

<u>TASK CLASS AND FUNCTIONS</u> ✅

User-Defined Functions

Get_user_input(prompt) = Takes input from the user, removes any leading/trailing spaces using .strip(), and returns it.

Get_list_input(prompt_singular, prompt_done_indicator) = Continuously collects multiple inputs (e.g., responsibilities) from the user until a specific keyword (like 'done') is entered.

Build_resume() = Main function that controls the overall resume-building process. It collects data, stores it in appropriate structures, and prints the final formatted resume.

🛠 Built-in Functions/Methods Used

Input(prompt)Built-in function = Collects input from the user via the console.

Print() Built-in function = Displays messages or data on the console.

.strip()String method = Removes any leading and trailing whitespace from strings.

.lower() String method = Converts a string to all lowercase (used to match "done" regardless of case).

.split(',') String method = Splits a comma-separated string into a list (used for splitting skills).

<u>MAIN PROGRAM FUNCTION</u>

main functions and components used in your **resume builder Python code**, along with their **purposes**:

🔷 1. `get_user_input(prompt)`

- **Type**: Function (User-defined)
- **Purpose**:
  To get a clean string input from the user by:
  - Displaying the prompt.
  - Collecting user input.
  - Removing unnecessary spaces using `.strip()`.

◈ 2. `get_list_input(prompt_singular, prompt_done_indicator)`

- **Type**: Function (User-defined)
- **Purpose**:
  To collect a **list of items** (like job responsibilities or achievements) from the user until a stop keyword (like `'done'`) is entered.

◈ 3. `build_resume()`

- **Type**: Function (User-defined - **Main Logic Controller**)
- **Purpose**:
  This is the **core function** of the program. It:
  - Calls other functions to collect:
    - Personal Information
    - Summary
    - Education
    - Experience
    - Skills
    - Projects
  - Stores data in structured formats (lists of dictionaries).
  - Prints a formatted resume as output.

◈ 4. `if __name__ == "__main__":`

- **Type**: Python special block
- **Purpose**:
  Ensures that the `build_resume()` function runs **only** when the script is executed directly (not when imported as a module).

◈ 5. Loops and Conditions

- **`while True` loops**
  - Repeatedly ask the user for inputs (e.g., multiple education/experience entries).
- **`if` conditions**
  - Used to check for `"done"` input or to conditionally display optional information (e.g., LinkedIn, Projects).

6. Lists and Dictionaries

- **Lists**
  - To store multiple entries like:
    - Education records
    - Job experiences
    - Skills
    - Projects
- **Dictionaries**
  - To represent structured data for each item (e.g., one job or one education record).

7. `print()` Statements

- **Purpose**:
  To display instructions, prompts, and the final formatted resume to the user.

✓ Summary Table

Component	Type	Purpose
`get_user_input()`	Function	Get clean string input from user
`get_list_input()`	Function	Collect multiple inputs in a list (e.g., responsibilities)
`build_resume()`	Function	Orchestrates data collection and printing of the resume
`if __name__ == "__main__"`	Special block	Ensures main function runs when script is executed
`while, if`	Control Flow	Looping and conditional checks
`list, dict`	Data Types	Store multiple and structured data entries

Component	Type	Purpose
`print(), input()`	Built-in funcs	Show output and get input from user

APPLICATIONS OF RESUME BUILDER

The Resume Builder Python project has several practical applications and real-world uses. Here are the key applications:

🔷 1. Personal Career Tool

Allows users (students, job seekers, freelancers) to quickly create customized and professional resumes.Helps tailor resumes for different job roles by generating multiple versions efficiently.

🔷 2. Educational Purpose

Great project for learning Python fundamentals like:

Input/output handling

File handling (PDF/Word/text generation)

String formatting Data structures (lists, dictionaries)Can also introduce students to modules like reportlab, fpdf, or docx.

🔷 3. Portfolio Project

Adds strong value to a Python developer’s portfolio.

Demonstrates skills in real-world applications and problem-solving.

🔷 4. HR & Recruitment Tools

Can be integrated into recruitment software to allow candidates to generate resumes directly from application forms.

🔷 5. Web-Based or App Integration

Can be upgraded with a GUI or web interface using Tkinter, Flask, or Django.

Can be deployed as a mobile app using Kivy or a web app for public us.

CONCLUSION

The Resume Builder project in Python is a practical and educational tool that demonstrates the power and flexibility of Python in real-world applications. By automating the process of creating resumes, this project saves time, reduces errors, and ensures a consistent and professional output.

It helps users, especially students and job seekers, quickly generate tailored resumes suited for various job opportunities. Moreover, it strengthens the developer's understanding of core Python concepts such as input handling, file manipulation, string formatting, and modular programming.

This project can be further enhanced with graphical user interfaces (GUI), web integration, and advanced features like template selection, PDF export, and even AI-powered suggestions. Overall, the Resume Builder not only serves as a valuable career tool but also stands as a strong portfolio project that showcases programming skills and creative thinking.
