Story Generator App Using Python

Last Updated : 27 Nov, 2023

In the realm of programming and natural language processing, there's an opportunity to create engaging and creative applications that spark the imagination. One such project is the development of a Story Generator App using Python. In this article, we'll embark on a journey to understand how to build a simple yet intriguing story generator application.

Steps to Build a Story Generator App Using Python

Storytelling has been an integral part of human culture for centuries. From epic tales of heroes and adventures to bedtime stories that capture the imagination of children, storytelling has a timeless appeal. With the advent of technology, we can harness the power of programming to create stories that are both captivating and limitless.

Below are the steps by which we can build a story generator app using Python:

Step 1: Installation

We must have the following things install in our system before starting:

- Flask Installation
- Python
- Visual Studio Code

We can simply install Flask by using the following command:

```
pip install flask
```

Step 2: Gathering Input Data

To generate stories, we'll need a dataset of sentences or phrases that our app can use as building blocks. You can create your dataset or use an existing one. For our example, we'll use a simple dataset of phrases:

Python3

```python
beginnings = ["Once upon a time", "In a land far away",
              "In the not-so-distant future"]
characters = ["a brave knight", "an adventure explorer",
              "a curious scientist"]
settings = ["a mysterious forest", "a bustling city",
            "an ancient castle"]
conflicts = ["batlling a fearsome dragon",
             "discovering a hidden treasure",
             "solving a perplexing mystery"]
endings = ["and they all lived happily ever after",
           "and the world was forever changed",
           "and they found their way back home."]
```

Step 3: Creating the Template (index.html)

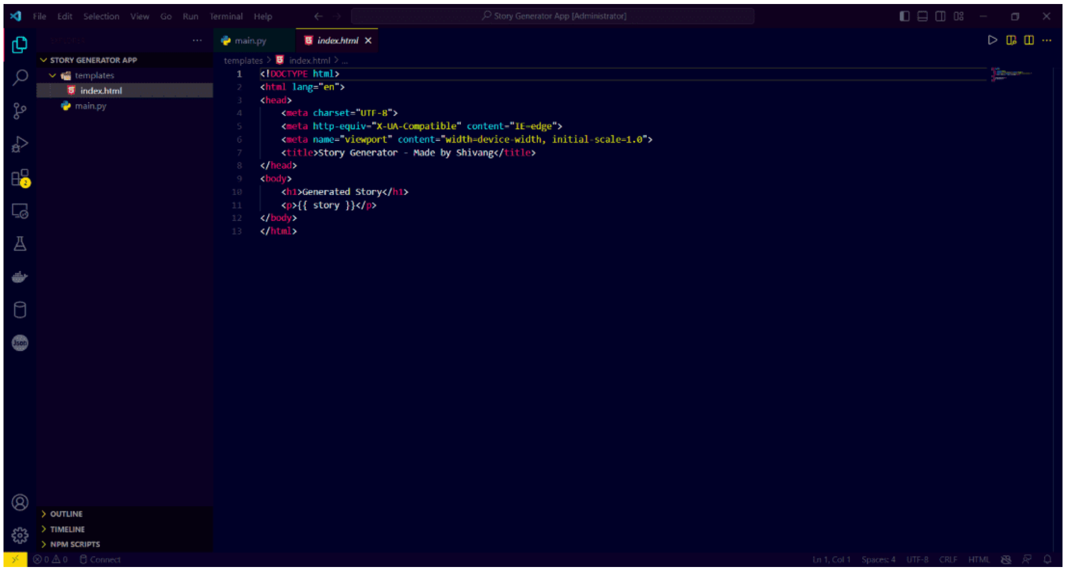
Next, create a folder named templates in the same directory as your Python script. Inside the templates folder, create an index.html file with the following content:

HTML

```html
<!DOCTYPE html>
<html>
<head>
    <title>Story Generator</title>
</head>
<body>
    <h1>Generated Story</h1>
    <p>{{ story }}</p>
</body>
</html>
```

To run your Flask app, execute your Python script. Open a web browser and navigate to http://127.0.0.1:5000/, and you will see a randomly generated story each time you refresh the page.

File Structure and Code Files will look like this..



File Edit Selection View Go Run Terminal Help
Story Generator App [Administrator]
main.py
index.html
templates > index.html > ...
OUTLINE
TIMELINE
NPM SCRIPTS

```
STORY GENERATOR APP
  templates
    index.html
  main.py
```

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Story Generator - Made by Shivang</title>
</head>
<body>
    <h1>Generated Story</h1>
    <p>{{ story }}</p>
</body>
</html>
```

Step 4: Generating a Story (main.py)

Now, let's create a function to generate a story by combining random phrases, nouns, verbs, and objects from our dataset:

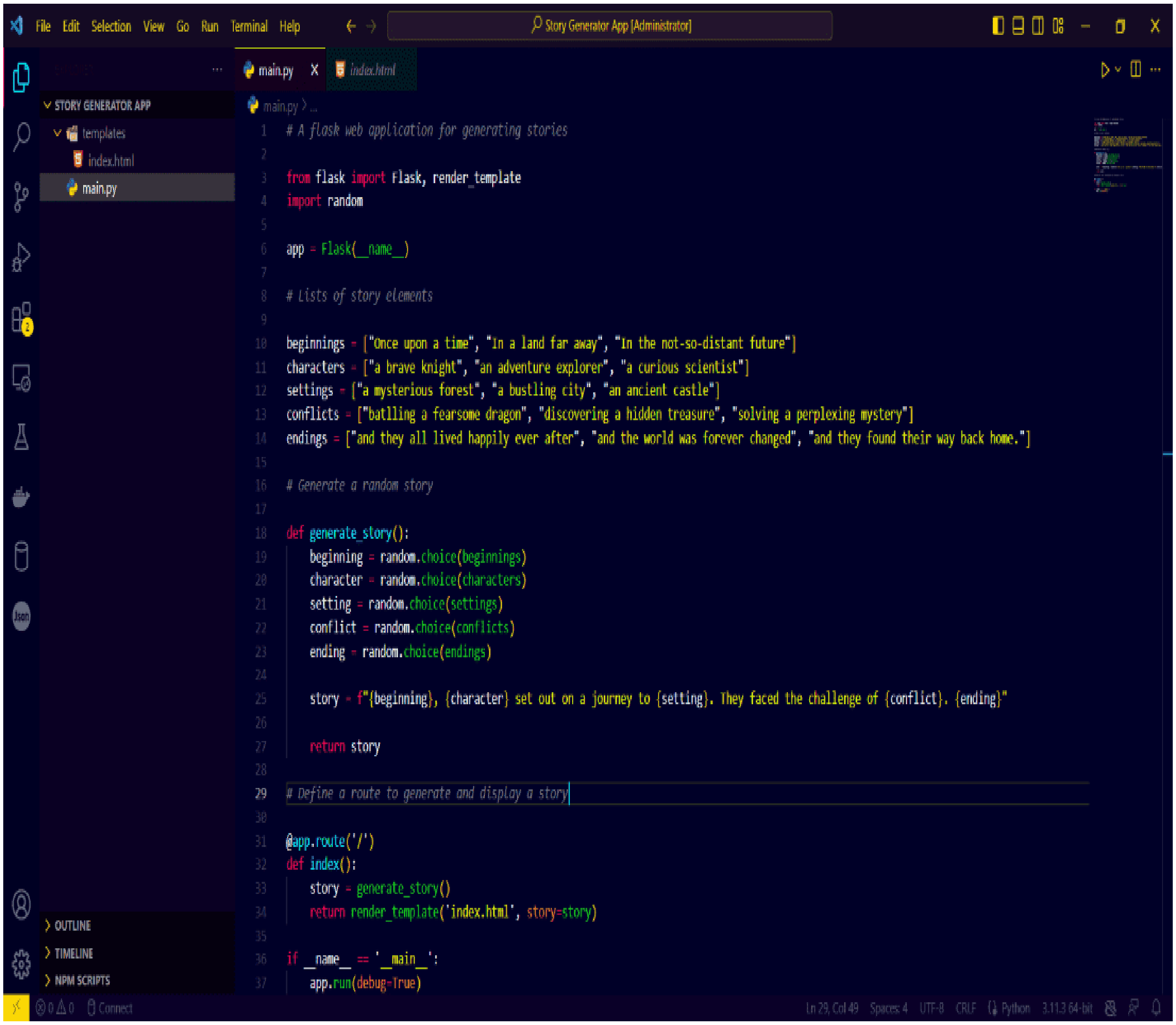
Python3

```
# A flask web application for generating stories

from flask import Flask, render_template
import random

app = Flask(__name__)

# Lists of story elements

beginnings = ["Once upon a time", "In a land far away",
              "In the not-so-distant future"]
```

Flask Templates Jinja2 Flask-REST API Python SQLAlchemy Flask Bcrypt Flask Cookies Json Postman

```
              "a curious scientist"]
settings = ["a mysterious forest", "a bustling city",
            "an ancient castle"]
conflicts = ["batlling a fearsome dragon",
             "discovering a hidden treasure",
             "solving a perplexing mystery"]
endings = ["and they all lived happily ever after",
           "and the world was forever changed",
           "and they found their way back home."]

# Generate a random story

def generate_story():
    beginning = random.choice(beginnings)
    character = random.choice(characters)
    setting = random.choice(settings)
    conflict = random.choice(conflicts)
    ending = random.choice(endings)

    story = f"{beginning}, {character}\
    set out on a journey to {setting}. \
    They faced the challenge of {conflict}. {ending}"

    return story

# Define a route to generate and display a story

@app.route('/')
def index():
    story = generate_story()
    return render_template('index.html', story=story)

if __name__ == '__main__':
    app.run(debug=True)
```

After writing the code make a folder named story generated app and save the file using python extension main.py

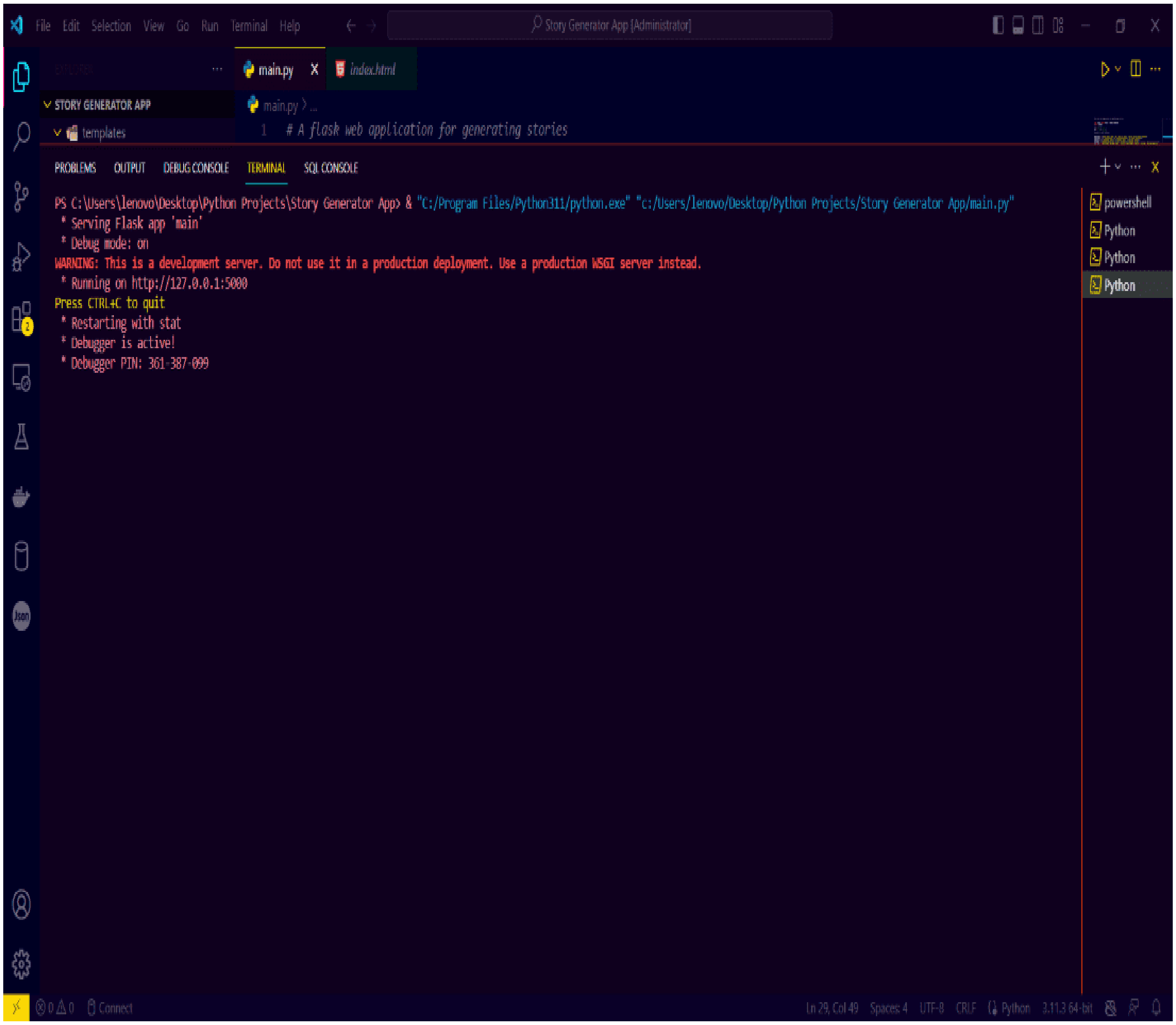

File Edit Selection View Go Run Terminal Help
Story Generator App [Administrator]
main.py
index.html
STORY GENERATOR APP
templates
index.html
main.py
OUTLINE
TIMELINE
NPM SCRIPTS
Ln 29, Col 49 Spaces: 4 UTF-8 CRLF Python 3.11.3 64-bit

```python
# A flask web application for generating stories

from flask import Flask, render_template
import random

app = Flask(__name__)

# Lists of story elements

beginnings = ["Once upon a time", "In a land far away", "In the not-so-distant future"]
characters = ["a brave knight", "an adventure explorer", "a curious scientist"]
settings = ["a mysterious forest", "a bustling city", "an ancient castle"]
conflicts = ["batlling a fearsome dragon", "discovering a hidden treasure", "solving a perplexing mystery"]
endings = ["and they all lived happily ever after", "and the world was forever changed", "and they found their way back home."]

# Generate a random story

def generate_story():
    beginning = random.choice(beginnings)
    character = random.choice(characters)
    setting = random.choice(settings)
    conflict = random.choice(conflicts)
    ending = random.choice(endings)

    story = f"{beginning}, {character} set out on a journey to {setting}. They faced the challenge of {conflict}. {ending}"

    return story

# Define a route to generate and display a story

@app.route('/')
def index():
    story = generate_story()
    return render_template('index.html', story=story)

if __name__ == '__main__':
    app.run(debug=True)
```

main.py

Step 5: Run the File

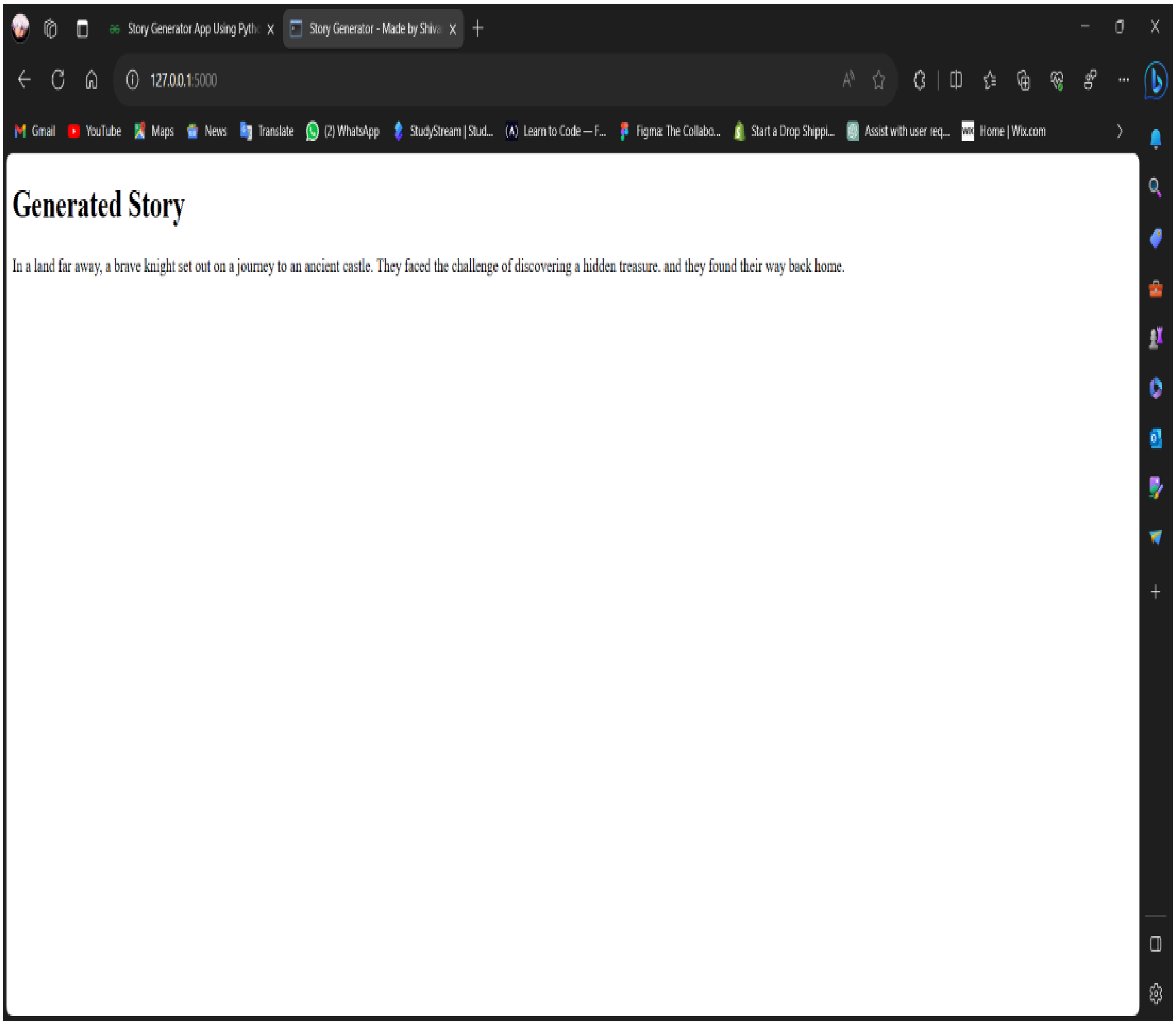
In this step, we will run our flask app by opening our web browser and navigating to http://127.0.0.1:5000/, and we will see a randomly generated story each time you refresh the page.



```
# A flask web application for generating stories
```

```
PS C:\Users\lenovo\Desktop\Python Projects\Story Generator App> & "C:/Program Files/Python311/python.exe" "c:/Users/lenovo/Desktop/Python Projects/Story Generator App/main.py"
 * Serving Flask app 'main'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 361-387-099
```

Output



Generated Story

In a land far away, a brave knight set out on a journey to an ancient castle. They faced the challenge of discovering a hidden treasure. and they found their way back home.

Here are some ideas for future improvements:

User accounts: Allow users to create accounts to save and share their favorite stories.

Story rating system: Implement a way for users to rate stories, making it easier to find the most popular ones.

Story themes: Create different story themes (fantasy, mystery, romance, etc.) and let users choose their preferred theme.

Story length customization: Allow users to specify the desired length of the generated story.

Mobile app version: Develop a mobile app version to reach a wider audience.

Looking to dive into the world of programming or sharpen your Python skills? Our Master Python: Complete Beginner to Advanced Course is your ultimate guide to becoming proficient in Python. This course covers everything you need to build a solid foundation from fundamental programming concepts to advanced techniques. With **hands-on projects**, real-world examples, and expert guidance, you'll gain the confidence to tackle complex **coding challenges**. Whether you're starting from scratch or aiming to enhance your skills, this course is the perfect fit. Enroll now and master Python, the language of the future!

S shya... 1

Previous Article

Wikipedia search app using Flask Framework - Python

Next Article

Similar Reads

How to build a Random Story Generator using Python?

In this section, we are going to make a very interesting beginner-level project of Python. It is a random story generator. The random story generator project aims to generate...

4 min read

Resume Generator App Using Python

In this article, we will explain how to create the Resume Generator App using Python. To generate the resume, we will utilize an API and demonstrate how we can easily create ...

5 min read

Fun Fact Generator Web App in Python

In this article, we will discuss how to create a Fun Fact Generator Web App in Python using the PyWebio module. Essentially, it will create interesting facts at random and...

3 min read

Build an AI Image Generator App With Tkinter

Let's take a brief look at the field of diffusion models, which are used to text to create images. Using a Markov chain, a diffusion model gradually adds noise to the data befor...

5 min read

Python | Random Password Generator using Tkinter

With growing technology, everything has relied on data, and securing this data is the main concern. Passwords are meant to keep the data safe that we upload on the Intern...

4 min read

Automated Certificate generator using Opencv in Python

Prerequisites: Introduction to OpenCV OpenCV is the huge open-source library for computer vision, machine learning, and image processing and now it plays a major role i...

2 min read

Automate getter-setter generator for Java using Python

Encapsulation is defined as the wrapping up of data under a single unit. Encapsulation can be achieved by declaring all the variables in the class as private and writing public...

3 min read

Python - SpongeBob Mocking Text Generator GUI using Tkinter

Prerequisites : Introduction to tkinter | SpongeBob Mocking Text Generator Python offers multiple options for developing a GUI (Graphical User Interface). Out of all the GUI...

4 min read

Wi-Fi QR Code Generator Using Python

Prerequisite: Getting Saved Wifi Passwords using Python We know the wireless network is the most common network adapter for today, Because of its supports portability and...

2 min read

Wikipedia Summary Generator using Python Tkinter

Prerequisite: Tkinter Wikipedia Python offers multiple options for developing a GUI (Graphical User Interface). Out of all the GUI methods, Tkinter is the most commonly...

2 min read

Article Tags :

Python

Geeks Premier League

OpenAI API

Geeks Premier League 2023

+2 More

Practice Tags :

python

GeeksforGeeks
Sanchhaya Education Private Limited

Corporate & Communications Address:-
A-143, 9th Floor, Sovereign Corporate
Tower, Sector- 136, Noida, Uttar Pradesh
(201305) | Registered Address:- K 061,
Tower K, Gulshan Vivante Apartment,
Sector 137, Noida, Gautam Buddh
Nagar, Uttar Pradesh, 201305

Company

- About Us
- Legal
- In Media
- Contact Us
- Advertise with us
- GFG Corporate Solution
- Placement Training Program
- GeeksforGeeks Community

Languages

- Python
- Java
- C++
- PHP
- GoLang
- SQL
- R Language
- Android Tutorial
- Tutorials Archive

DSA

- Data Structures
- Algorithms
- DSA for Beginners
- Basic DSA Problems
- DSA Roadmap
- Top 100 DSA Interview Problems
- DSA Roadmap by Sandeep Jain
- All Cheat Sheets

Data Science & ML

- Data Science With Python
- Data Science For Beginner
- Machine Learning
- ML Maths
- Data Visualisation
- Pandas
- NumPy
- NLP
- Deep Learning

Web Technologies

- HTML
- CSS
- JavaScript
- TypeScript
- ReactJS
- NextJS
- Bootstrap
- Web Design

Python Tutorial

- Python Programming Examples
- Python Projects
- Python Tkinter
- Web Scraping
- OpenCV Tutorial
- Python Interview Question
- Django

Computer Science

- Operating Systems
- Computer Network
- Database Management System
- Software Engineering
- Digital Logic Design
- Engineering Maths
- Software Development
- Software Testing

DevOps

- Git
- Linux
- AWS
- Docker
- Kubernetes
- Azure
- GCP
- DevOps Roadmap

System Design

- High Level Design
- Low Level Design

Inteview Preparation

- Competitive Programming
- Top DS or Algo for CP

UML Diagrams
Interview Guide
Design Patterns
OOAD
System Design Bootcamp
Interview Questions

Company-Wise Recruitment Process
Company-Wise Preparation
Aptitude Preparation
Puzzles

School Subjects

Mathematics
Physics
Chemistry
Biology
Social Science
English Grammar
Commerce
World GK

GeeksforGeeks Videos

DSA
Python
Java
C++
Web Development
Data Science
CS Subjects