



Profile Application using Python Flask and MySQL

Last Updated : 19 Apr, 2023

A framework is a code library that makes a developer's life easier when building web applications by providing reusable code for common operations. There are a number of frameworks for [Python](#), including Flask, Tornado, Pyramid, and [Django](#). Flask is a lightweight web application framework. It is classified as a micro-framework because it does not require particular tools or libraries. Side tabs are used for single-page web applications or to display different contents.

Pre-requisite:

- Knowledge of Python,
- MySQL Workbench, and
- basics of Flask Framework.

Profile Application using Flask

Working with MySQL Workbench

Step-1: Install MySQL workbench. Link to install :

<https://dev.mysql.com/downloads/workbench/> Know more about it :

<https://www.mysql.com/products/workbench/> **Step-2:** Install 'mysqlbd' module in your venv.

```
pip install flask-mysqldb
```

Step 3: Open MySQL workbench.

Step 4: Write the following code. The above SQL statement will create our database **geekprofile** with the table **accounts**.

Step-5: Execute the query.

Query 1 x accounts

Trending Now DSA Web Tech Foundational Courses Data Science Practice Problem Python Machine Le

```

3
4 CREATE TABLE IF NOT EXISTS `accounts` (
5     `id` int(11) NOT NULL AUTO_INCREMENT,
6     `username` varchar(50) NOT NULL,
7     `password` varchar(255) NOT NULL,
8     `email` varchar(100) NOT NULL,
9     `organisation` varchar(100) NOT NULL,
10    `address` varchar(100) NOT NULL,
11    `city` varchar(100) NOT NULL,
12    `state` varchar(100) NOT NULL,
13    `country` varchar(100) NOT NULL,
14    `postalcode` varchar(100) NOT NULL,
15    PRIMARY KEY (`id`)
16 ) ENGINE=InnoDB AUTO_INCREMENT=1 DEFAULT CHARSET=utf8;

```

Creating Project

Step 1: [Create an environment](#), and [install Flask](#)

Step 2: Create an empty folder 'geeksprofile'.

Step 3: Now open your code editor and open this 'geeksprofile' folder.

Step 4: Create 'app.py' folder and write the code given below.

Python3

Store this code in 'app.py' file

```

from flask import Flask, render_template, request, redirect, url_for, session
from flask_mysqldb import MySQL
import MySQLdb.cursors
import re

```

```

app = Flask(__name__)

```

```

app.secret_key = 'your secret key'

```

```

app.config['MYSQL_HOST'] = 'localhost'
app.config['MYSQL_USER'] = 'root'
app.config['MYSQL_PASSWORD'] = 'password'
app.config['MYSQL_DB'] = 'geekprofile'

```

```

mysql = MySQL(app)

@app.route('/')
@app.route('/login', methods=['GET', 'POST'])
def login():
    msg = ''
    if request.method == 'POST' and 'username' in request.form and 'password' in request.form:
        username = request.form['username']
        password = request.form['password']
        cursor = mysql.connection.cursor(MySQLdb.cursors.DictCursor)
        cursor.execute(
            'SELECT * FROM accounts WHERE username = % s \
            AND password = % s', (username, password, ))
        account = cursor.fetchone()
        if account:
            session['loggedin'] = True
            session['id'] = account['id']
            session['username'] = account['username']
            msg = 'Logged in successfully !'
            return render_template('index.html', msg=msg)
        else:
            msg = 'Incorrect username / password !'
    return render_template('login.html', msg=msg)

@app.route('/logout')
def logout():

    session.pop('loggedin', None)
    session.pop('id', None)
    session.pop('username', None)
    return redirect(url_for('login'))

@app.route('/register', methods=['GET', 'POST'])
def register():
    msg = ''
    if request.method == 'POST' and 'username' in request.form and 'password' in request.form and 'email' in request.form and 'address' in request.form and 'city' in request.form and 'country' in request.form and 'postalcode' in request.form and 'organisation' in request.form:
        username = request.form['username']
        password = request.form['password']
        email = request.form['email']

```

```

organisation = request.form['organisation']
address = request.form['address']
city = request.form['city']
state = request.form['state']
country = request.form['country']
postalcode = request.form['postalcode']
cursor = mysql.connection.cursor(MySQLdb.cursors.DictCursor)
cursor.execute(
    'SELECT * FROM accounts WHERE username = % s', (username, ))
account = cursor.fetchone()
if account:
    msg = 'Account already exists !'
elif not re.match(r'^@+@[^@]+\.[^@]+', email):
    msg = 'Invalid email address !'
elif not re.match(r'[A-Za-z0-9]+', username):
    msg = 'name must contain only characters and numbers !'
else:
    cursor.execute('INSERT INTO accounts VALUES \
        (NULL, % s, % s, % s, % s, % s, % s, % s, % s, % s)',
        (username, password, email,
         organisation, address, city,
         state, country, postalcode, ))
    mysql.connection.commit()
    msg = 'You have successfully registered !'
elif request.method == 'POST':
    msg = 'Please fill out the form !'
return render_template('register.html', msg=msg)

```

```

@app.route("/index")
def index():
    if 'loggedin' in session:
        return render_template("index.html")
    return redirect(url_for('login'))

```

```

@app.route("/display")
def display():
    if 'loggedin' in session:
        cursor = mysql.connection.cursor(MySQLdb.cursors.DictCursor)
        cursor.execute('SELECT * FROM accounts WHERE id = % s',
            (session['id'], ))
        account = cursor.fetchone()
        return render_template("display.html", account=account)
    return redirect(url_for('login'))

```

```

@app.route("/update", methods=['GET', 'POST'])
def update():

```

```

msg = ''
if 'loggedin' in session:
    if request.method == 'POST' and 'username' in request.form
    and 'password' in request.form and 'email' in request.form and
    'address' in request.form and 'city' in request.form and 'country'
    in request.form and 'postalcode' in request.form and
    'organisation' in request.form:
        username = request.form['username']
        password = request.form['password']
        email = request.form['email']
        organisation = request.form['organisation']
        address = request.form['address']
        city = request.form['city']
        state = request.form['state']
        country = request.form['country']
        postalcode = request.form['postalcode']
        cursor = mysql.connection.cursor(MySQLdb.cursors.DictCursor)
        cursor.execute(
            'SELECT * FROM accounts WHERE username = % s',
            (username, ))
        account = cursor.fetchone()
        if account:
            msg = 'Account already exists !'
        elif not re.match(r'^@+@[^@]+\.[^@]+', email):
            msg = 'Invalid email address !'
        elif not re.match(r'[A-Za-z0-9]+', username):
            msg = 'name must contain only characters and numbers !'
        else:
            cursor.execute('UPDATE accounts SET username =% s,\
password =% s, email =% s, organisation =% s, \
address =% s, city =% s, state =% s, \
country =% s, postalcode =% s WHERE id =% s', (
                username, password, email, organisation,
                address, city, state, country, postalcode,
                (session['id'], ), ))
            mysql.connection.commit()
            msg = 'You have successfully updated !'
        elif request.method == 'POST':
            msg = 'Please fill out the form !'
        return render_template("update.html", msg=msg)
    return redirect(url_for('login'))

if __name__ == "__main__":
    app.run(host="localhost", port=int("5000"))

```

Step 5: Create the folder **'templates'**. create the files **'index.html'**, **'display.html'**, **'update.html'**, **'login.html'**, and **'register.html'** inside the **'templates'** folder.

Step 6: Open the **'login.html'** file and write the code given below. In **'login.html'**, we have two fields i.e. username and password. When a user enters the correct username and password, it will route you to the index page otherwise **'Incorrect username/password'** is displayed.

html

```
<!--Store this code in 'login.html' file inside the 'templates' folder-->
<html>
  <head>
    <meta charset="UTF-8">
    <title> Login </title>
    <link rel="stylesheet" href="../static/style.css">
  </head>
  <body>
    <div class="logincontent" align="center">
      <div class="logintop">
        <h1>Login</h1>
      </div></br></br></br></br>
      <div class="loginbottom">
        <form action="{{ url_for('login')}}" method="post" autocomplete="off"
        <div class="msg">{{ msg }}</div>
        <input type="text" name="username" placeholder="Enter Your Username"
        <input type="password" name="password" placeholder="Enter Your Passwo
        <input type="submit" class="btn" value="Login">
      </form></br></br>
      <p class="worddark">New to this page? <a class="worddark" href="{{ ur
      </div>
    </div>
  </body>
</html>
```

Step 7: Open **'register.html'** file and write the code given below. In **'register.html'**, we have nine fields i.e. username, password, email, organisation, address, city, state, country, postal code. When user enters all the information, it stored the data in the database and **'Registration successful'** is displayed.

html

```
<!--Store this code in 'register.html' file inside the 'templates' folder-->
<html>
  <head>
    <meta charset="UTF-8">
    <title> register </title>
    <link rel="stylesheet" href="../static/style.css">
  </head>
  <body>
    <div class="registercontent" align="center">
      <div class="registertop">
        <h1>Register</h1>
      </div><br></br>
      <div class="registerbottom">
        <form action="{{ url_for('register')}}" method="post" autocomplete="o
        <div class="msg">{{ msg }}</div>
        <input type="text" name="username" placeholder="Enter Your Username"
        <input type="password" name="password" placeholder="Enter Your Passwo
        <input type="email" name="email" placeholder="Enter Your Email ID" cl
        <input type="text" name="organisation" placeholder="Enter Your Organi
        <input type="text" name="address" placeholder="Enter Your Address" cl
        <input type="text" name="city" placeholder="Enter Your City" class="t
        <input type="text" name="state" placeholder="Enter Your State" class=
        <input type="text" name="country" placeholder="Enter Your Country" cl
        <input type="text" name="postalcode" placeholder="Enter Your Postal C
        <input type="submit" class="btn" value="Register">
      </form>
      <p class="worddark">Already have account? <a class="worddark" href="{
    </div>
  </body>
</html>
```

Step 8: Open 'index.html' file and write the code given below. When a user logs in successfully, this page is displayed, and 'Logged in successful' is displayed.

html

```
<!--Store this code in 'index.html' file inside the 'templates' folder-->
<html lang="en">
```

```

<head>
  <title>index</title>
  <link rel="stylesheet" href="../static/style.css">
</head>

<body bgcolor="#e6ffee">
  <div class="one">
    <div class="two">
      <h1>Side Bar</h1>
      <ul>
        <li class="active"><a href="{{url_for('index')}}">Index</a></li>
        <li><a href="{{url_for('display')}}">Display</a></li>
        <li><a href="{{url_for('update')}}">Update</a></li>
        <li><a href="{{url_for('logout')}}">Log out</a></li>
      </ul>
    </div>
    <div class="content" align="center">
      <div class="topbar">
        <h2>Welcome!! You are in Index Page!! </h2>
      </div><br><br>
      <div class="contentbar">
        <div class="msg">{{ msg }}</div>
      </div>
    </div>
  </div>
</body>
</html>

```

Step 9: Open 'display.html' file and write the code given below. Here, the user information stored in database are displayed.

html

<!--Store this code in 'display.html' file inside the 'templates' folder-->

```

<html lang="en">
  <head>
    <title>display</title>
    <link rel="stylesheet" href="../static/style.css">
  </head>
  <body bgcolor="#e6ffee">
    <div class="one">
      <div class="two">

```



```

<h1>Side Bar</h1>
<ul>
    <li><a href="{{url_for('index')}}">Index</a></li>
    <li class="active"><a href="{{url_for('display')}}">Display</a></li>
    <li><a href="{{url_for('update')}}">Update</a></li>
    <li><a href="{{url_for('logout')}}">Log out</a></li>
</ul>
</div>
<div class="content" align="center">
    <div class="topbar">
        <h2>Welcome!! You are in Display Page!! </h2>
    </div> </br>
    <div class="contentbar">
        <h1>Your Details</h1> </br>
        {% block content %}
            <div class="border">
                <table class="worddark"></br></br></br></br>
                <tr>
                    <td>Username:</td>
                    <td>{{ account['username'] }}</td>
                </tr>
                <tr>
                    <td>Password:</td>
                    <td>{{ account['password'] }}</td>
                </tr>
                <tr>
                    <td>Email ID:</td>
                    <td>{{ account['email'] }}</td>
                </tr>
                <tr>
                    <td>Organisation:</td>
                    <td>{{ account['organisation'] }}</td>
                </tr>
                <tr>
                    <td>Address:</td>
                    <td>{{ account['address'] }}</td>
                </tr>
                <tr>
                    <td>City:</td>
                    <td>{{ account['city'] }}</td>
                </tr>
                <tr>
                    <td>State:</td>
                    <td>{{ account['state'] }}</td>
                </tr>
                <tr>
                    <td>Country:</td>
                    <td>{{ account['country'] }}</td>
                </tr>
            </table>
        </div>
    </div>
</div>

```

```

        <tr>
            <td>Postal code:</td>
            <td>{{ account['postalcode'] }}</td>
        </tr>
    </table>
</div>
{% endblock %}
</div>

</div>
</div>
</body>
</html>

```

Step 10: Open 'update.html' file and write the code given below. The user can update his/her data which also updates the database.

html

```

<!--Store this code in 'update.html' file inside the 'templates' folder-->
<html lang="en">
    <head>
        <title>update</title>
        <link rel="stylesheet" href="../static/style.css">
    </head>
    <body bgcolor="#e6ffee">
        <div class="one">
            <div class="two">
                <h1>Side Bar</h1>
                <ul>
                    <li><a href="{{url_for('index')}}">Index</a></li>
                    <li><a href="{{url_for('display')}}">Display</a></li>
                    <li class="active"><a href="{{url_for('update')}}">Update</a>
                    <li><a href="{{url_for('logout')}}">Log out</a></li>
                </ul>
            </div>
            <div class="content" align="center">
                <div class="topbar">
                    <h2>Welcome!! You are in Update Page!! </h2>
                </div><br><br>
                <div class="contentbar">
                    <h1>Fill Your Details to Update</h1><br>
                    <form action="{{ url_for('update') }}" method="post" autocomplete="of
                        <div class="msg">{{ msg }}</div>

```

```

<input type="text" name="username" placeholder="Enter Your Us
<input type="password" name="password" placeholder="Enter You
<input type="email" name="email" placeholder="Enter Your Emai
<input type="text" name="organisation" placeholder="Enter You
<input type="text" name="address" placeholder="Enter Your Add
<input type="text" name="city" placeholder="Enter Your City"
<input type="text" name="state" placeholder="Enter Your State
<input type="text" name="country" placeholder="Enter Your Cou
<input type="text" name="postalcode" placeholder="Enter Your
<input type="submit" class="btn" value="Update">

</form>
</div>

</div>
</div>
</body>
</html>

```

Step 11: Create the folder 'static'. create the file 'style.css' inside the 'static' folder and paste the given CSS code.

CSS

/*Store this code in 'style.css' file inside the 'static' folder*/

```

.logincontent{
    margin: 0 auto;
    height: 500px;
    width: 400px;
    background-color: #e6ffee;
    border-radius: 10px;
}

.registercontent{
    margin: 0 auto;
    height: 720px;
    width: 400px;
    background-color: #e6ffee;
    border-radius: 10px;
}

.logintop{
    height: 60px;
    width: 400px;
}

```

```
background-color: #009933;
color: #ffffff;
}

.registertop{
height: 60px;
width: 400px;
background-color: #009933;
color: #ffffff;
}

.textbox{
padding: 10px 40px;
background-color: #009933;
border-radius: 10px;
}

::placeholder {
color: #FFFFFF;
opacity: 1;
font-style: oblique;
font-weight: bold;
}

.btn {
padding: 10px 40px;
background-color: #009933;
color: #FFFFFF;
font-style: oblique;
font-weight: bold;
border-radius: 10px;
}

.worddark{
color: #009933;
font-style: oblique;
font-weight: bold;
}

.wordlight{
color: #FFFFFF;
font-style: oblique;
font-weight: bold;
}

*{
margin: 0;
padding: 0;
box-sizing: border-box;
```

```
list-style: none;
text-decoration: none;
font-family: 'Josefin Sans', sans-serif;
}

.one{
  display: flex;
  position: relative;
}

.one .two{
  width: 225px;
  height: 100%;
  background: #009933;
  padding: 30px 0px;
  position: fixed;
}

.one .two h1{
  color: #fff;
  text-transform: uppercase;
  text-align: center;
  margin-bottom: 30px;
  font-style: oblique;
  font-weight: bold;
}

.one .two h2{
  color: #fff;
  text-align: center;
}

.one .two .active{
  background: #0a8032;
}

.one .two ul li{
  text-align: center;
  padding: 15px;
  border-bottom: 0.1px solid white;
  border-top: 0.1px solid white;
}

.one .two ul li a{
  color: #ffffff;
  display: block;
}

.one .two ul li a .side{
```

```
        width: 25px;
        align:center;
    }

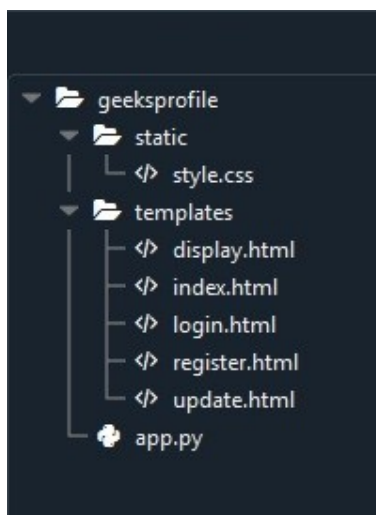
    .one .content{
        width: 100%;
        margin-left: 200px;
    }

    .one .content .topbar{
        text-align: center;
        padding: 20px;
        background: #00b33c;
        color: white;
    }

    .one .content .contentbar{
        margin: auto;
    }

    .one .content .contentbar h1{
        color: #11a844;
        text-align: center;
        font-style: oblique;
        font-weight: bold;
    }
}
```

Step 12: The project structure will look like this.



Run the Project

Step-1: Run the server.

Step-2: Browse the URL 'localhost:5000'.

Step-3: The output web page will be displayed.

Testing of the Application

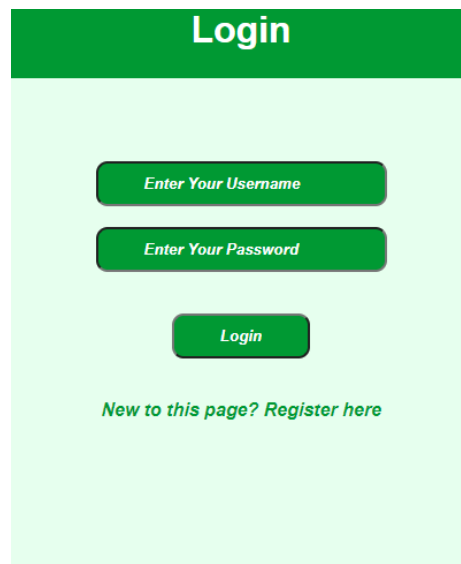
Step-1: If you are new user, go to sign up page and fill the details.

Step-2: After registration, go to login page. Enter your username and password and sign in.

Step-3: If your login is successful, you will be moved to index page and your name will be displayed.

Step-4: You can view your profile details in display page and also you can update your details in update page.

Output: Login page:

A screenshot of a web application's login page. The page has a light green background. At the top, there is a dark green header bar with the word "Login" in white text. Below the header, there are three input fields stacked vertically. The first two fields are labeled "Enter Your Username" and "Enter Your Password" in a light green font. The third field is a button labeled "Login" in a light green font. Below the input fields, there is a link that says "New to this page? Register here" in a light green font.

Register page:

Register

Enter Your Username

Enter Your Password

Enter Your Email ID

Enter Your Organisation

Enter Your Address

Enter Your City

Enter Your State

Enter Your Country

Enter Your Postal Code

Register

Already have account? [Login here](#)

If login is successful, Index Page:

SIDE BAR

Index

Display

Update

Log out

Welcome!! You are in Index Page!!

Logged in successfully!

Update Page:

SIDE BAR

Index

Display

Update

Log out

Welcome!! You are in Update Page!!

Fill Your Details to Update

Enter Your Username

Enter Your Password

Enter Your Email ID

Enter Your Organisation

Enter Your Address

Enter Your City

Enter Your State

Enter Your Country

Enter Your Postal Code

Update

Active
Go to Se

Before updation, Display page:

SIDE BAR

[Index](#)
[Display](#)
[Update](#)
[Log out](#)

Welcome!! You are in Display Page!!

Your Details

Username: Alex
Passworde: alex@123
Email ID: alex@gmail.com
Organisation: Grand University
Address: No. 12, Base street
City: Mumbai
State: Maharashtra
Country: India
Postal code: 368292

After updation, Display page:

SIDE BAR

[Index](#)
[Display](#)
[Update](#)
[Log out](#)

Welcome!! You are in Display Page!!

Your Details

Username: Ram
Passworde: ram@kumar
Email ID: ram@gmail.com
Organisation: Canal University
Address: No. 297, Base street
City: Bangalore
State: Karnataka
Country: India
Postal code: 443930

Database – Before update:

Query 1 accounts

Limit to 1000 rows

1 • `SELECT * FROM geekprofile.accounts;`

	id	username	password	email	organisation	address	city	state	country	postalcode
▶	1	Alex	alex@123	alex@gmail.com	Grand University	No. 12, Base street	Mumbai	Maharashtra	India	368292
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Database – After update:

Query 1 accounts accounts x

Limit to 1000 rows

1 • `SELECT * FROM geekprofile.accounts;`

Result Grid

	id	username	password	email	organisation	address	city	state	country	postalcode
▶	1	Ram	ram@kumar	ram@gmail.com	Canal University	No. 297, Base street	Bangalore	Karnataka	India	443930
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

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!



venni...



12

Previous Article

Python Flask Projects with Source Code
(Beginners to Advanced)

Next Article

Twitter Sentiment Analysis WebApp
Using Flask

Similar Reads

Create GitHub API to fetch user profile image and number of repositories...

GitHub is where developers shape the future of software, together, contribute to the open-source community, manage Git repositories, etc. It is one of the most...

5 min read

Documenting Flask Endpoint using Flask-Autodoc

Documentation of endpoints is an essential task in web development and being able to apply it in different frameworks is always a utility. This article discusses...

4 min read

Minify HTML in Flask using Flask-Minify

Flask offers HTML rendering as output, it is usually desired that the output HTML should be concise and it serves the purpose as well. In this article, we would...

12 min read

How to use Flask-Session in Python Flask ?

Flask Session - Flask-Session is an extension for Flask that supports Server-side Session to your application. The Session is the time between the client logs in to...

4 min read

How to Integrate Flask-Admin and Flask-Login

In order to merge the admin and login pages, we can utilize a short form or any other login method that only requires the username and password. This is know...

8 min read

Flask URL Helper Function - Flask url_for()

In this article, we are going to learn about the flask url_for() function of the flask URL helper in Python. Flask is a straightforward, speedy, scalable library, used f...

11 min read

Login and Registration Project Using Flask and MySQL

Project Title: Login and registration Project using Flask framework and MySQL Workbench. Type of Application (Category): Web application. Introduction: A...

6 min read

Load Balancing Flask Application using Nginx and Docker

Load balancing means efficiently distributing the incoming traffic to different server instances. Nginx is open-source software that can be used to apply load...

5 min read

Create Postman collection from Flask application using flask2postman

Prerequisite: Introduction to Postman, First App using Flask Since Postman is gaining popularity in the development domain, this article explains a way in...

3 min read

Flask NEWS Application Using Newsapi

In this article, we will create a News Web Application using Flask and NewsAPI. The web page will display top headlines and a search bar where the user can...

7 min read

Article Tags :

[Python](#)[Web Technologies](#)[Flask Projects](#)[Python Flask](#)

Practice Tags :

[python](#)

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](#)

Languages

[Python](#)[Java](#)[C++](#)[PHP](#)[GoLang](#)

GFG Corporate Solution
Placement Training Program
GeeksforGeeks Community

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

Web Technologies

HTML
CSS
JavaScript
TypeScript
ReactJS
NextJS
Bootstrap
Web Design

Computer Science

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

System Design

High Level Design
Low Level Design
UML Diagrams
Interview Guide
Design Patterns
OOAD
System Design Bootcamp
Interview Questions

School Subjects

Mathematics
Physics

Data Science & ML

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

Python Tutorial

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

DevOps

Git
Linux
AWS
Docker
Kubernetes
Azure
GCP
DevOps Roadmap

Interview Preparation

Competitive Programming
Top DS or Algo for CP
Company-Wise Recruitment Process
Company-Wise Preparation
Aptitude Preparation
Puzzles

GeeksforGeeks Videos

DSA
Python

Chemistry	Java
Biology	C++
Social Science	Web Development
English Grammar	Data Science
Commerce	CS Subjects
World GK	

@GeeksforGeeks, Sanchhaya Education Private Limited, All rights reserved