97 Flask interview questions to hire top engineers

Questions

1. What is Flask, in simple words?
2. Can you name a few advantages of using Flask?
3. What does 'WSGI' mean, and why is it important for Flask?
4. How do you create a simple 'Hello, World!' application in Flask?
5. What is a Flask route, and how do you define one?
6. How do you pass data from a Flask route to an HTML template?
7. What is Jinja2, and how does Flask use it?
8. How can you create dynamic web pages with Flask?
9. What is a Flask extension? Give an example.
10. Explain how to handle user input in Flask using forms.
11. How do you install Flask on your computer?
12. What is the purpose of the 'app.run()' method in Flask?
13. How do you set up a development environment for Flask?
14. What are the different HTTP methods, and how do you handle them in Flask routes?
15. How do you use sessions in Flask to store user data?
16. How can you structure a Flask application into multiple files or folders?
17. Explain how to connect a Flask application to a database.
18. What are Flask blueprints, and why would you use them?
19. How do you handle errors and exceptions in Flask?
20. Describe the process of deploying a Flask application to a web server.
21. What are some common security considerations when developing Flask applications?
22. Explain the difference between 'url_for' and hardcoding URLs in Flask templates. Why is 'url_for' preferred?
23. What is Flask, in super simple words?
24. Imagine your computer is a kitchen. What part of Flask helps you decide what dish to make when someone asks for it?
25. Can you explain what a 'route' is in Flask, like you're explaining directions to your friend's house?
26. What's the easiest way to show 'Hello World' on a webpage using Flask?
27. If a website needs to remember something about a visitor (like their name), what's a simple way Flask can do that?
28. What is the difference between GET and POST methods?
29. How would you create a basic HTML form and send the data to the server using Flask?
30. What is the purpose of Flask's 'debug mode' and when should you use it?
31. Explain Flask's 'template' system and why it's helpful.
32. How do you link CSS files to your HTML templates in Flask?
33. What is a 'virtual environment' and why is it important for Flask projects?
34. What are common HTTP methods, and can you give a basic use-case of each?
35. How can you get data from a URL in Flask? (e.g., getting the ID from '/users/<id>')?
36. What's one thing you would check if your Flask app isn't showing changes you made to the code?
37. How would you handle a simple error (like a user entering the wrong information) in Flask?
38. Can you describe a basic project structure for a Flask application?
39. What is the purpose of url_for() in Flask?
40. How can you serve static files like images or JavaScript files using Flask?
41. Explain how you'd use a conditional statement within a Jinja2 template (e.g., if/else).
42. How can you pass variables from your Flask route to your HTML template?
43. What are some common things to include in a requirements.txt file?
44. If you have to connect with a database, where would you put the connection string/code in your flask app?
45. What is a session in Flask, and how do you typically use it?
46. How do you install Flask, step by step, on your computer?
47. Explain what an 'endpoint' is in the context of a Flask API.
48. Describe the role of WSGI (Web Server Gateway Interface) in a Flask application.
49. How would you implement user authentication in a Flask application, detailing the steps involved and security considerations?
50. Explain how to use Flask-WTF for form handling, including validation and CSRF protection.
51. Describe how to structure a larger Flask application using blueprints, and why this is beneficial.
52. How can you implement caching in a Flask application to improve performance, and what are some different caching strategies?
53. Explain how to use Flask's session management, including how to store and retrieve user-specific data.
54. Describe how to handle file uploads in Flask, including security considerations and how to store the files.
55. How do you implement error handling in Flask, including custom error pages and logging?
56. Explain how to use Flask's testing framework to write unit tests for your application.
57. Describe how to deploy a Flask application to a production environment, such as using WSGI servers like Gunicorn or uWSGI.
58. How can you integrate a database (like PostgreSQL or MySQL) with a Flask application using SQLAlchemy?
59. Explain how to use Flask-Migrate for database migrations.
60. Describe how to implement RESTful APIs using Flask, including different HTTP methods and request/response formats.
61. How can you secure a Flask application against common web vulnerabilities, such as XSS and SQL injection?
62. Explain how to use Flask signals to trigger events in your application.
63. Describe how to use Flask's command-line interface (CLI) to create custom commands.
64. How can you implement background tasks in a Flask application using Celery?
65. Explain how to use Flask-RESTful to build REST APIs, and how it differs from using Flask directly.
66. Describe how to implement rate limiting in a Flask application to prevent abuse.
67. How can you use Flask to serve static files, and what are the performance considerations?
68. Explain how to use Flask's template inheritance to create reusable templates.
69. Describe how to implement internationalization (i18n) and localization (l10n) in a Flask application.
70. How can you monitor and log the performance of a Flask application in a production environment?
71. Explain how to use Flask-Mail to send emails from your application.
72. Describe how to implement WebSockets in a Flask application using Flask-SocketIO.
73. How can you use Flask to create a single-page application (SPA) using a JavaScript framework like React or Vue.js?
74. Explain how you would handle different environments (development, testing, production) with different configurations in Flask?
75. Explain the architectural patterns you've used with Flask and why you chose them.
76. Describe your experience with Flask extensions and how they can improve code maintainability.
77. How have you handled database migrations in Flask applications, and what tools did you use?
78. What strategies have you used to optimize Flask application performance, considering bottlenecks like database queries or slow API calls?
79. Discuss your experience with different deployment strategies for Flask applications, such as using WSGI servers or containerization.
80. How do you ensure application security in Flask, specifically addressing common vulnerabilities like Cross-Site Scripting (XSS) and SQL Injection?
81. Explain your approach to testing Flask applications, including unit, integration, and end-to-end tests.
82. Describe a time you had to debug a complex issue in a Flask application. What tools and techniques did you employ?
83. How do you handle user authentication and authorization in Flask applications, considering different authentication methods like OAuth or JWT?
84. Discuss your experience with background tasks and asynchronous processing in Flask applications using tools like Celery or Redis Queue.
85. What's your experience with implementing RESTful APIs using Flask, including versioning and documentation?
86. How have you integrated Flask applications with other services or APIs, and what challenges did you encounter?
87. Describe how you would approach monitoring and logging in a production Flask application.
88. How do you manage configuration settings across different environments (development, staging, production) in a Flask application?
89. What are the advantages and disadvantages of using Flask compared to other Python web frameworks like Django?
90. Describe how you've implemented caching strategies in Flask to improve performance and reduce database load.
91. How do you handle different forms of data validation and sanitization in Flask applications?
92. Describe your experience with writing custom Flask extensions and what problems they solved.
93. How do you approach scaling Flask applications to handle increased traffic and user load?
94. What are your preferred methods for handling file uploads and storage in Flask applications?
95. Explain how you would design and implement a microservices architecture using Flask.