Sharing a list of questions I have been asked. System design questions were very popular, please don't start with designing database tables. Start with overview like I will start by gathering user stories, understanding the client needs, if frontend has a lot of dynamic parts I will go for React, if static then HTML, CSS, JS.

On the backend - mention that instead of having a monolithic app I will opt for **Microservices** (you absolutely have to know what is microservices, here is a good link to start- you don't need to watch the entire video, first 20mins are good for microservices - https://www.youtube.com/watch?v=1xo-0gCVhTU&t=411s) that is dividing the business logic into small parts so that fault tolerance is high, if one part fails the entire backend won't fail.

For database- I will start by understanding client data, if it's large and simple then opt for NoSQL, if data has lots of complex relationships then go for SQL, if data size increases- I will take a layered approach- I will start with Indexing for better search, then create Views for popular queries, then create master slave architecture where you write to master and slave pulls data, if master fails- one of the slave gets nominated as the master, last is database sharding (super important to know what is database sharding). Here is a good one on **database sharding** https://youtu.be/5faMjKuB9bc. This is what they expect in system design questions.

System design prep - https://github.com/donnemartin/system-design-primer

Database study material for interviews:

Indexing - https://www.tutorialspoint.com/postgresql/postgresql_indexes
Views - https://www.tutorialspoint.com/postgresql/postgresql_views
Transactions - https://www.tutorialspoint.com/postgresql/postgresql_transactions
https://www.postgresql.org/docs/8.3/tutorial-transactions.html
A guide to understanding database scaling patterns -
https://www.freecodecamp.org/news/understanding-database-scaling-patterns/Here is a link to 5 Tips for System Design Interviews -
https://www.youtube.com/watch?v=CtmBGH8MkX4 .

Lots of good videos available on YouTube for system design questions. Here is one on System Design for Twitter
https://www.youtube.com/watch?v=KmAyPUv9gOY&t=1s

Learn about Firewall and how to block a particular malicious IP.
https://www.hostinger.com/tutorials/iptables-tutorial

Security.
https://www.freecodecamp.org/news/how-we-handled-a-denial-of-service-attack-a-simple-security-lesson-8cdd542d4def/

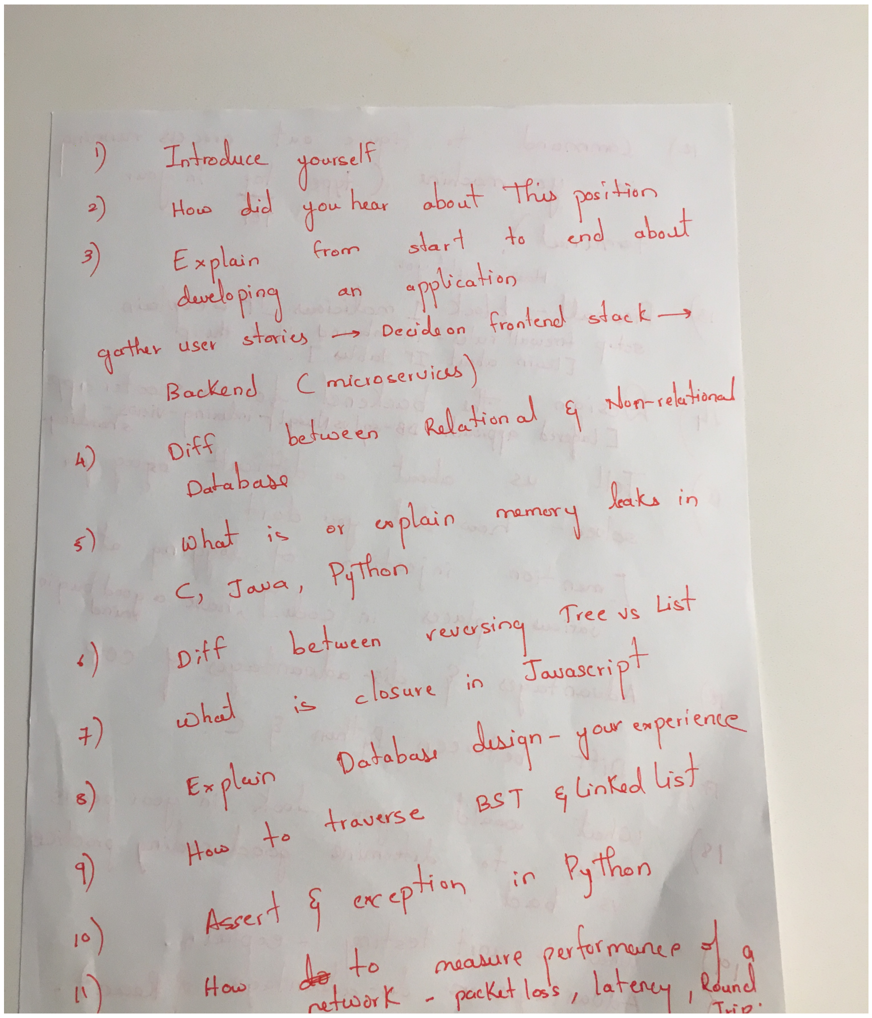
30 Linux Commands Every User Should Know.
https://www.hostinger.com/tutorials/linux-commands

https://www.hostinger.com/tutorials/manage-and-list-services-in-linux/

What is an API Gateway? -
https://www.youtube.com/watch?v=vHQqQBYJtLI&t=273s

For take home code challenges always provide unit testing, comments and a detailed README explaining your approach for your code challenge. Even if they don't ask for unit testing, it's expected. This could make a big difference.

Here are a few questions that I missed below:
1. Diff between Heap and Stack
2. Pros and cons of Microservices
3. Explain try and catch in Javascript (know this really well along with async and await, behavior of **this** keyword in JS) - awesome book on JS- https://javascript.info
4. Explain TCP and HTTP
5. What do you know about Amazon Web Services(AWS) or Google Cloud Platform(GCP) - definitely understand and study the overview of either of the 2 services, most Full-Stack/Backend positions expect you to know things like what is a load balancer, EC2 instance, lambda functions.
6. What is caching - to learn caching try integrating Redis in your Lab project if you have time. It's pretty easy. Here is a good link to understand and integrate Redis. https://youtu.be/ECz6Mv3T7Ec
7. What are cookies, explain.
8. When to use React and when not
9. What are your strengths and weaknesses
10. Have you ever had a disagreement with your manager-how did you approach that



1) Introduce yourself

2) How did you hear about This position

3) Explain from start to end about developing an application
gather user stories → Decide on frontend stack →
Backend (microservices)

4) Diff between Relational & Non-relational Database

5) what is or explain memory leaks in C, Java, Python

6) Diff between reversing Tree vs List

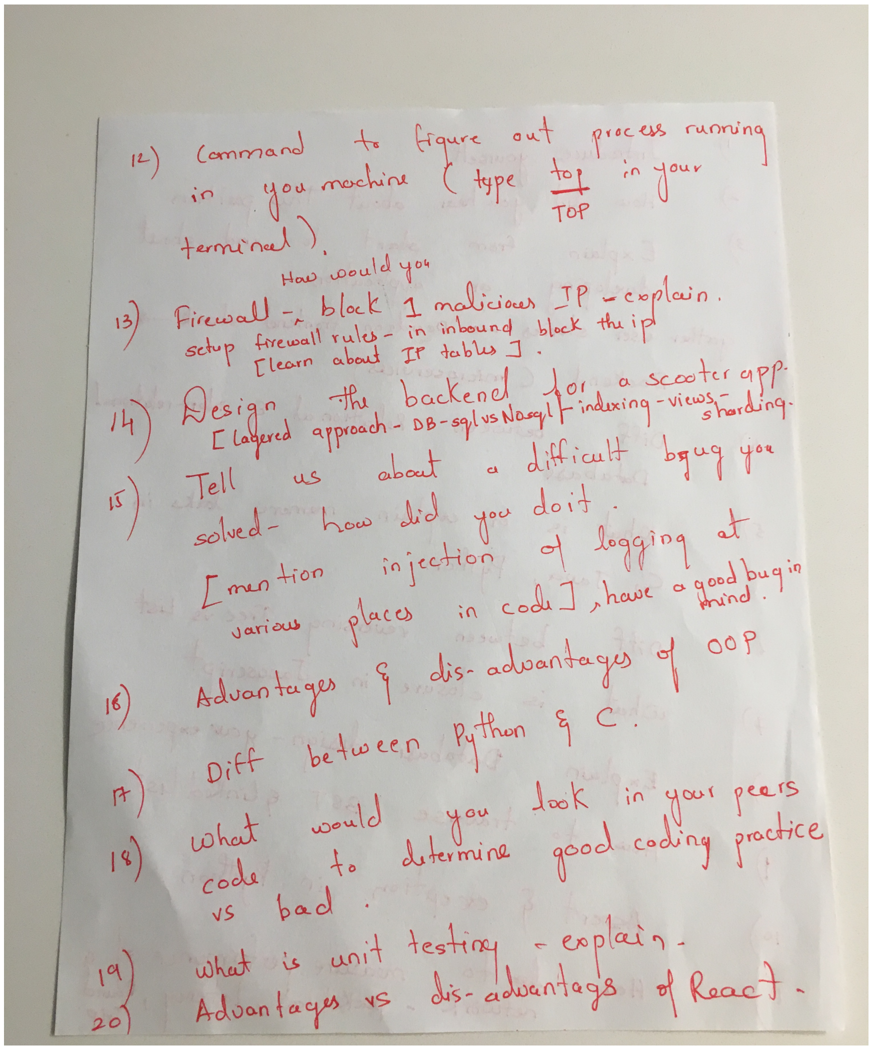
7) what is closure in Javascript

8) Explain Database design - your experience

9) How to traverse BST & Linked list

10) Assert & exception in Python

11) How ~~do~~ to measure performance of a network - packet loss, latency, Round Trip



12) Command to figure out process running in you machine (type top in your terminal).
TOP

How would you
13) Firewall -, block 1 malicious IP - explain.
setup firewall rules - in inbound block the ip
[learn about IP tables].

14) Design the backend for a scooter app.
[Layered approach - DB - sql vs Nosql] - indexing - views - sharding.

15) Tell us about a difficult bug you solved - how did you doit.
[mention injection of logging at various places in code], have a good bug in mind.

16) Advantages & dis-advantages of OOP

17) Diff between Python & C.

18) what would you look in your peers code to determine good coding practice vs bad.

19) what is unit testing - explain.

20) Advantages vs dis-advantags of React.