**Would You Hire Me as a Backend Developer?**

I simulated an interview with a U.S startup recruiter using ChatGPT to simulate the recruiter role.

Just hard questions about system design, breaking production, scaling architecture, and writing test plans like it’s my money on the line.

**Q1: What’s the hardest backend system you built?**

The hardest backend was the task management system as I had to do reliability, scalability, security and speed at the same time in one architecture and as you know they are 4 elements of a software that are most likely to be included in trade-offs whether this or that or that you can't do all the 3 of them at the same time but an idea popped up in my mind which is combining the SQL and NoSQL the SQL will be for the structure giving me more security transactions and the truth with relations between the tables which make it good for the security and performance and it will be more reliable but when it comes to speed joins and freeze would be too exhausting for the resources in terms of hardware and SQL database management so on the other side where I need the fast memory access and the speed I used the NoSQL with no more joins to collect the data and with no deep documents and wise usage of indexes the 4 elements where achieved with no trade-offs but the complexity and the risk of the data not being updated but did I leave it like that a living bug that can eat the system? no this was solved using the bridges the takes the data and splits it to the SQL and NoSQL databases making sure both are updated and taking data from them ensuring the data integrity and making the control of 2 different topologies easier and smoother with modularity also it gave the software a sweet and good spot for further growth in future and APIs that has meaningful namings and depends on functions so even the complexity was solved through good readability and clean code with some more effort in documentation complexity won't even be a concern anymore.

👉 [GitHub Repo](https://github.com/abdelrahman-hosam/task_management_system)

**Q2: You ship a feature that breaks production. What do you do?**

As a team member I will always be the team player that loves to make a usable good software with 3 or 4 names on the project rather than taking the full ownership just to take the credits so I will put a plan find 2 or 3 teammates who can help me and start working with them in increments that are seen by the team leader as there is no team with no manager or mastermind then start working on those increments not as a solo developer but giving each one what they can shine in and be special and on the technical level I won't mind some overtime and a good planning before heading to implementation which will maybe result in bugs.

**Q3: We send 10,000+ reports at 3 AM daily. How do you avoid overload?**

First let's think what makes the overload and overload in this case I would diagnose it as slow because the database hits, async functions a lot of them and bad resources handling as in bandwidth for real-time communication so what am I suggesting keeping all of those to minimum but how let's take it a step by step first the database hits can be kept to minimum by having the maximum of 2 hits per function which means the data to be inserted it should be sanitized within the system before going to the database it should be validated before going to the database also and do one hit which is the insertion which is usually fast in terms of some NoSQL databases like mongoDB which I personally suggest in this case or fast writes as in postgreSQL which I don't recommend because it comes in weak handling of the reading functionalities which will lead to a very fast writing but little bit slower reading and this is not good for the real-time communication for the async put a rules if you can sync then sync if there is a more complex operation - with a term it doesn't come with a trade-off with the speed we need that - then it's prioritized if an asynchronous is taking a lot of functions implement it with labeling it under scope to be watched later for making it less expensive or sync and for the bandwidth management a good handling for the sockets would be more than enough killers for zombie sockets, no idle connections, permanent connections and rooms are for chats or convos but for reports we will use temporary ones that are killed once it's sent and having the big socket in sub-sockets it won't do too much when it comes to speed but isolation for reports which have too much sensitive information would be more than excellent.

**Q4: You built a money transfer API. How do you test it?**

First it would be unit testing for each part of the system then combine the units till I can test the system as whole and I will test the connection failure before transaction start, while transaction, when the system has already done the database update but before sending the feedback besides all kinds of fraud and manipulation through having the client ID without their permission testing the authentication layers and types of it used I will try to open the account from another device testing the different layers of security through MAC addresses maybe take it a little bit far and open the account from 2 different devices at the same time also testing the security then I will make 2000 or more virtual users make them active for 12 hours with no stopping for the operations testing the reliability of the system and the handling of the requests - does it stop the user from sending more requests at some point - the last test would be for sending 2 or more requests for the same account at the same time testing the transaction system and if the requests freeze and give the updated info.

**Q5: PM requests a feature that’ll 5x DB load. How do you explain why it’s risky?**

Real life examples are a lot let's say they like football I would tell them yeah bringing a star player to the team would be a too good update and strong addition to the team but it will break our resources which means it will empty our pockets and maybe we will end up with can't even afford the team players salaries at the end of the season.

**Q6: You’re hired. What’s your 3-month growth plan?**

I will first see who are good with the idea of sharing experience and giving time to grow from the upper level if I'm junior then mid-seniors or seniors make a good connection with them and tell them that I try to hit a new level but at this point guidance is needed with listening more than talking when someone with more experience talks and says something whether technical or non-tech like how they handled the team, a conflict, salaries and all of that with more understanding of the domain in the tech side with more courses and personal projects and non-tech in reading books about it as in fundamentals to be more aware of what trade-offs I should do or avoid and last thing a small note that I can write my mistakes and how was it solved.

📬 **If you're interested, feel free to send me 2 or more questions in my DMs.**  
Happy to answer.

— Abdelrahman Hossam  
🔗 [linkedin.com/in/abdelrahman-hossam](https://linkedin.com/in/abdelrahman-hossam)  
📎 [github.com/abdelrahman-hosam/task\_management\_system](https://github.com/abdelrahman-hosam/task_management_system)