Hello,

Thank you so much for the second interview today. My name is Hasitha, and I’m from Kotmale. I have a bachelor’s degree in computer science from Eastern University of Sri Lanka, Trincomalee Campus, where I graduated with second-class lower division. During my time there, I was actively involved in student leadership, serving as the President of the Sports Council and the Secretary of the Student Union.

I began my professional journey as an intern at Omobio Pvt Ltd, where I was promoted first to Technical Support Officer and then to Operations Engineer. During my time at Omobio, I provided L1 and L2 support for mobile applications, including Dialog Selfcare, Mobitel, Airtel, Hutch, Vodafone, and Txtnation / working with core networking solutions like SMSC/USSD ncell/xl Axiata/celcom/rubi. I also worked and maintained Unix security patches and policies on production systems, specifically focusing on MBSS security patches and VA’s in Axiata products.

Currently, I am an Engineer Managed Service at ADL. In my role, I handle incidents and service requests related to applications like Dialog WOW Super App, Dialog SIRA, and Dialog RPA. I troubleshoot and resolve these issues, support these systems 24/7 in a roster, monitor and troubleshoot application performance, and occasionally implement monitoring solutions for production enterprise applications. I also manage containerized applications using Kubernetes and Docker Swarm, maintain comprehensive documentation (SOPs and MOPs), automate tasks using Bash/ report automation, conduct root cause analysis for critical incidents, and oversee continuous delivery and deployment.

I hold an RHCSA certification from Red Hat. Right now, I’m preparing for the CKA certification and the GCP Associate Cloud Engineer exam, which I aim to complete by this September  
.

If I'm hired, you can count on me to be a dedicated colleague and a positive representative of the team and the company. I believe my hard work and over three years of experience have prepared me well for this role. I’m a fast learner, and I work well with teams, staying calm and focused even under pressure. I’m also someone who takes responsibility seriously—when an issue comes up.   
And that’s pretty much about myself. Thanks again for your time.  
  
**Service Level Agreement,** is a contract between a service provider and a customer that outlines the expected level of service. It defines the specific services provided, the standards to which those services will be delivered, and the responsibilities of both parties

##########Where did you see your self in 5 years ? ###################

in 5 years i see myself becoming better at my job, leading projects in LSEG and working well with teams,

######### What are you passionate about?###############

I’m passionate about technology and finding new ways to solve problems. I also love sports, which helps me stay active and work well in teams.

############# why we want to hire you ######################

I am excited about LSEG’s goal to be a top global financial markets. The company’s focus on innovation, strength, and sustainability matches my values and career goals. LSEG use advanced technology to ensure reliable and efficient financial services. I look forward to contributing my skills to support and enhance LSEG's applications, ensuring they run smoothly and meet the highest standards.  
**######### Why are you leaving your current job? / Why did you leave your previous job?########**

I've really enjoyed my time at my current job and have learned a lot. However, I'm now looking for new challenges and opportunities to grow. I want to expand my skills in cloud computing, automation, and the fintech and banking domains. I believe that a new job will help me achieve my career goals and continue my professional development  
**#######challenging Situation your face and how to handle that at work ###################**

One of the challenging situation I faced was when the same day Axi Bot, responsible for main bank and Dialog payment reconciliation, went down due to container memory and CPU issues. The issue occurred in the morning, but it wasn’t identified until later in the evening.

That day, my senior member and tech lead were both on leave, so I was the only one available to address the problem. So I assign the ticket and find the issue from the logs, I quickly escalated the issue to get the necessary approvals while providing a detailed impact analysis.

Once I received the approval, I restarted the container and collected all the data from ActiveMQ. Then, I re-pushed every transaction through the bot. ticket also resolved withing the SLA time.

After resolving the issue, I Had to create a RCA document, Also, I recognized the need for better monitoring to prevent similar incidents. I decided to create a monitoring solution for the RPA bot, which I successfully completed two months ago.

**professional stories / Automation Monitoring task what have I done**

When I took over managing the Dialog RPA production environment, I quickly noticed something was off—we were dealing with a lot of recurring issues. Every day, it seemed like the same problems kept popping up, and it was causing a lot of headaches for the team. I knew we needed a proper monitoring solution to help us catch these problems before they became bigger issues.

So, I decided to take the lead on it. I started by getting all the scratch done in staging and necessary approvals, which took some time, but once I had the green light, I jumped into the implementation. I made sure our in-house RPA application servers and container-level metrics were fully integrated into the new monitoring system. I wanted to ensure that we could track everything smoothly and get ahead of any potential problems.

After everything was set up,. The number of support tickets dropped significantly, and it became so much easier for us to pinpoint and resolve issues. The best part was that the Head of Practice noticed my efforts and appreciated the work I had put in. That recognition really meant a lot to me and made all the hard work worth it.  
. **How do you prioritize and manage multiple support tickets simultaneously?**  
  
I will prioritize the issues based on urgency and impact, im focusing first on, **high-priority issues that affect the most users or have the biggest business impact**. For medium and low-priority tickets, I group similar ones to resolve them efficiently. Throughout, I keep stakeholders updated, ensuring quick resolutions and clear communication.  
  
**Describe your approach to troubleshooting complex technical issues.**

I like to start by gathering all the info I can—things like error messages, logs, and feedback from users. This helps me get a good idea of what's going on and spot any patterns.

Then, I break the problem down into smaller, manageable pieces and tackle them one by one, starting with the most critical parts. I use debugging tools, check system settings, and sometimes run tests to figure out the root cause.

As I go, I keep track of what I'm finding so I can keep the team in the loop. Once I find the problem, I work on a fix, test it to make sure it works, and then update any documentation. I also share what I learned with the team to avoid the same issue in the future. This step-by-step approach helps me solve problems efficiently and keep everything running smoothly.

**How to troubleshoot network related issue in Linux server. explain with step by step approach.**

|  |  |
| --- | --- |
| Verify Network Configuration | ip addr show : Ensure the server has the correct IP configuration  ip route show : Verify the default gateway is set correctly  cat /etc/resolv.conf : Ensure DNS servers are configured correctly |
| Test Network Connectivity | ping -c 4 192.168.1.1 # Gateway  ping -c 4 8.8.8.8 # Public DNS  ping -c 4 google.com # External site  traceroute google.com : Identify where the connection is failing. |
| Check for DNS Issues | nslookup google.com  dig google.com |
| Inspect Firewall Rules | sudo ufw status : Ensure the firewall is not blocking necessary traffic |
| Analyze Network Interfaces | ip link show  ifconfig  sudo systemctl restart networking : Restart the network service to reinitialize the interface. |
| Review Logs | sudo tail -f /var/log/syslog |
| Test with Alternative Tools | telnet google.com 80  nc -zv google.com 80  sudo tcpdump -i eth0 : Use tcpdump or wireshark: Capture and analyze packets for deeper network troubleshooting |

**<<<<<<<<What you do when P1 happen and same time your manager assign a task to do immediately>>>>>>>>>>>>**

First, I would immediately start working on resolving the P1 issue. I’d quickly gather all necessary information and begin troubleshooting to mitigate the impact as soon as possible.

If my manager assigns me another task at the same time, I would inform them about the P1 issue I’m handling and explain its severity and impact.

Once the P1 issue is under control or resolved, I would then turn my attention to the task my manager assigned, ensuring that it gets done as soon as possible.

**What strategies do you employ to communicate effectively with both technical and non-technical stakeholders?**

It is basically like that, When communicating with technical stakeholders, I focus on using precise terminology and providing detailed information about the issue. This allows us to have a clear understanding of the problem and collaborate effectively towards finding a solution.

On the other hand, when dealing with non-technical stakeholders, my approach is to simplify complex concepts and use analogies or real-life examples that are easy to understand. I make sure to avoid using technical terms and keep my explanations clear.   
  
**How to handle angry client**   
  
I make sure to listen carefully and let them express their concerns without interrupting. I acknowledged their frustration, saying, 'I understand how this is affecting you, and I’m really sorry.'

Next, I reviewed the issue to see what had been done and assured the client that I was taking over to resolve it quickly. I explained the steps I’d take and let them know when they could expect an update.

I kept the client informed throughout, even if there wasn’t much new to report. Eventually, the issue was resolved, and the client was satisfied.

By staying calm, showing empathy, and keeping the client in the loop, I managed to turn a tough situation around.  
  
**Can you discuss your experience with version control systems like Git?**

I have a kind of a knowledge about git actually, our all the Kubernetes configuration for the Dialog SIRA project is in our ADL Git. My work involves cloning the necessary files, creating new branches for changes, and then committing and pushing the configurations. These are the key tasks I’m responsible for when working with Git.  
  
Additionally, We have read/write access to the Git repository, which allows us to use the application’s source code to pinpoint and troubleshoot issues if needed.  
Also, I’m familiar with best practices such as committing small, incremental changes with clear commit messages, which helps maintain a clean and understandable project history.

**What is your experience with cloud-based platforms such as AWS, Azure, or Google Cloud?**

In my current role as an managed service engineer, I gained hands-on experience working with cloud-based platforms, like AWS. I used CloudWatch to monitor various resources like API Gateway, RDS, storage, memory, CPU, and latency, ensuring optimal performance and quickly identifying any issues.

I also have a general idea about other AWS services like EKS, ECR, ECS, IAM, EC2, Step Functions, lambda and S3 buckets. However, I didn’t actually configure or implement these myself because our cloud team handled all the cloud setups and implementations.  
Once I finished my CKA certification, I will start the AWS journey as soon as possible.

**Report automation >> give the sql >> create lamdba function >> choose runtime (python/node.js) >> write the function using RDS db >>schedule it with aws cloud watch event to execute daily or weekly**  
**Why does my database query sometimes take 5 seconds and other times 15 minutes to run?  
  
let me think about it,**

**A1:** One reason could be **caching**.  
The first time, the query runs fast because the data is already in memory, but later, if the cache has been cleared, the database has to fetch the data from disk, which is much slower.

**A2:** Another reason might be **querying plan changes**.

The first time the query runs, the database chooses an efficient plan, like using an index, so it finds the data quickly. But later, if the database updates its statistics, it might decide to do a full table scan instead, which means looking through every row and takes a lot longer.

**A3:** **Database load** can also affect query performance.  
  
The first time, the server might be under low load with plenty of resources available, but later, if it's busy with other queries or transactions, our query might have to wait for resources, causing delays.

**A6:** **Network latency** could be the reason, especially if the database is remote.  
  
The first time, the network is stable and fast, allowing for quick data transfer, but later, network issues can be happen, so it will get longer time to execute the query.

**A7:** **Resource throttling** might also be a factor, especially in cloud environments.

The first time, the query runs smoothly within the available CPU and I/O limits, but later, if the server hits its resource limits or starts throttling, the execution slows down significantly.*Throttling is when a system intentionally limits or reduces the amount of resources it provides to a process or user. This is often done to prevent overuse of resources, ensure fair distribution, or manage load.* **A8:** **Background processes** like backups, indexing, or maintenance tasks can consume resources and slow down queries.

The first time, there are no significant background processes running, so your query has full access to resources, but later, if a background task starts, it competes for those resources and slows down the query.

**Describe how you would identify and resolve a memory leak issue in an application.**Symptoms:

* The application consumes more memory over time without releasing it.
* The system may slow down or become unresponsive as available memory is exhausted.
* We can observe increased swap usage or frequent OutOfMemory errors.

Tools:

* **top or htop:** Monitor memory usage over time and look for processes that steadily consume more memory.
* **free -m or vmstat:** Check overall memory usage, including free and used memory.
* **Application-specific monitoring:** If available, use tools like JVM GC logs for Java applications or memory profiling tools like VisualVM, Valgrind (for C/C++), or memory profilers in IDEs.

**Run the Application in a Controlled Environment:** Try to replicate the issue in a test environment

**What is your understanding of microservices architecture and its benefits?**

Microservices architecture is a design approach where an application is built as a collection of small, independent services that communicate with each other using APIs. Each microservice focuses on a specific functionality and can be developed, deployed, and scaled independently from the rest of the application.  
  
**what is mean by the technical support in your words ? why you choose career in it ?**

"I choose to go into technical support because I’ve been fascinated by technology all my life, and I also like working with people. I want to use my technical knowledge to solve issues customers are having. I also want to apply my knowledge which is related to Unix as well as related to SQL to solve the customer issues.  
 **What is application support?**

Answer: The application support is supporting the delivered application to the customer using different technical skills. The user will require any of the technical skills. The Unix and SQL skills are required to troubleshoot the application-related issues. Mainly to work with technical support, the user needs to know about basics of SQL as well as basics of Unix.

**Tell me some important skills to do technical support job?**

**Technical Knowledge of Front End:** User must know about the technologies used in that specific application, whether it is Java, .NET, or any other technology.

**Technical Knowledge about Back End:** User must know about back-end technologies used in that application.

**SQL Skills:** User must know SQL to troubleshoot the application-level issues.

**Communication Skill:** User must have good communication skills so that they can understand the exact issue.

**Knowledge of Tools:** There are so many tools for issue tracking and management. The application support engineer must have knowledge of different tools.

**Understanding the Basics ITIL**

|  |  |
| --- | --- |
| Problem: | A problem is something that might cause one or more incidents. We usually spot problems when similar issues keep popping up or when we proactively check the IT environment. |
| Incident: | An incident is an unexpected interruption or drop in the quality of an IT service. Our main goal with incident management is to get everything back to normal as quickly as possible and minimize the impact on the business. |
| Service Request (SR): | A service request is when someone formally asks us to provide something, whether it’s information, advice, access to a service, or a standard change. |
| Change Request (CR): | A change request is a proposal to modify something, like a service or documentation. We manage these through a change management process to ensure they’re reviewed, approved, and implemented with minimal risk. |
| Priority 1 (P1): | A P1 is an urgent issue that needs immediate attention, like when a major system goes down and affects many people. |

**WHEN P1 HAPPENS,**

**Act Fast:** Quickly recognize and acknowledge the issue.

**Inform Everyone:**  Then, Let key people know about the problem and its impact. (initial COM is send by the Service manager)

**Check Details:** Gather all relevant information about the issue.

**Fix It:** Try to fix the problem right away or apply a temporary workaround.

**Update Regularly:** Keep everyone informed about progress and updates.(progress reports send by the service manager)

**Solve and Verify:** Once fixed, make sure everything is back to normal.

**Document and Review:** Record what happened and how it was resolved (RCA creating), then review what can be improved.  
  
**Severity Levels and SLA Times**

**P1 - Critical (Priority 1)**

* **Definition**: Major system failure affecting all users, causing a complete work stoppage.
* **SLA Response Time**: 15 minutes
* **SLA Resolution Time**: 2 hours

**P2 - High (Priority 2)**

* **Definition**: Significant system issue affecting a large number of users or critical business functions, but there is a workaround.
* **SLA Response Time**: 30 minutes
* **SLA Resolution Time**: 4 hours

**P3 - Medium (Priority 3)**

* **Definition**: Moderate impact issue affecting a small number of users or non-critical functions. Workarounds are available.
* **SLA Response Time**: 1 hour
* **SLA Resolution Time**: 8 hours

**P4 - Low (Priority 4)**

* **Definition**: Minor issues or general inquiries that do not significantly affect operations.
* **SLA Response Time**: 4 hours
* **SLA Resolution Time**: 24 hours

|  |  |
| --- | --- |
| **Incident Management Process** | |
| Identification and Logging: | We (L1 or L2) find out about incidents through our monitoring tools, user reports, emails,  and then , Every incident is recorded in our IT system with details like when it happened, what went wrong, which services were affected, and user info.(ITSM) |
| Categorization and Prioritization: | We sort incidents based on the issue type and affected service.  We use a priority matrix to decide how urgent and impactful each incident is, so we handle the most critical ones first. |
| Initial Diagnosis and Escalation: | Our service desk team tries to figure out what’s wrong and fix it.  If they can’t resolve it quickly, they pass it to a more specialized team (L1 to L2, then L2 to L3). |
| Investigation and Resolution: | The team responsible digs into the root cause of the incident. Once they find the cause, they implement a fix. |
| Incident Closure and Communication: | We mark the incident as resolved and document all details. We let the user know that the issue is fixed and if they need to do anything else. |

|  |  |  |
| --- | --- | --- |
| **Problem management** | | |
| **Identify the Problem** | Service Desk | * Identify a problem, either through a reported incident or through monitoring and analysis of IT systems. |
| **Log the Problem/ Categorization/ Prioritization** | Problem Manager | * Log the problem in a problem management system, which includes information such as the affected service, the impact of the problem, and any related documentation. * Label the problem with appropriate categorization. * Prioritize the problem. |
| **Diagnose the Problem** | Technical Support | * Determine the root cause of the problem |
| **Fix the Problem** | Problem Manager/ Change Management Team | * Identify a permanent solution to resolve the problem. * Plan and implement the fix through the ***change management process***, ensuring minimal disruption. * Test the solution in a controlled environment before deploying it in production. * Verify the problem is resolved and no new issues arise. |
| **Review and Close** | Problem Manager | * Conduct a post-implementation review to assess the effectiveness of the solution. * Document any lessons learned and update the Knowledge Management System. * Ensure all stakeholders are informed, and then close the problem record. * Analyze the problem data to prevent similar issues in the future. |

|  |  |
| --- | --- |
| **Change Management** | |
| Request Submission: | When someone needs something modified or updated, they submit a change request. We record these requests in our system with all the necessary details.(CR creation) |
| Assessment and Approval: | First, we make sure the request is clear. Then, we evaluate the potential impact and risks.  The Change Advisory Board (CAB) team reviews the request based on its risk level. They can approve it, reject it, or ask for more information. |
| Planning and Scheduling: | Next, we create a detailed step-by-step plan for implementing the change, including what to do if something goes wrong. (rollout plan) We then schedule the change for a time that causes the least disruption, usually during off-peak hours. |
| Communication: | We inform everyone who needs to know about the change, its impact, and timing. This includes any downtimes or service notes if necessary. |
| Implementation: | Execution: We carry out the change according to the plan. |

|  |  |
| --- | --- |
| **Release and Deployment Management** | |
| 1. Planning: | Release Planning: We outline what will be deployed, when, and who will do it.  Risk Assessment: We assess any risks and plan how to manage them. |
| 2. Build and Test: | Development: We build the new service or update.  Testing: We thoroughly test it to ensure it works properly. |
| 3. Preparation: | Configuration Management: We make sure all settings are documented.  Training: We train users and IT staff on the new features. |
| 4. Deployment: | Pilot Deployment: We first deploy the update to a small group to catch any issues early.  Full Deployment: We then roll it out to everyone in stages to manage any problems.  Monitoring: We keep an eye on the deployment to quickly address any issues. |
| 5. Review and Documentation: | Post-Deployment Review: After deployment, we review the process and document what went well and what didn’t.  Documentation: We update all records with the new information. |
| 6. Feedback and Improvement: | User Feedback: We gather feedback from users to identify any remaining issues.  Continuous Improvement: We use this feedback to improve future releases. |

**Escalation** is a way to handle issues (incidents and problems) that cannot be resolved quickly at the first level of support. There are two main types of escalation:

**1. Functional Escalation**

* **What it is**: Passing the issue to a more skilled or specialized team.
* **Example**: If the help desk can't fix your computer problem, they send it to a technical support team.

**2. Hierarchical Escalation**

* **What it is**: Moving the issue up the management chain.
* **Example**: If the problem is serious or urgent, it might be escalated to a manager or higher authority.

**When Escalation Happens**

* **SLA Breach**: The issue isn't resolved within the agreed time.
* **High Impact/Urgency**: The problem is very important or urgent.
* **Lack of Expertise**: The current team can't solve it due to a lack of knowledge or resources.

**What is Ansible?**

**Ansible** is an open-source automation tool used for configuration management, application deployment, and task automation.  
  
**What is an Ansible Playbook?**

An **Ansible Playbook** is a YAML file that contains a series of tasks to be executed on managed hosts. Playbooks are the core of Ansible’s configuration, deployment, and orchestration language. They describe the desired state of the system and how to achieve it.

1. What is DevOps and how does it benefit an organization?

Answer:

DevOps is a set of practices that combine software development (Dev) and IT operations (Ops). It aims to shorten the development lifecycle and deliver high-quality software continuously. The benefits of DevOps include:

Faster Time to Market: Automated deployments and continuous delivery pipelines speed up the release process.

Improved Collaboration: DevOps fosters a culture of collaboration between development and operations teams.

Increased Efficiency: Automation of repetitive tasks reduces manual errors and frees up time for more critical activities.

Enhanced Reliability: Continuous monitoring and testing ensure that issues are identified and resolved quickly, leading to more stable releases.

2. What are some key principles of DevOps?

Answer:

The key principles of DevOps include:

Automation: Automating repetitive tasks to improve efficiency and reduce errors.

Continuous Integration and Continuous Deployment (CI/CD): Integrating code changes frequently and deploying them automatically.

Collaboration: Encouraging communication and collaboration between development, operations, and other stakeholders.

Monitoring and Logging: Continuously monitoring applications and infrastructure to identify and resolve issues proactively.

Infrastructure as Code (IaC): Managing and provisioning infrastructure through code rather than manual processes.

3. What tools are commonly used in a DevOps environment?

Answer:

Common DevOps tools include:

Version Control: Git, SVN

CI/CD: Jenkins, GitLab CI, CircleCI, Travis CI

Configuration Management: Ansible, Puppet, Chef

Containerization: Docker, Kubernetes

Monitoring and Logging: Prometheus, Grafana, ELK Stack (Elasticsearch, Logstash, Kibana)

Cloud Platforms: AWS, Azure, Google Cloud

4. Can you explain the concept of Infrastructure as Code (IaC)?

Answer:

Infrastructure as Code (IaC) is the practice of managing and provisioning computing infrastructure through machine-readable definition files, rather than through physical hardware configuration or interactive configuration tools. IaC allows for consistent and repeatable infrastructure setups. Tools like Terraform, AWS CloudFormation, and Ansible are used to implement IaC, enabling version control, automated testing, and easier rollback

5. How do you ensure the security of your CI/CD pipeline?

Answer:

To ensure the security of a CI/CD pipeline:

Use Secure Code Practices: Regular code reviews, static code analysis, and security testing.

Access Control: Implement strict access controls and permissions for the CI/CD tools.

Secrets Management: Use tools like HashiCorp Vault or AWS Secrets Manager to securely manage credentials and sensitive information.

Regular Updates: Keep the CI/CD tools and dependencies up to date with the latest security patches.

Monitoring and Auditing: Continuously monitor the pipeline and conduct regular audits to detect and respond to potential security incidents.

6. What is continuous integration (CI) and why is it important?

Answer:

the automated building and testing of your application on every new commit, OR

Continuous Integration (CI) is a development practice where developers integrate code into a shared repository frequently, usually multiple times a day. Each integration is automatically verified by an automated build and automated tests to detect integration errors as quickly as possible. CI is important because it helps identify and fix bugs early in the development cycle, improves software quality, and reduces the time to release.

7. Explain the difference between continuous delivery and continuous deployment.

Answer:

Continuous Delivery: In continuous delivery, code changes are automatically built, tested, and prepared for a release to production. However, the deployment to production is a manual step.

Continuous Deployment: In continuous deployment, code changes are automatically built, tested, and deployed to production without any manual intervention.

8. What is Docker and how does it work?

Answer:

Docker is a platform that allows developers to automate the deployment of applications inside lightweight, portable containers. Containers encapsulate an application and its dependencies, ensuring consistency across multiple environments. Docker works by using a client-server architecture where the Docker client talks to the Docker daemon, which does the heavy lifting of building, running, and managing Docker containers.

9. Can you explain Kubernetes and its primary components?

Answer:

Kubernetes is an open-source container orchestration platform that automates the deployment, scaling, and management of containerized applications. Its primary components include:

Pods: The smallest deployable units in Kubernetes, which can contain one or more containers.

Nodes: Worker machines (virtual or physical) where Pods are deployed.

Cluster: A set of Nodes managed by Kubernetes.

Control Plane: The component that manages the Kubernetes cluster, including the API server, scheduler, controller manager, and etcd (the key-value store).

Services: Abstracts to expose applications running on Pods to other Pods or external users.

Namespaces: Virtual clusters within a Kubernetes cluster to provide isolation.

10. How do you handle logging and monitoring in a DevOps environment?

Answer:

In a DevOps environment, logging and monitoring are crucial for maintaining the health and performance of applications and infrastructure. Here are some strategies:

Centralized Logging: Use tools like ELK Stack (Elasticsearch, Logstash, Kibana) or Splunk to aggregate logs from different sources in a central location for easier analysis.

Monitoring Tools: Implement monitoring tools like Prometheus, Grafana, Nagios, or New Relic to track system performance and alert on any anomalies.

Log Management: Use log management practices to ensure logs are rotated, archived, and searchable.

Alerting: Set up alerts to notify the team of any critical issues that require immediate attention.

Dashboards: Create dashboards to visualize key metrics and provide insights into the system's performance.

#### Salary#######

I'm open to discussing what you believe to be a fair salary for the position. Could you please let me know if there is a budget range for this role?

However, based on my previous salary, my knowledge of the industry within last 3 and half year,

and my understanding of this geographic area,

I'd expect a take home salary in the general range of 250K to 300K.

Again, I'm open to discussing these numbers with you.

<<<<<<<<<About LSEG >>>>>>>>>  
  
LSEG plc, or the London Stock Exchange Group, is a big financial company in London. They run stock exchanges and provide financial data and analytics, helping businesses with trading and investment decisions. They're a key player in the global financial market

############ What are your weaknesses?###################

“One of my weaknesses is presenting in front of people phycally. I get nervous and sometimes struggle to convey my ideas clearly. However, I’m actively working on this by practicing more and attending spoken classes to improve my confidence and skills.”

###############what are your strengths ?#####################

as i mentioned before I was actively involved in student leadership as the President of the Sports Council and Secretary of the Student Union as well as worked as volley ball captain. Participate the inter university chess/badminton tournaments

perorally i got the second class on my academics, and i think it proven my strength of managing and working in the pressure on separate areas perorally. that’s I Bellew it is my one of strength

#####Do you have any questions?########

Thank you for giving me this opportunity to interview. I’m very interested in this role and the company. If I am selected, could you please tell me more about the specific responsibilities and expectations for this position?  
  
Could you tell me what the next steps in the hiring process are?