Script

Hi, my name is Kursheeka Milburn and welcome to my PowerPoint presentation on my understanding of a full stack serverless application with full integration. In this PowerPoint, I will be going over how full stack applications migrated over to AWS, how a full stack application was containerized, along with the benefits, challenges, and the end results of the application. Containerization and full stack development has docker for container creation. Which package applications and all its dependencies into a lightweight container. It ensures the app runs the same way on any system. Docker compose (orchestration) is a tool to define and run multi container applications. It makes starting and stopping the entire stack easy with only one command. As for MEAN stack in containers it consist of Mongo DB, which is a database container, Express JS, which is a backend, web framework, NodeJS, which the runtime environment hosting back in logic, and Angular, a front- and single-page app that runs and its own container. Orchestration in full stack development manages multi-container apps to stay available and scale when needed. A full stack often has many parts and to keep all the containers running smoothly; more than one container app is used. We can define how services connect and scale by controlling the network which decides how these containers talk to each other and by controlling scales of services if the traffic increases. We are also able to adjust easily for demand and in this case can scale up when traffic is high or down during low traffic to save resources. What is serverless? Serverless containers are managed servers which allows you to write code and leave the rest up to the cloud to manage. Here, developers focuses on writing the functions. No hardware maintenance is needed to patch, upgrade, or monitor the servers. The cloud provider handles the infrastructure, availability, and security. Automatic scaling allows instances to scale up instantly. When many requests come in at once or scale back when no one is using them. Lastly, you only pay for what

you use at no additional cost when your function isn't being run. For S3 storage versus local storage in full stacks, local storage has fixed size which is limited to the physical disc space that is available along with the hardware dependence being tied to one server which data can be lost in this case. With Amazon S3 storage you have unlimited capacity which you're able to store virtually any amount of data without having to worry about the space as well as being highly redundant where data is automatically copied across many different servers for durability. API Gateway acts as an entry point for different requests that comes from any mobile devices or the web. AWS Lambda runs the serverless back in code in response to API Gateway calls. Being that Lambda is serverless it does not manage any servers and can be connected to databases and other AWS services. With these two services together, it can replace a traditional backend server. Dynamo DB versus Mongo DB. Dynamo DB has key value and documents that are stored which optimize for fast lookups and scalable workloads. It can handle backups and updates without any servers to maintain and adjust capacity based on incoming and outgoing traffic. Mango DB stores data in JSON documents. You're also able to install configure and manage everything yourself in scaling requires manual work, which means that we need to set up different replications to handle growth of traffic. The cloud development principles ensures performance without overpaying for unused capacity by using elasticity which automatically skills up or down doing low usage or a demand increase. In addition, you only pay for the resources you use, which is cost-effective without the need to buy or maintain any expensive hardware. When securing cloud applications, IAM roles and lease privileges are very important. This gives users the minimum permission needed to do their job, which reduces the risks of any account being compromised. We can prevent functions from overreaching into other services with custom Lambda access policies, controlling S3 bucket permissions which should not be public unless it is necessary, and using API keys and

authentication for API gateways, which protects backend endpoints so only authorized individuals can call them. API and database security. When an event from S3 triggers a Lambda function AWS handles the secure connection automatically without exposing any credentials. API Gateway protects the back and functions by requiring an API token or a key which ensures only authorized individuals can trigger Lambda functions . Lambda uses IAM roles with permissions to access dynamo DB to prevent the function being misused. In conclusion, full stacks is a flexible scalable architecture, which make sure the apps can handle small and large traffic loads without any issues. It is efficient with pay for use where you're only paying for what you use and nothing more. It has strong security by design with manage services that comes with built-in security and different policies that are put in place for minimal access. Thank you