AWS Cloud Specialist

Project 1: Deploying a Scalable Web Application on AWS

- **Objective**: Complete Cloud Web Application Builder Training Course
- **Description**: AWS services to design and deploy a database-backed web application in the AWS Cloud without step-by-step guidance. The architecture must reflect the principles of the AWS Well-Architected Framework and be highly available, scalable, high performing, and secure.
- **Technologies to use**: Cloud Web Application Builder Training content.

Week 1: Project Planning and Architectural Design & Create a basic functional web application

Tasks:

- Creating an architectural diagram: Create an architectural diagram to illustrate what you plan to build. Consider how you will accomplish each requirement in the solution. .
- Developing a cost estimate: Develop a cost estimate that shows the cost to run the solution in the us-east-1 Region for 12 months.
- Creating a virtual network: Create a virtual network to host the web application using Amazon VPC.
- Creating a virtual machine: Create a virtual machine in the cloud to host the web application using AWS EC2.
- Testing the deployment: Test the deployment of the web application to ensure it is accessible from the internet and functional. Perform a few tasks, such as viewing, adding, deleting, or modifying records.

Deliverables:

- Create an architectural diagram to illustrate what you plan to build. Consider.
- Initial cost estimation using the AWS Pricing Calculator.
- Deployed web application running on an EC2 instance with a relational database (RDS).
- A virtual network (VPC) configured for security and public accessibility.
- Documentation outlining the separation of application layers (web server and database).

Week 2: Decoupling the application components
Tasks:

- Changing the VPC configuration: Update the virtual network components that are necessary to support hosting the database separately from the application
- Creating and configuring the Amazon RDS database: Create an Amazon Relational Database Service (Amazon RDS) database that runs a MySQL engine. You can choose to create a provisioned instance or run it serverlessly.
- Configuring the development environment: Provision an AWS Cloud9 environment to run AWS Command Line Interface (AWS CLI) commands in later tasks.
- Provisioning Secrets Manager: Use AWS Secrets Manager to create a secret to store the database credentials, and configure the web application to use Secrets Manager
- Provisioning a new instance for the web server: Create a new virtual machine to host the web application
- Migrating the database: Migrate the data from the original database, which is on an EC2 instance, to the new Amazon RDS database.
- Testing the application: Access the application and perform a few tasks to test it. For example, view, add, delete, and modify student records.

Deliverables:

- Separate the database and the web server infrastructure so that they run independently.
- Host the web application should run on a separate virtual machine.
- Host database should on a managed service infrastructure

Week 3: Implementing high availability and scalability
Tasks:

- Creating an Application Load Balancer: Launch a load balancer. The endpoint will be used to access the web application.
- Implementing Amazon EC2 Auto Scaling: Create a new launch template, and use an Auto Scaling group to launch the EC2 instances that host the web application.
- Accessing the application: Access the application and perform a few tasks to test it
- Load testing the application: Perform a load test on the application to monitor scaling.

Deliverables:

- Complete the design and fulfill the remaining solution requirements.
- Provide a fully scalable and highly available architecture for the web application.

AWS Cloud Specialist

Project 2: Securing and Monitoring Resources with AWS

- **Objective**: Complete AWS Academy Lab Project - Cloud Security Builder Training Course
- **Description**: use familiar AWS services, as well as AWS services that might be new to you, to create resources in AWS and to implement security on them. Throughout various AWS Academy courses, you have completed hands-on labs. You have used different AWS services and features to build a variety of solutions.

Week 1: Securing data in Amazon S3
Tasks:

- Create a bucket, apply a bucket policy, and test access: create a bucket, apply a bucket policy to it, and then test it's access.
- Enable versioning and object-level logging on a bucket: enable versioning and object-level logging on the *data-bucket*. With versioning enabled so all changes to objects that are stored in the bucket can be tracked and any object reverted to a previous version if needed. Object-level logging creates a detailed audit trail of the objects that are stored in a bucket, which helps in detecting security incidents quickly.
- Implement the S3 Inventory feature on a bucket: enable the S3 Inventory feature to monitor changes to objects that are stored in an S3 bucket. S3 Inventory provides a scheduled report of the metadata and object-level changes to your S3 objects and buckets. By using the feature, changes to the stored objects can be tracked and detect potential security incidents.
- Confirm that versioning works as intended: access the AWS account as the paulo user and upload an object to the data-bucket. Then, confirm that versioning is enabled on the object. also test access as the mary user.
- Confirm object-level logging and query the access logs by using Athena: confirm the S3 object-level logging you enabled earlier is successfully writing log data to S3. You will also use Athena to query these logs

Deliverables:

- Limit access to buckets to certain account managers, who are in the Account Manager group.

- Enable versioning on S3 buckets and all objects in them.
- Enable object logging in all S3 buckets.
- Encrypt all buckets by using server-side encryption with Amazon S3 managed keys (SSE-S3).
- Implement Amazon S3 Inventory to keep a running inventory of all files that are stored in Amazon S3.

Week 2: Securing VPCs
Tasks:

- Review LabVPC and its associated resources: familiarize with resources that already exist in the lab environment.
- Create a VPC flow log: create a VPC flow log for LabVPC. The VPC Flow Logs feature can help understanding how inbound and outbound traffic flows through the VPC. The feature also provides monitoring information that will provide insights for how to secure the web server, subnets, and VPC in later tasks.
- Access the WebServer instance from the internet and review VPC flow logs in CloudWatch: use a web browser to test access to the WebServer EC2 instance over port 80 (HTTP). And test access to port 22 (SSH) by using the netcat command, which will test whether inbound traffic is allowed. Then, review the VPC flow log to see how these attempts were recorded.
- Configure route table and security group settings: create a route for traffic from the internet to access the WebServerSubnet through an internet gateway. This will allow inbound HTTP traffic to be directed to the WebServer instance. You will also be challenged to modify the security group that is associated with the WebServer instance to allow inbound traffic on ports 22 (SSH) and 80 (HTTP)
- Secure the WebServerSubnet with a network ACL:configure a network access control list (ACL) to secure the subnet where the web server is running. The network ACL will provide an additional layer of security beyond the security group that you already configured.
- Review NetworkFirewallVPC and its associated resources: secure the VPC by using AWS Network Firewall.
- Create a network firewall: create a network firewall for the NetworkFirewallVPC,that haven’t yet used in this project.

- Create route tables: create and configure three new route tables, including one for each subnet in the NetworkFirewallVPC and one to handle inbound (ingress) traffic for the internet gateway in NetworkFirewallVPC.
- Configure logging for the network firewall: configure logging for the network firewall so that you can analyze details of network traffic requests.
- Configure the firewall policy and test access: define and add a stateful rule group to the network firewall's policy. With this policy, traffic to and from the internet and the NetworkFirewallVPC will be monitored and managed. Access over specific ports will be allowed, and access over other ports will be specifically denied.

Deliverables:

- Audit VPC Configuration: Review and document the current VPC settings, including subnets, route tables, and NAT gateways. Identify any misconfigurations, such as insecure routing or missing security layers.
- Implement Security Groups & NACLs: Correct and configure security groups and network access control lists (NACLs) to control inbound and outbound traffic for the web servers, ensuring only authorized access.
- Setup VPC Flow Logs: Enable VPC flow logs to capture network traffic data for monitoring and troubleshooting purposes. This ensures visibility into network activity and helps detect potential security threats.
- Configure Private Subnets & NAT: Adjust the network architecture to move backend resources (e.g., databases) into private subnets and configure NAT gateways for secure internet access without exposing internal systems.
- Update Route Tables & Internet Gateways: Review and update route tables to ensure proper routing of traffic to/from the internet gateway while maintaining tight security controls over network traffic.

Week 3: Securing AWS resources by using AWS KMS

Tasks:

- Create a customer managed key and configure key rotation: create an AWS KMS customer managed key. Then configure automatic key rotation on the key.
- Update the AWS KMS key policy and analyze an IAM policy: modify the policy of the AWS KMS key created so that the sofia user will be authorized to use the key. Alsoanalyze the IAM policy that controls what the sofia user can do in the AWS account.

- Use AWS KMS to encrypt data in Amazon S3: use the AWS KMS key created to encrypt an object in the data-bucket S3 bucket. Then test access to the object.
- Use AWS KMS to encrypt the root volume of an EC2 instance: Use the AWS KMS key again, but now use it to encrypt the root volume of a new EC2 instance.
- Use AWS KMS envelope encryption to encrypt data in place: use the AWS Command Line Interface (AWS CLI) to encrypt data in place by using the AWS KMS key. Also see how to decrypt the encrypted data. For convenience, Use the WebServer2 to complete this task.
- Use AWS KMS to encrypt a Secrets Manager secret:create a key-value pair (a secret), which you will encrypt with AWS KMS key and store in Secrets Manager. Then verify that you can retrieve the secret by using the AWS CLI.

Deliverables:

- Create and Manage AWS KMS Customer Managed Keys: Set up AWS KMS customer managed keys (CMKs) to handle the encryption of sensitive data, including PII, credit card numbers, and social security numbers, and configure key rotation for automatic updates.
- Encrypt Amazon S3 Data: Update the S3 bucket configurations to use Server-Side Encryption with AWS KMS (SSE-KMS) for encrypting data in S3, ensuring secure storage of sensitive information.
- Encrypt EC2 Root Volume: Modify EC2 instance configurations to ensure that the root EBS volumes are encrypted using the customer managed key (CMK), providing enhanced data protection for the virtual machines.
- Encrypt Secrets in AWS Secrets Manager: Use the customer managed keys to encrypt secrets stored in AWS Secrets Manager, such as sensitive credentials, ensuring secure access to critical application data.
- Policy Updates and Role Management: Analyze and modify key policies and IAM roles to ensure proper access control, granting and restricting permissions for specific users like "sofia" and other IAM roles based on security needs.

Week 4: Monitoring and logging

Tasks:

- Use CloudTrail to record Amazon S3 API calls: use CloudTrail to record API calls that are made to Amazon S3 buckets. This information will provide an audit trail to track when S3 objects are created, modified, or read.
- Use CloudWatch Logs to monitor secure logs: create a solution to monitor access to EncryptedInstance. In this task, you will configure CloudWatch Logs to monitor SSH access to the instance so that the company can

understand who accesses the server, where they access it from, when they access it, and what actions they take.
- Create a CloudWatch alarm to send notifications for security incidents: create an alarm to notify security team members in IT when such an incident occurs.
- Configure AWS Config to assess security settings and remediate the configuration of AWS resources: use AWS Config to report on whether object logging is configured on the S3 buckets in the AnyCompany Financial account. Also configure an automation script to remediate noncompliance.

Deliverables:

- Set Up AWS Config Rules: Implement AWS Config rules to monitor S3 buckets for object logging configuration. This will ensure continuous tracking and reporting of compliance with company policies.
- Generate Compliance Reports: Use AWS Config to generate compliance reports that show whether S3 buckets have logging enabled, identifying any noncompliant buckets in the AnyCompany Financial account.
- Automate Remediation Script: Create and configure an automation script using AWS Lambda or AWS Systems Manager to automatically enable object logging on S3 buckets that are found to be noncompliant.
- Notification and Alerts: Set up notifications via Amazon SNS to alert the IT or security team when an S3 bucket is found to be noncompliant with the logging configuration, ensuring prompt action.
- Test and Validate Remediation: Run tests on the automation script to ensure that it successfully remediates noncompliant S3 buckets by enabling logging and that the changes are reflected in AWS Config's compliance reports.