CSCI 5902 Advanced Cloud Architecting

GIRI SHARAN REDDY PUSULURU – B00913674

Term Assignment – December 04, 2023

Project Information:

Banking Web Application project developed by using Spring Boot and Angular

Repository link: https://github.com/hendisantika/Online-banking-angular-springboot-mysql.git

Used Technologies:

- Back-end: Spring Boot/Spring Data/Spring Security/Hibernate/MySQL
- Front-end: Angular2
- Security: JWT
- REST APIs
- Server Build: Maven

Reason to select this project:

- It utilizes a modern and popular tech stack (Spring Boot, Angular2), making me easy to understand it's architecture and work flow.
- It showcases various features and functionalities expected in a real-world banking application platform consisting of user and admin register/login, deposit/withdraw money from accounts, add/edit recipients, transfer money between accounts and recipients, view transactions, make appointments. This allows for designing a comprehensive and multifaceted cloud architecture.
- Banking application inherently require scalability, high availability and advanced security features, making it an ideal candidate for exploring elastic and cost-effective cloud services.

AWS Services used:

Compute: EC2

Database: RDS (MySQL)

Storage: S3

Network: VPC and Subnets

Security: IAM, security groups

Monitoring: Cloud Watch

Other: Load balancer, Auto scaling groups, NAT gateway, Internet gateway, Cloud Front, Route 53, AWS Shield Advanced, AWS Key Management Service

Amazon EC2 Instances with Auto Scaling Group offer several advantages:

Reliability: The distribution of EC2 instances across multiple Availability Zones enhances reliability and fault tolerance. If one instance fails, the Auto Scaling Group promptly replaces it, ensuring the continuous reliability of the application.

Scalability: Auto Scaling facilitates automatic horizontal or vertical scaling of EC2 instances based on demand. This feature is valuable for handling varying traffic levels, scaling up during peaks, and scaling down during periods of reduced activity.

Cost Efficiency: Auto Scaling is particularly beneficial for applications with fluctuating workloads like such as those in the banking sector. It optimizes resource usage, scaling up when necessary to meet demand and scaling down during less busy periods, thus promoting cost efficiency.

Performance Optimization: Dynamic allocation of resources by Auto Scaling responds to current loads, optimizing performance during high-traffic periods and preventing unnecessary resource allocation during low-traffic periods. This ensures a responsive application even under increased load, delivering a seamless user experience.

Ease of Management: Auto Scaling streamlines deployment and management tasks by automating the launch and termination of instances based on demand. This minimizes manual intervention, promoting efficient resource utilization.

While AWS Elastic Beanstalk is a convenient and user-friendly platform for deploying and managing web applications, the reason why I did not choose this is

a. Bank applications often need to adhere to strict regulatory compliance standards. Directly managing EC2 instances allows for fine-grained control over security configurations, ensuring compliance with industry-specific regulations.
b. For resource-intensive bank applications like ours, manually managing EC2 instances allows for precise tuning of performance parameters, ensuring that the infrastructure meets the application's unique performance requirements.

I have placed my bank application servers running in EC2 instance in a private subnet, away from internet access makes it much more difficult for hackers to attack the servers. EC2 offers a variety of instance types that are optimized for high-performance computing workloads like hpc6a.48xlarge. This is important for bank applications that require a lot of processing power, such as those that use machine learning or real-time analytics. EC2 can be used to deploy bank applications in a hybrid or multi-cloud environment. This allows banks to leverage the best of both on-premises and cloud infrastructure. EC2 provides a range of operating systems and security features allowing to isolate the instance from internet traffic.

Amazon RDS:

Amazon RDS with MySQL serves as a comprehensive managed database service, particularly beneficial for a Banking application. Manual database management in such a scenario would demand considerable

expertise and effort, risking data loss. However, RDS automates essential administrative tasks like database setup, patch management, and backups. This automation allows developers to concentrate on application development, streamlining the process and minimizing manual interventions.

- In terms of reliability and high availability, RDS excels by offering Multi-AZ deployments. This feature automatically replicates the database across multiple availability zones, ensuring both high availability and fault tolerance. Such reliability is paramount for a Banking application, where downtime must be minimized to provide users with a seamless experience.
- The scalability aspect of Amazon RDS is crucial for Banking platforms that encounter fluctuating workloads. Traditional database scaling can be intricate, but RDS simplifies this with options for both vertical scaling (upgrading to a larger instance) and horizontal scaling (adding read replicas). This flexibility allows the system to adapt to varying demands with minimal downtime, optimizing performance.
- Furthermore, Amazon RDS seamlessly integrates with popular technologies such as Spring Boot and Hibernate, which are widely utilized in Java-based applications for database connectivity and management. This compatibility ensures a smooth and efficient database solution for the Spring Boot backend of the Banking application.

In addition to these features, read/write replicas play a significant role in enhancing the performance and resilience of the database. RDS allows the creation of read replicas, enabling the distribution of read traffic across multiple database instances. This not only improves read performance but also provides a level of fault tolerance. Additionally, read replicas can be utilized to offload read-intensive workloads from the primary database, contributing to overall system efficiency and responsiveness. The ability to scale horizontally by adding read replicas aligns with the dynamic requirements often encountered in the Banking sector.

Amazon RDS creates a storage volume snapshot of our DB cluster, backing up the entire DB cluster and not just individual databases. Because the snapshot includes the entire storage volume, the size of files, such as temporary files, also affects the amount of time it takes to create the snapshot. Unlike automated backups, manual snapshots aren't subject to the backup retention period. Snapshots don't expire. For very long-term backups, exporting snapshot data to Amazon S3 is recommended. Moreover, we can make use of event bridges to trigger a serverless lambda function to take snapshots at regular intervals and once a snapshot is created or if any event fails, we can use SQS and SNS topics to get alerts for ourselves so that we can ensure data is not affected.

Using Identity Access Management (IAM), we can control who can be authenticated (signed in) and authorized (have permissions) to use Amazon RDS resources. An IAM user is an identity within AWS account that has specific permissions for a single person or application. Give users and roles only the essential permissions by adhering to the principle of least privilege.

By considering all the above important points made me to choose Amazon RDS.

Amazon S3:

S3 offers a powerful and versatile solution for addressing the challenges of high network requests, security, and backend server load in bank applications. Its robust infrastructure, comprehensive security

features, and ability to offload static content make it an essential component for building scalable, secure, and performant bank applications.

- S3 can be used to store documents, such as loan applications, financial statements, and tax returns. This data can be used for a variety of purposes, such as onboarding new customers, servicing loans, and complying with regulations.
- Whenever those documents are required application servers can access directly access documents from S3 by just storing the URL of the specific object, relieving backend servers to focus on dynamic content and business logic.
- S3 is designed to handle high volumes of traffic and can efficiently serve millions of requests per second. This makes it an ideal choice for storing and serving content for bank applications that experience high traffic spikes.
- S3's architecture is distributed across multiple data centers, ensuring that requests are routed to the nearest available data center for optimal performance. Additionally, S3 employs edge caching, which stores frequently accessed content closer to users, further reducing latency and improving response times.
- Integration with CDNs for Optimal Performance: Easily integrates with Content Delivery Networks (CDNs) to optimize content delivery and enhance global performance. Ensures efficient distribution of content, improving user experience and application performance.
- We can also make use of network ACLs (Access Control Lists), which allow us to control who can access your data based on their IP address or subnet. This provides an additional layer of security by restricting access to specific networks or subnets.

File Encryption: S3 offers different options for encrypting files stored in S3 buckets but I have used:

a. Object encryption using KMS: Amazon S3 integrates with AWS Key Management Service (AWS KMS) to provide server-side encryption of Amazon S3 objects. The encryption keys that protect your objects never leave AWS KMS unencrypted. This integration also enables you to set permissions on the AWS KMS key and audit the operations that generate, encrypt, and decrypt the data keys that protect your secrets.

Data Confidentiality: Encrypting our RDS snapshots using AWS KMS adds an additional layer of security. It ensures that even if unauthorized access occurs, the data remains confidential and unreadable without the appropriate decryption keys.

KMS integrates seamlessly with other AWS services, ensuring that encryption and decryption processes are transparent to our application. This simplifies the implementation of a robust security framework.

KMS provides detailed logging and audit trails for key usage. This feature helps in monitoring and reviewing key access, ensuring compliance with security policies and regulations.

But for storing the media related to users like profile photo, documents I have used default, standard encryption provided Amazon.

Server-side encryption with S3ManagedEncryption: This is the default encryption option for S3. With S3ManagedEncryption, AWS manages the encryption keys and handles the encryption and decryption process for us.

By enabling Amazon S3 versioning to maintain historical versions of objects, enabling easy recovery in case of accidental deletions or data corruption. Moreover, usage of single backup policy AWS Backup automatically organizes backups across different AWS services and third-party applications in one centralized, encrypted location known as a backup vault. Implementing cross-region replication in Amazon S3 to replicate critical document content to a secondary AWS region. This ensures data backup, allowing to recover data even if the primary region faces an outage. Lifecycle policies include transition objects to cheaper storage classes like Standard-IA, Deep Glacier or delete data that is no longer needed.

--- Taking snapshots of Amazon RDS databases and storing them in an S3 bucket is a crucial practice for data protection and disaster recovery in bank applications. It ensures that we have a backup of critical data in case of any hardware failures, software malfunctions, or accidental data deletion.

Data Protection: Snapshots provide a point-in-time copy of your database, allowing us to restore data to a specific state if needed. This is essential for safeguarding sensitive financial data and customer information in bank applications.

Disaster Recovery: Snapshots enable quick disaster recovery by restoring your database to a functional state after an outage or disaster. This minimizes downtime and ensures business continuity, which is paramount for bank operations.

Version Control: Snapshots act as version control for your database, allowing you to track changes and revert to previous versions if necessary. This is particularly useful for troubleshooting database issues or recovering from accidental data modifications.

Testing and Development: Snapshots can be used for testing and development purposes without affecting your production database. This allows our developers to safely test new features or configuration changes without disrupting live operations.

The frequency of taking snapshots depends on the criticality of our database and the acceptable level of data loss. In addition to regular snapshots, we can consider taking manual snapshots before making significant changes to your database, such as schema modifications or software updates. This provides an extra layer of protection in case of unexpected issues.

In summary, using S3 for different file storage and as a CDN with CloudFront offers a scalable, cost-effective, and performant solution for managing and delivering media assets in my bank application.

VPC (Virtual Private Cloud):

My Banking application, with its sensitive user credentials, financial transaction details, and other digital assets, demands a highly secure and isolated environment. By creating a private network within AWS, VPC restricts access to authorized resources and helps to prevent unauthorized access to the application. Additionally, VPC can be used to implement security controls such as network access control lists (ACLs) and security groups to further enhance security.

- Isolation and Security: The Banking application deals with sensitive by utilizing VPC, the application can create a logically isolated section within AWS, enhancing security by allowing the developer to define its IP address range, configure subnets, and control inbound and outbound traffic through security groups and network access control lists (ACLs).

- Scalability: VPC can be used to scale the banking application to meet the demands of a growing customer base. By creating additional subnets and routing tables, VPC can accommodate new instances and services without disrupting the existing infrastructure. Additionally, VPC can be used to integrate with AWS services such as Amazon Elastic Scale and Amazon CloudWatch to automate scaling and monitoring tasks.
- Cost-Effectiveness: VPC can be used to optimize the cost of the banking application by providing a more efficient and flexible network environment. VPC eliminates the need for on-premises network hardware and software, which can save money on maintenance and support costs. Additionally, VPC can be used to integrate with AWS services such as Amazon CloudWatch and Amazon Trusted Advisor to optimize network performance and costs.
- Network Configuration between resources: Dividing VPC into different subnets (private, public) allows me to organize and categorize my resources logically based on the functionality and its usage. Where the resources in public subnet face the internet while in private subnet are not allowed.
- High Availability: Distributing bank application across multiple availability zones (AZs) within a VPC makes it fault tolerance and high availability. If one Availability Zones goes down due to a unexpected reason or experiences issues, my application can continue running in other AZs without any disruption and data loss.

Subnet:

Subdividing the VPC into Public, Private, and Database subnets is a crucial architectural decision for the Banking application. Each subnet serves a specific purpose, contributing to the overall security, scalability, and performance of the application.

Public Subnet:

- Purpose: Hosting the Nate Gateway. Facilitating accessibility from the internet for the resources in private subnet to interact with outside world. The architecture uses a NAT gateway to provide internet access to the backend application in the private subnet. The NAT gateway translates the private IP addresses of the backend servers to a public IP address so that they can communicate with the internet.

Private Subnet:

- Hosting the bank application servers. To safeguard sensitive data and application logic, the banking application's servers are housed within a private subnet, shielded from direct internet access. This separation effectively restricts unauthorized intrusion, ensuring that only authorized resources can interact with these critical components.
- Advantage: Isolates the bank application servers from direct exposure to the internet, reducing the attack surface and enhancing security. Maintains a clean separation between the public-facing components and the internal application logic.

Database Subnet:

- To safeguard the banking application's crucial database, the Amazon RDS instance is hosted within a dedicated subnet. This isolation strategy creates a secure enclave, effectively severing direct access from both the public internet and the application servers. By enforcing this

separation, we implement the principle of least privilege, restricting access to the database only to authorized resources. This layered defense mechanism bolsters the application's security posture, ensuring that sensitive data remains protected from unauthorized access.

Performance: By placing components into different subnet groups, the architecture optimizes performance by preventing unnecessary exposure of sensitive components to the internet. This ensures that resources can focus on their specific tasks without being burdened by unnecessary traffic.

Security: The subnet configuration contributes significantly to the security of the Banking application. It follows the principle of least privilege by carefully controlling access to different components based on their sensitivity and the required level of exposure.

Scalability: The use of separate subnets allows for independent scaling of application and database components. This ensures that resources can be scaled based on their specific demands without impacting the others.

summary, using VPC and subnet groups for my application isolation provides a secure, scalable, and well-organized network infrastructure as a whole. It allows our administrator to control access, enhance availability, and align with best practices for managing cloud-based services.

Identity Access Management (IAM):

To safeguard the bank's sensitive data and infrastructure, the principle of least privilege must be strictly adhered to when managing and accessing AWS resources. Where we need to create IAM roles by following the above principle in such a way that only authenticated, authorized users only should be able to use ec2, s3 buckets, rds. By enforcing Multi Factor Authentication (MFA) for IAM users who have access to critical AWS resources adds an extra layer of security to prevent unauthorized access. Developing IAM policies that define access specificity to AWS resources and actions they can perform. By regularly reviewing and updating these policies to align with security best practices and evolving requirements.

AWS KMS is used to encrypt data at rest and in transit. Data at rest is encrypted before it is stored in the Amazon S3 buckets. Data in transit is encrypted before it is transmitted between servers.

S3 bucket encryption: The S3 buckets that store application logs and other sensitive data are encrypted using AWS KMS server-side encryption (SSE) at rest. This ensures that the data is encrypted even if unauthorized users gain access to the underlying storage.

AWS KMS provides a centralized location to manage all of the cryptographic keys that are used to encrypt data in the banking application. This makes it easier to control who has access to the keys and to protect the keys from unauthorized access.

Route 53 is a highly scalable and reliable DNS service that is fully managed by Amazon Web Services (AWS). It provides a cost-effective way to route traffic to our services. Route 53 integrates with AWS Identity and Access Management (IAM) for access control. This ensures that only authorized entities can make changes to DNS settings, contributing to a strong identity foundation and adhering to the security best practice of least privilege access. Route 53 can handle millions of queries per second, making it ideal for high-traffic websites and applications. Route 53 is integrated with a number of AWS services,

such as Amazon CloudFront, Amazon Elastic Load Balancing, and Amazon S3. Route 53 is highly reliable, with a 99.999% uptime guarantee.

Elastic Load Balancer (ELB) – Application Load Balancer:

Elastic Load Balancer (ELB) is a service that automatically distributes incoming traffic across multiple instances of an application. This ensures high availability, scalability, and fault tolerance. In a bank application, it is crucial to have high availability to avoid downtime and maintain a seamless experience for users. ELB also enables scalability by allowing for the addition of more instances as demand increases. Additionally, ELB enhances fault tolerance by automatically redirecting traffic away from unhealthy instances.

- High Availability: Bank applications must be highly available to ensure that users can always access their accounts and make transactions. ELB can help to achieve high availability by distributing traffic across multiple instances of the application. This means that if one instance goes down, ELB will automatically route traffic to another instance. This will help to prevent downtime and ensure that users can always access the application.
- Scalability: Bank applications can experience spikes in traffic, such as during release of government aids or weekends. ELB can help to accommodate these spikes in traffic by automatically adding more instances to the application. This will help to ensure that the application can handle the increased load and that users continue to have a good experience.
- Fault Tolerance Bank applications must be fault-tolerant to ensure that they can continue to operate even if there is a problem with one of the instances. ELB can help to achieve fault tolerance by automatically detecting unhealthy instances and redirecting traffic away from them. This will help to prevent the application from going down if one of the instances fails.

Security Groups:

Security groups are stateful. Security Groups are a fundamental security feature in AWS that allows for granular control over network traffic at the instance level. This is crucial in bank applications, where security is paramount due to the sensitivity of user data, financial transaction data, and authentication mechanisms. By following the principle of least privilege, Security Groups permit only the necessary traffic to instances, minimizing the attack surface and potential for unauthorized access. Moreover, Security Groups contribute to a defense-in-depth strategy, adding an extra layer of protection beyond other security measures like network ACLs or application-level security measures.

- Granular Access Control: In the Banking application, security is paramount, especially considering the sensitivity of user data, financial transaction data, authentication mechanisms. Security Groups allow for granular control over inbound and outbound traffic at the instance level.
- Principle of Least Privilege: Following the principle of least privilege is a fundamental security best practice. Security Groups enable the implementation of this principle by allowing only necessary traffic based on specific requirements.
- Improved Fault Tolerance: Security Groups can be configured to isolate instances that exhibit suspicious behavior or have been compromised, preventing the spread of malware or unauthorized access to other instances.

- Cost Optimization: Well-defined Security Groups can optimize cloud resource utilization by preventing unnecessary network traffic, reducing bandwidth consumption and associated costs.

We have to make sure to regularly review and update Security Groups to ensure they align with current security requirements and application needs. Remove outdated rules and ensure only authorized traffic is allowed. Moreover, administrator need to Monitor Security Group activity to detect anomalies or suspicious traffic patterns, indicating potential security breaches or misconfigurations.

Web Application Firewall:

Web Application Firewall (WAF) is a crucial security tool that protects web applications from a wide range of vulnerabilities and exploits. In the context of banking applications, where sensitive user data and financial transactions are involved, WAF plays a pivotal role in safeguarding against common web application threats. WAF provides protection against common web application threats, including SQL injection, cross-site scripting (XSS), and cross-site request forgery (CSRF). Given the sensitive nature of user data and transactions in a Banking application, shielding against these threats is paramount.

AWS WAF is a managed service, which means that AWS takes care of all the provisioning, configuration, and maintenance of the WAF. This makes it easy for you to get started with WAF and to keep it up to date.

By implementing WAF, organizations can demonstrate their commitment to data protection and meet compliance standards. WAF adds an additional layer of security to the overall architecture, contributing to a defense-in-depth strategy that protects against a wide range of threats.

Main reason to consider AWS WAF is it integrates with a variety of other AWS services, such as Amazon CloudFront and Amazon CloudWatch, which makes it easy to manage and monitor our web application security and is available in a number of AWS regions around the world, so we can protect our web applications from anywhere in the world.

Security:

AWS WAF is a highly secure service that is backed by AWS's global infrastructure. AWS WAF uses a variety of security features to protect our web applications, including:

a. Web ACLs: Web ACLs are rules that define what traffic is allowed or blocked to your web applications.
b. Managed rules: Managed rules are pre-configured rules that protect against common web application vulnerabilities, such as SQL injection and cross-site scripting (XSS).
c. IP sets: IP sets are lists of IP addresses that are allowed or blocked from accessing your web applications.

AWS Shield Advanced

AWS Shield Advanced offers a comprehensive suite of defense mechanisms designed to safeguard bank applications from a wide range of cyberattacks, including distributed denial-of-service (DDoS) attacks, application-layer attacks, and network attacks.

Security: Focuses on DDoS protection, ensuring the security of applications against various DDoS attack vectors. It includes features such as traffic inspection, machine learning for attack detection, and integration with a Web Application Firewall (WAF) for additional security against application-layer attacks.

Shield Advanced provides automated protection against application layer (L7) DDoS attacks, eliminating the need for manual intervention. This real-time defense quickly identifies and mitigates attacks targeting specific applications, ensuring continuous service availability.

Reliability: Enhances the reliability of applications by providing DDoS protection. The service helps in maintaining the high availability of applications even during DDoS attacks, ensuring a reliable and consistent user experience.

Cost Optimization: Offers cost optimization by providing a managed DDoS protection solution. Instead of organizations investing in their own infrastructure and resources to mitigate DDoS attacks, AWS Shield Advanced provides a cost-effective and scalable solution with pay-as-you-go pricing.

During DDoS attacks, the infrastructure resources required to handle increased traffic can lead to significant cost spikes. Shield Advanced provides cost protection by automatically scaling cloud resources during attacks, ensuring predictable costs and preventing unexpected expenses.

Performance Efficiency: Contributes to performance efficiency by efficiently mitigating DDoS attacks, preventing service degradation or downtime which is crucial for bank applications. The service is designed to handle large-scale, volumetric DDoS attacks while minimizing the impact on application performance.

Shield Advanced protects payment gateways, preventing disruptions to online transactions and safeguarding sensitive financial data.

AWS CloudFront:

In today's digital landscape, banks are embracing online platforms to deliver their services to customers, expanding their reach and enhancing customer experience. As online presence grows, the need for a robust and scalable content delivery network (CDN) becomes paramount. AWS CloudFront emerges as a powerful CDN solution, specifically tailored for bank applications, ensuring fast, reliable, and secure delivery of content, including images, videos, and web pages.

- CloudFront's global network of edge locations ensures that content is delivered from the nearest edge location to users, minimizing latency and improving delivery speeds. This is particularly important for bank applications like ours, where users worldwide need to access financial information and services quickly and reliably.
- CloudFront is designed for high availability, with redundant edge locations and multiple paths to deliver content. This ensures that content remains accessible even if individual edge locations or network paths experience disruptions. This reliability is crucial for bank applications, where downtime can have significant financial and reputational consequences.

Security: CloudFront incorporates multiple security features to protect sensitive data and prevent unauthorized access. These features include:

a. SSL/TLS encryption: CloudFront encrypts all data in transit between edge locations and users, safeguarding sensitive information such as financial data and customer credentials.
b. Origin Access Control (OAC): OAC restricts access to your origin servers, preventing unauthorized users from directly accessing your content.
c. Signed URLs: Signed URLs provide a secure way to grant temporary access to specific content, ensuring that only authorized users can access sensitive data.

CloudFront efficiently delivers static content, such as images, documents and CSS/JavaScript files, reducing the load on our application servers and improving overall application performance.

Cost Optimization: Supports cost optimization by offering a pay-as-you-go pricing model. It minimizes data transfer costs by caching and delivering content from edge locations closer to end-users, reducing the load on origin servers and optimizing overall content delivery costs. These features of Cloud front made to consider using this service.

Amazon CloudWatch is a monitoring and observability service that provides comprehensive insights into the health, performance, and utilization of AWS resources in our bank application. It collects and analyzes various metrics, logs, and events to provide a holistic view of our bank application's infrastructure and performance.

Root Cause Analysis: CloudWatch logs and metrics provide valuable data for troubleshooting and root cause analysis in case of failure. By correlating logs, metrics, and events, we can quickly identify the root cause of performance issues, software errors, or infrastructure problems. This capability is crucial for maintaining the stability and reliability of our bank application.

Cost Optimization: CloudWatch provides insights into resource utilization and performance patterns, enabling us to optimize our infrastructure costs in depth. By identifying underutilized or overprovisioned resources, you can make informed decisions to scale your infrastructure efficiently and reduce unnecessary expenses.

Alarm Notification and Automation: CloudWatch allows us to set alarms based on specific metrics, logs, or events. When alarms are triggered, CloudWatch can send notifications or execute automated actions to address potential issues promptly. This automation helps to minimize downtime and ensure the smooth operation of your bank application.

CloudWatch monitors database metrics such as query latency, connection concurrency, and storage utilization to ensure your databases are performing efficiently and handling workloads effectively. It can monitor CPU usage, memory consumption, and network traffic of your EC2 instances to ensure they are operating at optimal levels. Timely identification of performance issues can prevent application slowdowns or outages.

Architecture:

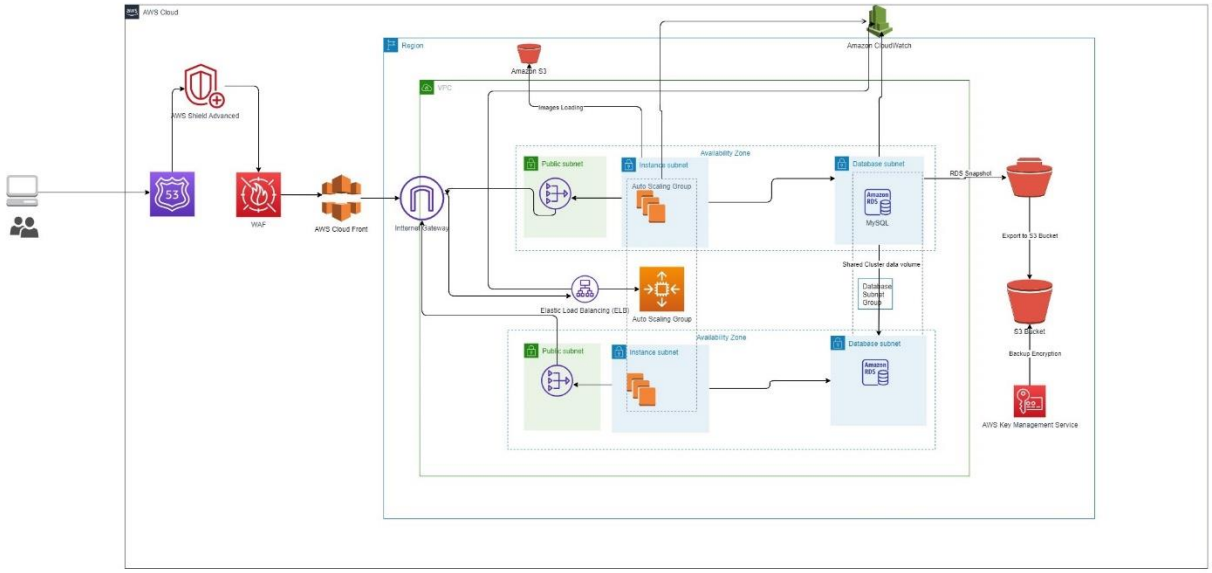

Figure 1: AWS Cloud Architecture of Banking Application[15]

References

[1] *Amazon.com*. [Online]. Available: https://docs.aws.amazon.com/vpc/latest/userguide/what-is-amazon-vpc.html. [Accessed: 04-Dec-2023].

[2] *Amazon.com*. [Online]. Available: https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/Welcome.html. [Accessed: 04-Dec-2023].

[3] *Amazon.com*. [Online]. Available: https://docs.aws.amazon.com/AmazonS3/latest/userguide/Welcome.html. [Accessed: 04-Dec-2023].

[4] *Amazon.com*. [Online]. Available: https://docs.aws.amazon.com/AmazonS3/latest/userguide/Welcome.html#S3Fe. [Accessed: 04-Dec-2023].

[5] *Amazon.com*. [Online]. Available: https://docs.aws.amazon.com/vpc/latest/userguide/vpc-nat-gateway.html. [Accessed: 04-Dec-2023].

[6] *Amazon.com*. [Online]. Available: https://docs.aws.amazon.com/autoscaling/ec2/userguide/what-is-amazon-ec2-auto-scaling.html. [Accessed: 04-Dec-2023].

[7] *Amazon.com*. [Online]. Available: https://aws.amazon.com/hpc/. [Accessed: 04-Dec-2023].

[8] *Amazon.com*. [Online]. Available: https://aws.amazon.com/shield/. [Accessed: 04-Dec-2023].

[9] *Amazon.com*. [Online]. Available: https://aws.amazon.com/waf/. [Accessed: 04-Dec-2023].

[10] *Amazon.com*. [Online]. Available: https://aws.amazon.com/route53/features/. [Accessed: 04-Dec-2023].

[11] *Amazon.com*. [Online]. Available: https://aws.amazon.com/cloudwatch/features/. [Accessed: 04-Dec-2023].

[12] *Amazon.com*. [Online]. Available: https://aws.amazon.com/cloudfront/features/?nc=sn&loc=2&whats-new-cloudfront.sort-by=item.additionalFields.postDateTime&whats-new-cloudfront.sort-order=desc. [Accessed: 04-Dec-2023].

[13] *Amazon.com*. [Online]. Available: https://docs.aws.amazon.com/vpc/latest/userguide/configure-subnets.html. [Accessed: 04-Dec-2023].

[14] *Amazon.com*. [Online]. Available: https://docs.aws.amazon.com/vpc/latest/userguide/vpc-security-groups.html. [Accessed: 04-Dec-2023].

[15] "VP Online - Online Drawing Tool," *Visual-paradigm.com*. [Online]. Available: https://online.visual-paradigm.com/app/diagrams/. [Accessed: 04-Dec-2023].