SOLUTION FOR E-COMMERCE PLATFORM

Sujith Delpachithra

1. Introduction

Assessment is based on the e-commerce platform, and as a solution architect I must consider few things when developing solution. Following points are captured when studying the requirement.

1. Company is a startup
2. Both services and products are available in the platform
3. CEO wants to go live as a minimum viable product
4. CEO expects a growth and need to facilitate
5. Need to provide both mobile and web-based applications
6. Only for Sri Lankans
7. Need to develop campaign management
8. Minimize the failure
9. Security concerns
10. Agile software development

This is a startup company. Therefor, we must consider about the cost of the solutions. As a solution developer, I would not recommend deploying full solution at once. As per CEO, I also agree with him to go live with minimum viable product where customers can do their basic needs in the e-commerce platform. Furthermore, this platform is required to have both products and services. Therefore, solution will be complex due to inventory/product management. In this solution, I will not discuss for much low level due to time constraints.

In addition, CEO expects a growth in the platform. Therefore, we must design solution to increase the performance without doing major architectural changes for the initial solution. As a solution architect, we must design solution at present to suit for future. Therefore, I have done some changes in architectural solution of the e-commerce platform to cater those future requirements.

Solution needs to cater both mobile and web-based applications for customer to use the platform. Therefore, I have design solution based on that. However, I would like to propose for the management to use responsive based mobile application at first to reduce cost and after that company can introduce a mobile app. Moreover, the suggested solution is designed for mobile app.

As per marketing team, they need to do campaign management and promotions based on the customer usage patterns. To achieve that feature, I must have some knowledge on data to collect for future

references. Therefore, solution is designed to capture data in both node and application levels. As per solution, all most all the data like network packet level data will be stored in platform to analyze and predict patterns and usages. I will discuss each of in the solution section.

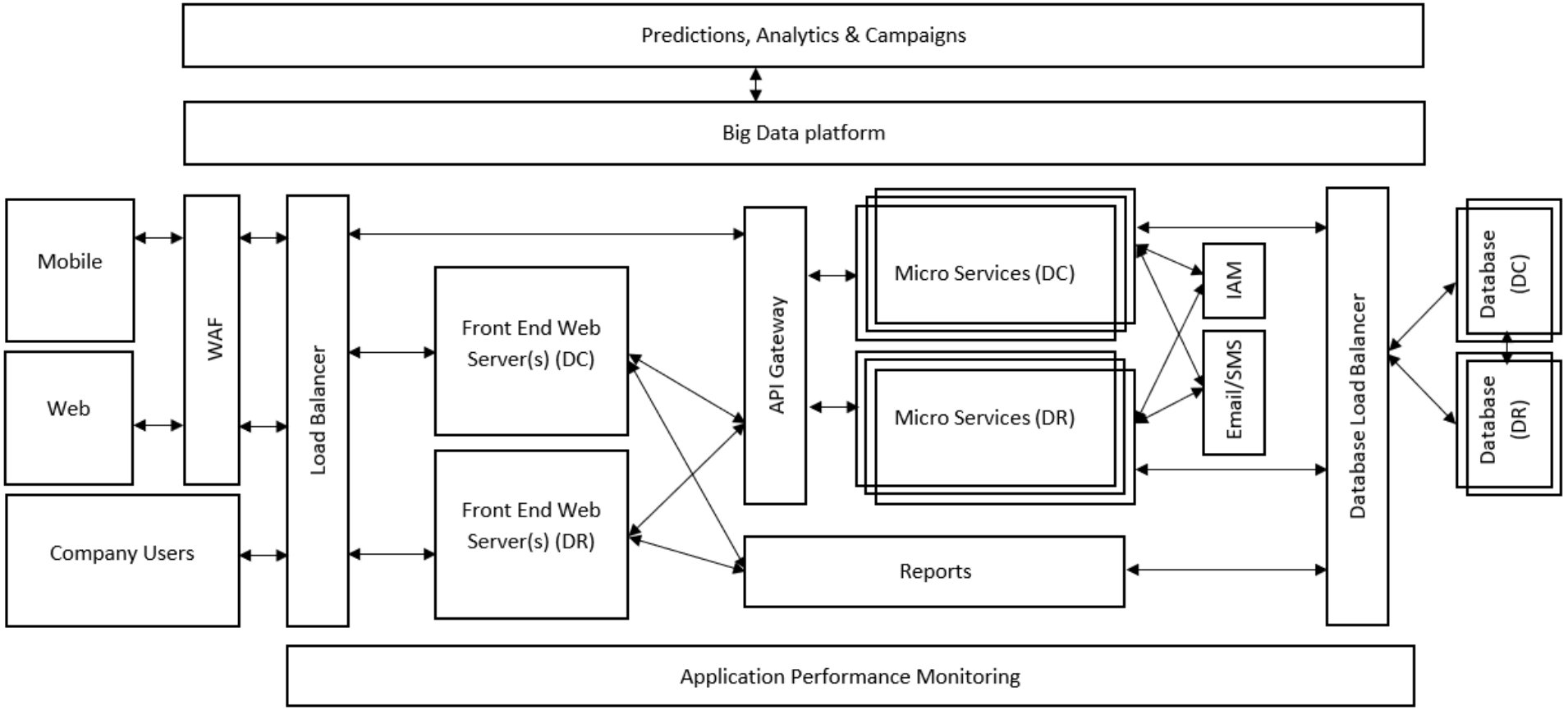
Availability of the systems is the main concern any company. Therefore, I have added feature to reduce failures of the system in each of point. However, we can remove some components to reduce cost of the solution. I will discuss each of the points where we can add or remove nodes in the solution description. In future, proposed solution must implement to reduce the failures of the e-commerce platform.

Security is the major concern of a solution architects of the company. As a solution developer, I always try to bring my knowledge to the solutions. Company must store customers' personally identifiable information in secure manner. Here in the platform, it is storing not only customer information but also payment details. Therefore, system must be fully secured. Since this is a new e-commerce platform attackers may try frequently. I have considered those concerns in my solution design.

One of the most important things is the software development phase. Since CEO wants to launch the platform as a minimum viable product, I have designed the solution to deploy as sprints. I will discuss the solution development in later of the solution section.

As per the requirement, there is no proper decision about the deployment of the solution. Whether it is on premises or cloud. Therefore, I will design two solutions for both on premise and cloud solutions.

Figure 1: High level design of the e-commerce platform (On-Premises)



Source: Author

2. Solution 1 (On-premises deployment)

Figure 1 is illustrating the high-level solution diagram for the e-commerce platform. In the solution diagram, I have tried to implement all the required features from stakeholders as discussed in introduction section and other suggestions from my experiences. However, due to time constraint I'm not going to deep dive into the architecture.

2.1 Discussion

Solution is capable of handling both mobile and web applications in the proposed solution. For security purposes, I have implemented a web application firewall (WAF) Infront of both web and mobile apps. Platform must be secured by internet users. However, I have excluded the internal stakeholders from the WAF. The reason behind this is to reduce traffic through WAF and improve performance of the platform. In addition, I assume that employees are connected through behind the company firewalls. Moreover, employee access can be secured using company VPN.

Both mobile and web application are exposed to WAF via application load balancer. This application load balancer will be able to balance the load depends on the traffic to sites. Micro services are exposed over API gateway to secure the micro services from unauthorized access. Frontend applications must use OAuth2 authentication to access the micro services. Moreover, there is a requirement to store customers' data in a secure manner. Therefore, it is suggested to deploy the IAM (Identity Access Manager) for user authentication and authorization. User management micro service(s) will directly use the IAM to create/validate users in IDM. I am proposing to differentiate two user categories in IDM. They are internal and external users. Because of that differentiation, we can store more data about customers in IAM than internal users. After successful validation of users in IDM, other micro services in the platform can use IAM token(s) to define the user privileges (authentication/authorization).

As per requirement, I have identified following high level micro services for the e-commerce platform. However, we need to go deep into each micro service and identify proper segregation of services to reduce complexities.

1. User Service
2. Product Service
3. Order Service
4. Inventory Service

5. Plans Service
6. Discount Service
7. Payment Service
8. Support Service

I have segregated the frontend and backend of the solution. This will improve the application performance and will lead for decoupling of business logics from user interfaces. Because of this each segregation, we can modify backend anytime with minimum changes to frontend. However, we must design the micro services properly with correct payloads/endpoints to minimize the interface changes between services and frontend applications. Because of micro services, we can increase the number of micro service instances to improve the application performance. For example, if there are high number of requests for order management micro service, we can increase its instances to increase the performance.

This e-commerce platform must be able to communicate with email and SMS services to send notifications to customers. Therefore, I have integrated a queue management solution for email and SMS. After micro services execution, they will push the notifications to email and SMS components. Those components will execute email/SMS based on queue management. We can use ActiveMQ or RabbitMQ as messaging platform to store messages. I will discuss relevant technologies in later chapter.

Customers and internal users also will connect to web application through frontend application. I suppose to have two separate applications for back office and customers. Because of that, it will reduce the complexity of the management in later. We can use existing micro services for both customer and back-office applications. As per proposed solution, N number of application servers will be deployed in data center (DC) and N number of servers are in disaster recovery (DR) data center since there are no clear definition about traffic. In addition, CEO expects a growth in e-commerce platform, architecture must be able to facilitate. However, I would like to suggest using two application server instances in DC and one instance in DR in the initial phase of deployment to reduce cost. After having proper statistics, QA team can be performing a performance acceptance testing (PAT) for required traffic and request for fine tune hardware and software according. All the traffic is followed through load balancer to frontend application. If there is a failure in DC servers, load balancer can be configured to send traffic to DR servers. Moreover, we redirect DC server failures automatically using probing in frontend applications. Therefore, failures in DC servers' traffic will be switched to DR site server automatically without having any human intervention while engineers are solving the DC issues. As per Figure 1, application performance monitoring application (APM) is deployed across the e-commerce platform. Nodes logs, application logs will be redirected to APM

tool using APM agent. Then, APM can analyze the error using patterns and action according. APM will generate server failures directly to application support team to action. I will discuss about tools in later chapters. Here, I am suggesting deploying DC and DR as ACTIVE/PASSIVE servers. Because DR site must deploy in another location and there may be a latency between DC and DR sites. Global server load balancing (GSLB) will help to handle DC server failures.

I have suggested to bypass API gateway to get report since there is no security issue or performance issue there. In addition, reports are used by internal staff of the company. Therefore, developers can develop reports using direct database access without having micro service access. However, it depends on the report type. We must think deep about reports in depth architectural discussions.

Databases are connected to applications using database load balancer. However, as per application servers, it is supposed to deploy ACTIVE/PASSIVE databases for failures. As discussed above for application servers, database must be deployed in separate location for disaster. Therefore, to reduce latency, it is better to deploy databases as ACTIVE/PASSIVE. ACTIVE node data changes (delta of changes) will be synced between DC and DR databases to reduce data loss. However, it is recommended to deploy hourly/daily data backup in tapes to reduce the data loss in case of both DC and DR site failures. Backup retention policy can be agreed upon discussion with storage teams.

There is a requirement from marketing head to deploy promotions based on usages and other statistics. Therefore, it our duty to collect as much as data from customers to analyze and predict patterns. Here, in the solution architecture, I'm proposing to deploy a big data cluster to collect every possible information. In Figure 1, I have illustrated the big data platform across the WAF to database layers. Using the collected information, big data team can analyze user patterns predict promotions and campaigns.

2.2 Software Development

As per assessment, there is a requirement to develop software as agile development. To achieve that, as discussed earlier, I have separated the frontend and backend applications. In addition, I have suggested to develop separate applications for internal and external users of the platform. Because of the use of micro services, we can use agile development practices for this. Moreover, as CEO wants to deploy the platform as MVP, we must identify MVP features in depth analysis with relevant stakeholders and agree on development. As per solution design, we can add features later phases on the platform enhancements.

2.3 Nonfunctional Requirement

Following nonfunctional requirements have been considered when designing solution.

2.3.1 Availability

Solution is deigned for high availability. As discussed above in the paragraphs, all possible failure nodes are designed to have disaster recovery nodes. Application servers, database servers, micro services are also deployed in DR site for recovery purposes.

2.3.2 Scalability

This e-commerce platform is designed to scale if there is any growth. As illustrated in Figure 1, e-commerce platform can expand its resources. Application servers, micro services and databases are designed to deploy many instances/nodes to provide scalability to the platform.

2.3.3 Security

In the proposed solution, it has used a WAF, API gateway, IAM to secure the e-commerce platform. Usages of these components have been discussed in above paragraphs.

2.3.4 Pricing

Since this is designed to deploy as a on-premises solution, solution will be bit costly. To reduce the cost, initially we can start with small number of required components. In initial stages of the deployment, we can implement only DC site. In addition, we can deploy a smaller number of application servers, micro services, and database servers to reduce cost. I will discuss about AWS version in solution 2.

2.3.5 Operational excellence

Proposed solution is developed to decouple components. Because of that, we can achieve operational performance. Moreover, internal staff will be provided a separate back-office system which will not affect for the external users. In case of back-office application failure, external user may be able to use e-commerce platform.

2.4 Technology stack

As a solution architect, I would like to suggest following technologies to implement proposed solution. However, some components depend on the deep requirement analysis. Some of the below tools can be used freely. As a startup, it would be an advantage.

2.4.1 Frontend application

I would like to suggest latest frontend application frameworks like Angular or React. Using these frameworks, we can provide interactive applications for customers which will attract customer to platform. These applications will call micro services.

2.4.2 Backend application

Spring boot can be developed to implement micro services in the platform. Because of the use of Spring boot, we can scale up or down microservice as required in simply.

2.4.3 Application performance manager

As discussed above, I would like to suggest deploying Elastic Search platform to feed logs and other relevant data. After that, APM application can detect any performance related issues in the platform and inform to support team to action through SMS or email. In addition, we can use Grafana to visualize the findings. Some of these tools are free and can implement easily.

2.4.4 Big data platform

As discussed, we must use big data platform (Apache Cassandra) to collect all required data to analyze and predict promotions. After that, those data can be used by big data team to analyze users, patterns using AI/BI tools like Apache Spark and predict future behaviors of the customers.

2.4.5 Database

I would like to propose a MySQL cluster to store data in the platform. In future, we can change the data source to another enterprise level database.

2.4.6 Application servers

We can use WildFly, JBoss or Tomcat servers to deploy frontend application to reduce cost in initial launch of the platform.

3 Solution 2 (Cloud deployment)

As mentioned in assessment, it has defined some values for failures, data losses. I think that the examiner wants to apply some cloud deployment strategies in deployment. However, I'm not an expertise on cloud technologies. At present, I am learning on AWS and I'm willing to apply my learning to develop a solution. I wish to complete my AWS associate software architect certification in January 2023.

3.1 Discussion

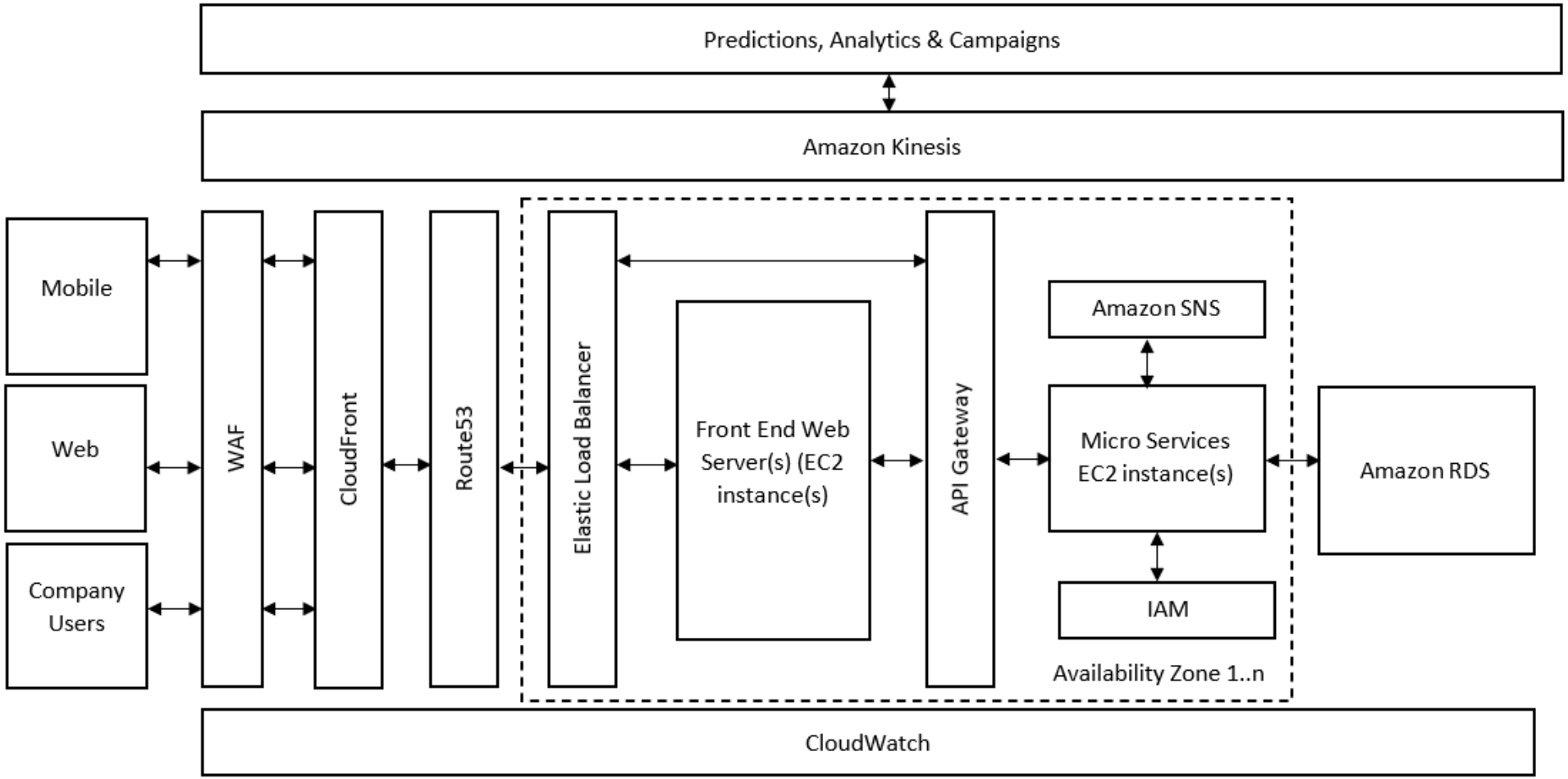
Figure 2 illustrates the high-level design of the e-commerce solution design which can be deployed in AWS cloud. In the proposed solution, virtual private network implementation is not designed. However, we can do it using easily using Amazon VPC configurations. As per solution 1, this solution also capable of handling web, mobile and internal user requests. Since e-commerce platform is opened for public through internet we must secure our cloud using WAF. Therefore, I have proposed to use a WAF Infront of all other components to secure the cloud. All incoming traffic from the internet will be secured by WAF. In solution 1, internal user traffic is not secured by WAF. However, in solution 2, we must secure it since users must login to platform through public internet.

After successful security, requests will be sent through CloudFront to enable caching to speed up the processing in future. CloudFront will manage application caching in edge locations to provide maximum performances for the application. Because of the CloudFront, static traffic will not hit backend micro services and provide cached responses.

In solution 2, I have proposed to user Amazon Route53 to manage routings of the requests. The reason to use Route53 is to provide high availability for the platform. We must use Route53 to route traffic between different availability zones. There are few routing policies available when using Route53 in Amazon cloud platform. In the initial stage of the platform deployment, we can use simple routing policy. I am supposing to deploy application in at least two availability zone to provide high availability for the platform.

Route53 will redirect traffic to elastic load balancer to provide high availability and scalability for the e-commerce platform. Elastic load balancer will be deployed in each availability zones. It is proposing to deploy at least two application web servers (frontend applications) with elastic load balancer. Depends on the growth of the requests, system will be able to facilitate the more EC2 instances with frontend application deployments. This will improve the performance of the system.

Figure 2: High level design of the e-commerce platform (Cloud)



Source: Author

As per solution 1, solution 2 backend is proposed to develop based on micro services. Each micro service can be deployed in dedicated EC2 instance, and we can increase the number of instances depends on the loads to each micro services. Above solution1, micro service segregation will be used in solution 2 as well. For example, if there are many requests for order micro service, we can increase the number of EC2 instances of that micro service. In addition, it is suggested to deploy these micro service instances across at least two availability zones to improve the performance of the e-commerce platform.

Amazon is providing IAM solution in its cloud platform. I am proposing to use Amazon IAM features to maintain internal and external users of the e-commerce platform using policies and roles. Amazon IAM is one of the main components of the Amazon cloud. Therefore, we can use its features for our e-commerce platform to authenticate and authorize users to enable access the features of the e-commerce platform.

Amazon SNS is designed to use for sending messages. Therefore, microservices' output messages will be sent to Amazon SNS to deliver to customer. For example, after successful transaction, we can deliver notification message to customers using Amazon SNS through email or SMS.

Amazon supports both relational and NoSQL databases. Amazon relational database service (RDS) can be used as our database services. As per Amazon specification, there are a few supported relational databases. This Amazon RDS will look after the scalability, durability, performance, backups, alerts etc. in relational databases which we are used in our platform. As in solution 1, I would like to suggest using MySQL as initial database deployment.

Amazon is providing features to collect data using streams. Amazon Kinesis is the feature that has been developed by Amazon to handle millions of data. In Kinesis, there are a few approaches to process data that are collected in various steps of the platform. I would like to suggest using Amazon Kinesis Analytics to process data in real time and store in Elasticsearch. Moreover, we can store processed data in S3 storage. Using these processed data, marketing teams can design promotions or campaign to attract users to e-commerce platform.

I have designed the solution to use Amazon CloudWatch to monitor usages of the solution. In CloudWatch, we can create alarms to get usage alerts and take actions accordingly. As a startup company, we must consider about costing. With the use of Amazon CloudWatch, it will generate alarms and alerts depending on usages to support teams. Therefore, they can optimize the resource utilizations.

3.2 Nonfunctional Requirement

With the use of cloud technologies, following nonfunctional requirements can be achieved.

3.3.1 Availability

Solution is designed to deploy on at least two availability zones to make the platform available. Moreover, it has used Amazon EC2, ELB, RDS components in the proposed solution. These components are designed by Amazon to make sure applications availability.

3.3.2 Scalability

In proposed solution, it has used EC2 instances to deploy frontend and backend applications. Depending on the usages, it can increase the number of EC2 instances easily in less than 5 minutes. In addition, other used Amazon components are supported for scalability if required. Therefore, it can guarantee that this platform can be scaled in future.

3.3.3 Security

In the proposed solution, it has used a WAF, API gateway, IAM to secure the e-commerce platform. Because of the use of these Amazon standard components, it can be guaranteed the security of the solution. However, we need to consider more when implementing the solution since it is hosted in public cloud.

3.3.4 Pricing

Pricing for solution 2 will be less than solution 1. We are paying for Amazon for the usages. In solution 1, we are paying much price for hardware. In this case, it will be less and do not need to consider about hardware. Amazon will take care about hardware, and we need to pay for usage only.

3.3.5 Operational excellence

Because of the Cloud deployment, it will provide some convenience for the engineers. They must follow some standards when deploying application in cloud. Because of cloud features like availability, scalability, durability etc., operations cab be executed smoothly.