CLOUD BASED

HEALTHCARE MANAGEMENT SYSTEM (CHMS)

Mithun Majumdar

(Technical Lead and Product Owner)

ABSTRACT

Serverless computing is emerging cloud service architecture in recent years for executing distributed applications, where services are provided based on a pay-as-you-go basis. It allows the developers to deploy and run their applications without worrying about the underlying architecture. Serverless architecture has been popular due to its cost-effective policies, auto-scaling, independent and simplified code deployment. The healthcare service can be made available as a serverless application that consists of distributed cloud services achieving the various requirements in healthcare industry. The services offered may be made available on premise on a cloud infrastructure to users and health service providers. This chapter presents a novel model for serverless architecture for health care systems, where the services are provided as functional units to various stake holders of healthcare system. Two case studies related to healthcare systems which have been adopted serverless framework are discussed.

INTRODUCTION

Serverless computing is a comparatively current computation model or paradigm which emerges from the rapid and continuous advancement of the Cloud. Such a type of service is provided by serverless computing with which users or developers can write and deploy application logic without the provision or management of containers or servers. Adopting the serverless model holds a significant effect on various aspects of software engineering in particular the pricing model, the development process, and QoS (Quality of Service) assurance. Even though serverless computing has various advantages and benefits, such as no up-front provisioning, zero server management, high availability, pay only for the resources used, and auto scalability, it has also certain shortcomings which should also be taken into consideration: As currently offered by providers, it is inappropriate for long term tasks through the limited-time service can run; furthermore, there is increasing complexity of the underlying architecture, which is strengthened by the lack of suitable operational tools.

Serverless computing is one of the most useful buzzword in the era of cloud computing

which present an architecture and deployment model to run the code by abstracting underlying infrastructure and the resources. In this platform developer does not concern about operational environment which includes resource configuration, environmental setup, security, reliability, encrypted messaging etc. It has been widely adapted by number of MNCs for their project to have more economical outputs. It poses a lot of challenges as well as opportunities to develop scalable application where user can focus only on writing business logic without worrying about the infrastructure. Serverless computing model can be thought of as the intermediary between PaaS and SaaS. It is based on function as a service model where an application can be developed by composing number of discrete functions which can be executed independently at a separate container at the server side. Sometimes, it makes it difficult for developer to understand the execution environment of their application as most of the controls are abstracted from the user. Data and application security is another concern in case of serverless platform as users are totally unaware of the location of the servers where the data and applications are residing. In serverless framework, functions are treated as the unit of execution, which can run in separate container to provide an isolation environment. Developers are facing challenging in writing serverless application as the framework is very limited to pre-packaged library.

Serverless architecture is observed to be more flexible as compared to platform as a service and less stringent to the software as a service. In software as a service, applications are executed as a unit of execution where all the processing relies on API calls through which certain software are provided as a service. All the code integration is not fixed with any hosted server. In fact, they are executed through AP calls. This approach has been used by many giant firms like Google, Yahoo etc. In comparison to SaaS, serverless provides a flexible environment which allows the developers to write and execute code in less expensive and auto-scalable manner. Serverless computing model has been widely accepted at many companies like Google, Yahoo, IBM, Microsoft etc. With the advancement of information and communication technology, health care systems has grown significantly over the past decade by capturing the huge amount of digital data. The healthcare business is transitioning to value-based care, which necessitates the collection of new data. Costs must be minimized for healthcare services to be accessible to a growing number of people. All of this is feasible thanks to cutting-edge technologies like AI, machine learning, and serverless computing. The healthcare IT business can profit from serverless technology in a variety of ways. Serverless computing has the potential to drastically alter the healthcare industry. In healthcare tech start-ups, here are some of the primary advantages of employing serverless technology.

1. **Improves the Interoperability:** The amount of data in the healthcare industry is enormous and growing all the time. The use of serverless technologies has made storing massive volumes of data simple. Outside of the hospital, the data can be accessed, cleansed, and modified for use in other investigations.

2. **Quick Growth or Rapid Changes:** If you launch a medical app, it's hard to predict all the features users will need. You can try to implement them all at once, but it will require a lot of time and cost a fortune well before the app is launched and you receive any income. With serverless technologies, a development team can quickly alter, update, or fix the app without affecting its performance. In other words, instead of guessing, you can match the app with real user expectations.

3. **High-Latency Background Tasks:** Using serverless architecture for healthcare applications that process loads of data or multimedia may be a natural choice. Thanks to scalable storage and asynchronous data processing, your users will continue enjoying premium app performance, regardless of how much data is being uploaded at the moment.

4. **Client-Heavy Applications:** The shift toward value-based care has made patient engagement and experience a top priority for many healthcare organizations, resulting in more patient-centered applications. With serverless architecture, the app's functionality can began on local servers closer to users, reducing latency and improving performance.

5. **Better security:** There is a risk of a data breach or other security incidents while using servers in the traditional way. Because the servers are protected by large IT companies like Amazon, serverless computing has far fewer security risks**.**

SERVERLESS COMPUTING AND ITS CHARACTERISTICS

A Serverless computing differs from the traditional cloud computing concept (we call it as a serverful in this chapter) in the sense that the infrastructure and platforms in which resources operate, are transparent to customers. In this way, customers are only concerned about the desired performance of their application and the rest is referred to the service provider. Engineers have to be conscious of these structures while selecting a platform. Some of the characteristics of serverless computing are discussed below:

1. **Cost**: In the serverless pricing model, the cost is usually calculated for the resources consumption only when the applications are executed through serverless functions. So when the applications are idle, the cost may be measured as zero which is the most attractive features in serverless platform. Metal-based services or resources, such as memory or CPU, as well as price models, such as less expensive discounts, vary from provider to provider.

2. **Performance and Limits**: The serverless architecture is found to be more scalable as compared to other deployment models. However, it also limits the resource requirements by the business logic to control the incoming traffic which can able to provide better service. The limitation may be increased in future based on the user demands.

3. **Programming Languages**: Another attractive feature of serverless framework is that it allows the users to write the code diverse number of programming language which includes Java, Java Script etc. It has large number of APIs written in different number of languages. Some serverless frameworks allow writing programs in any language which can be integrated to Docket image that supports diverse sets of APIs.

4. **Composability**: Building application is quite efficient as compared to other models in cloud as it allows the users to write code in modular manner without worrying about the composability. User can write the function independently for different types of business logic. Serverless framework provides a different ways of

composability where a function execution may be requested from another function. All these things will be taken care by the serverless framework. In this way, it allows the developers to build complex application in a simpler way.

5. **Deployment**: Ease of deployment is another attractive feature of serverless framework where users' needs to packages all source file in the form of Docker image in the form of binary file. The AWS Cloud Formation is used to deployment of source code. It also allows the users to automate the deployment process by continuous integration and continuous deployment (CI/CD) system.

6. **Security and Accounting**: Security is one of the major concerns when user deploys there code for execution in cloud. Serverless framework has elegant mechanism for separate the performance of tasks of different users. Based on the actual resource consumption, the accounting information is communicated in a separate channel.

7. **Monitoring and Debugging:** Architecture supports the correction of basic errors by using print statements recorded in the log file. Additional skills may be provided to help users identify issues, track errors, and better understand the work environment.

8. **Auto-scaling:** It is expected from a cloud platform to be capable of scaling resources instantly, automatically, and available on demand. Serverless models are come complete containerization with minimal initial delay, allowing for the provision of thousands of events within a few seconds. Likewise, in the absence of application requests traffic, operating conditions reach zero to keep inactive application execution.

Compared with IaaS frameworks, serverless environments offer different tradeoffs in terms of flexibility, control, and cost. Specifically, it compels clients or developers to consider carefully the cost of business logic when modifying their applications, rather than delays, flexibility, and scalability that is where a considerable development attempt has conventionally been utilized.

A serverless platform benefits both clients as well as providers. From a client's point of view, it is found to be more cost effective as compared to other deployment models due to its flexible pricing model. According to the pricing model of serverless, billing is setup only for the actual resource consume. It does not account for the idle times. Another major advantage is that client is not responsible for configuring and management of container or the server at the backend. The responsibility is usually migrated from client to the server side. Developers can focus only on the business logic without worrying about any kind of infrastructure. As the functions at the server side are executed in stateless manner, the service providers have more control over the application stack which allows them to configure security in transparent manner.

Despite of several advantages, some limitation related to transparency is there in serverless platform. It is quite challenging to measure the actual performance of serverless application as a very limited performance and cost models have been emerged recent years. This leads to confusion in deciding in adaption of serverless in business solutions. Based on the performance analysis of standalone application

developers can make the choices for solutions to the business problems. Another limitation lies in compatibility of complex and large application in serverless framework. As it's based on a FaaS (Function as a Service) model, where an application developed in a modular fashion, that's why some complex applications may not be supported in the framework. It might not support the latest version of Java or few libraries may not be available in the framework which makes challenging for developing a complex application.

SERVERLESSARCHITECTURE

Serverless architecture is a new paradigm for software design where the client or the developers are allowed to upload and execute their code without worrying about the servers and the infrastructure. It can also be called as function as a service deployment model in the era of cloud service provider. Developers are offloading the responsibility of managing the server, configuring infrastructure, security patches to the service providers. They can only focus on writing the business logic. The architecture is also found to be cost efficient as compared to other deployment models as it is based on pay-as-you-go. User doesn't have to pay for the idle time. They only have the pay for the exact amount of resources consume while the application is running at the server side. The backbone of serverless architecture is the function-as-a-service model, where users can focus on writing the application in the form of set of discrete functions, where each function may be corresponding to a specific task in the application.

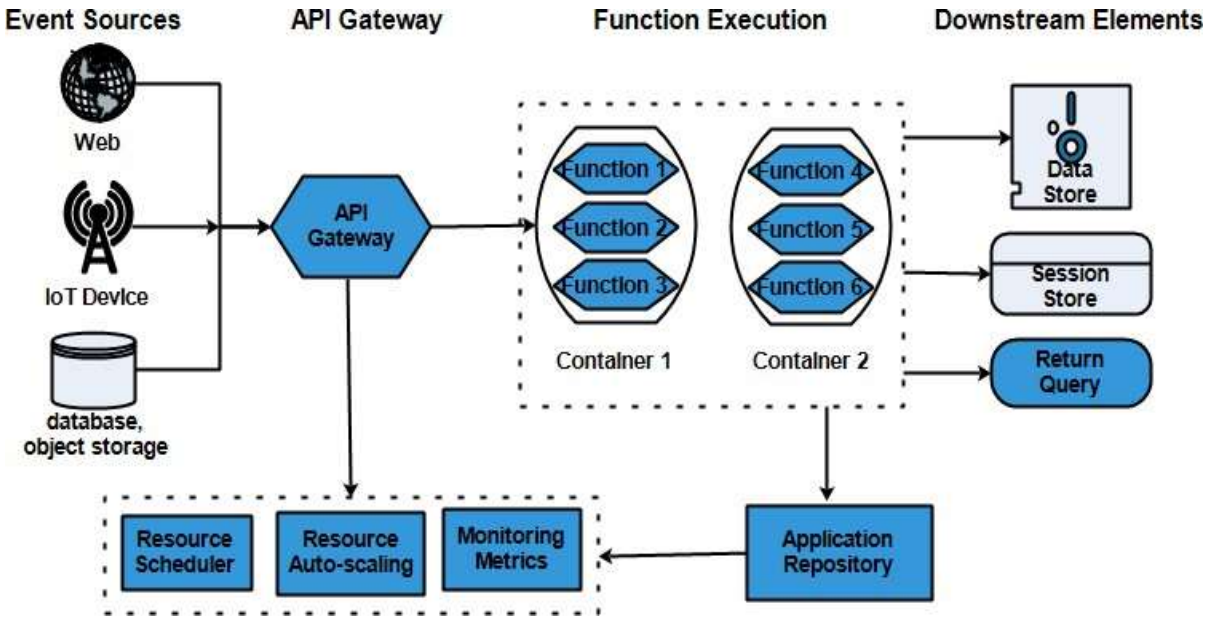

Figure: 1. Serverless Architecture

Each of these functions is required to upload to the cloud along with set of events through which functions can start executing. The events may be in the form of http request or any king of updates in the database or any king of notification from any IoT device etc. Upon triggering, the serverless vendors execute the function on a running server or the container. If no server is running, a new server may be provisioned for execution. All the triggers are to be PaaS through the API gateway on which the specific features of the application as defined earlier are processed. It may be related to

resource scheduling, the auto-scaling or the logging the events. Based on the request received from the API gateway the appropriate action will be take care by the cloud vendor. The scheduler will decide which execution node or the containers are to be provided to serve the request. The auto-scaling module is executed when the application demands for the additional resources. Once the execution environment is configured, the corresponding function is start running. The execution environment is usually abstracted from the client side. Once the execution of the application is completed, all the resources that have been allocated are de-provisioned and the created environment is destroyed. The state between the function calls is destroyed. Hence, it is also known as stateless execution framework. The execution process in serverless architecture is presented in Figure. 1.

FUNCTION AS A SERVICE (FAAS) COMPUTING MODEL

A serverless computing paradigm is a rising approach for growing cloud-primarily based applications. IBM defines it as "a method to computing which offloads responsibility for commonplace infrastructure management tasks (scheduling, provisioning, patching, scaling, and many others.) to cloud vendors and gear, allowing developers to focus their effort as well time on the enterprise logic particular to their programs or procedure. Serverless computing calls for much less proficiency than other self-controlled tactics. Users do no longer control without delay the runtime and infrastructure of the system, however, as a substitute, delegates its operations to the cloud vendor. Moreover, cloud companies can install finer-grain pricing regulations (e.g., based on service requests) for any of the presented services, which usually leads to decreasing costs for developers.

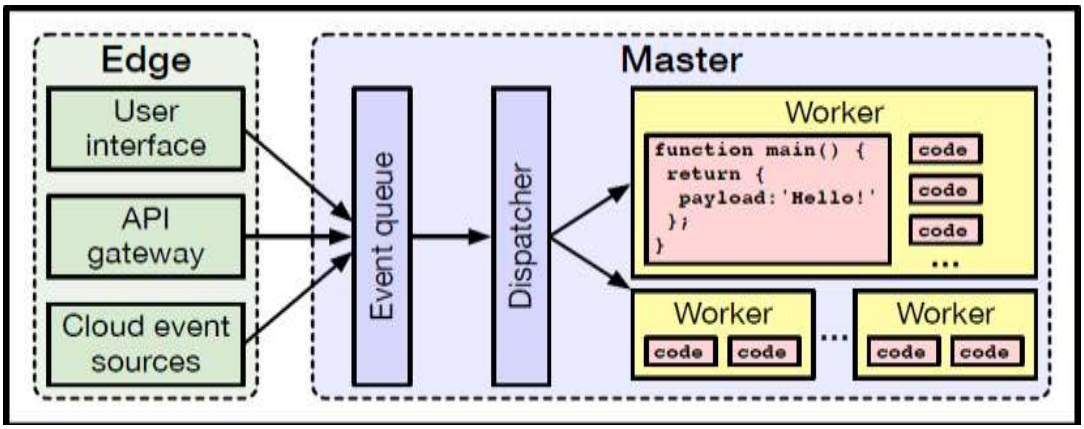

Figure 2. Traditional FaaS architecture. The key points to benchmark are: performance of a worker (i.e., execution speed of a function) and quality of the auto-scaling mechanism.

One can make a distinction between several serverless computing paradigms: (1) Function as a Service (FaaS) implemented as an example by means of AWS Lambda(2) Software as a Service (SaaS), through Google Cloud and (3) Database as A Service (DBaaS), as to be had through Microsoft Azure for PostgreSQL. FaaS can be taken into consideration as a hybrid among Platform as a Service (PaaS) and the Software as a Service (SaaS) provider version: infrastructure, as well as information, is fully controlled via the cloud vendor, even as the software is handled through the clients or developers.

FaaS systems or Function-as-a-Service permit developers to create and host an application (consisting of a single function or set of functions) that scales dynamically. These applications are meant to carry out a single function on demand and then forestall running till needed once more. Such systems provide enormous cost financial savings to application developers, permitting them to pay for only the instances all through which their serverless applications are executing. Moreover, web hosting companies can also recognize widespread savings in resource intake, for the reason that serverless functions do no longer require continuously running dedicated containerized or virtualized hardware resources. Even though at the start designed for the cloud, these frameworks are properly suited for fog/edge computing environments, in which assets are necessarily constrained. Figure 2 illustrates a standard FaaS framework. In the standard FaaS model, developers pass the setup, maintenance, along with the control of a compute node (like containers, virtual machines, or even bare metal). Instead, clients offer business logic for specific functions to be deployed and developed to the cloud. Specific activities such as storage, database situations, or HTTP requests cause function execution. Then, the cloud vendor handles the requests, scaling as well as the availability of resource requirements. Though the benefit of the serverless model, it is currently tough to determine a selected FaaS provider based on standards along with pricing scheme, performance, or workload adaptability.

HEALTHCARE MANAGEMENT SYSTEM IN CLOUD

Cloud computing refers to Internet infrastructure and on-demand availability of services, which allows the clients to utilize accessible cloud computing services and resources anytime from anywhere. It is a new paradigm of offering cloud computing resources, now not a brand-new technology. Examples of typically utilized non-healthcare services or applications like Google Docs, Microsoft Hotmail, etc. even as some better-recognized applications in health-care consist of Google Health platform, Microsoft HealthVault, etc. Despite that, in comparison to traditional cloud computing, the serverless paradigm offers new benefits such as computing resources to be had on-call for, removal of an up-the-front commitment utilizing clients, and price for a short duration to execute the client's business logic. Various blogs, articles, forums, research works have pronounced serverless application in an enterprise, commercial enterprise, national protection, transportation, education, security, etc. Health care, as with every other service operation, requires systematic as well as continuous innovation to remain efficient, cost-powerful, and convenient, and to offer outstanding services.

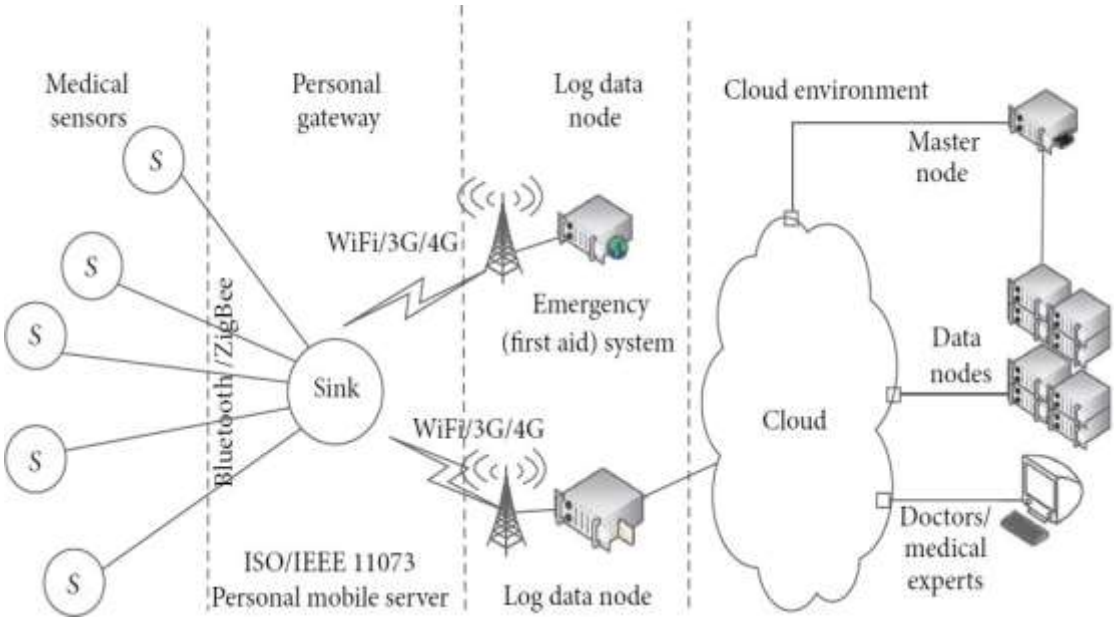

Figure: 3. The health system architecture in cloud environment

A wide number of researches have been carried out in solving the problem of digitization in the era of healthcare sector. Cloud service has been emerged as a promising technology in providing various healthcare related services over the Internet. It can reduce electronic healthcare cost which includes networking, hardware, software, proprietary licenses etc. It has been observed from many literatures that bioinformatics sector is getting number of benefits in adopting serverless computing model. Biomedical industries are facing difficulties in meeting the deadline to fulfill the research objectives especially when analyzing vast amount of genome data and with computational resource which are expensive in nature. Serverless platform is emerged to right solution for solving many such problems in the era of healthcare sectors. Despite of number of benefits, it is associated with lots of challenges and issues in implementation which should be pointed out.

It can be observed from the literature that a number of framework for healthcare sectors have been emerged by the cloud vendors to provide better service in healthcare sector. Different deployment models like PaaS. IaaS and SaaS are proving different kinds of services to healthcare sector. Each of them is applied at diverse number of domains. One of the popular applications in healthcare system is related to automate the process of patient data collection where IoT is integrated with cloud deployment models. It has number of advantages such as, it reduces the overhead for manual collection of samples, eradicate the mistakes due to typing error and the ease of deployment. A separate set of cloud protocols have been developed to provide security for mobile and other multimedia devices. A number of sensors can be planted on the mobile devices which will able to send the health-related information at the cloud. The cloud server can be used to store and process the data to extract the health behavior of user which can be sending back to the client side to the healthcare personals for taking appropriate actions. The cloud framework has been used to provide immediate attention of doctor in a fraction of

second which consequently improve the healthcare system. The layered architecture for integration of IoT based cloud architecture with the healthcare system is shown in Figure 3.

The layered architecture as shown in Figure 3 consists of numerous sensors to collect patient information using wireless protocols. The sensors can be used to measure the pulse rate, oxygen level, sugar level heart rate etc. Each of sensors collects various information and propagated toward the local area gateway using Bluetooth or the wireless protocols. Sometime the user's telephone can be used as the personal gateway which can be used to send the personal health related information to the cloud. The sensor resides on the mobile device periodically collect and sends the data to the cloud which can be accessed by any legitimate users. The data collected through sensor can be put into the message queue which may have various health related parameters. Apart from the actual data, the message queue is required to have metadata which includes mobile IP address, location, destination logical address etc. It can be collected through cloud specific optimized algorithms. After processing the messages, the actual instruction or the recommendation can be generated for the patient or the other stake holders. The information from the cloud service may be compressed and secured to provide end to end encryption delivery, which can be displayed on the mobile devices of the user. A specific set of cloud services may be provided to the patient for intensive care. In addition to that, the doctors and care givers can have direct communication with the patient when required.

SERVERLESS COMPUTING FOR HEALTHCARE MANAGEMENT SYSTEM

The healthcare business is transitioning to value-based care, which necessitates the collection of new data. Costs must be minimized for healthcare services to be accessible to a growing number of people. All of this is feasible thanks to cutting-edge technologies like AI, machine learning, and serverless computing. The healthcare IT business can profit from serverless technology in a variety of ways. The pandemic-related events of 2020-21 have certainly changed how various industry sectors are thinking about digital transformation. A recent survey by McKinsey said that COVID-19's impact had accelerated digitalization to such an extent that some companies were reporting that their integration schedule had been brought forward by three to four years. According to McKinsey, this demonstrated an aggressive attitude to a required digital transformation, with healthcare proving to be one of the most eager industries for change. As a result of the pandemic, the survey noted, companies had seen demonstrated value in digital investments, citing smooth operations, faster service and better customer experience as reasons why digital was the way forward. If we take healthcare as the example, hospitals are introducing online appointment booking, Tele- consultations (AKA Tele-health), remote diagnosis, remote monitoring and the full-scale digitization of patient records, to name but a few. Insurance companies have also shifted to the digital world, with policy purchase, claim initiation, claim verification and approval – all available digitally. Both sides of the healthcare coin – providers and insurance – need an effective digital strategy, especially when you factor in time-to-market, lowering costs and ROI.

MAJOR CONCERNIN HEALTHCARE SYSTEM

In speaking to migrant health care corporations or deploying their solutions to the cloud, we learn that they are usually not sure about how to start a difficult transition journey. First, corporations measure the capabilities as well as the resources required for extending the cloud model solution. The conception of converting and dismantling the common answer into a fast-paced environment matters of concern. Cost savings is the third one but not the last. Healthcare management systems rely upon the following three aspects which are used to address the above concerns:

1. **Enhancing Speed of Delivery and Scalability**

   Serverless architecture possibly leading several advantages if you configure it properly. Serverless applications on the scale of the server that does not automatically supply with growing usage and demand. Additionally, in the case of a few two-week sprints, functionality and infrastructure can be provided in a cloud-based environment - usually through infrastructures such as a business logic, which represents providing and managing infrastructure with descriptive code that facilitates translation control, testing, and rollbacks. Such frameworks are GCP, AWS Cloud Formation, Terraform, Pulumi, Azure Resource Manager, etc. Scalable serverless solutions might be utilized throughout the digital healthcare environment. For example, Medicare & Medicaid Services uses AWS Cloud services to host infrastructure that can handle hundreds of thousands of users at once. The corporations which are dealing with large data sets, the same powerful AWS features may be employed to build data repositories for better data collection, secure storage, and processing in an inexpensive way and analyzing large amounts of data.

2. **Ensuring Compliance**

   Healthcare enters the digital train through programs such as collaborating with the goal of well- designed serverless cloud architectures. For proceedings as AWS, as an illustration, the main objective is having as many services as possible and no longer work while maintaining along with other compliance. The AWS utilize the AWS secure environment to process, manage, and store protected health information (PHI). According to customer feedback, AWS is makes attempt to add more services to the system, which includes serverless techniques.

3. **Cost Optimization**

   Because there is no requirement to configure and manage the infrastructure, clients may easily download and upgrade products, reduce costs and increase speed to live. Infrastructure is often extremely complex, and the monthly cost of repairing and repairing it is stagnant - and unacceptably high. While DevOps Vicert works to transform one of our clients' digital health solutions, they move to the cloud, resulting in significant improvements in time and durability as well as a significant reduction in care costs. The clients increased their monthly service time to 99.9% with experiencing a strong downtime in the application. By deployment on the cloud and using special serverless solutions (like GCP, AWS Lambda, Azure functions, open whisk, etc.), improved security and faster upgrades for both - results in savings, easier and faster deployment, higher reliability, as well as service expiration.

SERVERLESS ARCHITECTUREFOR HEALTHCARE MANAGEMENT SYSTEM

The serverless technologies rely on the most advanced services to provide the automatization, flexibility, and personalization most needed in the medical field. It can be adopted successfully in several different organizations - for product companies, from startups to businesses, and from payers to suppliers. Beneath the lid, serverless technology relies on portable servers but is managed by cloud vendors like IBM, Amazon, Google, Microsoft, etc., and not the company itself. It means that the procedure for the provision and storage of infrastructure remains separate from internal procedures. This separated concern is one of the principal reasons many fast-developing health tech solutions use serverless technologies. If they rely on the serverless model, the corporation can build and deploy applications without having to manage downtime and compromised infrastructure and related security. It is particularly significant as health care organizations expand applications to wider areas as well as most difficult correlations. Therefore, speed, cost savings, repetition, and scaling are unfavorable to success. In contrast to the traditional cloud framework, where serverless technology is used, the site might be managed by several services that are based on usage as well as demand. In addition, authentication and authorization can be done using AWS's Identity and Access Management (IAM) Services, which can be used. The results are lower cost of easier maintenance, increased expansion, and building a portfolio. Various components of serverless healthcare management system are shown in Figure 4.

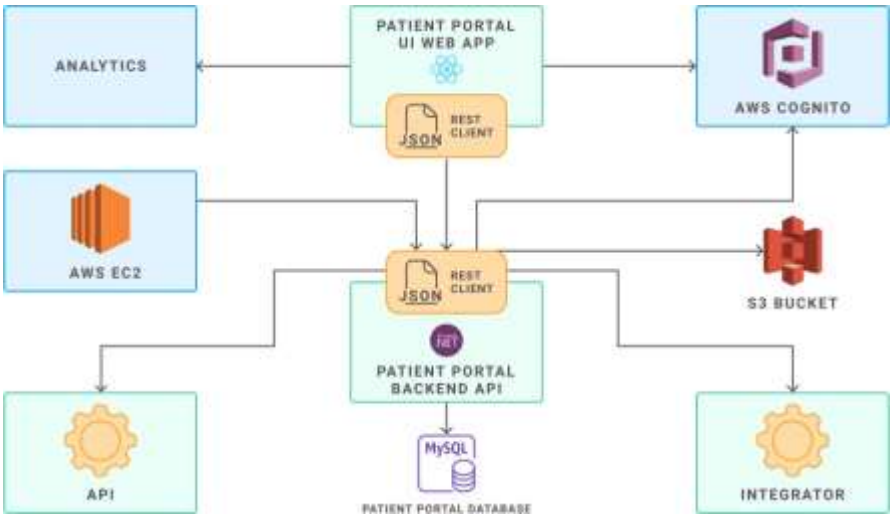

Figure4. Serverless architecture for healthcare management system

VARIOUS COMPONENTS FOR SERVERLESS FRAMEWORK FOR HEALTHCARE SYSTEM

Cloud platforms enable modern businesses to embrace agility and innovate. They allow businesses to experiment; quickly try out new ideas; and scale rapidly when an idea takes off. Just as importantly, their usage-based payment model makes it easy to decommission the ones that don't work. Try new things, see what works, and fail fast:

the mantra of agile businesses. In the past, compliance and privacy issues slowed innovation in the healthcare industry. That is changing. Offerings from all of the large cloud vendors now support and thus allowing companies to quickly assemble secure solutions from off-the-shelf components. It is easier than ever to build secure, compliant cloud applications that protect sensitive personal health information. Security and compliance are built into all of the major cloud platforms, allowing secure, scalable, and manageable products to be built on Microsoft's Azure, Google Cloud Platform (GCP), Amazon Web Services (AWS), and many of the other cloud platforms. Additionally, with interoperability between the platforms increasing, a best-of-breed approach leveraging components from two or more platforms is possible. However, for this article we are focused on our recent experience using AWS for a healthcare startup.

A. Data Storage

Data storage is at the core of most products and AWS offers quite a few ways of persisting data needed by an application. We've found these solutions work well and are Healthcare compliant:

1. **DynamoDB** is a fully managed NoSQL database, with the ability to use key-value stores, document stores like JSON, and graph databases. While it is not HITRUST-certifiable, it is eligible. DynamoDB requires very little overhead and is extremely easy to use, especially with other AWS services.

2. **Relational Database Services** (RDS) is a fully managed service that provides on- demand scalable, highly available relational database engines. RDS makes it easy to provision data stores to provide multi-tenant databases and master data. In addition, RDS has several database engines options, including Aurora DB, MySQL, PostgreSQL, Oracle, and MS SQL Server.

3. **ElastiCache** for Redis is in-memory data storage for scaling web applications. It offers two flavors of in-memory data storage. In just one example, we have used Redis to store notifications while the user is off-line, so they are not lost. When the user returns to the system, the missed notifications are presented from Redis.

B. Serverless Integration

Serverless solutions free organizations and engineers from managing servers and frameworks and simplify compliance.

1. **API Gateway** is used to publish, maintain, and monitor API written on AWS Lambda. With API Gateway, transmitting PHI (Protected Health Information) to and from web servers is secure and encrypted.

2. **Lambda** is a serverless architecture used to run code without provisioning or managing servers. It allows digital solution engineers to quickly ship applications in a variety of languages including Node, Java and more, without provisioning virtual machines and installing runtimes. Lambda also ensures PHI remains secure by using encrypted protocols such as HTTPS and by providing authentication and authorization support to the API Gateway itself.

3. **Amazon MQ** is a managed message queue broker for Apache ActiveMQ. Message queues enable applications to communicate asynchronously. Each system, regardless of platform or programing language, can integrate via a message queue in a broker. Amazon MQ allows easy configuration to provision Apache ActiveMQ on AWS.

C. Infrastructure Deployment and Monitoring

Managing solutions and data is every bit as critical to compliance as having secure technology building blocks. AWS offers a suite of solutions manage the deployment of components and monitor data access and usage.

1. **CloudWatch** is a monitoring and observability service used to set alarms, trigger action, discover insight, and troubleshoot issues. CloudWatch is primarily used to monitor operational logs from various services running within this architecture.

2. **Terraform** is an open-source infrastructure as a code (IaC) tool used to deploy the solution infrastructure safely and repetitively across multiple environments.

3. **Elastic Kubernetes Service** (EKS) is a fully managed Kubernetes service available on AWS that allows users to deploy, manage, and scale containerized applications**.**

HEALTHCARESERVICES WITH FUNCTION AS A SERVICE

Health care system needs efficient distribution of services in fast and efficient manner. Health services include emergency service, diagnosis, analysis, preventive, home care etc. Each of these services should be patient centered and high quality. All the stakeholders include healthcare workers, hospitals; medical professional requires several serverless services at many scales based on their needs. Each serverless service may be provided as function as a service framework. Some of them are described as below:

1. **Appointment Booking Service**

   Serverless framework can be used to develop web application through which services can be provided for booking the appointment. Healthcare service providers require application which can be deployed and executed in scalable and cost- effective manner. AWS APIs gateway can be used to create and publish the services for any point of scale which should able to handle hundreds to thousands of API calls. It also provides another service like traffic management, access control, and authorization to provide security to healthcare data. The core business logic for booking service application can be deployed in *AWS Lambda* which can be processed through either container or the Virtual machine.

2. **Remote Monitoring Service**

One way that COVID-19 has impacted the healthcare sector was through an increased requirement for remote patient monitoring. In this use case, smart health monitoring devices became a necessary part of patient care, albeit that these devices had to deal with the challenge of ingesting data at scale and processing that information in real-time to deliver urgent care to those infected. In the visual below, you can see the reference architecture for a remote patient monitoring system on AWS, which (again) provides serverless services and HIPPA Compliance.

3. Security Services

Privacy is one of the major concerns in healthcare sector especially when the data is exchanged between the stakeholders. Security services may be provided by the container in serverless environment which allows the software to be patched. The server without patch at the backend may leads to high security risk especially in healthcare services.

4. Ease of Access and Availability

Serverless is observed to be most promising solution in term of ease of access and availability of healthcare information. The Availability issues at any point of time may be challenging when the information is located at single server. This can be overcome by replicating multiple instances at several servers. However, it might leads to cost inefficient. In order to provide the cost effective, reliability and fault tolerant access, serverless architecture provide GB-Second pricing model. Serverless storage service keeps the copy of instances at multiple data centers, which solve the problem of common source failure. DynamoDB database automatically backup and restore the data which makes available at any point of time. Serverless framework also provides auto-scaling techniques which allow the clients to keep any amount of data at an ease.

5. Virtual Reality

VR has become a billion-dollar business. It is time that the healthcare industry jumps in on that too. Virtual reality helps patients deal with pain and suffering. VR can also be used to give a tour of the medical facilities. VR is a highly engaging technology and healthcare can benefit from it. So many people get anxious and worried about visiting a hospital, so giving them a tour of the procedure through VR can be really helpful.

6. Wearable

Wearable technology has come a long way. It’s from wearing for fun to wearing for health tracking. People have started to take care of themselves seriously. Almost everyone has smart watches to track their physical activity. It also keeps a track of heart rate. Apple released ‘Movement Disorder API’ which can be helpful in getting insights into Parkinson’s disease.

7. Voice Search

Voice Search has been helping patients with booking appointments and

get rides for the hospital. Voice search works as a personal assistant for the patients. Young patients might not require this technology but it works like a miracle for the elderly. They can rely on it entirely. Nuance Communications is a US-based company that provides automated telephone directory services, speech recognition, and telephone call steering systems. So these are some of the technologies that are taking over the healthcare industry and helping out people. With the use of technology, every patient can get personalized attention. Technology has improved healthcare so far, in the future we will see more innovations in healthcare.

HEALTHCARE SYSTEM

Traditionally web applications were deployed on dedicated. For executing a standard web application users' request, it could be forwarded to the backend (server) wherein the application is deployed, after processing the request, the server might return a response to the users or clients. Moreover, it could be a database system executing upon one or many data centers, servers, containers, etc. with handling the requirement of storage of utility. For enhancing traffic load and redundancy, load balancing can be utilized as a component of the framework. The role of a load balancer would be discovering those servers, in which the application is deployed, dropped, and redirected the service requests a long way off from it; along with it is also able to allocate service requests easily among numerous servers also.

WHAT IS THE NEED OF SERVERLESS IN WEB-BASED APPLICATION?

The Serverless paradigm eliminates the requirements for deploying applications on particular servers, also removes computer resource managers away from the engineers, and delegates the responsibility of external computer service vendors. Despite the term "serverless", the whole concept is still used in a group of servers, which is excluded. Since the functions are assigned to cloud vendors, it is not required to configure and manage servers. The serverless paradigm can be considered as the combination of BaaS (Backend as a service) and FaaS (Function as a service). Backend as a service describes the applications which integrate services along with cloud-based applications, controlling the backend context. With Function as a service, well known as serverless functions, it is the responsibility of the developers to focus only on their business code. Serverless computing saves a lot because the vendors only charge for the duration it takes to complete each request. It can be concluded if a serverless application receive any requests, it will not cost for ideal capacity - a primary advantage to the standard cloud. In a serverless environment cloud vendors may take full advantage of their server configuration to have minimal downtime, they can transfer those cost savings to their users. In serverless computing, developers are permitted to develop, deploy and also execute the business logic without worrying about complicated infrastructure management procedures. It simplifies those developers are only responsible for developing business code that will be used in areas of health care at a faster rate, bringing technical advancement quickly.

ARCHITECTURE FOR SERVERLESS WEB BASED APPLICATION

The features of AWS Cognito can be leveraged to protect the web application. It provides authentication and security services to new as well as existing clients. It can be customized to scaling up the number of concurrent users. The customized interface at the edge of web application can be provided by CloudFront. CloudFront component of AWS service can be used as content delivery networks which process and deliver the content at various access points across the globe. It allows the user to access the content in less response time by storing the content near to the vicinity of the user even if the application hosted somewhere far away. Amazon storage service S3 can be used at the back end to store unstructured file such as image analysis, patient information in every interval etc. Serverless framework can also be used to perform all kinds of linear and non-linear data analysis which is crucial for healthcare sector. The architecture for web-based healthcare application which leveraged the feature of serverless framework is presented in Figure. 5.

The frontend part of the web application is managed by a Java script tool known as VueJs. It is found to be more efficient as compared to other tools as it allows the application to be processed in modular fashion. It breaks the functionality into simple and reusable modules. The web application allows having static as well as dynamic domain. The construction of static domain is taken care by NuxtJs which perform over VueJs tools. It is integrated with Amazon S3 where static web application be hosted at a very low cost. However, it may depend on the size of the application. The beauty of AWS S3 is that it allows the application to be dynamic where various updates can be carried out over the time. Serverless architecture provides various APIs which can be used to download healthcare information and process them using Lambda function. The serverless functions are usually called through REST API. Apart from the AWS S3, Dynamo DB is also integrated at the backend to handle large amount of structured data which may be obtained in the form of Relational table or the XML format. It is same as the serverless NOSQL database.

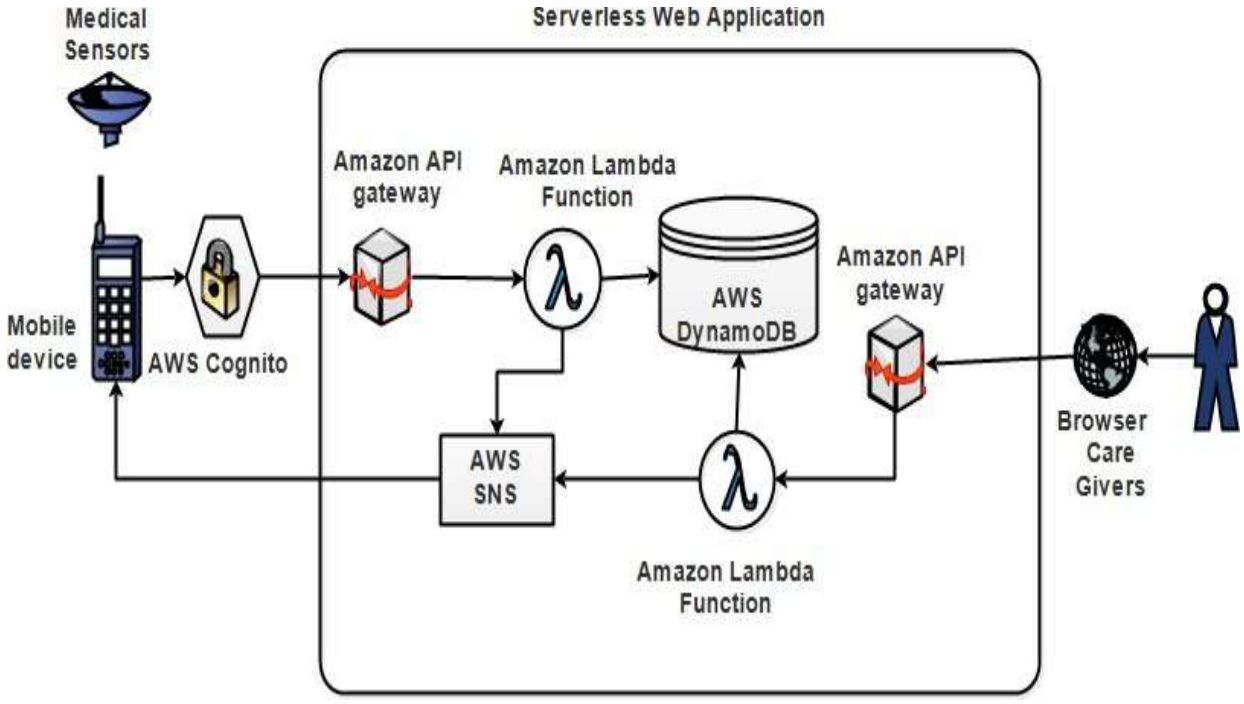

Figure 5. Serverless framework for web-based application

The overall processing of the web application is carried out by the Lambda operations which are the major part of Function as a service or the serverless functions. Any request from the client needs to pass through API gateways which is used to trigger the event. Whenever an event is triggered, the corresponding function is activated and executed to perform the action. The updates in static file is sent to DynamoDB where the updated related to graph analysis is sent to AWS S3 storage. The Lambda function in this architecture may related to perform sorting, searching, analyzing the healthcare information stored in either MySQL Database or DynamoDB or AWS S3. It also allows the health workers to upload the patient data and stored them at the backend and after performing the analysis the resultant graphs can be shown to authentic people using AWS Cognito. It can be shown on application analysis page for better interaction. The processing steps in healthcare web application are shown in Figure 6.

CASE STUDIES OF SERVERLESS APPLICATION FOR HEALTHCARE SYSTEM

Case Study I: Fast Healthcare Interoperability Resources (FHIR)

Fast Healthcare interoperability resources (FHIR) is one of the elegant digitized solution for managing and exchanging healthcare related information between various parties in secure manner. It ensures the privacy and ease of accessibility to the legitimate stake holders of healthcare systems. One of the serverless vendor i.e., AWS provides serverless platform for integrating API for FHIR interfaces. It allows the developers to access number of resource types and operations. It also provides various architecture patterns to develop integration of other solution to the existing healthcare solution. Serverless framework for FHIR allow the users (healthcare providers, vendors, developers etc.) to access patient and healthcare data from any kind of interfaces such as various browser, mobile device, laptop from any corner of the world. This solution also provides customized AWS APIS to tune the parameters based on the requirements of the organization. The prototypes of the solution need to be deployed in the serverless environment to access the storage services like MySQL DB S3, Dynamo DM etc. through FHIR API. The healthcare services can be obtained through several Lambda functions which can be invoked through AWS APIs. The processing steps along with various components for FHIT are presented in Figure 6.

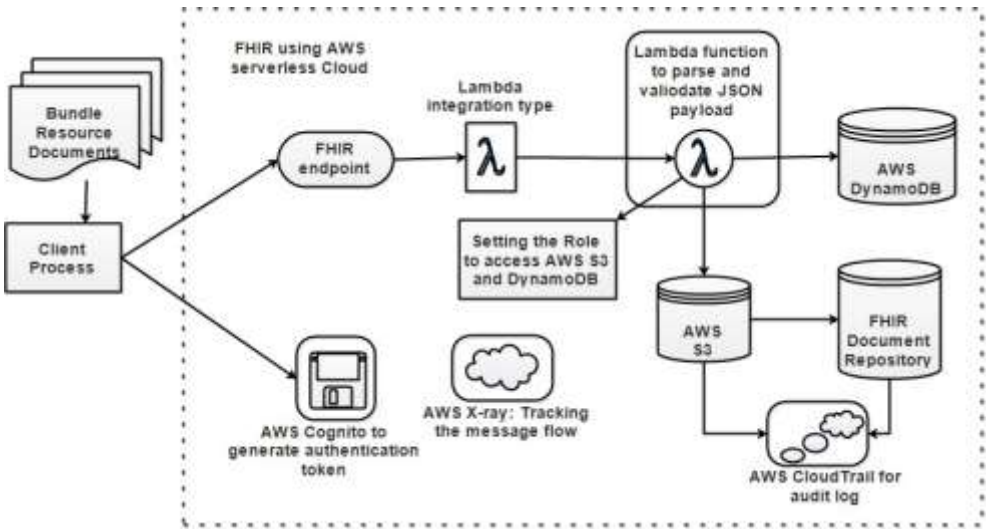

Figure: 6. Serverless Configuration for FHIR Messaging Interface

The serverless pattern deployed by Amazon's Cloud Formation tools can be used to provide services for FHIR request through APIS. The following components are provided to serve different kind of services as follows:

1. Amazon Cognito pools are used to provide authentication by verifying the user's identity and the group it belongs to.
2. Amazon APIs are provided to routing the request towards the Lambda function which can be invoked by the event a trigger.
3. Two vary specified AWS Lambda functions are used in the serverless solution for FHIR. One of them is used to process the FHIR request by routing the path towards the storage services like MySQL DB / Amazon S3 to store unstructured data, DynamoDB services to create, delete, and update operation on healthcare data or to OpenSearch Service to indexing the search process for faster access by the stack holders. The other Lambda function is used to read the updates and make changes in the indexing component.
4. The MySQL DB / DynamoDB storage unit is used to store all the unstructured healthcare data. Whenever any updates are carried out the MySQL DB / DynamoDB, the same is reflected back to the OpenSearch service. Sometimes Amazon's S3 is used to store binary data from healthcare resource such as X-ray or any ECG graphs etc.
5. The indexing and the search process for the incoming request from FHIR APIs are provided by Amazon OpenSearch service.
6. Various Key management services are used to encrypt all the storage components like MySQL / S3, DynamoDB or the OpenSearch service.
7. The CloudWatch service from the Amazon is used to log all API requests from the stakeholders of healthcare system.

Case Study II: IoT Based Remote Monitoring System Using Serverless Architecture

A number of healthcare applications have been integrated with IoT which leveraged the features of serverless frameworks. In traditional hospitals (and hospice

situations), patients come into the hospital or doctor's office when they are not feeling well or, in a worst -case scenario, they need urgent care and are brought directly into Emergency. Digital wearable devices such as smart watches, digital blood pressure and weight scales have made it possible for patients to self-diagnose problems long before they become urgent cases that require emergency care. However, self-diagnosis can be dangerous, and it is always preferable to have doctors view remote sensor data and perform diagnosis prior to clinic or hospital admittance. The challenge is to make such data available to doctors in real -time. The patients will typically be engaged in their daily tasks in a non-hospital environment (either at home/work or in a senior facility, for example). The future of cardiovascular care is envisioned in the following way by experts in the delivery of such care:

1. Incorporating remote monitoring for cardiovascular patients, using devices those record vital signs, heart rhythms like atrial fibrillation, coronary artery disease detecting heart blockages, and congestive heart failure symptoms, without being in the office or hospital.
2. Enabled by IoT and Cloud Technology, these remote physiologic monitors in conjunction with best practices at the clinic, have the potential to reduce patient hospitalizations and improve care.

This case study consists of number of cloud-based IoT device which has been integrated on AWS, which includes device transmitter code that simulates heart rates at the patient location, and sends the data to the AWS Cloud, where the data is stored in a NoSQL database, anomalies are detected in real-time via Streams processing, and results can be displayed in a dashboard at the doctor's office. Archival for offline analytics is also made available. The integration of IoT and serverless framework for Remote health monitoring system is presented in Figure 7.

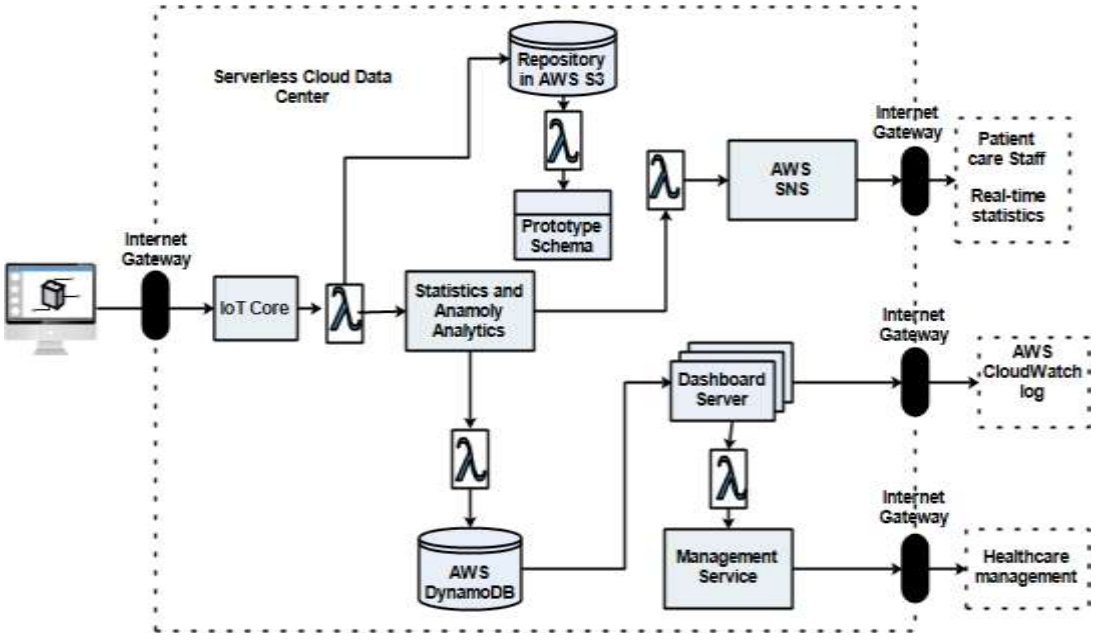

Fig7. Integration of IoT and Serverless in Remote Health Monitoring System

Components for Remote Monitoring System

The solution comprises the following:

- A front-end simulator program creates data by emulating a 'sensor'
- The sensor information will comprise the device details (unique identifier for the patient and device) and heart rate readings
- This data will feed into the AWS IoT Core at predefined intervals
- Based on the rules defined, data will be sent downstream to trigger various real- time events such as email notifications where applicable
- Downstream data will be partitioned and sorted into a NoSQL database table, which will be used to enable rapid storage and retrieval of time-series information, and to allow for critical decision-making in near-real time. Goal is to pay attention to scalability, availability, and reliability
- Based on the defined rules, real-time data will be routed to Kinesis Streams, where Kinesis Stream Application will sort data streams based on high and low heart rate and route to a separate Lambda function for real-time SNS and email notifications. Routed real-time sorted streams will be stored in a separate table in the NoSQL database.
- Dashboards will display the patient wise readings and will be designed to be made available to administrators with authorized access, for insights and action.
- Downstream data will also be sent via an alternative path for archival and offline analytics with secure access as needed by medical personnel for research and regulatory purposes.

CONCLUSION

Serverless is has been emerging as a promising solution for maintaining digitized environment in healthcare sector where several healthcare services can be provided in low cost and faster response time. Serverless provider may have various types of services based on the client requirement such as on-demand, on-reserve or the instance services. Healthcare sector popularly use on-demand service model of serverless which can handle the peak load at any point of time. It may be noted that different application in healthcare system like ECG, EEG, E-blood analysis etc. requires different pricing model. In this chapter, the serverless architecture for healthcare system is presented. Two case studies related to healthcare system such as Remote monitoring system and Fast Healthcare Interoperability Resources (FHIR) interface are presented which abundantly use the services from various serverless vendors like Google, Amazon, IBM, and Microsoft etc. In the present day, serverless architecture is found to be an efficient framework to provide various healthcare services. However, it poses a number of challenges in integrating IoT with serverless in healthcare system. As the serverless framework rely on number of independent stateless functions, they failed to communicate each other by exchanging data which is one of the major concern in healthcare management system. Another challenge in serverless healthcare management system lies in maintaining the

tradeoff between cost and deadline of various healthcare applications. Although it poses various challenges, it has been emerging as potential tool for handling various services in healthcare system, the identified research challenges may be considered as one of the research direction in the era of serverless framework in healthcare system.