Venkata Raman Gali

gali.ramana@gmail.com

­­­­

Abstract

AWS LAMP Stack ProServe

Architectural Document­­­­

Table of Contents

[Context 2](#_Toc71082853)

[Objective 2](#_Toc71082854)

[Goals 2](#_Toc71082855)

[Budget 2](#_Toc71082856)

[Requirements 2](#_Toc71082857)

[Assumptions 2](#_Toc71082858)

[Current State (As Is) 3](#_Toc71082859)

[Feature State (To Be) 3](#_Toc71082860)

[Problems in Traditional LAMP Stack Cloud Architectures 3](#_Toc71082861)

[Why LAMP Stack Serverless Technology 3](#_Toc71082862)

[Architecture Diagram 4](#_Toc71082863)

[Timelines 4](#_Toc71082864)

[Environments & CICD 5](#_Toc71082865)

[Appendix 5](#_Toc71082866)

[Document References 5](#_Toc71082867)

# Context

Social media start-up company which uses LAMP stack for main web & provide REST API for mobile services in the early stages of their operations, recommended by their investor to adopt AWS infrastructure.

# Objective

Provide manageable, secure, scalable, high performance, efficient, elastic, highly available,

fault tolerant and recoverable cloud architecture that allows the start-up to organically grow. The architecture should specifically address the requirements/concerns described.

# Goals

AWS Infrastructure technical architecture proposal succinctly presents an analysis of the start-ups requirements w.r.t to significant, rapid, un-quantified global growth proposal to adopt/migrate architecture to AWS.

# Budget

Allocated budget for the proposal is $250K for next XX years

# Requirements

1. Capacity to meet on demand, make effective of Seed funding ensuring required
2. Scalable Web application and the API services layer to be met on demand.
3. Effective distribution of load & throttle traffic to the API services layer.
4. Cost effective database & DAL with read & write throughput
5. High performance globally distributed large web application target large userbase
6. Secure access to mobile app users to API & sync user preferences across devices
7. Self-healing infrastructure to recover from failures
8. Capture, Analyse and Store security and Archive (6 months) for sentimental analysis data
9. Encryption at rest and In transit for entire application architecture
10. Secure and Manageable access to environment
11. Rapid support of optimise, Automated source code, build, test and deployment
12. Easily manageable and replicated multiple environments

# Assumptions

Its been almost 5+ year since AWS IT cloud infrastructure services was available. Here are the below considerations.

* Modernized AWS Serverless architectures will be considered
* Traditional cloud hosting architectures will be omitted or considered least priority
* Highly Scalable, reliable, cost effective and resilient solutions are considered
* AWS Managed services for entire architectures for rapid deployments
* New generation of LAMP stacks and upgrades for feature growth

# Current State (As Is)

Currently the architecture for their social media applications are using LAMP(Linux, Apache, MySQL & PHP) stack for the web application and providing a RESTful API for mobile services, written in NodeJS. Currently hosted in <<xyz>> based Virtual Private Service data centre

# Feature State (To Be)

Existing Lamp stack application will be Hosted in below new generation AWS Serverless LAMP stack Technology i.e.

* Lambda
* API Gateway
* MySQL
* PHP

with given [Requirements](#_Requirements)

# Problems in Traditional LAMP Stack Cloud Architectures

Here are the below problems with trad LAMP stack application grows

1. Required more capacity i.e. scale horizontally behind Load Balancer which leads to new Nodes/Instances
2. Network or Administration
3. Storage capacity
4. Backup & restore
5. Asset management
6. Data Synchronization issues between the instances if no. of Instances grows

# Why LAMP Stack Serverless Technology

Automatic Scaling & No Infrastructure Provisioning

Pay for value

Highly available and secure

Decoupled microservices

Compile own custom runtime or use open source layer

Developer productivity to move Dev to PRD easily and rapidly

# Architecture Solution

Proposed solution is AWS Serverless LAMP stack i.e. Lambda, API Gateway, MySQL and PHP

Entire application layers will be hosted in AWS Infra structure

Public facing based domain, its hosted zone configured in Route53

Edge location based cloud front distributions along with its Caching, Origins and behaviours

AppSync GraphQL APIs to hande the heavy lifting of securely connecting to data sources like AWS Aurora, Lambda etc

API Gateway for unified, secured, throttle & authenticate API requests

Open source run time API, layers, PHP bootstrap, PHP extensions, libraries and dependencies for Lambda code

NodeJS Lambda functions for mobile based microservice API calls

Managed, KMS encryption enabled Multi AZ Amazon Aurora MySQL DB with RDS Proxy for backend facing Aws Appsync

AWS Migration service to archive the data in S3 glacier

Amazon Comprehend for Sentiment Analysis data operations

CloudWatch logging for entire Infrastructure

AWS Amplify for configuration, CICD and deployments

# Architecture Diagram

Chart, box and whisker chart

Description automatically generated

# Timelines

|  |  |  |
| --- | --- | --- |
| Phase | Deliverables | Duration |
| Phase 1 | Create Resources, Network and IAM using CloudFormation | 15 days |
| Phase 2 | Configure DB and PHP Modules, Deploy | 1 Month |
| Phase 3 | Data Migration | 7 days |
|  |  |  |

# Environments & CICD

AWS Amplify to Staging & Production environments, Authentication, hosting CICD process and mange deployments

# Appendix

Document References:

<https://aws.amazon.com/blogs>

<https://docs.aws.amazon.com/index.html?nc2=h_ql_doc>