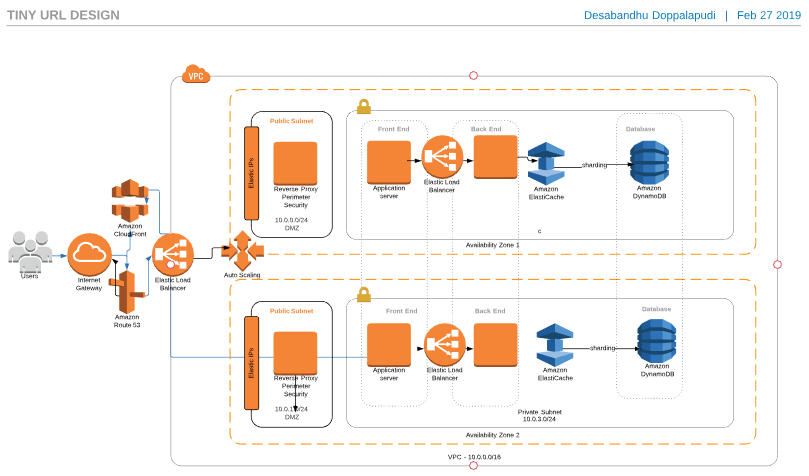
Tiny URL Design:

AWS EC2(T3 micro), Auto Scaling, Load Balancer, Multi AZ, Elastic Cache (redis),Cloud Front, Route 53 &DB First design came into my mind and I replied you with this.

1. Launch EC2 instance T3 Micro Instance
2. Configure Load Balancer ( internet facing load balancer) with HTTP protocol port 80, with Multi AZ in VPC; Security Group TCP open to all ( i.e port 80 source anywhere 0.0.0.0/0); configure routing and health checks
3. Create Auto scaling launch configuration, Auto scaling groups, Health checks we can use CPU utilization with 75% threshold metric for Auto scaling.
4. Use Cloud front for CDN, (edge location) to be fast for users across all continents.
5. use elastic cache for user requests.
6. use Dynamo db as our database
7. <https://s3-us-west-2.amazonaws.com/homework-design-tinyurl/Tiny+URL+Design_Home+work_Design.JPG>



# After browsing to know about second problem (which I already informed you that I don’t know) I come across this below article. Which has Cloudformation template with Node.js

# Serverless private URL shortener based on Amazon S3, AWS Lambda, Amazon CloudFront and API Gateway.

<https://aws.amazon.com/blogs/compute/build-a-serverless-private-url-shortener/>

How to build a Serverless URL shortener using AWS Lambda and S3

<https://medium.freecodecamp.org/how-to-build-a-serverless-url-shortener-using-aws-lambda-and-s3-4fbdf70cbf5c>

<https://github.com/danielireson/serverless-url-shortener>

<https://theburningmonk.com/2017/04/aws-lambda-build-yourself-a-url-shortener-in-2-hours/>

<https://github.com/pointtonull/serverless_shortener/commit/aeab8deea761359b462a807d4ac1fd0ed8fed60a>

Using Chalice framework for building Url shortener using Python.

<https://chalice.readthedocs.io/en/latest/index.html>

<https://github.com/aws/chalice>

