**Create Highly Available Architecture by Distributing Incoming Traffic among Healthy Service Instances in Cloud Services or Virtual Machines in a Load Balanced Set with the Help of Command-Line Interface**

To create high available architecture by distributing incoming traffic among healthy service instances in cloud services or virtual machines in a load-balanced set with the help of a command-line interface

**Description:**

The Rand Enterprises Corporation wants to deploy a web application in a highly available environment so that only the healthy instances will be serving the traffic so end users will not be facing any downtime. They have decided to work on an Azure public load balancer to implement the functionality.

The operations team at Rand decides to define the entire architecture using the load balancer and its backend pool, once that’s in place they intend to create the frontend IP and health probe along with virtual machines housing their application.

Rand Enterprises works extensively on delivering highly available web applications for their users in a secure way by avoiding directly exposing the virtual machines hosting the applications to the public internet. The communication from the application in the VM to the end-user must take place via the Load Balancer.

The expectation of the operation team is to create a reusable method that can be used for automation if in the future we need to deploy the same kind of infrastructure. So, rather than deploying resources in the Azure portal, they should leverage the command-line interface to deploy the resources so that in the future these commands can be used

As a security measure, you need to ensure that only the health instances of the virtual machine will be serving the traffic.

**Tools required:** Azure account with administrator access

**Prerequisites:**None

**Expected Deliverables:**

* Identify Virtual machines and Networking
* Configure the load balancer
* Extend the load balancer with backend pool and frontend IP
* Define the Health probe
* Extend the security with the bastion Hosts