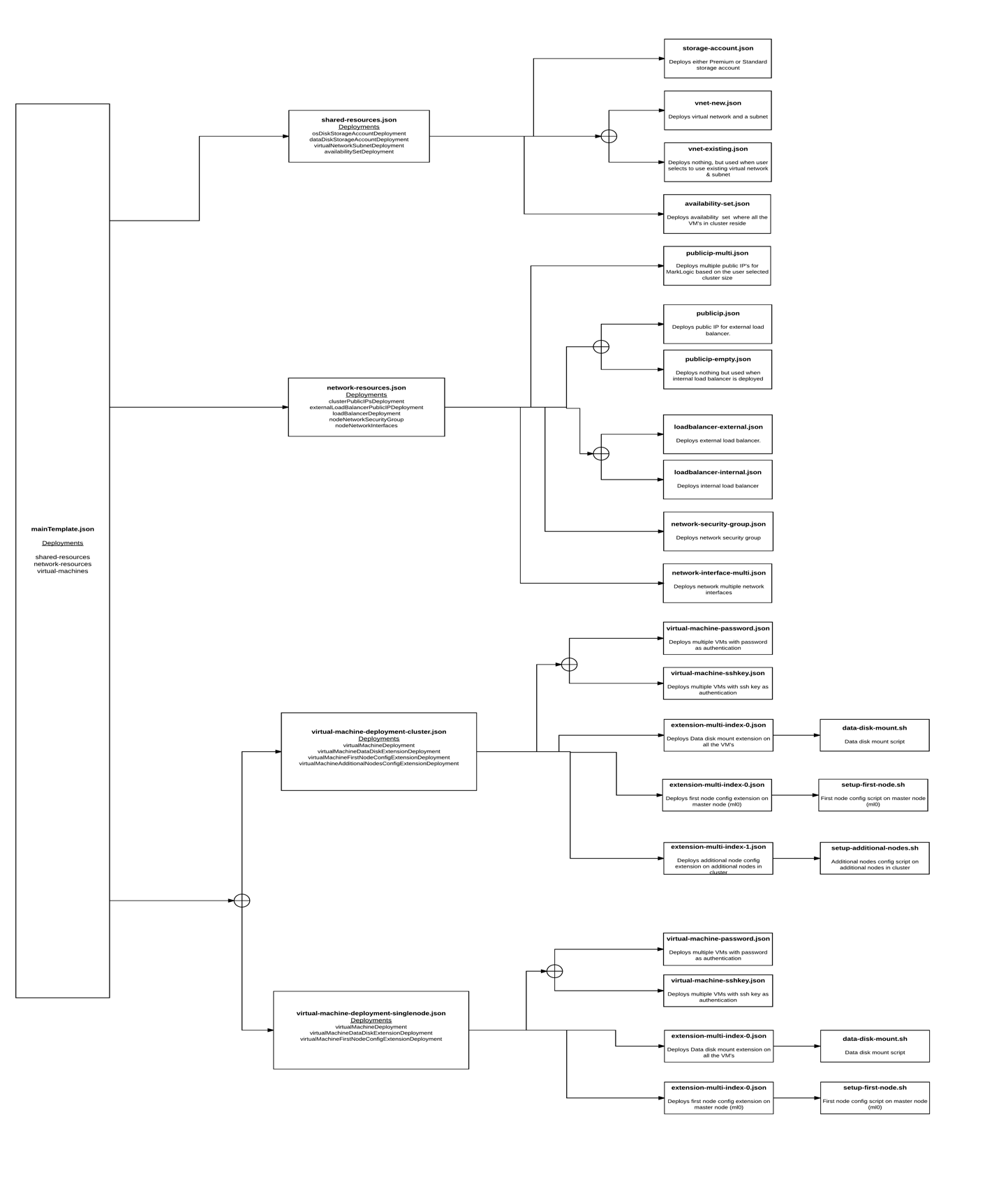
MarkLogic

Template Architecture (Azure Market Place)



Template deployment starts with mainTemplate.json, which has all the parameters for user input and variables that are constructed to use in the subsequent nested templates. mainTemplate.json has three deployments in the resources section (All the objects referred are from main templates variables section):

1. **Shared Resources (shared-resources):** Deploys shared resources such as Storage accounts, Availability set, Virtual Network and Subnet. ‘shared-resources.json’ is deployed and which has deployment (resources) sections:
   * **OS Disk Storage Account (osDiskStorageAccountDeployment):** Deploys standard storage account resource for MarkLogic VM’s OS disks through “storage-account.json”. Input to “stoarge-account.json” is “osDiskStoarge” object under “storageSettings” object.
   * **Data Disk Storage Account (dataDiskStorageAccountDeployment):** Deploys either premium or standard (based on user selection) storage account resource for MarkLogic VM’s data disks through “storage-account.json”. Input to “stoarge-account.json” is “dataDiskStoarge” object under “storageSettings” object.
   * **Virtual Network and Subnet (virtualNetworkSubnetDeployment):** Deploys either a new virtual network & subnet (one subnet) or use an existing one in that resource group (based on user selection). To create new “vnet-new.json” and to use existing “vnet-existing.json” are deployed. Input to “vnet-new.json” and “vnet-existing.json” is “networkSettings” object.
   * **Availability Set (availabilitySetDeployment):** Deploys availability set where all the MarkLogic cluster VM’s reside. “availability-set.json” is used to create availability set. By default, availability set deploys with fault domain as “3” update domain as “20”. Input to “availability-set.json” is “networkSettings” object.
2. **Network Resources (network-resources):** Deploys network resources such as public IP’s, external or internal load balancer, network security group and network interfaces.
   * **Public IP’s (clusterPublicIPsDeployment):** Deploys public IPs for each node in the cluster. “publicip-multi.json” file is used to create public IPs. Input to “publicip-multi.json” is “nodeNetworkSettings” object under “networkSettings” object.
   * **External Load Balancer Public IP (externalLoadBalancerPublicIPDeployment):** Deploys public IP for external load balancer. Based on the user selection it will identify the template Uri and deploys the public IPs. If the user selects “external” load balancer “publicip.json” template is deployed. If the user selects “internal” load balancer “publicip-empty.json” template is deployed. Input to “publicip.json” and “publicip-empty.json” is “loadBalancerNetworkSettings” object under “networkSettings” object.
   * **Load Balancer (loadBalancerDeployment):** Deploy the load balancer for the cluster, based on the user selection template deploys either “internal” or “external” load balancer. If the user selects “internal” or “external” load balancer templates “loadbalancer-internal.json” or “loadbalancer-external.json” will get deployed respectively. External load balancer has public IP and internal load balancer has static private IP address based on the user input with respective to the subnet the load balancer is present. As part of deployment there will not be any load balancing rule, which user can configure after the deployment. But the health probe is configured at port 7997. NAT rules for Admin UI at 8001 and Query UI port 8000 are configured as part of deployment. User can access the admin UI on node one through load balancer using http://<loadbalancer IP>:2200 and for node two it is http://<loadbalancer IP>:2201 and so on. Similarly, to access Query UI it is http://<loadbalancer IP>:3300 on first node, http://<loadbalancer IP>:3301 on node two etc. Input to “loadbalancer-internal.json” and “loadbalancer-external.json” is

“loadBalancerNetworkSettings” object under “networkSettings” object.

* + **Network Security Group (nodeNetworkSecurityGroup):** Deploy the network security group (NSG) for the cluster. In enables “ssh” at port number 22, “http” at port number 80, “admin” console port number at 8001, “health-check” port number at 7997, “query” console port number at 8000 and “manage” console port number at 8002. It uses “network-security-group.json” template for deploying NSG. Input to “network-security-group.json” is “nodeNetworkSettings” object under “networkSettings” object.
  + **Network Interface (nodeNetworkInterfaces):** Deploy the network Interfaces (NICs) for all the nodes in the cluster. “network-interface-multi.json” template is used to deploy NICs for all the nodes. Input to “network-security-group.json” is “nodeNetworkSettings” object under “networkSettings” object.

1. **Virtual Machines (virtual-machines):** Deploys virtual machines and virtual machine extensions. Virtual Machines deployment consists of two types based on the user selection

“Single node” deployment and “Cluster/ Multi node” deployment.

* **Single node:** Deploys single node when user selects node count as ‘1’. “virtual-machine-deployment-singlenode.json” template is used for the deployment.It consists of following deployments:
  + **Virtual Machine (virtualMachineDeployment):** Deploys virtual machine based on authentication type. If user selects the authentication type as “password” or “sshkey” then templates “virtual-machine-password.json” or “virtual-machine-sshkey.json” is deployed respectively. Either template deploys virtual machine(s) with “MarkLogic 8” SKU and 1GB datadisk. Input object is “virtualMachineSettings”.
  + **Data Disk Mount Extension (virtualMachineDataDiskExtensionDeployment):** Deploys Linux custom script extension using “extension-multi-index-0.json” template which will partition the attached data disk and make it available for writing the data into it. This extension uses shell script named “[data-disk-mount.sh](https://github.com/sysgain/MarkLogic/blob/mldevchange/scripts/data-disk-mount.sh)”. This extension runs on all the nodes in the cluster. Input to “extension-multi-index-0.json” is “dataDiskExtension” object under “virtualMachineSettings” object.
  + **First Node Configuration Extension (virtualMachineFirstNodeConfigExtensionDeployment):** Deploys Linux custom script extension using “extension-multi-index-0.json” template which will setup the first node in the cluster. The extension uses “[setup-first-node.sh](https://github.com/sysgain/MarkLogic/blob/mldevchange/scripts/setup-first-node.sh)” shell script. This extension runs on only master node (first node – ml0). Input to “extension-multi-index-0.json” is “firstNodeConfigExtension” object under “virtualMachineSettings” object.
* **Cluster / Multi node:** Deploys cluster/ multi nodes when user selects node count as 3, 4, 5, 7, 9. “virtual-machine-deployment-cluster.json ” template is used for this deployment. It consists of the following deployments:
  + **Virtual Machine (virtualMachineDeployment):** Deploys virtual machine based on authentication type. If user selects the authentication type as “password” or “sshkey” then templates “virtual-machine-password.json” or “virtual-machine-sshkey.json” is deployed respectively. Either template deploys virtual machine(s) with “MarkLogic 8” SKU and 1GB datadisk. Input object is “virtualMachineSettings”.
  + **Data Disk Mount Extension (virtualMachineDataDiskExtensionDeployment):** Deploys virtual machine based on authentication type. If user selects the authentication type as “password” or “sshkey” then templates “virtual-machine-password.json” or “virtual-machine-sshkey.json” is deployed respectively. Either template deploys virtual machine(s) with “MarkLogic 8” SKU and 1GB datadisk. Input object is “virtualMachineSettings”.
* **First Node Configuration Extension (virtualMachineFirstNodeConfigExtensionDeployment):** Deploys Linux custom script extension using “extension-multi-index-0.json” template which will setup the first node in the cluster. The extension uses “[setup-first-node.sh](https://github.com/sysgain/MarkLogic/blob/mldevchange/scripts/setup-first-node.sh)” shell script. This extension runs on only master node (first node – ml0). Input to “extension-multi-index-0.json” is “firstNodeConfigExtension” object under “virtualMachineSettings” object.
* **Additional Nodes Configuration Extension (virtualMachineAdditionalNodesConfigExtensionDeployment):** Deploys Linux Custom Script extension using “[extension-multi-index-1.json](https://github.com/sysgain/MarkLogic/blob/mldevchange/nested/extension-multi-index-1.json)” template which will setup additional nodes in the cluster except the first node.It uses “[setup-additional-nodes.sh](https://github.com/sysgain/MarkLogic/blob/mldevchange/scripts/setup-additional-nodes.sh)” script. Input to “extension-multi-index-1.json” is “additionalNodesConfigExtension” object under “virtualMachineSettings” object.

Finally, when the user deploys the main template either from market place or directly shared-resources will be deployed first followed by network-resources and virtual-machines.