One can bootstrap additional master, and have 2 or more masters.
But, before running **kubeadm** on the other master, you need to first copy the K8s CA cert from kubemaster01. To do this, you have two options:

Option 1: Copy with scp
ssh to the master node and cd to /etc/kubernetes/pki/
$ chown master * (change the ownership of the certs)
$ cd etcd
$ chown master *

SSH to the kubemaster02 and run the below commands.
$ sudo scp root@<kubemaster01-ip-address>:/etc/kubernetes/pki/* /etc/kubernetes/pki

Option 2: Copy paste
Copy the contents of /etc/kubernetes/pki/ca.crt, /etc/kubernetes/pki/ca.key, /etc/kubernetes/pki/sa.key and /etc/kubernetes/pki/sa.pub and create these files manually on kubemaster02

[The next step is to create a Load Balancer](#) that sits in front of your master nodes. How you do this depends on your environment; you could, for example, leverage a cloud provider Load Balancer, or set up your own using NGINX, keepalived, or HAproxy.

For bootstrapping use the config.yaml:
$ cat >config.yaml <<EOF

apiVersion: kubeadm.k8s.io/v1alpha1
kind: MasterConfiguration
api:
  advertiseAddress: <private-ip>
etcd:
  endpoints:
 - https://<your-ectd-ip>:2379
  caFile: /etc/kubernetes/pki/etcd/ca.pem
  certFile: /etc/kubernetes/pki/etcd/client.pem
  keyFile: /etc/kubernetes/pki/etcd/client-key.pem
networking:
  podSubnet: <podCIDR>
apiServerCertSANs:
- <load-balancer-ip>
apiServerExtraArgs:
  apiserver-count: "2"
EOF

Ensure that the following placeholders are replaced:
- your-ectd-ip the IP address your etcd
- private-ip it with the private IPv4 of the master server.
- <podCIDR> with your Pod CIDR
- load-balancer-ip endpoint to connect your masters

then you can run the command:

$ kubeadm init --config=config.yaml

and bootstrap the masters.

But if you really want a HA cluster please follow the documentation's minimal requirements and use 3 nodes for masters. They create these requirements for etcd quorum. On every master node they run the etcd which works very close to masters.