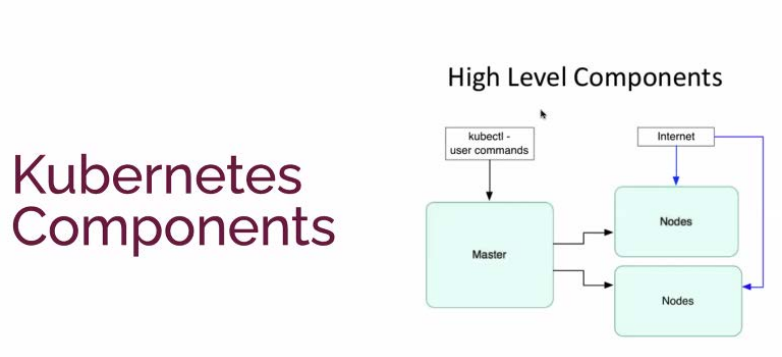
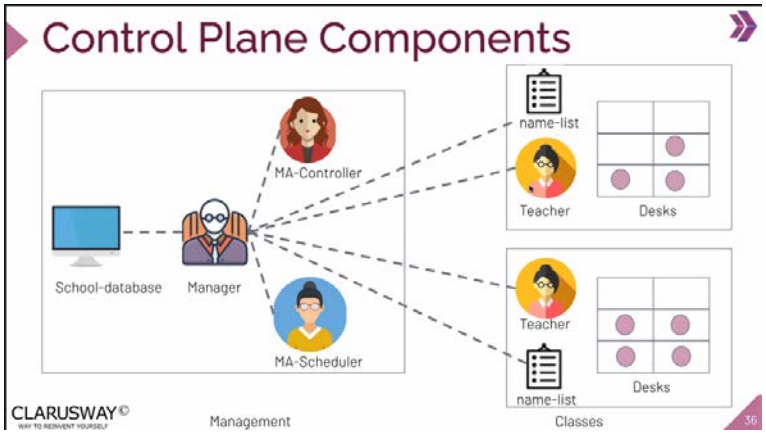
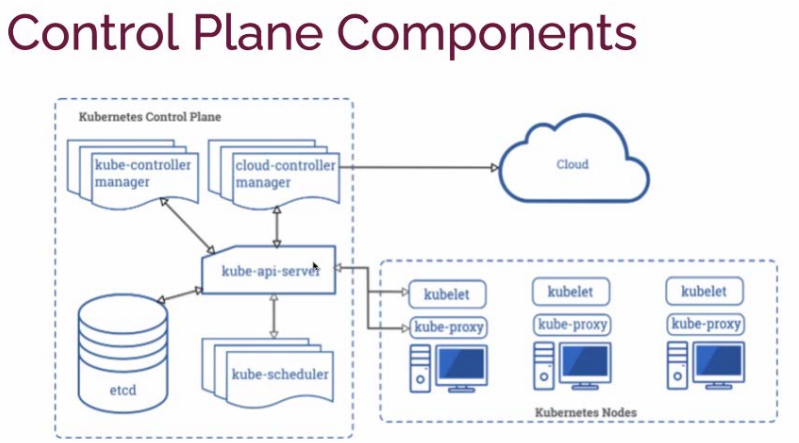
KUBERNETES-1

- **Monolith:**
  - ✓ Tüm servislerinin tek bir kod bloğunda yazılmış olan uygulamalar monolith yapılardır.
  - ✓ Karmaşık bir kod yapısı olduğu için anlaşılması güçtür.
  - ✓ Yeni bir teknolojiye uyum sağlamak çok zor oluyor.
- **Microservices:**
  - ✓ Her bir servisin farklı farklı programlama dillerinden yazılabildiği ve hiçbir servisin diğer servisi etkilemediği modüler yapılardır.
  - ✓ Kod küçük parçalara bölündüğü için anlaması kolaydır.
  - ✓ Herhangi bir hata çıktığında yalnızca hata alınan servise bakılmaktadır.
- **Orchestration:**
  - ✓ Birçok container'ın kubernetes üzerinden yüklenmesi, yönetilmesi ve organize edilmesini sağlar.
  - ✓ Belirlenen kurallara göre şartlara uyum sağlanır.
  - ✓ Rancher, Nomad, dockerSwarm,Mesos, Kubernetes gibi orchestration tool'ları vardır. Piyasada en çok kullanılan tool kubernetes'tir.
  - ✓ Container'larla alakalı tüm yönetimsel durumları içermektedir.
  - ✓ Tüm iş yükleri farklı farklı intances'larda çalışır ve yönetilebilir.
- **Declarative ve Imperative:**
  - ✓ Imperative belirli bir yapıya bağlı kalmadan bir servisin detaylı inşa edilmesi, declarative de ise detaya girmeden basit kuralların belirlenmesi.
  - ✓ Declarative de ne yapılacağına, Imperative de ise nasıl yapılacağına odaklanılıyor.
  - ✓ Piyasada en fazla declarative tercih edilmektedir.
- **Kubernetes:**
  - ✓ Google tarafından 2014 yılında kullanılan hizmetken bağışlanıyor.
  - ✓ 2015 yılında birinci versiyonu çıkmıştır.
  - ✓ K8s şeklinde kısaltması vardır.
  - ✓ ***Bir yazılımın dağıtımını, ölçeklendirmesini ve yönetimini otomatikleştirmek için açık kaynaklı bir kapsayıcı düzenleme sistemidir.***
  - ✓ Açık kaynak kodlu container'ların ocrhestration edilmesini sağlar
  - ✓ Herhangi bir serviste bir eksiklik veya aksaklık olursa anlık olarak ve otomatik olarak o kısma müdahale edilmesini sağlar.
  - ✓ Büyük ölçekli bir iş yükü söz konusuysa kübernetes kaçınılmazdır.
  - ✓ Tüm container'lar otomatik olarak kontrol edilir.
- **Kubernetes Components:**

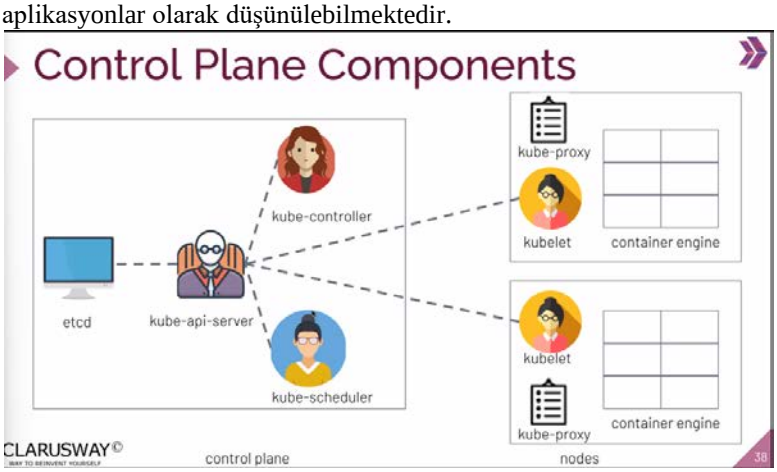


  - ✓ Master ve Nodes adlı iki adet komponent bulunmaktadır.
  - ✓ Master tam merkezde bulunmaktadır.
  - ✓ Node'lar arasında kullanılan ilişki master'dan alınan bilgilerle uygulanmaktadır.
  - ✓ Bir okul gibi düşünülürse okul müdürü ve yardımcıları bir tarafta, sınıflar ve içerisinde ki öğrenciler bir taraftadır.





- ✓ Kubernetes'in yapısı ikiye ayrılabilir, ***kubernetes control plane*** ve ***kubernetes nodes*** 'dur.
- ✓ *Kubernetes control plane* yönetim paneli olarak düşünülürken, *kubernetes nodes* çalışan aplikasyonlar olarak düşünülebilmektedir.



- ✓ **kube-apiserver**: Kübernetesin üzerinden bir işlem yapmak istendiğinde bu modül olmadan işlem yapılamamaktadır.
- ✓ **etcd**: Bir database tipidir, key value store tutulmaktadır, kubernetes bunu kullanmaktadır. Çalışan sistemde bir farklılık varsa tekrar eski haline dönmeyi sağlar.
- ✓ **kube-control-manager**: Tüm kontrol işlemlerinin yapılmasını sağlar. Etcd de tutulan data da fark varsa düzeltilir.
- ✓ **kube-scheduler**: İlgili servisi ilgili container'a planlama işlemi yapmaktadır.
- ✓ **kubelet**: Gelen kurala göre o container da işlemlerin yapılmasını sağlar.
- ✓ **kube-proxy:** Node lar üzerindeki network ağını kuran yapıdır.

- ✓ **Container Runtime Engine:** Containerd (docker), Cri-o, Rkt, Kata ,Virtlet gibi engine'ler vardır. En aktif kullanılan Containerd 'dir.
- ✓ **kubectl:** Bu tool sayesinde kubernetes API'si ile iletişime geçilmektedir.