PROBLEM 7

Create the Blue Deployment
The Deployment will start up a few nginx containers as the application. The Deployment has a name and version label. This is significant as the Service will use these labels to switch to the green version later.

```
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: nginx-1.10
spec:
  replicas: 3
  template:
    metadata:
      labels:
        name: nginx
        version: "1.10"
    spec:
      containers:
        - name: nginx
          image: nginx:1.10
          ports:
            - name: http
              containerPort: 80
```

$ kubectl apply -f kubernetes/blue-deploy.yaml

The service is of *type=LoadBalancer* so it can be accessed via a Network Load Balancer on any cloud platform. It uses the name and version labels specified in the Deployment to select the pods for the service.

```
apiVersion: v1
kind: Service
metadata:
  name: nginx
  labels:
    name: nginx
spec:
  ports:
    - name: http
      port: 80
      targetPort: 80
  selector:
    name: nginx
    version: "1.10"
  type: LoadBalancer
```

Create the Service:

$ kubectl apply -f kubernetes/service.yaml

Test the Blue Deployment

The currently deployed version can be tested in a separate window by polling the server. This will print the current deployed nginx version.

```
$ EXTERNAL_IP=$(kubectl get svc nginx -o jsonpath="{.status.loadBalancer.ingress[*].ip}")
$ while true; do curl -s http://$EXTERNAL_IP/version | grep nginx; sleep 0.5; done
```

Now we are ready to deploy a new version.

Update the application
A new Deployment will be created to update the application and the Service will be updated to point at the new version. This is mostly instantaneous.

Create the Green Deployment
The Green Deployment is cerated by updating to the next version. An entirely new Deployment will be created with different labels. Note that these labels don't match the Service yet and so requests will not be sent to pods in the Deployment.

```
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: nginx-1.11
spec:
  replicas: 3
  template:
    metadata:
      labels:
        name: nginx
        version: "1.11"
    spec:
      containers:
        - name: nginx
          image: nginx:1.11
          ports:
            - name: http
              containerPort: 80
```

You can update the Blue Deployment's file directly or use a tool like sed:

Create the new Deployment:

$ sed 's/1\.10/1.11/' kubernetes/blue-deploy.yaml | kubectl apply -f –

Switch Traffic to the Green Version
We will update the Service to select pods from the Green Deployment. This will cause new requests to be set to the new pods.

You can update the file directly or use a tool like sed:

```
$ sed 's/1\.10/1.11/' kubernetes/service.yaml
apiVersion: v1
kind: Service
metadata:
  name: nginx
  labels:
    name: nginx
spec:
  ports:
    - name: http
      port: 80
      targetPort: 80
  selector:
    name: nginx
    version: "1.11"
  type: LoadBalancer
Update the Service:
```

```
sed 's/1\.10/1.11/' kubernetes/service.yaml | kubectl apply -f -
```