Your DevOps team manages a Kubernetes cluster using Infrastructure as Code (IaC) with

Terraform. You need to allocate a task to scale the application based on increased traffic.

Question:

How would you allocate the task of scaling the application in the Kubernetes cluster using

Terraform and ensure it's implemented efficiently?

Tasks given

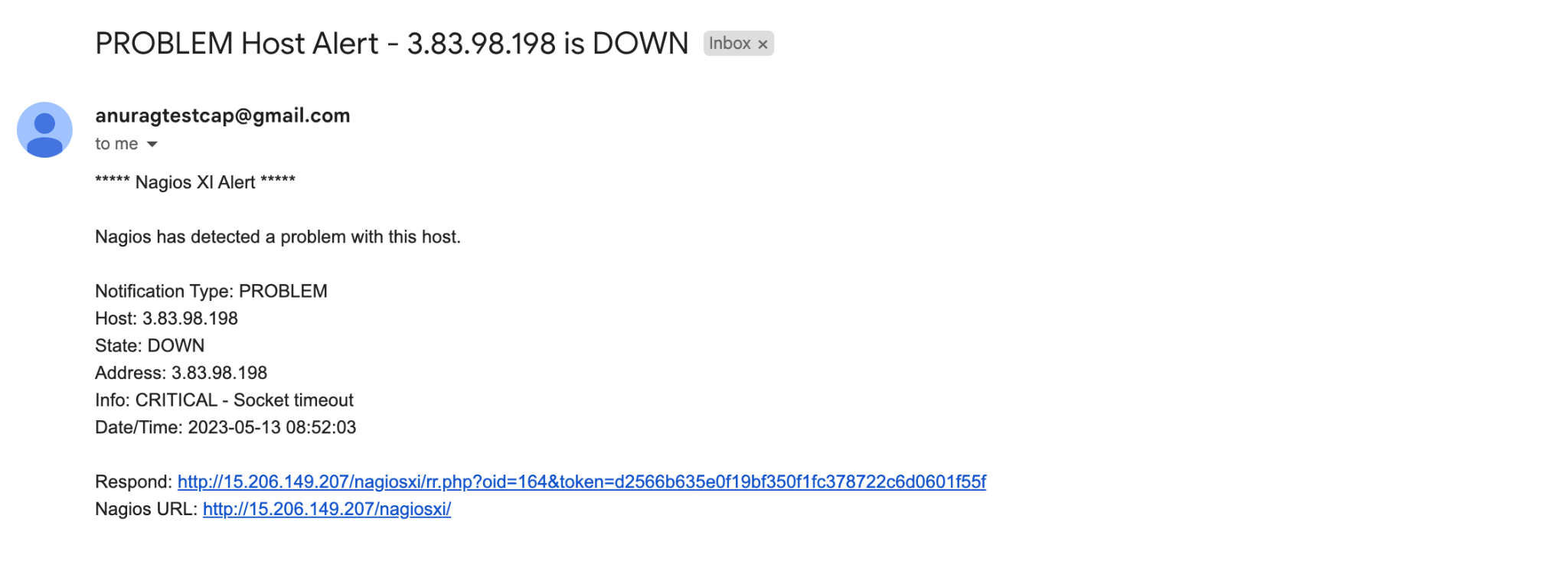
Task 1: Explain how to identify the need for scaling based on traffic metrics or other indicators.

**Answer:** The major reason for scaling the application can be identify by increasing in the number of the traffic of the user

* **Monitoring**: Implement a monitoring system for your Kubernetes cluster. Tools like Prometheus, Grafana, NagiOs. Define key performance indicators (KPIs) such as CPU usage, memory utilization, and network traffic.
* **Set Scaling Metrics**: Establish thresholds for each KPI that specify the way of scaling. For example, if CPU usage exceeds 80% for an extended period, it might trigger a event/emails.
* **Alerting:** Alerting the user by sending an email, SMS, Slack channel update once threshold is been reached

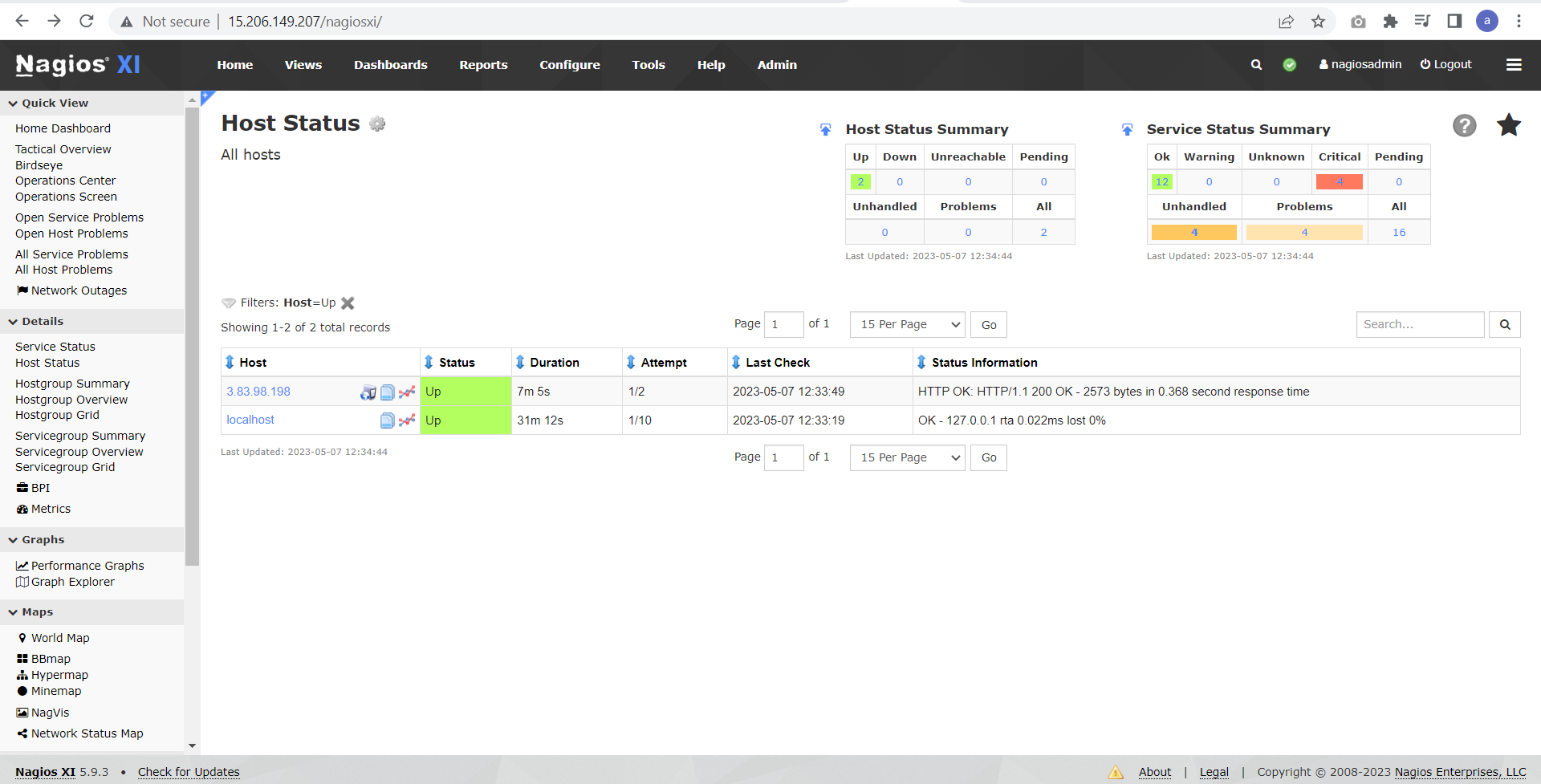
We can easily integrate the Nagios with Kubernetes, we can create a deployment file of the nagios and then apply the same file yaml file.

Once breaching the threshold we can get the email like this. Attaching an example of the emais we can get

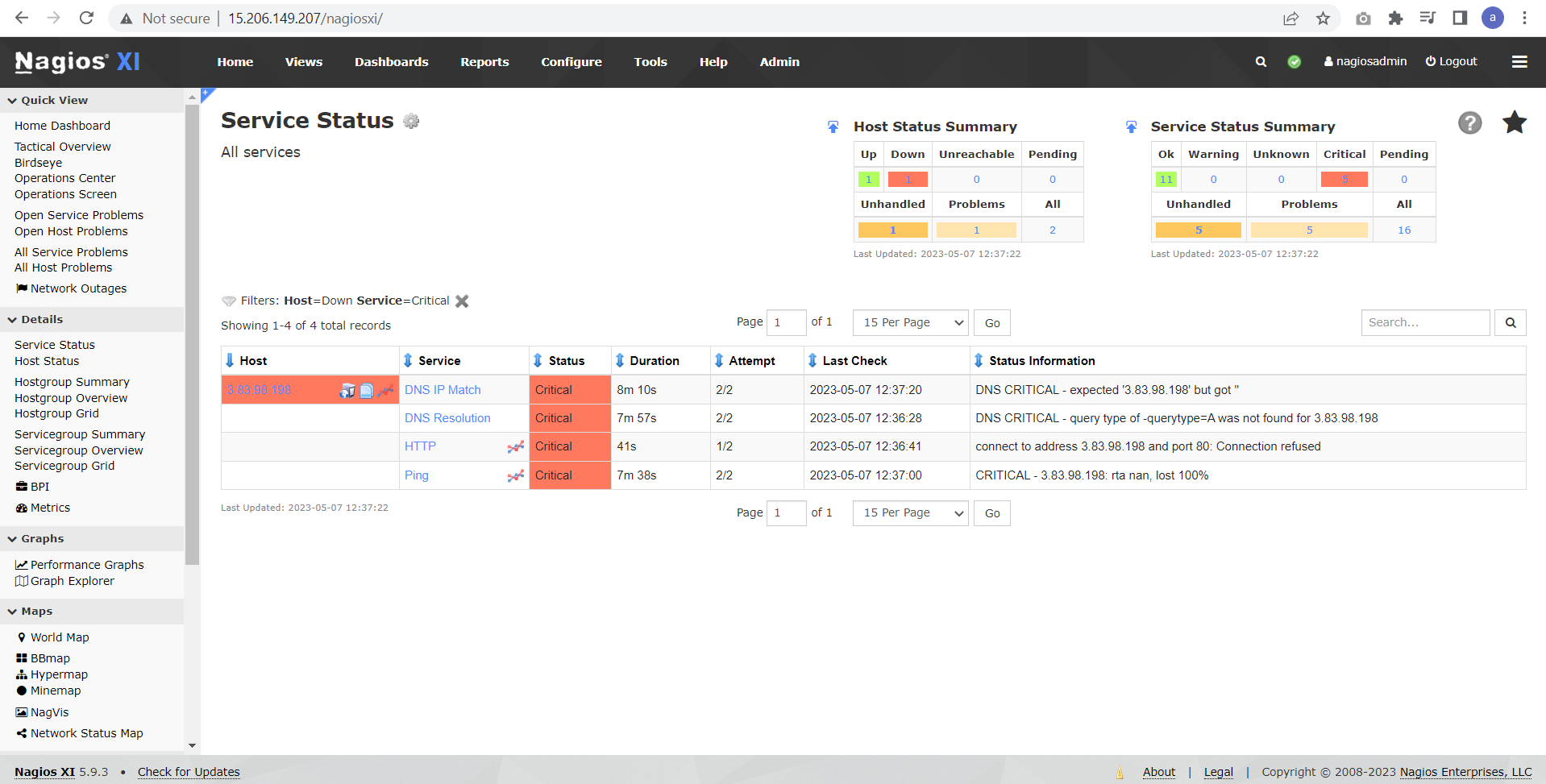


We can check the status of the pod in the nagios portal, attaching the screenshot of it.

If the status of the POD is okay



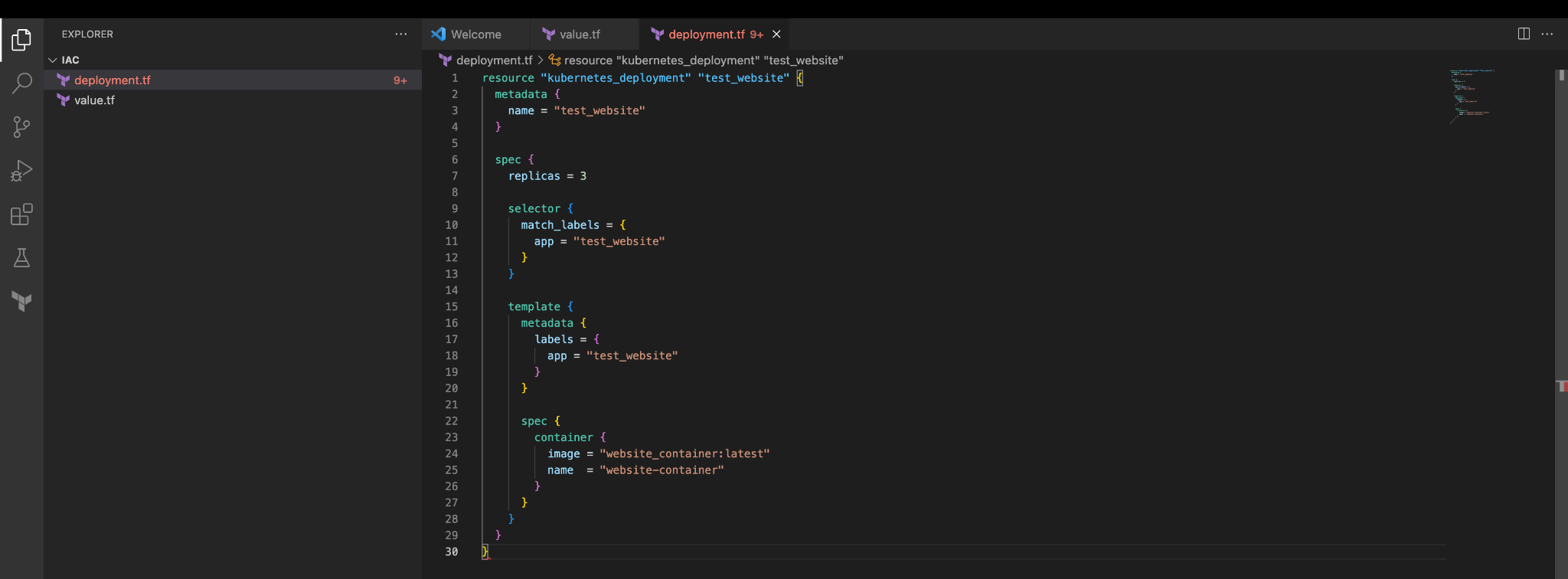
If the status is critical



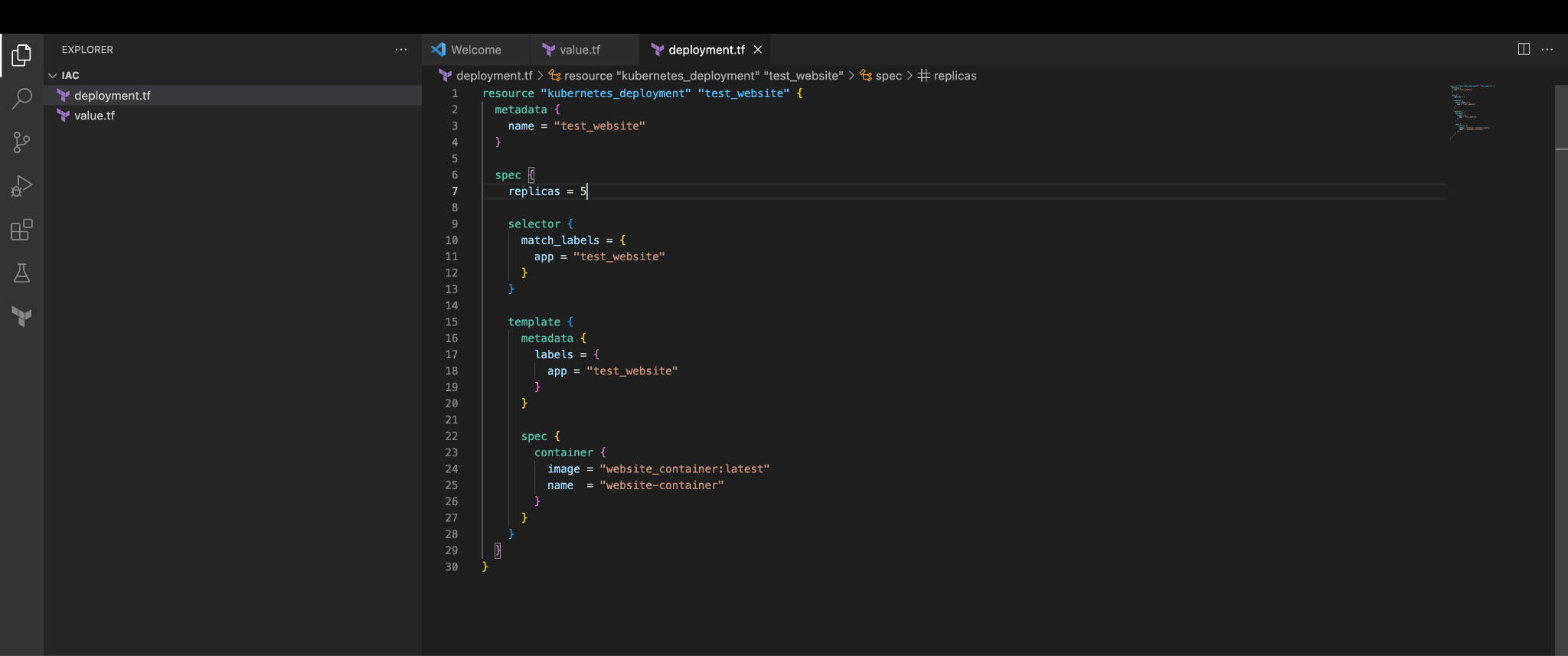
Task 2: Describe the process of creating or updating Terraform code to adjust the desired

replica count of the application.

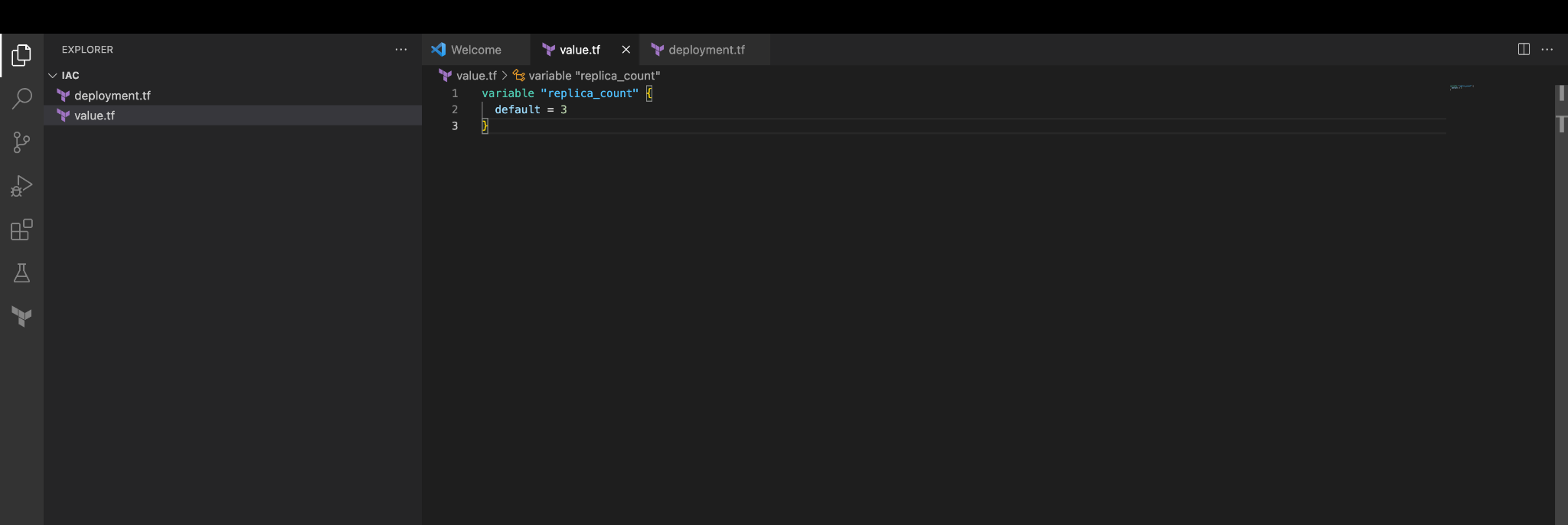
**Answer:** There is 2 way we can do this, first way is by updating the existing replica count defined in the file with the number something greater than existing one. Consider below example in current code we have kept the replica count to be 3, if the load increase we can increase the replica count to 5 and then run the command terraform apply

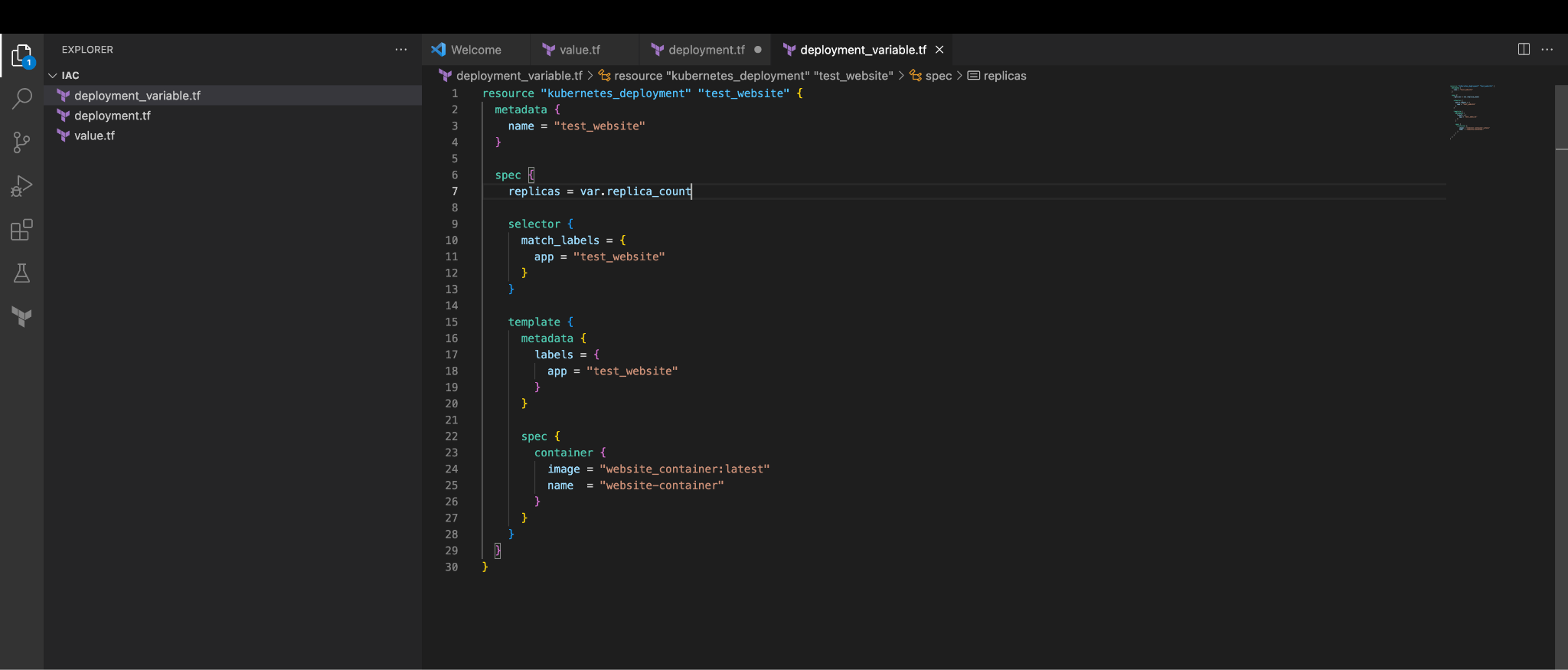


After updating the replica count



The second way we can do this is with the variable setting the value in replica and then update the variable while using terraform apply





Updating the existing count while applying using the command

terraform apply -var 'replica\_count=5'

Task 3: Provide guidelines for testing the scaling changes and deploying them to the

Kubernetes cluster while minimizing downtime.

**Answer:**

**Staging/UAT Environment**: Have a staging/UAT environment that mirrors the production setup. Test scaling changes in this staging environment first rather than directly pushing to production

**Blue-Green Deployments**: Implement blue-green deployments to further reduce downtime. This involves deploying a new version alongside the older version and gradually shifting traffic from the older version to new