# Step 3: Observability (Metrics + Health Checks)

This section describes how observability is set up in the k8s-istio-multiservice-app project using Docker Desktop with Kubernetes and Istio.  
It includes metrics collection via Istio and health probes for Kubernetes self-healing and traffic routing.

## A. Metrics Collection Using Istio

Istio demo profile comes pre-installed with key observability tools:  
- Prometheus: Metrics collection  
- Grafana: Dashboards and visualizations  
- Kiali: Service mesh visualization  
- Jaeger: Distributed tracing  
  
To access these tools locally, use port-forwarding in separate terminals:

kubectl port-forward svc/prometheus -n istio-system 9090:9090  
kubectl port-forward svc/grafana -n istio-system 3000:3000  
kubectl port-forward svc/kiali -n istio-system 20001:20001  
kubectl port-forward svc/jaeger-query -n istio-system 16686:16686

Then access the dashboards:

- Prometheus → http://localhost:9090

- Grafana → http://localhost:3000 (Login: admin / admin)

- Kiali → http://localhost:20001

- Jaeger → http://localhost:16686

## B. Health Checks (Liveness & Readiness Probes)

Each microservice includes Kubernetes health probes to allow auto-healing and traffic control.

### Frontend

Insert the following probes in `frontend-deployment.yaml`:

livenessProbe:  
 httpGet:  
 path: /  
 port: 80  
 initialDelaySeconds: 5  
 periodSeconds: 10  
readinessProbe:  
 httpGet:  
 path: /  
 port: 80  
 initialDelaySeconds: 5  
 periodSeconds: 10

### Backend

Ensure the backend exposes `/health` endpoint. Then insert these probes in `backend-deployment.yaml`:

livenessProbe:  
 httpGet:  
 path: /health  
 port: 5000  
 initialDelaySeconds: 5  
 periodSeconds: 10  
readinessProbe:  
 httpGet:  
 path: /health  
 port: 5000  
 initialDelaySeconds: 5  
 periodSeconds: 10

### Database

Use TCP probes for database service (`database-deployment.yaml`):

livenessProbe:  
 tcpSocket:  
 port: 5432  
 initialDelaySeconds: 5  
 periodSeconds: 10  
readinessProbe:  
 tcpSocket:  
 port: 5432  
 initialDelaySeconds: 5  
 periodSeconds: 10

## C. Apply Updated Manifests

After editing YAML files, apply the changes:

kubectl apply -f manifests/

Then verify pod health using:

kubectl describe pod <pod-name>

Look for 'Readiness probe succeeded' and 'Liveness probe succeeded'.

## Conclusion

With these steps, observability for the application is complete. Metrics, traces, and health statuses are now available through Istio’s integrated tools running on Docker Desktop's local Kubernetes environment.