

```
# crud-api 🛠️ A minimal, secure, and fully portable CRUD API built for  
DevSecOps demos
```

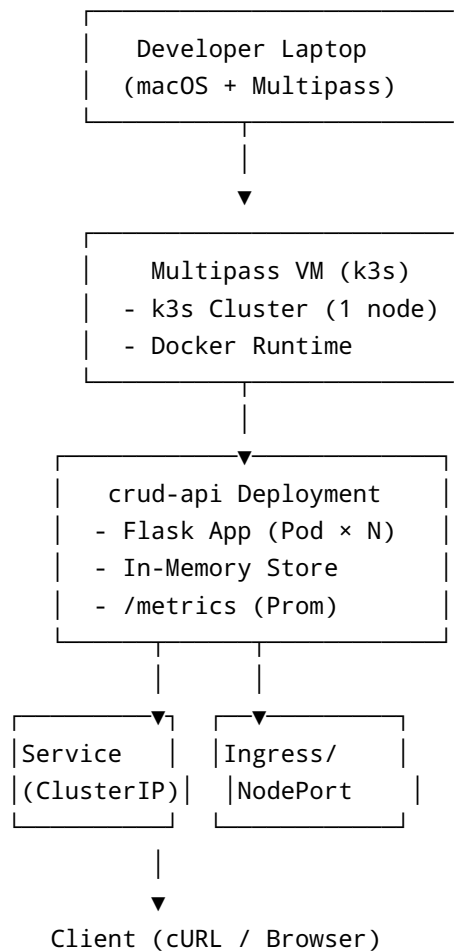
```
-
```

```
## 🕒 System Design Theme
```

This section outlines the high-level architecture, availability, and scalability considerations.

```
### 1. Architecture Overview
```

```
```plaintext
```



## 2. High Availability & Scalability

- **Replica Sets:** Increase replica count (`spec.replicas`) in `deployment.yaml` for multi-pod availability.
- **Load Balancing:** Use a Service of type `LoadBalancer` or a NodePort + external LB for traffic distribution.
- **Health Probes:** Leverage Kubernetes `readinessProbe` and `livenessProbe` to ensure only healthy pods receive traffic.
- **Horizontal Pod Autoscaling (HPA):** Configure HPA based on CPU/memory or custom Prometheus metrics (e.g., request rate).

- **Persistent Storage:** Swap in Redis or SQLite via a `PersistentVolumeClaim` for data durability.
- **Multi-Node Clusters:** Extend k3s cluster across multiple VM instances to tolerate node failure.

### 3. Component Breakdown

Component	Responsibility	Notes
<b>Multipass VM</b>	Hosts k3s cluster	Single node by default
<b>k3s (Kubernetes)</b>	Orchestrates containers	Lightweight, ideal for demos
<b>Docker</b>	Builds and packages the Flask application	Image stored locally
<b>Flask App (`)</b>	Exposes CRUD, health, and metrics endpoints	Stateless, in-memory storage
<b>Service</b>	Exposes pods internally (ClusterIP)	Port-forward for local access
<b>Security</b>	Validates <code>X-Key</code> header on mutating calls	Can be replaced with K8s Secrets
<b>Observability</b>	<code>/metrics</code> endpoint for Prometheus	Integrates with Grafana dashboards

### 4. CI/CD & Automation

```
# ci.sh
# 1. Build Docker image: docker build -t crud-api:latest .
# 2. Deploy to k3s: kubectl apply -f k8s/
# 3. Port-forward: kubectl port-forward svc/crud-api 8081:5000
```

- **Pipeline Stages:**
- **Build:** Compile, lint, and containerize the app.
- **Test:** (Optional) Run unit tests against the Flask app.
- **Scan:** Use Trivy/Bandit for SAST and container image scanning.
- **Deploy:** Apply Kubernetes manifests.
- **Verify:** Health check and metrics assertion.

### 5. Observability & Monitoring

- **Prometheus** scrapes `/metrics` for `http_requests_total`.
- **Grafana Dashboard** displays:
  - Request throughput
  - Error rates (4xx/5xx)
  - Pod resource utilization

### 6. Adaptability & Extensions

You can tailor this template to other use cases by renaming routes and fields:

Use Case	Route	Payload Fields
<b>Notes</b>	/notes	note_id, text
<b>Tasks</b>	/tasks	task_id, desc
<b>Configs</b>	/configs	key, value
<b>Policies</b>	/policies	policy_id, yaml

## 7. Cleanup & Teardown

```
kubectl delete deployment crud-api
kubectl delete svc crud-api
```

*Designed for secure, offline DevSecOps demos with focus on availability, observability, and simplicity.*