Absolutely. Here’s a list of **non-functional requirements (NFRs)** for a two-service system (UI and Backend API), both deployed as **microservices in Google Kubernetes Engine (GKE)**.

## **🔐 1. Security**

* **Authentication & Authorization**: Enforce authentication (e.g., via OAuth2, Identity-Aware Proxy) for UI and API.
* **Transport Security**: Use **HTTPS with TLS 1.2+** for all communication (e.g., between UI and API, ingress).
* **Service-to-Service Encryption**: Use **mTLS** between microservices (via Istio or Anthos Service Mesh).
* **Secrets Management**: Store sensitive config (e.g., API keys, DB creds) in **GCP Secret Manager** or Kubernetes Secrets.

## **📈 2. Scalability**

* **Horizontal Pod Autoscaling (HPA)**: Automatically scale UI and API pods based on CPU/memory or custom metrics.
* **Load Balancing**: Use **GKE Ingress or Internal Load Balancers** for distributing traffic evenly.
* **Statelessness**: Design both UI and API services to be stateless, making scaling easier.

## **☁️ 3. Availability & Reliability**

* **High Availability**: Deploy replicas across **multiple zones** using regional GKE clusters.
* **Readiness/Liveness Probes**: Ensure Kubernetes can restart unhealthy containers and only route traffic to healthy ones.
* **PodDisruptionBudgets**: Prevent downtime during upgrades or node maintenance.
* **Failover**: Use GCP Load Balancer with backend failover if a zone or pod fails.

## **📊 4. Observability**

* **Logging**: Centralized logging via **Cloud Logging** or sidecars like FluentBit/Fluentd.
* **Monitoring**: Use **Cloud Monitoring** (Stackdriver) or Prometheus to track metrics like latency, error rate, resource usage.
* **Tracing**: Distributed tracing using **OpenTelemetry** or GCP’s **Cloud Trace** for UI → API calls.

## **🔄 5. Deployment & Rollbacks**

* **CI/CD Pipelines**: Use Cloud Build, GitHub Actions, or Jenkins for automated deployments.
* **Canary/Rolling Updates**: Gradually roll out new versions using GKE deployment strategies.
* **Rollback Strategy**: Maintain fast rollback via Kubernetes Deployment revision history.

## **⏱️ 6. Performance**

* **Response Time SLA**: Backend API must respond within X ms for 95% of requests under normal load.
* **UI Load Time**: Ensure UI loads within 3 seconds for optimal UX.
* **Cold Start Time**: Keep pod cold start time < 1s using pre-warmed containers if necessary.

## **🔄 7. Resilience & Fault Tolerance**

* **Retry & Timeout Policies**: Set proper retry logic and timeouts between UI and API.
* **Circuit Breakers**: Use libraries or service mesh policies to prevent cascading failures.
* **Rate Limiting**: Prevent abuse using quota enforcement or API Gateway policies.

## **⚙️ 8. Maintainability**

* **Modular Design**: Cleanly separate UI and API codebases, each with independent versioning.
* **Clear Interfaces**: Define strong API contracts using OpenAPI/Swagger.
* **Documentation**: Maintain developer and ops documentation for setup, troubleshooting, and dependencies.

## **📁 9. Configuration Management**

* **Environment-based Configs**: Use ConfigMaps and Secrets per environment (dev, staging, prod).
* **Feature Flags**: Enable/disable features without redeploying using config-driven flags.

## **⚖️ 10. Compliance**

* **Data Residency**: Ensure data stays in allowed regions (e.g., for GDPR, HIPAA).
* **Audit Trails**: Enable auditing for sensitive actions via GCP’s **Cloud Audit Logs**.
* **Access Control**: Enforce **IAM** roles for infrastructure and Kubernetes access.

If you’re building for **production**, prioritizing **security, observability, and scalability** early on can save massive headaches later.

Would you like a checklist version of this in a table or Notion-style doc?

4o

##### **You said:**

Can you also define SLA SLI and SLO for the non functional requirements

###### **ChatGPT said:**

Absolutely! Let’s define **SLA**, **SLO**, and **SLI** in the context of your GKE microservices (UI and backend API) for the non-functional requirements mentioned.

## **📘 Definitions**

| **Term** | **Description** |
| --- | --- |
| **SLI (Service Level Indicator)** | A **metric** used to measure the performance of a service (e.g., latency, availability, error rate). |
| **SLO (Service Level Objective)** | A **target value or range** for an SLI over a period of time (e.g., 99.9% availability over 30 days). |
| **SLA (Service Level Agreement)** | A **formal commitment** (often legal) to meet specific SLOs, usually tied to penalties if not met. Used for external customers. |

## **🔐 1. Security**

| **Aspect** | **SLI** | **SLO** | **SLA** |
| --- | --- | --- | --- |
| Auth Success Rate | % of successful authenticated requests | ≥ 99.99% | ≥ 99.9% monthly |
| TLS Encryption | % of traffic encrypted via TLS | 100% | 100% |
| Secrets Rotation | Time since last secret rotation | Rotate every 90 days | Rotate every 90 days |

## **📈 2. Scalability**

| **Aspect** | **SLI** | **SLO** | **SLA** |
| --- | --- | --- | --- |
| Autoscaler Reaction Time | Time from high CPU to pod scale-up | ≤ 1 min | ≤ 2 min |
| Max Concurrent Users (UI) | Users served without degradation | 10,000 concurrent | 99% requests under 500ms |

## **☁️ 3. Availability & Reliability**

| **Aspect** | **SLI** | **SLO** | **SLA** |
| --- | --- | --- | --- |
| API Availability | % of successful responses (200s) | ≥ 99.9% over 30 days | ≥ 99.5% monthly |
| UI Availability | % of UI render success (no 5xx) | ≥ 99.9% over 30 days | ≥ 99.5% monthly |
| Downtime | Minutes of downtime | < 43.8 min/month (for 99.9%) | < 216 min/month (for 99.5%) |

## **📊 4. Observability**

| **Aspect** | **SLI** | **SLO** | **SLA** |
| --- | --- | --- | --- |
| Logging Latency | Time to ship logs to backend | ≤ 30s | ≤ 1 min |
| Metrics Coverage | % of endpoints with metrics | 100% of critical endpoints | 100% |
| Trace Completion Rate | % of requests with complete traces | ≥ 95% | ≥ 90% |

## **🔄 5. Deployment & Rollbacks**

| **Aspect** | **SLI** | **SLO** | **SLA** |
| --- | --- | --- | --- |
| Rollout Failure Rate | % of deployments rolled back | < 2% monthly | < 5% |
| Deployment Time | Time from push to rollout | ≤ 10 minutes | ≤ 15 minutes |

## **⏱️ 6. Performance**

| **Aspect** | **SLI** | **SLO** | **SLA** |
| --- | --- | --- | --- |
| API Latency (P95) | Time for 95% of API responses | ≤ 300ms | ≤ 500ms |
| UI Load Time (P95) | Time for UI to fully render | ≤ 2s | ≤ 3s |
| Cold Start Time | Time to serve first request after start | ≤ 1s | ≤ 2s |

## **🔄 7. Resilience**

| **Aspect** | **SLI** | **SLO** | **SLA** |
| --- | --- | --- | --- |
| Error Rate | % of failed requests (5xx) | ≤ 0.1% | ≤ 1% |
| Retry Success Rate | % of successful retries | ≥ 99% | ≥ 98% |
| Timeouts Honored | % of requests respecting timeout policy | ≥ 100% | 100% |

## **⚙️ 8. Maintainability**

| **Aspect** | **SLI** | **SLO** | **SLA** |
| --- | --- | --- | --- |
| Time to Restore Config | Time to apply new config | ≤ 15 min | ≤ 30 min |
| Docs Coverage | % of endpoints documented | 100% | ≥ 90% |

## **📁 9. Configuration Management**

| **Aspect** | **SLI** | **SLO** | **SLA** |
| --- | --- | --- | --- |
| Config Sync Time | Time to rollout config change | ≤ 5 min | ≤ 10 min |
| Feature Flag Toggle Time | Time to enable/disable a feature | ≤ 1 min | ≤ 2 min |

## **⚖️ 10. Compliance**

| **Aspect** | **SLI** | **SLO** | **SLA** |
| --- | --- | --- | --- |
| Audit Log Coverage | % of sensitive actions logged | 100% | 100% |
| Access Control Events | % access violations blocked | 100% | 100% |

### **✅ Example Summary (API Latency)**

* **SLI**: 95th percentile API response time
* **SLO**: ≤ 500ms for 95% of requests over 30 days
* **SLA**: ≤ 1s for 95% of requests, or customer gets 5% service credit