# Testkonzept

**Randbedingungen (Umfeld)**

* Testsystem: Linux-Host mit Docker & Docker-Compose
* Netzwerk: Alle Container laufen im dedizierten kmudienste\_net
* Browser: Firefox
* Testdaten: Dummy-Accounts, Beispiel-Dateien

# Testszenario (Testplan)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Testfall-Nr.** | **Dienst** | **Beschreibung** | **Vorgehen** | **Erwartetes Ergebnis** |
| T1 | **Nextcloud** | Upload und Persistenzprüfung | Datei in Nextcloud hochladen → Container stoppen und neu starten → erneut anmelden | Datei ist nach Neustart weiterhin vorhanden |
| T2 | **Nextcloud** | Benutzerverwaltung | Neuen Benutzer in Nextcloud anlegen → ausloggen → mit neuem Benutzer anmelden | Login mit neuem Benutzer funktioniert |
| T3 | **CVS** | Quellcode-Push | Lokales Projekt per git push ins Repository hochladen | Repository zeigt neuen Commit an |
| T4 | **CVS** | Pull & Versionierung | Projekt lokal clonen → Änderungen im Code → git pull testen | Änderungen werden korrekt synchronisiert |

## Eingesetzte Testmittel & Methoden

* Browser (Zugriff auf Webinterfaces von Nextcloud & GitLab/Gogs)
* docker ps, docker logs, docker-compose down/up (Containerüberwachung)
* Git-Befehle (git push, git pull, git clone)

## Erwartete Resultate

* Alle Dienste sind über definierte Ports erreichbar (8080, 8085, 9000)
* Daten (Wiki-Seiten, Nextcloud-Dateien, Git-Repositories) bleiben nach Neustarts erhalten
* Benutzer- und Rechteverwaltung funktioniert wie vorgesehen

# Sicherheitskonzept

### Containerspezifische Risiken & Massnahmen

1. **Klartext-Passwörter im Compose-File**
   * Risiko: Zugangsdaten könnten im Repository sichtbar sein
   * Massnahme: Verwendung von .env-Dateien → Passwörter nicht im docker-compose.yml
2. **Datenverlust bei Container-Neustart**
   * Risiko: Ohne Volumes gehen Daten verloren
   * Massnahme: Persistente Volumes für Nextcloud, MediaWiki, GitLab/Gogs
3. **Netzwerkoffenheit**
   * Risiko: Dienste wären von aussen erreichbar
   * Massnahme: Zugriff nur über definierte Ports (8080, 8085, 9000), internes Netzwerk kmudienste\_net
4. **Ressourcenverbrauch / DoS**
   * Risiko: Ein Container könnte CPU/RAM überlasten
   * Massnahme: Ressourcenkontrolle im Compose (mem\_limit, cpus)
5. **Monitoring**
   * Risiko: Probleme bleiben unbemerkt
   * Massnahme: Portainer für Überwachung + Log-Analyse