readme.md for @serve.zone/gitops

A unified dashboard for managing Gitea and GitLab instances — browse projects, manage secrets, monitor CI/CD pipelines, stream build logs, sync configurations, and receive webhook notifications, all from a single app. □□

Issue Reporting and Security

For reporting bugs, issues, or security vulnerabilities, please visit community.foss.global/. This is the central community hub for all issue reporting. Developers who sign and comply with our contribution agreement and go through identification can also get a code.foss.global/ account to submit Pull Requests directly.

□ Features

- **Multi-Provider** — Connect to Gitea and GitLab simultaneously via a unified provider abstraction
- **Secrets Management** — View, create, update, and delete CI/CD secrets across projects and groups
- **Managed Secrets** — Define secrets once and push them to multiple providers/scopes automatically
- **Pipeline Monitoring** — Browse pipelines with time-range filtering, view modes, group aggregation, and sorting; view jobs, retry failed builds, cancel running ones
- **Build Log Streaming** — Fetch and display raw job logs with monospace rendering and live polling
- **Sync Configurations** — Define repo sync rules across providers with status tracking
- **Action Log** — Global audit trail of all operations across the system
- **Webhook Integration** — Receive push/PR/pipeline events via `POST /webhook/:connectionId` and broadcast to all connected WebSocket clients in real-time
- **Secrets Cache & Scanning** — Background scan service fetches and caches all secrets every 24h with upsert-based deduplication
- **Secure Token Storage** — Connection tokens stored in OS keychain via `@push.rocks/smartsecret` (encrypted file fallback), never in plaintext on disk

- **Auto-Refresh** — Frontend polls for updates every 30s, with manual refresh available on every view
- **Embedded SPA** — Frontend is bundled (base64-encoded) and served from memory, no static file server needed

□□ Install

Prerequisites

- Deno v2+
- pnpm (for frontend deps and bundling)
- MongoDB-compatible database (auto-provisioned via `@push.rocks/smartmongo`)

Setup

```
# Clone the repository
git clone https://code.foss.global/serve.zone/gitops.git
cd gitops

# Install frontend dependencies
pnpm install

# Build the frontend bundle
pnpm build

# Start the server
deno run --allow-all mod.ts server
```

The app will be available at `http://localhost:3000`.

⚙ Configuration

All configuration is done through environment variables:

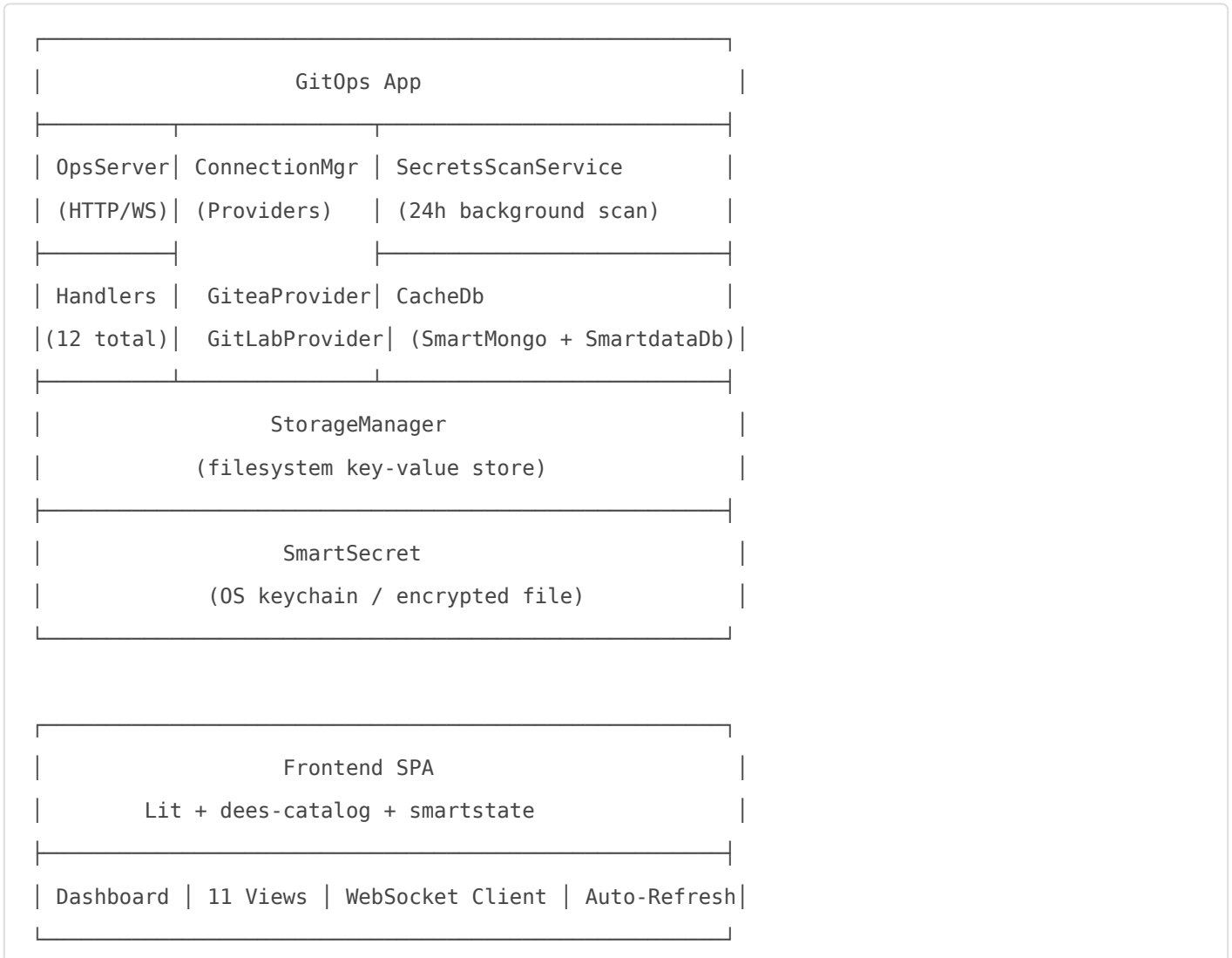
Variable	Default	Description
`GITOPS_PORT`	`3000`	HTTP/WebSocket server port

Variable	Default	Description
`GITOPS_ADMIN_USERNAME`	`admin`	Admin login username
`GITOPS_ADMIN_PASSWORD`	`admin`	Admin login password

Data is stored at `~/.serve.zone/gitops/`:

```
~/.serve.zone/gitops/
├── storage/          # Connection configs (JSON, tokens replaced with keychain refs)
│   └── connections/  # One file per connection
└── tsmdb/            # Embedded MongoDB data (cached secrets, projects)
```

□□ Architecture



Backend (`ts/`)

- **`GitopsApp`** — Main orchestrator. Owns all subsystems, handles startup/shutdown lifecycle.
- **`ConnectionManager`** — CRUD for provider connections. Tokens secured in OS keychain. Background health checks on startup.
- **`BaseProvider`** → **`GiteaProvider`** / **`GitLabProvider`** — Unified interface over both APIs (projects, groups, secrets, pipelines, jobs, logs).
- **`OpsServer`** — TypedServer-based HTTP/WebSocket server with 12 handler modules:
  - `AdminHandler` — JWT-based auth (login/logout/verify)
  - `ConnectionsHandler` — Connection CRUD + test
  - `ProjectsHandler` / `GroupsHandler` — Browse repos and orgs
  - `SecretsHandler` — Cache-first secret CRUD
  - `ManagedSecretsHandler` — Managed secret definitions and push operations
  - `PipelinesHandler` — Pipeline list/jobs/retry/cancel with filtering and aggregation
  - `LogsHandler` — Job log fetch
  - `WebhookHandler` — Custom HTTP route for incoming webhooks
  - `ActionsHandler` — Force scan / scan status
  - `ActionLogHandler` — Global audit trail queries
  - `SyncHandler` — Repo sync configuration and status
- **`SecretsScanService`** — Background scanner with upsert-based deduplication. Runs on startup and every 24h.
- **`CacheDb`** — Embedded MongoDB via `SmartMongo` + `SmartdataDb`. TTL-based expiration with periodic cleanup.
- **`StorageManager`** — Filesystem-backed key-value store with atomic writes.

Frontend (`ts_web/`)

- Built with Lit web components using TC39 standard decorators and @design.estate/dees-catalog UI library
- Reactive state management via `smartstate` (login, connections, data, UI state parts)
- 11 tabbed views: Overview, Connections, Projects, Groups, Secrets, Managed Secrets, Pipelines, Build Log, Actions, Action Log, Sync
- WebSocket client for real-time webhook push notifications
- Bundled to `ts_bundled/bundle.ts` via `@git.zone/tsbundle` (base64-encoded, committed to git)

Shared Types (`ts_interfaces/`)

- `data/` — Data models (`IProject`, `ISecret`, `IPipeline`, `IIdentity`, `IConnection`, `ISyncConfig`, `IManagedSecret`, `IActionLogEntry`, etc.)
- `requests/` — TypedRequest interfaces for all RPC endpoints

□□ API

All endpoints use TypedRequest — a typed RPC protocol over HTTP POST to `/typedrequest`.

Authentication

```
// Login → returns JWT identity
{ method: 'adminLogin', request: { username, password } }
// → { identity: { jwt, userId, role, expiresAt } }

// All other requests require identity
{ method: 'getProjects', request: { identity, connectionId } }
```

Connections

Method	Description
`getConnections`	List all connections (tokens masked)
`createConnection`	Add a new Gitea/GitLab connection
`updateConnection`	Update connection name/URL/token
`testConnection`	Verify connection is reachable
`deleteConnection`	Remove a connection

Data

Method	Description
`getProjects`	List projects (with search/pagination)
`getGroups`	List groups/orgs (with search/pagination)
`getAllSecrets`	Get all secrets for a connection+scope (cache-first)
`getSecrets`	Get secrets for a specific entity (cache-first)
`createSecret` / `updateSecret` / `deleteSecret`	Secret CRUD
`getPipelines`	List pipelines for a project (with time-range filtering)
`getPipelineJobs`	List jobs for a pipeline

Method	Description
`retryPipeline` / `cancelPipeline`	Pipeline actions
`getJobLog`	Fetch raw build log for a job

Managed Secrets

Method	Description
`getManagedSecrets`	List managed secret definitions
`createManagedSecret` / `updateManagedSecret` / `deleteManagedSecret`	Managed secret CRUD

Sync

Method	Description
`getSyncConfigs`	List sync configurations
`createSyncConfig` / `updateSyncConfig` / `deleteSyncConfig`	Sync config CRUD
`getRepoSyncStatus`	Get sync status for repos

Actions

Method	Description
`forceScanSecrets`	Trigger immediate full secrets scan
`getScanStatus`	Get scan status, last result, timestamp
`getActionLog`	Query global audit trail

Webhooks

```
# Register this URL in your Gitea/GitLab webhook settings
POST http://your-server:3000/webhook/<connectionId>
```

Events are parsed from `X-Gitea-Event` / `X-Gitlab-Event` headers and broadcast to all connected WebSocket clients as `webhookNotification`.

□□ Development

```
# Watch mode — auto-rebuilds frontend + restarts backend on changes
pnpm run watch

# Run tests (Deno)
pnpm test

# Build frontend bundle only
pnpm build

# Start server directly
deno run --allow-all mod.ts server
```

Project Structure

```
gitops/
├── mod.ts                    # Entry point
├── deno.json                 # Deno config + import map
├── package.json              # npm metadata + scripts
├── .smartconfig.json         # tsbundle + tswatch config
├── html/index.html           # HTML shell
├── ts/                       # Backend
│   ├── classes/              # GitopsApp, ConnectionManager, SyncManager, ActionLog
│   ├── providers/            # BaseProvider, GiteaProvider, GitLabProvider
│   ├── storage/              # StorageManager
│   ├── cache/                # CacheDb, CacheCleaner, SecretsScanService
│   │   └── documents/        # CachedProject, CachedSecret
│   └── opsserver/            # OpsServer + 12 handlers
│       ├── handlers/         # AdminHandler, SecretsHandler, SyncHandler, etc.
│       └── helpers/          # Guards (JWT verification)
├── ts_interfaces/            # Shared TypeScript types
│   ├── data/                 # IProject, ISecret, IPipeline, etc.
│   └── requests/             # TypedRequest interfaces
├── ts_web/                   # Frontend SPA
│   ├── appstate.ts           # Smartstate store + actions
│   └── elements/             # Lit web components
```

```
│       └── views/            # 11 view components
├── ts_bundled/bundle.ts      # Embedded frontend (base64, committed)
└── test/                     # Deno tests
```

License and Legal Information

This repository contains open-source code licensed under the MIT License. A copy of the license can be found in the license file.

Please note: The MIT License does not grant permission to use the trade names, trademarks, service marks, or product names of the project, except as required for reasonable and customary use in describing the origin of the work and reproducing the content of the NOTICE file.

Trademarks

This project is owned and maintained by Task Venture Capital GmbH. The names and logos associated with Task Venture Capital GmbH and any related products or services are trademarks of Task Venture Capital GmbH or third parties, and are not included within the scope of the MIT license granted herein.

Use of these trademarks must comply with Task Venture Capital GmbH's Trademark Guidelines or the guidelines of the respective third-party owners, and any usage must be approved in writing. Third-party trademarks used herein are the property of their respective owners and used only in a descriptive manner, e.g. for an implementation of an API or similar.

Company Information

Task Venture Capital GmbH Registered at District Court Bremen HRB 35230 HB, Germany

For any legal inquiries or further information, please contact us via email at hello@task.vc.

By using this repository, you acknowledge that you have read this section, agree to comply with its terms, and understand that the licensing of the code does not imply endorsement by Task Venture Capital GmbH of any derivative works.

Revision #3
Created 2026-03-28 11:14:40 UTC by foss.global Team
Updated 2026-03-28 12:21:26 UTC by foss.global Team