@serve.zone/gitops

easy maintenance of your gitea/gitlab instance.

- readme.md for @serve.zone/gitops
- changelog.md for @serve.zone/gitops

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:

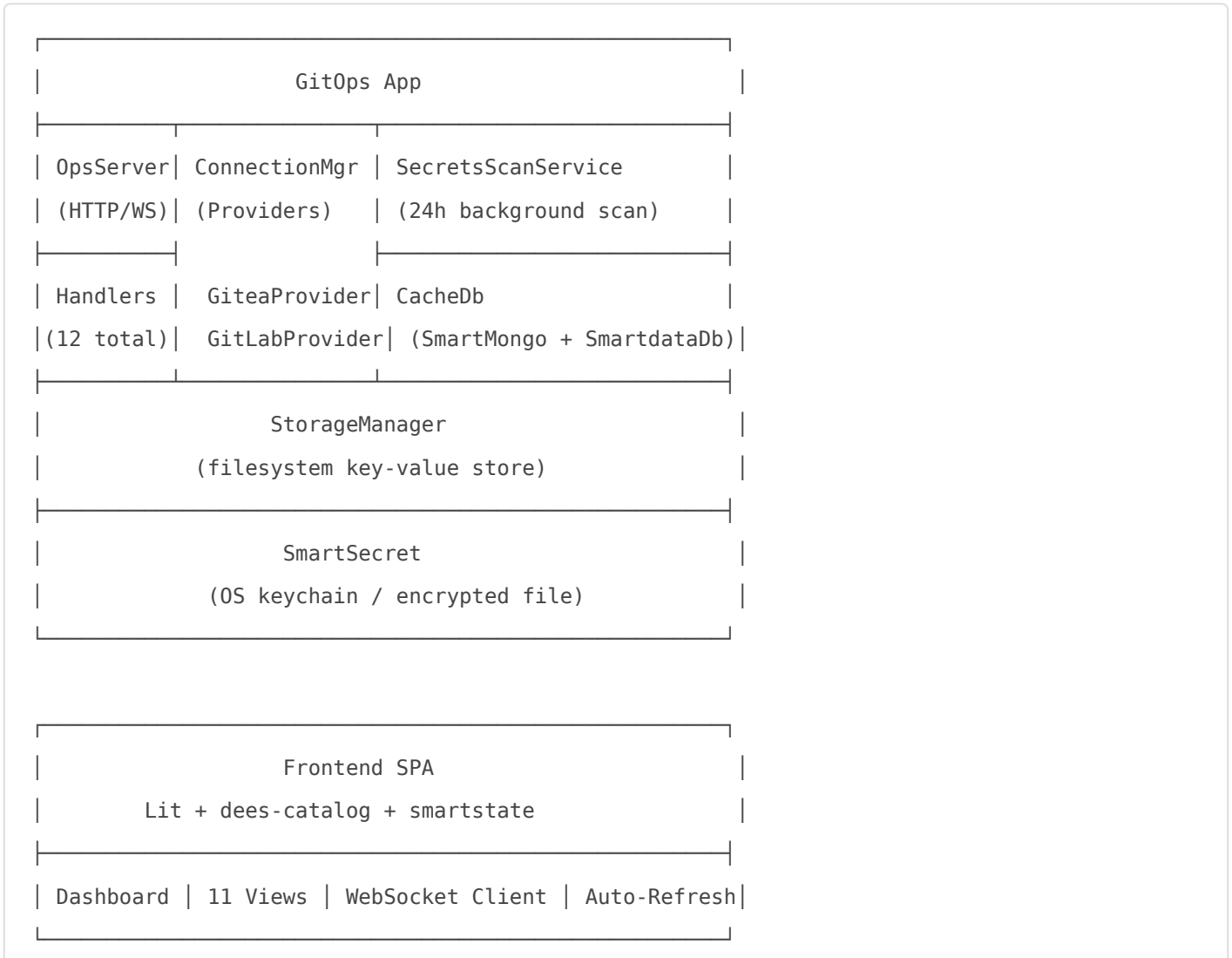
Variable	Default	Description
`GITOPS_PORT`	`3000`	HTTP/WebSocket server port

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.

changelog.md for @serve.zone/gitops

2026-03-28 - 2.13.0 - feat(cache,build,docs)

switch cache storage to SmartMongo and align build configuration with updated dependencies

- replaces LocalTsmDb usage with SmartMongo in the cache layer and related tests
- moves tsbundle and tswatch configuration from npmextra.json to .smartconfig.json
- updates Deno, frontend, and provider dependencies and adds @std/assert imports for tests
- refreshes README to document managed secrets, sync, action log, and the expanded handler and view set

2026-03-02 - 2.12.0 - feat(pipelines)

add pipelines view modes, time-range filtering, group aggregation, sorting, and job log polling

- Add view modes for pipelines: 'current', 'project', 'group', and 'error'; support timeRange and sortBy parameters on getPipelines requests and in the UI.
- Implement aggregated pipeline fetching across projects with batching, deduplication, and active-run prioritization (fetchCurrentPipelines, fetchAggregatedPipelines, fetchGroupPipelines, fetchErrorPipelines).
- Add filtering by time ranges (1h, 6h, 1d, 3d, 7d, 30d) and sorting options (created, duration, status) with status priority ordering.
- Extend BaseProvider API: add IPipelineListOptions (status, ref, source), add getGroupProjects, and update Gitea/GitLab providers to honor new options and include projectName mapping.

- Enhance web UI: new controls/state for viewMode, timeRange, sortBy, group selection, plus job log polling with auto-scroll and cleanup on disconnect.
- Bump dependencies: @apiclient.xyz/gitea 1.3.0 -> 1.4.0 and @apiclient.xyz/gitlab 2.4.0 -> 2.5.0.

2026-03-02 - 2.11.1 - fix(meta)

update repository metadata (non-functional change)

- Change was metadata-only (+1 -1) with no source code changes
- Current package.json version: 2.11.0 — recommend patch bump to 2.11.1

2026-03-02 - 2.11.0 - feat(sync)

add branch & tag listing support and improve sync mirroring and sync log routing

- Bump @apiclient.xyz/gitea to 1.3.0 and @apiclient.xyz/gitlab to 2.4.0
- Add IBranch and ITag interfaces and export them from ts_interfaces
- Add getBranches/getTags to BaseProvider and implement paginated branch/tag listing for Gitea and GitLab providers
- SyncManager now creates a temporary mirrors directory (RAM-backed), auto-cleans it on shutdown, and no longer requires a configured syncMirrorsPath (removed from paths and gitopsapp)
- Add refsMatch in SyncManager to accurately compare local branches/tags with target refs (uses for-each-ref and ls-remote) to avoid unnecessary pushes
- Introduce avatarUploadCache and other internal sync manager improvements
- Change log channel/tagging: sync log messages use 'sync' (was 'git') and TypedSocket broadcasts use a new 'syncLogClient' tag; web client now sets that tag when creating the socket

2026-02-28 - 2.10.0 - feat(managed-secrets)

add centrally managed secrets with GITOPS_ prefix pushed to multiple targets

- Add IManagedSecret, IManagedSecretTarget, IManagedSecretStored interfaces and TypedRequest contracts for CRUD + push operations

- Add ManagedSecretsManager with keychain-backed storage, upsert push logic, target diff on update, and best-effort delete
- Add ManagedSecretsHandler with 7 endpoints wired into OpsServer with auth guards and action logging
- Add frontend state part, 6 appstate actions, and Managed Secrets view with table, target picker, and push/edit/delete modals
- Add Managed Secrets tab to dashboard after Secrets
- Extend action log types with 'managed-secret' entity and 'push' action

2026-02-28 - 2.9.0 - feat(sync)

remove target avatar when source has none to keep avatars fully in sync

- Add removeProjectAvatar and removeGroupAvatar methods for GitLab and Gitea APIs
- In syncProjectMetadata, remove target project avatar when source has no avatar and no group fallback applies
- When useGroupAvatarsForProjects is enabled but the group also has no avatar, remove the target avatar
- In syncGroupMetadata, remove target group avatar when source group has no avatar

2026-02-28 - 2.8.0 - feat(sync)

add sync subsystem: SyncManager, OpsServer sync handlers, Sync UI and state, provider groupFilter support, and realtime sync log streaming via TypedSocket

- Introduce SyncManager and wire it into GitopsApp (init/stop) with a new syncMirrorsPath
- Add typedrequest SyncHandler with endpoints to create/update/delete/pause/trigger/preview sync configs and fetch repo statuses/logs
- Add sync data interfaces (ISyncConfig, ISyncRepoStatus, ISyncLogEntry) and action log integration for sync operations
- Add web UI: gitops-view-sync, appstate sync actions/selectors, and preview/status/modals for sync configs
- Add groupFilter and groupFilterId to connection model; migrate legacy baseGroup/baseGroupId to groupFilter fields on load
- Providers (Gitea/GitLab) and BaseProvider now accept groupFilterId and scope project/group listings accordingly (auto-pagination applies)
- Logging: add sync log buffer, getSyncLogs API, and broadcast sync log entries to connected clients via TypedSocket; web client listens and displays entries
- Update dependencies: bump @apiclient.xyz/gitea and gitlab versions and add @api.global/typedsocket
- Connections UI: expose Group Filter field and pass through on create/update

2026-02-24 - 2.7.1 - fix(repo)

update file metadata (mode/permissions) without content changes

- One file changed: metadata-only (+1,-1).
- No source or behavior changes — safe to bump patch version.
- Change likely involves file mode/permission or metadata update only.

2026-02-24 - 2.7.0 - feat(secrets)

add ability to fetch and view all secrets across projects and groups, include scopeName, and improve frontend merging/filtering

- Add new typed request and handler getAllSecrets to opsserver to bulk-fetch secrets across projects or groups (batched and using Promise.allSettled for performance).
- Extend ISecret with scopeName and update provider mappings (Gitea/GitLab) and secret return values to include scopeName.
- Frontend: add fetchAllSecretsAction, add an "All" option in the Secrets view, filter table by selected entity or show all, and disable "Add Secret" when "All" is selected.
- Create/update actions now merge only the affected entity's secrets into state instead of replacing the entire list; delete now filters by key+scope+scopeId to avoid removing unrelated secrets.
- UI: table now shows a Scope column using scopeName (or fallback to scopeId), selection changes trigger reloading of entities and secrets.

2026-02-24 - 2.6.2 - fix(meta)

update file metadata only (no source changes)

- One file changed: metadata-only (e.g. permissions/mode) with no content modifications.
- No code, dependency, or API changes detected; safe patch release recommended.
- Bump patch version from 2.6.1 to 2.6.2.

2026-02-24 - 2.6.1 - fix(package.json)

apply metadata-only update (no functional changes)

- Change is metadata-only (+1 -1) in a single file — no code or behavior changes
- Current package.json version is 2.6.0; recommend a patch bump to 2.6.1

2026-02-24 - 2.6.0 - feat(webhook)

add webhook endpoint and client push notifications, auto-refresh UI, and gitea id mapping fixes

- Add WebhookHandler with POST /webhook/:connectionId that parses provider-specific headers and broadcasts webhookNotification via TypedSocket to connected clients
- Frontend: add auto-refresh toggle, refresh-interval action, dashboard auto-refresh timer, and views subscribing to gitops-auto-refresh events to refresh data
- Frontend: add WebSocket client with reconnect logic to receive push notifications and trigger auto-refresh on webhook events
- Gitea provider: prefer repository full_name and organization name when mapping project and group ids to ensure stable identifiers
- Bump devDependencies: @git.zone/tsbundle ^2.9.0 and @git.zone/tswatch ^3.2.0
- Add ts_bundled/bundle.js and bundle.js.map to .gitignore

2026-02-24 - 2.5.0 - feat(gitea-provider)

auto-paginate Gitea repository and organization listing; respect explicit page option and default perPage to 50

- getProjects and getGroups now auto-fetch all pages when opts.page is not provided
- When opts.page is provided, the provider respects it and does not auto-paginate
- Defaults perPage to 50 for paginated requests
- Dependency @design.estate/dees-catalog bumped from ^3.43.0 to ^3.43.3

2026-02-24 - 2.4.0 - feat(opsserver)

serve embedded frontend bundle from committed ts_bundled instead of using external dist_serve directory

- Switch server to use bundledContent from committed ts_bundled bundle (base64ts) instead of pointing at a serveDir
- Update bundler config to emit ./ts_bundled/bundle.ts with outputMode 'base64ts' and includeFiles mapping
- Remove dist_serve from .gitignore and commit ts_bundled (embedded frontend bundle)
- Bump devDependency @git.zone/tsbundle to ^2.8.4 and deno dependency @api.global/typedserver to ^8.3.1

2026-02-24 - 2.3.0 - feat(storage)

add StorageManager and cache subsystem; integrate storage into ConnectionManager and GitopsApp, migrate legacy connections, and add tests

- Add StorageManager with filesystem and memory backends, key normalization, atomic writes and JSON helpers (getJSON/setJSON).
- ConnectionManager now depends on StorageManager, persists each connection as /connections/<id>.json, and includes a one-time migration from legacy .nogit/connections.json.
- Introduce cache subsystem: CacheDb (LocalTsmDb + Smartdata), CacheCleaner, CachedDocument and CachedProject for TTL'd cached provider data, plus lifecycle management in GitopsApp.
- GitopsApp now initializes StorageManager, wires ConnectionManager to storage, starts/stops CacheDb and CacheCleaner, and uses resolved default paths via resolvePaths.
- Export smartmongo and smartdata in plugins and add corresponding deps to deno.json.
- Add comprehensive tests: storage unit tests, connection manager integration using StorageManager, and a tsmdb + smartdata spike test.

2026-02-24 - 2.2.1 - fix(ts_bundled)

add generated bundled JavaScript and source map for ts build (bundle.js and bundle.js.map)

- Added ts_bundled/bundle.js (≈168 KB) - compiled/bundled output from ts sources
- Added ts_bundled/bundle.js.map (≈309 KB) - source map referencing ../ts/index.ts and ../ts_web/index.ts
- This is generated build output (deno bundle) and does not change runtime API

2026-02-24 - 2.2.0 - feat(opsserver)

Serve bundled frontend from a dedicated dist_serve directory and update frontend UI/packaging

- Serve static site using UtilityWebsiteServer with serveDir set to ./dist_serve and pass port into server.start()
- Update bundler config: output bundle to ./dist_serve/bundle.js, change outputMode to 'bundle', and include html/index.html
- Move root index.html into html/index.html and update .gitignore to ignore dist_serve/ (replace ts_bundled)
- Frontend enhancements: add iconName to view tabs and resolvedViewTabs, add Lucide icons for each tab, replace manual stats markup with dees-statsgrid using IStatsTile tiles
- Adjust shared CSS: center content, set max-width 1280px and adjust padding
- Add npm test script and rename/update tests (test.basic.ts -> test.basic_test.ts)

2026-02-24 - 2.1.0 - feat(opsserver)

switch to TypedServer and serve bundled UI assets; add index.html; update bundling output and dev watch configuration

- Replace UtilityWebsiteServer with TypedServer and load bundledContent from ts_bundled/bundle.ts; enable cors, spaFallback, injectReload, watch, and compression
- Add a minimal index.html SPA entry and include it in the bundle so it is served from the bundled assets
- Change tsbundle output to ./ts_bundled/bundle.ts with outputMode 'base64ts' and includeFiles ['./index.html']
- Add a tswatch bundle config and replace the previous watcher with a backend watcher that runs the server via 'deno run --allow-all mod.ts server' (restart enabled)
- Bump devDependency @git.zone/tswatch from ^2.3.13 to ^3.1.0 and update .gitignore to ignore ts_bundled/

2026-02-24 - 2.0.0 - BREAKING CHANGE(providers)

switch GitLab and Gitea providers to use @apiclient.xyz client libraries and export clients via plugins

- Add new dependencies: @apiclient.xyz/gitea@^1.0.3 and @apiclient.xyz/gitlab@^2.0.3 in deno.json
- Export giteaClient and gitlabClient from ts/plugins.ts
- Refactor GiteaProvider to use plugins.giteaClient.GiteaClient for all API interactions and update mapping types/fields
- Refactor GitLabProvider to use plugins.gitlabClient.GitLabClient for all API interactions and update mapping types/fields
- Remove BaseProvider.apiFetch helper and setAuthHeader abstract method (breaking change to BaseProvider API)
- Adjust mapping logic: simplify Gitea visibility, change group name/path and webUrl construction, update GitLab topics and projectCount handling

2026-02-24 - 1.0.0 - initial release

Initial commit and first release of the project.

- Project initialized with the initial codebase and repository scaffold.
- Base files and basic configuration added.
- Tagged first release as 1.0.0.