redsnano Guide

Mini Redis-style cache with hash validation and multi-surface APIs.

Overview

redsnano is a Redis-inspired cache that stores Python objects while continuously verifying them against the canonical data source using SHA-256 hashes. Whenever a hash mismatch is detected, the cache refreshes itself from the origin.

Installation

Clone the repository, then install in editable mode:

```
  git clone https://github.com/<org>/redsnano.git
  cd redsnano
  pip install -e .[api]
```

The [api] extra installs FastAPI/uvicorn for the HTTP demo.

Python API

```
from redsnano import MiniRedis, DictionaryOriginStore
origin = DictionaryOriginStore({'user:1': {'name': 'Alice'}})
cache = MiniRedis(origin, default_ttl=60)
cache.get('user:1')  # fetch + store
cache.set('user:2', {'name': 'Bob'}, ttl=30)
cache.delete('user:2')
```

FastAPI Service

Run: uvicorn redsnano.fastapi_app:app --reload

Endpoints:

```
 POST /users {username,email} -> upsert user, seed cache
 GET /users/{username} -> serve from cache, refresh via hash
```

SQLite persists origin data through SQLiteUserRepository.

Standalone HTTP Cache

Start the bundled HTTP server for cross-language clients:

```
  redsnano-server --origin-json origin.json --port 8080
```

Sample requests:

```
  curl http://localhost:8080/cache/user:1
  curl -X PUT http://.../cache/user:2 -d '{"value": {...}}'
  curl -X DELETE http://.../cache/user:2
```

Validating Data

Every cache entry stores compute_hash(value). When GET runs, the hash is compared with origin_store.fetch_hash(key). If different, the cache fetches fresh data and overwrites the entry before returning a response.

Testing

Unit tests cover the core cache plus FastAPI endpoints.

```
    pytest
```

CI can run the same command to guard against regressions.

Git Workflow

```
1. git status
2. git add <files>
3. git commit -m "feat: describe change"
4. git push origin main
```

Keep commits scoped and include tests.

Use Cases

- Protect API clients from stale DB reads while avoiding full re-fetches
- Deliver configuration caches that validate against JSON/SQL sources
- Provide lightweight Redis-like semantics without external services