

# Persistent Key-Value Store

Architectural & API Reference

Pujith Sai Kumar Korlepara

IIT Bombay (ID: 25M0787)

Email: [pujith@cse.iitb.ac.in](mailto:pujith@cse.iitb.ac.in) / [pujith22.sde@gmail.com](mailto:pujith22.sde@gmail.com)

GitHub: [pujith22](#) LinkedIn: [in/pujith22](#)

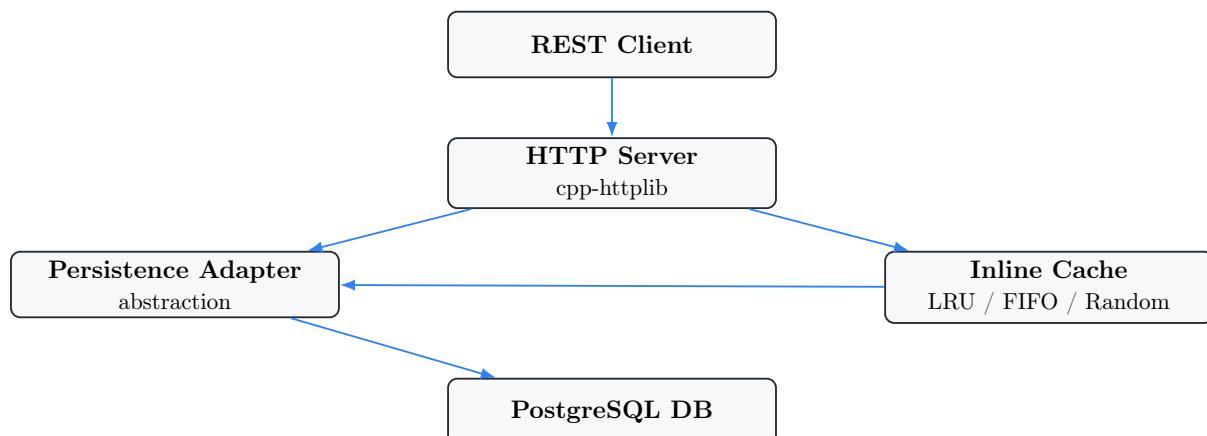
Portfolio: [cse.iitb.ac.in/ pujith](http://cse.iitb.ac.in/)

GitHub Repository: <https://github.com/pujith22/persistent-key-value-store>

## 1 Purpose

A C++17 HTTP service providing a persistent key-value cache backed by PostgreSQL. It exposes a JSON-centric REST API for CRUD and bulk operations while maintaining latency via an inline in-memory write-through cache. Startup fails fast if persistence cannot be reached, enforcing durability guarantees.

## 2 High-Level Architecture



**Figure 1:** Compact layered architecture: request flow across server, cache, and persistence.

## 3 Postman API Collection

An interactive Postman collection is published for quick exploration and manual testing of the API surface. You can import it directly via the shared workspace URL below or by copying the JSON specification.

Shared Collection URL: [Postman Workspace Link](#)

The local development server (see `main_server.cpp`) binds to host `localhost` on port 2222. The collection above targets that base URL.

## Minimal Postman Collection JSON

```
{
  "info": {
    "name": "Persistent Key Value Store API",
    "schema": "https://schema.getpostman.com/json/collection/v2
      .1.0/collection.json",
    "description": "CRUD, bulk query, transactional update and
      observability endpoints"
  },
  "item": [
    {"name": "Catalog", "request": {"method": "GET", "url": "http
      ://localhost:2222/"}},
    {"name": "Home", "request": {"method": "GET", "url": "http://
      localhost:2222/home"}},
    {"name": "Get Key (cache/persist)", "request": {"method": "GET"
      , "url": "http://localhost:2222/get_key/42"}},
    {"name": "Bulk Query", "request": {"method": "PATCH", "header":
      [{"key": "Content-Type", "value": "application/json"}], "body":
      {"mode": "raw", "raw": "{\n  \"data\": [1,2,3,4]\n}"}
      , "url": "http://localhost:2222/bulk_query"}},
    {"name": "Insert", "request": {"method": "POST", "url": "http
      ://localhost:2222/insert/42/alpha"}},
    {"name": "Bulk Update Txn", "request": {"method": "POST", "header":
      [{"key": "Content-Type", "value": "application/json"}], "body":
      {"mode": "raw", "raw": "{\n  \"ops\": [\n    {\n      \"type\": \"insert\", \"key\": 7, \"value\": \"seven\" },\n      {\n        \"type\": \"get\", \"key\": 7 }\n    ],\n    {\n      \"type\": \"update\", \"key\": 7, \"value\": \"SEVEN\"\n    }\n  ]\n}"}},
    {"name": "Delete Key", "request": {"method": "DELETE", "url": "http
      ://localhost:2222/delete_key/42"}},
    {"name": "Update Key", "request": {"method": "PUT", "url": "http
      ://localhost:2222/update_key/42/beta"}},
    {"name": "Health", "request": {"method": "GET", "url": "http://
      localhost:2222/health"}},
    {"name": "Metrics", "request": {"method": "GET", "url": "http
      ://localhost:2222/metrics"}},
    {"name": "Stop (dev only)", "request": {"method": "GET", "url":
      "http://localhost:2222/stop"}}
  ]
}
```

## 4 Endpoint Catalog

The service exposes the following HTTP endpoints.

Method	Path	Description
GET	/	Machine-readable service catalog
GET	/home	Formatted documentation for available routes
GET	/get_key/:key_id	Return the value for the provided numeric key, caching it if not present in cache
PATCH	/bulk_query	Retrieve multiple keys in one request; missing keys noted in response; always returns success with errors appended
POST	/insert/:key/:value	Insert a key/value pair; conflicts return 409 with existing value; write-through to cache and persistence
POST	/bulk_update	Transactional pipeline for create/get/insert/update; rolls back on failure and returns failure response
DELETE	/delete_key/:key	Remove the provided key from both cache and persistence layer
PUT	/update_key/:key/:value	Update an existing key with a new value in both cache and persistence layer
GET	/health	Report service health and uptime
GET	/metrics	Expose cache metrics including hit/miss counts
GET	/stop	Gracefully stop the server (testing/debug only; not for production)

## 5 Screenshots

```

{
  "description": "HTTP-accessible cache with inline persistence adapter hooks",
  "links": {
    "health": "/health",
    "home": "/home",
    "metrics": "/metrics"
  },
  "routes": [
    {
      "description": "Machine-readable service catalog",
      "method": "GET",
      "path": "/"
    },
    {
      "description": "Formatted documentation for available routes",
      "method": "GET",
      "path": "/home"
    },
    {
      "description": "Return the value for the provided numeric key caching it if not present in cache",
      "method": "GET",
      "path": "/get_key/:key_id"
    },
    {
      "description": "Retrieve multiple keys in one request; missing keys noted in response, always return success response with error appended to the response",
      "method": "PATCH",
      "path": "/bulk_query"
    },
    {
      "description": "Insert a key/value pair; conflicts return 409 with existing value, writes both to cache and persistence layer (note that we are using write-through type of cache)",
      "method": "POST",
      "path": "/insert/:key/:value"
    },
    {
      "description": "Transactional Commit pipeline for create/get/insert/update operations, rollbacks in case of failure and returns failure response",
      "method": "POST",
      "path": "/bulk_update"
    },
    {
      "description": "Remove the provided key from both the cache and persistence layer",
      "method": "DELETE",
      "path": "/delete_key/:key"
    },
    {
      "description": "Update an existing key with a new value to both the cache and persistence layer",
      "method": "PUT",
      "path": "/update_key/:key/:value"
    },
    {
      "description": "Report service health and uptime",
      "method": "GET",
      "path": "/health"
    },
    {
      "description": "Expose cache metrics including hit/miss counts",
      "method": "GET",
      "path": "/metrics"
    },
    {
      "description": "Gracefully stop the server (testing/debug only), shouldn't be available in prod environment",
      "method": "GET",
      "path": "/stop"
    }
  ],
  "service": "Persistent Key Value Store",
  "version": "1.0"
}

```

Figure 2: Root endpoint (service catalog)

**PERSISTENT KEY VALUE STORE**

Fast HTTP interface for managing key/value pairs with persistence layer in the backend (PostgreSQL)

[View JSON Service Catalog](#)

Available Routes		
METHOD	ROUTE	DESCRIPTION
GET	/	Machine-readable service catalog
GET	/home	Formatted documentation for available routes
GET	/get_key/:key_id	Return the cached value for the provided numeric key
POST	/bulk_query	Retrieve multiple keys in one request; missing keys noted in response
POST	/insert/:key/:value	Insert a key/value pair; conflicts return 409 with existing value
PATCH	/bulk_update	Partial commit pipeline for insert/update operations
DELETE	/delete_key/:key	Remove the provided key from the cache
PUT	/update_key/:key/:value	Update an existing key with a new value
GET	/health	Report service health and uptime
GET	/metrics	Expose cache metrics including hit/miss counts
GET	/stop	Gracefully stop the server (testing/debug only)

Copyright © Pujith Sri Kumar Korlepara. All rights reserved.

Figure 3: Home HTML endpoint

The screenshot shows the Postman interface with a history panel on the left listing various API requests. In the main center, a GET request is being viewed for the URL `http://localhost:2222/get_key/22`. The 'Query Params' section shows a key named 'Key' with a value of 'Key'. The 'Body' tab displays a JSON response:

```

1
2   "found": true,
3   "query_key": "22",
4   "value": "pujith"
5

```

The status bar at the bottom indicates a 200 OK status with a time of 2 ms and a size of 151 B.

**Figure 4:** Cache hit on `get_key`

The screenshot shows the Postman interface with a history panel on the left listing various API requests. In the main center, a GET request is being viewed for the URL `http://localhost:2222/get_key/2`. The 'Query Params' section shows a key named 'Key' with a value of 'Key'. The 'Body' tab displays a JSON response:

```

1
2   "cache_populated": true,
3   "found": true,
4   "query_key": "2",
5   "source": "persistence",
6   "value": "bar"
7

```

The status bar at the bottom indicates a 200 OK status with a time of 4 ms and a size of 193 B.

**Figure 5:** Hydrating uncached key from persistence

The screenshot shows the Postman interface with a history panel on the left and a request panel on the right. The history panel lists various API calls, including several bulk operations and a specific GET request for 'get\_key/222'. The request panel for 'get\_key/222' has a 'Params' tab selected, showing a 'Key' parameter set to 'Key'. The response body is displayed in JSON format:

```

1   {
2     "found": false,
3     "persisted": true,
4     "query_key": "222",
5     "reason": "key not present in cache or persistence"
6   }

```

The status bar at the bottom indicates a 404 Not Found error.

**Figure 6:** Key not found scenario

The screenshot shows the Postman interface with a history panel on the left and a request panel on the right. The history panel lists various API calls, including a POST request for 'insert/22/pujith'. The request panel for 'insert/22/pujith' has a 'Params' tab selected, showing a 'Key' parameter set to 'Key'. The response body is displayed in JSON format:

```

1   {
2     "created": true,
3     "key": "22",
4     "persisted": true,
5     "value": "pujith"
6   }

```

The status bar at the bottom indicates a 201 Created response.

**Figure 7:** Successful clean insert

The screenshot shows the Postman interface with a request to `http://localhost:2222/insert/22/pujith`. The response status is 409 Conflict, indicating that the key already exists. The response body is:

```

1 2
2   "error": "key exists",
3   "existing_value": "pujith",
4   "key": "22",
5   "reason": "insert rejected because key already exists",
6   "value": "pujith"
7

```

**Figure 8:** Insert conflict (409)

The screenshot shows the Postman interface with a request to `http://localhost:2222/update_key/250/elonmusk`. The response status is 200 OK, indicating success. The response body is:

```

1 2
2   "key": "250",
3   "persisted": true,
4   "persistence_checked": true,
5   "updated": true,
6   "value": "elonmusk"
7

```

**Figure 9:** Successful update operation

The screenshot shows a Postman request to `http://localhost:2222/update_key/250/buffalo` using the PUT method. The 'Body' tab is selected, showing a JSON payload with a single field 'key': '250'. The response status is 404 Not Found, with a response body containing an error message: "error": "not found", "key": "250", "persistence\_checked": true, "reason": "key not present in cache or persistence", "value": "elomusk".

**Figure 10:** Update when key absent (404)

The screenshot shows a Postman request to `http://localhost:2222/delete_key/10` using the DELETE method. The 'Body' tab is selected, showing a JSON payload with a single field 'key': '10'. The response status is 204 No Content.

**Figure 11:** Successful deletion (204)

The screenshot shows the Postman interface with a history panel on the left containing various API requests. A specific DELETE request is selected in the center:

```
DELETE http://localhost:2222/delete_key/2532
```

The request details show the URL and a query parameter table:

Key	Value
Key	2532

The response tab shows the following JSON output:

```

1 | {
2 |   "error": "not found",
3 |   "key": "2532",
4 |   "persistence_checked": true,
5 |   "reason": "key not present in cache or persistence"
6 | }

```

**Figure 12:** Deletion when key not found

The screenshot shows the Postman interface with a history panel on the left containing various API requests. A specific PATCH request is selected in the center:

```
PATCH http://localhost:2222/bulk_query
```

The request details show the URL and a JSON body:

```

1 | [
2 |   {
3 |     "data": [
4 |       250,
5 |       350,
6 |       1,
7 |       100,
8 |       "invalid type",
9 |       22,
10 |       2
11 |     ]
12 |   }
]

```

The response tab shows the following JSON output:

```

{"endpoint": "bulk_query", "results": [{"cache_populated": true, "found": true, "index": 0, "input": 250, "key": 250, "persistence_checked": true, "reason": "value hydrated from persistence", "source": "persistence", "status": "hit_persistence", "value": "buffalo"}, {"found": false, "index": 1, "input": 10, "key": 10, "persistence_checked": true, "reason": "key not present in cache or persistence", "status": "miss", "value": null}, {"found": false, "index": 2, "input": 350, "key": 350, "persistence_checked": true, "reason": "key not present in cache or persistence", "status": "miss", "value": null}, {"found": false, "index": 3, "input": 1, "key": 1, "persistence_checked": true, "reason": "key not present in cache or persistence", "status": "miss", "value": null}, {"found": false, "index": 4, "input": 100, "key": 100, "persistence_checked": true, "reason": "key not present in cache or persistence", "status": "miss", "value": null}, {"found": false, "index": 5, "input": "invalid type", "persistence_checked": true, "reason": "value hydrated from persistence", "source": "persistence", "status": "hit_persistence", "value": "invalid type"}, {"found": false, "index": 6, "input": 22, "key": 22, "persistence_checked": true, "reason": "key not present in cache or persistence", "status": "miss", "value": null}, {"found": false, "index": 7, "input": 2, "key": 2, "persistence_checked": true, "reason": "key not present in cache or persistence", "status": "miss", "value": null}], "misses": 3, "requested": 8, "top_level_errors": 0, "type_mismatch": 13}

```

**Figure 13:** Bulk query results with mixed statuses

The screenshot shows the Postman interface with a successful bulk update transaction. The request URL is `http://localhost:2222/bulk_update`. The request body is a JSON object with the following operations:

```

1 {
2   "operations": [
3     {
4       "operation": "insert",
5       "key": 25,
6       "value": "elephant"
7     },
8     {
9       "operation": "update",
10      "key": 1,
11      "value": "Lion"
12    },
13    {
14      "operation": "get",
15      "key": 1
16    },
17    {
18      "operation": "delete",
19      "key": 25
20    }
21  ]
22 }

```

The response status is 200 OK, and the response body contains the results of the bulk update operation.

**Figure 14:** Bulk update transaction success report

The screenshot shows the Postman interface with a failed bulk update transaction due to a key not present error. The request URL is `http://localhost:2222/bulk_update`. The request body is the same as in Figure 14. The response status is 200 OK, but the response body indicates a failure:

```

{
  "endpoint": "bulk_update",
  "reason": "key not present",
  "results": [
    {
      "index": 0,
      "input": {
        "key": 25,
        "operation": "insert",
        "value": "elephant"
      },
      "output": {
        "key": 25,
        "operation": "insert",
        "status": "ok"
      }
    },
    {
      "index": 1,
      "input": {
        "key": 1,
        "operation": "update",
        "value": "Lion"
      },
      "output": {
        "key": 1,
        "operation": "update",
        "status": "ok"
      }
    },
    {
      "index": 2,
      "input": {
        "key": 1,
        "operation": "get"
      },
      "output": {
        "key": 1,
        "value": "Lion"
      }
    },
    {
      "index": 3,
      "input": {
        "key": 25,
        "operation": "delete"
      },
      "output": {
        "key": 25,
        "operation": "delete",
        "status": "ok"
      }
    }
  ],
  "error": "key not present"
}

```

The response body also includes details about the failed operation and the transaction mode.

**Figure 15:** Bulk update transaction failure and rollback

The screenshot shows the Postman interface with the following details:

- History:** Shows a list of recent requests, including various GET, PUT, and POST operations on the host 127.0.0.1:2222.
- Request:**
  - Method:** GET
  - URL:** http://localhost:2222/metrics
  - Headers:** Authorization, Headers (6), Body, Pre-request Script, Tests, Settings
  - Query Params:** Key, Value
  - Body:** Status: 200 OK, Time: 3 ms, Size: 163 B, Save Response
- Response:**

```

1 1
2   "bytes": 245,
3   "entries": 4,
4   "evictions": 0,
5   "hits": 9,
6   "misses": 11
7

```

Figure 16: Metrics endpoint showing cache statistics

The screenshot shows the Postman interface with the following details:

- History:** Shows a list of recent requests, including various GET, PUT, and POST operations on the host 127.0.0.1:2222.
- Request:**
  - Method:** GET
  - URL:** http://localhost:2222/health
  - Headers:** Authorization, Headers (6), Body, Pre-request Script, Tests, Settings
  - Query Params:** Key, Value
  - Body:** Status: 200 OK, Time: 2 ms, Size: 138 B, Save Response
- Response:**

```

1 1
2   "status": "ok",
3   "uptime_ms": 1207725
4

```

Figure 17: Health endpoint with uptime

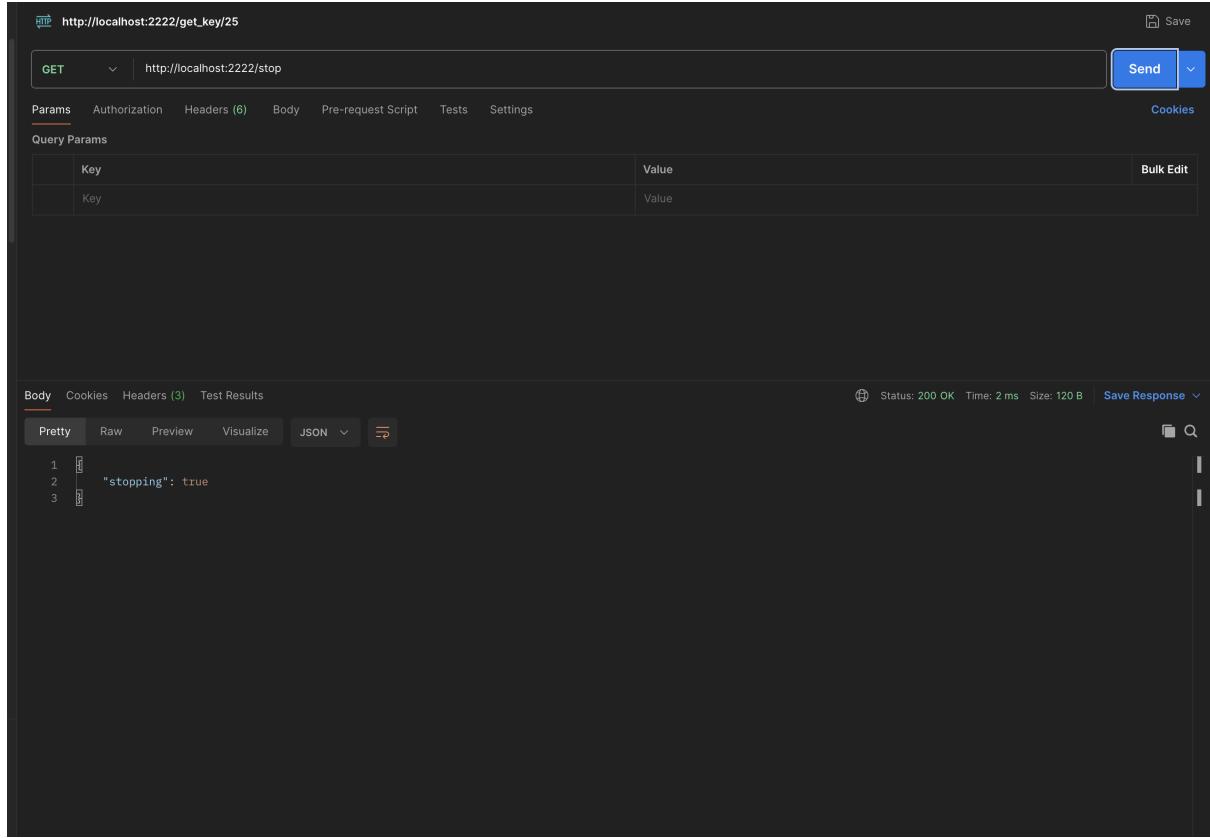


Figure 18: Stop endpoint (dev only)

```

o + persistent-key-value-store git:(feature/upgrade-system-to-use-http-server) ✘ ./kv_server.out --json-logs
{"cache_policy":"LRU","db_connection_status":"ok","json_logging_enabled":true,"listen":{"host":"localhost","port":2222}, "ready":true,"start_time_ms":1762528983299,"type":"start up"}
{"body_bytes":131,"method":"PATCH","path":"/bulk_query","path_param_count":0,"type":"request"}
{"body_bytes":150,"content_type":"application/json","duration_ms":4,"reason":"ok","status":200,"type":"response"}
{"body_bytes":150,"content_type":"application/json","duration_ms":4,"reason":"ok","status":200,"type":"response"}
{"body_bytes":688,"content_type":"application/json","duration_ms":3,"reason":"ok","status":200,"type":"response"}
{"body_bytes":475,"method":"POST","path":"/bulk_update","path_param_count":0,"type":"request"}
{"body_bytes":762,"content_type":"application/json","duration_ms":3,"reason":"ok","reason_detail":"key not present","status":200,"type":"response"}
{"body_bytes":505,"method":"POST","path":"/bulk_update","path_param_count":0,"type":"request"}
{"body_bytes":778,"content_type":"application/json","duration_ms":3,"reason":"ok","reason_detail":"key not present","status":200,"type":"response"}
{"body_bytes":505,"method":"POST","path":"/bulk_update","path_param_count":0,"type":"request"}
{"body_bytes":720,"content_type":"application/json","duration_ms":2,"reason":"ok","status":200,"type":"response"}
{"body_bytes":0,"method":"GET","path":"/get_key/25","path_param_count":1,"path_params":{"key_id":"25"}, "type":"request"}
{"body_bytes":47,"content_type":"application/json","duration_ms":0,"reason":"ok","status":200,"type":"response"}
{"body_bytes":408,"method":"POST","path":"/bulk_update","path_param_count":0,"type":"request"}
{"body_bytes":397,"content_type":"application/json","duration_ms":0,"reason":"ok","reason_detail":"one or more operations were invalid","status":200,"type":"response"}
{"body_bytes":22,"method":"DELETE","path":"/delete_key/2","path_param_count":1,"path_params":{"key":2}, "type":"request"}
{"body_bytes":0,"content_type":"application/json","duration_ms":1,"reason":"deleted","status":204,"type":"response"}
{"body_bytes":22,"method":"DELETE","path":"/delete_key/25","path_param_count":1,"path_params":{"key":25}, "type":"request"}
{"body_bytes":0,"content_type":"application/json","duration_ms":0,"reason":"deleted","status":204,"type":"response"}
{"body_bytes":22,"method":"DELETE","path":"/delete_key/2532","path_param_count":1,"path_params":{"key":2532}, "type":"request"}
{"body_bytes":112,"content_type":"application/json","duration_ms":0,"reason":"not_found","reason_detail":"key not present in cache or persistence","status":404,"type":"response"}
{"body_bytes":22,"method":"DELETE","path":"/delete_key/2","path_param_count":1,"path_params":{"key":2}, "type":"request"}
{"body_bytes":109,"content_type":"application/json","duration_ms":0,"reason":"not_found","reason_detail":"key not present in cache or persistence","status":404,"type":"response"}
{"body_bytes":22,"method":"DELETE","path":"/delete_key/22","path_param_count":1,"path_params":{"key":22}, "type":"request"}
{"body_bytes":0,"content_type":"application/json","duration_ms":0,"reason":"deleted","status":204,"type":"response"}
{"body_bytes":0,"method":"GET","path":"/get_key/2","path_param_count":1,"path_params":{"key_id":2}, "type":"request"}
{"body_bytes":109,"content_type":"application/json","duration_ms":0,"reason":"not_found","reason_detail":"key not present in cache or persistence","status":404,"type":"response"}
{"body_bytes":0,"method":"POST","path":"/insert/22/puith","path_param_count":2,"path_params":{"key":22,"value":"puith"}, "type":"request"}
{"body_bytes":61,"content_type":"application/json","duration_ms":1,"reason":"created","status":201,"type":"response"}
{"body_bytes":108,"method":"POST","path":"/insert/1/foo","path_param_count":2,"path_params":{"key":1,"value": "foo"}, "type":"request"}
{"body_bytes":124,"content_type":"application/json","duration_ms":0,"reason":"conflict_key_exists","reason_detail":"insert rejected because key already exists","status":409,"type":"response"}
{"body_bytes":0,"method":"POST","path":"/insert/2/bar","path_param_count":2,"path_params":{"key":2,"value": "bar"}, "type":"request"}
{"body_bytes":57,"content_type":"application/json","duration_ms":0,"reason":"created","status":201,"type":"response"}
{"body_bytes":22,"method":"PUT","path":"/update_key/250/elonmusk","path_param_count":2,"path_params":{"key":250,"value": "elonmusk"}, "type":"request"}
{"body_bytes":91,"content_type":"application/json","duration_ms":1,"reason":"updated","status":200,"type":"response"}
{"body_bytes":22,"method":"PUT","path":"/update_key/250/elonmusk","path_param_count":2,"path_params":{"key":250,"value": "elonmusk"}, "type":"request"}
{"body_bytes":91,"content_type":"application/json","duration_ms":1,"reason":"updated","status":200,"type":"response"}
 {"body_bytes":22,"method":"PUT","path":"/update_key/250/elonmusk","path_param_count":2,"path_params":{"key":250,"value": "elonmusk"}, "type":"request"}
 {"body_bytes":133,"content_type":"application/json","duration_ms":0,"reason":"not_found","reason_detail":"key not present in cache or persistence","status":404,"type":"response"}
 {"body_bytes":22,"method":"PUT","path":"/update_key/250/elonmusk","path_param_count":2,"path_params":{"key":250,"value": "elonmusk"}, "type":"request"}
 {"body_bytes":91,"content_type":"application/json","duration_ms":1,"reason":"updated","status":200,"type":"response"}
 {"body_bytes":22,"method":"PUT","path":"/update_key/250/elephant","path_param_count":2,"path_params":{"key":250,"value": "elephant"}, "type":"request"}
 {"body_bytes":0,"method":"GET","path":"/health","path_param_count":0,"type":"request"}
 {"body_bytes":0,"method":"GET","path":"/health","path_param_count":0,"type":"request"}
 {"body_bytes":0,"content_type":"application/json","duration_ms":0,"reason":"ok","status":200,"type":"response"}
 {"body_bytes":0,"content_type":"application/json","duration_ms":0,"reason":"ok","status":200,"type":"response"}
 {"body_bytes":60,"content_type":"application/json","duration_ms":0,"reason": "ok","status":200,"type":"response"}

```

Figure 19: Structured JSON logging output sample

```

- persistent-key-value-store git:(feature/upgrade-system-to-use-http-server) ✘ ./kv_server.out --policy=IFO --json-logs
{"cache_policy": "IFO", "db_connection_status": "ok", "json_logging_enabled": true, "listen": {"host": "localhost", "port": 2222}, "ready": true, "start_time_ms": 1762530921896, "type": "start-up"}
{"body_bytes": 585, "method": "POST", "path": "/bulk_update", "path_param_count": 0, "type": "request"}
{"bytes": 777, "content_type": "application/json", "duration_ms": 6, "reason": "ok", "reason_detail": "key not present", "status": 200, "type": "response"}
{"body_bytes": 0, "method": "GET", "path": "/get_key/25", "path_param_count": 1, "path_params": {"key_id": "25"}, "type": "request"}
{"bytes": 110, "content_type": "application/json", "duration_ms": 0, "reason": "not found", "reason_detail": "key not present in cache or persistence", "status": 404, "type": "response"}
{"body_bytes": 588, "method": "POST", "path": "/bulk_update", "path_param_count": 0, "type": "request"}
{"bytes": 360, "content_type": "application/json", "duration_ms": 1, "reason": "ok", "reason_detail": "key not present", "status": 200, "type": "response"}
{"body_bytes": 0, "method": "GET", "path": "/get_key/25", "path_param_count": 1, "path_params": {"key_id": "25"}, "type": "request"}
{"bytes": 110, "content_type": "application/json", "duration_ms": 0, "reason": "not found", "reason_detail": "key not present in cache or persistence", "status": 404, "type": "response"}
{"body_bytes": 63, "content_type": "application/json", "duration_ms": 1, "reason": "created", "status": 201, "type": "response"}
{"body_bytes": 0, "method": "GET", "path": "/get_key/25", "path_param_count": 1, "path_params": {"key_id": "25"}, "type": "request"}
{"bytes": 58, "content_type": "application/json", "duration_ms": 0, "reason": "ok", "status": 200, "type": "response"}
{"body_bytes": 588, "method": "POST", "path": "/bulk_update", "path_param_count": 0, "type": "request"}
{"bytes": 780, "content_type": "application/json", "duration_ms": 2, "reason": "ok", "reason_detail": "key not present", "status": 200, "type": "response"}
{"body_bytes": 0, "method": "GET", "path": "/get_key/25", "path_param_count": 1, "path_params": {"key_id": "25"}, "type": "request"}
{"bytes": 58, "content_type": "application/json", "duration_ms": 0, "reason": "ok", "status": 200, "type": "response"}
{"bytes": 780, "content_type": "application/json", "duration_ms": 3, "reason": "ok", "reason_detail": "key not present", "status": 200, "type": "response"}
{"body_bytes": 400, "method": "POST", "path": "/bulk_update", "path_param_count": 0, "type": "request"}
{"bytes": 610, "content_type": "application/json", "duration_ms": 3, "reason": "ok", "status": 200, "type": "response"}
{"body_bytes": 0, "method": "GET", "path": "/get_key/25", "path_param_count": 1, "path_params": {"key_id": "25"}, "type": "request"}
{"bytes": 110, "content_type": "application/json", "duration_ms": 0, "reason": "not found", "reason_detail": "key not present in cache or persistence", "status": 404, "type": "response"}
{"body_bytes": 0, "method": "GET", "path": "/stop", "path_param_count": 0, "type": "request"}
{"bytes": 17, "content_type": "application/json", "duration_ms": 0, "reason": "ok", "status": 200, "type": "response"}

```

**Figure 20:** Server startup with custom cache eviction policy.

## 6 Credits

Third-party libraries: [nlohmann/json](#) and [cpp-httplib](#). Licensed under MIT (see project LICENSE).