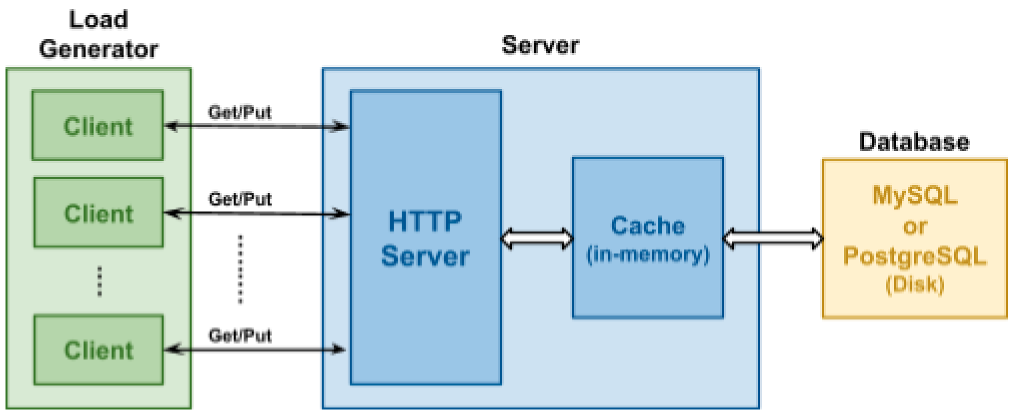
1. GitHub Repository Link

Link : `https://github.com/ashubirla/DECS_Project_24M0752.git`

2. System Description

This project is a functional, multi-tier HTTP-based Key-Value (KV) server built in C++ as required. The system is designed as a two-tier server (HTTP frontend + in-memory cache) connected to a persistent standalone database.

Architecture:



Phase 1:

- HTTP Server (Tier 1): The server is built using the `cpp-httplib` library. It uses a thread pool to handle concurrent client connections.
- In-Memory Cache (Tier 2): The server maintains an in-memory cache (`map<string, string> cache`) to fasten read requests.
- Database: The system is connected to a standalone MySQL database (`kv_db`) that persists all key-value pairs.
- Load Generator: A basic multi-threaded client is implemented to demonstrate functional correctness and concurrent request handling.

3. Functional Logic (Phase 1)

The system correctly implements the required logic for all operations:

- `create` (POST/PUT): When a `create` request is received, the server writes the key-value pair to *both* the persistent MySQL database and the in-memory cache.
- `read` (GET): The server first checks the in-memory cache.
    - Cache Hit: If the key is found, the value is returned directly from memory.

    - Cache Miss: If the key is not in the cache, the server fetches the value from the MySQL database, inserts it into the cache for future requests, and then returns the value to the client.
- `delete` (DELETE): The server deletes the key-value pair from *both* the MySQL database and the in-memory cache to ensure data consistency.