Design and Implementation of an LRU Cache in Java

Mini Project Report

Submitted by:
Maddepuri Ganesh

Department: Information Technology

College: Sreenidhi Institute of Science and Technology

Guide: [Project Guide Name]

GitHub Repository: https://github.com/Mganesh-creator/LRUCACHEPROJECT

Abstract

This mini project focuses on the design and implementation of a Least Recently Used (LRU) Cache in Java. The project demonstrates how efficient caching mechanisms can improve data retrieval speed and reduce redundant computations. By combining the principles of a HashMap and a Doubly Linked List, this implementation ensures constant-time complexity (O(1)) for both insertion and lookup operations. The project also emphasizes practical use cases such as memory management, database query caching, and web caching systems.

Objective

The primary objective of this project is to develop a Java-based implementation of the Least Recently Used (LRU) caching technique. This technique ensures that the most recently accessed data is retained while the least recently accessed data is removed once the cache reaches its capacity limit. The project aims to provide a clear understanding of efficient cache replacement strategies and the importance of time complexity in data structure design.

System Design and Implementation

The system uses two core data structures: a HashMap and a Doubly Linked List. The HashMap provides O(1) access to cached elements by key, while the Doubly Linked List maintains the order of access, ensuring that the least recently used element can be efficiently removed when the cache is full. The Java implementation includes two main files: **LRUCache.java** (core logic) and **Main.java** (driver/test file). This approach demonstrates the use of linked data structures and hash-based access patterns to achieve efficient cache behavior.

Sample Output

Below is an example of the program’s console output during execution:

Put(1, 10)
Put(2, 20)
Get(1) $\rightarrow$ 10
Put(3, 30) $\rightarrow$ Evicts key 2
Get(2) $\rightarrow$ -1 (Not found)

Conclusion and Future Scope

This project successfully demonstrates the design and implementation of an efficient caching mechanism using Java. The LRU Cache efficiently handles data access patterns while maintaining low time complexity. It can be further extended to include advanced features such as thread-safety, dynamic resizing, and integration with databases or web servers. The project serves as a strong

foundation for understanding caching systems and real-world performance optimization techniques.

References

1. GeeksforGeeks - LRU Cache Implementation in Java
2. LeetCode Problem #146 - LRU Cache
3. Java Documentation (Oracle)