# CS591A1 Spring 2019 - Systems Project

**Title:** *Implementation a LSM-Tree based key-value store*

**Background**: Log-structured merge-tree (LSM-trees) [1, 2] are one of the most commonly used data structures for persistent storage of key-value entries. LSM-tree-based storages are in use in several modern key-value stores including RocksDB at Facebook, LevelDB and BigTable at Google, bLSM and cLSM at Yahoo!, Cassandra and HBase at Apache, and so on. LSM-trees store data in the disk as immutable logs (also known as sorted sequence tables (SSTs)), which are maintained in hierarchical levels of increasing capacity. To bound the number of logs that a lookup has to probe, LSM-trees merge logs of similar sizes. The two possible merging strategies are (i) leveling (optimized for lookups) and (ii) tiering (optimized for updates).

**Objective**: The objective of the project is to implement a LSM-tree using only a single process thread. The workflow for this is as the following.

(a) Review LSM-tree literature to understand the principles and operations supported in an LSM-tree
(b) Implement an LSM-tree (vanilla implementation) using only a single process thread[1] for both merging strategies — leveling and tiering
(c) Design APIs to support all the standard LSM-tree operations (such as point lookup, range lookup, insert, update, and delete)

[1] Patrick E. O'Neil, Edward Cheng, Dieter Gawlick, Elizabeth J. O'Neil. **The Log-Structured Merge-Tree (LSM-Tree)**. Acta Inf. 33(4): 351-385 (1996)
[2] Niv Dayan, Manos Athanassoulis, Stratos Idreos. **Monkey: Optimal Navigable Key-Value Store**. SIGMOD Conference 2017: 79-94

[1] In many LSM-tree-based storages (such as RocksDB, LevelDB) merging usually takes place as a multi-threaded background process in order to enhance the update performance. However, in this project, the goal is to implement LSM-trees using only a single thread.