Architectural document

1. Designing the Application Architecture

Define the architecture of the task management application:

The application will consist of multiple nodes, each running a copy of the task management service.

Raft consensus algorithm will be used to ensure consistency and fault tolerance among these nodes.

The architecture will include components for client communication, Raft protocol implementation, and database interaction.

Determine the role of each node in the Raft consensus algorithm:

Leader: Responsible for coordinating operations and replicating log entries to other nodes.

Follower: Passive nodes that replicate the leader's log entries and respond to requests from the leader.

Candidate: Nodes that are attempting to become the leader in an election.

Plan the schema for storing task data in MySQL:

Table Structure:

Task table with columns for task ID, title, description, deadline, status, etc.

Relationships:

No complex relationships are anticipated, but foreign keys may be used if necessary for future expansion.

Indexing:

Indexing may be applied to columns frequently used for querying, such as task ID or deadline.

Let's break down each step further:

Define the application architecture:

Components:

Client Interface: Interface for users to interact with the task management system.

Raft Protocol Implementation: Implementation of Raft consensus algorithm for distributed coordination.

Database Interaction: Handles interaction with the MySQL database to store and retrieve task data.

Networking: Facilitates communication between nodes in the distributed system.

Determine the role of each node in Raft:

Leader:

Handles client requests, replicates log entries to followers, and maintains consistency.

Follower:

Passively replicates log entries from the leader and responds to leader requests.

Candidate:

Initiates leader election process when necessary.

Plan MySQL schema:

Task Table:

Columns: task_id, title, description, deadline, status, etc.

Primary Key: task_id

Indexing: Consider indexing on task_id, deadline, etc. depending on query patterns.

Once we have these defined, we can proceed to the implementation phase.