API CreateTopic (topic) The Create Topic API is used to create a topic by calling the node. After calling the Create Topic API, we first calculate the hash value of the topic. Hash function will return the index that refers to the Node that holds that topic. After we give that index parameter to calculate the hypercube route. We store the forwarding path into the array and include that array as a key-value into the request. Node will send the request to first node from that routing table, that node is the neighbor of the calling node. Once we forwarded it to the neighbor node, the neighbor node will deserialize it and look if it belongs to it. The destination key in the request is the final node that should accomplish the request. If the receiving node is the destination, it will check if the topic is already in its DHT. If not, it adds it to the DHT and send the acknowledgement response back. If it already exists, it sends back that topic already exist.

API DeleteTopic (topic) The Delete Topic API is used to delete a topic by calling the node. After calling the Delete Topic API, we first calculate the hash value of the topic. Hash function will return the index that refers to the Node that holds that topic. After we give that index parameter to calculate the hypercube route. We store the forwarding path into the array and include that array as a key-value into the request. Node will send the request to first node from that routing table, that node is the neighbor of the calling node. Once we forwarded it to the neighbor node, the neighbor node will deserialize it and look if it belongs to it. The destination key in the request is the final node that should accomplish the request. If the receiving node is the destination, it will check if the topic is in its DHT, it will delete and send the acknowledgement response back. If it topics does not exist, it sends back that topic is not in its DHT.

API Send (topic, message) The Send Topic API is used to delete a topic by calling the node. After calling the Send Topic API, we first calculate the hash value of the topic. Hash function will return the index that refers to the Node that holds that topic. After we give that index parameter to calculate the hypercube route. We store the forwarding path into the array and include that array as a key-value into the request. Node will send the request to first node from that routing table, that node is the neighbor of the calling node. Once we forwarded it to the neighbor node, the neighbor node will deserialize it and look if it belongs to it. The destination key in the request is the final node that should accomplish the request. If the receiving node is the destination, it will check if the topic is in its DHT. If there's such topic, it adds the message to the messages dictionary. If topic does not exist, it send back that topic does not exist to add message.

API Subscribe (topic) The Subscribe Topic API is used to delete a topic by calling the node. After calling the Subscribe Topic API, we first calculate the hash value of the topic. Hash function will return the index that refers to the Node that holds that topic. After we give that index parameter to calculate the hypercube route. We store the forwarding path into the array and include that array as a key-value into the request. Node will send the request to first node from that routing table, that node is the neighbor of the calling node. Once we forwarded it to the neighbor node, the neighbor node will deserialize it and look if it belongs to it. The destination key in the request is the final node that should accomplish the request. If the receiving node is the destination, it will check if the topic is in its DHT, then the destination node adds the calling node into the subscription list of that topic. If there's no such topic, it send back that there is no topic to subscribe.

API Pull (topic) The Pull Topic API is used to delete a topic by calling the node. After calling the Pull Topic API, we first calculate the hash value of the topic. Hash function will return the index that refers to the Node that holds that topic. After we give that index parameter to calculate the hypercube route. We store the forwarding path into the array and include that array as a key-value into the request. Node will send the request to first node from that routing table, that node is the neighbor of the calling node. Once we forwarded it to the neighbor node, the neighbor node will deserialize it and look if it belongs to it. The destination key in the request is the final node that should accomplish the request. If the receiving node is the destination,

it will check if the topic is in its DHT, if so it check if the calling node is in subscription of that topic. If it's subscribed to that topic it send back all new messages to the calling node. If it's not subscribed to the topic, it send that calling node is not subscribed to pull messages.

If all nodes that are subscribed to the topic have read messages, the garbage collection happens that removes all read message from the DHT. It also updates the readIndexes of all nodes that are subscribed to that topic.