Class Project-275

Kafka

Kafka Architecture

Who is the leader?
Who is the Leader?
Who is the leader?
Who is the Leader?

Heatbeat+data
Heatbeat+data
Heatbeat+data
Heatbeat+data

Topic Created/MessagePublished
Create Topic/Publish message
Poll

Producer 1
Broker 1(Leader)
Broker 2
Broker 3
Broker N
Consumer 1

Poll
Poll
Subscribe/GetMessages
Messages
Subscribe/GetMessages
Messages

Topic Created/MessagePublished
Create Topic/Publish message
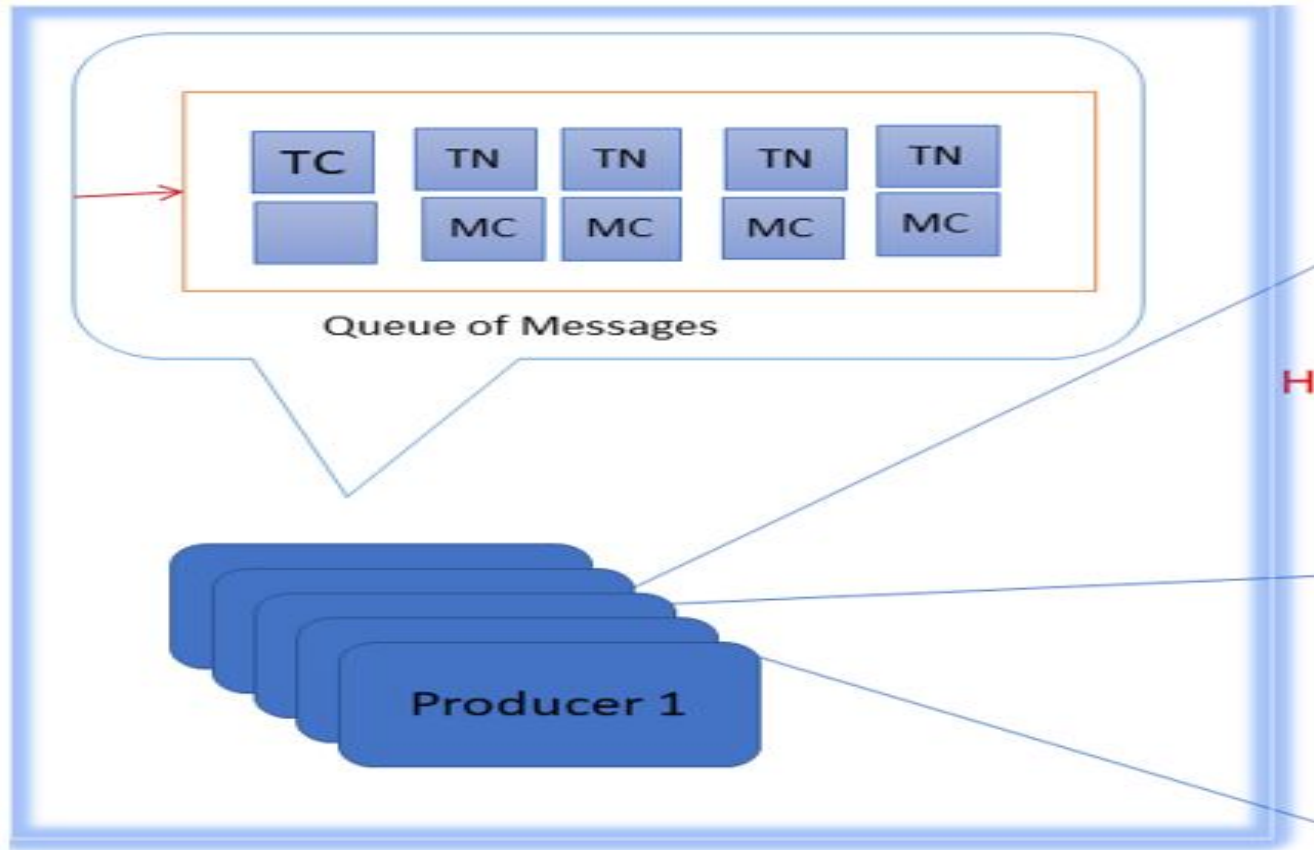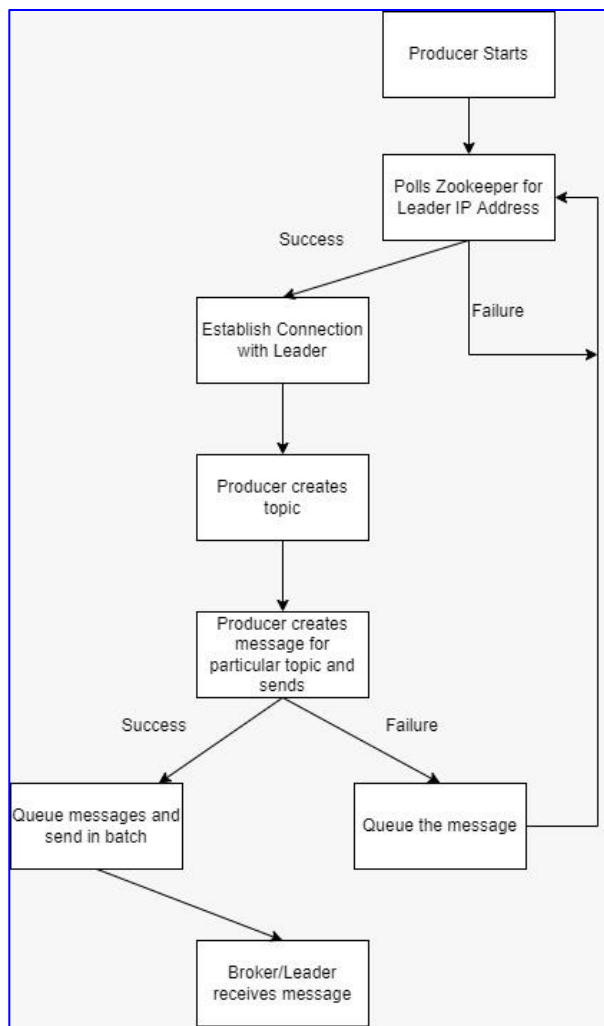
Producer N
Consumer N

Producer

Producer

- Produces messages of two types topic creation as well as message type which maps onto a specific topic.
- Producer is connected to both the leader-broker as well as the zookeeper and keeps sending messages to the leader until and unless a leader election takes place after which it continues to send messages onto the new leader.
- As of our architecture each producer maintains its own queue and sends messages in batches and queues them when unable to connect to leader.

TC–Topic creation Message
TN–Topic Name
MC–Message Content

Producer Starts

Polls Zookeeper for
Leader IP Address

Success

Failure

Establish Connection
with Leader

Producer creates
topic

Producer creates
message for
particular topic and
sends

Success

Failure

Queue messages and
send in batch

Queue the message

Broker/Leader
receives message

WORKFLOW OF PRODUCER

Broker & Consumer

Kafka Broker

- Registers itself with zookeeper and check whether it a leader.
- Receives heartbeats from zookeeper and updates the topic list if broker is not leader.
- Maintains topic list and offsets and updates it as per the heart beat information.
- Accepts create topic, produce message request from producer and subscribe topic and pull message from consumer.
- Allows replication of topic over multiple follower node.

Consumer

- Requests for Leader IP from Zookeeper.
- Subscribes for a topic.
- Sends Pull Message request to the broker after every 5 sec.
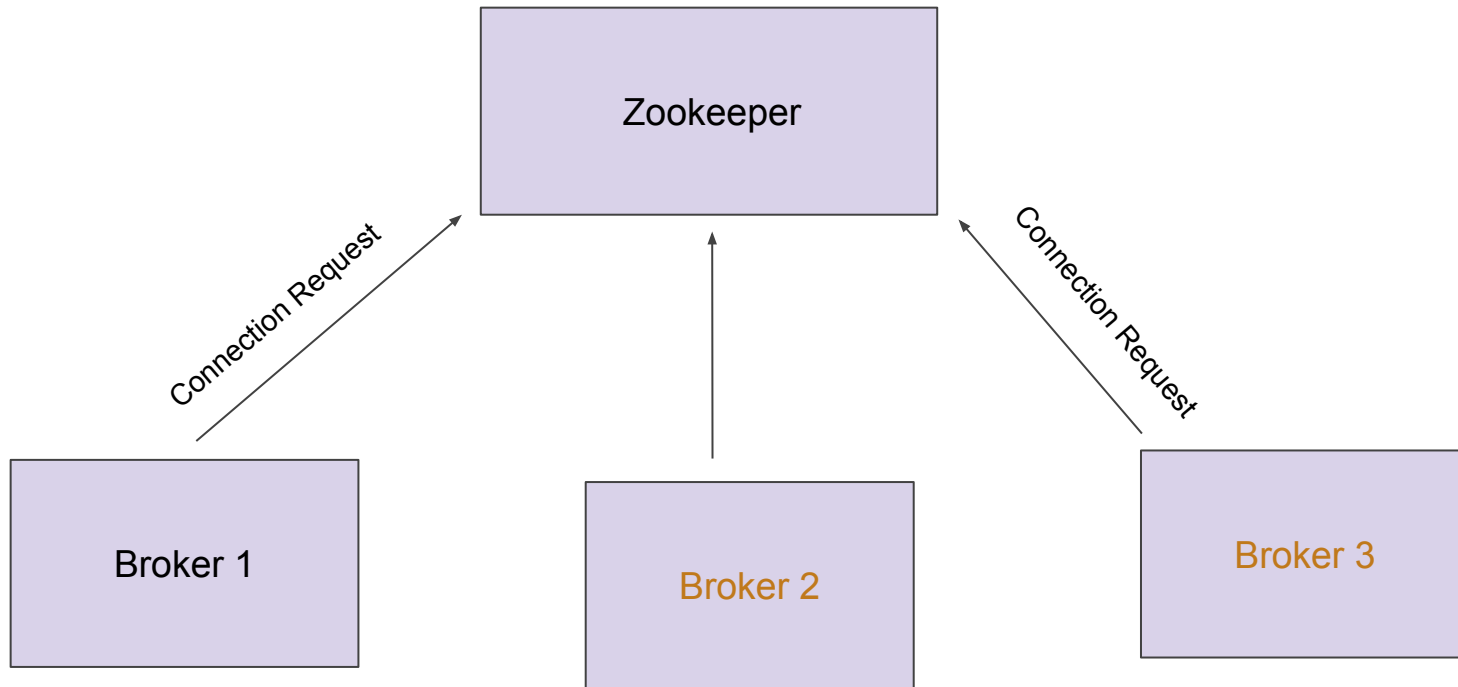- Uses Queue to put message received from broker.
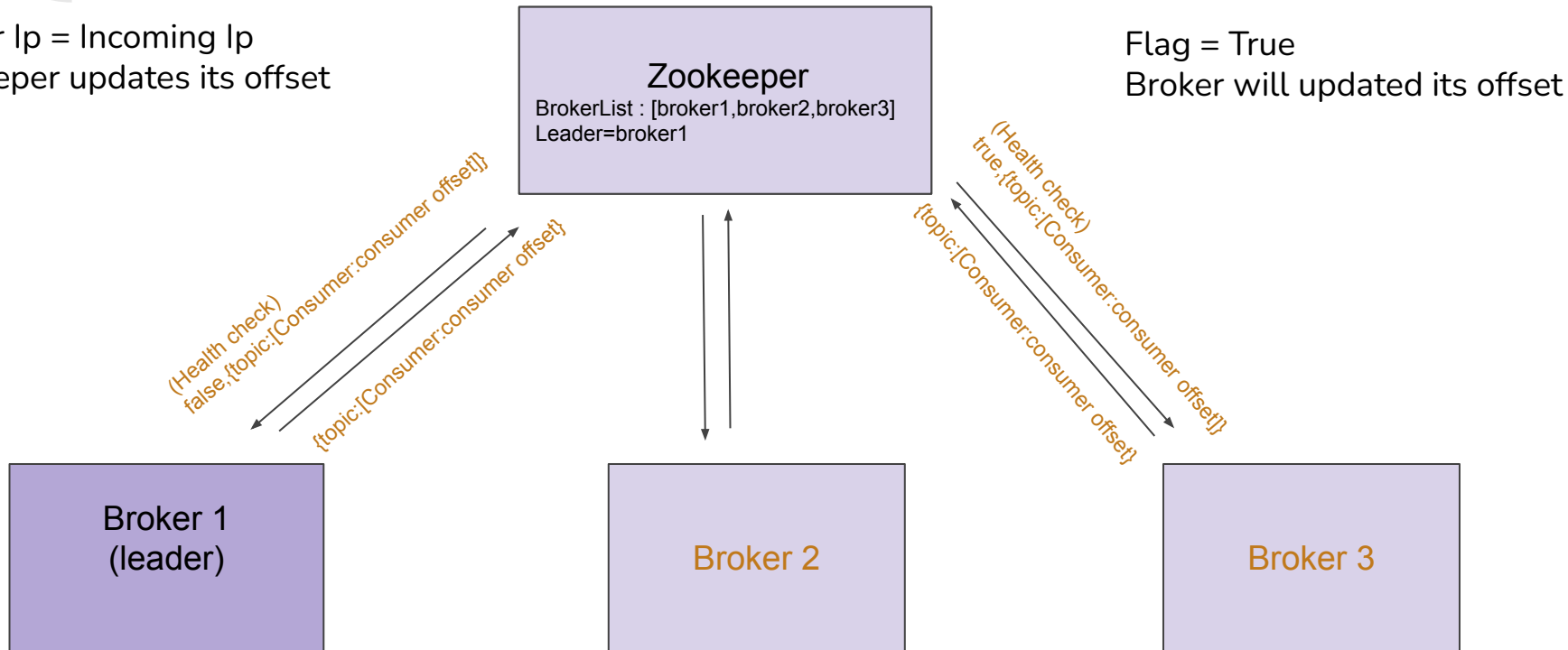
Kafka Broker
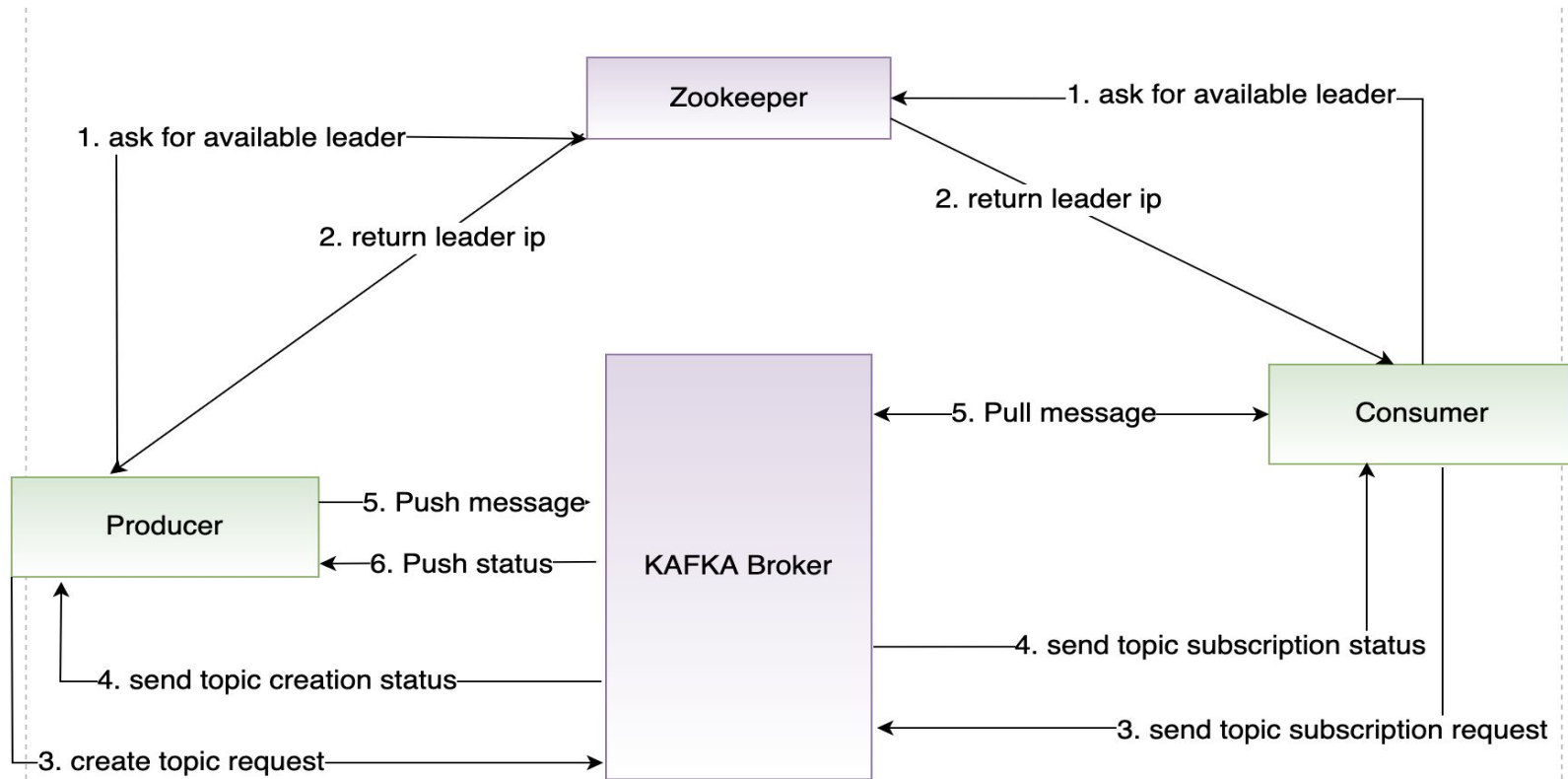
Zookeeper

System Initialization - **All brokers register their IPs with the Zookeeper. Based on connections Zookeeper decides on the Leader Broker**

Health Checks – **All Brokers will receive health checks. Follower brokers will set the offset (not the leader broker). The offset sent by the leader broker will be persisted by the zookeeper and the consecutive heartbeat will carry the updated offset**

Leader Ip = Incoming Ip
Zookeeper updates its offset

Flag = True
Broker will updated its offset

Zookeeper
BrokerList : [broker1,broker2,broker3]
Leader=broker1

(Health check)
false,{topic:[Consumer:consumer offset]}

{topic:[Consumer:consumer offset}

(Health check)
true,{topic:[Consumer:consumer offset]}

{topic:[Consumer:consumer offset}

Broker 1
(leader)

Broker 2

Broker 3

Leader Election - If the broker server doesn't respond for 3 consecutive health checks the broker is declared dead. If a follower broker goes down system continues but if leader broker goes down re-elections are triggered.

Exposing APIs - **All brokers register their IPs with the Zookeeper. Based on connections Zookeeper decides on the Leader Broker**

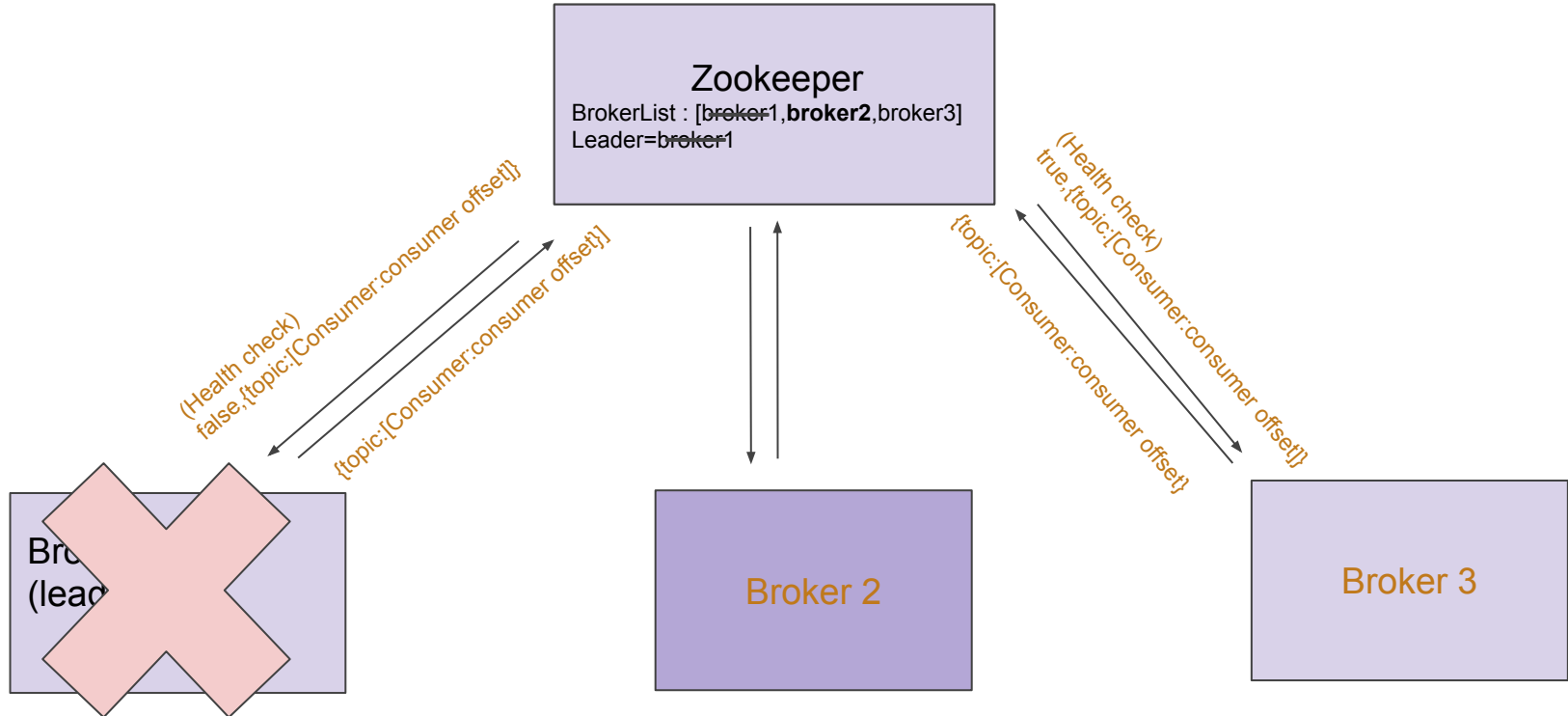| Producer	

Who should I write to? →

Ip of leader Broker ←

zookeeper.txt
Zookeeper

Who should I consumer from? ←

Ip of leader Broker →

Consumer

Which brokers are alive and we should perform replication on ?

Alive brokers

Leader Broker
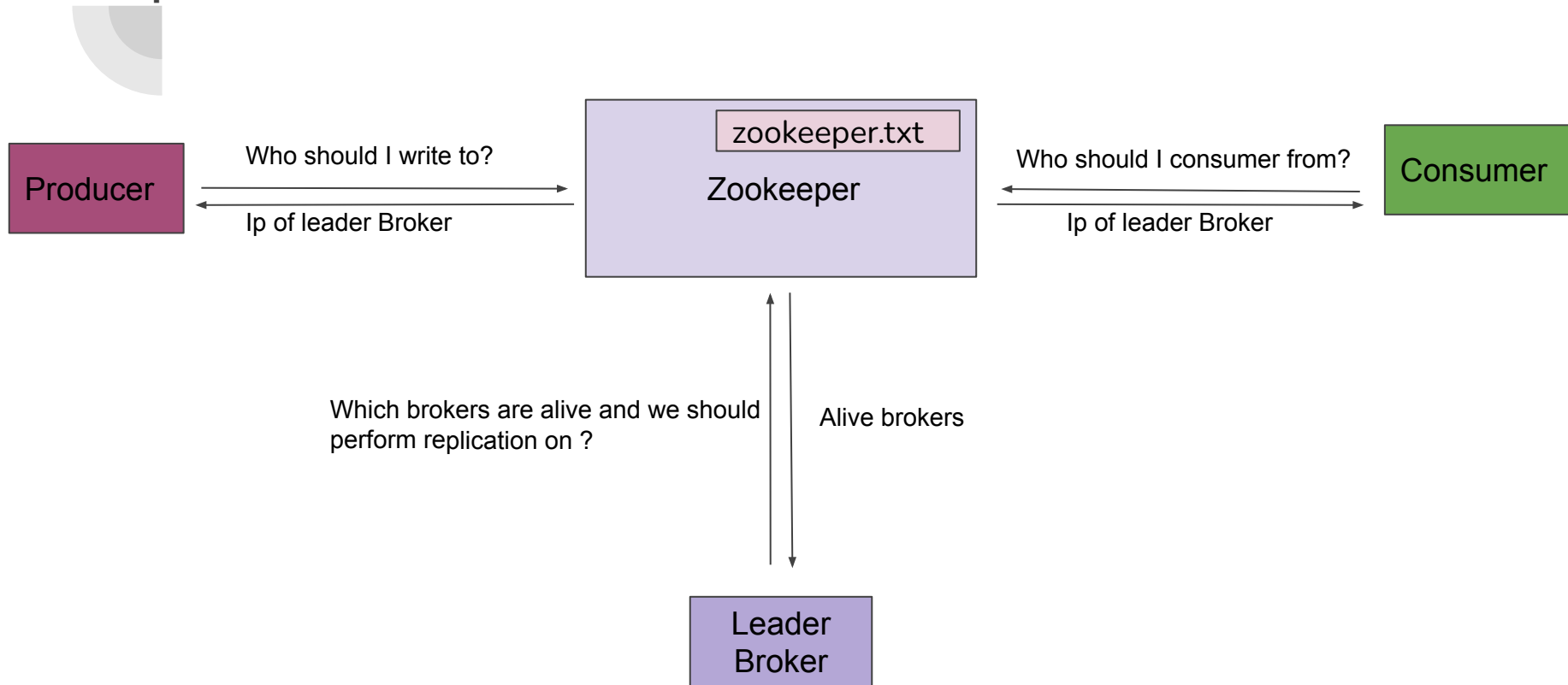
Replication Overview
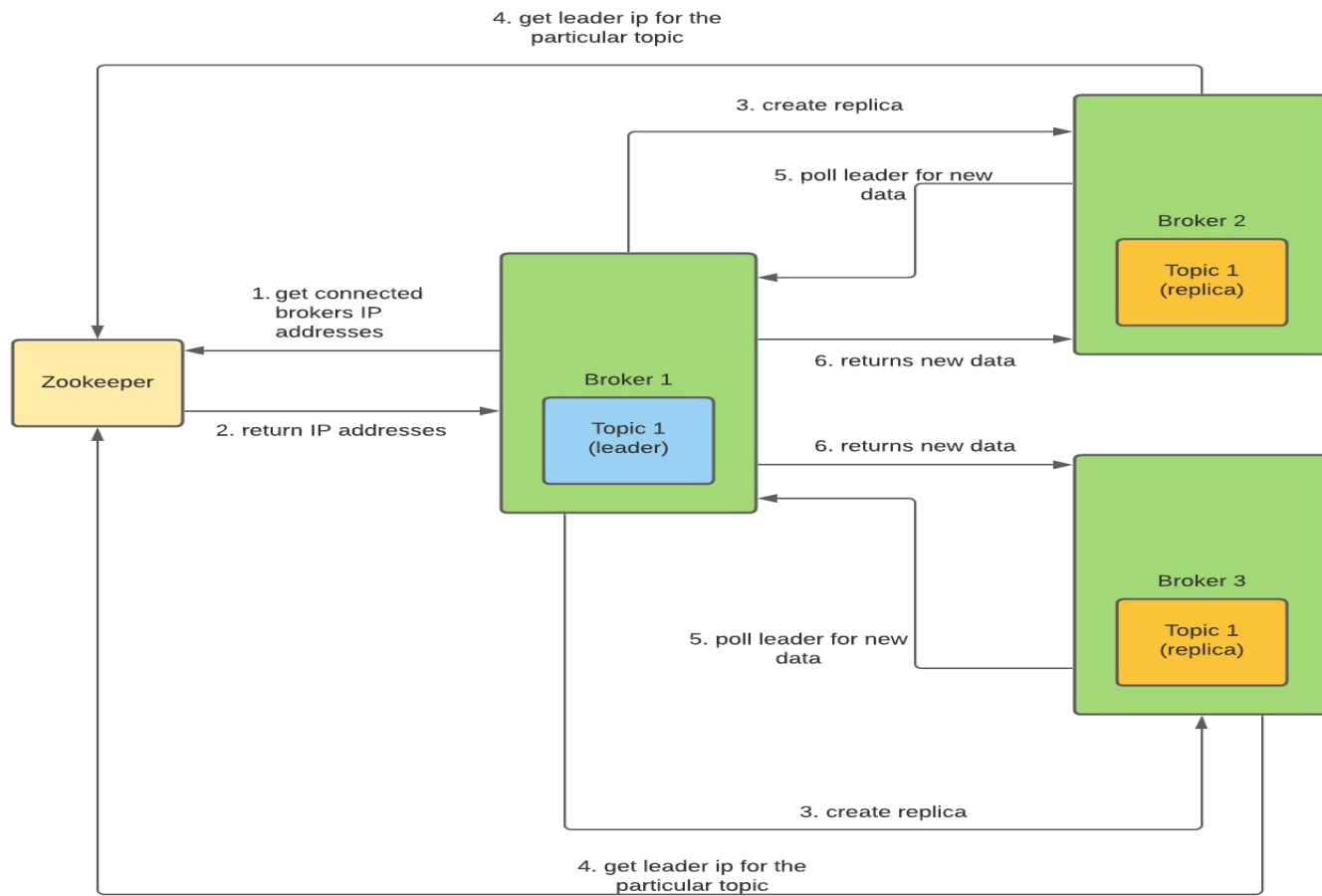
Workflow of Replication

```mermaid
flowchart

A[Create Replica API is called when a topic is created, the topic name and replication factor is passed as the parameter] --> B[Get the IP addresses of the connected brokers from Zookeeper.]

B --> C[Select replication factor number of IPs from the list of IP addresses obtained from Zookeeper]

C --> D[Call the create topic replica API for those IP addresses pass topic name in the request]

D --> E[The handler for the create replica topic API creates the topic on the broker and calls the start leader polling API]

E --> F[The leader polling API gets the current leader of the topic from Zookeeper and calls the leader sync API]

F --> G[Leader sync API keeps polling the leader every 1 second]

G --> H[New data based on the difference of the offset of the leader and the replica topic is returned as a response from the leader]

H --> I[The new data is then written to the replica file]
```
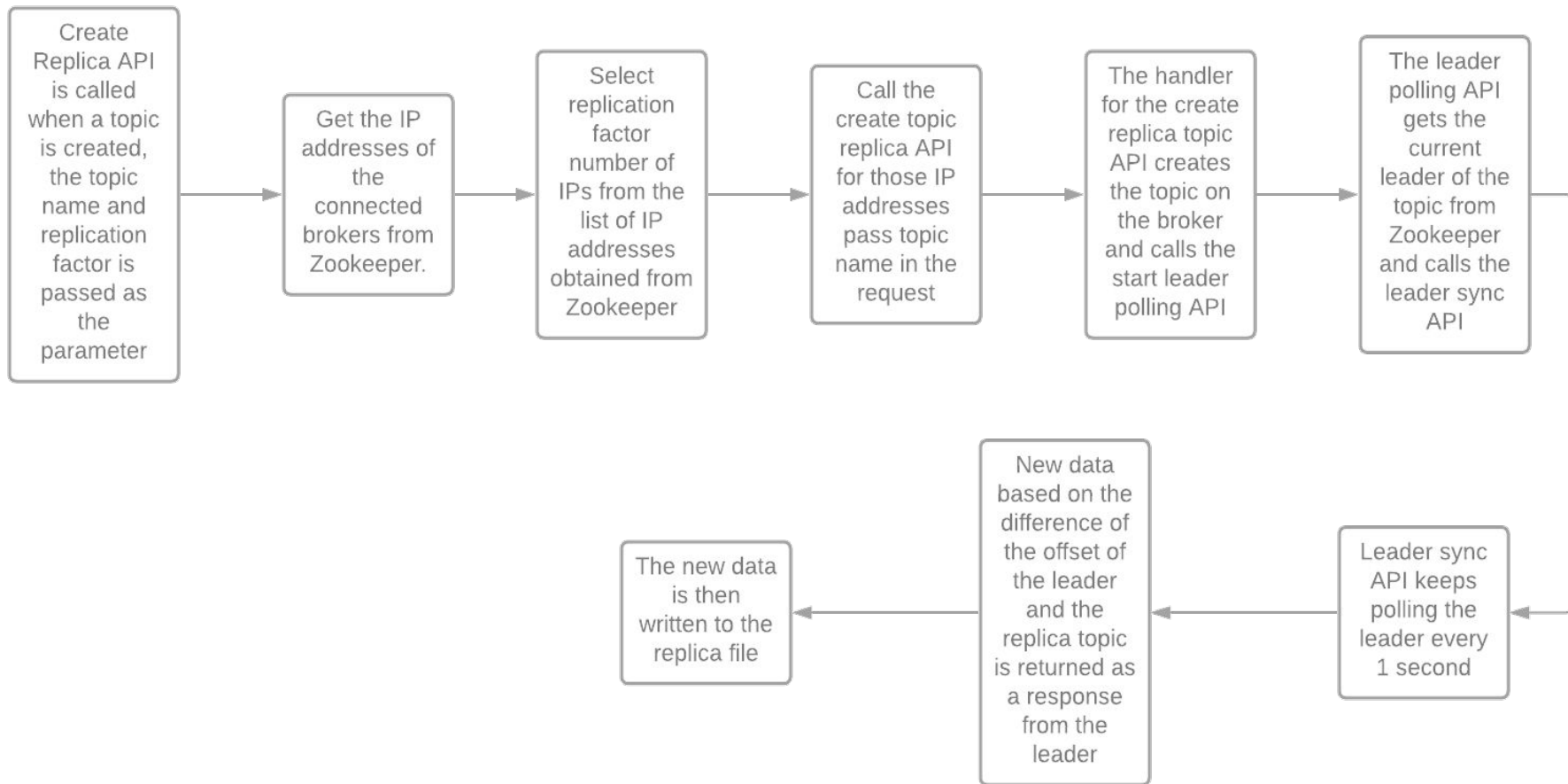
Thanks