## **API Layer**

## TributaryCLI - api:API = new API()+ main(String[] args) executeCommand(String[] arguments) - handleCreateCommand(String[] arguments) - handleCreateTopic(String[] arguments) - handleCreatePartition(String[] arguments) - handleCreateConsumerCommand(String[] arguments) - handleCreateConsumerGroup(String[] arguments) - handleCreateConsumer(String[] arguments) - handleCreateProducer(String[] arguments) - handleDeleteCommand(String[] arguments) - handleDeleteConsumer(String[] arguments) - handleProduceCommand(String[] arguments) - handleProduceEventCommand(String[] arguments) - handleConsumeCommand(String[] arguments) - handleConsumeEvent(String[] arguments) - handleConsumeEvents(String[] arguments) - handleShowCommand(String[] arguments) - handleShowTopicCommand(String[] arguments) - handleShowConsumerCommand(String[] arguments) - handleShowConsumerGroup(String[] arguments) - handleParallelCommand(String[] arguments) - handleParallelProduce(String[] arguments) - handleParallelConsume(String[] arguments) handleSetCommand(String[] arguments) - handleSetConsumerGroupRebalancing(String[] arguments) - handlePlayback(String[] arguments) - printUnknownCommand() API core:Core + createTopic(String id, String type):Coreresponse + createPartition(String topicId, String id):Coreresponse + createConsumergroup(String id, String topicId, String rebalancingStrategy):Coreresponse + createConsumer(String consumerGroupId, String id):Coreresponse + deleteConsumer(String consumerId):Coreresponse + createProducer(String id, String type, String allocationStrategy): Coreresponse + produceEvent(String producerId, String topicId, String eventFilename):Coreresponse + produceEvent(String producerId, String topicId, String eventFilename, String partitionId):Coreresponse + consumeEvent(String consumerId, String partitionId):Coreresponse + consumeEvents(StringconsumerId, String partitionId, String numberOfEvents):Coreresponse + showTopic(String topicId):Coreresponse + showConsumerGroup(String consumerGroupId):Coreresponse + parallelProduce(List<String> producerIds, List<String> topicIds, List<String> eventFilenames):Coreresponse + parallelConsume(List<String> consumerIds, List<String> partitionIds):Coreresponse + setConsumerGroupRebalancing(String consumerGroupId, String rebalancingStrategy):Coreresponse + playback(String consumerId, String partitionId, String offset):Coreresponse - tributaryCluster: TributaryCluster = new TributaryCluster() - consumerGroups:List<Consumergroup> = new ArrayList<ConsumerGroup>(); - producers:List<Producer> = new Arraylist<Producer>(); + createTopic(String id, String type) + createPartition(String topicId, String id): + createConsumergroup(String id, String topicId, String rebalancingStrategy): + createConsumer(String consumerGroupId, String id): + deleteConsumer(String consumerId): + createProducer(String id, String type, String allocationStrategy): + produceEvent(String producerId, String topicId, String eventFilename): + produceEvent(String producerId, String topicId, String eventFilename, String partitionId): + consumeEvent(String consumerId, String partitionId): + consumeEvents(StringconsumerId, String partitionId, String numberOfEvents): + showTopic(String topicId): + showConsumerGroup(String consumerGroupId): + parallelProduce(List<String> producerIds, List<String> topicIds, List<String> eventFilenames): + parallelConsume(List<String> consumerIds, List<String> partitionIds): + setConsumerGroupRebalancing(String consumerGroupId, String rebalancingStrategy): + playback(String consumerId, String partitionId, String offset):

## **Backend**

