**VMWARE Assignments**

Question-1Write a spring boot application that maintains an Inventory of Employee data.

1. API to accept uploading of data as a flat file with line separated data

a. POST /api/employee?action=upload should accept a file

b. FILE CONTENT is lines of names and age – example

GANGULY 32

SACHIN TEND 44

c. The file may have upto 1000,000 data points

d. POST api to upload file should return a task identifying this processing

e. Task should have a get status API to track the completion success or failure of

the task

f. When the processing is done the data should be persisted in the database.

2. CRUD API for employee objects

**Design of Employee Inventory System**

Objective

To create a scalable persistent and low latency system of Creating/Updating/Maintaining/Deleting the employee data of an organization. The design should be scalable to accommodate future growth of the company when the firm sees a rapid increase in its employee count.

Components

This section describes the components the system will have

1. Load Balancer
2. Random Number Generator (Pertaining to creation of Task IDs)
3. API servers (no stickiness to the clients’ requests)
4. Messaging System (Pub/Sub)
5. Key Value store
6. Database

Load Balancer

Since the Api servers have no affinity towards client requests, the load balancer will distribute requests among API servers in a Round Robin fashion.

Random Number Generator

This server will be responsible for generation of random numbers where the random numbers generated will be used as task ids and Employee IDs used in creation of the new employees. It will send a batch of unused random numbers generated (say 100) to the API servers when it receives a request from API servers to get the random number generated.

Algo:

1. Can use srand() to generate a random number with seed as the time when the number generator node came up.
2. Or can keep a unsigned long long variable and it increments it by one. In this case whatever the last number that has been sent to the API server will be written to the config file of this node so cases where this node crashes or is put down for any upgrades on the next boot up it starts generating from the last number +1 from the config file.

Similarly, it can also send the employee ID to the API servers when the requests comes to it from them. It generates the new Employee ID based on the Last Generated Employee ID by reading from its config files.

API Servers

Responsible for catering to clients’ requests.

On boot up, it will query the random number generator to get the unused numbers for the task ids that is to be sent to the client in response for the requests received for action = upload.

These servers will have the task of validating the HTTP request line to see if it maps to any of the API endpoint (Create/Update/Read/Delete/Upload) and also the semantic as well as syntactic input validation.

If validation fails it returns error to the client. Else depending upon the API endpoint invoked by the client this server will behave. For all the APIs except the file upload API ,the server will do the necessary steps and talk to the DB directly. For Upload API functionality, it will form a json buffer

{

“Action” : “Upload”

“task\_id”: “val1”

“Body”: “Request’s Body”

}

And publish it to the messaging system. Then it updates the Key Val store with key as taskId and val would be the status of the task which would be put up as “In Progress” from these API servers always.

On receiving creation of new employee request it does one extra thing to get the employee id from random number generator and then proceeds to talk to the DB similar to the read/delete/update APIs.

Messaging Sytem(Pub/Sub)

This messaging system ensures that requests are not lost once they are successfully responded back with tasks id.

*Publisher*

The publishers are the API servers that receives and validates client requests.

*Subscriber*

The subscribers are another set of nodes which are responsible for Updating DB and updating KV store task status from “In progress” to “Successful” or “Failed”.

*Topics*

This can be decided by the different data intensive functionalities the Employee Inventory Application supports and can be extended for new functionalities support. Currently it will have one topic for the functionality of “Uploading a file”.

Key Value store

It is deployed on a different node for faster response of get status check APIs. The getTask status api will be returned by seeing the status of the tasks in the Key Value store.

To keep a check on the memory of the Key value store that it does not get overrun we can think on the expiration of entries in the Key value store after a suitable time period.

Database

We can have a relational DB for the purpose of this application(indexed on primary key Employee ID). We can decide on indexing on other columns depending on the queries put up.

TABLE : EMPLOYEE

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID  (BIGINT) | Name (Varchar) | Age  (TINYINT) | Created At  (DATETIME) | Last Updated  (DATETIME) | Created  By  (BIGINT) | Last Updated By  (BIGINT) |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

For the purpose of this question the table needs the first 3 fields and indexed on Employee ID.

|  |  |  |
| --- | --- | --- |
| ID  (BIGINT) | Name  (Varchar) | Age  (TINYINT) |
|  |  |  |
|  |  |  |
|  |  |  |

Client Facing APIs (CRUD)

Create

API signature:

*unsigned int createEmployee (string employeeName, unsigned int age, unsigned long long \* employeeID = nullptr)*

This api will create new user in the System with EmployeeName and Age and returns the Employee ID in the out parameter on successful creation and returns the status code of response for the employee Create request.

Request

HTTP Method : POST

Body :

{

“EmployeeName” : “Ramesh R”,

“EmployeeAge”: 33

}

Response

Status Code : 200

Body:

{

“EmployeeID” : “200034562”,

“Message” : “Employee with employee ID 200034562 and employee Name Ramesh and age 33 is successfully created”

}

Delete

API signature:

*unsigned int deleteEmployeeRec (unsiged long long employeeID)*

This API deletes the respective employee from the system and receives the status code of the response for the Delete request.

Request

HTTP Method : DELETE

Body :

{

“EmployeeID” : “200034562”

}

Response

Status Code : 200

Body:

{

“Message” : “Employee with employee ID 200034562 is successfully Deleted”

}

Update

API signature:

*unsigned int updateEmployeeInfo (unsiged long long employeeID, string Name = NULL, unsigned int age =0)*

This API is overloaded depending on the usage of the client as to what field it exactly wants to update. The respective employee bearing the employee id will be updated in the the system and receives the status code of the response for the Update request. (Employee ID =0 will fetch all the employees in the database)

Request

HTTP Method : PUT

Body :

{

“EmployeeID” : “200034562”,

“EmployeeAge” : “34”

}

OR

Body :

{

“EmployeeID” : “200034562”,

“EmployeeName” : “Ramesh V”

}

OR

Body :

{

“EmployeeID” : “200034562”,

“EmployeeName” : “Ramesh V”,

“EmployeeAge” : “34”

}

Response

Status Code : 200

Body:

{

“Message” : “Employee with employee ID 200034562 is successfully Updated”

}

Read

API signature:

*unsigned int getEmployeeDetails (unsiged long long employeeID = 0)*

This API gets the respective employee details from the system and receives the status code of the response for the Get request.

Request

HTTP Method : Get

HEADER:

EmployeeID : “200034562”

Response

Status Code : 200

Body:

{

“Message” : “Employee Info with employee ID 200034562

Employee Name Ramesh V

Age 33”

}

Future Scope

We can introduce Batched APIs for each of the APIs listed above which will carry out activities in batch like creating/Deleting/Reading N number of employees in one request.

*Design Consideration*: All the batched APIs will be treated like the Upload API meaning that the API servers will not carry out this task synchronously rather it will respond back with the task id for such requests. It will post it in the Pub/Sub topics where Topics will now be created as per the service invoked as in for Batch Creation there will be one topic/for Batch deletion another topic and so on and so forth. Like in the Upload api request these API servers wll update the key value pair for these tasks id as well.

Question 2 \_Networks: For Both Backend and Full Stack

            2.1Explain the concepts of Default Gateway in IP

Ans: The Default Gateway is a special node(generally a router ) which the nodes within the network use for sending packets outside the network/within the network. The

***Case 1 : Sending Packets within the network of 4 nodes namely NodeA, NodeB, NodeC, NodeD***.

When a node say NodeA wants to send a packet to NodeB, the machine Node A knows the IP address of the NodeB node but in order to send the packet it should also know the MAC address of NodeB which it checks in its own local MAC cache. If it is there then it sends the packet to NodeB directly. If it is not then it first broadcasts MAC Discovery request for the destination IP to every node in the network receives but the only the node whose IP address matches responds to the requestor with the correct MAC address. Then the MAC address is put in NodeA local cache and the required packet is sent. The default gateway if behaving as a switch connecting the Nodes updates its cache with the MAC address as well if it does not have it its cache. The node issues MAC discover broadcast package only when it knows that it has to send the packet to a node within the network (by using the destination IP and subnet mask). Here, the default gateway is acting as a switch which the nodes in the local network can use to send/receive the packets within the network.

***Case 2: Sending Packets outside the network of 4 nodes namely NodeA, NodeB, NodeC, NodeD***.

When the sender of packet knows that it needs to talk to a node outside the network, say NodeX, it sends the packet to the default gateway. The default gateway looks in its routing table for forwarding the packet via the correct interface to intended network.

Now, two things can happen:

a) The sent packet is for a Network which is directly connected to one of the interfaces of the Gateway itself

b) The sent packet is for a Network which is not connected to interface of the router.

In both the cases the routing table is looked up and the decision is taken to send the packet via the correct interface. However, for the second case (that is case b) the packet is sent to ISP upstream router for the network.

            2.2Explain the concepts of SNAT and DNAT

Ans: NAT which stands for Network Address Transalation is used to convert a local ip address to a public routable IP address. It is generally used for two reasons 1) Security 2) Cost.

To make IP addresses routable we need Public IP which has an associated cost with it therefore in the event where we have many machines in our network which talks to the outside world then for each machine we need to buy a public IP which eventually adds to the expense especially if the network grows with addition of more machines within the network. Secondly, since the machines within the network are reachable via public IP it exposes the machines to external threats hence decreasing making the network less secure.

This is where NAT comes to the rescue. Basically with NATing, what is done is that the packets sent to the router with a local private Ip is converted to a public routable IP and then sent forward. This is achieved by maintaining a NAT table where we map the the local private IP to the respective Public IP that is used. There are two ways in which is NATing is achieved namely 1) Static NATing 2) Dynamic NATing

1. Static NAT(SNAT): This is done by obtaining a pool of public IP addresses and then mapping them statically to each machines’ private IP within the network. Here each machine within the network gets the public IP which is statically configured in the NAT table. This way of NATing helps in the security aspects of the network as the private machines are still not reachable via the outside world. Cost wise it does not benefit much.
2. Dynamic NAT(DNAT): This technique inherently makes use of Port Address Translation (PAT) technique to achieve dynamicity for conversion of private IP to public IP. By implementing this DNAT technique the need for obtaining the number of public IPs is drastically reduced (it can be brought down to one also). This is achieved via the PAT technique which essentially stores the port number of the local machines from where the request is initiated along with the private IP in the NAT table. Although the router while forwarding the packet outside, removes the local IP and puts forth the Public IP but it does not change the port number of the local machines’ port number which helps in delivering back the response to the original requestor as the response comes back on the same port number which was used as a source port number while sending the request, only this time in response it came back as destination port number. This technique of dynamic NATing ensures cost and security optimization.

            2.3

            A.192.168.101.2/24

            B.192.168.101.3/24

            C.192.168.102.2/24

            D.192.168.102.3/24

            A,B,C,D are the IPs to be assigned to four computers ;

            2.1What network elements are need to arrive at the above network architecture ; explain their configurations in terms at L3/L2

            Details of the IP assignments to be given to each node ;

Ans:

All the four IPs are private IPs following the convention of 192.168.x.x. All the ips have the same subnet mask of 255.255.255.0. It makes use of Classles Interet Domain Routing technique using whack method of addressing Network ID (that is out 32 bits how many bits are used for network id ). Here whack 24 means that 24 bits are used for network id.

The network id for both IP A and IP B is same that is 192.168.101.0(Network ID) and the network id of IP C and IP D are same that is 192.168.102.0 (Network ID). This implies that Computer1 (IP A) and Computer2 (IP B) are on the same network, say NetworkAB, whereas Computer3 (IP C) and Computer4 (IP D) are on another network, say NetworkCD.

To connect both Computer1 and Computer2 within the NetworkAB, a switch SwitchAB can be used. Similarly, for connecting Computer3 and Computer4 within the NetworkCD, another SwitchCD can be used.

However, to connect both the networks a Router, say NetworkABCD, should be used.

The default gateway router IP for NetworkAB can be configured as 192.168.101.1 and the default gateway router IP for NetworkCD can be configured as 192.168.102.1.

So on each node IP configurations will be like below:

Computer1:

IP address: 192.168.101.2

Subnet Mask: 255.255.255.0

Default Gateway router IP: 192.168.101.1

Computer2:

IP address: 192.168.101.3

Subnet Mask: 255.255.255.0

Default Gateway router IP: 192.168.101.1

Computer3:

IP address: 192.168.102.2

Subnet Mask: 255.255.255.0

Default Gateway router IP: 192.168.102.1

Computer4:

IP address: 192.168.102.3

Subnet Mask: 255.255.255.0

Default Gateway router IP: 192.168.102.1

            2.4 - Explain ARP

ARP is a protocol used to discover the physical address of a machine/NIC card. It stands for Address Resolution Protocol. This is used by the Layer2 layer that is the Data Layer of OSI convention. The IP addresses are the logical address that is used for sending/receiving, however the actual data transfer happens via the MAC address/physical address of the machine.

The ARP essentially can be broken down in two steps

1) Broadcasting MAC discovery request.

2) Unicast reply by the rightful node.

When a node wants to send a packet to another node within the network then it first checks within its local MAC cache if it has the MAC address of the destination IP, if yes then the packet is sent. Otherwise, the source machine sends a broadcasts message of MAC discovery request containing the Destination IP address to all the nodes within the network. Every node in the network receives this request but only the node whose own ip address matches with destination ip address replies with its MAC address to the sender. Rest all the other nodes remain silent. Then after sender receives the MAC address, it sends out the packet to the destination machine.

Similarly, for sending packet outside the network, if the sender does not have the Gateway/router MAC address, it does ARP first with Gateway IP and then sends the packet to the gateway.