

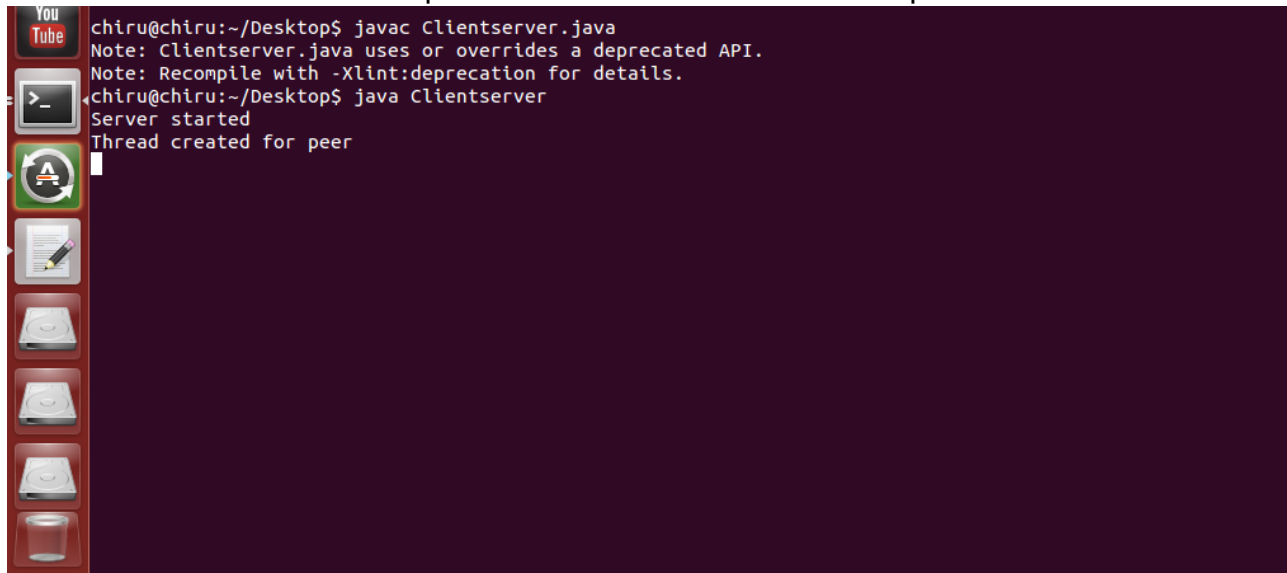
**CS550 Advanced Operating Systems**  
**Programming Assignment 2**  
**Output**

Submitted by:  
Chiranjeevi Ankamredy  
A20359837

The output generated at every step of execution is explained below with screen shots .

### **a. Server Initialization:**

When a server is started , And client connected to the server .Here we have give port in server code.Server awaits requests and created a thread to the peer.

A screenshot of a Linux terminal window. The terminal has a dark purple background. On the left side, there is a vertical dock with several icons: a YouTube icon, a terminal icon, a green circular icon with a white 'A' and a refresh symbol, a notepad icon, and four disk icons. The terminal text shows the user 'chiru' at a prompt 'chiru@chiru:~/Desktop\$' running 'javac Clientserver.java'. It displays two deprecation warnings from the Java compiler. Then, the user runs 'java Clientserver', and the terminal outputs 'Server started' and 'Thread created for peer' on separate lines.

```
chiru@chiru:~/Desktop$ javac Clientserver.java
Note: Clientserver.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
chiru@chiru:~/Desktop$ java Clientserver
Server started
Thread created for peer
```

### **b. Client Connections:**

When a Client connects it to the server it does by the selecting the node to connect to and multiple clients can connect to a single server. In the figure below, one client connect to the server at port 7777.

```
chiru@chiru: ~/Desktop
chiru@chiru:~/Desktop$ javac PeerClient.java
Note: PeerClient.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
chiru@chiru:~/Desktop$ java PeerClient
Enter key to connect server:
lkjh
connecting to the server 5
127.0.0.1
7777
Peer1 Intitialized

Hash Table Operations

1. PUT
2. GET
3. DELETE
```

### c. Client Operations:

The Client connected to server does put , get and delete operations on key-value pairs on DHT.

In the figure below, it displays three operations PUT,GET AND DELETE.

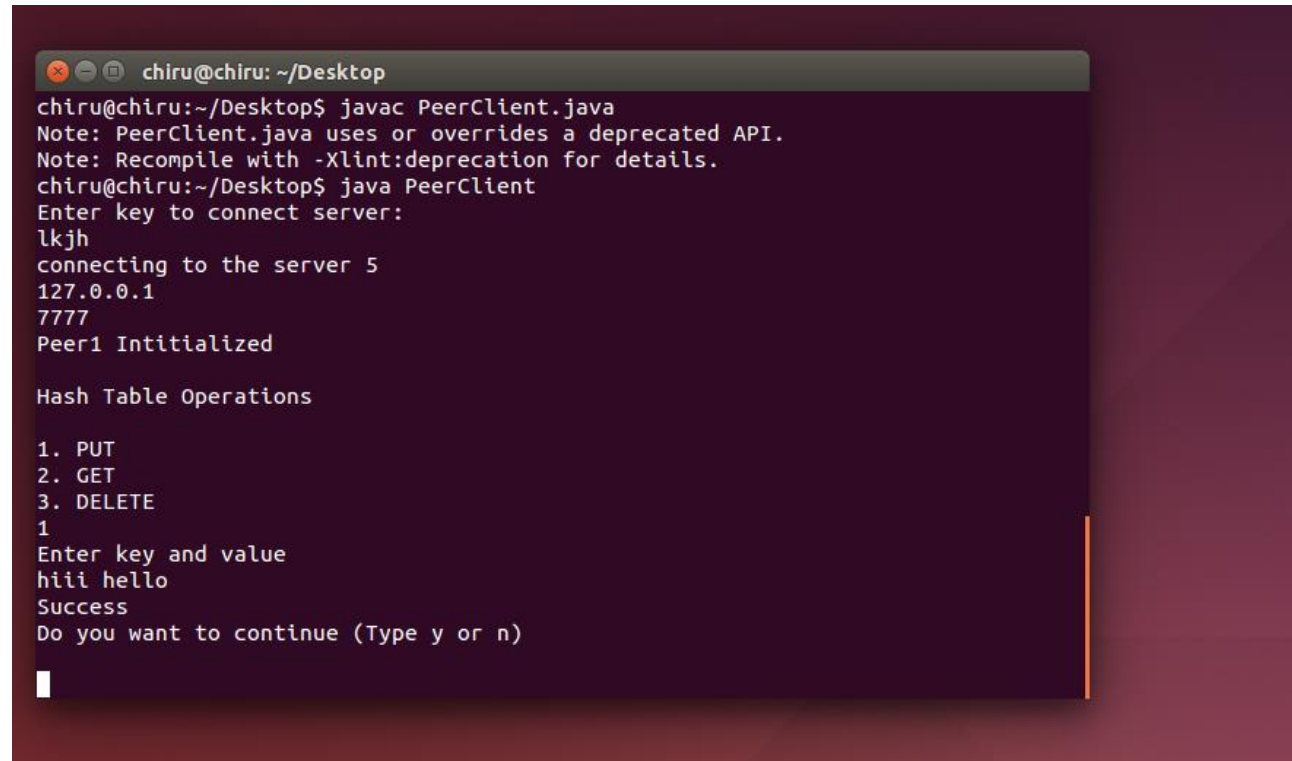
```
chiru@chiru: ~/Desktop
chiru@chiru:~/Desktop$ javac PeerClient.java
Note: PeerClient.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
chiru@chiru:~/Desktop$ java PeerClient
Enter key to connect server:
lkjh
connecting to the server 5
127.0.0.1
7777
Peer1 Intitialized

Hash Table Operations

1. PUT
2. GET
3. DELETE
```

#### d. PUT operation :

When a client does a put operation , the key is hashed and a node(server) is selected and key-value pair is put at the hash table on that server. and successful put returns 1 .



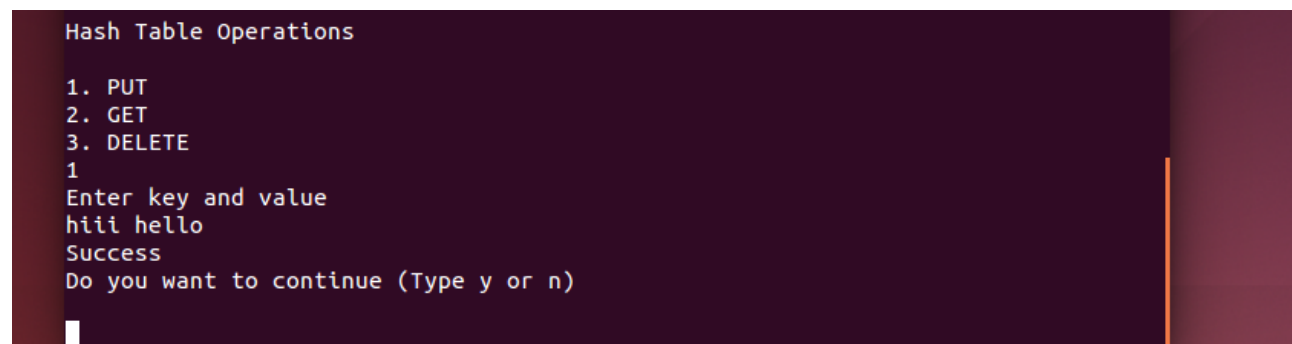
```
chiru@chiru: ~/Desktop
chiru@chiru:~/Desktop$ javac PeerClient.java
Note: PeerClient.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
chiru@chiru:~/Desktop$ java PeerClient
Enter key to connect server:
lkjh
connecting to the server 5
127.0.0.1
7777
Peer1 Intitialized

Hash Table Operations

1. PUT
2. GET
3. DELETE
1
Enter key and value
hihi hello
Success
Do you want to continue (Type y or n)
█
```

Put operation also returns “Success” on storing the key and value in Hash table, In case if the connect to the server on which the operation is made is lost .

In the figure below, the server which was connected initially is closed , and now when the client tries to do a PUT operation, it returns Success.

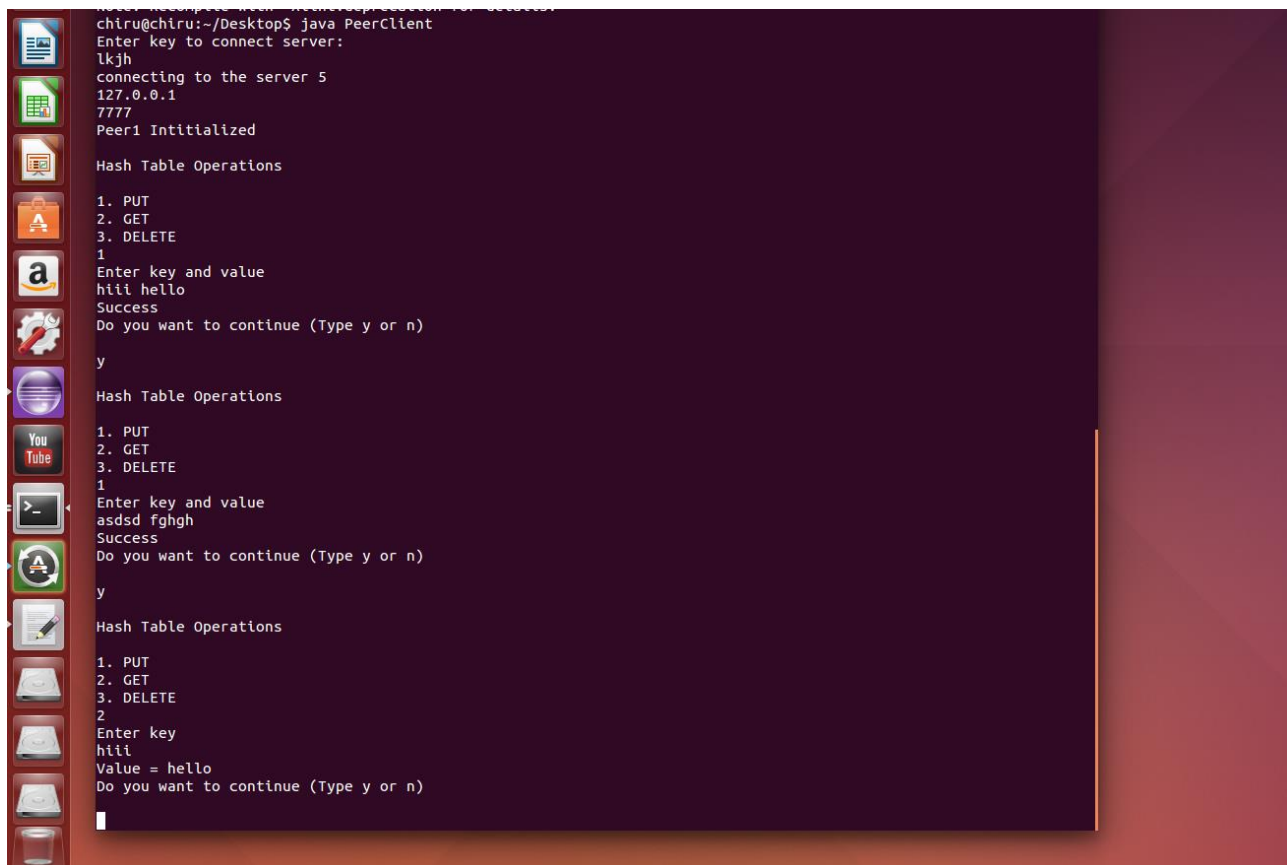


```
Hash Table Operations

1. PUT
2. GET
3. DELETE
1
Enter key and value
hihi hello
Success
Do you want to continue (Type y or n)
█
```

### e. RETRIEVE operation:

The client does the get operation by selecting 2 on the users console and gets a value for the key by entering the key. And it returns Key value.



```
chiru@chiru:~/Desktop$ java PeerClient
Enter key to connect server:
lkjh
connecting to the server 5
127.0.0.1
7777
Peer1 Intialized

Hash Table Operations

1. PUT
2. GET
3. DELETE
1
Enter key and value
hiii hello
Success
Do you want to continue (Type y or n)
y

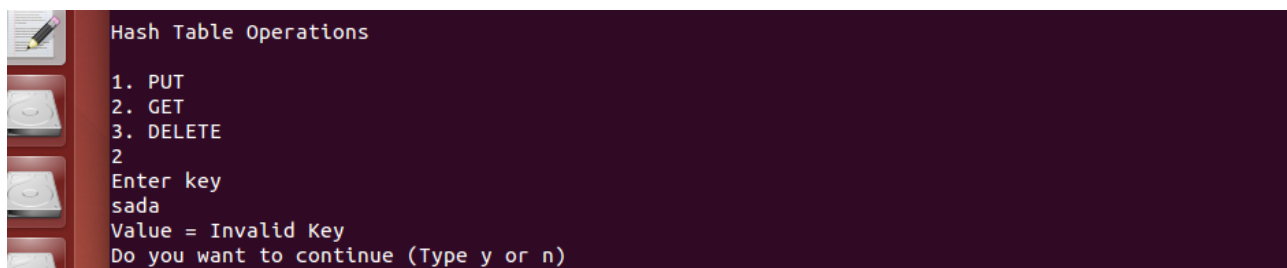
Hash Table Operations

1. PUT
2. GET
3. DELETE
1
Enter key and value
asdsd fghgh
Success
Do you want to continue (Type y or n)
y

Hash Table Operations

1. PUT
2. GET
3. DELETE
2
Enter key
hiii
Value = hello
Do you want to continue (Type y or n)
```

If the key that is searched for is not at any of the servers that client is connected to, “Invalid Key” message is returned to the client.

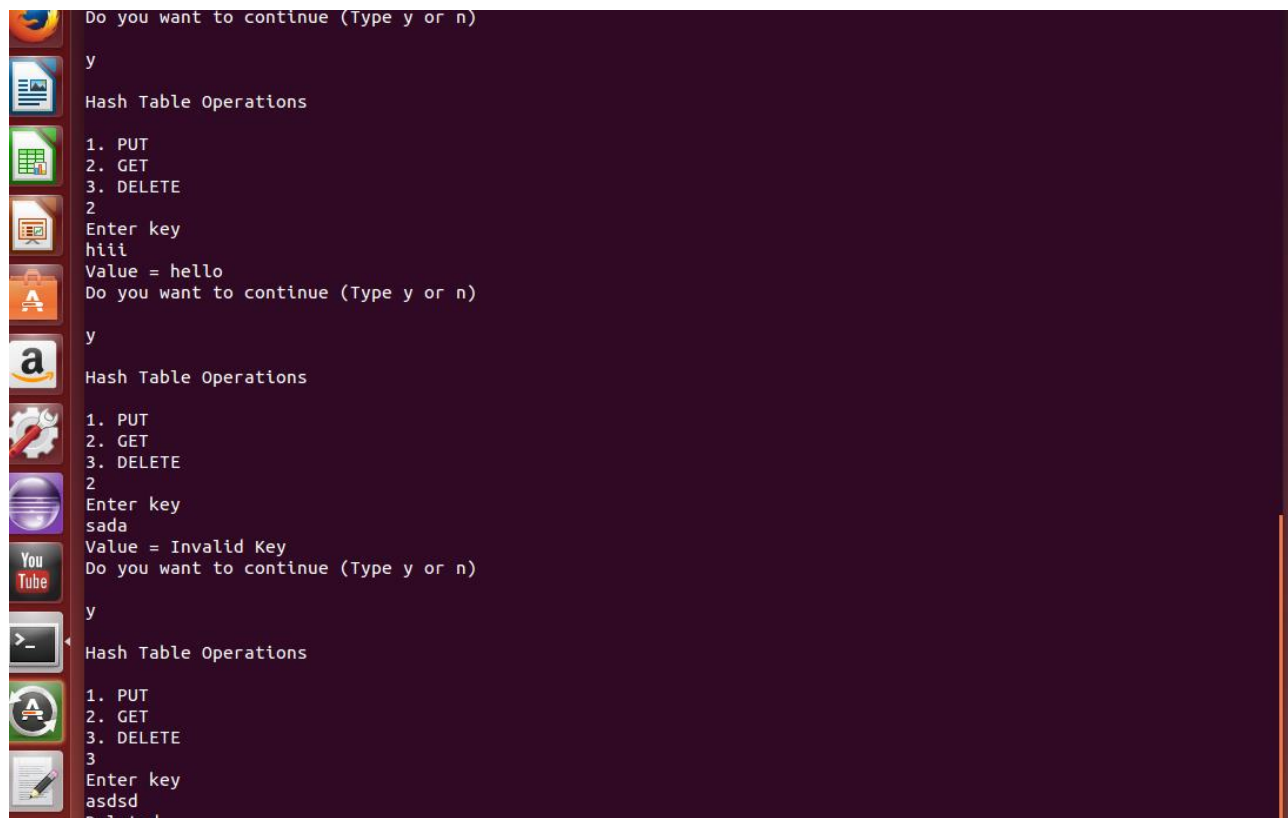


```
Hash Table Operations

1. PUT
2. GET
3. DELETE
2
Enter key
sada
Value = Invalid Key
Do you want to continue (Type y or n)
```

### f. DELETE Operation

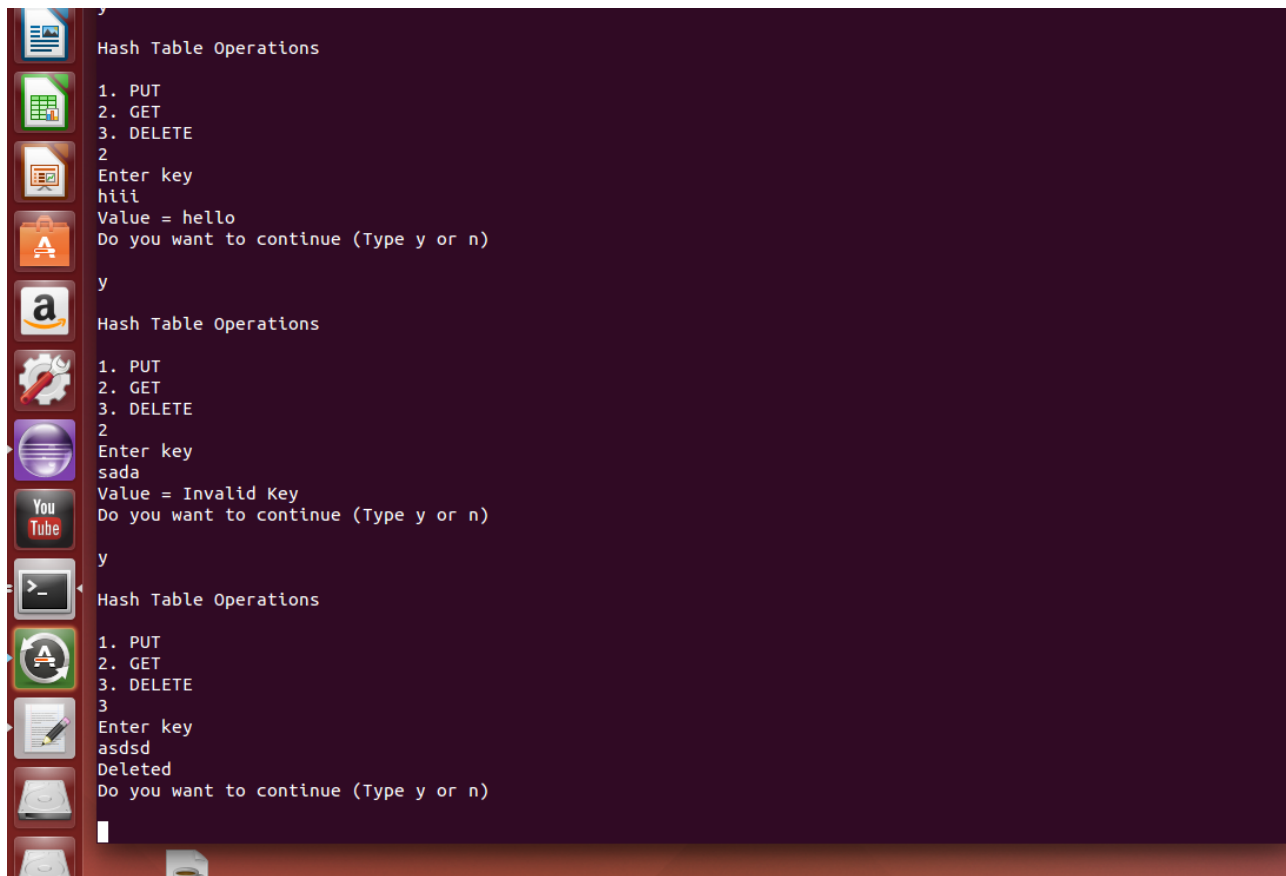
The Client does the delete operation by enter 3 on users console. The client enters the key to be deleted and 'Deleted' is returned on successful delete.



The screenshot shows a Windows desktop with a dark-themed taskbar on the left. The taskbar contains icons for Internet Explorer, a document, Excel, a presentation, a folder, Amazon, a gear, a globe, YouTube, a command prompt, a folder with a magnifying glass, and a notepad. A terminal window is open, displaying the following text:

```
Do you want to continue (Type y or n)
y
Hash Table Operations
1. PUT
2. GET
3. DELETE
2
Enter key
hihi
Value = hello
Do you want to continue (Type y or n)
y
Hash Table Operations
1. PUT
2. GET
3. DELETE
2
Enter key
sada
Value = Invalid Key
Do you want to continue (Type y or n)
y
Hash Table Operations
1. PUT
2. GET
3. DELETE
3
Enter key
asdsd
Deleted
```

On Success it returns DELETED on client side.



### g. Exit Client

If the client exits or shuts down connection by enter 'n' on the console .

### h. Replication (Extra Credit ):

When replication is allowed at servers on the nodes, Whenever client does the put operation , the key-value that is put in the hash table on the server is replicated at the next node .

The Following sequence in config file.

```
127.0.0.1 2222
127.0.0.1 3333
127.0.0.1 4444
127.0.0.1 5555
127.0.0.1 6666
127.0.0.1 7777
```

127.0.0.1 8888

127.0.0.1 9999

When client connects to 2222 server, it will do operations on “2222” server. If the server fails to do operations. Then it will store the operations on next node i.e “3333”  
Client does a put operation of key “(hi, hello)” and returns ‘Success’.

If 3333 fails it connects to 4444.

If 4444 fails it connects to 5555.

If 5555 fails it connects to 6666.

If 6666 fails it connects to 7777.

If 7777 fails it connects to 8888.

If 8888 fails it connects to 9999.

If 9999 fails it connects to 2222.

The key-value pair is received at the server in the form of message. The key and value is extracted from the message and is stored in its hash table , The key-value pair is also replicated at the next neighboring node (the above procedure explained).