Name: Akshata Yadav Roll No: - 522

Academic Year: 2022-2023 Class: MSC-I (CS)

Subject – Business Intelligence and Big Data .

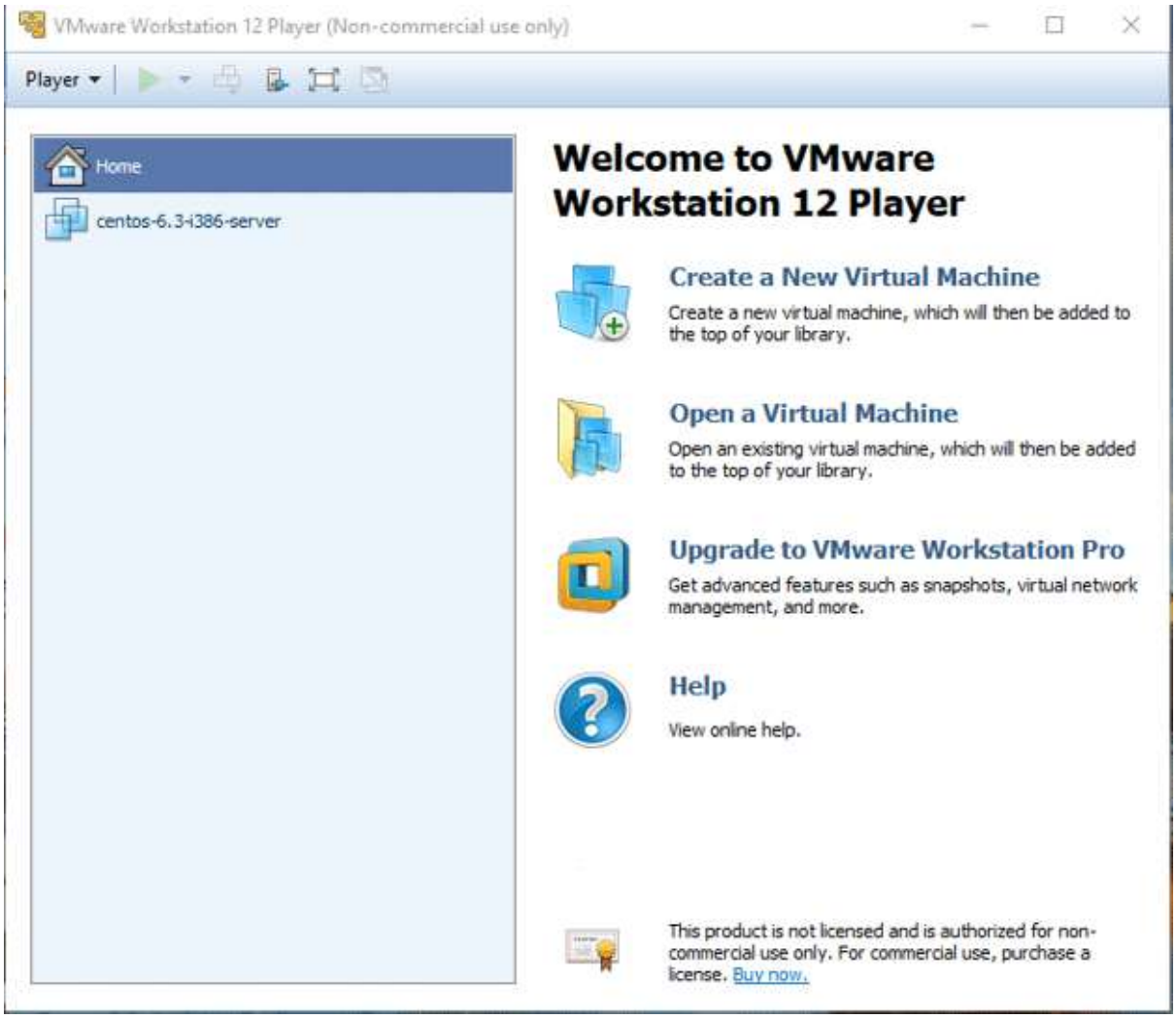
No	Date	Title	Sign
1		Installing and setting environment variable for working with Apache Hadoop.	
2		Implementing Map-Reduce Program for Word Count problem.	
3		Implementing Map-Reduce Program for Word Count problem	
4		Install HBase and use the HBase Data model store and retrieve data.	
5		Install Hive and use Hive Create and store structured database.	
6		write a program to construct different type of kshingles for a given document .	
7		Write a program for measuring similarity among documents and detecting passages which have been reused	
8		Write a program to compute the n-moment for a given stream where n is given.	
9		Write a program to demonstrate the Alon-Matias-Szegedy Algorithm for second moments	

Practical -1

Aim – Installing and setting environment variable for working with Apache Hadoop.

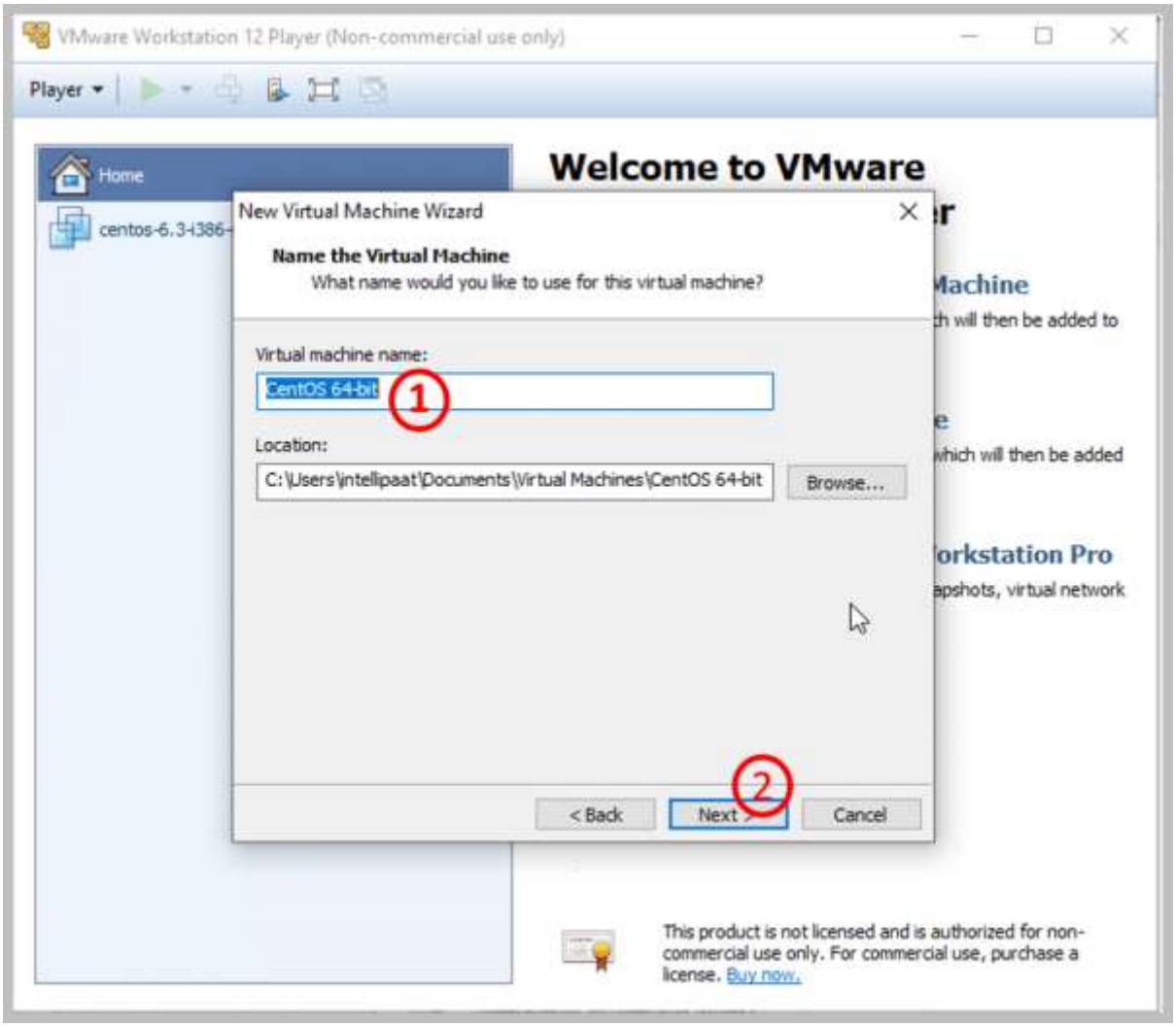
Step 1: Installing VMware Workstation

1) Download VMware Workstation from this
2) Once downloaded, open the .exe file and set the location as required
3) Follow the required steps of installation.



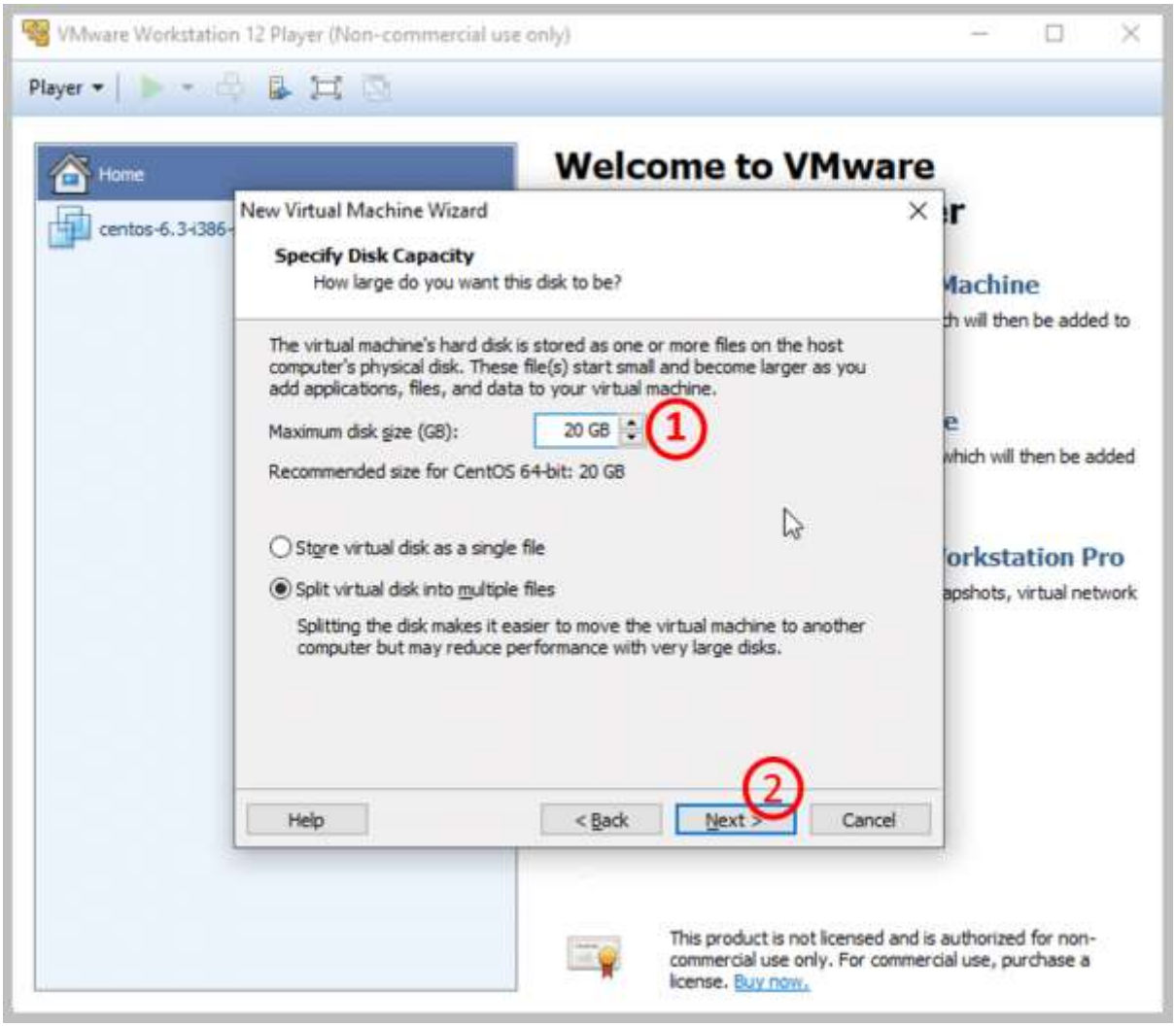
As seen in the screenshot above, **browse** the location of your CentOS file you downloaded. Note that it should be a **disc image file**

2. Click on **Next**



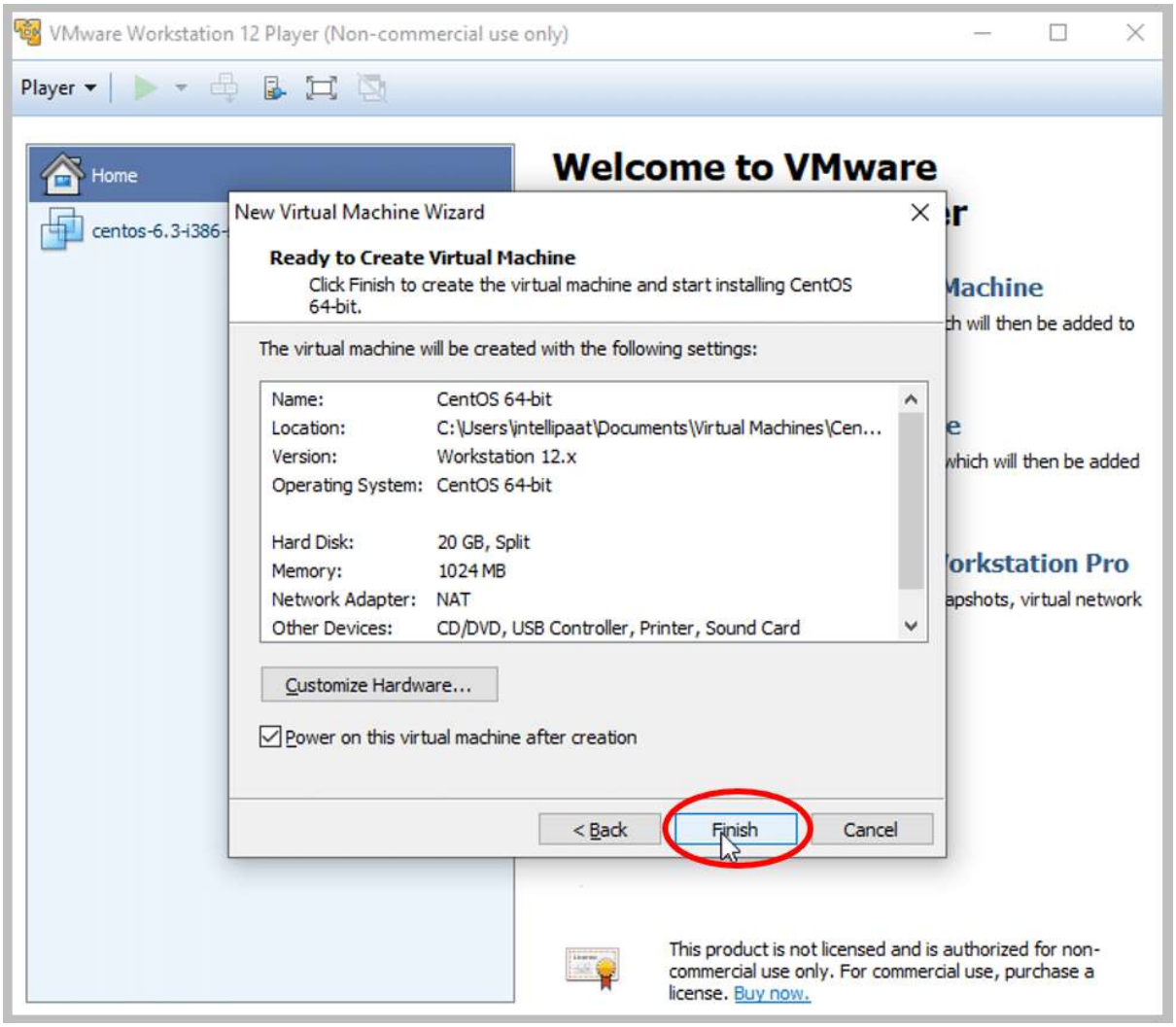
1. Choose the name of your machine. Here, I have given the name **CentOS 64-bit**

2. Then, click **Next**



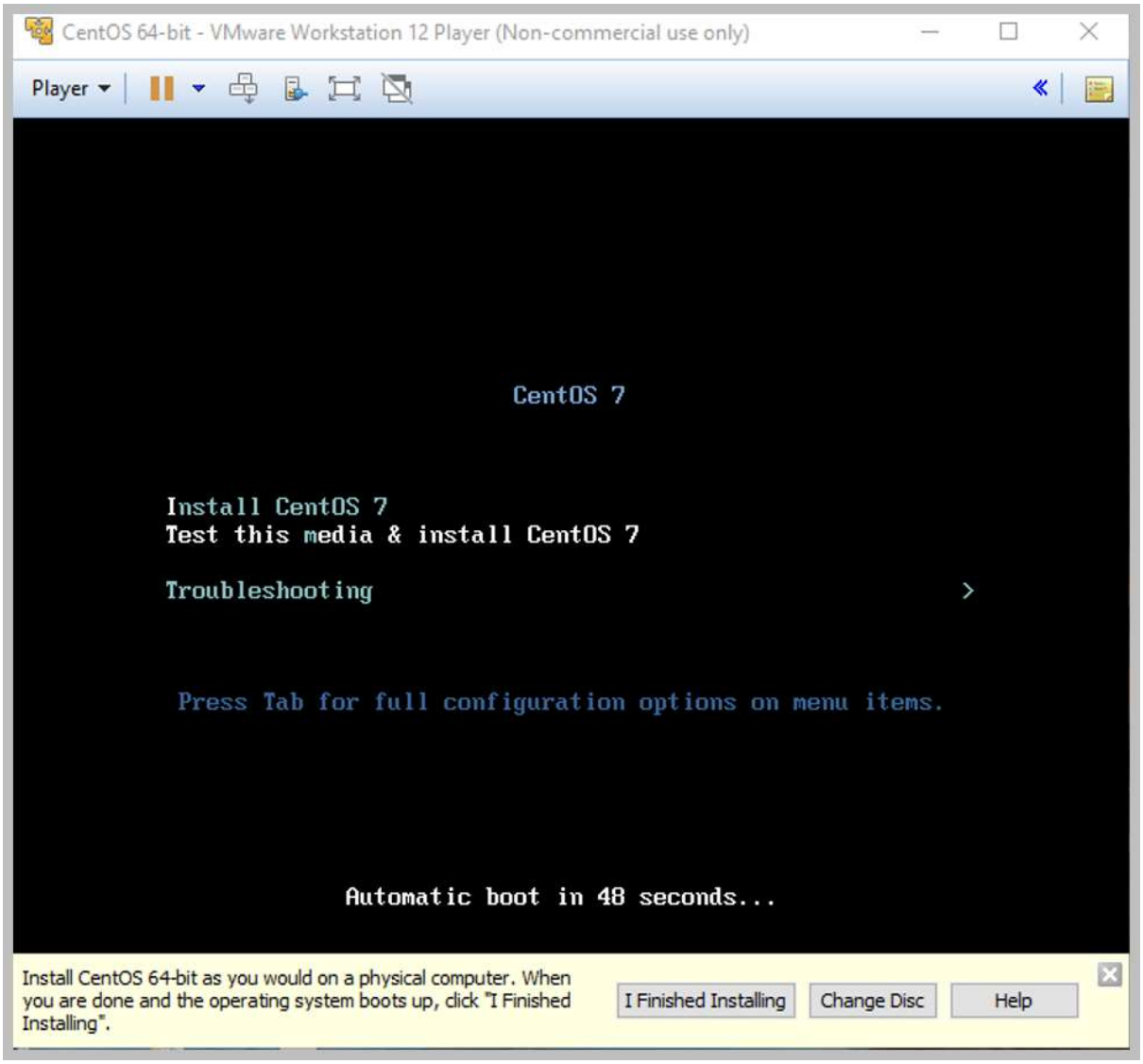
1. Specify the disk capacity. Here, I have specified it to be **20 GB**

2. Click **Next**



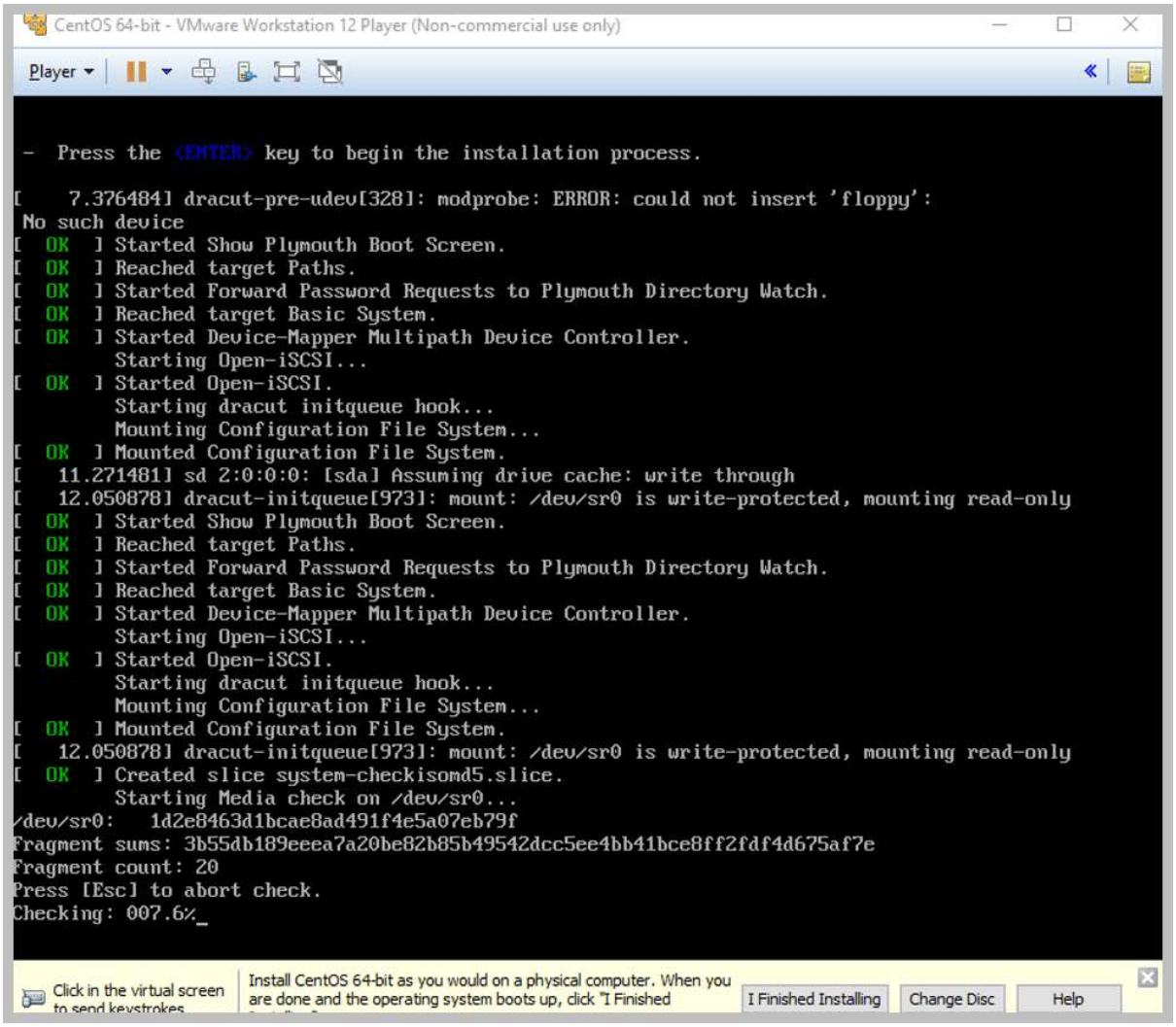
-
    - Click on **Finish**

- After this, you should be able to see a window as shown below. This screen indicates that you are booting the system and getting it ready for installation. You will be given a time of 60 seconds to change the option from **Install CentOS** to others. You will need to wait for 60 seconds if you need the option selected to be **Install CentOS**

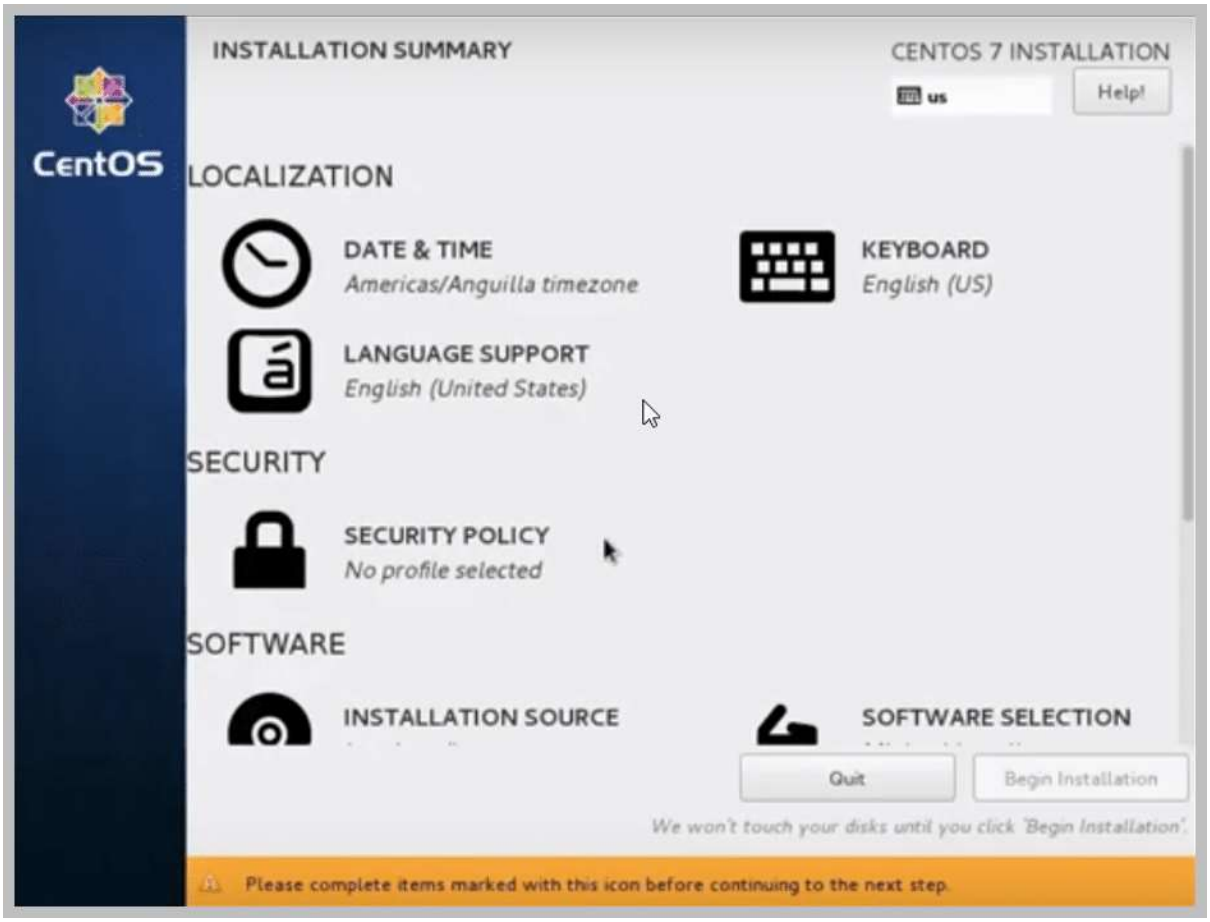


Note: In the image above, you can see three options, such as, **I Finished Installing**, **Change Disc**, and **Help**. You don't need to touch any of these until your CentOS is successfully installed.

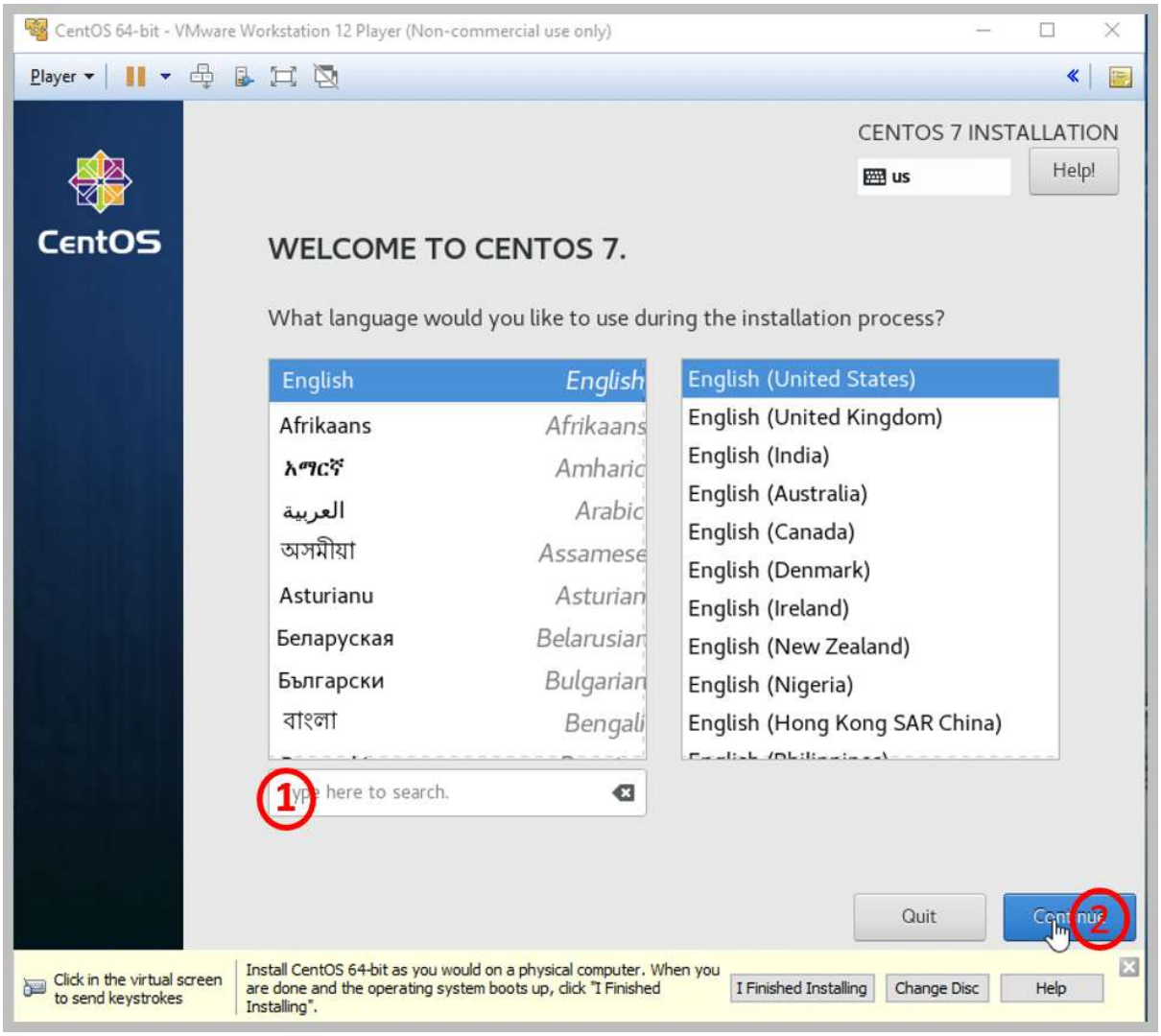
-
    - At the moment, your system is being checked and is getting ready for installation



Once the checking percentage reaches 100%, you will be taken to a screen as shown below



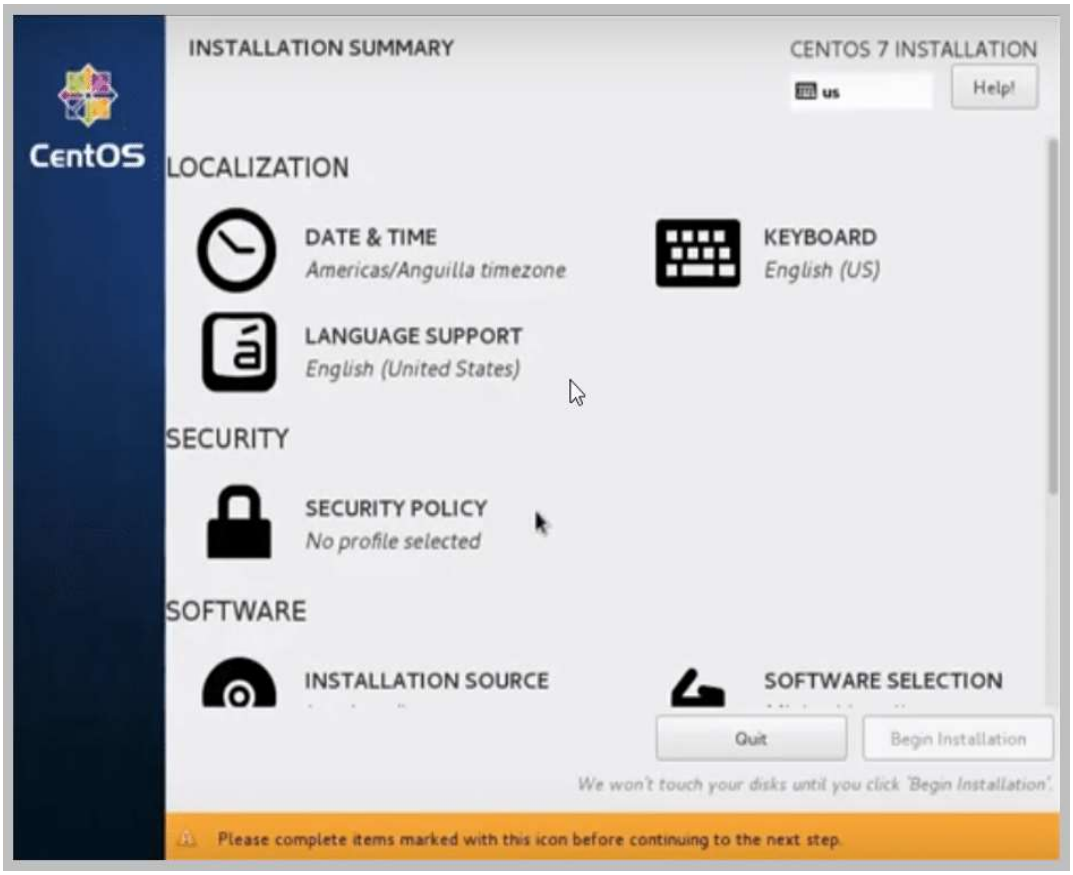
Step 4: Here, you can choose your language. The default language is English, and that is what I have selected



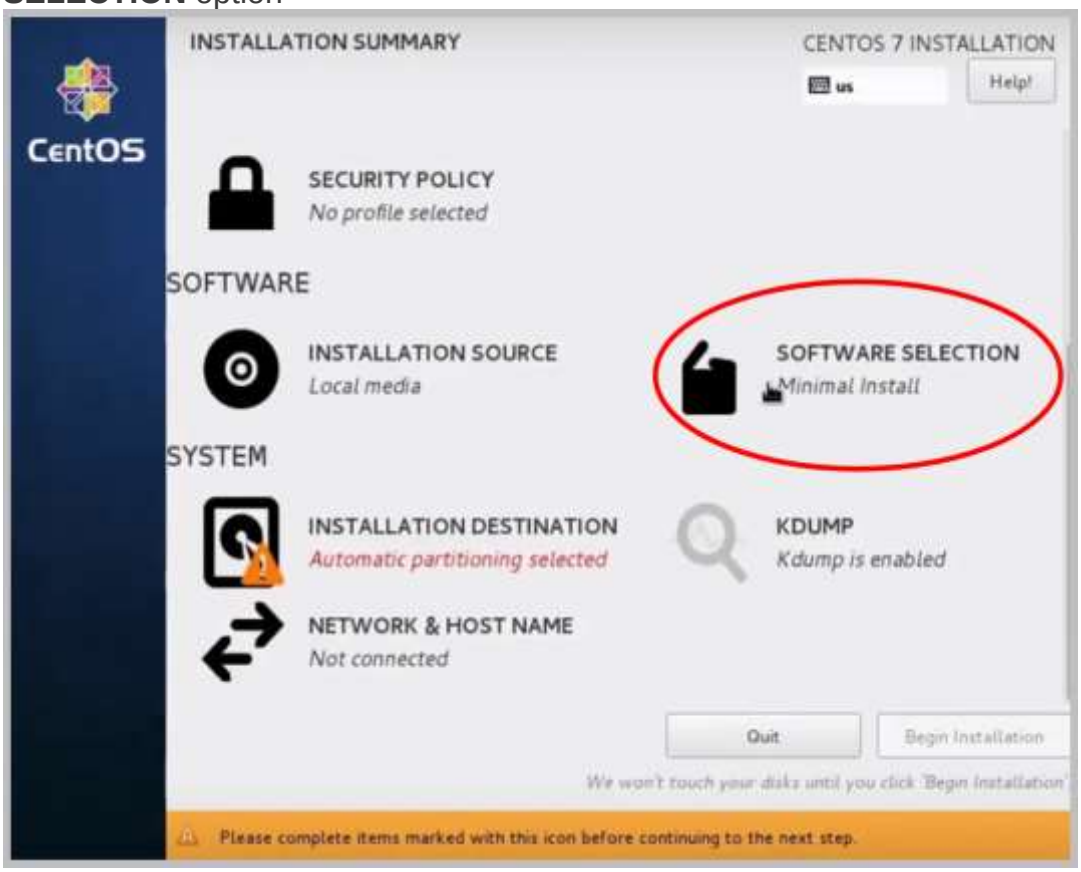
1. If you want any other language to be selected, specify it
2. Click on **Continue**

Step 5: Setting up the Installation Processes

-
    - From Step 4, you will be directed to a window with various options as shown below:



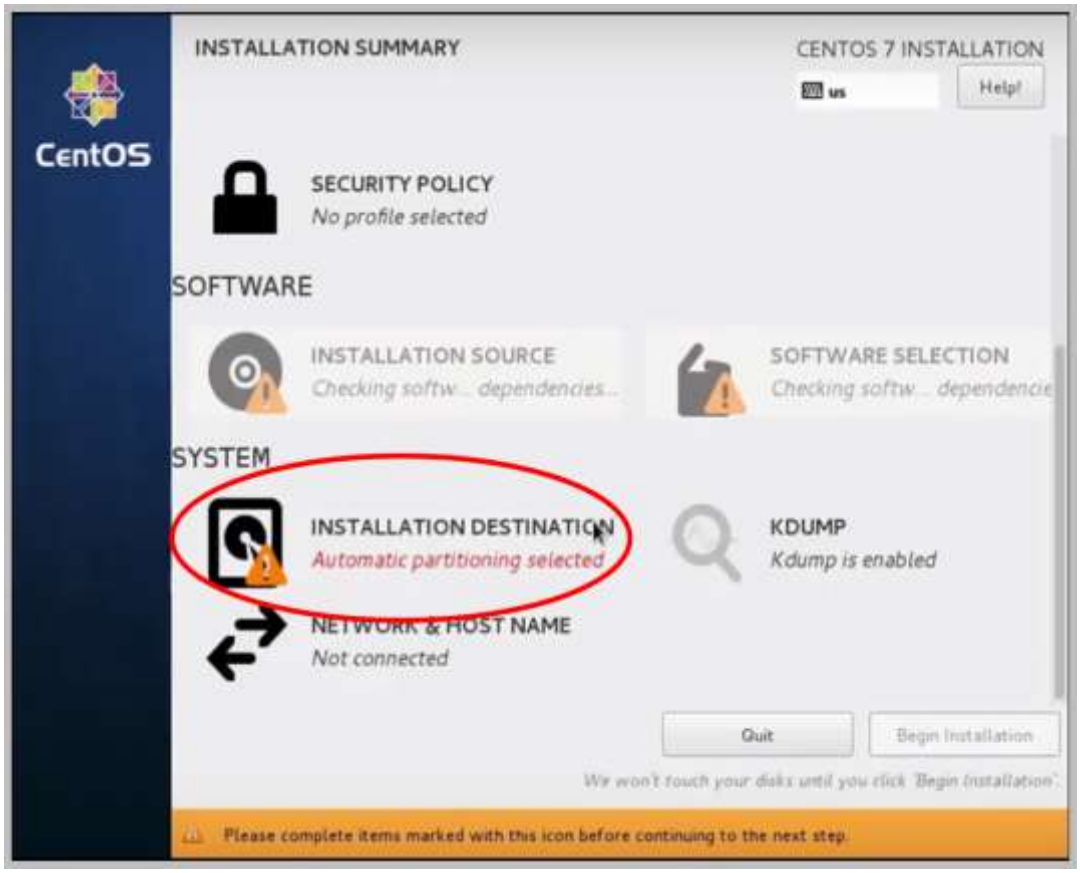
First, to select the software type, click on the **SOFTWARE SELECTION** option

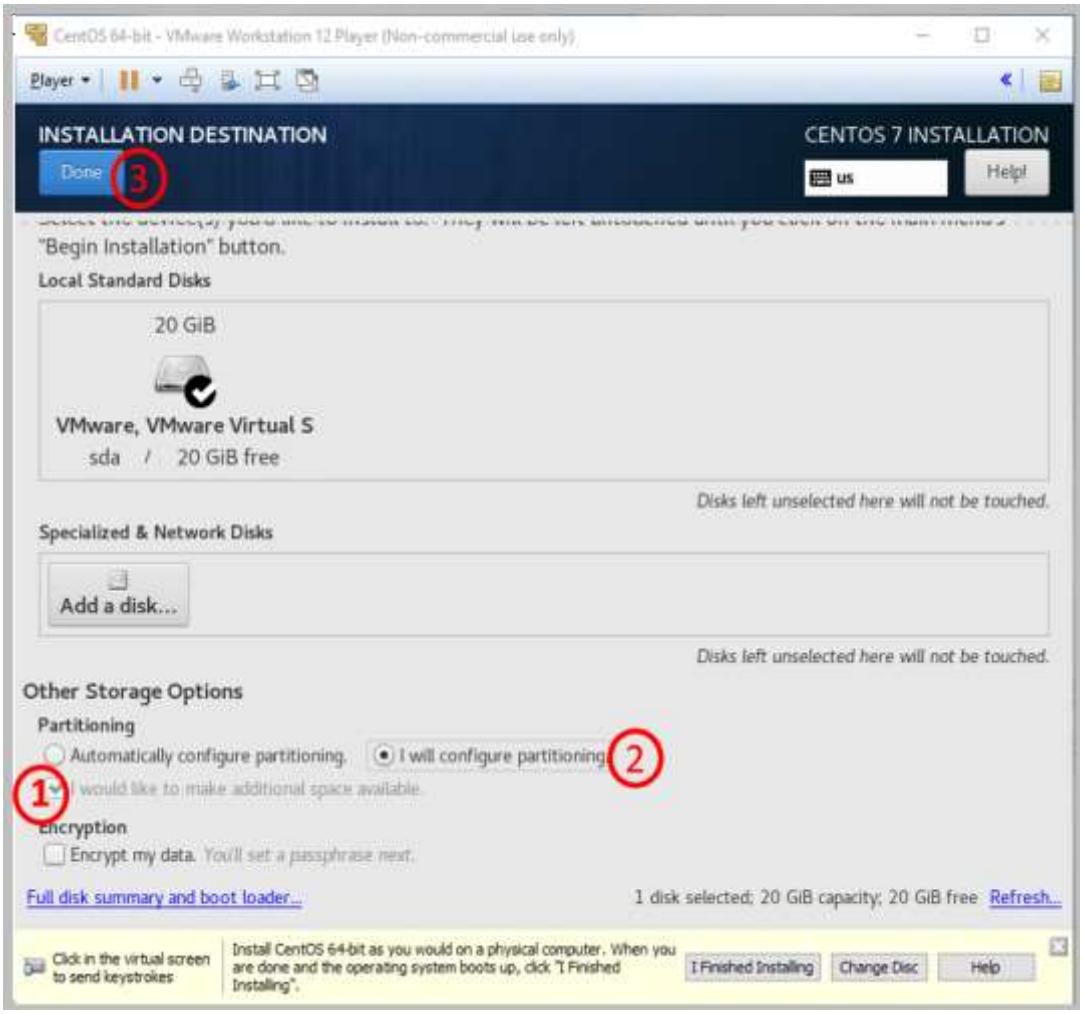


- Now, you will see the following window:1. Select the **Server with GUI** option to give your server a graphical appeal
  2. Click on **Done**

-
  - After clicking on **Done**, you will be taken to the main menu where you had previously selected **SOFTWARE SELECTION**

- Next, you need to click on **INSTALLATION DESTINATION**



- 

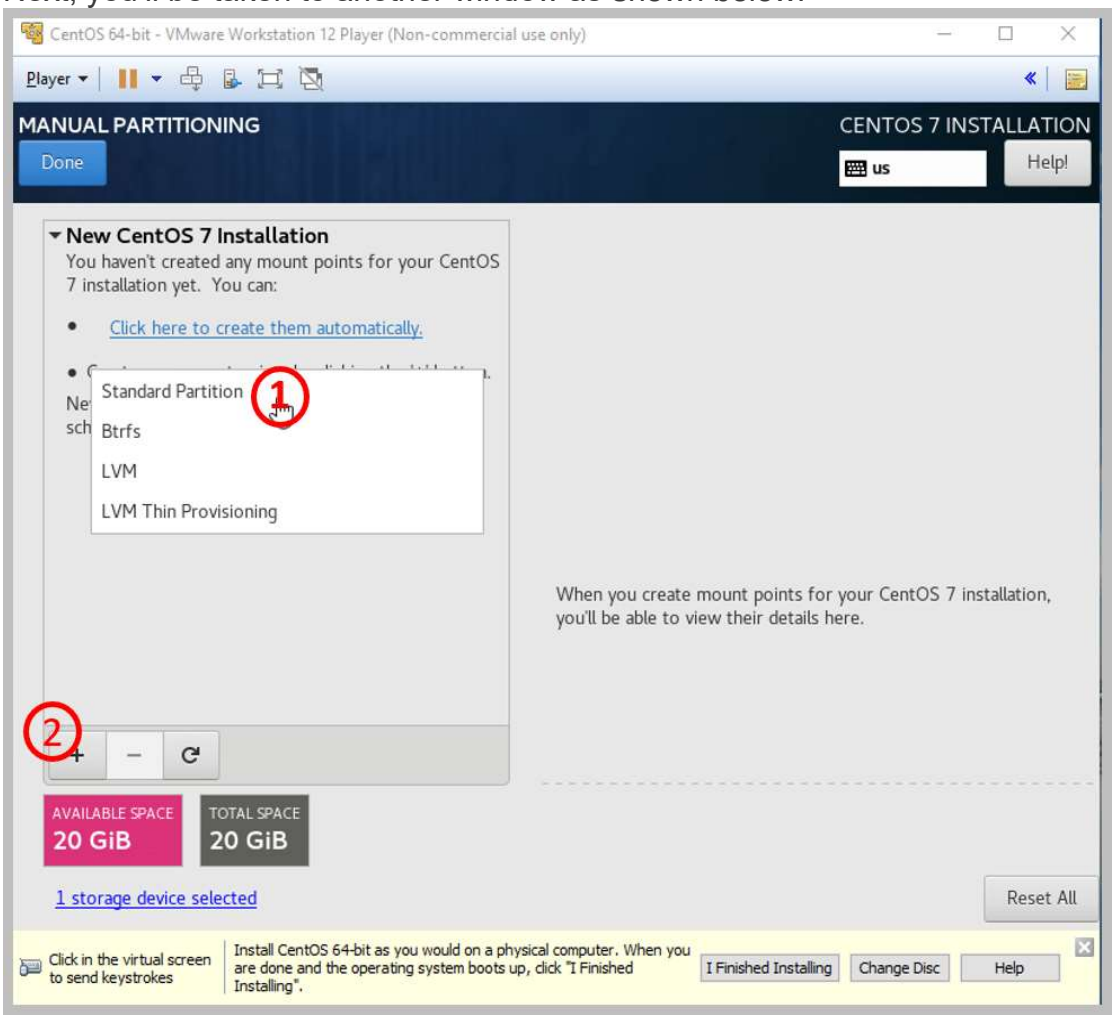
1.

Under **Other Storage Options**, select **I would like to make additional space available**

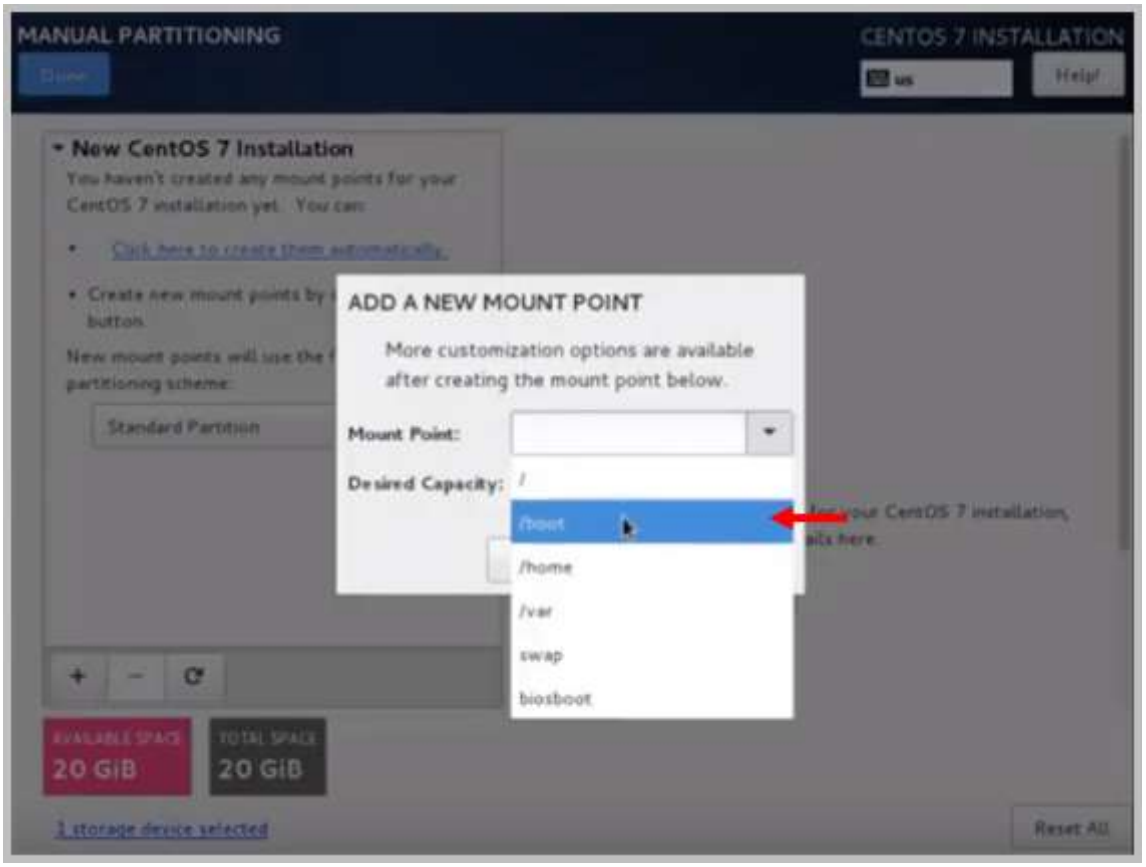
2. Then, select the radio button that says I will configure partitioning

3. Then, click on **Done**

- Next, you'll be taken to another window as shown below:

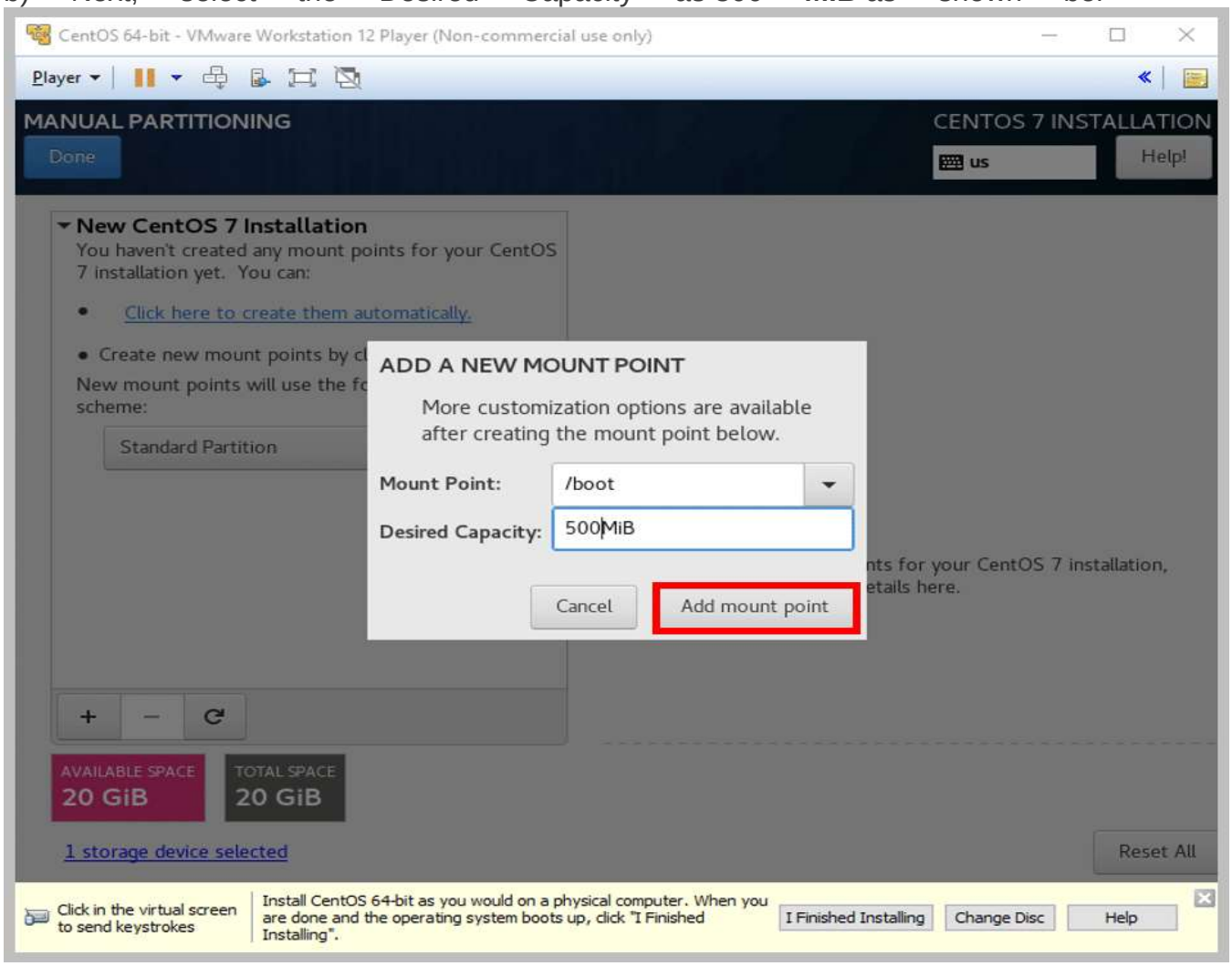


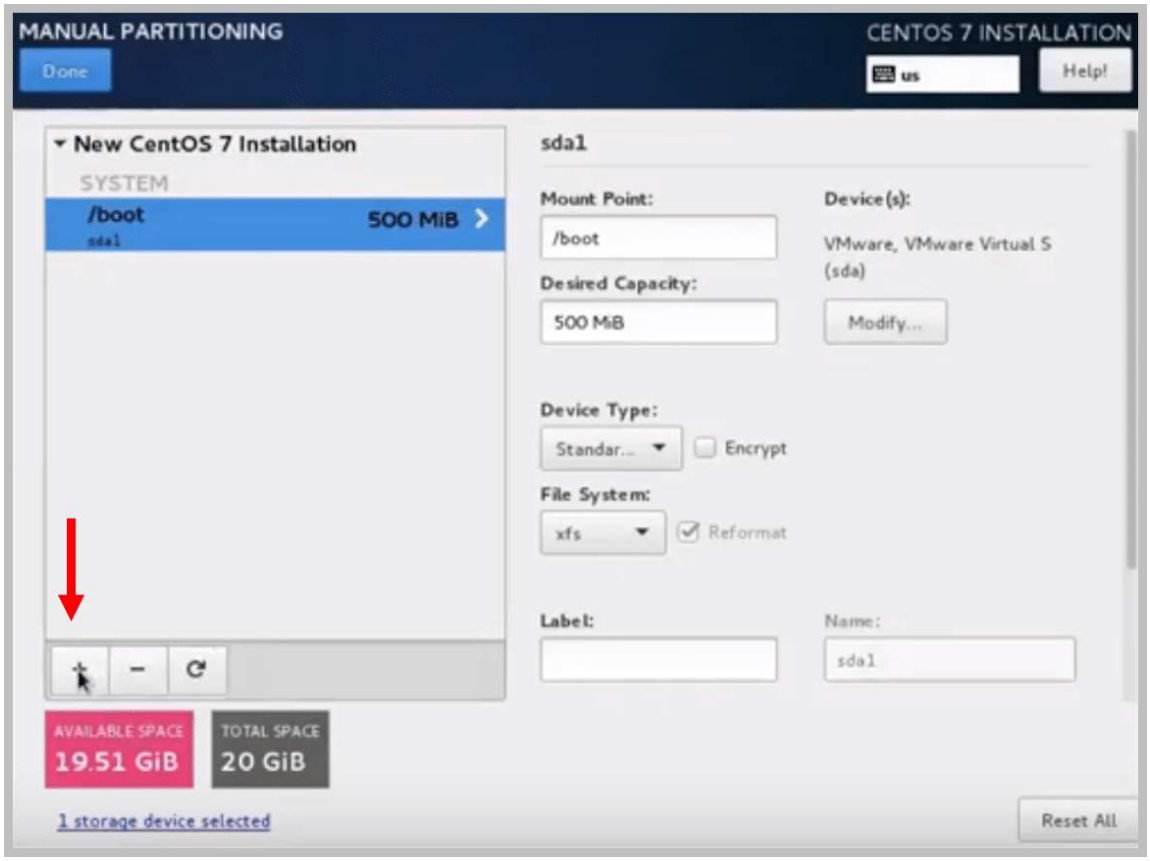
1. Select the partition scheme here as **Standard Partition**2. Now, you need to add three mount points here. For doing that, click on **'+'**


MANUAL PARTITIONING
CENTOS 7 INSTALLATION
Done
us
Help!
New CentOS 7 Installation
You haven't created any mount points for your CentOS 7 installation yet. You can:
Click here to create them automatically.
Create new mount points by
button.
New mount points will use the
partitioning scheme:
Standard Partition
ADD A NEW MOUNT POINT
More customization options are available after creating the mount point below.
Mount Point:
Desired Capacity:
/
/boot
/home
/var
swap
biosboot
AVAILABLE SPACE
20 GiB
TOTAL SPACE
20 GiB
1 storage device selected
Reset All

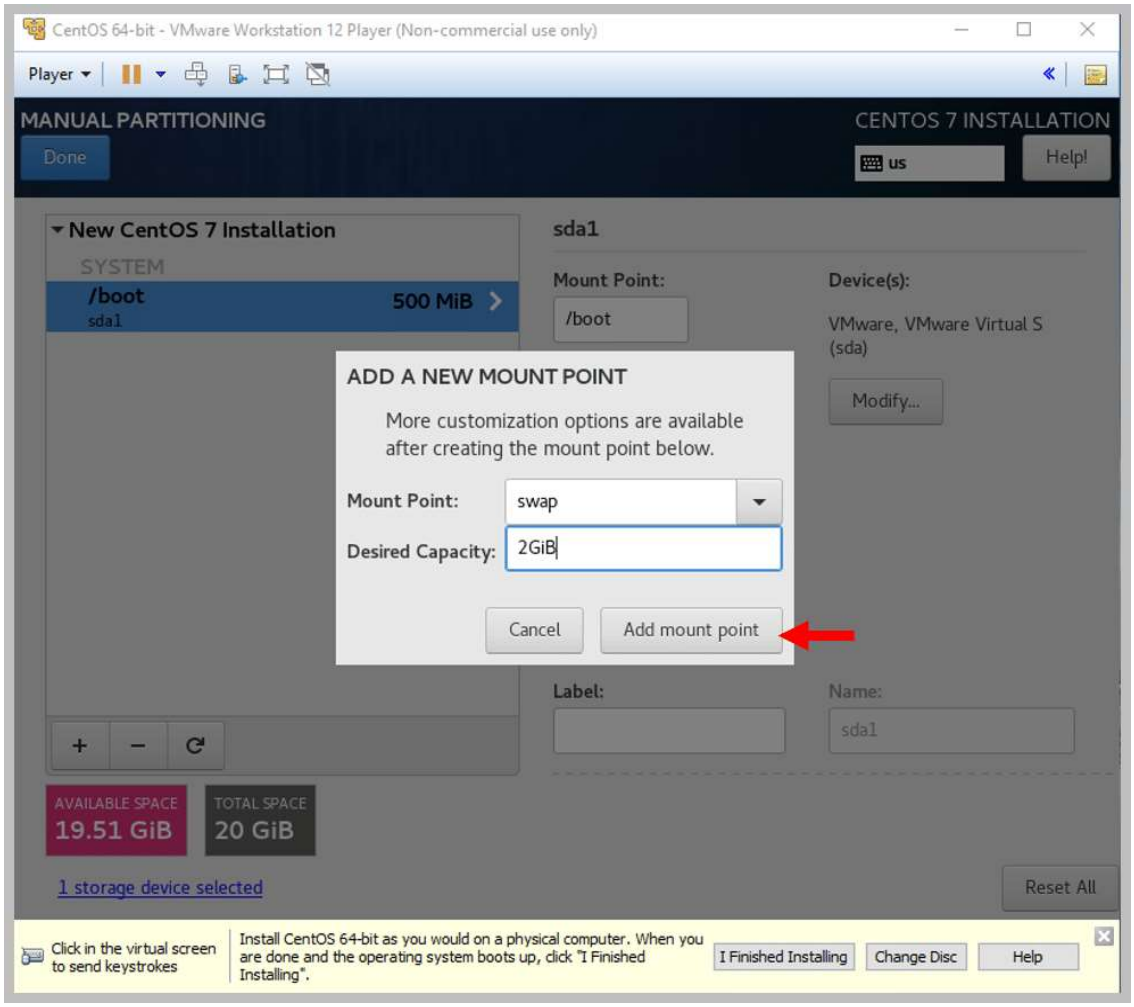
a) Select the Mount Point **/boot** as shown above

b) Next, select the Desired Capacity as **500 MiB** as shown bel



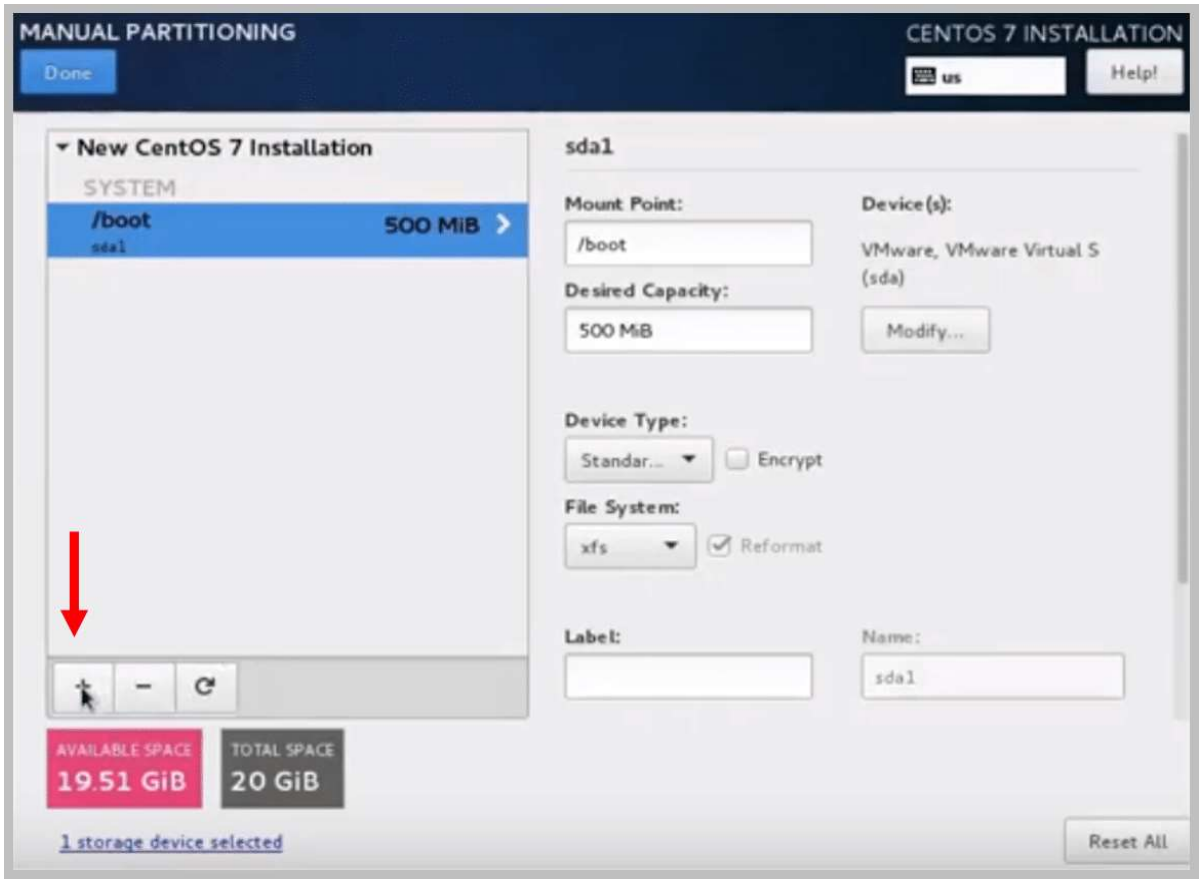


e) This time, select the Mount Point as **swap** and Desired Capacity as **2 GiB**

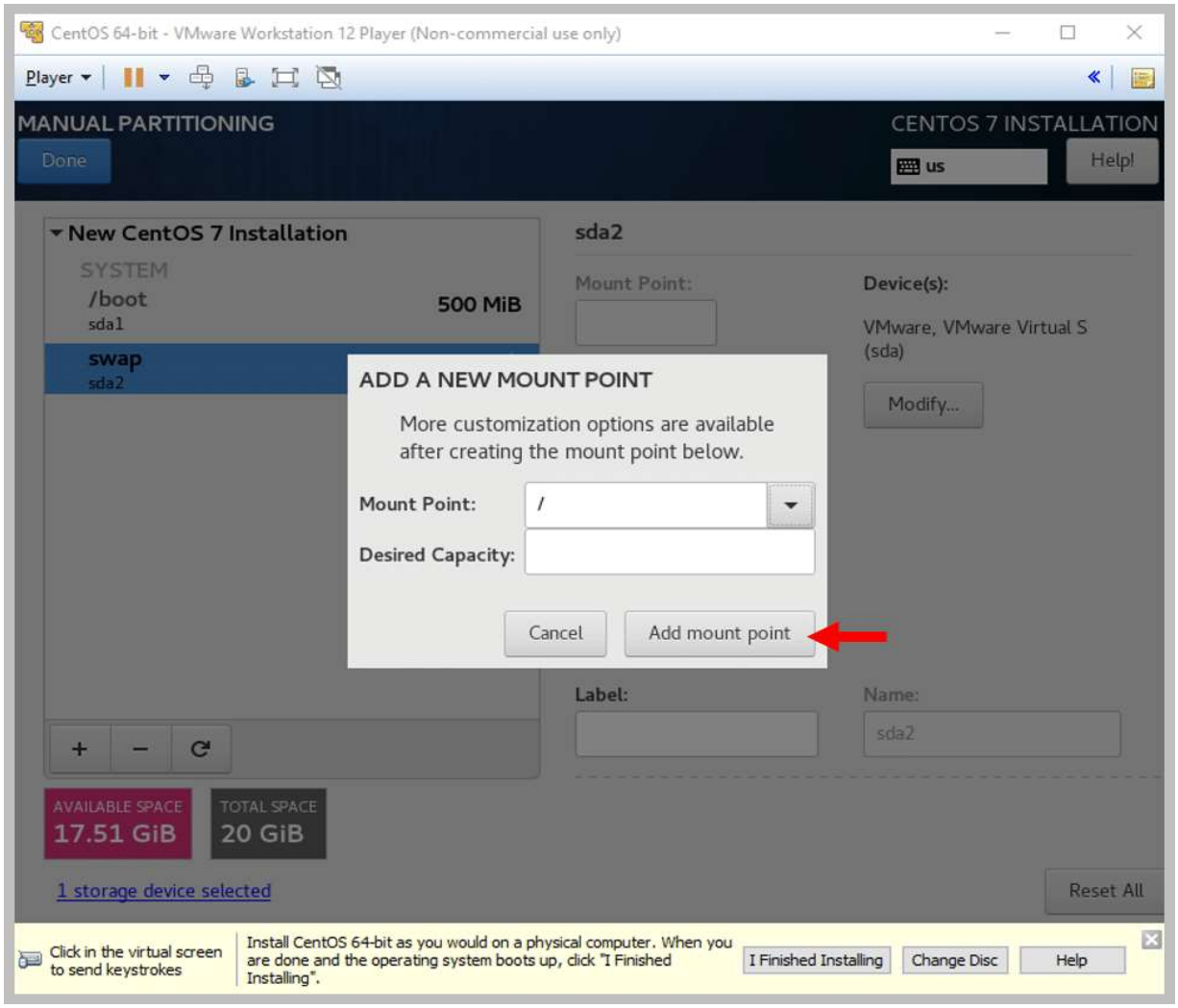


f) Click on **Add Mount Point**

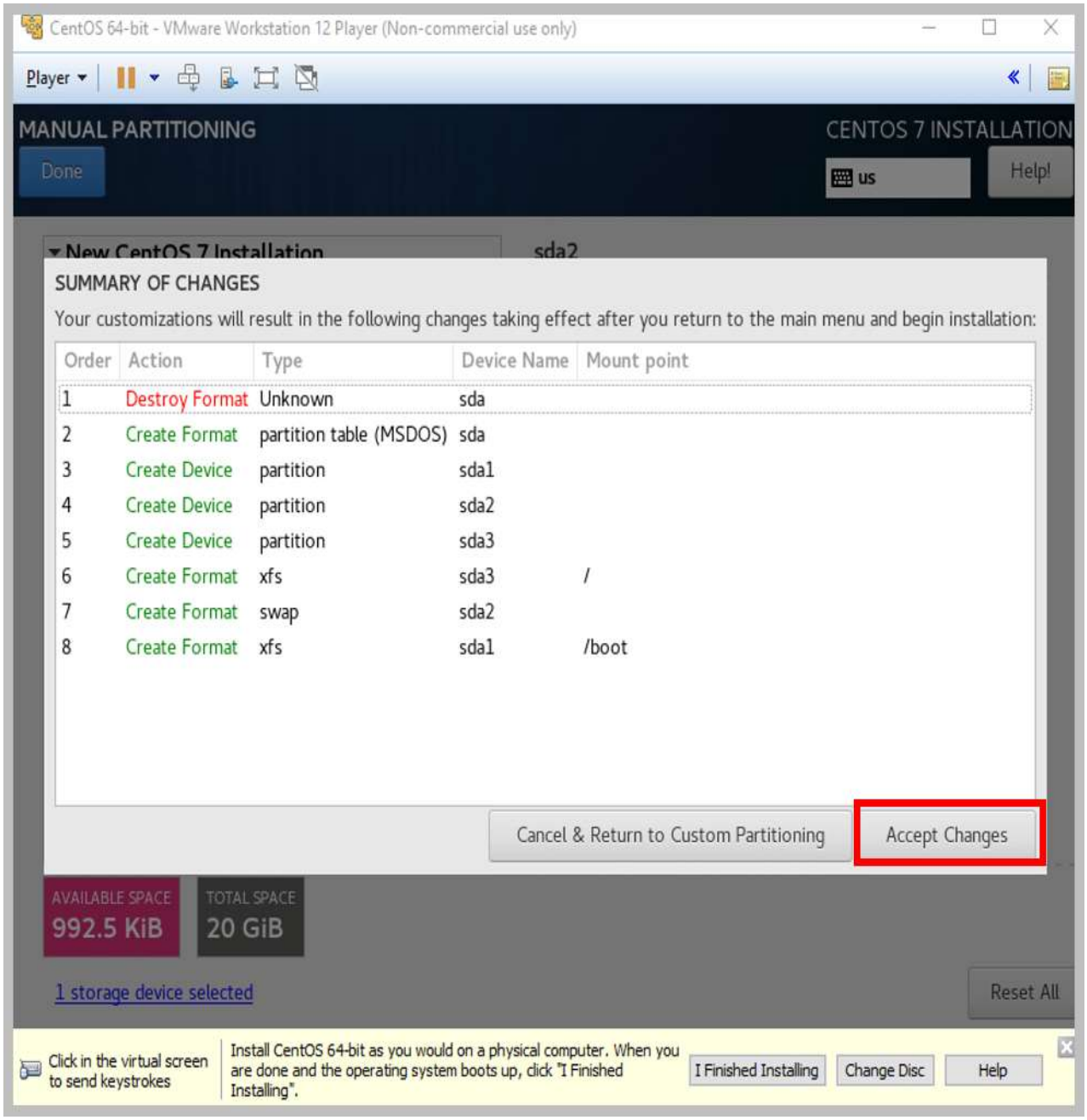
g) Now, to add the last Mount Point, click on **+** again



h) Add another Mount Point ‘/’ and click on Add Mount Point



i) Click on **Done**, and you will see the following window:

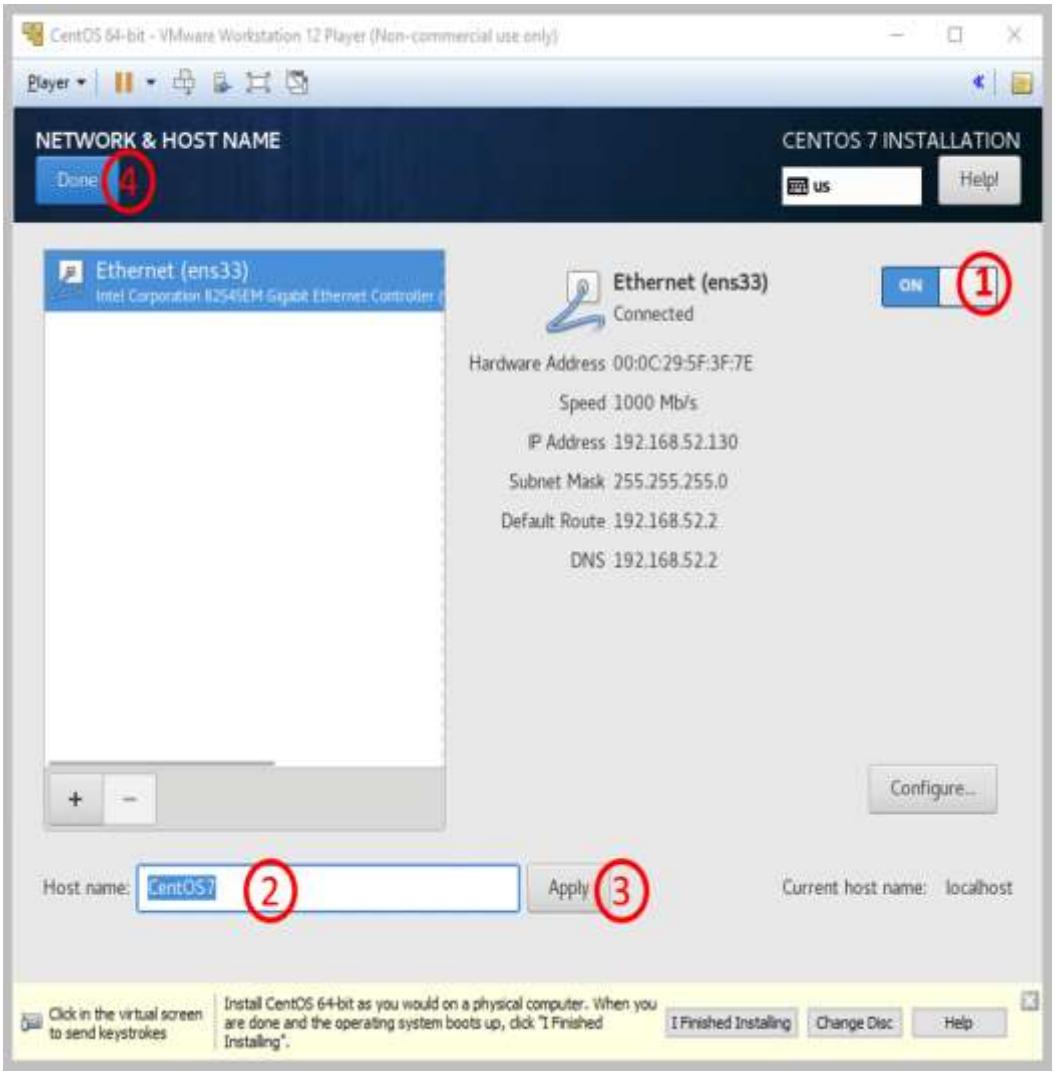


Note: This is just to make you aware of all the changes you had made in the partition of your drive

-
  -
    - Now, click on **Accept Changes** if you're sure about the partitions you have made

- Next, select **NETWORK & HOST NAME**

- You'll be taken to a window as shown below:



1. Set the **Ethernet settings** as **ON**
2. Change the **HOST name** if required
3. **Apply** the settings
4. Finally, click on **Done**

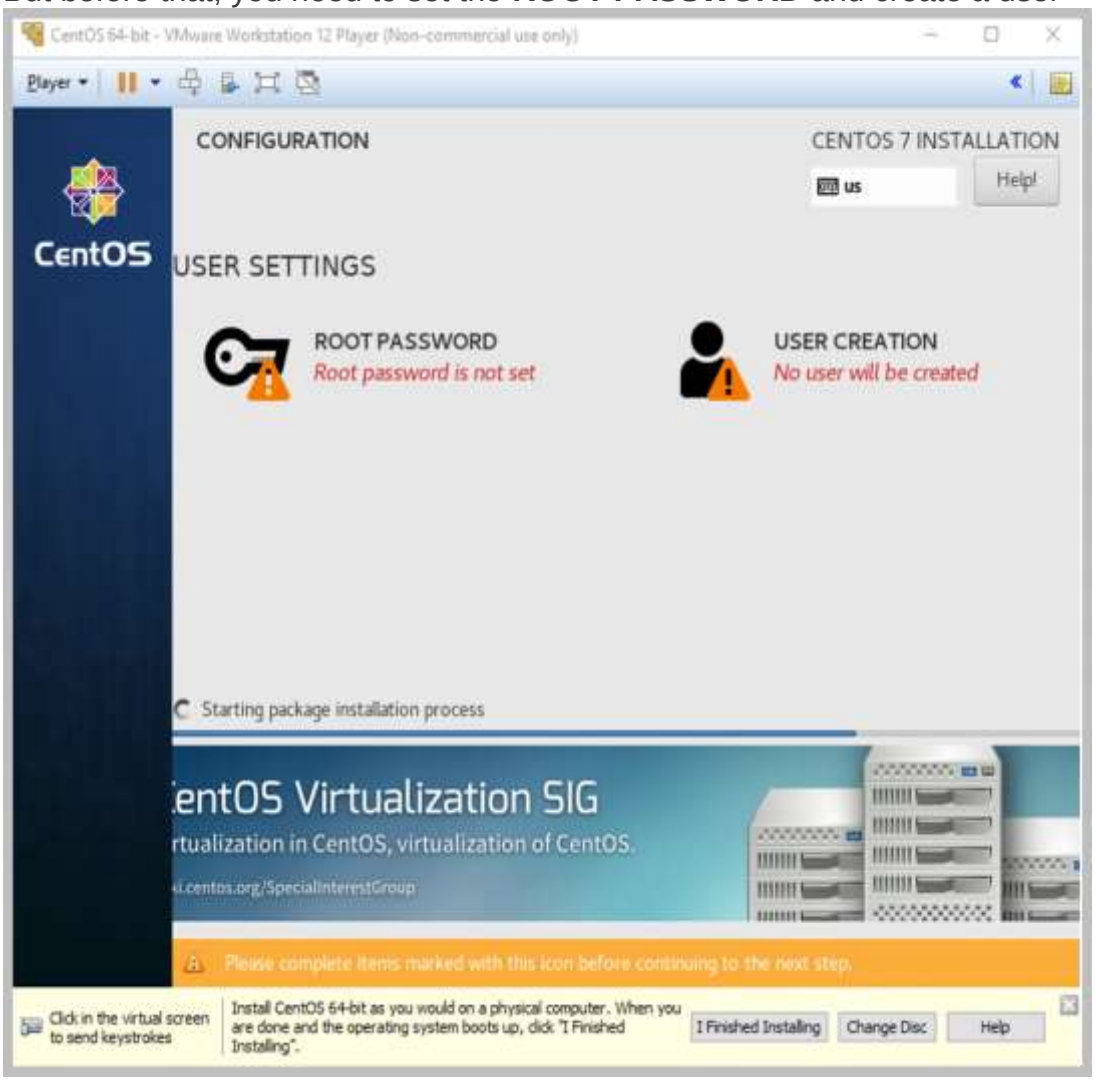
- Next, click on **Begin Installation**

If you have any doubts or queries related to Hadoop Installation, do post them on [Big Data Hadoop and Spark Community](#)!

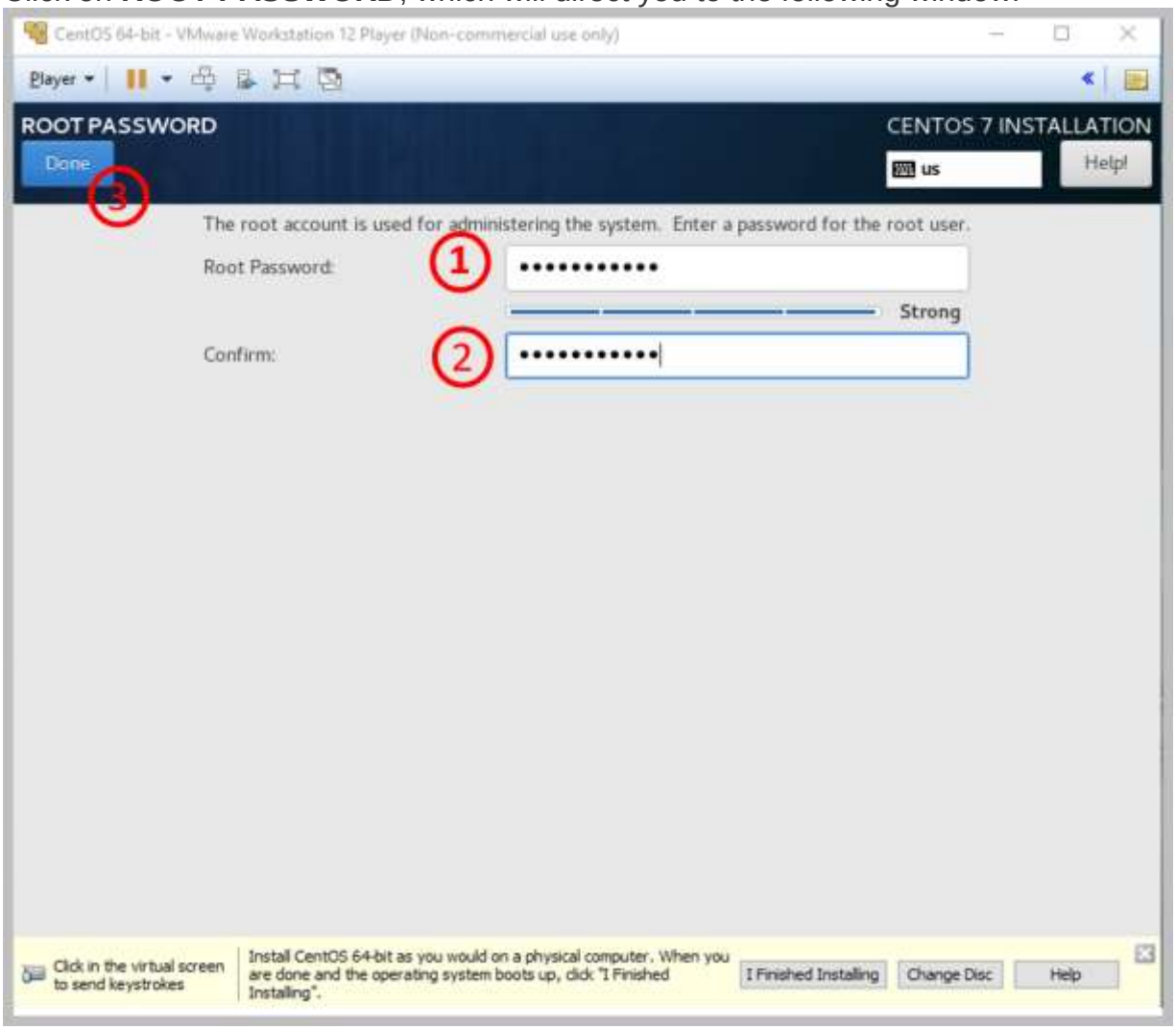
Step 6: Configuration

- Once you complete step 5, you will see the following window where the final installation process will be completed.

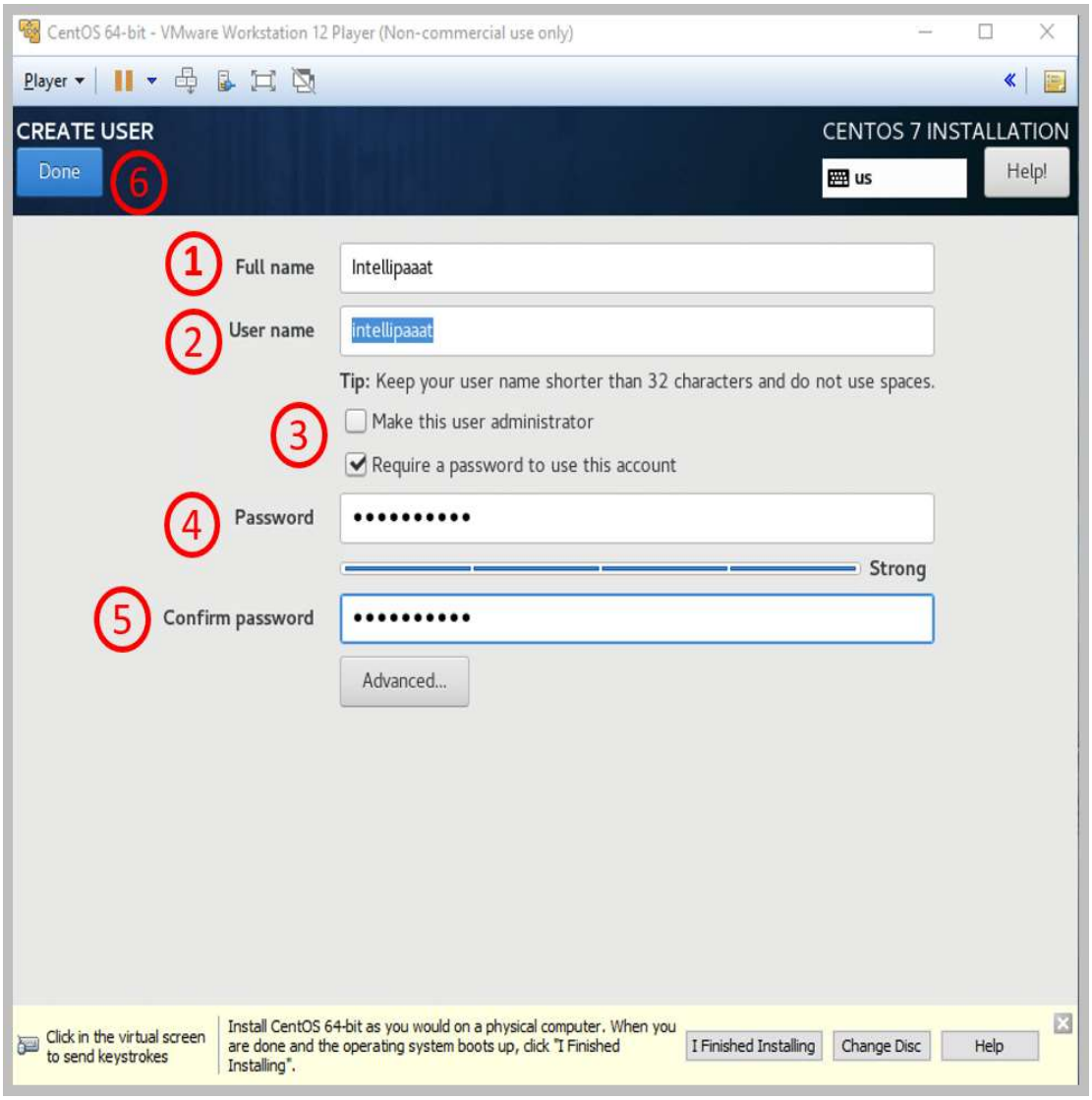
- But before that, you need to set the **ROOT PASSWORD** and create a user



- Click on **ROOT PASSWORD**, which will direct you to the following window:



1. **Enter** your root password here
2. **Confirm** the password
3. Click on **Done**
   - Now, click on **USER CREATION**, and you will be directed to the following window:



1. Enter your **Full name**. Here, I have entered Intellipaaat
2. Next, enter your **User name**; here, intellipaaat (This generally comes up automatically)
3. You can either make this password-based or make this a user administrator
4. Enter the password
5. Confirm your password
6. Finally, click on **Done**

-
    - You'll see the **Reboot** button, as seen below when your installation is done, which takes up to 20–30 minutes

In the next screen, you will see the installation process in progress

Note: It will take about 3 seconds for the CentOS to start.

-
    - Wait until a window pops up to accept your license info step **7: Setting up the License Information**

Practical -2

Aim – Implementing Map-Reduce Program for Word Count problem.

Steps-

1) First Open **Eclipse** -> then select **File** -> **New** -> **Java Project** ->Name it **WordCount** -> then **Finish**.
2) Next > libraries > Add external Jars > File system /usr/lib/Hadoop> Hadoop/client
   /Hadoop/client-0.20
3) Right click Project > Export > Jar file >
4) Right click on project name > new>class> wordcount.

Open terminal and enter the following commands .

1) hdfs dfs –ls /

2) sudo –u hdfs hadoop fs –mkdir /inputdirectory

3) hdfs dfs –ls /

4) cat > /home/cloudera/ProcessFile.txt

5) cat /home/cloudera/ProcessFile.txt

6) sudo –u hdfs hadoop fs –chmod -R 777 /inputdirceory.

7) hdfs dfs –ls /

8) sudo –u hdfs Hadoop fs –put /home/cloudera/ProcessFile.txt /inputdirectory.

9) hdfs dfs –ls /inputdirectory.

10) hadoop jar /home/cloudera /WordCount.java WordCount /inputdircetory /ProcessFile.txt

Hdfs dfs –ls /out1
Hdfs dfs -cat /out1/part-r-00000

Practical – 3

Aim Implementing Map-Reduce Program for Word Count problem.

Steps :

cat> /home/cloudera/input.csv

cat /home/cloudera/input.csv

pig -x local

lines = load '/home/cloudera/input.csv' as (line:chararray);

words = foreach lines GENERATE FLATTEN(TOKENIZE(line)) as woed;

grouped = GROUP words by woed;

wordcount = foreach grouped GENERATE group, COUNT(words);

dump wordcount;

Practical -4

Aim – Install HBase and use the HBase Data model store and retrieve data.

Steps :

//Start HBase

hbase shell

//HBase Commands

status

version,

table_help

whoami

//Data Definition Language

create ‘employee’, ’Name’, ’ID’, ’Designation’, ’Salary’, ’Department’

//Verify created table

list

//Disable single table

disable ‘employee’

scan ‘employee’

//or

is_disable ‘employee’

//Disable multiple tables

disable_all ‘e.*’

// Enabling table

enable‘employee’

//Or

is_enabled'employee'

//create new table

create‘student’, ‘name’, ‘age’, ‘course’

put ‘student’, ‘sharath’, ‘name:fullname’, ‘sharathkumar’

put ‘student’, ‘sharath’, ‘age:presentage’, ‘24’

put ‘student’, ‘sharath’, ‘course:pursuing’, ‘Hadoop’

put ‘student’, ‘shashank’, ‘name:fullname’, ‘shashank R

put ‘student’, ‘shashank’, ‘age:presentage’, ‘23’

put ‘student’, ‘shashank’, ‘course:pursuing’, ‘Java’

//Get Information

get ‘student’, ‘shashank’

get ‘student’, ‘sharath’

get ‘student’, ‘sharath’, ‘course’

get ‘student’, ‘shashank’, ‘course’

get ‘student’, ‘sharath’, ‘name’

//Scan

scan ‘student’

//Count

Count ‘student’

//Alter

alter ‘student’, NAME=>’name’, VERSIONS=>5
put ‘student’, ‘shashank’, ‘name:fullname’, ‘shashank Rao’
scan ‘student’

//Delete
delete ‘student’, ‘shashank’, ‘name:fullname’

Practical – 5

Aim – Install Hive and use Hive Create and store structured database.

Steps :

```
cat > /home/cloudera/employee.txt
        1~Sachine~Pune~Product Engineering~100000~Big Data
        2~Gaurav~Banglore~Sales~90000~CRM
        3~Manish~Chennai~Recruiter~125000~HR
        4~Bhushan~Hyderabad~Developer~50000~BFSI
cat  /home/cloudera/employee.txt
sudo -u hdfs hadoop fs -put /home/cloudera/employee.txt /inputdirectroy
hdfs dfs -ls /
hdfs dfs -ls /inputdirectory
hadoop fs -cat  /inputdirectory/employee.txt
hive
show databases;
create database organization;
show databases;
use organization;
show tables;
hive> create table employee(
    > id int,
    > name string,
    > city string,
    > department string,
    > salary int,
    > domain string)
    > row format delimited
```

```
    > fields terminated by '~';

show tables;
select * from employee;
show tables;
load data inpath '/inputdirectory/employee.txt' overwrite into table employee;
show tables;
select * from employee;
```

Practical -6

Aim – write a program to construct different type of kshingles for a given document .

```
install.packages("tm")
require("tm")
install.packages("devtools")
setwd("c:/msc/r-corpus/")
readinteger <- function()
{
  n <- readline(prompt = "Enter the value of k-1:")
  k <- as.integer(n)
  u1 <- readLines("data.txt")
  shingle <- 0
  i <-0
  while(i<nchar(u1)-k+1){
    shingle[i] <- substr(u1,start = i,stop = i+k)
    print(shingle[i])
    i=i+1
  }
}
if(interactive())readinteger()
```

Practical – 7

Aim – Write a program for measuring similarity among documents and detecting passages which have been reused.

Code –

```
C install.packages("tm")
require("tm")
install.packages("ggplot2")
install.packages("textreuse")
install.packages("devtools")
my.corpus <-Corpus(DirSource("c:/msc/r-corpus"))
my.corpus <-tm_map(my.corpus,removewords,stopWords("english"))
my.tdm <- TermDocumentMatrix(my.corpus)
my.dtm <- DocumentTermMatrix(my.corpus, control = list(weighting =
weightTfIdf, stopwords = TRUE))
my.df <- as.data.frame(inspect(my.tdm))
my.df.scale <- scale(my.df)
d <- dist(my.df.scale,method = "eucledian")
fit <- hclust(d,method = ward.D)
plot(fit)
```

Practical 8

Aim – Write a program to compute the n-moment for a given stream where n is given.

Code –

```
import java.io.*;
import java.util.*;
public class n_moment
{
	public static void main(String args[]) {
		int n=15;
		String stream[]=
{"a","b","c","b","d","a","c","d","a","b","d","c","a","a","b"};
		int
zero_moment=0,first_moment=0,second_moment=0,count=1,flag=0;
		ArrayList<Integer> arrlist=new ArrayList();
		System.out.println("Arraylist elements are::");
		for (int i=0;i<15;i++)
		{
			System.out.println(stream[i]+" ");
		}
		Arrays.sort(stream);
		for(int i=1;i<n;i++)
		{
			if(stream[i]==stream[i-1])
			{
				count++;

			}
			else
			{
				//System.out.println("Hello"+i);
				arrlist.add(count);
				count=1;
			}
		}
		arrlist.add(count);

		zero_moment=arrlist.size();
		System.out.println("\n\n\nValue of Zeroth moment for given
stream::"+zero_moment);
```

```
            for(int i=0;i<arrlist.size();i++)
            {
                  first_moment+=arrlist.get(i);
            }
            System.out.println("\n\nValue of First moment for given
stream::"+first_moment);
            for (int i=0;i<arrlist.size();i++)
            {
                  int j=arrlist.get(i);
                  second_moment+=(j*j);
            }
            System.out.println("\n\nValue of Second moment for given
stream::"+second_moment);

      }

}
```

Practical -9

Aim – Write a program to demonstrate the Alon-Matias- Szegedy Algorithm for second moments.

Code –

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */

package amsa;

/**
 *
 * @author shubham
 */
public class AMSA {

    /**
     * @param stream
     * @param XE
     * @param random
     * @param n
     * @return
     */

    public static int findCharCount(String stream,char XE, int random, int n){
        int countOccurance=0;
        for (int i = random; i<n; i++){
            if (stream.charAt(i)==XE){
                countOccurance++;
            }
        }
        return countOccurance;
    }

    public static int estimateValue(int XV1,int n){
        int ExpValue;
        ExpValue=n*(2*XV1-1);
        return ExpValue;
```

```
    }

    public static void main(String[] args) {
        // TODO code application logic here
        //int n=15;
        String stream="abcbdacdabdcaab";
        int n = stream.length();
        int random1=3,random2=8,random3=13;
        char XE1,XE2,XE3;
        int XV1,XV2,XV3;
        int ExpValuXE1,ExpValuXE2,ExpValuXE3;
        int apprSecondMomentValue;

        /*random1=Integer.parseInt(Math.random()+"");
        random2=Integer.parseInt(Math.random()+"");*/

        XE1=stream.charAt(random1-1);
        XE2=stream.charAt(random2-1);
        XE3=stream.charAt(random3-1);

        XV1=findCharCount(stream,XE1,random1-1,n);
        XV2=findCharCount(stream,XE2,random2-1,n);
        XV3=findCharCount(stream,XE3,random3-1,n);

        System.out.println(XE1+"="+XV1+"+"+XE2+"="+XV2+"
"+XE3+"="+XV3);

        ExpValuXE1=estimateValue(XV1,n);
        ExpValuXE2=estimateValue(XV2,n);
        ExpValuXE3=estimateValue(XV3,n);

        System.out.println("Expected Value for "+XE1+" is :: "+ExpValuXE1);
        System.out.println("Expected Value for "+XE2+" is :: "+ExpValuXE2);
        System.out.println("Expected Value for "+XE3+" is :: "+ExpValuXE3);


apprSecondMomentValue=(ExpValuXE1+ExpValuXE2+ExpValuXE3)/3;
        System.out.println("Approximate Second Moment value using Alon-
Matia-Szegedy is :: "+apprSecondMomentValue);


    }
}
```