1. Java Program to create TCP Sockets for C/S Communication

Server.java

```
import java.io.*;
import java.net.*;
public class MyServer {
public static void main(String[] args){
try{
ServerSocket ss=new ServerSocket(6666);
Socket s=ss.accept();//establishes connection
DataInputStream dis=new DataInputStream(s.getInputStream());
String str=(String)dis.readUTF();
System.out.println("message= "+str);
ss.close();
}catch(Exception e){System.out.println(e);}
}
}
```

Client.java

```
import java.io.*;
import java.net.*;
public class MyClient {
public static void main(String[] args) {
try{
Socket s=new Socket("localhost",6666);
DataOutputStream dout=new DataOutputStream(s.getOutputStream());
dout.writeUTF("Hello Server");
dout.flush();
dout.close();
s.close();
}catch(Exception e){System.out.println(e);}
}
}
```

2. Java Program to create UDP Sockets for C/S Communication

Server.java

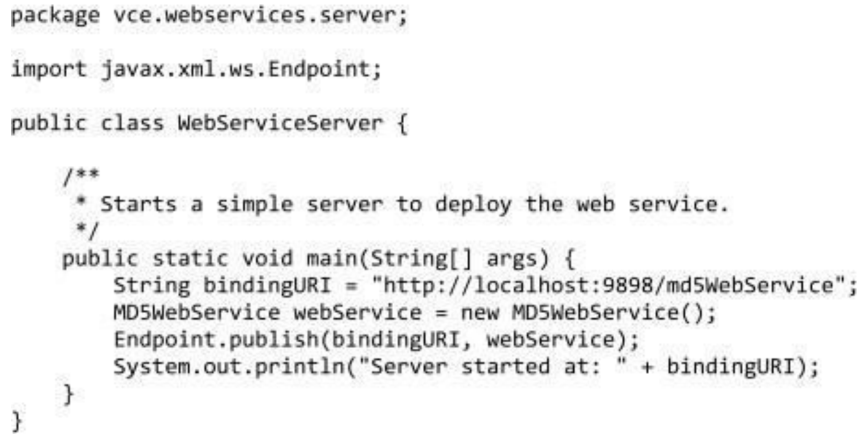
```
import java.io.IOException;
import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.InetAddress;
import java.net.SocketException;
public class udpBaseServer_2{
    public static void main(String[] args) throws IOException{
            DatagramSocket ds = new DatagramSocket(1234);
            byte[] receive = new byte[65535];
            DatagramPacket DpReceive = null;
            while (true){
                    DpReceive = new DatagramPacket(receive, receive.length);
                    ds.receive(DpReceive);
                    System.out.println("Client:-" + data(receive));
                    if (data(receive).toString().equals("bye")){
                            System.out.println("Client sent bye. ...EXITING");
                            break;
                    }
                    receive = new byte[65535];}}
    public static StringBuilder data(byte[] a){
            if (a == null) return null;
            StringBuilder ret = new StringBuilder();
            int i = 0;
            while (a[i] != 0){
                    ret.append((char) a[i]);
                    i++;
            }
            return ret;}}
```

Client.java

```
    import java.io.IOException;
    import java.net.DatagramPacket;
    import java.net.DatagramSocket;
    import java.net.InetAddress;
    import java.util.Scanner;
    public class udpBaseClient_2{
      public static void main (String args[]) throws IOException{
        Scanner sc = new Scanner (System.in);
        DatagramSocket ds = new DatagramSocket ();
        InetAddress ip = InetAddress.getLocalHost ();
        byte buf[] = null;
        while (true){
              String inp = sc.nextLine ();
                buf = inp.getBytes ();
              DatagramPacket DpSend = new DatagramPacket (buf, buf.length, ip, 1234);
                ds.send (DpSend);
              if (inp.equals ("bye"))
                  break;
        } }}
```

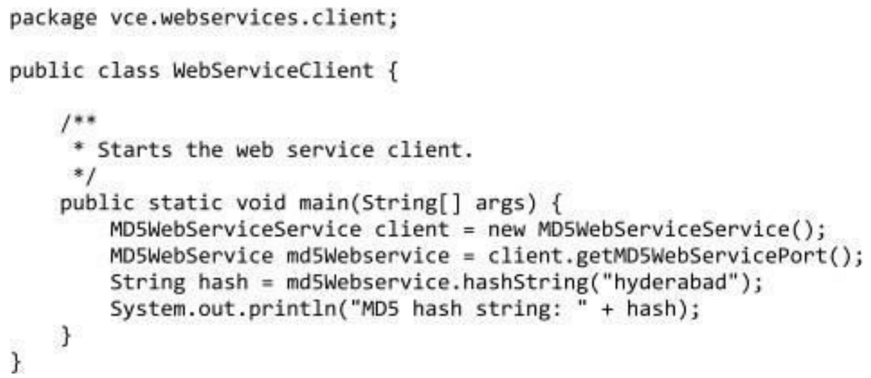
3. Design a Web service using Simple Object Access Protocol (SOAP)

Webserviceserver.java



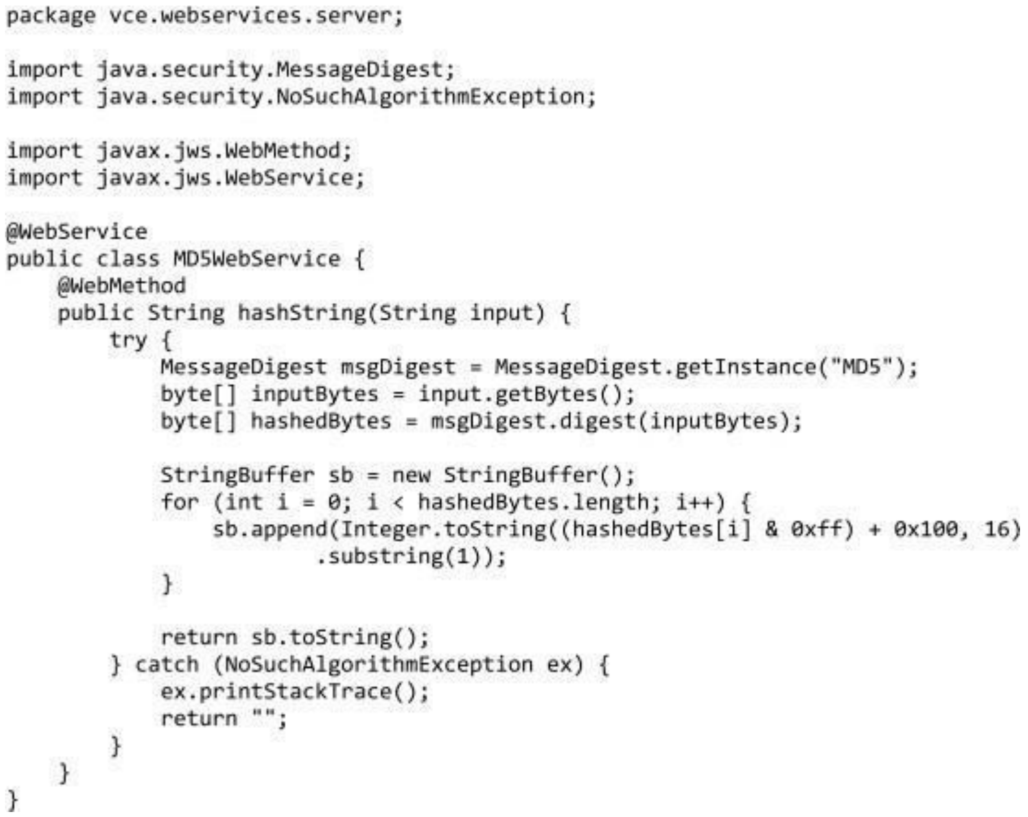
```
package vce.webservices.server;

import javax.xml.ws.Endpoint;

public class WebServiceServer {

    /**
     * Starts a simple server to deploy the web service.
     */
    public static void main(String[] args) {
        String bindingURI = "http://localhost:9898/md5WebService";
        MD5WebService webService = new MD5WebService();
        Endpoint.publish(bindingURI, webService);
        System.out.println("Server started at: " + bindingURI);
    }
}
```

webserviceClient.java



```
package vce.webservices.client;

public class WebServiceClient {

    /**
     * Starts the web service client.
     */
    public static void main(String[] args) {
        MD5WebServiceService client = new MD5WebServiceService();
        MD5WebService md5Webservice = client.getMD5WebServicePort();
        String hash = md5Webservice.hashString("hyderabad");
        System.out.println("MD5 hash string: " + hash);
    }
}
```

Md5webservice.java



```
package vce.webservices.server;

import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;

import javax.jws.WebMethod;
import javax.jws.WebService;

@WebService
public class MD5WebService {
    @WebMethod
    public String hashString(String input) {
        try {
            MessageDigest msgDigest = MessageDigest.getInstance("MD5");
            byte[] inputBytes = input.getBytes();
            byte[] hashedBytes = msgDigest.digest(inputBytes);

            StringBuffer sb = new StringBuffer();
            for (int i = 0; i < hashedBytes.length; i++) {
                sb.append(Integer.toString((hashedBytes[i] & 0xff) + 0x100, 16)
                        .substring(1));
            }

            return sb.toString();
        } catch (NoSuchAlgorithmException ex) {
            ex.printStackTrace();
            return "";
        }
    }
}
```

4. Developing a Multi chat application using Java.

```
import java.io.*;
import java.util.*;
import java.net.*;
public class Server{
    static Vector<ClientHandler> ar = new Vector<>();
    static int i = 0;
    public static void main(String[] args) throws IOException{
        ServerSocket ss = new ServerSocket(1234);
        Socket s;
        while (true){
            s = ss.accept();
            System.out.println("New client request received : " + s);
            DataInputStream dis = new DataInputStream(s.getInputStream());
            DataOutputStream dos = new DataOutputStream(s.getOutputStream());
            System.out.println("Creating a new handler for this client...");
            ClientHandler mtch = new ClientHandler(s,"client " + i, dis, dos);
            Thread t = new Thread(mtch);
            System.out.println("Adding this client to active client list");
            ar.add(mtch);
            t.start();
            i++;
        }
    }
}

class ClientHandler implements Runnable{
    Scanner scn = new Scanner(System.in);
    private String name;
    final DataInputStream dis;
    final DataOutputStream dos;
    Socket s;
    boolean isloggedin;
    public ClientHandler(Socket s, String name, DataInputStream dis, DataOutputStream dos) {
        this.dis = dis;
        this.dos = dos;
        this.name = name;
        this.s = s;
        this.isloggedin=true;
    }
    public void run() {
        String received;
        while (true){
            try{
                received = dis.readUTF();
                System.out.println(received);
                if(received.equals("logout")){
                    this.isloggedin=false;
                    this.s.close();
                    break;
                }
```

```
                        StringTokenizer st = new StringTokenizer(received, "#");
                        String MsgToSend = st.nextToken();
                        String recipient = st.nextToken();
                        for (ClientHandler mc : Server.ar){
                                if (mc.name.equals(recipient) && mc.isloggedin==true){
                                        mc.dos.writeUTF(this.name+" : "+MsgToSend);
                                        break;
                                        }
                                }
                        } catch (IOException e) {
                                e.printStackTrace();
                    }
                }
                try{
                        this.dis.close();
                        this.dos.close();

                }catch(IOException e){
                        e.printStackTrace();
                  }
            }
    }
```

5. Map reduce and Hadoop word frequency count

WCDriver.java

```
import java.io.IOException;
import org.apache.hadoop.conf.Configured;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.FileInputFormat;
import org.apache.hadoop.mapred.FileOutputFormat;
import org.apache.hadoop.mapred.JobClient;
import org.apache.hadoop.mapred.JobConf;
import org.apache.hadoop.util.Tool;
import org.apache.hadoop.util.ToolRunner;
public class WCDriver extends Configured implements Tool {
        public int run(String args[]) throws IOException{
                if (args.length < 2)   return -1;
                JobConf conf = new JobConf(WCDriver.class);
                FileInputFormat.setInputPaths(conf, new Path(args[0]));
                FileOutputFormat.setOutputPath(conf, new Path(args[1]));
                conf.setMapperClass(WCMapper.class);
                conf.setReducerClass(WCReducer.class);
                conf.setMapOutputKeyClass(Text.class);
                conf.setMapOutputValueClass(IntWritable.class);
                conf.setOutputKeyClass(Text.class);
                conf.setOutputValueClass(IntWritable.class);
                JobClient.runJob(conf);
                return 0;
        }
        public static void main(String args[]) throws Exception{
                int exitCode = ToolRunner.run(new WCDriver(), args);
                System.out.println(exitCode);
        }
}
```

WCMapper.java

```
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.MapReduceBase;
import org.apache.hadoop.mapred.Mapper;
import org.apache.hadoop.mapred.OutputCollector;
import org.apache.hadoop.mapred.Reporter;

public class WCMapper extends MapReduceBase implements Mapper<LongWritable, Text, Text,
IntWritable> {
        public void map(LongWritable key, Text value, OutputCollector<Text, IntWritable> output,
Reporter rep) throws IOException
        {
                String line = value.toString();
                for (String word : line.split(" ")){
                        if (word.length() > 0)
                                output.collect(new Text(word), new IntWritable(1));
                }
        }
}
```

WCReducer.java

```
import java.io.IOException;
import java.util.Iterator;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.MapReduceBase;
import org.apache.hadoop.mapred.OutputCollector;
import org.apache.hadoop.mapred.Reducer;
import org.apache.hadoop.mapred.Reporter;

public class WCReducer extends MapReduceBase implements Reducer<Text, IntWritable, Text,
IntWritable> {
        public void reduce(Text key, Iterator<IntWritable> value, OutputCollector<Text, IntWritable>
output, Reporter rep) throws IOException{
                int count = 0;
                while (value.hasNext()){
                        IntWritable i = value.next();
                        count += i.get();
                }
                output.collect(key, new IntWritable(count));
        }
}
```

Source Code:

Client.java

```
package TwoPC;
import java.io.*;
import java.net.*;
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import java.sql.*;

class  DBConnector {
        public static Connection getDBConnection(String dsn) throws Exception {
                Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
                return DriverManager.getConnection("jdbc:odbc:"+dsn);
        }
}
public class Client extends JFrame implements ActionListener{
        JButton b1,b2,b4,b5;
        JPanel p1,p2;
        JTextField t1;
        JLabel l1;
        ServerSocket ss;
        Socket s;
        DataOutputStream output;
        DataInputStream input;
        Connection con;
        Statement stmt;
        String serverMessage="Prepared";
        int port = 8890;
        String groupIP = "228.5.6.200";
        Client(){
                b1=new JButton("Prepared");
                b2=new JButton("NotPrepared");
                b4=new JButton("Execute");
                b5=new JButton("Exit");
                t1=new JTextField("",35);
                l1=new JLabel("SQL");
                p1=new JPanel();
                p2=new JPanel();
                p1.setLayout(new FlowLayout());
                p1.add(l1);
                p1.add(t1);
                p2.add(b1);
                p2.add(b2);
                p2.add(b4);
                p2.add(b5);
                add(p1);
                add(p2,"South");
                setSize(600,300);
                setTitle("Two Phase Commit Protocol: Client");
                b1.addActionListener(this);
                b2.addActionListener(this);
                b4.addActionListener(this);
                b5.addActionListener(this);
                setVisible(true);
                setDefaultCloseOperation(EXIT_ON_CLOSE);
                MulticastSocket ms =null;
                InetAddress group ;
                try {
   s = new Socket("localhost",8088);
   System.out.println("Client Connected");
   output=new DataOutputStream(s.getOutputStream());
   input=new DataInputStream(s.getInputStream());
   con = DBConnector.getDBConnection("my2pcdsn");
   stmt = con.createStatement();
   con.setAutoCommit(false);
   ms = new MulticastSocket(port);
   group= InetAddress.getByName(groupIP);
   ms.joinGroup(group);
   byte[] buffer = new byte[1024];
   output.writeUTF("NotPrepared");
   while (true) {
          DatagramPacket serMsg= new DatagramPacket(buffer, buffer.length);
          ms.receive(serMsg);
          String commitMsg = new String (serMsg.getData()).trim();
          if (commitMsg.equals("commit")) {
          System.out.println("Received "+commitMsg);
          con.commit();
          t1.setText("Transactions Committed");
          System.out.println("Transactions Committed");
          }
     }
 }
                catch (ConnectException ce)
                {
                        ce.printStackTrace();
                        System.exit(0);
                }
                catch (Exception e)
                {
                        e.printStackTrace();
                }
        }

        public void actionPerformed(ActionEvent ae){
                try
                {
                        String str=ae.getActionCommand();
```

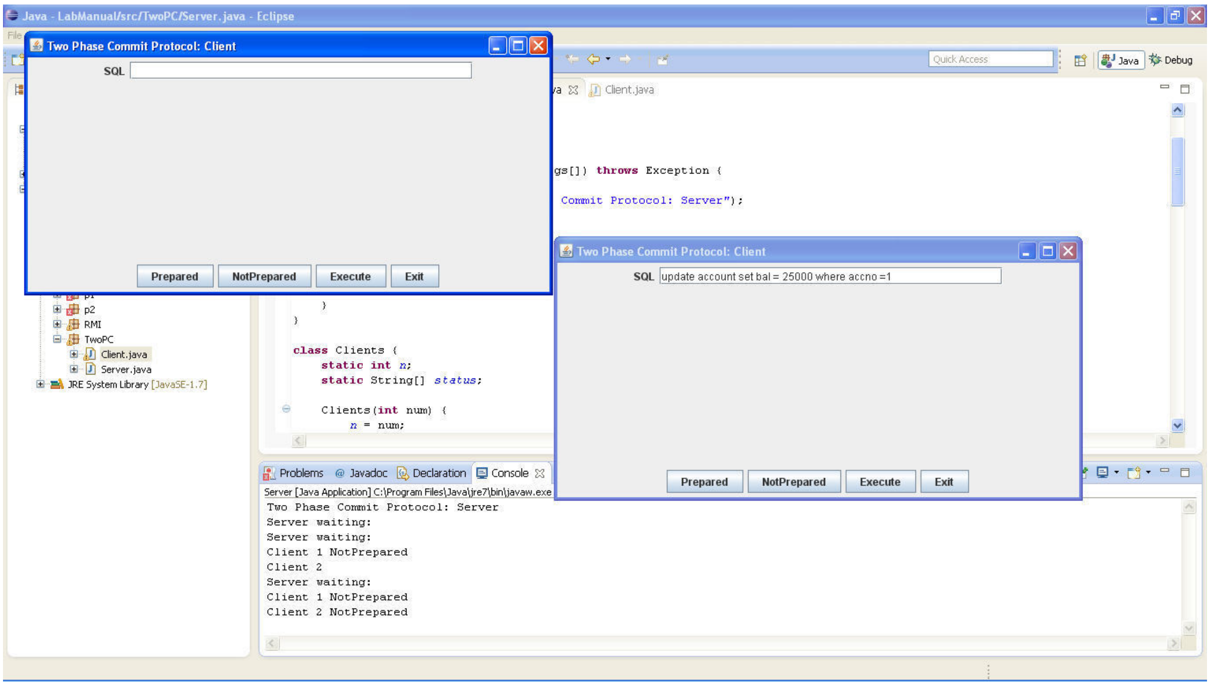
```
            if(str.equals("Execute")){
                String query = t1.getText();
                stmt.executeUpdate(query);
                t1.setText("Query Executed (NotPrepared)");
                output.writeUTF("NotPrepared");

            }

            if(str.equals("Prepared")){
                output.writeUTF("Prepared");
                t1.setText(input.readUTF());
            }

            if(str.equals("NotPrepared")){
            output.writeUTF("NotPrepared");
            t1.setText("NotPrepared");
            }

            if(str.equals("Exit")){
                output.writeUTF("Prepared");
                stmt.close();
                con.close();
                System.exit(0);
            }
        }
        catch(Exception e){
JLabel errorFields = new JLabel("<HTML><FONT COLOR =
BLUE>"+e.getMessage()+"</FONT></HTML>");
            JOptionPane.showMessageDialog(null,errorFields);
            e.printStackTrace();
        }
    }
    public static void main(String args[]){
        Client c=new Client();
    }
}
```

Server.java

```
package TwoPC;

import java.io.*;
import java.net.*;

public class Server {

    public static ServerSocket ss;

    public Server() {
    }
    public static void main(String args[]) throws Exception {
        ss = new ServerSocket(8088);
        System.out.println("Two Phase Commit Protocol: Server");
        new Clients(2);
        while (true) {
            System.out.println("Server waiting: ");
            Socket s = ss.accept();
            new Coordinator(s);
        }
    }
}

class Clients {
    static int n;
    static String[] status;

    Clients(int num) {
        n = num;
        status = new String[n];
        for (int j = 0; j < n; j++) {
            status[j] = new String("");
        }
    }
}

class Coordinator implements Runnable {

    public static int i = -1;
    int flag = 1;
    Socket s;
    Thread t;
    MulticastSocket ms = null;
    InetAddress group;
    DataInputStream input;
    DataOutputStream output;
    int port = 8890;
    String groupIP = "228.5.6.200";
    Coordinator(Socket c) {
        s = c;
        try {
            input = new DataInputStream(s.getInputStream());
            output = new DataOutputStream(s.getOutputStream());
            ms = new MulticastSocket(port);
            group = InetAddress.getByName(groupIP);
            ms.joinGroup(group);
        } catch (Exception e) {
            e.printStackTrace();
        }
        t = new Thread(this);
        t.start();
        i++;
```

```
      }

      public void run() {
          int index = i;
          String clientSattus;
          try {
              while (true) {
                  clientSattus = input.readUTF();
                  if (clientSattus.equalsIgnoreCase("Prepared")) {
                      output.writeUTF("Wait for others to prepare");
                  }
                  Clients.status[index] = new String(clientSattus);
                  for (int k = 0; k < Clients.n; k++) {
                      System.out.println("Client " + (k + 1) + " " +
  Clients.status[k]);
                      if (Clients.status[k].equalsIgnoreCase("Prepared"))
  {
                          continue;
                      } else {
                          flag = 0;
                      }
                  }
                  if (flag == 1) {
                      byte[] msg = new String("commit").getBytes();
                      DatagramPacket msgpack = new
  DatagramPacket(msg, msg.length, group, port);
                      ms.send(msgpack);
                      System.out.println("Commit message sent to clients:
  " + new String(msg));
                  }
                  flag = 1;
              }
          } catch (Exception e) {
              e.printStackTrace();
          }
      }
}
```

OUTPUT:



SQL update account set bal = 25000 where accno =1

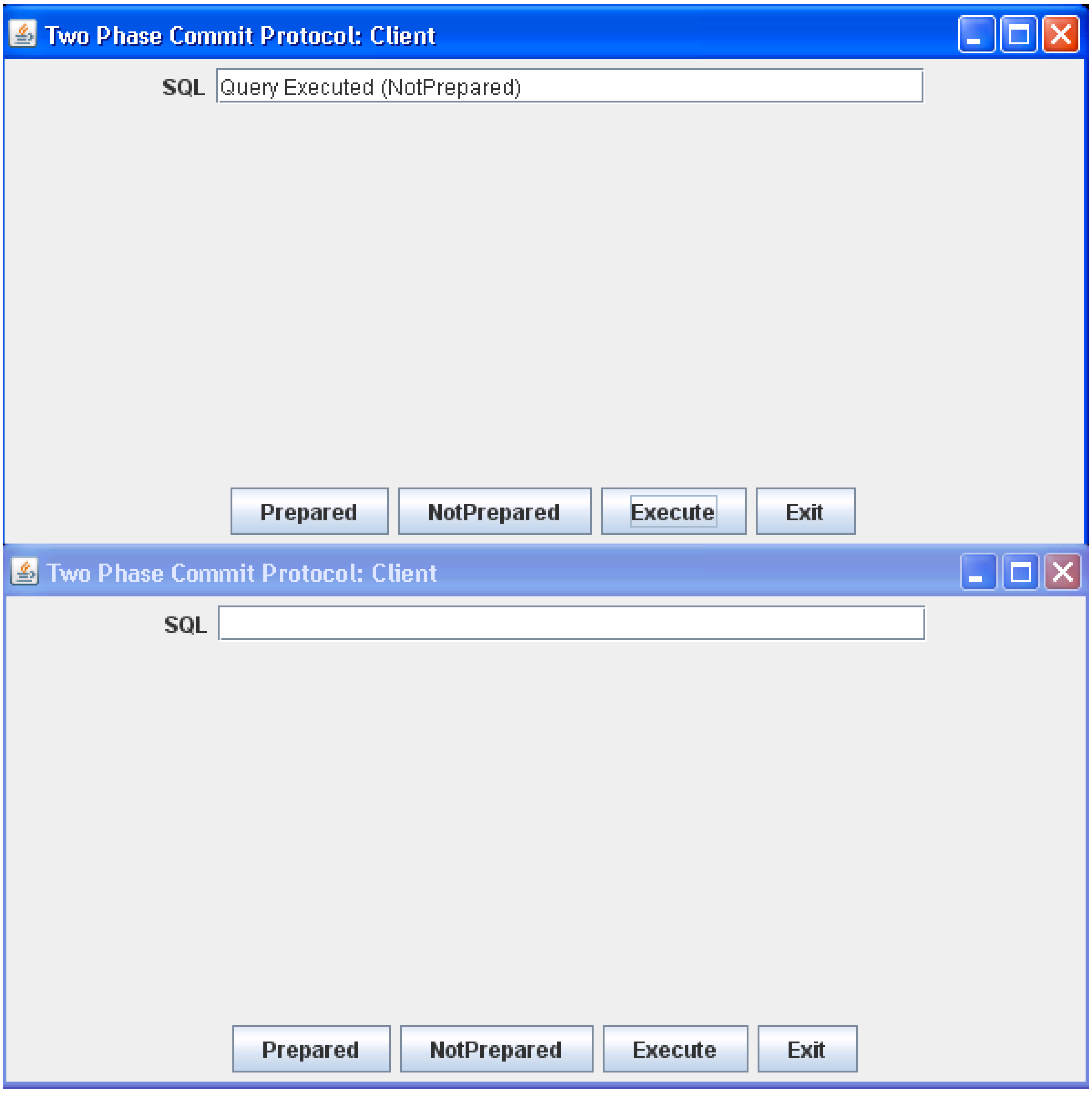
```
gs[]) throws Exception {

 Commit Protocol: Server");
```

```
        }
}

class Clients {
    static int n;
    static String[] status;

    Clients(int num) {
        n = num;
```

Server [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe

```
Two Phase Commit Protocol: Server
Server waiting:
Server waiting:
Client 1 NotPrepared
Client 2
Server waiting:
Client 1 NotPrepared
Client 2 NotPrepared
```

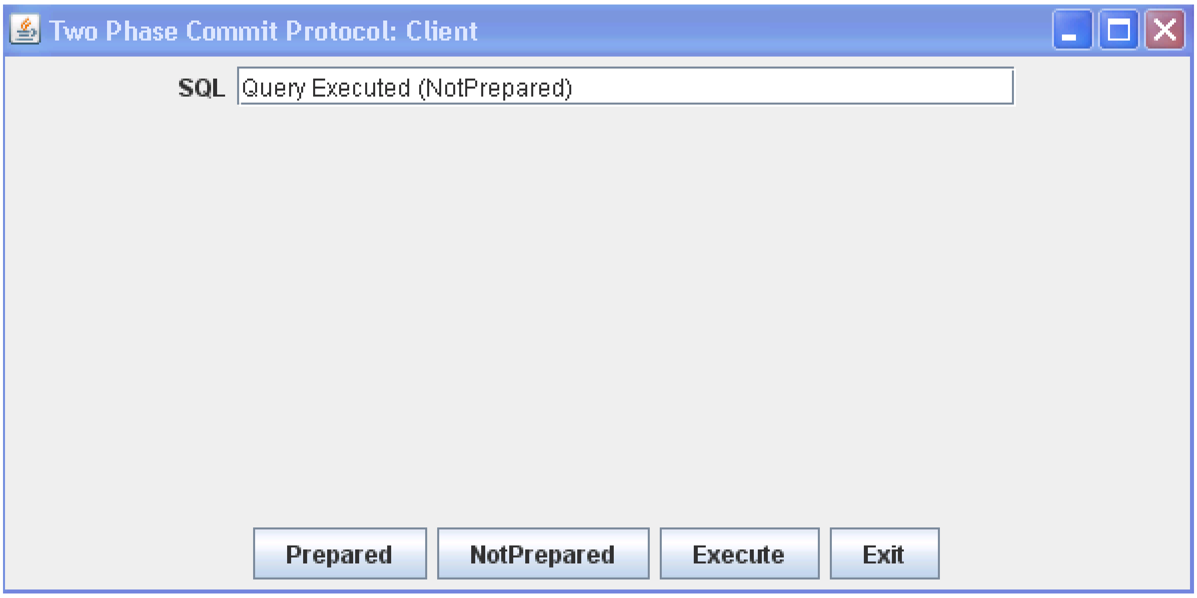
Client 1 executes Query



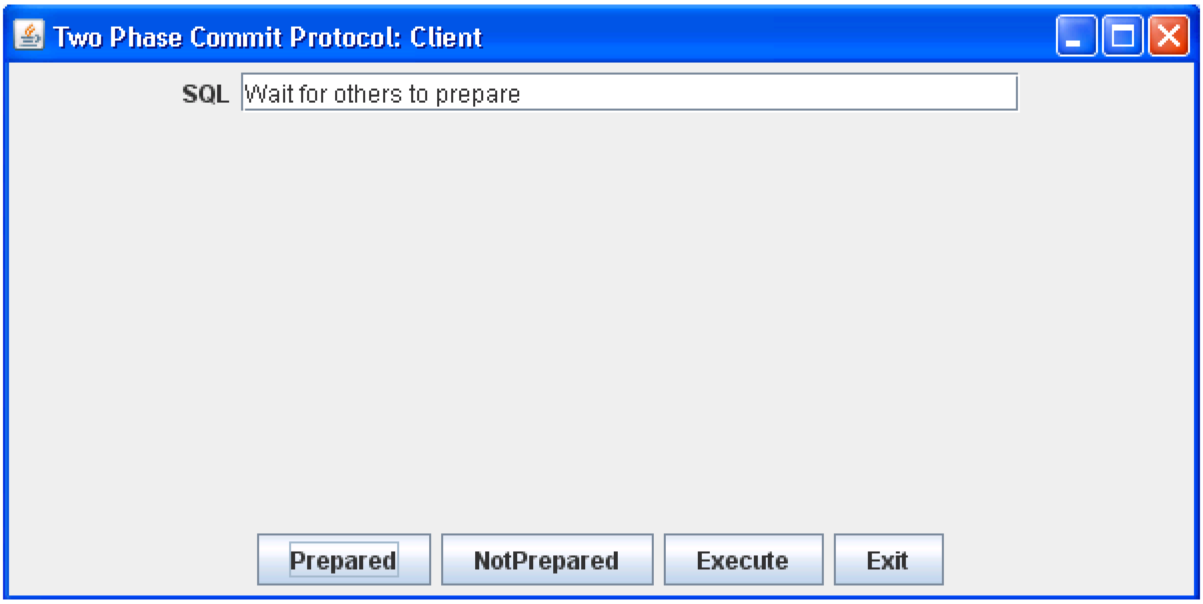
Two Phase Commit Protocol: Client
SQL
Prepared
NotPrepared
Execute
Exit
Two Phase Commit Protocol: Client
SQL
Prepared
NotPrepared
Execute
Exit

SQL: Query Executed (NotPrepared)

Client 2 says Prepared

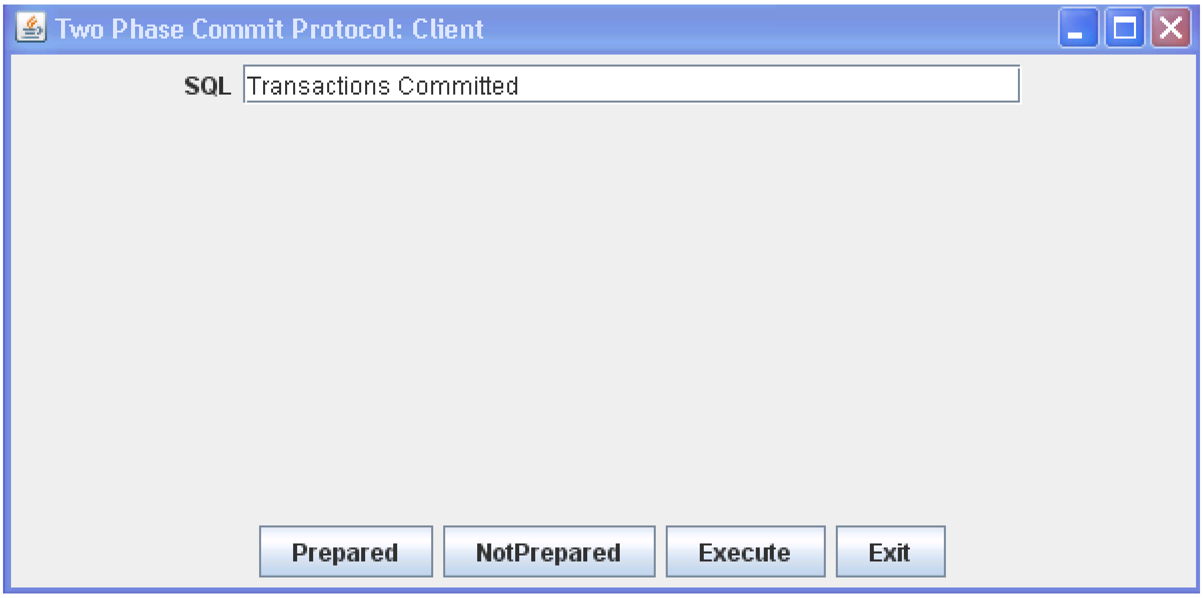


SQL: Query Executed (NotPrepared)

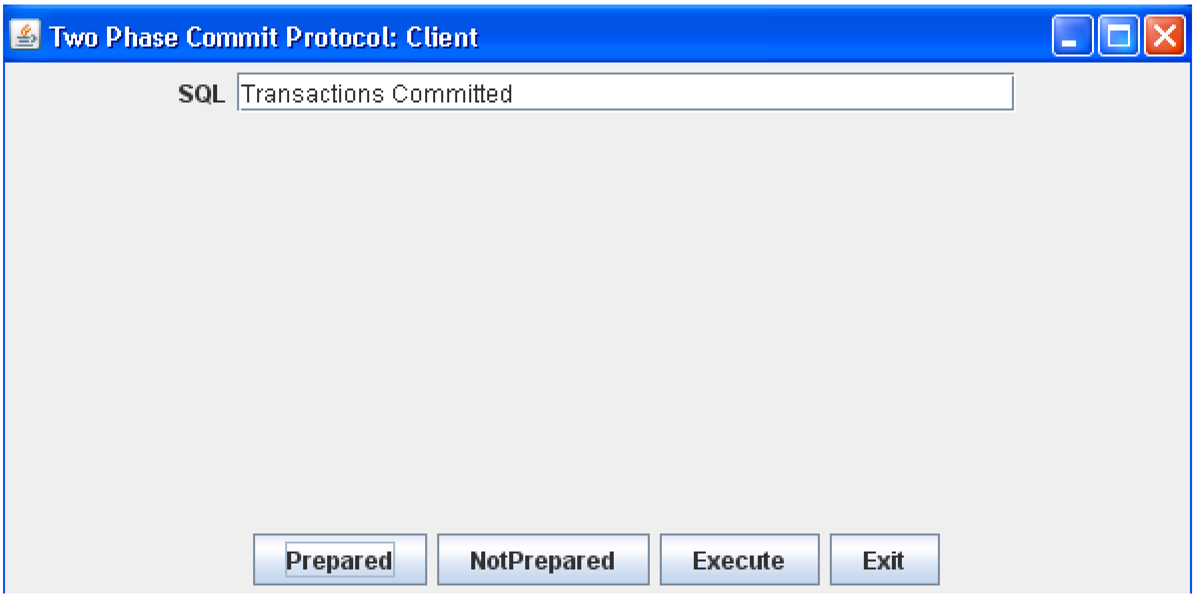


SQL: Wait for others to prepare

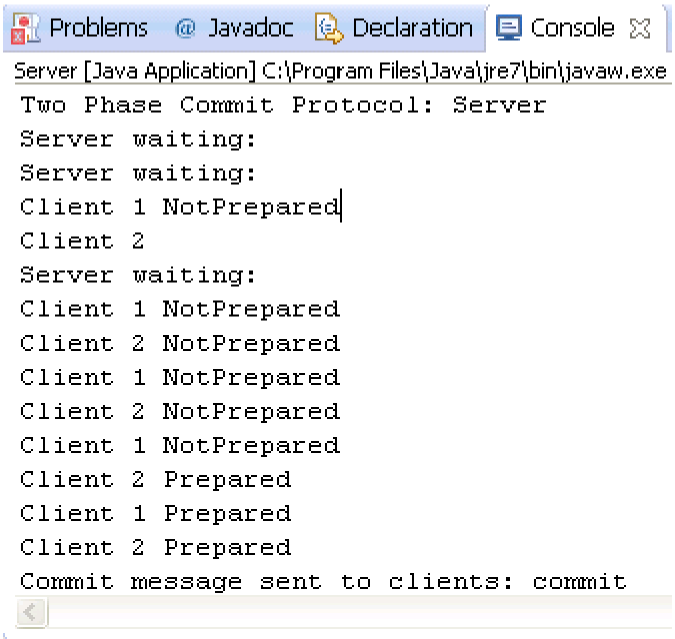
Client 1 says Prepared and since both clients said Prepared, the transaction Commits



Transactions Committed



Transactions Committed



Server [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe

```
Two Phase Commit Protocol: Server
Server waiting:
Server waiting:
Client 1 NotPrepared
Client 2
Server waiting:
Client 1 NotPrepared
Client 2 NotPrepared
Client 1 NotPrepared
Client 2 NotPrepared
Client 1 NotPrepared
Client 2 Prepared
Client 1 Prepared
Client 2 Prepared
Commit message sent to clients: commit
```

Lab Experiment

Hosting a Static Website

Accessing the AWS Management Console

1. At the top of these instructions, choose Start Lab to launch your lab.

   A **Start Lab** panel opens, and it displays the lab status.

2. Wait until the **Start Lab** panel displays the message *Lab status: ready*, then close the panel by choosing the **X**.
3. At the top of these instructions, choose AWS.

   This action opens the AWS Management Console in a new browser tab. The system automatically logs you in.

4. Arrange the **AWS Management Console** tab so that it displays alongside these instructions. Ideally, you will have both browser tabs open at the same time so that you can follow the lab steps more easily.

   Do not change the Region unless specifically instructed to do so.

Task 1: Creating a bucket in Amazon S3

In this task, you will create an S3 bucket and configure it for static website hosting.

In the **AWS Management Console**, on the **Services** menu, choose **S3**.

5. Choose **Create bucket**

   An S3 bucket name is globally unique, and the namespace is shared by all AWS accounts. After you create a bucket, the name of that bucket cannot be used by another AWS account in any AWS Region unless you delete the bucket.

   Thus, for this lab, you will use a bucket name that includes a random number, such as: *website-123*

6. For **Bucket name**, enter: `website-<123>` (replace *<123>* with a random number)

   Public access to buckets is blocked by default. Because the files in your static website will need to be accessible through the internet, you must permit public access.

   - Verify the **AWS Region** is set to **us-east-1** (if it is not, choose the us-east-1 Region)
7. In the **Object Ownership** section, select **ACLs enabled**, then verify **Bucket owner preferred** is selected.

8. Clear **Block all public access**, then select the box that states **I acknowledge that the current settings may result in this bucket and the objects within becoming public**.
9. Choose **Create bucket**.

   You can use tags to add additional information to a bucket, such as a project code, cost centre, or owner.

10. Choose the name of your new bucket.
11. Choose the **Properties** tab.
12. Scroll to the **Tags** panel.
13. Choose **Edit** then **Add tag** and enter:

- **Key:** Department
- **Value:** Marketing

14. Choose **Save changes** to save the tag.

    Next, you will configure the bucket for static website hosting.

15. Stay in the **Properties** console.
16. Scroll to the **Static website hosting** panel.
17. Choose **Edit**
18. Configure the following settings:
    - **Static web hosting:** Enable
    - **Hosting type:** Host a static website
    - **Index document:** index.html
      - **Note**: You must enter this value, even though it is already displayed.
    - **Error document:** error.html
19. Choose **Save changes**
20. In the **Static website hosting** panel, choose the link under **Bucket website endpoint**.

    You will receive a *403 Forbidden* message because the bucket permissions have not been configured yet. Keep this tab open in your web browser so that you can return to it later.

    Your bucket has now been configured to host a static website.

Task 2: Uploading content to your bucket

In this task, you will upload the files that will serve as your static website to the bucket.

21. Right-click each of these links and download the files to your computer:

    Ensure that each file keeps the same file name, including the extension.

    - index.html

- script.js
- style.css

22. Return to the Amazon S3 console and in the website-<123> bucket you created earlier, choose the **Objects** tab.
23. Choose **Upload.**
24. Choose **Add files**
25. Locate and select the three files that you downloaded.
26. If prompted, choose I acknowledge that existing objects with the same name will be overwritten.
27. Choose **Upload.**

    Your files are uploaded to the bucket.

    - Choose **Close**

Task 3: Enabling access to the objects

Objects that are stored in Amazon S3 are private by default. This ensures that your organization's data remains secure.

In this task, you will make the uploaded objects publicly accessible.

First, confirm that the objects are currently private.

28. Return to the browser tab that showed the *403 Forbidden* message.
29. Refresh the webpage

    You should still see a *403 Forbidden* message.

    Analysis: This response is expected! This message indicates that your static website is being hosted by Amazon S3, but that the content is private.

    You can make Amazon S3 objects public through two different ways:

    - To make either a whole bucket public, or a specific directory in a bucket public, use a *bucket policy*.
    - To make individual objects in a bucket public, use an *access control list (ACL)*.
30. Return to the web browser tab with the Amazon S3 console (but do not close the website tab).
31. Select all three objects.
32. In the **Actions** menu, choose **Make public via ACL**.

    A list of the three objects is displayed.

33. Choose **Make public**

Your static website is now publicly accessible.

34. Return to the web browser tab that has the *403 Forbidden* message.
35. Refresh the webpage.

You should now see the static website that is being hosted by Amazon S3.

Task 4: Updating the website

You can change the website by editing the HTML file and uploading it again to the S3 bucket.

36. On your computer, load the **index.html** file into a text editor (for example, Notepad or TextEdit).
37. Find the text **Served from Amazon S3** and replace it with Created by <YOUR-NAME>, substituting your name for <*YOUR-NAME*> (for example, *Created by Jane*).
38. Save the file.
39. Return to the Amazon S3 console and upload the **index.html** file that you just edited.
40. Select **index.html** and use the **Actions** menu to choose the **Make public via ACL** option again.
41. Return to the web browser tab with the static website and refresh the page.

    Your name should now be on the page.

Your static website is now accessible on the internet. Because it is hosted on Amazon S3, the website has high availability and can serve high volumes of traffic without using any servers.

You can also use your own domain name to direct users to a static website that is hosted on Amazon S3. To accomplish this, you could use the Amazon Route 53 Domain Name System (DNS) service in combination with Amazon S3.

Submitting your work

42. At the top of these instructions, choose **Submit** to record your progress and when prompted, choose **Yes**.
43. If the results don't display after a couple of minutes, return to the top of these instructions, and choose Grades
44. To find detailed feedback on your work, choose Details followed by **View Submission Report**.

Lab complete

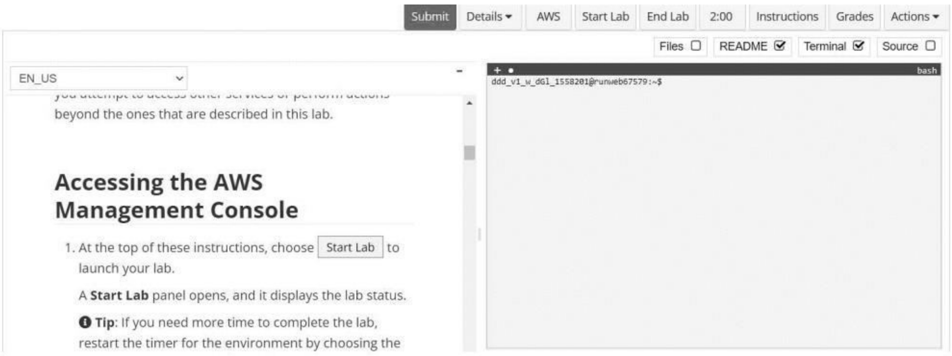
45. Choose End Lab at the top of this page, and then select **Yes** to confirm that you want to end the lab.

    A panel indicates that *DELETE has been initiated... You may close this message box now.*

46. Select the **X** in the top right corner to close the panel.

OUTPUT SCREENSHOTS:

Instructions Panel



beyond the ones that are described in this lab.

Accessing the AWS Management Console

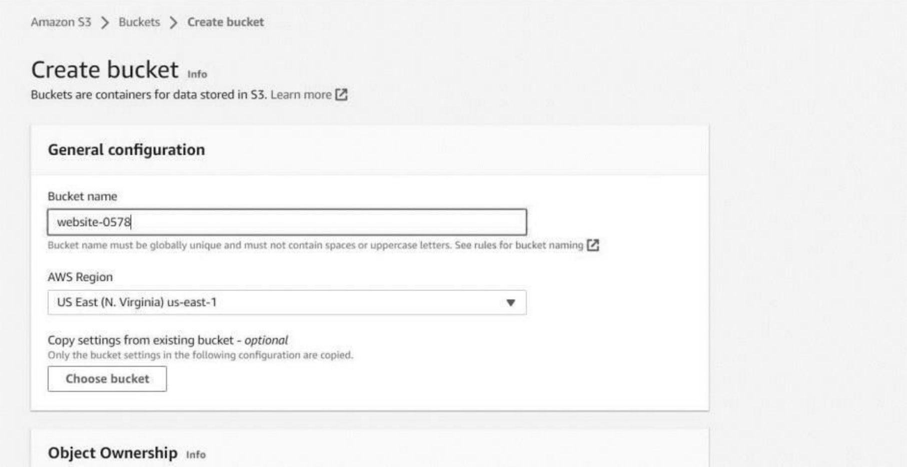
1. At the top of these instructions, choose Start Lab to launch your lab.

A **Start Lab** panel opens, and it displays the lab status.

Tip: If you need more time to complete the lab, restart the timer for the environment by choosing the

```
ddd_v1_w_dGl_1558201@runweb67579:~$
```

Bucket Creation



Create bucket

Buckets are containers for data stored in S3.

General configuration

Bucket name: website-0578

Bucket name must be globally unique and must not contain spaces or uppercase letters. See rules for bucket naming

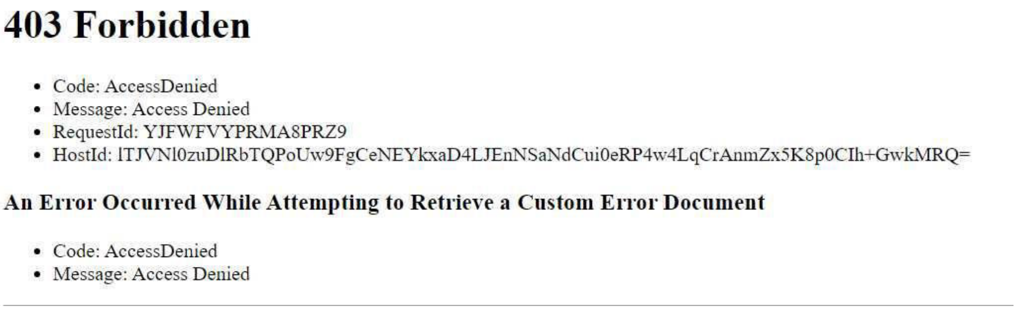
AWS Region: US East (N. Virginia) us-east-1

Copy settings from existing bucket - *optional*

Only the bucket settings in the following configuration are copied.

Object Ownership

Forbidden Error



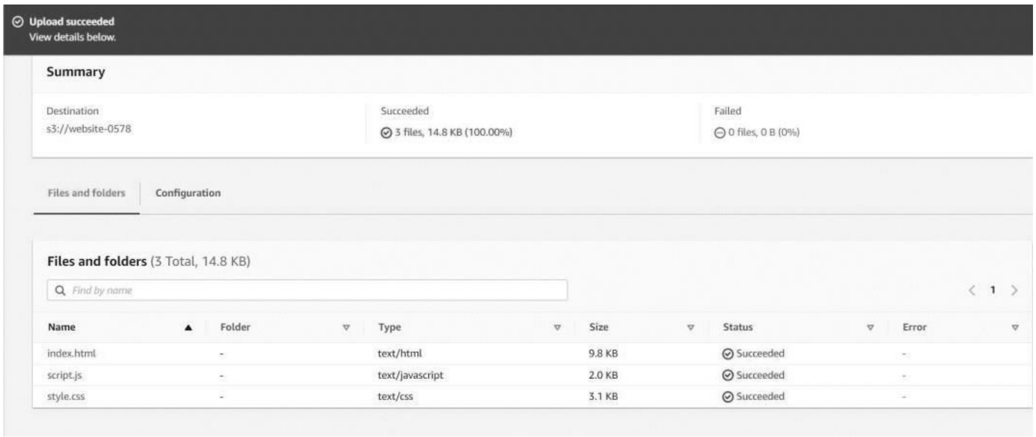
403 Forbidden

- Code: AccessDenied
- Message: Access Denied
- RequestId: YJFWFVYPRMA8PRZ9
- HostId: lTJVNl0zuDlRbTQPoUw9FgCeNEYkxaD4LJEnNSaNdCui0eRP4w4LqCrAnmZx5K8p0CIh+GwkMRQ=

An Error Occurred While Attempting to Retrieve a Custom Error Document

- Code: AccessDenied
- Message: Access Denied

Uploading Files



Upload succeeded
View details below.

Summary

Destination
s3://website-0578

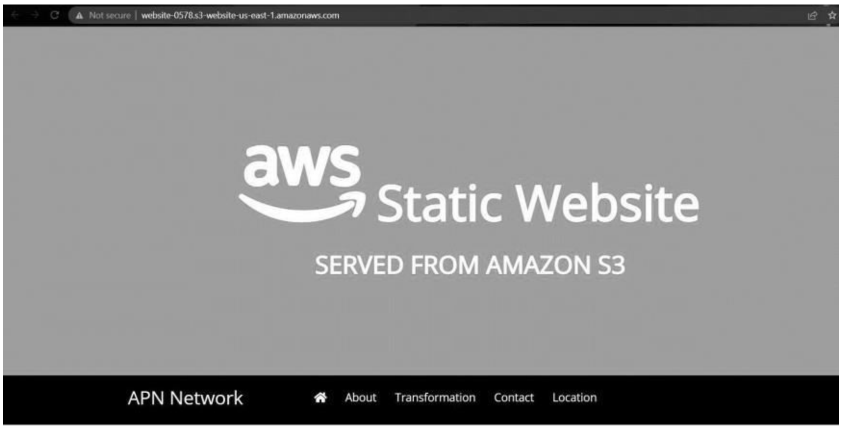
Succeeded
3 files, 14.8 KB (100.00%)

Failed
0 files, 0 B (0%)

Files and folders (3 Total, 14.8 KB)

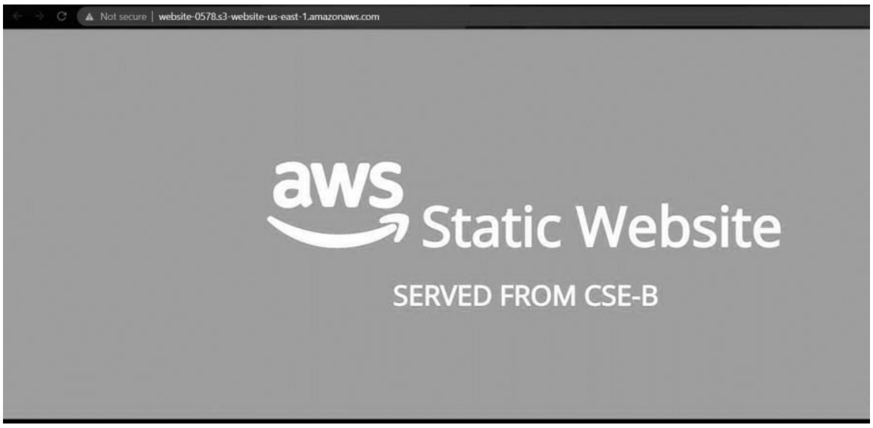
Name	Folder	Type	Size	Status	Error
index.html	-	text/html	9.8 KB	Succeeded	-
script.js	-	text/javascript	2.0 KB	Succeeded	-
style.css	-	text/css	3.1 KB	Succeeded	-

Static Website-1



Not secure | website-0578.s3-website-us-east-1.amazonaws.com

Static Website-2 (Updated)



Not secure | website-0578.s3-website-us-east-1.amazonaws.com

Lab Experiment

Introducing Amazon Elastic File System (Amazon EFS)

Accessing the AWS Management Console

1. At the top of these instructions, choose Start Lab to launch your lab.

   A **Start Lab** panel opens, and it displays the lab status.

   Tip: If you need more time to complete the lab, restart the timer for the environment by choosing the Start Lab button again.

2. Wait until the **Start Lab** panel displays the message *Lab status: ready*, then close the panel by choosing the **X**.
3. At the top of these instructions, choose AWS.
4. Arrange the **AWS Management Console** tab so that it displays alongside these instructions. Ideally, you will have both browser tabs open at the same time so that you can follow the lab steps more easily.

Task 1: Creating a security group to access your EFS file system

5. In the **AWS Management Console**, on the Services menu, choose **EC2**.
6. In the navigation pane on the left, choose **Security Groups**.
7. Copy the **Security group ID** of the *EFSClient* security group to your text editor.

   The Group ID should look similar to *sg-03727965651b6659b*.

8. Choose Create security group then configure:
   - **Security group name:** EFS Mount Target
   - **Description:** Inbound NFS access from EFS clients
   - **VPC:** *Lab VPC*
9. Under the **Inbound rules** section, choose **Add rule** then configure:
   - **Type:** *NFS*
   - **Source:**
     - *Custom*
     - In the *Custom* box, paste the security group's **Security group ID** that you copied to your text editor
   - Choose Create security group.

Task 2: Creating an EFS file system

10. On the Services menu, choose **EFS**.
11. Choose Create file system
12. In the **Create file system** window, choose **Customize**

13. On **Step 1**:
    - Uncheck Enable automatic backups.
    - **Lifecycle management:** Select *None*
    - In the **Tags** section, configure:
        - **Key:** Name
        - **Value:** My First EFS File System
14. Choose Next
15. For **VPC**, select *Lab VPC*.
16. Detach the default security group from each *Availability Zone* mount target by choosing the check box on each default security group.
17. Attach the **EFS Mount Target** security group to each *Availability Zone* mount target by:

- Selecting each **Security groups** check box.
- Choosing **EFS Mount Target**

A mount target is created for each subnet

18. Choose Next
19. On **Step 3**, choose Next
20. On **Step 4:**

- Review your configuration.
- Choose Create

Proceed to the next step after the **Mount target state** for each mount target changes to *Available*. Choose the screen refresh button after 2–3 minutes to check its progress.

Task 3: Connecting to your EC2 instance via SSH

In this task, you will connect to your EC2 instance by using Secure Shell (SSH).

21. Above these instructions that you are currently reading, choose the Details dropdown menu, and then select Show

A **Credentials** window opens.

22. Choose the **Download PPK** button and save the **labsuser.ppk** file.

Note: Typically, your browser saves the file to the **Downloads** directory.

23. Note the **EC2PublicIP** address if it is displayed.
24. Exit the **Details** panel by choosing the **X**.
25. To use SSH to access the EC2 instance, you must use ****PuTTY****. If you do not have PuTTY installed on your computer, download PuTTY.
26. Open **putty.exe**.
27. To keep the PuTTY session open for a longer period of time, configure the PuTTY timeout:

- Choose **Connection**
- **Seconds between keepalives**: 30

28. Configure your PuTTY session by using the following settings.

- Choose **Session**
- **Host Name (or IP address):** Paste the **EC2PublicIP** for the instance you noted earlier
    - Alternatively, return to the Amazon EC2 console and choose **Instances**
    - Select the instance you want to connect to
    - In the *Description* tab, copy the **IPv4 Public IP** value
- Back in PuTTY, in the **Connection** list, expand **SSH**
- Choose **Auth** (but don't expand it)
- Choose **Browse**
- Browse to the *labsuser.ppk* file that you downloaded, select it, and choose **Open**
- Choose **Open** again

29. To trust and connect to the host, choose **Yes**.
30. When you are prompted with **login as**, enter: `ec2-user.`

    This action connects you to the EC2 instance.

Task 4: Creating a new directory and mounting the EFS file system

31. In your SSH session, make a new directory by entering `sudo mkdir efs`
32. Back in the **AWS Management Console**, on the Services menu, choose **EFS**.
33. Choose **My First EFS File System**.
34. In the **Amazon EFS Console**, on the top right corner of the page, choose Attach to open the Amazon EC2 mount instructions.
35. Copy the entire command in the **Using the NFS client** section.

    The mount command should look similar to this example:

    ```
    sudo mount -t nfs4 -o nfsvers=4.1,rsize=1048576,wsize=1048576,hard,timeo=600,retrans=2,noresvport fs-bce57914.efs.us-west-2.amazonaws.com:/ efs
    ```

    The provided `sudo mount...` command uses the default Linux mount options.

36. In your Linux SSH session, mount your Amazon EFS file system by:
    - Pasting the command
    - Pressing ENTER

37. Get a full summary of the available and used disk space usage by entering:

    ```
    sudo df -hT
    ```

Task 5: Examining the performance behavior of your new EFS file system

38. Examine the write performance characteristics of your file system by entering:

```
sudo fio --name=fio-efs --filesize=10G --filename=./efs/fio-efs-test.img --bs=1M --nrfiles=1 --direct=1 --sync=0 --rw=write --iodepth=200 --ioengine=libaio
```

Monitoring performance by using Amazon CloudWatch

39. In the **AWS Management Console**, on the Services menu, choose **CloudWatch**.
40. In the navigation pane on the left, choose **Metrics**.
41. In the **All-metrics** tab, choose **EFS**.
42. Choose **File System Metrics**.
43. Select the row that has the **PermittedThroughput** Metric Name.

    You might need to wait 2–3 minutes and refresh the screen several times before all available metrics, including **PermittedThroughput**, calculate and populate.

44. On the graph, choose and drag around the data line. If you do not see the line graph, adjust the time range of the graph to display the period during which you ran the `fio` command.

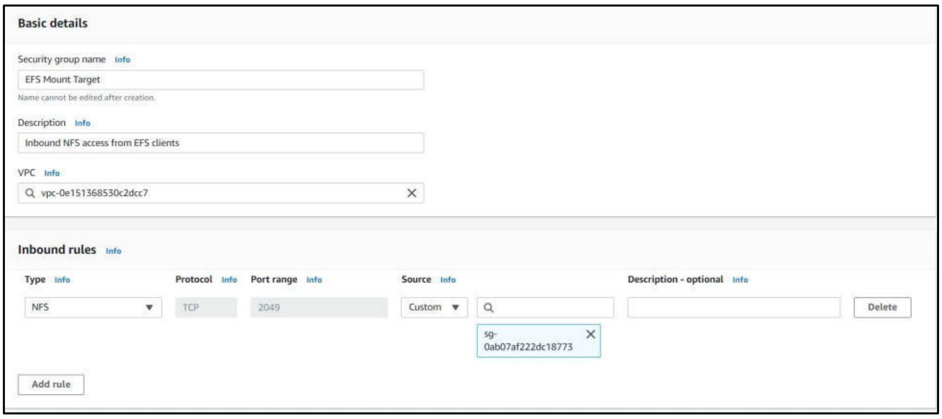
45. Pause your pointer on the data line in the graph. The value should be *105M*.
46. In the **All-metrics** tab, *uncheck* the box for **PermittedThroughput**.
47. Select the check box for **DataWriteIOBytes**.

    If you do not see *DataWriteIOBytes* in the list of metrics, use the **File System Metrics** search to find it.

48. Choose the **Graphed metrics** tab.
49. On the **Statistics** column, select **Sum**.
50. On the **Period** column, select **1 Minute**.
51. Pause your pointer on the peak of the line graph. Take this number (in bytes) and divide it by the duration in seconds (60 seconds). The result gives you the write throughput (B/s) of your file system during your test.

OUTPUT SCREENSHOTS:

Task-1: Creating a security group to access your EFS file system



Basic details

Security group name Info

EFS Mount Target

Name cannot be edited after creation.

Description Info

Inbound NFS access from EFS clients

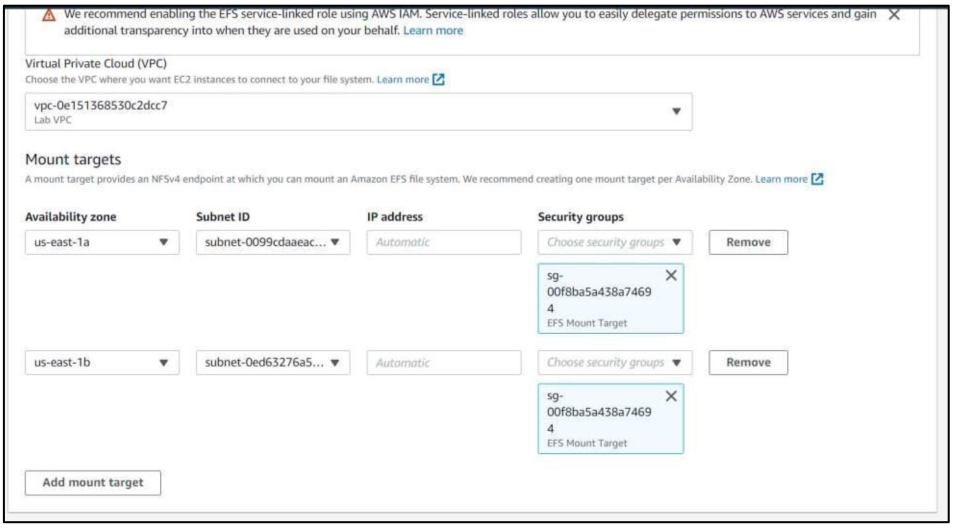
VPC Info

vpc-0e151368530c2dcc7

Inbound rules Info

Type	Protocol	Port range	Source		Description - optional
NFS	TCP	2049	Custom	sg-0ab07af222dc18773	

Task-2: Creating an EFS file system



We recommend enabling the EFS service-linked role using AWS IAM. Service-linked roles allow you to easily delegate permissions to AWS services and gain additional transparency into when they are used on your behalf. Learn more

Virtual Private Cloud (VPC)

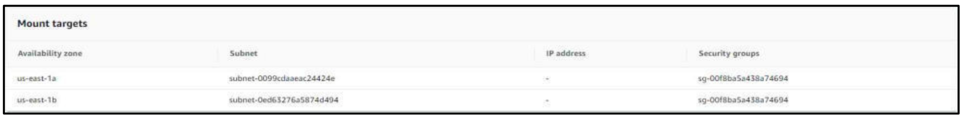
Choose the VPC where you want EC2 instances to connect to your file system. Learn more

vpc-0e151368530c2dcc7
Lab VPC

Mount targets

A mount target provides an NFSv4 endpoint at which you can mount an Amazon EFS file system. We recommend creating one mount target per Availability Zone. Learn more

Availability zone	Subnet ID	IP address	Security groups
us-east-1a	subnet-0099cdaaeac...	Automatic	sg-00f8ba5a438a74694 EFS Mount Target
us-east-1b	subnet-0ed63276a5...	Automatic	sg-00f8ba5a438a74694 EFS Mount Target

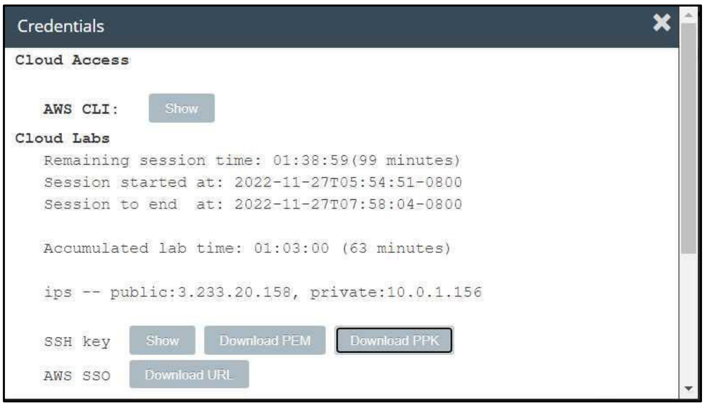


Mount targets

Availability zone	Subnet	IP address	Security groups
us-east-1a	subnet-0099cdaaeac24424e	-	sg-00f8ba5a438a74694
us-east-1b	subnet-0ed63276a5874d494	-	sg-00f8ba5a438a74694

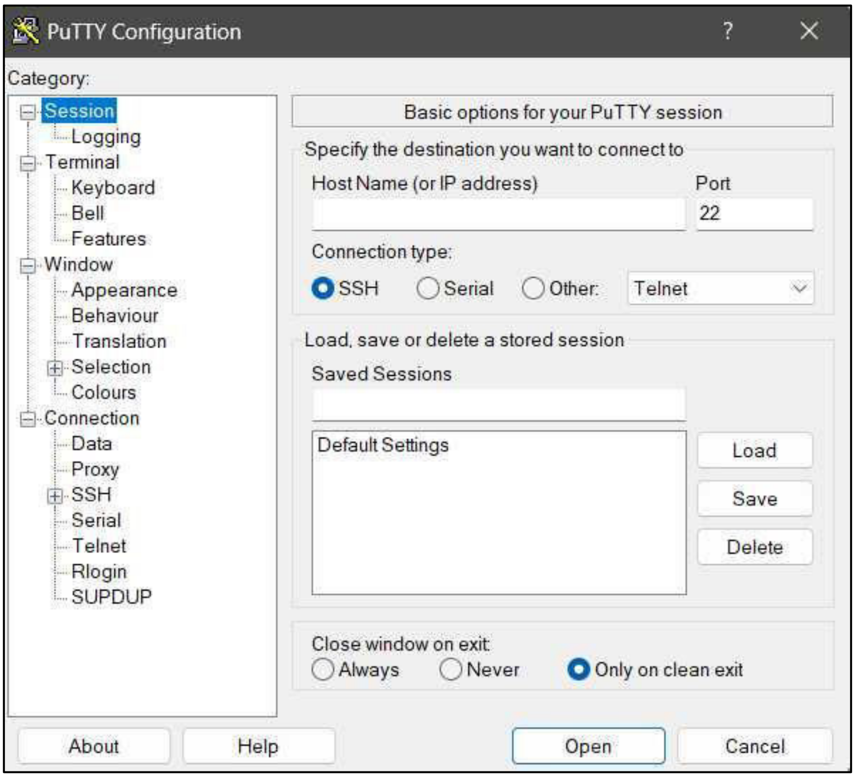
Task-3:

Credentials Tab



```
Credentials
Cloud Access

  AWS CLI:  Show
Cloud Labs
  Remaining session time: 01:38:59(99 minutes)
  Session started at: 2022-11-27T05:54:51-0800
  Session to end  at: 2022-11-27T07:58:04-0800

  Accumulated lab time: 01:03:00 (63 minutes)

  ips -- public:3.233.20.158, private:10.0.1.156

  SSH key  Show  Download PEM  Download PPK
  AWS SSO  Download URL
```

Putty Config



PuTTY Configuration

Basic options for your PuTTY session

Specify the destination you want to connect to

Host Name (or IP address):

Port: 22

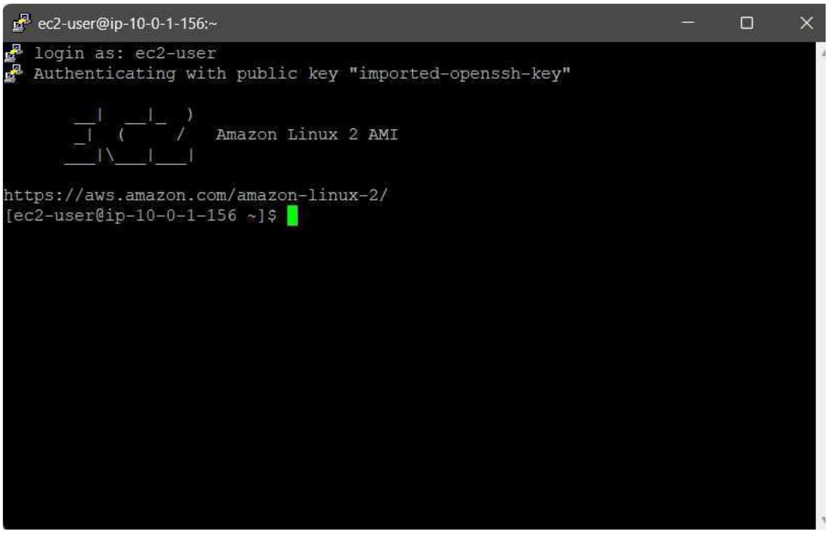
Connection type: ● SSH ○ Serial ○ Other: Telnet

Load, save or delete a stored session

Saved Sessions:

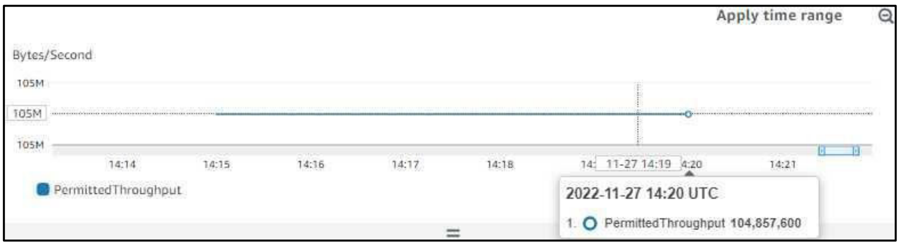
Default Settings

Close window on exit: ○ Always ○ Never ● Only on clean exit



```
login as: ec2-user
Authenticating with public key "imported-openssh-key"

       __|  __|_  )
       _|  (     /   Amazon Linux 2 AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-2/
[ec2-user@ip-10-0-1-156 ~]$ 
```

Task-5: Examining the performance behavior of your new EFS file system



2022-11-27 14:20 UTC

1. PermittedThroughput 104,857,600

LAB PROGRAM

Experiment: Deploying a Node.js Web Application on AWS

HARDWARE REQUIREMENTS: Core I5 Processor, 4 GB RAM, 40GB HDD

SOFTWARE REQUIREMENTS: Amazon AWS, EC2, VS Code/Eclipse, Node, NPM, GIT, Putty

Description:

Node.js is a JavaScript runtime environment that allows one to run JS on the server. It is built on the open-source V8 JavaScript engine used in Chrome and written in C++ which executes JS in a standalone environment.

In this experiment, we clone a Nodejs application from GITHUB and deploy this application on to Amazon EC2 instance, make it available over Amazon AWS URI.

Steps to configure EC2 Instance :

1. Create an EC2 instance and Launch it:

Choose amazon Ec2 instance machine image as Ubuntu 18.04 64 bit with type of micro.

(Login to AwsAcademy,

LMS-Dashboard - AWS Academy Learner Lab – Educator

Click on Modules

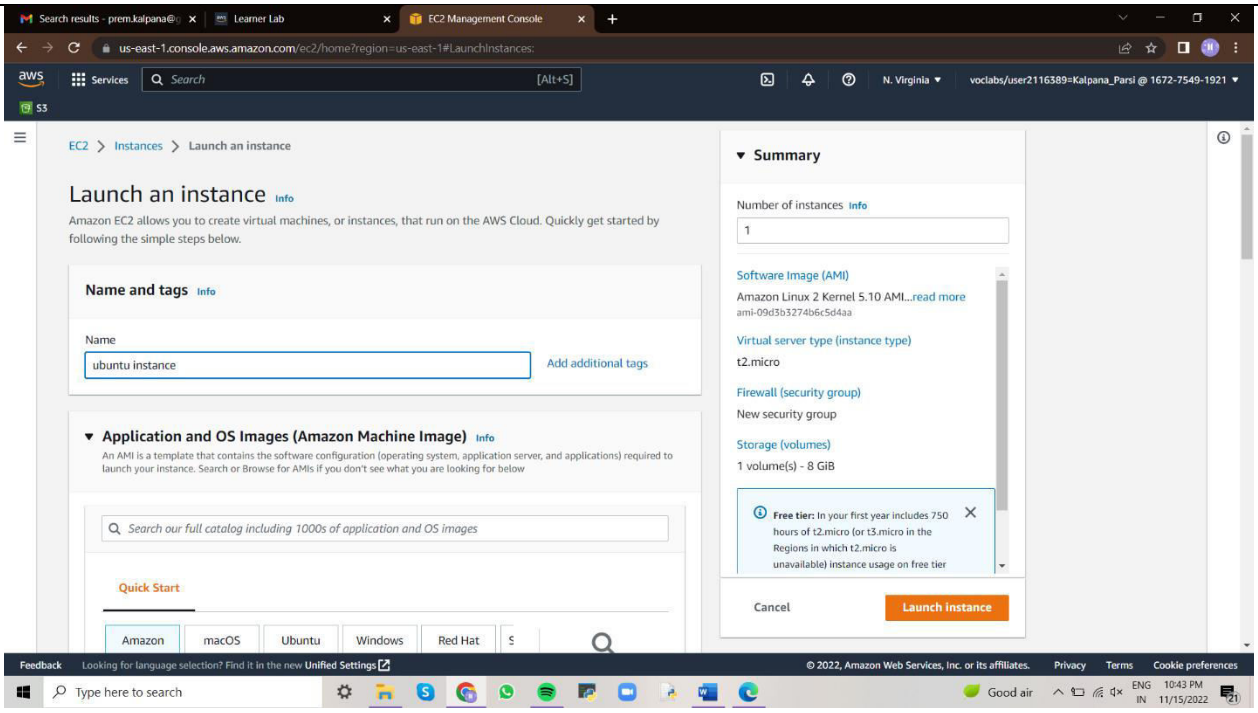
Click on Learner Lab

Click on Start Lab

Click on AWS

Services – EC2

EC2 – Instances – Launch an instance


us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#LaunchInstances:

EC2 > Instances > Launch an instance

Launch an instance Info

Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.

Name and tags Info

Name: ubuntu instance

Add additional tags

Application and OS Images (Amazon Machine Image) Info

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below

Search our full catalog including 1000s of application and OS images

Quick Start

Amazon | macOS | Ubuntu | Windows | Red Hat

Summary

Number of instances Info: 1

Software Image (AMI): Amazon Linux 2 Kernel 5.10 AMI...read more ami-09d3b3274b6c5d4aa

Virtual server type (instance type): t2.micro

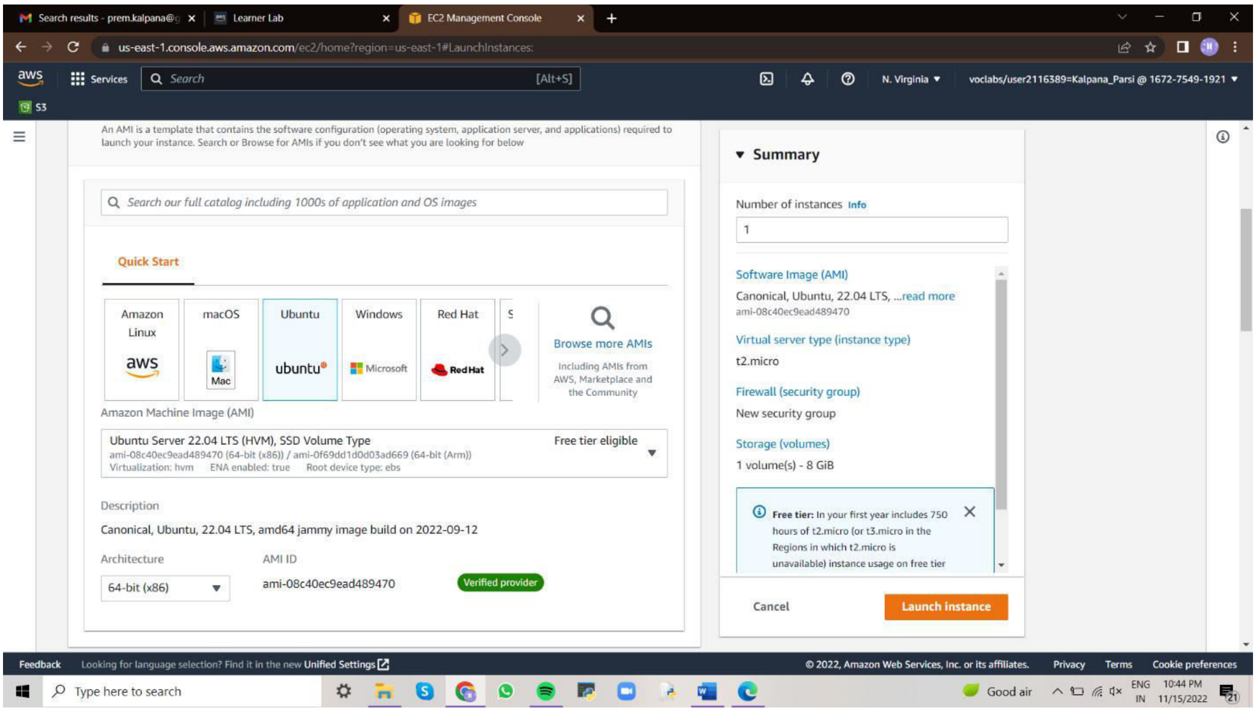
Firewall (security group): New security group

Storage (volumes): 1 volume(s) - 8 GiB

Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier

Cancel | Launch instance

Select Amazon Ubuntu


us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#LaunchInstances:

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below

Search our full catalog including 1000s of application and OS images

Quick Start

Amazon Linux | macOS | Ubuntu | Windows | Red Hat

Browse more AMIs
Including AMIs from AWS, Marketplace and the Community

Amazon Machine Image (AMI)

Ubuntu Server 22.04 LTS (HVM), SSD Volume Type — Free tier eligible
ami-08c40ec9ead489470 (64-bit (x86)) / ami-0f69dd1d0d03ad669 (64-bit (Arm))
Virtualization: hvm ENA enabled: true Root device type: ebs

Description: Canonical, Ubuntu, 22.04 LTS, amd64 jammy image build on 2022-09-12

Architecture: 64-bit (x86)

AMI ID: ami-08c40ec9ead489470 Verified provider

Summary

Number of instances Info: 1

Software Image (AMI): Canonical, Ubuntu, 22.04 LTS, ...read more ami-08c40ec9ead489470

Virtual server type (instance type): t2.micro

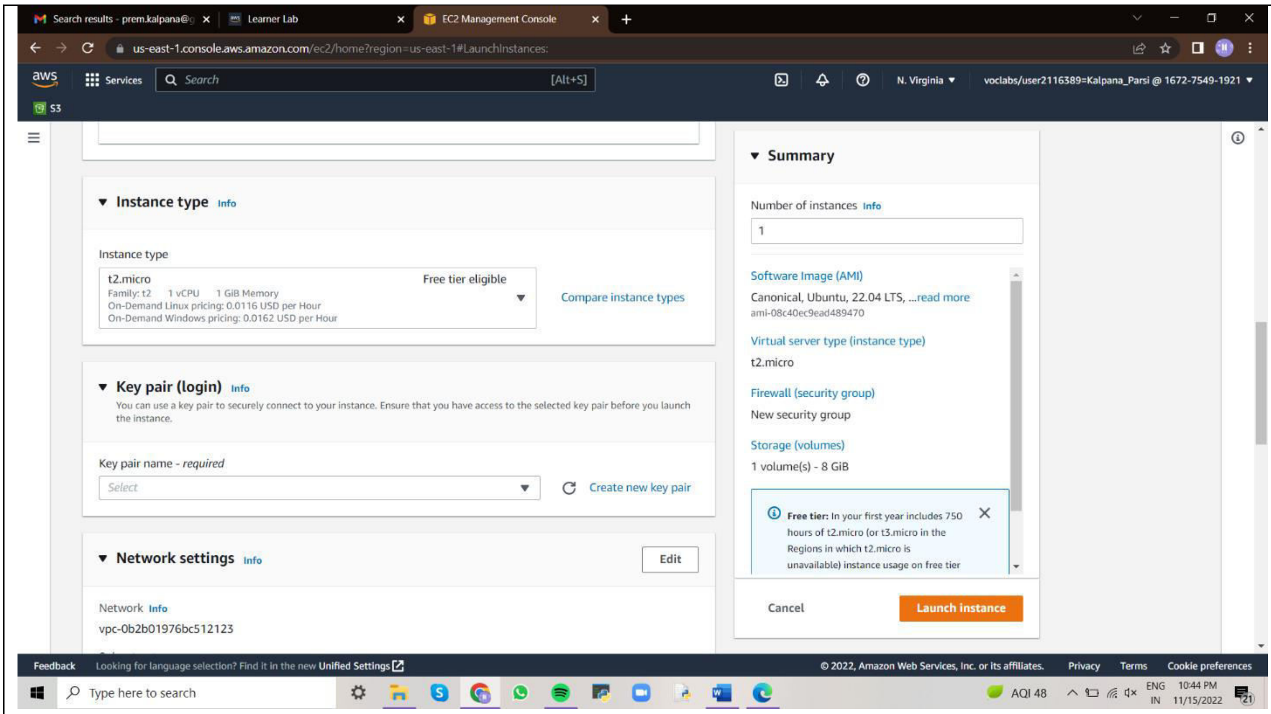
Firewall (security group): New security group

Storage (volumes): 1 volume(s) - 8 GiB

Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier

Cancel | Launch instance

Instance type - t2.micro)



Instance type Info

Instance type

t2.micro Free tier eligible
Family: t2 1 vCPU 1 GiB Memory
On-Demand Linux pricing: 0.0116 USD per Hour
On-Demand Windows pricing: 0.0162 USD per Hour

Compare instance types

Key pair (login) Info

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

Key pair name - *required*

Select

Create new key pair

Network settings Info

Edit

Network Info

vpc-0b2b01976bc512123

Summary

Number of instances Info

1

Software Image (AMI)

Canonical, Ubuntu, 22.04 LTS, ...read more

ami-08c40ec9ead489470

Virtual server type (instance type)

t2.micro

Firewall (security group)

New security group

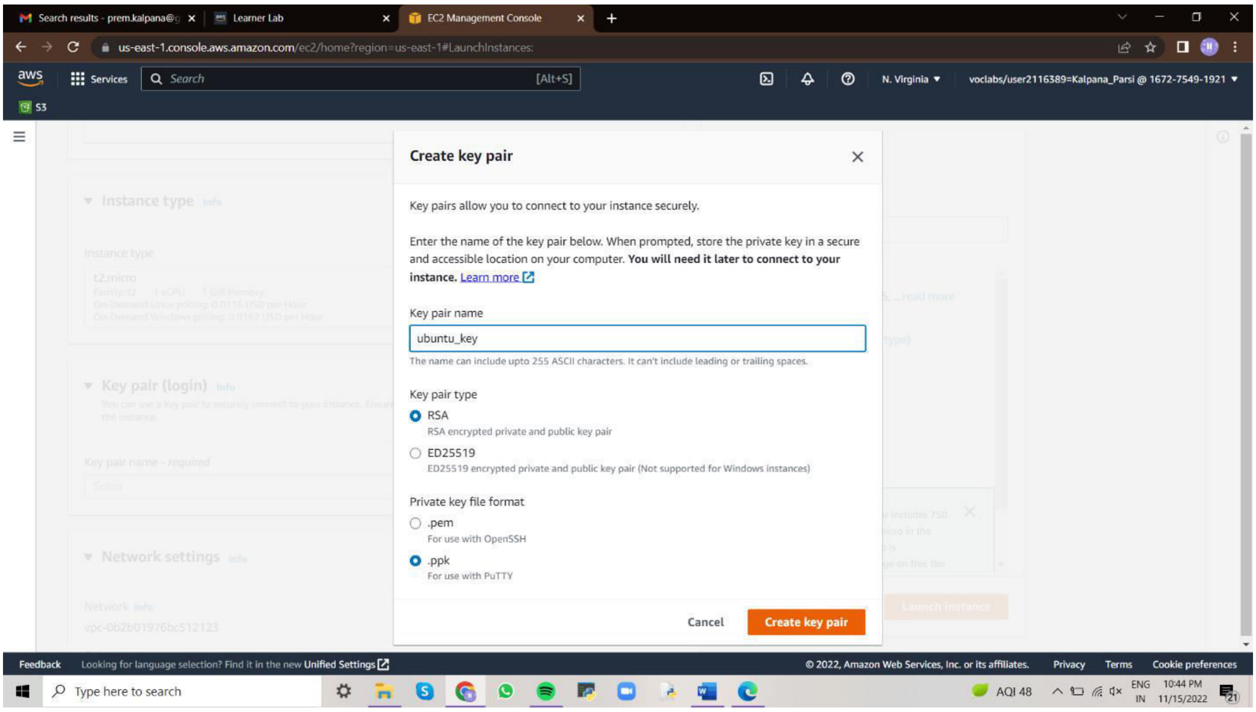
Storage (volumes)

1 volume(s) - 8 GiB

Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier

Cancel Launch instance

Create new key pair – Save the key pair as .ppk (to work with putty)



Create key pair

Key pairs allow you to connect to your instance securely.

Enter the name of the key pair below. When prompted, store the private key in a secure and accessible location on your computer. **You will need it later to connect to your instance.** Learn more

Key pair name

ubuntu_key

The name can include upto 255 ASCII characters. It can't include leading or trailing spaces.

Key pair type

● RSA
RSA encrypted private and public key pair

○ ED25519
ED25519 encrypted private and public key pair (Not supported for Windows instances)

Private key file format

○ .pem
For use with OpenSSH

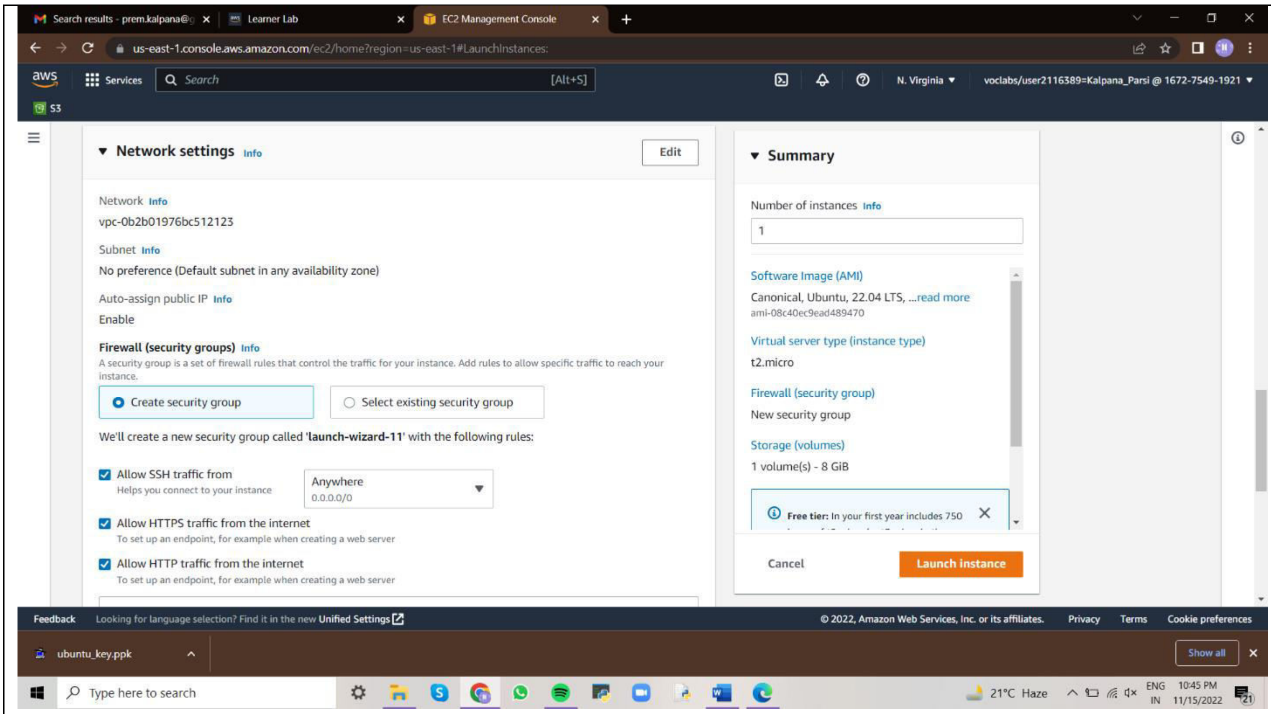
● .ppk
For use with PuTTY

Cancel Create key pair

Next Add storage

Next configure Security Group – Create security group.

In this step we need to allow http and https requests to access from any group.



Network settings Info

Network Info
vpc-0b2b01976bc512123

Subnet Info
No preference (Default subnet in any availability zone)

Auto-assign public IP Info
Enable

Firewall (security groups) Info
A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

● Create security group ○ Select existing security group

We'll create a new security group called 'launch-wizard-11' with the following rules:

☑ Allow SSH traffic from — Anywhere 0.0.0.0/0
Helps you connect to your instance

☑ Allow HTTPS traffic from the internet
To set up an endpoint, for example when creating a web server

☑ Allow HTTP traffic from the internet
To set up an endpoint, for example when creating a web server

Summary

Number of instances Info
1

Software Image (AMI)
Canonical, Ubuntu, 22.04 LTS, ...read more
ami-08c40ec9ead489470

Virtual server type (instance type)
t2.micro

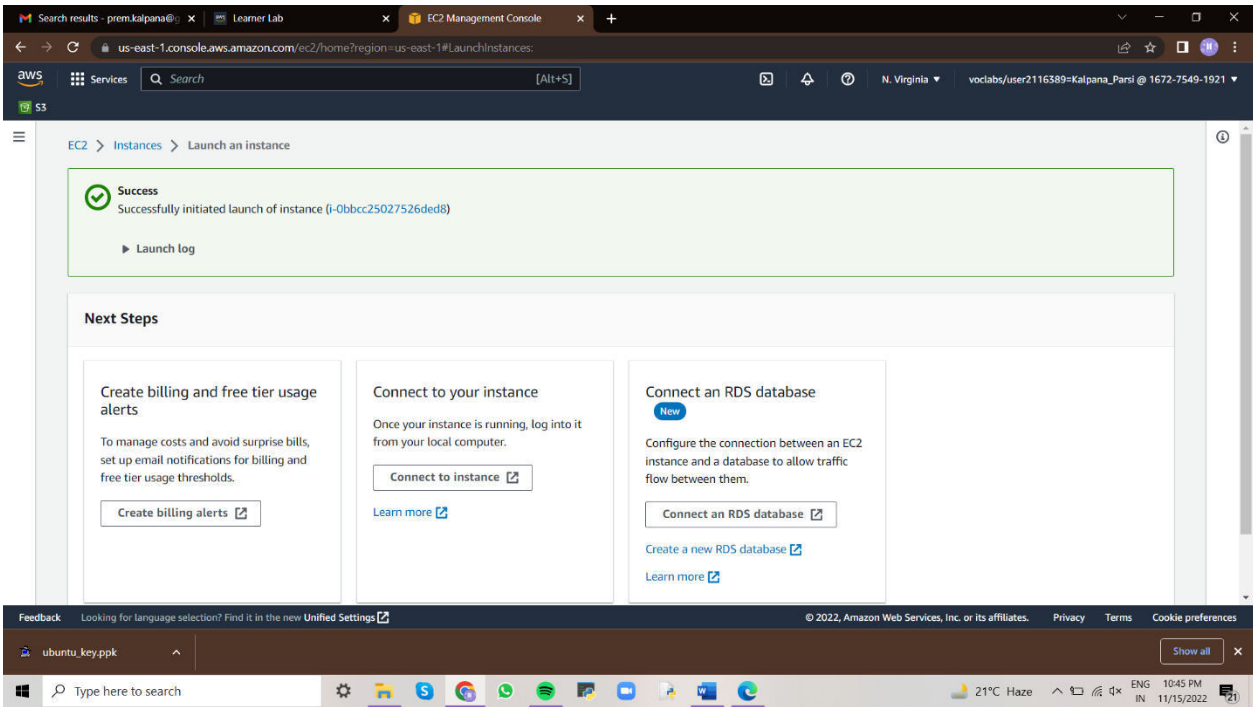
Firewall (security group)
New security group

Storage (volumes)
1 volume(s) - 8 GiB

Free tier: In your first year includes 750

Finally click on Launch instance.

We can see instance is launched successfully.



EC2 > Instances > Launch an instance

Success
Successfully initiated launch of instance (i-0bbcc25027526ded8)

▶ Launch log

Next Steps

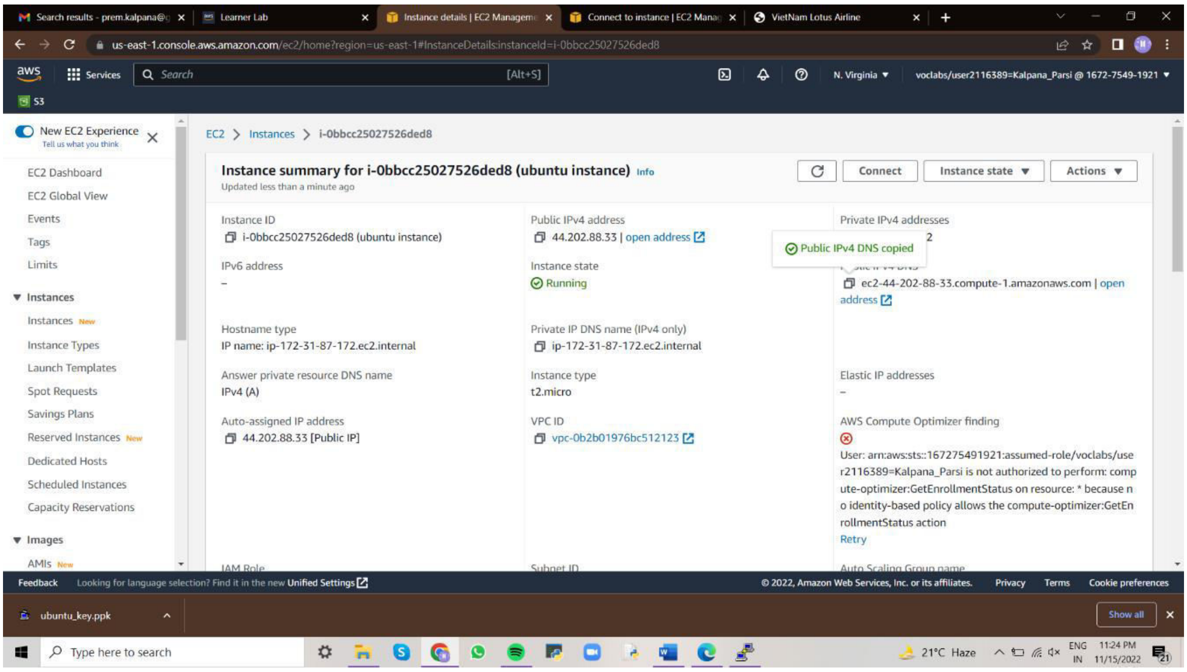
Create billing and free tier usage alerts
To manage costs and avoid surprise bills, set up email notifications for billing and free tier usage thresholds.

Connect to your instance
Once your instance is running, log into it from your local computer.

Connect an RDS database New
Configure the connection between an EC2 instance and a database to allow traffic flow between them.

When the instance state is running , it indicates that your instance was created successfully.

Copy the public DNS of your Instance. You can access different app running on your instance at a different port.

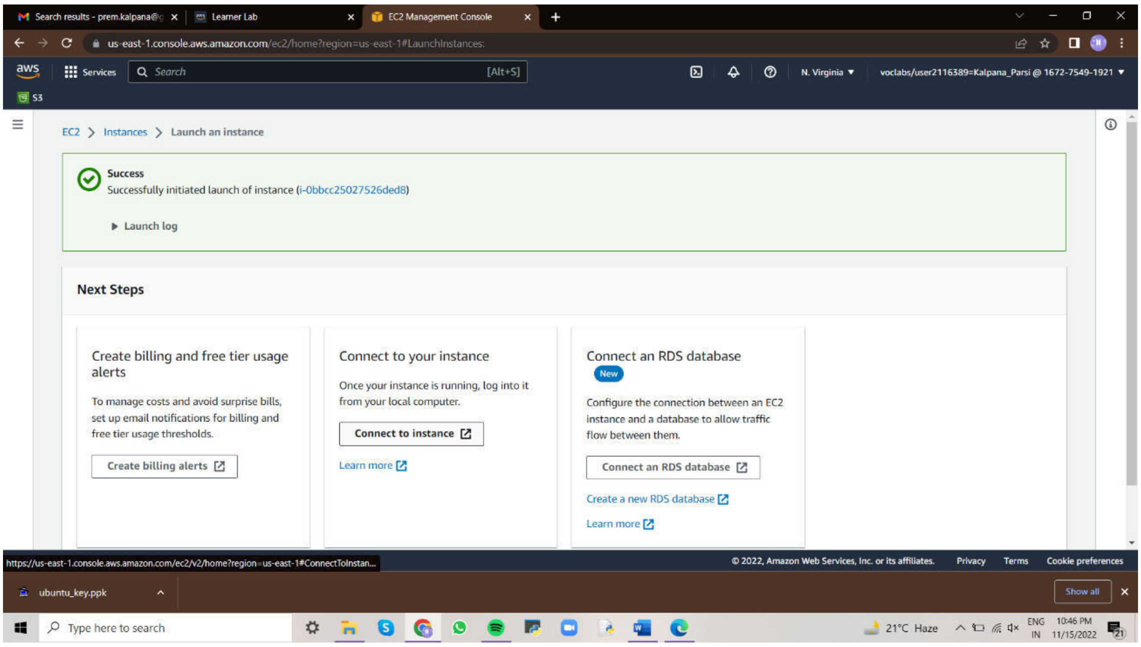


Instance summary for i-0bbcc25027526ded8 (ubuntu instance)
Updated less than a minute ago

Instance ID: i-0bbcc25027526ded8 (ubuntu instance)
Public IPv4 address: 44.202.88.33 | open address
Private IPv4 addresses: 2
IPv6 address: –
Instance state: Running
Public IPv4 DNS copied
ec2-44-202-88-33.compute-1.amazonaws.com | open address
Hostname type: IP name: ip-172-31-87-172.ec2.internal
Private IP DNS name (IPv4 only): ip-172-31-87-172.ec2.internal
Answer private resource DNS name: IPv4 (A)
Instance type: t2.micro
Elastic IP addresses: –
Auto-assigned IP address: 44.202.88.33 [Public IP]
VPC ID: vpc-0b2b01976bc512123
AWS Compute Optimizer finding: User: arn:aws:sts::167275491921:assumed-role/voclabs/user2116389=Kalpana_Parsi is not authorized to perform: compute-optimizer:GetEnrollmentStatus on resource: * because no identity-based policy allows the compute-optimizer:GetEnrollmentStatus action
Retry

2. Connect to your Instance:

Click on launch instance then it shows popup window giving details how to connect to your instance.



Success
Successfully initiated launch of instance (i-0bbcc25027526ded8)

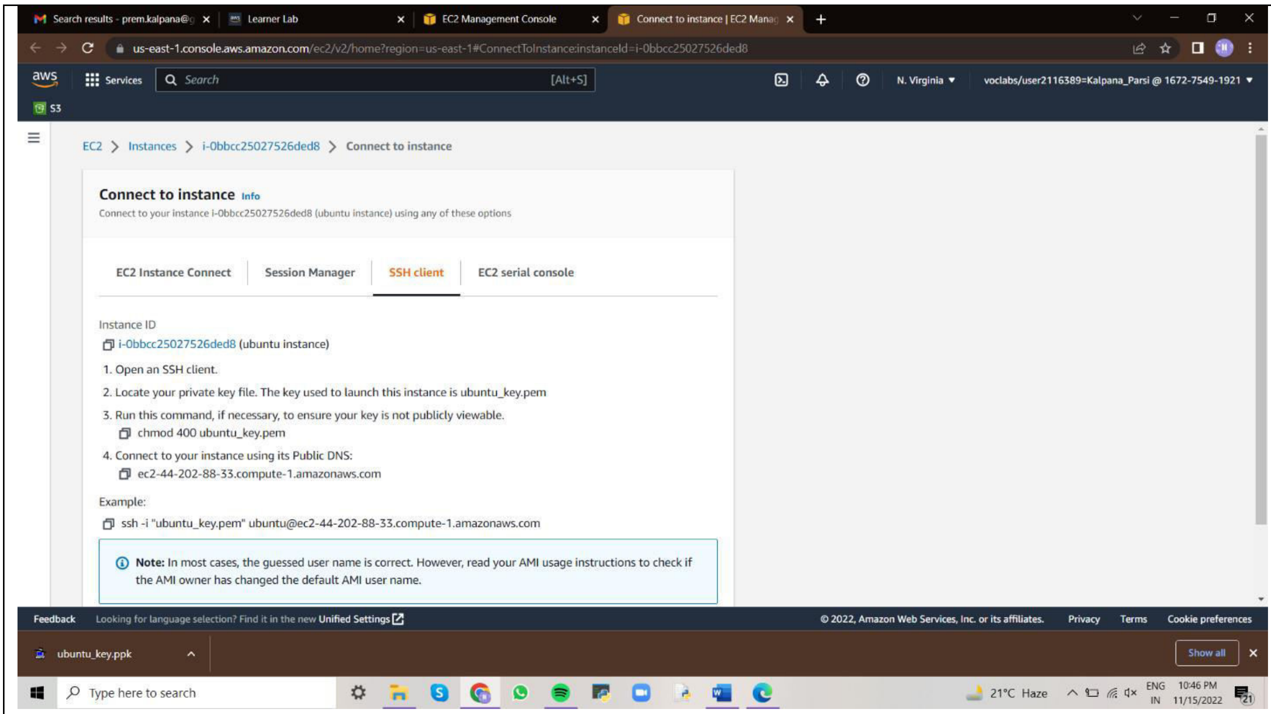
Next Steps

Create billing and free tier usage alerts
To manage costs and avoid surprise bills, set up email notifications for billing and free tier usage thresholds.

Connect to your instance
Once your instance is running, log into it from your local computer.

Connect an RDS database
Configure the connection between an EC2 instance and a database to allow traffic flow between them.

To open SSH client and If we are in windows platform we need to launch the instance with the help of putty soft.



us-east-1.console.aws.amazon.com/ec2/v2/home?region=us-east-1#ConnectToInstance:instanceId=i-0bbcc25027526ded8

EC2 > Instances > i-0bbcc25027526ded8 > Connect to instance

Connect to instance Info

Connect to your instance i-0bbcc25027526ded8 (ubuntu instance) using any of these options

EC2 Instance Connect | Session Manager | SSH client | EC2 serial console

Instance ID

i-0bbcc25027526ded8 (ubuntu instance)

1. Open an SSH client.
2. Locate your private key file. The key used to launch this instance is ubuntu_key.pem
3. Run this command, if necessary, to ensure your key is not publicly viewable.
   chmod 400 ubuntu_key.pem
4. Connect to your instance using its Public DNS:
   ec2-44-202-88-33.compute-1.amazonaws.com

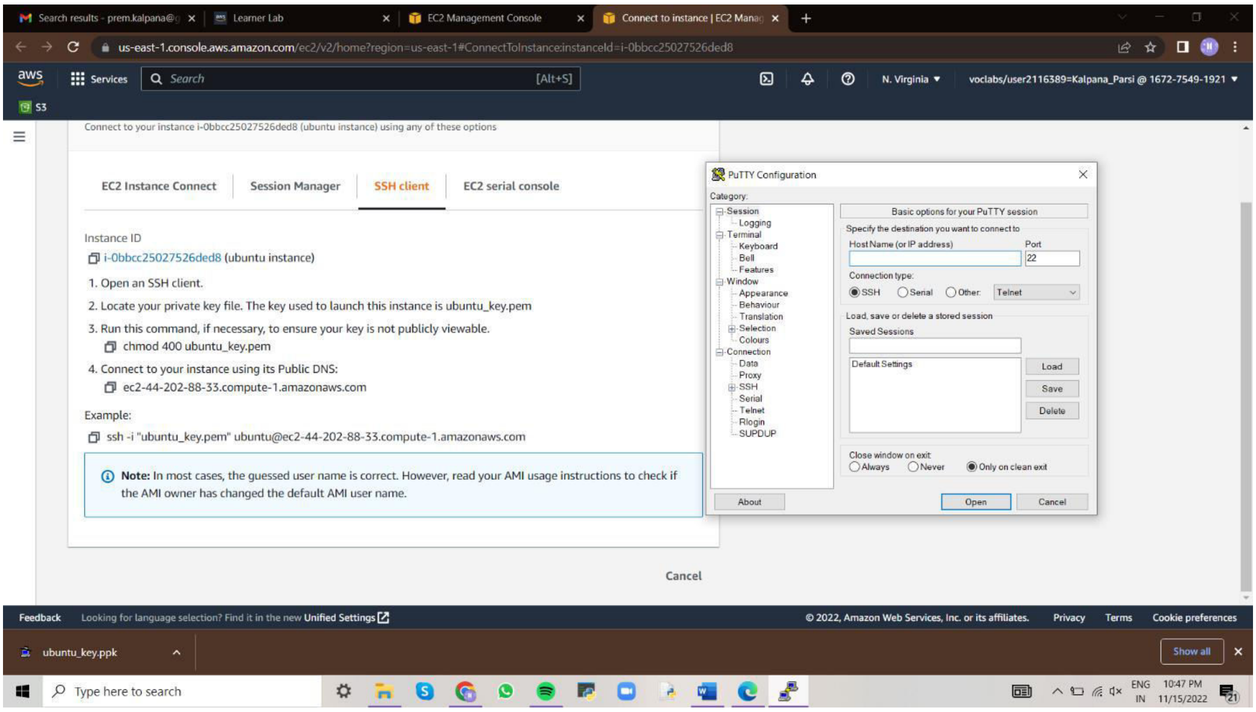
Example:

ssh -i "ubuntu_key.pem" ubuntu@ec2-44-202-88-33.compute-1.amazonaws.com

Note: In most cases, the guessed user name is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI user name.

Search results - prem.kalpana@g
Learner Lab
EC2 Management Console
Connect to instance | EC2 Manag
Services
Search
[Alt+S]
N. Virginia
voclabs/user2116389=Kalpana_Parsi @ 1672-7549-1921
S3
Feedback
Looking for language selection? Find it in the new Unified Settings
© 2022, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
ubuntu_key.ppk
Show all
Type here to search
21°C Haze
ENG IN
10:46 PM 11/15/2022

Open Putty



Connect to your instance i-0bbcc25027526ded8 (ubuntu instance) using any of these options

Instance ID

i-0bbcc25027526ded8 (ubuntu instance)

1. Open an SSH client.
2. Locate your private key file. The key used to launch this instance is ubuntu_key.pem
3. Run this command, if necessary, to ensure your key is not publicly viewable.
   chmod 400 ubuntu_key.pem
4. Connect to your instance using its Public DNS:
   ec2-44-202-88-33.compute-1.amazonaws.com

Example:

ssh -i "ubuntu_key.pem" ubuntu@ec2-44-202-88-33.compute-1.amazonaws.com

Note: In most cases, the guessed user name is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI user name.

PuTTY Configuration

Basic options for your PuTTY session

Specify the destination you want to connect to

Host Name (or IP address):

Port: 22

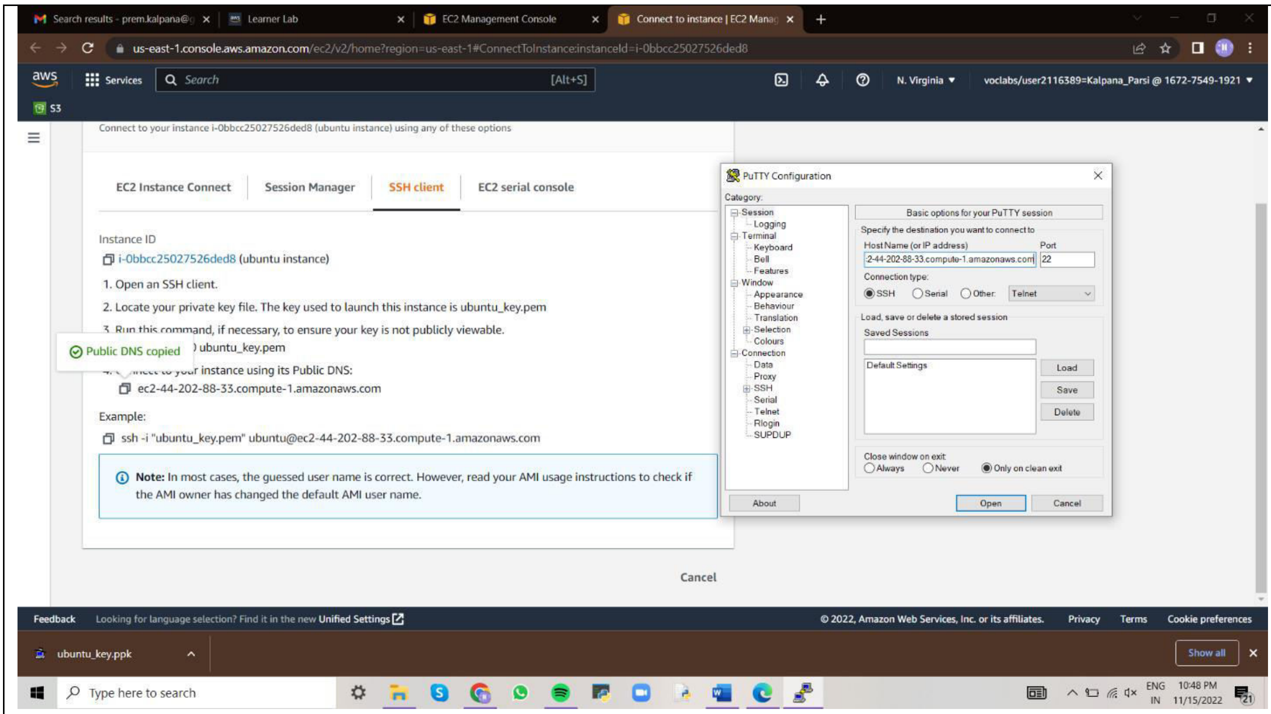
Connection type: ● SSH ○ Serial ○ Other: Telnet

Load, save or delete a stored session

Saved Sessions: Default Settings

Close window on exit: ○ Always ○ Never ● Only on clean exit

Enter the Public DNS of your Instance in Host Name(IP address)



us-east-1.console.aws.amazon.com/ec2/v2/home?region=us-east-1#ConnectToInstance:instanceId=i-0bbcc25027526ded8

Connect to your instance i-0bbcc25027526ded8 (ubuntu instance) using any of these options

EC2 Instance Connect | Session Manager | SSH client | EC2 serial console

Instance ID

i-0bbcc25027526ded8 (ubuntu instance)

1. Open an SSH client.
2. Locate your private key file. The key used to launch this instance is ubuntu_key.pem
3. Run this command, if necessary, to ensure your key is not publicly viewable.
) ubuntu_key.pem
4. Connect to your instance using its Public DNS:
   ec2-44-202-88-33.compute-1.amazonaws.com

Public DNS copied

Example:

```
ssh -i "ubuntu_key.pem" ubuntu@ec2-44-202-88-33.compute-1.amazonaws.com
```

Note: In most cases, the guessed user name is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI user name.

PuTTY Configuration

Basic options for your PuTTY session

Specify the destination you want to connect to

Host Name (or IP address): :2-44-202-88-33.compute-1.amazonaws.com   Port: 22

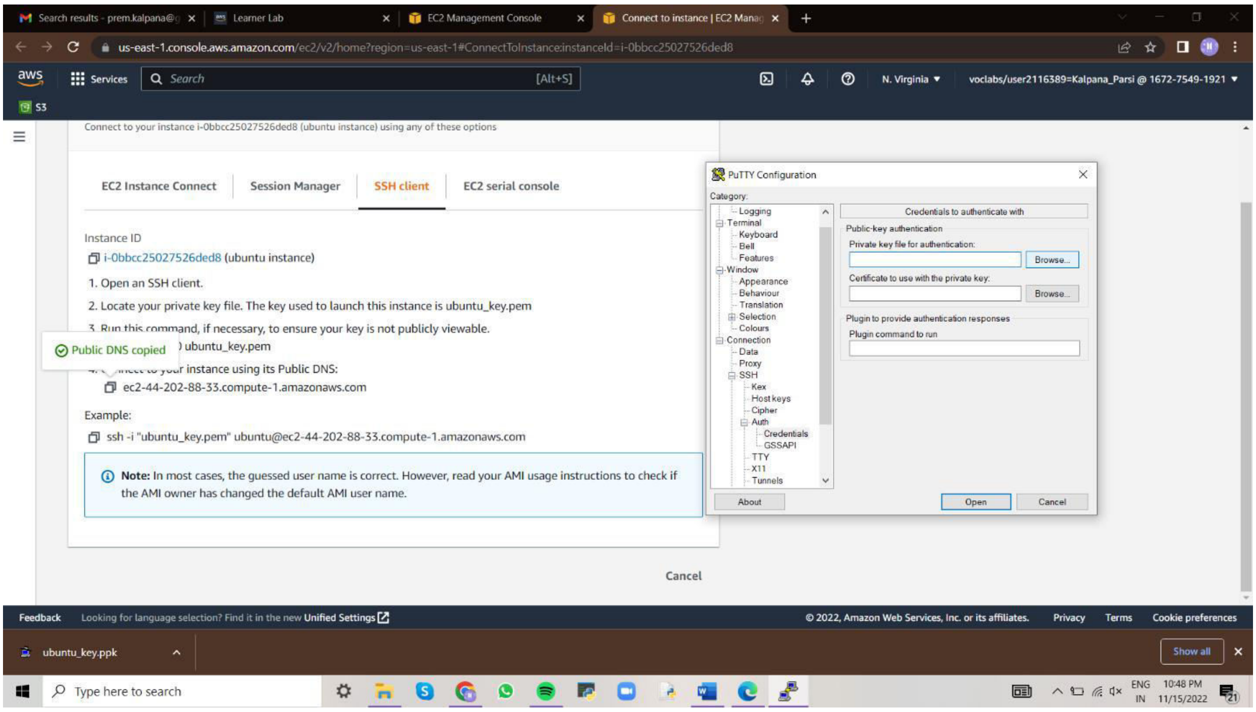
Connection type: ● SSH ○ Serial ○ Other: Telnet

Load, save or delete a stored session

Saved Sessions: Default Settings

Close window on exit: ○ Always ○ Never ● Only on clean exit

Click on Connection – SSH – Auth – Credentials –



Connect to your instance i-0bbcc25027526ded8 (ubuntu instance) using any of these options

EC2 Instance Connect | Session Manager | SSH client | EC2 serial console

Instance ID

i-0bbcc25027526ded8 (ubuntu instance)

1. Open an SSH client.
2. Locate your private key file. The key used to launch this instance is ubuntu_key.pem
3. Run this command, if necessary, to ensure your key is not publicly viewable.
) ubuntu_key.pem
4. Connect to your instance using its Public DNS:
   ec2-44-202-88-33.compute-1.amazonaws.com

Example:

ssh -i "ubuntu_key.pem" ubuntu@ec2-44-202-88-33.compute-1.amazonaws.com

Note: In most cases, the guessed user name is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI user name.

PuTTY Configuration

Credentials to authenticate with

Public-key authentication

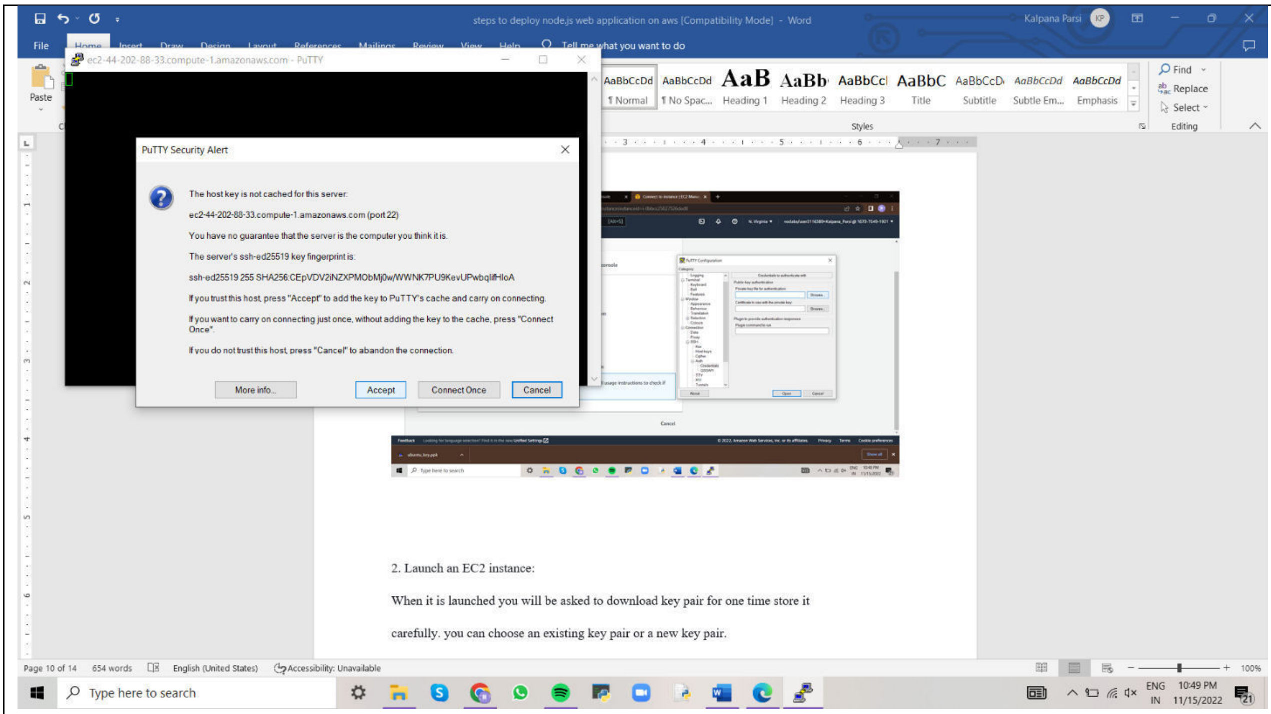
Private key file for authentication: [Browse...]

Certificate to use with the private key: [Browse...]

Plugin to provide authentication responses

Plugin command to run

Private key for Authentication - Browse - select the .ppk which was downloaded when EC2 instance is created



PuTTY Security Alert

The host key is not cached for this server:

ec2-44-202-88-33.compute-1.amazonaws.com (port 22)

You have no guarantee that the server is the computer you think it is.

The server's ssh-ed25519 key fingerprint is:

ssh-ed25519 255 SHA256:CEpVDV2iNZXPMObMj0w/WWNK7PU9KevUPwbqlifHloA

If you trust this host, press "Accept" to add the key to PuTTY's cache and carry on connecting.

If you want to carry on connecting just once, without adding the key to the cache, press "Connect Once".

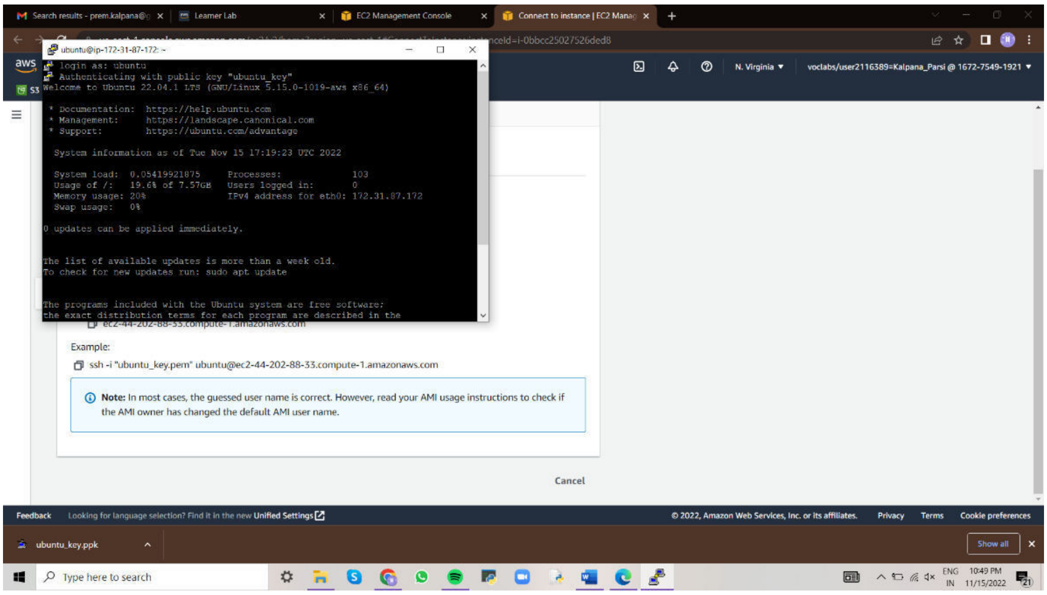
If you do not trust this host, press "Cancel" to abandon the connection.

2. Launch an EC2 instance:

When it is launched you will be asked to download key pair for one time store it carefully. you can choose an existing key pair or a new key pair.

Once entered, it will ask you to confirm, click on Accept

Once it is opened login as ubuntu



```
login as: ubuntu
Authenticating with public key "ubuntu_key"
Welcome to Ubuntu 22.04.1 LTS (GNU/Linux 5.15.0-1019-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Tue Nov 15 17:19:23 UTC 2022

  System load:  0.05419921875     Processes:             103
  Usage of /:   19.6% of 7.57GB   Users logged in:       0
  Memory usage: 20%               IPv4 address for eth0: 172.31.87.172
  Swap usage:   0%

0 updates can be applied immediately.


The list of available updates is more than a week old.
To check for new updates run: sudo apt update


The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
```

Example:

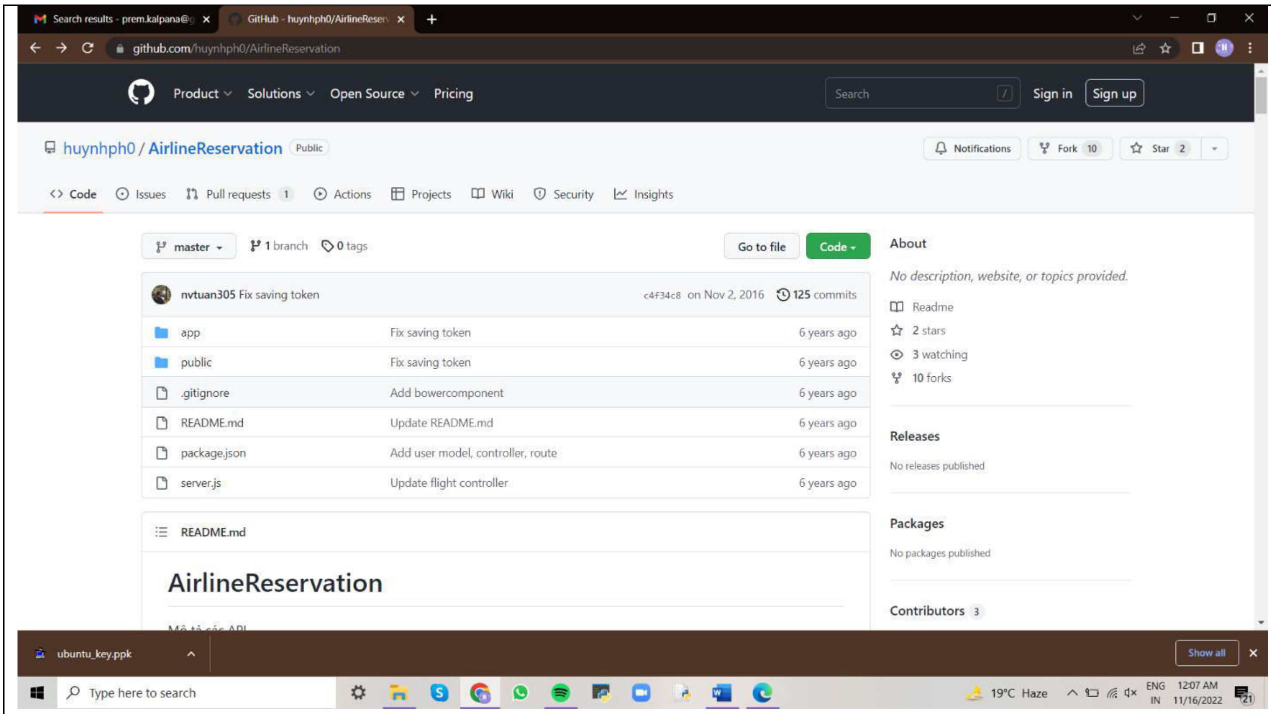
ssh -i "ubuntu_key.pem" ubuntu@ec2-44-202-88-33.compute-1.amazonaws.com

Note: In most cases, the guessed user name is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI user name.

mkdir demo

cd demo

git clone https://github.com/hoanghuynh1995/AirlineReservation



github.com/huynhph0/AirlineReservation

huynhph0 / AirlineReservation Public

master · 1 branch · 0 tags

nvtuan305 Fix saving token · c4f34c8 on Nov 2, 2016 · 125 commits

app	Fix saving token	6 years ago
public	Fix saving token	6 years ago
.gitignore	Add bowercomponent	6 years ago
README.md	Update README.md	6 years ago
package.json	Add user model, controller, route	6 years ago
server.js	Update flight controller	6 years ago

README.md

AirlineReservation

About

No description, website, or topics provided.

Readme · 2 stars · 3 watching · 10 forks

Releases

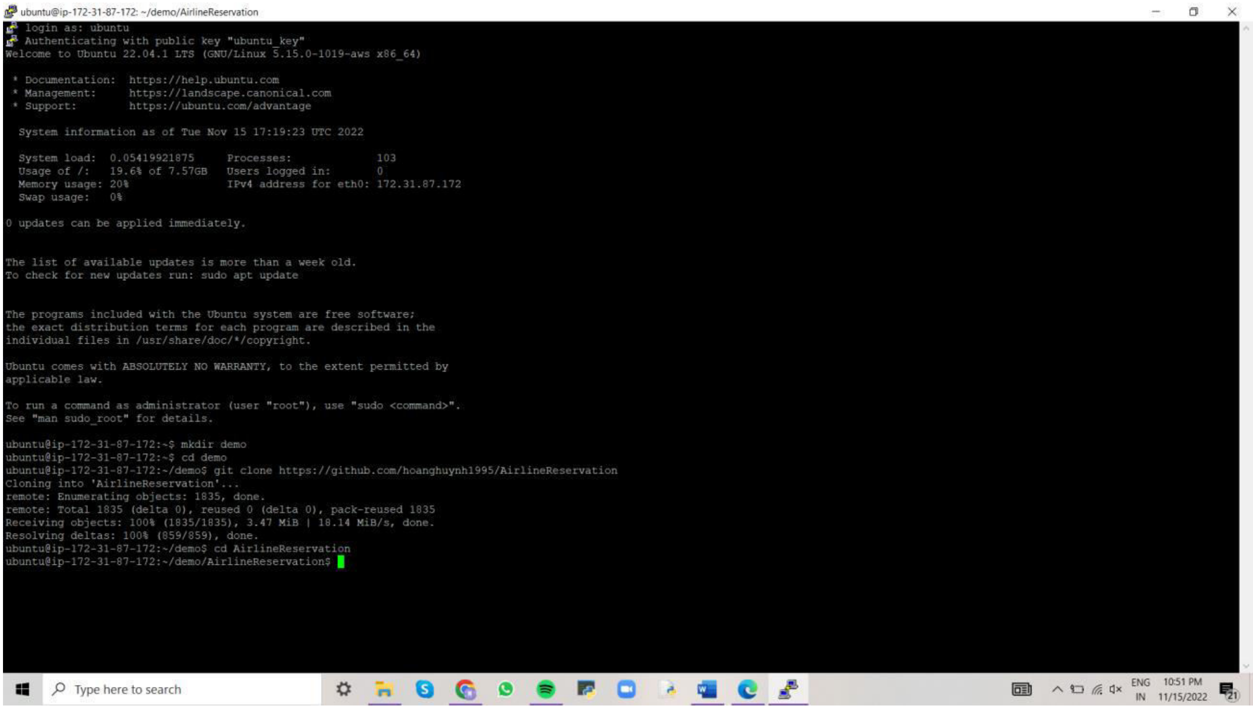
No releases published

Packages

No packages published

Contributors 3

cd AirlineReservation



ubuntu@ip-172-31-87-172: ~/demo/AirlineReservation

```
login as: ubuntu
Authenticating with public key "ubuntu_key"
Welcome to Ubuntu 22.04.1 LTS (GNU/Linux 5.15.0-1019-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Tue Nov 15 17:19:23 UTC 2022

  System load:  0.05419921875     Processes:             103
  Usage of /:   19.6% of 7.57GB   Users logged in:       0
  Memory usage: 20%               IPv4 address for eth0: 172.31.87.172
  Swap usage:   0%

0 updates can be applied immediately.


The list of available updates is more than a week old.
To check for new updates run: sudo apt update


The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-87-172:~$ mkdir demo
ubuntu@ip-172-31-87-172:~$ cd demo
ubuntu@ip-172-31-87-172:~/demo$ git clone https://github.com/hoanghuynh1995/AirlineReservation
Cloning into 'AirlineReservation'...
remote: Enumerating objects: 1835, done.
remote: Total 1835 (delta 0), reused 0 (delta 0), pack-reused 1835
Receiving objects: 100% (1835/1835), 3.47 MiB | 18.14 MiB/s, done.
Resolving deltas: 100% (859/859), done.
ubuntu@ip-172-31-87-172:~/demo$ cd AirlineReservation
ubuntu@ip-172-31-87-172:~/demo/AirlineReservation$
```

sudo apt-get update //to download package information from all configured sources

sudo apt-get install npm

//to install Node.js on ubuntu, we must first install npm (node package manager) select Yes

Ok

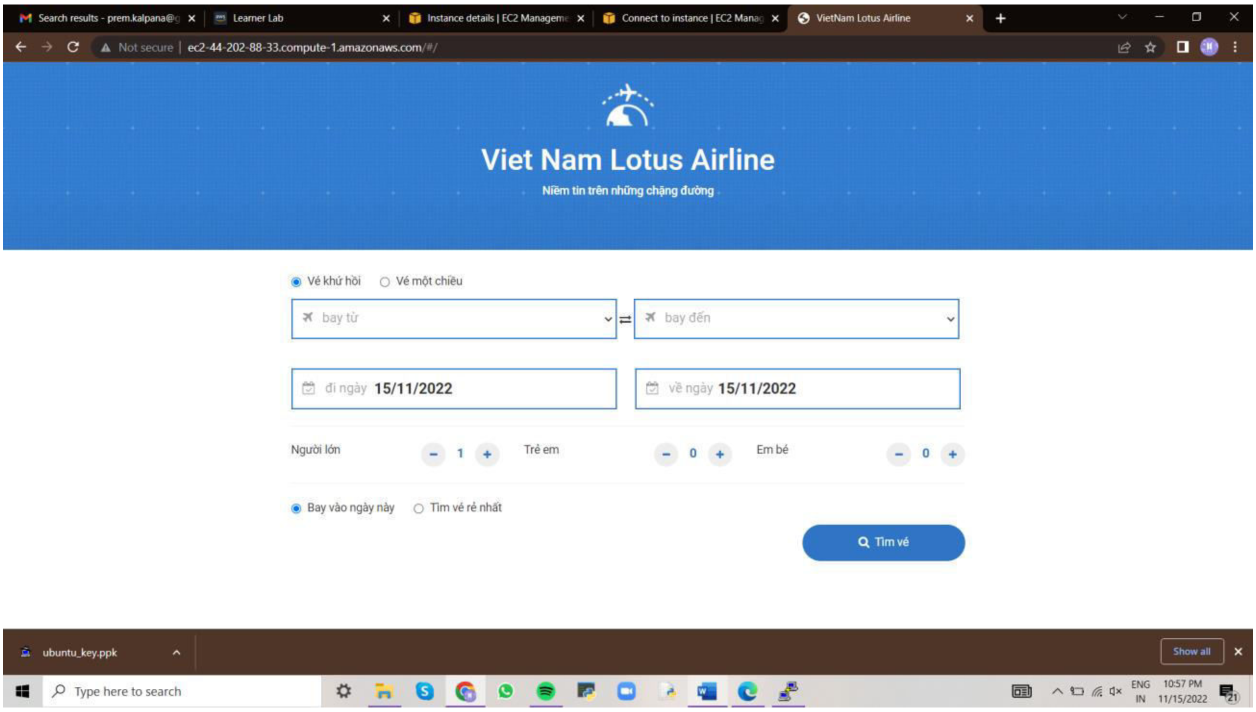
npm install

sudo apt-get install nodejs //to install Node.js on ubuntu

open server.js file using vi editor and change the port no to 80, and save file and exit

sudo node server.js

Copy public DNS of your instance in new tab and view the deployed web application.



Not secure | ec2-44-202-88-33.compute-1.amazonaws.com/#/

Viet Nam Lotus Airline

Niềm tin trên những chặng đường

● Vé khứ hồi ○ Vé một chiều

bay từ ⇄ bay đến

đi ngày 15/11/2022

về ngày 15/11/2022

Người lớn − 1 +

Trẻ em − 0 +

Em bé − 0 +

● Bay vào ngày này ○ Tìm vé rẻ nhất

LAB PROGRAMS

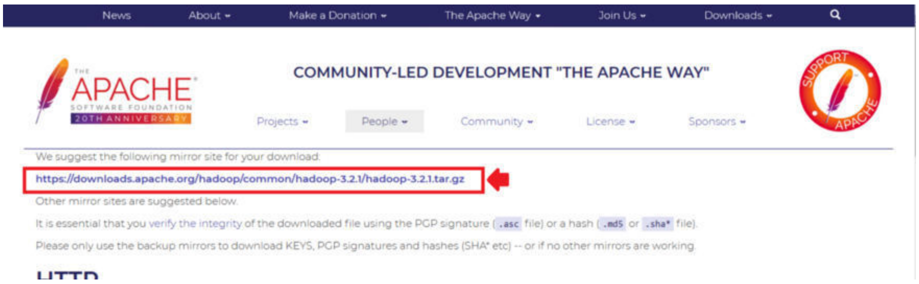
Implement a distributed application on Hadoop framework.

1. **Prerequisites**

   1. Java 8 runtime environment (JRE): Hadoop 3 requires a Java 8 installation. I prefer using the offline installer.
   2. Java 8 development Kit (JDK)
   3. To unzip downloaded Hadoop binaries, we should install 7zip.

2. Download Hadoop binaries

The first step is to download Hadoop binaries from the official website. The binary package size is about 342 MB.



We suggest the following mirror site for your download:

https://downloads.apache.org/hadoop/common/hadoop-3.2.1/hadoop-3.2.1.tar.gz

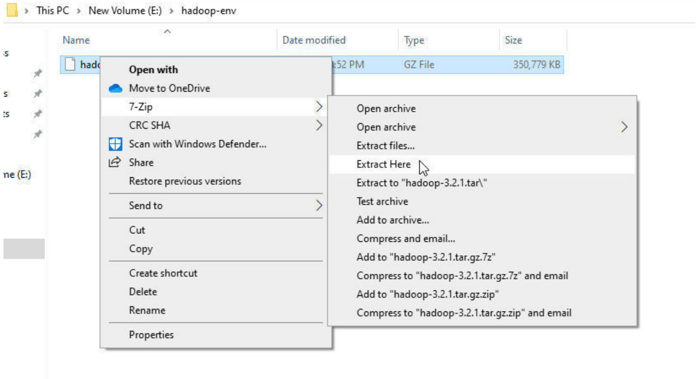
Other mirror sites are suggested below.

It is essential that you verify the integrity of the downloaded file using the PGP signature (.asc file) or a hash (.md5 or .sha* file).

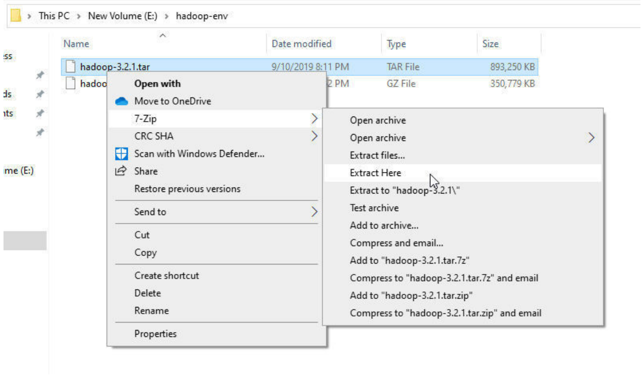
Please only use the backup mirrors to download KEYS, PGP signatures and hashes (SHA* etc) -- or if no other mirrors are working.

HTTP

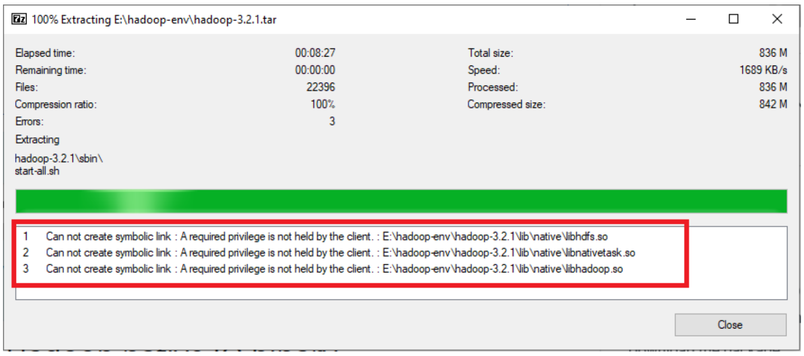
After finishing the file download, we should unpack the package using 7zip int two steps. First, we should extract the hadoop-3.2.1.tar.gz library, and then, we should unpack the extracted tar file:



This PC > New Volume (E:) > hadoop-env
Name
Date modified
Type
Size
hadc
:52 PM
GZ File
350,779 KB
Open with
Move to OneDrive
7-Zip
CRC SHA
Scan with Windows Defender...
Share
Restore previous versions
Send to
Cut
Copy
Create shortcut
Delete
Rename
Properties
Open archive
Open archive
Extract files...
Extract Here
Extract to "hadoop-3.2.1.tar\"
Test archive
Add to archive...
Compress and email...
Add to "hadoop-3.2.1.tar.gz.7z"
Compress to "hadoop-3.2.1.tar.gz.7z" and email
Add to "hadoop-3.2.1.tar.gz.zip"
Compress to "hadoop-3.2.1.tar.gz.zip" and email



The tar file extraction may take some minutes to finish. In the end, you may see some warnings about symbolic link creation. Just ignore these warnings since they are not related to windows.



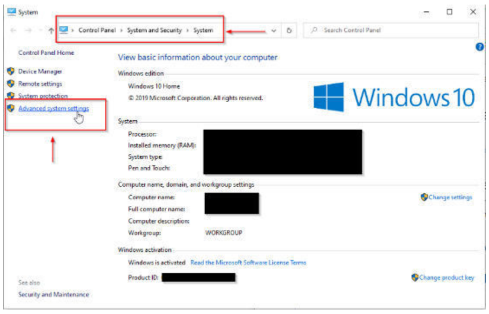
100% Extracting E:\hadoop-env\hadoop-3.2.1.tar

```
Elapsed time:            00:08:27      Total size:          836 M
Remaining time:          00:00:00      Speed:           1689 KB/s
Files:                      22396      Processed:           836 M
Compression ratio:           100%      Compressed size:     842 M
Errors:                         3
Extracting
hadoop-3.2.1\sbin\
start-all.sh

1   Can not create symbolic link : A required privilege is not held by the client. : E:\hadoop-env\hadoop-3.2.1\lib\native\libhdfs.so
2   Can not create symbolic link : A required privilege is not held by the client. : E:\hadoop-env\hadoop-3.2.1\lib\native\libnativetask.so
3   Can not create symbolic link : A required privilege is not held by the client. : E:\hadoop-env\hadoop-3.2.1\lib\native\libhadoop.so
```

3. Setting up environment variables

After installing Hadoop and its prerequisites, we should configure the environment variables to define Hadoop and Java default paths.

To edit environment variables, go to Control Panel > System and Security > System (or right-click > properties on My Computer icon) and click on the "Advanced system settings" link.



System

Control Panel > System and Security > System

View basic information about your computer

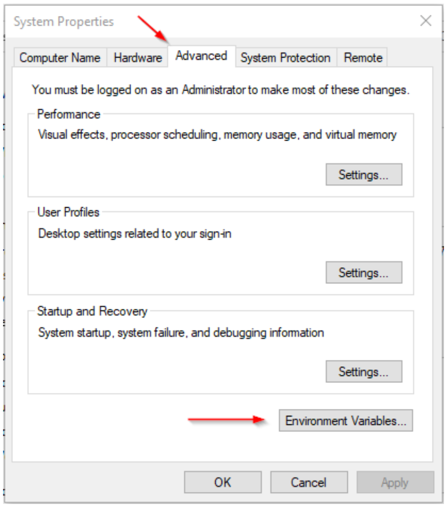
Windows edition
Windows 10 Home
© 2019 Microsoft Corporation. All rights reserved.

System
Processor: [redacted]
Installed memory (RAM):
System type:
Pen and Touch:

Computer name, domain, and workgroup settings
Computer name: [redacted]
Full computer name:
Computer description:
Workgroup: WORKGROUP

Windows activation
Windows is activated Read the Microsoft Software License Terms
Product ID: [redacted]

When the "Advanced system settings" dialog appears, go to the "Advanced" tab and click on the "Environment variables" button located on the bottom of the dialog.



System Properties

You must be logged on as an Administrator to make most of these changes.

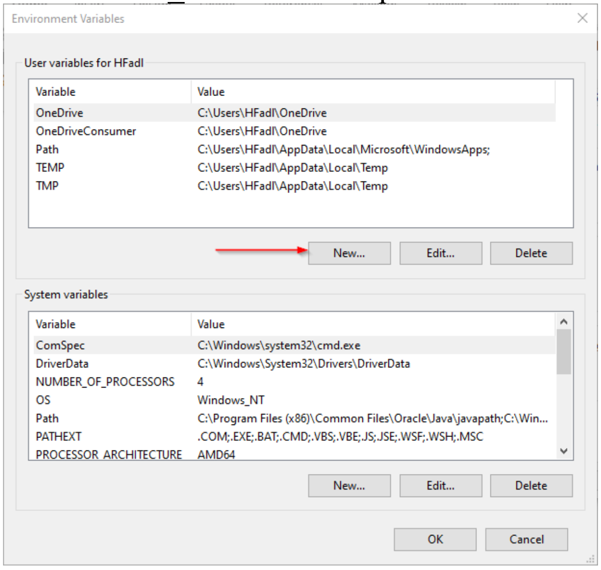
Performance
Visual effects, processor scheduling, memory usage, and virtual memory

User Profiles
Desktop settings related to your sign-in

Startup and Recovery
System startup, system failure, and debugging information

In the “Environment Variables” dialog, press the “New” button to add a new variable.There are two variables to define:

1. JAVA_HOME: JDK installation folder path
2. HADOOP_HOME: Hadoop installation folder path


Environment Variables

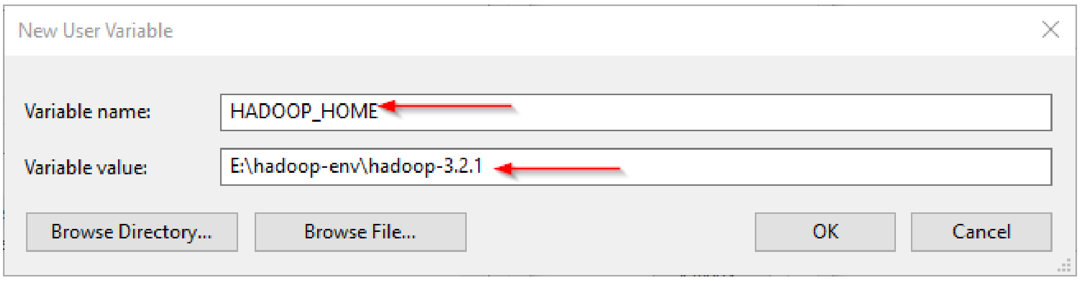
User variables for HFadl

Variable	Value
OneDrive	C:\Users\HFadl\OneDrive
OneDriveConsumer	C:\Users\HFadl\OneDrive
Path	C:\Users\HFadl\AppData\Local\Microsoft\WindowsApps;
TEMP	C:\Users\HFadl\AppData\Local\Temp
TMP	C:\Users\HFadl\AppData\Local\Temp

System variables

Variable	Value
ComSpec	C:\Windows\system32\cmd.exe
DriverData	C:\Windows\System32\Drivers\DriverData
NUMBER_OF_PROCESSORS	4
OS	Windows_NT
Path	C:\Program Files (x86)\Common Files\Oracle\Java\javapath;C:\Win...
PATHEXT	.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC
PROCESSOR_ARCHITECTURE	AMD64

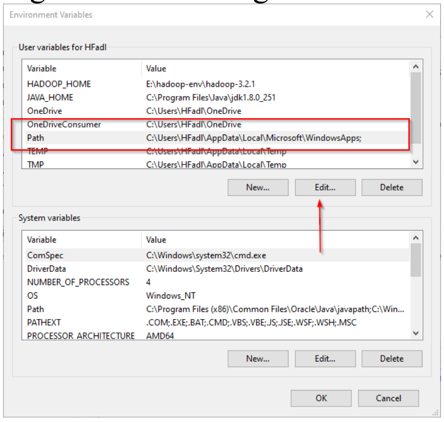
Now, we should edit the PATH variable to add the Java and Hadoop binaries paths as shown in the following screenshots.


New User Variable

Variable name: HADOOP_HOME

Variable value: E:\hadoop-env\hadoop-3.2.1

Figure 10 — Editing the PATH variable


Environment Variables

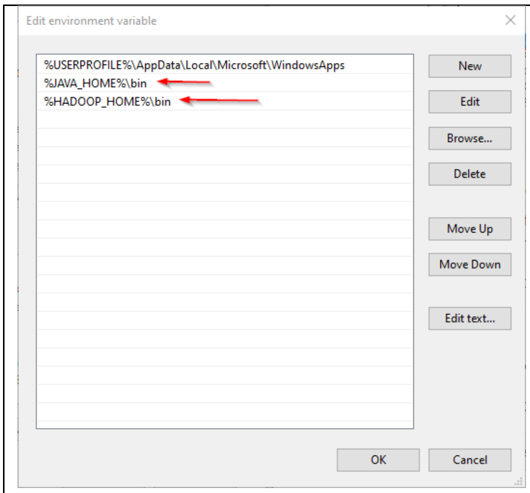
User variables for HFadl

Variable	Value
HADOOP_HOME	E:\hadoop-env\hadoop-3.2.1
JAVA_HOME	C:\Program Files\Java\jdk1.8.0_251
OneDrive	C:\Users\HFadl\OneDrive
OneDriveConsumer	C:\Users\HFadl\OneDrive
Path	C:\Users\HFadl\AppData\Local\Microsoft\WindowsApps;
TEMP	C:\Users\HFadl\AppData\Local\Temp
TMP	C:\Users\HFadl\AppData\Local\Temp

System variables

Variable	Value
ComSpec	C:\Windows\system32\cmd.exe
DriverData	C:\Windows\System32\Drivers\DriverData
NUMBER_OF_PROCESSORS	4
OS	Windows_NT
Path	C:\Program Files (x86)\Common Files\Oracle\Java\javapath;C:\Win...
PATHEXT	.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC
PROCESSOR_ARCHITECTURE	AMD64

Figure 11 — Editing PATH variable



%USERPROFILE%\AppData\Local\Microsoft\WindowsApps
%JAVA_HOME%\bin
%HADOOP_HOME%\bin

Figure 12— Adding new paths to the PATH variable

4. Configuring Hadoop cluster

There are four files we should alter to configure Hadoop cluster:

1. %HADOOP_HOME%\etc\hadoop\hdfs-site.xml
2. %HADOOP_HOME%\etc\hadoop\core-site.xml
3. %HADOOP_HOME%\etc\hadoop\mapred-site.xml
4. %HADOOP_HOME%\etc\hadoop\yarn-site.xml

4.1. HDFS site configuration

As we know, Hadoop is built using a master-slave paradigm. Before altering the HDFS configuration file, we should create a directory to store all master node (name node) data and another one to store data (data node). In this example, we created the following directories:

- E:\hadoop-env\hadoop-3.2.1\data\dfs\namenode
- E:\hadoop-env\hadoop-3.2.1\data\dfs\datanode

Now, let's open “hdfs-site.xml” file located in “%HADOOP_HOME%\etc\hadoop” directory, and we should add the following properties within

4.2. Core site configuration

Now, we should configure the name node URL adding the following XML code into the <configuration></configuration> element within “core-site.xml”:

```
<property><name>fs.default.name</name><value>hdfs://localhost:9820</value></property>
```

4.3. Map Reduce site configuration

Now, we should add the following XML code into the <configuration></configuration> element within "mapred-site.xml":

```
<property><name>mapreduce.framework.name</name><value>yarn</value><description>MapReduce
framework name</description></property>
```

4.4. Yarn site configuration

Now, we should add the following XML code into the <configuration></configuration> element within "yarn-site.xml":

```
<property><name>yarn.nodemanager.aux-
services</name><value>mapreduce_shuffle</value><description>Yarn Node Manager Aux
Service</description></property
```

5. Formatting Name node

After finishing the configuration, let's try to format the name node using the following command:

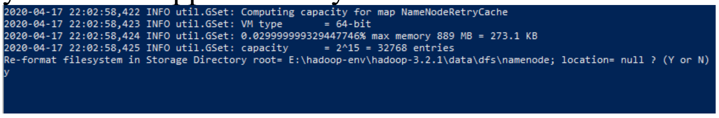
```
hdfs namenode -format
```

Due to a bug you will receive the following error.

This issue will be solved within the next release. For now, you can fix it temporarily using the following steps (reference):

1. Download hadoop-hdfs-3.2.1.jar file from the following link.
2. Rename the file name hadoop-hdfs-3.2.1.jar to hadoop-hdfs-3.2.1.bak in folder %HADOOP_HOME%\share\hadoop\hdfs
3. Copy the downloaded hadoop-hdfs-3.2.1.jar to folder %HADOOP_HOME%\share\hadoop\hdfs

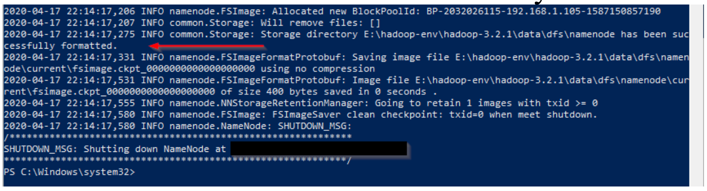
Now, if we try to re-execute the format command (Run the command prompt or PowerShell as administrator), you need to approve file system format.



```
2020-04-17 22:02:58,422 INFO util.GSet: Computing capacity for map NameNodeRetryCache
2020-04-17 22:02:58,423 INFO util.GSet: VM type       = 64-bit
2020-04-17 22:02:58,424 INFO util.GSet: 0.029999999329447746% max memory 889 MB = 273.1 KB
2020-04-17 22:02:58,425 INFO util.GSet: capacity      = 2^15 = 32768 entries
Re-format filesystem in Storage Directory root= E:\hadoop-env\hadoop-3.2.1\data\dfs\namenode; location= null ? (Y or N)
y
```

Figure 15 — File system format approval

And the command is executed successfully:



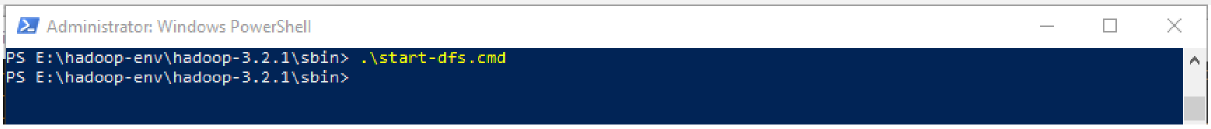
```
2020-04-17 22:14:17,206 INFO namenode.FSImage: Allocated new BlockPoolId: BP-2032026115-192.168.1.105-1587150857190
2020-04-17 22:14:17,207 INFO common.Storage: Will remove files: []
2020-04-17 22:14:17,275 INFO common.Storage: Storage directory E:\hadoop-env\hadoop-3.2.1\data\dfs\namenode has been suc
cessfully formatted.
2020-04-17 22:14:17,331 INFO namenode.FSImageFormatProtobuf: Saving image file E:\hadoop-env\hadoop-3.2.1\data\dfs\namen
ode\current\fsimage.ckpt_0000000000000000000 using no compression
2020-04-17 22:14:17,531 INFO namenode.FSImageFormatProtobuf: Image file E:\hadoop-env\hadoop-3.2.1\data\dfs\namenode\cur
rent\fsimage.ckpt_0000000000000000000 of size 400 bytes saved in 0 seconds .
2020-04-17 22:14:17,555 INFO namenode.NNStorageRetentionManager: Going to retain 1 images with txid >= 0
2020-04-17 22:14:17,580 INFO namenode.FSImage: FSImageSaver clean checkpoint: txid=0 when meet shutdown.
2020-04-17 22:14:17,580 INFO namenode.NameNode: SHUTDOWN_MSG:
/************************************************************
SHUTDOWN_MSG: Shutting down NameNode at [redacted]
************************************************************/
PS C:\Windows\system32>
```

Figure 16 — Command executed successfully

6. Starting Hadoop services

Now, we will open PowerShell, and navigate to "%HADOOP_HOME%\sbin" directory. Then we will run the following command to start the Hadoop nodes:

```
.\start-dfs.cmd
```

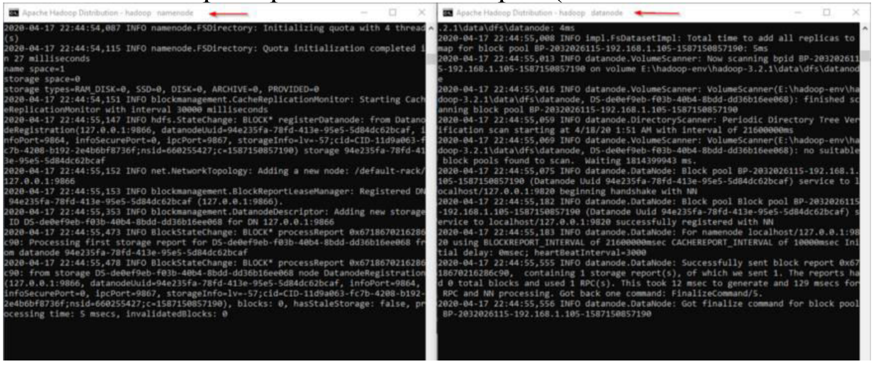


Administrator: Windows PowerShell

```
PS E:\hadoop-env\hadoop-3.2.1\sbin> .\start-dfs.cmd
PS E:\hadoop-env\hadoop-3.2.1\sbin>
```

Figure 17 — StartingHadoop nodes

Two command prompt windows will open (one for the name node and one for the data node) as follows:



Apache Hadoop Distribution - hadoop namenode

```
2020-04-17 22:44:54,087 INFO namenode.FSDirectory: Initializing quota with 4 thread
(s)
2020-04-17 22:44:54,115 INFO namenode.FSDirectory: Quota initialization completed i
n 27 milliseconds
name space=1
storage space=0
storage types=RAM_DISK=0, SSD=0, DISK=0, ARCHIVE=0, PROVIDED=0
2020-04-17 22:44:54,151 INFO blockmanagement.CacheReplicationMonitor: Starting Cach
eReplicationMonitor with interval 30000 milliseconds
2020-04-17 22:44:55,147 INFO hdfs.StateChange: BLOCK* registerDatanode: from Datano
deRegistration(127.0.0.1:9866, datanodeUuid=94e235fa-78fd-413e-95e5-5d84dc62bcaf, i
nfoPort=9864, infoSecurePort=0, ipcPort=9867, storageInfo=lv=-57;cid=CID-11d9a063-f
c7b-4208-b192-2e4b6bf8736f;nsid=660255427;c=1587150857190) storage 94e235fa-78fd-41
3e-95e5-5d84dc62bcaf
2020-04-17 22:44:55,152 INFO net.NetworkTopology: Adding a new node: /default-rack/
127.0.0.1:9866
2020-04-17 22:44:55,153 INFO blockmanagement.BlockReportLeaseManager: Registered DN
 94e235fa-78fd-413e-95e5-5d84dc62bcaf (127.0.0.1:9866).
2020-04-17 22:44:55,353 INFO blockmanagement.DatanodeDescriptor: Adding new storage
 ID DS-de0ef9eb-f03b-40b4-8bdd-dd36b16ee068 for DN 127.0.0.1:9866
2020-04-17 22:44:55,473 INFO BlockStateChange: BLOCK* processReport 0x6718670216286
c90: Processing first storage report for DS-de0ef9eb-f03b-40b4-8bdd-dd36b16ee068 fr
om datanode 94e235fa-78fd-413e-95e5-5d84dc62bcaf
2020-04-17 22:44:55,478 INFO BlockStateChange: BLOCK* processReport 0x6718670216286
c90: from storage DS-de0ef9eb-f03b-40b4-8bdd-dd36b16ee068 node DatanodeRegistration
(127.0.0.1:9866, datanodeUuid=94e235fa-78fd-413e-95e5-5d84dc62bcaf, infoPort=9864,
infoSecurePort=0, ipcPort=9867, storageInfo=lv=-57;cid=CID-11d9a063-fc7b-4208-b192-
2e4b6bf8736f;nsid=660255427;c=1587150857190), blocks: 0, hasStaleStorage: false, pr
ocessing time: 5 msecs, invalidatedBlocks: 0
```

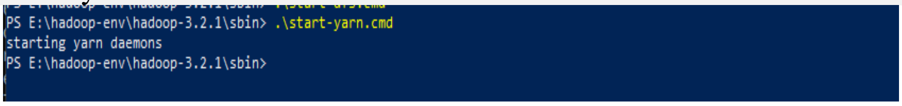
Apache Hadoop Distribution - hadoop datanode

```
.2.1\data\dfs\datanode: 4ms
2020-04-17 22:44:55,008 INFO impl.FsDatasetImpl: Total time to add all replicas to
map for block pool BP-2032026115-192.168.1.105-1587150857190: 5ms
2020-04-17 22:44:55,013 INFO datanode.VolumeScanner: Now scanning bpid BP-203202611
5-192.168.1.105-1587150857190 on volume E:\hadoop-env\hadoop-3.2.1\data\dfs\datanod
e
2020-04-17 22:44:55,016 INFO datanode.VolumeScanner: VolumeScanner(E:\hadoop-env\ha
doop-3.2.1\data\dfs\datanode, DS-de0ef9eb-f03b-40b4-8bdd-dd36b16ee068): finished sc
anning block pool BP-2032026115-192.168.1.105-1587150857190
2020-04-17 22:44:55,059 INFO datanode.DirectoryScanner: Periodic Directory Tree Ver
ification scan starting at 4/18/20 1:51 AM with interval of 21600000ms
2020-04-17 22:44:55,069 INFO datanode.VolumeScanner: VolumeScanner(E:\hadoop-env\ha
doop-3.2.1\data\dfs\datanode, DS-de0ef9eb-f03b-40b4-8bdd-dd36b16ee068): no suitable
 block pools found to scan.  Waiting 1814399943 ms.
2020-04-17 22:44:55,075 INFO datanode.DataNode: Block pool BP-2032026115-192.168.1.
105-1587150857190 (Datanode Uuid 94e235fa-78fd-413e-95e5-5d84dc62bcaf) service to l
ocalhost/127.0.0.1:9820 beginning handshake with NN
2020-04-17 22:44:55,182 INFO datanode.DataNode: Block pool Block pool BP-2032026115
-192.168.1.105-1587150857190 (Datanode Uuid 94e235fa-78fd-413e-95e5-5d84dc62bcaf) s
ervice to localhost/127.0.0.1:9820 successfully registered with NN
2020-04-17 22:44:55,183 INFO datanode.DataNode: For namenode localhost/127.0.0.1:98
20 using BLOCKREPORT_INTERVAL of 21600000msec CACHEREPORT_INTERVAL of 10000msec Ini
tial delay: 0msec; heartBeatInterval=3000
2020-04-17 22:44:55,555 INFO datanode.DataNode: Successfully sent block report 0x67
18670216286c90,  containing 1 storage report(s), of which we sent 1. The reports ha
d 0 total blocks and used 1 RPC(s). This took 12 msec to generate and 129 msecs for
 RPC and NN processing. Got back one command: FinalizeCommand/5.
2020-04-17 22:44:55,556 INFO datanode.DataNode: Got finalize command for block pool
 BP-2032026115-192.168.1.105-1587150857190
```

Figure 18 — Hadoop nodes command prompt windows

Next, we must start the Hadoop Yarn service using the following command:

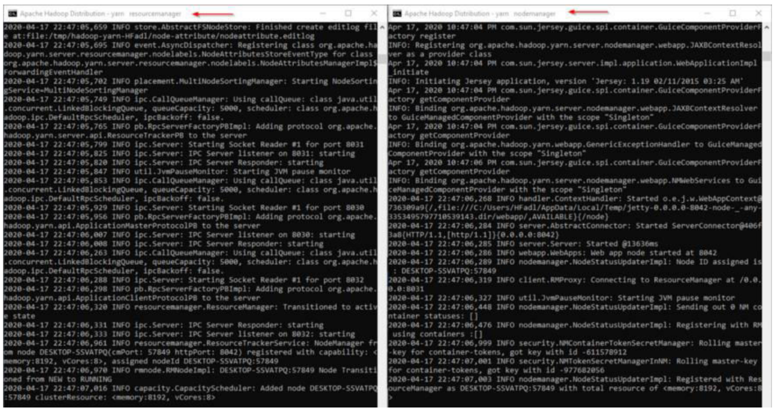
```
./start-yarn.cmd
```



```
PS E:\hadoop-env\hadoop-3.2.1\sbin> .\start-yarn.cmd
starting yarn daemons
PS E:\hadoop-env\hadoop-3.2.1\sbin>
```

Figure 19 — Starting Hadoop Yarn services

Two command prompt windows will open (one for the resource manager and one for the node manager) as follows:



Apache Hadoop Distribution - yarn resourcemanager

```
2020-04-17 22:47:05,659 INFO store.AbstractFSNodeStore: Finished create editlog fil
e at:file:/tmp/hadoop-yarn-HFadl/node-attribute/nodeattribute.editlog
2020-04-17 22:47:05,695 INFO event.AsyncDispatcher: Registering class org.apache.ha
doop.yarn.server.resourcemanager.nodelabels.NodeAttributesStoreEventType for class
org.apache.hadoop.yarn.server.resourcemanager.nodelabels.NodeAttributesManagerImpl$
ForwardingEventHandler
2020-04-17 22:47:05,702 INFO placement.MultiNodeSortingManager: Starting NodeSortin
gService=MultiNodeSortingManager
2020-04-17 22:47:05,749 INFO ipc.CallQueueManager: Using callQueue: class java.util
.concurrent.LinkedBlockingQueue, queueCapacity: 5000, scheduler: class org.apache.h
adoop.ipc.DefaultRpcScheduler, ipcBackoff: false.
2020-04-17 22:47:05,765 INFO pb.RpcServerFactoryPBImpl: Adding protocol org.apache.
hadoop.yarn.server.api.ResourceTrackerPB to the server
2020-04-17 22:47:05,799 INFO ipc.Server: Starting Socket Reader #1 for port 8031
2020-04-17 22:47:05,825 INFO ipc.Server: IPC Server listener on 8031: starting
2020-04-17 22:47:05,820 INFO ipc.Server: IPC Server Responder: starting
2020-04-17 22:47:05,847 INFO util.JvmPauseMonitor: Starting JVM pause monitor
2020-04-17 22:47:05,853 INFO ipc.CallQueueManager: Using callQueue: class java.util
.concurrent.LinkedBlockingQueue, queueCapacity: 5000, scheduler: class org.apache.h
adoop.ipc.DefaultRpcScheduler, ipcBackoff: false.
2020-04-17 22:47:05,929 INFO ipc.Server: Starting Socket Reader #1 for port 8030
2020-04-17 22:47:05,956 INFO pb.RpcServerFactoryPBImpl: Adding protocol org.apache.
hadoop.yarn.api.ApplicationMasterProtocolPB to the server
2020-04-17 22:47:06,007 INFO ipc.Server: IPC Server listener on 8030: starting
2020-04-17 22:47:06,008 INFO ipc.Server: IPC Server Responder: starting
2020-04-17 22:47:06,263 INFO ipc.CallQueueManager: Using callQueue: class java.util
.concurrent.LinkedBlockingQueue, queueCapacity: 5000, scheduler: class org.apache.h
adoop.ipc.DefaultRpcScheduler, ipcBackoff: false.
2020-04-17 22:47:06,288 INFO ipc.Server: Starting Socket Reader #1 for port 8032
2020-04-17 22:47:06,298 INFO pb.RpcServerFactoryPBImpl: Adding protocol org.apache.
hadoop.yarn.api.ApplicationClientProtocolPB to the server
2020-04-17 22:47:06,320 INFO resourcemanager.ResourceManager: Transitioned to activ
e state
2020-04-17 22:47:06,331 INFO ipc.Server: IPC Server Responder: starting
2020-04-17 22:47:06,333 INFO ipc.Server: IPC Server listener on 8032: starting
2020-04-17 22:47:06,961 INFO resourcemanager.ResourceTrackerService: NodeManager fr
om node DESKTOP-SSVATPQ(cmPort: 57849 httpPort: 8042) registered with capability: <
memory:8192, vCores:8>, assigned nodeId DESKTOP-SSVATPQ:57849
2020-04-17 22:47:06,970 INFO rmnode.RMNodeImpl: DESKTOP-SSVATPQ:57849 Node Transiti
oned from NEW to RUNNING
2020-04-17 22:47:07,016 INFO capacity.CapacityScheduler: Added node DESKTOP-SSVATPQ
:57849 clusterResource: <memory:8192, vCores:8>
```

Apache Hadoop Distribution - yarn nodemanager

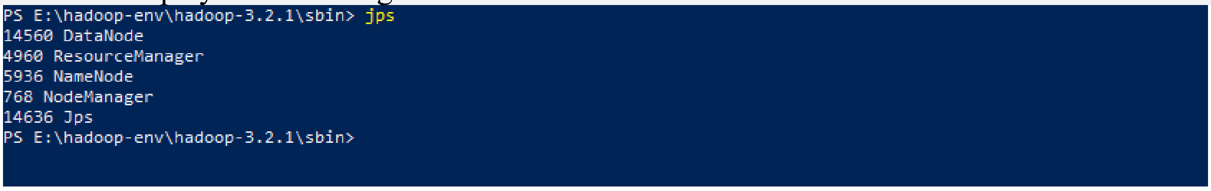
```
Apr 17, 2020 10:47:04 PM com.sun.jersey.guice.spi.container.GuiceComponentProviderF
actory register
INFO: Registering org.apache.hadoop.yarn.server.nodemanager.webapp.JAXBContextResol
ver as a provider class
Apr 17, 2020 10:47:04 PM com.sun.jersey.server.impl.application.WebApplicationImpl
_initiate
INFO: Initiating Jersey application, version 'Jersey: 1.19 02/11/2015 03:25 AM'
Apr 17, 2020 10:47:04 PM com.sun.jersey.guice.spi.container.GuiceComponentProviderF
actory getComponentProvider
INFO: Binding org.apache.hadoop.yarn.server.nodemanager.webapp.JAXBContextResolver
to GuiceManagedComponentProvider with the scope "Singleton"
Apr 17, 2020 10:47:04 PM com.sun.jersey.guice.spi.container.GuiceComponentProviderF
actory getComponentProvider
INFO: Binding org.apache.hadoop.yarn.webapp.GenericExceptionHandler to GuiceManaged
ComponentProvider with the scope "Singleton"
Apr 17, 2020 10:47:06 PM com.sun.jersey.guice.spi.container.GuiceComponentProviderF
actory getComponentProvider
INFO: Binding org.apache.hadoop.yarn.server.nodemanager.webapp.NMWebServices to Gui
ceManagedComponentProvider with the scope "Singleton"
2020-04-17 22:47:06,268 INFO handler.ContextHandler: Started o.e.j.w.WebAppContext@
736309a9{/,file:///C:/Users/HFadl/AppData/Local/Temp/jetty-0.0.0.0-8042-node-_-any-
3353495797710539143.dir/webapp/,AVAILABLE}{/node}
2020-04-17 22:47:06,284 INFO server.AbstractConnector: Started ServerConnector@406f
3a8{HTTP/1.1,[http/1.1]}{0.0.0.0:8042}
2020-04-17 22:47:06,285 INFO server.Server: Started @13636ms
2020-04-17 22:47:06,286 INFO webapp.WebApps: Web app node started at 8042
2020-04-17 22:47:06,289 INFO nodemanager.NodeStatusUpdaterImpl: Node ID assigned is
 : DESKTOP-SSVATPQ:57849
2020-04-17 22:47:06,319 INFO client.RMProxy: Connecting to ResourceManager at /0.0.
0.0:8031
2020-04-17 22:47:06,327 INFO util.JvmPauseMonitor: Starting JVM pause monitor
2020-04-17 22:47:06,448 INFO nodemanager.NodeStatusUpdaterImpl: Sending out 0 NM co
ntainer statuses: []
2020-04-17 22:47:06,476 INFO nodemanager.NodeStatusUpdaterImpl: Registering with RM
 using containers :[]
2020-04-17 22:47:06,999 INFO security.NMContainerTokenSecretManager: Rolling master
-key for container-tokens, got key with id -611578912
2020-04-17 22:47:07,001 INFO security.NMTokenSecretManagerInNM: Rolling master-key
for container-tokens, got key with id -977682056
2020-04-17 22:47:07,003 INFO nodemanager.NodeStatusUpdaterImpl: Registered with Res
ourceManager as DESKTOP-SSVATPQ:57849 with total resource of <memory:8192, vCores:8
>
```

Figure 20— Node manager and Resource manager command prompt windows

To make sure that all services started successfully, we can run the following command:

Jps

It should display the following services:



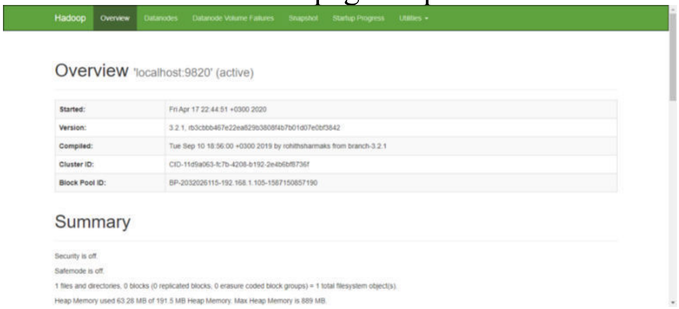
```
PS E:\hadoop-env\hadoop-3.2.1\sbin> jps
14560 DataNode
4960 ResourceManager
5936 NameNode
768 NodeManager
14636 Jps
PS E:\hadoop-env\hadoop-3.2.1\sbin>
```

Figure 21 — Executing jps command

7. Hadoop Web UI

There are three web user interfaces to be used:

- Name node web page: http://localhost:9870/dfshealth.html



Overview 'localhost:9820' (active)

Started:	Fri Apr 17 22:44:51 +0300 2020
Version:	3.2.1, rb3cbbb467e22ea829b3808f4b7b01d07e0bf3842
Compiled:	Tue Sep 10 18:56:00 +0300 2019 by rohithsharmaks from branch-3.2.1
Cluster ID:	CID-11d9a063-fc7b-4208-b192-2e4b6bf8736f
Block Pool ID:	BP-2032026115-192.168.1.105-1587150857190

Summary

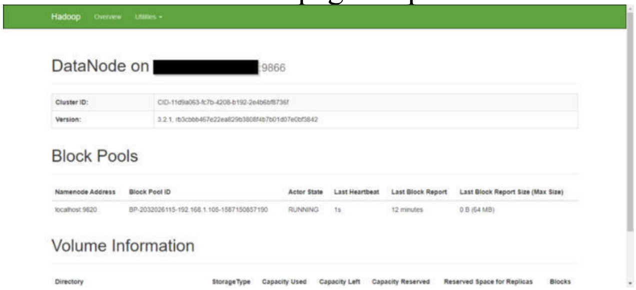
Security is off.

Safemode is off.

1 files and directories, 0 blocks (0 replicated blocks, 0 erasure coded block groups) = 1 total filesystem object(s).

Heap Memory used 63.28 MB of 191.5 MB Heap Memory. Max Heap Memory is 889 MB.

- Data node web page: http://localhost:9864/datanode.html



DataNode on [redacted]:9866

Cluster ID:	CID-11d9a063-fc7b-4208-b192-2e4b6bf8736f
Version:	3.2.1, rb3cbbb467e22ea829b3808f4b7b01d07e0bf3842

Block Pools

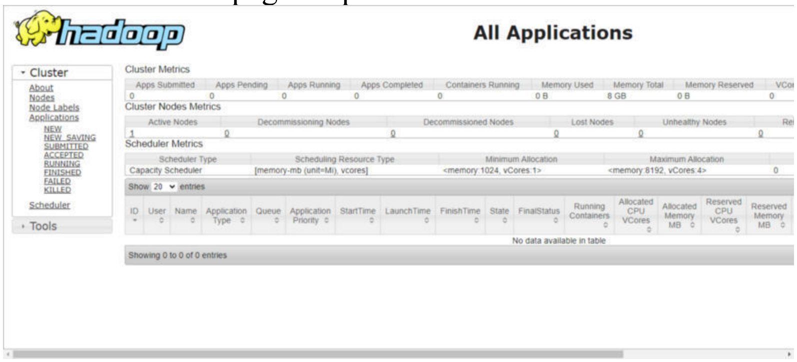
Namenode Address	Block Pool ID	Actor State	Last Heartbeat	Last Block Report	Last Block Report Size (Max Size)
localhost:9820	BP-2032026115-192.168.1.105-1587150857190	RUNNING	1s	12 minutes	0 B (64 MB)

Volume Information

Directory	StorageType	Capacity Used	Capacity Left	Capacity Reserved	Reserved Space for Replicas	Blocks

Figure 23 — Data node web page

- Yarn web page: http://localhost:8088/cluster



All Applications

Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved	VCo
0	0	0	0	0	0 B	8 GB	0 B	0

Cluster Nodes Metrics

Active Nodes	Decommissioning Nodes	Decommissioned Nodes	Lost Nodes	Unhealthy Nodes	Re
1	0	0	0	0	0

Scheduler Metrics

Scheduler Type	Scheduling Resource Type	Minimum Allocation	Maximum Allocation	
Capacity Scheduler	[memory-mb (unit=Mi), vcores]	<memory:1024, vCores:1>	<memory:8192, vCores:4>	0

Show 20 entries

ID	User	Name	Application Type	Queue	Application Priority	StartTime	LaunchTime	FinishTime	State	FinalStatus	Running Containers	Allocated CPU VCores	Allocated Memory MB	Reserved CPU VCores	Reserved Memory MB
No data available in table															

Showing 0 to 0 of 0 entries

10.Installation and deploying a PhP application on a Docker Container

Description:

Create a Machine Image of Ubuntu Bionic 18.04LTS or Xenial 16.04.

To install **Docker CE**, first, you need to remove older versions of **Docker** were called **docker**, **docker.io**, or **docker-engine** from the system using the following command.

```
$ sudo apt-get remove docker docker-engine docker.io containerd runc
```

Next, you need to set up the Docker repository to install and update Docker from the repository using following commands.

1. Update the apt package index

   ```
           $ sudo apt-get update
   ```

2. Install packages to allow apt to use a repository over HTTPS

   ```
   $ sudo apt-get install \
      apt-transport-https \
      ca-certificates \
      curl \
      gnupg-agent \
              software-properties-common
   ```

3. Add Docker's official GPG key

   ```
           $ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
   ```

4. Verify that you now have the key with the fingerprint 9DC8 5822 9FC7 DD38 854A E2D8 8D81 803C 0EBF CD88, by searching for the last 8 characters of the fingerprint

   ```
   sudo apt-key fingerprint 0EBFCD88

   pub   rsa4096 2017-02-22 [SCEA]
       9DC8 5822 9FC7 DD38 854A  E2D8 8D81 803C 0EBF CD88
   ```

```
uid           [ unknown] Docker Release (CE deb) <docker@docker.com>
sub   rsa4096 2017-02-22 [S]
```

5. Use the following command to set up the stable repository

   ```
   $ sudo add-apt-repository \
     "deb [arch=amd64] https://download.docker.com/linux/ubuntu \
   $(lsb_release -cs) \
     stable"
   ```

   The lsb_release -cs sub-command below returns the name of your Ubuntu distribution, such as xenial. Sometimes, in a distribution like Linux Mint, you might need to change $(lsb_release -cs) to your parent Ubuntu distribution. For example, if you are using Linux Mint Tessa, you could use bionic. Docker does not offer any guarantees on untested and unsupported Ubuntu distributions.

6. Update the apt package index and install the latest version of **Docker CE** using following commands.

   ```
   $ sudo apt-get update
   ```

7. Install the latest version of Docker Engine - Community and containerd, or go to the next step 8 to install a specific version

   ```
   $ sudo apt-get install docker-ce docker-ce-cli containerd.io
   ```

8. To install a specific version of Docker Engine - Community, list the available versions in the repo, then select and install: List the versions available in your repo:

   ```
   $ apt-cache madison docker-ce
   ```

9. Install a specific version using the version string from the second column, for example, 5:18.09.1~3-0~ubuntu-xenial

   ```
   $ sudo apt-get install docker-ce=<VERSION_STRING> docker-ce-
           cli=<VERSION_STRING> containerd.io
   ```

10. After successfully installing the **Docker CE** package, the service should be auto-started and auto-enabled to start at system boot, you can check its status using the following command.

    ```
    $ sudo systemctl status docker
    ```

11. Press CTRL C to exit

12. Verify that Docker Engine - Community is installed correctly by running the hello-world

image

```
$ sudo docker run hello-world
```

13. This command downloads a test image and runs it in a container. When the container runs, it prints the below informational message

```
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
1b930d010525: Pull complete
Digest:
        sha256:c3b4ada4687bbaa170745b3e4dd8ac3f194ca95b2d0518b417fb47e5879d9
        b5f
Status: Downloaded newer image for hello-world:latest
Hello from Docker!
This message shows that your installation appears to be working correctly.
To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
   (amd64)
 3. The Docker daemon created a new container from that image which runs the
   executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.
To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash
Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/
For more examples and ideas, visit:
https://docs.docker.com/get-started/
```

Dockerizing a Node.js web application

14. Create a new folder namely nodejsapp

15. Make a package.json file as follows

```
{
  "name": "docker_web_app",
  "version": "1.0.0",
  "description": "Node.js on Docker",
  "author": "Sashi’s First Nodejs Application on Container
           <sashi.mamidanna@gmail.com>",
  "main": "server.js",
  "scripts": {
    "start": "node server.js"
  },
  "dependencies": {
    "express": "^4.16.1"
  }
}
```

16. Then create a file server.js to create a program that runs on the node. The idea is to enable the server.js file to run on the container at port no 8081

```
'use strict';
const express = require('express');

// Constants
const PORT = 8081;
const HOST = '0.0.0.0';

// App
const app = express();
app.get('/', (req, res) => {
  res.send('Hello world\n');
});

app.listen(PORT, HOST);
```

```
console.log(`Running on http://${HOST}:${PORT}`);
```

17. Create a dockerfile now namely dockerfile in the same directory

```
$sudo nano dockerfile
```

18. Copy the source code into the dockerfile

```
FROM node:10

# Create app directory
WORKDIR /app
COPY . /app
RUN npm install

COPY . .

EXPOSE 8082
CMD [ "node", "server.js" ]
```

19. Now build the docker image with the node application on it

```
$sudo docker build -t nodejsapp .
```

20. Run the application by executing run command on docker

```
$sudo docker run -p 8082:8081 nodejsapp
```

21. The container engine will run the command node server.js that was initialized through the dockerfile. Now the server.js is listening to incoming requests on http://localhost:8081 on the host operating system. But the application is running on port number 8082 on the docker engine.

22. Open a new ssh connection on the same VM and run the command to send an outgoing request to the application running on docker

```
$sudo curl http://localhost:8082
Hello World
```

23. This response is a result of the application running on node, devoted on the docker container, that's running on Docker engine available on the Ubuntu OS.
24. Run the bow command to check if the docker image is present in the list of images on Docker C

    ```
    $sudo docker ps
    ```

25. To stop the docker container image

    ```
    $sudo docker stop <docker image ID>
    ```

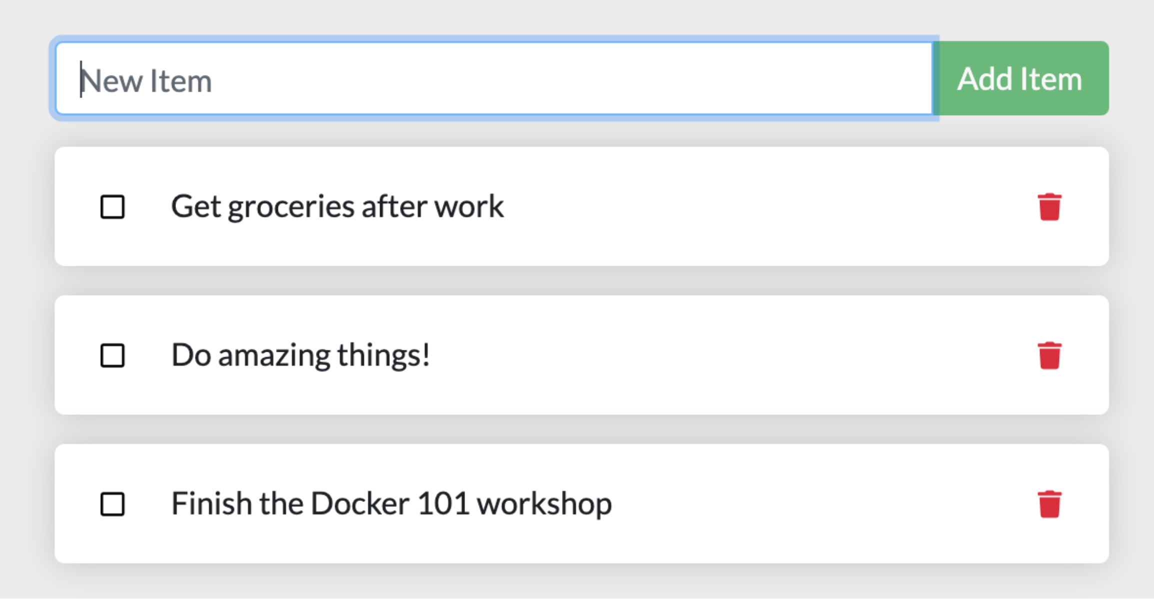
26. To remove the docker image
27. $sudo docker rmi <docker image ID>

12.Installation and deploying a node.js application on a Docker Container

1. Install Docker desktop and Git client

For the rest of this tutorial, we will be working with a simple todo list manager that is running in Node.js. If you're not familiar with Node.js, don't worry! No real JavaScript experience is needed!

At this point, your development team is quite small and you're simply building an app to prove out your MVP (minimum viable product). You want to show how it works and what it's capable of doing without needing to think about how it will work for a large team, multiple developers, etc.



□ Get groceries after work

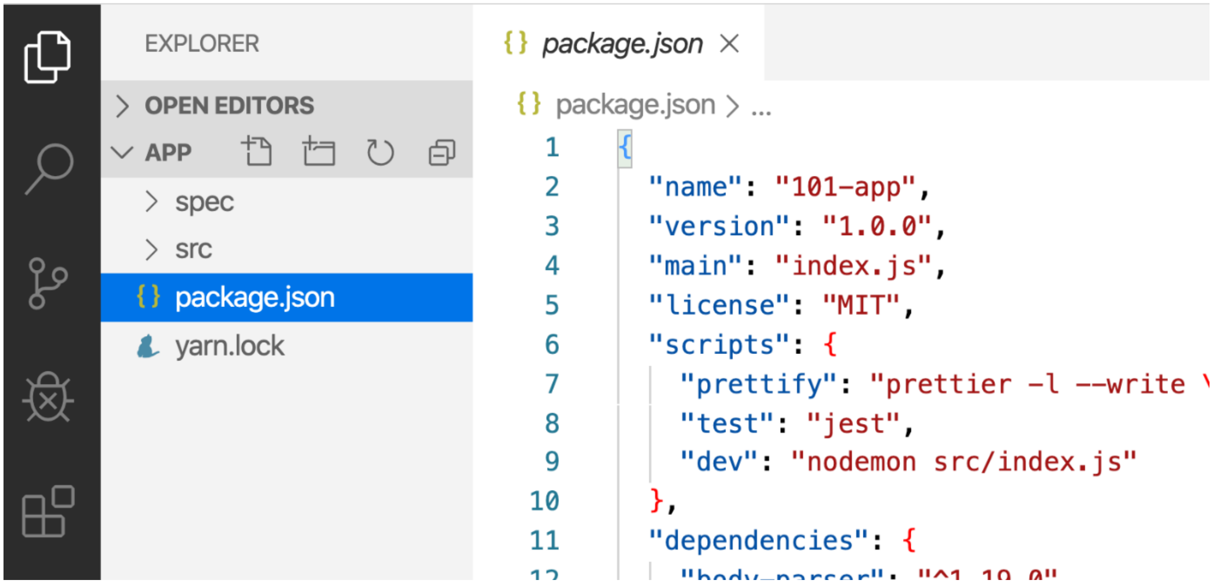
□ Do amazing things!

□ Finish the Docker 101 workshop

Getting our App¶

Before we can run the application, we need to get the application source code onto our machine. For real projects, you will typically clone the repo. But, for this tutorial, we have created a ZIP file containing the application.

1. Download the ZIP. Open the ZIP file and make sure you extract the contents.
2. Once extracted, use your favorite code editor to open the project. If you're in need of an editor, you can use Visual Studio Code. You should see the `package.json` and two subdirectories (`src` and `spec`).



```
{
  "name": "101-app",
  "version": "1.0.0",
  "main": "index.js",
  "license": "MIT",
  "scripts": {
    "prettify": "prettier -l --write \
    "test": "jest",
    "dev": "nodemon src/index.js"
  },
  "dependencies": {
    "body-parser": "^1.19.0"
```

Building the App's Container Image¶

In order to build the application, we need to use a `Dockerfile`. A Dockerfile is simply a text-based script of instructions that is used to create a container image. If you've created Dockerfiles before, you might see a few flaws in the Dockerfile below. But, don't worry! We'll go over them.

1. Create a file named `Dockerfile` in the same folder as the file `package.json` with the following contents.
2. `FROM node:18-alpine`
3. `WORKDIR /app`
4. `COPY . .`
5. `RUN yarn install --production`
6. `CMD ["node", "src/index.js"]`

   Please check that the file `Dockerfile` has no file extension like `.txt`. Some editors may append this file extension automatically and this would result in an error in the next step.
7. If you haven't already done so, open a terminal and go to the `app` directory with the `Dockerfile`. Now build the container image using the `docker build` command.
8. `docker build -t getting-started .`

   This command used the Dockerfile to build a new container image. You might have noticed that a lot of "layers" were downloaded. This is because we instructed the builder that we wanted to start from the `node:18-alpine` image. But, since we didn't have that on our machine, that image needed to be downloaded.

   After the image was downloaded, we copied in our application and used `yarn` to install our application's dependencies. The `CMD` directive specifies the default command to run when starting a container from this image.

   Finally, the `-t` flag tags our image. Think of this simply as a human-readable name for the final image. Since we named the image `getting-started`, we can refer to that image when we run a container.

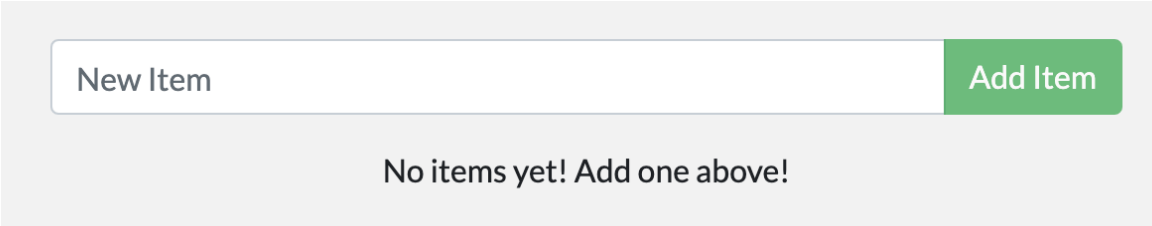
The `.` at the end of the `docker build` command tells that Docker should look for the `Dockerfile` in the current directory.

Starting an App Container¶

Now that we have an image, let's run the application! To do so, we will use the `docker run` command (remember that from earlier?).

1. Start your container using the `docker run` command and specify the name of the image we just created:
2. `docker run -dp 3000:3000 getting-started`

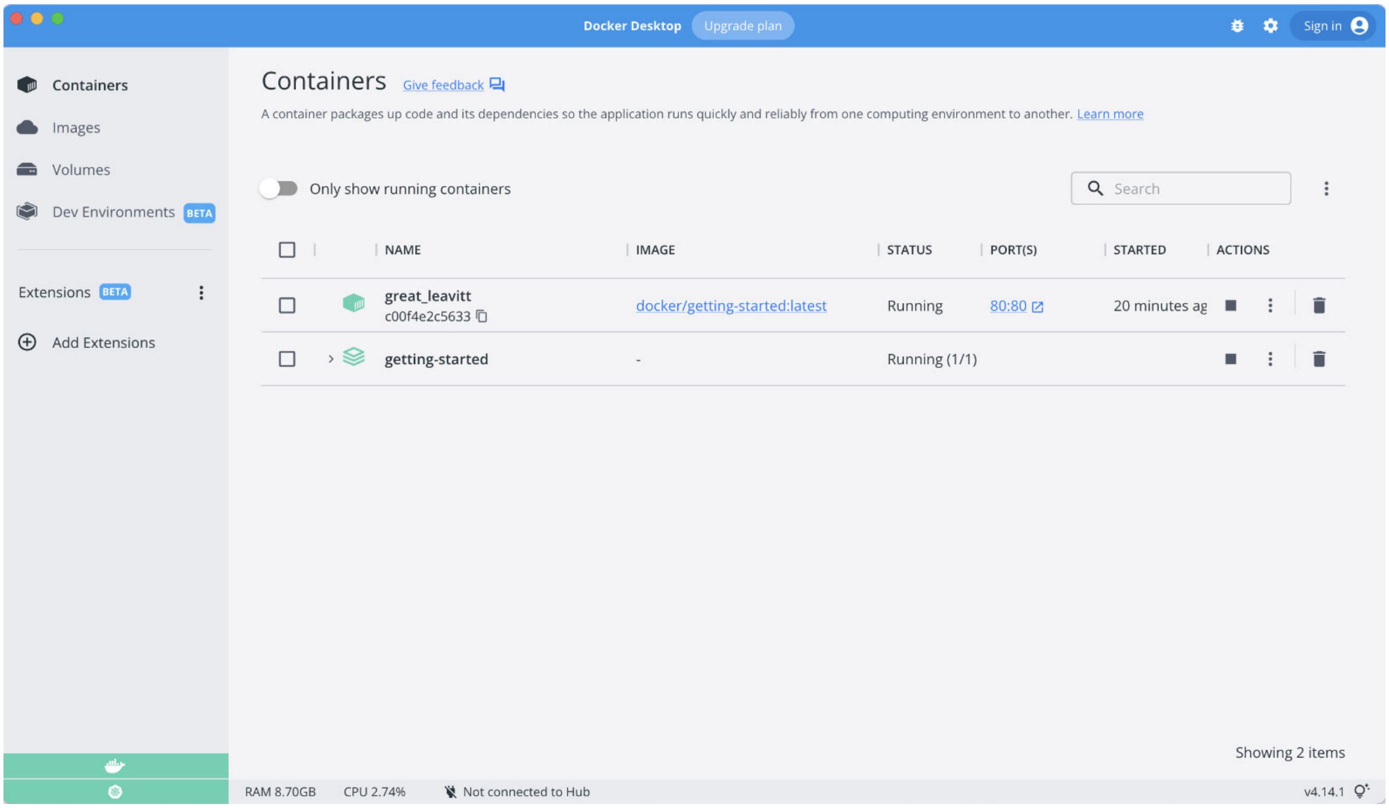
   Remember the `-d` and `-p` flags? We're running the new container in "detached" mode (in the background) and creating a mapping between the host's port 3000 to the container's port 3000. Without the port mapping, we wouldn't be able to access the application.
3. After a few seconds, open your web browser to http://localhost:3000. You should see our app!

   No items yet! Add one above!
4. Go ahead and add an item or two and see that it works as you expect. You can mark items as complete and remove items. Your frontend is successfully storing items in the backend! Pretty quick and easy, huh?

At this point, you should have a running todo list manager with a few items, all built by you! Now, let's make a few changes and learn about managing our containers.

If you take a quick look at the Docker Dashboard, you should see your two containers running now (this tutorial and your freshly launched app container)!



Containers Give feedback

A container packages up code and its dependencies so the application runs quickly and reliably from one computing environment to another. Learn more

Only show running containers

☐	NAME	IMAGE	STATUS	PORT(S)	STARTED	ACTIONS
☐	great_leavitt c00f4e2c5633	docker/getting-started:latest	Running	80:80	20 minutes ag	
☐	getting-started	-	Running (1/1)			

Showing 2 items

RAM 8.70GB CPU 2.74% Not connected to Hub v4.14.1

Recap¶

In this short section, we learned the very basics about building a container image and created a Dockerfile to do so. Once we built an image, we started the container and saw the running app!