Steps to install Hadoop:

1. Make sure java is installed.

   ```
   java -version
   ```

   If java is not installed, then type in the following commands:

   ```
   sudo apt-get install update
   sudo apt-get update
   sudo apt-get install default-jdk
   ```

   Make sure now java is installed.

   ```
   java -version
   ```

```
husseinfadl@husseinfadl:/home$ java -version
openjdk version "11.0.10" 2021-01-19
OpenJDK Runtime Environment (build 11.0.10+9-Ubuntu-0ubuntu1.18.04)
OpenJDK 64-Bit Server VM (build 11.0.10+9-Ubuntu-0ubuntu1.18.04, mixed mode, sharing)
husseinfadl@husseinfadl:/home$
```

2. Install ssh server

   ```
   sudo apt-get install ssh-server
   ```

   Generate public/private RSA key pair.

   ```
   ssh-keygen -t rsa -P ""
   ```

   When prompted for the file name to save the key, press Enter (leave it blank).

```
husseinfadl@husseinfadl:/home$ ssh-keygen -t rsa -P ""
Generating public/private rsa key pair.
Enter file in which to save the key (/home/husseinfadl/.ssh/id_rsa):
Your identification has been saved in /home/husseinfadl/.ssh/id_rsa.
Your public key has been saved in /home/husseinfadl/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:XqVfPkZlmsIG7z9I1pX/zMJZo2hWX7K9HtcgjIjLEyQ husseinfadl@husseinfadl
The key's randomart image is:
+---[RSA 2048]----+
|                 |
|                 |
|    E .   . .   +|
|     o . . O   *.|
|      o S + B.*..|
|     . + . +oBoo=|
|      + .  o=o*X*|
|       .   +.+*oB|
|          o   o=.|
+----[SHA256]-----+
```

   Type the following commands:

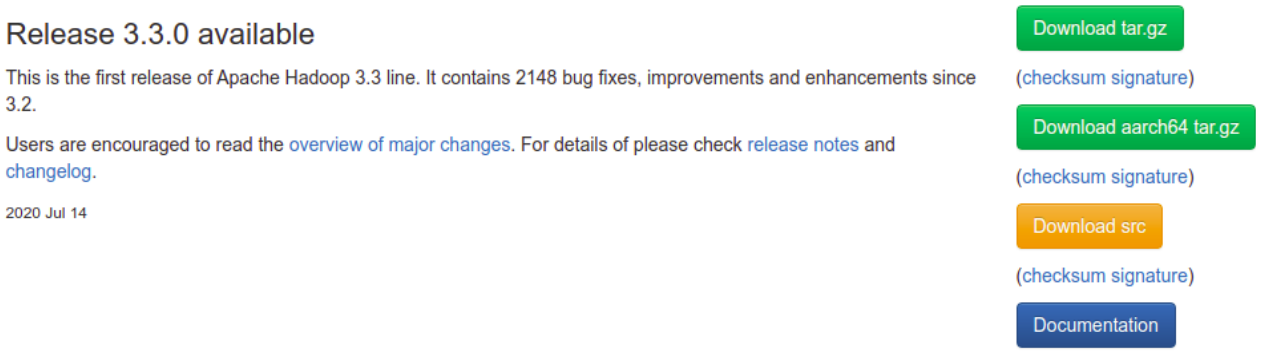
   ```
   cat $HOME/.ssh/id_rsa.pub >> $HOME/.ssh/authorized_keys
   ```

```
ssh localhost
exit
```

```
husseinfadl@husseinfadl:~$ ssh localhost
Welcome to Ubuntu 18.04.5 LTS (GNU/Linux 4.15.0-139-generic x86_64)

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

 * Canonical Livepatch is available for installation.
   - Reduce system reboots and improve kernel security. Activate at:
     https://ubuntu.com/livepatch

80 packages can be updated.
27 of these updates are security updates.
To see these additional updates run: apt list --upgradable

New release '20.04.2 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Tue Apr 13 15:29:36 2021 from 127.0.0.1
husseinfadl@husseinfadl:~$
```

```
husseinfadl@husseinfadl:~$ exit
logout
Connection to localhost closed.
husseinfadl@husseinfadl:/home$
```

3. Install Hadoop by navigating to the following link and downloading the tar.gz file for Hadoop version 3.3.0 (or a later version if you wish). (478 MB)
   https://hadoop.apache.org/release/3.3.0.html



4. Once downloaded, open the terminal and cd to the directory where it is downloaded (assume the desktop for example) and extract it as follows:

```
cd Desktop
```

```
sudo tar -xvzf hadoop-3.3.0.tar.gz
```

You can now check that there is an extracted file named hadoop-3.3.0 by typing the command “ls” or by visually inspecting the files.

5. Now, we move the extracted file to the location /usr/local/hadoop

```
sudo mv hadoop-3.3.0 /usr/local/hadoop
```

6. Let’s configure the hadoop system.

   Type the following command:

```
sudo gedit ~/.bashrc
```

At the end of the file, add the following lines: (Note: Replace the java version with the version number you already have. You can navigate to the directory /usr/lib/jvm and check the file name java-xx-openjdk-amd64)

export JAVA_HOME=/usr/lib/jvm/java-**11**-openjdk-amd64

export HADOOP_HOME=/usr/local/hadoop

export PATH=$PATH:$HADOOP_HOME/bin

export PATH=$PATH:$HADOOP_HOME/sbin

export HADOOP_MAPRED_HOME=$HADOOP_HOME

export HADOOP_COMMON_HOME=$HADOOP_HOME

export HADOOP_HDFS_HOME=$HADOOP_HOME

export YARN_HOME=$HADOOP_HOME

export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/native

export HADOOP_OPTS="-Djava.library.path=$HADOOP_HOME/native"

```
# enable programmable completion features (you don't need to enable
# this, if it's already enabled in /etc/bash.bashrc and /etc/profile
# sources /etc/bash.bashrc).
if ! shopt -oq posix; then
  if [ -f /usr/share/bash-completion/bash_completion ]; then
    . /usr/share/bash-completion/bash_completion
  elif [ -f /etc/bash_completion ]; then
    . /etc/bash_completion
  fi
fi

export JAVA_HOME=/usr/lib/jvm/java-11-openjdk-amd64
export HADOOP_HOME=/usr/local/hadoop
export PATH=$PATH:$HADOOP_HOME/bin
export PATH=$PATH:$HADOOP_HOME/sbin
export HADOOP_MAPRED_HOME=$HADOOP_HOME
export HADOOP_COMMON_HOME=$HADOOP_HOME
export HADOOP_HDFS_HOME=$HADOOP_HOME
export YARN_HOME=$HADOOP_HOME
export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/native
export HADOOP_OPTS="-Djava.library.path=$HADOOP_HOME/native"
```

7. Save the file and close it.
8. Now from the terminal, type the following command:

   ```
   source ~/.bashrc
   ```

9. We start configuring Hadoop by opening **hadoop-env.sh** as follows:

   ```
   sudo gedit /usr/local/hadoop/etc/hadoop/hadoop-env.sh
   ```

   Search for the line starting with **export JAVA_HOME=** and replace it with the following line.

   ```
   export JAVA_HOME=/usr/lib/jvm/java-11-openjdk-amd64
   ```

   Save the file by clicking on “Save” or (Ctrl+S)

```
# The java implementation to use. By default, this environment
# variable is REQUIRED on ALL platforms except OS X!
export JAVA_HOME=/usr/lib/jvm/java-11-openjdk-amd64

# Location of Hadoop.  By default, Hadoop will attempt to determine
# this location based upon its execution path.
# export HADOOP_HOME=

# Location of Hadoop's configuration information.  i.e., where this
# file is living. If this is not defined, Hadoop will attempt to
# locate it based upon its execution path.
#
```

10. Open **core-site.xml** as follows:

```
sudo gedit /usr/local/hadoop/etc/hadoop/core-site.xml
```

Add the following lines between the tags <configuration> and </configuration> and save it (Ctrl+S).

```
<property>
   <name>fs.default.name</name>
   <value>hdfs://localhost:9000</value>
</property>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<!--
  Licensed under the Apache License, Version 2.0 (the "License");
  you may not use this file except in compliance with the License.
  You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software
  distributed under the License is distributed on an "AS IS" BASIS,
  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
  See the License for the specific language governing permissions and
  limitations under the License. See accompanying LICENSE file.
-->

<!-- Put site-specific property overrides in this file. -->

<configuration>
 <property>
   <name>fs.default.name</name>
   <value>hdfs://localhost:9000</value>
 </property>
</configuration>
```

11. Open **hdfs-site.xml** as follows:

```
sudo gedit /usr/local/hadoop/etc/hadoop/hdfs-site.xml
```

Add the following lines between the tags <configuration> and </configuration> and save it (Ctrl+S).

```
<property>
  <name>dfs.replication</name>
  <value>1</value>
</property>
```

```
  <property>
   <name>dfs.namenode.name.dir</name>
   <value>file:/usr/local/hadoop_space/hdfs/namenode</value>
  </property>
  <property>
   <name>dfs.datanode.data.dir</name>
   <value>file:/usr/local/hadoop_space/hdfs/datanode</value>
  </property>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<!--
  Licensed under the Apache License, Version 2.0 (the "License");
  you may not use this file except in compliance with the License.
  You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software
  distributed under the License is distributed on an "AS IS" BASIS,
  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
  See the License for the specific language governing permissions and
  limitations under the License. See accompanying LICENSE file.
-->

<!-- Put site-specific property overrides in this file. -->

<configuration>
  <property>
    <name>dfs.replication</name>
    <value>1</value>
  </property>
  <property>
    <name>dfs.namenode.name.dir</name>
    <value>file:/usr/local/hadoop_tmp/hdfs/namenode</value>
  </property>
  <property>
    <name>dfs.datanode.data.dir</name>
    <value>file:/usr/local/hadoop_tmp/hdfs/datanode</value>
  </property>
</configuration>
```

12. Open **yarn-site.xml** as follows:

```
sudo gedit /usr/local/hadoop/etc/hadoop/yarn-site.xml
```

Add the following lines between the tags <configuration> and </configuration> and save it (Ctrl+S)

```
<property>
  <name>yarn.nodemanager.aux-services</name>
  <value>mapreduce_shuffle</value>
</property>
<property>
  <name>yarn.nodemanager.aux-services.mapreduce.shuffle.class</name>
  <value>org.apache.hadoop.mapred.ShuffleHandler</value>
```

```
 </property>
```

```
<?xml version="1.0"?>
<!--
  Licensed under the Apache License, Version 2.0 (the "License");
  you may not use this file except in compliance with the License.
  You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software
  distributed under the License is distributed on an "AS IS" BASIS,
  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
  See the License for the specific language governing permissions and
  limitations under the License. See accompanying LICENSE file.
-->

<configuration>
<!-- Site specific YARN configuration properties -->
 <property>
   <name>yarn.nodemanager.aux-services</name>
   <value>mapreduce_shuffle</value>
 </property>
 <property>
   <name>yarn.nodemanager.aux-services.mapreduce.shuffle.class</name>
   <value>org.apache.hadoop.mapred.ShuffleHandler</value>
 </property>
</configuration>
```

13. Open **mapred-site.xml** as follows:

```
sudo gedit /usr/local/hadoop/etc/hadoop/mapred-site.xml
```

Add the following lines between the tags <configuration> and </configuration> and save it (Ctrl+S)

```
 <property>
  <name>mapreduce.framework.name</name>
  <value>yarn</value>
 </property>
 <property>
  <name>yarn.app.mapreduce.am.env</name>
```

```
  <value>HADOOP_MAPRED_HOME=${HADOOP_HOME}</value>
 </property>
 <property>
  <name>mapreduce.map.env</name>
  <value>HADOOP_MAPRED_HOME=${HADOOP_HOME}</value>
 </property>
 <property>
  <name>mapreduce.reduce.env</name>
  <value>HADOOP_MAPRED_HOME=${HADOOP_HOME}</value>
 </property>


  Unless required by applicable law or agreed to in writing, software
  distributed under the License is distributed on an "AS IS" BASIS,
  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
  See the License for the specific language governing permissions and
  limitations under the License. See accompanying LICENSE file.
-->

<!-- Put site-specific property overrides in this file. -->

<configuration>
 <property>
  <name>mapreduce.framework.name</name>
  <value>yarn</value>
 </property>
 <property>
  <name>yarn.app.mapreduce.am.env</name>
  <value>HADOOP_MAPRED_HOME=${HADOOP_HOME}</value>
 </property>
 <property>
  <name>mapreduce.map.env</name>
  <value>HADOOP_MAPRED_HOME=${HADOOP_HOME}</value>
 </property>
 <property>
  <name>mapreduce.reduce.env</name>
  <value>HADOOP_MAPRED_HOME=${HADOOP_HOME}</value>
 </property>
</configuration>
```

14. Now, run the following commands on the terminal to create a directory for hadoop space, name node and data node.

```
sudo mkdir -p /usr/local/hadoop_space
sudo mkdir -p /usr/local/hadoop_space/hdfs/namenode
sudo mkdir -p /usr/local/hadoop_space/hdfs/datanode
```

Now we have successfully installed Hadoop.

15. Format the namenode as follows:

```
hdfs namenode -format
```

```
husseinfadl@husseinfadl:~$ hdfs namenode -format
2021-04-13 14:15:15,047 INFO namenode.NameNode: STARTUP_MSG:
/************************************************************
STARTUP_MSG: Starting NameNode
STARTUP_MSG:   host = husseinfadl/127.0.1.1
STARTUP_MSG:   args = [-format]
STARTUP_MSG:   version = 3.3.0
STARTUP_MSG:   classpath = /usr/local/hadoop/etc/hadoop:/usr/local/hadoop/share/hadoop/commo
n/lib/curator-recipes-4.2.0.jar:/usr/local/hadoop/share/hadoop/common/lib/failureaccess-1.0.
jar:/usr/local/hadoop/share/hadoop/common/lib/jackson-annotations-2.10.3.jar:/usr/local/hado
op/share/hadoop/common/lib/jersey-json-1.19.jar:/usr/local/hadoop/share/hadoop/common/lib/ke
rb-core-1.0.1.jar:/usr/local/hadoop/share/hadoop/common/lib/jackson-core-asl-1.9.13.jar:/usr
/local/hadoop/share/hadoop/common/lib/slf4j-log4j12-1.7.25.jar:/usr/local/hadoop/share/hadoo
p/common/lib/metrics-core-3.2.4.jar:/usr/local/hadoop/share/hadoop/common/lib/kerby-xdr-1.0.
1.jar:/usr/local/hadoop/share/hadoop/common/lib/j2objc-annotations-1.1.jar:/usr/local/hadoop
/share/hadoop/common/lib/javax.servlet-api-3.1.0.jar:/usr/local/hadoop/share/hadoop/common/l
ib/curator-client-4.2.0.jar:/usr/local/hadoop/share/hadoop/common/lib/kerb-server-1.0.1.jar:
/usr/local/hadoop/share/hadoop/common/lib/zookeeper-jute-3.5.6.jar:/usr/local/hadoop/share/h
adoop/common/lib/slf4j-api-1.7.25.jar:/usr/local/hadoop/share/hadoop/common/lib/curator-fram
```

This step should end by shutting down the namenode as follows:

```
2021-04-13 14:16:46,808 INFO namenode.FSImageFormatProtobuf: Image file /usr/local/hadoop_tm
p/hdfs/namenode/current/fsimage.ckpt_0000000000000000000 of size 406 bytes saved in 0 second
s .
2021-04-13 14:16:46,900 INFO namenode.NNStorageRetentionManager: Going to retain 1 images wi
th txid >= 0
2021-04-13 14:16:46,909 INFO namenode.FSImage: FSImageSaver clean checkpoint: txid=0 when me
et shutdown.
2021-04-13 14:16:46,909 INFO namenode.NameNode: SHUTDOWN_MSG:
/************************************************************
SHUTDOWN_MSG: Shutting down NameNode at husseinfadl/127.0.1.1
************************************************************/
```

16. Before starting the Hadoop Distributed File System (hdfs), we need to make sure that the rcmd type is “ssh” not “rsh” when we type the following command

```
pdsh -q -w localhost
```

```
husseinfadl@husseinfadl:~$ pdsh -q -w localhost
-- DSH-specific options --
Separate stderr/stdout  Yes
Path prepended to cmd   none
Appended to cmd         none
Command:                none
Full program pathname   /usr/bin/pdsh
Remote program path     /usr/bin/pdsh

-- Generic options --
Local username          husseinfadl
Local uid               1000
Remote username         husseinfadl
Rcmd type               rsh
One ^C will kill pdsh   No
Connect timeout (secs)  10
Command timeout (secs)  0
Fanout                  32
Display hostname labels Yes
Debugging               No

-- Target nodes --
localhost
```

17. If the rcmd type is “rsh” as in the above figure, type the following commands:

```
export PDSH_RCMD_TYPE=ssh
cat $HOME/.ssh/id_rsa.pub >> $HOME/.ssh/authorized_keys
chmod 0600 ~/.ssh/authorized_keys
```

Run Step 16 again to check that the rcmd type is now ssh.

If not, skip that step.

18. Start the HDFS System using the command.

```
start-dfs.sh
```

```
husseinfadl@husseinfadl:~$ start-dfs.sh
Starting namenodes on [localhost]
Starting datanodes
Starting secondary namenodes [husseinfadl]
2021-04-13 14:48:34,832 WARN util.NativeCodeLoader: Unable to load nat
ive-hadoop library for your platform... using builtin-java classes whe
re applicable
```

19. Start the YARN using the command

```
start-yarn.sh
```

```
husseinfadl@husseinfadl:~$ start-yarn.sh
Starting resourcemanager
Starting nodemanagers
```

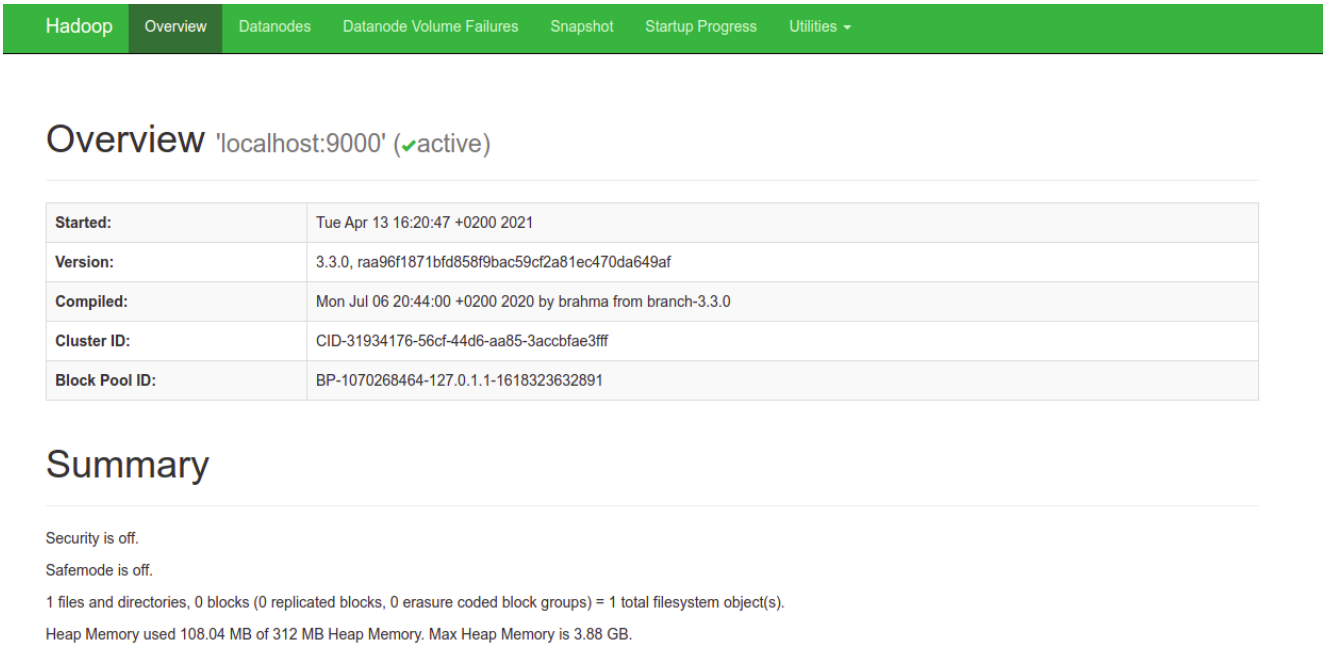
20. Type the following command. You should see an output similar to the one in the following figure.

```
jps
```

```
husseinfadl@husseinfadl:~$ jps
8484 ResourceManager
7719 NameNode
8680 NodeManager
8201 SecondaryNameNode
9517 Jps
9263 DataNode
```

Make sure these nodes are listed: (ResourceManager, NameNode, NodeManager, SecondaryNameNode, Jps and DataNode).

21. Go to localhost:9870 from the browser. You should expect the following



Steps to run WordCount Program on Hadoop:

1. Make sure Hadoop and Java are installed properly
   ```
   hadoop version
   javac -version
   ```

2. Create a directory on the Desktop named Lab and inside it create two folders; one called “Input” and the other called “tutorial_classes”.
   [You can do this step using GUI normally or through terminal commands]
   ```
   cd Desktop
   mkdir Lab
   mkdir Lab/Input
   mkdir Lab/tutorial_classes
   ```

3. Add the file attached with this document “WordCount.java” in the directory Lab

4. Add the file attached with this document “input.txt” in the directory Lab/Input.

5. Type the following command to export the hadoop classpath into bash.
   ```
   export HADOOP_CLASSPATH=$(hadoop classpath)
   ```
   Make sure it is now exported.
   ```
   echo $HADOOP_CLASSPATH
   ```
6. It is time to create these directories on HDFS rather than locally. Type the following commands.
   ```
   hadoop fs -mkdir /WordCountTutorial
   hadoop fs -mkdir /WordCountTutorial/Input
   hadoop fs -put Lab/Input/input.txt /WordCountTutorial/Input
   ```
7. Go to localhost:9870 from the browser, Open “Utilities → Browse File System”
   and you should see the directories and files we placed in the file system.
8. Then, back to local machine where we will compile the WordCount.java file. Assuming we are currently in the Desktop directory.
   ```
   cd Lab
   javac -classpath $HADOOP_CLASSPATH -d tutorial_classes WordCount.java
   ```
9. Put the output files in one jar file (There is a dot at the end)
   ```
   jar -cvf WordCount.jar -C tutorial_classes .
   ```
10. Now, we run the jar file on Hadoop.

```
hadoop jar WordCount.jar WordCount /WordCountTutorial/Input
/WordCountTutorial/Output
```

11. Output the result:

```
hadoop dfs -cat /WordCountTutorial/Output/*
```

Requirement:

Vodafone Egypt is launching a marketing campaign in Ramadan to promote their sales and increase their profit from selling the prepaid recharge cards. These cards are worth 5, 10, 15, 50, and 100 EGP.

The data science team at Vodafone are analyzing the customers' data which include the customer personal information, the prepaid card they purchased, the timestamp they registered the prepaid amount on their Vodafone accounts, among other information.

The details of the customers are omitted, and you are only provided with a file "in.csv" which includes two columns.

1. Customer ID. (Each ID maps to a certain customer, whose data is hidden for confidentiality).
2. Prepaid Card Amount.

Your task is to generate a report using MapReduce (similar to the WordCount program) showing the total amount of prepaid cards for each customer that they have purchased. For example, if a customer with ID 300 purchased 5 cards with 10, 15, 15, 10, 100, then the report should include that customer ID 300 bought cards with a total amount of 150.

Disclaimer: Thanks to Vodafone DS team who provided us with this real customer data.