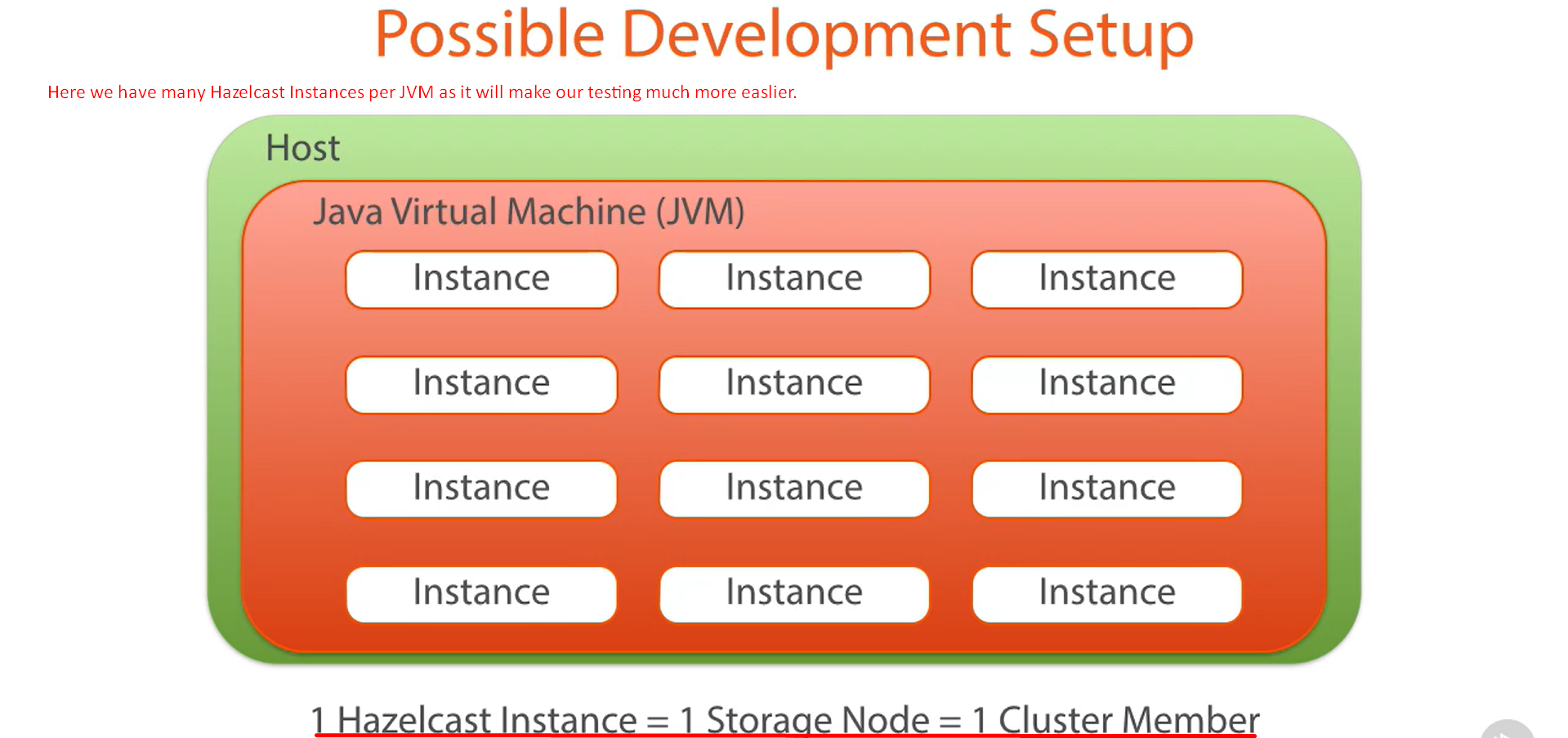
1. Now we have some data in our cluster.
2. Let’s see how resilient that data is.
3. Diagram

   Description automatically generated  
   Diagram

   Description automatically generated
4. Every **Hazelcast Instance** that is a member of a cluster, becomes a **Storage Node** for data.
5. That data becomes more **resilient** if you have **Storage Nodes** located on different machines in different **racks** in a **Data Center**.  
   It doesn’t matter if you have one **Hazelcast instance** (**Storage Node**) per JVM or many **Hazelcast instances** per JVM.
6. In Production, probably you use one Hazelcast instance per JVM as shown in the following snapshot where one **Hazelcast Instance per JVM**.  
   Graphical user interface

   Description automatically generated  
   However, running multiple Hazelcast instances (**Storage Nodes**) in a single JVM actually provides us a way to do some testing fairly easily & that is what we will generally be doing on this course.  
   
7. So, I revamped my test a little.  
   So, rather than using separate JVMs for my **Storage Nodes** (**Hazelcast instances**), I run multiple **Storage Nodes** (**Hazelcast Instances**) inside a single JVM.  
   I can therefore easily access them through code & run some test scenarios to mimic storage nodes coming and going from the cluster.
8. I have created this StorageNodeFactory.java.
9. The **ensureClusterSize**(int size) method allows me to spin up and shut down **cluster node members** (**Storage Node = Hazelcast Instance**) until the desired luster size is met.  
   Text

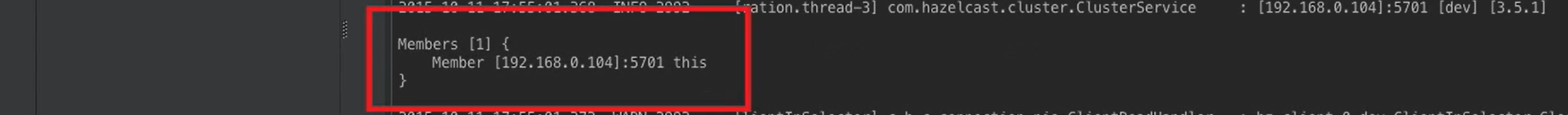
   Description automatically generated  
   Text

   Description automatically generated  
   When shutting the cluster node down however, I do it **synchronously**. It is more controlled and reduces the chances of data loss as if primary node and backup node are shut down at the same time, then it is possible to lose data from the cluster so shutting down **synchronously**.  
   Text

   Description automatically generated  
   In the above snapshot, we’re calling new CreateHazelcastInstance(latch, config). See the following snapshot for complete code.  
   Graphical user interface, application

   Description automatically generated
10. Back to **CustomerServiceTest.java**.
    1. We first spin a cluster with 4 storage nodes (members).
    2. Text

       Description automatically generated
    3. Graphical user interface, text

       Description automatically generated
    4. All of our test asserts are passed.
    5. Now you can see our cluster members are shutting down leaving only one running.   
       But our tests pass confirming that even after shutting down most of the cluster members we still have all of our data intact.  
       Obviously, there is a limit to this. If we have a large amount of data, you can’t store all of the data in a single JVM (In this test, we’re running multiple Storage Nodes under a single JVM).  
       So, this is for demonstration purpose only but it proves a point that Hazelcast automatically ensures that our data stays intact to the best of its ability.