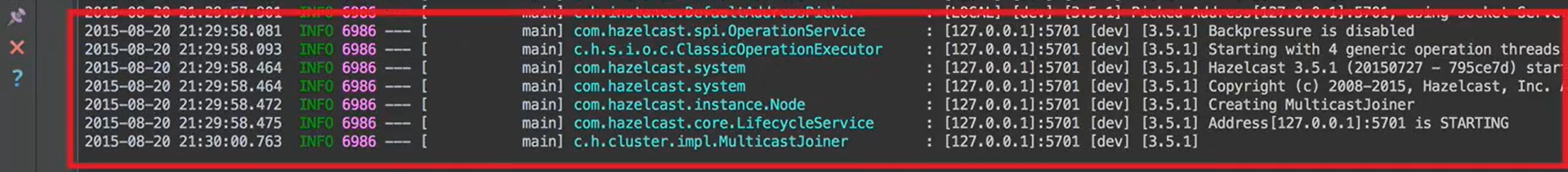
1. To work with **Hazelcast**, the first thing we need to do is actually to have a **cluster** in which to store the data.
2. We also want **a client** that can talk to the **cluster** but not become a member of the cluster.
3. So, let’s do that now.
4. We’re using Intellij IDE.
5. We have three modules inside of our project. All of which are based on **Spring Boot Application**.
   1. Text

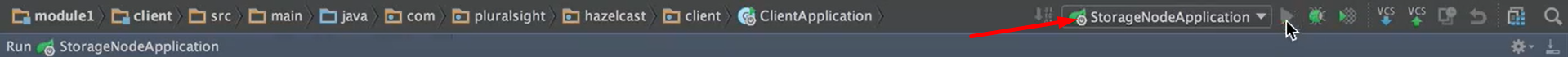
      Description automatically generated
   2. **Storage Node Module**:
      1. Jatin: From the phrase “**Storage Node**” in the **title**, you can understand that it will be part of the **Hazelcast Cluster**. So, this app will be providing the **memory** (storage) & **CPU** from the system on which it will be running.
      2. Small, Standalone Java App that will act as **Storage Node** for the **cluster**.  
         We can start as many of these as we want.
   3. **Client Module**:
      1. The intention of this module is to act **as a client to the Hazelcast Cluster**.
      2. We will be primarily using this to show how your app may use the **Hazelcast Cluster**.
   4. **Shared Module**:
      1. **Hazelcast** needs access to the classes that represent our stored information & tasks.  
         These are shared b/w the **Storage Node** (member) & the **Client** of the **Hazelcast Cluster**.  
         Therefore we will be using this module as a library for those two modules (**Storage Node Module & Client Module**).
6. **Dependency**:
   1. Text

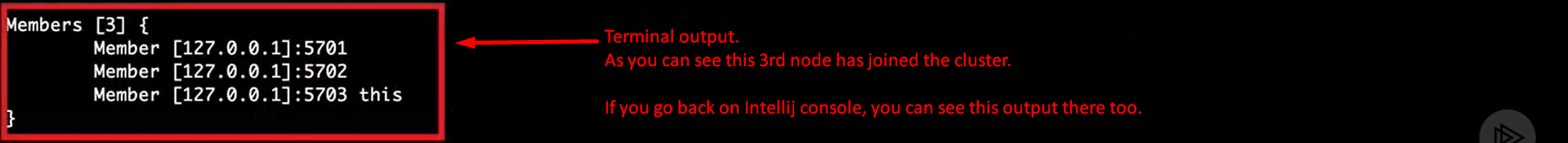
      Description automatically generated This JAR file contains all of the Hazelcast code in a single JAR file.  
      These component parts are also available in separate JAR files.  
      But that single JAR file is easiest way to go.
   2. **Let’s look at the storage-node module**.  
      Graphical user interface, website

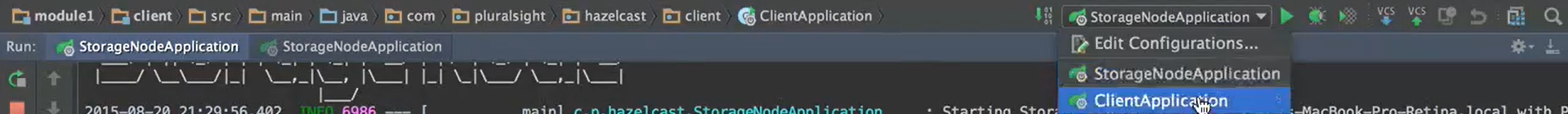
      Description automatically generated
      1. We have defined a bean. This is how to start a Storage Node for Hazelcast. As you can see it is extremely trivial.
      2. Hazelcast uses reasonable **defaults** which work for in the majority of environments.
      3. Note the destroy method on this bean definition 🡺 @Bean(**destroyMethod** = **“shutdown”**)  
         Spring will call shutdown() method on the created Hazelcast instance when it detects a CLTR + C has been pressed.  
         This will ensure a clean shutdown of the cluster node.  
         **NOTE**: if the return bean has **public shutdown() or public close() no argument method, then before destroying the bean, this method will be called automatically. So, no need for “destroymethod=”shutdown” as arg to**  @Bean.  
         Graphical user interface, text, application, email

         Description automatically generated
   3. Nothing in the **shared module**.
   4. Let’s take a look at the **Client module:**
      1. Graphical user interface, website

         Description automatically generated**NOTE**: Even **HazelcastClient.newHazelcastClient()** returns HazelcastInstance as **Hazelcast.newHazelcastInstance()** returns HazelcastInstance.  
         So from API perspective, once you have access to this Hazelcast instance, you don’t really know or care if you’re dealing with a full cluster member or a client. So, this makes the coding much simpler.
7. **Let’s start our cluster up**.
   1. First off, let’s start one storage node.
   2. 
   3. **Hazelcast Log**:  
      
      1. Here we can see **Hazelcast** is trying to form a **cluster**.
      2. It will try to connect to any existing **cluster** matching the **configuration** which there isn’t any in this case. So, it starts up a brand new **cluster** and outputs the log info about the **cluster** it has created.  
         In this case, it has one member with the IP address defined  
         Text

         Description automatically generated
   4. Let’s start another **cluster node** (**Storage Node**) to see it joining the **cluster**.
      1. 
      2. Graphical user interface, text, application

         Description automatically generated
      3. Just to show you that it has nothing to do with IntelliJ.   
         We can start a new code with command on CMD.  
           
           
         Text

         Description automatically generated  
         If you don’t see a member in the list, it means something didn’t go quite right & you need to investigate that.
   5. **Let’s run a Hazelcast client**.
      1. 
      2. The process is pretty much same & the output is slightly different.
      3. It has detected that it is connected to the cluster & outputs the cluster members but as it has not joined as a full member of the cluster. So, a port is not registered.   
         Graphical user interface, text, application

         Description automatically generated