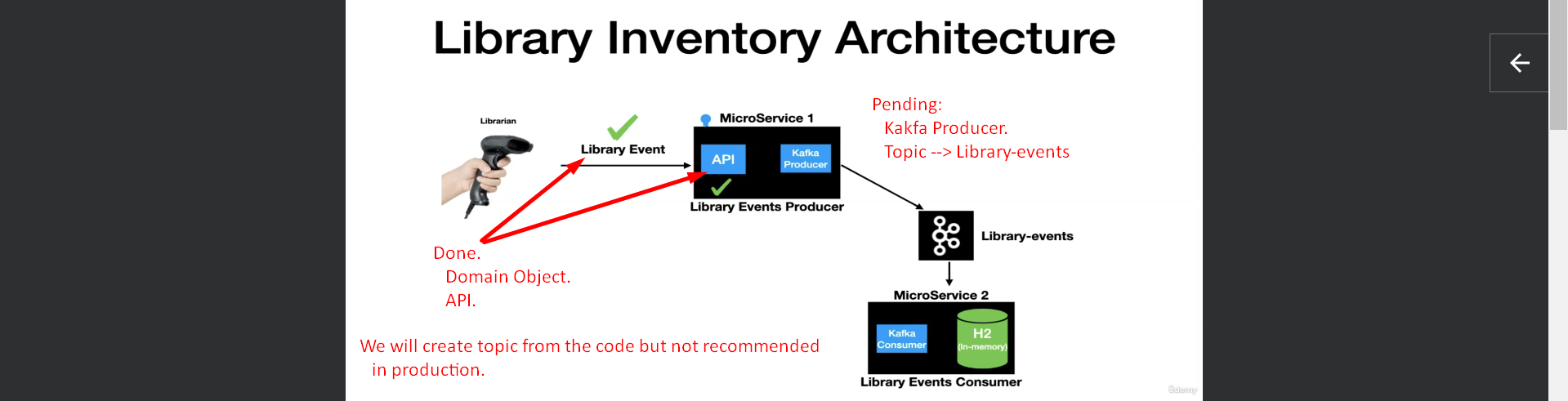
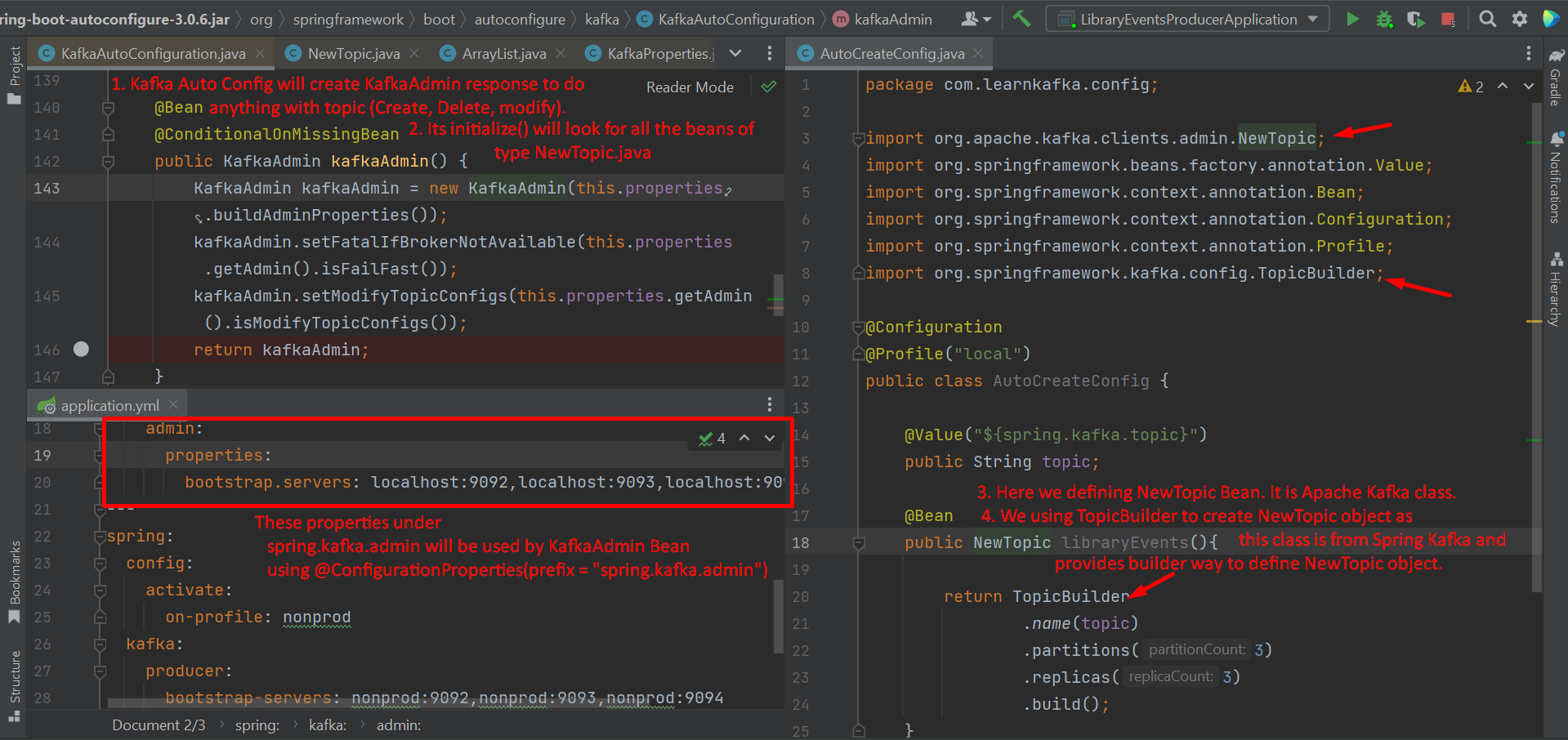
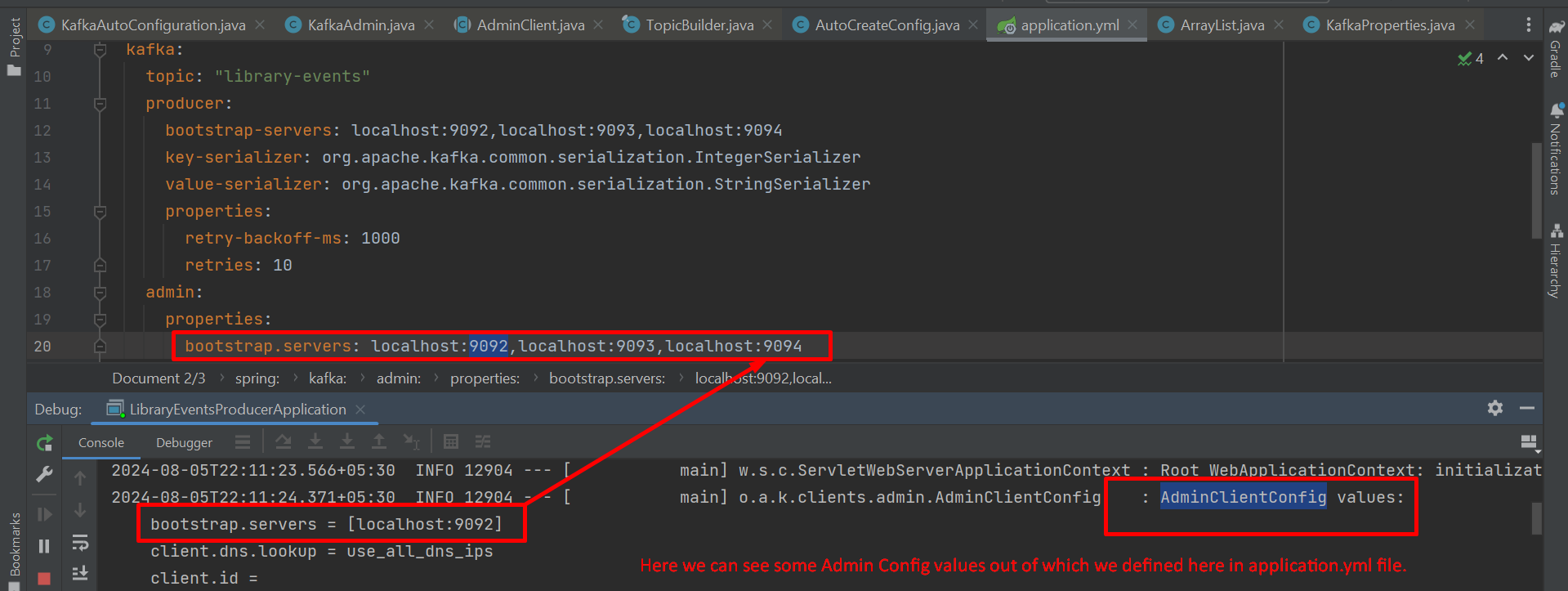
1. We will look at the current state of the app and what components we have created so far.
2. We did the following:
   1. **Domain Object** 🡺 Library Event.
   2. API is also created.
3. **Pending**;
   1. KafkaProducer.
   2. Library Event.
4. 
5. Topic Creation from Code.   
   **NOTE**: Class **NewTopic.java** comes from Apache Kafka not from Spring Kafka.  
   
6. Basically, the following classes will play role in creating a topic.
   1. **KafkaAdmin**: From Spring Kafka and will delegates requests to **org.apache.kafka.clients.admin.AdminClient**.
   2. **NewTopic**: From Apache. **org.apache.kafka.clients.admin.NewTopic**.
   3. **TopicBuilder**: From Spring Kafka. **org.springframework.kafka.config.TopicBuilder.**
7. 
   1. The above slide is very simple.
   2. First we register bean for **org.apache.kafka.clients.admin.NewTopic** object using builder class **org.springframework.kafka.config.TopicBuilder.**
   3. By default, Spring Auto Config KakfaAutoConfiguration.java will create **KakfaAdmin** bean which will use NewTopic bean and will delegate the request for topic creation to **org.apache.kafka.clients.admin.AdminClient** and **KakfaAdmin** also passes on bootstrap servers as it will read from application.yaml file under **spring.kafka.admin.properties.bootstrap.servers**.
8. on starting the app, the following Admin Config Values can be noticed.   
   
9. Next question is if the topic created or not?  
   