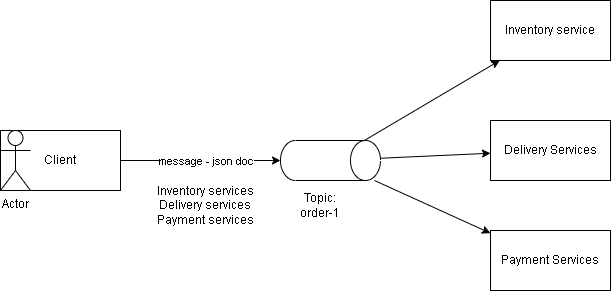
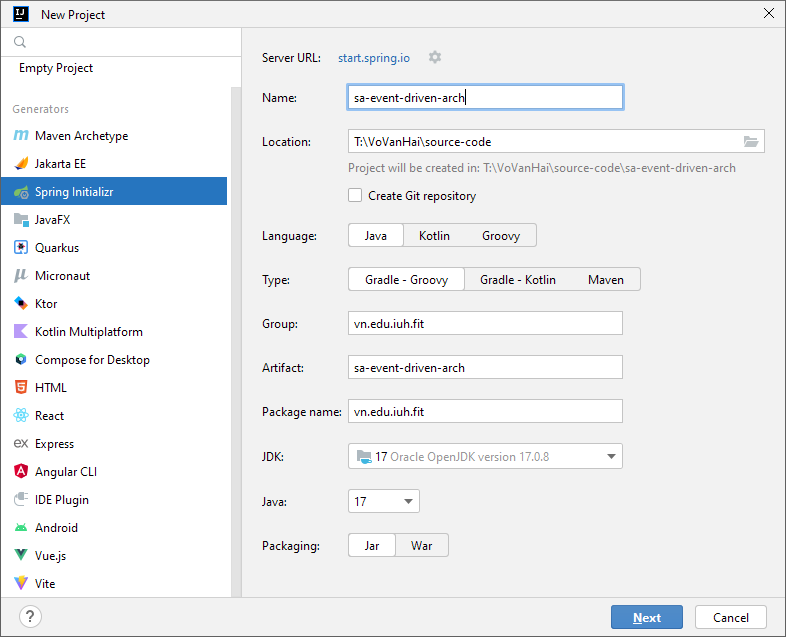
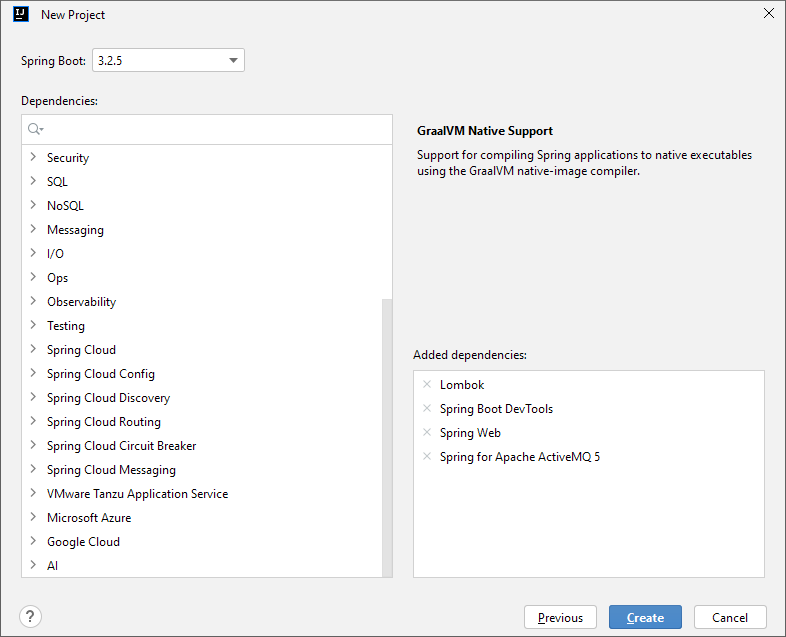
EVENT-DRIVEN ARCHITECTURE

Example model:







To work with ActiveMQ, we should configure some beans that are needed for jms working like, connection factory, jsm template, and so on.

|  |
| --- |
| package vn.edu.iuh.fit.configs;  import org.apache.activemq.ActiveMQConnectionFactory; import org.springframework.context.annotation.Bean; import org.springframework.context.annotation.Configuration; import org.springframework.jms.annotation.EnableJms; import org.springframework.jms.config.DefaultJmsListenerContainerFactory; import org.springframework.jms.core.JmsTemplate;  @Configuration @EnableJms *//@ComponentScan(basePackages = "vn.edu.iuh.fit")* public class JmsConfig {  private static final String BROKER\_URL = "tcp://127.0.0.1:61616";  private static final String BROKER\_USERNAME = "admin";  private static final String BROKER\_PASSWORD = "admin";   @Bean  public ActiveMQConnectionFactory connectionFactory() {  ActiveMQConnectionFactory connectionFactory = new ActiveMQConnectionFactory();  connectionFactory.setBrokerURL(BROKER\_URL);  connectionFactory.setUserName(BROKER\_USERNAME);  connectionFactory.setPassword(BROKER\_PASSWORD);   return connectionFactory;  }   @Bean  public JmsTemplate jmsTemplate() {  JmsTemplate template = new JmsTemplate();  template.setConnectionFactory(connectionFactory());   return template;  }   @Bean  public DefaultJmsListenerContainerFactory jmsListenerContainerFactory() {  DefaultJmsListenerContainerFactory factory = new DefaultJmsListenerContainerFactory();  factory.setConnectionFactory(connectionFactory());  factory.setConcurrency("3-10");  *// true: using jms topic, false: using jms queue* factory.setPubSubDomain(true); *// factory.setPubSubDomain(false);* return factory;  } } |

For listening the messgae that sent to specific topic, the following code will do that

|  |
| --- |
| import jakarta.jms.TextMessage; import org.springframework.context.annotation.ComponentScan; import org.springframework.jms.annotation.JmsListener; import org.springframework.messaging.Message; import org.springframework.messaging.handler.annotation.SendTo; import org.springframework.stereotype.Component;  @Component *//@ComponentScan(basePackages = "vn.edu.iuh.fit")* public class InventoryServices {  @JmsListener(destination = "inventory.topic\_01")  public void receiveMessage(Message<String> message){  System.out.println("\*\*\* received message: "+message);  } } |

For sending the message, the following code can be used

|  |
| --- |
| **import** jakarta**.**jms**.**Destination**;**  **import** jakarta**.**jms**.**JMSException**;**  **import** jakarta**.**jms**.**Message**;**  **import** jakarta**.**jms**.**Session**;**  **import** org**.**apache**.**activemq**.**command**.**ActiveMQDestination**;**  **import** org**.**apache**.**activemq**.**command**.**ActiveMQTopic**;**  **import** org**.**springframework**.**beans**.**factory**.**annotation**.**Autowired**;**  **import** org**.**springframework**.**jms**.**core**.**JmsTemplate**;**  **import** org**.**springframework**.**jms**.**core**.**MessageCreator**;**  **import** org**.**springframework**.**messaging**.**handler**.**annotation**.**SendTo**;**  **import** org**.**springframework**.**stereotype**.**Component**;**  **import** org**.**springframework**.**stereotype**.**Service**;**  @Service  public class SendOrderServices **{**  @Autowired  private JmsTemplate template**;**  @SendTo**(**"inventory.topic\_01"**)**  public void sendOrder**(**String jsonDocs**){**  Destination destination **=** **new** ActiveMQTopic**(**"inventory.topic\_01"**);**  MessageCreator msg**=new** MessageCreator**()** **{**  @Override  public Message createMessage**(**Session session**)** **throws** JMSException **{**  // String json="{'order-infos':'?????????'}";  **return** session**.**createTextMessage**(**jsonDocs**);**  **}**  **};**  template**.**send**(**destination**,**msg**);**  **}**  **}** |