**Event Sourcing with Microservices**

**Objectives:**

**1).Introduction**:This gives the brief overview about the application.

**2)**.**Steps to Configure the Application**:This provides the details about configuration of ReactApp,SpringBoot,MySQL and Apache Kafka after unzipping the file.

**3)**.**Steps to test/run the application**:This provides the working flow of the application and you can verify by the screenshots provided.

**4)**.**Appendices**:The table that is displayed here provides the details of tools ,softwares and their version that are used while developing the application.

**Introduction**: This Project is about an E-commerce food application for ordering Milkshake from Milkshake factory.A user can select the available flavours of Milkshake and select the quantity. After placing the order he is able to view the order details and can track the order.

**Process flow of the application:**

1).The architectural flow diagram of this application is provided in this link. <https://miro.com/app/board/o9J_kpSsh_M=/> .So, in this application we have four services i.e,orderservice,itemservice,milkshakefactory and eventhandler and we have two topics i.e,orders and events.

2).The orderservice receives the data of order from UI and generates order\_id and set the status to Orderplaced and send these order details to ‘events’ topic.Then EventHandler service listens to this topic and the order details which have sent by Order service.It stores the order details and events occurred on that particular order\_id in database and later sends that event to ‘orders’ topic.

3).Now,itemservice listens to this topic and verifies in the database whether the required quantity of the particular flavor is available or not.And it sets the order Status to ‘ItemValidated’ if the required quantity is present else it will set the status to ‘ItemsAreNotAvailable’ and send the event to ‘events’ topic and EventHandler will listen to this and updates the orderstatus in orderdetails and stores the events data and sends the event to ‘orders’ topic .

4).This will be listened by orders service and sets the order status to OrderAccepted.It will be listened by MilkShake service and it sets the orderstatus to MilkShakeStarted and sends to events topic.EventHandler will listen to this and updates the orderstatus in orderdetails and stores the events data and sends the event to ‘orders’ topic.

5).Then order service will listen to this and set the order status to OrderStarted and send the event to ‘events’ topic and EventHandler will listen to this and update the orderstatus in orderdetails and stores the events data and sends the event to ‘orders’ topic .

6).And Milkshake service will listen to this and set the status to MilkShakeFinished after the preparation of order and sends the event ‘MilkShakeFinished’ to event to ‘events’ topic and EventHandler will listen to this and updates the orderstatus in orderdetails and stores the events data and sends the event to ‘orders’ topic.And Order service will listen this and set the status to Orderdelivered and sends to events topic.EventHandler will listen to this and updates the orderstatus in orderdetails and stores the events data and sends the event to ‘orders’ topic

An end user can download the zip file and use the source code of UserInterface service and other microservices to use this application.

**Steps to configure the application**:

**1. REACTAPP:**

**a.)** Open the ‘***ReactApp***’ folder in VisualStudioCode or any IDE.

**b.)** Open the terminal that is visible in the current directory.

**c.)** Use the command **npm install** in the terminal to install the required packages. You are able to see those in node\_modules in your directory that has all the dependencies.

**2.SpringBoot:**

**a.)** Open the services in the ‘***SpringBoot’*** folder in Eclipse or STS or IntelliJIDEA or any IDE.

**b.)**Each service will take some time to be built in the respective IDE as it takes time to configure the dependencies in the pom.xml file of that particular service.

3.**Database Configuration**(MySQL):

**a.)** Open the MySQL shell or workbench on a particular port number(default port number- 3306) and enter the username and password.

**b.)** Make sure that the mysql port and credentials matches with details provided in application properties of the EventHandler service and Item service.

**c.)** Create a database named ‘**ecommerce**’ in the MySQL shell with the following query.

**create database ecommerce;**

**d.)**Create a table for item\_details to store the data for flavours and quantity with the following query.

**create table item\_details (id Integer primary key auto\_increment,flavour String,quantity Integer);**

**e.)**Insert Values into the item\_details table with the following query.

**Insert into item\_details values(1,’oreo’,20);**

**Insert into item\_details values(1,’kitkat’,20);**

**Insert into item\_details values(1,’chocolate’,20);**

**4).Installation and Running Apache Kafka:**

**Installation of Apache Kafka:**

Apache Kafka is supportable on Windows, macOS, as well as on Linux environments. Each operating system has its own steps/process to install Apache Kafka.

Reaching this stage means all aspirants might be clear with the basic concepts of Apache Kafka. Here, in this section, we will discuss the installation process on Windows, setting the path, and starting the zookeeper and Kafka server.

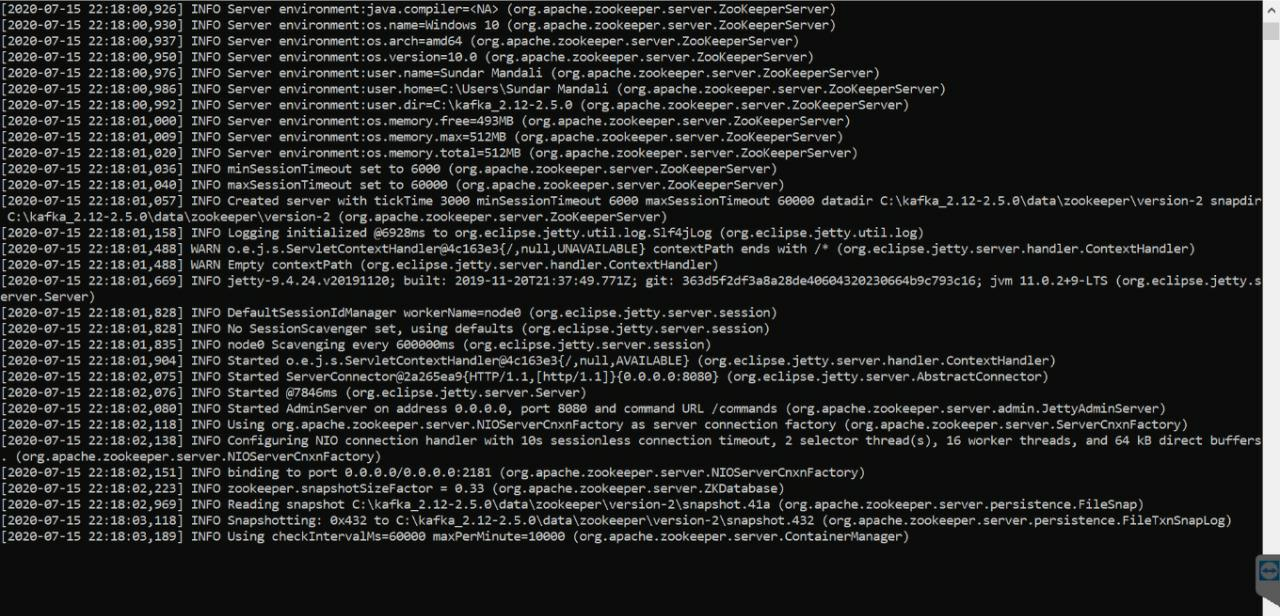
For further installation and configuring kafka follow the link below:

<https://www.javatpoint.com/installation-of-apache-kafka>

**Running of Apache Kafka:**

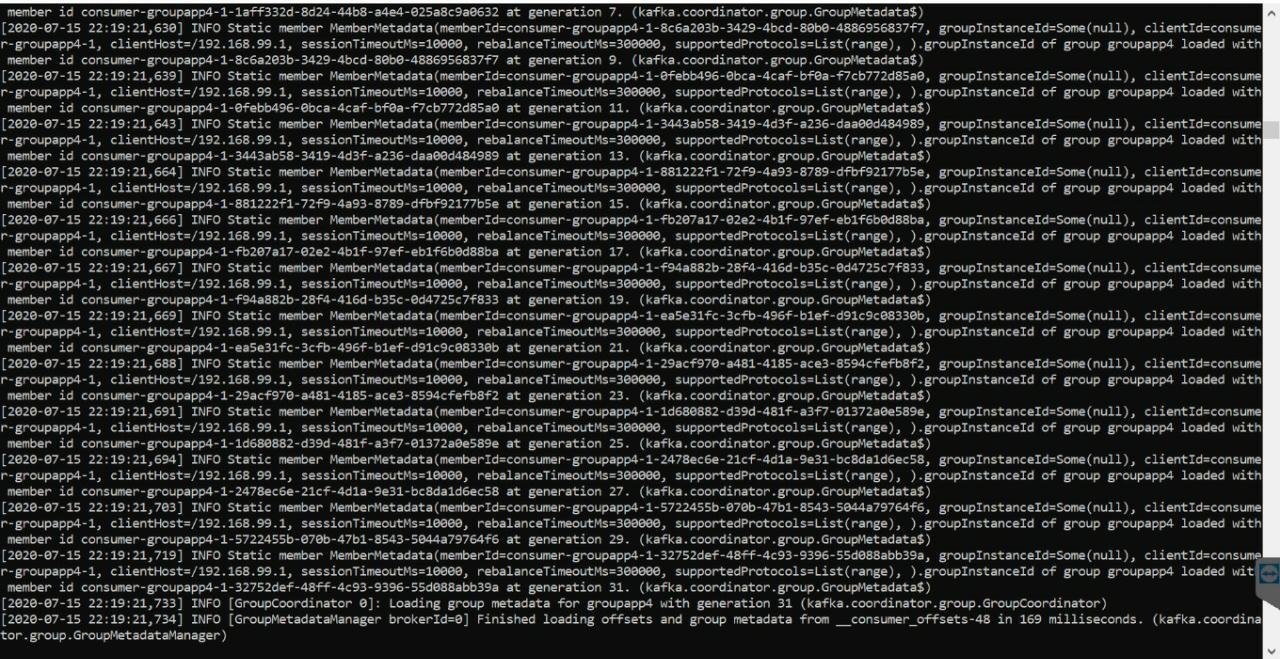
1. Run zookeeper server by using the command.

*zookeeper-server-start.bat C:\kafka\_2.12-2.5.0\config\zookeeper.properties*



1. Run kafka broker server by using the command.

*kafka-server-start.bat C:\kafka\_2.12-2.5.0\config\server.properties*

**

1. Create a kafka topic named as ‘orders’ using the command.

*kafka-topics.bat --zookeeper localhost:2181 --topic orders --create --partitions 1 --replication-factor 1*

4.)Create another topic named as ‘events’

*kafka-topics.bat --zookeeper localhost:2181 --topic events --create --partitions 1 --replication-factor 1*

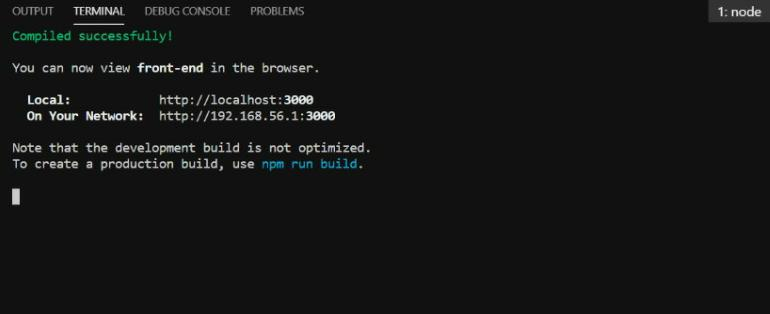
## **Steps to test/run the application:**

Step-1:Open mysql workbench/shell on the particular port number mentioned.

Step-2:Run the Zookeeper and Kafka Broker servers in command prompt.

Step-3:Run the springboot services(orderservice,itemservice,milkshakeservice, eventhandler)

Step-4:Run the React App by using command **npm start** in the terminal of the vscode



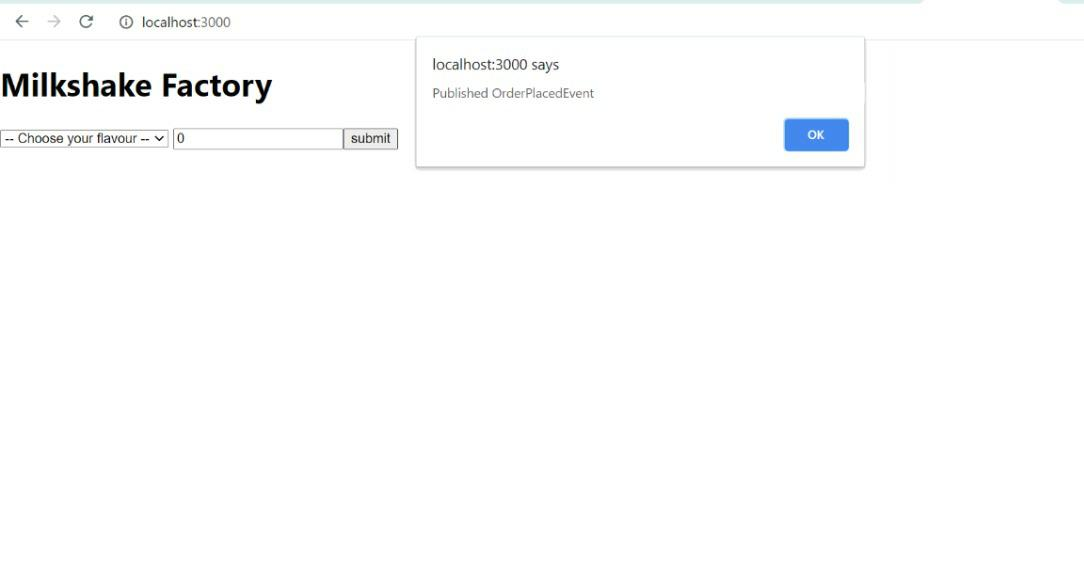
Step-5:After running the react app, the <http://localhost:3000/> will be opened in the default browser of your system and it would be like this..



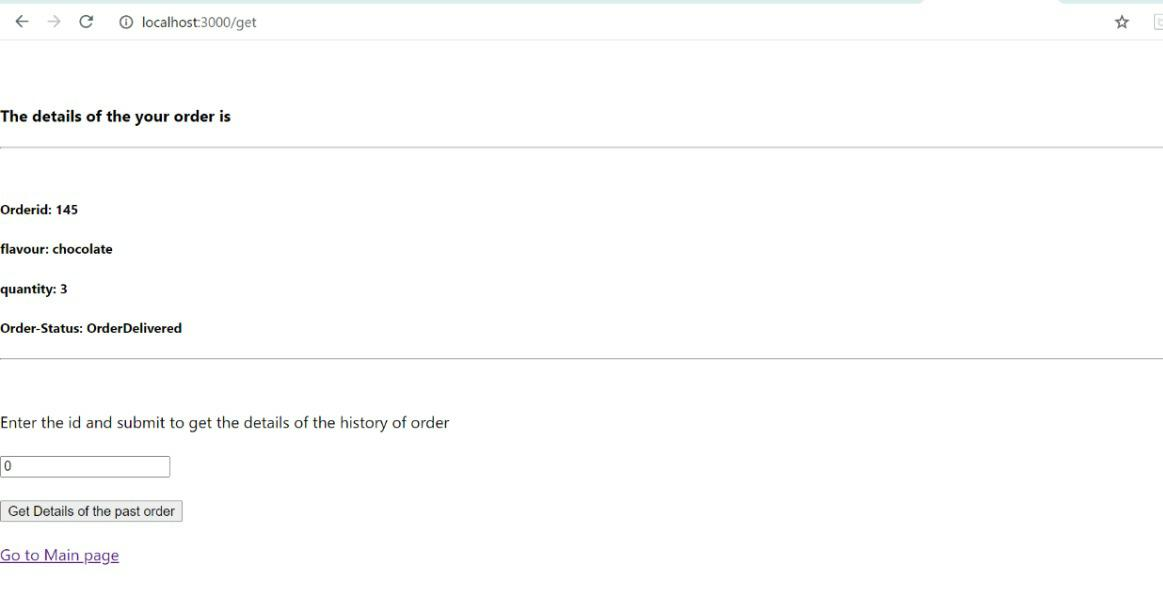
Step-6: Select the flavour and quantity and click on submit to order



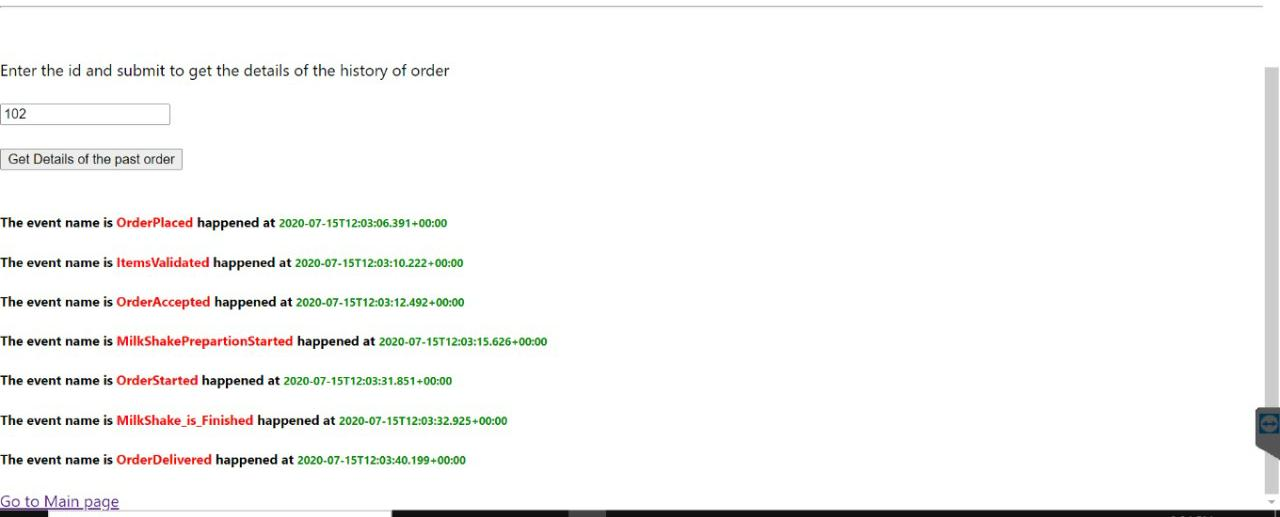
Step-7:A alert window will be visible displaying ‘OrderPlaced Successfully’ then click on OK to proceed.



Step-8:You can view your order details and status of the order



Step-9:We can also view the events that occured in previous orders by specifying the orderid.



## **Appendices:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.NO** | **TOOL/SOFTWARE** | **VERSION** | **PURPOSE** |
| **1.** | Visual Studio Code | 1.47.1 | IDE for react development |
| **2.** | Eclipse IDE | 4.11.0 | IDE for spring boot development |
| **3.** | Apache Kafka | 2.5.0 | Message Broker |
| **4.** | Openjdk | 11.0.2 | For Java Development |
| **5.** | MySQL | 8.0.20 | For Database |
| **6.** | React | 16.13.1 | Front End UI |
| **7.** | Spring Boot | 2.3.1 | Server side development (Java framework) |

## 