Following packages are created:

1. service-discovery (8761):
   1. Used as service registrar
   2. At startup each service register itself in service discovery
2. api-gateway (8080):
   1. Work as API gateway
   2. Single entry point for all the communication
   3. As per the requirement only configured for order-service end points through
      1. To get an order: GET [http://localhost:8080/order/{order-id}](http://localhost:8080/order/%7border-id%7d)
      2. To create an order: POST <http://localhost:8080/order>  
         Body: {

"productId": 1,

"quantity": 1,

"userId": 2

}

1. user-service (8090):
   1. Used to store user details (id, name)
   2. Static data persisted in map
   3. Only 2 users are stored
   4. Supports GET API [http://localhost:8090/user/{user-id}](http://localhost:8090/user/%7buser-id%7d) to fetch the user details
2. product-service (8091):
   1. User to store product details (id, name, price)
   2. Static data persisted in map
   3. Only 2 products are stored
   4. No provision of inventory
   5. Supports GET API [http://localhost:8092/product/{product-id}](http://localhost:8092/product/%7bproduct-id%7d) to fetch the product details
3. order-service (8092):
   1. Used for order creation and retrieve the created order details
   2. Storing order details in a map
   3. New order-id generated by the number of orders created + 1
   4. Currently supporting order for single product with multiple quantities.
   5. Create order payload is {product-id, user-id, quantity}
   6. While saving the order details, it fetches the user and product details from respective services via service discovery
   7. To get the order details of order-id 1 the API will be GET : [http://localhost:8091/order/{order-id}](http://localhost:8091/order/%7border-id%7d)
   8. To store the order details  
      POST: <http://localhost:8091/order>  
      Body {

"productId": 1,

"quantity": 1,

"userId": 2

}