# Per Microservices Tests

1. Catalog
   1. Goal: The catalog service should initialize every product in its stock with 100 in quantity for each. Client should be able to query products of interest through REST GET requests.  
      Step: Query catalog service at Port 8082 with two products: barbie and Lego.  
      Expected: The query service should return product data for barbie and Lego with 100 quantity for each.  
      Graphical user interface, text, application, email

      Description automatically generated  
      Graphical user interface, text, application, email

      Description automatically generated
   2. Goal: The catalog service should return error message when a product that does not exists.  
      Step: Query catalog service at Port 8082 with nonexistence product: test.  
      Expected: Error message stating that product does not exist.  
      Graphical user interface, text, application, email

      Description automatically generated
   3. Goal: The catalog service should be able to process order requested by other services.  
      Step: Request order at catalog service at port 8082  
      Expected: An order should be successfully processed with order data processed, and order number should be -1 because order number is processed by order service  
      Graphical user interface, text, application, email

      Description automatically generated
   4. Goal: The catalog service should be able to respond error message when an product does not exist in stock  
      Step: Request order at catalog service at port 8082  
      Expected: An error message should be returned  
      Graphical user interface, text, application, email

      Description automatically generated
   5. Goal: The stock should be updated according to successful orders  
      Step: Query catalog service with purchased product: barbie  
      Expected: Product data with updated quantity that reflects prior purchase  
      Graphical user interface, text, application, email

      Description automatically generated
   6. Goal: The catalog service should report insufficient stock when requesting quantity is greater than stock, the same test also applies to stock that is 0  
      Step: Request order with quantity greater than 94  
      Expected: Error message indicating that there is insufficient stocks.  
      Graphical user interface, text, application, email

      Description automatically generated
   7. Goal: catalog service should be able to restock commodity when quantity reaches zero  
      Step: Order a product so that its quantity becomes zero, query current stock, after 10 seconds, query the product  
      Expected: the product quantity should be 0 immediately after ordering and after 10 seconds the quantity has restocked to 100  
      Graphical user interface, text, application, email

      Description automatically generated  
      Graphical user interface, text, application, email

      Description automatically generated  
      Graphical user interface, text, application, email

      Description automatically generated
2. Order
   1. The order service should be able to process order requests and return the order information  
      Send order request to order service at port 8083  
      Expecting order data successfully from response  
      Graphical user interface, text, application, email

      Description automatically generated
   2. The order service should be able query a successful order placed before.  
      Send order query to order service at port 8083  
      Expecting previously placed order information  
      Graphical user interface, text, application, email

      Description automatically generated
   3. The order service should be able to replicate order information across three replicas  
      Send order query to other order services at port 8084 and 8085  
      Expecting same order information as before  
        
      Graphical user interface, text, application, email

      Description automatically generated  
      Graphical user interface, text, email

      Description automatically generated