Create a Service Bus Namespace

1. **Sign in to the Azure Portal:**
    a. Navigate to the Azure Portal and log in with your Azure credentials.
2. **Create a Service Bus Namespace:**
    a. In the left-hand navigation pane, click on "+ Create a resource".
    b. In the "New" window, click on "Integration", then select "Service Bus".
    c. Fill in the required information:
    d. Subscription: Choose the Azure subscription you want to use.
    e. Resource Group: Create a new one or select an existing resource group.
    f. Namespace Name: Provide a unique name for your Service Bus namespace.
    g. Region: Select the desired region for your Service Bus namespace.
    h. Pricing Tier: Choose between Basic, Standard, or Premium based on your requirements. (Note: Topics are not available in the Basic tier).
    i. Review the other tabs such as "Networking", "Advanced", and "Tags" for more advanced configurations, but for a basic setup, the default configurations should suffice.
    j. Click the "Review + Create" button. Azure will validate the configuration. Once validation passes, click the "Create" button.
3. **Create a Queue or Topic:**
    a. For Queue:
        i. Once your Service Bus namespace is created, navigate to it in the Azure Portal.
        ii. In the left-hand pane, under "Entities", click on "Queues".
        iii. Click the "+ Queue" button on the top of the pane.
        iv. Provide a name for your queue and configure the desired settings such as size, duration, and other properties based on your requirements.
        v. Click the "Create" button.
    b. For Topic (if you chose a Standard or Premium tier):
        i. In your Service Bus namespace, under "Entities", click on "Topics".
        ii. Click the "+ Topic" button.
        iii. Provide a name for your topic and configure the desired settings.
        iv. Click the “Create" button.
4. **Access Keys and Connection Strings:**
    a. To connect to your Service Bus from applications, you'll need connection strings and access keys.
    b. In your Service Bus namespace, navigate to "Shared access policies" on the left pane.
    c. Here, you can see policies and their respective connection strings and keys. The default is "RootManageSharedAccessKey" which has manage, send, and listen permissions.

Using Python to Send and Consume Messages with Azure Service Bus

1. Set Up the Project Environment:
    a. Navigate to the **Lab4** directory.
    b. Open the **SbDemo** project using PyCharm.
2. Install Required Python Packages:
    a. Open a command prompt or terminal ensuring that Python is available in its path.
    b. Navigate to the directory where the SbDemo project is located.
    c. Install the necessary Python packages for the Service Bus tutorial using the following commands:

```
pip install azure-servicebus
pip install azure-identity
pip install aiohttp
```

3. Inspect and Modify the Code:
    a. Open the code files within the SbDemo project.
    b. Look for the variables and replace them with your specific values:
        i. **FULLY_QUALIFIED_NAMESPACE**: Replace this with your Service Bus namespace.
        ii. **QUEUE_NAME**: Replace this with the name of your Service Bus queue.
4. Run the Code:
    a. In the command prompt or terminal, execute the following commands to send and then receive messages:

```
python send.py; python recv.py
```

Create an alert when Queue or Topic exceeds a specific threshold

1. Navigate to Azure Portal:
    a. Go to the Azure Portal and sign in with your Azure credentials.
2. Access Your Service Bus:
    a. In the left-hand menu, click on "All resources".
    b. Find and select your Service Bus namespace from the list.
3. 3elect Monitoring:
    a. Once inside your Service Bus namespace, look for the "Monitoring" section in the left-hand menu and click on "Alerts".

4. Create a New Alert Rule:
    a. Click on the "+ New alert rule" button.
5. Configure Alert Criteria:
    a. Under the "Condition" section, click on "Signal Name".
    b. From the list of available signals, select "Count of active messages in a Queue/Topic".
    c. In the "Alert logic" section, configure the rule to trigger when the metric is "Greater than" and set the threshold to "10".
6. Configure Action Group:
    a. Under the "Actions" section, click on "Select action group" to determine what actions to take when the alert is triggered. An action group defines a collection of notification preferences.
    b. If you already have an action group, you can select it.
    c. If not, you'll need to create one.
        i. This could include sending an email, and a push notification.
        ii. Setup the azure app for push notification.
7. Configure Alert Details:
    a. Under "Alert rule details":
        i. Alert rule name: Provide a meaningful name for your alert.
        ii. Description: Provide a description (optional but recommended).
        iii. Severity: Choose the appropriate severity level for this alert, like Sev3 (medium) or Sev4 (low).
8. Review and Create:
    a. After configuring all the necessary settings, click on the "Review + create" button. Review your configurations and then click on the "Create alert rule" button.

Get alert on “Count of active messages in a Queue/Topic”

1. Sending Messages:
    a. Open a command prompt or terminal window.
    b. Navigate to the directory containing the send.py script.
    c. Execute the following command to send messages:
    d.

```
python send.py
```

2. Receiving Alerts:
    a. After executing the script, monitor your Microsoft App for any alerts. Ensure notifications are enabled in the app.
3. Peeking at Messages:

    a. Open the Service Bus Explorer.
    b. Navigate to the respective Queue or Topic where the send.py script is sending messages.
    c. Use the "Peek" function to view the messages without removing them from the queue or topic.
4. Observing Sequence Numbers:
    a. As you peek at each message, take note of the sequence number associated with it. Sequence numbers are unique identifiers for messages and can provide insights into the order in which messages were added to the Service Bus
5. Receive the messages to clear the queue
    a. You can use the portal or the python script