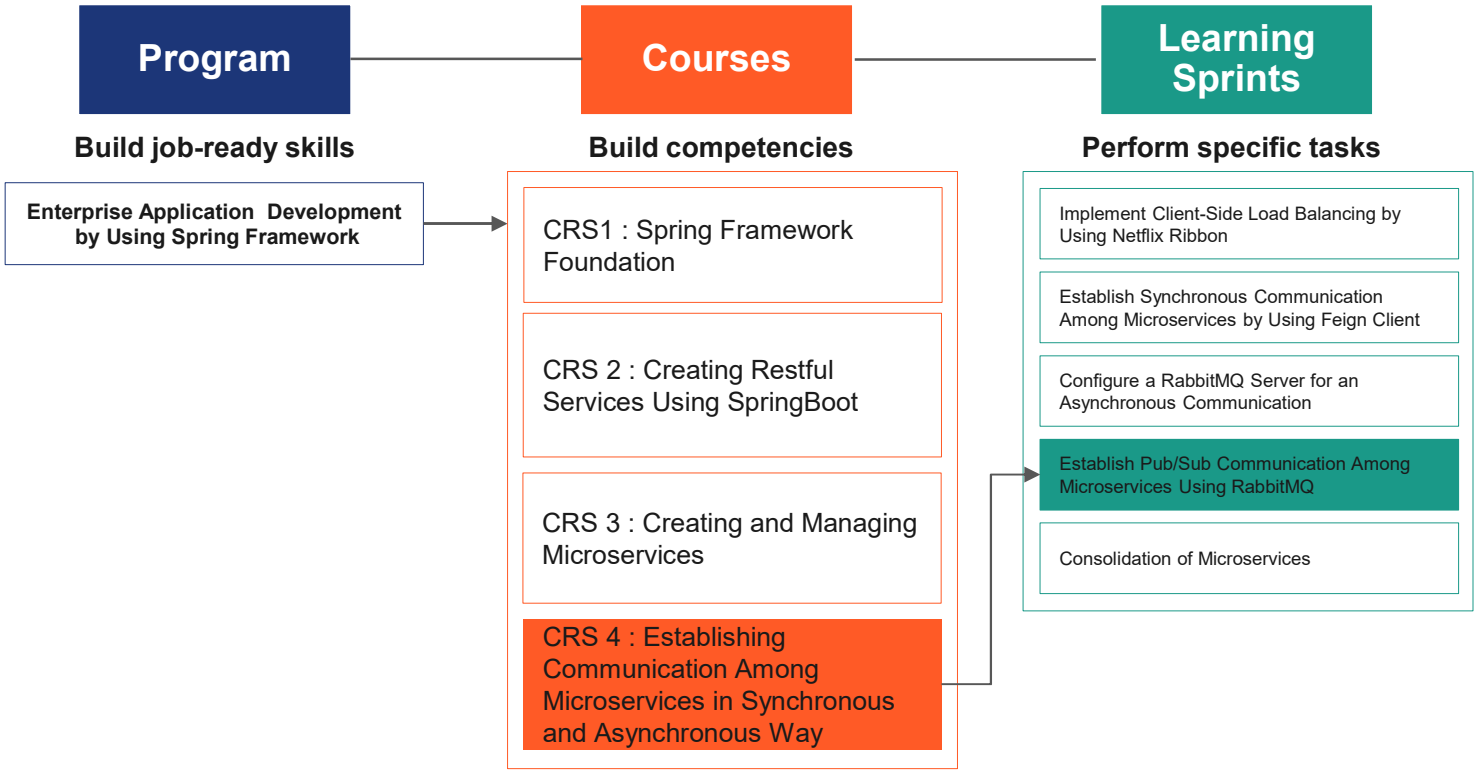
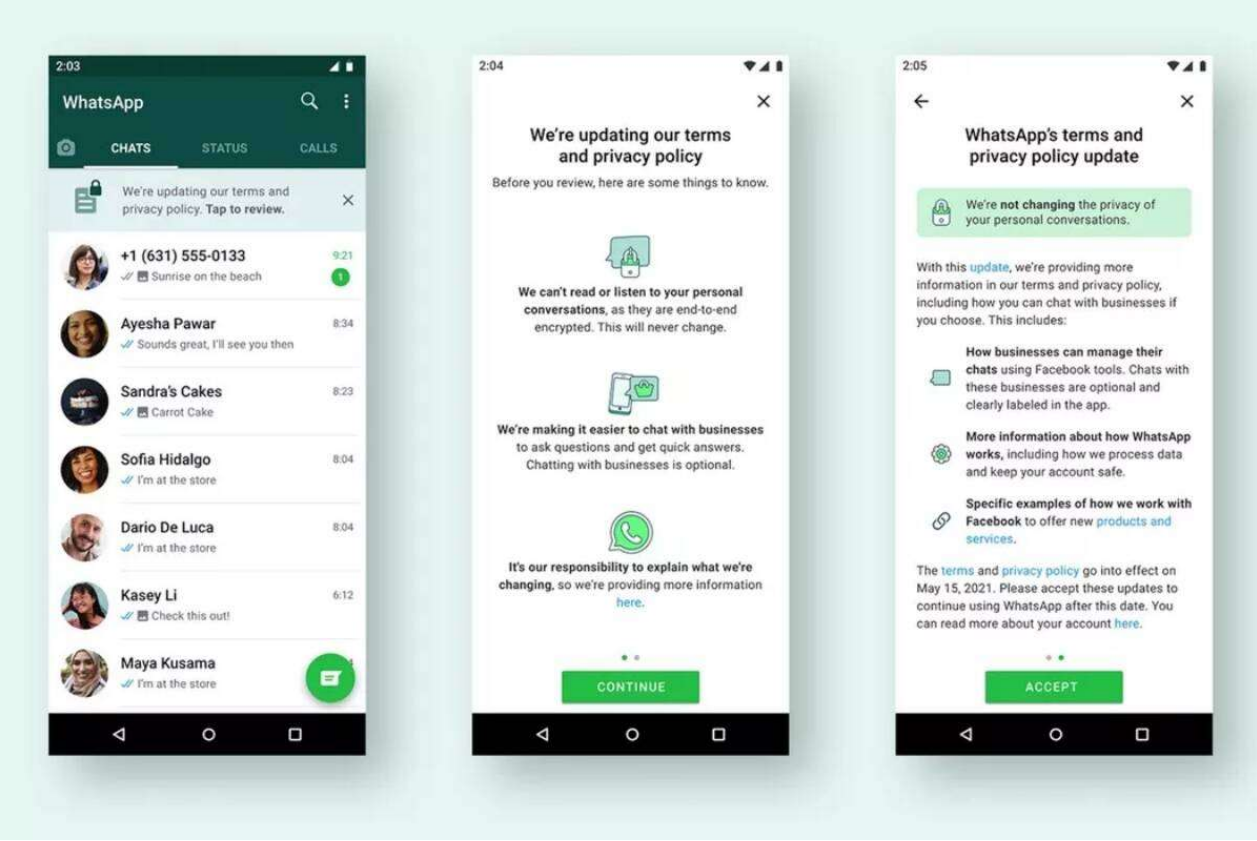
Backend Program: Course 3: Structure



WhatsApp Chatting



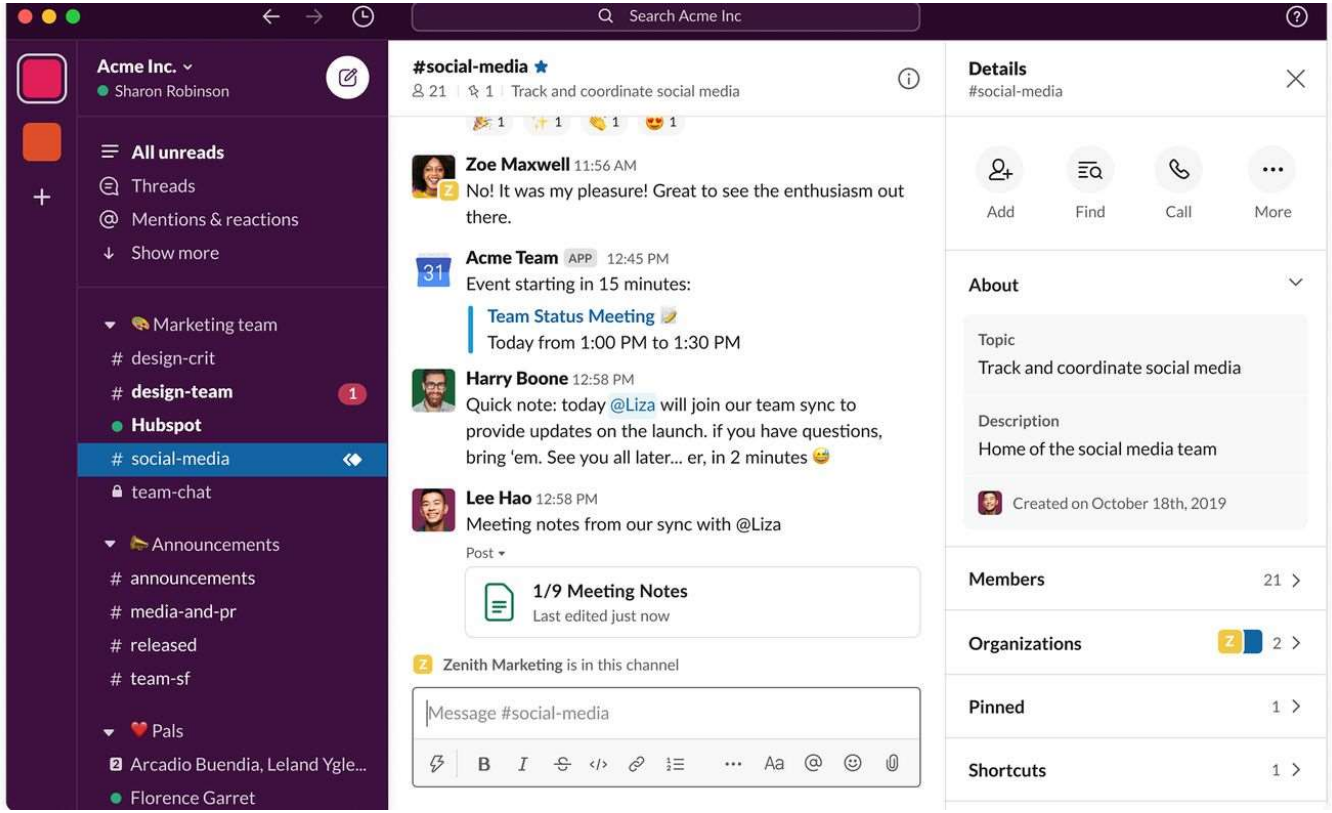
Source:financialexpress.com

Think and Tell

In chat applications like WhatsApp, users send messages to individuals or groups whose contact details they have saved on their cell phones. Those chat messages are seen by other users when they wish to see it.

- Where are the messages stored until they are viewed by the receivers of the message?
- Do messages get deleted if they are not viewed for a long time?
- Does the sender get an acknowledgment that the message has been received by the receiver?

Slack Messaging



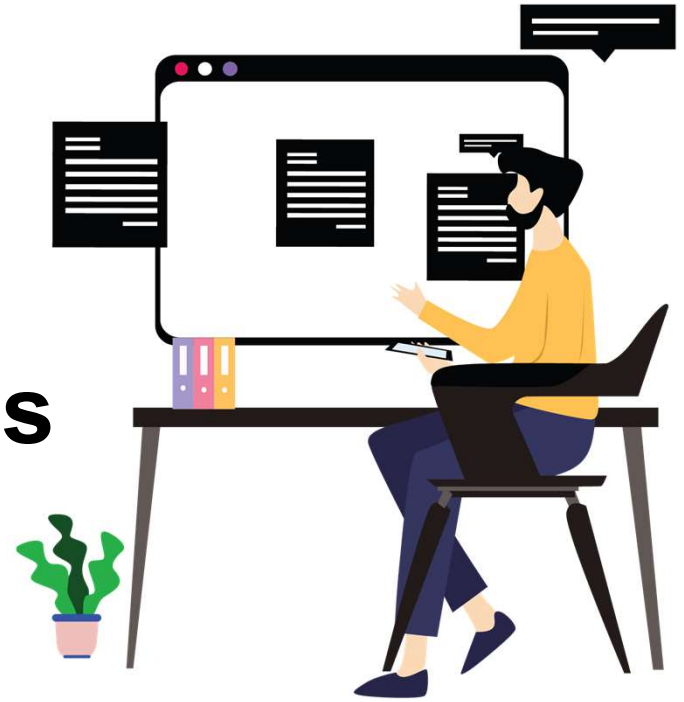
Source:theverge.com

Let Us Discuss

Slack Messaging is another type of chat application. In this the users send direct messages to individuals and groups as channels. Those messages are viewed by the receivers when they open the app.

- Does the sender expect the receiver to respond immediately after sending the message?
- Does the receiver get a notification that there are some messages waiting to be viewed?
- How long will the message will be available for the receiver to view?

Establish PubSub Communication Among Microservices Using RabbitMQ



Learning Objectives

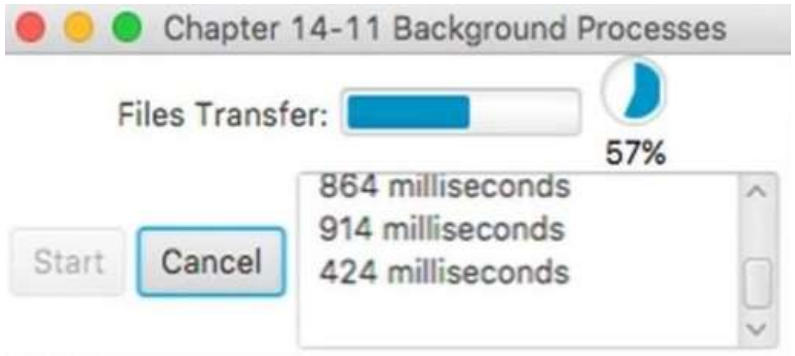
- Explore when and why to use RabbitMQ server
- Explain message queues and when to use them
- Describe exchange and its types
- Establish PubSub communication using direct exchange among microservices

Use of RabbitMQ

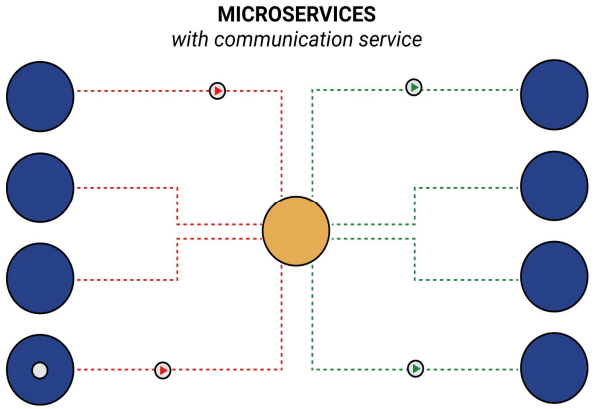
- RabbitMQ enables asynchronous processing, which means that it allows us to put a message in a queue without processing it immediately.
- It is ideal for long-running tasks or blocking tasks. It allows web servers to respond quickly to requests instead of completing computationally intensive tasks immediately that may delay the response time.
- It simply stores messages and passes them to consumers when ready.
- Message queuing is good when we want to distribute a message to multiple consumers or to balance loads between workers.

When Should We Use a Message Queue?

- A message queue is a component used for an inter-process communication. It passes control, content or messages to another component.
- Message queues are beneficial in the following use cases:
  - For long-running processes and background jobs
  - As a middleman in between microservices



Source: holooly.com



Source:cloudamqp.com

Long Running Processes and Background Jobs

- The perfect scenario to incorporate a message queue is when requests take a significant amount of time to process a request.
- Some real-life examples could include:
  - Images Scaling
  - Sending large/many emails
  - Search engine indexing
  - File scanning
  - Video encoding
  - Delivering notifications
  - PDF processing
  - Calculations

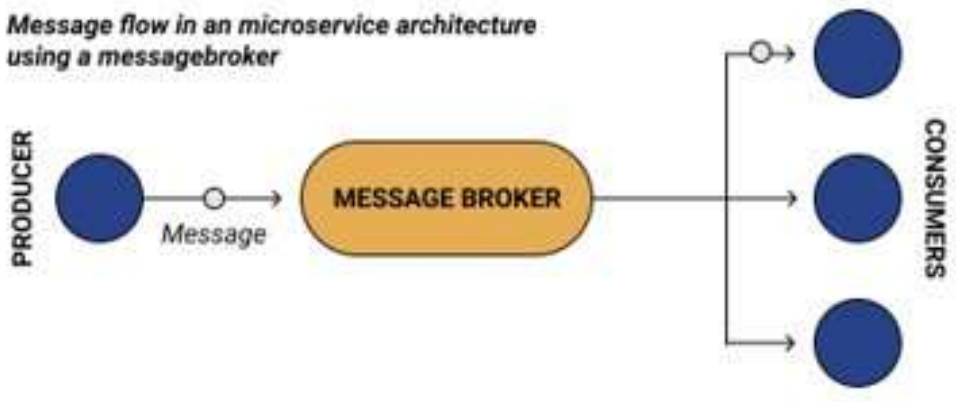
Middleman in Between Microservices

- A message queue is useful for communication and integration within and between applications. For example, as a middleman between microservices.
- When a system needs to notify another part of the system to start working on a task or when there are a lot of requests coming in at the same time, as in case of the following scenarios:
  - Order handling (place an order, update the status of the order, send the order, payment, etc.)
  - Food delivery service (Place an order, prepare an order, deliver food)
  - Any web service that needs to handle multiple requests.

RabbitMQ in Microservices Architecture

RabbitMQ is a reliable open-source message broker. It is scalable and flexible.

- It supports several standardized protocols such as AMQP, MQTT, STOMP, etc.
- It is used by many companies within various industries.
- It is user-friendly as it is easy to tweak the configurations to suit the intended purpose.



Source:cloudamqp.com

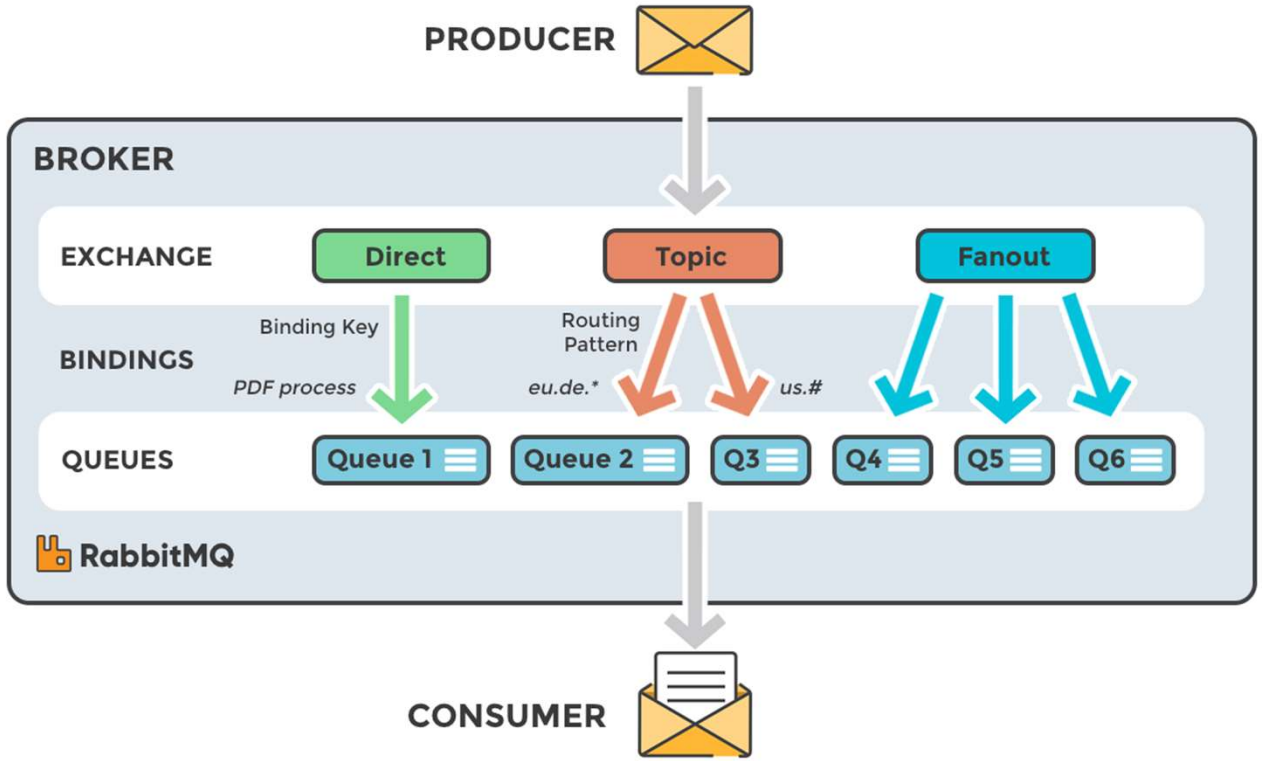
Exchanges in RabbitMQ

- An exchange routes messages to different queues with the help of bindings and routing keys.
- Messages are not published directly to a queue. A producer sends these messages instead to an exchange.
- A link between a queue and an exchange is called binding.

Types of Exchanges

- **Direct Exchange:** These exchanges route messages to queues that are bound to them, if the binding routing key is identical to the message routing key.
- **Fanout Exchange:** It route messages to all the queues that are bound to them. The message routing key is ignored.
- **Topic Exchange:** It route messages to bound queues if the message routing key matches the pattern specified in the binding routing key.
- **Header Exchange:** These exchanges route messages to bound queues based on multiple attributes expressed as message headers rather than the routing key.

Types of Exchanges



Source: cloudamqp.com

Attributes of Exchanges

Exchanges are declared with several attributes. The most important of ones are:

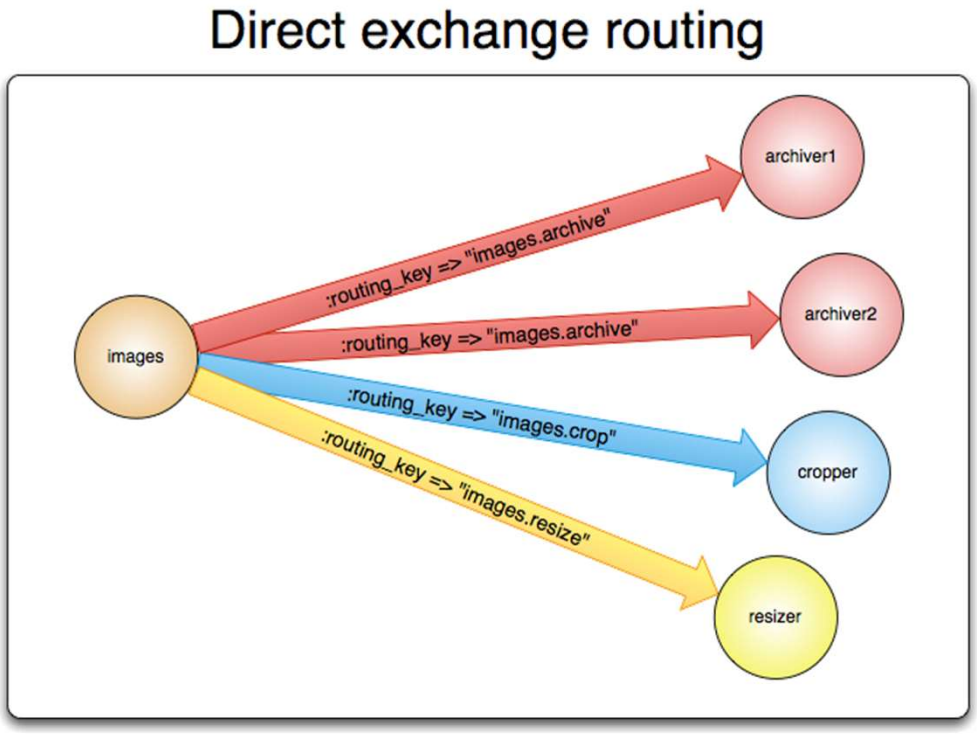
- Name
- Durability (exchanges survive broker restart)
- Auto-delete (exchange is deleted when last queue is unbound from it)
- Arguments (optional, used by plugins and broker-specific features)

Direct Exchange

A direct exchange delivers messages to queues based on the message routing key.

It works like this:
A queue binds to the exchange with the routing key K.

When a new message with routing key R arrives at the direct exchange, the exchange routes it to the queue if K = R.



Source: rabbitmq.com

Quick Check

Which one of the following is a message routing agent that routes the message to different queues after accepting the same from the producer application?

1. Message broker
2. Exchange
3. Bindings
4. Routing key

Quick Check

Which one of the following is a message routing agent that routes the message to different queues after accepting the same from the producer application?

1. Message broker
2. **Exchange**
3. Bindings
4. Routing key

RabbitMQ Elements and Concepts

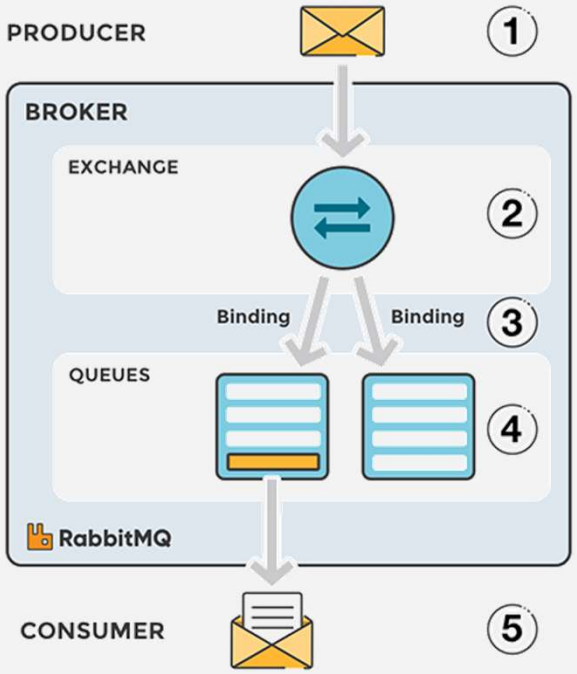
- **Producer**: It is an application that sends messages.
- **Consumer**: It is an application that receives the messages.
- **Queue**: It is a buffer that stores messages.
- **Message**: Information sent from the producer to a consumer using RabbitMQ.
- **Connection**: It is a TCP connection between the application and the RabbitMQ broker.
- **Channel**: A virtual connection inside a connection. Publishing and consuming messages from a queue is all done over a channel
- **Exchange**: Receives messages from producers and pushes them to queues depending on the rules defined by the exchange type. To receive messages, a queue needs to be bound to at least one exchange.
- **Binding**: A link between a queue and an exchange is called binding.

RabbitMQ Elements and Concepts (contd.)

- **Routing key**: The exchange uses the key to decide how to route the message to queues. A key can be thought like an address for the message.
- **AMQP**: Advanced Message Queuing Protocol is the protocol used by RabbitMQ for messaging.
- **Users**: Users can connect to RabbitMQ with a given username and password. Every user can be assigned permissions to read, write and configure privileges within the instance. Users can also be assigned permissions for specific virtual hosts.
- **Vhost, virtual host**: Segregate applications using the same RabbitMQ instance. Different users can have different permissions for different vhost. Queues and exchanges can be created, so that they only exist in one vhost.

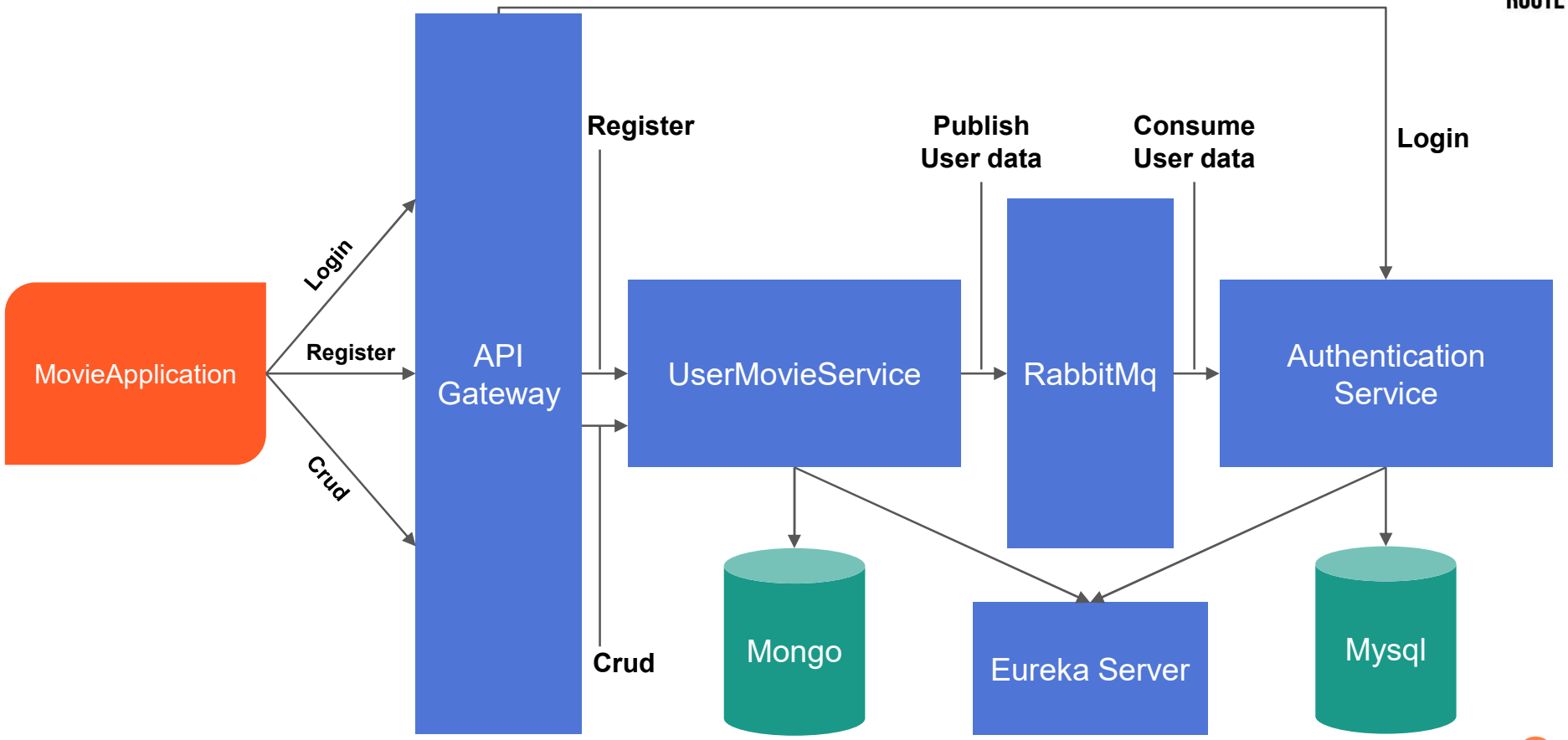
Message Flow in RabbitMQ

- The producer publishes a message to an exchange. Specifying the exchange type is a must while creating an exchange.
- The exchange receives the message and then routes the message. The exchange takes different message attributes into account, such as the routing key, depending on the exchange type for routing.
- Bindings must be created from the exchange to the queues.
- The messages stay in the queue until they are consumed by a consumer.
- The consumer processes the message.



Source:cloudampq.com

RabbitMQ in Movie Application



Publish or Subscribe User Data in Authentication Service

Publish the user data from the user service in the movie application which is later consumed by authentication service by subscribing to it.

DEMO

Publish and Subscribe Messages

RabbitMQ uses a protocol called AMQP by default. To be able to communicate with RabbitMQ you need a client library that understands the same protocol as RabbitMQ.

1. In a Spring Boot application, add the required AMQP client library as dependency in pom.xml.
2. In a Spring configuration file, declare/create a queue. A queue will be created if it does not already exist when we declare it. All queues must be declared before they can be used.
3. Configure exchanges and bind a queue to an exchange in the subscriber or consumer. All exchanges need to be declared before they can be used. An exchange accepts messages from a producer application which it routes them to message queues. For messages to be routed to queues, queues must be bound to an exchange using bindings.
4. In publisher, publish a message to an exchange.
5. In subscriber/consumer, consume a message from a queue.

Message Acknowledgments

- Auto Acknowledgment: Server removes the content from a message queue immediately after it is sent out (written to a TCP socket). These auto acknowledgements should be considered unsafe and not suitable for all workloads.

- Positive Acknowledgment: A manual client acknowledgement instructs RabbitMQ that messages have been delivered successfully and can be discarded. The method basic.ack is used for positive acknowledgements.

- Negative Acknowledgment: AMQP specification defines the basic.reject method which allows clients to reject individual, delivered messages, and thereby asks the broker to either discard them or requeue them. The method basic.nack can be used for bulk processing of messages which is not supported by basic.reject.

Quick Check

What type of exchange in RabbitMQ routes messages to all the queues that are bound to it without using the routing key?

1. Topic exchange
2. Direct exchange
3. Header exchange
4. Fanout exchange

Quick Check

What type of exchange in RabbitMQ routes messages to all the queues that are bound to it without using the routing key?

1. Topic Exchange
2. Direct Exchange
3. Header Exchange
4. **Fanout Exchange**

Key Takeaways

- Use of the RabbitMQ server
- Exchanges and type of exchanges
- Working of a direct exchange
- Publish or subscribe messages through direct exchange
- Message acknowledgements

STACK ROUTE

Thank you!