✓ 100 XP

Summary

1 minute

Your retail company experienced tremendous growth, forcing it to move past direct communication between microservices and consider more elegant communication solutions. The solution your scenario requires should be fault-tolerant, scalable, and resilient.

You explored the **Streams** and **Pub/Sub** features in Azure Cache for Redis. You found that these features helped in the following ways:

- Pub/Sub made it possible to create an event aggregation system where microservices could “fire and forget” events to clients without knowing how many, if any, were listening.
- Streams made it possible to create a message broker system where many different microservices could consume messages at scale.

Without Redis, your team would have to write the middleware manually to implement these features. But even with Redis, your team would have been responsible for managing a cluster of servers. With Azure Cache for Redis, your team got the benefits of Redis without having to worry about managing servers while still using many of the scalability and resiliency benefits of the Azure platform.

With the Pub/Sub and Streams features, your team can quickly and easily implement messaging between microservices and even add new features in the future with minimal planning or setup.

References

- Azure Cache for Redis overview
- Azure Cache for Redis documentation
- Quickstart: Create an open-source Redis cache
- Redis Pub/Sub
- Redis Streams

All units complete:

Complete module