27

Oracle Backend for Microservices and AI

This chapter explains the Oracle Backend for Microservices and AI platform (also referred to as OBaaS).

Topics:

- About Oracle Backend for Microservices and AI
- Getting Started with Oracle Backend for Microservices and AI
- CloudBank Sample Application

27.1 About Oracle Backend for Microservices and AI

Spring Boot is a popular framework for building modern, cloud-native applications. 80% of the modern enterprise application development in Java uses Spring Boot for microservices. As noted earlier, deploying microservices in the production environment is not an easy task. A Backend as a Self-Service (BaaS) is the most important pattern for success because it executes the other 11 patterns (discussed earlier) in an integrated platform, which can be deployed in a few clicks, on any cloud – public or private.

The Oracle Backend for Microservices and AI platform, which is also referred to as Oracle Backend as a Self-Service (OBaaS), includes these important features:

- OBaaS deploys with Kubernetes on multiple clouds, on-premises, and on a laptop for rapid development of microservices using Spring Starters for Oracle Database.
- Applications can take advantage of the converged Oracle Database that features a variety of data types – relational, document/JSON, graph, spatial – and serves both the data at rest and data in motion use cases for the most challenging enterprise applications.
- Oracle Database includes an event queue or a streaming feature called Transactional Event Queues (TxEventQ) that is very similar to Apache Kafka for streaming use cases, making Oracle Database a message broker.
- Containerization of Oracle Database with pluggable databases (PDBs) makes it a natural fit for isolating microservices to a single PDB, separated by bounded contexts for each microservice, or in some cases, using schema-level isolation with fewer PDBs to contain the costs of deployment.

27.2 Getting Started with Oracle Backend for Microservices and AI

This section provides essential information for developers looking to get started with the Oracle Backend for Microservices and AI platform.

The Oracle Backend for Microservices and AI platform (OBaaS) is described at the following location: https://bit.ly/OracleAI-microservices, which also includes information about using the deployment paths on Oracle Cloud Infrastructure (OCI) from the OCI Marketplace to deploy the platform in a few clicks using the OCI Resource Manager templates with Terraform and Ansible. Additionally, using the custom install path, you can deploy the platform on Microsoft

Azure, OpenShift, Tanzu – allowing Bring Your Own (BYO) of Oracle Database, Kubernetes cluster, and networking.

In addition to an Oracle Autonomous Database Serverless instance, the following software components are deployed in an OCI Container Engine for Kubernetes cluster (OKE cluster). The following list of components is an example of the services that run in OBaaS for supporting application deployment in the production environment.

Note:

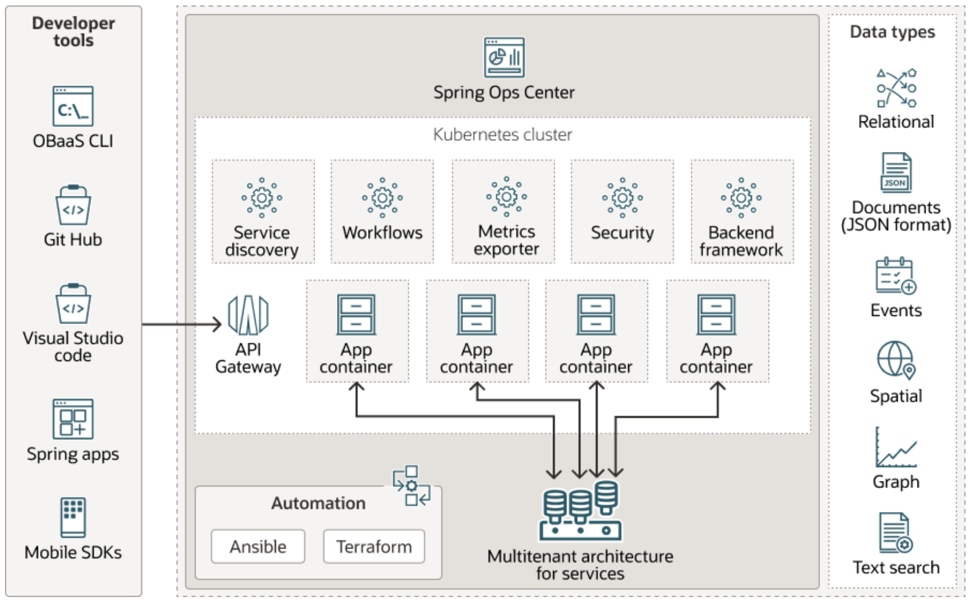
Always check for the latest release of these software components at the following location: https://bit.ly/OracleAI-microservices.

- Apache APISIX API Gateway and Dashboard
- Apache Kafka
- Coherence Operator
- Conductor Server
- Grafana
- HashiCorp Vault
- Jaeger
- Apache Kafka
- Loki
- Netflix Conductor
- OpenTelemetry Collector
- Oracle Autonomous Database Serverless
- Oracle Backend for Spring Boot Command Line Interface (CLI)
- Oracle Backend for Spring Boot Visual Studio Code plug-in
- Oracle Database Operator for Kubernetes (OraOperator or the operator)
- Oracle Transaction Manager for Microservices (MicroTx)
- Parse and Parse Dashboard (optional)
- Prometheus
- Promtail
- Spring Boot Admin dashboard
- Spring Cloud Config server
- Spring Cloud Eureka service registry
- Strimzi Kafka Operator

OBaaS integrates multiple platform services that are needed for applications and data; using Oracle Database with Kubernetes and the Oracle Database Operator for Kubernetes, thus making DevOps for microservices simple.

The following figure illustrates a high-level view of the OBaaS platform's architecture.

Figure 27-1   Oracle Backend for Microservices and AI (OBaaS)



Developers also have access to development or build-time services and libraries, including:

- A command-line interface (CLI) to manage service deployment and configuration, including database schema management
- Visual Studio Code (VS Code) plug-in to manage service deployment and configuration
- Spring Data (Java Persistence API (JPA) and Oracle JDBC) to access Oracle Database
- Oracle Java Database Connectivity (Oracle JDBC) Drivers
- Spring Cloud Config Client
- Spring Eureka Service Discovery Client
- Spring Cloud OpenFeign
- OpenTelemetry Collector (including automatic instrumentation)
- Spring Starters for Oracle Universal Connection Pool (UCP), Oracle Wallet, Oracle Advanced Queuing (AQ), and Transactional Event Queues (TxEventQ)

OBaaS is also extensible, and so you can add additional Kubernetes Operators (example, for Weblogic) and other components that they deem important (example, Redis). These additional components may get included in OBaaS in the future; if there are enough customers requesting a tested platform, and a path to support from the community (if open source), or from a vendor (if using a free or paid version).

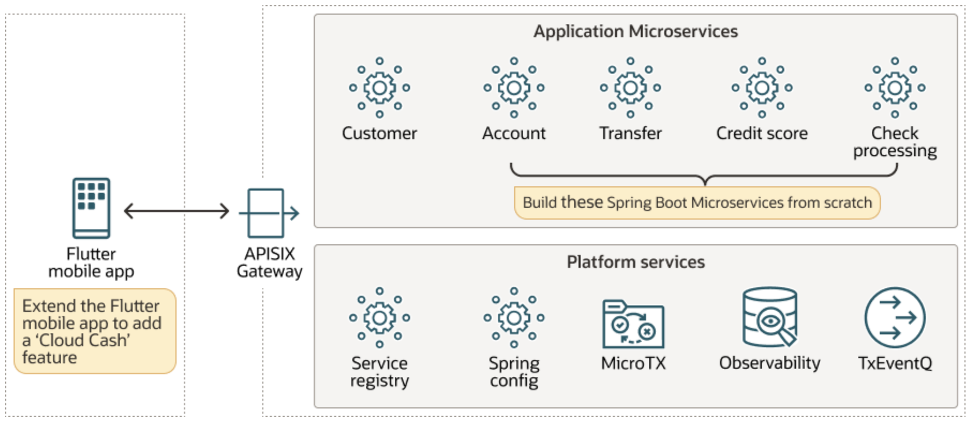
For more information about the OBaaS deployment and how to use OBaaS in the production environment, see https://bit.ly/OracleAI-microservices.

27.2.1 CloudBank Sample Application

Oracle provides a sample application called CloudBank that can be used to learn how to use the Oracle Backend for Microservices and AI platform (OBaaS).

CloudBank is a microservices-based application that installs the Oracle Backend for Microservices and AI platform and deploys a few microservices – Customer, Account, Transfer, Credit Score, Check Processing – as shown in the following figure.

Figure 27-2 CloudBank Sample Application



In the CloudBank hands-on lab, you can learn how to:

- Install Oracle Backend for Microservices and AI.
- Set up a development environment for Spring Boot.
- Build Spring Boot microservices from scratch using Spring Web to create Representational State Transfer (REST) services.
- Use service discovery and client-side load balancing.
- Use Spring Actuator to allow monitoring of services.
- Create services that use asynchronous messaging with Java Message Service (JMS) instead of REST.
- Implement the Saga pattern to manage data consistency across microservices.
- Use the APISIX API Gateway to expose services to clients.
- Extend a provided Flutter client to add a new "cloud cash" feature that uses the services you have built.

You can access the sample CloudBank application at the following location: https://bit.ly/cloudbankAI.