OKE Microservice initial set-up

Summary

- This document provides guidance on the steps needed for setting up initial OKE Micro-service

Steps to follow for new micro-service set-up

- ### Application packaging structure
    1. Create a new folder with project/application name
    2. Under root folder, create below sub folders
        - **eclipse** -- folder which contains the actual project files
        - **xfd** -- folder that contains all table/Index/synonyms/grants/sequences xdf files
        - **auto-sql-sqlplus** -- folder with initial scripts to load some DB data/ config properties etc
        - **.gitignore** file with Files and directories to ignore
        - **build.pipeline**
        - **install.pipeline**
        - **README.md**
    3. Eclipse folder will have the application specific code base

    Sample packaging structure

    ```
    ApplicationName
            auto-sql-sqlplus
            ecliple
            xdf
            .gitignore
            build.pipeline
            install.pipeline
            README.md
    ```

- ### Application Initial setup
    - Application code will reside in eclipse folder
    - use start.spring.io for initial setup with project groupId and artifactId. Then download the project and load into IDE, then Create 3 folders (application, domain, infrastructure) under main package and same for test folder as well

Sample packaging

```
ApplicationName
         src
                main
                        java
                                com.initial.setup.example
                                        application
                                        domain
                                        infrastructure
                                Application.java
                        resources
                test
                        java
                                com.initial.setup.example
                                        application
                                        domain
                                        infrastructure
                                ApplicationSmokeTest.java
                        resources
        Docker
        pom.xml
```

- **POM xml changes**
  - Add below dependencies/ profiles/ plugins/ repositories in POM.xml

Pom.xml structure

```
a. Spring-boot starter parent
b. Properties in pom
c. Profiles
d. Dependency Management
e. Dependencies
        1. spring-boot-starter-web
        2. spring-boot-starter-logging
        3. spring-cloud-starter-config
        4. spring-boot-starter-data-jpa
        5. lombok
        6. spring-boot-starter-log4j2
        7. log4j-web
        8. org.eclipse.persistence.jpa
        9. org.eclipse.persistence.extension
        10. Jackson dependencies
        11. org.springframework.kafka
        12. spring-boot-starter-test
        13. h2 -- for local
        14. spring-boot-starter-actuator
        15. micrometer-registry-prometheus
        16. oci-java-sdk-vault
        17. oci-java-sdk-secrets
        18. oauth-security -- for Oauth
f. Build plugins
        1. git-commit-id-plugin -- for git versioning information
        2. maven-jar-plugin
        3. spring-boot-maven-plugin
        4. maven-checkstyle-plugin
        5. dockerfile-maven-plugin
        6. maven-surefire-plugin
        7. jacoco-maven-plugin
        8. maven-javadoc-plugin
        9. dependency-check-maven
g. Repositories
h. Plugin Repositories
i. Reporting
```

pom.xml sample

```
<?xml version = "1.0" encoding = "UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-
4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>

  <groupId>oal.oracle.apps.ic.admin</groupId>
  <artifactId>oalcn-ms-admin-service</artifactId>
  <version>0.0.2-SNAPSHOT</version>
  <packaging>jar</packaging>

  <name>OalcnMsAdminService</name>
  <description>OAL MicroServices Dashboard</description>

  <parent>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-parent</artifactId>
    <version>2.6.6</version>
    <relativePath/> <!-- lookup parent from repository -->
  </parent>

  <properties>
    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
    <project.reporting.outputEncoding>UTF-8</project.reporting.outputEncoding>
    <java.version>11</java.version>
    <maven.compiler.source>${java.version}</maven.compiler.source>
    <maven.compiler.target>${java.version}</maven.compiler.target>
    <spring-cloud.version>2021.0.1</spring-cloud.version>
    <docker.image.prefix>oal</docker.image.prefix>
    <version.number>${git.commit.time}.${git.commit.id.abbrev}</version.number>
    <log4j2.version>2.17.1</log4j2.version>
  </properties>

  <dependencyManagement>
    <dependencies>
      <dependency>
        <groupId>org.springframework.cloud</groupId>
        <artifactId>spring-cloud-dependencies</artifactId>
        <version>${spring-cloud.version}</version>
        <type>pom</type>
        <scope>import</scope>
      </dependency>
      <dependency>
        <groupId>com.oracle.oci.sdk</groupId>
        <artifactId>oci-java-sdk-bom</artifactId>
        <version>1.17.1</version>
        <type>pom</type>
        <scope>import</scope>
      </dependency>
      <dependency>
        <groupId>com.fasterxml.jackson.core</groupId>
        <artifactId>jackson-databind</artifactId>
        <version>2.13.2.2</version>
      </dependency>
      <dependency>
        <groupId>com.fasterxml.jackson.core</groupId>
        <artifactId>jackson-core</artifactId>
        <version>2.13.0</version>
      </dependency>
      <dependency>
        <groupId>com.fasterxml.jackson.core</groupId>
        <artifactId>jackson-annotations</artifactId>
        <version>2.13.0</version>
      </dependency>
      <dependency>
        <groupId>org.glassfish.jersey.core</groupId>
        <artifactId>jersey-common</artifactId>
```

```
      <version>2.30.1</version>
    </dependency>
  </dependencies>
</dependencyManagement>

<profiles>
  <profile>
    <id>development</id>
    <properties>
      <dependency.scope>runtime</dependency.scope>
    </properties>
  </profile>
  <profile>
    <id>test</id>
    <properties>
      <dependency.scope>test</dependency.scope>
    </properties>
    <activation>
      <activeByDefault>true</activeByDefault>
    </activation>
  </profile>
</profiles>

<dependencies>
  <dependency>
    <groupId>org.springframework.cloud</groupId>
    <artifactId>spring-cloud-starter-config</artifactId>
    <exclusions>
      <exclusion>
        <artifactId>spring-security-crypto</artifactId>
        <groupId>org.springframework.security</groupId>
      </exclusion>
      <exclusion>
        <artifactId>spring-security-rsa</artifactId>
        <groupId>org.springframework.security</groupId>
      </exclusion>
    </exclusions>
  </dependency>
  <!-- Starter Web -->
  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-web</artifactId>
    <exclusions>
      <exclusion>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-logging</artifactId>
      </exclusion>
      <exclusion>
        <artifactId>tomcat-embed-logging-juli</artifactId>
        <groupId>org.apache.tomcat.embed</groupId>
      </exclusion>
      <exclusion>
        <artifactId>tomcat-embed-websocket</artifactId>
        <groupId>org.apache.tomcat.embed</groupId>
      </exclusion>
      <exclusion>
        <artifactId>tomcat-embed-el</artifactId>
        <groupId>org.apache.tomcat.embed</groupId>
      </exclusion>
      <exclusion>
        <groupId>org.yaml</groupId>
        <artifactId>snakeyaml</artifactId>
      </exclusion>
      <exclusion>
        <groupId>org.glassfish</groupId>
        <artifactId>jakarta.el</artifactId>
      </exclusion>
    </exclusions>
  </dependency>
  <!-- Starter for Persistence (JPA) -->
  <dependency>
```

```
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-data-jpa</artifactId>
  <exclusions>
    <exclusion>
      <artifactId>hibernate-entitymanager</artifactId>
      <groupId>org.hibernate</groupId>
    </exclusion>
    <exclusion>
      <groupId>org.hibernate</groupId>
      <artifactId>hibernate-core</artifactId>
    </exclusion>
  </exclusions>
</dependency>
<!-- Technical Starter for Log4J2 -->
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-log4j2</artifactId>
</dependency>

<dependency>
                    <groupId>org.apache.logging.log4j</groupId>
                    <artifactId>log4j-web</artifactId>
</dependency>
<!-- Test Starter -->
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-test</artifactId>
  <scope>test</scope>
  <exclusions>
    <exclusion>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-starter-logging</artifactId>
    </exclusion>
  </exclusions>
</dependency>
<!-- Actuator starter -->
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-actuator</artifactId>
  <optional>true</optional>
</dependency>

<dependency>
  <groupId>io.micrometer</groupId>
  <artifactId>micrometer-registry-prometheus</artifactId>
  <version>1.8.0</version>
</dependency>

<dependency>
  <groupId>org.springframework.kafka</groupId>
  <artifactId>spring-kafka</artifactId>
</dependency>

<dependency>
  <groupId>org.springframework.data</groupId>
  <artifactId>spring-data-rest-webmvc</artifactId>
  <version>3.6.3</version>
</dependency>

<dependency>
  <groupId>org.eclipse.persistence</groupId>
  <artifactId>org.eclipse.persistence.jpa</artifactId>
  <version>2.7.9</version>
</dependency>

<dependency>
  <groupId>org.eclipse.persistence</groupId>
  <artifactId>org.eclipse.persistence.extension</artifactId>
  <version>2.7.9</version>
</dependency>
```

```
  <dependency>
    <groupId>org.apache.commons</groupId>
    <artifactId>commons-lang3</artifactId>
    <version>3.12.0</version>
  </dependency>

  <dependency>
    <groupId>com.fasterxml.jackson.datatype</groupId>
    <artifactId>jackson-datatype-jsr310</artifactId>
    <version>2.13.0</version>
  </dependency>

  <dependency>
    <groupId>com.oracle.jdbc</groupId>
    <artifactId>ojdbc8</artifactId>
    <version>12.2.0.1</version>
  </dependency>

  <dependency>
    <groupId>org.projectlombok</groupId>
    <artifactId>lombok</artifactId>
    <version>1.18.22</version>
  </dependency>

  <dependency>
    <groupId>org.springframework.kafka</groupId>
    <artifactId>spring-kafka-test</artifactId>
    <scope>test</scope>
  </dependency>
  <!-- provided -->
  <dependency>
    <groupId>javax.servlet</groupId>
    <artifactId>javax.servlet-api</artifactId>
    <scope>provided</scope>
  </dependency>
  <!-- runtime -->
  <dependency>
    <groupId>com.h2database</groupId>
    <artifactId>h2</artifactId>
    <scope>${dependency.scope}</scope>
  </dependency>
  <!-- optional -->
  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-devtools</artifactId>
    <optional>true</optional>
  </dependency>
  <!-- oauth artifact -->
  <dependency>
    <groupId>oal.oracle.apps.ic</groupId>
    <artifactId>oauth-security</artifactId>
    <version>1.0.0</version>
  </dependency>

  <dependency>
    <groupId>com.oracle.oci.sdk</groupId>
    <artifactId>oci-java-sdk-vault</artifactId>
  </dependency>

  <dependency>
    <groupId>com.oracle.oci.sdk</groupId>
    <artifactId>oci-java-sdk-secrets</artifactId>
  </dependency>
</dependencies>

<build>
  <plugins>
    <plugin>
      <groupId>pl.project13.maven</groupId>
      <artifactId>git-commit-id-plugin</artifactId>
      <version>4.0.1</version>
```

```
  <executions>
    <execution>
      <phase>validate</phase>
      <goals>
        <goal>revision</goal>
      </goals>
    </execution>
  </executions>
  <configuration>
    <dateFormat>yyyy-MM-dd'T'HH:mm:ssZ</dateFormat>
    <dotGitDirectory>${project.basedir}/.git</dotGitDirectory>
    <offline>true</offline>
  </configuration>
</plugin>

<plugin>
  <groupId>org.apache.maven.plugins</groupId>
  <artifactId>maven-jar-plugin</artifactId>
  <version>3.2.0</version>
  <configuration>
    <finalName>${project.artifactId}-${version.number}</finalName>
    <archive>
      <manifestEntries>
        <Implementation-Version>${version.number}</Implementation-Version>
      </manifestEntries>
    </archive>
  </configuration>
</plugin>

<plugin>
  <groupId>org.owasp</groupId>
  <artifactId>dependency-check-maven</artifactId>
  <version>5.3.2</version>
  <configuration>
    <!-- Skip artifacts not bundled in distribution (provided scope) -->
    <skipProvidedScope>true</skipProvidedScope>
  </configuration>
  <executions>
    <execution>
      <goals>
        <goal>check</goal>
      </goals>
    </execution>
  </executions>
</plugin>

<plugin>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-maven-plugin</artifactId>
  <executions>
    <execution>
      <goals>
        <goal>build-info</goal>
      </goals>
    </execution>
  </executions>
  <configuration>
    <additionalProperties>
      <version>${version.number}</version>
    </additionalProperties>
  </configuration>
</plugin>

<plugin>
  <groupId>org.apache.maven.plugins</groupId>
  <artifactId>maven-checkstyle-plugin</artifactId>
  <version>3.1.1</version>
  <dependencies>
    <dependency>
      <groupId>com.puppycrawl.tools</groupId>
      <artifactId>checkstyle</artifactId>
```

```
            <version>8.16</version>
          </dependency>
        </dependencies>
        <executions>
          <execution>
            <phase>process-sources</phase>
            <goals>
              <goal>check</goal>
            </goals>
          </execution>
        </executions>
        <configuration>
          <consoleOutput>true</consoleOutput>
          <failsOnError>true</failsOnError>
          <configLocation>https://oitap.oracle.com/artifactory/oal-maven-snapshot-local/oal-
checkstyle.xml
          </configLocation>
        </configuration>
      </plugin>

      <plugin>
        <groupId>org.jacoco</groupId>
        <artifactId>jacoco-maven-plugin</artifactId>
        <version>0.8.5</version>
        <executions>
          <execution>
            <id>prepare-agent</id>
            <goals>
              <goal>prepare-agent</goal>
            </goals>
          </execution>
          <execution>
            <id>report</id>
            <phase>prepare-package</phase>
            <goals>
              <goal>report</goal>
            </goals>
          </execution>
          <execution>
            <id>post-unit-test</id>
            <phase>test</phase>
            <goals>
              <goal>report</goal>
            </goals>
            <configuration>
              <!-- Sets the output directory for the code coverage report. -->
              <outputDirectory>target/jacoco</outputDirectory>
            </configuration>
          </execution>
          <execution>
            <id>jacoco-check</id>
            <phase>test</phase>
            <goals>
              <goal>check</goal>
            </goals>
            <configuration>
              <rules>
                <rule implementation="org.jacoco.maven.RuleConfiguration">
                  <element>BUNDLE</element>
                  <limits>
                    <limit implementation="org.jacoco.report.check.Limit">
                      <counter>INSTRUCTION</counter>
                      <value>COVEREDRATIO</value>
                      <minimum>0</minimum>
                    </limit>
                  </limits>
                </rule>
              </rules>
            </configuration>
          </execution>
        </executions>
```

```
      </plugin>

      <plugin>
        <groupId>org.apache.maven.plugins</groupId>
        <artifactId>maven-javadoc-plugin</artifactId>
        <configuration>
          <doclint>none</doclint>
          <failOnError>false</failOnError>
        </configuration>
      </plugin>

      <plugin>
        <groupId>org.apache.maven.plugins</groupId>
        <artifactId>maven-surefire-plugin</artifactId>
        <version>3.0.0-M5</version>
        <configuration>
          <systemPropertyVariables>
            <listener>oal.oracle.apps.ic.template.infrastructure.messaging.KafkaRunListener<
/listener>
            <spring.cloud.bootstrap.enabled>false</spring.cloud.bootstrap.enabled>
          </systemPropertyVariables>
        </configuration>
      </plugin>

      <plugin>
        <groupId>com.spotify</groupId>
        <artifactId>dockerfile-maven-plugin</artifactId>
        <version>1.4.13</version>
        <configuration>
          <repository>${docker.image.prefix}/${project.artifactId}</repository>
        </configuration>
      </plugin>

      <plugin>
        <groupId>org.apache.maven.plugins</groupId>
        <artifactId>maven-dependency-plugin</artifactId>
        <executions>
          <execution>
            <id>unpack</id>
            <phase>package</phase>
            <goals>
              <goal>unpack</goal>
            </goals>
            <configuration>
              <artifactItems>
                <artifactItem>
                  <groupId>${project.groupId}</groupId>
                  <artifactId>${project.artifactId}</artifactId>
                  <version>${project.version}</version>
                </artifactItem>
              </artifactItems>
            </configuration>
          </execution>
        </executions>
      </plugin>
    </plugins>
  </build>

  <repositories>
    <repository>
      <id>central</id>
      <url>https://oitap.oracle.com/artifactory/oal-maven-snapshot-local</url>
      <snapshots>
        <enabled>false</enabled>
      </snapshots>
    </repository>
  </repositories>

  <pluginRepositories>
    <pluginRepository>
      <id>central</id>
```

```
      <name>artifactory</name>
      <url>https://oitap.oracle.com/artifactory/oal-maven-snapshot-local</url>
      <snapshots>
        <enabled>false</enabled>
      </snapshots>
    </pluginRepository>
  </pluginRepositories>

  <reporting>
    <plugins>
      <plugin>
        <groupId>org.jacoco</groupId>
        <artifactId>jacoco-maven-plugin</artifactId>
        <reportSets>
          <reportSet>
            <reports>
              <!-- select non-aggregate reports -->
              <report>report</report>
            </reports>
          </reportSet>
        </reportSets>
      </plugin>
    </plugins>
  </reporting>
</project>
```

- **Add Docker file**
  - Sample docker file content

Sample Docker file

```
FROM iad.ocir.io/oalprod/jdk11-oal:11.0.3
RUN groupadd --gid 1001 oracle
RUN useradd -g oracle -u 1000 oracleuser
ARG DEPENDENCY=target/dependency
#Copy Jars
COPY ${DEPENDENCY}/BOOT-INF/lib /app/lib
#Copy maven stuff
COPY ${DEPENDENCY}/META-INF /app/META-INF
#Copy all application classes
COPY ${DEPENDENCY}/BOOT-INF/classes /app
ENTRYPOINT java $JVM_OPTS -cp app:app/lib/* oal.oracle.apps.ic.admin.Application
#Enable this entrypoint to run the container on local profile
#ENTRYPOINT ["java","-Dspring.cloud.bootstrap.enabled=false", "-Dspring.profiles.
active=local","-cp","app:app/lib/*","oal.oracle.apps.ic.admin.Application"]
```

- **Add YAML files**
  - Add Deployment.yaml, Service.yaml, service-monitor.yaml files under resources folder

Sample Deployment.yaml

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: oal-admin-service
  labels:
    app: oal-admin-service
spec:
  replicas: 2
  selector:
    matchLabels:
      app: oal-admin-service
  template:
    metadata:
      labels:
        app: oal-admin-service
```

```
      annotations:
        sidecar.istio.io/inject: "false"
    spec:
      securityContext:
        runAsNonRoot: true
        fsGroup: 1001
      containers:
        - name: oal-admin-service
          securityContext:
            runAsNonRoot: true
            runAsGroup: 1001
            runAsUser: 1000
          image: iad.ocir.io/oalprod/oalcn-ms-admin-service:{{commit}}
          imagePullPolicy: Always
          env:
            # Define the environment variable for profile
            - name: SPRING_PROFILES_ACTIVE
              valueFrom:
                configMapKeyRef:
                  # The ConfigMap containing the value you want to assign to
SPRING_PROFILES_ACTIVE
                  name: oic-config-map
                  # Specify the key associated with the value
                  key: spring.profiles.active
            # Define the environment variable for ocid
            - name: OIC_MS_OCID
              valueFrom:
                secretKeyRef:
                  # The secret containing the value you want to assign to OCID
                  name: oic-secrets
                  # Specify the key associated with the value
                  key: oic.ms.ocid
            # Define the environment variable for env context
            - name: NODE_IP
              valueFrom:
                fieldRef:
                  fieldPath: spec.nodeName
            - name: POD_NAME
              valueFrom:
                fieldRef:
                  fieldPath: metadata.name
            - name: POD_NAMESPACEDB_USER
              valueFrom:
                fieldRef:
                  fieldPath: metadata.namespace
            - name: POD_IP
              valueFrom:
                fieldRef:
                  fieldPath: status.podIP
            - name: JVM_OPTS
              value: "-XX:MaxRAMPercentage=50 -XX:MaxMetaspaceSize=168M -XX:
CompressedClassSpaceSize=64M -XX:ReservedCodeCacheSize=64M -Xss585K -XX:-TieredCompilation"
          resources:
            limits:
              memory: "768Mi"
              cpu: 750m
            requests:
              memory: "400Mi"
              cpu: 350m
      imagePullSecrets:
        - name: regcred
```

Sample service.yaml file

```
apiVersion: v1
kind: Service
metadata:
  labels:
    app: oal-admin-service
  annotations:
    # Scrape from /metrics endpoint
    prometheus.io/scrape: "true"
    prometheus.io/port: "8080"
  name: oal-admin-service
spec:
  ports:
    - protocol: TCP
      port: 8080
      targetPort: 8080
      name: http
  selector:
    app: oal-admin-service
```

Sampel Service-monitor.yaml file

```
apiVersion: monitoring.coreos.com/v1
kind: ServiceMonitor
metadata:
  name: oal-admin-service
  namespace: oic-ms-{{environment}}
  labels:
    release: prom-oic-ms-{{environment}}
spec:
  endpoints:
    - interval: 15s
      path: /oalapp/services/msadmin/actuator/prometheus
      port: http
  jobLabel: oal-admin-service
  namespaceSelector:
    matchNames:
      - oic-ms-{{environment}}
  selector:
    matchLabels:
      app: oal-admin-service
```

- **Application properties**
  - Add application related properties to application.properties file for initial testing.

Sample properties for initial setup

```
spring.application.name=initial-application
server.port=8080
spring.main.banner-mode=off
server.servlet.context-path=/initial/setup/example

# Datasource configuration
spring.datasource.hikari.minimum-idle=4
spring.datasource.driver-class-name=org.h2.Driver
spring.datasource.username=
spring.datasource.password=

spring.cloud.config.enabled=false

# Actuator Endpoints
management.endpoints.web.exposure.include=*
management.endpoint.health.show-details=always

# JPA Config
service.audit.user=flupldr-user
spring.jpa.open-in-view=false
# https://www.eclipse.org/eclipselink/api/2.7/org/eclipse/persistence/config
/PersistenceUnitProperties.html
spring.jpa.properties.eclipselink.weaving=static
spring.jpa.properties.eclipselink.ddl-generation=create-tables
spring.jpa.properties.eclipselink.logging.level=INFO
spring.jpa.generate-ddl=true
spring.jpa.show-sql=true
```

- **Test file**
  - Add Test class for initial Application.java file under test package

Sample test class

```
package com.initial.setup.example;
import org.junit.jupiter.api.Test;
import org.springframework.boot.test.context.SpringBootTest;
@SpringBootTest
class InitialSetUpApplicationTests {

        @Test
        void contextLoads() {
        }
}
```

- **Build and Deploy for correctness**
  - Build the application by mvn and then run the application locally for correctness.

Sample logs in console on running the app

```
2022-09-12 20:34:32.172  INFO 80611 --- [           main] .s.d.r.c.
RepositoryConfigurationDelegate : Bootstrapping Spring Data JPA repositories in DEFAULT
mode.
2022-09-12 20:34:32.187  INFO 80611 --- [           main] .s.d.r.c.
RepositoryConfigurationDelegate : Finished Spring Data repository scanning in 5 ms. Found 0
JPA repository interfaces.
2022-09-12 20:34:32.374  INFO 80611 --- [           main] o.s.cloud.context.scope.
GenericScope     : BeanFactory id=e069f9d8-4c77-3e91-8e95-9340b6ec719d
2022-09-12 20:34:32.728  INFO 80611 --- [           main] o.s.b.w.embedded.tomcat.
TomcatWebServer  : Tomcat initialized with port(s): 8080 (http)
2022-09-12 20:34:32.737  INFO 80611 --- [           main] o.apache.catalina.core.
StandardService   : Starting service [Tomcat]
2022-09-12 20:34:32.737  INFO 80611 --- [           main] org.apache.catalina.core.
StandardEngine  : Starting Servlet engine: [Apache Tomcat/9.0.65]
2022-09-12 20:34:32.864  INFO 80611 --- [           main] o.a.c.c.C.[.[.[/initial
/setup]           : Initializing Spring embedded WebApplicationContext
2022-09-12 20:34:32.864  INFO 80611 --- [           main] w.s.c.
ServletWebServerApplicationContext : Root WebApplicationContext: initialization completed
in 1546 ms
2022-09-12 20:34:33.099  INFO 80611 --- [           main] com.zaxxer.hikari.
HikariDataSource       : HikariPool-1 - Starting...
2022-09-12 20:34:33.203  INFO 80611 --- [           main] com.zaxxer.hikari.
HikariDataSource       : HikariPool-1 - Start completed.
2022-09-12 20:34:33.248  INFO 80611 --- [           main] o.hibernate.jpa.internal.util.
LogHelper  : HHH000204: Processing PersistenceUnitInfo [name: default]
2022-09-12 20:34:33.299  INFO 80611 --- [           main] org.hibernate.
Version                    : HHH000412: Hibernate ORM core version 5.6.10.Final
2022-09-12 20:34:33.418  INFO 80611 --- [           main] o.hibernate.annotations.common.
Version   : HCANN000001: Hibernate Commons Annotations {5.1.2.Final}
2022-09-12 20:34:33.507  INFO 80611 --- [           main] org.hibernate.dialect.
Dialect            : HHH000400: Using dialect: org.hibernate.dialect.H2Dialect
2022-09-12 20:34:33.645  INFO 80611 --- [           main] o.h.e.t.j.p.i.
JtaPlatformInitiator       : HHH000490: Using JtaPlatform implementation: [org.hibernate.
engine.transaction.jta.platform.internal.NoJtaPlatform]
2022-09-12 20:34:33.652  INFO 80611 --- [           main] j.
LocalContainerEntityManagerFactoryBean : Initialized JPA EntityManagerFactory for
persistence unit 'default'
2022-09-12 20:34:34.305  INFO 80611 --- [           main] o.s.b.a.e.web.
EndpointLinksResolver      : Exposing 15 endpoint(s) beneath base path '/actuator'
2022-09-12 20:34:34.397  INFO 80611 --- [           main] o.s.b.w.embedded.tomcat.
TomcatWebServer  : Tomcat started on port(s): 8080 (http) with context path '/initial/setup'
2022-09-12 20:34:34.420  INFO 80611 --- [           main] c.initial.setup.
InitialSetUpApplication  : Started InitialSetUpApplication in 3.812 seconds (JVM running
for 5.081)
2022-09-12 20:34:34.633  INFO 80611 --- [-192.168.100.12] o.a.c.c.C.[.[.[/initial
/setup]           : Initializing Spring DispatcherServlet 'dispatcherServlet'
2022-09-12 20:34:34.633  INFO 80611 --- [-192.168.100.12] o.s.web.servlet.
DispatcherServlet        : Initializing Servlet 'dispatcherServlet'
2022-09-12 20:34:34.635  INFO 80611 --- [-192.168.100.12] o.s.web.servlet.
DispatcherServlet        : Completed initialization in 1 ms
```

- Application code set-up
  - **Infrastructure package set-up**
    - under infrastructure package create below folder and add respective files in each folder

1. **messaging**
   package for all Kafka related configurations
   a. KafkaConfig.java -- Configuration details for Kafka Integration. add ConcurrentKafkaListenerContainerFactory, kafkaTemplate to this class
   b. InstantDeSerializer.java, InstantSerializer.java -- used for serialization and deSerialization of Instant
2. **persistence**
   package for all DB related configurations
   a. DataSourceConfig.java -- Custom DataSource Configuration, create method dataSource for DB
   b. JpaConfig.java -- JPA Configuration

c. BooleanToStringConverter.java -- Converter to be used for converting Char to Boolean and vice-versa. Used for DB columns
d. AuditorAwareImpl.java -- This is used by Spring JPA to inject the user info into all @CreatedBy and @LastModifiedBy annotated fields in @Entity.
e. TestDataLoader.java -- to load data into local db when debugging/ testing the application locally.

3. **security**
   package for all security/ authentication configurations
   a. JWToken.java
   b. JWTokenConfig.java
   c. JWTokenException.java
   d. JWTokenValidator.java
   e. AuthenticationHandler.java
   f. Credentials.java
   g. CredentialsProvider.java
   h. OAuth2AuthenticationHandler.java
   i. OCIVaultCredentialsProvider.java
4. **shared**
   package with all required files that are shared across the application
   a. CachingConfig.java
   b. ProfileAndCondition.java
   c. RequestContext.java -- ThreadLocal based Context holder for every Request
5. **web**
   package for rest api filters and configurations
   a. WebApplicationConfig.java -- Configuration class for initialising Web Application Context Beans.
   b. ErrorResponse.java -- Object representing an Error to be sent to the API caller incase of an Error.
   c. LogRequestInterceptor.java -- used to log the requests that are coming into the Service.
   d. RequestContextInterceptor.java -- Interceptor to initialise the RequestContext for every Http Request and clear it after processing.
   e. RestResponseEntityExceptionHandler.java -- Global Exception Handler for Rest APIs
   f. ValidateRequestHeaderInterceptor.java -- Interceptor to validate the Header Attributes of the request.
6. **concurrent**
   package with thread related files
   a. MDCThreadPoolTaskExecutor.java -- ThreadPoolTaskExecutor that propagates MDC context from calling thread to executor thread
7. **monitor**
   package for Grafana related configurations
   a. MetricService.java -- initial metric service with MeterRegistry for capturing Grafana metric

- **Domain package set-up**
-

under Domain package, create below folders

1. **model**
   package that contains entities and repository
   a. under this package, we will place all entities/ repositories needed for application
2. **shared**
   package that contains the shared files across entities
   under shared package, add below files that are shared across entities
   a. AbstractBaseEntity.java -- this contains the columns like createdBy, lastUpdatedBy, lastUpdatedDate, creationDate which are common across all entities
   b. AbstractEntity.java -- this contains above columns along with version
   c. interface - Entity.java -- An Entity is a unique thing and is capable of being changed continuously over a long period of time. It has a unique identity and is mutable.
   d. interface - ValueObject.java -- Value objects compare by the values of their attributes, they don't have an identity
   e. interface - DomainEvent.java -- Domain Event to capture an occurrence of something that happened in the domain
   f. DomainEventPublisher.java -- publisher for domain event

- **Application package set-up**
  - This package contains all application related files.
  - create ApplicationConfiguration.java file under application package and this is used as Central Application Configuration
- **Properties files and log4j file**
  - Add application environment specific properties files under resources folder
  - Add log4j-paas.xml file with logging configurations under resource folder

Sample log4j file

```
<Configuration status="info">
  <Appenders>
    <Kafka name="KafkaAppender" topic="${bundle:application:oal.app.monitor.topic.name}"
      syncSend="false">
      <JSONLayout locationInfo="true" properties="true" charset="ISO-8859-1" compact="true">
        <KeyValuePair key="profile" value="${env:SPRING_PROFILES_ACTIVE}"/>
        <KeyValuePair key="service" value="${bundle:application:spring.application.name}"/>
        <KeyValuePair key="server" value="${env:NODE_IP}:${env:POD_IP}:${env:
POD_NAME}:${env:POD_NAMESPACE}"/>
      </JSONLayout>
      <Property name="bootstrap.servers"
        value="${bundle:application-${env:SPRING_PROFILES_ACTIVE}:spring.kafka.bootstrap-
servers}"/>
      <Property name="client.id"
        value="${bundle:application:spring.application.name}"/>
    </Kafka>
  </Appenders>
  <Loggers>
    <Logger name="org.apache.kafka" level="WARN"/> <!-- avoid recursive logging -->
    <Logger name="oal.oracle.apps.ic.admin.application"  level="INFO" additivity="false"> <!
-- log only adm srv info logs-->
      <AppenderRef ref="KafkaAppender"/>
    </Logger>
    <Logger name="org.springframework.kafka.listener.KafkaMessageListenerContainer" level="
INFO" additivity="false"> <!-- Spring kafka logs for checking kafka re-balancing-->
      <AppenderRef ref="KafkaAppender"/>
    </Logger>
    <Root level="ERROR">
      <AppenderRef ref="KafkaAppender"/>
    </Root>
  </Loggers>
</Configuration>
```

    - **Run the application**
        - Build the application after all the above configurations and then deploy the application locally.
- Code repository set-up
    - Create new git repository for this newly developed micro-service with application name as the git repository name
    - Create 3 new branches - master (private branch), cn-development, cn-build
    - Once branches are created, create dev branch from cn-build and then merge the complete new service set-up code to cn-build after L1, L2 code reviews (branching strategy KT is here)
- REMO set-up for deployments
    - Check with Release team ask for creating new remo for this new service for deploying the application to environments
    - Once the remo release set-up and new service set-up code is merged to cn-build branch, then trigger build to dev environments by using the remo.