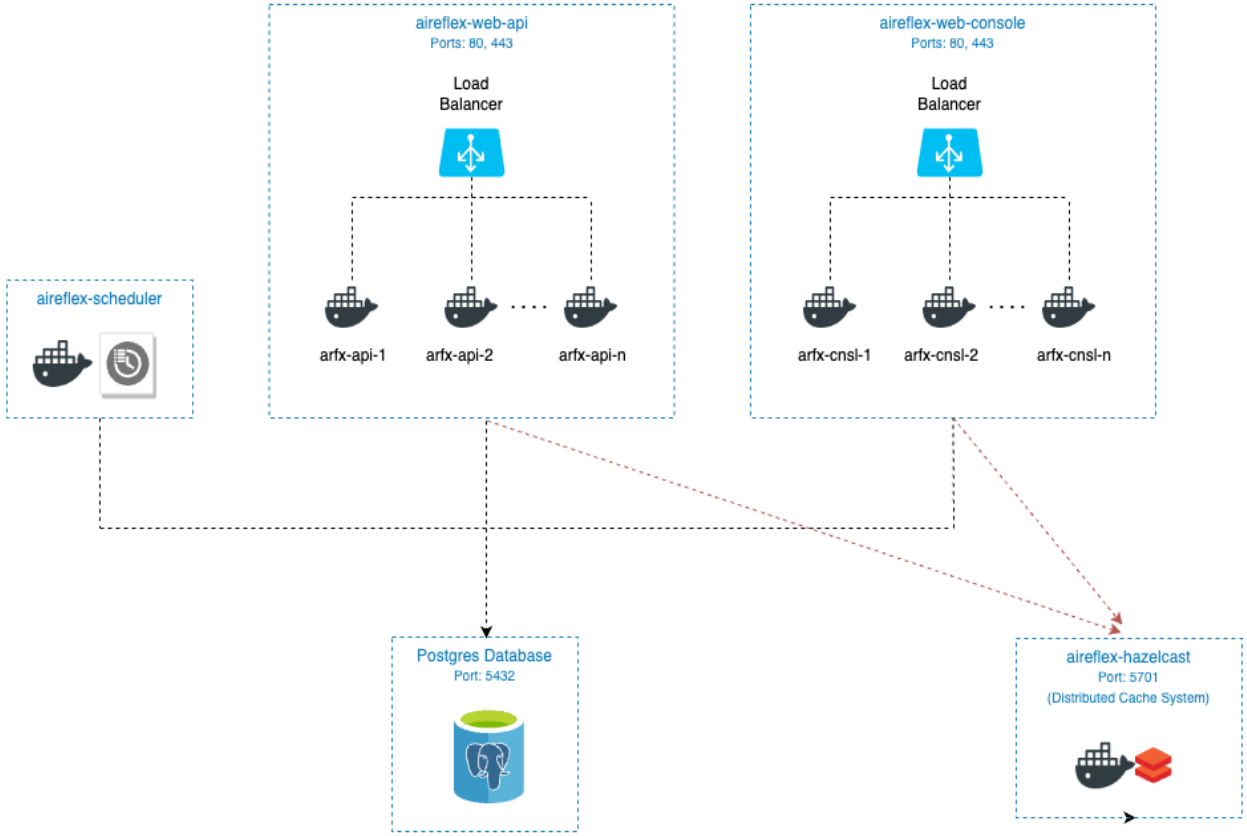
AI-Reflex Servisleri

AI-Reflex tarafından kullanılan 4 ana Docker imajı bulunmaktadır. Her biri sistem içinde belirli görevleri yerine getirir. Bu servisler şunlardır;

- aireflex-hazelcast
- aireflex-scheduler
- aireflex-web-api
- aireflex-web-console

Sistemin genel topolojisi ve bu servislerin birbirleriyle nasıl etkileşimde bulunduğunu gösteren basit bir şematik aşağıda sunulmuştur:



---

1. aireflex-hazelcast

Bu servis, sistem genelinde performansı artırmak adına önbellekleme işlemlerinden sorumludur.
Temel görevleri;

- Veritabanı sorgularının gereksiz yere tekrarlanmasını önlemek ve veri erişim sürelerini kısaltmak

- aireflex-web-console servisi üzerinden yapılan kural kitap değişikliklerini, diğer web-console servislerine topic yapısını kullanarak iletmek

Docker konteyneri ayağa kaldırılırken, "hazelcast.xml" adlı bir konfigürasyon dosyasının konteynere bind edilmesi gereklidir. Bu dosya, "aiReflex_imdg" isimli dizin altında yer alır.

Ayrıca servis, 5701 portu üzerinden çalışır ve environment variable değikenlerinin tanımlanması gerekmektedir.
Aşağıda bir docker-compose.yml dosyasında aireflex-hazelcast servisine ilişkin örnek bir yapılandırma gösterilmektedir:

```
services:
    aireflex-hazelcast:
        container_name: aireflex-hazelcast
        image: registry.fraud.com/aireflex/aireflex-hazelcast:v2.8.1.230710
        environment:
            - JAVA_OPTS=-Dhazelcast.config=/opt/hazelcast/config_ext/hazelcast.xml
            - CLASSPATH=/opt/hazelcast/CLASSPATH_EXT/*
        volumes:
            - /opt/aireflex/aiReflex_imdg:/opt/hazelcast/config_ext
        ports:
            - "5701:5701"
```

---

2. aireflex-scheduler

aireflex-scheduler, Aireflex sisteminin batch işlemlerini yöneten servistir. Belirli bir zaman aralığında veritabanından bir rapor çıkartılması, bir dizi hesaplama yapılması veya sistemdeki bir durumun düzenli olarak kontrol edilmesi (health-check vb.) gibi işlemler batch olarak çalıştırılır.
aireflex-scheduler servisi, WAR dosyası formatında build edildiğinden dolayı, belirli bir port (8081) üzerinde çalışacak şekilde yapılandırılması gerekmektedir. Ayrıca volume olarak host makinesindeki ***aireflex_conf*** dizini container'a bind edilmiştir. Örnek docker-compose.yml dosyasındaki yapılandırma şöyledir;

```
services:
    aireflex-scheduler:
        container_name: aireflex-scheduler
        image: registry.fraud.com/aireflex/aireflex-scheduler:v2.8.1.230710
        volumes:
            - /opt/aireflex/aireflex_conf:/usr/local/share/aireflex
        ports:
            - "8081:8081"
```

3.aireflex-web-api

aireflex-web-api, kullanıcı arayüzüne sahip olmayan bir servistir. Bu servis, AI-Reflex sistemi ile dış servisler arasındaki veri iletişimini yönetir. Dış sistemlerin AI-Reflex sistemine veri göndermeleri veya mevcut verilere geri bildirimde(feedback - verifyFraud - verifyNotFraud)

bulunmaları gerektiğinde, bu işlemler aireflex-web-api üzerinden gerçekleştirilir. Entegrasyon için ayrıntılı bilgiye bu link üzerinden ulaşabilirsiniz.

Bu servis aktif-aktif mimaride çalışacak şekilde kurgulanmıştır. Yani, birden fazla instance aynı anda çalışabilir ve birbirlerini destekler. Yoğun veri trafiği durumlarında, bu servisten yeni konteynerler oluşturulabilir. Bu sayede servisler dinamik bir şekilde ölçeklendirilebilir ve sistemin performansı korunabilir.

Docker konteynerini başlatırken, "aireflex_conf" isimli dizin bind edilmelidir. Bu dizin ve içeriği hakkındaki detaylı bilgi bölüm sonunda belirtilmiştir..

Aşağıda bir docker-compose.yml dosyasında aireflex-web-api servisine ilişkin örnek bir yapılandırma gösterilmektedir:

```
services:
    aireflex-web-api:
        container_name: aireflex-web-api
        image: dev-registry.fraud.com/aireflex/aireflex-web-api:v2.8.1.230707
        volumes:
            - /opt/aireflex/aireflex_conf:/usr/local/share/aireflex
            - /opt/aireflex/aireflex_conf/setenv.sh:/usr/local/tomcat/bin/setenv.sh        #optional
            - /opt/aireflex/aireflex_conf/server.xml:/usr/local/tomcat/conf/server.xml     #optional
            - /opt/aireflex/aireflex_conf/aireflex.jks:/usr/local/tomcat/conf/aireflex.jks #optional
        ports:
            - "80:8080"
            - "443:8443"
            - "8080:8080"
```

setenv.sh dosyası imaj içerisinde varsayılan olarak gelmektedir. Eğer JVM bellek ayarlarını özelleştirmek isterseniz, ayrı bir "setenv.sh" dosyası oluşturup aşapıdaki koyu renkli alanları değiştirmeniz sonrasında bu dosyayı konteynere bind etmeniz gerekmektedir. İmajda varsayılan olarak mevcut olan "setenv.sh" dosyasının içeriği aşağıdaki gibidir:

```
# setenv.sh
export JAVA_OPTS="-Xms2048M -Xmx2048M -server -Dfile.encoding=UTF-8
-Dlogback.configurationFile=/usr/local/tomcat/conf/logback.xml --add-opens
jdk.compiler/com.sun.tools.javac.file=ALL-UNNAMED --add-opens java.base/java.net=ALL-UNNAMED"
export CATALINA_OPTS="-Xms2048M -Xmx2048M"
```

Tomcat üzerinde SSL sertifika ayarları yapmak isterseniz, bir ".jks" uzantılı dosyayı bind etmeniz ve bu dosyanın yolunu "server.xml" dosyasında belirtmeniz gerekmektedir. Bu işlem için, "server.xml" dosyasını da konteynerin Tomcat ayarlarına bind etmeniz gerekmektedir.

---

4. aireflex-web-console

Bu servis, kullanıcıların AI-Reflex sistemine web tabanlı bir arayüz üzerinden erişimini sağlar. Kullanıcılar bu arayüz üzerinden çeşitli işlemler gerçekleştirebilirler. Örneğin; işlemleri

görüntüleme, kural motorunda kurallar tanımlama, event-stream'leri yapılandırma ve liste yönetimi yapma gibi..

Yine aktif-aktif mimaride çalışacak şekilde kurgulanmıştır. Kullanıcısı sayısındaki artışa bağlı olarak servisler ölçeklendirilebilir.

Bu Docker imajı üzerinden bir konteyner oluşturulduğunda, otomatik olarak bir "default user" tanımlanmış olarak gelir. Web arayüzden login ekranına giderek aşağıdaki kullanıcı bilgileri ile giriş yapabilirsiniz.

username: system
password: fdklmnjxga

Docker konteynerini başlatırken, "aireflex_conf" isimli dizin bind edilmelidir. Bu dizin ve içeriği hakkındaki detaylı bilgi bölüm sonunda belirtilmiştir..

Aşağıda bir docker-compose.yml dosyasında aireflex-web-console servisine ilişkin örnek bir yapılandırma gösterilmektedir:

```
services:
    aireflex-web-console:
        container_name: aireflex-web-console
        image: dev-registry.fraud.com/aireflex/aireflex-web-console:v2.8.1.230707
        volumes:
            - /opt/aireflex/aireflex_conf:/usr/local/share/aireflex
            - /opt/aireflex/aireflex_conf/setenv.sh:/usr/local/tomcat/bin/setenv.sh        #optional
            - /opt/aireflex/aireflex_conf/server.xml:/usr/local/tomcat/conf/server.xml     #optional
            - /opt/aireflex/aireflex_conf/aireflex.jks:/usr/local/tomcat/conf/aireflex.jks #optional
        ports:
            - "80:8080"
            - "443:8443"
            - "8080:8080"
```

setenv.sh dosyası imaj içerisinde varsayılan olarak gelmektedir. Eğer JVM bellek ayarlarını özelleştirmek isterseniz, ayrı bir "setenv.sh" dosyası oluşturup aşapıdaki koyu renkli alanları değiştirmeniz sonrasında bu dosyayı konteynere bind etmeniz gerekmektedir. İmajda varsayılan olarak mevcut olan "setenv.sh" dosyasının içeriği aşağıdaki gibidir:

```
# setenv.sh
export JAVA_OPTS="-Xms2048M -Xmx2048M -server -Dfile.encoding=UTF-8
-Dlogback.configurationFile=/usr/local/tomcat/conf/logback.xml --add-opens
jdk.compiler/com.sun.tools.javac.file=ALL-UNNAMED --add-opens java.base/java.net=ALL-UNNAMED"
export CATALINA_OPTS="-Xms2048M -Xmx2048M"
```

Tomcat üzerinde SSL sertifika ayarları yapmak isterseniz, bir ".jks" uzantılı dosyayı bind etmeniz ve bu dosyanın yolunu "server.xml" dosyasında belirtmeniz gerekmektedir. Bu işlem için, "server.xml" dosyasını da konteynerin Tomcat ayarlarına bind etmeniz gerekmektedir.

 # aireflex_conf

"aireflex_conf" dizini uygulamanın çeşitli özelliklerinin konfigürasyonunu sağlayan üç properties dosyası ve bu dosyalarla ilişkili sub dizinlerden oluşmaktadır. **Bu dizin Docker servislerinin ulaşabileceği ve paylaşılan bir NFS diske yerleştirilmelidir. Bu düzenlemeye ihtiyaç duyulmasının sebebi, birden fazla aktif servisin bir dosya yükleme işlemi gerçekleştirdiğinde, diğer servislerin de bu yüklenen dosyaları görmesini sağlamaktır.**

aireflex_conf dizini içerisinde yer alan dosyalar ve dizinler şunlardır;
aireflex_conf

- application.properties (FILE)
- datasource.properties (FILE)
- fcase-integration.properties (FILE)
- converters (FOLDER)
- custom-actions (FOLDER)
- static-lists (FOLDER)
- ml (FOLDER)
- offline-data-loader (FOLDER)
- file-reports (FOLDER)
- rule-book-repo (FOLDER)

1. application.properties

application.properties: Bu dosyada, uygulamanın genel ayarları ile 'aireflex-web-api' ve 'aireflex-web-console' servislerinin entegre olacağı Hazelcast servisine ilişkin bilgiler bulunur. Örnek bir içerik aşağıdaki gibi olmalıdır.

```
# In-Memory Data Grid (IMDG) settings
imdg.enable=true
imdg.memberServers=aireflex-hazelcast:5701

# Converter jar file directory path
converter.jars.dirPath=/usr/local/share/aireflex/converters

# Custom Action jar file directory path
customAction.jars.dirPath=/usr/local/share/aireflex/custom-actions

# Static List CSV file storage directory path
str.staticList.fileUpload.dirPath=/usr/local/share/aireflex/static-lists/str

# Machine Learning model directory path
ml.model.dirPath=/usr/local/share/aireflex/ml

# Rule Condition Check Null Value
rule.condition.checkNullValue=true

# Rule Engine Enable Data Status Setting
ruleEngine.enableDataStatusSetting=true

# Offline Data Loader directory path
offline.data.loader.dirPath=/usr/local/share/aireflex/offline-data-loader

# File Archive Export directory path
```

```
fileArchive.export.dirPath=/usr/local/share/aireflex/file-reports

# Rule Book Repo directory path
ruleBook.repo.dirPath=/usr/local/share/aireflex/rule-book-repo

# Auth Mode Setting: LDAP, ALL
authenticationMode.setting=all
```

- imdg.enable: IMDG'nin etkinleştirilip etkinleştirilmeyeceğini belirler. Varsayılan olarak TRUE set edilmiştir.
- imdg.memberServers: Hazelcast instance sunucu adreslerini belirtir.
- converter.jars.dirPath: Converter jar dosyalarının saklandığı dizin yolunu belirtir.
- customAction.jars.dirPath: Custom Action jar dosyalarının saklandığı dizin yolunu belirtir.
- str.staticList.fileUpload.dirPath: String statik listeleri için CSV dosyalarının saklandığı dizin yolunu belirtir.
- ml.model.dirPath: Makine öğrenmesi modellerinin saklandığı dizin yolunu belirtir.
- rule.condition.checkNullValue: Kural koşullarında NULL değer kontrolünün etkinleştirilip etkinleştirilmeyeceğini belirler. Varsayılan olarak TRUE set edilmiştir.
- ruleEngine.enableDataStatusSetting: Bu ayar, Kural Motorunun "Mark as Fraud" ve "Mark as Not Fraud" işlemlerini etkinleştirip etkinleştirmeyeceğini belirler. Bu, belirli işlemlerin dolandırıcılık veya dolandırıcılık olmadığı şeklinde işaretlenmesini sağlar.
- offline.data.loader.dirPath: Bu ayar, toplu veri işlemleri için kullanılan veri dosyalarının depolandığı dizinin yolunu tanımlar. Yani bu dosyalar çevrimdışı veri yükleme işlemlerinde kullanılır.
- fileArchive.export.dirPath: Dosya arşivi dışa aktarmalarının saklandığı dizin yolunu belirtir.
- ruleBook.repo.dirPath: Kural kitabı dosyalarının saklandığı dizin yolunu belirtir.
- authenticationMode.setting: Kimlik doğrulama modunu belirtir (örneğin, LDAP, SSO veya ALL). SSO entegrasyonu ile beraber devreye alınacaktır. Şu an için bu property ALL olarak set edilmelidir. (TODO)

2. datasource.properties

Bu dosyada, PostgreSQL veritabanına ait bilgiler yer alır. Örnek bir içerik aşağıdaki gibi olmalıdır. Koyu renkli alanlarda db ortamı bilgileri girilmelidir.

```
dataSourceClassName=org.postgresql.ds.PGSimpleDataSource
dataSource.username=DB_USER_NAME
dataSource.password=DB_PASSWORD
dataSource.databaseName=aireflex_db
dataSource.schemaName=adts_schema
dataSource.portNumber=DB_PORT
dataSource.serverName=DB_IP_ADDRESS
dataSource.maximumPoolSize=55
dataSource.idleTimeout=60000
dataSource.connectionInitSql=SET SEARCH_PATH TO adts_schema
```

3. fcase-integration.properties

Bu dosya, Fcase sistemine entegrasyon için gerekli URL ve token bilgilerini içerir. Örnek bir içerik aşağıdaki gibi olmalıdır:

fcase.api.url=https://dev.fcase.io/TmgRestApi/api
fcase.api.auth.token=eyJhbGciOiJIUzI1NiJ9.eyJodHRwOlwvXC90bWcuaWhzLmNvbS50ciI6dHJ1ZSwic3ViIjoiMTQ1NDQyMTcwNjgzNDcxMTU1NSIsIlJvbGVOYW1lIjoiRkRTX0NBU0VfUE9TVEVSIiwibmJmIjoxNTA4MTcxODI4LCJBbGxvd2VkTGlzdCI6eyJDUkVBVEVfVElDS0VUX0ZEUyI6IlwvY3VzdG9tZXJcL3RpY2tldFwvY3JlYXRlVmlhRmRzIn0sIkZpcnN0TmFtZSI6ImFkbWluIiwiTGFzdExvZ2luIjoiMjAxNy0xMC0xNiAxOTozNzowOC4zMzkiLCJpc3MiOiJ0bWcuaWhzLmNvbS50ciIsIkxhc3ROYW1lIjoiYWRtaW4iLCJleHAiOjE1MDgyNTgyMjgsIkZkc0Nhc2VDcmVhdGlvblJlcXVlc3QiOnRydWV9.CCvKfL0i7TQlUVE74X2PHHEnWYP8FsmUwxPxxfS5-j4