My Way of Android CI CD

- [x] Fastlane ve GitHub Actions Kullanımı: * Fastlane ve GitHub Actions gibi CI/CD araçlarını kullanarak bir mobil uygulamanın CI/CD sürecini oluşturmalı ve uygulamayı otomatik olarak dağıtım kanallarına gönderebilmelidir.
- [x] Farklı Environments ve Branch Yapıları: * Farklı ortamlar (environments) için dağıtım yapılmalıdır. Development ve master branch'leri üzerinden yapılandırmalar yapılmalı, her branch için uygun ayarlamalar gerçekleştirilmelidir.
- [x] Uygulamanın Dağıtımı: * Uygulama en az iki ayrı versiyon halinde dağıtılmalıdır:
  - Production versiyonu
  - Test versiyonu
- [x] Unit ve UI Test Entegrasyonu: * CI sürecinde Unit ve UI Testleri entegre edilmeli, test koşulları sağlanmadığı durumda pipeline fail olmalı

1. Preparation

1.1 Branches

- Create a `master` branch for the signed release version and a `develop` branch for the unsigned test version.
- **Release Builds:** Use the Google Play Store Beta channel for distribution.
- **Test Builds:**
  - Generate an unsigned `.apk` for debugging and upload it to GitHub artifacts.
  - Distribute a signed debug version via Google Play's internal testing channel. This build is triggered using the `deploy-internal-test*` tag.

1.2 Keystore file

- Create `keystore` file.

```
keytool -genkey -v -keystore my-release-key.jks -keyalg RSA -keysize 2048 -
validity 10000 -alias my-key-alias
```

- Encrypt the keystore file using your preferred tool:

```
openssl base64 -in my-release-key.jks -out my-release-key.jks.base64
```

1.3 Gradle settings

Add `signingConfigs` and `buildTypes` blocks to the app-level Gradle file. Use environment variables for sensitive configuration values. These variables can be set in the terminal:

```
export KEYSTORE_FILE="/path/to/my-release-key.jks"
export KEYSTORE_PASSWORD="***"
export KEY_ALIAS="my-key-alias"
export KEY_PASSWORD="***"
```

Example Gradle configuration:

```
signingConfigs {
    create("release") {
        storeFile = file("../my-release-key.jks")
        storePassword = System.getenv("KEYSTORE_PASSWORD")
        keyAlias = System.getenv("KEY_ALIAS")
        keyPassword = System.getenv("KEY_PASSWORD")
    }
}

buildTypes {
    getByName("release") {
        isMinifyEnabled = false
        proguardFiles(
            getDefaultProguardFile("proguard-android-optimize.txt"),
            "proguard-rules.pro"
        )
        signingConfig = signingConfigs.getByName("release")
    }

    getByName("debug") {
        signingConfig = signingConfigs.getByName("debug")
    }
}
```

1.4 Initialize `Fastlane`

- Install Fastlane: Follow the instructions at https://docs.fastlane.tools/getting-started/android/setup/

- Create a Google Play service account, download the authentication JSON file, and encrypt it.
- Configure the `Fastfile` with the following lanes:

```
lane :test do
gradle(task: "test")
end

lane :debug_build do
  gradle(
    task: 'assembleDebug',
    build_type: 'Debug'
  )
end


lane :upload_to_internal_test do
  keystore_path = File.expand_path("../my-release-key.jks", __dir__)
  gradle(
    task: 'bundle',
    build_type: 'Release',
    properties: {
      "android.injected.signing.store.file" => keystore_path,
      "android.injected.signing.store.password" => ENV["KEYSTORE_PASSWORD"],
      "android.injected.signing.key.alias" => ENV["KEY_ALIAS"],
      "android.injected.signing.key.password" => ENV["KEY_PASSWORD"]
    }
  )
  upload_to_play_store(
      track: "internal",
      release_status: 'draft',
      skip_upload_apk: true,
      skip_upload_metadata: true,
      skip_upload_images: true,
      skip_upload_screenshots: true
  )
end

lane :upload_to_beta do
  keystore_path = File.expand_path("../my-release-key.jks", __dir__)
  gradle(
    task: 'bundle',
    build_type: 'Release',
    properties: {
      "android.injected.signing.store.file" => keystore_path,
      "android.injected.signing.store.password" => ENV["KEYSTORE_PASSWORD"],
      "android.injected.signing.key.alias" => ENV["KEY_ALIAS"],
      "android.injected.signing.key.password" => ENV["KEY_PASSWORD"]
    }
  )
  upload_to_play_store(
    skip_upload_apk: true
    track: 'beta'
  )
end

lane :upload_to_production do
  keystore_path = File.expand_path("../my-release-key.jks", __dir__)
  gradle(
    task: 'bundle',
    build_type: 'Release',
    properties: {
      "android.injected.signing.store.file" => keystore_path,
      "android.injected.signing.store.password" => ENV["KEYSTORE_PASSWORD"],
      "android.injected.signing.key.alias" => ENV["KEY_ALIAS"],
      "android.injected.signing.key.password" => ENV["KEY_PASSWORD"]
    }
  )
  upload_to_play_store
end
```

1.5 Add environment variables

To securely store sensitive data, use GitHub Secrets:

1. Go to your repository's **Settings** > **Secrets and variables** > **Actions**.
2. Add the following secrets:

KEY	VALUE
KEYSTORE_FILE	Content of ./my-release-key.jks.base64
KEYSTORE_PASSWORD	Your password
KEY_ALIAS	Your created alias `my-key-alias`
KEY_PASSWORD	Your password

---

2. Test build

2. 1 Unsigned test build as `apk`

This build is triggered on every push to the `develop` branch and uploads an unsigned APK as a GitHub artifact.

```yaml
name: Build test apk

on:
  push:
    branches:
      - develop

jobs:
  build:
    runs-on: ubuntu-latest
    steps:
      - name: Checkout
        uses: actions/checkout@v4

      - name: Set Up JDK
        uses: actions/setup-java@v4
        with:
          distribution: 'zulu'
          java-version: 17

      - name: Setup Gradle
        uses: gradle/actions/setup-gradle@v3

      - name: Run Tests
        run: ./gradlew test

      - name: Build the project
        run: ./gradlew clean assembleDebug

      - name: Upload APK
        uses: actions/upload-artifact@v3
        with:
          name: test-apk
          path: app/build/outputs/apk/release/*.apk
```

2.2 Signed test build with Google Play's Internal Test

Triggered by tags matching `deploy-internal-test*`, this workflow builds a signed APK and deploys it to Google Play's internal test track.

```yaml
name: Build signed test apk

on:
  push:
    tags:
      - "deploy-internal-test*"

jobs:
  build:
    runs-on: ubuntu-latest
    steps:
      - name: Checkout
        uses: actions/checkout@v4

      - name: Set Up Ruby
        uses: ruby/setup-ruby@v1
        with:
          ruby-version: '3.1'

      - name: Install Dependencies
        run: |
          gem install bundler
          bundle install

      - name: Install OpenSSL
        run: sudo apt-get install -y openssl

      - name: Decode and Write Keystore File
        run: |
          echo "${{ secrets.KEYSTORE_FILE }}" | openssl base64 -d -out my-
release-key.jks
        env:
          KEYSTORE_FILE: ${{ secrets.KEYSTORE_FILE }}

      - name: Verify Keystore File
        run: |
          if [ -s my-release-key.jks ]; then
          echo "Keystore file created successfully."
          ls -l my-release-key.jks
          else
          echo "Keystore file is empty or missing!" && exit 1
          fi

      - name: Debug Environment Variables
        run: |
          echo "KEYSTORE_PASSWORD is set"
          echo "KEY_ALIAS is set"
          echo "KEY_PASSWORD is set"
        env:
          KEYSTORE_PASSWORD: ${{ secrets.KEYSTORE_PASSWORD }}
          KEY_ALIAS: ${{ secrets.KEY_ALIAS }}
          KEY_PASSWORD: ${{ secrets.KEY_PASSWORD }}

      - name: Check Keystore File Existence
        run: ls -l ./my-release-key.jks

      - name: Check Keystore Path
        run: realpath ./my-release-key.jks

      - name: Decrypt Play Store Credentials File
        run: |
          echo "${{ secrets.GOOGLE_PLAY_SERVICE_ACCOUNT }}" | openssl base64
-d -out ./fastlane/playstore_credentials.json

      - name: Install Fastlane
        run: sudo gem install fastlane

      - name: Test
        run: fastlane test

      - name: Deploy to Internal Testing
        run: fastlane internal_test
        env:
          KEYSTORE_PASSWORD: ${{ secrets.KEYSTORE_PASSWORD }}
          KEY_ALIAS: ${{ secrets.KEY_ALIAS }}
          KEY_PASSWORD: ${{ secrets.KEY_PASSWORD }}
```

---

3. Deploy to Google Play Store

This workflow deploys the app to the Google Play Store's beta or production track, triggered by pushes to the `master` branch.

```yaml
name: Deploy to Play Store

on:
  push:
    branches:
      - master

jobs:
  deploy:
    runs-on: ubuntu-latest

    steps:
      - name: Checkout Repository
        uses: actions/checkout@v4

      - name: Set Up Ruby
        uses: ruby/setup-ruby@v1
        with:
          ruby-version: '3.1'

      - name: Install Dependencies
        run: |
          gem install bundler
          bundle install

      - name: Install OpenSSL
        run: sudo apt-get install -y openssl

      - name: Decode and Write Keystore File
        run: |
          echo "${{ secrets.KEYSTORE_FILE }}" | openssl base64 -d -out my-
release-key.jks
        env:
          KEYSTORE_FILE: ${{ secrets.KEYSTORE_FILE }}

      - name: Decrypt Play Store Credentials File
        run: |
          echo "${{ secrets.GOOGLE_PLAY_SERVICE_ACCOUNT }}" | openssl base64
-d -out ./fastlane/playstore_credentials.json

      - name: Install Fastlane
        run: sudo gem install fastlane

      - name: Test
        run: fastlane test
      - name: Deploy to Internal Testing
        run: fastlane upload_to_beta
        env:
          KEYSTORE_PASSWORD: ${{ secrets.KEYSTORE_PASSWORD }}
          KEY_ALIAS: ${{ secrets.KEY_ALIAS }}
          KEY_PASSWORD: ${{ secrets.KEY_PASSWORD }}
```
