1DreamUnited CI/CD Setup Guide

Continuous Integration & Deployment

This guide covers setting up automated build, test, and deployment pipelines for all 1DreamUnited platforms using various CI/CD services.

Overview

The 1DreamUnited multi-platform ecosystem requires coordinated CI/CD across:
- **Mobile**: React Native (iOS/Android) with Expo/EAS
- **Desktop**: Electron (Windows/macOS/Linux)
- **Roku**: BrightScript channel deployment

GitHub Actions Setup

Complete Multi-Platform Workflow

```yaml
# .github/workflows/multi-platform-ci-cd.yml
name: 1DreamUnited Multi-Platform CI/CD

on:
  push:
    branches: [ main, develop ]
  pull_request:
    branches: [ main ]
  release:
    types: [ published ]

env:
  NODE_VERSION: '18'
  EXPO_TOKEN: ${{ secrets.EXPO_TOKEN }}
  APPLE_ID: ${{ secrets.APPLE_ID }}
  APPLE_ID_PASSWORD: ${{ secrets.APPLE_ID_PASSWORD }}

jobs:
  # Mobile CI/CD
  mobile:
    name: Mobile (React Native)
    runs-on: ubuntu-latest
    defaults:
      run:
        working-directory: ./mobile/react_native

    steps:
      - name:   Setup repo
        uses: actions/checkout@v3

      - name:   Setup Node
        uses: actions/setup-node@v3
        with:
          node-version: ${{ env.NODE_VERSION }}
          cache: npm
          cache-dependency-path: mobile/react_native/package-lock.json

      - name:   Setup Expo and EAS
        uses: expo/expo-github-action@v8
        with:
          expo-version: latest
          eas-version: latest
          token: ${{ secrets.EXPO_TOKEN }}

      - name:   Install dependencies
        run: npm ci

      - name:   Lint code
        run: npm run lint

      - name:   Run tests
        run: npm run test -- --coverage --watchAll=false

      - name:   Upload coverage
        uses: codecov/codecov-action@v3
        with:
          directory: ./mobile/react_native/coverage

      - name:   Type check
        run: npm run type-check

      - name:   Build development
```

```
        if: github.event_name == 'pull_request'
        run: |
          npx expo prebuild --clean --no-install
          echo "Development build completed"

      - name:   Build production (iOS)
        if: github.event_name == 'release'
        run: eas build --platform ios --non-interactive --wait

      - name:   Build production (Android)
        if: github.event_name == 'release'
        run: eas build --platform android --non-interactive --wait

      - name:   Submit to stores
        if: github.event_name == 'release'
        run: |
          eas submit --platform ios --non-interactive
          eas submit --platform android --non-interactive

  # Desktop CI/CD
  desktop:
    name: Desktop (Electron)
    strategy:
      matrix:
        os: [ubuntu-latest, windows-latest, macos-latest]
    runs-on: ${{ matrix.os }}
    defaults:
      run:
        working-directory: ./desktop/electron

    steps:
      - name:   Setup repo
        uses: actions/checkout@v3

      - name:   Setup Node
        uses: actions/setup-node@v3
        with:
          node-version: ${{ env.NODE_VERSION }}
          cache: npm
          cache-dependency-path: desktop/electron/package-lock.json

      - name:   Install dependencies
        run: npm ci

      - name:   Lint code
        run: npm run lint

      - name:   Run tests
        run: npm test

      - name:   Build application
        run: npm run package

      - name:   Create distributables
        if: github.event_name == 'release'
        run: npm run make

      - name:   Code sign (macOS)
        if: matrix.os == 'macos-latest' && github.event_name == 'release'
        env:
          CSC_LINK: ${{ secrets.MAC_CERTIFICATE }}
          CSC_KEY_PASSWORD: ${{ secrets.MAC_CERTIFICATE_PASSWORD }}
          APPLE_ID: ${{ secrets.APPLE_ID }}
```

```
          APPLE_ID_PASSWORD: ${{ secrets.APPLE_ID_PASSWORD }}
        run: npm run dist:mac

      - name:   Code sign (Windows)
        if: matrix.os == 'windows-latest' && github.event_name == 'release'
        env:
          CSC_LINK: ${{ secrets.WIN_CERTIFICATE }}
          CSC_KEY_PASSWORD: ${{ secrets.WIN_CERTIFICATE_PASSWORD }}
        run: npm run dist:win

      - name:   Build Linux packages
        if: matrix.os == 'ubuntu-latest' && github.event_name == 'release'
        run: npm run dist:linux

      - name:   Upload artifacts
        if: github.event_name == 'release'
        uses: actions/upload-artifact@v3
        with:
          name: desktop-${{ matrix.os }}
          path: |
            desktop/electron/out/make/**/*
            desktop/electron/dist/**/*

  # Roku CI/CD
  roku:
    name: Roku Channel
    runs-on: ubuntu-latest
    defaults:
      run:
        working-directory: ./roku

    steps:
      - name:   Setup repo
        uses: actions/checkout@v3

      - name:   Package Roku channel
        run: |
          zip -r Dr33MTV_Channel_${{ github.sha }}.zip . \
            -x "*.git*" "*.DS_Store" "deploy.sh" "README.md" "*.zip"

      - name:   Validate manifest
        run: |
          if [ ! -f "manifest" ]; then
            echo "Error: manifest file not found"
            exit 1
          fi
          echo "Manifest validation passed"

      - name:   Upload channel package
        uses: actions/upload-artifact@v3
        with:
          name: roku-channel
          path: roku/Dr33MTV_Channel_*.zip

      - name:   Deploy to test device
        if: github.event_name == 'push' && github.ref == 'refs/heads/develop'
        env:
          ROKU_DEV_TARGET: ${{ secrets.ROKU_DEV_TARGET }}
          ROKU_DEV_PASSWORD: ${{ secrets.ROKU_DEV_PASSWORD }}
        run: |
          if [ -n "$ROKU_DEV_TARGET" ] && [ -n "$ROKU_DEV_PASSWORD" ]; then
            curl -s -S -F "mysubmit=Install" \
                 -F "archive=@Dr33MTV_Channel_${{ github.sha }}.zip" \
```

```
                -u "rokudev:$ROKU_DEV_PASSWORD" \
                "http://$ROKU_DEV_TARGET/plugin_install"
              echo "Channel deployed to test device"
            else
              echo "Roku deployment skipped - credentials not configured"
            fi

  # Release Management
  release:
    name: Create Release
    if: github.event_name == 'release'
    needs: [mobile, desktop, roku]
    runs-on: ubuntu-latest

    steps:
      - name:   Setup repo
        uses: actions/checkout@v3

      - name:   Download all artifacts
        uses: actions/download-artifact@v3

      - name:   Create release packages
        run: |
          mkdir -p release-packages

          # Package desktop builds
          tar -czf release-packages/1DreamUnited-Desktop-Windows.tar.gz desktop-
windows-latest/
          tar -czf release-packages/1DreamUnited-Desktop-macOS.tar.gz desktop-macos-
latest/
          tar -czf release-packages/1DreamUnited-Desktop-Linux.tar.gz desktop-ubuntu-
latest/

          # Copy Roku package
          cp roku-channel/*.zip release-packages/

      - name:   Upload to release
        uses: softprops/action-gh-release@v1
        with:
          files: release-packages/*
          generate_release_notes: true
```

Mobile-Specific Workflow

```yaml
# .github/workflows/mobile-ci.yml
name: Mobile CI

on:
  push:
    paths: ['mobile/**']
  pull_request:
    paths: ['mobile/**']

jobs:
  mobile-test:
    runs-on: ubuntu-latest
    defaults:
      run:
        working-directory: ./mobile/react_native

    steps:
      - uses: actions/checkout@v3

      - name: Setup Node.js
        uses: actions/setup-node@v3
        with:
          node-version: '18'
          cache: 'npm'
          cache-dependency-path: mobile/react_native/package-lock.json

      - name: Setup Expo
        uses: expo/expo-github-action@v8
        with:
          expo-version: latest
          token: ${{ secrets.EXPO_TOKEN }}

      - name: Install dependencies
        run: npm ci

      - name: Run tests
        run: npm test -- --coverage --watchAll=false

      - name: Lint
        run: npm run lint

      - name: Type check
        run: npm run type-check

      - name: Build for preview
        if: github.event_name == 'pull_request'
        run: |
          eas build --platform all --profile preview --non-interactive --wait
```

GitLab CI/CD

Complete Pipeline Configuration

```yaml
# .gitlab-ci.yml
stages:
  - test
  - build
  - deploy
  - release

variables:
  NODE_VERSION: "18"
  DOCKER_DRIVER: overlay2

# Mobile Pipeline
mobile:test:
  stage: test
  image: node:18
  before_script:
    - cd mobile/react_native
    - npm ci
  script:
    - npm run lint
    - npm run type-check
    - npm test -- --coverage --watchAll=false
  coverage: '/Lines\s*:\s*(\d+\.\d+)%/'
  artifacts:
    reports:
      coverage_report:
        coverage_format: cobertura
        path: mobile/react_native/coverage/cobertura-coverage.xml
  only:
    changes:
      - mobile/**/*

mobile:build:
  stage: build
  image: node:18
  before_script:
    - cd mobile/react_native
    - npm ci
    - npm install -g @expo/cli eas-cli
  script:
    - expo login -u $EXPO_USERNAME -p $EXPO_PASSWORD
    - eas build --platform all --profile production --non-interactive
  only:
    - main
    - tags

# Desktop Pipeline
desktop:test:
  stage: test
  image: node:18
  before_script:
    - cd desktop/electron
    - npm ci
  script:
    - npm run lint
    - npm test
  only:
    changes:
      - desktop/**/*

desktop:build:linux:
  stage: build
```

```
  image: node:18
  before_script:
    - cd desktop/electron
    - npm ci
  script:
    - npm run package
    - npm run make
  artifacts:
    paths:
      - desktop/electron/out/
    expire_in: 1 week
  only:
    - main
    - tags

desktop:build:windows:
  stage: build
  tags:
    - windows
  before_script:
    - cd desktop/electron
    - npm ci
  script:
    - npm run dist:win
  artifacts:
    paths:
      - desktop/electron/dist/
    expire_in: 1 week
  only:
    - main
    - tags

# Roku Pipeline
roku:test:
  stage: test
  image: alpine:latest
  before_script:
    - apk add --no-cache zip
  script:
    - cd roku
    - |
      if [ ! -f "manifest" ]; then
        echo "Error: manifest file not found"
        exit 1
      fi
    - zip -r Dr33MTV_Channel_test.zip . -x "*.git*" "deploy.sh" "README.md"
  artifacts:
    paths:
      - roku/Dr33MTV_Channel_test.zip
    expire_in: 1 day
  only:
    changes:
      - roku/**/*

roku:deploy:
  stage: deploy
  image: alpine:latest
  before_script:
    - apk add --no-cache curl zip
  script:
    - cd roku
    - zip -r Dr33MTV_Channel.zip . -x "*.git*" "deploy.sh" "README.md"
    - |
```

```
      if [ -n "$ROKU_DEV_TARGET" ] && [ -n "$ROKU_DEV_PASSWORD" ]; then
        curl -s -S -F "mysubmit=Install" \
             -F "archive=@Dr33MTV_Channel.zip" \
             -u "rokudev:$ROKU_DEV_PASSWORD" \
             "http://$ROKU_DEV_TARGET/plugin_install"
      fi
  only:
    - develop

# Release Pipeline
release:
  stage: release
  image: alpine:latest
  before_script:
    - apk add --no-cache tar gzip
  script:
    - mkdir -p release-packages
    - tar -czf release-packages/1DreamUnited-Complete-$CI_COMMIT_TAG.tar.gz .
  artifacts:
    paths:
      - release-packages/
  only:
    - tags
```

Azure DevOps Pipelines

Multi-Platform Pipeline

```yaml
# azure-pipelines.yml
trigger:
  branches:
    include:
      - main
      - develop
  paths:
    include:
      - mobile/*
      - desktop/*
      - roku/*

pr:
  branches:
    include:
      - main

variables:
  nodeVersion: '18.x'

stages:
- stage: Test
  displayName: 'Test All Platforms'
  jobs:
  - job: MobileTest
    displayName: 'Mobile Tests'
    pool:
      vmImage: 'ubuntu-latest'
    condition: or(contains(variables['Build.SourceVersionMessage'], 'mobile'),
eq(variables['Build.Reason'], 'PullRequest'))
    steps:
    - task: NodeTool@0
      inputs:
        versionSpec: $(nodeVersion)
    - script: |
        cd mobile/react_native
        npm ci
        npm run lint
        npm run type-check
        npm test -- --coverage --watchAll=false
      displayName: 'Mobile CI'
    - task: PublishCodeCoverageResults@1
      inputs:
        codeCoverageTool: 'Cobertura'
        summaryFileLocation: 'mobile/react_native/coverage/cobertura-coverage.xml'

  - job: DesktopTest
    displayName: 'Desktop Tests'
    strategy:
      matrix:
        Linux:
          imageName: 'ubuntu-latest'
        Windows:
          imageName: 'windows-latest'
        macOS:
          imageName: 'macOS-latest'
    pool:
      vmImage: $(imageName)
    condition: or(contains(variables['Build.SourceVersionMessage'], 'desktop'),
eq(variables['Build.Reason'], 'PullRequest'))
    steps:
    - task: NodeTool@0
```

```yaml
      inputs:
        versionSpec: $(nodeVersion)
    - script: |
        cd desktop/electron
        npm ci
        npm run lint
        npm test
      displayName: 'Desktop CI'

- stage: Build
  displayName: 'Build All Platforms'
  dependsOn: Test
  condition: and(succeeded(), eq(variables['Build.SourceBranch'], 'refs/heads/main'))
  jobs:
  - job: MobileBuild
    displayName: 'Mobile Build'
    pool:
      vmImage: 'ubuntu-latest'
    steps:
    - task: NodeTool@0
      inputs:
        versionSpec: $(nodeVersion)
    - script: |
        cd mobile/react_native
        npm ci
        npm install -g @expo/cli eas-cli
        expo login -u $(EXPO_USERNAME) -p $(EXPO_PASSWORD)
        eas build --platform all --profile production --non-interactive
      displayName: 'Build Mobile Apps'
      env:
        EXPO_TOKEN: $(EXPO_TOKEN)

  - job: DesktopBuild
    displayName: 'Desktop Build'
    strategy:
      matrix:
        Linux:
          imageName: 'ubuntu-latest'
        Windows:
          imageName: 'windows-latest'
        macOS:
          imageName: 'macOS-latest'
    pool:
      vmImage: $(imageName)
    steps:
    - task: NodeTool@0
      inputs:
        versionSpec: $(nodeVersion)
    - script: |
        cd desktop/electron
        npm ci
        npm run package
        npm run make
      displayName: 'Build Desktop App'
    - task: PublishBuildArtifacts@1
      inputs:
        pathToPublish: 'desktop/electron/out'
        artifactName: 'desktop-$(Agent.OS)'

  - job: RokuBuild
    displayName: 'Roku Build'
    pool:
      vmImage: 'ubuntu-latest'
```

```
    steps:
    - script: |
        cd roku
        zip -r Dr33MTV_Channel_$(Build.BuildNumber).zip . \
          -x "*.git*" "deploy.sh" "README.md"
      displayName: 'Package Roku Channel'
    - task: PublishBuildArtifacts@1
      inputs:
        pathToPublish: 'roku/Dr33MTV_Channel_$(Build.BuildNumber).zip'
        artifactName: 'roku-channel'
```

Docker-Based CI/CD

Multi-Stage Dockerfile

```
# Dockerfile.ci
FROM node:18-alpine AS base
WORKDIR /app
RUN apk add --no-cache git python3 make g++ zip curl

# Mobile build stage
FROM base AS mobile
COPY mobile/react_native/package*.json ./mobile/react_native/
RUN cd mobile/react_native && npm ci
COPY mobile/react_native ./mobile/react_native/
RUN cd mobile/react_native && \
    npm run lint && \
    npm run type-check && \
    npm test -- --watchAll=false

# Desktop build stage
FROM base AS desktop
COPY desktop/electron/package*.json ./desktop/electron/
RUN cd desktop/electron && npm ci
COPY desktop/electron ./desktop/electron/
RUN cd desktop/electron && \
    npm run lint && \
    npm test && \
    npm run package

# Roku build stage
FROM alpine:latest AS roku
RUN apk add --no-cache zip
WORKDIR /app
COPY roku ./roku/
RUN cd roku && \
    zip -r Dr33MTV_Channel.zip . \
      -x "*.git*" "deploy.sh" "README.md"

# Final stage
FROM alpine:latest AS final
RUN apk add --no-cache tar gzip
WORKDIR /app
COPY --from=mobile /app/mobile ./mobile/
COPY --from=desktop /app/desktop ./desktop/
COPY --from=roku /app/roku ./roku/
RUN tar -czf 1DreamUnited-Complete.tar.gz .
```

Docker Compose for CI

```yaml
# docker-compose.ci.yml
version: '3.8'

services:
  mobile-ci:
    build:
      context: .
      dockerfile: Dockerfile.ci
      target: mobile
    volumes:
      - ./mobile:/app/mobile
      - mobile_node_modules:/app/mobile/react_native/node_modules
    environment:
      - CI=true
    command: |
      sh -c "
        cd mobile/react_native &&
        npm run lint &&
        npm run type-check &&
        npm test -- --watchAll=false --coverage
      "

  desktop-ci:
    build:
      context: .
      dockerfile: Dockerfile.ci
      target: desktop
    volumes:
      - ./desktop:/app/desktop
      - desktop_node_modules:/app/desktop/electron/node_modules
    environment:
      - CI=true
    command: |
      sh -c "
        cd desktop/electron &&
        npm run lint &&
        npm test &&
        npm run package
      "

  roku-ci:
    build:
      context: .
      dockerfile: Dockerfile.ci
      target: roku
    volumes:
      - ./roku:/app/roku
    command: |
      sh -c "
        cd roku &&
        zip -r Dr33MTV_Channel.zip . -x '*.git*' 'deploy.sh' 'README.md'
      "

volumes:
  mobile_node_modules:
  desktop_node_modules:
```

Security and Secrets Management

Required Secrets

Mobile (Expo/EAS)

```
# Expo credentials
EXPO_TOKEN=your_expo_token
EXPO_USERNAME=your_expo_username
EXPO_PASSWORD=your_expo_password

# Apple credentials
APPLE_ID=your_apple_id
APPLE_ID_PASSWORD=app_specific_password
APPLE_TEAM_ID=your_team_id

# Google Play credentials
GOOGLE_SERVICE_ACCOUNT_KEY=base64_encoded_key
```

Desktop (Electron)

```
# Code signing certificates
MAC_CERTIFICATE=base64_encoded_p12
MAC_CERTIFICATE_PASSWORD=certificate_password
WIN_CERTIFICATE=base64_encoded_p12
WIN_CERTIFICATE_PASSWORD=certificate_password

# Notarization
APPLE_ID=your_apple_id
APPLE_ID_PASSWORD=app_specific_password
```

Roku

```
# Development deployment
ROKU_DEV_TARGET=192.168.1.xxx
ROKU_DEV_PASSWORD=your_dev_password

# Channel store
ROKU_DEVELOPER_EMAIL=your_email
ROKU_DEVELOPER_PASSWORD=your_password
```

Secrets Configuration Examples

GitHub Secrets

```
# Add secrets via GitHub CLI
gh secret set EXPO_TOKEN --body "your_token_here"
gh secret set APPLE_ID --body "your_apple_id"
gh secret set MAC_CERTIFICATE --body "$(base64 -i certificate.p12)"
```

GitLab Variables

```yaml
# .gitlab-ci.yml variables section
variables:
  EXPO_TOKEN:
    description: "Expo authentication token"
    protected: true
    masked: true
```

Monitoring and Notifications

Slack Integration

```yaml
# Slack notification step
- name: Notify Slack
  if: always()
  uses: 8398a7/action-slack@v3
  with:
    status: ${{ job.status }}
    channel: '#ci-cd'
    webhook_url: ${{ secrets.SLACK_WEBHOOK }}
    fields: repo,message,commit,author,action,eventName,ref,workflow
```

Email Notifications

```yaml
# Email notification step
- name: Send Email
  if: failure()
  uses: dawidd6/action-send-mail@v3
  with:
    server_address: smtp.gmail.com
    server_port: 587
    username: ${{ secrets.EMAIL_USERNAME }}
    password: ${{ secrets.EMAIL_PASSWORD }}
    subject: "Build Failed: ${{ github.repository }}"
    body: "Build failed for commit ${{ github.sha }}"
    to: team@1dreamunited.com
```

Deployment Strategies

Blue-Green Deployment

- Maintain two identical production environments
- Switch traffic between environments for zero-downtime deployments
- Rollback capability by switching back

Canary Deployment

- Gradually roll out to percentage of users
- Monitor metrics and user feedback
- Full rollout or rollback based on results

Feature Flags

- Deploy code with features disabled

- Enable features for specific user groups
- A/B testing capabilities

Performance Monitoring

Build Performance Tracking

```
- name: Track Build Time
  run: |
    echo "BUILD_START_TIME=$(date +%s)" >> $GITHUB_ENV
    # ... build steps ...
    BUILD_END_TIME=$(date +%s)
    BUILD_DURATION=$((BUILD_END_TIME - BUILD_START_TIME))
    echo "Build completed in ${BUILD_DURATION} seconds"
```

Artifact Size Monitoring

```
- name: Check Bundle Size
  run: |
    cd mobile/react_native
    npm run build:analyze
    # Compare with previous builds and alert if significant increase
```

Troubleshooting CI/CD

Common Issues

Mobile Build Failures

- **Expo token expired**: Refresh EXPO_TOKEN secret
- **Certificate issues**: Verify Apple/Google credentials
- **Dependency conflicts**: Clear cache and reinstall

Desktop Build Failures

- **Native dependencies**: Ensure proper build tools installed
- **Code signing**: Verify certificate validity and passwords
- **Platform-specific issues**: Check OS-specific requirements

Roku Deployment Issues

- **Network connectivity**: Verify Roku device accessibility
- **Manifest errors**: Validate manifest file syntax
- **Package size**: Ensure channel package is under size limits

Debug Strategies

```
# Enable debug logging
- name: Debug CI Environment
  run: |
    echo "Node version: $(node --version)"
    echo "NPM version: $(npm --version)"
    echo "Working directory: $(pwd)"
    echo "Environment variables:"
    env | grep -E '^(CI|GITHUB|EXPO)' | sort
```

Best Practices

CI/CD Pipeline Design

1. **Fast Feedback**: Run quick tests first
2. **Parallel Execution**: Run independent jobs in parallel
3. **Fail Fast**: Stop pipeline on critical failures
4. **Artifact Management**: Store and version build artifacts
5. **Environment Parity**: Keep CI environment close to production

Security Best Practices

1. **Least Privilege**: Grant minimum required permissions
2. **Secret Rotation**: Regularly rotate authentication tokens
3. **Audit Logs**: Monitor CI/CD activity and access
4. **Dependency Scanning**: Check for vulnerable dependencies
5. **Code Signing**: Sign all production releases

Monitoring and Alerting

1. **Build Status**: Monitor build success/failure rates
2. **Performance Metrics**: Track build times and resource usage
3. **Deployment Health**: Monitor post-deployment application health
4. **User Impact**: Track user-facing metrics after deployments

---

This CI/CD setup provides comprehensive automation for the 1DreamUnited multi-platform ecosystem, ensuring consistent quality and reliable deployments across all platforms.