빌드	Newsnippet 서비스는 개발하는 과정에서 Github Action을 통해 TDD 개발을 할 수 있는 CI 환경을 구축하고자 했습니다. 'develop' 브랜치 하위 브랜치에서 기능 별 생성한 Test Code의 Pass여부에 따라 Remote의 'develop' 브랜치에 대한 접근 가능여부를 Github Action이 자동적으로 판단할 수 있습니다. 이러한 과정을 통해 오류가 나거나 기대하는 값을 반환하지 못하는 코드들을 필터링하여 Remote의 'develop' 브랜치를 쾌적하게 관리할 수 있습니다.		
Gradle	3.7		
Java	17		
Github Action Secrets	DB_URL	DB_USER_NAME	DB_PASSWORD
/workflows/gradle.yml			

```yaml
name: Java CI with Gradle

on:
  push:
    branches: [ "develop" ]
  pull_request:
    branches: [ "develop" ]

jobs:
  build:

    runs-on: ubuntu-latest
    permissions:
      contents: read

    steps:
      - uses: actions/checkout@v4
      - name: Set up JDK
        uses: actions/setup-java@v4
        with:
          java-version: '17'
          distribution: 'corretto'

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

      - name: Build with Gradle
        run: ./gradlew build
        env:
          DB_URL: DB URL = ${{ secrets.DB_URL }}
          DB_USER_NAME: USER = ${{ secrets.DB_USER_NAME }}
          DB_PASSWORD: PSW = ${{ secrets.DB_PASSWORD }}


  dependency-submission:

    runs-on: ubuntu-latest
    permissions:
      contents: write

    steps:
      - uses: actions/checkout@v4
      - name: Set up JDK
        uses: actions/setup-java@v4
        with:
          java-version: '17'
          distribution: 'corretto'
          cache: gradle

      - name: Generate and submit dependency graph
        uses: gradle/actions/dependency-submission@417ae3ccd767c252f5661f1ace9f835f9654f2b5 # v3.1.0
```

배포	Github Action에서 Pass된 접근에 대하여 Github WebHook을 통해 Jenkins의 Pipeline의 설정에 따라 자동 배포할 수 있는 CD환경도 구축하고자 했습니다. Pass된 변경사항들은 Jenkins에서 설정해둔 Docker 이미지 최신화 코드를 통해 Docker hub에 새롭게 갱신되고 컨테이너도 업데이트 되어 변경사항이 자동 반영되어 배포됩니다.	
Docker image	newsnippet_back	newsnippet-front
Worker Node	30001	30000
Container	7777	5173
Service	8001	8000

Jenkins-Pipeline

```
pipeline {
    agent any

    tools {
        gradle 'gradle'
        jdk 'openJDK17'
    }

    environment {
        DOCKERHUB_USERNAME = 'giyeonlee'
        GITHUB_URL = 'https://github.com/Triumers/Newsnippet-Back.git'
    }

    stages {
        stage('Preparation') {
            steps {
                script {
                    sh 'docker --version' // Docker가 설치되어 있는지 확인
                }
            }
        }
        stage('Source Build') {
            steps {
                // 소스파일 체크아웃
                git branch: 'main', url: 'https://github.com/Triumers/Newsnippet-Back.git'

                // 소스 빌드
                // 755권한 필요 (윈도우에서 Git으로 소스 업로드시 권한은 644)
                sh "chmod +x ./gradlew"
                sh "./gradlew build -Pjasypt.encryptor.password='jasypt_password'"
            }
        }
        stage('Container Build') {
            steps {

                // jar 파일 복사
                sh "cp ./build/libs/*.jar ."

                // 컨테이너 빌드 및 업로드
                sh "docker build -t ${DOCKERHUB_USERNAME}/newsnippet_back:latest ."

                // docker hub로 push
                withCredentials([usernamePassword(credentialsId: 'DOCKERHUB_PASSWORD', usernameVariable: 'DOCKERHUB_USER', passwordVariable: 'DOCKERHUB_PASS')]) {
                    sh "echo $DOCKERHUB_PASS | docker login --username $DOCKERHUB_USER --password-stdin"
                    sh "docker push ${DOCKERHUB_USERNAME}/newsnippet_back:latest"
                }
            }
        }
    }
}
```