Настройка GitHub Actions для CI/CD

1. Цель работы

Настроить GitHub Actions для автоматической сборки, тестирования и деплоя проекта. Создать CI/CD pipeline, который будет автоматически собирать проект, запускать тесты и развертывать приложение при каждом изменении кода.

2. Выбранный проект

2.1. Описание проекта

Для выполнения лабораторной работы был выбран проект "Calculator Application" - простое приложение-калькулятор на Java с использованием Maven. Проект реализует базовые математические операции: сложение, вычитание, умножение, деление, возведение в степень и извлечение квадратного корня.

2.3. Основной класс Calculator

```
package com.example.calculator;

public class Calculator {
	public double add(double a, double b) {
		return a + b;
	}
	public double subtract(double a, double b) {
		return a - b;
	}
	public double multiply(double a, double b) {
		return a * b;
	}
	public double divide(double a, double b) {
		if (b == 0) {
			throw new IllegalArgumentException("Деление на ноль невозможно");
		}
		return a / b;
	}
	public double power(double base, double exponent) {
		return Math.pow(base, exponent);
```

```
    }
    public double sqrt(double value) {
        if (value < 0) {
            throw new IllegalArgumentException("Квадратный корень из отрицательного числа невозможен");
        }
        return Math.sqrt(value);
    }
}
```

2.4. Тесты

Проект включает юнит-тесты, проверяющие все математические операции, а также обработку исключительных ситуаций (деление на ноль, квадратный корень из отрицательного числа).

3. Код файла ci-cd.yml

```
name: CI/CD Pipeline

on:
  push:
    branches: [ main, master, develop ]
  pull_request:
    branches: [ main, master ]

jobs:
  build-and-test:
    name: Build and Test
    runs-on: ${{ matrix.os }}
    strategy:
      matrix:
        os: [ubuntu-latest, windows-latest, macos-latest]
        java-version: [11, 17]
```

```
    exclude:
      - os: macos-latest
        java-version: 11

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

- name: Set up JDK ${{ matrix.java-version }}
  uses: actions/setup-java@v4
  with:
    java-version: ${{ matrix.java-version }}
    distribution: 'temurin'
    cache: maven

- name: Build project
  run: mvn clean compile

- name: Run tests
  run: mvn test

- name: Package application
  run: mvn package -DskipTests

- name: Upload artifacts
  uses: actions/upload-artifact@v4
  with:
    name: calculator-jar-${{ matrix.os }}-${{ matrix.java-version }}
    path: target/*.jar
```

```
      retention-days: 7

  deploy:
    name: Deploy Application
    needs: build-and-test
    runs-on: ubuntu-latest
    if: github.ref == 'refs/heads/main' || github.ref == 'refs/heads/master'

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

    - name: Set up JDK 11
      uses: actions/setup-java@v4
      with:
        java-version: '11'
        distribution: 'temurin'
        cache: maven

    - name: Build and package
      run: mvn clean package -DskipTests

    - name: Download all artifacts
      uses: actions/download-artifact@v4
      with:
        path: artifacts

    - name: Deploy via SSH
      uses: appleboy/ssh-action@v1.0.0
```

```
    with:
      host: ${{ secrets.SSH_HOST }}
      username: ${{ secrets.SSH_USERNAME }}
      key: ${{ secrets.SSH_KEY }}
      script: |
        mkdir -p /opt/calculator
        echo "Deployment completed at $(date)"

  - name: Create deployment summary
    run: |
      echo "## Deployment Summary" >> $GITHUB_STEP_SUMMARY
      echo "- Build completed successfully" >> $GITHUB_STEP_SUMMARY
      echo "- Tests passed" >> $GITHUB_STEP_SUMMARY
      echo "- Artifacts uploaded" >> $GITHUB_STEP_SUMMARY
```

4. Описание настроенных джобов

4.1. Триггеры workflow

Workflow запускается автоматически при следующих событиях:

- **push** в ветки: main, master, develop
- **pull_request** в ветки: main, master

4.2. Job: build-and-test

Назначение: Сборка проекта и запуск тестов на разных платформах и версиях Java.

Платформы:

- ubuntu-latest (Linux)
- windows-latest (Windows)
- macos-latest (macOS Intel и Apple Silicon)

Версии Java: 11, 17

Используемые GitHub Actions:

- actions/checkout@v4 - получение кода из репозитория
- actions/setup-java@v4 - настройка Java окружения с кэшированием Maven

- actions/upload-artifact@v4 - загрузка собранных артефактов

Шаги выполнения:

1. Checkout code - получение исходного кода
2. Set up JDK - установка Java и настройка Maven
3. Build project - компиляция проекта (mvn clean compile)
4. Run tests - запуск юнит-тестов (mvn test)
5. Package application - создание JAR файла (mvn package)
6. Upload artifacts - сохранение артефактов для последующего использования

4.3. Job: deploy

Назначение: Развертывание приложения на удаленном сервере после успешной сборки и тестирования.

Условия запуска:

- Запускается только после успешного завершения job build-and-test
- Запускается только при push в ветки main или master

Используемые GitHub Actions:

- actions/checkout@v4 - получение кода
- actions/setup-java@v4 - настройка Java
- actions/download-artifact@v4 - загрузка артефактов из предыдущего job
- appleboy/ssh-action@v1.0.0 - выполнение команд на удаленном сервере через SSH

Шаги выполнения:

1. Checkout code - получение исходного кода
2. Set up JDK 11 - установка Java 11
3. Build and package - сборка проекта
4. Download all artifacts - загрузка артефактов из job build-and-test
5. Deploy via SSH - развертывание на удаленном сервере через SSH
6. Create deployment summary - создание сводки о развертывании

4.4. Зависимости между джобами

Job	Зависит от	Описание зависимости
build-and-test	-	Независимый job, запускается первым
deploy	build-and-test	Запускается только после успешного завершения build-and-test. Использует артефакты, созданные в build-and-test.

4.5. Матричная стратегия сборки

Job build-and-test использует матричную стратегию для параллельной сборки на разных платформах:

- **Операционные системы:** Ubuntu, Windows, macOS
- **Версии Java:** 11, 17
- **Исключения:** macOS с Java 11 исключен из матрицы

Это обеспечивает проверку кроссплатформенной совместимости проекта.

5. Ссылка на репозиторий

Открытый репозиторий с настроенным pipeline:

URL: [https://github.com/BanDit116han/calculator/blob/main/.github/workflows/ci-cd.yml]

5.1. Скриншот репозитория

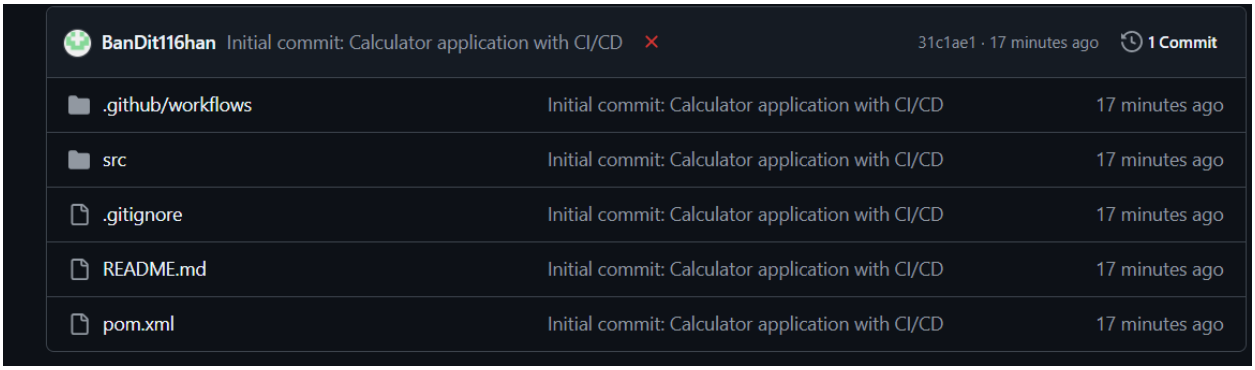

Рисунок 1 - Репозиторий на GitHub

5.2. Скриншот успешного выполнения workflow

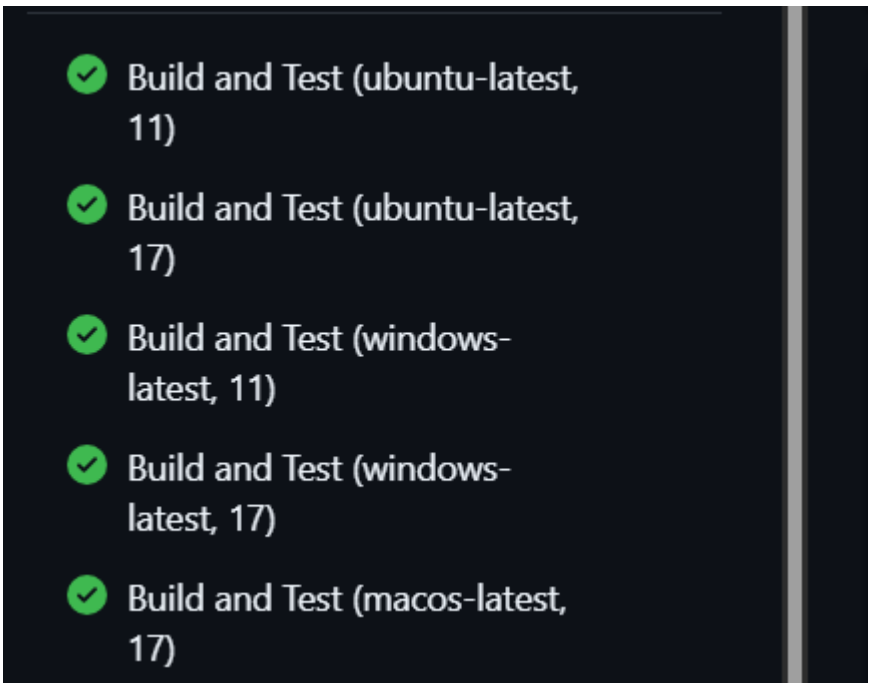

Рисунок 2 - Успешное выполнение CI/CD pipeline

6. Настройка Secrets для деплоя

Для работы job deploy необходимо настроить следующие secrets в настройках репозитория (Settings → Secrets and variables → Actions):

- SSH_HOST - адрес удаленного сервера
- SSH_USERNAME - имя пользователя для SSH подключения
- SSH_KEY - приватный SSH ключ для аутентификации

7. Выводы

В ходе выполнения лабораторной работы было успешно выполнено следующее:

1. Создан проект-калькулятор на Java с использованием Maven
2. Написаны юнит-тесты для проверки всех математических операций
3. Настроен GitHub Actions workflow для автоматической сборки и тестирования
4. Настроена матричная стратегия для проверки кроссплатформенной совместимости (Linux, Windows, macOS)
5. Настроено тестирование на разных версиях Java (11, 17)
6. Настроен автоматический деплой приложения после успешной сборки и тестирования
7. Создан открытый репозиторий на GitHub с настроенным CI/CD pipeline

CI/CD pipeline успешно работает и автоматически собирает, тестирует и развертывает приложение при каждом изменении кода в репозитории.