```yaml
name: Multi-Platform Auto Release

on:
  push:
    branches:
      - main
  workflow_dispatch:
    inputs:
      bump:
        description: "Version bump type"
        required: true
        default: "patch"
        type: choice
        options:
          - patch
          - minor
          - major

jobs:
  version:
    runs-on: ubuntu-latest
    permissions:
      contents: write
    outputs:
      new_tag: ${{ steps.bump.outputs.next }}

    steps:
      - name: Checkout repository
        uses: actions/checkout@v4
        with:
          fetch-depth: 0
          token: ${{ secrets.GITHUB_TOKEN }}

      - name: Configure Git
        run: |
          git config user.name "github-actions[bot]"
          git config user.email "github-actions[bot]@users.noreply.github.com"

      - name: Get latest tag
        id: get_tag
        run: |
          git fetch --tags
          LATEST_TAG=$(git describe --tags --abbrev=0 2>/dev/null || echo "v0.0.0")
          echo "latest=$LATEST_TAG" >> $GITHUB_OUTPUT

      - name: Decide bump type
        id: decide
        run: |
```

```
        if [ "${{ github.event_name }}" = "workflow_dispatch" ]; then
          echo "bump=${{ github.event.inputs.bump }}" >> $GITHUB_OUTPUT
        else
          echo "bump=patch" >> $GITHUB_OUTPUT
        fi

    - name: Bump version
      id: bump
      run: |
        LATEST=${{ steps.get_tag.outputs.latest }}
        BUMP=${{ steps.decide.outputs.bump }}
        BASE=${LATEST#v}
        IFS='.' read -r MAJOR MINOR PATCH <<< "$BASE"

        case $BUMP in
          patch) PATCH=$((PATCH+1)) ;;
          minor) MINOR=$((MINOR+1)); PATCH=0 ;;
          major) MAJOR=$((MAJOR+1)); MINOR=0; PATCH=0 ;;
        esac

        NEXT="v$MAJOR.$MINOR.$PATCH"
        echo "next=$NEXT" >> $GITHUB_OUTPUT
        echo "Next version: $NEXT"

    - name: Commit changes if any
      run: |
        git add .
        git commit -m "chore: automated sync" || echo "No changes"
        git push origin HEAD:${{ github.ref }}

    - name: Create and push tag
      run: |
        NEXT=${{ steps.bump.outputs.next }}
        git tag $NEXT
        git push origin $NEXT

build:
  needs: version
  runs-on: ${{ matrix.os }}
  strategy:
    matrix:
      include:
        # Linux builds
        - os: ubuntu-latest
          platform: linux
          arch: x64
          ext: ""
        - os: ubuntu-latest
          platform: linux
          arch: arm64
```

```
            ext: ""

          # macOS builds
          - os: macos-latest
            platform: darwin
            arch: x64
            ext: ""
          - os: macos-latest
            platform: darwin
            arch: arm64
            ext: ""

          # Windows builds
          - os: windows-latest
            platform: windows
            arch: x64
            ext: ".exe"

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

      - name: Set up Node.js
        uses: actions/setup-node@v4
        with:
          node-version: '20'

      # 🔧 Customize build commands for your project type
      - name: Install dependencies
        run: npm ci

      - name: Build project
        run: npm run build

      # 📦 Package artifacts
      - name: Create platform archive
        shell: bash
        run: |
          VERSION=${{ needs.version.outputs.new_tag }}
          PLATFORM=${{ matrix.platform }}
          ARCH=${{ matrix.arch }}
          ARTIFACT_NAME="myapp-${VERSION}-${PLATFORM}-${ARCH}"

          # Create dist package
          mkdir -p artifacts

          if [ "${{ matrix.platform }}" = "windows" ]; then
            # Windows: create zip
            powershell Compress-Archive -Path dist/* -DestinationPath
"artifacts/${ARTIFACT_NAME}.zip"
```

```
        else
          # Unix: create tar.gz
          tar -czf "artifacts/${ARTIFACT_NAME}.tar.gz" -C dist .
        fi

    # 🚀 Example: Build native binary (Go/Rust/etc.)
    # Uncomment and customize for compiled languages
    # - name: Build native binary
    #   run: |
    #     VERSION=${{ needs.version.outputs.new_tag }}
    #     PLATFORM=${{ matrix.platform }}
    #     ARCH=${{ matrix.arch }}
    #     EXT=${{ matrix.ext }}
    #     OUTPUT="myapp-${VERSION}-${PLATFORM}-${ARCH}${EXT}"
    #
    #     # Go example
    #     GOOS=${PLATFORM} GOARCH=${ARCH} go build -o ${OUTPUT} main.go
    #
    #     # Or Rust example
    #     # cargo build --release --target ${PLATFORM}-${ARCH}
    #
    #     mkdir -p artifacts
    #     mv ${OUTPUT} artifacts/

    - name: Upload build artifacts
      uses: actions/upload-artifact@v4
      with:
        name: ${{ matrix.platform }}-${{ matrix.arch }}
        path: artifacts/*
        retention-days: 1

release:
  needs: [version, build]
  runs-on: ubuntu-latest
  permissions:
    contents: write

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

    - name: Display structure
      run: ls -R all-artifacts

    - name: Create GitHub Release
      uses: softprops/action-gh-release@v1
      with:
        tag_name: ${{ needs.version.outputs.new_tag }}
```

```
        name: Release ${{ needs.version.outputs.new_tag }}
        generate_release_notes: true
        files: |
          all-artifacts/**/*
```