# GitHub Actions Workflow Documentation: Detect Merge Commits

**name: Detect Merge/Rebase from Base Branch and Notify Teams** *Workflow name to describe the purpose.*

**on:  
 pull\_request:  
 types: [opened, synchronize]** *Triggers the workflow when a pull request is opened or updated.*

**permissions:  
 contents: read  
 pull-requests: write** *Sets read access for repository contents and write access for pull requests.*

**jobs:  
 detect-conflict-merge-message:  
 runs-on: ubuntu-latest** *Defines the job and specifies the runner environment.*

**- name: Checkout code** *Checks out the code from the repository.*

**uses: actions/checkout@v4** *Uses the GitHub-provided checkout action.*

**with:  
 fetch-depth: 0** *Fetches full Git history to enable proper commit comparison.*

**- name: Determine merge commits after common ancestor** *Main logic step to detect merge commits.*

**id: detect** *Assigns an ID to this step for later reference.*

**run: |** *Indicates that the following lines are shell script commands.*

**BASE\_BRANCH="${{ github.base\_ref }}"** *Sets the base branch from the PR.*

**MAIN\_BRANCH="main"** *Defines main branch explicitly.*

**HEAD\_BRANCH="${{ github.head\_ref }}"** *Sets the source (feature) branch of the PR.*

**git fetch origin "$MAIN\_BRANCH" "$HEAD\_BRANCH"** *Fetches both main and feature branches.*

**COMMON=$(git merge-base origin/"$MAIN\_BRANCH" origin/"$HEAD\_BRANCH")** *Finds the common ancestor commit.*

**MERGES=$(git log --merges --first-parent --pretty=format:"• %h | %an | %ad | %s" --date=short "$COMMON..origin/$HEAD\_BRANCH")** *Finds merge commits after the common ancestor.*

**FIRST\_SHA=$(git rev-list --merges --first-parent "$COMMON..origin/$HEAD\_BRANCH" | head -n 1)** *Gets the SHA of the first merge commit.*

**if [ -n "$FIRST\_SHA" ]; then** *Checks if a merge commit was found.*

**AUTHOR=$(git show -s --format="%an" $FIRST\_SHA)** *Gets the author of the first merge commit.*

**DATE=$(git show -s --format="%cI" $FIRST\_SHA)** *Gets the date of the first merge commit.*

**PARENT1=$(git rev-parse "$FIRST\_SHA^1")** *Gets the first parent of the merge commit.*

**PARENT2=$(git rev-parse "$FIRST\_SHA^2")** *Gets the second parent of the merge commit.*

**MESSAGE=$(git show -s --format="%s" $FIRST\_SHA)** *Gets the commit message.*

**REPO\_URL="https://github.com/${{ github.repository }}"** *Forms the base repo URL.*

**COMMIT\_URL="$REPO\_URL/commit/$FIRST\_SHA"** *Forms the URL to the merge commit.*

**fi** *End of the conditional block.*

**{** *Start collecting outputs for use in later steps.*

**echo "merge\_list<<EOF"** *Defines the multi-line output start.*

**echo "$MERGES"** *Prints the list of merge commits.*

**echo "EOF"** *Ends the multi-line output.*

**echo "merge\_commit=$FIRST\_SHA"** *Stores the SHA as output.*

**echo "merge\_author=$AUTHOR"** *Stores the author as output.*

**echo "merge\_date=$DATE"** *Stores the date as output.*

**echo "merge\_message=$MESSAGE"** *Stores the message as output.*

**echo "parent1=$PARENT1"** *Stores parent1 SHA as output.*

**echo "parent2=$PARENT2"** *Stores parent2 SHA as output.*

**echo "commit\_url=$COMMIT\_URL"** *Stores the commit URL.*

**} >> "$GITHUB\_OUTPUT"** *Writes all output to the GitHub Actions output context.*

**if [ -n "$MERGES" ]; then** *If merge commits exist, display them and fail the job.*

**echo "❌ Merge commit(s) found:"** *Log message for detected merges.*

**echo "$MERGES"** *Prints the merge commits.*

**exit 1** *Exits the script with failure to trigger notifications.*

**else** *If no merges were found.*

**echo "✅ No merge commits found..."** *Success message.*

**fi** *End if block.*

**- name: Post PR comment if merge detected** *Posts a comment on the PR if merge commits were found.*

**if: failure()** *Only runs if the previous step failed.*

**env:** *Defines environment variables for the step.*

**GH\_TOKEN: ${{ secrets.GITHUB\_TOKEN }}** *GitHub token for authentication.*

**PR\_NUMBER: ${{ github.event.pull\_request.number }}** *PR number for the comment.*

**REPO: ${{ github.repository }}** *Repository name.*

**MERGE\_INFO: ${{ steps.detect.outputs.merge\_list }}** *Passes the detected merge list.*

**run: |** *Runs shell script.*

**gh api -X POST /repos/$REPO/issues/$PR\_NUMBER/comments -f body="..."** *Posts a structured message to the PR.*

**- name: Send Power Automate notification** *Sends a webhook notification to Power Automate.*

**if: failure()** *Only runs if the detect step failed.*

**env:** *Webhook and payload data.*

**POWER\_WEBHOOK: ${{ secrets.POWER\_AUTOMATE\_WEBHOOK }}** *Webhook URL stored in secrets.*

**run: |** *Runs shell script.*

**curl -X POST "$POWER\_WEBHOOK" -H 'Content-Type: application/json' -d "{...}"** *Sends a detailed JSON payload with commit metadata.*