AI PR Review (Azure DevOps)

AI PR Review (Azure DevOps)

Status: LIVE on QODY (PR #76, Build 334). Bilko ready (PR #77, paused by CEO).
Engine: Gemini 2.5 Flash (REST API, pluggable architecture)
Mode: Comment-only (non-blocking, never fails merge)
Evidence: `~/system/evidence/105098/`

---

1. What It Is

AI-powered PR review system for Azure DevOps repositories, equivalent to GitHub Copilot PR review functionality (which does not exist for Azure DevOps). Provides automated code review comments on pull requests using Gemini 2.5 Flash.

Why we built it: GitHub Copilot review is GitHub-only. Azure DevOps marketplace AI review extensions require sending our code to third-party vendors. This solution keeps control in-house while providing automated review feedback.

Key Features

- **Comment-only mode:** Never blocks merge, always `continueOnError: true`
- **Idempotent updates:** Re-runs update the same comment thread (marker: `<!-- AI-PR-REVIEW:v1 -->`), no spam on every push
- **Structured feedback:** Summary + CRITICAL + SHOULD-FIX + NIT sections with file:line references
- **Fail-safe:** Infrastructure failures skip review with a note, never red-gate the pipeline
- **Pluggable engine:** `REVIEWER_ENGINE=gemini|claude|forge` (Gemini default)
- **PR-only trigger:** Runs only on `Build.Reason = PullRequest`, absent on push/schedule/manual builds

---

2. Architecture

Components

1. **Script:** `tools/ai-pr-review.mjs` (plain Node.js ≥18, zero npm dependencies — built-in fetch + git)
2. **Pipeline job:** `ai_pr_review` in `CI_Gates` stage of `azure-pipelines.yml`
3. **LLM:** Gemini 2.5 Flash via REST API (`generativelanguage.googleapis.com`, header `x-goog-api-key`)
4. **Azure DevOps APIs:** PR iterations, threads, git refs

Data Flow

```
PR created/updated
   ↓
Branch Policy Build Validation triggers pipeline (NOTE: YAML pr: block is IGNORED on Azure Repos)
   ↓
ai_pr_review job runs (condition: Build.Reason = PullRequest)
   ↓
GET .../pullRequests/{id}/iterations → latest iteration SHA refs
   ↓
git fetch + git diff (commonRefCommit..sourceRefCommit)
   ↓
Diff filtering (exclude lock/generated/binary, 250KB limit, max 40 files)
   ↓
Gemini API call (one shot per run)
   ↓
GET .../threads (check for existing marker)
   ↓
POST new thread OR PATCH existing thread
   ↓
Build completes (success, never blocks merge)
```

Idempotency Mechanism

First line of every review comment: `<!-- AI-PR-REVIEW:v1 -->`
On each run:

1. Fetch all PR threads
2. Search for marker in thread comments
3. If found: `PATCH` that thread (update in place)
4. If not found: `POST` new thread

Result: 3 builds on same PR = 1 thread (verified live on QODY PR #76: builds 332/333/334 → thread id 115).

Fail-Safe Design

Every external operation (network, git, API) has timeout via `AbortSignal.timeout`. On any failure:

- Log error
- Post "Review skipped: [reason]" comment (if API accessible)
- `exit 0` (never fails the build)

Draft PRs are automatically skipped (check `SYSTEM_PULLREQUEST_ISDRAFT`).

---

3. Setup for NEW Repository

Prerequisites

- Azure DevOps organization with Azure Repos
- Self-hosted or Microsoft-hosted agent pool (Node.js ≥18)
- Gemini API key (from `~/system/config/secrets/gemini.json`)

Step-by-Step Setup

A. Copy Script

1. Copy `tools/ai-pr-review.mjs` from QODY or Bilko repo to your repo

```
# From QODY:
cp ~/business/ALAI-Holding-AS/products/qody/tools/ai-pr-review.mjs <your-repo>/tools/
```

B. Add Pipeline Job

2. Edit `azure-pipelines.yml`, add this job to your `CI_Gates` stage (or create stage if none):

```
stages:
  - stage: CI_Gates
    displayName: 'CI Gates'
    jobs:
      # ... existing jobs ...
```

```
      - job: ai_pr_review
        displayName: 'AI PR Review (Gemini, comment-only)'
        condition: eq(variables['Build.Reason'], 'PullRequest')
        continueOnError: true
        timeoutInMinutes: 10
        pool:
          name: your-pool-name  # e.g., bilko-selfhosted, qody-selfhosted, or
vmImage: ubuntu-latest
        steps:
          - checkout: self
            fetchDepth: 50  # CRITICAL: need history for git merge-base, NOT
fetchDepth: 1
            persistCredentials: true  # CRITICAL: for git fetch in script
          - script: node tools/ai-pr-review.mjs
            env:
              SYSTEM_ACCESSTOKEN: $(System.AccessToken)
              GEMINI_API_KEY: $(GEMINI_API_KEY)
              REVIEWER_ENGINE: 'gemini'
              REVIEWER_MODEL: 'gemini-2.5-flash'
```

C. Configure Secret Variable

3. **Option A (Recommended):** Azure Key Vault variable group

```
az pipelines variable-group create \
  --organization https://dev.azure.com/alai-holding \
  --project <project-name> \
  --name 'AI-Review-Secrets' \
  --authorize true \
  --variables GEMINI_API_KEY=<paste-key-here>
```

   Then reference in YAML: `variables: - group: AI-Review-Secrets`
4. **Option B:** Pipeline-level secret variable via UI
   - Go to Pipeline → Edit → Variables → New variable
   - Name: `GEMINI_API_KEY`
   - Value: (paste from `~/system/config/secrets/gemini.json`)
   - Check "Keep this value secret"

D. Set Repository Permissions

4. Grant Build Service permission to post PR comments:

```
# Via Azure CLI:
az devops security permission update \
  --organization https://dev.azure.com/alai-holding \
  --project <project-name> \
  --subject "<project-name> Build Service (alai-holding)" \
  --token "repoV2/<project-id>/<repo-id>" \
  --allow-bit 16384  # Contribute to pull requests
```

Or via UI:

- Project Settings → Repositories → [your repo] → Security
- Search for "[Project Name] Build Service (alai-holding)"
- Set "Contribute to pull requests" = **Allow**

E. Configure Branch Policy

5. **CRITICAL:** YAML `pr:` trigger block is **IGNORED** on Azure Repos (only works for GitHub/Bitbucket). PR builds require Branch Policy.

```
# Via Azure CLI:
az repos policy build create \
  --organization https://dev.azure.com/alai-holding \
  --project <project-name> \
  --repository-id <repo-id> \
  --branch main \
  --build-definition-id <pipeline-definition-id> \
  --display-name 'PR Build Validation' \
  --queue-on-source-update-only true \
  --manual-queue-only false \
  --blocking false  # Non-blocking for comment-only review
```

Or via UI:

- Project Settings → Repositories → [your repo] → Policies → Branch Policies → [main/master]
- Build Validation → Add build policy
- Select your pipeline
- Policy requirement: **Optional** (non-blocking)
- Trigger: Automatic

F. Test

6. Create a test PR with a trivial change (e.g., add comment, fix typo)
7. Verify:

- Build triggers automatically
- `ai_pr_review` job appears in pipeline
- Bot posts a comment thread (look for marker at top)
- Push another commit to same PR → same thread updates (footer shows new timestamp + build number)
- Merge is NOT blocked

---

4. Troubleshooting

Problem: 403 Forbidden on POST threads

Symptom: Job log shows "Error posting review: 403"
Cause: Build Service lacks "Contribute to pull requests" permission
Fix: Step D above (set allow-bit 16384 on repoV2 token)

Problem: "could not read Password for 'https://dev.azure.com'"

Symptom: `git fetch` fails with credential error
Cause: `persistCredentials: false` on checkout (default in some templates)
Fix: Explicitly set `persistCredentials: true` in checkout step, AND add auth header:

```
git -c http.extraheader="AUTHORIZATION: bearer $SYSTEM_ACCESSTOKEN" fetch origin $SHA
```

(Script already does this, but verify checkout step has `persistCredentials: true`)

Problem: Pathspec syntax error in git diff

Symptom: `fatal: pathspec ':(exclude)package-lock.json' did not match any files`
Cause: Passing pathspec through shell as string instead of argv array
Fix: Use `execFileSync` with array args (already implemented in script):

```
execFileSync('git', ['diff', sha1, sha2, '--', ':(exclude)*.lock'])
// NOT: execSync(`git diff ... :(exclude)*.lock`)  ← shell parsing breaks on parens
```

Problem: Old PR has no review / "Policy not evaluated"

Symptom: Policy shows "Not configured" or review never appears
Cause: Branch policies apply only to PRs created/updated AFTER the policy is configured
Fix: Push an empty commit to the PR branch to re-evaluate:

```
git commit --allow-empty -m "Trigger policy evaluation"
git push
```

Problem: Job not appearing on PR builds

Symptom: `ai_pr_review` job missing from pipeline run
Possible causes:

1. **No branch policy configured** → YAML `pr:` block does NOT work on Azure Repos. See step E above.
2. **Condition not met:** Verify `condition: eq(variables['Build.Reason'], 'PullRequest')` in YAML
3. **Wrong branch:** Policy may be configured for `main` but PR targets `develop`

Problem: Script times out / no comment posted

Symptom: Job runs 10 minutes then cancels
Causes:

- Very large diff (>250KB) taking too long to process → script auto-truncates, should not timeout
- Gemini API unreachable → script should skip with note
- Network partition on self-hosted agent

Debug: Check job log for last script output before timeout. Adjust `timeoutInMinutes` if needed (default 10).

Problem: Review quality is poor / too verbose

Tuning options:

- Switch model: `REVIEWER_MODEL=gemini-2.5-pro` (slower, deeper) or `gemini-2.0-flash-exp` (faster, lighter)
- Adjust prompt in script (look for `const prompt = ` ` section in `ai-pr-review.mjs`)
- Filter file types: edit `EXCLUDE_PATTERNS` array in script

---

5. Known Gaps & Roadmap

v1.0 (Current — LIVE on QODY)

- ✅ Comment-only review (non-blocking)
- ✅ Idempotent updates (one thread per PR)
- ✅ Gemini 2.5 Flash engine
- ✅ Structured feedback (Summary/CRITICAL/SHOULD-FIX/NIT)
- ✅ Fail-safe (never blocks merge)
- ✅ PR-only trigger

v1.1 (Planned — MC #105102)

- ⏳ **Per-file thread anchoring:** Currently only summary thread is created. Per-file threads with `threadContext` line anchoring for CRITICAL/SHOULD-FIX findings not working (script has `postFileComments` function but threads not appearing in live PR).
- ⏳ **PAT rotation:** MC #105103 — existing `.git/config` contains old hardcoded PAT (FAT0...), security debt, does not block current functionality.

v2.0 (Future)

- 🔮 **Blocking mode option:** PR Status API (`genre: ai-pr-review`) + branch policy "required" toggle — allows review to gate merge when signal/noise ratio proven
- 🔮 **Manual deep review:** `gemini-2.5-pro` on-demand run via pipeline parameter
- 🔮 **Auto-resolve stale threads:** Mark old comments as resolved when issues fixed in new commits
- 🔮 **Multi-engine support:** Claude API, local Ollama/MLX forge models

---

6. Live Deployments

QODY

- **Status:** ✅ LIVE (PR #76, Build 334)
- **Org:** dev.azure.com/alai-holding
- **Project:** QODY
- **Pool:** qody-selfhosted
- **Evidence:** `~/system/evidence/105098/qody-pr76-live-review-evidence.md`
- **Verified:** Idempotent (3 builds → 1 thread), non-blocking (mergeStatus=succeeded), secrets not leaked

Bilko

- **Status:** ◫◫ READY (PR #77, code deployed, pipeline paused by CEO decision)
- **Org:** dev.azure.com/alai-holding
- **Project:** Bilko
- **Pool:** bilko-selfhosted
- **Branch:** `feat/ai-pr-review-105098`
- **Next:** Awaiting CEO pipeline unpause to verify live

---

7. Security Notes

- **API Key:** `GEMINI_API_KEY` stored as Azure DevOps secret variable (masked in logs), never echoed by script
- **Access Token:** `System.AccessToken` auto-provided by Azure Pipelines, scoped to PR comment API only (permission bit 16384)
- **Data Exposure:** PR diff content sent to Google Gemini API — acceptable for internal ALAI code (Bilko/QODY), DO NOT enable on repos containing client data without explicit decision
- **Secret Detection:** Prompt instructs model to NEVER quote potential secrets from diff, only reference "possible secret at file:line"
- **Build Logs:** Verified clean (no leaked keys in QODY Build 334 logs, Proveo-verified)

---

8. Key Lessons Learned

Lesson 1: YAML pr: Block Ignored on Azure Repos

Azure DevOps YAML `pr:` trigger blocks work ONLY for GitHub/Bitbucket repos, NOT Azure Repos. PR builds require explicit Branch Policy → Build Validation configuration.

Cost: 3 build iterations on QODY to discover this (policy cfg id=2 created, then all old PRs needed empty commits to re-evaluate).

Lesson 2: Git Pathspec Must Use execFileSync Argv

Pathspec syntax `:(exclude)pattern` breaks when passed through shell string (`execSync`) due to parentheses. ALWAYS use `execFileSync` with argv array for git commands.

Lesson 3: persistCredentials Required for Git Fetch

Checkout step defaults to `persistCredentials: false` in some templates, breaking `git fetch` in job. Explicit `persistCredentials: true` + `http.extraheader` auth required.

Lesson 4: Build Service Permission Non-Obvious

"Contribute to pull requests" (bit 16384) not granted by default to Build Service identity. First-run 403 error inevitable without pre-setup. Now part of standard checklist.

---

9. Related Documentation

- **Architecture Plan:** `~/.claude/plans/hashed-soaring-pony.md`
- **Project Memory:** `~/.claude/projects/-Users-makinja/memory/project_azdo_ai_pr_reviewer_2026-07-09.md`
- **Proveo Verification:** `~/system/evidence/105098/proveo-verification-report.md`
- **MC Task:** #105098 (build), #105101 (this runbook), #105102 (per-file threads), #105103 (PAT rotation)

Last updated: 2026-07-09 | Author: Skillforge (John orchestration) | MC #105101

Revision #1
Created 2026-07-09 12:39:10 UTC by John
Updated 2026-07-09 12:39:10 UTC by John