Agent Playbook – Cloudflare Pages Auto-Repair (ZIP Bundle)

Version 1.0 • Generated: 2025-09-04 14:32

This playbook describes how an automated agent should install the provided ZIP bundle, verify critical Cloudflare Pages Functions are live, and repair them if necessary. It covers prerequisites, installation, verification, self■healing, and optional one■click installer workflow.

Overview

- Targets a Cloudflare Pages site with a `functions/` directory.
- Ensures two endpoints respond in Production: `GET /diag` → `diag ok`, and `GET /paypal/notify` (GET ping + POST JSON).
- Installs a Claude■powered agent (`agent.py`) and three GitHub workflows for bootstrap, healthchecks, and repairs.
- Performs idempotent updates; only touches the minimum required files.

Prerequisites

- Cloudflare Pages project is already set up; KV binding: `KV_STORE → AUTO_KV` on Production.
- GitHub repository connected to the Pages project.
- GitHub Actions secrets (Settings → Secrets and variables → Actions): `ANTHROPIC_API_KEY`, `GH_TOKEN` (contents:write), `GH_REPO` (e.g., `owner/repo`).
- Optional Actions Variable: `SITE_URL` (defaults to `https://autocaption-pro.pages.dev`).

ZIP Bundle Contents

- agent.py – Claude agent that reads/repairs repo and verifies endpoints
- requirements.txt – Python deps (anthropic, requests)
- .github/workflows/claude-agent.yml – scheduled/manual Claude agent
- .github/workflows/bootstrap.yml – scaffolds missing Functions & verifies
- .github/workflows/healthcheck.yml – hourly endpoint pings

Install (Agent Instructions)

1. Acquire the ZIP file and extract into the repository root (same level as `public/`).
2. Create a branch: `agent/bundle-install-YYYYMMDD-HHMM`.

3. Add/replace only paths in the bundle (do not overwrite unrelated files).
4. Commit and push the changes. Merge to `main` if permitted (or open PR).
5. Confirm GitHub Actions shows the new workflows (Bootstrap, Claude Agent, Healthcheck).
6. Set required secrets/variables: `ANTHROPIC_API_KEY`, `GH_TOKEN`, `GH_REPO`, (`SITE_URL` optional).
7. Run **Bootstrap Pages Functions** once to scaffold `functions/diag.js` and `functions/paypal/notify.js`.

Verification (Production URL)

- GET `/diag` must return exactly: `diag ok`.
- GET `/paypal/notify` must contain: `paypal notify GET ok`.
- If not ready immediately, retry for up to ~5 minutes with exponential backoff.

Self■Healing Procedure

- Ensure `functions/diag.js` exists with:

```
export async function onRequestGet() {
  return new Response('diag ok', { headers: { 'content-type': 'text/plain' } });
}
```

- Ensure `functions/paypal/notify.js` exists (GET ping + POST handler, minimal version):

```
const CORS = {
  'access-control-allow-origin': '*',
  'access-control-allow-methods': 'GET,POST,OPTIONS',
  'access-control-allow-headers': 'content-type',
};

export async function onRequest(context) {
  const { request, env } = context;
  const method = request.method.toUpperCase();

  if (method === 'OPTIONS') return new Response(null, { headers: CORS });

  if (method === 'GET') {
    return new Response('paypal notify GET ok', {
      headers: { ...CORS, 'content-type': 'text/plain' },
    });
  }

  if (method !== 'POST') {
    return new Response('Method Not Allowed', { status: 405, headers: CORS });
  }

  let body = {};
  try { body = await request.json(); } catch {}
  const { session_id, order_id, capture } = body || {};
  if (!session_id || !order_id || !capture) return json({ error: 'missing_fields' }, 400);

  const session = await env.KV_STORE.get(`session:${session_id}`, { type: 'json' });
  if (!session) return json({ error: 'unknown_session' }, 404);

  const pu = capture?.purchase_units?.[0];
  const cap = pu?.payments?.captures?.[0];
  const amount   = cap?.amount?.value;
  const currency = cap?.amount?.currency_code;
  const payerEmail = capture?.payer?.email_address;

  const job = {
    state: 'paid',
    order_id,
    amount,
    currency,
    payer_email: payerEmail,
    session,
    created_at: Date.now()
  };

  await env.KV_STORE.put(`job:${session_id}`, JSON.stringify(job), { expirationTtl: 86400 });
  session.state = 'paid';
  await env.KV_STORE.put(`session:${session_id}`, JSON.stringify(session), { expirationTtl: 86400
  return json({ ok: true });
}

function json(obj, status=200) {
  return new Response(JSON.stringify(obj), {
    status,
    headers: { 'content-type': 'application/json; charset=utf-8', ...CORS },
  });
}
```

Optional: One■Click Installer Workflow

Create `.github/workflows/install-bundle.yml` to accept a public ZIP URL and install the bundle automatically (then verify endpoints):

```
name: Install Agent Bundle
on:
  workflow_dispatch:
    inputs:
      bundle_url:
        description: URL to pages-agent-bundle.zip
        required: true
        type: string

jobs:
  install:
    runs-on: ubuntu-latest
    permissions:
      contents: write
      actions: write
    steps:
      - uses: actions/checkout@v4
      - name: Download bundle
        run: |
          set -e
          curl -fsSL "${{ inputs.bundle_url }}" -o /tmp/bundle.zip
          unzip -o /tmp/bundle.zip -d /tmp/bundle
          rsync -av /tmp/bundle/ ./
      - name: Commit bundle
        run: |
          git config user.name "pages-agent"
          git config user.email "pages-agent@users.noreply.github.com"
          if [ -z "$(git status --porcelain)" ]; then
            echo "No changes to commit."
          else
            git add -A
            git commit -m "Agent: install bundle (workflows + agent)"
            git push
          fi
      - name: Verify endpoints
        run: |
          set -e
          BASE="${{ vars.SITE_URL || 'https://autocaption-pro.pages.dev' }}"
          for i in $(seq 1 40); do
            D=$(curl -fsS "$BASE/diag" || true)
            N=$(curl -fsS "$BASE/paypal/notify" || true)
            echo "try $i -> /diag: $D | /paypal/notify: $N"
            if [ "$D" = "diag ok" ] && echo "$N" | grep -q "paypal notify GET ok"; then
              exit 0
            fi
            sleep 6
          done
          echo "::error ::Endpoints not responding"; exit 1
```

Troubleshooting

- If `/diag` or `/paypal/notify` still render the index page, ensure the `functions/` directory sits at the repo root (not under `public/`).
- Check Cloudflare Pages deployment logs for `Found Functions directory at /functions`.
- Confirm Pages KV binding: Settings → Bindings → `KV_STORE` → `AUTO_KV` on Production (and Preview if used).
- Disable ad■blockers/pop■up blockers during PayPal Sandbox tests.
- Use the Sandbox **Personal** buyer credentials in the PayPal popup (not your real account).

Verification Checklist

- `/diag` returns `diag ok`

- `/paypal/notify` returns a GET OK string and accepts POST
- Workflows visible in GitHub Actions; healthcheck passes
- KV write/read succeeds for `session:` and `job:`

Appendix A — Files in the Bundle

```
pages-agent-bundle/
■■ agent.py
■■ requirements.txt
■■ README.md
■■ .github/workflows/
   ■■ claude-agent.yml
   ■■ bootstrap.yml
   ■■ healthcheck.yml
```

Appendix B — Environment Variables / Secrets

```
ANTHROPIC_API_KEY  (secret)  – Claude API key
GH_TOKEN             (secret)  – GitHub token with repo contents:write
GH_REPO              (secret)  – owner/repo (e.g., IdeaMLabs/autocaption-pro)
SITE_URL             (variable) – Production URL (optional; defaults provided)
```