Forms E2E Testing Protocol

Forms E2E Testing Protocol

Purpose

This protocol defines the mandatory end-to-end validation steps for all contact forms, waitlist submissions, and user-facing form handlers. HTTP 200 response is NOT sufficient evidence of success — inbox delivery or database persistence MUST be verified.

Scope

Applies to:

- Contact forms (alai.no, snowit.ba, merdzanovic.ba, etc.)
- Waitlist/signup forms (getdrop.no, product landing pages)
- Feedback forms
- Any form that sends email or stores user data

Test Levels

Level 1 — HTTP Layer (Necessary but NOT Sufficient)

1. Submit form via browser (real user flow)
2. Verify HTTP response: 200 OK or 201 Created
3. Verify UI feedback: success message displayed
4. ⚠ **STOP:** This is NOT proof the form works. Proceed to Level 2.

Level 2 — Delivery Verification (MANDATORY)

For Email-Based Forms

1. **Submit test form** with known test data (e.g., `test@example.com`, subject: "E2E Test YYYY-MM-DD HH:MM")
2. **Check target inbox** within 60 seconds:
   - Via Himalaya CLI: `himalaya search --account info@alai.no --folder INBOX "subject:E2E Test"`
   - Via webmail: Log into one.com or Gmail, verify message received
   - Via IMAP client: Thunderbird, Apple Mail, etc.
3. **Verify message contents:**
   - All form fields present in email body
   - From address is correct (e.g., `noreply@alai.no` or form submitter)
   - Reply-to is user's submitted email (if applicable)
4. **Verify SMTP logs** (if accessible):
   - Check Resend dashboard (for Resend API)
   - Check Cloudflare Email Workers logs
   - Check backend logs (`journalctl -u form-handler` or PM2 logs)

For Database-Based Forms (e.g., waitlist with D1)

1. **Submit test form** with unique identifier (e.g., email `test+TIMESTAMP@alai.no`)
2. **Query database:**
   - CF D1: `wrangler d1 execute drop-waitlist --command "SELECT * FROM submissions WHERE email LIKE 'test+%'"`
   - PostgreSQL: `psql -h localhost -U user -d dbname -c "SELECT * FROM waitlist WHERE email = 'test@example.com';"`
   - SQLite: `sqlite3 ~/path/to/db.sqlite "SELECT * FROM forms WHERE email = 'test@example.com';"`
3. **Verify record fields:**
   - Timestamp is recent (within last 2 minutes)
   - All form fields stored correctly
   - No NULL values where data expected

Level 3 — Error Handling (Recommended for Production Forms)

1. **Submit invalid data:**
   - Missing required fields
   - Invalid email format
   - XSS/SQL injection attempts (if validation implemented)
2. **Verify error responses:**
   - HTTP 400 Bad Request (not 500 Internal Server Error)
   - Clear error message to user ("Email is required")
   - No server crash or 500 error
3. **Simulate backend failure:**
   - Temporarily break SMTP credentials or DB connection

- Verify graceful failure (user sees "Failed to send, please try again" — not silent success)
- Verify logging of failure (so ops team can detect issue)

Himalaya CLI Setup (for Email Verification)

Install

```
brew install himalaya
```

Configure Account

Add to `~/.config/himalaya/config.toml`:

```
[accounts.info-alai]
default = false
email = "info@alai.no"
display-name = "ALAI Info"

[accounts.info-alai.imap]
host = "imap.one.com"
port = 993
encryption = "tls"
login = "info@alai.no"
passwd.cmd = "bw get password 'Email - info@alai.no' --session $(cat /tmp/bw-session)"

[accounts.info-alai.smtp]
host = "send.one.com"
port = 587
encryption = "start-tls"
login = "info@alai.no"
passwd.cmd = "bw get password 'Email - info@alai.no' --session $(cat /tmp/bw-session)"
```

Usage

```
# List recent messages
himalaya list --account info-alai --folder INBOX --page-size 20

# Search for test submissions
himalaya search --account info-alai --folder INBOX "subject:E2E Test"

# Search by sender
himalaya search --account info-alai --folder INBOX "from:noreply@alai.no"

# Search by date
himalaya search --account info-alai --folder INBOX "since:2026-04-21"
```

Credentials: Store IMAP password in Bitwarden item "Email - info@alai.no"

Pre-Deployment Checklist

Before marking any form handler as "ready for production":

- [ ] Level 1 HTTP test passed (200 OK + UI feedback)
- [ ] Level 2 delivery test passed (inbox check OR database record confirmed)
- [ ] Level 3 error handling tested (invalid input returns 400, backend failure does NOT return silent success)
- [ ] Form submission logged to application logs (even if email/DB write fails)
- [ ] Monitoring/alerting configured (e.g., Grafana alert if zero submissions for 7 days)
- [ ] Documentation updated (which inbox receives submissions, which DB table stores records)

Migration Checklist (Static Hosting Migrations)

When migrating sites from Vercel/Netlify to Cloudflare Pages:

- [ ] Inventory all POST endpoints (forms, webhooks, API routes)
- [ ] Verify each has CF Pages Function equivalent (`/functions/*.js`)
- [ ] Update form action URLs if needed (e.g., `/api/contact` → CF Pages route)
- [ ] Run full E2E test (this protocol) BEFORE declaring migration complete
- [ ] Monitor inbox/DB for 24 hours post-migration to catch silent failures

Known Silent Failure Patterns

Pattern 1: Catch-All Webhook Returns False Success

- **Symptom:** HTTP 200 + `{ok: true}` but no email sent
- **Cause:** Reverse proxy or tunnel routes form POST to wrong backend (e.g., Documenso webhook with `app.post('/*', ...)`)
- **Example:** alai.no contact form incident (2026-04-21, MC #8587)
- **Prevention:** Audit all catch-all routes; prefer explicit path matching

Pattern 2: Serverless Function Not Migrated

- **Symptom:** 404 Not Found or 200 OK (if static fallback serves index.html)
- **Cause:** Vercel API routes (`/api/contact.js`) not ported to CF Pages Functions (`/functions/contact.js`)
- **Prevention:** Explicit checklist during migration (see above)

Pattern 3: SMTP Credentials Not Migrated

- **Symptom:** Backend logs show "SMTP authentication failed" but frontend receives 200 OK
- **Cause:** Environment variables not set in new hosting platform
- **Prevention:** Verify env vars (RESEND_API_KEY, SMTP_PASSWORD) before testing

Related Pages

- Email Pipeline Runbook
- Incident — 2026-04-21 alai.no Contact Form Failure
- Static Hosting Migration — Progress Log
- Definition of Done

MC Task References

- **#8587** — alai.no contact form fix + documentation (this page)
- **#8538** — Master quality gate (references forms E2E testing requirement)
- **#8591** — snowit.ba contact form migration (same risk)

Authored: 2026-04-21 | Owner: Skillforge (QA protocol) + Proveo (enforcement) | Approved: Angie Jones

---

Revision #2
Created 2026-04-21 11:40:58 UTC by John
Updated 2026-05-31 20:06:23 UTC by John