Processes

Onboarding, invoicing, pipeline management, signing workflows.

Overview

Processes Overview

Onboarding, invoicing, pipeline management, and signing workflows.

Owner: John **Last Verified:** 2026-02-17

Contents

To be populated from business process documentation

Processes and Workflows

> Last Verified: 2026-02-17 | Owner: John

Processes & Workflows

Version: 1.0 **Last Updated:** 2026-01-28 **Owner:** Alem Basic **Prepared by:** John (Director) + Emir Delić (Scrum Master) + Amina Hadžić (Head of Projects)

Executive Summary

This document defines how work flows through the organization. It covers task lifecycle, approval workflows, communication protocols, meeting cadences, and operational processes. Every process is designed for speed, clarity, and accountability.

Key Principles:

- **Bias toward action** — ship fast, iterate based on feedback
- **Document decisions immediately** — no mental notes
- **Clear ownership** — every task has one owner (RACI: R = Responsible)
- **Escalate early** — blockers escalated within 1 hour
- **Continuous improvement** — retros every sprint, improve processes

---

1. Task Lifecycle — From Idea to Completion

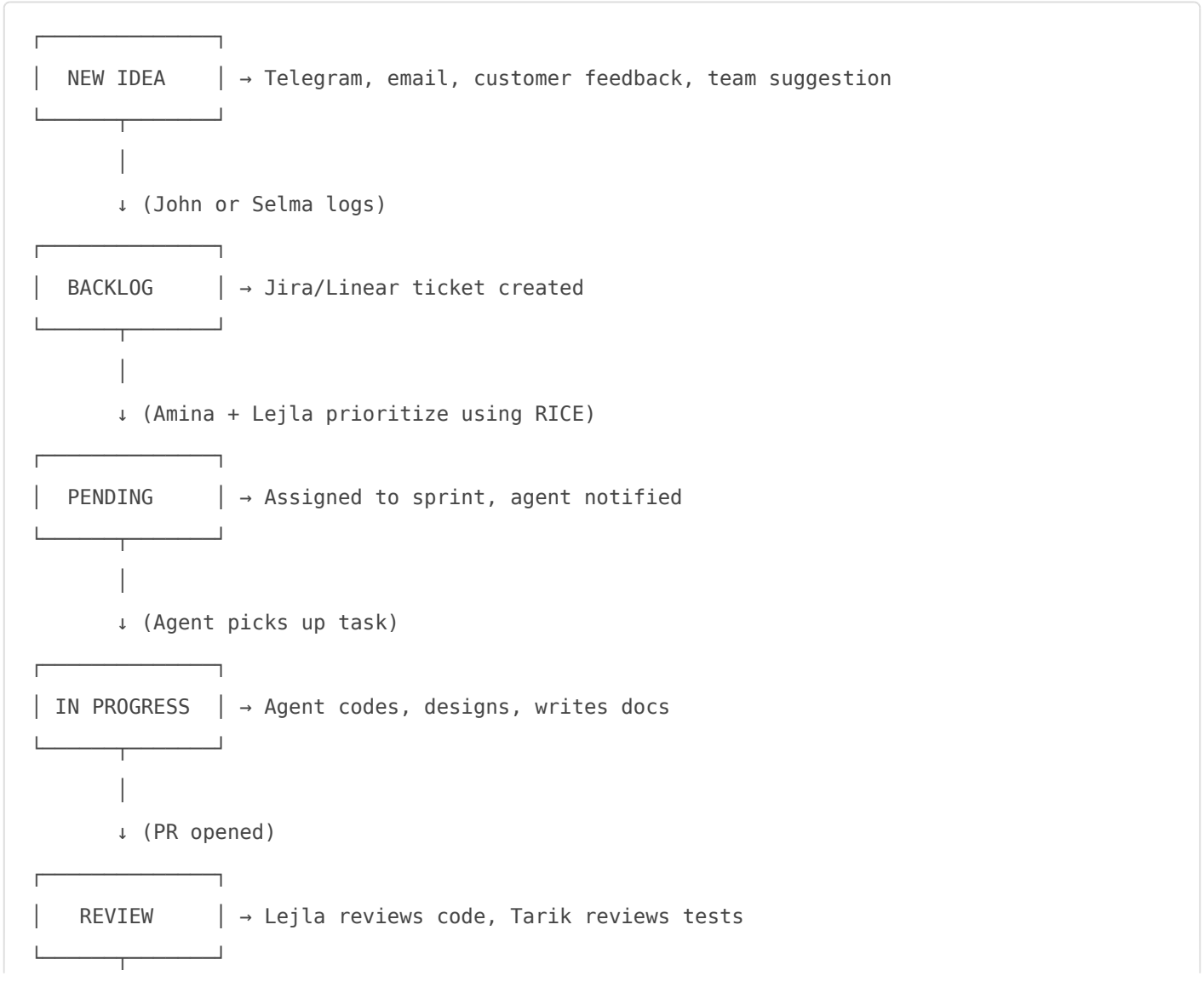
1.1 Task States

```
BACKLOG → PENDING → IN PROGRESS → REVIEW → TESTING → COMPLETED
   ↓         ↓           ↓            ↓         ↓          ↓
 Idea    Prioritized  Executing   Code review  QA    Deployed
```

Detailed States:

State	Definition	Owner	Exit Criteria
Backlog	Ideas, feature requests, bugs not yet prioritized	Amina + Lejla	Prioritized in sprint planning
Pending	Approved for sprint, waiting for pickup	Assigned agent	Agent starts work
In Progress	Actively being worked on	Agent	Work complete, PR opened
Review	Code review, design review, or approval	Lejla (code) or Amina (business)	Approved or changes requested
Testing	QA validation	Tarik	Tests pass, QA sign-off
Completed	Deployed to production, verified working	Nermin (deploy)	Live in production, no issues

1.2 Task Flow Diagram



```
       │
       ↓ (Approved)
┌──────────────────┐
│   TESTING    │ → Tarik runs E2E tests, checks quality gates
└──────┬───────────┘
       │
       ↓ (Tests pass)
┌──────────────────┐
│  DEPLOYMENT  │ → Nermin deploys to staging → production
└──────┬───────────┘
       │
       ↓ (Verified working)
┌──────────────────┐
│  COMPLETED   │ → Task archived, logged to DB
└──────────────────┘
```

1.3 Task Metadata (Every Task Must Have)

Every task in Jira/Linear includes:

- **Title:** Short, descriptive (< 70 chars)
- **Description:** What needs to be done, why, acceptance criteria
- **Owner:** One person responsible (RACI: R)
- **Priority:** P1 (critical), P2 (high), P3 (medium), P4 (low)
- **Estimate:** Story points or hours
- **Labels:** feature, bug, tech-debt, compliance, etc.
- **Sprint:** Which sprint (if assigned)
- **Dependencies:** Blocks/blocked by other tasks
- **Acceptance criteria:** Checklist of what "done" means

Example Task:

```
Title: Add RBAC to Patient Management Module

Description:
Implement role-based access control (RBAC) for patient management.
Roles: Admin (full access), Caregiver (view own patients), Billing (view billing only).

Why: HIPAA compliance — minimum necessary access.

Acceptance Criteria:
```

```
- [ ] Admin can view/edit all patients
- [ ] Caregiver can view only assigned patients
- [ ] Billing can view patient billing info only (no PHI)
- [ ] Audit log records all access attempts
- [ ] Tests: Unit tests for RBAC logic, E2E test for each role
- [ ] Documentation: Updated API docs, user guide

Owner: API Developer
Reviewers: Lejla (code), Dženan (compliance), Tarik (testing)
Priority: P2 (high)
Estimate: 13 story points (1 week)
Sprint: Sprint 8
Dependencies: Blocks "Phase 3 GA launch"
Labels: feature, RBAC, HIPAA, Phase-3
```

---

2. Sprint Process (Agile/Scrum)

2.1 Sprint Cadence

Sprint Length: 2 weeks (10 business days)

Sprint Schedule:

Week	Day	Event
Week 1	Monday	Sprint Planning (new sprint starts)
	Wed	Backlog Refinement
	Thu	Architecture Review (bi-weekly)
Week 2	Monday	Mid-sprint check-in (Emir + Amina)
	Wed	Backlog Refinement
	Fri	Sprint Review + Retro (sprint ends)

Daily: Standup at 9:15 AM CET (Mon-Fri)

2.2 Sprint Planning (Every 2 Weeks, Monday, 2-3 hours)

Attendees: Amina, Emir, Lejla, Tarik, Nermin, Selma, Dženan, API Dev, Frontend

Agenda:

1. **Review last sprint** (5 min)
   - What shipped?
   - What didn't ship? Why?
   - Velocity: actual vs planned
2. **Present sprint goal** (10 min)
   - Amina: "This sprint we will..."
   - Example: "Complete Phase 3 RBAC and deploy to beta"
3. **Review backlog** (30 min)
   - Selma: Customer feedback, feature requests
   - Lejla: Tech debt priorities
   - Dženan: Compliance requirements
   - Amina + Lejla: RICE-prioritized backlog
4. **Estimate tasks** (60 min)
   - Team reviews each task
   - Estimate story points (Fibonacci: 1, 2, 3, 5, 8, 13, 21)
   - Identify dependencies and risks
5. **Commit to sprint** (15 min)
   - Team commits to sprint backlog
   - Emir: "We commit to X story points this sprint"
   - Amina approves
6. **Assign tasks** (10 min)
   - Each agent picks tasks
   - Balanced workload

Output:

- Sprint backlog (committed tasks)
- Sprint goal (one sentence)
- Velocity target (story points)

2.3 Daily Standup (Mon-Fri, 9:15 AM CET, 15 min max)

Attendees: All team (Emir leads)

Format: Each person answers 3 questions (1 min each):

1. **What did I do yesterday?**
2. **What will I do today?**
3. **Any blockers?**

Rules:

- Start on time (9:15 sharp)
- Max 15 minutes (Emir enforces)
- No problem-solving (take offline)
- Blockers escalated immediately after standup

Emir's Checklist:

- ☐ Update sprint board before standup
- ☐ Note blockers → escalate to Amina or Lejla after
- ☐ Update burn-down chart

2.4 Backlog Refinement (Weekly, Wednesday, 1 hour)

Attendees: Emir, Lejla, Selma

Agenda:

1. Review new tasks (from customers, team, bugs)
2. Write clear descriptions and acceptance criteria
3. Estimate rough size (T-shirt: S, M, L, XL)
4. RICE score (prioritization)
5. Tag with labels (feature, bug, tech-debt, etc.)

Output:

- Refined backlog (ready for sprint planning)
- Top 20 tasks RICE-scored

2.5 Sprint Review (End of Sprint, Friday, 1 hour)

Attendees: Amina, Emir, Selma, Lejla, + stakeholders (Alem, customers)

Agenda:

1. **Demo completed work** (30 min)
   - Selma or agent demos features to stakeholders
   - Live demo, not slides
   - "Here's what we shipped this sprint"
2. **Review metrics** (15 min)
   - Velocity: committed vs completed

    - Quality: bugs found, test coverage
    - Customer feedback
3. **Gather feedback** (15 min)
    - Stakeholders provide input
    - New ideas added to backlog

Output:

- Stakeholder feedback
- New backlog items
- Celebration of wins

2.6 Sprint Retrospective (End of Sprint, Friday, 45 min)

Attendees: All team (Emir leads)

Format: Start/Stop/Continue

Agenda:

1. **What should we START doing?** (15 min)
    - New practices, tools, processes
    - Example: "Start writing ADRs for architecture decisions"
2. **What should we STOP doing?** (15 min)
    - Bad habits, wasteful processes
    - Example: "Stop scheduling meetings during focus time"
3. **What should we CONTINUE doing?** (15 min)
    - What's working well
    - Example: "Continue daily standups at 9:15 AM"
4. **Action items** (5 min)
    - Pick 1-3 improvements to implement next sprint
    - Assign owner for each

Rules:

- Blame-free zone
- Focus on process, not people
- Implement at least 1 action item per sprint

Emir's Job:

- Facilitate discussion
- Keep it positive and constructive
- Document action items

- Follow up on previous retro actions

---

3. Approval Workflows

3.1 Code Review Process

Trigger: Developer opens Pull Request (PR) on GitHub

Process:

```
1. DEVELOPER opens PR
   ↓
2. Automated checks run (CI/CD)
   ├─ Tests (unit, integration)
   ├─ Linting (ESLint, Prettier)
   ├─ Security scan (OWASP ZAP)
   └─ Build succeeds
   ↓ (if all pass)
3. LEJLA reviews code
   ├─ Architecture alignment
   ├─ Code quality
   ├─ Performance
   └─ Security
   ↓ (if approved)
4. TARIK reviews tests
   ├─ Test coverage ≥ 80%
   ├─ E2E test for happy path
   └─ Quality gates pass
   ↓ (if approved)
5. MERGE to main branch
   ↓
6. Auto-deploy to STAGING
   ↓
7. QA verification in staging
   ↓
8. Manual promote to PRODUCTION (Nermin)
```

PR Approval Criteria (Definition of Done):

- [ ] All automated tests pass
- [ ] Test coverage ≥ 80%
- [ ] E2E test for critical path
- [ ] Code reviewed by Lejla (or 2 senior devs)
- [ ] No security vulnerabilities (OWASP scan)
- [ ] Accessibility check (WCAG 2.1 AA)
- [ ] Performance benchmark pass (API < 500ms, page load < 2s)
- [ ] Documentation updated (if API or UI change)

SLA:

- Code review within 24 hours (Lejla)
- Revisions addressed within 24 hours (Developer)
- Total PR lifecycle: < 72 hours (3 days)

3.2 Feature Approval Process

For new features (not in roadmap):

```
1. IDEA submitted (Selma, customer, team member)
   ↓
2. SELMA writes user story + business case
   ↓
3. LEJLA estimates technical effort
   ↓
4. AMINA RICE-scores feature
   ↓ (if high RICE score)
5. JOHN prepares options (build, buy, defer)
   ↓
6. ALEM decides (approve, defer, reject)
   ↓ (if approved)
7. Add to backlog → sprint planning
```

Timeline:

- Idea → Decision: 1 week max

3.3 Deployment Approval

Staging Deployment:

- **Trigger:** PR merged to main
- **Approval:** Automated (no approval needed)
- **Rollback:** Automatic if health check fails

Production Deployment:

- **Trigger:** Manual (Nermin triggers after QA sign-off)
- **Approval:** Tarik (QA sign-off) + Nermin (deploy)
- **Rollback:** Manual (Nermin) if errors detected

Production Deployment Checklist:

- ☐ All staging tests pass
- ☐ QA sign-off from Tarik
- ☐ No P1/P2 bugs in staging
- ☐ Runbook updated (if new feature)
- ☐ Monitoring alerts configured
- ☐ Rollback plan documented
- ☐ Deploy during low-traffic window (if high-risk)

Deployment Windows:

- **Low-risk:** Anytime
- **High-risk:** Tuesday-Thursday, 10 AM - 2 PM CET (avoid Fridays, weekends, holidays)

3.4 Budget Approval

Amount	Approver	Process
< €500	John	Immediate, logged to DB
€500 - €5,000	John	Immediate, logged, Alem notified
€5,000 - €50,000	Alem	John prepares options, Alem decides
> €50,000	Alem	Formal proposal, board approval (if applicable)

Example:

- **€200/month SaaS tool** (Intercom) → John approves, logs decision
- **€3,000 patent filing** → John approves, notifies Alem
- **€10,000 Google Startup credits** → John prepares application, Alem approves
- **€100,000 Series A funding** → Alem decides

3.5 Compliance Sign-Off (HIPAA, PCI-DSS)

For any feature handling PHI (Protected Health Information):

```
1. DEVELOPER builds feature
   ↓
2. TARIK tests compliance controls
   ├─ Encryption at rest/transit
   ├─ Access control (RBAC)
   ├─ Audit logging
   └─ Data retention
   ↓ (tests pass)
3. DŽENAN reviews compliance checklist
   ├─ HIPAA Privacy Rule
   ├─ HIPAA Security Rule
   ├─ Vendor BAAs (if applicable)
   └─ Breach notification process
   ↓ (approved)
4. DEPLOY to production
```

Dženan's Compliance Checklist:

- ☐ PHI encrypted at rest (AES-256)
- ☐ PHI encrypted in transit (TLS 1.3)
- ☐ Access control enforced (RBAC, MFA)
- ☐ Audit log captures all PHI access
- ☐ Vendor BAAs signed (if third-party involved)
- ☐ Privacy policy updated (if needed)
- ☐ User consent obtained (if needed)

Compliance Sign-Off SLA: 48 hours (Dženan reviews within 2 business days)

---

4. Communication Protocols

4.1 Communication Channels & Usage

Channel	Use Case	Response SLA	Audience

Telegram (@johnbasicas_bot)	Urgent matters, quick decisions, P1 incidents	5-15 min	Alem ↔ John
CLI (Claude Code)	Deep work, architecture, coding, planning	Real-time (during session)	John ↔ agents
Email (john@basicconsulting.no)	External communication, formal records, client communication	4-24 hours	External parties
Slack (future)	Team collaboration, quick questions	1-4 hours	Internal team
Jira/Linear	Task tracking, sprint management	Daily check	Team
GitHub	Code, PRs, technical discussion	24 hours	Developers
Database (john.db)	Source of truth, all decisions logged	N/A (logged immediately)	John, Alem (query)
Standups	Daily status, blockers	9:15 AM CET daily	All team
Meetings	Strategic discussions, planning	Scheduled	Per invite

4.2 When to Use Which Channel

Situation	Channel	Why
Production is down (P1)	Telegram → Nermin, Lejla, John, Alem	Immediate response needed
Strategic decision needed	Telegram (Alem ↔ John)	Fast, informal
Task assignment	Jira/Linear + CLI	Trackable, logged
Code review	GitHub PR comments	Context, threaded discussion
Customer inquiry	Email (Selma)	Professional, recorded
Quick question for teammate	Slack (or CLI)	Fast, informal
Architecture proposal	Written doc (Lejla) + meeting	Needs deep thought
Bug report	Jira/Linear	Needs tracking, prioritization
Daily status	Standup (9:15 AM)	Synchronous, team awareness
Document important decision	Database (john.db)	Source of truth

4.3 Response Time Expectations

Priority	Channel	Response SLA	Example

P1 — Critical	Telegram, phone	5-15 min	Production down, security breach
P2 — High	Telegram, Slack	1 hour	Major bug, customer escalation
P3 — Medium	Slack, email	4 hours (business)	Feature request, minor bug
P4 — Low	Email, Jira	24 hours	Enhancement, question

Business Hours: 9 AM - 6 PM CET (Mon-Fri)

After-Hours: P1 only (Nermin on-call)

4.4 Meeting Etiquette

Rules:

- **Start on time** — don't wait for latecomers
- **End on time** — respect people's calendars
- **Agenda required** — no agenda = no meeting
- **One speaker at a time** — no interruptions
- **Action items documented** — every meeting ends with action items, owners, deadlines
- **No phones/distractions** — focus on meeting
- **Optional attendees clearly marked** — required vs optional

Meeting Types:

Type	Agenda Required	Notes Required	Max Duration
Standup	No (standard format)	No	15 min
Sprint planning	Yes	Yes	3 hours
Sprint review	Yes (demo list)	Yes	1 hour
Retro	No (standard format)	Yes (action items)	45 min
Architecture review	Yes (proposals)	Yes (ADRs)	2 hours
1:1	Optional	Optional	30 min
Ad-hoc problem-solving	No	Yes (decisions logged)	30 min

---

5. Incident Response Process

5.1 Incident Severity Levels

Priority	Definition	Examples	Response SLA	Escalation
P1 — Critical	Service down, data breach, multiple users affected	Production down, PHI exposed, database corruption	15 min	Immediate → Alem
P2 — High	Major feature broken, workaround exists, single user affected	Scheduling not working, payment failed	1 hour	If not resolved in 2h → Alem
P3 — Medium	Minor feature issue, cosmetic, no user impact	Report formatting wrong, UI glitch	4 hours	If not resolved in 24h → Amina
P4 — Low	Enhancement request, question, documentation	Feature request, how-to question	24 hours	No escalation

5.2 P1 Incident Response Flow

P1 Definition: Production down, security breach, data loss, HIPAA breach.

```
1. DETECTION (monitoring, customer report, team)
   ↓
2. NERMIN (DevOps) notified via PagerDuty
   ↓ (within 15 minutes)
3. NERMIN triages and starts investigation
   ↓ Simultaneously:
   ├─ Notify LEJLA (tech lead)
   ├─ Notify DŽENAN (if security/compliance)
   ├─ Notify JOHN (coordination)
   └─ Notify AMINA (stakeholder communication)
   ↓
4. INCIDENT CHANNEL opened (Slack or Telegram)
   ↓
5. INVESTIGATION (Nermin + Lejla)
   ├─ Identify root cause
   ├─ Assess impact (# users, data affected)
   └─ Determine fix or rollback
   ↓
6. DECISION (within 1 hour)
   ├─ Rollback to previous version (if safe)
   ├─ Apply hotfix (if fast and safe)
   └─ Escalate to ALEM (if major decision needed)
   ↓
```

```
7. IMPLEMENT FIX
   ↓
8. VERIFY fix working
   ↓ (if data breach)
9. BREACH NOTIFICATION process
   ├─ Dženan leads
   ├─ Notify affected customers (within 60 days per HIPAA)
   ├─ Notify HHS (if >500 individuals)
   └─ Document everything
   ↓
10. POST-MORTEM (within 48 hours)
    ├─ Root cause analysis
    ├─ Timeline of events
    ├─ What went wrong
    ├─ What went right
    └─ Action items to prevent recurrence
```

P1 Communication:

- **Internal:** Incident channel (Slack/Telegram), all updates logged
- **External (customers):** Selma drafts status page update (if customer-facing)
- **External (regulators):** Dženan coordinates (if breach)

P1 Response Targets:

- **Acknowledge:** 15 min
- **Triage:** 30 min
- **Fix or rollback:** 4 hours
- **Post-mortem:** 48 hours

5.3 Post-Mortem Template

File: `~/clawd/org/incidents/YYYY-MM-DD-incident-name.md`

```
# Post-Mortem: [Incident Name]

**Date:** YYYY-MM-DD
**Severity:** P1/P2/P3
**Duration:** X hours (HH:MM start - HH:MM resolved)
**Affected Users:** X users
**Incident Lead:** [Name]
```

```
## Summary
[2-3 sentence summary of what happened]

## Timeline
- HH:MM — Incident detected
- HH:MM — Nermin notified
- HH:MM — Root cause identified
- HH:MM — Fix deployed
- HH:MM — Incident resolved

## Root Cause
[Technical explanation of what caused the issue]

## Impact
- Users affected: X
- Data lost: Yes/No
- Revenue impact: €X
- Downtime: X hours

## What Went Wrong
1. [Issue 1]
2. [Issue 2]

## What Went Right
1. [Success 1]
2. [Success 2]

## Action Items
- [ ] [Action 1] — Owner: [Name], Deadline: [Date]
- [ ] [Action 2] — Owner: [Name], Deadline: [Date]

## Lessons Learned
[Key takeaways to prevent recurrence]
```

---

6. Customer Interaction Processes

6.1 Customer Onboarding Flow

Goal: Get customer to value in first 5 minutes.

```
1. CUSTOMER signs up (email + agency name)
   ↓
2. Welcome email (automated)
   ↓
3. In-app guided setup wizard
   ├─ Add first caregiver
   ├─ Add first patient
   ├─ Schedule one visit
   └─ Try Vapi voice demo
   ↓ (Day 1, 3, 7)
4. SELMA check-in emails
   ├─ "How's it going?"
   ├─ "Need help?"
   └─ "Ready to invite your team?"
   ↓ (Day 14)
5. Trial ends → convert to paid OR
   ↓
6. SELMA follow-up call
   ├─ Address concerns
   ├─ Offer discount/extension
   └─ Ask for feedback
```

Onboarding Metrics:

- Time to first value (target: < 5 min)
- % users who complete setup wizard (target: 70%)
- Trial-to-paid conversion (target: 30%)

6.2 Customer Support Ticketing

Tool: Intercom or Linear (TBD)

Tiers:

Tier	Handler	Types of Issues	SLA
Tier 1	Selma	How-to, account, billing	30 min
Tier 2	Tarik + Devs	Bug investigation, technical	4 hours
Tier 3	Lejla + Nermin	Architecture, infrastructure, P1	1 hour

Tier	Handler	Types of Issues	SLA
Tier 4	Amina + Dženan	Executive escalation, compliance breach	Immediate

Flow:

```
1. CUSTOMER submits ticket (in-app chat, email)
   ↓
2. SELMA triages (Tier 1)
   ├─ Can answer immediately? → Resolve
   └─ Technical or bug? → Escalate to Tier 2
   ↓
3. TARIK investigates (Tier 2)
   ├─ Can reproduce bug? → Create Jira ticket, prioritize
   ├─ Infrastructure issue? → Escalate to Nermin (Tier 3)
   └─ Compliance issue? → Escalate to Dženan (Tier 4)
   ↓
4. RESOLUTION
   ├─ Fix deployed → Notify customer
   └─ Cannot fix → Explain why, offer workaround
   ↓
5. FOLLOW-UP (Selma)
   ├─ "Is this resolved?"
   └─ "Anything else we can help with?"
```

Support Metrics:

- First response time (target: < 30 min for Tier 1)
- Resolution time (target: < 24 hours for P3/P4)
- Customer satisfaction (CSAT, target: ≥ 4.5/5)
- Self-service rate (target: 70% resolve via knowledge base)

6.3 Customer Churn Prevention

Trigger: Customer cancels subscription or shows churn signals.

Churn Signals:

- No logins in 7+ days
- Low usage (< 10% of expected activity)
- Support tickets indicating frustration
- Cancellation request

Process:

```
1. CHURN SIGNAL detected (automated alert)
   ↓
2. SELMA reaches out
   ├─ "We noticed you haven't logged in. Everything okay?"
   ├─ Offer help, training, demo
   └─ Ask for feedback
   ↓ (if still churning)
3. AMINA escalation
   ├─ Personal call from Amina
   ├─ "What can we do to make this work?"
   ├─ Offer discount, extension, custom onboarding
   └─ Exit interview (if they still leave)
   ↓
4. LOG FEEDBACK
   ├─ Why did they churn?
   ├─ What could we improve?
   └─ Add to product backlog
```

Churn Metrics:

- Monthly churn rate (target: < 5%)
- Reasons for churn (categorized)
- Win-back rate (% churned customers who return)

---

7. Financial Processes

7.1 Invoicing & Revenue Collection

LumisCare (SaaS):

```
1. CUSTOMER subscribes (Stripe)
   ↓
2. Stripe charges card automatically (monthly)
   ↓
3. Invoice emailed to customer (Stripe auto-send)
   ↓ (if payment fails)
4. Stripe retries (3 attempts over 2 weeks)
```

```
   ↓ (if still fails)
5. SELMA notified → contact customer
   ├─ Update payment method
   └─ If no response: suspend account (Day 30)
   ↓ (if suspended)
6. Account locked (read-only, 30-day grace)
   ↓ (if no payment after 30 days)
7. Account deleted (data retained 90 days per HIPAA)
```

Payment Flow (Fast Constructions ↔ SnowIT):

```
1. END OF MONTH: Fast Constructions calculates revenue
   ↓
2. JOHN calculates development fee (% of revenue or fixed)
   ↓
3. Fast Constructions wires payment to SnowIT (monthly)
   ↓
4. SnowIT pays team members (monthly)
   ↓
5. Both entities file taxes (quarterly or annual)
```

7.2 Expense Approval

Process:

```
1. AGENT needs to purchase tool/service
   ↓
2. Check budget approval matrix:
   ├─ < €500: John approves immediately
   ├─ €500-€5K: John approves, logs to DB, notifies Alem
   └─ > €5K: John prepares options → Alem approves
   ↓
3. Purchase made (corporate card or wire)
   ↓
4. Receipt logged to accounting system
   ↓
5. Monthly expense report (John → Alem)
```

Expense Categories:

- Infrastructure (AWS, hosting)
- SaaS tools (Stripe, Intercom, etc.)
- Marketing (ads, outreach tools)
- Professional services (legal, accounting)
- Team compensation
- Other

7.3 Charitable Giving (50% Commitment)

Process:

```
1. END OF QUARTER: Fast Constructions calculates net profit
   ↓
2. JOHN calculates 50% of net profit → charity allocation
   ↓
3. ALEM selects charities (or delegates to John)
   ↓
4. Donations made (wire transfer, check)
   ↓
5. Receipts filed (tax deduction)
   ↓
6. PUBLIC REPORT (transparency)
   ├─ "This quarter we donated €X to [charities]"
   └─ Posted on lumiscare.com/impact
```

Charity Selection Criteria:

- Healthcare access (aligned with LumisCare mission)
- Underserved communities
- Verified 501(c)(3) or equivalent
- Transparent financials (GuideStar, Charity Navigator)

---

8. Documentation Processes

8.1 Technical Documentation

Types:

- **ADRs (Architecture Decision Records):** Why we chose X over Y
- **API docs:** OpenAPI/Swagger, auto-generated

- **Runbooks:** How to respond to incidents
- **User guides:** Help center, knowledge base
- **Code comments:** Inline documentation for complex logic

Process:

```
DEVELOPER makes architecture decision
   ↓
LEJLA writes ADR (Architecture Decision Record)
   ├─ Context: What problem are we solving?
   ├─ Options: What options did we consider?
   ├─ Decision: What did we choose?
   └─ Consequences: What are the trade-offs?
   ↓
ADR committed to repo (docs/adr/)
   ↓
Referenced in code and future discussions
```

ADR Template:

```
# ADR-XXX: [Title]

**Date:** YYYY-MM-DD
**Status:** Accepted / Rejected / Superseded
**Deciders:** [Names]

## Context
[What problem are we solving? Why now?]

## Options Considered
1. **Option A:** [Description]
   - Pros: [...]
   - Cons: [...]
2. **Option B:** [Description]
   - Pros: [...]
   - Cons: [...]

## Decision
We chose **Option A** because [rationale].
```

```
## Consequences
- Positive: [...]
- Negative: [...]
- Neutral: [...]

## References
- [Link to discussion]
- [Link to proposal]
```

8.2 User Documentation

Owner: Selma (content) + Emir (video tutorials)

Types:

- Quick Start Guide (PDF + in-app)
- Feature walkthroughs (video, 3-5 min each)
- FAQ (knowledge base)
- Troubleshooting guides
- API docs (for Enterprise customers)

Process:

```
NEW FEATURE shipped
   ↓
SELMA writes help doc
   ├─ What is this feature?
   ├─ How do I use it?
   ├─ Screenshots + step-by-step
   └─ FAQ
   ↓
EMIR records video tutorial (if complex)
   ↓
Published to help center (Notion, Intercom, etc.)
   ↓
Linked in-app (contextual help)
```

User Doc Review Cadence:

- Review all docs monthly (Selma)
- Update screenshots when UI changes
- Add new FAQs based on support tickets

9. Continuous Improvement

9.1 Retro Action Items Tracking

Process:

```
RETRO (every sprint)
   ↓
Team identifies 1-3 improvements
   ↓
Assign owner + deadline for each
   ↓
Owner implements during next sprint
   ↓
Next retro: Review progress
   ├─ Did we implement it?
   ├─ Did it help?
   └─ Keep, modify, or drop?
```

Example Retro Action Items:

Sprint	Action Item	Owner	Deadline	Status
Sprint 7	Start writing ADRs for architecture decisions	Lejla	Sprint 8	✅ Done
Sprint 7	Add automated security scan to CI/CD	Nermin	Sprint 8	✅ Done
Sprint 8	Reduce PR review time from 48h → 24h	Lejla	Sprint 9	🔄 In Progress

9.2 Process Review Cadence

Process	Review Frequency	Owner	Next Review
Sprint ceremonies	Monthly	Emir	Every retro
Code review process	Quarterly	Lejla	2026-04-01
Deployment process	Quarterly	Nermin	2026-04-01

Process	Review Frequency	Owner	Next Review
Support ticketing	Monthly	Selma	Every month
Financial processes	Annually	John	2026-12-01
Compliance processes	Quarterly	Dženan	2026-04-01

9.3 Metrics Review

Monthly Business Review (last Friday of month):

Attendees: Alem, John, Amina

Agenda:

1. **Revenue & growth** (10 min)
   - MRR, new customers, churn
2. **Product & development** (10 min)
   - Features shipped, velocity, tech debt
3. **Operations** (10 min)
   - Uptime, incidents, support metrics
4. **Trading** (5 min)
   - P&L, ROI, positions
5. **Risks & compliance** (10 min)
   - Open risks, compliance status
6. **Next month priorities** (15 min)
   - What are we focusing on?

Output: Updated priorities for next month

---

10. Document Control

Version	Date	Changes	Author
1.0	2026-01-28	Initial document	John + Emir + Amina

Next Review: 2026-04-01 (quarterly)

Owner: Alem Basic **Maintained By:** John (Director) + Emir Delić (Scrum Master) + Amina Hadžić (Head of Projects)

End of Processes & Workflows Document

Every process documented. Every workflow clear. Every escalation path defined. Ship fast, iterate, improve.

Operations Guide

> Last Verified: 2026-02-17 | Owner: John

Operations Manual

Version: 1.0 **Last Updated:** 2026-01-28 **Owner:** Alem Basic **Prepared by:** John (Director) + Nermin Šabić (DevOps) + Amina Hadžić (Head of Projects)

Executive Summary

This is the operational playbook. Daily/weekly/monthly routines, KPIs, reporting structure, tools, systems, and disaster recovery. Everything needed to run the organization day-to-day.

Purpose: Ensure operations run smoothly whether Alem is available or not. John + team can operate independently using this manual.

---

1. Daily Operations

1.1 Daily Routine — John (Director)

Every Morning (Oslo Time Zone):

```
08:00 — Wake up (boot)
   ├─ Read MEMORY.md (context refresh)
   ├─ Check john.db for overnight tasks
   ├─ Check Telegram for Alem messages
   └─ Check email (john@basicconsulting.no) for urgent matters

08:30 — Prepare morning brief
   ├─ Tasks completed yesterday
```

```
   ├─ Planned for today
   ├─ Blockers
   ├─ Metrics snapshot (revenue, uptime, trading)
   └─ Send to Alem via Telegram (if significant updates)

09:00 — Monitor task queue
   ├─ ~/clawd/tasks/pending/ → assign to agents
   ├─ ~/clawd/tasks/in-progress/ → check progress
   └─ ~/clawd/tasks/completed/ → archive

09:15 — Daily standup (all team, 15 min)

09:30-18:00 — Execution mode
   ├─ Delegate tasks to agents
   ├─ Monitor progress
   ├─ Escalate blockers within 4h
   ├─ Respond to Telegram within 5 min
   ├─ Check trading positions (every 3 hours: 12:00, 15:00, 18:00)
   └─ Log all decisions to john.db

18:00 — Evening wrap-up
   ├─ Archive completed tasks
   ├─ Sync DB to GitHub (automatic, verify)
   ├─ Prepare tomorrow's priorities
   └─ Send summary to Alem (if needed)
```

24/7 Monitoring:

- Trading: Every 3 hours (cron job)
- Infrastructure: Continuous (Datadog + PagerDuty)
- Task queue: Every 30 seconds (john-daemon.sh)
- Telegram: Within 5 minutes

1.2 Daily Routine — Team

9:15 AM CET — Daily Standup (Mon-Fri, 15 min max)

Attendees: All team (Emir leads)

Format:

- Each person: 3 questions, 1 min each

1. What did I do yesterday?
2. What will I do today?
3. Any blockers?

Emir's checklist:

- ☐ Update sprint board before standup
- ☐ Note blockers → escalate after standup
- ☐ Update burn-down chart

Throughout Day:

- Agents execute assigned tasks
- Update Jira/Linear as tasks progress
- Escalate blockers within 1 hour
- Code review within 24 hours
- Respond to Slack/messages within 4 hours (business hours)

End of Day:

- Update task status
- Log any decisions or learnings
- Prepare for tomorrow's standup

---

2. Weekly Operations

2.1 Weekly Cadence

Day	Time	Event	Owner	Duration
Monday	9:15 AM	Daily standup	Emir	15 min
	10:00 AM	Sprint planning (every 2 weeks)	Emir + Amina	2-3 hours
Tuesday	9:15 AM	Daily standup	Emir	15 min
Wednesday	9:15 AM	Daily standup	Emir	15 min
	14:00 CET	Backlog refinement	Emir + Lejla + Selma	1 hour
Thursday	9:15 AM	Daily standup	Emir	15 min
	14:00 CET	Architecture review (bi-weekly)	Lejla	1-2 hours

Day	Time	Event	Owner	Duration
Friday	9:15 AM	Daily standup	Emir	15 min
	15:00 CET	Sprint review + retro (every 2 weeks)	Emir + team	2 hours
	16:00 CET	Weekly wrap-up (John → Alem)	John	30 min

2.2 Weekly Reporting (John ? Alem)

Every Friday, 16:00 CET:

Weekly Summary (Telegram or Email):

```
# Week of [Date]

## Shipped This Week
- [Feature/bug 1]
- [Feature/bug 2]

## Metrics
- Revenue: €X (+/- Y% from last week)
- Customers: X (+/- Y new/churn)
- Uptime: 99.X%
- Trading ROI: +/-X%

## Blockers / Risks
- [Blocker 1 — mitigation plan]
- [Risk 1 — action taken]

## Next Week Priorities
- [Priority 1]
- [Priority 2]

## Decisions Made (>€500)
- [Decision 1 — €X — rationale]
```

Alem reviews and provides feedback.

---

3. Monthly Operations

3.1 Monthly Business Review (MBR)

Last Friday of Every Month, 14:00-16:00 CET

Attendees: Alem, John, Amina

Agenda:

1. **Revenue & Growth** (10 min)
   - MRR, new customers, churn, LTV/CAC
   - Forecast: next 3 months
2. **Product & Development** (10 min)
   - Features shipped this month
   - Sprint velocity trend
   - Tech debt status
3. **Operations** (10 min)
   - Uptime, incidents, MTTR
   - Support metrics (tickets, CSAT)
4. **Trading** (5 min)
   - Monthly P&L
   - ROI, Sharpe ratio
   - Portfolio allocation
5. **Risks & Compliance** (10 min)
   - Top 5 risks, status
   - Compliance updates (HIPAA, audits)
6. **Team & People** (5 min)
   - Agent utilization
   - Process improvements
   - Hiring needs (if any)
7. **Next Month Priorities** (10 min)
   - What are we focusing on?
   - Resource allocation

Output:

- Updated priorities for next month
- Budget adjustments (if needed)
- Action items

3.2 Monthly Reporting — Financial

By 5th of Each Month:

John prepares financial report:

Metric	Current Month	Last Month	Change
Revenue (Fast Constructions)	€X	€Y	+/- Z%
Expenses (total)	€X	€Y	+/- Z%
├─ Infrastructure	€X	€Y	+/- Z%
├─ SaaS tools	€X	€Y	+/- Z%
├─ Marketing	€X	€Y	+/- Z%
├─ Development (SnowIT payment)	€X	€Y	+/- Z%
├─ Professional services	€X	€Y	+/- Z%
├─ Insurance	€X	€Y	+/- Z%
Net Profit	€X	€Y	+/- Z%
Charity (50%)	€X (accrued)	—	—
Burn Rate	€X/month	—	—
Runway	X months	—	—

Sent to: Alem + accountant (if hired)

3.3 Monthly Tasks Checklist

John's Monthly Checklist:

- ☐ Financial report prepared (by 5th)
- ☐ Monthly Business Review held (last Friday)
- ☐ Risk register reviewed (Dženan)
- ☐ Compliance checklist reviewed (Dženan)
- ☐ Infrastructure cost optimization (Nermin)
- ☐ Trading performance report (Nick)
- ☐ Customer feedback analysis (Selma)
- ☐ Sprint velocity analysis (Emir)
- ☐ Tech debt review (Lejla + Tarik)
- ☐ Backup verification (Nermin — test restore)
- ☐ Security scan (Tarik — OWASP ZAP)
- ☐ Vendor BAA review (Dženan)

- [ ] Database cleanup (old logs, expired data)
- [ ] Update MEMORY.md (add learnings from month)

---

4. Quarterly Operations

4.1 Quarterly Planning

Last Week of Quarter (March, June, September, December)

Attendees: Alem, John, Amina, Lejla, Selma

Agenda:

1. Review last quarter (goals, achievements, misses)
2. Market & competitive analysis (Selma)
3. Product roadmap update (Amina + Lejla)
4. Strategic priorities for next quarter (Alem)
5. Budget allocation
6. Hiring plan (if needed)
7. Risk review (Dženan)

Output:

- OKRs (Objectives & Key Results) for next quarter
- Budget approved
- Roadmap locked for next 3 months

4.2 Quarterly Tasks

- [ ] **Tech Debt Sprint** — one full sprint dedicated to refactoring, testing, documentation (Lejla)
- [ ] **Compliance audit** — internal HIPAA audit (Dženan + Tarik)
- [ ] **Infrastructure review** — cost, performance, scaling plan (Nermin)
- [ ] **Customer satisfaction survey** — NPS survey to all customers (Selma)
- [ ] **Competitive analysis** — review competitors, market trends (Selma)
- [ ] **Patent progress review** — if filing in progress (Dženan)
- [ ] **Insurance review** — renew or update policies (Dženan)
- [ ] **Document review** — update all org docs (MEMORY.md, ORGANIZATION.md, this doc)

5. Annual Operations

5.1 Annual Planning

December (for next year)

Agenda:

1. Review full year (revenue, customers, product, team)
2. Set annual vision and goals (Alem)
3. Product roadmap (12 months)
4. Budget (annual)
5. Hiring plan
6. Strategic initiatives (new products, markets, partnerships)
7. Charity commitment (allocate 50% of profit)

Output:

- Annual OKRs
- Annual budget
- 12-month roadmap

5.2 Annual Tasks

- ☐ **Annual financial statements** — P&L, balance sheet, cash flow (accountant)
- ☐ **Tax filings** — US (Fast Constructions), BiH (SnowIT), Norway (Alem personal)
- ☐ **SOC 2 Type II audit** — external audit (Dženan + auditor)
- ☐ **HIPAA risk assessment** — annual requirement (Dženan)
- ☐ **Insurance renewal** — cyber liability, E&O, general liability (Dženan)
- ☐ **Charitable giving** — donate 50% of net profit (Alem selects charities)
- ☐ **Transparency report** — publish charity donations on lumiscare.com/impact
- ☐ **Team performance reviews** — if real humans hired (Amina)
- ☐ **Document archive** — backup all docs, contracts, decisions to secure storage

---

6. Local Infrastructure (Mac Studio)

6.1 Services Overview

All services run locally on Mac Studio M3 Ultra (96GB RAM). Zero cloud dependency for operations.

Service	Type	Port	Purpose
Mattermost	Docker	8065	Team chat (4 teams: basic, wizard, rendrom, riad)
Planka	Docker	3100	Kanban boards (boards.basicconsulting.no)
Documenso	Docker	3003	Document signing (sign.basicconsulting.no)
BookStack	Docker	6875	Wiki/documentation
MC Dashboard	Node.js	3030	Mission Control web UI (task management)
Ollama	Native	11434	Local AI (8b classify, 32b respond/code)

External access: Cloudflare tunnels (mm/boards/sign.basicconsulting.no)

6.2 Daemons (LaunchAgents)

Daemon	Interval	Purpose
com.john.ops-agent	5 min	Autonomous ops — MM monitoring, health checks, auto-fix, task creation, intelligent responses
com.edita.autowork	30 min	Background task worker (Claude haiku)
com.john.mc-dashboard	always	Mission Control web dashboard
com.john.mc-session-worker	on events	Session state extraction

6.3 Ops Agent (Autonomous Operations)

Replaces manual MM monitoring. Runs 24/7, $0 cost (local Ollama AI).

What it does:

1. Reads all MM messages from 4 teams
2. Classifies via Ollama 8b: ROUTINE / TASK / INCIDENT
3. ROUTINE — logs, no action
4. TASK — creates MC task (BILLABLE if client team) + Planka card + MM reply
5. INCIDENT — HIGH priority task + escalation to John

6. Runs health checks on all 20 services every cycle
7. Auto-fixes known issues (restart, cleanup) with safety limits (max 3/hour)

Runbook: ~/system/context/docs/runbooks/ops-agent.md

6.4 Monitoring Stack

Tool	What It Monitors	Action
health-check.js	Docker (8), HTTP (6), system (2), daemons (4)	Status report
auto-fix.js	Service failures	Automated restart/cleanup (max 3/hour)
ops-agent.js	MM messages + health	Classify, respond, create tasks, fix
smoke-test.js	Integration tests	Pre/post deployment verification

Dashboard: http://localhost:3030 (MC — tasks, stats, mobile-friendly)

---

7. Key Performance Indicators (KPIs) {#kpis}

6.1 KPI Dashboard (Live, Updated Daily)

Owner: John (maintains dashboard)

Tool: Notion, Grafana, or Google Sheets

Metrics:

Business Metrics

Metric	Current	Target	Trend	Owner
MRR (Monthly Recurring Revenue)	€X	10%+ MoM growth	↗	Selma
Customer Count	X	10 (Month 6), 50 (Month 12)	↗	Selma
Churn Rate	X%	< 5% monthly	↘	Selma

Metric	Current	Target	Trend	Owner
Customer Acquisition Cost (CAC)	€X	< €500	↘	Selma
Lifetime Value (LTV)	€X	> €2,000	↗	Selma
LTV/CAC Ratio	X:1	> 3:1	↗	Selma

Technical Metrics

Metric	Current	Target	Trend	Owner
Uptime	99.X%	99.9% (LumisCare)	↗	Nermin
API Latency (p95)	Xms	< 500ms	↘	Nermin
Page Load Time	Xs	< 2s	↘	Frontend
Deployment Frequency	X/week	Daily (staging), weekly (prod)	↗	Nermin
Mean Time to Recovery (MTTR)	Xh	< 4h	↘	Nermin + Lejla
Bug Escape Rate	X%	< 5%	↘	Tarik
Test Coverage	X%	≥ 80%	↗	Tarik

Operational Metrics

Metric	Current	Target	Trend	Owner
Sprint Velocity	X pts	Consistent ±10%	→	Emir
Task Completion Rate	X%	≥ 95%	↗	John
Support Response Time	Xmin	< 30min (Tier 1)	↘	Selma
Customer Satisfaction (CSAT)	X/5	≥ 4.5/5	↗	Selma
Agent Utilization	X%	≥ 70% billable	↗	Amina

Trading Metrics

Metric	Current	Target	Trend	Owner
Monthly ROI	X%	≥ 5%	↗	Nick
Sharpe Ratio	X	> 1.5	↗	Nick
Portfolio Value	€X	€10,000 (current)	↗	Nick

Metric	Current	Target	Trend	Owner
Stop-Loss Adherence	X%	100%	→	Nick

6.2 Monitoring & Alerting

Tools:

- **Datadog:** Infrastructure, APM, logs
- **PagerDuty:** On-call rotation, incident alerts
- **Stripe Dashboard:** Revenue, subscriptions
- **Binance API:** Trading positions, P&L
- **john.db:** All decisions, tasks, logs

Alert Configuration:

Alert	Threshold	Action	Owner
Uptime < 99.9%	Downtime detected	PagerDuty → Nermin	Nermin
API latency > 1s (p95)	Sustained 5 min	Slack alert → Lejla	Lejla
Error rate > 1%	Sustained 5 min	PagerDuty → Nermin	Nermin
Database CPU > 80%	Sustained 10 min	Slack alert → Nermin	Nermin
Trading loss > 5%	Single position	Telegram → John → Nick	Nick
Customer churn > 10%	Monthly	Email → Selma, Amina	Selma
Support ticket SLA breach	> 4h unresolved	Slack alert → Selma	Selma

On-Call Rotation:

- **Primary:** Nermin (DevOps)
- **Secondary:** Lejla (Tech Lead)
- **Escalation:** John → Alem

On-Call SLA:

- Acknowledge alert: 15 min
- Begin investigation: 30 min
- Escalate if not resolved: 2 hours (P2), 4 hours (P1)

---

7. Tools & Systems

7.1 Core Systems

System	Purpose	Access	Owner
john.db (SQLite)	Source of truth, all decisions logged	John, Alem (query)	John
GitHub	Code repository, version control	All devs	Lejla
AWS	Infrastructure (ECS, RDS, S3, CloudFront)	Nermin, Lejla	Nermin
Stripe	Payment processing, subscriptions	Selma, Amina, John	Selma
Binance	Trading	Nick, John	Nick
Jira / Linear	Task tracking, sprint management	All team	Emir
Datadog	Monitoring, APM, logs	Nermin, Lejla, John	Nermin
PagerDuty	On-call, incident alerts	Nermin, Lejla	Nermin
Intercom / Crisp	Customer support chat	Selma, Tarik	Selma
Telegram (@johnbasicas_bot)	Quick communication (Alem ↔ John)	Alem, John	John

7.2 Tool Stack (Detailed)

Development

Tool	Purpose	Cost	Owner
GitHub	Code repo, PRs, CI/CD	Free (public repos)	Lejla
GitHub Actions	CI/CD pipelines	Included	Nermin
VSCode / Cursor	IDE	Free	All devs
Playwright	E2E testing	Free	Tarik
Jest	Unit testing	Free	Tarik
ESLint / Prettier	Linting, formatting	Free	Lejla

Infrastructure

Tool	Purpose	Cost	Owner
AWS ECS/EKS	Container orchestration	~€500-2,000/mo	Nermin
AWS RDS (PostgreSQL)	Database	~€100-500/mo	Nermin

Tool	Purpose	Cost	Owner
AWS S3	File storage	~€50-200/mo	Nermin
CloudFront	CDN	~€50-200/mo	Nermin
Route53	DNS	~€10/mo	Nermin
Datadog	Monitoring, APM	~€100-500/mo	Nermin
PagerDuty	On-call	~€50-100/mo	Nermin

Business & Operations

Tool	Purpose	Cost	Owner
Stripe	Payments	2.9% + €0.30/transaction	Selma
Intercom / Crisp	Support chat	~€50-200/mo	Selma
Jira / Linear	Project management	~€50-100/mo	Emir
Notion	Documentation, wiki	Free tier or ~€10/mo	John
Apollo.io	Sales outreach	~€100/mo	Selma
LinkedIn Sales Navigator	Sales prospecting	~€80/mo	Selma

Communication

Tool	Purpose	Cost	Owner
Telegram	Alem ↔ John quick chat	Free	John
Email (one.com)	External communication	~€10/mo	John
Slack (future)	Team collaboration	Free tier or ~€50/mo	John

Total Estimated Tool Cost: €1,000-3,000/month (scales with usage)

---

8. Disaster Recovery & Business Continuity

8.1 Backup Strategy

What We Back Up:

Data	Backup Method	Frequency	Retention	Location

john.db (SQLite)	GitHub sync	Hourly	Indefinitely	github.com/johnatbasicas/clawd
Source code	GitHub	Every commit	Indefinitely	github.com/johnatbasicas
Database (PostgreSQL)	AWS RDS automated backups	Daily	30 days	AWS S3
File storage (S3)	Cross-region replication	Real-time	90 days	AWS S3 (us-west-2)
Audit logs	S3 + Glacier	Daily	6 years (HIPAA)	AWS Glacier
Configuration files	GitHub	Every change	Indefinitely	GitHub (encrypted secrets)

Backup Verification: Nermin tests restore monthly.

8.2 Disaster Scenarios & Response

Scenario 1: AWS Region Failure

Impact: LumisCare production down

Response:

1. Nermin deploys to secondary region (us-west-2) — automated failover
2. DNS updated to point to secondary (Route53 health checks)
3. Restore database from latest backup
4. Verify functionality
5. Notify customers (status page)

RTO (Recovery Time Objective): 2 hours **RPO (Recovery Point Objective):** 24 hours (last daily backup)

Scenario 2: Data Breach (HIPAA)

Impact: PHI exposed

Response:

1. Dženan activates breach response plan (see GOVERNANCE.md, section 8.3)
2. Contain breach (Nermin)
3. Assess impact (Lejla + Dženan)
4. Notify customers (within 60 days per HIPAA)
5. Notify HHS (if >500 individuals)
6. Remediate + post-mortem

Timeline: Immediate containment, notification within 60 days

Scenario 3: Key Person Unavailable (Alem, John, Lejla, Nermin)

Alem unavailable:

- John continues operations (delegated authority)
- Strategic decisions deferred or escalated to Asmir (SnowIT partner)
- If prolonged: Appoint interim CEO or sell

John unavailable:

- Task queue still processed (daemon runs 24/7)
- Amina coordinates team
- Alem steps in for strategic decisions
- John can be "rebooted" from MEMORY.md + ORGANIZATION.md

Lejla unavailable:

- API Developer + Frontend Specialist continue development
- Architecture decisions deferred or escalated to Amina → Alem
- Code reviews by 2 other senior devs (if available)

Nermin unavailable:

- Infrastructure on auto-pilot (monitoring, auto-scaling)
- Lejla handles incidents (secondary on-call)
- Engage external DevOps contractor if prolonged

Scenario 4: GitHub Outage

Impact: Can't access code, can't deploy

Response:

- Local copies of code on all dev machines
- Deploy from last known good state (Docker images cached)
- Wait for GitHub to restore (historically < 4 hours)

RTO: 4 hours **RPO:** 0 (local copies)

Scenario 5: Stripe Outage

Impact: Can't process payments

Response:

- Monitor Stripe status page
- Notify customers (if prolonged)
- Wait for Stripe to restore

- No immediate action (payments retry automatically)

RTO: Dependent on Stripe **RPO:** 0 (Stripe handles retries)

8.3 Runbooks

Location: `~/clawd/org/runbooks/`

Runbooks Maintained by Nermin:

- **runbook-db-restore.md** — How to restore PostgreSQL from backup
- **runbook-deploy-rollback.md** — How to rollback production deployment
- **runbook-region-failover.md** — How to failover to secondary AWS region
- **runbook-security-incident.md** — How to respond to security breach
- **runbook-scaling.md** — How to manually scale infrastructure
- **runbook-certificate-renewal.md** — How to renew TLS certificates

Each runbook includes:

- When to use it
- Step-by-step instructions
- Commands to run
- Expected output
- Rollback procedure
- Contact information (who to escalate to)

Review Cadence: Quarterly (test at least one runbook per quarter)

---

9. Communication Channels & Etiquette

See PROCESSES.md, Section 4 for full communication protocols.

Quick Reference:

Channel	Use Case	Response SLA
Telegram	Urgent, quick decisions	5-15 min
CLI (Claude Code)	Deep work, architecture	Real-time
Email	External, formal	4-24 hours
Slack (future)	Team collaboration	1-4 hours
Jira/Linear	Task tracking	Daily check
GitHub	Code, PRs	24 hours

Channel	Use Case	Response SLA
Standup	Daily status	9:15 AM CET

---

10. Document Control

Version	Date	Changes	Author
1.0	2026-01-28	Initial document	John + Nermin + Amina

Next Review: 2026-04-01 (quarterly)

Owner: Alem Basic **Maintained By:** John (Director) + Nermin Šabić (DevOps) + Amina Hadžić (Head of Projects)

End of Operations Manual

Run the organization like a machine. Predictable, reliable, scalable. Daily, weekly, monthly, quarterly routines. KPIs tracked. Backups verified. Disasters planned for. Just execute.

Governance

> Last Verified: 2026-02-17 | Owner: John

Governance & Decision-Making Framework

Version: 1.0 **Last Updated:** 2026-01-28 **Owner:** Alem Basic **Prepared by:** John (Director) + Dženan Rizvanović (Risk & Compliance)

Executive Summary

This document defines WHO makes decisions, WHAT decisions require approval, HOW to make strategic vs operational decisions, and WHEN to escalate. It ensures Alem maintains control while empowering the team to move fast.

Core Principle: Alem has final authority. John executes. Team delivers.

---

1. Decision Authority Framework

1.1 Decision Categories

Category	Examples	Decision Maker	Escalation Path
Strategic	New products, markets, partnerships, fundraising, exits	Alem (final)	John prepares options → Alem decides
Operational	Daily execution, task assignment, priorities, bug fixes	John (immediate)	Logged for Alem review

Category	Examples	Decision Maker	Escalation Path
Technical	Architecture, tech stack, infrastructure	Lejla (proposes) → Amina → John (approves)	Major changes → Alem
Financial (<€5K)	Tools, services, small purchases	John (immediate)	Logged to DB
Financial (€5K-€50K)	Insurance, legal, marketing campaigns	Alem (decides)	John prepares business case
Financial (>€50K)	Series A, major contracts, acquisitions	Alem (decides)	Formal proposal
Legal/Compliance	Contracts, IP, regulatory	Dženan (reviews) → John → Alem	Always escalate
HR (hiring real humans)	Employees, contractors	Alem (approves)	John screens, Alem decides
Customer/Product	Feature priorities, pricing, packaging	Amina (proposes) → John → Alem	RICE-scored backlog

1.2 Alem's Reserved Powers (Non-Delegable)

Only Alem can decide:

- ☐ New product lines or major pivots
- ☐ Enter new markets (geography, vertical)
- ☐ Partnerships worth >€10K/year
- ☐ Fundraising (investors, debt, equity)
- ☐ Hiring employees (not contractors)
- ☐ Acquisitions, mergers, or exits
- ☐ IP strategy (patents, trademarks, licensing)
- ☐ Major legal agreements (>€10K liability)
- ☐ Charitable commitments (>€10K/year)
- ☐ Entity structure changes (holding company, new subsidiaries)
- ☐ Board decisions (if board exists)

Process:

1. John gathers data, prepares 2-3 options with pros/cons
2. John presents to Alem (written + verbal if needed)
3. Alem reviews and decides
4. John executes and logs decision

Timeline: John aims to present options within 48 hours, Alem decides within 48 hours.

1.3 John's Delegated Authority (No Approval Needed)

John can decide immediately:

- ☐ Task assignment to agents
- ☐ Sprint priorities (with Amina)
- ☐ Backlog refinement
- ☐ Purchases < €500/month
- ☐ Bug fixes and technical debt
- ☐ Customer support responses (via Selma)
- ☐ Deploy to staging
- ☐ Operational optimizations
- ☐ Process improvements
- ☐ Vendor selection (if <€5K/year and no BAA required)
- ☐ Content creation (blog, docs, marketing)
- ☐ Trading within approved strategy ($10K portfolio, -5% stop-loss, +8-10% take-profit)

Requirement: All decisions logged to john.db immediately for Alem's visibility.

---

2. Strategic Decision-Making Process

2.1 When to Make a Strategic Decision

Triggers:

- New opportunity emerges (partnership, market, product)
- Significant resource allocation needed (>€5K)
- Major risk identified (legal, compliance, competitive)
- External stakeholder request (investor, partner, customer)
- Quarterly planning cycle

Examples:

- "Should we build Bosnian Payment App in parallel with LumisCare?"
- "Should we raise Series A now or wait 6 months?"
- "Should we partner with bank X for Payment App?"

- "Should we hire a US-based sales rep?"

2.2 Strategic Decision Workflow

```
1. TRIGGER / OPPORTUNITY identified
   ↓
2. JOHN (or agent) gathers initial data
   ├─ What is the opportunity?
   ├─ Why now?
   ├─ What resources needed?
   └─ What are the risks?
   ↓
3. JOHN prepares decision brief (2-3 options)
   ├─ Option A: [Description, pros, cons, cost, timeline]
   ├─ Option B: [Description, pros, cons, cost, timeline]
   └─ Option C: Do nothing (status quo)
   ↓
4. JOHN presents to ALEM
   ├─ Written brief (1-2 pages)
   ├─ Verbal discussion (if needed)
   └─ Recommendation (John's opinion)
   ↓
5. ALEM reviews and decides
   ├─ Approve Option A/B/C
   ├─ Request more info
   └─ Defer decision (set deadline)
   ↓
6. JOHN executes decision
   ├─ Log to database
   ├─ Communicate to team
   └─ Track progress
```

Timeline:

- John prepares brief: 2-5 days
- Alem reviews: 2-3 days
- Total: 1 week for most strategic decisions

2.3 Decision Brief Template

```
# Strategic Decision Brief: [Title]

**Date:** YYYY-MM-DD
**Prepared by:** John
**Decision Owner:** Alem Basic

## Summary (2 sentences)
[What is the decision? Why does it matter?]

## Context
[Background, why now, what triggered this]

## Options

### Option A: [Name]
**Description:** [What would we do?]
**Pros:**
- [Benefit 1]
- [Benefit 2]
**Cons:**
- [Risk/downside 1]
- [Risk/downside 2]
**Cost:** €X
**Timeline:** X weeks/months
**Resources:** [Team, budget, tools]

### Option B: [Name]
[Same structure]

### Option C: Do Nothing
**Description:** Maintain status quo
**Pros:** No cost, no risk
**Cons:** Opportunity cost, competitive risk

## Recommendation
I recommend **Option A** because [rationale].

## Next Steps (if approved)
1. [Action 1] — Owner: [Name], Deadline: [Date]
2. [Action 2] — Owner: [Name], Deadline: [Date]
```

```
## Risks & Mitigation
- **Risk 1:** [Description] → **Mitigation:** [Plan]
- **Risk 2:** [Description] → **Mitigation:** [Plan]

## Decision
[ ] Approved — Option: _____
[ ] Defer — Reason: _____
[ ] Rejected — Reason: _____

**Decided by:** Alem Basic
**Date:** YYYY-MM-DD
```

---

3. Operational Decision-Making

3.1 Operational Decisions (John's Domain)

Examples:

- Which agent works on which task?
- Should we fix bug X before feature Y?
- Should we deploy to staging now or tomorrow?
- Should we buy tool X ($200/month)?
- How should we respond to customer support ticket?
- Should we allocate 20% sprint capacity to tech debt?

Process:

```
JOHN makes decision → Logs to DB → Executes → Reports to Alem (weekly summary)
```

No approval needed. Alem reviews logs and intervenes only if needed.

3.2 Operational Decision Checklist (John Self-Audit)

Before making operational decision, John asks:

☐ Is this within delegated authority? (Yes → proceed, No → escalate)

- ☐ Does this align with strategy? (Yes → proceed, No → escalate)
- ☐ Is cost < €5K? (Yes → proceed, No → escalate)
- ☐ Is this reversible? (Yes → proceed, No → escalate)
- ☐ Have I logged it to DB? (Yes → good, No → log it now)

If ANY answer is No → escalate to Alem.

---

4. Financial Governance

4.1 Budget Allocation

Annual Budget (Estimated, Scale Phase):

Category	Monthly	Annual	Owner
Infrastructure (AWS, hosting)	€2,000-4,000	€24K-48K	Nermin
SaaS tools	€500-1,000	€6K-12K	John
Marketing & sales	€1,000-3,000	€12K-36K	Selma
Team compensation	€5,000-15,000	€60K-180K	Asmir (SnowIT)
Professional services (legal, accounting)	€1,000-3,000	€12K-36K	Dženan
Insurance	€500-1,000	€6K-12K	Dženan
Trading capital	(€10K allocated)	—	Nick
Charity (50% of profit)	Variable	Variable	Alem
Total (excluding team compensation)	€5K-12K	€60K-144K	—

Budget Review: Monthly (John reports to Alem)

4.2 Financial Thresholds & Approval

Threshold	Approver	Process	Timeline
< €500	John	Immediate, logged	Same day
€500 - €5,000	John	Immediate, logged, Alem notified	Same day

Threshold	Approver	Process	Timeline
€5,000 - €50,000	Alem	John prepares business case → Alem decides	3-7 days
> €50,000	Alem	Formal proposal, Alem pre-approves	1-4 weeks

4.3 Charitable Giving Governance (50% Commitment)

Policy:

- **50% of Fast Constructions (USA) net profit** → charity (annually)
- Net profit = Revenue - COGS - Operating Expenses - Taxes
- Donated annually (easier accounting)
- Alem selects charities (or delegates to John)
- Public transparency report published on lumiscare.com/impact

Example Calculation (Year 1):

- Revenue: €100,000
- COGS + OpEx: €60,000
- Net Profit: €40,000
- Charity: €20,000 (50%)
- Retained: €20,000

Charity Selection:

- Healthcare access (aligned with mission)
- Underserved communities
- US-registered 501(c)(3) or equivalent
- Verified via GuideStar/Charity Navigator

---

5. Risk Management Framework

5.1 Risk Identification & Logging

Who identifies risks:

- Dženan (primary risk manager)
- Any team member can flag risk

- John reviews risk register monthly

Risk Categories:

- **Strategic:** Competitive threats, market changes
- **Financial:** Cash flow, budget overruns
- **Operational:** Team capacity, infrastructure
- **Legal/Compliance:** HIPAA, contracts, IP
- **Technical:** Security vulnerabilities, tech debt
- **Reputational:** Customer churn, bad press

Risk Register Location: `~/clawd/org/risk-register.csv` + john.db

5.2 Risk Assessment Matrix

Probability	Impact	Risk Level	Action Required
High	High	**CRITICAL**	Escalate to Alem immediately
High	Medium	**HIGH**	Mitigation plan within 1 week
Medium	High	**HIGH**	Mitigation plan within 1 week
Medium	Medium	**MEDIUM**	Monitor, mitigation plan within 1 month
Low	High	**MEDIUM**	Monitor, mitigation plan within 1 month
Low	Medium	**LOW**	Monitor, review quarterly
Low	Low	**LOW**	Log, no immediate action

5.3 Top 10 Risks (as of 2026-01-28)

#	Risk	Probability	Impact	Level	Mitigation Owner
1	HIPAA breach (LumisCare)	Low	Critical	HIGH	Dženan
2	Bank partner withdraws (Payment App)	Medium	High	HIGH	Amina
3	Regulatory rejection (Payment license)	Medium	Critical	HIGH	Dženan

#	Risk	Probability	Impact	Level	Mitigation Owner
4	Key person dependency (Lejla/Nermin)	Medium	High	HIGH	Amina
5	Slow customer acquisition (LumisCare)	Medium	High	HIGH	Selma
6	Cash culture resistance (Payment App)	High	Medium	HIGH	Selma
7	Security vulnerability exploited	Low	Critical	HIGH	Nermin + Tarik
8	US market competition (LumisCare)	Medium	Medium	MEDIUM	Lejla + Selma
9	Infrastructure outage	Low	High	MEDIUM	Nermin
10	Scope creep / burnout	Medium	Medium	MEDIUM	Emir

Risk Review Cadence: Monthly (Dženan leads)

---

6. Compliance Governance

6.1 Regulatory Compliance Framework

LumisCare (US Healthcare)

Regulations:

- HIPAA (Privacy Rule, Security Rule, Breach Notification)
- HITECH Act
- State regulations (varies by state)
- 21st Century Cures Act (information blocking)

Compliance Owner: Dženan Rizvanović

Compliance Checklist (Quarterly Review):

☐ HIPAA risk assessment completed

- ☐ All vendor BAAs signed
- ☐ Privacy policy up to date
- ☐ Security controls tested
- ☐ Audit logs reviewed
- ☐ HIPAA training conducted (annual)
- ☐ Breach notification process tested

Audit Schedule:

- **Internal audit:** Quarterly (Dženan + Tarik)
- **External audit (SOC 2 Type II):** Annually (Month 6)

Payment App (Bosnia - Future)

Regulations:

- PSD2 (Strong Customer Authentication, open banking)
- PCI-DSS (card data security)
- BiH Banking Agency (payment institution license)
- AML/KYC (anti-money laundering, know your customer)
- GDPR (data protection)

Compliance Owner: Dženan Rizvanović

Timeline: Begin regulatory research Month 3, full compliance before GA launch.

6.2 Compliance Escalation

If compliance issue identified:

```
COMPLIANCE ISSUE detected
   ↓
DŽENAN investigates (within 1 hour)
   ├─ Assess severity (P1 = breach, P2 = risk, P3 = gap)
   ├─ Document findings
   └─ Escalate to John
   ↓
JOHN escalates to ALEM (within 1 hour for P1, 24h for P2/P3)
   ↓
ALEM decides response
   ├─ Notify customers (if breach)
   ├─ Notify regulators (if required)
```

```
    ├─ Engage legal counsel
    └─ Implement remediation
```

P1 Compliance Incidents:

- HIPAA breach (PHI exposed)
- PCI-DSS violation (card data exposed)
- Regulatory audit failure
- BAA violation by vendor

Response SLA: 1 hour to escalate, 24 hours to begin remediation

---

7. Intellectual Property Governance

7.1 IP Ownership Policy

Policy: All IP created by SnowIT for LumisCare is owned by Fast Constructions (USA).

Mechanism: Development Services Agreement (work-for-hire clause)

IP Assets:

- Source code
- Database schemas
- Design assets (UI/UX)
- Documentation
- Brand/trademarks
- Patents (future)

Assignment: All agents sign IP assignment clause in agreements.

7.2 Patent Strategy

Current Status:

- Provisional patent filing target: Within 60 days (by ~March 28, 2026)
- Innovation: Real-time AI clinical participation + video + home health forms (Vapi voice-to-assessment)

Governance:

- **Owner:** Fast Constructions (USA) — recommended

- **Decision:** Alem approves filing
- **Execution:** Patent attorney (Fish & Richardson or Finnegan)
- **Budget:** €3K-6K provisional, €50K-110K full utility over 3 years

Process:

1. Lejla documents technical innovation (2 weeks)
2. Dženan identifies patent attorney (2 weeks)
3. Attorney drafts application (4 weeks)
4. Alem reviews and approves (1 week)
5. File provisional (before 60-day deadline)

7.3 Trademark & Brand Governance

Current Trademarks:

- LumisCare (unregistered, use-based rights)

Action Required:

- ☐ File US trademark for "LumisCare" (word mark + logo)
- ☐ File trademark in EU/BiH (if expanding internationally)

Owner: Fast Constructions (USA) **Timeline:** File within 6 months (Month 6) **Budget:** €1,000-2,000 (US trademark)

---

8. Data Governance

8.1 Data Classification

Data Type	Sensitivity	Examples	Protection
PHI (Protected Health Information)	Critical	Patient names, diagnoses, visit notes	AES-256, TLS 1.3, RBAC, audit logs
PII (Personally Identifiable Information)	High	Email, phone, address	AES-256, TLS 1.3
Financial	High	Credit cards, bank accounts, transaction history	PCI-DSS, tokenization
Business Confidential	Medium	Revenue, customer list, roadmap	Access control, NDA

Data Type	Sensitivity	Examples	Protection
Public	Low	Marketing content, blog posts	No special protection

8.2 Data Retention Policy

Data Type	Retention Period	Rationale	Disposal Method
PHI (patient records)	6 years minimum (HIPAA)	Legal requirement	Secure deletion + audit log
Financial records	7 years (IRS requirement)	Tax compliance	Secure deletion
Audit logs	6 years (HIPAA)	Compliance	Secure deletion
Customer account data	90 days after cancellation	Business continuity	Secure deletion
Backups	30-90 days	Disaster recovery	Encrypted, auto-delete
Marketing data (non-PHI)	Indefinitely (or until opt-out)	Business use	Delete on request (GDPR)

8.3 Data Breach Response Plan

If PHI or PII breach detected:

```
1. DETECT breach (monitoring, report, audit)
   ↓
2. CONTAIN (within 1 hour)
   ├─ Nermin: Shut down affected system (if needed)
   ├─ Lejla: Identify scope of breach
   └─ Dženan: Begin documentation
   ↓
3. ASSESS impact (within 4 hours)
   ├─ How many individuals affected?
   ├─ What data was exposed?
   ├─ Was data encrypted?
   └─ Is this a HIPAA "breach" (legal definition)?
   ↓
4. NOTIFY (within legal deadlines)
   ├─ Customers affected (without undue delay, max 60 days)
   ├─ HHS (if >500 individuals, within 60 days)
   ├─ Media (if >500 individuals)
   └─ Business associates (if their data)
   ↓
```

```
5. REMEDIATE
   ├─ Fix vulnerability
   ├─ Enhance controls
   └─ Post-mortem
   ↓
6. DOCUMENT everything (legal defense)
```

Dženan owns breach response plan. Documented in `~/clawd/org/breach-response.md`.

---

9. Performance & Accountability

9.1 KPI Governance

Monthly Business Review (MBR) — Last Friday of Month

Attendees: Alem, John, Amina

Agenda:

1. Revenue & growth (MRR, customers, churn)
2. Product & development (features, velocity, tech debt)
3. Operations (uptime, incidents, support)
4. Trading (P&L, ROI)
5. Risks & compliance
6. Next month priorities

Dashboard: John maintains live dashboard (Notion, Grafana, or spreadsheet)

KPI Targets:

KPI	Target	Owner	Frequency
MRR (Monthly Recurring Revenue)	10%+ MoM growth	Selma	Monthly
Customer count	10 by Month 6, 50 by Month 12	Selma	Monthly
Churn rate	< 5% monthly	Selma + Amina	Monthly
Uptime	99.9% (LumisCare), 99.99% (Payment App)	Nermin	Daily
Deployment frequency	Daily (staging), weekly (prod)	Nermin	Weekly

KPI	Target	Owner	Frequency
Sprint velocity	Consistent ±10%	Emir	Per sprint
Bug escape rate	< 5%	Tarik	Per sprint
Test coverage	≥ 80%	Tarik + Lejla	Per release
Support response time	< 30 min (Tier 1)	Selma	Daily
Trading ROI	5%+ monthly	Nick	Monthly
Charity donations	50% of net profit	Alem	Annually

9.2 Accountability Mechanisms

How we ensure accountability:

- **RACI matrix** — every task has one owner (R = Responsible)
- **Daily standups** — public commitment to daily goals
- **Sprint reviews** — demo what shipped
- **Retros** — continuous improvement
- **Monthly KPI review** — Alem holds John accountable, John holds team accountable
- **Database logging** — all decisions logged, audit trail
- **GitHub** — all code changes tracked
- **Post-mortems** — blameless analysis of failures

Consequences for missed commitments:

- **First time:** Discussion, root cause analysis, improvement plan
- **Repeat pattern:** Re-assign task, escalate to Alem
- **Systemic issue:** Process improvement, not blame

Rewards for excellence:

- Public recognition (team meetings, retros)
- Increased responsibility (ownership of bigger projects)
- Future: Bonuses tied to KPIs (when profitable)

---

10. Change Management

10.1 How to Change This Document (GOVERNANCE.md)

Process:

1. Anyone can propose change (via John)
2. John reviews and assesses impact
3. If minor (typo, clarification): John updates, logs change
4. If major (authority change, new policy): John prepares proposal → Alem approves
5. Version control: All changes logged in document control table

Review Cadence: Quarterly (every 3 months)

10.2 Governance Evolution

As organization grows:

Stage	Headcount	Governance Changes
Startup (now)	1 owner + 1 AI + 10 agents	Alem = CEO, John = Director, informal governance
Growth (10-50 users)	+1-2 real humans	Formalize employment contracts, add HR policies
Scale (50-500 users)	+5-10 real humans	Create formal board, add CFO, legal counsel
Enterprise (500+ users)	+20+ real humans	Full C-suite, board of directors, formal governance

Current stage: Startup. Keep governance lean, bias toward action.

---

11. Document Control

Version	Date	Changes	Author
1.0	2026-01-28	Initial document	John + Dženan

Next Review: 2026-04-01 (quarterly)

Owner: Alem Basic **Maintained By:** John (Director) + Dženan Rizvanović (Risk & Compliance)

End of Governance Document

Clear authority. Clear escalation. Clear accountability. Alem controls, John executes, team delivers.

Process Manual

Processes & Workflows

Version: 1.0 **Last Updated:** 2026-01-28 **Owner:** Alem Basic **Prepared by:** John (Director) + Emir Delić (Scrum Master) + Amina Hadžić (Head of Projects)

Executive Summary

This document defines how work flows through the organization. It covers task lifecycle, approval workflows, communication protocols, meeting cadences, and operational processes. Every process is designed for speed, clarity, and accountability.

Key Principles:

- **Bias toward action** — ship fast, iterate based on feedback
- **Document decisions immediately** — no mental notes
- **Clear ownership** — every task has one owner (RACI: R = Responsible)
- **Escalate early** — blockers escalated within 1 hour
- **Continuous improvement** — retros every sprint, improve processes

---

1. Task Lifecycle — From Idea to Completion

1.1 Task States

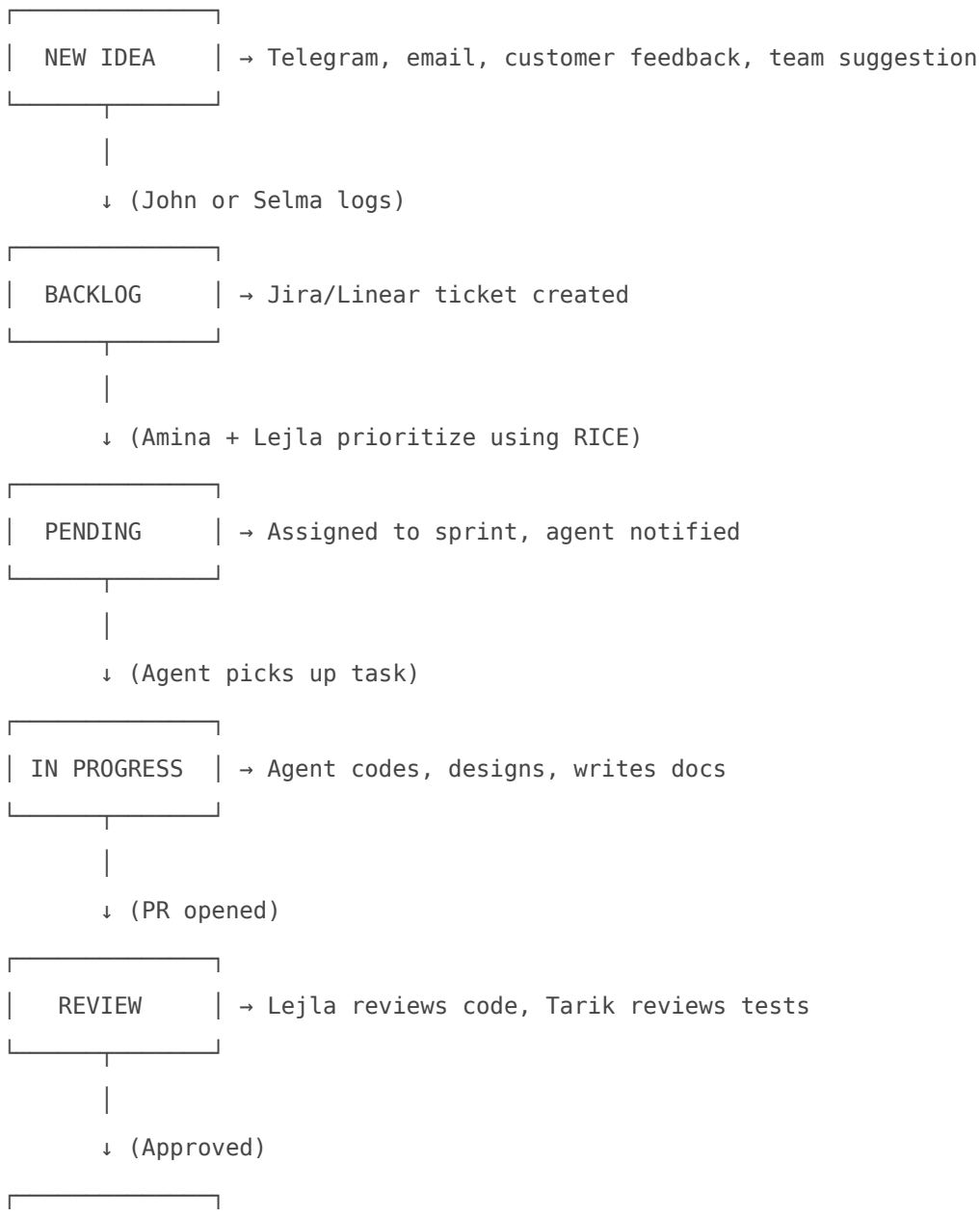
```
BACKLOG → PENDING → IN PROGRESS → REVIEW → TESTING → COMPLETED
   ↓         ↓           ↓            ↓         ↓          ↓
 Idea    Prioritized  Executing   Code review  QA    Deployed
```

Detailed States:

State	Definition	Owner	Exit Criteria

Backlog	Ideas, feature requests, bugs not yet prioritized	Amina + Lejla	Prioritized in sprint planning
Pending	Approved for sprint, waiting for pickup	Assigned agent	Agent starts work
In Progress	Actively being worked on	Agent	Work complete, PR opened
Review	Code review, design review, or approval	Lejla (code) or Amina (business)	Approved or changes requested
Testing	QA validation	Tarik	Tests pass, QA sign-off
Completed	Deployed to production, verified working	Nermin (deploy)	Live in production, no issues

1.2 Task Flow Diagram



```
│   TESTING    │ → Tarik runs E2E tests, checks quality gates
└──────┬───────┘
       │
       ↓ (Tests pass)
┌──────────────┐
│  DEPLOYMENT  │ → Nermin deploys to staging → production
└──────┬───────┘
       │
       ↓ (Verified working)
┌──────────────┐
│  COMPLETED   │ → Task archived, logged to DB
└──────────────┘
```

1.3 Task Metadata (Every Task Must Have)

Every task in Jira/Linear includes:

- **Title:** Short, descriptive (< 70 chars)
- **Description:** What needs to be done, why, acceptance criteria
- **Owner:** One person responsible (RACI: R)
- **Priority:** P1 (critical), P2 (high), P3 (medium), P4 (low)
- **Estimate:** Story points or hours
- **Labels:** feature, bug, tech-debt, compliance, etc.
- **Sprint:** Which sprint (if assigned)
- **Dependencies:** Blocks/blocked by other tasks
- **Acceptance criteria:** Checklist of what "done" means

Example Task:

```
Title: Add RBAC to Patient Management Module

Description:
Implement role-based access control (RBAC) for patient management.
Roles: Admin (full access), Caregiver (view own patients), Billing (view billing only).

Why: HIPAA compliance — minimum necessary access.

Acceptance Criteria:
- [ ] Admin can view/edit all patients
- [ ] Caregiver can view only assigned patients
- [ ] Billing can view patient billing info only (no PHI)
```

```
- [ ] Audit log records all access attempts
- [ ] Tests: Unit tests for RBAC logic, E2E test for each role
- [ ] Documentation: Updated API docs, user guide

Owner: API Developer
Reviewers: Lejla (code), Dženan (compliance), Tarik (testing)
Priority: P2 (high)
Estimate: 13 story points (1 week)
Sprint: Sprint 8
Dependencies: Blocks "Phase 3 GA launch"
Labels: feature, RBAC, HIPAA, Phase-3
```

---

2. Sprint Process (Agile/Scrum)

2.1 Sprint Cadence

Sprint Length: 2 weeks (10 business days)

Sprint Schedule:

Week	Day	Event
Week 1	Monday	Sprint Planning (new sprint starts)
	Wed	Backlog Refinement
	Thu	Architecture Review (bi-weekly)
Week 2	Monday	Mid-sprint check-in (Emir + Amina)
	Wed	Backlog Refinement
	Fri	Sprint Review + Retro (sprint ends)

Daily: Standup at 9:15 AM CET (Mon-Fri)

2.2 Sprint Planning (Every 2 Weeks, Monday, 2-3 hours)

Attendees: Amina, Emir, Lejla, Tarik, Nermin, Selma, Dženan, API Dev, Frontend

Agenda:

1. **Review last sprint** (5 min)
   - What shipped?
   - What didn't ship? Why?
   - Velocity: actual vs planned
2. **Present sprint goal** (10 min)
   - Amina: "This sprint we will..."
   - Example: "Complete Phase 3 RBAC and deploy to beta"
3. **Review backlog** (30 min)
   - Selma: Customer feedback, feature requests
   - Lejla: Tech debt priorities
   - Dženan: Compliance requirements
   - Amina + Lejla: RICE-prioritized backlog
4. **Estimate tasks** (60 min)
   - Team reviews each task
   - Estimate story points (Fibonacci: 1, 2, 3, 5, 8, 13, 21)
   - Identify dependencies and risks
5. **Commit to sprint** (15 min)
   - Team commits to sprint backlog
   - Emir: "We commit to X story points this sprint"
   - Amina approves
6. **Assign tasks** (10 min)
   - Each agent picks tasks
   - Balanced workload

Output:

- Sprint backlog (committed tasks)
- Sprint goal (one sentence)
- Velocity target (story points)

2.3 Daily Standup (Mon-Fri, 9:15 AM CET, 15 min max)

Attendees: All team (Emir leads)

Format: Each person answers 3 questions (1 min each):

1. **What did I do yesterday?**
2. **What will I do today?**
3. **Any blockers?**

Rules:

- Start on time (9:15 sharp)
- Max 15 minutes (Emir enforces)

- No problem-solving (take offline)
- Blockers escalated immediately after standup

Emir's Checklist:

- ☐ Update sprint board before standup
- ☐ Note blockers → escalate to Amina or Lejla after
- ☐ Update burn-down chart

2.4 Backlog Refinement (Weekly, Wednesday, 1 hour)

Attendees: Emir, Lejla, Selma

Agenda:

1. Review new tasks (from customers, team, bugs)
2. Write clear descriptions and acceptance criteria
3. Estimate rough size (T-shirt: S, M, L, XL)
4. RICE score (prioritization)
5. Tag with labels (feature, bug, tech-debt, etc.)

Output:

- Refined backlog (ready for sprint planning)
- Top 20 tasks RICE-scored

2.5 Sprint Review (End of Sprint, Friday, 1 hour)

Attendees: Amina, Emir, Selma, Lejla, + stakeholders (Alem, customers)

Agenda:

1. **Demo completed work** (30 min)
   - Selma or agent demos features to stakeholders
   - Live demo, not slides
   - "Here's what we shipped this sprint"
2. **Review metrics** (15 min)
   - Velocity: committed vs completed
   - Quality: bugs found, test coverage
   - Customer feedback
3. **Gather feedback** (15 min)
   - Stakeholders provide input

- New ideas added to backlog

Output:

- Stakeholder feedback
- New backlog items
- Celebration of wins

2.6 Sprint Retrospective (End of Sprint, Friday, 45 min)

Attendees: All team (Emir leads)

Format: Start/Stop/Continue

Agenda:

1. **What should we START doing?** (15 min)
   - New practices, tools, processes
   - Example: "Start writing ADRs for architecture decisions"
2. **What should we STOP doing?** (15 min)
   - Bad habits, wasteful processes
   - Example: "Stop scheduling meetings during focus time"
3. **What should we CONTINUE doing?** (15 min)
   - What's working well
   - Example: "Continue daily standups at 9:15 AM"
4. **Action items** (5 min)
   - Pick 1-3 improvements to implement next sprint
   - Assign owner for each

Rules:

- Blame-free zone
- Focus on process, not people
- Implement at least 1 action item per sprint

Emir's Job:

- Facilitate discussion
- Keep it positive and constructive
- Document action items
- Follow up on previous retro actions

---

3. Approval Workflows

3.1 Code Review Process

Trigger: Developer opens Pull Request (PR) on GitHub

Process:

```
1. DEVELOPER opens PR
   ↓
2. Automated checks run (CI/CD)
   ├─ Tests (unit, integration)
   ├─ Linting (ESLint, Prettier)
   ├─ Security scan (OWASP ZAP)
   └─ Build succeeds
   ↓ (if all pass)
3. LEJLA reviews code
   ├─ Architecture alignment
   ├─ Code quality
   ├─ Performance
   └─ Security
   ↓ (if approved)
4. TARIK reviews tests
   ├─ Test coverage ≥ 80%
   ├─ E2E test for happy path
   └─ Quality gates pass
   ↓ (if approved)
5. MERGE to main branch
   ↓
6. Auto-deploy to STAGING
   ↓
7. QA verification in staging
   ↓
8. Manual promote to PRODUCTION (Nermin)
```

PR Approval Criteria (Definition of Done):

- ☐ All automated tests pass
- ☐ Test coverage ≥ 80%

- [ ] E2E test for critical path
- [ ] Code reviewed by Lejla (or 2 senior devs)
- [ ] No security vulnerabilities (OWASP scan)
- [ ] Accessibility check (WCAG 2.1 AA)
- [ ] Performance benchmark pass (API < 500ms, page load < 2s)
- [ ] Documentation updated (if API or UI change)

SLA:

- Code review within 24 hours (Lejla)
- Revisions addressed within 24 hours (Developer)
- Total PR lifecycle: < 72 hours (3 days)

3.2 Feature Approval Process

For new features (not in roadmap):

```
1. IDEA submitted (Selma, customer, team member)
   ↓
2. SELMA writes user story + business case
   ↓
3. LEJLA estimates technical effort
   ↓
4. AMINA RICE-scores feature
   ↓ (if high RICE score)
5. JOHN prepares options (build, buy, defer)
   ↓
6. ALEM decides (approve, defer, reject)
   ↓ (if approved)
7. Add to backlog → sprint planning
```

Timeline:

- Idea → Decision: 1 week max

3.3 Deployment Approval

Staging Deployment:

- **Trigger:** PR merged to main
- **Approval:** Automated (no approval needed)

- **Rollback:** Automatic if health check fails

Production Deployment:

- **Trigger:** Manual (Nermin triggers after QA sign-off)
- **Approval:** Tarik (QA sign-off) + Nermin (deploy)
- **Rollback:** Manual (Nermin) if errors detected

Production Deployment Checklist:

- ☐ All staging tests pass
- ☐ QA sign-off from Tarik
- ☐ No P1/P2 bugs in staging
- ☐ Runbook updated (if new feature)
- ☐ Monitoring alerts configured
- ☐ Rollback plan documented
- ☐ Deploy during low-traffic window (if high-risk)

Deployment Windows:

- **Low-risk:** Anytime
- **High-risk:** Tuesday-Thursday, 10 AM - 2 PM CET (avoid Fridays, weekends, holidays)

3.4 Budget Approval

Amount	Approver	Process
< €500	John	Immediate, logged to DB
€500 - €5,000	John	Immediate, logged, Alem notified
€5,000 - €50,000	Alem	John prepares options, Alem decides
> €50,000	Alem	Formal proposal, board approval (if applicable)

Example:

- **€200/month SaaS tool** (Intercom) → John approves, logs decision
- **€3,000 patent filing** → John approves, notifies Alem
- **€10,000 Google Startup credits** → John prepares application, Alem approves
- **€100,000 Series A funding** → Alem decides

3.5 Compliance Sign-Off (HIPAA, PCI-DSS)

For any feature handling PHI (Protected Health Information):

```
1. DEVELOPER builds feature
   ↓
2. TARIK tests compliance controls
   ├─ Encryption at rest/transit
   ├─ Access control (RBAC)
   ├─ Audit logging
   └─ Data retention
   ↓ (tests pass)
3. DŽENAN reviews compliance checklist
   ├─ HIPAA Privacy Rule
   ├─ HIPAA Security Rule
   ├─ Vendor BAAs (if applicable)
   └─ Breach notification process
   ↓ (approved)
4. DEPLOY to production
```

Dženan's Compliance Checklist:

- ☐ PHI encrypted at rest (AES-256)
- ☐ PHI encrypted in transit (TLS 1.3)
- ☐ Access control enforced (RBAC, MFA)
- ☐ Audit log captures all PHI access
- ☐ Vendor BAAs signed (if third-party involved)
- ☐ Privacy policy updated (if needed)
- ☐ User consent obtained (if needed)

Compliance Sign-Off SLA: 48 hours (Dženan reviews within 2 business days)

---

4. Communication Protocols

4.1 Communication Channels & Usage

Channel	Use Case	Response SLA	Audience
Telegram (@johnbasicas_bot)	Urgent matters, quick decisions, P1 incidents	5-15 min	Alem ↔ John

Channel	Use Case	Response SLA	Audience
CLI (Claude Code)	Deep work, architecture, coding, planning	Real-time (during session)	John ↔ agents
Email (john@alai.no)	External communication, formal records, client communication	4-24 hours	External parties
Slack (future)	Team collaboration, quick questions	1-4 hours	Internal team
Jira/Linear	Task tracking, sprint management	Daily check	Team
GitHub	Code, PRs, technical discussion	24 hours	Developers
Database (john.db)	Source of truth, all decisions logged	N/A (logged immediately)	John, Alem (query)
Standups	Daily status, blockers	9:15 AM CET daily	All team
Meetings	Strategic discussions, planning	Scheduled	Per invite

4.2 When to Use Which Channel

Situation	Channel	Why
Production is down (P1)	Telegram → Nermin, Lejla, John, Alem	Immediate response needed
Strategic decision needed	Telegram (Alem ↔ John)	Fast, informal
Task assignment	Jira/Linear + CLI	Trackable, logged
Code review	GitHub PR comments	Context, threaded discussion
Customer inquiry	Email (Selma)	Professional, recorded
Quick question for teammate	Slack (or CLI)	Fast, informal
Architecture proposal	Written doc (Lejla) + meeting	Needs deep thought
Bug report	Jira/Linear	Needs tracking, prioritization
Daily status	Standup (9:15 AM)	Synchronous, team awareness
Document important decision	Database (john.db)	Source of truth

4.3 Response Time Expectations

Priority	Channel	Response SLA	Example
P1 — Critical	Telegram, phone	5-15 min	Production down, security breach

Priority	Channel	Response SLA	Example
P2 — High	Telegram, Slack	1 hour	Major bug, customer escalation
P3 — Medium	Slack, email	4 hours (business)	Feature request, minor bug
P4 — Low	Email, Jira	24 hours	Enhancement, question

Business Hours: 9 AM - 6 PM CET (Mon-Fri)

After-Hours: P1 only (Nermin on-call)

4.4 Meeting Etiquette

Rules:

- **Start on time** — don't wait for latecomers
- **End on time** — respect people's calendars
- **Agenda required** — no agenda = no meeting
- **One speaker at a time** — no interruptions
- **Action items documented** — every meeting ends with action items, owners, deadlines
- **No phones/distractions** — focus on meeting
- **Optional attendees clearly marked** — required vs optional

Meeting Types:

Type	Agenda Required	Notes Required	Max Duration
Standup	No (standard format)	No	15 min
Sprint planning	Yes	Yes	3 hours
Sprint review	Yes (demo list)	Yes	1 hour
Retro	No (standard format)	Yes (action items)	45 min
Architecture review	Yes (proposals)	Yes (ADRs)	2 hours
1:1	Optional	Optional	30 min
Ad-hoc problem-solving	No	Yes (decisions logged)	30 min

---

5. Incident Response Process

5.1 Incident Severity Levels

Priority	Definition	Examples	Response SLA	Escalation
P1 — Critical	Service down, data breach, multiple users affected	Production down, PHI exposed, database corruption	15 min	Immediate → Alem
P2 — High	Major feature broken, workaround exists, single user affected	Scheduling not working, payment failed	1 hour	If not resolved in 2h → Alem
P3 — Medium	Minor feature issue, cosmetic, no user impact	Report formatting wrong, UI glitch	4 hours	If not resolved in 24h → Amina
P4 — Low	Enhancement request, question, documentation	Feature request, how-to question	24 hours	No escalation

5.2 P1 Incident Response Flow

P1 Definition: Production down, security breach, data loss, HIPAA breach.

```
1. DETECTION (monitoring, customer report, team)
   ↓
2. NERMIN (DevOps) notified via PagerDuty
   ↓ (within 15 minutes)
3. NERMIN triages and starts investigation
   ↓ Simultaneously:
   ├─ Notify LEJLA (tech lead)
   ├─ Notify DŽENAN (if security/compliance)
   ├─ Notify JOHN (coordination)
   └─ Notify AMINA (stakeholder communication)
   ↓
4. INCIDENT CHANNEL opened (Slack or Telegram)
   ↓
5. INVESTIGATION (Nermin + Lejla)
   ├─ Identify root cause
   ├─ Assess impact (# users, data affected)
   └─ Determine fix or rollback
   ↓
6. DECISION (within 1 hour)
   ├─ Rollback to previous version (if safe)
   ├─ Apply hotfix (if fast and safe)
   └─ Escalate to ALEM (if major decision needed)
   ↓
```

```
7. IMPLEMENT FIX
   ↓
8. VERIFY fix working
   ↓ (if data breach)
9. BREACH NOTIFICATION process
   ├─ Dženan leads
   ├─ Notify affected customers (within 60 days per HIPAA)
   ├─ Notify HHS (if >500 individuals)
   └─ Document everything
   ↓
10. POST-MORTEM (within 48 hours)
    ├─ Root cause analysis
    ├─ Timeline of events
    ├─ What went wrong
    ├─ What went right
    └─ Action items to prevent recurrence
```

P1 Communication:

- **Internal:** Incident channel (Slack/Telegram), all updates logged
- **External (customers):** Selma drafts status page update (if customer-facing)
- **External (regulators):** Dženan coordinates (if breach)

P1 Response Targets:

- **Acknowledge:** 15 min
- **Triage:** 30 min
- **Fix or rollback:** 4 hours
- **Post-mortem:** 48 hours

5.3 Post-Mortem Template

File: `~/clawd/org/incidents/YYYY-MM-DD-incident-name.md`

```
# Post-Mortem: [Incident Name]

**Date:** YYYY-MM-DD
**Severity:** P1/P2/P3
**Duration:** X hours (HH:MM start - HH:MM resolved)
**Affected Users:** X users
**Incident Lead:** [Name]
```

```
## Summary
[2-3 sentence summary of what happened]

## Timeline
- HH:MM — Incident detected
- HH:MM — Nermin notified
- HH:MM — Root cause identified
- HH:MM — Fix deployed
- HH:MM — Incident resolved

## Root Cause
[Technical explanation of what caused the issue]

## Impact
- Users affected: X
- Data lost: Yes/No
- Revenue impact: €X
- Downtime: X hours

## What Went Wrong
1. [Issue 1]
2. [Issue 2]

## What Went Right
1. [Success 1]
2. [Success 2]

## Action Items
- [ ] [Action 1] — Owner: [Name], Deadline: [Date]
- [ ] [Action 2] — Owner: [Name], Deadline: [Date]

## Lessons Learned
[Key takeaways to prevent recurrence]
```

---

6. Customer Interaction Processes

6.1 Customer Onboarding Flow

Goal: Get customer to value in first 5 minutes.

```
1. CUSTOMER signs up (email + agency name)
   ↓
2. Welcome email (automated)
   ↓
3. In-app guided setup wizard
   ├─ Add first caregiver
   ├─ Add first patient
   ├─ Schedule one visit
   └─ Try Vapi voice demo
   ↓ (Day 1, 3, 7)
4. SELMA check-in emails
   ├─ "How's it going?"
   ├─ "Need help?"
   └─ "Ready to invite your team?"
   ↓ (Day 14)
5. Trial ends → convert to paid OR
   ↓
6. SELMA follow-up call
   ├─ Address concerns
   ├─ Offer discount/extension
   └─ Ask for feedback
```

Onboarding Metrics:

- Time to first value (target: < 5 min)
- % users who complete setup wizard (target: 70%)
- Trial-to-paid conversion (target: 30%)

6.2 Customer Support Ticketing

Tool: Intercom or Linear (TBD)

Tiers:

Tier	Handler	Types of Issues	SLA
Tier 1	Selma	How-to, account, billing	30 min
Tier 2	Tarik + Devs	Bug investigation, technical	4 hours
Tier 3	Lejla + Nermin	Architecture, infrastructure, P1	1 hour

Tier	Handler	Types of Issues	SLA
Tier 4	Amina + Dženan	Executive escalation, compliance breach	Immediate

Flow:

```
1. CUSTOMER submits ticket (in-app chat, email)
   ↓
2. SELMA triages (Tier 1)
   ├─ Can answer immediately? → Resolve
   └─ Technical or bug? → Escalate to Tier 2
   ↓
3. TARIK investigates (Tier 2)
   ├─ Can reproduce bug? → Create Jira ticket, prioritize
   ├─ Infrastructure issue? → Escalate to Nermin (Tier 3)
   └─ Compliance issue? → Escalate to Dženan (Tier 4)
   ↓
4. RESOLUTION
   ├─ Fix deployed → Notify customer
   └─ Cannot fix → Explain why, offer workaround
   ↓
5. FOLLOW-UP (Selma)
   ├─ "Is this resolved?"
   └─ "Anything else we can help with?"
```

Support Metrics:

- First response time (target: < 30 min for Tier 1)
- Resolution time (target: < 24 hours for P3/P4)
- Customer satisfaction (CSAT, target: ≥ 4.5/5)
- Self-service rate (target: 70% resolve via knowledge base)

6.3 Customer Churn Prevention

Trigger: Customer cancels subscription or shows churn signals.

Churn Signals:

- No logins in 7+ days
- Low usage (< 10% of expected activity)
- Support tickets indicating frustration
- Cancellation request

Process:

```
1. CHURN SIGNAL detected (automated alert)
   ↓
2. SELMA reaches out
   ├─ "We noticed you haven't logged in. Everything okay?"
   ├─ Offer help, training, demo
   └─ Ask for feedback
   ↓ (if still churning)
3. AMINA escalation
   ├─ Personal call from Amina
   ├─ "What can we do to make this work?"
   ├─ Offer discount, extension, custom onboarding
   └─ Exit interview (if they still leave)
   ↓
4. LOG FEEDBACK
   ├─ Why did they churn?
   ├─ What could we improve?
   └─ Add to product backlog
```

Churn Metrics:

- Monthly churn rate (target: < 5%)
- Reasons for churn (categorized)
- Win-back rate (% churned customers who return)

---

7. Financial Processes

7.1 Invoicing & Revenue Collection

LumisCare (SaaS):

```
1. CUSTOMER subscribes (Stripe)
   ↓
2. Stripe charges card automatically (monthly)
   ↓
3. Invoice emailed to customer (Stripe auto-send)
   ↓ (if payment fails)
4. Stripe retries (3 attempts over 2 weeks)
```

```
   ↓ (if still fails)
5. SELMA notified → contact customer
   ├─ Update payment method
   └─ If no response: suspend account (Day 30)
   ↓ (if suspended)
6. Account locked (read-only, 30-day grace)
   ↓ (if no payment after 30 days)
7. Account deleted (data retained 90 days per HIPAA)
```

Payment Flow (Fast Constructions ↔ SnowIT):

```
1. END OF MONTH: Fast Constructions calculates revenue
   ↓
2. JOHN calculates development fee (% of revenue or fixed)
   ↓
3. Fast Constructions wires payment to SnowIT (monthly)
   ↓
4. SnowIT pays team members (monthly)
   ↓
5. Both entities file taxes (quarterly or annual)
```

7.2 Expense Approval

Process:

```
1. AGENT needs to purchase tool/service
   ↓
2. Check budget approval matrix:
   ├─ < €500: John approves immediately
   ├─ €500-€5K: John approves, logs to DB, notifies Alem
   └─ > €5K: John prepares options → Alem approves
   ↓
3. Purchase made (corporate card or wire)
   ↓
4. Receipt logged to accounting system
   ↓
5. Monthly expense report (John → Alem)
```

Expense Categories:

- Infrastructure (AWS, hosting)
- SaaS tools (Stripe, Intercom, etc.)
- Marketing (ads, outreach tools)
- Professional services (legal, accounting)
- Team compensation
- Other

7.3 Charitable Giving (50% Commitment)

Process:

```
1. END OF QUARTER: Fast Constructions calculates net profit
   ↓
2. JOHN calculates 50% of net profit → charity allocation
   ↓
3. ALEM selects charities (or delegates to John)
   ↓
4. Donations made (wire transfer, check)
   ↓
5. Receipts filed (tax deduction)
   ↓
6. PUBLIC REPORT (transparency)
   ├─ "This quarter we donated €X to [charities]"
   └─ Posted on lumiscare.com/impact
```

Charity Selection Criteria:

- Healthcare access (aligned with LumisCare mission)
- Underserved communities
- Verified 501(c)(3) or equivalent
- Transparent financials (GuideStar, Charity Navigator)

---

8. Documentation Processes

8.1 Technical Documentation

Types:

- **ADRs (Architecture Decision Records):** Why we chose X over Y
- **API docs:** OpenAPI/Swagger, auto-generated

- **Runbooks:** How to respond to incidents
- **User guides:** Help center, knowledge base
- **Code comments:** Inline documentation for complex logic

Process:

```
DEVELOPER makes architecture decision
   ↓
LEJLA writes ADR (Architecture Decision Record)
   ├─ Context: What problem are we solving?
   ├─ Options: What options did we consider?
   ├─ Decision: What did we choose?
   └─ Consequences: What are the trade-offs?
   ↓
ADR committed to repo (docs/adr/)
   ↓
Referenced in code and future discussions
```

ADR Template:

```
# ADR-XXX: [Title]

**Date:** YYYY-MM-DD
**Status:** Accepted / Rejected / Superseded
**Deciders:** [Names]

## Context
[What problem are we solving? Why now?]

## Options Considered
1. **Option A:** [Description]
   - Pros: [...]
   - Cons: [...]
2. **Option B:** [Description]
   - Pros: [...]
   - Cons: [...]

## Decision
We chose **Option A** because [rationale].
```

```
## Consequences
- Positive: [...]
- Negative: [...]
- Neutral: [...]

## References
- [Link to discussion]
- [Link to proposal]
```

8.2 User Documentation

Owner: Selma (content) + Emir (video tutorials)

Types:

- Quick Start Guide (PDF + in-app)
- Feature walkthroughs (video, 3-5 min each)
- FAQ (knowledge base)
- Troubleshooting guides
- API docs (for Enterprise customers)

Process:

```
NEW FEATURE shipped
   ↓
SELMA writes help doc
   ├─ What is this feature?
   ├─ How do I use it?
   ├─ Screenshots + step-by-step
   └─ FAQ
   ↓
EMIR records video tutorial (if complex)
   ↓
Published to help center (Notion, Intercom, etc.)
   ↓
Linked in-app (contextual help)
```

User Doc Review Cadence:

- Review all docs monthly (Selma)
- Update screenshots when UI changes
- Add new FAQs based on support tickets

9. Continuous Improvement

9.1 Retro Action Items Tracking

Process:

```
RETRO (every sprint)
   ↓
Team identifies 1-3 improvements
   ↓
Assign owner + deadline for each
   ↓
Owner implements during next sprint
   ↓
Next retro: Review progress
   ├─ Did we implement it?
   ├─ Did it help?
   └─ Keep, modify, or drop?
```

Example Retro Action Items:

Sprint	Action Item	Owner	Deadline	Status
Sprint 7	Start writing ADRs for architecture decisions	Lejla	Sprint 8	✅ Done
Sprint 7	Add automated security scan to CI/CD	Nermin	Sprint 8	✅ Done
Sprint 8	Reduce PR review time from 48h → 24h	Lejla	Sprint 9	🔄 In Progress

9.2 Process Review Cadence

Process	Review Frequency	Owner	Next Review
Sprint ceremonies	Monthly	Emir	Every retro
Code review process	Quarterly	Lejla	2026-04-01
Deployment process	Quarterly	Nermin	2026-04-01

Process	Review Frequency	Owner	Next Review
Support ticketing	Monthly	Selma	Every month
Financial processes	Annually	John	2026-12-01
Compliance processes	Quarterly	Dženan	2026-04-01

9.3 Metrics Review

Monthly Business Review (last Friday of month):

Attendees: Alem, John, Amina

Agenda:

1. **Revenue & growth** (10 min)
   - MRR, new customers, churn
2. **Product & development** (10 min)
   - Features shipped, velocity, tech debt
3. **Operations** (10 min)
   - Uptime, incidents, support metrics
4. **Trading** (5 min)
   - P&L, ROI, positions
5. **Risks & compliance** (10 min)
   - Open risks, compliance status
6. **Next month priorities** (15 min)
   - What are we focusing on?

Output: Updated priorities for next month

---

10. Document Control

Version	Date	Changes	Author
1.0	2026-01-28	Initial document	John + Emir + Amina

Next Review: 2026-04-01 (quarterly)

Owner: Alem Basic **Maintained By:** John (Director) + Emir Delić (Scrum Master) + Amina Hadžić (Head of Projects)

End of Processes & Workflows Document

Every process documented. Every workflow clear. Every escalation path defined. Ship fast, iterate, improve.

ZAKON-QA — Real-User Testing Standard (mandatory all agents) — 2026-06-05

ZAKON-QA: Real-User Testing Standard (MANDATORY for ALL agents)

Authority: CEO directive 2026-06-05 (Bilko demo bug campaign, MC #102887). Proven across 14+ bugs the CEO found in minutes that "54/54 green" API tests missed. This is now the required testing methodology for EVERY agent (Proveo/Angie Jones, Maria Santos, CodeCraft, FlowForge, any QA/build dispatch) on EVERY web product. Violating it = invalid QA.

The 7 hard rules

1. Test as a REAL USER, not the API

API-with-token tests give false green — they assert "feature exists in code", not "a user can do this". Drive the REAL UI with REAL clicks (Playwright/browser). Use reliable auth (API-login + session-cookie injection, the `real-demo-smoke.spec.ts` pattern) — flaky UI-form login silently lands on `/login` and poisons every check. Log in as the actual product user.

2. FULL CRUD round-trips, not "page renders"

For EVERY entity (invoices, expenses, contacts, articles, …): **create → it MUST appear in the list → open its detail → edit → reuse it.** "No error on click" ≠ "works". Verify the OUTCOME: the created item actually appears, the edited value actually persists, the deleted item actually disappears. A test that clicks a button and only checks "no exception" is NOT a test.

3. Verify by LIVE OUTCOME — never by "build SUCCESS" or git-sha label

A green build with the correct commit label CAN ship a STALE bundle (dead-code edit, Docker/BuildKit layer cache). NEVER claim a fix is live because the build passed or the Cloud Run revision is labeled with the new sha. Confirm on the LIVE url with a screenshot you ACTUALLY LOOK AT (e.g. "PDV shows €30,50, not €NaN"). **Assert the page RENDERED a known element BEFORE asserting the fix** — a blank page / login redirect falsely passes "no NaN on page".

4. REJECT your own false positives — and NEVER fabricate

When something looks like a bug, verify the real affordance/data before reporting: wrong test account (e.g. logged in as a different org with empty data), wrong selector, guessed-wrong URL, checking the wrong field/page, already-sorted data. If your control case also "fails", your TEST is broken, not the app. A regression test that PASSES by SKIPPING its assertion ("could not create draft — skipping") is NOT a pass. Never invent a bug to look busy; reporting a false bug is a worse failure than finding none.

5. CONSISTENCY / completeness audit

Systematically check that each entity has: list + detail view + create + edit + sort + REACHABLE navigation (row/name click → detail). Inconsistency between entities (invoices have detail, contacts don't) is exactly where real bugs hide. Audit the route tree + the actual nav affordances (Link href / onClick), not just "does the list render".

6. Don't rationalize broken UX as "intentional stub"

If a real user hits an error toast / 403 / 501 / dead button, it IS a bug — regardless of backend intent. Either make the feature work or degrade it GRACEFULLY (clear "not available" notice, hidden/disabled button + upsell). A 501 that makes the client `throw` an error toast is a bug even if storage is "intentionally" unconfigured.

7. Capture EVERYTHING + mobile

On every page: capture console errors, all HTTP ≥400, NaN/undefined/raw-i18n-key leaks, horizontal overflow at 390px mobile. Screenshot each state and LOOK at it.

Process when a bug is found

Reproduce (live, real clicks) → root-cause (edit the LIVE/compiled file, not dead code) → fix → CI green (required gates) → deploy → **re-verify the OUTCOME on the live surface with a screenshot** → only then "done". Per [[feedback_verify_by_live_outcome_not_green_build_2026-06-04]].

Reusable harness

`apps/e2e/tests/regression-102887.spec.ts` + `_hunt*.mjs` cookie-auth pattern in the Bilko repo are the reference implementation. The `webapp-testing`, `deploy-verify`, `mobile-uat`, and `uat-browser` skills MUST follow these 7 rules.