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

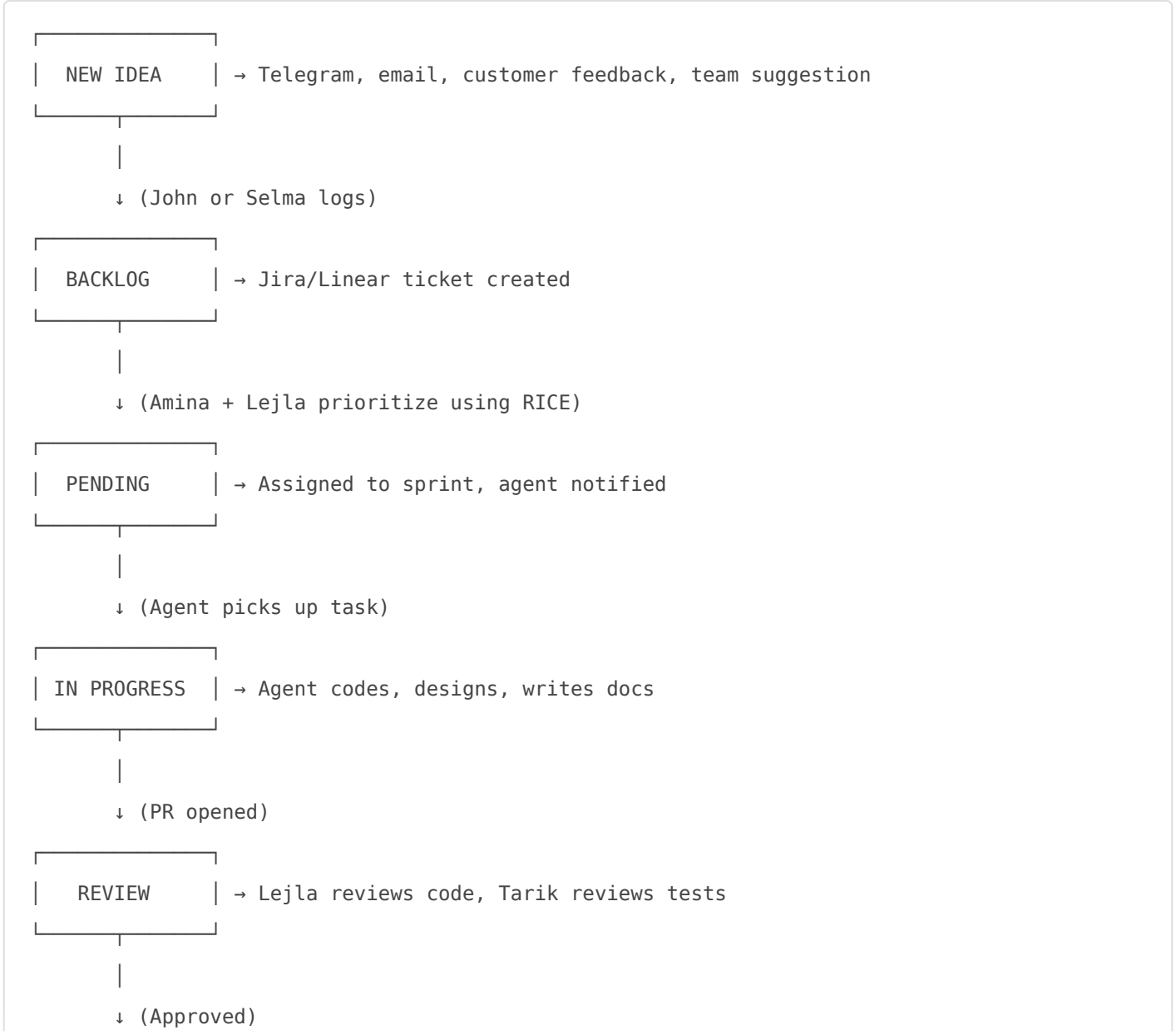
1.1 Task States

```
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.

Revision #7
Created 2026-02-20 21:33:18 UTC by John
Updated 2026-06-21 20:02:04 UTC by John