Release

Bilko release management — deployment checklists, release notes, rollback plans

- Deployment Checklist
- Release Notes
- Rollback Plan
- UAT Sign-Off

Deployment Checklist

Deployment Checklist

> **Project:** Bilko **Version:** 0.1 **Date:** 2026-02-23 **Author:** Ops Architect **Status:** Draft **Reviewers:** Tech Lead, Alem Bašić

Document History

Version	Date	Author	Changes
0.1	2026-02-23	Ops Architect	Initial draft

INSTRUCTIONS

Use this checklist for every production deployment. Create a copy for each release: `DEPLOY-CHECKLIST-YYYY-MM-DD-vX.X.X.md` in `docs/releases/`

Never skip items. If an item doesn't apply, mark N/A with reason.

Deployment Checklist: Bilko v[VERSION]

Deploy date: YYYY-MM-DD **Deploy time:** HH:MM CET **Release notes:** [Link to release notes]
Deployer: [Name] **Reviewer:** [Name]

Pre-Deployment (Run 24 Hours Before)

Code & Tests

- [ ] All CI checks green on main branch (lint, type-check, unit tests, integration tests, build, E2E)
- [ ] Coverage ≥ 80% overall, ≥ 95% financial logic
- [ ] All PRs in this release merged to main
- [ ] No `TODO` or `FIXME` comments referencing this release
- [ ] No skipped tests (`it.skip`, `test.skip`) in financial logic files
- [ ] Release notes prepared and reviewed

Database

- [ ] New migrations reviewed by Tech Lead
- [ ] Migrations tested on staging environment:
  - Applied successfully: Yes / No
  - Rollback (down migration) tested: Yes / No
  - Migration time on staging (record): ___ seconds
- [ ] Migration is backward-compatible (old API version can still read data): Yes / N/A
- [ ] If schema changes: Prisma Client regenerated and tested

Security

- [ ] No new secrets committed to git (`git log --all --grep='password\|secret\|key'` — should be clean)
- [ ] All new API endpoints require authentication (no accidental public endpoints)
- [ ] RBAC verified for new endpoints (correct roles enforced)
- [ ] Zod validation applied to all new request bodies
- [ ] No cross-org data leakage in new queries

Financial Logic (Skip if release has no financial changes)

- [ ] VAT calculations tested for all 3 countries (RS 20%, BA 17%, HR 25%)
- [ ] Double-entry validation enforced for all new transaction creation paths
- [ ] NUMERIC(19,4) used throughout — no JavaScript `number` for monetary values
- [ ] Exchange rate locking verified for multi-currency transactions

- [ ] Invoice total calculation verified: subtotal + VAT - discount = total (exact, no float)

Staging Verification (Run 2 Hours Before)

- [ ] Staging deployment successful (same build that will go to production)
- [ ] Staging health check passing: `curl https://staging-api.bilko.io/health`
- [ ] Manual smoke test on staging:
  - [ ] Login with test account
  - [ ] Create invoice (RSD, 20% VAT) — verify totals correct
  - [ ] Create expense with receipt upload
  - [ ] Generate VAT report
  - [ ] Verify report numbers are consistent with data
- [ ] E2E tests pass on staging
- [ ] No new Sentry errors after staging deploy (check Sentry for 30 min)
- [ ] Database migrations applied to staging: `bilko_staging` DB

Backup (Run 30 Minutes Before)

- [ ] Pre-deploy backup taken:

```
railway run pg_dump $DATABASE_URL -f pre_deploy_$(date +%Y%m%d_%H%M).dump
```

- [ ] Backup file accessible and non-empty
- [ ] Backup stored in secure location (note location here): _____

Deployment Execution

Step 1: Apply Database Migrations

- [ ] Migrations applied to production:

```
railway run npx prisma migrate deploy
```

- [ ] Migration successful: Yes / No
- [ ] Post-migration record count check:

```
railway run psql $DATABASE_URL -c "SELECT COUNT(*) FROM invoices, COUNT(*) FROM organizations;"
```

  - Invoice count: ___
  - Organization count: ___

Step 2: Deploy Backend (Railway)

- [ ] Push to main triggers Railway production deploy (automatic via CI) OR manual: `railway up --service api --environment production`
- [ ] Railway deployment successful
- [ ] Railway health check green: `curl https://api.bilko.io/health`
- [ ] No restart loops in Railway logs

Step 3: Deploy Frontend (Vercel)

- [ ] Vercel production deployment successful (automatic via CI) OR manual: `cd apps/web && vercel --prod`
- [ ] bilko.io loads correctly in browser
- [ ] No browser console errors on first load

Post-Deployment Verification (Run Immediately After Deploy)

- [ ] API health: `curl https://api.bilko.io/health` → `{"status":"ok","db":"ok"}`
- [ ] Frontend: `curl -I https://bilko.io` → HTTP/2 200
- [ ] Login flow: can log in with test account
- [ ] Invoice creation: create test invoice, verify totals
- [ ] BetterStack: all monitors green

- [ ] Sentry: no new error types in first 15 minutes
- [ ] Railway: CPU < 50%, Memory < 1GB

15-Minute Monitoring Window

Monitor for 15 minutes after deploy. Record observations:

Time	API Health	Error Rate	CPU	Memory	Notes
+0 min					
+5 min					
+10 min					
+15 min					

Deploy declared successful: [ ] Yes — at HH:MM by [Name]

Rollback Criteria

Rollback immediately if:

- Health check fails for > 3 consecutive minutes
- Error rate > 5% (Sentry)
- Any financial calculation producing incorrect results
- Authentication completely broken

See rollback procedure: rollback-plan.md

Post-Deploy Cleanup

- [ ] Rollback plan document created for this release (if major release)
- [ ] Release notes published
- [ ] Team notified in Slack #bilko-deploys: "v[VERSION] deployed successfully"
- [ ] Deploy notes added to GitHub release tag

Approval

Role	Name	Date	Signature
Deployer			
Reviewer	Alem Bašić		

Release Notes

Release Notes

> **Project:** Bilko **Version:** 0.1 **Date:** 2026-02-23 **Author:** Ops Architect **Status:** Draft (Template — fill in per release) **Reviewers:** Tech Lead, Alem Bašić

Document History

Version	Date	Author	Changes
0.1	2026-02-23	Ops Architect	Initial draft

INSTRUCTIONS

Create release notes for every production deploy. Audience: internal team and, for major releases, Bilko users.

File: `RELEASE-NOTES-vX.X.X.md` in `docs/releases/` Versioning: Semantic versioning — `MAJOR.MINOR.PATCH`

- MAJOR: breaking API changes or database schema changes requiring migration downtime
- MINOR: new features, non-breaking
- PATCH: bug fixes, performance improvements, security patches

---

Release Notes — Bilko v1.0.0 (MVP Launch)

Release Date: TBD (backend MVP complete) **Environment:** Production **Deploy window:** TBD

Summary

This is the first production release of Bilko — cloud accounting for Balkan SMBs. It includes the core accounting engine (invoicing, expenses, VAT reporting) with multi-country support for Serbia, Bosnia & Herzegovina, and Croatia.

New Features

Core Accounting Engine

Invoice Management

- Create invoices via 6-step wizard
- Support for RSD, EUR, BAM currencies
- Multi-country VAT: Serbia (20%), BiH (17%), Croatia (25%), zero-rate exports
- NUMERIC(19,4) precision throughout — no float arithmetic
- Invoice statuses: Draft → Sent → Paid → Void
- Auto-generated invoice numbers (INV-XXXX format)
- PDF generation and download
- Email invoice to customer via SendGrid

Expense Tracking

- Create expenses with receipt photo upload (JPG/PNG → Cloudflare R2)
- Expense statuses: Pending → Approved → Paid / Rejected
- RBAC: Admin/Owner approval required for expenses
- Vendor management (contact book)

Financial Reports

- VAT Report (RS/BA/HR compliant) — monthly export
- Profit & Loss statement (by date range)
- Revenue and expense summaries

Chart of Accounts

- Standard Balkan chart of accounts pre-loaded
- Double-entry bookkeeping enforced (debit = credit, always)
- Exchange rate locking at transaction date

Authentication & Multi-Tenancy

- Organization-scoped data (strict multi-tenancy)
- RBAC: owner, admin, accountant, viewer roles
- JWT access tokens (15 min expiry) + refresh tokens (7 days)
- bcrypt password hashing (12 rounds)
- Audit trail: LoggedAction table (append-only, GDPR compliant)

Infrastructure

- Frontend: Next.js 15 on Vercel (bilko.io)
- Backend: Express API on Railway EU West (api.bilko.io, GDPR EU data residency)
- Database: PostgreSQL 15 on Railway (15 models, Prisma ORM)
- Storage: Cloudflare R2 (zero egress fees, GDPR compliant)
- Email: SendGrid (`noreply@bilko.io`)

Bug Fixes

N/A — initial release

Security Updates

N/A — initial release

Breaking Changes

N/A — initial release

Database Migrations

Migration	Description
`0001_initial_schema`	Create all 15 models: Organization, User, Account, Invoice, Expense, Contact, Transaction, etc.
`0002_indexes`	Performance indexes on organizationId foreign keys

Migration time: < 10 seconds on empty database

Known Issues

Issue	Severity	Workaround	Fix in
Bank import not yet available	Medium	Manual transaction entry	v1.1.0
2FA not yet implemented	Medium	Strong password required	v1.1.0
PDF export not yet available	Low	Print from browser	v1.1.0

Configuration Changes

No new environment variables for existing deployments. For new deployments, see `docs/templates/INFRASTRUCTURE/environment-configuration.md`.

Rollback Instructions

See rollback-plan.md — initial release, no previous version to roll back to.

Dependencies Updated

Package	From	To	Breaking
Next.js	—	15.0.0	N/A
React	—	19.0.0	N/A
Prisma	—	Latest	N/A
Vitest	—	Latest	N/A
Playwright	—	Latest	N/A

Release Notes Template — v[MAJOR.MINOR.PATCH]

Release Date: YYYY-MM-DD **Environment:** Production

Summary

[2-3 sentences: what this release contains, why it matters]

New Features

[Feature Area]

- **[Feature name]:** [Description. How it helps Bilko users.]

Bug Fixes

#	Description	Severity	Reported by
[GitHub #XXX]	[Bug description — user-facing language]	P0/P1/P2	[User/Internal]

Security Updates

CVE / Issue	Severity	Component	Description
[CVE-XXXX-XXXX]	Critical / High / Medium	[Component]	[Description]

Breaking Changes

Change	Migration Required	How to Migrate
[Breaking change description]	Yes / No	[Migration steps]

Database Migrations

Migration file	Description	Time (staging)
`XXXXXXXX_name`	[What it does]	X seconds

Configuration Changes

Variable	Change	Required	Notes
`NEW_ENV_VAR`	Added	Yes	[Description]
`OLD_ENV_VAR`	Removed	N/A	Replaced by NEW_ENV_VAR

Rollback Instructions

[Specific rollback steps for this release, if different from standard procedure]

Approval

Role	Name	Date	Signature
Author			
Reviewer	Alem Bašić		

Rollback Plan

Rollback Plan

> **Project:** Bilko **Version:** 0.1 **Date:** 2026-02-23 **Author:** Ops Architect **Status:** Draft **Reviewers:** Tech Lead, Alem Bašić

Document History

Version	Date	Author	Changes
0.1	2026-02-23	Ops Architect	Initial draft

---

1. Overview

This document defines the rollback procedure for Bilko deployments. A rollback restores the production environment to the previous working state when a deployment causes critical issues.

Rollback authority: Alem Bašić (primary). Any engineer can initiate rollback for P0 issues. **Target rollback time:** < 5 minutes for frontend, < 10 minutes for backend, < 60 minutes for database

---

2. Rollback Decision Criteria

Automatic Rollback Triggers

Initiate rollback immediately without waiting:

- Health check `https://api.bilko.io/health` returns non-200 for > 3 consecutive minutes
- Error rate > 5% of all API requests (Sentry)
- Any financial calculation producing provably incorrect results (VAT, double-entry)
- Authentication completely broken (no user can log in)

- Database migrations caused data corruption

Manual Rollback Triggers (Alem Baši? decision)

- P99 latency > 5s sustained for > 5 minutes
- Critical feature broken with no quick fix available
- Security vulnerability discovered in new release
- User-reported data loss

Do NOT Roll Back For

- Performance degradation < 20% (investigate first)
- Non-critical feature broken with workaround available
- Minor UI regressions
- Single user reporting an issue (investigate first)

---

3. Rollback Procedures by Component

3.1 Frontend Rollback (Vercel) — < 5 minutes

Vercel keeps all previous deployments. Rollback is instant (no rebuild).

Via Vercel Dashboard (recommended):

1. Open https://vercel.com/alai/bilko/deployments
2. Find the last successful deployment (before current broken one)
3. Click "..." → "Promote to Production"
4. Wait 30 seconds for propagation
5. Verify: `curl -I https://bilko.io` → HTTP/2 200

Via Vercel CLI:

```
# List recent deployments
vercel ls --prod

# Promote specific deployment
vercel rollback <deployment-url>
```

Verification:

```
curl -I https://bilko.io
# Open bilko.io in browser — should show previous version
```

Estimated time: < 2 minutes

3.2 Backend Rollback (Railway) — < 5 minutes

Railway keeps the last 10 deployments.

Via Railway Dashboard (recommended):

1. Open Railway Dashboard → Project → api service
2. Click "Deployments" tab
3. Find last successful deployment (look at deployment timestamp + status)
4. Click "..." → "Redeploy"
5. Wait for deployment to complete (~2 min)
6. Verify health check

Via Railway CLI:

```
# List recent deployments
railway deployments list --service api

# Note the previous deployment ID
# Redeploy via dashboard (CLI redeploy not yet supported)
```

Pre-rollback — if migration was included:

If the broken release included database migrations, you must decide:

- **Option A (preferred):** Write a forward-fix migration and deploy instead of rolling back
- **Option B:** Database rollback (see 3.3) — use only if Option A is not possible

Verification:

```
curl https://api.bilko.io/health
# Expected: {"status":"ok","db":"ok","timestamp":"..."}

# Test auth
curl -X POST https://api.bilko.io/api/v1/auth/login \
  -H "Content-Type: application/json" \
  -d '{"email":"test@bilko.io","password":"test123"}'
```

Estimated time: < 5 minutes

3.3 Database Rollback — < 60 minutes

WARNING: Database rollbacks are destructive. Any data written since the bad migration WILL BE LOST.

Before rolling back database:

1. Export all data created since the bad migration (if any):

   ```
   railway run psql $DATABASE_URL -c "
   COPY (SELECT * FROM invoices WHERE created_at > '[migration_time]') TO STDOUT;"
   ```

2. Assess data loss: is losing this data acceptable?
3. Prefer forward-fix migration if at all possible

If rollback is necessary:

```
# Step 1: Stop API traffic (put up maintenance page)
# Railway → api → Suspend

# Step 2: Restore from pre-deploy backup
# Railway Dashboard → PostgreSQL → Backups → Select backup taken before deploy

# OR restore from manual backup:
railway run psql $DATABASE_URL < pre_deploy_YYYYMMDD_HHMM.dump

# Step 3: Verify backup integrity
railway run psql $DATABASE_URL -c "SELECT COUNT(*) FROM invoices;"
railway run psql $DATABASE_URL -c "SELECT COUNT(*) FROM organizations;"

# Step 4: Redeploy previous backend version (without the bad migration)
# Railway → api → Deployments → Redeploy previous

# Step 5: Verify
railway run npx prisma db pull  # Should match backup schema
curl https://api.bilko.io/health

# Step 6: Resume API traffic
# Railway → api → Resume
```

```
# Step 7: If any data was lost, manually re-enter from exported data
```

Estimated time: 30–60 minutes

---

4. Rollback Verification Checklist

After any rollback, verify ALL of these before declaring rollback successful:

- ☐ API health: `curl https://api.bilko.io/health` → `{"status":"ok","db":"ok"}`
- ☐ Frontend loads: https://bilko.io opens without errors
- ☐ Login works: test account can authenticate
- ☐ Invoice creation works: create test invoice, verify totals
- ☐ VAT calculation correct: verify Serbia 20% on 1000 RSD = 200 RSD VAT
- ☐ BetterStack: all monitors green
- ☐ Sentry: no new error types
- ☐ Database counts: record counts match pre-deploy snapshot (if DB rollback)

---

5. Rollback Communication

During Rollback

1. Post in Slack #bilko-alerts immediately: "Initiating rollback — [reason]"
2. Update status.bilko.io: "We are investigating an issue and reverting recent changes"
3. Do not provide ETAs until rollback is verified successful

After Successful Rollback

1. Post in Slack #bilko-alerts: "Rollback complete — previous version restored — investigating root cause"
2. Update status.bilko.io: "Service restored — all systems operational"
3. If any user impact: send email to affected organizations within 2 hours
4. Create incident report within 24 hours

---

6. Version-Specific Rollback Notes

Release	Frontend Tag	Backend Tag	DB Migration Included	Rollback Notes
v1.0.0	Initial	Initial	`0001_initial_schema`, `0002_indexes`	First release — no rollback possible

(Update this table with each release)

---

7. Rollback Testing

Each major release must include a rollback test on staging before production deploy:

```
# Deploy to staging
# ... (standard deploy)

# Test rollback on staging
vercel rollback  # Frontend
railway deployments redeploy <previous-id>  # Backend

# Verify staging is back on previous version
curl https://staging-api.bilko.io/health
```

Document rollback test results in deployment checklist.

Related Documents

- Deployment Checklist
- Disaster Recovery Plan
- Operational Runbook
- Incident Report
- DEPLOYMENT.md

Approval

Role	Name	Date	Signature
Author	Ops Architect	2026-02-23	
Reviewer	Tech Lead		
Approver	Alem Bašić		

UAT Sign-Off

UAT Sign-Off

> **Project:** Bilko **Version:** 0.1 **Date:** 2026-02-23 **Author:** Ops Architect **Status:** Draft (Template — fill in before each major release) **Reviewers:** Alem Bašić

Document History

Version	Date	Author	Changes
0.1	2026-02-23	Ops Architect	Initial draft

INSTRUCTIONS

UAT (User Acceptance Testing) is required before:

- Initial production launch (v1.0.0)
- Any release adding new financial features
- Any release changing VAT rates or accounting rules

UAT is performed on staging environment (`bilko-staging` Railway + Vercel preview). File: `UAT-SIGNOFF-vX.X.X.md` in `docs/releases/`

UAT Sign-Off — Bilko v1.0.0 (MVP)

UAT Period: YYYY-MM-DD to YYYY-MM-DD **Testing Environment:** https://staging.bilko.io **Tested by:** Alem Bašić **Release version:** v1.0.0

---

1. UAT Test Scenarios

1.1 Authentication & Account Setup

#	Scenario	Steps	Expected Result	Actual Result	Pass/Fail
A1	Register new organization	Go to /register → Fill company name (Test Firma d.o.o.), email, password → Submit	Account created, redirect to dashboard		
A2	Login with credentials	Go to /login → Enter email + password → Submit	Dashboard loads with correct org name		
A3	Logout	Click user menu → Logout	Redirected to /login, session cleared		
A4	Wrong password rejected	Login with wrong password	401 error, no user enumeration		
A5	Invite team member (if implemented)	Settings → Team → Invite → Enter email	Invite email sent to new user		

1.2 Invoice Creation (Core — P0)

#	Scenario	Input	Expected	Actual	Pass/Fail
I1	Create invoice — Serbia VAT	Customer: Acme Corp, Items: 10x 5000 RSD @ 20% VAT	Subtotal: 50,000 RSD, VAT: 10,000 RSD, Total: 60,000 RSD		
I2	Create invoice — zero-rate export	Items: 1x 10,000 RSD @ 0% VAT	VAT: 0 RSD, Total: 10,000 RSD		
I3	Create invoice — EUR currency	Items: 5x 100 EUR @ 20% VAT	Subtotal: 500 EUR, VAT: 100 EUR, Total: 600 EUR		
I4	NUMERIC precision — no float drift	Items: 1x 33.33 RSD @ 20% VAT	VAT: 6.67 RSD, Total: 40.00 RSD (not 39.996...)		

#	Scenario	Input	Expected	Actual	Pass/Fail
I5	Invoice number auto-generated	Create invoice	Number = INV-XXXX format		
I6	Multi-item different VAT rates	Item 1: 1000 RSD @ 20%, Item 2: 500 RSD @ 0%	VAT: 200 RSD (only on item 1), Total: 1700 RSD		
I7	Invoice status flow: Draft → Sent	Create invoice (Draft) → Send	Status changes to Sent, email delivered		
I8	Mark invoice as paid	Click "Mark as Paid"	Status = Paid, paidAt timestamp set		

Financial precision test — critical:

For I4: Open browser DevTools → check API response body:

```
{
  "subtotal": "33.3300",
  "taxAmount": "6.6700",
  "totalAmount": "40.0000"
}
```

All amounts must be strings with 4 decimal places, never floating-point numbers.

1.3 Expense Tracking

#	Scenario	Expected	Actual	Pass/Fail
E1	Create expense with amount	Expense saved, status = Pending		
E2	Upload receipt photo	JPG uploads successfully, preview visible		
E3	Approve expense (as admin)	Status changes to Approved		
E4	Reject expense	Status changes to Rejected		
E5	Viewer cannot approve expense	403 error		

1.4 VAT Reporting

#	Scenario	Expected	Actual	Pass/Fail
V1	Generate VAT report — Serbia	Select period → Generate	Report shows: Output VAT (collected on invoices), Input VAT (paid on expenses), Net VAT payable	
V2	VAT report includes only own org data	Cross-org data not visible		
V3	VAT totals match invoice list	Sum of VAT from individual invoices = VAT report total		
V4	Date range filter works	Report only includes transactions in selected period		

Critical — VAT accuracy check:

After creating 3 invoices in test period:

- INV-001: 50,000 RSD + 10,000 VAT (20%)
- INV-002: 30,000 RSD + 5,100 VAT (17% BiH)
- INV-003: 10,000 RSD + 0 VAT (export)

VAT report must show:

- Output VAT: 15,100 RSD (10,000 + 5,100 + 0)
- Not 15,099.99 or 15,100.01

1.5 Multi-Tenancy Security (P0)

#	Scenario	Expected	Actual	Pass/Fail
T1	Create two test organizations	Both created separately		
T2	Org A cannot see Org B's invoices	Invoice list returns only Org A invoices		
T3	Org A cannot access Org B's invoice by ID	404 (not 403 — no data enumeration)		
T4	Viewer role cannot create invoice	403 error		

1.6 Performance

#	Scenario	Expected	Actual	Pass/Fail
P1	Invoice list loads in < 2s	Dashboard response < 2s		
P2	VAT report generates in < 5s	Report appears in < 5s on staging		
P3	Receipt upload completes in < 5s	Progress indicator, file appears in < 5s		

---

2. UAT Defects Log

#	Scenario	Defect Description	Severity	Status

---

3. UAT Summary

Category	Total Scenarios	Passed	Failed	Blocked
Authentication	5			
Invoice (incl. financial precision)	8			
Expenses	5			
VAT Reporting	4			
Multi-tenancy security	4			
Performance	3			
TOTAL	**29**			

---

4. Financial Accuracy Verification

Required sign-off item. Alem Bašić must verify:

- ☐ Invoice I1: Subtotal 50,000.0000, VAT 10,000.0000, Total 60,000.0000 □

☐ Invoice I4: Decimal precision correct (no float drift) □

☐ VAT Report V3: Report totals match sum of individual invoices □

☐ Multi-tenancy T2: Confirmed two orgs cannot see each other's data □

I confirm that the financial calculations in this release are accurate to NUMERIC(19,4) precision and comply with the applicable VAT rules for Serbia, Bosnia, and Croatia.

---

5. UAT Sign-Off Decision

Recommendation

☐ **APPROVED** — All P0 scenarios pass, no P0 defects open. Release approved for production.

☐ **CONDITIONAL** — Approved with conditions: [conditions listed below]

☐ **REJECTED** — P0 defects found. Do not deploy until resolved.

Conditions (if conditional):

Open Defects at Sign-Off

#	Defect	Severity	Resolution

Sign-Off

I have tested the scenarios listed above in the staging environment and confirm that the system is ready (or not ready) for production deployment based on the recommendation above.

Role	Name	Date	Signature
Primary UAT	Alem Bašić		
Tech Lead			

Related Documents

- Deployment Checklist
- Test Plan
- Release Notes
- TESTING-GUIDE.md