📘 Backend Runtime Maintenance & Troubleshooting Guide

HV Battery Backend Server - Production Deployment Guide

Version: 2.0 (Updated: February 2026)

Dokumentasi ini dibuat untuk membantu teknisi atau user melakukan **maintenance**, **troubleshooting**, dan **operasional** pada sistem **Backend Runtime (Node.js + Express + Prisma + Socket.IO)** yang sudah dibundle dan berjalan di server offline/online.

📦 1. Struktur Folder Runtime

Setelah proses bundling (`npm run bundle:be`), struktur folder akan seperti berikut:

```
backend-runtime/
├── dist/                   # Hasil build TypeScript (compiled JavaScript)
├── node_modules/           # Production dependencies
├── prisma/                 # Database schema & migrations
│   └── schema.prisma
├── package.json            # Node.js configuration
├── package-lock.json       # Dependency lock file
├── .env                    # Environment variables (PRODUCTION CONFIG)
├── DEPLOYMENT.md           # Panduan deployment detail
├── start.bat               # Quick start (simple mode)
├── start-pm2.bat           # PM2 auto-start (RECOMMENDED)
└── install-pm2.bat         # One-time PM2 installation
```

File ZIP Bundle:

- Nama: `backend-runtime.zip` _(disabled by default, manual folder copy)_

Atau: Copy langsung folder `backend-runtime/`

🚀 2. Cara Deploy ke Server Production

Step 1: Copy Folder ke Server

Option A: Manual Copy (Default)

```
bash
```

Copy folder backend-runtime ke server via:

- Network share

- USB drive

- Remote desktop

Option B: ZIP Transfer (jika diaktifkan)

```
bash

# Copy backend-runtime.zip lalu extract di server

unzip backend-runtime.zip

# atau PowerShell:

Expand-Archive -Path backend-runtime.zip -DestinationPath .
```

Step 2: Install PM2 (One-Time Only)

Untuk server yang ADA INTERNET:

```
bash
cd backend-runtime
install-pm2.bat

# atau manual:

npm install -g pm2
```

Untuk server OFFLINE:

- Install PM2 sebelum server offline

Atau install dari offline npm cache
PM2 hanya perlu diinstall SEKALI per server

Step 3: Konfigurasi Environment

Edit file `.env` sesuai production settings:

```
env
```

Database Connection

```
DATABASE_URL="sqlserver://SERVER_IP:1433;database=DB_TMMIN1_KRW_PIS_HV_BATTERY;user=sa;password=YOUR_PASSWORD;encrypt=false;trustServerCertificat
```

Server Configuration

```
PORT=4001 NODE_ENV=production
```

Optional Features (set false jika tidak tersedia)

```
REDIS_ENABLED=false SENTRY_ENABLED=false OTEL_ENABLED=false
```

Jika Redis tersedia (untuk caching - 83% faster!)

```
REDIS_HOST=localhost REDIS_PORT=6379
```

Step 4: Start Server

RECOMMENDED: PM2 Auto-Start

```
bash

Windows: Double-click file ini

start-pm2.bat

Atau manual:

pm2 start dist/index.js --name "hv-battery-backend" --time
pm2 startup
pm2 save
```

Alternative: Simple Start

```
bash

Windows: Double-click

start.bat

Atau manual:

node dist/index.js
```

Step 5: Verify Server Running

```
bash

# Check process

pm2 list

# Check port

netstat -ano | findstr :4001

# Test health endpoint

curl http://localhost:4001/api/health

# Expected: {"status":"healthy","database":"connected"}

# View logs

pm2 logs hv-battery-backend
```

⚙️ 3. Server Management Scripts

Development Mode (Local PC)

Start Development Server

```
bash
npm run dev          # Auto port cleanup + TypeScript watch + nodemon
npm run dev:direct   # Direct start (no auto cleanup)
```

Features:

- ✅ Auto-kill port 4001 if in use

✅ TypeScript watch mode (auto-compile on save)
✅ Nodemon auto-restart on code changes
✅ No more EADDRINUSE errors

Build & Test

```
bash
```

`npm run build # Compile TypeScript npm test # Run Vitest tests npm run test:ui # Vitest UI mode`

Production Mode (Server)

Start Server

```
bash
npm start           # Auto port cleanup + start production
npm run start:direct # Direct start (no auto cleanup)
```

Stop Server

```
bash
npm run stop        # Gracefully stop server
```

Restart Server

```
bash
npm run restart     # Stop + start server
```

PM2 Commands (Production)

```
bash
pm2 start dist/index.js --name "hv-battery-backend" --time
pm2 stop hv-battery-backend
pm2 restart hv-battery-backend
pm2 delete hv-battery-backend
pm2 logs hv-battery-backend
pm2 monit
pm2 list
```

🔄 4. Update / Deploy Versi Baru

Development PC (Build Bundle)

1. **Build Bundle Baru:**

```
bash
npm run bundle:be
```

Output:

```
📦 Step 1: Build TypeScript
🧹 Step 2: Clean old bundle
📂 Step 3: Copy files (dist/, prisma/, package.json, .env)
📦 Step 4: Install production dependencies
📝 Step 5: Create deployment guide
🔧 Step 6: Create start scripts
✅ Success! Folder ready: backend-runtime/
```

2. **Copy ke Server:**

```
bash

Copy folder backend-runtime ke server production
```

Production Server (Deploy)

1. **Backup Versi Lama:**

```
bash

# Windows
xcopy backend-runtime backend-runtime-backup /E /I /H /Y

# Linux
cp -r backend-runtime backend-runtime-backup
```

2. **Stop Server Lama:**

```
bash
pm2 stop hv-battery-backend

# atau
npm run stop
```

3. **Replace dengan Versi Baru:**

```
bash

# Hapus folder lama (kecuali .env jika ada custom confi
rm -rf backend-runtime/dist
rm -rf backend-runtime/node_modules

# Copy folder baru
# (via network share, USB, etc)
```

4. **Preserve Environment Config:**

```
bash

# Copy .env dari backup jika ada perubahan custom
cp backend-runtime-backup/.env backend-runtime/.env
```

5. **Start Server Baru:**

```
bash
cd backend-runtime
start-pm2.bat

atau

pm2 start dist/index.js --name "hv-battery-backend" --time
```

6. **Verify:**

```
bash
pm2 logs hv-battery-backend
curl http://localhost:4001/api/health
```

🛠 5. Troubleshooting

❌ Problem 1: Port Already in Use (EADDRINUSE)

Symptoms:

```
Error: listen EADDRINUSE: address already in use 0.0.0.0:4001
```

Solution A: Auto Cleanup (Development)

```
bash
npm run dev     # Dev mode auto-kills port
npm start       # Production mode auto-kills port
```

Solution B: Manual Cleanup

```
bash

Windows - Find process using port 4001

netstat -ano | findstr :4001

Kill process:

taskkill /PID  /F

Or use stop script:

npm run stop

Linux

lsof -i :4001
kill -9
```

Solution C: PM2

```
bash
pm2 list                       # Check running processes
pm2 stop hv-battery-backend    # Stop conflicting process
pm2 delete hv-battery-backend  # Remove from PM2
pm2 start dist/index.js --name "hv-battery-backend" --time
```

❌ Problem 2: Database Connection Failed

Symptoms:

```
[FATAL] Database connection failed
Could not connect to SQL Server
```

Solution:

1. **Check SQL Server Running:**

```
bash

# Windows
sc query MSSQLSERVER

# Should show: RUNNING

# Start if stopped:
net start MSSQLSERVER
```

2. **Verify DATABASE_URL in .env:**

```
env
DATABASE_URL="sqlserver://localhost:1433;database=DB_TMMIN1_KRW_PIS_HV_BATTERY;user=sa;password=YOUR_PASSWORD;encrypt=false;trustServerCertifi
```

3. **Test Connection Manually:**

```
bash
sqlcmd -S localhost -U sa -P YOUR_PASSWORD -Q "SELECT @@VERSION"
```

4. **Check Firewall:**

```
bash

# Windows: Allow port 1433
netsh advfirewall firewall add rule name="SQL Server" dir=in action=allow protocol=TCP localport=1433
```

5. **Regenerate Prisma Client:**

```
bash
cd backend-runtime
npx prisma generate
pm2 restart hv-battery-backend
```

❌ Problem 3: Prisma Client Error

Symptoms:

```
PrismaClientInitializationError
Cannot find module '@prisma/client'
```

Solution:

```
bash
cd backend-runtime
npm install --omit=dev
npx prisma generate
pm2 restart hv-battery-backend
```

❌ Problem 4: Nodemon Restart Errors (Development)

Symptoms:

```
[nodemon] restarting...
Error: EADDRINUSE
```

Solution (Already Fixed):

- Nodemon delay increased: 2500ms → 3500ms

Graceful shutdown delay: +500ms
Total buffer: ~4 seconds

If still occurs:

```
bash
```

Stop and restart clean

`npm run stop npm run dev`

❌ Problem 5: Server Crashes / Not Responding

Check Logs:

```
bash
pm2 logs hv-battery-backend --lines 100
```

Common Issues:

1. **Out of Memory:**

```
bash

Increase Node.js memory (if needed)

pm2 start dist/index.js --name "hv-battery-backend" --max-memory-restart 1024M
```

2. **Uncaught Exception:**

- Check logs for stack trace

Graceful shutdown should catch most errors
Update code and redeploy

3. **PM2 Process Stuck:**

```
bash
```

`pm2 delete hv-battery-backend pm2 start dist/index.js --name "hv-battery-backend" --time`

❌ Problem 6: Socket.IO Not Connecting

Symptoms:

- WebSocket connections fail

Real-time updates not working

Solution:

1. **Check WebSocket URL:**

```
javascript
```

`// Client should connect to: ws://SERVER_IP:4001`

2. **Check Firewall:**

```
bash

Allow port 4001

netsh advfirewall firewall add rule name="HV Battery Backend" dir=in action=allow protocol=TCP localport=4001
```

3. **Check Backend Logs:**

```
bash
pm2 logs hv-battery-backend | grep "WS"

Look for WebSocket connection messages
```

4. **Test Manually:**

```
bash

In browser console:


const socket = io('http://localhost:4001')
socket.on('connect', () => console.log('Connected!'))
```

🧹 6. Maintenance Rutin

Daily

Check Server Status:

```
bash
pm2 list
pm2 monit
```

View Logs:

```
bash
pm2 logs hv-battery-backend --lines 50
```

Weekly

Restart Server (Clear Memory):

```
bash
pm2 restart hv-battery-backend
```

Clear Logs:

```
bash
pm2 flush
```

Monthly

Backup Database:

```
bash

SQL Server backup


BACKUP DATABASE DB_TMMIN1_KRW_PIS_HV_BATTERY
TO DISK = 'D:\Backup\HV_BATTERY_backup.bak'
WITH FORMAT, INIT, COMPRESSION
```

Backup Runtime Folder:

```
bash

# Windows
xcopy backend-runtime D:\Backup\backend-runtime-$(date +%Y%m%d) /E /I /H /Y

# Linux
tar -czf backup-$(date +%Y%m%d).tar.gz backend-runtime/
```

Check Disk Space:

```
bash

# Windows
wmic logicaldisk get size,freespace,caption

# Linux
df -h
```

Update Dependencies (if needed):

```
bash

# Only if security updates required
cd backend-runtime
npm update --omit=dev
pm2 restart hv-battery-backend
```

📊 7. Monitoring & Performance

Built-in Features

Logging (Pino)

```
bash

View structured JSON logs

pm2 logs hv-battery-backend

Filter by level

pm2 logs hv-battery-backend | grep ERROR
pm2 logs hv-battery-backend | grep WARN
```

Health Check Endpoint

```
bash
curl http://localhost:4001/api/health
```

Response:

```
json
{
  "status": "healthy",
  "database": "connected",
  "timestamp": "2026-02-07T10:30:00.000Z",
  "uptime": 3600
}
```

API Documentation (Swagger)

```
http://localhost:4001/api-docs
```

Cache Performance (if Redis enabled)

- Cache Hit Rate: 83.16% faster

Sequences: 30s TTL
Historical data: 15min TTL

Database Performance

- Indexes created: 98.19% faster queries

Pagination: 95.26% improvement

Optional Monitoring (if enabled)

Sentry Error Tracking

```
env
```

`SENTRY_ENABLED=true SENTRY_DSN=your_sentry_dsn`

OpenTelemetry Tracing

```
env
OTEL_ENABLED=true
OTEL_EXPORTER_OTLP_ENDPOINT=http://your-collector:4318
```

Redis Caching

```
env
REDIS_ENABLED=true
REDIS_HOST=localhost
REDIS_PORT=6379
```

🔒 8. Security & Best Practices

Environment Variables

NEVER commit .env to Git!

```
bash

Add to .gitignore


.env
.env.production
.env.local
```

Database Credentials

Change default passwords:

```
env

Bad (default)


DATABASE_URL="...user=sa;password=aas..."

Good (strong password)


DATABASE_URL="...user=sa;password=Str0ng_P@ssw0rd_2026..."
```

Firewall Rules

Only allow required ports:

```
bash

# Backend application

Port 4001 - Allow inbound TCP

# SQL Server

Port 1433 - Allow from specific IPs only

# Redis (if used)

Port 6379 - Localhost only
```

PM2 Security

Enable PM2 authentication (if exposed):

```
bash
pm2 web --auth username:password
```

Regular Updates

Keep Node.js updated:

- Current: v24.13.0

Check: https://nodejs.org/en/

Monitor security advisories:

```
bash
```

`npm audit npm audit fix`

📁 9. File Structure Reference

Development Files (backend/)

```
backend/
├── src/                    # TypeScript source code
│   ├── controllers/        # API controllers
│   ├── routes/             # Express routes
│   ├── services/           # Business logic
│   ├── ws/                 # WebSocket handlers
│   ├── utils/              # Utilities
│   ├── config/             # Configuration
│   ├── app.ts              # Express app setup
│   └── index.ts            # Entry point
├── prisma/
│   └── schema.prisma       # Database schema
├── dist/                   # Compiled JavaScript (git ignored)
├── package.json
├── tsconfig.json
├── nodemon.json
├── .env
├── start-server.ps1        # Dev server starter
├── stop-server.ps1         # Server stopper
├── restart-server.ps1      # Server restarter
├── dev-server.ps1          # Dev mode starter
└── build-backend-bundle.js # Bundle creator
```

Production Bundle (backend-runtime/)

```
backend-runtime/
├── dist/                   # Production JS
├── node_modules/           # Production dependencies only
├── prisma/                 # Database schema
├── package.json
├── .env                    # Production config (EDIT THIS!)
├── DEPLOYMENT.md           # Deployment guide
├── start.bat               # Simple start
├── start-pm2.bat           # PM2 start (recommended)
└── install-pm2.bat         # PM2 installer
```

📞 10. Kontak Support

Development Team:

```
Divisi Pengembang Sistem
Developer: Risyan
Email: risyan@adaptive.co.id
WhatsApp: +62 899-1908-349
```

Technical Support:

- Issue Tracker: (Link to internal ticketing system)

Documentation: PROJECT-COMPLETE.md, BUNDLE-GUIDE.md
Deployment: DEPLOYMENT.md

📚 11. Additional Documentation

Lihat file-file berikut untuk detail lebih lanjut:

1. **PHASE1-IMPLEMENTATION.md** - Security fixes & error handling

PHASE-2-SUMMARY.md - Modern tooling (Pino, Redis, Sentry, etc)
PHASE3-PROGRESS.md - Scalability (caching, indexes, pagination)
PROJECT-COMPLETE.md - Complete project summary
BUNDLE-GUIDE.md - Offline deployment guide

DEPLOYMENT-CHECKLIST.md - Pre-deployment verification
SERVER-MANAGEMENT.md - Server script documentation
QUICK-START.md - Quick deployment reference

✅ Penutup

Dokumen ini mencakup semua aspek maintenance dan troubleshooting untuk **HV Battery Backend Server**.

Key Features Implemented:

- ✅ Phase 1: Security (SQL injection prevention, CORS, validation, graceful shutdown)

✅ Phase 2: Modern tooling (Pino logging, Redis cache, Sentry, OpenTelemetry, Swagger)
✅ Phase 3: Scalability (Database indexes 98% faster, caching 83% faster, pagination 95% faster)
✅ Auto port cleanup (no more EADDRINUSE errors)
✅ Smart deployment scripts (Windows .bat files)
✅ PM2 auto-startup configuration
✅ Offline deployment support
✅ Comprehensive documentation

System Requirements:

- Node.js: v18.0.0+ (recommended: v24.13.0)

SQL Server: 2017+
RAM: 512MB minimum, 1GB+ recommended
Disk: 500MB+ for application

Performance Metrics:

- Database queries: 98.19% faster with indexes

Cache hits: 83.16% faster with Redis
Pagination: 95.26% improvement
Graceful shutdown: <5 seconds
Auto-restart: <10 seconds

Document Version: 2.0

Last Updated: February 7, 2026 **Status:** Production Ready ✅

Document Version: 2.0 **Last Updated:** February 7, 2026 **Status:** Production Ready ✅