# EMFoS Admin Guide

## System Administration & Configuration

**Version**: 1.0.0
**For**: System Administrators & IT Teams

---

## 🏷 Table of Contents



---

## System Overview

### Architecture

**Frontend**: Next.js 14 (React) **Backend**: Next.js API Routes **Database**: PostgreSQL **Email Service**: Resend **Hosting**: Vercel (recommended)

### Key Components

- **Dashboard**: Real-time task overview
- **Schedule**: Calendar-based task visualization
- **Task Management**: Status tracking and evidence upload
- **Reporting**: Analytics and compliance reports
- **Cron Jobs**: Automated email alerts

---

## User Management

## User Roles

### PROJECT_MANAGER

- Full create/edit/delete access
- Can manage all activities and tasks
- Access to settings and user management
- Can trigger manual alert checks

### VIEWER

- Read-only access to all pages
- Can view dashboards, schedules, reports
- Cannot create or modify data
- Cannot access settings

## Managing Users

### Add New User

1. **Access Settings** (PROJECT_MANAGER only)
2. **User Management** section
3. Click **"Invite New User"**
4. Enter:
    - Email address
    - Full name
    - Role (PROJECT_MANAGER or VIEWER)
5. Click **"Send Invite"**
6. User receives email with login link

### Change User Role

1. Go to **Settings → User Management**
2. Find user in list
3. Click **"Edit"**
4. Change role
5. Click **"Save"**

### Deactivate User

1. Go to **Settings → User Management**
2. Find user
3. Click **"Deactivate"**
4. Confirm action
5. User loses access immediately

### User Permissions Matrix

| Action | PROJECT_MANAGER | VIEWER |
|---|---|---|
| Create Activity | ✅ | ❌ |
| Edit Activity | ✅ | ❌ |
| Create Task | ✅ | ❌ |
| Update Task Status | ✅ | ❌ |
| Upload Evidence | ✅ | ❌ |
| View Dashboard | ✅ | ✅ |
| View Reports | ✅ | ✅ |
| Access Settings | ✅ | ❌ |
| Manage Users | ✅ | ❌ |

---

# Database Administration

## Connection Details

**Host**: localhost (or your configured host) **Port**: 5432 **Database**: env_monitor **User**: $PGUSER (environment variable) **Password**: $PGPASSWORD (environment variable)

## Backup Strategy

### Daily Backups

```
# Automated daily backup (add to cron)
0 2 * * * pg_dump -h localhost -U $PGUSER env_monitor > /backups/env_mo
```

### Manual Backup

```
pg_dump -h localhost -U $PGUSER env_monitor > env_monitor_backup.sql
```

### Restore from Backup

```
psql -h localhost -U $PGUSER env_monitor < env_monitor_backup.sql
```

## Database Maintenance

### Check Database Size

```
psql -h localhost -U $PGUSER -c "SELECT pg_size_pretty(pg_database_size
```

### Vacuum & Analyze

```
# Run weekly for performance
psql -h localhost -U $PGUSER -c "VACUUM ANALYZE;" env_monitor
```

### Monitor Connections

```
psql -h localhost -U $PGUSER -c "SELECT count(*) FROM pg_stat_activity W
```

## Schema Management

### View Current Schema

```
psql -h localhost -U $PGUSER env_monitor -c "\dt"
```

### Run Migrations

```
# After code updates
npx prisma migrate deploy

# Create new migration
npx prisma migrate dev --name description
```

---

# Email Configuration

## Resend Setup

### 1. Create Resend Account

- Go to https://resend.com
- Sign up for free account
- Verify email

### 2. Get API Key

1. Go to Resend Dashboard
2. Click "API Keys"
3. Create new API key
4. Copy key (starts with `re_`)

### 3. Configure in Application

1. Go to Vercel/hosting dashboard
2. Set environment variable: `RESEND_API_KEY`
3. Redeploy application

### 4. Verify Sender Email

1. In Resend dashboard, go to "Domains"
2. Add your domain or use default sender
3. Follow verification steps

## Email Alert Configuration

### Cron Schedule

Located in `vercel.json`:

```
{
  "crons": [
    {
      "path": "/api/cron/overdue-alerts",
      "schedule": "0 9 * * *"
    }
  ]
}
```

**Schedule Format** (cron):

- `0 9 * * *` = Daily at 9:00 AM UTC
- `0 14 * * *` = Daily at 2:00 PM UTC
- `0 */6 * * *` = Every 6 hours
- `0 9 * * 1` = Every Monday at 9:00 AM UTC

#### Manual Alert Trigger

1. Go to **Settings** → **Notifications**
2. Click **"Trigger Manual Alert Check"**
3. Check Resend dashboard for email delivery
4. Verify email was sent

#### Email Template

Located in `app/api/cron/overdue-alerts/route.ts`

Customize:

- Sender email: `from: 'alerts@yourdomain.com'`
- Subject line: `subject: 'URGENT: Monitoring Overdue'`
- Email body: HTML template

---

# Monitoring & Maintenance

## Application Health Checks

### Check Server Status

```
# Test API endpoint
curl https://your-domain.com/api/cron/overdue-alerts

# Should return 401 (unauthorized) without CRON_SECRET
```

### Monitor Logs

**Vercel Logs**:

1. Go to Vercel Dashboard
2. Click project
3. Go to "Deployments"
4. Click latest deployment
5. View "Logs" tab

**Database Logs**:

```
# View PostgreSQL logs
tail -f /var/log/postgresql/postgresql.log
```

## Performance Monitoring

### Database Query Performance

```
# Enable query logging
psql -h localhost -U $PGUSER -c "ALTER SYSTEM SET log_min_duration_state

# Restart PostgreSQL
sudo systemctl restart postgresql
```

### Application Performance

1. Enable Vercel Analytics
2. Monitor Core Web Vitals
3. Check response times
4. Review error rates

### Uptime Monitoring

Set up external monitoring:

- **Uptime Robot**: https://uptimerobot.com
- **Pingdom**: https://www.pingdom.com
- **StatusPage**: https://www.statuspage.io

Configure to monitor:

- Main application URL
- API endpoints
- Database connectivity

---

# Backup & Recovery

## Backup Schedule

**Daily**: Automated database backups **Weekly**: Full application backup (code + database) **Monthly**: Archive to cold storage

## Recovery Procedures

### Database Recovery

```
# 1. Stop application
# 2. Restore from backup
psql -h localhost -U $PGUSER env_monitor < backup.sql

# 3. Verify data
psql -h localhost -U $PGUSER -c "SELECT COUNT(*) FROM task;" env_monito

# 4. Restart application
```

### Application Recovery

```
# 1. Redeploy from GitHub
git pull origin main
npm run build
```

```
npm run start

# 2. Verify deployment
curl https://your-domain.com
```

---

# Security

## Access Control

### Environment Variables

**Never commit to git**:

- `DATABASE_URL`
- `RESEND_API_KEY`
- `CRON_SECRET`

**Store in**:

- Vercel Environment Variables
- `.env.local` (local development only)

### Database Security

```
# Change default password
ALTER USER $PGUSER WITH PASSWORD 'new_strong_password';

# Restrict connections
# Edit pg_hba.conf to limit access
```

### API Security

- All API routes validate `CRON_SECRET`
- Role-based access control enforced
- HTTPS required in production
- CORS configured for trusted domains

## Security Checklist

- [ ] Change default admin password
- [ ] Enable 2FA on all admin accounts
- [ ] Rotate API keys monthly
- [ ] Enable database backups
- [ ] Monitor access logs
- [ ] Keep dependencies updated
- [ ] Use strong CRON_SECRET (32+ characters)
- [ ] Enable HTTPS/SSL
- [ ] Configure firewall rules
- [ ] Regular security audits

### Dependency Updates

```
# Check for vulnerabilities
npm audit

# Update dependencies
npm update

# Update major versions (carefully)
npm outdated
npm install package@latest
```

---

# Troubleshooting

## Issue: Cron Jobs Not Running

**Symptoms**: Email alerts not being sent

**Solutions**:

1. Verify `vercel.json` exists in root
2. Check `CRON_SECRET` matches in code
3. Redeploy application
4. Check Vercel cron logs
5. Test manually: `curl https://your-domain.com/api/cron/overdue-alerts`

## Issue: Database Connection Errors

**Symptoms**: "Cannot connect to database"

**Solutions**:

1. Verify `DATABASE_URL` is correct
2. Check PostgreSQL is running: `pg_isready -h localhost`
3. Verify credentials: `psql -h localhost -U $PGUSER`
4. Check firewall rules
5. Verify database exists: `psql -l`

## Issue: Email Not Sending

**Symptoms**: No emails received from alerts

**Solutions**:

1. Verify `RESEND_API_KEY` is correct
2. Check Resend dashboard for errors
3. Verify sender email is configured
4. Test manually from Settings page
5. Check spam/junk folder
6. Verify email address is correct

## Issue: High Database Usage

**Symptoms**: Slow queries, high CPU

**Solutions**:

1. Run `VACUUM ANALYZE`
2. Check for missing indexes
3. Review slow query logs
4. Archive old data
5. Optimize queries
6. Increase database resources

## Issue: Application Crashes

**Symptoms**: 500 errors, blank page

**Solutions**:

1. Check Vercel logs for errors

2. Check database connection
3. Verify environment variables
4. Check for TypeScript errors: `npm run build`
5. Redeploy application
6. Rollback to previous version if needed

---

# Maintenance Schedule

## Daily

- Monitor application logs
- Check email alert delivery
- Verify database connectivity

## Weekly

- Run `VACUUM ANALYZE`
- Review performance metrics
- Check backup completion

## Monthly

- Update dependencies
- Review security logs
- Rotate API keys
- Archive old data
- Performance review

## Quarterly

- Full security audit
- Disaster recovery test
- Capacity planning review
- User access review

---

# Support & Resources

**Documentation**:

- User Manual: `USER_MANUAL.md`

- Quick Start: `QUICK_START_GUIDE.md`
- Deployment: `VERCEL_DEPLOYMENT_STEPS.md`

**External Resources**:

- PostgreSQL Docs: https://www.postgresql.org/docs/
- Vercel Docs: https://vercel.com/docs
- Next.js Docs: https://nextjs.org/docs
- Resend Docs: https://resend.com/docs

**Contact**:

- Email: admin@emfos.app
- Support: support@emfos.app

---

**Document Version**: 1.0.0
**Last Updated**: January 15, 2026
**Next Review**: April 15, 2026