Ampere Business Management System - NAS Deployment Guide

📦 Production-Ready Package

This package contains a clean, production-ready version of the Ampere Business Management System, optimized for NAS deployment.

🧹 What Was Cleaned Up

- ✅ Removed development documentation files
- ✅ Removed test scripts and development utilities
- ✅ Cleaned build cache and temporary files
- ✅ Removed backup and log files
- ✅ Optimized for production deployment
- ✅ Configured for containerized deployment

📁 Package Contents

```
ampere_business_management/
├── app/                      # Main application directory
│   ├── Dockerfile            # Docker configuration
│   ├── docker-compose.yml    # Docker Compose setup
│   ├── next.config.js        # Next.js production config
│   ├── .dockerignore         # Docker ignore patterns
│   ├── package.json          # Dependencies and scripts
│   ├── .env                  # Environment configuration
│   ├── prisma/               # Database schema and migrations
│   ├── scripts/              # Production scripts (seed only)
│   ├── app/                  # Next.js app directory
│   ├── components/           # React components
│   ├── lib/                  # Utility libraries
│   ├── types/                # TypeScript definitions
│   └── public/               # Static assets
```

🚀 NAS Deployment Instructions

Prerequisites

1. **Docker Support**: Ensure your NAS supports Docker
    - Synology: Docker package from Package Center

   - QNAP: Container Station from App Center
   - Other NAS: Docker Engine installed
2. **Required Resources**:
   - RAM: Minimum 4GB, Recommended 8GB
   - Storage: Minimum 10GB free space
   - Network: Internet connection for initial setup

Step 1: Transfer Files to NAS

1. Copy the entire `ampere_business_management` folder to your NAS
2. Recommended location: `/volume1/docker/ampere-business-management/`

Step 2: Configure Environment

1. Navigate to the app directory on your NAS
2. Edit the `.env` file and update the following values:

```
# Update this with your NAS IP or domain
NEXTAUTH_URL="http://your-nas-ip:3000"

# Update this with your NAS IP or domain
XERO_REDIRECT_URI="http://your-nas-ip:3000/api/xero/callback"
```

Example for NAS with IP 192.168.1.100:

```
NEXTAUTH_URL="http://192.168.1.100:3000"
XERO_REDIRECT_URI="http://192.168.1.100:3000/api/xero/callback"
```

Step 3: Deploy with Docker Compose

1. **Via SSH/Command Line**:

   ```
   cd /volume1/docker/ampere-business-management/app
   docker-compose up -d
   ```

2. **Via NAS GUI**:

   - **Synology**: Use Docker package, import docker-compose.yml
   - **QNAP**: Use Container Station, create stack from compose file

Step 4: Initialize Database

After the container is running, initialize the database:

```
# Enter the container
docker exec -it ampere-business-app sh

# Run database migrations
npx prisma migrate deploy
```

```
# Seed the database with initial users
npx prisma db seed

# Exit the container
exit
```

Step 5: Access the Application

1. Open web browser
2. Navigate to: `http://your-nas-ip:3000`
3. Login with default accounts:

Super Admin Accounts: - Username: `zack` / Password: `Czl914816` - Username: `endy` / Password: `Endy548930`

Other Accounts: - Project Manager: `pm` / Password: `password123` - Finance: `finance` / Password: `password123`

🔧 Configuration Options

Port Configuration

To change the port from 3000 to another port (e.g., 8080):

1. Edit `docker-compose.yml`:

   ```
   ports:
     - "8080:3000"  # Change 8080 to your desired port
   ```

2. Update `.env` file:

   ```
   NEXTAUTH_URL="http://your-nas-ip:8080"
   XERO_REDIRECT_URI="http://your-nas-ip:8080/api/xero/callback"
   ```

SSL/HTTPS Setup

For production use with HTTPS:

1. Set up a reverse proxy (nginx, Traefik)

2. Configure SSL certificates

3. Update environment variables:

   ```
   NEXTAUTH_URL="https://your-domain.com"
   XERO_REDIRECT_URI="https://your-domain.com/api/xero/callback"
   ```

External Database

To use an external PostgreSQL database:

1. Update `.env` with your database URL:

   ```
   DATABASE_URL="postgresql://username:password@hostname:5432/database"
   ```

🔍 Health Checks & Monitoring

Container Health

The container includes health checks. Monitor status:

```
docker ps
docker logs ampere-business-app
```

Application Health

Access health endpoint: `http://your-nas-ip:3000/api/health`

Log Monitoring

```
# View container logs
docker logs -f ampere-business-app

# View last 100 lines
docker logs --tail 100 ampere-business-app
```

🛠 Maintenance

Updates

To update the application:

1. Stop the container:

   ```
   docker-compose down
   ```

2. Replace files with updated version

3. Rebuild and start:

   ```
   docker-compose up -d --build
   ```

Backup

Database Backup:

```
docker exec ampere-business-app npx prisma db push --force-reset
```

File Backup: - Backup the entire application directory - Include uploaded files and certificates

Restart Services

```bash
# Restart application
docker-compose restart

# View status
docker-compose ps
```

📊 Performance Optimization

Resource Limits

Add to `docker-compose.yml`:

```yaml
services:
  ampere-business-management:
    # ... other config
    deploy:
      resources:
        limits:
          cpus: '2.0'
          memory: 4G
        reservations:
          cpus: '1.0'
          memory: 2G
```

Caching

The application uses: - Next.js built-in caching - Static asset optimization - Image optimization

🔒 Security Considerations

1. **Change Default Passwords**: Update all default user passwords
2. **Network Security**: Use firewall rules to limit access
3. **SSL/TLS**: Enable HTTPS in production
4. **Regular Updates**: Keep the application and Docker updated
5. **Backup Strategy**: Implement regular backups

🆘 Troubleshooting

Common Issues

1. **Port Conflicts**:

   ```
   # Check if port is in use
   netstat -tlnp | grep :3000
   ```

2. **Database Connection**:

   ```
   # Test database connectivity
   docker exec ampere-business-app npx prisma db push --force-reset
   ```

3. **Environment Variables**:

   ```
   # Check environment inside container
   docker exec ampere-business-app env | grep -E "(DATABASE_URL|NEXTAUTH)"
   ```

4. **File Permissions**:

   ```
   # Fix file permissions
   chmod -R 755 /volume1/docker/ampere-business-management/
   ```

Getting Help

- Check container logs for error messages
- Verify all environment variables are set correctly
- Ensure database is accessible
- Confirm network connectivity

✅ Production Checklist

- □ Files transferred to NAS
- □ Environment variables configured
- □ Docker container running
- □ Database migrated and seeded
- □ Application accessible via web browser
- □ Default passwords changed
- □ SSL/HTTPS configured (if required)
- □ Backup strategy implemented
- □ Firewall rules configured

📞 Support

This is a self-contained, production-ready deployment package. All dependencies and configurations are included for immediate deployment on your NAS system.

For additional customization or integration requirements, refer to the application documentation or contact your system administrator.

Package Version: Production-Ready v1.0

Last Updated: September 2024

Compatibility: Docker-enabled NAS systems