Timebars Solution Deployment Guide

This guide outlines the two deployment scenarios available for the Timebars solution, including system requirements, necessary expertise, and ongoing maintenance considerations.

Deployment Scenarios Overview

Timebars Ltd. offers two deployment options to meet different organizational needs:

- **Scenario 1: Docker Container Deployment** - Pre-built Docker images for rapid deployment
- **Scenario 2: Source Code Deployment** - Full source code access for custom builds and modifications

---

Scenario 1: Docker Container Deployment

Overview

The entire Timebars solution is containerized and deployed using **Docker**, with all images hosted on Docker Hub. This modern deployment approach simplifies installation, ensures consistency across environments, and enables rapid scaling. The complete solution requires just **5 Docker containers**:

- **jimecox807/tbrun** - Frontend client application (Timebars/Agilebars/Costbars)
- **jimecox807/tbwwwp** - Sales and support website with Cloud Dashboard
- **jimecox807/tbbe** - Strapi backend services
- **jimecox807/tbfm** - File management and CMS system
- **jimecox807/tbcdn** - CDN services for media delivery

All containers are orchestrated together to provide a complete, production-ready Timebars solution that can be deployed in minutes.

Access Credentials

Upon purchase, you will receive:

- Docker Hub username and password, or
- Docker Hub access token/key for secure image downloads
- Docker Compose configuration files
- Environment configuration templates
- Initial setup documentation

System Requirements

Hardware (Minimum for Production)

- 4 CPU cores
- 16 GB RAM
- 100 GB SSD storage
- Stable internet connection for initial download

Software

- Docker Engine 20.10 or higher
- Docker Compose 2.0 or higher
- Linux (Ubuntu 20.04+, CentOS 8+, or RHEL 8+), Windows Server 2019+, or macOS 11+
- PostgreSQL 13+ (can be deployed as separate container or managed service)

Network

- Open ports: 80 (HTTP), 443 (HTTPS), and custom ports as configured
- Domain name(s) configured with DNS
- SSL/TLS certificates (recommended: Let's Encrypt)

Required Personnel & Skills

For Initial Deployment (1-2 people)

DevOps Engineer or System Administrator

- **Required Skills:**
  - Docker and Docker Compose proficiency
  - Basic networking (DNS, SSL/TLS, port forwarding)
  - Linux/Windows server administration
  - Environment variable configuration
  - Reverse proxy setup (Nginx, Apache, or Traefik)
- **Estimated Time:** 4-8 hours for initial setup

Database Administrator (Optional but Recommended)

- **Required Skills:**
  - PostgreSQL installation and configuration
  - Database backup and recovery procedures
  - Performance tuning
- **Estimated Time:** 2-4 hours for database setup

For Ongoing Maintenance

System Administrator (Part-time)

- **Responsibilities:**
  - Monitor container health and logs
  - Apply Docker image updates
  - Manage backups and disaster recovery
  - Monitor system resources and scaling
  - Security patch management
- **Time Commitment:** 2-5 hours per week

Database Administrator (Part-time)

- **Responsibilities:**
  - Database backups and maintenance

    - Performance monitoring and optimization
    - Data integrity checks
- **Time Commitment:** 1-3 hours per week

Deployment Steps (High-Level)

1. Authenticate with Docker Hub using provided credentials
2. Download Docker images for all 5 containers
3. Configure environment variables and secrets
4. Set up PostgreSQL database
5. Deploy containers using Docker Compose
6. Configure reverse proxy and SSL certificates
7. Verify all services are running and communicating
8. Perform initial system configuration
9. Set up monitoring and backup procedures

Advantages

- **Fast deployment** - Production ready in hours, not days
- **Lower technical barrier** - Minimal configuration required
- **Consistent environments** - Same configuration across dev, staging, and production
- **Easy updates** - Pull new images and restart containers
- **Simplified troubleshooting** - Standardized container configurations

Considerations

- Limited ability to customize application code
- Dependent on Docker Hub for image access
- Requires Docker expertise for troubleshooting
- Updates managed by Timebars Ltd. release schedule

Scenario 2: Source Code Deployment

Overview

For organizations requiring full control and customization capabilities, the complete Timebars source code is available through Bitbucket private repositories. This option allows you to build, modify, and deploy the solution according to your specific requirements.

Access Credentials

Upon purchase, you will receive:

- Bitbucket account credentials or SSH keys
- Access to private repositories
- Build and deployment documentation
- API keys and configuration guides

Bitbucket Repository Structure

The Timebars solution consists of six repositories:

1. **cdn** - Plain HTML/CSS/JS website for static content delivery
2. **dockeretc** - Docker Compose files, deployment scripts, and technical documentation
3. **tbwwwp2** - Next.js 14 sales website and Cloud Dashboard
4. **tbbe** - Strapi v4 CMS backend
5. **tbrunp** - Timebars client applications (Timebars/Agilebars/Costbars)
6. **filemanager** - Optional component for uploading files to the CDN

System Requirements

Hardware (Minimum for Production)

- 4 CPU cores
- 16 GB RAM
- 150 GB SSD storage (additional space for source code and builds)
- Stable internet connection

Software

- Node.js 18.x or higher
- NPM 9.x or higher
- Git 2.30 or higher
- Docker Engine 20.10+ and Docker Compose 2.0+ (optional, for containerized deployment)
- PostgreSQL 13+
- Linux (Ubuntu 20.04+, CentOS 8+, or RHEL 8+), Windows Server 2019+, or macOS 11+

Build Tools

- Parcel.js (for tbrunp)
- Next.js build tools (for tbwwwp2)
- Strapi build dependencies (for tbbe)

Network

- Same requirements as Scenario 1
- May require additional ports for development environments

Required Personnel & Skills

For Initial Deployment (2-4 people)

Full-Stack Developer

- **Required Skills:**
  - JavaScript/Node.js expertise
  - React and Next.js framework knowledge
  - Understanding of Strapi CMS architecture
  - jQuery and jQuery UI experience (for client apps)
  - Package management (NPM/Yarn)
  - Build tools (Parcel.js, Webpack)

    - Git version control
- **Estimated Time:** 16-32 hours for initial build and deployment

DevOps Engineer

- **Required Skills:**
    - Server provisioning and configuration
    - CI/CD pipeline setup
    - Reverse proxy configuration
    - SSL/TLS certificate management
    - Environment management
    - Deployment automation
    - Docker (if containerizing custom builds)
- **Estimated Time:** 12-24 hours for infrastructure setup

Database Administrator

- **Required Skills:**
    - PostgreSQL installation and configuration
    - Database schema management
    - Migration scripts
    - Backup and recovery procedures
    - Performance tuning and optimization
- **Estimated Time:** 8-16 hours for database setup and configuration

QA/Testing Specialist (Recommended)

- **Required Skills:**
    - Web application testing
    - API testing
    - Integration testing
    - Browser compatibility testing
- **Estimated Time:** 16-24 hours for comprehensive testing

For Ongoing Maintenance

Full-Stack Developer (Part-time to Full-time)

- **Responsibilities:**
    - Custom feature development
    - Bug fixes and troubleshooting
    - Code maintenance and refactoring
    - Integration with internal systems
    - Dependency updates and security patches
    - Build process maintenance
- **Time Commitment:** 10-40 hours per week (depending on customization needs)

DevOps Engineer (Part-time)

- **Responsibilities:**

    - Deployment pipeline maintenance
    - Infrastructure monitoring and optimization
    - Security updates and patches
    - Scaling and performance optimization
    - Disaster recovery planning and testing
- **Time Commitment:** 5-15 hours per week

Database Administrator (Part-time)

- **Responsibilities:**
    - Database maintenance and optimization
    - Backup verification and recovery testing
    - Schema updates and migrations
    - Performance monitoring and tuning
    - Data integrity and security
- **Time Commitment:** 3-8 hours per week

Deployment Steps (High-Level)

1. Clone all repositories from Bitbucket
2. Install Node.js, NPM, and required dependencies
3. Configure environment variables for each component
4. Set up and configure PostgreSQL database
5. Build each component:
    - Build **tbrunp** using Parcel.js
    - Build **tbwwwp2** using Next.js build commands
    - Build **tbbe** Strapi backend
    - Configure **cdn** static site
    - Configure **filemanager** (if using)
6. Deploy built applications to web servers or containers
7. Configure reverse proxy and SSL certificates
8. Set up CI/CD pipelines for automated builds and deployments
9. Perform comprehensive testing
10. Configure monitoring, logging, and backup systems

Advantages

- **Full customization** - Modify any aspect of the application
- **Complete control** - Own the entire codebase and deployment process
- **Independent updates** - Update on your own schedule
- **Integration flexibility** - Integrate with existing systems and workflows
- **No vendor lock-in** - Deploy and maintain without ongoing dependencies

Considerations

- **Higher technical complexity** - Requires skilled development team
- **Longer deployment time** - Initial setup takes significantly longer
- **Ongoing maintenance burden** - Responsible for all updates and security patches
- **Build expertise required** - Must understand build processes for all components

- **Dependency management** - Must manage NPM packages and security vulnerabilities

---

Deployment Scenario Comparison

Aspect	Scenario 1: Docker	Scenario 2: Source Code
Deployment Time	4-8 hours	24-48 hours
Technical Expertise	DevOps/SysAdmin	Full-Stack Dev + DevOps + DBA
Initial Team Size	1-2 people	2-4 people
Customization	Limited	Full
Maintenance Effort	Low	Medium to High
Update Control	Vendor-managed	Self-managed
Cost (Personnel)	Lower	Higher
Time to Production	Days	Weeks

Recommended Deployment Path

Choose Scenario 1 (Docker) if:

- You need rapid deployment
- You have limited development resources
- Standard functionality meets your needs
- You prefer managed updates
- You want lower ongoing maintenance overhead

Choose Scenario 2 (Source Code) if:

- You require custom features or modifications
- You have in-house development expertise
- You need to integrate with proprietary systems
- You want complete control over the codebase
- You have strict compliance or security requirements that necessitate code review

---

Support and Documentation

Regardless of deployment scenario, Timebars Ltd. provides:

- Comprehensive deployment documentation
- Configuration templates and examples
- Technical support during initial setup
- Access to knowledge base and community forums
- Regular security updates and patches (Scenario 1)
- Migration assistance between scenarios (if needed)

For additional assistance, contact Timebars Ltd. support at [support@timebarssolutions.com].

Document Version: 1.0
Last Updated: October 2025