Tractatus Framework Deployment Guide

Document Type: Technical Documentation

Generated: October 26, 2025

Tractatus AI Safety Framework

https://agenticgovernance.digital

title: Deployment Guide slug: deployment-guide quadrant: OPERATIONAL persistence: HIGH version: 1.0 type: framework author: Tractatus Framework Team created: 2025-10-26 modified: 2025-10-26

Tractatus Framework Deployment Guide

This guide covers deploying the Tractatus governance framework for local development, staging, and production environments.

Prerequisites

System Requirements

- **Node.js**: 18.x or later
- **MongoDB**: 7.0 or later
- **Operating System**: Linux (Ubuntu 22.04+ recommended), macOS, or Windows with WSL2
- **Memory**: Minimum 2GB RAM (4GB+ recommended for production)
- **Storage**: Minimum 10GB available disk space

Required Tools

```
# Verify Node.js installation
node --version  # Should be 18.x or later

# Verify MongoDB installation
mongod --version  # Should be 7.0 or later

# npm is included with Node.js
npm --version
```

Local Development Setup

1. Clone and Install

```
# Clone the repository
git clone https://github.com/AgenticGovernance/tractatus-framework
cd tractatus-framework

# Install dependencies
npm install
```

2. Database Initialization

```
# Start MongoDB (if not running as service)
sudo systemctl start mongod

# Initialize database with default collections
npm run init:db

# This creates:
# - governance_rules collection
# - audit_logs collection
# - instruction_history collection
# - session_state collection
```

3. Environment Configuration

Create a `.env` file in the project root:

```
# Database Configuration
MONGODB_URI=mongodb://localhost:27017/tractatus_dev
MONGODB_DB_NAME=tractatus_dev

# Server Configuration
PORT=9000
NODE_ENV=development

# Session Configuration
SESSION_SECRET=your-random-secret-key-here

# Optional: Anthropic API (for advanced features)
ANTHROPIC_API_KEY=your-api-key-here
```

Security Note: Never commit `.env` files to version control. Use `.env.example` as a template.

4. Start Development Server

```
# Start in development mode with auto-reload
npm run dev

# Or start normally
npm start

# Server runs on http://localhost:9000
```

5. Verify Installation

```
# Run framework tests
npm test

# Expected output: All 238 tests passing

# Check framework services
node scripts/framework-stats.js
```

Access the local application:

- **Main site**: `http://localhost:9000`
- **Admin dashboard**: `http://localhost:9000/admin/audit-analytics.html`
- **Documentation**: `http://localhost:9000/docs.html`

Production Deployment

Server Setup

1. Provision Server

Recommended specifications:

- **CPU**: 2+ cores
- **RAM**: 4GB minimum (8GB recommended)
- **Storage**: 20GB SSD
- **OS**: Ubuntu 22.04 LTS

2. Install Dependencies

```
# Update system packages
sudo apt update && sudo apt upgrade -y

# Install Node.js 18.x
curl -fsSL https://deb.nodesource.com/setup_18.x | sudo -E bash -
sudo apt install -y nodejs

# Install MongoDB 7.0
curl -fsSL https://www.mongodb.org/static/pgp/server-7.0.asc | \
  sudo gpg --dearmor -o /usr/share/keyrings/mongodb-server-7.0.gpg

echo "deb [ arch=amd64,arm64 signed-by=/usr/share/keyrings/mongodb-server-7.0.gp
  https://repo.mongodb.org/apt/ubuntu jammy/mongodb-org/7.0 multiverse" | \
  sudo tee /etc/apt/sources.list.d/mongodb-org-7.0.list

sudo apt update
sudo apt install -y mongodb-org

# Start and enable MongoDB
sudo systemctl start mongod
sudo systemctl enable mongod
```

Production Environment Configuration

Create `/var/www/tractatus/.env`:

```
# Production Database
MONGODB_URI=mongodb://localhost:27017/tractatus_production
MONGODB_DB_NAME=tractatus_production

# Production Server
PORT=9000
NODE_ENV=production

# Security
SESSION_SECRET=generate-strong-random-key-here

# Optional: Production API keys
ANTHROPIC_API_KEY=prod-api-key-here
```

Service Management (systemd)

Create `/etc/systemd/system/tractatus.service`:

```
[Unit]
Description=Tractatus Framework Service
After=network.target mongodb.service
Wants=mongodb.service

[Service]
Type=simple
User=tractatus
WorkingDirectory=/var/www/tractatus
Environment="NODE_ENV=production"
EnvironmentFile=/var/www/tractatus/.env
ExecStart=/usr/bin/node src/server.js
Restart=always
RestartSec=10
StandardOutput=journal
StandardError=journal
SyslogIdentifier=tractatus

[Install]
WantedBy=multi-user.target
```

Enable and start service:

```
# Reload systemd
sudo systemctl daemon-reload

# Enable service to start on boot
sudo systemctl enable tractatus

# Start service
sudo systemctl start tractatus

# Check status
sudo systemctl status tractatus

# View logs
sudo journalctl -u tractatus -f
```

Reverse Proxy (Nginx)

Create `/etc/nginx/sites-available/tractatus`:

```
server {
    listen 80;
    server_name agenticgovernance.digital www.agenticgovernance.digital;

    location / {
        proxy_pass http://localhost:9000;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection 'upgrade';
        proxy_set_header Host $host;
        proxy_cache_bypass $http_upgrade;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
    }

    # Static assets caching
    location ~* \.(js|css|png|jpg|jpeg|gif|svg|ico|woff|woff2|ttf|eot)$ {
        proxy_pass http://localhost:9000;
        expires 1y;
        add_header Cache-Control "public, immutable";
    }
}
```

Enable site:

```
sudo ln -s /etc/nginx/sites-available/tractatus /etc/nginx/sites-enabled/
sudo nginx -t
sudo systemctl reload nginx
```

SSL/TLS Configuration (Let's Encrypt)

```
# Install Certbot
sudo apt install -y certbot python3-certbot-nginx

# Obtain SSL certificate
sudo certbot --nginx -d agenticgovernance.digital -d www.agenticgovernance.digit

# Auto-renewal is configured by default
# Test renewal
sudo certbot renew --dry-run
```

Docker Deployment

Dockerfile

Create `Dockerfile` in project root:

```
FROM node:18-alpine

# Set working directory
WORKDIR /app

# Copy package files
COPY package*.json ./

# Install production dependencies only
RUN npm ci --only=production

# Copy application files
COPY . .

# Create non-root user
RUN addgroup -g 1001 -S tractatus && \
    adduser -S tractatus -u 1001 && \
    chown -R tractatus:tractatus /app

# Switch to non-root user
USER tractatus

# Expose port
EXPOSE 9000

# Health check
HEALTHCHECK --interval=30s --timeout=3s --start-period=40s \
  CMD node -e "require('http').get('http://localhost:9000/health', (res) => proc

# Start application
CMD ["node", "src/server.js"]
```

Docker Compose

Create `docker-compose.yml`:

```yaml
version: '3.8'

services:
  mongodb:
    image: mongo:7.0
    container_name: tractatus-mongodb
    restart: always
    ports:
      - "27017:27017"
    volumes:
      - mongodb-data:/data/db
      - mongodb-config:/data/configdb
    environment:
      MONGO_INITDB_DATABASE: tractatus_production
    networks:
      - tractatus-network

  tractatus:
    build: .
    container_name: tractatus-app
    restart: always
    ports:
      - "9000:9000"
    environment:
      - NODE_ENV=production
      - MONGODB_URI=mongodb://mongodb:27017/tractatus_production
      - PORT=9000
    env_file:
      - .env
    depends_on:
      - mongodb
    networks:
      - tractatus-network
    volumes:
      - ./logs:/app/logs

volumes:
  mongodb-data:
  mongodb-config:
```

```
networks:
  tractatus-network:
    driver: bridge
```

Docker Commands

```
# Build and start containers
docker-compose up -d

# View logs
docker-compose logs -f tractatus

# Stop containers
docker-compose down

# Rebuild after code changes
docker-compose up -d --build

# Execute commands in container
docker-compose exec tractatus npm test

# Database backup
docker-compose exec mongodb mongodump --out=/backup
```

Cloud Deployment Patterns

AWS Deployment

ECS (Elastic Container Service)

```
# task-definition.json
{
  "family": "tractatus",
  "networkMode": "awsvpc",
  "requiresCompatibilities": ["FARGATE"],
  "cpu": "512",
  "memory": "1024",
  "containerDefinitions": [
    {
      "name": "tractatus",
      "image": "your-ecr-repo/tractatus:latest",
      "portMappings": [
        {
          "containerPort": 9000,
          "protocol": "tcp"
        }
      ],
      "environment": [
        {
          "name": "NODE_ENV",
          "value": "production"
        },
        {
          "name": "MONGODB_URI",
          "value": "mongodb://your-documentdb-cluster:27017/tractatus"
        }
      ],
      "logConfiguration": {
        "logDriver": "awslogs",
        "options": {
          "awslogs-group": "/ecs/tractatus",
          "awslogs-region": "us-east-1",
          "awslogs-stream-prefix": "ecs"
        }
      }
    }
  ]
}
```

Google Cloud Platform

Cloud Run Deployment

```
# Build container
gcloud builds submit --tag gcr.io/PROJECT-ID/tractatus

# Deploy to Cloud Run
gcloud run deploy tractatus \
  --image gcr.io/PROJECT-ID/tractatus \
  --platform managed \
  --region us-central1 \
  --allow-unauthenticated \
  --set-env-vars MONGODB_URI=mongodb://mongo-instance:27017/tractatus
```

Kubernetes

Create `k8s/deployment.yaml`:

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: tractatus
spec:
  replicas: 3
  selector:
    matchLabels:
      app: tractatus
  template:
    metadata:
      labels:
        app: tractatus
    spec:
      containers:
      - name: tractatus
        image: tractatus:latest
        ports:
        - containerPort: 9000
        env:
        - name: NODE_ENV
          value: "production"
        - name: MONGODB_URI
          valueFrom:
            secretKeyRef:
              name: tractatus-secrets
              key: mongodb-uri
        resources:
          requests:
            memory: "512Mi"
            cpu: "250m"
          limits:
            memory: "1Gi"
            cpu: "500m"
        livenessProbe:
          httpGet:
            path: /health
            port: 9000
          initialDelaySeconds: 30
          periodSeconds: 10
```

```
        readinessProbe:
          httpGet:
            path: /health
            port: 9000
          initialDelaySeconds: 5
          periodSeconds: 5
---
apiVersion: v1
kind: Service
metadata:
  name: tractatus-service
spec:
  selector:
    app: tractatus
  ports:
  - protocol: TCP
    port: 80
    targetPort: 9000
  type: LoadBalancer
```

Database Management

Backup and Restore

Backup MongoDB:

```
# Local backup
mongodump --db tractatus_production --out /backup/$(date +%Y%m%d)

# Automated daily backup script
#!/bin/bash
BACKUP_DIR="/var/backups/mongodb"
DATE=$(date +%Y%m%d_%H%M%S)
mongodump --db tractatus_production --out $BACKUP_DIR/$DATE
find $BACKUP_DIR -type d -mtime +7 -exec rm -rf {} +
```

Restore MongoDB:

```
mongorestore --db tractatus_production /backup/20251026/tractatus_production
```

Database Migrations

```
# Run migrations
npm run migrate

# Rollback last migration
npm run migrate:rollback
```

Monitoring and Logging

Health Checks

The framework includes a health endpoint:

```
// GET /health
{
  "status": "healthy",
  "uptime": 12345,
  "mongodb": "connected",
  "services": {
    "BoundaryEnforcer": "active",
    "InstructionPersistence": "active",
    "CrossReferenceValidator": "active",
    "ContextPressureMonitor": "active",
    "MetacognitiveVerifier": "active",
    "PluralisticDeliberationOrchestrator": "active"
  }
}
```

Application Logs

```
# View application logs (systemd)
sudo journalctl -u tractatus -f

# View specific time range
sudo journalctl -u tractatus --since "1 hour ago"

# Export logs
sudo journalctl -u tractatus > tractatus-logs.txt
```

Audit Dashboard

Access the audit analytics dashboard at:

```
https://your-domain.com/admin/audit-analytics.html
```

Security Considerations

Environment Variables

- **Never commit** `.env` files to version control
- Use environment-specific configuration files
- Rotate secrets regularly (SESSION_SECRET, API keys)

MongoDB Security

```
# Enable authentication
mongo
> use admin
> db.createUser({
    user: "tractatus_admin",
    pwd: "strong-password-here",
    roles: [{ role: "readWrite", db: "tractatus_production" }]
  })
> exit

# Update .env with authenticated MongoDB URI
# See MongoDB documentation for connection string format with authentication
# Store password in environment variable, never in code
```

Firewall Configuration

```
# Allow only necessary ports
sudo ufw allow 22/tcp   # SSH
sudo ufw allow 80/tcp   # HTTP
sudo ufw allow 443/tcp  # HTTPS
sudo ufw enable

# Block direct access to MongoDB from outside
sudo ufw deny 27017/tcp
```

SSL/TLS Best Practices

- Use Let's Encrypt for free SSL certificates
- Enable HSTS (HTTP Strict Transport Security)
- Configure strong cipher suites in Nginx
- Redirect all HTTP to HTTPS

Troubleshooting

Common Issues

MongoDB connection failed:

```
# Check MongoDB is running
sudo systemctl status mongod

# Check logs
sudo journalctl -u mongod -f

# Verify connection string
mongo $MONGODB_URI
```

Port already in use:

```
# Find process using port 9000
sudo lsof -i :9000

# Kill process
sudo kill -9 <PID>
```

Framework services not initialized:

```
# Run framework health check
node scripts/framework-stats.js

# Check session state
cat .claude/session-state.json
```

Performance Optimization

Database Indexing

```
// Create indexes for frequently queried fields
db.audit_logs.createIndex({ timestamp: -1 });
db.audit_logs.createIndex({ service: 1, timestamp: -1 });
db.governance_rules.createIndex({ id: 1 }, { unique: true });
db.instruction_history.createIndex({ session_id: 1, timestamp: -1 });
```

Caching Strategy

- Static assets served with 1-year cache headers
- API responses cached with appropriate TTL
- MongoDB query result caching for read-heavy operations

Resource Limits

```
# Increase Node.js memory limit
NODE_OPTIONS=--max-old-space-size=4096 node src/server.js
```

Next Steps

- **Implementation Guide**: See Implementation Guide for integration patterns
- **API Reference**: Visit `/api-reference.html` for complete API documentation
- **GitHub Repository**: Access code examples at https://github.com/AgenticGovernance/tractatus-framework

---

Document Version: 1.0 **Last Updated**: 2025-10-26 **Maintained by**: Tractatus Framework Team

---