IMPACT Deployment Guide

Version: 1.6.2

Last Updated: January 2026

Document Status: Production

Table of Contents

System Requirements

Hardware Requirements

Minimum Specifications:

- CPU: 2 cores (4 recommended)
- RAM: 4GB (8GB recommended)
- Storage: 50GB SSD (100GB+ recommended for production)
- Network: 100 Mbps Ethernet

Recommended Production Specifications:

- CPU: 4+ cores (Intel Xeon or AMD EPYC)
- RAM: 16GB+ DDR4
- Storage: 250GB+ NVMe SSD with RAID 10
- Network: 1 Gbps Ethernet with redundancy
- Backup: Separate storage for database backups

Software Requirements

Operating System:

- Ubuntu 22.04 LTS or 24.04 LTS (recommended)
- Debian 11 or 12
- Red Hat Enterprise Linux 8/9
- Other: Any Linux distribution with systemd support

Core Dependencies:

- Python 3.10 or higher (3.13 recommended)
- Node.js 18 LTS or 20 LTS
- MongoDB 6.0 or higher
- Git 2.34 or higher
- systemd (for service management)

Network Requirements:

- Open ports: 3000 (frontend), 8000 (backend), 27017 (MongoDB)
- Outbound HTTPS (443) for API lookups (ICD-10, OPCS-4, NHS ODS)
- Internal network access for user workstations

Pre-Installation Setup

1. Create Installation User

```
# Create impact user (if not using root)
sudo useradd -m -s /bin/bash impact
sudo usermod -aG sudo impact
sudo su - impact
```

2. Update System Packages

```
# Ubuntu/Debian
sudo apt update && sudo apt upgrade -y

# RHEL/Rocky
sudo dnf update -y
```

3. Install Core Dependencies

Ubuntu/Debian

```
# Install Python 3.10+
sudo apt install -y python3 python3-pip python3-venv python3-dev

# Install Node.js 20 LTS
curl -fsSL https://deb.nodesource.com/setup_20.x | sudo -E bash -
sudo apt install -y nodejs

# Install MongoDB 6.0
wget -qO - https://www.mongodb.org/static/pgp/server-6.0.asc | sudo apt-key add -
echo "deb [ arch=amd64,arm64 ] https://repo.mongodb.org/apt/ubuntu jammy/mongodb-org/6.0 multiverse" | sudo tee /etc/apt/sources.list.d/mongodb-org-6.0.list
sudo apt update
sudo apt install -y mongodb-org

# Install Git and other tools
sudo apt install -y git curl wget build-essential
```

RHEL/Rocky Linux

```
# Install Python 3.10+
sudo dnf install -y python3 python3-pip python3-devel

# Install Node.js 20 LTS
curl -fsSL https://rpm.nodesource.com/setup_20.x | sudo bash -
```

```
sudo dnf install -y nodejs


# Install MongoDB 6.0
sudo tee /etc/yum.repos.d/mongodb-org-6.0.repo <<EOF
[mongodb-org-6.0]
name=MongoDB Repository
baseurl=https://repo.mongodb.org/yum/redhat/8/mongodb-org/6.0/x86_64/
gpgcheck=1
enabled=1
gpgkey=https://www.mongodb.org/static/pgp/server-6.0.asc
EOF

sudo dnf install -y mongodb-org

# Install Git and other tools
sudo dnf install -y git curl wget gcc gcc-c++ make
```

4. Start and Enable MongoDB

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

# Verify MongoDB is running
sudo systemctl status mongod
mongosh --eval "db.version()"
```

Installation

1. Clone Repository

```
# Clone to /root/impact (or preferred location)
cd /root
git clone https://github.com/pdsykes2512/impact.git
cd impact
```

2. Backend Setup

```
cd /root/impact

# Create Python virtual environment
python3 -m venv .venv
source .venv/bin/activate

# Install backend dependencies
pip install --upgrade pip
pip install -r backend/requirements.txt
```

3. Frontend Setup

```
cd /root/impact/frontend

# Install Node dependencies
npm install

# Build production assets (optional for development)
npm run build
```

4. Database Initialization

```
cd /root/impact

# Create MongoDB databases and indexes
source .venv/bin/activate
python -m backend.app.database

# Create initial admin user
cd execution
python create_admin_user.py
```

Default Admin Credentials:

- Email: `admin@example.com`
- Password: `admin123`

■■ IMPORTANT: Change these credentials immediately after first login!

Configuration

1. Environment Variables

Create environment file at `/root/impact/.env`:

```
# MongoDB Configuration
MONGODB_URI=mongodb://admin:PASSWORD@localhost:27017/impact?authSource=admin
MONGODB_DB_NAME=impact
MONGODB_SYSTEM_DB_NAME=impact_system

# API Configuration
API_HOST=0.0.0.0
API_PORT=8000
API_TITLE=IMPACT API
API_VERSION=1.6.2

# Security Configuration
SECRET_KEY=GENERATE_SECURE_KEY_HERE_MIN_32_CHARS
ALGORITHM=HS256
ACCESS_TOKEN_EXPIRE_MINUTES=1440

# CORS Configuration
CORS_ORIGINS=["http://localhost:3000","http://impact.vps:3000"]
CORS_ORIGIN_REGEX=r"http://192\.168\.(10|11)\.\d{1,3}:\d+"

# Encryption Configuration
ENCRYPTION_KEY_FILE=/root/.field-encryption-key
ENCRYPTION_SALT_FILE=/root/.field-encryption-salt

# Frontend Configuration (frontend/.env)
VITE_API_URL=http://impact.vps:8000
```

2. Generate Secret Key

```
# Generate secure SECRET_KEY
python3 -c 'import secrets; print(secrets.token_urlsafe(32))'

# Copy output to .env file SECRET_KEY field
```

3. Create Secrets File

For production, store sensitive credentials separately:

```
# Create secrets directory
sudo mkdir -p /etc/impact
sudo chmod 700 /etc/impact

# Create secrets file
sudo tee /etc/impact/secrets.env <<EOF
# MongoDB Credentials
MONGODB_URI=mongodb://admin:YOUR_MONGODB_PASSWORD@localhost:27017/impact?authSource=admin

# Encryption Keys (auto-generated on first run)
ENCRYPTION_KEY_FILE=/root/.field-encryption-key
ENCRYPTION_SALT_FILE=/root/.field-encryption-salt

# API Secret Key
SECRET_KEY=$(python3 -c 'import secrets; print(secrets.token_urlsafe(32))')
EOF

sudo chmod 600 /etc/impact/secrets.env
```

4. MongoDB Authentication Setup

```
# Connect to MongoDB
mongosh

# Create admin user
use admin
db.createUser({
  user: "admin",
  pwd: "YOUR_SECURE_PASSWORD",
  roles: [{ role: "root", db: "admin" }]
})

# Create application user
use impact
db.createUser({
  user: "impact_app",
  pwd: "YOUR_APP_PASSWORD",
  roles: [
    { role: "readWrite", db: "impact" },
    { role: "readWrite", db: "impact_system" }
  ]
})

# Enable authentication
```

```
exit
```

Edit `/etc/mongod.conf`:

```
security:
  authorization: enabled
```

Restart MongoDB:

```
sudo systemctl restart mongod
```

Systemd Services Setup

1. Create Backend Service

Create `/etc/systemd/system/impact-backend.service`:

```
[Unit]
Description=IMPACT Backend API
After=network.target mongodb.service
Wants=mongodb.service

[Service]
Type=simple
User=root
WorkingDirectory=/root/impact
EnvironmentFile=/etc/impact/secrets.env
EnvironmentFile=/root/impact/.env
Environment="PATH=/usr/local/bin:/usr/bin:/bin"
ExecStart=/usr/bin/python3 -m uvicorn backend.app.main:app --host 0.0.0.0 --port 8000 --log-level info
Restart=always
RestartSec=10
StandardOutput=append:/root/.tmp/backend.log
StandardError=append:/root/.tmp/backend.log

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

2. Create Frontend Service

Create `/etc/systemd/system/impact-frontend.service`:

```
[Unit]
Description=IMPACT Frontend
After=network.target

[Service]
Type=simple
User=root
WorkingDirectory=/root/impact/frontend
EnvironmentFile=/etc/impact/secrets.env
EnvironmentFile=/root/impact/.env
Environment="PATH=/usr/local/bin:/usr/bin:/bin"
ExecStart=/usr/bin/npm run dev -- --host 0.0.0.0 --port 3000
Restart=always
RestartSec=10
StandardOutput=append:/root/.tmp/frontend.log
StandardError=append:/root/.tmp/frontend.log

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

3. Create Log Directory

```
mkdir -p /root/.tmp
chmod 755 /root/.tmp
```

4. Enable and Start Services

```
# Reload systemd configuration
sudo systemctl daemon-reload

# Enable services to start on boot
sudo systemctl enable impact-backend
sudo systemctl enable impact-frontend

# Start services
sudo systemctl start impact-backend
sudo systemctl start impact-frontend

# Check service status
sudo systemctl status impact-backend
```

```
sudo systemctl status impact-frontend
```

5. Service Management Commands

```
# Restart services
sudo systemctl restart impact-backend
sudo systemctl restart impact-frontend

# Stop services
sudo systemctl stop impact-backend
sudo systemctl stop impact-frontend

# View logs
journalctl -u impact-backend -f
journalctl -u impact-frontend -f
tail -f /root/.tmp/backend.log
tail -f /root/.tmp/frontend.log
```

Security Hardening

1. Firewall Configuration

```
# Install UFW (Ubuntu)
sudo apt install -y ufw

# Default policies
sudo ufw default deny incoming
sudo ufw default allow outgoing

# Allow SSH
sudo ufw allow 22/tcp

# Allow HTTP/HTTPS
sudo ufw allow 80/tcp
sudo ufw allow 443/tcp

# Allow application ports (internal network only)
sudo ufw allow from 192.168.10.0/24 to any port 3000
```

```
sudo ufw allow from 192.168.10.0/24 to any port 8000


# Enable firewall
sudo ufw enable
sudo ufw status
```

2. MongoDB Security

```
# Bind to localhost only
sudo nano /etc/mongod.conf
```

```
net:
  bindIp: 127.0.0.1
  port: 27017


security:
  authorization: enabled
```

```
# Restart MongoDB
sudo systemctl restart mongod
```

3. File Permissions

```
# Secure environment files
chmod 600 /root/impact/.env
chmod 600 /etc/impact/secrets.env


# Secure encryption keys
chmod 600 /root/.field-encryption-key
chmod 600 /root/.field-encryption-salt


# Application directory permissions
chown -R root:root /root/impact
find /root/impact -type f -exec chmod 644 {} \;
find /root/impact -type d -exec chmod 755 {} \;
```

4. Encryption Key Backup

```
# Backup encryption keys to secure offline storage
sudo cp /root/.field-encryption-key /secure/backup/location/
sudo cp /root/.field-encryption-salt /secure/backup/location/
```

```
# ■■ CRITICAL: Without these keys, encrypted data cannot be recovered!
```

5. SSL/TLS Certificate Setup

For production deployment, configure reverse proxy with SSL:

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

# Create Nginx configuration
sudo nano /etc/nginx/sites-available/impact
```

```
server {
    listen 80;
    server_name impact.yourhospital.nhs.uk;
    return 301 https://$server_name$request_uri;
}

server {
    listen 443 ssl http2;
    server_name impact.yourhospital.nhs.uk;

    ssl_certificate /etc/letsencrypt/live/impact.yourhospital.nhs.uk/fullchain.pem;
    ssl_certificate_key /etc/letsencrypt/live/impact.yourhospital.nhs.uk/privkey.pem;
    ssl_protocols TLSv1.2 TLSv1.3;
    ssl_ciphers HIGH:!aNULL:!MD5;

    # Frontend
    location / {
        proxy_pass http://localhost:3000;
        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;
    }

    # Backend API
    location /api {
        proxy_pass http://localhost:8000;
        proxy_http_version 1.1;
        proxy_set_header Host $host;
        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;
    }
}
```

```
# Enable site
sudo ln -s /etc/nginx/sites-available/impact /etc/nginx/sites-enabled/
sudo nginx -t
sudo systemctl restart nginx

# Obtain SSL certificate
sudo certbot --nginx -d impact.yourhospital.nhs.uk
```

Database Backup Configuration

1. Manual Backup

```
# Create backup directory
mkdir -p /backup/mongodb

# Backup all databases
mongodump --uri="mongodb://admin:PASSWORD@localhost:27017/?authSource=admin" --out=/backup/mongodb/$(date +%Y%m%d_%H%M%S)

# Compress backup
tar -czf /backup/mongodb/backup_$(date +%Y%m%d_%H%M%S).tar.gz /backup/mongodb/$(date +%Y%m%d_%H%M%S)
```

2. Automated Backup Script

Create `/root/impact/execution/backup_database.sh`:

```
#!/bin/bash

BACKUP_DIR="/backup/mongodb"
TIMESTAMP=$(date +%Y%m%d_%H%M%S)
MONGODB_URI="mongodb://admin:PASSWORD@localhost:27017/?authSource=admin"

# Create backup directory
mkdir -p $BACKUP_DIR

# Dump databases
mongodump --uri="$MONGODB_URI" --out=$BACKUP_DIR/$TIMESTAMP

# Compress backup
```

```
tar -czf $BACKUP_DIR/backup_$TIMESTAMP.tar.gz -C $BACKUP_DIR $TIMESTAMP

# Remove uncompressed backup
rm -rf $BACKUP_DIR/$TIMESTAMP

# Keep only last 7 days of backups
find $BACKUP_DIR -name "backup_*.tar.gz" -mtime +7 -delete

echo "Backup completed: $BACKUP_DIR/backup_$TIMESTAMP.tar.gz"
```

```
chmod +x /root/impact/execution/backup_database.sh
```

3. Automated Backup via Cron

```
# Edit crontab
crontab -e

# Add daily backup at 2 AM
0 2 * * * /root/impact/execution/backup_database.sh >> /root/.tmp/backup.log 2>&1
```

4. Using Built-in Backup System

IMPACT includes a web-based backup management system:

1. Navigate to **Admin** → **Backups**
2. Click **"Create Backup"**
3. Backup is created with encryption
4. Download via web interface
5. Automated cleanup of old backups

Monitoring and Logging

1. Application Logs

```
# Backend logs
```

```
tail -f /root/.tmp/backend.log


# Frontend logs
tail -f /root/.tmp/frontend.log


# Systemd journal
journalctl -u impact-backend -f
journalctl -u impact-frontend -f
```

2. MongoDB Logs

```
# View MongoDB logs
tail -f /var/log/mongodb/mongod.log


# Query slow queries
mongosh
use admin
db.setProfilingLevel(1, { slowms: 100 })
db.system.profile.find().limit(5).sort({ ts: -1 }).pretty()
```

3. System Resource Monitoring

```
# Install monitoring tools
sudo apt install -y htop iotop nethogs


# Monitor CPU and memory
htop


# Monitor disk I/O
iotop


# Monitor network
nethogs
```

4. Log Rotation

Create `/etc/logrotate.d/impact`:

```
/root/.tmp/*.log {
    daily
    rotate 30
```

```
    compress
    delaycompress
    missingok
    notifempty
    create 0644 root root
}
```

SSL/TLS Configuration

See Security Hardening section above for SSL/TLS setup with Nginx reverse proxy.

Troubleshooting

Services Won't Start

```
# Check service status
sudo systemctl status impact-backend
sudo systemctl status impact-frontend

# Check logs for errors
journalctl -u impact-backend -n 50
journalctl -u impact-frontend -n 50

# Verify environment files exist
ls -la /etc/impact/secrets.env
ls -la /root/impact/.env

# Test backend manually
cd /root/impact
source .venv/bin/activate
python -m uvicorn backend.app.main:app --host 0.0.0.0 --port 8000
```

MongoDB Connection Errors

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

# Test connection
mongosh --eval "db.version()"

# Check authentication
mongosh -u admin -p --authenticationDatabase admin

# Verify MongoDB URI in .env
cat /etc/impact/secrets.env | grep MONGODB_URI
```

Port Already in Use

```
# Check what's using port 8000
sudo lsof -i :8000

# Check what's using port 3000
sudo lsof -i :3000

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

Permission Denied Errors

```
# Fix ownership
sudo chown -R root:root /root/impact

# Fix permissions
chmod 755 /root/impact
chmod 644 /root/impact/.env
chmod 600 /etc/impact/secrets.env
```

Maintenance

Regular Maintenance Tasks

Daily:

- Check service status
- Monitor disk space
- Review error logs

Weekly:

- Review audit logs
- Check backup completion
- Monitor database size

Monthly:

- Update system packages
- Review user accounts
- Check SSL certificate expiry
- Database performance tuning

Update Procedure

```
# Backup database first
/root/impact/execution/backup_database.sh

# Pull latest code
cd /root/impact
git pull origin main

# Update backend dependencies
source .venv/bin/activate
pip install -r backend/requirements.txt

# Update frontend dependencies
cd frontend
npm install

# Restart services
sudo systemctl restart impact-backend
sudo systemctl restart impact-frontend

# Verify services are running
sudo systemctl status impact-backend
sudo systemctl status impact-frontend
```

Database Maintenance

```
# Compact databases
mongosh
use impact
db.runCommand({ compact: 'patients' })
db.runCommand({ compact: 'episodes' })
db.runCommand({ compact: 'treatments' })

# Rebuild indexes
db.patients.reIndex()
db.episodes.reIndex()
db.treatments.reIndex()

# Check database statistics
db.stats()
```

End of Deployment Guide

For additional documentation, see:

- USER_GUIDE.md
- TECHNICAL_SPECIFICATIONS.md
- SECURITY_AND_COMPLIANCE.md
- DATABASE_SCHEMA.md