# Paramedic Skills Matrix - Deployment Guide

## 🚨 Critical Deployment Fix for "No Content" Issue

The "no content" issue in deployment is primarily caused by **incorrect environment variable configuration**. Follow this guide to fix it.

## 📋 Pre-Deployment Checklist

### 1. Environment Variables (CRITICAL)

The most common cause of deployment failure is incorrect `NEXTAUTH_URL`.

**❌ Wrong (causes "no content"):**

```
NEXTAUTH_URL="http://localhost:3000"
```

**✅ Correct (for deployment):**

```
NEXTAUTH_URL="https://your-actual-deployment-url.com"
```

### 2. Required Environment Variables

Copy `.env.production` to `.env` and update these values:

```
# Database (use your actual database URL)
DATABASE_URL="postgresql://username:password@host:port/database"

# NextAuth (MUST match your deployment URL)
NEXTAUTH_URL="https://your-deployment-url.com"
NEXTAUTH_SECRET="your-secure-random-secret"

# API Keys
ABACUSAI_API_KEY="your-abacusai-api-key"
GEMINI_API_KEY="your-gemini-api-key"
```

## 🚀 Platform-Specific Deployment Instructions

### Vercel Deployment

1. **Set Environment Variables in Vercel Dashboard:**
   - Go to Project Settings → Environment Variables
   - Add all variables from `.env.production`
   - **CRITICAL:** Set `NEXTAUTH_URL` to your Vercel domain (e.g., `https://your-app.vercel.app`)

2. **Deploy Command:**
   `bash`
   `npm run build`

### Netlify Deployment

1. **Set Environment Variables in Netlify Dashboard:**
   - Go to Site Settings → Build & Deploy → Environment Variables
   - **CRITICAL:** Set `NEXTAUTH_URL` to your Netlify domain

### Railway/Render Deployment

1. **Set Environment Variables in Platform Dashboard**
2. **CRITICAL:** Set `NEXTAUTH_URL` to your platform's provided domain

### Docker Deployment

```
# Use the official Node.js image
FROM node:18-alpine

# Set working directory
WORKDIR /app

# Copy package files
COPY package*.json ./
COPY yarn.lock ./

# Install dependencies
RUN yarn install --frozen-lockfile

# Copy source code
COPY . .

# Generate Prisma client
RUN npx prisma generate

# Build the application
RUN npm run build

# Expose port
EXPOSE 3000

# Start the application
CMD ["npm", "start"]
```

## 🛠️ Database Setup

### 1. Prisma Database Setup

```
# Generate Prisma client
npx prisma generate

# Run database migrations
npx prisma db push

# Seed the database (optional)
npx prisma db seed
```

### 2. Database Connection

Ensure your `DATABASE_URL` is accessible from your deployment platform:
- For local development: Use local PostgreSQL
- For production: Use cloud database (PostgreSQL)

## 🔧 Troubleshooting Common Issues

### Issue 1: “No Content” or Blank Page

**Cause:** Incorrect `NEXTAUTH_URL`
**Fix:** Ensure `NEXTAUTH_URL` matches your deployment URL exactly

### Issue 2: Authentication Errors

**Cause:** Missing or incorrect `NEXTAUTH_SECRET`
**Fix:** Generate a secure secret: `openssl rand -base64 32`

### Issue 3: Database Connection Errors

**Cause:** Incorrect `DATABASE_URL` or network issues
**Fix:** Verify database URL and ensure it’s accessible from deployment platform

### Issue 4: API Endpoint Errors

**Cause:** Missing API keys
**Fix:** Ensure `ABACUSAI_API_KEY` and `GEMINI_API_KEY` are set

## 🧪 Testing Deployment Locally

Test your production configuration locally:

```
# Set production environment
export NODE_ENV=production

# Use production environment file
cp .env.production .env

# Update NEXTAUTH_URL to localhost for testing
# NEXTAUTH_URL="http://localhost:3000"

# Build and start
npm run build
npm start
```

## 📊 Performance Optimization

### 1. Enable Compression

The application includes built-in compression for better performance.

### 2. Static File Optimization

Images are set to unoptimized mode for broader deployment compatibility.

### 3. Bundle Analysis

```
# Analyze bundle size
npm run build
```

## 🔒 Security Considerations

### 1. Environment Variables

- Never commit `.env` files to version control
- Use secure, randomly generated secrets
- Rotate API keys regularly

### 2. Database Security

- Use connection pooling
- Enable SSL connections
- Restrict database access by IP

### 3. Application Security

- HTTPS is enforced in production
- Secure headers are configured
- Authentication is required for protected routes

## 📋 Final Deployment Verification

After deployment, verify these work:
- [ ] Homepage loads without errors
- [ ] Sign-in page is accessible
- [ ] Authentication flow works
- [ ] Student and lecturer dashboards load
- [ ] API endpoints respond correctly
- [ ] Database connections are stable

## 🆘 Emergency Recovery

If deployment fails:
1. Check deployment platform logs
2. Verify all environment variables are set
3. Ensure database is accessible
4. Test build locally first
5. Use `.env.example` as reference

## 📞 Support

For deployment issues:
1. Check this guide first
2. Verify environment variable configuration
3. Test locally with production settings
4. Check platform-specific documentation