CareAI Deployment Guide

This guide provides comprehensive instructions for deploying the CareAI application, including frontend, backend services, and database setup.

Overview

CareAI consists of:

- **Frontend**: React/TypeScript application
- **Gemini Proxy Backend**: Node.js/Express API for AI chatbot
- **ML Backend**: Python/FastAPI for machine learning models
- **Database**: Supabase (PostgreSQL)

Prerequisites

- Node.js 18+ and npm/yarn
- Python 3.8+
- Git
- Accounts on deployment platforms (Render, Vercel, Netlify)

1. Database Setup (Supabase)

1.1 Create Supabase Project

1. Go to supabase.com and create an account
2. Create a new project
3. Note down your project URL and anon key

1.2 Run Database Migrations

PROF

```
# Install Supabase CLI
npm install -g supabase

# Login to Supabase
supabase login

# Link your project
supabase link --project-ref YOUR_PROJECT_REF

# Run migrations
supabase db push
```

1.3 Set Up Row Level Security (RLS)

The migrations should automatically set up RLS policies. Verify in the Supabase dashboard.

2. Backend Deployment

2.1 Gemini Proxy Backend

Deploy to Render

1. Create a new Web Service on render.com
2. Connect your GitHub repository
3. Set the following configuration:
   - **Build Command**: `cd backend/gemini-proxy && npm install`
   - **Start Command**: `cd backend/gemini-proxy && npm start`
   - **Environment**: Node

Environment Variables for Gemini Proxy:

```
# Gemini API
GEMINI_API_KEY=eG7WIHusJI5OVDXetHgLLAa8VoboXzU0syw5KqXq

# Supabase
SUPABASE_URL=https://your-project.supabase.co
SUPABASE_ANON_KEY=your-anon-key
SUPABASE_SERVICE_ROLE_KEY=your-service-role-key

# Server Configuration
PORT=3001
NODE_ENV=production

# CORS
ALLOWED_ORIGINS=https://your-frontend-domain.com,https://your-
app.vercel.app

# Rate Limiting
RATE_LIMIT_WINDOW_MS=900000
RATE_LIMIT_MAX_REQUESTS=100
```

PROF

Deploy to Railway (Alternative)

1. Go to railway.app
2. Create new project from GitHub
3. Set root directory to `backend/gemini-proxy`
4. Add the same environment variables

2.2 ML Backend (Python)

Deploy to Render

1. Create a new Web Service on Render
2. Set configuration:

- **Build Command**: `cd backend && pip install -r requirements.txt`
- **Start Command**: `cd backend && uvicorn main:app --host 0.0.0.0 --port $PORT`
- **Environment**: Python 3

Environment Variables for ML Backend:

```
# Server Configuration
PORT=8000
ENVIRONMENT=production

# Model Configuration
MODEL_PATH=/opt/render/project/src/backend/ml/models
ENABLE_GPU=false

# API Configuration
API_TITLE=CareAI ML API
API_VERSION=1.0.0

# CORS
ALLOWED_ORIGINS=https://your-frontend-domain.com

# Logging
LOG_LEVEL=INFO
```

Deploy to Heroku (Alternative)

1. Install Heroku CLI
2. Create new app: `heroku create careai-ml-backend`
3. Set buildpack: `heroku buildpacks:set heroku/python`
4. Add Procfile in backend directory:

PROF

```
web: uvicorn main:app --host 0.0.0.0 --port $PORT
```

3. Frontend Deployment

3.1 Deploy to Vercel (Recommended)

Setup

1. Install Vercel CLI: `npm install -g vercel`
2. Login: `vercel login`
3. In project root: `vercel`

Environment Variables for Frontend:

```
# Supabase
VITE_SUPABASE_URL=https://your-project.supabase.co
VITE_SUPABASE_ANON_KEY=your-anon-key

# Backend APIs
VITE_GEMINI_PROXY_URL=https://your-gemini-proxy.onrender.com
VITE_ML_BACKEND_URL=https://your-ml-backend.onrender.com

# App Configuration
VITE_APP_NAME=CareAI
VITE_APP_VERSION=1.0.0
VITE_ENVIRONMENT=production

# Firebase (if using notifications)
VITE_FIREBASE_API_KEY=your-firebase-api-key
VITE_FIREBASE_AUTH_DOMAIN=your-project.firebaseapp.com
VITE_FIREBASE_PROJECT_ID=your-project-id
VITE_FIREBASE_STORAGE_BUCKET=your-project.appspot.com
VITE_FIREBASE_MESSAGING_SENDER_ID=your-sender-id
VITE_FIREBASE_APP_ID=your-app-id

# Payment Configuration (if using Stripe)
VITE_STRIPE_PUBLISHABLE_KEY=pk_live_your_stripe_key

# Analytics
VITE_GOOGLE_ANALYTICS_ID=G-XXXXXXXXXX
```

Vercel Configuration (vercel.json)

```
{
  "buildCommand": "npm run build",
  "outputDirectory": "dist",
  "framework": "vite",
  "rewrites": [
    {
      "source": "/(.*)",
      "destination": "/index.html"
    }
  ],
  "headers": [
    {
      "source": "/service-worker.js",
      "headers": [
        {
          "key": "Cache-Control",
          "value": "public, max-age=0, must-revalidate"
        }
      ]
    }
```

```
  ]
}
```

3.2 Deploy to Netlify (Alternative)

Setup

1. Connect GitHub repository to Netlify
2. Set build settings:
    - **Build Command**: `npm run build`
    - **Publish Directory**: `dist`

Netlify Configuration (_redirects file)

```
/*    /index.html   200
```

3.3 Deploy to Firebase Hosting (Alternative)

Setup

```
npm install -g firebase-tools
firebase login
firebase init hosting
firebase deploy
```

4. Domain and SSL Setup

4.1 Custom Domain

PROF

1. **Vercel**: Add domain in project settings
2. **Netlify**: Add domain in site settings
3. **Render**: Add custom domain in service settings

4.2 SSL Certificates

All platforms provide automatic SSL certificates for custom domains.

5. Environment-Specific Configurations

5.1 Production Optimizations

Frontend Build Optimizations

```
```

```
# Build with optimizations
npm run build

# Analyze bundle size
npm run analyze
```

Backend Optimizations

- Enable gzip compression
- Set up CDN for static assets
- Configure caching headers
- Enable database connection pooling

5.2 Monitoring and Logging

Frontend Monitoring

- Set up error tracking (Sentry)
- Configure analytics (Google Analytics)
- Monitor Core Web Vitals

Backend Monitoring

- Set up application monitoring (New Relic, DataDog)
- Configure log aggregation
- Set up health checks

6. CI/CD Pipeline

6.1 GitHub Actions Workflow

PROF

Create `.github/workflows/deploy.yml`:

```
name: Deploy CareAI

on:
  push:
    branches: [main]

jobs:
  deploy-frontend:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v3
      - uses: actions/setup-node@v3
        with:
          node-version: '18'
```

```
      - run: npm install
      - run: npm run build
      - uses: amondnet/vercel-action@v20
        with:
          vercel-token: ${{ secrets.VERCEL_TOKEN }}
          vercel-org-id: ${{ secrets.ORG_ID }}
          vercel-project-id: ${{ secrets.PROJECT_ID }}

  deploy-backend:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v3
      - name: Deploy to Render
        run: |
          curl -X POST ${{ secrets.RENDER_DEPLOY_HOOK }}
```

7. Post-Deployment Checklist

7.1 Functionality Testing

- [ ] User registration and login
- [ ] PIN setup and verification
- [ ] Onboarding flow
- [ ] AI chatbot functionality
- [ ] ML predictions
- [ ] Data persistence
- [ ] Real-time features

7.2 Performance Testing

- [ ] Page load times < 3 seconds
- [ ] API response times < 500ms
- [ ] Mobile responsiveness
- [ ] PWA functionality

7.3 Security Testing

- [ ] HTTPS enabled
- [ ] API endpoints secured
- [ ] Database RLS working
- [ ] Input validation
- [ ] XSS protection

8. Maintenance and Updates

8.1 Regular Updates

- Update dependencies monthly
- Monitor security vulnerabilities

- Review and rotate API keys quarterly
- Backup database regularly

8.2 Scaling Considerations

- Monitor resource usage
- Set up auto-scaling for backends
- Consider CDN for global distribution
- Implement database read replicas if needed

9. Troubleshooting

9.1 Common Issues

Build Failures

- Check Node.js version compatibility
- Verify environment variables
- Clear node_modules and reinstall

API Connection Issues

- Verify CORS settings
- Check API endpoint URLs
- Validate SSL certificates

Database Connection Issues

- Verify Supabase credentials
- Check RLS policies
- Monitor connection limits

9.2 Support Resources

PROF

- Vercel Documentation
- Render Documentation
- Supabase Documentation

10. Cost Optimization

10.1 Free Tier Limits

- **Vercel**: 100GB bandwidth/month
- **Render**: 750 hours/month free
- **Supabase**: 500MB database, 2GB bandwidth

10.2 Scaling Costs

- Monitor usage and upgrade plans as needed

- Consider serverless functions for cost efficiency
- Implement caching to reduce API calls

11. Connecting Deployed Services

11.1 Update Frontend Configuration

After deploying your backends, update the frontend environment variables:

```
# Update these with your actual deployed URLs
VITE_GEMINI_PROXY_URL=https://careai-gemini-proxy.onrender.com
VITE_ML_BACKEND_URL=https://careai-ml-backend.onrender.com
```

11.2 Backend URL Configuration

In your frontend code, the APIs are configured in:

- `src/lib/supabaseClient.ts` - Database connection
- `src/services/chatService.ts` - Gemini proxy connection
- `src/services/mlService.ts` - ML backend connection

11.3 CORS Configuration

Ensure your backend services allow requests from your frontend domain:

Gemini Proxy (backend/gemini-proxy/server.js):

```
const allowedOrigins = [
  'https://your-app.vercel.app',
  'https://your-custom-domain.com'
];
```

PROF

ML Backend (backend/main.py):

```
origins = [
    "https://your-app.vercel.app",
    "https://your-custom-domain.com"
]
```

12. Database Migration Commands

12.1 Apply All Migrations

```
```

```
# From project root
supabase db push

# Or apply specific migration
supabase db push --include-all
```

12.2 Reset Database (if needed)

```
supabase db reset
```

12.3 Generate Types (optional)

```
supabase gen types typescript --project-id YOUR_PROJECT_ID >
src/types/database.types.ts
```

13. Quick Deployment Commands

13.1 One-Command Deployment

Create a deployment script `deploy.sh`:

```
#!/bin/bash

echo "🚀 Deploying CareAI..."

# Build frontend
echo "📦 Building frontend..."
npm run build

# Deploy to Vercel
echo "🌐 Deploying frontend to Vercel..."
vercel --prod

# Trigger backend deployments (if using webhooks)
echo "⚙️ Triggering backend deployments..."
curl -X POST $RENDER_GEMINI_DEPLOY_HOOK
curl -X POST $RENDER_ML_DEPLOY_HOOK

echo "✅ Deployment complete!"
```

13.2 Environment Setup Script

Create `setup-env.sh`:

```
#!/bin/bash

echo "Setting up environment variables..."

# Copy example env files
cp .env.example .env.local

echo "Please update .env.local with your actual values:"
echo "- VITE_SUPABASE_URL"
echo "- VITE_SUPABASE_ANON_KEY"
echo "- VITE_GEMINI_PROXY_URL"
echo "- VITE_ML_BACKEND_URL"
```

Support

For additional support or questions:

- Check the project documentation
- Review deployment logs on your hosting platforms
- Contact the development team
- Use the debugging tools: `window.ThemeDebug` and `window.ReloadDebug` in browser console