AI Assistant Widget Template & Deployment Guide

Created: December 26, 2025
Status: Template Ready for Client Deployment
Deployment Time: 48 hours (with template)

🎯 Overview

This guide documents how to deploy the AI Assistant Widget as a standardized, repeatable product for clients. The template allows for 48-hour deployments with minimal customization.

📦 What's Included in the Template

Core Components (100% Reusable)

1. **Widget Component** (`components/AIAssistantWidget.tsx`)
   - Floating chat UI
   - Message handling
   - Streaming responses
   - WCAG 2.1 AA accessible
   - Link parsing & text overflow handling
2. **API Route** (`app/api/ai-assistant/route.ts`)
   - Rate limiting (10 req/min)
   - Error handling
   - Conversation storage
   - Streaming support
3. **Claude Integration** (`lib/claude.ts`)
   - Streaming responses
   - Error handling
   - Model: `claude-3-haiku-20240307`
4. **Database Schema** (`supabase/migrations/20250125_create_ai_assistant_tables.sql`)
   - Conversations table
   - Messages table
   - Indexes & RLS policies
5. **Rate Limiting** (`lib/rate-limit.ts`)
   - In-memory rate limiting
   - IP-based tracking

Customization Points (Per Client)

1. **Knowledge Base** (`lib/knowledge-base.ts`)
   - Services list
   - FAQs
   - Website links
   - System prompt
2. **Branding** (Widget component)
   - Colors (primary, accent)
   - Logo/icon

    - Widget position

3. **Environment Variables**
    - `ANTHROPIC_API_KEY` (client-specific or shared)
    - Supabase credentials

🏗️ Template Architecture

```
ai-assistant-template/
├── components/
│   └── AIAssistantWidget.tsx          # Widget UI (reusable)
├── app/
│   └── api/
│       └── ai-assistant/
│           └── route.ts               # API endpoint (reusable)
├── lib/
│   ├── claude.ts                      # Claude integration (reusable)
│   ├── knowledge-base.ts              # ⚠️ CUSTOMIZE PER CLIENT
│   └── rate-limit.ts                  # Rate limiting (reusable)
├── types/
│   └── ai-assistant.ts                # TypeScript types (reusable)
└── supabase/
    └── migrations/
        └── 20250125_create_ai_assistant_tables.sql  # Database (reusable)
```

🔧 Standardized Stack

Technology Stack (Fixed)

- **Frontend:** Next.js 16 (App Router), React 19, TypeScript
- **AI Model:** Claude 3 Haiku (`claude-3-haiku-20240307`)
- **Database:** Supabase (PostgreSQL)
- **Styling:** TailwindCSS
- **Deployment:** Vercel (recommended)

Why This Stack?

- **Next.js 16:** Server-side API routes, streaming support
- **Claude Haiku:** Cost-effective ($0.25/$1.25 per 1M tokens), fast responses
- **Supabase:** Managed PostgreSQL, RLS security, easy setup
- **Vercel:** Zero-config deployment, edge functions

📋 Deployment Checklist (48-Hour Template)

Phase 1: Setup (2 hours)

- [ ] Clone template repository
- [ ] Create Supabase project
- [ ] Run database migrations
- [ ] Set up Vercel project
- [ ] Configure environment variables

Phase 2: Customization (4 hours)

- [ ] Update knowledge base (services, FAQs)
- [ ] Customize system prompt
- [ ] Update website links
- [ ] Configure branding (colors, logo)
- [ ] Test knowledge base accuracy

Phase 3: Integration (2 hours)

- [ ] Add widget to client's layout
- [ ] Configure widget position/styling
- [ ] Test widget on all pages
- [ ] Verify accessibility (WCAG 2.1 AA)

Phase 4: Testing & Launch (2 hours)

- [ ] Test conversation flow
- [ ] Verify rate limiting
- [ ] Test error handling
- [ ] Check mobile responsiveness
- [ ] Deploy to production

Total Time: ~10 hours (can be done in 48 hours with buffer)

🎨 Customization Guide

1. Knowledge Base Configuration

File: `lib/knowledge-base.ts`

What to Customize:

```
// Services List
export const RWS_SERVICES: KnowledgeBaseService[] = [
  {
    name: 'Your Service Name',
    description: 'Service description',
    pricing: '$X,XXX - $X,XXX',
    timeline: 'X-X weeks',
    technologies: ['Tech1', 'Tech2']
  },
  // ... more services
];

// FAQs
export const RWS_FAQ: KnowledgeBaseFAQ[] = [
  {
    question: 'Your FAQ question?',
    answer: 'Your FAQ answer.',
    category: 'Category'
  },
  // ... more FAQs
];

// Website Links
export const WEBSITE_LINKS = {
  baseUrl: 'https://yourclient.com',
  startProject: '/your-questionnaire',
  bookConsultation: '/your-booking',
  services: {
    service1: '/services/service1',
    // ... more service links
  }
};
```

System Prompt Customization:

```
export function formatSystemPrompt(): string {
  return `You are an AI assistant for [CLIENT NAME], a [BUSINESS TYPE] based in [LOCATION].

Your role is to help potential clients understand our services and qualify leads.

SERVICES:
${servicesList}

FREQUENTLY ASKED QUESTIONS:
${faqList}

// ... rest of prompt
`;
}
```

2. Widget Branding

File: `components/AIAssistantWidget.tsx`

Customization Points:

- Colors: Update Tailwind classes or add custom CSS
- Logo: Replace icon or add image
- Position: Modify `fixed` positioning classes
- Size: Adjust `max-w-md` and height classes

Example:

```
// Change primary color
className="bg-primary" // Update in globals.css or use inline styles

// Change widget position
className="fixed bottom-4 right-4" // Change to left-4 for left side

// Add logo
<img src="/client-logo.png" alt="Client Logo" className="w-8 h-8" />
```

3. Environment Variables

Required:

```
ANTHROPIC_API_KEY=sk-ant-...           # Claude API key
NEXT_PUBLIC_SUPABASE_URL=https://...  # Supabase project URL
NEXT_PUBLIC_SUPABASE_ANON_KEY=...     # Supabase anon key
SUPABASE_SERVICE_ROLE_KEY=...         # Supabase service role key
```

Optional:

```
SENTRY_DSN=...                      # Error monitoring
NEXT_PUBLIC_URL=https://...         # Base URL for links
```

🚀 Deployment Workflow

Step 1: Template Setup

```
# Clone template (or copy from existing project)
git clone [template-repo] client-ai-assistant
cd client-ai-assistant

# Install dependencies
npm install

# Copy environment template
cp .env.example .env.local
```

Step 2: Supabase Setup

```
# Create new Supabase project
# Run migrations
supabase db push

# Or manually run:
# supabase/migrations/20250125_create_ai_assistant_tables.sql
```

Step 3: Customize Knowledge Base

```
# Edit lib/knowledge-base.ts
# Update services, FAQs, links, system prompt
```

Step 4: Configure Widget

```
# Edit components/AIAssistantWidget.tsx
# Update branding, colors, position
```

Step 5: Deploy

```
# Deploy to Vercel
vercel deploy --prod

# Or use Git integration
git push origin main
```

💰 Pricing Model (Template-Based)

One-Time Setup Fee

- **Template Deployment:** A$3,500–A$5,000
- **Includes:** Widget setup, knowledge base configuration, testing, deployment

Optional Add-Ons

- **Custom Branding:** A$500–A$1,000
- **Advanced Knowledge Base:** A$500–A$1,500 (complex FAQs/services)
- **Multi-Language Support:** A$1,000–A$2,000
- **Custom Integrations:** A$1,500–A$3,000 (CRM, email, etc.)

Ongoing Costs (Client Pays)

- **Claude API:** ~A$10–A$50/month (depending on usage)
- **Supabase:** Free tier or A$25/month (Pro)
- **Vercel:** Free tier or A$20/month (Pro)

Total Monthly Cost: A$10–A$95/month (client pays directly)

📊 Template Benefits

For RWS (Agency)

- **71–92% Margins:** Template reusable, minimal customization
- **48-Hour Deployment:** Standardized workflow
- **Scalable:** Can deploy to multiple clients
- **Maintainable:** Single codebase, easy updates

For Clients

- **Custom AI:** Not generic chatbot templates
- **Full Ownership:** Data stored in their Supabase
- **Unlimited Scalability:** No platform limits
- **Cost-Effective:** One-time fee vs. monthly subscriptions

🔄 Maintenance & Updates

Template Updates

- Update template repository
- Pull updates to client projects
- Test compatibility
- Deploy updates

Client-Specific Updates

- Knowledge base changes (new services, FAQs)
- Branding updates
- Feature additions

📝 Client Handoff Checklist

- [ ] Knowledge base documentation
- [ ] How to update FAQs/services
- [ ] How to view conversations in Supabase
- [ ] Claude API key management
- [ ] Supabase access credentials

- [ ] Vercel deployment access
- [ ] Support contact information

🎯 Success Metrics

Deployment Metrics

- **Time to Deploy:** Target 48 hours
- **Customization Time:** 4–6 hours
- **Testing Time:** 2 hours

Performance Metrics

- **Response Time:** < 2 seconds average
- **Uptime:** 99.9% (Vercel SLA)
- **Accessibility:** WCAG 2.1 AA compliant

Business Metrics

- **Lead Qualification:** Track via Supabase
- **Conversion Rate:** Measure questionnaire submissions
- **Cost per Conversation:** Monitor Claude API usage

🚨 Common Issues & Solutions

Issue: Claude API Rate Limits

Solution: Rate limiting already implemented (10 req/min). Can increase if needed.

Issue: Knowledge Base Too Large

Solution: Optimize system prompt, use RAG (Retrieval Augmented Generation) for large knowledge bases.

Issue: Client Wants Different AI Model

Solution: Update `lib/claude.ts` model parameter. Note: Sonnet requires upgraded plan.

Issue: Multi-Tenant Setup

Solution: Add `client_id` to database schema, filter by client in API route.

🔮 Future Enhancements

Phase 2 Features

- [ ] Admin dashboard for conversation management
- [ ] Analytics dashboard (conversations, leads, metrics)
- [ ] A/B testing for system prompts
- [ ] Multi-language support
- [ ] Voice input/output
- [ ] Custom integrations (CRM, email, SMS)

Phase 3 Features

- [ ] White-label solution
- [ ] Self-service portal
- [ ] Template marketplace

- [ ] API for third-party integrations

📞 Support & Resources

Documentation

- [Claude API Docs]
- [Supabase Docs]
- [Next.js Docs]

Internal Resources

- Template repository: [link]
- Deployment checklist: [link]
- Knowledge base examples: [link]

✅ Template Readiness Checklist

- [x] Widget component reusable
- [x] API route reusable
- [x] Database schema reusable
- [x] Knowledge base customizable
- [x] Branding customizable
- [x] Documentation complete
- [x] Deployment workflow documented
- [x] Pricing model defined

Status: ✅ Template Ready for Client Deployment

Last Updated: December 26, 2025
Template Version: 1.0
Next Review: After first 3 client deployments