**Elder Care Navigator - Development Roadmap**

**Project Overview**

**Product Name:** Elder Care Navigator  
**Mission:** Help adult children caring for aging parents with dementia from a distance by providing personalized, curated resources from authoritative sources.

**Core Value Proposition:** Intelligent resource matching that understands user context and surfaces the right resources from thousands of existing authoritative sources (NIH, Alzheimer's Association, Mayo Clinic, government programs, etc.)

**MVP Focus:** Adult children managing a parent with early-stage dementia from 100+ miles away

**Tech Stack**

**Frontend**

* **Framework:** Next.js 14 (App Router)
* **Language:** TypeScript
* **Styling:** Tailwind CSS
* **UI Components:** shadcn/ui
* **Forms:** React Hook Form + Zod validation
* **State Management:** React Context (minimal - most state in URL/database)

**Backend**

* **Database:** Supabase (PostgreSQL)
* **API Routes:** Next.js API routes
* **AI Integration:** Anthropic Claude API (claude-sonnet-4-20250514)
* **Authentication:** Supabase Auth (Phase 2 - not MVP)

**Services**

* **Email:** Resend or ConvertKit API
* **Analytics:** Vercel Analytics
* **Deployment:** Vercel
* **File Storage:** Supabase Storage (for PDFs)

**Development Tools**

* **Version Control:** Git + GitHub
* **Package Manager:** npm or pnpm
* **Linting:** ESLint + Prettier
* **Type Checking:** TypeScript strict mode

**Database Schema**

**Tables**

**resources**

CREATE TABLE resources (

id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

created\_at TIMESTAMP WITH TIME ZONE DEFAULT NOW(),

updated\_at TIMESTAMP WITH TIME ZONE DEFAULT NOW(),

-- Core Info

title TEXT NOT NULL,

url TEXT NOT NULL,

description TEXT NOT NULL,

best\_for TEXT, -- One sentence: "Best for [specific situation]"

-- Categorization

category TEXT[] NOT NULL, -- ['crisis', 'legal', 'financial', 'education', 'support', 'products', 'local']

conditions TEXT[] NOT NULL, -- ['dementia', 'mobility', 'chronic', 'mental\_health', 'multiple']

urgency\_level TEXT NOT NULL, -- 'immediate', 'this\_week', 'ongoing'

-- Location

location\_type TEXT NOT NULL, -- 'national', 'state\_specific', 'local'

states TEXT[], -- NULL if national, ['AZ', 'CA'] if state-specific

requires\_zip BOOLEAN DEFAULT FALSE,

-- Audience

audience TEXT[] NOT NULL, -- ['family\_caregiver', 'professional', 'patient']

living\_situation TEXT[], -- ['alone', 'with\_family', 'facility', 'long\_distance']

-- Resource Details

cost TEXT NOT NULL, -- 'free', 'paid', 'insurance', 'sliding\_scale'

contact\_phone TEXT,

contact\_email TEXT,

hours\_available TEXT, -- '24/7', 'business\_hours', 'varies'

-- Monetization

affiliate\_url TEXT, -- If different from main URL

affiliate\_network TEXT, -- 'amazon', 'cj', 'shareasale', null

is\_sponsored BOOLEAN DEFAULT FALSE,

-- Metadata

source\_authority TEXT NOT NULL, -- 'nih', 'alzheimers\_assoc', 'mayo\_clinic', 'government'

last\_verified TIMESTAMP WITH TIME ZONE,

click\_count INTEGER DEFAULT 0,

upvotes INTEGER DEFAULT 0,

-- Search

search\_vector tsvector GENERATED ALWAYS AS (

to\_tsvector('english', title || ' ' || description || ' ' || COALESCE(best\_for, ''))

) STORED

);

-- Indexes

CREATE INDEX idx\_resources\_category ON resources USING GIN(category);

CREATE INDEX idx\_resources\_conditions ON resources USING GIN(conditions);

CREATE INDEX idx\_resources\_urgency ON resources(urgency\_level);

CREATE INDEX idx\_resources\_location ON resources(location\_type);

CREATE INDEX idx\_resources\_search ON resources USING GIN(search\_vector);

**user\_sessions (for tracking, not auth)**

CREATE TABLE user\_sessions (

id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

created\_at TIMESTAMP WITH TIME ZONE DEFAULT NOW(),

-- User Context (from navigator)

relationship TEXT, -- 'adult\_child', 'spouse', 'other\_family', 'friend'

conditions TEXT[] NOT NULL,

zip\_code TEXT,

city TEXT,

state TEXT,

living\_situation TEXT NOT NULL,

urgency\_factors TEXT[] NOT NULL,

-- Contact (optional)

email TEXT,

email\_subscribed BOOLEAN DEFAULT FALSE,

-- Generated Results

matched\_resources UUID[], -- Array of resource IDs

ai\_guidance TEXT, -- Personalized text from Claude

-- Tracking

completed\_at TIMESTAMP WITH TIME ZONE,

resources\_clicked UUID[],

pdf\_downloaded BOOLEAN DEFAULT FALSE,

leads\_submitted INTEGER DEFAULT 0

);

**leads (for service providers)**

CREATE TABLE leads (

id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

created\_at TIMESTAMP WITH TIME ZONE DEFAULT NOW(),

session\_id UUID REFERENCES user\_sessions(id),

lead\_type TEXT NOT NULL, -- 'home\_care', 'elder\_law', 'memory\_care', 'financial\_planning'

-- Contact Info

name TEXT NOT NULL,

email TEXT NOT NULL,

phone TEXT,

-- Context

zip\_code TEXT NOT NULL,

message TEXT,

urgency TEXT, -- 'immediate', 'within\_week', 'researching'

-- Tracking

sent\_to\_provider TEXT, -- Which provider received this

provider\_notified\_at TIMESTAMP WITH TIME ZONE,

status TEXT DEFAULT 'new' -- 'new', 'sent', 'contacted', 'converted'

);

**resource\_feedback (community validation)**

CREATE TABLE resource\_feedback (

id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

created\_at TIMESTAMP WITH TIME ZONE DEFAULT NOW(),

resource\_id UUID REFERENCES resources(id) ON DELETE CASCADE,

session\_id UUID REFERENCES user\_sessions(id),

helpful BOOLEAN NOT NULL,

comment TEXT,

-- Prevent spam

ip\_hash TEXT,

UNIQUE(resource\_id, ip\_hash)

);

**Application Structure**

elder-care-navigator/

├── app/

│ ├── layout.tsx # Root layout with nav, footer

│ ├── page.tsx # Homepage

│ ├── navigator/

│ │ ├── page.tsx # Multi-step navigator form

│ │ └── components/

│ │ ├── StepIndicator.tsx

│ │ ├── RelationshipStep.tsx

│ │ ├── ConditionsStep.tsx

│ │ ├── LocationStep.tsx

│ │ └── UrgencyStep.tsx

│ ├── results/

│ │ ├── [sessionId]/

│ │ │ └── page.tsx # Results page (dynamic)

│ │ └── components/

│ │ ├── ImmediateActions.tsx

│ │ ├── ResourceCard.tsx

│ │ ├── ResourceCategory.tsx

│ │ └── SaveResults.tsx

│ ├── resources/

│ │ └── page.tsx # Browse all resources

│ ├── crisis/

│ │ └── page.tsx # Crisis resources page

│ ├── about/

│ │ └── page.tsx

│ ├── api/

│ │ ├── match-resources/

│ │ │ └── route.ts # Resource matching logic

│ │ ├── generate-guidance/

│ │ │ └── route.ts # Claude API integration

│ │ ├── submit-session/

│ │ │ └── route.ts # Save user session

│ │ ├── generate-pdf/

│ │ │ └── route.ts # Generate PDF of results

│ │ ├── send-email/

│ │ │ └── route.ts # Email results

│ │ ├── submit-lead/

│ │ │ └── route.ts # Service provider leads

│ │ └── feedback/

│ │ └── route.ts # Resource feedback

├── components/

│ ├── ui/ # shadcn/ui components

│ ├── Header.tsx

│ ├── Footer.tsx

│ ├── CrisisButton.tsx # Always-visible crisis access

│ └── EmailCapture.tsx

├── lib/

│ ├── supabase.ts # Supabase client

│ ├── anthropic.ts # Claude API client

│ ├── matching-algorithm.ts # Resource matching logic

│ ├── email.ts # Email sending

│ └── utils.ts

├── types/

│ └── index.ts # TypeScript types

├── public/

│ └── images/

└── prompts/

└── resource-guidance.txt # Claude system prompt

**Week-by-Week Build Plan**

**Week 1: Foundation & Setup (Days 1-7)**

**Day 1-2: Project Setup**

* [ ] Initialize Next.js 14 project with TypeScript
* [ ] Set up Tailwind CSS
* [ ] Install and configure shadcn/ui
* [ ] Set up Supabase project and get credentials
* [ ] Configure environment variables (.env.local)
* [ ] Set up Git repository
* [ ] Create basic layout with header/footer
* [ ] Deploy to Vercel (empty site, establish pipeline)

**Day 3-4: Database Setup**

* [ ] Create Supabase tables using schema above
* [ ] Set up Row Level Security (RLS) policies
* [ ] Create indexes for performance
* [ ] Seed initial 50 resources (manually curated)
* [ ] Create Supabase client utilities
* [ ] Test database connections

**Day 5-7: Homepage & Basic Routing**

* [ ] Build homepage with clear CTA
* [ ] Create basic header navigation
* [ ] Create footer with disclaimers
* [ ] Build crisis resources page (static content)
* [ ] Build about page
* [ ] Set up basic responsive design
* [ ] Add Google Analytics

**Deliverable:** Deployed site with homepage, database ready, can navigate between pages

**Week 2: Navigator Form (Days 8-14)**

**Day 8-10: Multi-Step Form**

* [ ] Create navigator page structure
* [ ] Build step indicator component
* [ ] Implement form state management
* [ ] Create Step 1: Relationship selection
* [ ] Create Step 2: Conditions checkboxes
* [ ] Create Step 3: Location input (zip code)
* [ ] Create Step 4: Living situation
* [ ] Create Step 5: Urgency assessment
* [ ] Add form validation (Zod schemas)
* [ ] Add progress saving to localStorage
* [ ] Mobile-responsive form design

**Day 11-12: Resource Matching**

* [ ] Build matching algorithm in /lib/matching-algorithm.ts
* [ ] Create API route /api/match-resources
* [ ] Implement tag-based filtering
* [ ] Implement location-based filtering
* [ ] Implement urgency-based prioritization
* [ ] Add scoring system for relevance
* [ ] Test matching with various inputs

**Day 13-14: Claude Integration**

* [ ] Set up Anthropic API client
* [ ] Create system prompt for personalized guidance
* [ ] Build API route /api/generate-guidance
* [ ] Integrate matched resources into Claude prompt
* [ ] Handle API errors gracefully
* [ ] Add loading states
* [ ] Test with various user scenarios

**Deliverable:** Working navigator that accepts input and generates matched resources

**Week 3: Results & Core Features (Days 15-21)**

**Day 15-16: Results Page**

* [ ] Create dynamic results page /results/[sessionId]
* [ ] Save user session to database
* [ ] Display "Immediate Actions" section
* [ ] Build ResourceCard component
* [ ] Group resources by category
* [ ] Add "Best For" explanations for each resource
* [ ] Implement resource click tracking
* [ ] Add external link warnings

**Day 17-18: Email & PDF Generation**

* [ ] Set up Resend or ConvertKit API
* [ ] Create email capture component
* [ ] Build email template for results
* [ ] Implement PDF generation (simple approach: pre-rendered HTML)
* [ ] Create API route /api/send-email
* [ ] Add "Email Me These Results" button
* [ ] Set up email list (for future nurture)

**Day 19-21: Lead Generation**

* [ ] Create lead capture forms for services
* [ ] Build "Connect with Home Care" flow
* [ ] Build "Find Elder Law Attorney" flow
* [ ] Create API route /api/submit-lead
* [ ] Store leads in database
* [ ] Set up notification emails (to you/providers)
* [ ] Add lead tracking

**Deliverable:** Complete user flow from navigator to results to email/leads

**Week 4: Polish & Launch (Days 22-28)**

**Day 22-24: UI/UX Polish**

* [ ] Design system consistency check
* [ ] Improve mobile responsiveness
* [ ] Add loading skeletons
* [ ] Improve error states
* [ ] Add success confirmations
* [ ] Optimize images
* [ ] Add micro-interactions
* [ ] Accessibility audit (keyboard nav, ARIA labels)

**Day 25-26: Content & SEO**

* [ ] Write compelling homepage copy
* [ ] Add meta tags and descriptions
* [ ] Create sitemap.xml
* [ ] Set up robots.txt
* [ ] Add Open Graph images
* [ ] Write legal disclaimers
* [ ] Create privacy policy
* [ ] Create terms of service

**Day 27-28: Testing & Launch**

* [ ] Test all user flows end-to-end
* [ ] Test on multiple devices/browsers
* [ ] Test form validation edge cases
* [ ] Test API error handling
* [ ] Performance optimization (Lighthouse audit)
* [ ] Set up error monitoring (Sentry)
* [ ] Soft launch to 10-20 test users
* [ ] Fix critical bugs
* [ ] Public launch

**Deliverable:** Production-ready MVP

**Key Features Detailed**

**1. Navigator Form**

**User Flow:**

1. User clicks "Get My Personalized Guide" on homepage
2. Multi-step form collects:
   * Relationship (radio buttons)
   * Conditions (checkboxes, multi-select)
   * Location (zip code input with autocomplete for city/state)
   * Living situation (radio buttons)
   * Urgency factors (checkboxes)
3. Form validates each step before proceeding
4. Progress bar shows completion
5. Can go back to edit previous answers
6. "Get My Resources" submits form

**Technical Requirements:**

* Client-side validation (immediate feedback)
* Server-side validation (security)
* Form state persists in URL params (shareable links)
* Mobile-optimized (large tap targets)
* Accessible (keyboard navigation, screen readers)

**2. Resource Matching Algorithm**

**Logic:**

function matchResources(userContext: UserContext): Resource[] {

// 1. Filter by conditions

let matches = resources.filter(r =>

r.conditions.some(c => userContext.conditions.includes(c))

);

// 2. Filter by location

matches = matches.filter(r => {

if (r.location\_type === 'national') return true;

if (r.location\_type === 'state\_specific') {

return r.states.includes(userContext.state);

}

if (r.location\_type === 'local') {

return r.requires\_zip && isNearZip(r, userContext.zip\_code);

}

});

// 3. Filter by living situation

matches = matches.filter(r =>

!r.living\_situation.length ||

r.living\_situation.includes(userContext.living\_situation)

);

// 4. Score by relevance

matches = matches.map(r => ({

...r,

score: calculateRelevanceScore(r, userContext)

}));

// 5. Sort by urgency level + score

matches.sort((a, b) => {

const urgencyOrder = ['immediate', 'this\_week', 'ongoing'];

const urgencyDiff = urgencyOrder.indexOf(a.urgency\_level) -

urgencyOrder.indexOf(b.urgency\_level);

if (urgencyDiff !== 0) return urgencyDiff;

return b.score - a.score;

});

return matches.slice(0, 50); // Top 50 results

}

**Relevance Scoring Factors:**

* Exact condition match: +10 points
* Urgency alignment: +5 points
* Location specificity (local > state > national): +3 points
* User click history (popular resources): +2 points
* Community upvotes: +1 per 10 upvotes
* Source authority: +2 for government/major orgs

**3. AI Guidance Generation**

**Claude System Prompt:**

You are a compassionate assistant helping someone care for an aging parent.

User context:

- Relationship: {relationship}

- Conditions: {conditions}

- Location: {city}, {state}

- Living situation: {living\_situation}

- Urgency: {urgency\_factors}

Matched resources:

{formatted\_resource\_list}

Task: Write a brief (150-200 words), warm, actionable introduction to their personalized resource list.

1. Start with empathy - acknowledge their specific situation

2. Highlight the 1-2 most urgent resources they should act on first

3. Explain WHY these matter for their situation specifically

4. Keep tone conversational, not clinical

5. End with encouragement

Do not:

- Give medical advice

- Make promises about outcomes

- Overwhelm with too many options

- Use jargon without explanation

**API Implementation:**

// /app/api/generate-guidance/route.ts

export async function POST(request: Request) {

const { userContext, matchedResources } = await request.json();

const prompt = buildPrompt(userContext, matchedResources);

const response = await anthropic.messages.create({

model: "claude-sonnet-4-20250514",

max\_tokens: 500,

messages: [{

role: "user",

content: prompt

}]

});

return Response.json({

guidance: response.content[0].text

});

}

**4. Results Page Layout**

**Structure:**

[Personalized greeting]

"Here's what we found for your situation..."

[AI-Generated Guidance Section]

Warm, personalized 2-3 paragraphs

[Immediate Actions - Top 3]

1. Most urgent resource

- What it is

- Why it matters NOW

- Big button: "Get Help"

2. Second priority

3. Third priority

[Email Capture]

"Save these resources - we'll email you this list"

[Comprehensive Resources by Category]

📚 Understanding Dementia

- Resource cards with descriptions

📍 Local Help in [City]

- Resource cards

💼 Financial & Legal

- Resource cards

❤️ Support For You

- Resource cards

🛒 Helpful Products

- Resource cards with affiliate links (clearly labeled)

[Service Provider CTAs]

"Need professional help? Connect with vetted providers"

- Find Home Care Agencies

- Find Elder Law Attorneys

- Find Memory Care Facilities

[Feedback]

"Was this helpful?" [Yes] [No]

**5. Resource Card Component**

**Design:**

<ResourceCard>

<Icon /> {/\* Category-specific icon \*/}

<Title>{resource.title}</Title>

<Authority>{resource.source\_authority}</Authority>

<Description>{resource.description}</Description>

<BestFor>

💡 Best for: {resource.best\_for}

</BestFor>

<Metadata>

{resource.cost === 'free' && <Badge>Free</Badge>}

{resource.hours\_available === '24/7' && <Badge>24/7</Badge>}

</Metadata>

<Actions>

<Button href={resource.url} target="\_blank">

Visit Resource

</Button>

{resource.contact\_phone && (

<Button variant="secondary">

Call {resource.contact\_phone}

</Button>

)}

</Actions>

<Feedback>

Was this helpful? [👍] [👎]

</Feedback>

</ResourceCard>

**Environment Variables**

Create .env.local:

# Supabase

NEXT\_PUBLIC\_SUPABASE\_URL=your\_supabase\_url

NEXT\_PUBLIC\_SUPABASE\_ANON\_KEY=your\_supabase\_anon\_key

SUPABASE\_SERVICE\_ROLE\_KEY=your\_service\_role\_key

# Anthropic

ANTHROPIC\_API\_KEY=your\_anthropic\_key

# Email (Resend)

RESEND\_API\_KEY=your\_resend\_key

RESEND\_FROM\_EMAIL=noreply@yourdomain.com

# Analytics

NEXT\_PUBLIC\_GA\_MEASUREMENT\_ID=G-XXXXXXXXXX

# App

NEXT\_PUBLIC\_APP\_URL=https://yourdomain.com

**Initial Resource Data**

**Seed with 150 high-quality resources across:**

**Crisis Resources (15):**

* 988 Suicide & Crisis Lifeline
* Eldercare Locator
* National Adult Protective Services
* Alzheimer's Association 24/7 Helpline
* State-specific APS numbers

**Dementia-Specific (40):**

* Alzheimer's Association resources
* NIH National Institute on Aging
* Mayo Clinic dementia guide
* Dementia caregiving courses
* Support groups (online and in-person)

**Legal & Financial (25):**

* Medicare.gov
* Medicaid by state
* Power of Attorney guides by state
* Elder law attorney directories
* Veterans benefits

**Local Services (30):**

* Area Agencies on Aging directory
* Adult day programs
* Respite care locators
* Home health care agencies
* Transportation services

**Caregiver Support (20):**

* Family Caregiver Alliance
* Caregiver support groups
* Burnout resources
* Self-care guides
* Counseling services

**Products (20):**

* Medical alert systems
* Medication management tools
* Mobility aids
* Home safety equipment
* GPS trackers

**Testing Checklist**

**Functionality Testing**

* [ ] All form fields validate correctly
* [ ] Form progresses through all steps
* [ ] Can navigate back and edit answers
* [ ] Resource matching returns relevant results
* [ ] AI guidance generates successfully
* [ ] Results page displays correctly
* [ ] Email sending works
* [ ] PDF generation works
* [ ] Lead submission stores correctly
* [ ] Feedback system works
* [ ] Analytics tracking fires

**User Experience Testing**

* [ ] Form is intuitive and clear
* [ ] Loading states are clear
* [ ] Error messages are helpful
* [ ] Success confirmations are visible
* [ ] Navigation is logical
* [ ] Mobile experience is smooth
* [ ] Desktop experience is smooth
* [ ] Tablet experience works

**Performance Testing**

* [ ] Lighthouse score > 90
* [ ] Page load < 3 seconds
* [ ] Database queries optimized
* [ ] Images optimized
* [ ] Code splitting implemented
* [ ] API responses < 2 seconds

**Security Testing**

* [ ] Environment variables secured
* [ ] SQL injection prevented (Supabase handles this)
* [ ] XSS prevention
* [ ] CSRF tokens where needed
* [ ] Rate limiting on API routes
* [ ] Input sanitization

**Accessibility Testing**

* [ ] Keyboard navigation works
* [ ] Screen reader compatible
* [ ] Color contrast sufficient
* [ ] Focus states visible
* [ ] Alt text on images
* [ ] ARIA labels present

**Monetization Implementation**

**Affiliate Links**

// Track affiliate clicks

async function trackAffiliateClick(resourceId: string, sessionId: string) {

await supabase

.from('resources')

.update({ click\_count: sql`click\_count + 1` })

.eq('id', resourceId);

await supabase

.from('user\_sessions')

.update({

resources\_clicked: sql`array\_append(resources\_clicked, ${resourceId})`

})

.eq('id', sessionId);

}

// Add UTM parameters to affiliate URLs

function buildAffiliateUrl(resource: Resource): string {

const url = new URL(resource.affiliate\_url || resource.url);

url.searchParams.append('utm\_source', 'eldercarenav');

url.searchParams.append('utm\_medium', 'referral');

url.searchParams.append('ref', AFFILIATE\_ID);

return url.toString();

}

**Lead Generation**

// Service provider lead form

<form onSubmit={handleLeadSubmit}>

<input name="name" required />

<input name="email" type="email" required />

<input name="phone" />

<select name="urgency" required>

<option>Immediate</option>

<option>Within a week</option>

<option>Researching options</option>

</select>

<textarea name="message" />

<button>Connect Me With Providers</button>

</form>

// Lead notification email

async function notifyProvider(lead: Lead) {

await resend.emails.send({

from: 'leads@eldercarenav.com',

to: PROVIDER\_EMAIL,

subject: 'New Lead: Home Care in [City]',

html: leadEmailTemplate(lead)

});

}

**Post-Launch Priorities (Week 5+)**

**Week 5-6: Optimization**

* Add more resources (target: 300+)
* Refine AI prompts based on user feedback
* A/B test homepage copy
* Improve matching algorithm
* Add search functionality
* Implement resource filtering/sorting

**Week 7-8: Community**

* Add simple forum/Q&A
* User accounts (optional)
* Resource reviews/ratings
* Share functionality
* Email nurture sequences

**Week 9-12: Growth**

* SEO content (blog posts)
* Google Ads campaigns
* Partner with Area Agencies on Aging
* Service provider partnerships
* Press outreach
* Social media presence

**Development Commands**

# Install dependencies

npm install

# Run development server

npm run dev

# Build for production

npm run build

# Run production server

npm start

# Type checking

npm run type-check

# Linting

npm run lint

# Database migrations (Supabase)

npx supabase migration new <name>

npx supabase db push

# Deploy to Vercel

git push origin main # Auto-deploys