JESSE by ENDevo

Digital Readiness Assessment — MVP Hackathon

Technical Handoff Document

Screen Flow | Data Flow | API Contract | Developer Responsibilities

Field	Detail
Product	Jesse — Digital Readiness Assessment
Version	MVP v1.0 — Hackathon Build
Demo Date	March 8, 2026
Team	Karna (Frontend) | Aryan (Backend) | Nermeen (Architect)
Stack	React + Node.js Express + Claude/OpenAI API + Resend Email
Storage	NONE — session memory only, zero persistence
Primary Output	Branded PDF emailed to user — 7-Day Action Plan

> *This document is the single source of truth for building Jesse. No database. No auth. No user accounts. The entire product is a stateless flow from landing page to email delivery. Build exactly what is here — nothing more.*

🗺 SECTION 1 — USER FLOW OVERVIEW

Complete User Journey (14 Steps)

The entire experience is linear and stateless. The user moves forward only. No logins, no saved state, no back-end database writes.

1	**Landing Screen** User arrives at endevo.life/jesse — sees Jesse branding and single CTA button
2	**Question 1 of 10** ABCD multiple choice — auto-advances on selection. Progress bar visible.
3	**Questions 2–9** Same pattern — one question per screen, slide-in animation, auto-advance
4	**Question 10 of 10** Final question — after selection, transitions to name/email capture
5	**Name + Email Capture** Two fields only: First Name + Email. CTA: 'Send My 7-Day Plan'
6	**POST /api/assess triggered** Frontend sends: { name, email, answers[] } to backend API
7	**Loading Screen** Jesse avatar animating. Rotating copy lines. 5–15 second wait.
8	**Backend: Score Calculation** Custom algorithm: A=10, B=6, C=3, D=0. Domain weights applied. Tier assigned.
9	**Backend: AI Prompt Built** Structured JSON payload assembled from scores + signals
10	**Backend: Claude/OpenAI Call** 7-day action plan generated from prompt template. Fallback to static plan if API fails.
11	**Backend: PDF Generated** Branded PDF: score, donut chart, tier, 7-day plan. Generated in <3 seconds.
12	**Backend: Email Sent**

PDF attached to branded email via Resend API. Subject: Your 7-Day Plan from Jesse

13

API Returns 200

Frontend receives success response. Transitions to confirmation screen.

14

Confirmation Screen

Checkmark + message: 'Your plan is on its way!' Optional: share link.

SECTION 2 — SCREEN-BY-SCREEN DESIGN

Screen Specifications

Every screen the user sees. Owner column shows who builds it. All screens are mobile-first, Tailwind CSS.

Screen 1: Landing / Hero [Frontend — Karna]	
Jesse Logo	ENDevo logo top-left, Jesse character illustration center
Headline	"Find out if your digital life is ready — in 90 seconds."
Subheadline	"Jesse will ask you 10 questions and build your personal 7-day readiness plan."
Disclaimer	Small text at bottom: "Not legal or financial advice. Free educational program. We do not store your data."
CTA Button	"Start My Assessment →" — large, branded button, center of screen
Background	Dark navy (#1B2A4A) gradient or brand color. Calm, not clinical.
UX Notes	*Single click to start. No form fields. No scroll required on mobile. Jesse character should feel warm and approachable.*

Screen 2: Quiz — Questions 1 through 10 (one screen each) [Frontend — Karna]	
Progress Bar	"Q3 of 10" — fills smoothly with each advance. Top of screen.
Domain Tag	Small subtle label: e.g. 'Access & Ownership Risk' — below progress bar
Question Text	Large, readable. Min 24px. Pull from questions array in config.
4 Answer Buttons	A, B, C, D — full-width stacked buttons. Each shows the answer text.
Selection State	On click: selected button highlights in brand orange. 0.5s delay then auto-advance.
Slide Animation	New question slides in from right. Previous slides out to left. Smooth, fast.
Back Arrow	Subtle back arrow bottom-left. Allows user to change previous answer. Does NOT reset all answers.
No Score Display	User NEVER sees their score during the quiz. Forward momentum only.
UX Notes	*Gamified like Duolingo / Typeform. The auto-advance after selection is critical — it creates momentum. Minimum 44px tap targets for mobile. Each question stored in local React state array.*

Screen 3: Name + Email Capture [Frontend — Karna]	
Jesse Copy	"Great — your plan is almost ready. Where should Jesse send it?"
First Name Field	Required. Placeholder: 'Your first name'
Email Field	Required. Email validation before allowing submit. Placeholder: 'your@email.com'
CTA Button	"Send My 7-Day Plan →" — triggers POST to /api/assess
Microcopy	"We'll send your personalized PDF to this email. No spam, ever."
Disable on Submit	Button disables immediately on click to prevent double-submission
UX Notes	*This screen appears immediately after Q10 is answered. Only 2 fields — name and email. Nothing else. This is the lead capture moment. Keep it clean and trustworthy.*

Screen 4: Loading — Jesse is Building Your Plan [Frontend — Karna]	
Jesse Animation	Jesse avatar animating — thinking/writing pose. CSS animation or Lottie file.
Rotating Copy	Lines cycle every 2.5 seconds: (1) Jesse is reviewing your answers... (2) Calculating your Readiness Score... (3) Building your personalized 7-day plan... (4) Almost ready — this is going to be good.
Progress Pulse	Pulsing bar or dots below Jesse. NOT a spinner. Calm, confident energy.
No Score Display	Do NOT show any score or tier on this screen.
Timeout Handling	If API takes >15 seconds, show: 'Still working on your plan... almost there.' Do not redirect.
UX Notes	*This screen holds while backend scores + calls AI + generates PDF + sends email. Target 5–10 seconds. If it takes 15 that's okay — the animation keeps the user engaged. The wait creates anticipation for the email payoff.*

Screen 5: Confirmation [Frontend — Karna]	
Checkmark	Animated checkmark. Green. Satisfying completion animation.
Headline	"Your plan is on its way, [First Name]!" — use name from form
Body Copy	"Check your inbox — Jesse has built your personalized 7-day Digital Readiness Plan. Check your spam folder if you don't see it within 2 minutes."
Share Button	Optional: 'Share this with someone you love' — copy link or WhatsApp share
Secondary CTA	"Learn more about ENDevo →" — links to endevo.life
UX Notes	*Clean finish. No score shown here either — the score lives in the PDF. The email is the wow moment. This screen just confirms it's coming.*

SECTION 3 — DATA FLOW & API CONTRACT

API: POST /api/assess

This is the only API endpoint in the entire product. Frontend calls it once. Backend does everything from that single call.

3.1 — Request Payload (Frontend → Backend)

Karna sends this JSON body when user submits name + email after completing all 10 questions.

```
POST /api/assess
Content-Type: application/json

{
  "name":  "Sarah",
  "email": "sarah@example.com",
  "answers": [
    { "q": 1,  "answer": "D" },
    { "q": 2,  "answer": "B" },
    { "q": 3,  "answer": "D" },
    { "q": 4,  "answer": "C" },
    { "q": 5,  "answer": "D" },
    { "q": 6,  "answer": "D" },
    { "q": 7,  "answer": "B" },
    { "q": 8,  "answer": "C" },
    { "q": 9,  "answer": "C" },
    { "q": 10, "answer": "A" }
  ]
}
```

3.2 — Scoring Engine (Backend — Aryan)

Custom algorithm. No AI involved in scoring. This runs before the AI call.

Answer	Points
A — Fully Prepared	10 points
B — Partially Prepared	6 points
C — Aware but not acted	3 points
D — Unaware / unprepared	0 points

Domain Weights applied to weighted score:

Domain	Weight | Questions
Access & Ownership Risk	30% | Q1, Q2, Q6

Data Loss Risk	20% | Q3, Q8
Platform Limitation Risk	15% | Q4, Q9
Stewardship Risk	15% | Q5, Q7
Financial Exposure	20% | Deferred to Phase 2 — distribute across others in MVP

Tier mapping from total Readiness Score (0–100):

Score Range	Tier Label
85 – 100	Peace Champion
60 – 84	On Your Way
35 – 59	Getting Clarity
0 – 34	Starting Fresh

3.3 — Calculated Payload (Internal — Aryan builds this)

After scoring, Aryan builds this object. This feeds the AI prompt AND the PDF generator.

```
{
  "name":            "Sarah",
  "email":           "sarah@example.com",
  "readiness_score": 42,
  "tier":            "Getting Clarity",
  "domain_scores": {
    "access_ownership":    18,
    "data_loss":            8,
    "platform_limitation":  7,
    "stewardship":          9
  },
  "critical_gaps": ["Q1-D", "Q4-D", "Q9-C"],
  "jesse_signals": [
    "No legacy contact set up",
    "No digital legacy manager designated",
    "2FA partially configured"
  ]
}
```

3.4 — AI Prompt Template (Backend — Aryan)

Prompt is stored server-side. Never exposed to frontend. This template is injected with the calculated payload values.

```
SYSTEM PROMPT:
You are Jesse, ENDevo's warm and trusted digital readiness guide.
You help people feel prepared and clear — not scared or overwhelmed.
Your tone is: warm, direct, practical, encouraging. Never legal or clinical.
No estate planning language. No medical or financial advice. Educational only.

USER PROMPT:
```

```
Generate a 7-day action plan for [name].
Their Readiness Score is [score]/100. Their tier is: [tier].
Their critical gaps are: [jesse_signals].
Their weakest domain is: [lowest_domain].

Format the output as:
Day 1: [Short Title]
[2 sentence warm, actionable description]
... repeat for Day 2 through Day 7

Each action must be specific, achievable in under 30 minutes.
Match the urgency level to their tier.
Do not mention legal documents, attorneys, or financial advisors.
```

3.5 — Fallback Plan (Aryan — CRITICAL for Demo Safety)

If the AI API call fails or times out (>10 seconds), use a pre-written static 7-day plan for that tier. User never knows. Demo never breaks.

> *Build 4 static plans — one per tier. Store them as constants in the backend. If AI throws any error: catch it, log it, use static fallback, continue PDF generation. DO NOT let the user see an error.*

```
const FALLBACK_PLANS = {
  "Peace Champion":   staticPlan_tier1,
  "On Your Way":      staticPlan_tier2,
  "Getting Clarity":  staticPlan_tier3,
  "Starting Fresh":   staticPlan_tier4
};

try {
  plan = await callClaudeAPI(payload);
} catch (err) {
  plan = FALLBACK_PLANS[tier]; // silent fallback
}
```

3.6 — PDF Structure (Aryan generates, Karna does not touch)

PDF is generated server-side using pdf-lib or Puppeteer. Two pages.

Page	Content
Page 1 — Score Profile	ENDevo logo + Jesse character top. User name + date. Large readiness score number. Tier badge with Jesse's opening line for that tier. Donut/ring chart: 4 domain scores (color coded). Disclaimer footer.
Page 2 — 7-Day Plan	Bold header: 'Your 7-Day Digital Readiness Plan'. Day 1 through Day 7 each with title + 2-sentence description. ENDevo branding footer. 'Want to go deeper? Visit endevo.life'

> *PDF must generate in under 3 seconds. Target file size under 500KB. Use ENDevo brand colors: Navy #1B2A4A, Orange #E8651A. Donut chart: Access=blue, DataLoss=teal, Platform=orange, Stewardship=green.*

3.7 — Email (Aryan — via Resend.com)

Field	Value
Provider	Resend.com — free tier, simple API, reliable delivery
From	jesse@endevo.life (configure in Resend dashboard)
Subject	Your 7-Day Digital Readiness Plan from Jesse
Body	Simple branded HTML email. Warm Jesse tone. 'Your plan is attached' + preview of tier label.
Attachment	The generated PDF file
Reply-To	hello@endevo.life

3.8 — API Response (Backend → Frontend)

After email is sent successfully, return this to Karna's frontend:

```
HTTP 200 OK
{
  "success": true,
  "message": "Plan sent successfully"
}

// On any error:
HTTP 500
{
  "success": false,
  "message": "Something went wrong. Please try again."
}
```

> *Frontend shows confirmation screen on 200. On 500, show a friendly error message and let the user try submitting again. Do NOT show technical error details to the user.*

🛠 SECTION 4 — TECH STACK & RESPONSIBILITIES

Tech Stack Decisions

Layer	Choice
Frontend Framework	React + Vite (preferred) or Create React App
Styling	Tailwind CSS — mobile-first, utility classes
Frontend State	React useState / useReducer — NO external state library needed
Backend Framework	Node.js + Express — lightweight, simple
AI Provider	Anthropic Claude API (claude-sonnet-4-6) — or OpenAI GPT-4o
PDF Generation	pdf-lib (lighter) or Puppeteer (more flexible for charts)
Email Delivery	Resend.com — free tier sufficient for MVP
Frontend Hosting	Vercel — free, instant deploy from GitHub
Backend Hosting	Railway or Render — free tier, deploys Node.js easily
Database	NONE — zero persistence, zero storage in MVP
Authentication	NONE — no user accounts required

Developer Responsibilities

Karna — Frontend (React)

- Build all 5 screen states as React components
- Store all 10 answers in React state array as user progresses
- Implement slide-in animation between questions (CSS transition or Framer Motion)
- Auto-advance logic: on answer select, 0.5s delay, then move to next question
- Email validation on capture form before allowing submit
- POST to /api/assess with { name, email, answers[] } on form submit
- Show loading screen while awaiting API response
- On 200: show confirmation screen. On 500: show retry message.
- Questions array stored in a config file (questions.js) — not hardcoded in components
- Mobile responsive — test on 375px width minimum

Aryan — Backend (Node.js)

- Single Express route: POST /api/assess
- Input validation: check name, email, and answers array (must have 10 items)
- Scoring engine: loop answers, apply point values, apply domain weights, calculate total
- Tier assignment based on score ranges
- Build calculated payload object (Section 3.3)

- Inject payload into AI prompt template, call Claude or OpenAI API
- Catch AI errors — use fallback static plans (one per tier)
- Generate PDF (pdf-lib or Puppeteer) — 2 pages per spec in Section 3.6
- Send email via Resend with PDF attached
- Return 200 on success, 500 on failure
- CORS configured to allow requests from Vercel frontend URL
- API key management: use .env file — NEVER commit keys to GitHub

Nermeen — Architecture & Integration

- AI prompt tuning and testing across all 4 tiers
- PDF visual design template and brand review
- Integration testing: full end-to-end flow from landing to email receipt
- Environment variable setup and deployment configuration
- Fallback plan copy writing (4 static tier plans)
- UAT coordination with Niki's testers before March 8
- GitHub repo setup on March 5th (hackathon repo — fresh commit)

SECTION 5 — BUILD TIMELINE & DEMO PLAN

3-Week Sprint to March 8

Week / Date	Deliverable
Week 1 — Now to Feb 22	Karna: All screen components built (static, no API). Aryan: Scoring engine complete + tested with sample inputs. Nermeen: Questions config file finalized, AI prompt drafted.
Week 2 — Feb 22 to Mar 1	Karna: API integration connected, loading/confirmation screens working. Aryan: PDF generation working, email sending working. Full end-to-end tested by team.
Week 3 — Mar 1 to Mar 7	UAT with Jim, Anna, Niki's security contacts. Bug fixes. AI prompt refinement. PDF polish. Fallback plans written. Demo rehearsal.
March 5 — Hackathon Opens	Create fresh GitHub repo. Single 'initial commit' with working code. Hackathon submission registered.
March 7 — Day Before Demo	Final bug fixes. Deploy to production URLs. Full demo run-through as a team.
March 8 — Demo Day	Demo flow: Jesse intro (10 sec) → Answer 3-4 questions live → Jump to pre-filled dataset → Show loading → Show PDF received in inbox → Walk through 7-day plan. Total: 90 seconds.

SECTION 6 — OUT OF SCOPE (DO NOT BUILD)

What We Are NOT Building in MVP

> *If any of these come up during development, say no and flag it. Scope creep will break the March 8 deadline.*

- User accounts or authentication of any kind
- Database or persistent storage of any user data
- Score or results dashboard on-screen (results live in PDF only)
- OAuth or account scanning integrations
- Payments or subscription flow
- Multi-user or admin portal
- HR dashboard or employer-facing features
- Document uploads or vaults
- Social sharing of score or badge
- Chat with Jesse in real-time
- More than 10 questions
- More than one endpoint

Questions? Ping Nermeen on Slack before building anything that isn't in this doc.