0. Product Overview

Puzzle AI is an AI thinking partner for creative work.

- It does **sense-making and convergence**, not just idea generation.
- Users work in two layers:
    - **Home Canvas** – dump and arrange ideas as fragments.
    - **Puzzle Sessions** – focused 4-quadrant “thinking puzzles” around a central question.
- An **AI mascot** sits at the edge of the canvas and:
    - teaches “how to play”,
    - launches puzzles (self-raised or AI-suggested),
    - occasionally reflects patterns in the work.

Core AI operations:

- **Clarify** – make vague intent more precise.
- **Expand** – grow or diversify what’s on the table.
- **Refine** – narrow and polish toward a stronger direction.
- **Connect** – link perspectives and sessions into a coherent whole (mainly on the canvas).

---

1. Core Objects & Data Model

1.1 Project & Process Aim

Each project has:

- **Process Aim / Ideation Brief**
    - Short text describing what the user is trying to do.
    - Stays visible at the top (e.g., “Explore the tension between analog warmth and digital”).
- Used as **long-horizon context** for all AI behavior.

type Project = {
id: string
title: string
processAim: string
}

1.2 Fragments (on Home Canvas)

Fragments are the smallest visible units on the Home Canvas:

- Examples: text notes, questions, images, screenshots, links, mood words, etc.
- Users can drag, resize, group them Miro-style.
- A fragment may later feed into **one or multiple puzzle pieces**, possibly partially.

Important clarification:

- A **puzzle piece ≠ one fragment**.
  It can be:
  - a *part* of a fragment (e.g., only one sentence of a longer note),
  - or a *combination* of several small/similar fragments.

So we do **not** store “this fragment belongs to quadrant X”.
Instead we store:

- **Color label on the fragment** = which **puzzle(s)** it has contributed to (e.g., orange label → “belongs to Nostalgic puzzle”).

type Fragment = {
id: string
type: "text" | "image" | "link" | "other"
content: any
position: { x: number; y: number }
labels: string[] // e.g. puzzle IDs or tags like "NOSTALGIC"
}

Links between fragments and puzzle pieces are many-to-many:

type FragmentLink = {
fragmentId: string
puzzlePieceId: string
// optional span/selector if it's a part of the fragment
}

1.3 Quadrants / Design Modes

Every **Puzzle Session** uses a fixed 4-quadrant frame as different “lenses”:

1. **FORM** – how it looks (shape, layout, visual structure)
2. **MOTION** – how it moves / changes (rhythm, transitions, energy)
3. **EXPRESSION** – what it feels like (mood, tone, personality)
4. **FUNCTION** – what it needs to do (goals, constraints, audience, channels)

Quadrants are visual areas; they don’t own fragments, they own **puzzle pieces**.

type DesignMode = "FORM" | "MOTION" | "EXPRESSION" | "FUNCTION"

1.4 Puzzle & Central Question

A **Puzzle** is a focused thinking unit around one **central question**.

- Central question can be:
    - user-initiated (“I’m not sure how to start the target audience exploration…”),
    - or AI-synthesized (“Let’s untangle how your features map to these audiences.”).
- The question is generated using:
    - user’s explicit request (if any),
    - **Process Aim**,
    - relevant fragments (and their labels),
    - summaries of previous puzzles in the same project.

type Puzzle = {
id: string
centralQuestion: string
projectId: string
createdFrom: "user_request" | "ai_suggested"
}

1.5 Puzzle Pieces (Clarify / Expand / Refine)

Inside a puzzle, users drag **puzzle pieces** out from the four corner hubs (one per quadrant).

Each piece is:

- typed as one of three **puzzle categories**:
    - **Clarify** – make something sharper / more specific.
    - **Expand** – add new angles or variety.
    - **Refine** – converge, choose, polish; user already has some idea.
- associated with one **quadrant (mode)**:
    - FORM / MOTION / EXPRESSION / FUNCTION.
- optionally linked to one or more fragments.

type PuzzlePieceCategory = "CLARIFY" | "EXPAND" | "REFINE"

type PuzzlePiece = {
id: string
puzzleId: string
mode: DesignMode
category: PuzzlePieceCategory
text: string // the question / prompt on the block
userAnnotation?: string // user's short answer or note
anchorIds: string[] // which anchors / nodes it's attached to
fragmentLinks: FragmentLink[]
source: "AI" | "USER" | "AI_SUGGESTED_USER_EDITED"
}

1.6 Anchors： Starting & Solution

Within a puzzle we keep two **Anchors**:

- **Starting (Why)** – underlying motivation / problem / tension.
- **Solution (What)** – the emerging direction or bet.

Puzzle pieces are visually attached to one or both anchors.

type AnchorType = "STARTING" | "SOLUTION"

type Anchor = {
id: string
puzzleId: string
type: AnchorType
text: string
}

1.7 AI Mascot

The **mascot** replaces the sparkle icon as the "AI portal".

Mascot roles:

1. **Guide / Coach**
    - Shows inline hints:
      "You can create new puzzle blocks by dragging the corner pieces."
    - Onboarding and light UX teaching.
2. **Puzzle Launcher**
    - Presents two choices:
        - **"Start from my question"** – user types or says what they're stuck on; mascot synthesizes a puzzle.

- **“Suggest a puzzle for me”** – mascot scans context and proposes a puzzle.

3. **Reflective Mirror (low-frequency)**
    - Occasionally surfaces pattern-based reflections, e.g.
      “You seem to gravitate toward gradient, analog-warm palettes. That supports your aim of …”
    - Never spams; triggered only on strong consistent patterns.

---

2. User Flows

2.1 Home Canvas – Fragments & Mascot

1. User creates or opens a project.
    - Top bar: **Project Name** + **Process Aim** (editable pill).
2. On the canvas, user:
    - adds text notes, images, etc.;
    - moves them around freely.
3. Mascot sits at the edge:
    - shows small hints for new users (how to pan, zoom, drag puzzle pieces, etc.);
    - when there are enough fragments, offers to start a puzzle.

AI in background:

- parses fragments into embeddings,
- detects themes / clusters and simple tags,
- logs which fragments often appear together.

We keep this light; Canvas is mainly for **human layout**, not heavy AI automation.

2.2 Starting a Puzzle

There are two main entry paths, both via the mascot.

A. Self-Raised Puzzle (User-initiated)

1. User clicks mascot → chooses **“Start from my question”**.
2. User writes / says something like:

   “I’m not sure how to start the target audience exploration, and how to map our potential features with their needs.”

3. AI collects context:
    - this utterance,
    - Process Aim,
    - nearby fragments (within view or same cluster),
    - summaries of previous puzzles.
4. AI synthesizes:
    - a **central question**,
    - optionally a first **Starting anchor** draft,
    - and maybe 1-2 seed puzzle pieces per quadrant.

User can edit the central question before entering the puzzle screen.

B. AI-Suggested Puzzle (Mascot-initiated)

1. User clicks mascot → chooses **“Suggest a puzzle for me”**.
2. AI scans:
    - Process Aim,
    - current fragment landscape,
    - which quadrants / topics feel under-explored,
    - patterns from previous puzzles (e.g., many form choices but no audience reasoning).
3. AI proposes:
    - a central question,
    - short explanation (“Why this puzzle?”),
    - and the quadrant focus (“This will explore FUNCTION and EXPRESSION for your target audience.”).

User can accept → opens puzzle, or decline → mascot quietly backs off.

2.3 Puzzle Session – 4 Quadrants

Once the puzzle opens:

- Center: **central question**.
- Top/bottom/left/right areas: **FORM / MOTION / EXPRESSION / FUNCTION quadrants**.
- Four corners: **Puzzle Hubs**. Dragging from a hub creates a new puzzle piece.

2.3.1 What users do

- Drag puzzle pieces out of hubs (Clarify / Expand / Refine).
- Attach them:
    - near the central question,
    - to Starting/Solution anchors,
    - or to each other (forming small argument chains).
- Answer directly inside the piece (user annotation).
- Optionally tag which fragments are relevant (via color / “link fragments” action).

Over time, the puzzle becomes a small **argument map** around the central question.

2.3.2 AI behavior by category

Clarify pieces

- Goal: sharpen vague statements.
- Typical triggers:
    - heavy use of fuzzy words (“clean”, “premium”, “fun”) with no concrete specifics,
    - contradictory adjectives,
    - missing audience / function info.

Examples per quadrant:

- FORM: “Should the design feel more geometric or organic?”
- MOTION: “Is the motion calm and smooth, or bouncy and energetic?”
- EXPRESSION: “Which 3 emotions best describe what you want people to feel?”
- FUNCTION: “Where will this mostly be seen (mobile, print, in-store)?”

Expand pieces

- Goal: bring in fresh angles and avoid local minima.
- Triggers:
    - quadrant is almost empty,
    - user repeatedly writes “not sure yet” or similar,
    - very narrow set of ideas.

Examples:

- FORM: “If this interface were an object, what would it feel like to touch?”
- MOTION: “Imagine a 2-second intro animation—what happens in it?”
- EXPRESSION: “If the product had a voice, whose voice would it sound like?”
- FUNCTION: “Are there constraints (budget, accessibility, timelines) we haven’t named yet?”

Refine pieces

- Goal: converge and polish once there is enough material.

Refine is its **own category**, not just a subtype of clarify:

- It assumes the user already **has** a direction; now we:
  - pick the few essentials,
  - reconcile conflicts,
  - choose trade-offs.

Triggers:

- quadrant has many notes / pieces (e.g., ≥5),
- there is explicit user signal like “I think I know, but it’s messy.”

Examples:

- FORM: “From everything here, what 2 visual elements *must* remain?”
- MOTION: “Which motion concept best fits your analog-warm goal?”
- EXPRESSION: “Which one emotional axis feels most important to commit to now?”
- FUNCTION: “If you had to prioritize only one primary job-to-be-done, which is it?”

2.3.3 Connect (inside puzzle)

Even inside a single puzzle, AI will sometimes offer **Connect** pieces that bridge quadrants:

- FORM ↔ EXPRESSION: “How do these shapes support the feeling of nostalgia you described?”
- MOTION ↔ FUNCTION: “How does this level of movement affect readability or clarity?”

These appear as optional suggested pieces in hubs, especially once multiple quadrants have some content.

2.4 Ending a Puzzle & Returning to Canvas

When the user feels they’ve “got something”:

1. They click “Finish puzzle” (or mascot suggests “Looks like you’ve converged—want to wrap this up?”).
2. AI performs a **puzzle synthesis**:
    - summarizes:
        - Starting & Solution anchors,
        - key Clarify/Expand/Refine pieces attached to them,
        - notable Connect pieces.
    - outputs a short structured summary:
        - `Direction statement`
        - `Key reasons (3–5 bullets)`
        - `Open questions / risks (optional)`
3. On Home Canvas, the system:
    - drops a **Puzzle Summary Card** near the most relevant fragment cluster,
    - color-labels any fragments linked to this puzzle (e.g., orange),
    - keeps a “View Puzzle” link from the card back to the puzzle layout.

From here, the user can:

- Use the canvas to **Connect across puzzles**:
    - physically cluster related summary cards,
    - draw connecting lines or frames,
    - ask mascot to “Compare these two directions” or “Help me choose”.

---

3. AI Operations & Orchestration

3.1 Operations at a glance

- **Clarify / Expand / Refine → Puzzle level, per quadrant**
- **Connect → two layers**:
    - inside puzzles (bridging quadrants),
    - on the canvas (bridging different puzzles / themes).

3.2 Context the AI sees

For each call, the orchestrator assembles:

- Project:
    - Process Aim
    - list of existing puzzle summaries
- Current view:
    - visible fragments and their text / tags
    - any selected fragments
- Current puzzle (if in session):
    - central question, anchors
    - existing puzzle pieces and annotations
- Interaction history:
    - recent mascot interactions
    - last N user edits (for responsiveness)

This is used to:

- synthesize central questions,
- choose which category (Clarify / Expand / Refine / Connect) to propose,
- generate reflections from mascot without being annoying.

---

4. Mascot Behavior Model

The mascot is the **single interface to AI**, instead of scattering icons.

Modes

1. **Guide mode (onboarding & micro-tips)**
    - Appears mainly when:
        - user is new to the tool,
        - user has not yet used puzzles,
        - user enters puzzle view for the first time.
    - Example balloons:
        - “You can create new puzzle blocks by dragging the corner pieces.”

        - “Try dropping a couple of thoughts on the canvas, then I’ll help you turn them into a puzzle.”

2. **Puzzle mode (launcher)**
    - When user clicks mascot:
        - show a simple panel:
            - □ “Start from my question”
            - □ “Suggest a puzzle for me”
    - From there, follow the flows described in §2.2.
3. **Reflection mode (low-frequency)**
    - Triggered only when strong patterns are detected (e.g., consistent color palette, repeated adjectives).
    - Example:

      “You keep using ‘analog warmth’ and choosing grainy photos. This seems central to your concept—do you want to make it explicit in your brief?”
    - This can be turned off or snoozed.

Feature simplification:

- Keep mascot’s abilities **focused** on:
    - teaching,
    - launching puzzles,
    - lightweight reflection.
- No heavy “chat” / open-ended conversation to later versions to avoid scope creep.

---

5. Technical Simplifications & Integrations

1. **No per-fragment quadrant ownership**
    - Only store links between fragments and pieces when the user explicitly associates them.
    - Use color labels at fragment level only for “belongs to this puzzle”.
2. **Exactly 3 puzzle categories**
    - Clarify / Expand / Refine in the UI and the prompts.
    - Connect is a separate system-level operation, not a piece category.

3. **Static 4-quadrant template**
    - FORM / MOTION / EXPRESSION / FUNCTION for all puzzles in v1.
    - Later we can look at domain-specific templates, but v1 stays unified.
4. **Single AI entrypoint (mascot)**
    - No extra sparkle icon or separate “Summarize” buttons at first.
    - Summaries & next steps are triggered through mascot (“wrap this puzzle”, “what next?”).
5. **Lightweight context store**
    - Start with a simple document store + embeddings (no heavy graph DB yet).
    - Schema only for:
        - projects,
        - fragments,
        - puzzles,
        - puzzle pieces,
        - anchors,
        - puzzle summaries.

---

clarifications:

- puzzle pieces can be sub/combos of fragments,
- fragments are tagged to puzzles via color/links rather than quadrants,
- **Refine** is a full puzzle category for convergence,
- central questions are synthesized from both explicit user queries and global context,
- mascot replaces sparkle and bundles guidance + puzzle launching + reflections.