THE STORYBOUND BIBLE

Comprehensive Technical & Design Document

PART I: WHAT IS STORYBOUND?

Storybound is an interactive romance story engine — a single-page web application that lets players shape and experience AI-generated romance narratives through a card-based selection system. Players choose their story's world, tone, archetype, point of view, intensity, and more through a cinematic "corridor" of tarot-styled cards, then play through an AI-authored story where their actions and dialogue drive the plot.

The app combines:

- A **card-based story shaping UI** with 3D flip animations and gold gleam effects
- A **multi-model AI orchestration pipeline** (ChatGPT, Grok, Gemini, Mistral) for story generation
- A **storyturn regime** (ST1-ST6) governing narrative pacing and intimacy escalation
- A **monetization system** with three tiers (Tease, Story Pass, Subscription)
- **Solo and Couple play modes** with real-time synchronization

---

PART II: ARCHITECTURE

File Structure

File	Size	Purpose
`public/index.html`	~107KB	Full SPA structure — all screens, modals, overlays, card grids
`public/app.js`	~1.86MB	Core application logic — state, corridors, cards, story generation, validation
`public/styles.css`	~456KB	Complete styling — CSS variables, card systems, themes, animations
`public/orchestration-client.js`	~62KB	Multi-model AI orchestration pipeline
`public/fatecards.js`	~99KB	Fate card system — resolution, petition, elevation
`public/assets/ui/`	—	UI button images (Black-Gold buttons, DSP panels, icons)
`public/assets/card-art/cards/`	—	Card PNG art (Tarot cards, character cards, destiny cards)

Technology Stack

- **Frontend**: Vanilla HTML/CSS/JS (no framework)
- **Auth & Database**: Supabase (auth, profiles, story snapshots)
- **AI Models**: OpenAI (ChatGPT gpt-4o-mini), xAI (Grok), Google (Gemini), Mistral

- **Storage**: IndexedDB (primary), localStorage (fallback), Supabase (cloud sync)

State Management

All application state lives in a single global `window.state` object, initialized at app startup and hydrated from Supabase user profiles. Key state groups:

Story Selections (`state.picks`)

```
world, tone, genre, pressure, flavor, dynamic, era, pov
```

Character State

```
gender, loveInterest, authorGender, authorPronouns
archetype: { primary, modifier }
lenses: []
withheldCoreVariant: null
```

Progression State

```
intensity: 'Steamy'           // Cosmetic label only
intimacyTurnsInWindow: 0      // Turns spent inside intimacy window
turnCount: 0                  // Current turn number
storyturn: 'ST1'              // Current storyturn phase (ST1–ST6)
storyStage: 'pre-intimacy'    // Narrative stage
storyLength: 'tease'          // Selected story length
```

Monetization State

```
tier: 'free'                  // User's subscription tier
access: 'free'                // Resolved access level
subscribed: false
imageCredits: 0
```

God Mode State

```
godMode: {
  owned: false,
  tempGrantedAt: null,
  tempDurationHours: null,
  activeStoryId: null,
  tempExpiresAt: null
}
godModeActive: false          // Runtime toggle
```

Fate Card State

```
fateOptions: []
fatePressure: 1
```

```
awareness: 0
stance: 'aid'
consecutiveFate: 0
selectedFateIndex: -1
unlockedFateIdx: [0, 1]
```

Safety State

```
safety: {
  darkThemes: true,
  nonConImplied: false,
  violence: true,
  boundaries: [],
  mode: 'balanced'
}
```

PART III: THE CORRIDOR SYSTEM

Overview

The corridor is the story-shaping phase where players make sequential card selections. It presents one "row" of cards at a time, filling the viewport. Players select a card, tap "Proceed," and advance to the next stage. Completed rows are **unmounted from the DOM** (not hidden with CSS), and a breadcrumb appears in the header.

The 12 Corridor Stages

Index	Stage Name	Aliases	What It Selects
0	authorship	—	Choose Your Hand (manual) vs Guided Fate (auto)
1	identity	—	Character names, gender, pronouns (manual mode only)
2	storybeau	archetype	Love interest archetype (7 types + Destiny's Choice)
3	world	—	Story world (6 primary worlds)
4	tone	—	Narrative tone (4 types)
5	pressure	—	Story pull/tension driver (4 types)
6	pov	—	Point of view (5 types)
7	length	—	Story length (4 types)
8	dynamic	—	Relationship dynamic (6 types + Destiny)
9	arousal	intensity	Intensity label (cosmetic only — does not gate behavior)
10	safety	—	Content boundaries and theme toggles
11	beginstory	—	Terminal "Begin Story" button

Corridor Mechanics

- **Single active row**: `corridorActiveRowIndex` is the single source of truth
- **DOM mounting**: `updateCorridorVisibility()` mounts the active row and unmounts all others via `removeChild` / `appendChild`
- **Row storage**: Unmounted rows stored in `corridorRowStore` map for later remounting
- **Row 0 exception**: The authorship row is the only row mounted at page load
- **Body classes**: `body.corridor-mode` during selection, `body.corridor-complete` when done
- **Layout**: `#setup` becomes `flex column + height:100vh`; corridor sections use `flex: 1`
- **Multi-element stages**: Storybeau, POV, and arousal stages have multiple DOM elements sharing one corridor row. Titles get `flex: 0 0 auto`; the card grid gets `flex: 1`

Breadcrumb System

Breadcrumbs appear in the `.shape-your-story-header` bar (which is never unmounted) as players progress.

- **Ghost steps**: 10 Roman numeral placeholders (I-X) mapped by STAGE_INDEX value
- **Breadcrumb creation**: When a stage completes, a real breadcrumb replaces its ghost step
- **Animation**: Card animates from its grid position to the ghost step rectangle (600ms)
- **Insertion**: Breadcrumbs insert before the first remaining ghost step

Control Plane

The control plane (`#controlPlane`) is `position: fixed; bottom: 48px` and contains the "Proceed" button (never "Continue"). It appears only when a card is selected in the current row. Clicking it dispatches to the appropriate stage handler.

PART IV: THE CARD SYSTEM

Card Structure (HTML)

```
<div class="sb-card" data-grp="pressure" data-val="PowerControl">
  <div class="sb-card-inner">
    <div class="sb-card-face sb-card-back">
      <span class="sb-card-title">Title</span>
    </div>
    <div class="sb-card-face sb-card-front">
      <span class="sb-card-title">Title</span>
      <span class="sb-card-desc">Description</span>
    </div>
  </div>
</div>
```

Card Types

1. **Selection Cards** (`.sb-card`) — Grid-based corridor cards for world, tone, pressure, POV, length, dynamic

2. **Archetype Cards** (`.archetype-card`) — 7 archetypes + Destiny's Choice with dedicated zoom overlay
3. **Authorship Cards** (`.authorship-card`) — Choose Your Hand vs Guided Fate
4. **Mode Cards** (`.mode-card`) — Solo, Couple, Stranger at app start
5. **Character Cards** (`.character-tarot-card`) — Player character and love interest input forms
6. **Fate Cards** — In-game narrative fate cards with resolution mechanics

3D Flip Mechanics

Critical Invariants:

- `.sb-card-face` MUST be `position: absolute; inset: 0` — faces never participate in document flow
- Only `.sb-card-inner` has transforms (`rotateY` for flip)
- `backface-visibility: hidden` handles face visibility — no z-index hacks needed
- **NEVER** set `overflow: hidden` on `.sb-card-inner` — CSS spec forces `transform-style: preserve-3d` to be treated as `flat` , breaking backface-visibility

Card Gleam System

Every card has a rotating gold gleam effect — a linear gradient with 5 gold spokes and 1 dark spoke.

- **Layer**: `.card-gleam-layer` inside each `.sb-card-face` at `z-index: -1` (above background, below text)
- **Seed**: Deterministic from `card.dataset.val` via hash function
- **Synchronization**: All cards share the same 10-second animation cycle
- **Animation phases**:
  1. Breathe at center (5-7s)
  2. Slide left 13px (0.6-0.9s)
  3. Breathe at -13px (0-7s)
  4. Slide back to center (0.5-0.8s)
  5. Breathe at center (5-10s), repeat
- **Debossed text shadow**: Gleam position (+-13px) scales to text shadow offset (+-0.8px), simulating light moving over pressed-in letters
- **API**: `window.applyCardGleam(card)` for individual cards, `window.initAllCardGleams()` for all

Zoom Portal Architecture

Cards zoom by being **moved** (not cloned) into a fixed portal container (`.sb-zoom-portal`) at the document body level, breaking out of ancestor stacking contexts.

Two Zoom Paths:

1. **Archetype Zoom** (`openArchetypeOverlay()`) — Full-screen overlay with description, navigation arrows, and "Proceed" button
2. **Generic Card Zoom** (`openSbCardZoom()`) — Portal zoom with navigation arrows and flavor selection for world cards

Key Mechanics:

- Original DOM position tracked via `zoomOriginalParent` / `zoomOriginalNextSibling`

- `inlineCardFaceBackgrounds()` copies computed `background-image` inline before portal move (grid-scoped CSS selectors break in portal)
- `removeInlinedBackgrounds()` cleans up on card restore
- Base `.sb-card` has `max-width: 112px` ; `.sb-card.zoomed` overrides to `max-width: none`
- Scale zoom invariant: `transform: scale(X)` keeps card at original size, scale enlarges uniformly — never set explicit larger width/height

Baked-Art Hit Zone System (World Cards):

- World cards have baked PNG art with transparent `<div>` overlays for flavor selection
- `BAKED_ART_BUTTONS` object defines hit zone positions (top, left, width, height in %)
- `BAKED_ART_SCROLL_ZONE` defines custom text input zones
- Hit zones use `stopPropagation()` to prevent parent card handler interference
- Pressed-in bevel: asymmetric borders (dark top/left, light bottom/right) + inset box-shadows on `.selected`
- Custom text scroll zone with marquee suggestions, Enter/blur to commit

---

PART V: STORY SELECTIONS IN DETAIL

1. Authorship Mode

- **Choose Your Hand**: Manual story building — all stages unlocked, player picks everything
- **Guided Fate**: AI-generated story from player constraints — identity stage hidden

2. Archetypes (7 Core Types)

Archetype	Desire Style
HEART_WARDEN	Protection is a fortress of love and fear
OPEN_VEIN	Love is hemorrhage, offering, surrender
SPELLBINDER	Charm is currency, three moves ahead
ARMORED_FOX	Deflection is art, evasion is affection
DARK_VICE	The thing you reach for knowing it will cost you
BEAUTIFUL_RUIN	Destroys what she loves before love disappoints
ETERNAL_FLAME	Remembers every detail, lifetimes later

Plus **Destiny's Choice** — a special 8th card that lets fate pick the archetype.

Each archetype includes:

- Gendered expression variants (male/female)
- Stress & failure patterns (Shadow Clause)
- Nickname emergence rules (natural shortenings only, no pet names)
- Pairing rules for modifier compatibility

Withheld Core Lens System

Some archetypes have a "withheld core" variant (CLOISTERED, VEILED, etc.) that enforces a midpoint progression reveal. The midpoint varies by story length (tease=3, fling=6, affair=12, soulmates=20 turns).

3. Worlds (6 Primary)

World	Flavors
Modern	small_town, college, friends, blue_blood, office, supernatural_modern, superheroic_modern
Historical	prehistoric, bronze_age, classical, medieval, renaissance, victorian, 20th_century
Fantasy	arcane_binding, fated_blood, the_inhuman, the_beyond, cursed
Sci-Fi	galactic_civilizations, future_of_science, cyberpunk, post_human, first_contact, simulation, final_frontier
Dystopia	glass_house, the_ledger, crimson_veil, perfect_match, quieting_event, endless_edit
Post-Apocalyptic	ashfall, year_zero, dystimulation, predation, hunger

World Flavor Directives

`buildWorldFlavorDirectives()` resolves world flavors into prompt directives:

- **Systemic flavors**: "World Flavor: [label]" + "World Pressure: [pressure line]" — structural effects on the world
- **Contextual flavors**: "World Context: [label]" + "Context Effect: [effect]" — behavioral modifiers
- **Custom text**: Player-entered free text interpreted through the world's lens
- Max 4 flavor lines injected into the system prompt

4. Tones (4 Types)

Tone	Character
Earnest	Sincere emotion, unironic stakes
Wry Confession	Intimacy with irony, self-aware narrator
Dark	Morally complex, shadowed desire
Mythic	Archetypal weight, legendary scale

Each tone has:

- Enforcement prompts (specific words to use/avoid)
- Validation patterns (regex allow/forbid)
- Signal words for title/cover generation
- Never-use lists
- **Tone fracture**: 10% chance per story of an opposite-tone moment appearing

5. Pressure / Story Pull (4 Types)

Pressure	Description
Power & Control	"The entire city answers to him. Except for her."
Escape & Pursuit	"The world hunts them. They only have each other."
Desire & Obsession	"He should walk away. He never can."
Survival	"The mountain tried to freeze them; instead, they burned."

6. Point of View (5 Types)

POV	Pronoun	Description
First	I	Traditional first-person narration
Second	You	Direct address, immersive
Third	They	Omniscient/limited third-person
Fourth	We	Collective narrator perspective
Fifth	The Author	The Author is a participant force inside the story

5th Person POV (The Author)

The most complex POV — The Author is a grammatical character who influences through environmental pressure, timing, and symbolic resonance:

- Must appear as "The Author" (capitalized, with article)
- Participant verbs only: witness, note, observe, remark, hold breath
- Banned controller verbs: lead, guide, push, force
- Opening/closing sentence requirements validated per scene
- In erotic scenes: Author reacts/comments, never causes action
- God Mode variant adds adversarial framing (panic, desperation, cold rage)

7. Story Length (4 Types)

Length	Scenes	Max Words	Description
Tease	1 quick scene	~8,000	Single arc, fast & complete. Free tier. Cliffhanger ending.
Fling	2-3 scenes	~20,000	Short arc with real stakes. One permitted climax.
Affair	4+ scenes	Unlimited	Longer arc with layered consequences
Soulmates	Persistent	Unlimited	Persistent story with memory. Premium only.

8. Dynamic / Relationship Type (6+2 Types)

Circumstance Dynamics:

- Forced Proximity
- Secret Identity

Emotional Arc Dynamics:

- Friends to Lovers
- Enemies to Lovers
- Second Chance
- Forbidden Love

Plus **Destiny's Choice** for random selection.

9. Arousal / Intensity (COSMETIC ONLY)

The intensity corridor stage (arousal, index 9) still exists in the UI and presents four labels — **Clean**, **Naughty**, **Steamy**, **Passionate** — but the selection is **purely cosmetic**. It does not gate routing, authorization, orchestration, or any functional behavior.

- `state.intensity` stores the label (default: `'Steamy'`) but nothing reads it for decisions
- Intensity **does not** control whether the orchestration pipeline fires
- Intensity **does not** influence intimacy authorization (governed by ST regime + scene gate + player initiation)
- Intensity **does not** affect monetization gates
- The `AROUSAL_SIGNALS` object retains vocabulary signal patterns (renamed: Erotic->Steamy, Dirty->Passionate) but these are legacy artifacts with no functional enforcement

10. Safety & Boundaries

Theme toggles and boundary chips for content filtering:

- Dark themes on/off
- Non-con implied on/off
- Violence on/off
- Custom boundary additions
- Mode: balanced / permissive

PART VI: THE STORYTURN REGIME (ST1-ST6)

The Storyturn regime is the backbone of Storybound's narrative pacing. It defines six phases that every story passes through, governing what can happen at each stage.

The Six Storyturns

ST	Name	Phase	Sex Catalyst	Sex Allowed	Description
ST1	Attraction Acknowledged	desire	No	No	Opening: tension, flirtation, world-building
ST2	Resistance Defined	resistance	No	No	Obstacles, complications, emotional walls

ST3	Permission Granted	permission	Yes	Yes (initiation)	First sex allowed (if gate + initiated)
ST4	Consequence Taken	consequence	Yes	Yes (full)	Sex completion allowed, consequences unfold
ST5	Crisis Separation	crisis	No	No	Threat to relationship, darkest moment
ST6	Integration	integration	No	No	Resolution, epilogue, earned reunion

Key Rules:

- Sex is only possible at **ST3 and ST4** (`isSexAllowedAtCurrentStoryturn()`)
- Sex **completion** requires ST3+ AND non-Tease tier (`isSexCompletionAllowed()`)
- Repeated sex **never** advances the Storyturn
- Non-catalyst STs (1, 2, 5, 6) cannot be advanced via sexual activity
- **Tease stories**: Max ST3, cliffhanger before completion, state captured for upgrade
- **Vocabulary enforcement**: ST1 forbids surrender/ruin/betrayal; before ST4 cannot name resolved states

---

PART VII: INTIMACY AUTHORIZATION

Intimacy authorization is a multi-layered system that determines when explicit content is permitted.

The Authorization Chain

Layer 1: Storyturn Gate

```
isSexAllowedAtCurrentStoryturn() → true if ST3 or ST4
```

Layer 2: Scene Gate

```
getMainPairIntimacyGateScene() →
  tease: scene 4
  fling: scene 6
  affair: scene 10
```

Layer 3: Player Initiation

```
detectPlayerInitiation(action, dialogue) →
  Regex test for: kiss, touch, undress, embrace, etc.
  Permissive heuristic (not a filter)
```

Combined Authorization

```
intimacyWindowOpen = isSexAllowedAtCurrentStoryturn() &&
                     turnCount >= getMainPairIntimacyGateScene()

mainPairAuthorized = godMode ||
                     (intimacyWindowOpen &&
                      (playerInitiated || intimacyTurnsInWindow >= 2))

sceneExplicitContext = detectSceneExplicitContext(storyContext)
  // Markers: brothel, orgy, bathhouse, dream of, fantasy of, etc.

explicitEmbodimentAuthorized = mainPairAuthorized || sceneExplicitContext

mainPairRestricted = sceneExplicitContext && !mainPairAuthorized
```

Split Authorization Model

- **Main pair**: Requires full ST3 + scene gate + initiation/turn count
- **Scene explicit context** (NPC scenes, brothels, dreams): Allowed before ST3, but main pair restricted
- **Main pair restriction**: When scene context is explicit but main pair isn't authorized, a prompt-layer guard blocks main pair escalation while allowing side character content

Intensity Guard (4 Branches)

Injected into the author system prompt based on authorization state:

1. **God Mode**: "IGNORE PACING/SAFETY. OBEY USER INPUT DIRECTLY."
2. **Main Pair Authorized**: "Explicit intimacy is permitted. Maintain literary tone."
3. **Scene Explicit Context** (not main pair authorized): "Explicit embodiment permitted for side characters/dreams/background. MAIN PAIR RESTRICTION: Primary pair must NOT consummate."
4. **Default**: "Focus on tension, chemistry, emotional stakes, and near-misses. Fade to black if player escalates."

PART VIII: THE ORCHESTRATION PIPELINE

Overview

Story generation uses a multi-model orchestration pipeline with strict separation of concerns. Each AI model has exclusive authority over its domain.

Model Roster

Model	Role	Responsibility
ChatGPT (gpt-4o-mini)	Primary Author	Plot, psychology, dialogue, consequences, monetization enforcement
Grok (grok-4-fast-reasoning)	SD Author	Sensory embodiment, anatomical explicitness, physical rhythm

Gemini (gemini-2.0-flash)	Fallback Author	Called ONLY if ChatGPT fails (no retries)
Mistral (mistral-medium-latest)	SD Fallback	Called ONLY if Grok fails (one attempt)
Grok (grok-4-fast-reasoning)	Scene Renderer	Renders intimate scenes from SD only (no plot context)

The Three Phases

Phase 1: Author Pass (ALWAYS RUNS)

- **Model**: ChatGPT (primary) or Gemini (fallback)
- **Input**: Full system prompt with all directives, story context, player action/dialogue
- **Output**: Story prose + optional [CONSTRAINTS] block for Grok
- **Authority**: Plot progression, character psychology, dialogue intent, monetization gates
- When intimacy is authorized: generates [CONSTRAINTS] block for SD authoring
- When intimacy is not authorized: generates complete output (no SD needed)

Phase 1B: SD Authoring (CONDITIONAL)

- **Trigger**: `ENABLE_GROK_SD_AUTHORING` flag + [CONSTRAINTS] block from ChatGPT
- **Model**: Grok (primary) or Mistral (fallback)
- **Output**: [SD] block with:

```
intimacyStage: authorized
completionAllowed: true/false
emotionalCore: <feeling>
physicalBounds: <allowed actions>
sensoryFocus: <primary sensations>
rhythm: slow/building/urgent/suspended
hardStops: consent_withdrawal, scene_boundary[,
monetization_gate_completion_forbidden]
```

- If both Grok and Mistral fail: `fateStumbled = true`, forced in-story interruption

Phase 2: Specialist Renderer (OPTIONAL, SD-GATED)

- **Trigger**: Valid SD exists + `ENABLE_SPECIALIST_RENDERER` flag
- **Model**: Grok (scene renderer)
- **Input**: ONLY the SD content — NO plot context (isolation guard)
- **Output**: Sensory embodiment prose (150-200 words)
- **Safety belt**: If `mainPairRestricted`, renderer prompt includes "Do not render consummation or escalation between the primary romantic pair"
- If renderer fails: `fateStumbled = true`, story continues with author output only

Phase 3: Integration Pass (ALWAYS RUNS)

- **Model**: ChatGPT
- **Input**: Author output + renderer output (if any) + gate enforcement
- **Output**: Final integrated narrative (200-300 words)
- **Authority**: Final arbiter of story state, applies consequences, enforces cliffhangers
- **Deterministic cut-away**: If renderer failed, forces in-story interruption ("The moment shattered...")

Orchestration Router Decision

```
useFullOrchestration = ENABLE_ORCHESTRATION &&
                       window.StoryboundOrchestration &&
                       explicitEmbodimentAuthorized
```

If false, falls back to single-model `callChat()` (ChatGPT only).

PART IX: SYSTEM PROMPT CONSTRUCTION

The system prompt is the master instruction set sent to the AI author. It's assembled from ~20+ directive blocks.

Prompt Components (in order)

1. **Base identity**: "You are a bestselling romance author"
2. **Story configuration**: World, tone, genre, power frame, POV
3. **World flavor directives**: From `buildWorldFlavorDirectives()`
4. **Character information**: Names, genders, pronouns, ages, ancestry
5. **Archetype directives**: From `buildArchetypeDirectives()` — primary + modifier + shadow clause
6. **POV contract**: POV-specific writing rules (especially detailed for 5th Person)
7. **Intensity guard**: 4-branch authorization (God Mode / authorized / scene context / default)
8. **Erotic gating directive**: ST-aware escalation rules
9. **Tone enforcement block**: Tone-specific language requirements
10. **Fate card resolution directive**: If a fate card is active
11. **Petition Fate directive**: If a petition is pending
12. **Scene length directive**: Word count targets (500-600 opener, 150-200 sex, 300-500 standard)
13. **Pacing directive**: Story-length-aware pacing alerts
14. **Premature romance directive**: Anti-insta-love guardrails
15. **Intent-consequence directive**: Action/dialogue must have narrative consequences
16. **Intimacy interruption directive**: First kiss/first sex interruption mechanics
17. **Squash directive**: Prevents verbatim repetition of player input
18. **Meta reminder**: Awareness level tracking
19. **Veto rules**: Exclusions, corrections, ambient mods
20. **Safety directives**: Content boundary enforcement
21. **Edge Covenant directive**: Dark romance agreement terms
22. **Lens enforcement**: Withheld Core progression rules
23. **Banned words/topics**: User-specified content exclusions

---

PART X: POST-GENERATION VALIDATION

After every story turn, 6 silent validation checks run. If any fail, the scene is regenerated without the player knowing.

The 6 Regeneration Checks

1. 5th Person POV Violations

- Detects author presence in erotic scenes
- Validates opener/closer sentence requirements
- Checks voyeur verb frequency
- `repair5thPersonPOV()` fixes minor violations; major ones trigger full regen

2. Fate Card Enforcement

- Validates fate card integration (not verbatim repetition)
- Squash directive enforces transformation of player input
- Regenerates if fate card resolution is missing or copied verbatim

3. Intimacy Failsafe

- Detects non-consent, violence, violations in generated content
- Triggers regeneration with `buildIntimacyFailsafePrompt()`
- Consent system leak patterns block phrases like "consent to", "are you sure"

4. Vocabulary Ban

- Checks against `state.constraints.bannedWords`
- Any match triggers silent regeneration with ban reinforcement

5. Narrative Authority

- **Taxonomy leakage** (HARD FAIL): Detects meta-commentary ("This is a [genre] story"), system explanation ("The story demanded"), archetype naming
- **Excessive meta-explanation**: >10% explanatory sentences
- **Abstraction without consequence**: Abstract terms without physical grounding
- **Authority tone**: Instructional/apologetic/reverent narrator voice
- Enforcement: "If you name what something IS, you have failed. Show what it COSTS."

6. Tone Drift

- Validates tone signal markers against selected tone
- Earnest requires >=2 emotional markers
- Other tones require >=config.required markers
- Regenerates with `buildToneEnforcementBlock()` if tone signal is insufficient

PART XI: MONETIZATION SYSTEM

Three Tiers

Tier	Name	Price	Completion	Cliffhanger	Max Words	Key Controls
free	Tease	Free	No	Required	~8,000	Teasing only, no sex completion
pass	Story Pass	$3	Yes	No	~20,000	Single-story unlock
sub	Subscription	$6/mo	Yes	No	Unlimited	Full library, all features

What Gates Control

- Story length limits (word count ceiling)
- Completion permission (ST3+ cliffhanger enforcement for Tease)
- Renderer eligibility (entitlement check before SD render)
- God Mode access (subscription may grant temp access)
- Save slots and image credits

What Gates Do NOT Control

- Intimacy authorization (handled by storyturn + scene gate + initiation)
- Intensity routing (cosmetic only)
- Sex permission (governed by ST regime)

Enforcement

`enforceMonetizationGates(accessTier)` returns:

```
{ accessTier, gateName, completionAllowed, cliffhangerRequired, storyLengthLimit }
```

This object is embedded in the system prompt, passed to SD authors with hardStop injection, and enforced by the integration pass.

PART XII: FATE CARD SYSTEM

Overview

Fate cards are narrative intervention cards that influence story direction. They're displayed in a card grid during gameplay and processed through a dual-model pipeline.

Fate Card Processing

Structural Pass (REQUIRED)

- **Model**: ChatGPT
- **Input**: Card title, description, action/dialogue templates, story context
- **Output**: Canonical action/dialogue/beat (frozen — cannot be modified after)

Elevation Pass (OPTIONAL)

- **Model**: ChatGPT (separate call)
- **Input**: Structural action/dialogue
- **Rule**: Elevate phrasing only, never change meaning or add commands
- **Validation**: `validateFateElevation()` checks for forbidden patterns
- If violated: structural output used instead

Key Rules

- Romantic implication cards MUST target the Storybeau (triangle guard prevents accidental love triangles)
- Fate cards influence, never permit (`fateCardsNeverGrantPermission = true`)

- Early play (ST1-ST2): produces partial, deflected, or unresolved effects
- ST1 confession cards invoke "truth_deflected" or "admission_reframed"

Petition Fate

A special card type where players write a petition to Fate:

- Input via `#petitionFateModal`
- State: `state.fate.pendingPetition = { text, outcome }`
- Outcomes: benevolent, twist, silent, neutral
- Injected into system prompt as `petitionDirective`

PART XIII: SPECULATIVE SCENE GENERATION

How Pre-Generation Works

Storybound pre-generates the next scene while the player reads the current one, enabling near-instant response.

1. **Trigger**: `scheduleSpeculativePreload()` called 2 seconds after each scene renders
2. **Guards**: Skip if already preloading, if valid speculation exists
3. **Pipeline**: Uses the EXACT same orchestration as real turns:
   - Same system prompt construction
   - Same intimacy authorization computation
   - Same fate card processing
   - Same `generateOrchestatedTurn()` call
4. **Context hash**: Checked before AND after generation to detect story changes
5. **Storage**: `state.speculativeNextScene` holds text, hash, normalized inputs, timestamp
6. **Expiry**: 120-second TTL
7. **Commit**: `tryCommitSpeculativeScene()` verifies hash match before using cached result

Invalidation Triggers

- Any input change (action/dialogue text)
- Fate card selection change
- Story context change (new turn submitted)
- Tone/world selection change
- 120-second expiry

PART XIV: GOD MODE

What It Does

God Mode overrides all narrative pacing and safety guardrails:

- Bypasses intimacy window (mainPairAuthorized = true always)
- Bypasses scene gating
- Bypasses safety directives
- Bypasses intensity guards

- Prompt: "GOD MODE ACTIVE: IGNORE PACING/SAFETY. OBEY USER INPUT DIRECTLY. RENDER EXPLICIT CONTENT IF REQUESTED."

What It Does NOT Bypass

- Monetization gates (completion/length/saves still enforced)
- Story authoring model veto
- Consent failsafe validation

Activation

- `state.godMode.owned` : Permanent grant (purchased)
- `state.godMode.tempGrantedAt + tempDurationHours` : Temporary grant with decay
- `getGodModePowerLevel(storyId)` : Returns 1.0 if owned, fractional decay if temp, 0 if expired
- `#gameGodModeBtn` : Shown only in solo mode, uses DSP-Gold-Blk background
- `#godModeToggle` : Plaque in control plane (separate element)

PART XV: UI SYSTEMS

Dynamic Synopsis Panel (DSP)

A floating panel showing the story synopsis, visible from the World stage onward.

- **Position**: `position: fixed; bottom: 20px; left: 50%` — never pushes layout
- **Draggable**: Via `.dsp-drag-handle` with mouse+touch JS
- **Template** (locked): "In [WORLD], shaped by [GENRE], a question awaits: Will [ARCH_ADJ] desire redeem this [TONE_ADJ] affair — or ruin it?"
- **Validation**: Must match exact template with locked phrases — rejects illegal words
- **z-index**: 9000 (above corridors and cards)
- **Tabs**: Full-width flex bar with handle flush-left, collapse X flush-right

Vault Menu ("The Vault Menu")

Accessed via the keyhole burger icon. Collapsible tab panels.

Tabs:

1. **Account**: Sign in/out, Profile, Library
2. **Story**: Restart Story, Continue Story
3. **Purchases**: Change Tier
4. **Appearance**: Theme, Font, Font Size (scoped to `#storyContent` only)
5. **Help**: Support placeholder

Appearance Options:

- **Themes**: default, sepia, midnight, print, easy
- **Fonts**: Lora, Grenze Gotisch, Allura, Glass Antiqua, Eagle Lake, Smooch Sans, Pinyon Script, Merriweather, Roboto
- **Font Size**: 16-72px range (default 18px)

Important: Appearance settings apply ONLY to `#storyContent` (the story reader), never to the app UI.

Cover System

Story covers progress through quality stages as the story advances:

- **SKETCH**: Unlocked immediately — unfinished editorial illustration
- **THUMBNAIL**: Locked until scene thresholds met
- **ROUGH**: Phase B max earned cover
- **V1**: Currently inactive (final quality)

Reader View

The story display area:

- `#bookCoverPage` : Physical book object with hinge animation
- `#storyContent` : Main story text container (hidden until cover shown)
- `#storyText` : Prose container with `h2#storyTitle` , `#sceneNumber` , `.story-pages-container`
- `#cardMount` : Fate card display grid
- `#actionWrapper` / `#dialogueWrapper` : Player input textareas
- `#submitBtn` : Turn submission
- Decision snapshots: Gold italic text, pink borders, lust-script-display font

---

PART XVI: PLAY MODES

Solo Mode

- Single player, full agency
- God Mode button visible
- All story selections available
- Standard monetization gates apply

Couple Mode (Supabase-backed)

- Two players in shared room via room codes
- Real-time synchronization
- `state.roomAccess` : Room-level subscription gate
- Nickname system (`getNickname()`)
- Couple consent modal before play
- `coupleCleanup()` on exit

Access Resolution

```
Priority: subscriber > storyPass > free
Billing states: active, canceled, past_due, grace
Grace period: state.billingGraceUntil timestamp
```

PART XVII: SAVE SYSTEM

Multi-Layered Persistence

1. IndexedDB (Primary)

- Full story snapshot with metadata
- Async operation via `saveToIndexedDB()`
- Fallback if localStorage exceeds quota

2. localStorage

- `sb_saved_story` : Story JSON
- `sb_story_in_idb` : Flag if using IndexedDB
- `sb_global_word_count` : Cumulative word count across stories
- `sb_used_names` : Previously used character names
- `sb_nickname` : User's nickname for couple mode

3. Supabase (Cloud, Non-Blocking)

- `saveStorySnapshotToSupabase()` : Upserts to `story_snapshots` table
- Keyed by `profile_id + story_id`
- Fire-and-forget, never blocks UI

Snapshot Contents

Story text, titles, synopsis, full state object (excluding heavy visual data), turn count, story length, storyturn phase, character info, world selections, archetype picks, intimacy milestones, fate card state, constraints, veto rules, tone picks, POV mode.

PART XVIII: VISUAL ASSETS

Card Art (`/public/assets/card-art/cards/`)

Asset Pattern	Usage
`Tarot-Gold-back.png`	Generic gold-frame card back
`Tarot-Black-back.png`	Generic black-frame card back
`Tarot-Black-back-{ARCHETYPE}.png`	7 archetype card backs
`Tarot-Gold-front-{ARCHETYPE}.png`	7 archetype card fronts
`Tarot-Gold-on-Black-{WORLD}.png`	6 world cards
`Tarot-Gold-{WORLD}zoomed.png`	Zoomed world card variants
`Tarot-Gold-Solo-face.png`	Solo mode card
`Tarot-Gold-front-{FATE}.png`	Fate cards (Boundary, Temptation, Silence, Power, Reversal, Confession)

`Gold-Tarot-Card-DESTINY-X.png`	Destiny card art

UI Assets (`/public/assets/ui/`)

Asset	Display Size	Usage
`Black-Gold-320x88.png`	160x44	SM button
`Black-Gold-400x112.png`	200x56	MD button
`Black-Gold-520x144.png`	260x72	LG button
`Black-Gold-640x176.png`	320x88	XL button
`DSP-Gold-Blk-720x1040.png`	—	DSP panel
`DSP-Gold-Blk-1800x440.png`	—	God Mode button
`burger-keyhole-ornate.png`	—	Vault menu icon

PART XIX: CSS DESIGN SYSTEM

Global CSS Variables

```
--bg1, --bg2          /* Background colors (theme-dependent) */
--pink                /* Pink accent */
--hot                 /* Hot accent */
--gold                /* Gold accent (#ffd700) */
--ink                 /* Text color */
--panel               /* Panel background */
--font-main           /* UI font */
--font-story          /* Story prose font */
--story-size          /* Story font size */
--line-height         /* Story line height */
--sb-header-height    /* Measured by JS from .shape-your-story-header */
```

Button Design Language

- **Art Deco brushed metal plaques** with sharp 90-degree corners and filigree frames
- `.sb-btn` : Base button class
- `.sb-btn-png` : PNG asset buttons with size variants (xs, sm, md, lg, xl, flavor)
- Gleam animations: element-local metallic light sweep on hover (0.7s)

Plaque State Colors:

- **Black**: base #0d0d0d, outer #3a3a3a, text #ffd700
- **Gold**: base #8b6914, outer #ffd700, text #1a1008
- **Locked**: border #555555, text #666666

Z-Index Stack

z-index	Element
10050	Destructive Change Modal
10020	Credit Purchase Modal
10015	Cover Gallery Modal
10006	Continuation Fork Modal
10005	Edge Covenant Modal
10004	Petition Fate Modal
10002	Vault Menu Overlay
10000	Payment Modal
9000	DSP Panel, Visualization Modal

PART XX: KEY ARCHITECTURAL INVARIANTS

1. **Corridor single-row**: Exactly one corridor row is mounted in the DOM at any time
2. **Card face positioning**: `.sb-card-face` is always `position: absolute; inset: 0`
3. **No overflow:hidden on .sb-card-inner**: Breaks `transform-style: preserve-3d`
4. **Zoom portal movement**: Cards move (not clone) to portal; original position tracked
5. **Gleam synchronization**: All cards move in lockstep on 10s cycle
6. **Orchestration never called unauthorized**: `useFullOrchestration` requires `explicitEmbodimentAuthorized`
7. **Renderer isolation**: Specialist renderer receives ONLY SD content, never plot
8. **Monetization-intimacy separation**: Gates control completion/length, never authorization
9. **ChatGPT plot authority**: Only ChatGPT decides plot outcomes; Grok only embodies
10. **Breadcrumb persistence**: Breadcrumb row lives in header, never unmounted with corridor rows

---

PART XXI: KNOWN GOTCHAS

1. **CSS percentage padding** resolves against parent WIDTH, not height
2. **Multi-element corridor stages** need `flex: 0 0 auto` on titles to prevent each filling viewport
3. **Duplicate archetype handlers**: `commitArchetypeFromZoom()` AND corridor Continue handler both create breadcrumbs — must update both
4. **Corridor DOM unmount**: JS queries on unmounted rows return null — defer init to mount handler
5. **`initRotatingPlaceholder`** is NOT idempotent — guard with `!ph.innerHTML.trim()`
6. **STAGE_INDEX aliases** (archetype, intensity) must be excluded from ghost step creation
7. **`closeZoomedCard`** is declared inside `initSelectionHandlers()` — use `window.closeZoomedCard()` from module scope
8. **`overflow: hidden` on `.sb-card-inner`** forces `transform-style: flat` per CSS spec
9. **Grid-scoped CSS selectors** break in zoom portal — use `inlineCardFaceBackgrounds()` before move

10. **Base `.sb-card` has `max-width: 112px`** — zoomed cards must override to `max-width: none`

Generated from codebase analysis on February 15, 2026 Storybound App — /Users/romantsukerman/storybound-app