SiteVoice: Consolidated Product Requirements Document (PRD)

Version: 1.1

Status: Draft / Strategic Alignment

Objective: To transform SiteVoice from a passive recording tool into an autonomous AI Project Assistant and Financial Ledger.

1. Module: Integrity Core (P1 - Foundation)

1.1 The Validation Gate (Edit-Once Logic)

- **Objective:** Ensure the "Final Transcript" is human-verified while maintaining an immutable raw record.
- **Requirements:**
  - Post-recording, the creator is presented with the AI-generated summary.
  - **Logic:** Only the note creator can edit this summary.
  - **Constraint:** Edits are allowed only once. After submission, `is_edited` is set to `true` and the field becomes read-only.
  - **Data:** Raw transcript remains in `transcript_raw_original`. Edited version saves to `transcript_final`.

1.2 Actionable Inbox (Intent Classification)

- **Objective:** Reduce manager cognitive load by categorizing work.
- **Requirements:**
  - AI classifies notes into: `Approval`, `Action Required`, or `Update`.
  - **Surface Actions:**
    - *Approval:* Approve, Inquire, Deny.
    - *Action Required:* Instruct, Forward, Resolve.
    - *Update:* Acknowledge, Add Note, Log.

1.3 Proof-of-Work (PoW) Gate

- **Objective:** Eliminate "paper completions" where tasks are marked done without physical evidence.
- **Requirements:**
  - Any task in `in_progress` requires a camera-captured photo to move to `verifying`.
  - **Constraint:** The "Mark as Completed" button remains disabled until a valid `proof_photo_url` is present in the database.

2. Module: AI Assistant (Proactive Orchestration)

2.1 Autonomous Follow-ups (Task Pings)

- **Objective:** Automate the "nagging" process to keep projects moving.
- **Requirements:**
  - AI extracts a `suggested_due_date` during initial processing.
  - **Logic:** If current date > `suggested_due_date` AND status != `completed` , AI Assistant triggers a notification.

2.2 Material Arrival Tracking (Memory)

- **Objective:** Close the loop on material dependencies.
- **Requirements:**
  - AI identifies "Expected Delivery" events.
  - **Logic:** On the expected date, AI pings the Site Lead: *"Tiles for Site X were expected today. Have they arrived?"*

2.3 AI Project Blueprinting (Staging)

- **Objective:** Generate dynamic timelines from voice inputs.
- **Logic:** AI suggests a 12-stage blueprint based on project type. Team refines via voice logs.

2.4 Sequence Deviation Alerts

- **Objective:** Prevent expensive re-work caused by out-of-order execution.
- **Alert Logic:** If "Plastering" is reported before "Conduit Wiring" is verified, trigger a Critical Alert.

3. Module: Finance Management (Fiscal Integrity)

3.1 Voice Ledger (Purchases & Petty Cash)

- **Objective:** Capture site spending at the point of transaction.
- **Requirements:** AI extracts `amount` , `item` , `unit_price` , and `vendor` to a `finance_transactions` table.

3.2 Labour Payment Engine (Headcount Link)

- **Objective:** Verify wage payouts against physical presence.
- **Logic:** Flag discrepancies between "Labour Payment" logs and "Morning Headcount" reports.

4. Module: Intelligence & Reporting

4.1 Daily Pulse 2.0 (Gap Analysis)

- **Objective:** Management-by-exception summary (Blockers, Slippage, Burn).

4.2 History Querying (RAG)

- **Objective:** Natural language search for historical project memory.

5. Technical Architecture (Mobile & Voice First)

5.1 Frontend: Flutter (PWA & Native)

- **Architecture:** Feature-first folder structure (e.g., `features/voice_notes` , `features/dashboard`).
- **State Management:** Reactive Streams using Supabase `stream()` and `StreamBuilder` for real-time feed updates.
- **Voice-First UX:**
  - **Raise-to-Record:** (Future Native) Integration with proximity sensors.
  - **Audio Processing:** Local caching of audio via `path_provider` to handle intermittent site connectivity.
  - **Haptic Feedback:** Vibrations used to signify "Recording Start," "Stop," and "Upload Success."

5.2 Backend: Supabase (Serverless Intelligence)

- **Database:** PostgreSQL with Row Level Security (RLS) ensuring `manager` -only access to finance data.
- **Storage:** S3-compatible buckets for `.m4a` audio and `.jpg` PoW images.
- **Edge Functions (Deno):**
  - **Orchestration:** Triggered on Storage Upload.
  - **ASR & Translation:** Groq/Gemini for low-latency transcription and translation.
  - **Intelligence:** Structured JSON extraction using LLMs to populate `action_items` and `finance_transactions` .

5.3 AI Intelligence Pipeline

1. **Audio Capture:** Flutter records AAC/M4A -> Upload to `voice_notes` bucket.
2. **Signal:** DB Trigger -> Edge Function.
3. **ASR:** Transcribe native language (Hindi/Telugu/etc.) -> Translate to English.
4. **Classification:** Intent detection (`approval` , `update` , etc.) -> Update `voice_notes` table.
5. **Extraction:** Populate `action_items` or `finance_transactions` based on intent.

6. Technical Guardrails

- **Immutable Truth:** `transcript_raw_original` is never modified.
- **Audit Trail:** Every human edit or status change is logged in `voice_note_edits` or `interaction_history`.
- **State Machine:** Status flow is strictly `pending -> in_progress -> verifying -> completed`.
- **Privacy:** Financial data is only visible to users with the `manager` or `admin` role.