

AI Meeting Recorder — Product Requirements Document (PRD)

Document Version: 1.0.0

Status: Draft

Date: February 18, 2026

Owner: Engineering Team

Classification: Internal — Confidential

Table of Contents

1. Executive Summary
 2. Product Vision & Goals
 3. Stakeholders
 4. System Architecture Overview
 5. Tech Stack
 6. Authentication & User Management
 7. Functional Requirements
 8. Non-Functional Requirements
 9. Data Models
 10. API Specifications
 11. Security & Compliance
 12. Infrastructure & DevOps
 13. Phased Delivery Plan
 14. Risk Register
 15. Glossary
-

1. Executive Summary

AI Meeting Recorder is an enterprise-grade, AI-powered meeting intelligence platform that automatically records, transcribes, summarizes, and extracts actionable insights from meetings. It eliminates manual note-taking, ensures accountability through action item tracking, and enables teams to search and query past meeting knowledge in natural language.

The platform is designed to integrate into existing enterprise workflows (Slack, Notion, Google Calendar, email) and process audio from any source — live microphone input, uploaded files, or direct Zoom/Google Meet integrations.

Problem Statement

Organizations lose an estimated **\$37 billion per year** due to unproductive meetings. Key pain points include:

- No reliable record of decisions made
- Action items forgotten or unassigned
- New team members unable to catch up on past discussions
- Meeting time wasted re-discussing previously resolved topics

Solution

An intelligent, always-on meeting layer that turns every conversation into structured, searchable, actionable knowledge — automatically.

2. Product Vision & Goals

Vision

“Every meeting becomes a permanent, queryable, actionable knowledge asset for the organization.”

Primary Goals

Goal	Success Metric
Reduce post-meeting admin work	80% reduction in manual note-taking time
Improve action item completion	40% increase in on-time task completion
Enable meeting knowledge retrieval	Users can find any decision in < 30 seconds
Reduce redundant meetings	20% reduction in “what did we decide?” meetings

Out of Scope (v1.0)

- Real-time live translation between languages
- Video recording and storage
- Custom LLM fine-tuning per organization
- Mobile native apps (iOS/Android)

3. Stakeholders

Role	Name/Team	Responsibility
Product Owner	TBD	Feature prioritization, acceptance criteria
Engineering Lead	TBD	Architecture, technical decisions

Role	Name/Team	Responsibility
ML Engineer	TBD	AI pipeline, model selection, RAG system
Backend Engineer	TBD	FastAPI services, database, integrations
Frontend Engineer	TBD	Streamlit/React UI, dashboard
DevOps Engineer	TBD	CI/CD, cloud infrastructure, monitoring
Security Officer	TBD	Compliance, data privacy, audit
End Users	Enterprise teams	Meeting participants, admins

4. System Architecture Overview

CLIENT LAYER
Streamlit Web UI / REST API Clients

HTTPS

API GATEWAY
FastAPI (Python 3.11+)
Auth Rate Limiting Request Routing

Audio Service	AI Service	Integration Service
ffmpeg	Whisper	SendGrid
pydub	Claude API	Slack API
pyannotate	ChromaDB	Google Cal

DATA LAYER
Firebase Firestore (primary)
ChromaDB Vector Store (embeddings)
Firebase Storage (audio files)

Service Breakdown

Audio Service — Handles all audio ingestion and pre-processing. Accepts uploads or live streams, converts formats via ffmpeg, chunks large files, and passes clean audio to the AI pipeline.

AI Service — The intelligence core. Runs Whisper transcription, pyannote speaker diarization, Claude API calls for summarization and action item extraction, and manages the ChromaDB vector store for RAG-based Q&A.

Integration Service — Handles all third-party outbound integrations: SendGrid for email delivery, Slack webhooks for channel notifications, Google Calendar for meeting metadata enrichment.

5. Tech Stack

Core Services

Layer	Technology	Version	Justification
Language	Python	3.11+	AI/ML ecosystem, async support
Web Framework	FastAPI	0.110+	Async-first, auto-docs, high performance
Frontend	Streamlit → React	Latest	Fast prototyping → production upgrade
Transcription	OpenAI Whisper	large-v3	SOTA accuracy, open source, multilingual
LLM	Anthropic Claude 3.5 Sonnet	Latest	200k context window, structured output
Speaker Diarization	pyannote.audio	3.1+	State-of-the-art open-source diarization
Vector Database	ChromaDB	0.4+	Lightweight, local-first, production-ready
Database	Firebase Firestore	Latest	Serverless, real-time, scalable
File Storage	Firebase Storage	Latest	Integrated with Firestore, CDN-backed
Audio Processing	ffmpeg + pydub	Latest	Industry standard, all format support
Email	SendGrid	v3 API	Reliable, 100/day free tier, great SDK
Authentication	Firebase Auth	Latest	JWT, OAuth2, enterprise SSO ready

Development Tools

Tool	Purpose
Poetry	Python dependency management
Docker + Docker Compose	Containerization

Tool	Purpose
GitHub Actions	CI/CD pipeline
pytest	Unit and integration testing
Black + Ruff	Code formatting and linting
Pydantic v2	Data validation and serialization

Environment Configuration

```
.env.development → Local development secrets
.env.staging     → Staging environment
.env.production  → Production secrets (never committed to git)
```

All secrets managed via environment variables. No hardcoded credentials anywhere in the codebase.

6. Authentication & User Management

6.1 Overview

Authentication is handled by **Firebase Authentication** — a production-grade identity platform by Google. It issues JWTs (ID tokens) that are validated on every API request by the FastAPI backend. No passwords are stored in our database — Firebase handles all credential management securely.

6.2 Authentication Flow Diagram

Client (UI)	Firebase Auth (Google)	FastAPI Backend
----------------	---------------------------	--------------------

```

1. Login (email/
   password / OAuth)
   >

2. Returns ID Token
   (JWT, expires 1hr)
<

3. API Request +
   Bearer Token
   >

4. Verify Token
```

<

5. Token Valid +
User Claims
>

6. API Response
<

6.3 Supported Login Methods

Method	Provider	Description	Priority
Email + Password	Firebase	Standard signup/login with email verification	P0
Google OAuth	Firebase + Google	One-click login via Google account	P0
Magic Link	Firebase	Passwordless — click a link sent to email	P1
Microsoft OAuth	Firebase + Microsoft	Enterprise SSO for MS365 organizations	P1
SAML SSO	Firebase + Enterprise	For large enterprise customers with their own IdP	P2

6.4 User Registration Flow

User Signup

Enter name, email, password (min 8 chars, 1 uppercase, 1 number, 1 special)

Firebase creates auth account

Email verification sent automatically

Backend creates User document in Firestore

Backend creates Organization document (if new org)
OR joins existing org via invite token

Redirect to onboarding screen

6.5 Session Management

Property	Value
ID Token lifetime	1 hour (auto-refresh via Firebase SDK)
Refresh Token lifetime	30 days (sliding window)
Session invalidation	On password change, account disable, or explicit logout
Concurrent sessions	Allowed (multiple devices)
Force logout	Admin can revoke all sessions for a user

6.6 Role-Based Access Control (RBAC)

Every user has a role within their organization. Roles are stored in the Firestore User document and validated server-side on every request — never trusted from the client.

Role	Permissions
Owner	Full access. Manage billing, delete org, assign all roles, all admin permissions
Admin	Manage members, configure integrations, view all meetings, manage API keys
Member	Upload meetings, view all org meetings, run Q&A, send email summaries
Viewer	Read-only access to meetings and summaries. Cannot upload or run Q&A

6.7 Organization & Invite System

Organization Lifecycle:

Owner creates org

Invite members via email

- System sends invite email with unique token (expires 48 hours)
- Invitee clicks link → signs up or logs in → joins org

Assign roles to members

Member leaves or is removed

- Access revoked immediately
- Their meetings remain in org

Owner can transfer ownership to another admin

6.8 Password Policy & Security

- Minimum 8 characters, at least 1 uppercase, 1 number, 1 special character
- Password reset via email link (expires in 1 hour)
- Account locked after 10 failed login attempts (Firebase default)
- All auth events (login, logout, password change, failed attempts) written to audit log

6.9 Authentication API Endpoints

Register New User

POST /auth/register

Request:

```
{
  "display_name": "Alice Johnson",
  "email": "alice@company.com",
  "password": "SecurePass@123",
  "org_name": "Acme Corp",           // creates new org
  "invite_token": "abc123"          // OR join existing org
}
```

Response 201:

```
{
  "user_id": "user_123",
  "org_id": "org_abc",
  "message": "Verification email sent. Please verify your email."
}
```

Login

POST /auth/login

Request:

```
{
  "email": "alice@company.com",
  "password": "SecurePass@123"
}
```

Response 200:

```
{
  "id_token": "<firebase_jwt>",
  "refresh_token": "<token>",
  "expires_in": 3600,
  "user": {
    "id": "user_123",
```

```
    "email": "alice@company.com",
    "display_name": "Alice Johnson",
    "role": "admin",
    "org_id": "org_abc"
  }
}
```

Refresh Token

POST /auth/refresh

Request:

```
{
  "refresh_token": "<token>"
}
```

Response 200:

```
{
  "id_token": "<new_firebase_jwt>",
  "expires_in": 3600
}
```

Logout

POST /auth/logout

Authorization: Bearer <id_token>

Response 200:

```
{
  "message": "Logged out successfully"
}
```

Reset Password

POST /auth/reset-password

Request:

```
{
  "email": "alice@company.com"
}
```

Response 200:

```
{
  "message": "Password reset link sent to alice@company.com"
}
```

Invite Member

POST /org/members/invite
Authorization: Bearer <id_token> (Admin or Owner only)

Request:
{
 "email": "bob@company.com",
 "role": "member"
}

Response 200:
{
 "message": "Invite sent to bob@company.com",
 "invite_token": "xyz789",
 "expires_at": "2026-02-20T10:00:00Z"
}

Get Current User Profile

GET /auth/me
Authorization: Bearer <id_token>

Response 200:
{
 "id": "user_123",
 "email": "alice@company.com",
 "display_name": "Alice Johnson",
 "role": "admin",
 "org_id": "org_abc",
 "org_name": "Acme Corp",
 "email_verified": true,
 "created_at": "2026-01-01T00:00:00Z",
 "last_login": "2026-02-18T09:00:00Z"
}

6.10 Frontend Auth Screens

Screen	Route	Description
Login	/login	Email/password + Google OAuth button
Register	/register	Signup form + org creation or invite token
Verify Email	/verify-email	Prompt to check inbox, resend option
Forgot Password	/forgot-password	Email input → trigger reset link

Screen	Route	Description
Reset Password	<code>/reset-password?token=xyz</code>	Reset password form
Onboarding	<code>/onboarding</code>	First-time user setup (name, preferences)
Profile Settings	<code>/settings/profile</code>	Update name, email, password, avatar
Org Settings	<code>/settings/org</code>	Manage members, roles, integrations (Admin only)

6.11 Auth Data Model (Firestore)

Collection: `users/{user_id}`

```
{
  "id": "user_123",
  "firebase_uid": "firebase_abc123",      // links to Firebase Auth
  "email": "alice@company.com",
  "display_name": "Alice Johnson",
  "avatar_url": "https://...",
  "org_id": "org_abc",
  "role": "admin",                        // owner | admin | member | viewer
  "email_verified": true,
  "is_active": true,
  "preferences": {
    "email_notifications": true,
    "slack_notifications": false,
    "default_language": "en",
    "timezone": "America/New_York"
  },
  "created_at": "2026-01-01T00:00:00Z",
  "last_login": "2026-02-18T09:00:00Z"
}
```

Collection: `organizations/{org_id}`

```
{
  "id": "org_abc",
  "name": "Acme Corp",
  "owner_id": "user_123",
  "plan": "free",                        // free | pro | enterprise
  "member_count": 12,
  "integrations": {
    "slack_webhook": "https://hooks.slack.com/...",
    "notion_api_key": null,
    "sendgrid_verified": true
  }
}
```

```

    },
    "limits": {
        "monthly_minutes": 600,           // 10 hrs/month on free tier
        "minutes_used": 142,
        "max_members": 5
    },
    "created_at": "2026-01-01T00:00:00Z"
}

Collection: invites/{invite_token}

{
    "token": "xyz789",
    "org_id": "org_abc",
    "email": "bob@company.com",
    "role": "member",
    "invited_by": "user_123",
    "status": "pending",                 // pending | accepted | expired
    "expires_at": "2026-02-20T10:00:00Z",
    "created_at": "2026-02-18T10:00:00Z"
}

```

7. Functional Requirements

7.1 Audio Ingestion

ID	Requirement	Priority
FR-001	System shall accept audio file uploads in MP3, MP4, WAV, M4A, OGG, WEBM formats	P0
FR-002	System shall support live microphone recording via browser	P0
FR-003	System shall accept files up to 500MB in size	P0
FR-004	System shall convert all formats to 16kHz mono WAV before processing	P0
FR-005	System shall split audio files longer than 30 minutes into overlapping chunks	P1
FR-006	System shall support Zoom recording URL ingestion	P2

FR-007	System shall provide upload progress indicator to the user	P1
--------	--	----

7.2 Transcription

ID	Requirement	Priority
FR-010	System shall transcribe audio to text with >95% accuracy for clear audio	P0
FR-011	System shall include timestamps for each utterance	P0
FR-012	System shall support transcription in English, Hindi, Spanish, French, German	P1
FR-013	System shall auto-detect spoken language	P1
FR-014	System shall handle multiple simultaneous speakers	P0
FR-015	System shall label speakers as Speaker 1, Speaker 2 etc. by default	P0
FR-016	System shall allow users to rename speaker labels	P1

7.3 AI Analysis

ID	Requirement	Priority
FR-020	System shall generate a structured meeting summary (max 500 words)	P0
FR-021	System shall extract all action items with assignee and deadline if mentioned	P0
FR-022	System shall identify key decisions made during the meeting	P0
FR-023	System shall extract open questions that remain unresolved	P1
FR-024	System shall generate a list of topics discussed with time ranges	P1
FR-025	System shall perform sentiment analysis on overall meeting tone	P2
FR-026	System shall identify the most engaged and least engaged speakers	P2

ID	Requirement	Priority
FR-027	System shall auto-tag meetings with relevant topic keywords	P1

7.4 Q&A on Meeting (RAG)

ID	Requirement	Priority
FR-030	System shall allow users to ask natural language questions about any meeting	P0
FR-031	System shall return answers with transcript references (timestamps)	P0
FR-032	System shall allow cross-meeting queries (e.g., “What was decided about X across all meetings?”)	P1
FR-033	System shall indicate confidence level of answers	P1
FR-034	System shall support follow-up questions in a conversational interface	P2

7.5 Meeting Dashboard

ID	Requirement	Priority
FR-040	System shall display all past meetings in a searchable list	P0
FR-041	System shall show summary, action items, and transcript for each meeting	P0
FR-042	System shall allow full-text search across all transcripts	P0
FR-043	System shall filter meetings by date, tags, participants, duration	P1
FR-044	System shall display meeting analytics (avg duration, topics over time)	P2

7.6 Integrations

ID	Requirement	Priority
FR-050	System shall send email summary to specified recipients after processing	P0
FR-051	System shall post summary to a configured Slack channel	P1
FR-052	System shall push action items to a Notion database	P2
FR-053	System shall sync with Google Calendar to auto-fetch meeting metadata	P2

7.7 User & Organization Management

ID	Requirement	Priority
FR-060	System shall support user registration and login via email/password	P0
FR-061	System shall support Google OAuth login	P1
FR-062	System shall support multi-tenant organizations (each org sees only their meetings)	P0
FR-063	System shall support admin and member roles within an organization	P1
FR-064	System shall allow admins to manage API key settings and integrations	P1

8. Non-Functional Requirements

8.1 Performance

ID	Requirement	Target
NFR-001	Transcription processing time	< 0.5x real-time (30 min meeting → 15 min processing)
NFR-002	Summary generation latency	< 30 seconds after transcription

ID	Requirement	Target
NFR-003	Q&A response time	< 5 seconds per query
NFR-004	Dashboard page load time	< 2 seconds
NFR-005	API endpoint response time (non-AI)	< 200ms at P95

8.2 Scalability

ID	Requirement
NFR-010	System shall handle 100 concurrent audio processing jobs
NFR-011	System shall support up to 10,000 organizations
NFR-012	System shall scale horizontally via containerization
NFR-013	Audio processing shall be queued and processed asynchronously

8.3 Availability & Reliability

ID	Requirement
NFR-020	System uptime SLA: 99.9% (< 8.7 hours downtime/year)
NFR-021	Graceful degradation — if LLM API fails, raw transcript still delivered
NFR-022	All audio jobs shall be retried up to 3 times on failure
NFR-023	Failed jobs shall alert the user and log to monitoring

8.4 Security

ID	Requirement
NFR-030	All data at rest encrypted (AES-256)
NFR-031	All data in transit encrypted (TLS 1.3)
NFR-032	Audio files deleted from processing servers after analysis
NFR-033	API keys rotated every 90 days
NFR-034	All user actions logged in audit trail
NFR-035	Role-based access control (RBAC) enforced at API level

8.5 Observability

ID	Requirement
NFR-040	Structured JSON logging for all services
NFR-041	Request tracing with correlation IDs across services

ID	Requirement
NFR-042	Metrics exposed for: processing time, error rate, LLM token usage, costs
NFR-043	Alerting on error rate > 1% or processing queue depth > 50

9. Data Models

9.1 Meeting Document (Firestore)

Collection: organizations/{org_id}/meetings/{meeting_id}

```
{
  "id": "mtg_01HXYZ123",
  "org_id": "org_abc",
  "created_by": "user_123",
  "title": "Q3 Planning Session",
  "date": "2026-02-18T10:00:00Z",
  "duration_seconds": 2700,
  "participants": ["alice@company.com", "bob@company.com"],
  "audio_file_url": "gs://bucket/org_abc/meetings/mtg_01HXYZ123.wav",
  "status": "completed",          // pending | processing | completed | failed
  "language": "en",
  "tags": ["planning", "q3", "budget"],
  "sentiment_score": 0.72,        // 0 (negative) to 1 (positive)
  "created_at": "2026-02-18T10:45:00Z",
  "updated_at": "2026-02-18T11:00:00Z"
}
```

9.2 Transcript Document (Firestore)

Collection: organizations/{org_id}/meetings/{meeting_id}/transcript

```
{
  "meeting_id": "mtg_01HXYZ123",
  "full_text": "Speaker 1: Good morning everyone...",
  "utterances": [
    {
      "speaker": "Speaker 1",
      "speaker_label": "Alice",          // user-assigned name
      "text": "Good morning everyone, let's begin.",
      "start_time": 0.0,
      "end_time": 3.4,
      "confidence": 0.97
    }
  ]
}
```

```

    }
  ],
  "word_count": 3420,
  "created_at": "2026-02-18T10:50:00Z"
}

```

9.3 Analysis Document (Firestore)

Collection: organizations/{org_id}/meetings/{meeting_id}/analysis

```

{
  "meeting_id": "mtg_01HXYZ123",
  "summary": "The team discussed Q3 planning priorities...",
  "action_items": [
    {
      "id": "act_001",
      "text": "Send updated budget proposal to finance team",
      "assignee": "Alice",
      "due_date": "2026-02-25",
      "completed": false
    }
  ],
  "decisions": [
    {
      "text": "Agreed to push product launch to Q4",
      "timestamp": 1240.0
    }
  ],
  "open_questions": [
    "What is the final headcount budget for Q3?"
  ],
  "topics": [
    { "topic": "Budget Review", "start_time": 0, "end_time": 600 },
    { "topic": "Hiring Plan", "start_time": 600, "end_time": 1800 }
  ],
  "model_used": "claude-3-5-sonnet-20241022",
  "tokens_used": 4821,
  "processing_time_seconds": 18.4,
  "created_at": "2026-02-18T10:55:00Z"
}

```

9.4 User Document (Firestore)

Collection: users/{user_id}

```

{

```

```

    "id": "user_123",
    "email": "alice@company.com",
    "display_name": "Alice Johnson",
    "org_id": "org_abc",
    "role": "admin", // admin | member | viewer
    "preferences": {
      "email_notifications": true,
      "slack_notifications": false,
      "default_language": "en"
    },
    "created_at": "2026-01-01T00:00:00Z",
    "last_login": "2026-02-18T09:00:00Z"
  }
}

```

10. API Specifications

Base URL

Production: <https://api.aimeetingrecorder.com/v1>
 Staging: <https://staging-api.aimeetingrecorder.com/v1>
 Development: <http://localhost:8000/v1>

Authentication

All endpoints require a Bearer JWT token in the Authorization header:

Authorization: Bearer <firebase_id_token>

Core Endpoints

Upload & Process Meeting

POST /meetings/upload

Request:

Content-Type: multipart/form-data

Body:

file: <audio_file>	(required)
title: string	(optional)
participants: string[]	(optional)
language: string	(optional, default: "auto")

Response 202 Accepted:

```

{
  "meeting_id": "mtg_01HXYZ123",
  "status": "processing",

```

```
    "estimated_completion_seconds": 180,  
    "status_url": "/meetings/mtg_01HXYZ123/status"  
  }  
}
```

Get Meeting Status

GET /meetings/{meeting_id}/status

Response 200:

```
{  
  "meeting_id": "mtg_01HXYZ123",  
  "status": "completed",  
  "progress_percent": 100,  
  "stages": {  
    "audio_processing": "completed",  
    "transcription": "completed",  
    "diarization": "completed",  
    "analysis": "completed"  
  }  
}
```

Get Meeting Details

GET /meetings/{meeting_id}

Response 200:

```
{  
  "meeting": { ...meeting document },  
  "transcript": { ...transcript document },  
  "analysis": { ...analysis document }  
}
```

List Meetings

GET /meetings?page=1&limit=20&from_date=2026-01-01&tags=planning

Response 200:

```
{  
  "meetings": [...],  
  "total": 142,  
  "page": 1,  
  "limit": 20  
}
```

Q&A Query

POST /meetings/{meeting_id}/query

```
Request:
{
  "question": "What did we decide about the budget?",
  "conversation_history": []    // optional for follow-up questions
}
```

```
Response 200:
{
  "answer": "The team decided to increase the Q3 budget by 15%...",
  "sources": [
    {
      "speaker": "Alice",
      "text": "I think we should increase the budget by 15%...",
      "timestamp": 842.0,
      "confidence": 0.92
    }
  ],
  "tokens_used": 1240
}
```

Send Email Summary

POST /meetings/{meeting_id}/send-email

```
Request:
{
  "recipients": ["bob@company.com", "carol@company.com"],
  "include_transcript": false,
  "custom_message": "FYI - summary from today's call"
}
```

```
Response 200:
{
  "status": "sent",
  "recipients_count": 2
}
```

11. Security & Compliance

Authentication & Authorization

Firebase Authentication handles user identity. All API requests validated via JWT tokens. Role-based access control enforced at the service layer — members can only access meetings within their organization. Admins can manage org

settings, integrations, and all members' meetings.

Data Privacy

Audio files are stored encrypted in Firebase Storage and deleted from processing servers within 1 hour of job completion. Transcripts and analysis are stored in Firestore with organization-level isolation. Users can request full data deletion at any time (GDPR Right to Erasure).

Compliance Considerations

Standard	Status	Notes
GDPR	Required	Data deletion, export APIs, consent tracking
SOC 2 Type II	Target (v2.0)	Audit logging, access controls
HIPAA	Not in scope v1	Future consideration for healthcare clients
ISO 27001	Target (v2.0)	Information security management

API Security

All endpoints protected against SQL injection (not applicable — NoSQL), XSS via input sanitization, CSRF via SameSite cookie policy, and rate limiting (100 req/min per user, 10 audio uploads/day on free tier).

12. Infrastructure & DevOps

Environments

Development → Local Docker Compose
Staging → Google Cloud Run (single replica)
Production → Google Cloud Run (auto-scaling, min 2 replicas)

CI/CD Pipeline (GitHub Actions)

Push to feature branch
→ Run tests (pytest)
→ Lint & format check (Black, Ruff)
→ Build Docker image

```

Merge to main
  → All above +
  → Deploy to Staging
  → Run smoke tests

Tag release (v*.*.*)
  → All above +
  → Deploy to Production
  → Notify team on Slack

```

Monitoring Stack

Tool	Purpose
Google Cloud Monitoring	Infrastructure metrics, uptime
Sentry	Error tracking and alerting
Firebase Analytics	User behavior and feature usage
Custom dashboard	LLM costs, token usage, processing times

Cost Management

LLM API costs tracked per organization and per meeting. Budget alerts configured at \$50/day. Token usage logged in every analysis document for cost attribution and optimization.

13. Phased Delivery Plan

Phase 1 — Core Pipeline (Week 1-2)

Goal: End-to-end working pipeline, no UI

- Audio upload and ffmpeg preprocessing
- Whisper transcription working locally
- Basic summary and action item extraction via Claude API
- Save results to Firestore
- Tested via Postman / curl

Deliverable: Upload an audio file, get a transcript + summary in Firestore

Phase 2 — Speaker Intelligence (Week 3)

Goal: Know who said what

- Integrate pyannote.audio for speaker diarization
- Merge diarization output with Whisper transcript

- Speaker renaming API endpoint
- Sentiment analysis per speaker

Deliverable: Transcript with “Alice: ...”, “Bob: ...” labels

Phase 3 — Web Dashboard (Week 4)

Goal: Something you can actually show people

- Streamlit frontend
- Meeting list view
- Meeting detail page (summary, action items, full transcript)
- Firebase Auth login
- Basic search

Deliverable: A beautiful, working web app

Phase 4 — RAG Q&A System (Week 5-6)

Goal: Ask anything about any meeting

- Chunk transcripts and generate embeddings
- Store in ChromaDB
- Q&A endpoint with Claude
- Chat interface in Streamlit
- Cross-meeting search

Deliverable: “What did we decide about X?” → instant answer with timestamp

Phase 5 — Integrations & Polish (Week 7-8)

Goal: Production-ready, interview-worthy

- SendGrid email summaries
- Slack webhook notifications
- Multi-language transcription
- Organization multi-tenancy
- Docker containerization
- GitHub Actions CI/CD
- README and architecture diagram

Deliverable: Fully deployable, enterprise-grade product

14. Risk Register

ID	Risk	Likelihood	Impact	Mitigation
R-001	Whisper accuracy poor for accented speech	Medium	High	Test with diverse audio, fallback to AssemblyAI
R-002	Claude API rate limits hit during load	Low	High	Implement request queuing, exponential backoff
R-003	pyannote diarization slow on long meetings	High	Medium	Run diarization async, show transcript first
R-004	Firebase costs exceed budget	Low	Medium	Set billing alerts, implement caching
R-005	Audio file too large for processing	Medium	Medium	Enforce 500MB limit, chunk files > 30 min
R-006	LLM hallucination in action item extraction	Medium	High	Validate output with Pydantic, prompt engineering
R-007	GDPR compliance gap	Low	Critical	Legal review before enterprise launch
R-008	Speaker diarization confuses similar voices	High	Medium	Allow manual correction in UI

15. Glossary

Term	Definition
ASR	Automatic Speech Recognition — converting spoken audio to text
Diarization	The process of segmenting audio by speaker identity (“who spoke when”)
RAG	Retrieval-Augmented Generation — technique of fetching relevant context before prompting an LLM
Embeddings	Numerical vector representations of text that capture semantic meaning
Vector Database	A database optimized for storing and searching embeddings by similarity
LLM	Large Language Model — AI model trained on text (e.g., Claude, GPT-4)
Utterance	A single continuous speech segment from one speaker
Chunking	Splitting large text/audio into smaller overlapping pieces for processing
RBAC	Role-Based Access Control — permission system based on user roles
Multi-tenancy	Architecture where one system serves multiple isolated organizations
JWT	JSON Web Token — secure token format used for authentication
SLA	Service Level Agreement — contractual uptime/performance commitment
P0 / P1 / P2	Priority levels: P0 = must have, P1 = should have, P2 = nice to have

Document maintained by Engineering Team. For questions or updates, open a PR against this document in the project repository.

Last updated: February 18, 2026 — v1.0.0