AI Meeting Assistant — Build Plan (Agentic-RAG MVP → V1)

A proactive, Jarvis-like co-pilot for Zoom/Teams/Meet that listens, understands, retrieves context, and takes action with strong privacy controls.

---

0) Product Goals & Success Criteria

Primary outcomes - Cut note-taking and follow-ups by ≥50% via automated summaries, tasks, and drafts. - Improve meeting quality: fewer attendees, clearer decisions, measurable action completion. - Enterprise-ready privacy: configurable data retention, local/offline modes, granular consent.

Key metrics (north-stars) - Time saved / user / week (target: 3–5h). - Action precision@1 (correct next action) ≥85% (human-approved). - Summary satisfaction CSAT ≥4.5/5. - WER (ASR) ≤12% on our meeting domain.

---

1) User Journeys (MVP scope)

1. **Auto-Join & Transcribe**
2. Bot joins Zoom/Teams/Meet when calendar says “record” or the user toggles “AI on”.
3. Real-time transcript with speaker diarization.
4. **Live Companion Panel** (web app / desktop overlay)
5. Shows agenda, facts fetched via RAG, suggested questions, detected decisions.
6. One-click: “Create task”, “Draft email”, “Log to CRM”.
7. **Post-Meeting Package**
8. Clean summary (objectives → decisions → action items with owners & dates).
9. Artifacts pushed to Jira/Asana/Slack/Email/CRM as configured.
10. **Privacy Modes**
11. *Notes-Only*, *Transcript+Notes*, *Strict Off-the-Record* (no storage; ephemeral processing).

---

2) System Architecture (high level)

Clients - **Meeting bot**: Zoom App + Teams Bot + Chrome extension for Google Meet. - **Companion UI**: Electron (desktop) or web (Next.js) overlay; shows live suggestions & approvals.

Backend services (FastAPI + Python) - **Ingestion gateway** (WebSocket): audio/video frames, metadata. - **ASR & Diarization**: WhisperX (GPU) + Pyannote; optional Krisp/RTX noise-suppression client-side. - **NLU pipeline**: turn segmentation, intents, entities, sentiment; topic & decision detectors. - **RAG service**: connectors → chunking → embeddings → vector DB → retrieval → re-ranking. - **Agentic Orchestrator**: tool-use planner/executor with human-in-the-loop policies. - **Action Integrations**: Gmail/Outlook, Google/ Microsoft Calendar, Slack, Jira, Asana, Notion, Salesforce. - **Storage**: Postgres (OLTP), Object store (S3/Cloud Storage) for media, Vector DB (pgvector/Milvus), Redis for real-time state. - **Event bus**: Kafka/Redpanda; async workers (Celery/RQ) for heavy jobs. - **Policy & Audit**: consent, retention, redaction, action approvals; immutable audit log.

Observability - Prometheus/Grafana + OpenTelemetry traces; label by meeting, account, feature flags.

---

3) Tech Stack (MVP picks)

- **Frontend**: Next.js + React (Companion), Tailwind, tRPC/REST; Electron wrapper for always-on overlay.
- **Bots**: Zoom Meeting SDK, MS Teams Bot Framework, Chrome Extension (Meet caption tap).
- **ASR**: OpenAI Whisper large-v3-turbo (or WhisperX) with VAD; diarization via Pyannote.
- **Embeddings**: bge-m3 or text-embedding-3-large (multilingual); rerank: Cohere Rerank or bge-reranker.
- **LLM**: GPT-4.1/GPT-4o-mini for reasoning; fallback open-weights (Llama 3.1 70B) for offline mode.
- **DB**: Postgres 16 + pgvector; Redis 7; S3-compatible store (MinIO in dev, AWS S3 in prod).
- **Infra**: Docker Compose (dev), Kubernetes + ArgoCD (prod), NGINX Ingress, cert-manager, Vault for secrets.

---

4) Data Flow (end-to-end)

1. **Join** → Calendar webhook schedules bot; auth via OAuth2 per provider.
2. **Stream** → Audio via WebRTC → Ingestion WS → ASR chunks (partial + final) → diarization.
3. **Understand** → NLU detects intents (decision, question, date/owner), entities (people, orgs), sentiment.
4. **Retrieve** → Agent crafts queries → RAG pulls docs (emails, CRM notes, wiki) with scoped permissions.
5. **Propose** → Companion UI shows suggestions (questions, facts, draft notes) → user approves or ignores.
6. **Act** → With approval or policy auto-approve, tools execute (create Jira, send draft mail, update CRM).
7. **Deliver** → Summary + tasks + clips shipped to destinations; analytics logged.

---

5) Privacy, Security, Compliance

- **Modes**: Notes-Only / Transcript+Notes / Off-the-Record (ephemeral buffers; no store).
- **PII Controls**: live redaction (names, emails, numbers) before storage; reversible under DSR (GDPR/CCPA).
- **Data Residency**: region-pin storage; per-tenant encryption keys (envelope via KMS).
- **Access**: SSO/SAML, SCIM, RBAC (owner, participant, auditor). Admin policy: allowed tools, retention windows.
- **Audit**: every action/tool-call logged with prompt, retrieved docs hash, decision reason, approver.

---

6) MVP Feature Spec (6–8 sprints, prioritize left → right)

A. Ingestion & ASR - WS ingestion, chunked ASR, speaker tags, timestamps, confidence.

B. Companion UI - Live transcript, highlights, suggestions tray (ask, fact, task), one-click approvals.

C. RAG v1 - Connectors: Gmail, Google Drive, Notion; chunking (semantic), embeddings, top-k retrieve + rerank.

D. Agentic Actions v1 - Tools: Email draft, Calendar draft, Jira ticket, Slack summary; human-approval required.

E. Post-Meeting - Auto summary (agenda → key points → decisions → actions). Export to Slack/Email/Notion page.

F. Privacy & Admin - Per-meeting mode, retention settings, export/delete, basic audit log.

---

7) Data Models (Postgres)

```sql
-- meetings
CREATE TABLE meetings (
  id UUID PRIMARY KEY, account_id UUID, platform TEXT, title TEXT,
  start_ts TIMESTAMPTZ, end_ts TIMESTAMPTZ, privacy_mode TEXT,
  created_at TIMESTAMPTZ DEFAULT now()
);

-- utterances
CREATE TABLE utterances (
  id BIGSERIAL PRIMARY KEY, meeting_id UUID REFERENCES meetings(id),
  speaker TEXT, start_ms INT, end_ms INT, text TEXT, conf REAL
);

-- suggestions (live agent prompts)
```

```
CREATE TABLE suggestions (
  id UUID PRIMARY KEY, meeting_id UUID, kind TEXT, payload JSONB,
  created_at TIMESTAMPTZ DEFAULT now(), approved_by UUID, status TEXT
);

-- actions (tool executions)
CREATE TABLE actions (
  id UUID PRIMARY KEY, meeting_id UUID, tool TEXT, input JSONB,
  output JSONB, status TEXT, approved_by UUID, created_at TIMESTAMPTZ DEFAULT
now()
);
```

Vector store: `documents(doc_id UUID, tenant_id UUID, source TEXT, text TEXT, meta JSONB, embedding VECTOR(1536))`.

---

8) Connectors (OAuth2 scopes & least-privilege)

- **Gmail/Outlook**: read metadata + read/draft (no send w/o approval).
- **Drive/OneDrive/Notion**: read within shared folders/spaces.
- **Salesforce/HubSpot**: read accounts/opportunities, write limited fields (notes, next step) post-approval.
- **Slack/Jira/Asana**: post messages, create issues/tasks with labels & links back to meeting.

---

9) Agentic Tooling & Guardrails

Tool set (v1): `search_docs`, `summarize`, `compose_email`, `create_ticket`, `schedule_event`, `log_crm_note`.

Policies - Required approval for any external write. - Source-attribution: every suggestion shows citations. - Rate limits & cooldowns per meeting to avoid spam.

Loop (pseudocode)

```
while meeting_active:
    obs = gather_signals(utterances, nlu, timeline)
    if should_retrieve(obs):
        ctx = rag.retrieve(obs.query)
    intent = decide_intent(obs, ctx)
    if intent in WRITE_ACTIONS:
        suggest(intent, ctx)
    elif intent == QUESTION_GAP:
        suggest_question(ctx)
```

10) Example Service Stubs (Python/FastAPI)

WebSocket Ingestion

```
@app.websocket("/ws/audio")
async def ingest(ws: WebSocket):
    await ws.accept()
    buffer = AudioBuffer()
    async for chunk in ws.iter_bytes():
        segments = asr.feed(chunk)  # yields partial/final
        for s in segments:
            diar = diarize(s.audio)
            store_utterance(s.text, diar.speaker, s.ts)
            enqueue_nlu(s.text)
            push_to_ui(s)
```

RAG Query

```
def rag_query(query, user_id, tenant_id):
    hits = vectordb.search(embed(query), k=8, filter={"tenant_id": tenant_id})
    reranked = rerank(query, hits)
    return format_context(reranked[:4])
```

Suggestion Envelope

```
suggestion = {
  "kind": "ask",
  "text": "Ask about Q3 budget alignment?",
  "reasons": ["budget gap mentioned", "client renewal in 45 days"],
  "citations": [doc_url_1, doc_url_2]
}
```

11) Companion UI (outline)

- **Live panel**: transcript, highlights, approvals, quick actions.
- **Citations popover**: shows sources used for a suggestion.
- **Privacy banner**: current mode; toggle + participants consent status.
- **Post-meeting page**: editable summary, actions checklist, export buttons.

12) Evaluation Plan

- **ASR**: WER by domain (sales/standup/support); collect gold transcripts from 20 seed meetings.
- **Summaries**: human rubric (coverage, correctness, clarity) 1–5; target ≥4.5.
- **Actions**: precision/recall on suggested vs accepted vs executed.
- **RAG**: grounding score (judge if answer is supported by retrieved docs) ≥0.9.

---

13) Deployment & Environments

- **Dev**: Docker Compose (ingestion, ASR, API, DB, vector, UI).
- **Staging**: K8s single-AZ, small GPU node for ASR; feature flags.
- **Prod**: Multi-AZ K8s, autoscaling ASR workers; regional S3 buckets.
- **Secrets**: Vault + cloud KMS; short-lived tokens for connectors.

---

14) Privacy UX & Legal

- Consent prompts on join; per-participant opt-out respected (mute their channel; exclude from storage).
- DSR: export/delete per user; retention default 30 days (admin-configurable).
- DPIA templates & SOC2 roadmap; pen-test before enterprise pilots.

---

15) Backlog (prioritized)

1. Zoom bot + WS ingestion + WhisperX + diarization.
2. Companion live panel (transcript, suggestions).
3. RAG v1 (Gmail, Drive, Notion) + citations.
4. Actions v1 (email, calendar, Jira, Slack) with approvals.
5. Post-meeting summary + exports.
6. Privacy modes + admin console + audit log.
7. MS Teams bot + Salesforce connector.
8. Screen-share OCR (charts/tables) + visual Q&A.
9. Mobile app (read-only summaries, approvals by voice).

---

16) Risks & Mitigations

- **Latency**: streaming ASR + incremental reasoning; cache embeddings; push vs poll.
- **Hallucinations**: strict retrieval-augmented prompting; abstain when uncertain; always show citations.
- **Privacy**: opt-in defaults; local/offline mode; redaction; narrow scopes.
- **Adoption**: clear ROI dashboards; easy admin rollout; templates per function (sales, support, eng).

---

17) V1 Differentiators (beyond MVP)

- **Proactive meeting optimizer** (pre-meeting agenda, attendee slimming, conflict alerts).
- **Multimodal insights** (screen-share understanding; chart callouts).
- **Personalized coaching** (talk-time balance, objection handling, empathy cues).
- **No-code agent builder** (department-specific skills with governed tool-sets).

---

18) Next Actions (you can start today)

- Provision repo with these services: `ingestion`, `asr`, `nlu`, `rag`, `agent`, `integrations`, `ui`.
- Scaffold FastAPI + Postgres + pgvector + Redis in Docker Compose.
- Implement WS ingestion + WhisperX pipeline; render live transcript in UI.
- Build Suggestions tray: accept/decline; log audit events.
- Wire Gmail/Drive/Notion OAuth; index user-shared folders; enable `search_docs` tool.
- Ship Post-Meeting summary export to Slack/Email.

Appendix: Prompting & Templates - System prompts for: summarizer, action-planner, email-drafter, ticket-writer. - Retrieval prompts enforce citations and abstention when unsupported.

Appendix: Testing Fixtures - Synthetic meetings: sales demo, standup, support triage; gold summaries & actions. - Load test: 50 concurrent meetings, 2h each; ASR autoscaling checks.