Data Annotation Platform

Project Execution Proposal

Version: 3.0

Date: February 2026

Prepared by: Tech Lead / Solution Architect

Status: Proposal for Client Review

Classification: Confidential

This proposal is a direct execution plan against PRD v2.0 requirements. All deliverables are mapped to PRD acceptance criteria to ensure contractual compliance.

Table of Contents

1. Executive Summary

We will build the Data Annotation Platform from scratch with a lean, senior 4-person team. Every team member is a senior engineer who owns full vertical slices -- no hand-offs, no coordination overhead, no waiting on dependencies. Four people who each deliver end-to-end, moving fast with high ownership.

Phase	Duration	Milestone
Phase 0 -- Discovery & Setup	3 weeks (W1-W3)	Infra live, architecture approved, portals scaffolded
Phase 1 -- MVP	8 weeks (W4-W11)	Production-ready internal annotation platform
Phase 2 -- Enterprise	10 weeks (W12-W21)	Consensus, benchmarks, linters, Okta SSO, customer portal
Total (Phases 0-2)	21 weeks	Complete enterprise platform in production
Phase 3 -- Optional	TBD (W22+)	SOW-dependent enhancements

Why a 4-Person Team Works

- **Zero management overhead** — The Tech Lead is simultaneously the PM, architect, and hands-on coder. Decisions are made in real time, eliminating layers of coordination.
- **No cross-team delays** — One standup, one backlog, one codebase. Work flows seamlessly without hand-offs or dependency bottlenecks.
- **Senior-only execution** — Every engineer has 5+ years of experience. Senior contributors deliver 3–5x faster than juniors, ensuring velocity without sacrificing quality.
- **Modern tooling advantage** — Frameworks like *shadcn/ui*, Prisma, and Turborepo remove weeks of boilerplate, enabling the team to focus on business logic and user experience.
- **Focused timelines** — With fewer people, each engineer owns deep vertical slices. This allows for thorough, high-quality implementation instead of rushed, fragmented delivery.

Result: The client receives the same enterprise-grade product defined in the PRD — but built by a smaller, sharper team with tighter ownership, lower cost, and higher accountability.

2. Our Understanding of the Requirement

The Data Annotation Platform is an internal, end-to-end system designed to centralize queueing, task workflows, annotation interfaces, operational tooling, quality processes, and batch-level data export. Today, annotation workflows are distributed across multiple tools and manual processes; this platform consolidates them into a unified environment.

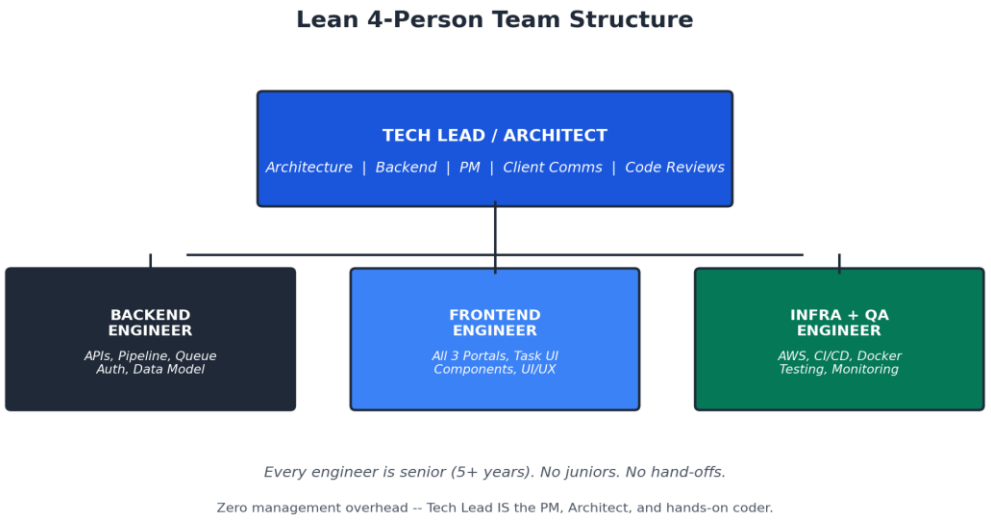
Core Capabilities

- Configurable, multi-layer annotation and review pipelines (L1, L2, Review, Hold, Archive)
- Schema-driven task UIs supporting multiple data types (text, image, audio, video, HTML/Markdown)
- FIFO queueing with deterministic claim locking (zero double-claims)
- Rater and reviewer task interfaces for completing and validating annotations
- Operational tools for task upload, export, tagging, and pipeline management
- Benchmarks, consensus mechanisms, linters, and time-tracking
- Secure storage, access control, and full audit logging
- Workspace -> Project -> Batch -> Task -> Annotation data hierarchy
- Customer-facing portal for seat management, API keys, and usage visibility

Confirmed Scale Requirements

Metric	Confirmed Target
Daily task volume	100,000 avg / 150,000 peak
Concurrent rater sessions	500 avg / 800 peak
Max media file size	500 MB per asset
Max batch upload size	No limit (streaming required)
API response time (p95)	< 300ms
UI load time (p95)	< 2 seconds
Claim collision rate	0% (deterministic locking)

3. Proposed Team Structure



#	Role	Key Skills	What They Own
P1	Tech Lead / Architect	Node.js, PostgreSQL, MongoDB, system design, AWS, leadership	Architecture, DB design, complex backend (claim locking, pipeline), code reviews, sprint planning, client demos. Writes 40% backend code.
P2	Sr. Backend Engineer	Node.js/Express, PostgreSQL, MongoDB, REST APIs, concurrency	All CRUD APIs, auth system, queue engine, assignments, export, upload, linters, audit. Writes 60% backend code.
P3	Sr. Frontend Engineer	React, TypeScript, Tailwind, shadcn/ui, media rendering, forms	All three portals (Ops, Rater, Customer), Task UI with media, schema-driven forms, component library. Owns all UI/UX.
P4	Infra + QA Engineer	AWS, Terraform, GitHub Actions, Docker, Playwright, k6, Datadog	AWS environments, CI/CD, Docker, deployment, monitoring, E2E tests, API tests, load tests, security. S3 integration.

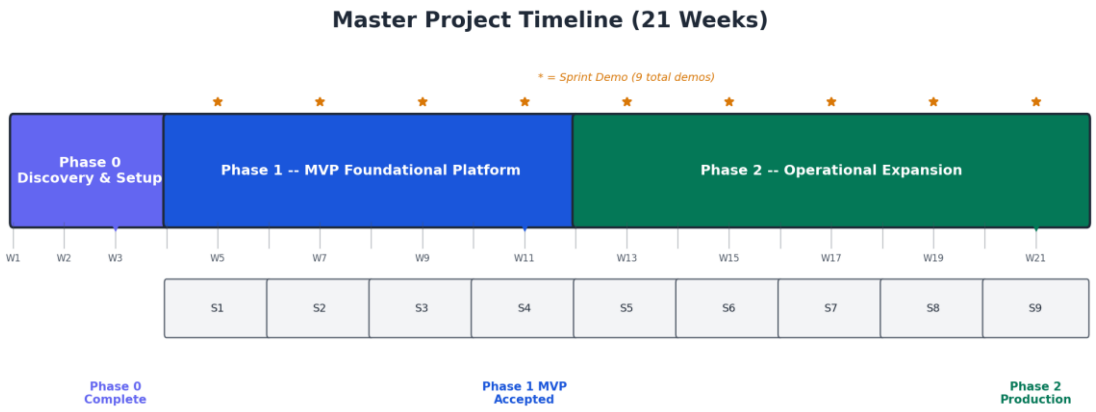
Why This Composition Works

Traditional (11 people)	Our Approach (4 people)
Dedicated PM	Tech Lead handles planning + client comms
Dedicated UI/UX Designer	Frontend Eng uses shadcn/ui (professional design system)
2 Backend + 2 Frontend + 1 Full-Stack	1 Backend + 1 Frontend + Tech Lead (both)
Dedicated QA	Infra Eng handles test automation; everyone writes unit tests
Dedicated DevOps	Infra + QA combined (both about system reliability)
Coordination meetings, hand-off docs	One Slack channel, one standup, PRs are the docs

Hiring & Onboarding Plan

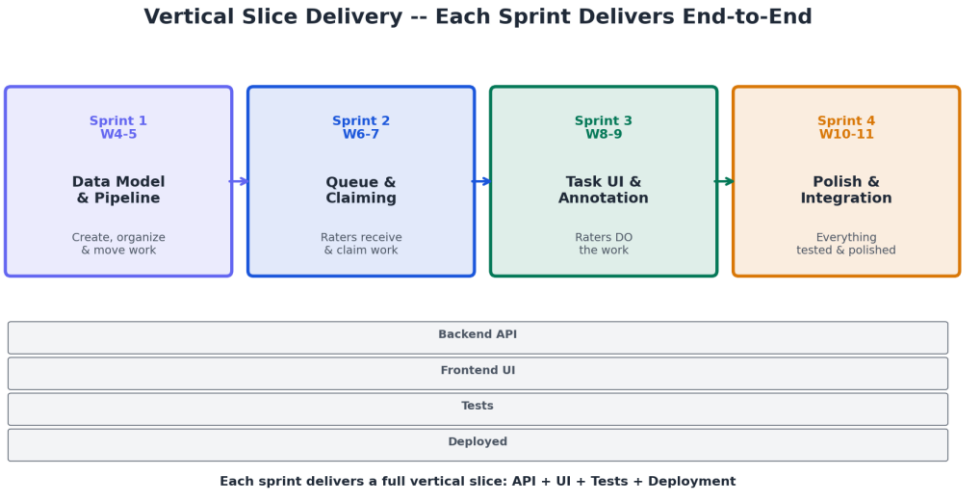
Week	Who Joins	What They Do Immediately
Week 1	Tech Lead + Infra Engineer (2 people)	Infra sets up AWS; Tech Lead does architecture + DB design
Week 2	+ Backend Eng + Frontend Eng (4 people)	Backend starts APIs on Day 1; Frontend starts component library
Week 3	Full team operational	Everyone is productive and shipping code

4. Master Timeline & Phases



Delivery Rhythm

- **Every 2 weeks:** Sprint demo to client (live, in staging)
- **Every week:** Friday status email (3 bullets + risks)
- **Every sprint:** Deployable increment -- never more than 2 weeks without a working feature
- **9 total demos** across Phases 1-2. Client sees progress 9 times before final delivery.



5. Phase 0 -- Discovery & Setup (3 Weeks)

Duration: 3 weeks (Weeks 1-3) | Team: 2 people (Week 1) -> 4 people (Weeks 2-3)

Goal: When Phase 1 starts on Week 4, every engineer opens their laptop and starts building features -- zero ambiguity about what to build, how to build it, or where to deploy it.

Week 1: Architecture & Infrastructure

- Client kickoff: access AWS, GitHub, Okta sandbox, Datadog; align on PRD questions
- Architecture decisions: monorepo structure, DB boundaries (PG vs Mongo), API conventions
- Database schema design: PostgreSQL ERD + MongoDB document schemas
- Provision AWS dev environment: VPC, RDS, DocumentDB, S3, ECR, ALB
- Create GitHub monorepo (Turborepo), GitHub Actions CI pipeline
- Docker Compose for local development

Week 2: Scaffolding & Design Direction

- Full team onboarding (4 people), codebase walkthrough, local env setup
- Implement PostgreSQL schema via Prisma + seed script
- Implement MongoDB schemas via Mongoose
- Build shared UI component library (shadcn/ui, 12+ base components)
- Build Ops Portal + Rater Portal shells (navigation, layout, placeholder pages)
- Implement auth module: JWT, bcrypt, login/logout/me, RBAC middleware
- Set up Playwright E2E framework + Datadog agent

Week 3: Integration & Sign-Off

- End-to-end login flow: login page -> API -> JWT -> portal -> role-based routing
- RBAC enforcement: rater cannot access Ops URLs (server-side)
- Deploy all apps to staging, S3 signed URL service working
- Sprint 1-4 backlog creation with acceptance criteria
- Client demo: staging login + portals + architecture walkthrough

What the Client Sees at End of Phase 0

A polished login page. The Ops Portal with a sidebar (Dashboard, Projects, Batches, Tasks, Users) -- pages are placeholder but navigation and design language are real. A separate Rater Portal. RBAC works: raters get 403 on Ops URLs. In GitHub: a clean monorepo with green CI and automated deploys. It looks like a real product. It just does not do anything yet. But the skeleton is rock-solid.

6. Phase 1 -- MVP Foundational Platform (8 Weeks)

Duration: 8 weeks (W4-W11) | 4 two-week sprints | Team: 4 people

Goal: A production-ready internal annotation platform that replaces manual task distribution.

Sprint 1 (Weeks 4-5): Data Model & Pipeline Engine

Theme: "Work can be created, organized, and moved through a pipeline"

- Pipeline engine: configurable stages in MongoDB, state machine for transitions
- Response schema system: JSON Schema storage, ajv validation, per-level schemas
- Full CRUD APIs: Workspace, Project, Batch, Task, Annotation
- Ops Portal: project pages, pipeline config UI, batch & task views
- S3 file upload integration with presigned URLs

Sprint 2 (Weeks 6-7): Queue, Assignment & Claiming

Theme: "Raters receive work and can claim it without conflicts"

- Claim locking engine: SELECT FOR UPDATE SKIP LOCKED -- zero double claims
- FIFO queue engine with Redis sorted set cache for 100K tasks/day
- Assignment system: manual and automatic, batch-level
- Rater Portal: task queue, "Start Next Task" button, claim flow with heartbeat
- Media renderers: image zoom/pan, audio waveform, video player, HTML/Markdown
- Concurrency load test: 50 raters claiming simultaneously -- zero collisions

Sprint 3 (Weeks 8-9): Task UI & Annotation Flow

Theme: "Raters open tasks, see rich media, fill out annotation forms, and submit"

HIGHEST RISK SPRINT: Client confirmed Task UI requires conditional logic, nested schemas, and repeater sections. Tech Lead pair-programs with Frontend Engineer during this sprint.

- Recursive schema-driven form renderer: conditional fields, nested objects, array repeaters
- Annotation submission pipeline: validate, store in MongoDB, update PG, advance pipeline
- Reviewer flow: see all prior annotations + reviewer-only fields
- Full rater flow: claim -> media + dynamic form -> submit -> auto-load next

Sprint 4 (Weeks 10-11): Polish, User Management & Phase 1 Acceptance

Theme: "Everything works end-to-end, is tested, documented, and ready for sign-off"

- User management: create, edit, list, deactivate, roles, bulk CSV import
- Basic export API (JSON/CSV), activity logging to Datadog
- Full UI polish: error states, loading skeletons, toast notifications, responsive design
- Full regression E2E suite (40+ tests), load test (100 concurrent raters)
- OpenAPI/Swagger docs, architecture diagrams, developer setup guide

Phase 1 – PRD P0 Acceptance Deliverables

- Data Model: Workspace -> Project -> Batch -> Task -> Annotation hierarchy, full CRUD APIs
- Pipeline Engine: Configurable multi-stage pipeline, status transitions, Ops controls
- Queue System: FIFO ordering, deterministic claim locking, assignment, heartbeat + timeout
- Task UI: Media rendering (text/image/audio/video/HTML), schema-driven response forms with conditional logic + repeaters
- Rater Portal: Task queue, claim, annotate, submit, auto-next
- Ops Portal: Project setup, pipeline config, batch creation, task upload, assignments, filtering
- Auth: JWT login, RBAC enforcement (server-side), internal vs external separation
- Storage: S3 structured storage with signed URLs for upload and retrieval
- Tests: 40+ E2E tests, API tests, concurrency load tests -- all green in CI
- Documentation: OpenAPI/Swagger API docs, architecture summary, setup guide

All Phase 1 deliverables correspond to PRD P0 requirements including CRUD APIs, pipeline engine, queueing, deterministic claim locking, Task UI, Ops Portal, Rater Portal, and baseline RBAC enforcement

7. Phase 2 -- Operational Expansion (10 Weeks)

Duration: 10 weeks (W12-W21) | 5 two-week sprints | Team: 4 people

Goal: Enterprise-grade operations -- 10+ concurrent projects, quality controls, Okta SSO, customer portal.

Sprint 5 (W12-13): Consensus & Benchmarks

- Consensus engine: N annotations per task, same task served to N raters, aggregation view
- Benchmark system: golden tasks, gate mode (blocks until X%), interleaved injection
- Rater levels and mass assignment by level
- Reviewer consensus view: side-by-side display of all N annotations

Sprint 6 (W14-15): Bulk Upload, Export & Ops Controls

- Streaming bulk upload: NDJSON/CSV, real-time progress, error reporting
- Batch export: CSV/JSON + media URLs + ZIP packaging
- Reset/archive/ignore controls (single + bulk) with full audit trail
- Load test: 10K task upload < 2 min, 5K export < 60s

Sprint 7 (W16-17): Linters, Time Tracking & Tags

- Pluggable linter framework: blocking vs warning mode, 3 built-in linters
- Time tracking: active time per task, idle detection, Ops queryable
- Tag management: bulk tags on tasks + raters, tag-based filtering

Sprint 8 (W18-19): Customer Portal, Okta SSO & Audit

- Okta SSO: OAuth2 PKCE, JIT provisioning, session management
- Customer Portal: org dashboard, seat management, API key, annotation volume
- Full audit logging: searchable, filterable, 15-day retention with cleanup

Sprint 9 (W20-21): GUI UI Builder, Impersonation & Final Polish

- GUI UI Builder: visual schema editor with live preview, versioned
- Admin impersonation: time-bounded, logged, with clear UI indicator
- Full regression suite (80+ E2E tests), load test (200 concurrent raters)
- Production deployment: all services, monitoring, alerting

Phase 2 – PRD P1 Acceptance Deliverables

- Consensus workflows (N annotations per task, side-by-side reviewer view)
- Benchmark system (golden tasks, gate + interleaved modes)
- Bulk upload (streaming NDJSON/CSV, progress, error reports) + batch export (CSV/JSON/ZIP)
- Linter framework (pluggable, blocking/warning, 3 built-in) + time tracking
- Tag management, rater levels, project-level assignment
- Okta SSO (OAuth2 PKCE) replacing JWT auth
- Customer Portal (subscription, seats, API key, usage metrics)
- Full audit logging (searchable, 15-day retention)
- GUI UI Schema Builder (visual editor, live preview)
- Admin impersonation (time-bounded, audited)
- Production environment deployed, monitored, and alerted
- 80+ E2E tests, 200-user load tests, security checklist complete

8. Phase 3 -- Optional Enhancements

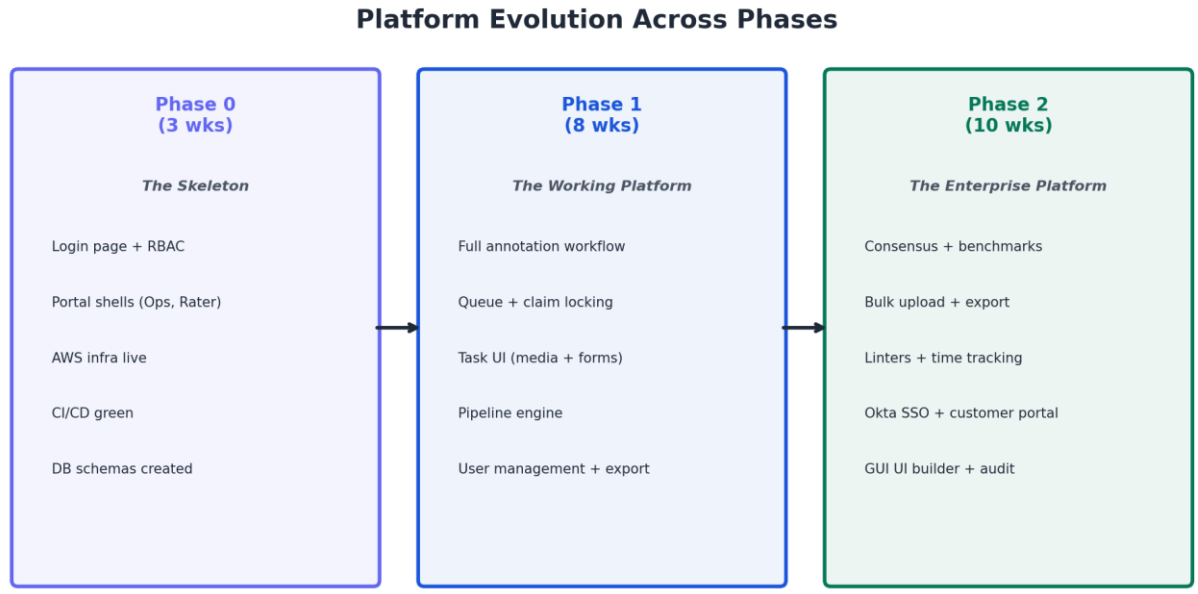
Duration: TBD (SOW-dependent) | Team: 4 people (or scale to 5-6)

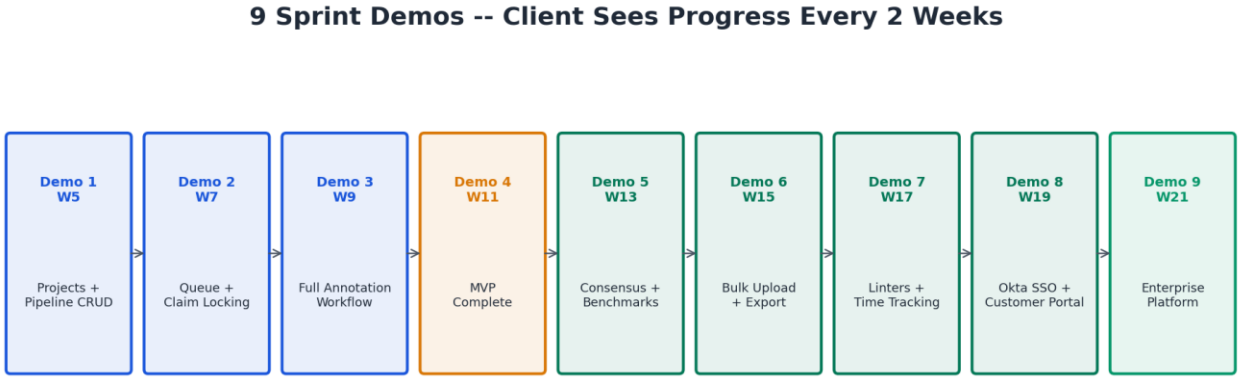
Feature	Estimated Effort	PRD Refs
Drafts / autosave	2 weeks	UI-06
Advanced assignment (random/LIFO/weighted)	1.5 weeks	QUEUE-05
Pre/post-processing pipelines (Lambda)	3 weeks	PROC-01
Prelabels/predictions in UI	2 weeks	PROC-02
Custom grader logic (secure sandbox)	3 weeks	MISC-05
Ops dashboard (status counts + drill-down)	2 weeks	OPS-05
Advanced task filtering	1.5 weeks	OPS-07
Notification center	1.5 weeks	MISC-04
Customer upload/download UI	2 weeks	CUST-03
Customer analytics dashboard	2 weeks	CUST-02

Total if all selected: ~22 weeks. We recommend selecting 8-10 weeks of features per SOW.

9. Platform Evolution Vision

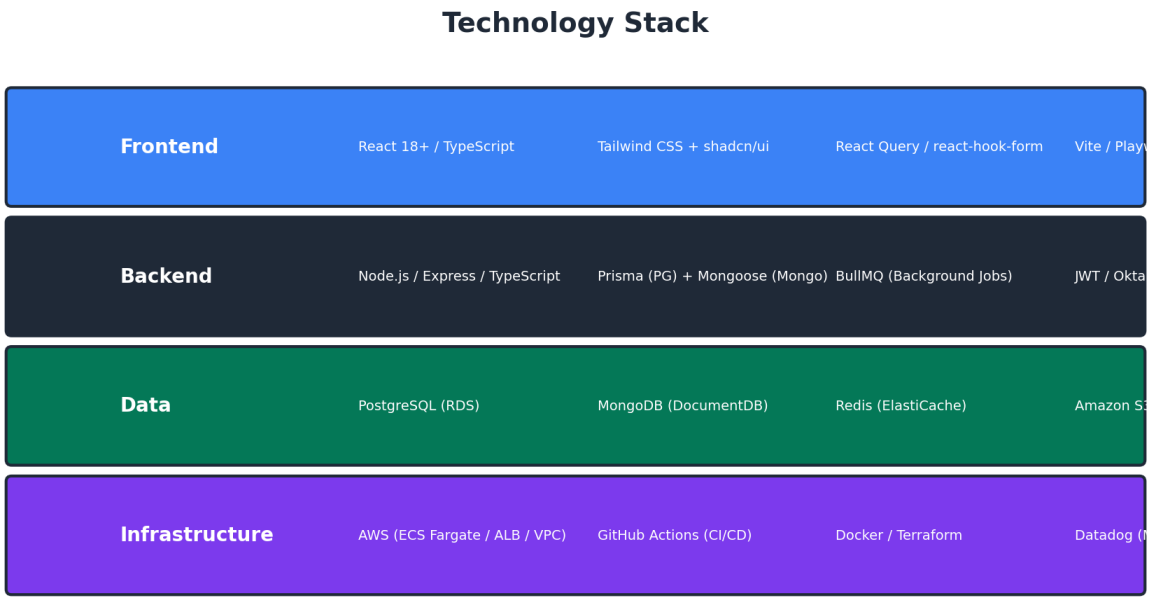
The platform grows incrementally across phases. Here is what the client will see at each milestone:





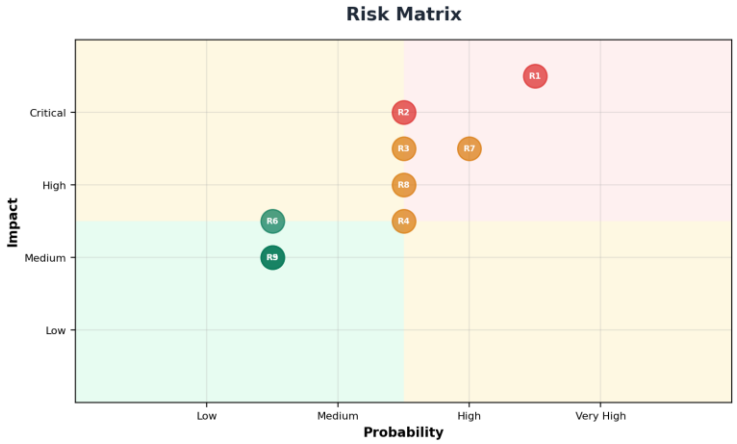
Live demos in staging every 2 weeks. If we fall behind, it is visible at the first demo -- not at the end.

10. Technology Stack



Layer	Choice	Why
Monorepo	Turborepo	Fast incremental builds, shared types, single CI pipeline
Backend	Node.js + Express + TypeScript	Client-mandated
Frontend	React + TypeScript + Tailwind CSS	Client-mandated
UI Components	shadcn/ui (Radix + Tailwind)	Professional design system without a designer
PostgreSQL ORM	Prisma	Type-safe queries, auto-migrations, great DX
MongoDB ODM	Mongoose	Schema validation, middleware hooks, mature
Cache Layer	Redis (ElastiCache)	Queue caching, sessions, task counts at 100K/day
Auth (Phase 1)	JWT (bcrypt + jsonwebtoken)	Simple, swappable via adapter pattern
Auth (Phase 2)	Okta OAuth2 PKCE	Client-mandated SSO
Object Storage	AWS S3 + presigned URLs	Client-mandated
CI/CD	GitHub Actions	Client-mandated
Containers	Docker + ECS Fargate	Serverless containers, no cluster management
Monitoring	Datadog	Client-mandated
Testing	Playwright + Jest + k6	E2E, unit/API, load testing

11. Risk Management & Mitigation



SN	Risk	Probability	Impact	Mitigation
R1	Bus factor = 1 per domain	High	Critical	Tech Lead is capable backup for both. All code PR-reviewed. Critical modules pair-programmed.
R2	Someone leaves or gets sick	Medium	Critical	All code documented + PR history. Tech Lead covers temporarily. Replace within 1-2 weeks.
R3	Phase 1 scope tight for 8 weeks	Medium	High	Sprint 4 is buffer sprint. Scope frozen at Phase 0 sign-off.
R4	No dedicated QA -- bugs slip	Medium	Medium	Infra Eng owns automation (E2E + load). All devs write unit tests. PRs blocked without tests.
R5	No designer -- UI quality risk	Low	Medium	shadcn/ui provides professional design out of the box. Frontend Eng has UI/UX sense.
R6	No PM -- communication gaps	Low	Medium	Tech Lead handles client comms directly. 2x/week sync. Friday status emails.
R7	Claim locking concurrency bugs	Medium	High	PostgreSQL SKIP LOCKED is proven. 1000-run concurrency test in Sprint 2.
R8	Client feedback delays	Medium	High	Sprint demos every 2 weeks. Async Slack with 4-hour SLA.
R9	Okta integration complications	Low	Medium	Auth adapter pattern isolates Okta from Phase 1 code. Dedicated sprint for auth swap.
R10	Client prerequisites delayed (AWS, Okta, Datadog access)	Medium	High	Dependencies clearly listed in Section 14. Delays shift timeline accountability to client side.

Bus Factor Mitigation

Domain	Primary	Backup	Emergency
Backend	Backend Eng	Tech Lead	Hire in 1-2 weeks
Frontend	Frontend Eng	Tech Lead	Hire in 1-2 weeks
Infra	Infra Eng	Backend Eng (CI/CD basics)	Tech Lead (AWS basics)

12. Communication & Reporting

Meeting	Frequency	Duration	Attendees	Purpose
Daily Standup	Daily	10 min	All 4 engineers	Blockers, priorities
Client Sync	2x/week (Tue + Thu)	30 min	Tech Lead + Client PO	Progress, decisions
Sprint Demo	Every 2 weeks	45 min	All 4 + Client stakeholders	Live demo in staging
Knowledge Share	Weekly (Fri)	30 min	All 4 engineers	Each person demos work
Sprint Planning	Every 2 weeks (Mon)	60 min	All 4 engineers	Next sprint backlog

With 4 people, total meeting time is ~3.5 hours/week. An 11-person team would spend 8-10 hours/week in meetings.

Artifact	Frequency	Content
Sprint Demo	Every 2 weeks	Live walkthrough of new features in staging
Friday Email	Weekly	3-bullet summary, risks, next week plan
Phase Report	Per phase	Deliverable checklist, test results, performance metrics

Purpose	Tool
Project tracking	Linear (lightweight, fast for small teams)
Communication	Slack: #annotation-platform (client) + #annotation-team (internal)
Code	GitHub (client org)
CI/CD	GitHub Actions
Monitoring	Datadog

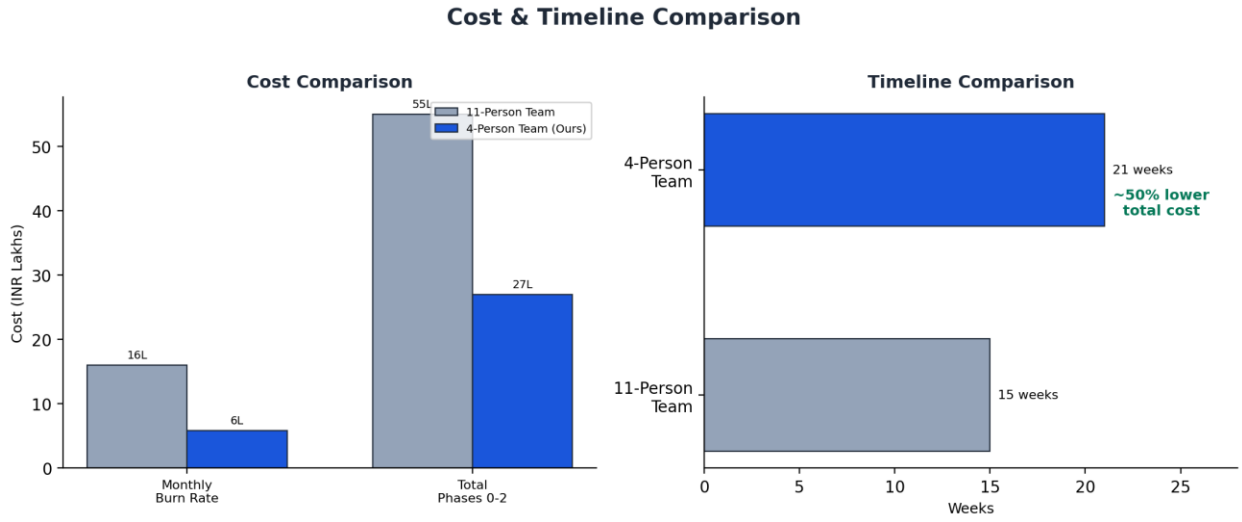
13. Cost & Resource Summary

Team Cost

Role	Count	Monthly Cost (INR)
Tech Lead / Architect	1	INR 1,40,000 - 1,50,000
Senior Backend Engineer	1	INR 1,40,000 - 1,50,000
Senior Frontend Engineer	1	INR 1,40,000 - 1,50,000
Infra + QA Engineer	1	INR 1,40,000 - 1,50,000
Total Team (4 people)	4	INR 5.6L - 6.0L /month

Phase Cost Projection

Phase	Duration	Estimated Cost (INR)
Phase 0 (ramp: avg 3 people)	3 weeks	INR 3.86L – 4.13L
Phase 1 (full team)	8 weeks	INR 10.3L - 11.1L
Phase 2 (full team)	10 weeks	INR 12.9L - 13.9L
Total Phases 0-2	21 weeks	INR 26.1L - 28.1L
Phase 3 (scope-dependent)	TBD	Rate x duration



The 4-person team costs ~50% less in total while taking 40% more calendar time. The tradeoff is time for money -- the client gets the same product at significantly lower total cost.

Our lean team reduces cost by ~INR 27L while still meeting PRD acceptance criteria. The tradeoff is calendar time, but risk-adjusted delivery ensures compliance with all P0/P1 requirements.

AWS Infrastructure (100K Tasks/Day Scale)

Note: AWS infrastructure is billed in USD globally. Costs shown in USD and INR equivalent (at ~INR 84/USD).

Service	Sizing	Monthly Cost (USD)	Monthly Cost (INR)
RDS PostgreSQL	db.r6g.xlarge + read replica	$800 - $1,200	INR 67K - 1.0L
DocumentDB (MongoDB)	db.r6g.xlarge	$500 - $750	INR 42K - 63K
ElastiCache Redis	cache.r6g.large	$200 - $350	INR 17K - 29K
ECS Fargate (4 services)	Auto-scaling API	$400 - $700	INR 34K - 59K
S3 (10 TB)	500MB files x high volume	$250 - $500	INR 21K - 42K
ALB + Route53		$50 - $100	INR 4K - 8K
ECR		$10 - $20	INR 1K - 2K
Total AWS (Staging + Prod)		$2,210 - $3,620/mo	INR 1.86L - 3.04L/mo

14. Prerequisites from Client

Everything below must be completed or confirmed BEFORE Week 1, Day 1. We recommend starting 2-3 weeks before the planned kickoff date.

Contractual & Legal

- SOW / Contract signed
- Mutual NDA for all 4 team members
- IP assignment agreement (work-for-hire confirmed)
- Change control process

AWS & Infrastructure

- AWS account access (Tech Lead + Infra Eng)
- AWS region confirmed
- AWS budget approved ($2K-$4K/month)
- Domain names decided

GitHub

- GitHub org access for all 4 members
- Repo creation permission
- GitHub Actions minutes confirmed
- PR review process agreed

Okta (for Phase 2)

- Okta sandbox access requested (can arrive by Week 2)

Monitoring

- Datadog org access
- API key + Application key provided

Client Stakeholders

- Product Owner named and available
- Technical Point of Contact
- Executive Sponsor
- Communication channel created
- Recurring meeting invites sent

Technical Decisions

- Tech stack confirmed
- Container orchestration (ECS vs EKS)
- IaC tool (Terraform vs CDK)
- Environments confirmed (dev/staging/prod)

Scale & Performance

- All scale assumptions reviewed and confirmed (100K tasks/day, 800 peak users)

Domain Knowledge

- Current workflow walkthrough scheduled
- Sample task data delivery by Week 2

15. Commitments & Guarantees

Senior-only team: Every person has 5+ years of experience. No juniors learning on your project. Four people who each operate independently at a high level.

Working software every 2 weeks: 9 sprint demos across Phases 1-2. The client sees real features running in staging every 14 days. If we fall behind, it is visible at the first demo -- not at the end.

Full transparency: Code in the client's GitHub. Staging accessible 24/7. Datadog dashboards shared. Linear board visible. No black boxes.

Deep ownership: Each engineer owns their domain end-to-end. The person who builds the API also writes the tests and monitors the logs. Fewer bugs, faster debugging, stronger accountability.

Scope discipline: Phase 1 P0 requirements frozen at Phase 0 sign-off. Sprint 4 is our built-in buffer. We plan conservatively and execute precisely.

Knowledge transfer: All architecture decisions in ADRs. APIs documented in OpenAPI/Swagger. Components with usage examples. Runbooks for operations. The codebase stands on its own.

Post-delivery warranty: 4-week warranty after each phase. P0 bugs (system down): 24hr response, 48hr fix. P1 bugs (feature broken): 48hr response, 1-week fix. P2 bugs (cosmetic): logged, no SLA.

Success Metrics

Phase	Success =
Phase 0	Staging live. Architecture approved. Backlog estimated. Team shipping code.
Phase 1	End-to-end annotation workflow in staging. All P0 requirements pass acceptance. 100 concurrent raters with zero double-claims.
Phase 2	10+ concurrent projects with isolation. Okta SSO working. Bulk upload/export at scale. Customer portal live. Full audit trail. Production deployed.

16. Key Milestone Calendar

Week	Milestone	What Client Sees
W1	Kickoff + infra provisioning	Architecture doc + AWS environments
W2	Full team onboard	Working local dev stack
W3	Phase 0 Complete	Login + portal shells in staging
W5	Sprint 1 Demo	Projects + pipelines + task CRUD working
W7	Sprint 2 Demo	Queue + claiming + assignment working
W9	Sprint 3 Demo	Full annotation flow (claim -> annotate -> review -> done)
W11	Phase 1 MVP Accepted	End-to-end platform in staging
W13	Sprint 5 Demo	Consensus + benchmarks + rater levels
W15	Sprint 6 Demo	Bulk upload + export + reset/archive
W17	Sprint 7 Demo	Linters + time tracking + tags
W19	Sprint 8 Demo	Customer portal + Okta SSO + audit logs
W21	Phase 2 Accepted	Enterprise platform in production
W22+	Phase 3 Kickoff	Per SOW

Why Choose This Approach

	Large Team (11 people)	Our Lean Team (4 people)
Decision speed	Committee discussions, review boards	Tech Lead decides in real-time, team executes
Communication	11 people x n channels = noise	4 people in one room = clarity
Code consistency	Multiple authors, different styles	Tech Lead reviews every PR, consistent patterns
Accountability	"That's the other team's module"	"I built it, I own it, I'll fix it"
Client relationship	PM relays messages through layers	Tech Lead talks to client directly
Total Cost	INR 53L - 57L	INR 26L - 28L

We have chosen the hardest path -- a small team with no room for passengers. Every person must perform. But the result is a tighter codebase, clearer ownership, and a lower bill.

This proposal is the technical foundation for the Statement of Work. All timelines, costs, and commitments are subject to final SOW agreement.