CPM‑AI Development & Technical Implementation White Paper

Version 2.2 – April 2025

---

0. Executive Summary

Construction Project Management AI (CPM‑AI) is a cloud‑native, micro‑services SaaS designed to automate and streamline bid evaluation, scope extraction, risk scoring, and reporting for construction projects. It ingests raw PDFs and drawings, applies state‑of‑the‑art AI (OCR, LayoutLM, ViT, RAG over LLMs, ensemble risk models), and delivers structured data, interactive PM assistance, and polished PDF/DOCX reports with full audit trails.

Key Benefits for Investors & Users

- **10× faster** bid takeoff and scope validation
- **Error reduction** via automated parsing and double‑model verification
- **Auditability** with source‑linked proofs for every AI decision
- **Scalable** architecture on AWS EKS, IaC via Terraform, observability with Prometheus/Grafana
- **Continuous learning** loop: user feedback loops into nightly fine‑tuning

---

1. Workflow Understanding

Construction projects require timely, accurate budgets built from disparate vendor quotes, drawings, and specifications. CPM‑AI's workflow aligns with this lifecycle:

1. **Initiate Project**: PM defines a new project (ID issued) and shares preliminary scope guidelines.

2. **Collect Documents**: Subcontractors upload quotes (PDFs), spec sheets, and permit drawings via the web UI or API.

3. **Ingestion & De-duplication**: Each file is hashed, de-duplicated, and persisted in S3 (`cpm-raw-docs/<org>/<project>`). Duplicates generate alerts to PM.

4. **Document Routing**: SQS events trigger the Pre-Processor:

    - **Quotes** → Textract/Tesseract → Quote Parser (spaCy) → structured rows in Postgres.

    - **Drawings** → Vision Sheet Classifier → CSI division labels + embeddings → pgvector.

    - **Specs** → RAG Scope Extractor → JSON scopes saved to `trade_scopes`.

5. **Scope Synthesis**: Once quotes and scopes are in place, the system assembles a per-trade scope matrix, merging pricing and scope items.

6. **Risk Scoring**: The Risk Service computes missing-scope probabilities and flags high-risk trades for PM review.

7. **Interactive Queries**: PM or team members ask natural-language questions (`/query`), e.g. “What’s missing?”, “Who’s best for HVAC?”. AI agent retrieves context, generates answers with `## Proof`, and auto-queues RFIs if confidence is low.

8. **Budget Optimization**: The MILP-based `optimal_budget` solver selects the lowest-cost, full-scope vendor combination. Results are stored and available for queries.

9. **Report Generation**: PM triggers the Decision Report Generator (PDF/DOCX) in either Executive or Instructional tone. A chained LLM pipeline drafts, critiques, polishes, then compiles to PDF with audit annotations.

10. **Delivery & Feedback**: Final report URL delivered via WebSocket/email. User accepts or rejects AI suggestions; feedback feeds nightly retraining of LoRA adapters.

This end-to-end flow ensures no manual Excel juggling, no orphaned scope gaps, and a fully auditable trail from raw documents to executive summary.

---

2. SaaS Capability Overview

1. **Ingestion & Pre-Processing**
   - **Front-End** (React + Tailwind): file upload ZIP or individual PDFs, GraphQL mutations via tRPC.
   - **Ingestion Service** (FastAPI + Celery): computes SHA-256, de-duplicates via Redis, stores raw files on S3 (`cpm-raw-docs/`), emits SQS events.
   - **Pre-Processor**: triggered by SQS, routes by doc type:
     1. **Quotes** → OCR & NLP path
     2. **Drawings** → Vision classification path
     3. **Specs** → RAG only
2. **AI Parsing & Classification**
   - **OCR/Text Extraction**: AWS Textract primary, Tesseract fallback; LayoutLMv3 fine-tuned for table/form structure.
   - **Quote Parser**: spaCy NER extracts vendor, trade, price, inclusions/exclusions; RapidFuzz text normalization; pydantic models write to PostgreSQL JSONB.
   - **Sheet Classifier**: ViT-Large/16 + LoRA predicts CSI division labels (F1≈0.98), produces 1536-dim embeddings stored in pgvector.
3. **RAG-Driven Scope Extraction**
   - **Retriever**: Batch-embedding (Ada-002) of question + documents, cosine similarity selects top-K.
   - **Generator**: Routed LLMs (GPT-4o-128k, Claude 3 Opus, Gemini 1.5 Pro, on-prem Llama-3) based on token count & sensitivity.
   - **Output**: JSON schema `{scope_items, materials, flags}` with `sources:[sheet_id,…]` and post-validation `risk_score`; saved in `trade_scopes`.
4. **Risk Scoring Micro-Service**
   - **Features**: scope coverage %, vendor history, sheet count, cost variance std-dev.

    - **Model**: XGBoost primary (AUC 0.83) + LightGBM fallback; served via FastAPI `/score`.

5. **Interactive Project Assistant**
    - **Service**: FastAPI `/query` with SSE and JWT auth via JWKS.
    - **Context**: loads quotes, scopes, and `optimal_budget` from Aurora Postgres.
    - **RAG context + system prompt** feed into streaming LLM answer, includes inline `## Proof` blocks linking plan/spec snippets, parsed JSON, and match logic.
    - **Confidence Loop**: LLM self-evaluation; if confidence < 0.7, auto-draft RFI to SQS for human review.
6. **Decision Report Generator**
    - **Lambda / FastAPI** (`decision_report_generator` v2.1)
    - **Writer's Mode** toggle (executive vs instructional tone) persisted in user profile
    - **Model Chain**:
        1. **GPT-4o** – initial Markdown draft
        2. **Claude 3 Opus** – JSON-Patch critique & corrections
        3. **Gemini 1.5 Pro** – concise, polished tone
        4. **(Optional)** Llama-3-70B on Bedrock for PII data
    - **Hallucination Detector**: mDeBERTa v3 mini filters risky sections (queued for manual review)
    - **Rendering**: Jinja2 → LaTeX template → Tectonic → PDF; optional Pandoc conversion to DOCX
7. **Continuous Learning & Feedback**
    - **User Actions**: Accept/Reject suggestions stored in `feedback_events`

   - **Airflow** nightly jobs sample feedback, retrain LoRA adapters (Sunday 03:00 UTC)

8. **Monitoring & Operations**
   - **K8s on EKS**: ArgoCD for GitOps, KEDA autoscale on SQS queue length
   - **Infra as Code**: Terraform modules + Helm charts
   - **Observability**: Prometheus metrics, Grafana dashboards, Loki logs, Sentry traces
   - **Notifications**: WebSocket & email signed URLs for report delivery
9. **Security & Compliance**
   - **Network**: VPC with private DB subnets; public ALB for gateways behind WAF
   - **Encryption**: AES-256 at rest (S3, RDS, pgvector), TLS 1.3 in transit
   - **Secrets**: AWS Secrets Manager for all API keys & DB creds
   - **AuthZ**: JWT scopes enforced in service middleware
   - **Auditing**: ELK-stack logs of prompts, contexts, proofs, decisions for forensics
   - **Compliance**: SOC 2 Type II readiness, change management, vulnerability scans
10. **Investor-Ready Roadmap & Extensions**

- **Neo4j Knowledge Graph**: unify cost, specs, vendor, schedule relationships
- **Streaming OCR**: live mobile capture → on-site scope deviation alerts
- **GNN Risk Models**: propagate risk across subcontractor networks
- **Voice Interface**: Alexa/Google Home integration for hands-free queries

---

3. Repository Structure & File Responsibilities

monorepo/

```
├─ services/
│  ├─ ingestion/        # FastAPI + Celery: file hash → S3 → SQS
│  ├─ ocr_pipeline/     # FastAPI: Textract/Tesseract → JSON → S3 interim
│  ├─ nlp_quote_parser/  # FastAPI: spaCy NER → Postgres
│  ├─ vision_sheet_cls/  # FastAPI: ViT classifier + embedder → pgvector
│  ├─ rag_scope_extractor/# FastAPI: RAG → scope JSON
│  ├─ risk_scoring/     # FastAPI: XGBoost predict_proba → JSON
│  └─ reports/          # Lambda/FastAPI decision_report_generator v2.1
├─ api_gateway/         # GraphQL-Yoga: `bids` query
├─ web/                 # React 18 + Tailwind + tRPC
├─ infra/               # Terraform modules & Helm charts
└─ tests/               # pytest + Playwright E2E
```

Data stores & messaging:

- **S3**: `cpm-raw-docs`, `cpm-interim-json`, `reports`
- **Postgres + pgvector**: metadata, quote/scope JSON, embeddings
- **Redis**: idempotency & rate limiting
- **SQS**: ingestion, OCR, parse, classification, extraction, RFIs
- **SNS / WebSocket**: front-end notifications

---

4. CI/CD & Deployment

- **CI**: GitHub Actions runs `bazel test //tests/...`, `bazel build //services/...`.
- **Docker**: Hermetic Bazel Docker targets push images to ECR.
- **Infra**: Terraform provisions EKS, RDS, S3, SQS, Redis, IAM, VPC.
- **GitOps**: ArgoCD monitors Helm charts, auto-deploys to dev/staging/prod.
- **Blue/Green**: ALB routing + automated rollback on failure.

---

5. Testing & Quality

- **Unit & Integration**: pytest, pytest-cov; Playwright for UI flows.
- **Data QA**: Great Expectations for JSON schema & field validation.
- **Load**: k6 stress tests on `/query` and `/generate-report`.
- **Security Scans**: Snyk dependency scans, AWS Inspector on ECR images.

---

6. Conclusion & Contact

CPM-AI v2.2 provides a robust, end-to-end platform that automates the most time-consuming construction takeoff tasks, embeds continuous AI learning, and ensures enterprise-grade security and compliance. Its modular design and micro-service architecture allow rapid feature extension—poised to become the industry standard cognitive engine for construction project management.

Contact: engineering@cpm-ai.com | www.cpm-ai.com