Project Blue Whale

Month 1 Implementation Plan

PE Tech Due Diligence SLM - MVP

Timeline	4 weeks
Budget	<$500 USD
Platform	GCP Vertex AI
Model	Gemma 2 9B
Focus	Use Cases A & B: Sourcing + Tech DD Evaluation
Date	November 17, 2025

Table of Contents

1. Executive Summary

Project Blue Whale Month 1 focuses on building the first PE-focused Small Language Model (SLM) for technology due diligence. This MVP will deliver automated deal sourcing evaluation and comprehensive technical assessment capabilities, enabling PE firms to evaluate AI and Web3 investment opportunities with unprecedented speed and depth.

Core Deliverables

- API-accessible SLM performing automated deal sourcing and tech due diligence
- Multi-modal capability for analyzing architecture diagrams and technical documentation
- Dual output format: Structured JSON + narrative Markdown reports
- Chain-of-thought reasoning for explainable assessments
- Training dataset of 800-1,200 examples from SEC filings, VC reports, and synthetic data
- Complete documentation and code repository for future expansion

Use Case Focus

Use Case A - Deal Sourcing: Interactive qualification system that asks strategic questions to evaluate technical feasibility and team capability before proceeding to full due diligence.

Use Case B - Tech Due Diligence Evaluation: Comprehensive assessment covering architecture quality, engineering team, technical debt, security posture, scalability, development velocity, technical moat, and infrastructure maturity.

2. Technology Stack Decision Matrix

Core Model: Gemma 2 9B

Why Gemma over Llama:
• Native GCP integration via Vertex AI Model Garden
• Lower inference costs on Vertex AI infrastructure
• Superior instruction-following for structured outputs
• Commercially permissive license with no usage restrictions
• Efficient parameter count balancing capability and cost

Multi-Modal Strategy

Primary Model: Gemma 2 9B for text analysis and reasoning
Visual Analysis: Gemini 1.5 Flash for architecture diagrams ($0.35/1M tokens)
Integration: Two-stage pipeline where Gemini extracts visual information, then Gemma performs holistic analysis combining text and visual insights

GCP Services Stack

Service	Purpose	Estimated Cost
Vertex AI (Gemma fine-tuning)	Model training	$80-150
Vertex AI Prediction (serverless)	Inference endpoint	$5-10
Cloud Storage	Training data & artifacts	$1-2
Gemini 1.5 Flash API	Visual analysis	$10-20
Cloud Functions	API wrapper	Free tier
Secret Manager	API keys	Free tier
TOTAL		**~$100-185**

Budget Buffer: $315-400 remaining for iterations, additional training runs, or data acquisition

3. Technical Assessment Framework Recommendation

Recommended Framework: Modified Deloitte Tech DD + YC Technical Diligence

This combination provides the optimal balance of enterprise rigor and startup-specific insights for PE technology investments.

Why This Combination:

- **Deloitte Tech DD Framework:** Industry standard covering architecture assessment, engineering talent evaluation, technical debt quantification, and scalability analysis
- **YC Technical Diligence:** Startup-specific additions including product-market fit technical indicators, founder technical capability, speed of iteration metrics, and technical moat assessment

Framework Implementation

We will encode this as a **structured prompt template** with scoring rubrics across 8 dimensions:

Dimension	Score Range	Assessment Focus
Architecture Quality	0-10	Modern stack, scalability, maintainability
Engineering Team	0-10	Talent quality, leadership, retention
Technical Debt	0-10	Code quality, test coverage, documentation
Security Posture	0-10	Vulnerabilities, compliance, practices
Scalability	0-10	Horizontal/vertical scaling, bottlenecks
Development Velocity	0-10	Release frequency, iteration speed
Technical Moat	0-10	Proprietary tech, patents, differentiation
Infrastructure Maturity	0-10	DevOps, monitoring, reliability

Output: Overall Tech DD Score (0-100) + narrative explanation + risk flags + caveats

4. Data Sourcing Strategy

Target: 800-1,200 training examples for MVP, balanced across three quality tiers to ensure model learns from both authoritative sources and diverse scenarios.

Tier 1: High-Quality Structured Data (200-300 examples)

- **Source 1: SEC Filings - Technology Risk Factors**
  Extract 'Risk Factors' and 'Business' sections from tech company S-1s and 10-Ks. Target recent IPOs (2020-2024) like Snowflake, Databricks, Unity, Palantir. Free via sec-edgar-downloader library. *Volume: ~150 examples*

- **Source 2: Public PE/VC Tech DD Reports**
  White papers and case studies from Bessemer (Cloud 100 reports), a16z (Technical Founder assessments), Sequoia (Arc evaluations), FirstMark Capital tech DD blog posts. Free via web scraping. *Volume: 40-60 reports*

- **Source 3: Acquisition Post-Mortems**
  Public M&A; integration reports from Forrester, Gartner, and public 'lessons learned' from failed acquisitions. Cost: $0-50 using free trials. *Volume: 30-50 examples*

Tier 2: Synthetic + Augmented Data (300-500 examples)

- **Source 4: Synthetic DD Reports**
  Generate realistic tech DD reports using GPT-4 based on real company profiles from StackShare, Crunchbase, AngelList. Controlled format ensures consistency. Cost: ~$20-30 in API calls. *Volume: 300 examples*

- **Source 5: GitHub Repository Analysis**
  Tech stack extraction and code quality metrics from public repos of acquired companies (2020-2024). Run automated static analysis (SonarQube, CodeClimate). Free and objective. *Volume: 100 repos*

Tier 3: Community Knowledge (200-400 examples)

- **Source 6: Reddit/HackerNews DD Discussions**
  Technical discussions from r/venturecapital, r/startups, r/entrepreneur and HN about acquisitions, tech choices, architecture decisions. Free via scraping. *Volume: 200-300 discussions*

- **Source 7: Technical Blogs - Acquired Companies**
  Engineering blogs from companies pre/post acquisition (Figma, GitHub, Slack). Reveals authentic tech decisions and challenges. Free. *Volume: 50-100 blog posts*

5. Month 1: Week-by-Week Implementation Plan

Week 1: Data Collection & Environment Setup

Days 1-2: GCP Setup & Data Infrastructure

- ■ Create GCP project: blue-whale-pe-dd
- ■ Enable APIs: Vertex AI, Cloud Storage, Cloud Functions, Secret Manager
- ■ Set up billing alerts ($50, $100, $150 thresholds)
- ■ Create Cloud Storage buckets: bw-training-data-raw, bw-training-data-processed, bw-model-artifacts
- ■ Set up local Python virtual environment with required dependencies

Skills Required: GCP console navigation (beginner), Python environment management (intermediate), Basic cloud storage concepts (beginner)
Budget: $0 (setup only)

Days 3-5: Data Collection Sprint

Day 3: SEC Filings collection using sec-edgar-downloader for 10 target companies (SNOW, PLTR, U, DDOG, NET, CRWD, ZM, DOCU, TWLO, MDB). Extract ~30 filings focusing on Risk Factors and Business sections.

Day 4: VC/PE Reports scraping from Bessemer Cloud 100, a16z enterprise content, Sequoia portfolio analyses. Target 40-50 reports via manual + automated collection.

Day 5: GitHub repository analysis for companies acquired 2020-2024 (Figma, GitHub, Slack, HashiCorp, Docker, MongoDB, Elastic). Generate 50 repo analyses with code quality metrics.

Skills Required: Web scraping (intermediate), API usage (beginner-intermediate), Data parsing (intermediate)
Budget: $0-10

Days 6-7: Synthetic Data Generation

Generate 300 realistic technical due diligence reports using Claude/GPT-4 API based on diverse company profiles. Create structured prompts that ensure consistent format with varying tech stacks, team sizes, revenue levels, and founded dates. Include architecture assessment, technical debt analysis, security posture, scalability concerns, and overall DD scores.

Skills Required: API integration (beginner), Prompt engineering (intermediate), Data validation (intermediate)
Budget: $20-30

Week 1 Deliverable: ■ 800-1,000 training examples collected and stored in Google Cloud Storage

Week 2: Data Processing & Model Fine-Tuning

Days 8-9: Data Preparation

Task 1: Convert all collected data to JSONL format with consistent message structure (system prompt, user input, assistant response).

Task 2: Data quality control - remove duplicates, validate JSON structure, balance examples across score ranges (need representation of low, medium, and high DD scores).

Task 3: Split dataset: 80% training, 10% validation, 10% test. Upload to GCS buckets for Vertex AI access.

Skills Required: Data cleaning (intermediate), JSON manipulation (intermediate), GCS CLI usage (beginner)
Budget: $1-2 (storage)

Days 10-12: Fine-Tuning on Vertex AI

Configure and launch fine-tuning job using Vertex AI Model Garden with Gemma 2 9B as base model. Use parameter-efficient fine-tuning (LoRA) with conservative hyperparameters: 3 epochs, learning rate 2e-5, LoRA rank 16, batch size 8.

- Monitor loss curves in Vertex AI console throughout training
- Validate on held-out set after each epoch to detect overfitting
- Implement early stopping if validation loss increases
- Expected training time: 4-8 hours on Vertex AI infrastructure

Skills Required: ML training basics (intermediate), Vertex AI console (beginner-intermediate), Hyperparameter intuition (beginner)
Budget: $80-150 (compute)

Day 13: Model Evaluation

Quantitative Metrics:
• Mean Absolute Error (MAE) on DD scores < 10 points (100-point scale)
• JSON validity > 95% for structured outputs
• Inference latency within target range (500ms-10sec)

Qualitative Review:
• Manually review 20 test predictions
• Check for hallucinations and validate caveat generation
• Assess narrative quality and coherence

Success Criteria: MAE < 10, JSON validity > 95%, no obvious hallucinations

Skills Required: Model evaluation (intermediate), Python scripting (intermediate)
Budget: $5 (inference)

Week 2 Deliverable: ■ Fine-tuned Gemma 2 9B model deployed to Vertex AI endpoint

Week 3: API Development & Multi-Modal Integration

Days 14-15: Core API Wrapper

Architecture: Cloud Function (Python) → Vertex AI Endpoint (Gemma 2 9B) → [optional] Gemini 1.5 Flash for visuals → Structured + Narrative Response

- HTTP POST endpoint accepting company metadata (name, tech stack, team size, description)
- Optional architecture diagrams for visual analysis
- Chain-of-thought reasoning prompts for explainability
- Dual output: structured JSON scores + narrative Markdown assessment
- Proper error handling and input validation
- Deployed via Cloud Functions with 512MB memory, 120s timeout

Skills Required: Python web development (intermediate), Cloud Functions (beginner-intermediate), API design (intermediate)
Budget: $0 (free tier)

Days 16-17: Multi-Modal Pipeline

Integrate Gemini 1.5 Flash for architecture diagram analysis. Build two-stage pipeline: (1) Gemini extracts technology stack, architecture patterns, scalability concerns, and red flags from visual diagrams, (2) Gemma incorporates visual analysis into holistic technical assessment.

- Create 5-10 sample architecture diagrams (AWS, Azure, GCP patterns)
- Validate Gemini extraction accuracy across different diagram styles
- Refine prompts for consistent structured output from visual analysis
- Test integration between visual and text analysis pipelines

Skills Required: Multi-modal AI usage (beginner), Image processing concepts (beginner)
Budget: $10-15 (Gemini API)

Days 18-19: Chain-of-Thought Implementation

Upgrade Gemma prompts to use explicit chain-of-thought reasoning. Structure prompts with step-by-step analysis: (1) Architecture Analysis, (2) Team Assessment, (3) Technical Debt, continuing through all 8 dimensions, then (4) Final Synthesis explaining how the overall score was determined.

- More explainable outputs for PE decision-makers
- Better accuracy on edge cases and ambiguous scenarios
- Easier debugging of reasoning errors
- Natural caveat and assumption generation

Skills Required: Prompt engineering (intermediate-advanced), Model behavior analysis (intermediate)
Budget: $5 (testing)

Day 20: Testing & Documentation

API Testing: Create test suite with diverse company profiles, validate response structure, measure latency, check error handling.

Documentation: Complete API specification (OpenAPI/Swagger format), usage examples with curl commands, response schema definitions, troubleshooting guide.

Skills Required: API testing (intermediate), Technical writing (beginner-intermediate)
Budget: $0

Week 3 Deliverable: ■ Working API with multi-modal support and chain-of-thought reasoning

Week 4: Deal Sourcing (Use Case A) & Integration

Days 21-22: Deal Sourcing Model

Build interactive sourcing flow inspired by boardy.ai model: progressive questioning that qualifies deals before triggering full technical due diligence. Implement smart follow-up logic, red flag detection for automatic rejection, and preliminary scoring to optimize analyst time.

Standard Qualification Questions:

- What problem does your product solve?
- Who are your primary customers?
- What is your current tech stack?
- How many engineers on your team?
- What is your current MRR/ARR?
- What are you raising and at what valuation?
- What are your key technical risks?
- Do you have any patents or proprietary tech?

Skills Required: Conversational AI design (intermediate), Business logic (intermediate)
Budget: $0

Days 23-24: Integration Layer

Connect sourcing qualification to full DD pipeline with external data enrichment. Pipeline stages: (1) Interactive sourcing session, (2) External data enrichment (tech stack from BuiltWith/Wappalyzer, company data from Clearbit, team from LinkedIn), (3) Full tech DD via Blue Whale API, (4) Generate dual-format output reports.

- BuiltWith API for tech stack detection (use free trial for MVP)
- Clearbit for company metadata (free tier available)
- Hunter.io for team email discovery (free tier)

Skills Required: API orchestration (intermediate), Data enrichment (intermediate)
Budget: $0 (free tiers)

Days 25-26: Output Formatting

Structured Output (JSON): Complete data structure with overall score, recommendation (Pass/Flag/Reject), dimension-by-dimension scores, key strengths/risks arrays, caveats list, estimated technical debt cost, timeline to production-ready.

Narrative Output (Markdown): Executive summary, detailed assessment for each dimension, final recommendation with conditions, caveats and unknowns section. Professional formatting suitable for LP presentation.

Skills Required: Document generation (beginner-intermediate), Template design (beginner)
Budget: $0

Day 27: End-to-End Testing

Full pipeline validation: submit 10 test companies through complete sourcing flow, verify data enrichment accuracy, validate DD API responses, check output format consistency, measure total latency (target <10 seconds). Optimize for performance through parallel API calls and caching where appropriate.

Skills Required: System testing (intermediate), Performance optimization (intermediate)
Budget: $5-10 (API calls)

Days 28-30: Documentation, Demo & Handoff

- **README.md:** Architecture overview, setup instructions, API documentation, example usage
- **Demo Video:** 5-7 minute walkthrough of sourcing flow, live API demo, output examples, next steps
- **Lessons Learned Doc:** What worked well, model limitations, data quality issues, v2 recommendations
- **Future Roadmap:** Use Cases C & D implementation plan, retraining strategy, production hardening, cost optimization

Skills Required: Technical writing (intermediate), Presentation (beginner)
Budget: $0

Week 4 Deliverable: ■ Complete MVP with sourcing + tech DD, documented and demo-ready

6. Detailed Budget Breakdown

Category	Item	Est. Cost
Data Collection		
	Synthetic data generation (GPT-4/Claude API)	$25
	Premium data sources (optional)	$20
Model Training		
	Vertex AI fine-tuning (Gemma 2 9B, 3 epochs)	$120
	Cloud Storage (GCS)	$2
Inference		
	Vertex AI endpoint (25 queries/day × 30 days)	$8
	Gemini 1.5 Flash (visual analysis, ~50 calls)	$15
Development & Testing		
	Cloud Functions (free tier)	$0
Contingency		
	Iterations, debugging, additional training	$50
TOTAL		**$245**
REMAINING	**Available for overruns or C&D**	**$255**

7. Skills Required Summary

Essential Skills (Must Have)

- **Python (intermediate):** Core development language for all scripts and API
- **GCP fundamentals (beginner):** Console navigation, basic service configuration
- **API usage (beginner):** REST APIs, authentication, request/response handling
- **Git/version control (beginner):** Code management and collaboration

Recommended Skills (Can Learn As You Go)

- **ML basics (beginner):** Understanding training, inference, evaluation metrics
- **Prompt engineering (intermediate):** Optimizing LLM outputs for specific tasks
- **Web scraping (intermediate):** Data collection from various sources
- **JSON/data manipulation (intermediate):** Parsing, transforming structured data

Optional Skills (Tools Can Compensate)

- Statistical analysis for detailed model evaluation
- Frontend development for future UI needs
- DevOps for production deployment (not needed for MVP)

8. Risk Mitigation Strategies

Risk	Mitigation Strategy
Model doesn't learn from training data	Start with simpler model (Gemma 2B), reduce task complexity, verify data quality early
Budget overruns from fine-tuning	Use LoRA parameter-efficient tuning, early stopping, limit to 3 epochs maximum
Poor output quality	Strict output parsing, structured JSON mode, validation layers, extensive testing
Data collection takes too long	Prioritize synthetic data, use GPT-4 to generate examples, skip low-value sources
API latency too high	Cloud Functions gen2 (faster cold start), optimize prompts, implement caching
Insufficient training data	Use few-shot prompting instead of fine-tuning, leverage base model knowledge

9. Success Criteria Checklist

At the end of Month 1, you should have:

Functional Requirements

- ■ API endpoint accepting company data (name, stack, team size, description)
- ■ Sourcing question flow implemented with smart follow-ups
- ■ Tech DD analysis producing 0-100 score across 8 dimensions
- ■ Both structured (JSON) and narrative (Markdown) outputs
- ■ Multi-modal capability for basic architecture diagram analysis
- ■ Chain-of-thought reasoning integrated into prompts
- ■ Response time < 10 seconds for typical request

Quality Requirements

- ■ 90%+ accuracy on test set (MAE < 10 points on 100-point scale)
- ■ No obvious hallucinations in outputs
- ■ Caveats and unknowns properly flagged in all assessments
- ■ JSON output validates against defined schema

Documentation

- ■ Complete API documentation with examples
- ■ README with setup and deployment instructions
- ■ Sample requests and responses for testing
- ■ Lessons learned document for iteration planning

Budget

- ■ Total Month 1 spend < $500
- ■ Monthly operational cost projection < $50

10. Data Sources Reference List

High-Quality Paid Sources (Future Scaling)

- **Gartner:** Tech DD frameworks, M&A; assessments. $30K+ annual. Gold-standard methodologies.
- **Forrester:** Technology research, vendor assessments. $15K-50K annual. Detailed tech stack analyses.
- **451 Research (S&P; Global):** Tech M&A; database. $25K+ annual. Historical acquisition data.
- **BuiltWith Pro:** Website tech stack detection. $295/month. Real-time data for 1B+ sites.
- **Wappalyzer:** Technology profiler. $149/month. API access to tech stack data.
- **Black Duck (Synopsys):** Open source security analysis. Custom pricing. Code risk assessment.

Free/Low-Cost Sources (Use Now)

- **SEC EDGAR:** Public company filings. Free via sec-edgar-downloader library.
- **Crunchbase:** Startup funding, acquisition data. $29-99/month. Acquisition histories.
- **GitHub:** Open source repos from acquired companies. Free. Actual code quality metrics.
- **AngelList:** Startup profiles and tech stacks. Free. Self-reported information.
- **Reddit/HackerNews:** Community discussions. Free. Practitioner insights.
- **Engineering Blogs:** Company tech blogs (Uber, Netflix, Airbnb). Free. Authentic challenges.
- **LinkedIn Sales Navigator:** Team composition analysis. $99/month. Growth and turnover data.

11. Code Repository Structure

```
blue-whale-pe-dd/ ■■■ README.md ■■■ requirements.txt ■■■ .gitignore ■■■ data/ ■ ■■■ raw/ ■
■ ■■■ sec_filings/ ■ ■ ■■■ vc_reports/ ■ ■ ■■■ github_analyses/ ■ ■ ■■■ synthetic/ ■ ■■■
processed/ ■ ■ ■■■ train.jsonl ■ ■ ■■■ val.jsonl ■ ■ ■■■ test.jsonl ■ ■■■ sources.md ■■■
scripts/ ■ ■■■ data_collection/ ■ ■ ■■■ sec_scraper.py ■ ■ ■■■ vc_report_scraper.py ■ ■
■■■ github_analyzer.py ■ ■ ■■■ synthetic_generator.py ■ ■■■ data_processing/ ■ ■ ■■■
format_converter.py ■ ■ ■■■ quality_control.py ■ ■ ■■■ upload_to_gcs.py ■ ■■■ training/ ■
■■■ fine_tune_gemma.py ■ ■■■ evaluate.py ■■■ api/ ■ ■■■ main.py ■ ■■■ requirements.txt ■
■■■ visual_analyzer.py ■ ■■■ sourcing_agent.py ■ ■■■ report_generator.py ■■■ tests/ ■ ■■■
test_api.py ■ ■■■ test_data_quality.py ■ ■■■ test_model_outputs.py ■■■ docs/ ■ ■■■
API_SPEC.md ■ ■■■ SETUP.md ■ ■■■ LESSONS_LEARNED.md ■ ■■■ FUTURE_ROADMAP.md ■■■ examples/
■■■ sample_requests.json ■■■ sample_outputs/
```

Next Steps After Month 1

Month 2-3 Options (Prioritized)

Option 1: Use Cases C & D Implementation ($200-300 budget)
Risk/cost assessment module analyzing cybersecurity posture, compliance requirements, technical debt remediation costs, and modernization timelines. Strategic alignment analyzer ensuring target's technology strategy aligns with PE investment goals and exit strategy.

Option 2: Model Improvement ($150-250 budget)
Collect real-world feedback from initial usage, retrain with enhanced dataset incorporating actual due diligence results, A/B test prompt variations for optimization, implement retrieval-augmented generation (RAG) for up-to-date technology trends and market intelligence.

Option 3: Production Hardening ($100-200 budget)
Implement authentication and authorization, add rate limiting and abuse protection, enhance error handling and logging, build monitoring dashboards for API health and model performance, create automated alerting for anomalies.

Option 4: User Interface ($300-400 budget)
Simple web UI for non-technical stakeholders, dashboard for tracking deals through pipeline stages, export functionality for reports (PDF, DOCX), integration with existing deal management systems.

Recommendation: Focus on Use Cases C & D if model quality meets success criteria. Focus on Model Improvement if accuracy is borderline or edge cases reveal limitations.

Getting Started - Week 1, Day 1 Action Items

You have confirmed GCP billing access, API keys for Anthropic/OpenAI, time commitment of 20-30 hours/week, and technical comfort with Python debugging. You are ready to begin immediately.

Immediate Actions (Today)

- ■ Create GCP project named 'blue-whale-pe-dd'
- ■ Enable required APIs: Vertex AI, Cloud Storage, Cloud Functions, Secret Manager
- ■ Set up billing alerts at $50, $100, and $150 thresholds
- ■ Clone or create new GitHub repository for project code
- ■ Set up Python virtual environment locally: python -m venv blue-whale-env
- ■ Install initial dependencies: google-cloud-aiplatform, google-cloud-storage, sec-edgar-downloader
- ■ Test SEC scraper on 2-3 companies as validation
- ■ Create GCS buckets: bw-training-data-raw, bw-training-data-processed, bw-model-artifacts

Questions or Issues?

If you encounter any blockers during implementation:

• Refer to the detailed code examples and scripts that will be provided separately
• Consult GCP Vertex AI documentation for service-specific issues
• Use the troubleshooting section in week-specific deliverables
• Document all challenges in your LESSONS_LEARNED.md for future reference