Portfolio Project Module 7: Project Codebase

Alexander Ricciardi

Colorado State University Global

CSC501: Management for the Computer Science Professional

Dr. Brian Holbert

January 4, 2025

Portfolio Project Module 7: Project Codebase

Introduction

This document outlines the Management Plan (MP) for Project Codebase, a proprietary internal system for Omega.py. The MP's objective is to develop and implement an intelligent platform combining an AI Agent with a Retrieval-Augmented Generation (RAG) system to assist software engineers and AI coding agents in visualizing architectures and querying large codebases. The MP describes the project lifecycle from inception to risk management, threading together six planning modules into one narrative. The MP lifecycle begins with System Selection (Module 1), defining the project scope. Followed by the Stakeholder Engagement (Module 2), identifying resistance from Security and Network administrators' stakeholders regarding data leakage and latency. Then it transitions into the structural design phase via the Work Breakdown Structure (Module 3) and Project Design (Module 4). These sections describe and set the scope baseline to prevent gold plating and outline a budget. This is followed by describing the selected Hybrid Design Methodology (Module 5), which utilizes predictive waterfall methods for infrastructure and adaptive agile sprints for AI development. Finally, the Risk Matrix (Module 6) sets the project value by identifying high-severity threats, such as AI hallucinations and integration delays.

System Selection – Module 1

For this Portfolio Project, I have chosen a project management approach to develop and implement a software system called Project Codebase. To meet the CSC501 portfolio project requirements, I have reframed the Project Codebase project as a fictional internal system being developed by a company called Omega.py (a domain currently owned by the project manager). The goal of the project is to manage the development and implementation of an intelligent

platform/AI agent that would assist Omega.py software engineers and AI coding agents in understanding existing and being developed software codebases and architectures.

For software engineers, Project Codebase will provide visual architecture mapping, project workflow visualization, context descriptions, and integration with IDE (Integrated Development Environment) tools. For AI coding agents, Project Codebase will enable AI coding agents to understand the constraints, requirements, and stakeholder needs of software projects, as well as their codebase and architecture. The core feature of Project Codebase is its combination of an AI Agent and RAG (Retrieval-Augmented Generation) system based on a knowledge graph Neo4j database, and on a custom markup language, DE-ML (Description Extractor Markup Language) embedded in code comments and docstrings for component description extraction.

Stakeholder Engagement Assessment – Module 2

A key component of a project's success is stakeholder engagement. This section provides a Stakeholder Engagement Assessment Matrix for the Omega.py Project Codebase, see Figure 1, along with a quick use case study detailing current and desired engagement levels, and strategies to transition stakeholders to their desired engagement level.

The Project Management Institute (PMI) has identified five engagement levels of a stakeholder as follows:

- Unaware: These are stakeholders who are unaware that the project exists.
- Resistant: These stakeholders do not want the project to be a success and may take active steps to ensure its failure.
- Neutral: These stakeholders are indifferent to the success or failure of the project. It makes no difference to them whether the project is a success or failure.

- Supportive: This group of stakeholders is satisfied that the project will be a benefit and want the project to be a success.
- Leading: These stakeholders want the project to be a success and are actively engaged to ensure project success.

(Ucertify, n.d.b, p45)

Figure 1

Stakeholder *Engagement Assessment Matrix*

Stakeholder	Unaware	Resistant	Neutral	Supportive	Leading
Fiona Chen (Legal Counsel)	C			D	
Marcus Thorne (Security Lead)		C		D	
Kevin Brooks (Network Admin)		C		D	
Sam Lopes (Lead Software Engineer/Dev.)				C	D
David Kim (Software Architect)				C	D
Priya Patel (Senior Software Developer)				C, D	
Mark O'Connor (Senior Software Engineer)			C	D	
Sarah Jenkins (Junior Software Developer)	C			D	
Linda Wu (Data Engineers)		C		D	
Elena Vance (Lead AI Engineer)				C	D
Jordan Lee (CEO)				C, D	
Linda Roberts (CFO)			C	D	
Alex Ricciardi (CPO, PM)					C, D

C: Current Engagement level
D: Desire Engagement level

Note: The matrix illustrates the current and desired engagement levels of stakeholders. Created with the LucidChart App.

Our Stakeholder Engagement Assessment Matrix used the PMI-identified levels to categorize the level of engagement of Omega.py stakeholders. Please see the next page for the matrix.

Stakeholder Engagement Assessment Case Study

The main goal of the Program Manager (PM) is to try to have stakeholders at their desired level. Within the Stakeholder Engagement Assessment Matrix, this translates to having all the Cs and Ds in the same box (Ucertify, n.d.b). Thus, the primary purpose of this Engagement Assessment Matrix case study is not only to assess the stakeholders' current and desired engagement levels but also to formulate targeted strategies that will help stakeholders not currently at their desired level transition to their desired engagement level. Below, the study lists the stakeholders from the currently least engaged to the most engaged:

Currently Unaware

- Fiona Chen is Omega.py's Legal Counsel. She is currently unaware that the project exists, as legal counsel is rarely involved in the early planning and development phases of a software project. However, as the project plans to use external AI models (e.g., Gemini or ChatGPT), which introduces issues with licensing and a liability risk due to possibly leaking clients' sensitive data and Omega.py's sensitive data to AI providers, her desired engagement level must be Supportive to formulate contracts of use with AI providers that ensure that Omega.py has strict governance over the data shared with the Ai model, and has intellectual property rights over all the outputs generated by the AI model. To transition Fiona to a Supportive level, she needs to be aware of the project RAG system functionality and goals and briefed about the external AI model's current terms of service related to data confidentiality and security.

- Sarah Jenkins is a junior software developer. She is currently unaware that the project exists. As a junior developer, her main focus is not on planning projects or on helping develop or engineer high-level software structures, but on bug fixes and learning the Omega.py coding

culture and software development practices. Nonetheless, her desired level of engagement needs to be supportive as she is expected to use Project Codebase extensively for coding tasks. To transition Sarah to the Supportive level, she needs to be aware of the project and train on DE-ML (Description Extractor Markup Language) is being developed, as well as how to test/use the Project Codebase during its various developing phases.

Currently Resistant

- Marcus Thorne is the security lead for Omega.py. He is currently resistant to the project. As a security lead, he is responsible for reducing risks and protecting Omega.py's systems, sensitive information, data, intellectual property, and facilities. In consequence, he sees the introduction of an external system, such as an external AI model having access to client and internal data, as well as network devices, as a security risk that can potentially result in massive "Data Leakage," a security breach of Omega.py systems. His desired engagement level must be Supportive for the project to be viable. To transition Marcus to a Supportive level, he must be actively included in planning, implementing, and reviewing security protocols, safety features, and tools securing the Project Codebase functionality and surrounding infrastructure.

- Kevin Brooks is the Omega.py network administrator. He is currently resistant to the project. As a network administrator, he worries about the network infrastructure's performance, reliability, and security. Introducing a system like Project Codebase that integrates an RAG system, servers for databases, and external API calls to AI model providers will increase network traffic and latency risks, as well as authentication and security issues. His desired engagement level must be Supportive for the project to be effectively and safely integrated within Omega.py infrastructure. To transition Kevin to a Supportive level, he must be involved in the planning and

implementation of local or cloud-based Neo4j databases, the API gateway configuration, and Project Codebase authentication protocols.

- Linda Wu is Omega.py data engineer. She is currently resistant to the project. She is responsible for maintaining Omega.py's data databases and the overall data pipeline. She is wondering why, in addition to existing SQL databases, which are a type of relational database, there is a need for Graph Databases, which are also a type of relational database. In other words, she views the implementation of new Neo4j Graph Databases to support Project Codebase alongside existing SQL databases as potentially unnecessary, increasing the complexity and overhead of the data pipeline. Her desired engagement level must be Supportive for the project to be viable. To transition Linda to a Supportive level, she needs to be briefed about why Graph Databases are necessary for the efficient functioning of Project Codebase. For example, by demonstrating how an AI agent can autonomously perform schema mapping, data injection, and query a Graph database within an RAG system. She should also have ownership of the Knowledge Graph template schema that Project Codebase AI needs to follow; this will further her engagement with the project.

Currently Neutral

- Mark O'Connor is an Omega.py senior software engineer. He is currently neutral to resistant to the project. As an experienced developer, he has seen many utility tools come and go. His neutrality on Project Codebase is based on a wait-and-see approach. However, he is skeptical about the value of "visual mapping architecture" using a tool, when manually creating UML diagrams is sufficient in most cases. Additionally, he feels uncomfortable about using AI; however, he is impressed by how much AI coding agents have improved in recent months, making them useful tools in specific cases. His desired engagement level is Supportive, as he

needs to be part of the Project Codebase developing team. To transition Mark to a Supportive level, he needs to be further exposed to AI coding agents and to the abilities of AI to understand and visually represent systems. By enrolling him in RAG training/bootcamp through Neo4j and by giving him ownership of a pilot Project Codebase RAG system.

- Linda Roberts is Omega.py Chief Financial Officer (CFO). She is currently neutral on the project. She is aware of the project's existence, but only from a financial perspective; she is only aware of the estimated cost of implementing the project, such as forecasted AI API tokens, cloud hosting/local hosting, and estimated development hours. She is waiting to see (further analysis) if the project yields real value in the long term for the organization. Her desired level is Supportive. Presenting to her a cost-benefit analysis of the project will show how the implementation of an AI agent system can reduce the training time for new hires (like Sarah Jenkins) and decrease the time senior engineers spend on documentation and troubleshooting issues. Showing that implementing such a system is very valuable to the organization because it translates employee workflow efficiency into dollar savings.

Supportive Moving to Leading

- Sam Lopes is Omega.py lead software engineer, David Kim is Omega.py software architect, and Elena Vance is the Omega.py lead AI engineer. They are currently supportive of the project. They were all part of the initial exploratory team for the project. They see the value of the project. However, they have been assigned to lead the project and need to be informed of and empowered to assume their leading role.

Supportive and Lead Already Aligned

- Priya Patel is an Omega.py senior developer. He is currently aligned at a Supportive level. He is an AI enthusiast, seeing the value of the project.

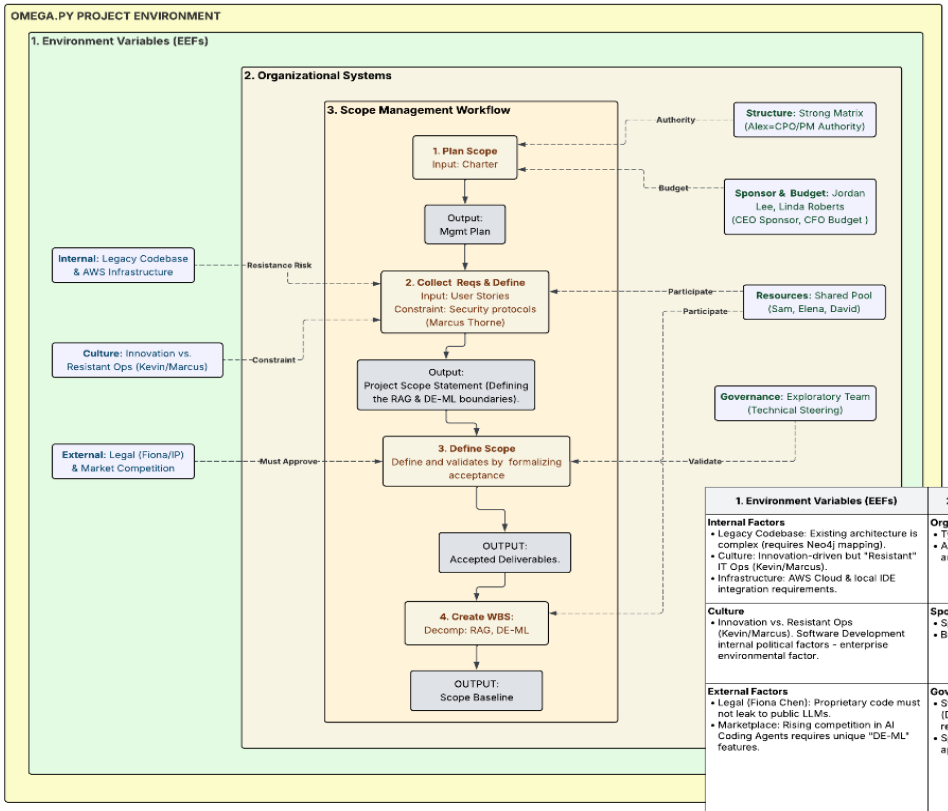
- Jordan Lee is Omega.py CEO. The Project Codebase exploratory team presented him with a Project Codebase proposal. He is very supportive of the project, seeing the value in it. He is currently aligned at a Supportive level.

- Alex Ricciardi is Omega.py Chief Product Officer (CTO) and Program Manager (PM) of Project Codebase. He is the source of the Project Codebase idea and the main cheerleader. He is currently aligned at a Leading level.

Stakeholder Initial WBS - Module 3

Figure 2

Initial Project Codebase PMP



1. Environment Variables (EEFs)	2. Organizational Systems (Governance)	3. Scope Processes (Workflow)
Internal Factors • Legacy Codebase: Existing architecture is complex (requires Neo4j mapping). • Culture: Innovation-driven but "Resistant" IT Ops (Kevin/Marcus). • Infrastructure: AWS Cloud & local IDE integration requirements.	**Organizational Structure** • Type - Strong Matrix. • Alex Ricciardi acts as both CPO and PM, high authority over resources (Sam, Elena).	**Step 1: Plan Scope** • Input: Project Charter. Action: CEO Authorization (completed), initial budget • Action: Define how the scope will be validated by the "Exploratory Team." • Output: Plan for the Scope Management Plan.
Culture • Innovation vs. Resistant Ops (Kevin/Marcus). Software Development internal political factors - enterprise environmental factor.	**Sponsor & Budget** • Sponsor - Jordan Lee (CEO). • Budget - Linda Roberts	**Step 2: Collect & Define** • Input: Plan for the Scope Management Plan and User Stories from Software Engineers. • Constraint: Security protocols (Marcus Thorne). • Output: Project Scope Statement (Defining the RAG & DE-ML boundaries).
External Factors • Legal (Fiona Chen): Proprietary code must not leak to public LLMs. • Marketplace: Rising competition in AI Coding Agents requires unique "DE-ML" features.	**Governance Framework** • Steering Committee - The "Exploratory Team" (David, Sam, Elena) acts as the technical review board. • Sponsor - Jordan Lee (CEO) provides budget approval. **Resources: Shared Pool** • Leads: Sam (Software), Elena (AI), David (Architect). • Dev Team: Priya, Mark, Sarah (Developers). • Data: Linda Wu (Data Engineer).	**Step 3: Define Scope** • Input: Project Scope Statement (Defining the RAG & DE-ML boundaries). • Key Branches: RAG Agent, DE-ML Parser, Knowledge Graph. • Validates: - Action: Formalizing acceptance (Obtaining Sign-off). - Detail: Reviewing deliverables with the "Exploratory Team" (Elena/Sam) to ensure they meet the acceptance criteria. Output: Accepted Deliverables (Signed-off). **Step 4: Create WBS** • Input: Accepted Deliverables. • Decomposition: RAG, DE-ML, ect. • Create: WBS Dictionary. • Output: Scope Baseline

Note: The diagram illustrates how the project will be managed, and the table gives more details to support the diagram.

The work breakdown structure (WBS) is a hierarchical graphical representation that shows all the work that needs to be accomplished on the project (Ucertify, n.d.d). It is part of the Project Management Plan (PMP) scope baseline artifact that contains the scope statement, WBS, and WBS dictionary (Ucertify, n.d.c). This section provides the WBS initial components breakdown and discusses how the WBS will work in the overall process of managing the development and implementation of the Omega.py Project Codebase.

As shown in Figure 2, the project uses the technique of decomposition. In the table, it can be found in "3. Scope Processes (Workflow) - Step 4: Create WBS." This technique consists of breaking down the scope into work packages, creating the initial WBS component. For Project Codebase WBS initial component breakdown can be listed as follows:

1.0 Plan Project Management:

- 1.1 Finalize Project Charter (completed).
- 1.2 CEO Authorization (completed), initial budget.
- 1.3 Define how the scope will be validated by the "Exploratory Team."
- 1.4 Create Scope Management Plan.

2.0 Infrastructure & Security - Addressing "Resistance Risk":

- 2.1 AWS Cloud Environment Setup.
- 2.2 Neo4j Knowledge Graph Instance Provisioning.
- 2.3 Legal & IP Compliance Audit.

3.0 Core AI Development:

- 3.1 Define DE-ML (Description Extractor Markup Language) Syntax.
- 3.2 Develop RAG Ingestion Pipeline.

4.0 Interface & Client Integration

- 4.1 IDE Plugin Development (VS Code):
    - 4.1.1 Authentication Module (LDAP/SSO).
    - 4.1.2 Context Menu Actions ("Explain Code").

    - 4.1.3 Inline Comments Renderer (Ghost text).
- 4.2 Architecture Visualization Mapper:
    - 4.2.1 Graph Rendering Engine (Frontend).
    - 4.2.2 Interactive Navigation (Click-to-Expand).

5.0 System Integration & Build

- 5.1 RAG Pipeline Integration: Retriever-Generator Link logic.
- 5.2 Description Extractor Integration: Parser-Database Hook.
- 5.3 Security Integration: PII Redaction Layer (Satisfying Security Lead requirements).

WBS Integration

The WBS goal is to control and validate the scope. It integrates within the project management process by breaking down the scope into works; for example, the “Infrastructure & Security” processes are separated into three different scheduled work sessions. By integrating with the Project Environment, it identifies environmental variable constraints, and it ensures that the security requirements (demanded by “Resistant” stakeholders) are addressed and implemented, rather than just being treated just has external constraints. It prevents Scope Creep and Gold Plating by clearly defining each work; for example, by limiting the “Interface & Client Integration” Project Codebase to be strictly a VS Code Plugin, excluding unapproved features such as being a browser plugin. It integrates with the project management process by aligning its work definitions with Omega.py System for Value Delivery; for example, by prioritizing the “Core AI Development” work phase to ensure that the Project Codebase end product will be a useful tool for software development, delivering the intended value of converting software engineering efficiency into financial gain.

Project Codebase Design and Outline - Module 4

The objective of the Project Codebase PMP is to manage the development and implementation of an intelligent AI agent and RAG (Retrieval-Augmented Generation) system to assist Omega.py software engineers in designing and implementing software applications. Project Codebase's features include visual architecture mapping, project workflow visualization, context descriptions, and integration with IDE (Integrated Development Environment) tools, as well as a custom markup language called DE-ML (Description Extractor Markup Language) integrated within code comments and docstrings for component description extraction. These features' functionality will be powered by Project Codebase AI's own agent to enable AI coding agents and software engineers to understand the constraints, requirements, and stakeholder needs of software projects, as well as their codebase and architecture. This section provides an outline of Project Codebase PMP, including the project scope and timeline estimates for inception, delivery, and maintenance, as well as discussing possible change requests and goal alignments.

Project Scope - Project Scope Statement

The project aligns with Omega.py's values of System for Value Delivery, which state that the value provided by a software is the ultimate indicator of project success or failure. Based on these values, the project's core objective is to develop a sophisticated AI agent and RAG system that will assist Omega.py software engineers and AI coding agents in understanding software codebases and architectures. The goal of the project is to deliver business value by translating employee workflow efficiency into financial savings.

Project requirements

Please refer to the table within Figure 2 for a detailed breakdown of the work packages and scope components. The following is the core requirements list:

- Knowledge Graph Database - Implementation of a Neo4j database to map the codebase and architecture of a software application.
- Project Codebase AI Agent - A customized AI backend agent for Project Codebase to assist software engineers and AI coding agents capable of querying Neo4j knowledge graph database, understanding constraints, requirements, and codebase context.
- RAG System - Manage the Neo4j knowledge graph database and RAG Ingestion Pipeline used by the Project Codebase AI Agent.
- Description Extractor Markup Language (DE-ML) – integrated within the RAG for code comment parsing, helping extraction by describing codebase components.
- VS Code Plugin – Project Codebase needs to be deployed as a ‘chat’ extension within VS Code.
- Programming Language Requirements
    - Python – Required for AI Development, which involves AI Agents and RAG pipelines.
    - TypeScript / JavaScript - Required for IDE Plugin Development.
    - Cypher (Query Language) - Required for creating, maintaining, and querying, Neo4j Knowledge Graph database.
- DE-ML (Proprietary Syntax): The project explicitly mandates the creation of a custom syntax called Description Extractor Markup Language (WBS 3.1) to be embedded within code comments.
- Exclusions - Browser functionality, mobile applications, or standalone desktop applications are currently out of scope.

Scope Baseline and WBS

The scope baseline artifact's primary goal is to control the scope by preventing scope creep and gold plating. The scope baseline includes components, which are the Scope Statement, the Work Breakdown Structure (WBS), and the WBS Dictionary (not provided). The Scope Statement was provided at the beginning of the Project Scope section. Figure 2 provides a decomposition of the scope into work packages, creating the initial WBS component.

Development Methodology and Project Schedule (Timeline)

A hybrid/adaptive model methodology was selected for this project; additionally, the Project Schedule (timeline) integrates this hybrid methodology within the project life cycle. Therefore, the WBS follows the project life cycle phases.

Table 1

Hybrid Development Approach

Project Phase	WBS Alignment	Development Approach	Delivery	Key Deliverables
Inception/ Planning	1.0, Initial 2.0	Predictive/ Sequential	Single Document	Establish the baselines.
Core AI Delivery	3.0, 4.0	Adaptive/Iterative	Periodic 2-week Sprints	MVP (Minimum Viable Product) increments.
Integration/ Deployment	5.0	Hybrid/Incremental	Multiple Modules/ Features)	Release tested functionality incrementally (e.g., Auth Module 4.1.1) to users.
Maintenance/ Operations	Post-Project	Sustaining Operations	Continuous/ On-Demand	Feature refinement (Product Backlog Grooming) and continuous defect repair as part of the product life cycle.

Note: The table illustrates the hybrid development approach integrated within the WBS.

Hybrid/Adaptive Model

The hybrid/adaptive model is a hybridization between the classic predictive model (waterfall) and the more modern Agile model. The predictive model (waterfall) was chosen for well-defined tasks and sequentially driven phases, such as the Plan Project Management phase (WBS 1.0) and the Infrastructure/Security (WBS 2.0) phase for legal and infrastructure compliance constraints. On the other hand, the adaptive model (Agile/iterative) was chosen for core development, such as the AI development (WBS 3.0, 4.0), which involves high technical complexity, unknown, and uncertainty related to the development and implementation of AI systems. The Agile model is well-suited for managing projects with high levels of unknown and uncertainty; its adaptivity approach will help the team to quickly adapt to changes caused by unknown and uncertain factors. Furthermore, its iterative approach, utilizing short sprints, will help break down the technical complexity of this project, making it more manageable. Therefore, as shown in Table 1, this project life cycle follows this hybrid approach.

Project Life Cycle Timeline Estimates - Project Schedule

The project total duration is estimated at 44 Weeks, about 11 months, with a buffer of 2 weeks. Note that the maintenance phase has a life duration of 3 to 5 years. See Table 2 for an illustration of the project life cycle timeline estimates.

Table 2

Project Life Cycle Timeline Estimates

Phase	Activity	Duration	Delivery
Inception WBS 1.0, 2.0	Phase 1: Initiation & Planning	4 Weeks	project charter, stakeholder register, WBS, and environment setup.
Delivery WBS 2.0, 3.0, 4.0	Phase 2: Core Development	32 Weeks	Neo4j setup, AI model training, API development, developing VS Code plugin, and documentation

Integration/ Deployments WBS 5.0	Phase 3: Integration & Deployment	8 Weeks	Deploy VS Code plugin, beta testing with the internal team, and update documentation.
Maintenance/ Handoff	Phase 4: Closure & Handoff	Ongoing: 3-5 Years	Final documentation, ongoing maintenance, user training, transition to ops.

Note: The table illustrates the project life cycle timeline estimates and the phase deliveries.

Change Request Management

To handle major changes, the project management utilizes a formal Change Control Process as defined in Integration Management. The process is as follows.

1. Any stakeholder must submit a Change Request Form describing the changes needed or wanted and their possible business value.
2. Next, the PM (Alex) performs an Impact Analysis, analyzing the potential impact that the requested change can have on the Scope, Schedule, and Cost.
3. The Change Control Board (likely Jordan Lee and the Steering Committee) reviews the request.
4. After a decision is made, the approved changes are integrated within the Project Management Plan; rejected changes are logged.

The Project Plan - Stakeholder Analysis

A Stakeholder Engagement Assessment Matrix was used to assess \stakeholder engagement, see Figure 2. The matrix used the PMI-identified levels to categorize the level of engagement. The matrix is a case study that illustrates the stakeholders' current and desired engagement levels

Responsibility Assignment

A Responsibility Assignment Matrix (RAM) using the RACI chart is used as an organizational tool for clarifying the team member roles at the work package level. See Table 3

for a High-Level RACI Matrix. Note that the RACI matrix uses dual accountability that can be used to change a team member from resistance into ownership.

Table 3

High-Level RACI Matrix

WBS Component	Alex Ricciardi - PM/CPO	Elena Vance -Lead AI Eng.	David Kim – Architect	Marcus Thorne - Security	Linda Wu - Data Eng	Sam Lopes - Lead SW Eng.
1.0 Plan Project Management	A	C	C	I	I	C
2.0 Infrastructure & Security	I	C	R	A	C	I
3.0 Core AI Development	I	A, R	C	I	A	C
4.0 Interface & Client Integration	I	C	A	I	I	R
5.0 System Integration & Build	A	R	C	C	C	R
WBS 2.3 Legal/IP Audit	A	I	I	R	I	I
R = Responsible		A = Accountable				
C = Consulted		I = Informed				

Note: The matrix illustrates a High-Level RAM using the RACI format to clarify team member roles within WBS components.

Project Budget (Cost Management)

The project budget was created using Bottom-Up Estimating, aggregating costs from the individual work packages defined in the WBS. Note that the total Budget at Completion (BAC) allocates funds across three categories, which are Direct Labor, Infrastructure/Materials, and Risk Reserves. See Table 4 for a preliminary project budget.

Table 4

Preliminary Project Budget Estimates

Cost Category	Item / WBS Reference	Estimate	Justification / Strategy
Direct Labor	Core Development Team	$240,000	3,000 hours @ $80/hr Team: Elena, Sam, David, Priya
Direct Cost (Training)	Resource Acquisition Strategy	$8,500	Mandatory RAG/Neo4j Bootcamp for Mark O'Connor -> Risk Mitigation and DE-ML training for Sarah Jenkins.
Infrastructure	AWS Cloud & Neo4j Licenses	$12,000	WBS 2.0 Environment setup and hosting for Dev/Test.
Variable Costs	AI API Tokens (Gemini/OpenAI)	$15,000	Volatile Cost -> Estimated usage for testing RAG ingestion and inference.
Reserves	Contingency Reserve (10%)	$27,550	Allocated for Known Risks -> token price fluctuation, integration delays
Total	Budget at Completion (BAC)	$303,050	

Note: The table illustrates the preliminary project budget estimates broken down by cost category, item justification, and strategy. It also includes a contingency reserve to mitigate known risks such as price fluctuations and integration delays.

Budget Brief Description

Volatile Cost Management: The Variable Costs for AI API tokens, $15,000, represent a high-volatility estimate. Because Large Language Models (LLMs) cost of use is consumption-based and prone to change. This will be monitored weekly during the Adaptive/Iterative phase (WBS 3.0) to prevent overrun. Contingency Reserve: A 10% reserve, $27,550, addresses "Known Risks," such as potential price fluctuations in cloud infrastructure

Section Brief Summary

This PMP Design and Outline establishes a management framework for the successful development and implementation of Omega.py Project Codebase. By adopting a Hybrid Methodology, the plan balances legal and security compliance with the agility and adaptability

required to develop and implement AI systems. By integrating specific Stakeholder Engagement Strategies, such as the “Dual Accountability” in the RACI matrix and funded training in the Budget, the plan addresses resistance from key technical leaders. This approach will facilitate the delivery of a functional and sophisticated AI agent and RAG system that will translate technical efficiency into long-term financial value for Omega.py.

Design Methodology Report - Module 5

A project methodology is an approach or framework that outlines the structure and processes used to execute the project work and deliver the final result (Ucertify, n.d.a). This report provides an outline of the processes required for the design methodology, more specifically, Omega.py Project Codebase, as well as a Gantt chart of the full project schedule (44 Weeks), including detailed integration phase activities. The Project Codebase utilizes a hybrid methodology that blends predictive and adaptive approaches. The Predictive (sequential) approach is used where requirements are stable, and compliance to constraints is high, such as for software infrastructure, security, and legal compliance. On the other hand, the adaptive (Agile/iterative) approach is used where uncertainty and complexity are high, such as for developing and integrating AI features and the RAG system.

Processes Required for the Design Methodology - Approach Selection Process

The processes start by selecting a design methodology that is best suited for the project environment, the project purpose, and the project complexity. For Project Codebase, the selection of the hybrid approach was based on the following project variables:

- Degree of innovation of the project - Project Codebase involves developing a markup language (DE-ML), a custom AI agent, and a RAG system. These technical requirements demonstrated a high degree of innovation, but also significant technical uncertainty due to their

novel nature. For such a highly innovative project, an adaptive approach, such as Agile/Scrum approach, is required to address evolving changes associated with innovative projects.

- Regulatory and security constraints - Project Codebase must follow strict security protocols regarding code access, and Personally Identifiable Information (PII) needs to be removed from data due to external AI used. Projects operating within strict oversight environments are better suited for a predictive approach to ensure compliance documentation is generated sequentially (Ucertify, n.d.a).

- Project complexity - Project Codebase is highly complex software that integrates a backend/API, including a Neo4j graph database with real-time AI power retrieval and an AI agent performing codebase and context analysis. Additionally, it integrates a frontend VS Code plug-in, including a visual interface illustrating the codebase and structure of the software being developed, as well as an interface for Omega.py engineers to interact with Project Codebase's own AI Agent for code and architecture analysis. The complexity of this project is best handled by an iterative approach (Agile/Scrum), breaking down a large, complex problem into more manageable modular components.

For all the reasons listed above, a hybrid methodology was selected for Project Codebase.

Project Life Cycle Definition

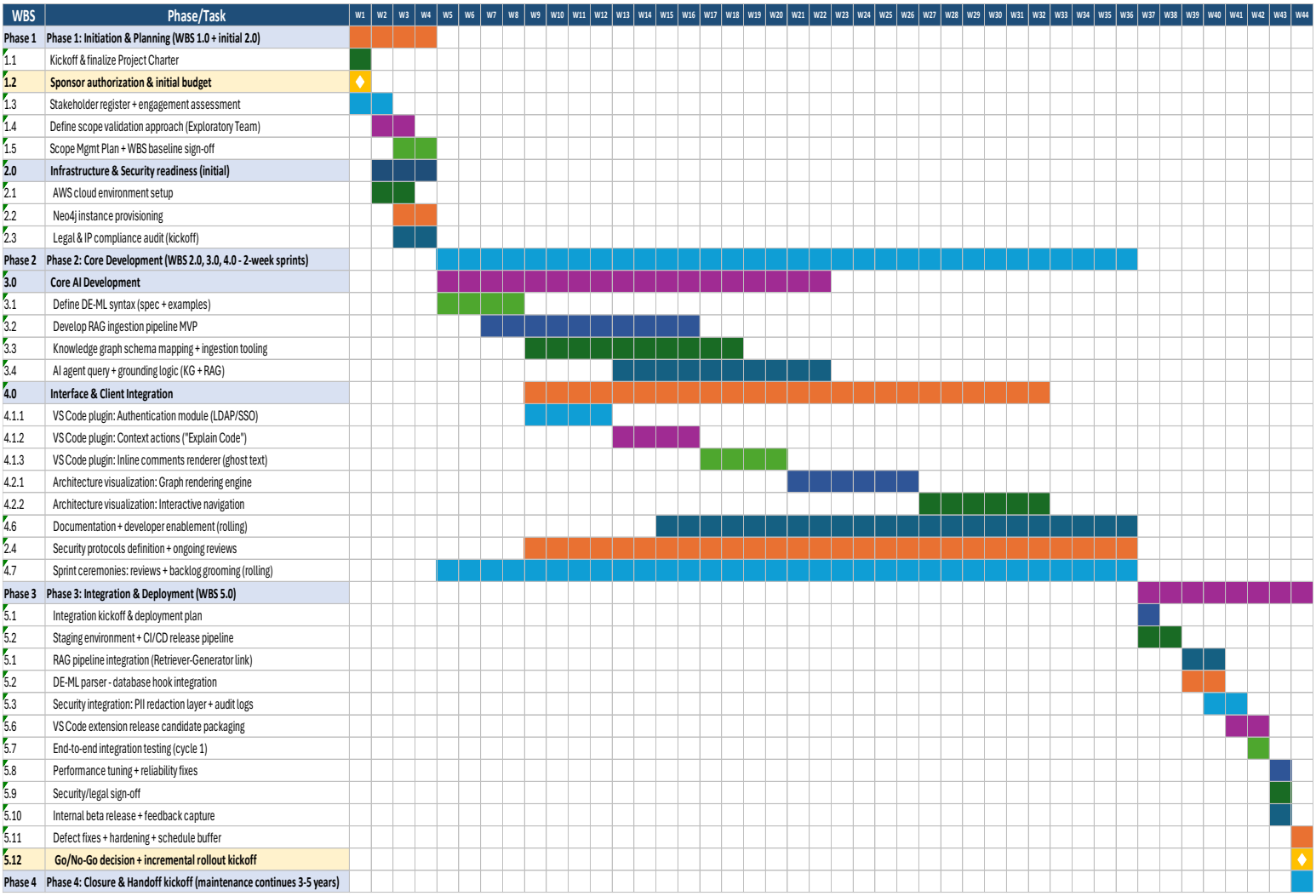
The next phase is to set the project life cycle. The project life cycle is structured into logically related phases that include project activities that end with the completion of all the tasks associated with these activities. The Project Codebase life cycle is divided into four process phases integrated within the Work Breakdown Structure (WBS). Please see Table 1 for an overview of the phases and Figures 3 and 4 for a Gantt chart with a detailed breakdown of the phases, as well as the attached Excel spreadsheet.

Project Schedule: Gantt Chart (44 Weeks)

A Gantt Chart is a horizontal bar chart used in project management to illustrate a project schedule. It is a visual representation of the project's Work Breakdown Structure (WBS) over time. Figure 3 illustrates the full 44-week project schedule, with a focus on the Integration Phase (WBS 5.0) tasks.

Figure 3

Gantt Chart Full Project Timeline



Note: The figure illustrates Project Codebase's full 44-week timeline using a Gantt chart with a detailed integration phase.

Figure 4

Project Schedule

WBS	Phase	Phase/Task	Owner	Start (Week)	Duration (Weeks)	End (Week)
Phase 1	**Inception/Planning**	**Phase 1: Initiation & Planning (WBS 1.0 + initial 2.0)**	**PM + Core Team**	**1**	**4**	**4**
1.1	Inception/Planning	Kickoff & finalize Project Charter	PM (Alex) + Sponsor	1	1	1
1.2	**Inception/Planning**	**Sponsor authorization & initial budget**	**CEO/CFO**	**1**	**1**	**1**
1.3	Inception/Planning	Stakeholder register + engagement assessment	PM	1	2	2
1.4	Inception/Planning	Define scope validation approach (Exploratory Team)	PM + Steering	2	2	3
1.5	Inception/Planning	Scope Mgmt Plan + WBS baseline sign-off	PM + Sponsor	3	2	4
2.0	**Inception/Planning**	**Infrastructure & Security readiness (initial)**	**Ops/Sec**	**2**	**3**	**4**
2.1	Inception/Planning	AWS cloud environment setup	Network Admin + DevOps	2	2	3
2.2	Inception/Planning	Neo4j instance provisioning	Architect + Data Eng	3	2	4
2.3	Inception/Planning	Legal & IP compliance audit (kickoff)	Legal Counsel	3	2	4
Phase 2	**Core Development**	**Phase 2: Core Development (WBS 2.0, 3.0, 4.0 - 2-week sprints)**	**Engineering Team**	**5**	**32**	**36**
3.0	**Core Development**	**Core AI Development**	**AI + Backend**	**5**	**18**	**22**
3.1	Core Development	Define DE-ML syntax (spec + examples)	AI Lead + Architect	5	4	8
3.2	Core Development	Develop RAG ingestion pipeline MVP	AI Lead + Data Eng	7	10	16
3.3	Core Development	Knowledge graph schema mapping + ingestion tooling	Architect + Data Eng	9	10	18
3.4	Core Development	AI agent query + grounding logic (KG + RAG)	AI Lead	13	10	22
4.0	**Core Development**	**Interface & Client Integration**	**Frontend + Backend**	**9**	**24**	**32**
4.1.1	Core Development	VS Code plugin: Authentication module (LDAP/SSO)	Lead Dev	9	4	12
4.1.2	Core Development	VS Code plugin: Context actions ("Explain Code")	Lead Dev	13	4	16
4.1.3	Core Development	VS Code plugin: Inline comments renderer (ghost text)	Lead Dev	17	4	20
4.2.1	Core Development	Architecture visualization: Graph rendering engine	Frontend Dev	21	6	26
4.2.2	Core Development	Architecture visualization: Interactive navigation	Frontend Dev	27	6	32
4.6	Core Development	Documentation + developer enablement (rolling)	PM + Leads	15	22	36
2.4	Core Development	Security protocols definition + ongoing reviews	Security Lead	9	28	36
4.7	Core Development	Sprint ceremonies: reviews + backlog grooming (rolling)	Scrum/PM	5	32	36
Phase 3	**Integration/Deployment**	**Phase 3: Integration & Deployment (WBS 5.0)**	**Engineering + Ops**	**37**	**8**	**44**
5.1	Integration/Deployment	Integration kickoff & deployment plan	PM + Leads	37	1	37
5.2	Integration/Deployment	Staging environment + CI/CD release pipeline	DevOps	37	2	38
5.1	Integration/Deployment	RAG pipeline integration (Retriever-Generator link)	AI + Backend	39	2	40
5.2	Integration/Deployment	DE-ML parser - database hook integration	Backend	39	2	40
5.3	Integration/Deployment	Security integration: PII redaction layer + audit logs	Security + Backend	40	2	41
5.6	Integration/Deployment	VS Code extension release candidate packaging	Lead Dev	41	2	42
5.7	Integration/Deployment	End-to-end integration testing (cycle 1)	QA + Team	42	1	42
5.8	Integration/Deployment	Performance tuning + reliability fixes	Engineering	43	1	43
5.9	Integration/Deployment	Security/legal sign-off	Security + Legal	43	1	43
5.10	Integration/Deployment	Internal beta release + feedback capture	PM + Team	43	1	43
5.11	Integration/Deployment	Defect fixes + hardening + schedule buffer	Engineering	44	1	44
5.12	**Integration/Deployment**	**Go/No-Go decision + incremental rollout kickoff**	**Sponsor + PM**	**44**	**1**	**44**
Phase 4	**Closure/Handoff**	**Phase 4: Closure & Handoff kickoff (maintenance continues 3-5 years)**	**PM + Ops**	**44**	**1**	**44**

Note: The figure illustrates Project Codebase's full 44-week schedule with a detailed integration phase.

Risk Matrix - Module 6

In risk management, a risk matrix, also called a probability-and-impact matrix, is a tool used to rate and prioritize a project's risks by mapping each risk within using a two-dimensional scoring system based on a probability metric defining the probability that a risk will occur and an impact metric defining the impact on project objectives if it does occur. This report provides a

probability-and-impact matrix for Project Code base, as well as a synopsis of why each identified risk scenario is plausible (trigger context) and what the potential impact would be if the risk occurs.

Risk Matrix

The risk matrix uses a percent-based probability metric and a time/cost/quality impact metric. In a probability-and-impact matrix, each cell value represents a relative risk score, which is calculated as the product of the probability and impact values assigned to the risk (Probability × Impact).

Probability Definitions

The risk matrix probability metric is based on a percent-based ratings style that aligns with the Project Management Body of Knowledge (PMBOK® Guide) (PMI, 2021). Below is a percent-based definition of the different risk levels used to rate Project Codebase risks.

Risk probability level definitions:

- High (H): > 70%
- Medium (M): 30%–69%
- Low (L): < 30%

Note that these levels have presumably been set by stakeholder consensus (Ucertify, n.d.e)

Impact Definitions

Same as the probability definitions, the risk matrix impact metric is based on a Time/Cost time/cost/quality ratings style that aligns with the PMBOK® Guide. Table 5 illustrates the impact level based on the three Project Codebase constraint values: time, cost, and quality, which are used to rate the severity of risk impacts.

Table 5

Risk Impact Level

Impact Level (I)	Time Impact	Cost Impact	Quality Impact
High (H)	> 3 months	> $250K	Significant impact on key functions
Medium (M)	4–12 weeks	$50K–$250K	Some impact on key functions
Low (L)	< 4 weeks	< $50K	Minor impact on key functions

Note: The table lists the different impact levels used to rate Project Codebase risk impacts. The overall impact value is represented as a single value in the risk matrix, as H, M, or L.

Project Risk Rating Descriptions

Table 6 provides risk rating descriptions for each identified Project Codebase risk, including the assigned probability and impact levels, the primary risk owner, and the trigger context (rationale) used to justify the assigned ratings.

Table 6

Risk Rating Descriptions

Risk ID	Risk Scenario	Prob.	Impact	Owner	Justification & Trigger Context
R1	Data leakage and security breach	M	H	Marcus Thorne (Security Lead)	External AI model access + Security Lead explicitly flagging “massive data leakage” potential.
R2	Legal/IP noncompliance	M	M	Fiona Chen (Legal Counsel)	External AI introduces licensing/liability/IP constraints; legal sign-off can block or delay release.
R3	Network latency and reliability issues	M	M	Kevin Brooks (Network Admin)	Network admin anticipates an increase in traffic/latency/auth from RAG + APIs.
R4	Neo4j schema/ingestion complexity is causing rework	M	M	Linda Wu (Data Eng) +	Data Engineer identifies integration risk; Knowledge Graph pipeline complexity can drive 4–12 week slips.

				David Kim (Architect)	
R5	RAG/agent quality and trust issues	H	M	Elena Vance (Lead AI Engineer)	AI/RAG generation results are “unknown/uncertain,” requiring monitoring and control; as hallucination/quality issues are common.
R6	VS Code extension integration delays & adoption issues	M	M	Sam Lopes (Lead Software Engineer)	VS Code delivery + Auth + Packaging are integration-heavy (WBS 4.0); issues often surface late in the lifecycle.
R7	Cost overrun from external AI token usage	M	M	Alex Ricciardi (PM) + Linda Roberts (CFO)	Token costs are labeled “volatile” and monitored weekly; it is a consumption-based pricing that could result in significant budget variance.
R8	Integration phase exceeding the 8-Week integration window	H	M	Alex Ricciardi (PM) + Sam Lopes (Eng. Leads)	Multiple gating activities (CI/CD, security, legal, E2E, go/no-go) happen during the Integration Window (WBS 5.0), creating a high risk of exceeding allocated time.
R9	Stakeholder resistance blocking approvals or asking for rework	M	M	Alex Ricciardi (PM/CPO)	Several key stakeholders are currently categorized as "Resistant" (Security, Network, Data); this may create delays.
R10	Scope creep destabilizes the baseline	M	M	Alex Ricciardi (PM/CPO)	Project warns against “Gold Plating” or expansions beyond the defined VS Code plugin scope.
R11	Skills gap and ramp-up delays	M	M	Alex Ricciardi (PM)	Budget includes funded bootcamp/training, indicating a credible risk of ramp-up delay.

Note: The table illustrates the risk rating descriptions, owners, and justifications for the identified Project Codebase risks.

Risks Matrix

The matrix used for this project is a 3x3 matrix including Severity (columns) and Probability (rows). See Table 7 for the table definition, and for the Project Codebase Risks Matrix, see Table 8.

Table 7

Risk Matrix Definition

Probability \ Severity	**Negligible (L)** *Less-than-minor injury/system damage; easily controlled*	**Moderate (M)** *Moderate-to-severe injury/system damage; requires immediate corrective action*	**Critical (H)** *Death or major system loss; requires immediate termination of activity*
Plausible (H) *Very likely to occur)*	Medium	High	High
Occasional (M) *Likely to have some modicum of occurrence*	Low	Medium	High
Implausible (L) *Very unlikely to occur*	Low	Low	Medium

Note: The table illustrates, based on probability and severity, the overall risk level. From "Module 6: Risk and Progress Management" (CSU Global, n.d.), modify.

The Low/Medium/Critical rating can be defined as follows:

A Low risk is a risk that needs monitoring; it may go on a watchlist.

- Occasional (M) x Negligible (L) → Low
- Implausible (L) x Negligible (M) → Low
- Implausible (L) x Moderate (M)→ Low

A Medium risk is a risk that needs a mitigation plan and periodic tracking.

- Plausible (H) x Negligible (M) → Medium
- Occasional (M) x Moderate (M) → Medium

- Implausible (L) x Critical (H) → Medium

A High risk is a risk that needs immediate attention.

- Plausible (H) x Critical (H) → High
- Plausible (H) x Moderate (M) → High
- Occasional (M) x Critical (H) → High

Table 8

The Risk Matrix

Probability \ Severity	**Negligible (L)** *<4 weeks, <$50K, minor impact on key functions*	**Moderate (M)** *4–12 weeks, $50K–$250K, significant impact on key functions*	**Critical (H)** *>3 months, >$250K, critical impact on key functions*
Plausible (H) *>70% (very likely)*	--- (Medium)	R5, R8 (High)	--- (High)
Occasional (M) *30%–69% (possible)*	--- (Low)	R2, R3, R4, R6, R7, R9, R10, R11 (Medium)	R1 (High)
Implausible (L) *<30% (unlikely)*	--- (Low)	--- (Low)	--- (Medium)

Note: The table illustrates the prioritization of Project Codebase risks by mapping them into the matrix based on their assigned probability and impact ratings

Risk Matrix Synopsis

The following matrix synopsis justifies the overall rating of each scenario by explaining why the risk can occur for each scenario, by summarizing the potential impact if the risk materializes, in terms of time, cost, and quality.

High-Risk Scenarios

R1 - Data leakage and security breach (M x H).

This scenario risk is occasionally plausible, as one of the core functionalities of Project Codebase involves sending code snippets and sensitive documentation to external AI model providers. This functionality could trigger an incident that can have a high impact with catastrophic consequences on the project deployment, resulting in rework of the data flows and security processes. If implemented, the rework would cause the three-month timeline overrun, and costs could exceed $250K because of it. Therefore, this scenario has an overall risk rating of high, requiring immediate termination of the activity and immediate remedy if it occurs.

R5 - RAG/agent quality and trust issues (H x M).

This scenario is highly plausible as LLM models can generate incorrect outputs referred to as AI hallucinations. Project Codebase's main goal is to help engineers understand a software architecture and query codebases; however, if the AI queries provide misrepresented relationships or hallucinated responses, users will likely lose trust in the system. This impact is moderate-to-serious and usually requires a 4 to12 week period to fix, and adds costs for validation and testing processes. Therefore, this scenario has an overall risk rating of high, needing monitoring and immediate attention if it occurs.

R8 - Integration phase exceeds the 8-week integration window (H x M).

This risk is highly plausible as the integration requires the implementation of dependencies such as CI/CD, security modules, legal sign-off, end-to-end testing, and go/no-go readiness that may cause significant delay and add hidden costs. If this occurs, the impact is moderate as implementing solutions for integrations will add cost and exceed the integration phase allocated time, and delay the release by several weeks. Therefore, this scenario has an overall risk rating of high and needs monitoring and immediate attention if it occurs.

Medium-Risk Scenarios

R2 - Legal/IP noncompliance (M x M).

This scenario is occasionally plausible, as exterior AI coding agent providers and local AI agent automated processing of code/documentation can create licensing, confidentiality, and intellectual property issues needing legal and regulatory reviews. This functionality could trigger an incident that can have a moderate impact on the release of the application, that is, until contracts with the AI coding agent providers and legal and regulatory compliance are implemented. Additionally, the needed rework may include restricting what content is processed, adding user guidance, or switching AI providers, resulting in a 4–12 week slippage over the set timeline. Therefore, this scenario has an overall risk rating of medium, needing monitoring and immediate attention if it occurs.

R3 - Network latency and reliability issues (M x M).

This scenario is occasionally plausible, as Project Codebase depends on multiple services such as the VS Code extension, backend services, Neo4j queries, and external API calls. These services' functionality could trigger network traffic and latency issues. Moreover, if the application becomes slow or unreliable due to these issues, Omega.py software engineers may avoid using it. This impact is moderate as performance issues can require network and resource upgrades or architectural changes, delaying the project release. Additionally, the project cost may also rise due to the upgrades and rework needed. Therefore, this scenario has an overall risk rating of medium, needing monitoring and immediate attention if it occurs.

R4 - Neo4j schema/ingestion complexity causing rework (M x M).

This scenario is occasionally plausible, as knowledge graphs are complex data structures. Software architecture modeling includes identifying components and their relationships;

therefore, identifying and performing ingestion of these elements as nodes and relationships within a knowledge graph is a complex process. This functionality could trigger an incident where the knowledge graph schema queries or injections are incorrect. If this occurs, the impact of it is moderate as the Omega.py internal AI agent needs to be adjusted/better trained/better prompted. Additionally, this can cause 4 to 12 week delays and may add engineering and infrastructure costs if the solution requires additional resources. Therefore, this scenario has an overall risk rating of medium, needing monitoring and immediate attention if it occurs.

R6 - VS Code extension integration delays and adoption issues (M x M**).**

This scenario is occasionally plausible, as the delivery of the application is constrained to only VS Code IDE environment. This functionality could trigger authentication conflicts and integration issues with Project Codebase backend services. This impact is moderate as it may create delays from debugging VS Code environment issues, login/authentication processes, or reworking UI/UX. Additionally, this may to the cost of the project due to the needed rework. Therefore, this scenario has an overall risk rating of medium, needing monitoring and immediate attention if it occurs.

R7 - Cost overrun from external AI token usage (M x M).

This scenario is occasionally plausible, as the price of (external AI coding agent) tokens is based on usage and can change based on the size of the prompts and conversation context size. This functionality could trigger an incident that increases cost significantly, as IDE AI agentic coding workflow usually requires numerous frequent API calls to the providers. This impact is moderate as the team may need to enforce quotas, restrict context/prompt size, cache results, or use cheaper AI models. Therefore, this scenario has an overall risk rating of medium, needing monitoring and immediate attention if it occurs.

R9 - Stakeholder resistance blocking approvals or demanding rework (M x M).

This scenario is occasionally plausible, as some of the stakeholders have expressed resistance (security, network, and data roles). Stakeholders' resistance can take the form of delayed reviews, added acceptance criteria, or requests for additional information that can slow down design decisions. This impact is moderate; these stakeholders' resistance can result in schedule delays with a range of 4 to 12 weeks. Additionally, they can also increase the project cost due to rework. Therefore, this scenario has an overall risk rating of medium, needing monitoring and immediate attention if it occurs.

R10 - Scope creep destabilizes the baseline (M x M).

This scenario is occasionally plausible, as Project Codebase is a novel platform and stakeholders may request additional features as the project progresses (for example, a new IDE or browser support). The impact of this scenario is moderate as the scope baseline is destabilized by scope creep. Therefore, this scenario has an overall risk rating of medium, needing monitoring and immediate attention if it occurs.

R11 - Skills gap and ramp-up delays (M x M).

This scenario is occasionally plausible, as the project needs specialized skill areas such as RAG/agent engineering, Neo4j graph modeling, ingestion tooling, security/compliance controls, and VS Code extension development. Even with training, teams need time to share knowledge and implement new acquired skills. This impact is moderate as skill gaps can emerge, unknown issues appear, affecting the quality, the cost, and the delivery of the project. Therefore, this scenario has an overall risk rating of medium, needing monitoring and immediate attention if it occurs.

Sensitivity Analysis

Sensitivity analysis of Project Codebase "helps determine the individual project risks or other sources of uncertainty that have the greatest potential impact on project objectives. It correlates variations in project outcomes to variations in elements of the quantitative risk analysis model" (Ucertify, n.d.e, p.115). The sensitivity analysis uses the Expected Monetary Value (EMV) metric to evaluate the financial exposure of the project by calculating the average cost of each risk based on its probability of occurrence. The EMV formula used in this report is:

$$EMV = Probability \times Cost$$

Where:

- EMV is the Expected Monetary Value
- $Probability$ is the ranges used (defined earlier in the Probability Definitions section): L: 0% to 29%, M: 30% to 69%, H: 70% to 90.
- $Cost$ is the range costs defined in Table 5: L: $0K to $49K, M: $50K to $250K, H: $251K to $400K.

This formula can be used to compute each risk (using Prob/Impact ratings from Table 2)

- $EMV\ Low = ProbLow \times CostLow$
- $EMV\ High = ProbHigh \times CostHigh$
- $EMV\ Mid = ProbMid \times CostMid\ (midpoints\ are\ averages)$

The analysis also uses a tornado diagram, see Figure 5, to illustrate how risks rank based on their capacity to influence the project's overall financial baseline, visually ordering which risk can create the largest potential variable impact in the project budget. For Project Codebase, the tornado diagram illustrates expected cost exposure (EMV) per risk.

Table 9

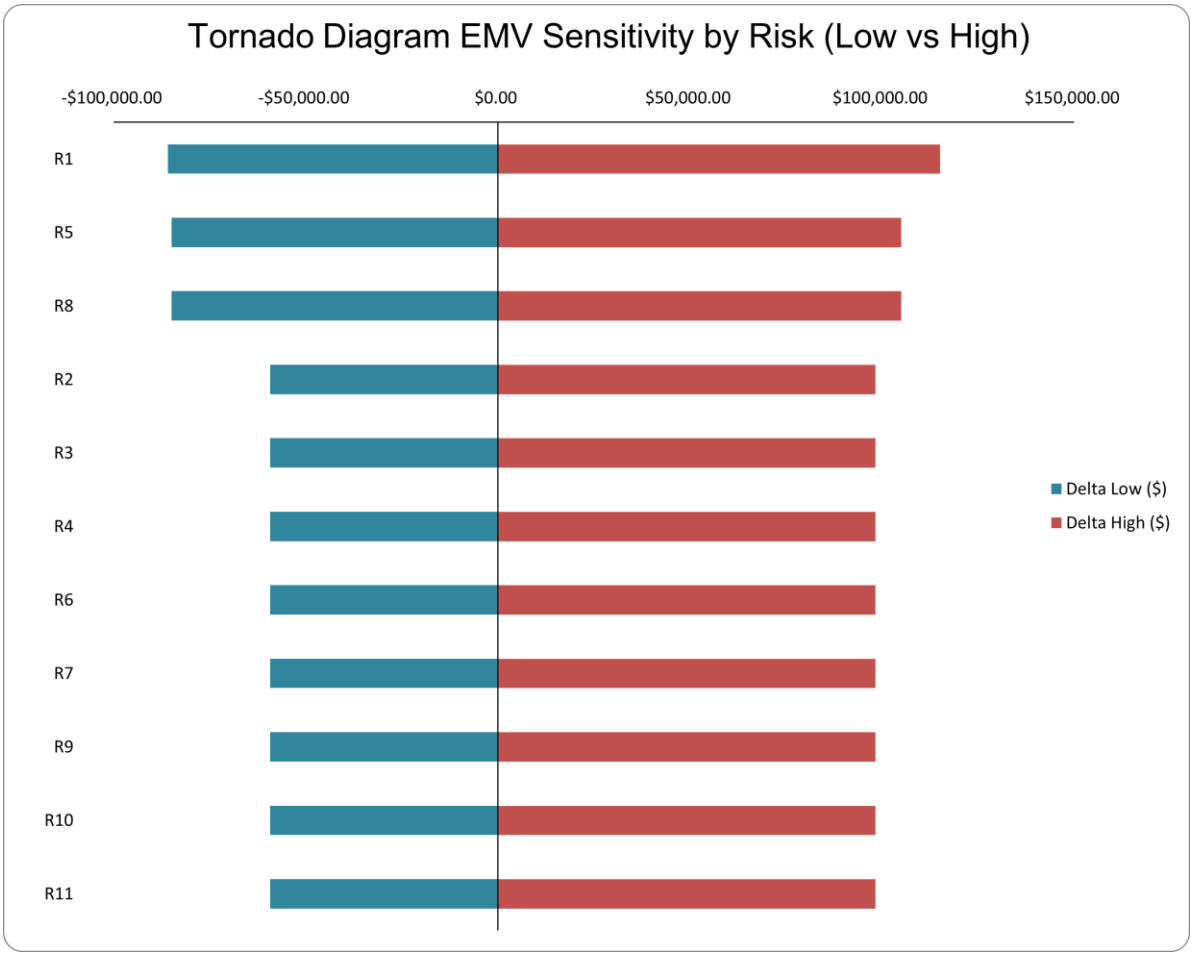
Sensitivity Tornado Diagram Cost Exposure EMV

ID	Risk Scenario	Prob Level	Impact Level	Prob Low	Prob High	Prob Mid	Cost Low ($)	Cost High ($)	Cost Mid ($)	EMV Low ($)	EMV Mid ($)	EMV High ($)	Delta Low ($)	Delta High ($)	Range Width ($)
R1	Data leakage and security breach	M	H	30.0%	69.0%	49.5%	$250,000.00	$400,000.00	$325,000.00	$75,000.00	$160,875.00	$276,000.00	-$85,875.00	$115,125.00	$201,000.00
R2	Legal/IP noncompliance	M	M	30.0%	69.0%	49.5%	$50,000.00	$250,000.00	$150,000.00	$15,000.00	$74,250.00	$172,500.00	-$59,250.00	$98,250.00	$157,500.00
R3	Network latency and reliability issues	M	M	30.0%	69.0%	49.5%	$50,000.00	$250,000.00	$150,000.00	$15,000.00	$74,250.00	$172,500.00	-$59,250.00	$98,250.00	$157,500.00
R4	Neo4j schema/ingestion complexity is causing rework	M	M	30.0%	69.0%	49.5%	$50,000.00	$250,000.00	$150,000.00	$15,000.00	$74,250.00	$172,500.00	-$59,250.00	$98,250.00	$157,500.00
R5	RAG/agent quality and trust issues	H	M	70.0%	90.0%	80.0%	$50,000.00	$250,000.00	$150,000.00	$35,000.00	$120,000.00	$225,000.00	-$85,000.00	$105,000.00	$190,000.00
R6	VS Code extension integration delays & adoption issues	M	M	30.0%	69.0%	49.5%	$50,000.00	$250,000.00	$150,000.00	$15,000.00	$74,250.00	$172,500.00	-$59,250.00	$98,250.00	$157,500.00
R7	Cost overrun from external AI token usage	M	M	30.0%	69.0%	49.5%	$50,000.00	$250,000.00	$150,000.00	$15,000.00	$74,250.00	$172,500.00	-$59,250.00	$98,250.00	$157,500.00
R8	Integration phase exceeds the 8-week integration window	H	M	70.0%	90.0%	80.0%	$50,000.00	$250,000.00	$150,000.00	$35,000.00	$120,000.00	$225,000.00	-$85,000.00	$105,000.00	$190,000.00
R9	Stakeholder resistance blocking approvals or demanding rework	M	M	30.0%	69.0%	49.5%	$50,000.00	$250,000.00	$150,000.00	$15,000.00	$74,250.00	$172,500.00	-$59,250.00	$98,250.00	$157,500.00
R10	Scope creep destabilizes the baseline	M	M	30.0%	69.0%	49.5%	$50,000.00	$250,000.00	$150,000.00	$15,000.00	$74,250.00	$172,500.00	-$59,250.00	$98,250.00	$157,500.00
R11	Skills gap and ramp-up delays	M	M	30.0%	69.0%	49.5%	$50,000.00	$250,000.00	$150,000.00	$15,000.00	$74,250.00	$172,500.00	-$59,250.00	$98,250.00	$157,500.00

Note: The table summarizes each risk data by listing the probability and impact cost ranges (Low, Mid, High), and the computed EMV based on those ranges.

Figure 5

Tornado Diagram Sensitivity by Risk



Note: The diagram illustrates the sensitivity of each risk, ranking them from top to bottom based on their potential financial variance (Range Width), negative to positive.

Note that the tornado 'wings' are calculated using the following formulas:

- $Delta\ Low\ =\ EMVLow\ -\ EMVMid\ (left\ side;\ negative)$
- $Delta\ High\ =\ EMVHigh\ -\ EMVMid\ (right\ side;\ positive)$

Additionally, the diagram shows that Risk R1 (Data Leakage and Security Breach) has the highest sensitivity, meaning that it has the largest potential financial swing with a range width of $201,000. justifying the allocation of extra contingency funds to minimize the potential impact from these risks.

Conclusion

Project Codebase Project Management Plan is designed to navigate the complexities of developing and implementing an intelligent AI agent and RAG system within the Omega.py software development ecosystem by planning the technical requirements, such as a knowledge graph Neo4j database, an integrated local AI agent, and DE-ML. Additionally, the project accomplishes this by upholding the Omegap.py System for Value Delivery goal, whose main intent is to develop a project that transforms “employee workflow efficiency” into financial savings. The PMP has selected a Hybrid Methodology that balances an adaptive and agile approach required to develop and implement a novel system and the uncertainty associated with AI Development, with a predictive approach that brings stability and rigidity, which are necessary to comply with legal and security constraints. This hybrid approach helps to mitigate High-Risk scenarios, such as RAG/agent quality issues and Integration phase delays, by using iterative testing sprints while maintaining a rigid timeline for compliance audits.

Furthermore, the plan addresses the human element of the project management process using a proactive Stakeholder Engagement Strategy that identifies early Resistant stakeholders in Security and Network Administration. The plan also incorporates specific mitigation strategies, such as Dual Accountability in the RACI matrix and funded training in the budget, to transition these stakeholders to a Supportive level. Additionally, the Sensitivity Analysis shows the plan is financially robust by identifying Data Leakage and Security Breach as the highest sensitivity risks, justifying the allocation of contingency funds. Finally, this PMP demonstrates that Project Codebase is technically viable and operationally secure, as the plan secures a budget that includes a Contingency Reserve for volatile costs and establishes a clear Scope Baseline to prevent scope creep and gold plating.

References

CSU Global (n.d.). Module 6: Risk and Progress Management, Canvas. https://portal.csuglobal.edu

PMI. (2021). The Standard for Project Management and a Guide to the Project Management Body of Knowledge (PMBOK guide) (7th ed.). Project Management Institute (PMI).

Ucertify (n.d.a). Lesson 3: Development Approach and Life Cycle Performance. Project Manager Professional (PMP) Based on PMBOK7. Ucertify. ISBN: 978-1-64459-415-5

Ucertify (n.d.b). Lesson 5: Stakeholder Engagement. Project Manager Professional (PMP) Based on PMBOK7. Ucertify. ISBN: 978-1-64459-415-5

Ucertify (n.d.c). Lesson 6: Project Scope. Project Manager Professional (PMP) Based on PMBOK7. Ucertify. ISBN: 978-1-64459-415-5

Ucertify (n.d.d). Lesson 7: Project Schedule: Project Scope. Project Manager Professional (PMP) Based on PMBOK7. Ucertify. ISBN: 978-1-64459-415-5

Ucertify (n.d.e). Lesson 13: Uncertainty. Project Manager Professional (PMP) Based on PMBOK7. Ucertify. ISBN: 978-1-64459-415-5