Agentic Project Management

Manage complex projects with a team of AI Assistants, smoothly and efficiently.

User Guide for APM v0.4 - August 2025

Documentation Suite

CobuterMan

Contents

1 About this Guide 2
1.1 Who is this guide for . . . 2

2 What is APM? 2
2.1 APM's Approach . . . 2
2.1.1 APM v0.4 Workflow Diagram . . . 3

3 Working with each Agent Type 4
3.1 Setup Agent . . . 4
3.1.1 Working Effectively with the Setup Agent . . . 4
3.1.2 Model Selection for the Setup Agent . . . 4
3.2 Manager Agent . . . 5
3.2.1 Working Effectively with the Manager Agent . . . 5
3.2.2 Model Selection for the Manager Agent . . . 5
3.3 Implementation Agents . . . 6
3.3.1 Working Effectively with the Implementation Agents . . . 6
3.3.2 Model Selection for Implementation Agents . . . 6
3.4 Ad-Hoc Agents . . . 7
3.4.1 Delegation Scenarios . . . 7
3.4.2 Model Selection for Ad-Hoc Agents . . . 7

4 Key Components 8
4.1 Implementation Plan . . . 8
4.1.1 Structure and Elements . . . 8
4.1.2 Working with Implementation Plans . . . 8
4.2 Memory System . . . 9
4.2.1 Memory System Strategies . . . 9
4.2.2 Memory Logs . . . 9

4.2.3 Phase Summaries 9
4.3 Handover Artifacts 10
4.3.1 Implementation Agent Handover Artifacts: 10
4.3.2 Manager Agent Handover Artifacts: 10
4.4 Meta-prompts 11
4.4.1 Types of Meta-prompts 11

5 Setting up your working environment **12**
5.1 Setting up APM Access 12
5.2 Adding APM Assets to Your Working Environment 13

6 Setup Phase **14**
6.1 Context Synthesis 14
6.1.1 Context Synthesis Phases 14
6.2 Project Breakdown 15
6.2.1 Reviewing the Initial Implementation Plan 15
6.2.2 How to Effectively Conduct Your Manual Review 16
6.3 Implementation Plan Review (Optional) 17
6.4 Enhancement & Memory Initialization 17
6.5 Manager Bootstrap Creation 17

7 Task Loop Phase **18**
7.1 Task Assignment 18
7.2 Task Execution 18
7.2.1 Task Execution Optimization 19
7.2.2 Memory Logging Optimization 20
7.3 Task Review 20

8 Handover Procedures **21**
8.1 Timing Strategies 21
8.1.1 Handover Timing Rules of Thumb 22
8.1.2 Performance Warning Signs 22
8.2 Verification Process 23

9 Optimization Strategies **24**
9.1 Model Selection Economics 24
9.2 Workflow Efficiency 24
9.2.1 Setup Phase Optimization 24
9.2.2 Task Execution Optimization 24

9.2.3 Memory Logging Optimization . . . 24
9.2.4 Proactive Handover Planning . . . 25
9.3 Optimizing APM Assets . . . 25

10 Troubleshooting **26**
10.1 Setup Phase Issues . . . 26
10.1.1 Omitted Requirements During Context Synthesis . . . 26
10.1.2 Interrupted Project Breakdown . . . 27
10.1.3 Interrupted Implementation Plan Enhancement . . . 27
10.2 Task Loop Issues . . . 28
10.2.1 Missing Context Integration Instructions . . . 28
10.2.2 Implementation Plan Needs Revisions . . . 28
10.3 Handover Problems . . . 30
10.3.1 Handling a Handover After the Context Window is Full . . . 30
10.3.2 Recovering from a Late Handover . . . 31

11 Summary **32**
11.1 Key Takeaways . . . 32
11.2 Optimizing Your APM Workflow . . . 32

1 About this Guide

This User Guide is designed as a comprehensive reference that provides in-depth coverage of all APM components, workflow patterns, troubleshooting, and optimization strategies to help you maximize your sessions. It serves as a complement to the main documentation suite.

Before using this guide, ensure you have **already read the *Quick Start Guide***, are familiar with the APM documentation, and have some hands-on experience with APM sessions.

The document is part of a two-tier guide set that includes:

- **User Guide** (this document)
- **Quick Start Guide**: For beginners and new users getting started with their first APM session

1.1 Who is this guide for

This guide is intended for users who are already familiar with APM's fundamentals, have practical experience, and wish to enhance their sessions, explore optimization strategies, or address common issues.

Reference Tip: Each section of this guide is self-contained, allowing you to consult any part independently.

Use the table of contents to quickly look up specific aspects of the framework as needed during your work.

2 What is APM?

Agentic Project Management (APM) is a structured multi-agent workflow framework enabling you to manage complex projects with AI assistants. APM distributes work across multiple agent instances, overcoming the context limitations that can arise in extended AI interactions.

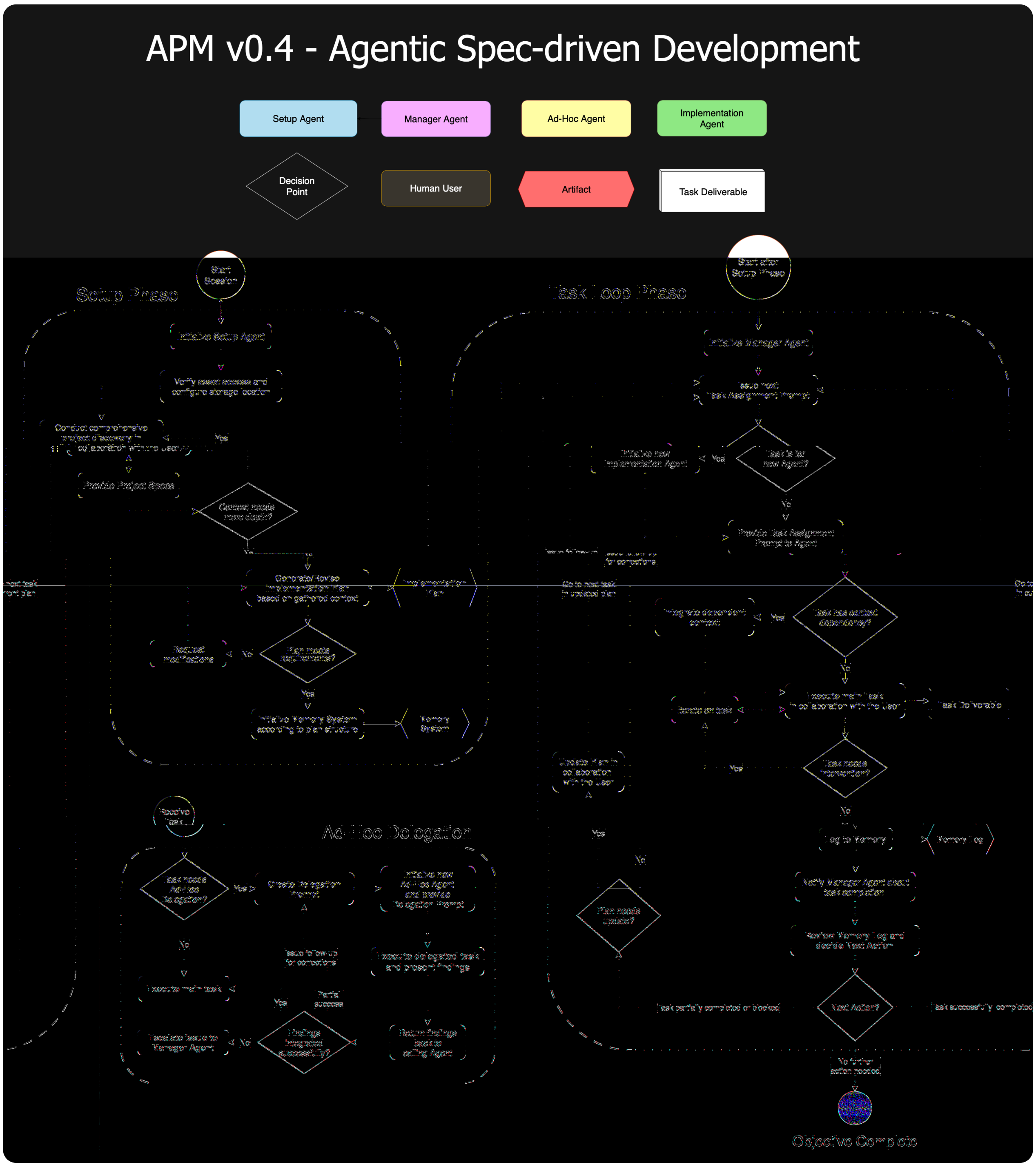
2.1 APM's Approach

Unlike traditional approaches, APM leverages **Dynamic Memory Management** for keeping project continuity across long-running sessions. Additionally, the framework uses **Meta-Prompts** for cross-agent coordination and **Handover Procedures** to ensure smooth context transfer when context window limits approach.

Note on Screenshots: The screenshots in this guide are taken from APM sessions using Cursor. However, most modern AI IDEs offer similar interfaces and configurations, so you can easily follow these patterns in your preferred environment.

2.1.1 APM v0.4 Workflow Diagram

To provide a clear overview of the APM v0.4 workflows, the following diagram illustrates the complete session lifecycle, from Setup through the Task Loop, highlighting all agent types, user actions, decision points, and key artifacts.



APM v0.4 - Agentic Spec-driven Development
Setup Agent
Manager Agent
Ad-Hoc Agent
Implementation Agent
Decision Point
Human User
Artifact
Task Deliverable
Setup Phase
Start Session
Initialize Setup Agent
Verify asset access and configure storage location
Conduct comprehensive project discovery in collaboration with the User
Yes
Provide Project Specs
Context needs more depth?
No
Generate/Revise Implementation Plan based on gathered context
Implementation Plan
Request modifications
No
Plan meets requirements?
Yes
Initialize Memory System according to plan structure
Memory System
Task Loop Phase
Start after Setup Phase
Initialize Manager Agent
Issue next Task Assignment Prompt
Initialize new Implementation Agent
Yes
Task is for new Agent?
No
Provide Task Assignment Prompt to Agent
for corrections
Go to next task in updated plan
Integrate dependent context
Yes
Task has context dependency?
No
Execute main Task in collaboration with the User
Task Deliverable
Iterate on task
Yes
Task needs intervention?
No
Update Plan in collaboration with the User
Yes
No
Plan needs update?
Log to Memory
Memory Log
Notify Manager Agent about task completion
Review Memory Log and decide Next Action
Task partially completed or blocked
Next Action?
Task successfully completed
No further action needed
Objective Complete
Receive Task
Ad-Hoc Delegation
Task needs Ad-Hoc Delegation?
Yes
Create Delegation Prompt
Initialize new Ad-Hoc Agent and provide Delegation Prompt
No
Issue follow-up for corrections
Execute delegated task and present findings
Execute main task
Yes
Partial success
Escalate issue to Manager Agent
No
Findings integrated successfully?
Return findings back to calling Agent

Figure 1: APM v0.4 workflow illustrating Agentic Spec-driven Development.

Note: Some steps of both the Setup Phase and the Task Loop Phase have been omitted in this diagram for brevity, while keeping the core workflow and agent interactions intact.

3 Working with each Agent Type

Understanding how to effectively work with each agent type is crucial for maximizing APM's efficiency. This section provides tips and strategies for working with each agent type.

3.1 Setup Agent

The Setup Agent initializes your APM session and breaks down your project into manageable tasks. Be communicative and detailed in your responses with this agent.

3.1.1 Working Effectively with the Setup Agent

The Setup Agent's effectiveness directly impacts your entire project. Consider these strategies when working with the Setup Agent:

- **Prepare Materials**: Have comprehensive PRDs, project requirements, and specifications ready before entering the Setup Phase.
- **Be Comprehensive**: Provide detailed responses during Context Synthesis, describing your project's needs in detail.
- **Review Thoroughly**: After Project Breakdown, carefully comb through the Implementation Plan before approval; ensure it fully aligns with your expectations and requirements, both in terms of task content and overall workflow structure; ask for clarifications and modifications if needed.
- **Ask Questions**: Don't hesitate to ask questions about uncertain or unclear aspects of your project; LLMs are not omniscient, and may make assumptions about your projects that do not align with your expectations. Make sure that you enter the Task Loop Phase with a claer understanding of your Implementation Plan.

Spec-driven Method: Take time to communicate your project requirements clearly to the Setup Agent. The quality of your input directly affects the usefulness of the Implementation Plan. Thorough, detailed responses and clarifications here will save you time and prevent costly issues later in the workflow.

3.1.2 Model Selection for the Setup Agent

The Setup Agent needs strong reasoning to break down your project into actionable tasks. For best results, use premium non-thinking agentic models. Recommended models:

- **Claude Sonnet 4 non-thinking is the best choice overall**, with excellent reasoning and agentic capabilities. Gemini 2.5 Pro is an acceptable alternative, but not as good.
- **Claude Sonnet 3.7** offers a budget-friendly option, but lacks agentic capabilities.

Model Switching Warning: Avoid switching models during Setup Phase. Context gaps from token caching disruptions can compromise project breakdown quality. Use one model throughout the entire Setup Phase.

3.2 Manager Agent

The Manager Agent is your central coordinator throughout the APM session, overseeing task assignment and making project decisions. The Manager also maintains the Memory System as the session progresses, playing a crucial role in context continuity.

3.2.1 Working Effectively with the Manager Agent

The Manager Agent handles project coordination, and you are the one who makes the final decisions. Guide the Manager Agent to make the right decisions by providing clear and concise instructions. Effective usage involves:

- **Clear Communication**: If a task uncovers key insights or issues, share this context with the Manager Agent while requesting a review of the relevant Memory Log to ensure nothing important is missed. For example:
```
"Task 3.1 has been completed, but I have identified a significant issue with
the database schema. Review the Memory Log, as well as the file related to
the database schema issue [relative file name here]. Should we update the
Implementation Plan or re-issue Task 2.5, which is responsible for creating the
database schema?"
```

- **Decision Making Support**: The success of your project relies on your direction for all key decisions. Provide clear and concise instructions to ensure the Manager Agent acts in line with your project goals. For example:
```
"I think we should re-issue Task 2.5 as a multi-step task. The first step
should instruct the Agent to read the Backend PRD to fully understand the
database requirements. Since this will impact Task 3.1, that task should also
be re-issued to reflect any changes resulting from the updated Task 2.5. Please
provide the updated Task Assignment Prompt for Task 2.5."
```

- **Ad-Hoc Delegations**: For context-intensive tasks (such as research, analysis or technical investigation), can be delegated directly to Ad-Hoc Agents. This is useful when additional information is needed to update the Implementation Plan or guide project direction.

3.2.2 Model Selection for the Manager Agent

The Manager Agent needs to be able to reason about project decisions and the tasks at hand. For best results, use premium non-thinking agentic models. Recommended models:

- **Claude Sonnet 4 non-thinking** is the best choice overall. It's agentic capabilities allow for all the routine actions of the Manager Agent during a task cycle to happen smoothly. Gemini 2.5 Pro and GPT 5 are good alternatives, but not as good.
- **Claude Sonnet 3.7** offers a budget-friendly option, but lacks agentic capabilities.

Efficiency Tip: During testing (August 2025), **Cursor Auto delivered exceptional Manager Agent performance**, making it the most efficient choice for this agent instance.

Model Switching Warning: Although switching models for the Manager Agent did not cause major issues during testing, it is still best practice to use the same model consistently for each Manager Agent instance.

3.3 Implementation Agents

The Implementation Agents are your focused task executors. They handle only the tasks assigned to them, have a narrow scope and do not make project decisions.

3.3.1 Working Effectively with the Implementation Agents

To ensure efficient and effective task execution, consider the following strategies:

- **Double-check the Task Assignment Prompt**: Make sure that the Task Assignment Prompt matches your requirements. Sometimes, the Manager Agent may make assumptions about your project that do not align with your expectations. Make corrections as needed.
- **Utilize Multi-step Tasks:**
  - **Natural Checkpoints** If the task is a multi-step task, you have natural opportunities to iterate with the agent between steps. Request clarifications, changes, corrections, and refinements as needed in order to ensure the task is completed to your satisfaction.
  - **Step Combination** You can combine adjacent steps for efficiency, if none of them require User feedback or external actions. If you deem that some steps are closely related, request that the agent execute them in a single response. For example:

    ```
    "Step 2 looks fine. Combine steps 3-4 in your next response."
    ```

    Use step combination with caution: Combine steps only when you are certain it will not compromise task quality. Do not combine steps that require User feedback, external actions, or are complex enough to need separate attention.

- **Leverage Ad-Hoc Agents:** As of August 2025, APM includes two official Ad-Hoc delegation guides (Research and Debug), but **you are free to delegate subtasks to an Ad-Hoc Agent without a formal guide**. To do so, simply ask the Implementation Agent to delegate the task and provide the delegation prompt in a markdown code block. For example:

  ```
  "I need to iterate on the design of this component. Please delegate this task to
  an Ad-Hoc Agent, and present the delegation prompt in a markdown code block."
  ```

  Note on informal delegations: Without a formal guide, Implementation Agents will improvise delegation prompts. Ad-Hoc Agents can usually handle these tasks, but **always review and add any needed context or clarifications to the delegation prompt**.

3.3.2 Model Selection for Implementation Agents

While premium models will deliver best task execution quality, APM's granular task breakdown enables the use of more affordable models for routine tasks.

- **Premium Models**: Claude Sonnet 4, Gemini 2.5 Pro, or GPT-5 all deliver great results.
- **Base Models**: GPT-4.1 in Copilot, Cursor Auto, or Windsurf SWE-1 are budget alternatives.

Model Switching Strategy: Implementation Agents handle model switching well without major context loss. However, **be cautious**; model switching may occasionally introduce subtle inconsistencies and context gaps.

3.4 Ad-Hoc Agents

Ad-Hoc Agents are temporary agent instances used to handle isolated, context-intensive tasks outside the main workflow. They operate in separate workflow branches and are concluded once their findings are returned to the calling Implementation Agent.

3.4.1 Delegation Scenarios

APM v0.4 currently covers two Ad-Hoc delegation scenarios: **Debugging** and **Research**. However, you are free to delegate any task to an Ad-Hoc Agent without a formal guide, as long as you provide a clear and concise delegation prompt.

Ask an Implementation Agent to delegate the task to an Ad-Hoc Agent, and provide the delegation prompt in a markdown code block. See example above.

Some examples of when you might delegate tasks to an Ad-Hoc Agent:

- **Solving a bug** that the Implementation Agent can't resolve.
- **Researching a new technology or tool** for the execution of the current task.
- **Performing data analysis** to inform a decision.
- **Reviewing or summarizing lengthy documentation** relevant to the current task.

Understanding Ad-Hoc Agents: The collaboration between Implementation Agents and Ad-Hoc Agents closely mirrors the relationship between Manager Agents and Implementation Agents. Here, the Implementation Agent prepares a structured delegation prompt (using the appropriate delegation guide or improvising one), and the Ad-Hoc Agent carries out the assigned task. The key distinction is that **Ad-Hoc Agents do not log their work to the Memory System.** Instead, they report back to the calling agent via the User.

Delegating from the Manager Agent: While Ad-Hoc Agents are typically called by Implementation Agents, you may also delegate tasks to them directly from the Manager Agent when appropriate. This is especially useful when the Manager Agent requires additional information or analysis, such as research or technical investigation, **to make updates to the Implementation Plan or guide project direction.** In these cases, the Manager Agent can initiate an Ad-Hoc delegation and incorporate the results as needed.

3.4.2 Model Selection for Ad-Hoc Agents

Ad-Hoc agents are completing one task per session, except if there is a follow-up task to the same Ad-Hoc agent instance. **Match model capability to delegation complexity**:

- **Complex Debugging**: Premium models for systemic issues.
- **Simple Fixes**: Budget models for environment problems
- **Research Tasks**: Mid-tier models balance capability and cost
- **Follow-up Tasks**: Same or better model as the previous task; **do not switch models, except if its an upgrade**.

4 Key Components

Understanding APM's key components enables you to get a complete understanding of the framework. This section provides in-depth coverage of each component's structure, purpose, and usage patterns.

4.1 Implementation Plan

The Implementation Plan serves as your project's comprehensive blueprint, created through systematic project breakdown by the Setup Agent.

4.1.1 Structure and Elements

- **Phases:** Logical groupings of related work with clear progression milestones
- **Tasks:** Distinct units of work with detailed specifications and success criteria
- **Sub-tasks:** Granular execution steps providing clear implementation guidance
- **Dependencies:** Explicitly marked producer-consumer relationships for context integration
- **Agent Assignments:** Distribution of work across Implementation Agents

4.1.2 Working with Implementation Plans

Drafting the Plan: During Project Breakdown, the Setup Agent creates an Implementation Plan tailored to your project requirements, as they were understood during Context Synthesis. To minimize issues during the Task Loop Phase, **carefully review this draft and request any necessary changes**. While APM v0.4 provides an optional AI-driven Systematic Review, note that it focuses on workflow and AI-specific issues, not on overall quality or requirement accuracy.

Tailor the Implementation Plan to both your project's requirements and your workflow preferences: Consider factors such as your platform's billing model, the cost of the models you use or any specific development preferences. For example:

- If you are billed per request, you might opt for more single-step tasks or keep multi-step tasks concise to control costs.
- If version control is important to you or your team, you can require that every task includes a commit subtask, ensuring that you keep a clean commit history.

See How to Effectively Conduct Your Manual Review for more guidance.

Modifying the Implementation Plan Mid-Session: It is common to update the Implementation Plan as your project evolves. Typical scenarios include new requirements, changes in scope, or discoveries during execution.

- **Minor changes:** Ask the Manager Agent to update the plan with clear, detailed instructions.
- **Major changes:** Return to the Setup Agent for systematic revision; consider context window usage.
- **Review updates:** Always verify that dependencies and related tasks are correctly updated.

For a complete troubleshooting workflow, see this section.

4.2 Memory System

The Memory System provides comprehensive project history and enables effective handover procedures. The Setup Agent automatically selects the appropriate strategy based on project complexity.

4.2.1 Memory System Strategies

APM v0.4 offers two Memory System strategies:

- **Simple Memory Bank:** A single-file approach where Memory Logs are maintained as inline sections within the file. Ideal for small projects with only a few tasks, such as:
  - Reviewing a small PR
  - Adding a small feature
  - Refactoring some modules of a codebase
- **Dynamic Memory Bank:** A directory-based system mapping Memory Logs to tasks within the Implementation Plan. The default approach for most projects beyond simple use cases.

  This system is designed to be scalable and flexible, as it **progressively expands as the project grows.** When changes are made to the Implementation Plan, the Manager Agent will automatically adapt the Memory System to the new structure.

4.2.2 Memory Logs

Memory Logs serve as **an abstraction layer for task execution**, capturing only the essential information about each task's completion. The Manager Agent reviews these concise summaries to maintain an overview of project progress, while the detailed execution remains with the Implementation Agent assigned to the task. If additional context is needed, the User can always provide it. See memory logging optimization for more.

In addition, Memory Logs are **fundamental building blocks of the project's context**. Since each task is completed by a single agent, its Memory Log becomes a discrete unit of contextual knowledge. This structure is crucial for robust and consistent Implementation Agent Handover Procedures: when a handover occurs, the incoming Implementation Agent reads the previous agent's Memory Logs to accurately transfer documented context.

Memory Log Quality: While this abstraction layer is essential for efficiency, it can sometimes omit important details and create context gaps. It's good practice to review the Memory Log entries to ensure all relevant information is included, giving the Manager Agent a complete and accurate understanding of the task execution.

4.2.3 Phase Summaries

At the end of each phase, the Manager Agent creates a phase summary at the Memory Root. These summaries offer a concise, high-level overview of the project's progress and context, serving as **an additional abstraction layer**, much like Memory Logs, to help the Manager Agent track and manage the project effectively. Phase summaries are also essential during Manager Agent Handover Procedures, ensuring smooth context transfer and continuity.

4.3 Handover Artifacts

Handover artifacts enable seamless context transfer between agent instances when approaching context window limits. The two-artifact system APM provides, preserves both technical state and working relationship context of the previous agent instance.

4.3.1 Implementation Agent Handover Artifacts:

To ensure a reliable and complete context transfer during Implementation Agent handovers, the two handover artifacts work together as follows:

- **1. Follow the Handover Prompt's Instructions:**
  - **a. Read Memory Logs (Chronological Order):** The new Implementation Agent begins by reading the previous agent's Memory Logs in order. This establishes a factual, ground-truth record of task execution and project history.
  - **b. Review Key Output Files:** Next, the agent reviews the most important output files produced during the previous agent's tenure, confirming the technical state of the project and ensures all deliverables are accounted for.
- **2. Consult the Handover File:** Finally, the agent reads the Handover File, which contains the outgoing agent's context dump. This captures undocumented insights, such as:
  - User-agent working relationships
  - Workflow preferences
  - Decisions not reflected in Memory Logs or output files

4.3.2 Manager Agent Handover Artifacts:

The process for Manager Agent handovers is similar, using the same two-artifact system, but tailored to the manager's broader oversight responsibilities:

- **1. Follow the Handover Prompt's Instructions:**
  - **a. Read Phase Summaries (Memory Root):** The new Manager Agent starts by reading phase summaries in the Memory Root, providing a high-level overview of the current project state.
  - **b. (Optional) Review Key Memory Logs:** If additional detail is needed, the manager may briefly review important Memory Logs from previous phases, focusing on tasks with producer-consumer dependencies to ensure continuity.
- **3. Read the Handover File:** The Manager Agent then reads the Handover File, which contains a context dump similar to that used in Implementation Agent handovers.

Handover Artifact Quality and Validation: Before proceeding with a handover, **carefully review both the Handover File and the Handover Prompt for accuracy and completeness.**

The Handover File is particularly susceptible to hallucinations or missing context, especially if the handover occurs late. Although the Handover Prompt is more standardized, it too can contain errors introduced by hallucinations. See Handover Problems for troubleshooting.

4.4 Meta-prompts

Meta-prompts are structured prompts that follow a standardized format, which **agents dynamically fill with context-specific content** depending on the situation, agent role, and task at hand. While meta-prompts are often presented as in-chat code blocks for easy copy-paste and transfer, they can also be generated as standalone files (such as Memory Logs) when persistent or file-based communication is needed.

These prompts serve as the primary mechanism for cross-agent communication, shifting the responsibility of prompt engineering from the User to the agents themselves. The User's main role is to transfer these meta-prompts between chat sessions or agents, either by copy-pasting or requesting review of files. **This approach empowers the User to intervene at any point**, ensuring that the prompts accurately reflect their intentions and preferences, while allowing agents to handle the technical details of prompt construction.

4.4.1 Types of Meta-prompts

- **Task Assignment Prompts:** Created by Manager Agents for Implementation Agents. Presented as **in-chat code blocks**.
- **Bootstrap Prompts:** Generated by Setup Agent for the first Manager Agent. Presented as **in-chat code blocks**.
- **Handover Artifacts:** Created for Handover Procedures by Implementation and Manager Agents. Handover Prompts are presented as **in-chat code blocks**, while Handover Files are presented as **standalone files**.
- **Delegation Prompts:** Created by Implementation Agents for Ad-Hoc Agents. Presented as **in-chat code blocks**.

Meta-prompt Iteration: Meta-prompts are either presented as in-chat code blocks or as standalone files. Since their contents are dynamically generated, the User can iterate on them as needed, requesting changes, modifications, or refinements.

For example, request a change to the Task Assignment Prompt from the Manager Agent:

```
"Task 3.1 has been completed successfully. Please review the log and continue
accordingly. When creating the next Task Assignment Prompt, be sure to include
clear explanation instructions for steps 2 through 5."
```

Another example, request an addition to the Memory Log from the Implementation Agent:

```
"Step 5 has been completed successfully. Please proceed to log to the
designated Memory Log. In your log entry, make sure to include that the User
wants to explore using a process manager (such as PM2 or nodemon) to improve the
development workflow and performance of the backend build process."
```

Modify Meta-prompts Cautiously: Iterating on meta-prompts is optional, and can be extremely valuable for making corrections or addressing mistakes. **However, when making additions or refinements to the prompts, each iteration should be approached thoughtfully, with careful review of the agent's actions.** Always verify that any changes remain consistent with the intended purpose and requirements of the prompt.

5 Setting up your working environment

5.1 Setting up APM Access

To access the APM framework assets, choose one of the following three options:

- **Option A: Clone the Upstream Repository (Default)**
  Clone the official APM repository directly from GitHub. This provides the latest, unmodified version of the APM assets.

  ```
  git clone https://github.com/sdi2200262/agentic-project-management
  ```

- **Option B: Use as Template (Recommended for Customization)**
  Use GitHub's "Use this template" feature to create your own copy of the APM repository. This allows you to customize prompts, guides, and workflows for your specific use case.

  ```
  git clone https://github.com/your-username/your-custom-apm-template
  ```

  Customization Tip: Using a template enables you to tailor APM prompts, guides, and workflows for your project's or organization's needs. See Modifying APM in the documentation for practical customization strategies and examples.

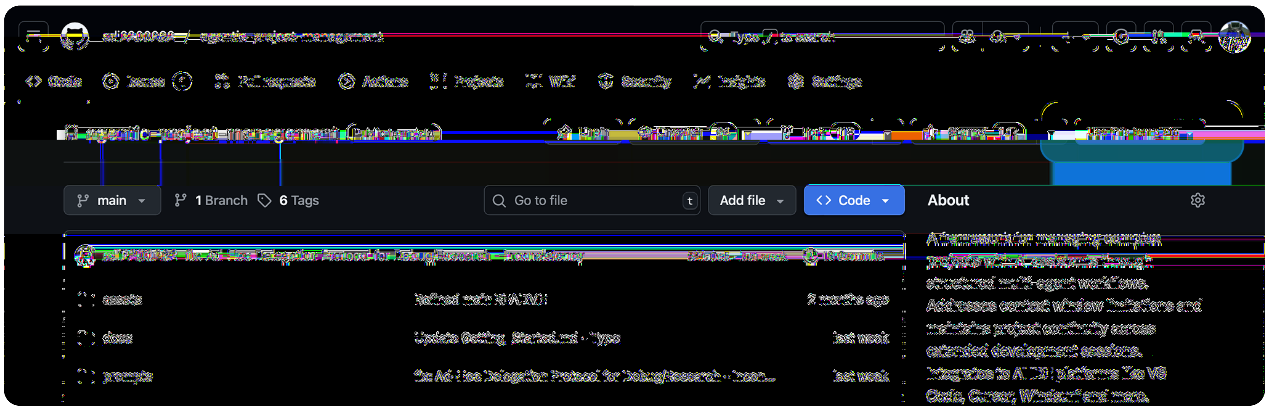


Figure 2: GitHub's "Use this template" feature - top right corner.

- **Option C: Manual Access**
  Manually copy and paste prompt and guide contents from GitHub or other sources as needed. This is not recommended, as it undermines agentic workflow efficiency.

  Manual Access Warning: If you choose this option, or any other option that does not give the agents direct access to the APM assets, you will need to describe your workflow approach in detail to the Setup Agent during the Asset Verification step.

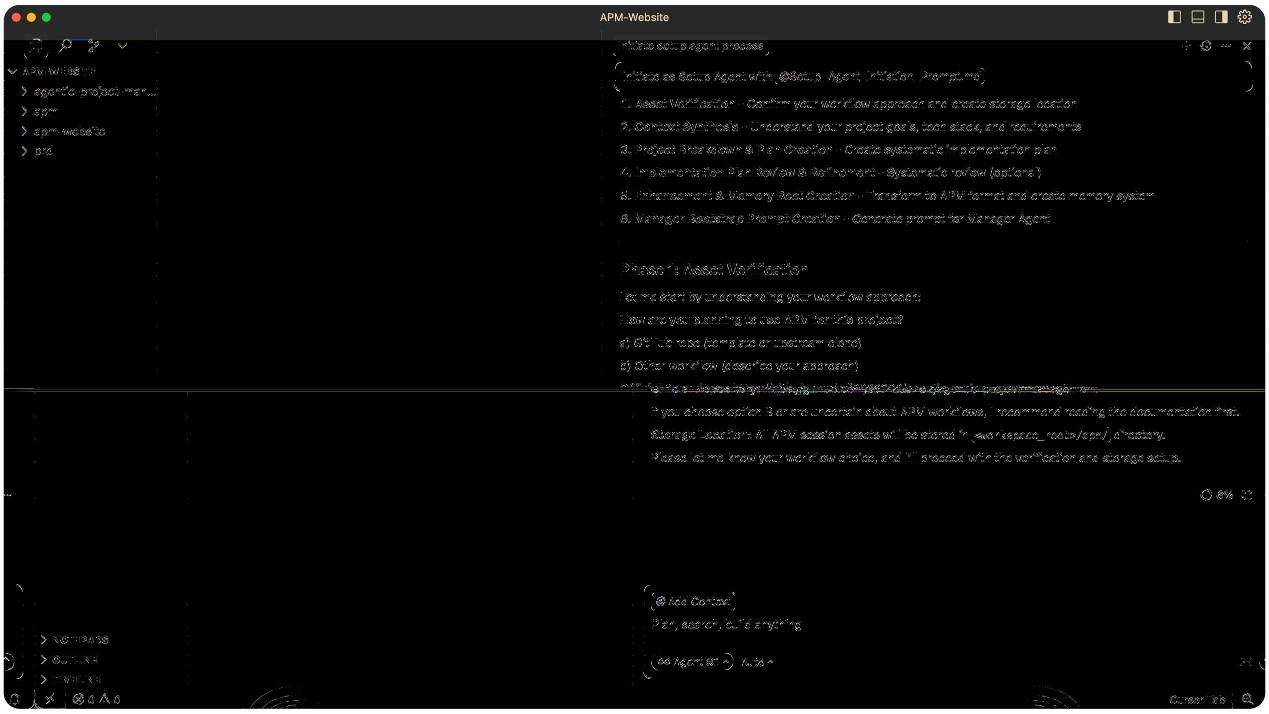
5.2 Adding APM Assets to Your Working Environment

Once you have chosen your preferred access method, you can integrate the APM assets into your working environment in several ways:

1. **Clone APM inside your project's directory**
   Clone the APM upstream or your custom template repository inside your project's directory. This creates an `apm/` directory at your project root for all APM assets.

   > **Git Implications:** You may wish to update your `.gitignore` to exclude the APM assets and `apm/` directory. If you prefer to track session progress, you can integrate commits in your task specifications.

2. **Set up a parent directory containing both your project and APM**
   Create a parent directory that will contain both your project's directory and the APM repo clone alongside it. **This avoids git implications within your project repository and keeps the `apm/` directory with session assets at the parent directory root.**



Initiate setup agent process

Initiate as Setup Agent with @Setup_Agent_Initiation_Prompt.md

1. Asset Verification - Confirm your workflow approach and create storage location
2. Context Synthesis - Understand your project goals, tech stack, and requirements
3. Project Breakdown & Plan Creation - Create systematic implementation plan
4. Implementation Plan Review & Refinement - Systematic review (optional)
5. Enhancement & Memory Root Creation - Transform to APM format and create memory system
6. Manager Bootstrap Prompt Creation - Generate prompt for Manager Agent

Phase 1: Asset Verification

Let me start by understanding your workflow approach:

How are you planning to use APM for this project?

a) GitHub repo (template or upstream clone)

b) Other workflow (describe your approach)

If you choose option B or are uncertain about APM workflows, I recommend reading the documentation first.

Storage Location: All APM session assets will be stored in `<workspace_root>/apm/` directory.

Please let me know your workflow choice, and I'll proceed with the verification and storage setup.

Figure 3: APM assets in a parent directory - bottom right corner.

3. **Clone APM in a shared location and add to your IDE workspace**
   Clone the APM directory in a location of your choice and add it to your IDE's workspace alongside your project. **This allows you to use the same APM assets across multiple projects by cloning it only once.** If you don't have project-specific assets, this is the 'tidiest' option.

   > **Codebase Indexing Implications:** This approach may have implications for codebase indexing in your IDE, as the `apm/` directory has no standard "root" to be stored in. If you use this method, be specific about where you want assets to be stored and how you intend to store the APM session assets within your workflow.

6 Setup Phase

The Setup Phase is crucial to project success. **A spec-driven approach to AI development is essential**, and this phase ensures all your requirements are fully captured and clearly understood by the session's agents.

For guidance on how to best prepare for a successful Setup Phase, refer to Working Effectively with the Setup Agent.

Preparation Impact: Well-prepared materials help the Setup Agent ask more informed questions and produce a higher-quality Implementation Plan.

6.1 Context Synthesis

During Context Synthesis, the Setup Agent leads a structured, adaptive Q&A process designed to capture all essential context for your project. This process unfolds across four targeted phases, with strategic follow-up questions to ensure a deep and accurate contextual understanding of your project's needs.

Be thorough and specific in your responses. This enables the Setup Agent to convert your requirements into an accurate Implementation Plan.

6.1.1 Context Synthesis Phases

Each phase of Context Synthesis targets a specific aspect of your project's requirements. **The Setup Agent uses internal indicators to adapt its project discovery strategy, systematically transforming your input into a comprehensive set of phases, tasks, and subtasks.**

- **Phase 1: Existing Materials and Vision** The Setup Agent gathers **a high-level understanding of your project's purpose and goals.** You will be asked to share any existing materials, documentation, or prototypes. This phase shapes the rest of Context Synthesis and informs the Setup Agent's strategy for project discovery.
- **Phase 2: Work Structure and Technical Requirements** The Setup Agent asks **targeted questions about your project's structure, dependencies, and technical constraints**, such as challenging components, technology stack, and resource limits. This ensures the Implementation Plan fully captures your project's complexity and technical parameters.
- **Phase 3: Process and Implementation Preferences** The Setup Agent collects **your preferences for workflow, quality standards, and implementation methods**. You may be asked about favored methodologies, review checkpoints, documentation, and/or to provide examples, templates or other reference materials. This ensures your development preferences and quality standards are clearly reflected in the task specifications of the plan.
- **Phase 4: Validation and Asset Format Selection** The Setup Agent **summarizes its understanding of your project for your review and feedback**, then asks you to select your preferred asset format (Markdown or JSON). At this stage, you can **request clarifications or modifications** to ensure the summary is accurate and aligned with your expectations.

Context Synthesis Tip: Timely input ensures the Setup Agent's project discovery works optimally, so provide all relevant information at the right phase and be prepared in advance. If you forget something, see Omitted Requirements During Context Synthesis.

6.2 Project Breakdown

Project Breakdown transforms all gathered context from Context Synthesis into a structured Implementation Plan using a carefully designed systematic analysis sequence.

This process, developed specifically for APM v0.4, is designed to **”force” Chain-of-Thought reasoning in widely available non-thinking models.** The Setup Agent conducts its analysis and reasoning within the chat, serving as the ”thinking” medium, while the Implementation Plan file serves as the structured ”output” medium.

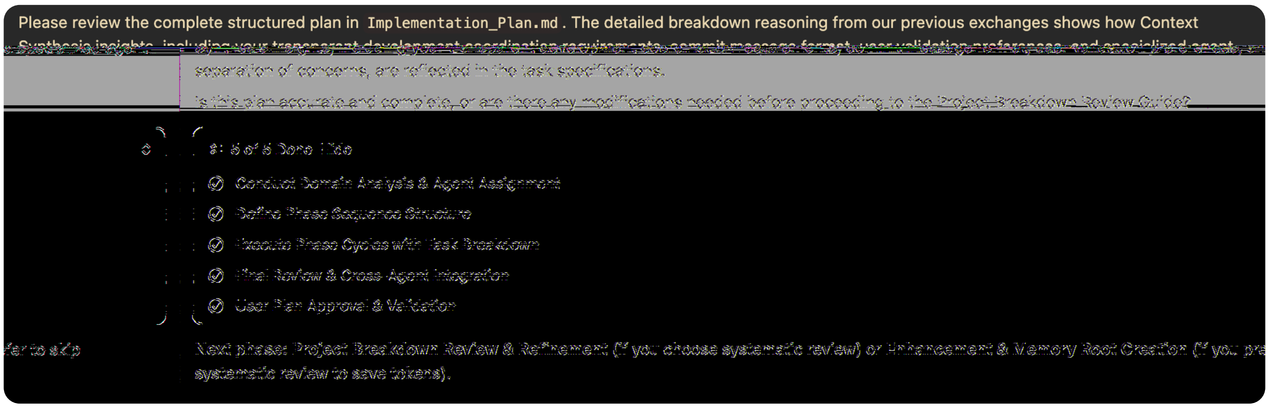
Forced CoT Implications: This process is intended to run as a single, uninterrupted response from the Setup Agent. However, interruptions may occur due to IDE system prompts or limitations of less-capable models. For best results, use non-thinking models with strong agentic capabilities, like **Claude Sonnet 4**, which can reliably switch between chat and file output. If the sequence is interrupted, see Interrupted Project Breakdown for recovery.

6.2.1 Reviewing the Initial Implementation Plan

A high-quality Implementation Plan depends on the thoroughness of the context gathered during Context Synthesis. However, even with comprehensive context, the Setup Agent can still make mistakes, miss important details, or misinterpret requirements. Since the Implementation Plan acts as the foundation for your entire APM session, **it is critical that you personally review it after Project Breakdown is complete.**

After Project Breakdown, the Setup Agent will prompt you to review the Implementation Plan and also ask if you wish to use an optional AI-driven Systematic Review to help identify certain issues. Regardless of whether you use the automated review, you must **always perform your own comprehensive manual review** to ensure the plan’s accuracy and completeness.

AI-Driven Review Warning: While APM v0.4 does offer an automated review conducted by the Setup Agent, **you should not rely solely on this option to ensure your plan’s accuracy.** The automated review conducted by the Setup Agent aims at AI-specific planning issues like task-packing, misclassified tasks, LLM errors, **not full project context.**


Please review the complete structured plan in `Implementation_Plan.md` . The detailed breakdown reasoning from our previous exchanges shows how Context Synthesis insights, including your transparent development coordination requirements, commit message format, user validation preferences, and specialized agent separation of concerns, are reflected in the task specifications.

Is this plan accurate and complete, or are there any modifications needed before proceeding to the Project Breakdown Review Guide?

s: 5 of 5 Done Hide

- Conduct Domain Analysis & Agent Assignment
- Define Phase Sequence Structure
- Execute Phase Cycles with Task Breakdown
- Final Review & Cross-Agent Integration
- User Plan Approval & Validation

fer to skip

Next phase: Project Breakdown Review & Refinement (if you choose systematic review) or Enhancement & Memory Root Creation (if you pre systematic review to save tokens).

Figure 4: After Project Breakdown, the Setup Agent presents a comprehensive summary and prompts you to review the plan or optionally use the AI-driven Systematic Review.

6.2.2 How to Effectively Conduct Your Manual Review

When reviewing the Implementation Plan, carefully read through each phase, task, and subtask. **The Project Breakdown process displays the agent's reasoning for all its decisions in the chat**, so be sure to review these explanations as well for additional context. As you examine the plan file itself, use the following checklist to guide your review:

- **Missing or Incomplete Requirements:** Check for any features, specifications, or contextual details that may have been overlooked or described too vaguely. Ensure that all your original requirements and expectations are represented, and that nothing important has been missed.
- **Task-Packing and Task Classification:** Look for tasks that may be combining unrelated actions, or tasks that are misclassified as multi-step when they should be single-step, or vice versa. Proper task granularity is crucial for efficient execution and clear agent instructions.
- **Logical or Workflow Errors:** Review the order of tasks and their dependencies. Make sure that the sequence of work makes sense, that all necessary prerequisites are accounted for, and that handoff points between agents or between you and the agent are clearly defined.
- **LLM-Specific Mistakes:** Be alert for errors that are common in AI-generated plans, such as hallucinated details, misunderstood instructions, or artificial similarities between tasks caused by pattern-matching rather than true understanding.
- **Opportunities for Optimization:** Look for tasks or subtasks that are redundant, meaning they could be combined with others to improve efficiency without sacrificing performance, unclear deliverables, or any areas where the plan could be streamlined for greater clarity or effectiveness.

Importance of Manual Review: Ensuring your Implementation Plan is carefully tailored to your project's unique needs helps prevent costly changes and revisions later in the session. **A thorough review early on, saves time, effort, and money down the line.**

Other Key Factors to Consider During Review

In addition to the general checklist above, there are some important factors that are especially relevant in AI-driven development workflows. As you review your plan, pay special attention to the following:

- **Billing Model Awareness:**
  - **If you are billed by token usage:** Aim to moderate total token consumption by removing non-essential tasks. Streamline instructions and avoid unnecessary complexity. **Obviously, this can also be applied for per-request billing models.**
  - **If you are billed by request:** Be mindful of the number of multi-step tasks and the number of steps within each, as each step may count as a separate request. To optimize costs and efficiency, consider **converting simple multi-step tasks into single-step tasks where possible** and **combining closely related steps within multi-step tasks to reduce the total number of requests.**
- **Workflow Preferences:** Ensure that your preferred development workflow is fully reflected in the task specifications. For example, if you require regular review checkpoints, automated build or lint checks, or other process requirements, confirm that these are explicitly included as tasks or subtasks.

6.3 Implementation Plan Review (Optional)

The Setup Phase includes this optional step, to help ensure the Implementation Plan is complete and accurate. If you opt for a systematic review, the Setup Agent highlights plan sections possibly needing extra attention for you to select. The user-selected sections are then analyzed and reviewed by the Setup Agent.

The agent-driven review targets AI-specific planning issues like task-packing, misclassified tasks, LLM errors, **not full project context.**

You should always conduct your own comprehensive review of the Implementation Plan instead of relying on the Agent's review.

Agent-Driven Review Tip: Choose to proceed with this agent-driven review if you have token budget to spare, or if you are a first-time APM user.

6.4 Enhancement & Memory Initialization

In this step, the Setup Agent transforms the reviewed Implementation Plan into a detailed APM artifact and initializes the Memory System accordingly.

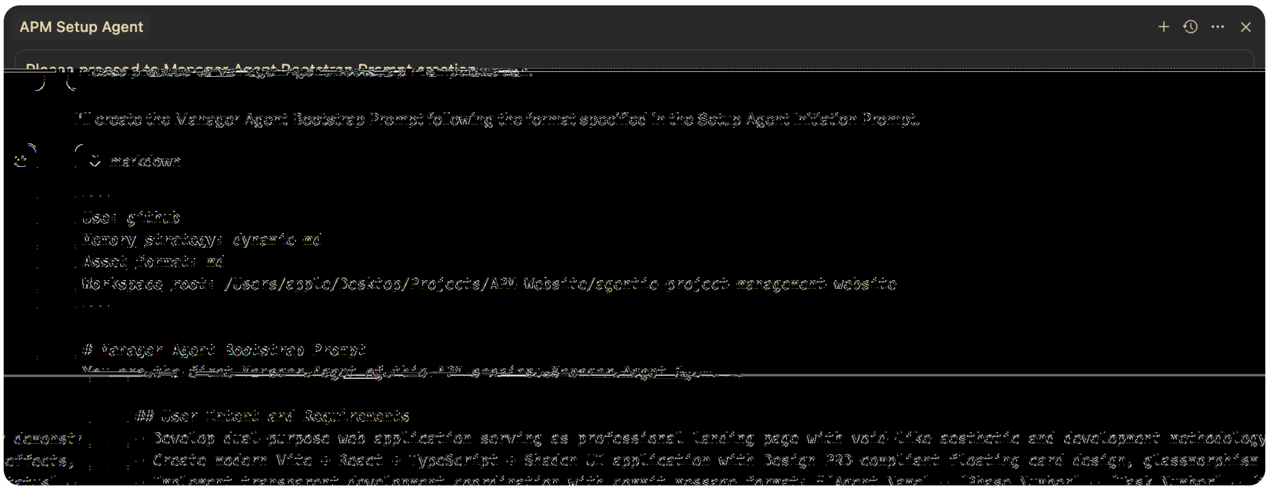
The transformation is completed in a single, structured response converting one phase at a time, while the Memory System is initialized based on the project complexity.

Implementation Plan Enhancement Warning: Like Project Breakdown, this sequence may be interrupted for similar reasons. See Troubleshooting section for guidance on recovery.

6.5 Manager Bootstrap Creation

After completing Implementation Plan Enhancement and Memory System initialization, the Setup Agent produces the session's first Meta-prompt: the Bootstrap Prompt. This prompt **contains all essential information required for the Manager Agent to initiate the Task Loop Phase.**

This Bootstrap Prompt is presented as a markdown code block for easy copy-paste. Store this prompt for use in the first Manager Agent initialization.



Please proceed to Manager Agent Bootstrap Prompt creation.

I'll create the Manager Agent Bootstrap Prompt following the format specified in the Setup Agent Initiation Prompt.

```markdown
---
Use: github
Memory strategy: dynamic-md
Asset format: md
Workspace root: /Users/apple/Desktop/Projects/APM Website/agentic-project-management-website
---

# Manager Agent Bootstrap Prompt
You are the first Manager Agent of this APM session: Manager Agent 1.

## User Intent and Requirements
- Develop dual-purpose web application serving as professional landing page with void-like aesthetic and development methodology demonst
- Create modern Vite + React + TypeScript + Shadcn UI application with Design PRD compliant floating card design, glassmorphism effects,
- Implement transparent development coordination with commit message format: "[Agent Name] - [Phase Number] - [Task Number] -
```

Figure 5: Setup Agent presenting a Bootstrap Prompt in a markdown code block

7 Task Loop Phase

The Task Loop Phase is the central execution cycle in which Manager and Implementation Agents work together to systematically complete project tasks. The effectiveness of this phase relies heavily on the quality of the Implementation Plan produced during the Setup Phase.

7.1 Task Assignment

Manager Agents parse the Implementation Plan and generate Task Assignment Prompts for each task. **These prompts' structure is designed to capture and communicate all the information an Implementation Agent should need to successfully complete the assigned work.** The intent is for the Manager Agent to fill out each prompt as completely as possible, ensuring clarity and completeness. However, in practice, there may be occasions where the Manager Agent misses important context or details.

Since Task Assignment Prompts are Meta-prompts, Users are encouraged to **review them and request modifications, corrections, or additional information as needed** to ensure the prompts fully meet their requirements and expectations.

Task Assignment Prompt Modification: If the Task Assignment Prompt requires **considerable or substantial changes, it is best to describe the issues to the Manager Agent and ask them to re-issue the prompt** with the requested modifications applied. This ensures the Manager Agent is aware of the changes and there is no context gap, but note that doing so will incur additional token usage.

If the change needed is **small or purely editorial** such as clarifying a step, file-path details, or adding a brief contextual reference, **you can manually edit the prompt as you paste it into the Implementation Agent's input.** In this case, the Manager Agent will not be aware of the change, but for minor adjustments this is usually acceptable and more efficient.

7.2 Task Execution

Implementation Agents receive their task assignments and follow the provided instructions to complete each task. **For tasks that are the "consumer" in a producer-consumer cross-agent dependency, Task Assignment Prompts include explicit context integration instructions.** The agent performs context integration first to ensure complete understanding before proceeding with the main task. See Missing Context Integration Instructions for guidance if the manager omits these steps.

Task executions in APM fall into two main categories: **single-step** and **multi-step** tasks. Additionally, tasks vary in scope:

- **Agent-executed tasks:** Fully executed by the Implementation Agent within the IDE environment, using file operations, terminal commands, and other tools as needed.
- **User-executed tasks:** Steps requiring external actions (manual configuration, account setup, or access outside the IDE) are guided by the agent for the User to perform.
- **Ad-Hoc delegations:** Context-intensive or isolated subtasks may be delegated to an Ad-Hoc Agent for specialized handling.

This structure enables flexible task execution and ensures both automated and manual steps are clearly managed. See Task Execution Optimization for optimization strategies.

7.2.1 Task Execution Optimization

Task execution can be optimized for both efficiency and effectiveness, especially in multi-step tasks where iterative improvement between steps is possible.

Single-Step Tasks: For single-step tasks, there is generally little room for optimization, since these tasks should already be granular enough for Implementation Agents to complete in a single, focused response. However, you can still support your agents in the following ways:

- **Enhance Task Assignment Prompts:** Provide extra context with the Task Assignment Prompt, either by asking the Manager Agent to include more details or by manually adding information when pasting the prompt into the Implementation Agent's input.

  For example, if the prompt context integration instructions that include the reading of a file, you could attach it directly using your IDE's context tools. This allows the agent to skip the file read step, since the contents are already available, letting them focus entirely on the main task.

- **Request Explanations:** At any stage, you can request agents to clarify their reasoning and outputs, and they will respond according to their interaction model protocols. Requesting explanations can deepen your understanding of the task execution, especially for tasks outside your expertise. Even if it does not directly affect the agent's performance, this practice helps you catch potential issues early.

Multi-Step Tasks: Multi-step tasks offer greater opportunities for optimization, as the natural pause points between steps enable iteration, clarification, and user feedback. This allows the User to guide and adjust the execution of subsequent steps as needed. For example:

- **Iterative Improvement:** If a step is not successful or not meeting your expectations, you can ask the agent to revise the plan and try again, before moving on to the next step.

- **Combine Steps:** If there are related steps that can be combined, you can ask the agent to combine them. When choosing to combine steps, provide explicit instructions such as:
  ```
  "Step 2 looks fine. Combine steps 3-4 and log in your next response."
  ```

  Step Combination Warning: Multi-step tasks are classified as such because of their complexity and/or internal sequential dependencies. Sometimes, combining steps can have the opposite effect and introduce confusion or errors. **Only combine steps that you, deem are closely related and can be safely executed together**. For complex or high-risk work, keep steps separate to allow for proper validation and control at each stage.

- **Request Explanations:** Similarly to single-step tasks, you can request explanations at any step of a multi-step task, using the natural checkpoints for feedback and clarification.

Ad-Hoc Delegations: If a task contains an isolated, context-intensive subtask, you can delegate it to an Ad-Hoc Agent. Without a formal guide, the Implementation Agent will improvise a suitable delegation prompt for your review. For reference, see this delegation example.

Ad-Hoc Delegation Tip: Ad-Hoc delegations are particularly useful for context-intensive subtasks whose purpose is to provide operational context for the Implementation Agent's main task. For research-related subtasks, APM v0.4 includes a formal Research Delegation Guide to streamline the process.

7.2.2 Memory Logging Optimization

After completing a task, Implementation Agents record their insights and findings in a Memory Log. The Manager Agent then reviews this log to evaluate progress and determine the appropriate next steps. Based on the Memory Log, the Manager may advance to the next task, re-issue a Task Assignment Prompt for corrections, or update the Implementation Plan as needed.

Memory Logs are essential because they both provide the Manager Agent with the necessary context to review completed tasks and also serve as a key context resource during Handover Procedures when Implementation Agents are replaced. You can improve the usefulness of Memory Logs by instructing the Implementation Agent to:

- **Include additional context** that will help the Manager Agent during task review.
- **Document important insights**, such as workflow preferences or special considerations, so that future agents can easily reference them during handovers.

7.3 Task Review

After the Implementation Agent has completed a task, the Manager Agent will review the Memory Log to assess the situation and determine the appropriate next steps, a crucial point that **demands User supervision.**

To initiate the task review, simply state that the task is completed and instruct the Manager to review the Memory Log. For example:

```
"Task 4.2 is completed, please review the log."
```

At this stage, the Manager Agent may suggest updates to the Implementation Plan, typically explaining its reasoning and requesting your confirmation before making any changes so you can approve, modify, or ask for clarification. **If the Manager Agent proceeds without pausing for your input, make sure to review any changes and request corrections as needed.**

For guidance on influencing the Manager Agent's next steps after a task, see this example.

Implementation Plan Updates Tip: The best time to request changes to your Implementation Plan is right after a task is completed and the Manager Agent is reviewing the Memory Log. **Keep in mind that the Manager Agent will not always notice when the Implementation Plan needs changes or updates.** If you want to be sure your requirements are reflected, you must explicitly state any desired changes or corrections when you return to the Manager Agent chat. For example:

```
"Task 4.2 is completed, please review the log. During its task execution, I
realized that the Landing Page design needs a major revision. I think it would
be best to insert a new phase after phase 4 for design iteration with a Design
Agent before moving on to phase 5."
```

Implementation Plan Update Guidance: When requesting revisions from the Manager Agent, provide clear, detailed instructions for all required changes. Carefully review dependencies and related tasks to avoid context gaps or broken links.

For major or structural changes, consider returning to the Setup Agent for systematic revision. See Implementation Plan Needs Revisions for further guidance.

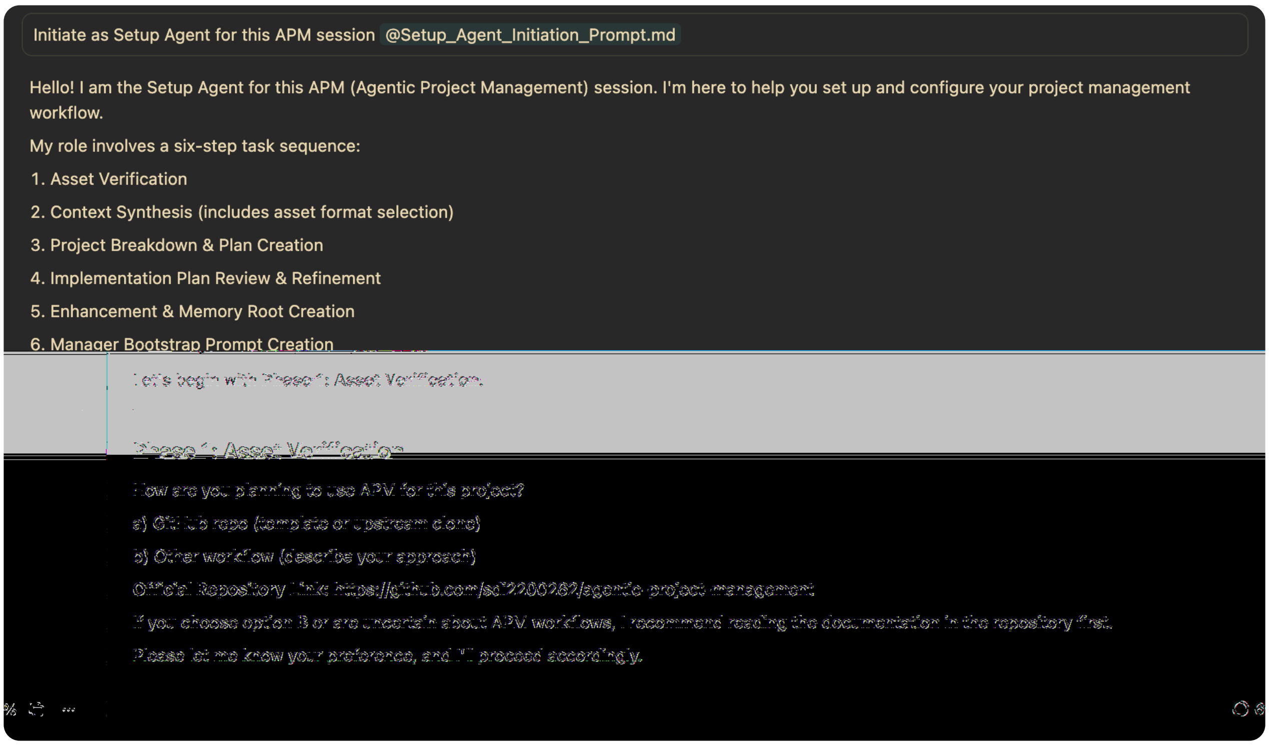
8 Handover Procedures

Handover Procedures ensure session continuity as agents approach context limits. APM's two-artifact system allows for seamless transfer of both the session state and undocummented context of the previous agent. Handovers are critical to minimize the risk of context loss and hallucinations for long-running sessions.

8.1 Timing Strategies

Initiating a handover at the right moment is essential. The general principle is **"Better safe than sorry"**. Consider the following strategies:

- **Monitor context window usage:** Use your AI IDE's context window visualization tools to track how much of the available context is being used. Plan to initiate a handover when usage reaches 80–90% of the model's context window. See Figure 5 for Cursor's example.
- **Monitor for declining agent performance:** Watch for warning signs like inconsistent decisions, fabricated details, or the agent forgetting recent instructions. These often signal that the context window is nearly full or has been exceeded.
- **Time handovers for upcoming demands and continuity:** If a complex or multi-step task is ahead, initiate the handover early so the new agent has enough context window to finish uninterrupted. Avoid disrupting critical operations, but don't wait until performance declines.


Initiate as Setup Agent for this APM session @Setup_Agent_Initiation_Prompt.md

Hello! I am the Setup Agent for this APM (Agentic Project Management) session. I'm here to help you set up and configure your project management workflow.

My role involves a six-step task sequence:

1. Asset Verification
2. Context Synthesis (includes asset format selection)
3. Project Breakdown & Plan Creation
4. Implementation Plan Review & Refinement
5. Enhancement & Memory Root Creation
6. Manager Bootstrap Prompt Creation

Let's begin with Phase 1: Asset Verification.

Phase 1: Asset Verification

How are you planning to use APM for this project?

a) GitHub repo (template or upstream clone)

b) Other workflow (describe your approach)

Official Repository Link: https://github.com/sdi2200262/agentic-project-management

If you choose option B or are uncertain about APM workflows, I recommend reading the documentation in the repository first.

Please let me know your preference, and I'll proceed accordingly.

Figure 6: Cursor's context window visualization example - bottom right corner.

Visual Context Window Indicators: Many AI IDE platforms offer similar context window indicators, **while some may not provide this feature at all**. Be sure to familiarize yourself with the capabilities of the specific platform you are using.

8.1.1 Handover Timing Rules of Thumb

Use These Rules of Thumb if your AI IDE platform does not provide a visual context window indicator

- **Implementation Agents:** Consider initiating a handover after **5-10 task cycles**, depending on task complexity and context depth. Task complexity can be estimated by the number of steps or instructions in the task and their own complexity. On some occasions, handovers may need to be even more frequent.
- **Manager Agents:** Manager Agents do not execute tasks, therefore their cycles are not dependent on task complexity. Instead, they are dependent on next action assessment and decision making that happens after every task review. If a task has revealed a complex issue or significant insight, the Manager Agent will need to 'spend more context' assessing the situation and making a decision.
  - For the **first Manager Agent instance** which parses the Bootstrap Prompt and initializes the Task Loop Phase, consider initiating a handover after **10-15 task cycles.**
  - For **replacement Manager Agents** created after a previous handover, consider initiating a handover after **15-20 task cycles**.

8.1.2 Performance Warning Signs

Watch for these indicators that an agent is approaching its context window limits:

- **Repetitive Questions:** Asking for information it was given earlier in the conversation
- **Inconsistent Responses:** Contradicting previous advice or decisions
- **Forgotten Context:** Not remembering key project details or requirements
- **Generic Responses:** Providing less specific, more generic advice than usual
- **Plan Confusion:** Misremembering task assignments or project structure

These signs are especially useful for tracking context window usage when your IDE lacks a context window visualization feature.

Manager Agent Handover Importance: Manager Agents are responsible for coordinating the project and making decisions. They are the most important agent type of the Task Loop Phase. It's important to perform handover with a Manager Agent that's still performing well than to wait until it starts making mistakes.

Doing a late handover with a Manager could result in a significant loss of context and session breakage. See Handover Problems for recovery steps if this occurs.

Implementation Agent Handover Flexibility: Context window issues with Implementation Agents are typically **less disruptive** because of their narrowed focus and task-specific context.

If handover issues arise with these agent types, **essential information can often be restored or repaired with minimal disruption**. See Handover Problems for recovery steps.

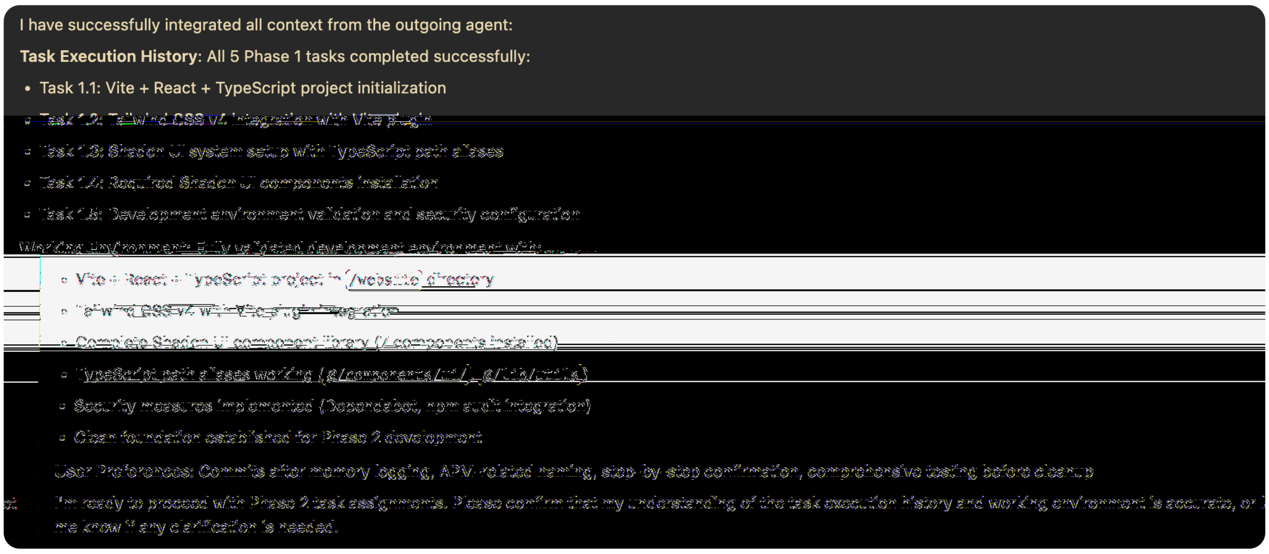
8.2 Verification Process

After a handover, it is vital to confirm that the new agent has fully understood the transferred context and is ready to proceed. After providing the Handover Prompt to the new agent, it will follow its instructions and provide a summary of its understanding. Verify the summary with the following steps:

- **Confirm awareness of recent decisions:** Ensure the agent can accurately recount the most recent project decisions, context changes, and rationale. Look for any inconsistencies or omissions in the agent's summary compared to the actual project history.
- **Check for contradictory or hallucinated details:** Scrutinize the agent's summary for any information that was not present in the original context or Memory Logs. Watch for invented facts, spurious reasoning, or details that do not align with the documented project state.
- **If you are suspicious, ask clarifying questions:** If the agent's summary seems unclear, inconsistent, or incomplete, ask specific questions to verify its understanding of key project aspects (such as dependencies, unresolved issues, or user preferences). This helps ensure the agent's grasp of the context is accurate and complete.

Verification Tip: If a summary is not broken (i.e. contaminated by hallucinations), but is instead incomplete or has slight gaps, **consider providing additional context or clarification to the agent before authorizing it to proceed.** See Recovering from a Late Handover for additional guidance.

Late Handover Risks: Waiting too long to initiate a handover can seriously compromise context integrity. When handovers are delayed until agent performance has already degraded, context corruption results in lost information, broken continuity, and the need for difficult recovery or even project rollback. **These issues are not always obvious in the new agent's summary, so be especially vigilant when verifying handovers that occur late or when the context window was nearly full.**



I have successfully integrated all context from the outgoing agent:

Task Execution History: All 5 Phase 1 tasks completed successfully:

- Task 1.1: Vite + React + TypeScript project initialization
- Task 1.2: Tailwind CSS v4 integration with Vite plugin
- Task 1.3: Shadcn UI system setup with TypeScript path aliases
- Task 1.4: Required Shadcn UI components installation
- Task 1.5: Development environment validation and security configuration

Working Environment: Fully validated development environment with:

- Vite + React + TypeScript project in `/website` directory
- Tailwind CSS v4 with Vite plugin integration
- Complete Shadcn UI component library (7 components installed)
- TypeScript path aliases working (`@/components/ui/`, `@/lib/utils`)
- Security measures implemented (Dependabot, npm audit integration)
- Clean foundation established for Phase 2 development

User Preferences: Commits after memory logging, APM-related naming, step-by-step confirmation, comprehensive testing before cleanup

I'm ready to proceed with Phase 2 task assignments. Please confirm that my understanding of the task execution history and working environment is accurate, or l
me know if any clarification is needed.

Figure 7: Replacement agent presenting a summary of its understanding of the transferred context.

9 Optimization Strategies

You can enhance the efficiency of your APM sessions through a variety of strategies, including thoughtful model selection for each agent type, streamlining task execution, and tailoring assets to fit your specific standards.

9.1 Model Selection Economics

Strategically selecting models according to agent roles and task complexity can greatly optimize costs without sacrificing quality. APM is designed to support flexible model assignment, allowing you to tailor economic strategies to your needs. For each agent type, you will find specific model recommendations in the following sections:

- **Setup Agent:** See subsection 3.1
- **Manager Agent:** See subsection 3.2
- **Implementation Agent:** See subsection 3.3
- **Ad-Hoc Agent:** See subsection 3.4

For detailed economic model strategies and token consumption analysis, refer to the Token Consumption Tips document.

9.2 Workflow Efficiency

Enhance your APM workflow by applying the best practices and strategies outlined below.

9.2.1 Setup Phase Optimization

See Setup Phase section for detailed strategies on:

- Preparing comprehensive materials beforehand
- Providing detailed responses during Context Synthesis
- Thoroughly reviewing Implementation Plans before approval

9.2.2 Task Execution Optimization

Refer to Task Execution Optimization for techniques including:

- Step combination strategies for multi-step tasks
- Context enhancement for single-step tasks
- Effective use of Ad-Hoc delegations

9.2.3 Memory Logging Optimization

See Memory Logging Optimization for approaches to:

- Including relevant context for Manager review
- Documenting insights for future handovers
- Maintaining comprehensive project history

9.2.4 Proactive Handover Planning

Refer to Handover Timing Strategies for guidance on:

- Proactively track context usage during the session, either by using your IDE's context window visualization feature if available, or relying on the recommended Rules of Thumb
- Planning handovers to occur on natural break points; ensuring current task cycles are finished before initiating a handover
- Initiating handovers during periods of lower task complexity for smoother transitions; avoiding disrupting critical operations

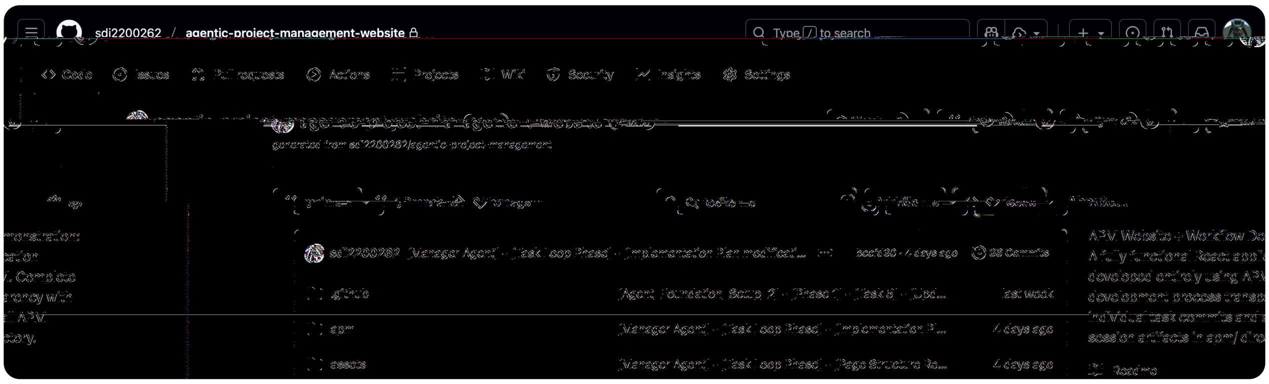
9.3 Optimizing APM Assets

Every project has its own unique requirements, and you can maximize APM's effectiveness by creating **customized APM assets** tailored to your needs. By using an APM GitHub template, you can modify prompts, guides, and workflows to better align with your project's goals.

Customizing APM assets offers two major advantages:

- **Project Scope Alignment:** If your project operates in a highly specialized domain, you can adjust Context Synthesis questions and refine the Setup Agent's project discovery strategy to focus more deeply on that field. This ensures that during the Task Loop Phase all agents are primed with the most relevant context from the outset.
- **Process Optimization:** You can fine-tune the Implementation Agent's task execution sequence to match your workflow. For example:
  - If version control is important to your development process, **add a mandatory instruction in the Implementation Agent Initiation Prompt requiring the agent to always commit changes to a dedicated branch** after completing each task and its associated logging.
  - If your team relies on automated testing, you could **require the Implementation Agent to run a specific test suite and report results as part of every task cycle**, ensuring quality is maintained throughout development.

For guidance and ideas on how to modify APM assets please refer to the Modifying APM document.



generated from sdi2200262/agentic-project-management

sdi2200262 [Manager Agent] - [Task Loop Phase] - [Implementation Plan modificat... · 4 days ago · 38 Commits

.github [Agent] Foundation Setup 2 - [Phase 1] - [Task 5] - [Upd... last week

apm [Manager Agent] - [Task Loop Phase] - [Implementation P... 4 days ago

assets [Manager Agent] - [Task Loop Phase] - [Page Structure Re... 4 days ago

APM Website + Workflow De
A fully functional React app
developed entirely using APM
development process transp
individual task commits and
session artifacts in apm/ dir

Figure 8: Example of a customized APM GitHub template repository. In this modified workflow, every agent is required to commit their changes after each action, using a specific message format.

10 Troubleshooting

Effective troubleshooting prevents minor issues from becoming project blockers. This section covers common scenarios and recovery strategies.

10.1 Setup Phase Issues

Issues in the Setup Phase can affect the entire project, so resolve them before moving to the Task Loop Phase. If major problems arise, consider re-starting to ensure a successful session.

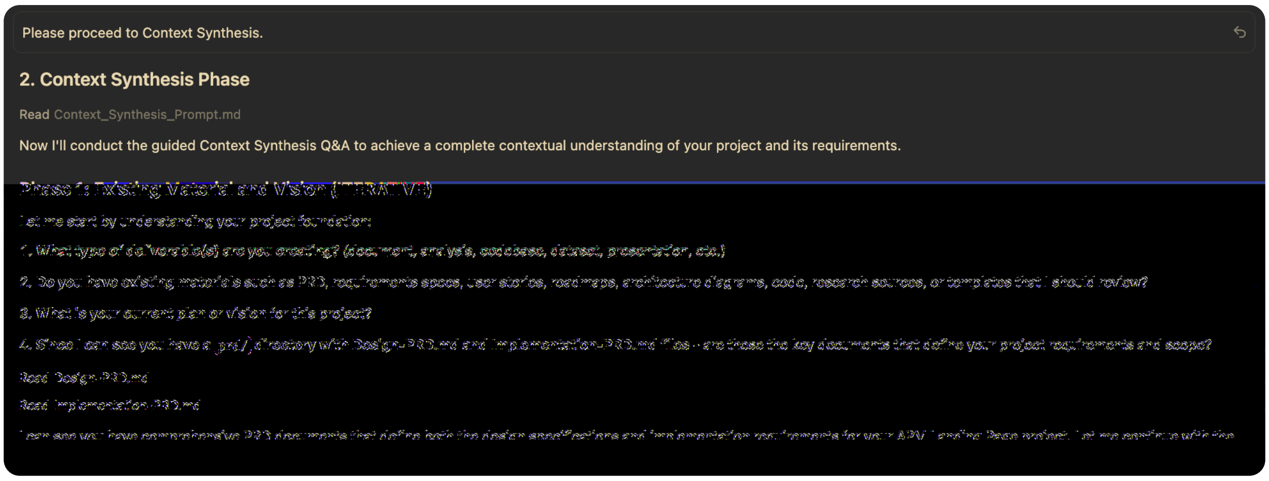
10.1.1 Omitted Requirements During Context Synthesis

If you forget to provide a requirement during its intended Context Synthesis phase and remember it later on, **the impact depends on how critical the information is and how far the project discovery has progressed.**

- **Minor Details:** If you recall a small or non-critical detail (e.g., a minor tech stack preference or secondary feature) after the relevant Context Synthesis phase, you can provide this information in a later phase or in response to a follow-up question. The Setup Agent is generally able to incorporate such minor additions without disrupting the project discovery process.
- **Major or Foundational Requirements:** If you omit a major requirement, such as a full PRD, core architectural document, or critical specification, after the relevant Context Synthesis phase has already progressed, it can **severely compromise the Setup Agent's understanding and the resulting Implementation Plan.**

Recommended Action:
If you discover that a major requirement or foundational document was not provided during Context Synthesis, it is best to **roll back and restart the Setup Phase from the beginning of Context Synthesis**. Doing so early is far less costly than having to redo the entire Setup Phase after more token and time-consuming steps like Project Breakdown.

Context Synthesis Early Context Tip: Always provide required information in the order the Setup Agent requests it, especially high-level overviews and key documents (like PRDs) during the first phase. **Early context shapes the agent's project discovery strategy.**



Please proceed to Context Synthesis.

2. Context Synthesis Phase

Read Context_Synthesis_Prompt.md

Now I'll conduct the guided Context Synthesis Q&A to achieve a complete contextual understanding of your project and its requirements.

Phase 1: Existing Material and Vision (ITERATIVE)

Let me start by understanding your project foundation:

1. What type of deliverable(s) are you creating? (document, analysis, codebase, dataset, presentation, etc.)

2. Do you have existing materials such as PRD, requirements specs, user stories, roadmaps, architecture diagrams, code, research sources, or templates that I should review?

3. What is your current plan or vision for this project?

4. Since I can see you have a `prd/` directory with Design-PRD.md and Implementation-PRD.md files - are these the key documents that define your project requirements and scope?

Read Design-PRD.md

Read Implementation-PRD.md

I can see you have comprehensive PRD documents that define both the design specifications and implementation requirements for your APM Landing Page project. Let me continue with the

Figure 9: Setup Agent reading PRDs during Phase 1 of Context Synthesis

10.1.2 Interrupted Project Breakdown

The Project Breakdown is a structured process designed to enforce CoT reasoning to widely available non-thinking models. Its effectiveness depends on both your chosen Setup Agent model and the system prompts of your AI IDE, as either an inadequate model or conflicting system prompts can interrupt the process.

Recommended Action:

- **If the issue is model-related:** Instruct the agent to continue after each incomplete or fragmented response until the entire Project Breakdown is finished. Be clear and specific in your guidance. For example:

  ```
  "You stopped the project breakdown after Phase 3. Phases 1 through 3 are
  complete; please continue and finish the breakdown for phases 4 and 5."
  ```

  If you wish to upgrade to a more capable, agentic model (such as Claude Sonnet 4), you may do so, but see the warning below regarding context gaps and mitigation.

  Model Switching Warning: Switching models during Project Breakdown can introduce context gaps. To mitigate this, instruct the new agent to re-do the **entire** Project Breakdown sequence from the beginning, to ensure the entire sequence is in the new model's context window.

- **If the issue is due to IDE system prompts:** Some system prompts can still cause interruptions, even with a suitable model. Instruct the agent to continue after each fragmented response until finished, and be clear and specific in your instructions as above.

Chat Checkpoint Tip: If your IDE supports checkpoint retrieval or message editing, use these features to restart Project Breakdown cleanly right after Context Synthesis.

Project Breakdown Interruption Warning: If you experience interruptions, carefully review the agent's output for alignment with your expectations and APM standards. If the sequence becomes disjointed after multiple interruptions, assess your current context window usage and decide next steps based on session continuity.

10.1.3 Interrupted Implementation Plan Enhancement

Interruptions during Implementation Plan Enhancement are uncommon, but when they occur (due to model or IDE system prompt issues), **follow the same troubleshooting steps as for Project Breakdown Interruption.**

Alternative Recovery: If the enhancement phase is interrupted near the context window limit, you can start a new Setup Agent session, repeat Context Synthesis to onboard the agent with the project requirements, and ask the agent to skip Project Breakdown & Review by providing the finalized Implementation Plan for enhancement. **In this approach, the remainder of the Setup Phase will be completed in the new Setup Agent chat session.**

Experimental Recovery Warning: This alternative recovery method is experimental and should only be used as a last resort. Proceed with caution, carefully verifying that the resulting Implementation Plan is complete and meets your requirements before proceeding.

10.2 Task Loop Issues

Troubleshooting problems that arise during the Task Loop phase requires specific, context-aware solutions tailored to the nature of each issue. Often, these problems stem from incomplete or poor-quality work in the Setup Phase.

10.2.1 Missing Context Integration Instructions

Sometimes, the Manager Agent may **omit explicit context integration steps in Task Assignment Prompts** for tasks that rely on work from another agent. This can lead to context gaps and confusion during execution.

How to identify:
In the Implementation Plan, **tasks with cross-agent dependencies are marked with a explicit tag** ”... by Agent X”, where Agent X is not the agent instance assigned to the current task. If you see a Task Assignment Prompt for one of these tasks that does not include clear context integration instructions, this issue is present.

Recommended Action:
Notify the Manager Agent and **request a revised Task Assignment Prompt** that explicitly includes the missing context integration steps. For example:

```
”I noticed that the Task Assignment Prompt you issued for task 4.2 does not include
context integration instructions, even though there is a cross-agent dependency
with task 3.8. Please re-issue this prompt, this time including the necessary
context integration steps.”
```

10.2.2 Implementation Plan Needs Revisions

It is common for the Implementation Plan to require updates as your project evolves, especially if the initial context provided to the Setup Agent was incomplete. Revisions may be needed due to new requirements, changes in project scope, or the discovery of issues.

How to identify:
Watch for signs such as missing important steps, outdated information, or misalignment with recent project decisions and your current goals. **Sometimes, the Manager Agent will flag the need for updates during task review.**

Types of Revisions:

- **Minor Revisions:** Small changes such as adding a new task, updating task details, or making minor adjustments based on recent insights. **These are typically manageable within the Manager Agent’s capabilities.**
- **Major Revisions:** Significant changes that affect the overall structure of the plan, such as switching core technologies, adding or removing entire phases, or responding to major shifts in requirements. **These usually require the Setup Agent’s involvement.**

Minor Revision Tip: Minor revisions can include inserting entire new phases with 3–4 tasks, as long as the rest of the Implementation Plan is not significantly affected. For example, after reaching a major milestone, you might want to add a ”Testing Phase” with tasks to comprehensively test core components.

Recommended Action:

- **For Minor Revisions:** Ask the Manager Agent to update the Implementation Plan, and **be sure to give clear, detailed instructions about the changes you want, including the structure and content of any additions**. For example:

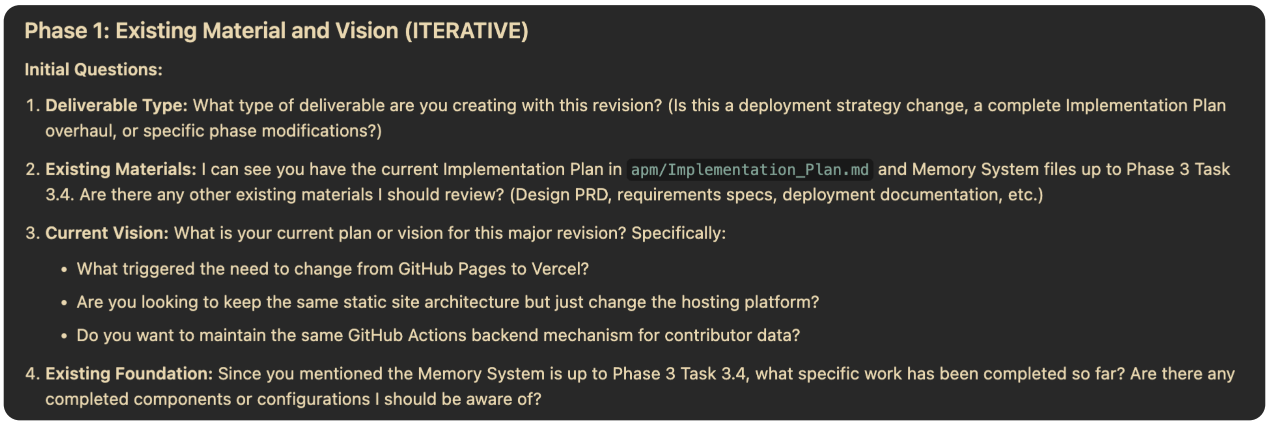
  ```
  "Please insert a new multi-step 'Documentation Update' task after 2.3 to cover
  the new API endpoints we identified as missing during the latest review, with
  steps for drafting endpoint descriptions, usage examples, and updating the API
  reference; ensure all related tasks reflect these additions."
  ```

  Clear Instructions Needed: The Manager Agent lacks the Setup Agent's systematic Project Breakdown sequence. Without explicit and detailed instructions about the changes you want, your Implementation Plan may be incomplete or inaccurate.

- **For Major Revisions:** For extensive changes, you should return to the Setup Agent, to utilize the strategic project discovery and systematic Project Breakdown sequence. However, **you should take into consideration the context window your initial Setup Phase consumed.**
  - **Sufficient context window:** If your original Setup Agent session still has enough available context window, simply proceed with the revision in that session.
  - **Context window nearly full:** If the context window is nearly full, **start a new Setup Agent session and repeat the Context Synthesis step**, providing the original project context, a summary of progress, and a clear description of the required changes.

    Progress Summary Tip: When summarizing progress, include the previous Implementation Plan and clearly indicate which tasks have been completed. This ensures the agent has full context and can focus revisions only where needed during the subsequent Project Breakdown.

  For both cases: At the Project Breakdown step, **instruct the agent to break down only the areas needing revision**. For the Enhancement step, **only enhance the newly broken-down sections** and **skip Memory System initialization** (since it is already set up). Once done, **copy the new Bootstrap Prompt and send it to the Manager Agent**, explaining that major revisions were made to the Implementation Plan and highlighting the updated sections.


Phase 1: Existing Material and Vision (ITERATIVE)

Initial Questions:

1. **Deliverable Type:** What type of deliverable are you creating with this revision? (Is this a deployment strategy change, a complete Implementation Plan overhaul, or specific phase modifications?)
2. **Existing Materials:** I can see you have the current Implementation Plan in `apm/Implementation_Plan.md` and Memory System files up to Phase 3 Task 3.4. Are there any other existing materials I should review? (Design PRD, requirements specs, deployment documentation, etc.)
3. **Current Vision:** What is your current plan or vision for this major revision? Specifically:
   - What triggered the need to change from GitHub Pages to Vercel?
   - Are you looking to keep the same static site architecture but just change the hosting platform?
   - Do you want to maintain the same GitHub Actions backend mechanism for contributor data?
4. **Existing Foundation:** Since you mentioned the Memory System is up to Phase 3 Task 3.4, what specific work has been completed so far? Are there any completed components or configurations I should be aware of?

Figure 10: Setup Agent asking clarifying questions during Context Synthesis after requesting a major revision of the Implementation Plan

10.3 Handover Problems

Handover problems often occur when an agent exceeds its context window, leading to corrupted context before or after the handover. Common scenarios include:

- **Before handover:** You notice the outgoing agent has exceeded its context window and its context is already corrupted.
- **After handover:** You discover, during verification, that the transferred context is corrupted because the outgoing agent had already exceeded its context window.

10.3.1 Handling a Handover After the Context Window is Full

If the outgoing agent's context window is full or exceeded, its context is likely corrupted or contaminated with hallucinations. **Performing a standard Handover Procedure at this stage risks transferring inaccurate information to the new agent instance.**

How to identify:
You can often identify this issue by checking your IDE's context window visualization feature. Alternatively, you may notice that the agent's responses are inconsistent, incomplete, or contain hallucinated details.

Recommended Action:

- **Request only the Handover Prompt:** Instruct the outgoing agent to generate a Handover Prompt using the appropriate Handover Guide for its agent type, but **exclude the cross-reference validation section** and **do not create a Handover File**. For example:

```
"Perform a Handover Procedure following the Implementation Agent Handover
Guide, but only produce the Handover Prompt. Do not create a Handover File.
Also, exclude the cross-reference validation section from the Handover Prompt."
```

  Handover Prompt Caution: While the Handover Prompt follows a standard template and is generally reliable, **always carefully review all agent-generated or dynamic sections** such as Current Session State, Current Task Context or Immediate Next Action. These may be inaccurate or incomplete if the outgoing agent's context is corrupted. **Manually verify and correct as needed to ensure accuracy.**

- **Transfer Context:** Initialize a new chat session with a fresh agent instance of the same type, and provide the **reviewed Handover Prompt** along with a clear explanation of the situation. For example:

```
"Here is a modified Handover Prompt from the previous agent, whose context
window was exceeded. The Handover File was omitted to prevent transferring
corrupted context. Please use the Handover Prompt to restore documented
context, and ask me any questions needed to recover missing or undocumented
details. [Pasted Handover Prompt Follows]"
```

- **Restore missing context:** After the Handover Prompt, provide any undocumented details the agent may need such as workflow preferences, agent-to-user relationship etc.

Note: The Handover File is omitted in this scenario to avoid transferring corrupted information. Work with the agent to restore any missing context directly after passing the Handover Prompt.

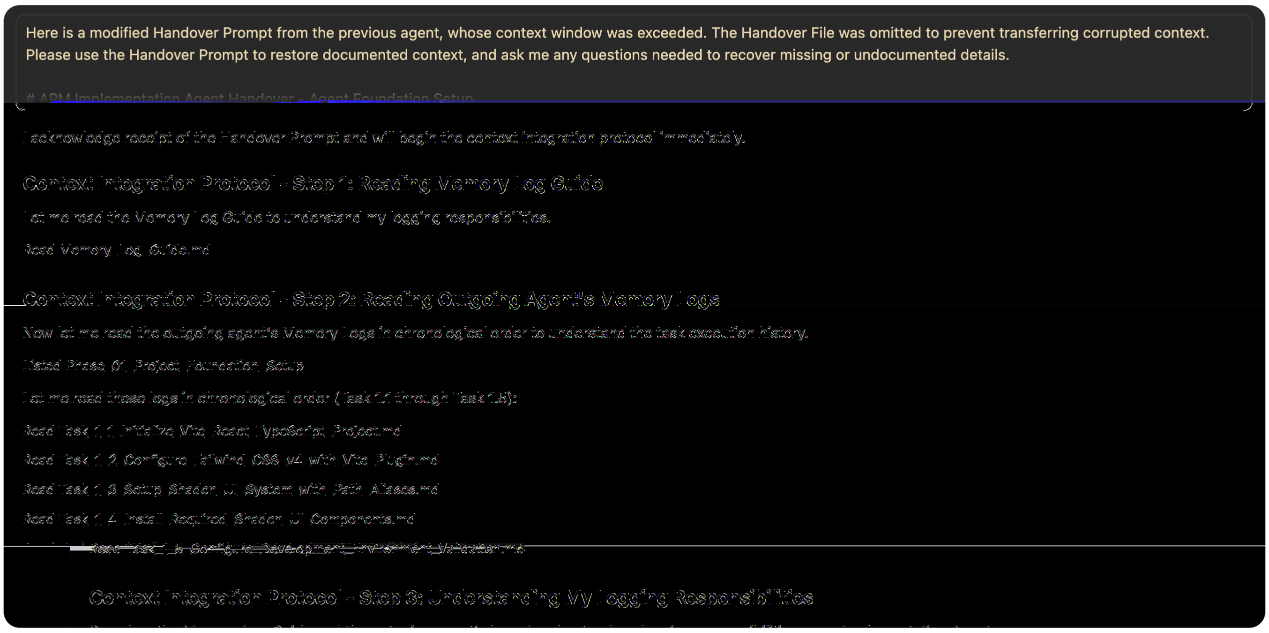
10.3.2 Recovering from a Late Handover

Sometimes, you may only realize **after** a handover has been completed that the outgoing agent had exceeded its context window or transferred corrupted context. In these cases, the replacement agent has already received and processed the Handover Prompt and the Handover File, and also provided a summary of its understanding, at which point you noticed gaps, inconsistencies, or outright errors. This is called a **late handover**.

How to identify:
You notice, during the verification step (when the new agent summarizes its understanding), that the summary contains missing, inconsistent, or hallucinated details, or otherwise reflects corrupted or incomplete context.

Recommended Recovery Actions:

- **If the context is mostly intact (minor gaps or errors):**
  - Review the agent's summary carefully.
  - Provide clarifications, corrections, or missing details directly in response. This may include re-explaining recent project decisions, clarifying workflow preferences, or correcting any misunderstandings.
  - Continue the session as normal once the agent's understanding is accurate and complete.
- **If the context is heavily corrupted (major errors, hallucinations, or critical omissions):**
  In this case, follow the recovery procedure described in Handling a Handover After the Context Window is Full, with the key difference that you must **manually review and repair the already generated Handover Prompt** before providing it to the new agent. The Handover File that has been generated by the outgoing agent is unreliable and should be omitted.



Here is a modified Handover Prompt from the previous agent, whose context window was exceeded. The Handover File was omitted to prevent transferring corrupted context. Please use the Handover Prompt to restore documented context, and ask me any questions needed to recover missing or undocumented details.

APM Implementation Agent Handover - Agent Foundation Setup

I acknowledge receipt of the Handover Prompt and will begin the context integration protocol immediately.

Context Integration Protocol - Step 1: Reading Memory Log Guide

Let me read the Memory Log Guide to understand my logging responsibilities.

Read Memory_Log_Guide.md

Context Integration Protocol - Step 2: Reading Outgoing Agent's Memory Logs

Now let me read the outgoing agent's Memory Logs in chronological order to understand the task execution history.

Listed Phase_01_Project_Foundation_Setup

Let me read these logs in chronological order (Task 1.1 through Task 1.5):

Read Task_1_1_Initialize_Vite_React_TypeScript_Project.md

Read Task_1_2_Configure_Tailwind_CSS_v4_with_Vite_Plugin.md

Read Task_1_3_Setup_Shadcn_UI_System_with_Path_Aliases.md

Read Task_1_4_Install_Required_Shadcn_UI_Components.md

Read Task_1_5_Configure_Development_Environment_Validation.md

Context Integration Protocol - Step 3: Understanding My Logging Responsibilities

Figure 11: A replacement agent completes the Handover Procedure using only a manually reviewed and corrected Handover Prompt.

11 Summary

APM transforms chaotic AI interactions into structured, efficient workflows capable of handling complex projects with confidence and clarity. This is achieved through systematic coordination between agents, comprehensive memory management, and seamless context transfer between agent instances.

For comprehensive documentation about the APM framework, including setup, agent roles, and workflow best practices, please visit the documentation suite in the GitHub repository.

For in-depth information on advanced prompt and context engineering, as well as the context and memory management techniques incorporated in APM, see these documents:

- Context and Memory Management
- Context and Prompt Engineering

11.1 Key Takeaways

As you begin using APM , remember these essential principles:

- **Structure Enables Flexibility:** The Setup Phase creates a foundation that makes execution highly efficient and adaptable
- **Context is Key:** Invest time in thorough Context Synthesis and Project Breakdown during Setup Phase; it prevents downstream problems. Comb through the Implementation Plan carefully before approval and request modifications or revisions as needed.
- **Proactive Management:** Initiate handovers before agents hit limits to maintain performance quality. Many agents can survive several handovers, but as projects progress context gets accumulated and transfer becomes inefficient.
- **You Are the Bridge:** Your role as communication coordinator between agents is crucial for project success. You are responsible of overseeing the interactions between agents, task execution and memory management, and are able to intervene when necessary.

11.2 Optimizing Your APM Workflow

This guide is designed to help you further refine your workflow, deepen your understanding of advanced features, and troubleshoot common scenarios.

Continue to leverage APM's modular structure to adapt the framework to your evolving project needs.

For specialized or complex projects, consider tailoring your APM assets by customizing prompts, guides, and workflows to suit your specific needs. For detailed instructions on how to modify APM, refer to the Modifying APM document.

Happy project managing!