­­­­­

|  |  |
| --- | --- |
|  |  |



Solution Design

Document

Table of Contents

[I. Purpose 3](#_Toc5787525)

[II. Automated process details 4](#_Toc5787528)

[3 Runtime guide 5](#_Toc5787529)

[3.1 Architectural structure of the Master Project 5](#_Toc5787530)

[3.2 Master Project Runtime Details 5](#_Toc5787531)

[3.3 Project name 1 6](#_Toc5787532)

[3.4 Project(s) workflows 6](#_Toc5787533)

[3.5 Packages 7](#_Toc5787534)

[3.6 Architectural structure of the Master Project 7](#_Toc5787535)

[4 Other Details 8](#_Toc5787536)

[Future Improvements 8](#_Toc5787537)

[Other Remarks 8](#_Toc5787538)

[5 Glossary 9](#_Toc5787539)

# Purpose



Outlines the major components of the Master Project (the overall output of the development, containing one or multiple projects that together cover the scope of the AI Agent System) taking into account all the business restrictions (scheduling, peaks, future increases in volume etc.). The focus of the Solution Architect will be on:

* Robustness;
* Scalability;
* Efficiency;
* Replicability

The information herein is targeted primarily at the developers that will initially implement the solution and subsequently at the support developers in case of change requests.

Release Notes Agent Chain  
  
1. Business goal  
The primary business goal of this process is to effectively communicate software updates and changes to different stakeholders within an organization. By creating tailored release notes for Program Managers, Engineers, and C-suite executives, the process aims to ensure that each group receives relevant, easily digestible information about the latest software developments. This targeted communication helps to improve understanding, decision-making, and alignment across various levels of the organization, ultimately supporting better product management, development processes, and strategic planning.  
  
2. Best Way to Accomplish This Business Goal  
The best way to accomplish this goal is through a combination of automation and human oversight. An automated system should handle the bulk of the work, including data gathering from GitHub, analysis of commits and issues, and initial drafting of audience-specific release notes. However, a human review step should be incorporated to ensure accuracy, relevance, and appropriate tone for each audience. This approach balances efficiency with quality control. Additionally, implementing a feedback loop from recipients would help refine the process over time, ensuring that the release notes continue to meet the needs of each audience group. Regular training of the AI system with this feedback would further improve its performance.  
  
3. How AI Agents Could Improve the Process  
a. Natural Language Processing (NLP): Advanced NLP models like GPT could be used to analyze commit messages, issue descriptions, and code changes more effectively. They could better understand the context and importance of each change, leading to more accurate and relevant release notes.  
  
b. API Integration: AI agents could more efficiently interact with GitHub's API, potentially using GraphQL to fetch precisely the needed data in fewer requests. This would speed up the data gathering process and reduce unnecessary data transfer.  
  
c. Adaptive Learning: By incorporating machine learning, the AI agent could learn from past release notes and user feedback to continually improve its output. It could learn to better categorize changes, prioritize information, and tailor language for each audience group.  
  
d. Predictive Analytics: AI could analyze historical data to predict the impact of changes, helping to highlight potentially significant updates for each audience group.  
  
e. Automated Quality Assurance: Instead of relying solely on human review, an AI agent could perform initial quality checks, ensuring consistency, identifying potential errors or omissions, and even suggesting improvements.  
  
f. Dynamic Content Generation: Using GPT or similar models, AI agents could generate more nuanced, context-aware content for each audience group, potentially even personalizing release notes based on individual stakeholders' interests or roles.  
  
g. Intelligent Scheduling: AI could analyze past engagement data to determine the optimal time and channel for distributing release notes to each audience group.  
  
4. AI Agent High-level Steps  
a. Trigger Recognition and Data Retrieval  
i. Reasoning: The AI agent needs to recognize the request for release notes from Slack and initiate the process. It then needs to interact with GitHub's API to retrieve relevant data, including commits, issues, and pull requests since the last release. This step requires basic natural language understanding and API interaction, which are relatively straightforward for modern AI systems.  
ii. Complexity: 2/5  
  
b. Data Analysis and Categorization  
i. Reasoning: The AI agent must analyze the retrieved data, categorizing changes into features, bug fixes, documentation updates, and breaking changes. This requires a deeper understanding of the codebase and the ability to interpret commit messages and code diffs. While challenging, this is within the capabilities of advanced NLP models.  
ii. Complexity: 4/5  
  
c. Audience-Specific Content Generation  
i. Reasoning: Using the categorized data, the AI agent needs to generate tailored release notes for each audience (Program Managers, Engineers, C-suite). This involves summarizing technical details appropriately, highlighting relevant information, and adjusting the language and depth of explanation for each group. This step requires sophisticated natural language generation capabilities.  
ii. Complexity: 5/5  
  
d. Quality Assurance Check  
i. Reasoning: The AI agent should perform an initial quality check on the generated release notes, ensuring consistency, accuracy, and appropriate tone. This involves cross-referencing the notes with the original data and applying learned standards for each audience type.  
ii. Complexity: 3/5  
  
e. Human Review Prompt  
i. Reasoning: While AI can handle most of the process, a human review is crucial for ensuring the final quality and appropriateness of the release notes. The AI agent needs to prompt a designated human reviewer and potentially highlight areas that may need special attention.  
ii. Complexity: 1/5  
  
f. Feedback Incorporation and Distribution  
i. Reasoning: After human review, the AI agent should incorporate any feedback or changes, finalize the release notes, and distribute them through appropriate channels (e.g., Slack, email). It should also store this feedback for future learning.  
ii. Complexity: 2/5  
  
g. Exception Handling: Insufficient Data  
i. Reasoning: If there's insufficient data for meaningful release notes (e.g., no significant changes since the last release), the AI agent needs to recognize this, notify the requester, and suggest alternative actions.  
ii. Complexity: 3/5  
  
h. Exception Handling: API Access Failure  
i. Reasoning: If the AI agent cannot access the GitHub API, it needs to implement retry logic, log the error, and if persistent, notify a system administrator and the requester about the delay.  
ii. Complexity: 2/5  
  
i. Continuous Learning and Optimization  
i. Reasoning: The AI agent should continuously learn from the process, including human reviews and user feedback, to improve its performance over time. This involves updating its models and refining its decision-making processes.  
ii. Complexity: 5/5  
  
5. Process trigger  
Based on the process description, the trigger to start the process is a user asking a question about release notes in Slack. Specifically, the process states: "The process is triggered by a question from user on slack." This means that the AI system should be constantly monitoring the designated Slack channel(s) for any queries related to release notes, and once such a query is detected, it should initiate the release notes generation process.  
  
6. Human Involvement  
a. Quality Check: The process description includes a "Quality Check" step, which states: "Review generated release notes for accuracy and clarity. Make necessary adjustments." While an AI agent can perform initial quality checks, human involvement at this stage is crucial to ensure the final output meets the required standards. Humans can catch nuances, context-specific issues, or sensitive information that an AI might miss. This involvement should be brief, focusing on a final review and any necessary adjustments before distribution.  
  
b. Exception Handling: Although not explicitly stated in the process description, human involvement may be necessary in cases of exceptions or errors that the AI cannot resolve. For instance, if there's "Insufficient data for meaningful release notes," as mentioned in the exception handling section, a human might need to decide whether to proceed with limited information or delay the release notes. Similarly, in cases of persistent "GitHub API access failure," human intervention might be required to resolve the issue or make a decision on how to proceed.  
  
7. Documentation  
a. Example Documentation for Audience-Specific Release Notes  
i. Reasoning: The process description mentions "There's example documentation for what is a good response for each audiences." This suggests that there are existing examples or guidelines for creating release notes tailored to Program Managers, Engineers, and C-suite executives. Such documentation would be crucial for the AI agent to understand the expected format, tone, and content for each audience group. It would serve as a training dataset and reference point for the AI to generate appropriate release notes.  
ii. Document name or link: Not specified in the process description.  
  
b. GitHub API Documentation  
i. Reasoning: Although not explicitly mentioned in the process description, the AI agent would need to access and understand the GitHub API to retrieve repository data, including commits, issues, and pull requests. This documentation would be essential for implementing the data retrieval step of the process efficiently.  
ii. Document name or link: Not specified in the process description, but typically available at https://docs.github.com/en/rest  
  
c. Slack API Documentation  
i. Reasoning: The process is triggered by a Slack message, and the AI agent needs to interact with Slack to receive requests and potentially distribute release notes. Understanding the Slack API is necessary for implementing this integration.  
ii. Document name or link: Not specified in the process description, but typically available at https://api.slack.com/  
  
d. Internal System Architecture Documentation  
i. Reasoning: To effectively integrate with existing systems and distribute the release notes through appropriate channels, the AI agent would benefit from documentation outlining the organization's internal system architecture and communication protocols.  
ii. Document name or link: Not specified in the process description.  
  
e. Release Notes Best Practices Guide  
i. Reasoning: A guide on best practices for creating release notes would help the AI agent understand general principles of effective communication, regardless of the specific audience. This would complement the audience-specific examples.  
ii. Document name or link: Not specified in the process description.  
  
8. Solution Reasoning  
The proposed solution leverages AI agents to automate and streamline the process of generating and distributing audience-specific release notes. By breaking down the process into high-level steps and assigning complexity ratings, we can focus development efforts on the more challenging aspects while efficiently handling simpler tasks. The solution maintains a balance between automation and human oversight, ensuring quality and appropriateness of the final output.   
  
The AI-driven approach allows for continuous improvement through machine learning, adapting to feedback and evolving needs. By minimizing human involvement to critical quality checks and exception handling, we maximize efficiency while maintaining the necessary human touch for complex decision-making and final approval.  
  
The solution also emphasizes the importance of proper documentation and API integrations, ensuring that the AI agent has access to all necessary information and can interact seamlessly with existing systems. This comprehensive approach addresses the business goal of effective stakeholder communication while optimizing the process for speed, accuracy, and relevance.

# process details

Details filled in need to reflect the actual information for the Master Project released for production. The following table will be populated:

|  |  |
| --- | --- |
| Item | Description |
| Master Project Name |  |
| Framework used | e.g. 2019.4 |

# Runtime guide

## Architectural structure of the Master Project

Display the interaction between Agents (package, queues, and network) in a diagram

## Master Project Runtime Details

Outlines the details of the automated process by filling in the table below.

|  |  |
| --- | --- |
| ITEM NAME | DESCRIPTION  *Fill in each bolded section - empty fields are not allowed. If the section does not apply to your automation then mark as n/a.* |
| Production environment details | ***Example:*** *Running on Sparky , the virtual backoffice machine. Scheduled every night after the report is generated from Zendesk.* |
| Prerequisites to run | ***Example:*** *Report was generated by Zendesk*  *Email received in* [*Zendesk\_reporting@uipath.com*](mailto:Zendesk_reporting@uipath.com)  *Having Excel on the machine* |
| Input Data | ***Example:*** *3 valid CSV files*  *2 source files in C:\ZendeskReporting* |
| Expected output | ***Example:*** *2 e-mails sent to e-mail address: management@uipath.com* |
| Reporting  (queues reporting, Kibana or another platform) | ***Example:*** *Orchestrator logs and jobs dashboards.* |
| How is Orchestrator used? | ***Example:*** *Orchestrator used for scheduling and asset passwords.* |
| Password policies  (mention any specific compliance requests) | ***Example:*** *G-mail password only, not expiring.* |
| Stored credentials  (Never use hardcoded credentials in the workflow!) | ***Example:*** *Stored in Orchestrator Assets* |

## Project name

|  |  |
| --- | --- |
| ITEM NAME | DESCRIPTION  *Fill in each section - empty fields are not allowed. If the section does not apply to your automation then mark as n/a.* |
| Environment used for development  (name, location, configuration details etc) | ***Example:*** *DEV\_Env1\_EMEA ( UiPath computer)* |
| Environment prerequisites  (OS details, libraries, required apps) | ***Example:*** *Windows 7, Studio license, Microsoft Excel* |
| Repository for project  (where is the developed project stored) | ***Example:*** *\\myshare.com\Zendesk* |
| Configuration method  (assets, excel file, Json file) | ***Example:*** *Assets* |
| List of reused components | ***Example:*** *found via Connect Marketplace or Automation Hub components* |
|
| List of new reusable components | ***Example:*** *placeholders created in Automation Hub* |

Add tables for as many projects as you need and fill them in.

## Project(s) workflows

Workflows specific to: Specify Project Name from section above

For the workflow files defined below please specify the input and output parameters.

|  |  |
| --- | --- |
| Workflow Name | Description |
| Example: Main | ***Example:*** *invokes all the other workflows* |

## Packages

Include the list of packages and high-level description for each of them, to explain their purpose

|  |  |
| --- | --- |
| Package Name | Description |
| *Example: ZendeskReports.1.0.6285.31077.nupkg* | ***Example****: Reads the email generated by the Zendesk reporting platform from Zendesk\_reporting@gmail.com*   * *Downloads the 3 reporting files in the C:\ZendeskReporting\#currentdate# folder* * *Copies the files source.xlsx and source\_fantastic.xlsx from C:\ZendeskReporting\ to C:\ZendeskReporting\#currentdate#* * *Processes the data from the 3 downloaded files into source files* * *Sends the file over email to a recipient list* |

## Agents

Agent\_ID: 1  
Name: Slack Interaction Agent  
Description: This agent handles all interactions with Slack, both for receiving requests and distributing release notes.  
Reasoning: To manage all Slack-based interactions efficiently and effectively  
Tasks:   
 • Monitor designated Slack channels for release note requests: To detect and respond to user queries in real-time Complexity: 2  
 • Parse and interpret Slack messages to identify valid requests: To ensure only relevant requests are processed Complexity: 3  
 • Distribute final release notes through appropriate Slack channels: To deliver the completed release notes to the intended audience Complexity: 2  
 • Handle any Slack-based user queries during the process: To provide support and clarification throughout the release note generation process Complexity: 3  
Type: ReAct  
Context:   
 • Internal communication protocols and guidelines: To understand the appropriate channels, tone, and format for Slack communications within the organization,   
Inputs:   
 • Slack: Release note requests and user queries  
Outputs:   
 • Agent 2: Parsed release note request details  
 • Slack: Final release notes  
Tools: Slack API Integration Tool : Enables real-time monitoring of Slack channels, reading and parsing of messages, sending messages, and distributing release notes to appropriate channels..  
Trigger: This agent is constantly active, monitoring the designated Slack channel(s) for any queries related to release notes. It initiates the process when it detects a relevant query.  
Decisions:   
 • Determine if incoming Slack messages are valid release note requests: Message contains specific keywords related to release notes, follows a predefined format for requests, or comes from an authorized sender  
System Prompt: You are an intelligent Slack Interaction Agent, managing all release note communications within the organization's Slack workspace. Your tasks include monitoring designated channels for requests, interpreting messages to identify valid inquiries, distributing finalized release notes, and addressing user queries throughout the process.  
  
Utilize the Slack API Integration Tool (slack\_api\_tool) to monitor channels in real-time, parse messages, and send responses. Adhere to internal communication guidelines (slack\_communication\_guidelines) to align with organizational standards.  
  
When monitoring, identify messages containing release note keywords, following predefined formats, or from authorized senders. Parse these carefully to extract relevant information.  
  
Maintain a professional, friendly tone in all interactions. Respond promptly to queries with clear, concise information. When distributing final release notes, ensure proper channel selection and formatting for Slack readability.  
  
Your inputs come from Slack (release note requests and user queries). Outputs include parsed request details to Agent 2 and final release notes to Slack.  
  
Be proactive in your role. If you notice patterns in requests or queries, suggest improvements to the process. Regularly update your knowledge of company products and release cycles to provide more accurate and helpful responses.  
  
Your effectiveness is crucial for smooth communication and timely distribution of release notes. Prioritize accuracy, timeliness, and clarity in all interactions.  
  
Agent\_ID: 2  
Name: GitHub Data Retrieval Agent  
Description: This agent focuses on interacting with GitHub's API to fetch necessary data for release notes.  
Reasoning: To efficiently retrieve and prepare GitHub data for release note generation  
Tasks:   
 • Authenticate and establish connection with GitHub API: To securely access the required GitHub data Complexity: 2  
 • Fetch commits, issues, and pull requests since last release: To gather all relevant information for the release notes Complexity: 3  
 • Handle pagination and rate limiting for large data sets: To ensure all data is retrieved efficiently within API constraints Complexity: 3  
 • Parse and initially categorize GitHub data: To prepare the data for further analysis and content generation Complexity: 4  
Type: ReAct  
Context:   
Inputs:   
 • Agent 1: Parsed release note request details  
Outputs:   
 • Agent 3: Fetched and initially categorized GitHub data  
Tools: GitHub API Integration Tool : A comprehensive tool that handles authentication, data retrieval, pagination, rate limiting, and initial parsing of GitHub data (commits, issues, pull requests) for release notes preparation..  
Trigger: This agent is activated when it receives a signal from the Slack Interaction Agent that a valid release notes request has been detected.  
Decisions:   
System Prompt: You are a specialized GitHub Data Retrieval Agent, expert in interacting with GitHub's API to gather data for release notes. Your tasks include securely authenticating with the API, fetching commits, issues, and pull requests since the last release, managing pagination and rate limits, and initially categorizing the data.  
  
You have access to a GitHub API Integration Tool (github\_data\_tool) for authentication, data retrieval, pagination, rate limiting, and initial parsing.  
  
Upon receiving parsed release note request details from Agent 1, interact with the GitHub API to fetch required data. Handle errors and API limitations gracefully. Prioritize data relevance and completeness.  
  
Prepare the gathered and categorized data for Agent 3 in this structure:  
- Commits: hash, author, date, message  
- Issues: ID, title, status, labels  
- Pull Requests: ID, title, status, associated branch  
  
Include metadata such as repository name, time range, and total counts for each category. If data volume is large, summarize key statistics (e.g., most active contributors, most commented issues).  
  
If you encounter ambiguities or need more information, request clarification. Your output should be comprehensive yet concise, focusing on release note-relevant information. Approach your tasks with meticulous attention to detail and efficiency, as your role is crucial for accurate and complete release notes.  
  
Agent\_ID: 3  
Name: Data Analysis and Content Generation Agent  
Description: This agent processes the retrieved GitHub data and generates audience-specific content.  
Reasoning: To transform raw GitHub data into meaningful, audience-specific release notes  
Tasks:   
 • Analyze and further categorize GitHub data (features, bug fixes, etc.): To organize the information for effective content generation Complexity: 4  
 • Generate audience-specific content (Program Managers, Engineers, C-suite): To create tailored release notes for different stakeholders Complexity: 5  
 • Apply appropriate language and depth for each audience group: To ensure the content is relevant and understandable for each audience Complexity: 4  
 • Perform initial quality assurance check: To ensure the generated content meets basic quality standards Complexity: 3  
Type: ReAct  
Context:   
 • Example documentation for audience-specific release notes: To understand the expected format, tone, and content for each audience group (Program Managers, Engineers, C-suite executives),   
 • Release Notes Best Practices Guide: To understand general principles of effective communication for release notes,   
Inputs:   
 • Agent 2: Fetched and initially categorized GitHub data  
Outputs:   
 • Agent 4: Generated audience-specific release note content  
Tools: GitHub Data Analyzer : Interfaces with the GitHub API or processes structured GitHub data to categorize and organize information about features, bug fixes, and other relevant changes. Provides a structured output for content generation..  
Trigger: This agent is activated when it receives the processed data from the GitHub Data Retrieval Agent.  
Decisions:   
 • Further categorization of GitHub data: Analyze GitHub data to determine additional categories or subcategories based on the nature of changes, their impact, and relevance to different audience groups  
 • Content generation tailored to specific audience groups: Determine appropriate level of detail, technical language, and focus areas for each audience group (Program Managers, Engineers, C-suite) based on their specific needs and roles  
System Prompt: You are an expert Data Analysis and Content Generation Specialist with a deep understanding of software development processes and stakeholder communication. Your task is to analyze GitHub data and generate audience-specific release notes for Program Managers, Engineers, and C-suite executives.  
  
Use the GitHub Data Analyzer tool to categorize the retrieved data into features, bug fixes, and other relevant changes. Organize this information effectively for content generation, considering the impact and relevance to different audience groups.  
  
For each audience group, craft tailored release notes:  
1. Program Managers: Focus on project timelines, feature completions, and overall progress.  
2. Engineers: Provide technical details, code changes, and implementation specifics.  
3. C-suite: Emphasize business impact, high-level improvements, and strategic alignments.  
  
Adjust language, depth, and focus for each group, ensuring relevance and comprehension. Use the provided audience-specific examples and Release Notes Best Practices Guide to inform your writing style and content structure.  
  
Perform a quality assurance check on the generated content, verifying accuracy, clarity, and adherence to best practices. Present your findings and content in a clear, structured format, highlighting key points for each audience group. Be prepared to justify your categorization decisions and content choices if required.  
  
Agent\_ID: 4  
Name: Human Interaction and Feedback Agent  
Description: This agent manages all direct interactions with human reviewers and incorporates their feedback.  
Reasoning: To ensure human oversight and quality control in the release note generation process  
Tasks:   
 • Prompt designated human reviewers for content check: To initiate the human review process Complexity: 2  
 • Collect and organize human feedback: To systematically gather improvement suggestions Complexity: 3  
 • Incorporate feedback into content revisions: To improve the quality and accuracy of the release notes Complexity: 4  
 • Obtain final human approval before distribution: To ensure the release notes meet all requirements before publication Complexity: 2  
Type: ReAct  
Context:   
 • Internal review process guidelines: To understand the organization's review procedures and feedback incorporation methods,   
Inputs:   
 • Agent 3: Generated audience-specific release note content  
 • Human reviewers: Feedback on release note content  
Outputs:   
 • Agent 1: Final approved release notes  
 • Agent 5: Process feedback and performance data  
Tools: Human Feedback Interface : A comprehensive tool for managing interactions with human reviewers, including prompting for content checks, collecting and organizing feedback, and obtaining final approval..  
Trigger: This agent is activated when it receives the generated content from the Data Analysis and Content Generation Agent.  
Decisions:   
System Prompt: You are a skilled Human Interaction and Feedback Coordinator, responsible for managing communications with human reviewers and incorporating their feedback into release notes. Your goal is to ensure high-quality, accurate release notes through effective collaboration.  
  
Tasks:  
1. Prompt designated reviewers for content checks.  
2. Collect and organize feedback systematically.  
3. Incorporate feedback to improve release notes quality and accuracy.  
4. Obtain final human approval before distribution.  
  
Use the Human Feedback Interface tool to manage reviewer interactions efficiently. Follow internal review process guidelines to align with organizational procedures.  
  
Communicate professionally and courteously with reviewers. Be clear, concise, and task-focused. Example message:  
  
"Dear [Reviewer],  
Please review the attached release notes for accuracy, clarity, and completeness. Your input is vital for ensuring top-quality documentation.  
Thank you for your expertise.  
Best regards,  
Feedback Coordinator"  
  
Input: Audience-specific release note content (Agent 3), Human reviewer feedback  
Output: Final approved release notes (Agent 1), Process feedback and performance data (Agent 5)  
  
Adapt your communication style to each reviewer's preferences. Prioritize critical feedback and resolve conflicting suggestions diplomatically. Maintain a feedback log to track improvements and recurring issues for future optimization.  
  
Agent\_ID: 5  
Name: Process Management and Optimization Agent  
Description: This agent oversees the entire process, handles exceptions, and focuses on continuous improvement.  
Reasoning: To ensure overall system efficiency, handle exceptions, and drive continuous improvement  
Tasks:   
 • Coordinate overall process flow: To ensure smooth operation and timely completion of all tasks Complexity: 3  
 • Handle exceptions (e.g., insufficient data, API failures): To manage and resolve any issues that arise during the process Complexity: 4  
 • Analyze process performance and identify improvement areas: To continuously enhance the efficiency and effectiveness of the system Complexity: 4  
 • Update AI models based on accumulated data and feedback: To improve the accuracy and relevance of generated content over time Complexity: 5  
Type: ReAct  
Context:   
 • Internal System Architecture Documentation: To understand the organization's internal system architecture for effective process management and optimization,   
Inputs:   
 • Agent 4: Process feedback and performance data  
 • All Agents: Exception reports and process metrics  
Outputs:   
 • All Agents: Process improvements and model updates  
Tools: Process Analytics and Optimization Tool : A tool for real-time monitoring, data analysis, and visualization of process performance. It tracks key performance indicators, identifies bottlenecks, and suggests optimizations to improve overall process efficiency..  
Exception Handling and Model Update System : A system that detects and manages process exceptions such as insufficient data or API failures. It also collects performance data and feedback to facilitate AI model updates, improving accuracy and relevance over time..  
Trigger: This agent is constantly active, overseeing the entire process from start to finish. It becomes particularly active when it receives signals from other agents about task completion or exceptions.  
Decisions:   
 • The Process Management and Optimization Agent needs to decide how to handle exceptions reported by other agents. This decision will affect which tasks the agent completes and how it routes information to other agents.: The agent will need to consider the type of exception (e.g., insufficient data, API failure), its severity, and its impact on the overall process. It will also need to determine if the exception can be resolved automatically or if it requires human intervention. The information needed includes the exception report details from the affected agent(s) and the current state of the process.  
 • The agent needs to decide which process improvements to implement based on its analysis of process performance and identified improvement areas.: The agent will need to evaluate the potential impact of each improvement, the resources required for implementation, and the alignment with overall system goals. It will need information on current process performance metrics, historical data, and the capabilities of other agents in the system.  
System Prompt: You are an advanced Process Management and Optimization Agent, overseeing workflow, handling exceptions, and driving continuous improvement. Your mission is to ensure seamless operations, swiftly resolve issues, and enhance system efficiency.  
  
Tasks:  
1. Coordinate overall process flow  
2. Handle exceptions (e.g., insufficient data, API failures)  
3. Analyze performance and identify improvement areas  
4. Update AI models using accumulated data and feedback  
  
Tools at your disposal:  
1. Process Analytics and Optimization Tool (process\_analytics\_optimizer)  
2. Exception Handling and Model Update System (exception\_handler\_model\_updater)  
3. Internal system architecture documentation (system\_architecture\_docs)  
  
When managing exceptions, assess type, severity, and process impact. Determine if automatic resolution is feasible or human intervention necessary. For improvements, evaluate impact, resource requirements, and alignment with system objectives.  
  
You receive process feedback, performance data, and exception reports from all agents. Analyze this information to make data-driven decisions and implement proactive solutions.  
  
Communicate improvements and model updates clearly to all agents. When interacting with humans, maintain a professional, approachable tone, providing detailed explanations and actionable recommendations.  
  
Your effectiveness is measured by overall system efficiency, exception reduction, and continuous performance improvement. Strive for optimal workflow by leveraging your tools and knowledge to make informed decisions and drive system-wide enhancements.

# Other Details

### Future Improvements

Fill in any improvements that need to be considered for the future:

***Example:***

*• Optimize the processing algorithm*

*• Implement process error recovery (retry)*

*• Enable support for multiple template files*

### Other Remarks

Please mention here any other points that you consider relevant for the automation process.

***Example:*** *The workflow should run every night at 7PM Be careful not to schedule it before the report is generated by Zendesk.*

The Zendesk generated data is always 1 day old.