BeSir Whitepaper Basic

BeSir Whitepaper: From Declaration to Execution - Completing an Enterprise AI Strategy through the AI Execution Layer

1. Introduction: Why Enterprise AI Falls into the PoC Trap

Most large enterprises declare an “enterprise-wide AI strategy” and promise AI-driven business innovation. However, the results are often disappointing. AI projects are repeated year after year, yet fail to scale into real production environments. As a result, enterprise AI strategies remain as mere “PoC collections” that list pilot initiatives, rather than becoming operational capabilities. AI continues to be confined to auxiliary roles such as document summarization or FAQ responses.

The fundamental cause of this failure is not the insufficient performance of AI models. Today, enterprises already operate in an environment where sufficiently powerful Large Language Models (LLMs) are readily available. The real issue lies not in technology, but in structure.

All existing enterprise systems are designed based on a fundamental assumption: that humans are the users. In these systems, humans read screens, understand context, click buttons, and interpret results. Within such a structure, AI can only exist as an external tool and cannot become a true user of the system.

This whitepaper analyzes the essence of this structural problem and presents a new paradigm—and its solution—for transforming AI into an execution主体 of enterprise systems.

Limitations of Current Approaches

The AI adoption approaches currently taken by enterprises can largely be categorized into three types. However, each approach delivers only partial results and contains clear structural limitations that prevent enterprise-wide expansion.

- **Chatbots / Copilots: “Smart advisors that cannot execute”**

  These approaches improve productivity to some extent through natural language Q&A and document summarization. However, they remain in the role of “advisors” that are disconnected from core business systems. While AI can suggest what should be done, it lacks both the authority and the means to directly access systems and execute tasks. This is similar to assigning a smart intern to write a report, while still requiring a human employee to handle approvals or system operations.

- **BI / Report Automation: “A dashboard, not a driver”**

  These solutions generate dashboards or automate reports using natural language. Just as a car dashboard displays speed and fuel status but cannot drive the vehicle, this type of AI only presents predefined data “results.” It shows *what* has happened, but cannot act as a “driver” that navigates across multiple systems to analyze context, validate hypotheses, and answer *why* something occurred.

- **RPA / Automation: “An automated button that repeats predefined actions”**

  RPA automates repetitive tasks based on predefined scenarios, but it does not understand the “meaning” of work. It merely follows fixed UI paths or scripts. Like an elevator button, it operates perfectly under predefined conditions, but stops easily when screen structures change or exceptions occur. As a result, it cannot be applied to complex tasks that require meaning-based judgment—the true value of AI.

Defining the Core Problem

The failure of all three approaches converges on a single common cause:

“AI is not the execution of the system.”

Because of this structural issue, AI adoption repeatedly falls into the following costly failure pattern:

1. **Model adoption**: A powerful, state-of-the-art AI model is introduced.
2. **PoC success**: Impressive results are demonstrated within limited data and scenarios, and the PoC is deemed successful.
3. **Attempted scaling**: The successful PoC is expanded toward real production environments.

4. **Collision with existing systems**: AI encounters the complexity of real environments—hundreds of tables, complex permission models, and distributed systems—and is unable to directly operate the systems.
5. **Project suspension**: Security, authorization, accountability, and audit issues arise, causing the project to stop at the pilot stage.
6. **New PoC initiation**: Without resolving the root problem, a new PoC begins using another technology.

To break this vicious cycle, the very question we ask must change.

A Shift in the Question

Until now, we have asked questions such as:

“What AI model should we use?” or “Where should we attach AI?”

However, these questions only ensure that AI remains an external add-on to the system.

Instead, we must now ask:

“What kind of structure is required for AI to directly understand and execute existing systems?”

This question is the key that transforms an enterprise AI strategy from “declaration” to “execution.” The answer to this new question marks the beginning of the new paradigm described in the next section.

2. Paradigm Shift: Making AI a “User” through the AI Execution Layer

The conventional approach of “human-centered systems with AI add-ons” permanently confines AI to an auxiliary role outside the system. As long as AI cannot understand the internal structure of systems or the business meaning of data, and as long as authority and responsibility models are not assigned, enterprise-wide adoption remains impossible—regardless of advances in model performance. Therefore, a fundamental paradigm shift is strategically essential, not a series of short-term feature additions, to allow AI to become a true member of the system.

The Concept of “Human-on-the-loop”

Existing systems follow a Human-in-the-loop structure. In this model, humans are always positioned at the center of all workflows, and AI plays a supporting role by providing information requested by humans. Final judgment and execution responsibility rest entirely with humans.

The new paradigm proposed by BeSir is Human-on-the-loop. In this structure, AI becomes the primary execution agent, while humans assume the role of supervisors. AI independently performs data retrieval, analysis, and execution proposals, while humans focus on supervising AI execution, handling exceptions, and making final responsibility decisions. This completes a new collaboration model in which humans are freed from repetitive analytical work and can concentrate on higher-level decision-making.

Definition of the AI Execution Layer

The core technical structure that enables the Human-on-the-loop paradigm is the “AI Execution Layer.” This layer is positioned as an intermediary between existing enterprise systems and AI models. It enables AI to understand systems as if it were a human user and to execute system operations under predefined rules and controls. The AI Execution Layer is responsible for the following four core functions.

Semantic Understanding

Beyond data structures such as tables and columns, Ontology is used to semantically structure business concepts, relationships, and context so that AI can understand the business meaning of data.

Executable Capability Abstraction

Actual business actions—such as queries, calculations, data changes, and approval requests—are defined as standardized, executable units that AI can invoke and execute. This is implemented through the Capability Graph.

Governed Execution

All AI executions are strictly controlled to occur only within predefined rules for authorization, approval procedures, and audit trails. This is an essential requirement for complying with enterprise-wide security and governance policies.

Human–AI Collaboration Interface

A workspace is provided to support the Human-on-the-loop structure, in which AI acts as the execution主体 and humans perform supervisory roles.

Summary of the Paradigm Shift

The differences between the conventional approach and the new approach proposed by BeSir can be clearly compared as shown in the table below.

Category	Conventional Approach	BeSir Approach
AI Position	External tool (Add-on)	System user (User)
System Understanding	None (developer-dependent)	Ontology-based (structural understanding)
Execution	Human-centered (Human-in-the-loop)	AI-centered with human approval (Human-on-the-loop)
Expansion Model	PoC-based (individual projects)	Enterprise standard (structural assets)
Maintenance	Repetitive redevelopment	Reuse of structural assets

BeSir is the standardized platform implementation of this new concept known as the AI Execution Layer. In the next section, we will examine in detail the specific architecture through which BeSir brings this paradigm into reality.

3. BeSir Platform Architecture: A Three-Layer Structure of Design, Execution, and Governance

BeSir is designed to meet the practical requirements faced by enterprise-wide AI strategies, including minimizing changes to existing systems, fully complying with enterprise security policies, and ensuring controlled execution. This architecture is structured into three layers to clearly answer three fundamental questions: “What must AI understand?”, “What can it execute?”, and “How is it governed?”

Reconfirming the One-Sentence Definition of BeSir

The core identity of BeSir can be clearly defined once again as follows:

“BeSir is an Enterprise AI Execution Layer platform that enables AI Agents to understand and execute existing enterprise systems without redevelopment.”

This definition includes the following important premises:

Preservation of existing systems: Existing systems such as ERP, PMS, and legacy systems remain unchanged.

Respect for existing policies: Existing data models, APIs, and permission structures are fully respected.

Staged authorization model: AI follows the sequence of “Read → Analyze → Propose → Approval-based execution,” and sensitive actions such as data modification always require human approval.

Accountability and auditability: Responsibility for execution and the ability to audit actions are structurally guaranteed.

Three-Layer Architecture Analysis

BeSir is composed of three core layers that clearly separate the design of AI execution structures, their use in real business operations, and the governance that controls all processes.

Design Layer: BeSir Studio

Role: A design environment that defines how AI understands systems and the scope within which it is allowed to execute. This is where semantic structures (Ontology) and execution structures (Capability Graph) for AI are created.

Key principle: The most important design principle is the separation of roles between business users and developers. Business experts define the “meaning” of data, while developers are responsible for ensuring that the structure can be executed safely. This enables business users without technical expertise to directly participate in AI design.

Execution Layer: BeSir Browser

Role: The execution interface through which users apply designed AI Agents in their actual work.

Key principle: Rather than a simple chatbot UI, this layer provides a workspace centered on conversation, data joining, and insight generation. Through natural language interactions, users can retrieve large volumes of data distributed across multiple systems in a single step and immediately transform the results into analytical reports or dashboards. Data modifications are always performed based on human approval.

Governance Layer: BeSir Agent Server

Role: An execution gateway that governs all execution requests between AI Agents and existing systems.

Key principle: This layer is responsible for strict enterprise security and governance requirements, including access control (RBAC), execution auditing,

and support for network-segregated environments. Because all AI executions must pass through the Agent Server, unauthorized access or execution is fundamentally blocked, and all activities are recorded in logs.

4. BeSir's Core IP: Designing "Structure," Not "Code"

The fundamental reason most enterprise AI projects fail is that the business "meaning" and "context" of systems are trapped inside implementation code. When personnel change or systems are modified, this knowledge disappears. The code remains, but the rationale behind it is lost. To address this problem, BeSir proposes a core philosophy for the AI era: the most important assets are not implementation code, but "structural assets" that explicitly define what AI understands (Ontology) and what it can execute (CapGraph).

The Concept and Value of Ontology

Ontology is a semantic structure that enables AI to understand an organization's systems as a "conceptual world." This is fundamentally different in purpose and perspective from an ERD (Entity-Relationship Diagram), which merely describes how data is stored. The distinction between the two can be clearly defined as follows.

An ERD answers "How is data stored?" whereas Ontology answers "What does the data mean?"

If an ERD shows a table named `SALES_TBL` and a column `PROD_ID`, Ontology informs AI that `SALES_TBL` represents the concept of "sales" and that `PROD_ID` is related to the concept of "product." Once Ontology is established, AI can structurally interpret the intent of complex questions such as "Analyze the cause of the decline in sales this quarter from a product perspective," and can integrate data scattered across multiple tables into a single "concept," enabling understanding-based reasoning that goes beyond simple question-and-answer interactions.

The Role and Structure of CapGraph

If Ontology is responsible for AI's "understanding," CapGraph (Capability Graph) defines AI's "actions." Real business operations do not follow a simple linear flow such as "A, then B, then C," but branch based on conditions, with subsequent actions determined by the outcomes of previous steps. CapGraph represents these non-linear business flows in a graph structure.

Node: Represents an individual executable unit (Capability) that AI can invoke, such as queries, calculations, or change requests.

Edge: Defines the conditions, flow, and constraints between nodes.

The Synergy of Ontology and CapGraph

Recall the example mentioned in Section 1: “a smart intern given only a menu.” Ontology is like giving the intern the entire kitchen blueprint, the purpose of every ingredient, and the chef’s recipes. CapGraph is like granting the authority and rules required to actually cook. Without these two elements, AI remains a smart intern holding only a menu; when they are combined, AI becomes a capable assistant to a skilled chef.

An AI Agent built on the combination of these two structures operates in a predictable manner as follows.

1. **Interpreting the meaning of the question**: The user’s natural language question is interpreted as business concepts through Ontology.
2. **Determining the relevant data scope**: The range of data associated with the interpreted concepts is identified.
3. **Exploring executable paths**: The optimal execution paths for handling the data are explored within the CapGraph.
4. **Permission-based execution or proposal**: Data retrieval is executed directly, while actions such as data modification are proposed to users for approval, in accordance with organizational authorization and approval rules.

In this way, AI does not act impulsively. Instead, it operates transparently and predictably only within clearly designed “structures.”

The Strategic Value of Structural Assets

Structural assets represented by Ontology and CapGraph are not one-time project deliverables. They are core intellectual property that can be reused and expanded across the entire organization. Building proprietary Ontology and CapGraph is equivalent to creating an exclusive “map” of business logic for AI. This structured knowledge forms a powerful competitive advantage that is difficult to replicate, creating a healthy lock-in effect based on accumulated value rather than technology dependence. As these assets accumulate, an organization’s AI execution capability improves exponentially.

Based on these structurally designed assets, the next section examines in detail how AI performs work reliably and is governed with rigor through concrete execution mechanisms.

5. Trustworthy AI Execution: Traceability, Analysis, and Governance

No matter how intelligently AI analyzes data or makes recommendations, it cannot be adopted in real production environments if its execution process remains a “black box” and cannot be trusted. In enterprise environments, every action taken by AI must be explainable—why a certain decision was made—and every execution process must be traceable and auditable. BeSir structurally addresses these stringent enterprise requirements through a “traceable execution architecture” and “centralized governance.”

A Traceable Execution Architecture Based on T-PTC

The technical core of BeSir Browser is T-PTC (Traceable Programmatic Tool Calling). This structure was designed to address the limitations of conventional LLM tool-calling mechanisms, namely the lack of transparency and difficulty in tracing execution processes. T-PTC clearly separates the role of AI from the role of actual data processing.

Role of AI: Responsible for understanding user intent and determining which tools to call and in what order, performing decision-making and orchestration.

Role of Tools: Responsible for code-based execution that receives explicit input parameters and performs actual data retrieval and computation.

This structure delivers two key business values. First, all AI execution paths become 100% traceable at the code level, making it possible to identify which tool was called, with which parameters, and at what time—thereby ensuring explainability and result reliability. Second, instead of transmitting large volumes of raw data to the LLM, data is processed and summarized within tools, and only the results are passed to the AI. This structurally resolves token limitations and cost issues associated with LLMs. In the POSCO DX PoC, this approach demonstrated scalability and efficiency for large-scale enterprise data processing by reducing input tokens by 98–99% during large-scale analysis.

Natural Language–Driven Dynamic Dashboard Generation

To maximize data accessibility for business users, BeSir Browser provides a natural language–based dynamic dashboard generation capability. Previously, creating dashboards required requests to BI specialists and took several days. With BeSir, users can now request dashboards in natural language and have AI dynamically generate them instantly.

The process works as follows:

1. A user requests, “Show regional sales trends for this quarter together with items that have low inventory turnover.”
2. AI interprets the meanings of concepts such as “sales” and “inventory turnover” through Ontology.
3. The necessary tools for retrieving and processing data are invoked sequentially.
4. Based on the resulting data, a complete dashboard—covering optimal queries, chart types, and layouts—is dynamically generated and presented to the user.

This capability dramatically reduces dependence on BI specialists and serves as a powerful tool for fostering a data-driven decision-making culture in which business users can independently ask questions and find answers based on data.

Enterprise Governance Framework

BeSir Agent Server serves as the central control gateway for AI execution and plays a critical role in enterprise-wide AI governance. All AI execution requests must pass through the Agent Server and are controlled across the following three key areas.

Access Control (RBAC – Role-Based Access Control): Fine-grained, role-based access control is enforced for users, AI Agents, and individual tools. This ensures that AI can access data and execute functions only within permitted boundaries.

Approval-Based Execution: BeSir follows a clear principle: data retrieval (Read) is automated, while data modification (Create/Update/Delete) must always go through an approval process. When AI proposes data changes, the Agent Server automatically generates approval workflows and enforces approval by responsible parties. AI can analyze and propose actions, but it cannot make final decisions on its own.

Audit Logs: Detailed logs are recorded for every AI execution, capturing who performed the action, when it occurred, which tools and parameters were used, and what the results were. These logs provide clear evidence for post-incident audits and accountability, ensuring operational trust.

The next section provides a clear comparative analysis of how these unique structures and execution mechanisms offered by BeSir fundamentally differ from other tools available in the market.

6. Competitive Landscape: Why Internal Tool Builders (Retool) Are Not Enough

BeSir is often compared to internal tool builders such as Retool or Appsmith. At a glance, all three products appear similar in that they leverage data from existing systems to create new operational interfaces. However, the point of departure in problem definition and the goals they pursue are fundamentally different. The purpose of this section is not to determine which product is superior, but to clearly explain how the “layer” of problems each tool aims to solve differs.

Comparative Analysis of Key Differences

The fundamental differences between Retool/Appsmith and BeSir become clear from the following five perspectives.

Category	Retool / Appsmith	BeSir
Primary User	Human	AI Agent
Design Philosophy	Improving UI productivity for human users	Designing structures for AI execution
System Understanding	Implicit (resides in developers’ knowledge)	Explicit (structured assets via Ontology)
Execution Model	UI event–driven (button click → API call)	T-PTC + Agent Server–based (AI reasoning → control → execution)
Enterprise AI Strategy Perspective	Auxiliary function (Add-on)	Strategic core (Core Engine)

The core value of Retool lies in rapidly and efficiently building user interfaces for internal systems that are inconvenient for humans to use. The semantic structure of the system exists implicitly in the developer’s mind and does not remain as a structured asset.

In contrast, the primary user of BeSir is the AI Agent. BeSir's design philosophy focuses on creating structures that allow AI to both understand and execute systems. Through Ontology, system meaning is explicitly captured as an asset, and through T-PTC and the Agent Server, all AI execution is controlled and audited. This positions BeSir not as an auxiliary feature of enterprise AI strategy, but as its core execution engine.

Explanation of Layer Differences

Based on the analysis above, it becomes clear that BeSir is not a replacement for Retool or Appsmith. The two solutions address problems at fundamentally different layers.

Retool / Appsmith: Human UI Layer (user interface layer for humans)

BeSir: AI Execution Layer (execution layer for AI)

While Retool creates "screens" that enable humans to work more conveniently, BeSir creates "structures" that allow AI to work safely. These two layers are not competitive, but complementary, each serving its own role. However, from the perspective of an enterprise AI strategy that aims for AI to become the execution engine of systems, relying solely on a Human UI Layer inevitably encounters fundamental limitations.

The next section demonstrates how these structural differences translate into concrete success stories in real enterprise environments through actual PoC results.

7. Adoption Cases: Validation through Real-World PoCs

BeSir's architecture and execution model are not merely theoretical; they have been validated through PoCs conducted in real enterprise environments. This section examines representative PoC cases carried out across different industries and operational contexts to demonstrate the tangible value BeSir delivers as an execution engine for enterprise AI strategies.

Case 1 : POSCO DX PoC

The POSCO DX PoC was conducted to validate BeSir's scalability and execution efficiency in a large-scale manufacturing enterprise environment. The primary objective of this PoC was to enable AI to directly analyze large volumes of

operational data and convert the results into insights that business users could immediately utilize.

Under conventional approaches, analyzing hundreds of thousands to millions of records required transmitting large volumes of raw data to an LLM, leading to high costs and severe performance constraints. BeSir structurally resolved this issue by applying the T-PTC architecture, processing data at the tool level and delivering only summarized results to the AI.

As a result, input tokens were reduced by 98–99% during large-scale analyses, and analysis time was shortened by 70–90%. This outcome goes beyond simple cost reduction; it demonstrates that large-scale enterprise data analysis can be practically realized using AI. Through the POSCO DX PoC, BeSir validated its capability as a stable AI Execution Layer operating reliably even in enterprise-wide data environments.

Case 2 : KMG

The KMG PoC was conducted in a retail environment operating multiple offline stores. The objective of this PoC was to enable AI to integrate and analyze sales and inventory data distributed across stores and regions, allowing business users to immediately apply the results to decision-making.

Through BeSir Browser, business users were able to ask natural language questions such as “Show yesterday’s sales by store” or “Compare items with low inventory turnover by store.” Based on Ontology and CapGraph, AI combined data scattered across multiple systems in real time and provided analytical results.

As a result, more than 90% of reports that were previously created manually were automated, and both the speed and accuracy of store-level, data-driven decision-making improved significantly. The KMG PoC demonstrates that BeSir goes beyond single-task automation and can transform how data is utilized across an entire organization.

Consolidated Conclusion of the Case Studies

Although the two cases differed significantly in industry, system scale, and data environment, they commonly demonstrated the following fact:

“The problem lies not in the performance of AI models or the volume of data, but in the structure. BeSir’s AI Execution Layer architecture (Ontology +

CapGraph + MCP) is reusable and capable of bridging PoC initiatives to real-world operations."

Based on these validated success cases, the next section presents a concrete roadmap illustrating how enterprises can adopt BeSir in a phased manner and expand it into an enterprise-wide standard.

8. Phased Expansion Strategy: From PoC to Enterprise Standard

A successful enterprise AI strategy does not aim for "perfect enterprise-wide adoption from the very beginning." Instead, it starts by carefully designing an expansion path that avoids failure. Based on the realistic constraints and requirements faced by large enterprises—such as preserving existing systems, complying with security policies, and validating results step by step—BeSir proposes a four-phase expansion scenario that minimizes risk and maximizes the probability of success.

Four-Phase Adoption Scenario

Each phase has a clear objective and success criteria, and the success of each phase forms the foundation for progressing to the next.

Phase 1: Foundation (Building the Base)

- **Objective :** This phase focuses not on technical experimentation for PoCs, but on securing and validating a common "structure" for enterprise AI execution.
- **Key Activities :** Select one or two core systems, automatically generate and review Ontology, define the basic execution scope (CapGraph), and deploy the Agent Server for execution control.
- **Success Criteria :** "Can AI safely understand and query our systems?"

Phase 2: Pilot Agent (Pilot Execution)

- **Objective :** Build AI Agents that solve real business scenarios on top of the established foundation, and validate their feasibility in actual production environments beyond PoC.
- **Key Activities :** Select one or two core business use cases—such as management analysis or inventory analysis—design Agents using BeSir

Studio, and have business users directly test them in BeSir Browser while collecting feedback.

- **Success Criteria :** “Can AI answer real business questions and safely propose data changes?”

Phase 3: Department Expansion (Department-Level Scaling)

- **Objective :** After pilot success, prevent the proliferation of isolated PoCs across departments and formally validate the “reuse” effect of the structural assets built in Phase 1.
- **Key Activities :** Rapidly expand new Agents by reusing the existing common Ontology and adding only department-specific requirements through CapGraph.
- **Success Criteria :** “Are new Agents created quickly, rather than being developed from scratch each time?”

Phase 4: Enterprise Standardization (Enterprise-Wide Standardization)

- **Objective :** Fully transform AI from a one-off project into a sustainable operational infrastructure for the organization.
- **Key Activities :** Manage Ontology and CapGraph as enterprise-standard assets, advance AI execution policies and audit standards, and systematically manage the Agent portfolio through an AI CoE (Center of Excellence).
- **Success Criteria :** “Has AI become a default execution option for the organization, rather than a tool used by a specific department?”

Summary of Changes Before and After Adoption

Through BeSir’s phased adoption approach, the way enterprises utilize AI fundamentally changes as follows.

Item	Before Adoption	After Adoption
AI Usage	Fragmented, department-level PoCs	Enterprise-standard execution
System Integration	Individual point-to-point implementations	Standardized integration based on the Execution Layer

Item	Before Adoption	After Adoption
Development Cost	Repeatedly increasing	Reduced through structural reuse
Operational Risk	High (lack of control, black-box behavior)	Governed and controlled (centralized governance)
Role of AI	Auxiliary tool (Add-on)	**Core execution engine**

Bringing together all discussions so far, the following conclusion presents the future of enterprise AI proposed by BeSir and its core message.

9. Conclusion: The Beginning of AI Execution

This whitepaper began with a single question:

“Why does enterprise AI stop at the PoC stage?”

We confirmed that the root cause is not the performance of AI models or the volume of data, but rather a *human-centered system architecture* that AI cannot directly understand or use. An enterprise AI strategy can never be completed by simply “attaching” AI as an additional feature on top of existing systems.

The only viable solution to this structural problem is to build an **AI Execution Layer**. BeSir is a solution that delivers this new paradigm in the form of a standardized platform. Through Ontology and CapGraph, BeSir provides AI with a structure that enables it to *understand* and *execute* systems, while the Agent Server *governs* and controls every execution. In doing so, BeSir transforms AI from a one-off project into a sustainable infrastructure, from an uncertain experiment into a reliable operation, and from a temporary initiative into a lasting organizational structure.

Final Checklist for an Enterprise AI Strategy

If your organization is currently formulating or restructuring an enterprise AI strategy, you should ask yourself the following questions:

- Does AI understand the *business meaning* of our systems, rather than just their data structures?
- Are all AI execution processes traceable and auditable without exception?
- Does AI operate strictly within the organization’s approval and authorization framework?

- Does every new AI requirement require starting development from scratch?
- As AI adoption increases, does risk become unmanageable, or does AI become a reusable organizational asset?

If you cannot clearly answer “yes” to these questions, there is a high likelihood that your current strategy will fall back into the PoC trap.

Final Message

The true success of an enterprise AI strategy should not be judged by whether AI has been “adopted.”

The only meaningful measure of success is whether *AI is actually doing real work within the organization*.

When AI can access real systems, analyze data, make meaningful recommendations, and execute tasks under approval, the strategy is finally complete.

BeSir is not the end of a complex AI adoption journey.

BeSir is the first key that enables the true beginning of **AI execution**.