Technical ToolKit

- Development Environment
- Technical Tools

Development Environment

Every reseller should establish a local development environment capable of supporting AI application development and deployment. A standard development environment includes:

- **Visual Studio Code (VS Code):** Primary integrated development environment (IDE) for writing and debugging code.
- **Python:** The primary programming language used for AI development and backend services.
- **Git:** Version control for managing source code.
- **Virtual Environments:** Isolated Python environments for managing project dependencies.
- **Environment Variables (.env):** Secure storage of API keys, database credentials, and configuration settings.

Maintaining a consistent development environment helps ensure compatibility across projects and facilitates collaboration with the THiNK engineering team.

Git & GitLab

Git is the version control system used throughout the THiNK ecosystem, while GitLab serves as the central platform for source code management, documentation, issue tracking, and continuous integration.

Partners are expected to use Git and GitLab for:

- Source code version control
- Branch and merge request management
- Project documentation
- Bug and issue tracking
- Continuous Integration/Continuous Deployment (CI/CD)
- Collaboration with mentors and QA teams

Following Git best practices—including meaningful commit messages, feature branching, and regular code reviews—helps maintain code quality and project traceability.

Technical Tools

FastAPI

FastAPI is the recommended backend framework for developing RESTful APIs that power THiNK AI solutions. It is lightweight, high-performance, and well suited for AI applications requiring rapid response times.

FastAPI is commonly used to:

- Build chatbot APIs
- Integrate AI models with client applications
- Expose Retrieval-Augmented Generation (RAG) services
- Connect databases and external systems
- Implement authentication and authorization

Its automatic API documentation and asynchronous capabilities make it an ideal framework for production AI deployments.

LlamaIndex

LlamaIndex is the primary framework used to connect large language models (LLMs) with custom datasets. It enables AI applications to retrieve relevant information from organizational knowledge bases before generating responses.

Typical uses include:

- Document indexing
- Knowledge retrieval
- Semantic search
- Question answering
- Retrieval-Augmented Generation (RAG)
- Enterprise chatbot development

Using LlamaIndex helps improve response accuracy while reducing hallucinations by grounding AI outputs in trusted organizational data.

ChromaDB

ChromaDB is the recommended vector database for storing and retrieving document embeddings used in AI search and RAG applications.

Key capabilities include:

- Storing vector embeddings
- Semantic similarity search
- Fast document retrieval
- Knowledge base management
- Persistent storage for AI applications

By enabling efficient retrieval of relevant information, ChromaDB significantly improves chatbot performance and user experience.

Groq APIs

Groq APIs provide access to high-speed inference for open-source large language models, enabling partners to build responsive AI applications with reduced latency.

Within the THiNK ecosystem, Groq APIs are commonly used for:

- Conversational AI
- Text generation
- Document summarization
- Information extraction
- AI-powered assistants

API credentials are provided to eligible partners during project implementation, and all API usage should comply with THiNK security and usage policies.

Docker

Docker provides a standardized approach to packaging and deploying AI applications across development, testing, and production environments.

Partners should use Docker to:

- Containerize applications
- Package dependencies
- Ensure consistent deployment environments
- Simplify application scaling
- Support cloud and on-premises deployments

Docker Compose may also be used to orchestrate multiple services, including APIs, databases, and vector stores.

PostgreSQL & MongoDB

THiNK solutions support both relational and document-based databases depending on project requirements.

PostgreSQL is recommended for structured application data such as user accounts, transactions, and system records.

MongoDB is suitable for storing flexible, document-oriented data such as chatbot conversations, logs, and metadata.

Database selection should be based on the application's functional and performance requirements.

Swagger

Swagger is used for API documentation and testing throughout the development lifecycle.

FastAPI automatically generates Swagger documentation, enabling developers and clients to:

- Explore API endpoints
- Test API functionality
- Validate request and response formats
- Simplify integration with external systems

Maintaining accurate API documentation is an essential project deliverable for every client implementation.

Documentation Tools

Comprehensive documentation supports collaboration, knowledge transfer, and long-term maintenance of AI solutions.

THiNK recommends:

- **GitLab:** Project documentation, technical notes, issue tracking, and implementation guides.
- **GitBook:** User manuals, deployment guides, architecture documentation, and knowledge sharing.

Project documentation should include installation procedures, configuration instructions, API references, architecture diagrams, and operational guidelines to support future maintenance and scalability.

Communication Channels

Effective communication is critical to successful project delivery. Partners are expected to use approved communication channels to collaborate with clients, mentors, and the THiNK team.

Primary communication tools include:

- **Google Meet:** Project meetings, technical mentoring, training sessions, and client demonstrations.
- **WhatsApp:** Day-to-day coordination, project updates, and quick technical support.
- **Email:** Formal communication, project approvals, and document sharing.
- **GitLab Issues:** Technical discussions, bug tracking, and task management.

Partners should maintain professional communication, provide regular project updates, and ensure timely responses throughout the project lifecycle. Consistent communication strengthens collaboration, improves transparency, and contributes to successful project outcomes.