Final Team Project Team 04

Project: Agentic ML Builder

AAI-590-IN1: Capstone Project

Prof Dr. Zahid Wani

Project Team 04
Mahesh Arcot Krishnamurthy | Ranjeet Das
Course: AAI-590

Introduction

Problem:

Too much time to convert ML Project Idea / Spec → fully functional ML pipeline

- No unified path from requirements to production-ready pipelines.
- **Manual** and **inconsistent ML model exploration**, model setup / MLOps slows delivery.
- **Hinders experimentation, bottlenecks** in onboarding, scalability.

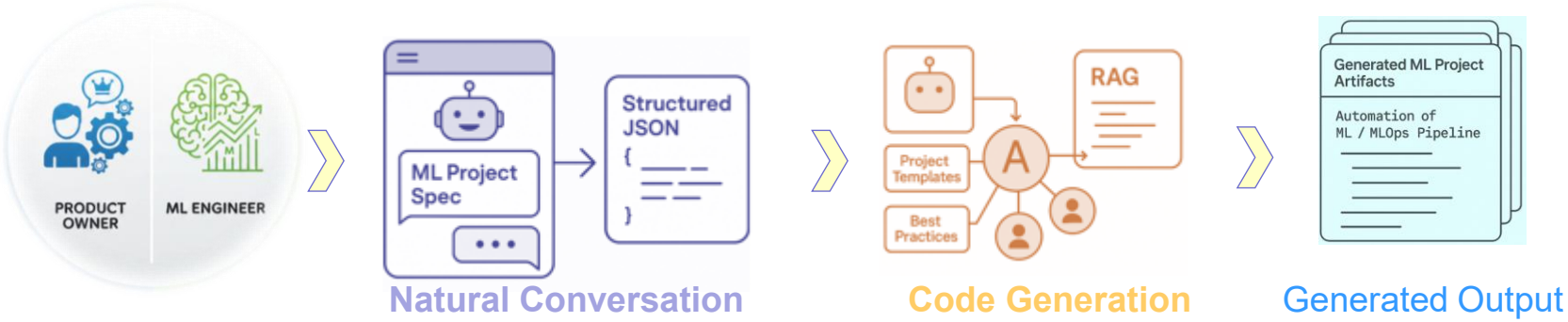
Objective:

Convert **ML Project Specification** involving **weeks of engineering effort** into **fully functional ML Project Pipeline in minutes**, using Agentic AI code generation guided by templates and best practices.

Solution:

Agentic ML Builder

A conversational, **agent-orchestrated ML Builder** that turns high-level requirements into enterprise-grade ML and MLOps pipelines at scale using **Generative Agentic AI** including **templated context** for **code/ document generation** using **RAG**.

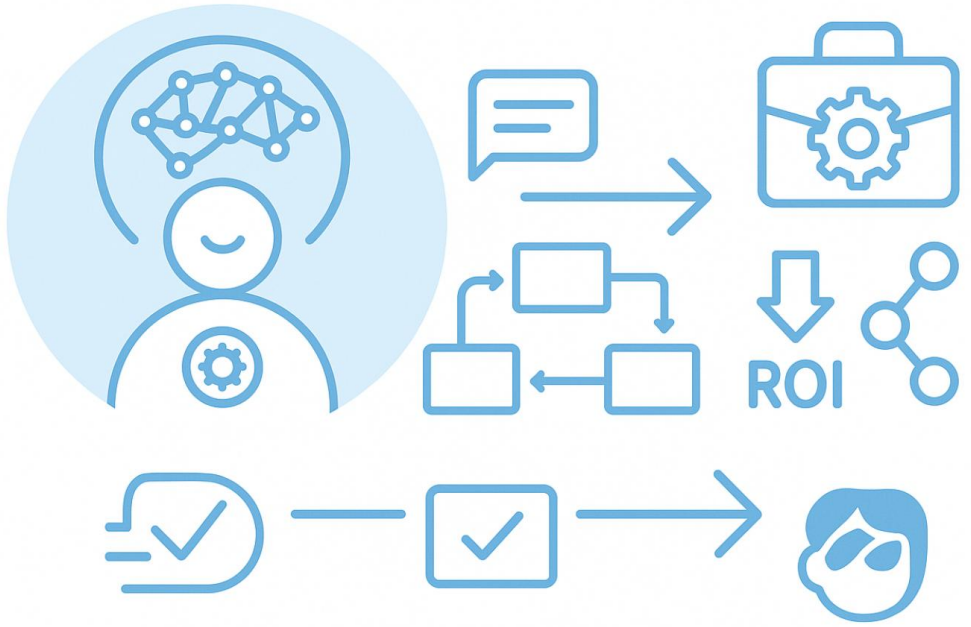


Business Overview

Accelerated ML Engineering
The platform transforms natural-language project descriptions into ready-to-run ML pipelines, reducing lead times from days or weeks to minutes.

Reduced Engineering Overhead
Automated specification, model selection, code generation, and validation dramatically reduce the operational burden on ML engineers.

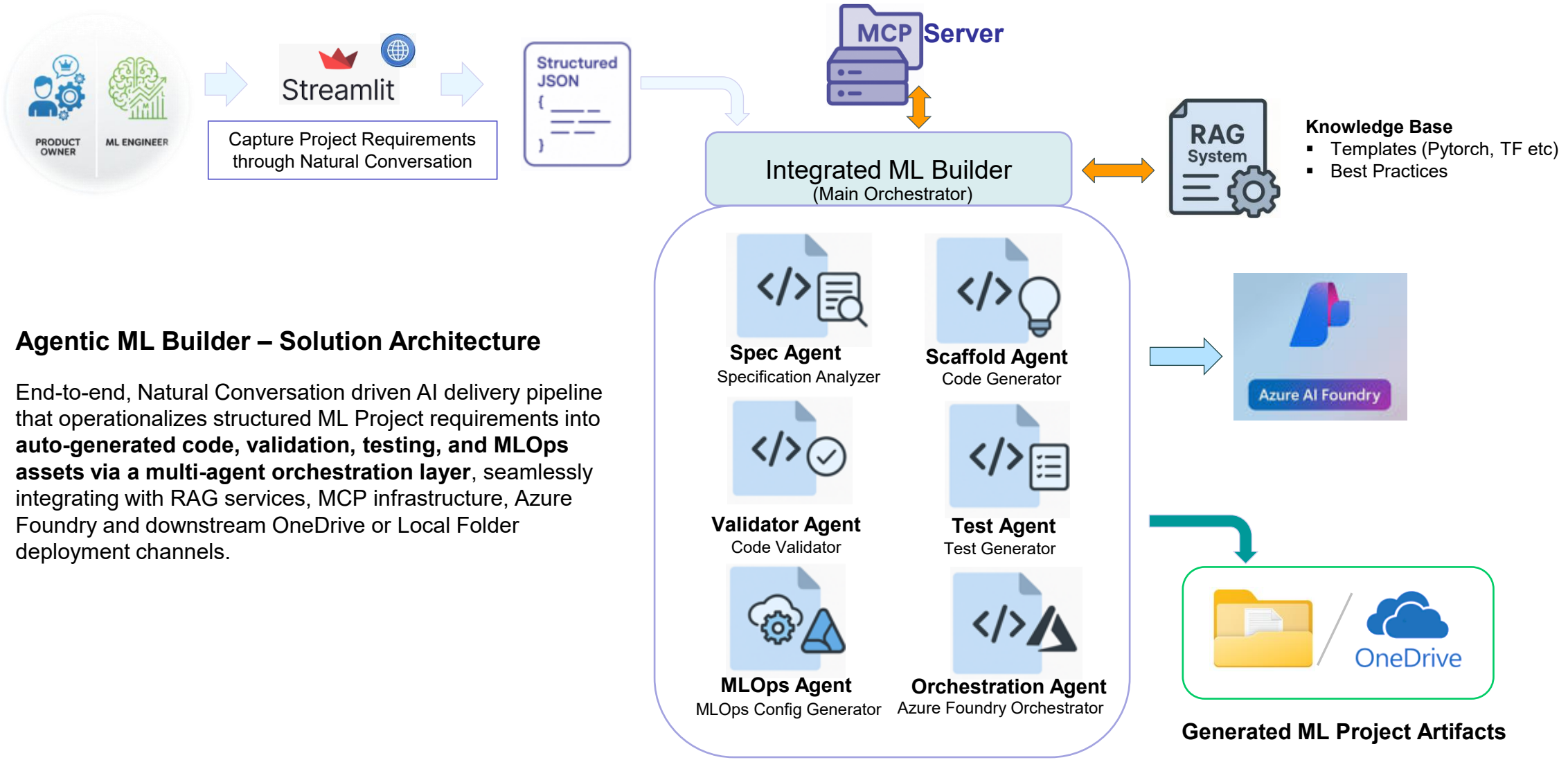
Production-Ready MLOps Integration
Generated scaffolds include tests, CI workflows, and deployment configurations suitable for enterprise environments including Azure ML and GitHub Actions.



ROI

- **Automates repetitive** ML setup tasks, **saving 20 to 40 hours per project**.
- Ensures standardized, reproducible, and validated MLOps structures.
- Bridges skill gaps between data scientists and DevOps engineers.
- Seamlessly integrates with Azure AI Foundry and GitHub Actions.

Solution Architecture



Agentic ML Builder – Solution Architecture

End-to-end, Natural Conversation driven AI delivery pipeline that operationalizes structured ML Project requirements into **auto-generated code, validation, testing, and MLOps assets via a multi-agent orchestration layer**, seamlessly integrating with RAG services, MCP infrastructure, Azure Foundry and downstream OneDrive or Local Folder deployment channels.

Technology Stack

Enterprise Enablement Layer

- Streamlit conversational intake, MCP multi-agent orchestration

Intelligence & Generation

- Spec Agent, Scaffolding Agent, Validator Agent, Test Agent, MLOps Agent, Orchestration Agent
- OpenAI SDK / Azure AI Inference, LangChain, RAG: ChromaDB, FAISS, Sentence Transformers

ML Engineering Runtime

- PyTorch, TorchVision, Scikit-Learn, Pandas, NumPy, TensorBoard
- Azure Blob Storage for artifacts & embeddings

MLOps Automation

- Azure AI Foundry + Azure ML orchestration
- OneDrive export via MSAL / MS Graph SDK

Developer Productivity & CI/CD

- Pytest, Docker, Flake8, Click, Rich, YAML, Jinja2

Code Template Framework

A sophisticated local template repository powers our system, providing battle-tested **Python ML scaffolds that eliminate boilerplate and accelerate development**. Our comprehensive template library covers the full spectrum of machine learning tasks, from classical tabular models to cutting-edge deep learning architectures.

Tabular ML

Scikit-Learn pipelines with preprocessing, feature engineering, and model training for structured data tasks

Vision Models

PyTorch CNN architectures optimized for image classification, object detection, and computer vision workflows

Text Classification

Transformer-based models leveraging pre-trained architectures for NLP tasks and sentiment analysis

Gradient Boosting & More

XGBoost implementations, clustering algorithms, and ensemble methods for advanced modeling scenarios

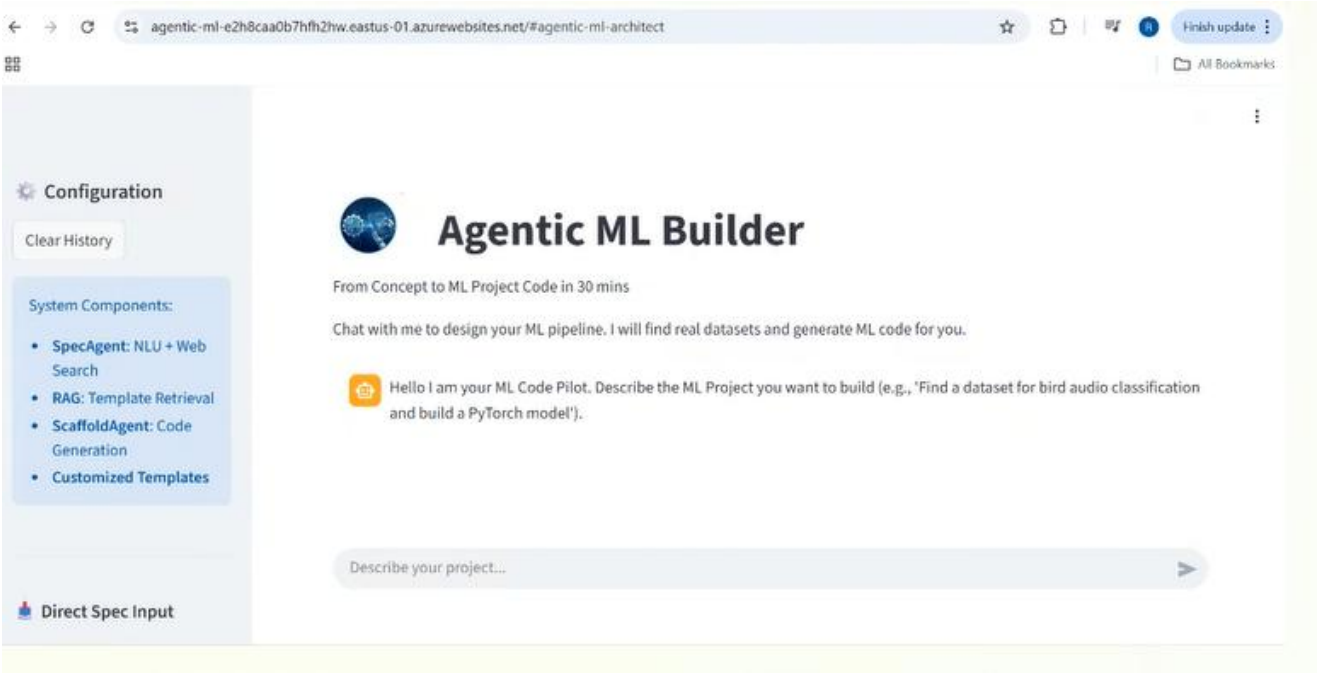
Our **RAG-powered retrieval system intelligently matches user requirements to the optimal template** using sophisticated keyword and task-type matching. This ensures correctness, maintains consistency across projects, and dramatically reduces hallucinations by grounding generation in verified code patterns.

Demo

End-to-End Flow

1. User inputs natural language request
2. ConversationAgent refines missing details
3. SpecAgent searches datasets via API (HF/Kaggle/OpenML)
4. TemplateRetriever identifies correct template
5. ScaffoldAgent generates **full ML Code**
6. ValidatorAgent checks and auto-fixes errors
7. User downloads fully working ML pipeline
8. VS Code / Powershell agent that **generates full project code** from a JSON spec

WebLink:
https://agentic-ml-e2h8caa0b7hfh2hw.eastus-01.azurewebsites.net/



Project Structure

```
agentic-ml-builder/
├── src/
│   ├── streamlit web UI              # Natural Conversation Interface
│   │
│   ├── main.py                       # Entry point
│   ├── integrated_ml_builder.py      # Main orchestrator
│   ├── agents/
│   │   ├── spec_agent.py             # Specification analyzer
│   │   ├── scaffold_agent.py         # Code generator
│   │   ├── test_agent.py             # Test generator
│   │   ├── validator_agent.py        # Code validator
│   │   ├── mlops_agent.py            # MLOps config generator
│   │   └── orchestration_agent.py    # Azure Foundry orchestrator
│   ├── rag/
│   │   ├── rag_system.py             # RAG implementation
│   │   └── templates/
│   │       └── pytorch_templates.py # ML templates
│   ├── mcp/
│   │   └── mcp_server.py             # MCP server
│   └── utils/
│       ├── onedrive_utils.py         # OneDrive integration
│       ├── azure_utils.py            # Azure utilities
│       └── file_utils.py             # File operations
├── deploy/
│   ├── setup_azure.ps1               # Azure setup
│   └── deploy_foundry.ps1            # Foundry deployment
├── tests/
│   └── test_agents.py                # Unit tests
├── input/                            # Input specifications
├── output/                           # Generated projects
├── requirements.txt
├── config.yaml
├── .env                              # Environment variables
└── README.md
```

Execution & Results

VS Code / Powershell - Execution

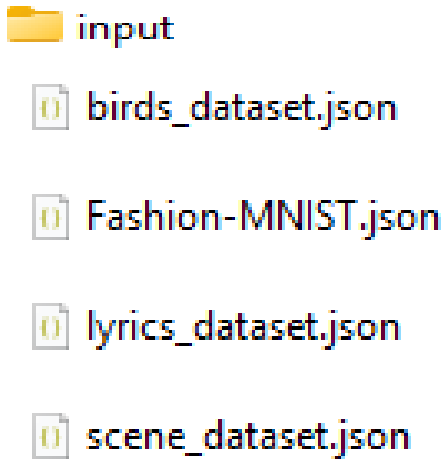
```
(venv) PS D:\03.2025USD\AAI-590\Proj\AgenticMLBuilder\Latest\agentic-ml-builder> powershell.exe -ExecutionPolicy Bypass -File .\run.ps1 --input input/Fashion-MNIST.json --output output

    ============================================================

        AGENTIC ML BUILDER v1.0.0

       AI-Powered ML & MLOps Scaffolding Generator
       Powered by OpenAI GPT-4o, RAG, MCP & Azure AI Foundry

    ============================================================


Configuration:
  [INPUT] Input:    input/Fashion-MNIST.json
  [OUTPUT] Output:   output
  [CONFIG] Mode:     local
  [OK] Validate: False

2025-12-08 18:40:02,485 - __main__ - INFO - Initializing Agentic ML Builder...
Initializing ML Builder...
2025-12-08 18:40:02,486 - int_ml_builder - INFO - Initializing components...
2025-12-08 18:40:02,488 - sentence_transformers.SentenceTransformer - INFO - Use pytorch device_name: cpu
2025-12-08 18:40:02,488 - sentence_transformers.SentenceTransformer - INFO - Load pretrained SentenceTransformer: all-MiniLM-L6-v2
2025-12-08 18:40:08,847 - chromadb.telemetry.product.posthog - INFO - Anonymized telemetry enabled. See                     https://docs.trychroma.com/telemetry for more information.
2025-12-08 18:40:11,478 - rag.rag_system - INFO - Initialized 8 templates
2025-12-08 18:40:11,478 - rag.rag_system - INFO - Created new collection: ml_templates
```

Execution & Results

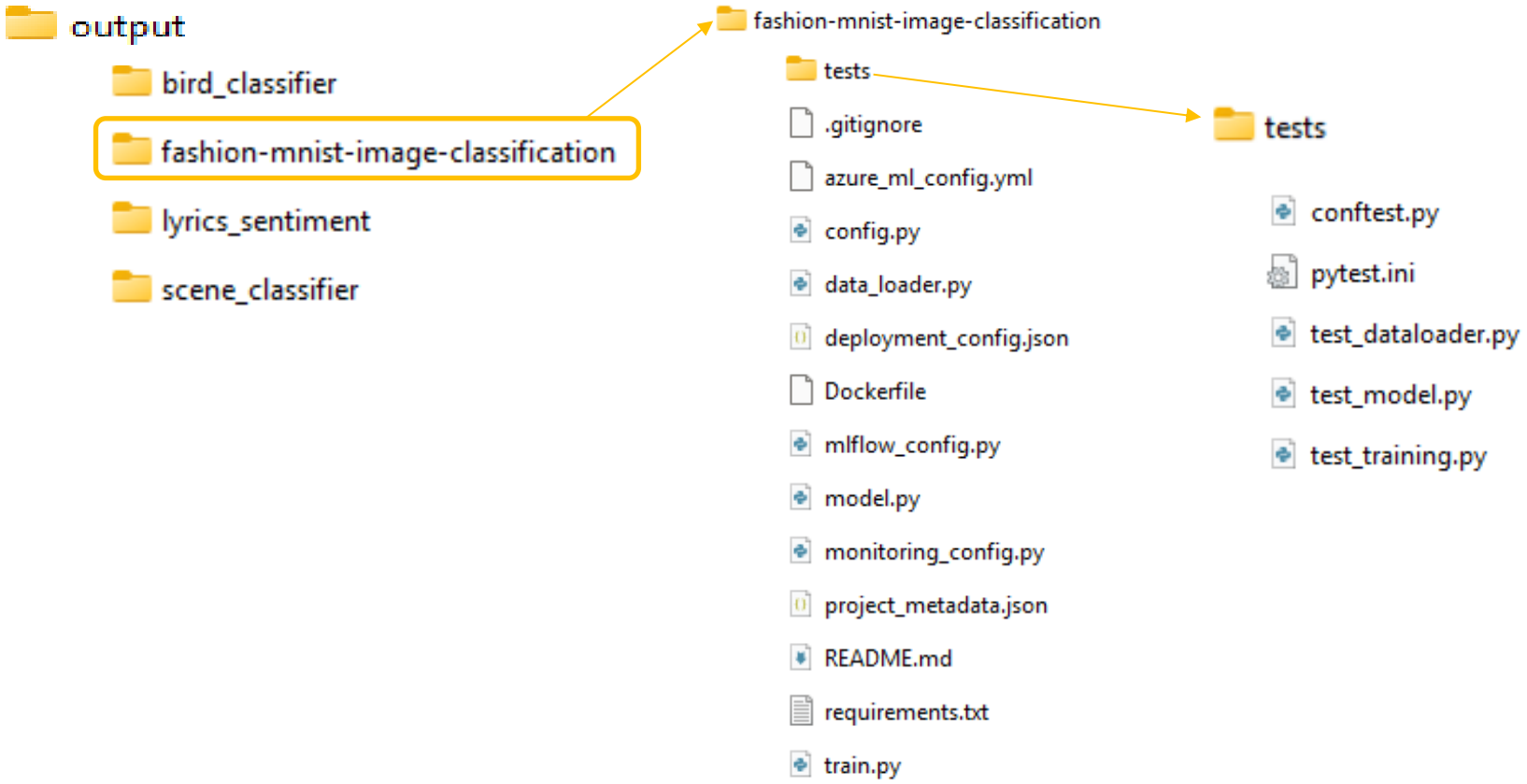
```
Executing ML project generation workflow...

⠋ Generating ML project...2025-12-08 18:40:12,956 - int_ml_builder - INFO - Step 1/7: Reading specification...
2025-12-08 18:40:12,985 - int_ml_builder - INFO - Step 2/7: Analyzing specification...
2025-12-08 18:40:12,986 - agents.spec_agent - INFO - Analyzing project specification...
⠙ Generating ML project...2025-12-08 18:40:18,247 - httpx - INFO - HTTP Request: POST https://api.openai.com/v1/chat/completions "HTTP/1.1 200 OK"
2025-12-08 18:40:18,261 - agents.spec_agent - INFO - Analysis complete: fashion-mnist-image-classification
2025-12-08 18:40:18,262 - int_ml_builder - INFO - Step 3/7: Generating ML scaffold...
2025-12-08 18:40:18,263 - agents.scaffold_agent - INFO - Generating ML scaffold...
⠸ Generating ML project...2025-12-08 18:40:40,346 - httpx - INFO - HTTP Request: POST https://api.openai.com/v1/chat/completions "HTTP/1.1 200 OK"
⠼ Generating ML project...2025-12-08 18:40:50,791 - httpx - INFO - HTTP Request: POST https://api.openai.com/v1/chat/completions "HTTP/1.1 200 OK"
⠴ Generating ML project...2025-12-08 18:41:08,290 - httpx - INFO - HTTP Request: POST https://api.openai.com/v1/chat/completions "HTTP/1.1 200 OK"
2025-12-08 18:41:08,295 - agents.scaffold_agent - INFO - Generated 8 scaffold files
2025-12-08 18:41:08,296 - int_ml_builder - INFO - Step 4/7: Generating tests...
2025-12-08 18:41:08,296 - agents.test_agent - INFO - Generating test suite...
⠦ Generating ML project...2025-12-08 18:41:24,948 - httpx - INFO - HTTP Request: POST https://api.openai.com/v1/chat/completions "HTTP/1.1 200 OK"
⠙ Generating ML project...2025-12-08 18:41:39,790 - httpx - INFO - HTTP Request: POST https://api.openai.com/v1/chat/completions "HTTP/1.1 200 OK"
⠙ Generating ML project...2025-12-08 18:42:04,648 - httpx - INFO - HTTP Request: POST https://api.openai.com/v1/chat/completions "HTTP/1.1 200 OK"
⠹ Generating ML project...2025-12-08 18:42:13,643 - httpx - INFO - HTTP Request: POST https://api.openai.com/v1/chat/completions "HTTP/1.1 200 OK"
2025-12-08 18:42:13,648 - agents.test_agent - INFO - Generated 5 test files
2025-12-08 18:42:13,649 - int_ml_builder - INFO - Step 5/7: Generating MLOps configuration...
2025-12-08 18:42:13,650 - agents.mlops_agent - INFO - Generating MLOps configuration...
2025-12-08 18:42:13,650 - agents.mlops_agent - INFO - Generated 4 MLOps files
2025-12-08 18:42:13,651 - int_ml_builder - INFO - Step 6/7: Writing output files...
⠧ Generating ML project...2025-12-08 18:42:13,718 - int_ml_builder - INFO - Step 7/7: Orchestration validation...
2025-12-08 18:42:13,721 - agents.orchestration_agent - INFO - Validating workflow orchestration...
⠧ Generating ML project...

[OK] ML Project generated successfully!

Project Details:
  [PROJECT] Project Name: fashion-mnist-image-classification
  [FOLDER] Location:     output\fashion-mnist-image-classification

Next Steps:
  1. cd output\fashion-mnist-image-classification
  2. pip install -r requirements.txt
  3. python train.py

(venv) PS D:\03.2025USD\AAI-590\Proj\AgenticMLBuilder\Latest\agentic-ml-builder>
```

Execution & Results

Input Specification (JSON)



Generated Code (Folders & Files)



Deployment

Streamlit UI Layer

Interactive web interface providing intuitive user experience for natural language input and real-time feedback on pipeline generation progress

MLOrchestrator Core

Central coordination engine managing multi-agent workflows, handling state transitions, and orchestrating communication between specialized agents

Template Retrieval System

Local repository with RAG-powered search capabilities for efficient template matching and version-controlled scaffold management

Security Configuration

Environment-based credential management using .env files, ensuring API keys and sensitive data remain protected across deployment environments

OpenAI Integration

Secure API connector with rate limiting, error handling, and fallback mechanisms for robust LLM-powered generation

Conclusion - Transform Ideas into Code

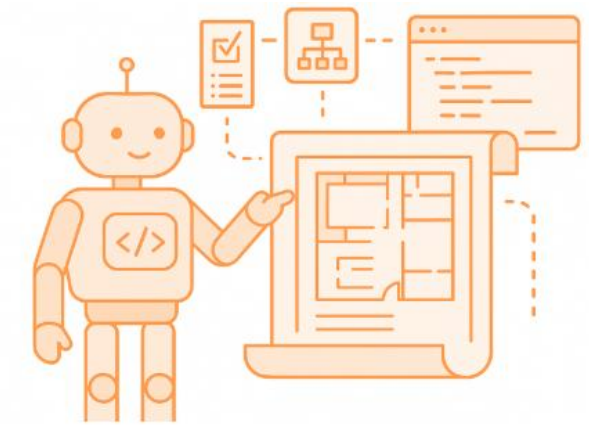
- ✓ **We have successfully demonstrated** We demonstrated an Agentic ML Builder that unifies Agentic AI, Generative AI, and **RAG-based retrieval of templates and best practices** to automate project setup and create intelligent, reproducible workflows.
- ✓ It boosts enterprise productivity by enabling rapid prototyping, consistent AI solution deployment, and seamless integration within the Azure ecosystem.
- ✓ We have successfully run the Agentic ML Builder against **4 Datasets / JSON Input Specifications (i.e. Fashion-MNIST, Scene Classifier, Lyrics-Sentiment and Bird Classifier** and **found high quality Project Code / Documents Generated** that are specific to the type of dataset and Model / Classification method used.

Agentic ML Builder revolutionizes machine learning development by transforming natural language descriptions into production-ready, deployable ML pipelines

Fast	Safe	Explainable	Reproducible
Minutes instead of hours for complete pipeline generation	Built-in ethical safeguards and validation at every step	Transparent code with clear documentation and reasoning	Version-controlled templates ensure consistent results

Future Utilization and Enhancements

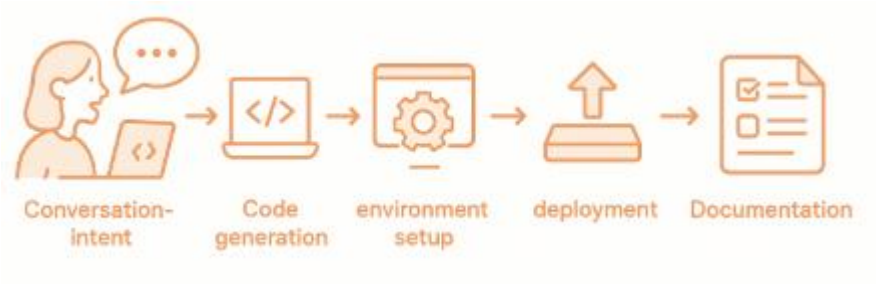
Autonomous Utilization & Enhancements



Multi-Agent Collaboration Ecosystems

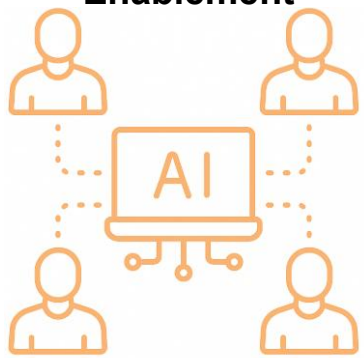
Scenario-Driven Simulation & Impact What-If Analysis

No-Code / Natural Language ML Lifecycle Execution



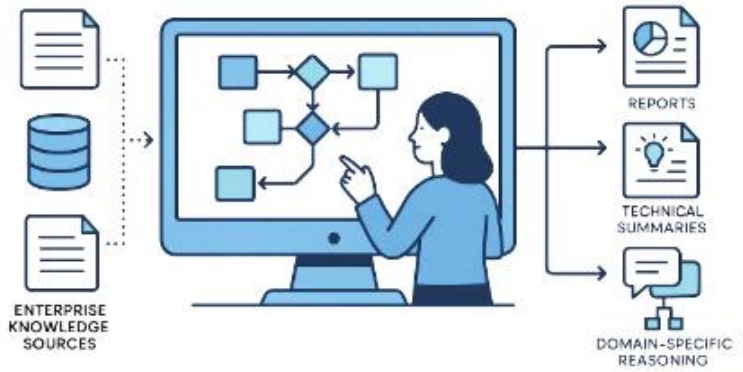
Continuous Compliance & Risk Automation

End-to-End Digital Workforce Enablement



Applications

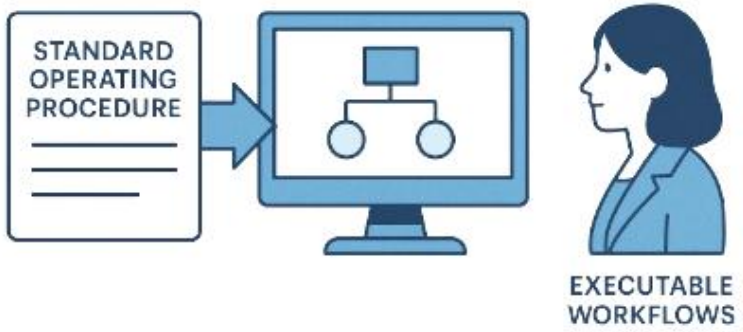
RAG-Enhanced Knowledge Workflow Generator



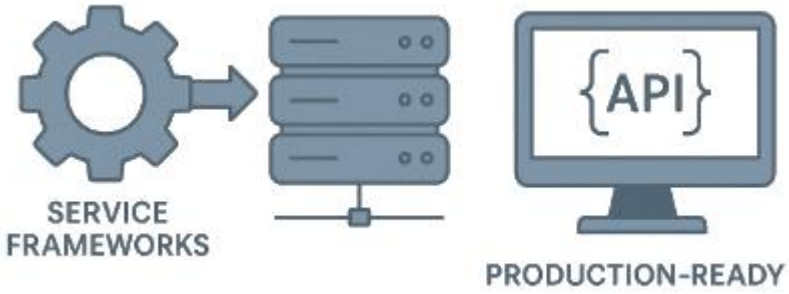
Accelerated Experimentation & Prototyping

Domain-Specific Code Translators & Refactoring Engines

Process Digitalization & SOP Automation



Instant Microservice & API Scaffolding



Automated Test Suite & QA Asset Generation

References

[1] Shi, Y., Wang, M., Cao, Y., Lai, H., Lan, J., Han, X., Wang, Y., Geng, J., Li, Z., Xia, Z., Chen, X., Li, C., Xu, J., Duan, W., & Zhu, Y. (2025). *Aime: Towards fully-autonomous multi-agent framework* (arXiv:2507.11988). arXiv. https://doi.org/10.48550/arXiv.2507.11988

[2] Higuchi, T., Henry, S., & Straight, E. (2025, October 1). *Introducing Microsoft Agent Framework: The open-source engine for agentic AI apps.* Azure AI Foundry Blog. https://devblogs.microsoft.com/foundry/introducing-microsoft-agent-framework-the-open-source-engine-for-agentic-ai-apps/

[3] Microsoft. (2025). agent-framework: A framework for building, orchestrating and deploying AI agents and multi-agent workflows [Computer software]. GitHub. https://github.com/microsoft/agent-framework

[4] Ashrafi, N., Bouktif, S., & Mediani, M. (2025). *Enhancing LLM code generation: A systematic evaluation of multi-agent collaboration and runtime debugging for improved accuracy, reliability, and latency* (arXiv preprint arXiv:2505.02133 v1). arXiv. https://doi.org/10.48550/arXiv.2505.02133

[5] Eken, B., Pallewatta, S., Tran, N. K., Tosun, A., & Babar, M. A. (2024). *A multivocal review of MLOps practices, challenges and open issues* (arXiv preprint arXiv:2406.09737v2). https://arxiv.org/abs/2406.09737

[6] OpenAI. (2024). *A practical guide to building agents* [PDF]. https://cdn.openai.com/business-guides-and-resources/a-practical-guide-to-building-agents.pdf

Thank You

Appendix