Artificial Intelligence Development

---

1. Core Programming Foundation

Used in: Almost every backend, AI, and data system

Skill Level: Beginner → Intermediate

Essential Libraries

- **Python** - Primary language
- **NumPy** - Numerical computing, array operations
- **Pandas** - Data manipulation, analysis, CSV/Excel handling
- **SciPy** - Scientific computing, optimization, signal processing

Common Use Cases

- Data preprocessing and cleaning
- Feature engineering for ML models
- Statistical analysis
- Mathematical operations across pipelines

2. Backend & REST API Frameworks

Used in: Model serving, microservices, production APIs

Skill Level: Intermediate → Advanced

Primary Frameworks

- **FastAPI** - Modern, async, auto-documentation (industry standard)

- **Flask** - Lightweight, flexible, simpler for small projects

When to Use

- Building REST APIs for ML models
- Creating AI-powered backends
- Microservices architecture
- Real-time inference endpoints

3. Core FastAPI Stack

Used in: Professional FastAPI development

Skill Level: Intermediate

Note: These are foundational dependencies for any FastAPI application

Non-Negotiable Components

- **FastAPI** - Main framework
- **Starlette** - ASGI framework (FastAPI's foundation)
- **Pydantic** - Data validation and settings management
- **Uvicorn** - ASGI server for running the application

4. Database & ORM

Used in: Persistent data storage, user management, ML metadata

Skill Level: Intermediate → Advanced

Relational Databases (Industry Standard)

- **SQLAlchemy** - Python SQL toolkit and ORM
- **Alembic** - Database migration tool
- **Psycopg3** - PostgreSQL adapter for Python

Async Database Options

- **Asyncpg** - Fast PostgreSQL driver for asyncio
- **Databases** - Async database support for SQLAlchemy Core

Typical Stack

FastAPI + SQLAlchemy + Alembic + PostgreSQL

5. Authentication & Security

Used in: User authentication, API security, access control

Skill Level: Intermediate → Advanced

Key Libraries

- **python-jose** - JavaScript Object Signing and Encryption
- **PyJWT** - JSON Web Token implementation
- **OAuthLib** - OAuth implementation
- **Passlib** - Password hashing and verification

Common Implementations

- JWT token-based authentication
- OAuth2 flows
- Secure password storage (bcrypt, argon2)
- Role-based access control (RBAC)

6. Background Tasks & Async Jobs

Used in: Email sending, notifications, long-running ML tasks

Skill Level: Intermediate → Advanced

Task Queue Systems

- **Celery** - Distributed task queue (most popular)
- **Redis** - In-memory data store (message broker)
- **RQ (Redis Queue)** - Simpler alternative to Celery

Use Cases

- Asynchronous email delivery
- Scheduled notifications
- Batch prediction jobs
- Model training in background

7. Caching, Rate Limiting & Performance

Used in: API optimization, preventing abuse, scaling

Skill Level: Intermediate

Performance Tools

- **Redis** - Caching layer, session storage
- **FastAPI Cache** - Response caching for FastAPI
- **SlowAPI** - Rate limiting middleware

Benefits

- Reduced database load
- Faster API responses
- Protection against API abuse
- Cost optimization

8. HTTP Clients & External Services

Used in: Calling external APIs, webhooks, service integration

Skill Level: Beginner → Intermediate

Client Libraries

- **HTTPX** - Modern async HTTP client (recommended)
- **Requests** - Simple HTTP library (sync only)

Recommendation: Use HTTPX for new projects (async support + sync compatibility)

9. Configuration & Environment Management

Used in: Managing secrets, environment variables, settings

Skill Level: Beginner → Intermediate

Configuration Tools

- **python-dotenv** - Load environment variables from `.env` files
- **Pydantic Settings** - Type-safe configuration management

Best Practices

- Never commit secrets to version control
- Use environment-specific configurations
- Validate configuration at startup

10. Testing & Quality Assurance

Used in: Ensuring code reliability, CI/CD pipelines

Skill Level: Intermediate → Advanced

Testing Framework

- **Pytest** - Feature-rich testing framework
- **pytest-asyncio** - Async test support
- **FastAPI TestClient** - Built-in API testing
- **Coverage.py** - Code coverage measurement

Testing Types

- Unit tests for individual functions
- Integration tests for API endpoints
- End-to-end testing for complete workflows

11. Logging, Monitoring & Error Tracking

Used in: Production debugging, observability, alerting

Skill Level: Intermediate → Advanced

Observability Stack

- **Loguru** - Simplified logging with better formatting
- **Sentry** - Error tracking and performance monitoring

Production Requirements

- Structured logging
- Real-time error alerts
- Performance metrics tracking
- Request tracing

12. Machine Learning

Used in: Tabular data, production ML models, baseline models

Skill Level: Intermediate → Advanced

Core ML Libraries

- **Scikit-learn** - General-purpose ML algorithms
- **XGBoost** - Gradient boosting (industry favorite)
- **LightGBM** - Fast gradient boosting by Microsoft
- **CatBoost** - Gradient boosting for categorical features

Common Applications

- Credit scoring and fraud detection
- Pricing models and demand forecasting
- Customer churn prediction
- A/B test analysis

13. Deep Learning

Used in: Neural networks, computer vision, NLP, generative AI

Skill Level: Advanced

Deep Learning Frameworks

- **PyTorch** - Research and production (most popular)
- **TensorFlow** - Google's framework, mature ecosystem
- **Keras** - High-level API (now integrated with TensorFlow)

When to Use Deep Learning

- Image and video processing
- Natural language understanding
- Speech recognition
- Recommendation systems
- Time series forecasting (advanced)

14. Generative AI & Large Language Models

Used in: Chatbots, copilots, content generation, RAG

Skill Level: Intermediate → Advanced

LLM Development Tools

- **LangChain** - LLM application framework
- **OpenAI SDK** - GPT-4, GPT-3.5, DALL-E, Whisper
- **LlamaIndex** - Data indexing for LLMs
- **Anthropic SDK** - Claude AI integration

Key Capabilities

- Building conversational AI applications
- Retrieval-Augmented Generation (RAG)
- Function/tool calling for actions
- Prompt engineering and optimization

15. Agentic AI & Workflows

Used in: Autonomous agents, multi-step reasoning, enterprise copilots

Skill Level: Advanced

Agent Frameworks

- **LangGraph** - State machines for agents (LangChain)
- **Semantic Kernel** - Microsoft's agentic framework
- **AutoGen** - Multi-agent conversation framework
- **CrewAI** - Role-based agent orchestration

Use Cases

- Autonomous research assistants
- Multi-step workflow automation
- Tool-using AI agents
- Complex decision-making systems

16. Natural Language Processing

Used in: Text analysis, search, language detection, entity extraction

Skill Level: Intermediate → Advanced

NLP Libraries

- **Hugging Face Transformers -** Pre-trained models (BERT, GPT, etc.)
- **spaCy -** Industrial-strength NLP
- **NLTK** - Educational and research NLP
- **fastText** - Efficient text classification (Facebook)

Common NLP Tasks

- Text classification and sentiment analysis
- Named Entity Recognition (NER)
- Search relevance and ranking
- Language detection and translation

17. Computer Vision

Used in: Image processing, object detection, video analysis

Skill Level: Intermediate → Advanced

CV Libraries

- **OpenCV** - Traditional computer vision
- **YOLO (Ultralytics) -** Real-time object detection
- **MediaPipe** - ML solutions for vision tasks (Google)
- **Pillow (PIL) -** Image manipulation

Applications

- Face detection and recognition
- Object tracking in video
- Quality control in manufacturing
- Medical image analysis
- Mobile and edge deployment

18. Vector Databases & Semantic Search

Used in: RAG systems, semantic search, recommendation engines

Skill Level: Intermediate → Advanced

Vector Database Options

- **FAISS** - Facebook's similarity search (local/fast)
- **Pinecone** - Managed vector database (cloud)
- **Weaviate** - Open-source vector search engine
- **Chroma** - Embedding database for LLM apps
- **Qdrant** - High-performance vector search

Use Cases

- Semantic search across documents
- Retrieval-Augmented Generation (RAG)
- Long-term memory for AI agents
- Similarity-based recommendations

19. Model Serving & Inference

Used in: Scalable model deployment, low-latency serving

Skill Level: Advanced

Inference Frameworks

- **TorchServe** - PyTorch model serving
- **TensorFlow Serving** - TensorFlow model serving
- **ONNX Runtime** - Cross-framework inference optimization
- **Triton Inference Server** - NVIDIA's multi-framework server

Benefits

- Optimized inference performance
- GPU acceleration
- Batch processing
- A/B testing of model versions

20. MLOps & Experiment Tracking

Used in: Model versioning, reproducibility, experiment management

Skill Level: Advanced

MLOps Platforms

- **MLflow** - End-to-end ML lifecycle management
- **Weights & Biases (W&B)** - Experiment tracking and collaboration
- **DVC (Data Version Control)** - Git for data and models
- **Neptune.ai** - Metadata store for ML experiments

Core MLOps Practices

- Track all experiments systematically
- Version datasets and models
- Reproduce training runs
- Monitor model performance over time

21. Containers, Cloud & DevOps

Used in: Deployment, scaling, CI/CD, infrastructure

Skill Level: Advanced

Infrastructure Tools

- **Docker** - Containerization
- **Kubernetes** - Container orchestration
- **GitHub Actions** - CI/CD automation
- **Azure DevOps** - Microsoft's DevOps platform

Cloud Platforms

- **AWS** - SageMaker, Lambda, EC2, S3
- **Azure** - Azure ML, Functions, VMs
- **GCP** - Vertex AI, Cloud Run, GCE

Deployment Patterns

- Container-based deployment
- Serverless inference
- Managed ML platforms
- CI/CD pipelines for ML

Learning Path Recommendations

Beginner Path (0-6 months)

1. Master Python, NumPy, Pandas
2. Learn FastAPI basics
3. Understand SQL and SQLAlchemy
4. Build simple REST APIs
5. Learn basic ML with Scikit-learn

Intermediate Path (6-18 months)

1. Deep dive into FastAPI + async
2. Implement authentication and security
3. Master Docker and basic cloud deployment
4. Learn PyTorch/TensorFlow fundamentals
5. Build end-to-end ML projects

Advanced Path (18+ months)

1. Implement production MLOps practices
2. Build RAG systems with LangChain
3. Deploy scalable inference services
4. Create agentic AI applications
5. Architect complete AI systems

Essential Tool Combinations

Starter API Stack

FastAPI + SQLAlchemy + PostgreSQL + Docker

Production ML Stack

PyTorch + FastAPI + MLflow + Docker + Kubernetes

Modern AI App Stack

LangChain + OpenAI + FAISS + FastAPI + Redis

Enterprise MLOps Stack

MLflow + DVC + Kubernetes + Prometheus + Grafana

Additional Resources

Package Management

- **pip** - Python package installer
- **Poetry** - Dependency management and packaging
- **Conda** - Package and environment management

Code Quality

- **Black** - Code formatter
- **Ruff** - Fast Python linter
- **mypy** - Static type checker
- **pre-commit** - Git hooks for code quality

Documentation

- **Sphinx** - Documentation generator
- **MkDocs** - Project documentation with Markdown
- **Swagger/OpenAPI** - API documentation (built into FastAPI)