Week 00

Building Production Ready ML Systems

A comprehensive 10-week journey to set the ground for transformation of you from a model builder to a E2E ML engineer

About Your Instructors

Isuru Alagiyawanna

Lead Instructor

7+ years experience in AI research and Academics

4+ years experience in industry applications

Welcome to Day 01: Course Introduction

This course bridges the gap between academic ML knowledge and industry-ready ML engineering skills

Today you'll learn:

- The structure of our 10-week journey
- Required tools, technologies, and resources
- Project expectations and support channels

Today's Agenda

Foundation (30 min)

- Course introduction
- Module-by-module summary
- Required software

Implementation (20 min)

- Tools & libraries
- Mini projects overview
- Capstone project intro

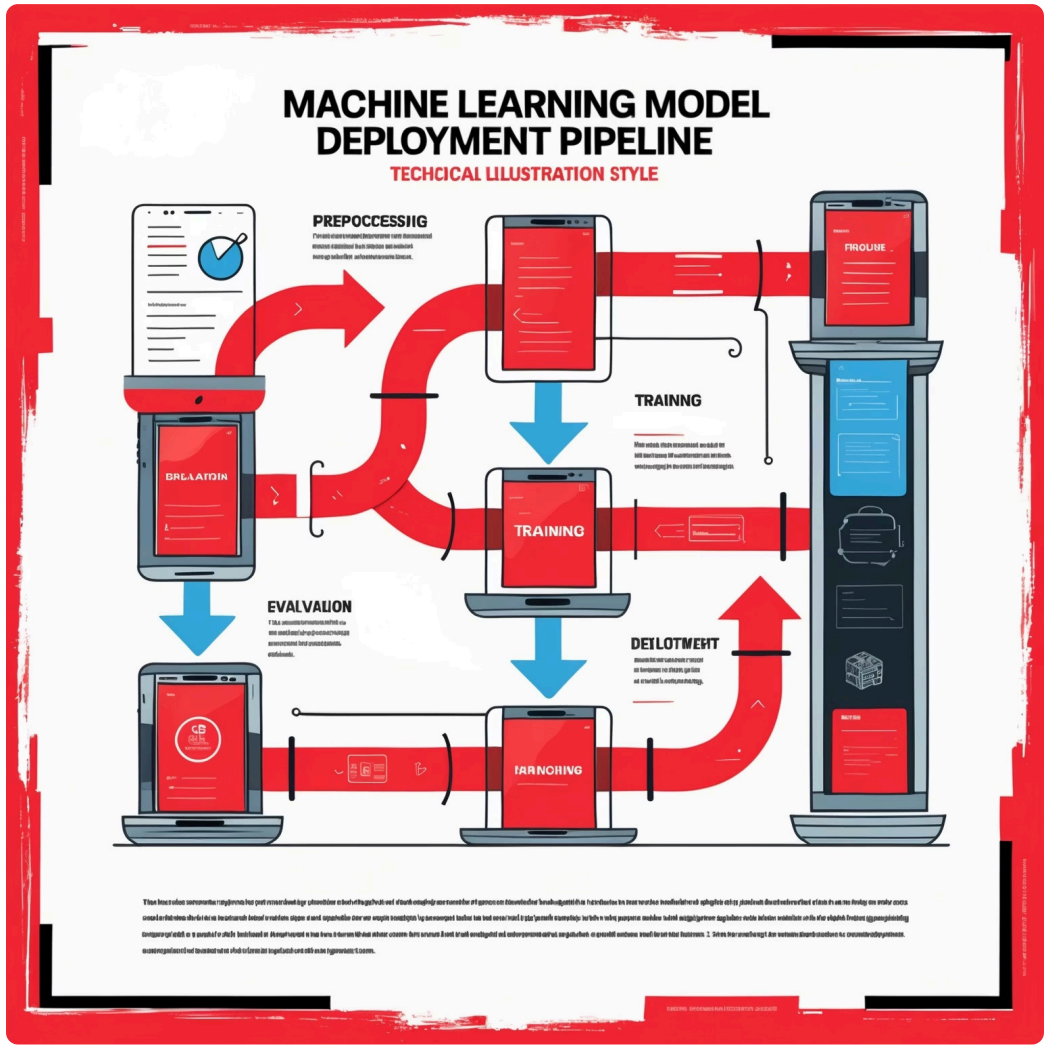
Operations (10 min)

- Hackathon details
- Assessment criteria
- Communication channels

Why "Production-Ready" Matters

In industry, ML excellence requires more than accurate models:

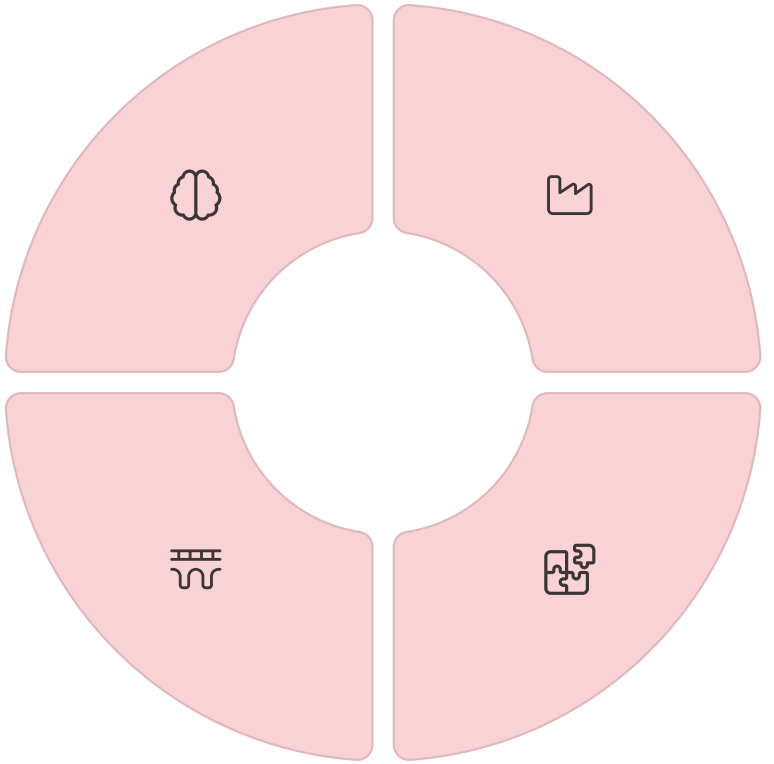
- Deployable in production environments
- Scalable to handle real-world data volumes
- Maintainable through infrastructure changes
- Monitorable for performance degradation
- Reproducible for debugging and auditing



The ML Production Gap

Academic ML

Focus on algorithm understanding and model accuracy on static datasets



Industry ML

Emphasis on reliable systems that handle changing data and scale with demand

Our Approach

Practical projects using industry tools to build and deploy end-to-end ML systems

The Gap

Skills in containerization, orchestration, monitoring, and scalable processing

Zuu Crew's Teaching Philosophy

Industry-Aligned Learning

- Learn by building: hands-on mini-projects
- Real-world tools: Kafka, Spark, Airflow, MLflow
- Practical workflows: CI/CD, monitoring, testing
- Failure resilience: error handling, fallbacks

Your Learning Journey: 10-Week Roadmap

1

Weeks 1-3: Foundations

EDA, base ML pipeline, experiment tracking with ZenML and MLflow

2

Weeks 4-6: Scalability

Data processing with Spark, workflow orchestration with Airflow, batch pipelines

3

Weeks 7-8: Real-time Systems

Streaming with Kafka, containerized deployment on AWS

4

Weeks 9-10: Capstone

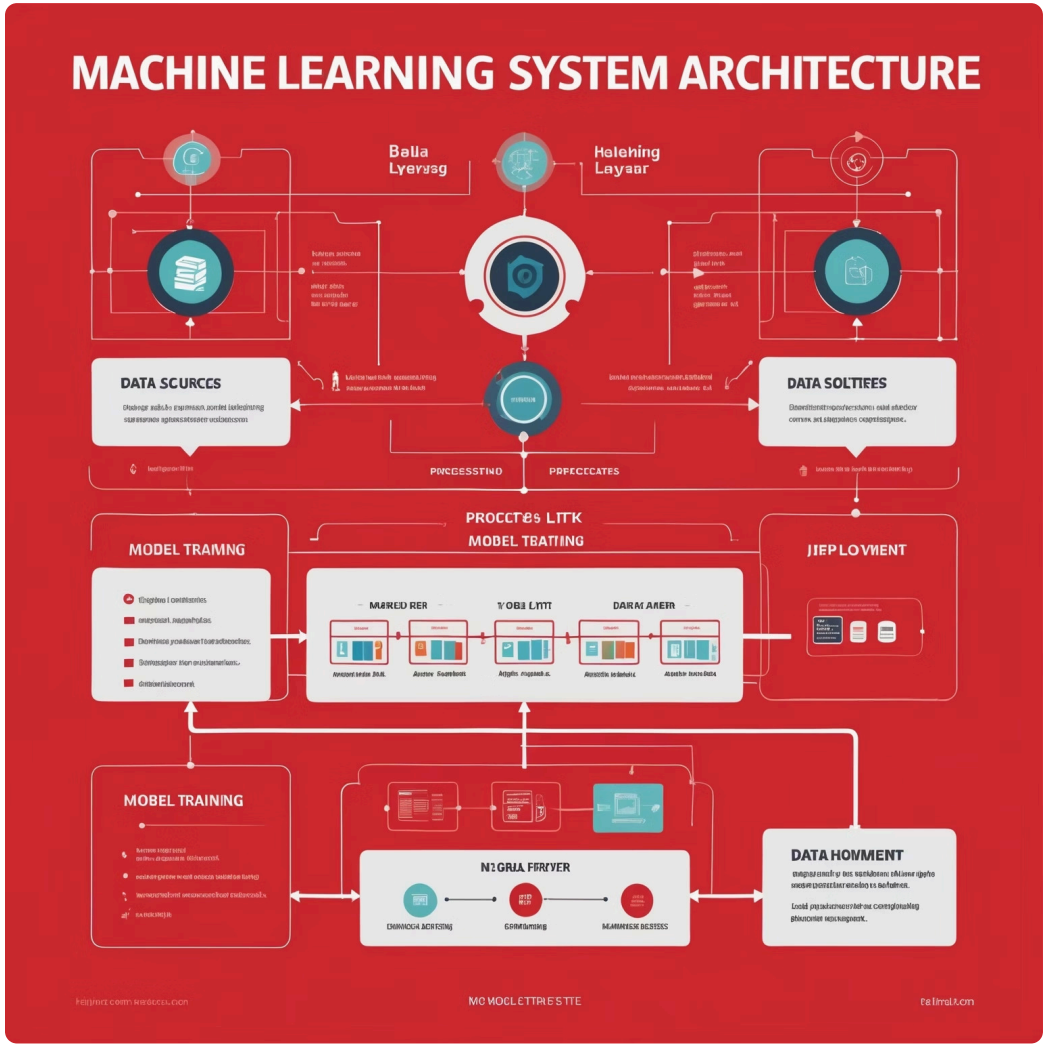
End-to-end streaming prediction system with REST API deployment

Week 1: Capstone Architecture + EDA

Learning Objectives:

- Understand end-to-end ML system architecture
- Plan component interactions and data flow
- Perform exploratory data analysis on the capstone dataset
- Define feature engineering strategies

Deliverables: EDA notebook, architecture diagram, feature engineering plan

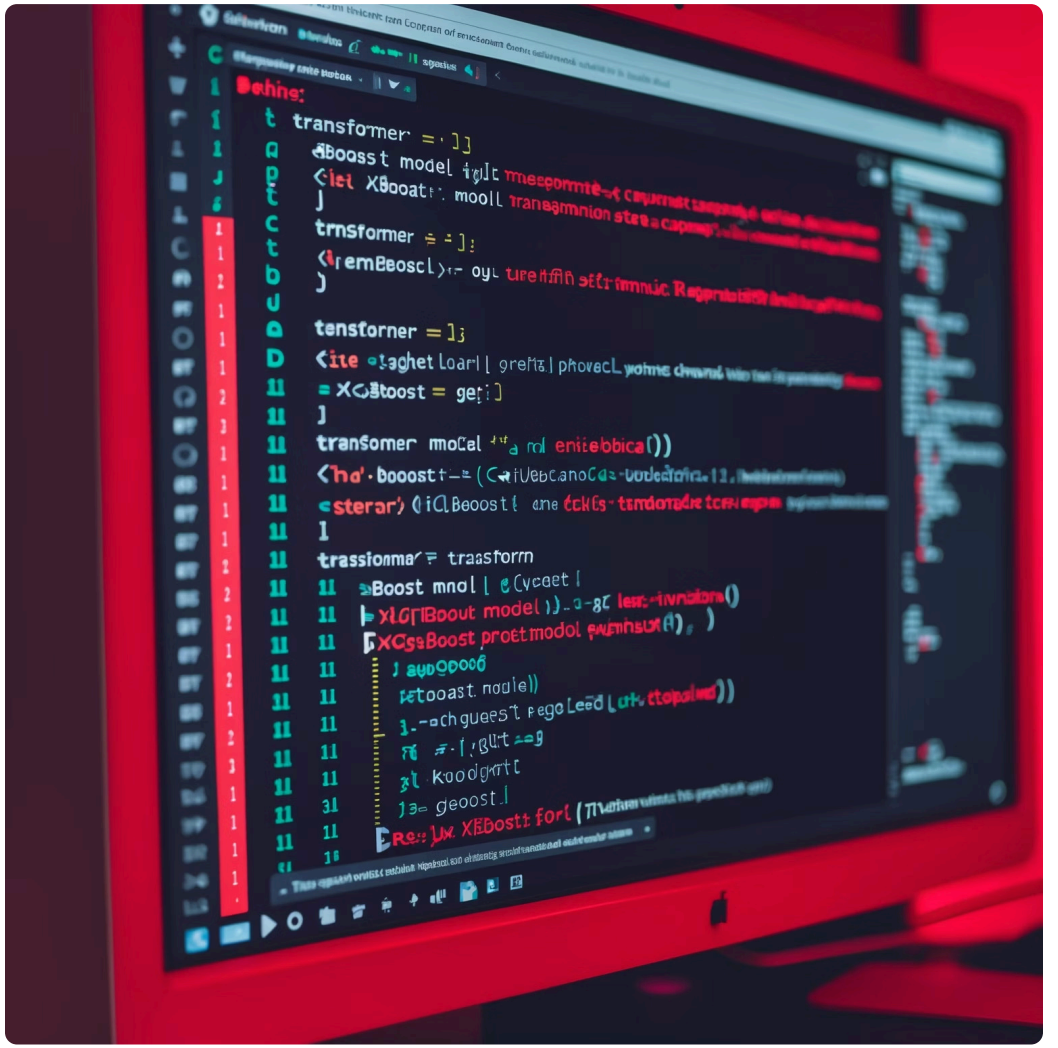


Week 2: Base ML Pipeline

Learning Objectives:

- Build reusable scikit-learn pipelines
- Implement feature engineering transformers
- Train and optimize XGBoost models
- Create modular, testable ML code

Deliverables: Working ML pipeline with metrics reporting



Week 3–4: ZenML + MLflow Integration

Learning Objectives:

- Track experiments systematically
- Version models and datasets
- Compare model performance
- Create reproducible ML pipelines

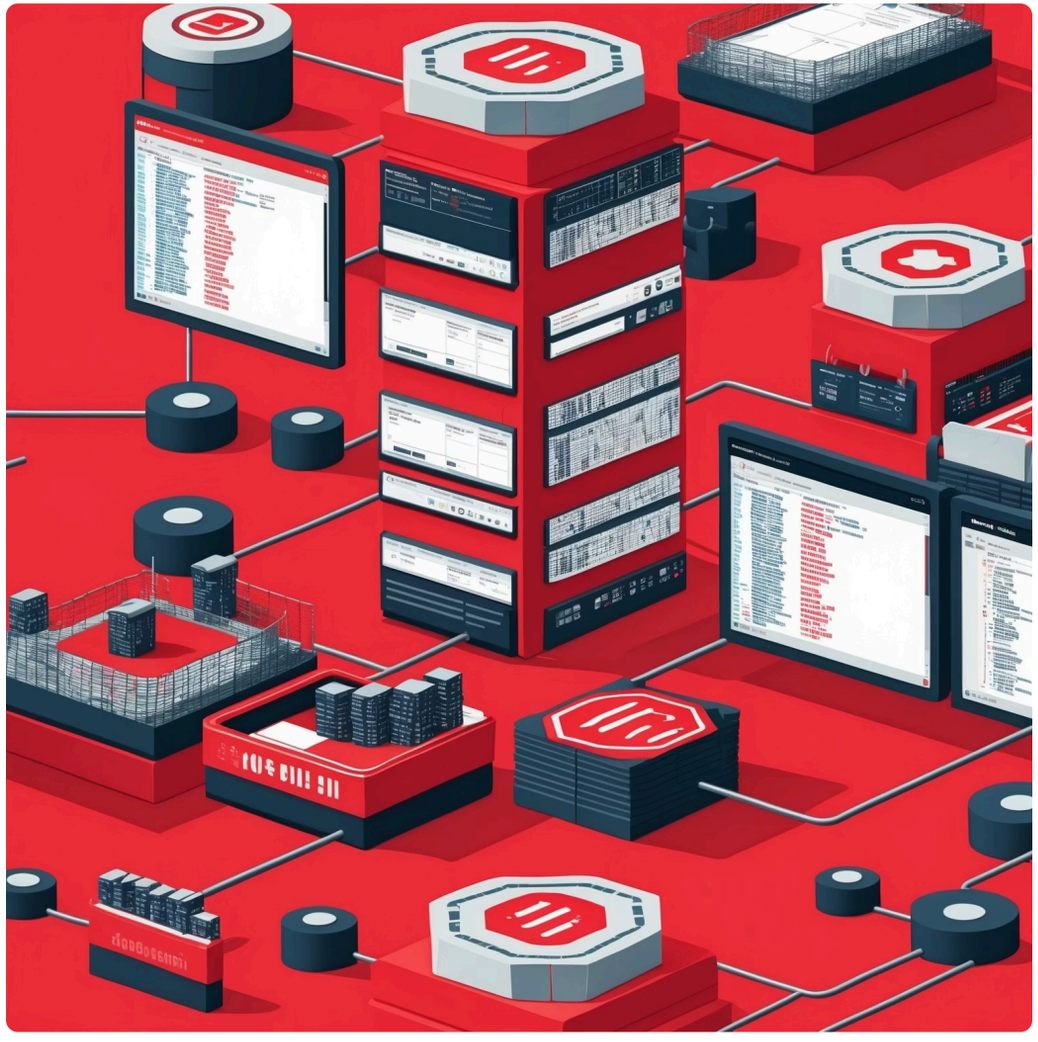
Deliverables: Mini-project 1 - Tracked experiments and registered models

Week 5: Scalable Processing with Apache Spark

Learning Objectives:

- Process large datasets with PySpark
- Implement distributed ML algorithms
- Optimize Spark jobs for performance
- Monitor resource utilization

Deliverables: PySpark data processing pipeline



Week 6: Orchestration using Apache Airflow

Learning Objectives:

- Design and implement DAGs
- Schedule recurring ML workflows
- Handle dependencies between tasks
- Monitor pipeline execution

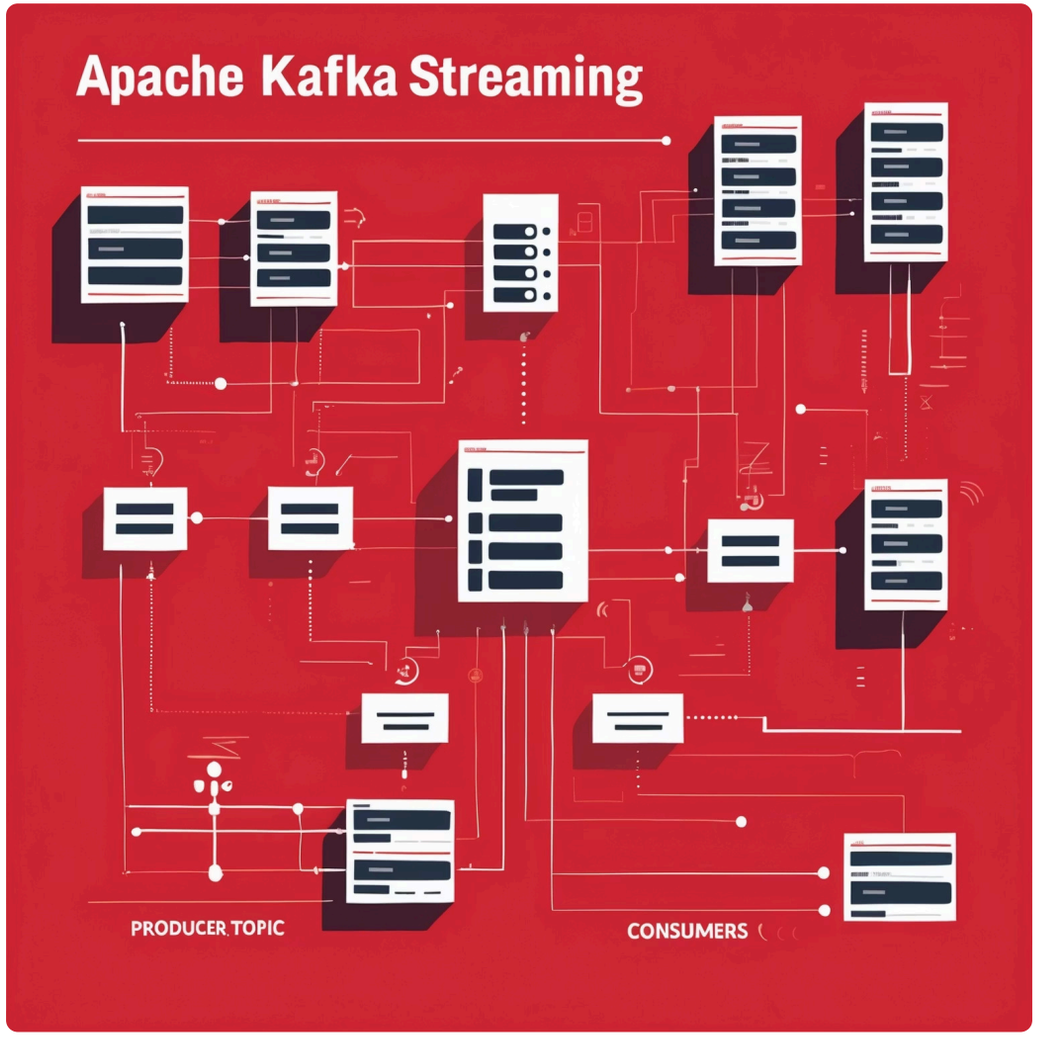
Deliverables: Airflow DAGs for ML pipeline orchestration

Week 7: Real-time Pipelines with Apache Kafka

Learning Objectives:

- Implement Kafka producers and consumers
- Process streaming data
- Design fault-tolerant pipelines
- Handle real-time prediction requests

Deliverables: Kafka streaming pipeline with real-time predictions



Week 8: Containerized Deployment on AWS

Learning Objectives:

- Containerize ML applications with Docker
- Configure Docker Compose for multi-service deployment
- Deploy to AWS infrastructure
- Implement monitoring and logging

Deliverables: Containerized ML system deployed on AWS

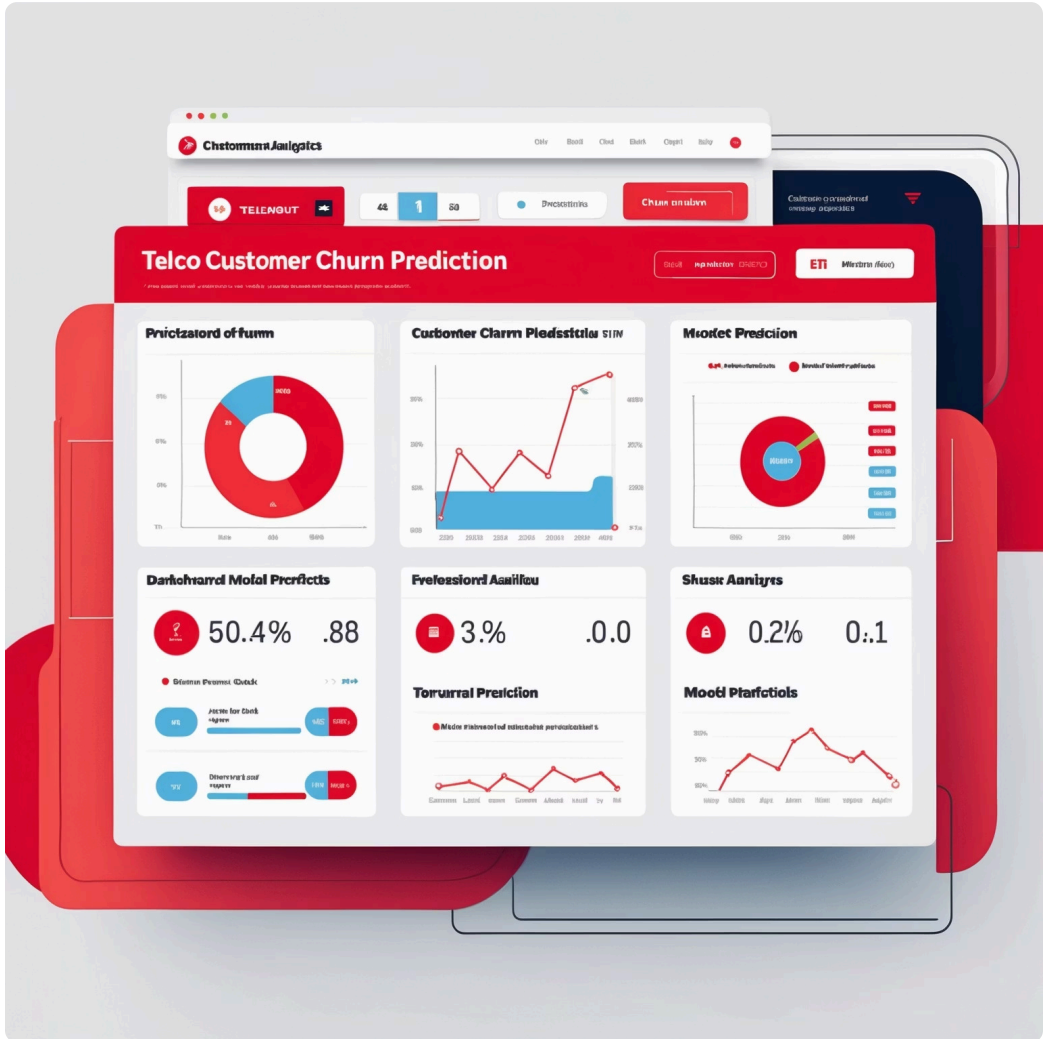
Week 9: Mini-Project 2 - Batch Pipeline

Project Scope:

Build a complete batch ML pipeline for Telco Customer Churn prediction

- Process data with Spark
- Orchestrate with Airflow
- Track with MLflow
- Automate retraining

Deliverables: Working end-to-end batch pipeline with documentation



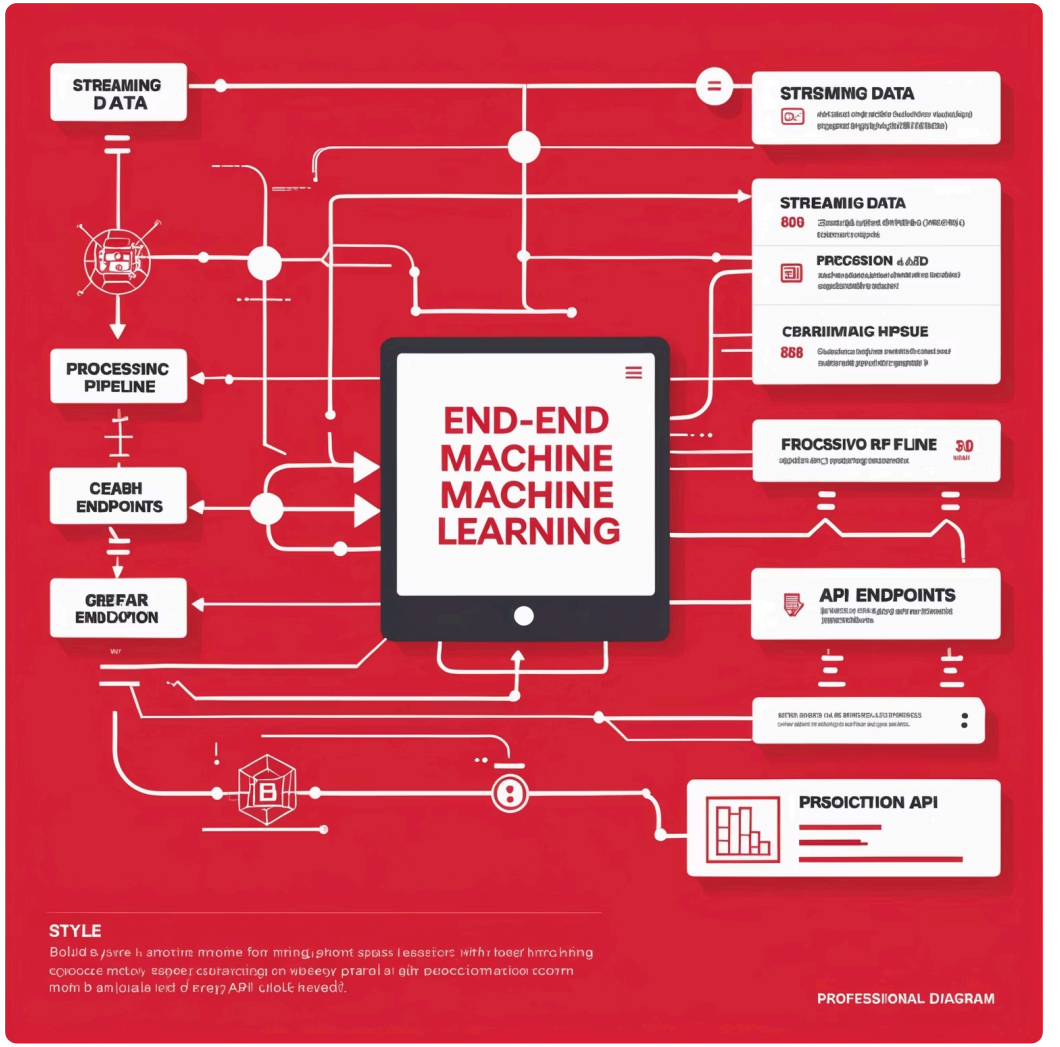
Weeks 10–11: Capstone Project

Project Challenge:

Build a complete real-time streaming prediction pipeline

- Part 1: Streaming ingestion and processing
- Part 2: Deployment and monitoring

Deliverables: Fully deployed REST API serving predictions with documentation and testing



Required Software Installation

Python 3.9+

Core programming language for all our ML implementations. We recommend Python 3.9 for optimal compatibility with all libraries.

Git

Version control system for tracking code changes and collaborating on projects. Required for accessing course materials.

Docker & Docker Compose

Containerization platform for creating reproducible environments and deployable applications.

Java 8+

Required runtime for Apache Spark. OpenJDK 11 is recommended for best compatibility and performance.

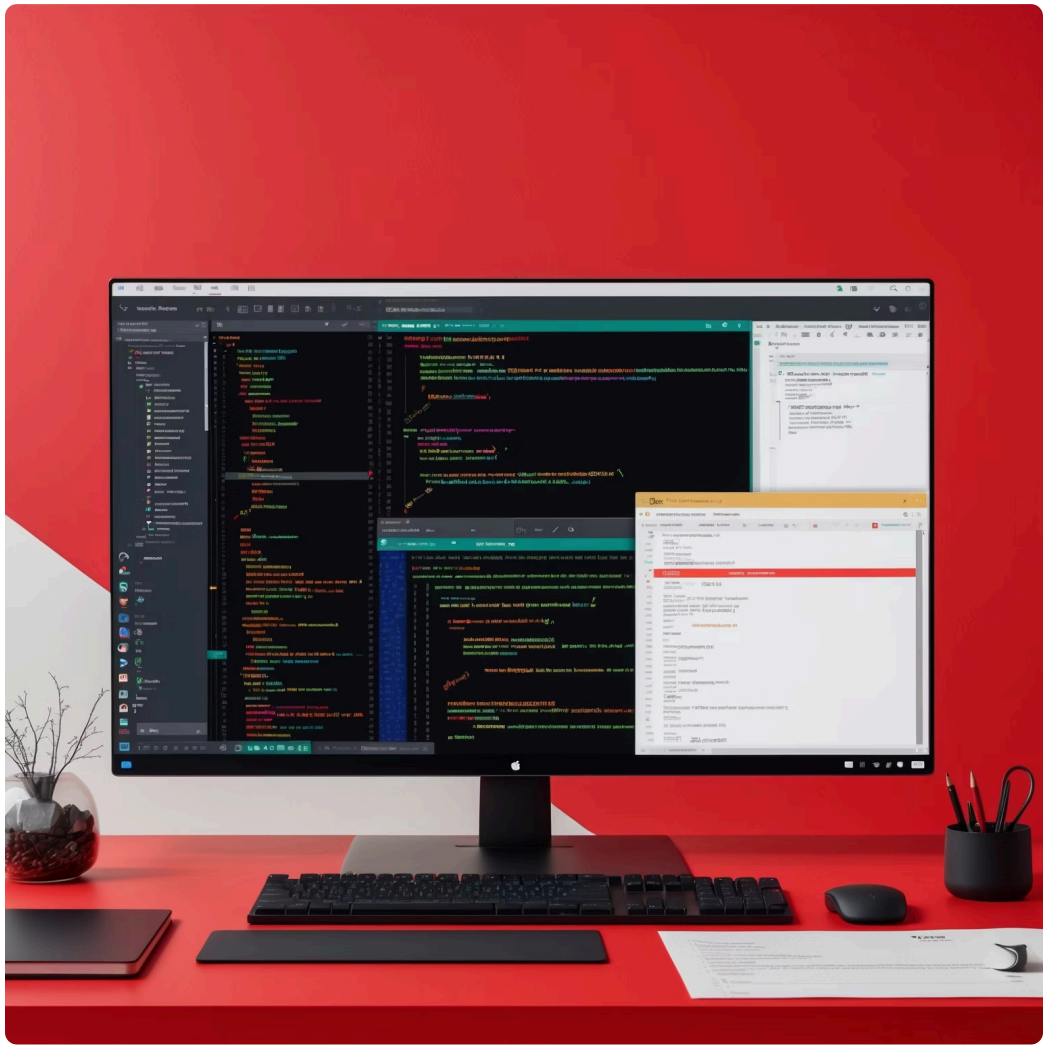
Homework: Complete all installations before Week 2 begins!

Development Environment Setup

Recommended Setup:

- VSCode with Python and Docker extensions
- Jupyter Lab for interactive analysis
- Terminal access for command-line tools
- GitHub account for assignments

All course examples will assume this environment configuration for consistency.



Environment Management

Conda

Cross-platform package and environment manager that handles both pip and conda packages

- Creates isolated environments with specific Python versions
- Installs binary packages without compilation
- Simplifies dependency management

Virtualenv/venv

Lightweight Python-specific virtual environment tool

- Standard in Python's standard library (venv)
- Faster creation than Conda
- Uses pip for package management

The course will use Conda for consistency, but either option is acceptable for your work.

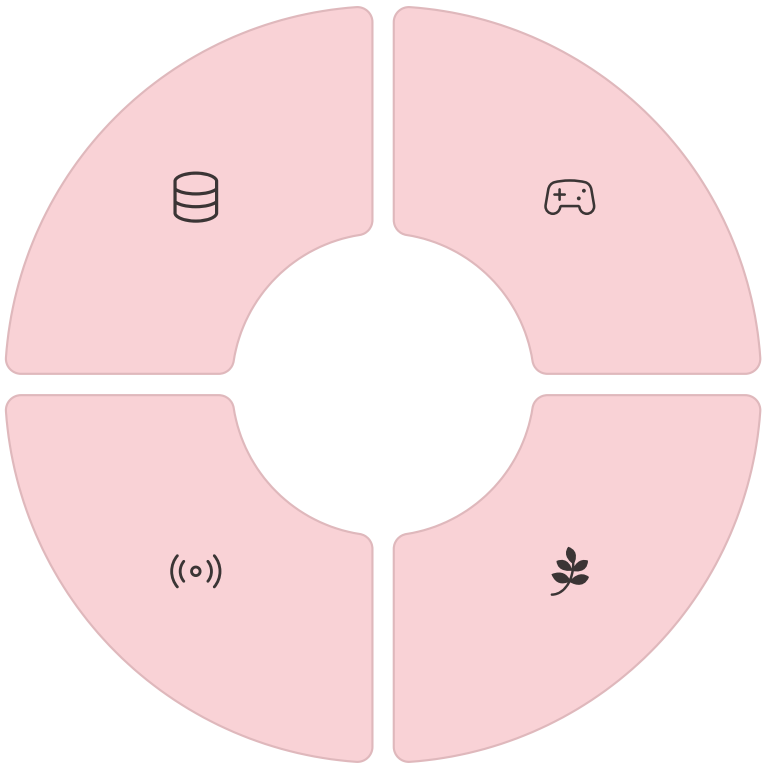
Key Python Libraries

Data Processing

- Pandas: tabular data manipulation
- NumPy: numerical computing
- PySpark: distributed data processing

Infrastructure

- apache-airflow: orchestration
- kafka-python: streaming
- boto3: AWS integration



ML Modeling

- scikit-learn: traditional ML algorithms
- XGBoost: gradient boosting
- Hyperopt: hyperparameter optimization

ML Ops

- ZenML: ML pipeline management
- MLflow: experiment tracking
- FastAPI: model serving

Data Processing Libraries Deep Dive

Pandas

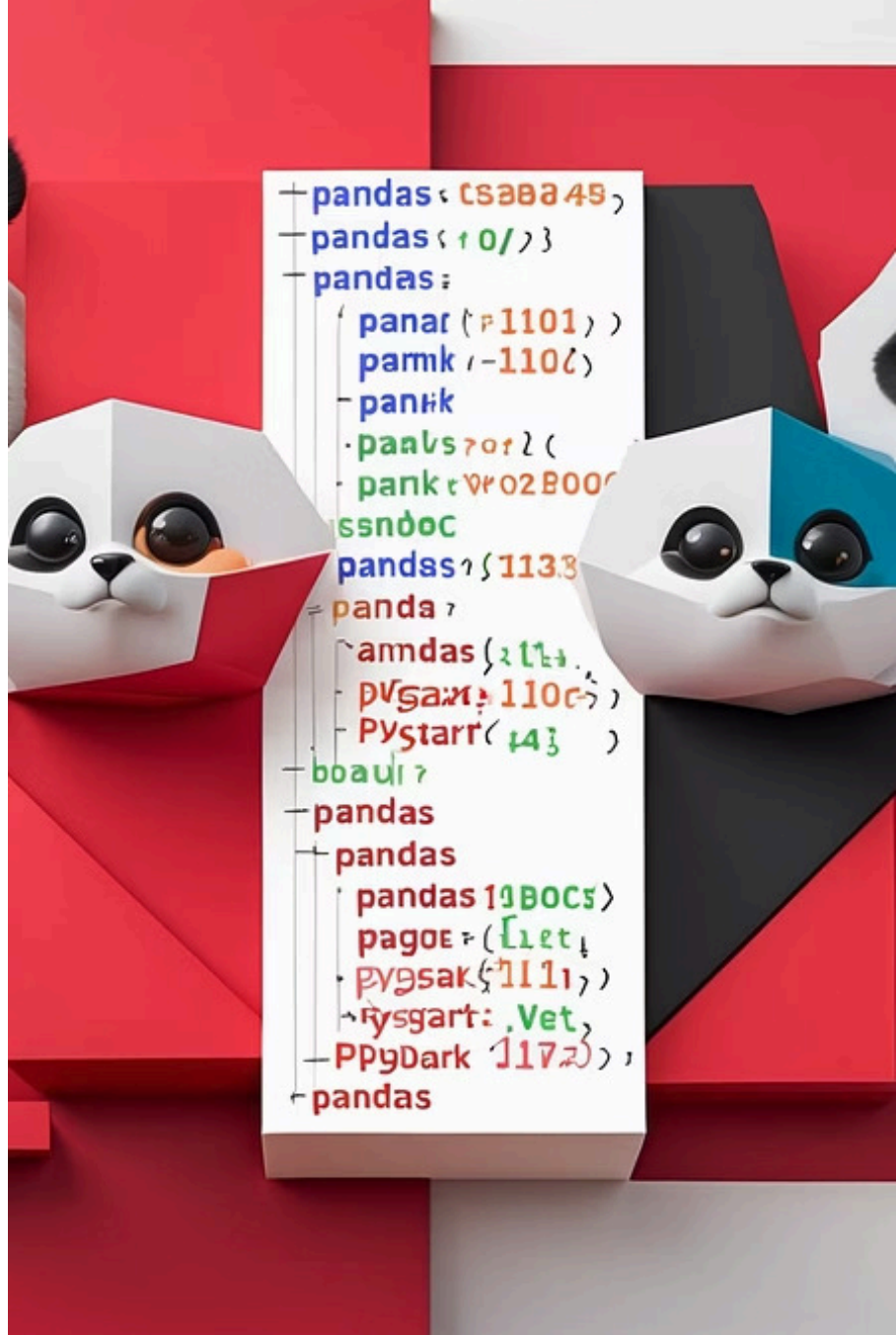
In-memory tabular data processing:

- DataFrame and Series data structures
- Data cleaning, transformation, and aggregation
- Memory-optimized for smaller datasets

PySpark

Distributed data processing for large datasets:

- DataFrame API similar to Pandas
- SQL query support
- Built-in ML algorithms (MLlib)
- Scales across clusters



ML Tracking and Pipeline Tools

MLflow

Open-source platform for managing the ML lifecycle:

- Tracking: log parameters, metrics, and artifacts
- Projects: package code for reproducibility
- Models: package models for deployment

ZenML

Framework for creating reproducible ML pipelines:

- Pipeline versioning
- Artifact tracking
- Integration with MLflow
- Multi-stage workflows

Container Tools Overview

Docker

Container platform for packaging applications:

- Dockerfile: defines environment
- Images: immutable snapshots
- Containers: running instances
- Registry: for sharing images

Docker Compose

Multi-container orchestration:

- YAML configuration
- Service definitions
- Networking between containers
- Volume management

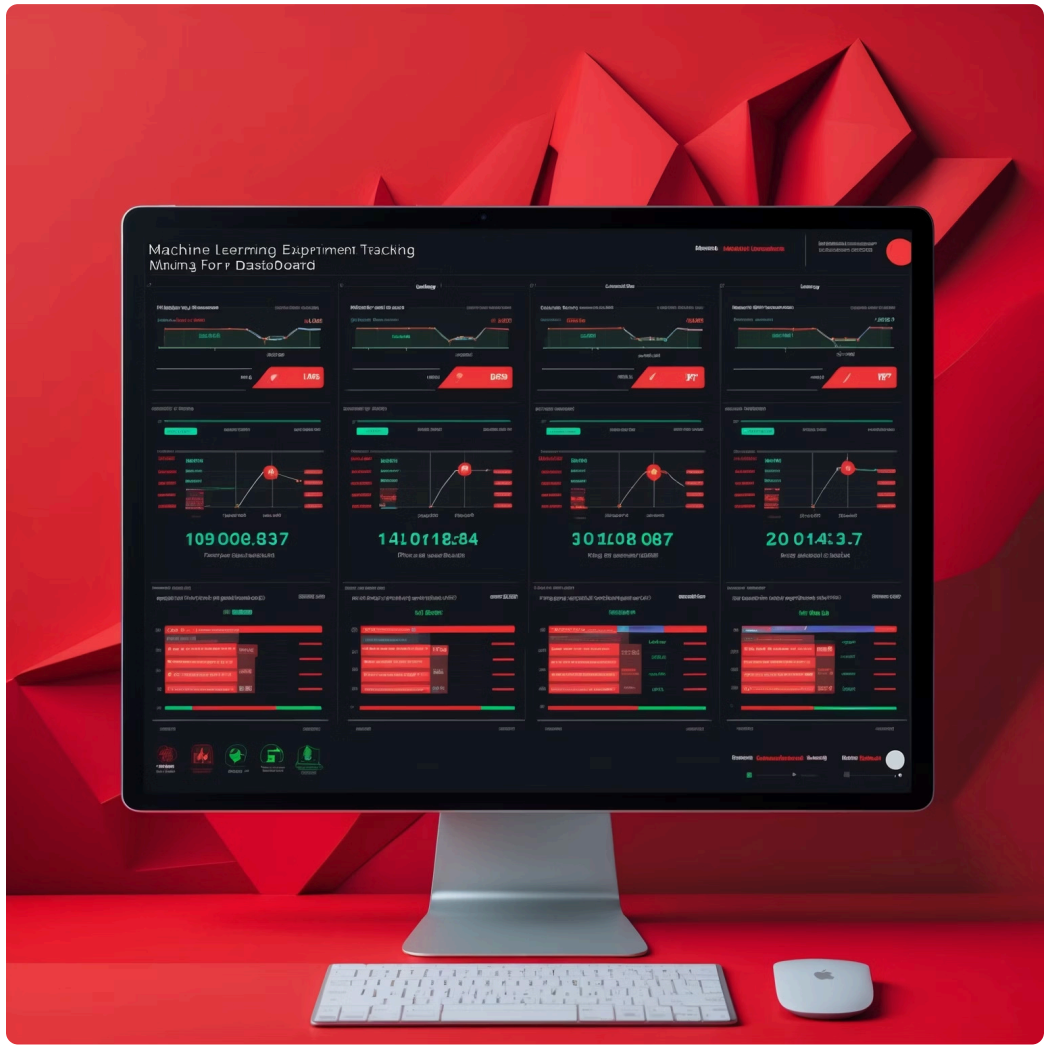
Mini Project 1: Experiment Tracking

Project Objective:

Implement a reproducible ML pipeline with experiment tracking

Components:

- ZenML pipeline definition
- MLflow experiment tracking
- Model versioning and registration
- Performance comparison dashboard



Mini Project 1: Technical Requirements

Pipeline Structure

- Data ingestion step
- Data validation step
- Feature engineering step
- Model training step
- Evaluation step

MLflow Integration

- Parameter logging
- Metric tracking
- Artifact storage
- Run comparison

Deliverables

- Working ZenML pipeline
- MLflow experiment UI
- Model registry
- Documentation

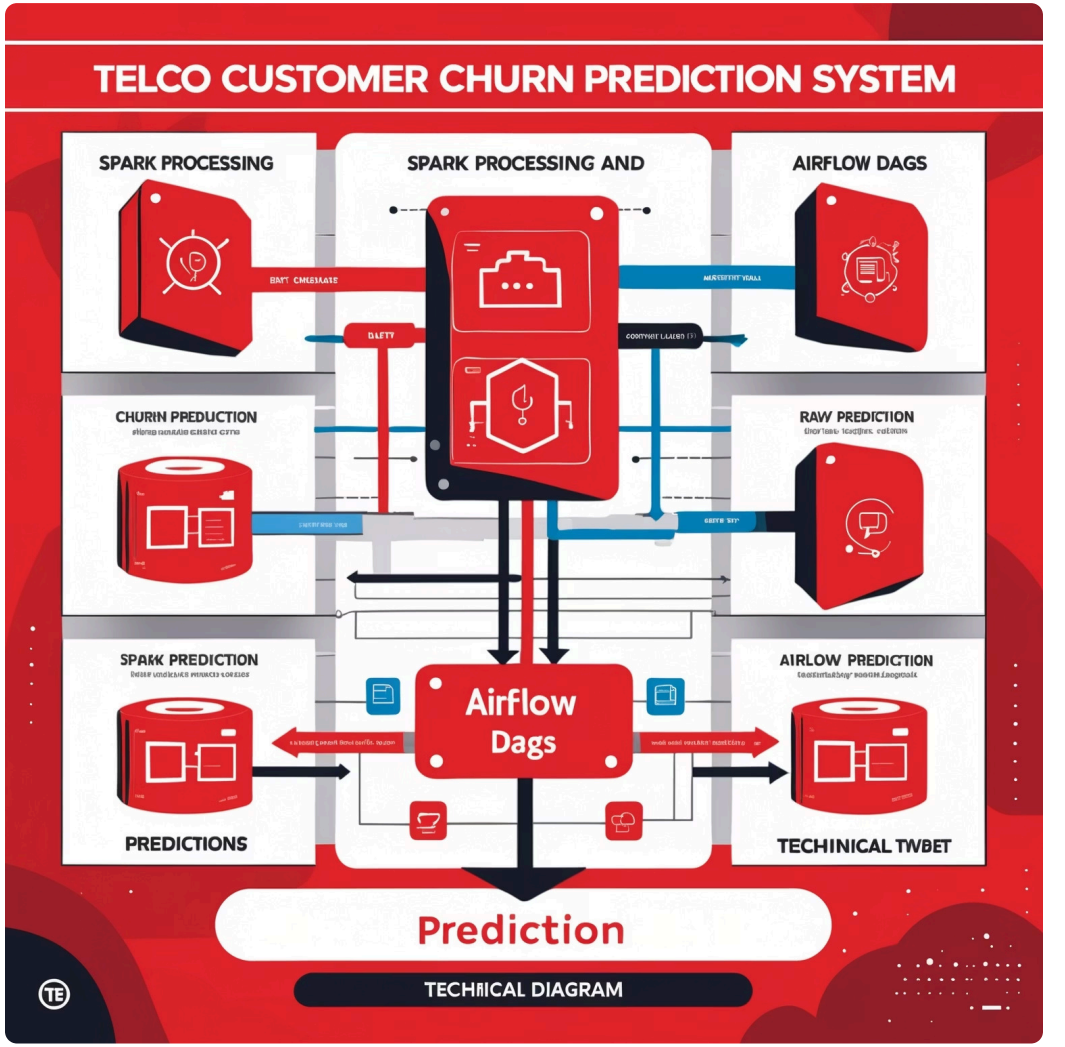
Mini Project 2: Batch Pipeline

Project Objective:

Build an end-to-end batch prediction pipeline for Telco Customer Churn

Components:

- Spark-based data processing
- Airflow orchestration DAGs
- Automated model retraining
- Batch prediction generation



Mini Project 2: Technical Requirements

Data Processing

PySpark-based ETL pipeline:

- Feature engineering
- Data cleaning
- Train/test splitting

Orchestration

Airflow DAGs for:

- Daily data ingestion
- Weekly model retraining
- Daily prediction generation

Monitoring

Performance tracking:

- Data drift detection
- Model metrics logging
- Pipeline execution stats

Capstone Project: Technical Requirements

Functional Requirements:

- Ingest streaming data from Kafka
- Process data in real-time with Spark Streaming
- Make predictions using deployed ML model
- Serve predictions via REST API
- Log all operations for auditability

Non-Functional Requirements:

- Latency < 500ms for prediction requests
- Throughput of 100+ requests/second
- 99.9% system availability
- Comprehensive error handling
- Monitoring dashboards

Hackathon Challenge

Test your skills in a 120-hour challenge

- **Format:** Solo competition
- **Theme:** Solve a provided E2E ML use case
- **Challenge:** Build a E2E ML pipeline
- **Prizes:** Course credit, Zuu Crew Merch and exclusive mentorship sessions

Top 10 Candidates will present their solutions live to the class.

Hackathon: Judging Criteria

Technical Implementation (40%)

- Correct use of ZenML and MLflow
- Pipeline structure and modularity
- Code quality and organization

ML Engineering Best Practices (30%)

- Data validation
- Model versioning
- Error handling
- Documentation

Innovation & Presentation (30%)

- Creative approaches
- Clear explanation of design choices
- Demonstration of working pipeline

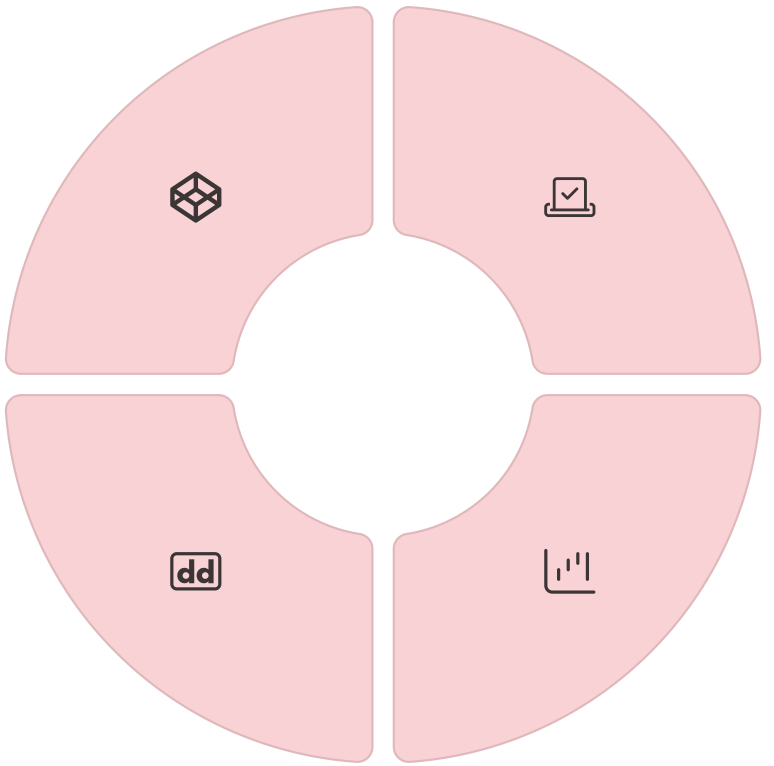
Assessment & Grading Criteria

Code Quality (30%)

- Readability and PEP8 compliance
- Modularity and reusability
- Comprehensive docstrings
- Thoughtful variable naming

System Architecture (20%)

- Component integration
- Scalability considerations
- Production readiness
- Documentation completeness



Best Practices (30%)

- Unit and integration testing
- CI-style validation checks
- Robust error handling
- Logging and monitoring

Model Performance (20%)

- Appropriate metrics
- Validation methodology
- Baseline comparisons
- Feature importance analysis

Communication & Support Channels

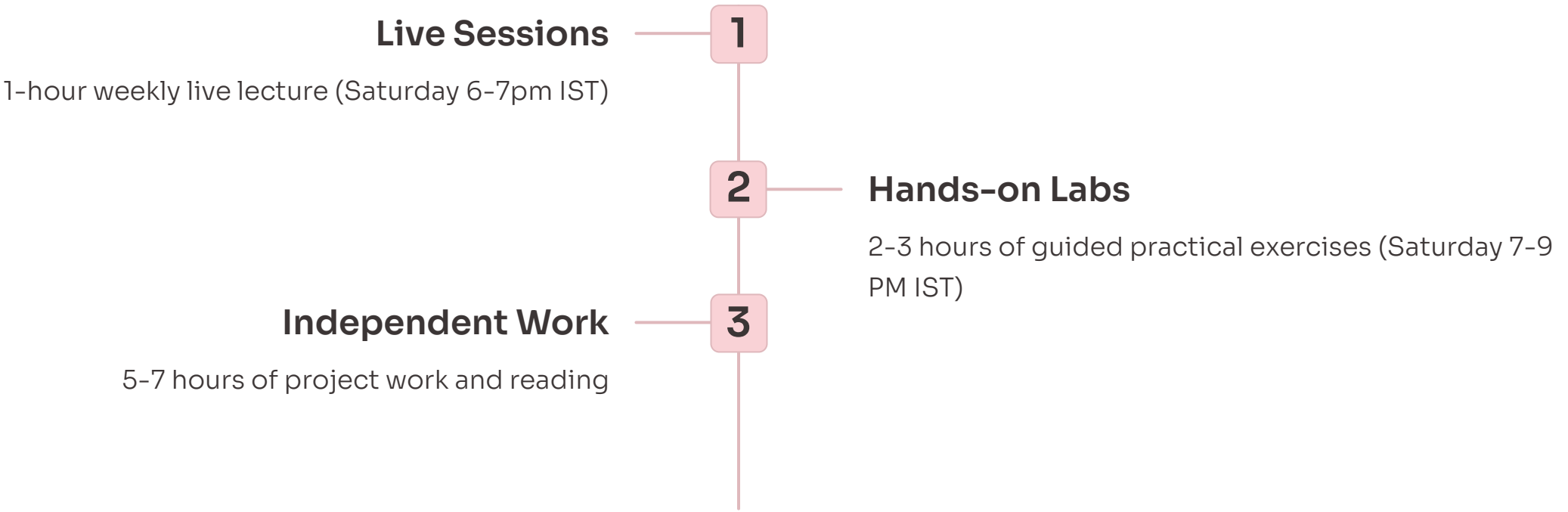
Discord Community

- **Link:** https://discord.gg/5FdQ5x5h
- **Channels:**
  - #announcements
  - #week1, #week2, etc.
  - #help-technical
  - #general-discussion

Office Hours & Resources

- **Office Hours:** Weekends (Sat/Sun)
- **AMA Threads:** Weekly Q&A
- **GitHub Repo:** Shared code examples
- **Response Time:** Within 24 hours

Weekly Schedule & Commitment



Total weekly commitment: 10-12 hours

Pre-Course Preparation

Required Tutorial Videos:

1. **Python OOP**
2. **FastAPI**
3. **Docker**
4. **Docker Compose**

Industry Context: Why These Tools Matter

Tool Adoption in Industry

According to the 2023 ML Tools Survey:

- 76% of companies use Docker for ML deployment
- 64% use Airflow for orchestration
- 52% use Spark for large-scale data processing
- 48% use Kafka for real-time data pipelines

Skills Gap in the Market

LinkedIn's 2023 Emerging Jobs Report shows:

- ML Engineers command 25% higher salaries than Data Scientists
- Production ML skills are mentioned in 82% of senior ML job postings
- MLOps expertise is the fastest-growing skill requirement

Career Opportunities After This Course

Graduates of this course have pursued various career paths in the machine learning and data engineering space:

ML Engineer

Build and deploy machine learning models at scale in production environments

MLOps Specialist

Focus on the operational aspects of machine learning systems including monitoring, governance, and CI/CD pipelines

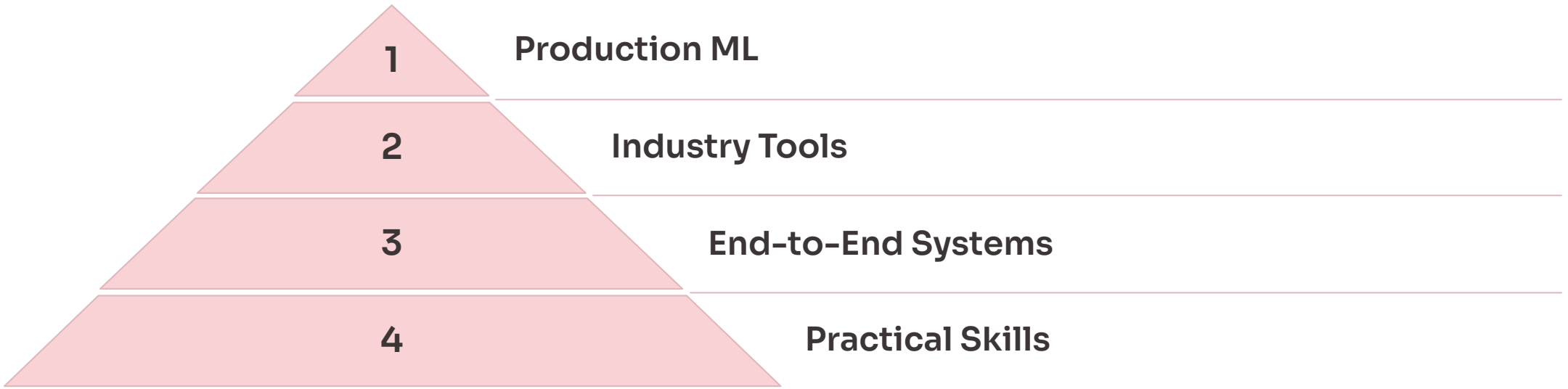
ML Skill Specialist

Develop specialized expertise in specific ML domains like NLP, computer vision, or reinforcement learning

ML Platform Engineer

Build internal ML platforms that enable data scientists to productionize their models

Key Takeaways



This course bridges the gap between academic machine learning and real-world ML engineering. You'll learn to build complete, scalable, and maintainable machine learning systems using industry-standard tools and best practices.

By the end, you'll have practical experience with every component of the ML lifecycle, from data ingestion to model deployment and monitoring.

Q & A

Any questions about the course structure or requirements?

Remember to check the pre-work instructions in the #week1 channel on Discord!

Join Discord | Email Instructors