# CAPSTONE PROJECT

# Power System Fault Detection and Classification

Presented By:
1. Srisurya M-Sri Venkateswara College of Engineering-CSE

# OUTLINE

- **Problem Statement** (Should not include solution)
- **Proposed System/Solution**
- **System Development Approach** (Technology Used)
- **Algorithm & Deployment**
- **Result (Output Image)**
- **Conclusion**
- **Future Scope**
- **References**

# Problem Statement

Design a machine learning model to detect and classify different types of faults in a power distribution system. Using electrical measurement data (e.g., voltage and current phasors), the model should be able to distinguish between normal operating conditions and various fault conditions (such as line-to-ground, line-to-line, or three-phase faults). The objective is to enable rapid and accurate fault identification, which is crucial for maintaining power grid stability and reliability.

# Proposed Solution

- Design a machine learning-based model capable of accurately detecting and classifying faults in a power distribution system using voltage and current phasor data.
- The model will utilize labeled electrical measurements to differentiate between normal operations and specific fault types like Line-to-Ground, Line-to-Line, and Three-Phase faults

**Key Components:**

**Data Collection**:

- We start by using a real-world dataset from Kaggle that contains labeled fault data, including voltage and current readings under different fault conditions. This gives us the raw material to train our model.

**Preprocessing**:

- Before we feed the data to any algorithm, we clean it up — removing noise, filling in any missing values, and normalizing everything so that the model doesn't get confused by scale differences. Think of it like prepping ingredients before cooking a dish .

**Model Training**:

- With clean data in hand, we train a machine learning model to recognize patterns and classify different fault types. Models like Decision Trees, Random Forest, or even CNNs help the system "learn" what each fault looks like based on past data.

**Evaluation**:

- Finally, we test how smart our model actually is — using accuracy, precision, recall, and F1-score to see how well it can identify faults. It's like giving your model a report card

# System Approach

Deployment on IBM Cloud Lite

- IBM Watson Studio is used for model training and testing
- IBM Cloud Object Storage hosts the dataset and artifacts
- IBM Cloud Functions exposes the model for real-time inference
- Dashboarding and logs are managed using IBM Monitoring

# Algorithm & Deployment

**Model Used**:

- Trained models like Random Forest, SVM, and 1D-CNN to classify fault types based on voltage and current phasors.

**Input Features**:

- Extracted values like RMS, peak, FFT, and wavelet components from the raw electrical signals.

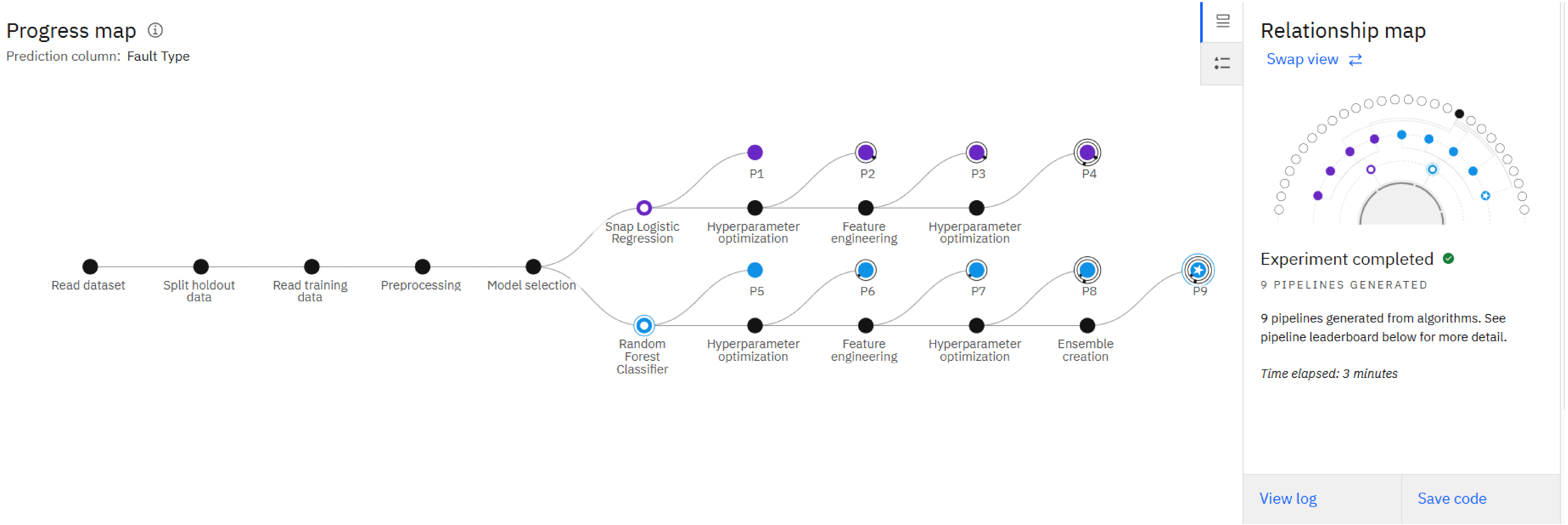
**Training Process**:

- Data was split into training and testing sets (e.g., 80/20), and tuned using cross-validation and hyperparameter tuning.

**Deployment**:

- The final model is deployed using IBM Watson Studio and made accessible through a Flask API or IBM Cloud Functions for real-time prediction

# Result



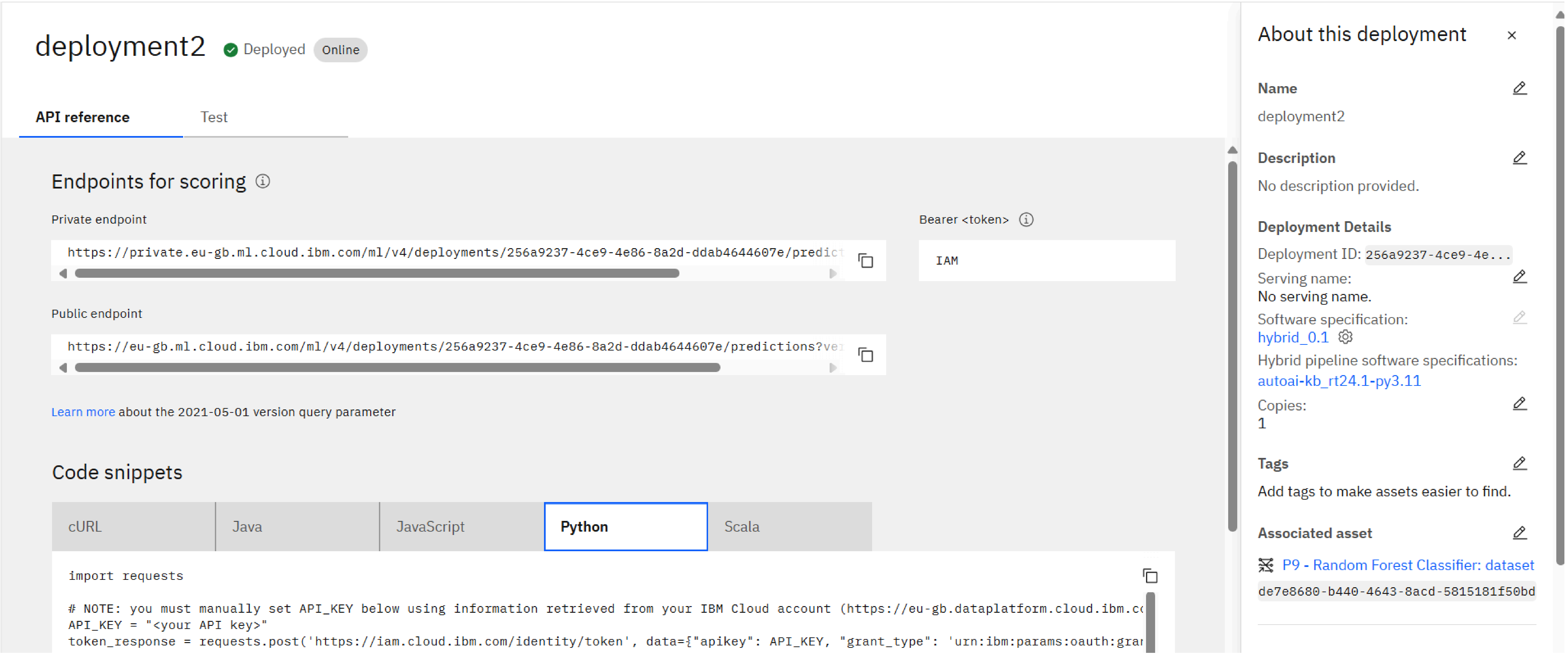
Progress map
Prediction column: Fault Type
Read dataset
Split holdout data
Read training data
Preprocessing
Model selection
Snap Logistic Regression
Random Forest Classifier
Hyperparameter optimization
Feature engineering
Hyperparameter optimization
Ensemble creation
P1
P2
P3
P4
P5
P6
P7
P8
P9
Relationship map
Swap view
Experiment completed
9 PIPELINES GENERATED
9 pipelines generated from algorithms. See pipeline leaderboard below for more detail.
Time elapsed: 3 minutes
View log
Save code

# Result

Pipeline leaderboard

| | Rank | Name | Algorithm | Specialization | Accuracy (Optimized) Cross Validation | Enhancements | Build time |
|---|---|---|---|---|---|---|---|
| ★ | 1 | Pipeline 9 | Batched Tree Ensemble Classifier (Random Forest Classifier) | INCR | 0.409 | HPO-1 FE HPO-2 BATCH | 00:00:58 |
| | 2 | Pipeline 8 | Random Forest Classifier | | 0.409 | HPO-1 FE HPO-2 | 00:00:53 |
| | 3 | Pipeline 4 | Snap Logistic Regression | | 0.393 | HPO-1 FE HPO-2 | 00:00:35 |
| | 4 | Pipeline 3 | Snap Logistic Regression | | 0.393 | HPO-1 FE | 00:00:30 |

# Result




edunet
foundation

# Result

deployment2 Deployed Online

API reference Test

Enter input data

Text JSON

Enter data manually or use a CSV file to populate the spreadsheet. Max file size is 50 MB.

Download CSV template   Browse local files ↗   Search in space ↗   Clear all ×

| | Fault ID (other) | Fault Location (Latitude, Longitude) (other) | Voltage (V) (double) | Current (A) (double) | Power Load (MW) (double) | Temperature (°C) (double) | Wind Speed (km/h) (double) | Weather Condition (other) | Maintenance Status (other) | Component Health (other) | Duration of Fault (hrs) (double) | Down time (hrs) (double) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | F002 | (34.056, -118.245) | 1800 | 180 | 45 | 28 | 15 | Rainy | Completed | Faulty | 3 | 5 |
| 2 | F017 | (34.9346, -118.9658) | 2263 | 229 | 55 | 21 | 16 | Thunderstorm | Completed | Faulty | 6 | 3.7 |
| 3 | F028 | (34.7606, -118.9892) | 1860 | 246 | 49 | 36 | 13 | Thunderstorm | Completed | Normal | 4.6 | 2.8 |
| 4 | F505 | (34.5034, -118.4528) | 1602 | 222 | 55 | 22 | 20 | Windstorm | Completed | Normal | 4.9 | 6.4 |

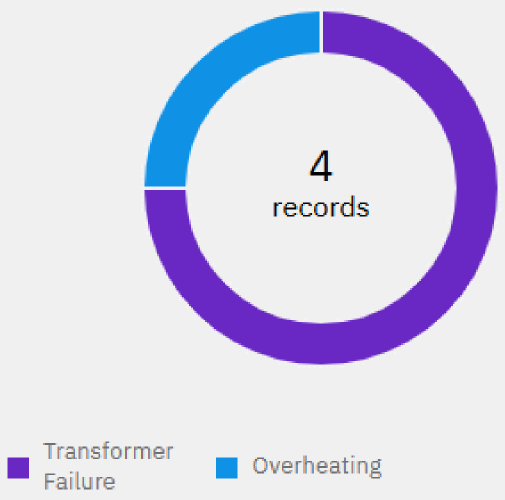
# Result

## Prediction results

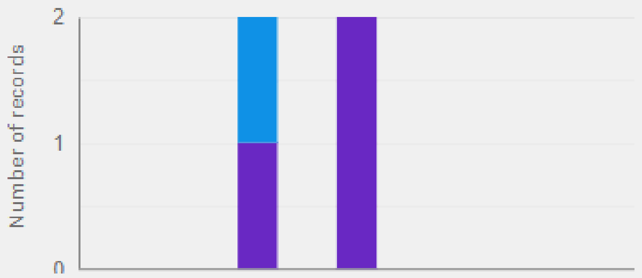
Prediction type

### Multiclass classification

**Prediction percentage**




**Confidence level distribution**




Display format for prediction results

(•) Table view ( ) JSON view

Show input data

| | Prediction | Confidence |
|---|---|---|
| 1 | Transformer Failure | 35% |
| 2 | Overheating | 37% |
| 3 | Transformer Failure | 43% |
| 4 | Transformer Failure | 44% |
| 5 | | |
| 6 | | |
| 7 | | |
| 8 | | |
| 9 | | |
| 10 | | |
| 11 | | |
| 12 | | |
| 13 | | |
| 14 | | |
| 15 | | |
| 16 | | |

# Conclusion

- This project shows how machine learning can make power systems smarter and more reliable.
- By using real phasor data and training models to detect different types of faults, we can spot problems early and respond faster — reducing downtime and protecting the grid.
- Deploying the model on IBM Cloud Lite makes it scalable, accessible, and ready for real-world use.
- Overall, this solution brings us one step closer to building smarter, self-aware energy systems.

# Future scope

- Deploy on edge devices like Raspberry Pi for real-time fault detection
- Extend to detect more fault types (e.g., transformer faults, harmonics)
- Use reinforcement learning for continuous self-improvement
- Add GPS-based fault location and instant alerts
- Build a live dashboard using IBM Cloud for real-time monitoring

# References


- Kaggle Dataset – Power System Faults Dataset
- IBM Cloud Docs – IBM Watson Studio, Cloud Object Storage, Cloud Functions
- IEEE Papers – Fault Detection in Smart Grids using Machine Learning
- scikit-learn Documentation – https://scikit-learn.org
- TensorFlow & Keras Docs – https://www.tensorflow.org

# IBM Certifications

7/24/25, 6:28 PM Completion Certificate | SkillsBuild

IBM **SkillsBuild** Completion Certificate

This certificate is presented to

Srisurya M

for the completion of

## Getting Started with Artificial Intelligence

(PLAN-E624C2604060)

According to the Your Learning Builder - Plans system of record

**Completion date:** 18 Jul 2025 (GMT)

edunet foundation

# IBM Certifications



Print Share ×

**IBM SkillsBuild** Completion Certificate

This certificate is presented to

Srisurya M

for the completion of

## Journey to Cloud: Envisioning Your Solution

(PLAN-32CB1E21D8B4)

According to the Your Learning Builder - Plans system of record

**Completion date:** 18 Jul 2025 (GMT)

# IBM Certifications



IBM **SkillsBuild** Completion Certificate

This certificate is presented to

Srisurya M

for the completion of

**Lab: Retrieval Augmented Generation with LangChain**

(ALM-COURSE_3824998)

According to the Adobe Learning Manager system of record

**Completion date:** 24 Jul 2025 (GMT)

**Learning hours:** 20 mins

# THANK YOU