Course Name: Python & Data Engineering

Duration: 13 Weeks.

Schedule: 4 days/week × 4 hours/day = **16 hours/week.**

Total: ~200 hours.

Level: Beginners / Fresh Graduates.

Language: Arabic / English.

Format: Hands-on + Real Company Scenario.

Bootcamp Learning Outcome

By the end, students will be able to:

- Use **Python for data engineering**, including:
    - Pandas & NumPy for data manipulation.
    - PySpark for distributed processing.
    - Requests & APIs for data ingestion.
    - Python-based **networking & automation** (scheduling, integrations, system tasks).
- Design **end-to-end data pipelines.**
- Ingest batch & streaming data.
- Model data for analytics.
- Build and manage a **local data warehouse.**
- Use modern data engineering tools (Docker, Spark, Kafka, Airflow).
- Serve data for analytics / BI.

Course Structure (5 Sections → 13 Weeks)

Section	Weeks 13
1. Foundations of Data Engineering	1 - 2
2. Data Ingestion and Data Processing	3
3. System , Automation & engineering practices	4 - 5
4. Data Storage	6 - 7
5. Data Ingestion & Processing	8 - 11
6. Data Serving & Final Project	12 - 13

SECTION 1: FOUNDATIONS OF DATA ENGINEERING (Weeks 1–2)

Goal: Build a solid conceptual and technical base in Data Engineering and Python, and understand how data is handled locally before scale.

week 1 (Data Engineering Foundations & Python Basics):

Topics:

- What is **Data Engineering** (vs Data Analysis / Data Science).
- Data Engineer career path & responsibilities..
- Python fundamentals.
- Working with Jupyter Notebook.

Tools:

- Python.
- Jupyter.

Week 2 (Python for Data Processing & Data Quality):

Topics:

- Advanced Python for Data workflows.

- NumPy for numerical computation.
- Pandas for data manipulation.
- Data quality & validation fundamentals..

Tools & Libraries:

- Python.
- NumPy.
- Pandas.

SECTION 2: DATA INGESTION & DATA SOURCES (Week 3)

Goal: Enable learners to collect data from real-world sources and understand different data formats and structures.

Week 3 (Data Sources, SQL & API):

Topics:

- SQL fundamentals for data extraction.
- Types of data sources:
    - Databases.
    - APIs.
    - Logs.
    - Files.
- Structured vs Semi-structured vs Unstructured.
- Generating fake data.
- REST API basics.

Tools:

- Faker.
- Python requests.
- Public APIs (weather, finance, etc.)

SECTION 3: SYSTEMS, AUTOMATION & ENGINEERING PRACTICES (Week 4 - 5)

Goal: Introduce production-level thinking: scalability, automation, infrastructure awareness, and collaboration.

Week 4 (Distributed Systems & Data Engineering Automation)

Topics:

- Automation & Scheduling.
- Machine Learning (DE Perspective).

Week 5 (Infrastructure, Networking & Version Control for Data Engineers):

Topics:

- Distributed systems basics.
- Networking basics.
- Networking Automation with Python.
- Linux fundamentals.
- Git & GitHub.
- GitFlow.

Tools:

- Linux(Ubuntu).
- Python.
- Git / Github.
- SSH / curl.

SECTION 4: DATA STORAGE (Weeks 6–7)

Goal: Design and choose the right data storage and modeling solutions for transactional and analytical data.

Week 6 (Databases & Modeling):

Topics:

- Database fundamentals
    - Normalization.
    - Indexing.
    - Transactions.
- OLTP vs OLAP.
- Horizontal vs Vertical scaling.
- Data modeling
    - Star schema.
    - Snowflake schema.
- Slowly Changing Dimensions (SCD 1, 2).

Tools:

- PostgreSQL.
- dbdiagram.io.

Week 7 (NoSQL, Warehouses & Local Analytics DB):

Topics:

- NoSQL databases
    - Document (MongoDB).
    - Key-Value (Redis).
    - Column (Cassandra – theory).
    - Graph (Neo4j – theory/demo).
- Data Warehouse concepts
    - Data Warehouse vs Data Mart.
    - Data Lake.
    - Data Mesh (conceptual).
    - Data Lakehouse
- Local analytical DB

    - ClickHouse.
- Hadoop Introduction.

Tools:

- MongoDB.
- Redis.
- ClickHouse.
- Docker.
- DBeaver.
- Hadoop.

SECTION 5: DATA INGESTION & PROCESSING (Weeks 8–9)

Goal: Learn how to ingest, process, and orchestrate large-scale data using batch and streaming architectures.

Week 8 (Batch Data Pipelines & Data Warehousing):

Topics:

- Batch processing architecture.
- Batch vs Streaming vs Hybrid.
- ETL vs ELT in data platforms..
- Data pipeline design patterns.
- Data warehousing concepts.
- Hadoop ecosystem overview (HDFS, YARN).
- Hive architecture and use cases.
- Partitioning and schema-on-read.

Tools:

- Apache Hive.
- Apache Hadoop (HDFS – conceptual & practical).
- Python.

Week 9 (Streaming Fundamentals & Spark Processing):

Topics:

- Streaming data fundamentals.
- Event-driven architectures.
- Kafka fundamentals (topics, partitions, offsets).
- Distributed processing concepts.
- Spark architecture (Driver, Executors, Cluster Manager).
- RDDs vs DataFrames vs Datasets.
- Spark transformations and actions.
- Introduction to Spark Structured Streaming.
- pipeline orchestration concepts.

Tools:

- Apache Spark.
- PySpark.
- Apache Kafka (conceptual + integration).
- Airflow.

Week 10 (Advanced Streaming & Real-Time Processing):

Topics:

- Limitations of micro-batch streaming.
- Apache Flink architecture.
- Event time vs processing time.
- Watermarks and late data handling.
- Stateful vs stateless stream processing.
- Fault tolerance and checkpoints.
- Flink vs Spark (design and use cases).
- Hybrid architectures (stream → Hive / batch layer).

Tools:

- Apache Flink.
- Apache Kafka.

Week 11 (DevOps for Data Engineers):

Topics:

- Docker fundamentals & Docker Compose.
- CI/CD basics (GitHub Actions).
- Monitoring concepts.
- Visualizing metrics.
- Code Testing.

Tools:

- Docker.
- GitHub Actions.
- Prometheus.
- Grafana.

- Pytest.

SECTION 6: DATA SERVING & FINAL PROJECT (Weeks 11–12)

Goal: Serve transformed data for analytics and demonstrate an end-to-end data engineering solution through a final project.

Week 11 (Data Serving & Analytics):

Topics:

- What is Data Serving?
- Serving layers.
- BI concepts.
- Query optimization..
- APIs for data access.

Tools:

- Metabase.
- FastAPI.

Week 12(Final Project):

Project Goal

Build End-to-End Data Platform

Project Requirements

- Data generation.
- Ingestion (batch + optional streaming).
- Storage (Postgres + ClickHouse).
- Transformation (Spark).
- Orchestration.
- Serving layer (BI or API).
- Dockerized setup.