# Weeks 11 & 12: Final Project Assignment

# Objective:

The objective of this project is to build a data pipeline that integrates at least four major components from the following ecosystem: **NiFi**, **HDFS**, **Hive**, **HBase**, and **Spark**. You will demonstrate the seamless flow of a single dataset through multiple stages, processing and transforming the data using the selected technologies. This project will showcase your understanding of how to move data between systems in a distributed environment while leveraging the power of big data tools for storage, querying, and processing.

# Requirements:

1. Pipeline Composition:

* Your pipeline must utilize at least four components from the following list:
  + NiFi for data ingestion
  + HDFS for data storage
  + Hive for data warehousing
  + HBase for NoSQL storage
  + Spark for data processing, querying, and machine learning

1. Dataset:

* Use a **single dataset** of your choice (must be unique; do not use the datasets provided in the example guides).
* The **same** dataset must be loaded at the beginning and processed across all the selected components.

1. Component Integration:

* The components must be fully connected and work together in sequence.
* For example:
  + Use NiFi to ingest data and load it into HDFS, then move it to Hive for querying, and finally query it with Spark.
  + Alternatively, use NiFi to load data into HDFS, use Spark to process the data from HDFS, and then write the output to HBase.
  + You may refer to the example guides for help in configuring these integrations, **but you must apply them to your own unique dataset and project.**

1. Demonstration of Skills:

* Clearly show how each component is utilized in your pipeline, and how they interact to process the dataset from start to finish.

# Important Notes:

You will be provided with examples that demonstrate how to connect NiFi, HDFS, Hive, Spark, and HBase. However, you must use a unique dataset and schema. Reusing data or schemas from the examples will result in a grade of zero.

Ensure the dataset and all downstream schemas are entirely unique.

# Submission Guidelines:

1. Submit your work in a comprehensive report.
2. Your report must:

* Include a clear explanation of any issues encountered and how they were resolved.
* Include the link to your chosen dataset, ensuring it is different from any provided examples.
* Include screenshots demonstrating the successful execution of your pipeline at each stage, showing the dataset flow across all components (NiFi, HDFS, Hive, HBase, Spark).
* Provide the full code used for the project (e.g., NiFi templates, PySpark scripts, Hive queries) in your report.
* Ensure that the report is well-structured, with each section clearly labeled (Introduction, Dataset, Pipeline Overview, Issues Encountered, Screenshots, Code, Conclusion).

# Evaluation Criteria:

* Completeness and correctness of your implementation.
* Effective use of the technologies taught in the course.
* Uniqueness of the dataset and schemas.
* Clarity and organization of your report.

## Examples:

You can refer to examples from the following repository: [DSC650 Week 11-12 Assignment Examples](https://github.com/bellevue-university/dsc650-updated/tree/main/week11-12/assignment/examples)  
  
You are encouraged to use these resources as guides, **but you cannot use the same datasets or schemas provided in these examples:**

**Please refer to the interactions in the examples below, as the Docker networking has already been configured and tested for seamless connectivity between the containers.**

* **Example\_Project.json:** This NiFi template downloads **customers-100.csv** from the assignment folder and puts it into HDFS in the /tmp directory. You can use your own data, load it to HDFS, and then put it in a Hive table. You may also explore Hive external tables, which allow Hive to read data from a directory without moving the data.
* **SparkAddLibraries.docx:** This guide walks through adding a Python library and running it with PySpark (e.g., using pip to install the requests library). You may explore other libraries in your project.
* **HdfsToHivePySpark.docx**: This document guides you through reading a CSV file from HDFS, creating a Hive table using PySpark, loading the data into the Hive table, and then querying the table using PySpark.
* **QueryHivePySpark.docx:** This document guides you through creating a Hive table with sample data and querying it using PySpark.
* **WriteHBasePyspark.docx:** This document walks you through installing the happybase Python library, creating an HBase table, and writing data to it using PySpark.
* **SparkMachineLearning.docx:** This document walks you through creating a Linear Regression Machine Learning model with PySpark, Hive, and Spark MLib.
* **HiveQueryHbase.docx**: This document walks you through creating an HBase table, loading data into it, and using Hive to query the data stored in the HBase table.