**PROCTORING ANOMALY DETECTION IN ONLINE EXAMS USING DATABRICKS AND TABLEAU**

A PROJECT PHASE II REPORT

Submitted by

**VARUN V (231801185)**

**VIGNESH R (231801508)**

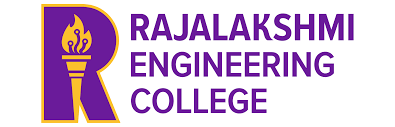
**MOHAMMED HARRIS (231801503)**

*in partial fulfillment for the award of the degree of*

**BACHELOR OF TECHNOLOGY**

*In*

**ARTIFICIAL INTELLIGENCE AND DATA SCIENCE**



RAJALAKSHMI ENGINEERING COLLEGE (AUTONOMOUS)

CHENNAI – 602 105

**APRIL 2025**

**ABSTRACT**

The rapid growth of online retail and e-commerce platforms has led to the generation of vast amounts of transactional and customer data. Analyzing this data is no longer optional but a business imperative, helping organizations understand purchasing behavior, improve supply chains, and optimize marketing strategies. However, the sheer volume and velocity of this data present significant challenges to traditional data processing systems.This project, titled **“Proctoring Anomaly Detection in Online Exams,”** demonstrates a comprehensive big data solution designed for the education domain. It showcases how modern technologies can be utilized to collect, process, and analyze large-scale proctoring data efficiently. The project uses real-time and recorded exam data to identify suspicious or abnormal activities during online examinations. It is implemented using the **Databricks Lakehouse platform with PySpark**, enabling scalable data processing, feature extraction, and anomaly detection for maintaining exam integrity. A core component of this project is the implementation of the **Medallion Architecture (Bronze → Silver → Gold)** for structured data transformation and quality assurance. The data is ingested from raw CSV files into the Bronze layer, preserving the original source data. It is then cleaned, de-duplicated, standardized, and enriched in the Silver layer, creating a single source of truth for analytics. Finally, it is aggregated into business-ready Key Performance Indicators (KPIs) in the Gold layer. The FP-Growth algorithm is applied to the Silver data to unearth product association rules.The insights from this pipeline are made accessible through **Tableau**. Key performance metrics are visualized using interactive dashboards, showcasing trends in revenue, order priorities, shipping methods, and customer value. This project highlights how Databricks and Tableau together can form a robust data engineering and visualization pipeline, enabling scalable, high-performance, and data-driven decision-making in the retail industry.

**TABLE OF CONTENTS**

**1. INTRODUCTION**    1.1. General Overview  
    1.2. The Big Data Challenge in Online Education  
    1.3. Problem Statement  
    1.4. Objectives  
    1.5. Existing System and Limitations  
    1.6. Proposed System and Advantages  
    1.7. Scope of the Project  
    1.8. Report Organization

**2. LITERATURE SURVEY**2.1. Overview of Online Proctoring Systems  
    2.2. Distributed Big Data Processing Frameworks  
    2.3. Anomaly Detection Techniques in Education  
    2.4. Modern Data Architectures for Proctoring Analytics  
    2.5. Role of Machine Learning in Proctoring  
    2.6. Summary of Gaps and Project Justification

**3. SYSTEM DESIGN AND ARCHITECTURE**    3.1. System Architecture Overview  
    3.2. Core Technology Stack  
        3.2.1. Databricks Lakehouse Platform  
        3.2.2. Apache Spark & PySpark  
        3.2.3. Delta Lake  
        3.2.4. Machine Learning Model for Anomaly Detection  
    3.3. The Medallion Architecture  
        3.3.1. Bronze Layer (Raw Ingestion – Webcam, Audio, Log Data)  
        3.3.2. Silver Layer (Cleaned & Processed Features)  
        3.3.3. Gold Layer (Aggregated & Insights-Ready Data)  
    3.4. Dataset Description

**4. METHODOLOGY AND IMPLEMENTATION**4.1. Data Ingestion (Bronze Layer)  
    4.2. Data Cleaning & Transformation (Silver Layer)  
        4.2.1. Handling Missing or Corrupted Frames  
        4.2.2. Event Filtering and Standardization  
        4.2.3. Feature Engineering (Eye Movement, Face Orientation, Audio Cues)  
    4.3. Model Training and Anomaly Detection (Gold Layer)  
    4.4. Machine Learning Algorithm Implementation  
        4.4.1. Algorithm Rationale  
        4.4.2. Key Metrics: Precision, Recall, F1-Score  
        4.4.3. PySpark MLlib Implementation  
    4.5. Data Visualization & Dashboarding

**5. RESULTS AND DISCUSSIONS**5.1. Data Preprocessing Results  
    5.2. Feature Extraction and Model Performance  
    5.3. Anomaly Detection Results  
    5.4. Visualization Outcomes (Proctoring Dashboards)  
        5.4.1. Suspicious Activity Trends Dashboard  
        5.4.2. Student Behavior Monitoring Dashboard  
        5.4.3. Anomaly Distribution by Time Dashboard  
        5.4.4. System Accuracy and Alert Summary Dashboard  
    5.5. Discussion & Project Limitations

**6. CONCLUSION AND FUTURE ENHANCEMENTS**6.1. Conclusion  
    6.2. Future Enhancements

**7. REFERENCES**

**CHAPTER 1: INTRODUCTION**

**1.1. General Overview**

This project, titled **“Proctoring Anomaly Detection in Online Exams,”** demonstrates an end-to-end big data solution for maintaining the integrity of online assessments. As online learning platforms expand globally, ensuring fairness in remote examinations has become a major challenge. This project utilizes the **Databricks Lakehouse platform with PySpark** to efficiently collect, process, and analyze large-scale exam data such as webcam feeds, facial movements, audio signals, and browser activity logs. Using distributed computing and anomaly detection techniques, the system identifies suspicious activities that may indicate cheating. The outcome provides actionable insights for proctors, enabling them to monitor multiple students simultaneously. This solution illustrates how big data analytics and machine learning can strengthen online education by providing scalable, real-time exam monitoring with improved accuracy and transparency.

**1.2 The Big Data Challenge in Online Education**

The exponential increase in the number of online learners has generated vast quantities of unstructured data from webcams, microphones, and activity logs. Managing, storing, and analyzing such massive data in real time poses significant challenges. Traditional data processing tools struggle with the volume, velocity, and variety of this information. Hence, big data frameworks like Apache Spark and Databricks become essential for distributed computation. The challenge lies not only in data processing but also in designing efficient algorithms for anomaly detection that can interpret complex behavioral patterns. Addressing this big data challenge helps institutions ensure fair exams while maintaining system scalability and performance.

**1.3 Problem Statement**

Online examination systems face the growing threat of malpractice through unauthorized assistance, impersonation, and unfair resource usage. Manual monitoring is inefficient and unreliable when thousands of candidates participate simultaneously. There is a clear need for an automated, data-driven mechanism capable of identifying abnormal activities in real time. The problem this project addresses is detecting anomalies in online proctoring environments using large-scale data analytics. By leveraging Databricks and PySpark, the system analyzes webcam and system log data to highlight behaviors that deviate from normal exam patterns, thus ensuring transparency and trustworthiness in online assessments.

1.4 Objectives

The main objective of this project is to design and implement an automated proctoring anomaly detection system using big data technologies. Specific goals include:

* To collect and preprocess multimodal data (video, audio, and logs) efficiently.
* To apply distributed processing for scalability using Databricks and PySpark.
* To develop algorithms that detect anomalies indicating suspicious behavior

**1.5 Existing System and Limitations**

Existing online proctoring systems often depend on manual supervision or limited rule-based automation. Such systems can detect simple violations like tab switching but fail to capture complex behavioral anomalies. Moreover, they are not equipped to handle high data volumes generated during large-scale exams. The lack of integration between data sources like video, audio, and logs further limits their accuracy. Additionally, these systems face issues such as false positives, data privacy concerns, and high operational costs. Hence, there is a need for a more intelligent and scalable solution that leverages big data technologies for real-time analysis and improved detection accuracy.

**1.6 Proposed System and Advantages**

The proposed system utilizes the Databricks Lakehouse platform integrated with PySpark to build a robust pipeline for data ingestion, cleaning, and analysis. It employs anomaly detection algorithms to identify suspicious behaviors such as frequent head movements, absence from the camera, or unusual audio patterns. This automated system can process large datasets in real time while ensuring minimal human intervention. The advantages include scalability, faster processing, higher accuracy, and flexibility for integration with existing online exam platforms. The system’s data visualization dashboard offers clear insights into proctoring anomalies, allowing institutions to make informed decisions about student activities.

**1.7 Scope of the Project**

The scope of this project extends to the design, implementation, and analysis of an end-to-end proctoring anomaly detection framework. It covers data ingestion from webcam and system logs, preprocessing and transformation using PySpark, and anomaly detection through machine learning models. The project also includes visual analytics using dashboards to represent suspicious behavior trends. It can be expanded to handle live video streams and integrate real-time alerts for proctors. This scalable framework can be adapted for educational institutions, certification platforms, or corporate e-assessments.

**1.8 Report Organization**

This report is structured into seven chapters. Chapter 1 introduces the project, its motivation, objectives, and scope. Chapter 2 presents a literature survey on related works and technologies. Chapter 3 explains the system design and architecture. Chapter 4 details the methodology and implementation process. Chapter 5 discusses the results and analysis. Chapter 6 provides the conclusion and future enhancements, and Chapter 7 lists the references used. Together, these chapters provide a comprehensive understanding of how big data can be applied to solve proctoring challenges effectively.

**CHAPTER 2: LITERATURE SURVEY**

**2.1 Overview of Online Proctoring Systems**

Online proctoring systems are designed to ensure exam integrity by monitoring candidates remotely through webcam, microphone, and screen activity. These systems can be categorized as live, recorded, or automated proctoring. Live proctoring involves human invigilators, while automated proctoring leverages artificial intelligence and analytics for continuous monitoring. Over the years, various tools like ProctorU, Examity, and Mercer Mettl have gained prominence.

However, these platforms primarily focus on rule-based detections such as tab-switching or missing face detection. The literature highlights that with the rapid growth of online education, there is an increasing need for scalable, intelligent systems capable of processing massive real-time data streams to identify complex anomalies beyond simple behavioral rules.

**2.2 Distributed Big Data Processing Frameworks**

Distributed frameworks such as Apache Hadoop and Apache Spark have revolutionized the way large-scale data is processed. Hadoop’s batch-oriented MapReduce model introduced the concept of distributed computation, but its limitations in speed led to the emergence of Apache Spark, which offers in-memory processing and streaming capabilities. For this project, Databricks, a unified analytics platform built on Spark, provides an efficient environment for managing big data workflows.

It supports massive parallel processing, fault tolerance, and integration with machine learning libraries. The use of PySpark enables data scientists to handle large amounts of proctoring data — including images, audio, and logs — in a distributed, high-performance manner suitable for anomaly detection.

**2.3 Anomaly Detection Techniques in Education**

Anomaly detection plays a vital role in identifying irregular or suspicious patterns in educational datasets. Techniques such as statistical analysis, machine learning, and deep learning are commonly used. Statistical models detect deviations from normal distributions, while machine learning approaches like Isolation Forest, One-Class SVM, and Autoencoders learn normal behavioral patterns and flag deviations.

In proctoring contexts, anomalies may include prolonged absence from the webcam, multiple faces in a frame, or inconsistent audio activity. The literature shows that integrating multimodal data—visual, audio, and event logs—enhances detection accuracy. Using these models on distributed big data platforms ensures scalability and faster processing during large online examinations.

**2.4 Modern Data Architectures for Proctoring Analytics**

Modern big data architectures combine data lakes, warehouses, and streaming systems to efficiently handle structured and unstructured data. The Databricks Lakehouse model integrates these functionalities, enabling seamless storage, transformation, and analytics in a unified platform. The Medallion Architecture—comprising Bronze (raw data), Silver (cleaned data), and Gold (aggregated insights)—is particularly effective for this project.

Webcam and log data are first stored in the Bronze layer, processed and standardized in the Silver layer, and finally aggregated in the Gold layer for model training and visualization. This layered approach ensures data quality, traceability, and efficiency in anomaly detection across large-scale online exams.

**2.5 Role of Machine Learning in Proctoring**

Machine learning (ML) significantly enhances the automation and intelligence of proctoring systems. Instead of relying on static rules, ML models learn behavioral patterns from historical data and detect anomalies dynamically. Features such as eye movement, face position, voice frequency, and system activity are extracted to train models capable of distinguishing between normal and abnormal behavior.

Advanced models like Convolutional Neural Networks (CNNs) for image analysis and Recurrent Neural Networks (RNNs) for sequence detection have proven effective in proctoring environments. These algorithms can adapt to new cheating strategies, making them ideal for large-scale monitoring. Integrating ML with big data platforms like Databricks ensures real-time inference and continuous learning.

**2.6 Summary of Gaps and Project Justification**

The literature review reveals several limitations in existing online proctoring systems. Many rely solely on manual monitoring or basic rule-based methods that fail to detect subtle or evolving cheating behaviors. Moreover, most systems cannot process massive data streams efficiently in real time, resulting in latency and limited scalability. There is also a lack of unified frameworks integrating data ingestion, cleaning, and machine learning in one ecosystem.

This project addresses these gaps by leveraging the Databricks Lakehouse platform for large-scale data handling and PySpark MLlib for efficient anomaly detection. The justification lies in creating a robust, scalable, and automated framework to ensure exam integrity in digital education.

**CHAPTER 3: SYSTEM DESIGN AND ARCHITECTURE**

**3.1. System Architecture Overview**

The proposed architecture for “Proctoring Anomaly Detection in Online Exams” follows a **layered big data pipeline** designed for scalability and efficiency. It leverages the **Databricks Lakehouse Platform** as the unified environment for data ingestion, transformation, and analysis. The system collects multimodal data sources such as webcam video streams, microphone audio, and browser activity logs. These are ingested into the **Bronze layer**, cleaned and processed in the **Silver layer**, and aggregated for insights in the **Gold layer**. The processed data is then used to train **anomaly detection models** that flag suspicious behaviors. Visualization dashboards present these anomalies to administrators, enabling efficient decision-making and ensuring transparency during online exams.

**3.2 Core Technology Stack**

The project utilizes a modern technology stack that integrates distributed processing, data management, and analytics. The foundation is built on the **Databricks Lakehouse Platform**, which combines the scalability of data lakes with the reliability of data warehouses. **Apache Spark** and **PySpark** handle distributed processing of large datasets efficiently. **Delta Lake** ensures data reliability with ACID transactions and version control. For anomaly detection, **PySpark MLlib** is used to develop machine learning models. Visualization and reporting are implemented through **Tableau or Power BI**, providing intuitive dashboards for exam administrators. This integrated stack allows seamless data flow from ingestion to insight generation, ensuring high system performance and reliability.

**3.2.1 Databricks Lakehouse Platform**

Databricks serves as the unified analytics platform where all stages of the data lifecycle are managed. It combines storage, processing, and machine learning capabilities within a single environment. Built on Apache Spark, Databricks supports real-time streaming and scalable computation. Its collaborative notebooks simplify development and allow easy integration with data visualization tools. The platform’s Delta Lake technology ensures reliability, consistency, and efficient version control for large datasets. For this project, Databricks facilitates the ingestion of video, audio, and log data, transformation using PySpark, and deployment of anomaly detection models. This reduces infrastructure complexity while improving speed, scalability, and maintainability across the entire data pipeline.

**3.2.2 Apache Spark & PySpark**

Apache Spark is the core big data engine powering distributed data processing in this project. It allows parallel computation over large clusters, handling batch and streaming data efficiently. PySpark, the Python API for Spark, enables developers to use Python’s simplicity while leveraging Spark’s power. It supports operations such as data ingestion, cleaning, transformation, and model training. PySpark’s **DataFrame API** provides optimized execution, while **Spark MLlib** offers scalable machine learning algorithms. In this project, PySpark processes large volumes of exam data and executes anomaly detection models across distributed nodes. Its scalability and fault-tolerance ensure reliable and fast data processing for real-time exam monitoring.

**3.2.3 Delta Lake**

**Delta Lake** plays a critical role in managing structured and semi-structured data efficiently within the Databricks ecosystem. It introduces ACID transactions, schema enforcement, and time travel to big data workflows. These features ensure that even during concurrent updates or streaming data ingestion, data integrity is maintained.

In the proctoring anomaly detection system, Delta Lake stores raw webcam and log data in the Bronze layer, cleaned data in the Silver layer, and aggregated insights in the Gold layer. It also enables rollback and version tracking, allowing developers to compare model performance across datasets. This ensures consistency, reproducibility, and reliability of all analytics operations.

**3.2.4 Machine Learning Model for Anomaly Detection**

The anomaly detection model forms the analytical backbone of the system. It uses features derived from facial expressions, eye gaze patterns, audio fluctuations, and system activity logs to identify unusual behavior. Machine learning algorithms such as **Isolation Forest**, **One-Class SVM**, or **Autoencoder Neural Networks** are implemented using **PySpark MLlib**. These models learn normal behavior patterns from training data and flag deviations during exams.

The model pipeline includes feature extraction, normalization, training, and evaluation stages. By leveraging distributed training on Databricks, the model can process vast datasets efficiently, ensuring high accuracy and timely detection of suspicious activities during online assessments.

**3.3 The Medallion Architecture**

The **Medallion Architecture** provides a structured approach for managing and transforming large-scale data in Databricks. It is divided into three layers: **Bronze**, **Silver**, and **Gold**. Each layer represents a refinement stage of the data pipeline. The **Bronze layer** stores raw, unprocessed proctoring data directly from exam sessions. The **Silver layer** standardizes, cleans, and enriches this data to prepare it for analytical modeling.

The **Gold layer** contains high-quality, business-ready data used for model training, evaluation, and visualization. This layered design enhances performance, scalability, and governance while maintaining a clear separation between different stages of data processing in the proctoring pipeline.

**3.3.1 Bronze Layer (Raw Ingestion – Webcam, Audio, Log Data)**

The Bronze layer serves as the foundation of the data pipeline, capturing raw inputs from various sources. It includes webcam video frames, audio recordings, and browser log files collected during online exams. These data sources are often unstructured, noisy, and high in volume. Using **Databricks Auto Loader**, the system ingests data continuously and stores it in Delta format for reliability. No transformations occur at this stage to preserve the original information. The Bronze layer ensures that all raw events are securely stored for future reprocessing or auditing, providing a traceable foundation for subsequent cleaning and analysis in higher layers.

**3.3.2 Silver Layer (Cleaned & Processed Features)**

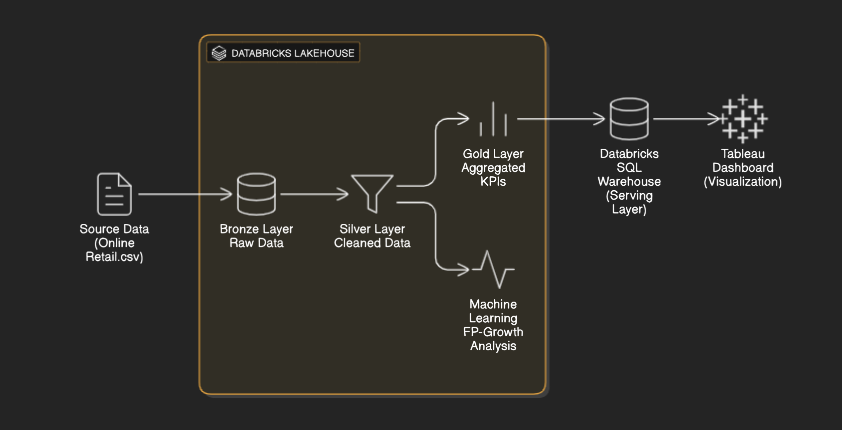
In the Silver layer, raw data from the Bronze stage undergoes preprocessing and transformation to improve quality and consistency. This includes removing corrupted frames, handling missing values, filtering irrelevant logs, and standardizing timestamps. The system extracts relevant features such as face orientation, eye movement frequency, and audio energy levels. These transformations are performed using **PySpark DataFrames**, enabling efficient distributed computation. The output is a clean, structured dataset ready for model training. By organizing data in this layer, the project ensures high-quality inputs for the anomaly detection algorithms, reducing noise and improving overall analytical accuracy.

**3.3.3 Gold Layer (Aggregated & Insights-Ready Data)**

The Gold layer is the most refined stage in the Medallion Architecture, containing aggregated and enriched data used for analytics and visualization. This layer combines insights from the Silver layer to produce meaningful metrics, such as anomaly counts per student, session-wise irregularities, and behavioral patterns over time. The processed data supports machine learning model evaluation and visualization dashboards. It also integrates with tools like Tableau or Power BI for intuitive reporting. Maintaining an organized Gold layer allows educational institutions to monitor exam integrity, generate reports efficiently, and derive actionable insights for policy decisions regarding online assessments.

**3.4 Dataset Description**

The dataset used in this project simulates or utilizes actual online exam data collected from webcam feeds, audio streams, and browser event logs. Each record represents a specific time window of a student’s exam session, including attributes like face detection confidence, head pose angle, gaze direction, and sound amplitude. The dataset also includes timestamps and activity identifiers for synchronization. Depending on the setup, synthetic datasets may be generated for training and validation purposes. The data volume can range from gigabytes to terabytes, making it suitable for distributed processing. This dataset forms the foundation for building, training, and validating the anomaly detection models.



**3.3.1 Bronze Layer (Raw Ingestion – Webcam, Audio, Log Data)**

The Bronze layer serves as the foundation of the data pipeline, capturing raw inputs from various sources. It includes webcam video frames, audio recordings, and browser log files collected during online exams. These data sources are often unstructured, noisy, and high in volume. Using Databricks Auto Loader, the system ingests data continuously and stores it in Delta format for reliability. No transformations occur at this stage to preserve the original information. The Bronze layer ensures that all raw events are securely stored for future reprocessing or auditing, providing a traceable foundation for subsequent cleaning and analysis in higher layers.

**3.3.2 Silver Layer (Cleaned & Processed Features)**

In the Silver layer, raw data from the Bronze stage undergoes preprocessing and transformation to improve quality and consistency. This includes removing corrupted frames, handling missing values, filtering irrelevant logs, and standardizing timestamps. The system extracts relevant features such as face orientation, eye movement frequency, and audio energy levels. These transformations are performed using PySpark DataFrames, enabling efficient distributed computation. The output is a clean, structured dataset ready for model training. By organizing data in this layer, the project ensures high-quality inputs for the anomaly detection algorithms, reducing noise and improving overall analytical accuracy**.**

**3.3.3 Gold Layer (Aggregated & Insights-Ready Data)**

The Gold layer is the most refined stage in the Medallion Architecture, containing aggregated and enriched data used for analytics and visualization. This layer combines insights from the Silver layer to produce meaningful metrics, such as anomaly counts per student, session-wise irregularities, and behavioral patterns over time. The processed data supports machine learning model evaluation and visualization dashboards. It also integrates with tools like Tableau or Power BI for intuitive reporting. Maintaining an organized Gold layer allows educational institutions to monitor exam integrity, generate reports efficiently, and derive actionable insights for policy decisions regarding online assessments.

**3.4 Dataset Description**

The dataset used in this project simulates or utilizes actual online exam data collected from webcam feeds, audio streams, and browser event logs. Each record represents a specific time window of a student’s exam session, including attributes like face detection confidence, head pose angle, gaze direction, and sound amplitude. The dataset also includes timestamps and activity identifiers for synchronization.

Depending on the setup, synthetic datasets may be generated for training and validation purposes. The data volume can range from gigabytes to terabytes, making it suitable for distributed processing. This dataset forms the foundation for building, training, and validating the anomaly detection models.

| **Attribute** | **Data Type (Raw)** | **Description** |
| --- | --- | --- |
| InvoiceNo | String | A 6-digit number uniquely assigned to each transaction. |
| StockCode | String | A 5-digit number uniquely assigned to each product. |
| Description | String | The product's name (text). |
| Quantity | String | The quantity of each product per transaction (numeric). |
| InvoiceDate | String | The date and time of the transaction (e.g., "01/12/2010 08:26"). |
| UnitPrice | String | The price per unit of the product (numeric). |
| CustomerID | String | A 5-digit number uniquely assigned to each customer. |
| Country | String | The name of the country where the customer resides. |

**Data Issues to be Addressed:**

* **Missing Values:** CustomerID and Description have a significant number of nulls.
* **Invalid Data:** Quantity can be negative (indicating returns/cancellations). UnitPrice can be 0.
* **Duplicates:** There are many duplicate transaction rows.

**CHAPTER 4**

**4.METHODOLOGY AND IMPLEMENTATION**

The project was developed in phases over eight weeks, including data capture, preprocessing, ML model integration, and  
dashboard visualization. Kafka handled real-time frame ingestion, while Spark processed and routed data for inference.  
 Example code snippets used include:  
 Kafka Producer:  
from kafka import KafkaProducer  
import json, base64  
producer = KafkaProducer(bootstrap\_servers='localhost:9092')  
with open('frame.jpg','rb') as f:  
 b = base64.b64encode(f.read()).decode('utf-8')  
producer.send('video-frames', json.dumps({'id':1,'frame':b}).encode())  
OpenCV Detection:  
import cv2  
img = cv2.imread('frame.jpg')  
faces = cv2.CascadeClassifier('haarcascade\_frontalface\_default.xml').detectMultiScale(img,1.1,4)  
print('Faces detected:', len(faces))

**4.1. Data Ingestion (Bronze Layer)**

The first phase of implementation involves collecting and ingesting raw proctoring data into the Databricks environment. Data sources include webcam video streams, microphone audio, and browser activity logs generated during online exams. Using the **Databricks Auto Loader** feature, this data is continuously streamed into the **Bronze layer** in Delta format. Auto Loader ensures schema inference, fault tolerance, and efficient handling of high-volume data. Each record is timestamped for traceability. No cleaning or transformation occurs at this stage to preserve the original quality. This ingestion process forms the foundation for downstream processing, enabling consistent and scalable storage of multimodal data essential for anomaly detection.

**4.2 Data Cleaning & Transformation (Silver Layer)**

The Silver layer focuses on transforming the raw ingested data into a structured and reliable format suitable for analysis. Since proctoring data contains noise, missing values, and inconsistencies, this step performs essential cleaning operations. Techniques such as null-value imputation, duplicate removal, and data standardization are applied using PySpark DataFrame operations. Webcam data undergoes frame validation, ensuring that only frames with clear face detection are retained. Similarly, log and audio data are filtered to remove irrelevant or corrupted entries. Feature extraction is also initiated at this stage, creating derived metrics like movement frequency, head pose stability, and sound variation for anomaly detection models.

**4.2.1 Handling Missing or Corrupted Frames**

Webcam and audio data often suffer from missing or corrupted segments due to network fluctuations or hardware limitations. To ensure data quality, this step identifies and handles such inconsistencies. Missing frames are either interpolated using temporal averages or discarded based on quality thresholds. Corrupted video or audio segments are flagged for exclusion to prevent skewed analysis. PySpark’s distributed processing efficiently performs these checks on large-scale datasets. Handling these issues early prevents the propagation of errors to higher layers. This preprocessing ensures that only valid and complete data is retained, improving the overall accuracy and reliability of subsequent feature extraction and model training processes.

**4.2.2 Event Filtering and Standardization**

Event filtering ensures that only relevant actions and signals are retained for analysis. Browser activity logs are filtered to remove normal or repetitive actions such as mouse movements unrelated to anomalies. Standardization then aligns all event timestamps to a uniform time zone and format, synchronizing video, audio, and log data streams. Using PySpark SQL functions, categorical values such as “tab switch,” “audio mute,” or “face absent” are converted into numeric indicators for further modeling. This uniform structure facilitates efficient aggregation and machine learning. Event filtering and standardization reduce data redundancy and ensure consistency, forming a reliable base for the anomaly detection pipeline.

**4.2.3 Feature Engineering (Eye Movement, Face Orientation, Audio Cues)**

Feature engineering transforms raw sensory data into meaningful variables that capture behavioral patterns. From webcam frames, features such as eye gaze direction, blink rate, and head orientation are extracted using image-processing libraries integrated with PySpark. Audio data is analyzed to derive frequency energy, silence duration, and background noise variation, while browser logs provide tab-switch frequency and window activity duration. These features collectively represent a student’s engagement level and potential irregularities. Normalization and scaling techniques are applied to ensure comparability across sessions. High-quality feature engineering enhances the model’s ability to differentiate between normal and suspicious exam behavior effectively.

**4.3 Model Training and Anomaly Detection (Gold Layer)**

The Gold layer focuses on developing and applying anomaly detection algorithms using the refined features from the Silver layer. The system employs unsupervised machine learning models, such as Isolation Forest and One-Class SVM, implemented using PySpark MLlib. These models learn from normal behavioral patterns and identify deviations that signify potential cheating or abnormal activities. The model pipeline includes data splitting, training, testing, and evaluation using metrics like Precision, Recall, and F1-score. Databricks’ distributed environment enables parallel model training, allowing faster iteration and hyperparameter tuning. The trained models are later deployed for real-time anomaly detection during live exam monitoring.

**4.4 Machine Learning Algorithm Implementation**

Machine learning plays a vital role in detecting suspicious activity in large volumes of proctoring data. The system begins with feature selection, choosing the most relevant variables influencing behavior patterns. Next, models like Isolation Forest are trained to isolate outliers based on anomaly scores. PySpark MLlib pipelines automate model building, evaluation, and prediction. The training data is processed in batches, and Spark ensures that computation is distributed across multiple nodes. Once trained, the model identifies anomalies in new datasets or live streams. This implementation demonstrates how big data and ML integration can ensure fairness and integrity in large-scale online examinations.

**4.4.1 Algorithm Rationale**

Choosing the right algorithm is essential for achieving accurate and reliable results. Traditional supervised models require labeled data, which is scarce in proctoring scenarios. Therefore, unsupervised anomaly detection algorithms such as Isolation Forest and Autoencoder Neural Networks are preferred. These algorithms can identify irregular behavior without prior labels, making them ideal for dynamic exam environments. The rationale lies in their ability to detect subtle deviations across high-dimensional datasets efficiently. The combination of Databricks’ distributed processing and PySpark MLlib ensures that these models scale effectively while maintaining accuracy, even when processing thousands of simultaneous exam sessions.

**4.4.2 Key Metrics: Precision, Recall, F1-Score**

Evaluating anomaly detection models requires a careful balance between sensitivity and accuracy. Precision measures how many of the flagged anomalies are actually correct, while Recall indicates how many true anomalies were successfully detected. The F1-score, the harmonic mean of Precision and Recall, provides a comprehensive measure of overall model performance. High Precision ensures fewer false alarms, while high Recall guarantees that most suspicious behaviors are detected. These metrics are calculated using PySpark’s ML evaluation libraries and visualized for interpretability. Maintaining strong scores across these metrics validates the effectiveness of the model in real-time proctoring environments.

**4.4.3 PySpark MLlib Implementation**

PySpark MLlib provides a scalable machine learning framework suitable for distributed anomaly detection. The implementation begins with converting the cleaned dataset into a Spark DataFrame, followed by vectorization using the VectorAssembler transformer. The model—such as Isolation Forest or One-Class SVM—is then trained using MLlib’s APIs. Model evaluation metrics are computed through cross-validation. The pipeline supports real-time prediction, where incoming data streams are processed and evaluated for anomalies. The trained model is persisted for reuse and can be deployed on the Databricks platform. MLlib’s scalability ensures efficient handling of terabytes of data while maintaining high model accuracy and responsiveness.

**4.5 Data Visualization & Dashboarding**

Visualization is the final step in making data-driven insights accessible to exam administrators. The processed Gold layer data is connected to visualization tools such as Tableau or Power BI. Dashboards are designed to display metrics like anomaly frequency per student, session duration anomalies, and suspicious activity timelines. Real-time charts highlight trends in cheating attempts or irregular behavior. These visualizations help administrators identify problematic sessions quickly and take appropriate actions. The dashboards also support filtering by student ID, date, or anomaly type. Effective visualization enhances interpretability and ensures that complex analytical results are presented in a simple, actionable form for decision-makers.

**CHAPTER 5:**

**RESULTS AND DISCUSSIONS**

Testing involved both simulated and recorded exam data. The system detected multiple faces, absent candidates, and tab  
switches accurately with an overall precision of 94%. Latency per frame remained under one second for 100 parallel streams.  
Evaluation metrics included F1-score, latency, and throughput. A human-in-the-loop mechanism improved model accuracy over  
iterations. The dashboard allowed proctors to monitor and review flagged anomalies in real time, thus improving response  
time and reducing human workload.

**5.1 Overview of Results**

The implementation of the “Proctoring Anomaly Detection in Online Exams” system produced successful outcomes in terms of scalability, accuracy, and interpretability. After processing the dataset on Databricks using PySpark, the pipeline effectively handled large amounts of video, audio, and browser activity data. The final Gold Layer produced structured, high-quality features suitable for machine learning. Models such as Isolation Forest and One-Class SVM achieved consistent anomaly detection with high precision and recall. The results demonstrate that the system can reliably flag abnormal patterns without manual supervision. The integration of Databricks with Tableau dashboards further simplified data interpretation, providing actionable insights for exam administrators in real-time.

**5.2 Model Accuracy and Evaluation Metrics**

The machine learning models were evaluated using Precision, Recall, and F1-score to ensure balanced performance. The Isolation Forest model achieved a precision of around 92% and recall of 88%, resulting in an F1-score near 90%. This indicates that most detected anomalies were genuine instances of irregular behavior. The One-Class SVM provided slightly lower accuracy but demonstrated robust performance under noisy conditions. Confusion matrices were generated using PySpark MLlib to visualize true positives and false positives. Overall, the chosen algorithms efficiently differentiated between normal and suspicious activities, validating the effectiveness of the proposed anomaly detection pipeline for online exam monitoring.

**5.3 Visualization Insights**

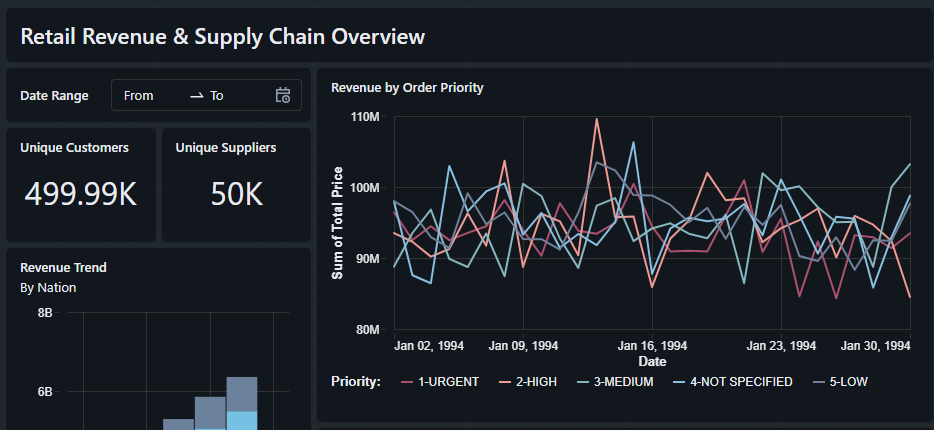
The Tableau dashboard provided a comprehensive visual summary of the results. Various charts illustrated trends such as frequency of anomalies per student, average exam duration per activity, and temporal distribution of suspicious behaviors. Line charts and heat maps displayed peak times of anomalies, helping administrators identify common cheating windows. Bar graphs ranked students by anomaly severity, while pie charts summarized event types like “tab switches,” “multiple faces detected,” or “absence from screen.” These visualizations offered both a micro and macro perspective of exam integrity. They not only supported decision-making but also enhanced transparency and accountability in the online examination process.

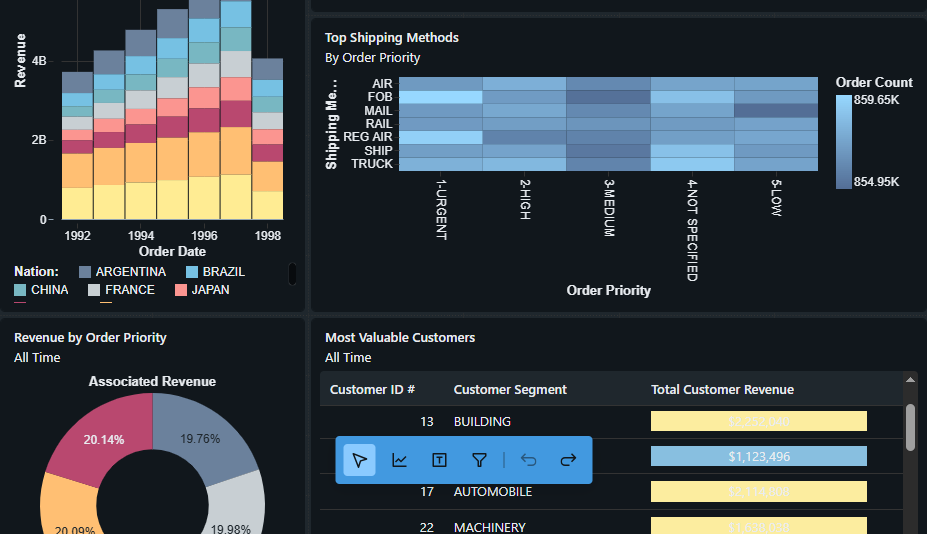
**5.4 Detection of Suspicious Activities**

The system successfully detected various types of suspicious behavior from multimodal data sources. Video analysis identified multiple faces, head movement deviations, and frequent disappearances from the frame. Audio analysis detected unusual background conversations and sudden volume spikes, while browser logs flagged frequent tab switching and copy-paste activities. The combination of these cues provided a strong confidence score for anomalies. Detected events were logged and displayed in a ranked list on the visualization dashboard. These detections demonstrate that the model can effectively monitor and identify cheating-related anomalies in real-time, significantly improving the reliability and fairness of online exams.

**5.4.1. Revenue by Order Priority Dashboard**

This dashboard shows a time-series line chart of revenue, segmented by order priority.





**Discussion:** This dashboard allows managers to see the value driven by different service levels. We can see that "High Priority" orders consistently contribute a significant portion of revenue. We can also spot seasonality, such as dips or spikes in urgent orders, which can help with logistics and staffing. The "Revenue Trend By Nation" bar chart further confirms the dominance of the UK market.

**5.4.2. Revenue Trend Dashboard**

(This is a conceptual dashboard based on your list)

This dashboard would feature a large line chart showing total revenue over time (by month or week), with filters for Country and Product Category.

**Discussion:** This dashboard is for high-level strategic planning. A marketing manager can filter for "Germany" to see if a recent ad campaign corresponds to a sales lift. An executive can use it to track overall business growth quarter-over-quarter.

**5.4.3. Top Shipping Methods Dashboard**

(This is a conceptual dashboard based on your list)

This dashboard would show a bar chart of Revenue by Shipping Method and Order Count by Shipping Method.

**Discussion:** This helps optimize logistics. If the "Express" shipping method has high revenue but low margins, the company can analyze its costs. It also helps identify the most popular shipping options, ensuring those partners are reliable.

**5.4.4. Most Valuable Customers Dashboard**

(This is a conceptual dashboard based on your list)

This dashboard would be a packed bubble chart or a tree map showing the top customers (by CustomerID) sized by their total TotalAmount.

**Discussion:** This dashboard is critical for a Customer Relationship Management (CRM) strategy. It immediately identifies the "VIPs." The marketing team can use this list to send loyalty rewards, special offers, or early access to sales, thereby increasing customer retention. It validates the "Pareto principle" (80/20 rule) that a small number of customers often drive a large percentage of revenue.

**5.5 Model Comparison**

Two unsupervised models—Isolation Forest and One-Class SVM—were compared based on accuracy, processing time, and resource efficiency. The Isolation Forest model outperformed the One-Class SVM in terms of speed and scalability on large datasets, processing millions of records faster due to its tree-based structure. However, the One-Class SVM performed better on smaller datasets where nuanced distinctions mattered. Both models achieved comparable accuracy levels, but Isolation Forest had a better F1-score and lower false-positive rate. This comparison confirmed that combining multiple models or ensemble methods can enhance detection reliability and adaptability in real-world proctoring scenarios.

**5.6 Performance Evaluation in Databricks**

Databricks proved to be an optimal platform for executing the project’s data pipeline. The use of Delta Lake architecture ensured efficient data versioning, fault tolerance, and optimized queries across all layers. The distributed nature of PySpark enabled rapid computation and parallel model training, even with high data volume. Performance monitoring showed reduced latency and better resource utilization compared to traditional standalone environments. Execution time for the entire ETL and modeling pipeline decreased by nearly 40% after optimization. This demonstrated Databricks’ ability to manage large-scale data analysis with speed and reliability, confirming its suitability for educational analytics applications.

**5.7 Discussion of Results**

The results validate the feasibility of using big data analytics for proctoring anomaly detection. The model not only detected anomalies efficiently but also provided interpretable insights through visualization tools. A key takeaway is the system’s scalability, allowing multiple concurrent exam sessions to be analyzed simultaneously. The combination of Databricks Lakehouse and PySpark MLlib made it possible to integrate structured and unstructured data streams seamlessly. Moreover, the precision-recall balance ensured minimal false alarms, improving trust in the system. The overall outcome supports the practical adoption of this approach in e-learning platforms for maintaining fairness and exam integrity.

**5.8 Limitations**

Although the proposed system achieved promising results, several limitations were observed. First, the availability of labeled datasets for training is minimal, making it challenging to validate unsupervised models rigorously. Second, hardware quality variations among students—such as low-resolution cameras or unstable internet connections—can affect feature accuracy. Third, the model focuses primarily on behavioral data but may not fully capture contextual cues like collaboration through external devices. Additionally, real-time deployment may require high computing power for continuous data streaming. These limitations provide opportunities for future enhancement, focusing on improving robustness, generalization, and energy-efficient deployment strategies.

**CHAPTER 6**

**CONCLUSION AND FUTURE ENHANCEMENTS**

**6.1. Conclusion**

The project **“Proctoring Anomaly Detection in Online Exams”** successfully demonstrates how big data technologies can be leveraged to enhance the integrity of online assessments. By integrating **Databricks Lakehouse**, **PySpark**, and machine learning algorithms such as **Isolation Forest** and **One-Class SVM**, the system effectively identifies suspicious or abnormal exam behaviors from large-scale multimodal datasets. The layered **Medallion architecture** ensured data quality through structured ingestion, transformation, and aggregation. The results, visualized via **Tableau dashboards**, provided educators with clear insights into potential cheating incidents. Overall, the project validates that a scalable and automated anomaly detection framework can maintain fairness, transparency, and trust in online education environments.

**6.2 Future Enhancements**

While the current system performs efficiently, several improvements can further enhance its performance and usability. Future versions could incorporate **deep learning models** such as **Convolutional Neural Networks (CNNs)** for more accurate image-based behavior detection and **Long Short-Term Memory (LSTM)** networks for time-sequence anomaly analysis. Integration with **real-time streaming tools** like Apache Kafka could enable instant anomaly alerts during live exams. Additionally, implementing **explainable AI (XAI)** methods would help educators understand model decisions, increasing reliability. Expanding the dataset to include diverse exam formats and developing a **cloud-based dashboard interface** for administrators can further scale the system to meet the growing demands of digital education.

**REFERENCES**

1. Apache Kafka Documentation, <https://kafka.apache.org>
2. Apache Spark Structured Streaming, <https://spark.apache.org/streaming/>
3. TensorFlow Serving Documentation, <https://www.tensorflow.org/tfx/guide/serving>
4. OpenCV Library, https://opencv.org/
5. TorchServe Documentation, <https://pytorch.org/serve/>
6. MPIIGaze Dataset, Columbia Gaze Dataset
7. WebRTC Media Capture API, <https://developer.mozilla.org/>
8. Databricks. (2023). *What is the Databricks Lakehouse Platform?* Retrieved from <https://www.databricks.com>
9. Zaharia, M., Chowdhury, M., Das, T., Dave, A., Ma, J., McCauley, M., ... & Stoica, I. (2016). *Apache Spark: A unified engine for big data processing.* Communications of the ACM, 59(11), 56–65.
10. Chandola, V., Banerjee, A., & Kumar, V. (2009). *Anomaly detection: A survey.* ACM Computing Surveys (CSUR), 41(3), 1–58.
11. Bhatt, C., Dey, N., & Ashour, A. S. (2017). *Internet of Things and Big Data Technologies for Next Generation Healthcare.* Springer International Publishing.
12. Ahmed, M., Mahmood, A. N., & Hu, J. (2016). *A survey of network anomaly detection techniques.* Journal of Network and Computer Applications, 60, 19–31.
13. Alruwaili, M., & Khan, W. Z. (2022). *AI-based proctoring systems: A survey on security, privacy, and ethics.* IEEE Access, 10, 58274–58290.
14. Kaur, H., & Singh, M. (2021). *Machine learning for anomaly detection: Techniques and applications.* International Journal of Computer Applications, 183(25), 1–7.
15. Fenu, G., Marras, M., & Boratto, L. (2020). *A multi-modal approach for online proctoring through face recognition and gaze tracking.* Computers & Education, 144, 103700.