A black and white logo

AI-generated content may be incorrect. A blue and gold logo

AI-generated content may be incorrect. A blue text on a white background

AI-generated content may be incorrect.

The capstone project represents the culmination of our Data engineering bootcamp, harnessing the skills we’ve developed to deliver a practical, impactful data solution. Undertaken by Abdullah Khan, Mugdha Patil, and Swati Sahu, this project introduces a Real-Time Seismic Activity Monitoring System designed to address the challenges of processing and integrating seismic data for real-time decision-making. Earthquakes and seismic events pose significant risks to human life, infrastructure, and economies globally. While existing systems provide raw data, efficiently transforming this information into actionable insights remains a complex task. Our objective is to bridge this gap by building an end-to-end data pipeline that ingests, processes, and visualizes seismic and demographic data, delivering timely alerts and risk assessments to a range of stakeholders.

To achieve this, we leverage multiple data sources: near real-time seismic data from the United States Geological Survey (USGS) Earthquake Hazards Program and the Incorporated Research Institutions for Seismology (IRIS) Seismic Data APIs, complemented by static population density data from WorldPop (India, 2020). The USGS and IRIS APIs provide dynamic event data—exceeding 1 million rows over time—with metadata such as magnitude, geolocation, depth, and timestamps, while the WorldPop dataset enriches our analysis with spatial population insights critical for risk evaluation. Utilizing Databricks as our core platform, we aim to construct a scalable ETL pipeline, perform robust data quality checks, and develop an interactive dashboard. The expected deliverables include a comprehensive data model, automated workflows, and a visualization interface tailored for disaster management agencies, seismologists, governments, and the public. This report chronicles our process—from problem definition to deployment—detailing our dataset and technology choices, methodology, challenges, and opportunities for future enhancement, while demonstrating our capability to manage diverse, large-scale datasets in an original and meaningful way.

**Features:**

* **Over 3 million rows of source data extracted**, with more than 1 million records each from the USGS Earthquake Hazards Program API, IRIS Seismic Data API, and WorldPop population density dataset (India, 2020), combining real-time seismic events with static demographic data for comprehensive analysis.
* A **Databricks Workflow** orchestrating the end-to-end data pipeline, scheduling Directed Acyclic Graphs (DAGs) within a scalable Databricks environment to automate ingestion, transformation, and loading processes seamlessly.
* **Data visualization of key performance indicators (KPIs) and use cases**, including earthquake magnitude trends, geographic event distribution, and population exposure risks, delivered through interactive Power BI dashboards for real-time monitoring and decision support.
* **Apache Spark jobs within Databricks**, leveraging Spark’s distributed computing power to process high-frequency seismic data and integrate geospatial population insights efficiently, ensuring scalability and performance.
* **Medallion architecture data design patterns**, organizing data into Bronze (raw), Silver (cleaned), and Gold (aggregated) layers within Databricks Delta Lake, optimizing storage, quality, and accessibility for downstream analytics.
* **Comprehensive architecture diagram**, depicting the data flow from ingestion through processing to visualization, spanning USGS, IRIS, and WorldPop sources.
* **Comprehensive data dictionary**, detailing all data sources and transformations applied, providing a clear audit trail for reproducibility and validation.

**Capstone Requirements**

1. Identify a problem you'd like to solve.
2. Scope the project and choose datasets.
3. Use at least 2 different data sources and formats (e.g., CSV, JSON, APIs), with at least 1 million rows.
4. Define your use case: analytics table, relational database, dashboard, etc.
5. Choose your tech stack and justify your choices.
6. Explore and assess the data.
7. Identify and resolve data quality issues.
8. Document your data cleaning steps.
9. Create a data model diagram and dictionary.
10. Build ETL pipelines to transform and load the data.
11. Run pipelines and perform data quality checks.
12. Visualize or present the data through dashboards, applications, or reports.
13. Your data pipeline must be running in a production cloud environment

**Problem Statement**

Earthquakes and seismic activities present persistent threats to human life, infrastructure, and economic stability worldwide, with devastating impacts often amplified by delayed detection and inadequate risk assessment. While real-time seismic data is available through sources like the USGS Earthquake Hazards Program and IRIS Seismic Data APIs, and demographic insights can be derived from datasets like WorldPop, the integration and processing of these diverse, high-volume data streams into a cohesive, actionable system remain significant challenges. Current monitoring solutions often struggle with data latency, quality inconsistencies, and the inability to correlate seismic events with population exposure in real time, limiting their effectiveness for rapid response and planning. This project aims to address these gaps by developing a Real-Time Seismic Activity Monitoring System using Databricks, capable of ingesting over 3 million records from multiple sources, transforming them via a scalable ETL pipeline, and delivering visualized insights through Power BI to empower disaster management agencies, seismologists, governments, and the public with timely, data-driven decision-making tools.

**Use Cases**

1. **Real-Time Earthquake Alerts**: Notify disaster management agencies of high-magnitude events (>5.0 Richter scale) with location and affected population estimates for rapid response.
2. **Seismic Risk Mapping**: Visualize earthquake-prone areas overlaid with population density from WorldPop to identify high-risk zones for city planners.
3. **Historical Trend Analysis**: Enable seismologists to study event frequency, magnitude, and depth patterns over time for research and forecasting.
4. **Resource Allocation Planning**: Provide governments with data on seismic activity and population exposure to prioritize infrastructure reinforcement and emergency preparedness.
5. **Public Safety Notifications**: Display real-time seismic activity updates on Power BI dashboards for public awareness and evacuation guidance.
6. **Data Quality Validation**: Monitor and flag incomplete or duplicate seismic records (e.g., missing depth) for system reliability.
7. **Anomaly Detection**: Identify unusual seismic patterns (e.g., sudden increases in low-magnitude events) for early warning of potential larger quakes.
8. **Cross-Regional Impact Assessment**: Combine USGS/IRIS event data with WorldPop to assess multi-region seismic impacts for coordinated disaster response.
9. **Performance Benchmarking**: Track ETL pipeline efficiency and visualization latency to ensure system scalability under high event volumes.
10. **Post-Event Analysis**: Generate reports on event aftermath (e.g., affected population, magnitude distribution) for debriefing and future planning.

**Data Sources**

* **USGS Earthquake Hazards Program API**
  + Description: Provides near real-time seismic event data, including magnitude, geolocation (latitude/longitude), depth, and timestamps, sourced from a global network of seismometers.
  + Format: JSON via API.
  + Link: <https://earthquake.usgs.gov/fdsnws/event/1/>
  + Volume: Over 1 million records in raw data.
* **IRIS Seismic Data API**
  + Description: Delivers real-time seismic data from the Incorporated Research Institutions for Seismology, offering detailed event metadata such as waveforms, station locations, and event types.
  + Format: JSON or other API-supported formats.
  + Link: [https://service.iris.edu/clients/](https://service.iris.edu/clients/?utm_source=chatgpt.com)
  + Volume: Over 1 million records in raw data.
* **WorldPop Population Density Dataset (India, 2020)**
  + Description: Supplies static geospatial population density data for India, enabling risk assessment by mapping seismic events to affected populations at a 100m resolution.
  + Format: Raster (e.g., GeoTIFF) or tabular (e.g., CSV after processing)
  + Link: <https://hub.worldpop.org/geodata/summary?id=6545>
  + Volume: Over 1 million records in raw data when converted to row-based format.

**Technologies Used:**

* **Python**: Programming language used for scripting Spark jobs within Databricks, handling API calls to USGS and IRIS, and processing WorldPop raster data into usable formats.
* **SQL**: Essential for querying, transforming, and aggregating structured seismic and population data within Databricks SQL and Power BI.
* **Databricks**: Unified analytics platform hosting the end-to-end pipeline, including Databricks Workflows for scheduling DAGs, data ingestion, transformation, and storage management.
* **Apache Spark**: Open-source distributed computing system within Databricks, powering large-scale processing of over 3 million rows from USGS, IRIS, and WorldPop sources with high performance and scalability.
* **Databricks Delta Lake**: Data lake technology supporting the Medallion architecture (Bronze, Silver, Gold layers), ensuring data reliability, versioning, and quality for seismic and population datasets.
* **Power BI**: Visualization tool for creating interactive dashboards, displaying 10 KPIs and use cases such as earthquake magnitude trends and population exposure risks.
* **REST APIs**: Interface for real-time data extraction from USGS Earthquake Hazards Program and IRIS Seismic Data APIs, integrated into Databricks via Python and Spark Streaming.
* **GeoPandas**: Python library for handling WorldPop’s geospatial population density data, enabling spatial joins with seismic event locations (optional—confirm if used).

**Architecture:**

The architecture diagram for our Real-Time Seismic Activity Monitoring System illustrates the end-to-end data pipeline, designed and implemented within Databricks, to ingest, process, and visualize seismic and demographic data for timely decision-making. The diagram employs a Medallion architecture pattern, structured into three layers—Bronze, Silver, and Gold—using Databricks Delta Lake as the underlying table format, ensuring scalability, reliability, and data quality.

**Data Ingestion (Sources)**: The pipeline begins with three primary data sources:

* The USGS Earthquake Hazards Program API (JSON format) and IRIS Seismic Data API (JSON format) provide near real-time seismic event data, including magnitude, geolocation, depth, and timestamps
* The WorldPop population density dataset (CSV format) supplies static geospatial data for India (2020), enabling risk assessment by correlating seismic events with population exposure.

These sources feed raw data into the **Bronze Layer**, stored in Delta Lake tables within Databricks. This layer captures unprocessed, near real-time data as it arrives, maintaining the original format and structure for auditability and historical analysis (e.g., usgs\_seismic\_events\_bronze, iris\_seismic\_events\_bronze, population\_data\_bronze).

**Data Transformation (Silver Layer)**: Apache Spark, integrated within Databricks, processes the Bronze data to cleanse, standardize, and enrich it. Quality checks are applied to handle missing or duplicate records, while geospatial joins integrate WorldPop data with seismic events. The transformed data is stored in the **Silver Layer** as curated Delta Lake tables (e.g., usgs\_seismic\_events\_silver, iris\_seismic\_events\_silver, population\_data\_silver), ready for analytics with improved consistency and structure.

**Data Aggregation (Gold Layer)**: Further aggregation and analysis occur in the **Gold Layer**, where Spark combines Silver data to produce aggregated insights, such as earthquake frequency trends, magnitude distributions, and population exposure risks. These refined datasets (e.g., seismic\_events\_gold, population\_data\_gold) are optimized for querying and visualization, stored in Delta Lake for performance and accessibility.

**Visualization (Output)**: The final Gold-layer data feeds into Power BI dashboards, enabling real-time visualization of 10 key performance indicators (KPIs) and use cases, such as earthquake magnitude trends and high-risk zone mapping. The Databricks Dashboard, as depicted, serves as the interface for stakeholders, including disaster management agencies, seismologists, and governments, to monitor and interact with the data.

**Technology Stack**: The diagram highlights the use of Databricks for orchestration (via Databricks Workflows), Apache Spark for processing, and Delta Lake for data management, ensuring a scalable and robust solution handling over 3 million rows from the source datasets.

This architecture ensures end-to-end data integrity, from ingestion to visualization, while addressing challenges like high data volumes, real-time processing, and data quality, as outlined in our project scope.

A diagram of a diagram

AI-generated content may be incorrect.

**Data Dictionary and Audit Trail**

This section outlines all data sources used in the analysis and the transformations applied to prepare the data for downstream use. The documentation ensures transparency, enabling full reproducibility and validation of the results.

**Data Source:**

1. **Population Data (Bronze Layer)**
   * **Description**: ZIP code metadata including population statistics, geographic centroids (latitude/longitude), city names, state, and demographics.
   * **Location**: deep\_sync\_us\_zip\_code\_metadata\_populations\_geo\_centroid\_lat\_lng\_city\_names\_state\_dma\_demographics.default.zip\_code\_metadata
   * **Format**: Delta Table
   * **Fields**:
     + ZIP: ZIP code identifier (String)
     + GEOPOINT: Geographic point data (String)
     + LATITUDE: Latitude coordinate (Double)
     + LONGITUDE: Longitude coordinate (Double)
     + TOTAL\_POPULATION: Total population count (Integer)
     + TOTAL\_MALE\_POPULATION: Male population count (Integer)
     + TOTAL\_FEMALE\_POPULATION: Female population count (Integer)
     + MEDIAN\_AGE: Median age of population (Double)
   * **Source Origin**: Loaded from a shared dataset into tabular.dataexpert.population\_data\_bronze on 2025-03-02.
2. **USGS Seismic Events (Bronze Layer)**
   * **Description**: Earthquake data from the USGS API, backfilled for 2018–2024.
   * **Location**: https://earthquake.usgs.gov/fdsnws/event/1/query
   * **Format**: GeoJSON (converted to Delta Table)
   * **Fields**:
     + event\_id: Unique event identifier (String)
     + magnitude: Earthquake magnitude (Double)
     + place: Location description (String)
     + time: Event timestamp (Timestamp)
     + longitude: Longitude coordinate (Double)
     + latitude: Latitude coordinate (Double)
     + depth: Depth in kilometers (Double)
     + [Additional fields like felt, cdi, mmi, etc., as per schema]
   * **Source Origin**: Retrieved via API calls, batched monthly, and stored in tabular.dataexpert.usgs\_seismic\_events\_bronze.
3. **IRIS Seismic Events (Bronze Layer)**
   * **Description**: Earthquake data from the IRIS API for 2018–2024, minimum magnitude 1.0.
   * **Location**: https://service.iris.edu/fdsnws/event/1/query
   * **Format**: Text (pipe-separated, converted to Delta Table)
   * **Fields**:
     + EventId: Unique event identifier (String, renamed to event\_id)
     + Time: Event timestamp (Timestamp, renamed to time)
     + Latitude: Latitude coordinate (Double, renamed to latitude)
     + Longitude: Longitude coordinate (Double, renamed to longitude)
     + Depth/km: Depth in kilometers (Double, renamed to depth)
     + Magnitude: Earthquake magnitude (Double, renamed to magnitude)
     + EventLocationName: Location name (String, renamed to event\_location\_name)
     + [Additional metadata fields like Author, MagType, etc.]
   * **Source Origin**: Fetched via API and stored in tabular.dataexpert.iris\_seismic\_events\_bronze.

**Data Transformations:**The following transformations were applied to process the raw data into silver and gold layers, ensuring quality and usability.

**Population Data Pipeline**

1. **Bronze to Silver (Population Data)**
   * **Source**: tabular.dataexpert.population\_data\_bronze
   * **Transformation**:
     + Rounded latitude and longitude to 2 decimal places using round(col("latitude"), 2) and round(col("longitude"), 2).
     + Applied quality checks:
       - Null ZIP codes: df.filter(col("ZIP").isNull()).count()
       - Null GEOPOINTs: df.filter(col("GEOPOINT").isNull()).count()
       - Negative population values: df.filter((col("TOTAL\_POPULATION") < 0) | ...)
       - Population mismatch: df.filter(col("TOTAL\_POPULATION") != (col("TOTAL\_MALE\_POPULATION") + col("TOTAL\_FEMALE\_POPULATION")))
       - Invalid lat/lon: df.filter((col("LATITUDE") < -90) | ...
       - Invalid median age: df.filter((col("MEDIAN\_AGE") < 0) | col("MEDIAN\_AGE") > 120))
     + Data written to tabular.dataexpert.population\_data\_silver only if all checks pass (counts = 0).
   * **Output**: Delta Table with cleaned coordinates and validated data.
2. **Silver to Gold (Population Data)**
   * **Source**: tabular.dataexpert.population\_data\_silver
   * **Transformation**:
     + Computed min/max latitude and longitude per ZIP using a window function: Window.partitionBy("ZIP").
     + Calculated land area using Haversine distance (haversine\_udf) between min/max coordinates, with a minimum of 1 sq.km for zero areas.
     + Joined land area with population data and computed population\_density using try\_divide(TOTAL\_POPULATION, land\_area\_sq\_km).
   * **Output**: Delta Table stored as tabular.dataexpert.population\_data\_gold.

**USGS Seismic Data Pipeline**

1. **Bronze Layer Ingestion**
   * **Source**: USGS API
   * **Transformation**:
     + Fetched data in monthly batches (2018–2024), with weekly chunks for failed requests.
     + Converted GeoJSON to PySpark DataFrame with defined schema.
     + Added year and month columns for partitioning.
   * **Output**: tabular.dataexpert.usgs\_seismic\_events\_bronze (partitioned by year, month, source).
2. **Bronze to Silver**
   * **Source**: tabular.dataexpert.usgs\_seismic\_events\_bronze
   * **Transformation**:
     + Deduplicated by event\_id, time, longitude, latitude, resolving magnitude conflicts with median.
     + Rounded latitude and longitude to 2 decimals.
     + Filled missing depth with regional averages, and missing coordinates with 0.
     + Dropped invalid entries: latitude (-90 to 90), longitude (-180 to 180), magnitude (0 to 10).
     + Standardized time to yyyy-MM-dd HH:mm:ss.
   * **Output**: tabular.dataexpert.usgs\_seismic\_events\_silver (partitioned by year, month).
3. **Silver to Gold**
   * **Source**: tabular.dataexpert.usgs\_seismic\_events\_silver and tabular.dataexpert.iris\_seismic\_events\_silver
   * **Transformation**:
     + Added severity based on depth (<70: High, 70-300: Moderate, >300: Low).
     + Joined with IRIS data on latitude and longitude to add event\_location\_name (renamed to location).
     + Computed min\_region\_magnitude and max\_region\_magnitude per location using window functions.
     + Dropped unused columns (detail, timezone, url, ids, types).
   * **Output**: tabular.dataexpert.seismic\_events\_gold (partitioned by year, month).

**IRIS Seismic Data Pipeline**

1. **Bronze Layer Ingestion**
   * **Source**: IRIS API
   * **Transformation**:
     + Fetched data for 2018–2024, parsed text format into a DataFrame, and cleaned column names (e.g., removed #, replaced spaces with \_).
   * **Output**: tabular.dataexpert.iris\_seismic\_events\_bronze.
2. **Bronze to Silver**
   * **Source**: tabular.dataexpert.iris\_seismic\_events\_bronze
   * **Transformation**:
     + Renamed columns for consistency (e.g., EventId to event\_id).
     + Deduplicated and resolved magnitude conflicts as in USGS pipeline.
     + Rounded coordinates, handled missing values, validated ranges, and standardized time to yyyy-MM-dd'T'HH:mm:ss.
   * **Output**: tabular.dataexpert.iris\_seismic\_events\_silver (partitioned by year, month).

**Reproducibility and Validation**

* **Environment**: Executed in Databricks using PySpark (Spark [version]), with libraries like pandas, requests, and math.
* **Code Location**: Scripts are stored in Databricks notebooks (e.g., unnamed notebooks for population, USGS, IRIS pipelines).
* **Audit Trail**: Intermediate datasets are saved as Delta Tables (e.g., bronze, silver, gold layers) with partitioning for efficient querying.
* **Validation Checks**: Quality checks (e.g., null counts, range validations) and post-transformation verifications (e.g., row counts, schema inspection) ensure data integrity.

This data dictionary and transformation log provide a complete record of the data pipeline, ensuring that all steps are traceable and reproducible.