ENSF 612
Fall 2025 Project Summary/Proposal

Group 12: Jack Shenfield (30274142), Truman Huang (30301429), Ibrahim Khan (30289806)

November 7th 2025

Automated Bug Severity Classification

The engineering problem our project aims to solve is improving the software development process via assigning levels of priority and severity to bugs in large software project repositories.

Preprocessing tools:

- Databricks + PySpark:
  Use Spark DataFrame operations for large bug report datasets.
  Handle cleaning (removing duplicates, missing values), tokenization, and label balancing.
  Ideal for distributed preprocessing and feature extraction (e.g., TF-IDF, embeddings).
- Feature Extraction:
  TF-IDF or Sentence Transformers (e.g., all-MiniLM-L6-v2) to convert textual bug descriptions into numerical representations.
  These embeddings can later be indexed for retrieval in the RAG pipeline.

LLMs for Classification:

BERT / RoBERTa / DistilBERT (TF-IDF built-in) fine-tuned for bug severity prediction.
Output: labels like Critical, Major, Minor, Trivial.

Dataset:

Bugzilla / Eclipse / Mozilla / Jira Bug Reports Dataset / GitHub Issues NLP

Content:

Fields: Bug ID, Title, Description, Severity Label, Component, Status, Timestamp, etc.

Why this dataset:

Real-world, multi-project, and multi-severity data.
Supports both classification (severity levels) and retrieval (similar past bug contexts). Clean structure for text-based and metadata-based features.

How is this relevant to big data engineering or software engineering?

Big Data Engineering:

Uses distributed data processing (Spark), feature engineering, and vector storage on Databricks to scale over large bug repositories. Involves efficient data pipelines for text ingestion, embedding generation, and retrieval.

Software Engineering:

Improves the defect triaging process by automating bug severity assignment, reducing developer workload, and providing context-aware reasoning (via RAG).

AI + RAG Integration:

Combines retrieval (vector similarity search for historical bugs) and generation (LLM explanations) to produce a transparent and explainable severity classification system