🧾 COMP3110 Project Report — Line Mapping Tool

Student: Emmanuel Omonua
Student ID: 110106145
Course: COMP-3110
Programming Language Focus: Java
Repository: https://github.com/EmmanuelOmonua/COMP3110-line-tracker

🧩 1. Introduction

This project designs a **line mapping tool** that identifies how each line in an **old version** of a source file maps to lines in a **new version**.
Instead of developing a full implementation, the focus is on the **design, dataset, evaluation plan, and visualization**.

The goal is to help developers analyze code changes more effectively by understanding:

- Which lines stayed the same
- Which were modified, added, deleted, split, or merged

The project also explores **how to visualize** these mappings in clear, user-friendly graphical interfaces.

⚙️ 2. Algorithm Design

The line mapping algorithm compares the content of two versions of a file — "old" and "new" — and identifies how their lines correspond.

The process includes:

1. **Preprocessing** (normalize spaces and cases)
2. **Exact matching** (detect unchanged lines)
3. **Similarity scoring** (Levenshtein distance and token similarity)
4. **Split and merge detection**
5. **Confidence scoring** (0–1 based on match accuracy)
6. **Mapping output** (table showing type and confidence)

Example:

Old Line	New Line(s)	Type	Confidence
12	14	unchanged	1.0
15	18,19	split	0.9
21	-	deleted	-

The algorithm prioritizes **accuracy and interpretability**, providing both developers and researchers with clear insight into how code evolves between versions.

🧾 3. Dataset Design

The dataset contains **25 file pairs** (`pair_001_old.java` → `pair_025_new.java`) stored in:

```
dataset/
├─ pair_001_old.java
├─ pair_001_new.java
...
├─ pair_025_old.java
├─ pair_025_new.java
└─ ground_truth.csv
```

Each pair contains small, realistic Java modifications:

- Variable renames
- Added or deleted lines
- Modified text
- Added comments
- Split or merged statements

The `ground_truth.csv` file provides the **true line mappings** used to evaluate the tool's performance.

Example entry:

```
pair_id,old_line,new_line,mapping_type
pair_001,4,4,modified
pair_003,4,3,deleted
pair_004,3,4,added_line
```

This dataset supports both qualitative inspection (looking at pairs) and quantitative evaluation (using precision and recall).

📊 4. Evaluation Plan

The tool's performance will be measured using **mapping accuracy** metrics such as:

Metric	Description
Precision	% of predicted mappings that are correct
Recall	% of true mappings that were found
F1-score	Harmonic mean of precision and recall
Mapping Coverage	% of old lines that were mapped somewhere

Formula Examples:

```
Precision = TruePositives / (TruePositives + FalsePositives)
Recall = TruePositives / (TruePositives + FalseNegatives)
```

The evaluation will be done by comparing the tool's output against the ground_truth.csv file.

To ensure fairness, the dataset includes diverse file types and modification styles.

🖥️ 5. Visualization Design

Three graphical interfaces are proposed for displaying line-mapping results.

🧩 1. Side-by-Side Line Mapping View

- Displays old file (left) and new file (right)
- Uses colored curved lines to connect related lines
- Ideal for developers reviewing code manually

🌊 2. Sankey Flow Diagram

- Shows how lines "flow" from old to new files
- Wide colored bands represent splits, merges, and deletions
- Best for high-level visualization of large files

🔥 3. Heatmap Confidence View

- Displays mapping strength as a grid
- Darker cells indicate higher confidence
- Useful for quickly spotting uncertain mappings

🧠 6. Bonus: Bug-Introducing Change Detection

A proposed extension uses line mapping to detect bug-introducing commits:

- After identifying mappings between old and new lines,
- The tool can trace whether deleted or modified lines were previously linked to bug fixes.
- If so, it flags the corresponding change as a potential bug-introducing change (BIC).

This feature builds on research in software evolution analysis and could be implemented later for extra marks or research credit.

✅ 7. Conclusion

This project outlines the full design of a line mapping visualization tool without implementation. It includes:

- A detailed algorithm
- A 25-file dataset
- A structured evaluation plan
- Clear visualization mockups
- A proposed extension for advanced change detection

The repository demonstrates both technical understanding and design clarity, fulfilling all project requirements.

📎 GitHub Repository

https://github.com/EmmanuelOmonua/COMP3110-line-tracker