Assignment Instructions

Dataset for following Assignments:

PURE: a Dataset of Public Requirements Documents

Dataset Reference:
Ferrari, A., Spagnolo, G. O., & Gnesi, S. (2017, September). *PURE: A Dataset of Public Requirements Documents.* In 2017 IEEE 25th International Requirements Engineering Conference (RE) (pp. 502–505). IEEE.
Dataset Link: https://zenodo.org/records/1414117

Task:

1. Download the **PURE** dataset from the given link.
2. Select the **Software Requirements Specification (SRS)** documents provided in the datasets.
3. Identify and count the frequency of **requirement "bad smells"** which are specified in Assignments (1-5). Examples of bad smells include: ambiguity, vagueness, incompleteness, redundancy, etc. For **model training and testing**, you can use some other dataset as well.
4. For each identified bad smell, suggested improvement or rewrite to remove/mitigate the smell.
5. Implement your detection and improvement process in code (any programming language of your choice) and upload it to a **public GitHub repository** (along with the **data set** used).

Deliverables:

1. A **summary table** showing:

- ✓ Type of bad smell
- ✓ Frequency in the dataset
- ✓ Example(s) from the SRS
- ✓ Solution for improvements

2. A **short report** (max 3 pages) explaining your findings, methodology, and key observations.
3. GitHub repository link containing:

- ✓ Source code used for analysis and improvement
- ✓ Instructions to run the code (README file)

Assignment 1 – Passive Voice & Conditional Modal Detection in Requirements Documents

Design and implement a tool that analyzes software requirement specification (SRS) documents to detect:

1. Sentences written in **passive voice**.
2. Sentences containing **conditional modals** (e.g., *could*, *might*, *should*, *would*).

Your tool should parse text using NLP techniques and highlight the problematic sentences. Provide a report summarizing the frequency of these occurrences, their positions in the document, and potential rewrites in an active, direct style.

Assignment 2 – Detection of Missing Information & Vague Pronoun Detector

Develop a requirement-quality checker that identifies:

1. **Indicators of missing information** necessary for a complete specification (e.g., "TBD", "to be decided", blank values).
2. Detect pronouns whose referents are unclear or ambiguous in the context (e.g., it, they, this, those without clear antecedents). Flag sentences where multiple possible antecedents exist. (**Anaphora Resolution**)

The tool should produce an annotated version of the document, flagging such instances, and generate recommendations for more precise replacements.

Assignment 3 – Comparative & Non-atomic Requirement Analysis

Build a parser-based analysis system to:

1. Detect **superlatives or comparatives** in requirements (e.g"best", "better", "fastest").
2. Identify **non-atomic requirements** caused by coordinators (e.g., "and", "or", "as well as") where multiple requirements are combined into one.

Provide statistical data on how often these occur and propose splitting them into independent, measurable requirements.

Assignment 4 – Sentence Length, Complexity & Simplification in SRS

Create a linguistic-analysis tool that:

1. Measures **sentence length** and flags those exceeding or falling below a defined threshold.

2. Calculates **complexity** using syntactic parsing and clause counting.

Suggests **simplified versions** for overly complex or verbose sentences. Additionally, check punctuation consistency and correctness within the document.

Assignment 5 – Abbreviation Completeness & Subjective Terms Checker

Implement a system to:

1. Identify **missing abbreviation definitions** (first use of an abbreviation/acronym without its expansion). Detect inconsistencies in abbreviation usage (e.g., use of multiple abbreviations for the same term).

3. **Vague, imprecise, or subjective terms** (e.g., “user-friendly”, “fast”, “efficient”).

The output should include a glossary of abbreviations with their first-occurrence definitions and highlight any missing entries. The tool flagging such instances, and generate recommendations for more precise replacements for subjective terms.
