Introduction

This document outlines the key technical challenges encountered and the engineering solutions implemented during the development of the NER pipeline. The project successfully fulfills all six assignment tasks, resulting in a complete, end-to-end system. The following sections detail the process of overcoming real-world data and environment issues, and provide an analysis of the final system's limitations and potential avenues for improvement.

Technical Challenges & Solutions

1. **Challenge: Handling Long Text Sequences**
    - **Problem:** The chosen transformer model (BERT) has a maximum input limit of 512 tokens. Several forum posts in the dataset exceeded this limit, causing runtime errors during the scaled evaluation (Task 5).
    - **Solution:** The prediction pipeline in **Task 2** was re-architected to be robust to inputs of any length. A sliding window (chunking) strategy was implemented, where long texts are automatically split into overlapping segments. The model processes each chunk individually, and the results are then stitched together, with a deduplication step to handle entities found in the overlapping regions. This solution demonstrates the ability to handle a common limitation of transformer models.
2. **Challenge: Aligning Model Output with Project Requirements**
    - **Problem:** The pre-trained model's output labels (e.g., `problem`, `treatment`) were generic and did not match the specific target labels required by the assignment (`ADR`, `Disease`, `Symptom`).
    - **Solution:** A rule-based mapping layer (`get_refined_label` function) was developed. This function uses curated keyword lists to re-classify the model's generic outputs into the required, fine-grained categories. This demonstrates a practical approach to adapting pre-trained models to specific task requirements without retraining.
3. **Challenge: Data Sparsity for Entity Linking**
    - **Problem:** Demonstrating the entity linking functionality in **Task 6** was difficult because the dataset's standard code annotations (`sct` files) were sparse. Many files with Adverse Drug Reactions (`ADR`s) lacked corresponding standard codes.
    - **Solution:** Instead of guessing, a programmatic approach was taken. A script was written to scan the entire dataset to locate a "golden" sample file that was

guaranteed to contain the necessary annotations to successfully test and validate the string-matching and embedding-based linking functions.

Limitations & Future Improvements

1. **Limitation: Heuristic Label Mapping**
    - The current keyword-based mapping is a simple heuristic and is the primary factor limiting the system's performance (especially the low recall for ADR and Disease seen in Task 5). My lack of a formal medical background restricted the scope and accuracy of these keyword lists.
    - **Next Step:** With more time or access to a domain expert, these lists could be significantly expanded using medical dictionaries (gazetteers), which would directly improve performance.
2. **Limitation: Reliance on a Pre-trained Model**
    - The project uses a general biomedical model without re-training. As stated, hardware constraints (lack of access to a suitable GPU) prevented the ideal next step of fine-tuning.
    - **Next Step:** The most significant performance gain would come from **fine-tuning** the model on the CADEC dataset. This would teach the model the specific nuances and labels of this dataset, likely yielding a substantial F1-score increase and removing the need for the heuristic mapping layer altogether.

Conclusion

Despite the challenges, a complete, working pipeline was successfully built, debugged, and evaluated, demonstrating a wide range of practical NLP and software engineering skills. The final system is a robust baseline that performs the required tasks, and a clear, data-driven path for its future improvement has been identified.