TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

KHOA CÔNG NGHỆ THÔNG TIN

---o0o---

BÀI TẬP LAB MÔN CĐ TCDL

BÀI TẬP 4

Giảng viên hướng dẫn : **VŨ QUỐC HOÀNG**

Sinh viên thực hiện: **CAO QUỐC VIỆT**

MSSV: **22810218**

Lớp : **CN2022/2**

Khoá : **2022/2**

TP. Hồ Chí Minh, tháng 04 năm 2025

Báo cáo BT4: UTF-ANSI Translator

1. Tổng quan dự án

UTF-ANSI Translator là một công cụ chuyển đổi giữa mã hóa UTF-8 và ASCII/ANSI, đặc biệt được tối ưu cho việc xử lý tiếng Việt. Dự án bao gồm hai thành phần chính:

1. **Model Service**: Dịch vụ phục hồi dấu tiếng Việt sử dụng mô hình học máy transformer.
2. **UnA (UTF-ANSI)**: Giao diện web cho phép người dùng chuyển đổi văn bản và tập tin.

2. Kiến trúc hệ thống

2.1. Mô hình dịch vụ

Hệ thống được thiết kế theo kiến trúc microservice:

- **Model Service**: Chạy độc lập trên cổng 5001, cung cấp API để phục hồi dấu cho văn bản tiếng Việt.
- **UnA Web Interface**: Chạy trên cổng 5000, cung cấp giao diện người dùng và xử lý các yêu cầu chuyển đổi.

2.2. Cấu trúc thư mục

- UTF-ANSI-Translator/
  - Model/
    - model_service.py: Dịch vụ REST API cho việc phục hồi dấu
    - run_model.py: Module xử lý mô hình transformer
    - requirements.txt: Dependencies cho Model Service
    - [files mô hình]: Các file mô hình transformer
  - UnA/
    - app.py: Ứng dụng Flask chính
    - runserver.py: Script khởi chạy server
    - modules/
      - converter.py: Module chuyển đổi UTF-8 và ASCII
      - utils.py: Tiện ích xử lý file và encoding
      - **init**.py: Khởi tạo package

        - static/: Tài nguyên static (CSS, JS)
        - templates/: Templates HTML
        - temp/: Thư mục chứa file tạm
    - start_model_service.bat: Script khởi động Model Service
    - start_translator.bat: Script khởi động UnA web interface
    - README.md: Tài liệu hướng dẫn

3. Tính năng chi tiết

3.1. Chuyển đổi UTF-8 sang ASCII

- **Xử lý ký tự Tiếng Việt**: Chuyển đổi các ký tự tiếng Việt có dấu thành không dấu thông qua bảng ánh xạ trực tiếp.
- **Xử lý ký tự đặc biệt**: Các ký tự không thuộc bảng mã ASCII (như emoji, ký tự đặc biệt, chữ Trung/Nhật/Hàn) được thay thế bằng ký tự "?".
- **Bảo toàn cấu trúc văn bản**: Giữ nguyên dạng xuống dòng, khoảng trắng và định dạng của văn bản gốc.
- **Bảo toàn ký tự ASCII**: Giữ nguyên các ký tự thuộc bảng mã ASCII (0-127).

3.2. Chuyển đổi ASCII sang UTF-8 (phục hồi dấu)

- **Phục hồi dấu tiếng Việt**: Sử dụng mô hình transformer để phục hồi dấu cho văn bản tiếng Việt.
- **Xử lý đa dòng**: Xử lý từng dòng riêng biệt để bảo toàn cấu trúc văn bản có nhiều đoạn.
- **Bảo toàn dòng trống**: Giữ nguyên các dòng trống trong văn bản.
- **Xử lý văn bản dài**: Khả năng xử lý các văn bản dài mà không làm mất cấu trúc.

3.3. Xử lý file

- **Hỗ trợ upload file**: Cho phép người dùng tải lên file để chuyển đổi.
- **Hỗ trợ đa định dạng**: Có thể xử lý các file văn bản với nhiều định dạng mã hóa khác nhau.
- **Tạo file duy nhất**: Tạo tên file đầu ra duy nhất dựa trên UUID để tránh xung đột.
- **Tự động dọn dẹp**: Xóa các file tạm sau 1 giờ để tiết kiệm không gian lưu trữ.

3.4. Giao diện người dùng

- **Giao diện web thân thiện**: Dễ dàng sử dụng với các tùy chọn đơn giản.
- **Xem trước kết quả**: Hiển thị kết quả chuyển đổi trực tiếp trên trình duyệt.
- **Tải xuống kết quả**: Cho phép tải xuống file đã được chuyển đổi.

- **Hỗ trợ nhập trực tiếp**: Cho phép người dùng nhập văn bản trực tiếp thay vì phải tải lên file.

4. Xử lý ngoại lệ

4.1. Xử lý lỗi mã hóa

- **Phát hiện mã hóa thông minh**: Sử dụng chardet để tự động phát hiện mã hóa của file input.
- **Fallback mã hóa**: Thử nhiều mã hóa khác nhau khi phát hiện mã hóa thất bại.
- **Xử lý khi chardet trả về None**: Sử dụng UTF-8 làm mã hóa mặc định khi không phát hiện được.
- **Thử nhiều mã hóa phổ biến**: Thử lần lượt 'utf-8', 'cp1252', 'latin-1', 'ascii' khi đọc file.
- **Sử dụng latin-1 làm phương án cuối cùng**: Mã hóa latin-1 có thể đọc bất kỳ chuỗi byte nào.

4.2. Xử lý lỗi khi ghi file

- **Thay thế ký tự không hợp lệ**: Sử dụng tùy chọn errors='replace' để thay thế ký tự không hỗ trợ bằng "?".
- **Xử lý lỗi write_file**: Try-except để bắt các lỗi khi ghi file và thử lại với cách tiếp cận khác.
- **Đảm bảo encoding hợp lệ**: Kiểm tra và cung cấp giá trị mặc định cho encoding khi giá trị là None.

4.3. Xử lý lỗi dịch vụ Model

- **Kiểm tra kết nối Model Service**: Thử kết nối đến Model Service khi khởi động ứng dụng.
- **Xử lý khi Model Service không khả dụng**: Thông báo cho người dùng và vẫn cho phép chuyển đổi UTF-8 sang ASCII.
- **Xử lý timeout khi gọi API**: Đặt timeout hợp lý và xử lý lỗi khi gọi Model Service.
- **Kiểm tra trạng thái phản hồi từ API**: Kiểm tra HTTP status code và xử lý khi không thành công.

4.4. Xử lý lỗi chung của ứng dụng

- **Xử lý lỗi khi tệp quá lớn**: Giới hạn kích thước tệp tải lên (5MB) và trả về lỗi 413 khi vượt quá.
- **Ghi log đầy đủ**: Ghi lại thông tin chi tiết về lỗi và quá trình xử lý để dễ dàng debug.
- **Thông báo lỗi rõ ràng**: Trả về thông báo lỗi dễ hiểu cho người dùng.
- **Bảo vệ ứng dụng khỏi crash**: Bắt tất cả các ngoại lệ không mong muốn để ứng dụng không bị dừng đột ngột.

5. Cách chạy dự án

5.1. Chạy tự động với script batch

1. **Khởi động Model Service**:

   ```
   start_model_service.bat
   ```

   File batch này sẽ:
   - Chuyển đến thư mục Model
   - Kích hoạt môi trường ảo
   - Cài đặt các dependencies cần thiết
   - Khởi động dịch vụ Model trên cổng 5001
2. **Khởi động UnA Web Interface**:

   ```
   start_translator.bat
   ```

   File batch này sẽ:
   - Chuyển đến thư mục UnA
   - Kích hoạt môi trường ảo
   - Cài đặt các dependencies cần thiết
   - Khởi động giao diện web trên cổng 5000

5.2. Chạy thủ công từng bước

1. **Cài đặt Model Service**:

   ```
   cd Model
   python -m venv venv
   venv\Scripts\activate
   pip install -r requirements.txt
   python model_service.py
   ```

2. **Cài đặt UnA Web Interface** (trong terminal riêng):

   ```
   cd UnA
   python -m venv venv
   venv\Scripts\activate
   pip install -r requirements.txt
   python runserver.py
   ```

Sau khi khởi động cả hai dịch vụ, truy cập http://localhost:5000 để bắt đầu sử dụng ứng dụng chuyển đổi UTF-8/ASCII.

> **Ghi chú:** Từng thử viết mã huấn luyện một mô hình Transformer (tham khảo của François Chollet) trên laptop cá nhân (Ryzen 9 5900HX, RAM 32GB, GPU RTX 3080). Tuy nhiên, việc huấn luyện quá chậm (ngay cả với GPU T4 cũng mất khoảng 17 tiếng), nên đã quyết định sử dụng mô hình có sẵn từ Hugging Face để đảm bảo tiến độ và hiệu quả.