# IR Project with Web Crawler Integration

This project extends the Information Retrieval (IR) application with a Python web crawler to collect and index documents from the internet.

## Overview

The project consists of two main components:

1. **Java IR Application** - An Information Retrieval system built with Apache Lucene that processes and indexes documents in the Cranfield format, performs queries, and evaluates results.
2. **Python Web Crawler** - A web crawler that fetches academic papers from the internet and formats them to be compatible with the IR application.

## Requirements

### Java Application

* Java 8 or higher
* Maven

### Python Web Crawler

* Python 3.6 or higher
* Required Python packages (install with pip install -r requirements.txt):
  + requests
  + beautifulsoup4
  + argparse

### Setting up Virtual Environment (Recommended)

1. Create a virtual environment:

* python -m venv venv

1. Activate the virtual environment:
   * On Linux/Mac:
   * source venv/bin/activate
   * On Windows:
   * .\venv\Scripts\Activate.ps1
2. Install dependencies in the virtual environment:

* pip install -r requirements.txt

1. When you’re done, you can deactivate the virtual environment:

* deactivate

## Setup

1. Clone the repository
2. Install Java dependencies using Maven:

* mvn clean install

1. Install Python dependencies:

* pip install -r requirements.txt

## Usage

### Running the Web Crawler

The web crawler can be run independently:

python web\_crawler.py --urls https://arxiv.org/list/cs.IR/recent --max-pages 50

Options: - --urls: One or more starting URLs (separated by spaces) - --output: Output directory (default: src/main/resources/crawled) - --max-pages: Maximum number of pages to crawl (default: 100) - --delay: Delay between requests in seconds (default: 1.0) - --workers: Number of concurrent workers (default: 5)

### Integrating with the Java Application

To run the crawler and integrate with the Java application:

python run\_crawler.py --urls https://arxiv.org/list/cs.IR/recent --max-pages 50 --run-java

This will: 1. Run the web crawler to collect documents 2. Format the documents in Cranfield format 3. Create query and relevance judgment files 4. Set up the files for the Java application 5. Optionally run the Java application (with –run-java flag)

### Running the Java Application Manually

After running the crawler, you can run the Java application manually:

mvn clean compile exec:java -Dexec.mainClass=IR.App

## How It Works

1. The web crawler starts from the provided URLs and collects academic papers.
2. The papers are processed and formatted according to the Cranfield format used by the IR application.
3. The crawler creates three main files:
   * cran.all.crawled: Contains the formatted documents
   * cran.qry.crawled: Contains sample queries generated from document titles
   * cranqrel: Contains relevance judgments for evaluation
4. The integration script copies these files to the locations expected by the Java application.
5. The Java application indexes the documents, processes queries, and evaluates results using the Lucene framework.

## Customization

You can customize the crawler behavior by modifying:

* Starting URLs: Use --urls to specify different academic sources
* Crawl depth: Adjust --max-pages to control how many documents are collected
* Integration: Edit run\_crawler.py to change how files are prepared for the Java application

## Recent Changes

### Java Application Updates

The Java application has been modified to handle missing directories:

* QueryIndex.java: Added automatic creation of the results directory if it doesn’t exist
* // Create results directory if it doesn't exist  
  File resultsDir = new File(RESULTS\_DIRECTORY);  
  if (!resultsDir.exists()) {  
   resultsDir.mkdirs();  
  }
* This change fixes the FileNotFoundException error that occurred when trying to save search results.

## Troubleshooting

* **Missing Files**: Ensure the crawler has completed successfully by checking the output directory
* **Java Application Errors**: Check the Java application logs for specific error messages
* **Crawler Issues**: Check the crawler.log file for details on any crawling problems