Test Case Specification

Group 1

January 5, 2026

1 Introduction

This document describes test cases for the LTU Search Engine project.

2 Test Case: TC-FRQ-1010

Related Requirements

FRQ-1010

Description

Verify that the crawler does not extract or follow hyperlinks contained inside PDF documents. Hyperlinks embedded in PDF files must not be added to the crawl frontier or visited, even if the crawler downloads the PDF file.

Preconditions

- Crawler component is operational
- Crawling of normal HTML hyperlinks is enabled
- A reachable PDF document exists that contains one or more hyperlinks
- Logging of visited and scheduled URLs is enabled

Test Steps

1. Configure a seed URL that links to a PDF document.
2. Start the crawler with default link-following enabled.
3. Allow the crawler to access and download the PDF document.
4. Inspect the crawl frontier (scheduled URL list).
5. Inspect the list of visited URLs and crawler logs.

Expected Results

- The crawler may download or access the PDF file itself.
- No hyperlinks embedded inside the PDF are extracted.
- No hyperlinks embedded inside the PDF are scheduled for crawling.
- No hyperlinks embedded inside the PDF are visited.
- The crawler continues processing other allowed links normally.
- No errors or crashes occur as a result of encountering PDF links.

Expected Results

- Only visible textual content is extracted.
- Images, videos, scripts, and binary resources are ignored.
- No non-textual data is stored in the search index.
- Extracted text is suitable for term-based indexing.

3 Test Case: TC-FRQ-2001

Related Requirements

FRQ-2001, FRQ-2004

Description

Verify that the system extracts only textual content from HTML pages and ignores all non-textual content during the indexing process.

Preconditions

- An HTML page containing visible text content
- The same page contains images, videos, scripts, and binary files
- Search index is empty

Test Steps

1. Submit the HTML page to the indexing component.
2. Parse the HTML document.
3. Extract all indexable content.
4. Inspect extracted data before storage.
5. Store extracted content in the search index.

Expected Results

- Only visible textual content is extracted.
- Images, videos, scripts, and binary resources are ignored.
- No non-textual data is stored in the search index.
- Extracted text is suitable for term-based indexing.

4 Test Case: TC-FRQ-2002

Related Requirements

FRQ-2002

Description

Verify that indexed terms are stored together with references to the pages in which they appear using an inverted index structure.

Preconditions

- Two or more HTML pages containing overlapping keywords
- Search index is empty

Test Steps

1. Submit all pages to the indexing component.
2. Extract and tokenize textual terms from each page.
3. Normalize extracted terms.
4. Store terms in the search index.
5. Inspect the internal index structure.

Expected Results

- Each unique term is stored exactly once in the index.
- Each term maps to one or more page references (URLs or document IDs).
- Pages containing the same term are associated with that term.
- The index follows the inverted index model.

5 Test Case: TC-FRQ-2003

Related Requirements

FRQ-2003

Description

Verify that the system supports incremental updates of the search index without rebuilding the entire index.

Preconditions

- Existing search index populated with indexed pages
- One existing page is modified or a new page is added

Test Steps

1. Run the indexing process on the initial dataset.
2. Modify an existing page or add a new page.
3. Run the indexing process again.
4. Monitor which pages are re-indexed.
5. Compare the index state before and after the update.

Expected Results

- Only new or modified pages are re-indexed.
- Unchanged indexed pages remain untouched.
- No full index rebuild occurs.
- The index remains consistent and searchable after the update.

6 Test Case: TC-FRQ-3001

Related Requirements

TC-FRQ-3001

Description

Verify that the search engine accepts and correctly processes search queries composed of simple terms and Boolean operators (e.g., AND).

Preconditions

- The search system is running and accessible.
- The search index is populated with test data containing known terms.

  Example setup:

  - ∗ Document A contains only "cats".
  - ∗ Document B contains only "dogs".
  - ∗ Document C contains both "cats" and "dogs".

Test Steps

1. Simple Term Search: Submit a search query with a single term: cats.
2. Multiple Terms Search: Submit a search query with multiple terms: cats dogs.
3. Operator Search: Submit a search query using the AND operator: cats AND dogs.

Expected Results

- Simple Term: The system returns Document A and Document C (all docs containing "cats").
- Multiple Terms: The system returns Document A, B, and C (depending on default ranking, but all relevant docs should be found).
- Operator Search: The system returns only Document C (the document containing both terms). This confirms the AND operator is working logically and not just finding any document with either word.