

Session: 2021 – 2025

**Submitted by:**

Abdul Mateen 2021-CS-190

**Supervised to:**

Dr. Khaldoon Syed Khurshid

Department of Computer Science

**University of Engineering and Technology**

**Lahore Pakistan**

## **1. Objective of the System**

This system is designed to search and manage text documents in a directory. The key features include:

* **Indexing documents**: Building an index for quick search of words across documents.
* **Search functionality**: Searching for documents by content or title.
* **File upload**: Allowing users to upload new .txt files and automatically updating the index with the newly added documents.

## **2. Key Components of the System**

* **Document Indexing**: When the server starts, it builds an index of all text documents in the DOCUMENTS\_DIR. This includes:
  + An **index** of words to the documents they appear in.
  + A **title index** that maps document titles to their full content.
  + A **documents** dictionary that stores the entire content of each document.
* **Search Functions**:
  + The system supports two types of searches:
    - **Content-based search**: Searching for a word across all documents and retrieving the documents where the word appears.
    - **Title-based search**: Searching for a specific document by its title.
  + Additionally, it highlights the matching lines containing the search query in the documents.
* **File Upload**: Users can upload .txt files through a web interface. Once the file is uploaded, the system reads its content, updates the document index, and saves it in the DOCUMENTS\_DIR.

## **3. Detailed Process Flow**

### Initialization

* + On server startup, the system automatically builds the index for all existing documents in the DOCUMENTS\_DIR.
  + The build\_index() function reads each .txt file and creates mappings for word occurrences, document titles, and the complete content.

### Search Process

* + The search functionality works in two modes:
    - **Title search**: The user submits a title query, and the system checks if the title exists in the TITLE\_INDEX.
    - **Content search**: The user submits a word or phrase, and the system searches for that word across all documents by checking the index dictionary.

### File Upload Process

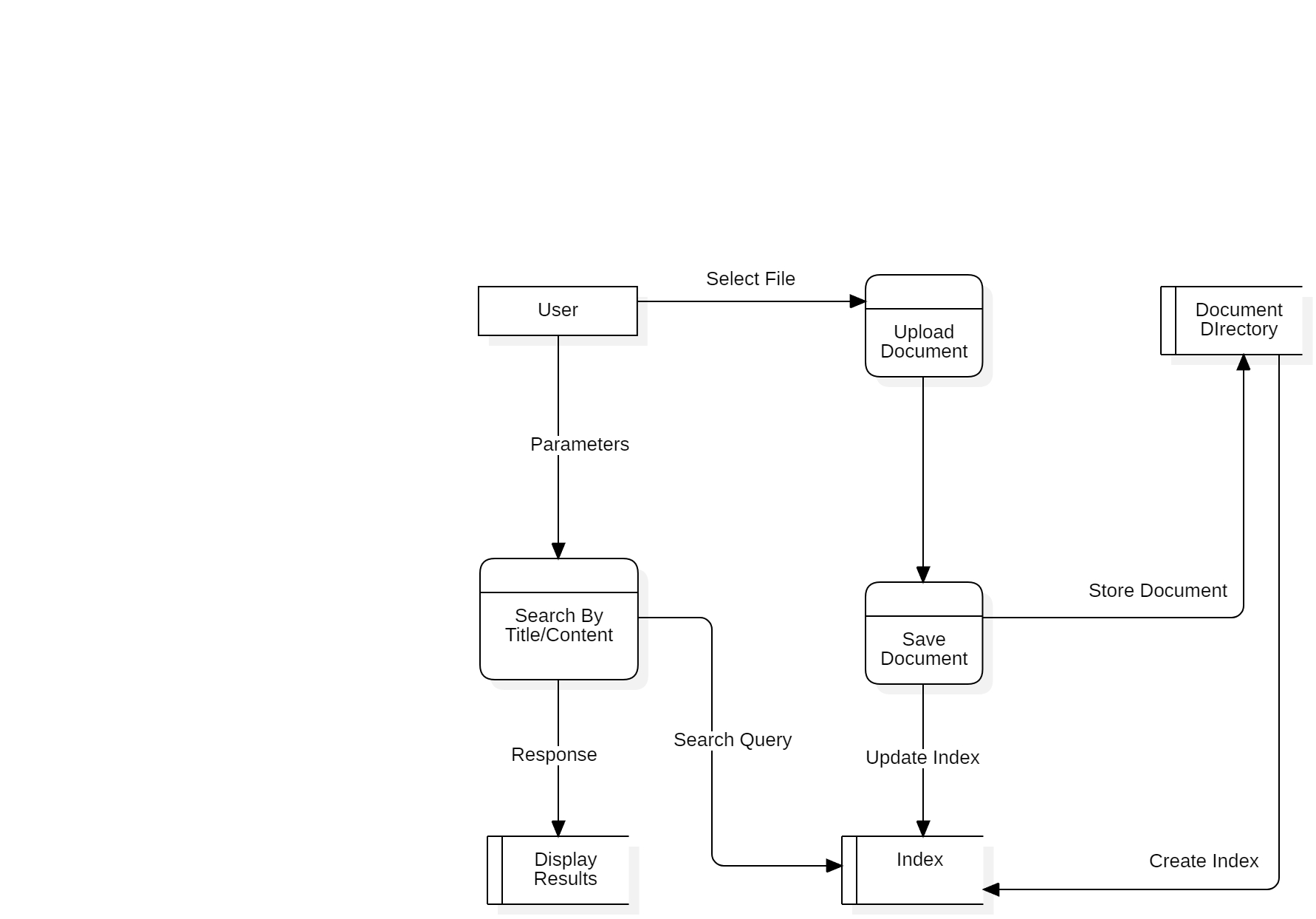
* + Users can upload .txt files. The system verifies the file type and saves the file to the DOCUMENTS\_DIR.
  + After saving the file, the update\_index() function is called to update the indexes (index, title\_index, and documents) with the new file's content.

## 4. Data Flow Diagrams (DFDs)

Below is a **Level 1 DFD** outlining the major steps involved in the process:

### Level 1 DFD

* **External Entity**: User
* **Process**: Search & File Upload System
* **Data Stores**:
  + **Documents Directory (DOCUMENTS\_DIR)**: Stores all .txt files.
  + **Indexes**: Stores word-to-document mappings (index), title-to-content mappings (title\_index), and the full content of each document (documents).



* **External Entity (User)**: The user interacts with the system to either search for documents or upload new files.
* **Search & File Upload System**: This process handles both the search and file upload functionality. It processes the requests and interacts with the document storage system.
* **Documents Directory (DOCUMENTS\_DIR)**: This is where the actual .txt files are stored.
* **Indexes**: The system maintains indexes for efficient searching.

## 5. Modular-Level Code Representation

* **build\_index()**: This function initializes the system's indexes and stores all documents in memory for fast access. It tokenizes the content of each document into words and creates mappings from words to document names.
* **search\_by\_content(query)**: This function searches for a query word in the indexed documents and returns the list of documents that contain the word.
* **search\_by\_title(title)**: This function searches for a document by its title from the index and returns the document's name if found.
* **find\_matching\_lines(query, documents)**: This function finds and highlights lines containing the search query in the documents.
* **update\_index(file\_path, filename)**: This function is called whenever a new file is uploaded or an existing file is modified. It updates the document indexes with the new content.
* **upload\_file\_view(request)**: Handles the file upload process. It ensures that only .txt files are accepted and updates the index with the newly uploaded file.

## 6. Conclusion

This system offers a robust solution for managing and searching documents efficiently. The combination of indexing, content-based and title-based search, and real-time updates upon file upload ensures that users can easily access the documents they need. The Data Flow Diagrams (DFDs) offer a clear representation of how the system handles data throughout its various processes, while the modular code structure ensures that each part of the system functions independently for easy maintenance and updates.