# 3 Design

## 3.1 Design Context

### 3.1.1 Broader Context

Buildertrend provides construction project management software and solutions for construction companies, homeowners, and builders. Our search feature will enable them to save time and increase productivity by reducing the time spend searching through various documents.

| **Area** | **Description** | **Project Relevance** |
|---|---|---|
| Public health, safety, and welfare | How does your project affect the general well-being of various stakeholder groups? These groups may be direct users or may be indirectly affected (e.g., solution is implemented in their communities) | Our project tries to make information inside uploaded files more accessible to the different users. It also allows users to find all the files relevant to their search and cross reference them. The goal is to save the users time. |
| Global, cultural and social | How well does your project reflect the values, practices, and aims of the cultural groups it affects? Groups may include but are not limited to specific communities, nations, professions, workplaces, and ethnic cultures. | The main contextual groups are professionals in the construction industry. This project will enable them to have much greater search capabilities in terms of not only technical specifications, but also contextual bindings, user requests, etc. Our project doesn’t conflict with any ethnic or racial group and is transparent to different groups.. However, the product is for now limited to the construction industry. |
| Environmental | Direct and indirect environmental impact such as deforestation or unsustainable practices related to materials manufacture or procurement. | By having the capability and access to different search filters, construction workers can gain environmental awareness (e.g. municipality constraints on forestry practices near construction sites). Should future software versions scale beyond a threshold, resources such as energy consumption may be a concern. |

| | | |
|---|---|---|
| Economic | Possible financial viability of your product within the team, company, cost to consumers, or broader economic effects on communities, markets, nations, and other groups. | The reduced search time will allow Buildertrend customers to save on labor/billed time costs and work on more projects. Giving them an edge over the competition, and attracting more customers for buildtrend increasing sales and profits. In addition, other constraints with financial impacts may be searchable. |

### 3.1.2 User Needs

Our application has three different user groups, their needs are defined below.

**Builders/Contractors:** Builders and workers need a quick  and efficient method to search for information on site such as measurements, designs, orders, etc. in order to complete the project according to requirements.

**Homeowners:** Homeowners need a way to search documents regarding the construction of their home at any time. The documents could include important information such as previous contracts, costs, or design parts/specifications as necessary in order to make a decision regarding the project.

**Buildertrend Staff:** Buildertrend staff need a way to provide support to the users on any potential issues they face with the search feature. They will have access to the full entries saved in Elasticsearch in order to understand why certain results were returned for a query.

### 3.1.3 Prior Work/Solutions

Text extraction from files has been explored in various solutions. Other solutions typically provide support to a subset of file types our solution aims to provide support to. These solutions aim to solve the common goal of information retrieval system's (IRS) which is "finding relevant information or a document that satisfies user information needs" (Sharma). To achieve this goal, IRS's usually implement 3 processes: 1. Indexing, 2. Filtering, 3. Searching (Sharma). These 3 steps can be shown in the following diagram (Fig. 1)  from *A Survey on Information Retrieval Models, Techniques And Applications,* and is the common workflow of other solutions.

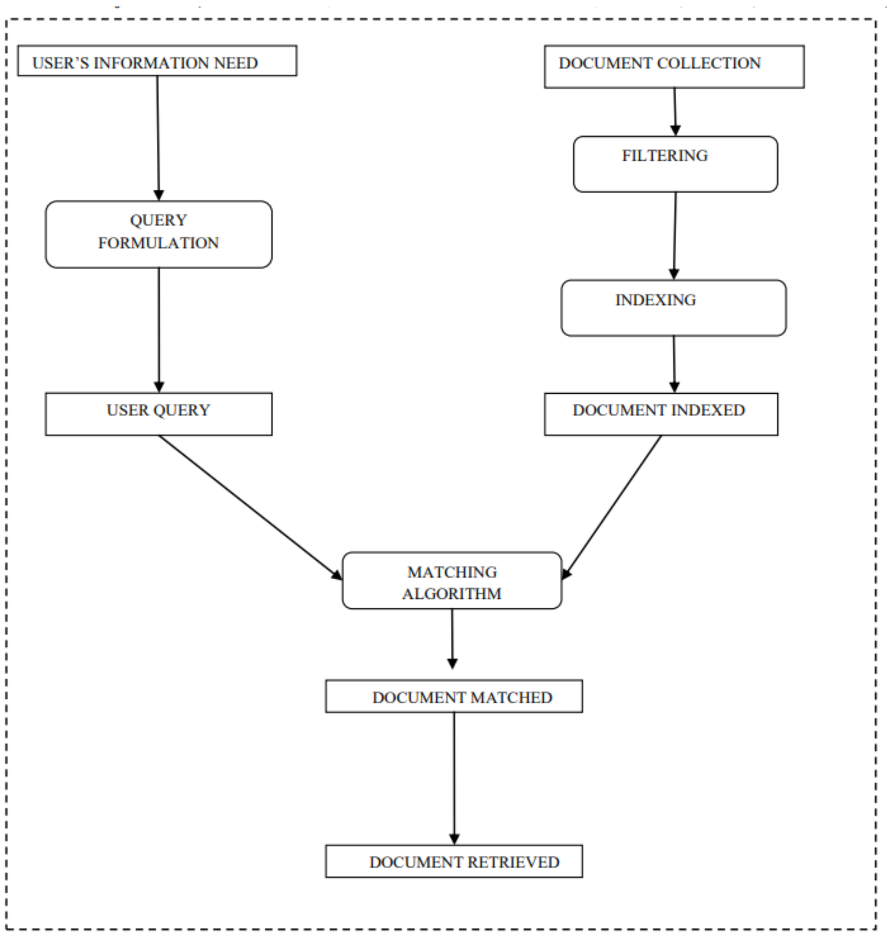

**Fig 1 A general framework of IR System**

Buildertrend has a current implementation of Elasticsearch that searches files only by file name. There is also an implementation that allows staff members to search for files based on their SQL representation.

Our solution will follow a similar flow as other IRS's (as shown in diagram above). It will allow Buildertrend to integrate the tool into their existing application, which would not be possible with other solutions. It will also support the file types that are relevant to Buildertrend's work. It is also important to note that other solutions utilize different database backends, however we are constrained to using Elasticsearch for storing our data.

### 3.1.4 Technical Complexity

The design consist of several subsystems of various complexity each requiring background knowledge and/or additional research, these subsystems include:

- Searching algorithm: Implement a search algorithm to find a return the best-fit result within 10-seconds.
  - Requirements: Algorithm analysis and implementation, CI/CD

    - How can we maximize query result precision and recall
- Simple frontend UI/UX: User-friendly web app with ability to navigate, search, and upload files
    - Requirements: Web development, React, CI/CD
- Backend architecture: use decided tech stack to implement a system to extract and index provided documents to be served to the frontend UI when queried.
    - Requirements: Database management, Tika, Elasticsearch, Investigate the impact of "mediator" that translates the text from different file types
- Test cases: design proper testing suite to check for project requirements as well as optimal functionality
    - Requirements: Experience with previous testing libraries(e.g. JUnit)
- Project management: We must combine all components of the project which requires the frontend, backend, and other components to be properly implemented and tested to enable them to communicate seamlessly and efficiently.

## 3.2 Design Exploration

### 3.2.1 Design Decisions

We elected to use the following technology stack to accomplish our task following research and interactions with the client:

- Frontend Framework (React)
- Backend Communication (REST)
- Backend API's (Apache Tika, Elasticsearch)
- Application type (Web application)

### 3.2.2 Ideation

We were given the flexibility to use various tools and technologies for this project by the client. Some of the options considered for the tech stack include:

- React JS (frontend/UI)
- Apache Lucene (file search)
- Apache Tika (type detection and data extraction)
- Elasticsearch (file indexing and search)

Upon researching and exploring various options, our decisions for the tech stack was based on the given requirements and basic functionality as this project would serve as a standalone feature to be implemented in another application at a later date.

### 3.2.3 Decision-Making and Trade-Off

| **Tools and Technologies** | **Pros** | **Cons** |
|---|---|---|
| **Frontend Framework** (React) | - Simple to learn<br>- Well-supported<br>- Can easily be expanded to add more functionality | - Unfamiliar to group members |
| **Backend Communication** (REST) | - Easy to implement<br>- Logistically simple<br>- Group members already familiar | - Possibly difficult to pick up for new developers |
| **Backend API's** (Apache Tika, Elasticsearch) | - Scalable<br>- Fast indexing and search responses | - (Elasticsearch) Not atomic (can read and write indexes at the same time) |
| **Application type** (Web application) | - Easily accessible<br>- Compatible on any platform (e.g. Windows, Mac, Linux) | - Accessible to anybody with URL unless secured (user login, network-specific accessibility) |

## 3.3 Proposed Design

### 3.3.1 Design Visual and Description

The following diagram, Figure 1, shows a high level overview of the interactions of our application with relevant external systems. The following are brief explanation of the numbered interactions:

1. Recieves a list of files relevant to the query made, and display them to the user
2. Triggers a query using a keyword filter
3. Add other filters to a query
4. Upload a file to have the text content extracted
5. Store a copy of the upload file on server
6. Returns a list of files from Elasticsearch that match the query filters
7. Query the data stored in Elasticsearch that matches the given filters
8. Stores the metadata and content of the uploaded file into elasticsearch

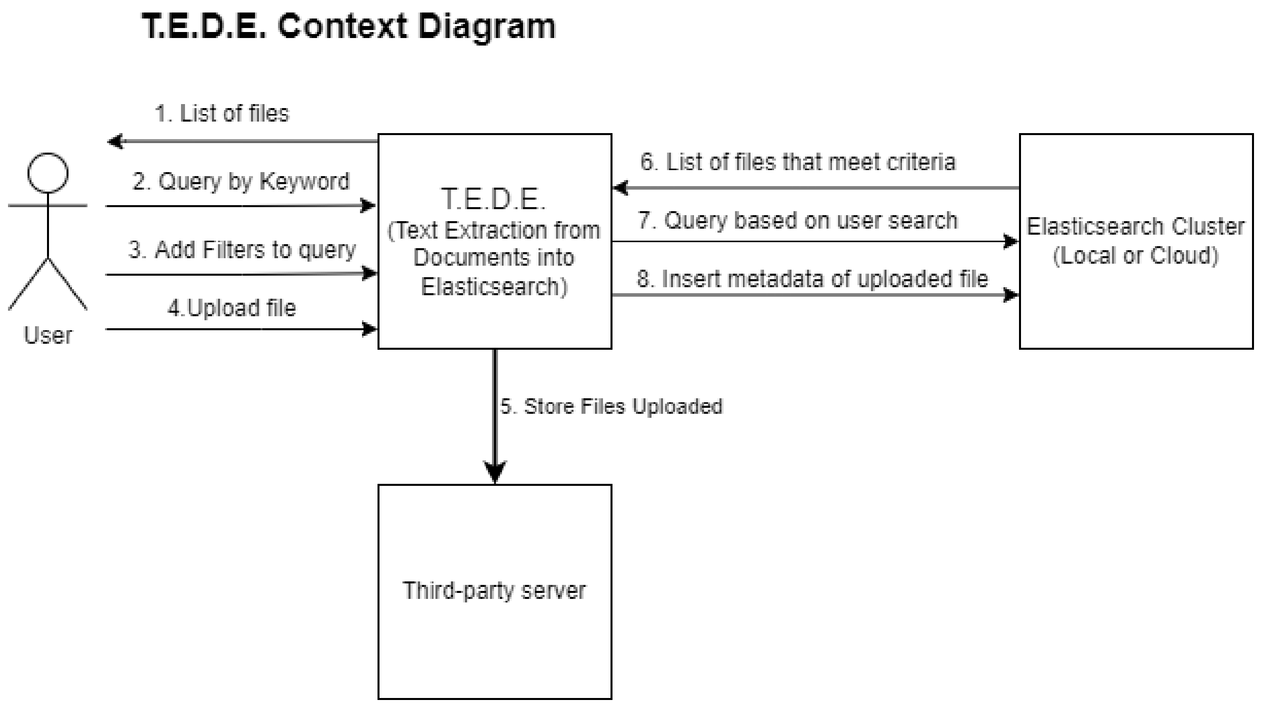
T.E.D.E. Context Diagram
1. List of files
2. Query by Keyword
3. Add Filters to query
4.Upload file
User
T.E.D.E.
(Text Extraction from Documents into Elasticsearch)
6. List of files that meet criteria
7. Query based on user search
8. Insert metadata of uploaded file
Elasticsearch Cluster (Local or Cloud)
5. Store Files Uploaded
Third-party server

*Figure 1: TEDE Context Diagram*

The following diagram, Figure 2, shows a breakdown of the components of TEDE. The components are responsible for the following responsibilities:

- Web Application System:
  - Search Bar: Handles the applying keyword and other filters from the user and creating a query call.

    - Upload Cache: Initializes an upload of a file to store the file on the server and triggers a text extraction/ index of the files contents.
- Apache Tika System:
    - Metadata: Extracts the metadata and text content from the given file and calls Elasticsearch to index/store the data
- Elasticsearch System:
    - Cluster: Insert and retrieves data from the Elasticsearch cluster

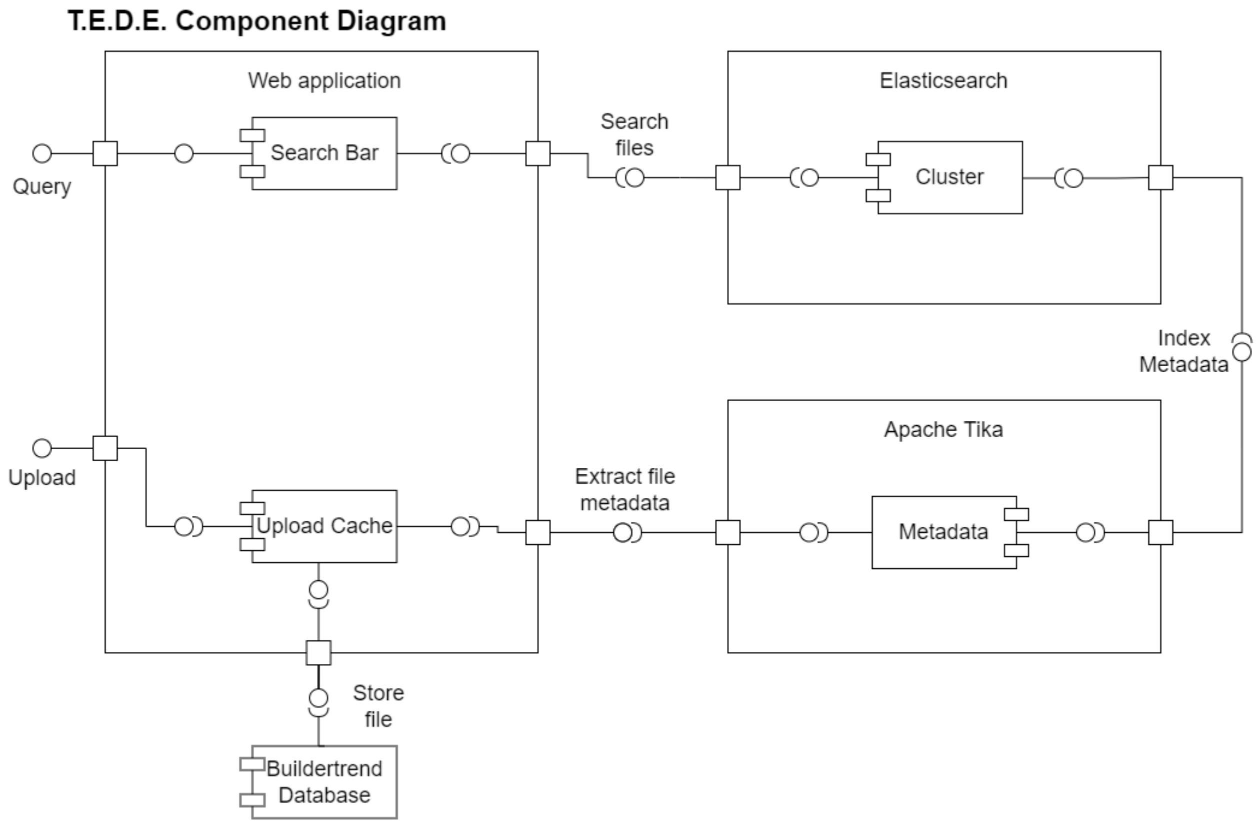

*Figure 2: TEDE Component Diagram*

### 3.3.2 Functionality

Users of our application will be able to search using keywords and search filters to find relevant documents. It will also be able to extract text from uploaded files and store the important information from the file. The application would support various file types like txt, docx, pdf, pptx, and so on.

Our current design addresses these function needs for the specified file type but may need to be adjusted based on what is attainable. For the UI/UX requirements we might need further client approval.

### 3.3.3 Areas of Concern and Development

We have the following concerns for our current design, as well as our plan of handling them.

- Uploading docs:
  - Concern: No clear specification of what this feature entails.
  - Plan:  Plan is to be able to upload document types that we can search. As we scale our search document types, we also diversify our upload document types.

- UI/UX:
  - Concern: No detailed specifications for this one either. We will have to reiterate our design until we get approval.
  - Plan: Propose a UI design and receive feedback. And, keep reiterating till we get final approval.
- Accuracy of results:
  - Concern: No qualitative measures specified by the client.
  - Plan: For now, we'll just follow the traditional measure - if the result fits the user's need or what the user was actually looking for.
- Query speed:
  - Concern: Client hasn't specified anything yet. Nonetheless, we want to return our results within 10s. Anything longer than this, we'll have to rework and optimize our algorithm.
  - Plan: Develop a search algorithm and keep optimizing it until an acceptable speed is attained.
- Scalability of doc types:
  - Concern: The list of document types is quite open-ended. Less of concern and more of a development.
  - Plan: We would be starting with txt and docx files. As we find success, we'll progress to other types like pdf, ppt, jpeg, and so on.

Overall, most of our concerns can be resolved with better client specifications.

---

## Below sections to be completed for final design

## 3.4 Technology Considerations

Highlight the strengths, weakness, and trade-offs made in technology available.

Discuss possible solutions and design alternatives

## 3.5 Design Analysis

– Did your proposed design from 3.3 work? Why or why not?

– What are your observations, thoughts, and ideas to modify or iterate over the design?

## 3.6 Design Plan

Describe a design plan with respect to use-cases within the context of requirements, modules in your design (dependency/concurrency of modules through a module diagram, interfaces, architectural overview), module constraints tied to requirements.

References:

Sharma, Manish, and Rahul Patel. “A Survey on Information Retrieval Models, Techniques and Applications.” *IJETAE*, vol. 3, no. 11, Nov. 2013.