**Design Document**

This document explains the design decisions that were made throughout the development process. We touch on a few algorithms and decisions that were made by the previous research teams that affect our product directly.

**Quality Attributes**

Most of the design decisions are influenced by the quality attributes that the system should possess at any time during the development. We address these attributes according to the overall requirements of the client. We have listed the quality attributes that were given prime focus according to our sponsor’s requirements.

1. **Accessibility**

The previous teams working on the trust based recommendation system had a research focus. The prototypes they built were not accessible to the intended audience which in turn resulted in lack of proper feedback. Therefore, one of our main focuses was to make the system accessible via modern browsers at all times. This way, the product could be accessed by any user and we could integrate continuous feedback into our system

1. **Usability**

When we achieve the accessibility, the next biggest concern will be how usable the system is. Usability has got two main aspects; the ease of usage and the performance of the system. Our focus was to improve user experience by it was a natural choice to use a graphical user interface.

1. **Extensibility**

The systems provides users with information about publications and researchers in a particular field. It also calculates a trust score based on knowledge and social factors. To calculate the most accurate trust score and to porvide users with the most accurate information about the research happening in any field, it is important to be able to extend the system to add varied data sources. The whole product has been developed in such a way that multiple data sources can be integrated into the system at any point of time.

**System Architecture**

**Dynamic View**

Since the system focuses on consuming big data and constructing meaningful representations out of it, dynamic flow of the data becomes an important aspect of the architecture. The following schema diagram shows this flow as well as the underlying physical devices that are required for deployment.

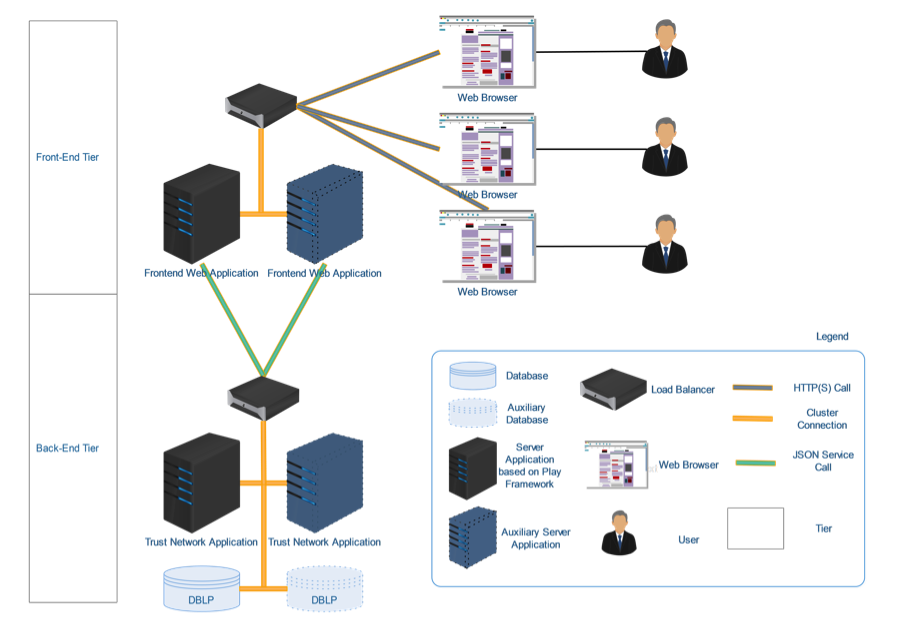


Figure 1: Dynamic View of the System

As seen in Figure 1, the overall system follows a two-tiered architecture for better maintainability. The back-end tier consists of the processed data to be consumed and the trust modeling application based on that data. The front-end tier consists of the web application to be served to the users. As shown in the figure, the number of servers can be extended according to performance requirements and scale. This will require some load balancer servers to distribute the request load to the servers in a reasonable way.

As mentioned above, the two-tier architecture let us enhance modularity of the whole system. The data is served as JSON and the front-end tier applications consume them via service calls. Therefore, the back end system is completely independant of the front-end system and can be used to serve another application in need of a similar back-end engine

Play framework has been used for both the back-end and front-end applications. Play is built on top of the asynchronous framework, Netty. Play is an open source framework developed in Java and Scala. The static nature of the back-end and front-end tiers improves the efficiency of the overall system. Another motivation to use the same framework for back-end and front end was the ability to share the models that we have used for data representations

**Static View**

Static view refers to models and objects defined for the system. Since the publication network includes different types of publications, we have come up with a hierarchical structure to represent the data in an efficient and easily maintainable way. The schema diagram below shows the hierarchical structure and relation between all the tables.

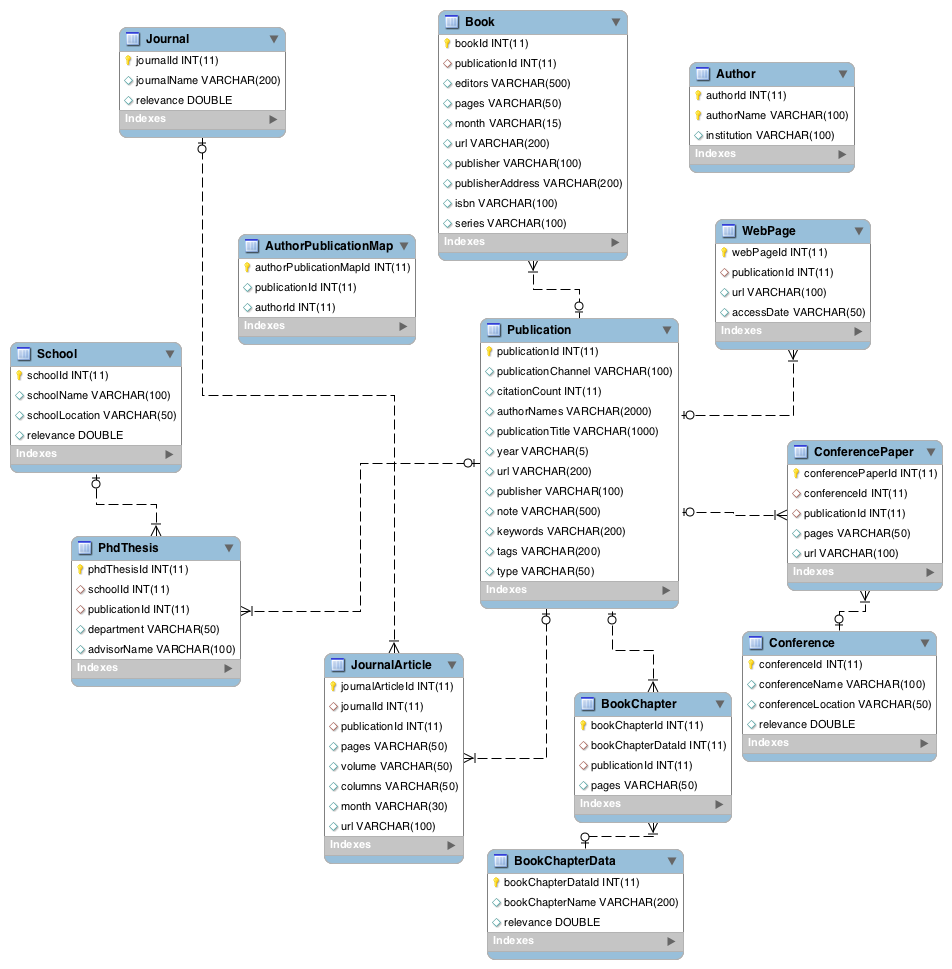


Figure 2: Database Models Schema

How Attributes are Met

The previous team who worked on this project had developed a desktop-based offline application that was not accessible to the client. Therefore, the solution of a web-based application will fulfill the accessibility requirement as suggested in previous sections. The web-based application that provides graphical helps achieve the goal of better usability. Multi-tiered architecture as well as normalized data modeling makes the system easily extensible.