BRITISH COUNCIL

Research

PROJECT REPORT

Acastry: An AI-based System to Develop Collaboration between Academia and Industry

Project Funded by the British Council under Researcher Connect Grant 2022

Chief Investigator

Md Saidur Rahman Kohinoor
Lecturer, Department of CSE,
Leading University

Research Assistants

Ayesha Loylus Miah, Graduate Student, CSE, LU
Shah Fayez Ali, Graduate Student, CSE, LU

Mentors

Md Sabir Hossain
Assistant Professor, CSE, CUET

Md Zafrul Alam
UI Developer, Marine Scotland

www.britishcouncil.org/
research-policy-insight

Abstract

This study makes an effort to address the disparity brought on by the growing gap between industry and academia. The ongoing challenges that students encounter at the industrial level are a result of the obscurity that is brought on by a lack of skills and incoherent academic tasks compared to the complexity at the professional level, among other factors. Acastry is an AI-based solution to the existing problems by creating a web platform to connect firms and academic institutions and could possibly improve the present situation. For this, the widely used Natural Language Processing (NLP) techniques are examined in the context of extensive data. The literature review on sub-module methodologies has led to the development of a deft choice that will assist academics, professionals, and students. The proposed methodology captures the essence of industry-academia collaboration as course instructors choose the tasks for the evaluation of the performance of students in concerned courses with respect to industrial-standard tasks on the basis of automatic keyword extraction where we developed a new method named “PAKE – PoS tagger Augmented Keyword Extraction” with a higher accuracy and precision. The keywords are obtained from titles and descriptions of assigned tasks by both sectors that would further help in recommending the relevant tasks for the students with respect to the course being taken. In a nutshell, Acastry intends to build an academic-industry partnership and possibly impact the future by bridging the gap.

Content

List of Figures

List of Tables

List of Algorithms

CHAPTER I
INTRODUCTION

The detachment between academia and industry has been growing for many years for various reasons, which has indirectly impacted students' capacity to perform well in the real-life work environment. Undoubtedly, we require a solution that will address the complexities of this gap and give students the idea about the skills they need to work hard and prepare them for job market. For this, we must deliberately reduce the gap's ever-widening size, and our suggested system is a viable means of overcoming the challenges brought on by the enormous gap.

1.1 Background

The high unemployment rate hampers Bangladesh's progress among university graduates and firms' difficulty in hiring skilled workers. In the most recent report from the World Bank, graduates reported difficulty finding employment, with more than a third of them still looking for employment one or two years after receiving their degrees. In accordance with the same study, 66 percent of graduates from National universities (colleges that award degrees in tertiary education) are unemployed, which is an appalling situation [1].

Employers conspicuously feel that graduates looking for work lack the necessary skills. A lack of qualified candidates for professional positions was reported by 46% of employers in a 2021 study by the Centre for Policy Dialogue (CPD). Lack of required qualifications (32 percent of employers) and lack of work experience (42 percent of employers) are two significant additional barriers that employers have identified in their search to find the most qualified applicant for a job opening (35 percent of employers). Employers cited work experience (51 percent), hard skills (65 percent), and soft skills (83 percent) as the top three most significant factors considered when hiring new employees [2].

While academia focuses on creating new knowledge through research and providing education to students, industry frequently focuses on addressing solutions that are of near-term commercial value. The lack of industry – academia collaboration is one of the factors contributing to the skill mismatch. Many subjects, including software testing, project management, quality assurance, technical writing, ethics and professionalism, leadership skills etc. have been found to be taught to students through their employment. To help students develop their skills, fundamental industrial practices must be taught in academia.

As indicated earlier, it has become more and more obvious that reducing the gap between universities and industries is crucial. For technology to innovate and grow, industry and academic cooperation are essential. The results of this kind of partnership between private businesses and academic institutions have always been excellent for both industries and academia, especially in the ICT era of modern technology and the 4th Industrial Revolution. The creation of such a platform for collaboration is an extremely timely and well-planned initiative.

1.2 Brief Overview of the Proposed System

The proposed study design focuses on developing and designing an interactive system based on the fundamental concepts of artificial intelligence, machine learning, and data mining. The system allows course lecturers to incorporate industry-specific tasks and assignments while providing students with real-world experience. It also aims to serve as a recommender system for educators and industrial partners while assisting with the auto-extraction of keywords for relevant suggestions. We also choose the Agile methodology and Scrum, a simple framework for effective project management to benefit the research process through accountability, continuous review and adaptation, and teamwork.

Artificial intelligence (AI) is emerging as a prevalent technology having a profound impact on educational purposes. This research intends to utilize this aspect of AI and provide an industrially oriented learning environment. Here, the target audience comprises (a) students who may be assigned industry-specific tasks; (b) course teachers who see personalized recommendations based on courses, and create relevant tasks based on intelligent assistance and feedback from professionals; (c) industry partners who evaluate and assign tasks. Hence, the intended outcome of this research is to reduce the academia-industry gap and equip students with the required skill set for the jobs whereby fill the shortage of skilled workforce in the industry.

Although our proposed system won't solve the job problem entirely, it will be a significant step toward resolving the problem. We think this collaboration will result in positive changes for both sectors. Students would gain a thorough understanding of work culture by hearing from industrial professionals about their knowledge and work behaviour. Industries can improve the professional behaviour of their employees by way of learning experiences at the institute level rather than spending money on expensive training later. Additionally, by hiring the students they are already collaborating with, they can also speed up the hiring process.

1.3 Objectives of the Study

The main goal of this system is to provide students with hands-on experience in the workings of an industry, to allow educators to keep an eye on their classes and establish a connection between the industry and the student, and finally to give the industry access to skilled workers. The long-term goal of the research is to bridge the industry-academia gap in order to maximize the potential of both sectors. The study has the following objectives and sub-objectives:

To develop a task recommendation system based on machine learning algorithms

- To create intelligent assistance for task description completion
- To undertake the generation of relevant automated keywords
- To analyze, evaluate and categorize the feedback received by an academic

The optimal sub-objectives with respect to the proposed web app are as follows:

- Enable teachers and students to conduct their academic activities online.
- Allow students to take part in collaborative projects where they can learn about industry practices and new technology.
- Enables students to work on real-world projects and expands employment opportunities for graduates.
- Using the faculty's new research knowledge from the collaborative activity, to maintain a current and relevant academic curriculum.
- Make it simple for industry practitioners to access qualified personnel.

The remainder of this report is organized as follows: the prior research with respect to the techniques used for the specific modules of the system will be briefly discussed in **Chapter 2**. Then, we elaborate upon the suggested model in **Chapter 3**. After that, the intricate details of the structural design and the core features of the suggested web application are expanded upon in **Chapter 4** and **Chapter 5** respectively. **Chapter 6** highlights dataset used with the chosen metrics followed by our discussion on the comparison analysis. Finally, **Chapter 7** concludes the report with possible suggestions and improvisations for future implementations.

CHAPTER 2
Literature Review

It is well-acknowledged that collaborations between academic institution (specifically higher and technical education) and business organization have the powerful prospects to ensure high gains to both participating entities, such as research validation context for the institution and increased access to new knowledge for the organization. But there has been very limited number of research and holistic studies conducted on best practices for maintaining a proper bridge in this regard. Unfortunately, most of them; specially in Asian subcontinent, are far from complete, realistic, and usually focuses on the software and computing industries.

As part of a larger project reducing gap and governing beneficial actions for establishing effective, sustainable, collaborative relationships among university authorities, academicians, industry professionals, and students, we have reviewed the following available materials [3] –[6] where discussed about the key factors of the bi-directional reflex between the entities which helped us to identify the gaps that exist in course plan, teaching and evaluation approach, and practical experience relating to preparation for careers.

Apart from that, we have keenly observed numerous studies deal with extracting keywords automatically and recommendation systems to make the system more convenient, interactive and ensure simplicity. We will in-depth discuss some of the key revelations of each, as well as potential options to aid in moving forward with our work. This thorough literature review assists in choosing the strategy for our proposed work.

2.1 Automatic Keyword Extraction

Keywords are an integral part of information retrieval systems as well as have several applications in the field of Text Mining, and Natural Language Processing (NLP) [7]. These terms capture the essence of the documents and reflect their overall content, making it simple for readers to anticipate their areas of interest. Most important information is often represented in the document with terms such as key phrases, key segments, key terms, or simply keywords; and are frequently used interchangeably. All of the aforementioned synonyms have the same function: they describe the subjects covered in a document [8].

The primary issue is that many documents do not have keywords assigned to them yet and doing so manually is undoubtedly time-consuming. Automatic keyword extraction can help to solve this problem. It concerns “the automatic selection of important and topical phrases from the body of a document” [9]. In other words, it is a text-based information processing task that entails the extraction of representative and characteristic phrases automatically from a document expressing all its salient aspects.

Moreover, such methods fall roughly into two categories: (1) keyword assignment and (2) keyword extraction [10] – [12]. These focus on a similar issue: picking the ideal keyword. When assigning keywords, documents are grouped into classes based on their content, and keywords are selected from a predefined taxonomy or controlled vocabulary [7].

A document is enriched with keywords mentioned explicitly in the text through keyword extraction. The words that appear in the document are examined to determine which ones are the most indicative, usually by looking at the source properties or heuristics and not using a thesaurus to choose the keywords [13].

However, automatic extraction of keywords is harrowing due to the complexity of natural language, diversity in input documents, and the various kinds of keywords that must be extracted [14]. One of the most significant challenges is defining relevance, which is difficult when dealing with texts of various sizes, concise texts, genres, and languages. There are also problems with the exact match restriction, the issue of missing keywords, and producing a large number of candidate keywords from a single text [15]. These problems demonstrate how challenging it is to create an all-encompassing solution that works effectively in every situation and drives the demand for additional study.

Methods for extracting key phrases can be either supervised or unsupervised. Unsupervised approaches are more favourable as they are domain independent and do not require labeled training data. Hence it is preferable to supervised approaches, which have subjectivity problems and need a sizable time and financial investment [14]. On the other hand, previous studies show that supervised methods outperform unsupervised ones in terms of accuracy and have more robust modeling capabilities.

2.1.1 Unsupervised Methods

An unsupervised keyword extraction approach generally involves the following steps: determine candidate terms based on specific heuristics and techniques; rank the candidate terms; and finally, form keywords from the words' top-ranked ones [16]. These methods are further divided into techniques based on statistics and those based on graphs.

Statistics-based Approaches:

TF-IDF, which stands for Term Frequency and Inverse Document Frequency, is the most common baseline for unsupervised approaches. Many TF-IDF variations appear to be effective [15] and can help score schemes, such as the one suggested by Florescu and Caragea [17]. In addition to TF and IDF scores, KP-Miner [18], introduced by El-Beltagy and Rafea, also employs statistical data, a scoring function, and highly efficient filtering of candidate terms. The system also considers a cutoff constant (CutOff), determined by the count of words after which a phrase first appears and the least allowable seen frequency (LASF) factor. The system ranks each candidate phrase based on the TF and IDF scores, term position, and a boost factor for compound terms over single terms.

The co-occurrence statistics and statistical metrics based on external sources were used to determine the semantic similarity between the document's candidate terms. Liu et al. present KeyCluster [19], a keyword extraction tool to extract key phrases covering each document's significant topics. It clusters the candidate terms based on several metrics such as co occurrence-based or Wikipedia-based and uses spectral clustering to determine how semantically related they are. Lastly, extracting keyphrases from the given text locates exemplar terms of each cluster. Rapid Automatic Keyword Extraction (RAKE) is another instance of a keyword extraction system developed by Rose et al. [20], which uses co occurrence matrices. To divide the text into candidate phrases, RAKE uses a list of stop words, a set of phrase delimiters, and a set of word delimiters as input parameters. Then, a word-word co-occurrences graph is made, and each candidate phrase gets a score based on word frequency, word degree, and the ratio of the degree to frequency, which is the sum of the scores of the words that make up the phrase. In addition, RAKE can identify key terms containing interior stop words with the help of adjoined word pairs occurring at least twice in the corresponding document, in the exact order. Finally, the ultimate keywords for the document are the top T-ranked candidate words.

Campos et al. highlight the importance of using statistics and context information with YAKE. YAKE first pre-processes the text into independent terms. Second, a set of five features is calculated for each term. Third, the candidate term score for each word is then computed using all of these features, emphasizing the importance of the word based on how smaller the value is. Finally, a sliding window of 3-g is used to create a continuous chain of 1-, 2-, and 3-g candidate keywords [15]. We explore the work put forth in YAKE in developing our approach, which we will be elaborated in the next chapter

Furthermore, the method put forth by Won et al. shows that combining basic textual statistical features can yield outcomes that are on par with state-of-the-art techniques [21]. This method begins by selecting candidate phrases based on morpho-syntactic patterns. Next, the following characteristics are calculated for each candidate: Term Frequency (TF), Inverse Document Frequency (IDF), the cumulative probability using Relative First Occurrence, Candidate Frequency (k), and Length, a straightforward rule that assigns unigrams a score of 1 and the other sizes a score of 2. Finally, these are multiplied together to produce each candidate's final score [21].

Graph-based Approaches:

The fundamental concept behind the graph-based ranking technique is to extract candidate keywords from the document and turn them into a graph, where each graph edge connects related candidate keywords [14]. The ultimate aim is to rank the nodes using a graph-based ranking method or to solve a proposed optimization problem on the graph. Mihalcea and Tarau proposed the first graph-based keyword extraction method, TextRank [22]. This algorithm determines the significance of a vertex within a graph by recursively computing global information from the entire graph rather than relying solely on local vertex-specific information. The pre-processing phase involves tokenization and annotation of parts-of-speech tags to apply syntactic filters, followed by the post-processing phase, which consists of the construction of an undirected unweighted graph on which the improved PageRank algorithm [23] is used until it converges.

Wan and Xia introduce SingleRank [24] as an add-on to TextRank [22] that subsume weights to edges. The quantity of co-occurrences between the two relative words determines the edge weight for each edge. For each consecutive string of nouns as well as adjectives in the text, the summing up of each word score is done in a post-processing stage, and as a result, the top T candidates as key phrases are returned. Along with emphasizing documents that stand alone from one another, the authors broaden the scope by using ExpandRank [24] to consider data from documents that are close to the target document, where a vector of TF-IDF scores represents each document.

SGRank [25], developed by Danesh et al., extracts all n-grams from the given text and filters out any that contain punctuation or words other than nouns, adjectives, or verbs. In the second phase, the candidate n-grams are ranked using a modified TF-IDF variant like KP-Miner. Finally, the top candidates are reranked in the third phase using additional statistical heuristics, such as term length and position of the first occurrence. The final ranking of keyphrase candidates is computed by incorporating the stage three ranking into a graph-based algorithm leading to an overall performance boost of the system. Likewise, Florescu and Caragea propose PositionRank [26], another unsupervised graph-based technique to identify familiar locutions by considering word-to-word co-occurrences in accord with the positions in the text. The key terms are then evaluated and ranked.

2.1.2 Supervised Methods

Turney performed better than general-purpose algorithms of machine learning with a custom designed algorithm called GenEx, a hybrid of the genetic algorithm and parameterized key extraction algorithm (Extractor) [9]. The Extractor operates by the assignment of numerical scores to each phrase in the given document and eventually produces a list of phrases with the highest scores. Twelve numerical parameters influence the scoring function's behaviour. The Genitor adjusts the parameters to enhance the Extractor's performance on the training data. The study is one of the first attempts at automatic keyword extraction from texts as a supervised learning binary task. KEA [12] is another benchmark implementation, which uses the Naive Bayes algorithm along with TF-IDF and the term's first occurrence to determine what is and is not a keyword. Many other pieces of work are based on enhanced versions of KEA, such as those proposed in the given study [10] and [27].

Hulth [28], depicts the plausibility of using simple statistical measures and syntactic information from given documents as input to a machine learning algorithm for successful keyword extraction. The author then juxtaposes the efficacy of the chosen approach: noun phrase (NP) chunks, n-grams, and POS tags, with four discriminative features of these terms as inputs to a rule induction with bagging supervised machine learning algorithm. CeKE [29] is another supervised algorithm that combines data from the citation networks of research papers to use the information from the neighborhood, where conventional features are obtained from the given input document itself. Many other strategies make use of neural networks and deep learning models. Meng et al. proposed one such model, which predicted keywords from scientific texts using a neural network deep learning model. Gollapalli and Li [27] looked at the notion of extracting acceptable keywords from research papers as a sequence tagging task. The studies elucidated in [16] and [14] provide a more comprehensive deliberation and thorough review of extracting keywords

2.2 Recommendation System

A recommendation system plays a pivotal role in decision-making when there are many options to choose from. This scenario arises quite often in the modern world where a user of Web 2.0 is flooded with so much information. This makes it challenging to analyze the relevancy of the item the user is being bombarded with. Recommender systems were created primarily to assist people who lack the experience or knowledge to navigate the bewildering range of options available. In the e-commerce world, the users can be provided with appropriate products, whereas concerning entertainment, the suggestions could help to decide what movie to watch, what music to loop next, or what video would be worth watching. Many platforms try to tailor their recommendations to fit the user's learning style and interests regarding the educational component of recommendation systems. Nonetheless, even the tech giants continually try to evolve their recommenders to provide the best to their consumers.

A growing body of literature acknowledges the significance of recommender systems. It has resulted in extensive research in this field because it has the potential to benefit businesses and, as a result, has been the focus of great concern in both academia and industry for the past few years [30]. It uses numerous sources of information to predict users' preferences for items of interest. We can say that recommender systems are designed to determine whether an item is worth recommending and to estimate its utility. According to the study [31], a recommender system's fundamental component is:

$$f: E \times P \rightarrow S$$

This function determines a particular item's (P) utility. The ultimate list of recommendations contains a selection of ranked items based on the utility of whole items the user has not yet explored. Recommender systems maximize the utility function, formulated as follows to find the suitable product for the end user:

$$\forall e \in E, \arg\max_{p \in P} f(e,p)$$

In terms of the recommendation algorithm, utility of any items differ for a specific user. Previous studies categorize recommendation techniques as content-based, collaborative filtering-based, or knowledge-based. Before attempting learning modules in an e-learning environment, a recommender can offer recommendations for familiarization activities [32]. Besides that, it creates "user rating resource matrices" and makes personalized recommendations for learning materials based on users' learning preferences and learning paths in online learning systems.

The following subsections will go over common approaches for a recommendation system.

2.2.1 Content-based Recommendation Systems:

This approach uses an item’s content description to predict its usefulness depending on the profile of the user. The systems-based solution aim to suggest products comparable to those that have previously piqued a particular user's interest. Several different item attributes are usually extracted from documents obtained directly from structured or unstructured data. For instance, TF-IDF weighting depended on vector space model used as one of the most widely retrieval techniques where a keyword-based model to obtain the values associated with the properties [33]. Next, these types of systems generate a profile of a user from the consumption records and preferences of that user, which typically includes information about the user's previous likes or dislikes. Thus, this is also viewed as a typical binary classification problem, and this step may employ traditional approaches like as Naive Bayes, nearest neighbour algorithms, and decision trees [34], [31]. Shu et al. [35] used the student data that had been subjected to a content-based recommendation algorithm to make predictions about instructional materials. Furthermore, Rahman and Abdullah's [36] system uses decision trees as well as fuzzy clustering to make it possible for a recommender to suggest web-based, learner-centric materials on any e-learning platform by categorizing

The system creates the user's profile, compares the profile with the attributes of item, and identifies the items that mostly pertinent to use as the basis for a recommendation list. Therefore, content-based recommenders are more likely to filter and match the representationof the item and profile of the user obtained in the first couple of steps in terms of the features. Accordingly, accuracy of the representation of item and user profile determine whether the matched items are forwarded and whether the items the user dislikes are removed [37]. There are several benefits to the content-based recommender system [38], [39]:

1. A content-based recommendation is user-independent since it is based on item representation. Therefore, the data sparsity issue does not affect this type of system.

2. Content-based recommender systems can address the new item cold-start issue by recommending new items to users.

3. Content-based recommender systems can explain the recommendation outcome in detail.

Content-based recommender systems do, however, have some drawbacks [30], [31]. The new item problem is not available in these systems, but the new user problem still they have. The reason behind that the shortage of information regarding the user's profile significantly impacts the recommendation result and its accuracy. Additionally, these systems always present users with comparable items, which results in overspecialization of the suggestions. The fact that items are not always easy to represent precisely by such recommenders is one of the many cons. Hence, content-based recommendation systems may not be suitable for music and images.

2.2.2 Collaborative Filtering-Based Recommendation System:

Collaborative recommendation systems generally use user ratings to deduce the utility of an item. The fundamental premise of the CF technique is that users with comparable interests will purchase similar goods. As a result, a system that employs collaborative filtering relies on data provided by users who share the same preferences as the given user. Memory-based and model based collaborative filtering are the categories into which CF-based techniques are divided. Memory-based CF is the oldest one to calculate similarities between items or users by using heuristic algorithms and subdivided into item-based CF and user-based CF based on the results [30].

Memory-based collaborative filtering fundamentally employs the nearest neighbour algorithm. Based on the neighbour ratings of a user or item, the recommendation calculates and ranks the target user's rating on various items. This technique has simple implementation and generally effective and practical application producing precise results but still has some insignificant constraints [31]:

1. The cold-start problem is unable to solve. That means there's no possibility to produce prediction for a new user/item while it entersthe system since there's no rating available to utilize.

2. If an item is not popular with consumers despite being old, it may have too poor ratings which will limit the coverage of that item's recommendation.

3. Providing real-time recommendation is not a part of these systems.

Instead of using heuristic techniques, the model-based collaborative filtering approach is about creating a model to predict the rating of a user on items using either data mining or machine learning algorithms. Although this method was initially developed to address the flaws arising from memory-based collaborative filtering [30], it is being extensively used in resolving issues in other fields. This technique involves the use of matrix factorization, which is beneficial for the following reasons: To begin, the user-item rating matrix's dimension can be dramatically reduced, ensuring the system's scalability. Furthermore, to mitigate the sparsity issue, a dense rating matrix is formed in the process. Besides that, matrix factorization significantly enhances memory-based methods, enabling users with lower ratings to receive more accurate recommendations. Hence, it proves to be an excellent tool for incorporating various other information to aid profile user preferences and facilitate the systems' performance.

2.2.3 Knowledge-Based Recommendation Systems

These systems provide suggestions from previous knowledge or rules already in place regarding user needs and item functions. Knowledge-based recommendation systems [30] maintain a knowledge base with information extracted from a user's prior records. The base consists of prior issues, limitations, and comparable solutions. When a new recommendation issue occurs in the system, it refers to the knowledge in the knowledge base.

Finding the similarities between items need more organised representations than content-based recommender systems. The current problem is compared to a prior instance in this procedure, and the solution is modified. The con of this technique is that the system management and setup cost while creating and maintaining the knowledge base is typically high. Nonetheless, it also has the following benefits: (1) This approach is particularly beneficial in the areas of financial services, healthcare decision support, and real estate sales. These services are distinguished by specialized domain expertise, and every case presents an exceptional circumstance. (2) The fact that prior knowledge has been acquired and stored in the knowledge base means that one benefit of this technique is that it eliminates the new item/user issue. (3) Users' ability to place restrictions on the results of recommendations is an additional benefit [40].

2.2.4 Hybrid Recommendation Systems

A hybrid recommendation system is a combination of content-based and collaborative filtering that combine characteristics of both approaches by merging individual predictions into one, adding content information to collaborative models, or obtaining final recommendations based on the combined rankings [32]. For example, content-based filtering systems can recommend "cold-start" items with no training data, but they have lower accuracy than collaborative filtering systems [32], [41]. In contrast, the collaborative filtering approach frequently produces accurate recommendations but fails when dealing with cold start items. Hybrid schemes attempt to aggregate these various types of information to produce efficient recommendation results [41].

In general, it appears that the research on recommendation systems has greatly benefited from the combined interest and efforts that industry and academia have invested in this field. Meanwhile, the primary goal and scope of this study also influence the choice of an appropriate technique for keyword extraction, while noting the numerous limitations outlined in the literature review.

CHAPTER 3
Methodology

In this chapter, we discuss the methods and techniques that helped us to achieve our desired output. It elaborately discusses each step to provide ultimate comprehension of the output we expect. We specifically emphasise on two modules of the system architecture and expand subsequently in the following subsections:

3.1 System Architecture

The proposed system targets users of three groups: academia, industry and students. When it comes to the educators from the academia, the focus is on the lesson plans they curate along with the assignments and projects. The evaluation of the task given influences the submitted tasks to generate relevant recommendations. This takes place with the cooperation from industry practitioners who also provide industry specific and skill-based tasks and jobs in accordance with the specific course conducted by the respective course instructor. These practitioners evaluate and respond with feedback for the improvement of future submitted tasks. The following Figure 3.1 illustrates the system architecture of the proposed system and is followed by an overview of each module:

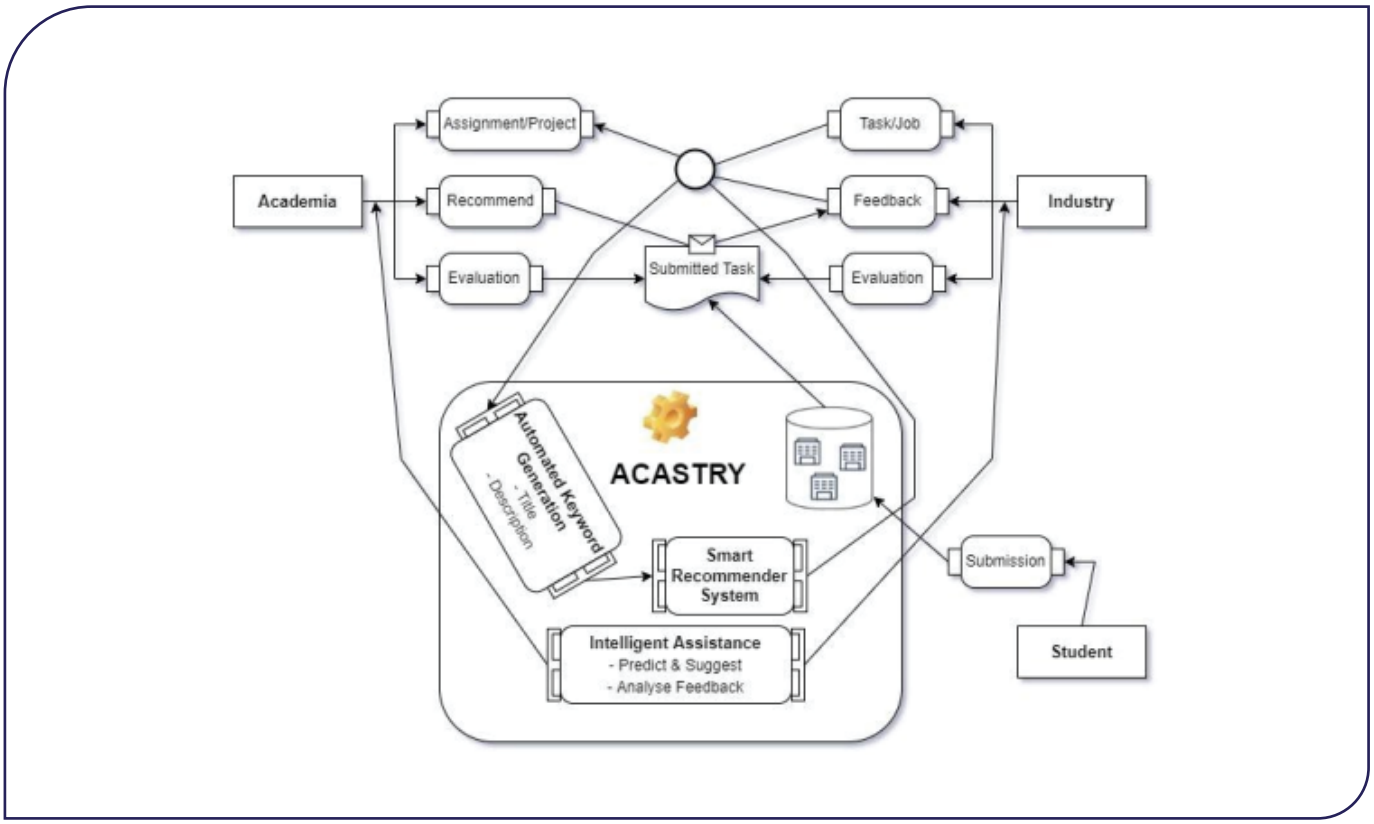

Figure 3.1: Overview of the Proposed System

Automated Keyword Generation: The function of this module is to extract keywords from the lesson plans, tasks' title and description assigned by the course instructors and partnered practitioners along with the timeline feed of instructors. This is done with the help of our proposed unsupervised approach that makes use of lexical terminologies as tags for the syntactic meaning while generating the n-grams with scores that describe their relevancy. The extracted keywords aid the other two modules with relevant keywords as input. The research on generating the keywords appropriately matching the skill set mentioned in the task title and description automatically is in progress and is a future endeavour.

Smart Recommender: A considerable amount of literature has been published on recommendation systems as reviewed in Chapter 2. An implication of this is the possibility that the best suited recommendation technique for our proposed system at the moment is content based. This is because the keywords extracted with the help of above-mentioned module is found relatively from the content of the lesson plans, timeline feed and so on. It revolves around comparing the keywords extracted from academic and the industry side using cosine similarity to suggest relevant assignments and tasks for the students.

Intelligent Assistance: The module emphasises on auto-completing the title and descriptions of the assignments with the help of the extracted keywords and feedback analysis from the partnered firms and practitioners. This would assist the particular user in assigning the tasks relevant with industrial level easily and making efficient use of the time. Also, this module would suggest similar types of public tasks to the user at the time of creating new one to avoid redundancy and assist for utilizing the previous feeds.

At present, Acastry organizes the suggested tasks with respect to all the users from the three groups by combining the following modules: Automated Keyword Generation and Smart Recommender. The function of each specific module is elaborated in the following sections.

3.2 Automated Keyword Generation Module

When it comes to keyword extraction from short texts, it proves to be a challenging task as depending solely on repetition seems ineffective due to the fewer amounts of words present in the text. This information retrieval task would require an efficient combination of additional conditions and techniques to extract more meaningful and relevant keywords. Concerning this, we proceed with **PAKE – PoS tagger Augmented Keyword Extraction**, an unsupervised approach for our system to extract keywords because there aren't enough appropriate labeling data for short texts. The features necessary to pick suitable candidate terms are chosen using a relative inference to YAKE's system [15]. We want to take advantage of the morphological syntactic value of PoS tags and propose an intricate improvement while concentrating on a statistics-based approach that considers term frequency and other potential influences on the overall success of coming up with the desired keywords. Table 3.1 provides an excerpt that would be used in this study to comprehend the process in its entirety further.

Table 3.1: Sample text from freelancer.com that will be used for demonstration [45]

Nvidia Jetson emmc full encryption with secure boot There isn't a built-in root filesystem encryption tool for the nvidia jetson line. They have secure boot and also bootloader encryption but not rootfs encryption. This is possible through customizing the u-boot to decrypt the fs although it is not built in by default. Looking for someone to build an image with encryption that I can flash to an xavier nx. Then help with instructions for enabling secure boot and testing resilience of system. Added details. This encryption and security method also needs to work with the jetson nano emmc. This will make it slightly harder as the nano can't encrypt the bootloader. Refer to this post [login to view URL] If someone can access the key mentioned in the post the encryption can be handled by storing a encrypted version of the root fs AES key in the bootloader. So, an attacker could read the encrypted storage key but only that particular jetson could unencrypt the AES key at boot and then mount the rootfs. Hope this helps. Skills: Linux, Security, Encryption, Hardware Security Module

POS tagging, which is also known elaborately as part-of-speech tagging, is a process of grouping words in a given corpus with respect to the specific part-of-speech depending on the context and the meaning of the word. It is one of the fundamental techniques in natural language processing (NLP) that helps analyze texts where tags represent the lexical terms and much more. These tags in a set are called tagset, which may differ according to language; for instance, the general datasets for the English language might consist of the basic parts of speech such as nouns, verbs, and adjectives. Moreover, one can customize and create personalized tagsets to complement their targeted study. The intention to use POS tagging in relation to the research objectives is predicated on the presumption that the pertinent tags indicate specific skills; consequently, they are likely to be a noun accompanied by an adjective or a verb, etc. This will make it easier to extract relevant keywords and influence the users to recognize the task's requirements. The algorithm introduced comprises five pivotal steps closely following the algorithm proposed in [15]:

- Pre-processing text and candidate term recognition
- Feature extraction
- Word score computation
- Generating n-grams and calculating scores of each candidate keyword
- De-duplication of data along with final ranking

The first and foremost step is critical because it converts the document to be comprehended by the machine to detect possible candidate terms. This stage is where the POS tagger applies to the text because POS patterns tend to indicate relevant keywords. The second step attempts to transform a list of individual terms into statistical features. These calculated features are put together into a single score to imply the term's significance in the third phase. The fourth phase entails generating possible keywords using an n-gram formulation methodology and scoring them on the

basis of relevancy. Lastly, data de-duplication using similarity measures like cosine similarity can ensure the production of the final list of candidate keywords without overlapping. Figure 3.2 gives a brief illustration of the process.

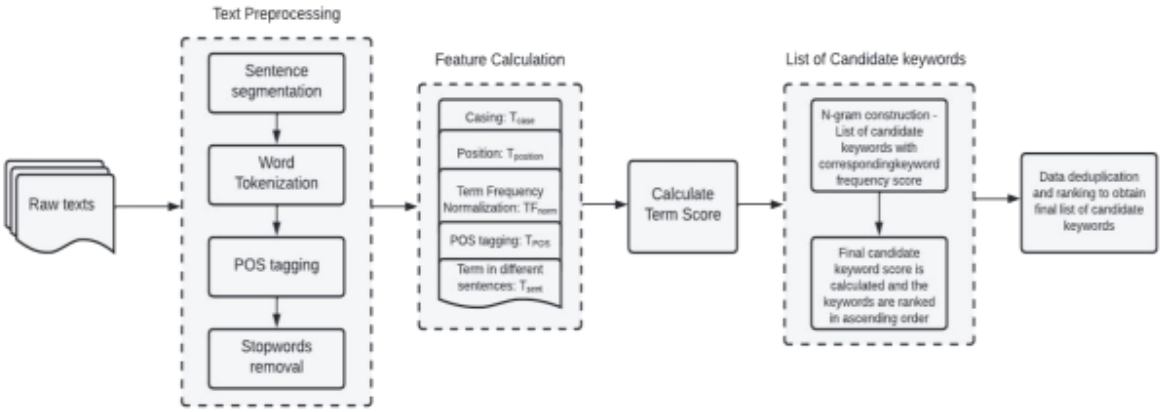

Figure 3.2: Illustration of the Proposed Algorithm

Pre-processing text and candidate term recognition: The text pre-processing stage generally involves cleaning the text, sentence segmentation, tokenization, stop word identification, stemming, and lemmatization to figure out relevant chunks from the text. The pipeline for our keyword extraction follows a similar order and involves text pre-processing with sentence splitting, text annotation, tokenization, and removing stop words. For this, we also use the rule-based segtok sentence segmenter [15] to divide the text into sentences based on a predefined pattern. This is shown in Table 3.2 where the sentence is split into chunks with certain tags that denote it as parsable content(<p>), has an acronym attached (<a>) and also includes tags to mark the uppercase characters (<U>). There are additional tags that prove to be useful for separating other information such as numbers and noisy data like email addresses, URLs and so on with the aid of the web tokenizer module of the segtok segmenter. Lastly, we get rid of any irrelevant terms by implementing the stoplists.

Figure 3.1: Overview of the Proposed System

So an attacker could read the encrypted storage key but only that particular jetson could unencrypt the AES key at boot and then mount the rootfs.

Chunk 1: <p>so<p>an<p>attacker<p>could<p>read<p>the<p>encrypted<p>storage<p>key<p>but<p>only<p>that<p>particular<p>jetson<p>could<p>unencrypt<p>the

Chunk 2: <a> AES

Chunk 3: <p>key<p>at<p>boot<p>and<p>then<p>mount<p>the<p>rootfs

After using PoS-tagger: ('so', 'RB'), ('an', 'DT'), ('attacker', 'NN'), ('could, 'MD'), ('read', 'NN'), ('the', 'DT'), ('encrypted', 'VBN'), ('storage', 'NN'), ('key', 'NN'), ('but', 'CC'), ('only', 'RB'), ('that', 'IN'), ('particular', 'JJ'), ('Jetson', 'NN'), ('unencrypt, 'NN'), ('aes', 'NN'), ('at', 'IN'), ('boot', 'NN'), ('and', 'CC'), ('then', 'RB'), ('mount', 'NN'), ('rootf', 'NN')

This part of the algorithm is demonstrated below.

Algorithm 3.1: Pre-processing Text and Recognition of the Potential Terms

```
Input: given text, lang
1. sents = separate given text into sents
2. start loop1 each sent in sents
3. chunk = separate sent into chunk
4. Start loop2 each c in chunk
5. tkns = separate c in tkns
6. start loop3 each tkn in tkns do
7. tag(tkn), to_lowercase(tkn), is_stopwords (tken, lang)
8. end loop3
9. end loop2
10. end loop1
Output: List of annotated sentences as sents and chunk
```

Feature Extraction: The feature extraction process begins after completing the statistical computation of each term. We mainly consider the following features – Casing, Position of the term in the sentence, frequency normalization of the term, the term appearing in different sentences and POS tag as per each term. The features are described in detail as follows:

Casing: This is a salient feature based on the assumption that the uppercase terms are usually more significant than the lowercase terms [15]. The aspect of selection is not only to consider the terms starting with a capital letter but also words that are a part of acronyms. Moreover, using the following heuristic, the error of calculating the occurrence of uppercase terms twice is avoided.

$$W_{Case} = \frac{\max(TF(U(t)), TF(A(t)))}{\ln(TF(t))}$$

Word position: The word's position throughout the document is crucial in determining the keyword's relevance. Many studies have considered this a feature to boost the system's retrieval performance. Moreover, the authors of YAKE [15] similarly assumed that words appearing in earlier sentences are more significant than those occurring later. Hence an increased value is designated to those words found in early sentences. Key phrases are unlikely to be found in brackets. As a result, they would receive negative scores during computation.

$$W_{Position} = \ln(\ln(3+\text{Median}(Sent_t))$$

Frequency normalization of the word: The feature shows the prevalence of the potential word t in the given document. Here, we intend to consider all those plausible candidate words whose frequencies are well above the average and balanced by a certain degree of significant dispersion.

$$WF_{norm} = \frac{TF(t)}{\text{MeanTF}+1*\sigma}$$

Word different sentence: This feature measures the occurrence of a candidate word within other sentences. It indicates the presumption that candidates appearing in various disparate sentences are more likely significant.

$$W_{sentDiff} = \frac{SF(t)}{\text{Total number of Sentences}}$$

Word POS: We use the POS tagger in the NLTK library to generate tags for specific words. This will help to identify the feature that will assess how much weight different parts of speech have. Based on the research of Hulth [28], noun chunks are more significant than verbs and adjectives. Adjectives, on the other hand, will be given a slight advantage over verbs. As a result, we intend to give noun chunks a weight of 1, adjectives a weight of 0.5, and verbs a weight of 0.25. The following Algorithm 3.2 and Algorithm 3.3 are developed based on these.

Algorithm 3.2: Term Statistics Calculation

```
Input: sents, chunk, w
1. words = []
2. start loop1 each sent in sents
3. start loop2 each c in chunk
4. tkns = split chunk into tkns
5. start loop3 i = 0..| tkns |
6. words [index (tkns [i])].TermFreq ++
7. words [index (tkns [i])].offset_sents += index (sent)
8. if tkns.tag == 'a':
9. words [index (tkns [i])].TermFreq_a ++
10. if tkns.tag == 'U':
11. words [index (tkns [i])].TermFreq_U ++
12. end loop3
13. end loop2
14. end loop1
Output: Terms' list with associated statistics
```

Algorithm 3.3: Computing Features

Input: words
1. start loop i = 0... |words|
2. words[i]. WCase = max(words[i]). TermFreq_a, words[i]). TermFreq_U)/ (1+ ln (words [i].TF)
3. words [i]. WPosition = ln ((3 + median (words [i]. offset_sents)))
4. words [i]. WFNorm
5. validTermFreqs = [word. TermFreq for word in words if not word. stop word]
6. averageTermFreq = mean (validTermFreqs)
7. stdTermFreq = stf (validTermFreqs)
8. words[i]. WFNorm = words [i]. TermFreq / (averageTermFreq + stdTermFreq)
9. words [i]. WPOS (word) -> 'NN'|| 'NNP' || 'NNPS' || 'NNS' = 1, 'JJ' = 0.5, 'VB' = 0.25
10. words [i]. WSentDiff = | words [i]. offset_sents | / | sents |
11. end loop
Output: Terms' list with calculated features and corresponding statistics.

Word score computation: The feature weights combine into a single score to calculate the word score [15]. The significance of the 1-gram term (t) increases as the word score value decreases. Algorithm 3.4 illustrates the computation of the word score. The first thirty single terms obtained after this process is given in the Figure 3.3.

$$Word_{Score} = \frac{W_{Position}}{W_{Case} + WF_{Norm} + W_{POS} + W_{SentDif\,f}}$$

Algorithm 3.4: Calculating feature score

Input: words
1. start loop i = 0... |words|
2. words[i]. S_w= WPosition/(WCase+ WNorm+ WPOS+WSentDiff)
3. end loop
Output: Terms' list with score of the word, comparable statistics, and computed features.

Top 30 words

encryption, 0.01753	emmc, 0.12271	decrypt, 0.28492
Jetson, 0.042247	root, 0.14151	default, 0.28492
boot, 0.045015	rootfs, 0.14814	build, 0.31263
Nvidia, 0.05234	nano, 0.15364	image, 0.31263
secure, 0.05533	post, 0.162327	flash, 0.31263
key, 0.06538	filesystem, 0.18244	xavier, 0.31263
full, 0.08891	tool, 0.182445	instructions, 0.33341
bootloader, 0.089272	line, 0.18244	resilience, 0.33341
built, 0.09900	encrypted, 0.25387	system, 0.33341
AES, 0.11765)	URL, 0.26978	details, 0.34972

Figure 3.3: Top 30 single terms

Generating n-grams and calculating scores of each candidate keyword:
We consider a sliding window with size n to produce a continuous term's sequence. Only term sequences that not only belong to the same sentence but also the same chunk create candidate keywords. By doing this, we can stop random formations of n-grams. Each candidate is given a score after completing the n-gram generation of keywords. The computation follows a similar process of splitting, tokenization, and recognition of stop words, but with particular emphasis on interior stopwords by using Bigram Probability, which considers only the conditional probability of one preceding word to approximate the probability of a word given all the prior words. In other words, irrespective of a high S(t) value, the interior stop words will attain a Bigram Probability score of 1.

$$S_k = \frac{\Pi_{t \epsilon k} \; S(t)}{KF(k) * (1 + \Sigma_{t \epsilon k} S(t))}$$

De-duplication of data along with final ranking: In our last step, we want to see if eliminating similar potential candidates using cosine similarity as another possible measure alongside Sequence matcher, Levenshtein similarity and Jaro-Winkler similarity measures would improve ranking results. The results are demonstrated in the following Table 3.3.

Table 3.3: De-duplication of Data and Final Ranking Output

Data de-duplication using Cosine Similarity and then ranked
emmc full encryption 0.00284
nvidia jetson line 0.00343
filesystem encryption tool 0.00439
secure boot, 0.01026
AES key, 0.032788

3.3 Smart Recommendation Module

Acastry is specifically designed to target the gap by mitigating the lack of well-oriented and aligned tasks that could integrate the industry-academia standards. The need for recommendations increases significantly as the growth in the number of assigned tasks will make it inevitable to make informed decisions to choose practical and balanced tasks. The module will focus on the keywords extracted from the course instructors' materials and industry-related tasks to make it easier for the course instructor to decide the relevant next activity/assignment for the students, precisely aligned with the outcomes expected in the real world.

All recommendation systems rely on data that is either explicitly or implicitly collected [32]. When the data is either collected by using the users' profile information or while the user registers via forms, then it is termed explicit data. It may also be collected from online user ratings. For instance, Acastry uses both the user information and the data obtained during registration for efficient suggestions. On the other hand, implicit data are unprocessed information that can be divided into two types: exhaust data which is secondary user activity whose usage varies depending on the recommendation task, and intentionally collected data streams such as search and timeline history etc. In addition, our system considers the timeline feed of the course instructor to extract relevant keywords to boost the overall performance of the recommender.

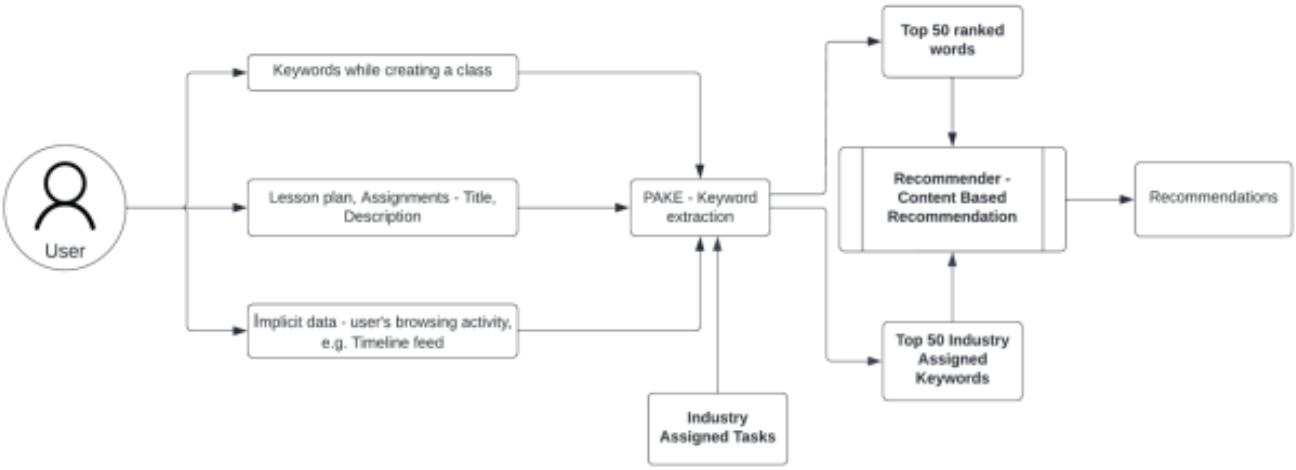

Figure 3.4: Illustration of the Proposed Recommendation Module

We intend to propose a recommendation system as shown in the Figure 3.4 using the content based approach for providing relevant suggestions. The underlying fundamental concept of content-based filtering is to compare previously consumed content with new items that may be recommended to the user. In other words, to suggest similar content that the user has been interested in formerly.

Content-based filtering depends on the following data: a set of users; and a set of keywords assigned to or extracted from the available item descriptions. Moreover, in the process of creating a user profile, the recommender will concentrate on two types of information: a user's preference model; and the user's interaction log with the recommender system.

The method uses an item profile with distinct attributes to qualify the item within the system. With respect to our recommender system, extracted keywords from the given content descriptions of lesson plans and assignments have been interpreted with applicable meta-information to determine a similarity between the documents.

The above Figure 3.4 shows the overall mechanism for our proposed content-based recommendation system, which includes the following step:

1. Obtain the relevant keywords for recommendation.

2. Compare the keywords extracted from industry-assigned tasks with the active user's preferences.

3. Recommend items according to features that fulfil the user's interests.

To elaborate, when creating a class, the user, in this case, a course instructor, selects keywords pertinent to the course and is stored in the database. PAKE, the proposed keyword extraction system, extracts relevant keywords from the lesson plan, title, and descriptions of assignments, as well as the content obtained from observing the timeline feed of the course instructor.

After successfully extracting keywords, the top 50 ranked keywords from input documents using implicit and explicit data and the top 50 ranked keywords from industry-related tasks are fed to the recommender. Here the recommender uses cosine similarity (the equation is given below), which makes it easier to group similar documents. As the output, the relevant suggestions show up in the results.

$$\text{cosine}_{\text{similarity}}(P,Q) = \cos\theta = \frac{P \cdot Q}{||P||\ ||Q||}$$

Overall, this study strengthens the idea that a possible content-based recommender system aided with PoS-tagger Augmented Keyword Extraction can be efficient for the proposed system concerning the needs of the users with respect to the expectations arising from the complexities of a particular job and in recommending the tasks that demonstrate the skills required to tackle such difficult scenarios.

Chapter 4
Structural Design

This chapter focuses on the intricate fundamental building blocks that keep the system in place. The data flow diagrams show how the design is implemented at various levels, while the ER diagrams show how important the entities and relationships are to the database. Together with the system flow chart, the use case diagram makes it possible to visualize the requirements at a higher level, which will ultimately aid in decision-making.

4.1 Data Flow Diagram

Data flow diagrams (DFD) display how information enters a system from external sources. Here, different DFD levels are displayed for better understanding the mechanism of functions in Acastry, including Level 0, Level 1, and Level 2.

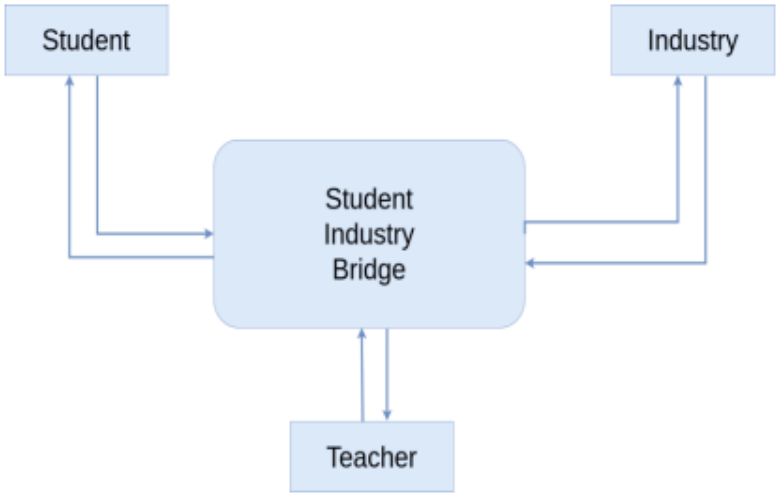

Figure 4.1: Data Flow Diagram (Level 0)

Level 0: The data flow diagram (DFD) at the context level describes the entire system. It intends to provide a quick overview of the system along with the system's relationship to external entities. Our system (Figure 4.1) has three external entities, including student, teacher, and industry (practitioners), who send and receive data through the system.

Level 1: First Level DFD of Acastry demonstrates how the system is broken down into smaller systems (processes), each of which deals with one or more data flows and stores data in a database.

According to the Figure 4.2, Process 1 is responsible for logging into the system. If the entity is a teacher, process 2 can be used to create a classroom and save data in the classroom database. Processes 3, 4 and 5 are used to depict further classroom activities including adding assignments, materials, and viewing the submitted assignments and grading the submissions.

A student deals with process 6, which allows him to enter a classroom. While an industry entity adds work and grade work submission utilizing steps 7, 8, and 9.

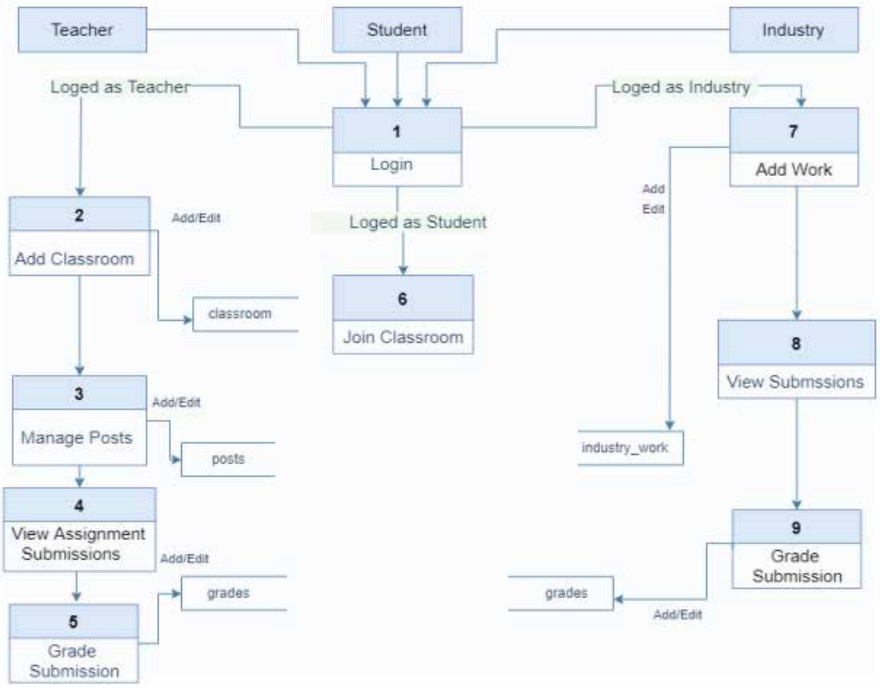

Figure 4.2: Data Flow Diagram (Level 1)

Level 2: At this level of DFD in Figure 4.3, more comprehensive information is provided on the academic and industrial activity. A student entity should be able to submit assignments and industry work tasks as shown in processes 7 and 8 if they have access to posts and industry works' data. After submitting a task, the student entity would also access the grade data provided by the teacher and the industry entity, as shown in process 9.

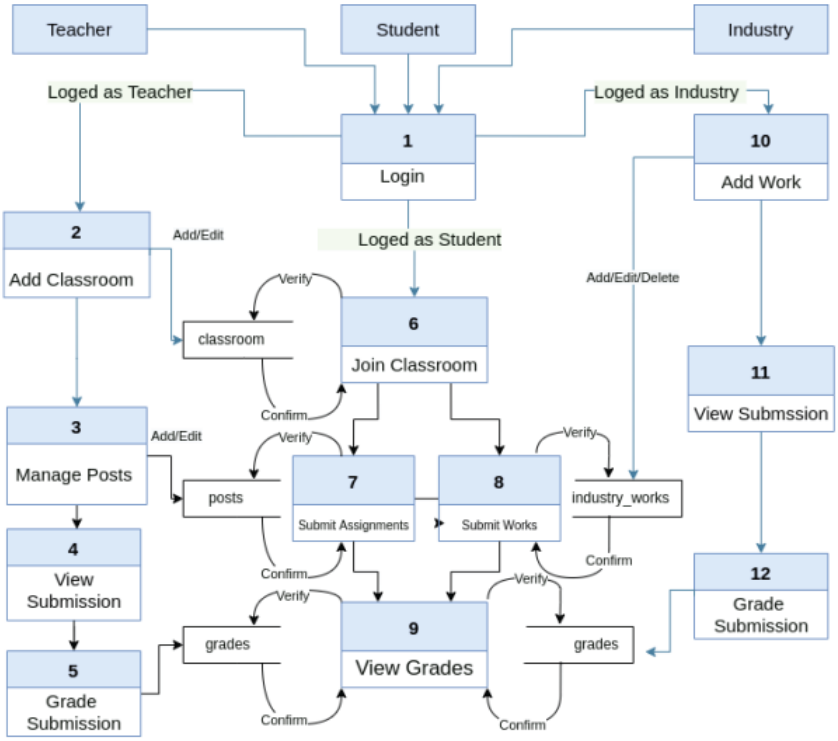

Figure 4.3: Data Flow Diagram (Level 2)

4.2 Entity Relationship Diagram

An entity relationship (ER) diagram shows the connections between entity sets that are stored in a database. Our ER-model is divided into three main subject areas, as shown in Figure 4.4, Figure 4.5, and Figure 4.6. The first diagram in Figure 4.4 depicts the connections between the entities for users, industry_works, and classrooms. Depending on the user role, a user can add multiple classrooms and industry tasks because the users entity has a one-to-many relationship with the industry_works and classrooms entities. Since there is a one-to-many relationship between the classroom_members and classrooms entity, multiple users can enrol in a classroom. The classroom_members entity stores member id and classroom id as a foreign key. Attribute role determines whether a classroom member is its host or not.

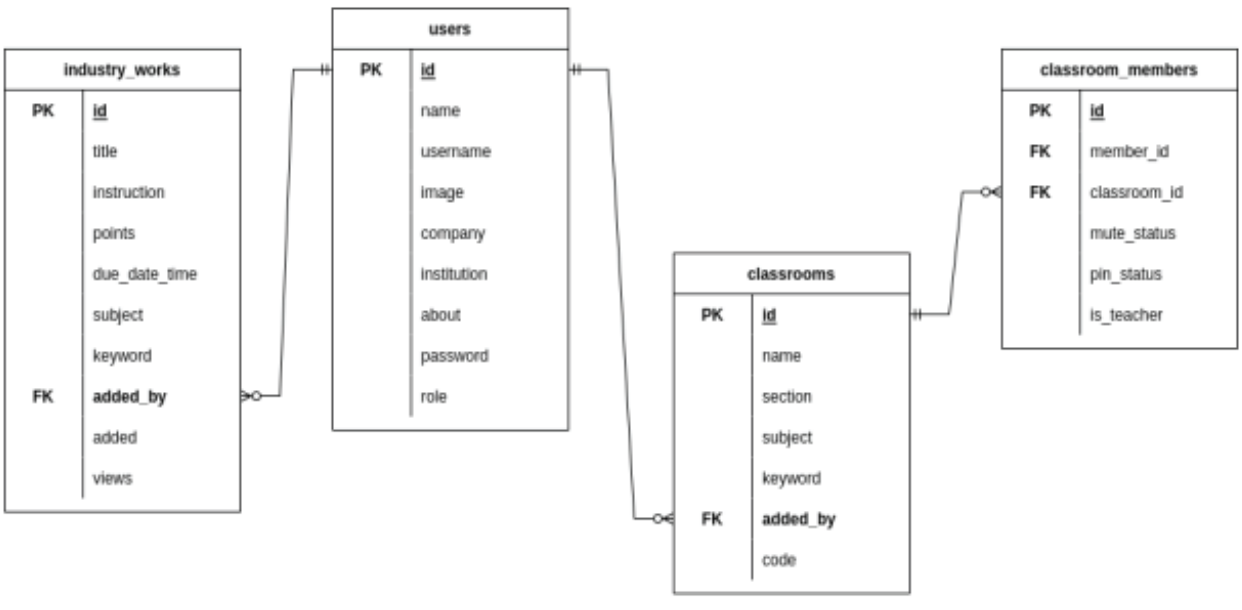

Figure 4.4: Entity Relationship Diagram 1

Figure 4.5 begins by illustrating the connection between the classroom and the classroom posts. The classrooms entity has one to many relationships with topics and classroom_posts entity, meaning that a classroom may have zero or more topics and classroom posts. A classroom post could be a material, general_post, or assignment because the classroom entity has a one-to-one relationship with each of these entities. Materials, general_posts, and assignments have a one-to-many relationship with their corresponding attachment entities, which means that these entities may have multiple attachments. Additionally, the assignments entity has a one-to-many relationship with the entities assignment_submissions and assignment_comments. Assignment_submissions have a one-to-one relationship with assignment submission_grades, and a one-to-many relationship with assignment_submission_attachments. It implies that a submission can have more than one attachment but only one grade.

Figure 4.6 focuses on the subject industry work. As was previously mentioned, a user with the user role "industry" may have several industry tasks assigned for teachers to add to the classroom. Industry_works has a one-to-many relationship with industry_work_attachments because industry work may have multiple attachments. Additionally, the industry_works entity has a one-to-many relationship with the classroom_industry_work entity, allowing for the addition of an industry work in multiple classrooms. It's important to note that classroom industry work stores iw_id and classroom_id as a foreign key, enabling us to determine which industry work has been added to which classroom. Entity classroom_industry_work has a one-to-many relationship with industry_work_submissions, so an industry work may have multiple submissions.

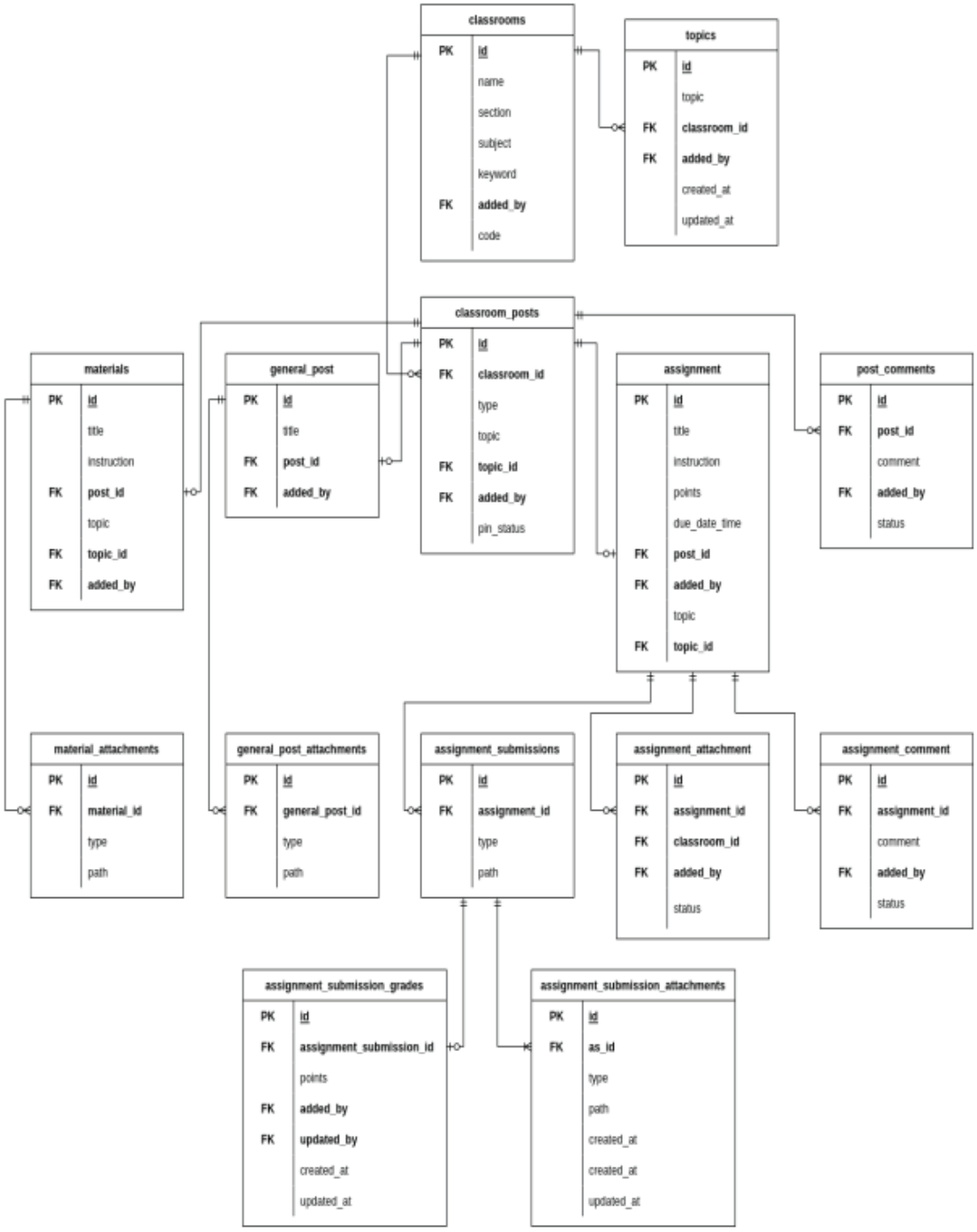

Figure 4.5: Entity Relationship Diagram 2

There may be multiple attachments but only one grade in a submission, so industry_work_submissions has a one-to-many relationship with i_w_submission_attachments and a one to one relationship with i_w_submission_grades.

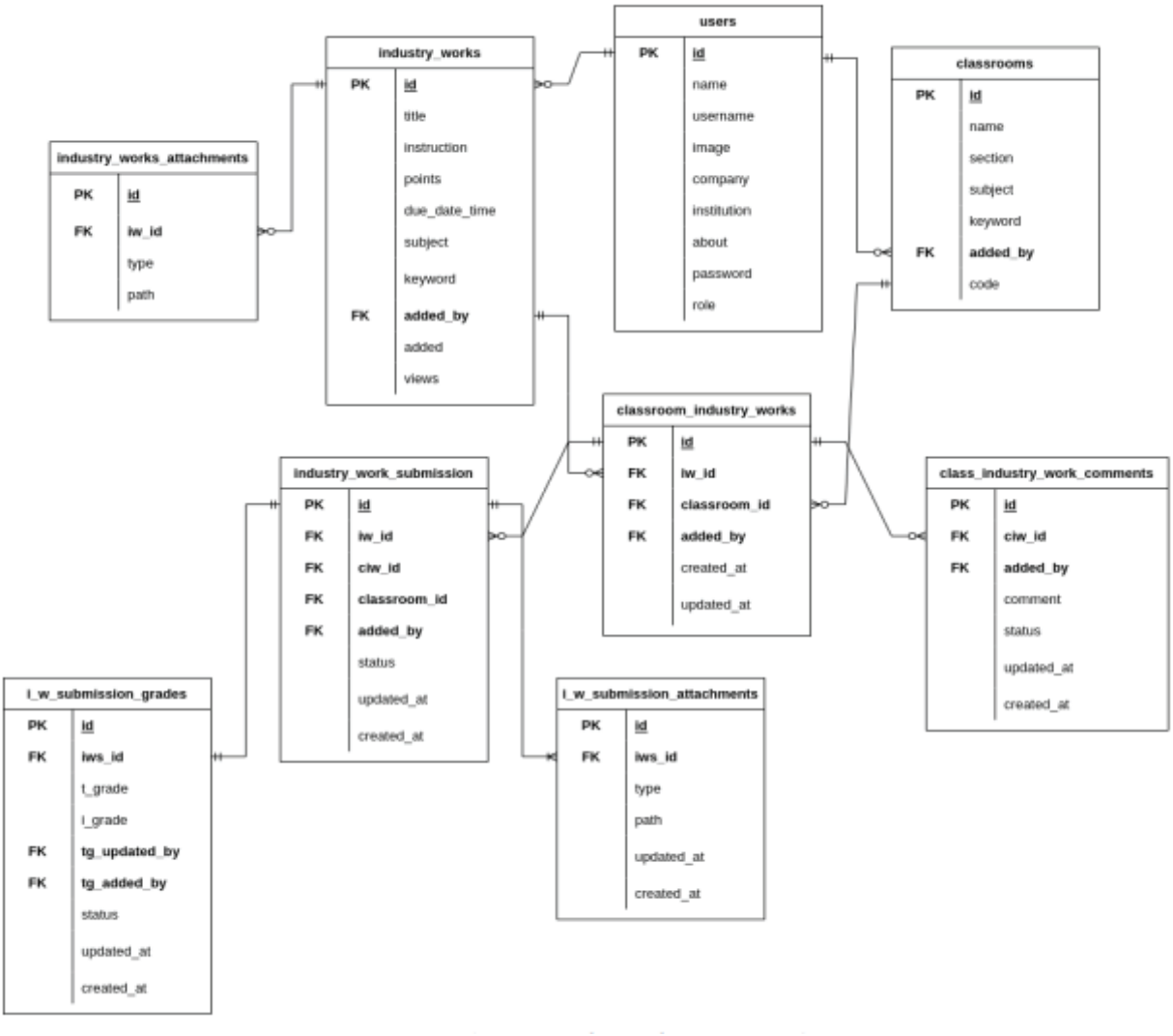

Figure 4.6: Entity Relationship Diagram 3

4.3 Use Case Diagram

A use case explains how a user applies a system to reach a certain goal. It makes the details of how people interact with a system easier to understand**.**

The Figure 4.7 express that, the system is used by the teacher, industry, and student as actors in academic activity. **Create classroom, post assignments, assigns grades,** and **adds jobs** are some top-level use cases for teacher. The teacher is able to create a new classroom by using the **create classroom** use case. Using the **add post** use case, a teacher can include an industry task into the classroom that has been added by the industry. Being able to **enter a classroom** and **submit assignments** and **industry jobs** would be another major use case for a teacher.

The student actor can join a classroom and submits an assignment using the **join classroom** and **submit assignment** case, just like the teacher can. A student may also want to view and submit industry jobs, which he can do by using the **view jobs** and **submit jobs** use case.

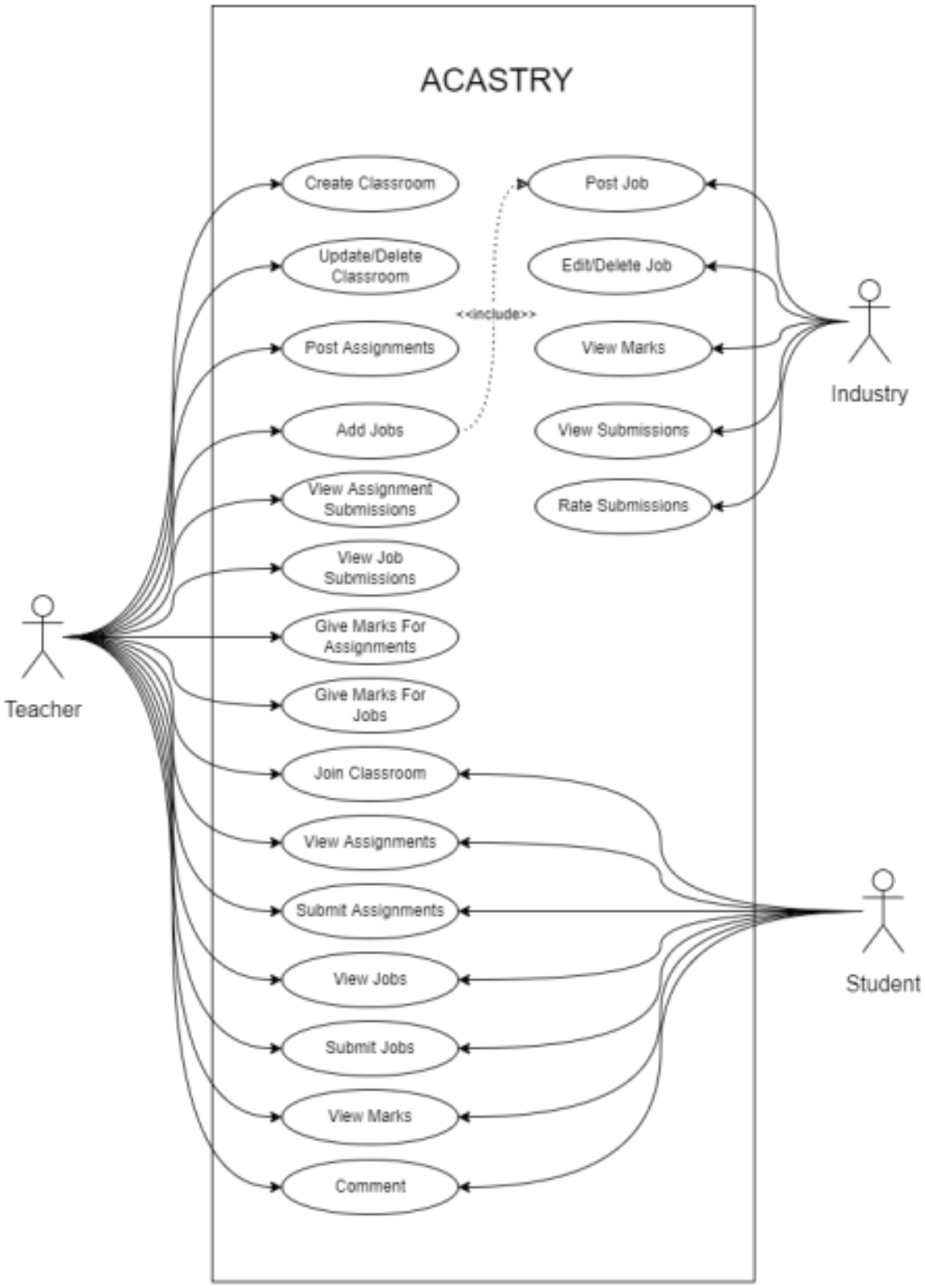

Figure 4.7: Use Case Diagram

Other major use cases, post job and rate job submission are specific to industry actor. The post jobs use case includes add jobs because an industry must post a job before it can be added in a classroom by a teacher.

4.4 System Flow Chart

System flowcharts are a means to show how decisions are made to regulate events and how data flows in a system. The system begins when the users enter the website. Following user verification, the system will lead registered users to the login stage, while unregistered users are directed to the registration stage where they can enter their credentials or inputs. The system will then determine the user's role; if the user is a teacher, it will then permit them to create and join classrooms. When a teacher joins a class, they have the option in the following processes to view classroom posts or submit assignments and industry tasks. If a teacher creates a classroom, they can include industrial work or classroom posts in the classroom and they can assess those submissions afterwards.

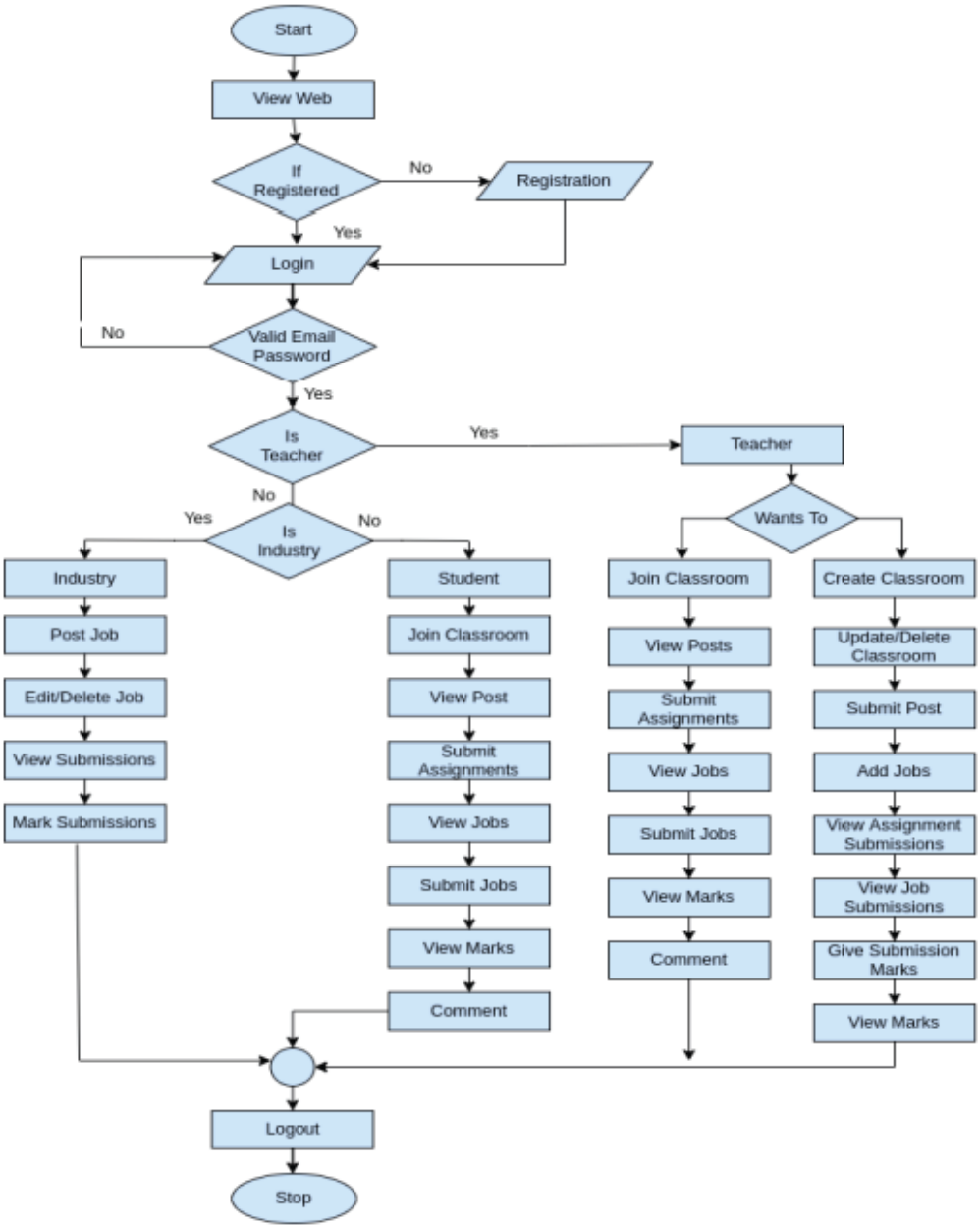

Figure 4.8: System Flow Chart

On the other hand, a student can only enter a classroom. Therefore, a user with the "student" role who logs into the system can only join an existing classroom; he or she cannot create a new classroom. A student may also access the classroom posts and submit assignments and industry works. The following stage allows the student to view their grade if the assignment has been graded.

A user with the role "industry" will be redirected to the industry dashboard where industry can offer jobs/tasks for students. The following step allows an industry to review submitted work and provide grades for it. The system terminates when a user clicks the logout button to leave the system.

The insights gained from this chapter may be of assistance to comprehend the system at its core and influence the further attempts of establishing a fully AI-aided platform to support the collaboration between academia and the industry.

CHAPTER 5
Implementation Details

This chapter provides a brief glance of the user interfaces that one would come across while navigating through our proposed system. Basically, the intention is to familiarise the user with all the functions that a particular user has access to and at the same time reflect the user-friendly navigation process.

5.1 Web View of the System

Login Page and Register Page:

Wherever the application is launched, a login page similar to Figure 5.1 appears. To create a new account, an unregistered user can go to the register page. To create a new account, a user must enter their name, username, email, and other necessary information.

Figure 5.1: Login Page (left) and Registration Page (right)

Teacher, Student, and Industry Dashboard:

After successfully logging in, the user will be taken to the dashboard. A dashboard similar to Figure 5.2 will appear on the screen when a

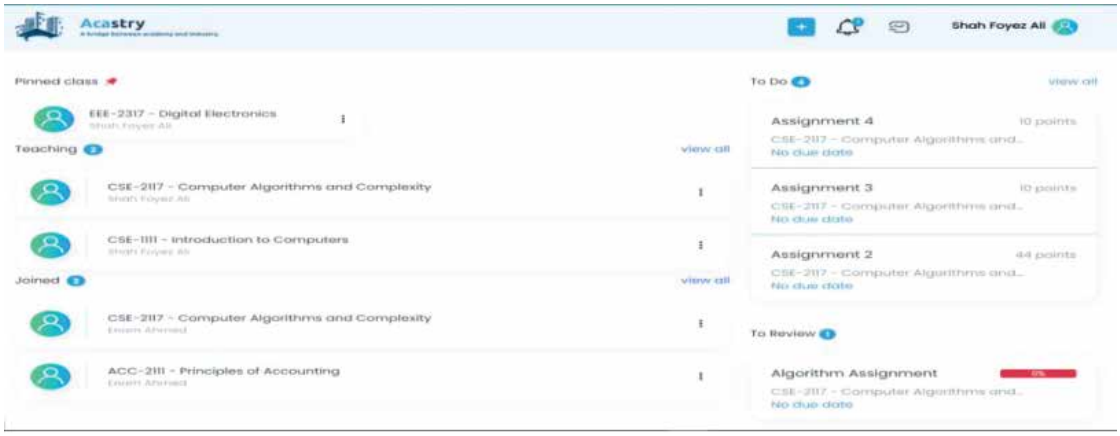

Figure 5.2: Teacher and Student Dashboard

user with the roles of "student" or "teacher" logs into the system. A dashboard similar to Figure 5.3 will appear if an industry user logs into the system.

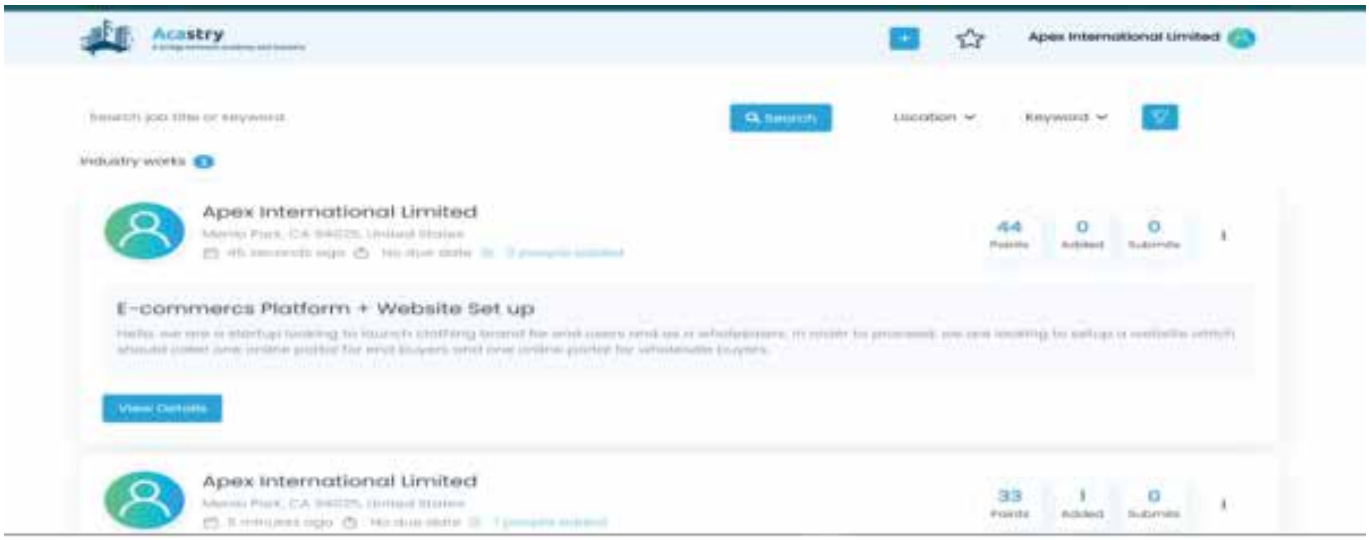

Figure 5.3: Industry Dashboard

The dashboard offers a variety of options and data for the teacher and student to view. They can view the classroom they have joined or any assignments they might have. However, teachers can create new classrooms; students can only enter existing ones.

Create or Join a Classroom:

A student user can only join a classroom; a teacher on the other hand can create and join classrooms in this system. A pop-up window where the user must enter the classroom title, subject, and other information appears when the teacher hits the "create classroom" button like as Figure 5.4. A classroom code is automatically generated when a teacher creates a class. To join the class, a student must enter the classroom code in the join classroom popup depicted in Figure 5.5.

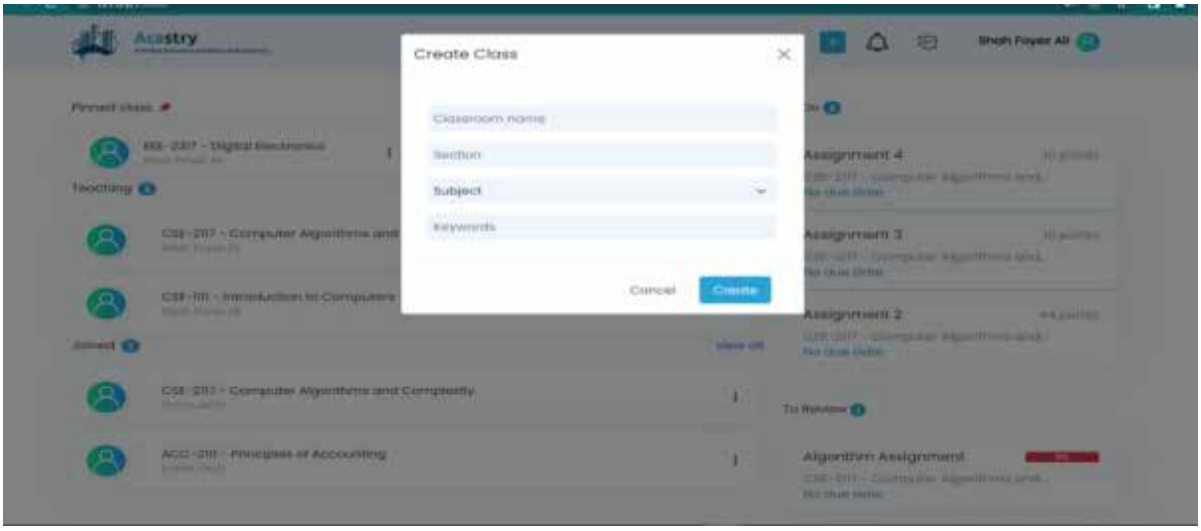

Figure 5.4: Create Classroom Popup

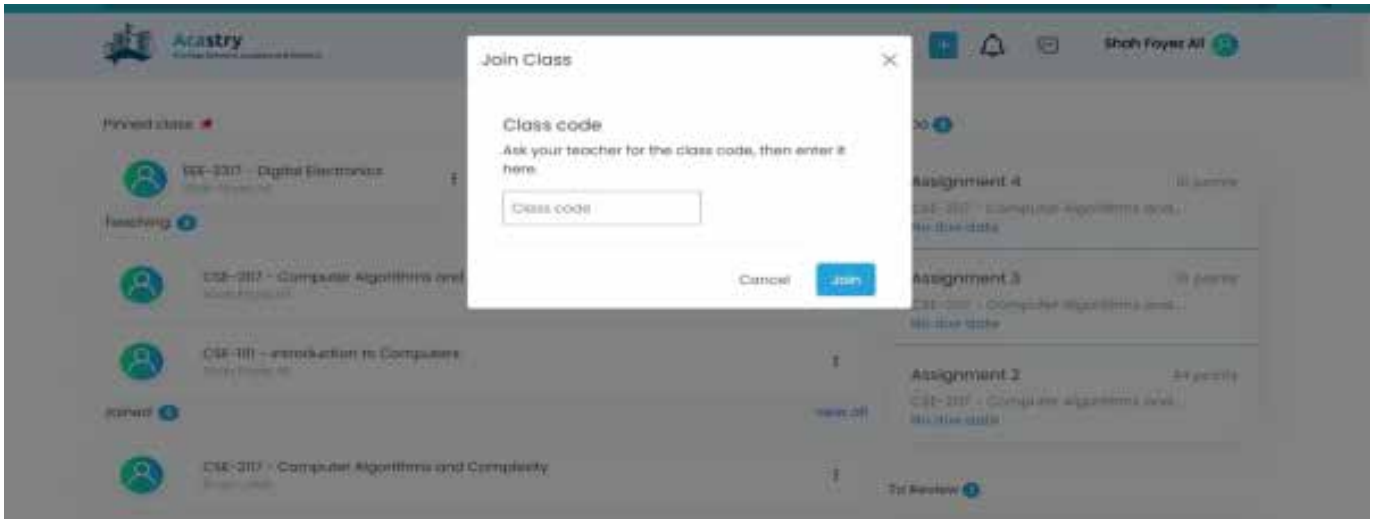

Figure 5.5: Join Classroom Popup

Classroom Stream:

Students and teachers can both access the classroom stream. Every time a teacher posts something or adds new materials and assignments, it will appear on the classroom stream. Actions like edit or delete stream content are restricted, only the teacher has the access to modify stream content.

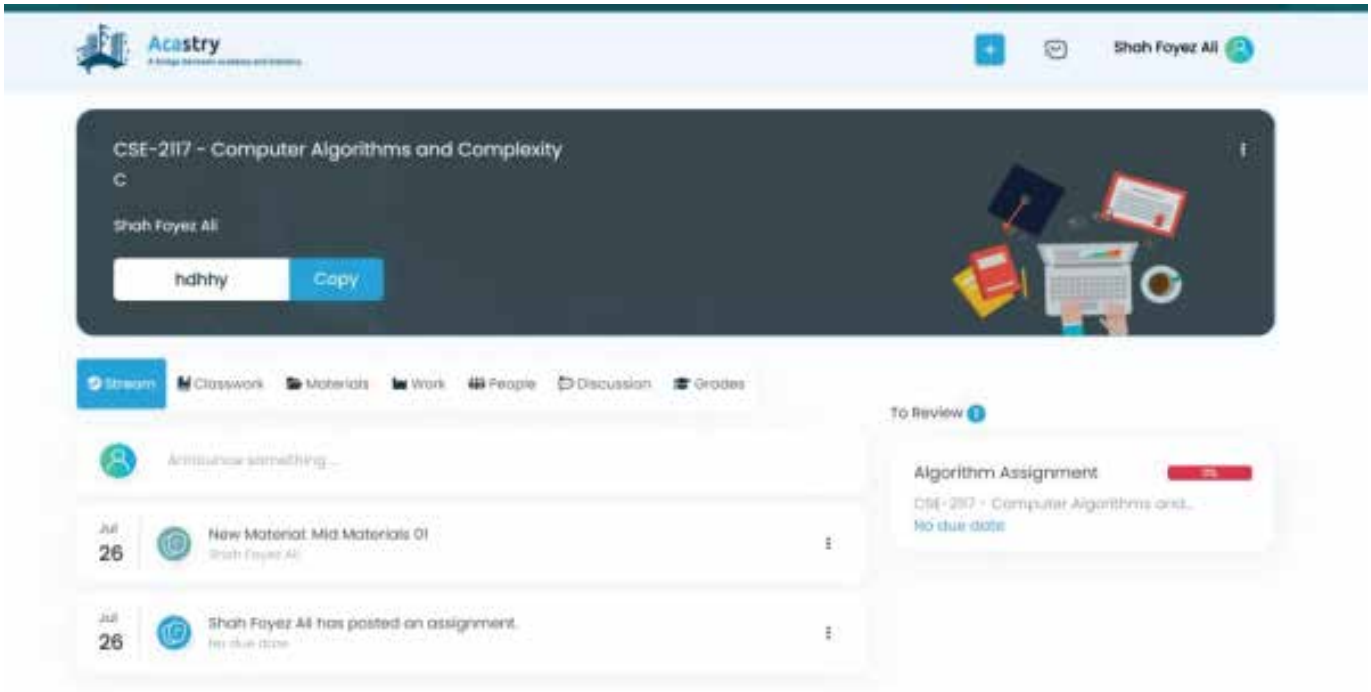

Figure 5.6: Teacher Stream

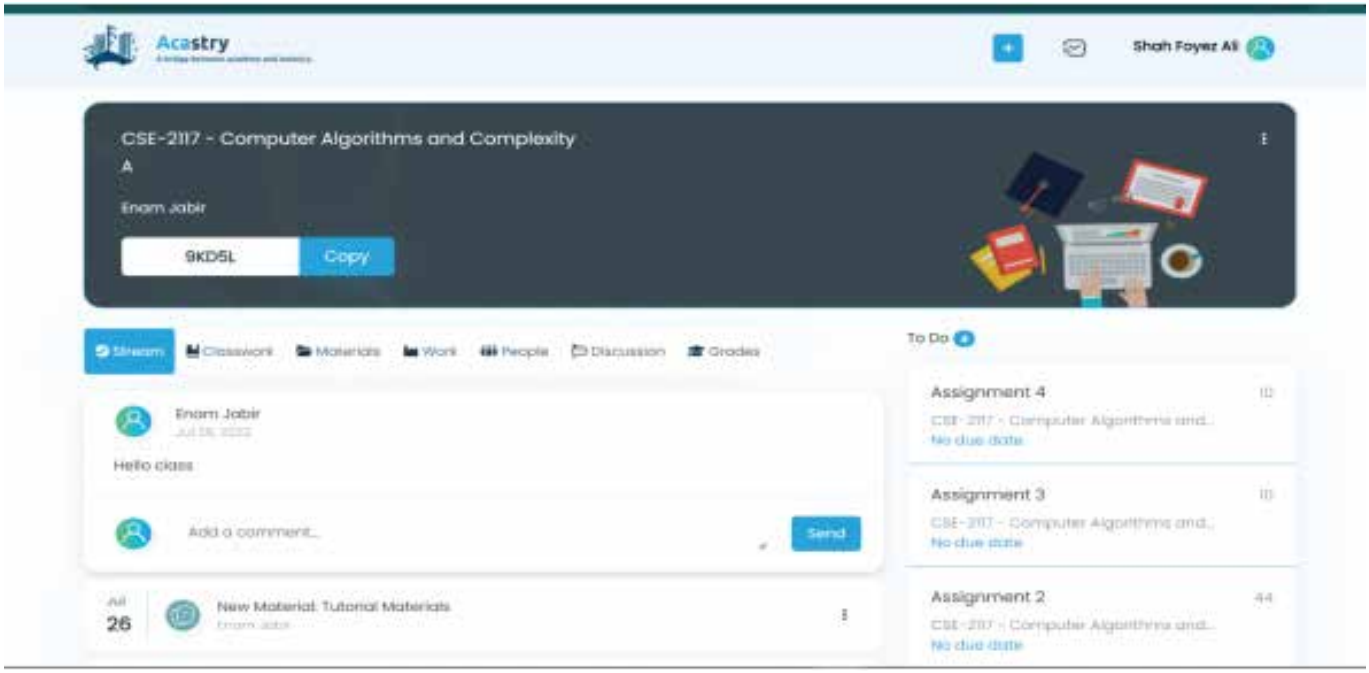

Figure 5.7: Student Stream

Class work and Materials:

The classroom's assignments and materials are shown on the class work and materials page. The classroom teacher is only permitted to add new assignments and materials, but both the teacher and students can view those classroom content. As seen in Figure 5.9 and Figure 5.10, a popup will appear when the teacher clicks the create button, allowing them to enter the information they need for the assignment or material.

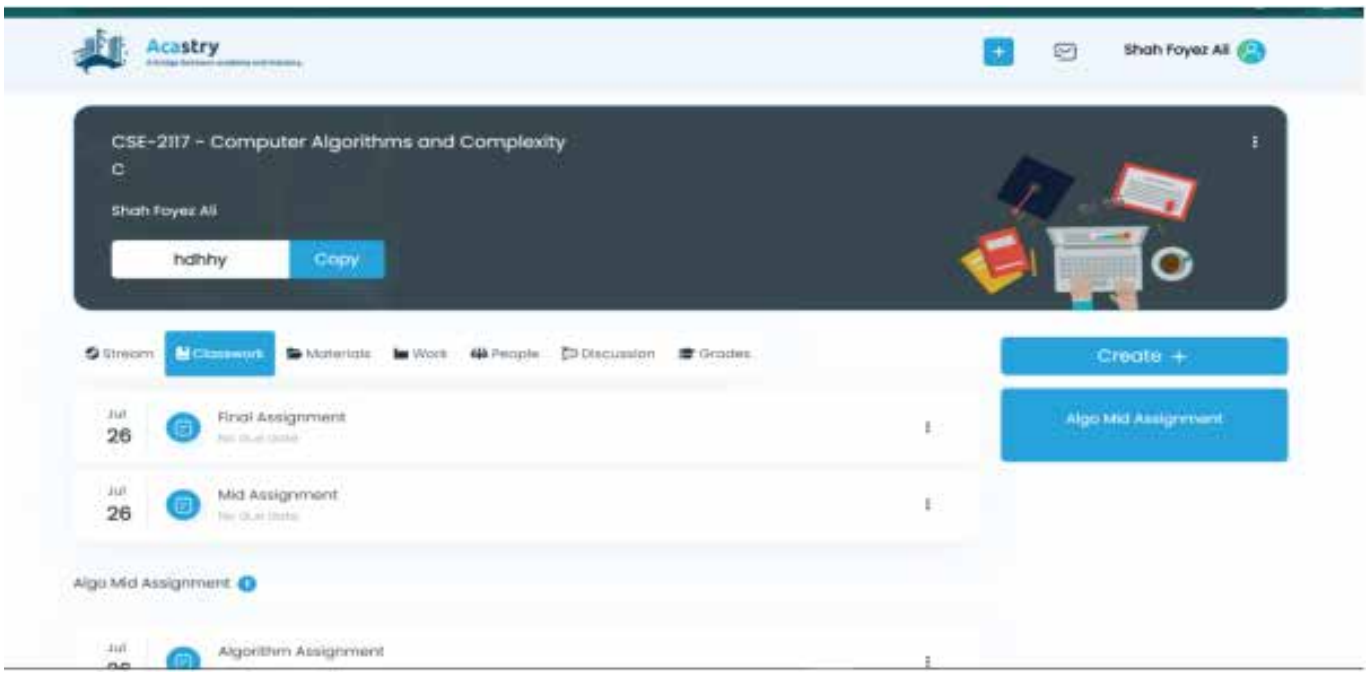

Figure 5.8: Class work Page

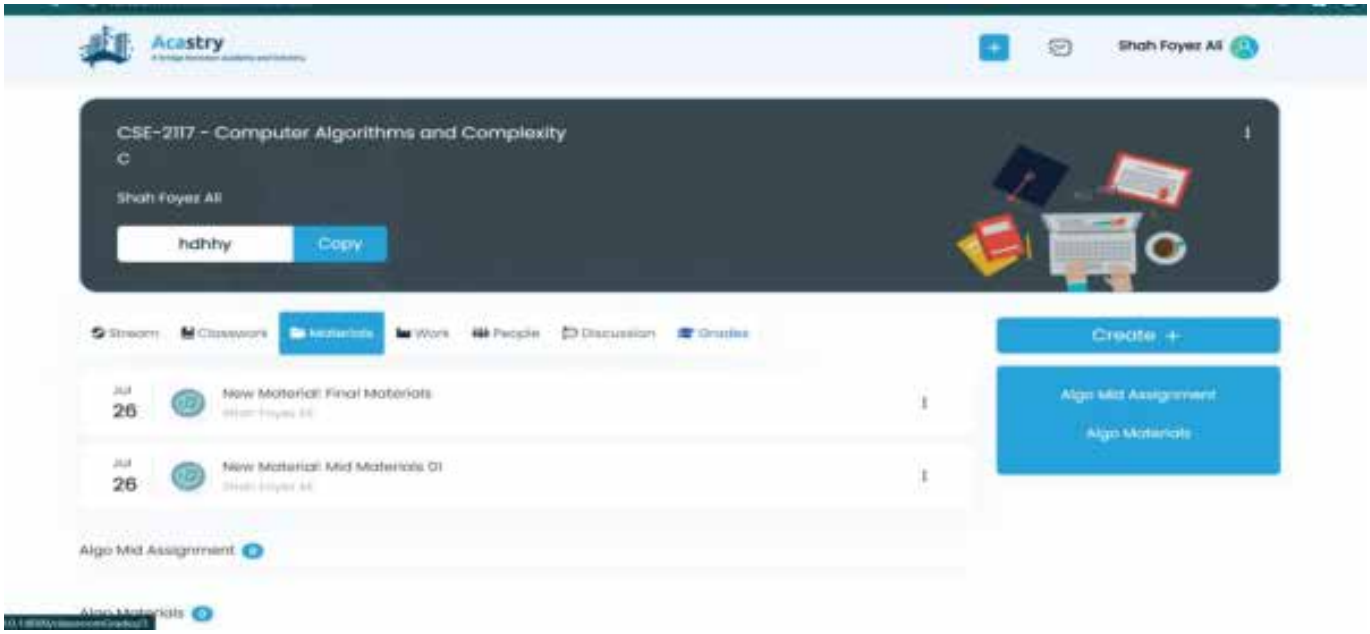

Figure 5.11: Class Material Page

Figure 5.9: Add Material Popup

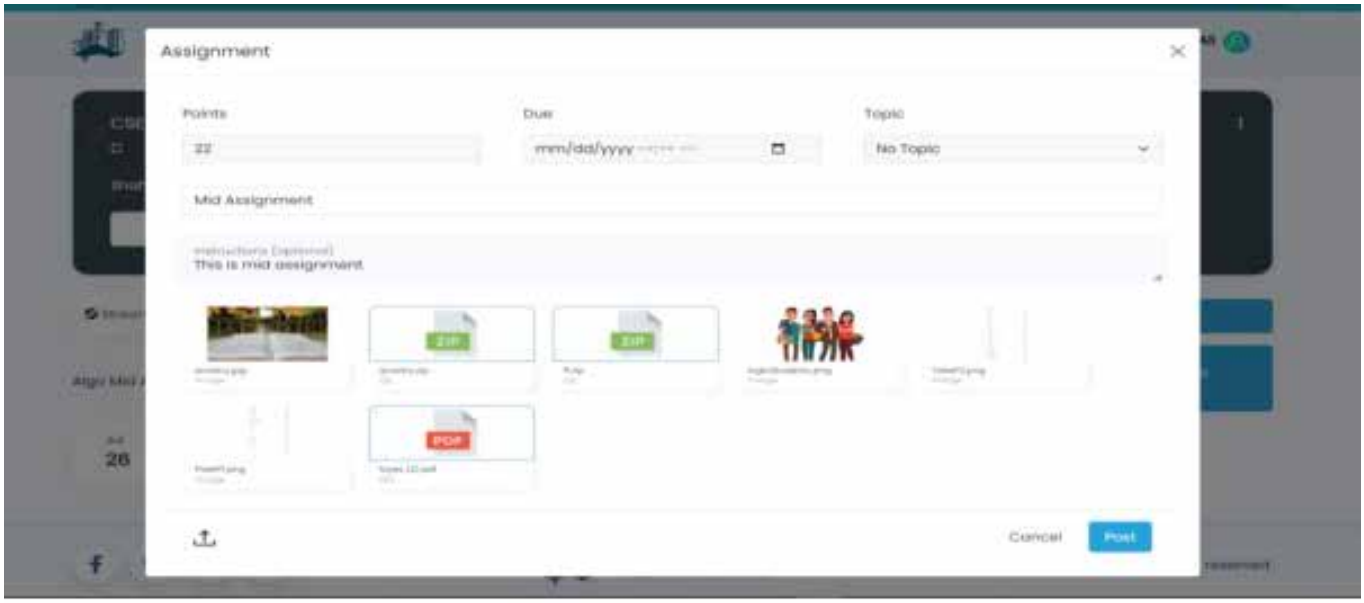

Figure 5.10: Add Assignment Popup

Assignment Submission and Evaluation:

Students can see the instructions and upload their assignments on the assignment submit page (Figure 5.12). On the assignment submissions page, a teacher can view all of the assignment submissions (Figure 5.13) and grade each one. Once their submission has been graded, a student can review their grade on the assignment submit page.

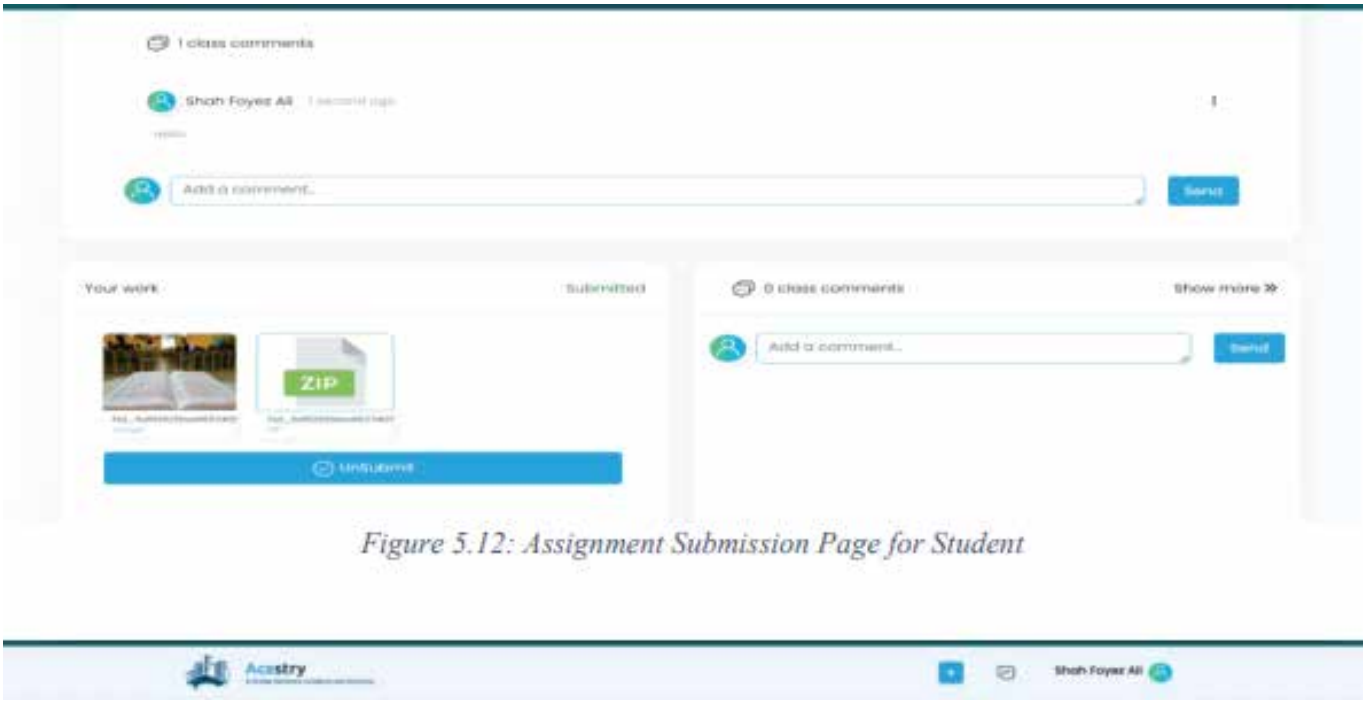

Figure 5.12: Assignment Submission Page for Student

Figure 5.12: Assignment Submission Page for Student

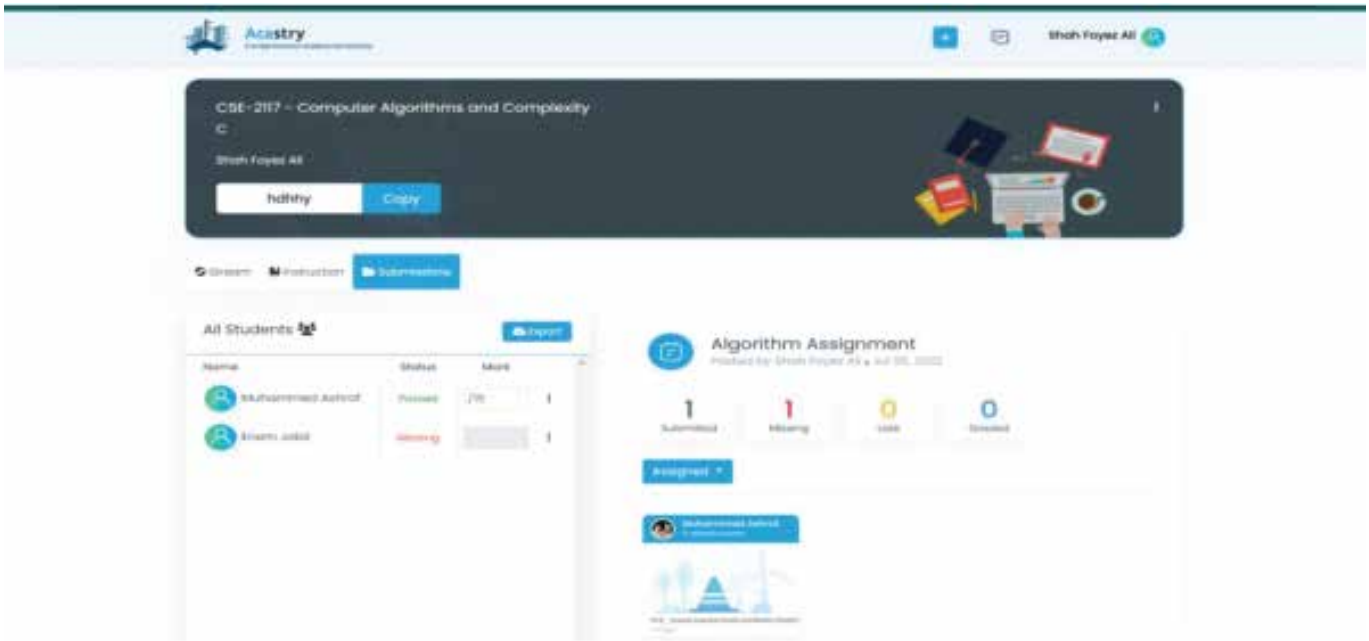

Figure 5.13: Assignment Grade Page for Teacher

Industry Work:

Figure 5.14 shows the pop-up window that allows an industry user to post an industry work. An industry user must provide the required input to post an industry work. Once the work is posted, it will show up on the industry dashboard, as shown in Figure 5.15.

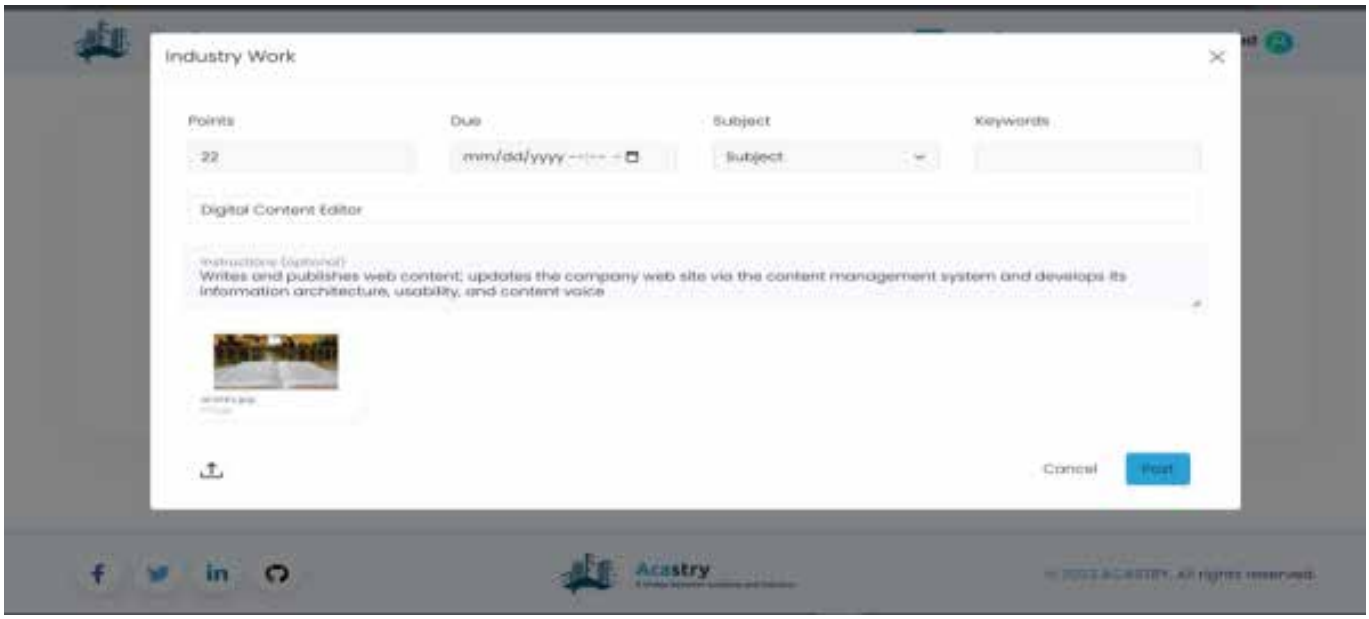

Figure 5.14: Add Industry Work Popup

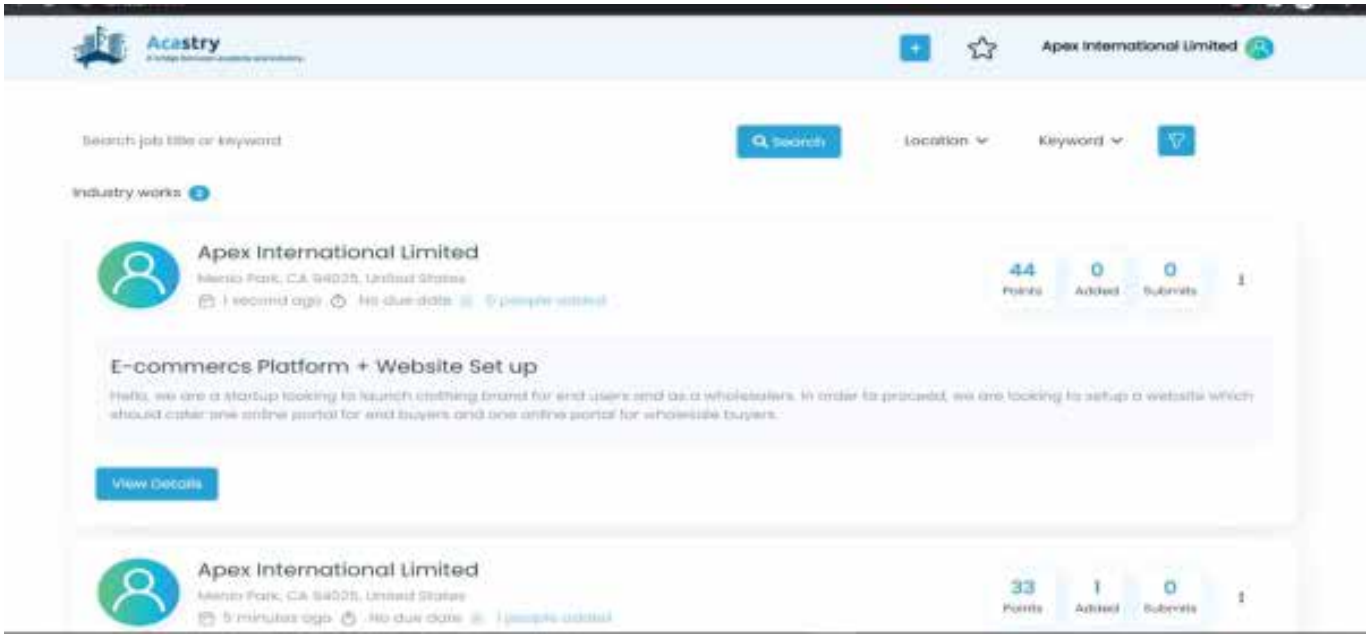

Figure 5.15: Industry Dashboard

Added Work and Related Work:

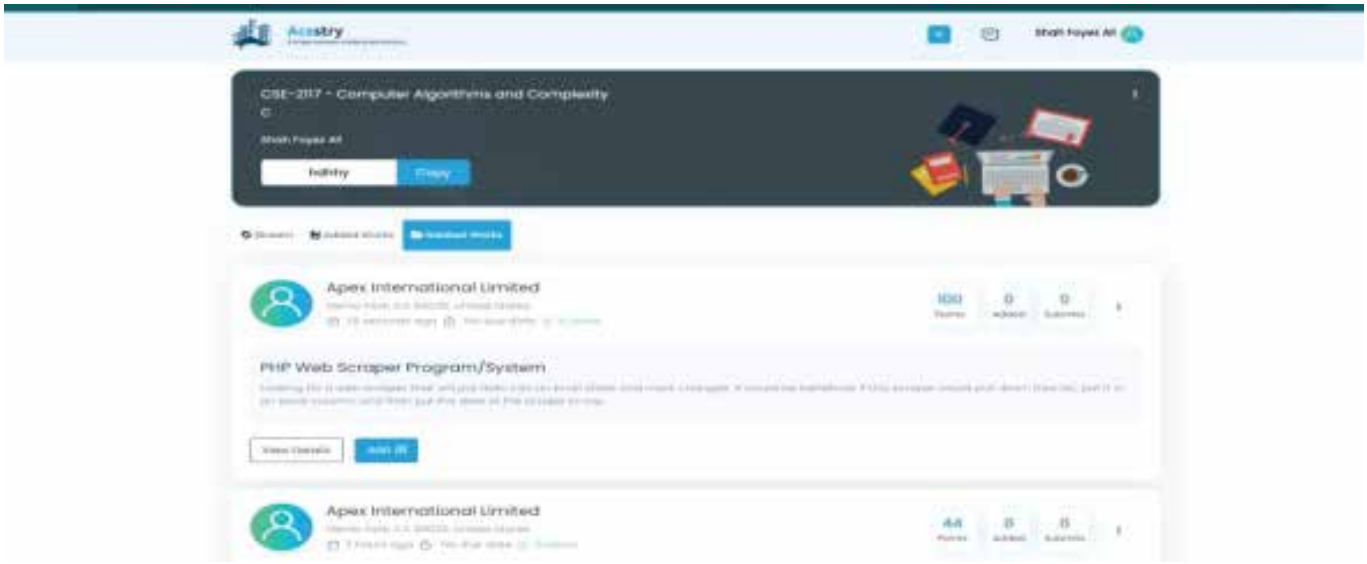

Figure 5.16: Related Work Page

A teacher of a classroom might view his classroom-related industry works on the related work (Figure 5.16) page. The teacher can review the work's details and add it to his class, allowing his students to submit the industry task. A task will

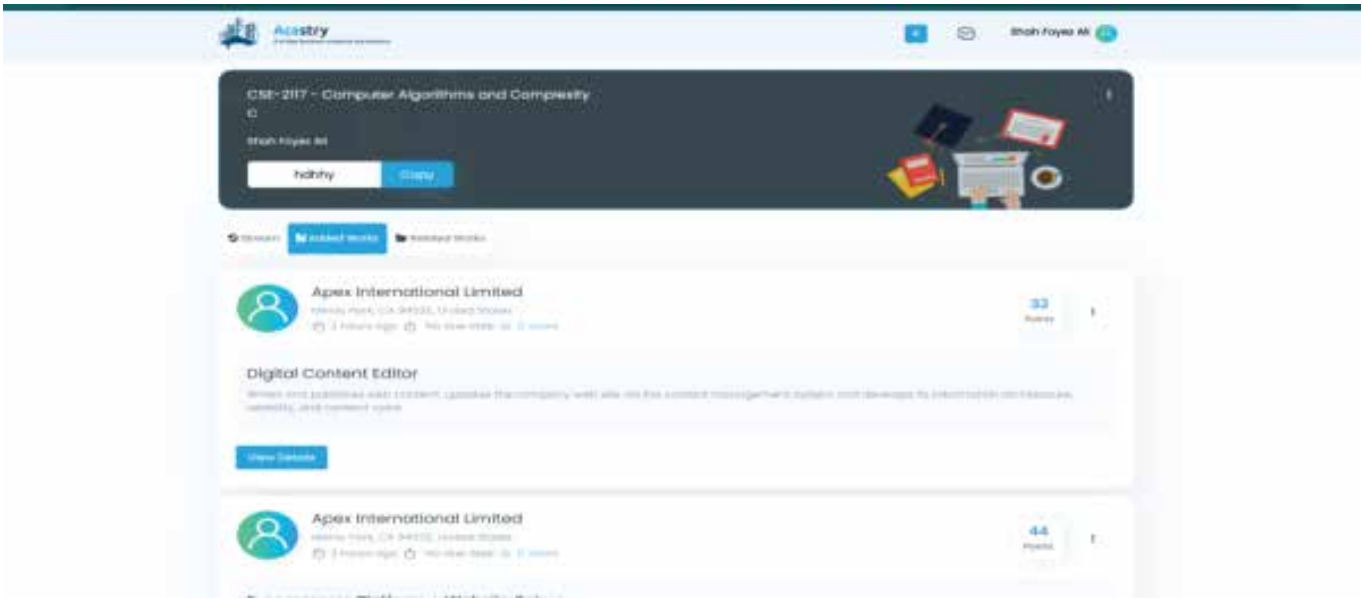

Figure 5.17: Added Work Page

appear in the added work page (Figure 5.17) and accessible for submission once it has been added to the classroom.

Industry Work Submission and Scoring:

The submissions may be graded by both the teacher and an industry user; however, a teacher may only grade student submissions from his or her own classroom. The grade that each party assigned to a task can be seen by both the teacher and industry practitioner.

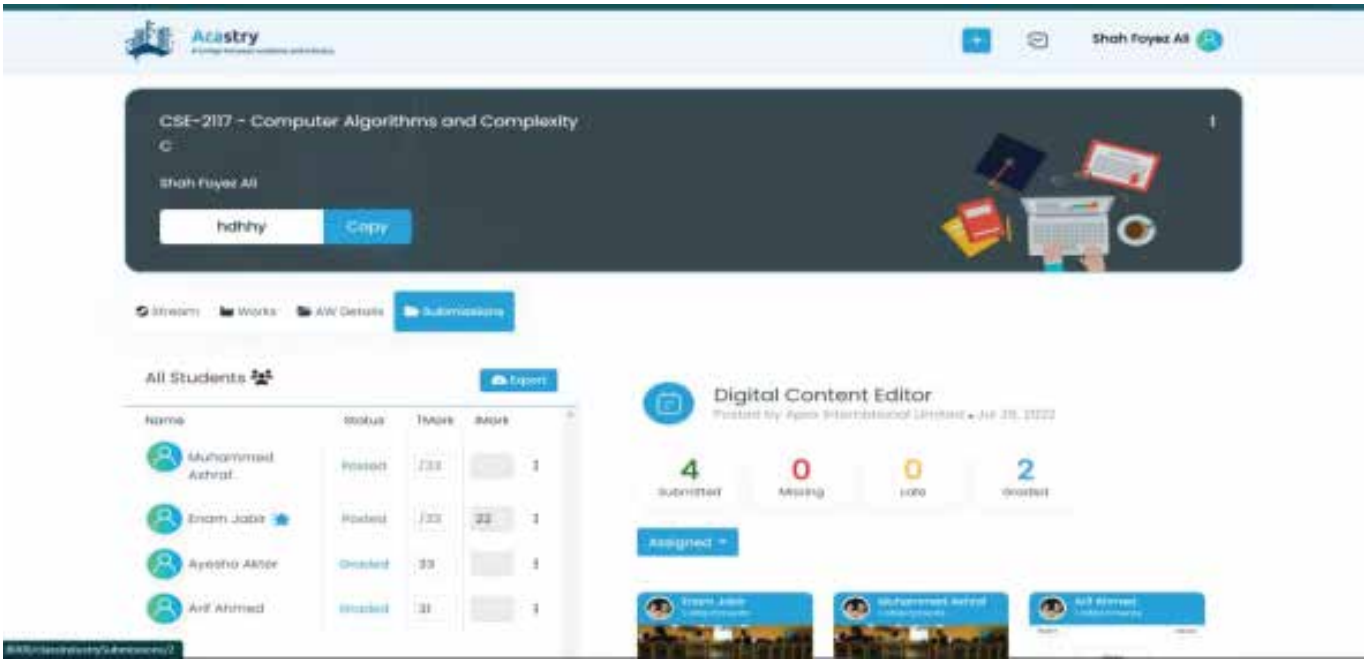

Figure 5.18: Teacher's View of Industry Work Submissions

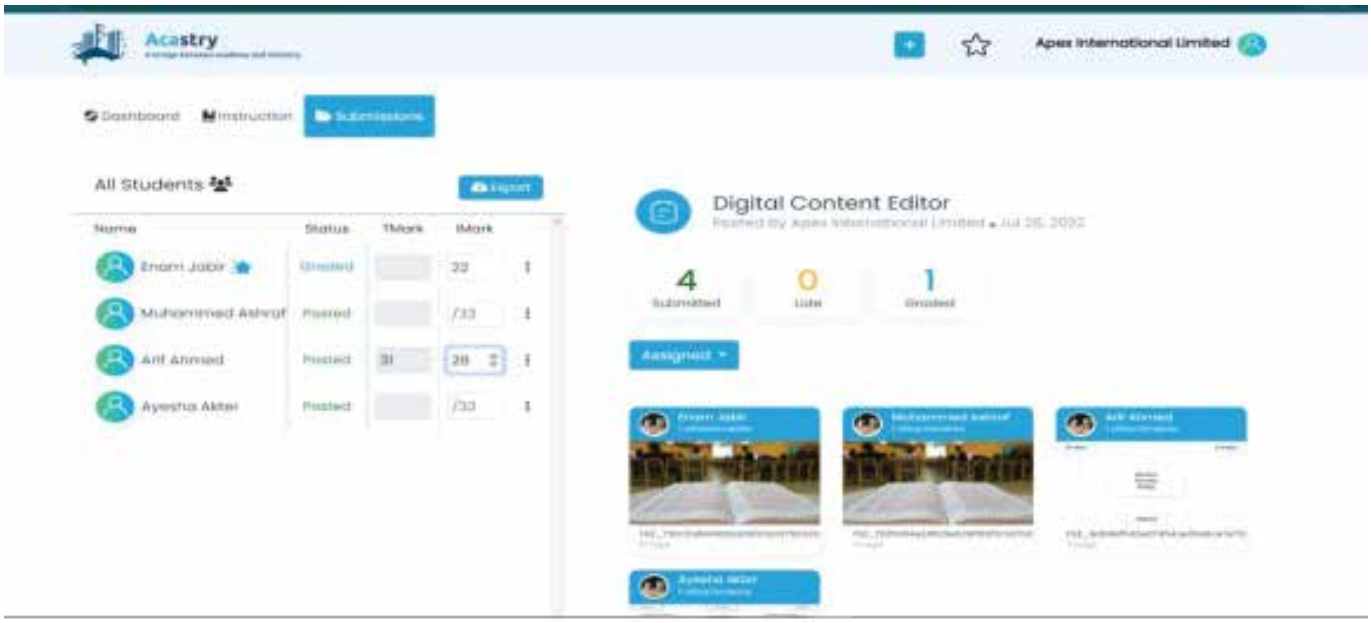

Figure 5.19: Industry Practitioner's View of Industry Work Submissions

5.2 Modules Integration

The system: Acastry, enables academicians to make posts for the classroom that include materials and assignments. An industry user may also post an industry task using our system. To allow students to submit them, these industrial tasks can be added to the classroom. Our keyword extraction module will use the title and instruction of the assignments, materials, or industry works to extract keywords. Recommendation module will be useful when a teacher adds industry tasks into their classroom. Using the keyword extraction module, our system will read the classroom activity and suggest to the teacher any industry task that is relevant for the classroom.

We have developed a Python program called "PAKE" that extracts keywords from an input string. We have integrated PAKE (keyword extraction module) with our system. Each time a teacher uses the route to create assignments and materials, or an industry user uses the route to create industry tasks, our system will store the input data in the database. The main

system will then concatenate the title and instruction before passing them to the PAKE system for execution. PAKE performs the necessary calculations, extracts the keywords, and assigns a key score to each keyword. The top 10 keywords with the lowest key scores will be stored in the relevant data table.

Now we have a data table that carries keywords extracted from classroom activities and another data table that holds keywords extracted from industry tasks. When a teacher accesses the "related work" page, the system performs a check using the cosine similarity metric, comparing the keywords in the classroom_keywords data table with keywords from industry tasks in the iw_keywords data table. If the system detects a minimum required match in the comparison, it will suggest the tasks for the classroom. Thus, a teacher will only see tasks from the industry that are relevant to the classroom

In a nutshell, the demonstration of all the pages that a user comes across, regardless of whether the user is an academic or an industrial practitioner, is relevantly easy to navigate, and the design also intends to make it interesting for students to relate to, making it easier for all users to have a better experience.

Chapter 6
Results and Discussion

This chapter focuses primarily on the results obtained after closely adhering to the methodology described for the automatic keyword extraction process. It includes the datasets used, performance metrics for evaluation, a comparison analysis of the proposed system with other established baselines, and a comprehensive assessment of the results obtained.

6.1 Datasets

We put our suggested keyword extraction system to the test on two different document collections to gauge its effectiveness. The collections include texts with varying text lengths, and we have made use of well-known databases like Inspec [24] and SemEval2010 [43].

From the ACM Digital Library, 244 complete scientific papers obtained for SemEval2010 [43] for evaluating keyword extraction and they are previously used one of the most well-known datasets. There each research paper consists of a set of keywords selected by the individual author along with a sizable group of keywords chosen by professional editors; the input text might not necessarily contain keywords from both given sets. This dataset ranges in length 6 - 8 pages spanning over four different fields of computer science research which are as follows:

• Social and behavioral sciences (economics)

• Distributed Artificial Intelligence (multiagent systems)

• Information search, retrieval, and distributed systems

Moreover, the specific version of the dataset used for evaluation refers to the works of [44], developed by their team to incorporate four refined standards for the overall processing of documents. It included manual extraction of the PDF files, processing the input into suitable columns,and choosing the most content-rich documents in the process.

The Inspec dataset [28] includes 2000 abstracts of computer science journal articles gathered between 1998 and 2002. It focuses on the following disciplines: Computers and Control; and Information Technology. Each article has its respective title along with the keywords obtained from Inspec's database, and a designated indexer assigned two sets of keywords to each abstract: a collection of controlled terms, or words strictly adhering to the Inspec thesaurus, and a group of uncontrolled terms that may contain any relevant word. At the same time, it may not be necessary that the abstracts would consist of terms from both sets of controlled and uncontrolled words.

6.2 Performance Metrics

The most important question that arises is how good our model is at supporting the system. So, evaluating the proposed model is essential to delineate how effectively our system can extract the keywords. This requires specific performance metrics to demonstrate the evaluation. Beforehand, a sneak peek of the underlying terminologies can demonstrate the explanation more understandably.

True Positives (TP): These are the correctly predicted positive values stating that the value of the actual class and the value of the predicted class are both true. In this case, the extracted keywords and the golden keywords are the same

True Negatives (TN): These are the correctly predicted negative values inferring that the value of the actual class and value of the predicted class both are false. This means the extracted keywords and the golden keywords do not match

False positives (FP) and **False Negatives (FN)** happens when the actual class contradicts the predicted class.

Precision - Precision is the ratio of correctly predicted positive observations to the total predicted positive observations. High precision relates to the low false positive rate. In other words, it measures how many positive predictions are correct (true positives).

$$\text{Precision} = \frac{TP}{TP + FP}$$

Recall (Sensitivity) - Recall is the ratio of correctly predicted positive observations to all observations in the actual class - True.

$$\text{Recall} = \frac{TP}{TP + FN}$$

F1 score - F1-Score is a measure combining precision and recall. It is the weighted average of Precision and Recall. Therefore, this score takes both false positives and false negatives into account. F1 is usually more helpful than accuracy when there is an uneven class distribution. F1-score becomes high only when both precision and recall are high.

$$\text{F1 Score} = \frac{2(\text{Precision} \times \text{Recall})}{(\text{Precison} + \text{Recall})}$$

6.3 Comparison Analysis

The distribution of the top-N candidate keywords is conventionally based on precisely matching the golden keywords with the extracted ones. In some contexts of literature review, keywords that are close enough semantically are also considered as part of the standard keywords of the given corpus. Since our task involves the Inspec and the SemEval2010 dataset, we pursue the traditional approach of matching the keywords exactly. We determine the precision, recall, and F-score by comparing the provided presumed key phrases in the dataset. The performance of each system is evaluated by examining the output as the top-5 and top-10 candidate keywords. A summary of the results is shown in Table 6.1, collates PAKE's usefulness with that of the baselines on each dataset in percentage terms.

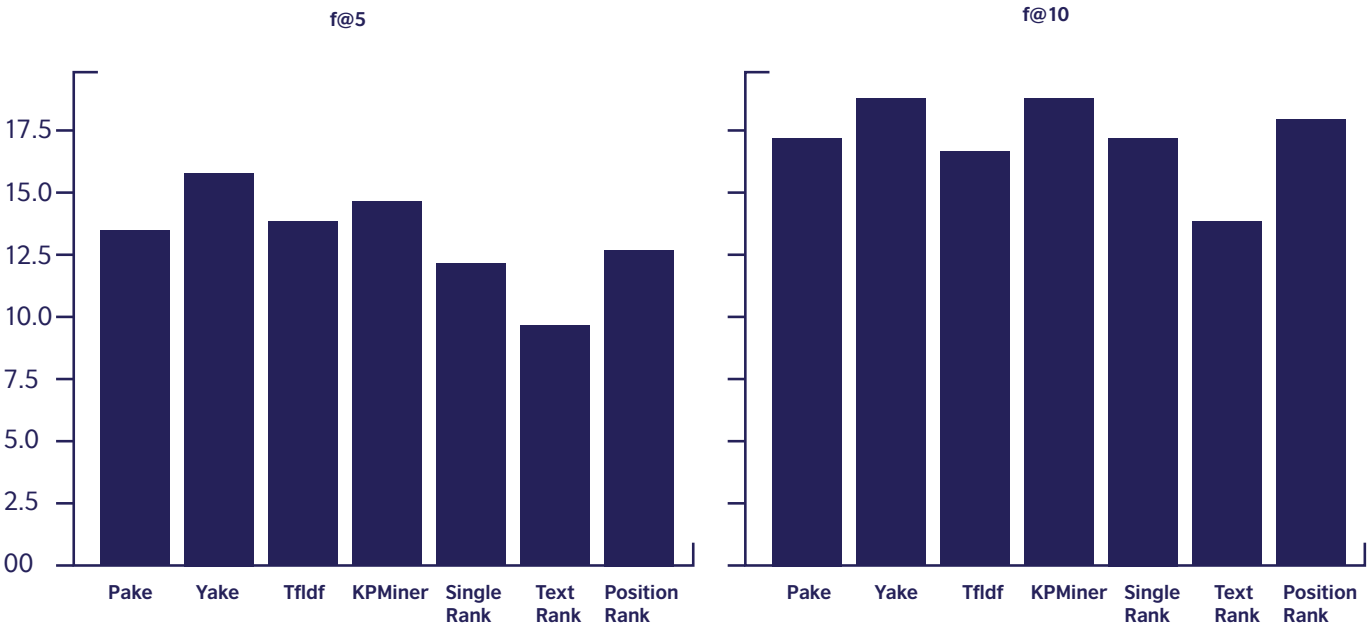

Figure 6.1: F-scores on the SemEval2010 Dataset

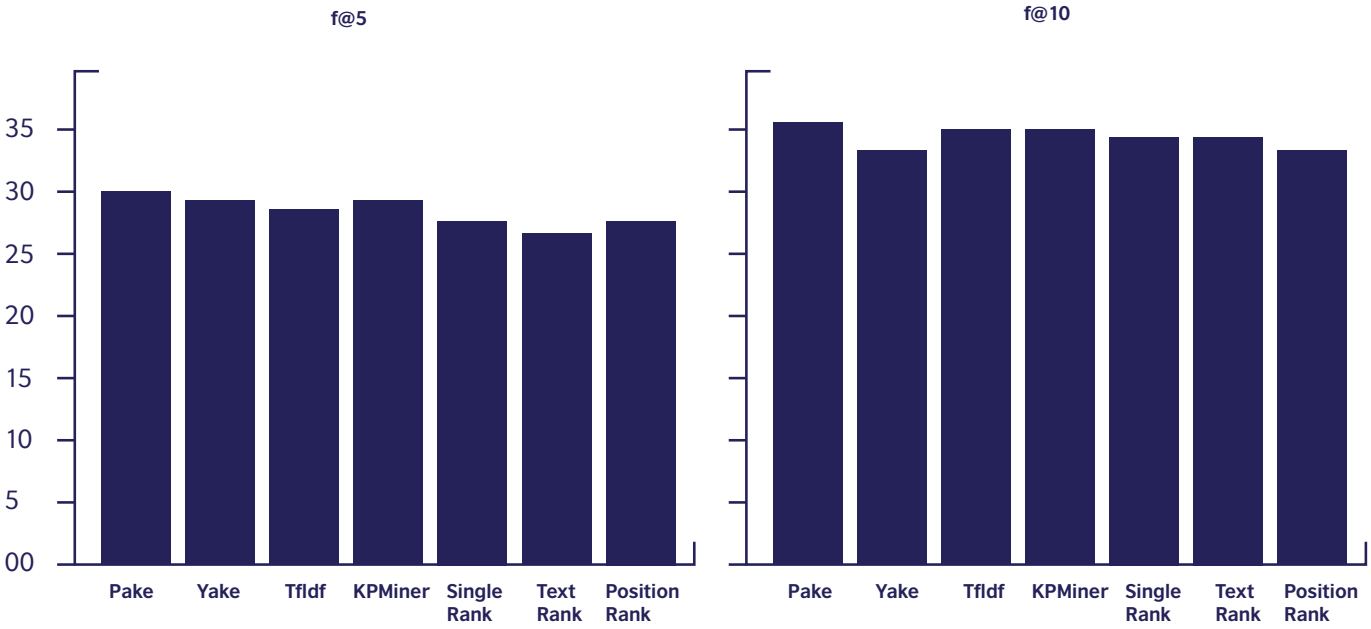

Figure 6.2: F-scores on the Inspec Dataset

It is evident to corroborate that the overall efficacy of our proposed keyword extraction is substandard for the following reasons: (1) The statistical methods do not filter out the corpus on the basis of lexical terms as compared to PAKE and other graph-based models, thereby resulting in biased results Figure 6.1. (2) The certain constraints faced in any NLP task along with the lack of a customised dataset.

Table 6.1: Comparison of F-scores on the Datasets

Dataset	Total Docs	PAKE		YAKE		TF-IDF		KPminer		SingleRank		TextRank		PositionRank	
		F@5	F@10	F@5	F@10	F@5	F@10	F@5	F@10	F@5	F@10	F@5	F@10	F@5	F@10
Inspec	244	30.13	35.75	29.54	32.94	28.75	34.94	28.93	35.07	27.80	34.18	27.03	34.20	28.04	33.20
SemEval 2010	2000	12.93	16.63	15.73	17.97	13.20	16.08	14.48	17.90	11.55	16.29	9.32	13.42	12.54	17.32

We conclude this analysis by suggesting that f-scores show that overall, the results increase and are higher than the graph-based models that employ POS taggers. Furthermore, a slight contrast arises when compared to PositionRank, which marginally provides favorable results. Finally, regarding the statistical methods, the system's output of f-scores exhibits considerably lower scores than statistical baselines, especially YAKE! and KPminer for the SemEval2010 dataset. However, for F@10, the improvement in the scores indicates better precision and recall. Remarkably, the performance boosts when using the Inspec Dataset and PAKE subsequently has higher F-scores in contrast to the mentioned baselines. Therefore, we can infer that PAKE is a plausible substitute for state-of-the-art graph-based algorithms to extract keywords from a given corpus concerning the usage of POS taggers and can be significantly enhanced to match the performance concerning the statistical baselines.

6.4 Discussion

The substantial performance of our proposed system spans over the chosen number of candidate keywords and the varying categories of author-assigned and professional indexer assigned keywords. The evaluation demonstrates the relevant achievement of our proposed keyword extraction system compared to some of the existing graph-based alternatives and is conditionally at par with the statistical baselines.

We can assume that obtaining keywords automatically depends on the particular study's aims. Hence, it can be subjective from the perspective of a researcher. Undoubtedly, the system can continually be improved for effectiveness as well as better performance. However, irrespective of using POS tagging, there is still a need for an evaluation in terms of the semantic understanding of the candidate keywords. This will eventually lead to a better approach to generating keywords with improved results in evaluation metrics.

CHAPTER 7
Conclusion and Acknowledgment

At the end, Acastry is trying to solve a crucial problem of our society where the industry, academia, and graduating students will be benefitted in long run to ensure impactful collaboration and synchronous development. This report emphasizes on what the research intends to fulfil in terms of bridging the gap and the approach we take.

7.1 Conclusion

The proposed system attempts to mitigate the differences between academia and the industry with the intent of successful collaboration through our AI-influenced system. The web application is aided by a POS tagger Augmented Keyword Extraction system that eventually works to provide better recommendations to the respective user. The potential benefits are highlighted throughout the report, and we believe the system will benefit both students and industry practitioners. If we consider from the student's perspective, the system will help the student gain relevant industry experience by understanding the kind of projects available at the industry level. Thus, refining the profile with adequate skills will help the student enter the job market. From the other perspective, it is a better way to hire talented students or possibly outsource students to complete existing projects. This could potentially be a way for industries to include the component of corporate social responsibility and help minimize the growing gap.

7.2 Future Work

We have only been able to carry out hands-on work with ideas in a few areas and some preliminary work on evaluating the proposed modules for the system. Although we feel that substantial progress has been made towards the overall goal of defining a general framework for the design of evaluation methodologies, we are aware that much remains to be done. In particular, we intend to extend the keyword extraction system with plausible semantic meaning attached to it and build a supervised model that will help generate relevant skills specific to the skills required for the various jobs in the IT sector. It would require an elaborate and exhaustive dataset of keywords depicting the skills and thus is a future endeavour. Moreover, we intend to compare various models using different recommendation techniques to evaluate the overall performance to decide the best suitable approach for our existing system. This may involve using a content-based method aided with machine learning algorithms such as Naive Bayes etc. It may possibly extend to using collaborative filtering to connect the users at a community level or increase the overall efficiency by using hybrid recommenders for the system. In the future, the amalgamation of all the scenarios mentioned earlier can elevate the performance of our AI-based web app and eventually achieve the goal of reducing the gap to a large extent.

7.3 Acknowledgment

We are incredibly appreciative of the funding made available through the Researcher Connect Grant 2022 as well as the great opportunity provided by the British Council to support the research. We are grateful to our mentors Md Sabir Hossain, Assistant Professor, Department of CSE, CUET, and Md Zafrul Alam, UI Developer, Scotland, for their enthusiastic support and for offering helpful critiques and expertise that greatly aided the research.

7.4 Signature of the Chief Investigator

Name: **Md Saidur Rahman Kohinoor**

Signature: 

Date Signed: 15/08/2022

Note: If you have any questions, please do not hesitate to contact us at 01732477046 or via email at kohinoor_cse@lus.ac.bd

References

[1] World Bank. 2019, “Bangladesh Development Update, October 2019: Tertiary Education and Job Skills,” World Bank, Dhaka. https://openknowledge.worldbank.org/handle/10986/32533

[2] F. Khatun and S. Y. Saadat, “Overview of Youth Employment in Bangladesh,” in Youth Employment in Bangladesh, Springer Singapore, 2019, pp. 29–40. doi: 10.1007/978-981-15-1750-1_3.

[3] A. M. Lucietto, D. L. Peters, M. R. Taleyarkhan, and S. Tan, “Academic and Industry Collaboration: A Literature Review,” School of Engineering Education Faculty Publications, Paper 74, available at: https://docs.lib.purdue.edu/enepubs/74.

[4] M. M. Gandhi, “Industry-academia collaboration in India: Recent initiatives, issues, challenges, opportunities and strategies,” The Business & Management Review, vol. 5, no. 2, p. 45, 2014.

[5] C. Wohlin and P. Runeson, “Guiding the selection of research methodology in industry–academia collaboration in software engineering,” Information and Software Technology, vol. 140, p. 106678, 2021.

[6] J. Baleeiro Passos, D. Valle Enrique, C. Costa Dutra, and C. Schwengber ten Caten, “University industry collaboration process: a systematic review of literature,” International Journal of Innovation Science, vol. ahead-of-print, no. ahead-of-print, Jan. 2022, doi: 10.1108/IJIS-11-2021-0216.

[7] N. Firoozeh, A. Nazarenko, F. Alizon, and B. Daille, “Keyword extraction: Issues and methods,” Natural Language Engineering, vol. 26, no. 3, pp. 259–291, Nov. 2019, doi: 10.1017/s1351324919000457.

[8] S. Beliga, “Keyword extraction: a review of methods and approaches,” University of Rijeka, Department of Informatics, Rijeka, vol. 1, no. 9, 2014, Accessed: Aug. 12, 2022. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.704.9230&rep=rep1&type=pdf

[9] P. D. Turney, “Learning algorithms for keyphrase extraction,” Inf Retr Boston, vol. 2, no. 4, pp. 303–336, 2000.

[10] T. D. Nguyen and M.-Y. Kan, “Keyphrase Extraction in Scientific Publications,” in Asian Digital Libraries. Looking Back 10 Years and Forging New Frontiers, Springer Berlin Heidelberg, pp. 317–326. doi: 10.1007/978-3-540-77094-7_41.

[11] O. Medelyan and I. H. Witten, “Thesaurus based automatic keyphrase indexing,” 2006. doi: 10.1145/1141753.1141819.

[12] I. H. Witten, G. W. Paynter, E. Frank, C. Gutwin, and C. G. Nevill-Manning, “KEA,” in Design and Usability of Digital Libraries, IGI Global, 2005, pp. 129–152. doi: 10.4018/978-1-59140-441-5.ch008.

[13] Y.-H. Woo et al., “Automated Keyword Extraction Using Category Correlation of Data,” in Computational Science and Its Applications - ICCSA 2006, Springer Berlin Heidelberg, 2006, pp. 224–230. doi: 10.1007/11751588_24.

[14] E. Papagiannopoulou and G. Tsoumakas, “A review of keyphrase extraction,” WIREs Data Mining and Knowledge Discovery, vol. 10, no. 2, Sep. 2019, doi: 10.1002/widm.1339.

[15] R. Campos, V. Mangaravite, A. Pasquali, A. Jorge, C. Nunes, and A. Jatowt, “YAKE! Keyword extraction from single documents using multiple local features,” Information Sciences, vol. 509, pp. 257–289, Jan. 2020, doi:10.1016/j.ins.2019.09.013.

[16] K. S. Hasan and V. Ng, “Automatic Keyphrase Extraction: A Survey of the State of the Art,” 2014. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) (pp. 1262-1273), doi: 10.3115/v1/p14-1119.

[17] C. Florescu and C. Caragea, “A New Scheme for Scoring Phrases in Unsupervised Keyphrase Extraction,” in Lecture Notes in Computer Science, Springer International Publishing, 2017, pp. 477–483. doi: 10.1007/978-3-319-56608-5_37.

[18] S. R. El-Beltagy and A. Rafea, “KP-Miner: A keyphrase extraction system for English and Arabic documents,” Information Systems, vol. 34, no. 1, pp. 132–144, Mar. 2009, doi: 10.1016/j.is.2008.05.002.

[19] Z. Liu, P. Li, Y. Zheng, and M. Sun, “Clustering to find exemplar terms for keyphrase extraction”, In Proceedings of the 2009 conference on empirical methods in natural language processing (pp. 257-266). August 2009. doi: 10.3115/1699510.1699544.

[20] S. Rose, D. Engel, N. Cramer, and W. Cowley, “Automatic Keyword Extraction from Individual Documents,” in Text Mining, John Wiley & Sons, Ltd, 2010, pp. 1–20. doi: 10.1002/9780470689646.ch1.

[21] M. Won, B. Martins, and F. Raimundo, “Automatic extraction of relevant keyphrases for the study of issue competition.”, In Proceedings of the 20th international conference on computational linguistics and intelligent text processing (pp. 7-13), Apr. 2019. doi: 10.29007/mmk4.

[22] R. Mihalcea and P. Tarau, “TextRank: Bringing Order into Text,” Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, Barcelona, Spain, pp. 404–411, Jul. 2004. Accessed: Aug. 12, 2022. [Online]. Available: https://aclanthology.org/W04-3252

[23] S. J. Kim and S. H. Lee, “An improved computation of the pagerank algorithm,” In European Conference on Information Retrieval (pp. 73-85). Springer, Berlin, Heidelberg, March 2002.

[24] X. Wan and J. Xiao, “Single document keyphrase extraction using neighborhood knowledge.” in AAAI, 2008, vol. 8, pp. 855–860.

[25] S. Danesh, T. Sumner, and J. H. Martin, “SGRank: Combining Statistical and Graphical Methods to Improve the State of the Art in Unsupervised Keyphrase Extraction,” 2015. doi: 10.18653/v1/s15-1013.

[26] C. Florescu and C. Caragea, “Positionrank: An unsupervised approach to keyphrase extraction from scholarly documents,” in Proceedings of the 55th annual meeting of the association for computational linguistics (volume 1: long papers), 2017, pp. 1105–1115.

[27] S. das Gollapalli, X. Li, and P. Yang, “Incorporating Expert Knowledge into Keyphrase Extraction,” Proceedings of the AAAI Conference on Artificial Intelligence, vol. 31, no. 1, Feb. 2017, doi: 10.1609/aaai.v31i1.10986.

[28] A. Hulth, “Improved automatic keyword extraction given more linguistic knowledge,” In Proceedings of the 2003 conference on Empirical methods in natural language processing, pp. 216-223, 2003. doi: 10.3115/1119355.1119383.

[29] S. das Gollapalli and C. Caragea, “Extracting Keyphrases from Research Papers Using Citation Networks,” Proceedings of the AAAI Conference on Artificial Intelligence, vol. 28, no. 1, Jun. 2014, doi: 10.1609/aaai.v28i1.8946.

[30] Q. Zhang, J. Lu, and Y. Jin, “Artificial intelligence in recommender systems,” Complex &amp Intelligent Systems, vol. 7, no. 1, pp. 439–457, Nov. 2020, doi: 10.1007/s40747-020-00212-w.

[31] G. Adomavicius and A. Tuzhilin, “Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions,” IEEE Transactions on Knowledge and Data Engineering, vol. 17, no. 6, pp. 734–749, Jun. 2005, doi: 10.1109/tkde.2005.99.

[32] S. S. Khanal, P. W. C. Prasad, A. Alsadoon, and A. Maag, “A systematic review: machine learning based recommendation systems for e-learning,” Education and Information Technologies, vol. 25, no. 4, pp. 2635–2664, Dec. 2019, doi: 10.1007/s10639-019-10063-9.

[33] G. Salton, A. Wong, and C. S. Yang, “A vector space model for automatic indexing,” Commun ACM, vol. 18, no. 11, pp. 613–620, Nov. 1975, doi: 10.1145/361219.361220.

[34] F. Sebastiani, “Machine learning in automated text categorization,” ACM Computing Surveys, vol. 34, no. 1, pp. 1–47, Mar. 2002, doi: 10.1145/505282.505283.

[35] J. Shu, X. Shen, H. Liu, B. Yi, and Z. Zhang, “A content-based recommendation algorithm for learning resources,” Multimedia Systems, vol. 24, no. 2, pp. 163–173, Mar. 2017, doi: 10.1007/s00530-017-0539-8.

[36] M. M. Rahman and N. A. Abdullah, “A Personalized Group-Based Recommendation Approach for Web Search in E-Learning,” IEEE Access, vol. 6, pp. 34166–34178, 2018, doi: 10.1109/access.2018.2850376.

[37] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl, “Evaluating collaborative filtering recommender systems,” ACM Transactions on Information Systems, vol. 22, no. 1, pp. 5–53, Jan. 2004, doi: 10.1145/963770.963772.

[38] P. Lops, M. de Gemmis, and G. Semeraro, “Content-based Recommender Systems: State of the Art and Trends,” in Recommender Systems Handbook, Springer US, 2010, pp. 73–105. doi: 10.1007/978-0-387-85820-3_3.

[39] P. B.Thorat, R. M. Goudar, and S. Barve, “Survey on Collaborative Filtering, Content-based Filtering and Hybrid Recommendation System,” International Journal of Computer Applications, vol. 110, no. 4, pp. 31–36, Jan. 2015, doi: 10.5120/19308-0760.

[40] A. Felfernig and R. Burke, “Constraint-based recommender systems: technologies and research issues.” In Proceedings of the 10th international conference on Electronic commerce, pp. 1-10. 2008, doi: 10.1145/1409540.1409544.

[41] N. D. Almalis, G. A. Tsihrintzis, N. Karagiannis, and A. D. Strati, “FoDRA - A new content based job recommendation algorithm for job seeking and recruiting”, In 2015 6th International Conference on Information, Intelligence, Systems and Applications (IISA), pp. 1-7. IEEE, July2015. doi: 10.1109/iisa.2015.7388018.

[42] R. Campos, V. Mangaravite, A. Pasquali, A. Jorge, C. Nunes, and A. Jatowt, “YAKE! Keyword extraction from single documents using multiple local features,” Information Sciences, vol. 509, pp. 257–289, Jan. 2020, doi: 10.1016/j.ins.2019.09.013.

[43] S. Kim, O. Medelyan, M. Kan, T. Baldwin, and L. Pingar, “SemEval-2010 Task 5: Automatic Keyphrase Extraction from Scientific,” academia.edu, 2010, Accessed: Aug. 12, 2022. [Online].
Available: https://www.academia.edu/download/30756048/task5.pdf

[44] F. Boudin, “pke: an open source python-based keyphrase extraction toolkit,” in Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: System Demonstrations, Dec. 2016, pp. 69–73. [Online]. Available: https://aclanthology.org/C16-2015

[45] “Nvidia Jetson emmc full encryption with secure boot | Linux | Security | Encryption | Hardware Security Module | Freelancer,” Freelancer.com, 2022.
https://www.freelancer.com/projects/linux/nvidia-jetson-emmc-full-encryption (accessed Aug. 12, 2022)