Engineering and Technology Journal e-ISSN: 2456-3358

Volume 09 Issue 08 August-2024, Page No.- 4958-4965

DOI: 10.47191/etj/v9i08.47, I.F. – 8.227

© 2024, ETJ

Development of Natural Language Processing-Based Descriptive Answer Evaluation Platform (Gradescriptive)

Kosisochukwu Azubogu[1], Emmanuel Chibuogu Asogwa[2], Ifeanyi Charles Ezeugbor[3], Chukwuogo Okwuchukwu Ejike[4], Macdonald Nduaku Onyeizu[5]

ABSTRACT: The manual method of descriptive answer evaluation inherently comes with a lot of problems like the stressful nature of the task, the subjectivity of the grading process as well as the delayed delivery of results. This research involved the development of a computer-based test platform utilizing Natural Language Processing (NLP) as a transformative solution for evaluating descriptive answer examinations. The motivation for this project are the issues of slow turnaround times, potential bias, and limited scalability faced in the manual method of evaluating descriptive answers. Leveraging a state-of-the-art large language model, the MERN (MongoDB, Express.js, React.js and Node.js) stack and Cascading Style Sheets (CSS), a system that meticulously analyzes student responses using criteria like textual semantic similarity, keyword matching and answer length, was developed. The results of the project include timely and accurate feedback, alleviating anxieties and uncertainties around students' performances. It showed that descriptive questions can evaluate students' critical thinking, problem-solving, and creativity, unlike objective tests. Meanwhile, lecturers are relieved of the immense stress associated with traditional manual grading, fostering a more positive and productive learning environment.

KEY WORDS: Natural Language Processing (NLP), Descriptive-Answer, Computer-based test, Large Language Models, Embeddings.

I. INTRODUCTION

In the academic world, exams have always been an integral part of the activities being carried out as a means of testing the knowledge or skills of students in a particular area or subject. The two main methods by which exams are carried out are through objective questions and descriptive questions. For the objective type, the student is to pick an option from a list of provided options as the correct answer to a question, while for the descriptive type, the student is to correctly explain certain concepts as taught in the classroom. For many years, these exam types have been written with pen and paper by the students and marked likewise by the lecturers. These means prove to be time-consuming and tiring, mainly for the lecturers, but with the rise in computer literacy, software platforms have been created to aid the writing and evaluation of these exams especially the objective type. For the descriptive type however, not much progress has been made in trying to automate the process. Many institutions, especially in Nigeria, still make use of the pen and paper method for both writing and evaluating descriptive examinations. Again, as a result of the slow and stressful nature of evaluating and producing results in descriptive examinations, students are unable to see their grades on time which leads to a host of other problems in their academic lives such as the uncertainty of the academic performance and standing of a student or the unnecessary extension in the length of time spent in the university by students. The automatic descriptive answer analysis systems are very cooperative for numerous universities and academic institutions to assess a student's performance terribly effectively [1].

[2]The automatic answer script analysis supported by Natural Language process (NLP) can facilitate us to beat the difficulties featured within the manual analysis. Here a student's written answer is provided as input and also the system can automatically score marks once the analysis. The system considers all attainable factors like orthography error, grammatical error, and varied similarity measures for scoring marks.

The primary objective of this work is to introduce a descriptive grade system (Grade scriptive) to eliminate the challenges associated with objective examination test.

With the rise in the technologies of artificial intelligence and machine learning (particularly Natural Language Processing), there has been an increase in the number of language models that are able to understand natural languages to a decent level and even generate text in these same languages. This research work proposes solution to the problems posed by the pen and paper method of writing and marking exams, by developing a web-based computer-based test (CBT) platform that utilizes natural language processing techniques like stop-word removal, stemming, lemmatization, and language models for assigning marks based on the level of similarity between the students’ answers and the lecturers’ answers. The system leverages the use of natural language processing techniques and models to perform the task of evaluating the level of similarity between the answer given by the student and that in the lecturer’s marking guide.

This platform will also enable lecturers to automatically send the results of the evaluation to the email addresses of the students. The development and use of this platform would be able to solve the problems of stress on the lecturer, delay in marking of exam scripts and the release of students’ results.

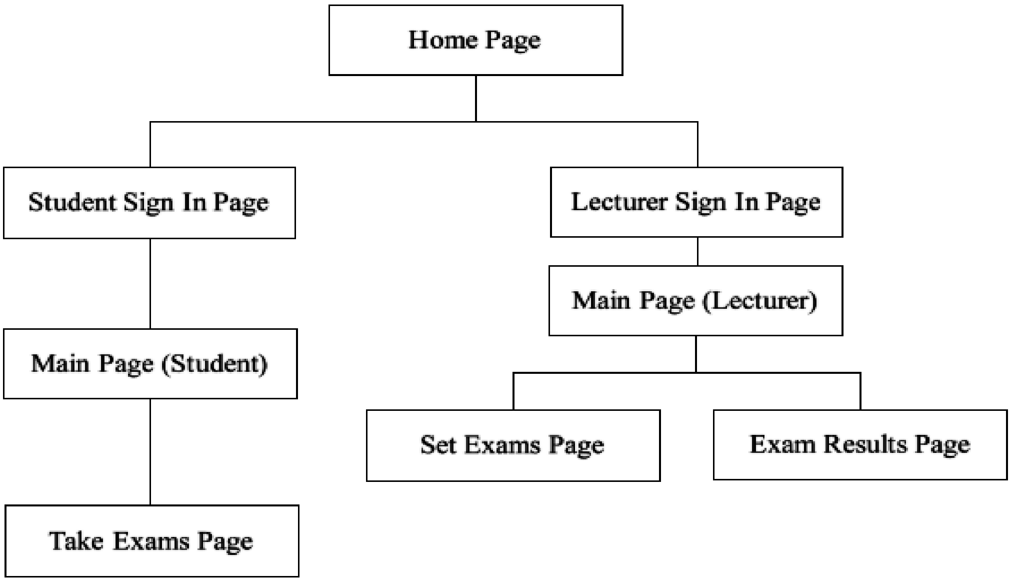


Figure 1: High Level Model of Gradescriptive

II. RELATED LITERATURE

Several articles have been published on the development of descriptive answer evaluation systems with varying methods of evaluating answers. [3], developed a web application that made use of natural language processing for subjective answer evaluation. It was also divided into four modules: the login module, the preprocessing module, the information extraction module and the score generation module. In the information extraction module, the important and relevant keywords are matched using cosine similarity. The formula for cosine similarity is as follows:

$$\frac{Dotproduct(teacher, student)}{\| teacher \| \times \| student \|}$$

[4], developed a system which used the technique of keyword matching to ascertain the correctness of students’ answers was built. Keyword matching involves comparing the keywords expected in a particular answer against the words used by the students in answering the question. The more the keywords matched, the more the marks earned for that question. Amongst the works making use of keyword matching, some considered the use of synonyms when evaluating answers. [5], in their research made use of the WordNet tool for generating synonyms of the keywords for a fairer and more accurate evaluation. Other works made use of textual semantic similarity by converting the answers to vector embeddings and then comparing the closeness of these vectors using a similarity measure e.g cosine similarity. The closer the vectors, the more similar the answers are. For the conversion of text to embeddings, [6] made use of the transformer-based bidirectional encoder representations from transformers (BERT) model, taking advantage of the attention mechanism built into transformers. Some other works added an extra layer of complexity by making use of optical character recognition (OCR) for hand-written exams. One common process amongst most of the works was the use of some of the popular natural language processing (NLP) techniques like stop-word removal, lemmatization, stemming, etc to clean up the text for a more efficient evaluation process. In their work, [7] trained a model to evaluate answers without the need of keywords. They first evaluated the answers using keywords and some similarity-based techniques like word mover’s distance (WMD). Then from the obtained results, they trained the model to predict the possible amount of marks a particular answer would get. Asides WMD they made use of other similarity measures like the cosine similarity, Jaccard similarity, Bag of Words (BoW) and TF-IDF (Term Frequency-Inverse Document Frequency). They also used preprocessing techniques and a word embedding generator called Word2Vec.

[7], developed a system that used keyword matching between the answers given by the students and the expected keywords specified for that question. They employed a multi-layered neural network algorithm to be trained to detect these keywords that may appear in the students’ answers. The model’s activation function was the Rectified Linear unit (ReLU). Answer length was also considered as a parameter for scoring the students. The students’ answers for this work was extracted from an image - most probably an image of the student’s answer sheet.

[8] In their study, explored keyword-based text summarization. They also involved the use of NLTK for preprocessing techniques such as tokenization, stopword removal, lemmatization, bigram creation and word frequency count. [9] did a similar work with Supriya et al., only that this time, they involved the minimum length of the answer provided by the lecturer as a parameter for scoring. If the student’s answer was less than the stipulated minimum answer length, the student was assigned a mark from 0 to 4 depending on the percentage of matched keywords and if the student’s answer was more than or equal the stipulated minimum answer length, the student was assigned a mark from 0 to 10, again, depending on the percentage of matched keywords.

From the list of works that were reviewed, there are certain similarities and flaws spotted in their functionalities. Firstly, some of the works made use of strictly keyword matching, not considering the fact that in some situations some students might decide to use words or phrases related in meaning to these keywords. This could restrict the creativity of the students, especially those with a large vocabulary, as they are being limited to using only certain words in order to attain the maximum possible marks for a question. Some other works that did consider the use of similar words, did not consider context, i.e. they did not evaluate the semantic meanings of the sentences. These limitations formed the knowledge gap which and strenghthens our motive for carring out this research.

III. MATERIALS AND METHOD

The entire system was built using the MERN (MongoDB, Express.js, React.js and Node.js) stack. React.js was used for the front-end (or client-side), Express and Node.js was used for the back-end (or server-side) and MongoDB was used as the database. For an accurate evaluation process, we considered three criteria for the development of this system. Each of these methods have different weights of importance which they contribute to the overall marks assigned to a question. These methods include: keyword or key phrase matching, textual semantic similarity and answer length evaluation.

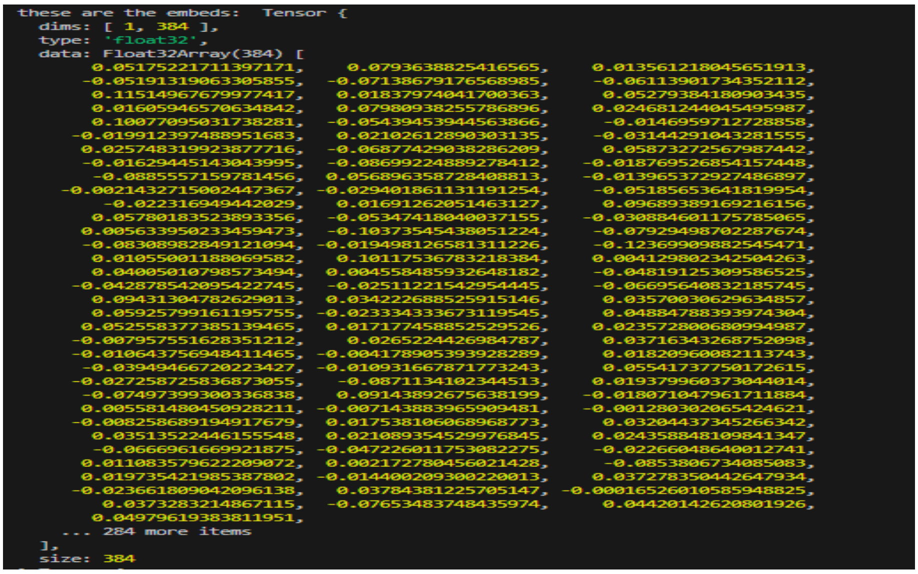
3.1 Keyword or Key Phrase Matching

As stated above, keyword matching involves searching the answer provided by the student for the occurrence of the keywords expected for a proper description of a concept. The keywords, in this case, will be provided by the lecturer. As expected the students will most likely not use the exact same words when giving answers. Hence, there will be the use of the WordNet tool to generate synonyms of each of the keywords for a well-rounded evaluation. The percentage weight attached to this criteria is 45%, i.e. a perfect score for this metric earns you 45% of the total marks allotted to the question in particular. The exact use of key phrases, as well as the use of semantically similar keyphrases were also considered. A keyphrase is made up of two or more words that are crucial for accurately defining a concept. A keyphrase may not have been exactly used when describing an answer, e.g the phrase “fuse together” and “come together“ are certainly not the same in terms of the words used, but they portray a similar meaning [10]. A transformer-based language model by name “Xenova/all-MiniLM-L12-v2”, was used to compare the semantic similarity between phrases to determine if a keyphrase was actually used in the student’s answer. Embeddings of the phrases were generated and compared using cosine similarity for closeness, i.e. the closer the embeddings, the more similar the phrases. Cosine similarity outputs range from the values of 0 to 1, 0 for least similar and 1 for most similar. The formula for measuring the cosine similarity between embeddings is:

$$\frac{Dotproduct(teacher, student)}{\| teacher \| \times \| student \|}$$

3.2 Textual Semantic Similarity

For calculating textual semantic similarity, embeddings for both of the answers were generated using the transformer model mentioned in the previous section, i.e. the student’s answer and the lecturer’s answer. Then using the cosine similarity measure, the closeness of the answers were calculated. Again, the closer the answers are, the more likely it is that the student has provided the correct definition of a concept. The vector embeddings were plotted against a dimensional space of 384 dimensions. The percentage weight assigned to this criteria was also 45%, i.e. achieving the full marks for this criteria earns you 45% of the marks allotted to that question. Below is a screenshot of what a vector embedding generated with this language model looks like:



```
these are the embeds:  Tensor {
  dims: [ 1, 384 ],
  type: 'float32',
  data: Float32Array(384) [
       0.05175221711397171,    0.0793638825416565,    0.013561218045651913,
      -0.05191319063305855,  -0.07138679176568985,   -0.06113901734352112,
       0.11514967679977417,   0.01837974041700363,    0.05279384180903435,
       0.01605946570634842,   0.07980938255786896,   0.024681244045495987,
       0.10077095031738281,  -0.05439453944563866,    -0.0146959712728858,
     -0.019912397488951683,   0.02102612890303135,   -0.03144291043281555,
      0.025748319923877716,  -0.06877429038286209,    0.05873272567987442,
      -0.01629445143043995,  -0.08699224889278412,  -0.018769526854157448,
       -0.0885557159781456,  0.056896358728408813,  -0.013965372927486897,
    -0.0021432715002447367, -0.029401861131191254,   -0.05185653641819954,
        -0.022316949442029,   0.01691262051463127,    0.09689389169216156,
       0.05780183523893356,  -0.05347418040037155,  -0.030884601175785065,
      0.005633950233459473,  -0.10373545438051224,   -0.07929498702287674,
      -0.08308982849121094, -0.019498126581311226,   -0.12369909882545471,
       0.01055001188069582,   0.10117536783218384,   0.004129802342504263,
       0.04005010798573494,  0.004558485932648182,   -0.04819125309586525,
     -0.042878542095422745,  -0.02511221542954445,   -0.06695640832185745,
       0.09431304782629013,  0.034222688525915146,    0.03570030629634857,
       0.05925799161195755, -0.023334333673119545,    0.04884788393974304,
      0.052558377385139465,  0.017177458852529526,   0.023572800680994987,
     -0.007957551628351212,    0.0265224426984787,    0.03716343268752098,
     -0.010643756948411465, -0.004178905393928289,    0.01820960082113743,
      -0.03949466720223427, -0.010931667871773243,    0.05541737750172615,
     -0.027258725836873055,   -0.0871134102344513,   0.019379960373044014,
      -0.07497399300336838,   0.09143892675638199,  -0.018071047961711884,
      0.005581480450928211, -0.007143883965909481,  -0.0012803020654224621,
     -0.008258689194917679,  0.017538106068968773,    0.03204437345266342,
       0.03513522446155548,  0.021089354529976845,   0.024358848109841347,
       -0.0666961669921875, -0.047226011753082275,   -0.02266048640012741,
      0.011083579622209072, 0.0021727804560214 28,    -0.0853806734085083,
      0.019735421985387802, -0.014400209300220013,   0.037278350442647934,
     -0.023661809042096138,   0.03784381225705147, -0.00016526010585948825,
        0.0373283214867115,  -0.07653483748435974,    0.04420142620801926,
       0.04979619383811951,
    ... 284 more items
  ],
  size: 384
```

Figure 2: A sample of a vector embedding generated by the Xenova/all-MiniLM-L12-v2 language model.

3.3 Answer Length Evaluation

The length of the students’ and lecturer’s answers were also compared to determine the correctness of the students’ answer. If the length student’s answer was at least 70% of the length of the lecturer’s answer, the full marks for answer length evaluation are given, if not the marks are allotted based on the percentage to which it was as long as. For example, if the student’s answer is 60% of the length of the lecturer’s answer, the total marks for answer length evaluation will be 60% of the marks allotted for answer length evaluation. The percentage weight of importance for this similarity measure is 10%.

All the marks for each criteria are calculated independently and are summed up after calculation to get the total marks scored for a question

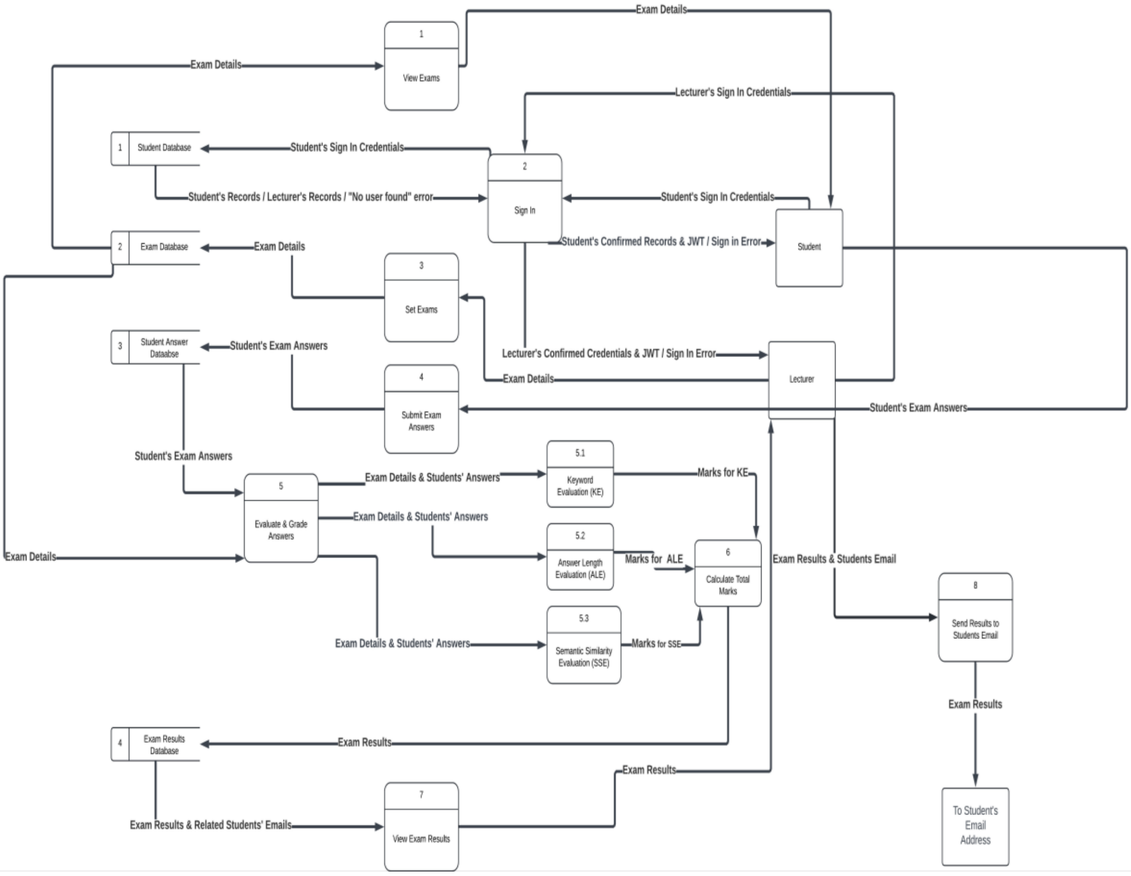

Exam Details
1
View Exams
Exam Details
Lecturer's Sign In Credentials
1
Student Database
Student's Sign In Credentials
2
Sign In
Student's Records / Lecturer's Records / "No user found" error
Student's Sign In Credentials
2
Exam Database
Exam Details
Student's Confirmed Records & JWT / Sign in Error
Student
3
Set Exams
3
Student Answer Dataabse
Student's Exam Answers
Lecturer's Confirmed Credentials & JWT / Sign In Error
Exam Details
Lecturer
4
Submit Exam Answers
Student's Exam Answers
Student's Exam Answers
5
Evaluate & Grade Answers
Exam Details & Students' Answers
5.1
Keyword Evaluation (KE)
Marks for KE
Exam Details & Students' Answers
5.2
Answer Length Evaluation (ALE)
Marks for ALE
6
Calculate Total Marks
Exam Results & Students Email
Exam Details
5.3
Semantic Similarity Evaluation (SSE)
Exam Details & Students' Answers
Marks for SSE
8
Send Results to Students Email
Exam Results
4
Exam Results Database
Exam Results
Exam Results
7
View Exam Results
Exam Results & Related Students' Emails
To Student's Email Address

Figure 3: A Data flow diagram (DFD) of Gradescriptive model

The entire system has different processes and therefore different algorithms. This section outlines the algorithms for each process ongoing during the operation of the system. The various processes are:

3.5 Algorithm for setting exams (for the lecturer)

1. Click on the "Set Exams" button.
2. On the "Set Exams” page enter the exam details, including title, duration, questions, answers, and keywords.
3. After filling in all the information, click the "Submit" button.

On submission, a request is made to the appropriate end point on the server to save the exam details to the database. The request is processed and a response is returned indicating whether the operation was successful. If successful, a toast notification is triggered to inform the lecturer about the successful save of the exam answers.

3.5.1 Taking exams (for the student)

1. Click on the "Take Exams" button.
2. On the "Exams List" page, select a particular exam to write.
3. On the exam interface for the selected exam, answer the questions to the best of your knowledge and click submit when done. The student should remember to provide his or her email address before submission.

On submission using the “submit” button, a request is made to the appropriate end point to save the answers to the database. The request is processed and a response is returned indicating whether the operation was successful. If successful, a toast notification is triggered to inform the student about the successful save of the exam answers.

3.5.2 Answer evaluation process

The answer evaluation process, is further divided into three modules as stated in the methodology section: answer length evaluation, textual semantic similarity and keyphrase or keyword matching.

3.5.3 Answer Length Evaluation

The algorithm for answer length evaluation goes as follows:

1. After the student has submitted the answers to his or her exams, exams are saved to the database, in the appropriate collection.
2. After saving, the student’s answers are retrieved alongside the exam questions as well as the lecturer’s answers.
3. Both the exam details (set by the lecturer) and the student’s answers are passed in as parameters to the function for calculating the marks awarded for answer length.
4. In the function, the student’s answers and the lecturer’s answers are preprocessed and their length are calculated.
5. Using these lengths, the percentage of the lecturer’s answer which was covered by the student is evaluated.
6. If the percentage is greater than 70, the student’s answer is then awarded full marks. This is good indication that student may have written extensively on the question being asked and may have covered all necessary points for a proper answer. If the percentage however, is less than 70, the marks awarded will be based off of the calculated percentage.
7. Finally, the evaluated marks are returned from the function.

3.5.4 Algorithm for Textual Semantic Similarity

1. The student’s answers and the exam details are retrieved and passed as parameters to the function for calculating the marks awarded for the semantic similarity.
2. In the function, the language model for generating the vector embeddings is loaded.
3. After loading, the model is used to generate the vector embeddings for the similarity comparison.
4. The vector embeddings for the student’s answers and the lecturer’s answers are then compared using the cosine similarity measure.
5. The similarity result is then converted to a percentage value.
6. The percentage value from the similarity result is then used to evaluate marks to be awarded to the student.
7. The evaluated marks are then returned at the end of the function.

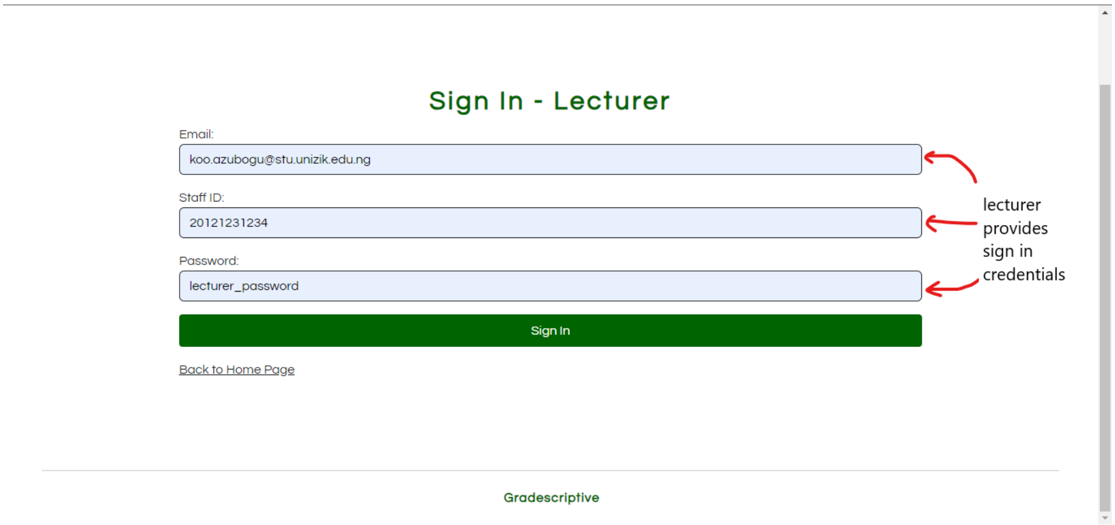
3.5.5 Keyword or Key Phrase Evaluation

1. The student’s answer and the exam details set by the lecturer are retrieved and passed as parameters into the function.
2. The student’s answer is tokenized and the stop words of the answer are removed.
3. From the exam details, the key terms are retrieved. For the key phrases, the stop words are removed.
4. An array for storing the percentage level of key terms with a match is created.
5. Each key term is then checked for if it is a phrase or just a word.
6. For keywords, a list of possible synonyms is generated and then a direct comparison is made with the tokens of the student’s answer. If there is a match, it is assigned a match percentage of 100%. This value is pushed to the match percentage array.
7. For key phrases, the vector embeddings of the phrase is generated and compared with the vector embeddings of the n-grams of the student’s tokenized answer. This comparison is done with cosine similarity. Depending on the value of the cosine similarity, the match percentage value is calculated and is appended to the array.

At the end, the array of match percentage values is returned to be saved in the exam results collection of the database. These results are then displayed to the lecturer on the lecturer module to be sent via email to the respective students. Below is a flowchart of the answer evaluation process:

IV. DISCUSSION OF RESULTS

The result of this project was achieved using NLP, MERN, Xenova/all-MiniLM-L12-v2, WordNet language model for the development. Our findings showed that descriptive automated questions-answer model is effective in evaluating students' critical thinking, problem-solving, and creativity, in contrast to objective tests. Furthermore, descriptive model enabled us to replicate real-world situations, resulting in a more genuine evaluation of students' skills and difficulties. Our model is divided into two modules, the student module and the lecturer module. Both modules have a sign-in form for user authentication. For the lecturer module, the main function performed is the preparation of exam questions along with the answers, keywords, the marks for each question and the duration of the exam. This information is filled in by the lecturer and is submitted to the database, particularly the exams collection. The exams collection in the database is a collection of all the exams that have been set by the lecturers, which is then retrieved on request and displayed in the browser for the student. The keywords for an answer have to be separated with commas as this helps in data extraction and the preprocessing of these keywords. The lecturer can send the results of the students to their email while he uses his copy to produce the general result.



Sign In - Lecturer

Email: koo.azubogu@stu.unizik.edu.ng

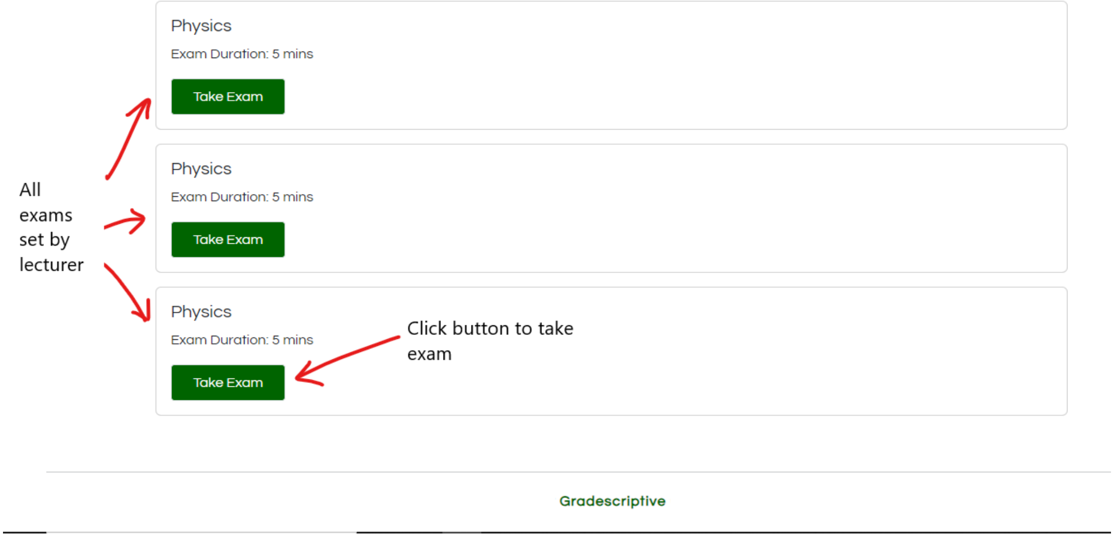
Staff ID: 20121231234

Password: lecturer_password

lecturer provides sign in credentials

Figure 5: Login for Lecturer and Students

For the student's module, the students get to select an exam from the whole collection of exams available. When a student makes a selection a request is made to the back end and the details for that exam are retrieved. Whenever the student is done with answering the set questions, he or she can click the submit button and the exam is submitted along with the provided answers to the back-end for evaluation and for storage in the database. At the back-end, the scores of the answers are calculated and sent to the lecturer's module from where the lecturer has the option of sending the results to the respective emails of the students.



Physics
Exam Duration: 5 mins

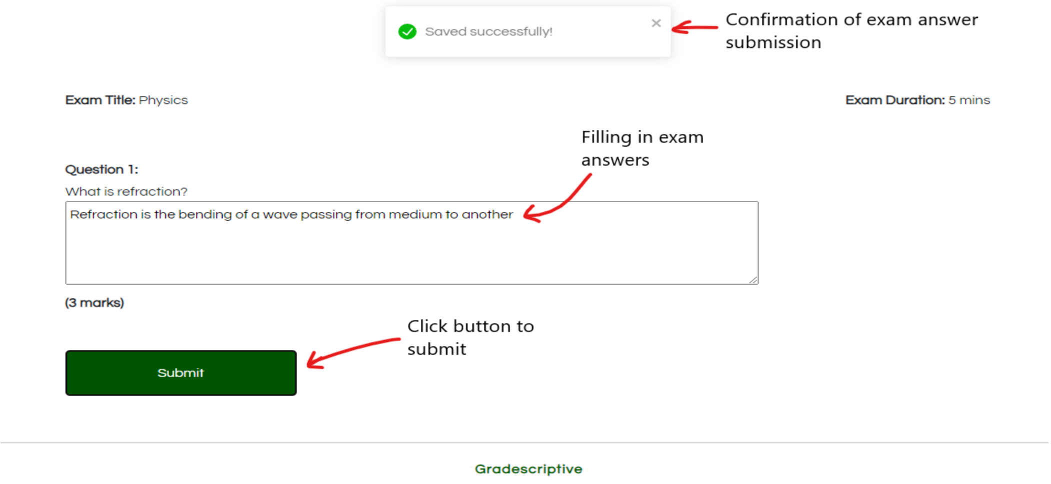
Physics
Exam Duration: 5 mins

Physics
Exam Duration: 5 mins

All exams set by lecturer

Click button to take exam

Figure 6: Descriptive exam page



Saved successfully! ×

Confirmation of exam answer submission

Exam Title: Physics

Exam Duration: 5 mins

Filling in exam answers

Question 1:

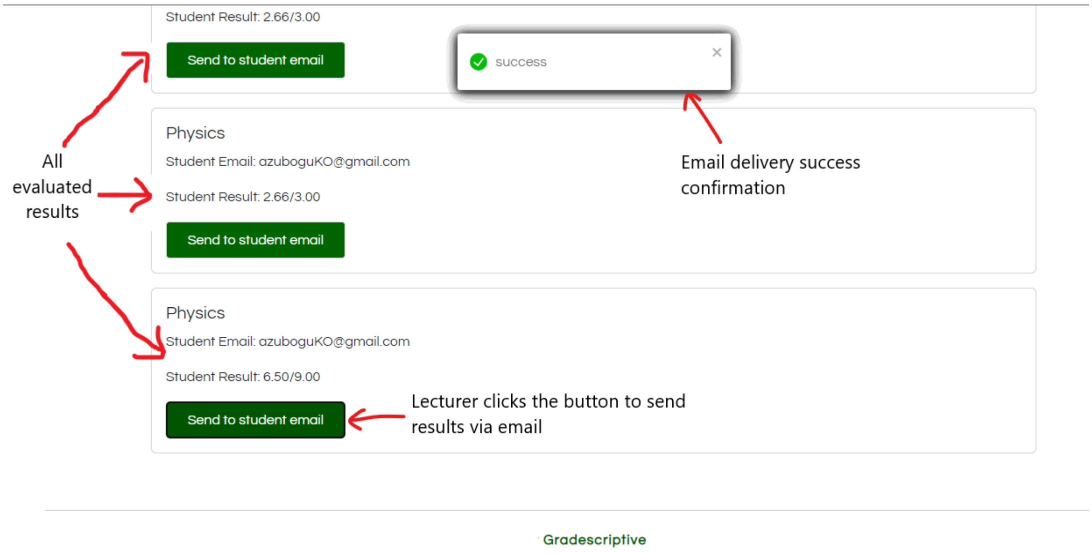
What is refraction?

Refraction is the bending of a wave passing from medium to another

(3 marks)

Click button to submit

Figure 7: Student’s successfully answered question page



Student Result: 2.66/3.00

success ×

Email delivery success confirmation

All evaluated results

Physics

Student Email: azuboguKO@gmail.com

Student Result: 2.66/3.00

Physics

Student Email: azuboguKO@gmail.com

Student Result: 6.50/9.00

Lecturer clicks the button to send results via email

Figure 8: Successfully result delivery page

V. CONCLUSION AND FUTURE WORK

This project report presents the design and development of an answer evaluation system for descriptive answer examinations. The system incorporated natural language processing (NLP) techniques like lemmatization, stopword removal to build up the modules for evaluating student’s results. Assessing critical thinking, problem-solving, and creativity of the students can be accomplished by using the model. Additionally, descriptive questions allow the lecturer to simulate real-world tasks and challenges. Three criteria were considered when building out the system, they include: answer length, textual semantic similarity and keyword matching. The system also involved sending emails containing the evaluation results to the appropriate destination emails to reduce the hassles students and lecturers go through when checking for or delivering results respectively. This system eliminates the delay usually experienced by students who wait for their results for an extended period of time and are being kept in the dark concerning their academic performance and standing. The result of our study is an automated evaluation platform that efficiently assesses descriptive answers across various subjects, significantly reducing the workload of human graders.

REFERENCES

1. V. Paul and J. D. Pawar (2014). Use of Syntactic Similarity Based Similarity Matrix for Evaluating Descriptive Answer. 2014 IEEE Sixth International Conference on Technology for Education, Clappana, pp. 253- 256
2. Prof. Sumedha P Raut1 et al., (2022). Automatic Evaluation of Descriptive Answers Using NLP and Machine Learning. 2022 International Journal of Advanced Research in Science, Communication and Technology (IJARSCT). Page 736.
3. Meenakshi et al. (2022). Web App for Quick Evaluation of Subjective Answers Using Natural Language Processing. Scientific and Technical

Journal of Information Technologies, Mechanics and Optics. 22(3) page 594.

4. Das, I., Sharma, B., Rautaray, S. S., & Pandey, M. (2019). An Examination System Automation Using Natural Language Processing. Conference: 2019 International Conference on Communication and Electronics Systems (ICCES). https://doi.org/10.1109/icces45898.2019.9002048.
5. R. K. Rambola et al.,(2021). Development of Novel Evaluating Practices for Subjective Answers Using Natural Language Processing. Springer Link. Page 206.
6. Harsh et al., (2022). Automatic Grading of Handwritten Answers. International Research Journal of Engineering and Technology (IRJET). 9(5) page 409.
7. Bashir, M. F., Arshad, H., Javed, A. R., Kryvinska, N., & Band, S. S. (2021). Subjective Answers Evaluation Using Machine Learning and Natural Language Processing. IEEE Access, 9, 158972-158983. https://doi.org/10.1109/ACCESS.2021.3130902.
8. Sinha, P., Bharadia, S., Kaul, A., & Rathi, S. (2018). Answer Evaluation Using Machine Learning. McGraw-Hill Publications.
9. Raut, S. P., Chaudhari, S. D., Waghole, V. B., Jadhav, P. U., & Saste, A. B. (2022). Automatic evaluation of descriptive answers using NLP and machine learning. International Journal of Advanced Research in Science, Communication and Technology, 735–745. https://doi.org/10.48175/ijarsct-3030.
10. Nandita et al. (2021). Automatic Answer Evaluation Using Machine Learning. International Journal of Information Technology (IJIT). 7(2). Page 1.