SummUp (CheckMyAnswer)

Submitted in partial fulfillment of the requirements
of the degree of

BACHELOR OF ENGINEERING

(Computer Engineering)

by

1. **Ms. Pooja G. Bhagat (BE/A/110)**
2. **Ms. Priyanka D. Korde (BE/A/113)**
3. **Mr. Asmit S. Patil (BE/A/131)**
4. **Mr. Raghuwardayal A. Maurya (BE/A/154)**

Guide

Prof. Rajendra D. Gawali

Department of Computer Engineering

Lokmanya Tilak College of Engineering

Sector-4, Koparkhairne, Navi Mumbai

(2022-2023)

Certificate

This is to certify that the project entitled "**SummUp (CheckMyAnswer)**" is a bonafide work of **Pooja Bhagat (BEA-110), Priyanka Korde (BEA-113), Asmit Patil (BEA-131), Raghuwardayal Maurya (BEA-152)** submitted to the University of Mumbai in partial fulfillment of the requirement for the award of the degree of "**Bachelor of Engineering**" in **"Computer Engineering**"

(Prof. Rajendra D. Gawali)
Guide

(Prof. Sonal Bankar)
Head of Department

(Dr. Vivek Sunnapwar)
Principal

PROJECT REPORT APPROVAL FOR B.E.

The project report entitled "**SummUp (CheckMyAnswer)**" by **Pooja Bhagat (BEA-110), Priyanka Korde (BEA-113), Asmit Patil (BEA-131), Raghuwardayal Maurya (BEA-154)** is approved for the award of "**Bachelor of Engineering**" degree in "**Computer Engineering**".

Examiners

1. ______________________________

2. ______________________________

Date:

Place: Koparkhairane, Navi Mumbai

DECLARATION

I declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principals of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/ data / fact / source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

________________________	________________________
Ms. Pooja G. Bhagat	Ms. Priyanka D. Korde
________________________	________________________
Mr. Asmit S. Patil	Mr. Raghuwardayal A. Maurya

Date:

ABSTRACT

It has been observed that a wide range of students applied for different kinds of exams, including institutional, non-institutional, and occasionally even competitive ones. We are dealing with multiple-choice questions in several tests because they are objective. In these circumstances, automated scoring is applied. We are still having difficulties automating scoring for subjective papers. Designing an algorithm to automate answer evaluation is the major objective of our research work and deliver an automated score utilizing Machine learning approach to maintain the scoring process while reducing the amount of human effort.

ACKNOWLEDGEMENT

We remain immensely obliged to Prof. Rajendra D. Gawali for providing us with the idea of the topic, for his invaluable support in gathering resources, his guidance and supervision which made this work successful.

We would like to give our thanks to the Head of the Computer Department, Prof. Sonal Bankar, Vice Principal Dr. Subhash Shinde and our Principal Dr. Vivek Sunnapwar.

We are also thankful to the faculty and staff of the Computer Engineering Department and Lokmanya Tilak College of Engineering, Navi Mumbai for their invaluable support.

We would like to say that it has indeed been a fulfilling experience working out this project topic.

LIST OF FIGURES

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

Examinations have always been part of every educational, and non-educational organization. Examinations can be either descriptive or objective or both. Every examination needs evaluation. The majority of competitive exams are objective in structure. They happen on similar machines that have been examined. These systems, or any other related methods, offer greater advantages in terms of resource conservation. However, it has been noticed that these systems can only contain multiple-choice questions and cannot be expanded to include subjective questions. These methods cannot be used in board exams or university exams where students give subjective answers due to a few issues, hence there is a need for software that will aid in conserving resources.

It can be seen the amount of pressure that is held on the education system and teachers to evaluate the n number of answer copies of the students. On average, each institute has four examinations per year, resulting in more than 6.4 million answer sheets being generated. Keeping faculty in mind, evaluating papers, and assigning grades are time-consuming tasks. The traditional exam typically consisted of subjective answers, which were not the most effective way of assessing the student's understanding of the subject. Because examiners can be bored checking so many answer sheets, there may be an increase in false evaluations. As a result, the Answer Verifier is required to grade the student after he or she has completed the question paper. The process of evaluating the descriptive answer will not only save resources but will also overcome human limitations. It will also help to speed up the overall educational system because students will not have to wait as long for a reason.

The evaluation of subjective answers has long been a challenge for educators, employers, and researchers. CheckMyAnswer, powered by machine learning algorithms, has emerged as a solution to this challenge. These checkers can analyse and evaluate subjective responses, providing objective and consistent feedback to users. However, the development and implementation of machine learning algorithms come with challenges, including the need for large amounts of training data and ensuring the transparency and explain ability of the algorithms.

1.2 MOTIVATION

Subjective answer checking is an essential process in many areas, such as education, recruitment, and competitive exams. However, manual evaluation of subjective answers is time-consuming, expensive, and prone to subjective biases. As a result, there is a growing need for automated subjective answer checking systems that can efficiently and accurately evaluate subjective responses. This can significantly reduce the time and effort required to evaluate subjective responses.

Furthermore, it can provide consistent and reliable feedback to students or candidates, which can help to improve their performance and increase their chances of success. Moreover, a subjective answer checker can enable the scaling up of evaluation processes, making it possible to evaluate a large number of responses accurately.

In this project, CheckMyAnswer we will analyse and evaluate subjective responses, providing score to students or users.

CHAPTER 2

LITERTURE SURVEY

2.1 SURVEY OF EXISTING SYSTEM

Sr. No.	Authors	Title of the paper & year of publication (Old to recent)	Major contribution
1.	Ashutosh Shinde, Nishit Nirbhavane, Sharda Mahajan, Vikas Katkar, Supriya Chaudhary	Ai Answer Verifier 2018	Used to grade the Multiple choice questions and provided scoring on keywords, length and grammar.
2.	Jagadamba G, And Chaya Shree G	Online Subjective answer verifying system Using Artificial Intelligence 2020	Used cosine similarity, text gears grammar API. Systems efficiency was found to be in range of 60-90% based on different parameters(text length, keyword matching and both).
3.	Muhammad Farrukh Bashir, Hamza Arshad, Abdul Rehman Javed Natalia Kryvinska And Shahab S. Band	Subjective Answers Evaluation Using Machine Learning and Natural Language Processing 2021	Cosine similarity performs poorly semantic-wise compared to WDM but can make some pretty good estimates where semantics are unnecessary.
4.	Prof. Era Johri, Prem Chandak, Nidhi Dedhia, Hunain Adhikari, Kunal Bohra	ASSESS – Automated subjective answer evaluation using Semantic Learning 2021	Google's USE algorithm to generate sentence embeddings. It is use to encode both the answers into vectors and similarity.

5.	Shreya Singh, Prof. Uday Rote, Omkar Manchekar, Prof. Sheetal Jagtap, Ambar Patwardha, Dr. Hariram Chavan	Tool for Evaluating Subjective Answers using AI 2021 (TESA)	Implementation of character recognition. Also sentence splitting jaccard similarity and BERT was used.
6.	Ronika Shrestha, Raj Gupta and Priya Kumari	Automatic Answer Sheet Checker 2022	The answer is captured in form of photos. Then the answers are scanned and the keywords are picked from the photos. And the system will automatically calculate result using two algorithms of NLP and ANN.

CHAPTER 3

PROPOSED SYSTEM

3.1 PROBLEM STATEMENT AND OBJECTIVES

Problem Statement:

Traditional methods of grading and assessment can be time-consuming, prone to human error. As a result, organizations may struggle to accurately assess the answers provided by employees or students, leading to suboptimal performance and decreased productivity. CheckMyAnswer provides a solution by automating the evaluation process and providing more accurate and efficient feedback.

Objectives:

The objective of a subjective answer checking model is to automate the process of evaluating and grading subjective answers given by students to open-ended questions in educational assessments. The model should be able to assess factors such as relevance, coherence, clarity, and overall understanding of the topic.

The main goals of a subjective answer checking model are:

1. To reduce the time and effort to evaluate large volumes of subjective answers, while maintaining the accuracy and fairness of the grading process.
2. To provide consistent and reliable grading of subjective answers, reducing the risk of human bias and variability.
3. To improve the quality of educational assessments by ensuring that subjective answers are graded based on standardized criteria and objective measures.

By automating the subjective answer checking process, educational institutions can save time and resources, allowing teachers and administrators to focus on other important tasks such as developing curriculum, providing feedback to students, and improving the overall quality of education. Additionally, students can receive prompt score on their performance.

3.2 SCOPE OF THE WORK

CheckMyAnswer is an online tool designed to help students and teachers check the accuracy of their answer.

The scope of CheckMyAnswer includes the following:

- Providing immediate score: CheckMyAnswer provides immediate feedback to students on the accuracy of their answers, helping them to identify and correct any mistakes in their answers.
- Helping teachers to save time: CheckMyAnswer can help teachers save time by automating the process of checking students' answers, allowing them to focus on other aspects of teaching.
- Facilitating independent learning: CheckMyAnswer can help students to learn independently by allowing them to check their answers without the need for a teacher's intervention.
- Promoting self-assessment: CheckMyAnswer can promote self-assessment and self-correction, as students can use the tool to identify and correct their own mistakes.

3.3 ANALYSIS

Cosine Similarity:

Cosine similarity is a popular similarity measure used in text analytics for comparing the similarity of two text documents. It measures the cosine of the angle between two vectors in a multi-dimensional space, and is often used because of the following reasons:

1. It is computationally efficient: Cosine similarity is relatively simple and efficient to compute, especially when compared to other more complex similarity measures, such as Euclidean distance.

2. It is scale-invariant: Cosine similarity is not affected by the scale of the vectors being compared, as it only measures the angle between them. This means that it can be used to compare documents of different lengths without any normalization or weighting.

3. It is widely used in natural language processing: Cosine similarity is a widely used similarity measure in the field of natural language processing, and many popular machine learning models, such as BERT and word2vec, use cosine similarity as part of their training and evaluation.

Cosine similarity is a useful measure for finding text similarity between two documents or vectors, but it may not always be the best choice for comparing the similarity of two sentences. Here are some reasons why:

1. Sentence length: Cosine similarity is sensitive to the length of the vectors being compared. In the case of sentences, two sentences with different lengths will be represented by vectors of different dimensions, which can make the cosine similarity less effective.
2. Syntax and grammar: Cosine similarity does not consider the syntax or grammar of the sentences, which can be important for determining their similarity. For example, two sentences with the same words in a different order may have a low cosine similarity score even if they are semantically similar.

3. Word order and semantics: Cosine similarity also does not take into account the order of words in a sentence or their semantic relationships. Two sentences that express the same idea but use different words or phrasing may have a low cosine similarity score, even if they are semantically equivalent.

4. Context: Cosine similarity does not consider the context in which the sentences appear. Two sentences that have different meanings in different contexts may have a high cosine similarity score, even if they are not semantically similar in the given context.

Cosine similarity can be useful for finding text similarity between documents or vectors, it may not always be the best choice for comparing the similarity of two sentences.

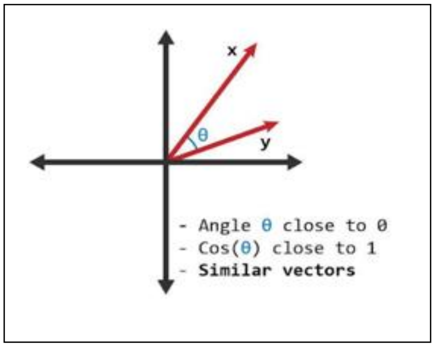


Fig 1: Similar vectors (Perfect)

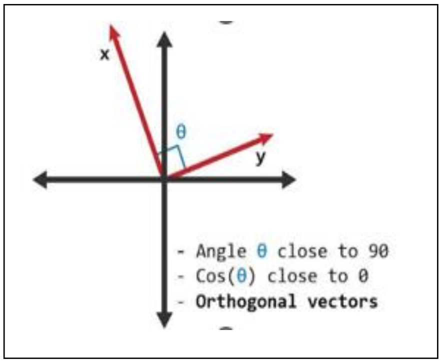


Fig 2: Orthogonal vectors (Neutral)

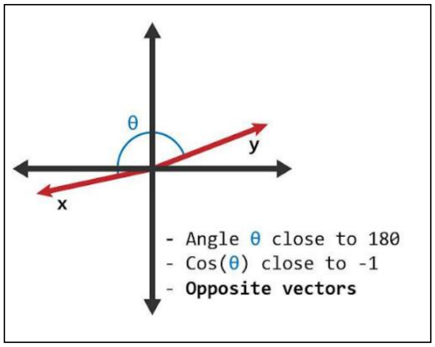


Fig 3: Opposite vectors (Contradiction)

Jaccard Similarity:

Jaccard similarity is a useful measure for finding text similarity between two sentences in certain situations, especially when the order of the words is not important and the focus is on the presence or absence of certain words.

Jaccard similarity measures the similarity between two sets by looking at the intersection and union of the elements in the sets. In the context of text similarity, the words in each sentence are treated as sets of tokens, and the Jaccard similarity score is calculated as the size of the intersection divided by the size of the union of the two sets.

Jaccard similarity can be useful in the following scenarios:

1. Keyword matching: If the goal is to identify documents or sentences that contain certain keywords, Jaccard similarity can be useful in identifying sentences that have a high degree of overlap in terms of the presence or absence of those keywords.

2. Short text analysis: When dealing with short texts, such as tweets or headlines, Jaccard similarity can be a useful measure to capture the similarity of the text.

3. Plagiarism detection: Jaccard similarity can be used to detect plagiarism by comparing the similarity between a given text and a set of reference documents.

However, Jaccard similarity has limitations in capturing the overall semantic similarity between two sentences. It doesn't consider the order, synonyms, or paraphrases of the words, and therefore it may not be the best choice for certain applications where semantic similarity is crucial. In such cases, other measures such as semantic similarity using pre-trained language models may be more appropriate.

Jaccard similarity may not always be the best choice for finding text similarity between two sentences for the following reasons:

1. Word order: Jaccard similarity only considers whether two sets contain the same items or not, and does not take into account the order or position of the items. This means that two sentences with the same words in a different order will have the same Jaccard similarity score, even if they have a different meaning.

2. Synonyms and paraphrases: Jaccard similarity is sensitive to exact word matches, which means that two sentences with similar meaning but different wording will have a low Jaccard similarity score. This can be a problem in natural language processing applications where it is important to capture semantic similarity, even if the words used are not exactly the same.

3. Stop words: Jaccard similarity treats all words equally, including stop words such as "the", "and", "a", etc. This means that stop words can have a disproportionate influence on the Jaccard similarity score, which may not be desirable for some applications.

4. Context: Jaccard similarity does not consider the context in which the sentences appear. This means that two sentences that have different meanings in different contexts may have a high Jaccard similarity score, even if they are not semantically similar in the given context.

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}$$

3.4 FRAMEWORK

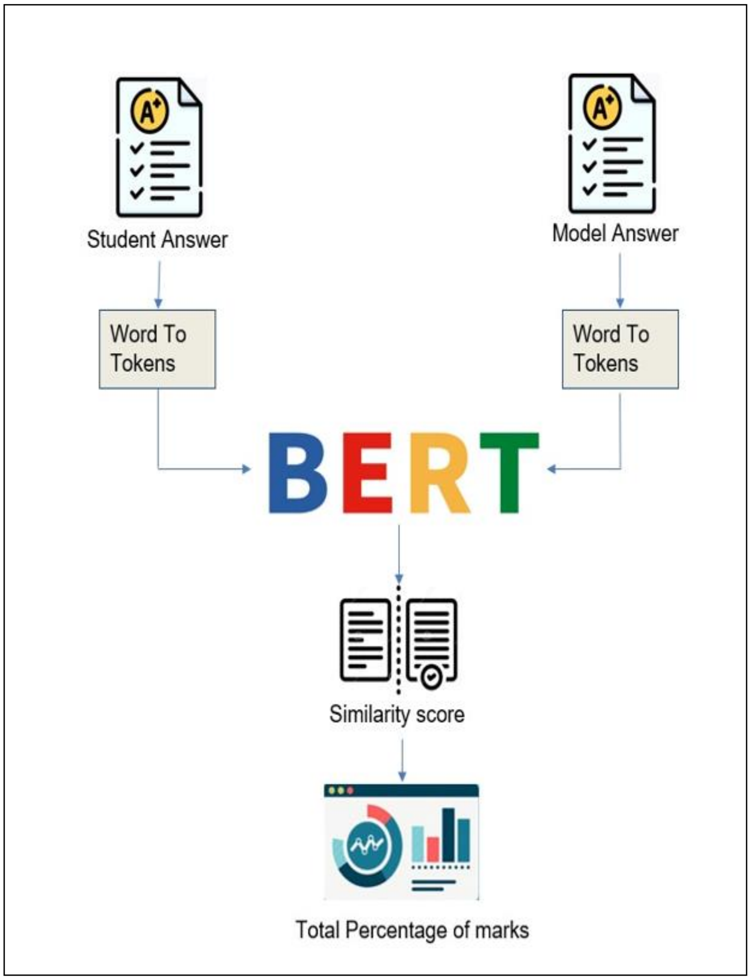


Fig 4: Design of CheckMyAnswer

Workflow of System:

- We take the model answer from the teacher.
- Then the student enters his or her answer in the student answer text box.
- Then both answers are tokenized using the Bert tokenizer and the Hugging Face transformer.
- After this tokenized answer, we transfer it to our Bert model to check the similarity and grammar of the student's answer.
- There is a similarity score based on three parameters: perfect, neutral, and contradiction.
- The final result is displayed in the "Total percentage of marks" format.

3.5 ALGORITHMS

Hugging Face Transformer:

Hugging Face is a company that develops and maintains an open-source software library called Transformers, which provides a simple interface for using state-of-the-art natural language processing (NLP) models such as BERT, GPT, RoBERTa, and more. The library is built on top of PyTorch and TensorFlow, and includes pre-trained models that can be fine-tuned for a wide range of NLP tasks such as text classification, question answering, sentiment analysis, and more.

Hugging Face is used for several reasons:

1. Pre-trained models: Hugging Face provides access to pre-trained NLP models that have been trained on large amounts of text data. These models can be fine-tuned on specific tasks and used to achieve state-of-the-art results with minimal training data.

2. Ease of use: The Transformers library provides a simple and consistent interface for using pre-trained models, making it easy for developers and researchers to experiment with different models and tasks.

3. Community support: Hugging Face has a large and active community of developers and researchers who contribute to the library and provide support to users.

4. Flexibility: The library is highly customizable, allowing users to fine-tune existing models, create their own models, or even use multiple models in combination to achieve better performance on specific tasks.

The transformer architecture is based on the idea of self-attention, which allows the model to weigh the importance of different parts of the input sequence when making predictions. Self-attention can be thought of as a mechanism for the model to focus on the parts of the input sequence that are most relevant to the task at hand.

The transformer architecture consists of an encoder and a decoder. The encoder takes an input sequence and produces a sequence of hidden states, which represent the encoded information in the input sequence. The decoder then takes this sequence of hidden states as input and generates an output sequence.

One key feature of the transformer architecture is the use of multi-head attention, which allows the model to attend to different parts of the input sequence simultaneously. This helps the model to capture more complex dependencies between different parts of the input sequence.

Another important aspect of the transformer architecture is the use of positional encoding, which allows the model to encode the order of the input sequence without using recurrent or convolutional layers.

The transformer architecture has been widely used in a variety of NLP tasks, including machine translation, sentiment analysis, and question answer checking.

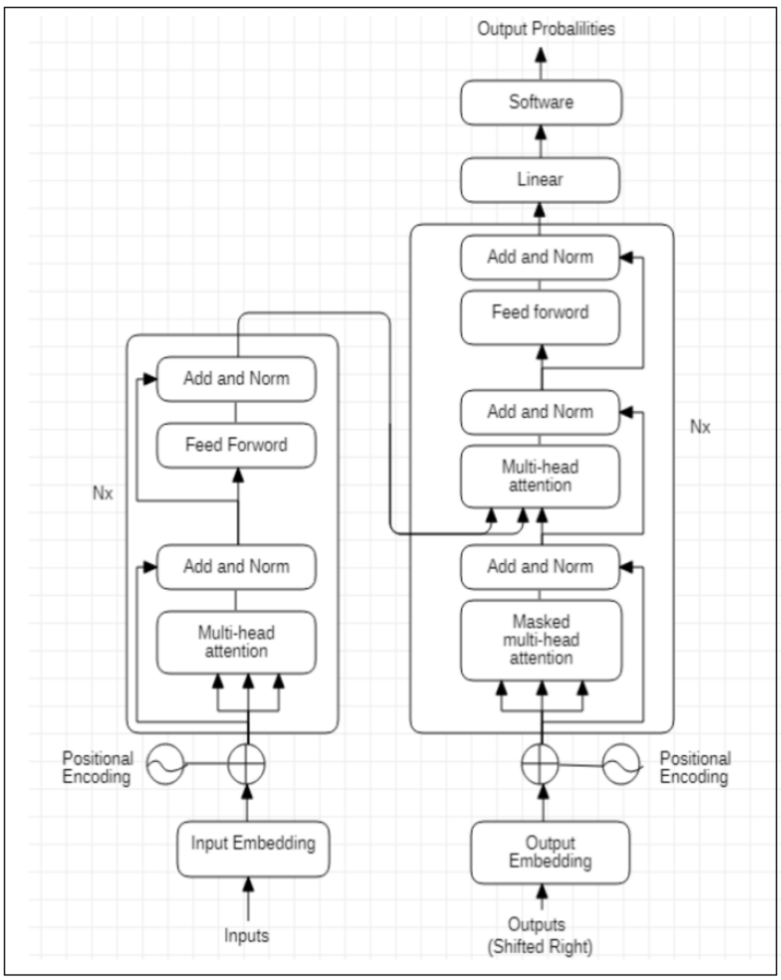


Fig 5: Transformer architecture

BERT:

BERT, or Bidirectional Encoder Representations from Transformers, is a language model developed by Google in 2018 that has revolutionized the field of natural language processing (NLP). It is a pre-trained neural network that can be fine-tuned for a wide range of NLP tasks, such as sentiment analysis, text classification, and question answering.

One of the key innovations of BERT is its ability to capture context-dependent meaning in language. This is achieved through the use of a bi-directional transformer, which allows the model to consider the entire input sequence when making predictions. By contrast, previous language models, such as the popular Word2Vec and GloVe models, only consider the context of each word in the sequence.

BERT is pre-trained on massive amounts of text data, such as Wikipedia and the Common Crawl corpus, using a task known as masked language modeling. In this task, the model is presented with a sentence with some of its words randomly masked, and it must predict the missing words based on the context of the sentence. This pre-training allows BERT to learn a rich representation of the meaning of words and phrases that can be fine-tuned for specific NLP tasks.

One of the key benefits of BERT is that it allows for transfer learning, where the pre-trained model can be fine-tuned for specific NLP tasks with relatively little training data. This is particularly useful for applications where labelled data is scarce or expensive to obtain. BERT has been shown to achieve state-of-the-art performance on a wide range of NLP tasks, including sentiment analysis, text classification, question answering, and named entity recognition.

3.6 DETAILS OF HARDWARE & SOFTWARE

Hardware Details:

CPU Processor: i3 10 generation

RAM: 4GB

Operating System: Linux, Windows, Mac

Graphics: NVIDIA GeForce GTX 1650

Operating System Architecture: 64 bits

Software Details:

Google Colaboratory

Navigator Anaconda version 3

Visual Studio Code version 2019

Libraries Details:

Python Version 3.9

Numpy

Pandas

Keras

Tensorflow

3.7 DESIGN DETAILS

Python can be used in various ways to develop CheckMyAnswer. Python's extensive library ecosystem provides developers with tools and frameworks to build end-to-end CheckMyAnswer systems, from natural language processing and machine learning models to web development and deployment. With libraries such as numpy, pandas, and nltk, Python offers powerful natural language processing and machine learning capabilities that can be leveraged to develop sophisticated algorithms for evaluating subjective answers. Additionally, Python frameworks such as Flask can be used to build web applications that provide a user-friendly interface for submitting and viewing answers. Python's versatility, coupled with its rich library ecosystem, makes it an excellent choice for developing CheckMyAnswer.

TensorFlow is an open-source library for building and training machine learning models, including those used in CheckMyAnswer. TensorFlow provides a high-level API, Keras, that allows developers to easily build and train deep learning models for a variety of tasks, including natural language processing. Using TensorFlow, developers can train models on large datasets of answers and their corresponding scores or probabilities and use those models to evaluate new answers submitted by users. TensorFlow's powerful computational abilities, combined with its ease of use and flexibility, make it a popular choice for building CheckMyAnswer, which can provide accurate and consistent feedback to users.

Flask is a Python web framework that can be used to build web applications, including CheckMyAnswer. Flask provides a lightweight and flexible structure for building web applications, allowing developers to focus on the logic of the answer-checking algorithm. Using Flask, CheckMyAnswer can be built as a web application where a user submits their answer through a web form. The Flask application processes the answer using a pre-trained model or algorithm and returns a score or probability for the correctness of the answer. Flask's ability to handle HTTP requests and responses makes it a powerful tool for building CheckMyAnswer, which can be accessed through a web browser.

Hugging Face Transformers is an open-source library that provides pre-trained models for natural language processing tasks such as text classification, question answering, and language translation. Using these pre-trained models, Hugging Face Transformers can be utilized as a CheckMyAnswer by comparing a student's response to a given prompt to the pre-existing knowledge within the model.

HTML and CSS can be used to build the front-end of CheckMyAnswer, providing a user-friendly interface for submitting and viewing answers. HTML provides the structure and content of the webpage, including forms for users to submit their answers, while CSS allows developers to style the page and create an intuitive layout for users to navigate. By combining HTML and CSS, developers can create a visually appealing and responsive web application for subjective answer checking. This front-end interface can be connected to back-end technologies such as Flask or TensorFlow to process and evaluate submitted answers, providing an end-to-end solution for subjective answer checking.

3.8 METHODOLOGY

Dataset:

The SNLI corpus [9] (version 1.0) is a collection of 570k human-written English sentence pairs manually labeled for balanced classification with the labels *entailment*, *contradiction*, and *neutral*, supporting the task of natural language inference (NLI), also known as recognizing textual entailment (RTE). We aim for it to serve both as a benchmark for evaluating representational systems for text, especially including those induced by representation learning methods, as well as a resource for developing NLP models of any kind.

Model Building:

Tokenize the input data: Convert the raw text input into numerical inputs that can be fed into the model. Hugging Face provides a tokenizer for each pre-trained model, which converts the text input into numerical input sequences.

Choose a pre-trained model: Hugging Face offers a range of pre-trained models for different NLP tasks such as question answering. Choose a pre-trained model that is appropriate for the task.

Model Training:

Fine-tune the model: Fine-tuning involves training the pre-trained model on specific task using given dataset. Hugging Face provides a range of utilities for fine-tuning the pre-trained models, including trainers, optimizers, and schedulers.

Evaluate the model: Evaluate the performance of the model on a validation set to determine how well it generalizes to new data. Hugging Face provides tools for evaluating the model's performance on a range of metrics, such as accuracy.

Model Deploying:

Once the model has been trained and evaluated, it can be deployed using HTML, CSS, JavaScript in front-end and flask in back-end. Hugging Face provides tools for deploying the model as a web application.

3.9 IMPLEMENTATION

Code snippet for main.py:

```
main.py
import numpy as np
from huggingface_hub import from_pretrained_keras

app = Flask(__name__)


class BertSemanticDataGenerator(tf.keras.utils.Sequence):
    """Generates batches of data."""

    def __init__(
            self,
            sentence_pairs,
            labels,
            batch_size=32,
            shuffle=True,
            include_targets=True,
    ):
        self.sentence_pairs = sentence_pairs
        self.labels = labels
        self.shuffle = shuffle
        self.batch_size = batch_size
        self.include_targets = include_targets
        # Load our BERT Tokenizer to encode the text.
```

Code snippet for Home page:

index.html

templates > index.html > style > div.container h1

```
            color: black;
            padding-top: 40px;
            font-weight: 400;
            padding-bottom: 80px;
            padding-right: 30px;
        }
    </style>

    <div class="container">
        <div class = "row">
        <div class="col">
            <center>
            <h1>CheckMyAnswer</h1>
            </center>
        </div>
        <div class="col">
            <center>
            <p><h2>Hii friends,</h2> I can help you evaluate subjective answers
                <br> based on certain criteria and guidelines provided to me. However,
                <br> it is important to note that subjective answers can be influence
                <br> by various factors such as personal biases, emotions, and cultural
                <br>background, which may not be fully understood by AI. Therefore,
                <br> while I can provide a level of objective analysis, I cannot replace
                <br> the value of human judgement and contextual understanding
                <br> in evaluating subjective answers.</p>
            </center>
        </div>
        </div>
    </div>
```

Code snippet for Layout page:

layout.html

templates > layout.html > html > body > nav.navbar.navbar-expand-lg.navbar-light.bg-light > div.container-fluid > div#navbarSupportedContent.colla

```
        }
    </style>

    {% block style %} {% endblock %}
</head>

<body>
    <!-- Navbar -->
    <nav class="navbar navbar-expand-lg navbar-light bg-light">
        <div class="container-fluid">
            <!-- Logo -->
            <a class="navbar-brand" href="/">CheckMyAnswer</a>
            <button class="navbar-toggler" type="button" data-bs-toggle="collapse"
                data-bs-target="#navbarSupportedContent" aria-controls="navbarSupportedContent" aria-expanded="false"
                aria-label="Toggle navigation">
                <span class="navbar-toggler-icon">Hey</span>
            </button>

            <div class="collapse navbar-collapse" id="navbarSupportedContent">
                <ul class="navbar-nav me-auto mb-2 mb-lg-0">
                    <!-- Home -->
                    <li class="nav-item">
                        <a class="nav-link active" aria-current="page" href="/">Home</a>
                    </li>

                    <!-- Student -->
                    <li class="nav-item">
                        <a class="nav-link" href="/student">Student</a>
                    </li>

                </ul>
            </div>
```

Code snippet for Student page:

```
student_home.html
templates > student_home.html > div.container > div.row > div.login-wrap > div.login-html > div.col-8 > div.row > div.lo
    <div class ="col-8">
      <h1>Asmit Patil</h1>

      <div class="row">
      <div class="login-wrap2">
      <div class="login-html2">

        <div class="col">
          <h2>Subject: <br>Machine Learning</h1>
          <h2>Teacher: <br>Pooja Bhagat</h2>
          <h2>Time: <br>11.00AM To 12.00PM</h2>
          <a href="/checkMyAnswer"><button type="submit" class="btn btn-primary">Exam</button></a>
        </div>

      </div>
      </div>
      </div>

    </div>

  </div>
  </div>
```

Code snippet for CheckMyAnswer page:

```
checkMyAnswer.html
templates > checkMyAnswer.html > html > body > div.container
        <center>
            <h1>CheckMyAnswer</h1>
        </center>
        <hr>
        <!-- Question & Model Answer -->
        <div class="row">
            <!-- For student & model answer -->
            <div class="col">
                <div class="login-wrap">
                    <div class="login-html">
                <form action="/predict" method="POST">
                    <div class="mb-3">
                        <label for="Question" class="form-label"> Question)  What is Machine Learning. </label>
                    </div>
                    <div class="mb-3">
                        <label for="Student answer" class="form-label">Student Answer:</label>
                        <textarea class="form-control" id="student_ans" name="student_ans" placeholder="Enter your text here.
                    </div>
                    <div class="mb-3">
                        <label for="Model answer" class="form-label">Model Answer:</label>
                        <textarea class="form-control" id="model_ans" name="model_ans" placeholder="Enter your text here..."
                    </div>
                    <div class="row">
                        <div class="col d-grid">
                            <input class="btn btn-outline-secondary" type="reset" value="reset">
                        </div>

                        <div class="col d-grid">
                            <!-- <input class="btn btn-outline-warning" type="submit" value="Submit"> -->
                            <button type="submit" class="btn btn-outline-warning" id="submit_msg">Submit</button>
                        </div>
                    </div>
                </form>
                </div>
                </div>
            </div>
        </div>
```

Code snippet for Answer page:

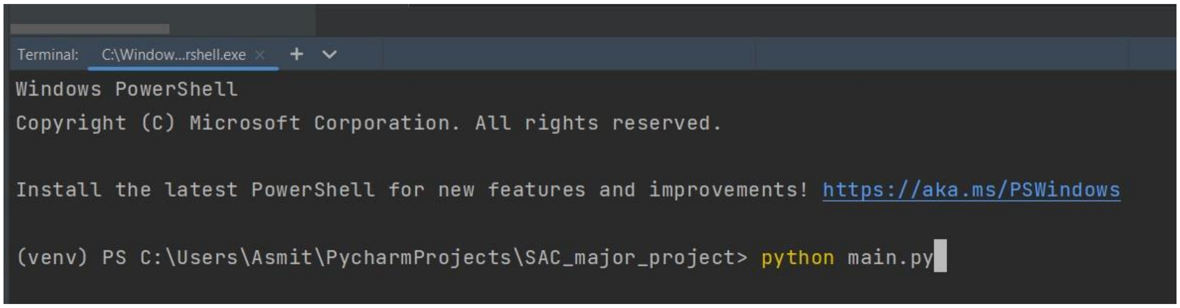
answers.html

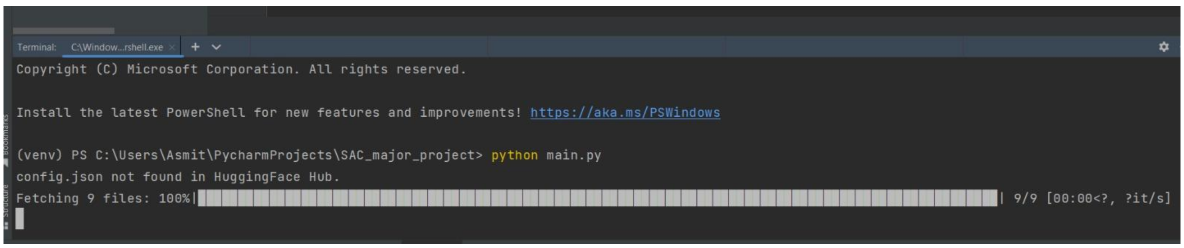
templates > answers.html > html > head > style > .login-wrap1

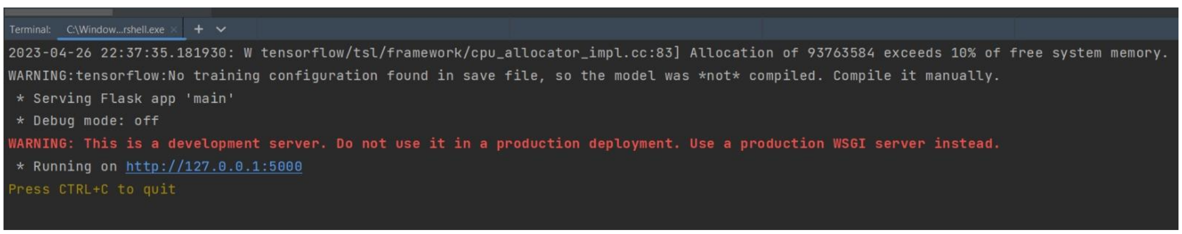
```
<div class="col">
    <div class="login-wrap1">
        <div class="login-html1">
    <center>
        <h2>Exam Score</h2>
    </center>
    <hr>
    <div class="row">
        <div class="col-9">
            <h4>Marks</h4>
                <button id="ms" onclick="myFunction()">Analysis</button>
                <div id="myDIV">
                    <div class="row">
                        <div class="col-9">
                            <h4>Perfect  Match</h4>
                            <h4>Semantic Match</h4>
                            <h4>Contradiction</h4>
                        </div>
                        <div class="col-3">
                            <h4>{{dict["Perfect"]}} %</h4>
                            <h4>{{dict["Neutral"]}} %</h4>
                            <h4>{{dict["Neutral"]}} %</h4>
                        </div>
```

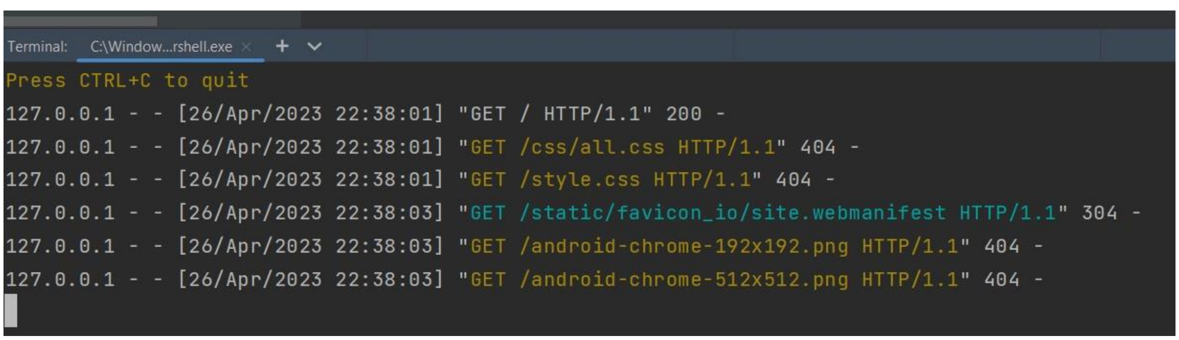
3.10 TESTING

Running Main.py:







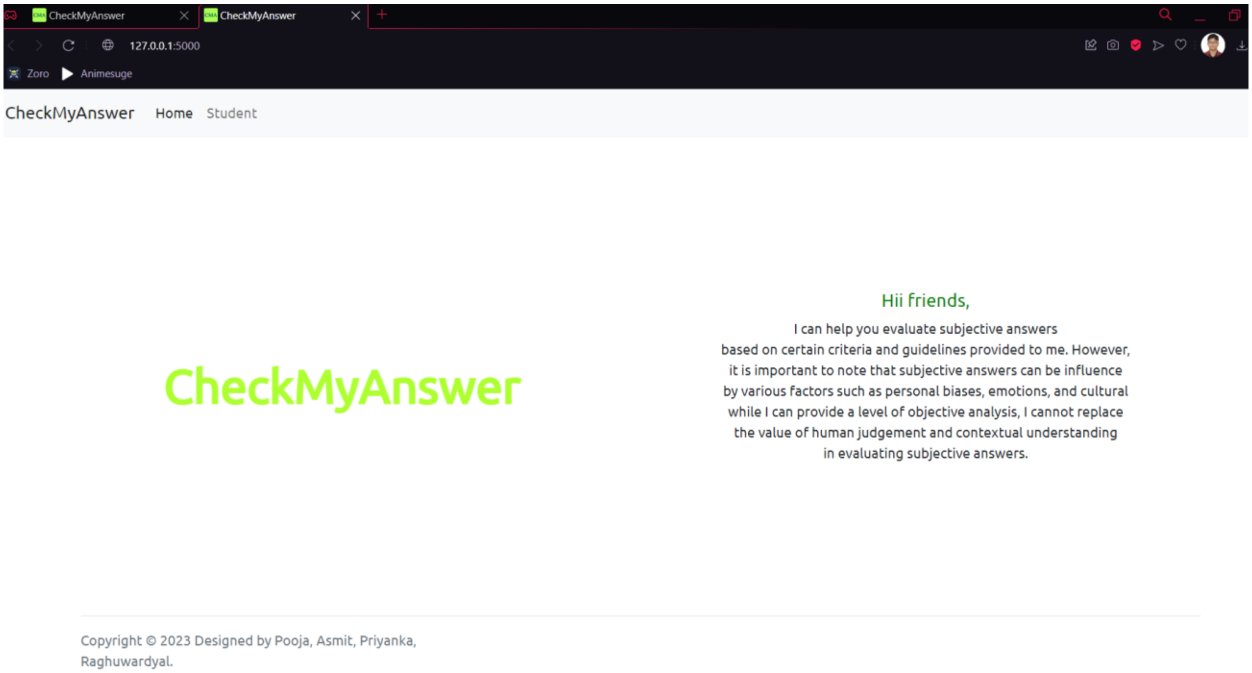


CHAPTER 4

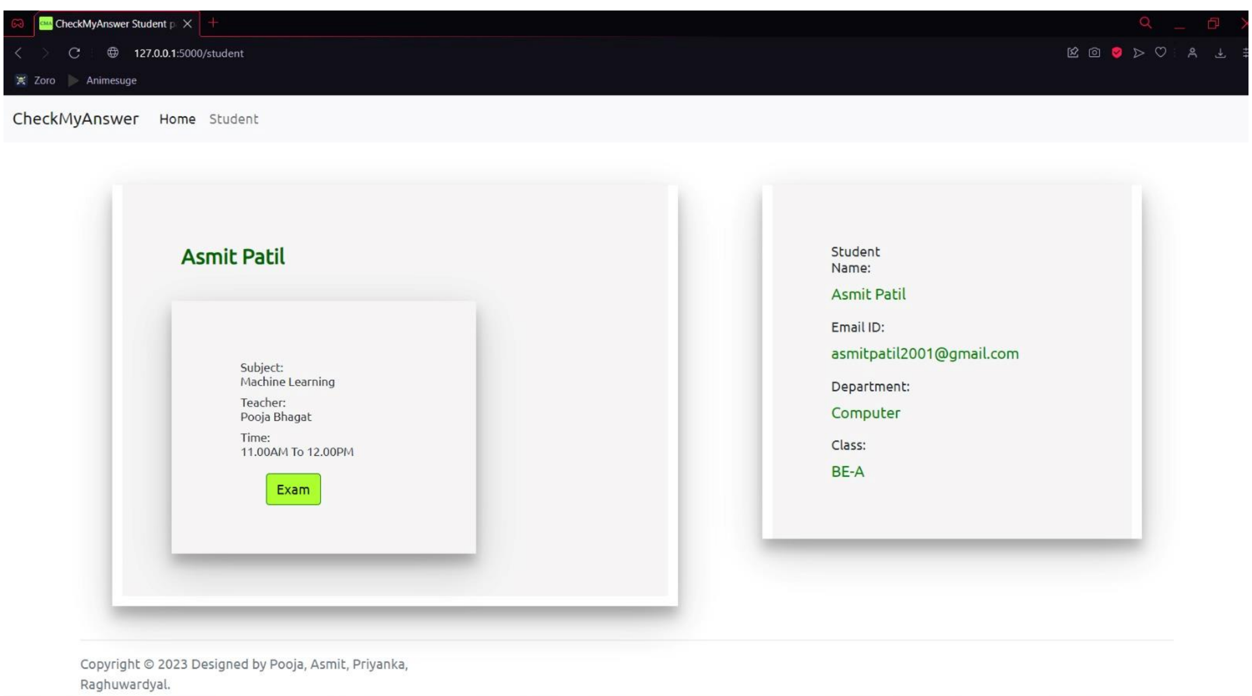
RESULTS

4.1 RESULTS

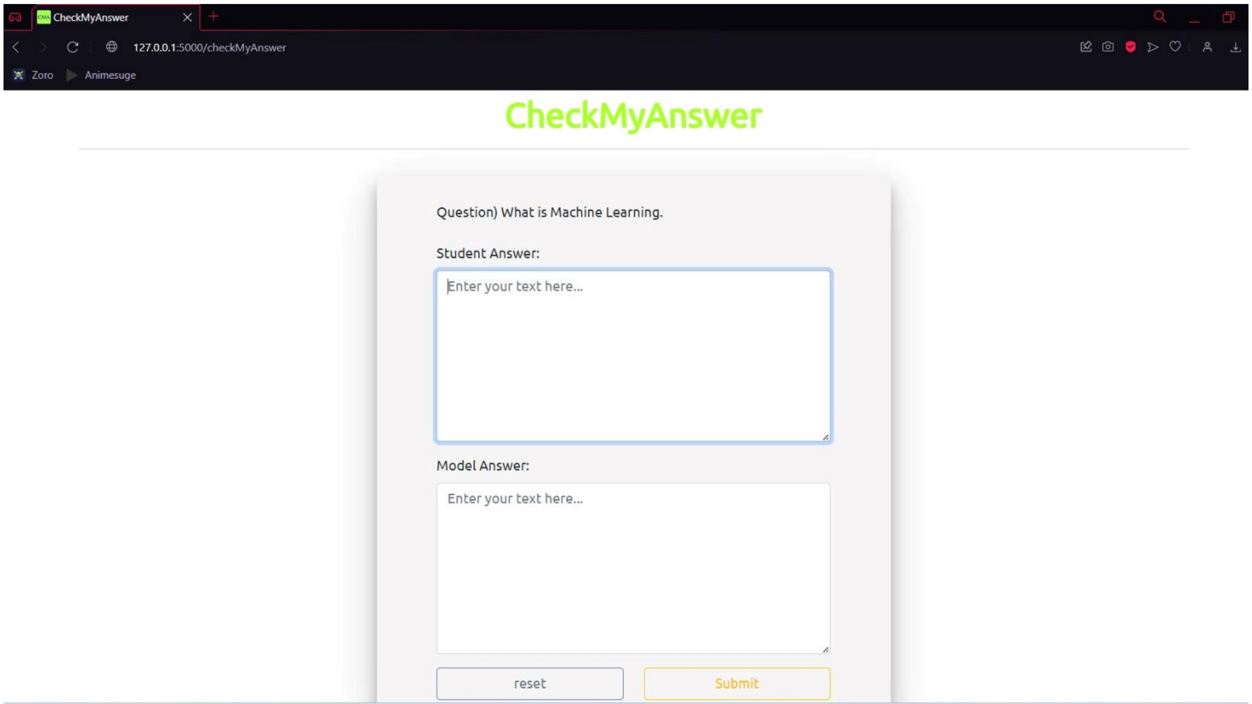
Home Page of Website:

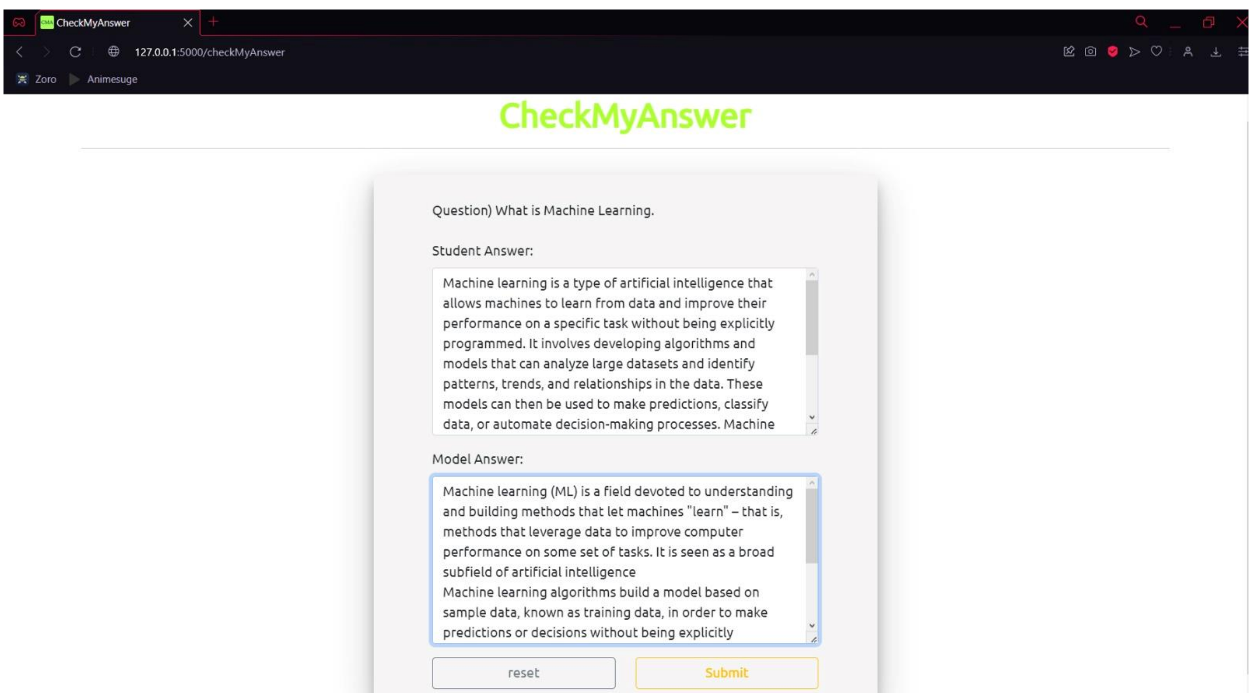


Student Page of Website:

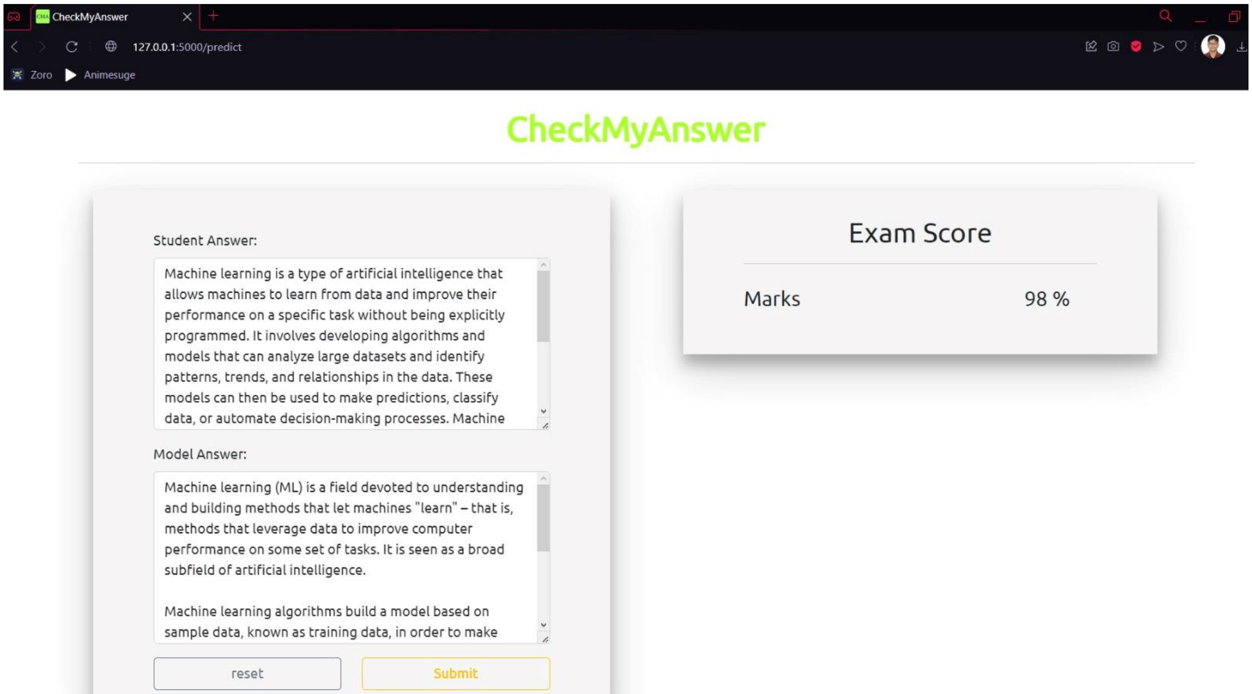


Exam Page of Website:

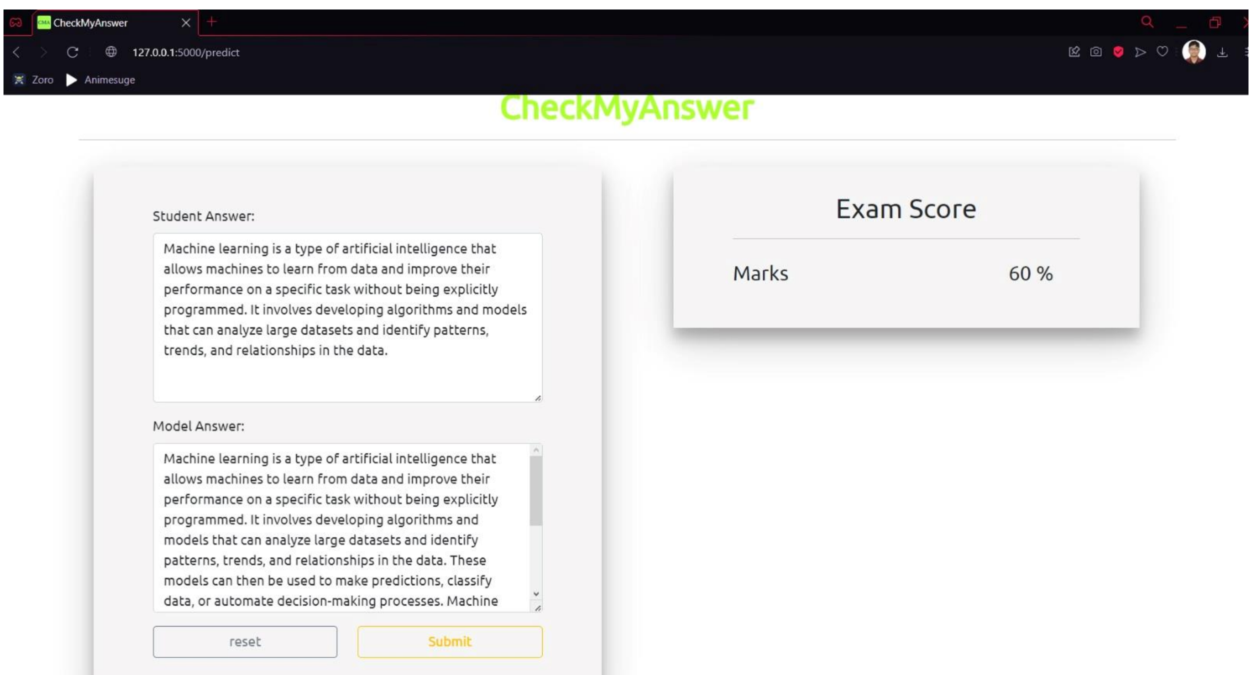




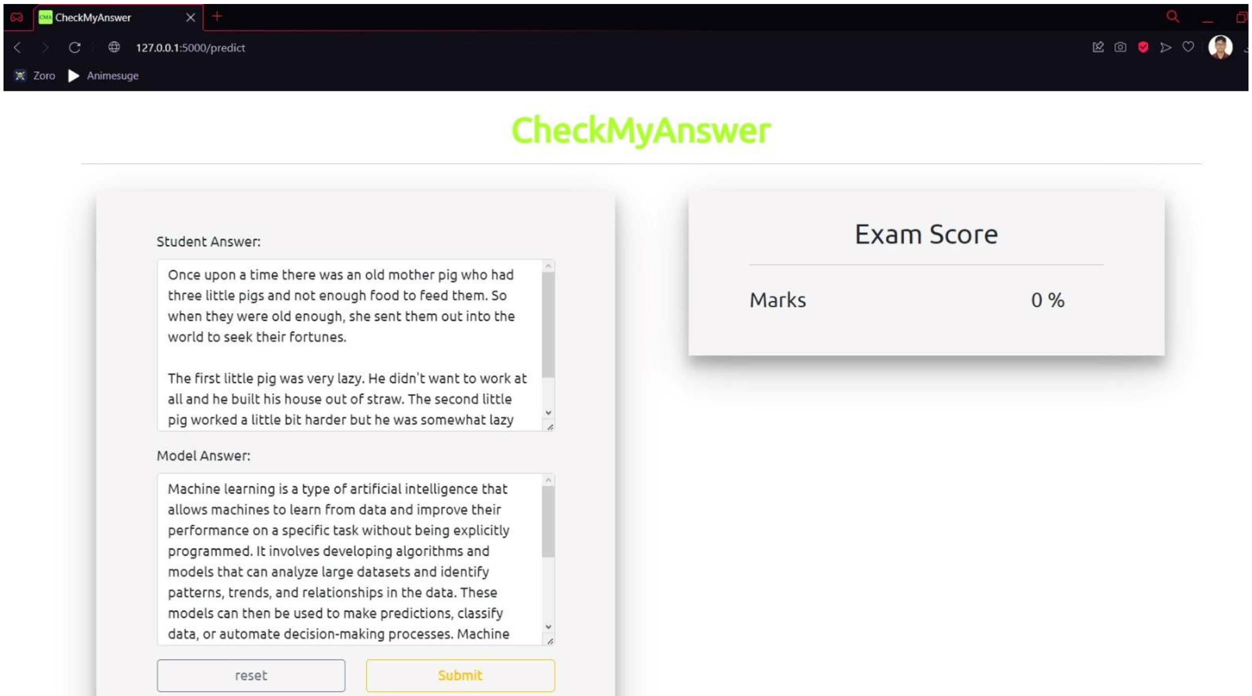
Result page of website: When answer by student and teacher is similar.



Results page of website: When answer by student is correct but wordings are different than teacher’s answer.



Results page of website: When answer by student is wrong and doesn't match to teacher answer.



CHAPTER 5

CONCLUSION

5.1 CONCLUSION

We have successfully implemented our project which is CheckMyAnswer. We have implemented website and provided a specific platform for subjective answer checking.

Universities always evaluate student’s performance based on their grades in coursework and final exams. Now, while a majority of the exam types is online multiple-choice questions, online testing machines are available to grade them. On the contrary, end semester exams are generally subjective, so there is a great demand for a system that can automatically score these brief answers without spending too much time.

Thus, our system provides a platform for all educational institutes and assigns accurate grades to the student's subjective answers. It aims to score student responses using the parameter like similarity index and words matching. The model solution provided by the teacher will be compared to the solution submitted by the student, and suitable grades will be assigned based on the above parameter. Such methods can be useful in many online evaluation platforms and college portals since they save time and hassle examining bundles of answer sheets.

5.2 FUTURE SCOPE

In the future we are planning to evaluate subjective answers by text extraction from image of student answer with diagrams and mathematical expressions. The current system only evaluates answers written in English. Further it can be extended to evaluate answers written in other languages also. Analysis of score can be seen examiner for creating analysis of results.

The current system gives only score so in future analysis of score which can be seen by examiner.

CHAPTER 6

REFERENCES

1. Johri, Era and Dedhia, Nidhi and Bohra, Kunal and Chandak, Prem and Adhikari, Hunain, ASSESS - Automated Subjective Answer Evaluation Using Semantic Learning (May 7, 2021). Proceedings of the 4th International Conference on Advances in Science & (ICAST2021).

2. G. Jagadamba and C. Shree G., "Online Subjective answer verifying system Using Artificial Intelligence," 2020 Fourth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC), Palladam, India, 2020, pp. 1023-1027.

3. "Automated Essay Grading Using Machine Learning" by A. Singh et al. (2015), which describes a machine learning approach to grading essays.

4. "Subjective Answers Evaluation Using Machine Learning and Natural Language Processing 2021 " by Muhammad Farrukh Bashir, Hamza Arshad, Abdul Rehman Javed Natalia Kryvinska And Shahab S. Band.

5. “Attention Is All You Need” by Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin.

6. “Ai Answer Verifier 2018” by Ashutosh Shinde, Nishit Nirbhavane, Sharda Mahajan, Vikas Katkar, Supriya Chaudhary.

7. “Tool for Evaluating Subjective Answers using AI 2021 (TESA)” by Shreya Singh, Prof. Uday Rote, Omkar Manchekar, Prof. Sheetal Jagtap, Ambar Patwardha, Dr. Hariram Chavan.

8. “Automatic AnswerSheet Checker 2022” by Ronika Shrestha, Raj Gupta and Priya Kumari.

9. https://www.kaggle.com/datasets/stanfordu/stanford-natural-language-inference-corpus

ANNEXURE I

Project Presentation for NAAC Committee

Poster: