A Project Report On

Automated Evaluation of Answer Scripts Platform

Submitted in partial fulfillment of requirements for 8th Semester

Bachelor Of Engineering

in

Computer Science and Engineering

Submitted By

AASHKA VIJAPURA	**1DS16CS002**
ITISHA PAUL	**1DS16CS038**
JOSHUA PAUL GOMES	**1DS16CS040**
LEENA HAORONGBAM	**1DS16CS048**

Under The Guidance Of

Dr. Vindhya P Malagi
Professor
Dept. of CSE, DSCE
Bangalore-78

Mrs. Sarala D.V
Assistant Professor
Dept of CSE, DSCE
Bangalore-78

2019-2020

Department of Computer Science and Engineering

DAYANANDA SAGAR COLLEGE OF ENGINEERING

BANGALORE - 560078

VISVESVARAYA TECHNOLOGICAL UNIVERSITY
DAYANANDA SAGAR COLLEGE OF ENGINEERING
Shavige Malleshwara Hills, Kumaraswamy Layout, Bangalore - 560078
Department of Computer Science and Engineering

CERTIFICATE

This is to certify that the project entitled **"Automated Evaluation of Answer Scripts Platform"** is a bonafide work carried out by AASHKA VIJAPURA[1DS16CS002], ITISHA PAUL[1DS16CS038], JOSHUA PAUL GOMES [1DS16CS040], LEENA HAORONGBAM[1DS16CS048] in partial fulfilment of 8th semester, Bachelor of Engineering in Computer Science and Engineering under Visvesvaraya Technological University, Belgaum during the year 2019-20.

Mrs. Sarala D.V	**Dr. Ramesh Babu**	**Dr. C P S Prakash**
(Internal Guide)	Vice Principal & Head of	Principal,
Assistant Professor,	Department, CSE,	DSCE
Dept. of CSE, DSCE	DSCE	

Name of the Examiners:	Signature with date:
1.....	.....
2.....	.....

Acknowledgements

We express immense pleasure in successful completion of this project on **"Automated Evaluation of Answer Scripts Platform."**

We would like to take this opportunity to express our gratitude to **Dr. C P S Prakash**, Principal of DSCE, for permitting us to utilize all the necessary facilities of the institution.

We are also very grateful to our respected Vice Principal, HOD of Computer Science and Engineering, DSCE, Bangalore, **Dr. Ramesh Babu D R**, for his support and encouragement.

We are immensely grateful to our respected and learned guide, **Mrs. Sarala D.V**, Assistant Professor, CSE, DSCE for their valuable help and guidance. We are extremely thankful to everyone for all the encouragement and guidance they have given us during every stage of the project.

We would like to thank our project co-ordinators **Dr. Vindhya M.**, Associate Professor, CSE, DSCE for her guidance and support.

We are also thankful to all the other faculty and staff members of our department for their kind co-operation and help.

Lastly, we would like to express our deep appreciation towards our classmates and our family for providing us with constant moral support and encouragement.

AASHKA VIJAPURA [1DS16CS002]
ITISHA PAUL [1DS16CS038]
JOSHUA PAUL GOMES[1DS16CS040]
LEENA HAORONGBAM [1DS16CS048]

Contents

List of Figures

1

Introduction

1.1 Automatic Evaluation

Automatic evaluation of subjective answer script requires Natural Language Processing for assessment of answer scripts. Ontology, Statistical method and Semantic similarity matching are various methods used. The purpose of providing education is to make student learn a selected topic or domain, in order that the scholar is in a position to use those knowledge and knowledge within the practical field. This will be possible as long as the scholar is in a position to understand it properly. Evaluation of student is important to know how much student has learned and understood. It's important to gauge what proportion of knowledge has been absorbed by the scholar. For this, one has got to determine the degree of learning of a student by conducting some written test of specific pattern which can include descriptive or objective questions or through some practical examination and evaluating it to seek out the degree of learning. Evaluation of objective answers are comparatively easy to descriptive answers and are also well supported in many systems, whereas descriptive answers can be an open problem. Evaluation work is extremely cumbersome as far as descriptive answer is concerned.
Automatic Evaluation deals with deals with correction of exam papers without human involvement and it uses different algorithms to score the answer given by the student. The questions pattern can be subjective or objective. Automatic answer evaluation of objective questions is less complicated as compared to subjective answers and have high accuracy, but in our project, we focus on subjective answer evaluation. Paper evaluation and then declaration of result can be a long step by step tedious process which can be erroneous due to human effort factor. Manual correction of answer sheet can consume a large and significant amount of evaluators' valuable precious time and can be an expensive process. Different security concerns regarding paper leakage is one among the various challenges to overcome.
In this project we build an automated examination evaluation system using machine learning, natural language toolkit (NLTK), python environment, flask framework and web technologies. This project aims to provide an easy to use and alternative to current examination system. In this project we focus to design algorithm for automation of evaluation process of descriptive answer.

This project ensure uniform evaluation of answer script , which aims to reduce errors and time spend on paper evaluation.
At present time paper correction and declaration of result in restricted time frame is becoming difficult to achieve for educational institutions like school, universities etc. To make the task easier and more accurate we propose a software for accurate automatic exam paper evaluation and grading system, the system evaluates the paper based on matching the key terms between answer script given by student and the correct answer to which it is to be compared.
Academic institutions generally have descriptive question answer pattern for assessment of the scholars. It requires tons of time to correct the paper by the evaluator and as the volume increases, it becomes time-consuming and there's no alternative solution for this. This reason requires the evaluation platform to be automated by the method that is accurate and efficient enough to precisely correct the answers given by students in their own words. Here we are taking answers typed by the student and then compare the answer to correct answer generate by the system for the question from a paragraph passed as input and then score is awarded. The scores are then saved. The system also generates question and answer pair from a text file that it takes as input. The appropriate approach in NLP that we will use shall help us deal with a different task that is good in its word and phrases selected from the correct answers and should also omit the probability of selecting the terms not related to the questions. The comparison also focuses on the selection of approaches which are efficient in predicting the synonyms of the actual keyword and also providing antonyms for them in order to figure out wrong answers given by students.

1.2 Natural Language Processing

Natural Language Processing (NLP), can be defined as automatic manipulation of Natural Language, like speech and text by software. Natural Language Processing has been there for over 50 years and with the rise of computers it grew out in the field of linguistics. The field of natural language processing deals with human language like speech and text and the concept is very interesting, the original value of this technology comes from use cases.
NLP can help you with lots of tasks and the fields of application just seem to increase on a daily basis, like:

- Electronic health record's and patient's own speech is used by NLP for prediction and recognition of diseases. Health conditions like cardiovascular diseases, depression and schizophrenia are being explored with this capability. For example,

- Amazon Comprehend Medical service uses NLP for disease condition extraction, medication and treatment outcome from patient notes, clinical trial reports and other electronic health records.

- Amazon's Alexa and Apple's Siri are some of the intelligent voice driven interfaces that uses NLP to respond to vocal prompts and do everything like tell us the weather forecast, suggest the best route to the school and turn off the lights at home.

- NLP is additionally getting used in both the search and selection phases of talent recruitment, identifying the talents of potential hires and also spotting prospects before they become active on the work market.

Syntax and semantic analysis are two major techniques that are used for natural language processing. Syntax refers to arrangement to words in a sentence that are grammatically correct. NLP uses syntax analysis to extract meaning from a sentence using various grammatical rules. Syntax technique use parsing that is grammatical analysis of a sentence, word segmentation which divides large pieces of texts into smaller units, sentence breaking which places sentence boundaries in large texts, morphological segmentation which divides words into groups and stemming which divides words with inflection in them to root forms. Semantic analysis involves analysis of the words used and meaning behind the words. NLP applies various algorithms to understand the meaning and structure of sentences. most frequently used algorithms in NLP when defining the vocabulary of terms:

Bag of Words: This model is commonly used to count all the words that are used in a text. It creates an occurrence matrix of words in a sentence or document, disregarding the grammar and word order. The classifier is then trained using these word frequencies and occurrences.

Tokenization: It is the process of segmenting running text into sentences and words. Its task is to diving text into pieces where each individual piece is called token and at the same time neglecting certain characters, such as

punctuation.

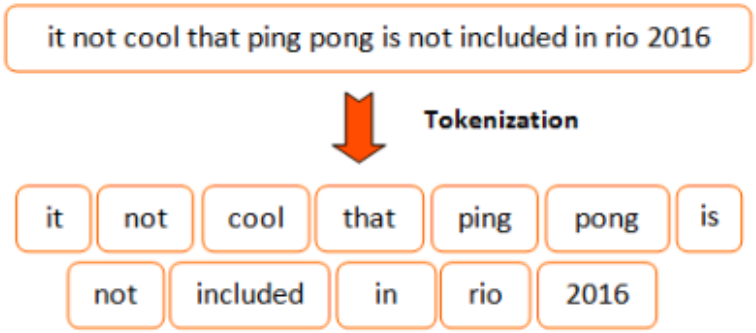

Figure 1.1: Tokenization

Stop Words Removal: It includes getting obviate common language articles, pronouns and prepositions like “and”, “the” or “to” in English. During this process some quite common words that appear to supply little or no value to the NLP objective are filtered and excluded from the text to be processed, hence removing widespread and frequent terms that aren’t informative about the corresponding text. there’s no universal list of stop words. These are often pre-selected or built from scratch. A possible approach is to start by adopting pre-defined stop words and add words to the list afterward.

Stemming: Stemming is the process of slicing the beginning and end of words with the motive of removing affixes. This is done because affixes can create or expand new forms of same word (inflectional affixes), or even create new words (derivational affixes).

Lemmatization: It has the objective of reducing a word to its base form and grouping together different forms of the same word.

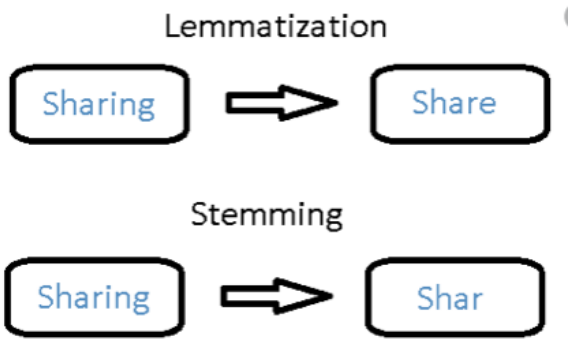

Figure 1.2: Tokenization

Topic Modelling: It is a way for uncovering hidden structures in sets of texts or documents. In essence its clusters texts to get latent topics supported their contents, processing individual words and assigning them values supported their distribution. This system is predicated on the assumptions that every document consists of a mix of topics which each topic consists of a group of words, which suggests that if we will spot these hidden topics, we will unlock the meaning of our texts.

1.3 Organisation of Project Report

The project report is organized as follows:

In Chapter (2), we will discuss about the problem statement and our solution to the problem. The Chapter that follows i.e chapter (3) consists of the details on the literature survey of the papers to the problem statement and the proposed solution. In Chapter (4), we present the System Overview and Proposed system in the form of Data flow diagram and the sequence diagram. The next chapter, chapter (5)gives the requirements and detains about the implementation of the proposed system. Chapter 6 deals with the testing of the system and their results. The chapter (7) concludes the paper along with mention of the Future Enhancements. Chapter (8) is details about the references made during the development of the system. The other supporting information and the source code are gathered in the Appendix.

2

Problem Statement and Proposed Solution

2.1 Problem Statement

To develop a system that is capable of evaluating subjective answer scripts which are typed by the student with high accuracy and also score the given answer. This software aims in reducing the task of the teacher by providing an automatic testing platform where the answers given by the student are marked based on the correct answer fed in the system. System generates both the question and associated answer and is built to form and evaluated questions.

2.2 Proposed Solution

This project is implemented using Natural Language Processing for automatic evaluation of answer scripts. The questions and answers will be generated by the system. Students will login to the system. There will be two options where the students will want to prefer, i.e. Objective exam and Subjective exam. If the student prefers Subjective exam, then the details will be stored in the subjective file. Then questions will be displayed on the screen and the students have to write the answers. The answers written by the student will be matched with original subjective question and answer sets which are already there in the system. And the system will generate the score automatically. Meanwhile, in Objective exam, the details of the students will be stored in objective file. Questions for the respective subject will be displayed on the screen and an entry box is given below each questions where the students have to answer the question. After submitting the answers, final scores will be displayed.

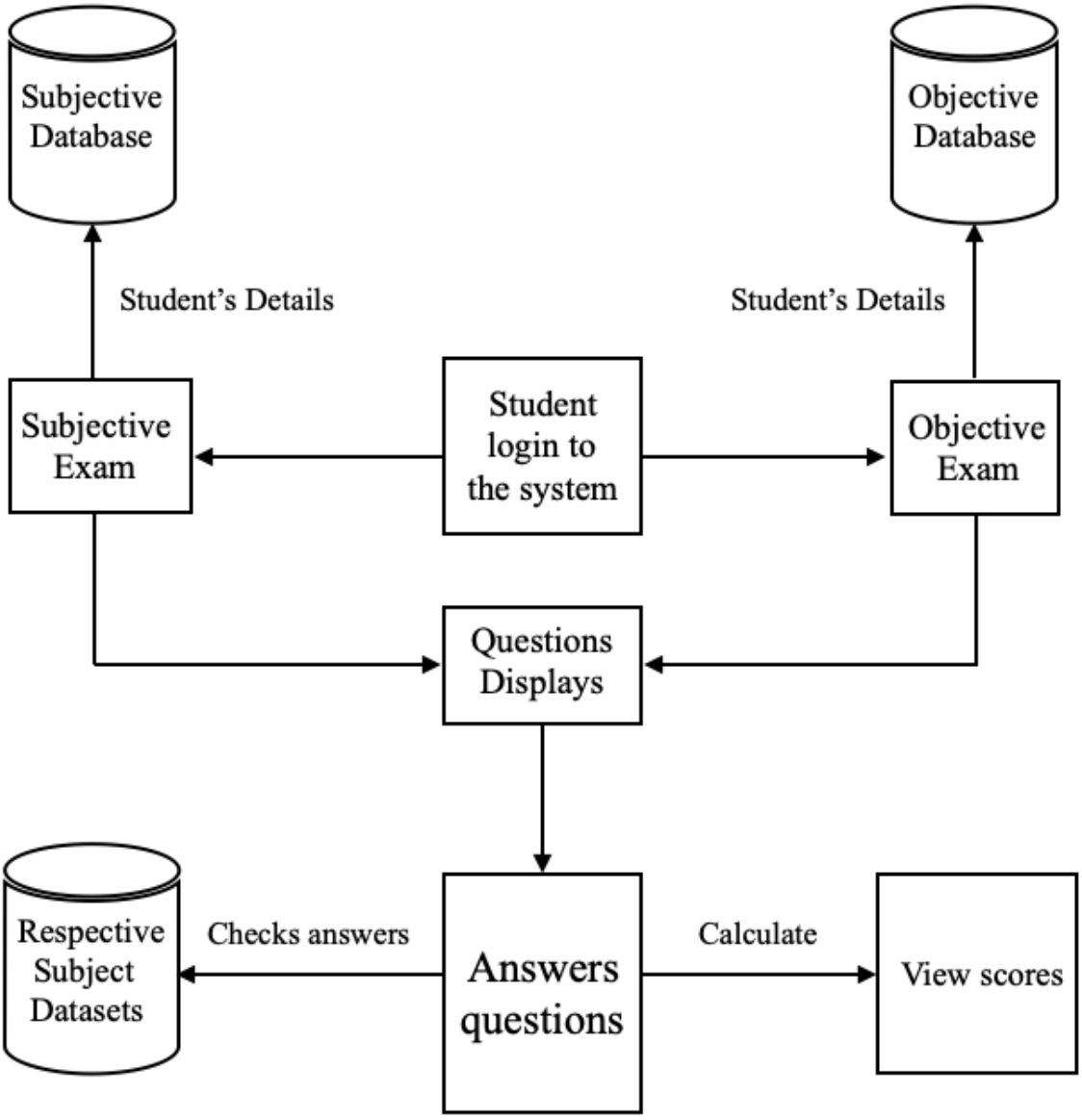

Figure 2.1: Block Diagram of the Proposed Solution

2.3 System Requirements

The system requirements for the proposed system is represented using the Use Case Diagram in Fig. 2.2

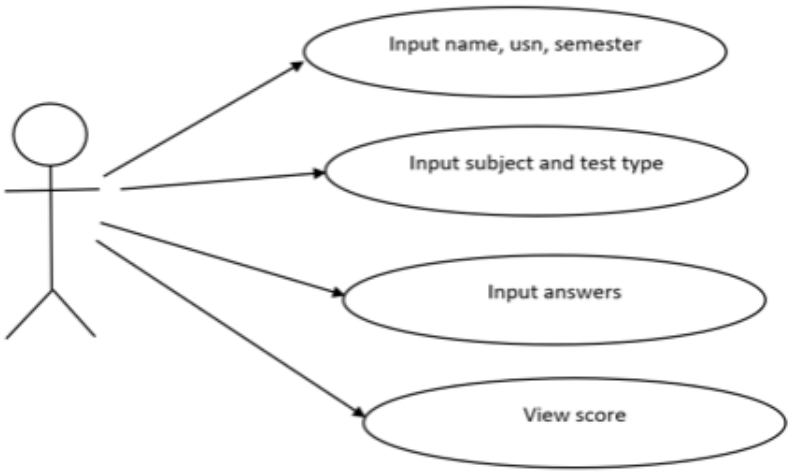

Figure 2.2: Use Case Diagram

The system should be able to:

1. Take input answer from student.
2. Generate questions and associated answers.
3. Perform different NLP algorithm on the answer submitted.
4. Compare the answer given by student to the correct answer.
5. Evaluate the answer with high accuracy.
6. Display output score awarded.

3

Literature Survey

1. **A Scoring Tool for Electronic Paper Exams, in IEEE, 2007.**
   Authors: Narcis Lozano, Koichi Hirosawa and Masaki Nakagawa.

   Description: The paper presents a scoring tool for electronic paper exams. The system automatically recognizes questions and scores them. The answer written on electronic pen devices. Although there are test scoring systems for scanned images, they are often limited to selection questions. Due to the inherent difficulty of segmenting foreground and background in images, they usually require the answers to be fulfilled in a separated sheet of paper. The scoring system takes a hybrid approach where it can have automatic scoring for selection type questions or questions for which answers are limited and human assisted scoring for questions that require to write free-text answer. The teachers first prepare the exams using a text processor and print it. Finally, they save the template or templates in the server. Then, they prepare copies of the exam paper, which are distributed to the students. The students fulfil the exam using the pen and paper device. They first write the exam identifier, which is given to them by the teacher, in the special area reserved for this purpose. The students then start to answer the questions. When they finish, they store their answer files to the server and the paper exams written by the students are also gathered. Server identifies exam using identifier and receives corresponding template. For automatic scoring questions, ink is recognized and compared with the correct answer and evaluated. The rest of the questions are kept in the InkML format. Teacher then loads the answer files from serves, and answers that need manual correction are corrected.

2. **Indonesian Question Answering System for Solving Arithmetic Word Problems on Intelligent Humanoid Robot, 2018**
   Authors: Alexander A S Gunawan, Pribadi R Mulyono and Widodo Budiharto

   Description: Humanoid robots can be made an important part of educational system. These robots with human like appearance adds another

dimension altogether for interaction with humans. Humans can interact with the robot in their natural language and will get the reply from the robot. These robots can be effectively used in the education where they can be used as a guide for every student, a student may ask a question to the robot in his regular language and the robot will convert it into English text and process it and give the answer in return to the student in the language they asked question in. The challenge here is that by including AI in the robots they have to make them interact with humans in natural language. They have used a previously made robot RAPIRO which is a small robot designed to work with a Raspberry Pi. However, the main focus is kept on the answering capability of robot using NLP and Pattern Matching. Artificial Intelligence has many applications, one of which is Question Answering (QA) System, this system finds an answer to the question given as input to it in a natural language spoken by the user. The dataset here should be in text form for the QA system to process it and hence in the research done here the human will dictate the word problem to the robot and the Google Speech to Text will convert the question to text form , further Google translate will translate the question in Bahasa Indonesia language to English which will be processed using NLP and Pattern Matching to find the solution by forming various equations using predefined list of pattern schema and finally the answer will again be converted back to the Bahasa language and back to speech form and will be given as an answer to the human who asked the word problem.

3. **Textual Question Answering for Semantic Parsing in Natural Language Processing**
Authors: Jaydeb Sarker, Mustain Billah and Md. Al Mamun

Description: Question answering may be a difficult task which needs the understanding of the meaning of a text. For building such system, it's necessary to know the question and input text. Query Regression Network (QRN) approach, a neighborhood of (RNN), is in a position to effectively handle such scenarios. This method is additionally used for machine comprehension. Todai robot has learned the exam answers and test that from the scholar answers. they need provided a way to seek out the solution to given questions by a general searching algorithm. Their method will first analyses the input text and questions and find the facts. It finds

the solution from the facts. During this paper, an approach of semantic parsing of texts is introduced. The proposed approach within the paper uses the tactic supported lambda calculus for semantic parsing to realize logical sorts of sentences. The questions are going to be analyzed the system by collecting significant features to seek out correct answers from the facts.

4. **Intelligent Question Answering System based on Artificial Neural Network**

   Description: The proposed question answering system (QAS) uses deep cases along, with ANN to know the contents of the documents. It divides the sentences of tongue into knowledge units and assign deep case to every word to enhance the standard of data extraction. QAS is employed for fast information extraction by exempting user to read unnecessary information which could not cause the solution . This paper proposes a way to make a deep neural network from the documents provide by the user and storing them for future use. It tries to imitate human information recalling feature by processing the document first. It processes the question asked by the user and comprehending the question and understand what answer is required then attempt to find the solution from the deep neural network created previously from the documents provided. An application has been designed to implement the proposed algorithm to answer questions using internet beans IDE. The user has got to upload a document. After uploading the document, the system adds the knowledge into its database. The following steps describe all the operation that will be performed in each of the phase of the system.

   (a) Here, we use the “Input Data” or the textual information from the user.

   (b) Here, we divide the sentence given into words and allocate a special ID to the word.

   (c) Now, we extract the “knowledge units” from the “Input Data”Step

   (d) Here, we allocate a type to each and every word.Step

(e) Now, we allocate "deep cases" to every word.

(f) Here, we explain the connection for "word" and "knowledge units"

(g) At last, we produce a network simply constructed from the sentence.

(h) This network contains "knowledge". It can therefore answer questions upon using this knowledge.

(i) Here, we look for words from this network and take out the "knowledge unit" which connects with the words.

(j) Here the output is given using the knowledge unit.

The conclusion to this paper is that, we use knowledge units and neural networks to fasten our information extraction and improve the ability to answer the questions using fields such as "ANN" and "NLP".

5. **Student's Academic Performance Evaluation Method Using Fuzzy Logic System**
Authors: Abdul Aziz, Md. Asaf-uddowla Golap and M. M. A. Hashem

Description: Fuzzy Inference System evaluates the student's attendances, time spent in the class, marks scored in various assessment tests. Each of these have different weightage where the system will evaluate a student's performance. Fuzzy Logic deals with incorrect, fallacious and mistaken data, where the fuzzy controller will take inputs, evaluate and provides an output. Fuzzification is a process of converting crisp input values into fuzzy values using fuzzy membership functions. Defuzzification is the reverse process of fuzzification which converts this latter output into crisp values again. There exist a grading system such as A+, B+, C+ etc where each grades has an integer interval value . So, if a students secures 90, and another secure 100, both will be graded as A+, which is unfair for a student, so the FIS will provide decency, equity, equitableness, where each single mark will be considered. Here the system will be divided into two parts, Student's continuous assessment and Semester final examination. The dataset for Student's continuous assessment will be number of class, time spent in the class and class test, while the dataset for Semester final examination will be

Part A and Part B. In the fuzzification process of Student's continuous assessment, three linguistic variables are used #Class(Total Number of class), Time(Total time spent in the Class) and Class_Test. The linguistic values and interval for the variables will be obtained and a trapezoidal member function is used for the input variables. The output variable for this part is assessment whose linguistic value is as same as input variable. In the fuzzification process of Semester final examination, two linguistic variables are used Part-A and Part -B. The output variable for this part is assessment whose linguistic value is as same as input variable.

6. **Computerized Paper Evaluation Using Neural Network**
   Authors: Tanupriya Choudhury, Kartikeya Jain, Lakshya Aggarwal, Ayushi Gupta and Garv Saxena

   Description: The traditional way of evaluating includes answers written by the scholars are manually corrected or evaluated by the school . this might end in several errors. There are reasons why manual corrections aren't appreciated in today's generation. the school are often bias towards the solution scripts, thanks to lack of your time the teachers tries to end evaluating maximum number of papers during a short period of your time . Sometimes, the school awards marks with students who has good handwriting but not enough knowledge quite students who has more knowledge but not good hand writing. And moreover, the manual correction takes far more time and therefore the results of the scholars also will be delayed. A neural network may be a computational model that's supported the neural network within the citizenry . A neural network generally solves problems by learning and training. A neural network id trained to form the info entered in order that it could provide the expected output. the primary and therefore the easiest sort of ANN is that the feed forward network. during this network, the info stream flows just in one direction i.e. forward direction from input point by means of concealed points to the yield point. A Recurrent Neural Network (RNN) may be a time working neural network. It confirms the related input vector then refreshes its hidden states (if any) through non-direct commands and uses it to shape forecast on output. A Modular Neural Network (MNN) consists of variety of modules, each module finishes up in one sub assigned value of the neural system's

reality conditions, and every module is functionally embedded. Kohonen self-organizing maps measure a kind of neural system. They require no administration and subsequently referred to as Self Organizing. The system has 3 steps: Image Processing, Training and Detection. In image processing the image of the acceptable response contents to the framework and therefore the framework preprocesses the image through different preprocessing systems which are grayscale, obscure, edge and diminishing of the image . The Training of the proposed framework is completed utilizing the synthetic Neural Network. Preparing is finished with the help of simulated neurons. The characters identified by the system are matched with the characters within the database and therefore the performance of the scholar is calculated by each of the right answer.

4

Architecture and Design

4.1 System Overview

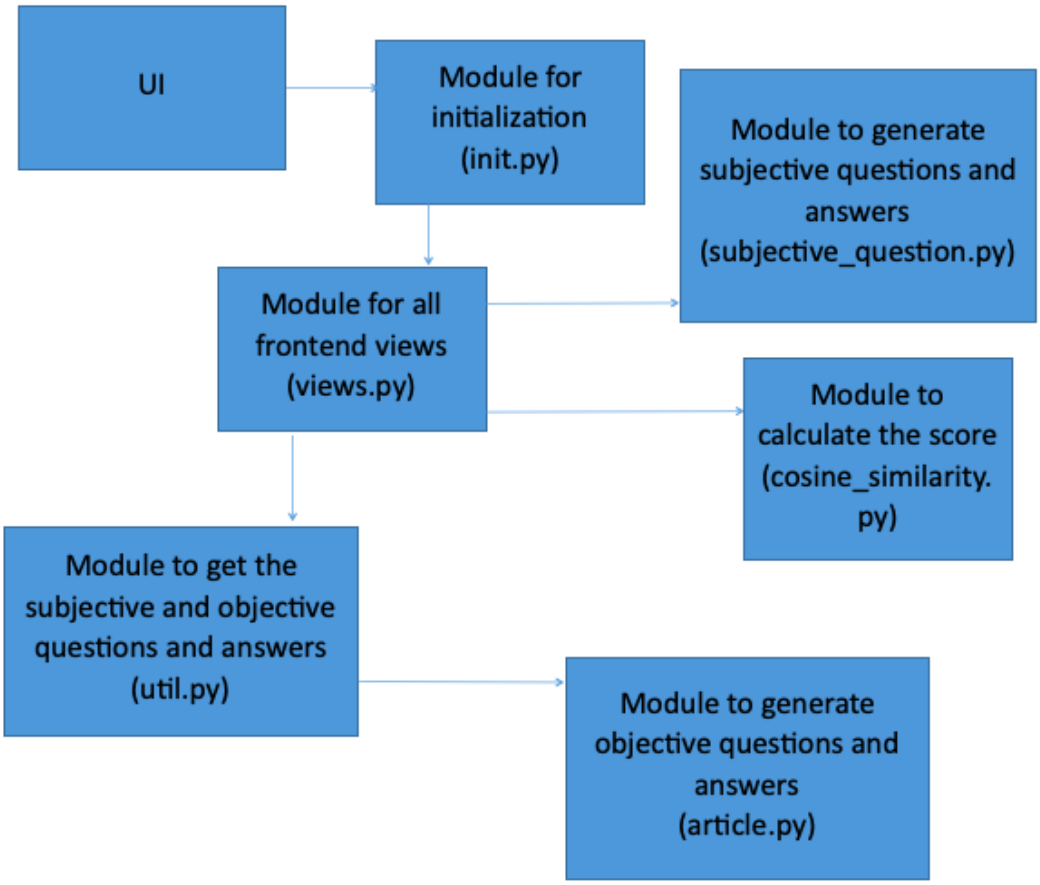

Figure 4.1: System Overview Diagram

1. Through the user interface the user will enter the details which is accepted by HTML files used to make that particular page.

2. The entries made by the user has ids assigned to it which will be referred in other py files or the modules handling different functions.

3. The views module is the file which has few functions which gets different user information and renders different HTML templates.

4. There is a separate file for subjective questions and answer generation which is called subjective_question.py , this file will generate questions and answers based on the selection of the type of the test and subjects

made by the students.Later on it applies the NLP concepts to the subject word file and form the questions and answers.

5. A module or file named article.py will generate objective questions and answers by using NLP concepts same like subjective questions.

6. There is a file which is used to access the subjective and objective questions and answers based on the choice made by the students. This file has functions to access the question answer pair .

7. The views.py file then finally compares the answers given by students to the answers made by the system and calls the cosine_similarity.py for final assignment of scores according to the similarity of answers , this file also calculates max, min and mean of the scores for the subject selected.

4.2 Software Architecture

4.2.1 System Block Diagram

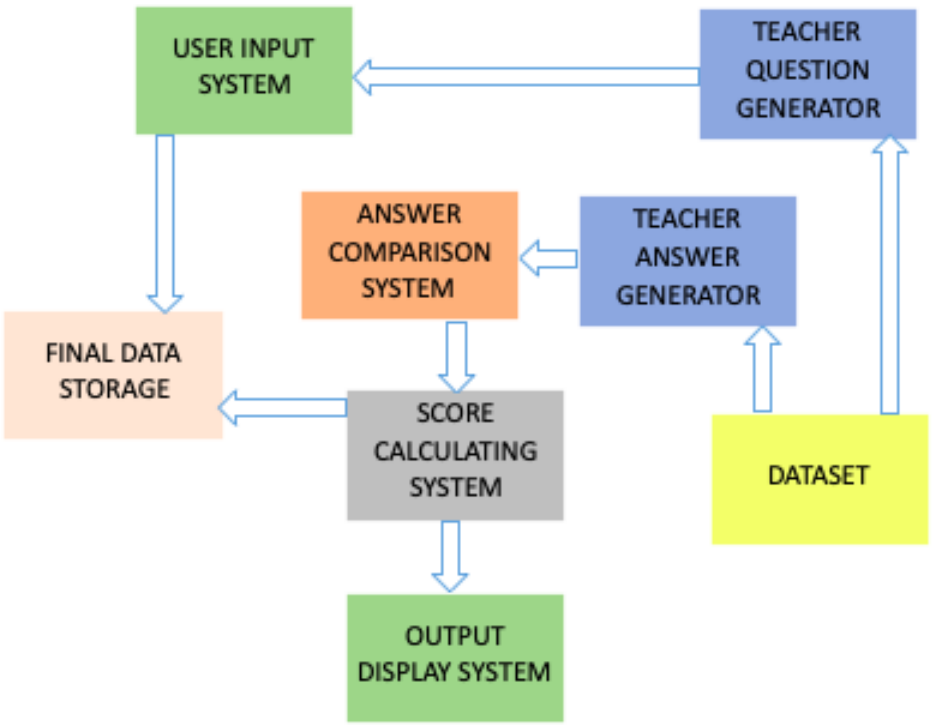

Figure 4.2: System Block Diagram

User input system accepts the input from the students. Following are the inputs given by the students:

1. Username
2. USN
3. Semester
4. Type of Exam(Subjective or Objective)
5. Subject

Teacher question generator system will generate questions from the dataset files of the subjects using concepts of NLP.This system has the regular expressions which creates questions from the paragraphs given .It creates questions of both types , subjective and objective.

The answer generator system will generate answers consecutive to the questions from the dataset files of all the subjects . It picks up the necessary keywords from the questions and forms answers related to those keywords.

The answer comparison system will compare the answers given by the students and the answers that are decided by the system for that particular subjects questions. It will compare and divide 100 marks between the number of questions.

Finally the score calculating system will calculate the final score based on similarity of the answers between system answers and students answer and will assign the scores accordingly in percentage . It also calculates mean score, min score and max score achieved in that particular test .

Our system has three files for three subjects which has the information related to that subject and question and answer generator systems will use those paragraphs to create questions and answers.

All the details entered by the students will be saved in the excel files of objective and subjective tests which was previously created by the system , this file has all the input values along with date , and final scores.

4.2.2 Data Flow Diagram

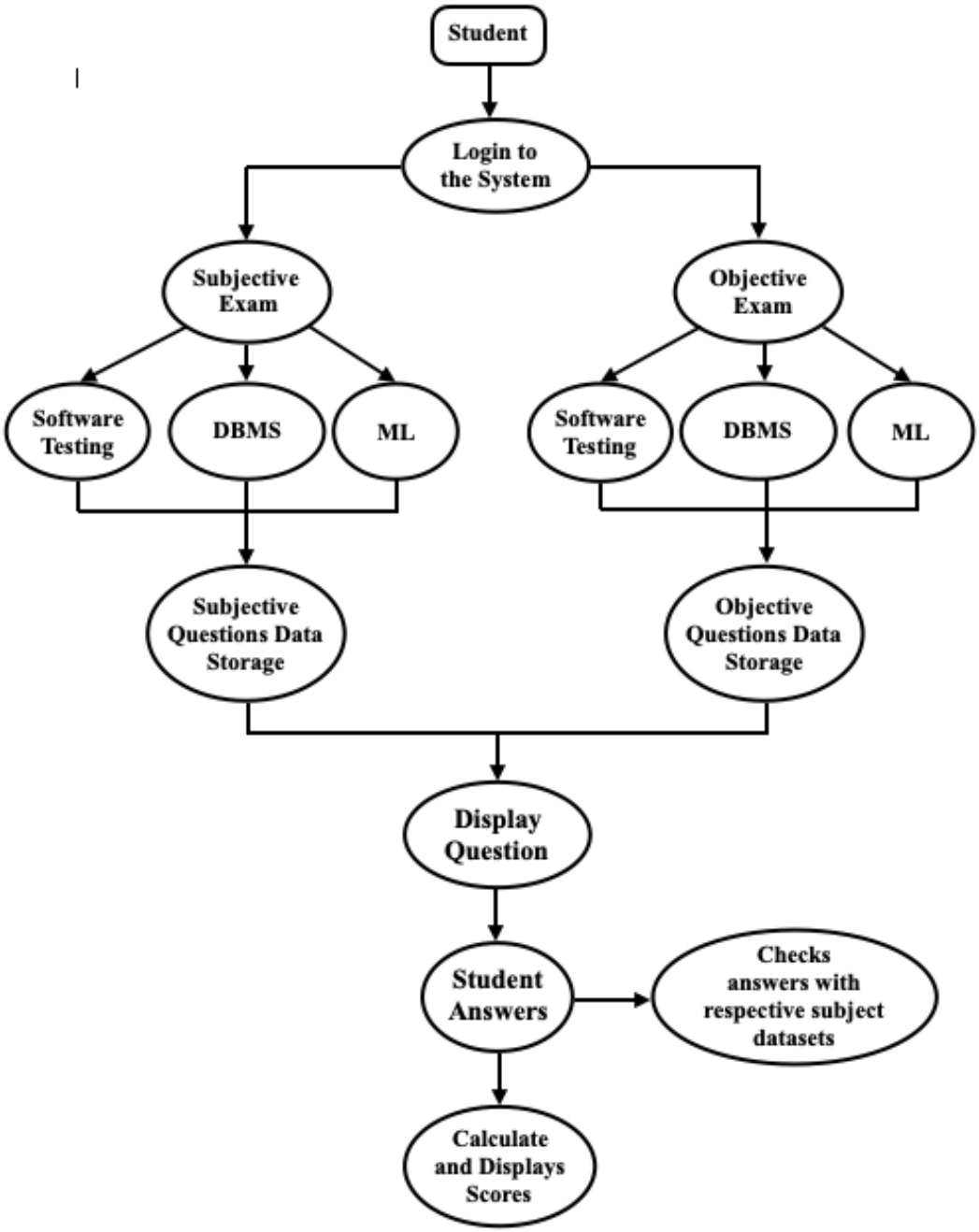

Figure 4.3: Data Flow Diagram

A data flow diagram (DFD) is a graphical representation of the flow of data through an information system. A DFD gives the preliminary overview of the system without going into great detail. Fig.4.3 represents the DFD of our proposed system. The flow of the system is as follows:

1. The student logs in to the system showing two options of Subjective

Exam and Objective Exam.

2. Under respective exam, students have the options to select any of the 3 subjects i.e. Software Testing, DBMS and ML are present.

3. Details entered by the students will be saved automatically to the respective subjective and objective files.

4. The system will automatically generate each subject's questions and answers.

5. Questions will be displayed on the screen for which the students have to answer the question

6. The answers given by the student will be automatically checked with respective subject's question answer sets which is already there in the system.

7. Scores will be calculated according to the answers given by the student and display it on the screen.

4.2.3 Sequence Diagram

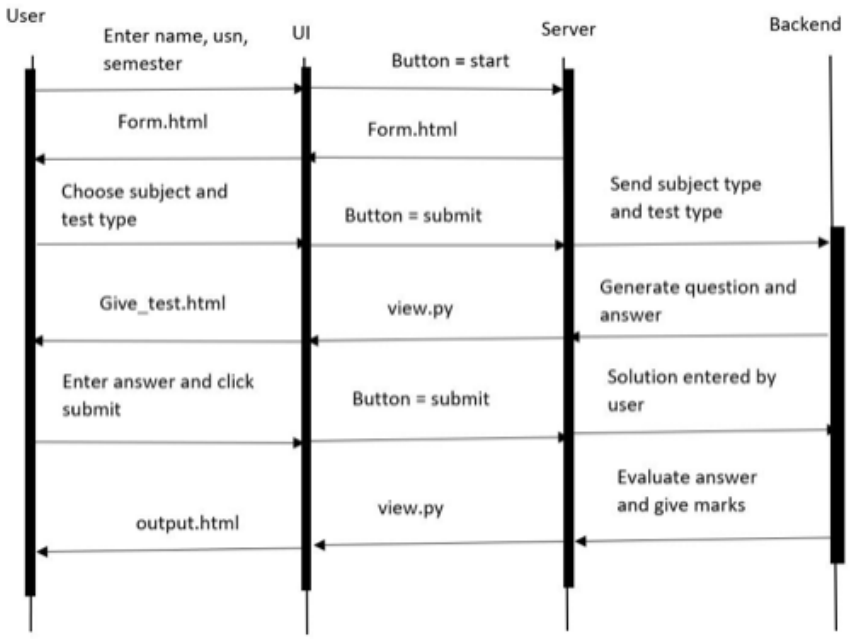

Figure 4.4: Sequence Diagram

1. Sequence diagram are interaction diagrams that detail how operations are carried out. They capture the interaction between objects in the context of a collaboration.

2. Above sequence diagrams shows the interaction between user, UI, server and backend.

3. As shown in the diagram the user first needs to enter name, use and semester and it goes to next page, which displays subject and type of exam.

4. Server passes this information to backend code which generates question answer pair and questions are displayed in the next page where the user enters the answer and next answer is evaluated and score is displayed in next page.

5

Implementation

5.1 Implemenation Platform

5.1.1 Software

- **Development Environment:** Microsoft Visual Studio
- **Web Framework:** Flask, Bootstrap
- **Programming Language:** Python
- **Front End:** JavaScript, HTML, CSS
- **Natural Language Processing** is used for evaluation of answer.

5.2 Implementation Details

5.2.1 Directory Structure Diagram

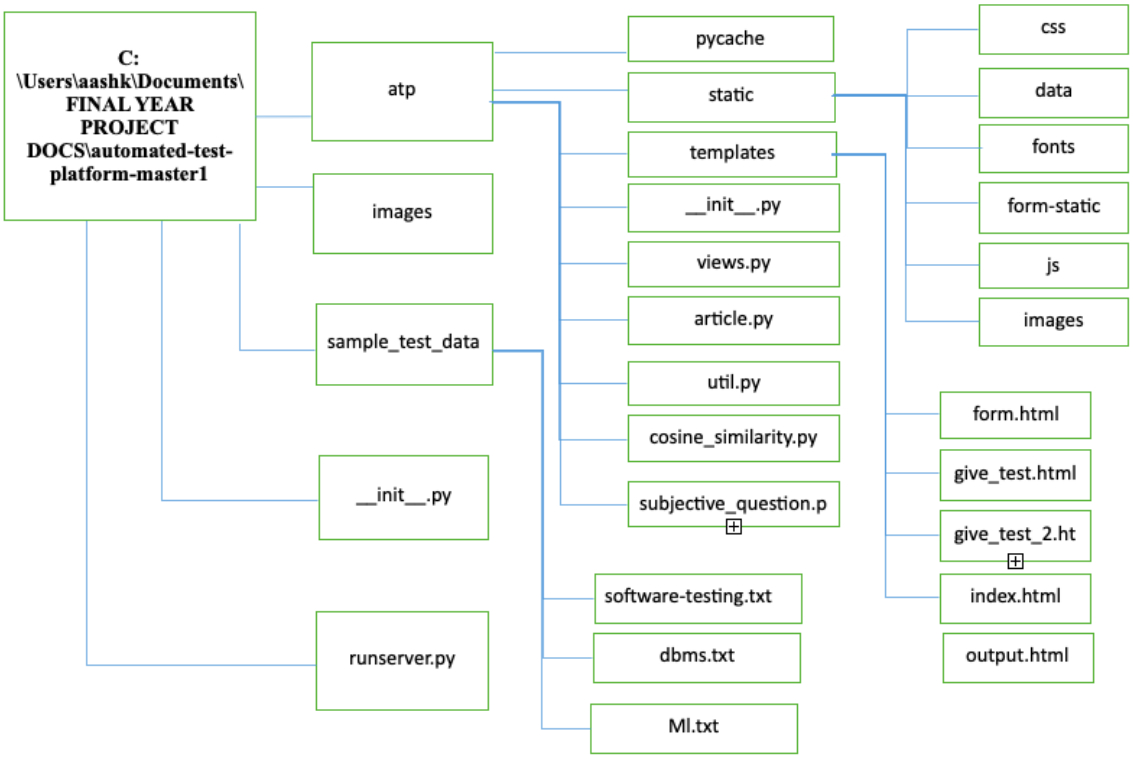

Figure 5.1: Directory Structure Diagram

5.2.2 Dataset Collection

As the dataset we have used three text files corresponding to three subjects that we have given as options to choose from for students. We have given software testing, dbms, and ml as three subjects for the students to choose from.

For these three subjects we have three text files which has paragraphs of information written about these subjects.When the student selects the subject , the respective text file will be extracted through its path given and rest of the processing is done on the paragraphs by applying concepts of Natural Language Processing.We can add more subjects and hence add more text files to be worked on.

5.2.3 Data Entry

A student will start by entering the details like username, USN, semester and will press the enter button.Each of these variables are provided a unique id in frontend to deal with those values in backend.

After this the student will select the type of exam he/she wants to give, that is objective or subjective and also has to choose the subject .Once that is done the values will be passed on to views.py after they are collected through html files.

views.py file is the integration point for the frontend and backend ,it has a function for each page that is displayed on the screen and calls all the functions from each of the other files to deal with the entered data.

5.2.4 Objective Question and Answer Generation

Once the entered data is accepted a function from article.py file is called to act upon the entered data.If a student selects a subject “ML” then the text file containing information of ML will be called and read .Later the paragraphs in that file will be changed to sentences and further into phrases along with replacement of phrases for generation of questions and answers.

This file has a regular expression which we have chosen according to accuracy of its selection of phrases, this grammar helps in chunking , that is extraction

of meaningful phrases.

Further a list of dictionaries if formed with each dictionary having three keys namely “Answer”, “Answer_key”, “Question” along with its values that extracted from the file.

5.2.5 Subjective Question and Answer Generation

If a student chooses subjective type of exam and chooses a subject then the subjective_question.py file functions are called with argument as the name of the file corresponding to the subject that the student has chosen.

Again the concepts of NLP will be applied on that file and the Subjective questions and answers will be generated. These questions and answers are also stored in the list of dictionaries with the keys namely “Answer”, “Answer_key”, “Question” along with its values that extracted from the file.

5.2.6 Getting the pair of Generated Question and Answers

In order to compare the answers written by the students with answers that are generated by the system the views.py file calls the functions from util.py file which gets the generated questions and answers.

This file returns pairs of questions and answers generated by functions from article.py file and subjective_question.py. util.py file has functions which call the functions from the above mentioned files to generate the objective and subjective questions and answers.

5.2.7 Score Calculation

Finally when both the answers that is the answer from the students and the answer from the system are compared ,the views.py file calls the functions from cosine_similarity.py file which calculates the score , mean, minimum and maximum values achieved in that particular test till now.

In this file the vectors of both the answers are taken and comparison of words is done , if the keyword is found in main answer than 1 is added to the list and if not than 0.Then the square root of the final vector is taken for the student answer and for system answer and their dot product is found

and is divided by the cross product and multiplied by 100 to get answer in percentage.

6

Testing

Testing is done by giving the details of the students first in the homepage after which the options to select type of exam is given along with the subjects. Students chooses their preferred type of exam and questions are displayed on the screen with a box to write down their answers. After the evaluations of their answers, mark scores are displayed.

Below are the images of the result of testing: Below are the images of the result of testing:

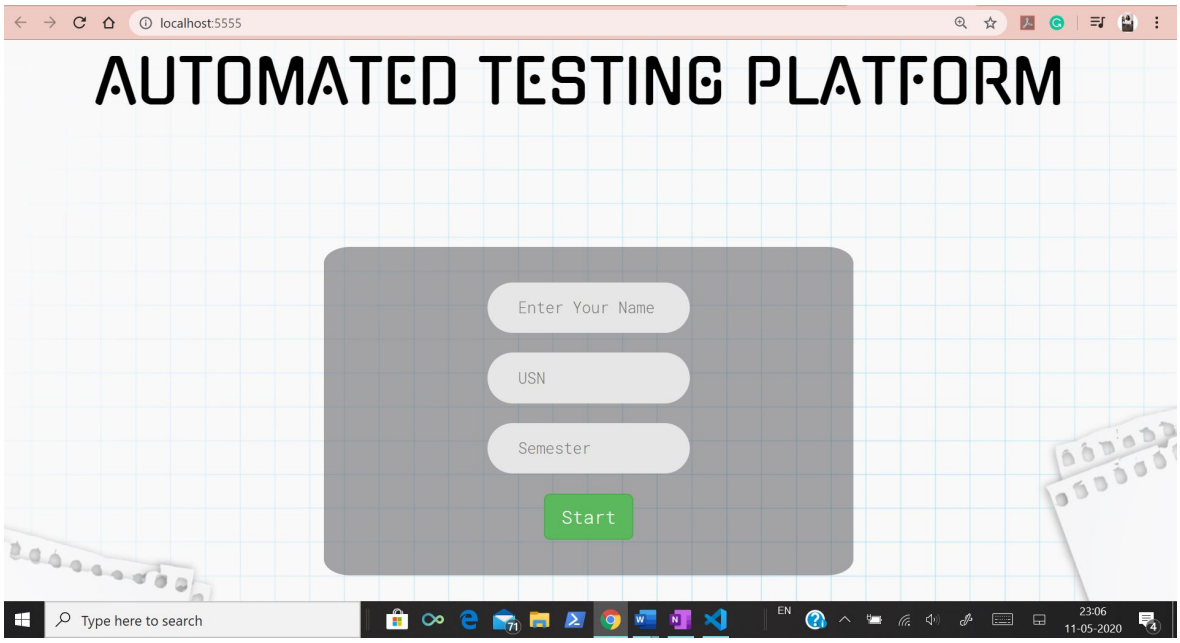

Figure 6.1: Homepage

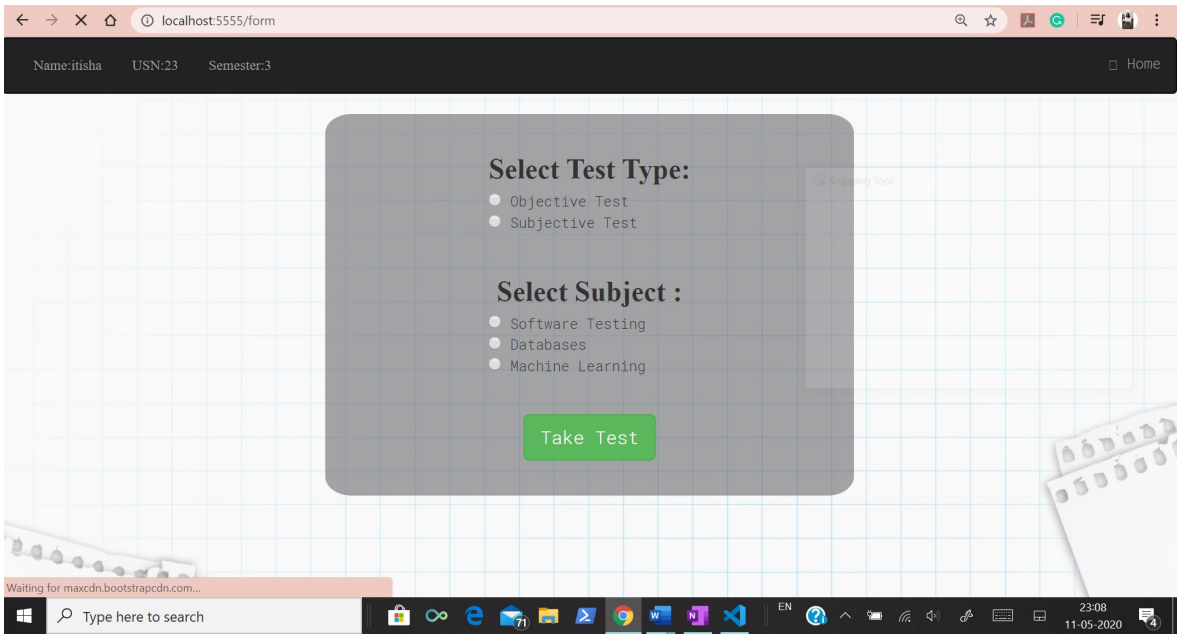

Figure 6.2: Options to Select Type of Exam and Subjects

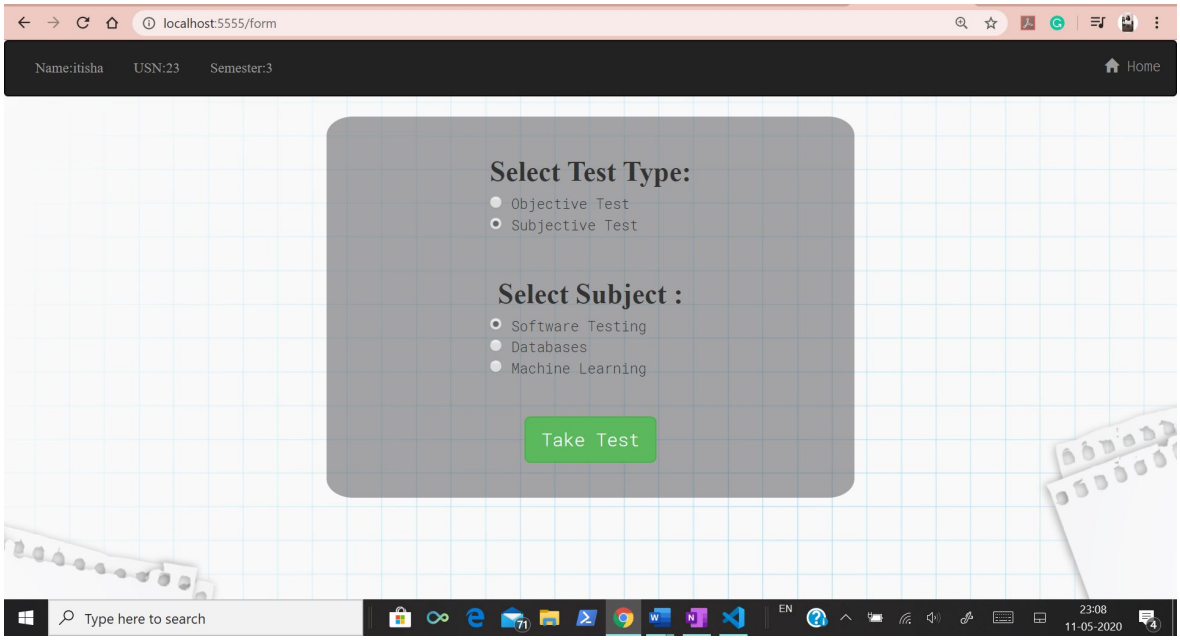

Figure 6.3: Options are selected i.e. Subjective Exam

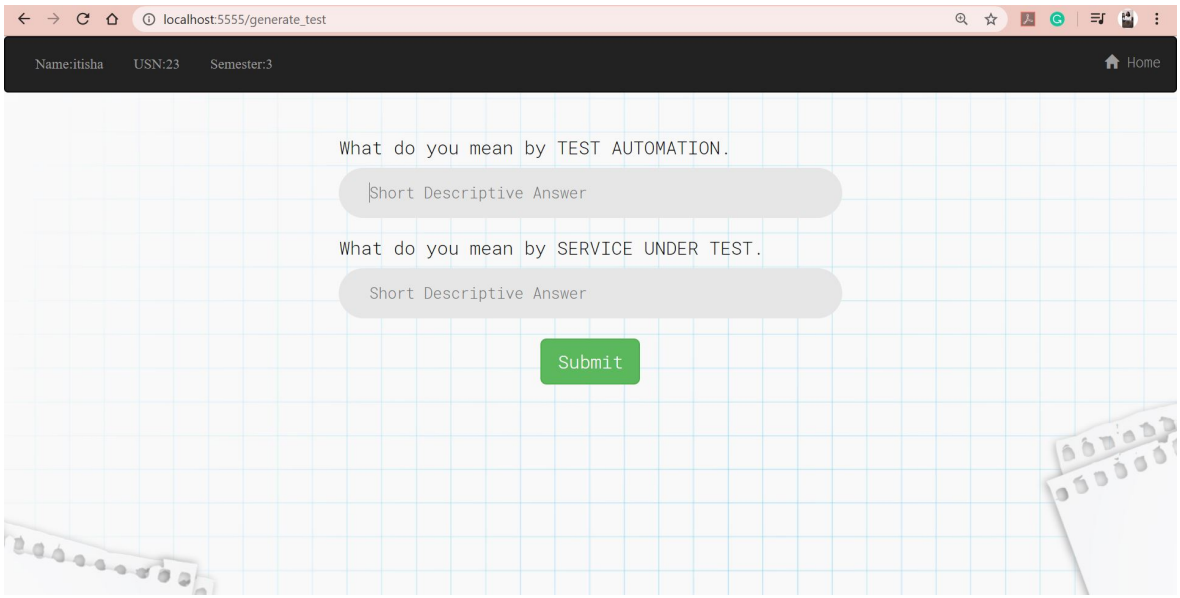

Figure 6.4: Questions are Displayed

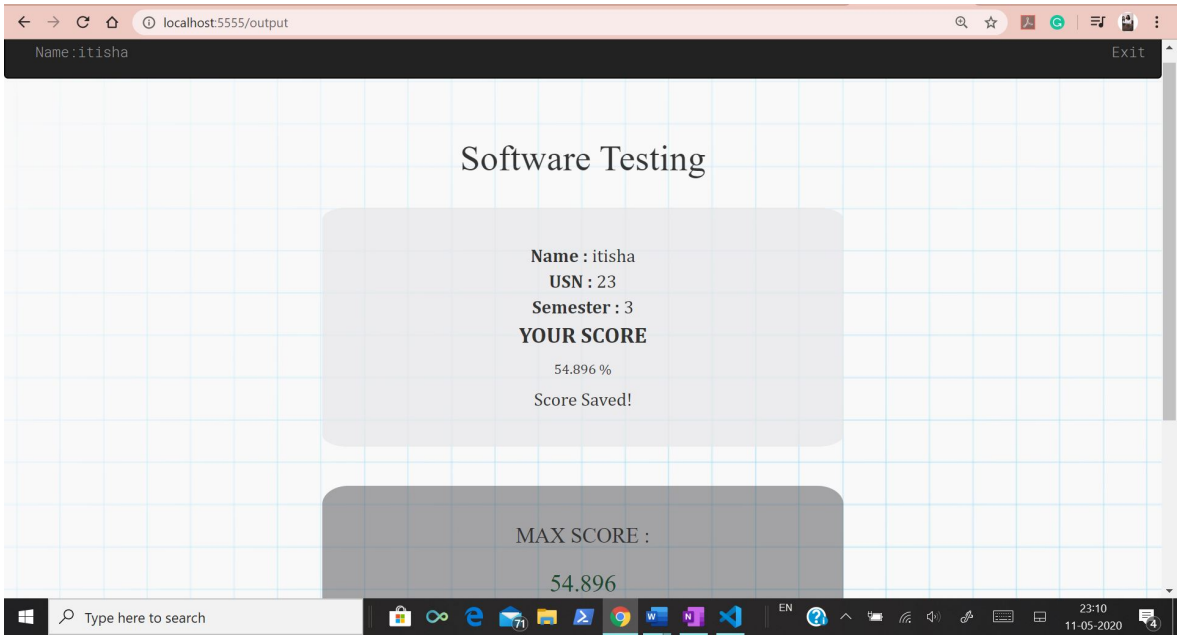

Figure 6.5: Scores

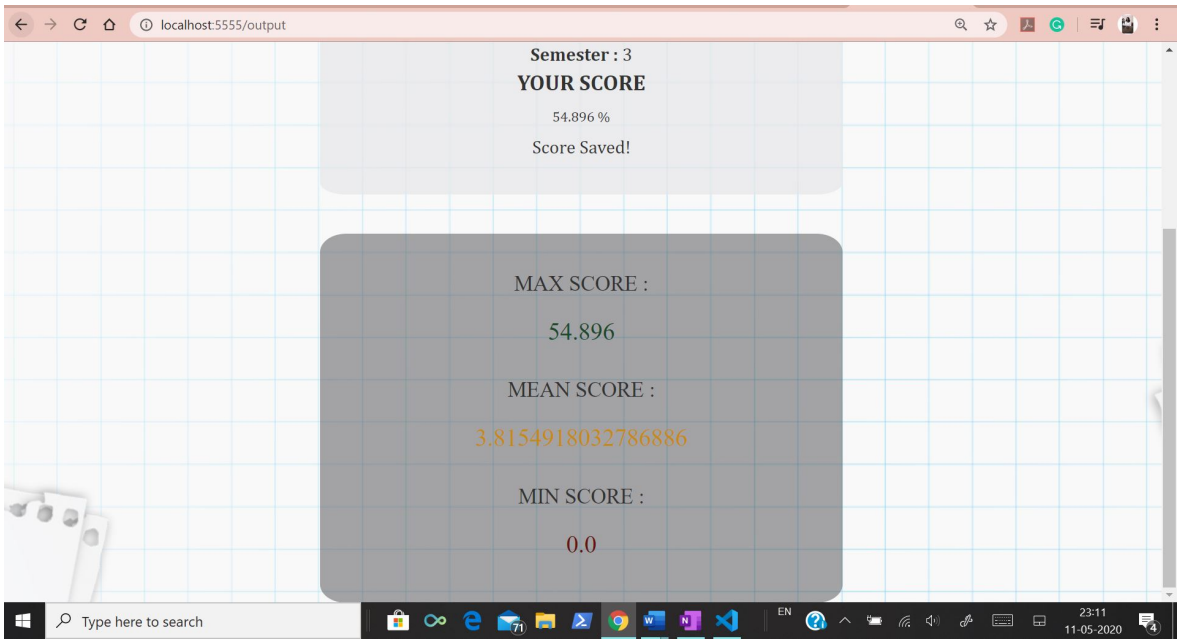

Figure 6.6: Maximum, Minimum and Mean Scores

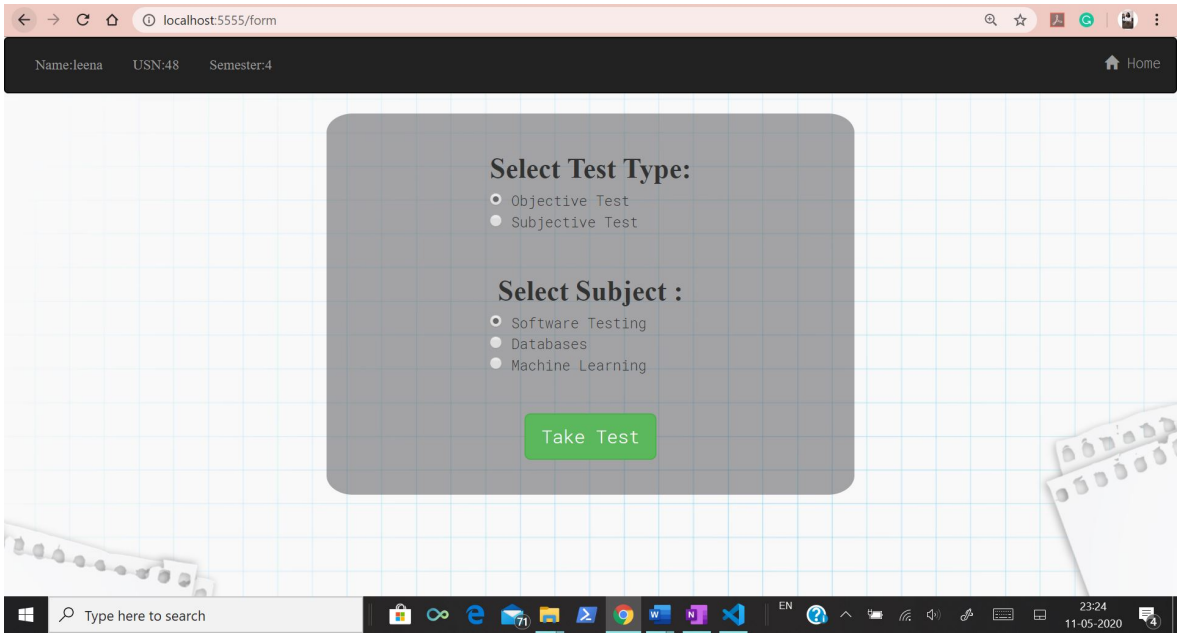

Figure 6.7: Opetions are selected i.e. Objective Exam

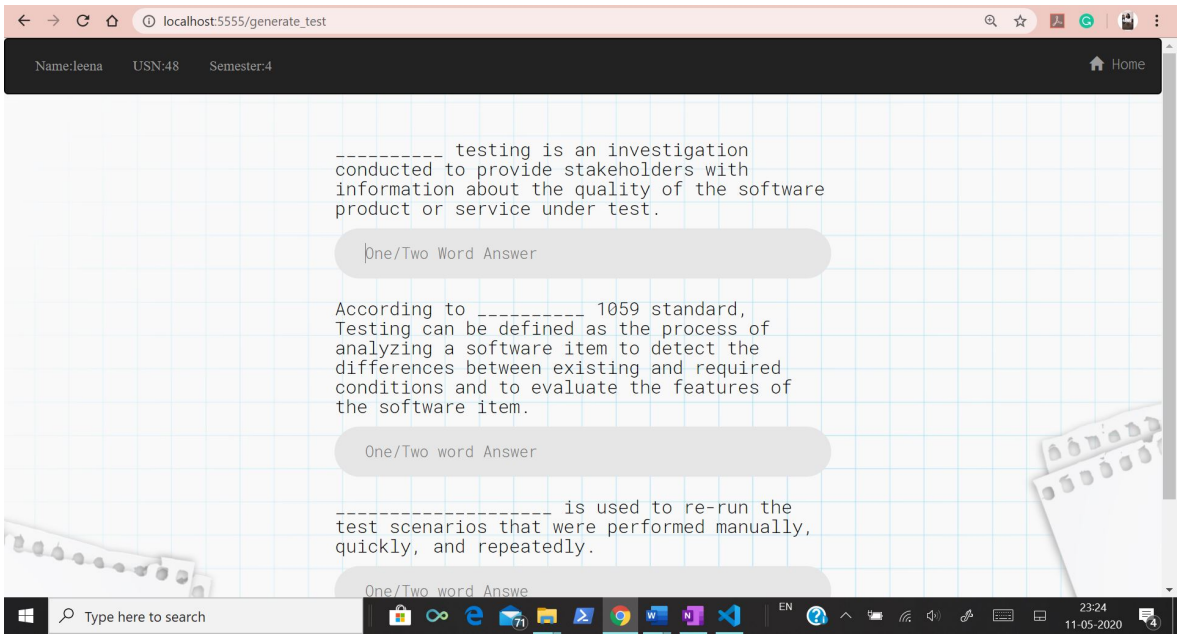

Figure 6.8: Questions are Displayed

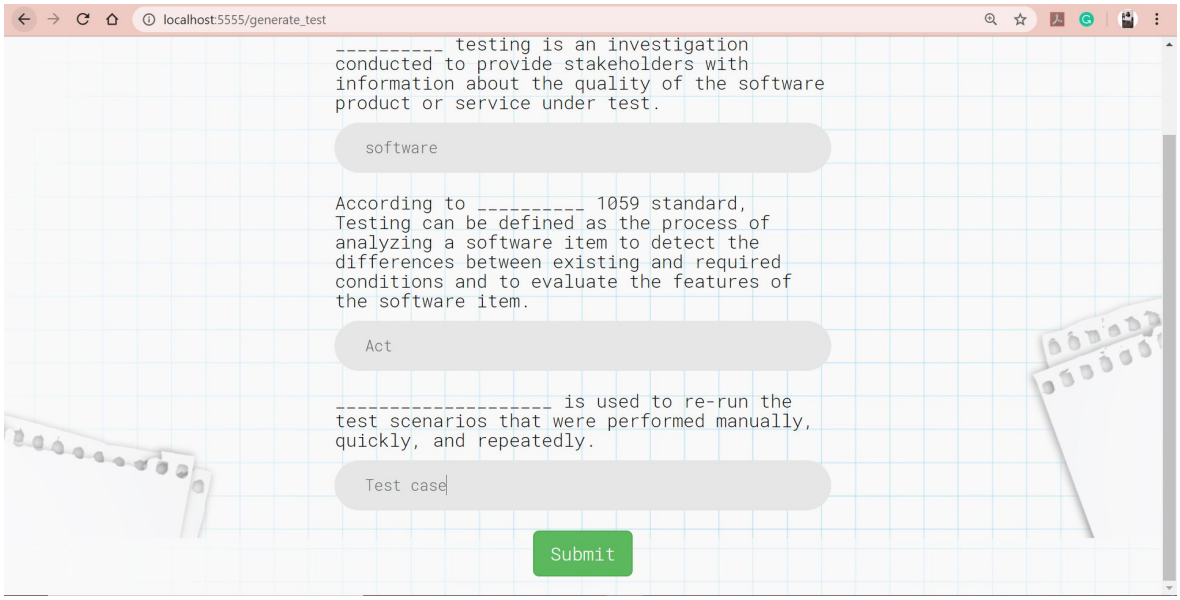

Figure 6.9: Answers

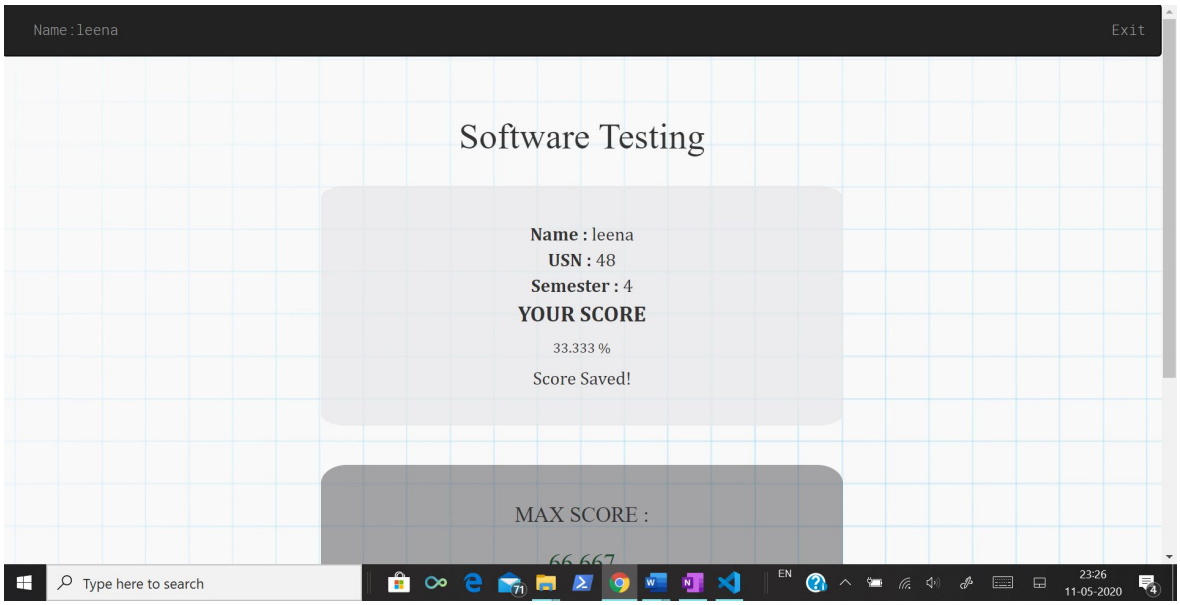

Figure 6.10: Scores

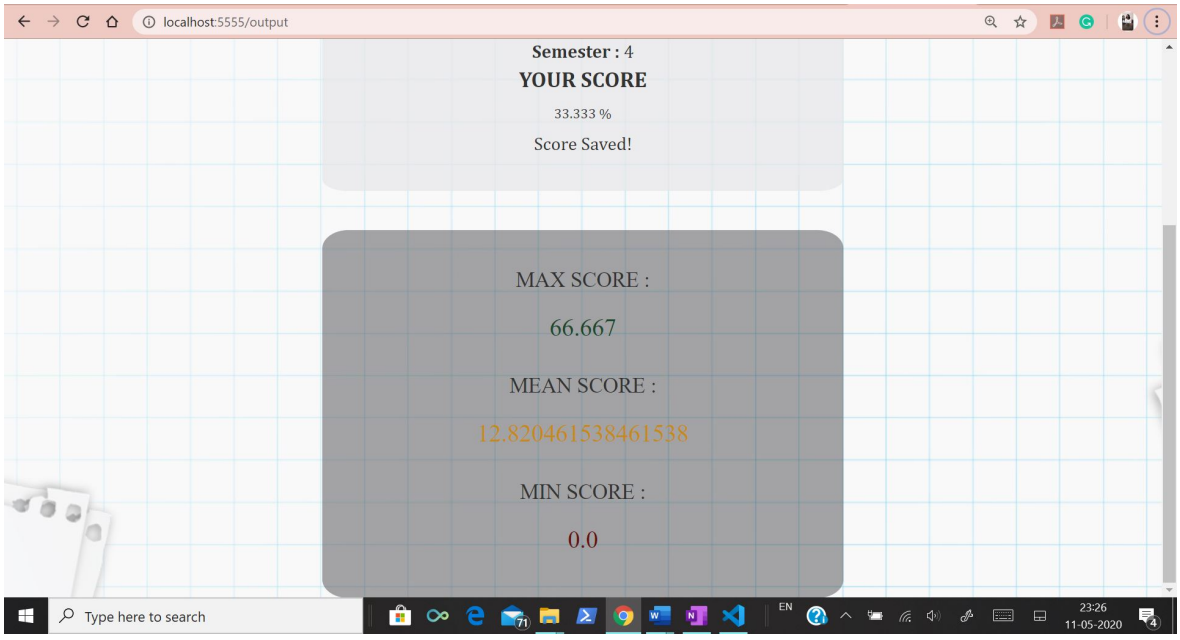

Figure 6.11: Maximum, Minimum and Mean Scores

7

Conclusion

The proposed system is achieved using Natural Language Processing for automatic evaluation of answer scripts. The objective of the proposed system is achieve using the following modules: views.py integrates backend and frontend files.

Subjective and objective questions and answers are obtained by util.py. article.py generates objective questions and answers. subjective_question.py generates subjective questions and answers. Final score are evaluated by cosine_similarity.py.

The stress of manual correcting large amount of answer scripts by the faculties will be reduced with the help of this proposed system. Several error like totalling scores which could have occurred are also prevented. Student's handwriting need not worry about their handwriting while writing the answers.

Future Enhancements

The proposed system gives a method for automated evaluation of answer scripts using natural language processing. Some of the future enhancements for the proposed system can be:

1. Improved database can be used to store the details.
2. It will be able to evaluate long answers written by the students.
3. Handwritten answers by the students can be scanned using image processing and evaluate the marks accordingly.
4. Teachers will be able to login, update questions and review answers.

The proposed system can be evolved to. meet several other operations which are not included in this project. Expanding the system will result in more efficient and hassle free operations.

8

References

1. A Scoring Tool for Electronic Paper Exams, in IEEE, 2007. Narcis Lozano, Koichi Hirosawa and Masaki Nakagawa.

2. Indonesian Question Answering System for Solving Arithmetic Word Problems on Intelligent Humanoid Robot, 2018. Alexander A S Gunawan, Pribadi R Mulyono and Widodo Budiharto

3. Textual Question Answering for Semantic Parsing in Natural Language Processing. Jaydeb Sarker, Mustain Billah and Md. Al Mamun

4. Intelligent Question Answering System based on Artificial Neural Network

5. Student's Academic Performance Evaluation Method Using Fuzzy Logic System. Abdul Aziz, Md. Asaf-uddowla Golap and M. M. A. Hashem

6. Computerized Paper Evaluation Using Neural Network. Tanupriya Choudhury, Kartikeya Jain, Lakshya Aggarwal, Ayushi Gupta and Garv Saxena

9

Appendix: Code

Objective_gen.py

```
from nltk.corpus import wordnet as wn
from textblob import TextBlob
import re
import nltk

class objective :
    def __init__(self, title):
        self.title = title
        with open(title, mode="r") as fp:
            self.summary = fp.read()

    def sentence_gen(self):
        Sentences1 = nltk.sent_tokenize(self.summary)
        lis_sentences = list()
        for i in Sentences1:
            Single_sen = self.Evalsen(i)
            if Single_sen:
                lis_sentences.append(Single_sen)
        return Single_sen


    def getsimiwords(self, word):

        syns = wn.synsets(word, pos="n")
        if len(syns) == 0:
            return []
        else:
            synset = syns[0]
hypnym = synset.hypernyms()[0]
        hyponyms1 = hypnym.hyponyms()

        simiwords = []
        for hypo in hyponyms1:
            simiwords = hypo.lemmas()[0].name().replace("_", " ")
            if simiwords != word:
```

```
            simiwords.append(simiwords)
        if len(simiwords) == 8:
            break
    return simiwords


def Evalsen(self, sentence):

    tagslab = nltk.pos_tag(sentence)
    if tagslab[0][1] == "RB" or len(nltk.word_tokenize(sentence)) < 4:
        return None

    taglow = {word.lower(): t for w, t in tagslab}


    nounphr = list()

    gram2 = r"""
        CHUNK: {<NN>+<IN|DT>*<NN>+}
            {<NN>+<IN|DT>*<NNP>+}
            {<NNP>+<NNS>*}
        """
    cHun = nltk.RegexpParser(gram2)
    tOke = nltk.word_tokenize(sentence)
    postoken = nltk.tag.pos_tag(tOke)
    maintree = chunker.parse(postoken)
    for j in tree.subtrees():
        if j.label() == "CHUNK":
            temp = ""
            for k in j:
                Temp1 += k[0]
                Temp1 += " "
            Temp1 = Temp1.strip()
            nounphr.append(Temp1)

    replnoun = list()
    for w, t in tagslab:
        for phr in nounphr:
            if phr[0] == '\'':
               break
```

```
            if w in phr:
                [replnoun.append(phrword) for phrword in phr.split()[-2:]]
                break

        if len(replnoun) == 0:
            replnoun.append(w)
        break

    if len(replnoun) == 0:
                    return None
    Val1 = 99
    for l in replnoun:
        if len(l) < Val1 :
            Val1 = len(l)

    Single_sen = {
        "answer": " ".join(replnoun),
        "answerkey": Val1
    }

    if len(replnoun) == 1:

        Single_sen["similar_words"] = self.getsimiwords(replnoun[0])
    else:

        Single_sen["similar_words"] = []

          replphr = " ".join(replnoun)
    blnphr= ("_________" * len(replnoun)).strip()

    mainexpression = re.compile(re.escape(replphr), re.IGNORECASE)
    sentence = mainexpression.sub(blnphr, str(sentence), count=1)

    Single_sen["Question"] = sentence
    return Single_sen
```

ScoreCal.py

```
import nltk as nlp
import math

import numpy as np
import os

def modofvector(v):
    Sum1 = 0
    for x in v:
        Sum1 += x**2
    return math.sqrt(Sum1)


def comvector(slis, mlis):
    myvc = list()
    for i in range(len(mlis)):
        Key1 = mlis[i]
        if Key1 in slis:
            myvc.append(1)
        else:
            myvc.append(0)
    return myvc


def getvc(stristr):
    sen= nlp.sent_tokenize(stristr)
    tlist = list()
    for s in sen:
        tempo1 = nlp.word_tokenize(s)
        for x in tempo1:
            tlist.append(x)
    return tlist


def subjanseval(orians, userans):
    score = 0

    orilis = getvc(orians)
```

```
    userlis = getvc(userans)
    mlis= orilis + userlis

    veco1 = comvector(orilis, mlis)
    veco2 = comvector(userans, mlis)

    v1 = modofvector(vecor1)
    v2 = modofvector(vecor2)

    v1dotprov2 = np.dot(veco1, veco2)

    dista = v1dotprov2 / (v1 * v2)
    scrobtain = dista * 100

    return scrobtain
```

Subjective_gen.py

```
import nltk as nlp
import numpy as np
from nltk.corpus import stopwords


qform = [
    "Explain in detail ",
    "Define ",
    "Write a short note on ",
    "What do you mean by "
    ]
grammer = r"""
    CHUNK: {<NN>+<IN|DT>*<NN>+}
    {<NN>+<IN|DT>*<NNP>+}
    {<NNP>+<NNS>*}
    """


def gensubjective(filepath):

    flp = open(filepath, mode="r")
    data = flp.read()
```

```
flp.close()

Sentences1 = nlp.sent_tokenize(data)

stopwor = list(stopwords.words("english"))
dictofqa = dict()
  cp = nlp.RegexpParser(grammer)

for sen in Sentences1:

    wordtag = nlp.pos_tag(nlp.word_tokenize(sen))

    Tree1 = cp.parse(wordtag)
    for subtr1 in Tree1.subtrees():
        if subtr1.label() == "CHUNK":
                    tempp = ""

            for s in subtr1:
                tempp += s[0]
                tempp += " "
            tempp = tempp.strip()
            tempp = tempp.upper()
                        if tempp not in dictofqa:
                if len(nlp.word_tokenize(sen)) > 20:
                    dictofqa[tempp] = sen
            else:
                dictofqa[tempp] += sen

keylist = list(dictofqa.keys())
qapair2 = list()

for x in range(3):
    rand_num = np.random.randint(0, len(keylist))
    selkey = keylist[rand_num]
    ans = dictofqa[selkey]

    rand_num %= 4
    queformat1 = qform[rand_num]

    quest = queformat1 + selected_key + "."
```

```
        qapair2.append({"Question": quest, "Answer": ans})
    return qapair2
```

test1.html

```
<html lang="en">
<head>
   <title>{{testname1}}</title>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1">
   <link rel="stylesheet" type="text/css"
      href="../static/formstatic/css/main.css">
   <link rel="stylesheet" href="../static/css/style.css">
   <link rel="stylesheet"
      href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.0/css/bootstrap.min.css">
    <script
      src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>
    <script
      src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.0/js/bootstrap.min.js"></script
</head>
<body class="bod1" >

   <nav class="navbar navbar-inverse">
      <div class="contfluid">
        <ul class="nav navbar-nav">
         <li><p class="nav navbar-text">Name:{{username}}</p></li>
         <li><p class="nav navbar-text">USN:{{usn}}</p></li>
         <li><p class="nav navbar-text">Semester:{{semester}}</p></li>
        </ul>
        <ul class="nav navbar-nav navbar-right">
         <li><a href="/"><span class="glyphicon glyphicon-home"></span> Home
            </a></li>

        </ul>
      </div>
   </nav>
   <div class="contact1new">
      <form style="color:black" class="contact1-form validate-form"
         action="/output" method="POST" enctype="multipart/form-data">
```

```
        <div class="wrap-input21 validate-input" data-validate="">
           <h4>{{que1}}</h4>
           <input autocomplete="off" class="in1" type="text" name="ans1"
              id="ans1" placeholder="One to Two Word Answer">
        </div>
        <div class="wrap-input21 validate-input" data-validate="">
           <h4>{{que2}}</h4>
           <input autocomplete="off" class="in1" type="text" name="ans2"
              id="ans2" placeholder="One toTwo word Answer">
        </div>
        <div class="wrap-input21 validate-input" data-validate="">
           <h4>{{ques3}}</h4>
           <input autocomplete="off" class="in1" type="text" name="ans3"
              id="answer3" placeholder="One toTwo word Answe">
        </div>
        <div class="container-contact1-form-btn">
           <button class="btn btn-success btn-lg">
              <span>Submit<i class="" aria-hidden="true"></i></span>
           </button>
        </div>
        <br>
     </form>
  </div>
</body>
</html>
```

outputfile.html

```
<html lang="en">
<head>
   <title>{{username}}</title>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1">
   <link rel="stylesheet" href="../static/css/style.css">
   <link rel="stylesheet"
      href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.0/css/bootstrap.min.css">
    <script
       src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>
    <script
       src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.0/js/bootstrap.min.js"></script
```

```
   <style>
      .vl {
         border-left: 2px solid #ffcd48;
         height: 150%;
      }
      </style>
</head>
<body class="bod2">
   <nav class="navbar navbar-inverse">
      <div class="container-fluid">
        <ul class="nav navbar-nav">
         <li><p class="nav navbar-text">Name:{{username}}</p></li>
        </ul>
        <ul class="nav navbar-nav navbar-right">
         <li><a href="/"> Exit </a></li>

        </ul>
      </div>
   </nav>
   <br>
   <center>
      <div style="font-family:'Times New Roman', Times, serif;
          "><h1>{{subjectname}}</h1></div>
      <br>
   </center>
   <center>
      <div class="row3">
         <div>
            <span style="font-weight: 900;">Name :
                </span><span>{{username}}</span><br>
            <span style="font-weight: 900;"> USN :
                </span><span>{{usn}}</span><br>
            <span style="font-weight: 900;">Semester :
                </span><span>{{semester}}</span>
         </div>
         <center>
            <span style="font-weight: 900; font-size:larger;">YOUR SCORE
                </span><h5 id="fs" class="h1_beta">{{show_score}} %</h5>
         </center>
         <div style="text-align: center;">{{status}}</div>
```

```
      </div>
   </center><br><br>
   <center>
      <div class="row2">
         <span style="font-size: 150%; font-family:'Times New Roman', Times,
            serif">MAX SCORE : <h3 style="color: rgb(27, 73,
            45)";>{{max_score}}</h3></span>
         <br>
         <span style="font-size: 150%; font-family:'Times New Roman', Times,
            serif">MEAN SCORE : <h3 style="color: rgb(199, 138,
            24);">{{mean_score}}</h4></span>
         <br>
         <span style="font-size: 150%; font-family:'Times New Roman', Times,
            serif">MIN SCORE : <h3 style="color: rgb(102, 19,
            13);">{{min_score}}</h5></span>
      </div>
   </center>
</body>
</html>

   <script>
      var font_s = {show_score};
      if (font_s < 55){
         document.getElementById("fs").style.color="red";
      }
      else{
         font_s += "px";
         document.getElementById("fs").style.fontSize=font_s;
      }
   </script>
```

test2.html

```
<html lang="en">
<head>
   <title>{{testname}}</title>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1">
   <link rel="stylesheet" type="text/css"
      href="../static/form-static/css/main.css">
```

```
    <link rel="stylesheet" href="../static/css/style.css">
    <link rel="stylesheet"
        href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.0/css/bootstrap.min.css">
     <script
        src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>
     <script
        src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.0/js/bootstrap.min.js"></script
</head>
<body class="body2">
    <nav class="navbar navbar-inverse">
      <div class="container-fluid">
        <ul class="nav navbar-nav">
         <li><p class="nav navbar-text">Name:{{username}}</p></li>
         <li><p class="nav navbar-text">USN:{{usn}}</p></li>
         <li><p class="nav navbar-text">Semester:{{semester}}</p></li>
        </ul>
        <ul class="nav navbar-nav navbar-right">
         <li><a href="/"><span class="glyphicon glyphicon-home"></span> Home
             </a></li>

        </ul>
      </div>
    </nav>
    <div class="contact1new">
      <form style="color:black" class="contact1-form validate-form"
          action="/output" method="POST" enctype="multipart/form-data">
        <div class="wrap-input21 validate-input" data-validate="">
          <h4>{{que1}}</h4>
          <input autocomplete="off" class="in1" type="text" name="ans1"
              id="ans1" placeholder="Short Descriptive Answer">
        </div>
        <div class="wrap-input21 validate-input" data-validate="">
          <h4>{{que2}}</h4>
          <input autocomplete="off" class="in1" type="text" name="ans2"
              id="ans2" placeholder="Short Descriptive Answer">
        </div>
        <div class="container-contact1-form-btn">
          <button class="btn btn-success btn-lg">
            <span>Submit<i class="" aria-hidden="true"></i></span>
          </button>
```

```
        </div>
        <br>
      </form>
   </div>
</body>
</html>
```