TESTANA

(Examination App)

Submitted in partial fulfilment of the requirements of the degree of

Bachelor of Technology

by

1. Asif Ali Khan (1605213015)
2. Saumitra Chauhan (1605213041)
3. Ritik Rajak (1605213032)

Supervisor

Ms. Mudita Sharan (IT Programme)

(Information Technology Programme)

INSTITUTE OF ENGINEERING & TECHNOLOGY LUCKNOW

(2020)

Contents

DECLARATION

CERTIFICATE

ACKNOWLEDGEMENT

ABSTRACT

LIST OF FIGURES AND TABLES

1. INTRODUCTION

2. LITERATURE REVIEW

3. METHODOLOGY

    3.1 Web app

    3.2 Generating Random Questions and Their Respective Answers

    3.3 User Input

    3.4 Evaluating User Answers

    3.5 Storing the Results

    3.6 Process

    3.7 Requirements

4. EXPERIMENTAL RESULTS

5. CONCLUSION

REFERENCES

ANNEXURE

Declaration

We hereby declare that this submission is our own work and that, to the best of our belief and knowledge, it contains no material previously published or written by another person or material which to a substantial error has been accepted for the award of any degree or diploma of university or other institute of higher learning, except where the acknowledgement has been made in the text. The project has not been submitted by us at any other institute for requirement of any other degree.

Submitted by: - Date: 12th August 2020

(1) Name: Saumitra Chauhan
Roll No: 1605213041
Branch: IT
Signature:

(2) Name: Asif Ali Khan
Roll No: 1605213015
Branch: IT
Signature:

(2) Name: Ritik Rajak
Roll No: 1605213032
Branch: IT
Signature:

Certificate

This is to certify that the project report entitled “TESTANA” presented by Saumitra Chauhan, Asif Ali Khan and Ritik Rajak in the partial fulfilment for the award of Bachelor of Technology in Information Technology, is a record of work carried out by them under my supervision and guidance at the Programme of Information Technology at Institute of Engineering and Technology, Lucknow.

It is also certified that this project has not been submitted at any other Institute for the award of any other degrees to the best of my knowledge.

Ms. Mudita Sharan
Information Technology Programme
Institute of Engineering and Technology
Lucknow

ACKNOWLEDGEMENT

In performing our project, we had to take the help and guideline of some respected persons, who deserve our greatest gratitude. The completion of this report for the project gives us much pleasure. We would like to show our gratitude to Ms.Mudita Sharan, Project Instructor, IT programme for giving us a good guideline for the project throughout numerous consultations. We would also like to expand our deepest gratitude to all those who have directly and indirectly guided us in writing this report.

In addition, a thank you to Professor Mr Pawan Kumar Tiwari Sir, who introduced us to the methodology of work, and whose passion for the “underlying structures” had lasting effect. Many people, especially our classmates and team members itself, have made valuable comment suggestions on this proposal which gave us inspiration to improve our report. We thank all the people for their help directly and indirectly to complete our assignment.

ABSTRACT

Due to worldwide pandemic CORONAVIRUS, it is becoming challenging to take a pen and paper based examination. There is an increasing need for an online examination system currently in all the schools and colleges. Also, it is hectic to make questions for an exam and then evaluate the user's answers. Unlike a traditional examination system, an online examination system is more enhanced in numerous ways.

From easier setup to remote access, to faster evaluations, an online exam system is better at fulfilling its primary role of being an evaluative tool. We implement a model to automatically generate questions with their respective answers and assess user responses . The candidate will answer the questions and his score will be evaluated as per the matching of his answer to the generated answer.

1. Introduction

The traditional examination system has been in place for a long time. Over the years, tweaks in exam duration, question format, evaluation methods, and feedback have evolved, but generally speaking this pen and paper model has persisted over decades of education. So why the change to an online examination system now? The traditional pen-paper procession begins with extended discussions on paper creation. Burning the midnight oil comes to life in the eleventh hour, where everyone scans through everything. The questions are handwritten/ typed, proofread, and again checked for repetition. Once your efforts pay off in the form of a beautifully designed paper, don't take a sigh of relief, and now is when the real efforts begin. Get set to find the invigilators and space for conducting the exams. Straining, stressing, draining, isn't it? What if there's a solution? Yes, there's a solution and it is known world-over as the most effective way to conduct examinations, and that is called online examination.

Simply put, with innovations in technology, an increasingly globalized world, and data-driven insights into teaching and learning methodologies, an online examination system might provide a better perspective into the scholarly journey of a student. Examinations are not just a part and parcel of student life. They are an evaluative tool that allows students, teachers, and parents to track and measure the student's scholarly progress. It also serves as a concrete metric on which to base learning and teaching decisions for the future. Unlike a traditional examination system, an online examination system is more enhanced in numerous ways. From easier setup to remote access, to faster evaluations, an online exam system is better at fulfilling its primary role of being an evaluative tool.

Online examination, also known as e-examination, enables examiners to conduct examinations through the internet or a company-wide intranet for remote candidates. Most **online exam** systems include answer processing modules, which allows evaluators to issue results as soon as candidates finish the test. This fully automated system evaluates the examinee thoroughly and presents the result in less time. Unlike traditional pen-and-paper exams, it is

possible to conduct open-book exams on an online exam system. In an online exam, a candidate can answer the questions in a given time, and the test window collapses once the exam is over, and evaluators keep answers ready for evaluation. The evaluator assesses responses, whether it be through an automated or manual process, and the result will be conveyed to the candidate either by mail or information available on the website. Many companies are resorting to online examination software as their preferred medium of choice for assessing professional capability and career adaptability. Conducting examination and answer sheet evaluation are hectic testing tools for assessing academic achievement, integration of ideas and ability to recall, but are expensive, resource and time consuming to generate questions and evaluate responses manually. Manual evaluation of the answer sheet takes up a significant amount of instructors' valuable time and hence is an expensive process. Also different security concerns regarding paper leakage is one of the other challenges to conquer.

1.1 Objective

Ever thought how hectic it sometimes becomes for the teacher to select good questions from the set of defined syllabus? Even what's more hectic is the evaluation of the answer sheets and display of the marks to the students. We the final year students bring into you the most curated examination application TESTana .

1.2 Scope

TESTana aims to build a computerized examination system using machine learning, natural language processing in a python environment using a flask framework, and web technologies to provide an inexpensive alternative to the current examination system.

We implement a model to automatically generate questions with their respective answers and assess user responses . The candidate will answer the questions and his score will be evaluated as per the matching of his answer to the generated answer. The cumulative score will then be stored in the database of the exam controllers from where the declaration of the result can be followed up.

2. Literature Survey

The paper (Automatic Thai Subjective Examination using Cosine Similarity Pongsakorn Saipech and Pusadee Seresangtakul Natural Language and Speech Processing Laboratory Department of Computer Science, Khon Kaen University Khon Kaen, Thailand, 40002) presents an automatic subjective examination system for Thai language based on semantic similarity, applying cosine similarity techniques together with the applicable synonym. The examination answers are input into the system in short Thai sentences, without word boundaries. The input answers which are reference answers, and student answers, were segmented into a sequence of Thai words; using the longest matching algorithm and dictionary (Lexitron®). These answers were transformed into vectors using TF-IDF, which is a frequency computation technique. The synonyms were also considered. Cosine similarity was used to measure the similarity of the answers. In order to evaluate the performance of the proposed system, we prepared a dataset of five questions from the final exam of the "Database Management System and Database Design" course, and asked 70 students to provide their answers. The datasets were prepared for both systems, with and without synonyms, in order to score the student answers. We cross checked the scores of both systems with an expert, their teacher. In comparing the results of both systems, we found that the scores produced from the system using cosine similarity with synonyms were similar to those obtained from the expert.

The paper (An E-Exam Management System under E-Network Management Course Shaimaa H. Shaker, Noor M. Jaafar) talks about the methods of teaching the E-learning become the one of great invention, there are many assessment methods in E-learning, one of these assessment is the essay, when teacher want to correct manually it may take long time, so the automatic scoring of essay in recent years has been increasing rapidly. Using usually the method of cosine similarity, it depends on feature extraction in the texts of answers, no matter about the order of the words in the statement. This study shows how the text similarity can have evaluated the student’s answers, so its need to be more accurate.

This paper (International Journal for Innovations in Engineering, Science and Management Volume 3, Issue 4, April 2015 Automatic Question Generation from Text) discusses how Humans are curious by nature. They ask questions to satisfy their never-ending quest for knowledge. For instance, students ask questions to learn more from their teachers, teachers ask questions to help themselves evaluate performance of the students, and our day-to-day lives involve asking questions in conversations. Questions are the significant constituent of countless learning interactions from one-to-one tutoring sessions to extensive assessments in addition to real life discussions One noticeable fact is that humans are not often very skilled in asking good questions because of their inconsistent mind in certain situations. It has been found that most people have trouble identifying their own knowledge deficiency This becomes our main motivation to automate the generation of questions with the hope that the potential benefits from an automated QG system could assist humans in meeting their useful inquiry needs. Question Generation (QG) and Question Answering (QA) became the two major challenges for natural language understanding communities, recently QG has turned into an essential element of learning environments, systems, information seeking systems, etc Considerable interest from the Natural Language Processing (NLP), Natural Language Generation , Intelligent Tutoring System, and Information Retrieval (IR) communities have currently identified the Text-to-Question generation task as a promising candidate for the shared task In the Text-to-Question generation task, a QG system is given a text (such as a word, a set of words, a single sentence, a text, a set of texts, a stretch of conversational discourse, an inadequate question, and so on), and its goal would be to make a set of questions for which the text contains answers.

This paper (EVALUATING STUDENTS' DESCRIPTIVE ANSWERS USING NATURAL LANGUAGE PROCESSING AND ARTIFICIAL NEURAL NETWORKS 1V. Lakshmi, 2Dr. V. Ramesh 1Research Scholar, Department of Computer Science and Applications 2Assistant Professor, Department of CSA, Sri Chandrasekharendra Saraswathi Viswa Maha Vidyala, Kanchipuram, India) gives brief about Computer based evaluation of students' performance is playing a vital role in world wide for all kinds of examinations. This method is somewhat faster

than our manual evaluating process. In this study a new method is proposed to evaluate the students' brief answers such as descriptive answers using Artificial Neural Networks [ANN]algorithm and Natural Language Processing [NLP] algorithms. In this system staff members create an answer sheet and keyword dataset for the examination process. These datasets are stored in data storage and students enter their answers in the examination page. This system automatically calculates results using two algorithms of NLP and ANN. Before this evaluation process the pre-processing technique is applied on the answers entered by the students. In this study, we used an Artificial Neural Networks algorithm for the normal answer comparison and stores marks for this in the database and also evaluated the same answer using Natural language processing [NLP] algorithm to check grammar mistakes and store the marks in the database and finally compares both marks and provides a final result. By these methods we can get an efficient result. The results given by the system is compared with the evaluation done by the faculty member.

This paper (Automatic Question and Answer Generation from Course Materials. A.S.M Nibras, M.F.F Mohamed, I.S.M Arham, A.M.M Mafaris, M.P.A.W Gamage Sri Lanka Institute of Information Technology)presents a Question and Answer Generating System based on the approach of Natural Language Processing. An examiner needs to ask suitable and good questions and prepare correct answers. The potential benefits of this system may assist examiners meeting appropriate questions and answers. When a text input is given, this system deeply analyzes the text content and generates a set of questions and suitable answers. This system functionally divided into four main steps namely, identifying key phrases and words, forming proper questions, validating generated questions and Find answers for validated questions. When analyzing text, a named entity recognizer and a part of speech are applied on each of these sentences to extract necessary information. Also classify the sentences based on their subject, verb, object and preposition for determining the possible type of questions to be generated. Once questions are generated, for further accuracy, system using a custom developed algorithm for validating the questions. Then, a suitable answer for the question will be generated. This paper provides an overview of Question and Answer Generation and a discussion on possible approaches found in the recent research as a guide to the development of an automated system.

3. Methodologies

3.1 Web App

We are building our application to be user-friendly and anyone with little knowledge of computers also use our application. The user will login to our web app and a session will be created for that user. The application will have different pages where the subjects will be shown on the page and the user can choose from the desired subject list. It will also have a page 2 different pages for subjective and objective questions respectively. Currently we will be testing the web app for 10 questions on the objective page and 10 questions on the subjective page.After that an evaluation page will be also there.

3.2 Generating Random Questions and Their Respective Answers

In order to generate questions and answers, the text content will be put into the process of Information Extraction. Information Extraction is the task of automatically extracting structured information from unstructured and/or semi-structured documents.

1. **Tokenization -** Its important step in information retrieval, the task of this step is splitting large strings made of texts into smaller pieces, which are called tokens which are meaningful elements. The big chunks of text which are smaller than large strings can be processed by tokenization and converted into sentences, then at the end these sentences can be tokenized into words.
2. **Stop Word Removal -** This step can improve the performance of the system by reducing the size of text by 20-30% of overall word counts in a specific text document. In English language these words have been defined as conjunctions, prepositions, adverbs.
3. **Stemming -** The most important purpose for stemming process is the words that sharing the exact root or stem will reduce the total number of words that need to process and save time and memory space.

4. **Chunking -** Chunking is a term referring to the process of taking individual pieces of information (chunks) and grouping them into larger units. In our system, we use part of speech tags as chunks and group them by using custom grammar. Grammar is defined with regular expressions.
5. **Marking Unmovable Phrases -** A set of Tregex expressions is used to mark the phrases in an input tree which cannot be answered due to constraints on WH-movement. Each parse tree node subject to a movement constraint is renamed by extracting the node's current label and adding a special prefix marker ("UNMOVABLE-") to its label. Then, in the following step of answer phrase selection, these nodes are skipped so that the system does not generate questions for them.
6. **Generating Question Phrases -** To generate the question phrases, the system annotates the source sentence with a set of high level semantic types using the super sense tagger. For a given answer phrase, the system uses these high-level semantic tags along with the syntactic structure to generate a set of one or more possible question phrases, each of which is used to generate a final question sentence. Recall that answer phrases can be noun phrases (labeled "NP"), prepositional phrases (labeled "PP"), or subordinate clauses (labeled "SBAR").
7. **Generating Answer Phrases -** First step the type of the question has to be identified in order to decide the way of analyzing the sentences. The system reads the questions one by one and parses the questions to the Stanford parser and identifies the initial labels. Then have to invert the subject in the question and also have to remove the question word to convert the question to its predicate format. Once the format is converted to its predicate format can use the cosine similarity checking for all the sentences similar to the converted sentence. The highest matching sentence will be retrieved to generate the answer.
8. **Term Frequency – Inverse Document Frequency (TF-IDF)** - In TF-IDF, it is typically assumed that a good class of distinction occurs if a term has a high frequency in one document, and low frequency in all other documents. Therefore, the weight of the TF-IDF is proportional to the frequency of the word in the current document, and

inversely proportional to the number of occurrences in the other documents. TF-IDF is intended to reflect how relevant a term is in a given document. If the original TF-IDF algorithm is applied to a text feature selection, a large number of rare words may be selected. After which, additional dimensions of features will be needed to represent the document text as well.

3.3 User Input

Firstly the user will input his Name in the startup form and then has to choose the subject as his input. After selection between objective and subjective the user would be entering the respective answers to the generated questions on the screen. Those answers will be reflected and saved as per user for further evaluation of his score. The Question paper would be objective type means one or two word answer or it could be subjective ones with long answer responses from the user.

3.4 Evaluating User Answers

The user answers will be brought and firstly preprocessed using these methods for later evaluation. Then the score would be computed on matching the student answer and the system generated answer for our algorithm and give a particular score on the basis of the degree of match.

1. **Tokenization -** Its important step in information retrieval, the task of this step is splitting large strings made of texts into smaller pieces, which are called tokens which are meaningful elements. The big chunks of text which are smaller than large strings can be processed by tokenization and converted into sentences, then at the end these sentences can be tokenized into words.
2. **Stop Word Removal -** This step can improve the performance of the system by reducing the size of text by 20-30% of overall word counts in a specific text document. In English language these words have been defined as conjunctions, prepositions, adverbs.

3. **Stemming -** The most important purpose for stemming process is the words that sharing the exact root or stem will reduce the total number of words that need to process and save time and memory space.
4. **Term Frequency** - The TF vectors are calculated for the student answers . These vectors then help to calculate similarity with system generated answers using cosine similarity.
5. **Cosine Similarity** - Cosine similarity is the traditional method used to measure the degree of similarity between two vectors, obtained from the cosine angle multiplication value of two vectors, and is often combined with the TF-IDF. Weighted term results are used for the calculation of similarity between the reference answers and the student answers.

3.5 Storing the Results

The result would be stored in the form of the table in the excel data sheet. The following data would be saved for each user:-

1. UserName
2. Subject Name
3. Marks Scored
4. Date of Examination

3.6 Process

In this we present the computerized examination system, which is divided into two main stages: the preparation stage and the similarity scoring stage. First, the preparation stage aims to prepare a reference answer in terms of the TF-IDF vectors, consisting of two processes; preprocessing and TF-IDF calculation. Secondly, the similarity scoring stage aims to evaluate the student answers, and determine the student scores. There are three processes in this stage; preprocessing, TF calculation, and similarity and scoring computation.

A. Preparation stage : The aim of this stage is to prepare the reference answer, which is a short Thai sentence, and transform it into vectors; namely, the TF-IDF vectors. The preparation stage consists of the following processes:

1) Preprocessing: Since the reference answer is in English, which is segmented language, the reference answer (sentence) is divided into a sequence of words, and any stop words are removed. The longest matching algorithm and Lexitron® dictionary are applied to segment the answer into words. These words are operated to eliminate stop words.

2) TF-IDF calculation: After the preprocessing process, word frequency is calculated in terms of TF-IDF. The TF-IDF of the reference answer is stored in the reference answer database.

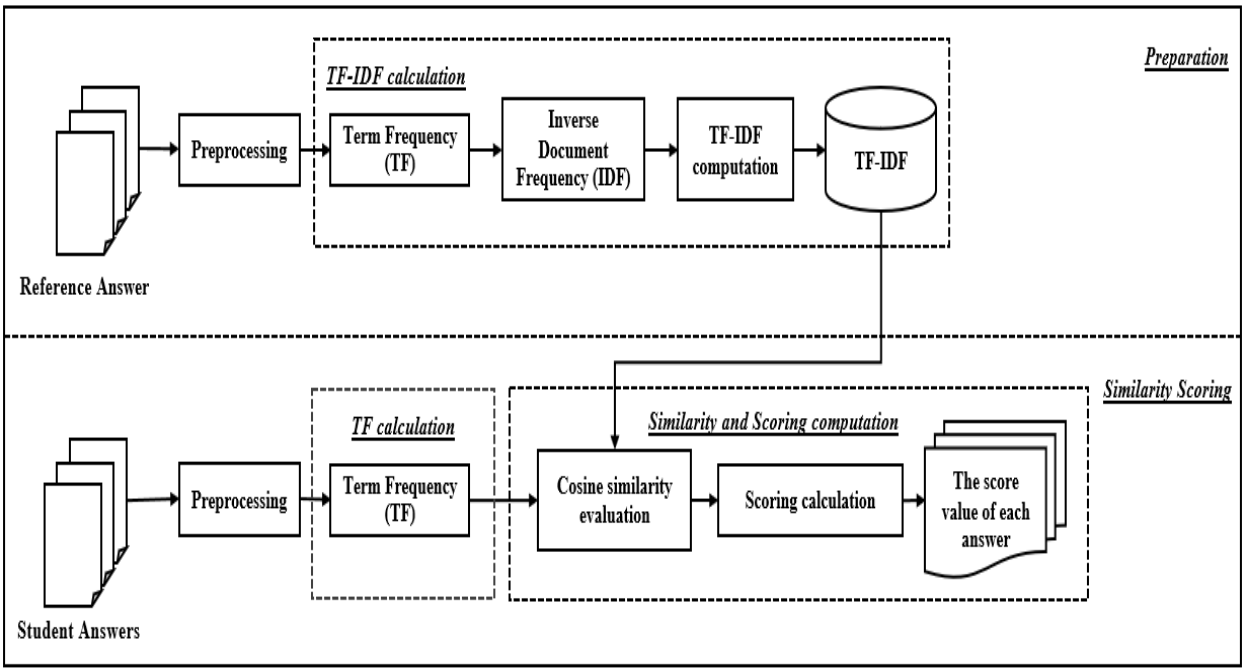
B. Similarity scoring stage : In the similarity scoring stage, a student answer is also evaluated in a preprocessing process, along with the consideration of the word‟s synonym, which is a word, or phrase that means exactly or nearly the same as another word or phrase in the same language, and then transformed into TF vectors. The student vectors are then compared with the reference vectors, using the cosine similarity method, stored in the database, to evaluate the score of the student answer.

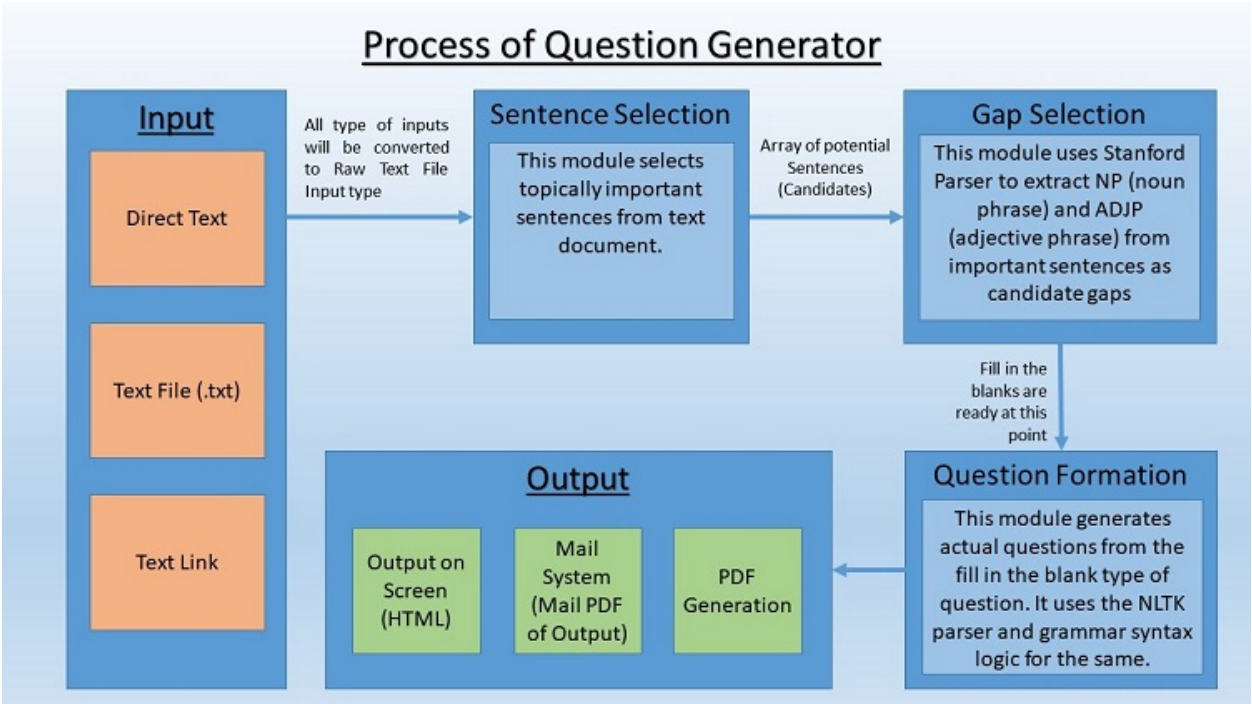
1) Preprocessing: The student‟s answers are operated similarly to the preparation stage, utilizing both word segmentation and stop word elimination. Additionally, the sequence of words is considered to determine the synonym.

2) TF calculation: A sequence of words, after undergoing the preprocessing process, is calculated for frequency in terms of TF vectors.

3) Similarity and Scoring computation: In this process, cosine similarity is applied to evaluate the similarity of the reference answer and the student answer. The TF-IDF of the reference answer is queried from the database, and compared with the TF of the student answer. The cosine similarity of the two answers ranged from 0 to 1. A value close to 1 means that the student‟s answer is very similar to the reference answers, whereas a value close to 0 implies that

the student's answer is distant from the reference answer. Then, cosine similarity is used to calculate the actual score of the student‟s answer with the maximum score possible in each question.





3.7 Requirements

3.7.1 Software Requirements

1. HTML5 - Use to design web pages.
2. CSS3 - Used to style web pages
3. Bootstrap 4 - Make web app as mobile friendly
4. JavaScript - Helps in creating web pages
5. Flask - Frontend Framework
6. Python 3.6 - Backend Language
7. NLTK Library - Helps in NLP like Tokenization
8. TextBlob Library - Helps in NLP
9. Pandas and Numpy Libraries - Supporting Python libraries
10. MS-Excel - Database

3.7.2 Hardware Requirements

- Working laptop
- 160Mb Disk Space
- 2 GB RAM
- Windows/Linux

4. Experimental Results

In order to evaluate the performance of the proposed system, we conducted the following experiment: We took data of the “Database Management System and Database Design” course to prepare a question dataset and asked our user to answer each question. We then asked 10 friends to join the experiment by giving their answers to the questions. The mentor then evaluated the answers, and provided a score for their answers, which were then compared with scores obtained from the proposed system in both methods; namely with and without the synonym, in which to compare the similarity with the reference answer.

In an existing system to evaluate the students performance NLP process is applied. In study we developed a tool to calculate the results using Natural Language Processing (NLP). The result is obtained through the NLP algorithm and the result is produced as a final result. These algorithms are used successfully and produce efficient results. The system generated results are compared with evaluation done by the faculty members. We found that there is no vast difference between the system generated results and evaluation done by the faculty member. The accuracy of the experiment is 79% .

5. Conclusion

In future work, Deep learning method can be used as the advanced technology in the process of handwriting recognition of the student answer sheet. To improve the efficiency of the proposed system, we intend to integrate semantic and context analysis into future work. The future seems bright. After successful implementation of this work ,in future it will lead to Question Generation from entire document.In that case it would be required to first find out those sentences from a document on whom questions can be formed.After that the current system algorithm with minor improvement can be deployed.The ultimate system will be able to independently handle any pdf or word or any other type of text file ,analyze it and find important sentences for QG.Further from those sentences a variety of questions could be formed with minimum inaccuracy.

The main and important problem that we have faced during this research was accuracy of the results. Since we are dealing with human language, there may be a lack of accuracy due to the complexity of the user input. Accuracy can be improved in the future. Checking of plagiarism is also added to future work.

We have created it with respect to English Language only. It could further be extended to different languages like Hindi, Tamil, Malayalam, Thai, French, Russian, etc.

The speech to text feature can also be added so the person just has to speak and our system converts it to text answer and evaluates the results.

6. References

[1] **Automatic Thai Subjective Examination using Cosine Similarity** Pongsakorn Saipech and Pusadee Seresangtakul Natural Language and Speech Processing Laboratory Department of Computer Science, Khon Kaen University Khon Kaen, Thailand, 40002

[2] International Journal of New Technology and Research (IJNTR) ISSN: 2454-4116, Volume-5, Issue-10, October 2019 **An E-Exam Management System under E-Network Management Course** Shaimaa H. Shaker, Noor M. Jaafar.

[3] International Journal for Innovations in Engineering, Science and Management Volume 3, Issue 4, April 2015 27 All Rights Reserved © 2015 IJIESM **Automatic Question Generation from Text** Himanshu Jethwani (Student M.Tech) Mohd Shahid Husain (CSE Assistant Professor Dept. Of CSE) Mohd Akbar (Assistant Professor Dept. Of CSE)

[4] 2017 IJCRT | Volume 5, Issue 4 December 2017 | ISSN: 2320-2882 IJCRT1704424 International Journal of Creative Research Thoughts (IJCRT) 3168 **EVALUATING STUDENTS' DESCRIPTIVE ANSWERS USING NATURAL LANGUAGE PROCESSING AND ARTIFICIAL NEURAL NETWORKS** 1V. Lakshmi, 2Dr. V. Ramesh 1Research Scholar, Department of Computer Science and Applications 2Assistant Professor, Department of CSA, Sri Chandrasekharendra Saraswathi Viswa Maha Vidyala, Kanchipuram, India.

[5] International Journal of Scientific and Research Publications, Volume 7, Issue 11, November 2017 ISSN 2250-3153 **Automatic Question and Answer Generation from Course Materials.** A.S.M Nibras, M.F.F Mohamed, I.S.M Arham, A.M.M Mafaris, M.P.A.W Gamage Sri Lanka Institute of Information Technology.

[6] Proceedings of The 9th International Natural Language Generation conference, pages 51–60, Edinburgh, UK, September 5-8-2016 Association for Computational Linguistics Infusing NLU into Automatic Question Generation Karen Mazidi and Paul Tarau Department of Computer Science and Engineering University of North Texas, Denton TX 76207 USA

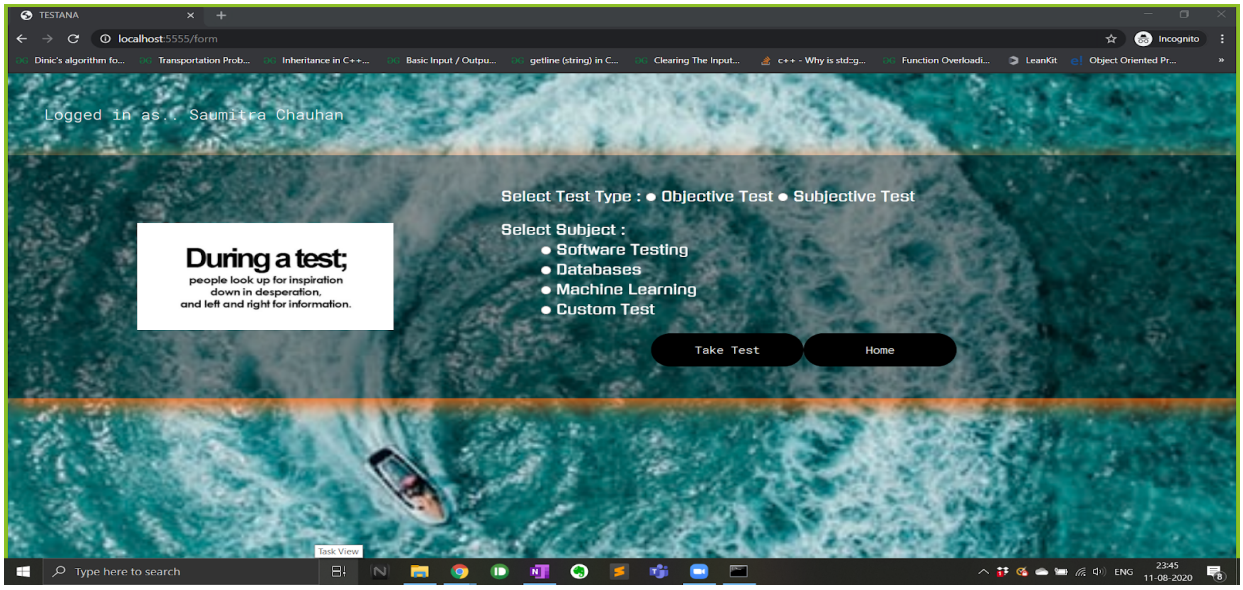
[7] International Journal of Artificial Intelligence in Education (2020) 30:121–204 Published online: 21 November 2019 A Systematic Review of Automatic Question Generation for Educational Purposes Ghader Kurdi1 · Jared Leo1 · Bijan Parsia1 · Uli Sattler1 · Salam Al-Emari2

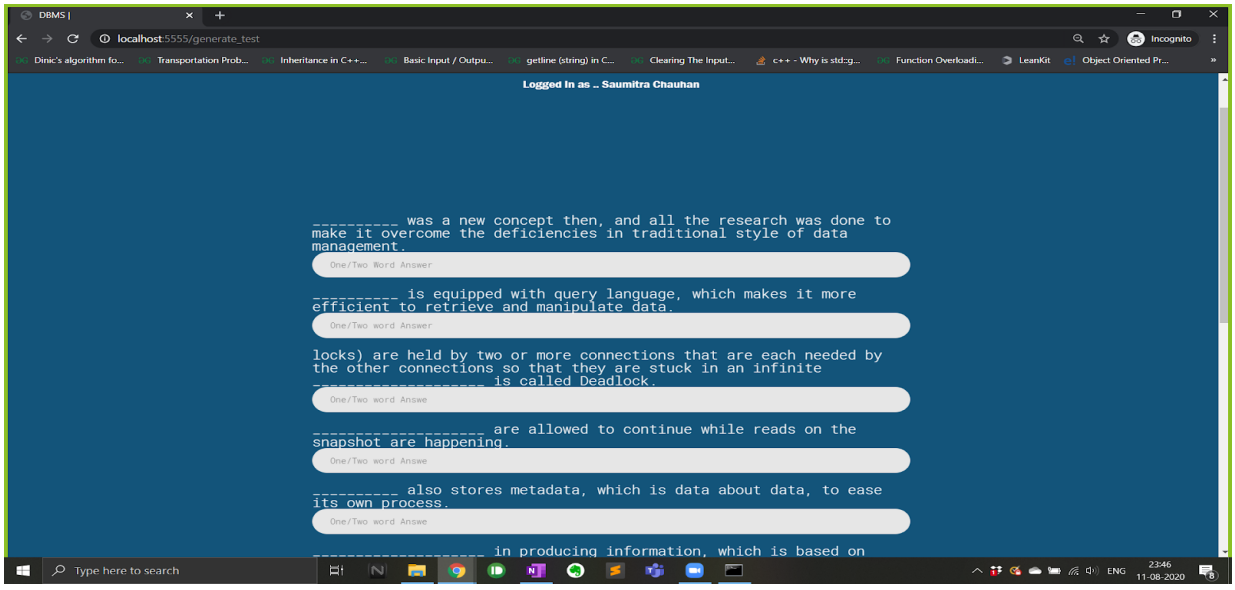
[8] International Journal of Recent Technology and Engineering (IJRTE) ISSN: 2277-3878, Volume-7, Issue-6, March 2019 Retrieval Number: F2253037619/19©BEIESP 1712 Published By: Blue Eyes Intelligence Engineering & Sciences Publication Deep learning for Short Answer Scoring Surya K, Ekansh Gayakwad, Nallakaruppan, M.K

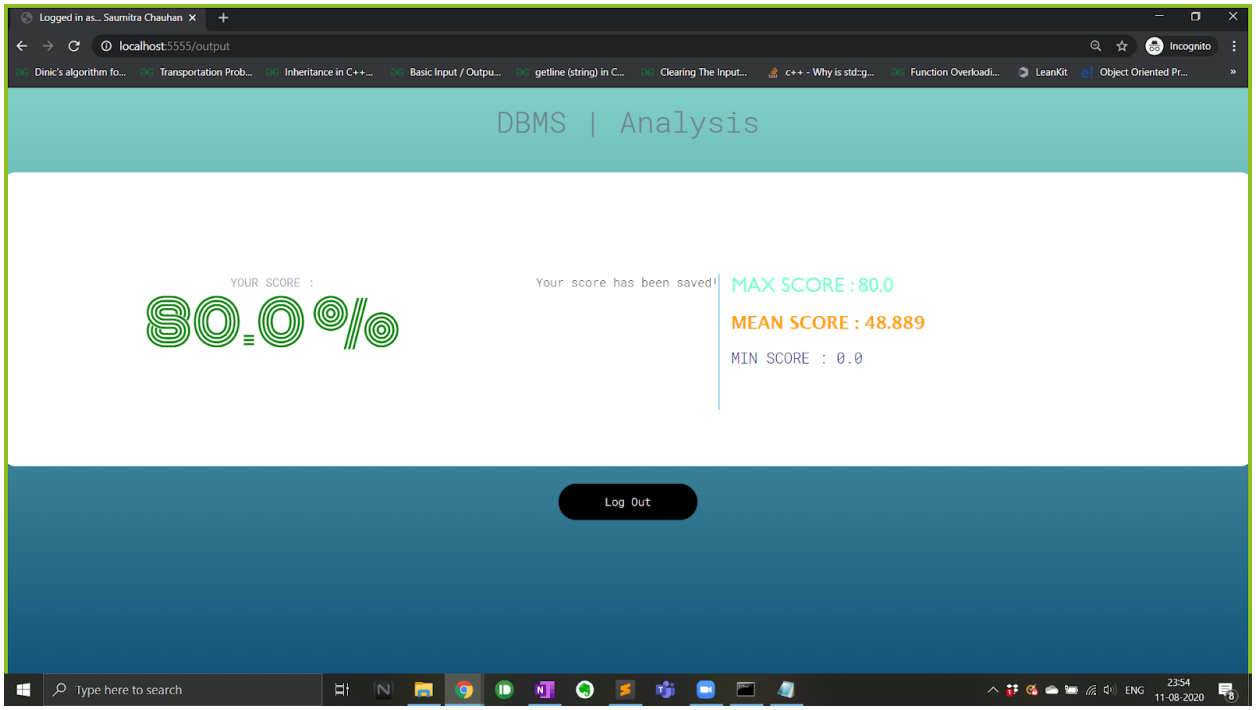
[9] INCITEST 2019 IOP Conf. Series: Materials Science and Engineering 662 (2019) 022111 IOP Publishing doi:10.1088/1757-899X/662/2/022111 Measuring the Level of Plagiarism of Thesis using Vector Space Model and Cosine Similarity Methods I Indriyanto1*, I D Sumitra2 1Pascasarjana Departemen Sistem Informasi, Universitas Komputer Indonesia, Bandung, indonesia 2Information Management Department, Universitas Komputer Indonesia, Bandung, Indonesia

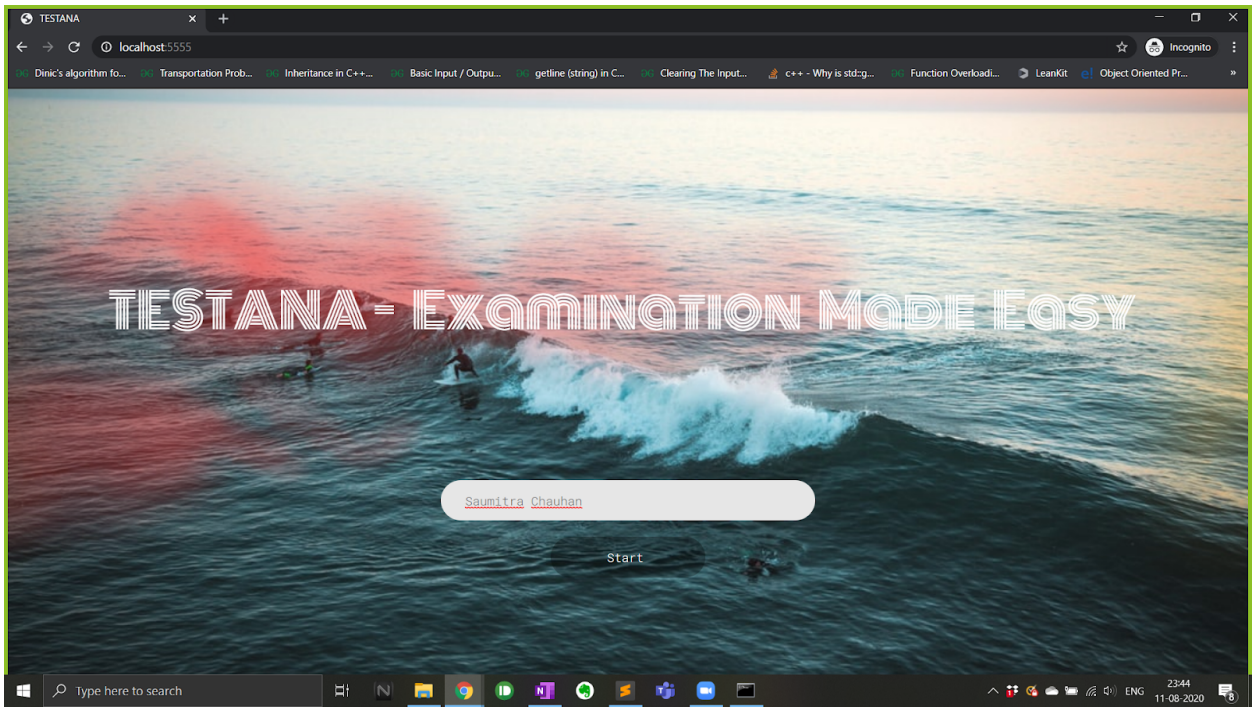
Annexure

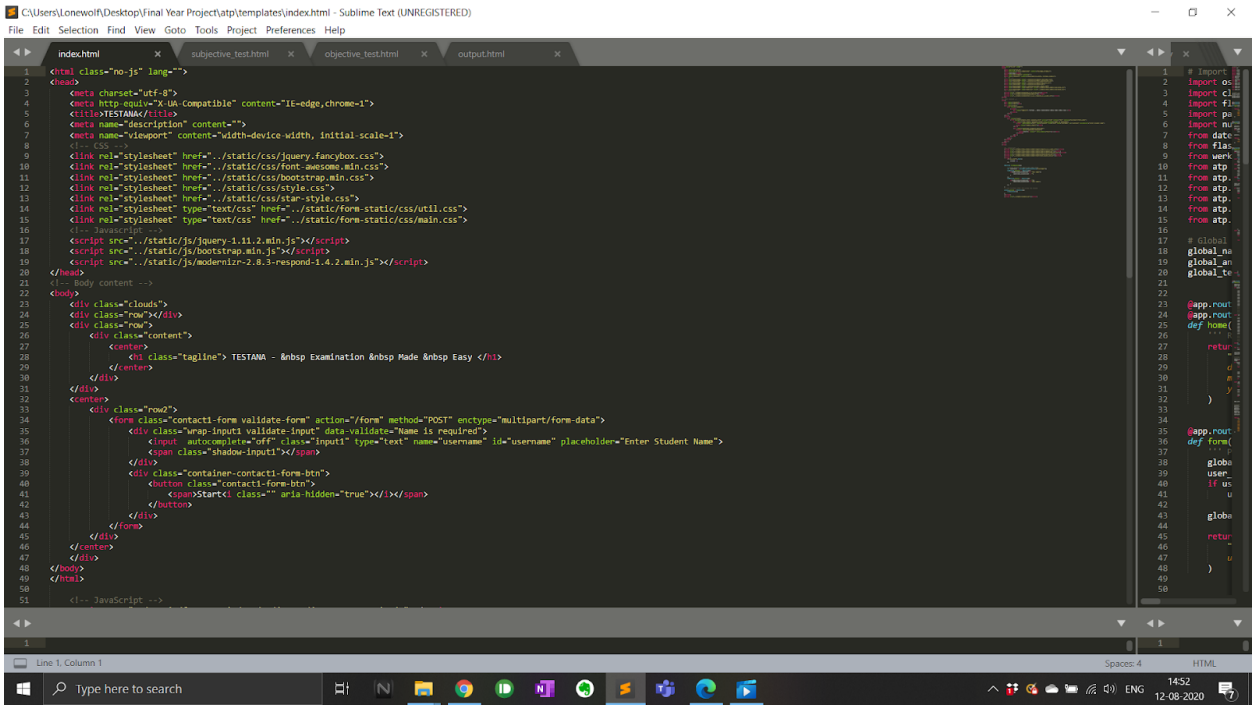
Frontend Module






Logged in as... Saumitra Chauhan
localhost:5555/output
Incognito
DBMS | Analysis
YOUR SCORE :
80.0%
Your score has been saved!
MAX SCORE : 80.0
MEAN SCORE : 48.889
MIN SCORE : 0.0
Log Out
Type here to search
ENG
23:54
11-08-2020


TESTANA
localhost:5555
Incognito
TESTANA - Examination Made Easy
Saumitra Chauhan
Start
Type here to search
ENG
23:44
11-08-2020


C:\Users\Lonewolf\Desktop\Final Year Project\atp\templates\index.html - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
index.html
subjective_test.html
objective_test.html
output.html
<html class="no-js" lang="">
<head>
<meta charset="utf-8">
<meta http-equiv="X-UA-Compatible" content="IE=edge,chrome=1">
<title>TESTANA</title>
<meta name="description" content="">
<meta name="viewport" content="width=device-width, initial-scale=1">
<!-- CSS -->
<link rel="stylesheet" href="../static/css/jquery.fancybox.css">
<link rel="stylesheet" href="../static/css/font-awesome.min.css">
<link rel="stylesheet" href="../static/css/bootstrap.min.css">
<link rel="stylesheet" href="../static/css/style.css">
<link rel="stylesheet" href="../static/css/star-style.css">
<link rel="stylesheet" type="text/css" href="../static/form-static/css/util.css">
<link rel="stylesheet" type="text/css" href="../static/form-static/css/main.css">
<!-- Javascript -->
<script src="../static/js/jquery-1.11.2.min.js"></script>
<script src="../static/js/bootstrap.min.js"></script>
<script src="../static/js/modernizr-2.8.3-respond-1.4.2.min.js"></script>
</head>
<!-- Body content -->
<body>

<h1 class="tagline"> TESTANA -   Examination   Made   Easy </h1>

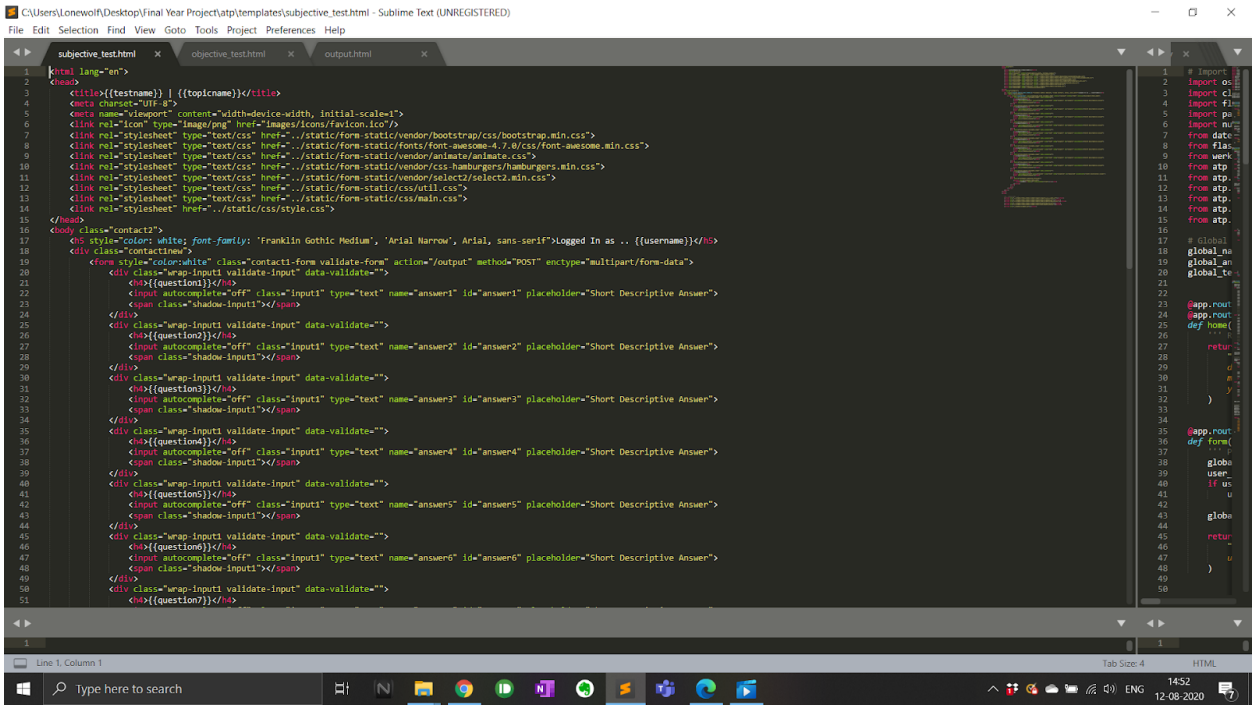
<form class="contact1-form validate-form" action="/form" method="POST" enctype="multipart/form-data">

<input autocomplete="off" class="input1" type="text" name="username" id="username" placeholder="Enter Student Name">

<button class="contact1-form-btn">
Start
</button>

</form>

</body>
</html>
<!-- JavaScript -->
Line 1, Column 1
Spaces: 4
HTML
Type here to search
ENG
14:52
12-08-2020


C:\Users\Lonewolf\Desktop\Final Year Project\atp\templates\subjective_test.html - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
subjective_test.html
objective_test.html
output.html
<html lang="en">
<head>
<title>{{testname}} | {{topicname}}</title>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="icon" type="image/png" href="images/icons/favicon.ico"/>
<link rel="stylesheet" type="text/css" href="../static/form-static/vendor/bootstrap/css/bootstrap.min.css">
<link rel="stylesheet" type="text/css" href="../static/form-static/fonts/font-awesome-4.7.0/css/font-awesome.min.css">
<link rel="stylesheet" type="text/css" href="../static/form-static/vendor/animate/animate.css">
<link rel="stylesheet" type="text/css" href="../static/form-static/vendor/css-hamburgers/hamburgers.min.css">
<link rel="stylesheet" type="text/css" href="../static/form-static/vendor/select2/select2.min.css">
<link rel="stylesheet" type="text/css" href="../static/form-static/css/util.css">
<link rel="stylesheet" type="text/css" href="../static/form-static/css/main.css">
<link rel="stylesheet" href="../static/css/style.css">
</head>
<body class="contact2">
<h5 style="color: white; font-family: 'Franklin Gothic Medium', 'Arial Narrow', Arial, sans-serif">Logged In as .. {{username}}</h5>

<form style="color:white" class="contact1-form validate-form" action="/output" method="POST" enctype="multipart/form-data">

<h4>{{question1}}</h4>
<input autocomplete="off" class="input1" type="text" name="answer1" id="answer1" placeholder="Short Descriptive Answer">

<h4>{{question2}}</h4>
<input autocomplete="off" class="input1" type="text" name="answer2" id="answer2" placeholder="Short Descriptive Answer">

<h4>{{question3}}</h4>
<input autocomplete="off" class="input1" type="text" name="answer3" id="answer3" placeholder="Short Descriptive Answer">

<h4>{{question4}}</h4>
<input autocomplete="off" class="input1" type="text" name="answer4" id="answer4" placeholder="Short Descriptive Answer">

<h4>{{question5}}</h4>
<input autocomplete="off" class="input1" type="text" name="answer5" id="answer5" placeholder="Short Descriptive Answer">

<h4>{{question6}}</h4>
<input autocomplete="off" class="input1" type="text" name="answer6" id="answer6" placeholder="Short Descriptive Answer">

<h4>{{question7}}</h4>
Line 1, Column 1
Tab Size: 4
HTML
Type here to search
ENG
14:52
12-08-2020

C:\Users\Lonewolf\Desktop\Final Year Project\atp\templates\objective_test.html - Sublime Text (UNREGISTERED)

File Edit Selection Find View Goto Tools Project Preferences Help

objective_test.html | output.html

```
<html lang="en">
<head>
    <title>{{testname}} | {{topicname}}</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="icon" type="image/png" href="images/icons/favicon.ico"/>
    <link rel="stylesheet" type="text/css" href="../static/form-static/vendor/bootstrap/css/bootstrap.min.css">
    <link rel="stylesheet" type="text/css" href="../static/form-static/fonts/font-awesome-4.7.0/css/font-awesome.min.css">
    <link rel="stylesheet" type="text/css" href="../static/form-static/vendor/animate/animate.css">
    <link rel="stylesheet" type="text/css" href="../static/form-static/vendor/css-hamburgers/hamburgers.min.css">
    <link rel="stylesheet" type="text/css" href="../static/form-static/vendor/select2/select2.min.css">
    <link rel="stylesheet" type="text/css" href="../static/form-static/css/util.css">
    <link rel="stylesheet" type="text/css" href="../static/form-static/css/main.css">
    <link rel="stylesheet" href="../static/css/style.css">
</head>
<body class="contact2">
    <h5 style="_color: White; font-family: 'Franklin Gothic Medium', 'Arial Narrow', Arial, sans-serif">Logged In as .. {{username}}</h5>
    <div class="contactinew">
        <form style="color:white" class="contact1-form validate-form" action="/output" method="POST" enctype="multipart/form-data">
            <div class="wrap-input1 validate-input" data-validate="">
                <h4>{{question1}}</h4>
                <input autocomplete="off" class="input1" type="text" name="answer1" id="answer1" placeholder="One/Two Word Answer">
                <span class="shadow-input1"></span>
            </div>
            <div class="wrap-input1 validate-input" data-validate="">
                <h4>{{question2}}</h4>
                <input autocomplete="off" class="input1" type="text" name="answer2" id="answer2" placeholder="One/Two word Answer">
                <span class="shadow-input1"></span>
            </div>
            <div class="wrap-input1 validate-input" data-validate="">
                <h4>{{question3}}</h4>
                <input autocomplete="off" class="input1" type="text" name="answer3" id="answer3" placeholder="One/Two word Answe">
                <span class="shadow-input1"></span>
            </div>
            <div class="wrap-input1 validate-input" data-validate="">
                <h4>{{question4}}</h4>
                <input autocomplete="off" class="input1" type="text" name="answer4" id="answer4" placeholder="One/Two word Answe">
                <span class="shadow-input1"></span>
            </div>
            <div class="wrap-input1 validate-input" data-validate="">
                <h4>{{question5}}</h4>
                <input autocomplete="off" class="input1" type="text" name="answer5" id="answer5" placeholder="One/Two word Answe">
                <span class="shadow-input1"></span>
            </div>
            <div class="wrap-input1 validate-input" data-validate="">
                <h4>{{question6}}</h4>
                <input autocomplete="off" class="input1" type="text" name="answer6" id="answer6" placeholder="One/Two Word Answer">
                <span class="shadow-input1"></span>
            </div>
            <div class="wrap-input1 validate-input" data-validate="">
                <h4>{{question7}}</h4>
```

```
# Import
import os
import cla
import fla
import pa
import nu
from date
from fla
from werk
from atp
from atp.
from atp.
from atp.
from atp.

# Global
global_na
global_an
global_te

@app.rout
@app.rout
def home(
    '''
    retur
        "
        d
        m
        y
    )

@app.rout
def form(
    '''
    globa
    user_
    if us
        u

    globa

    retur
        "
        u
    )
```

Line 1, Column 1 — Tab Size: 4 — HTML

Type here to search — ENG — 14:52 — 12-08-2020

C:\Users\Lonewolf\Desktop\Final Year Project\atp\templates\form.html - Sublime Text (UNREGISTERED)

File Edit Selection Find View Goto Tools Project Preferences Help

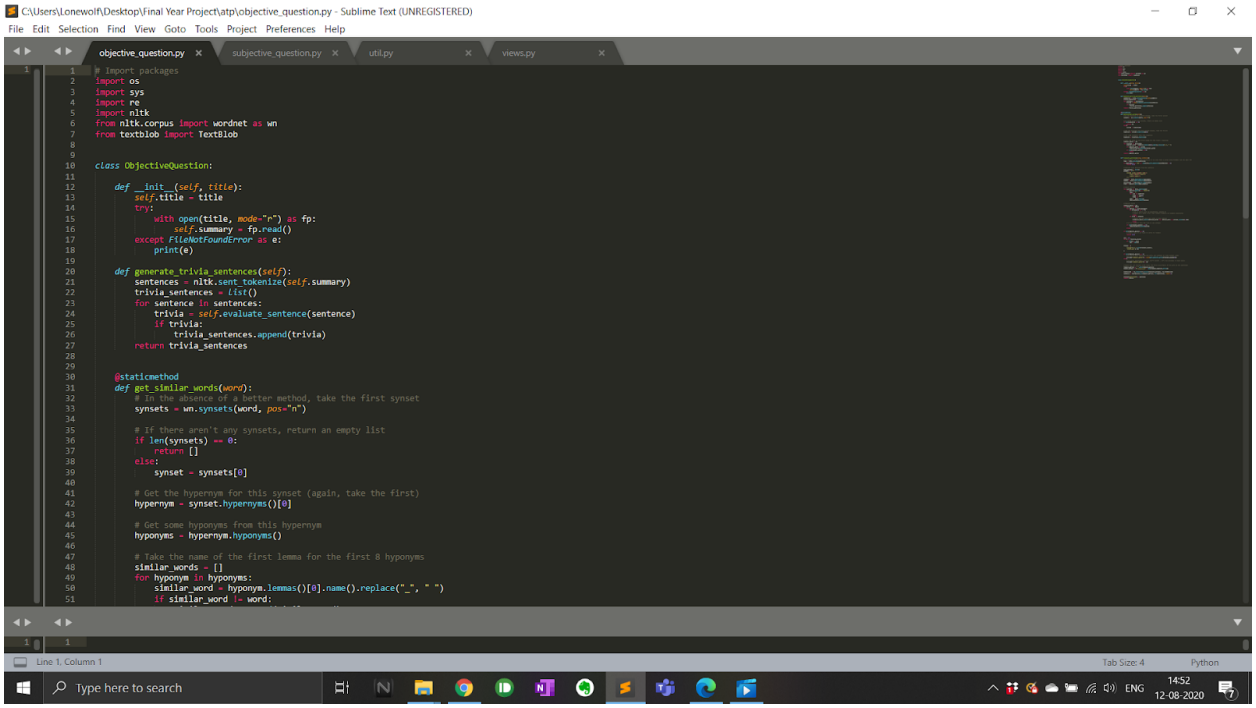
form.html

```
<html lang="en">
<head>
    <title>TESTANA</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="icon" type="image/png" href="images/icons/favicon.ico"/>
    <link rel="stylesheet" type="text/css" href="../static/form-static/vendor/bootstrap/css/bootstrap.min.css">
    <link rel="stylesheet" type="text/css" href="../static/form-static/fonts/font-awesome-4.7.0/css/font-awesome.min.css">
    <link rel="stylesheet" type="text/css" href="../static/form-static/vendor/animate/animate.css">
    <link rel="stylesheet" type="text/css" href="../static/form-static/vendor/css-hamburgers/hamburgers.min.css">
    <link rel="stylesheet" type="text/css" href="../static/form-static/vendor/select2/select2.min.css">
    <link rel="stylesheet" type="text/css" href="../static/form-static/css/util.css">
    <link rel="stylesheet" type="text/css" href="../static/form-static/css/main.css">
    <link rel="stylesheet" href="../static/css/style.css">
</head>
<body class="body2">
    <div class="head-space">
        <h5>Logged in as.. {{username}}</h5>
    </div>
    <div class="contact1 mywwnfont" style="box-shadow: 5px 8px 32px rgb(255, 94, 0);">
        <div class="contact1-pic js-tilt" data-tilt>
            <img src="../static/images/quote2.png" alt="IMG">
        </div>
        <form style="color:white" class="contact1-form validate-form" action="/generate_test" method="POST" enctype="multipart/form-data">
<!--            <div class="wrap-input1 validate-input" data-validate="Subject Name is required">
                <input autocomplete="off" class="input1" type="text" name="subjectname" id="subjectname" placeholder="Subject Name">
                <span class="shadow-input1"></span>
            </div> -->
            <div class="wrap-input1 validate-input" data-validate="">
                Select Test Type :
                <span>
                    <input type="radio" name="test_id" value="obj"> Objective Test
                    <input type="radio" name="test_id" value="subj"> Subjective Test
                </span>
            </div>
            <div class="wrap-input1 validate-input" data-validate="">
                Select Subject : <br>
                <blockquote style="padding-Left: 50px">
                    <input type="radio" name="subject_id" value="os" id="myCheck4" onclick="myFunction2()"> Software Testing <br>
                    <input type="radio" name="subject_id" value="db" id="myCheck3" onclick="myFunction2()"> Databases <br>
                    <input type="radio" name="subject_id" value="ml" id="myCheck2" onclick="myFunction2()"> Machine Learning <br>
                    <input type="radio" name="subject_id" value="custom" id="myCheck" onclick="myFunction()"> Custom Test
                </blockquote>
            </div>
            <div id="file_br" class="wrap-input1 validate-input" data-validate="Test file is required" style="display:none">
                <input type="file" name="file" />
            </div>
            <div class="container-contact1-form-btn">
                <button class="contact1-form-newbtn">
                    <span>Take Test<i class="" aria-hidden="true"></i></span>
                </button>
```

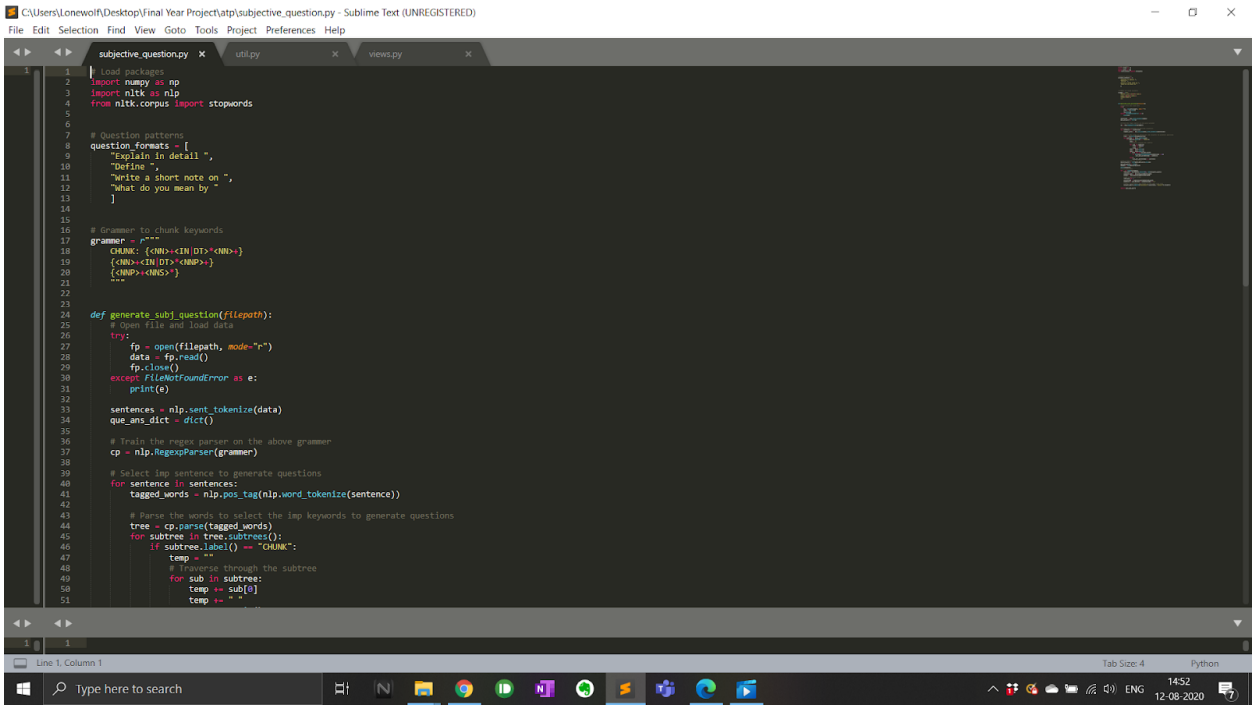
Line 1, Column 1 — Tab Size: 4 — HTML

Type here to search — ENG — 14:51 — 12-08-2020

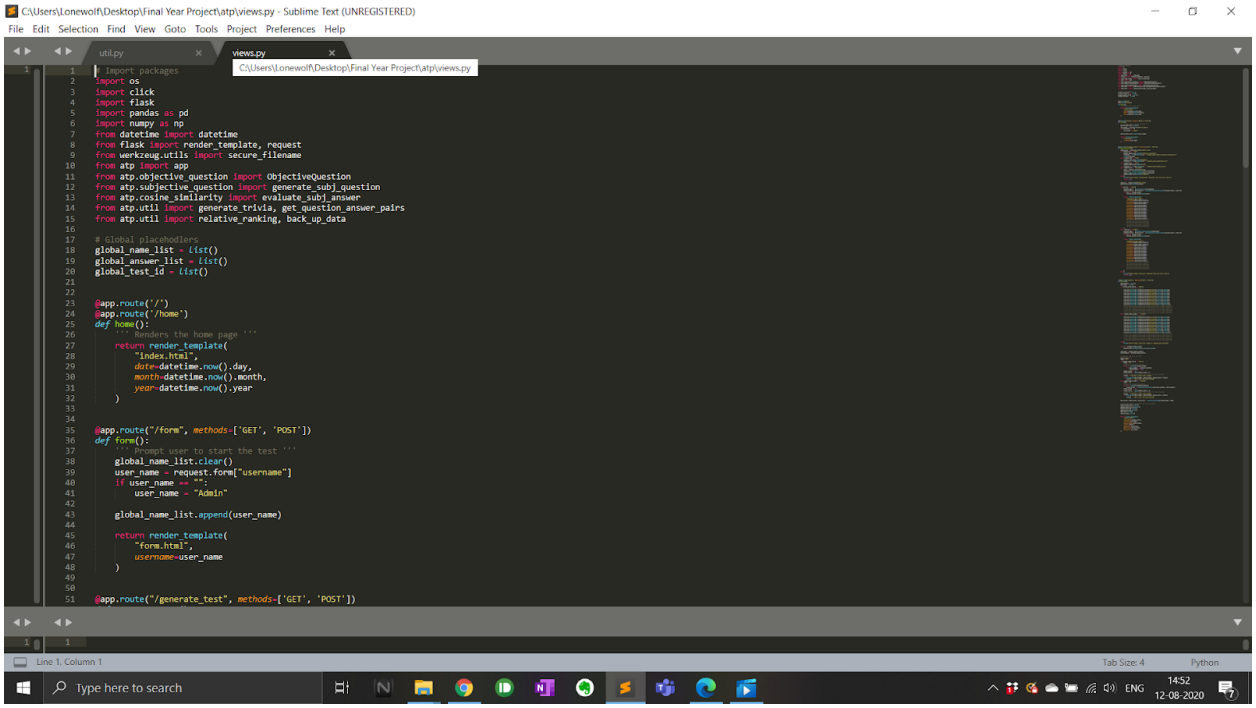
Backend Module

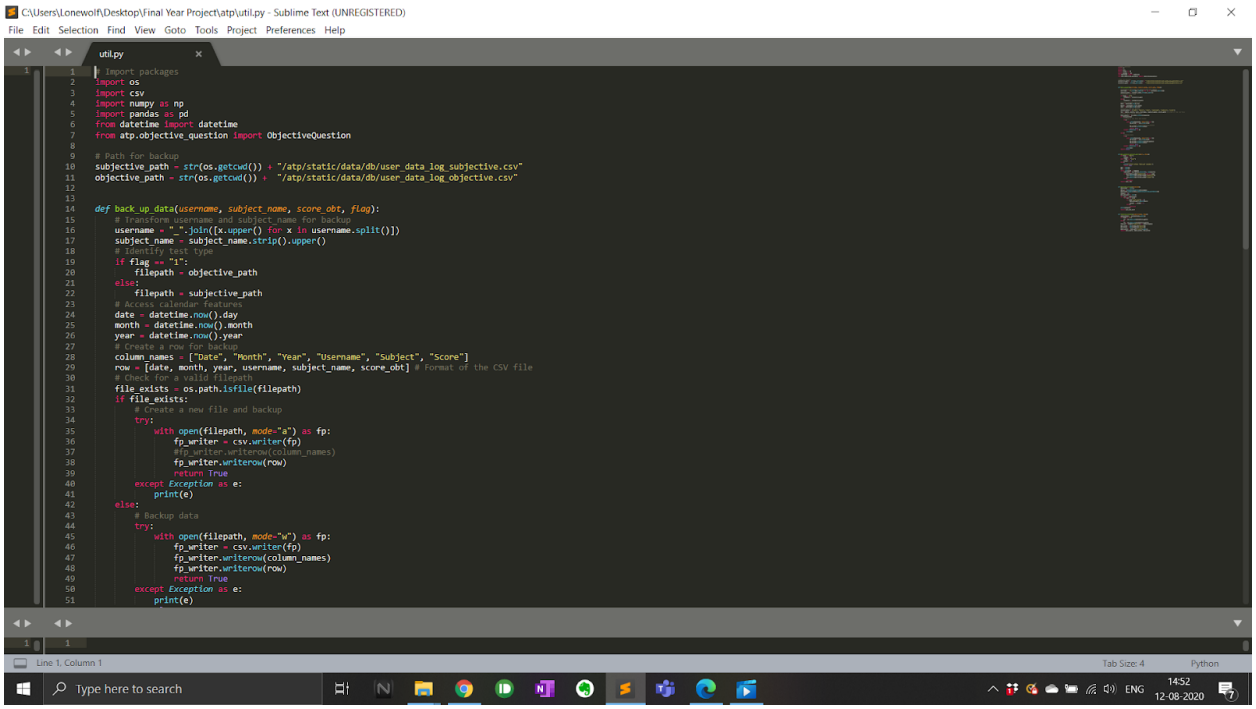


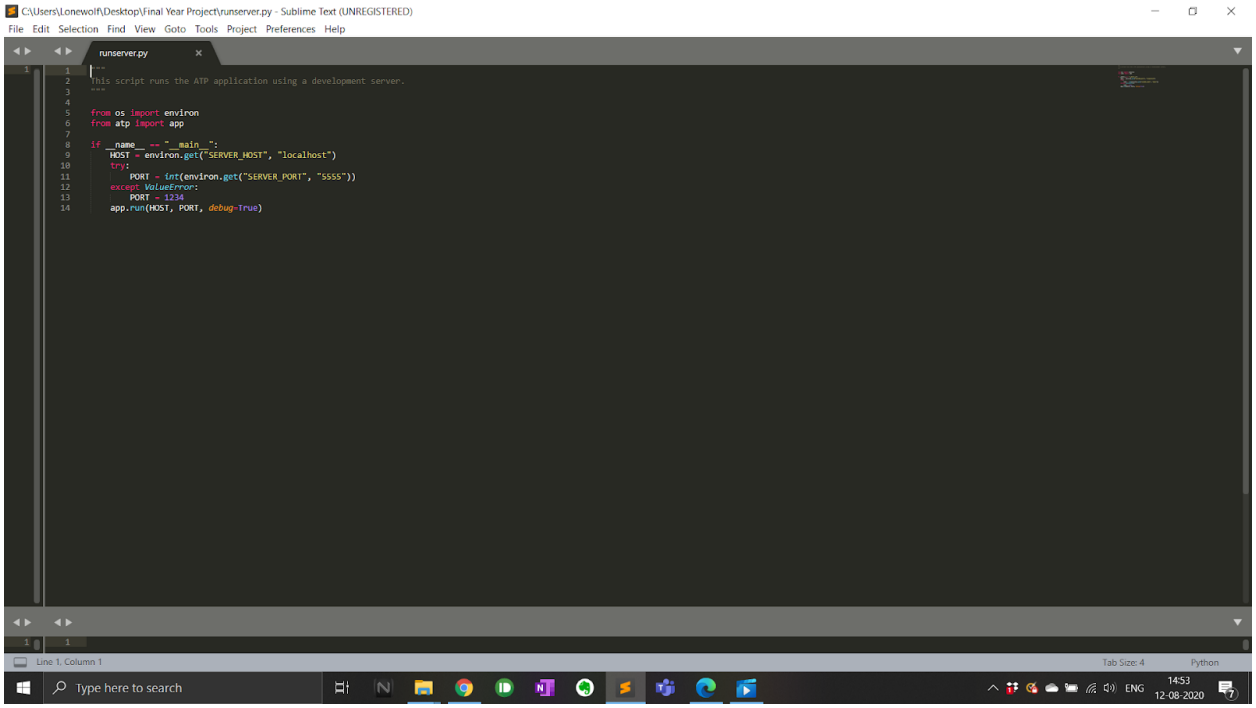

```python
# Import packages
import os
import sys
import re
import nltk
from nltk.corpus import wordnet as wn
from textblob import TextBlob


class ObjectiveQuestion:

    def __init__(self, title):
        self.title = title
        try:
            with open(title, mode="r") as fp:
                self.summary = fp.read()
        except FileNotFoundError as e:
            print(e)

    def generate_trivia_sentences(self):
        sentences = nltk.sent_tokenize(self.summary)
        trivia_sentences = list()
        for sentence in sentences:
            trivia = self.evaluate_sentence(sentence)
            if trivia:
                trivia_sentences.append(trivia)
        return trivia_sentences


    @staticmethod
    def get_similar_words(word):
        # To the advance of a better method, take the first synset
        synsets = wn.synsets(word, pos="n")

        # If there aren't any synsets, return an empty list
        if len(synsets) == 0:
            return []
        else:
            synset = synsets[0]

        # Get the hypernym for this synset (again, take the first)
        hypernym = synset.hypernyms()[0]

        # Get some hyponyms from this hypernym
        hyponyms = hypernym.hyponyms()

        # Take the name of the first lemma for the first 8 hyponyms
        similar_words = []
        for hyponym in hyponyms:
            similar_word = hyponym.lemmas()[0].name().replace("_", " ")
            if similar_word != word:
```

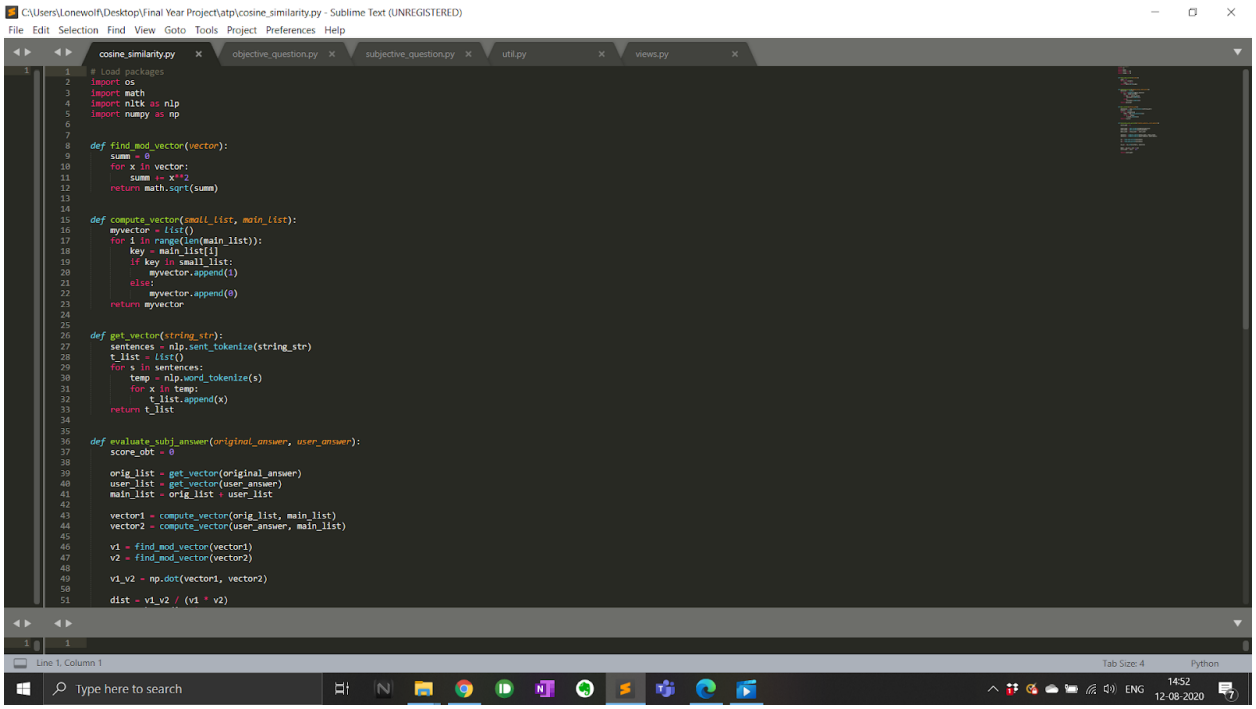



```python
# Load packages
import numpy as np
import nltk as nlp
from nltk.corpus import stopwords


# Question patterns
question_formats = [
    "Explain in detail ",
    "Define ",
    "Write a short note on ",
    "What do you mean by "
    ]


# Grammar to chunk keywords
grammer = r"""
    CHUNK: {<NN>+<IN|DT>*<NN>+}
    {<NN>+<IN|DT>*<NNP>+}
    {<NNP>+<NNS>*}
    """


def generate_subj_question(filepath):
    # Open file and load data
    try:
        fp = open(filepath, mode="r")
        data = fp.read()
        fp.close()
    except FileNotFoundError as e:
        print(e)

    sentences = nlp.sent_tokenize(data)
    que_ans_dict = dict()

    # Train the regex parser on the above grammer
    cp = nlp.RegexpParser(grammer)

    # Select imp sentence to generate questions
    for sentence in sentences:
        tagged_words = nlp.pos_tag(nlp.word_tokenize(sentence))

        # Parse the words to select the imp keywords to generate questions
        tree = cp.parse(tagged_words)
        for subtree in tree.subtrees():
            if subtree.label() == "CHUNK":
                temp = ""
                # Traverse through the subtree
                for sub in subtree:
                    temp += sub[0]
                    temp += " "
```


C:\Users\Lonewolf\Desktop\Final Year Project\atp\views.py - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
util.py
views.py
C:\Users\Lonewolf\Desktop\Final Year Project\atp\views.py
# Import packages
import os
import click
import flask
import pandas as pd
import numpy as np
from datetime import datetime
from flask import render_template, request
from werkzeug.utils import secure_filename
from atp import app
from atp.objective_question import ObjectiveQuestion
from atp.subjective_question import generate_subj_question
from atp.cosine_similarity import evaluate_subj_answer
from atp.util import generate_trivia, get_question_answer_pairs
from atp.util import relative_ranking, back_up_data
# Global placeholders
global_name_list = list()
global_answer_list = list()
global_test_id = list()
@app.route('/')
@app.route('/home')
def home():
''' Renders the home page '''
return render_template(
"index.html",
date=datetime.now().day,
month=datetime.now().month,
year=datetime.now().year
)
@app.route("/form", methods=['GET', 'POST'])
def form():
''' Prompt user to start the test '''
global_name_list.clear()
user_name = request.form["username"]
if user_name == "":
user_name = "Admin"
global_name_list.append(user_name)
return render_template(
"form.html",
username=user_name
)
@app.route("/generate_test", methods=['GET', 'POST'])
Line 1, Column 1
Tab Size: 4
Python
Type here to search
ENG
14:52
12-08-2020


C:\Users\Lonewolf\Desktop\Final Year Project\atp\util.py - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
util.py
# Import packages
import os
import csv
import numpy as np
import pandas as pd
from datetime import datetime
from atp.objective_question import ObjectiveQuestion
# Path for backup
subjective_path = str(os.getcwd()) + "/atp/static/data/db/user_data_log_subjective.csv"
objective_path = str(os.getcwd()) + "/atp/static/data/db/user_data_log_objective.csv"
def back_up_data(username, subject_name, score_obt, flag):
# Transform username and subject_name for backup
username = "_".join([x.upper() for x in username.split()])
subject_name = subject_name.strip().upper()
# Identify test type
if flag == "1":
filepath = objective_path
else:
filepath = subjective_path
# Access calendar features
date = datetime.now().day
month = datetime.now().month
year = datetime.now().year
# Create a row for backup
column_names = ["Date", "Month", "Year", "Username", "Subject", "Score"]
row = [date, month, year, username, subject_name, score_obt] # Format of the CSV file
# Check for a valid filepath
file_exists = os.path.isfile(filepath)
if file_exists:
# Create a new file and backup
try:
with open(filepath, mode="a") as fp:
fp_writer = csv.writer(fp)
#fp_writer.writerow(column_names)
fp_writer.writerow(row)
return True
except Exception as e:
print(e)
else:
# Backup data
try:
with open(filepath, mode="w") as fp:
fp_writer = csv.writer(fp)
fp_writer.writerow(column_names)
fp_writer.writerow(row)
return True
except Exception as e:
print(e)
Line 1, Column 1
Tab Size: 4
Python
Type here to search
ENG
14:52
12-08-2020


C:\Users\Lonewolf\Desktop\Final Year Project\runserver.py - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
runserver.py
"""
This script runs the ATP application using a development server.
"""
from os import environ
from atp import app
if __name__ == "__main__":
HOST = environ.get("SERVER_HOST", "localhost")
try:
PORT = int(environ.get("SERVER_PORT", "5555"))
except ValueError:
PORT = 1234
app.run(HOST, PORT, debug=True)
Line 1, Column 1
Tab Size: 4
Python
Type here to search
ENG
14:53
12-08-2020


C:\Users\Lonewolf\Desktop\Final Year Project\atp\cosine_similarity.py - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
cosine_similarity.py
objective_question.py
subjective_question.py
util.py
views.py
# Load packages
import os
import math
import nltk as nlp
import numpy as np
def find_mod_vector(vector):
summ = 0
for x in vector:
summ += x**2
return math.sqrt(summ)
def compute_vector(small_list, main_list):
myvector = list()
for i in range(len(main_list)):
key = main_list[i]
if key in small_list:
myvector.append(1)
else:
myvector.append(0)
return myvector
def get_vector(string_str):
sentences = nlp.sent_tokenize(string_str)
t_list = list()
for s in sentences:
temp = nlp.word_tokenize(s)
for x in temp:
t_list.append(x)
return t_list
def evaluate_subj_answer(original_answer, user_answer):
score_obt = 0
orig_list = get_vector(original_answer)
user_list = get_vector(user_answer)
main_list = orig_list + user_list
vector1 = compute_vector(orig_list, main_list)
vector2 = compute_vector(user_answer, main_list)
v1 = find_mod_vector(vector1)
v2 = find_mod_vector(vector2)
v1_v2 = np.dot(vector1, vector2)
dist = v1_v2 / (v1 * v2)
Line 1, Column 1
Tab Size: 4
Python
Type here to search
ENG
14:52
12-08-2020