**VISVESVARAYA TECHNOLOGICAL UNIVERSITY, BELGAUM 590014**

****

Programming In Python

**“Automatic Answer Script Correction”**

By

Deepthi Bhat (1BM16CS003)

Aditi Awasthi (1BM16CS008)

Medhini Oak (1BM16CS047)

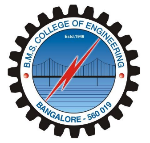
Under the Guidance of

**Prof. Selva Kumar S**

Assistant Professor, Department of CSE

BMS College of Engineering

Self-study Projectcarried out at

****

Department of Computer Science and Engineering

BMS College of Engineering

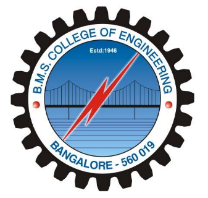
(Autonomous college under VTU)

P.O. Box No.: 1908, Bull Temple Road, Bangalore-560 019

Aug-Dec 2019

**BMS COLLEGE OF ENGINEERING**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

****

***CERTIFICATE***

This is to certify that the Programming In Python titled “**Automatic Answer Script Correction**” has been carried out by Deepthi Bhat (1BM16CS003), Aditi Awasthi (1BM16CS008), Medhini Oak(1BM16CS047) during the academic year Jan – May 2019.

Signature of the guide

**Prof. Selva Kumar S**

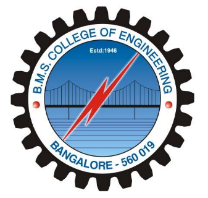
Assistant Professor

Department of Computer Science and Engineering

BMS College of Engineering, Bangalore

**BMS COLLEGE OF ENGINEERING**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

****

***DECLARATION***

We, Deepthi Bhat (1BM16CS003), Aditi Awasthi (1BM16CS008) and Medhini Oak (1BM16CS047) students of 6th Semester, B.E, Department of Computer Science and Engineering, BMS College of Engineering, Bangalore, hereby declare that, this Programming In Pythonwork entitled "**Automatic Answer Script Correction**" has been carried out by us under the guidance of Prof. Selva Kumar S, Assistant Professor, Department of CSE, BMS College of Engineering, Bangalore during the academic semester Jan-May 2019.

We also declare that to the best of our knowledge and belief, the development reported here is not from a part of any other report by any other students.

Signature

Aditi Awasthi (1BM16CS008)

Deepthi Bhat (1BM16CS003)

Medhini Oak (1BM16CS047)

1. Introduction

Automatic Answer Script Correction is a module that is designed for the use of examiners to automatically grade answer scripts of students based on a given answer scheme. This project has the following modules :

1. Upload of question scheme and answer script of student
2. Answer Image Segmentation
3. Handwriting Optical Character Recognition (OCR)
4. No-SQL Database
5. Rapid Automatic Keyword Extraction (RAKE)
6. Keyword matching and marks calculator

The uploaded picture of the answer script should have the following format:

1. The answers must be written in above average handwriting in block letters
2. The answers must be written on a blank sheet, with each answer separated by a clearly visible line
3. The answers have to be written in order
4. The images should be clear and individual characters legible.
5. Description of the modules

A. Upload of question scheme and answer script of student

The teacher must enter the question along with the correct answer to the it and the number of marks allotted to the question.

Once the required scheme is uploaded, for a particular instance, the question paper is selected and the scanned pictures of the answer script of a particular student are uploaded into the application.

For creation of graphical user interface for all modules, 2 libraries - Tkinter, easygui have been used. These are the most commonly used GUI toolkits in python.

B. Answer Image Segmentation

Libraries used : opencv, numpy, PIL

**OpenCV** (*Open source computer vision*) is a library of programming functions mainly aimed at real-time computer vision. Hough Transform is a popular technique to detect any shape, if you can represent that shape in mathematical form. It makes use of the numpy library.

The starting and ending coordinates of each line which separates two answers in the answer script are found. These coordinates are then used to crop out the different answers from the original image and store them as different images. The PIL (Python Imaging Library) has been used for this purpose.

C. Handwriting Optical Character Recognition (OCR)

Libraries used : google-api-python-client

Google Cloud AI provides modern machine learning services, with pre-trained models and a service to generate own tailored models. This project makes use of Google’s neural-net-based ML service, since it has better training performance and increased accuracy compared to other deep learning systems and its services are fast, scalable, and easy to use.

The Vision API can detect and extract text from images. These are the annotation features that support OCR:

* TEXT\_DETECTION
* DOCUMENT\_TEXT\_DETECTION
* HANDWRITING\_DETECTION (beta phase)

The project makes use of HANDWRITING\_DETECTION annotation feature. A Google Cloud account has been created and the Cloud Vision API enabled. After generation of a private key to make use of the API’s services, it can be used to convert the handwriting presented in the segmented answer scripts and store it in text files, which are RAKEd again.

D. No-SQL Database

Libraries used : pymongo

MongoDB is a DBMS that uses a document-oriented database model.Our project makes use of it because it involves processing jobs and data that doesn't fit well in a rigid relational model. Instead of using tables and rows as in relational databases, the MongoDB architecture is made up of collections and documents, which is very well-suited to the storage of a variable number of keywords along with the weight of each keyword in json format.

For the purpose of uniformity, student database is also maintained in MongoDB itself.

E. Rapid Automatic Keyword Extraction (RAKE)

Libraries used : rake-nltk

Rapid Automatic Keyword Extraction algorithm is a domain independent keyword extraction algorithm which tries to determine key phrases in a body of text by analyzing the frequency of word appearance and its co-occurrence with other words in the text.

This algorithm runs in two different places in the project. First, it ranks the keywords in the scheme uploaded by the teacher and assigns weights to every keyword present in the correct answer.

Second, once the answer written by the student is converted to a computer readable format through OCR, it identifies the keywords present in the answer, making it easier to compare the correct answer and the answer written in the answer script

F. Keyword Matching And Marks Calculator

This part of the project compares the keywords which should be present in the answer and the keywords which are present in the given answer script. Marks allotted for that question are calculated using an algorithm which takes into account the weights of the keywords present in the answer and scales it according to the maximum marks which can be awarded for that question.

G. Add Students to the Database

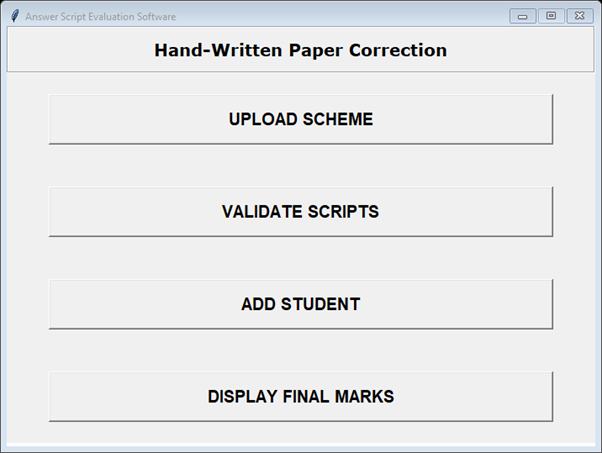
This module enables the examiner to add more students to the database. It takes the USN of the student to be added and stores it in the database.

H. Edit the Raked Scheme

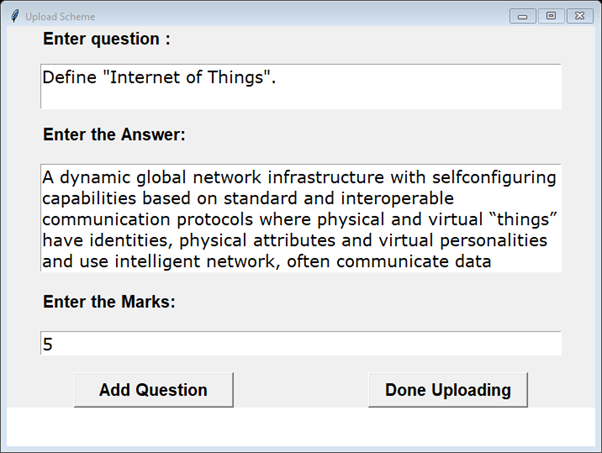
After the teacher has entered the question, answer and marks for a question, a window with the rake-results id displayed, with each keyword displayed with its corresponding weights. The teacher can change the scheme and the changed scheme is stored in the database.

2. Graphical User Interface screenshots

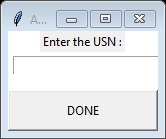
2.1 Home Page



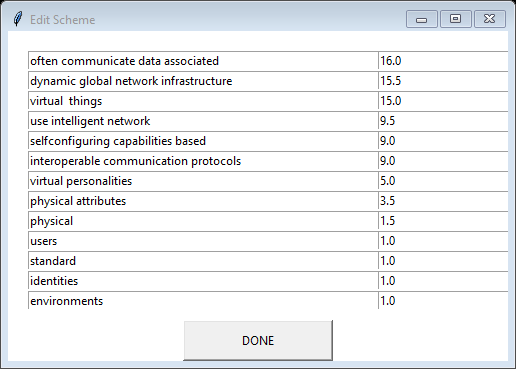
2.2 Add Scheme



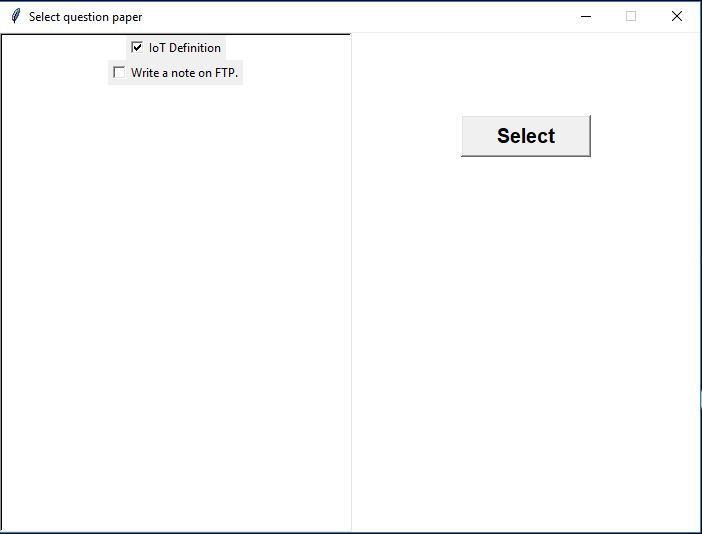
2.3 Add Student



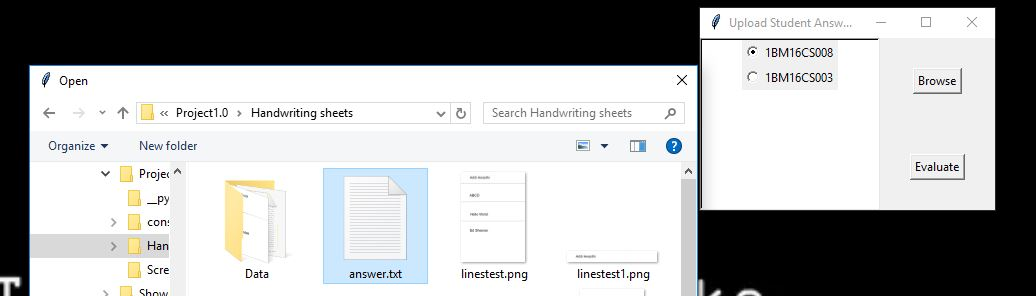
2.3 Edit Scheme



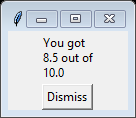
2.3 Choose Questions



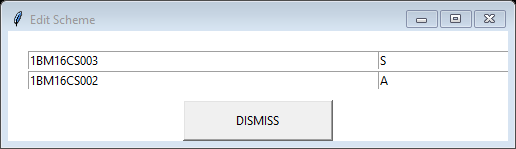
2.4 Select student and upload answer script



2.5 Evaluate Scripts

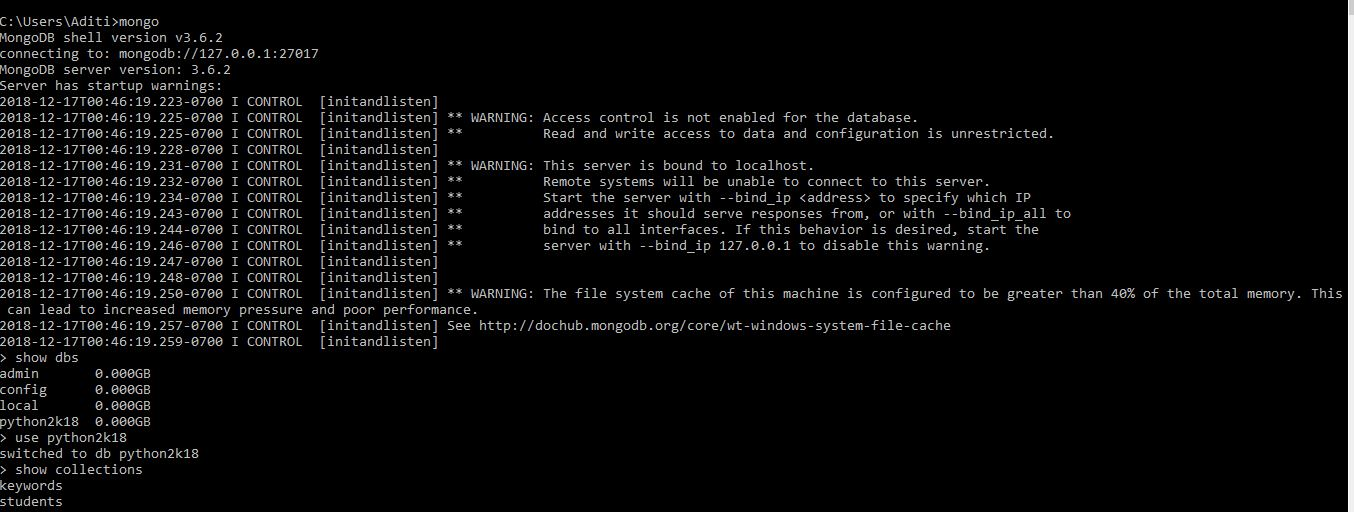


2.6 Display Grades of all Students

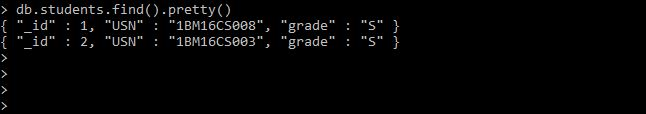


3. Database screenshots

1. Structure of the database



b. Students Collection



c. Keywords Collection

