A Project Report

on

Grammar Checker Using Machine Learning

Submitted in partial fulfillment of the

requirement for the award of the degree of

Bachelor of Technology in Computer Science and Engineering

GALGOTIAS UNIVERSITY

(Established under Galgotias University Uttar Pradesh Act No. 14 of 2011)

Under The Supervision of
Dr. J.N. Singh
Professor
Department of Computer Science and Engineering

Submitted By

18SCSE1010729 - NIVEDITA MANDAL

18SCSE1010678 - JAYANT

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

GALGOTIAS UNIVERSITY, GREATER NOIDA, INDIA

DECEMBER - 2021

SCHOOL OF COMPUTING SCIENCE ANDENGINEERING
GALGOTIAS UNIVERSITY, GREATER NOIDA

CANDIDATE'S DECLARATION

I/We hereby certify that the work which is being presented in the project, entitled **" Grammar Checker Using Machine Learning "** in partial fulfillment of the requirements for the award of the **BACHELOR OF TECHNOLOGY IN COMPUTER SCIENCE AND ENGINEERING** submitted in the **School of Computing Science and Engineering** of Galgotias University, Greater Noida, is an original work carried out during the period of **JULY-2021 to DECEMBER-2021**, under the supervision of **Dr. J.N.Singh, Professor**, **Department of Computer Science and Engineering** of School of Computing Science and Engineering , Galgotias University, Greater Noida

The matter presented in the project has not been submitted by me/us for the award of any other degree of this or any other places.

18SCSE1010729 - NIVEDITA MANDAL
18SCSE1010678 - JAYANT

This is to certify that the above statement made by the candidates is correct to the best of my knowledge.

Supervisor

(Dr.J.N.Singh,

Professor)

CERTIFICATE

The Final Thesis/Project/ Dissertation Viva-Voce examination of **18SCSE1010729 - NIVEDITA MANDAL, 18SCSE1010678 - JAYANT** has been held on____________and his/her work is recommended for the award of **BACHELOR OF TECHNOLOGY IN COMPUTER SCIENCE AND ENGINEERING**.

Signature of Examiner(s) **Signature of Supervisor(s)**

Signature of Project Coordinator **Signature of Dean**

Date:

Place:

ABSTRACT

This paper models the working of Machine Learning algorithm to detect errors from a given sentence or a paragraph.When the user input the text then it highlight the error in the text.The database contains the correct words which we generally use in English language.The errors are checked in the database and then it returns the correct word for the highlighted error.It also highlight the corrected word in the corrected text so that the user will understand the mistakes.At last we get the corrected text.The types of error which can be in a sentence written in an English language are spelling, misused, split and merge, subject verb agreement, common and proper, double words, comparative superlative, indefinite article, definite article, plurality, pronouns, consecutive nouns and tenses.We developed this software because we saw that it is needed by most of the people because when we write long essays then it is very difficult to find the mistakes.So in order to save the time we created this software.With the help of this software students will save there time. Mostly people make a lot of mistakes while writing essay or resumes.Our software will benefit students as well as a working professional.With the help of our software students can can easily find out the errors they generally commit while writing an essay or any paragraph.It will also help them to improve there writing capability.Freshers or working people who need to write   resumes can use our software as it will help them to write there resumes without any error.Our main purpose is to help more and more people to write error free sentences or paragraph.Our software is user-friendly.So it can be used by any age group people.Software which are available in the market, charges a lot in order to correct the sentences.

TABLE OF CONTENTS

CHAPTER NO.		TITLE	PAGE NO.
		ABSTRACT	**iii**
		LIST OF TABLE	**iv**
		LIST OF FIGURES	**vi**
		LIST OF SYMBOLS	**vii**
1.		**INTRODUCTION**	**6**
	1.1	Introduction	6
2.		**REQUIREMENTS**	**11**
	2.1	Requirements	11
	2.2	Feasibility	11
	2.3	Scope	12
3.		**ANALYSIS**	**14**
	3.1	Analysis	14
	3.2	Activity Time Schedule(PERT)	15
4.		**DESIGN**	**24**
	4.1	Design	24
5.		**IMPLEMENTATAION**	**25**
	5.1	Implementation	25
	5.2	Solution Approach	25
	5.3	Project Synopsis	26
	5.4	Algorithm	28
	5.5	Testing	29
6.		**LIMITATIONS**	**31**
	6.1	Limitations	31
	6.2	Future Scope of the Project	32
7.		**CONCLUSION**	**33**
	7.1	Outcome	33
	7.2	Examples	33
	7.3	Web Page	44
8.		**REFERENCES**	**48**
	8.1	References	48

Chapter 1: Introduction about the Project

Introduction:

Our objective is to develop **Machine Learning** algorithms with an aim to detect the grammatical errors and contextual errors from a given sentence or paragraph and then recognize the same.A Grammar Checker is software or a program which is used to find grammatical errors from a given sentence or a paragraph. That is to say, it checks for improper sentence structure and word usage (e.g., their instead of there), poorly placed or unnecessary punctuation, and other more esoteric errors.

The Grammar Checker helps you write better English and efficiently corrects texts. Based on the context of complete sentences, Grammar Checker uses Machine Learning to correct grammar mistakes, spelling mistakes and misused words, with unmatched accuracy. Grammar check software improves your text just like a human reviewer would.Grammar checkers underlines the mistakes given in sentence or a paragraph and Ginger uses groundbreaking technology to detect grammar and spelling errors in sentences and to correct them with unmatched accuracy. From singular vs plural errors to the most sophisticated sentence or tense usage errors, Ginger picks up on mistakes and corrects them.

Grammar Checker uses Ginger API tool to correct all types of mistakes which present in a sentence or a paragraph written in English language. Ginger corrects all types of grammatical mistakes including topics that are not addressed by any other grammar correction program.

The main purpose of our project is to remove all types of errors in a particular sentence. Machine learning is widely used in this project as it detects errors in a sentence.

Errors can be many types like a spelling error, comparison, grammatical error, plural, punctuation error, spelling, misuse, definite article, consecutive nouns, separations and consonants, subject consonant, common noun, original noun, dual noun, superlative, indefinite noun, pronouns and period. We have built a software written in Python language using the concept of Machine Learning. When a user enters a sentence that contains system errors the software checks the correct name for the particular error. The sentence is then edited and the corrected sentence is printed. Our software's database contains words, sentences and paragraph that are widely used. So whenever a user enters a sentence, then checks the correct error name provided.

It can only edit the grammar of a sentence or paragraph written in English. So we need to adjust the grammar in other languages. The sentence or paragraph given to the user should not exceed more than 600 characters and therefore cannot look for a sentence or paragraph with more than 600 characters at a time. So we need to exceed the letter limit. All types or errors cannot be removed from sentences as there are many variations in the English language. So a revised text cannot guarantee 100% accuracy. We therefore need to add more sentences to our website to improve the level of accuracy.

Our software will not charge any money because it is free to use. So it will help people to write an accurate sentence or paragraph. Grammar checker can be used by people who need to write a flawless document. People can learn to write flawless sentences as our software highlights errors in sentences. Our software highlights the error in red and highlights the error corrected in green. This way it will help people to find errors while writing important text. It will also save people's time as it takes 2 seconds to print the correct sentence. It's very simple to use and will help people to discover spelling and grammar errors while writing sentences.

It is very easy to use and will help people to find spelling and grammatical errors while writing the sentences.

As shown in Figure 1, such as a spelling error, comparison, grammatical error, plural, punctuation error, spelling, misuse, definite article, sequential nouns, separating and consolidation, subject verb agreement, common noun, proper noun, dual noun, initials minor vague, pronouns and verbs.

If the given sentence contains a spelling error, then you need to check the correct spelling name for the website. If the correct spelling name is found, it sends the correct word and the main function. In our software we have collected and stored millions of correct words and sentences that are most commonly used. If the correct name is not found, then retrieve the correct name in the database. So whenever a user enters a sentence they look for the correct word on the website.

Another type of error is a duplicate of two words, sometimes when we write a paragraph, we accidentally write the same words twice. Phrases can be corrected using our software. Our software detects the wrong period used and corrects the correct period. Descriptions with incorrect endless article may be detected by the system inspected.

Sequence of consecutive nouns can be detected. Sometimes when we type we mistakenly give extra space between words. So it creates an error. Our software can also be used to remove additional spaces between words.

Sometimes sentences contain words that are not often used. So our software can also remove words that have been misused in a printed sentence.

Our software can also highlight errors in sentences in red and corrective words in green. This helps the user to understand the mistakes they made while writing a sentence or paragraph. It saves user time by helping them find errors quickly and easily. It takes very less time to print error-free sentence. The user can improve his or her English language. The software doesn't require any kind of money. Some software available in the market is not free and charge a large amount to check for errors in sentences.

When a user enters a sentence that contains system errors the software checks the correct name for the particular error. The sentence is then edited and the corrected sentence is printed. The software contains millions of words and paragraphs which we generally use. So whenever a user enters a sentence and checks the correct word for a given error. So with the help of machine learning, we have created software that detects errors in a given sentence or paragraph by highlighting the error and after highlighting the error, printer the sentence or section appropriate.

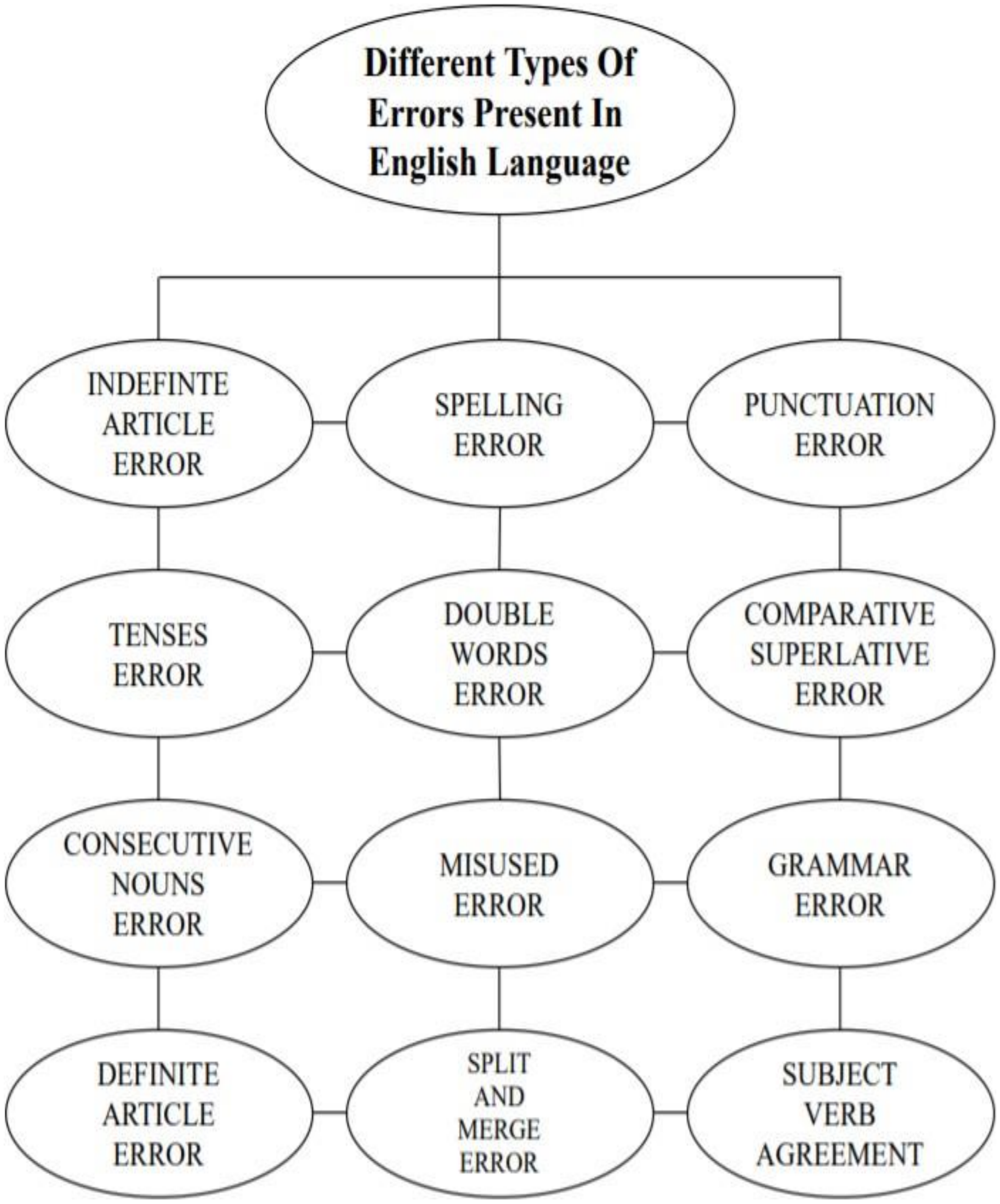


Figure 1: Different types of errors present in English Language

Chapter 2: Requirements , Feasibility and Scope/Objective

Requirements

1) In order to run our software we need a computer or laptop.

2) Our software will run on Google Collab.

3) A good internet connectivity is required to run our software.

4) Tools which are used to build are software is Python, ColoredText,Ginger Api.

Feasibility

While context-sensitive spell-check systems (such as AutoCorrect) are able to automatically correct a large number of input errors in instant messaging, email, and SMS messages, they are unable to correct even simple grammatical errors. For example, the message "I'm going to store" would be unaffected by typical autocorrection systems, when the user most likely intendend to communicate "I'm going to the store".

I decided to try using Ginger API to solve this problem. Specifically, I set out to construct models capable of processing a sample of conversational written English and generating a corrected version of that sample.

Correcting Grammatical Errors with Machine Learning

The basic idea behind this project is that we can generate large training datasets for the task of grammar correction by starting with grammatically correct samples and introducing small errors to produce input-output pairs. The details of how we construct these datasets, train models using them, and produce predictions for this task are described below

Scope/Objective

- Our objective is to develop **Machine Learning** algorithms with an aim to detect the grammatical errors and contextual errors from a given sentence or paragraph and then recognize the same.

- A Grammar Checker is software or a program which is used to find grammatical errors from a given sentence or a paragraph. That is to say, it checks for improper sentence structure and word usage (e.g., their instead of there), poorly placed or unnecessary punctuation, and other more esoteric errors.

The Grammar Checker helps you write better English and efficiently corrects texts.Based on the context of complete sentences, Grammar Checker uses Machine Learning to correct grammar mistakes, spelling mistakes and misused words, with unmatched accuracy. Grammar check software improves your text just like a human reviewer would.

- Grammar checkers underlines the mistakes given in sentence or a paragraph and Ginger uses groundbreaking technology to detect grammar and spelling errors in sentences and to correct them with unmatched accuracy. From singular vs plural errors to the most sophisticated sentence or tense usage errors, Ginger picks up on mistakes and corrects them.

- Grammar Checker uses Ginger API tool to correct all types of mistakes which present in a sentence or a paragraph written in English language. Ginger corrects all types of grammatical mistakes including topics that are not addressed by any other grammar correction program.

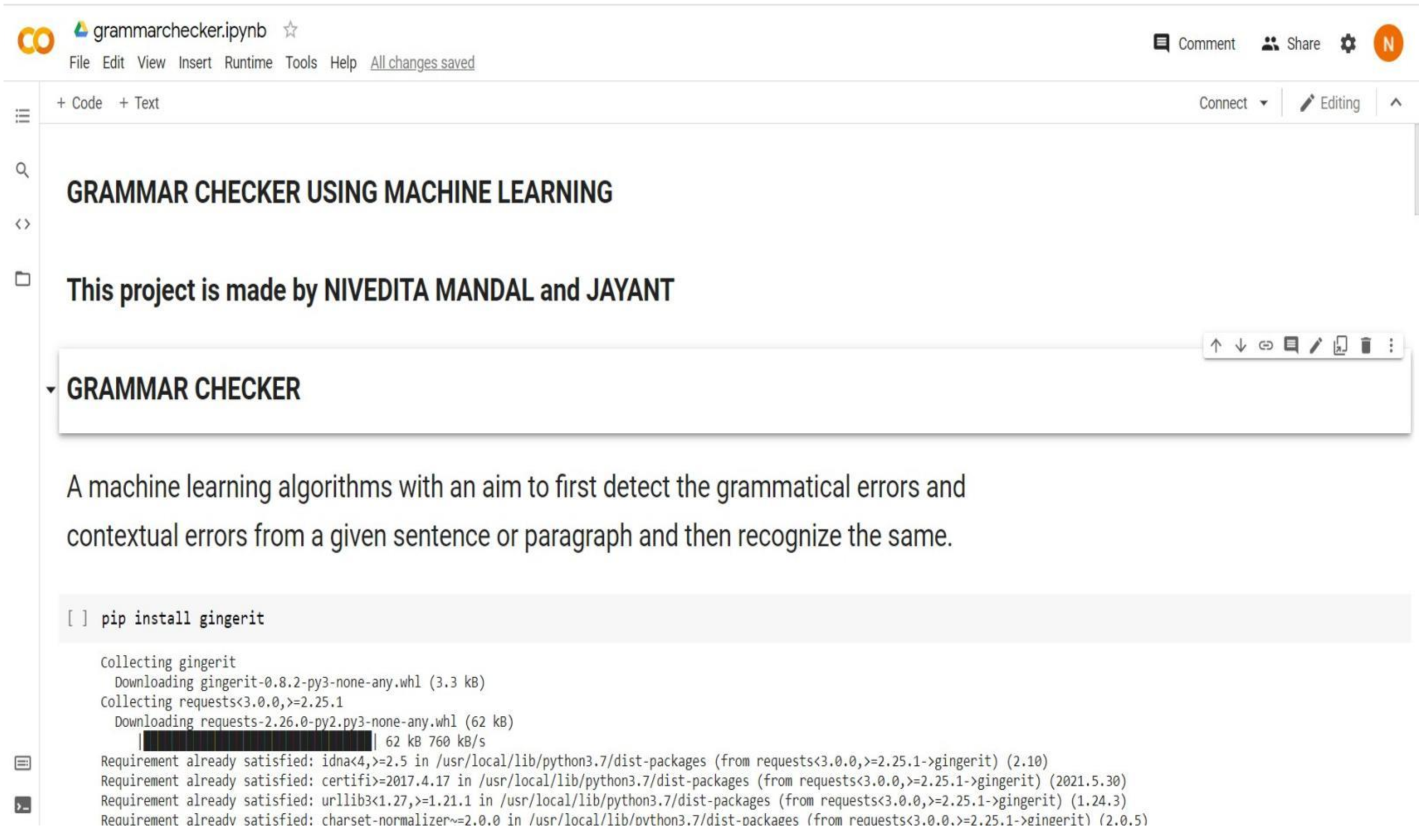
Chapter 3: Analysis , Activity Time Schedule (PERT)

Analysis

We have developed a **Machine Learning** algorithms with an aim to detect the grammatical errors and contextual errors from a given sentence or paragraph and then recognize the same.We have named the software as “Grammar Checker” .

A Grammar Checker is software or a program which is used to find grammatical errors from a given sentence or a paragraph. That is to say, it checks for improper sentence structure and word usage (e.g., their instead of there), poorly placed or unnecessary punctuation, and other more difficult errors.

Activity Time Schedule (PERT)



GRAMMAR CHECKER USING MACHINE LEARNING

This project is made by NIVEDITA MANDAL and JAYANT

GRAMMAR CHECKER

A machine learning algorithms with an aim to first detect the grammatical errors and contextual errors from a given sentence or paragraph and then recognize the same.

```
[ ] pip install gingerit
```

```
Collecting gingerit
  Downloading gingerit-0.8.2-py3-none-any.whl (3.3 kB)
Collecting requests<3.0.0,>=2.25.1
  Downloading requests-2.26.0-py2.py3-none-any.whl (62 kB)
     |████████████████████████████████| 62 kB 760 kB/s
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.7/dist-packages (from requests<3.0.0,>=2.25.1->gingerit) (2.10)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.7/dist-packages (from requests<3.0.0,>=2.25.1->gingerit) (2021.5.30)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in /usr/local/lib/python3.7/dist-packages (from requests<3.0.0,>=2.25.1->gingerit) (1.24.3)
Requirement already satisfied: charset-normalizer~=2.0.0 in /usr/local/lib/python3.7/dist-packages (from requests<3.0.0,>=2.25.1->gingerit) (2.0.5)
```

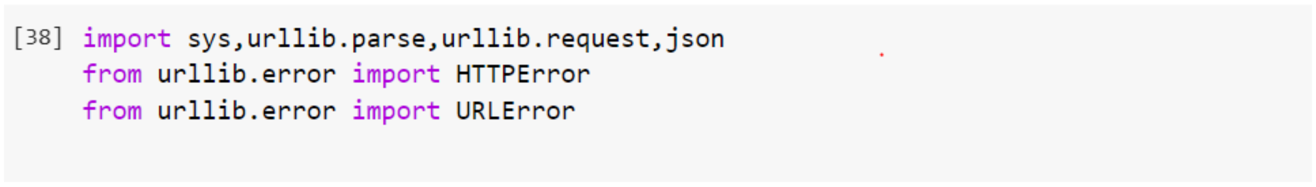
We have implemented our software on Google Colab.We have used Python as our project is based on Machine Learning .

In the starting we have written the title of the project ,our group number, name of the software and team members.

Also we have wrote the main objective of project .

CODE:

Lets import the libraries

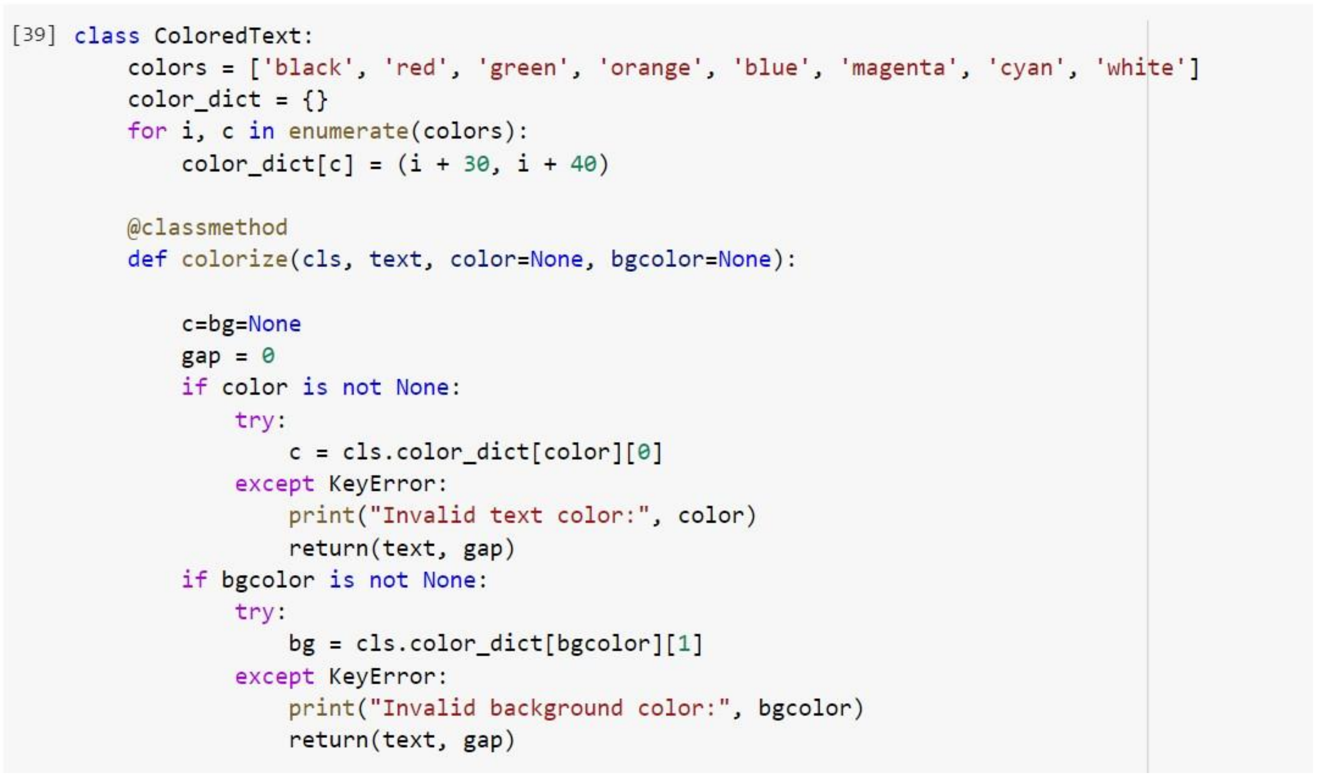


```
[38] import sys,urllib.parse,urllib.request,json
     from urllib.error import HTTPError
     from urllib.error import URLError
```

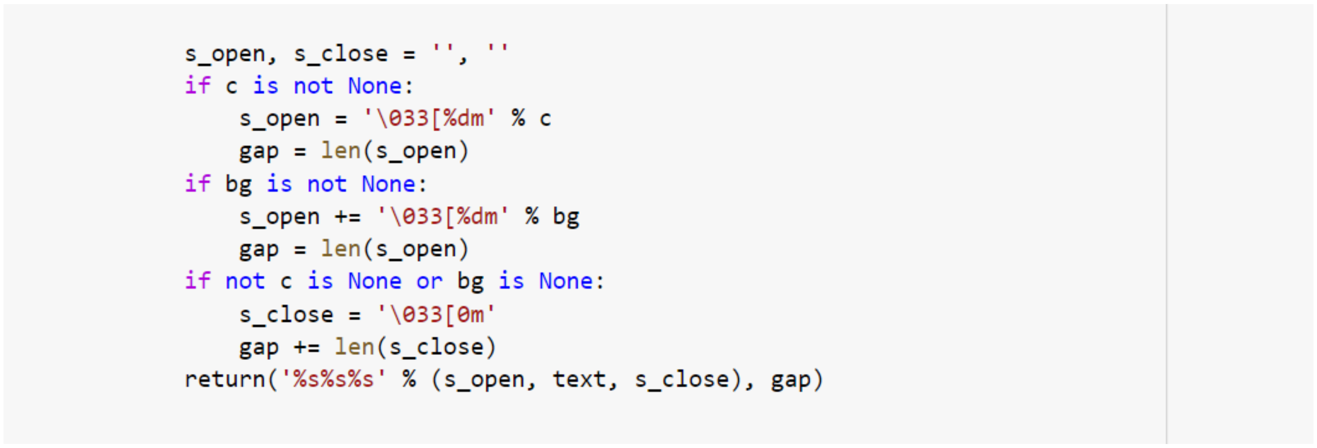
We are importing the required libraries import sys, urllib.parse, urllib.request, json

from urllib.error import HTTPError, from urllib.error import URLError

Below is a class created to colorize output texts



```
[39] class ColoredText:
         colors = ['black', 'red', 'green', 'orange', 'blue', 'magenta', 'cyan', 'white']
         color_dict = {}
         for i, c in enumerate(colors):
             color_dict[c] = (i + 30, i + 40)

         @classmethod
         def colorize(cls, text, color=None, bgcolor=None):

             c=bg=None
             gap = 0
             if color is not None:
                 try:
                     c = cls.color_dict[color][0]
                 except KeyError:
                     print("Invalid text color:", color)
                     return(text, gap)
             if bgcolor is not None:
                 try:
                     bg = cls.color_dict[bgcolor][1]
                 except KeyError:
                     print("Invalid background color:", bgcolor)
                     return(text, gap)
```



```
        s_open, s_close = '', ''
        if c is not None:
            s_open = '\033[%dm' % c
            gap = len(s_open)
        if bg is not None:
            s_open += '\033[%dm' % bg
            gap = len(s_open)
        if not c is None or bg is None:
            s_close = '\033[0m'
            gap += len(s_close)
        return('%s%s%s' % (s_open, text, s_close), gap)
```

We are using Class ColouredText, here we colorize output text.

a. Here we define a list with different colors.

b. Create an empty dictionary.

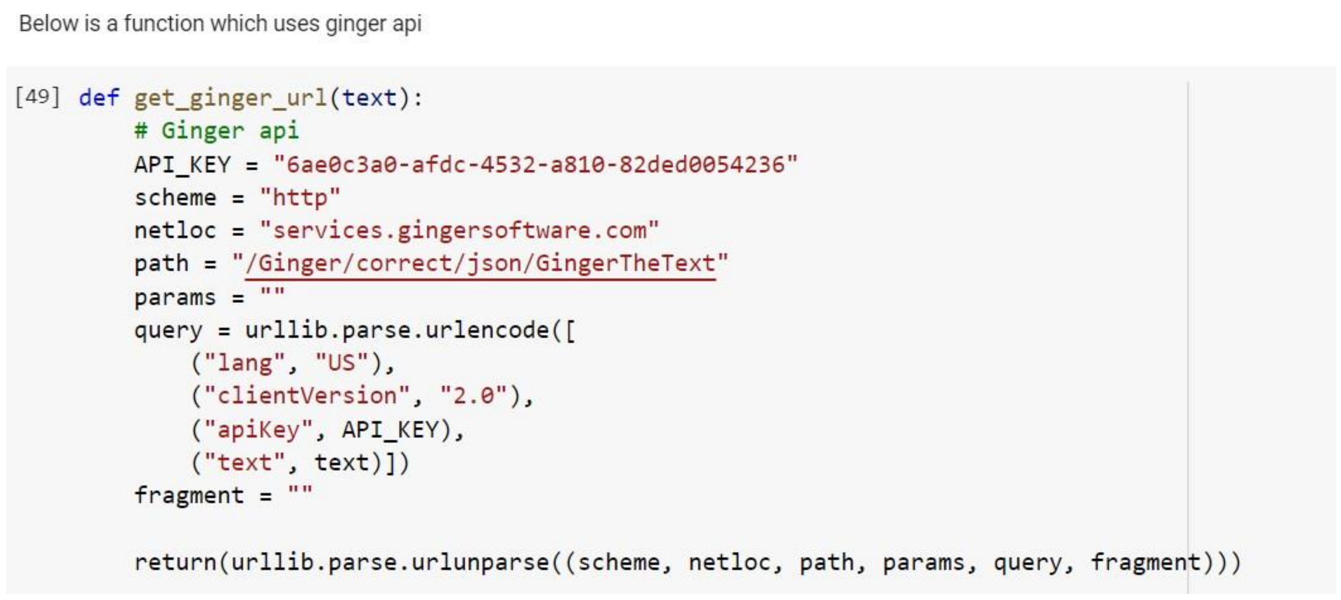
c. Using for loop we fill our dictionary(i.e. key value pairs.)

d. def colorize:(cls(colour), text(word),color=None, bgcolor(background color)=None)

e. C = Background color.

f. try and except block is there for decoding with exceptional cases(i.e. words with color that we defined in our list)

g. s_open, s_close=”,” -- It is for opening and closing of a file which contains the paragraph or sentence.



Below is a function which uses ginger api

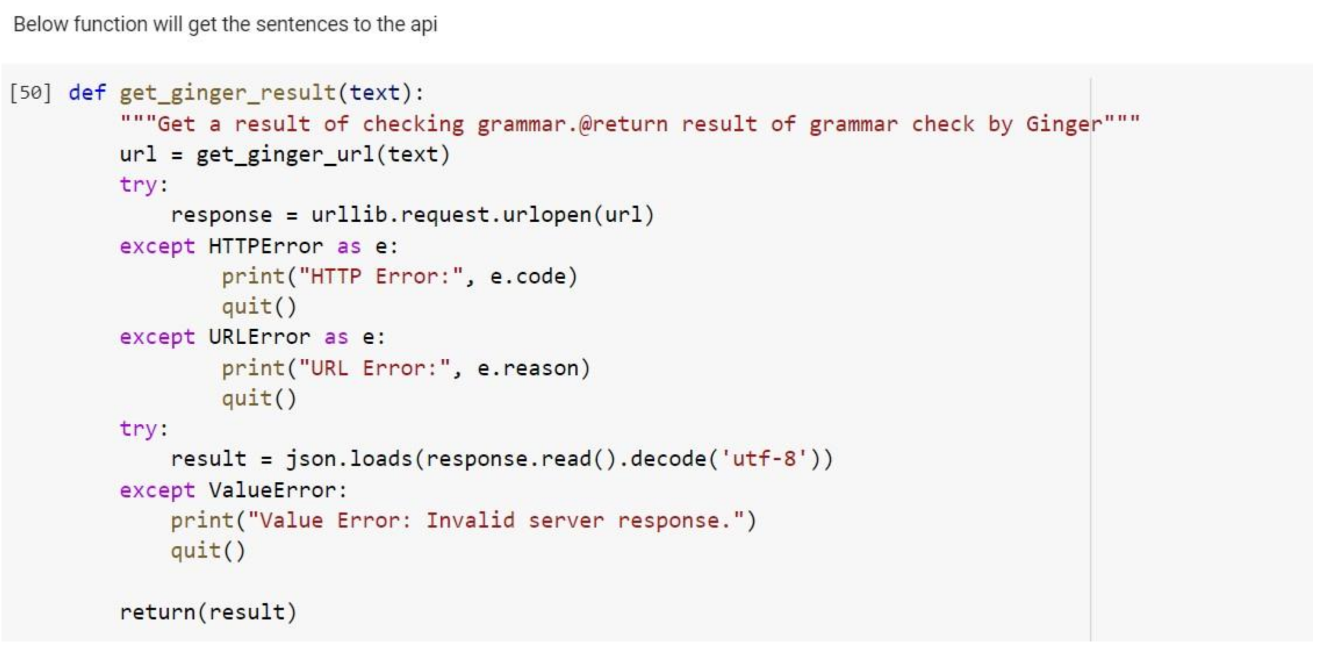
```
[49] def get_ginger_url(text):
         # Ginger api
         API_KEY = "6ae0c3a0-afdc-4532-a810-82ded0054236"
         scheme = "http"
         netloc = "services.gingersoftware.com"
         path = "/Ginger/correct/json/GingerTheText"
         params = ""
         query = urllib.parse.urlencode([
             ("lang", "US"),
             ("clientVersion", "2.0"),
             ("apiKey", API_KEY),
             ("text", text)])
         fragment = ""

         return(urllib.parse.urlunparse((scheme, netloc, path, params, query, fragment)))
```

Then we are using get_ginger_result(text) -- This function will get the sentences to the api.

a) It means that with the help of this function we send our sentence to the ginger_api_url where it is checked for its grammatical and overall correctness.

b) Again here, we use try and except blocks for dealing with exceptional cases(i.e. the sentences or word which isn’t there in our database.)

c) It checks the sentence and then prompt the message accordingly.(i.e. it can even raise an error message)



Below function will get the sentences to the api

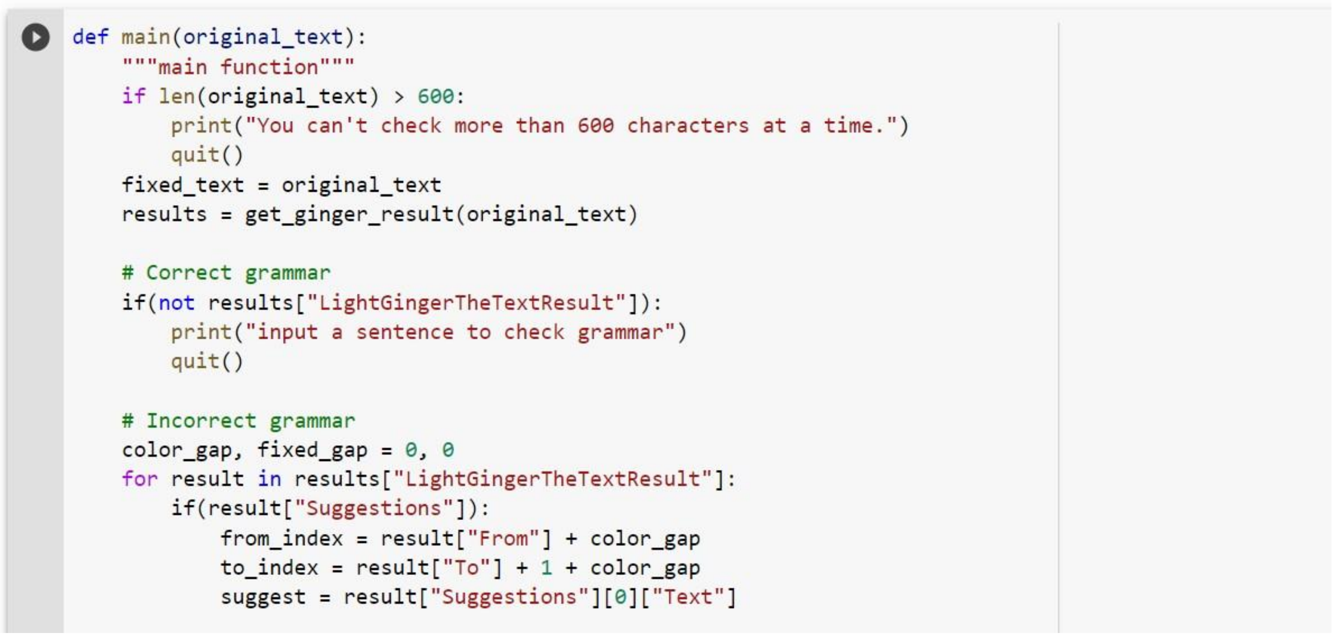
```
[50] def get_ginger_result(text):
        """Get a result of checking grammar.@return result of grammar check by Ginger"""
        url = get_ginger_url(text)
        try:
            response = urllib.request.urlopen(url)
        except HTTPError as e:
                print("HTTP Error:", e.code)
                quit()
        except URLError as e:
                print("URL Error:", e.reason)
                quit()
        try:
            result = json.loads(response.read().decode('utf-8'))
        except ValueError:
            print("Value Error: Invalid server response.")
            quit()

        return(result)
```

Then we are using get_ginger_url(Ginger api)

a) It is function which uses Ginger api.Here, Ginger api act as a database which contains words suggestions(in its correct spelling and grammatical form)

b) Here, it uses the API_KEY, scheme, path, params, query and fragment.

Finally the main function to test and correct our grammar



```
def main(original_text):
    """main function"""
    if len(original_text) > 600:
        print("You can't check more than 600 characters at a time.")
        quit()
    fixed_text = original_text
    results = get_ginger_result(original_text)

    # Correct grammar
    if(not results["LightGingerTheTextResult"]):
        print("input a sentence to check grammar")
        quit()

    # Incorrect grammar
    color_gap, fixed_gap = 0, 0
    for result in results["LightGingerTheTextResult"]:
        if(result["Suggestions"]):
            from_index = result["From"] + color_gap
            to_index = result["To"] + 1 + color_gap
            suggest = result["Suggestions"][0]["Text"]
```

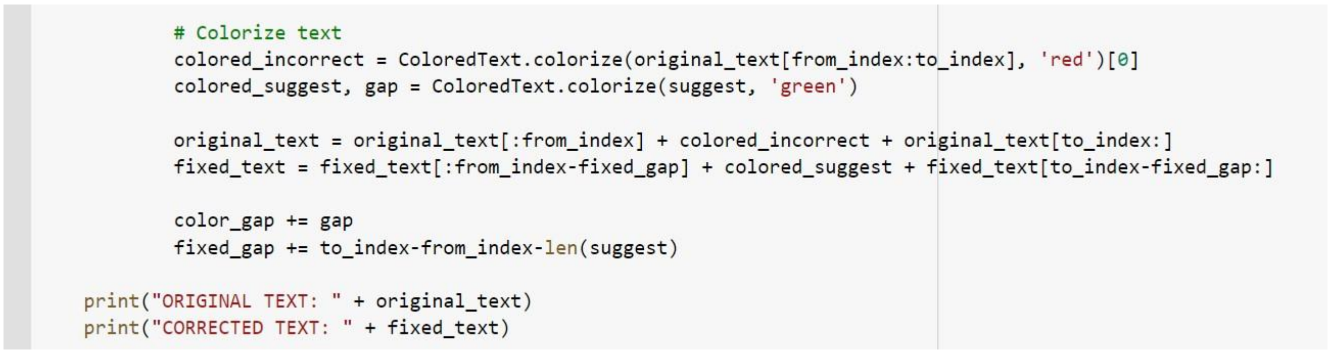
At last we created the main function

a) In the main function the message is received form get_ginger_result.

b) If the grammar is correct then the color is changed to green and then it is printed on the terminal window.

c) If error is received then the part of sentence or word is searched and then it send back to the previous functions for re-checking and correction.

d) Finally, the corrected sentence is recieved and then gets printed as the output.



```
            # Colorize text
            colored_incorrect = ColoredText.colorize(original_text[from_index:to_index], 'red')[0]
            colored_suggest, gap = ColoredText.colorize(suggest, 'green')

            original_text = original_text[:from_index] + colored_incorrect + original_text[to_index:]
            fixed_text = fixed_text[:from_index-fixed_gap] + colored_suggest + fixed_text[to_index-fixed_gap:]

            color_gap += gap
            fixed_gap += to_index-from_index-len(suggest)

    print("ORIGINAL TEXT: " + original_text)
    print("CORRECTED TEXT: " + fixed_text)
```

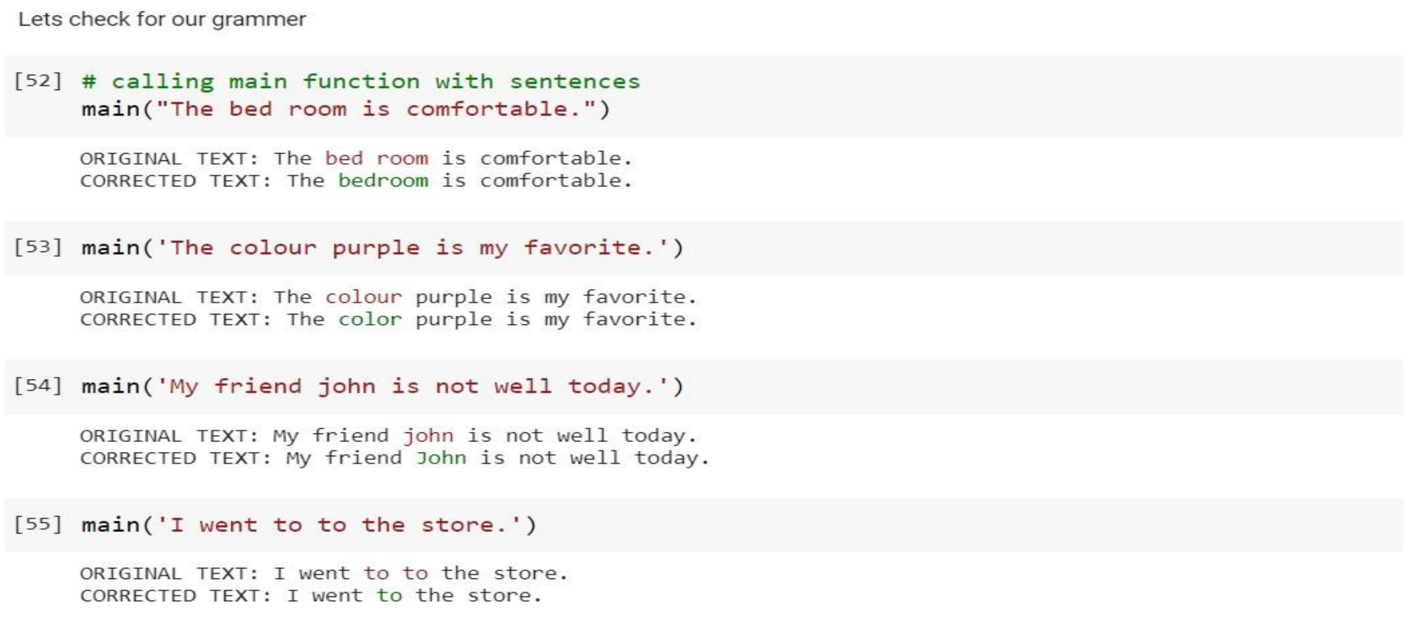
Then there is an option for the user to select color so we have selected red and green color.

a) The purpose of the red color is to highlight the error in the sentence which is the original text.

b) The green color highlight the corrected text.

c) The ORIGINAL TEXT is the sentence or paragraph given by the user.

d) The CORRECTED TEXT is the sentence or paragraph with no error corrected with the help of Grammar Corrector


Lets check for our grammer

```
[52] # calling main function with sentences
     main("The bed room is comfortable.")

     ORIGINAL TEXT: The bed room is comfortable.
     CORRECTED TEXT: The bedroom is comfortable.

[53] main('The colour purple is my favorite.')

     ORIGINAL TEXT: The colour purple is my favorite.
     CORRECTED TEXT: The color purple is my favorite.

[54] main('My friend john is not well today.')

     ORIGINAL TEXT: My friend john is not well today.
     CORRECTED TEXT: My friend John is not well today.

[55] main('I went to to the store.')

     ORIGINAL TEXT: I went to to the store.
     CORRECTED TEXT: I went to the store.
```

Now we are typing the sentence or paragraph in the main to check whether there is any error.

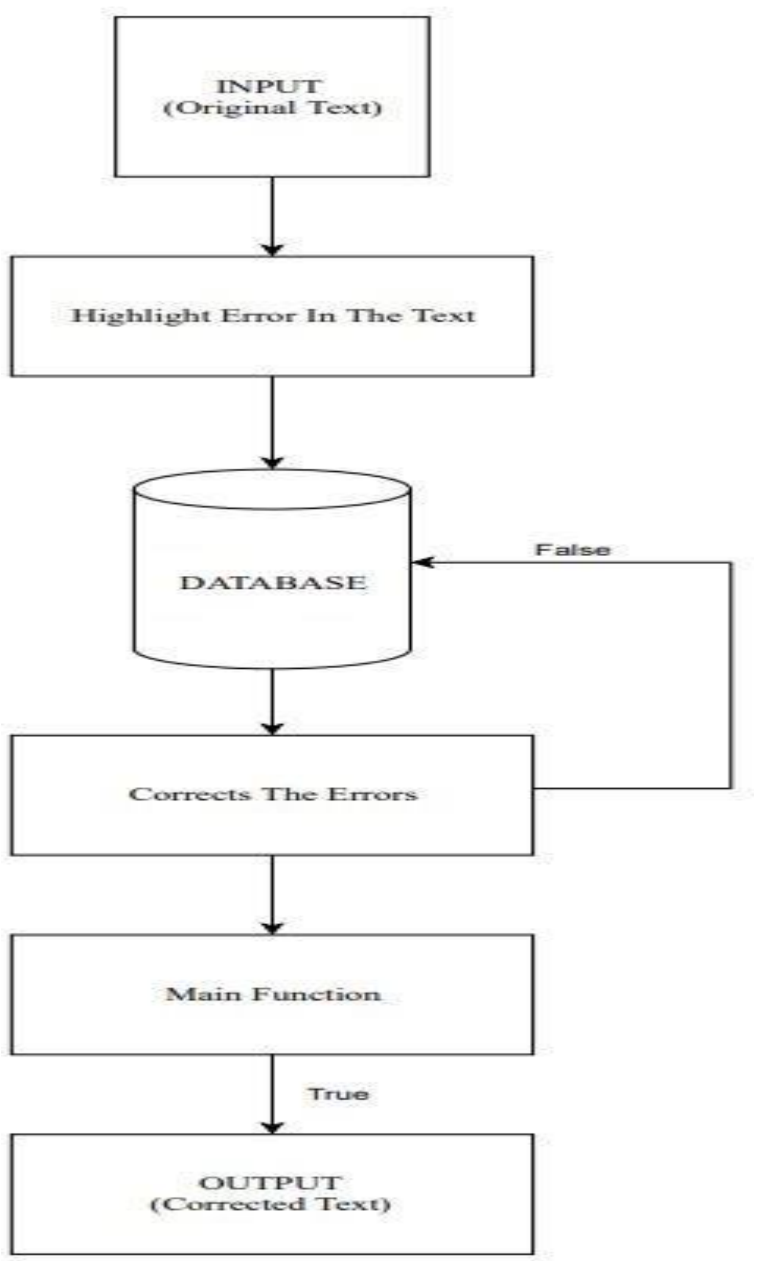
- The ORIGINAL TEXT has the sentence given by the user and CORRECTED TEXT has the sentence which is corrected.
- So the first sentence that we have checked is “The bed room is comfortable”
- The mistake in the sentence is “bed room”. Bedroom is a single word so our Grammar Checker corrects the sentence.
- The second sentence that we have checked is “The colour purple is my favorite”
- The mistake in the sentence is “colour”. The spelling of the word colour should be color.
- The third sentence that we have checked is “My friend john is not well”
- The mistake in the sentence is “john”. The word is a name of a person so the first letter should be in capital letter.

- The fourth sentence that we have checked is “I went to to the store”
- The mistake in the sentence is “to to”.Grammar checker corrects the repeated word .

Chapter 4: Design

Design

Use Case Diagram for Grammar Checker



Chapter 5: Implementation & Testing

Implementation

Solution Approach:

1) In this algorithm we are using Ginger’s proofreading API to correct spelling, grammar, punctuation,

vocabulary and style issues in documents.

2) The user should not input more than 600 characters at a time.

3) The sentence should be written in utf-8 format.

4) At the starting of the code we are importing all the necessary libraries which are required.

5) Then we create colorized output texts.

6) A function is created which uses ginger_api.

7) Then again we created a function which will get the sentences to the api.

8) Then at last we create a main function to test and correct our output.

9) Finally, check the text(original text) and then you will get the corrected text .

10) Errors in the original text will be highlighted in red color and the errors which are rectified in the corrected text will be highlighted in green color.This helps the user the understand the errors in the text.

Project Synopsis:

1) In the starting of our code, we are importing the required libraries.

2) Class ColouredText, here we colorize output text.

*Here we define a list with different colors.

*Create an empty dictionary.

*Using for loop we fill our dictionary(i.e. key value pairs.)

*def colorize:(cls(colour), text(word),color=None, bgcolor(background color)=None)

*C = Background color.

*try and except block is there for decoding with exceptional cases(i.e. words with color that we defined in our list)

*s_open, s_close=”,” -- It is for opening and closing of a file which contains the paragraph or sentence.

3) get_ginger_url(Ginger api) -- It is function which uses Ginger api.Here, Ginger api act as a database which contains words suggestions(in its correct spelling and grammatical form)

Here, it uses the API_KEY, scheme, path, params, query and fragment.

4) get_ginger_result(text) -- This function will get the sentences to the api.

*It means that with the help of this function we send our sentence to the ginger_api_url where it is checked for its grammatical and overall correctness.

*Again here, we use try and except blocks for dealing with exceptional cases(i.e. the sentences or word which isn’t there in our database.)

*It checks the sentence and then prompt the message accordingly.(i.e. it can even raise an error message)

5) In the main function the message is received form get_ginger_result.

*If the grammar is correct then the color is changed to green and then it is printed on the terminal window.

*If error is received then the part of sentence or word is searched and then it send back to the previous functions for re-checking and correction.

*Finally, the corrected sentence is recieved and then gets printed as the output.

Algorithms:

Step 1: START

Step 2: Import the libraries sys, urllib.parse, urllib.request, json, HTTPError and URLError.

Step 3: Create a class to colorize output texts.

Step 4: Create a function using Ginger API.

Step 5: Then again create a function to get the sentences to the Ginger API.

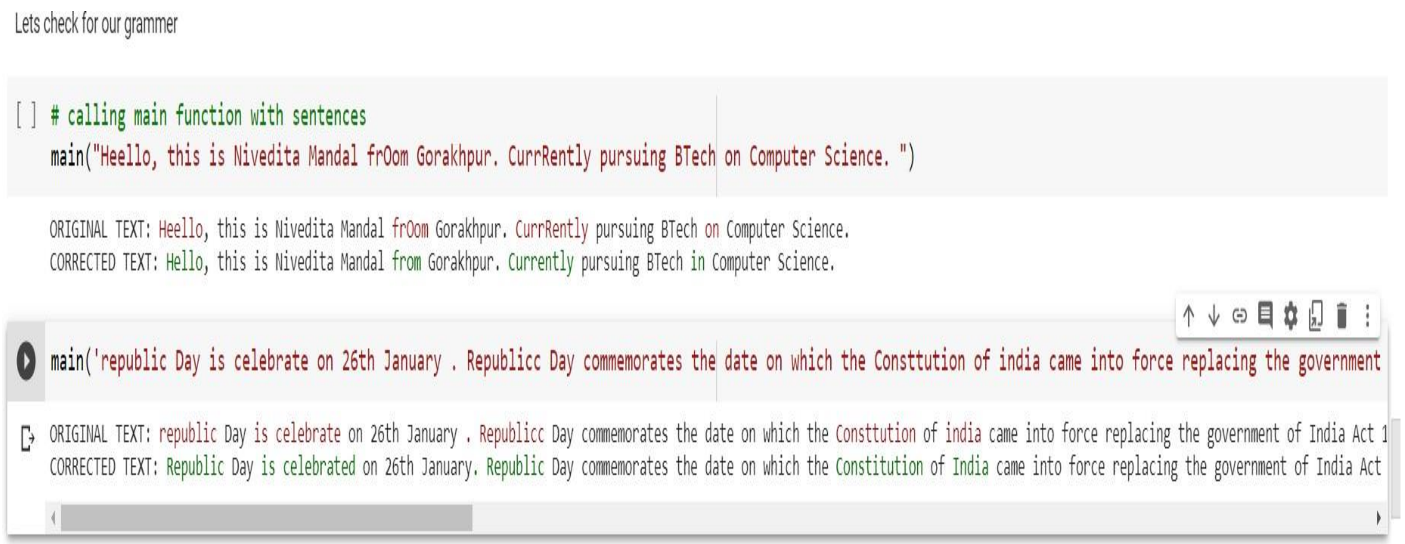
Step 6: Create a main function to test and correct our grammar.

Step 7: Enter the text and check the grammar of the given text.

Step 8: STOP

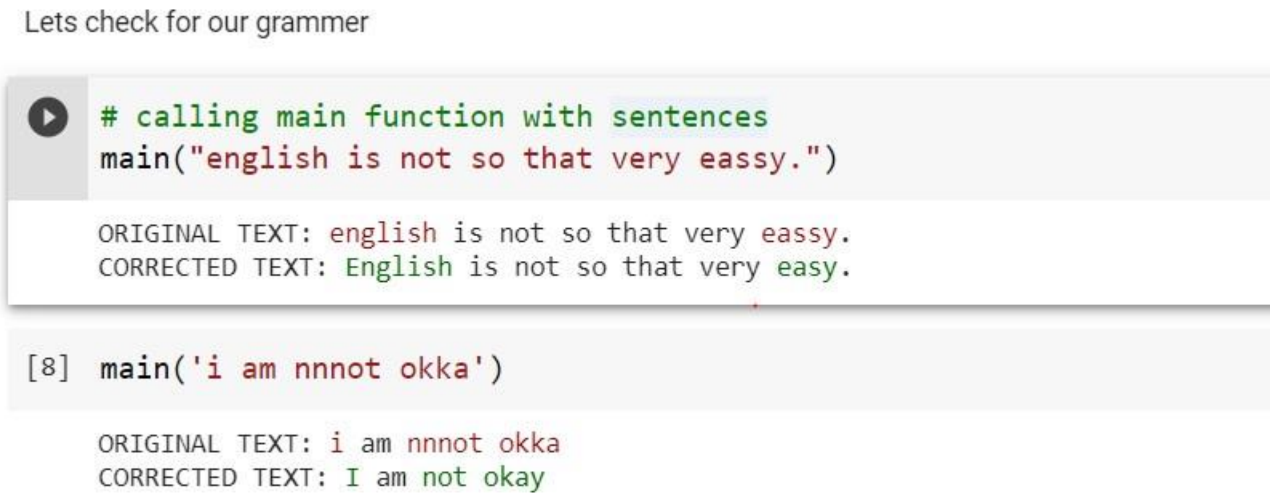
Testing

We have successfully implemeted the program and got desired output.



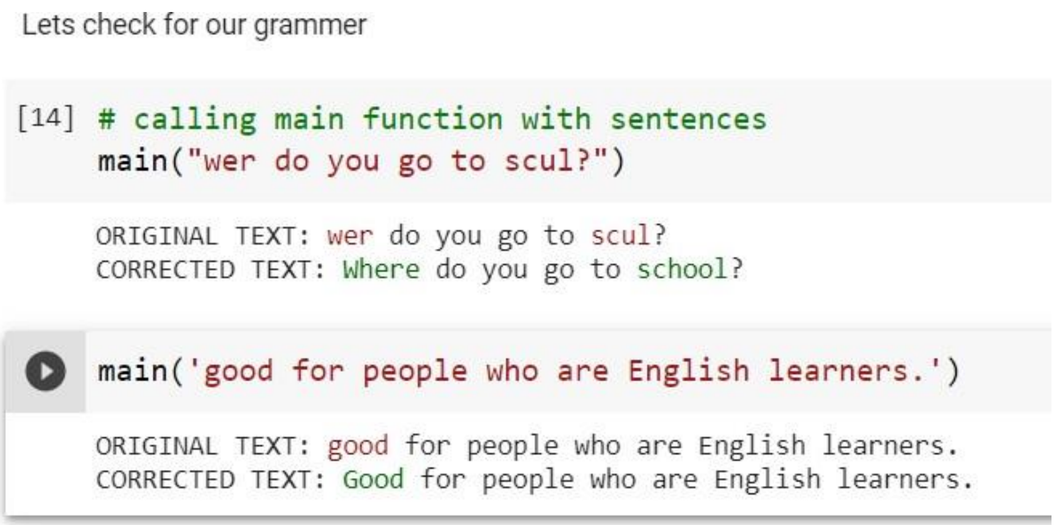
Lets check for our grammer

```
[ ] # calling main function with sentences
    main("Heello, this is Nivedita Mandal fr0om Gorakhpur. CurrRently pursuing BTech on Computer Science. ")

    ORIGINAL TEXT: Heello, this is Nivedita Mandal fr0om Gorakhpur. CurrRently pursuing BTech on Computer Science.
    CORRECTED TEXT: Hello, this is Nivedita Mandal from Gorakhpur. Currently pursuing BTech in Computer Science.

    main('republic Day is celebrate on 26th January . Republicc Day commemorates the date on which the Consttution of india came into force replacing the government

    ORIGINAL TEXT: republic Day is celebrate on 26th January . Republicc Day commemorates the date on which the Consttution of india came into force replacing the government of India Act 1
    CORRECTED TEXT: Republic Day is celebrated on 26th January. Republic Day commemorates the date on which the Constitution of India came into force replacing the government of India Act
```



Lets check for our grammer

```
    # calling main function with sentences
    main("english is not so that very eassy.")

    ORIGINAL TEXT: english is not so that very eassy.
    CORRECTED TEXT: English is not so that very easy.

[8] main('i am nnnot okka')

    ORIGINAL TEXT: i am nnnot okka
    CORRECTED TEXT: I am not okay
```

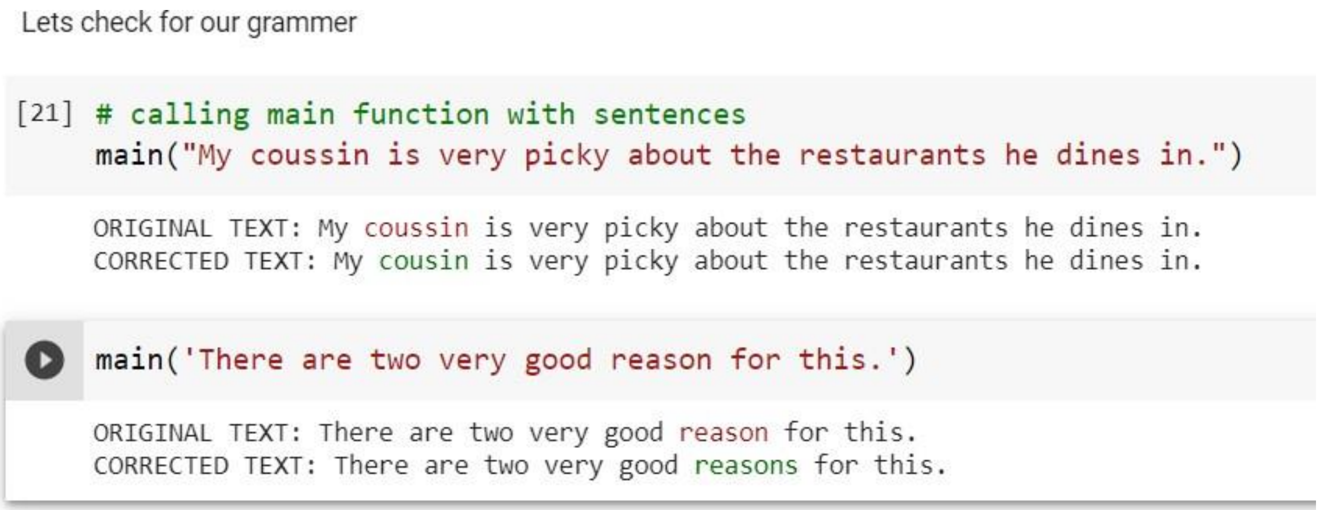

Lets check for our grammer

```
[14] # calling main function with sentences
     main("wer do you go to scul?")
```

```
ORIGINAL TEXT: wer do you go to scul?
CORRECTED TEXT: Where do you go to school?
```

```
main('good for people who are English learners.')
```

```
ORIGINAL TEXT: good for people who are English learners.
CORRECTED TEXT: Good for people who are English learners.
```

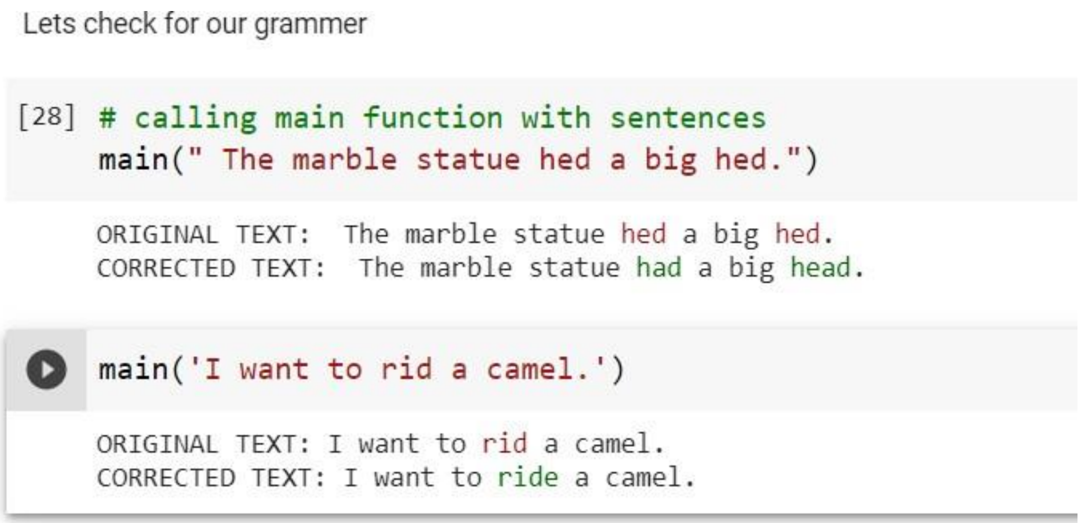

Lets check for our grammer

```
[21] # calling main function with sentences
     main("My coussin is very picky about the restaurants he dines in.")
```

```
ORIGINAL TEXT: My coussin is very picky about the restaurants he dines in.
CORRECTED TEXT: My cousin is very picky about the restaurants he dines in.
```

```
main('There are two very good reason for this.')
```

```
ORIGINAL TEXT: There are two very good reason for this.
CORRECTED TEXT: There are two very good reasons for this.
```

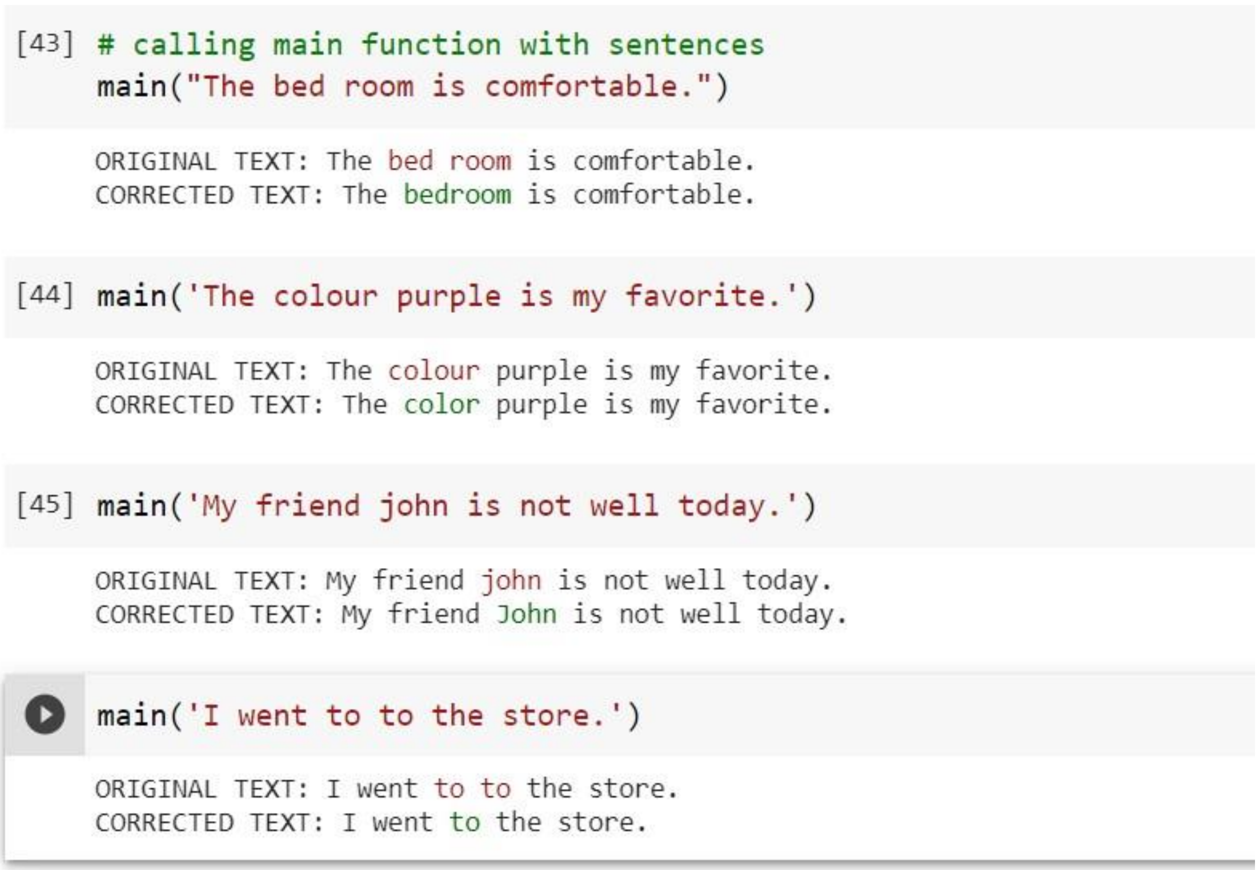

Lets check for our grammer

```
[28] # calling main function with sentences
     main(" The marble statue hed a big hed.")
```

```
ORIGINAL TEXT:  The marble statue hed a big hed.
CORRECTED TEXT:  The marble statue had a big head.
```

```
main('I want to rid a camel.')
```

```
ORIGINAL TEXT: I want to rid a camel.
CORRECTED TEXT: I want to ride a camel.
```



```
[43] # calling main function with sentences
     main("The bed room is comfortable.")
```

```
ORIGINAL TEXT: The bed room is comfortable.
CORRECTED TEXT: The bedroom is comfortable.
```

```
[44] main('The colour purple is my favorite.')
```

```
ORIGINAL TEXT: The colour purple is my favorite.
CORRECTED TEXT: The color purple is my favorite.
```

```
[45] main('My friend john is not well today.')
```

```
ORIGINAL TEXT: My friend john is not well today.
CORRECTED TEXT: My friend John is not well today.
```

```
main('I went to to the store.')
```

```
ORIGINAL TEXT: I went to to the store.
CORRECTED TEXT: I went to the store.
```

Chapter 6: Limitations and Future Scope of the Project

Limitations

1) It can only correct the grammar of sentence or paragraph written in English language.

2) The user should not input more than 600 characters at a time.

3) The sentence should be written in utf-8 format.

4) It cannot write digits instead of spelling out the numbers 0-9 .

Future Scope of the Project

1) We cannot check more than 600 characters at a time .So we need to exceed the character limit.

2) It cannot completely correct the grammar of the sentence. Therefore we need enhance it.

3) It can only correct the grammar of sentence or paragraph written in English language.So we need to correct the grammar for other languages.

4) We need to add a feature named "Numeral Spelling Out" which write digits instead of spelling out the numbers 0-9.

5) The sentence should be written in utf-8 format so we need to add other writing format.

Chapter 6: CONCLUSION

We have successfully completed the project.

Link to Code and executable file:

https://colab.research.google.com/drive/1e7pCsbAw_877r82dTHGMcsx2ED3s6VfA?usp=sharing

Outcome:

It first detects the grammatical errors and contextual errors from a given sentence or paragraph and then gives the corrected text.

Examples:

1) **Spelling :** It corrects a word which is not spelled correctly.

ORIGINAL TEXT: The marble statue had a big hed.

CORRECTED TEXT: The marble statue had a big head.

2) **Misused:** It corrects the confusion between words which sound similar or have a similar spelling.

ORIGINAL TEXT: I want to rid a camel.

CORRECTED TEXT: I want to ride a camel.

3) **Split and Merge:** It corrects the word which was accidentally split in two or vice-versa, two words were accidentally merged into a single word.

ORIGINAL TEXT: The bed room is comfortable.

CORRECTED TEXT: The bedroom is comfortable.

4) **Subject Verb Agreement :**It corrects the subject verb agreement.

ORIGINAL TEXT: The smell of flowers bring back memories.

CORRECTED TEXT: The smell of flowers brings back memories.

5) **Common and Proper Nouns :** It corrects a proper or common noun which is not capitalized.

ORIGINAL TEXT: My friend john is not well today.

CORRECTED TEXT: My friend John is not well today.

6) **Double Words :** It corrects accidental repetition of a word.

ORIGINAL TEXT: I went to to the store.

CORRECTED TEXT: I went to the store.

7) **Indefinite Article:** It corrects confusion between a/an or omission of an indefinite article when it is required .

ORIGINAL TEXT: John is studying for a MBA degree.

CORRECTED TEXT: John is studying for an MBA degree.

8) **Definite Article:** It corrects omission of the definite article ("the") when it is required.

ORIGINAL TEXT: I had time of my life on this vacation.

CORRECTED TEXT: I had the time of my life on this vacation.

9) **Consecutive Nouns :** It corrects wrong usage of two or more nouns in a row, either in possessive form or as modifiers of each other.

ORIGINAL TEXT: Sheryl went to the <u>tickets </u>office.

CORRECTED TEXT: Sheryl went to the ticket office.

10) **Plurality:** It corrects the confusion between the singular and plural form of a noun.

ORIGINAL TEXT: We bought a number of <u>item</u>.

CORRECTED TEXT: We bought a number of items.

11) **Pronouns:** It corrects the wrong pronoun or not using a pronoun when one is required.

ORIGINAL TEXT: I need <u>you </u>help.

CORRECTED TEXT: I need your help.

12) **Comparative Superlative:** It corrects the wrong usage of comparative and superlative structures.

ORIGINAL TEXT: This movie is <u>bad</u> than anything I have seen.

CORRECTED TEXT: This movie is worse than anything I have seen.

13) **Tenses:** It corrects the wrong verb tense or form is being used.

ORIGINAL TEXT: It is important to submitted the paper on time.

CORRECTED TEXT: It is important to submit the paper on time.

14) **Present Simple :** It corrects by applying the present simple tense.

ORIGINAL TEXT: The battery is lasting for only two hours.

CORRECTED TEXT: The battery lasts for only two hours.

15) **Present Perfect:** It corrects by applying the present perfect tense.

ORIGINAL TEXT: I have develop this idea for days.

CORRECTED TEXT: I have developed this idea for days.

16) **Present Progressive:** It corrects by applying the present progressive tense.

ORIGINAL TEXT: Terry has writing a letter at the moment.

CORRECTED TEXT: Terry is writing a letter at the moment.

17) **Past Simple:** It corrects applying the past simple tense.

ORIGINAL TEXT: I go to the store yesterday.

CORRECTED TEXT: I went to the store yesterday.

18) **Past Perfect:** It corrects by applying the past perfect tense.

ORIGINAL TEXT: My parents has never been to Florida before this summer.

CORRECTED TEXT: My parents had never been to Florida before this summer.

19) **Past Progressive:** It corrects by applying the past progressive tense.

ORIGINAL TEXT: He was sell fruit on the side of the road.

CORRECTED TEXT: He was selling fruit on the side of the road.

20) **Future Tense:** It corrects by applying the future tense.

ORIGINAL TEXT: Amy try going to the chess club tomorrow.

CORRECTED TEXT: Amy will try going to the chess club tomorrow.

21) **Subject Verb Agreement:** It corrects mismatch between the plurality of the verb and the subject.

ORIGINAL TEXT: A bouquet of yellow roses lend color and fragrance to the room.

CORRECTED TEXT: A bouquet of yellow roses lends color and fragrance to the room.

22) **The Infinitive:** It corrects using the infinitive form.

ORIGINAL TEXT: She expects it is ready by five.

CORRECTED TEXT: She expects it to be ready by five.

23) **Participles:** It corrects errors related to incorrect usage or incorrect form of participles.

ORIGINAL TEXT: We are looking for employees with proved experience.

CORRECTED TEXT: We are looking for employees with proven experience.

24) **Adverbial Modifiers:** It corrects incorrect use of adverbs instead of adjectives or vice-versa or using the wrong adverb.

ORIGINAL TEXT: She is a beautifully woman.

CORRECTED TEXT: She is a beautiful woman.

25) **Prepositions:** It corrects the wrong preposition.

ORIGINAL TEXT: We arrived to the station.

CORRECTED TEXT: We arrived at the station.

26) **Prepositions:** It corrects the confusion between the usage of the prepositions in, on and at.

ORIGINAL TEXT: She lives in 666 Elm Street.

CORRECTED TEXT: She lives at 666 Elm Street.

ORIGINAL TEXT: The relevant data appeared on the rightmost column.

CORRECTED TEXT: The relevant data appeared in the rightmost column.

ORIGINAL TEXT: They laughed on all the jokes.

CORRECTED TEXT: They laughed at all the jokes.

27) **Beginning Of Sentence Capitalization:** It corrects the first word in a sentence which is not capitalized.

ORIGINAL TEXT: this is not right.

CORRECTED TEXT: This is not right.

28) **Comma Addition:** It adds a comma where there should be one, e.g. after an introductory phrase, between list items and so on.

ORIGINAL TEXT: Students will demonstrate understanding of spoken words syllables and sounds.

CORRECTED TEXT: Students will demonstrate understanding of spoken words, syllables and sounds.

29) **Question Mark Addition:** It corrects by adding a question mark at the end of a sentence .

ORIGINAL TEXT: How did you manage to do it

CORRECTED TEXT: How did you manage to do it?

30) **Vocabulary:** It corrects semantically unsuitable word in a sentence.

ORIGINAL TEXT: Can you make me a favor?

CORRECTED TEXT: Can you do me a favor?

31) **Singular/Pural nouns:** It corrects singular and pural nouns.

ORIGINAL TEXT: Six people lost their <u>life</u> in the accident.

ORIGINAL TEXT: Six people lost their lives in the accident.

32) **Consecutive Nouns:** It corrects consecutive nouns.

ORIGINAL TEXT: Sheryl went to the <u>tickets</u> office.

CORRECTED TEXT: Sheryl went to the ticket office.

33) **Misused Words Correction:** It corrects misused words.

Using its contextual grammar checker, Ginger recognizes the misused words in any sentence and replaces them with the correct ones.

ORIGINAL TEXT: I was <u>wandering </u>if there's any news.

CORRECTED TEXT: I was wondering if there's any news.

34) **Contexual Spelling Correction:** It corrects contextual spelling.

The Grammar Checker is a contextual spell checker which identifies the correction that best fits the meaning of the original sentence. When combined with the Grammar Checker, you can correct entire sentences in a single click. The same misused word will have a different correction based on the context.

ORIGINAL TEXT: The marble statue hed a big hed .

CORRECTED TEXT: The marble statue had a big head.

35) **Phonetic spelling:** It corrects phonetic spelling.
Phonetic spelling mistakes are corrected even if the correct spelling is very different from the way they were originally written.

ORIGINAL TEXT: I like books, exspecaley the classics.

CORRECTED TEXT: I like books, especially the classics.

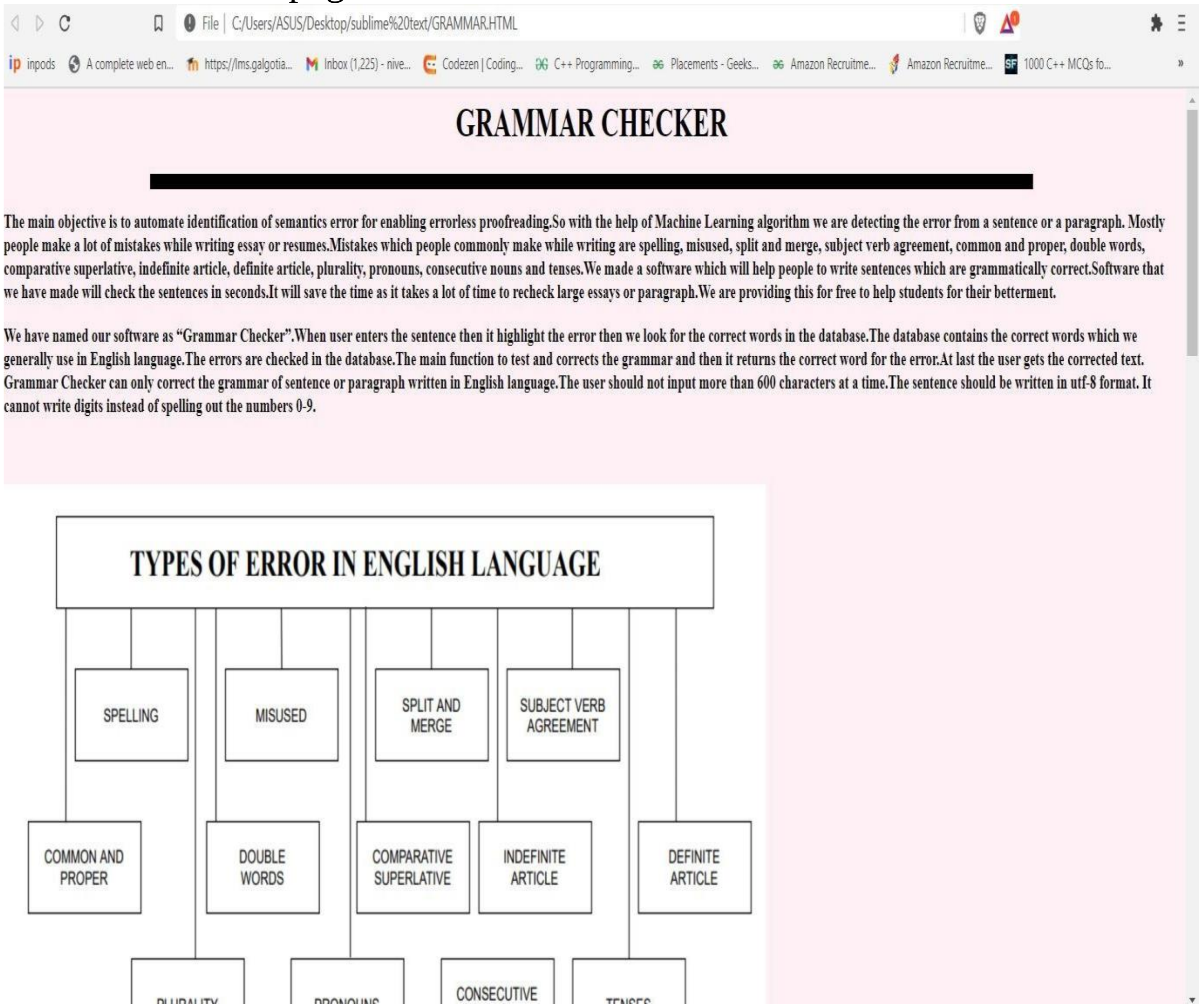
36) **Irregular verb conjugations:** It corrects irregular verb conjugations.
Irregular verb conjugations are corrected as well.

ORIGINAL TEXT: He flyed to Vancouver.

CORRECTED TEXT: He flew to Vancouver.

WEB PAGE

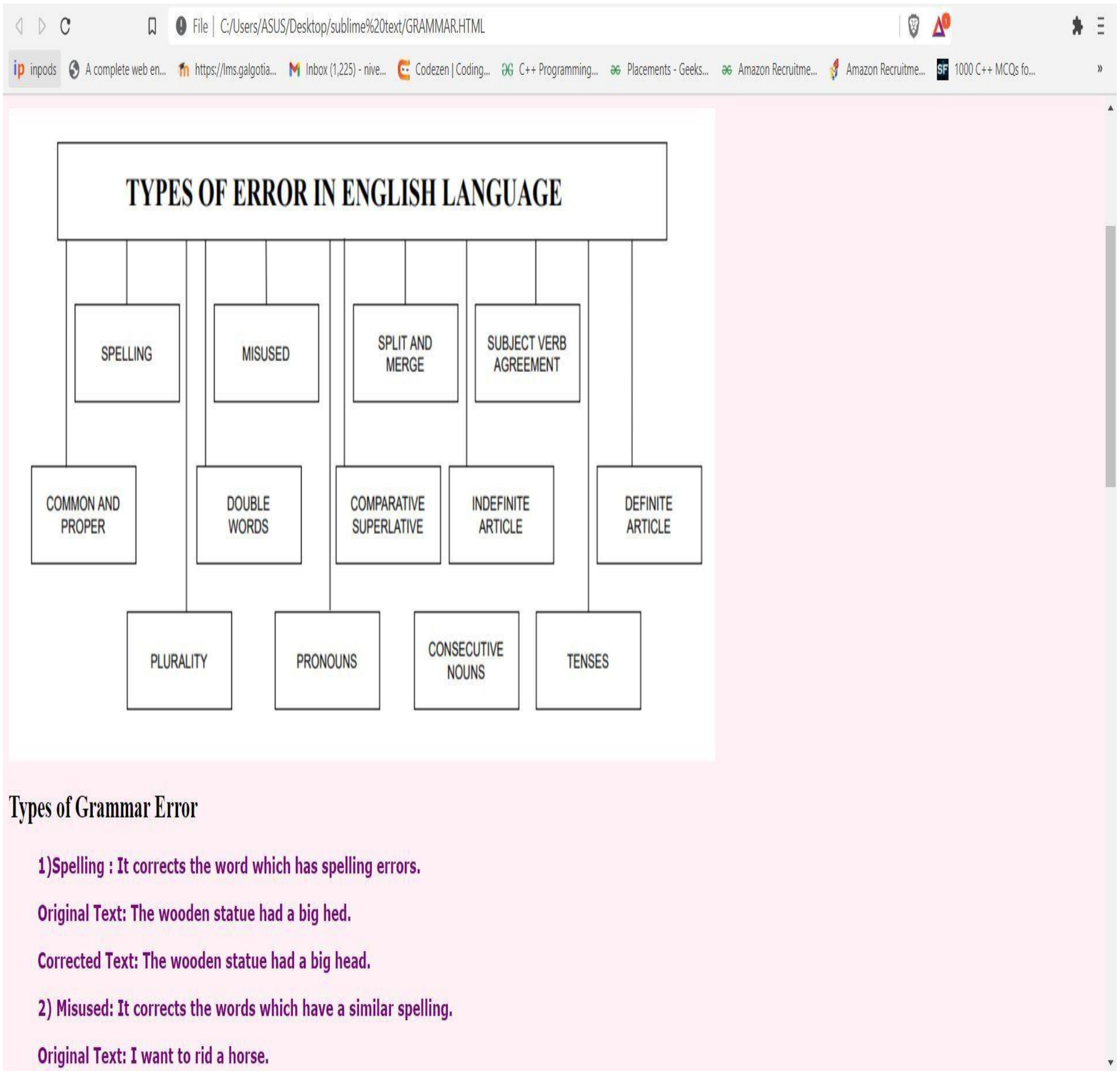
We created a web page to tell about the features to our customers.



GRAMMAR CHECKER

The main objective is to automate identification of semantics error for enabling errorless proofreading.So with the help of Machine Learning algorithm we are detecting the error from a sentence or a paragraph. Mostly people make a lot of mistakes while writing essay or resumes.Mistakes which people commonly make while writing are spelling, misused, split and merge, subject verb agreement, common and proper, double words, comparative superlative, indefinite article, definite article, plurality, pronouns, consecutive nouns and tenses.We made a software which will help people to write sentences which are grammatically correct.Software that we have made will check the sentences in seconds.It will save the time as it takes a lot of time to recheck large essays or paragraph.We are providing this for free to help students for their betterment.

We have named our software as “Grammar Checker”.When user enters the sentence then it highlight the error then we look for the correct words in the database.The database contains the correct words which we generally use in English language.The errors are checked in the database.The main function to test and corrects the grammar and then it returns the correct word for the error.At last the user gets the corrected text. Grammar Checker can only correct the grammar of sentence or paragraph written in English language.The user should not input more than 600 characters at a time.The sentence should be written in utf-8 format. It cannot write digits instead of spelling out the numbers 0-9.



Types of Grammar Error

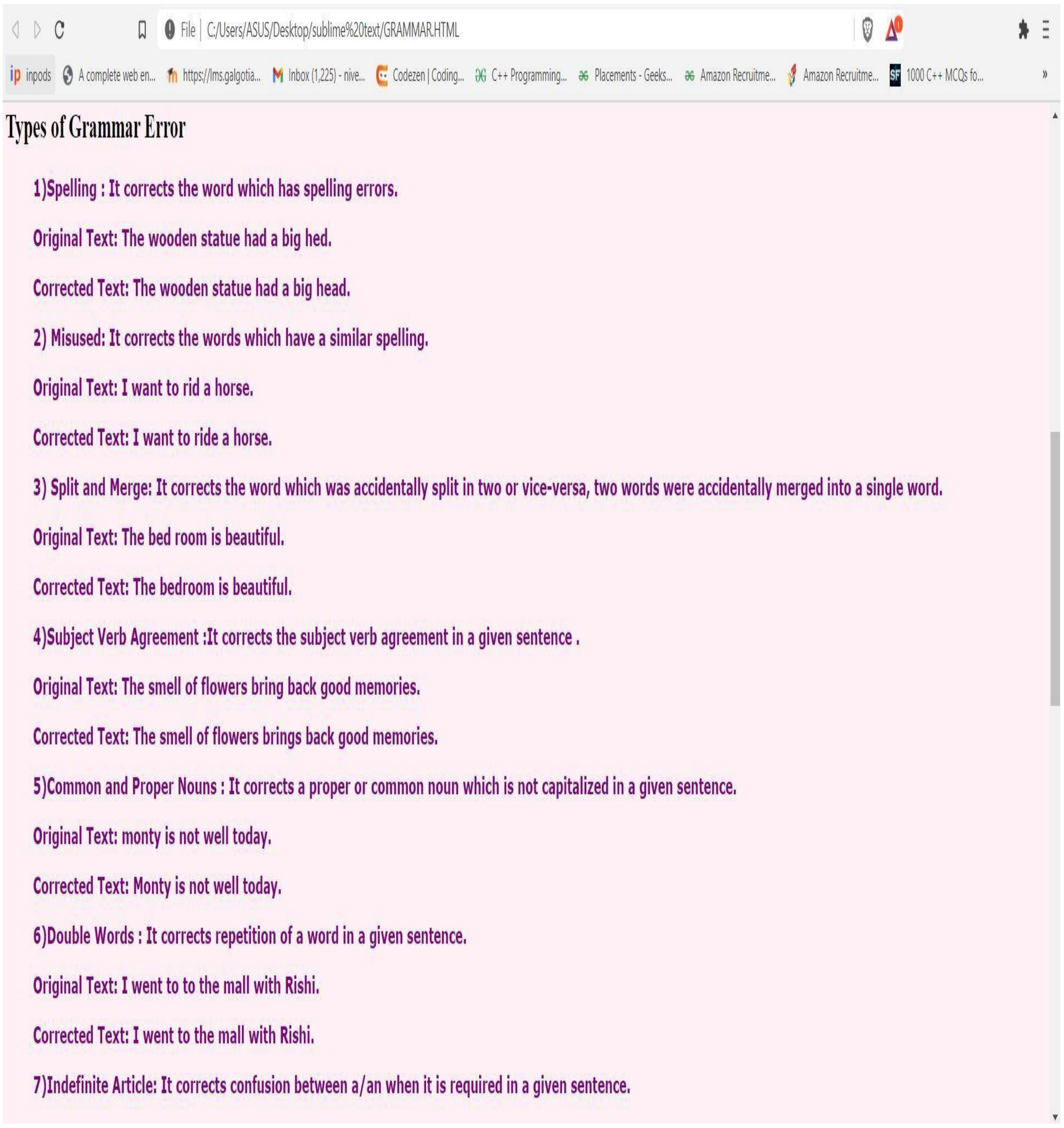
1)Spelling : It corrects the word which has spelling errors.

Original Text: The wooden statue had a big hed.

Corrected Text: The wooden statue had a big head.

2) Misused: It corrects the words which have a similar spelling.

Original Text: I want to rid a horse.



Types of Grammar Error

1)Spelling : It corrects the word which has spelling errors.

Original Text: The wooden statue had a big hed.

Corrected Text: The wooden statue had a big head.

2) Misused: It corrects the words which have a similar spelling.

Original Text: I want to rid a horse.

Corrected Text: I want to ride a horse.

3) Split and Merge: It corrects the word which was accidentally split in two or vice-versa, two words were accidentally merged into a single word.

Original Text: The bed room is beautiful.

Corrected Text: The bedroom is beautiful.

4)Subject Verb Agreement :It corrects the subject verb agreement in a given sentence .

Original Text: The smell of flowers bring back good memories.

Corrected Text: The smell of flowers brings back good memories.

5)Common and Proper Nouns : It corrects a proper or common noun which is not capitalized in a given sentence.

Original Text: monty is not well today.

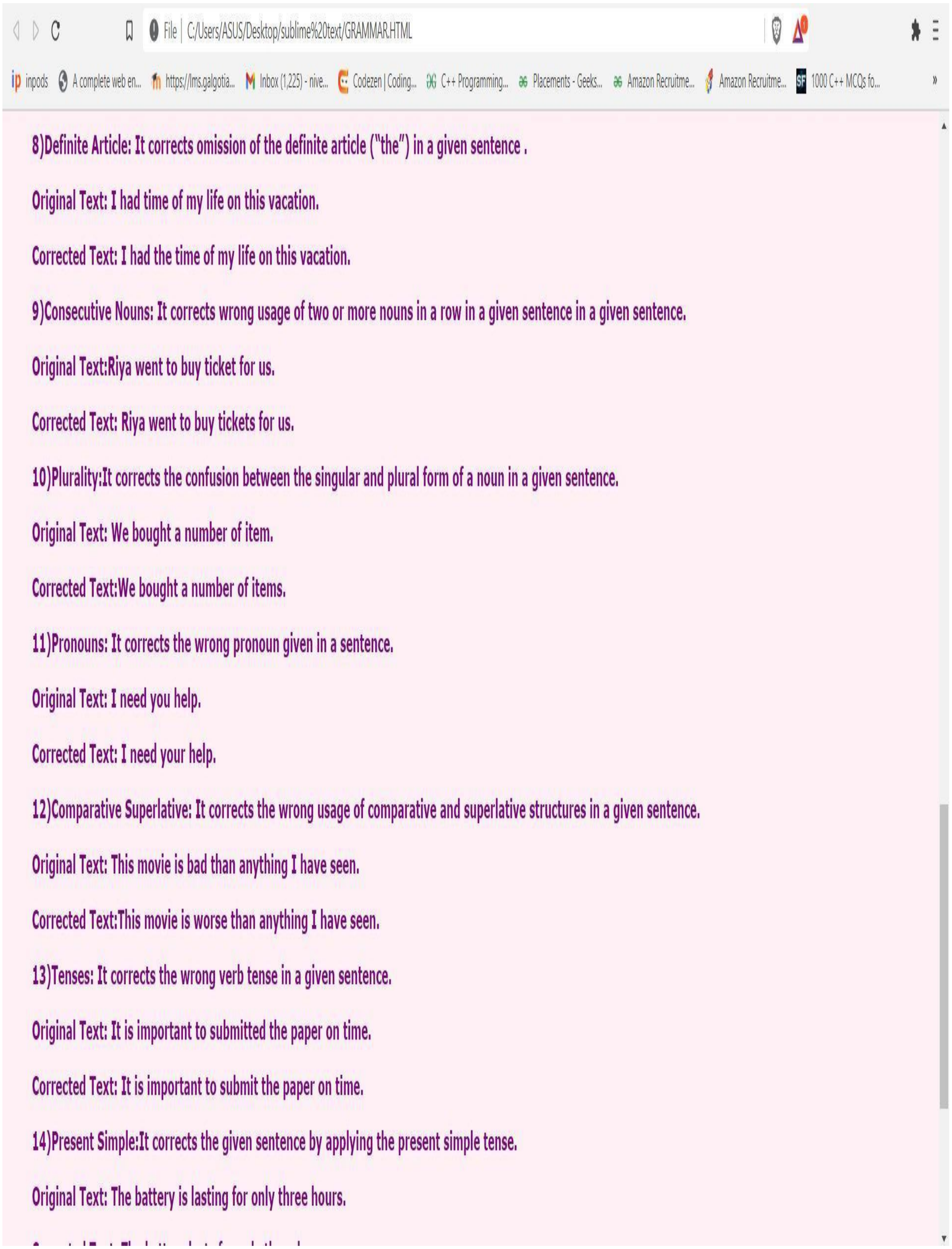
Corrected Text: Monty is not well today.

6)Double Words : It corrects repetition of a word in a given sentence.

Original Text: I went to to the mall with Rishi.

Corrected Text: I went to the mall with Rishi.

7)Indefinite Article: It corrects confusion between a/an when it is required in a given sentence.



File | C:/Users/ASUS/Desktop/sublime%20text/GRAMMAR.HTML

inpods | A complete web en... | https://lms.galgotia... | Inbox (1,225) - nive... | Codezen | Coding... | C++ Programming... | Placements - Geeks... | Amazon Recruitme... | Amazon Recruitme... | 1000 C++ MCQs fo...

8)Definite Article: It corrects omission of the definite article ("the") in a given sentence .

Original Text: I had time of my life on this vacation.

Corrected Text: I had the time of my life on this vacation.

9)Consecutive Nouns: It corrects wrong usage of two or more nouns in a row in a given sentence in a given sentence.

Original Text:Riya went to buy ticket for us.

Corrected Text: Riya went to buy tickets for us.

10)Plurality:It corrects the confusion between the singular and plural form of a noun in a given sentence.

Original Text: We bought a number of item.

Corrected Text:We bought a number of items.

11)Pronouns: It corrects the wrong pronoun given in a sentence.

Original Text: I need you help.

Corrected Text: I need your help.

12)Comparative Superlative: It corrects the wrong usage of comparative and superlative structures in a given sentence.

Original Text: This movie is bad than anything I have seen.

Corrected Text:This movie is worse than anything I have seen.

13)Tenses: It corrects the wrong verb tense in a given sentence.

Original Text: It is important to submitted the paper on time.

Corrected Text: It is important to submit the paper on time.

14)Present Simple:It corrects the given sentence by applying the present simple tense.

Original Text: The battery is lasting for only three hours.

REFERENCES

1. AS Hornby. 1995. Guide to Patterns and Usage in English. (1995)

2. John Lee and Stephanie Seneff. 2008. Correcting Misuse of Verb Forms.. In ACL. 174–182

3. Daniel Naber. 2003. A rule-based style and grammar checker. (2003).

4. Alla Rozovskaya, Kai-Wei Chang, Mark Sammons, and Dan Roth. 2013. The University of Illinois system in the CoNLL-2013 shared task. In Proceedings of the Seventeenth Conference on Computational Natural Language Learning:Shared Task. 13–19.

5. Joachim Wagner, Jennifer Foster, and Josef van Genabith. 2007. A comparative evaluation of deep and shallow approaches to the automatic detection of common grammatical errors. Association for Computational Linguistics

6. PC Wren and H Martin. 2000. English Grammar & Composition. S. Chand & Company Ltd (2000).

7. Chak Yan Yeung and John Lee. 2015. Automatic Detection of Sentence Fragments.. In ACL (2). 599–603.

8. Zheng Yuan. 2017. Grammatical error correction in non-native English. Technical Report. University of Cambridge, Computer Laboratory

9. AM. Wing, and A. D. Baddeley, Spelling errors in handwriting: a corpus and distributional analysis, Cognitive processes in spelling, London: Academic Press, pp.251-285, 1980

10. Wei Zhao, Liang Wang, Kewei Shen, Ruoyu Jia, Jingming Liu,Improving Grammatical Error Correction via Pre-Training a Copy-Augmented Architecture with Unlabeled Data,Yuanfudao Research / Beijing, China ,2019

11. Emily M Bender, Dan Flickinger, Stephan Oepen, Annemarie Walsh, and Timothy Baldwin. 2004.

Arboretum: Using a precision grammar for grammar checking in CALL. In Instil/icall symposium 2004

12. Chris Brockett, William B Dolan, and Michael Gamon. 2006. Correcting ESL errors using phrasal SMT techniques. In Proceedings of the 21st International Conference on Computational Linguisticsand the 44th annual meeting of the Association for Computational Linguistics. Association for Computational Linguistics, 247–256.
13. Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In Proceedings of the 25th international conference on Machine learning, pages 160–167. ACM.
14. Daniel Dahlmeier and Hwee Tou Ng. 2011. Grammatical error correction with alternating structure optimization. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1. Association for Computational Linguistics, 915–923
15. Bryan McCann, James Bradbury, Caiming Xiong, and Richard Socher. 2017. Learned in translation: Contextualized word vectors. In Advances in Neural Information Processing Systems, pages 6294–6305.
16. Bestgen, Yves, and Sylviane Granger. "Categorising spelling errors to assess L2 writing." International Journal of Continuing Engineering Education and Life-Long Learning 21.2-3 (2011): 235- 252
17. Peters, Matthew E., et al. "Deep contextualized word representations." arXiv preprint arXiv:1802.05365 (2018).
18. Staffs Keele et al. 2007. Guidelines for performing systematic literature reviews in software engineering. In Technical report, Ver. 2.3 EBSE Technical Report. EBSE. sn.
19. Akshat Kumar and Shivashankar Nair. 2007. An artificial immune system based approach for English grammar checking. Artificial immune systems (2007), 348–357

20. Guihua Sun, Xiaohua Liu, Gao Cong, Ming Zhou, Zhongyang Xiong, John Lee, and Chin-Yew Lin. 2007. Detecting erroneous sentences using automatically mined sequential patterns. In ACL. 23–30