Dissertation on
Information Extraction from Scanned Documents

Thesis submitted towards partial fulfilment
of the requirements for the degree of

Master of Technology in IT (Courseware Engineering)

Submitted by
Palash Priya Das

EXAMINATION ROLL NO.: M4CWE23002
UNIVERSITY REGISTRATION NO.: 160379 of 2021-2022

Under the guidance of
Prof. Dr. Matangini Chattopadhyay
School of Education Technology
Jadavpur University

Course affiliated to
Faculty of Engineering and Technology
Jadavpur University
Kolkata-700032
India

2023

M.Tech. IT (Courseware Engineering)
Course affiliated to
Faculty of Engineering and Technology
Jadavpur University
Kolkata, India

CERTIFICATE OF RECOMMENDATION

This is to certify that the thesis entitled **"**Information Extraction from Scanned Documents**"** is a bonafide work carried out by Palash Priya Das under our supervision and guidance for partial fulfillment of the requirements for the degree of Master of Technology in IT (Courseware Engineering) in School of Education Technology, during the academic session 2022-2023.

SUPERVISOR
School of Education Technology
Jadavpur University,
Kolkata-700 032

DIRECTOR
School of Education Technology
Jadavpur University,
Kolkata-700 032

DEAN - FISLM
Jadavpur University,
Kolkata-700 032

M.Tech. IT (Courseware Engineering)
Course affiliated to
Faculty of Engineering and Technology
Jadavpur University
Kolkata, India

CERTIFICATE OF APPROVAL **

This foregoing thesis is hereby approved as a credible study of an engineering subject carried out and presented in a manner satisfactory to warranty its acceptance as a prerequisite to the degree for which it has been submitted. It is understood that by this approval the undersigned do not endorse or approve any statement made or opinion expressed or conclusion drawn therein but approve the thesis only for purpose for which it has been submitted.

Committee of final examination for evaluation of Thesis

** Only in case the thesis is approved.

DECLARATION OF ORIGINALITY AND COMPLIANCE OF ACADEMIC ETHICS

I hereby declare that this thesis contains literature survey and original research work by the undersigned candidate, as part of her **Master of Technology in IT (Courseware Engineering)** studies.

All information in this document has been obtained and presented in accordance with academic rules and ethical conduct.

I also declare that, as required by this rule and conduct, I have fully cited and referenced all materials and results that are not original to this work.

NAME:

EXAMINATION ROLL NUMBER:

THESIS TITLE:

SIGNATURE: DATE:

Acknowledgement

Firstly,I would like to express my sincere gratitude to my supervisor Prof. Dr. Matangini Chattopadhyay for her continuous support in my thesis work and related research. Her guidance and valuable suggestions always helped me at time of research and writing this thesis. I could not have imagined having a better advisor and mentor for my Master Degree study. I am very much thankful to her for the motivation and support given during the entire duration of research work.

Besides this, I would like to thank Dr. Saswati Mukherjee and Mr. Joydeep Mukherjee for their continuous encouragement and support during my entire period of study in the School of Education Technology. I would like to express my gratitude to the entire staff, Lab Assistant and research workers specially, Dr. Susovan Jana & Mr. Hindol Bhattacharya and all those who were associated with my work for their co-operation and opinion. I am very grateful to all my classmates of M. Tech IT(Courseware Engineering) and also Master in Multimedia Development courses for their suggestions and continuous support.

I would like to thank my parents for always supporting me in every ups and downs during my entire period of work. I am also thankful to my friends and well wishers who always have faith in me.

_____

Palash Priya Das
Examination Roll No: M4CWE23002
Univ.Registration No. – 160379 of 2021-2022
M.Tech IT(Courseware Engineering)
School of Education Technology
Jadavpur University, Kolkata – 700032

Dedicated to,
My Parents

Contents

List Of Figures

List of Abbreviations

Abbreviations

OCR - Optical Character Recognition

Document AI - Document Artificial Intelligence

PRLib - Pre - Recognize Library

KV Pair - Key – Value Pair

Auth.- Authentication

AWS- Amazon Web Services

NLP - Natural Language Processing

CRNN - Convolutional Recurrent Neural Network

CTC – Connectionist Temporal Classification

LKE – LearnITy Knowledge Engine

VM Instance – Virtual Machine Instance

Executive Summary

One of the most important parts of education system is documents. Today in order to carry out research work, students and teachers need already published research papers. These research papers are available in the form of digital contents, mostly as pdf, docx, etc.

Text extraction from documents is sometimes not possible for the cases where we cannot copy text to put it into another documents. In pdf files or scanned documents we cannot select or edit or perform copy paste or search operation anything. The solution to this problem is text extraction using remote services and using Programmatic Access to make our documents ready.

This research work presents how text is extracted from documents, specially in question papers and various other files. Here, using Tesseract or Eclipse IDE we have extracted text from various documents and pdf files. We have also used Postman, Curl to fetch or extract text . Also we have use Amazon Textract and Google Cloud Console to fetch or extract text and their programmatic access.

CHAPTER 1

1.0 Introduction

1.0. Overview

Text analysis is the process of automatically organizing and evaluating unstructured text (documents, customer feedback, social media, email, etc.). It uses machine learning with Natural Language Processing (NLP) to break down text and "understand" it, in order to gather information, structure data, and reach conclusion.

The most useful text analysis techniques are text extraction and text classification, which can help you quickly glean data-driven insights at scale.

Text extraction, often referred to as keyword extraction, uses machine learning to automatically scan text and from unstructured data like news articles, surveys, and customer service tickets.

Text extraction can be used to:-

- Extract entities
- Extract specific information
- Extract keywords
- Summarize a text or document
- Clean data

1.2 Problem Statement

To extract text from scanned documents using Tesseract using Programmatic Access using Amazon Textract and Google Cloud Console or Document AI

1.3 Objectives

The objectives of my research work are as follows –

- ❖ To study Optical Character Recognition.
- ❖ To study research papers on text extraction from scanned documents.
- ❖ Learn Java programming for implementation.
- ❖ Learn JavaScript and XML for implementation.
- ❖ To get familiarity with LearnITY Knowledge Engine (LKE) for text extraction.

- ❖ To find confidence score in Amazon Textract and programmatically access with text extraction of the documents in Google Cloud Console

1.4 Assumptions and Scopes

1.4.1 Assumptions

- ❖ Developers must have good network connection with laptops or PC.
- ❖ Developers must have good knowledge of Java , Javascript and XML for programmatic access.

1.4.2 Scopes

- ❖ To do text extraction from image files such as JPEG, JPG files
- ❖ To learn about OCR, Java and Javascript

1.5 Concept and Problem Analysis

Optical Character Recognition (OCR) is sometimes referred to as text recognition. An OCR program extracts and repurposes data from scanned documents, camera images and image-only pdfs. OCR is easy for artificial intelligence (AI) systems. The concept is to upload an image or pdf on the console or IDE, then we have to extract text. We can analyse the text in Greyscale, Skew Rotation and various other methods. There are some ideas which are already proposed as this is a vast area of research and still going on. Information Extraction (IE) is the automatic retrieval of specific information related to a selected topic from a body or bodies of text. IE is used in NLP to extract structured text from unstructured text. IE is a challenging task for artificial intelligence (AI) systems.

The main benefit of OCR technology is that it simplifies the data-entry process by creating effortless text searches, editing and storage. OCR allows businesses and individuals to store files on their computers, laptops and other devices, ensuring constant access to all document.

The benefits of employing OCR technology include the following:

- Reduce costs
- Accelerate workflows
- Automate document routing and content processing
- Centralize and secure data (no fires, break-ins or documents lost in the bank vaults)

1. *Tesseract*

Tesseract is an open source Optical Character Recognition. It extracts text from images and documents without a text layer and outputs these documents into a file, pdf, docx, etc. This software is runnable on Windows and Linux also.

The image quality can be improved by –

- Rescaling
- Binarisation
- Noise Removal
- Dilation/Erosion
- Rotation/Deskewing
- Borders
- Transparency or Alpha Channel
- Tools or Libraries

KV Pair (Key – Value Pair Extraction) is at the heart of document processing. They are two related items, a key and a value. Key defines the data and this s fixed and the value is variable which describes the key.

2. Postman

Postman is a scalable API tool for quickly integrating images into pipeline. The five reasons that we use postman are : –

- ✓ Accessibility – To use Postman tool, we would need to log in to their accounts which is easy to access files anytime and anywhere.
- ✓ Use of collections – Postman allows users to create collections of text for their API calls
- ✓ Collaboration – Collections and environments can be imported or exported.
- ✓ Creating Environments – We are using multiple environments which helps in less repetition of tests.
- ✓ Creation of Tests – Test checkpoints such as verifying HTTP response status which can be added to each Postman API calls.

3.Paddle Paddle

This is an OCR framework which provides multilingual practical OCR tools for application and training different models in a few lines of code.Paddle OCR offers two different models –

a. Lightweight Models

b. Server Models

4. Google Document AI

This automates data processing of documents at a scale. This is built from the decades of AI research of Google to provide detailed information regarding a particular document beyond words. Document AI incorporates a human-in-the-loop concept which would allow users to suggest mistakes within the document retrieval.

5.Amazon Textract

This is a scalable document analysis service. We can extract printed text, handwritten text and structured data from documents like financial reports, medical records, tax forms and so on. This is also capable of processing receipts, invoices or expense related documents at scale. This uses deep learning technology. We can extract data from document without any templates or configuration. Amazon Textract can extract printed text and handwritten text, in Spanish, Italian, French, Portuguese and German, ASCII Symbols, etc.

1.6 Organization of the Thesis

- Chapter 1 – This chapter contains the introduction of the thesis which includes overview , problem statement, objectives, assumptions, scopes, concept and problem analysis
- Chapter 2 – It includes all the literature surveys done to carry out the research work.
- Chapter 3 – It includes proposed approach of the softwares that we have use to extract text.
- Chapter 4 – This chapter contains the implementation and result.
- Chapter 5 – This chapter describes the conclusion and scope of future works.
- Chapter 6 – All the references are given.
- Appendix A – This part contains system requirements, software requirements, programming requirements and download speed.
- Appendix B – This part contains all the codes that have been used here.

CHAPTER 2

2.0. Literature Survey

Chandni Kaundilya, Diksha Chawla, Yatin Chopra [1] have suggested that many students, engineers and doctors need various images according to their varying needs. They have to access images based on its primitive features or associated text. Text in those images can provide meaningful information. Tesseract is currently the most accurate optical character recognition invented by HP Labs and is currently owned by Google. Here, extraction of text is done by using text localization, segmentation and binarization techniques. Similar processes are done for coloured image. Also, they discuss about how text extraction is done for ASCII text.

Rominkumar Busa, Shahira K C [2] said that an accuracy of 30% is achieved after they used Convolutional Recurrent Neural Network (CRNN) with Connectionist Temporal Classification (CTC) loss methods and by using this method they have gained an accuracy of 48%. They also suggested that the efficiency of this method on chart image text and other small text images will improve the accuracy of further upto 68%.

Kamrul Hasan Talukder , Tania Mallick [3] proposed that the extracted text is converted into greyscale image. Also, they have used detection, localization, tracking, extraction, enhancement, and recognition of the text from images. They claimed that their proposed method is superior compared to some recent approaches. They also used connected components for text recognition.

Anand Kumar, Pardeep Singh and Kusum Lata [4] suggested that text extraction can be done not only on images but also on handwritten documents and printed medical reports. They have also said that AWS Text Extract, Google Vision and Google's Cloud Console or Vision have extracted text with an accuracy of 95%. But they have claimed that they have achieved an accuracy of 95% for Amazon Textract for printed dataset and 89.42% for handwritten IAM dataset.

Anubhav Kumar [5] has proposed an algorithm for text extraction for complex images. This is a robust and efficient algorithm using edge map generation, Line edge detection mask, text area segmentation using vertical and the horizontal projection using Heuristic Filtering. He has claimed that this method is useful and is better than other proposed algorithms. He claimed that other pre processed methods are effective and cannot be used on video frames but his method is useful for video frames. He obtained an

accuracy of 99.61% and precision rate is 96.20% having an average computational time of 1.31 second/frames.

Yang Liu, Yonghong Song, Yuanlin Zhang, Quan Meng [6] proposed video texts method which has high level information contributing to video indexing and retrieval. Basically they have dealt with Chinese characters. Firstly, candidate text regions are detected by a wavelet based algorithm. Secondly, horizontal, vertical, slant, curve or arc text lines are merged by colour and space relationship based on arrangement structures of multi-oriented text lines in these candidate regions. And then character segmentation will automatically choose best fit strategies based on structure analysis, due to the complex structures (single, up down, left-right, encircling) of Chinese characters. They have obtained an accuracy of 99%.

CHAPTER 3

3.0 Proposed Approach

3.1. Postman

The work has been done on GET, POST requests.

Working with GET Request

- The following URL has been used https://jsonplaceholder.typicode.com/users.
- In the workspace firstly set up HTTP request to GET.
- In the request URL field https://jsonplaceholder.typicode.com/users then click on SEND. Then we achieve “200 OK” message.

Working with POST Request

- POST request are different from GET request as there is data manipulation with the user adding data to the endpoint. The same data has been used in GET request
- A new tab is clicked to create a new request
- In the new tab we have set up our request to POST.
- The following URL https://jsonplaceholder.typicode.com/users is used.

- Then we switched to Body tab
- In Body tab we clicked Raw and then click on JSON
- The user result has been copy and pasted from the previous GET request
- The Send button is clicked and “Status 201” Created is displayed
- We can parameterize requests where we set up HTTP request again to GET
- This link is written – https://jsonplaceholder.typicode.com/users.
- The first part of the link is used such as {{url}}. Now we click on Send.

- Now we click on Eye Icon (in this icon we see the new collections that a developer has created) where we get new collection to edit we set the variable to a global environment.
- In variable again we set the name to url https://jsonplaceholder.typicode.com.
- The variable name is created.
- The close button is clicked if we see the next screen.
- We are switched to Tests tab. Now we get snippet codes(snippet codes are those codes where we get the code body and we can edit or input our details).
- From the snippets section we choose “Status Code : Code is 200”.
- From Snippets section click on “Response body : JSON value check”.
- Also we have created collections by the same step of putting URL and by clicking on New Collections Icon.
- Also we have created a new token with the help of Google Cloud API with POSTMAN.

3.2. Google Document AI

- In Google Document AI firstly create an account.
- Then the project and project ID has been created.
- Then we have to select API & Services option after that we have to enable API & services.
- Then the API has been enabled.
- VM Instances(Virtual Machine Instance) page has popped up where we have to create Instances Page.
- The project has been choosed from the projects which have been created by an user.
- The text has been extracted from the document.
- After extraction of text we have to choose Enable Option
- We have to input our data to the Service Accounts Page.
- After inputting our data we have to select Processor page where we have to select Parser Processor.

3.3. Amazon Textract

- In this section you have to create an account to the user.
- We have to open Amazon Textract Console
- Then we arrive to Console Home Page of AWS Console
- Then we can extract text.
- We have to go to the User then choose Add User.
- The following details have been given like name, id ,etc.
- Here security credentials has also been given..

CHAPTER 4

4.0 Experimentations and Results

1.LKE

LKE(LearnIT Knowledge Engine) is that web browser where one can upload the documents and also images and perform text extraction from the documents . Aunwesha Knowledge Technologies Pvt. Ltd. has a product named LKE. In this research work LKE has been used for text extraction purposes.

The first image appears due to the following reasons-

- ✓ Firstly we have to open the browser
- ✓ Secondly we have to open the CoreAdmin by giving the following link – https://www.localhost:8081/Information%20Extraction/coreadministration.2.Login.
- ✓ Then the Core Admin page opens

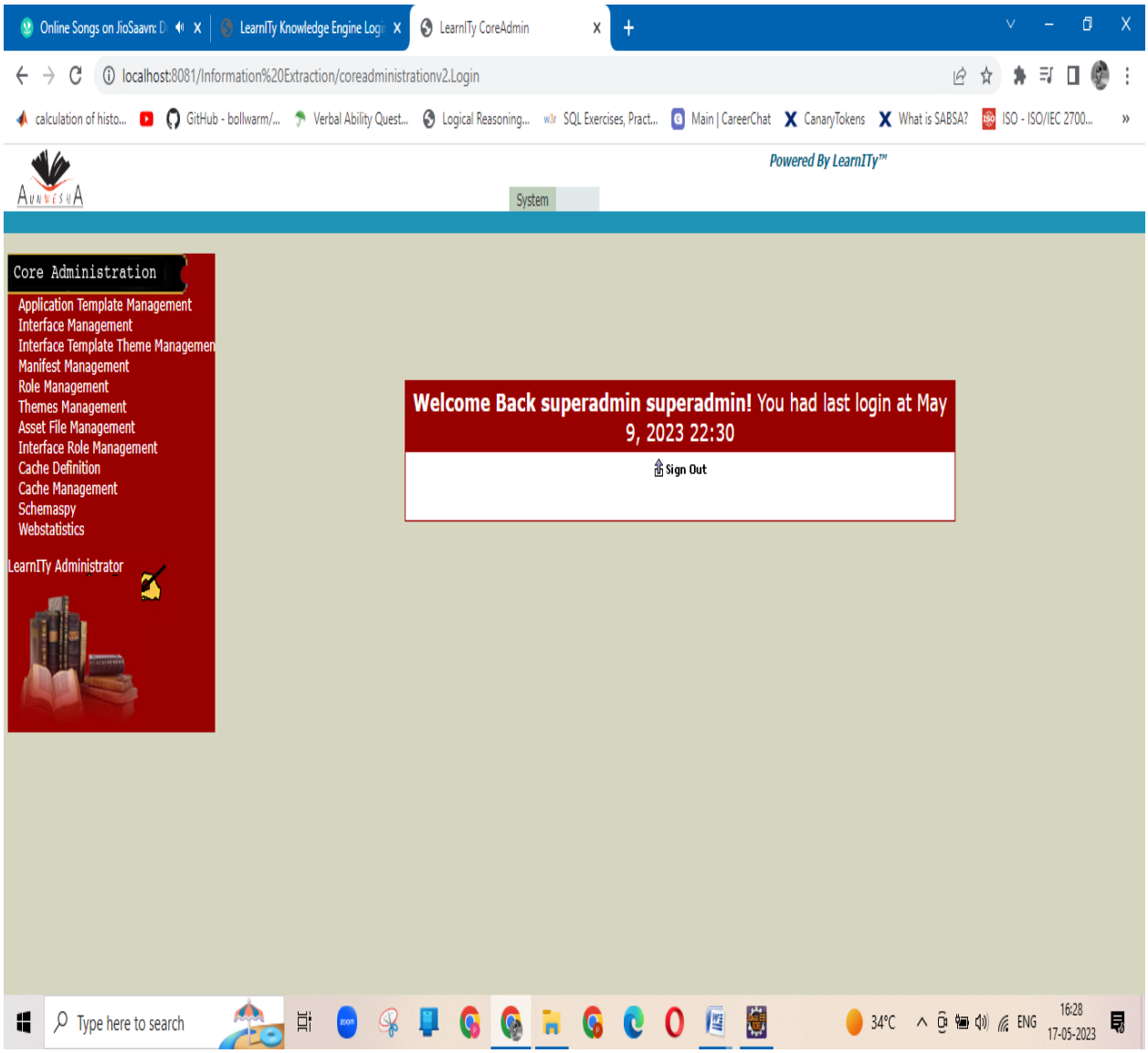


Figure 1: - Admin Page

Here in this figure we opened the Admin Page of LKE (LearnITy Knowledge Engine) to perform text extraction.

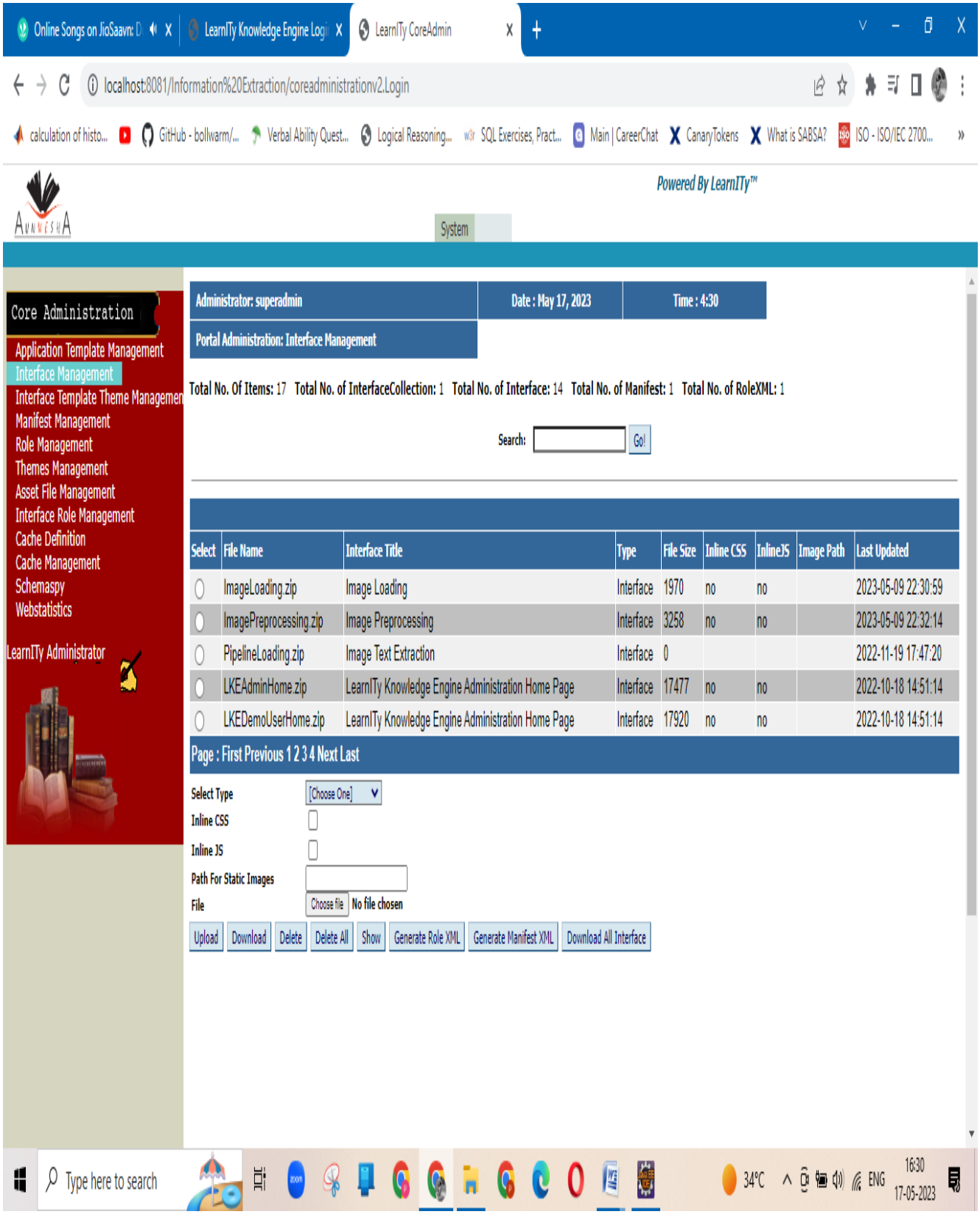


Figure 2:- Interface Adjustments

Here in this figure we are adjusting the interface part.

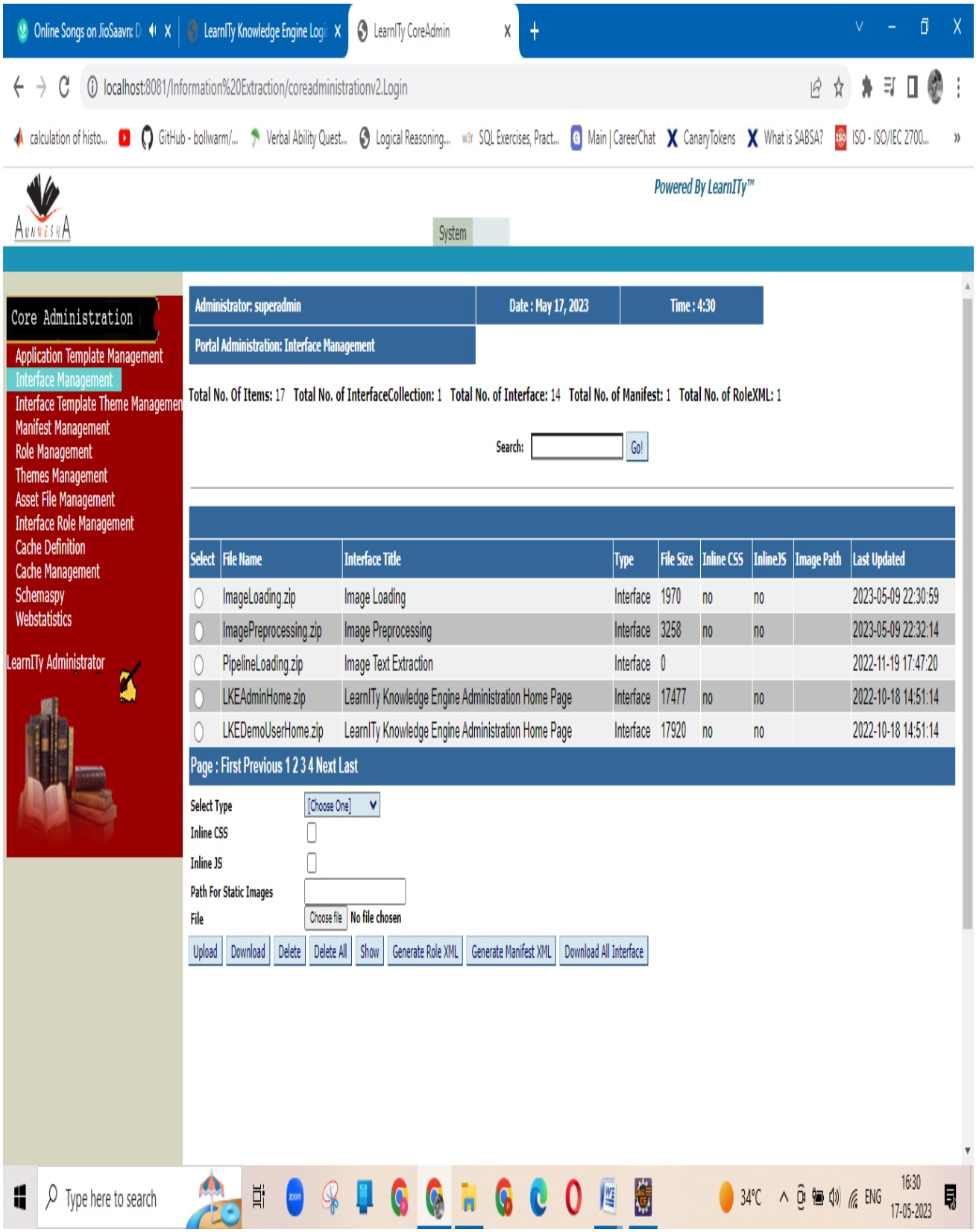


Figure 3:- Uploading of interface.xml and JS files

Here in this figure how the interface and JS files are uploaded is shown

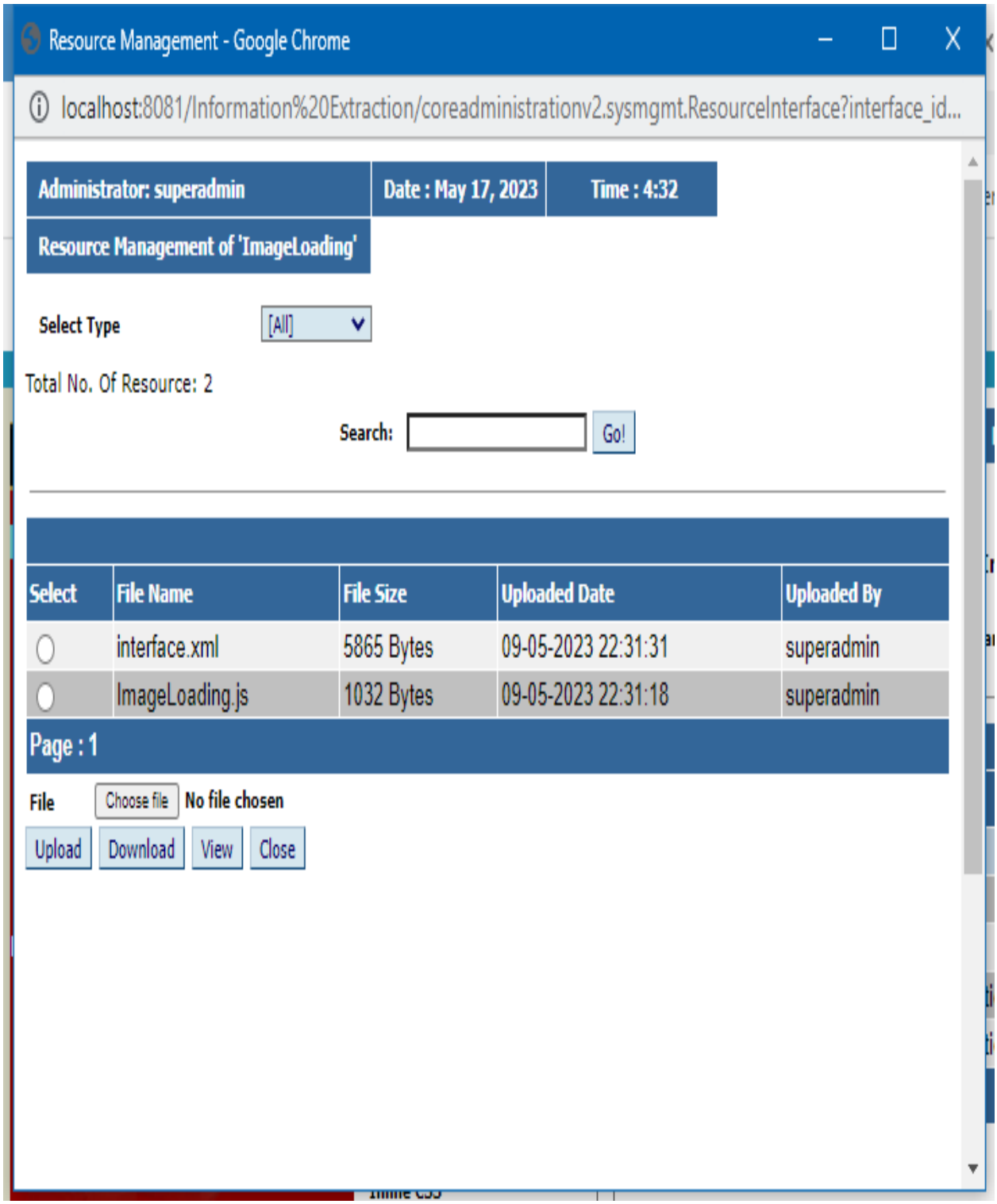


Figure 4:- Uploading of interface.xmls and Java Files

Here in this figure how the interface files are uploaded is shown.

Currently Defined Images

Image Title	Image Description	Image File Type	Image File Metadata	Width of an image	Height of an image	Bits per pixel	Resolution of an image	Date Image Saved	Comment
PHD Maths	JPEG	JPEG						2023-03-21 16:36:51	Good
Bengali Scanned Image	JPEG	JPEG						2023-01-27 21:02:04	Good
Medical Extracted Report	JPEG	JPEG						2023-01-07 17:25:36	Good
Calendar	PNG	JPEG						2023-01-05 20:59:43	Good
Colored Poster	JPEG	JPEG						2023-01-07 17:29:59	Good
Qp Sample	JPEG	JPEG						2022-12-17 21:29:06	Good

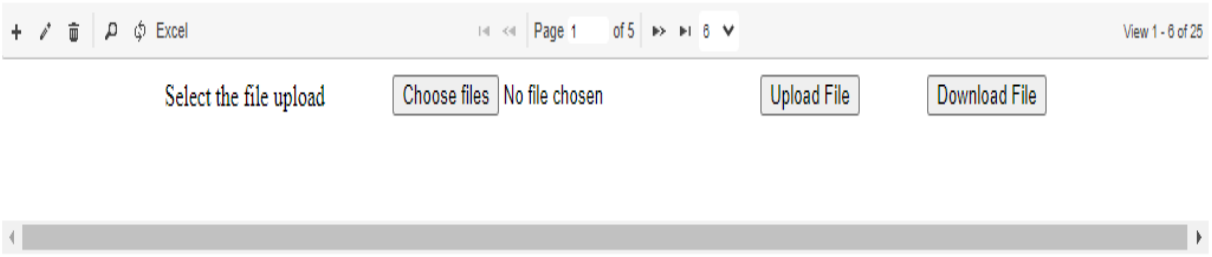


Figure 5:- Image Loading Page

- Here in this figure how the image is uploaded into LKE website is shown.

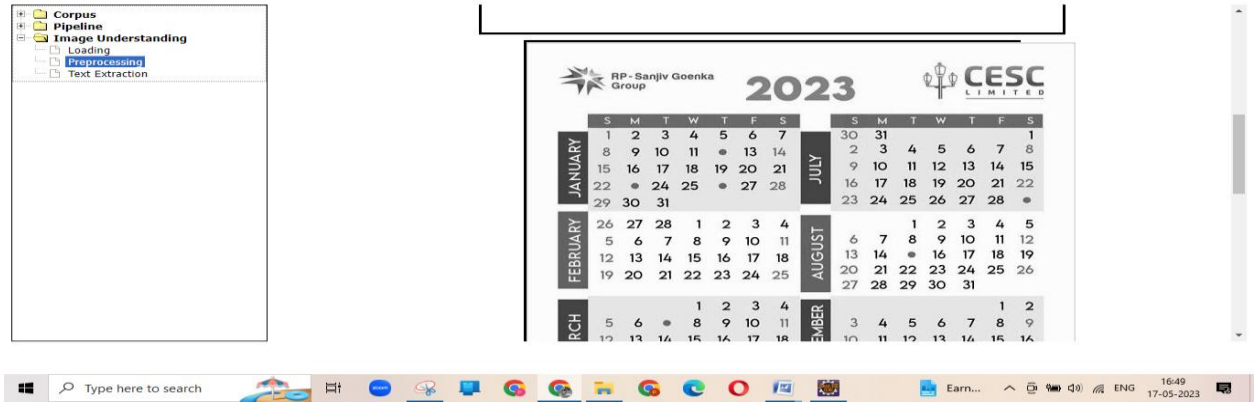


Figure 6:- Image upper part in Image Preprocessing Page

Here in this figure we are giving upper part of the Image(Calendar)

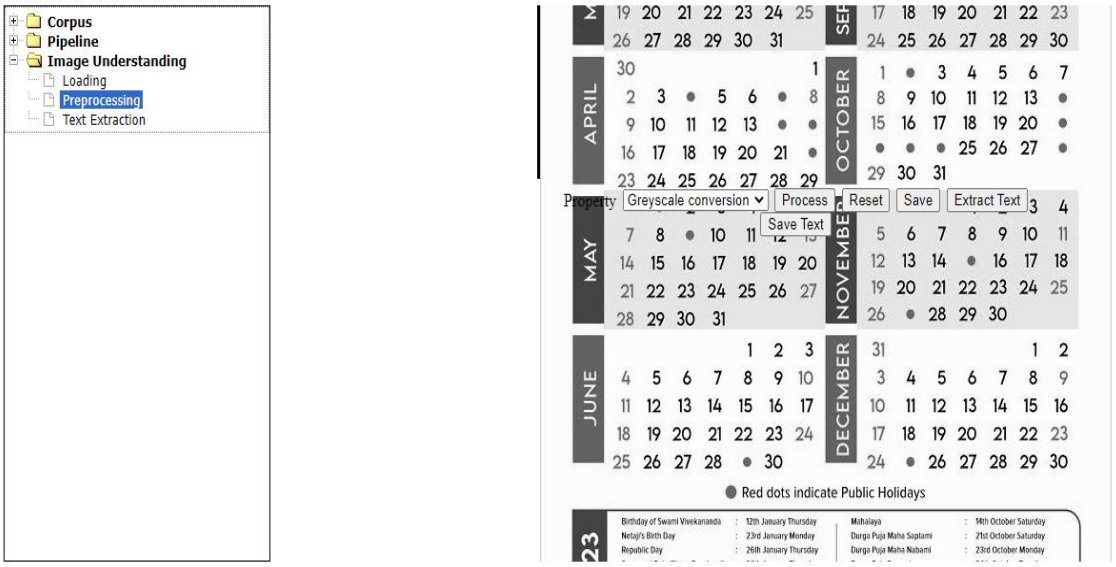


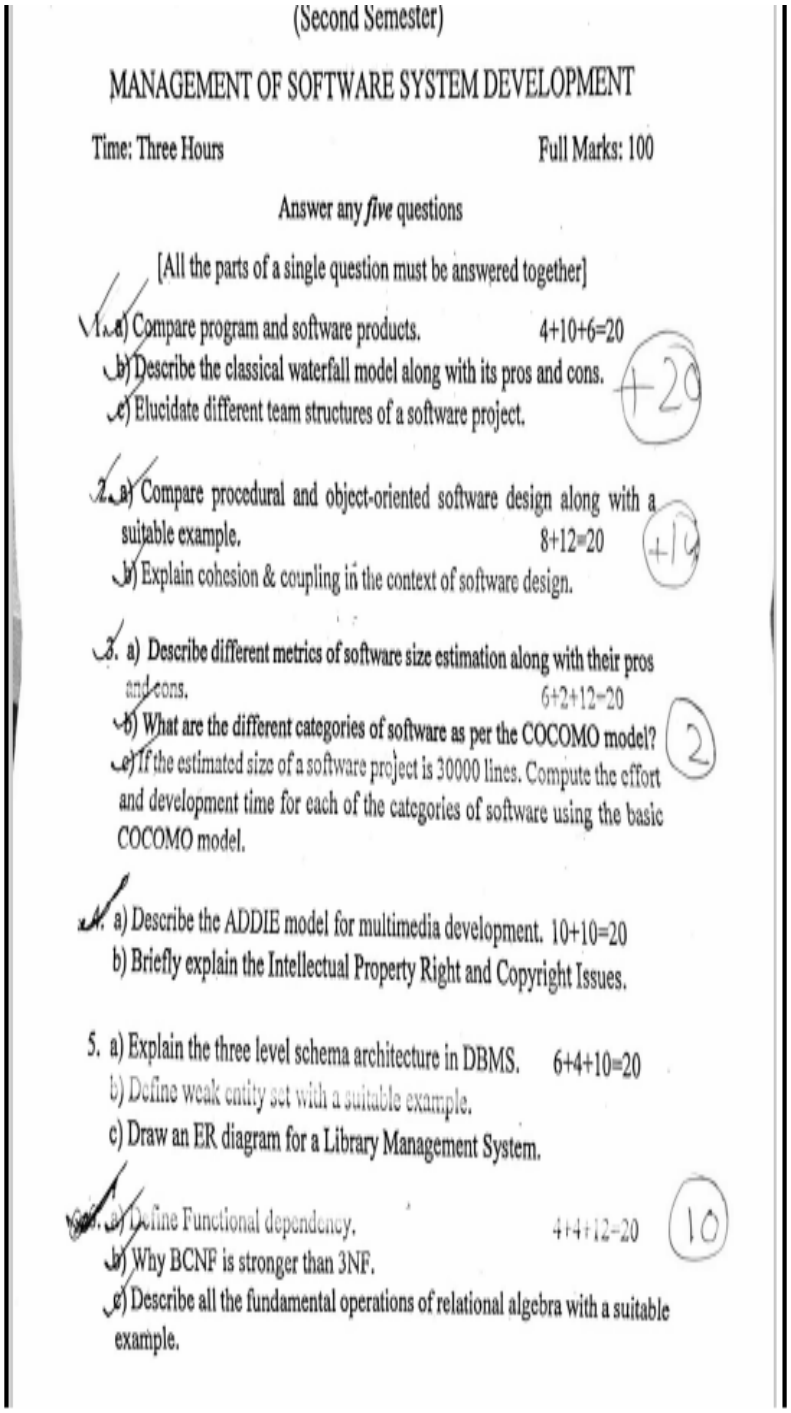
Figure 7:- Image lower part in Image Preprocessing Page

Here in this figure we are giving lower part of the image calendar

SJ fa) a eee 2023 yo CESC ciwires my 1234567 30 31 1 bas 910 1 e 13 4 A 2 3 4 5 6 7 8 S15 16 17 18 19 20 21 =) 9 10 1 12:13 4 15 Eloo 0 24 25 © 27 28 [my 16 17 18 19 20 21 22 M45 30 31 23 24 25 26 27 28 © 26 27 28 «1 2 3 4 12345 X56 7890 n/a 6 7 8 910 N12 12 3 4 6 6 7 8D 13:14. 0:16. 7 18. 19 Hl) 19 20 21 22 23 24 25 |) 20 21 22 23 24 25 26 27 28 29 30 31 123 47% 12 ms 62 8 90 ny 34 5 6 7 8 9 Fy 12 13 14 15 16 17 18 yo 1 12 13 4 15 16 By) 19 20 21 22 23 24 25 |W 17 18 19 20 21 22 23 26 27 28 29 30 31 ME 24 25 26 27 28 29 30 30 Fy ie 34567 el 2 3 e¢ 5 6 © 8h 8 9 1 z a io 112 2B e S| 9 10 1 12 13 © 02 15 6 7 8 19 20 © 6 7 18 19 20 21 ef e © © 25 26 27 © 23 24 25 26 27 28 29 29 30 31 © 23 40 69F 123 4 S78 eo nwa sey 5 6 7 8 90 N SJ 5 6 7 18 19 20, 12 13 4 e¢ 6 7 1B 21 22 23 24 25 26 27 ley 19 20 21 22 23 24 25 28 29 30 31 r41 26 © 28 29 30 12 3 3 12 S)4 5678 90/83 45678 9 = 112 8B 4 6 6 7 |= 10 N12 Bw 5 16 18 19 20 21 22 23 24 |=) 17 18 19 20 21 22 23 25 26 27 28 © 30 24 © 26 27 28 29 30 @ Red dots indicate Public Holidays Ba Swerrussnerincum Zena tency | uhm. ett sy coke nares stay latch 2th october Ssdoy Q ees oon esr — ‘rene eet DO Bete irermmernreioy Cmnstor ‘nected BEM cewinerersserocmnn ‘soronsarey | The folaing PU Hola lon Soncys < [ees Sane | eee Saleieeg oes Saye See eae ag aiemieee fey cece eee. | eee tren zon nyse aot ot 2nd oc Sy inp Soamattecty — | tate peeanarend ecu rect aoceeriedy | Gow ‘pnnrete Soty In case of power interruption and billing issues, please call 033 3501 1912 / 033 4403 1912 / 1860 500 1912 / 1912 or WhatsApp us at 74390 01912 ® You may also lodge a complaint for power interruption by sending SMS to 56070. Write CESC or you can lodge your complaint at our website www.cesc.co.in ------- Save paper - Do not print this unless necessary ---

Figure 8:- Output of Figure 6

Another example – We are using a sample question paper to perform text extraction



(Second Semester)

MANAGEMENT OF SOFTWARE SYSTEM DEVELOPMENT

Time: Three Hours Full Marks: 100

Answer any *five* questions

[All the parts of a single question must be answered together]

1. a) Compare program and software products. 4+10+6=20
   b) Describe the classical waterfall model along with its pros and cons.
   c) Elucidate different team structures of a software project.

2. a) Compare procedural and object-oriented software design along with a suitable example. 8+12=20
   b) Explain cohesion & coupling in the context of software design.

3. a) Describe different metrics of software size estimation along with their pros and cons. 6+2+12=20
   b) What are the different categories of software as per the COCOMO model?
   c) If the estimated size of a software project is 30000 lines. Compute the effort and development time for each of the categories of software using the basic COCOMO model.

4. a) Describe the ADDIE model for multimedia development. 10+10=20
   b) Briefly explain the Intellectual Property Right and Copyright Issues.

5. a) Explain the three level schema architecture in DBMS. 6+4+10=20
   b) Define weak entity set with a suitable example.
   c) Draw an ER diagram for a Library Management System.

6. a) Define Functional dependency. 4+4+12=20
   b) Why BCNF is stronger than 3NF.
   c) Describe all the fundamental operations of relational algebra with a suitable example.

Figure 9:- Sample Question Paper(Scanned)

Here in this figure we are uploading a sample question paper for text extraction in LKE

Master in Multimedia Vevelopmicns oe and M.Tech. IT (Courseware Engineering) Examination, 2022 (Second Semester) MANAGEMENT OF SOFTWARE SYSTEM DEVELOPMENT Time: Three Hours ull Marks: 100, Answer any five questions [All the parts ofa single question must be answered together} \1CafGompare program and sotvare produ. an0s620 be he ls waterfl model long wth its pros and con. /{ usiat different team structures ofa sftware project re ALsy Compare procedural and objectorionted software design alone with a _Sipileceamoe BH2=20 C.) Explain cohesion & coupling iti the context of software design. SY (<2 2) Describe diferent mets of software sz eatmation along with thei pros apcore genera) te cages of spt a pec the COCOMO mode? (>) Fitecatinatd sins of sofa peers SO00D ines Compose ret and development time foreach of the eatgores of software wing te baste EOCOMO model. 4106) Descite the ADDIE model for multimedia development. 10+10-20 ©) Briely explain the Intellectual Property Right and Copytight Issues, 5. 8) Explain the three level schema architecture in DBMS, 644+10-20 6) Draw an ER diagram fora Library Management Systm, AHO. Psion dpendeny srviz20 (10 'WySvhy BONE is stronger than SNE " [ydevrve al he fundamental operans of elon algebra witha stable example. [Tam over

Figure 10:- Extracted text from the sample question paper as shown in Figure 8.

Another example – The next figure is about the list of number of students appeared in English with MA Honours. This is a scanned document.

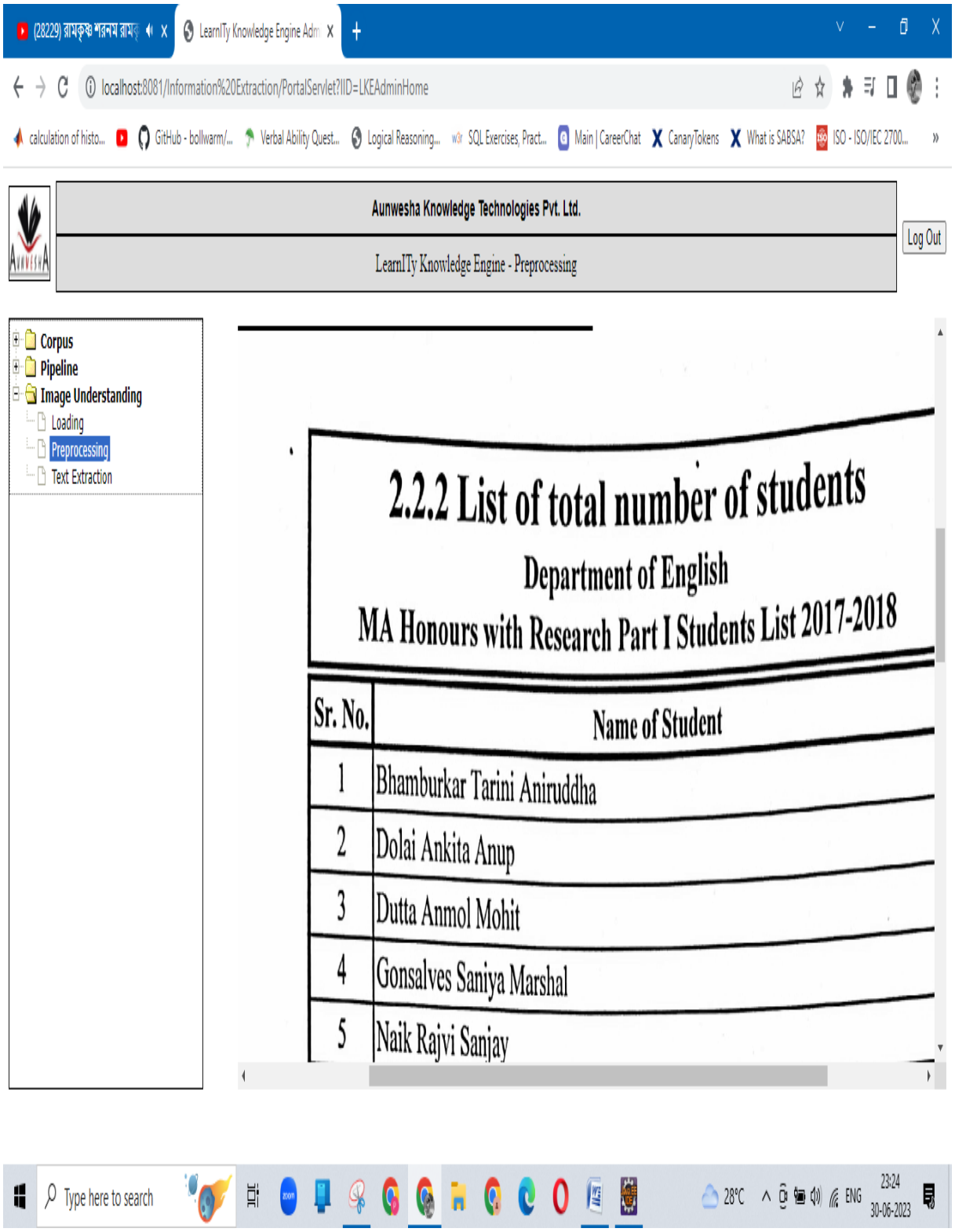


Figure 11:- Student Sample

Here in this figure we are uploading Student Sample for experimenting text extraction in LKE

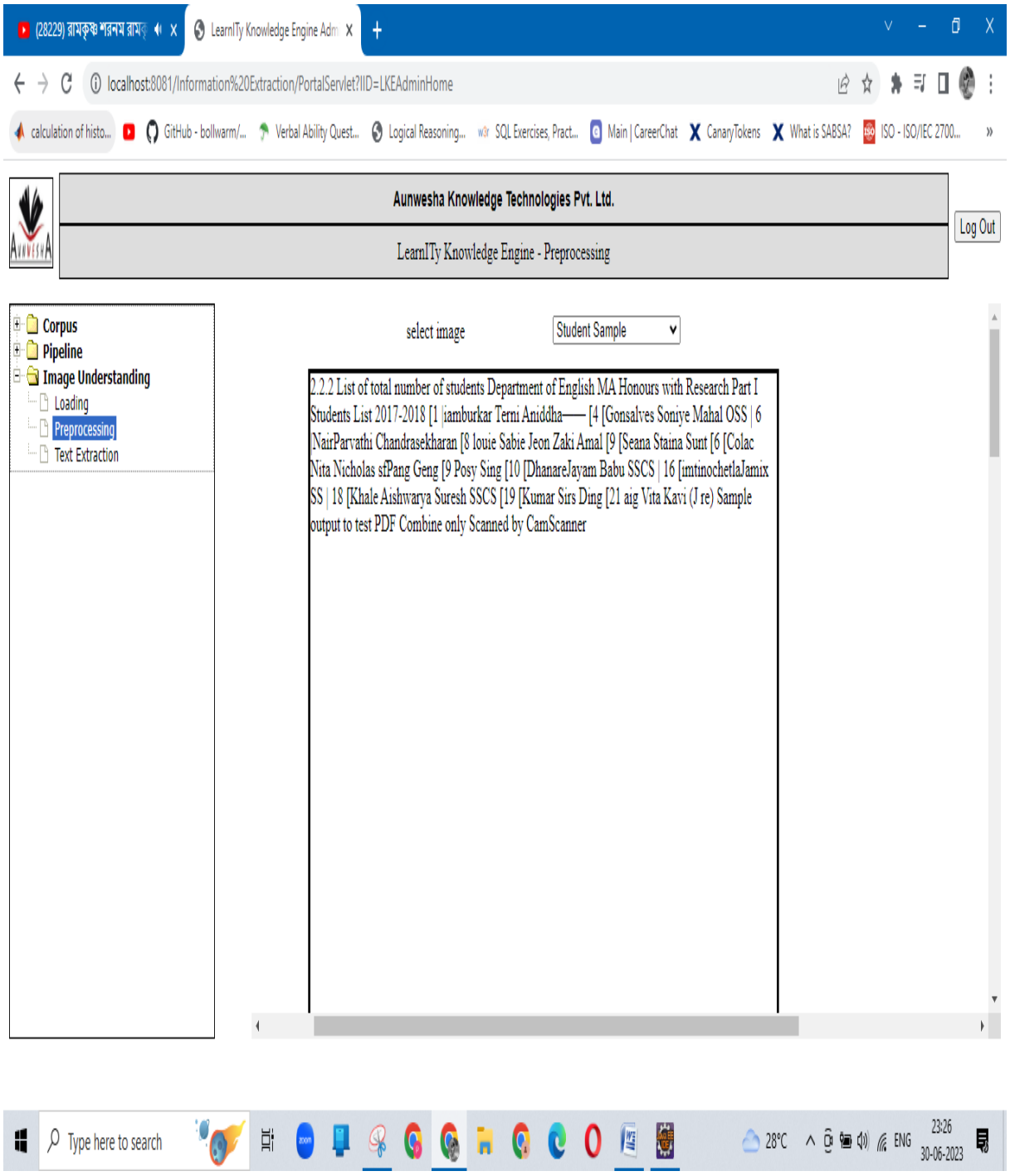


Figure 12:- Output of Figure 11

Here in this figure we are getting the output of student sample as uploaded in the previous figure

2.Postman

We get the below figure by giving the GET request to the GET request option and we are sending the request to this software

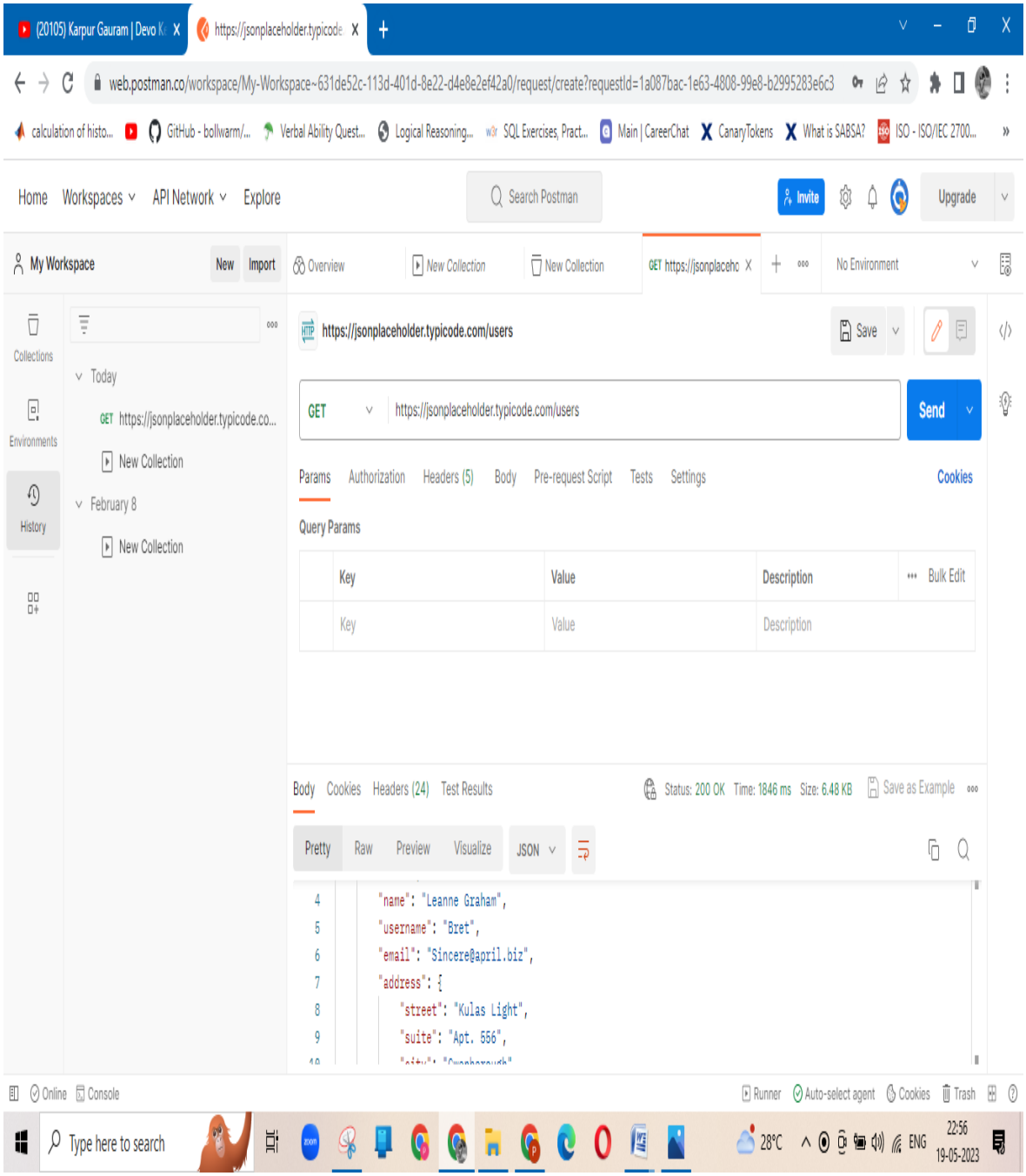


Figure 13:- Checking of OK message

.

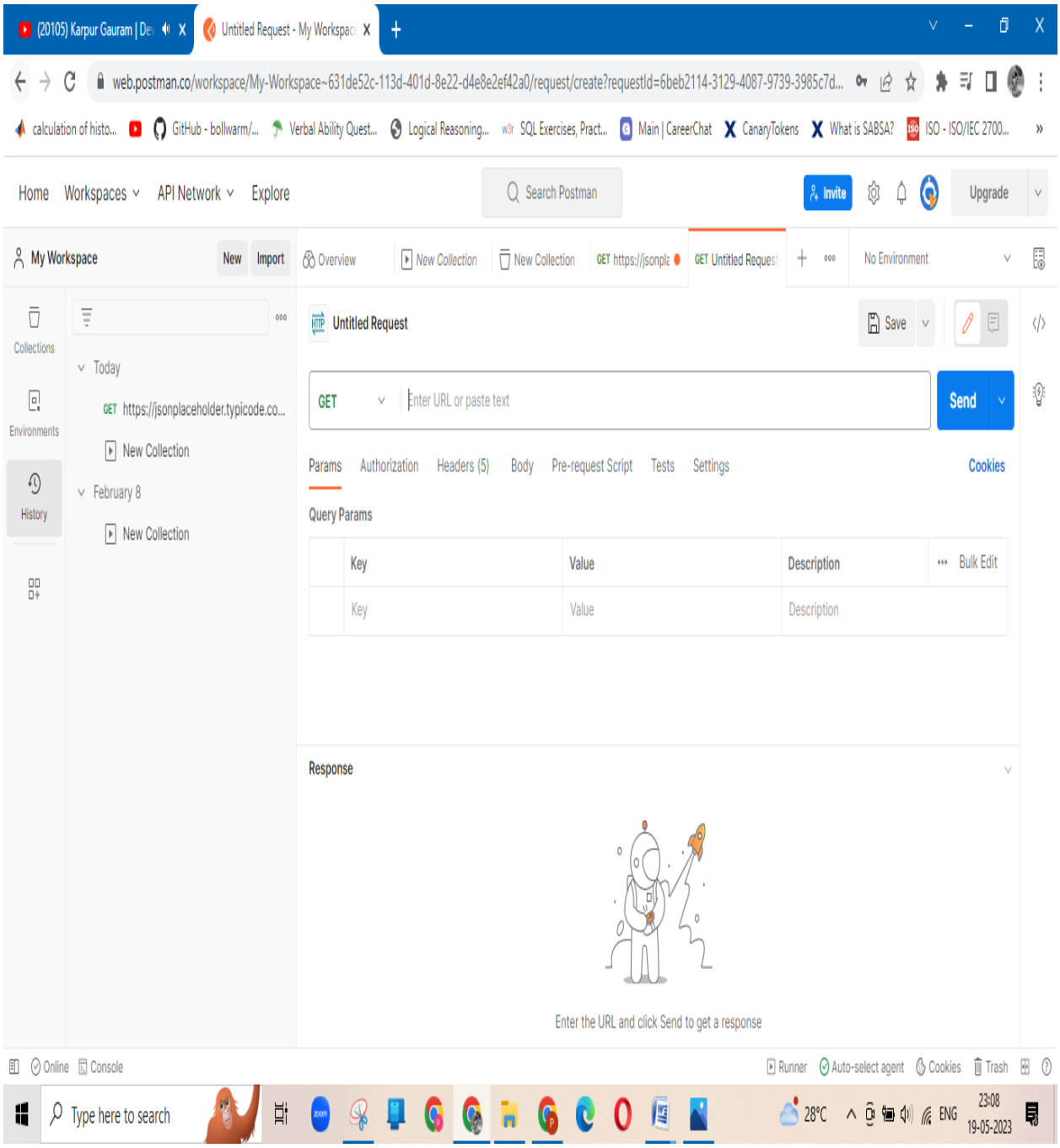


Figure 14:- Clicking of new tab to get request

Here in this figure we are clicking the new tab to get the request for further processes.

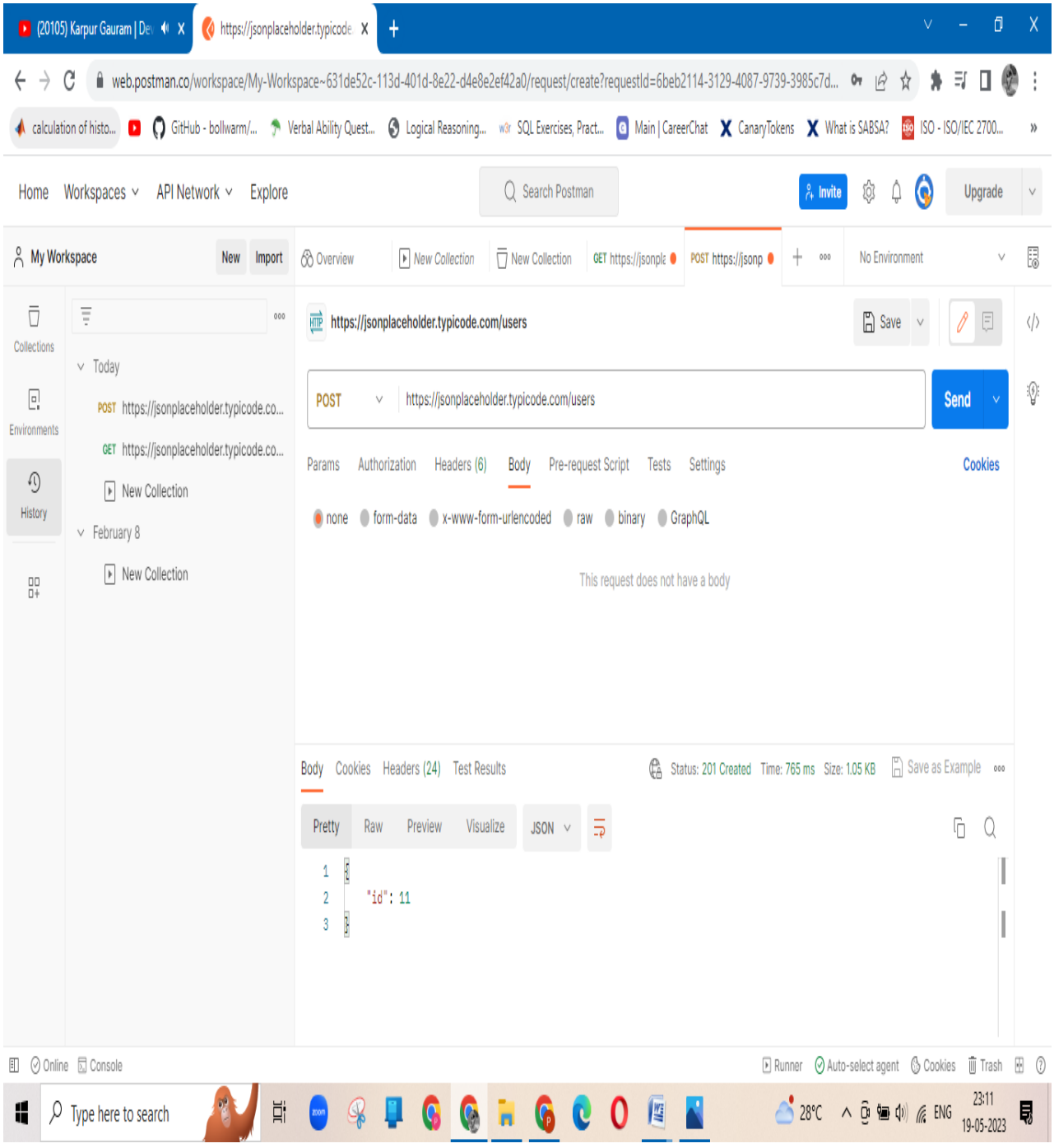


Figure 15:- HTTP Request to POST

Here in this figure we are giving HTTP Request To Post icon where we are giving our id value. Same HTTP request is given to Post icon.

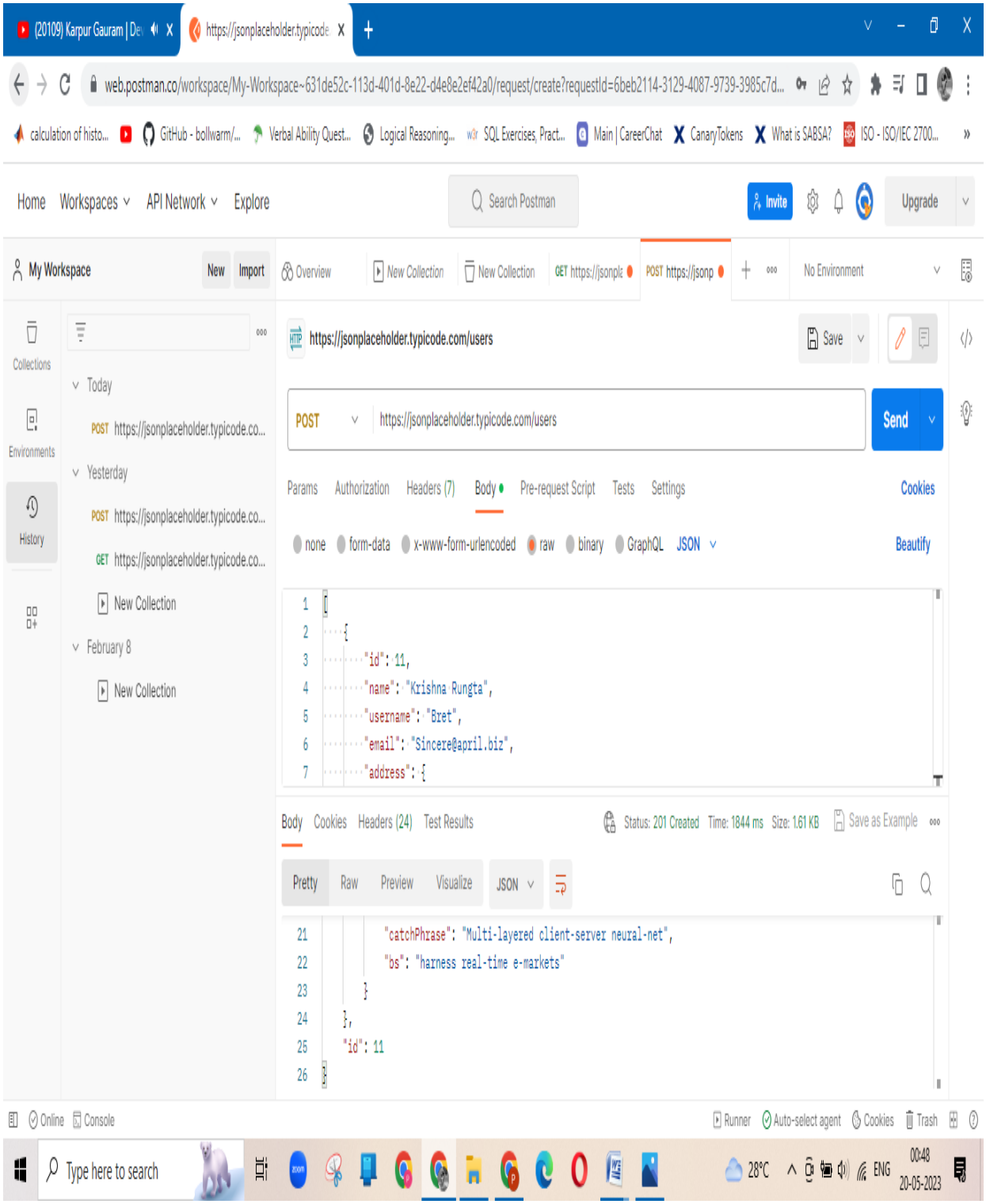


Figure 16:- Sending and checking status

Here in this figure Body Icon is selected and we are sending the post request to the post command and checking the status of the Post request.

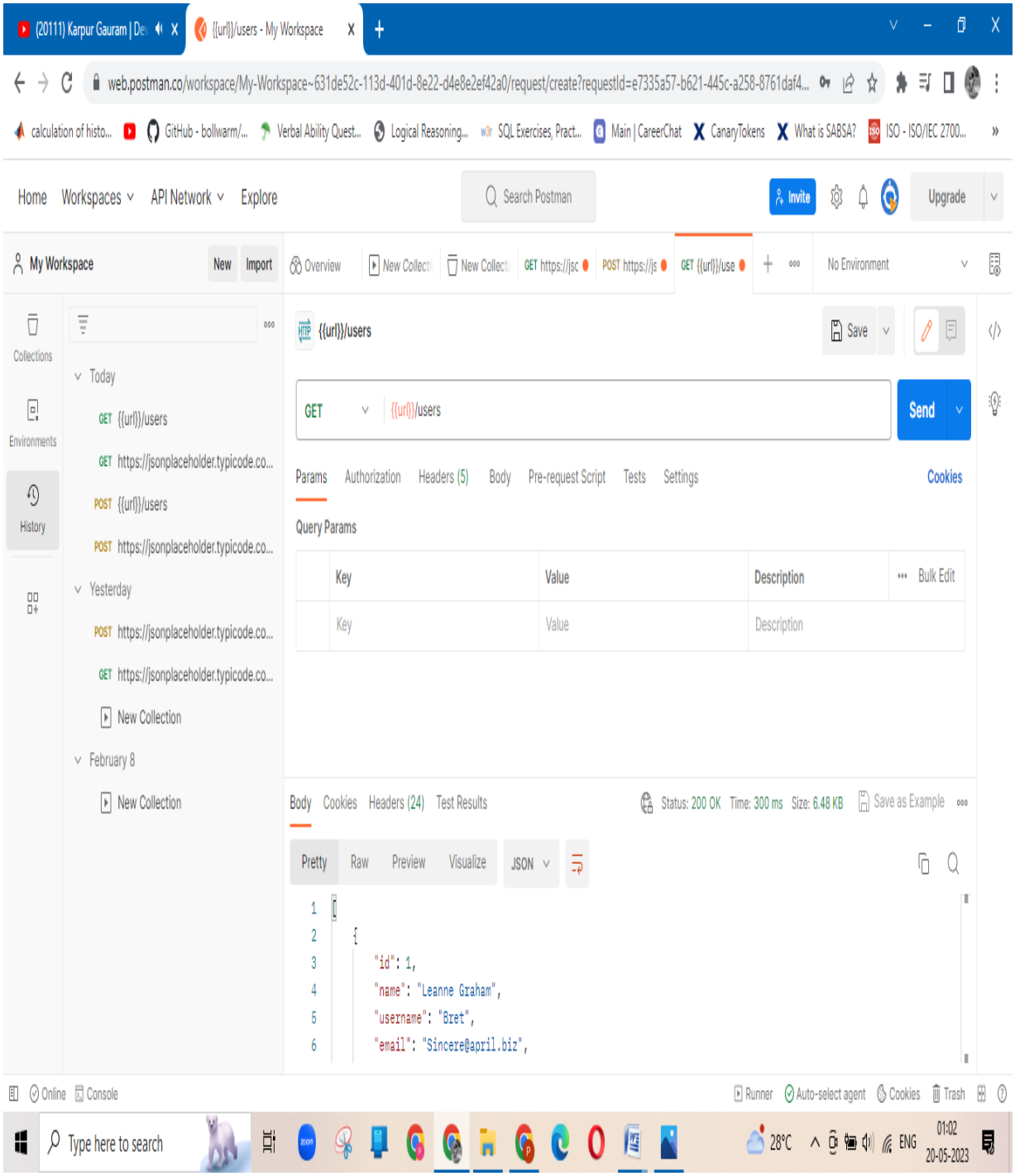


Figure 17:- Parametric Request

Here in this figure again we are using Get Request by giving this [{url}]/users to implement parametric request.

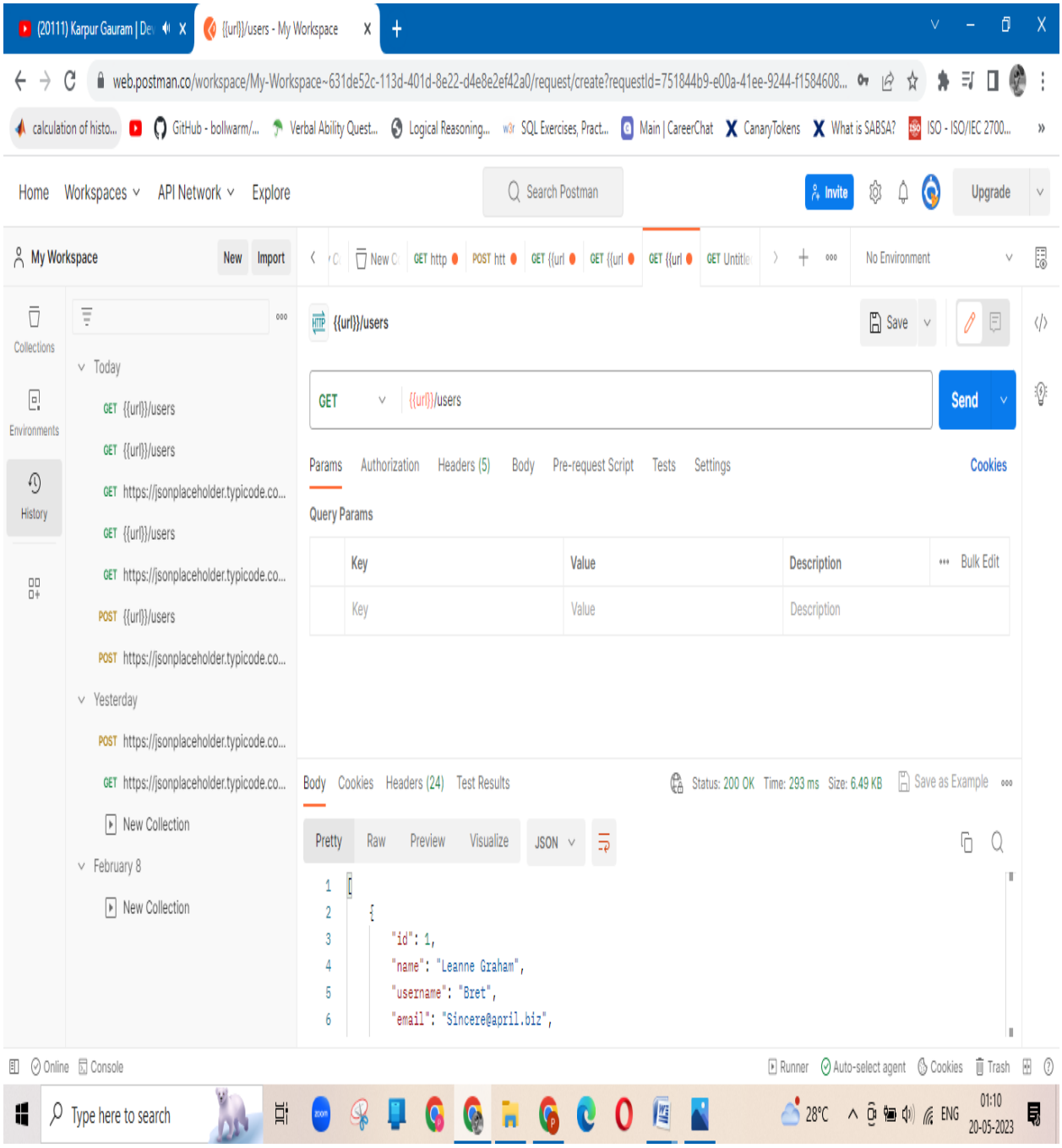


Figure 18:- Getting of Parameters

Here in this figure we are sending the Get request and we are writing the code in the body section where we are getting key value.

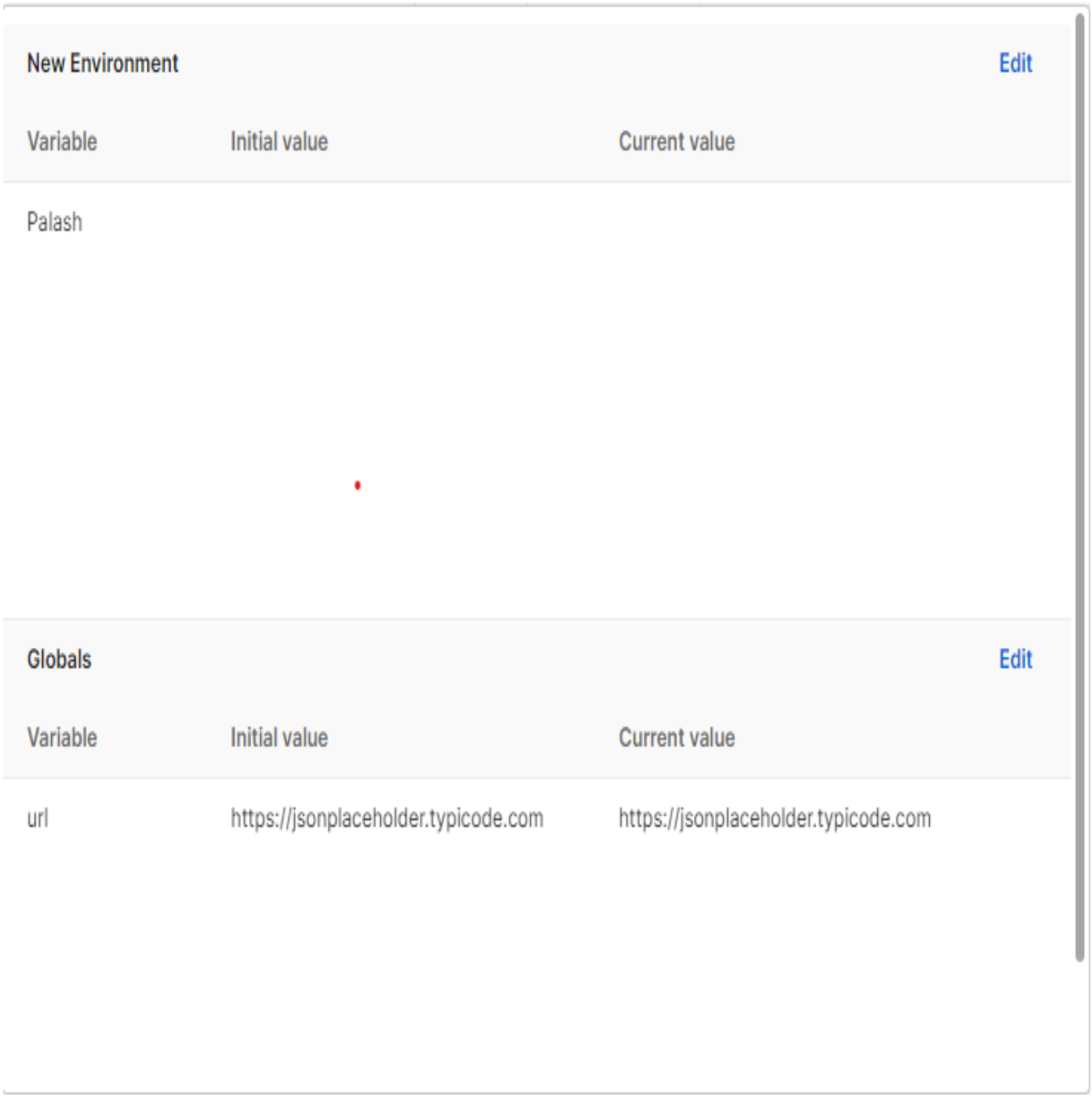


Figure 19:- Variable name is created

Here in this figure variable name is created with the name Palash and we are using Global variable where we are giving initial and current value using the following url – https://jsonplaceholder.typicode.com.

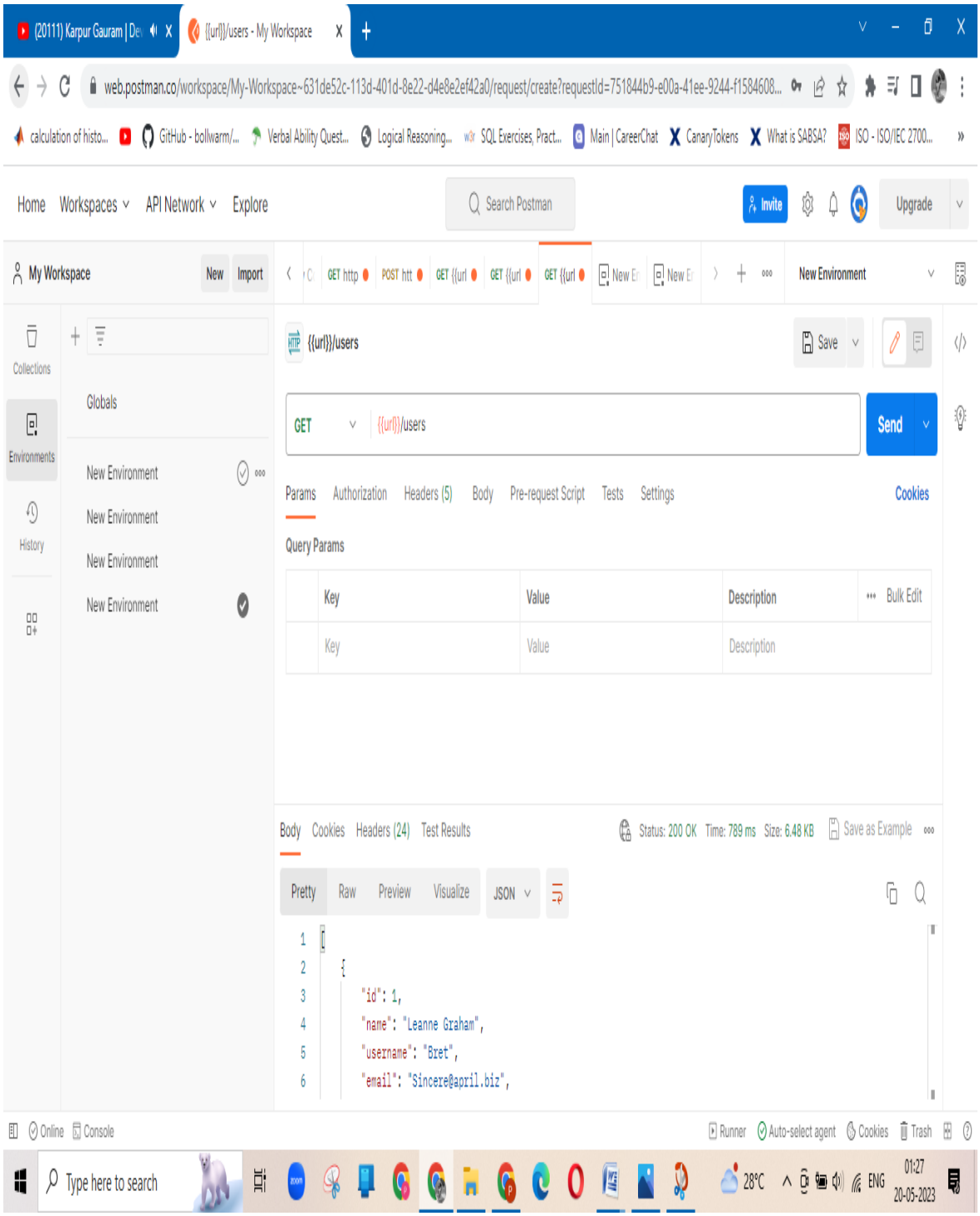


Figure 20:- Result of GET Request

Here in this figure we get the result of previous GET Request –[{url}]/users command in Get Request Function.

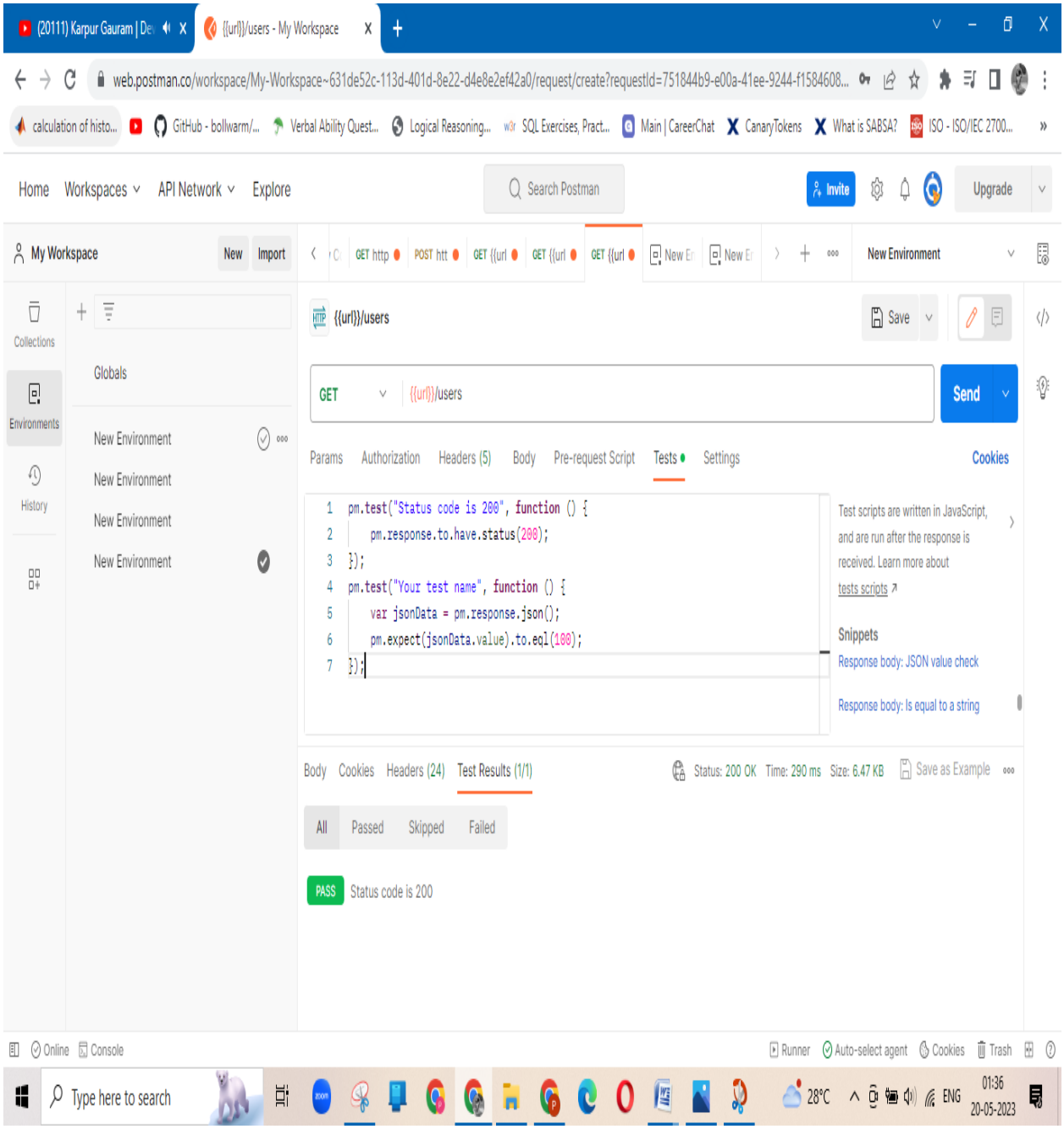


Figure 21:- Getting of Test Result

Here in this figure we are getting the result of Test function when we are writing the code in the Get Request Tab.

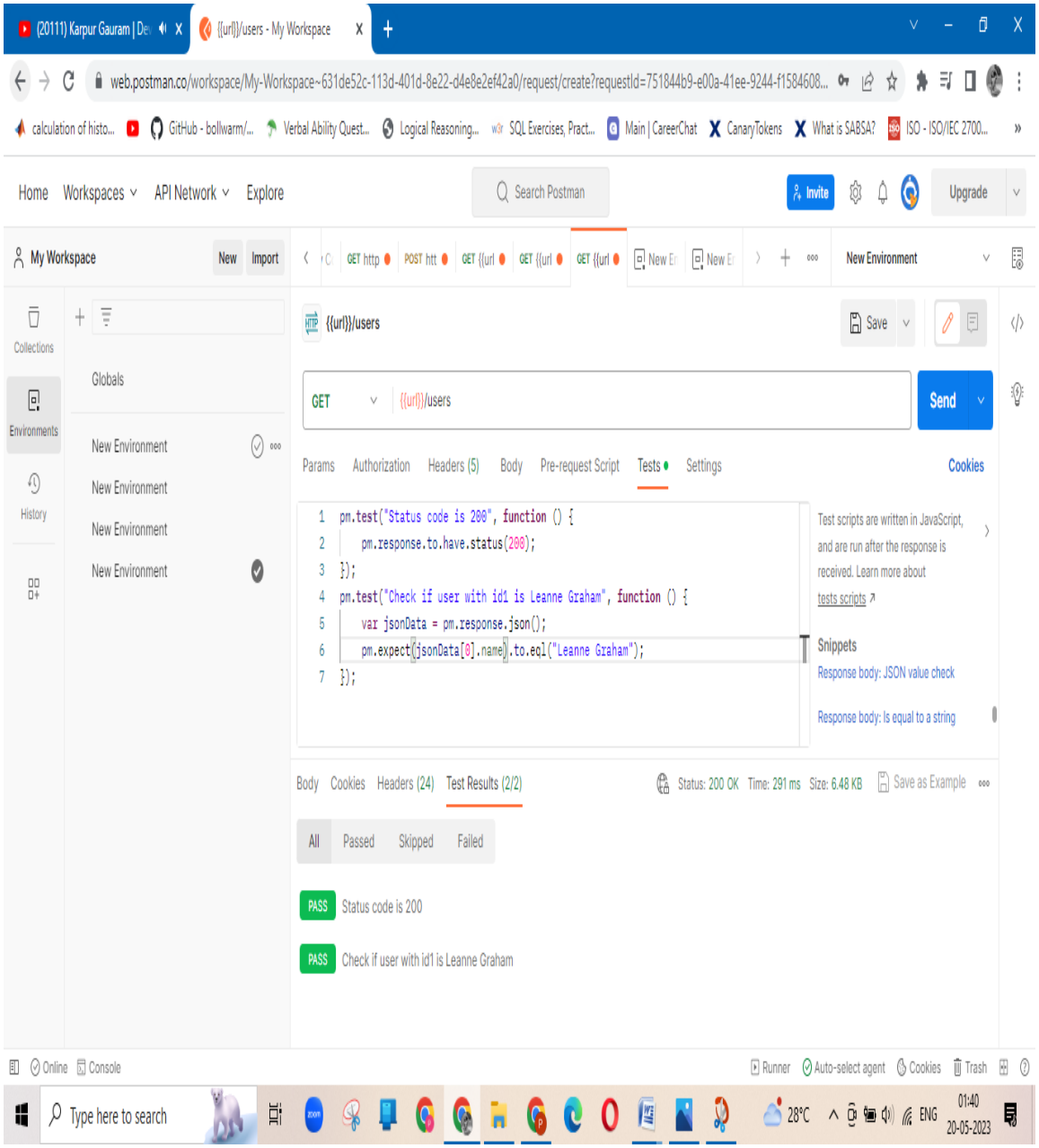


Figure 22:- Result is OK

Here in this figure we are getting the result after applying Test option.

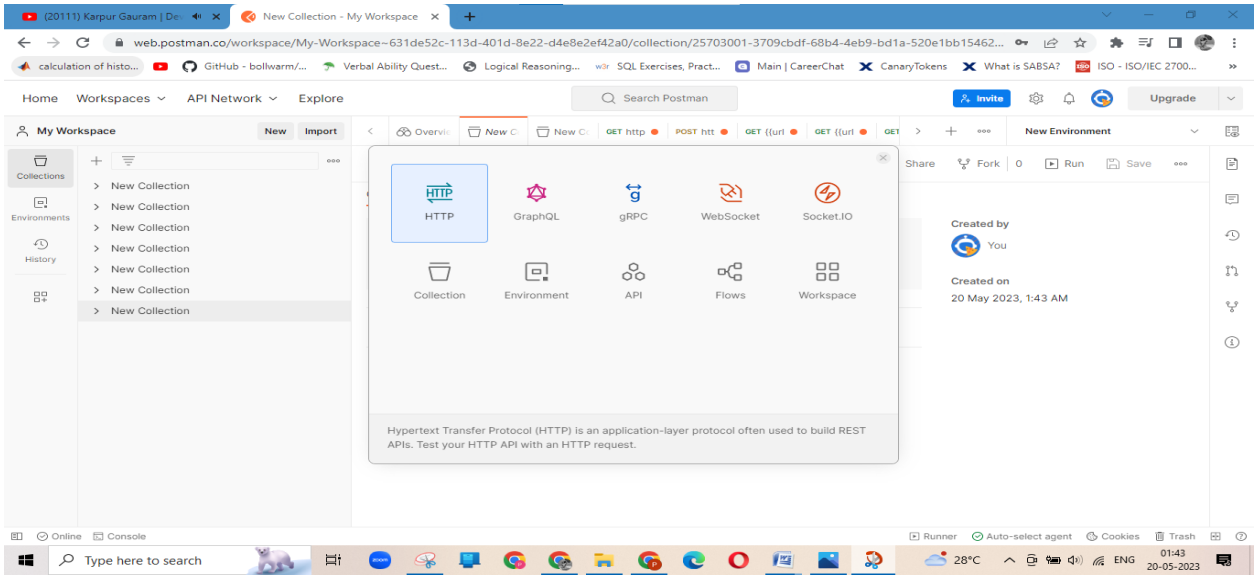


Figure 23: – New button is clicked

Here in this figure we are clicking the New button of http image.

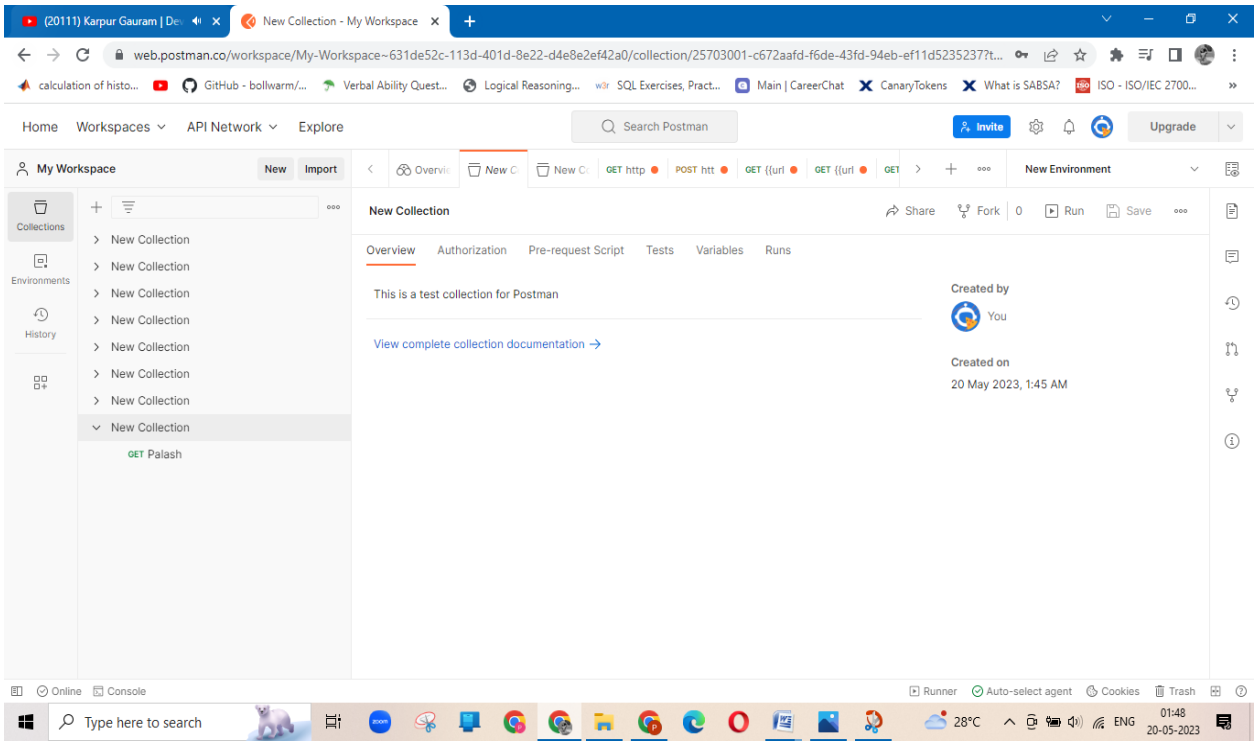


Figure 24:– Get Request is again given

Here in this figure get request is again given to perform the following actions.

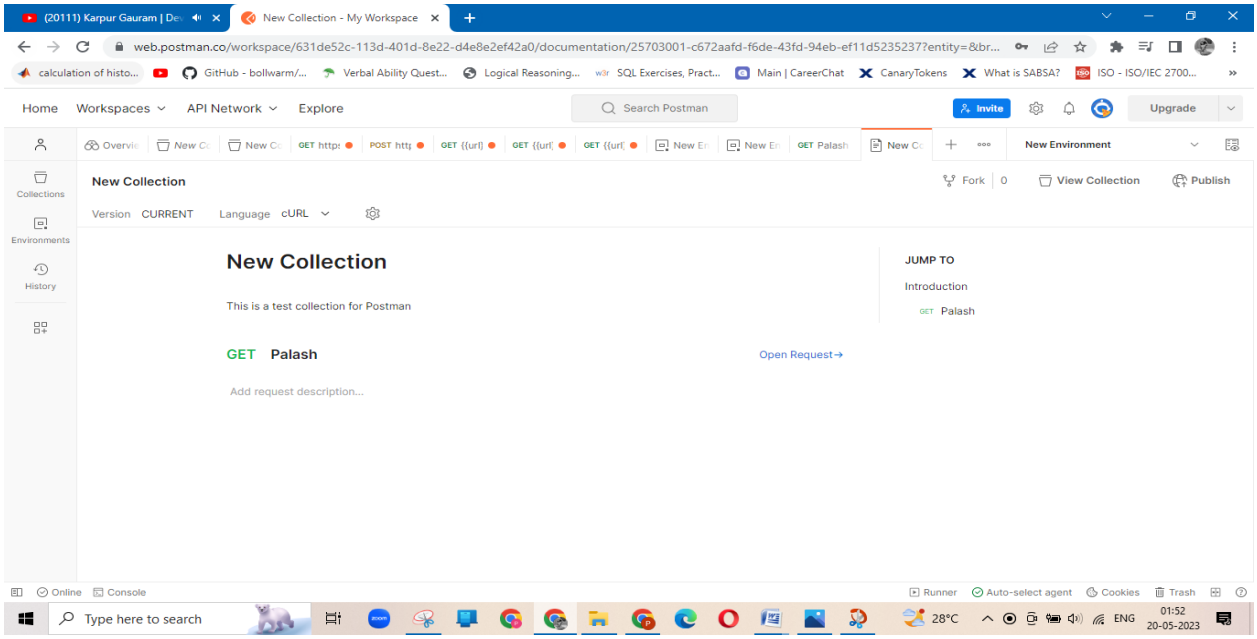


Figure 25:– New Collection is created

Here in this figure New Collection is created and added in the stored collections which is already given.

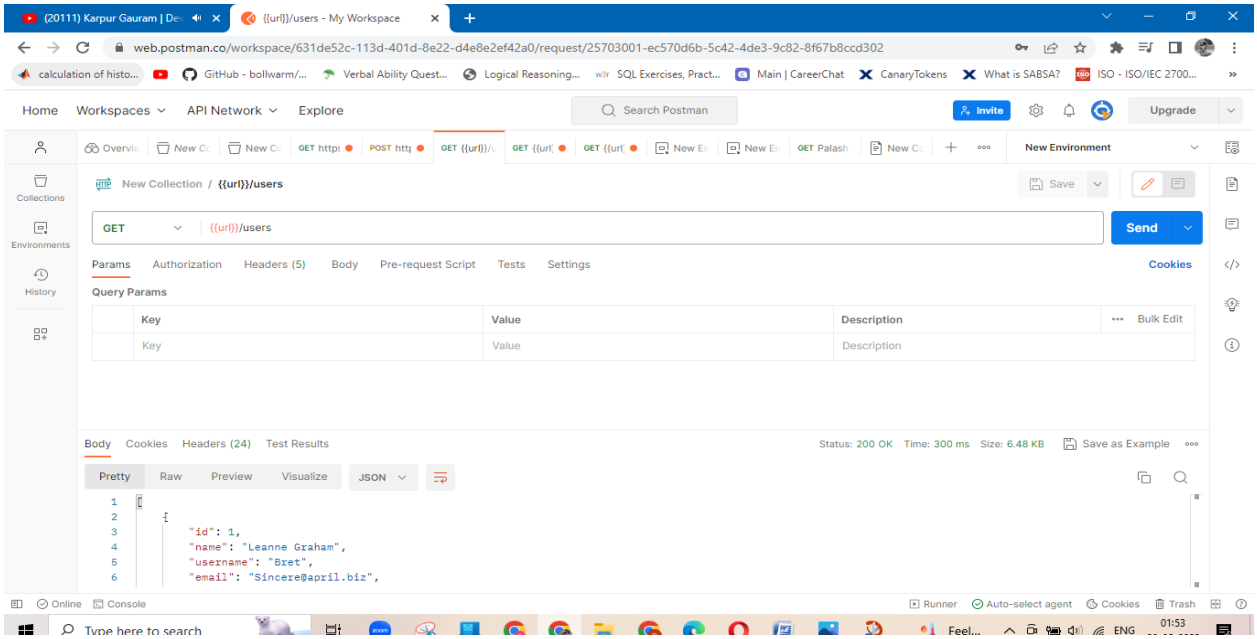


Figure 26:– Get Request is again given

Here in this figure we are again Get Request is given and the body tab is selected to write the code.

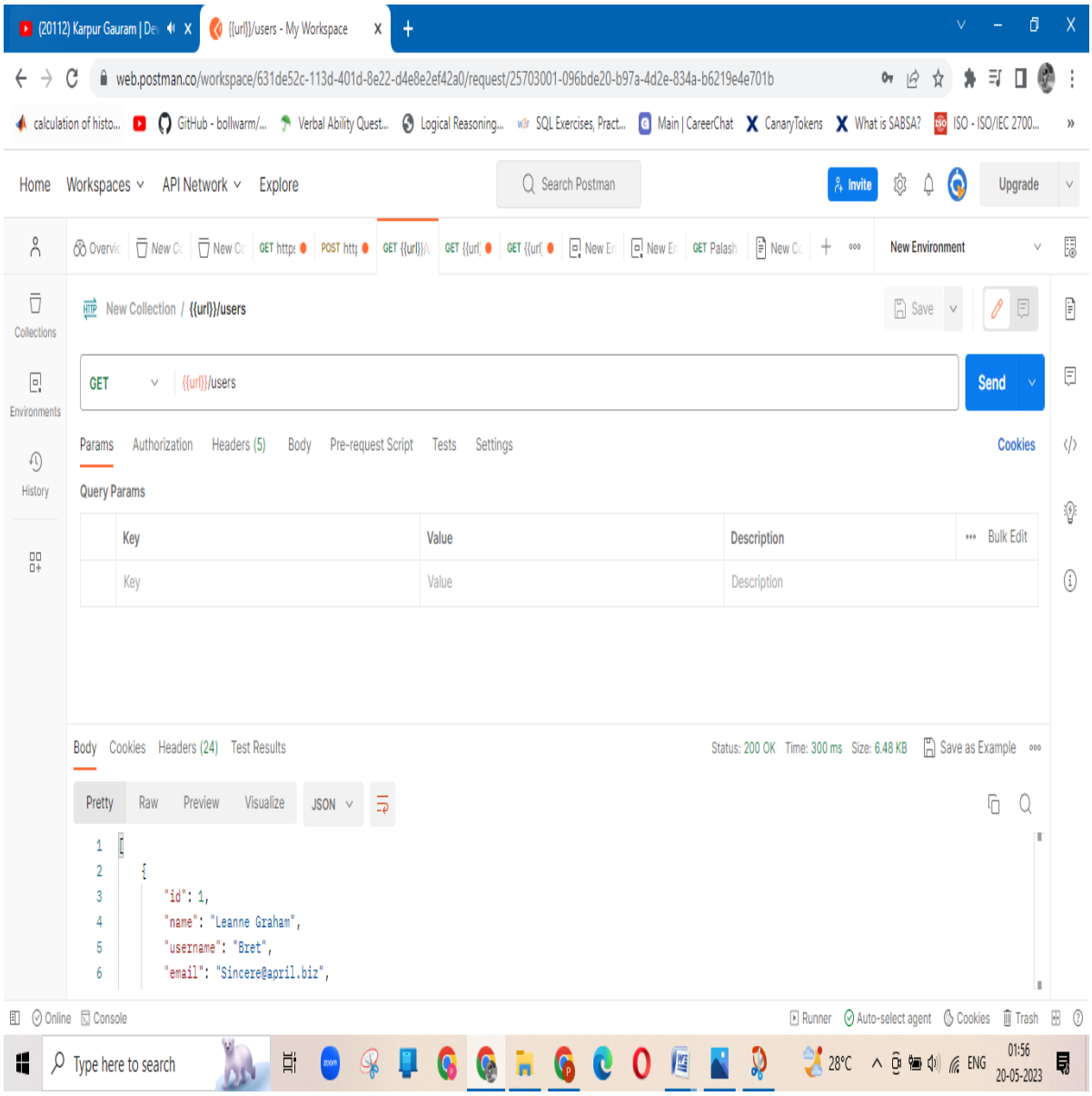


Figure 27:– Postman Test Collection is created

Here in this figure we are again performing Postman operation to get the Test Collection where previous tests we have performed are there.

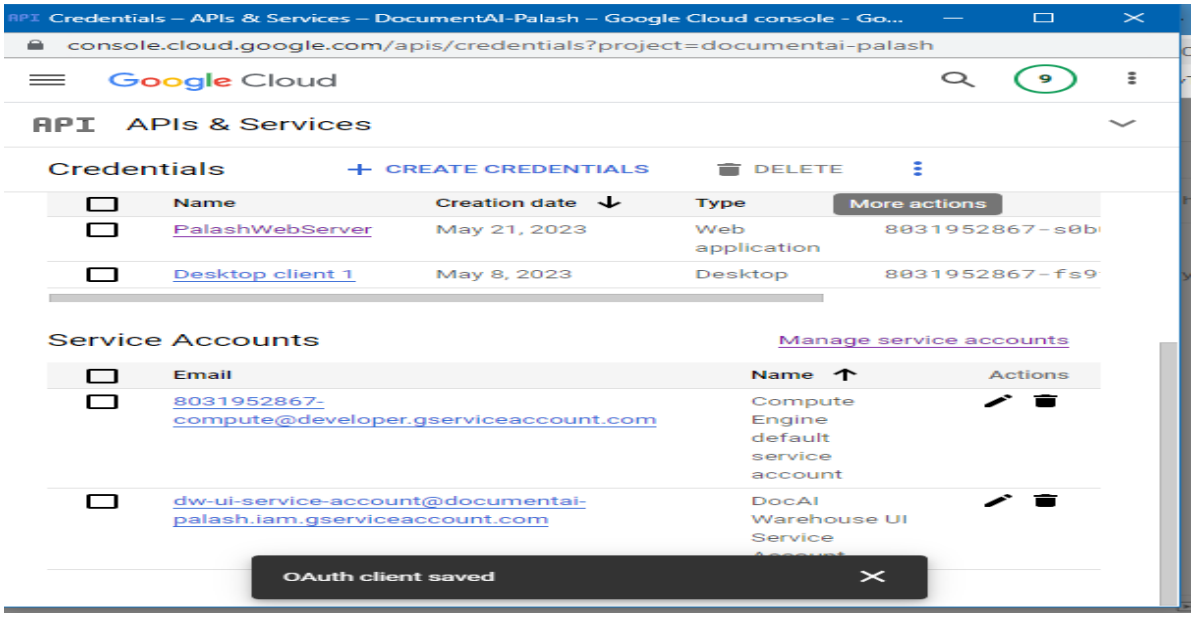


Figure 28: – API created

Here in this figure we are creating API(Application Programming Interface) in Google Cloud Console to extract text.

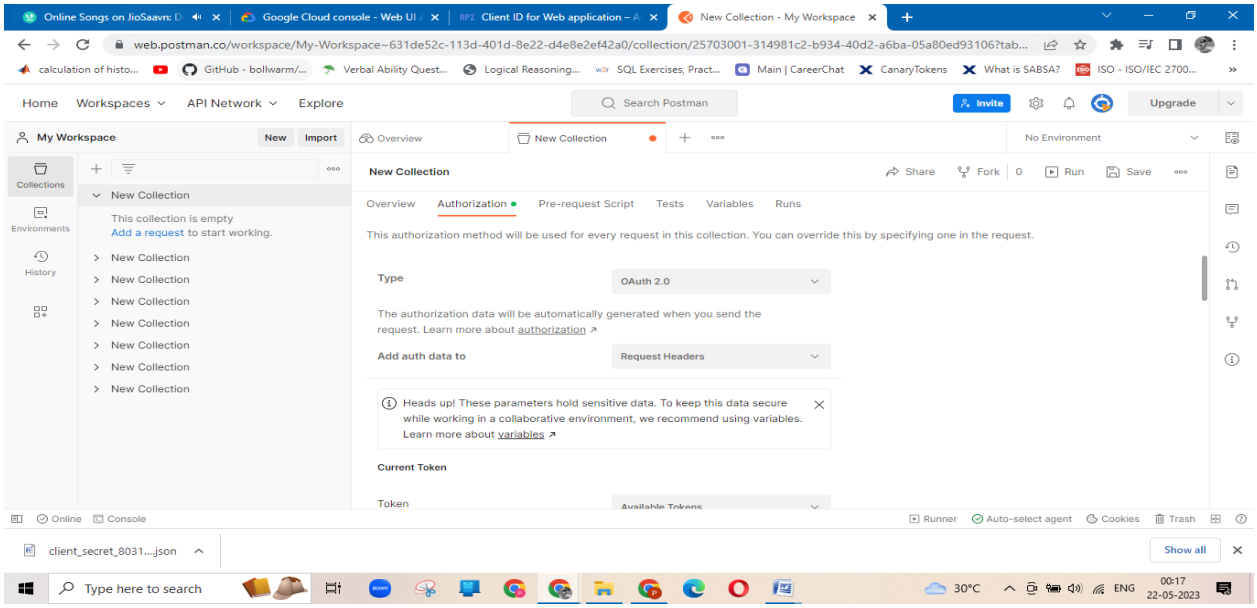


Figure 29: – Authetication option

Here in this figure we are clicking authentication tab to authenticate the text as shown in the figure.

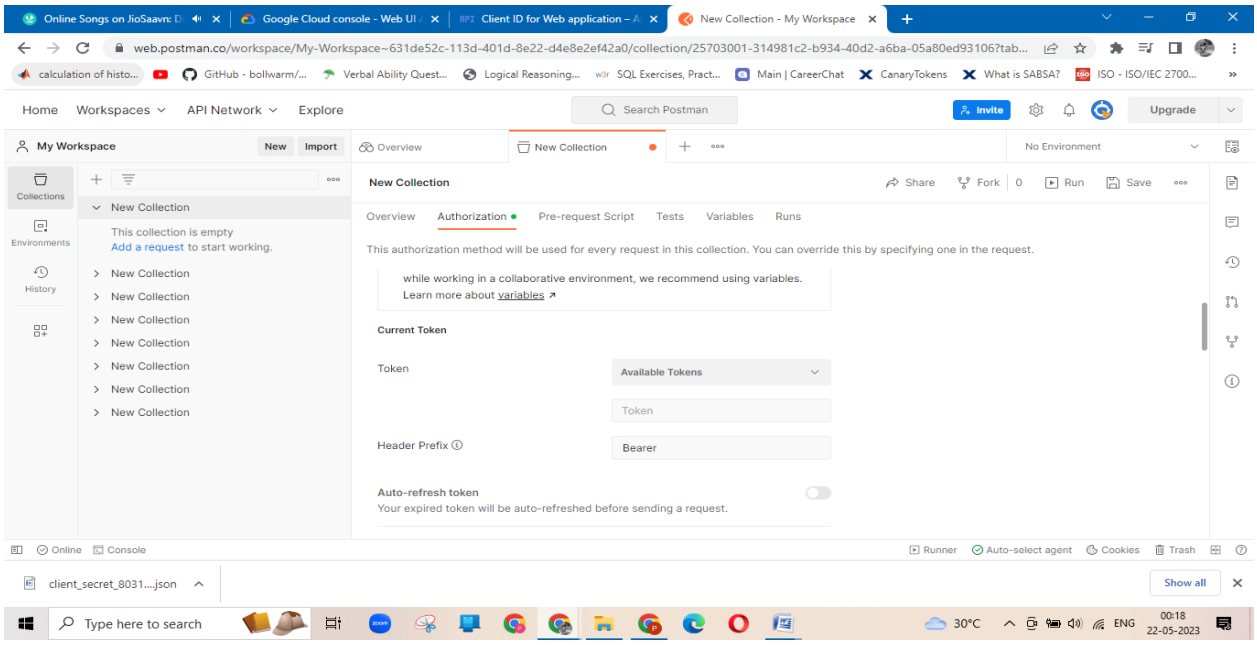


Figure 30: – Token name generated

Here in this figure Token name is generated from authentication option.

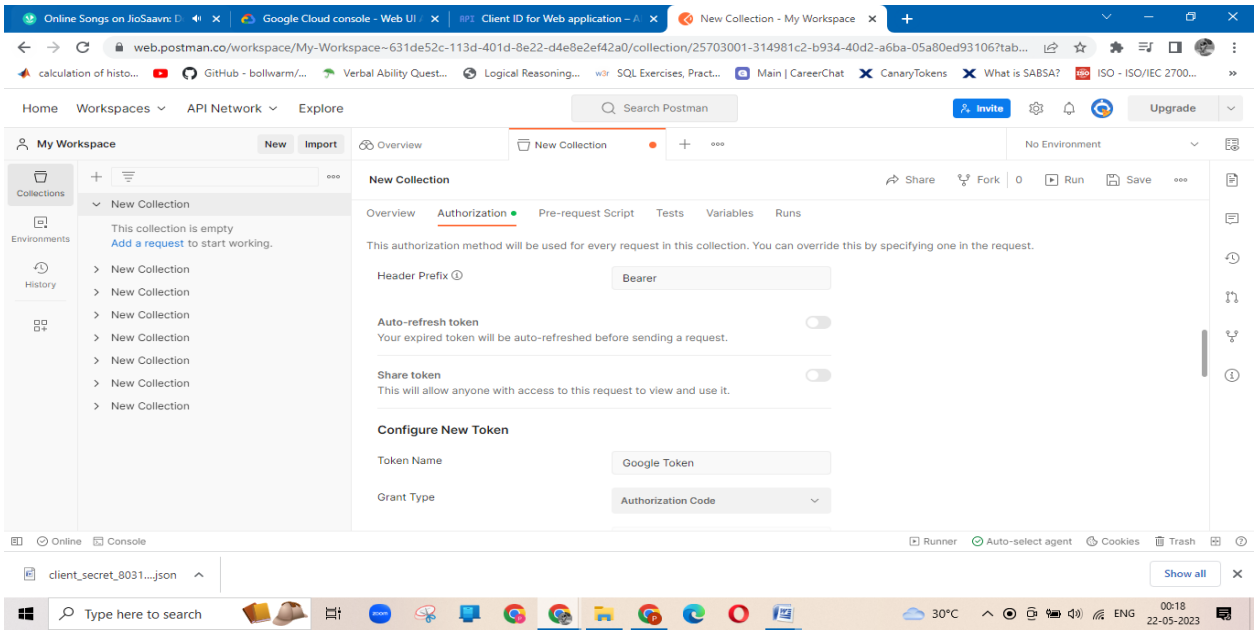


Figure 31:– Other options generated

Here in this figure other options are checked as shown in the figure.

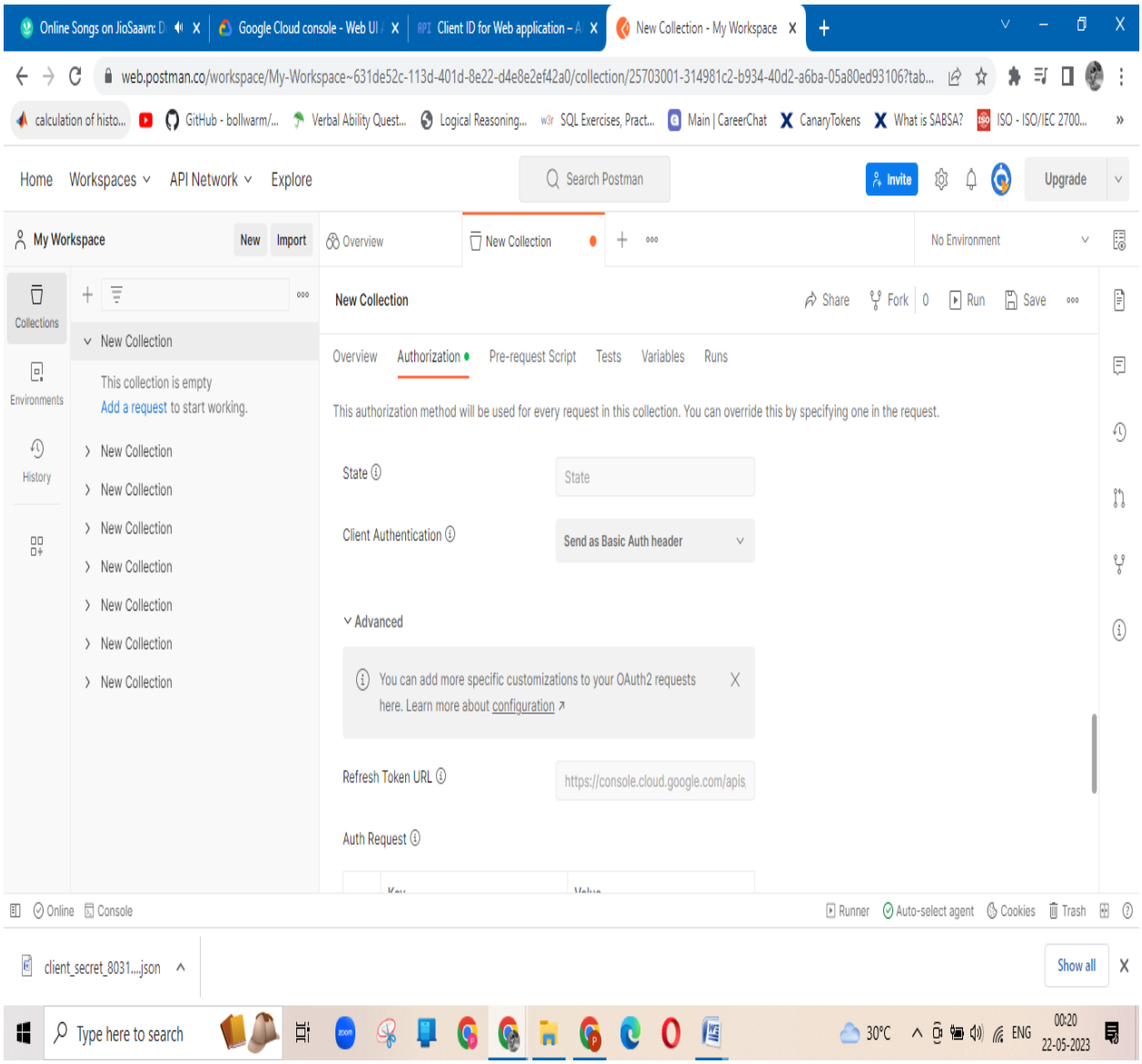


Figure 32: – Authentication Request is generated

Here in this figure we are authenticate the request to show the previous steps.

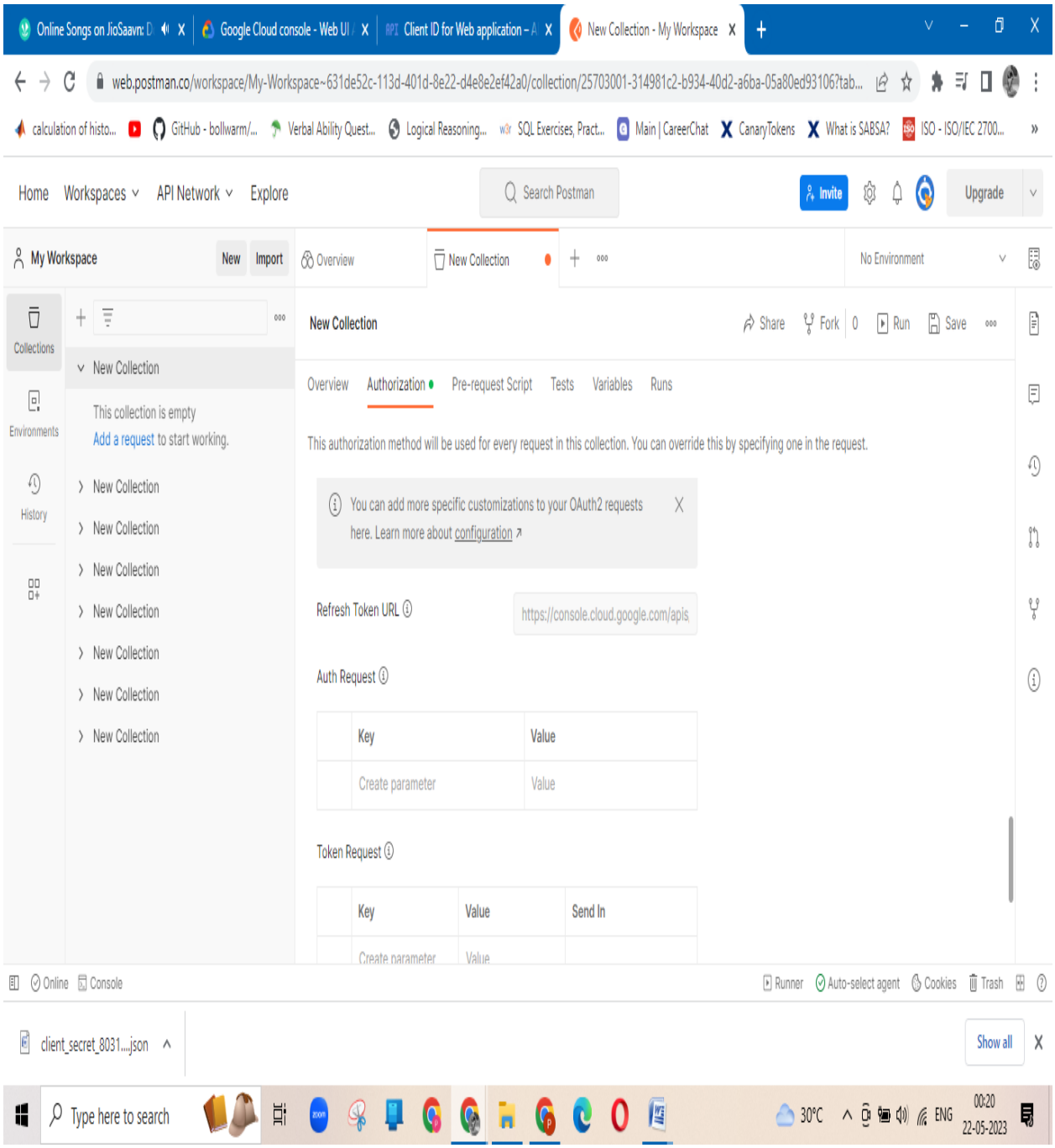


Figure 33:– Key is generated

Here in this figure we are generating the key to determine the value

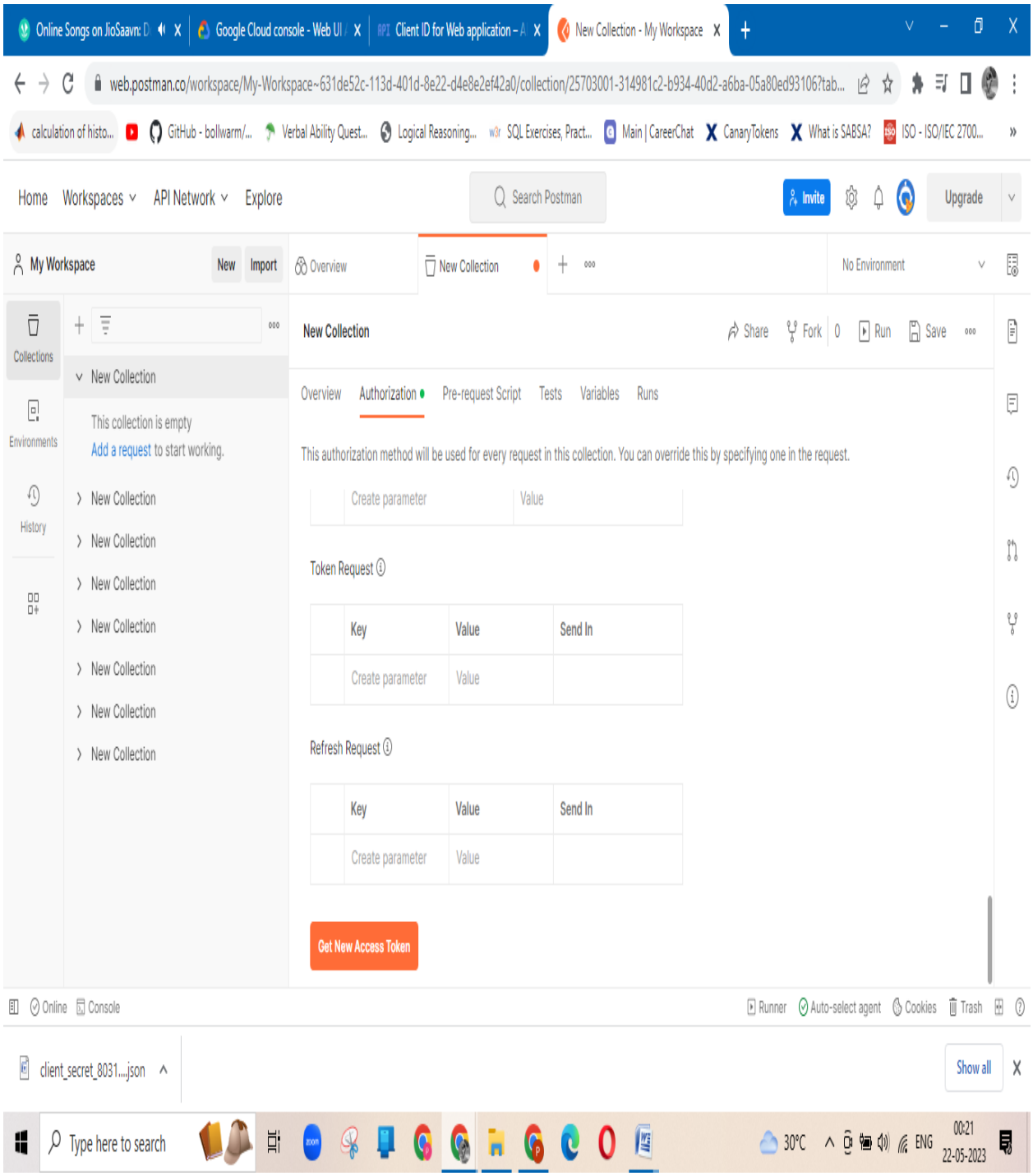


Figure 34: – All the parameters shown

Here in this figure only the parameters are shown all the parameters that are used in Postman.

4.3.Google Document AI

- ✓ First we have to open Google Cloud Console
- ✓ Then we have to register ourselves.
- ✓ Then after registration we have to open API & Services
- ✓ After that enable API & Services.

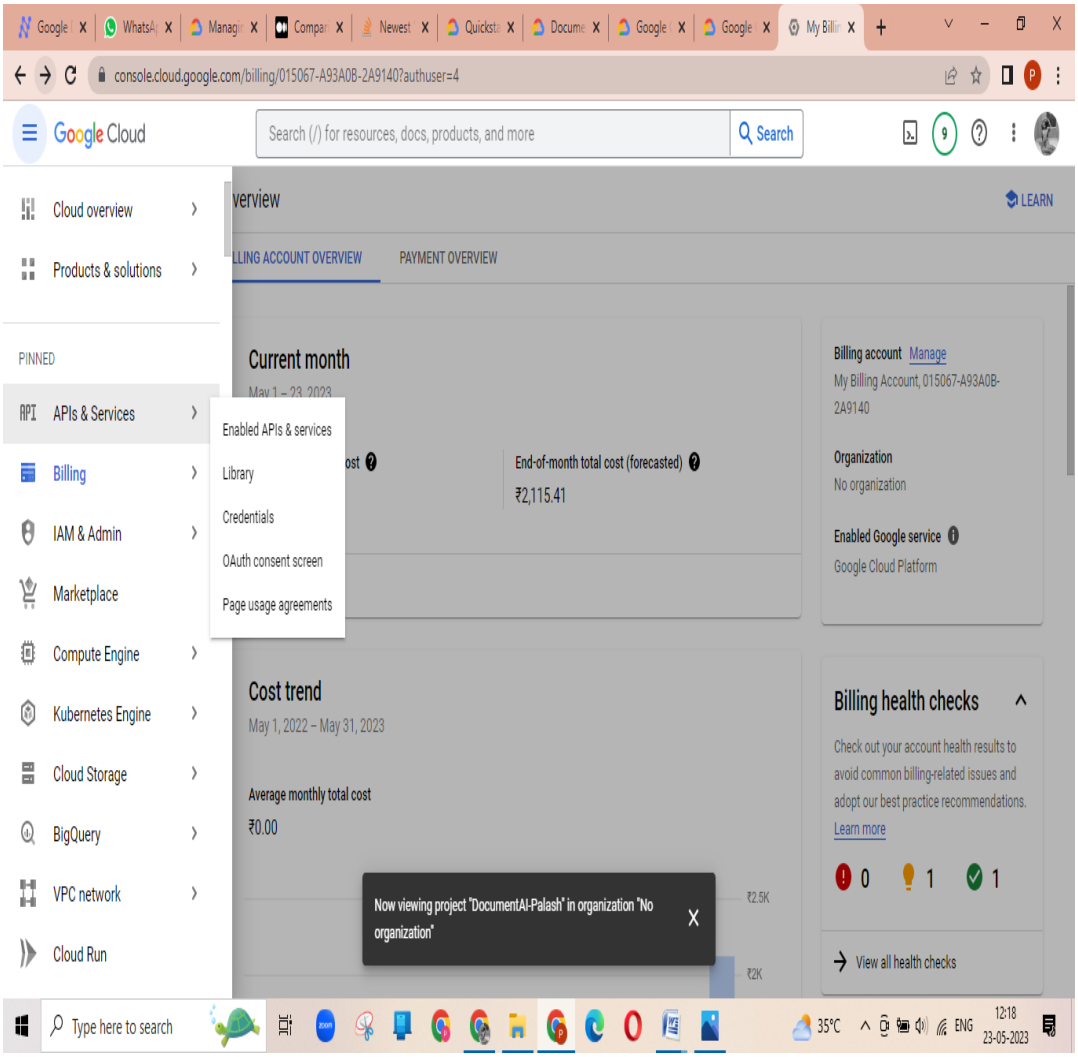


Figure 35: – Creation of Project and Project ID

Here in this figure we are creating the project and project ID from API&Services.

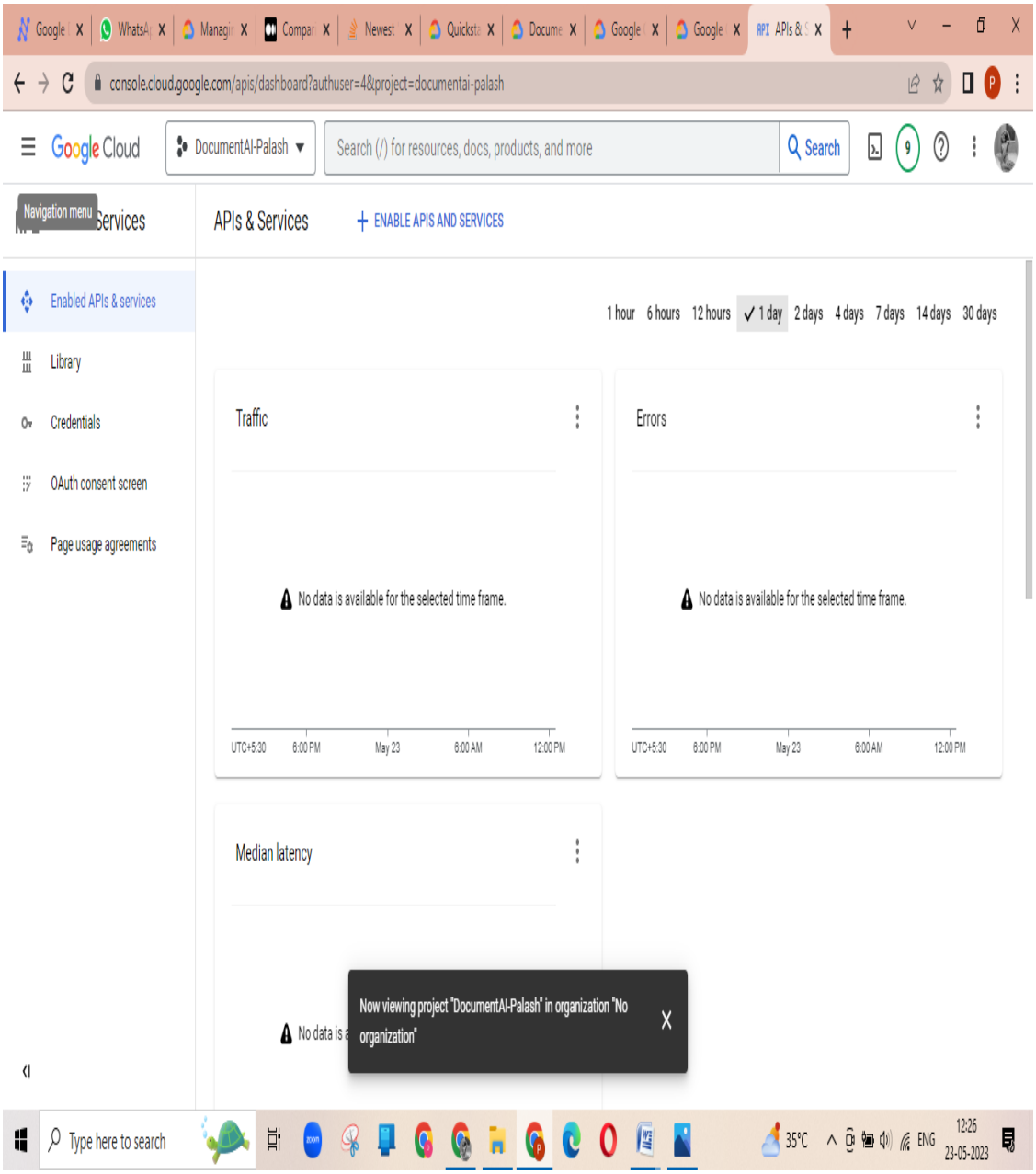


Figure 36 :– Enabling of API

Here in this figure we are enabling API option from Enabling API & Services

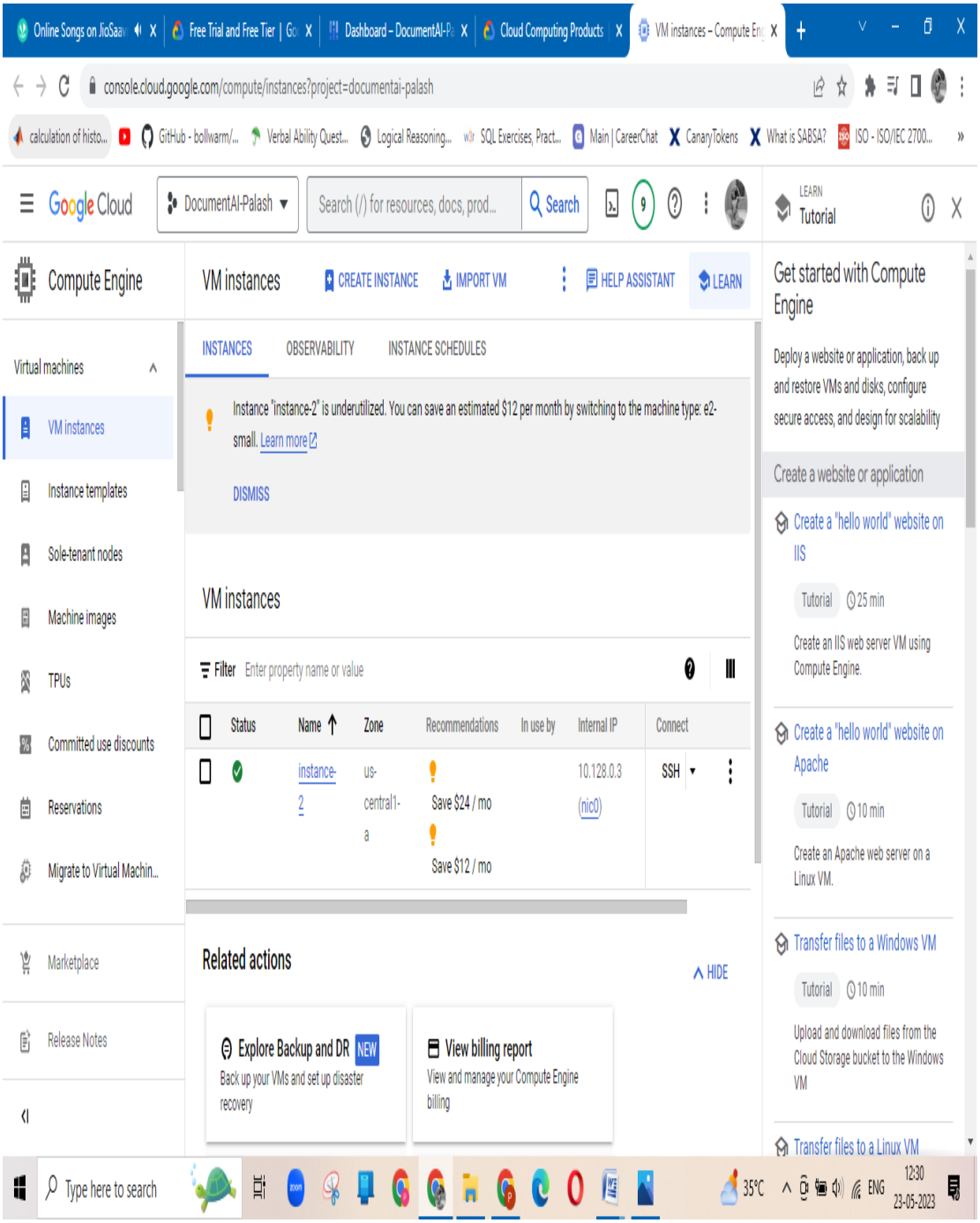


Figure 37: – VM Instances Page

Here in this figure VM (Virtual Machine) Instances page is opened and we are seeing the status of our instance that we have generated.

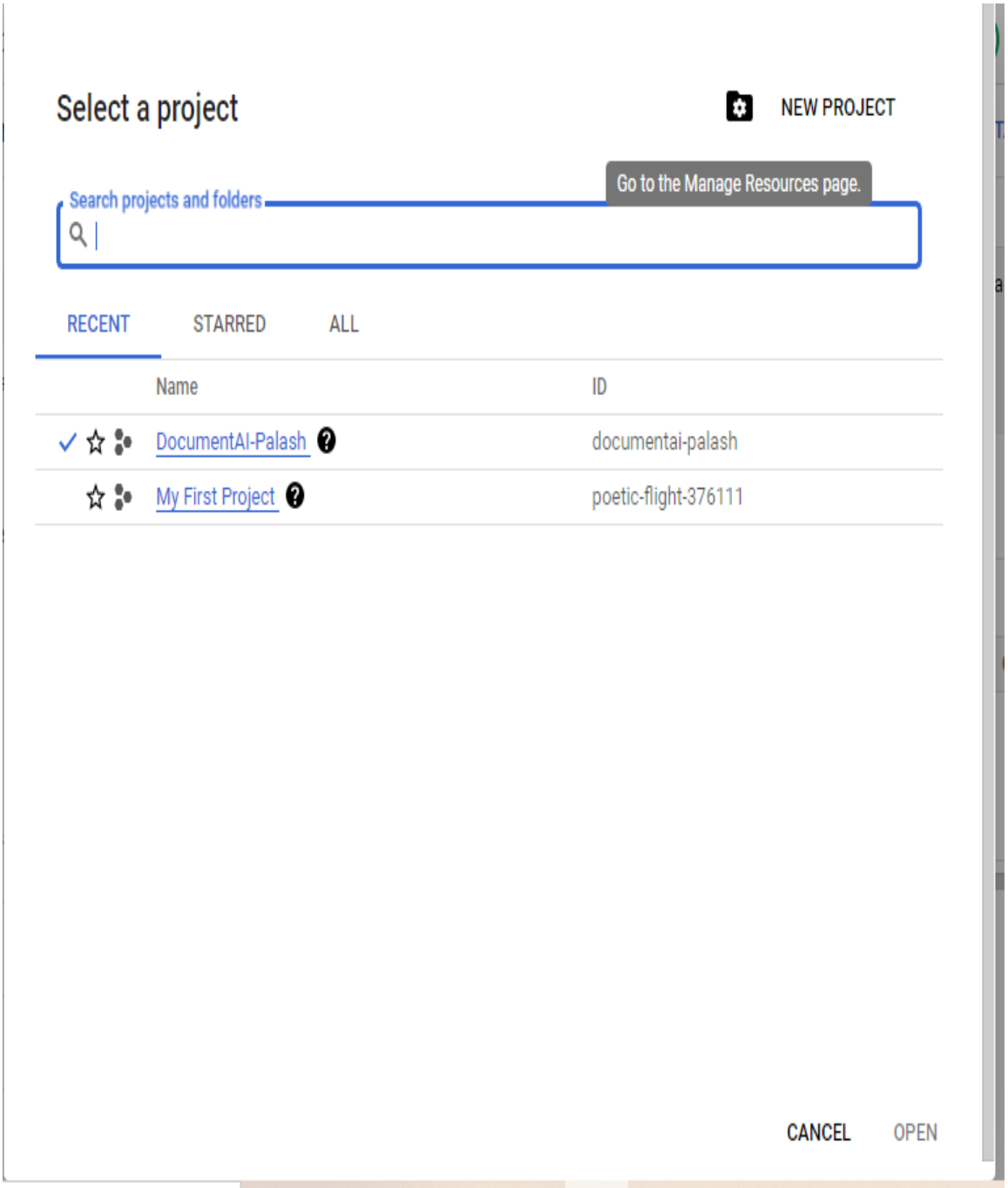


Figure 38:– Choosing Of Project

Herein this figure we are clicking the project which we have created in VM instance page.

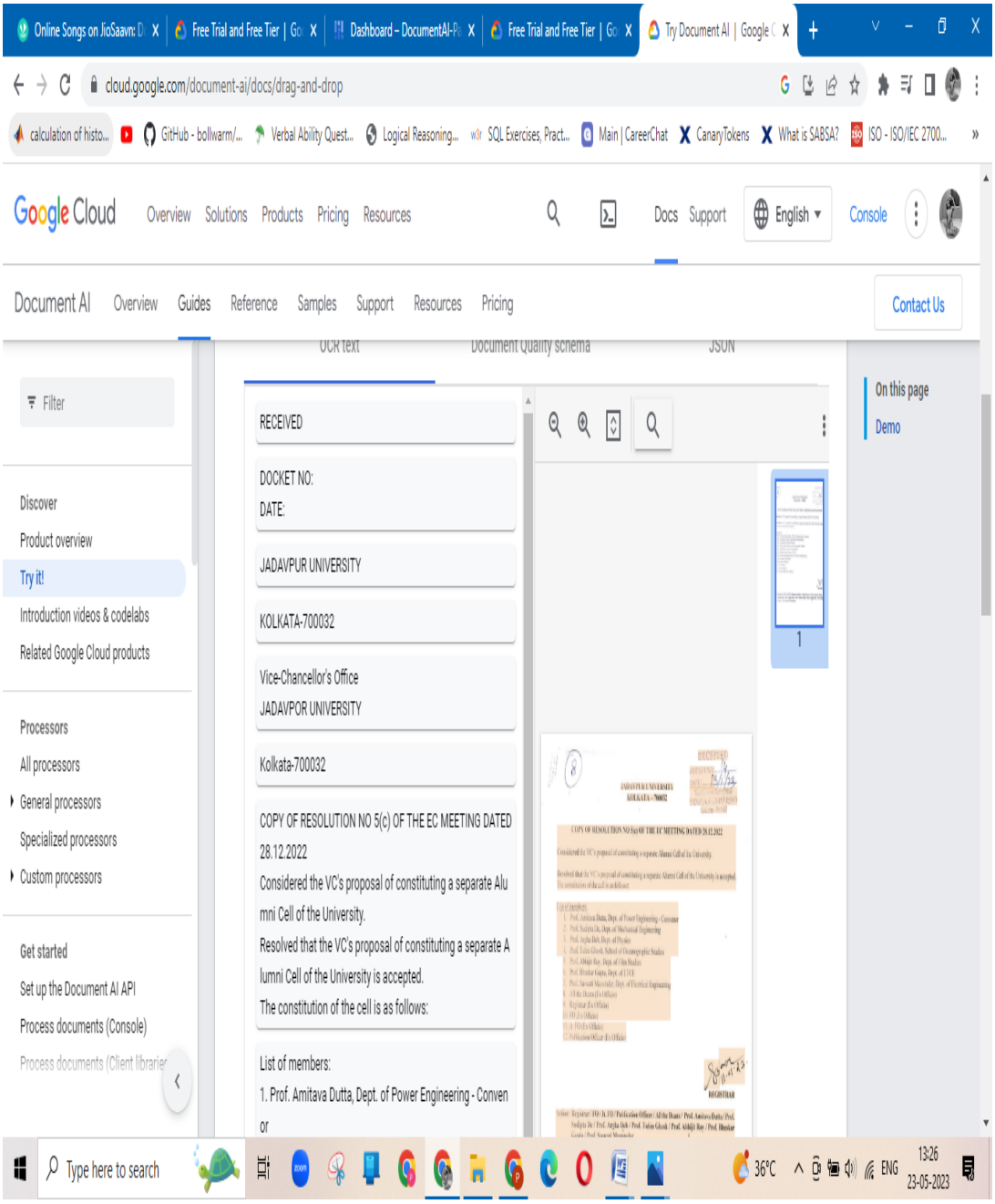


Figure 39:– Document Extraction of a sample student report

Here in this figure we have uploaded the document and extracted the text from it.

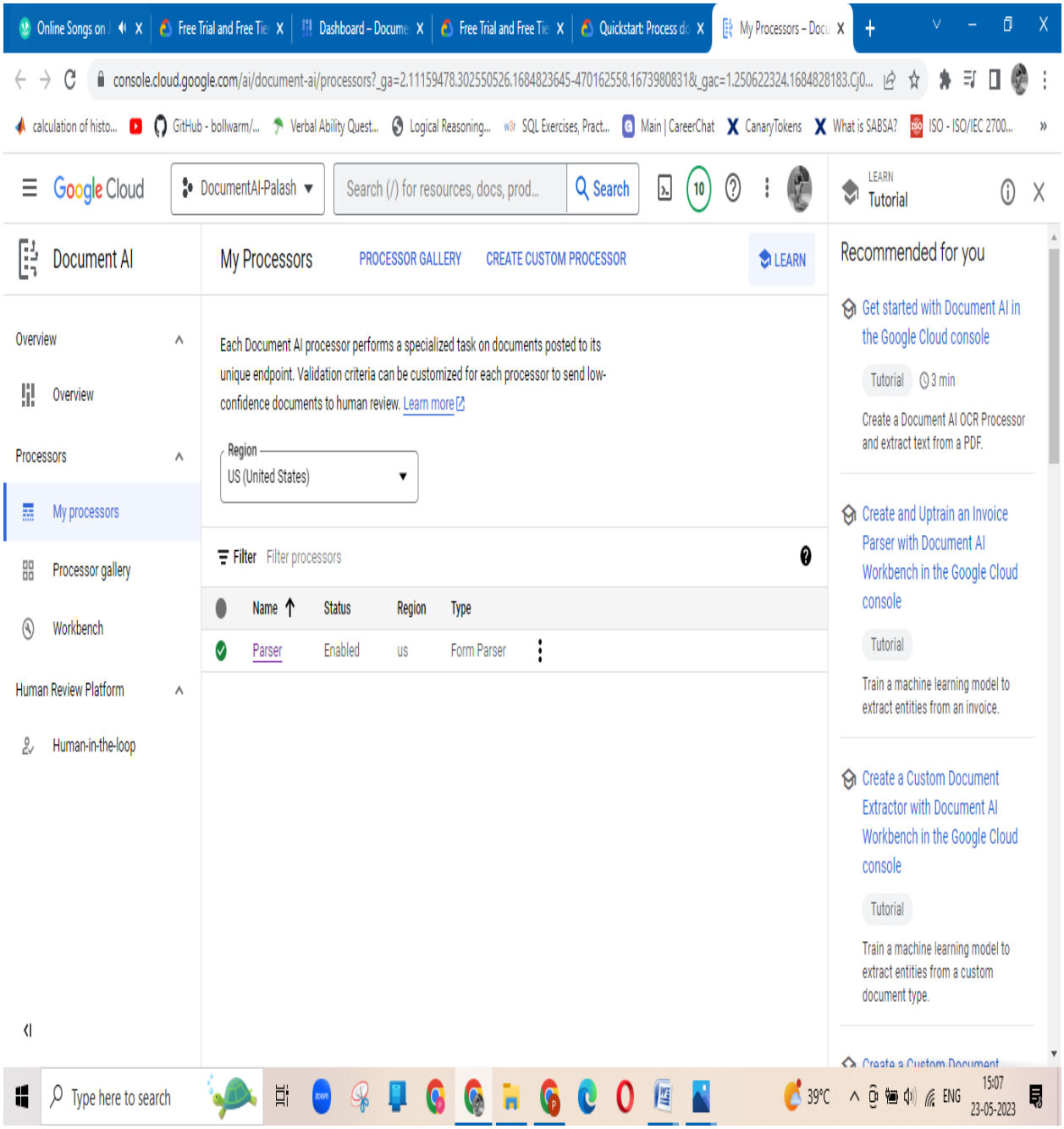


Figure 40: – Processor Page

Here in this figure we are choosing a processor which we will use in our text extraction using Google Document AI.

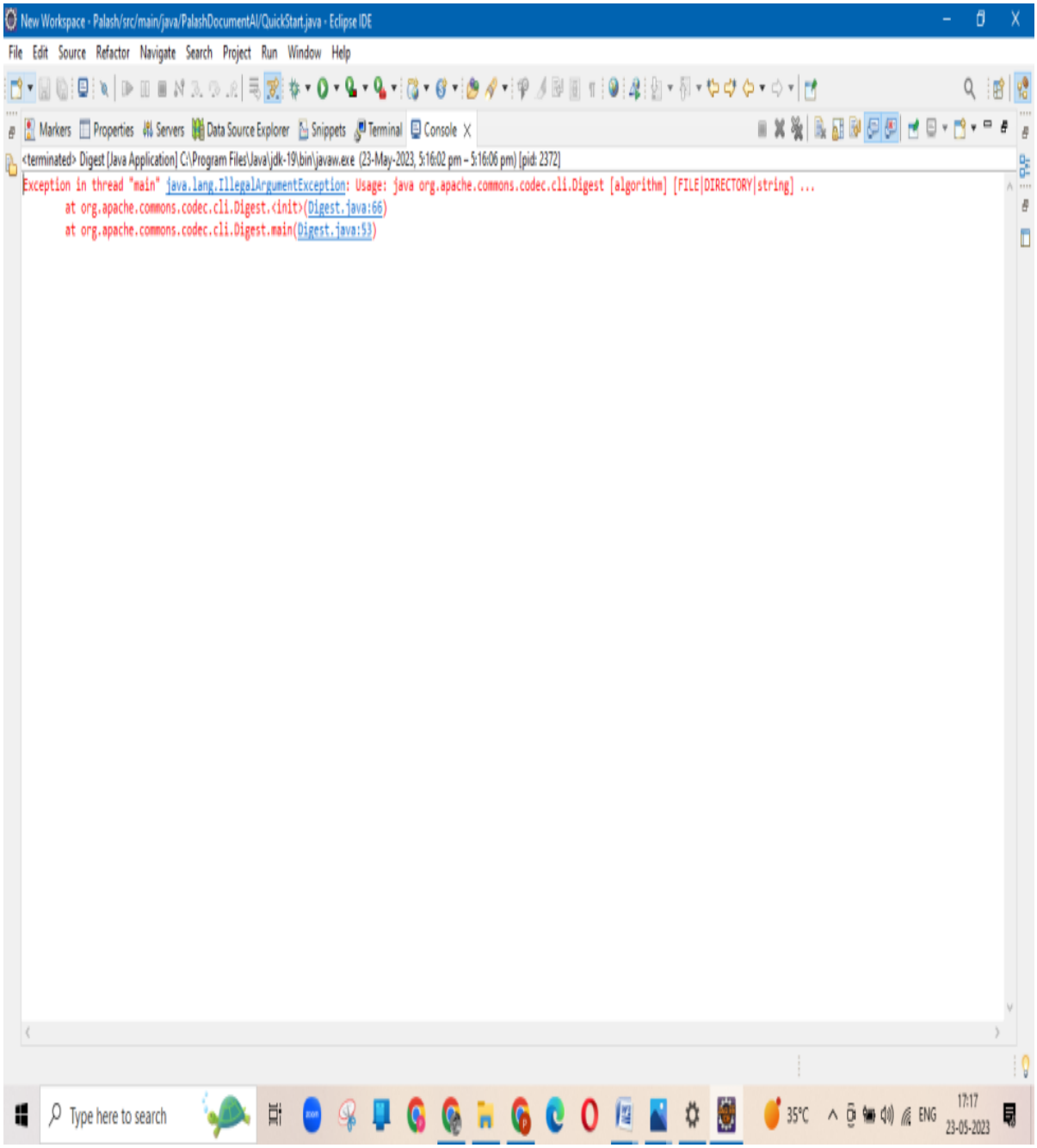


Figure 41:– Output of Programmatic Access in Google Document AI

Here in this figure we are getting output of programmatic access using Google Document AI.

4.***Amazon Textract***

- First we have to register ourselves in Amazon Web Services.
- Then we have to open the Console Home Page by giving the location as NorthVirginia.
- Then the following screen appears.

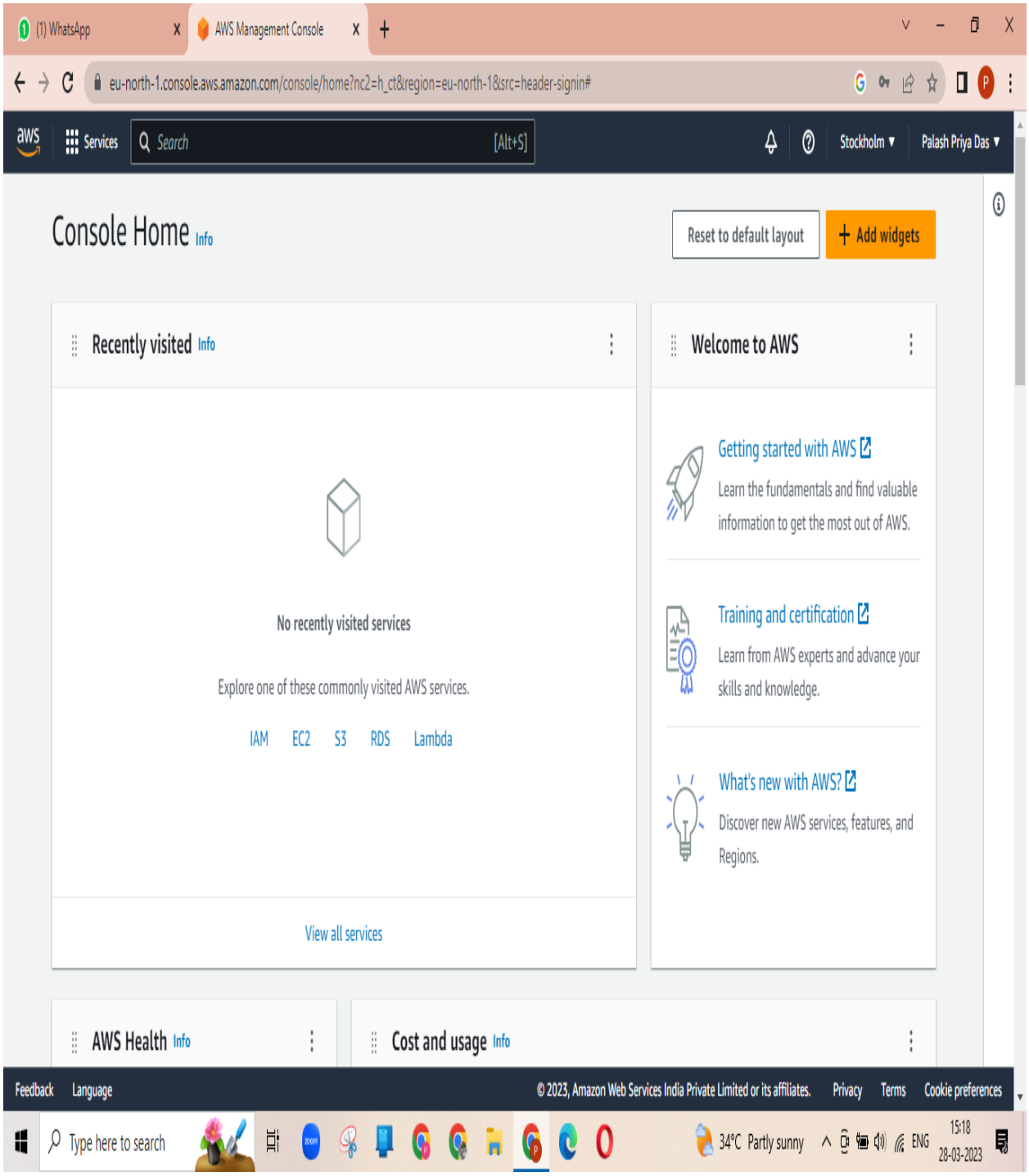


Figure 42:– Console Home Page of AWS

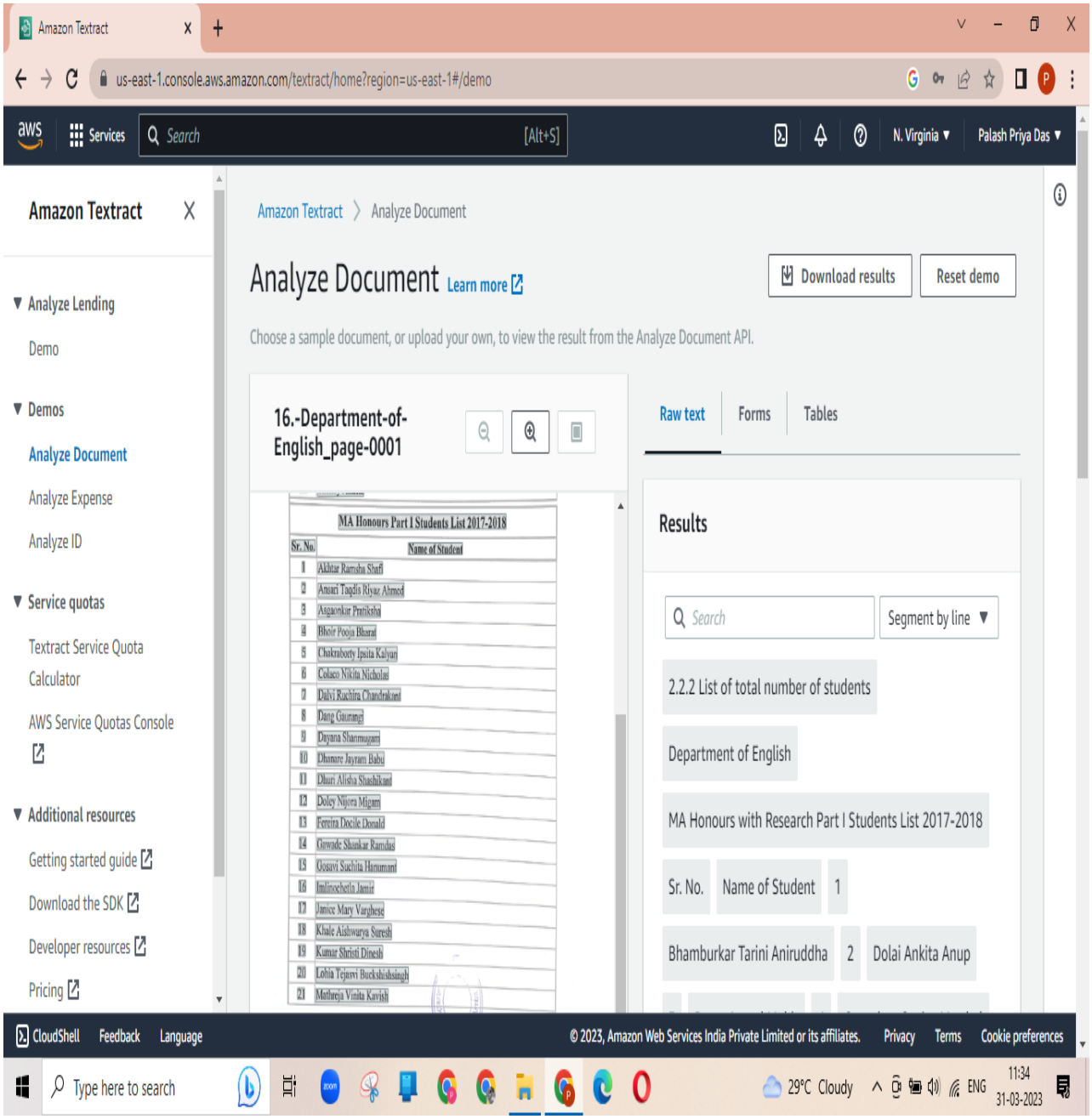


Figure 43:– AWS Text Extraction of a student sample

Here in this figure we are performing text extraction using raw text from a student sample of English MA Honours.

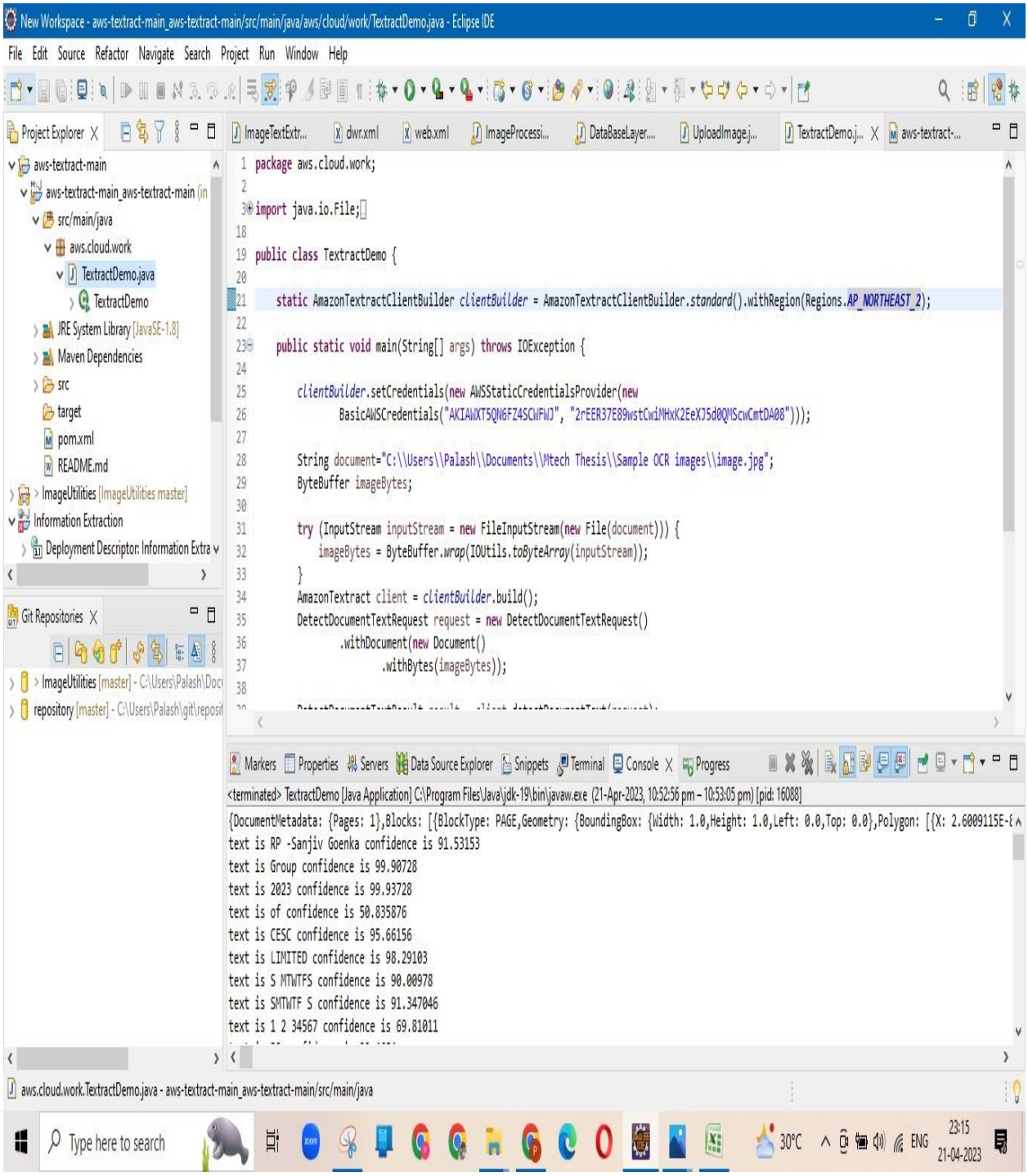


Figure 44: – Output of Programmatic Access in AWS

Here we get the output of programmatic access in AWS of the document that we are used for text extraction.

5.0 Conclusions and Future Scopes

In the 21st century almost everything is digitally connected. The handwritten letters and rarely used printed texts are no longer sent. Instead, we use mobile phones to call and Whats App to share the messages. Our life is dependent on computers which makes our life easier and comfortable. We are currently using digitalize physical papers in a way to be electronically edited, manipulated, searched, managed, stored, and especially interpreted by machines.

Optical Character Recognition made it possible to convert text captured in images, handwritten or printed text into digitized and usable machine coded text.

This is a field of research in Artificial Intelligence and Computer Vision that consists of extraction of text from images.

Today OCR is known to be as an unprecedented revolution and we are grateful for Artificial Intelligence tools. OCR not only becomes an image-to-text traditional conversion process but also a human mistakes checker.

There are many sectors that are using OCR. Banking industry, healthcare industry, logistics companies and academic world are using OCR effectively in their day-today activities.

In future more scope and research is there in this OCR and text extraction field.

We can optimise OCR(Optical Character Recognition) in various fileds like banking, logistics, healthcare and academics.

References

[1] Chandni Kaundilya, Diksha Chawla, Yatin Chopra, "Automated Text Extraction from Images using OCR System", IEEE, 6th International Conference on Computing for Sustainable Global Development (INDIACom), 2019.

[2] Rominkumar Busa, Shahira K C, Lijiya, Graduate student member, IEEE, and Lijiya A, Member, IEEEA "Small Text Extraction from Documents and Chart Images", IEEE , 9th India Council International Conference (INDICON), 2022.

[3] Kamrul Hasan Talukder, Tania Mallick, "Connected Component Based Approach for Text Extraction from Color Image", 17th International Conference on Computer and Information Technology (ICCIT) , 2014.

[4] Anand Kumar, Pardeep Singh, Kusum Lata, "Comparative Study of Different Optical Character Recognition Models on Handwritten and Printed Medical Reports", International Conference on Innovative Data Communication Technologies and Application (ICIDCA) , IEEE, 2023.

[5] Anubhav Kumar , "An Efficient Text Extraction Algorithm in Complex Images" , IEEE , 2013.

[6] Yang Liu, Yonghong Song, Yuanlin Zhang, Quan Meng , "A Novel Multi-Oriented Chinese Text Extraction Approach from Videos" , 12th International Conference on Document Analysis and Recognition, 2013.

Appendix A

➢ System Requirements

Personal Computer or laptop having the following features –

- Intel Core i5
- 16GB RAM
- Internet Connection
- Windows Operating System
- 15.6” HD display,Built in DVD Drive,Wireless and Bluetooth Connection and HDMI
- For accelerated emulator:64 bit operating system and Intel processor.

➢ Software Requirements

- Web-server : Tomcat control panel v3.3.0,
- Server Software Module : MySQL,Eclipse IDE.
- Browser:Windows internet explorer and Google Chrome
- Microsoft word 2010
- Notepad++

➢ Programming Requirements

- JavaScript
- Java

➢ Download Speeds

- Internet speed is measured in Mbps
- 3 -5 Mbps is recommended for the Amazon Textract

➢ Loading Testing Tool

- Google Browser

Appendix B

Programmatic Access Using Google Document AI

```
package PalashDocumentAI;
import com.google.cloud.documentai.v1.Document;
import com.google.cloud.documentai.v1.DocumentProcessorServiceClient;
import com.google.cloud.documentai.v1.ProcessRequest;
import com.google.cloud.documentai.v1.ProcessResponse;
import com.google.cloud.documentai.v1.RawDocument;
import com.google.protobuf.ByteString;
import java.io.IOException;
import java.nio.file.Files;
import java.nio.file.Paths;
import java.util.List;
import java.util.concurrent.ExecutionException;
import java.util.concurrent.TimeoutException;

public class QuickStart {
  public static void main(String[] args)
      throws IOException, InterruptedException, ExecutionException,
TimeoutException {

    String projectId = "DocumentAI-Palash";
    String location = "US";
    String processorId = "36973360d6026a99";
    String filePath = "C:\\Users\\Palash\\AppData\\Local\\Google\\Cloud
SDK\\DocumentAI-Palash.pdf";
    quickStart(projectId, location, processorId, filePath);
  }

  public static void quickStart(
      String projectId, String location, String processorId, String filePath)
      throws IOException, InterruptedException, ExecutionException,
TimeoutException {


    try (DocumentProcessorServiceClient client =
DocumentProcessorServiceClient.create()) {

      String name =
```

```
        String.format("projects/%s/locations/%s/processors/%s", projectId,
location, processorId);

    byte[] imageFileData = Files.readAllBytes(Paths.get(filePath));

    ByteString content = ByteString.copyFrom(imageFileData);

    RawDocument document =

RawDocument.newBuilder().setContent(content).setMimeType("application/pdf
").build();

    ProcessRequest request =

ProcessRequest.newBuilder().setName(name).setRawDocument(document).buil
d();

    ProcessResponse result = client.processDocument(request);
    Document documentResponse = result.getDocument();
    String text = documentResponse.getText();

    System.out.println("The document contains the following paragraphs:");
    Document.Page firstPage = documentResponse.getPages(0);
    List<Document.Page.Paragraph> paragraphs = firstPage.getParagraphsList();

    for (Document.Page.Paragraph paragraph : paragraphs) {
      String paragraphText = getText(paragraph.getLayout().getTextAnchor(),
text);
      System.out.printf("Paragraph text:\n%s\n", paragraphText);
    }
   }
  }

  private static String getText(Document.TextAnchor textAnchor, String text) {
   if (textAnchor.getTextSegmentsList().size() > 0) {
    int startIdx = (int) textAnchor.getTextSegments(0).getStartIndex();
    int endIdx = (int) textAnchor.getTextSegments(0).getEndIndex();
    return text.substring(startIdx, endIdx);
   }
   return "[NO TEXT]";
  }

}
```

XML Code

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/maven-v4_0_0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>aws.cloud.work</groupId>
  <artifactId>aws.textract</artifactId>
  <version>1.0-SNAPSHOT</version>
  <properties>
    <maven.compiler.source>1.8</maven.compiler.source>
    <maven.compiler.target>1.8</maven.compiler.target>
    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
  </properties>
  <dependencies>
        <dependency>
                    <groupId>com.amazonaws</groupId>
                    <artifactId>aws-java-sdk</artifactId>
                    <version>1.11.959</version>
                    <scope>compile</scope>
                </dependency>
                  <dependency>
          <groupId>com.amazonaws</groupId>
          <artifactId>aws-java-sdk-core</artifactId>
          <version>1.11.959</version>
                    <scope>compile</scope>
       </dependency>
    <dependency>
                    <groupId>commons-io</groupId>
                    <artifactId>commons-io</artifactId>
                    <version>2.5</version>
                    <scope>compile</scope>
                </dependency>

                <!-- https://mvnrepository.com/artifact/com.amazonaws/aws-java-
sdk-textract -->

                <dependency>
                    <groupId>com.amazonaws</groupId>
                    <artifactId>aws-java-sdk-textract</artifactId>
                    <version>1.11.959</version>
                    <scope>compile</scope>
```

```
            </dependency>
  </dependencies>

</project>
```

Amazon Textract

```
   package aws.cloud.work;

    import java.io.File;

    import java.io.FileInputStream;

    import java.io.IOException;

    import java.nio.ByteBuffer;

    import java.io.InputStream;


    import com.amazonaws.auth.AWSStaticCredentialsProvider;

    import com.amazonaws.auth.BasicAWSCredentials;

    import com.amazonaws.regions.Regions;

    import com.amazonaws.services.textract.AmazonTextract;

    import com.amazonaws.services.textract.AmazonTextractClientBuilder;

    import com.amazonaws.services.textract.model.DetectDocumentTextRequest;

    import com.amazonaws.services.textract.model.DetectDocumentTextResult;

    import com.amazonaws.services.textract.model.Document;

    import com.amazonaws.util.IOUtils;


    public class TextractDemo {
```

```
   static AmazonTextractClientBuilder clientBuilder =
AmazonTextractClientBuilder.standard().withRegion(Regions.AP_NORTHE
AST_2);


   public static void main(String[] args) throws IOException {


            clientBuilder.setCredentials(new
AWSStaticCredentialsProvider(new


    BasicAWSCredentials("AKIAWXT5QN6FZ4SCWFWJ",
"2rEER37E89wstCwiMHxK2EeXJ5d0QMScwCmtDA08")));


      String document="C:\\Users\\Palash\\Documents\\Mtech
Thesis\\Sample OCR images\\image.jpg";

            ByteBuffer imageBytes;


            try (InputStream inputStream = new FileInputStream(new
File(document))) {

                  imageBytes =
ByteBuffer.wrap(IOUtils.toByteArray(inputStream));

            }

            AmazonTextract client = clientBuilder.build();

            DetectDocumentTextRequest request = new
DetectDocumentTextRequest()

                        .withDocument(new Document()

                                    .withBytes(imageBytes));
```

```
            DetectDocumentTextResult result =
    client.detectDocumentText(request);

            System.out.println(result);



            result.getBlocks().forEach(block ->{

                    if(block.getBlockType().equals("LINE"))

                    System.out.println("text is "+ block.getText() + " confidence
    is "+ block.getConfidence());

            });



        }

}
```