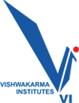
## Department of Electronics Engineering Vishwakarma Institute of Technology, Pune



Course Project Report

Internet of Things

**Title: Deploying Text Summarization Code on AWS**

**BATCH: A3**

**GROUP MEMBERS**

|  |  |  |  |
| --- | --- | --- | --- |
| **Div.** | **Roll No.** | **G.R. No.** | **Name** |
| A | 54 | 11810237 | Dhananjay Joijode |
| A | 57 | 11810764 | Sameer Karoshi |

|  |  |  |
| --- | --- | --- |
| **Sr. No.** | **Content** | **Page No** |
| 1 | Introduction | 1 |
| 2 | Theory | 1 |
| 3 | Methodology | 2 |
| 4 | Results | 9 |
| 5 | Dashboard | 10 |
| 5 | Conclusion | 11 |
| 6 | Reference | 11 |

**Deploying Text Summarization Code on AWS**

**Introduction –**

Hello everyone. Our Project is Text Summarization website using AWS and Flask API. This project is done by me, Dhananjay Joijode and Sameer Karoshi.

Text Summarization is an effective technique to extract important phrases and sentences from a long and monotonous document. It has applications in education, research, news articles, etc. But even if a fully functional text summarizer is developed, it cannot be of much use if it works only on our local machine. The code must be deployed on the internet so the people who need it can be benefitted. Keeping this is mind, we have deployed a text summarizer on AWS and created a website wherein the users will enter a short text and the summary will be displayed in few sentences.

Now let us have a look at the scripts.

In this project, we have summarized a text using NLP (Natural language Processing). It helps computers to understand and process human languages. We have implemented our project using TF-IDF (Term Frequency-Inverse Document Frequency) algorithm.

We have created an API using flask in python and saved it as app.py. That API is used to communicate between the website and python script on EC2 instance.

Onto the website now. The input from the website goes to the Summarizer through the Flask API. It summarizes the text, and the path is retraced.

Coming to the working of our project, on the AWS EC2 Server, we have created an instance text\_summarization. All our scripts are on this server which were transferred from the local machine via the FileZilla Software.

Putty is a software used to run scripts on the server. There are a few steps we must follow. We have created a session and we will load it. In the SSH there is authentication where we will insert our private key and a terminal will open.

There are some commands that we must execute. Dir shows us all the directories on the server. We must open the website… directory so we will write that command. We must run the API script app.py. You can see that it is in running state.

Then we go to our web browser and copy this address, paste in the address bar along with the port number and our website is up and running. We paste a small text in the text box and bam our summary is generated. This was all about our project. Thank You!

**Theory –**

1. **Extractive Text Summarization**

Automatic text summarization is the task of producing a concise and fluent summary while preserving key information content and overall meaning. There can be two approaches for text summarization, abstractive and extractive.

Text Summarization

Abstractive

Extractive

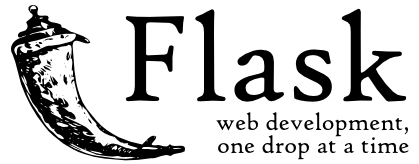
Abstractive - We generate new sentences from the original text. The sentences generated through abstractive summarization might not be present in the original text.

Extractive - We identify the important sentences or phrases from the original text and extract only those from the text. Those extracted sentences would be our summary.

In this project we have used extractive text summarization technique.

1. **AWS EC2 Instance**

We have used EC2 as a web service. Amazon Elastic Compute Cloud (Amazon EC2) is a web service that provides secure, resizable compute capacity in the cloud. It is designed to make web-scale cloud computing easier for developers. Amazon EC2’s simple web service interface allows you to obtain and configure capacity with minimal friction. It provides you with complete control of your computing resources and lets you run on Amazon’s proven computing environment.

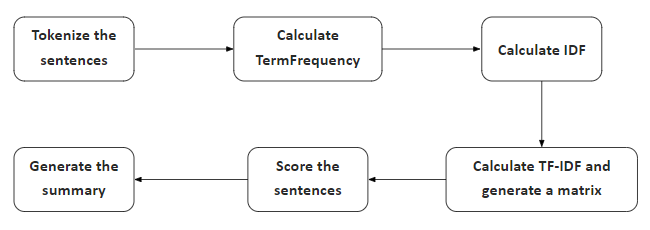
 

1. **Flask API**

API stands for Application Programming Interface. It is a software that allows two applications to communicate with each other. We have used Flask API. Flask is a python framework that provides functionality for building web applications. We have given inputs like a text or a text document to our API.

**Methodology –**

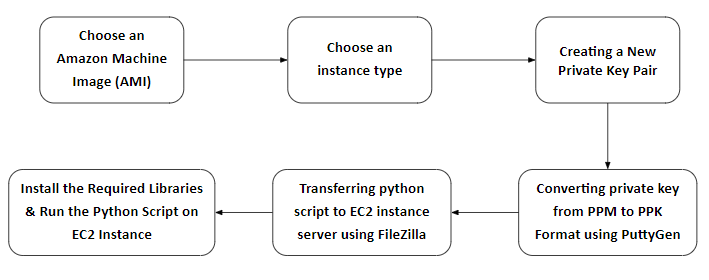
**Text Summarization**



Using natural language processing (NLP) we have done extractive text summarization. Extractive text summarization means important information or sentences are extracted from the given text.

1. For this code we have used following libraries:
2. **NLTK:** NLTK stands for natural language toolkit. NLTK is a leading platform for building Python programs to work with human language data.
3. **MATH:** The python math library provides us access to some common math functions and constants in Python, which we can use throughout our code for more complex mathematical computations.
4. Read the data: Read the data in the form of string in python.
5. Split Text into Sentences (Tokenize the sentences): To break the text into individual sentences and give weight to sentences.
6. Create the Frequency matrix of the words in each sentence: We have calculated the frequency of words in each sentence.
7. Calculate TermFrequency and generate a matrix: We found the TermFrequency for each word in a paragraph.
8. Creating a table for documents per words: This simple table which helps in calculating IDFmatrix.
9. Calculate IDF and generate a matrix: We found the IDF for each word in a paragraph.
10. Calculate TF-IDF and generate a matrix: Multiplying the TF and IDF matrix and generating a new matrix.
11. Score the sentences: The TF-IDF score of words in a sentence to give weight to paragraph.
12. Generate the summary: Select a sentence for a summarization if the sentence score is more than the average score.

**AWS EC2 Instance**



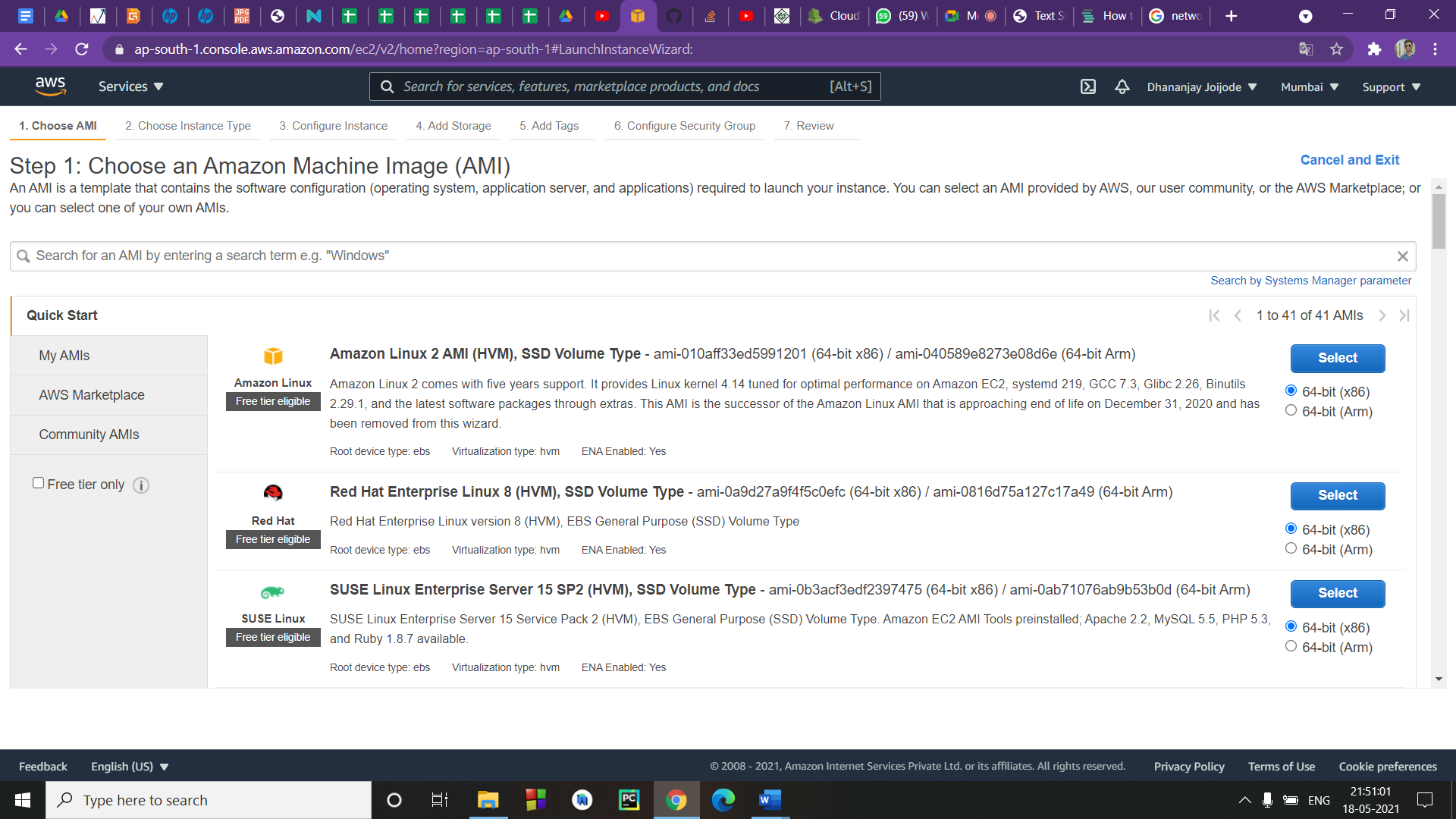
* **Creating AWS EC2 Instance:**

First, we need to create an account to use AWS. After account creation and login to AWS. We need to click on services option and then you will be able to see services. In the compute section click on EC2 and after that click on Launch Instance to create a new EC2 instance.

Following steps need to be followed while creating an EC2 instance.

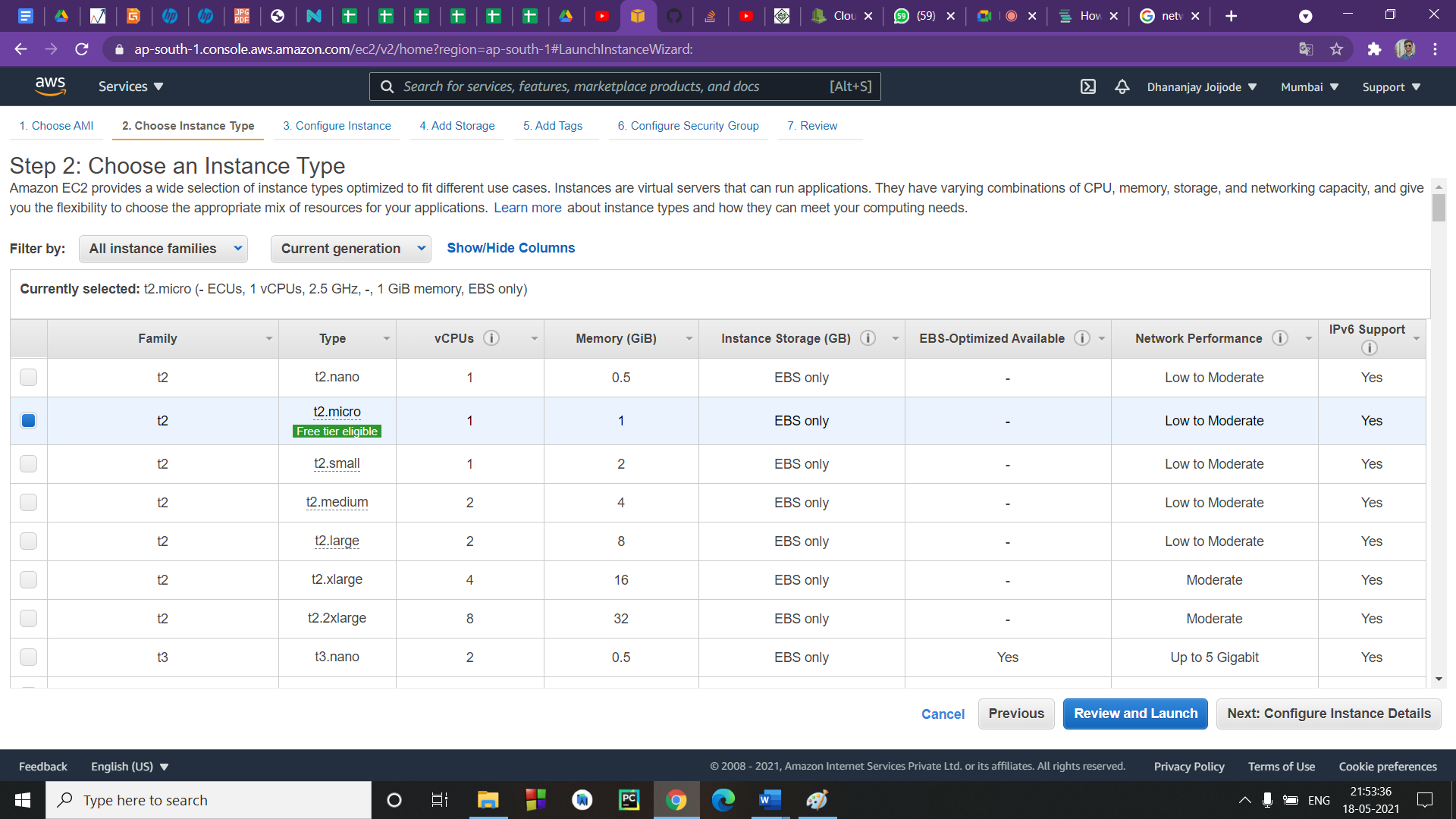
1. **Choose an Amazon Machine Image (AMI):**

We need to choose AMI which means choosing an OS for the server. In our project, we are choosing the Linux which is a free tier and proceed next.



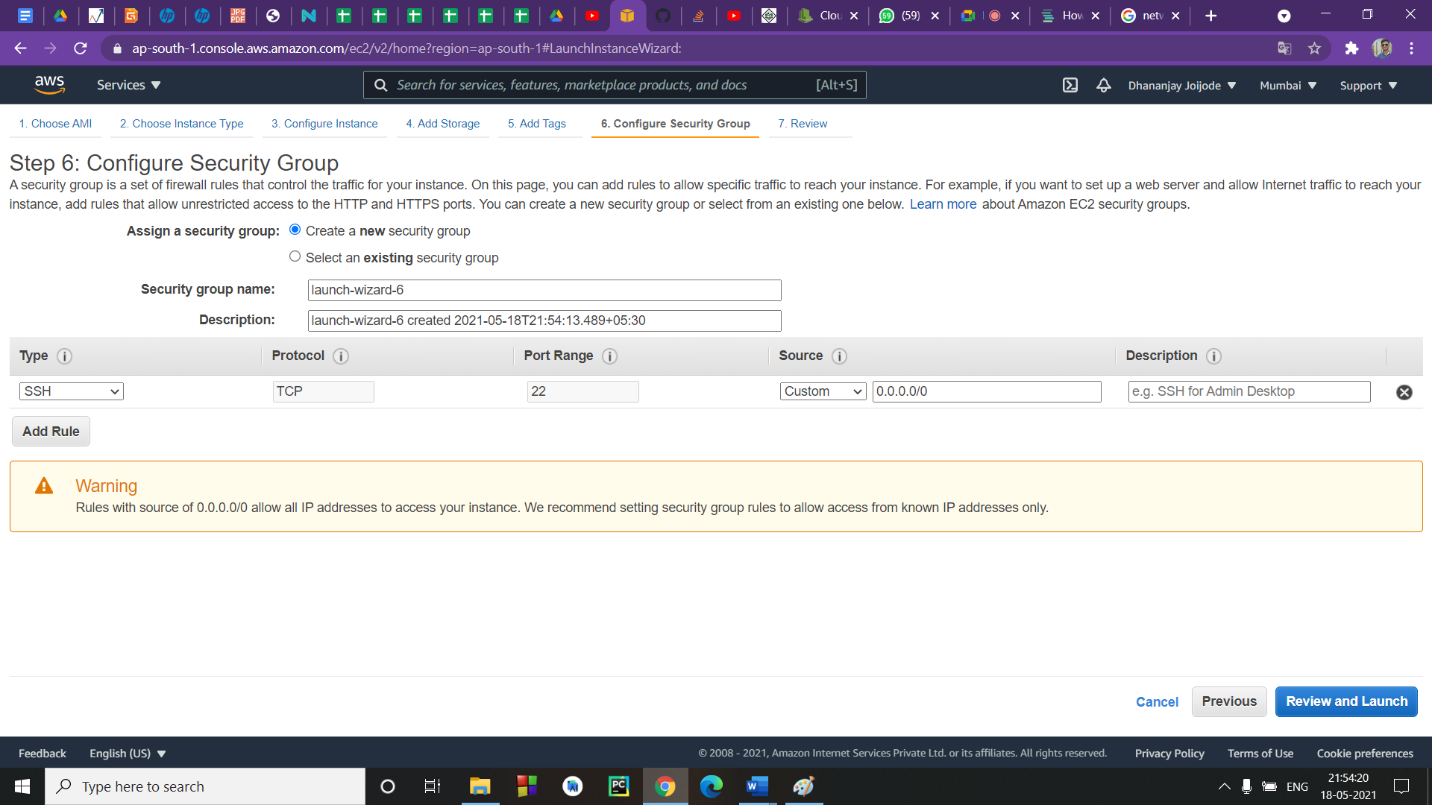
1. **Choose an instance type:**

Amazon EC2 provides a wide selection of instance types optimized to fit different use cases. Instances are virtual servers that can run applications. We need to choose instance type for our server. In this case, we can go with a free tier instance that is t2.micro.



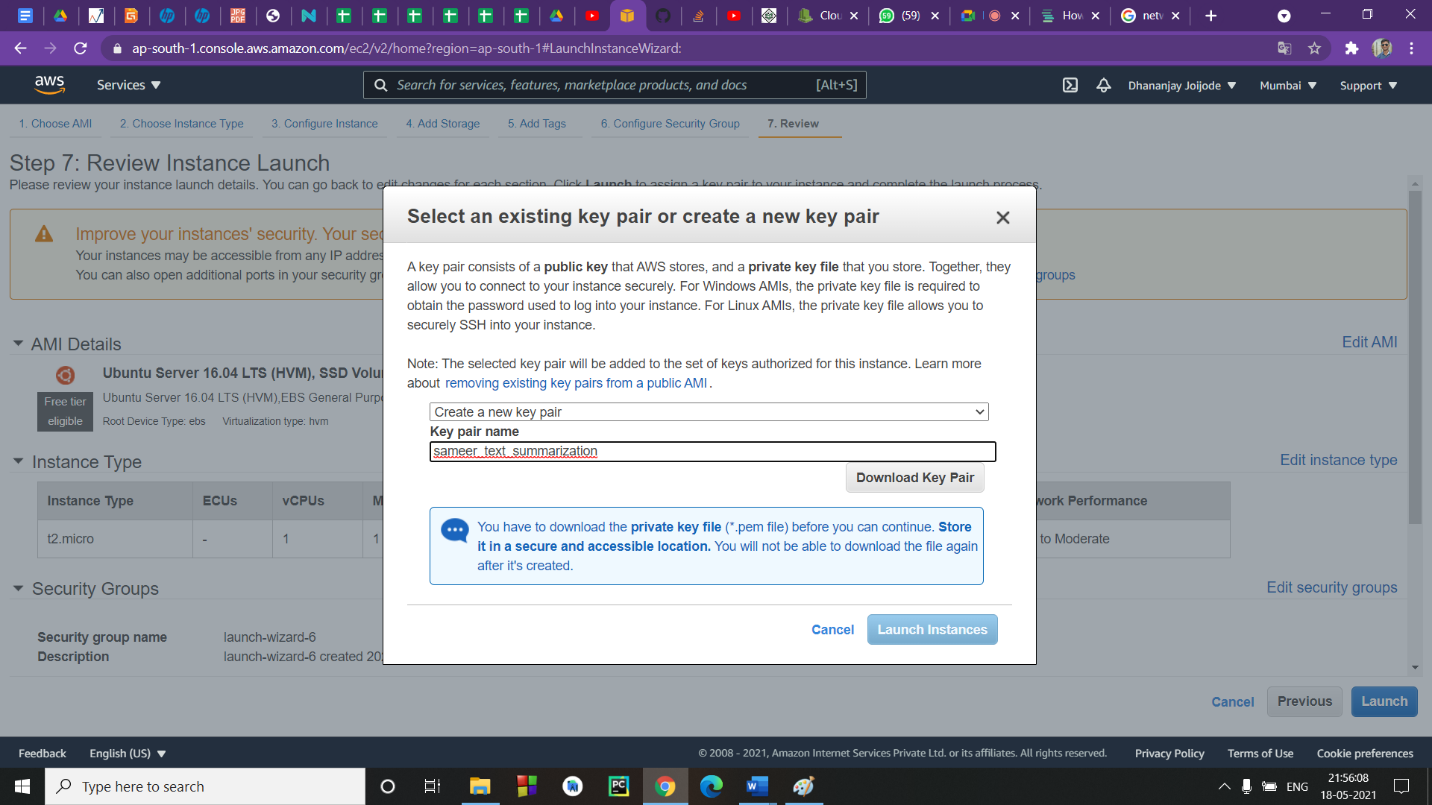
1. **Edit Security Groups:**

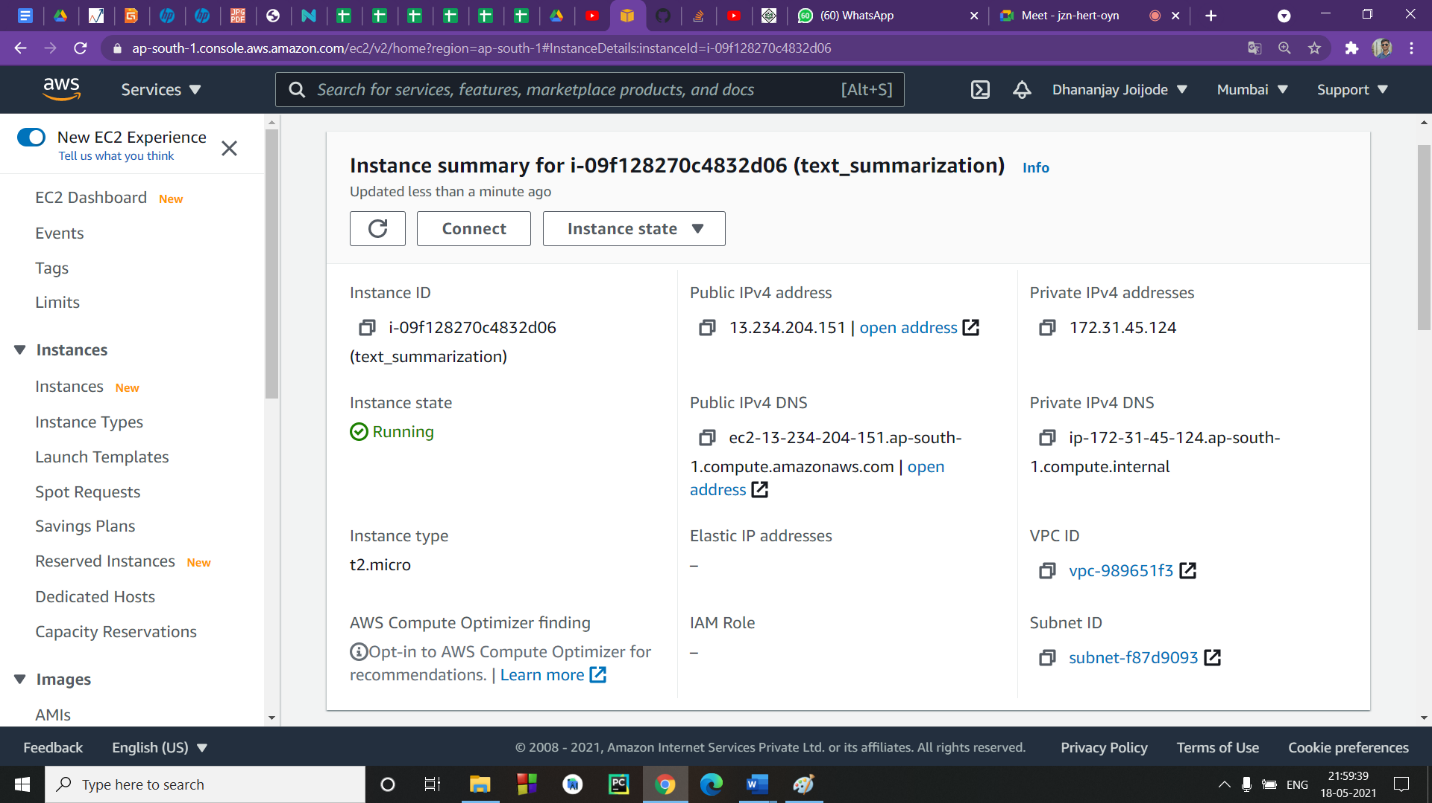
We need to “Edit Security Groups” in the “Review and Launch” page. Once you go to the “Edit Security Groups” Page you need to create a security group. It helps us to edit which kind of traffic that should be allowed while accessing the instance and it helps us to decide which IPs can access our application which allows us to restrict the usage.



1. **Creating a New Private Key Pair:**

We get a pop-up for choosing a private key pair once we review our instance launch. Create a new private key pair and choose a keypair name. We need to download the keypair name.





1. **Downloading Required Software:**

The required software to be downloaded which helps in connecting the local machine with the AWS EC2 instance.

1. **Putty:** It is a free and open-source terminal emulator, serial console, and network file transfer application. It supports several network protocols, including SCP, SSH, Telnet, rlogin, and raw socket connection. It can also connect to a serial port.

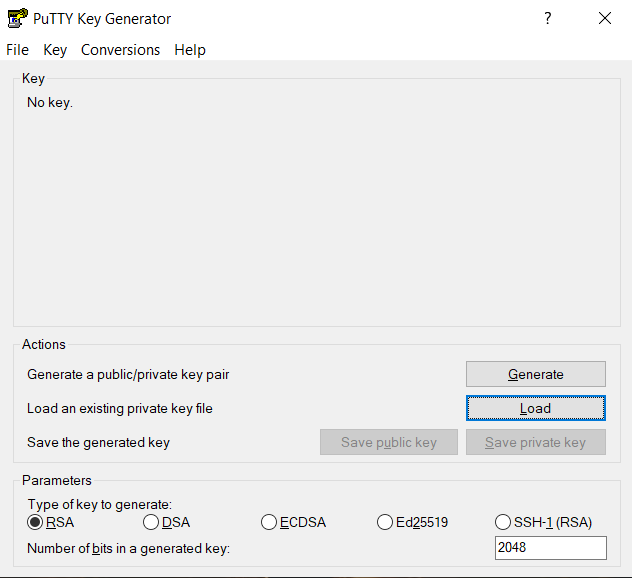
<https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html>

1. **FileZilla:** It is a powerful and free software for transferring files to the Internet or to the AWS EC2 instance. It is a very popular FTP client and is used by webmasters from all over the world.

<https://filezilla-project.org/download.php?platform=win64>

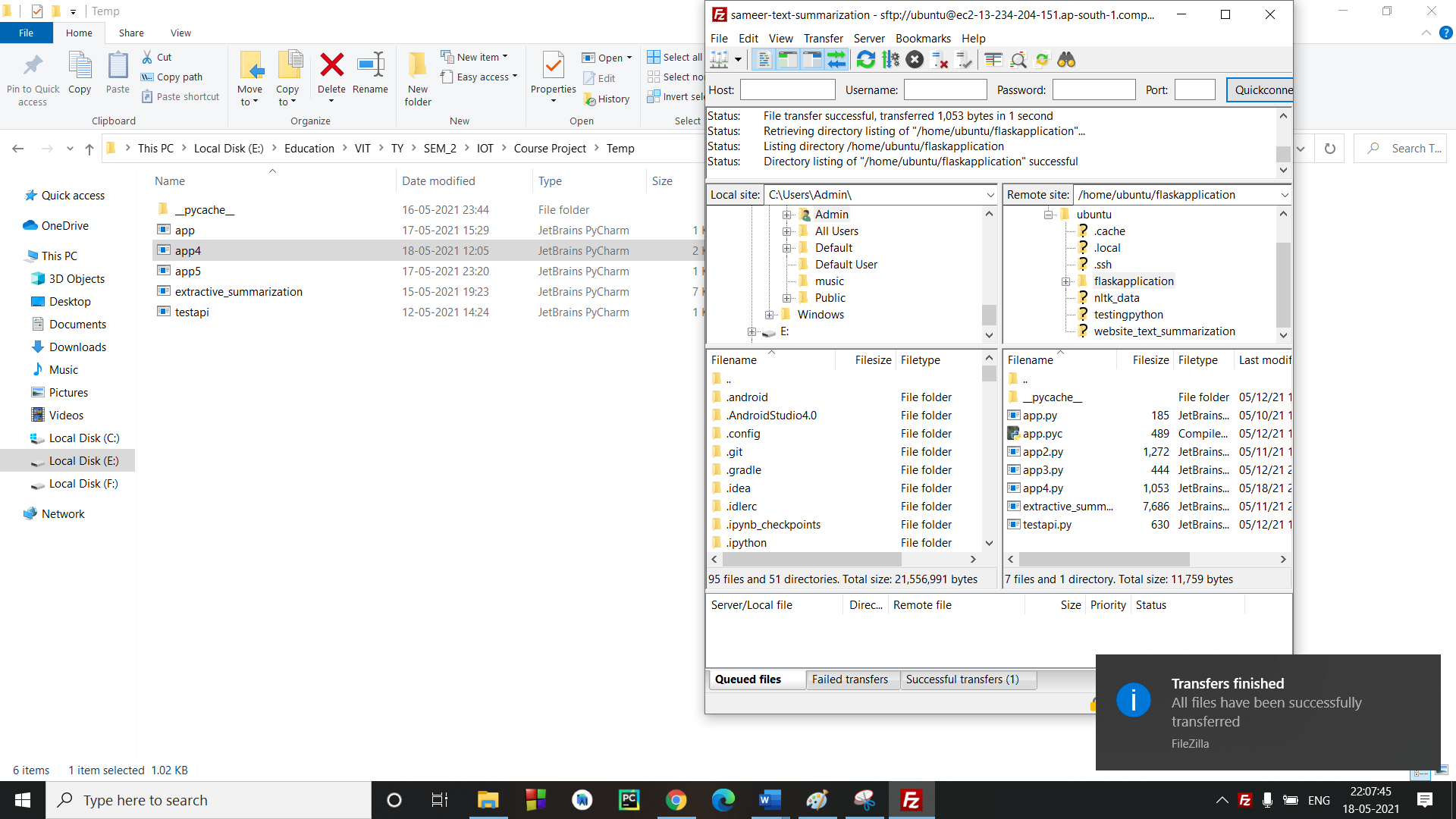
1. **Converting private key from PPM to PPK Format using PuttyGen:**

We need to convert a private key from .ppm extension to .ppk extension using PuttyGen software which we installed earlier. It will be very much useful in the future for getting access to transfer files from local machines to Ubuntu Server. To convert firstly you need to open PuttyGen Software which is already installed by default when you installed Putty.



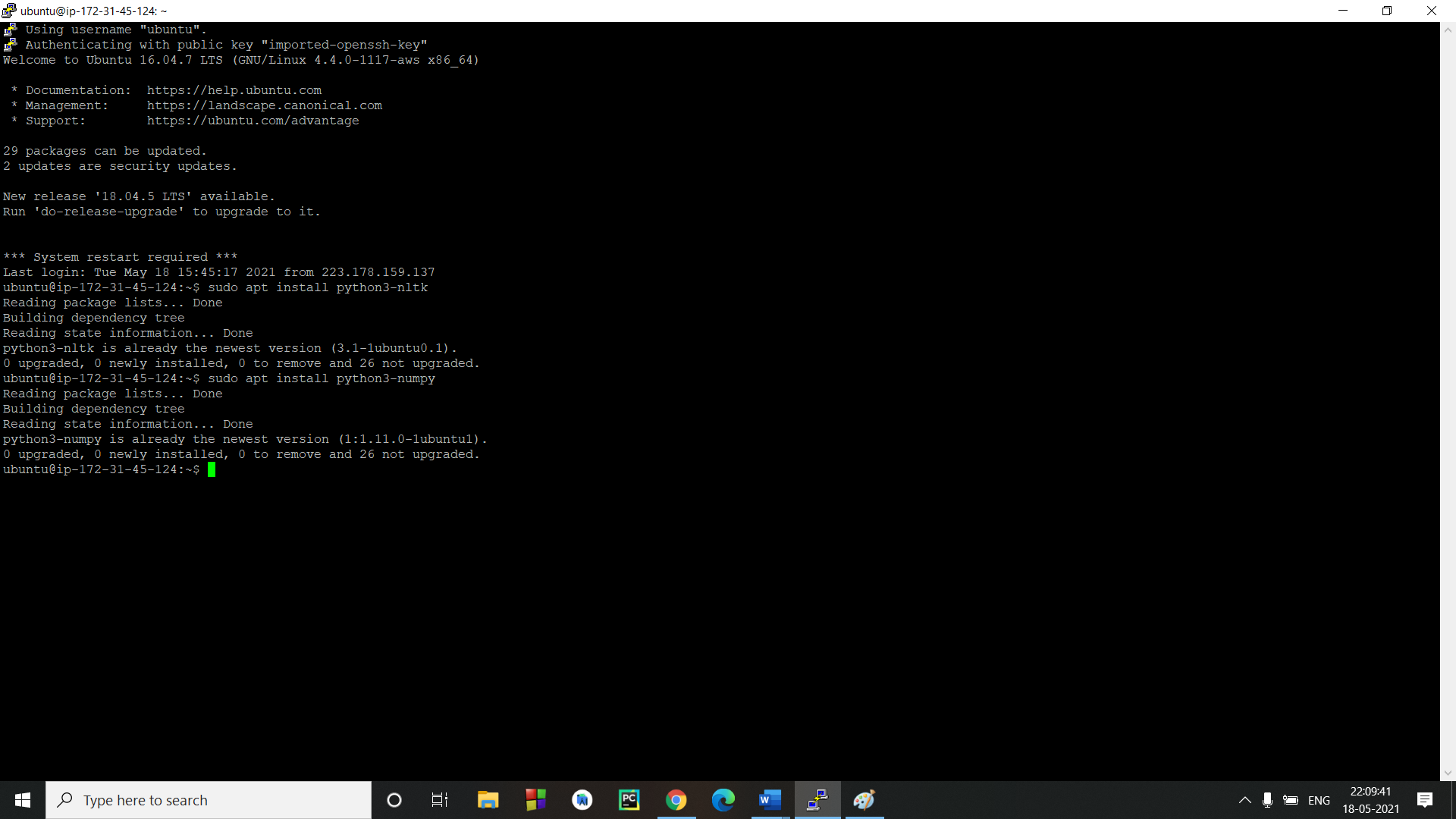
1. **Transferring python script to EC2 instance server using FileZilla:**

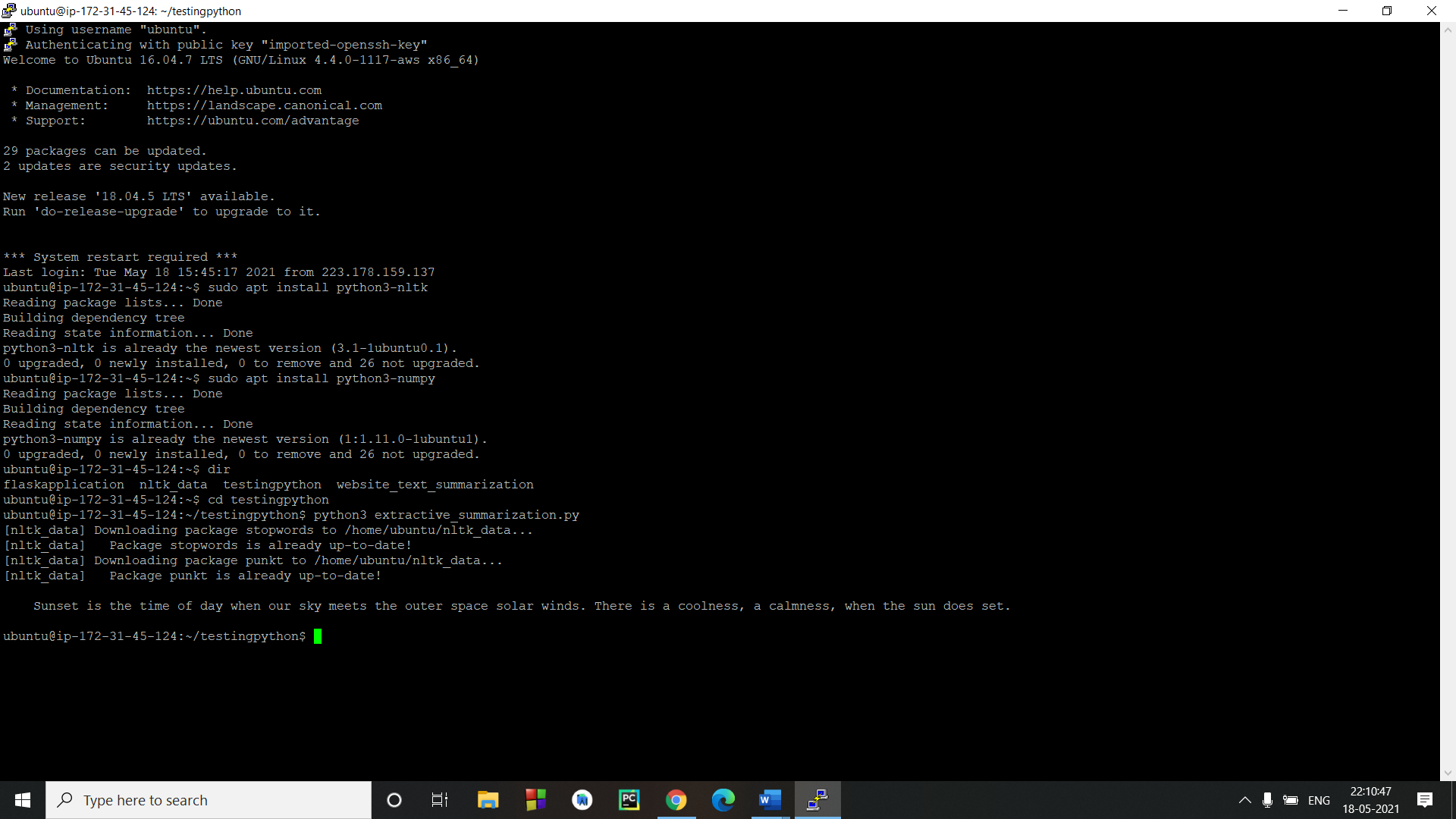
Open FileZilla. Once you open the software click file and open the site manager. In this site manager create a new directory. And then choose SFTP that is protocol for transferring files from your local machine to AWS EC2 instance. Then enter host Ip address and enter username after that click on connect button. Now you successfully connect to EC2 instance and after this open the directory in FileZilla and choose which file you want to transfer to server then drag and drop the file in that directory which is in the FileZilla.



1. **Install the Required Libraries & Run the Python Script on EC2 Instance:**

Then go back to the Putty software and using various commands install required libraries. Once you installed all required libraries then run the python script on putty.





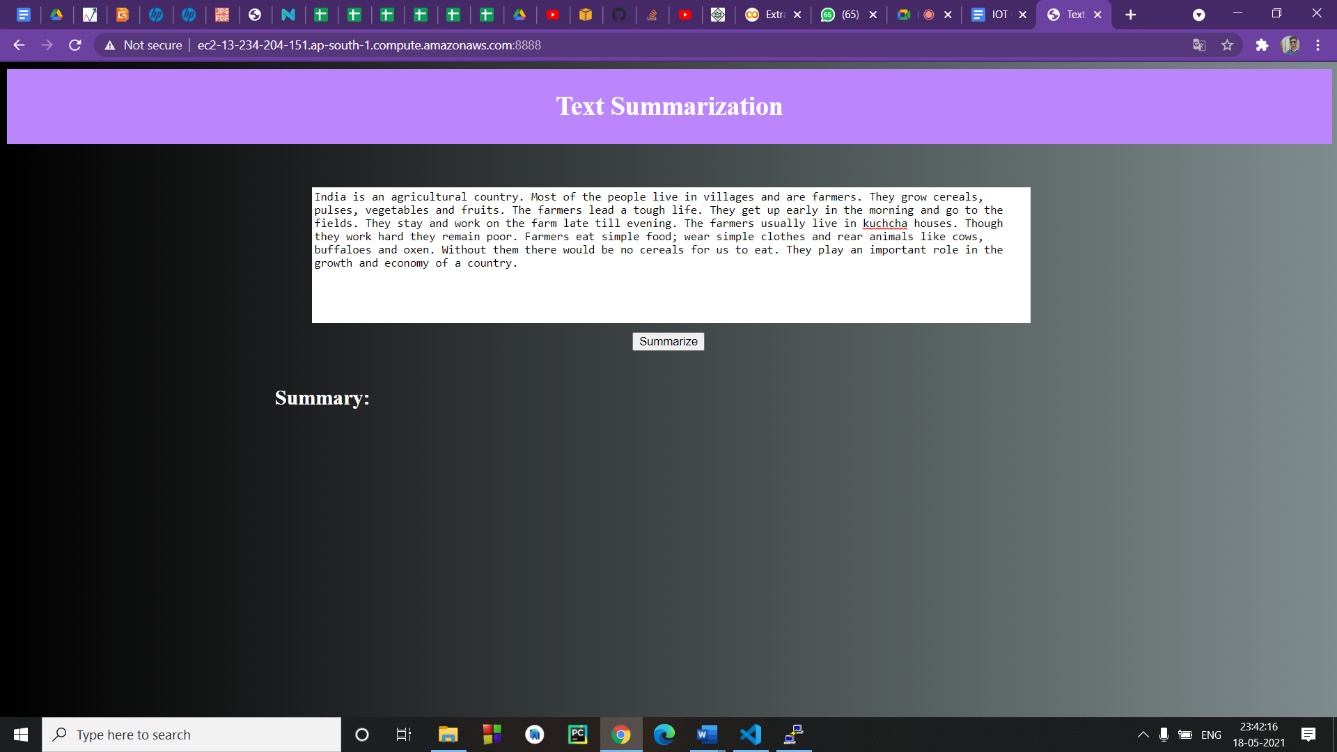
* **Creating API and Website:**

We have created an API using flask in python. That API is used to communicate between the website and python script on EC2 instance. First the input from the website goes to the Summarizer through the Flask API. It summarizes the text, and the path is retraced.

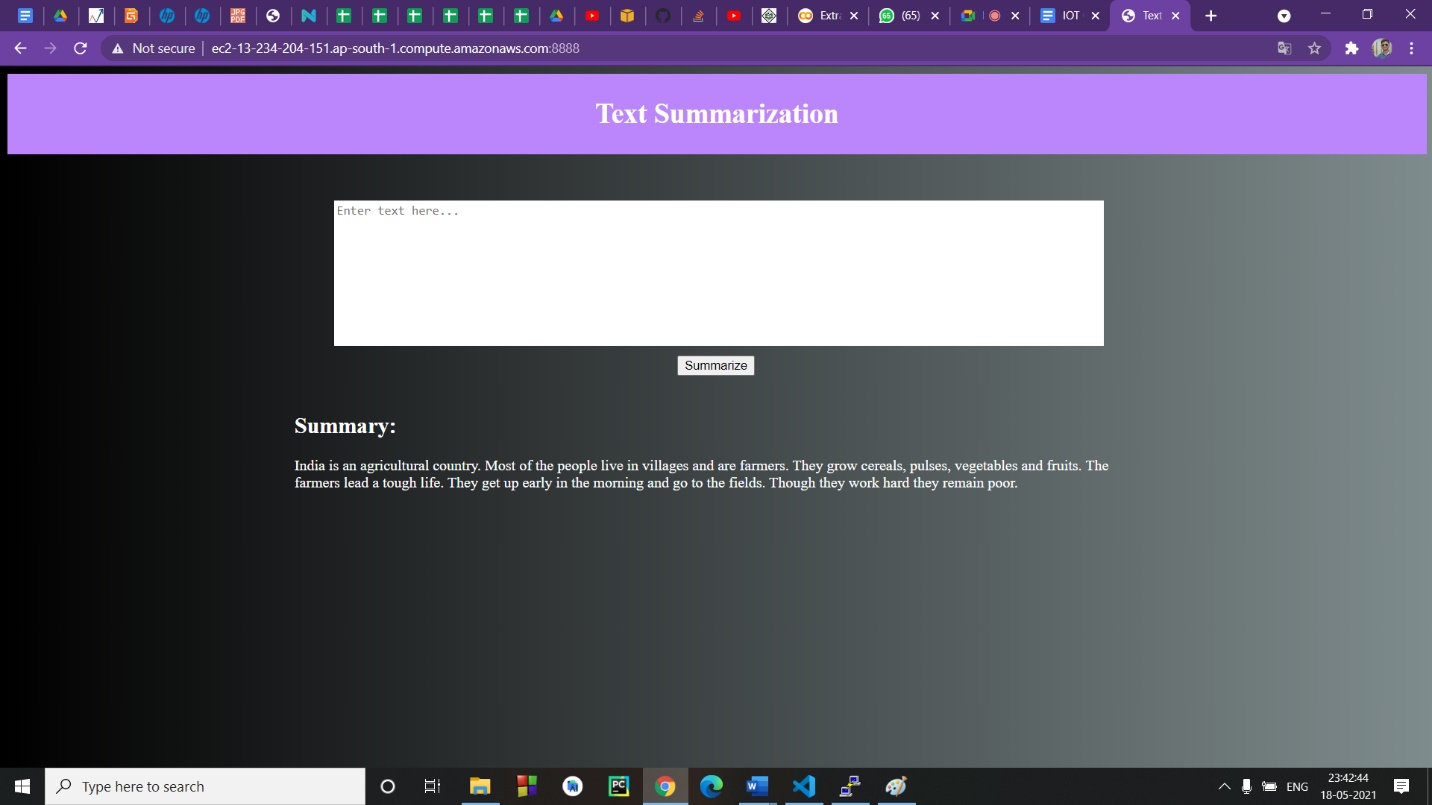


**Results –**

**Website Interface**

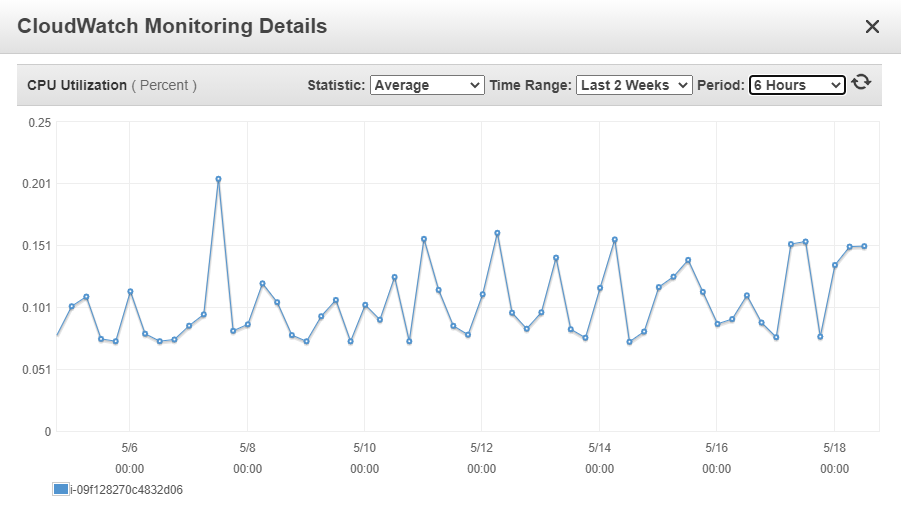


**The Generated Summary**

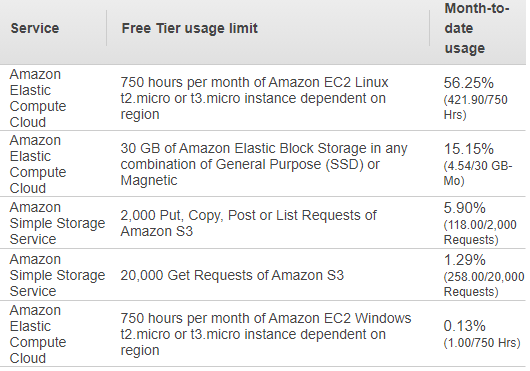


**EC2 Instance Dashboard –**

**CPU Utilization**



**EC2 Instance Usage**



**Conclusion –**

Amazon Web Services is an outstanding platform for students with limitless resources to deploy their codes and show their talents to the world. In this project we learned about the EC2 Instance and how to create an API. We were able to run our python code on EC2 Instance and deployed our website on AWS Server.

**References –**

[1] I. Bermudez, S. Traverso, M. Mellia and M. Munafò, "Exploring the cloud from passive measurements: The Amazon AWS case," 2013 Proceedings IEEE INFOCOM, 2013, pp. 230-234, doi: 10.1109/INFCOM.2013.6566769.

[2] P. Kokkinos, T. A. Varvarigou, A. Kretsis, P. Soumplis and E. A. Varvarigos, "Cost and Utilization Optimization of Amazon EC2 Instances," 2013 IEEE Sixth International Conference on Cloud Computing, 2013, pp. 518-525, doi: 10.1109/CLOUD.2013.52.

[3] S. Shokeen and A. Singh, "Deploying an e-commerce website using Amazon Web Services," 2019 International Conference on contemporary Computing and Informatics (IC3I), 2019, pp. 94-100, doi: 10.1109/IC3I46837.2019.9055586.