Architecture Document

**Image Captioning**

Revision No: 1.0

Last Date of Revision: 23/11/2023.

**Document Version Control-**

|  |  |  |  |
| --- | --- | --- | --- |
| Date Issued | Version | Description | Author |
| **23/11/2023** | 1.0 | LLD – V- 1.0 | Mahesh. A |
|  |  |  |  |
|  |  |  |  |

**Contents -**

|  |  |
| --- | --- |
| **Description** | **Page**  **No.** |
| Document Version Control | 2 |
| 1. Introduction | 4 |
| 1.1 What is Low-Level design document | 4 |
| 1.2 Scope | 4 |
| 2. Architecture | 4 |
| 3. Architecture Description | 5 |
| 3.1 Data Accessing | 5 |
| 3.2 Data Pre-Processing | 7 |
| 3.3 Splitting the Data | 9 |
| 3.4 Model Building | 9 |
| 3.5 Create Front End User Module using flask | 9 |
| 3.6 Testing the Model | 12 |
| 4.0 KPI | 17 |

**Abstract**

By using the pre-trained of Image Captioning model user can get the auto caption for the uploaded image…

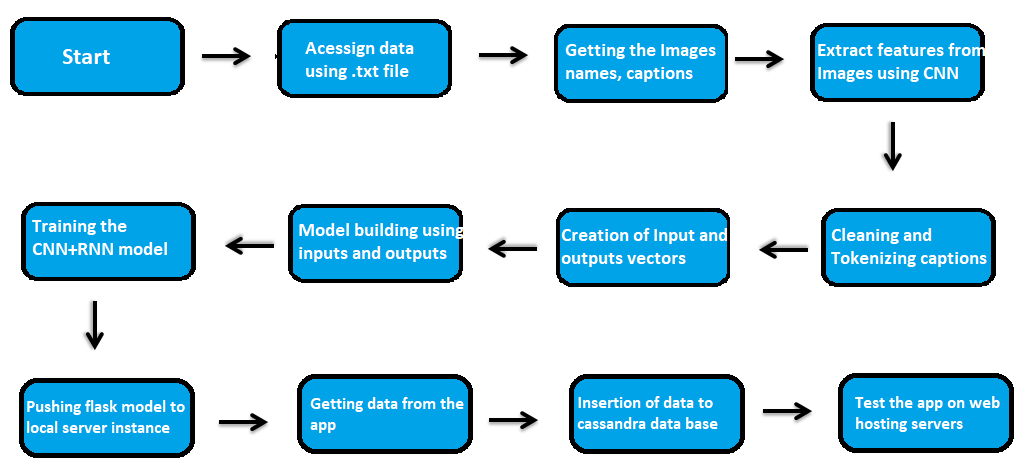
1. **Introduction**
   1. **What is document required?**

The goal of Architecture is to give the internal logical design of the actual programmed code for Image captioning model. Describes the captioned relations with images .It describes the modules so that the programmer can directly code the program from the document.

* 1. **Scope**

Architecture document is a component-level design process that follows a step-by-step process. This process can be used for designing data structures, required software architecture, source code and ultimately, performance algorithms. Overall, the data organization may be defined during requirement analysis and then refined during data design work.

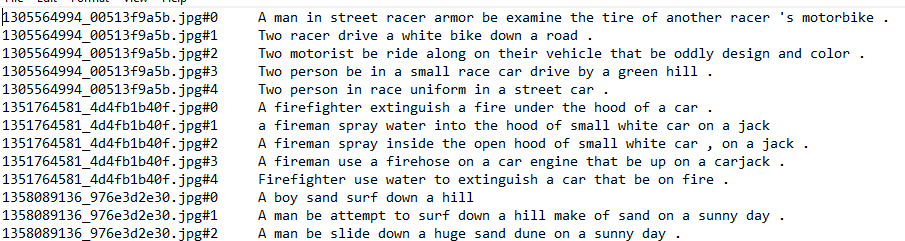
1. **Architecture**

****

1. **Architecture Description**

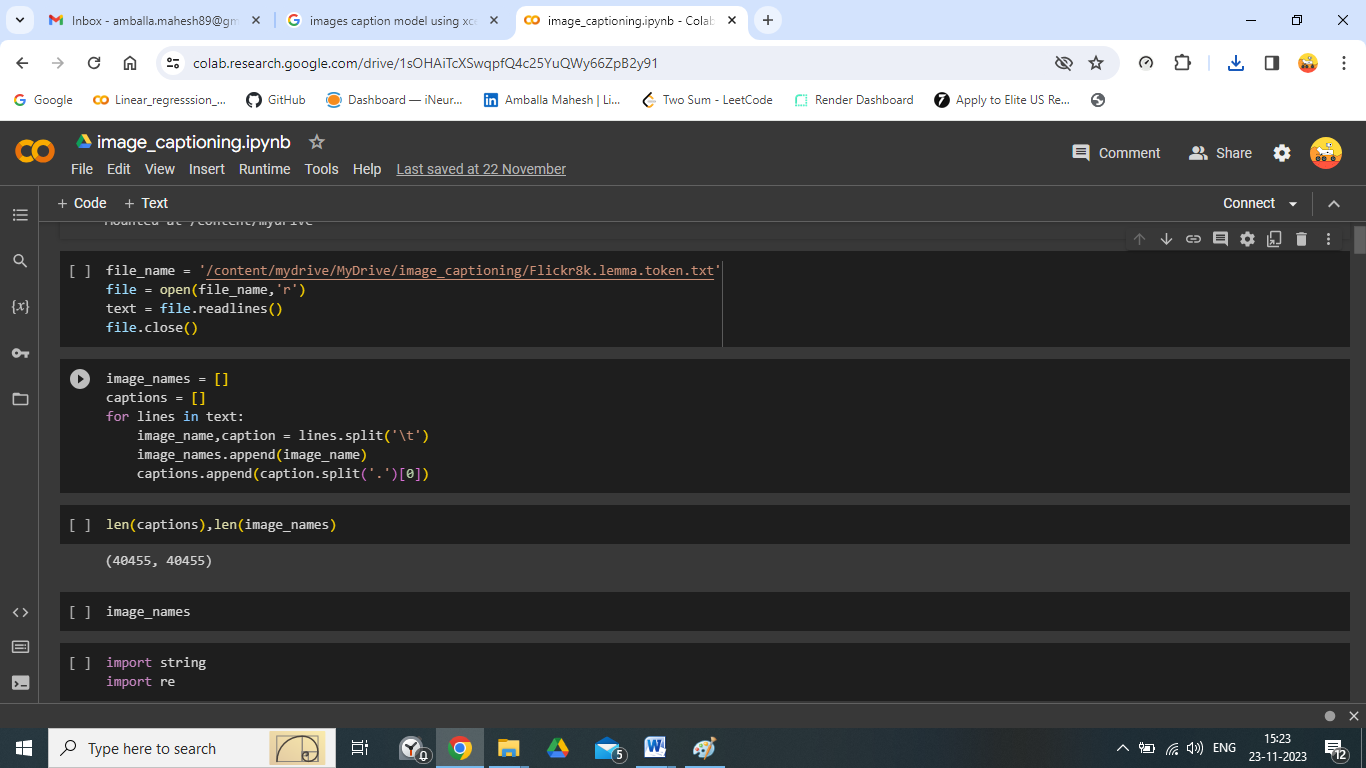
# Data Accessing

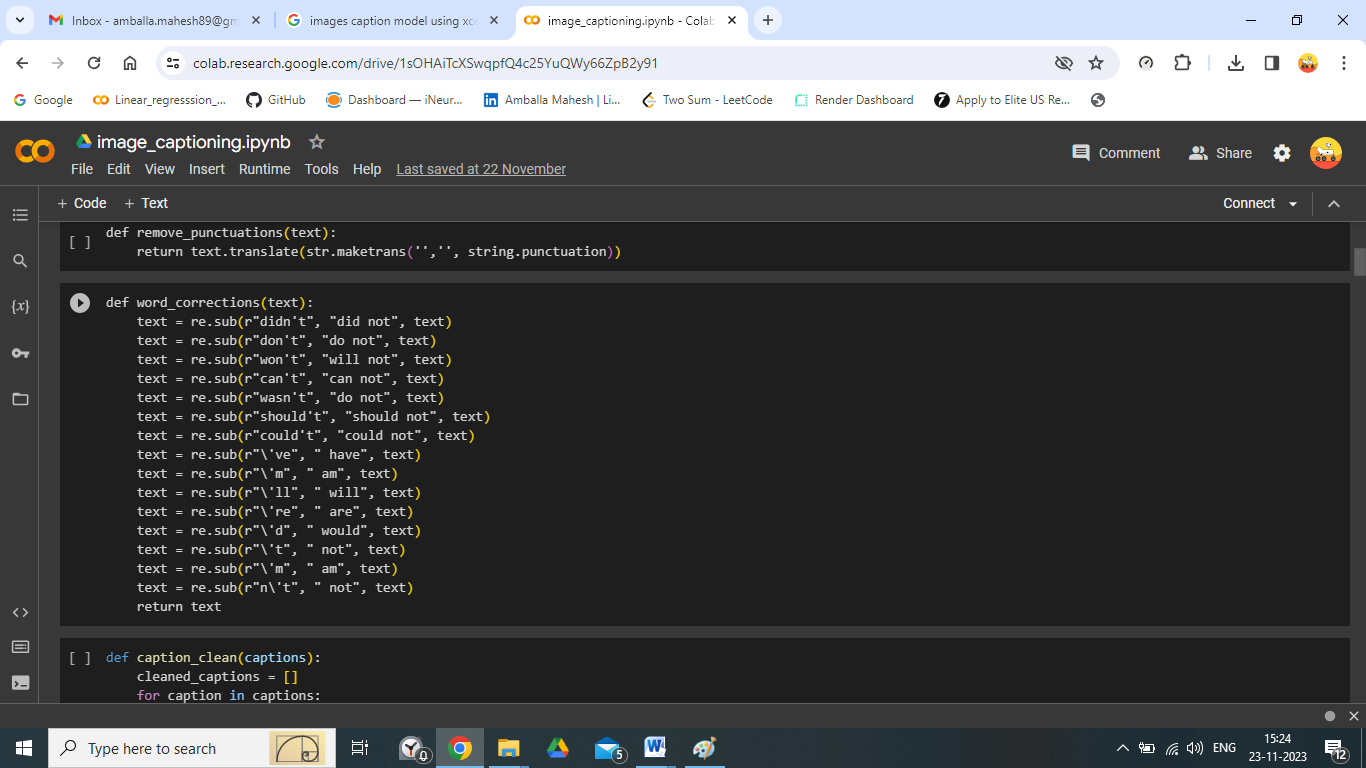
We can access the data in the from the download link as available in the project. We load the data to the framework using the pandas read function.

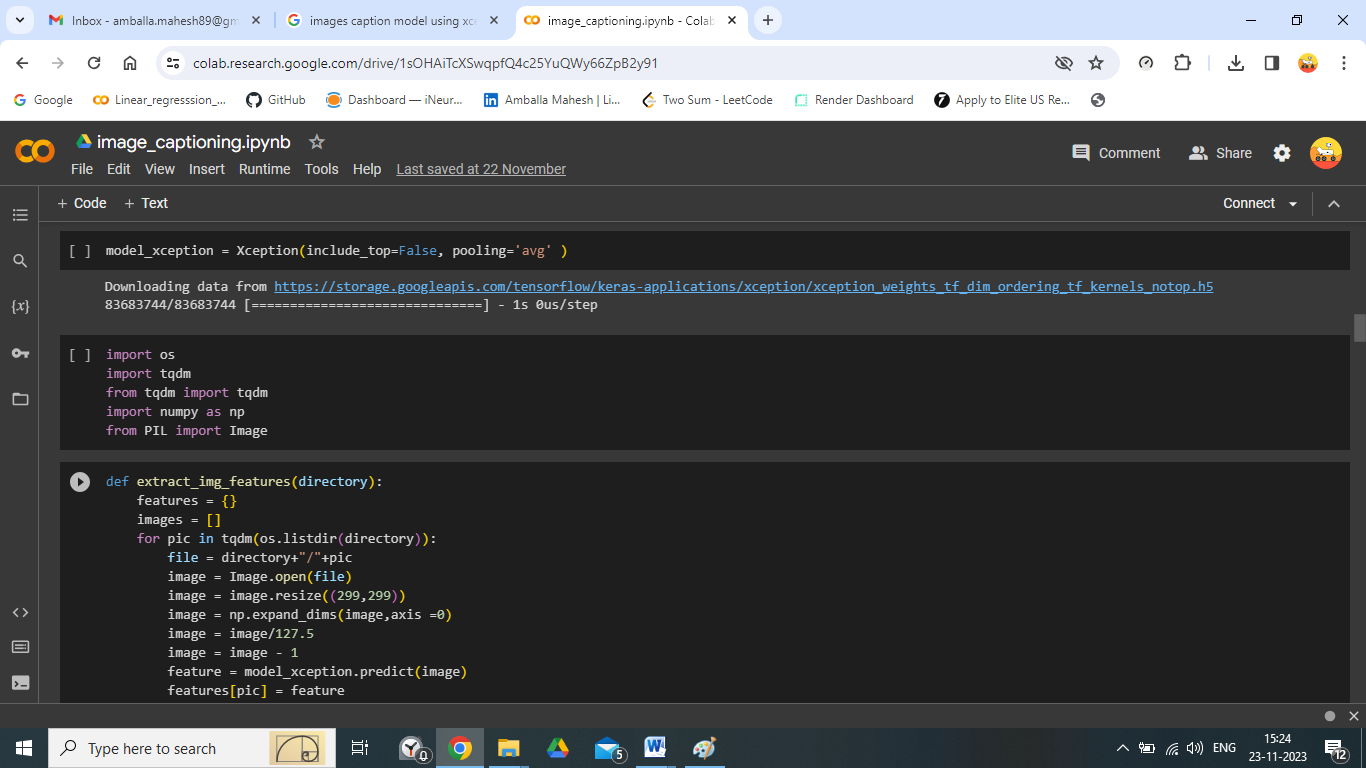


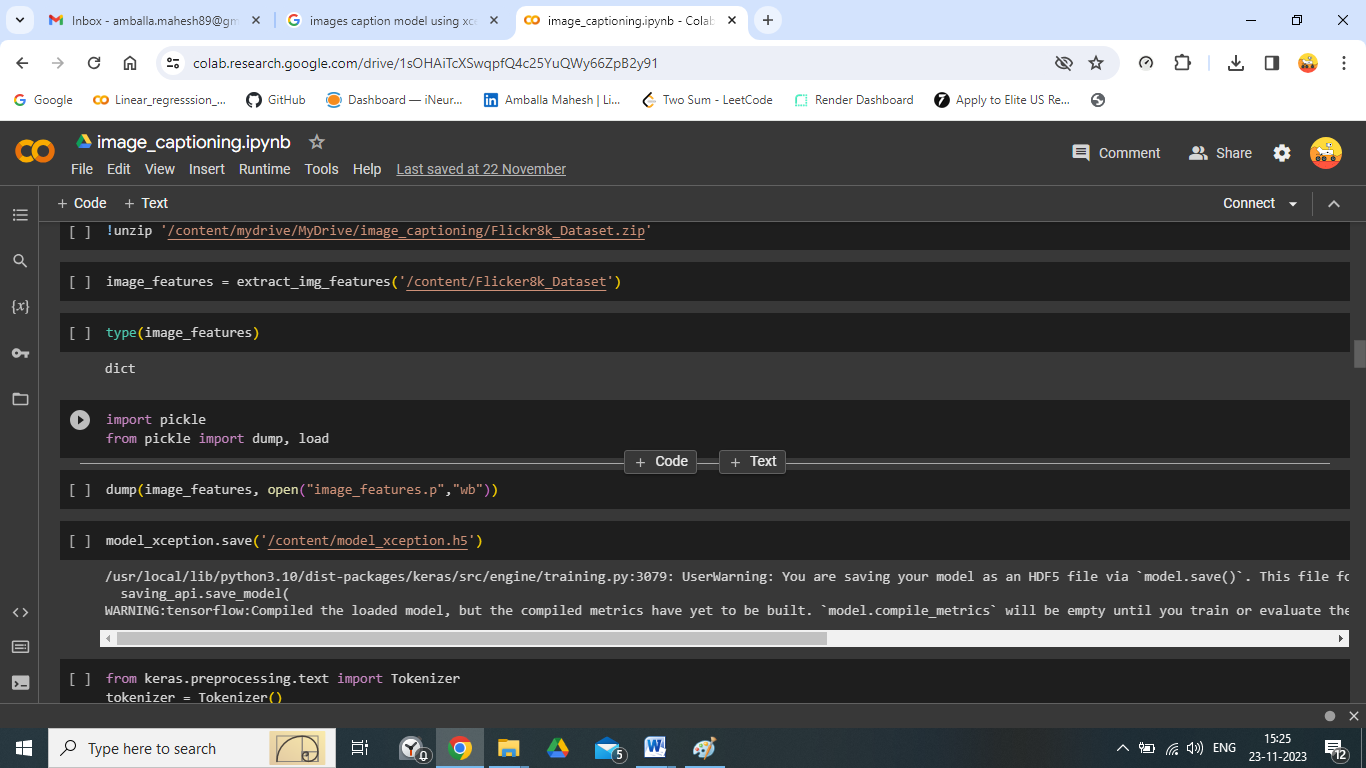
# Data Pre-Processing

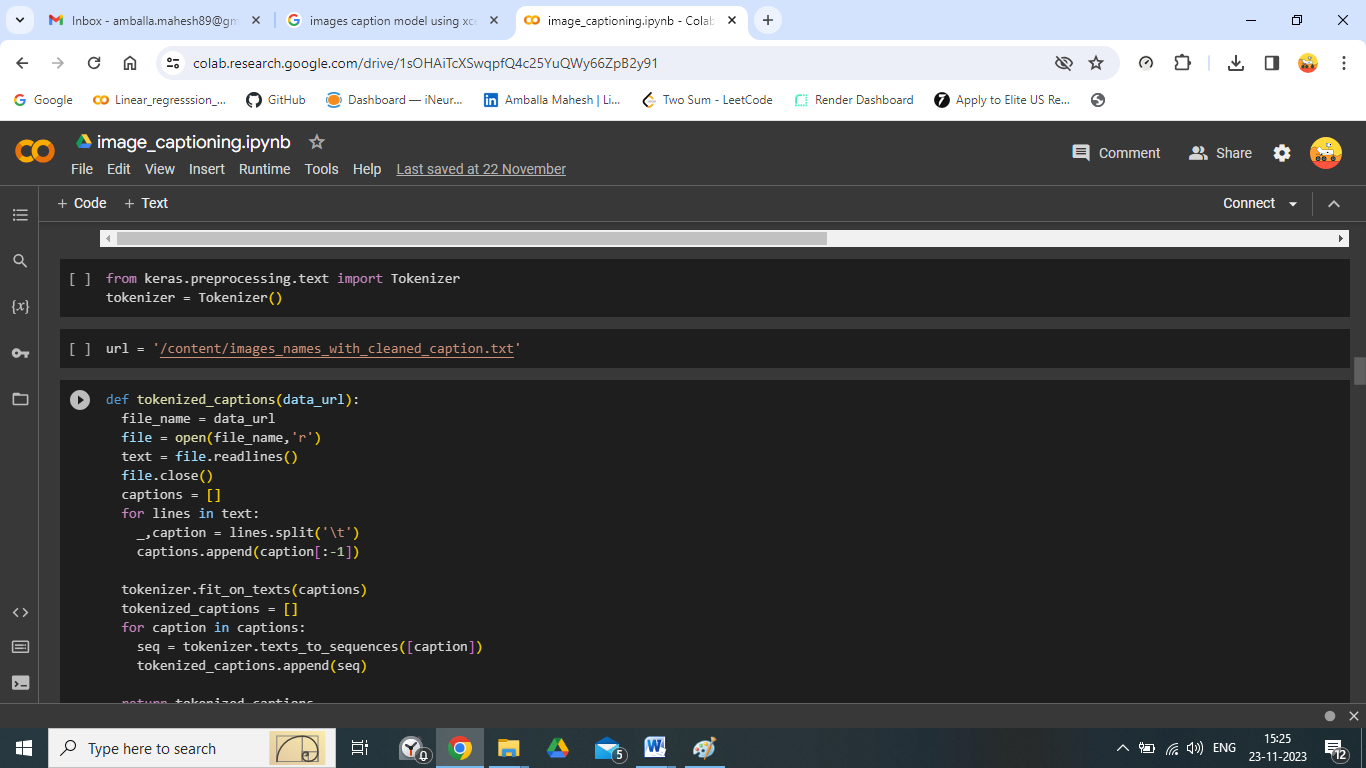
By the usage of the different data manipulation techniques we will remove unwanted words by this reduce the dimensions and make all the features in numerical data type using Keras tokenizer to get the vector of caption and Keras Xception model to get features from images.

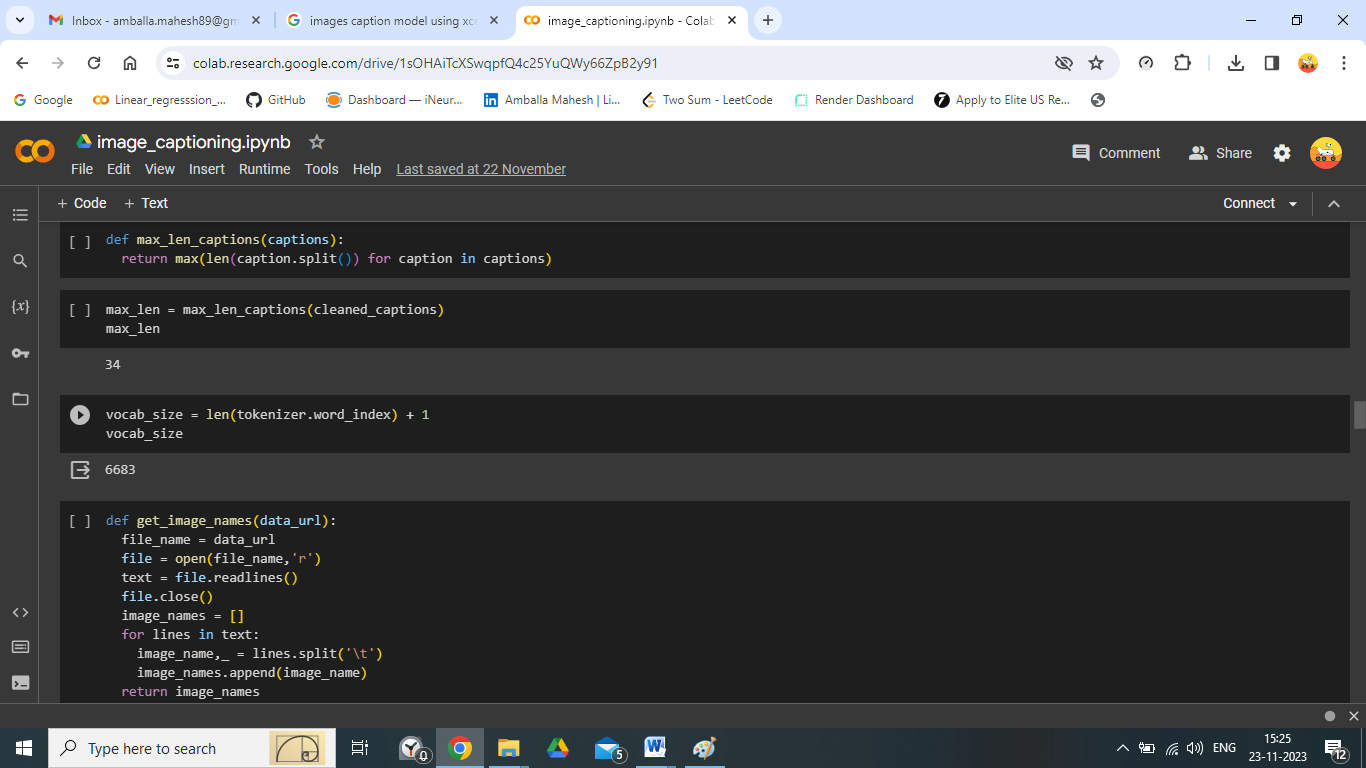


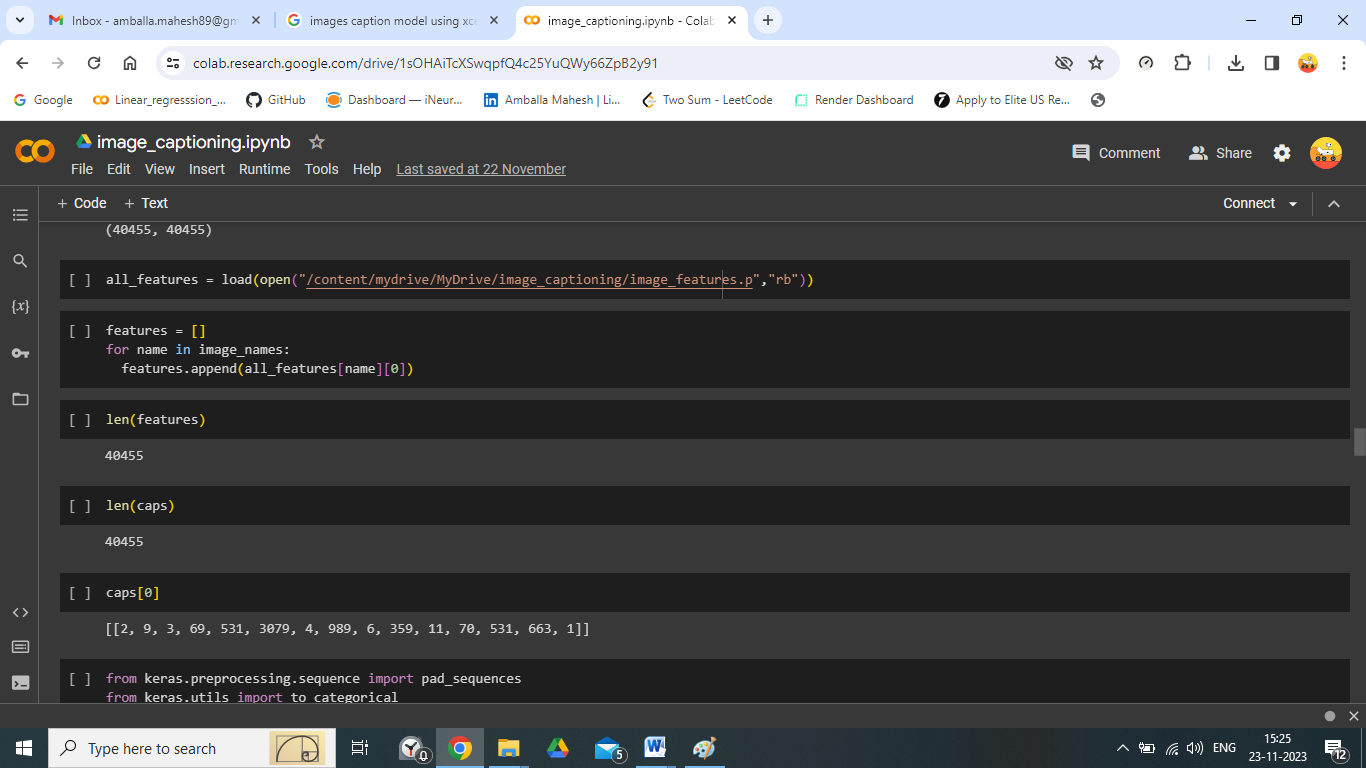




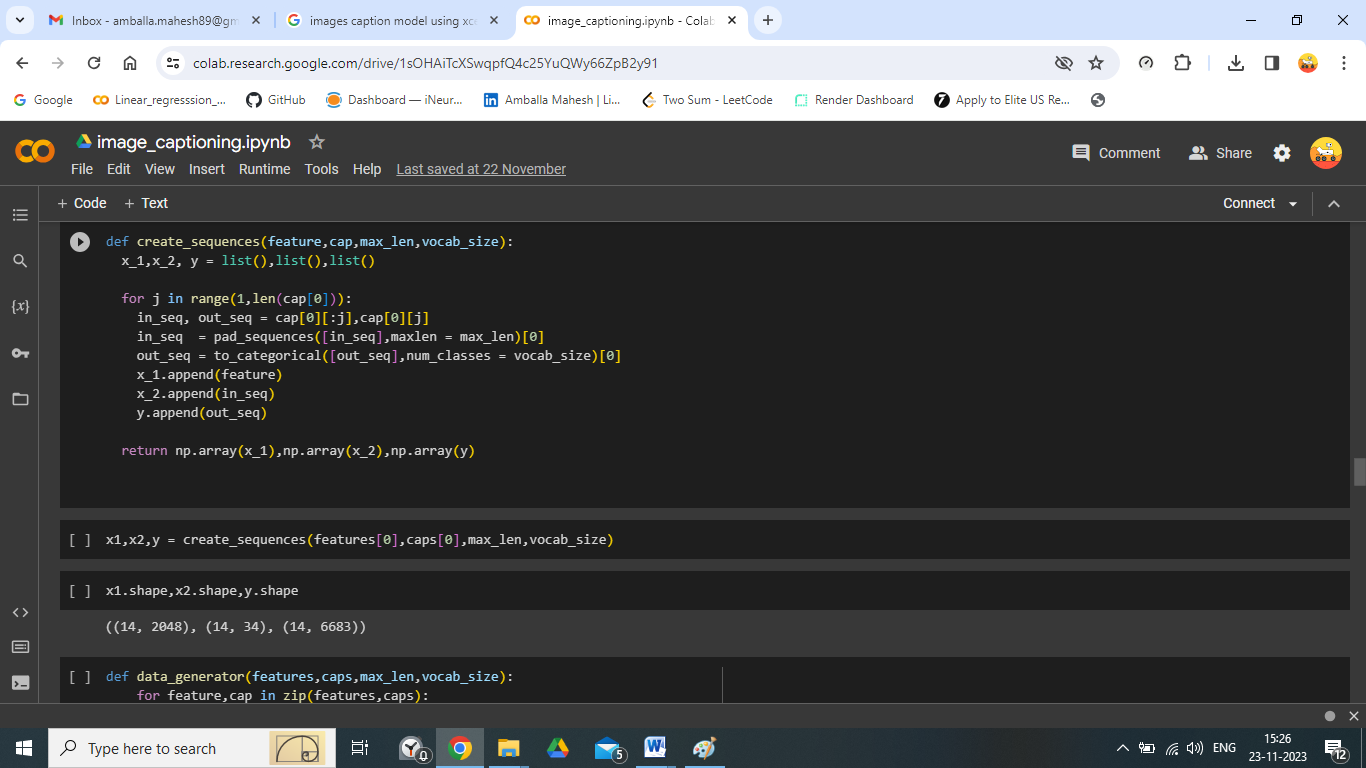


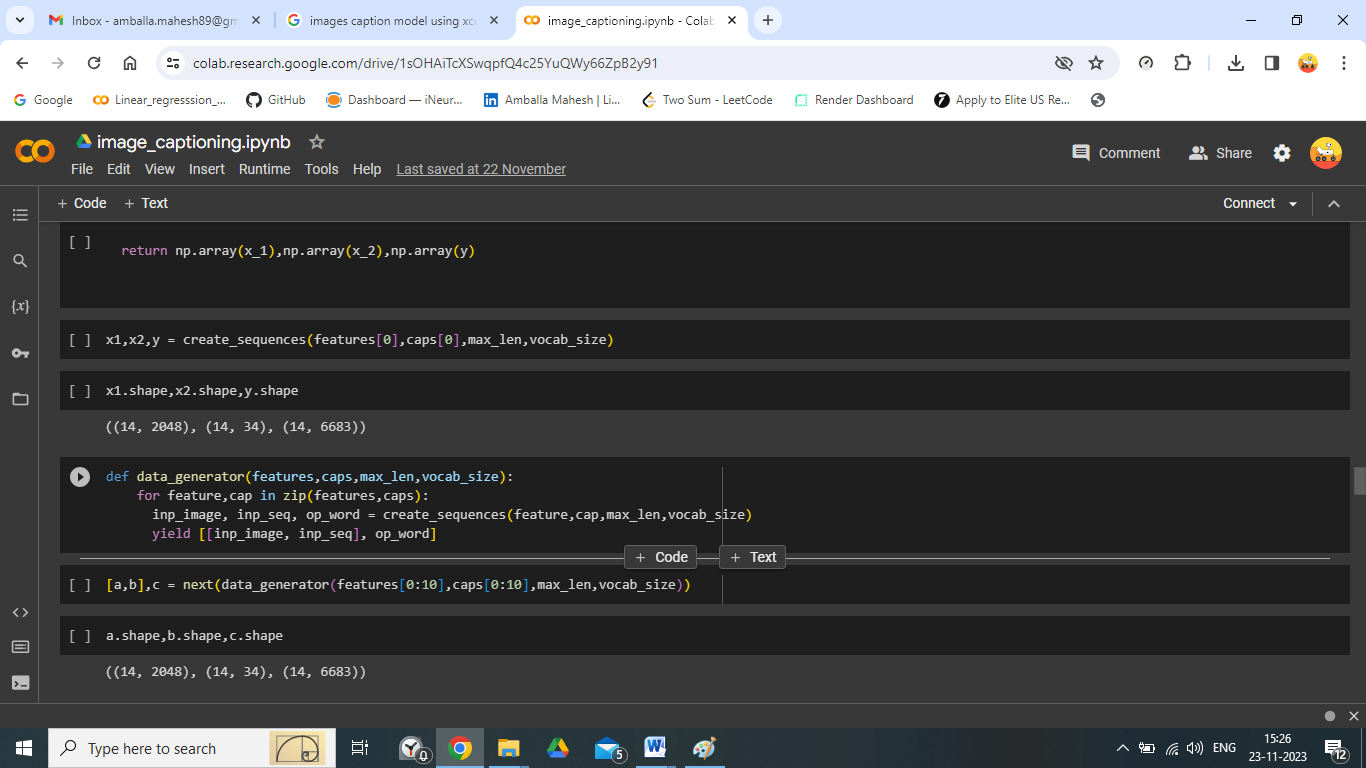






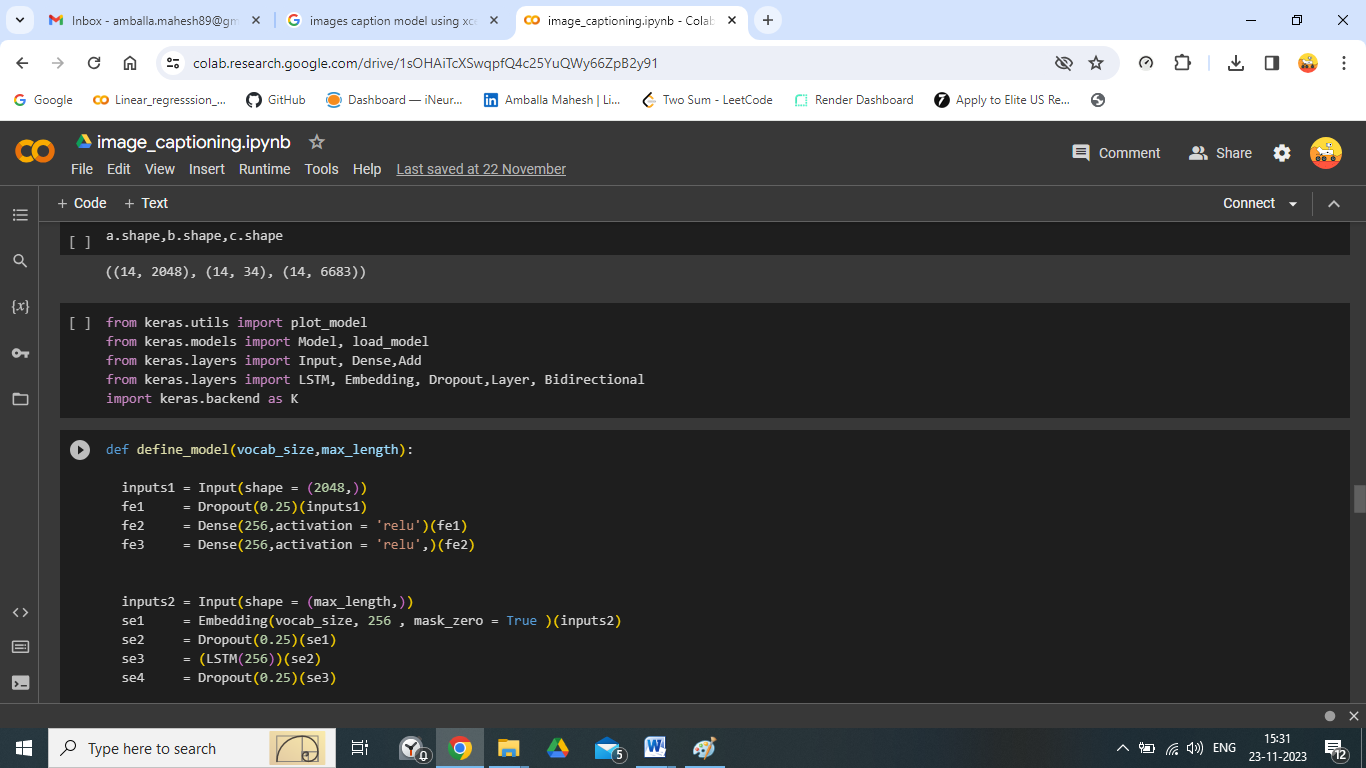
# Creating Input and Output data

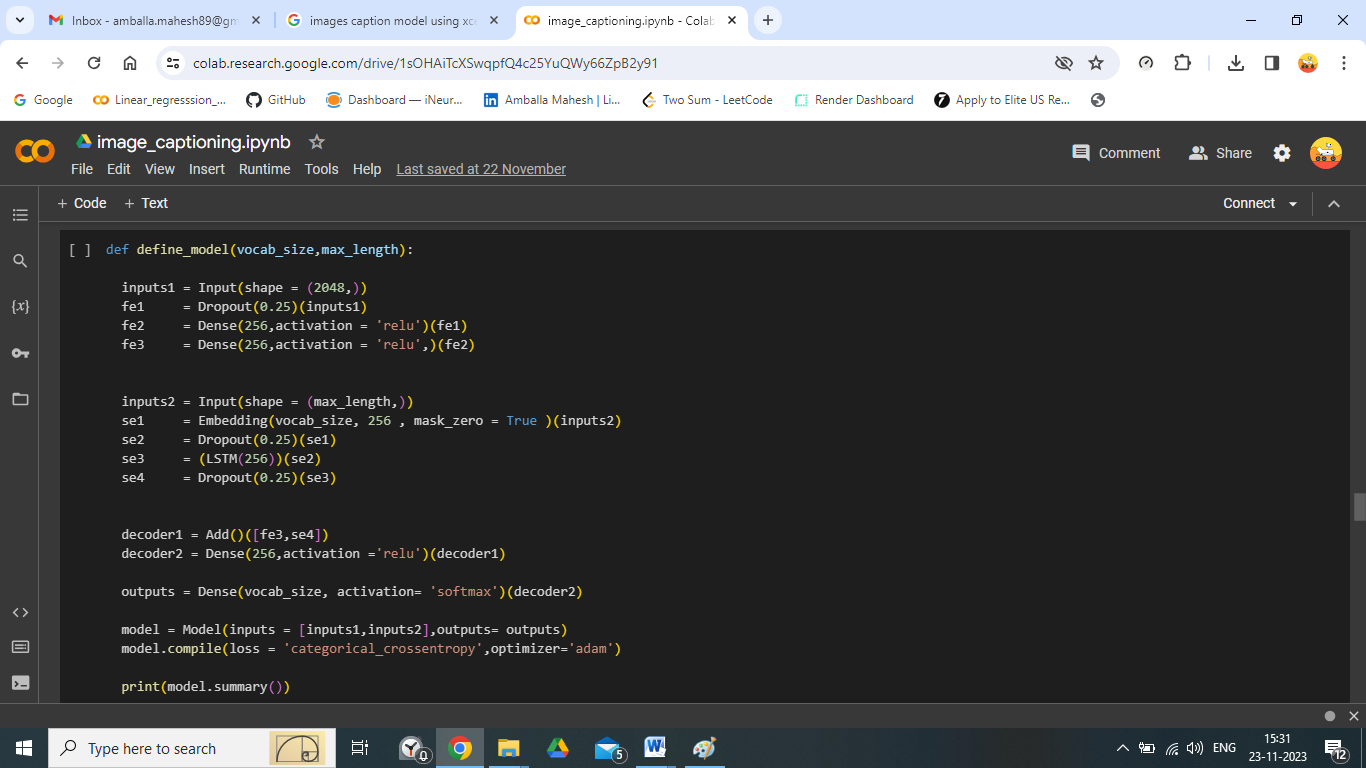


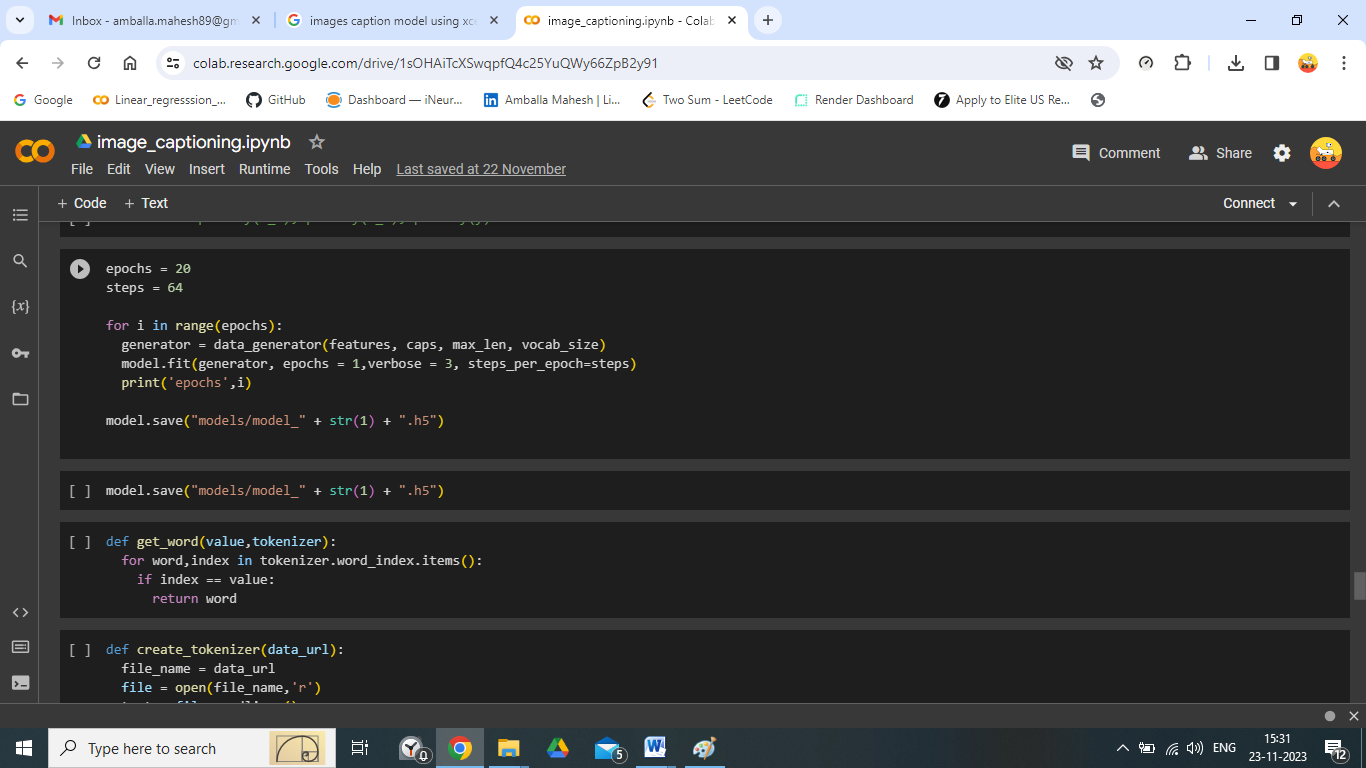


# Model Building

We create CNN + RNN base model using Keras as mentioned below.

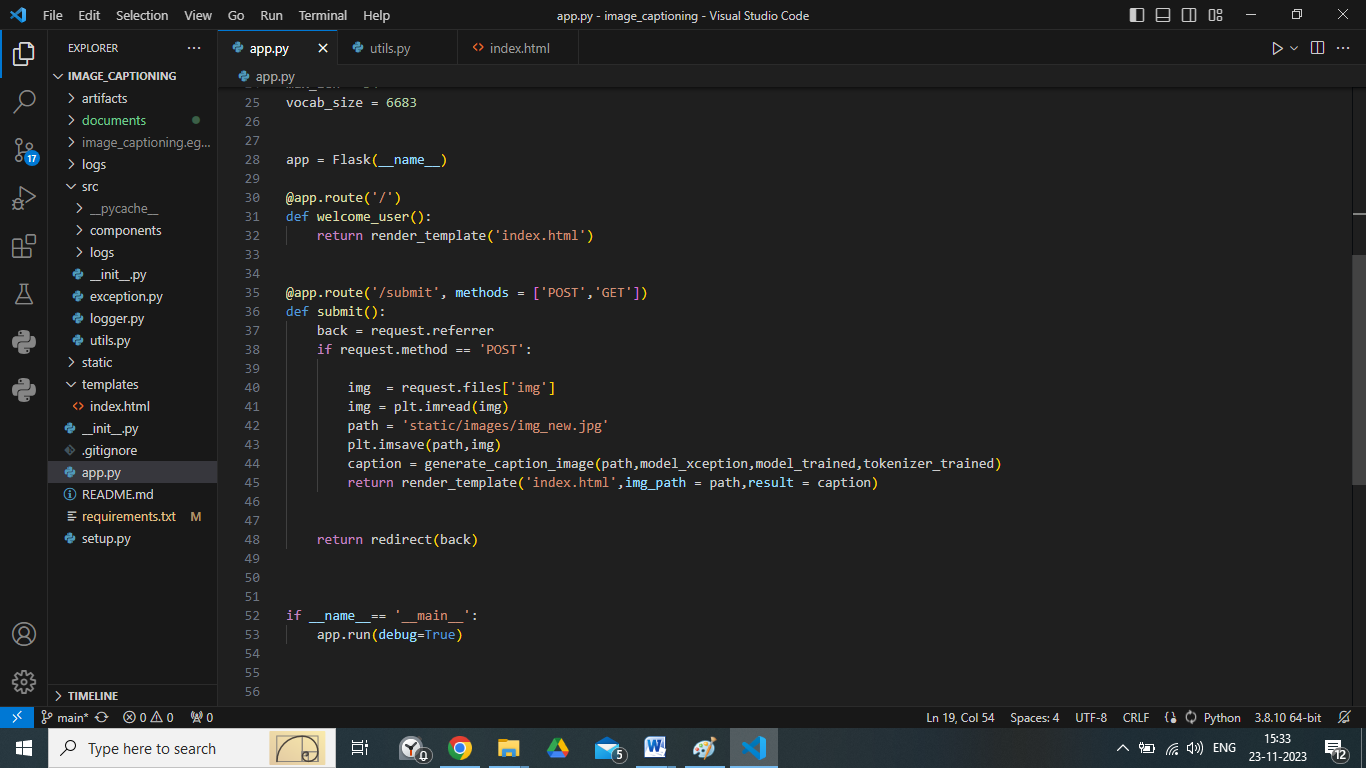


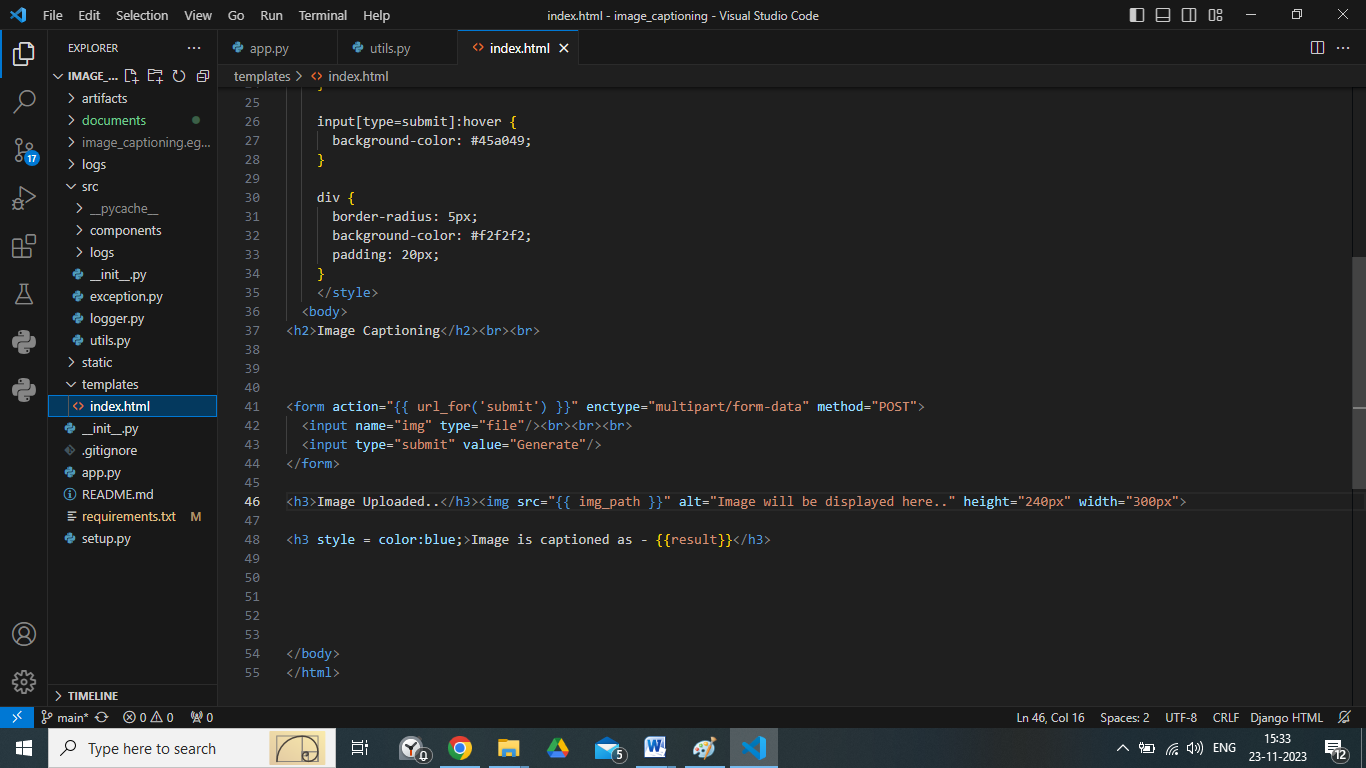




# Create Front End User Module using flask

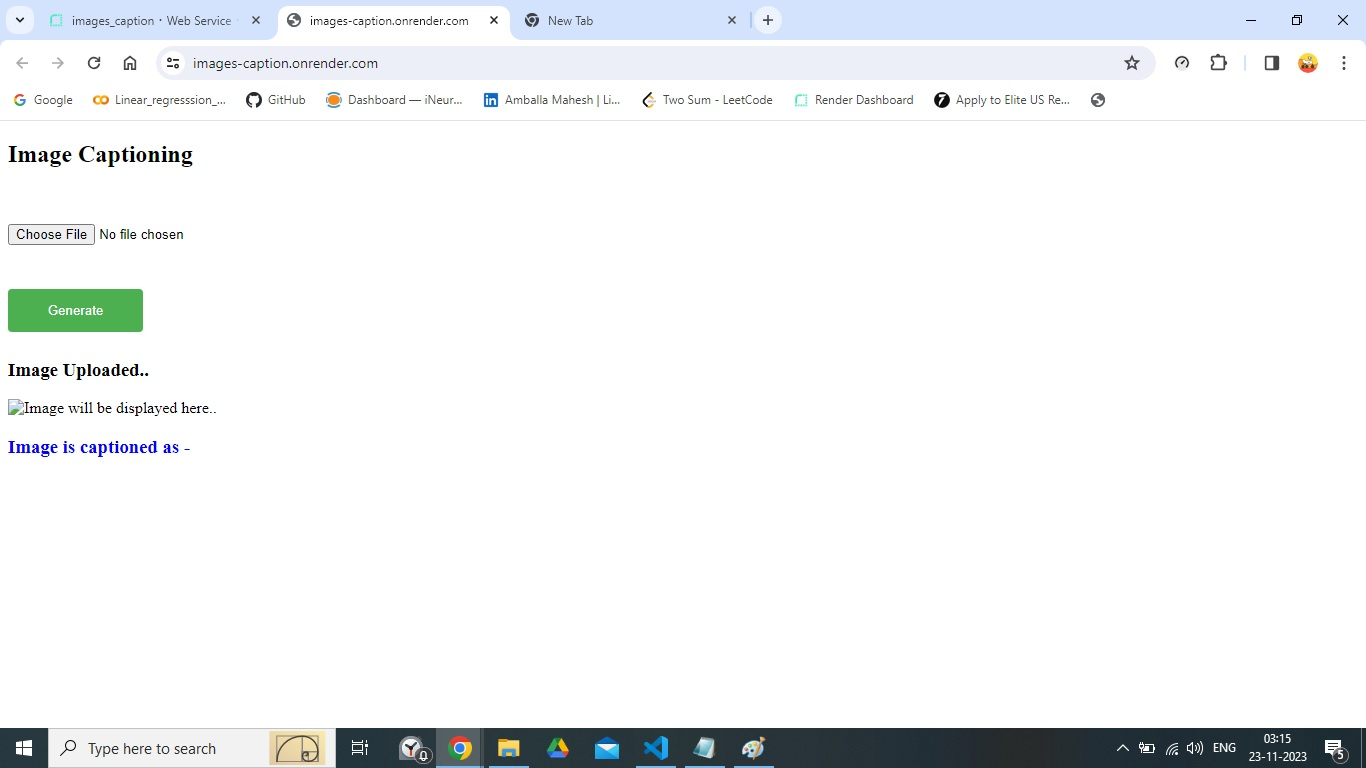
Once the model is created download and save the model and now we create GUI for front end user using the flask incorporated with HTML, CSS. Align and map the user data to the data base created. From user data create the data frame and load it to the model for the prediction the same prediction is send back to the user GUI and well saved in the data base (MySQL and Cassandra).

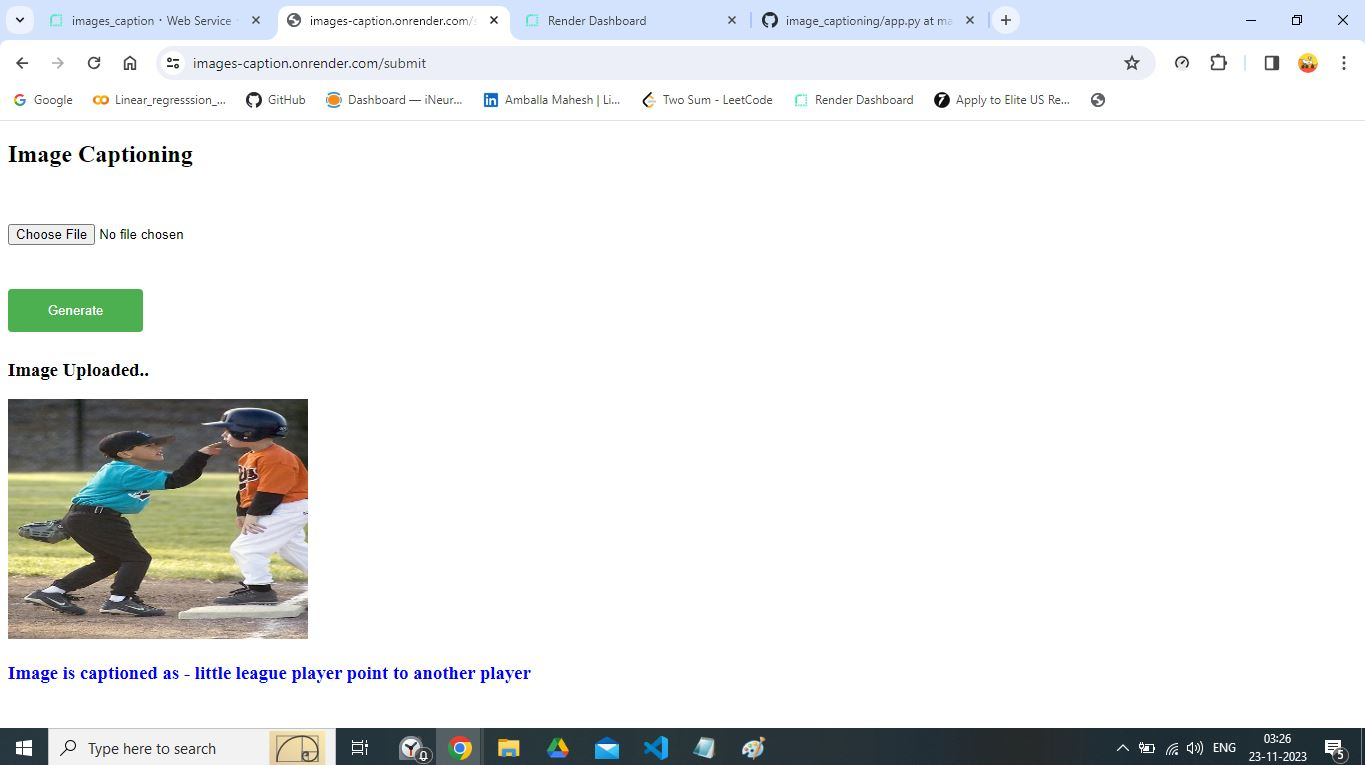


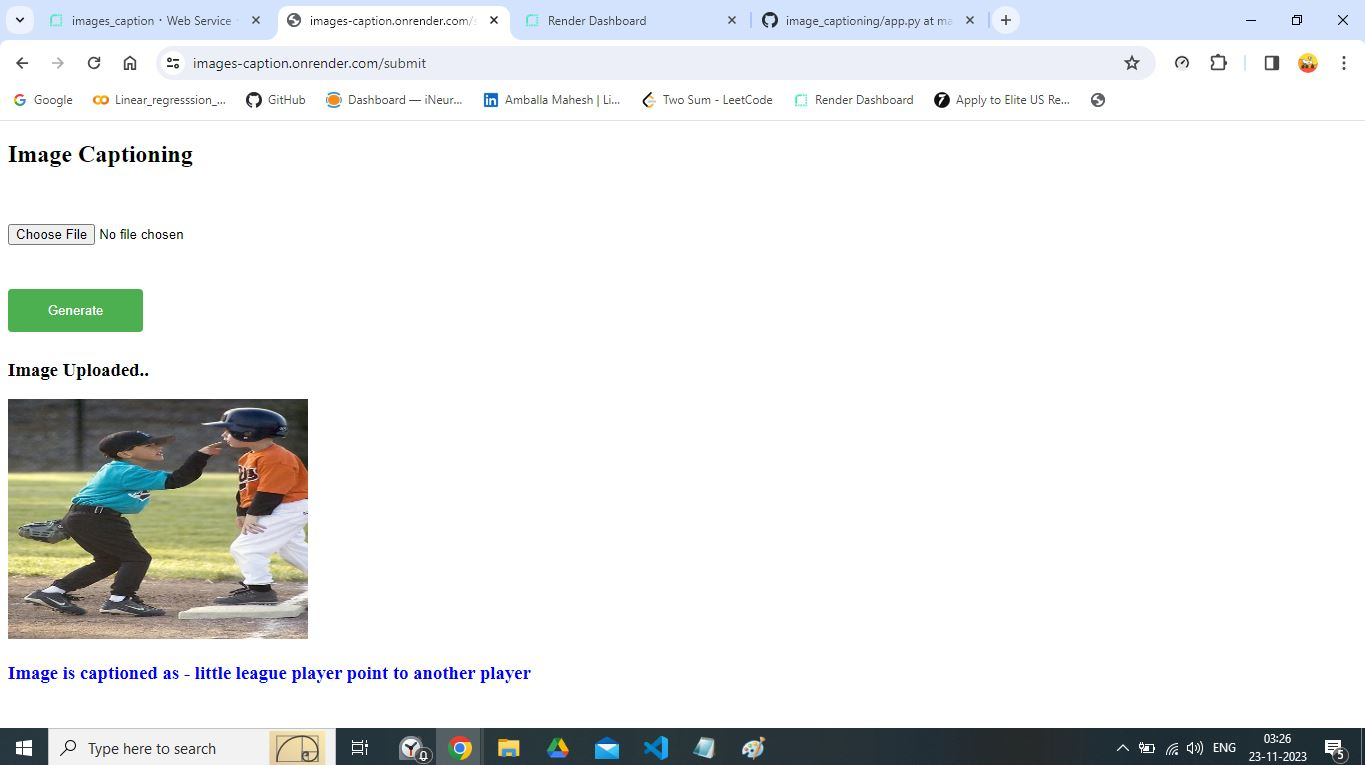


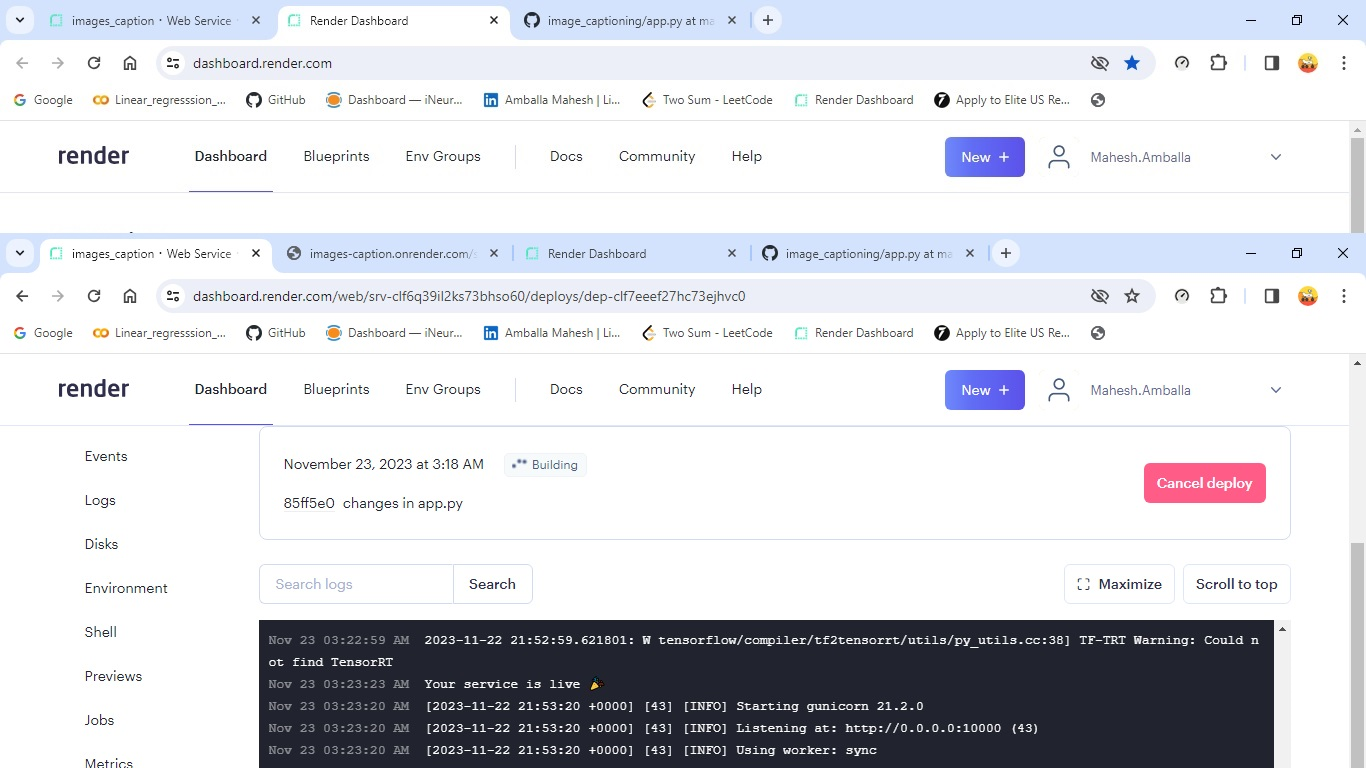
# Testing the Model

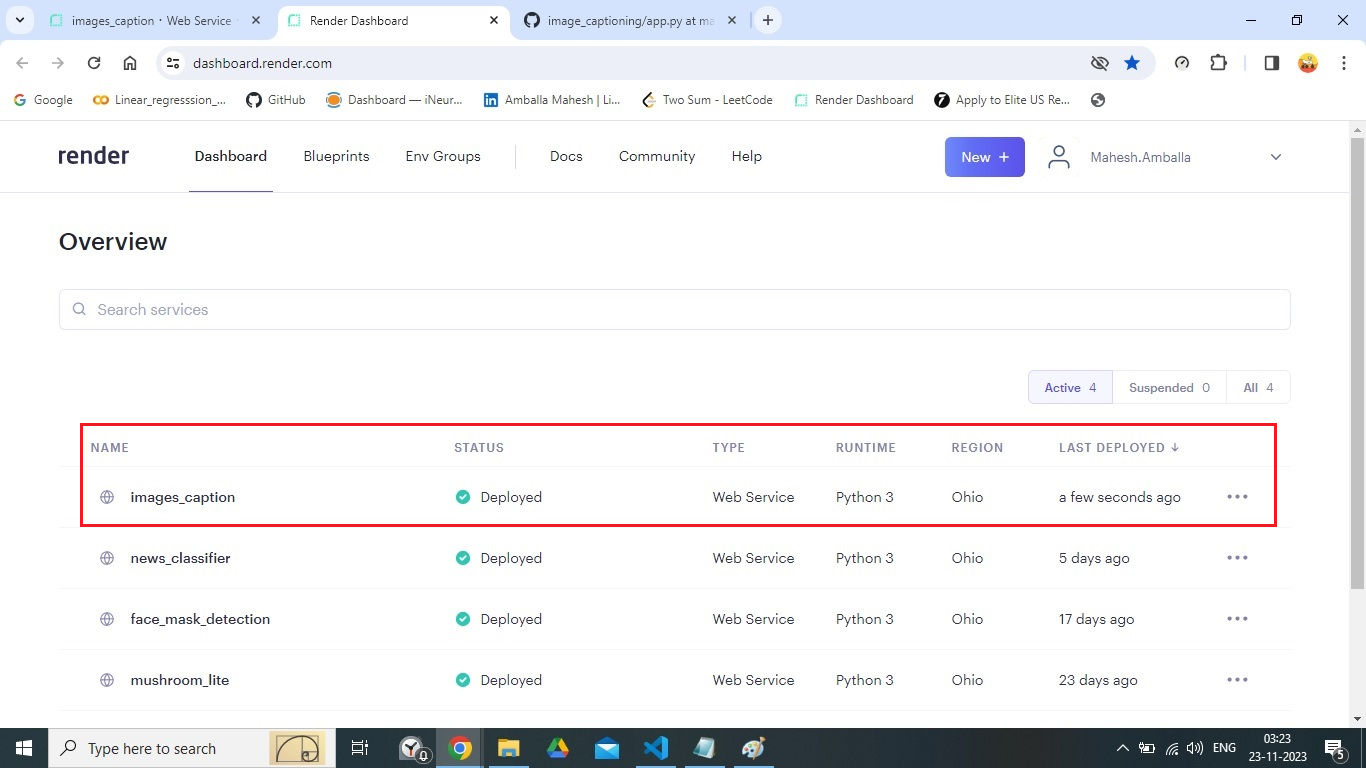
* Verify whether the application is the loading on the local server instance.
* Verify whether the user can access the application.
* Verify the user can access the different fields for selection and can be visible
* Once the user selection the fields and made the submit
* Check the user can get the result or prediction.
* Once he gets the prediction.
* Check the data form the user and prediction from the model is loaded into the local MySQL and Cassandra Data base.
* Verify whether the application is the loading on the web service instance.
* Check the database and download the data…











HOST WEB ADDRESS: <https://image-captioning-iwhw.onrender.com>

# Key performance indicators (KPI)

* + Time and work load reduction by using the flask model.
  + Compare the accuracy of model using prediction and actual results.
  + Check for the wrong predictions
  + If found any wrong predictions again train the model with the new data along with previous data
  + Retest the model unless the productions attain the good results.